WenQuanYi Micro Hei [Scale=0.9]WenQuanYi Micro Hei Mono song-WenQuanYi Micro Hei sfWenQuanYi Micro Hei "zh" = 0pt plus 1pt

ãĂŁLPython CookbookãĂŃçňňäÿL’çL’Ĺ

åŔŚåÿČ 2.0.0

çĘŁèČ¡

9ælJĹ 27, 2017

Contents

Contents:

CHAPTER 1

Copyright

CHAPTER 2

åL’■èÍĂ

éąźçŻőäÿżéąţ

https://github.com/yidao620c/python3-cookbook

èŕŚèĂĚçŽĎèŕİ

äžžçŤ§èŃęç§■ïijŇæĹŚçŤÍPythonïijĄ

èŕŚèĂĚäÿĂçŻť åİŽæŇĄä¡£çŤÍPython3ïijŇåŻääÿžåőČäżčèąÍäžĘPythonçŽĎæIJłæİěãĂĆèŹ¡çĎűåŘŚåŘŐ
èĂŇäÿŤPython3çŽĎæIJłæİěéIJĂèęĄæŕŔäÿłäžžçŽĎåÿőåŁl’åŠŇæŤŕæŇĄãĂĆ
çŻőåL’■åÿĆéİćäÿŁçŽĎæŢŹçÍŃäźęçś■ïijŇç¡ŚäÿŁçŽĎæL’ŃåĘŇåď ğéČÍåĹĘå§žæIJňéČ¡æŸŕ2.xçşżåĹŮçŽĎïij

æIJĂè£ŚçIJŃåĹřäÿĂæIJňãĂŁPython CookbookãĂŃ3rd Edi-
tionïijŇåőŇåĚÍå§žäžŐPython3ïijŇåĘŹçŽĎäź§å¿Ĺäÿ■éŤŹãĂĆ
äÿžäžĘPython3çŽĎæŹőåŘŁïijŇæĹŚäź§äÿ■èĞłéĞŔåŁŻïijŇæČşåĄŽçĆźäżĂäźĹäžŃæČĚãĂĆäžŐœŸŕäźŐïijŃ
è£Źäÿ■æŸŕäÿĂéąźè¡żæİ¿çŽĎåůěä¡IJïijŇå■ťæŸŕäÿĂäżűåĂijå¿ŮåĄŽçŽĎåůěä¡IJïijŽäÿ■äżĚæŰźä¿£äžĘåĹńä

èŕŚèĂĚäijŽåİŽæŇĄåŕźèĞłåůśæŕŔäÿĂåŘěçŽĎç£żèŕŚèť§èť čïijŇåŁŻæśĆénŸèť íéĞŔãĂĆä¡ĘåŘŮèČ¡åŁŽ
åęĆæđIJèŕŚæŰĞäÿ■æIJL’äżĂäźĹéŤŹæijŔçŽĎåIJřæŰźèŕůåď ğåőűèğĄèřĚïijŇäź§æňćè£Őåď ğåőűéŽŘæŮűæŇ
yidao620@gmail.com

ä¡IJèĂĚçŽĎèŕİ

èĞłäżŐ2008åźť äžěæİěïijŇPython3æĺłçľ žåĞžäÿŰåźűæĚćæĚćè£ŻåŇŰãĂĆPython3çŽĎætĄèąŇäÿĂçŻť
äžŇåőđäÿŁïijŇåĹřæĹŚåĘŹè£ŹæIJňäźęçŽĎ2013åźť ïijŇçżİåď ğéČÍåĹĘçŽĎPythonçÍŃåžŘåŚŸäż■çĎűåIJÍçŤ§

æIJĂäÿżèęĄæŸŕåŽăäÿžPython3äÿ■åŘŚåŘŐåĚijåőźãĂĆæŕńæŮăçŰŚéŮőïijŇåŕźäžŐåůëä¡IJåIJĺéĄŮçŢŹäżčçă
ä¡ĘæŸŕæŤ¿çIJijæIJłæİèïijŇä¡ăåŕśäijŽåŘŚçŐřPython3çżŹä¡ăåÿęæİèäÿ■äÿĂæăůçŽĎæČŁåŰIJãĂĆ

æ■čåęĆPython3äżčèąĺæIJłæİèäÿĂæăůïijŇæŰřçŽĎãĂŁPython Cook-
bookãĂŃçĽĹæIJňçŻÿæŕŤè¿ČäźŃåĽ■çŽĎçĽĹæIJňæIJĽäžĘäÿĂäÿłåĚĺæŰřçŽĎæŤźåŘŸãĂĆ
æIJĂéĞ■èęĄçŽĎæŸŕïijŇè£ŹäÿłæĎŘåŚşçİĂæIJňäźęæŸŕäÿĂæIJńéİđåÿÿåĽ■æš£çŽĎåŘĆèĂČäźęãĂĆäźęäÿ■
åźűæšąæIJĽèĂČèŹŚäźŃåĽ■èĂĄçĽĹæIJňçŽĎåĚijåőźæĂğïijŇäź§æšąæIJĽæăĞæşĺæŮğçĽĹæIJňäÿŃçŽĎèğ
ä¡ĘæŸŕæĹŚäžňæIJĂçżĹçŽĎçŽőçŽĎæŸŕåĘŹäÿĂæIJňåőŇåĚĺå§žäžŐæIJĂæŰřæIJĂåĚĹè£ŽåůèåĚůåŠŇèŕ■ĕĹ
åÿŇæIJŻè£ŹæIJňäźęèČ¡æĹŘäÿžåIJĺPython3äÿŃçijŰçăĄåŠŇæČşå■ĞçžğäźŃåĽ■éĄŮçŢŹäżčçăĄçŽĎçĺŃåžŘ

æŕńæŮăçŰŚéŮőïijŇçijŰåĘŹäÿĂæIJňè£ŹæăůçŽĎäźęäijŽåĘŠäÿĂåőŽçŽĎçijŰè¿ŚéčŐéŹľãĂĆåęĆæđIJå
æŕŤåęĆActiveStateâĂŹs Python recipesæĹŰèĂĚStack Over-
flowïijŇä¡ĘæŸŕçżİåď ğéČĺåĹĘéČ¡åůšçżŔæŸŕè£ĞæŮűçŽĎäžĘãĂĆ
è£ŹäžŻæŢŹçĺŃéŹď äžĘæŸŕå§žäžŐPython2çijŰåĘŹäźŃåď ŰïijŇåŔŕèČ¡è£ŸæIJĽå¿Ĺåď Žèğčå ĘşæŰźæą ĹåIJ
åŔęåď ŰïijŇåőČäžňè£ŸäijŽçżŔåÿÿä¡£çŤĺäÿĂäžŻè£ĞæŮűçŽĎæĹĂæIJŕïijŇè£ŹäžŻåůšçżŔåĘĚç¡őåĹřPython

è£ŹæIJňäźęçŽĎæĽĂæIJĽäÿżécŸéČ¡æŸŕå§žäžŐåůšçżŔå■ŸåIJĺçŽĎäžčçăĄåŠŇæĹĂæIJŕïijŇèĂŃäÿ■æ
åIJĺåŐ§æIJĽäžčçăĄå§žçąĂäÿŁïijŇæĹŚäžňåőŇåĚĺä¡£çŤĺæIJĂæŰřçŽĎPythonæĹĂæIJŕåŐżæŤźéĂããĂĆ
æĽĂäžěïijŇäżżä¡ŢæČşä¡£çŤĺæIJĂæŰřæĹĂæIJŕçijŰåĘŹäžčçăĄçŽĎçĺŃåžŘåŚŸïijŇéČ¡åŔŕäžěårĘæIJňäźęå¡Ş

åIJĺèőĺèőžçŽĎäÿżécŸéĂĽæŃľæŰźéİćïijŇæĹŚäžňäÿ■åŔŕèČ¡åŻŁæŃňPythonécĘå§§æĽĂæIJĽçŽĎäÿIJ
åŻăæ■ď ïijŇæĹŚäžňäijŸåĚĹéĂĽæŃľäžĘPythonèŕ■èĺĂæäÿå£ČéČĺåĹĘïijŇäżěåŘŁäÿĂäžŻåIJĺåijĂåŘŚäÿ■åÿ
åŔęåď ŰïijŇè£ŹéĞŃèőĺèőžçŽĎå¿Ĺåď ŽæŁĂæIJŕéČ¡æŸŕPython
3æIJĂæŰřæĽ■åĞžçŐřçŽĎïijŇæĽĂäžěåęĆæđIJåůëä¡IJåIJĺPythonèĂĄçĽĹæIJňäÿŃïijŇ
å■şä¿£æŸŕæIJĂæIJĽçżŔéłŇçŽĎçĺŃåžŘåŚŸåŔŕèČ¡äźŃåĽ■äźś§æšąèğĄè£Ğè£ŹäžŻäÿIJèě£ãĂĆ
åŔęåď ŰïijŇè£ŹäžŻçď žä¿ŃçĺŃåžŘäź§äijŽåĄŘåŘŚäžŐåśŢçď žäÿĂäžŻæIJĽçŤĺçŽĎçijŰçĺŃæŁĂæIJŕ(æŕŤåę
èĂŃäÿ■æŸŕäžĚäžĚåőŽä¡■åIJĺäÿĂäžŻåĚůä¡ŞçŽĎéŮőécŸäÿŁãĂĆår¡çőąäź§æŔŘåŔŁåĹřäžĘæIJĽäÿĂäžŻçńň

è£ŹælJňäźęéĂĆåŘĹèřĄ

è£ŹæIJňäźęçŽĎçŽőæăĞèŕżèĂĚæŸŕéĆčäžŻæČşæůšåĚěçŘĘèğčPythonèŕ■èĺĂæIJžåĹůåŠŇæIJĂæŰřçijŰ
å¿Ĺåď Žèőĺèőžéč¡æŸŕæăĞåĞĘåžŞïijŇæąĘæđűåŠŇåžŤçŤĺçĺŃåžŘä¡£çŤĺåĹřçŽĎéńŸçžğæŁĂæIJŕãĂĆ
æIJňäźęæĽĂæIJĽçď žä¿ŃåİĞåĄĞèő¿èŕżèĂĚåůšçżŘæIJĽäžĘäÿĂåőŽçŽĎçijŰçĺŃèČŇæŹŕåźűäÿŤåŘŕäžěå¿Ĺ
(æŕŤåęĆå§žæIJňçŽĎèőąçőŮæIJžçğŚå■ęç§ěèŕĘïijŇæŢŕæ■őçżŞæđĎç§ěèŕĘïijŇçőŮæşŢåď ■æİĆåžęïijŇçşžçž§
åŔęåď ŰïijŇæŕŘäÿłçď žä¿ŃéČ¡åŘłæŸŕäÿĂäÿłåĚééŮĺæŇĞåŕijïijŇåęĆæđIJèŕżèĂĚæČşæůšåĚěçăŤçľ űïijŇéIJĂ
åŻăæ■ď ïijŇæĹŚäžňåĄĞåőŽèŕżèĂĚåŘŕäžěå¿ĹçĘ§çżČçŽĎä¡£çŤĺæŘIJçť ćåijŢæŞŐäžěåŘŁç§ěéĄŞæĂŐæăůæ

è£ŹæIJňäźęäÿ■éĂĆåŘĹPythonçŽĎåĹİå■ęèĂĚãĂĆäžŃåőđäÿŁïijŇæIJňäźęåůšçżŘåĄĞåőŽäžĘèŕżèĂĚåů
æIJňäźęäź§äÿ■æŸŕéĆčçğ■å£ńéĂ§åŘĆèĂČæĽŃåĘŃ(åŘŕäžěå¿Ĺå£ńçŽĎæ§ěèŕćæ§ŘäÿłæĺąåİŮäÿŃçŽĎæ§Ř
æIJňäźęæŮĺåIJĺèĄŽçĎęåĞääÿłæIJĂéĞ■èęĄçŽĎäÿżécŸïijŇæijŢçď žåĞăçğ■åŘŕèČ¡çŽĎèğčåĘşæŰźæąĹïijŇä
ä¡ăåŘŕäžěçżŘæ■ď è£ŽåĚěäÿĂäžŻæŻť énŸçžğçŽĎäÿżécŸïijŇè£ŹäžŻåŘŕäžěåIJĺç¡Śäÿ ŁæĹŰèĂĚåŘĆèĂČæĽ

ælJňäźęçď žä¿ŃäžčçăĄ

æIJňäźęåĞääźŐæĽĂæIJĽæžŘäžčçăĄåİĞåŘŕäžěåIJĺ http://github.com/dabeaz/
python-cookbook äÿŁéİćæĽ¿åĹřãĂĆ ä¡IJèĂĚæńćè£ŐåŘĎä¡■ä£őæ■čbugïijŇæŤźè£ŽäžčçăĄåŠŇèŕĐèőžãĂĆ

æIJňäźęåřśæŸŕåÿőåŁľä¡ăåőŇæĹŘä¡ăçŽĎåůëä¡IJãĂĆäÿĂèĹňæİèèőšïijŇåŘłèęĄåIJĺæIJňäźęäÿŁéİćçŽĎå
ä¡ăéČ¡åŘŕäžěéŽŘæŮűæŃ£è£ĞåŐżåIJĺä¡ăçŽĎæžŘçăĄåŠŇæŰĞæąčäÿ■ä¡£çŤĺãĂĆä¡ăäÿ■éIJĂèęĄåŘŚæĹŚäž
éŹď'éIđä¡ăæŁĎèć■çŽĎåď'łè£ĞåĹĘäžĘãĂĆæŕŤåęĆèŕť'åď'■åĹűåĞääÿłäžčçăĄçL'Ğæőţå0żåőŇæĹŘäÿĂäÿłçĺŃ
èť'l'å■ŰæĹŰèĂĚåĹĘåŘŚåőđä¿Ńäžčçă ĄçŽĎåĚL'çŻŸäź§äÿ■éIJĂèęĄèőÿåŕŕijŇåijŢçŤĺæIJňäźęåŠŇåőđä¿Ńä
ä¡ĘæŸŕŕijŇåŘĹåźűåď'ğéĞŘçŽĎäžčçăĄåĹŕä¡ăçŽĎæ■čåijŔäžğåŞĄæĹŰæŰĞæąčäÿ■åŐżå£Ěéążå¿ŮåĹřæĹŚä

æĹŚäžňäÿ■äijŽèęĄæśĆä¡ăæůżåŁăäžčçăĄçŽĎåĞžåď'ĎïijŇåŇĚæŃňæăĞéćŸïijŇä¡IJèĂĚïijŇåĞžçL'Ĺçď'
æŕŤåęĆïijŽPython Cookbook, 3rd edition, by David Beazley and Brian K. Jones (OâĂŹReilly). Copyright 2013 David Beazley and Brian Jones, 978-1-449-34037-7.
ä¡ĘæŸŕåęĆæđIJä¡ăè£ŹäźĹåĄŽäžĘïijŇæĹŚäžňäijŽå¿ĹæĎ§æ£ĂçŽĎãĂĆ

èĄŤçşżæĹŚäžň

èŕůåřĘåĚşäžŐæIJňäźęçŽĎèŕĎèőžåŠŇéŮőécŸåŘŚéĂĄçżŹåĞžçL'Ĺçď'¿ïijŽ

OâĂŹReilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

æIJňäźęç¡ŚçńŹ: http://oreil.ly/python_cookbook_3e ïijŇäÿŁéİćæIJL'åŃŸèŕŕèąĺïijŇçď'žä¿ŃåŠŇäÿĂäžŻå

åęĆæđIJæČşèęĄèŕĎèőžæĹŰèĂĚæŸŕéŮőäÿĂäÿŇæIJňäźęæŁĂæIJŕæŰźéİćçŽĎéŮőécŸïijŇ
èŕůåŘŚéĂĄéĆőäžűèĞşïijŽ bookquestions@oreilly.com

æŻť'åď'ŽåĚşäžŐæĹŚäžňçŽĎäźęçś■ïijŇèőĺèőžäijŽïijŇæŰřéŮżïijŇ
èŕůèő£éŮőæĹŚäžňçŽĎç¡ŚçńŹïijŽ http://www.oreilly.com

åIJĺFacebookäÿŁæ§ěæL'¿æĹŚäžň: http://facebook.com/oreilly

åIJĺTwitteräÿŁåĚşæşĺæĹŚäžň: http://twitter.com/oreillymedia

åIJĺYouTubeäÿŁèğĆçIJŃæĹŚäžň: http://www.youtube.com/oreillymedia

æĎ§èřć

æĹŚäžňçŤśèąůçŽĎæĎ§èřćæIJňäźęçŽĎæŁĂæIJŕåőąææäÿèĂĚJake Vanderplas, Robert Kern åŠŇ Andrea CrottiçŽĎéİđåÿÿæIJL'çŤĺçŽĎèŕĎèőžåŠŇåžžèőőïijŇ
è£ŸæIJL'Pythonçď'¿åŇžçŽĎåÿőåŁľåŠŇéijŞåŁśãĂĆæĹŚäžňè£ŸæČşæĎ§èřćäÿŁäÿĂäÿłçL'ĹæIJňçŽĎçijÚè¿Ś

Vanderplas, Robert Kern,and Andrea CrottiãĂĆ åř¡çőąè£ŹäÿłçĽĹæIJňæŸŕæIJĂæŰřçŽĎïijŇä¡ĘæŸŕåĽ■äÿĂä
æIJĂåŘŐïijŇæIJĂæIJĂéĞ■èęĄçŽĎåřśæŸŕïijŇæĹŚäžňèęĄæĎ§èřćæĽĂæIJL'écĎèğĹçĽĹæIJňçŽĎèŕżèĂĚïij

CHAPTER 3

çňňäÿĂçńăïijŽæŢřæ■őçżŞæđĎåŠŇçőŮæşŢ

PythonæŔŘä¿ŻäžĘåď ğéĞŔçŽĎåĘĚç¡őæŢřæ■őçżŞæđĎïijŇåŇĚæŃňåĹŮèąĺïijŇéŻĘåŘĹäžěåŔŁå■ŮåĚý
ä¡ĘæŸŕïijŇæĹŚäžňäź§äijŽçżŔåÿÿçćřåĹřěŕÿåęĆæ§ěèŕćïijŇæŐŠåžŔåŠŇè£Ğæżď ç■L’ç■L’è£ŹäžŻæŹőéĄ■
åŻăæ■ď ïijŇè£ŹäÿĂçńăçŽĎçŻőçŽĎåřśæŸŕèőĺèőžè£ŹäžŻæŕŤè¿Čåÿÿè ğĄçŽĎéŮőécŸåŠŇçőŮæşŢãĂĆ
åŘęåď ŰïijŇæĹŚäžňäź§äijŽçżŹåĞžåIJĺéŻĘåŘĹæĺąåİŮ `collections`
å¡Şäÿ■æŞ■ä¡IJè£ŹäžŻæŢřæ■őçżŞæđĎçŽĎæŰźæşŢãĂĆ

Contents:

1.1 èğčåŐŃåžŔåĹŮèţŃåĂijçżŹåď ŽäÿłåŔŸéĞŔ

éŮőécŸ

çŐřåIJĺæIJL’äÿĂäÿłåŇĚåŘńNäÿłåĚČçt’ăçŽĎåĚČçżĎæĹŰèĂĚæŸŕåžŔåĹŮïijŇæĂŐæăůårĘåőČéĞŇéİć

èğčåĘşæŰźæąĹ

äżżä¡ŢçŽĎåžŔåĹŮ(æĹŰèĂĚæŸŕåŔŕè£■äżčårźèśą)åŔŕäżěéĂŽè£ĞäÿĂäÿłçőĂå■ŢçŽĎèţŃåĂijèŕ■åŔèèğ
åŤŕäÿĂçŽĎåL’■æŘŘårśæŸŕåŔŸéĞŔçŽĎæŢřéĞŔå£Ěéążèů§åžŔåĹŮåĚČçt’ăçŽĎæŢřéĞŔæŸŕäÿĂæăůçŽĎãA

äżčçăĄçď žä¿ŃïijŽ

```
>>> p = (4, 5)
>>> x, y = p
>>> x
4
>>> y
5
```

```
>>>
>>> data = [ 'ACME', 50, 91.1, (2012, 12, 21) ]
>>> name, shares, price, date = data
>>> name
'ACME'
>>> date
(2012, 12, 21)
>>> name, shares, price, (year, mon, day) = data
>>> name
'ACME'
>>> year
2012
>>> mon
12
>>> day
21
>>>
```

åęĆæđIJåŔŸéĞŕäÿłæŢřåŠŇåžŕåĹŮåĚČçť ăçŽĎäÿłæŢřäÿ■åŇźéĚ■ïijŇäijŽäžğçŤ§äÿĂäÿłåijĆåÿÿãĂĆ

äžčçăĄçď žä¿ŃïijŽ

```
>>> p = (4, 5)
>>> x, y, z = p
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: need more than 2 values to unpack
>>>
```

èőĺèőž

åőđéŹĚäÿŁïijŇè£Źçğ■èğčåŐŃètŃåĂijåŔŕäžěçŤĺåIJĺäżża¡ŢåŔŕè£■äžčåŕźèśąäÿŁéİćïijŇèĂŇäÿ■äžĚäżĚæåŇĚæŃňå■ŮçňęäÿšïijŇæŰĞäžűåŕźèśąïijŇè£■äžčåŹĺåŠŇçŤ§æĹŘåŹĺãĂĆ

äžčçăĄçď žä¿ŃïijŽ

```
>>> s = 'Hello'
>>> a, b, c, d, e = s
>>> a
'H'
>>> b
'e'
>>> e
'o'
>>>
```

æIJL'æŮűåĂŹïijŇä¡ääŔŕèČ¡åŔłæČşèğčåŐŃäÿĂéČĺåĹĘïijŇäÿćåijČåĚűäžŰçŽĎåĂijãĂĆåŕźäžŐè£Źçğ■ä¡ĘæŸŕä¡ääŔŕäžěä¡£çŤĺäżżæĎŘåŔŸéĞŔåŘ■åŐżå■ää¡■ïijŇåĹřæŮűåĂŹäÿćæŐL'è£ŹäžŻåŔŸéĞŔåřśèąŇäžĘ

äžčçăĄçď žä¿ŃïijŽ

```
>>> data = [ 'ACME', 50, 91.1, (2012, 12, 21) ]
>>> _, shares, price, _ = data
>>> shares
50
>>> price
91.1
>>>
```

ä¡ăå£Ěéążä£ÌèŕĄä¡ăéĂĽ'çŤĺçŽĎéĆčäžŻå■ăä¡■åŔŸéĞŔåŘ■åIJĺåĚűäžŰåIJřæŰźæšąèćńä¡£çŤĺåĹřãĂĆ

1.2 èğčåŐŃåŘŕè£■äžčåŕźèśąèţŃåĂijçżŹåď'ŽäÿłåŔŸéĞŔ

éŮőécŸ

åęĆædIJäÿĂäÿłåŘŕè£■äžčåŕźèśąçŽĎåĚČçt'ääÿłæŢřèűĚè£ĞåŔŸéĞŔäÿłæŢřæŮűïijŇäijŽæŁŻåĞžäÿĂäÿł
`ValueError` ãĂĆ éĆčäźĹæĂŐæăůæĽ■èČ¡äžŐè£ŹäÿłåŘŕè£■äžčåŕźèśąäÿ■èğčåŐŃåĞžNäÿłåĚČçt'ääĞžæÌě

èğčåĘşæŰźæąĹ

PythonçŽĎæŸ§åŔůèąĺè¿¿åijŔåŘŕäžěçŤĺæÌěèğčåEşè£ŹäÿłéŮőécŸãĂĆæŕŤåęĆïijŇä¡ăåIJĺå■ęäźääÿĂéŮĺ
ä¡ăæČşçż§èőąäÿŇåőűåž■ä¡IJäÿŽçŽĎåźşåÌĞæĹŘçżl'ïijŇä¡ĘæŸŕæŐŠéŹď'æŐL'çňňäÿĂäÿłåŠŇæIJĂåŘŐäÿĂä
ä¡ĘåęĆædIJæIJL'24äÿłåŚćïij§è£ŹæŮűåĂŹæŸ§åŔůèąĺè¿¿åijŔåŕśæť¿äÿŁçŤĺåIJžäžĘïijŽ

```
def drop_first_last(grades):
    first, *middle, last = grades
    return avg(middle)
```

åŘęåď'ŰäÿĂçğ■æČĚåĘţïijŇåĄĞèő¿ä¡ăçŐřåIJĺæIJL'äÿĂäžŽçŤĺæĹůçŽĎèőřå¡ŢåĹŮèąĺïijŇærŔæÌąèőřå¡Ţ
ä¡ääŘŕäžěåČŘäÿŃéÌćè£ŹæăůåĹĘèğčè£ŹäžŻèőřå¡ŢïijŽ

```
>>> record = ('Dave', 'dave@example.com', '773-555-1212', '847-555-
↪1212')
>>> name, email, *phone_numbers = record
>>> name
'Dave'
>>> email
'dave@example.com'
>>> phone_numbers
['773-555-1212', '847-555-1212']
>>>
```

åĂijå¿ŮæşĺæĎŘçŽĎæŸŕäÿŁéÌćèğčåŐŃåĞžçŽĎ `phone_numbers`
åŘŸéĞŔæřÿè£IJéČ¡æŸŕåĹŮèąĺçśžåđŃïijŇäÿ■çőąèğčåŐŃçŽĎçŢţèŕÌåŘůçĂąæŢřéĞŔæŸŕåď'ŽåŕŚ(åŇĚæŃñ0
æĽĂäžěïijŇäżżä¡Ţä¡£çŤĺåĹř `phone_numbers` åŘŸéĞŔçŽĎäžčçĂąåŕśäÿ■éIJĂèęĄåĄŽåď'Žä¡ŹçŽĎçśžåđŃ

æŸ§åŕůèąĺè¿¿åijŔäź§èČ¡çŤĺåIJĺåĹŮèąĺçŽĎåijĂåğŃéČĺåĹĘãĂĆæŕŤåęĆïijÑä¡ăæIJĽäÿĂäÿłåĚňåŔÿåĽ■ä¡ĘæŸŕä¡ăæČşçIJŃäÿŃæIJĂè£ŚäÿĂäÿłæIJĹæŢřæ■őåŠÑåĽ■éİć7äÿłæIJĹçŽĎåźşåİĞåĂijçŽĎårźæŕŤãĂĆä¡ăă

```
*trailing_qtrs, current_qtr = sales_record
trailing_avg = sum(trailing_qtrs) / len(trailing_qtrs)
return avg_comparison(trailing_avg, current_qtr)
```

äÿŃéİćæŸŕåIJĺPythonèğčéĞŁåŹĺäÿ■æĽğèąŇçŽĎçżŞæđIJïijŽ

```
>>> *trailing, current = [10, 8, 7, 1, 9, 5, 10, 3]
>>> trailing
[10, 8, 7, 1, 9, 5, 10]
>>> current
3
```

èőĺèőž

æĽľ'åśŢçŽĎè£■äžčèğčåŐŃèŕ■æşŢæŸŕäÿŞéŮĺäÿžèğčåŐŃäÿ■çąőåőŽäÿłæŢřæĹŰäżżæĎŔäÿłæŢřåĚČçţ'ä
éĂŽåÿÿïijÑè£ŹäžŽåŔŕè£■äžčåŕźèśąçŽĎåĚČçţ'ăçżŞæđĎæIJĽçąőåőŽçŽĎèğĎåĹŹïijĹæŕŤåęĆçñň1äÿłåĚČçţ'ă
æŸ§åŕůèąĺè¿¿åijŔèőľ'åijĂåŔŚäžžåŚŸåŔŕäżěå¿ĹåőźæŸ§çŽĎåĹľçŤĺè£ŹäžŽèğĎåĹŹæİèèğčåŐŃåĞžåĚČçţ'ă
èĂŇäÿ■æŸŕéĂŽè£ĞäÿĂäžŽæŕŤè¿Čåď'■æİĆçŽĎæĽŃæőţåŐżèŐůåŔŰè£ŹäžŽåĚşèĄŢçŽĎçŽĎåĚČçţ'ääĂijã

åĂijå¿ŮæşĺæĎŔçŽĎæŸŕïijÑæŸ§åŕůèąĺè¿¿åijŔåIJĺè£■äžčåĚČçţ'ääÿžåŕŕåŔŸéŢ£åĚČçżĎçŽĎåžŔåĹŮæ
æŕŤåęĆïijÑäÿŃéİćæŸŕäÿĂäÿłåÿęæIJĽæăĞç■¿çŽĎåĚČçżĎåžŔåĹŮïijŽ

```
records = [
    ('foo', 1, 2),
    ('bar', 'hello'),
    ('foo', 3, 4),
]

def do_foo(x, y):
    print('foo', x, y)

def do_bar(s):
    print('bar', s)

for tag, *args in records:
    if tag == 'foo':
        do_foo(*args)
    elif tag == 'bar':
        do_bar(*args)
```

æŸ§åŕůèğčåŐŃèŕ■æşŢåIJĺå■ŮçňęäÿšæŞ■ä¡IJçŽĎæŮűåĂŹäź§äijŽå¿ĹæIJĽçŤĺïijÑæŕŤåęĆå■Ůçňęäÿšç

äžčçăĄçď'žä¿ŃïijŽ

```
>>> line = 'nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/
↪false'
>>> uname, *fields, homedir, sh = line.split(':')
```

```
>>> uname
'nobody'
>>> homedir
'/var/empty'
>>> sh
'/usr/bin/false'
>>>
```

æIJL'æŮűåĂŹïijŇä¡ăæČşèğčåŐŃäÿĂäžŻåĚČçt'ääŘŐäÿćåijČåőČäžňïijŇä¡ăäÿ■èČ¡çőĂå■Ţåřśä¡£çŤÍ * ïijŇ ä¡ĘæŸŕä¡ääŔŕäžëä¡£çŤĺäÿĂäÿłæŹőéĂŽçŽĎåž§åijČåŘ■çğřïijŇæŕŤåęĆ _ æĹŰèĂĚ ign ãĂĆ

äžčçăĄçď'žä¿ŃïijŽ

```
>>> record = ('ACME', 50, 123.45, (12, 18, 2012))
>>> name, *_, (*_, year) = record
>>> name
'ACME'
>>> year
2012
>>>
```

åIJĺå¿Ĺåď'ŽåĞ¡æŢřåijŔèŕ■èĺĂäÿ■ïijŇæŸ§åŘůèğčåŐŃèŕ■æşŢèů§åĹŮèąĺåď'ĎçŘĘæIJL'èőÿåď'ŽçŻÿäijijä ä¡ăäŔŕäžëå¿Ĺåőźæ ŸŞçŽĎåřĘåőČåĹĘåL'šæĹŘåL'■åŘŐäÿď'éČĺåĹĘïijŽ

```
>>> items = [1, 10, 7, 4, 5, 9]
>>> head, *tail = items
>>> head
1
>>> tail
[10, 7, 4, 5, 9]
>>>
```

åęĆæđIJä¡ăåď'§èĄłæŸŐçŽĎèŕİïijŇè£ŸèČ¡çŤĺè£Źçğ■åĹĘåL'šèŕ■æşŢåŐżåůğåęŹçŽĎåőđçŐřéĂŠå¡ŠçőŮ

```
>>> def sum(items):
...     head, *tail = items
...     return head + sum(tail) if tail else head
...
>>> sum(items)
36
>>>
```

çĎűåŘŐïijŇçŤśäžŐèŕ■èĺĂåśĆéİćçŽĎéŹŘåĹűïijŇéĂŠå¡Šåźűäÿ■æŸŕPythonæŞĚéŢ£çŽĎãĂĆ åŻăæ■d'ïijŇæIJĂåŘŐéĆčäÿłéĂŠå¡ŠæijŤçd'žäžĚäžĚæŸŕäÿłåě¡åěĞçŽĎæŐćçt'ćç¡ćäžĘïijŇåŕźè£Źäÿłäÿ■èęĄå

1.3 ä£ÌçŢŹæIJĂåŘŐNäÿłåĚČçť ă

éŮőécŸ

åIJÍè£■äžčæŞ■ä¡IJæĹŰèĂĚåĚűäżŰæŞ■ä¡IJçŽĎæŮűåĂŹïijŇæĂŐæăůåŔłä£ÌçŢŹæIJĂåŘŐæIJL'éŹŘåĞă

èğčåĘşæŰźæąĹ

ä£ÌçŢŹæIJL'éŹŘåŐĘåŔšèőřå¡Ţæ■čæŸŕ `collections.deque` åď ğæŸ¿èžńæĽ'ŃçŽĎæŮűåĂŹãĂĆæŕŤåęĆïijŇäÿŃéİćçŽĎäžčçăĄåIJÍåď Žèą Ňäÿ ŁéİćåĄŽçőĂå■ŢçŽĎæŰĞæ åźűè£ŤåŽđåŇźéĚ■æĽĂåIJÍèąŇçŽĎåĽ■NèąŇïijŽ

```
from collections import deque


def search(lines, pattern, history=5):
    previous_lines = deque(maxlen=history)
    for li in lines:
        if pattern in li:
            yield li, previous_lines
        previous_lines.append(li)

# Example use on a file
if __name__ == '__main__':
    with open(r'../../cookbook/somefile.txt') as f:
        for line, prevlines in search(f, 'python', 5):
            for pline in prevlines:
                print(pline, end='')
            print(line, end='')
            print('-' * 20)
```

èőÍèőž

æĹŚäžňåIJÍåĘŹæ§ěèŕćåĚČçť ăçŽĎäžčçăĄæŮűïijŇéĂŽåÿÿäijŽä¡£çŤÍåŇĚåŘń `yield` èąÍè¿¿åijŔçŽĎçŤ§æĹŘåŹÍåĞ¡æŢŕïijŇäź§ářśæŸŕæĹŚäžňäÿŁéİćçď'žä¿ŃäžčçăĄäÿ■çŽĎéĆčæăůãĂĆ è£ŹæăůåŘŕäžěářĘæŘIJçť'ćè£ĞçÍŃäžčçăĄåŠŇä¡£çŤÍæŘIJçť'ćçżŞædIJäžčçăĄèğčèĂęãĂĆåęĆædIJä¡ăè£Ÿäÿ■a

ä¡£çŤÍ `deque(maxlen=N)` ædĎéĂăåĞ¡æŢŕäijŽæŰřåžžäÿĂäÿłåŽžåőŽåď ğářŔçŽĎéŸ§åĹŮãĂĆå¡ŞæŰ æIJĂèĂĄçŽĎåĚČçť ăäijŽèĞłåŁÍèćńçğžéŹď æŐĽ ãĂĆ

äžčçăĄçď žä¿ŃïijŽ

```
>>> q = deque(maxlen=3)
>>> q.append(1)
>>> q.append(2)
>>> q.append(3)
>>> q
```

```
deque([1, 2, 3], maxlen=3)
>>> q.append(4)
>>> q
deque([2, 3, 4], maxlen=3)
>>> q.append(5)
>>> q
deque([3, 4, 5], maxlen=3)
```

åř¡çőąä¡ăäź§åŔŕäžěæĽŃåŁĺåIJĺäÿĂäÿłåĹÙèąĺäÿŁåőđçŐřè£ŹäÿĂçŽĎæŞ■ä¡IJ(æŕŤåęĆåćđåŁãăĂĄåĹăéŽ

æŻť'äÿĂèĹňçŽĎïijŇ `deque` çśżåŔŕäžěèćńçŤĺåIJĺäżżä¡Ţä¡ăåŔłéIJĂèęĄäÿĂäÿłçőĂå■ŢéŸ§åĹŮæŢřæ■ő
åęĆæđIJä¡ăäÿ■èő¿ç¡őæIJĂåď'ğéŸ§åĹŮåď'ğářŔïijŇéĆčäźĹářśäijŽå¿ŮåĹřäÿĂäÿłæŮăéŽŘåď'ğářŔéŸ§åĹŮïijŇ

äżčçăĄçď'žä¿ŃïijŽ

```
>>> q = deque()
>>> q.append(1)
>>> q.append(2)
>>> q.append(3)
>>> q
deque([1, 2, 3])
>>> q.appendleft(4)
>>> q
deque([4, 1, 2, 3])
>>> q.pop()
3
>>> q
deque([4, 1, 2])
>>> q.popleft()
4
```

åIJĺéŸ§åĹŮäÿď'çńŕæŔŠåĚěæĹŰåĹăéŹď'åĚČçť'ăæŮűéŮť'åď'■æİĆåžęéČ¡æŸŕ `O(1)` ïijŇèĂŇåIJĺåĹŮèąĺçŽĎåijĂåď'ť'æŔŠåĚěæĹŰåĹăéŹď'åĚČçť'ăçŽĎæŮűéŮť'åď'■æİĆåžęäÿž `O(N)` ãĂĆ

1.4 æ§ěæĽ'¿æIJĂåď'ğæĹŰæIJĂåřŔçŽĎNäÿłåĚČçť'ă

éŮőécŸ

æĂŐæăůäžŐäÿĂäÿłéŻĘåŘĹäÿ■èŐůå¿ŮæIJĂåď'ğæĹŰèĂĚæIJĂåřŔçŽĎNäÿłåĚČçť'ăåĹŮèąĺïij§

èğčåĘşæŰźæąĹ

heapqæĺąåİŮæIJL'äÿď'äÿłåĞ¡æŢřïijŽ`nlargest()` åŠŇ `nsmallest()` åŔŕäžěåőŇç¿ŐèğčåĘşè£ŹäÿłéŮőécŸãĂĆ

```
import heapq
nums = [1, 8, 2, 23, 7, -4, 18, 23, 42, 37, 2]
print(heapq.nlargest(3, nums)) # Prints [42, 37, 23]
print(heapq.nsmallest(3, nums)) # Prints [-4, 1, 2]
```

äÿď'äÿłåĞ¡æŢřéČ¡èČ¡æŐěåŔŮäÿĂäÿłåĚşéŤőå■ŮåŔĆæŢřïijŇçŤĺäžŐæŻť'åď■æİĆçŽĎæŢřæ■őçżŞædĎă

```
portfolio = [
    {'name': 'IBM', 'shares': 100, 'price': 91.1},
    {'name': 'AAPL', 'shares': 50, 'price': 543.22},
    {'name': 'FB', 'shares': 200, 'price': 21.09},
    {'name': 'HPQ', 'shares': 35, 'price': 31.75},
    {'name': 'YHOO', 'shares': 45, 'price': 16.35},
    {'name': 'ACME', 'shares': 75, 'price': 115.65}
]
cheap = heapq.nsmallest(3, portfolio, key=lambda s: s['price'])
expensive = heapq.nlargest(3, portfolio, key=lambda s: s['price'])
```

èŕŚèĂĚæşĺïijŽäÿŁéİćäžčçăĄåIJĺåŕźæŕŔäÿłåĚČçť'äè£ŻèąŇåŕźæŕŤçŽĎæŮűåĂŹïijŇäijŽäžě
`price` çŽĎåĂijè£ŻèąŇæŕŤè¿ČãĂĆ

èőĺèőž

åęĆæđIJä¡ăæČşåIJĺäÿĂäÿłéŻĘåŘĹäÿ■æ§ěæL'¿æIJĂåřŔæĹŰæIJĂåď'ğçŽĎNäÿłåĚČçť'äïijŇåźűäÿŤNåřŔ
åŻääÿžåIJĺåžŢåśĆåőđçŐřéĞŇéİćïijŇéęŰåĚĹäijŽåĚĹåřĘéŻĘåŘĹæŢřæ■őè£ŻèąŇäåĘæŐŠåžŔåŘŐæŤ¿åĚěäÿ

```
>>> nums = [1, 8, 2, 23, 7, -4, 18, 23, 42, 37, 2]
>>> import heapq
>>> heapq.heapify(nums)
>>> nums
[-4, 2, 1, 23, 7, 2, 18, 23, 42, 37, 8]
>>>
```

åäĘæŢřæ■őçżŞædĎæIJĂéĞ■èęĄçŽĎçL'źå¿ĄæŸŕ `heap[0]`
æřyè£IJæŸŕæIJĂåřŔçŽĎåĚČçť'ăãĂĆåźűäÿŤåL'ľä¡ŹçŽĎåĚČçť'ääŔŕäžěå¿Ĺåőźæŷ§çŽĎéĂŽè£ĞèřČçŤĺ
`heapq.heappop()` æŰźæşŢå¿ŮåĹřïijŇ èřěæŰźæşŢäijŽåĚĹåřĘçňňäÿĂäÿłåĚČçť'ääijźåĞžæİěïijŇçĎűåŘŐ
N)ïijŇŇæŸŕåäĘåď'ğåřŔ)ãĂĆ æŕŤåęĆïijŇåęĆædIJæČşèęĄæ§ěæL'¿æIJĂåřŔçŽĎ3äÿłåĚČçť'äïijŇä¡ääŔŕäžěè£Ź

```
>>> heapq.heappop(nums)
-4
>>> heapq.heappop(nums)
1
>>> heapq.heappop(nums)
2
```

å¡ŞèęĄæ§ěæL'¿çŽĎåĚČçť'ääÿłæŢřçŻÿåŕźæŕŤè¿ČåŕŔçŽĎæŮűåĂŹïijŇåĞ¡æŢř
`nlargest()` åŠŇ `nsmallest()` æŸŕå¿ĹåŘĹéĂĆçŽĎãĂĆ
åęĆædIJä¡ääžĚäžĚæČşæ§ěæL'¿åŤŕäŸĂçŽĎæIJĂåřŔæĹŰæIJĂåď'ğ(N=1)çŽĎåĚČçť'äçŽĎèŕÏïijŇéĆčäźĹä¡£çŤ
çśżäijijçŽĎïijŇåęĆædIJNçŽĎåď'ğåřŔåŠŇéŻĘåŘĹåď'ğåřŔæŐěè£ŠçŽĎæŮűåĂŹïijŇéĂŽåÿÿåĚĹæŐŠåžŔè£Ź

(sorted(items)[:N] æĹŰèĂĚæŸŕ sorted(items)[-N:])ãĂĆ éIJĂèęĄåIJĺæ■čçąőåIJžåŘĹä¡£çŤĺåĞ¡æŢřnlargest() åŠŇ nsmallest()æĽ■èČ¡åŔŚæŇěåőČäžňçŽĎäijŸåŁ£ (åęĆæđIJNå£ńæŐěè£ŚéŻĘåŘĹåd'ğářŔäžĘïijŇéĆčäźĹä¡£çŤĺæŐŠå

åř¡ç̃őąä¡äæšąæIJL'å£ĚèęĄäÿĂåőŽä¡£çŤĺè£ŹéĞŇçŽĎæŰźæşŢïijŇä¡ĘæŸŕåăĘæŢřæ■őçżŞæđĎçŽĎåőđçĆ å§žæIJňäÿŁåŔŀèęĄæŸŕæŢřæ■őçżŞæđĎåŠŇçőŮæşŢäźęçś■éĞŇéİćéČ¡äijŽæIJL'æŔŘåŔŁåĹřãĂĆ heapq æĺąåİŮçŽĎåőŸæŰźæŰĞæąčéĞŇéİćäź§èŕęçżĘçŽĎäżŃçż■äžĘăăĘæŢřæ■őçżŞæđĎåžŢåśĆçŽĎåőđçĆ

1.5 åőđçŐřäÿĂäÿłäijŸåĚĹçžğéŸ§åĹŮ

éŮőécŸ

æĂŐæăůåőđçŐřäÿĂäÿłæŇL'äijŸåĚĹçžğæŐŠåžŔçŽĎéŸ§åĹŮïij§ åźűäÿŤåIJĺè£ŹäÿłéŸ§åĹŮäÿŁéİćæŕŔæňąpopæŞ■ä¡IJæĂżæŸŕè£ŤåŻđäijŸåĚĹçžğæIJĂénŸçŽĎéĆčäÿłåĚČçt'

èğčåĘşæŰźæąĹ

äÿŃéİćçŽĎçśżåĹ'çŤĺ heapq æĺąåİŮåőđçŐřäžĘäÿĂäÿłçőĂå■ŢçŽĎäijŸåĚĹçžğéŸ§åĹŮïijŽ

```
import heapq

class PriorityQueue:
    def __init__(self):
        self._queue = []
        self._index = 0

    def push(self, item, priority):
        heapq.heappush(self._queue, (-priority, self._index, item))
        self._index += 1

    def pop(self):
        return heapq.heappop(self._queue)[-1]
```

äÿŃéİćæŸŕåőČçŽĎä¡£çŤĺæŰźåijŔïijŽ

```
>>> class Item:
...     def __init__(self, name):
...         self.name = name
...     def __repr__(self):
...         return 'Item({!r})'.format(self.name)
...
>>> q = PriorityQueue()
>>> q.push(Item('foo'), 1)
>>> q.push(Item('bar'), 5)
>>> q.push(Item('spam'), 4)
>>> q.push(Item('grok'), 1)
>>> q.pop()
```

```
Item('bar')
>>> q.pop()
Item('spam')
>>> q.pop()
Item('foo')
>>> q.pop()
Item('grok')
>>>
```

äžŤçžĘèğĆåŕ§åŔŕäžěåŔŚçŐřïijŇçňňäÿĂäÿł `pop()` æŞ■ä¡IJè£ŤåŻđäijŸåĚĹçžğæIJĂéńŸçŽĎåĚČçť'ăãĂ
åŔęåđ'ŰæşĺæĎŔåĹřåęĆæđIJäÿđ'äÿłæIJL'çİĂçŻÿåŘŇäijŸåĚĹçžğçŽĎåĚČçť'ă(`foo` åŠŇ `grok`)ïijŇpopæŞ■ä¡IJæŇL'çĚğåőČäžňèćńæŔŠåĚěåĹřéŸ§åĹŮçŽĎéąžåžŔè£ŤåŻđçŽĎãĂĆ

èőĺèőž

è£ŹäÿĂåřŔèŁĆæĹŚäžňäÿżèęĄåĚşæşĺ `heapq` æĺąåİŮçŽĎä¡£çŤĺãĂĆ
åĞ¡æŢř `heapq.heappush()` åŠŇ `heapq.heappop()`
åĹĘåĹńåIJĺéŸ§åĹŮ `_queue` äÿŁæŔŠåĚěåŠŇåĹăéŹđ'çňňäÿĂäÿłåĚČçť'äïijŇ
åźűäÿŤéŸ§åĹŮ_queueä£İèŕĄçňňäÿĂäÿłåĚČçť'ăæŇěæIJL'æIJĂåřŔäijŸåĚĹçžğ(1.4èŁĆåůšçżŔèőĺèőžè£ĞèŁŹ
`heappop()` åĞ¡æŢřæĂżæŸŕè£ŤåŻđ"æIJĂåřŔçŽĎ"çŽĎåĚČçť'äïijŇè£ŹåřśæŸŕä£İèŕĄéŸ§åĹŮpopæŞ■ä¡IJè
åŔęåđ'ŰïijŇçŤśäžŐpushåŠŇpopæŞ■ä¡IJæŮűéŮť'åđ'■æİĆåžęäÿžO(log
N)ïijŇåĚűäÿ■NæŸŕåăEçŽĎåđ'ğåřŔïijŇåŻăæ■đ'åŕśçőŮæŸŕNå¿Ĺåđ'ğçŽĎæŮűåĂŹåőČäžňè£ŘèąŇéĂ§åžęäź§

åIJĺäÿŁéİćäžčçăĄäÿ■ïijŇéŸ§åĹŮåŇĚåŘńäžEäÿĂäÿł `(-priority, index, item)` çŽĎåĚČçżĎãĂĆ äijŸåĚĹçžğäÿžèť'§æŢřçŽĎçŻőçŽĎæŸŕä¡£å¿ŮåĚČçť'ăæŇL'çĚğäijŸåĚĹçžğäžŐéńŸ
è£Źäÿłèů§æŹőéĂŽçŽĎæŇL'äijŸåĚĹçžğäżŐä¡ŐåĹřéńŸæŐŠåžŔçŽĎåăEæŐŠåžŔæĄřåůğçŻÿåŔ■ãĂĆ

`index` åŔŸéĞŔçŽĎä¡IJçŤĺæŸŕä£İèŕĄåŘŇç■L'äijŸåĚĹçžğåĚČçť'ăçŽĎæ■čçąőæŐŠåžŔãĂĆ
éĂŽè£Ğä£Ía■ŸäÿĂäÿłäÿ■æŰ■åćđåŁăçŽĎ `index` äÿŃæăĞåŔŸéĞŔïijŇåŔŕäžěçąőä£İåĚČçť'ăæŇL'çĚğåőČä
èĂŇäÿŤïijŇ `index` åŔŸéĞŔäź§åIJĺçŻÿåŘŇäijŸåĚĹçžğåĚČçť'ăæŕŤè¿ČçŽĎæŮűåĂŹètůåĹřéĞ■èęĄä¡IJçŤĺã

äÿžäžĘéŸŘæŸŐè£ŹäžŻïijŇåĚĹåĄĞåőŽItemåőđä¿ŃæŸŕäÿ■æŤræŇĄæŐŠåžŔçŽĎïijŽ

```
>>> a = Item('foo')
>>> b = Item('bar')
>>> a < b
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: unorderable types: Item() < Item()
>>>
```

åęĆæđIJä¡ăä¡£çŤĺåĚČçżĎ `(priority, item)` ïijŇåŔłèęĄäÿđ'äÿłåĚČçť'ăçŽĎäijŸåĚĹçžğäÿ■åŘŇår
ä¡ĘæŸŕåęĆæđIJäÿđ'äÿłåĚČçť'ăäijŸåĚĹçžğäÿĂæăůçŽĎèŕİïijŇéĆčäźĹæŕŤè¿ČæŞ■ä¡IJårśäijŽèů§äźŇåL'■äÿĂ

```
>>> a = (1, Item('foo'))
>>> b = (5, Item('bar'))
>>> a < b
True
>>> c = (1, Item('grok'))
```

```
>>> a < c
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: unorderable types: Item() < Item()
>>>
```

éĂŽè£ĞåijŢåĚěåŔęåď ŰçŽĎ `index` åŔŸéĞŔçżĎæĹŘäÿĽåĚČçżĎ `(priority, index, item)` ïijŇåřśèČ¡å¿Ĺåě¡çŽĎéĄ£åĚ■äÿŁéİćçŽĎéŤŹèŕřïijŇ åŻääÿžäÿ■åŔŕèČ¡æIJL'äÿď äÿłåĚČçť ăæIJL'çŻÿåŘŇçŽĎ `index` åĂijãĂĆPythonåIJĺåĄŽåĚČçżĎæŕŤè¿ČæŮűåĂŹïijŇåęĆæđIJåL'■éİćçŽĎæŕŤè¿ČäžěåŔŁåŔŕäžěçąőåőŽçżŞæđ åŘŐéİćçŽĎæŕŤè¿ČæŞ■ä¡IJåřśäÿ■äijŽåŔŚçŤ§äžĘïijŽ

```
>>> a = (1, 0, Item('foo'))
>>> b = (5, 1, Item('bar'))
>>> c = (1, 2, Item('grok'))
>>> a < b
True
>>> a < c
True
>>>
```

åęĆæđIJä¡ăæČşåIJĺåď Žäÿłçž£çĺŃäÿ■ä¡£çŤĺåŘŇäÿĂäÿłéŸ§åĹŮïijŇéĆčäźĹä¡ăéIJĂèęĄáćđåŁăéĂĆå¡ŞçŽ åŔŕäžěæ§ěçIJŃ12.3åŕŘèŁĆçŽĎä¿Ńå■ŘæijŢçď žæŸŕæĂŐæăůåĄŽçŽĎãĂĆ

`heapq` æĺąåİŮçŽĎåőŸæŰźæŰĞæąčæIJL'æŻť'èŕęçżEçŽĎä¿Ńå■ŘçĺŃåžŔäžěåŘŁåŕźäžŐåăEçŘĘèőžåŘŁ

1.6 å■ŮåĚÿäÿ■çŽĎéŤőæŸăåřĎåď Žäÿłå Äij

éŮőécŸ

æĂŐæăůåőđçŐřäÿĂäÿłéŤőåŕźåžŤåď Žäÿłå ÄijçŽĎå■ŮåĚÿ(äź§åŔń `multidict`)ïij§

èğčåĘşæŰźæąĹ

äÿĂäÿłå■ŮåĚÿåřśæŸŕäÿĂäÿłéŤőåŕźåžŤäÿĂäÿłå■ŢåÄijçŽĎæŸăåřĎãĂĆåęĆæđIJä¡ăæČşèęĄäÿĂäÿłéŤőå æŕŤåęĆåĹŮèąĺæĹŰèĂĚéŽĘåŘĹéĞŃéİćãĂĆæŕŤåęĆïijŇä¡ăåŔŕäžěåČŔäÿŃéİcè£ŹæăůæđĎéĂăè£ŹæăůçŽĎå■

```
d = {
    'a' : [1, 2, 3],
    'b' : [4, 5]
}
e = {
    'a' : {1, 2, 3},
    'b' : {4, 5}
}
```

éĂĽ'æŃľ'ä¡£çŤĺåĹŮèąÍè£ŸæŸŕéŽĘåŘĹåŔŰåĘşäžŐä¡ăçŽĎåőđéŹĚéIJĂæśĆãĂĆåęĆæđIJä¡ăæČşä£İæŇĂ
åęĆæđIJæČşåŐżæŐL'éĞ■åd'■åĚČçť'ăåŕšä¡£çŤĺéŽĘåŘĹïijĹåźűäÿŤäÿ■åĚşå£ČåĚČçť'ăçŽĎéąžåžŔéŮőécŸïijĽ

ä¡ăåŔŕäžěå¿ĹæŰźä¿£çŽĎä¡£çŤĺ `collections` æĺąåİŮäÿ■çŽĎ `defaultdict` æİěæđĎéĂäè£Źæăůç ŽĎå■ŮåĚÿãĂĆ `defaultdict` çŽĎäÿĂäÿłçĽźå¿ĄæŸŕåőČäijŽèĞłåŁĺåĹİåğŇåŇŰæŕŘäÿł `key` åĹŽåijĂåğŇåŕźåžŤçŽĎåĂijïijŇæĽ'Ăäžěä¡ăåŔłéIJĂèęĄåĚşæşĺæůżåŁăåĚČçť'ăæŞ■ä¡IJäžĘãĂĆæŕŤåęĆïijŽ

```
from collections import defaultdict

d = defaultdict(list)
d['a'].append(1)
d['a'].append(2)
d['b'].append(4)

d = defaultdict(set)
d['a'].add(1)
d['a'].add(2)
d['b'].add(4)
```

éIJĂèęĄæşĺæĎŘçŽĎæŸŕïijŇ `defaultdict` äijŽèĞłåŁĺäÿžåřĘèęĄèő£éŮőçŽĎéŤő(åřšçőŮçŻőåĽ'■å■Ů åęĆæđIJä¡ăåźűäÿ■éIJĂèęĄè£ŹæăůçŽĎçĽźæĂğïijŇä¡ăåŔŕäžěåIJĺäÿĂäÿłæŹőéĂŽçŽĎå■ŮåĚÿäÿŁä¡£çŤĺ `setdefault()` æŰźæşŢæİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
d = {} # A regular dictionary
d.setdefault('a', []).append(1)
d.setdefault('a', []).append(2)
d.setdefault('b', []).append(4)
```

ä¡ĘæŸŕå¿Ĺåď'ŽçÍŇåžŘåŚŸèğĽå¿Ů `setdefault()` çŤĺèţůæİěæIJĽ'çĆźåĹńæĽ'■ãĂĆåŻääÿžæŕŘæňąèř

èőĺèőž

äÿĂèĹňæİěèőšïijŇåĹŻåžžäÿĂäÿłåď'ŽåĂijæŸäářĎå■ŮåĚÿæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆä¡ĘæŸŕïijŇåęĆæđIJä
ä¡ăåŔŕèČ¡äijŽåČŘäÿŃéİćè£ŹæăůæİěåőđçŐřïijŽ

```
d = {}
for key, value in pairs:
    if key not in d:
        d[key] = []
    d[key].append(value)
```

åęĆæđIJä¡£çŤĺ `defaultdict` çŽĎèŕİäžčçăĄåřśæŻť'åŁăçőĂæť'ĄäžĘïijŽ

```
d = defaultdict(list)
for key, value in pairs:
    d[key].append(value)
```

è£ŹäÿĂåřŘèŁĆæL'Ăèőĺèőžç ŽĎéŮőécŸèů§æŢřæ■őåď'ĎçŘĘäÿ■çŽĎèőřå¡Ţå¡ŠçśźéŮőécŸæIJL'åď'ğçŽĎ

1.7 å■ŮåĚÿæŐŠåžŔ

éŮőécŸ

ä¡ăæČşåĹŻåžžäÿĂäÿłå■ŮåĚÿïijŇåźűäÿŤåIJÍè£■äžčæĹŰåžŔåĹŮåŇŰè£ŹäÿłåŮåĚÿçŽĎæŮűåĂŹèČ¡åd

èğčåĘşæŰźæąĹ

äÿžäžĘèČ¡æŐğåĹűäÿĂäÿłå■ŮåĚÿäÿ■åĚČçť'äçŽĎéąžåžŔïijŇä¡ăåŔŕäžěä¡£çŤÍ collections æÍąåİŮäÿ■çŽĎ OrderedDict çśżãĂĆ åIJÍè£■äžčæŞ■ä¡IJçŽĎæŮűåĂŹåőČäijŽä£İæŇĄåĚČçť'äèćńæŔŠåĚěæŮűçŽĎéąžåžŔïijŇçď'žä¿ŃåęĆäÿŃïijŽ

```
from collections import OrderedDict
def ordered_dict():
    d = OrderedDict()
    d['foo'] = 1
    d['bar'] = 2
    d['spam'] = 3
    d['grok'] = 4
    # Outputs "foo 1", "bar 2", "spam 3", "grok 4"
    for key in d:
        print(key, d[key])
```

å¡Şä¡ăæČşèęĄæđĎåžžäÿĂäÿłårĘæİěéIJĂèęĄåžŔåĹŮåŇŰæĹŰçijŰçăĄæĹŘåĚűäžŰæäijåijŔçŽĎæŸăåřI OrderedDict æŸŕéİđåÿÿæIJL'çŤÍçŽĎãĂĆ æŕŤåęĆïijŇä¡ăæČşçš¿çąőæŐğåĹűäžěJSONçijŰçăĄåŘŐå■Ůæč OrderedDict æİěæđĎåžžè£ŹæăůçŽĎæŢřæ■őïijŽ

```
>>> import json
>>> json.dumps(d)
'{"foo": 1, "bar": 2, "spam": 3, "grok": 4}'
>>>
```

èőÍèőž

OrderedDict åĘĚéČÍçżť'æŁď'çİĂäÿĂäÿłæăźæ■őéŤőæŔŠåĚěéąžåžŔæŐŠåžŔçŽĎåŔŇåŘŠéŞ¿èąÍãĂĆ åőČäijŽèćńæŤ¿åĹřéŞ¿èąÍçŽĎår¿éČÍãĂĆårźäžŐäÿĂäÿłåůšçżŔå■ŸåIJÍçŽĎéŤőçŽĎéĞ■åď'■ètŇåĂijäÿ■äijŽæT

éIJĂèęĄæşÍæĐŔçŽĎæŸŕïijŇäÿĂäÿł OrderedDict çŽĎåď'gårŔæŸŕäÿĂäÿłæŹőéĂŽå■ŮåĚÿçŽĎäÿď'å æĽĂäžěåęĆædIJä¡äèęĄæđĎåžžäÿĂäÿłéIJĂèęĄåď'géĞŔ OrderedDict åőđä¿ŃçŽĎæŢřæ■őçżŞædĎçŽĎæŮűåĂŹ(æŕŤåęĆèŕźåŔŰ100,000èąŇCSVæŢřæ■őåĹřäÿĂäÿł OrderedDict åĹŮèąÍäÿ■åŐż)ïijŇ éĆčäźĹä¡ăåršå¿ŮäžŤçżĘæİČèąąäÿĂäÿŃæŸŕåŘęä¡£çŤÍ OrderedDict åÿęæİěçŽĎåě¡åď'ĎèęĄåď'gè£ĞécÍåď'ŰåĘĚå■ŸæűĹèĂŮçŽĎå¡šåŞ■ãĂĆ

1.8 å■ŮåĚÿçŽĎè£ŘçőŮ

éŮőécŸ

æĂŐæăůåIJÍæŢřæ■őå■ŮåĚÿäÿ■æĽğèąŇäÿĂäžŻèőąçőŮæŞ■ä¡IJ(æŕŤåęĆæśĆæIJĂåřŔåĂijãĂĄæIJĂåď

èğčåĘşæŰźæąĹ

èĂČèŹŚäÿŃéİćçŽĎèĆąçĕĺåŘ■åŠŇäžůæäijæŸăåřĎå■ŮåĚÿïijŽ

```
prices = {
    'ACME': 45.23,
    'AAPL': 612.78,
    'IBM': 205.55,
    'HPQ': 37.20,
    'FB': 10.75
}
```

äÿžäžĘåŕźå■ŮåĚÿåĂijæĽğèąŇèőąçőŮæŞ■ä¡IJïijŇéĂŽåÿÿéIJĂèęĄä¡£çŤÍ zip() åĞ¡æŢřåĚĹåřĘéŤőåŠŇåĂijåŔ■è¡ňè£ĞæİěãĂĆ æŕŤåęĆïijŇäÿŃéİćæŸŕæ§ěæĽ¿æIJĂåřŔåŠŇæIJĂåď ğèĆąçĕĺ

```
min_price = min(zip(prices.values(), prices.keys()))
# min_price is (10.75, 'FB')
max_price = max(zip(prices.values(), prices.keys()))
# max_price is (612.78, 'AAPL')
```

çśżäijijçŽĎïijŇåŔŕäžěä¡£çŤÍ zip() åŠŇ sorted() åĞ¡æŢřæİěæŐŠåĹŮå■ŮåĚÿæŢřæ■őïijŽ

```
prices_sorted = sorted(zip(prices.values(), prices.keys()))
# prices_sorted is [(10.75, 'FB'), (37.2, 'HPQ'),
#                   (45.23, 'ACME'), (205.55, 'IBM'),
#                   (612.78, 'AAPL')]
```

æĽğèąŇè£ŹäžŻèőąçőŮçŽĎæŮűåĂŹïijŇéIJĂèęĄæşĺæĎŔçŽĎæŸŕ zip() åĞ¡æŢřåĹŻåżžçŽĎæŸŕäÿĂäÿłåŔłèČ¡èő£éŮőäÿĂæňąçŽĎè£■äžčåŹĺãĂĆ æŕŤåęĆïijŇäÿŃéİćçŽĎäžčçăĄåřśäijŽäžğçŤ§éŤŹèŕŕïijŽ

```
prices_and_names = zip(prices.values(), prices.keys())
print(min(prices_and_names)) # OK
print(max(prices_and_names)) # ValueError: max() arg is an empty␣
↪sequence
```

èőĺèőž

åęĆædIJä¡ăåIJĺäÿĂäÿłå■ŮåĚÿäÿŁæĽğèąŇæŹőéĂŽçŽĎæŢřå■ęè£ŘçőŮïijŇä¡ääijŽåŔŚçŐřåőČäžňäžĚäž

```
min(prices) # Returns 'AAPL'
max(prices) # Returns 'IBM'
```

è£ŹäÿłçżŝæđIJåźűäÿ■æŸŕä¡ăæČşèęĄçŽĎïijŇåŻăäÿžä¡ăæČşèęĄåIJĺå■ŮåĚÿçŽĎåĂijéŻĘåŘĹäÿŁæĽğèą
æĹŰèőÿä¡ăäijŽårİèŕŢçİĂä¡£çŤĺå■ŮåĚÿçŽĎ values() æŰźæşŢæİěèğčåĘşè£ŹäÿłéŭőécŸïijŽ

```
min(prices.values()) # Returns 10.75
max(prices.values()) # Returns 612.78
```

äÿ■åźÿçŽĎæŸŕïijŇéĂŽåÿÿè£ŹäÿłçżŝæđIJåŘŇæăůäź§äÿ■æŸŕä¡ăæČşèęĄçŽĎãĂĆ
ä¡ăåŘŕèČ¡è£ŸæČşèęĄç§ěéĄŞårźåžŢçŽĎéŤőçŽĎä£ąæĄŕ(æŕŤåęĆéĆčçğ■èĆąçĚläžůæäijæŸŕæIJĂä¡ŐçŽĎïij§

ä¡ăåŘŕäžěåIJÍ min() åŠŇ max() åĞ¡æŢřäÿ■æŔŘä¿Ż key
åĞ¡æŢřåŔĆæŢřæİěèŐůåŘŰæIJĂårŘåĂijæĹŰæIJĂåd'ğåĂijårźåžŢçŽĎéŤőçŽĎä£ąæĄŕãĂĆærŤåęĆïijŽ

```
min(prices, key=lambda k: prices[k]) # Returns 'FB'
max(prices, key=lambda k: prices[k]) # Returns 'AAPL'
```

ä¡ĘæŸŕïijŇåęĆæđIJè£ŸæČşèęĄå¿ŮåĹřæIJĂårŘåĂijïijŇä¡ăåŔĹå¿ŰæĽğèąŇäÿĂæňąæ§ěæĽ¿æŞ■ä¡IJãĂ

```
min_value = prices[min(prices, key=lambda k: prices[k])]
```

åĽ■éİćçŽĎ zip() åĞ¡æŢřæŰźæąĹéĂŽè£ĞårĘå■ŮåĚÿ"åŔ■è¡ň"äÿž(åĂijïijŇéŤő)åĚČçżĎåžŘåĹŰæİěè
å¡ŞærŤè¿Čäÿd'äÿłåĚČçżĎçŽĎæŰűåĂŹïijŇåĂijäijŽåĚĹè£ŻèąŇærŤè¿ČïijŇçĎűåŘŐæĽ■æŸŕéŤõãĂĆ
è£ŹæăůçŽĎèŕİä¡ăårśèČ¡éĂŽè£ĞäÿĂæİąçőĂå■ŢçŽĎèŕ■åŘěårśèČ¡å¿Ĺè¡żæİ¿çŽĎåőđçŐřåIJĺå■ŮåĚÿäÿŁçŽĎ

éIJĂèęĄæşĺæĎŘçŽĎæŸŕåIJĺèőąçőŰæŞ■ä¡IJäÿ■ä¡£çŤĺåĹřäžĘ(åĂijïijŇéŤő)årźãĂĆå¡Şåd'Žäÿłåőđä¡ŞæŇ
ærŤåęĆïijŇåIJĺæĽğèąŇ min() åŠŇ max() æŞ■ä¡IJçŽĎæŰűåŹïijŇåęĆæđIJæĄřåůğæIJĂårŘæĹŰæIJĂåd'

```
>>> prices = { 'AAA' : 45.23, 'ZZZ': 45.23 }
>>> min(zip(prices.values(), prices.keys()))
(45.23, 'AAA')
>>> max(zip(prices.values(), prices.keys()))
(45.23, 'ZZZ')
>>>
```

1.9 æ§ěæĽ¿äÿd'å■ŮåĚÿçŽĎçŻÿåŘŇçĆź

éŮőécŸ

æĂŐæăůåIJĺäÿd'äÿłå■ŮåĚÿäÿ■årżårżæĽ¿çŻÿåŘŇçĆź(ærŤåęĆçŻÿåŘŇçŽĎéŤõãĂĄçŻÿåŘŇçŽĎåĂijç■

èğčåĘşæŰźæąĹ

èĂČèŹŚäÿŃéİćäÿd'äÿłå■ŮåĚÿïijŽ

```
a = {
    'x' : 1,
    'y' : 2,
    'z' : 3
}

b = {
    'w' : 10,
    'x' : 11,
    'y' : 2
}
```

äÿžäžĘåŕżæĽ¿äÿd'äÿłå■ŮåĚÿçŽĎçŻÿåŘŇçĆźïijŇåŔŕäžěçőĂå■ŢçŽĎåIJÍäÿd'å■ŮåĚÿçŽĎ `keys()` æĹŰèĂĚ `items()` æŰźæşŢè£ŤåŽđçżŞæđIJäÿŁæĽ'ğèąŇéŻĘåŘĹæŞ■ä¡IJãĂĆæŕŤåęĆïijŽ

```
# Find keys in common
a.keys() & b.keys() # { 'x', 'y' }
# Find keys in a that are not in b
a.keys() - b.keys() # { 'z' }
# Find (key,value) pairs in common
a.items() & b.items() # { ('y', 2) }
```

è£ŹäžŻæŞ■ä¡IJäź§åŔŕäžěçŢĺäžŐä£őæŢźæĹŰèĂĚè£Ğæżd'å■ŮåĚÿåĚČçt'ããĂĆ æŕŤåęĆïijŇåĄĞåęĆä¡ãæČşäžěçŐřæIJL'å■ŮåĚÿæđĎéĂääÿĂäÿłæŐŠéŹd'åĞääÿłæŇĞåőŽéŤőçŽĎæŰřå■ŮåĚ äÿŃéÍćåĹl'çŤĺå■ŮåĚÿæŐĺåŕijæÌěåőđçŐřè£ŹæăůçŽĎéIJĂæśĆïijŽ

```
# Make a new dictionary with certain keys removed
c = {key:a[key] for key in a.keys() - {'z', 'w'}}
# c is {'x': 1, 'y': 2}
```

èőÍèőž

äÿĂäÿłå■ŮåĚÿåŕśæŸŕäÿĂäÿłéŤőéŻĘåŘĹäÿŐåĂijéŻĘåŘĹçŽĎæŸăåŕĎåĚşçşžãĂĆ å■ŮåĚÿçŽĎ `keys()` æŰźæşŢè£ŤåŽđäÿĂäÿłåśŢçŐřéŤőéŻĘåŘĹçŽĎéŤőèğĘåŻ¿åŕźèśąãĂĆ éŤőèğĘåŻ¿çŽĎäÿĂäÿłå¿ĹåřŚèćńäžĘèğčçŽĎçL'źæĂğåŕśæŸŕåőĆäžňäź§æŤŕæŇĄéŻĘåŘĹæŞ■ä¡IJïijŇærŤåęĆ æL'ĂäžěïijŇåęĆæđIJä¡ãæČşåŕźéŻĘåŘĹçŽĎéŤőæL'ğèąŇäÿĂäžŻæŹőéĂŽçŽĎéŻĘåŘĹæŞ■ä¡IJïijŇåŔŕäžěçŻt'

å■ŮåĚÿçŽĎ `items()` æŰźæşŢè£ŤåŽđäÿĂäÿłåŇĚåŘń(éŤőïijŇåĂij)åŕźçŽĎåĚČçt'ãèğĘåŻ¿åŕźèśąãĂĆ è£ŹäÿłåŕźèśąåŘŇæăůäź§æŤŕæŇĄéŻĘåŘĹæŞ■ä¡IJïijŇázűäÿŤåŔŕäžěèćńçŤĺæÌěæ§ěæL'¿äÿd'äÿłå■ŮåĚÿæIJL'å

år¡çőąå■ŮåĚÿçŽĎ `values()` æŰźæşŢäź§æŸŕçśżäijijïijŇä¡ĘæŸŕåőČåźűäÿ■æŤŕæŇĄè£ŹéĞŇäżŃçż■ç æ§Řçğ■çÍŃåžęäÿŁæŸŕåŽăäÿžåĂijèğĘåŻ¿äÿ■èČ¡ä£ÌèŕĄæL'ĂæIJL'çŽĎåĂijäžŠäÿ■çŽÿåŘŇïijŇè£ŹæăůäijŽår äÿ■è£ĞïijŇåęĆæđIJä¡ãçąňèęĄåIJÍåĂijäÿŁéÌćæL'ğèąŇè£ŹäžŻéŽĘåŘĹæŞ■ä¡IJçŽĎèŕÍïijŇä¡ååŔŕäžěåĚĹåřĘåĂ

1.10 åĹäéŹďåžŕåĹŮçŻÿåŘŇåĚČçťďäźűä£ÌæŇĄéąžåžŔ

éŮőécŸ

æĂŐæăůåIJÍäÿĂäÿłåžŕåĹŮäÿŁéÍćä£ÌæŇĄåĚČçťďäéąžåžŔçŽĎåŘŇæŮűæűĹéŹďéĞ■åď■çŽĎåĂijïij§

èğčåĘşæŰźæąĹ

åęĆæđIJåžŕåĹŮäÿŁçŽĎåĂijéČ¡æŸŕ `hashable` çśžåđŃïijŇéĆčäźĹåŕŕäžěå¿ĹçőĂå■ŢçŽĎåĹľçŤĺéŻĘ

```
def dedupe(items):
    seen = set()
    for item in items:
        if item not in seen:
            yield item
            seen.add(item)
```

äÿŃéÍćæŸŕä¡£çŤĺäÿŁè£ŕåĞ¡æŢřçŽĎä¿Ńå■ŘïijŽ

```
>>> a = [1, 5, 2, 1, 9, 1, 5, 10]
>>> list(dedupe(a))
[1, 5, 2, 9, 10]
>>>
```

è£ŹäÿłæŰźæşŢäžĚäžĚåIJÍåžŕåĹŮäÿ■åĚČçť'ääÿž `hashable` çŽĎæŮűåĂŹæĽ■çőąçŤĺãĂĆ åęĆæđIJä¡äæČşæűĹéŹďåĚČçť'ääÿ■åŕŕåŞĹåÿŇ(æŕŤåęĆ `dict` çśžåđŃ)çŽĎåžŕåĹŮäÿ■éĞ■åď■åĚČçť'ăçŽĎèŕÏïijŇä¡äéIJÄèęĄåřĘäÿŁè£ŕäžčçăĄçÍ■å¿őæŤźåŔŸäÿĂäÿŃïijŇ

```
def dedupe(items, key=None):
    seen = set()
    for item in items:
        val = item if key is None else key(item)
        if val not in seen:
            yield item
            seen.add(val)
```

è£ŹéĞŇçŽĎkeyåŔĆæŢřæŇĞåőŽäžĘäÿĂäÿłåĞ¡æŢřïijŇåřĘåžŕåĹŮåĚČçť'äè¡ňæ■ćæĹŘ `hashable` çśžåđŃãĂĆäÿŃéÍćæŸŕåőČçŽĎçŤĺæşŢçď'žä¿ŃïijŽ

```
>>> a = [ {'x':1, 'y':2}, {'x':1, 'y':3}, {'x':1, 'y':2}, {'x':2, 'y
↪':4}]
>>> list(dedupe(a, key=lambda d: (d['x'],d['y'])))
[{'x': 1, 'y': 2}, {'x': 1, 'y': 3}, {'x': 2, 'y': 4}]
>>> list(dedupe(a, key=lambda d: d['x']))
[{'x': 1, 'y': 2}, {'x': 2, 'y': 4}]
>>>
```

åęĆæđIJä¡äæČşå§žäžŐå■Ţäÿłå■Ůæőţã ĂĄåśđæĂğæĹŰèĂĚæ§ŘäÿłæŻť'åď'ğçŽĎæŢřæ■őçżŞæđĎæÍěæűĺ

èőĺèőž

åęĆæđIJä¡ăäžĚäžĚåřśæŸŕæČşæűĹéŹď´éĞ■åď■åĚČçť´äïijŇéĂŽåÿÿåŔŕäžěçőĂå■ŢçŽĎæđĎéĂääÿĂäÿłé

```
>>> a
[1, 5, 2, 1, 9, 1, 5, 10]
>>> set(a)
{1, 2, 10, 5, 9}
>>>
```

çĎűèĂŇïijŇè£Źçğ■æŰźæşŢäÿ■èČ¡çżť´æŁď´åĚČçť´ăçŽĎéąžåžŔïijŇçŤ§æĹŘçŽĎçżŞæđIJäÿ■çŽĎåĚČçť

åIJĺæIJňèŁĆäÿ■æĹŚäžňä¡£çŤĺäžĘçŤ§æĹŘåŹĺåĞ¡æŢřèőľæĹŚäžňçŽĎåĞ¡æŢřæŻť´åŁáéĂŽçŤĺïijŇäÿ■äžł
ærŤåęĆïijŇåęĆæđIJåęĆæđIJä¡ăæČşèŕžåŘŰäÿĂäÿłæŰĞäžűïijŇæűĹéŹď´éĞ■åď■èąŇïijŇä¡ăåŘŕäžěå¿ĹåőźæŸ

```
with open(somefile,'r') as f:
for line in dedupe(f):
    ...
```

äÿŁè£řkeyåĞ¡æŢřåŔĆæŢřæĺąäż£äžĘ `sorted()` , `min()` åŠŇ `max()` ç■Ľ´åĘĚç¡őåĞ¡æŢřçŽĎçŻÿäijijåŁ§èČ¡ãĂĆ åŘŕäžěåŔĆèĂČ1.8åŠŇ1.13åřŔèŁĆäžĘèğčæŻť´åď´ŽãĂĆ

1.11 åŚ¡åŘ■åĹĞçĽĞ

éŮőécŸ

ä¡ăçŽĎçÍŇåžŔåůšçżŔåĞžçŐřäÿĂåď´gåäĘåůšæŮăæşŢçŻť´èğĘçŽĎçąňçijŰçăĄåĹĞçĽĞäÿŃæăĞïijŇçĎű

èğčåĘşæŰźæąĹ

åĄĞåőŽä¡ăæIJĽäÿĂæőţäżčçăĄèęĄäżŐäÿĂäÿłèőřå¡Ţå■Ůçňęäÿšäÿ■åĞääÿłåŽžåőŽä¡■ç¡őæŔŘåŔŰåĞžçl

```
###### 01234567890123456789012345678901234567890123456789012345678901234567890
↪'
record = '.....................100 .......513.25 ..........'
cost = int(record[20:23]) * float(record[31:37])
```

äÿŐåĚűéĆčæăůåĘŹïijŇäÿžäžĂäźĹäÿ■æČşè£Źæăůå Ś¡åŘ■åĹĞçĽĞåŚćïijŽ

```
SHARES = slice(20, 23)
PRICE = slice(31, 37)
cost = int(record[SHARES]) * float(record[PRICE])
```

çňňäžŇçğ■çĽĹæIJňäÿ■ïijŇä¡ăéĄ£åĚ■äžĘåď´géĞŔæŮăæşŢçŘĘèğčçŽĎçąňçijŰçăĄäÿŃæăĞïijŇä¡£å¿Ůă

èőĺèőž

äÿĂèĹňæİěèőšïijŇäžčçăĄäÿ■åęĆæđIJåĞžçŐřåď ğéĞŔçŽĎçąňçijŰçăĄäÿŃæăĞåĂijäijŽä¡£å¿ŮåŔŕèřžæĂ
æŕŤåęĆïijŇåęĆæđIJä¡ăåŽđè£ĞæİěçIJŃçIJŃäÿĂåźť åL'■ä¡ăåEŹçŽĎäžčçăĄïijŇä¡ăäijŽæŚÿçİĂèĎŚèćŃæČşéĆ
è£ŹéĞŇçŽĎèğčåĘşæŰźæąĹæŸŕäÿĂäÿłå¿ĹçőĂå■ŢçŽĎæŰźæşŢèől'ä¡ăæŻť åŁăæÿĚæŹřçŽĎèąĺè¿¿äžčçăĄåĹ

åĘĚç¡őçŽĎ `slice()` åĞ¡æŢřåĹŻåžžäžĘäÿĂäÿłåĹĞçL'ĞåŕźèšąïijŇåŔŕäžěèćńçŤĺåIJĺäžžä¡ŢåĹĞçL'ĞåĚ

```
>>> items = [0, 1, 2, 3, 4, 5, 6]
>>> a = slice(2, 4)
>>> items[2:4]
[2, 3]
>>> items[a]
[2, 3]
>>> items[a] = [10,11]
>>> items
[0, 1, 10, 11, 4, 5, 6]
>>> del items[a]
>>> items
[0, 1, 4, 5, 6]
```

åęĆæđIJä¡ăæIJL'äÿĂäÿłåĹĞçL'ĞåŕźèšąaïijŇä¡ăåŔŕäžěåĹĘåĹńèřČçŤĺåőČçŽĎ `a.start`, `a.stop`, `a.step` åšđæĂğæİěèŐůåŔŰæŻť åď ŽçŽĎä£ąæĄŕãĂĆæŕŤåęĆïijŽ

```
>>> a = slice(5, 50, 2)
>>> a.start
5
>>> a.stop
50
>>> a.step
2
>>>
```

åŔęåď ŰïijŇä¡ăè£ŸèČ¡éĂŽè£ĞèřČçŤĺåĹĞçL'ĞçŽĎ `indices(size)` æŰźæşŢåřĘåőČæŸăåřĎåĹřäÿĂäÿłçąőåőŽåď ğåřŔçŽĎåžŔåĹŮäÿŁïijŇ è£ŹäÿłæŰźæşŢè£ŤåŽđäÿĂäÿłäÿL'åĚČçžĎ `(start, stop, step)` ïijŇæL'ĂæIJL'åĂijéČ¡äijŽèćńåŘĹéĂĆçŽĎçjl'åřŔäžěæžąèűşè¿źçŢŇéŹŘåĹűïijŇ äžŐèĂŇä¡£çŤĺçŽĎæŮűåĂŹéĄ£åĚ■åĞžçŐř `IndexError` åijĆåÿÿãĂĆæŕŤåęĆïijŽ

```
>>> s = 'HelloWorld'
>>> a.indices(len(s))
(5, 10, 2)
>>> for i in range(*a.indices(len(s))):
... print(s[i])
...
W
r
d
>>>
```

1.12 åžŔåĹŮäÿ■åĞžçŐřæňąæŢřæIJĂåď'ŽçŽĎåĚČçť'ă

éŮőécŸ

æĂŐæăůæL'¿åĞžäÿĂäŷłåžŔåĹŮäÿ■åĞžçŐřæňąæŢřæIJĂåď'ŽçŽĎåĚČçť'ääŚćïij§

èğčåĘşæŰźæąĹ

collections.Counter çśžåřśæŸŕäÿŞéŮĺäÿžè£ŹçśžéŮőécŸèĂŇèő¿èőąçŽĎïijŇ åőČçŤŽèĞşæIJL'äÿĂäŷłæIJL'çŤĺçŽĎ most_common() æŰźæşŢçŻť'æŐěçżŹäžĘä¡äç■ŤæąĹãĂĆ

äÿžäžĘæijŤçď'žïijŇåĚĹåĄĞèő¿ä¡äæIJL'äÿĂäŷłå■Ţèŕ■åĹŮèąĺåźűäÿŤæČşæL'¿åĞžåŞłäŷłå■Ţèŕ■åĞžçŐřé

```
words = [
    'look', 'into', 'my', 'eyes', 'look', 'into', 'my', 'eyes',
    'the', 'eyes', 'the', 'eyes', 'the', 'eyes', 'not', 'around',
↪'the',
    'eyes', "don't", 'look', 'around', 'the', 'eyes', 'look', 'into
↪',
    'my', 'eyes', "you're", 'under'
]
from collections import Counter
word_counts = Counter(words)
# åĞžçŐřécŚçŐĞæIJĂéńŸçŽĎ3äŷłå■Ţèŕ■
top_three = word_counts.most_common(3)
print(top_three)
# Outputs [('eyes', 8), ('the', 5), ('look', 4)]
```

èőĺèőž

ä¡IJäÿžè¿ŞåĚëïijŇ Counter åŕźèśąåŔŕäżěæŐěåŘŮäżżæĎŔçŽĎ hashable åžŔåĹŮåŕźèśąãĂĆ åIJĺåžŢåśĆåőđçŐřäŸŁïijŇäÿĂäŷł Counter åŕźèśąåřśæŸŕäÿĂäŷłå■ŮåĚÿïijŇåřĘåĚČçť'ăæŸäărĎåĹřåőČåĞžçŐřçŽĎæňąæŢřäŸŁãĂĆærŤåęĆïijŽ

```
>>> word_counts['not']
1
>>> word_counts['eyes']
8
>>>
```

åęĆædIJä¡ăæČşæL'ŃåŁĺåćđåŁăèőąæŢřïijŇåŔŕäżěçőĂå■ŢçŽĎçŤĺåŁăæşŢïijŽ

```
>>> morewords = ['why','are','you','not','looking','in','my','eyes']
>>> for word in morewords:
... word_counts[word] += 1
...
```

```
>>> word_counts['eyes']
9
>>>
```

æĹŰèĂĚä¡ăåŕŕäžěä¡£çŤĺ `update()` æŰźæşŢïijŽ

```
>>> word_counts.update(morewords)
>>>
```

`Counter` åőđä¿ŃäÿĂäÿłéšIJäÿžäžžç§ěçŽĎçĽźæĂğæŸŕåőČäžňåŕŕäžěå¿ĹåőźæŸŞçŽĎèů§æŢřå■ęè£Řç

```
>>> a = Counter(words)
>>> b = Counter(morewords)
>>> a
Counter({'eyes': 8, 'the': 5, 'look': 4, 'into': 3, 'my': 3, 'around
 ↪': 2,
"you're": 1, "don't": 1, 'under': 1, 'not': 1})
>>> b
Counter({'eyes': 1, 'looking': 1, 'are': 1, 'in': 1, 'not': 1, 'you
 ↪': 1,
'my': 1, 'why': 1})
>>> # Combine counts
>>> c = a + b
>>> c
Counter({'eyes': 9, 'the': 5, 'look': 4, 'my': 4, 'into': 3, 'not':␣
 ↪2,
'around': 2, "you're": 1, "don't": 1, 'in': 1, 'why': 1,
'looking': 1, 'are': 1, 'under': 1, 'you': 1})
>>> # Subtract counts
>>> d = a - b
>>> d
Counter({'eyes': 7, 'the': 5, 'look': 4, 'into': 3, 'my': 2, 'around
 ↪': 2,
"you're": 1, "don't": 1, 'under': 1})
>>>
```

æŕńæŮăçŰŚéŮőïijÑ `Counter` åŕźèśąåIJĺåĞäáźŐæĽĂæIJĽéIJÄèęĄåĹűèąĺæĹŰèĂĚèőąæŢřæŢřæ■őçŽĺ
åIJĺèğčåEşè£ŹçśžéŮőécŸçŽĎæŮűåĂŹä¡ăåžŤèŕěäijŸåĚĹéĂĽ'æŃľåőČïijÑèĂŇäÿ■æŸŕæĽŃåŁĺçŽĎåĹľçŤĺå

1.13 éĂŽè£Ğæ§ŘäÿłåĚşéŤőå■ŮæŐŠåžŔäÿĂäÿłå■ŮåĚÿåĹŮèąĺ

éŮőécŸ

ä¡ăæIJĽäÿĂäÿłå■ŮåĚÿåĹŮèąĺïijÑä¡ăæČşæăźæ■őæ§ŘäÿłæĹŰæ§ŘåĞääÿłå■ŮåĚÿå■ŮæőţæİěæŐŠåžŔè£

èğčåĘşæŰźæąĹ

éĂŽè£Ğä¡£çŤĺ operator æĺąåİŮçŽĎ itemgetter åĞ¡æŢřïijŇåŔřäžěéİđåÿÿåőźæŸŞçŽĎæŐŠåžŔè£ŹæăůçŽĎæŢřæ■őçżŞæđĎãĂĆ åĄĞèő¿ä¡ääžŐæŢřæ■őåžŞäÿ■æčĂçť´ćåĞžæİěç¡ŚçńŹäijŽåŚŸä£ąæĄŕåĹŮèąĺïijŇåźűäÿŤäžěäÿŃåĹŮçŽĎæŢřa

```
rows = [
    {'fname': 'Brian', 'lname': 'Jones', 'uid': 1003},
    {'fname': 'David', 'lname': 'Beazley', 'uid': 1002},
    {'fname': 'John', 'lname': 'Cleese', 'uid': 1001},
    {'fname': 'Big', 'lname': 'Jones', 'uid': 1004}
]
```

æăźæ■őäżžæĎŔçŽĎå■ŮåĚÿå■ŮæőţæİěæŐŠåžŔè¿ŞåĚěçżŞæđIJèąŇæŸŕå¿ĹåőźæŸŞåőđçŐřçŽĎïijŇäžč

```
from operator import itemgetter
rows_by_fname = sorted(rows, key=itemgetter('fname'))
rows_by_uid = sorted(rows, key=itemgetter('uid'))
print(rows_by_fname)
print(rows_by_uid)
```

äžčçăĄçŽĎè¿ŞåĞžåęĆäÿŃïijŽ

```
[{'fname': 'Big', 'uid': 1004, 'lname': 'Jones'},
{'fname': 'Brian', 'uid': 1003, 'lname': 'Jones'},
{'fname': 'David', 'uid': 1002, 'lname': 'Beazley'},
{'fname': 'John', 'uid': 1001, 'lname': 'Cleese'}]
[{'fname': 'John', 'uid': 1001, 'lname': 'Cleese'},
{'fname': 'David', 'uid': 1002, 'lname': 'Beazley'},
{'fname': 'Brian', 'uid': 1003, 'lname': 'Jones'},
{'fname': 'Big', 'uid': 1004, 'lname': 'Jones'}]
```

itemgetter() åĞ¡æŢřäź§æŤræŇĄåď´ŽäÿłkeysïijŇæŕŤåęĆäÿŃéİćçŽĎäžčçăĄ

```
rows_by_lfname = sorted(rows, key=itemgetter('lname','fname'))
print(rows_by_lfname)
```

äijŽäžğçŤ§åęĆäÿŃçŽĎè¿ŞåĞžïijŽ

```
[{'fname': 'David', 'uid': 1002, 'lname': 'Beazley'},
{'fname': 'John', 'uid': 1001, 'lname': 'Cleese'},
{'fname': 'Big', 'uid': 1004, 'lname': 'Jones'},
{'fname': 'Brian', 'uid': 1003, 'lname': 'Jones'}]
```

èőÍèőž

åIJÍäÿŁéİćä¿Ńå■Řäÿ■ïijŇ rows èćńäijåéĂŠçżŹæŐěåŔŮäÿĂäÿłåĚşéŤőå■ŮåŔĆæŢřçŽĎ sorted() åĘĚç¡őåĞ¡æŢřãĂĆ è£ŹäÿłåŔĆæŢřæŸŕ callable çśžåđŇïijŇåźűäÿŤäžŐ rows

äÿ■æŐěåŔŮäÿĂäÿłå■ŢäÿĂåĚČçť'äïijŇçĎűåŘŐè£ŤåŽđèćńçŤĺæĺěæŐŠåžŔçŽĎåĂijãĂĆ `itemgetter()` åĞ¡æŢřåřśæŸŕèť§èť'čåĹŽåžžè£ŹäÿŁ `callable` åŕźèśąçŽĎãĂĆ

`operator.itemgetter()` åĞ¡æŢřæIJL'äÿĂäÿłèćńrowsäÿ■çŽĎèőřå¡ŢçŤĺæĺěæ§ěæL'¿åĂijçŽĎçť'ćåï äÿĂäÿłæŢť'å¡ćåĂijæĹŰèĂĚäžża¡ŢèČ¡åd'§äijååĚěäÿĂäÿłåŕźèśąçŽĎ `__getitem__()` æŰźæşŢçŽĎåĂijãĂĆ åęĆæđIJä¡ăäijååĚěåd'Žäÿłçť'ćåijŢåŔĆæŢřçžŹ `itemgetter()` ïijŇåőČçŤ§æĹŘçŽĎ `callable` åŕźèśąäijŽè£ŤåŽđäÿĂäÿłåŇĚåŘńæL'ĂæIJL'åĚČçť'ääĂijçŽĎåĚČçžĎïijŇ åźűäÿŤ `sorted()` åĞ¡æŢřäijŽæăźæ■őè£ŹäÿłåĚČçžĎäÿ■åĚČçť'ăéąžåžŔåŐżæŐŠåžŔãĂĆ ä¡Ęä¡ăæČşèęĄåŘŇæŮűåIJĺåĞăäÿłå■Ůæőţäÿŀéĺćè£ŽèąŇæŐŠåžŔ(æŕŤåęĆéĂŽè£ĞåğŞåŠŇåŘ■æĺěæŐŠåžŔïi

`itemgetter()` æIJL'æŮűåĂŹäź§åŔŕäžěçŤĺ `lambda` èąĺè¿¿åijŔäžčæŽ£ïijŇærŤåęĆïijŽ

```
rows_by_fname = sorted(rows, key=lambda r: r['fname'])
rows_by_lfname = sorted(rows, key=lambda r: (r['lname'],r['fname']))
```

è£Źçğ■æŰźæąĹäź§äÿ■éŤŹãĂĆä¡ĘæŸŕïijŇä¡£çŤĺ `itemgetter()` æŰźåijŔäijŽè£ŘèąŇçŽĎçĺ■å¿ő£ńçĆźãĂĆåŽăæ■d'ïijŇåęĆæđIJä¡ăåŕźæĂğèČ¡èęĄæśĆærŤè¿ČénŸçŽĎèŕĺåřś `itemgetter()` æŰźåijŔãĂĆ

æIJĂåŘŐïijŇäÿ■èęĄå£ŸäžĘè£ŹèŁĆäÿ■åśŢçd'žçŽĎæŁĂæIJŕäź§åŘŇæăůéĂĆçŤĺäžŐ `min()` åŠŇ `max()` ç■L'åĞ¡æŢřãĂĆærŤåęĆïijŽ

```
>>> min(rows, key=itemgetter('uid'))
{'fname': 'John', 'lname': 'Cleese', 'uid': 1001}
>>> max(rows, key=itemgetter('uid'))
{'fname': 'Big', 'lname': 'Jones', 'uid': 1004}
>>>
```

1.14 æŐŠåžŔäÿ■æŤræŇĄåŐ§çŤ§ærŤè¿ČçŽĎåŕźèśą

éŮőécŸ

ä¡ăæČşæŐŠåžŔçśžåđŇçŻÿåŘŇçŽĎåŕźèśąïijŇä¡ĘæŸŕäžŰäžňäÿ■æŤræŇĄåŐ§çŤ§çŽĎærŤè¿ČæŞ■ä¡IJã

èğčåĘşæŰźæąĹ

åĘĚç¡őçŽĎ `sorted()` åĞ¡æŢræIJL'äÿĂäÿłåĚşéŤőå■ŮåŔĆæŢř `key` ïijŇåŔŕäžěäijăåĚěäÿĂäÿł `callable` åŕźèśąçŹåőČïijŇ è£Źäÿł `callable` åŕźèśąåŕźærŔäÿłäijăåĚěçŽĎåŕźèśąè£ŤåŽđäÿĂäÿłåĂijïijŇè£ŹäÿłåĂijäijŽèćń `sorted` çŤĺæĺěæŐŠåžŔè£ŹäžŽåŕźèśąãĂĆ ærŤåęĆïijŇåęĆæđIJä¡ăåIJĺåžŤçŤĺçĺŃåžŔéĞŇéĺćæIJL'äÿĂäÿł `User` åőđä¿ŃåžŔåĹŮïijŇåźűäÿŤä¡ăåÿŇæIJŽéĂŽè£ĞäžŰäžňçŽĎ `user_id` åśđæĂğè£ŽèąŇæŐŠåžŔïijŇ ä¡ăåŔŕäžěæŔŘä¿ŽäÿĂäÿłäžě `User` åőđä¿Ńä¡IJäÿžè¿ŞåĚěåźűè¿ŞåĞžåŕźåžŤ `user_id` åĂijçŽĎ `callable` åŕźèśąãĂĆærŤåęĆïijŽ

```
class User:
    def __init__(self, user_id):
        self.user_id = user_id

    def __repr__(self):
        return 'User({})'.format(self.user_id)


def sort_notcompare():
    users = [User(23), User(3), User(99)]
    print(users)
    print(sorted(users, key=lambda u: u.user_id))
```

åŔęåď Űäÿ Ăçğ■æŰźåijŔæŸŕä¡£çŤĺ operator.attrgetter() æİěäžčæŻ£lambdaåĞ¡æŢřïijŽ

```
>>> from operator import attrgetter
>>> sorted(users, key=attrgetter('user_id'))
[User(3), User(23), User(99)]
>>>
```

èőĺèőž

éĂĽ'æŃl'ä¡£çŤĺlambdaåĞ¡æŢřæĹŰèĂĚæŸŕ attrgetter() åŔŕèČ¡åŔŰåĘşäžŐäÿłäžžåŰIJåě¡ãĂĆ ä¡ĘæŸŕïijŇ attrgetter() åĞ¡æŢřéĂŽåÿÿäijŽè£ŘèąŇçŽĎå£ńçĆźïijŇåźűäÿŤè£ŸèČ¡åŘŇæŮűåĚĄèőÿåď Žäÿłå■Ůæőţè£ŻèąŇærŤè¿ČãĂ è£Źäÿłèů§ operator.itemgetter() åĞ¡æŢřä¡IJçŤĺäžŐå■ŮåĚÿçśžåđŃå¿Ĺçśžäijij(åŔĆèĂČ1.13åřŔèŁĆ ä¿ŃåęĆïijŇåęĆæđIJ User åőđä¿Ńè£ŸæIJĽäÿĂäÿł first_name åŠŇ last_name åśđæĂğïijŇéĆčäźĹåŔŕäžěåŘŠäÿŃéİćè£ŹæăůæŐŠåžŔïijŽ

```
by_name = sorted(users, key=attrgetter('last_name', 'first_name'))
```

åŘŇæăůéIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇè£ŹäÿĂåřŔèŁĆçŤĺåĹřçŽĎæŁĂæIJŕåŘŇæăůéĂĆçŤĺäžŐåČŔ min() åŠŇ max() äźŃçśżçŽĎåĞ¡æŢřãĂĆærŤåęĆïijŽ

```
>>> min(users, key=attrgetter('user_id')
User(3)
>>> max(users, key=attrgetter('user_id')
User(99)
>>>
```

1.15 éĂŽè£Ğæ§Řäÿłå■Ůæőţår Ęèőřå¡ŢåĹĘçżĎ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłå■ŮåĚÿæĹŰèĂĚåőđä¿ŃçŽĎåžŔåĹŮïijŇçĎűåŘŐä¡ăæČşæăźæ■őæ§Řäÿł çL'źåőŽçŽĎå■Ů
`date` æÌěåĹĘçżĎè£■äžčèő£éŮőãĂĆ

èğčåĘşæŰźæąĹ

`itertools.groupby()` åĞ¡æŢřåŕźäžŐè£ŹæăůçŽĎæŢřæ■őåĹĘçżĎæŞ■ä¡IJéÌđåÿÿåőđçŤĺãĂĆ äÿžäžĘæijŤçd'žïijŇåĄĞèő¿ä¡ăåůšçżŔæIJL'äžĘäÿŃåĹŮçŽĎå■ŮåĚÿåĹŮèąĺïijŽ

```
rows = [
    {'address': '5412 N CLARK', 'date': '07/01/2012'},
    {'address': '5148 N CLARK', 'date': '07/04/2012'},
    {'address': '5800 E 58TH', 'date': '07/02/2012'},
    {'address': '2122 N CLARK', 'date': '07/03/2012'},
    {'address': '5645 N RAVENSWOOD', 'date': '07/02/2012'},
    {'address': '1060 W ADDISON', 'date': '07/02/2012'},
    {'address': '4801 N BROADWAY', 'date': '07/01/2012'},
    {'address': '1039 W GRANVILLE', 'date': '07/04/2012'},
]
```

çŐřåIJĺåĄĞèő¿ä¡ăæČşåIJĺæŇL'dateåĹĘçżĎåŘŐçŽĎæŢřæ■őåİŮäÿŁè£ŻèąŇè£■äžčãĂĆäÿžäžĘè£ŹæăůåĹ `date`)æŐŠåžŔïijŇ çĎűåŘŐèřČçŤĺ `itertools.groupby()` åĞ¡æŢřïijŽ

```
from operator import itemgetter
from itertools import groupby

# Sort by the desired field first
rows.sort(key=itemgetter('date'))
# Iterate in groups
for date, items in groupby(rows, key=itemgetter('date')):
    print(date)
    for i in items:
        print(' ', i)
```

è£ŘèąŇçżŞædIJïijŽ

```
07/01/2012
  {'date': '07/01/2012', 'address': '5412 N CLARK'}
  {'date': '07/01/2012', 'address': '4801 N BROADWAY'}
07/02/2012
  {'date': '07/02/2012', 'address': '5800 E 58TH'}
  {'date': '07/02/2012', 'address': '5645 N RAVENSWOOD'}
  {'date': '07/02/2012', 'address': '1060 W ADDISON'}
07/03/2012
  {'date': '07/03/2012', 'address': '2122 N CLARK'}
```

```
07/04/2012
  {'date': '07/04/2012', 'address': '5148 N CLARK'}
  {'date': '07/04/2012', 'address': '1039 W GRANVILLE'}
```

èőÍèőž

groupby() åĞ¡æŢřæĽńæŔŘæŢť äŷłåžŘåĹŮåźűäŷŤæ§ěæĽ¿è£đçż■çŻŷåŘŇåĂij(æĹŰèĂĚæăźæ■őæŇ
åIJÍæŕŘæňąè£■äžčçŽĎæŮűåĂŹïijŇáőČäijŽè£ŤåŻđäŷĂäŷłåĂijåŠŇäŷĂäŷłè£■äžčåŹÍåŕźèšąäijŇ
è£ŹäŷłèŁ■äžčåŹÍåŕźèšąåŘŕäžěçŤ§æĹŘåĚČçť ăåĂijåĚÍéČÍç■L'äžŐäŷŁéÍćéĆčäŷłåĂijçŽĎçżĎäŷ■æL'ĂæIJL'åŕ

äŷĂäŷłéÍđåŷÿéĞ■èęĄçŽĎåĞĘåď'Ğæ■ěéłď'æŸŕèęĄæăźæ■őæŇĞåőŽçŽĎå■ŮæőţåřĕæŢřæ■őæŐŠåžŔãĂ
åŻăäŷž groupby() äžĚäżĚæčĂæ§ěè£đçż■çŽĎåĚČçť ăïijŇåęĆæđIJäžŃåĚĹáźűæšąæIJL'æŐŠåžŘåőŇæĹŘç

åęĆæđIJä¡ăäżĚäżĚåŘłæŸŕæČşæăźæ■ődateå■ŮæőţåřĕæŢřæ■őåĹĘçżĎåĹřäŷĂäŷłåď'ğçŽĎæŢřæ■őçżŞæć
éĆčäźĹä¡ăæIJĂåě¡ä¡£çŤÍ defaultdict() æÍěæđĎåżžäŷĂäŷłåď'ŽåĂijå■ŮåĚÿïijŇåĚşäžŐåď'ŽåĂijå■ŮåĚÿ

```
from collections import defaultdict
rows_by_date = defaultdict(list)
for row in rows:
    rows_by_date[row['date']].append(row)
```

è£ŹæăůçŽĎèŕÍä¡ăåŘŕäžěå¿Ĺè¡żæİ¿çŽĎåřśèČ¡åŕźæŕŘäŷłæŇĞåőŽæŮěæIJ§èő£éŮőåŕźåžŤçŽĎèőřå¡ŢïijŽ

```
>>> for r in rows_by_date['07/01/2012']:
...  print(r)
...
{'date': '07/01/2012', 'address': '5412 N CLARK'}
{'date': '07/01/2012', 'address': '4801 N BROADWAY'}
>>>
```

åIJÍäŷŁéÍćè£ŹäŷłäżÑå■Řäŷ■ïijŇæĹŚäžňæšąæIJL'å£ĚèęĄåĚĹåřĘèőřå¡ŢæŐŠåžŔãĆćŽăæ■d'ïijŇåęĆæđ
è£Źçğ■æŰźåijŔäijŽærŤåĚĹæŐŠåžŘçĎűåŘŐåĘ■éĂŽè£Ğ groupby()
åĞ¡æŢřè£■äžčçŽĎæŰźåijŔè£ŘèąŇå¿Ůå£ńäŷĂäžŽãĂĆ

1.16 è£Ğæžď'åžŔåĹŮåĚČçť'ă

éŮőécŸ

ä¡ăæIJL'äŷĂäŷłæŢřæ■őåžŔåĹŮïijŇæČşåĹl'çŤÍäŷĂäžŽèğĎåĹŹäžŐäŷ■æŘŘåŘŰåĞžéIJĂèęĄçŽĎåĂijæĺ

èğčåĘşæŰźæąĹ

æIJĂçőĂå■ŢçŽĎè£Ğæžď'åžŔåĹŮåĚČçť'ăçŽĎæŰźæşŢářśæŸŕä¡£çŤÍåĹŮèąÍæŐÍåŕijãĂĆærŤåęĆïijŽ

```
>>> mylist = [1, 4, -5, 10, -7, 2, 3, -1]
>>> [n for n in mylist if n > 0]
[1, 4, 10, 2, 3]
>>> [n for n in mylist if n < 0]
[-5, -7, -1]
>>>
```

ä¡£çŤĺåĹŮèąĺæŐĺåŕijçŽĎäÿĂäÿłæ¡IJåIJĺçijžéŹůåřśæŸŕåęĆæđIJè¿ŞåĚéíđåÿÿåď ğçŽĎæŮűåĂŹäijŽäžğç
åęĆæđIJä¡ääŕźåĘĚå■ŸæŕŤè¿ČæŢŔæĎ§ïijŇéĆčäźĹä¡ääŔŕäžěä¡£çŤĺçŤ§æĹŘåŹĺèąĺè¿¿åijŔè£■äžčäžğçŤ§è£Ğ

```
>>> pos = (n for n in mylist if n > 0)
>>> pos
<generator object <genexpr> at 0x1006a0eb0>
>>> for x in pos:
... print(x)
...
1
4
10
2
3
>>>
```

æIJL'æŮűåĂŹïijŇè£Ğæžđ'èğĎåĹŹæŕŤè¿Čåđ'■æİĆïijŇäÿ■èČ¡çőĂå■ŢçŽĎåIJĺåĹŮèąĺæŐĺåŕijæĹŰèĂĚç
æŕŤåęĆïijŇåĄĞèő¿è£Ğæžđ'çŽĎæŮűåĂŹéIJĂèęĄåđ'ĎçŘĘäÿĂäžŽåijĆåÿÿæĹŰèĂĚåĚűäžŰåđ'■æİĆæČĚåĘ
çĎűåŘŐä¡£çŤĺåĘĚåžžçŽĎ `filter()` åĞ¡æŢřãĂĆçđ'žä¿ŃåęĆäÿŃïijŽ

```
values = ['1', '2', '-3', '-', '4', 'N/A', '5']
def is_int(val):
    try:
        x = int(val)
        return True
    except ValueError:
        return False
ivals = list(filter(is_int, values))
print(ivals)
# Outputs ['1', '2', '-3', '4', '5']
```

`filter()` åĞ¡æŢřåĹŽåžžäžĘäÿĂäÿłè£■äžčåŹĺïijŇåŽăæ■đ'åęĆæđIJä¡ăæČşå¿ŮåĹřäÿĂäÿłåĹŮèąĺçŽĎè
`list()` åŐżè¡ňæ■ćãĂĆ

èőĺèőž

åĹŮèąĺæŐĺåŕijåŠŇçŤ§æĹŘåŹĺèąĺè¿¿åijŔéĂŽåÿÿæČĚåĘţäÿŇæŸŕè£Ğæžđ'æŢřæ■őæIJĂçőĂå■ŢçŽĎæŰ
åĚűåőđåőČäžňè£ŸèČ¡åIJĺè£Ğæžđ'çŽĎæŮűåĂŹè¡ňæ■ćæŢřæ■őãĂĆæŕŤåęĆïijŽ

```
>>> mylist = [1, 4, -5, 10, -7, 2, 3, -1]
>>> import math
>>> [math.sqrt(n) for n in mylist if n > 0]
```

```
[1.0, 2.0, 3.1622776601683795, 1.4142135623730951, 1.
↪7320508075688772]
>>>
```

è£ĞæżďæŞ■ä¡IJçŽĎäÿĂäÿłåŔŸçğ■åřśæŸŕåřĘäÿ■çňęåŘĹæİąäžűçŽĎåĂijçŤĺæŰřçŽĎåĂijäžčæŻ£ïijŃèĂ
æŕŤåęĆïijŃåIJĺäÿĂåĹŮæŢřæ■őäÿ■ä¡ăåŔŕèČ¡äÿ■äżĚæČşæL'¿åĹřæ■čæŢřïijŃèĂŇäÿŤè£ŸæČşåřĘäÿ■æŸŕæ■
éĂŽè£ĞåřĘè£Ğæżď æİąäžűæŤ¿åĹřæİąäžűèąĺè¿¿åijŔäÿ■åŐżïijŃåŔŕäžěå¿Ĺåőźæ ŸŞçŽĎèğčåĘşè£ŹäÿłéŮőéćŸ

```
>>> clip_neg = [n if n > 0 else 0 for n in mylist]
>>> clip_neg
[1, 4, 0, 10, 0, 2, 3, 0]
>>> clip_pos = [n if n < 0 else 0 for n in mylist]
>>> clip_pos
[0, 0, -5, 0, -7, 0, 0, -1]
>>>
```

åŔęåď Űäÿ Ăäÿłå Äijå¿ŮåĚşæşĺçŽĎè£Ğæżď åůěåĚůåřśæŸŕ itertools.compress() ïijŃ åőČäžěäÿĂäÿł iterable årźèśąåŠŇäÿĂäÿłçŻÿåŕźåžŤçŽĎ Boolean éĂL'æŃľ åŹĺåžŔåĹŮä¡IJäÿžè¿ŞåĚěåŔĆæŢřãĂĆ çĎűåŘŐè¿ŞåĞž iterable årźèśąäÿ■årźåžŤéĂL'æŃľ åŹĺäÿž True çŽĎåĚČçť ăãĂĆ å¡Şä¡ăéIJĂèęĄçŤĺåŔęåď Űäÿ ĂäÿłçŻÿåĚşèĄŤçŽĎåžŔåĹŮæİěè£Ğæżď æ§RäÿłåžŕåĹŮçŽĎæŮűåĂŹïijŃè£Źä
æŕŤåęĆïijŃåĄĞåęĆçŐřåIJĺä¡ăæIJL'äÿŃéİćäÿď åĹŮæŢřæ■őïijŽ

```
addresses = [
    '5412 N CLARK',
    '5148 N CLARK',
    '5800 E 58TH',
    '2122 N CLARK'
    '5645 N RAVENSWOOD',
    '1060 W ADDISON',
    '4801 N BROADWAY',
    '1039 W GRANVILLE',
]
counts = [ 0, 3, 10, 4, 1, 7, 6, 1]
```

çŐřåIJĺä¡ăæČşåřĘéĆčäžŻåŕźåžŤ count åĂijåď ğäžŐ5çŽĎåIJřåİĂåĚĺéĆĺè¿ŞåĞžïijŃéĆčäźĹä¡ääŔŕäžěè£

```
>>> from itertools import compress
>>> more5 = [n > 5 for n in counts]
>>> more5
[False, False, True, False, False, True, True, False]
>>> list(compress(addresses, more5))
['5800 E 58TH', '4801 N BROADWAY', '1039 W GRANVILLE']
>>>
```

è£ŹéĞŃçŽĎåĚşéŤőçĆźåIJĺäžŐåĚĹåĹŻåžžäÿĂäÿł Boolean åžŔåĹŮïijŃæŃĞçď žåŞłäžŻåĚČçť ăåď ■åŘĹæİąäžűãĂĆ çĎűåŘŐ compress() åĞ¡æŢřæăźæ■őè£ŹäÿłåžŔåĹŮåŐźéĂL'æŃľè¿ŞåĞžåŕźåžŤä¡■ç¡őäÿž True çŽĎåĚČçť ăãĂĆ

åŠŃ filter() åĞ¡æŢřçśžäijijïijŃ compress() äźß§æŸŕè£ŤåŻđçŽĎäÿĂäÿłè£■äžčåŹĺãĂĆåŻăæ■d'ïij
éĆčäźĹä¡ääIJĂèęĄä¡£çŤĺ list() æİěåŕĘçżŞæđIJè¡ñæ■ćäÿžåĹŮèąĺçśžåđŃãĂĆ

1.17 äžŐå■ŮåĚÿäÿ■æŔŘåŔŰå■ŘéŻĘ

éŮőécŸ

ä¡ăæČşæđĎéĂăäÿĂäÿłå■ŮåĚÿïijŇåőČæŸŕåŔęåď´ŰäÿĂäÿłå■ŮåĚÿçŽĎå■ŘéŻĘãĂĆ

èğčåĘşæŰźæąĹ

æIJĂçőĂå■ŢçŽĎæŰźåijŔæŸŕä¡£çŤĺå■ŮåĚÿæŐĺåŕijãĂĆæŕŤåęĆïijŽ

```
prices = {
    'ACME': 45.23,
    'AAPL': 612.78,
    'IBM': 205.55,
    'HPQ': 37.20,
    'FB': 10.75
}
# Make a dictionary of all prices over 200
p1 = {key: value for key, value in prices.items() if value > 200}
# Make a dictionary of tech stocks
tech_names = {'AAPL', 'IBM', 'HPQ', 'MSFT'}
p2 = {key: value for key, value in prices.items() if key in tech_
↪names}
```

èőĺèőž

åď´gåď´ŽæŢřæČĚåĘţäÿŃå■ŮåĚÿæŐĺåŕijèČ¡åĄŽåĹřçŽĎïijŇéĂŽè£ĞåĹŻåžžäÿĂäÿłåĚČçżĎåžŔåĹŮçĎű
`dict()` åĞ¡æŢřäź§èČ¡åőđçŐřãĂĆæŕŤåęĆïijŽ

```
p1 = dict((key, value) for key, value in prices.items() if value >␣
↪200)
```

ä¡ĘæŸŕïijŇå■ŮåĚÿæŐĺåŕijæŰźåijŔèąĺæĎŔæŻť´æÿĚæŹřïijŇåźűäÿŤåőđéŹĚäÿŁäź§äijŽè£ŘèąŇçŽĎæŻť´
(åIJĺè£ŹäÿłäżŃå■ŘäÿïijŇåőđéŹĚæţŃèŕŢåĞääźŐæŕŤ `dcit()`
åĞ¡æŢřæŰźåijŔå£ńæŢť´æŢť´äÿĂåĂ■)ãĂĆ

æIJĽæŮűåĂŹåőŇæĹŘåŘŇäÿĂäżűäžŇäijŽæIJĽåď´Žçğ■æŰźåijŔãĂĆæŕŤåęĆïijŇçňňäžŇäÿłäżŃå■ŘçĺŃ

```
# Make a dictionary of tech stocks
tech_names = { 'AAPL', 'IBM', 'HPQ', 'MSFT' }
p2 = { key:prices[key] for key in prices.keys() & tech_names }
```

ä¡ĘæŸŕïijŇè£ŘèąŇæŮűéŮť´æţŃèŕŢçżŞæđIJæŸ¿çď´žè£ŹçğæŰźæąĹåď´gæęĆæŕŤçňňäÿĂçğ■æŰźæąĹæ
åęĆæđIJåŕźçĺŃåžŔè£ŘèąŇæĂğèČ¡èęĄæśĆæŕŤè¿ČéńŸçŽĎèŕĺïijŇéIJĂèęĄèĽśçĆźæŮűéŮť´åŐżåĄŽèőąæŮűą
åĚşäžŐæŻť´åď´ŽèőąæŮűåŠŇæĂğèČ¡æţŃèŕŢïijŇåŔŕäžěåŔĆèĂČ14.13åŕŘèŁĆ

1.18 æŸăåřĎåŘ■çğřåĹřåžŕåĹŮåĚČçťă

éŮőécŸ

ä¡ăæIJL'äÿĂæőţéĂŽè£ĞäÿŃæăĞèő£éŮőåĹŮèąĺæĹŰèĂĚåĚČçżĎäÿ■åĚČçť'ăçŽĎäžčçăĄïijŇä¡ĘæŸŕè£Ź
äžŐæŸŕä¡ăæČşéĂŽè£ĞåŘ■çğřæİěèő£éŮőåĚČçť'ăãĂĆ

èğčåĘşæŰźæąĹ

`collections.namedtuple()` åĞ¡æŢřéĂŽè£Ğä¡£çŤĺäÿĂäÿłæŹőéĂŽçŽĎåĚČçżĎåŕźèśąæİěåÿőä¡ă
è£ŹäÿłåĞ¡æŢřåőđéŹĚäÿŁæŸŕäÿĂäÿłè£ŤåŻđPythonäÿ■æăĞåĞĘåĚČçżĎçśžåđŇå■ŘçśžçŽĎäÿĂäÿłåůěåŐĆæ
ä¡ăéIJĂèęĄäijăéĂŠäÿĂäÿłçśžåđŇåŘ■åŠŇä¡ăéIJĂèęĄçŽĎå■Ůæőţ¢ŹåőČïijŇçĎűåŘŐåőČåŕšäijŽè£ŤåŻđäÿĂ
äžčçăĄçd'žä¿ŃïijŽ

```
>>> from collections import namedtuple
>>> Subscriber = namedtuple('Subscriber', ['addr', 'joined'])
>>> sub = Subscriber('jonesy@example.com', '2012-10-19')
>>> sub
Subscriber(addr='jonesy@example.com', joined='2012-10-19')
>>> sub.addr
'jonesy@example.com'
>>> sub.joined
'2012-10-19'
>>>
```

åř¡çőą `namedtuple` çŽĎåőđä¿ŃçIJŃètůæİěåČŔäÿĂäÿłæŹőéĂŽçŽĎçśžåőđä¿ŃïijŇä¡ĘæŸŕåőČèů§åĚČ
ærŤåęĆïijŽ

```
>>> len(sub)
2
>>> addr, joined = sub
>>> addr
'jonesy@example.com'
>>> joined
'2012-10-19'
>>>
```

åŚ¡åŘ■åĚČçżĎçŽĎäÿĂäÿłäÿżèęĄçŤĺéĂŤæŸŕåřĘä¡ăçŽĎäžčçăĄäžŐäÿŃæăĞæŞ■ä¡IJäÿ■èğčèĎšåĞžæİě
åŻăæ■d'ïijŇåęĆædIJä¡ääžŐæŢřæ■őåžŞèřČçŤĺäÿ■è£ŤåŻđäžĘäÿĂäÿłå¿Ĺåd'ğçŽĎåĚČçżĎåĹŮèąĺïijŇéĂŽè£Č
å¡Şä¡ääIJĺèąĺäÿ■æůżåŁääžĘæŰřçŽĎåĹŮçŽĎæŮűåĂŹä¡ăçŽĎäžčçăĄåŔŕèČ¡åŕšäijŽåĞžéŤŹäžĘãĂĆä¡ĘæŸŕå

äÿžäžĘèŕť'æŸŐæÿĚæěŽïijŇäÿŃéİćæŸŕä¡£çŤĺæŹőéĂŽåĚČçżĎçŽĎäžčçăĄïijŽ

```
def compute_cost(records):
total = 0.0
for rec in records:
    total += rec[1] * rec[2]
return total
```

äÿŃæăĞæŞ■ä¡IJéĂŽåÿÿäijŽèőľäžčçăĄèąÍæĎŔäÿ■æÿĚæŹřïijŇåźűäÿŤéÍđåÿÿä¿ÌètŰèőřå¡ŢçŽĎçżŞæđĎä
äÿŃéÍćæŸřä¡£çŤĺåŚ¡åŘ■åĚČçżĎçŽĎçĽĹæIJňïijŽ

```
from collections import namedtuple

Stock = namedtuple('Stock', ['name', 'shares', 'price'])
def compute_cost(records):
    total = 0.0
    for rec in records:
        s = Stock(*rec)
    total += s.shares * s.price
    return total
```

èőÍèőž

åŚ¡åŘ■åĚČçżĎåŔęäÿĂäÿłçŤĺéĂŤåřśæŸřä¡IJäÿžå■ŮåĚÿçŽĎæŻ£äžčïijŇåŻääÿžå■ŮåĚÿå■ŸåĆĺéIJĂèęĄ
åęĆæđIJä¡ăéIJĂèęĄæđĎåżžäÿĂäÿłéÍđåÿÿåď´gçŽĎåŇĚåŘńå■ŮåĚÿçŽĎæŢřæ■őçżŞæđĎïijŇéĆčäźĹä¡£çŤĺåŚ
ä¡ĘæŸŕéIJĂèęĄæşÍæĎŔçŽĎæŸřïijŇäÿ■åČŔå■ŮåĚÿéĆčæăůïijŇäÿĂäÿłåŚ¡åŘ■åĚČçżĎæŸřäÿ■åŔŕæŻť´æŤźç

```
>>> s = Stock('ACME', 100, 123.45)
>>> s
Stock(name='ACME', shares=100, price=123.45)
>>> s.shares = 75
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>>
```

åęĆæđIJä¡ăçIJ§çŽĎéIJĂèęĄæŤźåŔŸçĎűåŘŐçŽĎåśđæĂğïijŇéĆčäźĹåŔŕäżěä¡£çŤĺåŚ¡åŘ■åĚČçżĎåőđä¿
_replace() æŰźæşŢïijŇ åőČäijŽåĹŻåžžäÿĂäÿłåĚÍæŰřçŽĎåŚ¡åŘ■åĚČçżĎåźűårĘåŕźåžŤçŽĎå■Ůæ§ţŤĺa

```
>>> s = s._replace(shares=75)
>>> s
Stock(name='ACME', shares=75, price=123.45)
>>>
```

_replace() æŰźæşŢè£ŸæIJL'äÿĂäÿłå¿ĹæIJL'çŤĺçŽĎçL'źæĂğåřśæŸřå¡Şä¡äçŽĎåŚ¡åŘ■åĚČçżĎæŃě
åőČæŸřäÿĂäÿłéÍđåÿÿæŰźä¿£çŽĎåąńåĚĚæŢřæ■őçŽĎæŰźæşŢãĂĆ
ä¡ăåŕźäžěåĚĹåĹŻåžžäÿĂäÿłåŃĚåŘńçijžçIJĄåĂijçŽĎåŐ§ðNåĚČçżĎïijŇçĎűåŘŐä¡£çŤĺ
_replace() æŰźæşŢåĹŻåžžæŰřçŽĎåĂijècńæŻť´æŰrè£ĞçŽĎåőđä¿ÑãĂĆærŤåęĆïijŽ

```
from collections import namedtuple

Stock = namedtuple('Stock', ['name', 'shares', 'price', 'date',
↪'time'])

# Create a prototype instance
stock_prototype = Stock('', 0, 0.0, None, None)
```

```
# Function to convert a dictionary to a Stock
def dict_to_stock(s):
    return stock_prototype._replace(**s)
```

äÿŃéÍćæŸŕåőČçŽĎä¡£çŤÍæŰźæşŢïijŽ

```
>>> a = {'name': 'ACME', 'shares': 100, 'price': 123.45}
>>> dict_to_stock(a)
Stock(name='ACME', shares=100, price=123.45, date=None, time=None)
>>> b = {'name': 'ACME', 'shares': 100, 'price': 123.45, 'date':
↪'12/17/2012'}
>>> dict_to_stock(b)
Stock(name='ACME', shares=100, price=123.45, date='12/17/2012',␣
↪time=None)
>>>
```

æIJĂåŘŐèęĄèŕť'çŽĎæŸřïijŇåęĆæđIJä¡ăçŽĎçŻőæăĞæŸŕåőŽäźL'äÿĂäÿłéIJĂèęĄæŻť'æŰřå¿Ĺåď'Žåőđä¿è£ŹæŮűåĂŹä¡ååžŤèŕěèĂČèŹŚåőŽäźL'äÿĂäÿłåŇĚåŘń `__slots__` æŰźæşŢçŽĎçśż(åŔĆèĂČ8.4åŕŔèŁĆ)ãĂĆ

1.19 è¡ñæ■ćåźűåŘŇæŮűèőąçőŮæŢřæ■ő

éŮőécŸ

ä¡ăéIJĂèęĄåIJÍæŢřæ■őåžŔåĹŮäÿŁæL'ğèąŇèĄŽéŻĘåĞ¡æŢř(æŕŤåęĆ `sum()` , `min()` , `max()`)ïijŇ ä¡ĘæŸŕéęŰåĚĹä¡ăéIJĂèęĄåĚĹè¡ñæ■ćæĹŰèĂĚè£Ğæżđ'æŢřæ■ő

èğčåĘşæŰźæąĹ

äÿĂäÿłéIđåÿÿäijŸéŻĚçŽĎæŰźåijŔåŐżçżŞåŘĹæŢřæ■őèőąçőŮäÿŐè¡ñæ■ćåřśæŸŕä¡£çŤÍäÿĂäÿłçŤ§æĹŘæŕŤåęĆïijŇåęĆæđIJä¡ăæČşèőąçőŮåźşæŰźåŠŇïijŇåŔŕäžěåČŔäÿŃéÍćè£ŹæăůåĄŽïijŽ

```
nums = [1, 2, 3, 4, 5]
s = sum(x * x for x in nums)
```

äÿŃéÍćæŸŕæŻť'åď'ŽçŽĎä¿Ńå■ŘïijŽ

```
# Determine if any .py files exist in a directory
import os
files = os.listdir('dirname')
if any(name.endswith('.py') for name in files):
    print('There be python!')
else:
    print('Sorry, no python.')
# Output a tuple as CSV
s = ('ACME', 50, 123.45)
```

```
print(','.join(str(x) for x in s))
# Data reduction across fields of a data structure
portfolio = [
    {'name':'GOOG', 'shares': 50},
    {'name':'YHOO', 'shares': 75},
    {'name':'AOL', 'shares': 20},
    {'name':'SCOX', 'shares': 65}
]
min_shares = min(s['shares'] for s in portfolio)
```

èőÍèőž

äÿŁéÍćçŽĎçď'žä¿ŃåŘŚä¡ăæijŤçď'žäžĘå¡ŞçŤ§æĹŘåŹÍèąÍè¿¿åijŔä¡IJäÿžäÿĂäÿłå■ŢçŃňåŔĆæŢřäijăéĂŠç
ærŤåęĆïijŇäÿŃéÍćè£ŹäžŻèŕ■åŘěæŸŕç■Ľ'æŢĹçŽĎïijŽ

```
s = sum((x * x for x in nums)) #␣
↪æŸ¿çd'žçŽĎäijăéĂŠäÿĂäÿłçŤ$æĹŘåŹÍèąÍè¿¿åijŔårźèśą
s = sum(x * x for x in nums) #␣
↪æŻt'åŁääijŸéŻĚçŽĎåőđçŐřæŰźåijŔïijŇçIJĄçŢěäžĘąæŃňåŔů
```

ä¡£çŤÍäÿĂäÿłçŤ§æĹŘåŹÍèąÍè¿¿åijŔä¡IJäÿžåŔĆæŢřäijŽærŤåĚĹåĹŻåżžäÿĂäÿläÿť'æŮűåĹŮèąÍæŻť'åŁăér
ærŤåęĆïijŇåęĆædIJä¡ăäÿ■ä¡£çŤÍçŤ§æĹŘåŹÍèąÍè¿¿åijŔçŽĎèŕÏïijŇä¡ăåŔřèČ¡äijŽèĂČèŹŚä¡£çŤÍäÿŃéÍćçŽĎå

```
nums = [1, 2, 3, 4, 5]
s = sum([x * x for x in nums])
```

è£Źçğ■æŰźåijŔåŘŇæăůåŔřäžěè¿¿åĹřæČşèęĄçŽĎæŢĹædIJïijŇä¡ĘæŸŕåőČäijŽåď'ŽäÿĂäÿłæ■ééłď'ïijŇå
åŕźäžŐåřŘådŇåĹŮèąÍåŔřèČ¡æšąäżĂäźĹåĚşçşżïijŇä¡ĘæŸŕåęĆædIJåĚČçť'ăæŢřéĞŔéÍđåÿÿåď'ğçŽĎæŮűåĂŹi
åőČäijŽåĹŻåżžäÿĂäÿłåůĺåď'ğçŽĎäżĚäżÈèćńä¡£çŤÍäÿĂæňąåřśèćńäÿćåijČçŽĎäÿť'æŮűæŢřæ■őçżŞædĐãĂĆè

åIJÍä¡£çŤÍäÿĂäžŻèĄŽéŻĘåĞ¡æŢřærŤåęĆ `min()` åŠŇ `max()`
çŽĎæŮűåĂŹä¡ăåŔřèČ¡æŻť'åŁăåĂ¿åŘŚäžŐä¡£çŤÍçŤ§æĹŘåŹÍçL'ĹæIJňïijŇ
åőČäżňæŐěåŔŮçŽĎäÿĂäÿłkeyåĚşéŤőå■ŮåŔĆæŢřæĹŰèőÿåŕźä¡ăå¿ĹæIJL'åÿőåŁl'ãĂĆ
ærŤåęĆïijŇåIJÍäÿŁéÍćçŽĎèŕĄåĹÿä¿Ńå■Řäÿ■ïijŇä¡ăåŔřèČ¡äijŽèĂČèŹŚäÿŃéÍćçŽĎåőđçŐřçL'ĹæIJňïijŽ

```
# Original: Returns 20
min_shares = min(s['shares'] for s in portfolio)
# Alternative: Returns {'name': 'AOL', 'shares': 20}
min_shares = min(portfolio, key=lambda s: s['shares'])
```

1.20 åŘĹåźűåď'Žäÿłå■ŮåĚÿæĹŰæŸăåřĎ

éŮőécŸ

çŐřåIJÍæIJL'åď'Žäÿłå■ŮåĚÿæĹŰèĂĚæŸăåřĎïijŇä¡ăæČşåřĘåőČäżňäžŐéĂżè¿ŚäÿŁåŘĹåźűäÿžäÿĂäÿłå■
ærŤåęĆæ§ěæL'¿åĂijæĹŰèĂĚæčĂæ§ěæ§ŘäžŹéŤőæŸŕåŘęå■ŸåIJÍãĂĆ

èğčåĘşæŰźæąĹ

åĄĞåęĆä¡ăæIJL'åęĆäÿŃäÿd'äÿłå■ŮåĚÿ:

```
a = {'x': 1, 'z': 3 }
b = {'y': 2, 'z': 4 }
```

çŐřåIJĺåĄĞèő¿ä¡ăå£ĚéążåIJĺäÿd'äÿłå■ŮåĚÿäÿ■æL'ğèąŇæ§ěæL'¿æŞ■ä¡IJ(æŕŤåęĆåĚĹäżŐ a äÿ■æL'¿ïijŇåęĆædIJæL'¿äÿ■åĹřåĘ■åIJĺ b äÿ■æL'¿)ãĂĆ äÿĂäÿłéIdåÿÿçőĂå■ŢçŽĎèğčåĘşæŰźæąĹářśæŸŕä¡£çŤĺ collections æĺąåİŮäÿ■çŽĎ ChainMap çśżãĂĆæŕŤåęĆïijŽ

```
from collections import ChainMap
c = ChainMap(a,b)
print(c['x']) # Outputs 1 (from a)
print(c['y']) # Outputs 2 (from b)
print(c['z']) # Outputs 3 (from a)
```

èőĺèőž

äÿĂäÿł ChainMap æŐěåŔŮåd'Žäÿłå■ŮåĚÿåźűåřĘåőČäžňåIJĺéĂżè¿ŚäÿŁåŔŸäÿžäÿĂäÿłå■ŮåĚÿãĂĆ çĎűåŘŐïijŇè£ŹäžŽå■ŮåĚÿåźűäÿ■æŸŕçIJ§çŽĎåŘĹåźűåIJĺäÿĂètůäžĘïijŇ ChainMap çśżåŘłæŸŕåIJĺåĘĚéČĺåĹŽåžžäžĘäÿĂäÿłåőźçžşè£ŹäžŽå■ŮåĚÿçŽĎåĹŮèąĺ åźűéĞ■æŰřåőŽäźL'äžĘäÿĂäžŽåÿÿèğAçŽĎå■ŮåĚÿæŞ■ä¡IJæÌěéĄ■åŐĘè£ŹäÿłåĹŮèąĺãĂĆåd'ğéČĺåĹĘå■ŮåĚ

```
>>> len(c)
3
>>> list(c.keys())
['x', 'y', 'z']
>>> list(c.values())
[1, 2, 3]
>>>
```

åęĆædIJåĞžçŐřéĞ■åd'■éŤőïijŇéĆčäźĹçňňäÿĂæňąåĞžçŐřçŽĎæŸăářĎåĂijäijŽèćńè£ŤåŻđãĂĆ åŻăæ■d'ïijŇä¿Ńå■ŘçÍŃåžŔäÿ■çŽĎ c['z'] æĂżæŸŕäijŽè£ŤåŻđå■ŮåĚÿ a äÿ■åŕźåžŢçŽĎåĂijïijŇèĂŇäÿ■æŸŕ b äÿ■åŕźåžŢçŽĎåĂijãĂĆ

åŕźäžŐå■ŮåĚÿçŽĎæŻt'æŰřæĹŮåĹăéŹd'æŞ■ä¡IJæĂżæŸŕå¡šåŞ■çŽĎæŸŕåĹŮèąĺäÿ■çňňäÿĂäÿłå■ŮåĚÿã

```
>>> c['z'] = 10
>>> c['w'] = 40
>>> del c['x']
>>> a
{'w': 40, 'z': 10}
>>> del c['y']
Traceback (most recent call last):
...
KeyError: "Key not found in the first mapping: 'y'"
>>>
```

ChainMap åŕźäžŐçijŰçÍŃèŕ■èÍĂäÿ■çŽĎä¡IJçŤÍèŇČåŻťåŔŸéĞŔ(æŕŤåęĆ globals , locals ç■Ľ)æŸŕéÍđåÿÿæIJĽçŤÍçŽĎãĂĆ äžŃåőđäÿŁïijŇæIJĽäÿĂäžŻæŰźæşŢåŔŕäžëä¡£åőČåŔŸå¿ŮçőĂå■

```
>>> values = ChainMap()
>>> values['x'] = 1
>>> # Add a new mapping
>>> values = values.new_child()
>>> values['x'] = 2
>>> # Add a new mapping
>>> values = values.new_child()
>>> values['x'] = 3
>>> values
ChainMap({'x': 3}, {'x': 2}, {'x': 1})
>>> values['x']
3
>>> # Discard last mapping
>>> values = values.parents
>>> values['x']
2
>>> # Discard last mapping
>>> values = values.parents
>>> values['x']
1
>>> values
ChainMap({'x': 1})
>>>
```

ä¡IJäÿž ChainMap çŽĎæŻ£äżčïijŇä¡ăåŔŕèČ¡äijŽèĂČèŹŚä¡£çŤÍ update() æŰźæşŢåřĘäÿd'äÿłå■ŮåĚÿåŘĹåźűãĂĆæŕŤåęĆïijŽ

```
>>> a = {'x': 1, 'z': 3 }
>>> b = {'y': 2, 'z': 4 }
>>> merged = dict(b)
>>> merged.update(a)
>>> merged['x']
1
>>> merged['y']
2
>>> merged['z']
3
>>>
```

è£Źæăůäź§èČ¡èąŇå¿ŮéĂŽïijŇä¡ĘæŸŕåőČéIJĂèęĄä¡ăåĹŻåżžäÿĂäÿłåőŇåĚíäÿ■åŘŇçŽĎå■ŮåĚÿåŕźèśą(åŘŇæŮűïijŇåęĆædIJåŐ§å■ŮåĚÿåĄŽäžĘæŻťæŰřïijŇè£Źçğ■æŤźåŔŸäÿ■äijŽåŔ■åžŤåĹřæŰřçŽĎåŘĹåźűå■Ů

```
>>> a['x'] = 13
>>> merged['x']
1
```

ChainMap ä¡£çŤÍåŐ§æÌěçŽĎå■ŮåĚÿïijŇåőČèĞłåůśäÿ■åĹŻåżžæŰřçŽĎå■ŮåĚÿãĂĆæĽĂäžěåőČåźűäÿ

```
>>> a = {'x': 1, 'z': 3 }
>>> b = {'y': 2, 'z': 4 }
>>> merged = ChainMap(a, b)
>>> merged['x']
1
>>> a['x'] = 42
>>> merged['x'] # Notice change to merged dicts
42
>>>
```

CHAPTER 4

çňňäžŇçńäïijŽå■Ůçňęäÿšå ŠŇæŰĞæIJň

åĞäáźŐæĽ ĂæIJL'æIJL'çŤÍçŽĎçÍŃåžŔéČ¡äijŽæűĽ'åŔŁåĹřæ§ŘäžŻæŰĞæIJňåď ĎçŘĘïijŇäÿ■çőąæŸŕè
è£ŹäÿĂçńäářĘéĞ■çĆźåĚşæşĺæŰĞæIJňçŽĎæŞ■ä¡IJåď ĎçŘĘïijŇæŕŤåęĆæŔŘåŔŰå■ŮçňęäÿšïijŇæŘIJçť ćïijŇ
åď ğéČĺåĹĘçŽĎéŰőéćŸéČ¡èČ¡çőĂå■ŢçŽĎèřČçŤĺå■ŮçňęäÿšçŽĎåĘĚåżžæŰźæşŢåőŇæĹŘãĂĆ
ä¡ĘæŸŕïijŇäÿĂäžŻæŻť äÿžåď ■æİĆçŽĎæŞ■ä¡IJåŘŕèČ¡éIJĂèęĄæ■čåĹŹèąĺè¿¿åijŔæĹŰèĂĚåijžåď ğçŽĎèğčæ
åźűäÿŤåIJĺæŞ■ä¡IJUnicodeæŮűåĂŹçćřåĹřçŽĎäÿĂäžŻæčŸæĽ'ŃçŽĎéŰőéćŸåIJĺè£ŹéĞŇäź§äijŽèćńæŔŘåŔ

Contents:

2.1 ä¡£çŤÍåď Žäÿłç ŢŇåőŽçňęåĹĘåĽ'šå■Ůçňęäÿš

éŮőécŸ

ä¡ăéIJĂèęĄåřĘäÿĂäÿłå■Ůçňęäÿšå ĹĘåĽ'šäÿžåď Žäÿłå■ŮæőţïijŇä¡ĘæŸŕåĹĘéŽŤçňę(è£ŸæIJL'åŚĺåŻť çŽĬ

èğčåĘşæŰźæąĹ

`string` åŕźèśąçŽĎ `split()` æŰźæşŢåŔłéĂĆåžŤäžŐéİđåÿÿçőĂå■ŢçŽĎå■Ůçňęäÿšå ĹĘåĽ'šæČĚå¡ćïij
åőČåźűäÿ■åĚĄèőÿæIJL'åď Žäÿłå ĹĘéŽŤçňęæĹŰèĂĚæŸŕåĹĘéŽŤçňęåŚĺåŻť äÿ■çąőåőŽçŽĎçľžæäijãĂĆ
å¡Şä¡ăéIJĂèęĄæŻť åŁăçĄţæť żçŽĎåĹĞåĽ'šå■ŮçňęäÿšçŽĎæŮűåĂŹïijŇæIJĂåě¡ä¡£çŤÍ `re.split()` æŰźæşŢïijŽ

```
>>> line = 'asdf fjdk; afed, fjek,asdf, foo'
>>> import re
>>> re.split(r'[;,\s]\s*', line)
['asdf', 'fjdk', 'afed', 'fjek', 'asdf', 'foo']
```

èőĺèőž

åĞ¡æŢř `re.split()` æŸŕéİđåÿÿåőđçŤĺçŽĎïijŇåŽääÿžåőČåĚĄèőÿä¡ääÿžåĹĘéŽŤçňęæŇĞåőŽåď'Žäÿł
ærŤåęĆïijŇåIJĺäÿŁéİćçŽĎä¿Ńå■Řäÿ■ïijŇåĹĘéŽŤçňęåŕŕäžěæŸŕéĂŮåŕůïijŇåĹĘåŕůæĹŰèĂĚæŸŕçľžæăjïij
åŕłèęĄè£ŹäÿłæĺąåijŔèćńæĽ'¿åĹřïijŇéĆčäźĹåŇźéĚ■çŽĎåĹĘéŽŤçňęäÿď'è¿źçŽĎåőđä¡ŞéČ¡äijŽèćńå¡ŞæĹŘæ
è£ŤåŻđçżŞædIJäÿžäÿĂäÿłå■ŮæőţåĹŮèąĺïijŇè£Źäÿłèů§ `str.split()`
è£ŤåŻđåĂijçśžådŇæŸŕäÿĂæăůçŽĎãĂĆ

å¡Şä¡ää¡£çŤĺ `re.split()` åĞ¡æŢřæŮűåĂŹïijŇéIJĂèęĄçĽźåĹńæşĺæĎŔçŽĎæŸŕæ■čåĹŹèąĺè¿¿åijŔäÿ
åęĆædIJä¡£çŤĺäžĘæ■ŢèŐůåĹĘçżĎïijŇéĆčäźĹèćńåŇźéĚ■çŽĎæŰĞæIJñäź§ářĘåĞžçŐřåIJĺçżŞædIJåĹŮèąĺäÿ

```
>>> fields = re.split(r'(;|,|\s)\s*', line)
>>> fields
['asdf', ' ', 'fjdk', ';', 'afed', ',', 'fjek', ',', 'asdf', ',',
 ↪'foo']
>>>
```

èŐůåŔŰåĹĘåĽšå■Ůçňęåİĺæ§ŘäžŻæČĚåĘţäÿŃäź§æŸŕæIJĽçŤĺçŽĎãĂĆ
ærŤåęĆïijŇä¡ääŔŕèČ¡æČşä£ĺçŢŹåĹĘåĽšå■ŮçňęäÿšïijŇçŤĺæĺěåIJĺåŘŐéĺćéĞ■æŰřædĎéĂääÿĂäÿłæŰřçŽĎè

```
>>> values = fields[::2]
>>> delimiters = fields[1::2] + ['']
>>> values
['asdf', 'fjdk', 'afed', 'fjek', 'asdf', 'foo']
>>> delimiters
[' ', ';', ',', ',', ',', '']
>>> # Reform the line using the same delimiters
>>> ''.join(v+d for v,d in zip(values, delimiters))
'asdf fjdk;afed,fjek,asdf,foo'
>>>
```

åęĆædIJä¡ääÿ■æČşä£ĺçŢŹåĹĘåĽšå■ŮçňęäÿšåĹřçżŞædIJåĹŮèąĺäÿ■åŐżïijŇä¡Ęäż■çĎűéIJĂèęĄä¡£çŤĺåĺ
çąőä£Ïä¡ăçŽĎåĹĘçżĎæŸŕéİđæ■ŢèŐůåĹĘçżĎïijŇå¡ćåęĆ `(?:...)` ãĂĆærŤåęĆïijŽ

```
>>> re.split(r'(?:,|;|\s)\s*', line)
['asdf', 'fjdk', 'afed', 'fjek', 'asdf', 'foo']
>>>
```

2.2 å■ŮçňęäÿšåijĂåď'æĹŰçżŞář¿åŇźéĚ■

éŮőécŸ

ä¡äéIJĂèęĄéĂŽè£ĞæŇĞåőŽçŽĎæŰĞæIJñæĺąåijŔåŐżæčĂæ§ěå■ŮçňęäÿšçŽĎåijĂåď'æĹŰèĂĚçżŞář¿
Schemeç■L'ç■L'ãĂĆ

èğčåĘşæŰźæąĹ

æčĂæ§ěå■Ůçňęäÿšåij Ååď'ťæĹŰçżŞář¿çŽĎäÿĂäÿłçőĂå■ŢæŰźæşŢæŸŕä¡£çŤĺ `str.startswith()` æĹŰèĂĚæŸŕ `str.endswith()` æŰźæşŢãĂĆæŕŤåęĆïijŽ

```
>>> filename = 'spam.txt'
>>> filename.endswith('.txt')
True
>>> filename.startswith('file:')
False
>>> url = 'http://www.python.org'
>>> url.startswith('http:')
True
>>>
```

åęĆæđIJä¡ăæČşæčĂæ§ěåď'Žçğ■åŇźéĚ■åŕŕèČ¡ïijŇåŕłéIJĂèęĄåřĘæL'ĂæIJL'çŽĎåŇźéĚ■éąźæŤ¿åĚěåĹi çĎűåŘŐäijăçżŹ `startswith()` æĹŰèĂĚ `endswith()` æŰźæşŢïijŽ

```
>>> import os
>>> filenames = os.listdir('.')
>>> filenames
[ 'Makefile', 'foo.c', 'bar.py', 'spam.c', 'spam.h' ]
>>> [name for name in filenames if name.endswith(('.c', '.h')) ]
['foo.c', 'spam.c', 'spam.h'
>>> any(name.endswith('.py') for name in filenames)
True
>>>
```

äÿŃéĺćæŸŕåŕęäÿĂäÿłä¿Ńå■ŘïijŽ

```
from urllib.request import urlopen

def read_data(name):
    if name.startswith(('http:', 'https:', 'ftp:')):
        return urlopen(name).read()
    else:
        with open(name) as f:
            return f.read()
```

åěĞæĂłçŽĎæŸŕïijŃè£ŹäÿłæŰźæşŢäÿ■å£ĚéążèęĄè¿ŞåĚěäÿĂäÿłåĚČçżĎä¡IJäÿžåŔĆæŢřãĂĆ åęĆæđIJä¡ăæĄřåůğæIJL'äÿĂäÿł `list` æĹŰèĂĚ `set` çśžåđŃçŽĎéĂL'æŃl'éąźïijŃ èęĄçąőä£ĺäijăéĂŠåŔĆæŢřåL'■åĚĹèřČçŤĺ `tuple()` åřĘåĚűè¡ńæ■ćäÿžåĚČçżĎçśžåđŃãĂĆæŕŤåęĆïijŽ

```
>>> choices = ['http:', 'ftp:']
>>> url = 'http://www.python.org'
>>> url.startswith(choices)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: startswith first arg must be str or a tuple of str, not␣
 ↪list
>>> url.startswith(tuple(choices))
```

```
True
>>>
```

èőÍèőž

startswith() åŠŇ endswith() æŰźæşŢæŔŘä¿ŻäžĘäÿĂäÿłéÍđåÿÿæŰźä¿£çŽĎæŰźåijŔåŐżåĄŽå
çśżäijijçŽĎæŞ■ä¡IJäź§åŔŕäžěä¡£çŤĺåĹĞçĽĞæÍěåőđçŐřïijŇä¡ĘæŸŕäžčçăĄçIJŃèţůæÍěæšąæIJĽéĆčäźĹäijŸé

```
>>> filename = 'spam.txt'
>>> filename[-4:] == '.txt'
True
>>> url = 'http://www.python.org'
>>> url[:5] == 'http:' or url[:6] == 'https:' or url[:4] == 'ftp:'
True
>>>
```

ä¡ăåŔŕäžěèČ¡è£ŸæČşä¡£çŤĺæ■čåĹŹèąĺè¿¿åijŔåŐżåőđçŐřïijŇærŤåęĆïijŽ

```
>>> import re
>>> url = 'http://www.python.org'
>>> re.match('http:|https:|ftp:', url)
<_sre.SRE_Match object at 0x101253098>
>>>
```

è£Źçğ■æŰźåijŔäź§èąŇå¿ŮéĂŽïijŇä¡ĘæŸŕåŕźäžŐçőĂå■ŢçŽĎåŇźéĚ■åőđåIJĺæŸŕæIJĽçĆźåřŔæİŘåd'ğ

æIJĂåŘŐæŔŘäÿĂäÿŃïijŇå¡ŞåŠŇåĚűäżŰæŞ■ä¡IJæŕŤåęĆæŹőéĂŽæŢřæ■őèĄŽåŘĹçŻÿçżŞåŘĹçŽĎæŮú
startswith() åŠŇ endswith() æŰźæşŢæŸŕå¿Ĺäÿ■éŤŹçŽĎãĂĆ
æŕŤåęĆïijŇäÿŃéİćè£Źäÿłèŕ■åŘèæčĂæ§ěæ§ŘäÿłæŰĞäžűåd'źäÿ■æŸŕåŘęå■ŸåIJĺæŇĞåőŽçŽĎæŰĞäžűçśžåđ

```
if any(name.endswith(('.c', '.h')) for name in listdir(dirname)):
...
```

2.3 çŤÍShelléĂŽéĚ■çňęåŇźéĚ■å■Ůçňęäÿš

éŮőécŸ

ä¡ăæČşä¡£çŤÍ **Unix Shell** äÿ■åÿÿçŤĺçŽĎéĂŽéĚ■çňę(æŕŤåęĆ *.py , Dat[0-9]*.csv
ç■L')åŐżåŇźéĚ■æŰĞæIJňå■Ůçňęäÿš

èğčåĘşæŰźæąĹ

fnmatch æĺąåİŮæŔŘä¿ŻäžĘäÿd'äÿłåĞ¡æŢřâĂŤâĂŤ fnmatch() åŠŇ
fnmatchcase() ïijŇåŔŕäžěçŤĺæİěåőđçŐřè£ŹæăůçŽĎåŇźéĚ■ãĂĆçŤĺæşŢåęĆäÿŃïijŽ

```
>>> from fnmatch import fnmatch, fnmatchcase
>>> fnmatch('foo.txt', '*.txt')
True
>>> fnmatch('foo.txt', '?oo.txt')
True
>>> fnmatch('Dat45.csv', 'Dat[0-9]*')
True
>>> names = ['Dat1.csv', 'Dat2.csv', 'config.ini', 'foo.py']
>>> [name for name in names if fnmatch(name, 'Dat*.csv')]
['Dat1.csv', 'Dat2.csv']
>>>
```

fnmatch() åĞ¡æŢřä¡£çŤĺåžŢåśĆæŞ■ä¡IJçşżçż§çŽĎåď ğåřŕåĘŹæŢŕæĎ§èğĎåĹŹ(äÿ■åŘŇçŽĎçşżçż§

```
>>> # On OS X (Mac)
>>> fnmatch('foo.txt', '*.TXT')
False
>>> # On Windows
>>> fnmatch('foo.txt', '*.TXT')
True
>>>
```

åęĆæđIJä¡ăåŕźè£ŹäÿłåŇžåĹńå¿ĹåIJĺæĎŕïijŇåŕŕäžěä¡£çŤĺ fnmatchcase() æİěäžčæŻ£ãĂĆåőČåőŇåĚĺä¡£çŤĺä¡ăçŽĎæĺąåijŕåď ğåřŕåĘŹåŇźéĚ■ãĂĆæŕŤåęĆïijŽ

```
>>> fnmatchcase('foo.txt', '*.TXT')
False
>>>
```

è£ŹäÿďäÿłåĞ¡æŢřéĂŽåÿÿäijŽèćńå£¡çŢěçŽĎäÿĂäÿłçĽźæĂğæŸŕåIJĺåď ĎçŘĘéİđæŰĞäžűåŘ■çŽĎå■Ůç æŕŤåęĆïijŇåĄĞèő¿ä¡ăæIJĽäÿĂäÿłèąŮéĄŞåIJřåİĂçŽĎåĹŮèąĺæŢřæ■őïijŽ

```
addresses = [
    '5412 N CLARK ST',
    '1060 W ADDISON ST',
    '1039 W GRANVILLE AVE',
    '2122 N CLARK ST',
    '4802 N BROADWAY',
]
```

ä¡ăåŕŕäžěåČŕè£ŹæăůåĘŹåĹŮèąĺæŐĺåŕijïijŽ

```
>>> from fnmatch import fnmatchcase
>>> [addr for addr in addresses if fnmatchcase(addr, '* ST')]
['5412 N CLARK ST', '1060 W ADDISON ST', '2122 N CLARK ST']
>>> [addr for addr in addresses if fnmatchcase(addr, '54[0-9][0-9]␣
↪*CLARK*')]
['5412 N CLARK ST']
>>>
```

èőĺèőž

fnmatch() åĞ¡æŢřåŇźéĚ■èČ¡åŁŻäžŇäžŐçőĂå■ŢçŽĎå■ŮçňęäÿšæŰźæşŢåŠŇåijžåď˙ğçŽĎæ■čåĹŹèą
åęĆæđIJåIJĺæŢřæ■őåď˙ĎçŘĘæŞ■ä¡IJäÿ■åŔłéIJĂèęĄçőĂå■ŢçŽĎéĂŽéĚ■çňęářśèČ¡åőŇæĹŘçŽĎæŮűåĂŹïijŇ

åęĆæđIJä¡ăçŽĎäžčçăĄéIJĂèęĄåĄŽæŰĞäžűåŘ■çŽĎåŇźéĚ■ïijŇæIJĂåě¡ä¡£çŤĺ glob æĺąåİŮãĂĆåŔĆèĂČ5.13åřŔèŁĆãĂĆ

2.4 å■Ůçňęäÿšå Ňźé Ě■åŠŇæŘIJçť ć

éŮőécŸ

ä¡ăæČşåŇźéĚ■æĹŰèĂĚæŘIJçť ćçL´źåőŽæĺąåijŔçŽĎæŰĞæIJň

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşåŇźéĚ■çŽĎæŸŕå■ŮéIćå■ŮçňęäÿšïijŇéĆčäźĹä¡ăéĂŽåÿÿåŔłéIJĂèęĄèřČçŤĺå§žæIJňå■Ů
æŕŤåęĆ str.find() , str.endswith() , str.startswith() æĹŰèĂĚçśžäijijçŽĎæŰźæşŢïijŽ

```
>>> text = 'yeah, but no, but yeah, but no, but yeah'
>>> # Exact match
>>> text == 'yeah'
False
>>> # Match at start or end
>>> text.startswith('yeah')
True
>>> text.endswith('no')
False
>>> # Search for the location of the first occurrence
>>> text.find('no')
10
>>>
```

åŕźäžŐåď˙■æİĆçŽĎåŇźéĚ■éIJĂèęĄä¡£çŤĺæ■čåĹŹèąĺè¿¿åijŔåŠŇ re æĺąåİŮãĂĆ äÿžäžEèğčéĞŁæ■čåĹŹèąĺè¿¿åijŔçŽĎå§žæIJňåŐ§çŘĘïijŇåĄĞèő¿ä¡ăæČşåŇźéĚ■æŢřå■ŮæăijåijŔçŽĎæŮěæ
11/27/2012 ïijŇä¡ăåŔŕäžěè£ŹæăůåĄŽïijŽ

```
>>> text1 = '11/27/2012'
>>> text2 = 'Nov 27, 2012'
>>>
>>> import re
>>> # Simple matching: \d+ means match one or more digits
>>> if re.match(r'\d+/\d+/\d+', text1):
... print('yes')
... else:
```

```
... print('no')
...
yes
>>> if re.match(r'\d+/\d+/\d+', text2):
... print('yes')
... else:
... print('no')
...
no
>>>
```

åęĆæđIJä¡ăæČşä¡£çŤĺåŘŇäÿĂäÿłæĺąåijŔåŐżåĄŽåď´ŽæňąåŇźéĚ■ïijŇä¡ååžŤèŕěåĚĹåřĘæĺąåijŔå■Ůçňęä

```
>>> datepat = re.compile(r'\d+/\d+/\d+')
>>> if datepat.match(text1):
... print('yes')
... else:
... print('no')
...
yes
>>> if datepat.match(text2):
... print('yes')
... else:
... print('no')
...
no
>>>
```

match() æĂżæŸŕäżŐå■ŮçňęäÿšåijĂåğŃåŐżåŇźéĚ■ïijŇåęĆæđIJä¡ăæČşæ§ěæL'¿å■ŮçňęäÿšäżżæĎŔé
ä¡£çŤĺ findall() æŰźæşŢåŐżäżčæŽ£ãĂĆæŕŤåęĆïijŽ

```
>>> text = 'Today is 11/27/2012. PyCon starts 3/13/2013.'
>>> datepat.findall(text)
['11/27/2012', '3/13/2013']
>>>
```

åIJĺåőŽäźL'æ■čåĹŽåijŔçŽĎæŮűåĂŹïijŇéĂŽåÿäijŽåĹl'çŤĺæŃňåŔůåŐżæ■ŢèŐůåĹĘçżĎãĂĆæŕŤåęĆïij

```
>>> datepat = re.compile(r'(\d+)/(\d+)/(\d+)')
>>>
```

æ■ŢèŐůåĹĘçżĎåŔŕäżěä¡£å¿ŮåŘŐéİćçŽĎåď´ĎçŘĘæŻť'åŁäçőĂå■ŢïijŇåŽääÿžåŔŕäżěåĹĘåĹńářĘæŕŔäÿ

```
>>> m = datepat.match('11/27/2012')
>>> m
<_sre.SRE_Match object at 0x1005d2750>
>>> # Extract the contents of each group
>>> m.group(0)
'11/27/2012'
>>> m.group(1)
'11'
```

```
>>> m.group(2)
'27'
>>> m.group(3)
'2012'
>>> m.groups()
('11', '27', '2012')
>>> month, day, year = m.groups()
>>>
>>> # Find all matches (notice splitting into tuples)
>>> text
'Today is 11/27/2012. PyCon starts 3/13/2013.'
>>> datepat.findall(text)
[('11', '27', '2012'), ('3', '13', '2013')]
>>> for month, day, year in datepat.findall(text):
... print('{}-{}-{}'.format(year, month, day))
...
2012-11-27
2013-3-13
>>>
```

findall() æŰźæşŢäijŽæŘIJçť'ćæŰĞæIJňåźűäžěåĹÙèąĺå¡ćåijŕè£ŤåŻđæĽĂæIJL'çŽĎåŇźéĚ■ãĂĆ åęĆæđIJä¡ăæČşäžěè£■äžčæŰźåijŕè£ŤåŻđåŇźéĚ■ïijŇåŕŕäžěä¡£çŤĺ finditer() æŰźæşŢæİěäžčæŻ£ïijŇæŕŤåęĆïijŽ

```
>>> for m in datepat.finditer(text):
... print(m.groups())
...
('11', '27', '2012')
('3', '13', '2013')
>>>
```

èőĺèőž

åĚşäžŐæ■čåĹŹèąĺè¿¿åijŕçŘĘèőžçŽĎæŢŹçĺŃåůšçżŕèűĚåĞžäžĘæIJňäźęçŽĎèŇČåŻťãĂĆ äÿ■è£ĞïijŇè£ŹäÿĂèŁĆéŸŘè£řäžĘä¡£çŤĺreæĺąåİŮè£ŻèąŇåŇźéĚ■åŠŇæŘIJçť'ćæŰĞæIJňçŽĎæIJĂå§žæIJňæ æäÿå£Čæ■ěéłď'åŕśæŸŕåĚĹä¡£çŤĺ re.compile() çijŰèŕŚæ■čåĹŹèąĺè¿¿åijŕå■ŮçňęäÿšïijŇ çĎűåŘŐä¡£çŤĺ match() , findall() æĹŰèĂĚ finditer() ç■L'æŰźæşŢãĂĆ

å¡ŞåĘŹæ■čåĹŹåijŕå■ŮçňęäÿšçŽĎæŮűåĂŹïijŇçŻÿåŕźæŹőéĄ■çŽĎåĄŽæşŢæŸŕä¡£çŤĺåŐ§ågŃå■Ůçňęä r'(\d+)/(\d+)/(\d+)' ãĂĆ è£Źçğ■å■ŮçňęäÿšåřĘäÿ■åŐżèğčæđŘåŕ■æŰIJæİäïijŇè£ŹåIJĺæ■čåĹŹèąĺè åęĆæđIJäÿ■è£ŹæăůåĄŽçŽĎèŕİïijŇä¡ăå£Ěéążä¡£çŤĺäÿd'äÿłåŕ■æŰIJæİäïijŇçśžäijij '(\\d+)/(\\d+)/(\\d+)' ãĂĆ

éIJĂèęĄæşĺæĎŕçŽĎæŸŕ match() æŰźæşŢäžĚäžĚæčĂæ§ěå■ŮçňęäÿšçŽĎåijĂågŃéČĺåĹĘãĂĆåőČçŽ

```
>>> m = datepat.match('11/27/2012abcdef')
>>> m
<_sre.SRE_Match object at 0x1005d27e8>
>>> m.group()
```

```
'11/27/2012'
>>>
```

åęĆæđĲä¡ăæČşçš¿çąőåŇźéĚ■ïijŇçąőä£Īä¡ăçŽĎæ■čåĹŹèąÍè¿¿åijŔäżĕ$çżŞåř¿ïijŇåřśåČŔè£ŹäźLè£Źæă

```
>>> datepat = re.compile(r'(\d+)/(\d+)/(\d+)$')
>>> datepat.match('11/27/2012abcdef')
>>> datepat.match('11/27/2012')
<_sre.SRE_Match object at 0x1005d2750>
>>>
```

æIJĂåŘŐïijŇåęĆæđIJä¡ăäżĚäżĚæŸŕåĄŽäÿĂæňąçőĂå■ŢçŽĎæŰĞæIJňåŇźéĚ■/æŘIJçťćæŞ■ä¡IJçŽĎèŕÍ re æĺąåİÚçžğåĹńçŽĎåĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> re.findall(r'(\d+)/(\d+)/(\d+)', text)
[('11', '27', '2012'), ('3', '13', '2013')]
>>>
```

ä¡ĘæŸŕéIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇåęĆæđIJä¡ăæĽŞçőŮåĄŽåď ğéĞŔçŽĎåŇźéĚ■åŠŇæŘIJçť ćæŞ■ä¡IJ æĺąåİÚçžğåĹńçŽĎåĞ¡æŢřäijŽåřĘæIJĂè£ŠçijŰèŕŚè£ĞçŽĎæĺąåijŔçijŞå■Ÿèţůæİěïij Ňåŏăæ■ď åźűäÿ■äijŽæűĹ ä¡ĘæŸŕåęĆæđIJä¡£çŤĺéćĎçijŰèŕŚæĺąåijŔçŽĎèŕİïijŇä¡ăåřĘäijŽåĞŔåřŚæ§ěæĽ¿åŠŇäÿĂäžŻéćİåď ŰçŽĎåď Ď

2.5 å■ŮçňęäÿšæŘIJçť ćåŠŇæŻ£æ■ć

éŮőéćŸ

ä¡ăæČşåIJĺå■Ůçňęäÿšäÿ■æŘIJçť ćåŠŇåŇźéĚ■æŇĞåőŽçŽĎæŰĞæIJňæĺąåijŔ

èğčåĘşæŰźæąĹ

åŕźäžŐçőĂå■ŢçŽĎå■ŮéİćæĺąåijŔïijŇçŻť æŐěä¡£çŤĺ str.repalce() æŰźæşŢå■şåŔŕïijŇæŕŤåęĆïijŽ

```
>>> text = 'yeah, but no, but yeah, but no, but yeah'
>>> text.replace('yeah', 'yep')
'yep, but no, but yep, but no, but yep'
>>>
```

åŕźäžŐåď■æİĆçŽĎæĺąåijŔïijŇèŕůä¡£çŤĺ re æĺąåİŮäÿ■çŽĎ sub() åĞ¡æŢřãĂĆ äÿžäžĘèŕť æŸŐè£ŹäÿłïijŇåĄĞèő¿ä¡ăæČşåřĘå¡ćåijŔäÿž 11/27/2012 çŽĎæŮěæIJ§å■ŮçňęäÿšæŤźæĹŘ 2012-11-27 ãĂĆçď žä¿ŃåęĆäÿŃïijŽ

```
>>> text = 'Today is 11/27/2012. PyCon starts 3/13/2013.'
>>> import re
>>> re.sub(r'(\d+)/(\d+)/(\d+)', r'\3-\1-\2', text)
```

```
'Today is 2012-11-27. PyCon starts 2013-3-13.'
>>>
```

sub() åĞ¡æŢřäÿ■çŽĎçňňäÿĂäÿłåŔĆæŢřæŸŕèćńåŇźéĚ■çŽĎæĺąåijŔïijŇçňňäžŇäÿłåŔĆæŢřæŸŕæŻ£æ
\3 æŇĞåŘŚåĽ■éÍćæĺąåijŔçŽĎæ■ŢèŐůçżĎåŔůãĂĆ

åęĆæđIJä¡ăæĽŞçőŮçŤĺçŻÿåŘŇçŽĎæĺąåijŔåĄŽåď'Žæňąæ£æ■ćïijŇèĂČèŹŚåĚĹçijŰèŕŚåőČæİěæŔŘå

```
>>> import re
>>> datepat = re.compile(r'(\d+)/(\d+)/(\d+)')
>>> datepat.sub(r'\3-\1-\2', text)
'Today is 2012-11-27. PyCon starts 2013-3-13.'
>>>
```

åŕźäžŐæŻt'åŁååď'■æİĆçŽĎæŻ£æ■ćïijŇåŔŕäžěäijăéĂŠäÿĂäÿłæŻ£æ■ćåŻđèřČåĞ¡æŢřæİěäžčæŻ£ïijŇæŕ

```
>>> from calendar import month_abbr
>>> def change_date(m):
... mon_name = month_abbr[int(m.group(1))]
... return '{} {} {}'.format(m.group(2), mon_name, m.group(3))
...
>>> datepat.sub(change_date, text)
'Today is 27 Nov 2012. PyCon starts 13 Mar 2013.'
>>>
```

äÿĂäÿłæŻ£æ■ćåŻđèřČåĞ¡æŢřçŽĎåŔĆæŢřæŸŕäÿĂäÿł match åŕźèśąïijŇäź§åřśæŸŕ
match() æĹŰèĂĚ find() è£ŤåŻđçŽĎåŕźèśąãĂĆ ä¡£çŤĺ group()
æŰźæşŢæİěæŔŘåŘŰçĽźåőŽçŽĎåŇźéĚ■éČĺåĹĘãĂĆåŻđèřČåĞ¡æŢřæIJĂåŘŐè£ŤåŻđæŻ£æ■ćå■Ůçňęäÿšã

åęĆæđIJéŹd'äžĘæŻ£æ■ćåŘŐçŽĎçżŞæđIJåď'ŰïijŇä¡ăè£ŸæČşç§ěéĄŞæIJĽåď'ŽåřŚæŻ£æ■ćåŔŚçŢ§äžĘï
re.subn() æİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
>>> newtext, n = datepat.subn(r'\3-\1-\2', text)
>>> newtext
'Today is 2012-11-27. PyCon starts 2013-3-13.'
>>> n
2
>>>
```

èőĺèőž

åĚşäžŐæ■čåĹŹèąĺè¿¿åijŔæŘIJçť'ćåŠŇæŻ£æ■ćïijŇäÿŁéİćæijŤçď'žçŽĎ sub()
æŰźæşŢå§žæIJňåůšçżŔæűţçŽŰäžĘæĽĂæIJĽãĂĆ åĚűåőđæIJĂéŽ¿çŽĎéČĺåĹĘåřśæŸŕçijŰåĘŹæ■čåĹŹèąĺè¿¿

2.6 å■Ůçňęäÿšå£¡çŢěåď˘ğåřŕåĘźçŽĎæŘIJçť´ćæŻ£æ■ć

éŮőécŸ

ä¡ăéIJĂèęĄäżěå£¡çŢěåď˘ğåřŕåĘźçŽĎæŰźåijŕæŘIJçť´ćäÿŐæŻ£æ■ćæŰĞæIJňå■Ůçňęäÿš

èğčåĘşæŰźæąĹ

äÿžäžĘåIJĺæŰĞæIJňæŞ■ä¡IJæŮűå£¡çŢěåď˘ğåřŕåĘźïijŇä¡ăéIJĂèęĄåIJĺä¡£çŤĺ re æĺąåİŮçŽĎæŮűåĂźżŹè£ŹäžŻæŞ■ä¡IJæŔŘä¿Ż re.IGNORECASE æăĞå£ŮåŔĆæŢřãĂĆærŤåęĆïijŽ

```
>>> text = 'UPPER PYTHON, lower python, Mixed Python'
>>> re.findall('python', text, flags=re.IGNORECASE)
['PYTHON', 'python', 'Python']
>>> re.sub('python', 'snake', text, flags=re.IGNORECASE)
'UPPER snake, lower snake, Mixed snake'
>>>
```

æIJĂåŘŐçŽĎéĆčäÿłä¿Ńå■ŘæŔ■çď˘žäžĘäÿĂäÿłåřŕçijžéŹůïijŇæŻ£æ■ćå■Ůçňęäÿšåźűäÿ■äijŽèĞłåŁĺèů§äÿžäžĘä£őåď˘■è£ŹäÿłïijŇä¡ăåŕŕèČ¡éIJĂèęĄäÿĂäÿłè¿ĚåŁľåĞ¡æŢřïijŇářšåČŔäÿŃéİćçŽĎè£ŹæăůïijŽ

```
def matchcase(word):
    def replace(m):
        text = m.group()
        if text.isupper():
            return word.upper()
        elif text.islower():
            return word.lower()
        elif text[0].isupper():
            return word.capitalize()
        else:
            return word
    return replace
```

äÿŃéİćæŸŕä¡£çŤĺäÿŁè£ŕåĞ¡æŢřçŽĎæŰźæşŢïijŽ

```
>>> re.sub('python', matchcase('snake'), text, flags=re.IGNORECASE)
'UPPER SNAKE, lower snake, Mixed Snake'
>>>
```

èŕŚèĂĚæşĺïijŽ matchcase('snake') è£ŤåŻđäžĘäÿĂäÿłåŻđèřČåĞ¡æŢř(åŔĆæŢřå£ĚéążæŸŕ match åŕźèśą)ïijŇåĽ■éİćäÿĂèŁĆæŔŘåĹřè£ĞïijŇ sub() åĞ¡æŢřéŹď˘äžĘæŐěåŔŮæŻ£æ■ćå■Ůçňęäÿšåď˘ŰïijŇè£ŸèČ¡æŐěåŔŮäÿĂäÿłåŻđèřČåĞ¡æŢřãĂĆ

èőĺèőž

åŕźäžŐäÿĂèĹňçŽĎå£¡çŢěåď ğåřŔåĘŹçŽĎåŇźéĚ■æŞ■ä¡IJïijŇçőĂå■ŢçŽĎäijàéĂŠäÿĂäÿł
`re.IGNORECASE` æăĞå£ŮåŔĆæŢřåŕšåůšçżŔèűşåď§äžĘãĂĆ
ä¡ĘæŸŕéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijŇè£ŹäÿłåŕźäžŐæ§ŘäžŻéIJĂèęĄåď ğåřŔåĘŹè¡ňæ■ćçŽĎUnicodeåŇźéĚ■å
åŔĆèĂČ2.10åřŔèŁĆäžĘèğčæŻť åď ŽçżĘèŁĆãĂĆ

2.7 æIJĂç§■åŇźéĚ■æÍąåijŔ

éŮőécŸ

ä¡ăæ■čåIJĺèřŢçİĂçŤĺæ■čåĹŹèąĺè¿¿åijŔåŇźéĚ■æ§ŘäÿłæŰĞæIJňæÍąåijŔïijŇä¡ĘæŸŕåőČæL'¿åĹřçŽĎæŸ
èĂŇä¡ăæČşä£őæŤźåőČåŔŸæĹŘæ§ěæL'¿æIJĂç§■çŽĎåŔŕèČ¡åŇźéĚ■ãĂĆ

èğčåĘşæŰźæąĹ

è£ŹäÿłéŮőécŸäÿĂèĹňåĞžçŐřåIJĺéIJĂèęĄåŇźéĚ■äÿĂåŕźåĹĘéŽŤçňęäźŃéŮt'çŽĎæŰĞæIJňçŽĎæŮűåĂ
äÿžäžĘèŕť æŸŐæŸĚæěŽïijŇèĂČèŹŠåęĆäÿŃçŽĎä¿Ňå■ŘïijŽ

```
>>> str_pat = re.compile(r'\"(.*)\"')
>>> text1 = 'Computer says "no."'
>>> str_pat.findall(text1)
['no.']
>>> text2 = 'Computer says "no." Phone says "yes."'
>>> str_pat.findall(text2)
['no." Phone says "yes.']
>>>
```

åIJĺè£ŹäÿłäĄ¿Ňå■ŘäÿïijŇæÍąåijŔ `r'\"(.*)\"'` çŽĎæĎŔåŻ¿æŸŕåŇźéĚ■èćńåŔŇåijŢåŔůåŇĚåŘńçŽĺ
ä¡ĘæŸŕåIJĺæ■čåĹŹèąĺè¿¿åijŔäÿ■*æŞ■ä¡IJçňęæŸŕèť łåľłçŽĎïijŇåŻăæ■d'åŇźéĚ■æŞ■ä¡IJäijŽæ§ěæL'¿æIJĂéŢ
äžŐæŸŕåIJĺçňňäžŇäÿłäĄ¿Ňå■ŘäÿïæŘIJçť ć `text2` çŽĎæŮűåĂŹè£ŤåŻđçżŞæđIJåźűäÿïæŸŕæĹŚäžňæČşèęĄç

äÿžäžĘä£őæ■čè£ŹäÿłéŮőécŸïijŇåŔŕäžěåIJĺæÍąåijŔäÿ■çŽĎ*æŞ■ä¡IJçňęåŘŐéİćåŁääÿŁ?ä£őéěŕçňęïijŇå

```
>>> str_pat = re.compile(r'\"(.*?)\"')
>>> str_pat.findall(text2)
['no.', 'yes.']
>>>
```

è£Źæăůåŕšä¡£å¿ŮåŇźéĚ■åŔŸæĹŘéİđèť łåľłæÍąåijŔïijŇäžŐèĂŇå¿ŮåĹřæIJĂç§■çŽĎåŇźéĚ■ïijŇäź§åŕšа

èőĺèőž

è£ŹäÿĂèŁĆåśŢçd'žäžĘåIJĺåĘŹåŇĚåŘńçĆź(.)å■Ůçňęç ŽĎæ■čåĹŹèąĺè¿¿åijŔçŽĎæŮűåĂŹéĄĞåĹřçŽĎäÿ
åIJĺäÿĂäÿłæÍąåijŔå■Ůçňęäÿšäÿ■ïijŇçĆź(.)åŇźéĚ■éŹd'äžĘæ■ćèąŇåd'ŰçŽĎäżżä¡Ţå■Ůçňęã ĂĆ

çĎűèĂŃïijŇåęĆæđIJä¡ăåřĘçĆź(.)åŔůæŤ¿åIJĺåijĂåğŃäÿŐçżŞæİ§çňę(æŕŤåęĆåijŢåŔů)äźŃéŮť çŽĎæŮűåĂŹi
è£Źæăůé ĂŽåÿÿäijŽåřijèĞť å¿Ĺåd'Žäÿ■éŮť çŽĎèćńåijĂåğŃäÿŐçżŞæİ§çňęåŇĚåŘńçŽĎæŰĞæIJňèćńå£¡çŢěa
éĂŽè£ĞåIJÍ * æĹŰèĂĚ + è£Źæăůç ŽĎæŞ■ä¡IJçňęåŘŐéİćæůżåŁăäÿĂäÿł ?
åŔŕäżěåijžåĹűåŇźéĚ■çőŮæşŢæŤźæĹŘåŕżæL'¿æIJĂç§■çŽĎåŔŕèČ¡åŇźéĚ■ãĂĆ

2.8 åd'ŽèąŇåŇźéĚ■æĺąåijŔ

éŮőécŸ

ä¡ăæ■čåIJĺèŕŢçİĂä¡£çŤĺæ■čåĹŹèąĺè¿¿åijŔåŐżåŇźéĚ■äÿĂåd'ğåİŮçŽĎæŰĞæIJňïijŇèĂŇä¡ăéIJĂèęĄèůİ

èğčåĘşæŰźæąĹ

è£ŹäÿłéŮőécŸå¿ĹåĚÿåđŃçŽĎåĞžçŐřåIJĺå¡Şä¡ăçŤĺçĆź(.)åŐżåŇźéĚ■äżżæĎŔå■ŮçňęçŽĎæŮűåĂŹïijŇä
æŕŤåęĆïijŇåĄĞèő¿ä¡ăæČşèŕŢçİĂåŐżåŇźéĚ■Cèŕ■èĺĂåĹĘåĽšçŽĎæşĺéĞŁïijŽ

```
>>> comment = re.compile(r'/\*(.*?)\*/')
>>> text1 = '/* this is a comment */'
>>> text2 = '''/* this is a
... multiline comment */
... '''
>>>
>>> comment.findall(text1)
[' this is a comment ']
>>> comment.findall(text2)
[]
>>>
```

äÿžäžĘä£őæ■čè£ŹäÿłéŮőécŸïijŇä¡ăåŔŕäżěä£őæŤźæĺąåijŔå■ŮçňęäÿšïijŇåćđåŁăåŕźæ■ćèąŇçŽĎæŤŕæŇ

```
>>> comment = re.compile(r'/\*((?:.|\n)*?)\*/')
>>> comment.findall(text2)
[' this is a\n multiline comment ']
>>>
```

åIJĺè£ŹäÿłæĺąåijŔäÿ■ïijŇ `(?:.|\n)` æŇĞåőŽäžĘäÿĂäÿłéİđæ■ŢèŐůçżĎ
(äź§åřśæŸŕåőČåőŽäźĽäžĘäÿĂäÿłäżĚäżĚçŤĺæİěåĄŽåŇźéĚ■ïijŇèĂŇäÿ■èČ¡éĂŽè£Ğå■ŢçŃňæ■ŢèŐůæĹŰèĂ

èőĺèőž

`re.compile()` åĞ¡æŢŕæŐěåŔŮäÿĂäÿłæăĞå£ŮåŔĆæŢŕåŔń `re.DOTALL`
ïijŇåIJĺè£ŹéĞŇéİđåÿÿæIJĽçŤĺãĂĆ åőČåŔŕäžèèőľæ■čåĹŹèąĺè¿¿åijŔäÿ■çŽĎçĆź(.)åŇźéĚ■åŇĚæŃňæ■ćèąŇ

```
>>> comment = re.compile(r'/\*(.*?)\*/', re.DOTALL)
>>> comment.findall(text2)
[' this is a\n multiline comment ']
```

åŕźäžŐçőĂå■ŢçŽĎæČĚåĘţä¡£çŤÍ `re.DOTALL` æäĞèőŕåŔĆæŢřåůëä¡IJçŽĎå¿Ĺåë¡ïijŇ ä¡ĘæŸŕåęĆæđIJæĺąåijŔéİđåÿÿåď■æİĆæĹŰèĂĚæŸŕäÿžäžĘæđĎéĂää■Ůçňęäÿšäžd'çĽŇèĂŇåŕĘåď Žäÿłæĺąåi è£ŹæŮűåĂŹä¡£çŤĺè£ŹäÿłæäĞèőŕåŔĆæŢřåŕšåŔŕèČ¡åĞžçŐŕäÿĂäžŹéŮőécŸãĂĆ åęĆæđIJèőľ'ä¡ăéĂĽ'æŃľçŽĎèŕÍïijŇæIJĂåë¡è£ŸæŸŕåőŽäźĽ'èĞłåůšçŽĎæ■čåĹŹèąĺè¿¿åijŔæĺąåijŔïijŇè£Źæău

2.9 åřĘUnicodeæŰĞælJňæäĞåĞĘåŇŰ

éŮőécŸ

ä¡ăæ■čåIJĺåď ĎçŘĘUnicodeå■Ůçňęäÿšïij ŇéIJĂèęAçąőä£İæĽ ĂæIJĽå■Ůçňęäÿšå IJĺåžŢåśĆæIJĽçŻÿåŘŇ

èğčåĘşæŰźæąĹ

åIJÍUnicodeäÿ■ïijŇæ§ŘäžŻå■Ůçňęè Č¡åď §çŤĺåď Žäÿłå ŘĹæşŢçŽĎçijŰçăAèąĺçď žãĂĆäÿžäžEèŕť æŸŐïij

```
>>> s1 = 'Spicy Jalape\u00f1o'
>>> s2 = 'Spicy Jalapen\u0303o'
>>> s1
'Spicy JalapeÃńo'
>>> s2
'Spicy JalapeÃńo'
>>> s1 == s2
False
>>> len(s1)
14
>>> len(s2)
15
>>>
```

è£ŹéĞŇçŽĎæŰĞæIJň"Spicy JalapeÃńo"ä¡£çŤĺäžĘäÿď çğ■å¡ćåijŔæİěèąĺçď žãĂĆ çňňäÿĂçğ■ä¡£çŤĺæŢt'ä¡Şå■Ůçňę"Ãś"(U+00F1)ïijŇçňňäžŇçğ■ä¡£çŤĺæŃL'äÿĄå■Ůæŕ■"n"åŘŐéİćèů§äÿĂäÿł

åIJĺéIJĂèęĄæŕŤè¿Ča■ŮçňęäÿšçŽĎçĺŃåžŔäÿ■ä¡£çŤĺå■ŮçňęçŽĎåď Žçğ■èąĺçď žäijŽäžğçŤ§éŮőécŸãĂĆ äÿžäžĘä£őæ■čè£ŹäÿłéŮőécŸïijŇä¡ăåŔŕäžěä¡£çŤĺunicodedataæĺąåİŮåĚĹåŕĘæŰĞæIJňæäĞåĞĘåŇŰïijŽ

```
>>> import unicodedata
>>> t1 = unicodedata.normalize('NFC', s1)
>>> t2 = unicodedata.normalize('NFC', s2)
>>> t1 == t2
True
>>> print(ascii(t1))
'Spicy Jalape\xf1o'
```

```
>>> t3 = unicodedata.normalize('NFD', s1)
>>> t4 = unicodedata.normalize('NFD', s2)
>>> t3 == t4
True
>>> print(ascii(t3))
'Spicy Jalapen\u0303o'
>>>
```

normalize() çňňäÿĂäÿłåŔĆæŢřæŇĞåőŽå■ŮçňęäÿšæăĞåĞĘåŇŰçŽĎæŰźåijŔãĂĆ NFCèąĺçď žå■ŮçňęåžŤèŕěæŸŕæŢť ä¡ŞçżĎæĹŘ(æŕŤåęĆåŔŕèČ¡çŽĎèŕİåřšä¡£çŤĺå■ŢäÿĂçijŰçăĄ)ïijŇèĂŇNF

PythonåŘŇæăůæŤŕæŇĄæĽľåśŢçŽĎæăĞåĞĘåŇŰå¡ćåijŔNFKCåŠŇNFKDïijŇåőČäżňåIJĺåď ĎçŘĘæ§Ř

```
>>> s = '\ufb01' # A single character
>>> s
'ïňĄ'
>>> unicodedata.normalize('NFD', s)
'ïňĄ'
# Notice how the combined letters are broken apart here
>>> unicodedata.normalize('NFKD', s)
'fi'
>>> unicodedata.normalize('NFKC', s)
'fi'
>>>
```

èőĺèőž

æăĞåĞĘåŇŰåŕźäžŐäżżä¡ŢéIJĂèęĄäżěäÿĂèĞť çŽĎæŰźåijŔåď ĎçŘĘUnicodeæŰĞæIJňçŽĎçĺŃåžŔéČ¡ å¡Şåď ĎçŘĘæİěèĞłçŤĺæĹůè¿ŞåĚěçŽĎå■ŮçňęäÿšèĂŇä¡ää¿ĹéŽ¿åŐżæŐğåĹűçijŰçăĄçŽĎæŮűåĂŹåřď åĚűå

åIJĺæÿĚçŘĘåŠŇè£Ğæżď æŰĞæIJňçŽĎæŮűåĂŹå■ŮçňęçŽĎæăĞåĞĘåŇŰäź§æŸŕå¿ĹéĞ■èęĄçŽĎãĂĆ æŕŤåęĆïijŇåĄĞèő¿ä¡ăæČşæÿĚéŹď æŐĽäÿĂäžŻæŰĞæIJňäÿŁéİćçŽĎåŔŸé§§çňęçŽĎæŮűåĂŹ(åŔŕèČ¡æŸŕä

```
>>> t1 = unicodedata.normalize('NFD', s1)
>>> ''.join(c for c in t1 if not unicodedata.combining(c))
'Spicy Jalapeno'
>>>
```

æIJĂåŘŐäÿĂäÿłä¿Ńå■ŘåśŢçď žäžĘ unicodedata æĺąåİŮçŽĎåŔęäÿĂäÿłéĞ■èęĄæŰźéİćïijŇäź§åřšæ combining() åĞ¡æŢřåŔŕäżěæţŃèŕŢäÿĂäÿłå■ŮçňęæŸŕåŘęäÿžåŠŇé§§å■ŮçňęãĂĆ åIJĺè£ŹäÿłæĺąåİŮäÿ■è£ŸæIJĽåĚűäżŰåĞ¡æŢřçŤĺäžŐæ§ěæĽ¿å■ŮçňęçśżåĹńïijŇæţŃèŕŢæŸŕåŘęäÿžæŢřå■ŮåŠ

UnicodeæŸ¿çĎűæŸŕäÿĂäÿłå¿Ĺåď ğçŽĎäÿżéćŸãĂĆåęĆæđIJæČşæŻť æůśåĚěçŽĎäžĘèğčåĚşäžŐæăĞåĞ èŕůçIJŃèĂČ UnicodeåőŸç¡Śäÿ■åĚşäžŐè£ŹéČĺåĹĘçŽĎèŕť æŸŐ Ned BatchelderåIJĺ äžŰçŽĎç¡ŚçńŹ äÿŁåŕźPythonçŽĎUnicodeåď ĎçŘĘéŮőéćŸäź§æIJĽäÿĂäÿłå¿Ĺåě¡çŽĎäżŃçż■ãĂĆ

2.10 åIJÍæ■čåĹŹåijŕäÿ■ä¡£çŤÍUnicode

éůőécŸ

ä¡ăæ■čåIJÍä¡£çŤÍæ■čåĹŹèąÍè¿¿åijŕåd'ĎçŘĘæŰĞæIJñïijŇä¡ĘæŸŕåĚşæşÍçŽĎæŸŕUnicodeå■Ůçňęåd'Ďç

èğčåĘşæŰźæąĹ

éžŸèőd'æČĚåĘţäÿŃ re æÍąåİŮåůšçżŕåŕźäÿĂäžŻUnicodeå■Ůçňęçśżæ IJL'äžĘå§žæIJňçŽĎæŤŕæŇĄãĂĆ æŕŤåęĆïijŇ \\d åůšçżŕåŇźéĚ■äżżæĎŔçŽĎunicodeæŢřå■Ůå■Ůçňęäžĕïijž

```
>>> import re
>>> num = re.compile('\d+')
>>> # ASCII digits
>>> num.match('123')
<_sre.SRE_Match object at 0x1007d9ed0>
>>> # Arabic digits
>>> num.match('\u0661\u0662\u0663')
<_sre.SRE_Match object at 0x101234030>
>>>
```

åęĆædIJä¡ăæČşåIJÍæÍąåijŕäÿ■åŇĚåŘńæŇĞåőŽçŽĎUnicodeå■ŮçňęïijŇä¡ääŕŕäżěä¡£çŤÍUnicodeå■Ůçň \uFFF æĹŰèĂĚ \UFFFFFFF)ãĂĆ æŕŤåęĆïijŇäÿŇéİćæŸŕäÿĂäÿłåŇźéĚ■åĞăäÿłäÿ■åŘŇéŸ£æŃL'äijŕçijŰç

```
>>> arabic = re.compile('[\u0600-\u06ff\u0750-\u077f\u08a0-\u08ff]+
↪')
>>>
```

å¡ŞæL'ğèąŇåŇźéĚ■åŠŇæŘIJçt'ćæŞ■ä¡IJçŽĎæŮűåĂŹïijŇæIJĂåě¡æŸŕåĚĹæăĞåĞĘåŇŰåźűäÿŤæÿĚçŘE ä¡ĘæŸŕåŘŇæăůäź§åžŤèŕěæşĺæĎŔäÿĂäžŻçL'źæőŁæČĚåĘţïijŇærŤåęĆåIJĺå£¡çŢěåd'ğářŔåĘŹåŇźéĚ■åŠŇåd'

```
>>> pat = re.compile('stra\u00dfe', re.IGNORECASE)
>>> s = 'straÃ§e'
>>> pat.match(s) # Matches
<_sre.SRE_Match object at 0x10069d370>
>>> pat.match(s.upper()) # Doesn't match
>>> s.upper() # Case folds
'STRASSE'
>>>
```

èőÍèőž

æůůåŘĹä¡£çŤÍUnicodeåŠŇæ■čåĹŹèąÍè¿¿åijŔéĂŽåÿÿäijŽèőľä¡ăæŁŞçŃĆãĂĆ åęĆædIJä¡ăçIJ§çŽĎæL'ŞçőŮè£ŹæăůåĄŽçŽĎèŕĬïijŇæIJĂåě¡èĂČèŹŚäÿŃåőL'èčĚçňňäÿL'æŰźæ■čåĹŹåijŔåžŞ åőČäžňäijŽäÿžUnicodeçŽĎåd'ğářŔåĘŹè¡ňæ■ćåŠŇåĚűäżŰåd'ğéĞŔæIJL'èűčçL'źæĂğæŔŘä¿ŽåĚĺéİćçŽĎæŤŕ

2.11 åĹĺäéŹďå■Ůçňęäÿšäÿ■äÿ■élJĂèęĄçŽĎå■Ůçňę

éŮőécŸ

ä¡ăæČşåŐżæŐĽ'æŰĞæIJňå■Ůçňęäÿšåij Ååď't'ïijŇçżŞář¿æĹŰèĂĚäÿ■éŮt'äÿ■æČşèęĄçŽĎå■Ůçňęïij Ňæŕ

èğčåĘşæŰźæąĹ

strip() æŰźæşŢèČ¡çŤĺäžŐåĹăéŹď'åijĂåğŃæĹŰçżŞář¿çŽĎå■Ůçňęã ĂĆ lstrip() åŠŇ rstrip() åĹĘåĹńäżŐåůęåŠŇäżŐåŔşæL'ğèąŇåĹăéŹď'æŞ■ä¡IJãĂĆ éżŸèőď'æČĚåĘţäÿŃïijŇè£ŹäžŻæŰźæşŢäijŽåŐżéŹď'çl'žçŹ¡å■Ůçňęïij Ňä¡Ęæ Ÿŕä¡ääź§åŔŕäżěæŇĞåőŽåĚűäż Ú

```
>>> # Whitespace stripping
>>> s = ' hello world \n'
>>> s.strip()
'hello world'
>>> s.lstrip()
'hello world \n'
>>> s.rstrip()
' hello world'
>>>
>>> # Character stripping
>>> t = '-----hello====='
>>> t.lstrip('-')
'hello====='
>>> t.strip('-=')
'hello'
>>>
```

èőÍèőž

è£ŹäžŻ strip() æŰźæşŢåIJÍèŕżåŔŰåŠŇæÿĚçŘĘæŢřæ■őäżěåď'ĞåŘŐçż■åď'ĎçŘĘçŽĎæŮűåĂŹæŸŕç æŕŤåęĆïijŇä¡ääŔŕäżěçŤÍåőČäžňæİěåŐżæŐĽ'çl'žæăijïijŇåijŢåŘůåŠŇåőŇæĹŘåĚűäżŰäżżåŁąãĂĆ

ä¡ĘæŸŕéIJĂèęĄæşĺæĎŘçŽĎæŸŕåŐżéŹď'æŞ■ä¡IJäÿ■äijŽåŕźå■ŮçňęäÿšçŽĎäÿ■éŮt'çŽĎæŰĞæIJňäžğçŤ

```
>>> s = ' hello      world \n'
>>> s = s.strip()
>>> s
'hello     world'
>>>
```

åęĆæđIJä¡ăæČşåď'ĎçŘĘäÿ■éŮt'çŽĎçl'žæăijïijŇéĆčäźĹä¡ăéIJĂèęĄæśĆåŁl'åĚűäżŰæŁĂæIJŕãĂĆæŕŤåę replace() æŰźæşŢæĹŰèĂĚæŸŕçŤÍæ■čåĹŹèąÍè¿¿åijŔæŽ£æ■ćãĂĆçd'žä¿ŇåęĆäÿŃïijŽ

```
>>> s.replace(' ', '')
'helloworld'
>>> import re
>>> re.sub('\s+', ' ', s)
'hello world'
>>>
```

éĂŽåÿÿæČĚåĘţäýŃä¡ăæČşåřĘå■Ůçňęäÿš strip æŞ■ä¡IJåŠŇåĚűäžŰè£■äžčæŞ■ä¡IJçŻÿçżŞåŘĹïijŇæŕ
åęĆæđIJæŸŕè£ŹæăůçŽĎèŕÍïijŇéĆčäźĹçŤ§æĹŘåŹĺèąĺè¿¿åijŔåřśåŔŕäžěåď ğæŸ¿èžńæĽŃäžĘãĂĆæŕŤåęĆïijŽ

```
with open(filename) as f:
    lines = (line.strip() for line in f)
    for line in lines:
        print(line)
```

åIJĺè£ŹéĞŇïijŇèąĺè¿¿åijŔ lines = (line.strip() for line in f)
æĽğèąŇæŢřæ■őè¡ňæ■ćæŞ■ä¡IJãĂĆ è£ŹçğæŰźåijŔéÍđåÿÿéńŸæŢĹïijŇåŻääÿžåőČäÿ■éIJĂèęĄécĎåĚĹèŕžåĚ
åőČäžĚäžĚåŘłæŸŕåĹŻåžžäÿĂäÿłçŤ§æĹŘåŹĺïijŇåźűäÿŤæŕŘæňąè£ŤåŽđèąŇäźŃåĽ■äijŽåĚĹæĽğèąŇstripæŞ

åŕźäžŐæŻť éńŸéŸŰçŽĎstripïijŇä¡ääŔŕèČ¡éIJĂèęĄä¡£çŤÍ translate()
æŰźæşŢãĂĆèŕůåŔĆéŸĚäÿŃäÿĂèŁĆäžĘèğčæŻť åď ŽåĚşäžŐå■ŮçňęäÿšæÿĚçŘĘçŽĎåĘĚåőźãĂĆ

2.12 åőąæ§ěæÿĚçŘĘæŰĞælJňå■Ůçňęäÿš

éŮőécŸ

äÿĂäžŻæŮăèĄŁçŽĎåźijçÍŽéżŠåőćåIJĺä¡ăçŽĎç¡ŚçńŹéąţéÍćèąĺå■Ţäÿ■è¿ŞåĚěæŰĞælJň”pÃ¡tÄěÃűÃś”ïi

èğčåĘşæŰźæąĹ

æŰĞælJňæÿĚçŘĘéŮőécŸäijŽæűL’åŔŁåĹråŇĚæŃňæŰĞælJňèğčæđŘäÿŐæŢřæ■őåď ĎçŘĘç■L’äÿĂçşżå
åIJĺéÍđåÿÿçőĂå■ŢçŽĎæČĚå¡ćäÿŃïijŇä¡ääŔŕèČ¡äijŽéĂL’æŃl’ä¡£çŤĺå■ŮçňęäÿšåĞ¡æŢř(æŕŤåęĆ
str.upper() åŠŇ str.lower())åřĘæŰĞælJňè¡ňäÿžæăĞåĞĘæäijåijŔãĂĆ ä¡£çŤÍ
str.replace() æĹŰèĂĚ re.sub() çŽĎçőĂå■ŢæŻ£æ■ćæŞ■ä¡IJèČ¡åĹäéŹď æĹŰèĂĚæŤźåŔŸæŇĞåőŹ
ä¡ääŘŇæăůè£Ÿåŕźäžěä¡£çŤÍ2.9åřŔèŁĆçŽĎ unicodedata.normalize()
åĞ¡æŢřåřĘunicodeæŰĞælJňæăĞåĞĘåŇŰãĂĆ

çĎűåŘŐïijŇæIJL’æŮűåĂŹä¡ääŔŕèČ¡è£ŸæČşåIJĺæÿĚçŘĘæŞ■ä¡IJäÿŁæŻť è£ŻäÿĂæ■ěãĂĆæŕŤåęĆïijŇä¡
äÿžäžĘè£ŹæăůåĄŽïijŇä¡ääŔŕäžěä¡£çŤÍçżŔåÿÿäijŽècńå£¡èğĘçŽĎ str.translate()
æŰźæşŢãĂĆ äÿžäžĘæijŤçď žïijŇåĄĞèő¿ä¡ăçŐřåIJĺæIJL’äÿŃéÍćè£ŹäyłåĞŇäźśçŽĎå■Ůçňęäÿšïijž

```
>>> s = 'pÃ¡tÄěÃűÃś\fis\tawesome\r\n'
>>> s
'pÃ¡tÄěÃűÃś\x0cis\tawesome\r\n'
>>>
```

çňňäÿĂæ■ěæŸŕæÿĚçŘĘçľžçŹ¡å■Ůçňęã ĂĆäÿžäžĘè£ŹæăůåĄŽïijŇåĚĹåĹŻåžžäÿĂäÿłåřŔçŽĎè¡ňæ■ćèąĺ translate() æŰźæşŢïijŽ

```
>>> remap = {
...     ord('\t') : ' ',
...     ord('\f') : ' ',
...     ord('\r') : None # Deleted
... }
>>> a = s.translate(remap)
>>> a
'pÃ¡tÄěÃűÃś is awesome\n'
>>>
```

æ■čåęĆä¡ăçIJŃçŽĎéĆčæăůïijŇçľžçŹ¡å■Ůçňę \t åŠŇ \f åůšçżŔèćńéĞ■æŰřæŸäářĎåĹřäÿĂäÿłçľžæăijãĂĆåŻđè¡ęå■ŮçňęrçŻt'æŐěèćńåĹäéŹd'ãĂĆ

ä¡ăåŔŕäžěäžěè£Źäÿłèąĺæăijäÿžå§žçąĂè£ŻäÿĂæ■ěædĎåžžæŻt'åd'ğçŽĎèąĺæăijãĂĆæŕŤåęĆïijŇèőľ'æĹŚä

```
>>> import unicodedata
>>> import sys
>>> cmb_chrs = dict.fromkeys(c for c in range(sys.maxunicode)
...                          if unicodedata.combining(chr(c)))
...
>>> b = unicodedata.normalize('NFD', a)
>>> b
'pÃ¡tÄěÃűÃś is awesome\n'
>>> b.translate(cmb_chrs)
'python is awesome\n'
>>>
```

äÿŁéİćä¿Ńå■Řäÿ■ïijŇéĂŽè£Ğä¡£çŤÍ dict.fromkeys() æŰźæşŢædĎéĂääÿĂäÿłå■ŮåĚÿïijŇæŕŘäÿłUnicodeåŠŇé§şçňęä¡IJäÿžéŤőïijŇåŕźäžŐçŽĎåĂijåĚĺéČĺäÿž None ãĂĆ

çĎűåŘŐä¡£çŤÍ unicodedata.normalize() åřĘåŐ§ågŃè¿ŞåĚěæăĞåĞĘåŇŰäÿžåĹĘèğčå¡ćåijŔå■ çĎűåŘŐåĘ■èřČçŤÍ translate åĞ¡æŢřåĹăéŹd'æĽ'ĂæIJL'éĞ■é§şçňęãĂĆ åŘŇæăůçŽĎæŁĂæIJŕäź§åŘŕäžěèćńçŤÍæİěåĹăéŹd'åĚűäžŰçśžåđŃçŽĎå■Ůçňę(æŕŤåęĆæŐğåĹűå■Ůçňęç■L')ã

ä¡IJäÿžåŔęäÿĂääÿłä¿Ńå■ŘïijŇè£ŹéĞŇæđĎéĂääÿĂäÿłåřĘæĽ'ĂæIJL'UnicodeæŢřå■Ůå■ŮçňęæŸäářĎåĹi

```
>>> digitmap = { c: ord('0') + unicodedata.digit(chr(c))
...           for c in range(sys.maxunicode)
...           if unicodedata.category(chr(c)) == 'Nd' }
...
>>> len(digitmap)
460
>>> # Arabic digits
>>> x = '\u0661\u0662\u0663'
>>> x.translate(digitmap)
'123'
>>>
```

åŕęäÿĂçğ■æÿĚçŘĘæŰĞæIJňçŽĎæŁĂæIJŕæűĽ'åŔŁåĹřI/Oèğčçă Ąäÿ Őçij Ůçă Ąåğ¡æŢřăĂĆè£ŹéĞŃçŽĎ
çĎűåŘŐåĘ■çżŞåŘĹ `encode()` æĹŰèĂĚ `decode()` æŞ■ä¡IJæĬěæÿĚéŹď'æĹŰä£őæŤźåőČãĂĆæŕŤåęĆïijŽ

```
>>> a
'pÃ¡tÄëÃűÃś is awesome\n'
>>> b = unicodedata.normalize('NFD', a)
>>> b.encode('ascii', 'ignore').decode('ascii')
'python is awesome\n'
>>>
```

è£ŹéĞŃçŽĎæăĞåĞĘåŃŰæŞ■ä¡IJåřĘåŐ§æĬěçŽĎæŰĞæIJňåĹĘèğčäÿžå■ŢçŃňçŽĎåŠŃé§şçňęãĂĆæŐě
å¡ŞçĎűïijŃè£Źçğ■æŰźæşŢäżĚäżĚåŔłåIJĺæIJĂåŘŐçŽĎçŻőæăĞåřśæŸŕèŐůåŔŰåĹřæŰĞæIJňåŕźåžŤACSIIèą

èőĺèőž

æŰĞæIJňå■Ůçňęæÿ ĚçŘĘäÿĂäÿłæIJĂäÿżèęĄçŽĎéŮőécŸåžŤèŕěæŸŕè£ŘèąŃçŽĎæĂğèČ¡ãĂĆäÿĂèĹňæ
åŕźäžŐçőĂå■ŢçŽĎæŻ£æ■ćæŞ■ä¡IJïijŃ `str.replace()` æŰźæşŢéĂŽåÿÿæŸŕæIJĂå£ńçŽĎïijŃçŤŽèĞşåIJ
æŕŤåęĆïijŃäÿžäžĘæÿĚçŘĘçľžçŹ¡å■Ůçňęïij Ńä¡ăåŔŕäżěè£ŹæăůåĄŽïijŽ

```
def clean_spaces(s):
    s = s.replace('\r', '')
    s = s.replace('\t', ' ')
    s = s.replace('\f', ' ')
    return s
```

åęĆæđIJä¡ăåŐżæţŃèŕŢçŽĎèŕĺïijŃä¡åářśäijŽåŔŚçŐřè£Źçğ■æŰźåijŔäijŽæŕŤä¡£çŤĺ
`translate()` æĹŰèĂĚæ■čåĹŹèąĺè¿¿åijŔèęĄå£ńå¿Ĺåď'ŽãĂĆ

åŕęäÿĂæŰźéĺćïijŃåęĆæđIJä¡ăéIJĂèęĄæL'ğèąŃäżżä¡Ţåď'■æĺĆå■Ůçňęåŕźå■Ůçňę çŽĎéĞ■æŰřæŸăářĎæ
`tanslate()` æŰźæşŢäijŽéİđåÿÿçŽĎå£ńãĂĆ

äżŐåď'ğçŽĎæŰźéĺćæİěèőšïijŃåŕźäžŐä¡ăçŽĎåžŤçŤĺçĺŃåžŔæİěèŕť'æĂğèČ¡æŸŕä¡ääÿ■å¿Ůäÿ■åŐżèĞłåů
äÿ■åźÿçŽĎæŸŕïijŃæĹŚäžňäÿ■åŔŕèČ¡çżŹä¡ăåžžèőőäÿĂäÿłçL'źåőŽçŽĎæŁĂæIJŕïijŃä¡£åőČèČ¡åď'§éĂĆåžŤa
åŽăæ■ď'åőđéŹĚæČĚåĘţäÿ■éIJĂèęĄä¡äèĞłåůšåŐżåŕİèŕŢäÿ■åŘŃçŽĎæŰźæşŢåźűèŕĎäijřåőČãĂĆ

år¡çőąè£ŹäÿĂèŁĆéŻĘäÿ■èőĺèőžçŽĎæŸŕæŰĞæIJňïijŃä¡ĘæŸŕçśžäijijçŽĎæŁĂæIJŕäź§åŔŕäžěéĂĆçŤĺäž

2.13 å■Ůçňęäÿšåŕźé¡Ř

éŮőécŸ

ä¡ăæČşéĂŽè£Ğæ§Řçğ■åŕźé¡ŘæŰźåijŔæİěæäijåijŔåŃŰå■Ůçňęäÿš

èğčåĘşæŰźæąĹ

åŕźäžŐå§žæIJńçŽĎå■Ůçňęäÿšåŕźé¡ŘæŞ■ä¡IJïijŇåŔŕäžěä¡£çŤĺå■Ůçňęäÿšç ŽĎ `ljust()` , `rjust()` åŠŇ `center()` æŰźæşŢãĂĆæŕŤåęĆïijŽ

```
>>> text = 'Hello World'
>>> text.ljust(20)
'Hello World          '
>>> text.rjust(20)
'          Hello World'
>>> text.center(20)
'    Hello World     '
>>>
```

æĽĂæIJL'è£ŹäžŻæŰźæşŢéČ¡èČ¡æŐěåŘŮäÿĂäÿłåŔŕéĂĽçŽĎåąńåĚĚå■Ůçňęã ĂĆæŕŤåęĆïijŽ

```
>>> text.rjust(20,'=')
'==========Hello World'
>>> text.center(20,'*')
'****Hello World*****'
>>>
```

åĞ¡æŢř `format()` åŘŇæăůåŔŕäžěçŤĺæİěå¿Ĺåőźæ ŸŞçŽĎåŕźé¡Řå■Ůçňęäÿšã ĂĆ ä¡ãèęĄåĄŽçŽĎåřśæŸŕä¡£çŤĺ <, > æĹŰèĂĚ ^ å■Ůçňęå ŘŐéİćçť ğèů§äÿĂäÿłæŇĞåőŽçŽĎåő¡åžęã ĂĆæŕŤåęĆ

```
>>> format(text, '>20')
'          Hello World'
>>> format(text, '<20')
'Hello World          '
>>> format(text, '^20')
'    Hello World     '
>>>
```

åęĆæđIJä¡ăæČşæŇĞåőŽäÿĂäÿłéİđçľžæăijçŽĎåąńåĚĚå■Ůçňęïij ŇårĘåőČåĘŹåĹřåŕźé¡Řå■Ůçňęç ŽĎåĽ'■

```
>>> format(text, '=>20s')
'==========Hello World'
>>> format(text, '*^20s')
'****Hello World*****'
>>>
```

å¡Şæăijåij ŔåŇŰåd'ŽäÿłåĂijçŽĎæŮůåĂŹïijŇè£ŹäžŻæăijåijŔäžčçăĄäź§åŕźäžèèćńçŤĺåIJÍ `format()` æŰźæşŢäÿ■ãĂĆæŕŤåęĆïijŽ

```
>>> '{:>10s} {:>10s}'.format('Hello', 'World')
'     Hello      World'
>>>
```

`format()` åĞ¡æŢřçŽĎäÿĂäÿłåě¡åd'ĎæŸŕåőČäÿ■äžĚéĂĆçŤĺäžŐå■Ůçňęäÿšã ĂĆåőČåŔŕäžěçŤĺæİěæăij æŕŤåęĆïijŇä¡ãåŔŕäžěçŤĺåőČæİěæăijåijŔåŇŰæŢřå■ŮïijŽ

```
>>> x = 1.2345
>>> format(x, '>10')
'    1.2345'
>>> format(x, '^10.2f')
'   1.23   '
>>>
```

èőÍèőž

åIJÍèĂĄçŽĎäžčçăĄäÿ■ïijŇä¡ăçżŘåÿÿäijŽçIJŃåĹřèćńçŤÍæÌěæäijåijŔåŇŰæŰĞæIJňçŽĎ % æŞ■ä¡IJçňęãĂĆærŤåęĆïijŽ

```
>>> '%-20s' % text
'Hello World         '
>>> '%20s' % text
'         Hello World'
>>>
```

ä¡ĘæŸfïijŇåIJÍæŰřçĽĹæIJňäžčçăĄäÿ■ïijŇä¡ăåžŤèřěäijŸåĚĹéĂĽæŃľ `format()` åĞ¡æŢřæĹŰèĂĚæŰźæşŢãĂĆ `format()` èęĄærŤ % æŞ■ä¡IJçňęçŽĎåŁ§èČ¡æŻť'äÿžåijžåď'ğãĂĆ åźűäÿŤ `format()` äź§ærŤä¡£çŤÍ `ljust()` , `rjust()` æĹŰ `center()` æŰźæşŢæŻť'éĂŽçŤÍïijŇ åŻääÿžåőČåŘfäžěçŤÍæÌěæäijåijŔåŇŰäžżæĎŘåŕźèśąïijŇèĂŇäÿ■äžĚäžĚæŸfå■Ůçňęäÿšã ĂĆ

åęĆæđIJæČşèęĄåőŇåĚÍäžĘèğč `format()` åĞ¡æŢřçŽĎæIJL'çŤÍçL'źæĂğïijŇ èřůåŘĆèĂČ åIJÍçž£PythonæŰĞæąč

2.14 åŘĹåźűæŃijæŐěå■Ůçňęäÿš

éŮőécŸ

ä¡ăæČşåřĘåĞääÿłåřŘçŽĎå■Ůçňęäÿšå ŘĹåźűäÿžäÿĂäÿłåď'ğçŽĎå■Ůçňęäÿš

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşèęĄåŘĹåźűçŽĎå■Ůçňęäÿšæ ŸfåIJÍäÿĂäÿłåžŘåĹŮæĹŰèĂĚ `iterable` äÿ■ïijŇéĆčäźĹæIJĂå£nçŽĎæŰźåijŔářśæŸfä¡£çŤÍ `join()` æŰźæşŢãĂĆærŤåęĆïijŽ

```
>>> parts = ['Is', 'Chicago', 'Not', 'Chicago?']
>>> ' '.join(parts)
'Is Chicago Not Chicago?'
>>> ','.join(parts)
'Is,Chicago,Not,Chicago?'
>>> ''.join(parts)
```

```
'IsChicagoNotChicago?'
>>>
```

åĹİçĪŃèţůæİëĳŇè£Źçğ■èŕ■æşŢçĪŃäÿŁåŐżäijŽæŕŤè¿ČæĂłĳŇä¡ĘæŸŕ join() èćńæŇĞåőŽäÿžå■Ůçňęäÿşçİ... è£Źæäůå ĄŽçŽĎéČĺåĹĘåŐ§åŽäæŸŕä¡ăæČşåŐżè£đæŐěçŽĎåŕźèśąåŔŕèČ¡æİëèĞłåŔĎçğ■äÿ■åŘŇçŽĎæŢřæ... åęĆæđĲåĲĺæĽĂæĲĽè£ŹäžŻåŕźèśąäÿŁéČ¡åőŽäźĽäÿĂäÿł join() æŰźæşŢæŸŐæŸ¿æŸŕåĘŮä¡ŹçŽĎãĂĆ åŽäæ■đ'ä¡äåŔłéĲĂèęĄæŇĞåőŽä¡ăæČşèęĄçŽĎåĹĘåĽšå■Ůçňęäÿšå... join() æŰźæşŢåŐżåřĘæŰĞæĲňçĽĞæőţçžĎåŔĹèţůæİěãĂĆ

åęĆæđĲä¡ăäžĚäžĚåŔłæŸŕåŔĹåźűåřŠæŢřåĞäÿłå■Ůçňęäÿšïij Ňä¡£çŤĺåŁăåŔů(+)éĂŽåÿÿåůšçżŔèůşåď§...

```
>>> a = 'Is Chicago'
>>> b = 'Not Chicago?'
>>> a + ' ' + b
'Is Chicago Not Chicago?'
>>>
```

åŁăåŔů(+)æŞ■ä¡ĲçňęåĲĺä¡ĲäÿžäÿĂäžŻåď■æİĆå■Ůçňęäÿšæäijåij ŔåŇŰçŽĎæŽ£äžčæŰźæąĹçŽĎæŮű...

```
>>> print('{} {}'.format(a,b))
Is Chicago Not Chicago?
>>> print(a + ' ' + b)
Is Chicago Not Chicago?
>>>
```

åęĆæđĲä¡ăæČşåĲĺæžŘçăĄäÿ■åřĘäÿď'äÿłå■ŮéİĆå■Ůçňęäÿšå ŔĹåźűèţůæİěïijŇä¡ăåŔłéĲĂèęĄçőĂå■ŢçŽ...

```
>>> a = 'Hello' 'World'
>>> a
'HelloWorld'
>>>
```

èőĺèőž

å■Ůçňęäÿšå ŘĹåźűå ŔŕèČ¡çĲŃäÿŁåŐżåźűäÿ■éĲĂèęĄçŤĺäÿĂæŢ'èŁĆæİěèőĺèőžãĂĆ ä¡ĘæŸŕäÿ■åžŤèŕěåřŔçĲŃè£ŹäÿłéŮőéćŸïijŇçĺŃåžŔåŚŸéĂŽåÿÿåĲĺå■Ůçňęäÿšæäijåij ŔåŇŰçŽĎæŮűåĂŹåŽ...

æĲĂéĞ■èęĄçŽĎéĲĂèęĄåijŢèţůæşĺæĎŔçŽĎæŸŕïijŇä¡ŞæĽŠäžŇä¡£çŤĺåŁăåŔů(+)æŞ■ä¡ĲçňęåŐżè£đæ... åŽääÿžåŁăåŔůè£đæŐěäijŽåijŢèţůåĘĚå■Ÿåď'■åĹűäžěåŔŁåđČåĲ¿åŽđæŤűæŞ■ä¡ĲãĂĆ çĽźåĹńçŽĎïijŇä¡ăæřÿè£ĲéČ¡äÿ■åžŤåČŔäÿŃéİćè£Źæäůå ĘŹå■Ůçňęäÿšè£đæŐěäżčçăĄïijŽ

```
s = ''
for p in parts:
    s += p
```

è£Źçğ■åĘŹæşŢäijŽæŕŤä¡£çŤĺ join() æŰźæşŢè£ŘèąŇçŽĎèęĄæĚćäÿĂäžŻïijŇåŽääÿžæŕŔäÿĂæňąæL... ä¡ăæĲĂåě¡æŸŕåĚĹæŤűéŻĘæĽĂæĲĽçŽĎå■ŮçňęäÿšçĽĞæőţçĎűåŘŐåĘ■åřĘåőČäžňè£đæŐěèţůæİěãĂĆ

äÿĂäÿłçŻÿåŕźæŕŤè¿ČèĄłæŸŐçŽĎæŁĂåůğæŸŕåĹľçŤĺçŤ§æĹŘåŹĺèąĺè¿¿åijŔ(åŔČèĂČ1.19åřŔèŁĆ)è¡ň...

```
>>> data = ['ACME', 50, 91.1]
>>> ','.join(str(d) for d in data)
'ACME,50,91.1'
>>>
```

åŘŇæăůè£Ÿå¿ŮæşÍæĎŕäÿ■å£ÈèęĄçŽĎå■Ůçňęäÿšè£đæŐěæŞ■ä¡IJãĂĆæIJL'æŮűåĂŹçÍŃåžŕåŚŸåIJÍæ

```
print(a + ':' + b + ':' + c) # Ugly
print(':'.join([a, b, c])) # Still ugly
print(a, b, c, sep=':') # Better
```

å¡ŞæůůåŘĹä¡£çŤÍI/OæŞ■ä¡IJåŠŇå■Ůçňęäÿšè£đæŐěæŞ■ä¡IJçŽĎæŮűåĂŹïijŇæIJL'æŮűåĂŹéIJĂèęĄäż
ærŤåęĆïijŇèĂČèŹŚäÿŃéÍćçŽĎäÿd'çńŕäžčçăĄçL'Ğæőţïijž

```
# Version 1 (string concatenation)
f.write(chunk1 + chunk2)

# Version 2 (separate I/O operations)
f.write(chunk1)
f.write(chunk2)
```

åęĆæđIJäÿd'äÿłå■Ůçňęäÿšå¿ĹåřŔïijŇéĆčäźĹçňňäÿĂäÿłçL'ĹæIJňæĂğèČ¡äijŽæŻt'åě¡äžŻïijŇåŻääÿžI/Oç
åŔęåd'ŰäÿĂæŰźéÍćïijŇåęĆæđIJäÿd'äÿłå■Ůçňęäÿšå¿Ĺåd'ğïijŇéĆčäźĹçňňäžŇäÿłçL'ĹæIJňåŔŕèČ¡äijŽæŻt'åŁ
åŽääÿžåőČéĄ£åĚ■äžĘåĹŻåżžäÿĂäÿłå¿Ĺåd'ğçŽĎäÿt'æŮűçżŞæđIJåźűäÿŤèęĄåd'■åĹűåd'ğéĞŔçŽĎåĘĚå■Ÿå
è£ŸæŸŕéĆčåŔèèŕÍïijŇæIJL'æŮűåĂŹæŸŕéIJĂèęĄæăźæ■ő䡿äçŽĎåžŤçŤÍçÍŃåžŔçL'źçĆźæÍěåĘşåőŽåžŤèŕěä¡£

æIJĂåŘŐèřĹäÿĂäÿŃïijŇåęĆæđIJä¡ääĞĘåd'ĞçijŰåĘŹæđĎåżžåd'ğéĞŔåřŔå■ŮçňęäÿšçŽĎè¿ŞåĞžäžčçăA
ä¡ăæIJĂåě¡èĂČèŹŚäÿŃä¡£çŤÍçŤ§æĹŘåŹĺåĞ¡æŢřïijŇåĹl'çŤÍyieldèŕ■åŔěäžğçŤ§è¿ŞåĞžçL'Ğæőţã ĆæŕŤåęĆ

```
def sample():
    yield 'Is'
    yield 'Chicago'
    yield 'Not'
    yield 'Chicago?'
```

è£ŹçğæŰźæşŢäÿĂäÿłæIJL'èűčçŽĎæŰźéÍćæŸŕåőČåźűæšąæIJL'åŕźè¿ŞåĞžçL'ĞæőţåĺřåžŢèęĄæĂŐæăů
ä¿ŃåęĆïijŇä¡ääŔŕäžěçőĂå■ŢçŽĎä¡£çŤÍ `join()` æŰźæşŢårĘè£ŹäžŻçL'Ğæőţå ŘĹåźűètůæÍěïijž

```
text = ''.join(sample())
```

æĹŰèĂĚä¡ääź§åŔŕäžěårĘå■Ůçňęäÿšçl'Ğæőţé Ğ■åőŽåŘŠåĹřI/Oïijž

```
for part in sample():
    f.write(part)
```

åĘ■æĹŰèĂĚä¡ăè£ŸåŔŕäžěåĘŹåĞžäÿĂäžŻçżŞåŘĹI/OæŞ■ä¡IJçŽĎæůůåŘĹæŰźæąĹïijž

```
def combine(source, maxsize):
    parts = []
    size = 0
    for part in source:
```

```
        parts.append(part)
        size += len(part)
        if size > maxsize:
            yield ''.join(parts)
            parts = []
            size = 0
        yield ''.join(parts)

# çżŞåŘĹæŰĞäżűæŞ■ä¡IJ
with open('filename', 'w') as f:
    for part in combine(sample(), 32768):
        f.write(part)
```

è£ŹéĞŇçŽĎåĚşéŤőçĆźåIJĺäžŐåŐ§ågŃçŽĎçŤ§æĹŘåŹĺåĞ¡æŢŕåźűäÿ■éIJÄèęAç§ěéĄŞä¡£çŤĺçżĘèŁĆïij

2.15 å■Ůçňęäÿšäÿ■æŔŠåĚěåŔŸéĞŔ

éŮőécŸ

ä¡ăæČşåĹŻåżžäÿĂäÿłåĘĚåţŇåŔŸéĞŔçŽĎå■ŮçňęäÿšïijŇåŔŸéĞŔèćńåőČçŽĎåĂijæĽĂèąĺçd'žçŽĎå■Ů

èğčåĘşæŰźæąĹ

PythonåźűæšąæIJL'åŕźåIJĺå■Ůçňęäÿšäÿ■çőÅå■ŢæŽ£æ■ćåŔŸéĞŔåĂijæŔŘä¿ŻçŻť'æŐěçŽĎæŤŕæŇĄãĂ
ä¡ĘæŸŕéĂŽè£Ğä¡£çŤĺå■Ůçňęäÿšç ŽĎ `format()` æŰźæşŢæİèèğčåĘşè£ŹäÿłéŮőécŸãĂĆæŕŤåęĆïijŽ

```
>>> s = '{name} has {n} messages.'
>>> s.format(name='Guido', n=37)
'Guido has 37 messages.'
>>>
```

æĹŰèĂĚïijŇåęĆæđIJèęĄèćńæŽ£æ■ćçŽĎåŔŸéĞŔèČ¡åIJĺåŔŸéĞŔå§§äÿ■æL'¿åĹřïijŇ
éĆčäźĹä¡ääŔŕäžěçżŞåŘĹä¡£çŤĺ `format_map()` åŠŇ `vars()`
ãĂĆåřśåČŔäÿŇéİćè£ŹæăůïijŽ

```
>>> name = 'Guido'
>>> n = 37
>>> s.format_map(vars())
'Guido has 37 messages.'
>>>
```

`vars()` è£ŸæIJL'äÿĂäÿłæIJL'æĎŔæĂİçŽĎçL'źæĂğåřśæŸŕåőČäź§éĂĆçŤĺäžŐåŕźèśąääðđä¿ŇãĂĆæŕŤå

```
>>> class Info:
...     def __init__(self, name, n):
...         self.name = name
```

```
...         self.n = n
...
>>> a = Info('Guido',37)
>>> s.format_map(vars(a))
'Guido has 37 messages.'
>>>
```

format åŠŇ format_map() çŽĎäÿĂäÿłçijžéŹůåřśæŸŕåőČäżňåźűäÿ■èČ¡å¿Ĺåě¡çŽĎåď ĎçŘĘåŔŸéĞ

```
>>> s.format(name='Guido')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'n'
>>>
```

äÿĂçğ■éĄ£åĚ■è£Źçğ■éŤŹèŕŕçŽĎæŰźæşŢæŸŕåŔęåď ŰåőŽäźĽäÿĂäÿłåŘńæIJĽ __missing__() æŰźæşŢçŽĎå■ŮåĚÿåŕźèśąïijŇåřśåČŔäÿŃéİćè£ŹæăůïijŽ

```
class safesub(dict):
"""éŸšæ■ćkeyæL'¿äÿ■åĹř"""
def __missing__(self, key):
    return '{' + key + '}'
```

çŐřåIJĺä¡ăåŔŕäżěåĽľçŤĺè£ŹäÿłçśżåŇĚèčĚè¿ŞåĚěåŘŐäijăéĂŠçżŹ format_map() ïijŽ

```
>>> del n # Make sure n is undefined
>>> s.format_map(safesub(vars()))
'Guido has {n} messages.'
>>>
```

åęĆædIJä¡ăåŔŚçŐřèĞłåůśåIJĺäżčçăĄäÿ■éćŚçźĄçŽĎæL'ğèąŇè£ŹäžŻæ■ěéłd'ïijŇä¡ăåŔŕäżěåřĘåŔŸéĞŔa

```
import sys

def sub(text):
    return text.format_map(safesub(sys._getframe(1).f_locals))
```

çŐřåIJĺä¡ăåŔŕäżěåČŔäÿŃéİćè£ŹæăůåĘŹäžĘïijŽ

```
>>> name = 'Guido'
>>> n = 37
>>> print(sub('Hello {name}'))
Hello Guido
>>> print(sub('You have {n} messages.'))
You have 37 messages.
>>> print(sub('Your favorite color is {color}'))
Your favorite color is {color}
>>>
```

èőĺèőž

åď Žåźť äžěæİěçŤśäžŐPythonçijžäźŔåŕźåŔŸéĞŔæŽ£æ■ćçŽĎåĘĚç¡őæŤŕæŇĄèĂŇåŕijèĞť äžĘåŘĎçğ■äÿ ä¡IJäÿžæIJňèŁĆäÿ■åśŢçď žçŽĎäÿĂäÿłåŔŕèČ¡çŽĎèğčåĘşæŰźæąĹïijŇä¡ääŔŕäžěæIJL'æŮűåĂŹäijŽçIJŃåĹřåČ

```
>>> name = 'Guido'
>>> n = 37
>>> '%(name) has %(n) messages.' % vars()
'Guido has 37 messages.'
>>>
```

ä¡ääŔŕèČ¡è£ŸäijŽçIJŃåĹřå■Ůçňęäÿšæĺąæİ£çŽĎä¡£çŤĺïijŽ

```
>>> import string
>>> s = string.Template('$name has $n messages.')
>>> s.substitute(vars())
'Guido has 37 messages.'
>>>
```

çĎűèĂŇïijŇ `format()` åŠŇ `format_map()` çŻÿæŕŤè¿ČäÿŁéİćè£ŹäžŻæŰźæąĹèĂŇåůšæŻť åŁääĚĹ ä¡£çŤĺ `format()` æŰźæşŢè£ŸæIJL'äÿĂäÿłåě¡åď ĎåřśæŸŕä¡ääŔŕäžěèŐůå¿Ůåŕźå■ŮçňęäÿšæäijåijŔåŇŰçŽĎ èĂŇè£ŹäžŻçL'źæĂgæŸŕä¡£çŤĺåČŔæĺąæİ£å■ŮçňęäÿšäźŇçśżçŽĎæŰźæąĹäÿ■åŔŕèČ¡èŐůå¿ŮçŽĎãĂĆ

æIJňæIJžè£ŸéČĺåĹĘäžŃçż■äžĘäÿĂäžŻéńŸçžğçL'źæĂgãĂĆæŸåřĎæĹŰèĂĚå■ŮåĚÿçśżäÿ■éšIJäÿžäžž `__missing__()` æŰźæşŢåŔŕäžěèőľ ä¡ääőŽäźL'åęĆä¡Ţåď ĎçŘĘçijžåď śçŽĎåĂijãĂĆ åIJÍ `SafeSub` çśżäÿ■ïijŇè£ŹäÿłæŰźæşŢèćńåőŽäźL'äÿžåŕźçijžåď śçŽĎåĂijè£ŤåŻđäÿĂäÿłå■ää¡■çňęãĂĆ ä¡ääŔŕäžěåŘŚçŐřçijžåď śçŽĎåĂijäijŽåĞžçŐřåIJĺçżŞæđIJå■Ůçňęäÿšäÿ■(åIJĺèřČèŕŢçŽĎæŮűåĂŹåŔŕèČ¡å¿Ĺa `KeyError` åijĆåÿÿãĂĆ

`sub()` åĞ¡æŢŕä¡£çŤĺ `sys._getframe(1)` è£ŤåŻđèřČçŤĺèĂĚçŽĎæăĹåÿğãĂĆåŔŕäžěäžŐäÿ■èő£éŮ `f_locals` æĺěèŐůå¿ŮåśĂéČĺåŘŸéĞŔãĂĆ æŕńæŮăçŰŚéŮőçżIåď ğéČĺåĹĘæČĚåĘţäÿŇåIJĺäžčçăĄäÿ■åŐżç ä¡ĘæŸŕïijŇåŕźäžŐåČŔå■ŮçňęäÿšæŻ£æ■ćåůěåĚůåĞ¡æŢřèĂŇèĺĂåőČæŸŕéIđåÿÿæIJL'çŤĺçŽĎãĂĆ åŘęåď ŰïijŇåĂijå¿ŮæşĺæĎŘçŽĎæŸŕ `f_locals` æŸŕäÿĂäÿłåď ■åĹűěřČçŤĺåĞ¡æŢřçŽĎæIJňåIJřåŘŸéĞŔçŽ åř¡çőąä¡ääŔŕäžěæŤźåŘŸ `f_locals` çŽĎåĘĚåőźïijŇä¡ĘæŸŕè£ŹäÿłäŁőæŤźåŕźäžŐåŘŐéİćçŽĎåŘŸéĞŔèő£é æĽĂäžěïijŇèŹ¡èŕť'èő£éŮőäÿĂäÿłæăĹåÿğçIJŃäÿŁåŐżå¿ĹéĆłæĄűïijŇä¡ĘæŸŕåŕźåőČçŽĎäžżä¡ŢæŞ■ä¡IJäÿ■ä

2.16 äžěæŇĞåőŽåĹŮåő¡æăijåijŔåŇŰå■Ůçňęäÿš

éŮőécŸ

ä¡ăæIJL'äÿĂäžŻéŢŁå■ŮçňęäÿšïijŇæČşäžěæŇĞåőŽçŽĎåĹŮåő¡ářĘåőČäžňéĞ■æŰřæăijåijŔåŇŰãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤĺ `textwrap` æĺąåİŮæİěæăijåijŔåŇŰå■ŮçňęäÿšçŽĎè¿ŞåĞžãĂĆæŕŤåęĆïijŇåĄĞåęĆä¡ăæIJL'äÿŇå

```
s = "Look into my eyes, look into my eyes, the eyes, the eyes, \
the eyes, not around the eyes, don't look around the eyes, \
look into my eyes, you're under."
```

äÿŃéÌćæijŤçd'žä¡£çŤÍ `textwrap` æăijåijŕåŇŰå■Ůçňęäÿšç ŽĎåd'Žçğ■æŰźåijŕïijŽ

```
>>> import textwrap
>>> print(textwrap.fill(s, 70))
Look into my eyes, look into my eyes, the eyes, the eyes, the eyes,
not around the eyes, don't look around the eyes, look into my eyes,
you're under.

>>> print(textwrap.fill(s, 40))
Look into my eyes, look into my eyes,
the eyes, the eyes, the eyes, not around
the eyes, don't look around the eyes,
look into my eyes, you're under.

>>> print(textwrap.fill(s, 40, initial_indent='    '))
    Look into my eyes, look into my
eyes, the eyes, the eyes, the eyes, not
around the eyes, don't look around the
eyes, look into my eyes, you're under.

>>> print(textwrap.fill(s, 40, subsequent_indent='    '))
Look into my eyes, look into my eyes,
    the eyes, the eyes, the eyes, not
    around the eyes, don't look around
    the eyes, look into my eyes, you're
    under.
```

èőÍèőž

`textwrap` æÍąåİŮåŕźäžŐå■Ůçňęäÿšæ L'Şå■řæŸŕéÍđåÿÿæIJL'çŤÍçŽĎïijŇçL'źåĹńæŸŕå¡Şä¡ăåÿŇæIJŻè¿Ş
ä¡ăåŔŕäžěä¡£çŤÍ `os.get_terminal_size()` æŰźæşŢæÌèèŐůåŔŰçżĹçńŕçŽĎåd'ğářŔářžåŕÿãĂĆæŕŤåęĆ

```
>>> import os
>>> os.get_terminal_size().columns
80
>>>
```

`fill()` æŰźæşŢæŐěåŔŮäÿĂäžŽåĚűäžŰåŔŕéĂL'åŔĆæŢřæÌěæŐğåĹűtabïijŇèŕ■åŔěçżŞår¿ç■L'ãĂĆ åŔĆéŸĚ textwrap.TextWrapperæŰĞæąč èŐůåŔŰæŻť'åd'ŽåĘĚåőźãĂĆ

2.17 åIJĺå■Ůçňęäÿšäÿ■åď˙ĎçŘĘhtmlåŠŇxml

éŮőécŸ

ä¡ăæČşåřĘHTMLæĹŰèĂĚXMLåőđä¡ŞåęĆ `&entity;` æĹŰ `&#code;` æŽ£æ■ćäÿžåŕźåžŤçŽĎæŰĞæIJñãĂĆ åĘ■èĂĚïijŇä¡ăéIJĂèęĄè¡ňæ■ćæŰĞæIJñäÿ■çĽźåőŽçŽĎå■Ůçňę(æŕŤåę >, æĹŰ &)ãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşæŽ£æ■ćæŰĞæIJňå■Ůçňęäÿšäÿ■çŽĎ ‘<’ æĹŰèĂĚ ‘>’ ïijŇä¡£çŤĺ `html.escape()` åĞ¡æŢřåŔŕäžěå¿ĹåőźæŸŞçŽĎåőŇæĹŘãĂĆæŕŤåęĆïijŽ

```
>>> s = 'Elements are written as "<tag>text</tag>".'
>>> import html
>>> print(s)
Elements are written as "<tag>text</tag>".
>>> print(html.escape(s))
Elements are written as &quot;&lt;tag&gt;text&lt;/tag&gt;&quot;.

>>> # Disable escaping of quotes
>>> print(html.escape(s, quote=False))
Elements are written as "&lt;tag&gt;text&lt;/tag&gt;".
>>>
```

åęĆæđIJä¡ăæ■čåIJĺåď˙ĎçŘĘçŽĎæŸŕASCIIæŰĞæIJñïijŇåźűäÿŤæČşåřĘéİđASCIIæŰĞæIJňåŕźåžŤçŽĎç åŔŕäžěçżŹæ§ŘäžŻI/OåĞ¡æŢřäijäéĂŠåŔĆæŢř `errors='xmlcharrefreplace'` æİěè¿¿åĹřè£ŹäÿłçŻőãĂĆæŕŤåęĆïijŽ

```
>>> s = 'Spicy JalapeÃ±o'
>>> s.encode('ascii', errors='xmlcharrefreplace')
b'Spicy Jalape&#241;o'
>>>
```

äÿžäžĘæŽ£æ■ćæŰĞæIJñäÿ■çŽĎçijŰçăĄåőđä¡ŞïijŇä¡ăéIJĂèęĄä¡£çŤĺåŘęåď˙ŰäÿĂçğ■æŰźæşŢãĂĆ åęĆæđIJä¡ăæ■čåIJĺåď˙ĎçŘĘHTMLæĹŰèĂĚXMLæŰĞæIJñïijŇèŕŢçİĂåĚĹä¡£çŤĺäÿĂäÿłåŘĹéĂĆçŽĎHTML éĂŽåÿÿæČĚåĘţäÿŃïijŇè£ŹäžŽåůěåĚůäijŽèĞłåŁĺæŽ£æ■ćè£ŹäžŽçijŰçăĄåĂijïijŇä¡ăæŮăéIJĂæŃĚå£ČãĂĆ

æIJL’æŮűåĂŹïijŇåęĆæđIJä¡ăæŐěæŤůåĹřäžĘäÿĂäžŽåŘńæIJL’çijŰçăĄåĂijçŽĎåŐ§ågŃæŰĞæIJñïijŇéI éĂŽåÿÿä¡ăåŘłéIJĂèęĄä¡£çŤĺHTMLæĹŰèĂĚXMLèğčæđŘåŹĺçŽĎäÿĂäžŽçŻÿåĚşåůěåĚůåĞ¡æŢř/æŰźæşŢå■

```
>>> s = 'Spicy &quot;Jalape&#241;o&quot.'
>>> from html.parser import HTMLParser
>>> p = HTMLParser()
>>> p.unescape(s)
'Spicy "JalapeÃ±o".'
>>>
>>> t = 'The prompt is &gt;&gt;&gt;'
```

```
>>> from xml.sax.saxutils import unescape
>>> unescape(t)
'The prompt is >>>'
>>>
```

èőÍèőž

åIJÍçŤ§æĹŘHTMLæĹŰèĂĚXMLæŰĞæIJňçŽĎæŮűåĂŹïijŇåęĆæđIJæ■čçąőçŽĎè¡ňæ■ćçL'źæőŁæăĞèć
çL'źåĹńæŸŕå¡Şä¡ăä¡£çŤÍ `print()` åĞ¡æŢřæĹŰèĂĚåĚűäžŰå■ŮçňęäÿšæăijåijŔåŇŰæİěäžğçŤ§è¿ŞåĞžçŽĎa
ä¡£çŤÍåČŔ `html.escape()` çŽĎåůěåĚůåĞ¡æŢřåŔŕäžěå¿ĹåőźæŸŞçŽĎèğčåĘşè£ŹçśźéŮőécŸãĂĆ

åęĆæđIJä¡ăæČşäžěåĚűäžŰæŰźåijŔåď ĎçŘĘæŰĞæIJňïijŇè£ŸæIJL'äÿĂäžŻåĚűäžŰçŽĎåůěåĚůåĞ¡æŢřa
`xml.sax.saxutils.unescape()` åŔŕäžěåÿőåŁľä¡ăãĂĆ
çĎűèĂŇïijŇä¡ăåžŤèŕěåĚĹèřČçăŤæÿĚæěŽæĂŐæăůä¡£çŤÍäÿĂäÿłåŘĹéĂĆçŽĎèğčæđŘåŹÍãĂĆ
æŕŤåęĆïijŇåęĆæđIJä¡ăåIJÍåď ĎçŘĘHTMLæĹŰXMLæŰĞæIJňïijŇ
ä¡£çŤÍæ§ŘäÿłèğčæđŘæÍąåİŮæŕŤåęĆ `html.parse` æĹŰ `xml.etree.ElementTree`
åůšçżŔåÿőä¡äèĞłåŁÍåď ĎçŘĘäžĘçŻÿåĚşçŽĎæŻ£æ■ćçżĘèŁĆãĂĆ

2.18 å■Ůçňęäÿšäžď'çL'ŇèğčæđŘ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłå■ŮçňęäÿšïijŇæČşäžŐåůęèĞşåŔşåřĘåĚűèğčæđŘäÿžäÿĂäÿłäžď'çL'ŇætĄãĂĆ

èğčåĘşæŰźæąĹ

åĄĞåęĆä¡ăæIJL'äÿŃéİćè£ŹæăůäÿĂäÿłæŰĞæIJňå■ŮçňęäÿšïijŽ

```
text = 'foo = 23 + 42 * 10'
```

äÿžäžĘäžď'çL'ŇåŇŰå■ŮçňęäÿšïijŇä¡ăäÿ■äžĚéIJÄèęĄåŇźéĚ■æĺąåijŔïijŇè£Ÿå¿ŮæŇĞåőŽæĺąåijŔçŽĎç
æŕŤåęĆïijŇä¡ăåŘŕèČ¡æČşåřĘå■Ůçňęäÿšå ČŔäÿŃéİćè£ŹæăůèĄňæ■ćäÿžäžŔåĹŮåŕźïijŽ

```
tokens = [('NAME', 'foo'), ('EQ','='), ('NUM', '23'), ('PLUS','+'),
    ('NUM', '42'), ('TIMES', '*'), ('NUM', 10')]
```

äÿžäžĘæL'ğèąŇè£ŹæăůçŽĎåĹĞåĹĘïijŇçňňäÿĂæ■ěářśæŸŕåČŔäÿŃéİćè£ŹæăůåĹl'çŤÍåŚ¡åŘ■æ■ŢèŐůçž

```
import re
NAME = r'(?P<NAME>[a-zA-Z_][a-zA-Z_0-9]*)'
NUM = r'(?P<NUM>\d+)'
PLUS = r'(?P<PLUS>\+)'
TIMES = r'(?P<TIMES>\*)'
EQ = r'(?P<EQ>=)'
```

```
WS = r'(?P<WS>\s+)'

master_pat = re.compile('|'.join([NAME, NUM, PLUS, TIMES, EQ, WS]))
```

åIJĺäÿŁéİćçŽĎæĺąåijŕäÿ■ïijŇ ?P<TOKENNAME> çŤĺäžŐçżŹäÿĂäÿłæĺąåijŕåŚ¡åŘ■ïijŇä¿ŻåŘŐéİćä¡£çŤ

äÿŃäÿĂæ■ěïijŇäÿžäžĘäžd'çĽŇåŇŰïijŇä¡£çŤĺæĺąåijŕåŕźèśąå¿ĹåřŚècńäžžç§ěéĄŞçŽĎ scanner() æŰźæşŢãĂĆ è£ŹäÿłæŰźæşŢäijŽåĹŻåžžäÿĂäÿł scanner åŕźèśąïijŇ åIJĺè£ŹäÿłåŕźèśąäÿŁäÿ■æŰ■çŽĎèřČçŤĺ match() æŰźæşŢäijŽäÿĂæ■ěæ■ěçŽĎæĽńæŔŘçŽőæăĞæŰĞæIJñïijŇærŕæ■ěäÿĂäÿłåŇźéĚ■ãĂĆ äÿŃéİćæŸŕæijŤçd'žäÿĂäÿł scanner åŕźèśąåęĆä¡Ţåůëä¡IJçŽĎäžd'äžŠåijŕä¿Ńå■ŘïijŽ

```
>>> scanner = master_pat.scanner('foo = 42')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('NAME', 'foo')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('WS', ' ')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('EQ', '=')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('WS', ' ')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('NUM', '42')
>>> scanner.match()
>>>
```

åőđéŹĚä¡£çŤĺè£Źçğ■æŁĂæIJŕçŽĎæŮűåĂŹïijŇåŔŕäžěå¿ĹåőźæŸ§çŽĎåČŔäÿŃéİćè£Źæăůår̨äÿŁè£řäž

```
def generate_tokens(pat, text):
Token = namedtuple('Token', ['type', 'value'])
scanner = pat.scanner(text)
for m in iter(scanner.match, None):
    yield Token(m.lastgroup, m.group())

# Example use
for tok in generate_tokens(master_pat, 'foo = 42'):
    print(tok)
# Produces output
# Token(type='NAME', value='foo')
# Token(type='WS', value=' ')
# Token(type='EQ', value='=')
```

```
# Token(type='WS', value=' ')
# Token(type='NUM', value='42')
```

åęĆæđIJä¡ăæČşè£ĞæżďäżďçĽŇæţĄïijŇä¡ăåŔŕäžěåőŽäźĽæŻťåď˙ŽçŽĎçŢ§æĹŘåŹĺåĞ¡æŢřæĹŰèĂĚä¡
æŕŤåęĆïijŇäÿŃéİćæijŢçď˙žæĂŐæăůè£ĞæżďæĽ˙ĂæIJĽçŽĎçľžçŹ¡äżďçĽŇïijŽ

```
tokens = (tok for tok in generate_tokens(master_pat, text)
            if tok.type != 'WS')
for tok in tokens:
    print(tok)
```

èőĺèőž

éĂŽåÿÿæİěèőšäżďçĽŇåŇŰæŸŕå¿Ĺåď˙ŽéńŸçžğæŰĞæIJňèğčæđŘäÿŐåď˙ĎçŘEçŽĎçňňäÿĂæ■ěãĂĆ
äÿžäžĘä¡£çŤĺäÿŁéİćçŽĎæĽ˙ńæŔŘæŰźæşŢïijŇä¡ăéIJĂèęĄèőřä¡Ŕè£ŹéĞŇäÿĂäžŻéĞ■èęĄçŽĎåĞăçĆźãĂĆ
çňňäÿĂçĆźåřśæŸŕä¡ăå£Ěéążçąőèőď˙ä¡ää¡£çŤĺæ■čåĹŹèąĺè¿¿åijŔæŇĞåőŽäžĘæĽ˙ĂæIJĽè¿ŞåĚěäÿ■åŔŕèČ¡åĞ
åęĆæđIJæIJĽäżżä¡Ţäÿ■åŔŕåŇźéĚ■çŽĎæŰĞæIJňåĞžçŐřäžĘïijŇæĽ˙ńæŔŘåřśäijŽçŻť˙æŐěåĄIJæ■ćãĆĆè£Źä

äżď˙çĽŇçŽĎéąžåžŔäź§æŸŕæIJĽå¡śåŞ■çŽĎãĂĆ `re` æĺąåİŮäijŽæŇĽ˙çĚğæŇĞåőŽåě¡çŽĎéąžåžŔåŐżåĄ
åŻäæ■ď˙ïijŇåęĆæđIJäÿĂäÿłæĺąåijŔæĄřåě¡æŸŕåŔęäÿĂäÿłæŻť˙éŢ£æĺąåijŔçŽĎå■Řå■ŮçňęäÿšïijŇéĆčäźĹä¡ăé

```
LT = r'(?P<LT><)'
LE = r'(?P<LE><=)'
EQ = r'(?P<EQ>=)'

master_pat = re.compile('|'.join([LE, LT, EQ])) # Correct
# master_pat = re.compile('|'.join([LT, LE, EQ])) # Incorrect
```

çňňäžŇäÿłæĺąåijŔæŸŕéŤŹçŽĎïijŇåŻääÿžåőČäijŽářĘæŰĞæIJň<=åŇźéĚ■äÿžäżď˙çĽŇLTçť˙ğèů§çİĂEQi

æIJĂåŘŐïijŇä¡ăéIJĂèęĄçŢŹæĎŔäÿŃå■Řå■Ůçňęäÿšå¡ćåijŔçŽĎæĺąåijŔãĂĆæŕŤåęĆïijŇåĄĞèő¿ä¡ăæIJI

```
PRINT = r'(?P<PRINT>print)'
NAME = r'(?P<NAME>[a-zA-Z_][a-zA-Z_0-9]*)'

master_pat = re.compile('|'.join([PRINT, NAME]))

for tok in generate_tokens(master_pat, 'printer'):
    print(tok)

# Outputs :
# Token(type='PRINT', value='print')
# Token(type='NAME', value='er')
```

åĚşäžŐæŻť˙énŸéŸűçŽĎäżď˙çĽŇåŇŰæŁĂæIJřïijŇä¡ăåŔŕèČ¡éIJĂèęĄæ§ěçIJŃ PyParsing æĹŰèĂĚ PLY åŇĚãĂĆ äÿĂäÿłèřČçŤÍPLYçŽĎä¿Ńå■ŘåIJĺäÿŃäÿĂèŁĆäijŽæIJĽæijŢçď˙žãĂĆ

2.19 åőđçŐřäÿĂäÿłçőĂå■ŢçŽĎéĂŠå¡ŠäÿŃéŹ■åĹĘæđŘåŹĺ

éŮőécŸ

ä¡ăæČşæăźæ■őäÿĂçżĎèŕ■æşŢèğĎåĹŹèğčæđŘæŰĞæIJňåźűæL'ğèąŇåŚ¡äžd'ïijŇæĹŰèĂĚæđĎéĂääÿĂä
åęĆæđIJèŕ■æşŢéİđåÿÿçőĂå■ŢïijŇä¡ăåŔŕäžěèĞłåůśåĘŹè£ŹäÿłèğčæđŘåŹĺïijŇèĂŇäÿ■æŸŕä¡£çŤĺäÿĂäžŻæąĘ

èğčåĘşæŰźæąĹ

åIJĺè£ŹäÿłéŮőécŸäÿ■ïijŇæĹŚäžňéŻĘäÿ■èőĺèőžæăźæ■őçL'źæőŁèŕ■æşŢåŐżèğčæđŘæŰĞæIJňçŽĎéŮőé
äÿžäžĘè£ŹæăůåĄŽïijŇä¡ăéęŰåĚĹèęĄäžěBNFæĹŰèĂĚEBNFå¡ćåijŔæŇĞåőŽäÿĂäÿłæăĞåĞĘèŕ■æşŢãĂĆ
æŕŤåęĆïijŇäÿĂäÿłçőĂå■ŢæŢřå■ęèąĺè¿¿åijŔèŕ■æşŢåŔŕèČ¡åČŔäÿŃéİćè£ŹæăůïijŽ

```
expr ::= expr + term
    |    expr - term
    |    term

term ::= term * factor
    |    term / factor
    |    factor

factor ::= ( expr )
    |    NUM
```

æĹŰèĂĚïijŇäžěEBNFå¡ćåijŔïijŽ

```
expr ::= term { (+|-) term }*

term ::= factor { (*|/) factor }*

factor ::= ( expr )
    |    NUM
```

åIJÍEBNFäÿ■ïijŃècńåŇĚåŘńåIJÍ { . . . }* äÿ■çŽĎèğĎåĹŹæŸŕåŔŕéĂL'çŽĎãĂĆ*äžčèąĺ0æňąæĹŰåd'Žæ

çŐřåIJĺïijŇåęĆæđIJäj¡ăåŕźBNFçŽĎåůěä¡IJæIJžåĹűè£Ÿäÿ■æŸŕå¿ĹæŸŐçŹ¡çŽĎèŕİïijŇåřśæŁŁåőČå¡ŞåĄ
äÿĂèĹňæİěèőšïijŃèğčæđŘçŽĎåŐ§çŘĘåřśæŸŕä¡ăåĹl'çŤĺBNFåőŇæĹŘåd'ŽäÿłæŻ£æ■ćåŠŇæL'l'åśŢäžěåŇźéĚ
äÿžäžĘæijŤçd'žïijŇåĄĞèő¿ä¡ăæ■čåIJÍèğčæđŘå¡ćåęĆ 3 + 4 * 5 çŽĎèąĺè¿¿åijŔãĂĆ
è£Źäÿłèąĺè¿¿åijŔåĚĹèęĄéĂŽè£Ğä¡£çŤĺ2.18èŁĆäÿ■äžŃçż■çŽĎæŁĂæIJŕåĹĘèğčäÿžäÿĂçżĎäžd'çL'ŇæţĄãĂ
çżŞæđIJåŔŕèČ¡æŸŕåČŔäÿŃåĹŮè£ŹæăůçŽĎäžd'çL'ŇåžŔåĹŮïijŽ

```
NUM + NUM * NUM
```

åIJÍæ■d'å§žçąĂäÿŁïijŃ èğčæđŘåŁĺä¡IJäijŽèŕŢçİĂåŐžéĂŽè£ĞæŻ£æ■ćæŞ■ä¡IJåŃźéĚ■èŕ■æşŢåĹřè¿ŞåĚ

```
expr
expr ::= term { (+|-) term }*
expr ::= factor { (*|/) factor }* { (+|-) term }*
```

```
expr ::= NUM { (*|/) factor }* { (+|-) term }*
expr ::= NUM { (+|-) term }*
expr ::= NUM + term { (+|-) term }*
expr ::= NUM + factor { (*|/) factor }* { (+|-) term }*
expr ::= NUM + NUM { (*|/) factor}* { (+|-) term }*
expr ::= NUM + NUM * factor { (*|/) factor }* { (+|-) term }*
expr ::= NUM + NUM * NUM { (*|/) factor }* { (+|-) term }*
expr ::= NUM + NUM * NUM { (+|-) term }*
expr ::= NUM + NUM * NUM
```

äÿŃéÍćæĽĂæIJĽçŽĎèğčæđŘæ■ěéłď'åŘŕèČ¡éIJĂèęĄèŁśçĆźæŮűéŮť'åijĎæŸŐçŹ¡ïijŇä¡ĘæŸŕåőČäżňåĆ
çňňäÿĂäÿłè¿ŞåĚěäžď'çĽŇæŸŕfNUMïijŇåŽăæ■d'æŽ£æ■ćéęŰåĚĹäijŽåŇźéĚ■éĆčäÿłéČĺåĹĘãĂĆ
äÿĂæŮęåŇźéĚ■æĹŘåŁ§ïijŇåřśäijŽè£ŻåĚěäÿŃäÿĂäÿłäžď'çĽŇ+ïijŇäżěæ■d'çśżæŐĺãĂĆ
å¡ŞåůšçżŕçąőåőŽäÿ■èČ¡åŇźéĚ■äÿŃäÿĂäÿłäžď'çĽŇçŽĎæŮűåĂŹïijŇåŘşè¿źçŽĎéČĺåĹĘ(æŕŤåęĆ
{ (*/) factor }*)åřśäijŽèćńæÿĚçŘĘæŐĽãĂĆ åIJĺäÿĂäÿłæĹŘåŁ§çŽĎèğčæđŘäÿ■ïijŇæŢť'äÿłåŘşè¿źé

æIJĽäžĘåĽ■éÍćçŽĎç§ěèŕĘèČŇæŹŕïijŇäÿŃéÍćæĹŚäżňäÿ¿äÿĂäÿłçŐÅå■Ţçđ'žä¿ŃæÌěåśŢçđ'žåęĆä¡ŢæđĹ

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
"""
Topic: äÿŃéŹ■èğčæđŘåŹÍ
Desc :
"""
import re
import collections

# Token specification
NUM = r'(?P<NUM>\d+)'
PLUS = r'(?P<PLUS>\+)'
MINUS = r'(?P<MINUS>-)'
TIMES = r'(?P<TIMES>\*)'
DIVIDE = r'(?P<DIVIDE>/)'
LPAREN = r'(?P<LPAREN>\()'
RPAREN = r'(?P<RPAREN>\))'
WS = r'(?P<WS>\s+)'

master_pat = re.compile('|'.join([NUM, PLUS, MINUS, TIMES,
                                  DIVIDE, LPAREN, RPAREN, WS]))
# Tokenizer
Token = collections.namedtuple('Token', ['type', 'value'])


def generate_tokens(text):
    scanner = master_pat.scanner(text)
    for m in iter(scanner.match, None):
        tok = Token(m.lastgroup, m.group())
        if tok.type != 'WS':
            yield tok
```

```
# Parser
class ExpressionEvaluator:
    '''
    Implementation of a recursive descent parser. Each method
    implements a single grammar rule. Use the ._accept() method
    to test and accept the current lookahead token. Use the ._
↪expect()
    method to exactly match and discard the next token on on the␣
↪input
    (or raise a SyntaxError if it doesn't match).
    '''

    def parse(self, text):
        self.tokens = generate_tokens(text)
        self.tok = None  # Last symbol consumed
        self.nexttok = None  # Next symbol tokenized
        self._advance()  # Load first lookahead token
        return self.expr()

    def _advance(self):
        'Advance one token ahead'
        self.tok, self.nexttok = self.nexttok, next(self.tokens,␣
↪None)

    def _accept(self, toktype):
        'Test and consume the next token if it matches toktype'
        if self.nexttok and self.nexttok.type == toktype:
            self._advance()
            return True
        else:
            return False

    def _expect(self, toktype):
        'Consume next token if it matches toktype or raise␣
↪SyntaxError'
        if not self._accept(toktype):
            raise SyntaxError('Expected ' + toktype)

    # Grammar rules follow
    def expr(self):
        "expression ::= term { ('+'|'-') term }*"
        exprval = self.term()
        while self._accept('PLUS') or self._accept('MINUS'):
            op = self.tok.type
            right = self.term()
            if op == 'PLUS':
                exprval += right
            elif op == 'MINUS':
                exprval -= right
        return exprval
```

```
    def term(self):
        "term ::= factor { ('*'|'/') factor }*"
        termval = self.factor()
        while self._accept('TIMES') or self._accept('DIVIDE'):
            op = self.tok.type
            right = self.factor()
            if op == 'TIMES':
                termval *= right
            elif op == 'DIVIDE':
                termval /= right
        return termval

    def factor(self):
        "factor ::= NUM | ( expr )"
        if self._accept('NUM'):
            return int(self.tok.value)
        elif self._accept('LPAREN'):
            exprval = self.expr()
            self._expect('RPAREN')
            return exprval
        else:
            raise SyntaxError('Expected NUMBER or LPAREN')


def descent_parser():
    e = ExpressionEvaluator()
    print(e.parse('2'))
    print(e.parse('2 + 3'))
    print(e.parse('2 + 3 * 4'))
    print(e.parse('2 + (3 + 4) * 5'))
    # print(e.parse('2 + (3 + * 4)'))
    # Traceback (most recent call last):
    #   File "<stdin>", line 1, in <module>
    #   File "exprparse.py", line 40, in parse
    #   return self.expr()
    #   File "exprparse.py", line 67, in expr
    #   right = self.term()
    #   File "exprparse.py", line 77, in term
    #   termval = self.factor()
    #   File "exprparse.py", line 93, in factor
    #   exprval = self.expr()
    #   File "exprparse.py", line 67, in expr
    #   right = self.term()
    #   File "exprparse.py", line 77, in term
    #   termval = self.factor()
    #   File "exprparse.py", line 97, in factor
    #   raise SyntaxError("Expected NUMBER or LPAREN")
    #   SyntaxError: Expected NUMBER or LPAREN
```

```
if __name__ == '__main__':
    descent_parser()
```

èőÍèőž

æŰĞæIJňèğčæđŘæŸŕäÿĂäÿłå¿Ĺåď'ğçŽĎäÿżéćŸïijŇ äÿĂèĹňäijŽå■ăçŤĺå■ęçŤ§å■ęäźăçijŰèŕŚèŕ¿çĺŃæŮ
åęĆæđIJä¡ääIJĺæĽ¿åŕżåĚşäžŐèŕ■æşŢïijŇèğčæđŘçőŮæşŢç■Ľçŕ§åĚşçŽĎèČŇæŹŕç§ěèŕĘçŽĎèŕİïijŇä¡ääžŤèŕ
å¿ĹæŸ¿çĎűïijŇåĚşäžŐè£ŹæŰźéİćçŽĎåĘĚåőźåď'łåď'ŽïijŇäÿ■åŔŕèČ¡åIJĺè£ŹéĞŇåĚĺéČĺåśŢåijĂãĂĆ

åř¡çőąåęĆæ■ď'ïijŇçijŰåĘŹäÿĂäÿłéĂŠå¡ŠäÿŃéŹ■èğčæđŘåŹĺçŽĎæŢť'ä¡ŞæĂİèůŕæŸŕæŕŤè¿ČçőĂå■ŢçŽ
åijĂåğŃçŽĎæŮűåĂŹïijŇä¡ääĚĹèŐůå¿ŮæĽ'ĂæIJĽçŽĎèŕ■æşŢèğĎåĹŹïijŇçĎűåŘŐåřĘåĚűè¡ňæ■ćäÿžäÿĂäÿ
åŽăæ■ď'åęĆæđIJä¡ăçŽĎèŕ■æşŢçśżäijijè£Źæăůïijž

```
expr ::= term { ('+'|'-') term }*

term ::= factor { ('*'|'/') factor }*

factor ::= '(' expr ')'
    | NUM
```

ä¡ääžŤèŕěéęŰåĚĹåřĘåőČäżňè¡ňæ■ćæĹŘäÿĂçżĎåČŔäÿŃéİćè£ŹæăůçŽĎæŰźæşŢïijž

```
class ExpressionEvaluator:
    ...
    def expr(self):
    ...
    def term(self):
    ...
    def factor(self):
    ...
```

æŕŔäÿłæŰźæşŢèęĄåőŇæĹŘçŽĎäżżåŁąå¿ĹçőĂå■Ţ - åőČå£ĚéążäżŐåůęèĞşåŔşéĄ■åŐĘèŕ■æşŢèğĎåĹŹ
äżŐæ§Řçğ■æĎŔäźĽäÿŁèőšïijŇæŰźæşŢçŽĎçŻőçŽĎåřśæŸŕèęĄäźĹåď'ĎçŘĘåőŇèŕ■æşŢèğĎåĹŹïijŇèęĄäźĹ
äÿžäžĘè£Źæăůå ĄŽïijŇéIJĂéĞĞçŤĺäÿŃéİćçŽĎè£ŹäžŻåőđçŐřæŰźæşŢïijŽ

- åęĆæđIJèğĎåĹŹäÿ■çŽĎäÿŃäÿłçňęåŔůæŸŕåŔęåď'ŰäÿĂäÿłèŕ■æşŢèğĎåĹŹçŽĎåŘ■å■Ů(æŕŤåęĆtermæ
  è£Źåřśæ Ÿŕèřěçőůæşŧäÿ■"äÿŃéŹ■"çŽĎçŤśæİě - æŐğåĹűäÿŃéŹ■åĹřåŔęäÿĂäÿłèŕ■æşŢèğĎåĹŹäÿ■åŐ
  æIJĽæŮűåĂŹèğĎåĹŹäijŽèřČçŤĺåůšçżŔæĽğèąŇçŽĎæŰźæşŢ(æŕŤåęĆïijŇåIJĺ `factor ::= '('expr ')'` äÿ■åŕźexprçŽĎèřČçŤĺ)ãĂĆ
  è£ŹåřśæŸŕçőŮæşŢäÿ■"éĂŠå¡Š"çŽĎçŤśæİěãĂĆ
- åęĆæđIJèğĎåĹŹäÿ■äÿŃäÿĂäÿłçňęåŔůæŸŕäÿłçĽźæőŁçňęåŔů(æŕŤåęĆ()ïijŇä¡ää¿Ůæ§ěæĽ¿äÿŃäÿĂäÿ
  åęĆæđIJäÿ■åŇźéĚ■ïijŇåřśäžğçŤ§äÿĂäÿłèŕ■æşŢéŤŹèŕŕãĂĆè£ŹäÿĂèŁĆäÿ■çŽĎ `_expect()` æŰźæşŢåřśæŸŕçŤĺæİěåĄŽè£ŹäÿĂæ■ěçŽĎãĂĆ
- åęĆæđIJèğĎåĹŹäÿ■äÿŃäÿĂäÿłçňęåŔůäÿžäÿĂäžŻåŔŕèČ¡çŽĎéĂĽæŃľéąź(æŕŤåęĆ + æĹŰ -)ïijŇ ä¡ääå£ĚéążåřźæŕŔäÿĂçğ■åŔŕèČ¡æČĚåĘţæčĂæ§ěäÿŃäÿĂäÿłäżď'çĽŇïijŇåŔłæIJĽå¡ŞåőČåŇ
  è£Źäź§æŸŕæIJňèŁĆçď'žä¿Ňäÿ■ `_accept()` æŰźæşŢçŽĎçŻőçŽĎãĂĆ

åőČçŻÿå¡Şäž0_expect()æŰźæşŢçŽĎåijśåŇŰçĽĹæIJñïijŇåŻääÿžåęĆæđIJäÿĂäÿłåŇźéĚ■æĽ¿åĹřäžĘåę
ä¡ĘæŸŕåęĆæđIJæšąæĽ¿åĹřïijŇåőČäÿ■äijŽäžğçŢ§éŤŹèŕŕèĂŇæŸŕåŻđæżŽ(åĚĄèőÿåŘŐçż■çŽĎæčĂæş

- åŕźäžŐæIJĽéĞ■åď■éČĺåĹĘçŽĎèğĎåĹŹ(æŕŤåęĆåIJĺèğĎåĹŹèąĺè¿¿åijŔ ::= term { ('+'|'-') term }* äÿ■)ïijŇ éĞ■åď■åŁĺä¡IJéĂŽè£ĞäÿĂäÿłwhileå¿łçŐŕæİěåődçŐřãĂĆ
å¿łçŐŕäÿżä¡Şäij ŽæŤűéŽĘæĹŰåď ĎçŘĘæĽĂæIJĽçŽĎéĞ■åď■åĚČçt'ăçŻt'åĹřæšąæIJĽåĚűäžŰåĚČçt'ă
- äÿĂæŮęæŢt'äÿłèŕ■æşŢèğĎåĹŹåď ĎçŘĘåőŇæĹŘïijŇæŕŔäÿłæŰźæşŢäijŽè£ŤåŻđæ§Řçğ■çżŞæđIJçżŹè
è£ŹåřśæŸŕåIJĺèğčæđŘè£ĞçÍŃäÿ■åĂijæŸŕæĂŐæăůçt'ŕåŁăçŽĎåŐ§çŘĘãĂĆ
æŕŤåęĆïijŇåIJĺèąĺè¿¿åijŔæśĆåĂijçÍŃåžŔäÿ■ïijŇè£ŤåŻđåĂijäžčèąĺèąĺè¿¿åijŔèğčæđŘåŘŐçŽĎéČĺåĹĘç
æIJĂåŘŐæĽĂæIJĽåĂijäijŽåIJĺæIJĂéąűåśĆçŽĎèŕ■æşŢèğĎåĹŹæŰźæşŢäÿ■åŘĹåźűètůæİěãĂĆ

åř¡çőąåŘŚä¡ăæijŤçd'žçŽĎæŸŕäÿĂäÿłçőĂå■ŢçŽĎä¿Ńå■ŘïijŇéĂŠå¡ŠäÿŃéŹ■èğčæđŘåŹĺåŔŕäžěçŤĺæİěå
æŕŤåęĆïijŇPythonèŕ■èĺĂæIJñèžńåřśæŸŕéĂŽè£ĞäÿĂäÿłéĂŠå¡ŠäÿŃéŹ■èğčæđŘåŹĺåŐžèğčéĞŁçŽĎãĂĆ
åęĆæđIJä¡ååŕźæ■d'æĎ§åĚt'èűčïijŇä¡ăåŔŕäžěéĂŽè£Ğæ§ěçIJŃPythonæžŘçăĄæŰĞäžűGrammar/GrammaræĺçIJŃåőŇä¡ăäijŽåŘŠÇŕïijŇéĂŽè£ĞæĽŃåŁĺæŰźåijŔåŐžåőđçŐřäÿĂäÿłèğčæđŘåŹĺåĚűåőđäijŽæIJĽå¿Ĺåd'Žç

åĚűäÿ■äÿĂäÿłåśĂéŹŘåřśæŸŕåőČäžňäÿ■èČ¡èćńçŤĺäžŐåŇĚåŘńäžża¡Ţåůę éĂŠå¡ŞçŽĎèŕ■æşŢèğĎåĹŹäÿ

```
items ::= items ',' item
    | item
```

äÿžäžĘè£ŹæăůåĄŽïijŇä¡ăåŔŕèČ¡äijŽåČŔäÿŃéÍćè£ŹæăůäĄ£çŤÍ `items()` æŰźæşŢïijŽ

```
def items(self):
    itemsval = self.items()
    if itemsval and self._accept(','):
        itemsval.append(self.item())
    else:
        itemsval = [ self.item() ]
```

åŤŕäÿĂçŽĎéŮőécŸæŸŕè£ŹäÿłæŰźæşŢæăźæIJñäÿ■èČ¡åůëä¡IJïijŇäžŃåőđäÿŁïijŇåőČäijŽäžğçŢ§äÿĂäÿ

åĚşäžŐèŕ■æşŢèğĎåĹŹæIJñèžńä¡ăåŔŕèČ¡äź§äijŽçćřåĹřäÿĂäžŻæčŸæĽŃçŽĎéŮőécŸãĂĆ
æŕŤåęĆïijŇä¡ăåŔŕèČ¡æČşç§ěéĄŞäÿŃéÍćè£ŹäÿłçőĂå■ŢæĽijèŕ■æşŢæŸŕåŘęèąĺè£řå¿Ůå¡ŞïijŽ

```
expr ::= factor { ('+'|'-'|'*'|'/') factor }*

factor ::= '(' expression ')'
    | NUM
```

è£Źäÿłèŕ■æşŢçIJŃäÿŁåŐżæšąåŢěéŮőécŸïijŇä¡ĘæŸŕåőČå■t'äÿ■èČ¡år§ègĽåĹřæăĞåĞĘåŻŻåĹŹè£ŘçőŮ
æŕŤåęĆïijŇèąĺè¿¿åijŔ "3 + 4 * 5" äijŽå¿ŮåĹř35èĂŃäÿ■æŸŕæIJ§æIJŻçŽĎ23.
åĹĘåijÄä¡£çŤĺ"expr"åŠŃ"term"èğĎåĹŹåŕźäžěèőĺ'åőČæ■čçąőçŽĎåůëä¡IJãĂĆ

åŕźäžŐåď■æİÇçŽĎèŕ■æşŢïijŇä¡ăæIJĂåě¡æŸŕéĂĽ'æŃľæ§ŘäÿłèğčæđŘåůëåĚůæŕŤåęĆPyParsingæĹŰèĂ
äÿŃéÍćæŸŕä¡£çŤÍPLYæİěéĞ■åĘŹèąĺè¿¿åijŔæśĆåĂijçÍŃåžŔçŽĎäžčçăĄïijŽ

```
from ply.lex import lex
from ply.yacc import yacc

# Token list
tokens = [ 'NUM', 'PLUS', 'MINUS', 'TIMES', 'DIVIDE', 'LPAREN',
↪'RPAREN' ]
```

```
# Ignored characters
t_ignore = ' \t\n'
# Token specifications (as regexs)
t_PLUS = r'\+'
t_MINUS = r'-'
t_TIMES = r'\*'
t_DIVIDE = r'/'
t_LPAREN = r'\('
t_RPAREN = r'\)'

# Token processing functions
def t_NUM(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Error handler
def t_error(t):
    print('Bad character: {!r}'.format(t.value[0]))
    t.skip(1)

# Build the lexer
lexer = lex()

# Grammar rules and handler functions
def p_expr(p):
    '''
    expr : expr PLUS term
         | expr MINUS term
    '''
    if p[2] == '+':
        p[0] = p[1] + p[3]
    elif p[2] == '-':
        p[0] = p[1] - p[3]


def p_expr_term(p):
    '''
    expr : term
    '''
    p[0] = p[1]


def p_term(p):
    '''
    term : term TIMES factor
    | term DIVIDE factor
    '''
    if p[2] == '*':
        p[0] = p[1] * p[3]
```

```
    elif p[2] == '/':
        p[0] = p[1] / p[3]

def p_term_factor(p):
    '''
    term : factor
    '''
    p[0] = p[1]

def p_factor(p):
    '''
    factor : NUM
    '''
    p[0] = p[1]

def p_factor_group(p):
    '''
    factor : LPAREN expr RPAREN
    '''
    p[0] = p[2]

def p_error(p):
    print('Syntax error')

parser = yacc()
```

è£ŹäÿłçĺŃåžŕäÿ■ïijŇæĽĂæIJL'äžčçăĄéČ¡ä¡■äžŐäÿĂäÿłæŕŤè¿ČéńŸçŽĎåśĆæňąãĂĆä¡ääŕŁéIJĂèęĄäÿ
èĂŇåőđéŹĚçŽĎè£ŘèąŇèğčæđŘåŹĺïijŇæŐěåŔŮäžď'çĽŇç■Ľ'ç■Ľ'åžŢåśĆåŁĺä¡IJåůšçżŔèćńåžŞåĞ¡æŢřåőđçČ

äÿŃéİćæŸŕäÿĂäÿłæĂŐæăůä¡£çŤĺå¿ŮåĹřçŽĎèğčæđŘåŕźèšąçŽĎä¿Ńå■ŘïijŽ

```
>>> parser.parse('2')
2
>>> parser.parse('2+3')
5
>>> parser.parse('2+(3+4)*5')
37
>>>
```

åęĆæđIJä¡ăæČşåIJĺä¡ăçŽĎçijŰçĺŃè£ĞçĺŃäÿ■æİěçĆźæŇŚæĹŸåŠŇåĹžæ£ĂïijŇçijŰåĘŹèğčæđŘåŹĺåŠŇ
åĘ■æňąïijŇäÿĂæIJňçijŰèŕŚåŹĺçŽĎäźęçś■äijŽåŇĚåŘńå¿Ĺåď'ŽåžŢåśĆçŽĎçŘĘèőžç§ěèŕĘãĂĆäÿ■è£Ğå¿Ĺåď'
PythonèĞłåůšçŽĎastæĺąåİŮäź§åĂijå¿ŮåŐžçIJŃäÿĂäÿŃãĂĆ

2.20 å■ŮèŁĆå■ŮçňęäÿšäÿŀçŽĎå■ŮçňęäÿšæŞ■ä¡IJ

éŮőécŸ

ä¡ăæČşåIJĺå■ŮèŁĆå■ŮçňęäÿšäÿŀæĽğèąŇæŹőéĂŽçŽĎæŰĞæIJňæŞ■ä¡IJ(æŕŤåęĆçğżéŹď'ïijŇæŘIJçť'ćå

èğčåĘşæŰźæąĹ

å■ŮèŁĆå■ŮçňęäÿšåŘŇæăůäź§æŤŕæŇĄåď ğéČĺåĹĘåŠŇæŰĞæIJňå■Ůçňęäÿšäÿ Ăæăůç ŽĎåĘĚç¡őæŞ■ä¡

```
>>> data = b'Hello World'
>>> data[0:5]
b'Hello'
>>> data.startswith(b'Hello')
True
>>> data.split()
[b'Hello', b'World']
>>> data.replace(b'Hello', b'Hello Cruel')
b'Hello Cruel World'
>>>
```

è£ŹäžŻæŞ■ä¡IJåŘŇæăůäź§éĂĆçŤĺäžŐå■ŮèŁĆæŢřçżĎãĂĆæŕŤåęĆïijŽ

```
>>> data = bytearray(b'Hello World')
>>> data[0:5]
bytearray(b'Hello')
>>> data.startswith(b'Hello')
True
>>> data.split()
[bytearray(b'Hello'), bytearray(b'World')]
>>> data.replace(b'Hello', b'Hello Cruel')
bytearray(b'Hello Cruel World')
>>>
```

ä¡ăåŔŕäžěä¡£çŤĺæ■čåĹŹèąÍè¿¿åijŔåŇźéĚ■å■ŮèŁĆå■ŮçňęäÿšïijŇä¡ĘæŸŕæ■čåĹŹèąÍè¿¿åijŔæIJňèžńå£Ě

```
>>>
>>> data = b'FOO:BAR,SPAM'
>>> import re
>>> re.split('[:,]',data)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "/usr/local/lib/python3.3/re.py", line 191, in split
return _compile(pattern, flags).split(string, maxsplit)
TypeError: can't use a string pattern on a bytes-like object
>>> re.split(b'[:,]',data) # Notice: pattern as bytes
[b'FOO', b'BAR', b'SPAM']
>>>
```

èőÍèőž

åď ğåď ŽæŢřæČĚåĘţäÿŃïijŇåIJÍæŰĞæIJňå■Ůçňęäÿšäÿ ŁçŽĎæŞ■ä¡IJåİĞåŔŕçŤĺäžŐå■ŮèŁĆå■Ůçňęäÿš
çĎűèĂŇïijŇè£ŹéĞŇäź§æIJL'äÿĂäžŻéIJĂèęĄæşÍæĎŔçŽĎäÿ■åŘŇçĆźãĂĆéęŰåĚĹïijŇå■ŮèŁĆå■Ůçňęäÿšç

```
>>> a = 'Hello World' # Text string
>>> a[0]
'H'
>>> a[1]
'e'
>>> b = b'Hello World' # Byte string
>>> b[0]
72
>>> b[1]
101
>>>
```

è£Źçğ■èŕ■äźĽäÿŁçŽĎåŇžåĹńäijŽåŕźäžŐåď ĎçŘĘéİćåŘŠå■ŮèŁĆçŽĎå■Ůçňęæţřæ■őæIJL˙å¡śåŞ■ãĂĆ

çňňäžŇçĆźïijŇå■ŮèŁĆå■Ůçňęäÿšäÿ■äijŽæŔŘä¿ŻäÿĂäÿłç¿ŐèğĆçŽĎå■Ůçňęäÿšèąĺçď˙žïijŇäź§äÿ■èČ¡å

```
>>> s = b'Hello World'
>>> print(s)
b'Hello World' # Observe b'...'
>>> print(s.decode('ascii'))
Hello World
>>>
```

çśžäijijçŽĎïijŇäź§äÿ■å■ŸåIJĺäżża¡ŢéĂĆçŤĺäžŐå■ŮèŁĆå■ŮçňęäÿšçŽĎæäijåijŔåŇŰæŞ■ä¡IJïijŽ

```
>>> b'%10s %10d %10.2f' % (b'ACME', 100, 490.1)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for %: 'bytes' and 'tuple'
>>> b'{} {} {}'.format(b'ACME', 100, 490.1)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: 'bytes' object has no attribute 'format'
>>>
```

åęĆæđIJä¡ăæČşæäijåijŔåŇŰå■ŮèŁĆå■ŮçňęäÿšïijŇä¡ăå¿ŮåĚĹä¡£çŤĺæăĞåĞĘçŽĎæŰĞæIJňå■Ůçňęäÿš

```
>>> '{:10s} {:10d} {:10.2f}'.format('ACME', 100, 490.1).encode(
↪'ascii')
b'ACME 100 490.10'
>>>
```

æIJĂåŘŐéIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇä¡£çŤĺå■ŮèŁĆå■ŮçňęäÿšåŔŕèČ¡äijŽæŤźåŔŸäÿĂäžŻæŞ■ä¡IJçŽİ
æŕŤåęĆïijŇåęĆæđIJä¡ăä¡£çŤĺäÿĂäÿłçijŰçăĄäÿžå■ŮèŁĆçŽĎæŰĞäżűåŘ■ïijŇèĂŇäÿ■æŸŕäÿĂäÿłæŹőéĂŽçŽ

```
>>> # Write a UTF-8 filename
>>> with open('jalape\xf1o.txt', 'w') as f:
...     f.write('spicy')
...
>>> # Get a directory listing
>>> import os
```

```
>>> os.listdir('.') # Text string (names are decoded)
['jalapeÃśo.txt']
>>> os.listdir(b'.') # Byte string (names left as bytes)
[b'jalapen\xcc\x83o.txt']
>>>
```

æşÍæĎŔä¿Ńå■Řäÿ■çŽĎæIJĂåŘŐéČĺåĹĘçżŹçŻőå¡ŢåŘ■äijăéĂŠäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšæŸŕæĂŐæă
åIJĺçŻőå¡Ţäÿ■çŽĎæŰĞäžűåŘ■åŇĚåŘńåŐ§ågŃçŽĎUTF-8çijŰçăĄãĂĆ
åŔĆèĂČ5.15åřŔèŁĆèŐůåŔŰæŻť åď ŽæŰĞäžűåŘ■çŻÿåĚşçŽĎåĘĚåőźãĂĆ

æIJĂåŘŐæŔŘäÿĂçĆźïijŇäÿĂäžŻçĺŃåžŔåŚŸäÿžäžĘæŔŘå■ĞçĺŃåžŔæĽğèąŇçŽĎéĂ§åžęäijŽåĂ¿åŘŚäž
åř¡çőąæŞ■ä¡IJå■ŮèŁĆå■ŮçňęäÿšçąőåőđäijŽæŕŤæŰĞæIJňæŻť åŁăénŸæŢĹ(åŻăäÿžåď ĎçŘĘæŰĞæIJňåŻžæI
è£Źæăůå ĄŽéĂŽåÿÿäijŽåŕijèĞť éİđåÿÿæİĆäźśçŽĎäžčçăĄãĂĆä¡ăäijŽçżŔåÿÿåŔŚçŐřå■ŮèŁĆå■Ůçňęäÿšåźűäÿ
åźűäÿŤä¡ăè£Ÿå¿ŮæĽŃåŁĺåď ĎçŘĘæĽĂæIJĽçŽĎçijŰçăĄ/èğčçăĄæŞ■ä¡IJãĂĆ
åİęçŹ¡èőšïijŇåęĆæđIJä¡ăåIJĺåď ĎçŘĘæŰĞæIJňçŽĎèŕİïijŇåřśçŻť æŐěåIJĺçĺŃåžŔäÿ■ä¡£çŤĺæŹőéĂŽçŽĎæŰĞ

CHAPTER 5

çňňäÿĽçńäïijŽæŢřå■ÙæÙěæIJ§åŠŇæÙűéÙť

åIJÍPythonäÿ■æĽğèąŇæŢťæŢřåŠŇæţőçĆźæŢřçŽĎæŢřå■ęè£ŘçőÙæÙűå¿ĹçőĂå■ŢçŽĎãĂĆ åř¡çőąåęĆæ■ď'ïijŇåęĆæđIJä¡ăéIJĂèęĄæĽğèąŇåĹĘæŢřãĂĄæŢřçżĎæĹŰèĂĚæŸŕæÙěæIJ§åŠŇæÙűéÙťçŽĎ æIJňçńäéŻĘäÿ■èőÍèőžçŽĎåřśæŸŕè£ŹäžŻäÿżécŸãĂĆ

Contents:

3.1 æŢřå■ÙçŽĎåŻŻèĹ■äžŤåĚě

éÙőécŸ

ä¡ăæČşåŕźæţőçĆźæŢřæĽğèąŇæŇĞåőŽçš¿åžęçŽĎèĹ■åĚěè£ŘçőÙãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐçőĂå■ŢçŽĎèĹ■åĚěè£ŘçőŮïijŇä¡£çŤĺåĘĚç¡őçŽĎ `round(value, ndigits)` åĞ¡æŢřå■şåŔŕãĂĆæŕŤåęĆïijŽ

```
>>> round(1.23, 1)
1.2
>>> round(1.27, 1)
1.3
>>> round(-1.27, 1)
-1.3
>>> round(1.25361,3)
1.254
>>>
```

å¡Şäÿ Ăäÿłå Ăijå ĹŽåě¡åIJÍäÿď äÿłè¿źçŢŇçŽĎäÿ■éŮť çŽĎæŮűůå ĂŹïijŇ round åĞ¡æŢřè£ŤåŻđçężåőČæIJĂè£ŚçŽĎåĄűæŢřãĂĆ äź§åřśæŸŕèŕť ïijŇåŕź1.5æĹŰèĂĚ2.5çŽĎèĹ■åĚèè£ŘçőŮéČ¡

äijăçżŹ round() åĞ¡æŢřçŽĎ ndigits åŔĆæŢřåŔŕäžěæŸŕèť §æŢřïijŇè£Źçğ■æČĚåĘţäÿŃïijŇ èĹ■åĚèè£ŘçőŮäijŽä¡IJçŤĺåIJÍå■Ąä¡■ãĂĄçŹ¿ä¡■ãĂĄå■Čä¡■ç■Ľäÿ ŁéÍćãĂĆærŤåęĆïijŽ

```
>>> a = 1627731
>>> round(a, -1)
1627730
>>> round(a, -2)
1627700
>>> round(a, -3)
1628000
>>>
```

èőÍèőž

äÿ■èęĄåřĘèĹ■åĚěåŠŇæăijåijŔåŇŰè¿ŞåĞžæŘđæůůæůĘäžĘãĂĆ åęĆæđIJä¡ăçŽĎçŽőçŽĎåŔłæŸŕçőĂå■ŢçŽĎè¿ŞåĞžäÿĂåőŽåő¡åžęçŽĎæŢřïijŇä¡ăäÿ■éIJĂèęĄä¡£çŤÍ round() åĞ¡æŢřãĂĆ èĂŇäžĚäžĚåŔłéIJĂèęĄåIJÍæăijåijŔåŇŰçŽĎæŮűůåĂŹæŇĞåőŽçš¿åžęå■şåŔŕãĂĆærŤ

```
>>> x = 1.23456
>>> format(x, '0.2f')
'1.23'
>>> format(x, '0.3f')
'1.235'
>>> 'value is {:0.3f}'.format(x)
'value is 1.235'
>>>
```

åŘŇæăůïijŇäÿ■èęĄèŕŢçÍĂåŐżèĹ■åĚěæţőçĆźåĂijæÍě”ä£őæ■č”èąĺéÍćäÿŁçIJŃèţůæÍěæ■čçąőçŽĎéŮőéćŸ

```
>>> a = 2.1
>>> b = 4.2
>>> c = a + b
>>> c
6.300000000000001
>>> c = round(c, 2) # "Fix" result (???)
>>> c
6.3
>>>
```

åŕźäžŐåď ğåď ŽæŢřä¡£çŤĺåĹřæţőçĆźçŽĎçĺŃåžŔïijŇæšąæIJL’å£ĚèęĄäź§äÿ■æŐÍè■Řè£ŹæăůåĄŽãĂĆ åř¡çőąåIJÍèőąçőŮçŽĎæŮűůåĂŹäijŽæIJL’äÿĂçĆźçĆźåřŔçŽĎèŕřåůőïijŇä¡ĘæŸŕè£ŹäžŽåřŔçŽĎèŕřåůőæŸŕèČ¡è åęĆæđIJäÿ■èČ¡åĚĄèőÿè£ŹæăůçŽĎåřŔèŕřåůő(æŕŤåęĆæűĽ’åŘŁåĹřéĞŚèđ■écĘå§§)ïijŇéĆčäźĹåřšå¿ŮèĂČèŹ decimal æÍąåİŮäžĘïijŇäÿŃäÿĂèŁĆæĹŚäžnäijŽèŕęçżĘèőÍèőžãĂĆ

3.2 æĽğèąŇçš¿çąőçŽĎæţőçĆźæŢřè£ŘçőŮ

éŮőécŸ

ä¡ăéIJĂèęĄåŕźæţőçĆźæŢřæĽğèąŇçš¿çąőçŽĎèőąçőŮæŞ■ä¡IJïijŇåźűäÿŤäÿ■åÿŇæIJŻæIJL'äżżä¡ŢåřŔèŕŕ

èğčåĘşæŰźæąĹ

æţőçĆźæŢřçŽĎäÿĂäÿłæŹőéĄ■éŮőécŸæŸŕåőČäžňåźűäÿ■èČ¡çš¿çąőçŽĎèąĺçd'žå■Ąè£ŻåĹűæŢřãĂĆ åźűäÿŤïijŇå■şä¡£æŸŕæIJĂçőĂå■ŢçŽĎæŢřå■ęè£ŘçőŮäź§äijŽäžğçŤ§åřŔçŽĎèŕŕåůőïijŇæŕŤåęĆïijŽ

```
>>> a = 4.2
>>> b = 2.1
>>> a + b
6.300000000000001
>>> (a + b) == 6.3
False
>>>
```

è£ŹäžŻéŤŹèŕŕæŸŕçŤśåžŢåśĆCPUåŠŇIEEE 754æăĞåĞĘéĂŽè£ĞèĞłåůśçŽĎæţőçĆźå■Ţä¡■åŐżæL'ğèąŇçŤśäžŐPythonçŽĎæţőçĆźæŢřæ■őçśżåđŇä¡£çŤĺåžŢåśĆèąĺçd'žå■ŸåĆĺæŢřæ■őïijŇåŽăæ■d'ä¡äæšąåŁđæşŢåŐ

åęĆæđIJä¡äæČşæŻť'åŁäçš¿çąő(åźűèČ¡åőźå£■äÿĂåőŽçŽĎæĂğèČ¡æ■§èĂŮ)ïijŇä¡ääŔŕäžěä¡£çŤĺ decimal æĺąåİŮïijŽ

```
>>> from decimal import Decimal
>>> a = Decimal('4.2')
>>> b = Decimal('2.1')
>>> a + b
Decimal('6.3')
>>> print(a + b)
6.3
>>> (a + b) == Decimal('6.3')
True
```

åĹİçIJŃètůæİěïijŇäÿŁéİćçŽĎäżčçăĄåě¡åČŔæIJL'çĆźåěĞæĂłïijŇæŕŤåęĆæĹŚäžnçŤĺå■Ůçňęäÿšæİèèąĺçd çĎűèĂŇïijŇ Decimal åŕźèśąäijŽåČŔæŹőéĂŽæţőçĆźæŢřäÿĂæăůçŽĎåůěä¡IJ(æŤŕæŇĄæL'ĂæIJL'çŽĎåÿÿçŤ åęĆæđIJä¡äæL'Şå■ŕåőČäžňæĹŰèĂĚåIJĺå■ŮçňęäÿšæăijåijŔåŇŰåĞ¡æŢřäÿ■ä¡£çŤĺåőČäžňïijŇçIJŃètůæİèèů§a

decimal æĺąåİŮçŽĎäÿĂäÿłäÿżèęĄçL'źå¿ĄæŸŕåĚĄèőÿä¡äæŐğåĹűèőąçőŮçŽĎærŔäÿĂæŰźéİćïijŇåŇ äÿžäžĘè£ŹæăůåĄŽïijŇä¡ääĚĹå¿ŮåĹŻåžžäÿĂäÿłæIJňåIJřäÿŁäÿŇæŰĞåźűæŽť'æŤźåőČçŽĎèő¿ç¡őïijŇæŕŤåęĆ

```
>>> from decimal import localcontext
>>> a = Decimal('1.3')
>>> b = Decimal('1.7')
>>> print(a / b)
0.76470588235294117647058823529
>>> with localcontext() as ctx:
...     ctx.prec = 3
```

```
... print(a / b)
...
0.765
>>> with localcontext() as ctx:
... ctx.prec = 50
... print(a / b)
...
0.76470588235294117647058823529411764705882352941176
>>>
```

èőĺèőž

decimal æĺąåİŮåőđçŐřäžĘIBMçŽĎ"éĂŽçŤĺåřŔæŢřè£ŘçőŮèğĎèŃČ"ãĂĆäÿ■çŤĺèŕť'ïijŇæIJĽå¿Ĺåď'Ž

PythonæŰřæĽŃäijŽåĂ¿åŔŚäžŐä¡£çŤĺ decimal æĺąåİŮæİĕåď'ĎçŘĘæţőçĆźæŢřçŽĎçš¿çąőè£ŘçőŮãĂ çĎűèĂŇïijŇåĚĹçŘĘèğčä¡ăçŽĎåžŤçŤĺçĺŃåžŔçŻőçŽĎæŸŕéİđåÿÿéĞ■èęAçŽĎãĂĆ åęĆæđIJä¡ăæŸŕåIJĺåĄŽçğŚå■ęèőąçőŮæĹŰåůěçĺŃéćĘå§§çŽĎèőąçőŮãĂAçŤţèĎŚçżŸåŻ¿ïijŇæĹŰèĂĚæŸŕç éĆčäźĹä¡£çŤĺæŹőéĂŽçŽĎæţőçĆźçśżåđŃæŸŕæŕŤè¿ČæŹőéĄ■çŽĎåĄŽæşŢãĂĆ åĚűäÿ■äÿĂäÿłåŐ§åŻăæŸŕïijŇåIJĺçIJ§åőđäÿŰçŢŇäÿ■å¿ĹåřŚäijŽèęĄæśĆçš¿çąőåĹřæŹőéĂŽæţőçĆźæŢřèČ¡æ åŻăæ■d'ïijŇèőąçőŮè£ĞçĺŃäÿ■çŽĎéĆčäźĹäÿĂçĆźçĆźçŽĎèŕŕåůőæŸŕèćńåĚĄèőÿçŽĎãĂĆ çňňäžŇçĆźåřśæŸŕïijŇåŐ§çŤ§çŽĎæţőçĆźæŢřèőąçőŮèęĄå£ńçŽĎåď'Ž-æIJĽæŮűåĂŹä¡ăåIJĺæĽğèąŇåď'ğéĞŔè£ŘçőŮçŽĎæŮűåĂŹéĂ§åžęäź§æŸŕéİđåÿÿéĞ■èęAçŽĎãĂĆ

å■şä¿£åęĆæ■d'ïijŇä¡ăå■ťäÿ■èČ¡åőŇåĚĺå£¡çŢěèŕŕåůőãĂĆæŢřå■ęåőűèŁśäžĘåď'ğéĞŔæŮűéŮť'åŐżçăŤçl ä¡ăäź§å¿ŮæşĺæĎŔäÿŇåĞŔæşŢåĹăéŹď'äżěåŔŁåď'ğæŢřåŠŇåřŔæŢřçŽĎåŁăåĹĘè£ŘçőŮæĽĂåÿęæİěçŽĎå¡śå

```
>>> nums = [1.23e+18, 1, -1.23e+18]
>>> sum(nums) # Notice how 1 disappears
0.0
>>>
```

äÿŁéİćçŽĎéŤŹèŕŕåŔŕäżěåĹľçŤĺ math.fsum() æĽĂæŔŘä¿ŻçŽĎæŻť'çš¿çąőèőąçőŮèČ¡åŁŻæİĕèğčåĘ

```
>>> import math
>>> math.fsum(nums)
1.0
>>>
```

çĎűèĂŇïijŇåŕźäžŐåĚűäżŰçŽĎçőŮæşŢïijŇä¡ăåžŤèŕěäžŤçżĘçăŤçľ'űåőČåźűçŘĘèğčåőČçŽĎèŕŕåůőäžğçŤ

æĂżçŽĎæİĕèŕť'ïijŇ decimal æĺąåİŮäÿżèęAçŤĺåIJĺæűL'åŔŁåĹřéĞŚèđ■çŽĎéćĘå§§ãĂĆ åIJĺè£ŹçśżçĺŃåžŔäÿ■ïijŇåŞłæĂŢæŸŕäÿĂçĆźåřŔåřŔçŽĎèŕŕåůőåIJĺèőąçőŮè£ĞçĺŃäÿ■èŤŞåżűéČ¡æŸŕäÿ■åĚA åŻăæ■d'ïijŇ decimal æĺąåİŮäÿžèğčåĘşè£ŹçśżéŮőéćŸæŔŘä¿ŻäžĘæŰźæşŢãĂĆ å¡ŞPythonåŠŇæŢřæ■őåžŞæĽŞäžď'éĄŞçŽĎæŮűåĂŹäź§éĂŽåÿÿäijŽéĄĞåĹř Decimal åŕźèśąïijŇåźűäÿŤïijŇéĂŽåÿÿäź§æŸŕåIJĺåď'ĎçŘĘéĞŚèđ■æŢřæ■őçŽĎæŮűåĂŹãĂĆ

3.3 æŢřå■ŮçŽĎæăijåijŔåŇŰè¿ŞåĞž

éŮőécŸ

ä¡ăéIJĂèęĄåřĘæŢřå■Ůæăijåijŕ́åŇŰåŘŐè¿ŞåĞžïijŇåźűæŐğåĹűæŢřå■ŮçŽĎä¡■æŢřãĂĄåŕźé¡ŘãĂĄå■Č

èğčåĘşæŰźæąĹ

æăijåijŔåŇŰè¿ŞåĞžå■Ţäÿłæ Ţřå■ŮçŽĎæŮűåĂŹïijŇåŔŕäžëä¡£çŤĺåĘĚç¡őçŽĎ format() åĞ¡æŢřïijŇærŤåęĆïijŽ

```
>>> x = 1234.56789

>>> # Two decimal places of accuracy
>>> format(x, '0.2f')
'1234.57'

>>> # Right justified in 10 chars, one-digit accuracy
>>> format(x, '>10.1f')
'    1234.6'

>>> # Left justified
>>> format(x, '<10.1f')
'1234.6    '

>>> # Centered
>>> format(x, '^10.1f')
'  1234.6  '

>>> # Inclusion of thousands separator
>>> format(x, ',')
'1,234.56789'
>>> format(x, '0,.1f')
'1,234.6'
>>>
```

åęĆædIJä¡ăæČşä¡£çŤĺæŇĞæŢřèőřæşŢïijŇåřĘfæŤźæĹŘeæĹŰèĂĚE(åŔŰåĘşäžŐæŇĞæŢřè¿ŞåĞžçŽĎå

```
>>> format(x, 'e')
'1.234568e+03'
>>> format(x, '0.2E')
'1.23E+03'
>>>
```

åŘŇæŮűæŇĞåőŽåő¡åžęåŠŇçš¿åžęçŽĎäÿĂèĹňå¡ćåijŔæŸŕ '[<>^]?width[,]?(.digits)?' ïijŇ åĚűäÿ■ width åŠŇ digits äÿžæŢťæŢřïijŇïij§äžčèąĺåŔŕéĂL'éČĺåĹĘãĂĆ åŘŇæăůçŽĎæăijåijŔäź§èćńçŤĺåIJĺå■Ůçňęäÿšç ŽĎ format() æŰźæşŢäÿ■ãĂĆærŤåęĆïijŽ

```
>>> 'The value is {:0,.2f}'.format(x)
'The value is 1,234.57'
>>>
```

èőĺèőž

æŢřå■Ůæăijåijŕå№Ůè¿ŞåĞžéĂŽåÿÿæŸŕæŕŤè¿ČçőĂå■ŢçŽĎãĂĆäÿŁéĺćæijŤçď žçŽĎæŁĂæIJŕåŘ№æŮű decimal æĺąåİŮäÿ■çŽĎ Decimal æŢřå■Ůåŕźèśąã ĂĆ

å¡ŞæŇĞåőŽæŢřå■ŮçŽĎä¡■æŢřåŘŐïijŇçżŞæđIJåĂijäijŽæăźæ■ő round() åĞ¡æŢřåŘŇæăůçŽĎèğĎåĹŹè£Żèą№åŻŻèĹ■äžŤåĚěåŘŐè£ŤåŻđãĂĆæŕŤåęĆïijŽ

```
>>> x
1234.56789
>>> format(x, '0.1f')
'1234.6'
>>> format(-x, '0.1f')
'-1234.6'
>>>
```

å№ĚåŘńå■Ča¡■çňęçŽĎæăijåijŕå№Ůèů§æIJňåIJřå№Ůæšąæ IJL'åĚşçşžãĂĆ åęĆæđIJä¡ăéIJĂèęĄæăźæ■őåIJřå№ŽæİèæŸ¿çď žå■Ča¡■çňęïijŇä¡ăéIJĂèęĄèĞłåůšåŐżèřČæ§ěäÿŃ locale æĺąåİŮäÿ■çŽĎåĞ¡æŢřäžĘãĂĆ ä¡ååŘŇæăůäź§åŕŕäżěä¡£çŤĺå■Ůçňęäÿšç ŽĎ translate() æŰźæşŢæİěäžď'æ■ćå■Ča¡■çňęãĂĆæŕŤåęĆïijŽ

```
>>> swap_separators = { ord('.'):',', ord(','):'.' }
>>> format(x, ',').translate(swap_separators)
'1.234,56789'
>>>
```

åIJĺå¿Ĺåď'ŽPythonäžčçăĄäÿ■äijŽçIJŃåĹřä¡£çŤĺ%æİěæăijåijŕå№ŮæŢřå■ŮçŽĎïijŇæŕŤåęĆïijŽ

```
>>> '%0.2f' % x
'1234.57'
>>> '%10.1f' % x
'    1234.6'
>>> '%-10.1f' % x
'1234.6    '
>>>
```

è£Źçğ■æăijåijŕå№ŮæŰźæşŢäź§æŸŕåŕŕèąŇçŽĎïijŇäÿ■è£ĞæŕŤæŻť'åŁăåĚĹè£ŻçŽĎ format() èęĄåůőäÿĂçĆźãĂĆ æŕŤåęĆïijŇåIJĺä¡£çŤĺ%æŞ■ä¡IJçňęæăijåijŕå№ŮæŢřå■ŮçŽĎæŮűåĂŹïijŇäÿ

3.4 äžŇåĚńå■ĄåĚ■è£ŻåĹűæŢťæŢř

éŮőécŸ

ä¡ăéIJĂèęÀè¡ňæ■ćæĹŰèĂĚè¿ŞåĞžä¡£çŤĺäžŇè£ŻåĹűïijŇåĚńè£ŻåĹűæĹŰå■ĄåĚ■è£ŻåĹűèąÍçd'žçŽĎæŢ

èğčåĘşæŰźæąĹ

äÿžäžĘåřĘæŢť'æŢřè¡ňæ■ćäÿžäžŇè£ŻåĹűãĂĄåĚńè£ŻåĹűæĹŰå■ĄåĚ■è£ŻåĹűçŽĎæŰĞæIJňäÿšïijŇ åŕŕäžěåĹĘåĹńä¡£çŤĺ `bin()`,`oct()` æĹŰ `hex()` åĞ¡æŢřïijŽ

```
>>> x = 1234
>>> bin(x)
'0b10011010010'
>>> oct(x)
'0o2322'
>>> hex(x)
'0x4d2'
>>>
```

åŕęåd'ŰïijŇåęĆæđIJä¡ăäÿ■æČşè¿ŞåĞž `0b`, `0o` æĹŰèĂĚ `0x` çŽĎåL'■çijĂçŽĎèŕĬïijŇåŕŕäžěä¡£çŤĺ `format()` åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> format(x, 'b')
'10011010010'
>>> format(x, 'o')
'2322'
>>> format(x, 'x')
'4d2'
>>>
```

æŢť'æŢřæŸŕæIJL'çňęåŕůçŽĎïijŇæL'Ăäžěåę Ćæđ IJä¡ăåIJĺåd'ĎçŘĘèť§æŢřçŽĎèŕĬïijŇè¿ŞåĞžçżŞæđIJäij

```
>>> x = -1234
>>> format(x, 'b')
'-10011010010'
>>> format(x, 'x')
'-4d2'
>>>
```

åęĆæđIJä¡ăæČşäžğçŤ§äÿĂäÿłæŮăçňęåŕůåĂijïijŇä¡ăéIJĂèęÀåćđåŁääÿĂäÿłæŇĞçd'žæIJÅåd'ğä¡■éŢ£åž

```
>>> x = -1234
>>> format(2**32 + x, 'b')
'11111111111111111111101100101110'
>>> format(2**32 + x, 'x')
'fffffb2e'
>>>
```

äÿžäžĘäžěäÿ■åŘŇçŽĎè£ŻåĹűè¡ňæ■ćæŢťæŢřå■ŮçňęäÿšïijŇçőĂå■ŢçŽĎä¡£çŤĺåÿęæIJL'è£ŻåĹűçŽĎ int() åĞ¡æŢřå■şåŕŕïijŽ

```
>>> int('4d2', 16)
1234
>>> int('10011010010', 2)
1234
>>>
```

èőĺèőž

åď'ğåď'ŽæŢřæČĚåĘţäÿŃåď'ĎçŘĘäžŇè£ŻåĹűãĂĄåĚńè£ŻåĹűåŠŇå■ĄåĚ■è£ŻåĹűæŢť'æŢřæŸŕå¿ĹçőĂår åŘłèęĄèőřä¡Ŕè£ŹäžŻè¡ňæ■ćåśđäžŐæŢť'æŢřåŠŇåĚűåŕźåžŢçŽĎæŰĞæIJňèąĺçď'žäźŃéŮť'çŽĎè¡ňæ■ćå■şåŕŕã

æIJĂåŘŐïijŇä¡£çŤĺåĚńè£ŻåĹűçŽĎçĺŃåžŔåŚŸæIJL'äÿĂçĆźéIJĂèęĄæşĺæĎŔäÿŃãĂĆ PythonæŇĞåőŽåĚńè£ŻåĹűæŢřçŽĎèŕ■æşŢèů§åĚűäžŰèŕ■èĺĂçĺ■æIJL'äÿ■åŘŇãĂĆæŕŤåęĆïijŇåęĆæđIJä¡ääČ

```
>>> import os
>>> os.chmod('script.py', 0755)
    File "<stdin>", line 1
        os.chmod('script.py', 0755)
                              ^
SyntaxError: invalid token
>>>
```

éIJĂçąőä£İåĚńè£ŻåĹűæŢřçŽĎåL'■çijĂæŸŕ 0o ïijŇårśåČŔäÿŃéİćè£ŹæăůïijŽ

```
>>> os.chmod('script.py', 0o755)
>>>
```

3.5 å■ŮèŁĆåĹřåď'ğæŢť'æŢřçŽĎæL'ŞåŇĚäÿŐèğčåŇĚ

éŮőécŸ

ä¡ääæIJL'äÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšåźűæČşårĘåőČèğčåŐŃæĹŔäÿĂäÿłæŢť'æŢřãĂĆæĹŰèĂĚïijŇä¡äéIJĂ

èğčåĘşæŰźæąĹ

åĄĞèő¿ä¡äçŽĎçĺŃåžŔéIJĂèęĄåď'ĎçŘĘäÿĂäÿłæŃěæIJL'128ä¡■éŢ£çŽĎ16äÿłåĚČçt'äçŽĎå■ŮèŁĆå■Ůç

```
data = b'\x00\x124V\x00x\x90\xab\x00\xcd\xef\x01\x00#\x004'
```

äÿžäžĘåřĘbytesèğčæđŘäÿžæŢť'æŢřïijŇä¡£çŤĺ int.from_bytes() æŰźæşŢïijŇåźűåČŔäÿŃéİćè£ŹæăůæŇĞåőŽå■ŮèŁĆéąžåžŔïijŽ

```
>>> len(data)
16
>>> int.from_bytes(data, 'little')
69120565665751139577663547927094891008
>>> int.from_bytes(data, 'big')
94522842520747284487117727783387188
>>>
```

äÿžäžĘåřĘäÿĂäÿłåď˙ğæŢť˙æŢřè¡ňæ■ćäÿžäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšïijŇä¡£çŤÍ `int.to_bytes()` æŰźæşŢïijŇåźűåČŔäÿŃéÍćè£ŹæăůæŇĞåőŽå■ŮèŁĆæŢřåŠŇå■ŮèŁĆéąžåžŔïijŽ

```
>>> x = 94522842520747284487117727783387188
>>> x.to_bytes(16, 'big')
b'\x00\x124V\x00x\x90\xab\x00\xcd\xef\x01\x00#\x004'
>>> x.to_bytes(16, 'little')
b'4\x00#\x00\x01\xef\xcd\x00\xab\x90x\x00V4\x12\x00'
>>>
```

èőÍèőž

åď˙ğæŢť˙æŢřåŠŇå■ŮèŁĆå■Ůçňęäÿšäź˙ŃéŮť˙çŽĎè¡ňæ■ćæŞ■ä¡IJåźűäÿ■åÿÿèğĄãĂĆ çĎűèĂŇïijŇåIJÍäÿĂäžŽåžŤçŤÍéćĘå§§æIJL˙æŮűåĂŹäź§äijŽåĞžçŐřïijŇæŕŤåęĆářĘçăĄå■ęæŁŰèĂĚç¡ŚçżIJã ä¿ŃåęĆïijŇIPv6ç¡ŚçżIJåIJřåİĂä¡£çŤÍäÿĂäÿł128ä¡■çŽĎæŢť˙æŢřèąÍçď˙žãĂĆ åęĆæđIJä¡äèęĄäžŐäÿĂäÿłæŢřæ■őèőřå¡Ţäÿ■æŔŘåŔŰè£ŹæăůçŽĎåĂijçŽĎæŮűåĂŹïijŇä¡ăářšäijŽéÍćářźè£Ź

ä¡IJäÿžäÿĂçğ■æŻ£äžčæŰźæąĹïijŇä¡ăåŔřèČ¡æČşä¡£çŤÍ6.11ařŔèŁĆäÿ■æĽĂäżŃçż■çŽĎ `struct` æÍąåİŮæİěèğčåŐŃå■ŮèŁĆãĂĆ è£ŹæăůäźšèąŇå¿ŮéĂŽïijŇäÿ■è£ĞåĹľçŤÍ `struct` æÍąåİŮæİěèğčåŐŃářźäžŐæŢť˙æŢřçŽĎåď˙ğářŔæŸřæIJL˙éŹŘåĹűçŽĎãĂĆ åŻăæ■ď˙ïijŇä¡ăåŔřèČ¡æČşèğčåŐŃåď˙Žäÿłå■ŮèŁĆäÿšåźűåřĘçżŞæđIJåŘĹåźűäÿžæIJĂçżĹçŽĎçżŞæđIJïijŇářš

```
>>> data
b'\x00\x124V\x00x\x90\xab\x00\xcd\xef\x01\x00#\x004'
>>> import struct
>>> hi, lo = struct.unpack('>QQ', data)
>>> (hi << 64) + lo
94522842520747284487117727783387188
>>>
```

å■ŮèŁĆéąžåžŔèğĎåĹŹ(littleæĹŰbig)äżĚäżĚæŇĞåőŽäžĘæđĎåżžæŢť˙æŢřæŮűçŽĎå■ŮèŁĆçŽĎä¡Őä¡■ æĹŚäžñäžŐäÿŃéÍćçš¿å£ČæđĎéĂăçŽĎ16è£ŻåĹűæŢřçŽĎèąÍçď˙žäÿ■åŔřäžěå¿Ĺåőźæ˙ŸŞçŽĎçIJŃåĞžæÍěïijŽ

```
>>> x = 0x01020304
>>> x.to_bytes(4, 'big')
b'\x01\x02\x03\x04'
>>> x.to_bytes(4, 'little')
b'\x04\x03\x02\x01'
>>>
```

åęĆæđIJä¡ăèŕŢçİĂåřĘäÿĂäÿłæŢťæŢřæĽŞåŇĚäÿžå■ŮèŁĆå■ŮçňęäÿšïijŇéĆčäźĹåőČåřśäÿ■åŘĹéĂĆäžE
åęĆæđIJéIJĂèęAçŽĎèŕÍïijŇä¡ååŔŕäżěä¡£çŤÍ `int.bit_length()`
æŰźæşŢæİěåĘşåőŽéIJĂèęAåď˙ŽåřŚå■ŮèŁĆä¡■æİěå■ŸåĆĺè£ŹäÿłåĂijãĂĆ

```
>>> x = 523 ** 23
>>> x
335381300113661875107536852714019056160355655333978849017944067
>>> x.to_bytes(16, 'little')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
OverflowError: int too big to convert
>>> x.bit_length()
208
>>> nbytes, rem = divmod(x.bit_length(), 8)
>>> if rem:
... nbytes += 1
...
>>>
>>> x.to_bytes(nbytes, 'little')
b'\x03X\xf1\x82iT\x96\xac\xc7c\x16\xf3\xb9\xcf...\xd0'
>>>
```

3.6 åď■æŢřçŽĎæŢřå■ęè£ŘçőŮ

éŮőécŸ

ä¡ååĘŹçŽĎæIJĂæŰřçŽĎç¡ŚçżIJèőď˙èŕĄæŰźæąĹäžčçăĄéĄĞåĹřäžĘäÿĂäÿłéŽ¿éćŸïijŇåźűäÿŤä¡ååŤřäÿ
åĘ■æĹŰèĂĚæŸŕä¡ääżĚäżĚéIJĂèęAä¡£çŤĺåď■æŢřæİěæĽ˘ğèąŇäÿĂäžŻèőąçőŮæŞ■ä¡IJãĂĆ

èğčåĘşæŰźæąĹ

åď■æŢřåŔŕäżěçŤĺä¡£çŤĺåĞ¡æŢř `complex(real, imag)`
æĹŰèĂĚæŸŕåÿęæIJĽåŘŐçijĂjçŽĎæţőçĆźæŢřæİěæŇĞåőŽãĂĆæŕŤåęĆïijŽ

```
>>> a = complex(2, 4)
>>> b = 3 - 5j
>>> a
(2+4j)
>>> b
(3-5j)
>>>
```

åŕźåžŤçŽĎåőđéČĺãĂąèŹŽéČĺåŠŇåĚśè¡■åď■æŢřåŔŕäżěå¿Ĺåőźæ˙ŸŞçŽĎèŐůåŘŰãĂĆåŕśåČŔäÿŇéİćè£Ź

```
>>> a.real
2.0
```

```
>>> a.imag
4.0
>>> a.conjugate()
(2-4j)
>>>
```

åŘęåď ŰïijŇæĽ ĂæĲĽåÿÿèğĄçŽĎæŢřå■ęè£Řçő ŮéČ¡åŔŕäżĕåůĕä¡ĲïijŽ

```
>>> a + b
(5-1j)
>>> a * b
(26+2j)
>>> a / b
(-0.4117647058823529+0.6470588235294118j)
>>> abs(a)
4.47213595499958
>>>
```

åęĆæđĲèęĄæĽ ğèąŇåĚűäżŰçŽĎåď■æŢřåĞ¡æŢřæŕŤåęĆæ■čåijęãĂĄä¡ŹåijęæĹŰåźşæŰźæăźïijŇä¡£çŤĺ `cmath` æĺąåİŮïijŽ

```
>>> import cmath
>>> cmath.sin(a)
(24.83130584894638-11.356612711218174j)
>>> cmath.cos(a)
(-11.36423470640106-24.814651485634187j)
>>> cmath.exp(a)
(-4.829809383269385-5.5920560936409816j)
>>>
```

èőĺèőž

Pythonäÿ■åď ğéČĺåĹĘäÿŐæŢřå■ęçŻÿåĚşçŽĎæĺąåİŮéČ¡èČ¡åď ĎçŘĘåď■æŢřãĂĆ æŕŤåęĆåęĆæđĲä¡ăä¡£çŤĺ `numpy` ïijŇåŔŕäżĕå¿Ĺåőźæ ŸŞçŽĎæđĎéĂăäÿĂäÿłåď■æŢřæŢřçżĎåźűåĲĺèž£Źäÿłæ

```
>>> import numpy as np
>>> a = np.array([2+3j, 4+5j, 6-7j, 8+9j])
>>> a
array([ 2.+3.j,  4.+5.j,  6.-7.j,  8.+9.j])
>>> a + 2
array([  4.+3.j,   6.+5.j,   8.-7.j,  10.+9.j])
>>> np.sin(a)
array([    9.15449915  -4.16890696j,   -56.16227422 -48.50245524j,
        -153.20827755-526.47684926j,  4008.42651446-589.49948373j])
>>>
```

PythonçŽĎæăĞåĞĘæŢřå■ęåĞ¡æŢřçąőåőđæČĚåĘţäÿŃåźűäÿ■èČ¡äžğçŤ§åď■æŢřåĂijïijŇåŽæ■ď ä¡ăçŽ

```
>>> import math
>>> math.sqrt(-1)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ValueError: math domain error
>>>
```

åęĆæđIJä¡ăæČşçŢ§æĹŘäÿĂäÿłåd'■æŢřè£ŤåŻđçżŞæđIJïijŇä¡ăå£Ěéążæ Ÿ¿çd'žçŽĎä¡£çŤĺ cmath æĺąåİŮïijŇæĹŰèĂĚåIJĺæ§Řäÿłæ ŤŕæŇĄåd'■æŢřçŽĎåžŞäÿ■åčřæ ŸŐåd'■æŢřçśžåđŃçŽĎä¡£çŤĺãĂĆæ

```
>>> import cmath
>>> cmath.sqrt(-1)
1j
>>>
```

3.7 æŮăçľůåd'ğäÿŐNaN

éŮőécŸ

ä¡ăæČşåĹŻåžžæĹŰæţŃèŕŢæ■čæŮăçľůãĂĄèt'§æŮăçľůæĹŰNaN(éİđæŢřå■Ů)çŽĎæţőçĆźæŢřãĂĆ

èğčåĘşæŰźæąĹ

PythonåźűæšąæIJL'çL'źæőŁçŽĎèŕ■æşŢæİěèąĺçd'žè£ŹäžŻçL'źæőŁçŽĎæţőçĆźåĂijïijŇä¡ĘæŸŕåŔŕäžěä¡£ float() æİěåĹŻåžžåőČäžñãĂĆæŕŤåęĆïijŽ

```
>>> a = float('inf')
>>> b = float('-inf')
>>> c = float('nan')
>>> a
inf
>>> b
-inf
>>> c
nan
>>>
```

äÿžäžĘæţŃèŕŢè£ŹäžŻåĂijçŽĎå■ŸåIJĺïijŇä¡£çŤĺ math.isinf() åŠŇ math.isnan() åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> math.isinf(a)
True
>>> math.isnan(c)
True
>>>
```

èőĺèőž

æČşäžĘèğčæŻťˇåďŽè£ŹäžŻçĽźæőŁæţőçĆźåĂijçŽĎä£ąæĄŕïijŇåŕŕäžěåŔĆèĂČIEEE 754èğĎèŇČãĂĆ çĎűèĂŇïijŇäź§æIJĽäÿĂäžŻåIJřæŰźéIJĂèęĄä¡ăçĽźåĹńæşĺæĎŕïijŇçĽźåĹńæŸŕèů§æŕŤè¿

æŮăçľůåďğæŢřåIJĺæĽğèąŇæŢřå■ęèőąçőŮçŽĎæŮűåĂŹäijŽäijăæŠ■ïijŇærŤåęĆïijŽ

```
>>> a = float('inf')
>>> a + 45
inf
>>> a * 10
inf
>>> 10 / a
0.0
>>>
```

ä¡ĘæŸŕæIJĽäžŻæŞ■ä¡IJæŮűæIJłåőŽäźĽçŽĎåźűäijŽè£ŤåŻđäÿĂäÿłNaNçżŞæđIJãĂĆærŤåęĆïijŽ

```
>>> a = float('inf')
>>> a/a
nan
>>> b = float('-inf')
>>> a + b
nan
>>>
```

NaNåĂijäijŽåIJĺæĽĂæIJĽæŞ■ä¡IJäÿ■äijăæŠ■ïijŇèĂŇäÿ■äijŽäžğçŤ§åijĆåÿÿãĂĆærŤåęĆïijŽ

```
>>> c = float('nan')
>>> c + 23
nan
>>> c / 2
nan
>>> c * 2
nan
>>> math.sqrt(c)
nan
>>>
```

NaNåĂijçŽĎäÿĂäÿłçĽźåĹńçŽĎåIJřæŰźæŮűåőČäžňäźŃéŮťçŽĎærŤè¿ČæŞ■ä¡IJæĂżæŸŕè£ŤåŻđFalse

```
>>> c = float('nan')
>>> d = float('nan')
>>> c == d
False
>>> c is d
False
>>>
```

çŤśäžŐè£ŹäÿłåŐ§åŻäïijŇætŃèŕŢäÿĂäÿłNaNåĂijå¿ŮåŤŕäÿĂåőĽåĚĺçŽĎæŰźæşŢåřśæŸŕä¡£çŤĺ `math.isnan()` ïijŇäź§åřśæŸŕäÿŁéİćæijŤçd’žçŽĎéĆčæăůãĂĆ

æIJĽæŮűåĂŹçÍŃåžŔåŚŸæČşæŤźåŔŸPythonézŸèőď'èąŇäÿžïijŇåIJÍè£ŤåŻđæŮăçľ'ůåď'ğæĹŰNaNçżŞæ
`fpectl` æÍąåİŮåŘŕäžěçŤÍæİěæŤźåŔŸè£Źçğ■èąŇäÿžïijŇä¡ĘæŸŕåőČåIJÍæăĞåĞĘçŽĎPythonædĎåžžäÿ■åźűå
åźűäÿŤéŠĹåŕźçŽĎæŸŕäÿŞåőűçžğçÍŃåžŔåŚŸãĂĆåŘŕäžěåŔĆèĂČåIJÍçž£çŽĎPythonæŰĞæąčèŐůåŘŰæŻť'åd

3.8 åĹĘæŢřè£ŘçőŮ

éŮőécŸ

ä¡ăè£ŻåĚěæŮűéŮť'æIJžåŹĺïijŇçłĄçĎűåŔŚçŐřä¡ăæ■čåIJÍåĄŽåřŔå■ęåőűåž■ä¡IJäÿŽïijŇåźűæűĽ'åŔŁåĹřă
æĹŰèĂĚä¡ăåŘŕèČ¡éIJĂèęĄåĘŹäžčçăĄåŐžèőąçőŮåIJÍä¡ăçŽĎæIJÍåůěåůěåŐĆäÿ■çŽĎætŃéĞŔåĂijãĂĆ

èğčåĘşæŰźæąĹ

`fractions` æÍąåİŮåŘŕäžěèćńçŤÍæİěæĽ'ğèąŇåŇĚåŘńåĹĘæŢřçŽĎæŢřå■ęè£ŘçőŮãĂĆærŤåęĆïijŽ

```
>>> from fractions import Fraction
>>> a = Fraction(5, 4)
>>> b = Fraction(7, 16)
>>> print(a + b)
27/16
>>> print(a * b)
35/64

>>> # Getting numerator/denominator
>>> c = a * b
>>> c.numerator
35
>>> c.denominator
64

>>> # Converting to a float
>>> float(c)
0.546875

>>> # Limiting the denominator of a value
>>> print(c.limit_denominator(8))
4/7

>>> # Converting a float to a fraction
>>> x = 3.75
>>> y = Fraction(*x.as_integer_ratio())
>>> y
Fraction(15, 4)
>>>
```

èőĺèőž

åIJĺåď˙gåď˙ŽæŢřçĺŃåžŕäÿ■äÿĂèĹňäÿ■äijŽåĞžçŐřåĹĘæŢřçŽĎèőąçőŮéŮőécŸïijŇä¡ĘæŸŕæIJL˙æŮűåĂŽ
ærŤåęĆïijŇåIJĺäÿĂäÿłåĚĄèőÿæŐěåŔŮåĹĘæŢřå¡ćåijŔçŽĎætŃèŕŢå■Ţä¡■åźűäžěåĹĘæŢřå¡ćåijŔæĽ˙gèąŇè£Ř
çŻť˙æŐěä¡£çŤĺåĹĘæŢřåŔŕäžěåĞŔåřŚæĽ˙ŃåŁĺè¡ňæ■ćäÿžåřŔæŢřæĹŰæţőçĆźæŢřçŽĎåůěä¡IJãĂĆ

3.9 åď˙gådŃæŢřçżĎè£ŘçőŮ

éŮőécŸ

ä¡ăéIJĂèęĄåIJĺåď˙gæŢřæ■őéŻĘ(ærŤåęĆæŢřçżĎæĹŰç¡Śæăij)äÿŁéİćæĽ˙gèąŇèőąçőŮãĂĆ

èğčåĘşæŰźæąĹ

æűĽ˙åŔŁåĹřæŢřçżĎçŽĎéĞ■éĞŔçžğè£ŘçőŮæŞ■ä¡IJïijŇåŔŕäžěä¡£çŤĺ `NumPy` åžŞãĂĆ
`NumPy` çŽĎäÿĂäÿłäÿżèęĄçĽźå¿ĄæŸŕåőĆäijŽçżŹPythonæŔŘä¿ŻäÿĂäÿłæŢřçżĎåŕźèśąïijŇçŻÿærŤæăĞåĞĘ
äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎåřŔä¿Ńå■ŘïijŇåŘŚä¡ăåśŢçd˙žæăĞåĞĘåĹŮèąĺåŕźèśąåŠŇ
`NumPy` æŢřçżĎåŕźèśąäźŃéŮt˙çŽĎåůőåĹńïijŽ

```
>>> # Python lists
>>> x = [1, 2, 3, 4]
>>> y = [5, 6, 7, 8]
>>> x * 2
[1, 2, 3, 4, 1, 2, 3, 4]
>>> x + 10
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: can only concatenate list (not "int") to list
>>> x + y
[1, 2, 3, 4, 5, 6, 7, 8]

>>> # Numpy arrays
>>> import numpy as np
>>> ax = np.array([1, 2, 3, 4])
>>> ay = np.array([5, 6, 7, 8])
>>> ax * 2
array([2, 4, 6, 8])
>>> ax + 10
array([11, 12, 13, 14])
>>> ax + ay
array([ 6, 8, 10, 12])
>>> ax * ay
array([ 5, 12, 21, 32])
>>>
```

æ■čåęĆæĽĂèğĄïijŇäÿď'çğ■æŰźæąĹäÿ■æŢřçżĎçŽĎå§žæIJňæŢřå■ęè£ŘçőŮçżŞæđIJåźűäÿ■çŻÿåŘŇãĂ
çĽźåĹńçŽĎïijŇ NumPy äÿ■çŽĎæăĞéĞŔè£ŘçőŮ(æŕŤåęĆ ax
* 2 æĹŰ ax + 10)äijŽä¡IJçŤĺåIJĺæŕŔäÿĂäÿłåĚČçt'ăäÿŁãĂĆ
åŔęåď'ŰïijŇå¡Şäÿď'äÿłæŞ■ä¡IJæŢřéČ¡æŸŕæŢřçżĎçŽĎæŮűåĂŹæĽ'ğèąŇåĚČçt'ăåŕźç■Ľ'ä¡■ç¡őèőąçőŮïijŇåźű
åŕźæŢt'äÿłæŢřçżĎäÿ■æĽ'ĂæIJĽ'åĚČçt'ăåŘŇæŮűæĽ'ğèąŇæŢřå■ęè£ŘçőŮåŔŕäžěä¡£å¿Ůä¡IJçŤĺåIJĺæŢt'ă
æŕŤåęĆïijŇåęĆædIJä¡ăæČşèőąçőŮåď'ŽéąźåijŕçŽĎåĂijïijŇåŔŕäžěè£ŹæăůåĄŽïijŽ

```
>>> def f(x):
... return 3*x**2 - 2*x + 7
...
>>> f(ax)
array([ 8, 15, 28, 47])
>>>
```

NumPy è£ŸäÿžæŢřçżĎæŞ■ä¡IJæŔŘä¿ŻäžEåď'ğéĞŕçŽĎéĂŽçŤĺåĞ¡æŢřïijŇè£ŹäžŻåĞ¡æŢřåŕŕäžěä¡IJäÿ
math æĺąåİŮäÿ■çśžäijijjåĞ¡æŢřçŽĎæŻ£äžčãĂĆæŕŤåęĆïijŽ

```
>>> np.sqrt(ax)
array([ 1. , 1.41421356, 1.73205081, 2. ])
>>> np.cos(ax)
array([ 0.54030231, -0.41614684, -0.9899925 , -0.65364362])
>>>
```

ä¡£çŤĺè£ŹäžŻéĂŽçŤĺåĞ¡æŢřèęĄæŕŤå¿łçŐŕæŢřçżĎåźűä¡£çŤĺ
math æĺąåİŮäÿ■çŽĎåĞ¡æŢřæĽ'ğèąŇèőąçőŮèęĄå£ńçŽĎåď'ŽãĂĆ
åŻäæ■d'ïijŇåŔłèęĄæIJĽ'åŔŕèČ¡çŽĎèŕĺåř¡éĞŔéĂĽ'æŃl' NumPy çŽĎæŢřçżĎæŰźæąĹãĂĆ

åžŢåśĆåőđçŐřäÿ■ïijŇ NumPy æŢřçżĎä¡£çŤĺäžĘCæĹŰèĂĚFortranèŕ■èĺĂçŽĎæIJžåĹűåĹĘéĚ■åĘĚå■Ÿã
äźş§åřśæŸŕèŕťïijŇåőČäžňæŸŕäÿĂäÿłéİđåÿÿåď'ğçŽĎè£đçż■çŽĎåźűçŤśåŘŇçśžåđŇæŢřæ■őçżĎæĹŘçŽĎåĘĚå
æĽ'ĂäžěïijŇä¡ăåŔŕäžěæđĎéĂääÿĂäÿłæŕŤæŹőéĂŽPythonåĹŮèąĺåď'ğçŽĎåď'ŽçŽĎæŢřçżĎãĂĆ
æŕŤåęĆïijŇåęĆædIJä¡ăæČşæđĎéĂääÿĂäÿł10,000*10,000çŽĎæţőçĆźæŢřäžŇçžt'ç¡ŚæäijïijŇå¿Ĺè¡żæİ¿ïijŽ

```
>>> grid = np.zeros(shape=(10000,10000), dtype=float)
>>> grid
    array([[ 0., 0., 0., ..., 0., 0., 0.],
    [ 0., 0., 0., ..., 0., 0., 0.],
    [ 0., 0., 0., ..., 0., 0., 0.],
    ...,
    [ 0., 0., 0., ..., 0., 0., 0.],
    [ 0., 0., 0., ..., 0., 0., 0.],
    [ 0., 0., 0., ..., 0., 0., 0.]])
>>>
```

æĽ'ĂæIJĽçŽĎæŹőéĂŽæŞ■ä¡IJè£ŸæŸŕäijŽåŘŇæŮűä¡IJçŤĺåIJĺæĽ'ĂæIJĽ'åĚČçt'ăäÿŁïijŽ

```
>>> grid += 10
>>> grid
array([[ 10., 10., 10., ..., 10., 10., 10.],
    [ 10., 10., 10., ..., 10., 10., 10.],
    [ 10., 10., 10., ..., 10., 10., 10.],
    ...,
```

```
    [ 10., 10., 10., ..., 10., 10., 10.],
    [ 10., 10., 10., ..., 10., 10., 10.],
    [ 10., 10., 10., ..., 10., 10., 10.]])
>>> np.sin(grid)
array([[-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    ...,
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111]])
>>>
```

åĚşäžŐ NumPy æIJL'äÿĂçĆźéIJĂèęĄçL'źåĹńçŽĎäÿżæĎŔïijŇéĆčåřśæŸŕåőČæL'ľåśŢPythonåĹŮèąĺçŽĎ - çL'źåĹńæŸŕåŕźäžŐåď'Žçžť'æŢřçżĎãĂĆ äÿžäžĘèŕť'æŸŐæÿĚæěŽïijŇåĚĹæđĎéĂääÿĂäÿłçőĂå■ŢçŽĎäžŇçżt

```
>>> a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
>>> a
array([[ 1, 2, 3, 4],
[ 5, 6, 7, 8],
[ 9, 10, 11, 12]])

>>> # Select row 1
>>> a[1]
array([5, 6, 7, 8])

>>> # Select column 1
>>> a[:,1]
array([ 2, 6, 10])

>>> # Select a subregion and change it
>>> a[1:3, 1:3]
array([[ 6, 7],
        [10, 11]])
>>> a[1:3, 1:3] += 10
>>> a
array([[ 1, 2, 3, 4],
        [ 5, 16, 17, 8],
        [ 9, 20, 21, 12]])

>>> # Broadcast a row vector across an operation on all rows
>>> a + [100, 101, 102, 103]
array([[101, 103, 105, 107],
        [105, 117, 119, 111],
```

```
        [109, 121, 123, 115]])
>>> a
array([[ 1, 2, 3, 4],
        [ 5, 16, 17, 8],
        [ 9, 20, 21, 12]])

>>> # Conditional assignment on an array
>>> np.where(a < 10, a, 10)
array([[ 1, 2, 3, 4],
        [ 5, 10, 10, 8],
        [ 9, 10, 10, 10]])
>>>
```

èőÍèőž

NumPy æŸŕPythonécĘå§§äÿ■å¿Ĺåď'ŽçğŚå■ęäÿŐåůęÍŃåžŞçŽĎå§žçąĂïijŇåŘŇæŮűäź§æŸŕèćńäź£æşŽ
å■şä¿£åęĆæ■d'ïijŇåIJĺåĹŽåijĂåğŃçŽĎæŮűåĂŹéĂŽè£ĞäÿĂäžŽçőĂå■ŢçŽĎä¿Ńå■ŘåŠŇçŐl'åĚůçÍŃåžŔäź§

éĂŽåÿÿæĹŚäžñåŕijåĚě NumPy æĺąåİŮçŽĎæŮűåĂŹäijŽä¡£çŤÍèŕ■åŔě import numpy as np ãĂĆ è£ŹæăůçŽĎèŕÍä¡ăåřšäÿ■çŤĺåĘ■ä¡ăçŽĎçÍŃåžŔéĞŇéİćäÿĂéĄ■éĄ■çŽĎæŢšåĚě numpy ïijŇåŘłéIJĂèęÀè¿ŞåĚě np åřšèąŇäžĘïijŇèŁĆçIJĄäžĘäÿ■åřŚæŮűéŮť'ãĂĆ

åęĆæđIJæČşèŐůåŔŰæŽť'åď'ŽçŽĎä£ąæĄŕïijŇä¡ăå¡ŞçĎűå¿ŮåŐż NumPy åőŸç¡ŚéĂŽéĂŽäžĘïijŇç¡ŚåİĂæŸŕïijŽ http://www.numpy.org

3.10 ç§l'éŸţäÿŐçž£æĂğäžčæŢřè£ŘçőŮ

éŮőécŸ

ä¡ăéIJĂèęĄæL'ğèąŇç§l'éŸţåŠŇçž£æĂğäžčæŢřè£ŘçőŮïijŇærŤåęĆç§l'éŸţäźŸæşŢãĂĄåŕžæL'¿èąŇåĹŮå

èğčåĘşæŰźæąĹ

NumPy åžŞæIJL'äÿĂäÿłç§l'éŸţåŕźèśąåŘŕäžëçŤĺæİèèğčåĘşè£ŹäÿłéŮőécŸãĂĆ

ç§l'éŸţçśžäijijäžŐ3.9åřŔèŁĆäÿ■æŢřçżĎåŕźèśąïijŇä¡ĘæŸŕéĄţå¿łçž£æĂğäžčæŢřçŽĎèőąçőŮèğĎåĹŹãĂ

```
>>> import numpy as np
>>> m = np.matrix([[1,-2,3],[0,4,5],[7,8,-9]])
>>> m
matrix([[ 1, -2, 3],
        [ 0, 4, 5],
        [ 7, 8, -9]])

>>> # Return transpose
```

```
>>> m.T
matrix([[ 1,  0,  7],
        [-2,  4,  8],
        [ 3,  5, -9]])

>>> # Return inverse
>>> m.I
matrix([[ 0.33043478, -0.02608696,  0.09565217],
        [-0.15217391,  0.13043478,  0.02173913],
        [ 0.12173913,  0.09565217, -0.0173913 ]])

>>> # Create a vector and multiply
>>> v = np.matrix([[2],[3],[4]])
>>> v
matrix([[2],
        [3],
        [4]])
>>> m * v
matrix([[ 8],
        [32],
        [ 2]])
>>>
```

åŕŕäžěåIJĺ `numpy.linalg` å■ŘåŇĚäÿ■æĽ¿åĹřæŻťďåďŽçŽĎæŞ■ä¡IJåĞ¡æŢřïijŇæŕŤåęĆïijŽ

```
>>> import numpy.linalg

>>> # Determinant
>>> numpy.linalg.det(m)
-229.99999999999983

>>> # Eigenvalues
>>> numpy.linalg.eigvals(m)
array([-13.11474312,  2.75956154,  6.35518158])

>>> # Solve for x in mx = v
>>> x = numpy.linalg.solve(m, v)
>>> x
matrix([[ 0.96521739],
        [ 0.17391304],
        [ 0.46086957]])
>>> m * x
matrix([[ 2.],
        [ 3.],
        [ 4.]])
>>> v
matrix([[2],
        [3],
        [4]])
>>>
```

èőÍèőž

å¿ĹæŸ¿çĎűçž£æĂğäžčæŢřæŸŕäÿłéÍđåÿÿåď˘ğçŽĎäÿžécŸïijŇåůšçżŔèűĚåĞžäžĘæIJňäźęèČ¡èőÍèőžçŽĎè
ä¡ĘæŸŕïijŇåęĆæđIJä¡ăéIJĂèęĄæŞ■ä¡IJæŢřçżĎåŠŇåŘŚéĞŔçŽĎèŕÍïijŇ NumPy
æŸŕäÿĂäÿłäÿ■éŤŹçŽĎåĚěåŔčçĆźãĂĆ åŔŕäžěèőŁéŮő NumPy åőŸç¡Ś http://www.numpy.org
èŐůåŘŰæŻť åď˘Žä£ąæĄŕãĂĆ

3.11 éŽŔælJžéĂĽæŃľ

éŮőécŸ

ä¡ăæČşäżŐäÿĂäÿłåžŔåĹŮäÿ■éŽŔæIJžæŁ¡åŘŰèŃěźšåĚČçť˘ăïijŇæĹŰèĂĚæČşçŢ§æĹŘåĞääÿłéŽŔæIJ

èğčåĘşæŰźæąĹ

random æĺąåİŮæIJL˘åď˘ğéĞŔçŽĎåĞ¡æŢřçŤĺæİěäžğçŢ§éŽŔæIJžæŢřåŠŇéŽŔæIJžéĂĽæŃľ˘åĚČçť˘ããĂĆ
ærŤåęĆïijŇèęĄæČşäżŐäÿĂäÿłåžŔåĹŮäÿ■éŽŔæIJžçŽĎæŁ¡åŘŰäÿĂäÿłåĚČçť˘ăïijŇåŕŕäžěä¡£çŤĺ
random.choice() ïijŽ

```
>>> import random
>>> values = [1, 2, 3, 4, 5, 6]
>>> random.choice(values)
2
>>> random.choice(values)
3
>>> random.choice(values)
1
>>> random.choice(values)
4
>>> random.choice(values)
6
>>>
```

äÿžäžĘæŔŘåŘŰåĞžNäÿłäÿ■åŘŇåĚČçť˘äçŽĎæăůæIJňçŤĺæİěåĄŽè£ŻäÿĂæ■ěçŽĎæŞ■ä¡IJïijŇåŕŕäžěä¡£
random.sample() ïijŽ

```
>>> random.sample(values, 2)
[6, 2]
>>> random.sample(values, 2)
[4, 3]
>>> random.sample(values, 3)
[4, 3, 1]
>>> random.sample(values, 3)
[5, 4, 1]
>>>
```

åęĆæđIJä¡ăäžĚäžĚåŕłæŸŕæČşæĽŞäźśåžŔåĹŮäÿ■åĚČçť˘ăçŽĎéąžåžŔïijŇåŔŕäžěä¡£çŤÍ `random.shuffle()` ïijŽ

```
>>> random.shuffle(values)
>>> values
[2, 4, 6, 5, 3, 1]
>>> random.shuffle(values)
>>> values
[3, 5, 2, 1, 6, 4]
>>>
```

çŤ§æĹŘéŽŔæIJžæŢť˘æŢřïijŇèŕůä¡£çŤÍ `random.randint()` ïijŽ

```
>>> random.randint(0,10)
2
>>> random.randint(0,10)
5
>>> random.randint(0,10)
0
>>> random.randint(0,10)
7
>>> random.randint(0,10)
10
>>> random.randint(0,10)
3
>>>
```

äÿžäžĘçŤ§æĹŘ0åĹř1èŇČåŻť˘åĘĚåİĞåŇĂåĹĘåÿČçŽĎæţőçĆźæŢřïijŇä¡£çŤÍ `random.random()` ïijŽ

```
>>> random.random()
0.9406677561675867
>>> random.random()
0.133129581343897
>>> random.random()
0.4144991136919316
>>>
```

åęĆæđIJèęĄèŐůåŔŰNä¡■éŽŔæIJžä¡■(äžŇè£ŻåĹű)çŽĎæŢť˘æŢřïijŇä¡£çŤÍ `random.getrandbits()` ïijŽ

```
>>> random.getrandbits(200)
335837000776573622800628485064121869519521710558559406913275
>>>
```

èőÍèőž

`random` æÍąåİŮä¡£çŤÍ *Mersenne Twister* çőŮæşŢæİěèőąçőŮçŤ§æĹŘéŽŔæIJžæŢřãĂĆè£ŹæŸŕäÿĂäÿłçää¡ĘæŸŕä¡ăåŔŕäžěéĂŽè£Ğ `random.seed()` åĞ¡æŢřä£őæŤźåĹİåğŃåŇŰçğ■å■ŘãĂĆærŤåęĆïijŽ

```
random.seed() # Seed based on system time or os.urandom()
random.seed(12345) # Seed based on integer given
random.seed(b'bytedata') # Seed based on byte data
```

éŹď'äžĘäÿŁè£řäżŃçż■çŽĎåŁ§èČ¡ïijŇrandomæĺąåİŮè£ŸåŇĚåŘńå§žäžŐåİĞåŇĂåĹĘåÿČãĂĄéńŸæŰŕåæŕŤåęĆïijŇ `random.uniform()` èőąçŐŮåİĞåŇĂåĹĘåÿČéŽŔæIJžæŢřijŇ `random.gauss()` èőąçŐŮæ■čæĂĄåĹĘåÿČéŽŔæIJžæŢřãĂĆ åŕźäžŐåĚűäżŰçŽĎåĹĘåÿČæČĚåĘţèŕůåŘĆèĂČåIJÍçž£æŰĞæąčãĂĆ

åIJÍ `random` æĺąåİŮäÿ■çŽĎåĞ¡æŢřäÿ■åžŤèŕěçŤĺåIJÍåŠŇåŕEçăĄå■ęçŻÿåĚşçŽĎçÍŃåžŔäÿ■ãĂĆ åęĆædIJä¡ăçąőåőđéIJĂèęĄçśżäijijçŽĎåŁ§èČ¡ïijŇåŘŕäžěä¡£çŤĺsslæĺąåİŮäÿ■çŻÿåžŤçŽĎåĞ¡æŢřãĂĆ æŕŤåęĆïijŇ `ssl.RAND_bytes()` åŘŕäžěçŤĺæİěçŤ§æĹŘäÿĂäÿłåőĽåĚÍçŽĎéŽŔæIJžå■ŮèŁĆåžŔåĹŮãĂĆ

3.12 å§žælJňçŽĎæŮěælJ§äÿŐæŮűéŮť è¡ňæ■ć

éŮőécŸ

ä¡ăéIJĂèęĄæL'ğèąŇçőĂå■ŢçŽĎæŮűéŮť'è¡ňæ■ćïijŇæŕŤåęĆåd'l'åĹřçğŠïijŇåřŔæŮűåĹřåĹĘéŠ§ç■L'çŽĎ

èğčåĘşæŰźæąĹ

äÿžäžĘæL'ğèąŇäÿ■åŘŇæŮűéŮť'å■Ţä¡■çŽĎè¡ňæ■ćåŠŇèőąçőŮïijŇèŕůä¡£çŤĺ `datetime` æĺąåİŮãĂĆ æŕŤåęĆïijŇäÿžäžĘèąĺçd'žäÿĂäÿłæŮűéŮť'æőţïijŇåŔŕäžěåĹŻåžžäÿĂäÿł `timedelta` åőđä¿ŃïijŇåŕśåČŔäÿŃéİćè£ŹæăůïijŽ

```
>>> from datetime import timedelta
>>> a = timedelta(days=2, hours=6)
>>> b = timedelta(hours=4.5)
>>> c = a + b
>>> c.days
2
>>> c.seconds
37800
>>> c.seconds / 3600
10.5
>>> c.total_seconds() / 3600
58.5
>>>
```

åęĆædIJä¡ăæČşèąĺçd'žæŇĞåőŽçŽĎæŮěæIJ§åŠŇæŮűéŮť'ïijŇåĚĹåĹŻåžžäÿĂäÿł `datetime` åőđä¿ŃçĎűåŘŐä¡£çŤĺæăĞåĞEçŽĎæŢřå■ęè£ŘçőŮæİěæŞ■ä¡IJåőČäžñãĂĆæŕŤåęĆïijŽ

```
>>> from datetime import datetime
>>> a = datetime(2012, 9, 23)
>>> print(a + timedelta(days=10))
2012-10-03 00:00:00
```

```
>>>
>>> b = datetime(2012, 12, 21)
>>> d = b - a
>>> d.days
89
>>> now = datetime.today()
>>> print(now)
2012-12-21 14:54:43.094063
>>> print(now + timedelta(minutes=10))
2012-12-21 15:04:43.094063
>>>
```

åIJĺèőąçőŮçŽĎæŮűůåĂŹïijŇéIJĂèęĄæşĺæĎŕçŽĎæŸý datetime äijŽèĞłåŁĺåď ĎçŘĘéŮřåźť ãĂĆærŤåęĆïijŽ

```
>>> a = datetime(2012, 3, 1)
>>> b = datetime(2012, 2, 28)
>>> a - b
datetime.timedelta(2)
>>> (a - b).days
2
>>> c = datetime(2013, 3, 1)
>>> d = datetime(2013, 2, 28)
>>> (c - d).days
1
>>>
```

èőĺèőž

åŕźåď ğåď ŽæŢřå§žæIJňçŽĎæŮěæIJ§åŠŇæŮűéŮť åď ĎçŘĘéŮőécŸïijŇ datetime æĺąåİŮäžěåŔŁèűşåď §äžĘãĂĆ åęĆæđIJä¡ăéIJĂèęĄæL'ğèąŇæŻť åŁăåď■æİĆçŽĎæŮěæIJ§æŞ■ä¡IJïijŇærŤåęĆ åŔŕäžěèĂČèŹŠä¡£çŤĺ dateutilæĺąåİŮ

èőÿåď ŽçśżäijijçŽĎæŮűéŮť èőąçőŮåŔŕäžěä¡£çŤĺ dateutil.relativedelta() åĞ¡æŢřäžčæŻ£ãĂĆ ä¡ĘæŸŕïijŇæIJL'äÿĂçĆźéIJĂèęĄæşĺæĎŔçŽĎåřśæŸŕïijŇåőČäijŽåIJĺåď ĎçŘĘæIJĹäż¡(è

```
>>> a = datetime(2012, 9, 23)
>>> a + timedelta(months=1)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: 'months' is an invalid keyword argument for this function
>>>
>>> from dateutil.relativedelta import relativedelta
>>> a + relativedelta(months=+1)
datetime.datetime(2012, 10, 23, 0, 0)
>>> a + relativedelta(months=+4)
datetime.datetime(2013, 1, 23, 0, 0)
>>>
>>> # Time between two dates
```

```
>>> b = datetime(2012, 12, 21)
>>> d = b - a
>>> d
datetime.timedelta(89)
>>> d = relativedelta(b, a)
>>> d
relativedelta(months=+2, days=+28)
>>> d.months
2
>>> d.days
28
>>>
```

3.13 èőąçőŮæIJĂåŘŐäÿĂäÿłåŚÍäžŤçŽĎæŮěæIJ§

éŮőécŸ

ä¡ăéIJĂèęĄæ§ěæL'¿æŸ§æIJ§äÿ■æ§ŘäÿĂåd'l'æIJĂåŘŐåĞžçŐřçŽĎæŮěæIJ§ïijŇæŕŤåęĆæŸ§æIJ§äžŤãĂ

èğčåĘşæŰźæąĹ

PythonçŽĎ `datetime` æĺąåİŮäÿ■æIJL'åůěåĚůåĞ¡æŢřåŠŇçśżåŔŕäžěåÿőåŁl'ä¡ăæL'ğèąŇè£ŹæăůçŽĎèőäÿŃéİćæŸŕåŕźçśżäijijè£ŹæăůçŽĎéŮőécŸçŽĎäÿĂäÿłéĂŽçŤĺèğčåĘşæŰźæąĹïijŽ

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
"""
Topic: æIJĂåŘŐçŽĎåŚÍäžŤ
Desc :
"""
from datetime import datetime, timedelta

weekdays = ['Monday', 'Tuesday', 'Wednesday', 'Thursday',
            'Friday', 'Saturday', 'Sunday']


def get_previous_byday(dayname, start_date=None):
    if start_date is None:
        start_date = datetime.today()
    day_num = start_date.weekday()
    day_num_target = weekdays.index(dayname)
    days_ago = (7 + day_num - day_num_target) % 7
    if days_ago == 0:
        days_ago = 7
    target_date = start_date - timedelta(days=days_ago)
    return target_date
```

åIJÍäžď'äžŠåijŔèğčéĞŁåŹÍäÿ■ä¡£çŤĺåęĆäÿŃïijŽ

```
>>> datetime.today() # For reference
datetime.datetime(2012, 8, 28, 22, 4, 30, 263076)
>>> get_previous_byday('Monday')
datetime.datetime(2012, 8, 27, 22, 3, 57, 29045)
>>> get_previous_byday('Tuesday') # Previous week, not today
datetime.datetime(2012, 8, 21, 22, 4, 12, 629771)
>>> get_previous_byday('Friday')
datetime.datetime(2012, 8, 24, 22, 5, 9, 911393)
>>>
```

åŔŕéĂĽçŽĎ `start_date` åŔĆæŢřåŔŕäžěçŤśåŔęåď'ŰäÿĂäÿł `datetime` åőđä¿ŃæİěæŔŘä¿ŻãĂĆæŕŤåęĆïijŽ

```
>>> get_previous_byday('Sunday', datetime(2012, 12, 21))
datetime.datetime(2012, 12, 16, 0, 0)
>>>
```

èőÍèőž

äÿŁéİćçŽĎçőŮæşŢåŐ§çŘĘæŸŕè£Źæăůç ŽĎïijŽåĚĹářĘåijĂåğŃæŮěæIJ§åŠŃçŻőæăĞæŮěæIJ§æŸăářĎ
çĎűåŘŐéĂŽè£ĞæĺąèŁŘçőŮèőąçőŮåĞžçŻőæăĞæŮěæIJ§èęAçżŔè£Ğåď'ŽářŚåď'ľæĽ■èČ¡åĹřè¿¿åijĂåğŃæŮ

åęĆædIJä¡ăèęĄåČŔè£ŹæăůæĽğèąŇåď'ğéĞŔçŽĎæŮěæIJ§èőąçőŮçŽĎèŕİïijŇä¡ăæIJĂåě¡åőĽ'èčĚçňňäÿL
`python-dateutil` æİěäžčæŻ£ãĂĆ æŕŤåęĆïijŇäÿŃéİćæŸŕæŸŕä¡£çŤĺ `dateutil` æĺąåİŮäÿ■çŽĎ `relativedelta()` åĞ¡æŢřæĽğèąŇåŘŇæăůçŽĎèőąçőŮïijŽ

```
>>> from datetime import datetime
>>> from dateutil.relativedelta import relativedelta
>>> from dateutil.rrule import *
>>> d = datetime.now()
>>> print(d)
2012-12-23 16:31:52.718111

>>> # Next Friday
>>> print(d + relativedelta(weekday=FR))
2012-12-28 16:31:52.718111
>>>

>>> # Last Friday
>>> print(d + relativedelta(weekday=FR(-1)))
2012-12-21 16:31:52.718111
>>>
```

3.14 èőąçőŮå¡ŞåĽ■æIJĹäż¡çŽĎæŮěæIJ§èŇČåŻť

éŮőécŸ

ä¡ăçŽĎäžčçăĄéIJĂèęĄåIJĺå¡ŞåĽ■æIJĹäż¡äÿ■å¿łçŐŕæŕŔäÿĂåd'l'ïijŇæČşæL'¿åĹřäÿĂäÿłèőąçőŮè£Źäÿłaŕ

èğčåĘşæŰźæąĹ

åIJĺè£ŹæăůçŽĎæŮěæIJ§äÿŁå¿łçŐŕåźűéIJĂèęĄäžŃåĚĹæđĎéĂääÿĂäÿłåŇĚåŘńæL'ĂæIJL'æŮěæIJ§çŽĎ
ä¡ăåŔŕäžěåĚĹèőąçőŮåĞžåijĂåğŃæŮěæIJ§åŠŇçżŞæİ§æŮěæIJ§ïijŇ
çĎűåŘŐåIJĺä¡ăæ■čè£ŻçŽĎæŮűåĂŹä¡£çŤĺ `datetime.timedelta`
åŕźèśąéĂŠåćđè£ŹäÿłæŮěæIJ§åŔŸéĞŔå■şåŔŕãĂĆ

äÿŃéİćæŸŕäÿĂäÿłæŐěåŘŮäżżæĎŔ `datetime` åŕźèśąåźűè£ŤåŻđäÿĂäÿłçŤšå¡ŞåĽ■æIJĹäż¡åijĂåğŃæŮ

```
from datetime import datetime, date, timedelta
import calendar

def get_month_range(start_date=None):
    if start_date is None:
        start_date = date.today().replace(day=1)
    _, days_in_month = calendar.monthrange(start_date.year, start_
↪date.month)
    end_date = start_date + timedelta(days=days_in_month)
    return (start_date, end_date)
```

æIJL'äžĘè£ŹäÿłåŕšåŔŕäžěå¿ĹåőźæŸŞçŽĎåIJĺè£ŤåŻđçŽĎæŮěæIJ§èŇČåŻť'äÿŁéİćåĄŽå¿łçŐŕæŞ■ä¡IJäž

```
>>> a_day = timedelta(days=1)
>>> first_day, last_day = get_month_range()
>>> while first_day < last_day:
...     print(first_day)
...     first_day += a_day
...
2012-08-01
2012-08-02
2012-08-03
2012-08-04
2012-08-05
2012-08-06
2012-08-07
2012-08-08
2012-08-09
#... and so on...
```

èőĺèőž

äÿŁéİćçŽĎäžčçăĄåĚĹèőąçőŮåĞžäÿĂäÿłåŕźåžŤæIJĹäż¡çňňäÿĂåď'ĺçŽĎæŮĕæIJ§ãĂĆ äÿĂäÿłå£ńéĂ§çŽĎæŰźæşŢåřśæŸŕä¡£çŤĺ `date` æĹŰ `datetime` åŕźèśąçŽĎ `replace()` æŰźæşŢçőĂå■ŢçŽĎåřĘ `days` åśđæĂğëő¿ç¡őæĹŘ1å■şåŔŕãĂĆ `replace()` æŰźæşŢäÿĂäÿłåě¡åď'ĎåŕśæŸŕåőČäijŽåĹŻåžžåŠŇä¡ăåijĂåğŃäijăåĚěåŕźèśąçśžåđŃçŻÿåŘŇçŽĎåŕźèśąãĂĆ æĽ'Ăäżěïij Ňåę Ćæđ IJè¿ŞåĚěåŔĆæŢřæŸŕäÿĂäÿł `date` åőđä¿ŃïijŇéĆčäźĹçżŞæđIJäź§æŸŕäÿĂäÿł `date` åőđä¿ŃãĂĆ åŘŇæăůçŽĎïijŇåęĆæđIJè¿ŞåĚěæŸŕäÿĂäÿł `datetime` åőđä¿ŃïijŇéĆčäźĹä¡ăå¿ŮåĹřçŽĎåŕśæŸŕäÿĂäÿł `datetime` åőđä¿ŃãĂĆ

çĎűåŘŐïijŇä¡£çŤĺ `calendar.monthrange()` åĞ¡æŢřæİěæĽ¿åĞžèŕěæIJĹçŽĎæĂżåď'ĺæŢřãĂĆ äżżä¡ŢæŮűåĂŹåŔłèęĄä¡ăæČşèŐůå¿ŮæŮĕåŐEä£ąæĄŕïijŇéĆčäźĹ `calendar` æĺąåİŮåŕśéİđåÿÿæIJĽçŤĺäžEąĂĆ `monthrange()` åĞ¡æŢřäijŽè£ŤåŽđåŇĚåŘńæŸ§æIJ§åŠŇèŕěæIJĹåď'ĺæŢřçŽĎåĚČçżĎãĂĆ

äÿĂæŮęèŕěæIJĹçŽĎåď'ĺæŢřåůšç§ëäžĘïijŇéĆčäźĹçżŞæİ§æŮĕæIJ§åŕśåŔŕäžěéĂŽè£ĞåIJĺåijĂåğŃæŮĕæ æIJĽäÿłéIJĂèęĄæşĺæĎŔçŽĎæŸŕçżŞæİ§æŮĕæIJ§åźűäÿ■åŇĚåŘńåIJĺè£ŹäÿłæŮĕæIJ§èŇČåŻť'åĘĚ(äžŃåőđäÿ è£ŹäÿłåŠŇPythonçŽĎ `slice` äÿŐ `range` æŞ■ä¡IJèąŇäÿžä£ĺæŇĄäÿĂèĞť'ïijŇåŘŇæăůäź§äÿ■åŇĚåŘńçżŞåï

äÿžäžĘåIJĺæŮĕæIJ§èŇČåŻť'äÿŁå¿łçŐŕïijŇèęĄä¡£çŤĺåĹřæăĞåĞEçŽĎæŢřå■ęåŠŇæŕŤè¿ČæŞ■ä¡IJãĂĆ æŕŤåęĆïijŇåŘŕäžěåĹ'çŤĺ `timedelta` åőđä¿ŃæİěéĂŠáćđæŮĕæIJ§ïijŇåŕŘäžŐåŘů<çŤĺæİěæčĂæ§ëäÿĂäÿła

çŘĘæČşæČĚåĘţäÿŇïijŇåęĆæđIJèČ¡äÿžæŮĕæIJ§è£■äžčåĹŻåžžäÿĂäÿłåŘŇåĘĚç¡őçŽĎ `range()` åĞ¡æŢřäÿĂæăůçŽĎåĞ¡æŢřåŕśåě¡äžĘãĂĆ åźÿè£ŘçŽĎæŸŕïijŇåŘŕäžěä¡£çŤĺäÿĂäÿłçŤ§æĹŘåŹĺæİ

```
def date_range(start, stop, step):
    while start < stop:
        yield start
        start += step
```

äÿŃéİćæŸŕä¡£çŤĺè£ŹäÿłçŤ§æĹŘåŹĺçŽĎä¿Ńå■ŘïijŽ

```
>>> for d in date_range(datetime(2012, 9, 1), datetime(2012,10,1),
                        timedelta(hours=6)):
...     print(d)
...
2012-09-01 00:00:00
2012-09-01 06:00:00
2012-09-01 12:00:00
2012-09-01 18:00:00
2012-09-02 00:00:00
2012-09-02 06:00:00
...
>>>
```

è£Źçğ■åőđçŐřäźŃæĽĂäžěè£ŹäźĹçőĂå■ŢïijŇè£Ÿå¿Ůå¡ŠåŁ§äžŐPythonäÿ■çŽĎæŮĕæIJ§åŠŇæŮűéŮť'ă

3.15 å■Ůçňęäÿšè¡ňæ■ćäÿžæŮěæIJ§

éŮőécŸ

ä¡äçŽĎåžŤçŤĺçĺŃåžŔæŐěåŔŮå■ŮçňęäÿšæäijåijŔçŽĎè¿ŞåĚěïijÑä¡ĘæŸŕä¡äæČşåřĘåőČäžňè¡ňæ■ćäÿž datetime åŕźèśąäžěä¿£åIJĺäÿŁéĺćæĽğèąŇéİđå■Ůçňęäÿšæş■ä¡IJãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤĺPythonçŽĎæăĞåĞĘæĺąåİŮ datetime åŔŕäžěå¿ĹåőźæŸ§çŽĎèğčåĘşè£ŹäÿłéŮőécŸãĂĆæŕŤåęĆ

```
>>> from datetime import datetime
>>> text = '2012-09-20'
>>> y = datetime.strptime(text, '%Y-%m-%d')
>>> z = datetime.now()
>>> diff = z - y
>>> diff
datetime.timedelta(3, 77824, 177393)
>>>
```

èőĺèőž

datetime.strptime() æŰźæşŢæŤŕæŇĄå¿Ĺåd'ŽçŽĎæăijåijŔåŇŰäžčçăĄïijÑ æŕŤåęĆ %Y äžčèąĺ4ä¡■æŢřåźt'äž¡ïijÑ %m äžčèąĺäÿd'ä¡■æŢřæIJĹäž¡ãĂĆ è£ŸæIJL'äÿĂçĆźåĂijå¿ŮæşĺæĎŔçŽĎæŸŕè£ŹäžŻæăijåijŔåŇŰå■ăä¡■çňęäź§åŔŕäžěåŔ■è£Ğæĺěä¡£çŤĺïijÑårE

æŕŤåęĆïijÑåĄĞèő¿ä¡äçŽĎäžčçăĄäÿ■çŤ§æĹŘäžĘäÿĂäÿł datetime åŕźèśąïijÑ ä¡äæČşåřĘåőČæăijåijŔåŇŰäÿžæijĆäžőæŸŞèŕžå¡ćåijŔåŘŐæŤ¿åIJĺèĞłåŁĺçŤ§æĹŘçŽĎä£ąäžűæĹŰèĂĚæŁěå

```
>>> z
datetime.datetime(2012, 9, 23, 21, 37, 4, 177393)
>>> nice_z = datetime.strftime(z, '%A %B %d, %Y')
>>> nice_z
'Sunday September 23, 2012'
>>>
```

è£ŸæIJL'äÿĂçĆźéIJĂèęĄæşĺæĎŔçŽĎæŸŕïijÑ strptime() çŽĎæĂğèČ¡èęĄæŕŤä¡äæČşèśąäÿ■çŽĎåůőå¿Ĺåd'ŽïijÑ åŻăäÿžåőČæŸŕä¡£çŤĺçžŕPythonåðđçŐřïijÑåźűäÿŤå£Ł åęĆæđIJä¡äèęĄåIJĺäžčçăĄäÿ■éIJĂèęĄèğčæđŘåd'ğéĞŔçŽĎæŮěæIJ§åźűäÿŤåůšçżŔç§ěéĄŞäžĘæŮěæIJ§å■Ůç æŕŤåęĆïijÑåęĆæđIJä¡äåůšçżŔç§ěéĄŞæĽĂäžěæŮěæIJ§æăijåijŔæŸŕ YYYY-MM-DD ïijÑä¡äåŔŕäžěåČŔäÿŃéİćè£ŹæăůåőđçŐřäÿĂäÿłèğčæđŘåĞ¡æŢřïijŽ

```
from datetime import datetime
def parse_ymd(s):
    year_s, mon_s, day_s = s.split('-')
    return datetime(int(year_s), int(mon_s), int(day_s))
```

åőđéŹĚæţŃèŕŢäÿ■ïijŇè£ŹäÿłåĞ¡æŢřæŕŤ `datetime.strptime()` å£ń7åĂ■åď˙ŽãĂĆ
åęĆæđIJä¡ãèęĄåď˙ĎçŘĘåď˙ğéĞŔçŽĎæűĽ˙åŔŁåĹřæŮĕæIJ§çŽĎæŢřæ■őçŽĎèŕİïijŇéĆčäźĹæIJĂåě¡èĂČèŹŚäÿ

3.16 çžŞåŘĹæŮűåŇžçŽĎæŮĕæIJ§æŞ■ä¡IJ

éŮőécŸ

ä¡ăæIJL˙äÿĂäÿłåőL˙æŐŠåIJĺ2012åźt˙12æIJĹ21æŮĕæŮl˙äÿŁ9:30çŽĎçŢţèŕİäijŽèőőïijŇåIJřçĆźåIJĺèŁİåŁă
èĂŇä¡ăçŽĎæIJŃåŔŃåIJĺå■řåžęçŽĎçŔ■åŁăç¡ŮåřŤïijŇéĆčäźĹäžŮåžŤèŕěåIJĺå¡ŞåIJřæŮűéŮt˙åĞăçĆźåŔĆåŁă

èğčåĘşæŰźæąĹ

åŕźåĞääźŐæL˙ĂæIJL˙æűL˙åŔŁåĹřæŮűåŇžçŽĎéŮőécŸïijŇä¡äéČ¡åžŤèŕěä¡£çŤĺ
`pytz` æĺąåİŮãĂĆè£ŹäÿłåŇĚæŔŘä¿ŻäžĘOlsonæŮűåŇžæŢřæ■őåžŞïijŇ
åőĆæŸřæŮűåŇžä£ąæĄŕçŽĎäžŃåőđäÿŁçŽĎæăĞåĞĘïijŇåIJĺå¿Ĺåď˙Žèŕ■èĺĂåŠŇæŞ■ä¡IJçşżçż§éĞŇéİćéČ¡åŔ

`pytz` æĺąåİŮäÿĂäÿłäÿżèęĄçŤĺéĂŤæŸŕårĘ `datetime`
åžŞåĹŻåżžçŽĎçőĂå■ŢæŮĕæIJ§årźèśąæIJňåIJřåŇŰãĂĆ ærŤåęĆïijŇäÿŃéİcåęĆä¡Ţèąĺçď˙žäÿĂäÿłèŁİåŁääŞěa

```
>>> from datetime import datetime
>>> from pytz import timezone
>>> d = datetime(2012, 12, 21, 9, 30, 0)
>>> print(d)
2012-12-21 09:30:00
>>>

>>> # Localize the date for Chicago
>>> central = timezone('US/Central')
>>> loc_d = central.localize(d)
>>> print(loc_d)
2012-12-21 09:30:00-06:00
>>>
```

äÿĂæŮęæŮĕæIJ§èćńæIJňåIJřåŇŰäžĘïijŇ åőČårśåŔŕäžèè¡ňæ■ćäÿžåĚűäžŰæŮűåŇžçŽĎæŮűéŮt˙äžĘãĂĆ
äÿžäžĘå¿ŮåĹřçŔ■åŁăç¡ŮåřŤårźåžŤçŽĎæŮűéŮt˙ïijŇä¡ääŔŕäžèè£ŹæăůåĄŽïijŽ

```
>>> # Convert to Bangalore time
>>> bang_d = loc_d.astimezone(timezone('Asia/Kolkata'))
>>> print(bang_d)
2012-12-21 21:00:00+05:30
>>>
```

åęĆæđIJä¡ăæL˙ŞçőŮåIJĺæIJňåIJřåŇŰæŮĕæIJ§äÿŁæL˙ğèąŇèőąçőŮïijŇä¡ăéIJĂèęĄçL˙źåĹńæşĺæĎŔåď˙Ŕä
ærŤåęĆïijŇåIJĺ2013åźt˙ïijŇç¿ŐåŻ¡æăĞåĞĘåď˙Ŕäžd˙æŮűæŮűéŮt˙åijĂågŇäžŐæIJňåIJřæŮűéŮt˙3æIJĹ13æŮ
åęĆæđIJä¡ăæ■čåIJĺæL˙ğèąŇæIJňåIJŕèőąçőŮïijŇä¡ääijŽå¿ŮåĹřäÿĂäÿłéŤŹèŕřãĂĆærŤåęĆïijŽ

```
>>> d = datetime(2013, 3, 10, 1, 45)
>>> loc_d = central.localize(d)
>>> print(loc_d)
2013-03-10 01:45:00-06:00
>>> later = loc_d + timedelta(minutes=30)
>>> print(later)
2013-03-10 02:15:00-06:00 # WRONG! WRONG!
>>>
```

çżŞæđIJéŤŹèŕŕæŸŕåŻääÿžåőČåźűæšąæIJL'èĂČèŹŚåIJĺæIJňåIJřæŮűéŮt'äÿ■æIJL'äÿĂåřŔæŮűçŽĎèůşèů äÿžäžĘä£őæ■čè£ŹäÿłéŤŹèŕŕïijŇåŔŕäżěä¡£çŤĺæŮűåŇžåŕźèšą `normalize()` æŰźæşŢãĂĆæŕŤåęĆïijŽ

```
>>> from datetime import timedelta
>>> later = central.normalize(loc_d + timedelta(minutes=30))
>>> print(later)
2013-03-10 03:15:00-05:00
>>>
```

èőĺèőž

äÿžäžĘäÿ■èől'ä¡ăèćńè£ŹäžŻäÿIJäÿIJåijĎçŽĎæŹŢåd'ť'è¡ňåŘŚïijŇåd'ĎçŘĘæIJňåIJřåŇŰæŮěæIJ§çŽĎéĂ åźűçŤĺåőČæİěæL'ğèąŇæL'ĂæIJL'çŽĎäÿ■éŮt'å■ŸåĆĺåŠŇæŞ■ä¡IJãĂĆæŕŤåęĆïijŽ

```
>>> print(loc_d)
2013-03-10 01:45:00-06:00
>>> utc_d = loc_d.astimezone(pytz.utc)
>>> print(utc_d)
2013-03-10 07:45:00+00:00
>>>
```

äÿĂæŮęè¡ňæ■ćäÿžUTCïijŇä¡ăåřśäÿ■çŤĺåŐżæŃĚå£Čèů§åd'Ŕäžd'æŮűçŻÿåĚşçŽĎéŮőécŸäžĘãĂĆ åŻăæ■d'ïijŇä¡ăåŔŕäżěèů§äźŃåL'■äÿĂæăůæŤ¿å£ČçŽĎæL'ğèąŇåÿÿèğĄçŽĎæŮěæIJ§èőąçőŮãĂĆ å¡Şä¡ăæČşåřĘè¿ŞåĞžåŔŸäÿžæIJňåIJřæŮűéŮt'çŽĎæŮűåĂŹïijŇä¡£çŤĺåŘĹéĂĆçŽĎæŮűåŇžåŐżè¡ňæ■ćäÿŃå

```
>>> later_utc = utc_d + timedelta(minutes=30)
>>> print(later_utc.astimezone(central))
2013-03-10 03:15:00-05:00
>>>
```

å¡ŞæűL'åŔŁåĹřæŮűåŇžæŞ■ä¡IJçŽĎæŮűåĂŹïijŇæIJL'äÿłéŮőécŸåřśæŸŕæĹŚäżňåęĆä¡Ţå¿ŮåĹřæŮűåŇ æŕŤåęĆïijŇåIJĺè£ŹäÿŀäżŃå■Řäÿ■ïijŇæĹŚäżňåęĆä¡Ţç§ěéĄŞâĂIJAsia/KolkataâĂİåřśæŸŕå■řåžęåŕźåžŤçŽĎæ äÿžäžĘæ§ěæL'¿ïijŇåŔŕäżěä¡£çŤĺISO 3166åŻ¡åőűäžčçăĄä¡IJäÿžåĚşéŤőå■ŮåŐżæ§ěéŸĚå■ŮåĚÿ `pytz.country_timezones` ãĂĆæŕŤåęĆïijŽ

```
>>> pytz.country_timezones['IN']
['Asia/Kolkata']
>>>
```

æşĺïijŽå¡Şä¡ăéŸĚèŕžåĹřè£ŹéĞŇçŽĎæŮűåĂŹïijŇæIJL'åŔŕèČ¡ pytz
æĺąåİŮäžěåŔŁäÿ■åĘ■åżžèőőä¡£çŤĺäžĘïijŇåŻääÿžPEP431æŔŘåĞžäžĘæŻť'åĚĹè£ŻçŽĎæŮűåŇžæŤŕæŇĄãĂ
ä¡ĘæŸŕè£ŹéĞŇèřĹåĹřçŽĎå¿Ĺåď'ŽéŮőécŸè£ŸæŸŕæIJL'åŔĆèĂČäžůåĂijçŽĎ(æŕŤåęĆä¡£çŤĺUTCæŮěæIJ§ç

CHAPTER 6

çňňåŻŻçńäïijŽè£■äžčåŹÍäÿŐçŤ§æĹŘåŹÍ

è£■äžčæŸŕPythonæIJĂåijžåd'ğçŽĎåŁ§èČ¡äźŃäÿĂãĂĆåĹİçIJŃèţůæİěïijŇä¡ăåŔŕèČ¡äijŽçőĂå■ŢçŽĎèőd
çĎűèĂŇïijŇçżIéİđäžĚäžĚåřśæŸŕåęĆæ■d'ïijŇè£ŸæIJL'å¿Ĺåd'Žä¡ăåŔŕèČ¡äÿ■ç§ěéĄŞçŽĎïijŇ
æŕŤåęĆåĹŻåžžä¡ăèĞłåůśçŽĎè£■äžčåŹÍåŕźèśąïijŇåIJĺitertoolsæĺąåİŮäÿ■ä¡£çŤĺæIJL'çŤĺçŽĎè£■äžčæĺąåijŔïijĺ
è£ŹäÿĂçńăçŽőçŽĎåřśæŸŕåŘŚä¡ăåśŢçd'žèů§è£■äžčæIJL'åĚşçŽĎåŘĎçğ■åÿÿèğĄéŮőécŸãĂĆ

Contents:

4.1 æĽŃåŁĺéĄ■åŐĘè£■äžčåŹÍ

éŮőécŸ

ä¡ăæČşéĄ■åŐĘäÿĂäÿłåŔŕè£■äžčåŕźèśąäÿ■çŽĎæL'ĂæIJL'åĚČçt'ăïijŇä¡ĘæŸŕå■t'äÿ■æČşä¡£çŤĺforå¿łçČ

èğčåĘşæŰźæąĹ

äÿžäžĘæL'ŃåŁĺçŽĎéĄ■åŐĘåŔŕè£■äžčåŕźèśąïijŇä¡£çŤĺ `next()`
åĞ¡æŢřåźűåIJĺäžčçăĄäÿ■æ■ŢèŐů `StopIteration` åijĆåÿÿãĂĆ
æŕŤåęĆïijŇäÿŃéİćçŽĎä¿Ńå■ŘæL'ŃåŁĺèŕźåŘŰäÿĂäÿłæŰĞäžűäÿ■çŽĎæL'ĂæIJL'èąŇïijŽ

```
def manual_iter():
    with open('/etc/passwd') as f:
        try:
            while True:
                line = next(f)
                print(line, end='')
```

```
        except StopIteration:
            pass
```

éĂŽåÿÿæİěèőšïijŇ StopIteration çŤĺæİěæŇĞçd'žè£■äžčçŽĎçżŞář¿ãĂĆ çĎűèĂŇïijŇåęĆæđIJä¡ăæĽŃåŁĺä¡£çŤĺäÿŁéİćæijŤçd'žçŽĎ next() åĞ¡æŢřçŽĎèŕİïijŇä¡ăè£ŸåŕäžěéĂŽè£Ğè£ŤåŻđäÿĂäÿłæŇĞåőŽåĂijæİěæăĞèőřçżŞář¿ïijŇæŕŤåęĆ None ãĂĆ äÿŃéİćæŸŕçd'žä¿ŃïijŽ

```
with open('/etc/passwd') as f:
    while True:
        line = next(f, None)
        if line is None:
            break
        print(line, end='')
```

èőĺèőž

åd'ğåd'ŽæŢřæČĚåĘţäÿŃïijŇæĹŚäžňäijŽä¡£çŤĺ for å¿łçŐŕèŕ■åŔěçŤĺæİěéĄ■åŐĘäÿĂäÿłåŔŕè£■äžčåŕźè
ä¡ĘæŸŕïijŇåĄűåřŤäź§éIJĂèęĄåŕźè£■äžčåĄŽæŻt'åŁăçš¿çąőçŽĎæŐğåĹűïijŇè£ŹæŮűåĂŹäžĘèğčåžŢåśĆè£■

äÿŃéİćçŽĎäžd'äžŠçd'žä¿ŃåŘŚæĹŚäžňæijŤçd'žäžĘè£■äžčæIJ§éŮt'æĽĂåŔŚçŤ§çŽĎå§žæIJňçżĘèŁĆïijŽ

```
>>> items = [1, 2, 3]
>>> # Get the iterator
>>> it = iter(items) # Invokes items.__iter__()
>>> # Run the iterator
>>> next(it) # Invokes it.__next__()
1
>>> next(it)
2
>>> next(it)
3
>>> next(it)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
StopIteration
>>>
```

æIJňçńäæŐěäÿŃæİěåĞăåřŔèŁĆäijŽæŻt'æůšåĚěçŽĎèőšèğčè£■äžčçŻÿåĚşæŁĂæIJŕïijŇåĽ■æŔŘæŸŕä¡ăă
æĽĂäžěçąőä£Ïä¡ååůšçżŔæŁŁè£ŹçńäçŽĎåĘĚåőźçĽćçĽćèőřåIJĺå£Čäÿ■ãĂĆ

4.2 äžčçŘĘè£■äžč

éůőécŸ

ä¡ăæđĎåžžäžĘäÿĂäÿłèĞłåőŽäźĽåőźåŹĺåŕźèśąïijŇéĞŇéİćåŇĚåŘńæIJL'åĹŮèąĺãĂĄåĚČçżĎæĹŰåĚűäżÚ
ä¡ăæČşçŻť'æŐěåIJĺä¡ăçŽĎè£ŹäÿłæŰřåőźåŹĺåŕźèśąäÿŁæL'ğèąŇè£■äžčæŞ■ä¡IJãĂĆ

èğčåĘşæŰźæąĹ

åőđéŹĚäÿŁä¡ăåŘłéIJĂèęĄåőŽäźĽäÿĂäÿł `__iter__()`
æŰźæşŢïijŇářĘè£■äžčæŞ■ä¡IJäžčçŘĘåĹřåőźåŹĺåĘĚéČĺçŽĎåŕźèśąäÿŁåŐžãĂĆærŤåęĆïijŽ

```
class Node:
    def __init__(self, value):
        self._value = value
        self._children = []

    def __repr__(self):
        return 'Node({!r})'.format(self._value)

    def add_child(self, node):
        self._children.append(node)

    def __iter__(self):
        return iter(self._children)

# Example
if __name__ == '__main__':
    root = Node(0)
    child1 = Node(1)
    child2 = Node(2)
    root.add_child(child1)
    root.add_child(child2)
    # Outputs Node(1), Node(2)
    for ch in root:
        print(ch)
```

åIJĺäÿŁéİćäžčçăĄäÿ■ïijŇ `__iter__()` æŰźæşŢåŘłæŸŕçőĂå■ŢçŽĎåřĘè£■äžčèŕůæśĆäijäéĂŠçżŹåĘĚé
`_children` åśđæĂğãĂĆ

èőĺèőž

PythonçŽĎè£■äžčåŹĺå■Řèőőéİ JĂèęĄ `__iter__()` æŰźæşŢè£ŤåŽđäÿĂäÿłåőđçŐřäžĘ
`__next__()` æŰźæşŢçŽĎè£■äžčåŹĺåŕźèśąãĂĆ åęĆæđIJä¡ăåŘłæŸŕè£■äžčéĄ■åŐĘåĚűäžŮåőźåŹĺçŽĎåĘĚå

è£ŹéĞŇçŽĎ `iter()` åĞ¡æŢřçŽĎä¡£çŤĺçőĂåŇŰäžĘäžčçăĄïijŇ
`iter(s)` åŘłæŸŕçőĂå■ŢçŽĎéĂŽè£ĞèřČçŤĺ `s.__iter__()`

æŰźæşŢæİěè£ŤåŻđåŕźåžŤçŽĎè£■äžčåŹĺåŕźèśąïijŇ åŕśèů§ `len(s)` äijŽèřČçŤÍ `s.__len__()` åŐ§çŘĘæŸŕäÿĂæăůçŽĎãĂĆ

4.3 ä¡£çŤÍçŤ§æĹŘåŹĺåĹŻåžžæŰřçŽĎè£■äžčæĺąåijŔ

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłèĞłåőŽäźL'è£■äžčæĺąåijŔïijŇèů§æŹőéĂŽçŽĎåĘĚç¡őåĞ¡æŢřæŕŤåęĆ `range()`,`reversed()` äÿ■äÿĂæăůãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşåőđçŐřäÿĂçğ■æŰřçŽĎè£■äžčæĺąåijŔïijŇä¡£çŤÍäÿĂäÿłçŤ§æĹŘåŹĺåĞ¡æŢřæİěåőŽäźL'åäÿŃéIćæŸŕäÿĂäÿłçŤ§äžğæ§ŘäÿłèŇČåŻt'åĘĚæţőçĆźæŢřçŽĎçŤ§æĹŘåŹĺïijŽ

```
def frange(start, stop, increment):
    x = start
    while x < stop:
        yield x
        x += increment
```

äÿžäžĘä¡£çŤÍè£ŹäÿłåĞ¡æŢřïijŇ ä¡ăåŔŕäžěçŤÍforå¿łçŐŕè£■äžčåőČæĹŰèĂĚä¡£çŤÍåĚűäžŰæŐěåŔŮäÿĂä`sum()`,`list()` ç■Ľ')ãĂĆçd'žä¿ŃåęĆäÿŃïijŽ

```
>>> for n in frange(0, 4, 0.5):
...     print(n)
...
0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
>>> list(frange(0, 1, 0.125))
[0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875]
>>>
```

èőÍèőž

äÿĂäÿłåĞ¡æŢřäÿ■éIJĂèęĄæIJL'äÿĂäÿł `yield` èŕ■åŘěå■şåŔŕårĘåĚűè¡ñæ■ćäÿžäÿĂäÿłçŤ§æĹŘåŹĺãĂĆèů§æŹőéĂŽåĞ¡æŢřäÿ■åŘŃçŽĎæŸŕïijŇçŤ§æĹŘåŹĺåŔłèČ¡çŤÍäžŐè£■äžčæŞ■ä¡IJãĂĆäÿŃéIćæŸŕäÿĂäÿłåőđéłŃïijŇåŘŚä¡ăåśŢçd'žè£ŹæăůçŽĎåĞ¡æŢřåžŢåśĆåůěä¡IJæIJžåĹűïijŽ

```
>>> def countdown(n):
...     print('Starting to count from', n)
...     while n > 0:
...         yield n
...         n -= 1
...     print('Done!')
...

>>> # Create the generator, notice no output appears
>>> c = countdown(3)
>>> c
<generator object countdown at 0x1006a0af0>

>>> # Run to first yield and emit a value
>>> next(c)
Starting to count from 3
3

>>> # Run to the next yield
>>> next(c)
2

>>> # Run to next yield
>>> next(c)
1

>>> # Run to next yield (iteration stops)
>>> next(c)
Done!
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
StopIteration
>>>
```

äÿĂäÿłçŤ§æĹŘåŹĺåĞ¡æŢřäÿżèęĄçĽźå¿ĄæŸŕåőČåŔłäijŽåŻđåžŤåIJĺè£■äžčäÿ■ä¡£çŤĺåĹřçŽĎ
next æŞ■ä¡IJãĂĆ äÿĂæŮęçŤ§æĹŘåŹĺåĞ¡æŢřè£ŤåŻđéĂĂåĞžiijŇè£■äžčçżĹæ■ćãĂĆæĹŚäžňåIJĺè£■äžčäÿ■é

4.4 åőđçŐřè£■äžčåŹĺå■Ŕèőő

éŮőécŸ

ä¡ăæČşæđĎåžžäÿĂäÿłèČ¡æŤŕæŇĄè£■äžčæŞ■ä¡IJçŽĎèĞłåőŽäźL’ářźèśąïijŇåźűåÿŇæIJŻæL’¿åĹřäÿĂäÿł

èğčåĘşæŰźæąĹ

çŽőåĽ■äÿžæ■ćïijŇåIJĺäÿĂäÿłåŕźèśąäÿŁåőđçŐřè£■äžčæIJĂçőĂå■ŢçŽĎæŰźåijŔæŸŕä¡£çŤĺäÿĂäÿłçŤ§æ
åIJĺ4.2åřŔèŁĆäÿ■ïijŇä¡£çŤĺNodeçśžæİèèąĺçđ'žæăŚå¡ćæŢřæ■őçżŞæđĎãĂĆä¡ăåŔřèČ¡æČşåőđçŐřäÿĂäÿłäžěa
äÿŃéİćæŸŕäžčçăĄçđ'žä¿ŃïijŽ

```
class Node:
    def __init__(self, value):
        self._value = value
        self._children = []

    def __repr__(self):
        return 'Node({!r})'.format(self._value)

    def add_child(self, node):
        self._children.append(node)

    def __iter__(self):
        return iter(self._children)

    def depth_first(self):
        yield self
        for c in self:
            yield from c.depth_first()

# Example
if __name__ == '__main__':
    root = Node(0)
    child1 = Node(1)
    child2 = Node(2)
    root.add_child(child1)
    root.add_child(child2)
    child1.add_child(Node(3))
    child1.add_child(Node(4))
    child2.add_child(Node(5))

    for ch in root.depth_first():
        print(ch)
    # Outputs Node(0), Node(1), Node(3), Node(4), Node(2), Node(5)
```

åIJĺè£Źæőţäžčçă Ąäÿ■ïijŇdepth_first() æŰźæşŢçőĂå■ŢçŽťèğĆãĂĆ
åőČéęŰåĚĹè£ŤåŻđèĞłåůśæIJňèžńäźűè£■äžčæŕŔäÿĂäÿłå■ŘèŁĆçĆźåźű
éĂŽè£ĞèřČçŤĺå■ŘèŁĆçĆźçŽĎ depth_first() æŰźæşŢ(ä¡£çŤĺ yield from
èŕ■åŔě)è£ŤåŻđåŕźåžŤåĚČçť'ããĂĆ

èőĺèőž

PythonçŽĎè£■äžčå■ŔèőőèęĄæśĆäÿĂäÿł __iter__()
æŰźæşŢè£ŤåŻđäÿĂäÿłçL'źæőŁçŽĎè£■äžčåŹĺåŕźèśąïijŇ è£Źäÿłè£■äžčåŹĺåŕźèśąåőđçŐřäžĘ

__next__() æŰźæşŢåźűéĂŽè£Ğ StopIteration åijĆåÿÿæăĞèŕĘè£■äžčçŽĎåőŇæĹŘãĂĆ ä¡ĘæŸŕřijŇåőđçŐřè£ŹäžŽéĂŽåÿÿäijŽæŕŤè¿ČçźĄçŘŘãĂĆ äÿŃéİćæĹŚäžňæijŤçd'žäÿŃè£Źçğ■æŰźåijŔřijŇå depth_first() æŰźæşŢïijŽ

```
class Node2:
    def __init__(self, value):
        self._value = value
        self._children = []

    def __repr__(self):
        return 'Node({!r})'.format(self._value)

    def add_child(self, node):
        self._children.append(node)

    def __iter__(self):
        return iter(self._children)

    def depth_first(self):
        return DepthFirstIterator(self)


class DepthFirstIterator(object):
    '''
    Depth-first traversal
    '''

    def __init__(self, start_node):
        self._node = start_node
        self._children_iter = None
        self._child_iter = None

    def __iter__(self):
        return self

    def __next__(self):
        # Return myself if just started; create an iterator for␣
↪children
        if self._children_iter is None:
            self._children_iter = iter(self._node)
            return self._node
        # If processing a child, return its next item
        elif self._child_iter:
            try:
                nextchild = next(self._child_iter)
                return nextchild
            except StopIteration:
                self._child_iter = None
                return next(self)
        # Advance to the next child and start its iteration
        else:
```

```
            self._child_iter = next(self._children_iter).depth_
↪first()
            return next(self)
```

DepthFirstIterator çśżåŠŇäÿŁéÍćä¡£çŤÍçŤ§æĹŘåŹÍçŽĎçĽ'ĹæIJňåůěä¡IJåŐ§çŘĘçśżäijijïijŇ
ä¡ĘæŸŕåőČåĘŹèţůæÍěå¿ĹçźAçŘŘïijŇåŻääÿžè£■äžčåŹÍå£ĚéążåIJÍè£■äžčåď'ĎçŘĘè£ĞçÍŃäÿ■çżt'æŁď'åď'ğéČ
åÍęçŹ¡æÍěèőšïijŇæšąäžžæĎ£æĎŔåĘŹè£ŹäźĹæŹęæűľçŽĎäžčçăĄãĂĆåřĘä¡äçŽĎè£■äžčåŹÍåőŽäźL'äÿžäÿĂäÿ

4.5 åŔ■åŘŚè£■äžč

éŮőécŸ

ä¡ăæČşåŔ■æŰźåŘŚè£■äžčäÿĂäÿłåžŔåĹŮ

èğčåĘşæŰźæąĹ

ä¡£çŤÍåĘĚç¡őçŽĎ `reversed()` åĞ¡æŢřïijŇærŤåęĆïijŽ

```
>>> a = [1, 2, 3, 4]
>>> for x in reversed(a):
...     print(x)
...
4
3
2
1
```

åŔ■åŘŚè£■äžčäžĚäžĚå¡ŞåŕźèśąçŽĎåď'ğåřŔåŔŕécĎåĚÍçąőåőŽæĹŰèĂĚåŕźèśąåőđçŐřäžĘ
__reversed__() çŽĎçL'źæőŁæŰźæşŢæŮűæL'■èČ¡çŤ§æŢĹãĂĆ
åęĆæđIJäÿd'èĂĚéČ¡äÿ■çňęåŘĹïijŇéĆčä¡ăå£ĚéążåĚĹåřĘåŕźèśąè¡ňæ■ćäÿžäÿĂäÿłåĹŮèąÍæL'■èąŇïijŇærŤåęĆ

```
# Print a file backwards
f = open('somefile')
for line in reversed(list(f)):
    print(line, end='')
```

èęĄæşÍæĎŘçŽĎæŸŕåęĆæđIJåŔŕè£■äžčåŕźèśąåĚČçť'ăå¿Ĺåď'ŽçŽĎèŕÎïijŇåřĘåĚűécĎåĚĹè¡ňæ■ćäÿžäÿĂ

èőÍèőž

å¿Ĺåď'ŽçÍŃåžŔåŚŸåźűäÿ■ç§ěéĄŞåŔŕäžěéĂŽè£ĞåIJÍèĞłåőŽäźL'çśżäÿŁåőđçŐř
__reversed__() æŰźæşŢæÍěåőđçŐřåŔ■åŘŚè£■äžčãĆćærŤåęĆïijŽ

```
class Countdown:
    def __init__(self, start):
        self.start = start

    # Forward iterator
    def __iter__(self):
        n = self.start
        while n > 0:
            yield n
            n -= 1

    # Reverse iterator
    def __reversed__(self):
        n = 1
        while n <= self.start:
            yield n
            n += 1

for rr in reversed(Countdown(30)):
    print(rr)
for rr in Countdown(30):
    print(rr)
```

åőŽäźĽäÿĂäÿłåŔ■åŘŚè£■äžčåŹĺåŔŕäžěä¡£å¿ŮäžčçăĄéİđåÿÿçŽĎéńŸæŢĹïijŇ
åŻääÿžåőČäÿ■åĘ■éIJĂèęĄåřĘæŢřæ■őåąńåĚĚåĹřäÿĂäÿłåĹŮèąĺäÿ■çĎűåŘŐåĘ■åŐżåŔ■åŘŚè£■äžčè£ŹäÿłåĹ

4.6 åÿęælJL'åď'ŰéČÍçŁűæĂĄçŽĎçŢ§æĹŘåŹĺåĞ¡æŢř

éŮőécŸ

ä¡ăæČşåőŽäźL'äÿĂäÿłçŢ§æĹŘåŹĺåĞ¡æŢřïijŇä¡ĘæŸŕåőČäijŽèřČçŢĺæ§Řäÿłä¡ăæČşæŽťéIJšçżŹçŢĺæĹůä

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăæČşèőľ'ä¡ăçŽĎçŢ§æĹŘåŹĺæŽťéIJšåď'ŰéČÍçŁűæĂĄçżŹçŢĺæĹůïijŇ
åĹńå£ŸäžĘä¡ååŕŕäżěçőĂå■ŢçŽĎåřĘåőČåőđçŐřäÿžäÿĂäÿłçśżïijŇçĎűåŘŐæŁŁçŢ§æĹŘåŹĺåĞ¡æŢřæŢ¿åĹř
`__iter__()` æŰźæşŢäÿ■è£ĞåŐżãĂĆærŤåęĆïijŽ

```
from collections import deque

class linehistory:
    def __init__(self, lines, histlen=3):
        self.lines = lines
        self.history = deque(maxlen=histlen)

    def __iter__(self):
```

```
        for lineno, line in enumerate(self.lines, 1):
            self.history.append((lineno, line))
            yield line

    def clear(self):
        self.history.clear()
```

äÿžäžĘä¡£çŤĺè£ŹäÿłçśżïijŇä¡ååŕŕäžěåřĘåőČå¡ŞåĄŽæŸŕäÿĂäÿłæŹőéĂŽçŽĎçŤ§æĹŘåŹĺåĞ¡æŢřãĂĆ çĎűèĂŇïijŇçŤśäžŐåŕŕäžěåĹŽåžžäÿĂäÿłåőđä¿ŃåŕźèśąïijŇäžŐæŸŕä¡ääŕŕäžěèő£éŮőåĘĚéČĺåšđæĂğåĂijïijŇ ærŤåęĆ `history` åšđæĂğæĹŰèĂĚæŸý `clear()` æŰźæşŢãĂĆäžčçăĄçđ'žä¿ŃåęĆäÿŇïijŽ

```
with open('somefile.txt') as f:
    lines = linehistory(f)
    for line in lines:
        if 'python' in line:
            for lineno, hline in lines.history:
                print('{}:{}'.format(lineno, hline), end='')
```

èőĺèőž

åĚşäžŐçŤ§æĹŘåŹĺïijŇå¿Ĺåőźæ ŸŞæŐL'è£ŽåĞ¡æŢřæŮăæL'Ăäÿ■èČ¡çŽĎéŹůéŸśãĂĆ åęĆæđIJçŤ§æĹŘåŹĺåĞ¡æŢřéIJĂèęĄèů§ä¡ăçŽĎçĺŃåžŔåĚűäžŰéČĺåĹĘæL'Şäžđ'éĄŞçŽĎèŕİ(ærŤåęĆæŽt'éIJšå åŕŕèČ¡äijŽåřijèĞt'ä¡ăçŽĎäžčçăĄåijĆåÿÿçŽĎåđ'■æİĆãĂĆ åęĆæđIJæŸŕè£ŹçğæČĚåĘţçŽĎèŕİïijŇåŕŕäžěèĂĆ åIJĺ `__iter__()` æŰźæşŢäÿ■åőŽäźL'ä¡äçŽĎçŤ§æĹŘåŹĺäÿ■äijŽæŤźåŔŸä¡ääžżä¡ŢçŽĎçőŮæşŢéĂżè¿ŚãĂĆ çŤśäžŐåőČæŸŕçśżçŽĎäÿĂéČĺåĹĘïijŇæL'ĂäžěåĚĄèőÿä¡ääőŽäźL'åŘĎçğ■åšđæĂğåŠŇæŰźæşŢæİěä¿ŻçŤĺæĹ

äÿĂäÿłéIJĂèęĄæşĺæĎŔçŽĎåřŔåIJřæŰźæŸŕïijŇåęĆæđIJä¡ääIJĺè£■äžčæŞ■ä¡IJæŮűäÿ■ä¡£çŤĺforå¿łçŐŕè `iter()` åĞ¡æŢřãĂĆærŤåęĆïijŽ

```
>>> f = open('somefile.txt')
>>> lines = linehistory(f)
>>> next(lines)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: 'linehistory' object is not an iterator

>>> # Call iter() first, then start iterating
>>> it = iter(lines)
>>> next(it)
'hello world\n'
>>> next(it)
'this is a test\n'
>>>
```

4.7 è£■äžčåŹĺåĹĞçĽĞ

éŮőécŸ

ä¡ăæČşå¿ŮåĹřäÿĂäÿłçŤśè£■äžčåŹĺçŤ§æĹŘçŽĎåĹĞçĽĞåŕźè śąïijŇä¡ĘæŸŕæăĞåĞĘåĹĞçĽĞæŞ■ä¡IJåź

èğčåĘşæŰźæąĹ

åĞ¡æŢř `itertools.islice()` æ■čåě¡éĂĆçŤĺäžŐåIJĺè£■äžčåŹĺåŠŇçŤ§æĹŘåŹĺäÿŁåĄŽåĹĞçĽĞæŞ

```
>>> def count(n):
...     while True:
...         yield n
...         n += 1
...
>>> c = count(0)
>>> c[10:20]
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: 'generator' object is not subscriptable

>>> # Now using islice()
>>> import itertools
>>> for x in itertools.islice(c, 10, 20):
...     print(x)
...
10
11
12
13
14
15
16
17
18
19
>>>
```

èőĺèőž

è£■äžčåŹĺåŠŇçŤ§æĹŘåŹĺäÿ■èČ¡ä¡£çŤĺæăĞåĞĘçŽĎåĹĞçĽĞæŞ■ä¡IJïijŇåŻăäÿžåőČäžňçŽĎéŢ£åžęäžŇ åĞ¡æŢř `islice()` è£ŤåŻđäÿĂäÿłåŕźäžěçŤ§æĹŘæŇĞåőŽåĚČçţ'äçŽĎè£■äžčåŹĺïijŇåőČéĂŽè£ĞéĄ■åŐĘå çĎűåŘŐæĽ'■åijĂåğŃäÿĂäÿłäÿłçŽĎè£ŤåŻđåĚČçţ'äïijŇåźűçŻţ'åĹřåĹĞçĽĞçżŞæİ§çţ'ćåijŢä¡■ç¡õãĂĆ

è£ŹéĞŇèęĄçİĂéĞ■åijžèřČçŽĎäÿĂçĆźæŸŕ `islice()` äijŽæűĹèĂŮæŐĽ'äijăåĚěçŽĎè£■äžčåŹĺäÿ■çŽĎæŢřæ■őãĂĆ å£Ěéążè ĂČèŹŚåĹřè£■äžčåŹĺæŸŕäÿ■åŘŕéĂĘçŽ æĽ'Ăäžěåę ĆæđIJä¡äéIJĂèęĄäźŃåŘŐåĘ■æňąèő£éŮőè£Źäÿłè£■äžčåŹĺçŽĎèŕİïijŇéĆčä¡ăářšå¿ŮåĚĹářĘåőČéĞ

4.8 èůşè£ĞåŔŕè£■äžčåŕźèśąçŽĎåijĂåğŃéČĺåĹĘ

éŮőécŸ

ä¡ăæČşéĄ■åŐĘäÿĂäÿłåŔŕè£■äžčåŕźèśąïijÑä¡ĘæŸŕåőČåijĂåğŃçŽĎæ§ŘäžŽåĚČçť'ää¡ååźűäÿ■æĎ§åĚť'è

èğčåĘşæŰźæąĹ

itertools æĺąåİŮäÿ■æIJL'äÿĂäžŻåĞ¡æŢřåŔŕäžěåőŇæĹŘè£ŹäÿłäżżåŁąãĂĆ éęŰåĚĹäžŃçż■çŽĎæŸŕ itertools.dropwhile() åĞ¡æŢřãĂĆä¡£çŤĺæŮűïijÑä¡ăçżŹåőČäijåéĂŠäÿĂäÿ åőČäijŽè£ŤåŻđäÿĂäÿłè£■äžčåŹĺåŕźèśąïijÑäÿćåijČåŐ§æIJL'åžŘåĹŮäÿ■çŻť'åĹřåĞ¡æŢřè£ŤåŻđTrueäźŃåL'■ç

äÿžäžĘæijŤçď'žïijÑåĄĞåőŽä¡ăåIJĺèŕżåŔŰäÿĂäÿłåijĂåğŃéČĺåĹĘæŸŕåĞäèąŇæşĺéĞŁçŽĎæžŘæŰĞäżűã

```
>>> with open('/etc/passwd') as f:
... for line in f:
...     print(line, end='')
...
##
# User Database
#
# Note that this file is consulted directly only when the system is␣
↪running
# in single-user mode. At other times, this information is provided␣
↪by
# Open Directory.
...
##
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
...
>>>
```

åęĆæđIJä¡ăæČşèůşè£ĞåijĂåğŃéČĺåĹĘçŽĎæşĺéĞŁèąŇçŽĎèŕİïijÑåŔŕäžěè£ŹæăůåĄŽïijŽ

```
>>> from itertools import dropwhile
>>> with open('/etc/passwd') as f:
...     for line in dropwhile(lambda line: line.startswith('#'), f):
...         print(line, end='')
...
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
...
>>>
```

è£Źäÿłä¿Ńå■ŘæŸŕå§žäžŐæăźæ■őæ§ŘäÿłæţŃèŕŢåĞ¡æŢřèůşè£ĞåijĂåğŃçŽĎåĚČçť'ăăĂĆ åęĆæđIJä¡ăåůšçżŔæŸŐçąőç§ěéĄŞäžĘèęĄèůşè£ĞçŽĎåĚČçť'ăçŽĎäÿłæŢřçŽĎèŕİïijŃéĆčäźĹåŔŕäžěä¡£çŤĺ itertools.islice() æİěäžčæŻ£ãĂĆærŤåęĆïijŽ

```
>>> from itertools import islice
>>> items = ['a', 'b', 'c', 1, 4, 10, 15]
>>> for x in islice(items, 3, None):
...     print(x)
...
1
4
10
15
>>>
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇ islice() åĞ¡æŢřæIJĂåŘŐéĆčäÿł None åŔĆæŢřæŇĞåőŽäžĘä¡ăèęAèŐůåŔŰäżŐçňň3äÿłåĹřæIJĂåŘŐçŽĎæL'ĂæIJL'åĚČçť ăïijŇ åęĆæđIJ None åŠŇ3çŽĎä¡■ç¡őåŕźèřČïijŇæĎŔæĂİåřśæŸŕäžĚäžĚèŐůåŔŰåL'■äÿL'äÿłåĚČçť ăæĄřæĄřçŻÿåŔ (è£Źäÿłèů§åĹĞçL'ĞçŽĎçŻÿåŔ■æŞ■ä¡IJ [3:] åŠŇ [:3] åŐ§çŘĘæŸŕäÿĂæăůçŽĎ)ãĂĆ

èőÍèőž

åĞ¡æŢř dropwhile() åŠŇ islice() åĚűåőđåřśæŸŕäÿď äÿłåÿőåŁľåĞ¡æŢřïijŇäÿžçŽĎåřśæŸŕéĄ£å

```
with open('/etc/passwd') as f:
    # Skip over initial comments
    while True:
        line = next(f, '')
        if not line.startswith('#'):
            break

    # Process remaining lines
    while line:
        # Replace with useful processing
        print(line, end='')
        line = next(f, None)
```

èůşè£ĞäÿĂäÿłåŔŕè£■äżčåŕźèśąçŽĎåijĂåğŃéČĺåĹĘèů§éĂŽåÿÿçŽĎè£Ğæżď æŸŕäÿ■åŘŇçŽĎãĂĆ æŕŤåęĆïijŇäÿŁè£řäżčçăĄçŽĎçňňäÿĂäÿłéČĺåĹĘåŔŕèČ¡äijŽè£ŹæăůéĞ■åĘŹïijŽ

```
with open('/etc/passwd') as f:
    lines = (line for line in f if not line.startswith('#'))
    for line in lines:
        print(line, end='')
```

è£ŹæăůåEŹçąőåőđåŔŕäžěèůşè£ĞåijĂåğŃéČĺåĹĘçŽĎæşĺéĞŁèąŇïijŇä¡ĘæŸŕåŘŇæăůäź§äijŽèůşè£ĞæŰ æ■ćåŘĕèŕĺèőšïijŇæĹŠäžňçŽĎèğčåĘşæŰźæąĹæŸŕäžĚäžĚèůşè£ĞåijĂåğŃéČĺåĹĘæżąèűşæţŃèŕŢæĺąäżűçŽĎè

æIJĂåŘŐéIJĂèęAçİĂéĞ■åijžèřČçŽĎäÿĂçĆźæŸŕïijŇæIJňèĹĆçŽĎæŰźæąĹéĂĆçŤĺäžŐæL'ĂæIJL'åŔŕè£ æŕŤåęĆçŤ§æĹŘåŹĺïijŇæŰĞäżűåŔŁåĚűçśżäijijçŽĎåŕźèśąãĂĆ

4.9 æŐŠåĹŮçżĎåŘĹçŽĎè£■äžč

éŮőécŸ

ä¡ăæČşè£■äžčéĄ■åŐĘäÿĂäÿłéŻĘåŘĹäÿ■åĚČçť ăçŽĎæĽĂæIJL'åŕřèČ¡çŽĎæŐŠåĹŮæĹŰçżĎåŘĹ

èğčåĘşæŰźæąĹ

itertoolsæĺąåİŮæŔŘä¿ŻäžĘäÿĽäÿłåĞ¡æŢřæİěèğčåĘşè£ŹçśžéŮőécŸãĂĆ
åĚűäÿ■äÿĂäÿłæŸŕ itertools.permutations() ïijŇ
åőČæŐěåŘŮäÿĂäÿłéŻĘåŘĹåźűäžğçŢ§äÿĂäÿłåĚČçżĎåžŔåĹŮïijŇæŕŔäÿłåĚČçżĎçŢśéŻĘåŘĹäÿ■æĽĂæIJL'
äź§åřśæŸřèřťéĂŽè£ĞæĽŞäźśéŻĘåŘĹäÿ■åĚČçť ăæŐŠåĹŮéąžåžŔçŢ§æĹŘäÿĂäÿłåĚČçżĎïijŇæŕŤåęĆïijŽ

```
>>> items = ['a', 'b', 'c']
>>> from itertools import permutations
>>> for p in permutations(items):
...     print(p)
...
('a', 'b', 'c')
('a', 'c', 'b')
('b', 'a', 'c')
('b', 'c', 'a')
('c', 'a', 'b')
('c', 'b', 'a')
>>>
```

åęĆæđIJä¡ăæČşå¿ŮåĹřæŇĞåőŽéŢ£åžęçŽĎæĽĂæIJL'æŐŠåĹŮïijŇä¡ääŔŕäžěäijăéĂŠäÿĂäÿłåŔŕéĂĽ'çŽ

```
>>> for p in permutations(items, 2):
...     print(p)
...
('a', 'b')
('a', 'c')
('b', 'a')
('b', 'c')
('c', 'a')
('c', 'b')
>>>
```

ä¡£çŤÍ itertools.combinations() åŔřå¿ŮåĹřè¿ŞåĚěéŻĘåŘĹäÿ■åĚČçť ăçŽĎæĽĂæIJL'çŽĎçżĎ

```
>>> from itertools import combinations
>>> for c in combinations(items, 3):
...     print(c)
...
('a', 'b', 'c')

>>> for c in combinations(items, 2):
```

```
...     print(c)
...
('a', 'b')
('a', 'c')
('b', 'c')

>>> for c in combinations(items, 1):
...     print(c)
...
('a',)
('b',)
('c',)
>>>
```

åŕźäžŐ `combinations()` æİěèőšïijŇåĚČçť ăçŽĎéąžåžŔåůšçżŔäÿ■éĞ■èęĄäžĘãĂĆäź§åřśæŸŕèŕťïijŇçżĎåŘĹ `('a', 'b')` èů§ `('b', 'a')` åĚűåőđæŸŕäÿĂæăůçŽĎ(æIJĂçźĹåŔłäijŽè¿ŞåĞžåĚűäÿ■äÿĂäÿł)ãĂĆ

åIJĺèőąçőŮçżĎåŘĹçŽĎæŮűåĂŹïijŇäÿĂæŮęåĚČçť ăèćńéĂĽåŔŰåřśäijŽäżŐåĂŹéĂĽäÿ■åĽŤéŹđ'æŐĽèĂŇåĞ¡æŢř `itertools.combinations_with_replacement()` åĚĄèőÿåŘŇäÿĂäÿłåĚČçť ăèćńéĂĽæŃľåđ'ŽæňąïijŇærŤåęĆïijŽ

```
>>> for c in combinations_with_replacement(items, 3):
...     print(c)
...
('a', 'a', 'a')
('a', 'a', 'b')
('a', 'a', 'c')
('a', 'b', 'b')
('a', 'b', 'c')
('a', 'c', 'c')
('b', 'b', 'b')
('b', 'b', 'c')
('b', 'c', 'c')
('c', 'c', 'c')
>>>
```

èőĺèőž

è£ŹäÿĂåřŔèŁĆæĹŚäżňåŘŚä¡ăåśŢçđ'žçŽĎäżĔäżĔæŸŕ `itertools` æĺąåİŮçŽĎäÿĂéČĺåĹĘåŁ§èČ¡ãĂĆ år¡çőąä¡ăäź§åŔŕäžèèĞłåůśæĽŃåŁĺåőđçŐřæŐŠåĹŮçżĎåŘĹçőŮæşŢïijŇä
å¡ŞæĹŚäżňçćřåĹřçIJŃäÿŁåŐżæIJĽäžŽåđ'■æİĆçŽĎè£■äžčéŮőécŸæŮűïijŇæIJĂåě¡åŔŕäžěåĚĹåŐżçIJŃçIJŃi
åęĆæđIJè£ŹäÿłéŮőécŸå¿ĹæŹőéĄ■ïijŇéĆčäźĹå¿ĹæIJĽåŔŕèČ¡äijŽåIJĺéĞŇéİćæĽ¿åĹřèğčåĘşæŰźæąĹïijĄ

4.10 åžŕåĹŮäÿŁçť ćåijŢåĂijè£■äžč

éŮőécŸ

ä¡äæČşåIJÍè£■äžčäÿĂäÿłåžŕåĹŮçŽĎåŘŇæŮűèů§èÿłæ■čåIJÍècńåď ĎçŘĘçŽĎåĚČçť ăçť ćåijŢãĂĆ

èğčåĘşæŰźæąĹ

åĘĚç¡őçŽĎ `enumerate()` åĞ¡æŢřåŕŕäžěå¿Ĺåě¡çŽĎèğčåĘşè£ŹäÿłéŮőécŸïijŽ

```
>>> my_list = ['a', 'b', 'c']
>>> for idx, val in enumerate(my_list):
...     print(idx, val)
...
0 a
1 b
2 c
```

äÿžäžĘæŇĽäijăçż§èąŇåŕůè¿ŞåĞž(èąŇåŕůäżŐ1åijĂåğŃ)ïijŇä¡ăåŕŕäžěäijăéĂŠäÿĂäÿłåijĂåğŃåŕĆæŢř

```
>>> my_list = ['a', 'b', 'c']
>>> for idx, val in enumerate(my_list, 1):
...     print(idx, val)
...
1 a
2 b
3 c
```

è£Źçğ■æČĚåĘţåIJÍä¡ăéA■åŐĘæŰĞäžűæŮűæČşåIJÍéŤŹèŕŕæűĹæĄŕäÿ■ä¡£çŤÍèąŇåŕůåőŽä¡■æŮűåĂŹéĹ

```
def parse_data(filename):
    with open(filename, 'rt') as f:
        for lineno, line in enumerate(f, 1):
            fields = line.split()
            try:
                count = int(fields[1])
                ...
            except ValueError as e:
                print('Line {}: Parse error: {}'.format(lineno, e))
```

`enumerate()` åŕźäžŐèů§èÿłæ§ŘäžŽåĂijåIJÍåĹŮèąÍäÿ■åĞžçŐřçŽĎä¡■ç¡őæŸŕå¿ĹæIJL'çŤÍçŽĎãĂĆ æL'ĂäžěïijŇåęĆædIJä¡ăæČşåřĘäÿĂäÿłæŰĞäžűäÿ■åĞžçŐřçŽĎå■Ţèŕ■æŸăåřĎåĹřåőČåĞžçŐřçŽĎèąŇåŕůäÿ `enumerate()` æİěåőŇæĹŘïijŽ

```
word_summary = defaultdict(list)

with open('myfile.txt', 'r') as f:
    lines = f.readlines()
```

```
for idx, line in enumerate(lines):
    # Create a list of words in current line
    words = [w.strip().lower() for w in line.split()]
    for word in words:
        word_summary[word].append(idx)
```

åęĆæđIJä¡ăåd'ĎçŘĘåőŇæŰĞäżűåŘŐæĽ'Şå■ř word_summary ïijŇäijŽåŔŚçŐřåőČæŸŕäÿĂäÿłå■ŮåËÿ(åĞEçąőæİèèőšæŸŕäÿĂäÿł defaultdict)ïijŇ åŕźäžŐæŕŔäÿłå■Ţèŕ■æIJL'äÿĂäÿł key ïijŇæŕŔäÿł key åŕźåžŢçŽĎåĂijæŸŕäÿĂäÿłçŢśè£Źäÿłå■Ţèŕ■åĞžçŐřçŽĎèąŇåŘůçżĎæĹŘçŽĎåĹŮèąĺãĂĆ åęĆæđIJæ§Řäÿłå■Ţèŕ■åIJĺäÿĂèąŇäÿ■åĞžçŐřè£Ğäÿď'æňąïijŇéĆčäźĹè£Źäÿłèą ŇåŘůäź§äijŽåĞžçŐřäÿď'æňąï åŘŇæŮűäź§åŘŕäżěä¡IJäÿžæŰĞæIJňçŽĎäÿĂäÿłçőĂå■Ţçż§èőąãĂĆ

èőĺèőž

å¡Şä¡ăæČşéćİåď'ŰåőŽäźL'äÿĂäÿłèőąæŢřåŔŸéĞŔçŽĎæŮűåĂŹïijŇä¡£çŢĺ enumerate() åĞ¡æŢřäijŽæŻť'åŁăçőĂå■ŢãĂĆä¡ăåŔŕèČ¡äijŽåČŔäÿŃéİćè£ŹæăůåĘŹäżčçăĄïijŽ

```
lineno = 1
for line in f:
    # Process line
    ...
    lineno += 1
```

ä¡ĘæŸŕåęĆæđIJä¡£çŢĺ enumerate() åĞ¡æŢřæİěäžčæŻ£åřśæŸ¿å¿ŮæŻť'åŁăäijŸéŻĚäžĘïijŽ

```
for lineno, line in enumerate(f):
    # Process line
    ...
```

enumerate() åĞ¡æŢřè£ŤåŻđçŽĎæŸŕäÿĂäÿł enumerate åŕźèśąåőđä¿ŃïijŇ åőČæŸŕäÿĂäÿłè£■äżčåŹĺïijŇè£ŤåŻđè£đçż■çŽĎåŇĚåŘńäÿĂäÿłèőąæŢřåŠŇäÿĂäÿłåĂijçŽĎåĚČçżĎïijŇ åĚČçżĎäÿ■çŽĎåĂijéĂŽè£ĞåIJĺäijääĚěåžŔåĹŮäÿŁèřČçŢĺ next() è£ŤåŻđãĂĆ

è£ŸæIJL'äÿĂçĆźåŔŕèČ¡åźűäÿ■å¿ĹéĞ■èęĄïijŇä¡ĘæŸŕäź§åĂijå¿ŮæşĺæĎŔïijŇ æIJL'æŮűåĂŹå¡Şä¡ăåIJĺäÿĂäÿłåůšçżŔèğčåŐŇåŘŐçŽĎåĚČçżĎåžŔåĹŮäÿŁä¡£çŢĺ enumerate() åĞ¡æŢřæŮűå¿ĹåőźæŸŞèřČåĚěéŹůéŸśãĂĆ ä¡ăå¿ŮåČŔäÿŃéİćæ■čçąőçŽĎæŰźåijŔè£ŹæăůåĘŹïijŽ

```
data = [ (1, 2), (3, 4), (5, 6), (7, 8) ]

# Correct!
for n, (x, y) in enumerate(data):
    ...
# Error!
for n, x, y in enumerate(data):
    ...
```

4.11 åŘŇæŮűè£■äžčåď'Žäÿłåžŕ̊åĹŮ

éŮőécŸ

ä¡äæČşåŘŇæŮűè£■äžčåď'Žäÿłåžŕ̊åĹŮïijŇæŕŕÆňąåĹĘåĹńäżŐäÿĂäÿłåžŕ̊åĹŮäÿ■åŔŰäÿĂäÿłåĚČçť'ãã

èğčåĘşæŰźæąĹ

äÿžäžĘåŘŇæŮűè£■äžčåď'Žäÿłåžŕ̊åĹŮïijŇä¡£çŤÍ `zip()` åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> xpts = [1, 5, 4, 2, 10, 7]
>>> ypts = [101, 78, 37, 15, 62, 99]
>>> for x, y in zip(xpts, ypts):
...     print(x,y)
...
1 101
5 78
4 37
2 15
10 62
7 99
>>>
```

`zip(a, b)` äijŽçŤ§æĹŘäÿĂäÿłåŕŕè£ŤåŻđåĚČçżĎ `(x, y)` çŽĎè£■äžčåŹÍïijŇåĚűäÿ■xæİěèĞłaïijŇyæİěèĞłbãĂĆ äÿĂæŮęåĚűäÿ■æ§Řäÿłåžŕ̊åĹŮåĹřåžŢçżŞår¿ïijŇè£■äż
åŻăæ■ď'è£■äžčéŢ£åžęèů§åŔĆæŢřäÿ■æIJĂç§■åžŕ̊åĹŮéŢ£åžęäÿĂèĞť'ãĂĆ

```
>>> a = [1, 2, 3]
>>> b = ['w', 'x', 'y', 'z']
>>> for i in zip(a,b):
...     print(i)
...
(1, 'w')
(2, 'x')
(3, 'y')
>>>
```

åęĆæđIJè£Źäÿłäÿ■æŸŕä¡ăæČşèęĄçŽĎæŢĹæđIJïijŇéĆčäźĹè£Ÿåŕŕäžëä¡£çŤÍ `itertools.zip_longest()` åĞ¡æŢřæİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
>>> from itertools import zip_longest
>>> for i in zip_longest(a,b):
...     print(i)
...
(1, 'w')
(2, 'x')
(3, 'y')
(None, 'z')
```

```
>>> for i in zip_longest(a, b, fillvalue=0):
...     print(i)
...
(1, 'w')
(2, 'x')
(3, 'y')
(0, 'z')
>>>
```

èőÍèőž

å¡Şä¡ăæČşæĹŘåŕžåď˙ĎçŘĘæŢřæ■őçŽĎæŮűåĂŹ zip() åĞ¡æŢřæŸŕå¿ĹæIJL˙çŤÍçŽĎãĂĆ ærŤåęĆïijŇåĄĞèő¿ä¡ăåď˙ťåĹŮèąĺåŠŇäÿĂäÿłåĂijåĹŮèąĺïijŇåŕšåČŔäÿŃéIc

```
headers = ['name', 'shares', 'price']
values = ['ACME', 100, 490.1]
```

ä¡£çŤÍzip()åŔŕäžěèőľä¡ăåřĘåőČäžňæĽ˙ŞåŇĚåźűçŤ§æĹŘäÿĂäÿłå■ŮåĚÿïijŽ

```
s = dict(zip(headers,values))
```

æĹŰèĂĚä¡ăäź§åŔŕäžěåČŔäÿŃéIćè£Źæăůäžğçŧ§è¿ŞåĞžïijŽ

```
for name, val in zip(headers, values):
    print(name, '=', val)
```

èŹ¡çĎűäÿ■åÿÿèğĄïijŇä¡ĘæŸŕ zip() åŔŕäžěæŐěåŔŮåď˙ŽäžŐäÿď˙äÿłçŽĎåžŔåĹŮçŽĎåŔĆæŢřãĂĆ è£ŹæŮűåĂŹæĽ˙ĂçŤ§æĹŘçŽĎçżŞæđIJåĚČçżĎäÿ■åĚČçţ˙ääÿłæŢřèů§è¿ŞåĚěåžŔåĹŮäÿłæŢřäÿĂæăůãĂĆærŤ

```
>>> a = [1, 2, 3]
>>> b = [10, 11, 12]
>>> c = ['x','y','z']
>>> for i in zip(a, b, c):
...     print(i)
...
(1, 10, 'x')
(2, 11, 'y')
(3, 12, 'z')
>>>
```

æIJĂåŘŐåijžèřČäÿĂçĆźåřśæŸŕïijŇ zip() äijŽåĹŻåžžäÿĂäÿłè£■äžčåŹĺæÍěä¡IJäÿžçżŞæđIJè£ŤåŻđãĂĆ åęĆæđIJä¡ăéIJĂèęĄåřĘçżŞåŕźçŽĎåĂijå■ŸåĆĺåIJĺåĹŮèąĺäÿ■ïijŇèęĄä¡£çŤÍ list() åĞ¡æŢřãĂĆærŤåęĆïijŽ

```
>>> zip(a, b)
<zip object at 0x1007001b8>
>>> list(zip(a, b))
```

```
[(1, 10), (2, 11), (3, 12)]
>>>
```

4.12 äÿ■åŘŇéŻĘåŘĹäÿŁåĚČçťäçŽĎè£■äžč

éŮőécŸ

ä¡ăæČşåIJĺåď'ŽäÿłåŕźèśąæL'ğèąŇçŻÿåŘŇçŽĎæŞ■ä¡IJïijŇä¡ĘæŸŕè£ŹäžŻåŕźèśąåIJĺäÿ■åŘŇçŽĎåőźåŹĺä

èğčåĘşæŰźæąĹ

itertools.chain() æŰźæşŢåŔŕäžěçŤĺæİěçőĂåŇŰè£ŹäÿłäżżåŁąãĂĆ åőČæŐěåŘŮäÿĂäÿłåŔŕè£■äžčåŕźèśąåĹŮèąĺä¡IJäÿžè¿ŞåĚčïijŇåźűè£ŤåŻđäÿĂäÿłè£■äžčåŹĺïijŇæIJL'æŢĹçŽĎ äÿžäžĘæijŤçd'žæÿĚæěŽïijŇèĂČèŹŚäÿŇéİćè£ŹäÿłäżŃå■ŘïijŽ

```
>>> from itertools import chain
>>> a = [1, 2, 3, 4]
>>> b = ['x', 'y', 'z']
>>> for x in chain(a, b):
... print(x)
...
1
2
3
4
x
y
z
>>>
```

ä¡£çŤĺ chain() çŽĎäÿĂäÿłåÿÿèğĄåIJžæŹŕæŸŕå¡Şä¡ăæČşåŕźäÿ■åŘŇçŽĎéŻĘåŘĹäÿ■æL'ĂæIJL'åĚČç

```
# Various working sets of items
active_items = set()
inactive_items = set()

# Iterate over all items
for item in chain(active_items, inactive_items):
    # Process item
```

è£Źçğ■èğčåĘşæŰźæąĹèęĄærŤåČŔäÿŇéİćè£Źæăůä¡£çŤĺäÿd'äÿłå■ŢçŇňçŽĎå¿łçŐŕæŻt'åŁääijŸéŻĚïijŇ

```
for item in active_items:
    # Process item
    ...
```

```
for item in inactive_items:
    # Process item
    ...
```

èőÍèőž

itertools.chain() æŐěåŔŮäÿĂäÿłæĹŰåď˙ŽäÿłåŔŕè£■äžčåŕźèśąæIJĂäÿžè¿ŞåĚěåŔĆæŢřãĂĆ çĎűåŘŐåĹŻåžžäÿĂäÿłè£■äžčåŹĺïijÑä¿İæňąè£đçż■çŽĎè£ŤåŻđæŕŔäÿłåŔŕè£■äžčåŕźèśąäÿ■çŽĎåĚČçť˙ãăĂĆ è£Źçğ■æŰźåijŔèęĄæŕŤåĚĹåřĘåžŔåĹŮåŘĹåźűåĘ■è£■äžčèęĄénŸæŢĹçŽĎåď˙ŽãĂĆæŕŤåęĆïijŽ

```
# Inefficent
for x in a + b:
    ...

# Better
for x in chain(a, b):
    ...
```

çňňäÿĂçğ■æŰźæąĹäÿ■ïijÑ a + b æŞ■ä¡IJäijŽåĹŻåžžäÿĂäÿłåĚĺæŰřçŽĎåžŔåĹŮåźűèęĄæśĆaåŠÑbçŽ chian() äÿ■äijŽæIJL˙è£ŹäÿĂæ■ěïijÑæĽ˙ĂäžěåęĆæđIJè¿ŞåĚěåžŔåĹŮéIđåÿÿåď˙ğçŽĎæŮűåĂŹäijŽå¿ĹçIJA åźűäÿŤå¡ŞåŔŕè£■äžčåŕźèśąçśžåđÑäÿ■äÿĂæăůçŽĎæŮűåĂŹ chain() åŘÑæăůåŔŕäžěå¿Ĺåě¡çŽĎåůěä¡IJãĂĆ

4.13 åĹŻåžžæŢřæ■őåď˙ĎçŘĘçőąéĄŞ

éŮőécŸ

ä¡ăæČşäžěæŢřæ■őçőąéĄŞ(çśžäijijUnixçőąéĄŞ)çŽĎæŰźåijŔè£■äžčåď˙ĎçŘĘæŢřæ■őãĂĆ æŕŤåęĆïijÑä¡ăæIJL˙äÿłåď˙ğéĞŔçŽĎæŢřæ■őéIJĂèęĄåď˙ĎçŘĘïijÑä¡ĘæŸŕäÿ■èČ¡åřĘåőČäžñäÿĂæñąæĂğæŤ¿

èğčåĘşæŰźæąĹ

çŤ§æĹŘåŹĺåĞ¡æŢřæŸŕäÿĂäÿłåőđçŐřçőąéĄŞæIJžåĹűçŽĎåě¡åŁđæşŢãĂĆ äÿžäžĘæijŤçď˙žïijÑåĄĞåőŽä¡äèęĄåď˙ĎçŘĘäÿĂäÿłéIđåÿÿåď˙ğçŽĎæŮěå£ŮæŰĞäžűçŻőå¡ŢïijŽ

```
foo/
    access-log-012007.gz
    access-log-022007.gz
    access-log-032007.gz
    ...
    access-log-012008
bar/
    access-log-092007.bz2
```

```
    ...
    access-log-022008
```

åĄĞèő¿æŕŕäÿłæŮěå£ŮæŰĞäžűåŇĚåŘńè£ŹæăůçŽĎæŢřæ■őïijŽ

```
124.115.6.12 - - [10/Jul/2012:00:18:50 -0500] "GET /robots.txt ..."␣
↪200 71
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /ply/ ..." 200␣
↪11875
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /favicon.ico ..
↪." 404 369
61.135.216.105 - - [10/Jul/2012:00:20:04 -0500] "GET /blog/atom.xml␣
↪..." 304 -
...
```

äÿžäžĘåď˙ĎçŘĘè£ŹäžŽæŰĞäžűïijŇä¡åŔŕäžěåőŽäźĽäÿĂäÿłçŤśåď˙ŽäÿłæĽğèąŇçĽźåőŽäżżåŁąçŃňçńŇ

```
import os
import fnmatch
import gzip
import bz2
import re

def gen_find(filepat, top):
    '''
    Find all filenames in a directory tree that match a shell␣
↪wildcard pattern
    '''
    for path, dirlist, filelist in os.walk(top):
        for name in fnmatch.filter(filelist, filepat):
            yield os.path.join(path,name)

def gen_opener(filenames):
    '''
    Open a sequence of filenames one at a time producing a file␣
↪object.
    The file is closed immediately when proceeding to the next␣
↪iteration.
    '''
    for filename in filenames:
        if filename.endswith('.gz'):
            f = gzip.open(filename, 'rt')
        elif filename.endswith('.bz2'):
            f = bz2.open(filename, 'rt')
        else:
            f = open(filename, 'rt')
        yield f
        f.close()

def gen_concatenate(iterators):
    '''
```

```
    Chain a sequence of iterators together into a single sequence.
    '''
    for it in iterators:
        yield from it

def gen_grep(pattern, lines):
    '''
    Look for a regex pattern in a sequence of lines
    '''
    pat = re.compile(pattern)
    for line in lines:
        if pat.search(line):
            yield line
```

çŐřåIJĺä¡ăåŔŕäžěå¿Ĺåőźæ ŸŞçŽĎåřĘè£ŹäžŻåĞ¡æŢřè£đèţůæİěåĹŻåžžäÿĂäÿłåď ĎçŘĘçőąéĄŞãĂĆ
æŕŤåęĆïijŇäÿžäžĘæ§ěæL'¿åŇĚåŘńå■Ţèŕ■pythonçŽĎæL'ĂæIJL'æŮěå£ŮèąŇïijŇä¡ăåŔŕäžěè£ŹæăůåĄŽïijŽ

```
lognames = gen_find('access-log*', 'www')
files = gen_opener(lognames)
lines = gen_concatenate(files)
pylines = gen_grep('(?i)python', lines)
for line in pylines:
    print(line)
```

åęĆæđIJåřĘæİěçŽĎæŮűůåĂŹä¡ăæČşæL'l'åśŢçőąéĄŞïijŇä¡ăçŤŽèĞşåŔŕäžěåIJĺçŤ§æĹŘåŹĺèąĺè¿¿åijŔäÿ■
æŕŤåęĆïijŇäÿŃéİćè£ŹäÿłçL'ĹæIJňèőąçőŮåĞžäijăè¿ŞçŽĎå■ŮèŁĆæŢřåźűèőąçőŮåĚűæĂžåŠŇãĂĆ

```
lognames = gen_find('access-log*', 'www')
files = gen_opener(lognames)
lines = gen_concatenate(files)
pylines = gen_grep('(?i)python', lines)
bytecolumn = (line.rsplit(None,1)[1] for line in pylines)
bytes = (int(x) for x in bytecolumn if x != '-')
print('Total', sum(bytes))
```

èőĺèőž

äžěçőąéĄŞæŰźåijŔåď ĎçŘĘæŢřæ■őåŔŕäžěçŤĺæİèèğčåĘşåŘĎçśžåĚűäžŰéŮőéćŸïijŇåŇĚæŃňèğčæđŘïi

äÿžäžĘçŘĘèğčäÿŁè£řäžčçăĄïijŇéĞ■çĆźæŸŕèęĄæŸŐçŹ¡ `yield`
èŕ■åŔëä¡IJäÿžæŢřæ■őçŽĎçŤ§äžğèĂĚèĂŇ `for` å¿łçŐŕèŕ■åŔëä¡IJäÿžæŢřæ■őçŽĎæűĹèt'źèĂĚãĂĆ
å¡Şè£ŹäžŻçŤ§æĹŘåŹĺèćńè£đåIJĺäÿĂètůåŘŐïijŇærŔäÿł `yield`
äijŽåřĘäÿĂäÿłå■ŢçŃňçŽĎæŢřæ■őåĚČçt'ăäijăéĂŠçźŹè£■äžčåď ĎçŘĘçőąéĄŞçŽĎäÿŃäÿĂéŸűæőţãĂĆ
åIJĺä¿Ńå■ŘæIJĂåŘŐéČĺåĹĘïijŇ `sum()` åĞ¡æŢřæŸŕæIJĂçźĹçŽĎçĺŃäžŔél'šåŁĺèĂĚïijŇærŕŔæňąäžŐçŤ§æĹŘ

è£ŹçğæŰźåijŔäÿĂäÿłéİđåÿÿåě¡çŽĎçL'źçĆźæŸŕæŕŔäÿłçŤ§æĹŘåŹĺåĞ¡æŢřå¿ĹåřŔåźűäÿŤéČ¡æŸŕçŃňç
å¿Ĺåď ŽæŮűůåĂŹïijŇè£ŹäžŻåĞ¡æŢřåęĆæđIJærŤè¿ČéĂŽçŤĺçŽĎèŕĺåŔŕäžěåIJĺåĚűäžŰåIJžæŹŕéĞ■åď ■ä¡£çŤĺ
åźűäÿŤæIJĂçźĹåřĘè£ŹäžŻçżĎäžűçżĎåŘĹèţůæİěçŽĎäžčçăĄçIJŃäÿŁåŐżéİđåÿÿçőĂå■ŢïijŇäź§å¿Ĺåőźæ ŸŞç

ä¡£çŤĺè£ŹçğæŰźåijŔçŽĎåĘĚå■ŸæŢĹçŐĞäź§äÿ■å¿Ůäÿ■æŔŘãĂĆäÿŁè£řäžčçăĄå■şä¿£æŸŕåIJĺäÿĂäÿ
äžŃåŏđäÿŁïijŇçŤśäžŐä¡£çŤĺäžĘè£■äžčæŰźåijŔåď ĎçŘĘïijŇäžčçăĄè£ŘèąŇè£ĞçĺŃäÿ■åŔłéIJĂèęĄå¿ĹåřŔå

åIJĺèřČçŤĺ `gen_concatenate()` åĞ¡æŢřçŽĎæŮűåĂŹä¡ääŔŕèČ¡äijŽæIJL'äžŻäÿ■åď łæŸŐçŹ¡ãĂĆ
è£ŹäÿłåĞ¡æŢřçŽĎçŽőçŽĎæŸŕåřĘè¿ŞåĚěåžŘåĹŮæŃijæŐěæĹŘäÿĂäÿłå¿ĹéŢ£çŽĎèąŇåžŘåĹŮãĂĆ
`itertools.chain()` åĞ¡æŢřåŘŇæăůæIJL'çśżäijijçŽĎåŁ§èČ¡ïijŇä¡ĘæŸŕåőČéIJĂèęĄåřĘæL'ĂæIJL'åŔŕè
åIJĺäÿŁéİćè£ŹäÿłäżŃå■Řäÿ■ïijŇä¡ääŔŕèČ¡äijŽåĘŹçśżäijijè£ŹæăůçŽĎèŕ■åŔě

```
lines = itertools.chain(*files)
```

ïijŇ ä¡£å¿Ů `gen_opener()` çŤ§æĹŘåŹĺèČ¡è¢ńåĚĺéČĺæűĹèťźæŐL'ãĂĆ ä¡ĘçŤśäžŐ `gen_opener()` çŤ§æĹŘåŹĺæŕŔæňąçŤ§æĹŘäÿĂäÿłæL'ŞåijĂè£ĞçŽĎæŰĞäžűïijŇ ç■L'åĹřäÿŃäÿĂäÿłè£■äžčæ■ěéłď æŮűæŰĞäžűůřśåĚşéŮ■äžĘïijŇåŽăæ■ď `china()` åIJĺè£ŹéĞŇäÿ■èČ¡è£Źæăůä¡£çŤĺãĂĆ äÿŁéİćçŽĎæŰźæąĹåŔŕäžěéĄ£åĚ■è£Źçğ■æČĚåĘţãĂĆ

`gen_concatenate()` åĞ¡æŢřäÿ■åĞžçŐřè£Ğ `yield from` èŕ■åŔěijŇåőČåřĘ `yield` æŞ■ä¡IJäžčçŘĘåĹřçĹűçŤ§æĹŘåŹĺäÿŁåŐžãĂĆ èŕ■åŔě `yield from` `it` çőĂå■ŢçŽĎè£ŤåŽđçŤ§æĹŘåŹĺ `it` æL'Ăäžğç Ť§çŽĎæL'ĂæIJL'åĂijãĂĆ
åĚşäžŐè£ŹäÿłæĹŚäžňåIJĺ4.14åřŔèŁĆäijŽæIJL'æŻť'è£ŻäÿĂæ■ěçŽĎæŔŔè£řãĂĆ

æIJĂåŘŐè£ŸæIJL'äÿĂçĆźéIJĂèęĄæşĺæĎŔçŽĎæŸŕijŇçőąéĄŞæŰźåijŔåźűäÿ■æŸŕäÿĞèČ¡çŽĎãĂĆ
æIJL'æŮűåĂŹä¡äæČşçńŇå■şåď ĎçŘĘæL'ĂæIJL'æŢřæ■őãĂĆ çĎűèĂŃïijŇå■şä¿£æŸŕè£ŹçğæČĚåĘţïijŇä¡£ç

David Beazley åIJĺäžŰçŽĎ Generator Tricks for Systems Programmers
æŢŹçĺŃäÿ■åŕźäžŐè£ŹçğæŁĂæIJŕæIJL'éİđåÿÿæůśåĚěçŽĎèőšèğčãĂĆåŔŕäžěåŔĆèĂČè£ŹäÿłæŢŹçĺŃèŐůåŔ

4.14 åśŢåijĂåţŇåěŮçŽĎåžŔåĹŮ

éŮőécŸ

ä¡äæČşåřĘäÿĂäÿłåď ŽåśĆåţŇåěŮçŽĎåžŔåĹŮåśŢåijĂæĹŘäÿĂäÿłå■ŢåśĆåĹŮèąĺ

èğčåĘşæŰźæąĹ

åŔŕäžěåĘŹäÿĂäÿłåŇĚåŘń `yield from` èŕ■åŘěçŽĎéĂŠå¡ŠçŤ§æĹŘåŹĺæİěè¡żæİ¿èğčåĘşè£ŹäÿłéŮőéć

```
from collections import Iterable

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_
↪types):
            yield from flatten(x)
        else:
            yield x

items = [1, 2, [3, 4, [5, 6], 7], 8]
# Produces 1 2 3 4 5 6 7 8
for x in flatten(items):
    print(x)
```

åIJÍäÿŁéİćäžčçăĄäÿ■ïijŇ `isinstance(x, Iterable)` æčĂæ§ěæ§ŘäÿłåĚČçť˘ăæŸŕåŘęæŸŕåŘŕè£■äžčçŽĎãĂĆ åęĆæđIJæŸŕçŽĎèŕİïijŇ `yield from` åřśäijŽè£ŤåŽđæĽĂæIJĽå■Řä¿ŃçÍŃçŽĎåĂijãĂĆæIJĂçżĹè£ŤåŽđçżŞæđIJåřśæŸŕäÿĂäÿłæšąæIJĽåţŇå

éćİåđ˘ŰçŽĎåŔĆæŢř `ignore_types` åŠŇæčĂæţŃèŕ■åŘě `isinstance(x, ignore_types)` çŤÍæİěåřĘå■Ůçňęäÿšå ŠŇå■ŮèŁĆæŐŠéŹđ˘åIJÍåŔŕè£■äžčåŕźèśąåđ˘ŰïijŇéŸšæ■ćåřĘåőČäz è£ŹæăůçŽĎèŕİå■ŮçňęäÿšæŢřçżĎåřśèČ¡æIJĂçźĹè£ŤåŽđæĹŚäžňæĽĂæIJ§æIJŻçŽĎçżŞæđIJäžĘãĂĆæŕŤåęĆ

```
>>> items = ['Dave', 'Paula', ['Thomas', 'Lewis']]
>>> for x in flatten(items):
...     print(x)
...
Dave
Paula
Thomas
Lewis
>>>
```

èőÍèőž

èŕ■åŘě `yield from` åIJÍä¡äæČşåIJÍçŤ§æĹŘåŹÍäÿ■èřČçŤÍåĚűäžŰçŤ§æĹŘåŹÍä¡IJäÿžå■Řä¿ŃçÍŃçŽĎæ åęĆæđIJä¡ääÿ■ä¡£çŤÍåőČçŽĎèŕİïijŇéĆčäźĹåřśå£Ěéąžå ĘŹéćÍåđ˘ŰçŽĎ `for` å¿łçŐŕäžĘãĂĆæŕŤåęĆïijŽ

```
def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_
↪types):
            for i in flatten(x):
                yield i
        else:
            yield x
```

åř¡çőąåŔłæŤźäžĘäÿĂçĆźçĆźïijŇä¡ĘæŸŕ `yield from` èŕ■åŘěçIJŃäÿŁåŐżæĎ§èğĽæŽť˘åě¡ïijŇåźűäÿŤäź§ä¡£å¿Ůäžčçă ĄæŽť˘çőĂæť˘Ąæÿ ĚçĹ¡ãĂĆ

äźŃåĽ■æŔŘåĹřçŽĎåŕźäžŐå■Ůçňęäÿšå ŠŇå■ŮèŁĆçŽĎéćÍåđ˘ŰæčĂæ§ěæŸŕäÿžäžĘéŸšæ■ćåřĘåőČäžňå åęĆæđIJè£ŸæIJĽåĚűäžŰä¡ääÿ■æČşåśŢåijĂçŽĎçśžåđŃïijŇä£őæŤźåŔĆæŢř `ignore_types` å■şåŔŕãĂĆ

æIJĂåŘŐèęĄæşĺæĎŔçŽĎäÿĂçĆźæŸŕïijŇ `yield from` åIJĺæűĽ˘åŔŁåĹřå§žäžŐå■ŔçÍŃåŠŇçŤ§æĹŘåŹÍçŽĎåźűåŔŠçijŰçÍŃäÿ■æĽ˘őæijŤçİĂæŽť˘åŁäéĞ■èęĄçŽĎèğŠè åŔŕäžěåŔĆèĂČ12.12åřŔèŁĆæ§ěçIJŃåŘęåđ˘ŰäÿĂäÿłä¿Ńå■ŘãĂĆ

4.15 éąžåžŕè£■äžčåŘĹåźűåŘŐçŽĎæŐŠåžŕè£■äžčåŕźèšą

éŮőécŸ

ä¡ăæIJL'äÿĂçşżåĹŮæŐŠåžŔåžŔåĹŮïijŇæČşåřĘåőČäżňåŘĹåźűåŘŐå¿ŮåĹřäÿĂäÿłæŐŠåžŔåžŔåĹŮåźűå

èğčåĘşæŰźæąĹ

heapq.merge() åĞ¡æŢřåŕŕäžěåÿőä¡äèğčåĘşè£ŹäÿłéŮőécŸãĂĆærŤåęĆïijŽ

```
>>> import heapq
>>> a = [1, 4, 7, 10]
>>> b = [2, 5, 6, 11]
>>> for c in heapq.merge(a, b):
...     print(c)
...
1
2
4
5
6
7
10
11
```

èőĺèőž

heapq.merge åŔŕè£■äžčçL'źæĂğæĎŔåŠşçİĂåőČäÿ■äijŽçńÑél'ňèŕżåŔŰæL'ĂæIJL'åžŔåĹŮãĂĆ
è£ŹåřśæĎŔåŠşçİĂä¡ăåŔŕäžěåIJĺéİđåÿÿéŢ£çŽĎåžŔåĹŮäÿ■ä¡£çŤĺåőČïijŇèĂŇäÿ■äijŽæIJL'åd'łåd'ğçŽĎåijÄé
ærŤåęĆïijŇäÿÑéİćæŸŕäÿĂäÿłä¿Ńå■ŘæİěæijŤçd'žåęĆä¡ŢåŘĹåźűäÿd'äÿłæŐŠåžŘæŰĞäžűïijŽ

```
with open('sorted_file_1', 'rt') as file1, \
    open('sorted_file_2', 'rt') as file2, \
    open('merged_file', 'wt') as outf:

    for line in heapq.merge(file1, file2):
        outf.write(line)
```

æIJL'äÿĂçĆźèęĄåijžèřČçŽĎæŸŕ heapq.merge() éIJĂèęĄæL'ĂæIJL'è¿ŞåĚěåžŔåĹŮå£ĚéążæŸŕæŐŠ
çL'źåĹńçŽĎïijŇåőČåźűäÿ■äijŽécĎåĚĹèŕżåŔŰæL'ĂæIJL'æŢŕæ■őåĹřåäĘæăĹäÿ■æĹŰèĂĚécĎåĚĹæŐŠåžŔïij
åőČäžĚäżĚæŸŕæčĂæ§ěæL'ĂæIJL'åžŔåĹŮçŽĎåijĂåğŃéČĺåĹĘåźűè£ŤåŻđæIJĂåřŔçŽĎéĆčäÿłïijŇè£ŹäÿłèŁČ

4.16 è£■äžčåŹĺäžčæŻ£whileæŮåéŹŘå¿łçŐŕ

éŮőécŸ

ä¡ăåIJĺäžčçăĄäÿ■ä¡£çŤĺ while å¿łçŐŕæĭěè£■äžčåď ĎçŘĘæŢřæ■őïijŇåŻääÿžåőČéIJĂèęĄèřČçŤĺæ§Řäÿ èČ¡äÿ■èČ¡çŤĺè£■äžčåŹĺæĭééĞ■åĘŹè£Źäÿłå¿łçŐŕåŚćïij§

èğčåĘşæŰźæąĹ

äÿĂäÿłåÿÿèğĄçŽĎIOæŞ■ä¡IJçĺŇåžŔåŔŕèČ¡äijŽæČşäÿŇéİćè£Źæăůïij Ž

```
CHUNKSIZE = 8192

def reader(s):
    while True:
        data = s.recv(CHUNKSIZE)
        if data == b'':
            break
        process_data(data)
```

è£Źçğ■äžčçăĄéĂŽåÿÿåŔŕäžěä¡£çŤĺ iter() æĭěäžčæŻ£ïijŇåęĆäÿŇæĽĂçď žïijŽ

```
def reader2(s):
    for chunk in iter(lambda: s.recv(CHUNKSIZE), b''):
        pass
        # process_data(data)
```

åęĆæđIJä¡ăæĂĂçŰŠåőČåĹŕåžŢèČ¡äÿ■èČ¡æ■čåÿÿåůěä¡IJïijŇåŔŕäžěèŕŢéłŇäÿŇäÿĂäÿłçőĂå■ŢçŽĎä¿Ńå

```
>>> import sys
>>> f = open('/etc/passwd')
>>> for chunk in iter(lambda: f.read(10), ''):
...     n = sys.stdout.write(chunk)
...
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
daemon:*:1:1:System Services:/var/root:/usr/bin/false
_uucp:*:4:4:Unix to Unix Copy Protocol:/var/spool/uucp:/usr/sbin/
↪uucico
...
>>>
```

èőÍèőž

iter callable åĞ¡æŢřäÿĂäÿłéšIJäÿžäžžç§ěçŽĎçĽźæĂğæŸŕåőČæŐěåŔŮäÿĂäÿłåŔŕéĂĽçŽĎ åŕźèśąåŠŇäÿĂäÿłæăĞèőř(çżŞář¿)åĂijä¡IJäÿžè¿ŞåĚěåŔĆæŢřãĂĆ

å¡Şäżčè£ŹçğæŰźåijŔä¡£çŤĺçŽĎæŮűåĂŹïijŇåőČäijŽåĹŻåżžäÿĂäÿłè£äżčåŹĺïijŇ
è£ŹäÿłèäżčåŹĺäijŽäÿæŰèřČçŤĺ `callable` åŕźèśąçŻťåĹřè£ŤåŻđåĂijåŠŇæăĞèőřåĂijçŻÿç■L'äÿžæ■ćã

è£ŹçğçL'źæőŁçŽĎæŰźæşŢåŕźäžŐäÿĂäžŻçL'źåőŽçŽĎäijŽèćńéĞ■åď'■èřČçŤĺçŽĎåĞ¡æŢřå¿ĹæIJL'æŢĺ
äÿ¿ä¿ŃæİěèőšïijŇåęĆæđIJä¡ăæČşäżŐåěŮæŐěå■ŮæĹŰæŰĞäžűäÿ■äżěæŢřæ■őåİŮçŽĎæŰźåijŔèŕżåŔŰæŢřæ
`read()` æĹŰ `recv()` ïijŇ åźűåIJĺåŘŐéİćçť ğèů§äÿĂäÿłæŰĞäžűçżŞář¿æţŃèŕŢæİěåĘşåőŽæŸŕåŘęçżĹæ■ćã
`iter()` èřČçŤĺåřśåŔŕäžěåřĘäÿd'èĂĚçżŞåŘĹèţůæİěäžĘãĂĆ åĚűäÿ■ `lambda`
åĞ¡æŢřåŔĆæŢřæŸŕäÿžäžĘåĹŻåżžäÿĂäÿłæŮăåŔĆçŽĎ `callable` åŕźèśąïijŇåźűäÿž `recv`
æĹŰ `read()` æŰźæşŢæŔŘä¿ŻäžĘ `size` åŔĆæŢřãĂĆ

CHAPTER 7

çňňäžŤçńäïijŽæŰĞäżűäÿŐIO

æĽĂæIJL’çÍŃåžŔéČ¡èęĄåď’ĎçŘĘè¿ŞåĚěåŠŇè¿ŞåĞžãĂĆ
è£ŹäÿĂçńäářĘæűţçŻŰåď’ĎçŘĘäÿ■åŘŇçśžåđŃçŽĎæŰĞäżűïijŇåŇĚæŃňæŰĞæIJňåŠŇäžŇè£ŻåĹűæŰĞäżűï
åŕźæŰĞäżűåŘ■åŠŇçŻőå¡ŢçŽĎæŞ■ä¡IJäź§äijŽæűĽ’åŔŁåĹřãĂĆ

Contents:

5.1 èŕżåĘŹæŰĞæIJňæŢřæ■ő

éŮőécŸ

ä¡ăéIJĂèęĄèŕżåĘŹåŘĎçğ■äÿ■åŘŇçijŰçăĄçŽĎæŰĞæIJňæŢřæ■őïijŇæŕŤåęĆASCIIïijŇUTF-8æĹŰUTF-16çijŰçăĄç■L’ãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤĺåÿęæIJL’ `rt` æĺąåijŔçŽĎ `open()` åĞ¡æŢřèŕżåŔŰæŰĞæIJňæŰĞäżűãĂĆåęĆäÿŃæL’Ăçď’žïijŽ

```
# Read the entire file as a single string
with open('somefile.txt', 'rt') as f:
    data = f.read()

# Iterate over the lines of the file
with open('somefile.txt', 'rt') as f:
    for line in f:
        # process line
        ...
```

çśžäijijçŽĎïijŇäÿžäžĘåĘŹåĚěäÿĂäÿłæŰĞæIJňæŰĞäżűïijŇä¡£çŤĺåÿęæIJĽ wt æĺąåijŔçŽĎ open() åĞ¡æŢřïijŇ åęĆæđIJäźŃåĽ■æŰĞäżűåĘĚåőźå■ŸåIJĺåĹŹæÿĚéŹđ'åźűèęĘçŻŰæŐĽãĂĆ

```
# Write chunks of text data
with open('somefile.txt', 'wt') as f:
    f.write(text1)
    f.write(text2)
    ...

# Redirected print statement
with open('somefile.txt', 'wt') as f:
    print(line1, file=f)
    print(line2, file=f)
    ...
```

åęĆæđIJæŸŕåIJĺåůšå■ŸåIJĺæŰĞäżűäÿ■æůżåŁäåĘĚåőźïijŇä¡£çŤĺæĺąåijŔäÿž at çŽĎ open() åĞ¡æŢřãĂĆ

æŰĞäżűçŽĎèŕżåĘŹæŞ■ä¡IJéżŸèőđ'ä¡£çŤĺçşżçż§çijŰçăĄïijŇåŔŕäžěéĂŽè£ĞèřČçŤĺ sys.getdefaultencoding() æİěå¿ŮåĹřãĂĆ åIJĺåđ'ğåđ'ŽæŢřæIJžåŹĺäÿŁéİćéČ¡æŸŕutf-8çijŰçăĄãĂĆåęĆæđIJäj¡ääůšçżŔç§ěéĄŞä¡ãèęĄèŕżåĘŹçŽĎæŰĞæIJňæŸŕåĚűäżŰçijŰçăĄæŰźåijŔïijŇéĆčäźĹåŔŕäžěéĂŽè£Ğäijäé ĂŠäÿĂäÿłåŔŕéĂĽçŽĎ encoding åŘĆæŢřçżZopen()åĞ¡æŢřãĂĆåęĆäÿŃæĽĂçđ'žïijŽ

```
with open('somefile.txt', 'rt', encoding='latin-1') as f:
    ...
```

PythonæŤŕæŇĄéİđåÿÿåđ'ŽçŽĎæŰĞæIJňçijŰçăĄãĂĆåĞääÿłåÿÿèğĄçŽĎçijŰçăĄæŸŕascii, latin-1, utf-8åŠŇutf-16ãĂĆ åIJĺwebåžŤçŤĺçÍŇåžŔäÿ■éĂŽåÿÿéČ¡ä¡£çŤĺçŽĎæŸŕUTF-8ãĂĆ asciiåŕźåžŤäžŐU+0000åĹřU+007FèŇČåŻť'åĘĚçŽĎ7ä¡■å■Ůçňęã ĂĆ latin-1æŸŕå■ŮèŁĆ0-255åĹřU+0000èĞşU+00FFèŇČåŻť'åĘĚUnicodeå■ŮçňęçŽĎçŻť'æŐěæŸäåřĎãĂĆ å¡Şèŕżå ŔŰäÿĂäÿłæIJłç§ěçijŰçăĄçŽĎæŰĞæIJňæŮűä¡£çŤĺlatin-1çijŰçăĄæřÿè£IJäÿ■äijŽäžğçŤ§èğčçăĄéŤŹèŕŕãĂĆ ä¡£çŤĺlatin-1çijŰçăĄèŕżåŔŰäÿĂäÿłæŰĞäżűçŽĎæŮűåĂŹäź§èőÿäÿ■èČ¡äžğçŤ§åőŇåĚĺæ■ççąőçŽĎæŰĞæIJňèğčçăĄæŢřä¡ĘæŸŕåőČäź§èČ¡äżŐäÿ■æŔŘåŔŰåĞžèűşåđ'§åđ'ŽçŽĎæIJL'çŤĺæŢřæ■őãĂĆåŘŇæŮűïijŇåęĆæđIJä¡ääźŃåŘ

èőĺèőž

èŕżåĘŹæŰĞæIJňæŰĞäżűäÿĂèĹňæİěèőšæŸŕæŕŤè¿ČçőĂå■ŢçŽĎãĂĆä¡ĘæŸŕäź§åĞäçĆźæŸŕéIJĂèęĄæşéęŰåĚĹïijŇåIJĺä¿Ńå■ŘçÍŇåžŔäÿ■çŽĎwithèŕ■åŘěçżŹèćńä¡£çŤĺåĹřçŽĎæŰĞäżűåĹŻåžžäžĘäÿĂäÿłäÿŁäÿŃæä¡Ę with æŐğåĹűåİŮçżŞæİ§æŮűïijŇæŰĞäżűäijŽèĞłåŁĺåĚşéŮ■ãĂĆä¡ääź§åŔŕäžěäÿ■ä¡£çŤĺ with èŕ■åŘěïijŇä¡ĘæŸŕè£ŹæŮűåĂŹä¡ääŕšå£Ěéążèőŕå¿ŮæĽŃåŁĺåĚşéŮ■æŰĞäżűïijŽ

```
f = open('somefile.txt', 'rt')
data = f.read()
f.close()
```

åŘęåđ'ŰäÿĂäÿłéŮőécŸæŸŕåĚşäžŐæ■ćèąŇçňęçŽĎèŕĘåĹńéŮőécŸïijŇåIJĺUnixåŠŇWindowsäÿ■æŸŕäÿ■éżŸèőđ'æČĚåĘţäÿŃïijŇPythonäijŽäžěçż§äÿĂæĺąåijŔåđ'ĎçŘĘæ■ćèąŇçňęãĂĆ

è£ŹçğæÍąåijŔäÿŃïijŇåIJÍèŕžåŔŰæŰĞæIJňçŽĎæŮűåĂŹïijŇPythonåŔŕäžěèŕĘåĹńæĽĂæIJL'çŽĎæŹőéĂŽæ
\n å■ŮçňęãĂĆ çśžäijijçŽĎïijŇåIJÍè¿ŞåĞžæŮűäijŽářĘæ■ćèąŇçňę \n
è¡ňæ■ćäÿžçşżçż§éżŸèőď'çŽĎæ■ćèąŇçňęãĂĆ åęĆæđIJä¡ăäÿ■åÿŇæIJŻè£Źçğ■éżŸèőď'çŽĎåď'ĎçŘĘæŰźåijŔi
`open()` åĞ¡æŢřäijăåĚěåŔĆæŢř `newline=''` ïijŇářšåČŔäÿŃéÍćè£ŹæăůïijŽ

```
# Read with disabled newline translation
with open('somefile.txt', 'rt', newline='') as f:
    ...
```

äÿžäžĘèŕť'æŸŐäÿď'èĂĚäźŃéŮť'çŽĎåůőåijĆïijŇäÿŃéÍćæĹŠåIJÍUnixæIJžåŹÍäÿŁéÍćèŕžåŔŰäÿĂäÿłWindo
`hello world!\r\n` ïijŽ

```
>>> # Newline translation enabled (the default)
>>> f = open('hello.txt', 'rt')
>>> f.read()
'hello world!\n'

>>> # Newline translation disabled
>>> g = open('hello.txt', 'rt', newline='')
>>> g.read()
'hello world!\r\n'
>>>
```

æIJĂåŘŐäÿĂäÿłéŮőéćŸåřśæŸŕæŰĞæIJňæŰĞäžűäÿ■åŔŕèČ¡åĞžçŐřçŽĎçijŰçăĄéŤŹèŕŕãĂĆ
ä¡Ęä¡ăèŕžåŔŰæĹŰèĂĚåĘŹåĚěäÿĂäÿłæŰĞæIJňæŰĞäžűæŮűïijŇä¡ăåŔŕèČ¡äijŽéĄĞåĹřäÿĂäÿłçijŰçăĄæĹŰè

```
>>> f = open('sample.txt', 'rt', encoding='ascii')
>>> f.read()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "/usr/local/lib/python3.3/encodings/ascii.py", line 26, in␣
↪decode
        return codecs.ascii_decode(input, self.errors)[0]
UnicodeDecodeError: 'ascii' codec can't decode byte 0xc3 in position
12: ordinal not in range(128)
>>>
```

åęĆæđIJåĞžçŐřè£ŹäÿłéŤŹèŕŕïijŇéĂŽåÿÿèąÍçď'žä¡ăèŕžåŔŰæŰĞæIJňæŮűæŇĞåőŽçŽĎçijŰçăĄäÿ■æ■čç
ä¡ăæIJĂåě¡äžŤçžĘéŸĚèŕžèŕť'æŸŐåźűçąőèőď'ä¡ăçŽĎæŰĞäžűçijŰçăĄæŸŕæ■čçąőçŽĎ(æŕŤåęĆä¡£çŤÍUTF-
8èĂŇäÿ■æŸŕLatin-1çijŰçăĄæĹŰåĚűäžŰ)ãĂĆ åęĆæđIJçijŰçăĄéŤŹèŕŕè£ŸæŸŕå■ŸåIJÍçŽĎèŕÍïijŇä¡ăåŔŕäžě
`open()` åĞ¡æŢřäijăéĂŠäÿĂäÿłåŔŕéĂL'çŽĎ `errors` åŔĆæŢřæÍěåď'ĎçŘĘè£ŹäžŻéŤŹèŕŕãĂĆ
äÿŃéÍćæŸŕäÿĂäžŻåď'ĎçŘĘåÿÿèğĄéŤŹèŕŕçŽĎæŰźæşŢïijŽ

```
>>> # Replace bad chars with Unicode U+fffd replacement char
>>> f = open('sample.txt', 'rt', encoding='ascii', errors='replace')
>>> f.read()
'Spicy Jalape?o!'
>>> # Ignore bad chars entirely
>>> g = open('sample.txt', 'rt', encoding='ascii', errors='ignore')
>>> g.read()
'Spicy Jalapeo!'
```

```
>>>
```

åęĆæđIJä¡ăçżŕåÿÿä¡£çŤĺ errors åŕĆæŢřæİěåď ĎçŘEçijŰçăĄéŤŹèŕřïijŇåŕřèČ¡äijŽèőľä¡ăçŽĎçŤ§æt
åŕźäžŐæŰĞæIJňåď ĎçŘEçŽĎéęŰèęĄåŐ§åĹŹæŸŕçąőä£İä¡ăæĂżæŸŕä¡£çŤĺçŽĎæŸŕæ■čçąőçijŰçăĄãĂĆå¡Ş
8)ãĂĆ

5.2 æĽŞå■řè¿ŞåĞžèĞşæŰĞäžűäÿ■

éŮőécŸ

ä¡ăæČşåřĘ print() åĞ¡æŢřçŽĎè¿ŞåĞžéĞ■åőŽåŘŚåĹřäÿĂäÿłæŰĞäžűäÿ■åŐżãĂĆ

èğčåĘşæŰźæąĹ

åIJĺ print() åĞ¡æŢřäÿ■æŇĞåőŽ file åĚşéŤőå■ŮåŔĆæŢřijŇåČŔäÿŃéİćè£ŹæăůïijŽ

```
with open('d:/work/test.txt', 'wt') as f:
    print('Hello World!', file=f)
```

èőĺèőž

åĚşäžŐè¿ŞåĞžéĞ■åőŽåŘŚåĹřæŰĞäžűäÿ■åřśè£ŹäžŻäžĘãĂĆä¡ĘæŸŕæIJĽäÿĂçĆźèęĄæşĺæĎŔçŽĎåřśæ
åęĆæđIJæŰĞäžűæŸŕäžŇè£ŻåĹűæĺąåijŔçŽĎèŕİïijŇæĽŞå■řåřśäijŽåĞžéŤŹãĂĆ

5.3 ä¡£çŤĺåĚűäžŰåĹĘéŽŤçňęæĹŰèąŇçżĹæ■ćçňęæĽŞå■ř

éŮőécŸ

ä¡ăæČşä¡£çŤĺ print() åĞ¡æŢřè¿ŞåĞžæŢřæ■őïijŇä¡ĘæŸŕæČşæŤźåŘŸéżŸèőď çŽĎåĹĘéŽŤçňęæĹŰè

èğčåĘşæŰźæąĹ

åŔŕäžěä¡£çŤĺåIJĺ print() åĞ¡æŢřäÿ■ä¡£çŤĺ sep åŠŇ end
åĚşéŤőå■ŮåŔĆæŢřijŇäžěä¡ăæČşèęĄçŽĎæŰźåijŔè¿ŞåĞžãĂĆærŤåęĆïijŽ

```
>>> print('ACME', 50, 91.5)
ACME 50 91.5
>>> print('ACME', 50, 91.5, sep=',')
ACME,50,91.5
>>> print('ACME', 50, 91.5, sep=',', end='!!\n')
```

```
ACME,50,91.5!!
>>>
```

ä¡£çŤĺ end åŕĆæŢřäź§åŕŕäžěåIJĺè¿ŞåĞžäÿ■çęĄæ■ćæ■ćèąŇãĂĆærŤåęĆïijŽ

```
>>> for i in range(5):
...     print(i)
...
0
1
2
3
4
>>> for i in range(5):
...     print(i, end=' ')
...
0 1 2 3 4 >>>
```

èőĺèőž

å¡Şä¡ăæČşä¡£çŤĺéİđçľžæăijåĹĘéŽŤçňęæİèè¿ŞåĞžæŢřæ■őçŽĎæŮűåĂŹïijŇçżŹ print() åĞ¡æŢřäijăéĂŠäÿĂäÿł sep åŕĆæŢřæŸŕæIJĂçőÅå■ŢçŽĎæŰźæąĹãĂĆ æIJľæŮűåĂŹä¡ääijŽçIJŃåĹřäÿĂäžŽçĺŃåžŕåŚŸäijŽä¡£çŤĺ str.join() æİěåőŇæĹŘåŘŇæăůçŽĎäžŃæČĚãĂĆærŤåęĆïijŽ

```
>>> print(','.join(('ACME','50','91.5')))
ACME,50,91.5
>>>
```

str.join() çŽĎéŮőécŸåIJĺäžŐåőČäżĚäżĚéĂĆçŤĺäžŐå■Ůçňęäÿšã ĂĆè£ŹæĎŕåŚşçİĂä¡ăéĂŽåÿÿéIJ

```
>>> row = ('ACME', 50, 91.5)
>>> print(','.join(row))
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: sequence item 1: expected str instance, int found
>>> print(','.join(str(x) for x in row))
ACME,50,91.5
>>>
```

ä¡ăå¡ŞçĎűåŕŕäžěäÿ■çŤĺéĆčäźĹéžżçČęïijŇäżĚäżĚåŕłéIJĂèęĄåČŕäÿŃéİćè£ŹæăůåĘŹïijŽ

```
>>> print(*row, sep=',')
ACME,50,91.5
>>>
```

5.4 èŕžåĘŹå■ŮèŁĆæŢřæ■ő

éŮőécŸ

ä¡ăæČşèŕžåĘŹäžŇè£ŻåĹűæŰĞäżűïijŇæŕŤåęĆåŻ¿çĽĞïijŇåčŕé§şæŰĞäżűç■Ľç■Ľã ĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤĺæĺąåijŔäÿž `rb` æĹŰ `wb` çŽĎ `open()` åĞ¡æŢřæİěèŕžåŔŰæĹŰåĘŹåĚěäžŇè£ŻåĹűæŢřæ■őãĆćæŕ

```
# Read the entire file as a single byte string
with open('somefile.bin', 'rb') as f:
    data = f.read()

# Write binary data to a file
with open('somefile.bin', 'wb') as f:
    f.write(b'Hello World')
```

åIJĺèŕžåŔŰäžŇè£ŻåĹűæŢřæ■őæŮűïijŇéIJĂèęĄæŇĞæŸŐçŽĎæŸŕæĽĂæIJĽè£ŤåŻđçŽĎæŢřæ■őéČ¡æ
çśżäijijçŽĎïijŇåIJĺåĘŹåĚěçŽĎæŮűåĂŹïijŇå£Ěéążä£ÌèŕĄåŔĆæŢřæŸŕäżěå■ŮèŁĆå¡ćåijŔåŕźåď ŰæŽťéIJšæŢ

èőĺèőž

åIJĺèŕžåŔŰäžŇè£ŻåĹűæŢřæ■őçŽĎæŮűåĂŹïijŇå■ŮèŁĆå■Ůçňęäÿšå ŠŇæŰĞæIJňå■Ůçňęäÿšç ŽĎèŕ■äźĽ
çĽźåĹńéIJĂèęĄæşĺæĐŔçŽĎæŸŕïijŇçť ćåijŢåŠŇè£■äžčåŁĺä¡IJè£ŤåŻđçŽĎæŸŕå■ŮèŁĆçŽĎåĂijèĂŇäÿ■æŸ

```
>>> # Text string
>>> t = 'Hello World'
>>> t[0]
'H'
>>> for c in t:
...     print(c)
...
H
e
l
l
o
...
>>> # Byte string
>>> b = b'Hello World'
>>> b[0]
72
>>> for c in b:
...     print(c)
...
```

```
72
101
108
108
111
...
>>>
```

åęĆæđIJä¡ăæČşäżŐäžŇè£ŻåĹűæĺąåijŔçŽĎæŰĞäżűäÿ■èŕżåŔŰæĹŰåĘŹåĚěæŰĞæIJňæŢřæ■őïijŇå£Ěéą

```
with open('somefile.bin', 'rb') as f:
    data = f.read(16)
    text = data.decode('utf-8')

with open('somefile.bin', 'wb') as f:
    text = 'Hello World'
    f.write(text.encode('utf-8'))
```

äžŇè£ŻåĹűI/Oè£ŸæIJL'äÿĂäÿłéšIJäÿžäžžç§ěçŽĎçL'źæĂgåřśæŸŕæŢřçżĎåŠŇCçżŞæđĎä¡Şçśżåđ́NèČ¡çŹ

```
import array
nums = array.array('i', [1, 2, 3, 4])
with open('data.bin','wb') as f:
    f.write(nums)
```

è£ŹäÿłéĂĆçŤĺäžŐäżżä¡Ţåőđç̃ŐřäžĘèćńçğřäźŃäÿž"çijŞåĘšæŐěåŔč"çŽĎåŕźèśąïijŇè£Źçğ■åŕźèśąäijŽçŻt
äžŇè£ŻåĹűæŢřæ■őçŽĎåĘŹåĚěåřśæŸŕè£ŹçśżæŞ■ä¡IJäźŃäÿĂãĂĆ

å¿ĹåđŽåŕźèśąè£ŸåĚĄèőÿéĂŽè£Ğä¡£çŤĺæŰĞäżűåŕźèśąçŽĎ `readinto()`
æŰźæşŢçŻť'æŐěèŕżåŔŰäžŇè£ŻåĹűæŢřæ■őåĹřåĚűåžŢåśĆçŽĎåĘĚå■Ÿäÿ■åŐżãĂĆærŤåęĆïijŽ

```
>>> import array
>>> a = array.array('i', [0, 0, 0, 0, 0, 0, 0, 0])
>>> with open('data.bin', 'rb') as f:
...     f.readinto(a)
...
16
>>> a
array('i', [1, 2, 3, 4, 0, 0, 0, 0])
>>>
```

ä¡ĘæŸŕä¡£çŤĺè£Źçğ■æŁĂæIJŕçŽĎæŮűåĂŹéIJĂèęĄæăijåđ'ŰåřŔå£ČïijŇåŽăäÿžåőČéĂŽåÿÿåĚůæIJL'åź
åŔŕäżěæ§ěçIJŃ5.9åřŔèŁĆäÿ■åŔęåđ'ŰäÿĂäÿłèŕżåŔŰäžŇè£ŻåĹűæŢřæ■őåĹřåŔŕä£őæŤźçijŞåĘšåŇžçŽĎä¿Ńå

5.5 æŰĞäžűäÿ■å■ŸåIJÍæĽ■èČ¡åĘŹåĚě

éŮőécŸ

ä¡ăæČşåČŔäÿĂäÿłæŰĞäžűäÿ■åĘŹåĚěæŢřæ■őïijŇä¡ĘæŸŕåĽ■æŔŘå£ĚéążæŸŕè£ŹäÿłæŰĞäžűåIJÍæŰĞäźş§åřśæŸŕäÿ■åĚĄèőÿèęEçŻŰåůšå■ŸåIJÍçŽĎæŰĞäžűåĘĚåőźãĂĆ

èğčåĘşæŰźæąĹ

åŔŕäžěåIJÍ `open()` åĞ¡æŢřäÿ■ä¡£çŤÍ `x` æÍąåijŔæİěäžčæŻ£ `w` æÍąåijŔçŽĎæŰźæşŢæİěèğčåĘşè£ŹäÿłéŮőécŸãĂĆæŕŤåęĆïijŽ

```
>>> with open('somefile', 'wt') as f:
...     f.write('Hello\n')
...
>>> with open('somefile', 'xt') as f:
...     f.write('Hello\n')
...
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
FileExistsError: [Errno 17] File exists: 'somefile'
>>>
```

åęĆæđIJæŰĞäžűæŸŕäžŇè£ŻåĹűçŽĎïijŇä¡£çŤÍ `xb` æİěäžčæŻ£ `xt`

èőÍèőž

è£ŹäÿĂåřŔèŁĆæijŢçd'žäžĘåIJÍåĘŹæŰĞäžűæŮűéĂŽåÿÿäijŽéĄĞåĹřçŽĎäÿĂäÿłéŮőécŸçŽĎåőŇç¿Őèğ
äÿĂäÿłæŻ£äžčæŰźæąĹæŸŕåĚĹæţŃèŕŢè£ŹäÿłæŰĞäžűæŸŕåŘęå■ŸåIJÍïijŇåČŔäÿŃéÍćè£ŹæăůïijŽ

```
>>> import os
>>> if not os.path.exists('somefile'):
...     with open('somefile', 'wt') as f:
...         f.write('Hello\n')
... else:
...     print('File already exists!')
...
File already exists!
>>>
```

æŸ¿èĂŇæŸŞèğĄïijŇä¡£çŤÍxæŰĞäžűæÍąåijŔæŻt'åŁăçőÅå■ŢãĂĆèęĄæşÍæĎŔçŽĎæŸŕxæÍąåijŔæŸŕäÿ
`open()` åĞ¡æŢřçĽźæIJĽçŽĎæĽl'åśŢãĂĆ åIJÍPythonçŽĎæŮğçĽĹæIJňæĹŰèĂĚæŸýPythonåőđçŐřçŽĎåžŢ

5.6 å■Ůçňęäÿšç ŽĎl/OæŞ■ä¡IJ

éŮőécŸ

ä¡äæČşä¡£çŤĺæŞ■ä¡IJçśżæŰĞäžűåŕźèśąçŽĎçĺŃåžŔæİěæŞ■ä¡IJæŰĞæIJňæĹŰäžŇè£ŻåĹűå■Ůçňęäÿšã

èğčåĘşæŰźæąĹ

ä¡£çŤĺ `io.StringIO()` åŠŇ `io.BytesIO()` çśżæİěåĹŻåžžçśżæŰĞäžűåŕźèśąæŞ■ä¡IJå■Ůçňęäÿšæ

```
>>> s = io.StringIO()
>>> s.write('Hello World\n')
12
>>> print('This is a test', file=s)
15
>>> # Get all of the data written so far
>>> s.getvalue()
'Hello World\nThis is a test\n'
>>>

>>> # Wrap a file interface around an existing string
>>> s = io.StringIO('Hello\nWorld\n')
>>> s.read(4)
'Hell'
>>> s.read()
'o\nWorld\n'
>>>
```

`io.StringIO` åŔłèČ¡çŤĺäžŐæŰĞæIJňãĂĆåęĆæđIJä¡ăèęĄæŞ■ä¡IJäžŇè£ŻåĹűæŢřæ■őïijŇèęĄä¡£çŤĺ `io.BytesIO` çśżæİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
>>> s = io.BytesIO()
>>> s.write(b'binary data')
>>> s.getvalue()
b'binary data'
>>>
```

èőĺèőž

å¡Şä¡äæČşæĺąæŃ§äÿĂäÿłæŹőéĂŽçŽĎæŰĞäžűçŽĎæŮűåĂŹ `StringIO` åŠŇ `BytesIO` çśżæŸŕå¿ĹæIJL'çŤĺçŽĎãĂĆ æŕŤåęĆïijŇåIJĺå■ŢåĚČæţŃèŕŢäÿ■ïijŇä¡ăåŔŕäžěä¡£çŤĺ `StringIO` æİěåĹŻåžžäÿĂäÿłåŇĚåŘńæţŃèŕŢæŢřæ■őçŽĎçśżæŰĞäžűåŕźèśąïijŇè£ŹäÿłåŕźèśąåŔŕäžěèćńäijăçżŹæ§ŘäÿłåŔĆæŢřäÿžæŹőéĂŽæŰĞäžűåŕźèśąçŽĎåĞ¡æŢřãĂĆ

éIJĂèęĄæşĺæĎŘçŽĎæŸŕïijŇ `StringIO` åŠŇ `BytesIO` åőđä¿ŃåźűæšąæIJL'æ■čçąőçŽĎæŢť'æŢřçśżåđŃçŽĎæŰĞäžűæŔŔè£řçňęãĂĆ

åŻăæ■ď˙ïijŇåőČäžňäÿ■èČ¡åIJĺéĆčäžŻéIJĂèęĄä¡£çŤĺçIJ§åőđçŽĎçşżçż§çžğæŰĞäžűåęĆæŰĞäžűïijŇçőąéĄŞ

5.7 èŕżåĘŹåŐŃçijľæŰĞäžű

éŮőécŸ

ä¡ăæČşèŕżåĘŹäÿĂäÿłgzipæĹŰbz2æäijåijŔçŽĎåŐŃçijľæŰĞäžűãĂĆ

èğčåĘşæŰźæąĹ

gzip åŠŇ bz2 æĺąåİŮåŔŕäžěå¿Ĺåőźæ˘ŞçŽĎåď˙ĎçŘĘè£ŹäžŻæŰĞäžűãĂĆ äÿď˙äÿłæĺąåİŮéČ¡äÿž open() åĞ¡æŢřæŔŘä¿ŻäžĘåŘęåď˙ŰçŽĎåőđçŐřæİèèğčåEşè£ŹäÿłéŮőécŸãĂĆ ærŤåęĆïijŇäÿžäžĘäžěæŰĞæIJňå¡ćåijŔèŕżåŔŰåŐŃçijľæŰĞäžűïijŇåŔŕäžěè£ŹæăůåĄŽïijŽ

```
# gzip compression
import gzip
with gzip.open('somefile.gz', 'rt') as f:
    text = f.read()

# bz2 compression
import bz2
with bz2.open('somefile.bz2', 'rt') as f:
    text = f.read()
```

çśżäijijçŽĎïijŇäÿžäžĘåĘŹåĚěåŐŃçijľæŢřæ■őïijŇåŔŕäžěè£ŹæăůåĄŽïijŽ

```
# gzip compression
import gzip
with gzip.open('somefile.gz', 'wt') as f:
    f.write(text)

# bz2 compression
import bz2
with bz2.open('somefile.bz2', 'wt') as f:
    f.write(text)
```

åęĆäÿŁïijŇæL˘ĂæIJL˘çŽĎI/OæŞ■ä¡IJéČ¡ä¡£çŤĺæŰĞæIJňæĺąåijŔåźűæL˘ğèąŇUnicodeçŽĎçijŰçăA/èğčç
çśżäijijçŽĎïijŇåęĆædIJä¡ăæČşæŞ■ä¡IJäžŇè£ŻåĹűæŢřæ■őïijŇä¡£çŤĺ rb æĹŰèĂĚ wb
æŰĞäžűæĺąåijŔå■şåŔŕãĂĆ

èőĺèőž

åď˙ğéČĺåĹĘæČĚåĘţäÿŃèŕżåĘŹåŐŃçijľæŢřæ■őéČ¡æŸŕå¿ĹçőĂå■ŢçŽĎãĂĆä¡ĘæŸŕèęĄæşĺæĐŔçŽĎæŸ
åęĆædIJä¡ăäÿ■æŇĞåőŽæĺąåijŔïijŇéĆčäźĹézŸèőď˙çŽĎåřśæŸŕäžŇè£ŻåĹűæĺąåijŔïijŇåęĆædIJè£ŹæŮůåĂŹç
gzip.open() åŠŇ bz2.open() æŐěåŔŮèů§åĘĚç¡őçŽĎ open()

åĞ¡æŢřäÿĂæăůçŽĎåŔĆæŢřïijŇ åŇĚæŃň `encoding`ïijŇ`errors`ïijŇ`newline` ç■Ľç■Ľ˜ãĂĆ

å¡ŞåĘŹåĚěåŐŃçijľæŢřæ■őæŮűïijŇåŕräžěä¡£çŤÍ `compresslevel` è£ŹäÿłåŔréĂĽçŽĎåĚşéŤőå■ŮåŔĆæŢřæİěæŇĞåőŽäÿĂäÿłåŐŃçijľçžğåĹńãĂĆærŤåęĆïijŽ

```
with gzip.open('somefile.gz', 'wt', compresslevel=5) as f:
    f.write(text)
```

éżŸèőd'çŽĎç■Ľ˜çžğæŸŕ9ïijŇäź§æŸŕæIJĂénŸçŽĎåŐŃçijľç■Ľ˜çžğãĂĆç■Ľ˜çžğèűŁä¡ŐæĂğèČ¡èűŁåě¡ïij

æIJĂåŘŐäÿĂçĆźïijŇ `gzip.open()` åŠŇ `bz2.open()` è£ŸæIJĽäÿĂäÿłå¿ĹářŠèćńç§ěéĄŞçŽĎçĽźæĂğïijŇ åőČäžňåŕräžěä¡IJçŤĺåIJĺäÿĂäÿłåůšå■ŸåIJĺäźűäžěäžŇè£

```
import gzip
f = open('somefile.gz', 'rb')
with gzip.open(f, 'rt') as g:
    text = g.read()
```

è£ŹæăůářšåĚĄèőÿ `gzip` åŠŇ `bz2` æĺąåİŮåŕräžěåůěä¡IJåIJĺèőÿåd'ŽçśżæŰĞäžűářźèśąäÿŁïijŇæŕŤåęĆåě

5.8 åŻžáőŽåd'ğářŕèőřå¡ŢçŽĎæŰĞäžűè£■äžč

éŮőécŸ

ä¡ăæČşåIJĺäÿĂäÿłåŻžåőŽéŢ£åžęèőřå¡ŢæĹŰèĂĚæŢřæ■őåİŮçŽĎéŻĘåŘĹäÿŁè£■äžčïijŇèĂŇäÿ■æŸŕåIJ

èğčåĘşæŰźæąĹ

éĂŽè£ĞäÿŃéİćè£ŹäÿłářŕæŁĂåůğä¡£çŤÍ `iter` åŠŇ `functools.partial()` åĞ¡æŢřijŽ

```
from functools import partial

RECORD_SIZE = 32

with open('somefile.data', 'rb') as f:
    records = iter(partial(f.read, RECORD_SIZE), b'')
    for r in records:
        ...
```

è£Źäÿłä¿Ňå■Řäÿ■çŽĎ `records` årźèśąæŸŕäÿĂäÿłåŕrè£■äžčårźèśąïijŇåőČäijŽäÿ■æŰ■çŽĎäžğçŤ§åŻž èęĄæşĺæĎŔçŽĎæŸŕåęĆædIJæĂżèőřå¡Ţåd'ğářŘäÿ■æŸŕåİŮåd'ğářŔçŽĎæŢt'æŢřåĂ■çŽĎèŕĪïijŇæIJĂåŘŐäÿĂ

èőĺèőž

iter() åĞ¡æŢřæIJL'äÿĂäÿłéšIJäÿžäžžç§ěçŽĎçL'źæĂğåřśæŸŕïijŇåęĆæđIJä¡äçżŹåőČäijăéĂŠäÿĂäÿłåI
è£Źäÿłè£■äžčåŹĺäijŽäÿĂçŻť'èřČçŤĺäijăåĚěçŽĎåŕèřČçŤĺåŕźèśąçŻť'åĽřåőČè£ŤåŻđæăĞèőřåĂijäÿžæ■ćïijŇè£

åIJĺä¿Ńå■Řäÿ■ïijŇ functools.partial çŤĺæİěåĹŻåžžäÿĂäÿłæŕŔæňąèćńèřČçŤĺæŮűäżŐæŰĞäžűä
æăĞèőřåĂij b'' åřśæŸŕå¡ŞåĽřè¿¿æŰĞäžűçżŞåř¿æŮűçŽĎè£ŤåŻđåĂijãĂĆ

æIJĂåŘŐåĘ■æŔŘäÿĂçĆźïijŇäÿŁéIćçŽĎä¿Ńå■Řäÿ■çŽĎæŰĞäžűæŮűäžěäžŇè£ŻåĽűæĺąåijŔæĽŞåijĂç
åęĆæđIJæŸŕèŕžåŔŰåŻžåőŽåđ'ğåřŔçŽĎèőřå¡ŢïijŇè£ŹéĂŽåÿÿæŸŕæIJĂæŹőéĄ■çŽĎæČĚåĘţãĂĆ
èĂŇåŕźäžŐæŰĞæIJňæŰĞäžűïijŇäÿĂèąŇäÿĂèąŇçŽĎèŕžåŔŰ(éżŸèőd'çŽĎè£■äžčèąŇäÿž)æŻť'æŹőéĄ■çĆźã

5.9 èŕžåŔŰäžŇè£ŻåĹűæŢřæ■őåĹřåŔŕåŔŸçijŞåĘšåŇžäÿ■

éŮőécŸ

ä¡ăæČşçŻť'æŐèèŕžåŔŰäžŇè£ŻåĹűæŢřæ■őåĹřäÿĂäÿłåŔŕåŔŸçijŞåĘšåŇžäÿ■ïijŇèĂŇäÿ■éIJĂèęĄåĄŽäż
æĹŰèĂĚä¡ăæČşåŐ§åIJřă£őæŤźæŢřæ■őåźűårĘáőČåĘŹåŻđåĹřäÿĂäÿłæŰĞäžűäÿ■åŐžãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘèŕžåŔŰæŢřæ■őåĹřäÿĂäÿłåŔŕåŔŸæŢřçżĎäÿ■ïijŇä¡£çŤĺæŰĞäžűåŕźèśąçŽĎ
readinto() æŰźæşŢãĂĆæŕŤåęĆïijŽ

```
import os.path

def read_into_buffer(filename):
    buf = bytearray(os.path.getsize(filename))
    with open(filename, 'rb') as f:
        f.readinto(buf)
    return buf
```

äÿŃéIćæŸŕäÿĂäÿłæijŤçd'žè£ŹäÿłåĞ¡æŢřä¡£çŤĺæŰźæşŢçŽĎä¿Ńå■ŘïijŽ

```
>>> # Write a sample file
>>> with open('sample.bin', 'wb') as f:
...     f.write(b'Hello World')
...
>>> buf = read_into_buffer('sample.bin')
>>> buf
bytearray(b'Hello World')
>>> buf[0:5] = b'Hallo'
>>> buf
bytearray(b'Hallo World')
>>> with open('newsample.bin', 'wb') as f:
...     f.write(buf)
...
```

```
11
>>>
```

èőĺèőž

æŰĞäžűåŕźèśąçŽĎ `readinto()` æŰźæşŢèČ¡èćńçŤĺæİěäÿžécĎåĚĹåĹĘéĚ■åĘĚå■ŸçŽĎæŢřçżĎåąńåĚ `array` æĺąåİŮæĹŰ `numpy` åžŞåĹŻåžžçŽĎæŢřçżĎãĂĆ åŠŇæŹőéĂŽ `read()` æŰźæşŢäÿ■åŘŇçŽĎæŸŕïijŇ `readinto()` åąńåĚĚåůšå■ŸåIJĺçŽĎçijŞåĘšåŇžèĂŇäÿ■æŸŕäÿžæŰřåŕźèśąéĞ åŽăæ■ď'ïijŇä¡ăåŔŕäžěä¡£çŤĺåőČæİěéĄ£åĚ■åď'ğéĞŔçŽĎåĘĚå■ŸåĹĘéĚ■æŞ■ä¡IJãĂĆ æŕŤåęĆïijŇåęĆæđIJä¡ăèŕžåŔŰäÿĂäÿłçŤśçŻÿåŘŇåď'ğåŕŔçŽĎèőřå¡ŢçżĎæĹŘçŽĎäžŇè£ŽåĹűæŰĞäžűæŮűïij

```
record_size = 32 # Size of each record (adjust value)

buf = bytearray(record_size)
with open('somefile', 'rb') as f:
    while True:
        n = f.readinto(buf)
        if n < record_size:
            break
        # Use the contents of buf
        ...
```

åŔęåď'ŰæIJL'äÿĂäÿłæIJL'èűčçL'źæĂğåřśæŸŕ `memoryview` ïijŇ åőČåŔŕäžěéĂŽè£ĞéŽűåď'■åĹűçŽĎæŰźåijŔåŕźåůšå■ŸåIJĺçŽĎçijŞåĘšåŇžæL'ğèąŇåĹĞçL'ĞæŞ■ä¡IJïijŇçŤŽè

```
>>> buf
bytearray(b'Hello World')
>>> m1 = memoryview(buf)
>>> m2 = m1[-5:]
>>> m2
<memory at 0x100681390>
>>> m2[:] = b'WORLD'
>>> buf
bytearray(b'Hello WORLD')
>>>
```

ä¡£çŤĺ `f.readinto()` æŮűéIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇä¡ăå£ĚéążæčĂæ§ěåőČçŽĎè£ŤåŽđåĂijïijŇä

åęĆæđIJå■ŮèŁĆæŢřåŕŔäžŐçijŞåĘšåŇžåď'ğåŕŔïijŇèąĺæŸŐæŢřæ■ôèćńæĹłæŰ■æĹŰèĂĚèćńçăť'åİŔäžĘ

æIJĂåŘŐïijŇçŢŹå£ČègĆår§åĚűäžŰåĞ¡æŢřåžŞåŠŇæĺąåİŮäÿ■åŠŇ `into` çŽÿåĚşçŽĎåĞ¡æŢř(æŕŤåęĆ `recv_into()` ïijŇ `pack_into()` ç■Ľ)ãĂĆ PythonçŽĎå¿Ĺåď'ŽåĚűäžŰéČĺåĹĘåůšçżŔèČ¡æŤŕæŇĄçŽť'æŐěçŽĎI/OæĹŰæŢřæ■ôèő£éŮőæŞ■ä¡IJïijŇè£Źä

åĚşäžŐèğčæđŘäžŇè£ŽåĹűçżŞæđĎåŠŇ `memoryviews` ä¡£çŤĺæŰźæşŢçŽĎæŽť'énŸçžğä¿Ňå■ŘïijŇèŕůåŔĆèĂČ6.12åŕŔèŁĆãĂĆ

5.10 åĘĚå■ŸæŸăåřĎçŽĎäžŇè£ŻåĹűæŰĞäžű

éŮőécŸ

ä¡äæČşåĘĚå■ŸæŸăåřĎäÿĂäÿłäžŇè£ŻåĹűæŰĞäžűåĹřäÿĂäÿłåŔŕåŔŸå■ŮèŁĆæŢřçżĎäÿ■ïijŇçŻőçŽĎåĿ

èğčåĘşæŰźæąĹ

ä¡£çŤÍ mmap æĺąåİŮæİěåĘĚå■ŸæŸăåřĎæŰĞäžűãĂĆ
äÿŃéİćæŸŕäÿĂäÿłåůěåĚůåĞ¡æŢřïijŇåŘŚä¡ăæijŤçd'žäžĘåęĆä¡ŢæĽ'ŞåijĂäÿĂäÿłæŰĞäžűåźűäžëäÿĂçğ■ä¿£æ

```
import os
import mmap

def memory_map(filename, access=mmap.ACCESS_WRITE):
    size = os.path.getsize(filename)
    fd = os.open(filename, os.O_RDWR)
    return mmap.mmap(fd, size, access=access)
```

äÿžäžĘä¡£çŤÍè£ŹäÿłåĞ¡æŢřïijŇä¡ăéIJĂèęĄæIJL'äÿĂäÿłåůšåĹŻåžžåźűäÿŤåĘĚåőźäÿ■äÿžçl'žçŽĎæŰĞäž
äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijŇæŢŹä¡ăæĂŐæăůåĹİåğŃåĹŻåžžäÿĂäÿłæŰĞäžűåźűåřĘåĚűåĘĚåőźæL'l'åĚĚåĹř

```
>>> size = 1000000
>>> with open('data', 'wb') as f:
...     f.seek(size-1)
...     f.write(b'\x00')
...
>>>
```

äÿŃéİćæŸŕäÿĂäÿłåĹl'çŤÍ memory_map() åĞ¡æŢřçśżåĘĚå■ŸæŸăåřĎæŰĞäžűåĘĚåőźçŽĎä¿Ńå■ŘïijŽ

```
>>> m = memory_map('data')
>>> len(m)
1000000
>>> m[0:10]
b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
>>> m[0]
0
>>> # Reassign a slice
>>> m[0:11] = b'Hello World'
>>> m.close()

>>> # Verify that changes were made
>>> with open('data', 'rb') as f:
... print(f.read(11))
...
b'Hello World'
>>>
```

mmap() è£ŤåŻđçŽĎ mmap åŕźèśąåŘŇæăůäźŠåŔŕäžěä¡IJäÿžäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺæİěä¡£çŤĺïij
è£ŹæŮűåĂŹåžŢåśĆçŽĎæŰĞäžűäijŽèćńèĞłåŁĺåĚşéŮ■ãĂĆæŕŤåęĆïijŽ

```
>>> with memory_map('data') as m:
...     print(len(m))
...     print(m[0:10])
...
1000000
b'Hello World'
>>> m.closed
True
>>>
```

éżŸèőd'æČĚåĘţäÿŃïijŇ memeory_map() åĞ¡æŢřæL'Şåij ĂçŽĎæŰĞäžűåŘŇæŮűæŤŕæŇĄèŕźåŠŇåĘŹ
äžżä¡ŢçŽĎä£őæŤźåĘĚåőźéČ¡äijŽåd'■åĹűåŻđåŐ§æİěçŽĎæŰĞäžűäÿ■ãĂĆ
åęĆæđIJéIJĂèęĄåŔłèŕźçŽĎè£éŮőæĺąåjŔïijŇåŔŕäžěçżŹåŔĆæŢř access èţŃåĂijäÿž
mmap.ACCESS_READ ãĂĆæŕŤåęĆïijŽ

```
m = memory_map(filename, mmap.ACCESS_READ)
```

åęĆæđIJä¡ăæČşåIJĺæIJňåIJřä£őæŤźæŢřæ■őïijŇä¡ĘæŸŕåŔĹäÿ■æČşåřĘä£őæŤźåĘŹåŻđåĹřåŐ§ågŃæŰĞ
mmap.ACCESS_COPY ïijŽ

```
m = memory_map(filename, mmap.ACCESS_COPY)
```

èőĺèőž

äÿžäžĘéŽŔæIJžè£éŮőæŰĞäžűçŽĎåĘĚåőźïijŇä¡£çŤĺ mmap
åřĘæŰĞäžűæŸăåřĎåĹřåĘĚå■ŸäÿæŸŕäÿĂäÿłéńŸæŢĹåŠŇäijŸéŻĘçŽĎæŰźæşŢãĂĆ
ä¿ŃåęĆïijŇä¡ăæŮăéIJĂæL'ŞåijĂäÿĂäÿłæŰĞäžűåźűæL'ğèąŇåď'ğéĞŔçŽĎ seek() ïijŇ
read() ïijŇ write() èřČçŤĺïijŇ åŔłéIJĂèęĄçőĂå■ŢçŽĎæŸăåřĎæŰĞäžűåźűä¡£çŤĺåĹĞçL'ĞæŞ■ä¡IJè£é

äÿĂèĹňæİěèőšïijŇ mmap() æL'ĂæŽť'éIJšçŽĎåĘĚå■ŸçIJŃäÿŁåŐżåŕśæŸŕäÿĂäÿłäžŇè£ŻåĹűæŢřçżĎåŕź
ä¡ĘæŸŕïijŇä¡ăåŔŕäžěä¡£çŤĺäÿĂäÿłåĘĚå■ŸèğĘåŻ¿æİěèğčæđŘåĚűäÿ■çŽĎæŢřæ■őãĂĆæŕŤåęĆïijŽ

```
>>> m = memory_map('data')
>>> # Memoryview of unsigned integers
>>> v = memoryview(m).cast('I')
>>> v[0] = 7
>>> m[0:4]
b'\x07\x00\x00\x00'
>>> m[0:4] = b'\x07\x01\x00\x00'
>>> v[0]
263
>>>
```

éIJĂèęĄåijžèřČçŽĎäÿĂçĆźæŸŕïijŇåĘĚå■ŸæŸăåřĎäÿĂäÿłæŰĞäžűåźűäÿ■äijŽåŕijèĞť'æŢť'äÿłæŰĞäžűë
äź§åŕśæŸŕèŕť'ïijŇæŰĞäžűåźűæšąæIJL'èćńåď'■åĹűåĹřåĘĚå■ŸçijŞå■ŸæĹŰæŢřçżĎäÿ■ãĂĆçŻÿåŔ■ïijŇæŞ■ä¡

å¡Şä¡ãèő£éŮőæŰĞäžűçŽĎäÿ■åŘŇåŇžå§§æŮűïijŇè£ŹäžŻåŇžå§§çŽĎåĘĚåőźæĽ■æăźæ■őéIJĂèęĄećńèŕžåŕ
èĂŇéĆčäžŻäžŐæšąèćńèő£éŮőåĹřçŽĎéČĺåĹĘè£ŸæŸŕçŢŹåIJĺçčĄçŻŸäÿŁãĂĆæĽĂæIJĽè£ŹäžŻè£ĞçĺŃæŸ

åęĆæđIJåď´Žäÿł Pythonèğčé ĞŁåŹĺåĘĚå■ŸæŸăåŕĎåŘŇäÿĂäÿłæŰĞäžűïijŇå¿ŮåĹřçŽĎ
mmap åŕźèśąèČ¡åď§èćńçŤĺæİěåIJĺèğčéĞŁåŹĺçŻť æŐěäžď æ■ćæŢřæ■őãĂĆ
äź§åřśæŸŕèŕť´ïijŇæĽĂæIJĽèğčéĞŁåŹĺéČ¡èČ¡åŘŇæŮűèŕżåĘŹæŢřæ■őïijŇåźűäÿŤåĚűäÿ■äÿĂäÿłèğčéĞŁåŹĺa
å¿ĹæŸŐæŸ¿ïijŇè£ŹéĞŇéIJĂèęĄèĂČèŹŚåŘŇæ■ęçŽĎéŮőécŸãĂĆä¡ĘæŸŕè£Źçğ■æŰźæşŢæIJĽæŮűåĂŹåŕ

è£ŹäÿĂåřŔèŁĆäÿ■åĞ¡æŢřår¡éĞŔåĘŹå¿Ůå¿ĹéĂŽçŤĺïijŇåŘŇæŮűéĂĆçŤĺäžŐUnixåŠŇWindowsåźşåŔi
èęĄæşĺæĎŔçŽĎæŸŕä¡£çŤĺ mmap() åĞ¡æŢřæŮűäijŽåIJĺåžŢåśĆæIJĽäÿĂäžŽåźşåŔřçŽĎåůőåijĆæĂğãĂĆ
åŘęåď´ŮïijŇè£ŸæIJĽäÿĂäžŽéĂĽéąźåŔŕäžěçŤĺæİěåĹŻåżžåŇ£åŘ■çŽĎåĘĚå■ŸæŸăåřĎåŇžå§§ãĂĆ
åęĆæđIJä¡ăåŕźè£ŹäÿłæĎ§åĚť èűčïijŇçąőä£Ía¡ăäžŤçżĘçăŤèŕżäžĘPythonæŰĞæąčäÿ■
è£ŹæŰźéĺççŽĎåĘĚåőź ãĂĆ

5.11 æŰĞäžűèůŕå¿ĎåŘ■çŽĎæŞ■ä¡lJ

éŮőécŸ

ä¡ăéIJĂèęĄä¡£çŤĺèůŕå¿ĎåŘ■æİěèŐůåŔŰæŰĞäžűåŘ■ïijŇçŻőå¡ŢåŘ■ïijŇçżĺåŕźèůŕå¿Ďç■Ľç■Ľ ãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤĺ os.path æĺąåİŮäÿ■çŽĎåĞ¡æŢřæİěæŞ■ä¡IJèůŕå¿ĎåŘ■ãĂĆ
äÿŇéĺćæŸŕäÿĂäÿłäžď´äžŠåijŔä¿Ńå■ŘæİěæijŤçd´žäÿĂäžŽåĚşéŤőçŽĎçĽźæĂğïijŽ

```
>>> import os
>>> path = '/Users/beazley/Data/data.csv'

>>> # Get the last component of the path
>>> os.path.basename(path)
'data.csv'

>>> # Get the directory name
>>> os.path.dirname(path)
'/Users/beazley/Data'

>>> # Join path components together
>>> os.path.join('tmp', 'data', os.path.basename(path))
'tmp/data/data.csv'

>>> # Expand the user's home directory
>>> path = '~/Data/data.csv'
>>> os.path.expanduser(path)
'/Users/beazley/Data/data.csv'

>>> # Split the file extension
>>> os.path.splitext(path)
```

```
('~/Data/data', '.csv')
>>>
```

èőÍèőž

åŕźäžŐäżżä¡ŢçŽĎæŰĞäżűåŘ■çŽĎæŞ■ä¡IJïijŇä¡äéČ¡åžŤèŕěä¡£çŤÍ os.path æÍąåİŮïijŇèĂŇäÿ■æŸŕä¡£çŤÍæăĞåĞĘå■ŮçňęäÿšæŞ■ä¡IJæİěæđĎéĂäèĞłåůśçŽĎäżčçăĄãĂĆçĽźåĹńæŸŕäÿžäžĘåŔŕçğżæď■æĂğèĂČèŹŚçŽĎæŰűåĂŹæŻť åžŤåęĆæ■ď'ïijŇ åŻääÿž os.path æÍąåİŮç§ěéĄŞUnixåŠŇWindowsçşżçż§äźŃéŮť çŽĎåůőåijĆåźűäÿŤèČ¡åď'§åŔŕéÍăåIJřåď'ĎçŘĘçśżäijij Data/data.csv åŠŇ Data\data.csv è£ŹæăůçŽĎæŰĞäżűåŘ■ãĂĆ åĚűæňąïijŇä¡äçIJ§çŽĎäÿ■åžŤèŕěæţłèť źæŮűéŮť åŐżéĞ■åď'■éĂäè¡őå■ŘãĂĆéĂŽåÿÿæIJĂåě¡æŸŕçŻť æŐëä¡£

èęĄæşÍæĎŔçŽĎæŸŕ os.path è£ŸæIJL'æŻť åď'ŽçŽĎåŁ§èČ¡åIJÍè£ŹéĞŇåźűæšąæIJL'åĹŮäÿ¿åĞžæÍěã åŔŕäžěæ§ěéŸĚåőŸæŰźæŰĞæąčæÍèèŐůåŔŰæŻť åď'ŽäÿŐæŰĞäżűæţŃèŕŢïijŇçňęåŘůéŞ¿æŐęç■L'çŽÿåĚşçŽ

5.12 æţŃèŕŢæŰĞäżűæŸŕåŘęå■ŸåIJÍ

éŮőécŸ

ä¡ăæČşæţŃèŕŢäÿĂäÿłæŰĞäżűæĹŰçŻőå¡ŢæŸŕåŘęå■ŸåIJÍãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤÍ os.path æÍąåİŮæÍěæţŃèŕŢäÿĂäÿłæŰĞäżűæĹŰçŻőå¡ŢæŸŕåŘęå■ŸåIJÍãĂĆærŤåęĆïijŽ

```
>>> import os
>>> os.path.exists('/etc/passwd')
True
>>> os.path.exists('/tmp/spam')
False
>>>
```

ä¡ăè£ŸèČ¡è£ŽäÿĂæ■ěæţŃèŕŢè£ŹäÿłæŰĞäżűæŮűäżĂäźĹçśżåđŃçŽĎãĂĆ åIJÍäÿŃéİćè£ŹäžŽæţŃèŕŢäÿ■ïijŇåęĆæđIJæţŃèŕŢçŽĎæŰĞäżűäÿ■å■ŸåIJÍçŽĎæŮűåĂŹïijŇçżŞæđIJéČ¡äijŽè

```
>>> # Is a regular file
>>> os.path.isfile('/etc/passwd')
True

>>> # Is a directory
>>> os.path.isdir('/etc/passwd')
False

>>> # Is a symbolic link
```

```
>>> os.path.islink('/usr/local/bin/python3')
True

>>> # Get the file linked to
>>> os.path.realpath('/usr/local/bin/python3')
'/usr/local/bin/python3.3'
>>>
```

åęĆæđIJä¡ăè£ŸæČşèŐůåŔŰåĚČæŢřæ■ő(æŕŤåęĆæŰĞäžűåď˘ğåřŔæĹŰèĂĚæŸŕä£őæŤźæŮěæIJ§)ïijŇäź os.path æÍąåİŮæİěèğčåEşïijŽ

```
>>> os.path.getsize('/etc/passwd')
3669
>>> os.path.getmtime('/etc/passwd')
1272478234.0
>>> import time
>>> time.ctime(os.path.getmtime('/etc/passwd'))
'Wed Apr 28 13:10:34 2010'
>>>
```

èőÍèőž

ä¡£çŤÍ os.path æİěè£ŻèąŇæŰĞäžűæţŃèŕŢæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆ åIJĺåĘŹè£ŹäžŻèĎŽæIJňæŮűïijŇåŔŕèČ¡åŤŕäÿĂéIJĂèęĄæşĺæĎŔçŽĎåřśæŸŕä¡ăéIJĂèęĄèĂČèŹŚæŰĞäžűæİČ

```
>>> os.path.getsize('/Users/guido/Desktop/foo.txt')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "/usr/local/lib/python3.3/genericpath.py", line 49, in␣
 ↪getsize
        return os.stat(filename).st_size
PermissionError: [Errno 13] Permission denied: '/Users/guido/
 ↪Desktop/foo.txt'
>>>
```

5.13 èŐůåŔŰæŰĞäžűåď˙źäÿ■çŽĎæŰĞäžűåĹŮèąĺ

éŮőécŸ

ä¡ăæČşèŐůåŔŰæŰĞäžűçşżçż§äÿ■æ§ŘäÿłçŻőå¡ŢäÿŃçŽĎæĽĂæIJĽæŰĞäžűåĹŮèąĺãĂĆ

èğčåEşæŰźæąĹ

ä¡£çŤÍ os.listdir() åĞ¡æŢřæİěèŐůåŔŰæ§ŘäÿłçŻőå¡Ţäÿ■çŽĎæŰĞäžűåĹŮèąĺïijŽ

```
import os
names = os.listdir('somedir')
```

çżŞæđIJäijŽè£ŤåŻđçŻőå¡Ţäÿ■æĽĂæIJL'æŰĞäžűåĹŮèąĺïijŇåŇĚæŃňæĽĂæIJL'æŰĞäžűïijŇå■ŘçŻőå¡Ţ
åęĆæđIJä¡ăéIJĂèęĄéĂŽè£ĞæŞŘçğ■æŰźåijŔè£Ğæżď'æŢřæ■őïijŇåŕŕäžèèĂČèŹŚçżŞåŘĹ
`os.path` åžŞäÿ■çŽĎäÿĂäžŻåĞ¡æŢřæİěä¡£çŤĺåĹŮèąĺæŐĺåŕijãĂĆæŕŤåęĆïijŽ

```
import os.path

# Get all regular files
names = [name for name in os.listdir('somedir')
        if os.path.isfile(os.path.join('somedir', name))]

# Get all dirs
dirnames = [name for name in os.listdir('somedir')
        if os.path.isdir(os.path.join('somedir', name))]
```

å■Ůçňęäÿšç`ŽĎ `startswith()` åŠŇ `endswith()`
æŰźæşŢåŕźäžŐè£ĞæżďäÿĂäÿłçŻőå¡ŢçŽĎåĘĚåőźäź§æŸŕå¿ĹæIJL'çŤĺçŽĎãĂĆæŕŤåęĆïijŽ

```
pyfiles = [name for name in os.listdir('somedir')
           if name.endswith('.py')]
```

åŕźäžŐæŰĞäžűåŘ■çŽĎåŇźéĚ■ïijŇä¡ăåŔŕèČ¡äijŽèĂČèŹŚä¡£çŤĺ `glob` æĹŰ `fnmatch`
æĺąåİŮãĂĆæŕŤåęĆïijŽ

```
import glob
pyfiles = glob.glob('somedir/*.py')

from fnmatch import fnmatch
pyfiles = [name for name in os.listdir('somedir')
           if fnmatch(name, '*.py')]
```

èőĺèőž

èŐůåŔŰçŻőå¡Ţäÿ■çŽĎåĹŮèąĺæŸŕå¿ĹåőźæŸ§çŽĎïijŇä¡ĘæŸŕåĚűè£ŤåŻđçżŞæđIJåŔłæŸŕçŻőå¡Ţäÿ■åč
åęĆæđIJä¡ăè£ŸæČşèŐůåŔŰåĚűäžŰçŽĎåĚČä£ąæĄfïijŇæŕŤåęĆæŰĞäžűåď'ğåřŔïijŇä£őæŤźæŮűéŮť'ç■Ľ'ç■l
ä¡ăæĹŰèőÿè£ŸéIJĂèęĄä¡£çŤĺåĹř `os.path` æĺąåİŮäÿ■çŽĎåĞ¡æŢřæĹŰçİĂ `os.stat()`
åĞ¡æŢřæİěæŤűéŽĘæŢřæ■őãĂĆæŕŤåęĆïijŽ

```
# Example of getting a directory listing

import os
import os.path
import glob

pyfiles = glob.glob('*.py')
```

```
# Get file sizes and modification dates
name_sz_date = [(name, os.path.getsize(name), os.path.
↪getmtime(name))
                for name in pyfiles]
for name, size, mtime in name_sz_date:
    print(name, size, mtime)

# Alternative: Get file metadata
file_metadata = [(name, os.stat(name)) for name in pyfiles]
for name, meta in file_metadata:
    print(name, meta.st_size, meta.st_mtime)
```

æIJĂåŘŐè£ŸæIJL'äÿĂçĆźèęĄæşĺæĎŔçŽĎåřśæŸŕřijŇæIJL'æŮűåĂŹåIJĺåd'ĎçŘĘæŰĞäžűåŘ■çijŰçăĄé
éĂŽåÿÿæİěèőšïijŇåĞ¡æŢř `os.listdir()` è£ŤåŻđçŽĎåőđä¡ŞåĹŮèąĺäijŽæăźæ■őçşžçž§éžŸèőd'çŽĎæŰĞä
ä¡ĘæŸŕæIJL'æŮűåĂŹäź§äijŽçćřåĹřäÿĂäžŻäÿ■èČ¡æ■čåÿÿèğčçăĄçŽĎæŰĞäžűåŘ■ãĂĆ
åĚşäžŐæŰĞäžűåŘ■çŽĎåd'ĎçŘĘéŮőécŸïijŇåIJĺ5.14åŠŇ5.15åřŔèŁĆæIJL'æŻt'èŕęçżĘçŽĎèőšèğčãĂĆ

5.14 å£¡çŢěæŰĞäžűåŘ■çijŰçăĄ

éŮőécŸ

ä¡ăæČşä¡£çŤĺåŐ§ågŇæŰĞäžűåŘ■æL'ğèąŇæŰĞäžűçŽĎI/OæŞ■ä¡IJïijŇäź§åřśæŸŕèŕť'æŰĞäžűåŘ■åźűæš

èğčåĘşæŰźæąĹ

éżŸèőd'æČĚåĘţäÿŃïijŇæL'ĂæIJL'çŽĎæŰĞäžűåŘ■éČ¡äijŽæăźæ■ő `sys.getfilesystemencoding()` è£ŤåŻđçŽĎæŰĞæIJňçijŰçăĄæİěçijŰçăĄæĹŰèğčçăĄãĂĆæŕŤåęĆïijŽ

```
>>> sys.getfilesystemencoding()
'utf-8'
>>>
```

åęĆæđIJåŻääÿžæ§Řçğ■åŐ§åŻää¡ăæČşå£¡çŢěè£Źçğ■çijŰçăĄïijŇåŔŕäžěä¡£çŤĺäÿĂääÿłåŐ§ågŃå■ŮèŁĆå

```
>>> # Wrte a file using a unicode filename
>>> with open('jalape\xf1o.txt', 'w') as f:
...     f.write('Spicy!')
...
6
>>> # Directory listing (decoded)
>>> import os
>>> os.listdir('.')
['jalapeÃ±o.txt']

>>> # Directory listing (raw)
>>> os.listdir(b'.') # Note: byte string
```

```
[b'jalapen\xcc\x83o.txt']

>>> # Open file with raw filename
>>> with open(b'jalapen\xcc\x83o.txt') as f:
...     print(f.read())
...
Spicy!
>>>
```

æ■čåęĆä¡ăæĽĂèğĄïijŇåIJÍæIJĂåŘŐäÿd'äÿłæŞ■ä¡IJäÿ■ïijŇå¡Şä¡ăçżŹæŰĞäžűçŻÿåĚşåĞ¡æŢřåęĆ
`open()` åŠŇ `os.listdir()` äijăéĂŠå■ÚèŁĆå■ŮçňęäÿšæŮűïijŇæŰĞäžűåŘ■çŽĎåd'ĎçŘĘæŰźåijŔäijŽçĺ

èőĺèőž

éĂŽåÿÿæİěèőšïijŇä¡ăäÿ■éIJĂèęĄæŃĚå£ČæŰĞäžűåŘ■çŽĎçijŰçăĄåŠŇèğčçăĄïijŇæŹőéĂŽçŽĎæŰĞäž
ä¡ĘæŸŕïijŇæIJL'äžŻæŞ■ä¡IJçşżçż§åĚĄèőÿçŤĺæĹůéĂŽè£ĞåĄűçĎűæĹŰæĄűæĎŔæŰźåijŔåŐżåĹŻåżžåŘ■å■
è£ŹäžŻæŰĞäžűåŘ■åŔŕèČ¡äijŽçěđçğŸåIJŕäÿ■æŰ■éĆčäžŻéIJĂèęĄåd'ĎçŘĘåd'ğéĞŔæŰĞäžűçŽĎPythonçĺŃ

èŕżåŔŰçŽőå¡ŢåźűéĂŽè£ĞåŐ§ågŇæIJłèğčçăĄæŰźåijŔåd'ĎçŘĘæŰĞäžűåŘ■åŔŕäžěæIJL'æŢĹçŽĎéĄ£å
år¡çőąè£ŹæăůäijŽåÿęæİěäÿĂåőŽçŽĎçijŰçĺŃéŽ¿åžęãĂĆ

åĚşäžŐæL'Şå■řäÿ■åŔŕèğčçăĄçŽĎæŰĞäžűåŘ■ïijŇèŕůåŔĆèĂČ5.15åřŔèŁĆãĂĆ

5.15 æL'Şå■řäÿ■åŘĹæşŢçŽĎæŰĞäžűåŘ■

éŮőécŸ

ä¡ăçŽĎçĺŃåžŔèŐůåŔŰäžĘäÿĂäÿłçŻőå¡Ţäÿ■çŽĎæŰĞäžűåŘ■åĹŮèąĺïijŇä¡ĘæŸŕå¡ŞåőČèŕŢçİĂåŐżæL'Ş
åĞžçŐřäžĘ `UnicodeEncodeError` åijĆåÿÿåŠŇäÿĂæİąåěĞæĂłçŽĎæűĹæĄŕâĂŤâĂŤ
`surrogates not allowed` ãĂĆ

èğčåĘşæŰźæąĹ

å¡ŞæL'Şå■řæIJłç§ěçŽĎæŰĞäžűåŘ■æŮűïijŇä¡£çŤĺäÿŃéİćçŽĎæŰźæşŢåŔŕäžěéĄ£åĚ■è£ŹæăůçŽĎéŤŹè

```
def bad_filename(filename):
    return repr(filename)[1:-1]

try:
    print(filename)
except UnicodeEncodeError:
    print(bad_filename(filename))
```

èőĺèőž

è£ŹäÿĂåřŔèŁĆèőĺèőžçŽĎæŸŕåIJĺçijŰåEŹå£ĚéąžåďĎçŘĘæŰĞäžűçşžçž§çŽĎçĺŃåžŔæŮűäÿĂäÿłäÿ■åd
éžŸèőďæČĚåĘţäÿŃïijŇPythonåĄĞåőŽæĽĂæIJL'æŰĞäžűåŘ■éČ¡åůšçżŔæăźæ■ő
`sys.getfilesystemencoding()` çŽĎåĂijçijŰçăĄè£ĞäžĘãĂĆ
ä¡ĘæŸŕïijŇæIJL'äÿĂäžŽæŰĞäžűçşžçž§åźűæšąæIJL'åijžåĹűèęĄæśĆè£ŹæăůåĄŽïijŇåŽăæ■ď'åĚĄèőÿåĹŽåžž
è£ŹçğæČĚåĘţäÿ■åď'łåÿÿèğĄïijŇä¡ĘæŸŕæĂżäijŽæIJL'äžŽçŤĺæĹůåĘŠéŹľè£ŹæăůåĄŽæĹŰèĂĚæŸŕæŮăæĽ
åŔŕèČ¡æŸŕåIJĺäÿĂäÿłæIJL'çijžéŹůçŽĎäžčçăĄäÿ■çżŹ `open()`
åĞ¡æŢřäijăéĂŠäžĘäÿĂäÿłäÿ■åŘĹèğĎèŇČçŽĎæŰĞäžűåŘ■)ãĂĆ

å¡ŞæĽğèąŇçśżäijij `os.listdir()` è£ŹæăůçŽĎåĞ¡æŢřæŮűïijŇè£ŹäžŻäÿ■åŘĹèğĎèŇČçŽĎæŰĞäžűź
äÿĂæŰźéİćïijŇåőČäÿ■èČ¡äžĚäžĚåŘłæŸŕäÿćåijČè£ŹäžŻäÿ■åŘĹæăijçŽĎåŘ■å■ŮãĂĆèĂŇåŔęäÿĂæŰźéİćïij
Pythonåŕźè£ŹäÿłéŮőéćŸçŽĎèğčåĘşæŰźæąĹæŸŕäžÔæŰĞäžűåŘ■äÿ■èÔůåŘŰæIJłèğčçăĄçŽĎå■ŮèŁĆåĂijæ
`\xhh` åźűåřĘåőČæŸăåřĎæĹŘUnicodeå■Ůçňę `\udchh` èąĺçď'žçŽĎæĽ'ĂèřŞçŽĎ"äžčçŘĘçijŰçăĄ"ãĂĆ
äÿŃéİćäÿĂäÿłä¿Ńå■ŘæijŤçď'žäžĘå¡ŞäÿĂäÿłäÿ■åŘĹæăijçŽőå¡ŢåĹŮèąĺäÿ■åŘńæIJL'äÿĂäÿłæŰĞäžűåŘ■äÿžb
1èĂŇäÿ■æŸŕUTF-8çijŰçăĄ)æŮűçŽĎæăůå■ŘïijŽ

```
>>> import os
>>> files = os.listdir('.')
>>> files
['spam.py', 'b\udce4d.txt', 'foo.txt']
>>>
```

åęĆæđIJä¡ăæIJL'äžčçăĄéIJĂèęĄæŞ■ä¡IJæŰĞäžűåŘ■æĹŰèĂĚårĘæŰĞäžűåŘ■äijăéĂŠçżŹ
`open()` è£ŹæăůçŽĎåĞ¡æŢřïijŇäÿĂåĹĞéČ¡èČ¡æ■čåÿÿåůëä¡IJãĂĆ
åŘłæIJL'å¡Şä¡ăæČşèęĄè¿ŞåĞžæŰĞäžűåŘ■æŮűæL'■äijŽçćřåĹřäžŻéžžçČę(æŕŤåęĆæL'Şå■řè¿ŞåĞžåĹřåśŔåź
çL'źåĹńçŽĎïijŇå¡Şä¡ăæČşæL'Şå■řäÿŁéİćçŽĎæŰĞäžűåŘ■åĹŮèąĺæŮűïijŇä¡ăçŽĎçĺŃåžŔårśäijŽåť'l'æžČïijŽ

```
>>> for name in files:
...     print(name)
...
spam.py
Traceback (most recent call last):
    File "<stdin>", line 2, in <module>
UnicodeEncodeError: 'utf-8' codec can't encode character '\udce4' in
position 1: surrogates not allowed
>>>
```

çĺŃåžŔåť'l'æžČçŽĎåŐ§åŽăářśæŸŕå■Ůçňę `\udce4` æŸŕäÿĂäÿłéİđæşŢçŽĎUni-
codeå■Ůçňęă ĂĆ åőČåĚűåőđæŸŕäÿĂäÿłèćńçğřäÿžäžčçŘĘå■ŮçňęářźçŽĎåŔŇå■ŮçňęçžĎåŘĹçŽĎåŘŐå■ŁéČ
çŤśäžŐçijžářŠäžĘåĽ■å■ŁéČĺåĹĘïijŇåŽăæ■ď'åőČæŸŕäÿłéİđæşŢçŽĎUnicodeă ĂĆ
æĽĂäžěïijŇåŤŕäÿĂèČ¡æĹŘåŁ§è¿ŞåĞžçŽĎæŰźæşŢářśæŸŕå¡ŞéĄĞåĹŕäÿ■åŘĹæşŢæŰĞäžűåŘ■æŮűéĞĞåŔ
æŕŤåęĆåŔŕäžěářĘäÿŁè£ŕäžčçăĄä£őæŤźåęĆäÿŃïijŽ

```
>>> for name in files:
... try:
...     print(name)
... except UnicodeEncodeError:
...     print(bad_filename(name))
...
```

```
spam.py
b\udce4d.txt
foo.txt
>>>
```

åIJÍ bad_filename() åĞ¡æŢřäÿ■æĂŐæăůåď Ďç¡őåŔŰåĘşäžŐä¡äèĞłåůśãĂĆ åŔęåď ŰäÿĂäÿłéĂĽ'æŃĽ'åřśæŸŕéĂŽè£Ğæ§Řçğ■æŰźåijŔéĞ■æŰřçijŰçăĄïijŇçď'žä¿ŃåęĆäÿŃïijŽ

```
def bad_filename(filename):
    temp = filename.encode(sys.getfilesystemencoding(), errors=
↪'surrogateescape')
    return temp.decode('latin-1')
```

èŕŚèĂĚæşÍ:

```
surrogateescape:
è£Źçğ■æŸŕPythonåIJÍçżİåď'ğéČĺåĹĘéİćåŘŚOSçŽĎAPIäÿ■æL'Ăä¡£çŤÍçŽĎéŤźèŕŕåď'ĎçŘĘåŹĺïi
åőČèČ¡äžěäÿĂçğ■äijŸéŽĚçŽĎæŰźåijŔåď'ĎçŘĘçŤśæŞ■ä¡IJçşżçżŞæŔŘä¿ŽçŽĎæŢřæ■őçŽĎçijŰçăĄ
åIJÍèğčçăĄåĞžéŤźæŮűäijŽåřĘåĞžéŤźå■ÛèŁĆå■ŸåĆĺåĹřäÿĂäÿłå¿ĹåřŚèćńä¡£çŤĺåĹřçŽĎUnicode
åIJÍçijŰçăĄæŮűåřĘéĆčäžŽéŽŘèŮŔåĂijåŔĹè£ŸåŐ§åŻđåŐ§åĚĹèğčçăĄåd'śèt'ěçŽĎå■ÛèŁĆåžŔåĹŮ
åőČäÿ■äžĚåŕźäžŐOS␣
↪APIéİđåÿÿæIJL'çŤÍïijÑäź§èČ¡å¿ĹåőźæŸŞçŽĎåd'ĎçŘĘåĚűäżŰæČĚåĘţäÿŃçŽĎçijŰçăĄéŤźèŕŕã
```

ä¡£çŤÍè£ŹäÿłçĽĹæIJňäžğçŤ§çŽĎè¿ŞåĞžåęĆäÿŃïijŽ

```
>>> for name in files:
...     try:
...         print(name)
...     except UnicodeEncodeError:
...         print(bad_filename(name))
...
spam.py
bÃd'd.txt
foo.txt
>>>
```

è£ŹäÿĂåřŔèŁĆäÿžécŸåŔŕèČ¡äijŽèćńåď'ğéČĺåĹĘèŕżèĂĚæL'Ăå£¡çŢěãĂĆä¡ĘæŸŕåęĆædIJä¡ăåIJÍçijŰåĘ
åřśå£Ěéąžå¿ŮèĂČèŹŚåĹřè£ŹäÿłãĂĆåŘęåĹŹä¡ăåŔŕèČ¡äijŽåIJÍæ§Řäÿłå ŚÍæIJńèćńåŔńåĹřåŁđåĚňåőd'åŐżèřČ

5.16 åćđåŁăæĹŰæŤźåŔŸåůšæL'ŞåijĂæŰĞäžűçŽĎçijŰçăĄ

éŮőécŸ

ä¡ăæČşåIJÍäÿ■åĚşéŮ■äÿĂäÿłåůšæL'ŞåijĂçŽĎæŰĞäžűåL'■æŔŘäÿŃåćđåŁăæĹŰæŤźåŔŸåőČçŽĎUnicod

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşçżŹäÿĂäÿłäžëäžŇè£ŻåĹűæĺąåijŔæĽŞåijĂçŽĎæŰĞäžűæůżåŁăUnicodeçijŰçăĄ/èğčçăĄ
åŕŕäžëä¡£çŤĺ `io.TextIOWrapper()` åŕźèśąåŇĚèčĚåőČãĂĆæŕŤåęĆïijŽ

```
import urllib.request
import io

u = urllib.request.urlopen('http://www.python.org')
f = io.TextIOWrapper(u, encoding='utf-8')
text = f.read()
```

åęĆæđIJä¡ăæČşä£őæŤźäÿĂäÿłåůšçżŔæĽŞåijĂçŽĎæŰĞæIJñæĺąåijŔçŽĎæŰĞäžűçŽĎçijŰçăĄæŰźåijŔ
`detach()` æŰźæşŢçğżéŹd'æŐĽåůšå■ŸåIJĺçŽĎæŰĞæIJñçijŰçăĄåśĆïijŇ
åźűä¡£çŤĺæŰřçŽĎçijŰçăĄæŰźåijŔäžčæŻ£ãĂĆäÿŇéĺćæŸŕäÿĂäÿłåIJĺ `sys.stdout`
äÿŁä£őæŤźçijŰçăĄæŰźåijŔçŽĎä¿Ńå■ŘïijŽ

```
>>> import sys
>>> sys.stdout.encoding
'UTF-8'
>>> sys.stdout = io.TextIOWrapper(sys.stdout.detach(), encoding=
↪'latin-1')
>>> sys.stdout.encoding
'latin-1'
>>>
```

è£Źæăůå˛ŽåŔŕèČ¡äijŽäÿ■æŰ■ä¡ăçŽĎçżĹçńŕïijŇè£ŹéĞŇäžĚäžĚæŸŕäÿžäžĘæijŢçd'žèĂŇåůšãĂĆ

èőĺèőž

I/Oçşżçż§çŤśäÿĂçşżåĹŮçŽĎåśĆæňąædĎåžžèĂŇæĹŘãĂĆä¡ăåŔŕäžëèŕŢçİÀè£ŘèąŇäÿŃéĺćè£ŹäÿłæŞ■ä

```
>>> f = open('sample.txt','w')
>>> f
<_io.TextIOWrapper name='sample.txt' mode='w' encoding='UTF-8'>
>>> f.buffer
<_io.BufferedWriter name='sample.txt'>
>>> f.buffer.raw
<_io.FileIO name='sample.txt' mode='wb'>
>>>
```

åIJĺè£Źäÿłä¿Ńå■ŘäÿïijŇ`io.TextIOWrapper` æŸŕäÿĂäÿłçijŰçăĄåŠŇèğčçăĄU-
nicodeçŽĎæŰĞæIJñåď˘ĎçŘĘåśĆïijŇ `io.BufferedWriter`
æŸŕäÿĂäÿłåď˘ĎçŘĘäžŇè£ŻåĹűæŢŕæ■őçŽĎåÿęçijŞåĘšçŽĎI/OåśĆïijŇ `io.FileIO`
æŸŕäÿĂäÿłèąĺçď˘žæŞ■ä¡IJçşżçż§åžŢåśĆæŰĞäžűæŔŘè£ŕçňęçŽĎåŐ§ågŃæŰĞäžűãĂĆ
åćđåŁäæĹŰæŤźåŔŸæŰĞæIJñçijŰçăĄäijŽæűL'åŔŁåćđåŁäæĹŰæŤźåŔŸæIJĂäÿŁéĺćçŽĎ
`io.TextIOWrapper` åśĆãĂĆ

äÿĂèĹňæİèèőšïijŇåČŘäÿŁéİćä¿Ńå■Řè£ŹæăůéĂŽè£Ğèő£éŮőåšđæĂğåĂijæİěçŻť'æŐěæŞ■ä¡IJäÿ■åŘŇçä¿ŃåęĆïijŇåęĆæđIJä¡ãèŕŢçİĂä¡£çŤĺäÿŃéİćè£ŹæăůçŽĎæŁĂæIJŕæŤźåŔŸçijŰçăĄçIJŃçIJŃäijŽåŘŠçŤ§äżĂä

```
>>> f
<_io.TextIOWrapper name='sample.txt' mode='w' encoding='UTF-8'>
>>> f = io.TextIOWrapper(f.buffer, encoding='latin-1')
>>> f
<_io.TextIOWrapper name='sample.txt' encoding='latin-1'>
>>> f.write('Hello')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ValueError: I/O operation on closed file.
>>>
```

çżŞæđIJåĞžéŤŹäžĘïijŇåŻääÿžfçŽĎåŐ§ågŃåĂijåůšçżŔèćńçăť'åİŔäžĘåźűåĚşéŮ■äžĘåžŢåśĆçŽĎæŰĞäz

detach() æŰźæşŢäijŽæŰ■åijĂæŰĞäžűçŽĎæIJĂéąűåśĆåźűè£ŤåŻđçňňäžŇåśĆïijŇäźŃåŘŐæIJĂéąűå

```
>>> f = open('sample.txt', 'w')
>>> f
<_io.TextIOWrapper name='sample.txt' mode='w' encoding='UTF-8'>
>>> b = f.detach()
>>> b
<_io.BufferedWriter name='sample.txt'>
>>> f.write('hello')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ValueError: underlying buffer has been detached
>>>
```

äÿĂæŮęæŰ■åijĂæIJĂéąűåśĆåŘŐïijŇä¡ăářśåŔŕäžěçżŹè£ŤåŻđçżŞæđIJæůżåŁääÿĂäÿłæŰřçŽĎæIJĂéąűå

```
>>> f = io.TextIOWrapper(b, encoding='latin-1')
>>> f
<_io.TextIOWrapper name='sample.txt' encoding='latin-1'>
>>>
```

ář¡çőąåůšçżŔåŘŚä¡ăæijŤçď'žäžĘæŤźåŔŸçijŰçăĄçŽĎæŰźæşŢïijŇ
ä¡ĘæŸŕä¡ăè£ŸåŔŕäžěåĹ'çŤĺè£Źçğ■æŁĂæIJŕæİěæŤźåŔŸæŰĞäžűèąŇåď'ĎçŘĘãĂĄéŤŹèŕŕæIJžåĹűäžěåŘŁæ

```
>>> sys.stdout = io.TextIOWrapper(sys.stdout.detach(), encoding=
↪'ascii',
...                                errors='xmlcharrefreplace')
>>> print('Jalape\u00f1o')
Jalape&#241;o
>>>
```

æşĺæĎŔäÿŃæIJĂåŘŐè¿ŞåĞžäÿ■çŽĎéIđASCIIå■Ůçňę Ãś æŸŕåęĆä¡Ţèćń ñ åŘŰäžčçŽĎãĂĆ

5.17 åřĘå■ÙèŁĆåĘŹåĚěæŰĞæIJňæŰĞäžű

éŮőécŸ

ä¡ăæČşåIJĺæŰĞæIJňæĺąåijŔæĽŞåijĂçŽĎæŰĞäžűäÿ■åĘŹåĚěåŐ§ågŃçŽĎå■ÙèŁĆæŢřæ■őãĂĆ

èğčåĘşæŰźæąĹ

åřĘå■ÙèŁĆæŢřæ■őçŻťæŐěåĘŹåĚěæŰĞäžűçŽĎçijŞåĘšåŇžå■şåŔŕïijŇä¿ŃåęĆïijŽ

```
>>> import sys
>>> sys.stdout.write(b'Hello\n')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: must be str, not bytes
>>> sys.stdout.buffer.write(b'Hello\n')
Hello
5
>>>
```

çśżäijijçŽĎïijŇèČ¡åď§éĂŽè£ĞèŕżåŔŰæŰĞæIJňæŰĞäžűçŽĎ buffer åśđæĂğæİèèŕżåŔŰäžŇè£ŻåĹűæŢřæ■őãĂĆ

èőĺèőž

I/Oçşżçż§äžěåśĆçžğçżŞæđĎçŽĎå¡ćåijŔæđĎåżžèĂŇæĹŘãĂĆ
æŰĞæIJňæŰĞäžűæŸŕéĂŽè£ĞåIJĺäÿĂäÿłæŃěæIJL'çijŞåĘšçŽĎäžŇè£ŻåĹűæĺąåijŔæŰĞäžűäÿŁåćđåŁääÿĂäÿ
buffer åśđæĂğæŇĞåŘŚåŕźåžŢçŽĎåžŢåśĆæŰĞäžűãĂĆåęĆæđIJä¡äçŻťæŐèèő£éŮőåőĆçŽĎèŕİåŕśäijŽçżŢè

æIJňåřŔèŁĆä¿Ńå■ŘåśŢçd'žçŽĎ sys.stdout åŔŕèČ¡çIJŃètůæİěæIJL'çĆźçL'źæőŁãĂĆ
éżŸèőd'æČĚåĘţäÿŃïijŇsys.stdout æĂżæŸŕäžěæŰĞæIJňæĺąåijŔæL'ŞåijĂçŽĎãĂĆ
ä¡ĘæŸŕåęĆæđIJä¡ăåIJĺåĘŹäÿĂäÿłéIJĂèęĄæL'Şå■ŕäžŇè£ŻåĹűæŢřæ■őåĹŕæăĞåĞĘè¿ŞåĞžçŽĎèĎŽæIJňçŽĎ

5.18 åřĘæŰĞäžűæŔŔè£ŕçňęåŇĚèčĚæĹŘæŰĞäžűåŕźèśą

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłåŕźåžŤäžŐæŞ■ä¡IJçşżçż§äÿŁäÿĂäÿłåůšæL'ŞåijĂçŽĎI/OéĂŽéĄŞ(æŕŤåęĆæŰĞäžűãĂĄç
ä¡ăæČşåřĘåőČåŇĚèčĚæĹŘäÿĂäÿłæŻťéńŸåśĆçŽĎPythonæŰĞäžűåŕźèśąãĂĆ

èğčåĘşæŰźæąĹ

äÿĂäÿłæŰĞäžűæŔŔè£řçňęåŠŇäÿĂäÿłæĽŞåijĂçŽĎæŹőéĂŽæŰĞäžűæŸŕäÿ■äÿĂæăůçŽĎãĂĆ æŰĞäžűæŔŔè£řçňęäžĚäžĚæŸŕäÿĂäÿłçŤśæŞ■ä¡IJçşżçż§æŇĞåőŽçŽĎæŢť æŢřïijŇçŤĺæĚěæŇĞäžčæ§Řäÿłçş åęĆæđIJä¡ăçćřåůğæIJĽè£ŹäźĹäÿĂäÿłæŰĞäžűæŔŔè£řçňęïijŇä¡ăåŔŕäžěéĂŽè£Ğä¡£çŤÍ `open()` åĞ¡æŢřæİěářĘåĚűåŇĚèčĚäÿžäÿĂäÿłPythonçŽĎæŰĞäžűåŕźèśąãĂĆ ä¡ääžĚäžĚåŘłéIJĂèęĄä¡£çŤĺè£ŹäÿłæŢť æŢřåĂijçŽĎæŰĞäžűæŔŔè£řçňęä¡IJäÿžçňňäÿĂäÿłåŔĆæŢřæİěäžčæŽ

```
# Open a low-level file descriptor
import os
fd = os.open('somefile.txt', os.O_WRONLY | os.O_CREAT)

# Turn into a proper file
f = open(fd, 'wt')
f.write('hello world\n')
f.close()
```

å¡ŞéńŸåśĆçŽĎæŰĞäžűåŕźèśąèćńåĚşéŮ■æĹŰèĂĚçăť åİŔçŽĎæŮűåĂŹïijŇåžŢåśĆçŽĎæŰĞäžűæŔŔè£ì åęĆæđIJè£Źäÿłåźűäÿ■æŸŕä¡ăæČşèęĄçŽĎçżŞæđIJïijŇä¡ăåŔŕäžěçżŹ `open()` åĞ¡æŢřäijåéĂŠäÿĂäÿłåŔŕéĂĽçŽĎ `colsefd=False` ãĂĆærŤåęĆïijŽ

```
# Create a file object, but don't close underlying fd when done
f = open(fd, 'wt', closefd=False)
...
```

èőĺèőž

åIJĺUnixçşżçż§äÿ■ïijŇè£Źçğ■åŇĚèčĚæŰĞäžűæŔŔè£řçňęçŽĎæŁĂæIJŕåŔŕäžěå¿ĹæŰźä¿£çŽĎåřĘäÿĂä åęĆçőąéĄŞãĂĄåěŮæŐěå■Ůç■Ľ˜ãĂĆäÿ¿ä¿ŃæİěëőšïijŇäÿŃéİćæŸŕäÿĂäÿłæŞ■ä¡IJçőąéĄŞçŽĎä¿Ńå■ŘïijŽ

```
from socket import socket, AF_INET, SOCK_STREAM

def echo_client(client_sock, addr):
    print('Got connection from', addr)

    # Make text-mode file wrappers for socket reading/writing
    client_in = open(client_sock.fileno(), 'rt', encoding='latin-1',
                     closefd=False)

    client_out = open(client_sock.fileno(), 'wt', encoding='latin-1
↪',
                      closefd=False)

    # Echo lines back to the client using file I/O
    for line in client_in:
        client_out.write(line)
        client_out.flush()

    client_sock.close()
```

```
def echo_server(address):
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(address)
    sock.listen(1)
    while True:
        client, addr = sock.accept()
        echo_client(client, addr)
```

éIJĂèęĄéĞ■çĆźåijžèřČçŽĎäÿĂçĆźæŸŕïijŇäÿŁéÍćçŽĎä¿Ńå■ŘäżĚäžĚæŸŕäÿžäžĘæijŢçď žåĘĚç¡őçŽĎ `open()` åĞ¡æŢřçŽĎäÿĂäÿłçĽźæĂğïijŇåźűäÿŤäź§åŔłéĂĆçŢĺäžŐå§žäžŐUnixçŽĎçşżçż§ãĂĆ åęĆæđIJä¡ăæČşåřĘäÿĂäÿłçśżæŰĞäżűæŐěåŔčä¡IJçŢĺåIJĺäÿĂäÿłåěŮæŐěå■Ůåźűåÿ ŇæIJŻä¡ăçŽĎäżčçăĄåŔŕä `makefile()` æŰźæşŢãĂĆ ä¡ĘæŸŕåęĆæđIJäÿ■èĂČèŹŚåŔŕçğżæd'■æĂğçŽĎèŕÏijŇéĆčäÿŁéÍćçŽĎèğčåĘşæ `makefile()` æĂğèČ¡æŻť'åě¡äÿĂçĆźãĂĆ

ä¡ăäź§åŔŕäżěä¡£çŤĺè£Źçğ■æŁĂæIJŕæÌěædĎéĂääÿĂäÿłåĹńåŘ■ïijŇåĚĄèőÿäżěäÿ■åŘŇäžŐçňňäÿĂæňąąä¿ŃåęĆïijŇäÿŃéÍćæijŢçď žåęĆä¡ŢåĹŻåżžäÿĂäÿłæŰĞäżűåŕźèśąïijŇåőČåĚĄèőÿä¡äè¿ŞåĞžäžŇè£ŻåĹűæŢřæ

```
import sys
# Create a binary-mode file for stdout
bstdout = open(sys.stdout.fileno(), 'wb', closefd=False)
bstdout.write(b'Hello World\n')
bstdout.flush()
```

åř¡çőąåŔŕäżěåřĘäÿĂäÿłåůšå■ŸåIJĺçŽĎæŰĞäżűæŔŔè£řçňęåŇĚèčĚæĹŘäÿĂäÿłæ■čåÿÿçŽĎæŰĞäżűåŕźè ä¡ĘæŸŕèęĄæşÍæĎŔçŽĎæŸŕåźűäÿ■æŸŕæĽĂæIJĽçŽĎæŰĞäżűæÍąåijŔéČ¡èćńæŤŕæŇĄïijŇåźűäÿŤæ§ŘäžŻçš (çĽźåĹńæŸŕæűĽåŔŁåĹřéŤŹèŕŕåď ĎçŘĘãĂĄæŰĞäżűçżŞář¿æİąäžűç■Ľç■Ľç ŽĎæŮűåĂŹ)ãĂĆ åIJĺäÿ■åŘŇçŽĎæŞ■ä¡IJçşżç§äÿŁè£Źçğ■èąŇäÿžäź§æŸŕäÿ■äÿĂæăůïijŇçĽźåĹńçŽĎïijŇäÿŁéÍćçŽĎä¿Ńå■Ře æĹŚèŕťäžĘè£ŹäźĹåď ŽïijŇæĎŔæĂİåŕśæŸŕèőľä¡ăåĚĚåĹĘæţŃèŕŢèĞłåůśçŽĎåőđçŐřäžčçăĄïijŇçąőä£İåőČè

5.19 åĹŻåżžäÿťæŮűæŰĞäżűåŠŇæŰĞäżűåď ź

éŮőécŸ

ä¡ăéIJĂèęĄåIJĺçĺŃåžŔæĽğèąŇæŮűåĹŻåżžäÿĂäÿłäÿťæŮűæŰĞäżűæĹŰçŻőå¡ŢïijŇåźűåÿŇæIJŻä¡£çŤĺå

èğčåĘşæŰźæąĹ

`tempfile` æĺąåİŮäÿ■æIJĽå¿Ĺåď ŽçŽĎåĞ¡æŢřåŕŕäžěåőŇæĹŘè£ŹäżżåŁąãĂĆ äÿžäžĘåĹŻåżžäÿĂäÿłåŇ£åŘ■çŽĎäÿťæŮűæŰĞäżűïijŇåŕŕäžěä¡£çŤĺ `tempfile.TemporaryFile` ïijŽ

```
from tempfile import TemporaryFile

with TemporaryFile('w+t') as f:
    # Read/write to the file
```

```
    f.write('Hello World\n')
    f.write('Testing\n')

    # Seek back to beginning and read the data
    f.seek(0)
    data = f.read()

# Temporary file is destroyed
```

æĹŰèĂĚïijŇåęĆæđIJä¡ăåŰIJæňćïijŇä¡ãè£ŸåŔŕäžěåČŔè£Źæăůä¡£çŤĺäÿť æŮűæŰĞäžűïijŽ

```
f = TemporaryFile('w+t')
# Use the temporary file
...
f.close()
# File is destroyed
```

TemporaryFile() çŽĎçňňäÿĂäÿłåŔĆæŢřæŸŕæŰĞäžűæĺąåijŔïijŇéĂŽåÿÿæİěèőšæŰĞæIJňæĺąåijŔä w+t ïijŇäžŇè£ŻåĹűæĺąåijŔä¡£çŤĺ w+b ãĂĆ è£ŹäÿłæĺąåijŔåŘŇæŮűæŤŕæŇĄèŕžåŠŇåĘŹæŞ■ä¡IJïijŇåIJĺè£Ź TemporaryFile() åŔęåď Űè£ŸæŤŕæŇĄèů§åĘĚç¡őçŽĎ open() åĞ¡æŢřäÿĂæăůçŽĎåŔĆæŢřãĂĆæŕŤåęĆïijŽ

```
with TemporaryFile('w+t', encoding='utf-8', errors='ignore') as f:
    ...
```

åIJĺåď ğåď ŽæŢřUnixçşżçż§äÿŁïijŇéĂŽè£Ğ TemporaryFile() åĹŻåžžçŽĎæŰĞäžűéČ¡æŸŕåŇ£åŘ■çŽĎïijŇçŤŽèĞşè£đçŻőå¡ŢéČ¡æšąæIJL'ãĂĆ åęĆæđIJä¡ăæČşæL'Şçăť è£ŹäÿłéŹŘåĹűïijŇåŔŕäžěä¡£çŤĺ NamedTemporaryFile() æİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
from tempfile import NamedTemporaryFile

with NamedTemporaryFile('w+t') as f:
    print('filename is:', f.name)
    ...

# File automatically destroyed
```

è£ŹéĞŇïijŇèćńæL'ŞåijĂæŰĞäžűçŽĎ f.name åšđæĂğåŇĚåŘńäžĘèŕěäÿť æŮűæŰĞäžűçŽĎæŰĞäžűåŘ å¡Şä¡ăéIJĂèęĄåřĘæŰĞäžűåŘ■äijăéĂŠçžŹåĚűäžŰäžčçăĄæİěæL'ŞåijĂè£ŹäÿłæŰĞäžűçŽĎæŮűåĂŹïijŇè£Źäÿ åŠŇ TemporaryFile() äÿĂæăůïïijŇçžŞæđIJæŰĞäžűåĚşéÚ■æŮűäijŽèćńèĞłåŁĺåĹăéŹď æŐL'ãĂĆ åęĆæđIJä¡ăäÿ■æČşè£ŹäźĹåĄŽïijŇåŔŕäžěäijăéĂŠäÿĂäÿłåĚşéŤőå■ŮåŔĆæŢř delte=False å■şåŔŕãĂĆæŕŤåęĆïijŽ

```
with NamedTemporaryFile('w+t', delete=False) as f:
    print('filename is:', f.name)
    ...
```

äÿžäžĘåĹŻåžžäÿĂäÿłäÿť æŮűçŻőå¡ŢïijŇåŔŕäžěä¡£çŤĺ tempfile.TemporaryDirectory() ãĂĆæŕŤåęĆïijŽ

```
from tempfile import TemporaryDirectory

with TemporaryDirectory() as dirname:
    print('dirname is:', dirname)
    # Use the directory
    ...
# Directory and all contents destroyed
```

èőÍèőž

TemporaryFile() ãĂĄNamedTemporaryFile() åŠŇ TemporaryDirectory() åĞ¡æŢř åžŤèřěæŸřåď˘ĎçŘĘäÿť æŮűæŰĞäžűçŽőå¡ŢçŽĎæIJĂçőĂå■ŢçŽĎæŰź åIJÍäÿĂäÿłæŻť ä¡ŐçŽĎçžğåĹńïijŇä¡ăåŘŕäžěä¡£çŤÍ mkstemp() åŠŇ mkdtemp() æÌěåĹŻåžžäÿť æŮűæŰĞäžűåŠŇçŽőå¡ŢãĂĆæŕŤåęĆïijŽ

```
>>> import tempfile
>>> tempfile.mkstemp()
(3, '/var/folders/7W/7WZl5sfZEF0pljrEB1UMWE+++TI/-Tmp-/tmp7fefhv')
>>> tempfile.mkdtemp()
'/var/folders/7W/7WZl5sfZEF0pljrEB1UMWE+++TI/-Tmp-/tmp5wvcv6'
>>>
```

ä¡ĘæŸŕïijŇè£ŹäžŻåĞ¡æŢřåźűäÿ■äijŽåĄŽè£ŻäÿĂæ■ęçŽĎçőąçŘĘäžĘãĂĆ ä¿ŇåęĆïijŇåĞ¡æŢř mkstemp() äžĚäžĚåřśè£ŤåŻđäÿĂäÿłåŐ§ågŇçŽĎOSæŰĞäžűæŔŘè£řçňęïijŇä¡ăéIJĂèę åŘŇæăůä¡ăè£ŸéIJĂèęĄèĞłåůśæÿĚçŘĘè£ŹäžŻæŰĞäžűãĂĆ

éĂŽåÿÿæÌěèőšïijŇäÿť æŮűæŰĞäžűåIJÍçşżçż§éżŸèőď˘çŽĎä¡■ç¡őèćńåĹŻåžžïijŇæŕŤåęĆ /var/tmp æĹŰçśżäijijçŽĎåIJřæŰźãĂĆ äÿžäžĘèŐůåŔŰçIJ§åőđçŽĎä¡■ç¡őïijŇåŔŕäžěä¡£çŤÍ tempfile.gettempdir() åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> tempfile.gettempdir()
'/var/folders/7W/7WZl5sfZEF0pljrEB1UMWE+++TI/-Tmp-'
>>>
```

æL'ĂæIJL'åŠŇäÿť æŮűæŰĞäžűçŻÿåĚşçŽĎåĞ¡æŢřéČ¡åĚĄèőÿä¡ăéĂŽè£Ğä¡£çŤÍåĚşéŤőå■ŮåŔĆæŢř prefix ãĂĄsuffix åŠŇ dir æÌěèĞłåőŽäźL'çŽőå¡ŢäžěåŔŁåŚ¡åŘ■ègĎåĹŹãĂĆæŕŤåęĆïijŽ

```
>>> f = NamedTemporaryFile(prefix='mytemp', suffix='.txt', dir='/tmp
↪')
>>> f.name
'/tmp/mytemp8ee899.txt'
>>>
```

æIJĂåŘŐè£ŸæIJL'äÿĂçĆźïijŇår¡åŔŕèČ¡äžěæIJĂåőL'åĚÍçŽĎæŰźåijŔä¡£çŤÍ tempfile æÍąåİŮæÌěåĹŻåžžäÿť æŮűæŰĞäžűãĂĆ åŇĚæŃňäžĚçżŹå¡ŞåL'■çŤÍæĹůæŐĹæİČèő£éŮőäžěåŔŁåIJÍæŰĞäžű èęĄæşÍæĎŔçŽĎæŸŕäÿ■åŘŇçŽĎåźşåŘŕåŔŕèČ¡äijŽäÿ■äÿĂæăůãĂĆåŻăæ■ď˘ä¡ăæIJĂåě¡éŸĚèŕż åőŸæŰźæŰĞæąč æÌěäžĘèğčæŻť åď˘ŽçŽĎçżĘèŁĆãĂĆ

5.20 äÿŐäÿšèąŇçńŕåŔčçŽĎæŢřæ■őéĂŽä£ą

éŮőécŸ

ä¡ăæČşéĂŽè£ĞäÿšèąŇçńŕåŔčèŕżåĘŹæŢřæ■őïijŇåĚÿåđŃåIJžæŹŕåřśæŸŕåŠŇäÿĂäžŽçąňäžűèő¿åđ'ĞæĽ'Ş

èğčåĘşæŰźæąĹ

åř¡çőąä¡ăåŔŕäžěéĂŽè£ĞäĄ£çŤÍPythonnåĘĚç¡őçŽĎI/OæÍąåİŮæİěåőŇæĹŘè£ŹäÿłäżżåŁąïijŇä¡Ęåŕźäž̃Őäÿ
pySerialåŇĚ ãĂĆ è£ŹäÿłåŇĚçŽĎä¡£çŤÍéİđåÿÿçőĂå■ŢïijŇåĚĹáőĽ'èčĚpySerialïijŇä¡£çŤÍçśżäijijäÿŃéİćè£Źæ

```
import serial
ser = serial.Serial('/dev/tty.usbmodem641', # Device name varies
                    baudrate=9600,
                    bytesize=8,
                    parity='N',
                    stopbits=1)
```

èő¿åđ'ĞåŘ■åŕźäžŐäÿ■åŘŇçŽĎèő¿åđ'ĞåŠŇæŞ■ä¡IJçşżçż§æŸŕäÿ■äÿĂæăůçŽĎãĂĆ
æŕŤåęĆïijŇåIJÍWindowsçşżçż§äÿŁïijŇä¡ăåŔŕäżěä¡£çŤÍ0, 1ç■L'èąÍçđ'žçŽĎäÿĂäÿłèő¿åđ'ĞæİěæL'ŞåijĂéĂŽä‡
äÿĂæŮęçńŕåŔčæL'ŞåijĂïijŇéĆčåŕšåŔŕäżěä¡£çŤÍ `read()`ïijŇ`readline()` åŠŇ `write()`
åĞ¡æŢřèŕżåĘŹæŢřæ■őäžĘãĂĆä¿ŃåęĆïijŽ

```
ser.write(b'G1 X50 Y50\r\n')
resp = ser.readline()
```

åđ'gåđ'ŽæŢřæČĚåĘţäÿŃïijŇçőĂå■ŢçŽĎäÿšåŔčéĂŽä£ąäżŐæ■đ'åŔŸå¿Ůå■ĄåĹĘçőĂå■ŢãĂĆ

èőÍèőž

åř¡çőąèąÍéİćäÿŁçIJŃèţůæİěå¿ĹçőĂå■ŢïijŇåĚűåőđäÿšåŔčéĂŽä£ąæIJL'æŮűåĂŹäź§æŸŕæŇžéžżçČęçŽĎ
æŐÍè■Řä¡ää¡£çŤÍçňňäÿL'æŰźåŇĚåęĆ `pySerial` çŽĎäÿĂäÿłåŐ§åŽăæŸŕåőČæŔŘä¿ŻäžĘåŕźéńŸçžğçL'źæĂ
(æŕŤåęĆèűĚæŮűïijŇæŐğåĹűæţĄïijŇçijŞåĘšåŇžåĹůæŰřïijŇæŔąæL'Ńå■Řèőőç■L'ç■L')ãĂĆäÿ¿äÿłä¿Ńå■Řïij
`RTS-CTS` æŔąæL'Ńå■Řèőőïij Ň ä¡ăåŔłéIJĂèęĄçżŹ `Serial()` äijăéĂŠäÿĂäÿł
`rtscts=True` çŽĎåŔĆæŢřå■şåŔŕãĂĆ åĚűåőŸæŰźæŰĞæąčéİđåÿÿåőŇåŰĎïijŇåŽăæ■đ'æĹŠåIJÍè£ŹéĞŇa

æŮűåĹżèőŕä¡ŔæL'ĂæIJL'æűL'åŔŁåĹřäÿšåŔčçŽĎI/OéČ¡æŸŕäžŇè£ŻåĹűæĺąåijŔçŽĎãĂĆåŽăæ■đ'ïijŇçą
(æĹŰæIJL'æŮűåĂŹæL'ğèąŇæŰĞæIJńçŽĎçijŰçăĄ/èğčçăĄæŞ■ä¡IJ)ãĂĆ
åŔęåđ'Ůå¡Şä¡ăéIJĂèęĄåĹŻåžžäžŇè£ŻåĹűçijŰçăĄçŽĎæŇĞäžđ'æĹŰæŢřæ■őåŇĚçŽĎæŮűåĂŹïijŇ`struct`
æĺąåİŮäź§æŸŕéİđåÿÿæIJL'çŤÍçŽĎãĂĆ

5.21 åžŕåĹŮåŇŰPythonåŕźèšą

éŮőécŸ

ä¡ăéIJĂèęĄåřĘäÿĂäÿłPythonåŕźèšąåžŕåĹŮåŇŰäÿžäÿĂäÿłå■ŮèŁĆæţĄïijŇäžěä¿£åřĘåőČä£Íå■ŸåĹřäÿĂ

èğčåĘşæŰźæąĹ

åŕźäžŐåžŕåĹŮåŇŰæIJĂæŹőéĄ■çŽĎåĄŽæşŢåŕśæŸŕä¡£çŤÍ `pickle` æĺąåİŮãĂĆäÿžäžĘåřĘäÿĂäÿłåŕźèšąä£Íå■ŸåĹřäÿĂäÿłæŰĞäžűäÿ■ïijŇåŕŕäžěè£ŹæăůåĄŽïijŽ

```
import pickle

data = ... # Some Python object
f = open('somefile', 'wb')
pickle.dump(data, f)
```

äÿžäžĘåřĘäÿĂäÿłåŕźèšąè¡ňåĆÍäÿžäÿĂäÿłå■ŮçňęäÿšïijŇåŕŕäžěä¡£çŤÍ `pickle.dumps()` ïijŽ

```
s = pickle.dumps(data)
```

äÿžäžĘäżŐå■ŮèŁĆæţĄäÿ■æĄćåď■äÿĂäÿłåŕźèšąïijŇä¡£çŤÍ `picle.load()` æĹŰ `pickle.loads()` åĞ¡æŢřãĂĆærŤåęĆïijŽ

```
# Restore from a file
f = open('somefile', 'rb')
data = pickle.load(f)

# Restore from a string
data = pickle.loads(s)
```

èőÍèőž

åŕźäžŐåď˘ğåď˘ŽæŢřåžŤçŤÍçÍŃåžŕæİèèőšïijŇdump() åŠŇ load() åĞ¡æŢřçŽĎä¡£çŤÍåřśæŸŕä¡äæIJL˘æŢĹä¡£çŤÍ `pickle` æĺąåİŮæL˘ĂéIJĂçŽĎåĚÍéČÍäžĘãĂĆ åőČåŕŕéĂĆçŤÍäžŐçżÍåď˘ğéČÍåĹĘPythonæŢřæ■őçśžåđŃåŠŇçŤÍæĹůèĞłåőŽäźL˘çśžçŽĎåŕźèšąåőđä¿ŃãĂĆ åęĆæđIJä¡ăçćřåĹřæ§ŘäÿłåžŞåŕŕäžěèőľä¡ååIJÍæŢřæ■őåžŞäÿ■ä£Íå■Ÿ/æĄćåď■PythonåŕźèšąæĹŰèĂĚæŸŕéĂŽ éĆčäźĹå¿ĹæIJL˘åŕŕèČ¡è£ŹäÿłåžŞçŽĎåžŢåśĆåŕśä¡£çŤÍäžĘ `pickle` æĺąåİŮãĂĆ

`pickle` æŸŕäÿĂçğ■PythonçL˘źæIJL˘çŽĎèĞłæŔŕŕè£ŕçŽĎæŢřæ■őçijŰçăĄãĂĆ éĂŽè£ĞèĞłæŔŕŕè£ŕïijŇèćńåžŕåĹŮåŇŰåŘŐçŽĎæŢřæ■őåŇĚåŘńæŕŔäÿłåŕźèšąåijĂåğŃåŠŇçżŞæİ§äžěåŘŁåő åŽăæ■d˘ïijŇä¡ăæŮăéIJĂæŃĚå£Čåŕźèšąèőŕå¡ŢçŽĎåőŽäźL˘ïijŇåőČæĂżæŸŕèČ¡åůěä¡IJãĂĆ äÿ¿äÿłä¿Ńå■ŘïijŇåęĆæđIJèęĄåď˘ĎçŘEåď˘ŽäÿłåŕźèšąïijŇä¡ååŕŕäžěè£ŹæăůåĄŽïijŽ

```
>>> import pickle
>>> f = open('somedata', 'wb')
>>> pickle.dump([1, 2, 3, 4], f)
>>> pickle.dump('hello', f)
>>> pickle.dump({'Apple', 'Pear', 'Banana'}, f)
>>> f.close()
>>> f = open('somedata', 'rb')
>>> pickle.load(f)
[1, 2, 3, 4]
>>> pickle.load(f)
'hello'
>>> pickle.load(f)
{'Apple', 'Pear', 'Banana'}
>>>
```

ä¡ăè£ŸèČ¡åžŕåĹŮåŇŰåĞ¡æŢřïijŇçśżïijŇè£ŸæIJL'æŐěåŕčïijŇä¡ĘæŸŕçżŞæđIJæŢřæ■őäżĚäżĚářĘåőČä

```
>>> import math
>>> import pickle.
>>> pickle.dumps(math.cos)
b'\x80\x03cmath\ncos\nq\x00.'
>>>
```

å¡ŞæŢřæ■őåŕ■åžŕåĹŮåŇŰåŻđæİěçŽĎæŮűåĂŹïijŇäijŽåĚĹåĄĞåőŽæL'ĂæIJL'çŽĎæžŘæŢřæ■őæŮűåŕ
æĺąåİŮãĂĄçśżåŠŇåĞ¡æŢřäijŽèĞłåŁĺæŇL'éIJĂåŕijåĚèè£ŻæİěãĂĆåŕźäžŐPythonæŢřæ■ôècńäÿ■åŘŇæIJžåŹĺ
æŢřæ■őçŽĎä£Ía■ŸåŕŕèČ¡äijŽæIJL'éŮőécŸïijŇåŽăäÿžæL'ĂæIJL'çŽĎæIJžåŹĺéČ¡å£Ëéążèő£éŮőåŘŇäÿĂäÿł

æşÍ

```
å■ČäÿĞäÿ■èęĄåŕźäÿ■ä£ąäżżçŽĎæŢřæ■őä¡£çŤÍpickle.load()ãĂĆ
pickleåIJÍåŁăè¡¡æŮűæIJL'äÿĂäÿłåL'ŕä¡IJçŤÍåřśæŸŕåőČäijžèĞłåŁÍåŁăè¡¡çŻÿåžŤæÍąåİŮåź
ä¡ĘæŸŕæ§ŘäÿłåİŔäžžåęĆæđIJç§ěéĄ§pickleçŽĎåůëä¡IJåŐ§çŘĘïijŇ
äżŰåřśåŔŕäżěåĹżåżžäÿĂäÿłæĄűæĎŔçŽĎæŢřæ■őåŕijèĞt'PythonæL'ğèąŇéŽŔæĎŔæŇĞåőŽçŽĎçşżçż
åŽăæ■d'ïijŇäÿĂåőŽèęĄä£ÌèŕĄpickleåŔłåIJÍçŻÿäžŠäźŃéŮt'åŔŕäżěèőd'èŕĄåŕźæŰźçŽĎèğčæđŘ
```

æIJL'äžŻçśżåđŃçŽĎåŕźèśąæŸŕäÿ■èČ¡èćńåžŔåĹŮåŇŰçŽĎãĂĆè£ŹäžŻéĂŽåÿÿæŸŕéĆčäžŻä¿ÌèţŰåđ'Ůé
æŕŤåęĆæL'ŞåijĂçŽĎæŰĞäżűïijŇç¡ŚçżIJè£đæŐěïijŇçž£çÍŃïijŇè£ŻçÍŃïijŇæăĹåÿğç■L'ç■L'ãĂĆ
çŤÍæĹůèĞłåőŽäźL'çśżåŔŕäżěéĂŽè£ĞæŔŘä¿Ż `__getstate__()`
åŠŇ `__setstate__()` æŰźæşŢæİěçżŢè£Ğè£ŹäžŻéŹŔåĹűãĂĆ
åęĆæđIJåőŽäźL'äžĘè£Źäÿd'äÿłæŰźæşŢïijŇ`pickle.dump()`
åŕśäijŽèŕČçŤÍ `__getstate__()` èŐůåŔŰåžŔåĹŮåŇŰçŽĎåŕźèśąãĂĆ
çśżäijijçŽĎïijŇ`__setstate__()` åIJÍåŔ■åžŔåĹŮåŇŰæŮűèćńèŕČçŤÍãĂĆäÿžäžĘæijŢçd'žè£Źäÿłåůëä¡IJåĆ
äÿŃéÍćæŸŕäÿĂäÿłåIJÍåĘĚéČÍåőŽäźL'äžĘäÿĂäÿłçž£çÍŃä¡Ęäż■çĎűåŔŕäżěåžŔåĹŮåŇŰåŠŇåŔ■åžŔåĹŮåŇŰç

```
# countdown.py
import time
import threading

class Countdown:
    def __init__(self, n):
        self.n = n
```

```
        self.thr = threading.Thread(target=self.run)
        self.thr.daemon = True
        self.thr.start()

    def run(self):
        while self.n > 0:
            print('T-minus', self.n)
            self.n -= 1
            time.sleep(5)

    def __getstate__(self):
        return self.n

    def __setstate__(self, n):
        self.__init__(n)
```

èŕŢçİĂè£Řèą Ňäÿ Ńéİćç ŽĎåžŔåĹŮåŇŰèŕŢéłŇäžčçăĄïijŽ

```
>>> import countdown
>>> c = countdown.Countdown(30)
>>> T-minus 30
T-minus 29
T-minus 28
...

>>> # After a few moments
>>> f = open('cstate.p', 'wb')
>>> import pickle
>>> pickle.dump(c, f)
>>> f.close()
```

çĎűåŘŐéĂĂåĞžPythonèğčæđŘåŹĺåźűéĞ■åŘŕåŘŐåĘ■èŕŢéłŇäÿŃïijŽ

```
>>> f = open('cstate.p', 'rb')
>>> pickle.load(f)
countdown.Countdown object at 0x10069e2d0>
T-minus 19
T-minus 18
...
```

ä¡ăåŔŕäžěçIJŃåĹřçž£çĺŃåŔĹåěĞè£źèĹňçŽĎéĞ■çŤ§äžĘïijŇäžŐä¡ăçňňäÿĂæňąåžŔåĹŮåŇŰåőČçŽĎåIJ

pickle åŕźäžŐåď ğåđŃçŽĎæŢřæ■őçżŞæđĎæŕŤåęĆä¡£çŤĺ array æĹŰ numpy æĺąåİŮåĹŻåžžçŽĎäžŇè£ŻåĹűæŢřçżĎæŢĹçŐĞåźűäÿ■æŸŕäÿĂäÿłéńŸæŢĹçŽĎçijŰçăĄæŰźåijŔãĂĆ åęĆæđIJä¡ăéIJĂèęĄçğżåŁĺåď ğéĞŔçŽĎæŢřçżĎæŢřæ■őïijŇä¡ăæIJĂåě¡æŸŕåĚĹåIJĺäÿĂäÿłæŰĞäžűäÿ■årĘåĚ (éIJĂèęĄçňňäÿĽæŰźåžŞçŽĎæŤŕæŇĄ)ãĂĆ

çŤśäžŐ pickle æŸŕPythonçĽźæIJĽçŽĎåźűäÿŤéŹĎçİĂåIJĺæžŘçăĄäÿŁïijŇæĽĂæIJĽåęĆæđIJéIJĂèę ä¿ŃåęĆïijŇåęĆæđIJæžŘçăĄåŔŸåŁĺäžĘïijŇä¡ăæĽĂæIJĽçŽĎå■ŸåĆĺæŢřæ■őåŔŕèČ¡äijŽèćńçăť åİŔåźűäÿŤåŔ åİęçŹ¡æİěèőšïijŇårźäžŐåIJĺæŢřæ■őåžŞåŠŇå■ŸæąčæŰĞäžűäÿ■å■ŸåĆĺæŢřæ■őæŮűïijŇä¡ăæIJĂåě¡ä¡£çŤĺæŽ è£ŹäžŽçijŰçăĄæäijåijŔæŽť æăĞåĞĘïijŇåŔŕäžěèćńäÿ■åŘŇçŽĎèŕ■èĺĂæŤŕæŇĄïijŇåźűäÿŤäź§èČ¡å¿Ĺåě¡çŽĎ

æIJĂåŘŐäÿĂçĆźèęĄæşÍæĎŔçŽĎæŸí `pickle` æIJL'åď ğéĞŔçŽĎéĚ■ç¡őéĂL'éąźåŠŇäÿĂäžŻæčŸæL'Ń
åŕźäžŐæIJĂåÿÿèğĄçŽĎä¡£çŤĺåIJžæŹríijŇä¡ääÿ■éIJĂèęĄåŐżæŃĚå£Čè£ŹäÿłiijŇä¡ĘæŸŕåęĆæđIJä¡äèęĄåIJĺ
æIJĂåě¡åŐżæ§ěéŸĚäÿĂäÿŃ åőŸæŰźæŰĞæąč ãĂĆ

CHAPTER 8

çňňåĚ■çńäïijŽæŢřæ■őçijŰçăĄåŠŇåďĎçŘĘ

è£ŹäÿĂçńääÿżèęĄèőÍèőžä¡£çŤÍPythonåď˙ĎçŘĘåŘĎçğ■äÿ■åŘŇæŰźåijŔçijŰçăĄçŽĎæŢřæ■őïijŇæŕŤåę
åŠŇæŢřæ■őçżŞæđĎéĆčäÿĂçńääÿ■åŘŇçŽĎæŸŕïijŇè£Źçńääÿ■äijŽèőÍèőžçL’źæőŁçŽĎçőŮæşŢéŮőécŸïijŇè

Contents:

6.1 èŕżåĘŹCSVæŢřæ■ő

éŮőécŸ

ä¡ăæČşèŕżåĘŹäÿĂääyłCSVæäijåijŔçŽĎæŰĞäžűãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐåď˙ğåď˙ŽæŢřçŽĎCSVæäijåijŔçŽĎæŢřæ■őèŕżåĘŹéŮőécŸïijŇéČ¡åŔŕäžëä¡£çŤÍ
`csv` åžŞãĂĆ ä¿ŃåęĆïijŽåĄĞèő¿ä¡ăåIJÍäÿĂääyłåŘ■åŔństocks.csvæŰĞäžűäÿ■æIJL’äÿĂäžŽèĆąçĕÍåÿĆåIJžæŢ

```
Symbol,Price,Date,Time,Change,Volume
"AA",39.48,"6/11/2007","9:36am",-0.18,181800
"AIG",71.38,"6/11/2007","9:36am",-0.15,195500
"AXP",62.58,"6/11/2007","9:36am",-0.46,935000
"BA",98.31,"6/11/2007","9:36am",+0.12,104800
"C",53.08,"6/11/2007","9:36am",-0.25,360900
"CAT",78.29,"6/11/2007","9:36am",-0.23,225400
```

äÿŃéİćåŘŚä¡ăåśŢçď˙žåęĆä¡ŢåřĘè£ŹäžŻæŢřæ■őèŕżåŔŰäÿžäÿĂäÿłåĚČçżĎçŽĎåžŔåĹŮïijŽ

```
import csv
with open('stocks.csv') as f:
    f_csv = csv.reader(f)
    headers = next(f_csv)
    for row in f_csv:
        # Process row
        ...
```

åIJÍäÿŁéİćçŽĎäžčçăĄäÿ■ïijŇ `row` äijŽæŸŕäÿĂäÿłåĹŮèąĺãĂĆåŽăæ■d'ïijŇäÿžäžĘèő£éŮőæ§Řäÿłå■Ůæő `row[0]` èő£éŮőSymbolïijŇ `row[4]` èő£éŮőChangeãĂĆ

çŤśäžŐè£ŹçğäÿŃæăĞèő£éŮőéĂŽåÿÿäijŽåijŢètţůæůůæůĘïijŇä¡ăåŔŕäžěèĂČèŹŚä¡£çŤĺåŚ¡åŘ■åĚČçżĎ

```
from collections import namedtuple
with open('stock.csv') as f:
    f_csv = csv.reader(f)
    headings = next(f_csv)
    Row = namedtuple('Row', headings)
    for r in f_csv:
        row = Row(*r)
        # Process row
        ...
```

åőČåĚĄèőÿä¡ää¡£çŤĺåĹŮåŘ■åęĆ `row.Symbol` åŠŇ `row.Change` äžčæŽ£äÿŃæăĞèő£éŮőãĂĆ éIJĂèęĄæşĺæĎŔçŽĎæŸŕè£ŹäÿłåŔłæIJL'åIJÍåĹŮåŘ■æŸŕåŘĹæşŢçŽĎPythonæă ä¡ăåŔŕèČ¡éIJĂèęĄä£őæŤźäÿŃåŐ§ăgŃçŽĎåĹŮåŘ■(åęĆåřĘéİđæăĞèŕĘçňęå■ŮçňęæŽ£æ■ćæĹŘäÿŃåĹŠçž£äź

åŔęåď'ŰäÿĂäÿłéĂL'æŃľåřśæŸŕåřĘæŢřæ■őèŕžåŔŰåĹřäÿĂäÿłå■ŮåĚÿåžŔåĹŮäÿ■åŐżãĂĆåŔŕäžěè£ŹæĂ

```
import csv
with open('stocks.csv') as f:
    f_csv = csv.DictReader(f)
    for row in f_csv:
        # process row
        ...
```

åIJÍè£ŹäÿłçĽĹæIJñäÿ■ïijŇä¡ăåŔŕäžěä¡£çŤĺåĹŮåŘ■åŐżèő£éŮőæŕŔäÿĂèąŃçŽĎæŢřæ■őäžĘãĂĆæŕŤåęĆ æĹŰèĂĚ `row['Change']`

äÿžäžĘåĘŹåĚěCSVæŢřæ■őïijŇä¡ääž■çĎűåŔŕäžěä¡£çŤĺcsvæĺąåİŮïijŇäÿ■è£ĞèŹæŮűåĂŹåĚĹåĹŻåžžäÿ `writer` åŕźèśąãĂĆä¿ŃåęĆ:

```
headers = ['Symbol','Price','Date','Time','Change','Volume']
rows = [('AA', 39.48, '6/11/2007', '9:36am', -0.18, 181800),
        ('AIG', 71.38, '6/11/2007', '9:36am', -0.15, 195500),
        ('AXP', 62.58, '6/11/2007', '9:36am', -0.46, 935000),
       ]

with open('stocks.csv','w') as f:
    f_csv = csv.writer(f)
    f_csv.writerow(headers)
    f_csv.writerows(rows)
```

åęĆæđIJä¡ăæIJL'äÿĂäÿłå■ŮåĚÿåžŔåĹŮçŽĎæŢřæ■őïijŇåŔŕäžěåČŔè£ŹæăůåĄŽïijŽ

```
headers = ['Symbol', 'Price', 'Date', 'Time', 'Change', 'Volume']
rows = [{'Symbol':'AA', 'Price':39.48, 'Date':'6/11/2007',
         'Time':'9:36am', 'Change':-0.18, 'Volume':181800},
         {'Symbol':'AIG', 'Price': 71.38, 'Date':'6/11/2007',
         'Time':'9:36am', 'Change':-0.15, 'Volume': 195500},
         {'Symbol':'AXP', 'Price': 62.58, 'Date':'6/11/2007',
         'Time':'9:36am', 'Change':-0.46, 'Volume': 935000},
         ]

with open('stocks.csv','w') as f:
    f_csv = csv.DictWriter(f, headers)
    f_csv.writeheader()
    f_csv.writerows(rows)
```

èőÍèőž

ä¡ăåžŤèŕěæĂżæŸŕäijŸåĚĹéĂL'æŃľcsvæĺąåİŮåĹĘåL'šæĹŰèğčæđŘCSVæŢřæ■őãĂĆä¿ŃåęĆïijŇä¡ăåŔŕ

```
with open('stocks.csv') as f:
for line in f:
    row = line.split(',')
    # process row
    ...
```

ä¡£çŤĺè£Źçğ■æŰźåijŔçŽĎäÿĂäÿłçijžçĆźåřśæŸŕä¡ăäż■çĎűéIJĂèęĄåŐżåď ĎçŘĘäÿĂäžŻæčŸæL'ŃçŽĎç
æŕŤåęĆïijŇåęĆæđIJæ§ŘäžŻå■Ůæőţå Äijèćńåij ŢåŔůåŇĚåŻť ïijŇä¡ăäÿ■å¿ŮäÿåŐżéŹď è£ŹäžŻåijŢåŔůãĂĆ
åŔęåď ŰïijŇåęĆæđIJäÿĂäÿłèćńåij ŢåŔůåŇĚåŻť çŽĎå■ŮæőţçćřåůğåŕńæIJL'äÿĂäÿłéĂŮåŔůïijŇéĆčäźĹçÍŃåž

éżŸèőď æČĚåĘţäÿŃïijŇcsv åžŞåŔŕèŕĘåĹńMicrosoft Ex-
celæL'Ăä¡£çŤĺçŽĎCSVçijŰçăĄèğĎåĹŹãĂĆ è£ŹæĹŰèőÿäź§æŸŕæIJĂåÿÿèğĄçŽĎå¡ćåijŔïijŇåźűäÿŤäź§äijŽ
çĎűèĂŇïijŇåęĆæđIJä¡ăæ§ěçIJŃcsvçŽĎæŰĞæąčïijŇåŕśäijŽåŔŚçŐŕæIJL'å¿Ĺåď Žçğ■æŰźæŞŢåŕĘåőĆåžŤçŤĺ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăæČşèŕžåŔŰäžětabåĹĘåL'šçŽĎæŢřæ■őïijŇåŔŕäžěè£ŹæăůåĄŽïijŽ

```
# Example of reading tab-separated values
with open('stock.tsv') as f:
    f_tsv = csv.reader(f, delimiter='\t')
    for row in f_tsv:
        # Process row
        ...
```

åęĆæđIJä¡ăæ■čåIJĺèŕżåŔŰCSVæŢřæ■őåźűåřĘåőČäžňè¡ňæ■ćäÿžåŚ¡åŘ■åĚČçżĎïijŇéIJĂèęĄæşĺæĎŔåŕź
ä¿ŃåęĆïijŇäÿĂäÿłCSVæäijåijŔæŰĞäžűæIJL'äÿĂäÿłåŇĚåŘńéİđæşŢæĂĞèŕĘçňęçŽĎåĹŮåď t'èąŇïijŇçśžäijijä

```
StreetÂăAddress,Num-Premises,Latitude,Longitude 5412ÂăNÂăCLARK,10,
↪41.980262,-87.668452
```

è£ŹæåůæIJĂçżĹäijŽåŕijèĞťåIJĺåĹŻåžžäÿĂäÿłåŚ¡åŘ■åĚČçżĎæŮűäžğçŢ§äÿĂäÿł ValueError åijĆåÿÿèĂŇåďśèťěãĂĆ äÿžäžĘèğčåĘşè£ŹéŮőécŸïijŇä¡ăåŔŕèČ¡äÿ■å¿Ůäÿ■åĚĹåŐżä£őæ■čå ä¿ŇåęĆïijŇåŔŕäžěåČŔäÿŃéİćè£ŹæåůåIJĺéİđæşŢæĂĞèŕĘçňęäÿŁä¡£çŢĺäÿĂäÿłæ■čåĹŹèąĺè¿¿åijŔæŽ£æ■ćïijŽ

```
import re
with open('stock.csv') as f:
    f_csv = csv.reader(f)
    headers = [ re.sub('[^a-zA-Z_]', '_', h) for h in next(f_csv) ]
    Row = namedtuple('Row', headers)
    for r in f_csv:
        row = Row(*r)
        # Process row
        ...
```

è£ŸæIJL'éĞ■èęĄçŽĎäÿĂçĆźéIJĂèęĄåijžèřČçŽĎæŸŕïijŇcsväžğçŢ§çŽĎæŢřæ■őéČ¡æŸŕå■Ůçňęäÿšçśżå åęĆædIJä¡ăéIJĂèęĄåĄŽè£ŹæåůçŽĎçśżåđŃè¡ňæ■ćïijŇä¡ăå£Ëéążè Ğłåůśæ L'ŃåŁĺåŐżåőđçŐřãĂĆ äÿŃéİćæŸŕäÿĂäÿłåIJĺCSVæŢřæ■őäÿŁæL'ğèąŇåĚűäžŰçśżåđŃè¡ňæ■ćçŽĎä¿Ńå■ŘïijŽ

```
col_types = [str, float, str, str, float, int]
with open('stocks.csv') as f:
    f_csv = csv.reader(f)
    headers = next(f_csv)
    for row in f_csv:
        # Apply conversions to the row items
        row = tuple(convert(value) for convert, value in zip(col_
↪types, row))
        ...
```

åŔęåď'ŰïijŇäÿŃéİćæŸŕäÿĂäÿłè¡ňæ■ćå■ŮåĚÿäÿ■çL'źåőŽå■ŮæőţçŽĎä¿Ńå■ŘïijŽ

```
print('Reading as dicts with type conversion')
field_types = [ ('Price', float),
                ('Change', float),
                ('Volume', int) ]

with open('stocks.csv') as f:
    for row in csv.DictReader(f):
        row.update((key, conversion(row[key]))
                   for key, conversion in field_types)
        print(row)
```

éĂŽåÿÿæİèèőšïijŇä¡ăåŔŕèČ¡åźűäÿ■æČşè£ĞåďŽåŐżèĂČèŹŚè£ŹäžŻè¡ňæ■ćéŮőécŸãĂĆ åIJĺåőđéŹĚæČĚåĘţäÿ■ïijŇCSVæŰĞäžűéČ¡æĹŰåď'ŽæĹŰåřŚæIJL'äžŽçijžåď'śçŽĎæŢřæ■őïijŇèćńçăť'åİŔçŽİ åŻăæ■d'ïijŇéŹď'éİđä¡ăçŽĎæŢřæ■őçąőåőđæIJL'ä£ĺéŽIJæŸŕåĞEçąőæŮăèŕŕçŽĎïijŇåŘęåĹŹä¡ăå£Ëéążè ĂČèŹ

æIJĂåŘŐïijŇåęĆædIJä¡ăèŕżåŔŰCSVæŢřæ■őçŽĎçŻőçŽĎæŸŕåĄŽæŢřæ■őåĹĘædŘåŠŇçż§èőąçŽĎèŕİïij ä¡ăåŔŕèČ¡éIJĂèęĄçIJŃäÿĂçIJŃ Pandas åŇĚãĂĆPandas åŇĚåŔńäžĘäÿĂäÿłéİđåÿÿæŰźä¿£çŽĎåĞ¡æŢřåŃ pandas.read_csv() ïijŇ åőČåŔŕäžěåŁăè¡¡CSVæŢřæ■őåĽräÿĂäÿł DataFrame årźèśąäÿ■åŐżãĂĆ çĎűåŘŐåĹl'çŤĺè£Źäÿłårźèśąä¡ăåŕśåŔŕäžěçŢ§æĹŘåŘĎçğ■å¡ćåijŔçŽĎçż§èőąãĂĄè£ĞæżďæŢřæ■őäžěåŘŁæL åIJĺ6.13åřŔèŁĆäÿ■äijŽæIJL'è£ŹæåůäÿĂäÿłä¿Ńå■ŘãĂĆ

6.2 èŕżåĘŹJSONæŢřæ■ő

éŮőéćŸ

ä¡ăæČşèŕżåĘŹJSON(JavaScript Object Notation)çijŰçăĄæăijåijŔçŽĎæŢřæ■őãĂĆ

èğčåĘşæŰźæąĹ

json æĺąåİŮæŔŘä¿ŻäžĘäÿĂçğ■å¿ĹçőĂå■ŢçŽĎæŰźåijŔæİěçijŰçăĄåŠŇèğčçăĄJSONæŢřæ■őãĂĆ åĚűäÿ■äÿď äÿłäÿżèęĄçŽĎåĞ¡æŢřæŸŕ json.dumps() åŠŇ json.loads() ïijŇ èęĄæŕŤåĚűäżŰåžŔåĹŮåŇŰåĞ¡æŢřåžŞåęĆpickleçŽĎæŐěåŔčåřŚå¿Ůåď ŽãĂĆ äÿŃéİćæijŤçď žåęĆä¡ŢåřĘäÿĂäÿłPythonæŢřæ■őçżŞæđĎè¡ňæ■ćäÿžJSONïijŽ

```
import json

data = {
    'name' : 'ACME',
    'shares' : 100,
    'price' : 542.23
}

json_str = json.dumps(data)
```

äÿŃéİćæijŤçď žåęĆä¡ŢåřĘäÿĂäÿłJSONçijŰçăĄçŽĎå■Ůçňęäÿšè¡ňæ■ćåŽđäÿĂäÿłPythonæŢřæ■őçżŞæđĎ

```
data = json.loads(json_str)
```

åęĆæđIJä¡ăèęĄåď ĎçŘĘçŽĎæŸŕæŰĞäżűèĂŇäÿ■æŸŕå■Ůçňęäÿšïijňä¡ăåŔŕäżěä¡£çŤĺ json.dump() åŠŇ json.load() æİěçijŰçăĄåŠŇèğčçăĄJSONæŢřæ■őãĂĆä¿ŃåęĆïijŽ

```
# Writing JSON data
with open('data.json', 'w') as f:
    json.dump(data, f)

# Reading data back
with open('data.json', 'r') as f:
    data = json.load(f)
```

èőĺèőž

JSONçijŰçăĄæŤŕæŇĄçŽĎå§žæIJňæŢřæ■őçśżåđŃäÿž None ïijŇ bool ïijŇ int ïijŇ float åŠŇ str ïijŇ äżěåŔŁåŇĚåŔńè£ŹäžŻçśżåđŃæŢřæ■őçŽĎlistsïijŇtuplesåŠŇdictionariesãĂĆ åŕźäžŐdictionariesïijŇkeyséIJĂèęĄæŸŕå■ŮçňęäÿšçśżåđŃ(å■ŮåĚÿäÿ■äżżä¡Ţéİđå■ŮçňęäÿšçśżåđŃçŽĎkeyåĹ äÿžäžĘéĄţå¿łJSONèğĎèŇČïijŇä¡ăåžŤèŕěåŔłçijŰçăĄPythonçŽĎlistsåŠŇdictionariesãĂĆ èĂŇäÿŤïijŇåIJĺwebåžŤçŤĺçĺŃåžŔäÿ■ïijŇéąűåśĆåŕźèśąèćńçijŰçăĄäÿžäÿĂäÿłå■ŮåĚÿæŸŕäÿĂäÿłæăĞåĞĘåA

JSONçijŰçăĄçŽĎæăijåijŔåŕźäžŐPythonèŕ■æşŢèĂŇåůšåĞääźŐæŸŕåőŇåĚĺäÿĂæăůçŽĎïijŇéŹd'äžĘäÿĂ
æŕŤåęĆïijŇTrueäijŽèćńæŸăåřĎäÿžtrueïijŇFalseèćńæŸăåřĎäÿž-
falseïijŇèĂŇNoneäijŽèćńæŸăåřĎäÿžnullãĂĆ äÿŃéĺćæŸŕäÿĂäÿłä¿Ńå■ŘïijŇæijŤçd'žäžĘçijŰçăĄåŘŐçŽĎå■

```
>>> json.dumps(False)
'false'
>>> d = {'a': True,
...      'b': 'Hello',
...      'c': None}
>>> json.dumps(d)
'{"b": "Hello", "c": null, "a": true}'
>>>
```

åęĆædIJä¡ăèŕŢçİĂåŐżæčĂæ§ěJSONèğčçăĄåŘŐçŽĎæŢřæ■őïijŇä¡ăéĂŽåÿÿå¿ĹéŽ¿éĂŽè£ĞçőĂå■ŢçŽİ
çĽźåĹńæŸŕå¡ŞæŢřæ■őçŽĎåţŇåěŮçżŞædĎåśĆæňąå¿ĹæůśæĹŰèĂĚåŇĚåŘńåd'ğéĞŔçŽĎå■Ůæőţæ̊ŮűãĂĆ
äÿžäžĘèğčåĘşè£ŹäÿłéŮőécŸïijŇåŕäžěèĂČèŹŚä¡£çŤĺpprintæĺąåİŮçŽĎ
pprint() åĞ¡æŢřæİěäžčæŻ£æŹőéĂŽçŽĎ print() åĞ¡æŢřãĂĆ
åőĆäijŽæŇL'çĚğkeyçŽĎå■Ůæí■éąžåžŔåźűäžěäÿĂçğ■æŻťåŁăç¿ŐèğĆçŽĎæŰźåijŔè¿ŞåĞžãĂĆ
äÿŃéĺćæŸŕäÿĂäÿłæijŤçd'žåęĆä¡ŢæijĆäžőçŽĎæĽŞå■řè¿ŞåĞžTwitteräÿŁæŘIJçt'ćçžŞædIJçŽĎä¿Ńå■ŘïijŽ

```
>>> from urllib.request import urlopen
>>> import json
>>> u = urlopen('http://search.twitter.com/search.json?q=python&
↪rpp=5')
>>> resp = json.loads(u.read().decode('utf-8'))
>>> from pprint import pprint
>>> pprint(resp)
{'completed_in': 0.074,
'max_id': 264043230692245504,
'max_id_str': '264043230692245504',
'next_page': '?page=2&max_id=264043230692245504&q=python&rpp=5',
'page': 1,
'query': 'python',
'refresh_url': '?since_id=264043230692245504&q=python',
'results': [{'created_at': 'Thu, 01 Nov 2012 16:36:26 +0000',
             'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:14 +0000',
             'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:13 +0000',
             'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:07 +0000',
             'from_user': ...
             }
             {'created_at': 'Thu, 01 Nov 2012 16:36:04 +0000',
             'from_user': ...
             }],
'results_per_page': 5,
'since_id': 0,
```

```
'since_id_str': '0'}
>>>
```

äÿĂèĹňæİěèőšïijŇJSONèğčçăĄäijŽæăźæ■őæŔŘä¿ŻçŽĎæŢřæ■őåĹŻåžždictsæĹŰlistsãĂĆ åęĆæđIJä¡ăæČşèęĄåĹŻåžžåĚűäžŰçśżåđŃçŽĎåŕźèšąïijŇåŔŕäžěçżŹ `json.loads()` äijäéĂŠobject_pairs_hookæĹŰobject_hookåŔĆæŢřãĂĆ ä¿ŃåęĆïijŇäÿŃéİćæŸŕæijŢçd'žåęĆä¡Ţèğčçă ĄJSONæŢřæ■őåźűåIJĺäÿĂäÿłOrderedDictäÿ■ä£ĮçŢŹåĚűéąžåžŔ

```
>>> s = '{"name": "ACME", "shares": 50, "price": 490.1}'
>>> from collections import OrderedDict
>>> data = json.loads(s, object_pairs_hook=OrderedDict)
>>> data
OrderedDict([('name', 'ACME'), ('shares', 50), ('price', 490.1)])
>>>
```

äÿŃéİćæŸŕåęĆä¡ŢåřĘäÿĂäÿłJSONå■ŮåĚÿè¡ňæ■ćäÿžäÿĂäÿłPythonåŕźèšąä¿Ńå■ŘïijŽ

```
>>> class JSONObject:
...     def __init__(self, d):
...         self.__dict__ = d
...
>>>
>>> data = json.loads(s, object_hook=JSONObject)
>>> data.name
'ACME'
>>> data.shares
50
>>> data.price
490.1
>>>
```

æIJĂåŘŐäÿĂäÿłä¿Ńå■Řäÿ■ïijŇJSONèğčçăĄåŘŐçŽĎå■ŮåĚÿä¡IJäÿžäÿĂäÿłå■ŢäÿłåŔĆæŢřäijăéĂŠçżŹ `__init__()` ãĂĆ çĎűåŘŐïijŇä¡ăåřšåŔŕäžěéŽŔå£ČæĽĂæňšçŽĎä¡£çŤĺåőČäžĘïijŇæŕŤåęĆä¡IJäÿžäÿĂäÿłå

åIJÍçijŰçăĄJSONçŽĎæŮűåĂŹïijŇè£ŸæIJĽäÿĂäžŽéĂĽéąźå¿ĹæIJĽçŤĺãĂĆ åęĆæđIJä¡ăæČşèŐůå¿ŮæijĆäžőçŽĎæăijåijŔåŇŰå■Ůçňęäÿšå ŘŐè¿ŞåĞžïijŇåŔŕäžěä¡£çŤĺ `json.dumps()` çŽĎindentåŔĆæŢřãĂĆ åőČäijŽä¡£å¿Ůè¿ŞåĞžåŠŇpprint()åĞ¡æŢřæŢĹæđIJçśžäijijãĂĆæŕŤ

```
>>> print(json.dumps(data))
{"price": 542.23, "name": "ACME", "shares": 100}
>>> print(json.dumps(data, indent=4))
{
    "price": 542.23,
    "name": "ACME",
    "shares": 100
}
>>>
```

åŕźèšąåőđä¿ŃéĂŽåÿÿåźűäÿ■æŸŕJSONåŔŕåžŔåĹŮåŇŰçŽĎãĂĆä¿ŃåęĆïijŽ

```
>>> class Point:
...     def __init__(self, x, y):
...         self.x = x
...         self.y = y
...
>>> p = Point(2, 3)
>>> json.dumps(p)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "/usr/local/lib/python3.3/json/__init__.py", line 226, in␣
↪dumps
        return _default_encoder.encode(obj)
    File "/usr/local/lib/python3.3/json/encoder.py", line 187, in␣
↪encode
        chunks = self.iterencode(o, _one_shot=True)
    File "/usr/local/lib/python3.3/json/encoder.py", line 245, in␣
↪iterencode
        return _iterencode(o, 0)
    File "/usr/local/lib/python3.3/json/encoder.py", line 169, in␣
↪default
        raise TypeError(repr(o) + " is not JSON serializable")
TypeError: <__main__.Point object at 0x1006f2650> is not JSON␣
↪serializable
>>>
```

åęĆædIJä¡ăæČşåžŕåĹŮåŇŰåŕźèśąåőđä¿ŃïijŇä¡ăåŕŕäżěæŔŘä¿ŻäÿĂäÿłåĞ¡æŢřïijŇåőČçŽĎè¿ŞåĚěæŸŕ

```
def serialize_instance(obj):
    d = { '__classname__' : type(obj).__name__ }
    d.update(vars(obj))
    return d
```

åęĆædIJä¡ăæČşåŔ■è£ĞæİěèŐůåŔŰè£Źäÿłåőđä¿ŃïijŇåŔŕäżěè£ŹæăůåĄŽïijŽ

```
# Dictionary mapping names to known classes
classes = {
    'Point' : Point
}

def unserialize_object(d):
    clsname = d.pop('__classname__', None)
    if clsname:
        cls = classes[clsname]
        obj = cls.__new__(cls) # Make instance without calling __
↪init__
        for key, value in d.items():
            setattr(obj, key, value)
        return obj
    else:
        return d
```

äÿŃéÌćæŸŕåęĆä¡Ţä¡£çŤĺè£ŹäžŻåĞ¡æŢřçŽĎä¿Ńå■ŘïijŽ

```
>>> p = Point(2,3)
>>> s = json.dumps(p, default=serialize_instance)
>>> s
'{"__classname__": "Point", "y": 3, "x": 2}'
>>> a = json.loads(s, object_hook=unserialize_object)
>>> a
<__main__.Point object at 0x1017577d0>
>>> a.x
2
>>> a.y
3
>>>
```

json æĺąåİŮè£ŸæIJL'å¿Ĺåd'ŽåĚűäžŰéĂL'éąźæÌěæŐğåĹűæŻt'ä¡ŐçžğåĹńçŽĎæŢřå■ŮãĂĄçL'źæőŁåĂi åŔŕäžěåŔĆèĂČåőŸæŰźæŰĞæąčèŐůåŔŰæŻt'åd'ŽçżĘèŁĆãĂĆ

6.3 èğčædŘçőĂå■ŢçŽĎXMLæŢřæ■ő

éŮőécŸ

ä¡ăæČşäžŐäÿĂäÿłçőĂå■ŢçŽĎXMLæŰĞæąčäÿ■æŔŘåŔŰæŢřæ■őãĂĆ

èğčåĘşæŰźæąĹ

åŕŕäžěä¡£çŤĺ xml.etree.ElementTree æĺąåİŮäžŐçőĂå■ŢçŽĎXMLæŰĞæąčäÿ■æŔŘåŔŰæŢřæ■őãĂĆ äÿžäžĘæijŤçd'žïijŇåĄĞèő¿ä¡ăæČşèğčædŘPlanet PythonäÿŁçŽĎRSSæžŘãĂĆäÿŃéÌćæŸŕçŻÿåžŤçŽĎäžčçăĄïijŽ

```
from urllib.request import urlopen
from xml.etree.ElementTree import parse

# Download the RSS feed and parse it
u = urlopen('http://planet.python.org/rss20.xml')
doc = parse(u)

# Extract and output tags of interest
for item in doc.iterfind('channel/item'):
    title = item.findtext('title')
    date = item.findtext('pubDate')
    link = item.findtext('link')

    print(title)
    print(date)
    print(link)
    print()
```

è£ŘèąŇäÿŁéİćçŽĎäžčçăĄïijŇè¿ŞåĞžçżŞæđIJçśżäijijè£ŹæăůïijŽ

```
Steve Holden: Python for Data Analysis
Mon, 19 Nov 2012 02:13:51 +0000
http://holdenweb.blogspot.com/2012/11/python-for-data-analysis.html

Vasudev Ram: The Python Data model (for v2 and v3)
Sun, 18 Nov 2012 22:06:47 +0000
http://jugad2.blogspot.com/2012/11/the-python-data-model.html

Python Diary: Been playing around with Object Databases
Sun, 18 Nov 2012 20:40:29 +0000
http://www.pythondiary.com/blog/Nov.18,2012/been-...-object-
 ↪databases.html

Vasudev Ram: Wakari, Scientific Python in the cloud
Sun, 18 Nov 2012 20:19:41 +0000
http://jugad2.blogspot.com/2012/11/wakari-scientific-python-in-
 ↪cloud.html

Jesse Jiryu Davis: Toro: synchronization primitives for Tornado␣
 ↪coroutines
Sun, 18 Nov 2012 20:17:49 +0000
http://feedproxy.google.com/~r/EmptysquarePython/~3/_DOZT2Kd0hQ/
```

å¿ĹæŸ¿çĎűïijŇåęĆæđIJä¡ăæČşåĄŽè£ŻäÿĂæ■ěçŽĎåď ĎçŘĘïijŇä¡ăéIJĂèęĄæŻ£æ■ć `print()` èí■åŔěæİěåőŇæĹŘåĚűäžŰæIJL'èűčçŽĎäžŇãĂĆ

èőĺèőž

åIJĺå¿Ĺåď ŽåžŤçŤĺçĺŇåžŔäÿ■åď ĎçŘĘXMLçijŰçăĄæăijåijŔçŽĎæŢřæ■őæŸŕå¿Ĺåÿÿè§ĄçŽĎãĂĆäÿ■äžĚåŻääÿžXMLåIJĺInternetäÿŁéİćåůšçżŔèćńáź£æşŻåžŤçŤĺäžŐæŢřæ■őäžď æ■ćïijŇåŘŇæŮűåőČäź§æŸŕäÿĂçğ■å■ŸåĆĺåžŤçŤĺçĺŇåžŔæŢřæ■őçŽĎåÿÿçŤĺæăijåijŔ(æŕŤåęĆå■Ůåď ĎçŘĘïijŇé§§äæŐěäÿŇæİěçŽĎèőĺèőžäijŽåĚĹåĄĞåőŽèŕżèĂĚåůšçżŔåŕźXMLå§žçąĂæŕŤè¿ČçĘ§æĆL'äžĘãĂĆ

åIJĺå¿Ĺåď ŽæČĚåĘţäÿŇïijŇå¡Şä¡£çŤĺXMLæİěäżĚäżĚå■ŸåĆĺæŢřæ■őçŽĎæŮűåĂŹïijŇåŕźåžŤçŽĎæŰĞä¿ŇåęĆïijŇäÿŁéİćä¿Ňå■Ŕäÿ■çŽĎRSSèőćéŸĚæžŘçśżäijijäžŐäÿŇéİćçŽĎæăijåijŔïijŽ

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:dc="http://purl.org/dc/elements/1.1/">
    <channel>
        <title>Planet Python</title>
        <link>http://planet.python.org/</link>
        <language>en</language>
        <description>Planet Python - http://planet.python.org/</
 ↪description>
        <item>
            <title>Steve Holden: Python for Data Analysis</title>
            <guid>http://holdenweb.blogspot.com/...-data-analysis.
 ↪html</guid>
```

```
            <link>http://holdenweb.blogspot.com/...-data-analysis.
↪html</link>
            <description>...</description>
            <pubDate>Mon, 19 Nov 2012 02:13:51 +0000</pubDate>
        </item>
        <item>
            <title>Vasudev Ram: The Python Data model (for v2 and␣
↪v3)</title>
            <guid>http://jugad2.blogspot.com/...-data-model.html</
↪guid>
            <link>http://jugad2.blogspot.com/...-data-model.html</
↪link>
            <description>...</description>
            <pubDate>Sun, 18 Nov 2012 22:06:47 +0000</pubDate>
        </item>
        <item>
            <title>Python Diary: Been playing around with Object␣
↪Databases</title>
            <guid>http://www.pythondiary.com/...-object-databases.
↪html</guid>
            <link>http://www.pythondiary.com/...-object-databases.
↪html</link>
            <description>...</description>
            <pubDate>Sun, 18 Nov 2012 20:40:29 +0000</pubDate>
        </item>
        ...
    </channel>
</rss>
```

xml.etree.ElementTree.parse() åĞ¡æŢŕèğčædŘæŢť'äÿłXMLæŰĞæąčåźűåřĘåĚűè¡ňæ■ćæĹŘçĎűåŘŐïijŇä¡ăåřśèČ¡ä¡£çŤÍ find() ãĂĄiterfind() åŠŇ findtext() ç■L'æŰźæşŢæİěæŘIJçť'écL'źåőŽçŽĎXMLåĚČçť'ääžĘãĂĆè£ŹäžŻåĞ¡æŢŕçŽĎåŔĆæŢřåřśæŸŕæ§Řäÿłæ ŇĞåőŽçŽĎæăĞç■¿åŘ■ïijŇä¿ŃåęĆ channel/item æĹŰ title ãĂĆ

æŕŔæňąæŇĞåőŽæ§Řäÿłæă Ğç■¿æŮűïijŇä¡ăéIJĂèęĄéĄ■åŐĘæŢť'äÿłæŰĞæąčçżŞædĎãĂĆæŕŔæňąæŘIJåŘŇæăůïijŇæŕŔæňąæŞ■ä¡IJæL'ĂæŇĞåőŽçŽĎæăĞç■¿åŘ■äź§æŸŕètůåğŃåĚČçť'ăçŽĎçŻÿåŕźèůŕå¿ĎãĂĆä¿ŃåęĆïijŇæL'ğèąŇ doc.iterfind('channel/item') æİěæŘIJçť'ćæL'ĂæIJL'åIJÍ channel åĚČçť'ääÿŃéİćçŽĎ item åĚČçť'ăãĂĆ doc äžčèąÍæŰĞæąčçŽĎæIJĂéąűåśĆ(äź§åřśæŸŕçňňäÿĂçžğçŽĎ rss åĚČçť'ă)ãĂĆçĎűåŘŐæŐěäÿŃæİěçŽĎèřČçŤÍ item.findtext() äijŽäžŐåůšæL'¿åĹŕçŽĎ item åĚČçť'ää¡■ç¡őåijĂåğŃæŘIJçť'ćãĂĆ

ElementTree æÍąåİŮäÿ■çŽĎæŕŔäÿłåĚČçť'ăæIJL'äÿĂäžŻéĞ■èęĄçŽĎåśđæĂğåŠŇæŰźæşŢïijŇåIJÍèğčtag åśđæĂğåŃĚåŘńäžĘæăĞç■¿çŽĎåŘ■å■ŮïijŇtext åśđæĂğåŃĚåŘńäžĘåĘéĆÍçŽĎæŰĞæIJňïijŇèĂŇget() æŰźæşŢèČ¡èŐůåŘŰåśđæĂğåĂijãĂĆä¿ŃåęĆïijŽ

```
>>> doc
<xml.etree.ElementTree.ElementTree object at 0x101339510>
>>> e = doc.find('channel/title')
```

```
>>> e
<Element 'title' at 0x10135b310>
>>> e.tag
'title'
>>> e.text
'Planet Python'
>>> e.get('some_attribute')
>>>
```

æIJL'äÿĂçĆźèęĄåijžèřČçŽĎæŸŕ `xml.etree.ElementTree` åźűäÿ■æŸŕXMLèğčæđŘçŽĎåŤŕäÿĂæŰźæşŢãĂĆ åŕźäžŐæŻť énŸçžğçŽĎåžŤçŤĺçĺŃåžŔïijŇä¡äéIJĂèęĄèĂČèŹŚä¡£çŤĺ `lxml` ãĂĆ åőČä¡£çŤĺäžĘåŠŇElementTreeåŘŇæăůçŽĎçijŰçĺŃæŐěåŔčïijŇåŻăæ■ď'äÿŁéİćçŽĎä¿Ńå■ŘåŘŇää¡ăåŘłéIJĂèęĄåřĘåĹŽåijĂâğŃçŽĎimportèŕ■åŘěæ■ćæĹŘ `from lxml.etree import parse` åřśèąŇäžĘãĂĆ `lxml` åőŇåĚĺéĄţå¿ŁXMLæăĞåĞĘïijŇåźűäÿŤéĂ§åžęäź§éİđåÿÿå£ńïijŇåŘŇæŮűè£Ÿa

6.4 åćđéĞŔåijŔèğčæđŘåď'ğåđŃXMLæŰĞäžű

éŮőécŸ

ä¡ăæČşä¡£çŤĺår¡åŔŕèČ¡ářŚçŽĎåĘĚå■ŸäżŐäÿĂäÿłèűĚåď'ğçŽĎXMLæŰĞæąčäÿ■æŘŘåŘŰæŢřæ■őãĂĆ

èğčåĘşæŰźæąĹ

äżżä¡ŢæŮűåĂŹåŔłèęĄä¡ăéĄĞåĹřåćđéĞŔåijŔçŽĎæŢřæ■őåď'ĎçŘĘæŮűïijŇçňňäÿĂæŮűéŮť'åřśåžŤèŕěa äÿŃéİćæŸŕäÿĂäÿłå¿ĹçőĂå■ŢçŽĎåĞ¡æŢřïijŇåŔłä¡£çŤĺå¿ĹářŚçŽĎåĘĚå■ŸåřśèČ¡åćđéĞŔåijŔçŽĎåď'ĎçŘĘä

```
from xml.etree.ElementTree import iterparse

def parse_and_remove(filename, path):
    path_parts = path.split('/')
    doc = iterparse(filename, ('start', 'end'))
    # Skip the root element
    next(doc)

    tag_stack = []
    elem_stack = []
    for event, elem in doc:
        if event == 'start':
            tag_stack.append(elem.tag)
            elem_stack.append(elem)
        elif event == 'end':
            if tag_stack == path_parts:
                yield elem
                elem_stack[-2].remove(elem)
            try:
                tag_stack.pop()
```

```
                elem_stack.pop()
            except IndexError:
                pass
```

äÿžäžĘæţŃèŕŢè£ŹäÿłåĞ¡æŢřïijŇä¡äéIJĂèęĄåĚĹæIJL'äÿĂäÿłåď'ğåđŃçŽĎXMLæŰĞäžűãĂĆ éĂŽåÿÿä¡ăåŘŕäžěåIJÍæŤ£åžIJç¡ŚçńŹæĹŰåĚňåĚśæŢřæ■ős¡ŚçńŹäÿŁæL'¿åĹřè£Źæăůç˘ĎæŰĞäžűãĂĆ ä¿ŃåęĆïijŇä¡ăåŘŕäžěäÿŃè¡¡XMLæăijåijŘçŽĎèŁİåŁăåŞěå§ŐåÿĆéĄŞèůŕåİŚæt'ijæŢřæ■ősžŞãĂĆ åIJĺåĘŹè£ŹæIJňäźęçŽĎæŮűåĂŹïijŇäÿŃè¡¡æŰĞäžűåůšçżŘåŇĚåŘŕnèűĚè£Ğ100,000èąŇæŢřæ■ősijŇçijŰçăĄ

```
<response>
    <row>
        <row ...>
            <creation_date>2012-11-18T00:00:00</creation_date>
            <status>Completed</status>
            <completion_date>2012-11-18T00:00:00</completion_date>
            <service_request_number>12-01906549</service_request_
↪number>
            <type_of_service_request>Pot Hole in Street</type_of_
↪service_request>
            <current_activity>Final Outcome</current_activity>
            <most_recent_action>CDOT Street Cut ... Outcome</most_
↪recent_action>
            <street_address>4714 S TALMAN AVE</street_address>
            <zip>60632</zip>
            <x_coordinate>1159494.68618856</x_coordinate>
            <y_coordinate>1873313.83503384</y_coordinate>
            <ward>14</ward>
            <police_district>9</police_district>
            <community_area>58</community_area>
            <latitude>41.808090232127896</latitude>
            <longitude>-87.69053684711305</longitude>
            <location latitude="41.808090232127896"
            longitude="-87.69053684711305" />
        </row>
        <row ...>
            <creation_date>2012-11-18T00:00:00</creation_date>
            <status>Completed</status>
            <completion_date>2012-11-18T00:00:00</completion_date>
            <service_request_number>12-01906695</service_request_
↪number>
            <type_of_service_request>Pot Hole in Street</type_of_
↪service_request>
            <current_activity>Final Outcome</current_activity>
            <most_recent_action>CDOT Street Cut ... Outcome</most_
↪recent_action>
            <street_address>3510 W NORTH AVE</street_address>
            <zip>60647</zip>
            <x_coordinate>1152732.14127696</x_coordinate>
            <y_coordinate>1910409.38979075</y_coordinate>
            <ward>26</ward>
```

```
            <police_district>14</police_district>
            <community_area>23</community_area>
            <latitude>41.91002084292946</latitude>
            <longitude>-87.71435952353961</longitude>
            <location latitude="41.91002084292946"
            longitude="-87.71435952353961" />
        </row>
    </row>
</response>
```

åĄĞèő¿ä¡ăæČşåĘŹäýĂäýłèĎŽæIJňæİěæŇĽçĚğåİŚæť'ijæŁěåŚŁæŢřéĞŕæŐŠåĹŮéĆőçijŰåŕůçăĄãĂĆä

```
from xml.etree.ElementTree import parse
from collections import Counter

potholes_by_zip = Counter()

doc = parse('potholes.xml')
for pothole in doc.iterfind('row/row'):
    potholes_by_zip[pothole.findtext('zip')] += 1
for zipcode, num in potholes_by_zip.most_common():
    print(zipcode, num)
```

è£ŹäýłèĎŽæIJňåŤŕäýĂçŽĎéŮőécŸæŸŕåőČäijŽåĚĹåřĘæŢť'äýłXMLæŰĞäžűåŁäè¡¡åĹŕåĘĚå■Ÿäý■çĎűå
åIJĺæĹŚçŽĎæIJžåŹĺäýŁïijŇäýžäžĘè£ŘèąŇè£ŹäýłçĺŇåžŕéIJĂèęĄçŤĺåĹř450MBåůęåŔşçŽĎåĘĚå■Ÿçľ'žéŮť'ă
åęĆæđIJä¡£çŤĺåęĆäýŇäžčçăĄïijŇçĺŇåžŔåŔłéIJĂèęĄä£őæŤźäýĂçĆźçĆźïijŽ

```
from collections import Counter

potholes_by_zip = Counter()

data = parse_and_remove('potholes.xml', 'row/row')
for pothole in data:
    potholes_by_zip[pothole.findtext('zip')] += 1
for zipcode, num in potholes_by_zip.most_common():
    print(zipcode, num)
```

çżŞæđIJæŸŕïijŽè£ŹäýłçĽ'ĹæIJňçŽĎäžčçăĄè£ŘèąŇæŮűåŔłéIJĂèęĄ7MBçŽĎåĘĚå■Ÿ–
åď'ğåď'ğèŁĆçžęäžĘåĘĚå■ŸètĎæžŘãĂĆ

èőĺèőž

è£ŹäýĂèŁĆçŽĎæŁĂæIJŕäijŽä¿ÌètŰ `ElementTree` æĺąåİŮäý■çŽĎäýď'äýłæăýå£ČåŁ§èČ¡ãĂĆ
çňňäýĂïijŇ`iterparse()` æŰźæşŢåĚĄèőÿåŕźXMLæŰĞæąčè£ŻèąŇåćđéĞŕæŞ■ä¡IJãĂĆ
ä¡£çŤĺæŮűïijŇä¡äéIJĂèęĄæŔŘä¿ŻæŰĞäžűåŘ■åŠŇäýĂäýłåŇĚåŘńäýŇéÍćäýĂçğ■æĹŰåď'Žçğ■çśžåđŃçŽĎä
`start` , `end`, `start-ns` åŠŇ `end-ns` ãĂĆ çŤś `iterparse()`
åĹŻåžžçŽĎè£■äžčåŹĺäijŽäžğçŤ§å¡ćåęĆ `(event, elem)` çŽĎåĚČçżĎïijŇ åĚűäý■
`event` æŸŕäýŁè£ŕäžŇäžűåĹŮèąĺäý■çŽĎæ§ŘäýĂäýłïijŇèĂŇ `elem` æŸŕçŻýåžŤçŽĎXM-
LåĚČçť'ãăĂĆä¿ŃåęĆïijŽ

```
>>> data = iterparse('potholes.xml',('start','end'))
>>> next(data)
('start', <Element 'response' at 0x100771d60>)
>>> next(data)
('start', <Element 'row' at 0x100771e68>)
>>> next(data)
('start', <Element 'row' at 0x100771fc8>)
>>> next(data)
('start', <Element 'creation_date' at 0x100771f18>)
>>> next(data)
('end', <Element 'creation_date' at 0x100771f18>)
>>> next(data)
('start', <Element 'status' at 0x1006a7f18>)
>>> next(data)
('end', <Element 'status' at 0x1006a7f18>)
>>>
```

start äžŃäžűåIJÍæ§Řäÿ łåĚČçť ăçňňäÿĂæňąèćńåĹŻåžžåźűäÿŤè£ŸæšąæIJL'èćńæŔŠåĚèåĚűäžŰæŢřæ■èĂŇ end äžŃäžűåIJÍæ§Řäÿ łåĚČçť ăåůšçżŔåőŇæĹŘæŮűèćńåĹŻåžžãĂĆ åř¡çőąæšąæIJL'åIJÍä¿Ńå■Řäÿ■æijŤçď žïijŇ start-ns åŠŇ end-ns äžŃäžűèćńçŤÍæÌěåď ĎçŘĘXMLæŰĞæąčåŚ¡åŘ■çľ žéŮť çŽĎåčřæŸŐãĂĆ

è£ŹæIJňèŁCä¿Ńå■Řäÿ■ïijŇ start åŠŇ end äžŃäžűèćńçŤÍæÌěçőąçŘĘåĚČçť ăåŠŇæăĞç■¿æăĹãĂĆ æăĹäžčèąÍäžĘæŰĞæąčèćńèğčædŘæŮűçŽĎåśĆæňąçżŞædĎïijŇ è£ŸèćńçŤÍæÌěåĹď æŰ■æ§Řäÿ łåĚČçť ăæŸŕåŘęåŇźéĚ■äijäçżŹåĞ¡æŢř parse_and_remove() çŽĎèůŕå¿ĎãĂĆ åęĆædIJåŇźéĚ■ïijŇářšåĹľçŤÍ yield èŕ■åŘěåŘŚèřČçŤÍèĂĚè£ŤåŻđè£ŹäÿłåĚČçť ăãĂĆ

åIJÍ yield äźŃåŘŐçŽĎäÿŃéİćè£Źäÿłèŕ■åŘěæL'■æŸŕä¡£å¿ŮçÍŃåžŔå■ăçŤÍædĄářŠåĘĚå■ŸçŽĎElemen

```
elem_stack[-2].remove(elem)
```

è£Źäÿłèŕ■åŘëä¡£å¿ŮäźŃåL'■çŤś yield äžğçŢ§çŽĎåĚČçť ăäžŐåőČçŽĎçĹűèŁĆçĆźäÿ■åĹăéŹđ'æŐL'ãA åĄĞëő¿åůšçżŔæšąæIJL'åĚűåőČçŽĎåIJřæŰźåijŢçŤÍè£ŹäÿłåĚČçť ăäžĘïijŇéĆčäźLè£ŹäÿłåĚČçť ăářšèćńéŤĂæ

åŕźèŁĆçĆźçŽĎè£■äžčåijŔèğčædŘåŠŇåĹăéŹđ'çŽĎæIJĂçżĹæŢĹædIJåřśæŸŕäÿĂäÿłåIJÍæŰĞæąčäÿŁéńŸ æŰĞæąčæăŚçżŞædĎäžŐåğŃèĞłçżĹæšąèćńåőŇæŢť'çŽĎåĹŻåžžè£ĞãĂĆåř¡çőąåęĆæ■d'ïijŇè£ŸæŸŕèČ¡éĂŽ

è£ŹçğæŰźæąĹçŽĎäÿżèęĄçijžéŹůářśæŸŕåőČçŽĎè£ŘèąŇæĂğèČ¡äžĘãĂĆ æĹŚèĞłåůśæţŃèŕŢçŽĎçżŞædIJæŸŕïijŇèŕżåŘŰæŢť'äÿłæŰĞæąčåĹřåĘĚå■ŸäÿçŽĎçL'ĹæIJňçŽĎè£ŘèąŇéĂ§ ä¡ĘæŸŕåőČå■ť'ä¡£çŤÍäžĘèűĚè£ĞåŘŐèĂĚ60åĂ■çŽĎåĘĚå■ŸãĂĆ åŻăæ■d'ïijŇåęĆædIJä¡ăæŻť'åĚşå£ČåĘĚå■Ÿä¡£çŤÍéĞŘçŽĎèŕÏïijŇéĆčäźĹåćđéĞŔåijŔçŽĎçL'ĹæIJňåőŇèČIJă

6.5 åřĘå■ŮåĚÿè¡ňæ■ćäÿžXML

éŮőécŸ

ä¡ăæČşä¡£çŤÍäÿĂäÿłPythonå■ŮåĚÿå■ŸåĆÍæŢřæ■őïijŇåźűåřĘåőČè¡ňæ■ćæĹŘXMLæăijåijŔãĂĆ

èğčåĘşæŰźæąĹ

åř¡çőą xml.etree.ElementTree åžŞéĂŽåÿÿçŤĺæĺěåĄŽèğčæđŘåůëä¡IJïijŇåĚűåőđåőČäź§åŔŕäżěåĺä¿ŃåęĆïijŇèĂČèŹŚåęĆäÿŃè£ŹäÿłåĞ¡æŢřïijŽ

```
from xml.etree.ElementTree import Element

def dict_to_xml(tag, d):
'''
Turn a simple dict of key/value pairs into XML
'''
elem = Element(tag)
for key, val in d.items():
    child = Element(key)
    child.text = str(val)
    elem.append(child)
return elem
```

äÿŃéĺćæŸŕäÿĂäÿłä¡£çŤĺä¿Ńå■ŘïijŽ

```
>>> s = { 'name': 'GOOG', 'shares': 100, 'price':490.1 }
>>> e = dict_to_xml('stock', s)
>>> e
<Element 'stock' at 0x1004b64c8>
>>>
```

è¡ňæ■ćçżŞæđIJæŸŕäÿĂäÿł Element åőđä¿ŃãĂĆåŕźäžŐI/OæŞ■ä¡IJïijŇä¡£çŤĺ xml.etree.ElementTree äÿ■çŽĎ tostring() åĞ¡æŢřå¿ĹåőźæŸŞåřśèČ¡åřĘåőČè¡ňæ■ćæĹŘäÿĂäÿłå■ŮèŁ

```
>>> from xml.etree.ElementTree import tostring
>>> tostring(e)
b'<stock><price>490.1</price><shares>100</shares><name>GOOG</name></
↪stock>'
>>>
```

åęĆæđIJä¡ăæČşçżŹæ§ŘäÿłåĚČçť'ăæůżåŁăåśđæĂğåĂijïijŇåŔŕäżěä¡£çŤĺ set() æŰźæşŢïijŽ

```
>>> e.set('_id','1234')
>>> tostring(e)
b'<stock _id="1234"><price>490.1</price><shares>100</shares><name>
↪GOOG</name>
</stock>'
>>>
```

åęĆæđIJä¡ăè£ŸæČşä£ĺæŇąåĚČçť'ăçŽĎéąžåžŘïijŇåŔŕäżěèĂČèŹŚæđĎéĂääÿĂäÿł OrderedDict æĺěäžčæŽ£äÿĂäÿłæŹőéĂŽçŽĎå■ŮåĚÿãĆčŕůåŔĆèĂČ1.7åŕŘèŁĆãĂĆ

èőĺèőž

å¡ŞåĹŻåžžXMLçŽĎæŮűåĂŹïijŇä¡ăèćńéŹŘåĹűåŔłèČ¡æđĎéĂăå■ŮçňęäÿšçśżåđŃçŽĎåĂijãĂĆä¿ŃåęĆï

```
def dict_to_xml_str(tag, d):
    '''
    Turn a simple dict of key/value pairs into XML
    '''
    parts = ['<{}>'.format(tag)]
    for key, val in d.items():
        parts.append('<{0}>{1}</{0}>'.format(key,val))
    parts.append('</{}>'.format(tag))
    return ''.join(parts)
```

éŮőécŸæŸŕåęĆæđIJä¡ăæĽŃåŁĺçŽĎåŐżæđĎéĂăçŽĎæŮűåĂŹåŔŕèČ¡äijŽçćřåĹřäÿĂäžŽéžżçČęãĂĆä¿Ń

```
>>> d = { 'name' : '<spam>' }

>>> # String creation
>>> dict_to_xml_str('item',d)
'<item><name><spam></name></item>'

>>> # Proper XML creation
>>> e = dict_to_xml('item',d)
>>> tostring(e)
b'<item><name>&lt;spam&gt;</name></item>'
>>>
```

æşĺæĎŔåĹřçÍŃåžŔçŽĎåŘŐéÍćéĆčäÿłä¿Ńå■Řäÿ■ïijŇå■Ůçňę ‘<’ åŠŇ ‘>’ èćńæŻ£æ■ćæĹŘäžĘ `<` åŠŇ `>`

äÿŃéÍćäżĚä¿ŻåŔĆèĂČïijŇåęĆæđIJä¡ăéIJĂèęĄæĽŃåŁĺåŐżè¡ñæ■ćè£ŹäžŻå■Ůçňę¨ïijŇ åŔŕäžěä¡£çŤÍ `xml.sax.saxutils` äÿ■çŽĎ `escape()` åŠŇ `unescape()` åĞ¡æŢřãĂĆä¿ŃåęĆïijŽ

```
>>> from xml.sax.saxutils import escape, unescape
>>> escape('<spam>')
'&lt;spam&gt;'
>>> unescape(_)
'<spam>'
>>>
```

éŹď’äžĘèČ¡åĹŻåžžæ■čçąőçŽĎè¿ŞåĞžåď’ŰïijŇè£ŸæIJL’åŔęåď’ŰäÿĂäÿłåŐ§åŻăæŐÍè■Řä¡ääĹŻåžž `Element` åőđä¿ŃèĂŇäÿ■æŸŕå■ŮçňęäÿšïijŇ éĆčářśæŸŕä¡£çŤĺå■ŮçňęäÿšçżĎåŘĹæđĎéĂääÿĂäÿłæŻť’åď’ğç èĂŇ `Element` åőđä¿ŃåŔŕäžěäÿ■çŤÍèĂČèŹŚèğčæđŘXMLæŰĞæIJňçŽĎæČĚåĘţäÿŃéĂŽè£Ğåď’Žçğ■æŰźåi äź§ářśæŸŕèŕť’ïijŇä¡ääŔŕäžěåIJÍäÿĂäÿłéńŸçžğæŢřæ■őçżŞæđĎäÿŁåőŇæĹŘä¡ăæĽ’ĂæIJL’çŽĎæŞ■ä¡IJïijŇåźű

6.6 èğčædŘåŠŇä£őæŤźXML

éŮőécŸ

ä¡äæČşèŕžåŔŰäÿĂäÿłXMLæŰĞæąčïijŇåŕźåőČæIJĂäÿĂäžŻä£őæŤźïijŇçĎűåŘŐåřEçżŞædIJåĘŹåŻđXM

èğčåĘşæŰźæąĹ

ä¡£çŤĺ `xml.etree.ElementTree` æĺąåİŮåŔŕäżěå¿ĹåőźæŸŞçŽĎåď˙ĎçŘĘè£ŹäžŻäżžåŁąãĂĆ çňňäÿĂæ■ěæŸŕäžěéĂŽåÿÿçŽĎæŰźåijŔæİěèğčædŘè£ŹäÿłæŰĞæąčãĂĆä¿ŃåęĆïijŇåĄĞèő¿ä¡äæIJL'äÿĂäÿłå `pred.xml` çŽĎæŰĞæąčïijŇçśżäijijäÿŃéİćè£ŹæăůïijŽ

```
<?xml version="1.0"?>
<stop>
    <id>14791</id>
    <nm>Clark &amp; Balmoral</nm>
    <sri>
        <rt>22</rt>
        <d>North Bound</d>
        <dd>North Bound</dd>
    </sri>
    <cr>22</cr>
    <pre>
        <pt>5 MIN</pt>
        <fd>Howard</fd>
        <v>1378</v>
        <rn>22</rn>
    </pre>
    <pre>
        <pt>15 MIN</pt>
        <fd>Howard</fd>
        <v>1867</v>
        <rn>22</rn>
    </pre>
</stop>
```

äÿŃéİćæŸŕäÿĂäÿłåĹľçŤĺ `ElementTree` æİèèŕžåŔŰè£ŹäÿłæŰĞæąčåźűåŕźåőČåĄŽäÿĂäžŻä£őæŤźçŽ

```
>>> from xml.etree.ElementTree import parse, Element
>>> doc = parse('pred.xml')
>>> root = doc.getroot()
>>> root
<Element 'stop' at 0x100770cb0>

>>> # Remove a few elements
>>> root.remove(root.find('sri'))
>>> root.remove(root.find('cr'))
>>> # Insert a new element after <nm>...</nm>
```

```
>>> root.getchildren().index(root.find('nm'))
1
>>> e = Element('spam')
>>> e.text = 'This is a test'
>>> root.insert(2, e)

>>> # Write back to a file
>>> doc.write('newpred.xml', xml_declaration=True)
>>>
```

åď ĎçŘEçżŞæđIJæŸŕäÿĂäÿłåČŔäÿŃéÍćè£ŹæăůæŰřçŽĎXMLæŰĞäžűïijŽ

```
<?xml version='1.0' encoding='us-ascii'?>
<stop>
    <id>14791</id>
    <nm>Clark &amp; Balmoral</nm>
    <spam>This is a test</spam>
    <pre>
        <pt>5 MIN</pt>
        <fd>Howard</fd>
        <v>1378</v>
        <rn>22</rn>
    </pre>
    <pre>
        <pt>15 MIN</pt>
        <fd>Howard</fd>
        <v>1867</v>
        <rn>22</rn>
    </pre>
</stop>
```

èőÍèőž

ä£őæŤźäÿĂäÿłXMLæŰĞæąčçżŞæđĎæŸŕå¿ĹåőźæŸ§çŽĎïijŇä¡ĘæŸŕä¡ăå£ĚéążçĽćèőřçŽĎæŸŕæĽĂæI
åřĘåőČä¡IJäÿžäÿĂäÿłåĹŮèąÍæÍěåď ĎçŘĘãĂĆä¿ŃåęĆïijŇåęĆæđIJä¡ăåĹăéŹď æ§ŘäÿłåĚČçť ăïijŇéĂŽè£ĞěřČ
`remove()` æŰźæşŢäżŐåőČçŽĎçŻť æŐěçĹűèŁĆçĆźäÿ■åĹăéŹď ãĂĆ
åęĆæđIJä¡ăæŔŠåĚěæĹŰåćđåŁăæŰřçŽĎåĚČçť ăïijŇä¡ăåŘŇæăůä¡£çŤĺçĹűèŁĆçĆźåĚČçť ăçŽĎ
`insert()` åŠŇ `append()` æŰźæşŢãĂĆ è£ŸèČ¡åŕźåĚČçť ăä¡£çŤĺçť ćåijŢåŠŇåĹĞçĽĞæŞ■ä¡IJïijŇæŕŤåęĆ
`element[i]` æĹŰ `element[i:j]`

åęĆæđIJä¡ăéIJĂèęĄåĹŻåżžæŰřçŽĎåĚČçť ăïijŇåŔŕäžěä¡£çŤĺæIJňèŁĆæŰźæąĹäÿ■æijŤçď žçŽĎ
`Element` çśżãĂĆæĹŠäžňåIJĺ6.5åřŔèŁĆåůšçżŔèŕęçżĘèőÍèőžè£ĞäžĘãĂĆ

6.7 åĹľçŤĺåŚ¡åŘ■çľžéŮťèğčæđŘXMLæŰĞæąč

éŮőécŸ

ä¡ăæČşèğčæđŘæ§ŘäÿłXMLæŰĞæąčïijŇæŰĞæąčäÿ■ä¡£çŤĺäžĘXMLåŚ¡åŘ■çľžéŮťãĂĆ

èğčåĘşæŰźæąĹ

èĂČèŹŚäÿŃéİćè£ŹäÿłäĄ£çŤĺäžĘåŚ¡åŘ■çľžéŮťçŽĎæŰĞæąčïijŽ

```
<?xml version="1.0" encoding="utf-8"?>
<top>
    <author>David Beazley</author>
    <content>
        <html xmlns="http://www.w3.org/1999/xhtml">
            <head>
                <title>Hello World</title>
            </head>
            <body>
                <h1>Hello World!</h1>
            </body>
        </html>
    </content>
</top>
```

åęĆæđIJäĄăèğčæđŘè£ŹäÿłæŰĞæąčåźűæĽğèąŇæŹőéĂŽçŽĎæ§ěèŕćïijŇäĄăäijŽåŔŚçŐřè£Źäÿłåźűäÿ■æŸ

```
>>> # Some queries that work
>>> doc.findtext('author')
'David Beazley'
>>> doc.find('content')
<Element 'content' at 0x100776ec0>
>>> # A query involving a namespace (doesn't work)
>>> doc.find('content/html')
>>> # Works if fully qualified
>>> doc.find('content/{http://www.w3.org/1999/xhtml}html')
<Element '{http://www.w3.org/1999/xhtml}html' at 0x1007767e0>
>>> # Doesn't work
>>> doc.findtext('content/{http://www.w3.org/1999/xhtml}html/head/
 ↪title')
>>> # Fully qualified
>>> doc.findtext('content/{http://www.w3.org/1999/xhtml}html/'
... '{http://www.w3.org/1999/xhtml}head/{http://www.w3.org/1999/
 ↪xhtml}title')
'Hello World'
>>>
```

ä¡ăåŔŕäžěéĂŽè£ĞářĘåŚ¡åŘ■çľžéŮťăđ´ĎçŘĘéĂżè¿ŚåŇĚèčĚäÿžäÿĂäÿłåůěåĚůçśzæİěçőĂåŇŰè£ŹäÿłèŁ

```
class XMLNamespaces:
    def __init__(self, **kwargs):
        self.namespaces = {}
        for name, uri in kwargs.items():
            self.register(name, uri)
    def register(self, name, uri):
        self.namespaces[name] = '{'+uri+'}'
    def __call__(self, path):
        return path.format_map(self.namespaces)
```

éĂŽè£ĞäÿŃéİćçŽĎæŰźåijŔä¡£çŤĺè£Źäÿłçśżïij Ž

```
>>> ns = XMLNamespaces(html='http://www.w3.org/1999/xhtml')
>>> doc.find(ns('content/{html}html'))
<Element '{http://www.w3.org/1999/xhtml}html' at 0x1007767e0>
>>> doc.findtext(ns('content/{html}html/{html}head/{html}title'))
'Hello World'
>>>
```

èőĺèőž

èğčæđŘåŘńæIJL'åŚ¡åŘ■çľ'žéŮť'çŽĎXMLæŰĞæąčäijŽæŕŤè¿ČçźĄçŘŘãĂĆ äÿŁéİćçŽĎ XMLNamespaces äżĚäżĚæŸŕåĚĄèőÿä¡ää¡£çŤĺçijľ'çŢĕåŘ■äžčæŽ£åőŇæŢť'çŽĎURIåřEåĚűåŔŸå¿Ůçĺ■å¿őç

å¿Łäÿ■åźÿçŽĎæŸŕïijŇåIJĺå§žæIJňçŽĎ ElementTree èğčæđŘäÿ■æšąæIJL'äżżä¡ŢéĂŤå¿ĎèŐůåŔŰåŚ¡åŘ■çľ'žéŮť'çŽĎä£ąæĄŕãĂĆ ä¡ĘæŸŕïijŇå¿ĆæđIJä¡ää¡£çŤĺ iterparse() åĞ¡æŢřçŽĎèŕİåřśåŔŕäżèèŐůåŔŰæŻť'åď'ŽåĚşäžŐåŚ¡åŘ■çľ'žé

```
>>> from xml.etree.ElementTree import iterparse
>>> for evt, elem in iterparse('ns2.xml', ('end', 'start-ns', 'end-
↪ns')):
... print(evt, elem)
...
end <Element 'author' at 0x10110de10>
start-ns ('', 'http://www.w3.org/1999/xhtml')
end <Element '{http://www.w3.org/1999/xhtml}title' at 0x1011131b0>
end <Element '{http://www.w3.org/1999/xhtml}head' at 0x1011130a8>
end <Element '{http://www.w3.org/1999/xhtml}h1' at 0x101113310>
end <Element '{http://www.w3.org/1999/xhtml}body' at 0x101113260>
end <Element '{http://www.w3.org/1999/xhtml}html' at 0x10110df70>
end-ns None
end <Element 'content' at 0x10110de68>
end <Element 'top' at 0x10110dd60>
>>> elem # This is the topmost element
<Element 'top' at 0x10110dd60>
>>>
```

æIJĂåŘŐäÿĂçĆźïijŇåęĆæđIJä¡ăèęĄåďĎçŘĘçŽĎXMLæŰĞæIJňéŹď'äžĘèęĄä¡£çŤĺåĹřåĚűäżŰéńŸçžğ åżžèőőä¡ăæIJĂåě¡æŸŕä¡£çŤĺ lxml åĞ¡æŢřåžŞæİěäżčæŽ£ ElementTree ãĂĆ

ä¿ŃåęĆïijŇlxml åŕźåĹľçŤÍDTDéłŇèŕĄæŰĞæąčãĂĄæŻť'åě¡çŽĎXPathæŤŕæŇĄåŠŇäÿĂäžŻåĚűäžŰéńŸçžğ
è£ŹäÿĂåřŔèŁĆåĚűåőđåŔłæŸŕæŢŹä¡ääęĆä¡Ţèőľ'XMLèğčædŘçĺ■å¿őçőĂå■ŢäÿĂçĆźãĂĆ

6.8 äÿŐåĚşçşžåđŃæŢřæ■őåžŞçŽĎäžd'äžŠ

éŮőécŸ

ä¡ăæČşåIJĺåĚşçşžåđŃæŢřæ■őåžŞäÿ■æ§ěèŕćãĂĄåćđåŁăæĹŰåĹăéŹd'èőřå¡ŢãĂĆ

èğčåĘşæŰźæąĹ

Pythonäÿ■èąĺçd'žåd'ŽèąŇæŢřæ■őçŽĎæăĞåĞĘæŰźåijŔæŸŕäÿĂäÿłçŤśåĚČçżĎæđĎæĹŘçŽĎåžŔåĹŮãĂ

```
stocks = [
    ('GOOG', 100, 490.1),
    ('AAPL', 50, 545.75),
    ('FB', 150, 7.45),
    ('HPQ', 75, 33.2),
]
```

ä¿Ìæ■őPEP249ïijŇéĂŽè£Ğè£Źçğ■å¡ćåijŔæŔŘä¿ŻæŢřæ■őïijŇ
åŔŕäžěå¿ĹåőźæŸ§çŽĎä¡£çŤÍPythonæăĞåĞĘæŢřæ■őåžŞAPIåŠŇåĚşçşžåđŃæŢřæ■őåžŞè£ŻèąŇäžd'äžŠãĂĆ
æĽĂæIJĽæŢřæ■őåžŞäÿŁçŽĎæŞ■ä¡IJéČ¡éĂŽè£ĞSQLæ§ěèŕćèŕ■åŔěæÌěåőŇæĹŘãĂĆæŕŔäÿĂèąŇè¿ŞåĚěè¿Ş

äÿžäžĘæijŤçd'žèŕť'æŸŐïijŇä¡ăåŔŕäžěä¡£çŤÍPythonæăĞåĞĘåžŞäÿ■çŽĎ `sqlite3`
æĺąåİŮãĂĆ åęĆædIJä¡ăä¡£çŤÍçŽĎæŸŕäÿĂäÿłäÿ■åŘŇçŽĎæŢřæ■őåžŞ(æŕŤåęĆMySqlãĂĄPostgresqlæĹŰèĂ
è£Ÿå¿ŮåőĽèčĚçŽÿåžŤçŽĎçňňäÿĽæŰźæĺąåİŮæÌěæŔŘä¿ŻæŤŕæŇĄãĂĆ
äÿ■è£ĞçŽÿåžŤçŽĎçijŰçÍŇæŐěåŔčåĞääźŐéČ¡æŸŕäÿĂæăůçŽĎïijŇéŹd'äžĘäÿĂçĆźçĆźżĘå¿őåůőåĹńåd'Ůã

çňňäÿĂæ■ěæŸŕè£đæŐěåĹŕæŢřæ■őåžŞãĂĆéĂŽåÿÿä¡äèęĄæĽğèąŇ `connect()`
åĞ¡æŢřïijŇ çżŹåőČæŔŘä¿ŻäÿĂäžŻæŢřæ■őåžŞåŘ■ãĂĄäÿżæIJžãĂĄçŤÍæĹůåŘ■ãĂĄåŕĘçăĄåŠŇåĚűäžŰå£Ě

```
>>> import sqlite3
>>> db = sqlite3.connect('database.db')
>>>
```

äÿžäžĘåd'ĎçŘĘæŢřæ■őïijŇäÿŃäÿĂæ■ěä¡äéIJÄèęĄåĹŻåžžäÿĂäÿłæÿÿæăĞãĂĆ
äÿĂæŮęä¡ăæIJĽäžĘæÿÿæăĞïijŇéĆčäźĹä¡ăåŕśåŘŕäžěæĽğèąŇSQLæ§ěèŕćèŕ■åŘěäžĘãĂĆæŕŤåęĆïijŽ

```
>>> c = db.cursor()
>>> c.execute('create table portfolio (symbol text, shares integer,␣
↪price real)')
<sqlite3.Cursor object at 0x10067a730>
>>> db.commit()
>>>
```

äÿžäžĘåŘŚæŢřæ■őåžŞèąÍäÿ■æŘŠåĚěåd'ŽæÌąèőřå¡ŢïijŇä¡£çŤÍçśżäijijäÿŃéÌćè£ŹæăůçŽĎèŕ■åŘěïijŽ

```
>>> c.executemany('insert into portfolio values (?,?,?)', stocks)
<sqlite3.Cursor object at 0x10067a730>
>>> db.commit()
>>>
```

äÿžäžĘæĽ'ğèąŇæ§Řäÿłæ§ěèŕćïijŇä¡£çŤĺåČŔäÿŃéIćè£ŹæăůçŽĎèŕ■åŔěïijŽ

```
>>> for row in db.execute('select * from portfolio'):
...     print(row)
...
('GOOG', 100, 490.1)
('AAPL', 50, 545.75)
('FB', 150, 7.45)
('HPQ', 75, 33.2)
>>>
```

åęĆæđIJä¡ăæČşæŐěåŔŮçŤĺæĹůè¿ŞåĚěä¡IJäÿžåŔĆæŢřæIěæĽ'ğèąŇæ§ěèŕćæŞ■ä¡IJïijŇå£Ěéążçąőä£Ia¡ă

```
>>> min_price = 100
>>> for row in db.execute('select * from portfolio where price >= ?
↪',
                          (min_price,)):
...     print(row)
...
('GOOG', 100, 490.1)
('AAPL', 50, 545.75)
>>>
```

èőÍèőž

åIJĺæŕŤè¿Čä¡ŐçŽĎçžğåĹńäÿŁåŠŇæŢřæ■őåžŞäžd'äžŠæŸŕéIđåÿÿçőĂå■ŢçŽĎãĂĆ
ä¡ăåŔłéIJĂæŔŘä¿ŻSQLèŕ■åŔěåźűèřČçŤĺçŻÿåžŤçŽĎæĺąåIŮåŕšåŔŕäžěæŻt'æŰřæĹŰæŔŘåŔŰæŢřæ■őäžĘãĂ
èŹ¡èŕt'åęĆæ■d'ïijŇè£ŸæŸŕæIJL'äÿĂäžŻæŕŤè¿ČæčŸæL'ŃçŽĎçżĘèŁĆéŮőécŸéIJĂèęĄä¡äéĂŘäÿłåĹŮåĞžåČ

äÿĂäÿłéŽ¿çĆźæŸŕæŢřæ■őåžŞäÿ■çŽĎæŢřæ■őåŠŇPythonçśžåđŃçŻt'æŐěçŽĎæŸåårĎãĂĆ
åŕźäžŐæŮěæIJ§çśžåđŃïijŇéĂŽåÿÿåŔŕäžěä¡£çŤĺ `datetime` æĺąåIŮäÿ■çŽĎ `datetime`
åőđä¿ŃïijŇ æĹŰèĂĚåŔŕèČ¡æŸŕ `time` æĺąåIŮäÿ■çŽĎçşżçż§æŮűéŮt'æĹşãĂĆ
åŕźäžŐæŢřå■ŮçśžåđŃïijŇçL'źåĹńæŸŕä¡£çŤĺåĹřårŔæŢřçŽĎéĞŚèđ■æŢřæ■őïijŇåŔŕäžěçŤĺ
`decimal` æĺąåIŮäÿ■çŽĎ `Decimal` åőđä¿ŃæIěèąĺçđ'žãĂĆ
äÿ■åźÿçŽĎæŸŕïijŇårźäžŐäÿ■åŘŇçŽĎæŢřæ■őåžŞèĂŇèĺĂåĚůä¡ŞæŸäårĎèğĎåĹŹæŸŕäÿ■äÿĂæăůçŽĎïijŇä¡

åŔęåđ'ŰäÿĂäÿłæŻt'åŁåđ'■æIĆçŽĎéŮőécŸårśæŸŕSQLèŕ■åŔěå■ŮçňęäÿšçŽĎædĎéĂããĂĆ
ä¡ăå■ČäÿĞäÿ■èęĄä¡£çŤĺPythonå■ŮçňęäÿšæăijåijŔåŇŰæŞ■ä¡IJçňę(åęĆ%)æĹŰèĂĚ
`.format()` æŰźæşŢæIěåĹŻåžžè£ŹæăůçŽĎå■Ůçňęäÿšã ĂĆ
åęĆæđIJäijăéĂŠçżŹè£ŹäžŻæăijåijŔåŇŰæŞ■ä¡IJçňęçŽĎåĂijæIěèĞłäžŐçŤĺæĹůçŽĎè¿ŞåĚěïijŇéĆčäźĹä¡äçŽ
http://xkcd.com/327)ãĂĆ æ§èèŕćèŕ■åŔěäÿ■çŽĎéĂŽéĚ■çňę ?
æŇĞçd'žåŘŐåŔřæŢřæ■őåžŞä¡£çŤĺåőČèĞłåůśçŽĎå■Ůçňęäÿšæ Ż£æ■ćæIJžåĹűïijŇè£ŹæăůæŻt'åŁäçŽĎåőL'å

äÿ■åźÿçŽĎæŸŕïijŇäÿ■åŘŇçŽĎæŢřæ■őåžŞåŘŐåŔřåŕźäžŐéĂŽéĚ■çňęçŽĎä¡£çŤĺæŸŕäÿ■äÿĂæăůçŽĎã

? æĹŰ %s ïijŇ è£ŸæIJĽåĚűäžŰäÿĂäžŻä¡£çŤĺäžĘäÿ■åŘŇçŽĎçňęåŔůïijŇæŕŤåęĆ:0æĹŰ:1æİěæŇĞçd'žåŔĆ
åŘŇæăůçŽĎïijŇä¡ãè£ŸæŸŕå¿ŮåŐżåŔĆèĂČä¡ää¡£çŤĺçŽĎæŢřæ■őåžŞæĺąåİŮçŻÿåžŤçŽĎæŰĞæąčãĂĆ
äÿĂäÿłæŢřæ■őåžŞæĺąåİŮçŽĎ paramstyle åšđæĂğåŇĚåŘńäžĘåŔĆæŢřåijŢçŤĺéčŐæäijçŽĎä£ąæĄŕãĂĆ

åŕźäžŐçőĂå■ŢçŽĎæŢřæ■őåžŞæŢřæ■őçŽĎèŕżåĘŹéŮőécŸïijŇä¡£çŤĺæŢřæ■őåžŞAPIéĂŽåÿÿéİđåÿÿçőĂ
åęĆæđIJä¡ãèęĄåd'ĎçŘĘæŻt'åŁãåd'■æİĆçŽĎéŮőécŸïijŇåżžèőőä¡ää¡£çŤĺæŻt'åŁãéńŸçžğçŽĎæŐěåŔčïijŇæí
çśżäijij SQLAlchemy è£ŹæăůçŽĎåžŞåĚĄèőÿä¡ää¡£çŤĺPythonçśżæİèèąĺçd'žäÿĂäÿłæŢřæ■őåžŞèąĺïijŇ
åźűäÿŤèČ¡åIJĺéŽŘèŮŔåžŢåśĆSQLçŽĎæČĚåĘţäÿŃåőđçŐřåŘĎçğ■æŢřæ■őåžŞçŽĎæŞ■ä¡IJãĂĆ

6.9 çijŰçăĄåŠŇèğčçăĄå■ĄåĚ■è£ŻåĹűæŢř

éŮőécŸ

ä¡ăæČşåřĘäÿĂäÿłå■ĄåĚ■è£ŻåĹűå■Ůçňęäÿšèğčçă ĄæĹŘäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšæĹŰèĂĚåřĘäÿĂäÿłå

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăåŔłæŸŕçőĂå■ŢçŽĎèğčçăĄæĹŰçijŰçăĄäÿĂäÿłå■ĄåĚ■è£ŻåĹűçŽĎåŐ§ågŃå■ŮçňęäÿšïijŇå
æĺąåİŮãĂĆä¿ŃåęĆïijŽ

```
>>> # Initial byte string
>>> s = b'hello'
>>> # Encode as hex
>>> import binascii
>>> h = binascii.b2a_hex(s)
>>> h
b'68656c6c6f'
>>> # Decode back to bytes
>>> binascii.a2b_hex(h)
b'hello'
>>>
```

çśżäijijçŽĎåŁ§èČ¡åŘŇæăůåŔŕäžěåIJĺ base64 æĺąåİŮäÿ■æL'¿åĹřãĂĆä¿ŃåęĆïijŽ

```
>>> import base64
>>> h = base64.b16encode(s)
>>> h
b'68656C6C6F'
>>> base64.b16decode(h)
b'hello'
>>>
```

èőĺèőž

åď ğéČĺåĹĘæČĚåĘţäÿŃïijŇéĂŽè£Ğä¡£çŢĺäÿŁè£řçŽĎåĞ¡æŢřæİěè¡ñæ■ćå■ĄåĚ■è£ŻåĹűæŸŕå¿ĹçőĂå■Ţäÿ
Łéİćäÿďçğ■æŁĂæIJŕçŽĎäÿżèęĄäÿ■åŘŇåIJĺäžŐåď ğåřŕåĘźçŽĎåď ĎçŘĘãĂĆ
åĞ¡æŢř `base64.b16decode()` åŠŇ `base64.b16encode()`
åŘłèČ¡æŞ■ä¡IJåď ğåĘŹå¡ćåijŔçŽĎå■ĄåĚ■è£ŻåĹűå■Ůæŕ■ïijŇ èĂŇ `binascii`
æĺąåİŮäÿ■çŽĎåĞ¡æŢřåď ğåřŕåĘźéČ¡èČ¡åď ĎçŘĘãĂĆ

è£ŸæIJL'äÿĂçĆźéIJĂèęĄæşĺæĎŔçŽĎæŸŕçijŰçăĄåĞ¡æŢřæL'Ăäžğç Ť§çŽĎè¿ŞåĞžæĂżæŸŕäÿĂäÿłå■Ů
åęĆæđIJæČşåijžåĹűäžéUnicodeå¡ćåijŔè¿ŞåĞžïijŇä¡ăéIJĂèęĄåćđåŁääÿĂäÿłéćİåď ŰçŽĎçŢŇéİćæ■ěéłď ãĂĆ

```
>>> h = base64.b16encode(s)
>>> print(h)
b'68656C6C6F'
>>> print(h.decode('ascii'))
68656C6C6F
>>>
```

åIJĺèğčçăĄå■ĄåĚ■è£ŻåĹűæŢřæŮűïijŇåĞ¡æŢř `b16decode()`
åŠŇ `a2b_hex()` åŔŕäžěæŐěåŔŮå■ŮèŁĆæĹŰunicodeå■Ůçňęäÿšã ĂĆ
ä¡ĘæŸŕïijŇunicodeå■Ůçňęäÿšå£ĚéążäžĚäžĚåŔłåŇĚåŘńASCIIçijŰçăĄçŽĎå■ĄåĚ■è£ŻåĹűæŢřãĂĆ

6.10 çijŰçăĄèğčçăĄBase64æŢřæ■ő

éŮőécŸ

ä¡ăéIJĂèęĄä¡£çŢĺBase64æăijåijŔèğčçăĄæĹŰçijŰçăĄäžŇè£ŻåĹűæŢřæ■őãĂĆ

èğčåĘşæŰźæąĹ

`base64` æĺąåİŮäÿ■æIJL'äÿď äÿłåĞ¡æŢř `b64encode()` and `b64decode()`
åŔŕäžěåÿőä¡äèğčåĘşè£ŹäÿłéŮőécŸãĂĆä¿ŃåęĆ;

```
>>> # Some byte data
>>> s = b'hello'
>>> import base64

>>> # Encode as Base64
>>> a = base64.b64encode(s)
>>> a
b'aGVsbG8='

>>> # Decode from Base64
>>> base64.b64decode(a)
b'hello'
>>>
```

èőĺèőž

Base64çijŰçăĄäžËäžËçŤĺäžŐéİćåŘŚå■ŮèŁĆçŽĎæŢřæ■őæŕŤåęĆå■ŮèŁĆå■Ůçňęäÿšåš
æ■ď'åď'ŰïijŇçijŰçăĄåď'ĎçŘĘçŽĎè¿ŞåĞžçżŞæđIJæĂżæŸŕäÿĂäÿłå■ŮèŁĆå■Ůçňęäÿšã
åęĆæđIJä¡ăæČşæůůåŘĹä¡£çŤĺBase64çijŰçăĄçŽĎæŢřæ■őåŠŇUnicodeæŰĞæIJñïijŇä¡åå£Ëéążæůżå

```
>>> a = base64.b64encode(s).decode('ascii')
>>> a
'aGVsbG8='
>>>
```

å¡Şèğčçă Ą Base64çŽĎæŮűåĂŹïijŇå■ŮèŁĆå■Ůçňęäÿšå
ä¡ĘæŸŕïijŇUnicodeå■Ůçňęäÿšå

6.11 èŕżåĘŹäžŇè£ŻåĹűæŢřçżĎæŢřæ■ő

éŮőécŸ

ä¡ăæČşèŕżåĘŹäÿĂäÿłäžŇè£ŻåĹűæŢřçżĎçŽĎçżŞæđĎåŇŰæŢřæ■őåĽřPythonåĚČçżĎäÿ■ãĂĆ

èğčåĘşæŰźæąĹ

åŕŕäżěä¡£çŤĺ struct æĺąåİŮåď'ĎçŘĘäžŇè£ŻåĹűæŢřæ■őãĂĆ
äÿŇéİćæŸŕäÿĂæőţçď'žä¿ŃäžčçăĄåŕĘäÿĂäÿłPythonåĚČçżĎåĹŮèąĺåĘŹåĚěäÿĂäÿłäžŇè£ŻåĹűæŰĞäžűïijŇå
struct åŕĘæŕŘäÿłåĚČçżĎçijŰçăĄäÿžäÿĂäÿłçżŞæđĎä¡ŞãĂĆ

```
from struct import Struct
def write_records(records, format, f):
    '''
    Write a sequence of tuples to a binary file of structures.
    '''
    record_struct = Struct(format)
    for r in records:
        f.write(record_struct.pack(*r))

# Example
if __name__ == '__main__':
    records = [ (1, 2.3, 4.5),
                (6, 7.8, 9.0),
                (12, 13.4, 56.7) ]
    with open('data.b', 'wb') as f:
        write_records(records, '<idd', f)
```

æIJL'å¿Ĺåď'Žçğ■æŰźæşŢæİèèŕżåŘŰè£ŹäÿłæŰĞäžűåźűè£ŤåŻđäÿĂäÿłåĚČçżĎåĹŮèąĺãĂĆ
éęŰåĚĹïijŇåęĆæđIJä¡ăæL'ŞçőŮäžěåİŮçŽĎå¡ćåijŔåćđéĞŔèŕżåŘŰæŰĞäžűïijŇä¡ååŕŕäżěè£ŹæăůåĄŽïijŽ

```
from struct import Struct

def read_records(format, f):
    record_struct = Struct(format)
    chunks = iter(lambda: f.read(record_struct.size), b'')
    return (record_struct.unpack(chunk) for chunk in chunks)

# Example
if __name__ == '__main__':
    with open('data.b','rb') as f:
        for rec in read_records('<idd', f):
            # Process rec
            ...
```

åęĆæđIJä¡ăæČşåřĘæŢť äÿłæŰĞäżűäÿĂæňąæĂğèŕżåŔŰåĹřäÿĂäÿłå■ŮèŁĆå■Űçňęäÿšäÿ■ïijŇçĎűåŘŐåĹ

```
from struct import Struct

def unpack_records(format, data):
    record_struct = Struct(format)
    return (record_struct.unpack_from(data, offset)
            for offset in range(0, len(data), record_struct.size))

# Example
if __name__ == '__main__':
    with open('data.b', 'rb') as f:
        data = f.read()
    for rec in unpack_records('<idd', data):
        # Process rec
        ...
```

äÿď çğ■æČĚåĘţäÿŃçŽĎçżŞæđIJéČ¡æŸŕäÿĂäÿłåŔŕè£ŤåŻđçŤÍæİěåĹŻåżžèŕéæŰĞäżűçŽĎåŐ§åğŃåĚČç

èőĺèőž

åŕźäžŐéIJĂèęĄçijŰçăĄåŠŇèğčçăĄäžŇè£ŻåĹűæŢřæ■őçŽĎçĺŃåžŔèĂŇèĺĂïijŇéĂŽåÿÿäijŽä¡£çŤÍ `struct` æĺąåİŮãĂĆ äÿžäžĘåčřæŸŐäÿĂäÿłæŰřçŽĎçżŞæđĎä¡ŞïijŇåŔłéIJĂèęĄåČŔè£Źæăůåĺ Żåżžäÿ Ääÿł `Struct` åőđä¿Ńå■şåŔŕïijŽ

```
# Little endian 32-bit integer, two double precision floats
record_struct = Struct('<idd')
```

çżŞæđĎä¡ŞéĂŽåÿÿäijŽä¡£çŤÍäÿĂäžŻçżŞæđĎçăĄåĂij, d, fç■Ľ [åŔĆèĂČ PythonæŰĞæąč]ãĂĆ è£ŹäžŻäžčçăĄåĹĘåĹńäżčèąĺæ§Řäÿłç ĽźåőŽçŽĎäžŇè£ŻåĹűæŢřæ■őçśżåđŃåęĆ32ä¡■ çňňäÿĂäÿłå■Ůçňę < æŃĞåőŽäžĘå■ŮèŁĆéąžåžŔãĂĆåIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇåőČèąĺçď ž”ä¡Őä¡■åIJÍåĽ■”ã æŽť æŤźè£Źäÿłå■Ůçňęäÿž > èąĺçď žéńŸä¡■åIJÍåĽ■ïijŇæĹŰèĂĚæŸŕ ! èąĺçď žç¡ŚçżIJå■ŮèŁĆéąžåžŔãĂĆ

äžğçŤ§çŽĎ `Struct` åőđä¿ŇæIJĽå¿Ĺåď ŽåśđæĂğåŠŇæŰźæşŢçŤĺæİěæŞ■ä¡IJçŻÿåžŤçśżåđŃçŽĎçżŞæ

size åśđæĂğåŇĚåŘńäžĘçżŞæđĎçŽĎå■ŮèŁĆæŢřïijŇè£ŹåIJÍI/Oæŝ■ä¡IJæŮűéİđåÿÿæIJĽçŤĺãĂĆ pack() åŠŇ unpack() æŰźæşŢèćńçŤĺæİěæĽŞåŇĚåŠŇèğčåŇĚæŢřæ■őãĂĆæŕŤåęĆïijŽ

```
>>> from struct import Struct
>>> record_struct = Struct('<idd')
>>> record_struct.size
20
>>> record_struct.pack(1, 2.0, 3.0)
b
↪'\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\x00\x00\x00\x00\x00\x00\x08@
↪'
>>> record_struct.unpack(_)
(1, 2.0, 3.0)
>>>
```

æIJĽæŮűåĂŹä¡ăè£ŸäijŽçIJŃåĹř pack() åŠŇ unpack() æŞ■ä¡IJäżěæĺąåİŮçžğåĹńåĞ¡æŢřèćńèřČçŤĺïijŇçśżäijijäÿŃéİćè£ŹæăůïijŽ

```
>>> import struct
>>> struct.pack('<idd', 1, 2.0, 3.0)
b
↪'\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\x00\x00\x00\x00\x00\x00\x08@
↪'
>>> struct.unpack('<idd', _)
(1, 2.0, 3.0)
>>>
```

è£Źæăůåŕŕäżěåůěä¡IJïijŇä¡ĘæŸŕæĎ§èğĽæšąæIJĽåőđä¿ŃæŰźæşŢéĆčäźĹäijŸéŽĚïijŇçĽźåĹńæŸŕåIJĺ éĂŽè£ĞåĹŻåżžäÿĂäÿł Struct åőđä¿ŃïijŇæäijåijŔäżčçăĄåŔłäijŽæŇĞåőŽäÿĂæňąåźűäÿŤæĽĂæIJĽçŽĎæ è£Źæăůäÿ‌Ăæİěäżčçă‌Ąçżť æŁď åřśåŔŸå¿ŮæŽť åŁăçőĂå■ŢäžĘ(åŽăäÿžä¡ăåŔłéIJĂèęĄæŤźåŔŸäÿĂåď Ďäżčçă

èŕżåŔŰäžŇè£ŻåĹűçżŞæđĎçŽĎäżčçăĄèęĄçŤĺåĹřäÿĂäžŻéİđåÿÿæIJĽèűčèĂŇäijŸç¿ŐçŽĎçijŰçĺŃæŁĂ åIJĺåĞ¡æŢřãĂĂread_records äÿ■ïijŇiter() èćńçŤĺæİěåĹŻåżžäÿĂäÿłè£ŤåŽđåŻžåőŽåď ğåřŔæŢř■őå è£Źäÿłè£■äżčåŹĺäijŽäÿ■æŰ■çŽĎèřČçŤĺäÿĂäÿłçŤĺæĹůæŔŘä¿ŻçŽĎåŔŕèřČçŤĺåŕźèśą(æŕŤåęĆ lambda: f.read(record_struct.size))ïijŇ çŻť åĹřåőČè£ŤåŽđäÿĂäÿłçĽźæőŁçŽĎåĂij(åęĆb’‘

```
>>> f = open('data.b', 'rb')
>>> chunks = iter(lambda: f.read(20), b'')
>>> chunks
<callable_iterator object at 0x10069e6d0>
>>> for chk in chunks:
... print(chk)
...
b'\x01\x00\x00\x00ffffff\x02@\x00\x00\x00\x00\x00\x00\x12@'
b'\x06\x00\x00\x00333333\x1f@\x00\x00\x00\x00\x00\x00"@'
b'\x0c\x00\x00\x00\xcd\xcc\xcc\xcc\xcc\xcc*@\x9a\x99\x99\x99\x99YL@'
>>>
```

åęĆä¡ăæĽĂèğĄïijŇåĹŻåżžäÿĂäÿłåŔŕè£■äżčåŕźèśąçŽĎäÿĂäÿłåŐ§åŽăæŸŕåőČèČ¡åĚĄèőÿä¡£çŤĺäÿĂäÿ åęĆæďIJä¡ăäÿ■ä¡£çŤĺè£Źçğ■æŁĂæIJŕïijŇéĆčäźĹäżčçăĄåŔŕèČ¡äijŽåČŔäÿŃéİćè£ŹæăůïijŽ

```
def read_records(format, f):
    record_struct = Struct(format)
    while True:
        chk = f.read(record_struct.size)
        if chk == b'':
            break
        yield record_struct.unpack(chk)
```

åIJÍåĞ¡æŢř unpack_records() äÿ■ä¡£çŤÍäžĘåŔęåď ŰäÿĂçğ■æŰźæşŢ unpack_from() ãĂĆ unpack_from() åŕźäžŐäżŐäÿĂäÿłåď ğåđŃäžŇè£ŻåĹűæŢřçżĎäÿ■æŔŘåŔŰäžŇè åŻääÿžåőČäÿ■äijŽäžğçŤ§äżżä¡ŢçŽĎäÿť æŮűåŕźèśąæĹŰèĂĚè£Żèą ŇåĘĚå■Ÿåď ■åĹűæŞ■ä¡IJãĂĆ ä¡ăåŔłéIJĂèęAçżŹåőČäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšæĹŰæŢřçżĎ)åŠŇäÿĂäÿłå■ŮèŁĆåĄŔçğżéĞŔïijŇåőČäijŽä

åęĆæđIJä¡ăä¡£çŤÍ unpack() æÍěäžčæŻ£ unpack_from() ïijŇ ä¡ăéIJĂèęĄä£őæŤźäžčçăĄæÍěæđĎéĂăåď ğéĞŔçŽĎåřŔçŽĎåĹĞçĽĞäžěåŔŁè£ŻèąŇåĄŔçğżéĞŔçŽĎèőąçőŮ

```
def unpack_records(format, data):
    record_struct = Struct(format)
    return (record_struct.unpack(data[offset:offset + record_struct.
↪size])
            for offset in range(0, len(data), record_struct.size))
```

è£Źçğ■æŰźæąĹéŹď äžĘäžčçăĄçIJŃäÿŁåŐżå¿Låď ■æÍĆåď ŰïijŇè£Ÿå¿ŮåĄŽå¿Låď ŽéćÍåď ŰçŽĎåůěä¡Í åď ■åĹűæŢřæ■őäžěåŔŁæđĎéĂăåřŔçŽĎåĹĞçĽĞåŕźèśąăĂĆ åęĆæđIJä¡ăåĞĘåď ĞäżŐèŕżåŔŰåĹřçŽĎäÿĂäÿłå äijŽèąĺçŐřçŽĎæŻť åĞžèĽšãĂĆ

åIJÍèğčåŇĚçŽĎæŮűåĂŹïijŇcollections æĺąåÍŮäÿ■çŽĎåŚ¡åŘ■åĚČçżĎåŕźèśąæĹŰèőÿæŸŕä¡ăæČşè åőČåŔŕäžěèőľä¡ăçżŹè£ŤåŻđåĚČçżĎèő¿ç¡őåšđæĂğåŘ■çğřãĂĆä¿ŃåęĆïijŽ

```
from collections import namedtuple

Record = namedtuple('Record', ['kind','x','y'])

with open('data.p', 'rb') as f:
    records = (Record(*r) for r in read_records('<idd', f))

for r in records:
    print(r.kind, r.x, r.y)
```

åęĆæđIJä¡ăçŽĎçÍŃåžŔéIJĂèęĄåď ĎçŘĘåď ğéĞŔçŽĎäžŇè£ŻåĹűæŢřæ■őïijŇä¡ăæIJĂåě¡ä¡£çŤÍ numpy æĺąåÍŮãĂĆ ä¿ŃåęĆïijŇä¡ăåŔŕäžěåřĘäÿĂäÿłäžŇè£ŻåĹűæŢřæ■őèŕżåŔŰåĹřäÿĂäÿłçžŞæđĎåŇŰæŢřçż

```
>>> import numpy as np
>>> f = open('data.b', 'rb')
>>> records = np.fromfile(f, dtype='<i,<d,<d')
>>> records
array([(1, 2.3, 4.5), (6, 7.8, 9.0), (12, 13.4, 56.7)],
dtype=[('f0', '<i4'), ('f1', '<f8'), ('f2', '<f8')])
>>> records[0]
(1, 2.3, 4.5)
>>> records[1]
```

```
(6, 7.8, 9.0)
>>>
```

æIJĂåŘŐæŔŘäÿĂçĆźïijŇåęĆæđIJä¡ăéIJĂèęĄäżŐåůšç§ěçŽĎæŰĞäžűæăijåijŔ(åęĆåŻ¿çĽĞæăijåijŔïijŇ
åĚĹæčĂæ§ěçIJŃçIJŃPythonæŸŕäÿ■æŸŕåůšçżŔæŔŘä¿ŻäžEçŐřå■ŸçŽĎæĺąåİŮãĂĆåŻăäÿžäÿ■åĹřäÿĞäÿ■å¿

6.12 èŕżåŔŰåţŇåěŮåŠŇåŔŕåŔŸéŢ£äžŇè£ŻåĹűæŢřæ■ő

éŮőécŸ

ä¡ăéIJĂèęĄèŕżåŔŰåŇĚåŘńåţŇåěŮæĹŰèĂĚåŔŕåŔŸéŢ£èőřå¡ŢéŻęåŘĹçŽĎåď■æİĆäžŇè£ŻåĹűæăijåijŔ

èğčåĘşæŰźæąĹ

`struct` æĺąåİŮåŘŕèćńçŤĺæİěçijŰçăĄ/èğčçăĄåĞääźŐæL'ĂæIJL'çśżåđŃçŽĎäžŇè£ŻåĹűçŽĎæŢřæ■őçż
æİěèąĺçđ'žäÿĂäÿłçżĎæĹŘäÿĂçşżåĹŮåđ'Žè¿źå¡ćçŽĎçĆźçŽĎéŻęåŘĹïijŽ

```
polys = [
    [ (1.0, 2.5), (3.5, 4.0), (2.5, 1.5) ],
    [ (7.0, 1.2), (5.1, 3.0), (0.5, 7.5), (0.8, 9.0) ],
    [ (3.4, 6.3), (1.2, 0.5), (4.6, 9.2) ],
]
```

çŐřåIJĺåĄĞèő¿è£ŹäÿłæŢřæ■őèćńçijŰçăĄåĹřäÿĂäÿłäżëäÿŇåĹŮåđ't'éČĺåijĂåğŇçŽĎäžŇè£ŻåĹűæŰĞäż

```
+------+--------+-----------------------------------+
|Byte  | Type   |  Description                      |
+======+========+===================================+
|0     | int    |  æŰĞäżűäżčçăĄïijĹ0x1234ïijŇåřŔçńŕïijL'          |
+------+--------+-----------------------------------+
|4     | double |  x çŽĎæIJĂåřŔåĂijïijĹåřŔçńŕïijL'                |
+------+--------+-----------------------------------+
|12    | double |  y çŽĎæIJĂåřŔåĂijïijĹåřŔçńŕïijL'                |
+------+--------+-----------------------------------+
|20    | double |  x çŽĎæIJĂåd'ğåĂijïijĹåřŔçńŕïijL'                 |
+------+--------+-----------------------------------+
|28    | double |  y çŽĎæIJĂåd'ğåĂijïijĹåřŔçńŕïijL'                 |
+------+--------+-----------------------------------+
|36    | int    |  äÿL'èğŠå¡ćæŢřéĞŔïijĹåřŔçńŕïijL'                |
+------+--------+-----------------------------------+
```

çt'ğèů§çİĂåđ't'éČĺæŸŕäÿĂçşżåĹŮçŽĎåđ'Žè¿źå¡ćèőřå¡ŢïijŇçijŰçăĄæăijåijŔåęĆäÿŇïijŽ

```
+------+--------+-------------------------------------------+
|Byte  | Type   |  Description                              |
+======+========+===========================================+
```

```
|0      | int     |  èőřå¡ŢéŢ£åžęïijĹNå■ŮèŁĆïijĽ'                          ␣
↪  |
+------+--------+-----------------------------------------+
|4-N    | Points |  (X,Y) åİŘæăĞïijŇäżĕæţőçĆźæŢřèąĺçd'ž                  ␣
↪   |
+------+--------+-----------------------------------------+
```

äÿžäžĘåĘŹèŁŹæăůçŽĎæŰĞäżűïijŇä¡ăåŔŕäżěä¡£çŤĺåęĆäÿŃçŽĎPythonäżčçăĄïijŽ

```python
import struct
import itertools

def write_polys(filename, polys):
    # Determine bounding box
    flattened = list(itertools.chain(*polys))
    min_x = min(x for x, y in flattened)
    max_x = max(x for x, y in flattened)
    min_y = min(y for x, y in flattened)
    max_y = max(y for x, y in flattened)
    with open(filename, 'wb') as f:
        f.write(struct.pack('<iddddi', 0x1234,
                            min_x, min_y,
                            max_x, max_y,
                            len(polys)))
        for poly in polys:
            size = len(poly) * struct.calcsize('<dd')
            f.write(struct.pack('<i', size + 4))
            for pt in poly:
                f.write(struct.pack('<dd', *pt))
```

åřĘæŢřæ■őèŕżåŔŰåŻđæİěçŽĎæŮűåĂŹïijŇåŔŕäżěåĹľçŤĺåĞ¡æŢř struct.unpack() ïijŇäżčçăĄå¿ĹçŻÿäijijïijŇå§žæIJňåřśæŸŕäÿŁéİćåĘŹæŞ■ä¡IJçŽĎéĂĘåžŔãĂĆåęĆäÿŃïijŽ

```python
def read_polys(filename):
    with open(filename, 'rb') as f:
        # Read the header
        header = f.read(40)
        file_code, min_x, min_y, max_x, max_y, num_polys = \
            struct.unpack('<iddddi', header)
        polys = []
        for n in range(num_polys):
            pbytes, = struct.unpack('<i', f.read(4))
            poly = []
            for m in range(pbytes // 16):
                pt = struct.unpack('<dd', f.read(16))
                poly.append(pt)
            polys.append(poly)
    return polys
```

åř¡çőąèŁŹäÿłäżčçăĄåŔŕäżěåůěä¡IJïijŇä¡ĘæŸŕéĞŇéİćæůůæİĆäžĘå¿Ĺåď'ŽèŕżåŔŰãĂĄèğčåŇĚæŢřæ■őçż
éĆčæIJłåĚ■äź§åď'łçźĄæİĆäžĘçĆźãĂĆåŽăæ■ď'å¿ĹæŸ¿çĎűåžŤèŕěæIJĽåŘęäÿĂçğ■èğčåĘşæŰźæşŢåŔŕäżěçő

åIJÍæIJňářŔèŁĆæŐěäýŃæÍěçŽĎéČĺåĹĘïijŇæĹŚäijŽéĂŘæ■ěæijŤçd'žäýĂäýłæŽt'åŁääijŸçğĂçŽĎèğčæć
çŻőæăĞæŸŕåŘŕäžěçżŹçÍŃåžŘåŚŸæŔŘä¿ŻäýĂäýłénŸçžğçŽĎæŰĞäžűæäijåijŔåŇŰæŰźæşŢïijŇåźűçőĂåŇŰ
æIJňářŔèŁĆæŐěäýŃæÍěçŽĎéČĺåĹĘäžčçăĄåžŤèŕěæŸŕæŢt'æIJňäźęäÿ■æIJĂåd'■æÍĆæIJĂénŸçžğçŽĎä¿Ńå■Í
äýĂåőŽèęĄäżŤçżEçŽĎéŸĚèŕżæĹŚäžňçŽĎèőÍèőžéČĺåĹĘïijŇåŘęåd'Űäź§èęĄåŔĆèĂČäýŃåĚűäžÚçńäèŁĆåĘ

éęŰåĚĹïijŇå¡Şèŕżå ŔŰå■ÙèŁĆæŢřæ■őçŽĎæŮűåĂŹïijŇéĂŽåÿÿåIJÍæŰĞäžűåijĂåğŃéČĺåĹĘäijŽåŇĚåŘ
ář¡çőąstructæÍąåİŮåŘŕäžěèğčåŇĚè£ŹäžŻæŢřæ■őåĹřäÿĂäýłåĚČçżĎäÿ■åŐžïijŇåŘęåd'ŰäÿĂçğ■èąÍçd'žè£Źçğ
åŕšåČŘäýŃéÍćè£Źæăůïijž

```
import struct

class StructField:
    '''
    Descriptor representing a simple structure field
    '''
    def __init__(self, format, offset):
        self.format = format
        self.offset = offset
    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            r = struct.unpack_from(self.format, instance._buffer,␣
↪self.offset)
            return r[0] if len(r) == 1 else r

class Structure:
    def __init__(self, bytedata):
        self._buffer = memoryview(bytedata)
```

è£ŹéĞŇæĹŚäžňä¡£çŤÍäžĘäÿĂäýłæŔŘè£řåŹÍæÍěèąÍçd'žærŕŘäýłçżŞæđĎå■ŮæőţïijŇærŕŘäýłæŔŘè£řåŹÍåŇ
å■ŸåĆÍåIJÍåĘĚéČÍçŽĎåĘĚå■ŸçijŞåĘšäÿ■ãĂĆåIJÍ `__get__()` æŰźæşŢäÿ■ïijŇ`struct.unpack_from()` åĞ¡æŢřèćńçŤÍæÍěäžŐçijŞåĘšäÿ■èğčåŇĚäÿĂäýłåĂijïijŇçIJĄåŐžäžĘéćÍåd'ŰçŽĎåĹĘçL'Č

`Structure` çśžåřśæŸŕäÿĂäýłå§žçąĂçśżïijŇæŐěåŔŮå■ŮèŁĆæŢřæ■őåźűå■ŸåĆÍåIJÍåĘĚéČÍçŽĎåĘĚå■
`StructField` æŔŘè£řåŹÍä¡£çŤÍãĂĆ è£ŹéĞŇä¡£çŤÍäžĘ `memoryview()` ïijŇæĹŚäžňäijŽåIJÍåŘŐéÍćèŕęçżĘèőšèğčåőĆæŸŕçŤÍæÍěåźšåŸŻçŽĎãĂĆ

ä¡£çŤÍè£ŹäýłäžčçăĄïijŇä¡ăçŐřåIJÍåřśèČ¡åőŽäźL'äÿĂäýłénŸåśĆæňąçŽĎçżŞæđĎåŕźèśąæÍěèąÍçd'žäýŁéÍć

```
class PolyHeader(Structure):
    file_code = StructField('<i', 0)
    min_x = StructField('<d', 4)
    min_y = StructField('<d', 12)
    max_x = StructField('<d', 20)
    max_y = StructField('<d', 28)
    num_polys = StructField('<i', 36)
```

äÿŃéÍćçŽĎä¿Ńå■ŘåĹl'çŤÍè£ŹäýłçśżæÍèèŕżåŔŰäźŃåL'■æĹŚäžňåĘŹåĚěçŽĎåd'Žè¿źå¡ćæŢřæ■őçŽĎåd't'

```
>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader(f.read(40))
```

```
>>> phead.file_code == 0x1234
True
>>> phead.min_x
0.5
>>> phead.min_y
0.5
>>> phead.max_x
7.0
>>> phead.max_y
9.2
>>> phead.num_polys
3
>>>
```

è£Źäÿłå¿ĹæIJL'èűčïijŇäÿ■è£Ğè£Źçğ■æŰźåijŔè£ŸæŸŕæIJL'äÿĂäžŻçČęäžžçŽĎåIJřæŰźãĂĆéęŰåĚĹïijŇä¡ĘæŸŕè£Źäÿłäžčçăąè£ŸæŸŕæIJL'çĆźèĞČèĆ£ïijŇè£ŸéIJĂèęąä¡£çŤĺèĂĚæŇĞåőŽå¿Ĺåď'ŽåžŢåśĆçŽĎçżĘ StructField ïijŇæŇĞåőŽåĄŔçğžéĞŔç■L')ãĂĆ åŔęåď'ŰïijŇè£ŤåŻđçŽĎçżŞæđIJçśžåŘŇæăůçąőåőđäÿĂä

äžžä¡ŢæŮűåĂŹåŔłèęĄä¡ăéĄĞåĹřäžĘåČŔè£ŹæăůåĘŮä¡ŹçŽĎçśžåőŽäźL'ïijŇä¡ääžŤèŕěèĂČèŹŚäÿŇä¡£çåĚČçśžæIJL'äÿĂäÿłçL'źæĂğåřśæŸŕåőČèČ¡åď'§èćńçŤĺæİěåąńåĚĚèőÿåď'Žä¡ŐåśĆçŽĎåőđçŐřçżĘèŁĆïijŇäžŐäÿŇéİćæĹŚæİěäÿ¿äÿłä¿Ńå■ŘïijŇä¡£çŤĺåĚČçśžçĺ■å¿őæŤźéĂääÿŇæĹŚäžňçŽĎ Structure çśžïijŽ

```
class StructureMeta(type):
    '''
    Metaclass that automatically creates StructField descriptors
    '''
    def __init__(self, clsname, bases, clsdict):
        fields = getattr(self, '_fields_', [])
        byte_order = ''
        offset = 0
        for format, fieldname in fields:
            if format.startswith(('<','>','!','@')):
                byte_order = format[0]
                format = format[1:]
            format = byte_order + format
            setattr(self, fieldname, StructField(format, offset))
            offset += struct.calcsize(format)
        setattr(self, 'struct_size', offset)

class Structure(metaclass=StructureMeta):
    def __init__(self, bytedata):
        self._buffer = bytedata

    @classmethod
    def from_file(cls, f):
        return cls(f.read(cls.struct_size))
```

ä¡£çŤĺæŰřçŽĎ Structure çśžïijŇä¡ääŔŕäžěåČŔäÿŇéİćè£ŹæăůåőŽäźL'äÿĂäÿłçżŞæđĎïijŽ

```
class PolyHeader(Structure):
    _fields_ = [
        ('<i', 'file_code'),
        ('d', 'min_x'),
        ('d', 'min_y'),
        ('d', 'max_x'),
        ('d', 'max_y'),
        ('i', 'num_polys')
    ]
```

æ■čåęĆä¡ăæĽĂèğĄïijŇè£ŹæăůåĘŹåřśçőĂå■Ţåď ŽäžĘãĂĆæĹŚäžňæůżåŁăçŽĎçśżæŰźæşŢ from_file() èőľ æĹŚäžňåIJĺäÿ■éIJĂèęĄç§ěéĄŞäżżä¡ŢæŢřæ■őçŽĎåď ğářŕåŠŇçżŞæđĎçŽĎæČĚåĘţäÿŃă

```
>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader.from_file(f)
>>> phead.file_code == 0x1234
True
>>> phead.min_x
0.5
>>> phead.min_y
0.5
>>> phead.max_x
7.0
>>> phead.max_y
9.2
>>> phead.num_polys
3
>>>
```

äÿĂæŮęä¡ăåijĂåğŃä¡£çŤĺäžĘåĚČçśżïijŇä¡ăåřśåŕŕäżěèőľ åőČåŔŸå¿ŮæŻť åŁăæŹžèČ¡ãĂCä¿ŃåęĆïijŇå äÿŃéÍćæŸŕåŕźåĽ■éÍćåĚČçśżçŽĎäÿĂäÿłåřŔçŽĎæŤźè£ŻïijŇæŔŘä¿ŻäžĘäÿĂäÿłæŰřçŽĎè¿ĚåĽľæŔŔè£řåŹĺa

```
class NestedStruct:
    '''
    Descriptor representing a nested structure
    '''
    def __init__(self, name, struct_type, offset):
        self.name = name
        self.struct_type = struct_type
        self.offset = offset

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            data = instance._buffer[self.offset:
                                    self.offset+self.struct_type.struct_
↪size]
            result = self.struct_type(data)
            # Save resulting structure back on instance to avoid
```

```
            # further recomputation of this step
            setattr(instance, self.name, result)
            return result

class StructureMeta(type):
    '''
    Metaclass that automatically creates StructField descriptors
    '''
    def __init__(self, clsname, bases, clsdict):
        fields = getattr(self, '_fields_', [])
        byte_order = ''
        offset = 0
        for format, fieldname in fields:
            if isinstance(format, StructureMeta):
                setattr(self, fieldname,
                        NestedStruct(fieldname, format, offset))
                offset += format.struct_size
            else:
                if format.startswith(('<','>','!','@')):
                    byte_order = format[0]
                    format = format[1:]
                format = byte_order + format
                setattr(self, fieldname, StructField(format,␣
↪offset))
                offset += struct.calcsize(format)
        setattr(self, 'struct_size', offset)
```

åIJÍè£Źæőţäžčçă Ąäÿ■ïijŇ`NestedStruct` æŔŔè£řåŹÍèćńçŤĺæİěåŔăåŁăåŔęåď ŰäÿĂäÿłåőŽäźĽåIJÍæ§ åőČéÂŽè£ĞåřĘåŐ§åğŃåĘĚå■ŸçijŞåĘšèŽèąŇåĹĞçĽĞæŞ■ä¡IJåŘŐåőđä¿ŃåŇŰçżŹåőŽçŽĎçżŞæđĎçśżåđŇ æĽ Ääżěè£Źçğ■åĹĞçĽĞæŞ■ä¡IJäÿ■äijŽåijŢåŔŚäżżä¡ŢçŽĎéćİåď ŰçŽĎåĘĚå■Ÿåď■åĹűãĂÇçŻÿåŔ■ïijŇåőČ åŔęåď ŰïijŇäÿžäžĘéŸšæ■éĞ■åď■åőđä¿ŃåŇŰïijŇéĂŽè£Ğä¡£çŤĺåŠŇ8.10åřŔèŁĆåŘŇæăůçŽĎæŁĂæIJŕïijŇ

ä¡£çŤĺè£ŹäÿłæŕřçŽĎä£őæ■čçĽĹïijŇä¡ääřśåŔŕäżěåČŔäÿŃéİćè£Źæăůçij ŰåĘŹïijŽ

```
class Point(Structure):
    _fields_ = [
        ('<d', 'x'),
        ('d', 'y')
    ]

class PolyHeader(Structure):
    _fields_ = [
        ('<i', 'file_code'),
        (Point, 'min'), # nested struct
        (Point, 'max'), # nested struct
        ('i', 'num_polys')
    ]
```

äżď äžžæČŁèőűçŽĎæŸŕïijŇåőČäź§èČ¡æŃĽçĚğéćĎæIJ§çŽĎæ■čåÿÿåůěä¡IJïijŇæĹŚäżňåőđéŹĚæŞ■ä¡IJ

```
>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader.from_file(f)
>>> phead.file_code == 0x1234
True
>>> phead.min # Nested structure
<__main__.Point object at 0x1006a48d0>
>>> phead.min.x
0.5
>>> phead.min.y
0.5
>>> phead.max.x
7.0
>>> phead.max.y
9.2
>>> phead.num_polys
3
>>>
```

åĹřçŽőåL'■äÿžæ■ćïijŇäÿĂäÿłåď'ĎçŘĘåőŽéŢ£èőřå¡ŢçŽĎæąĘæđűåůšçżŔåĘŹåě¡äžĘãĂĆä¡ĘæŸŕåęĆæđĽæŕŤåęĆïijŇåď'Žè¿źå¡ćæŰĞäżűåŇĚåŘńåŔŸéŢ£çŽĎéČĺåĹĘãĂĆ

äÿĂçğ■æŰźæąĹæŸŕåĘŹäÿĂäÿłçśżæİěèąÍçď'žå■ŮèŁĆæŢřæ■őïijŇåŘŇæŮűåĘŹäÿĂäÿłåůěåĚůåĞ¡æŢřæ

```
class SizedRecord:
    def __init__(self, bytedata):
        self._buffer = memoryview(bytedata)

    @classmethod
    def from_file(cls, f, size_fmt, includes_size=True):
        sz_nbytes = struct.calcsize(size_fmt)
        sz_bytes = f.read(sz_nbytes)
        sz, = struct.unpack(size_fmt, sz_bytes)
        buf = f.read(sz - includes_size * sz_nbytes)
        return cls(buf)

    def iter_as(self, code):
        if isinstance(code, str):
            s = struct.Struct(code)
            for off in range(0, len(self._buffer), s.size):
                yield s.unpack_from(self._buffer, off)
        elif isinstance(code, StructureMeta):
            size = code.struct_size
            for off in range(0, len(self._buffer), size):
                data = self._buffer[off:off+size]
                yield code(data)
```

çśżæŰźæşŢ `SizedRecord.from_file()` æŸŕäÿĂäÿłåůěåĚůïijŇçŤĺæİěäżŐäÿĂäÿłæŰĞäżűäÿ■èŕżå
è£Źäź§æŸŕå¿Ĺåď'ŽæŰĞäżűæäijåijŔåÿÿçŤĺçŽĎæŰźåijŔãĂĆä¡IJäÿžè¿ŞåĚěïijŇåőČæŐěåŔŮäÿĂäÿłåŇĚåŘńå
åŘŕéĂĽçŽĎ `includes_size` åŘĆæŢřæŇĞåőŽäžĘå■ŮèŁĆæŢřæŸŕåŘęåŇĚåŘńåď't'éČĺåď'ğåřŔãĂĆ
äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘæŢŹä¡äæĂŐæăůä¡£çŤĺäżŐåď'Žè¿źå¡ćæŰĞäżűäÿ■èŕżåŔŰå■ŢçŃńçŽĎåď'Žè¿źå¡ća

```
>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader.from_file(f)
>>> phead.num_polys
3
>>> polydata = [ SizedRecord.from_file(f, '<i')
...              for n in range(phead.num_polys) ]
>>> polydata
[<__main__.SizedRecord object at 0x1006a4d50>,
<__main__.SizedRecord object at 0x1006a4f50>,
<__main__.SizedRecord object at 0x10070da90>]
>>>
```

åŔŕäżěçIJŃåĞžïijŇSizedRecord åőđä¿ŃçŽĎåĘĚåőźè£ŸæšąæIJL'èćńèğčæđŘåĞžæİěãĂĆ åŔŕäżěä¡£çŤĺ iter_as() æŰźæşŢæİěè¿¿åĹřçŻőçŽĎïijŇè£ŹäÿłæŰźæşŢæŐěåŔŮäÿĂäÿłçżŞæđĎæăijåijŔål Structure çśżä¡IJäÿžè¿ŞåĚěãĂĆ è£ŹæăůåŘåŔŕäżěå¿ĹçĄţæť žçŽĎåŐżèğčæđŘæŢřæőïijŇä¿ŃåęĆïijŽ

```
>>> for n, poly in enumerate(polydata):
...     print('Polygon', n)
...     for p in poly.iter_as('<dd'):
...         print(p)
...
Polygon 0
(1.0, 2.5)
(3.5, 4.0)
(2.5, 1.5)
Polygon 1
(7.0, 1.2)
(5.1, 3.0)
(0.5, 7.5)
(0.8, 9.0)
Polygon 2
(3.4, 6.3)
(1.2, 0.5)
(4.6, 9.2)
>>>

>>> for n, poly in enumerate(polydata):
...     print('Polygon', n)
...     for p in poly.iter_as(Point):
...         print(p.x, p.y)
...
Polygon 0
1.0 2.5
3.5 4.0
2.5 1.5
Polygon 1
7.0 1.2
5.1 3.0
0.5 7.5
0.8 9.0
```

```
Polygon 2
3.4 6.3
1.2 0.5
4.6 9.2
>>>
```

åřĘæĽĂæIJĽè£ŹäžŻçżŞåŘĹèţůæÌěïijŇäÿŃéÌćæŸŕäÿĂäÿł read_polys() åĞ¡æŢřçŽĎåŔęåď´ÚäÿĂäÿłä£őæ■čçĽĹïijŽ

```
class Point(Structure):
    _fields_ = [
        ('<d', 'x'),
        ('d', 'y')
    ]

class PolyHeader(Structure):
    _fields_ = [
        ('<i', 'file_code'),
        (Point, 'min'),
        (Point, 'max'),
        ('i', 'num_polys')
    ]

def read_polys(filename):
    polys = []
    with open(filename, 'rb') as f:
        phead = PolyHeader.from_file(f)
        for n in range(phead.num_polys):
            rec = SizedRecord.from_file(f, '<i')
            poly = [ (p.x, p.y) for p in rec.iter_as(Point) ]
            polys.append(poly)
    return polys
```

èőÍèőž

è£ŹäÿĂèŁĆåŘŚä¡ăåśŢçd'žäžĘèőÿåď'ŽénŸçžğçŽĎçijŰçÍŃæŁĂæIJŕïijŇåŇĚæŃňæŔŔè£řåŹÍïijŇåžűè£§è çĎűèĂŇïijŇåőČäžňéČ¡äÿžäžĘåŘŇäÿĂäÿłçĽźåőŽçŽĎçŽőæăĞæIJ■åŁąãĂĆ

äÿŁéÌćçŽĎåőđçŐřçŽĎäÿĂäÿłäÿżèęAçĽźå¿ĄæŸŕåőČæŸŕå§žäžŐæĞŠèğčåŇĚçŽĎæĂÌæČşãĂĆå¡ŞäÿĂä Structure åőđä¿ŃèćńåĹŽåžžæŮűïijŇ __init__() äžĚäžĚåŔłæŸŕåĹŽåžžäÿĂäÿłå■ŮèŁĆæŢřæ■őçŽĎå çĽźåĹńçŽĎïijŇè£ŹæŮűåĂŹåźűæšąæIJĽäżżä¡ŢçŽĎèğčåŇĚæĹŰèĂĚåĚűäžŰäÿŐçżŞæđĎçŽÿåĚşçŽĎæŞ■ä¡ è£ŹæăůåĄŽçŽĎäÿĂäÿłåŁÍæIJžæŸŕä¡ăåŔŕèČ¡äžĚäžĚåŔłåŕźäÿĂäÿłå■ŮèŁĆèőřå¡ŢçŽĎæ§ŘäÿĂåřŔéČĺåĹĘæ

äÿžäžĘåőđçŐřæĞŠèğčåŇĚåŠŇæĽŞåŇĚïijŇéIJĂèęAä¡£çŤÍ StructField æŔŔè£řåŹÍçśżãĂĆ çŤÍæĹůåIJÍ _fields_ äÿ■åĹŮåĞžæÌěçŽĎæŕŔäÿłåśđæĂğéČ¡äijŽèćńè¡ňåŇŰæĹŘäÿĂäÿł StructField æŔŔè£řåŹÍïijŇ åőČåřĘçŽÿåĚşçżŞæđĎæăijåijŔçăĄåŠŇåĄŔçğżåĂijä£Ìå■ŸåĹřå■ŸåĆÍçijŞå■Ÿäÿ■ãĂĆåĚČçś StructureMeta åIJĺåď'Žäÿłçż§æđĎçśèćńåőŽäźĽæŮűèĞłåŁÍåĹŽåžžäžĘè£ŹäžŻæŔŔè£řåŹÍãĂĆ æĹŚäžňä¡£çŤÍåĚČçśżçŽĎäÿĂäÿłäÿżèęĄåŐ§åŽăæŸŕåőČä¡£å¿ŮçŤÍæĹůéÌđåÿÿæŰźä¿£çŽĎéĂŽè£ĞäÿĂäÿłér

StructureMeta çŽĎäÿĂäÿłå¿Ĺå¿őåęŹçŽĎĎåIJřæŰźåřśæŸŕåőČäijŽåŽžåőŽå■ŮèŁĆæŢřæ■őéąžåžŔãĂ
äź§åřśæŸŕèŕť'ïijŇåęĆæđIJäżżæĎŔçŽĎåśđæĂğæŇĞåőŽäžĘäÿĂäÿłå■ŮèŁĆéąžåžŔ(<èąĺçd'žä¡Öä¡■äijŸåĚĹ
æĹŰèĂĚ >èąĺçd'žéńŸä¡■äijŸåĚĹ)ïijŇ éĆčåŘŐéİćæL'ĂæIJL'å■Ůæőţç ŽĎéąžåžŔéČ¡äžèè£ŹäÿłéąžåžŔäÿžåĞE
ærŤåęĆïijŇä¡ăåŔŕèČ¡æIJL'äÿĂäžŻærŤè¿Čåd'■æİĆçŽĎçżŞæđĎïijŇåřśåČŔäÿŇéİćè£ŹæăûïijŽ

```
class ShapeFile(Structure):
    _fields_ = [ ('>i', 'file_code'), # Big endian
        ('20s', 'unused'),
        ('i', 'file_length'),
        ('<i', 'version'), # Little endian
        ('i', 'shape_type'),
        ('d', 'min_x'),
        ('d', 'min_y'),
        ('d', 'max_x'),
        ('d', 'max_y'),
        ('d', 'min_z'),
        ('d', 'max_z'),
        ('d', 'min_m'),
        ('d', 'max_m') ]
```

äźŃåL'■æĹŚäžňæŔŘåĹřè£ĞïijŇmemoryview() çŽĎä¡£çŤĺåŔŕäžěåÿőåŁl'æĹŚäžňéĄ£åĚ■åĘĚå■ŸçŽĎ
å¡ŞçżŞæđĎå■ŸåIJĺåţŇåěŮçŽĎæŮűåĂŹïijŇmemoryviews åŔŕäžěåŔäåŁäåŘŇäÿĂåĘĚå■ŸåŇžå§§äÿŁåőŽä
è£ŹäÿłçL'źæĂğæŕŤè¿Čå¿őåęŹïijŇä¡ĘæŸŕåőČåĚşæşĺçŽĎæŸŕåĘĚå■ŸèğĘåŻ¿äÿŐæŹőéĂŽå■ŮèŁĆæŢřçżĎç
åęĆæđIJä¡äåIJĺäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšæĹŰå■ŮèŁĆæŢřçżĎäÿŁæL'ğèąŇåĹĞçL'ĞæŞ■ä¡IJïijŇä¡äéĂŽåÿÿä
èĂŇåĘĚå■ŸèğĘåŻ¿åĹĞçL'Ğäÿ■æŸŕè£ŹæăûçŽĎïijŇåőČäžĚäžĚæŸŕåIJĺåůšå■ŸåIJĺçŽĎåĘĚå■ŸäÿŁéİćåŔäă

è£ŸæIJL'å¿Ĺåd'ŽçŻÿåĚşçŽĎçńäèŁĆåŔŕäžěåÿőåŁl'æĹŚäžňæL'l'åśŢè£ŹéĞŇèőĺèőžçŽĎæŰźæąĹãĂĆ
åŘĆèĂČ8.13åřŔèŁĆä¡£çŤĺæŔŔè£řåŹĺæđĎåžžäÿĂäÿłçśžåđŃçşżçż§ãĂĆ
8.10åřŔèŁĆæIJL'æŻt'åd'ŽåĚşäžŐåžűè£§èőąçőŮåśđæĂğåĂijçŽĎèőĺèőžïijŇåźűäÿŤèů§NestedStructæŔŔè£řå
9.19åřŔèŁĆæIJL'äÿĂäÿłä¡£çŤĺåĚČçśżæİěåĹİåğŃåŇŰçśżæĹŘåŚŸçŽĎä¿Ńå■ŘïijŇåŠŇ
StructureMeta çśżéİđåÿÿçŻÿäijijãĂĆ PythonçŽĎ ctypes
æžŘçăĄåŘŇæăûäź§å¿ĹæIJL'èűčïijŇåőČæŔŔä¿ŻäžĘåŕźåőŽäźL'æŢřæ■őçżŞæđĎãĂĄæŢřæ■őçżŞæđĎåţŇåěŮ

6.13 æŢřæ■őçŽĎçť'ŕåŁääÿŐçż§èőąæŞ■ä¡IJ

éŮőécŸ

ä¡ăéIJĂèęĄåd'ĎçŘĘäÿĂäÿłå¿Ĺåd'ğçŽĎæŢřæ■őéŻĘåźűéIJĂèęĄèőąçőŮæŢřæ■őæĂżåŠŇæĹŰåĚűäžŰçż

èğčåĘşæŰźæąĹ

åŕźäžŐäżżä¡ŢæűL'åŘŁåĹřçż§èőąãĂĄæŮűéŮť'äžŔåĹŮäžěåŘŁåĚűäžŰçŻÿåĚşæŁĂæIJŕçŽĎæŢřæ■őåĹĘ
PandasäžŞ ãĂĆ

äÿžäžĘèőľ'ä¡ăåĚĹä¡ŞéłŇäÿŃïijŇäÿŃéİćæŸŕäÿĂäÿłä¡£çŤĺPandasæİěåĹĘæđŘèŁİåŁăåŞěå§ŐåÿĆçŽĎ
èĂĄéijăåŠŇåŢőé¡£çśżåŁĺçL'l'æŢřæ■őäžŞ çŽĎä¿Ńå■ŘãĂĆ åIJĺæĹŚåĘŹè£ŹçŕĞæŰĞçńäçŽĎæŮűåĂŹïijŇè£Z

```
>>> import pandas

>>> # Read a CSV file, skipping last line
>>> rats = pandas.read_csv('rats.csv', skip_footer=1)
>>> rats
<class 'pandas.core.frame.DataFrame'>
Int64Index: 74055 entries, 0 to 74054
Data columns:
Creation Date 74055 non-null values
Status 74055 non-null values
Completion Date 72154 non-null values
Service Request Number 74055 non-null values
Type of Service Request 74055 non-null values
Number of Premises Baited 65804 non-null values
Number of Premises with Garbage 65600 non-null values
Number of Premises with Rats 65752 non-null values
Current Activity 66041 non-null values
Most Recent Action 66023 non-null values
Street Address 74055 non-null values
ZIP Code 73584 non-null values
X Coordinate 74043 non-null values
Y Coordinate 74043 non-null values
Ward 74044 non-null values
Police District 74044 non-null values
Community Area 74044 non-null values
Latitude 74043 non-null values
Longitude 74043 non-null values
Location 74043 non-null values
dtypes: float64(11), object(9)

>>> # Investigate range of values for a certain field
>>> rats['Current Activity'].unique()
array([nan, Dispatch Crew, Request Sanitation Inspector],␣
 ↪dtype=object)
>>> # Filter the data
>>> crew_dispatched = rats[rats['Current Activity'] == 'Dispatch␣
 ↪Crew']
>>> len(crew_dispatched)
65676
>>>

>>> # Find 10 most rat-infested ZIP codes in Chicago
>>> crew_dispatched['ZIP Code'].value_counts()[:10]
60647 3837
60618 3530
60614 3284
60629 3251
60636 2801
60657 2465
60641 2238
```

```
60609 2206
60651 2152
60632 2071
>>>

>>> # Group by completion date
>>> dates = crew_dispatched.groupby('Completion Date')
<pandas.core.groupby.DataFrameGroupBy object at 0x10d0a2a10>
>>> len(dates)
472
>>>

>>> # Determine counts on each day
>>> date_counts = dates.size()
>>> date_counts[0:10]
Completion Date
01/03/2011 4
01/03/2012 125
01/04/2011 54
01/04/2012 38
01/05/2011 78
01/05/2012 100
01/06/2011 100
01/06/2012 58
01/07/2011 1
01/09/2012 12
>>>

>>> # Sort the counts
>>> date_counts.sort()
>>> date_counts[-10:]
Completion Date
10/12/2012 313
10/21/2011 314
09/20/2011 316
10/26/2011 319
02/22/2011 325
10/26/2012 333
03/17/2011 336
10/13/2011 378
10/14/2011 391
10/07/2011 457
>>>
```

åŮřïijŇçIJŃæăůå■Ř2011åźť10æIJĹ7æŮěářźèĂĄéijääżňæİěèřť'æŸŕäÿłå¿Ĺå£ŹçćŇçŽĎæŮěå■ŘåŢŁïijĄ

èőĺèőž

PandasæŸŕäÿĂäÿłæŃěæIJL'å¿Ĺåď'ŽçĽźæĂğçŽĎåď'ğåđŃåĞ¡æŢřåžŞïijŇæĹŚåIJĺè£ŹéĞŇäÿ■åŔŕèČ¡äżŃ
ä¡ĘæŸŕåŔłèęĄä¡ăéIJĂèęĄåŐżåĹĘæđŘåď'ğåđŃæŢřæ■őéŻĘåŘĹãĂĄåŕźæŢřæ■őåĹĘçżĎãĂĄèőąçőŮåŘĎçğ■

CHAPTER 9

çňňäÿČçńäïijŽåĞ¡æŢř

ä¡£çŤÍ `def` èí■åŔěåőŽäźĽ'åĞ¡æŢřæŸŕæĽ'ĂæIJL'çÍŃåžŔçŽĎå§žçąĂãĂĆ
æIJňçńăçŽĎçŻőæăĞæŸŕèőšèğčäÿĂäžŻæŻť'åŁăéńŸçžğåŠŇäÿ■åÿÿèğĄçŽĎåĞ¡æŢřåőŽäźĽ'äÿŐä¡£çŤÍæÍąåijŔ
æűĽ'åŔŁåĹřçŽĎåĘĚåőźåŇĚæŃňéżŸèőď'åŔĆæŢřãĂĄäżżæĎŔæŢřéĞŔåŔĆæŢřãĂĄåijžåĹűåĚşéŤőå■ŮåŔĆ
åŔęåď'ŰïijŇäÿĂäžŻéńŸçžğçŽĎæŐğåĹűæţĄåŠŇåĽ'çŤÍåŻđèřČåĞ¡æŢřäijăéĂŠæŢřæ■őçŽĎæŁĂæIJŕåIJÍè£Ź

Contents:

7.1 åŔŕæŐěåŔŮäżżæĎŔæŢřéĞŔåŔĆæŢřçŽĎåĞ¡æŢř

éŮőécŸ

ä¡ăæČşæđĎéĂääÿĂäÿłåŔŕæŐěåŔŮäżżæĎŔæŢřéĞŔåŔĆæŢřçŽĎåĞ¡æŢřãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘèČ¡èőĺ'äÿĂäÿłåĞ¡æŢřæŐěåŔŮäżżæĎŔæŢřéĞŔçŽĎä¡■ç¡őåŔĆæŢřïijŇåŔŕäžěä¡£çŤÍäÿĂäÿł*åŔĆ

```
def avg(first, *rest):
    return (first + sum(rest)) / (1 + len(rest))

# Sample use
avg(1, 2) # 1.5
avg(1, 2, 3, 4) # 2.5
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇrestæŸŕçŤśæĽ'ĂæIJL'åĚűäżŰä¡■ç¡őåŔĆæŢřçżĎæĹŘçŽĎåĚČçżĎãĂĆçĎűåŘĆ

äÿžäžĘæŐěåŔŮäżżæĎŔæŢřéĞŔçŽĎåĚşéŤőå■ŮåŔĆæŢřïijŇä¡£çŤÍäÿĂäÿłäžě**åijĂåď'ť'çŽĎåŔĆæŢřã

```
import html

def make_element(name, value, **attrs):
    keyvals = [' %s="%s"' % item for item in attrs.items()]
    attr_str = ''.join(keyvals)
    element = '<{name}{attrs}>{value}</{name}>'.format(
                name=name,
                attrs=attr_str,
                value=html.escape(value))
    return element

# Example
# Creates '<item size="large" quantity="6">Albatross</item>'
make_element('item', 'Albatross', size='large', quantity=6)

# Creates '<p>&lt;spam&gt;</p>'
make_element('p', '<spam>')
```

åIJÍè£ŹéĞŃïijŇattrsæŸŕäÿĂäÿłåŇĚåŘńæĽĂæIJL'èćńäijäåĚèè£ŻæİęçŽĎåĚşéŤőå■ŮåŔĆæŢřçŽĎå■ŮåĚ

åęĆæđIJä¡ăè£ŸåÿŇæIJŻæ§ŘäÿłåĞ¡æŢřèČ¡åŘŇæŮűæŐěåŔŮäżżæĎŔæŢřéĞŔçŽĎä¡■ç¡őåŔĆæŢřåŠŇål

```
def anyargs(*args, **kwargs):
    print(args) # A tuple
    print(kwargs) # A dict
```

ä¡£çŤÍè£ŹäÿłåĞ¡æŢřæŮűïijŇæĽĂæIJL'ä¡■ç¡őåŔĆæŢřäijŽèćńæŤ¿åĹřargsåĚČçżĎäÿ■ïijŇæĽĂæIJL'åĚş

èőÍèőž

äÿĂäÿł*åŔĆæŢřåŔłèČ¡åĞžçŐřåIJÍåĞ¡æŢřåőŽäźL'äÿ■æIJĂåŘŐäÿĂäÿłä¡■ç¡őåŔĆæŢřåŘŐéİćïijŇèĂŇ
**åŔĆæŢřåŔłèČ¡åĞžçŐřåIJÍæIJĂåŘŐäÿĂäÿłåŔĆæŢřãĆ æIJL'äÿĂçĆźèęĄæşĺæĎŔçŽĎæŸŕïijŇåIJÍ*åŔĆæ

```
def a(x, *args, y):
    pass

def b(x, *args, y, **kwargs):
    pass
```

è£Źçğ■åŔĆæŢřåřśæŸŕæĹŠäžňæĽĂèŕť'çŽĎåijžåĹűåĚşéŤőå■ŮåŔĆæŢřïijŇåIJÍåŘŐéİć7.2åřŔèŁĆè£Ÿäij

7.2 åŕłæŐěåŔŮåĚşéŤőå■ŮåŔĆæŢřçŽĎåĞ¡æŢř

éŮőécŸ

ä¡ăåÿŇæIJŻåĞ¡æŢřçŽĎæ§ŘäžŻåŔĆæŢřåijžåĹűä¡£çŤÍåĚşéŤőå■ŮåŔĆæŢřäijăéĂŠ

èğčåĘşæŰźæąĹ

åřĘåijžåĹűåĚşéŤőå■ŮåŘĆæŢřæŤ¿åĹřæ§Řäÿł*åŘĆæŢřæĹŰèĂĚå■Ţäÿł*åŘŐéİćåřśèČ¡è¿¿åĹřè£Źçğ■æ

```
def recv(maxsize, *, block):
    'Receives a message'
    pass

recv(1024, True) # TypeError
recv(1024, block=True) # Ok
```

åĹľçŤĺè£Źçğ■æŁĂæIJřïijŇæĹŚäžňè£ŸèČ¡åIJĺæŐěåŘŮäżżæĎŘåď˙Žäÿłä¡■ç¡őåŘĆæŢřçŽĎåĞ¡æŢřäÿ■æ

```
def mininum(*values, clip=None):
    m = min(values)
    if clip is not None:
        m = clip if clip > m else m
    return m

minimum(1, 5, 2, -5, 10) # Returns -5
minimum(1, 5, 2, -5, 10, clip=0) # Returns 0
```

èőĺèőž

å¿Ĺåď˙ŽæČĚåĘţäÿŃïijŇä¡£çŤĺåijžåĹűåĚşéŤőå■ŮåŘĆæŢřäijŽæŕŤä¡£çŤĺä¡■ç¡őåŘĆæŢřèąĺæĎŘæŻť˙åŁä
ä¿ŃåęĆïijŇèĂČèŹŚäÿŃåęĆäÿŃäÿĂäÿłåĞ¡æŢřèřČçŤĺïijŽ

```
msg = recv(1024, False)
```

åęĆæđIJèřČçŤĺèĂĚåŕźrecvåĞ¡æŢřåźűäÿ■æŸŕå¿ĹçĘ§æĆL˙ïijŇéĆčäžŰèĆŕåőŽäÿ■æŸŐçŹ¡éĆčäÿłFalseå
ä¡ĘæŸŕïijŇåęĆæđIJäžčçăĄåŘŸæĹŘäÿŃéİćè£Źæăůå■ŘçŽĎèŕİåřśæÿĚæěŽåď˙ŽäžĘïijŽ

```
msg = recv(1024, block=False)
```

åŘęåď˙ŰïijŇä¡£çŤĺåijžåĹűåĚşéŤőå■ŮåŘĆæŢřäź§äijŽæŕŤä¡£çŤĺ**kwargsåŘĆæŢřæŻť˙åě¡ïijŇåŽääÿžåIJ

```
>>> help(recv)
Help on function recv in module __main__:
recv(maxsize, *, block)
    Receives a message
```

åijžåĹűåĚşéŤőå■ŮåŘĆæŢřåIJĺäÿĂäžŽæŻť˙énŸçžğåIJžåŘĹåŘŇæăůäź§å¿ĹæIJL˙çŤĺãĆĆ
ä¿ŃåęĆïijŇåőČäžňåŔŕäżěèćńçŤĺæİěåIJĺä¡£çŤĺ*argsåŠŇ**kwargsåŘĆæŢřä¡IJäÿžè¿ŞåĚěçŽĎåĞ¡æŢřäÿ■æŘŠ

7.3 çžŹåĞ¡æŢřåŔĆæŢřåćđåŁääĚČä£qæĄŕ

éŮőécŸ

ä¡ääĘŹåě¡äžĘäÿĂäÿłåĞ¡æŢřijŇçĎűåŘŐæČşäÿžè£ŹäÿłåĞ¡æŢřçŽĎåŔĆæŢřåćđåŁääÿĂäžŽécÍåd'ŰçŽĎ

èğčåĘşæŰźæąĹ

ä¡£çŤĺåĞ¡æŢřåŔĆæŢřæşĺèğčæŸŕäÿĂäÿłå¿Ĺåě¡çŽĎåŁđæşŢijŇåőČèČ¡æŔŘçd'žçĺŇåžŔåŚŸåžŤèŕěæĂŐ
ä¿ŇåęĆijŇäÿŇéÍćæIJL'äÿĂäÿłècńæşĺèğčäžĘçŽĎåĞ¡æŢřijŽ

```
def add(x:int, y:int) -> int:
    return x + y
```

pythonèğčéĞŁåŹĺäÿ■äijŽåŕźè£ŹäžŽæşĺèğčæůžåŁääżä¡ŢçŽĎèŕ■äźL'ãĂĆåőČäžňäÿ■äijŽècńçśžåđŇæčĂ
çĎűèĂŇïijŇåŕźäžŐéĆčäžŽéŸĚèŕżæžŘçăĄçŽĎäžžæÍěèőšåřšå¿ĹæIJL'åÿőåŁl'åŢęãĂĆçňňäÿL'æŰźåůěåĚůåŠŇ

```
>>> help(add)
Help on function add in module __main__:
add(x: int, y: int) -> int
>>>
```

åř¡çőąä¡ääŔŕäžěä¡£çŤĺäżżæĎŘçśžåđŇçŽĎåŕźèśąçżŹåĞ¡æŢřæůžåŁäæşĺèğč(ä¿ŇåęĆæŢřå■ŰïijŇå■Ůçňęä

èőÍèőž

åĞ¡æŢřæşĺèğčåŔłå■ŸåĆĺåIJĺåĞ¡æŢřçŽĎ `__annotations__`
åśđæĂğäÿ■ãĂĆä¿ŇåęĆijŽ

```
>>> add.__annotations__
{'y': <class 'int'>, 'return': <class 'int'>, 'x': <class 'int'>}
```

åř¡çőąæşĺèğčçŽĎä¡£çŤĺæŰźæşŢåŔŕèČ¡æIJL'å¿Ĺåd'Žçğ■ïijŇä¡ĘæŸŕåőČäžňçŽĎäÿżèęĄçŤĺéĂŤè£ŸæŸŕ
åŽääÿžpythonåźűæšąæIJL'çśžåđŇåčřæŸŐïijŇéĂŽåÿÿæÍěèőšäżĚäžĚéĂŽè£ĞéŸĚèŕżæžŘçăĄå¿ĹéŽ¿ç§ěéĄŞå
è£ŹæŮűåĂŹä¡£çŤĺæşĺèğčåŕśèČ¡çžŹçĺŇåžŔåŚŸæŽt'åd'ŽçŽĎæŔŘçd'žïijŇèől'äžŰäžňåŔŕäžěäžL'åŘŰçŽĎä¡£

åŔĆèĂČ9.20åŕŔèŁĆçŽĎäÿĂäÿłæŽt'åŁäéńŸçžğçŽĎä¿Ňå■ŘïijŇæijŤçd'žäžĘåęĆä¡ŢåĹl'çŤĺæşĺèğčæÍěåć

7.4 è£ŤåŻđåd'ŽäÿłåĂijçŽĎåĞ¡æŢř

éŮőécŸ

ä¡ääÿŇæIJŻædĎéĂääÿĂäÿłåŔŕäžěè£ŤåŻđåd'ŽäÿłåĂijçŽĎåĞ¡æŢř

èğčåĘşæŰźæąĹ

äÿžäžĘèČ¡è£ŤåŻđåď'Žäÿ łåĂijïijŇåĞ¡æŢřçŻť'æŐěreturnäÿĂäÿłåĚČçżĎářšèąŇäžĘãĂĆä¿ŃåęĆïijŽ

```
>>> def myfun():
... return 1, 2, 3
...
>>> a, b, c = myfun()
>>> a
1
>>> b
2
>>> c
3
```

èőĺèőž

åř¡çőąmyfun()çIJŃäÿŁåŐżè£ŤåŻđäžĘåd'Žäÿ łåĂijïijŇåőđéŹĚäÿŁæŸŕåĚĹåĹŻåžžäžĘäÿĂäÿłåĚČçżĎçĎu
è£Źäÿłèŕ■æşŢçIJŃäÿŁåŐżæŕŤè¿ČåěĞæĂłïijŇåőđéŹĚäÿŁæĹŚäžňä¡£çŤĺçŽĎæŸŕéĂŮåŘůæİěçŤ§æĹŘäÿĂäÿ

```
>>> a = (1, 2) # With parentheses
>>> a
(1, 2)
>>> b = 1, 2 # Without parentheses
>>> b
(1, 2)
>>>
```

å¡ŞæĹŚäžňèřČçŤĺè£ŤåŻđäÿĂäÿłåĚČçżĎçŽĎåĞ¡æŢřçŽĎæŮűåĂŹ
ïijŇéĂŽåÿÿæĹŚäžňäijŽářĘçżŞæđIJèţŃåĂijçżŹåď'Žäÿ łåŔŸéĞŔïijŇářšåČŔäÿŁéİćçŽĎéĆčæăůãĂĆ
åĚűåőđè£ŹářšæŸŕ1.1ářŔèŁĆäÿ■æĹŚäžňæL'Ăèŕť'çŽĎåĚČçżĎèğčåŇĚãĂĆè£ŤåŻđçżŞæđIJäź§åŔŕäžěèţŃåĂij
è£ŹæŮűåĂŹè£ŹäÿłåŔŸéĞŔåĂijåřšæŸŕåĞ¡æŢřè£ŤåŻđçŽĎéĆčäÿłåĚČçżĎæIJňèžňäžĘïijŽ

```
>>> x = myfun()
>>> x
(1, 2, 3)
>>>
```

7.5 åőŽäźĽælJL'éżŸèőď'åŔĆæŢřçŽĎåĞ¡æŢř

éŮőécŸ

ä¡ăæČşåőŽäźL'äÿĂäÿłåĞ¡æŢřæĹŰèĂĚæŰźæşŢïijŇåőČçŽĎäÿĂäÿłæĹŰåď'Žäÿ łåŔĆæŢřæŸŕåŔŕéĂL'çŽ

èğčåĘşæŰźæąĹ

åőŽäźĽäÿĂäÿłæIJL'åŔŕéĂĽ'åŔĆæŢřçŽĎåĞ¡æŢřæŸŕéİđåÿÿçőĂå■ŢçŽĎïijŇçŻt'æŐěåIJÍåĞ¡æŢřåőŽäźĽ

```
def spam(a, b=42):
    print(a, b)

spam(1) # Ok. a=1, b=42
spam(1, 2) # Ok. a=1, b=2
```

åęĆæđIJéžŸèőd'åŔĆæŢřæŸŕäÿĂäÿłåŔŕä£őæŤźçŽĎåőźåŹÍærŤåęĆäÿĂäÿłåĹŮèąÍãĂĄéŻĘåŘĹæĹŰèĂĚ

```
# Using a list as a default value
def spam(a, b=None):
    if b is None:
        b = []
    ...
```

åęĆæđIJä¡ăåźűäÿ■æČşæŔŘä¿ŻäÿĂäÿłéžŸèőd'åĂijïijŇèĂŇæŸŕæČşäžĔäžĔæţŃèŕŢäÿŃæ§ŘäÿłéžŸèőd'å

```
_no_value = object()

def spam(a, b=_no_value):
    if b is _no_value:
        print('No b value supplied')
    ...
```

æĹŚäžňæţŃèŕŢäÿŃè£ŹäÿłåĞ¡æŢřijž

```
>>> spam(1)
No b value supplied
>>> spam(1, 2) # b = 2
>>> spam(1, None) # b = None
>>>
```

äžŤçżĘèğĆåŕ§åŔŕäžěåŘŚçŐřåĹřäijăéĂŠäÿĂäÿłNoneåĂijåŠŇäÿ■äijăåĂijäÿd'çğ■æČĚåĘţæŸŕæIJL'åůőåĚ

èőÍèőž

åőŽäźĽåÿęéžŸèőd'åĂijåŔĆæŢřçŽĎåĞ¡æŢřæŸŕå¿ĹçőĂå■ŢçŽĎïijŇä¡EçżÍäÿ■äžĔäžĔåŔłæŸŕè£ŹäÿłïijŇ

éęŰåĚĹïijŇéžŸèőd'åŔĆæŢřçŽĎåĂijäžĔäžĔåIJÍåĞ¡æŢřåőŽäźĽçŽĎæŮűåĂŹèţŃåĂijäÿĂæňąãĂĆèŕŢçĺ

```
>>> x = 42
>>> def spam(a, b=x):
...     print(a, b)
...
>>> spam(1)
1 42
>>> x = 23 # Has no effect
```

```
>>> spam(1)
1 42
>>>
```

æşĺæĎŕåĹřå¡ŞæĹŚäžňæŤźåŔŸxçŽĎåĂijçŽĎæŮűåĂŹåŕźéŻŸèőď'åŔĆæŢřåĂijåźűæšąæIJL'å¡śåŞ■ïijŇè

åĚűæňąïijŇéżŸèőď'åŔĆæŢřçŽĎåĂijåžŤèŕěæŸŕäÿ■åŔŕåŔŸçŽĎåŕźèśąïijŇæŕŤåęĆNoneãĂĄTrueãĂĄFa
çĽźåĹńçŽĎïijŇå■ČäÿĞäÿ■èęĄåČŔäÿŃéİćè£Źæăůåĕ́Źäżčçă Ąïij Ž

```
def spam(a, b=[]): # NO!
    ...
```

åęĆæđIJä¡ăè£ŹäźĹåĄŽäžĘïijŇå¡ŞéżŸèőď'åĂijåIJĺåĚűäżŮåIJřæŰźèćńä£őæŤźåŘŐä¡ăářĘäijŽéĄĞåĹřåŘ

```
>>> def spam(a, b=[]):
...     print(b)
...     return b
...
>>> x = spam(1)
>>> x
[]
>>> x.append(99)
>>> x.append('Yow!')
>>> x
[99, 'Yow!']
>>> spam(1) # Modified list gets returned!
[99, 'Yow!']
>>>
```

è£Źçğ■çżŞæđIJåžŤèŕěäÿ■æŸŕä¡ăæČşèęĄçŽĎãĂĆäÿžäžĘéĄ£åĚ■è£Źçğ■æČĚåĘţçŽĎåŔŞçŤ§ïijŇæIJĂå
çĎűåŘŐåIJĺåĞ¡æŢřéĞŇéİćæčĂæ§ěåőČïijŇåL'■éİćçŽĎä¿Ńå■Řářśæ Ÿŕè£Źæăůå ĄŽçŽĎãĂĆ

åIJĺæţŃèŕŢNoneåĂijæŮűä¡£çŤĺ is æŞ■ä¡IJçňęæŸŕå¿ĹéĞ■èęĄçŽĎïijŇäź§æŸŕè£Źçğ■æŰźæąĹçŽĎåĚş
æIJL'æŮűåĂŹåd'ğåőűäijŽçŁŕäÿŃäÿŃéİćè£Źæăůç ŽĎéŤŹèŕřïijŽ

```
def spam(a, b=None):
    if not b: # NO! Use 'b is None' instead
        b = []
    ...
```

è£ŹäźĹåĘŹçŽĎéŮőécŸåIJĺäžŐår¡çőąNoneåĂijçąőåődæŸŕècńå¡ŞæĹŘFalseïijŇ
ä¡ĘæŸŕè£ŸæIJL'åĚűäżŮçŽĎårźèśą(æŕŤåęĆéŢ£åžęäÿž0çŽĎå■Ůçňęäÿšã ĂĄåĹŮèąĺãĂĄåĚČçżĎãĂĄå■ŮåĚÿ
åŻăæ■d'ïijŇäÿŁéİćçŽĎäžčçăĄäijŽèŕŕåŕĘäÿĂäžŻåĚűäżŮè¿ŞåĚěäź§å¡ŞæĹŘæŸŕæšąæIJL'è¿ŞåĚěãĂĆæŕŤåęĆ

```
>>> spam(1) # OK
>>> x = []
>>> spam(1, x) # Silent error. x value overwritten by default
>>> spam(1, 0) # Silent error. 0 ignored
>>> spam(1, '') # Silent error. '' ignored
>>>
```

æIJĂåŘŐäŷĂäŷłéŮőécŸæŕŤè¿Čå¿őåęŹïijŇéĆčåřśæŸŕäŷĂäŷłåĞ¡æŢřéIJĂèęĄæţŃèŕŢæ§ŘäŷłåŕŕéĂĽ'åŕ
è£ŹæŮűåĂŹéIJĂèęĄåřŕå£ČçŽĎæŸŕä¡ääŷ■èČ¡çŤĺæ§Řäŷłéżŷèőď'åĂijæŕŤåęĆNoneãĂĄ
0æĹŰèĂĚFalseåĂijæİěæţŃèŕŢçŤĺæĹůæŔŘä¿ŻçŽĎåĂij(åŻăäŷžè£ŹäžŻåĂijéČ¡æŸŕåŘĹæşŢçŽĎåĂijïijŇæŸ
åŻăæ■ď'ïijŇä¡äéIJĂèęĄåĚűäžŰçŽĎèğčåEşæŰźæąĹäžĘãĂĆ

äŷžäžĘèğčåEşè£ŹäŷłéŮőécŸïijŇä¡ääŔŕäżěåĹŻåżžäŷĂäŷłçŃňäŷĂæŮăäžŇçŽĎçğĄæIJL'åŕźèśąåőđä¿Ńïij
åIJĺåĞ¡æŢřéĞŇéÍćïijŇä¡ääŔŕäżěéĂŽè£ĞæčÄæ§ěèćńäijäéĂŠåŔĆæŢřåĂijèů§è£Źäŷłåőđä¿ŃæŸŕåĘäŷĂæăůa
è£ŹéĞŇçŽĎæÄİèůŕæŸŕçŤĺæĹůäŷ■åŔŕèČ¡åŐżäijäéĂŠè£Źäŷł_no_valueåőđä¿Ńä¡IJäŷžè¿ŞåĚěãĂĆ
åŻăæ■ď'ïijŇè£ŹéĞŇéĂŽè£ĞæčÄæ§ěè£ŹäŷłåÄijåŕśèČ¡çąőåőŽæ§Řäŷłå ŔĆæŢřæŸŕåĘèćńäijäéĂŠè£ŽæİěäžE

è£ŹéĞŇåŕź `object()` çŽĎä¡£çŤĺçIJŃäŷŁåŐżæIJL'çĆźäŷ■åď'łåŷŷèğĄãĂĆ`object`
æŸŕpythonäŷ■æL'ĂæIJL'çśžçŽĎå§žçśžãĂĆ ä¡ääŔŕäżěåĹŻåżž `object`
çśžçŽĎåőđä¿ŃïijŇä¡EæŸŕè£ŹäžŻåőđä¿ŃæšąäžĂäźĹåőđéŹĚçŤĺåď'ĎïijŇåŻääŷžåőČåźűæšąæIJL'äżżä¡ŢæIJL'
äź§æšąæIJL'äżżä¡Ţåőđä¿ŃæŢřæ■ő(åŻăäŷžåőČæšąæIJL'äżżä¡ŢçŽĎåőđä¿Ńå■ŮåĚŷïijŇä¡ăçŤŽèĞşéČ¡äŷ■èČ¡è
ä¡ääŤŕäŷĂèČ¡åĄŽçŽĎåŕśæŸŕæţŃèŕŢåŘŇäŷĂæĂğãĂĆè£ŹäŷłåĹŽåě¡çňęåŔĹæĹŚçŽĎèęĄæśĆïijŇåŻääŷžæĹ

7.6 åőŽäźĽåŇ£åŘ■æĹŰåĘĚèĄŤåĞ¡æŢř

éŮőécŸ

ä¡ăæČşäŷž `sort()` æŞ■ä¡IJåĹŻåżžäŷĂäŷłå¿Ĺç§■çŽĎåŻđèřČåĞ¡æŢřïijŇä¡EåŔĹäŷ■æČşçŤĺ
`def` åŐżåĘŹäŷĂäŷłå■ŢèąŇåĞ¡æŢřïijŃ èĂŇæŸŕåŷŇæIJŻéĂŽè£Ğæ§Řäŷłå£ńæ■ůæŰźåijŔäžěåĘĚèĄŤæŰźåi

èğčåEşæŰźæąĹ

å¡ŞäŷĂäžŻåĞ¡æŢřå¿ĹçőĂå■ŢïijŇäżĚäżĚåŔłæŸŕèőąçőŮäŷĂäŷłèąĺè¿¿åijŔçŽĎåĂijçŽĎæŮűåĂŹïijŇåřś

```
>>> add = lambda x, y: x + y
>>> add(2,3)
5
>>> add('hello', 'world')
'helloworld'
>>>
```

è£ŹéĞŇä¡£çŤĺçŽĎlambdaèąĺè¿¿åijŔèů§äŷŇéÍćçŽĎæŢĹædIJæŸŕäŷĂæăůçŽĎïijŽ

```
>>> def add(x, y):
...     return x + y
...
>>> add(2,3)
5
>>>
```

lambdaèąĺè¿¿åijŔåĚŷådŇçŽĎä¡£çŤĺåIJžæŹŕæŸŕæŐŠåžŔæĹŰæŢřæ■őreduceç■L'ïijŽ

```
>>> names = ['David Beazley', 'Brian Jones',
...          'Raymond Hettinger', 'Ned Batchelder']
```

```
>>> sorted(names, key=lambda name: name.split()[-1].lower())
['Ned Batchelder', 'David Beazley', 'Raymond Hettinger', 'Brian␣
↪Jones']
>>>
```

èőÍèőž

åř¡çőąlambdaèąÍè¿¿åijŔåĚĄèőÿä¡ăåőŽäźĽçőĂå■ŢåĞ¡æŢřïijŇä¡ĘæŸŕåőČçŽĎä¡£çŤÍæŸŕæIJL'éŹŘåĹűç
ä¡ăåŔłèČ¡æŇĞåőŽå■Ţäÿłèąlè¿¿åijŔïijŇåőČçŽĎåĂijåřśæŸŕæIJĂåŘŐçŽĎè£ŤåŻđåĂijãĂĆäź§åřśæŸŕèŕť'äÿ■è
åŇĚæŃňåd'Žäÿłèŕ■åŔěãĂĄæİąäżűèąÍè¿¿åijŔãĂĄè£■äžčäžěåŔĽåijĆåÿÿåd'ĎçŘEç■L'ç■L˜ãĂĆ

ä¡ăåŔŕäžěäÿ■ä¡£çŤÍlambdaèąÍè¿¿åijŔåřśèČ¡çijŰåĘŹåd'ğéČÍåĹĘpythonäžčçăĄãĂĆ
ä¡ĘæŸŕïijŇå¡ŞæIJL'äžžçijŰåĘŹåd'ğéĞŔèőąçőÙèąÍè¿¿åijŔåĂijçŽĎç§■åřŔåĞ¡æŢřæĹŰèĂĚéIJĂèęĄçŤÍæĹůa
ä¡ăåřśäijŽçIJŃåĹřlambdaèąÍè¿¿åijŔçŽĎèžńå¡śäžĘãĂĆ

7.7 åŇ£åŘ■åĞ¡æŢřæ■ŢèŐůåŔŸéĞŔåĂij

éŮőécŸ

ä¡ăçŤÍlambdaåőŽäźL'äžĘäÿĂäÿłåŇ£åŘ■åĞ¡æŢřïijŇåźűæČşåIJÍåőŽäźL'æŮűæ■ŢèŐůåĹřæ§ŘäžŻåŔŸéĞ

èğčåĘşæŰźæąĹ

åĚĹçIJŃäÿŃäÿŃéİćäżčçăĄçŽĎæŢĹæđIJïijŽ

```
>>> x = 10
>>> a = lambda y: x + y
>>> x = 20
>>> b = lambda y: x + y
>>>
```

çŐřåIJÍæĹŚéŮőä¡ăïijŇa(10)åŠŇb(10)è£ŤåŻđçŽĎçżŞæđIJæŸŕäžĂäźĹïij§åęĆæđIJä¡àèőd'äÿžçżŞæđIJæŸ

```
>>> a(10)
30
>>> b(10)
30
>>>
```

è£ŹåĚűäÿ■çŽĎåěěåęŹåIJÍäžŐlambdaèąÍè¿¿åijŔäÿ■çŽĎxæŸŕäÿĂäÿłèĞłçŤšåŔŸéĞŔïijŇ
åIJÍè£ŘèąŇæŮűçżŚåőŽåĂijïijŇèĂŇäÿ■æŸŕåőŽäźL'æŮűářçżŚåőŽïijŇè£Źèů§åĞ¡æŢřçŽĎéżŸèőd'åĂijåŔĆæ
åŽäæ■d'ïijŇåIJÍèŕĆçŤÍè£ŹäÿłlambdaèąÍè¿¿åijŔçŽĎæŮűåĂŹïijŇxçŽĎåĂijæŸŕæL'ğèąŇæŮűçŽĎåĂijãĂĆä¿Ň

```
>>> x = 15
>>> a(10)
25
>>> x = 3
>>> a(10)
13
>>>
```

åęĆæđIJä¡ăæČşèőľæ§Řäÿłåй£åŘ■åĞ¡æŢřåIJĺåőŽäźĽæŮűåřśæ■ŢèŐůåĹřåĂijïijŇåŕŕäžěářĘéĆčäÿłåŔĆ

```
>>> x = 10
>>> a = lambda y, x=x: x + y
>>> x = 20
>>> b = lambda y, x=x: x + y
>>> a(10)
20
>>> b(10)
30
>>>
```

èőĺèőž

åIJĺè£ŹéĞŇåĹŮåĞžæĺěçŽĎéŮőécŸæŸŕæŰřæĽŃå¿Ĺåőźæ ŸŞçŁŕçŽĎéŤŹèŕŕïijŇæIJL'äžŻæŰřæĽŃåŕŕè æŕŤåęĆïijŇéĂŽè£ĞåIJĺäÿĂäÿłå¿łçŐŕæĹŰåĹŮèąĺæŐĺåŕijäÿ■åĹŻåžžäÿĂäÿłlambdaèąĺè¿¿åijŔåĹŮèąĺïijŇåźűa

```
>>> funcs = [lambda x: x+n for n in range(5)]
>>> for f in funcs:
... print(f(0))
...
4
4
4
4
4
>>>
```

ä¡ĘæŸŕåőđéŹĚæŢĹæđIJæŸŕè£ŘèąŇæŸŕnçŽĎåĂijäÿžè£■äžčçŽĎæIJĂåŘŐäÿĂäÿłåĂijãĂĆçŐřåIJĺæĹŚă

```
>>> funcs = [lambda x, n=n: x+n for n in range(5)]
>>> for f in funcs:
... print(f(0))
...
0
1
2
3
4
>>>
```

éĂŽè£Ğä¡£çŤĺåĞ¡æŢřéżŸèőď˙åĂijåŔĆæŢřå¡ćåijŔïijŇlambdaåĞ¡æŢřåIJĺåőŽäźĽ˙æŮűåřśèČ¡çżŚåőŽåĹřå

7.8 åĞŔåřŚåŔŕèřČçŤĺåŕźèśąçŽĎåŔĆæŢřäÿłæŢř

éŮőécŸ

ä¡ăæIJL˙äÿĂäÿłèćńåĚűäżŰpythonäżčçăĄä¡£çŤĺçŽĎcallableåŕźèśąïijŇåŔŕèČ¡æŸŕäÿĂäÿłåŻđèřČåĞ¡æŢřä¡ĘæŸŕåőČçŽĎåŔĆæŢřåď˙łåď˙ŽäžĘïijŇåŕijèĞt˙èřČçŤĺæŮűåĞžéŤŹãĂĆ

èğčåĘşæŰźæąĹ

åęĆædIJéIJĂèęĄåĞŔåřŚæ§ŘäÿłåĞ¡æŢřçŽĎåŔĆæŢřäÿłæŢřijŇä¡ăåŔŕäżěä¡£çŤĺ
`functools.partial()` ãĂĆ `partial()` åĞ¡æŢřåĚĄèőÿä¡ăçżŹäÿĂäÿłæĹŰåď˙ŽäÿłåŔĆæŢřèő¿ç¡őåŻžäÿžäžĘæijŤçď˙žæÿĚæěŽïijŇåĄĞèő¿ä¡ăæIJL˙äÿŃéÍćè£ŹæăůçŽĎåĞ¡æŢřijŽ

```
def spam(a, b, c, d):
    print(a, b, c, d)
```

çŐřåIJĺæĹŚäżňä¡£çŤĺ `partial()` åĞ¡æŢřæİěåŽžåőŽæ§ŘäžŽåŔĆæŢřåĂijïijŽ

```
>>> from functools import partial
>>> s1 = partial(spam, 1) # a = 1
>>> s1(2, 3, 4)
1 2 3 4
>>> s1(4, 5, 6)
1 4 5 6
>>> s2 = partial(spam, d=42) # d = 42
>>> s2(1, 2, 3)
1 2 3 42
>>> s2(4, 5, 5)
4 5 5 42
>>> s3 = partial(spam, 1, 2, d=42) # a = 1, b = 2, d = 42
>>> s3(3)
1 2 3 42
>>> s3(4)
1 2 4 42
>>> s3(5)
1 2 5 42
>>>
```

åŔŕäžěçIJŃåĞž `partial()` åŽžåőŽæ§ŘäžŽåŔĆæŢřåźűè£ŤåŽđäÿĂäÿłæŰřçŽĎcallableåŕźèśąãĂĆè£Źã
çĎűåŘŐèů§äźŃåL˙■åůšçżŔèţŃåĂijè£ĞçŽĎåŔĆæŢřåŘĹåźűèţůæİěïijŇæIJĂåŘŐářĘæL˙ĂæIJL˙åŔĆæŢřäijäéĂ

èőĺèőž

æIJňèŁĆèęĄèğčåĘşçŽĎéŮőécŸæŸŕèőľ˙åŐ§æIJňäÿ■åĚijåőźçŽĎäžčçăĄåŔŕäžěäÿĂèţůåůěä¡IJãĂĆäÿŃéI

çňňäÿĂäÿłä¿Ńå■ŘæŸřïijŇåĄĞèő¿ä¡ăæIJL'äÿĂäÿłçĆźçŽĎåĹŮèąĺæİěèąĺçđ'ž(x,y)åİŘæăĞåĚČçżĎãĂĆ ä¡ăåŔřäžěä¡£çŤĺäÿŃéİćçŽĎåĞ¡æŢřæİěëőąçőŮäÿđ'çĆźäźŃéŮť'çŽĎèůİçężïijŽ

```
points = [ (1, 2), (3, 4), (5, 6), (7, 8) ]

import math
def distance(p1, p2):
    x1, y1 = p1
    x2, y2 = p2
    return math.hypot(x2 - x1, y2 - y1)
```

çŐřåIJĺåĄĞèő¿ä¡ăæČşäžěæ§ŘäÿłçĆźäÿžå§žçĆźïijŇæăźæ■őçĆźåŠŇå§žçĆźäźŃéŮť'çŽĎèůİçężæİěæŐŠå åĹŮèąĺçŽĎ `sort()` æŰźæşŢæŐěåŔŮäÿĂäÿłåĚşéŤőå■ŮåŔĆæŢřæİěèĞłåőŽäźL'æŐŠåžŔéĂżè¿ŚïijŇ ä¡ĘæŸŕåőČåŔłèČ¡æŐěåŔŮäÿĂäÿłå■ŢäÿłåŔĆæŢřçŽĎåĞ¡æŢř(distance()å¿ĹæŸŐæŸ¿æŸŕäÿ■çňęåŘĹæİąäžűçŐřåIJĺæĹŚäżňåŔŕäžěéĂŽè£Ğä¡£çŤĺ `partial()` æİěèğčåĘşè£ŹäÿłéŮőécŸïijŽ

```
>>> pt = (4, 3)
>>> points.sort(key=partial(distance,pt))
>>> points
[(3, 4), (1, 2), (5, 6), (7, 8)]
>>>
```

æŻt'è£ŻäÿĂæ■ëïijŇ`partial()` éĂŽåÿÿèćńçŤĺæİěå¿őèřČåĚűäžŮåžŞåĞ¡æŢřæL'Ăä¡£çŤĺçŽĎåŻđèřČå ä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿĂæőţäžčçăĄïijŇä¡£çŤĺ `multiprocessing` æİěåijĆæ■ěèőąçőŮäÿĂäÿłçżŞæđIJåĂijïijŇ çĎűåŘŐè£ŹäÿłåĂijèćńäijąéĂŠçżŹäÿĂäÿłæŐěåŘŮäÿĂäÿłresultåĂ

```
def output_result(result, log=None):
    if log is not None:
        log.debug('Got: %r', result)

# A sample function
def add(x, y):
    return x + y

if __name__ == '__main__':
    import logging
    from multiprocessing import Pool
    from functools import partial

    logging.basicConfig(level=logging.DEBUG)
    log = logging.getLogger('test')

    p = Pool()
    p.apply_async(add, (3, 4), callback=partial(output_result,␣
↪log=log))
    p.close()
    p.join()
```

å¡ŞçżŹ `apply_async()` æŔŘä¿ŻåŻđèřČåĞ¡æŢřæŮűïijŇéĂŽè£Ğä¡£çŤĺ `partial()` äijąéĂŠéćİåď'ŰçŽĎ `logging` åŘĆæŢřãĂĆ èĂŇ `multiprocessing` åŕźè£ŹäžŻäÿĂæŮăæL'Äç§ëâĂŤâĂŤåőČäžĔäžĔåŘłæŸŕä¡£çŤĺå■ŢäÿłåĂijæİěèřČçŤĺåŻđèřČåĞ¡æŢřãĂĆ

ä¡IJäÿžäÿĂäÿłçśžäijijçŽĎä¿Ńå■ŘïijŇèĂČèŹŚäÿŃçijŰåĘŹç¡ŚçżIJæIJ■åŁąåŹĺçŽĎéŮőéćŸïijŇsocketsæĺąåİŮèőľåőČåŔŸå¿Ůå¿Ĺåőźæ ŸŞãĂĆ äÿŃéİćæŸŕäÿłçőĂå■ŢçŽĎechoæIJ■åŁąåŹĺïijŽ

```
from socketserver import StreamRequestHandler, TCPServer

class EchoHandler(StreamRequestHandler):
    def handle(self):
        for line in self.rfile:
            self.wfile.write(b'GOT:' + line)

serv = TCPServer(('', 15000), EchoHandler)
serv.serve_forever()
```

äÿ■è£ĞïijŇåĄĞèő¿ä¡ăæČşçżŹEchoHandlerårćđåŁääÿĂäÿłåŔŕäžěæŐěåŔŮåĚűäžŰéĚ■ç¡őéĂĽéąźçŽĎ__init__ æŰźæşŢãĂĆærŤåęĆïijŽ

```
class EchoHandler(StreamRequestHandler):
    # ack is added keyword-only argument. *args, **kwargs are
    # any normal parameters supplied (which are passed on)
    def __init__(self, *args, ack, **kwargs):
        self.ack = ack
        super().__init__(*args, **kwargs)

    def handle(self):
        for line in self.rfile:
            self.wfile.write(self.ack + line)
```

è£ŹäźĹä£őæŤźåŘŐïijŇæĹŚäžňåřšäÿ■éIJĂèęĄæŸ¿åijŔåIJřåIJÍTCPServerçśžäÿ■æůžåŁääĽ■çijĂäžĘãĂĆä¡ĘæŸŕä¡ääĘ■æňąè£ŘèąŇçÍŃåžŔåŘŐäijŽæŁěçśžäijijäÿŃéİćçŽĎéŤŹèŕŕïijŽ

```
Exception happened during processing of request from ('127.0.0.1',␣
↪59834)
Traceback (most recent call last):
...
TypeError: __init__() missing 1 required keyword-only argument: 'ack
↪'
```

åĹİçIJŃètůæİěåě¡åČŔå¿ĹéŽ¿ä£őæ■čè£ŹäÿłéŤŹèŕŕïijŃéŹd'äžĘä£őæŤźsocketserver æĺąåİŮæžŘäžčçăĄæĹŰèĂĚä¡£çŤÍæ§ŘäžŽåěĞæĂłçŽĎæŰźæşŢäźŃåd'ŰãĂĆä¡ĘæŸŕïijŇåęĆæđIJä¡£çŤÍ partial() åŕśèČ¡å¿Ĺè¡żæİ¿çŽĎèğčåĘşâĂŤâĂŤçžŹåőČäijăéĂŠack åŔĆæŢřçŽĎåĂijæİěåĹİåğŃåŇŰå■şåŔŕïijŇåęĆäÿŃïijŽ

```
from functools import partial
serv = TCPServer(('', 15000), partial(EchoHandler, ack=b'RECEIVED:
↪'))
serv.serve_forever()
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇ__init__() æŰźæşŢäÿ■çŽĎack-åŔĆæŢřåčřæŸŐæŰźåijŔçIJŃäÿŁåŐžå¿ĹæIJL'èűčïijŇåĚűåőđåŕśæŸŕåčřæŸŐackäÿžäÿĂäÿłåijžåĹűåĚşéŤőå■åĚşäžŐåijžåĹűåĚşéŤőå■ŮåŔĆæŢréŮőécŸæĹŚäžňåIJĺ7.2åŕŘèŁĆæĹŚäžňåůšçžŔèőĺèőžè£ĞäžĘïijŇèŕźèĂĚåŔ

å¿Ĺåď'ŽæŮűåĂŹ partial() èČ¡åőđçŐřçŽĎæŢĹæđIJïijŇlambdaèąĺè¿¿åijŔäź§èČ¡åőđçŐřãĂĆærŤåęĆ

```
points.sort(key=lambda p: distance(pt, p))
p.apply_async(add, (3, 4), callback=lambda result: output_
↪result(result,log))
serv = TCPServer(('', 15000),
        lambda *args, **kwargs: EchoHandler(*args, ack=b'RECEIVED:',
↪ **kwargs))
```

è£ŹæăůåĘŹäź§èČ¡åőđçŐřåŘŇæăůçŽĎæŢĹæđIJïijŇäÿ■è£ĞçŻÿæŕŤèĂŇåůšäijŽæŸ¿å¿ŮæŕŤè¿ČèĞČèĆ
è£ŹæŮűåĂŹä¡£çŤĺ `partial()` åŕŕäžěæŻť'åŁăçŻť'èğĆçŽĎèąĺè¿¿ä¡ăçŽĎæĎŕåŻ¿(çžŹæ§ŘäžŽåŕĆæŢřécĎ

7.9 åřĘå■ŢæŰźæşŢçŽĎçśżè¡ňæ■ćäÿžåĞ¡æŢř

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłéŹđ' `__init__()` æŰźæşŢåđ'ŰåŕłåőŽäźL'äžĘäÿĂäÿłæŰźæşŢçŽĎçśżãĂĆäÿžäžĘçőĂ

èğčåĘşæŰźæąĹ

åđ'ğåđ'ŽæŢřæČĚåĘţäÿŃïijŇåŕŕäžěä¡£çŤĺéŮ■åŇĚæİěårĘå■ŢäÿłæŰźæşŢçŽĎçśżè¡ňæ■ćæĹŘåĞ¡æŢřãĂ
äÿ¿äÿłä¿Ńå■ŘïijŇäÿŃéİćçđ'žä¿Ńäÿ■çŽĎçśżåĚĄèőÿä¡£çŤĺèĂĚæăźæ■őæ§Räÿłæĺąæİ£æŰźæąĹæİèèŐůåŔŰå

```
from urllib.request import urlopen

class UrlTemplate:
    def __init__(self, template):
        self.template = template

    def open(self, **kwargs):
        return urlopen(self.template.format_map(kwargs))

# Example use. Download stock data from yahoo
yahoo = UrlTemplate('http://finance.yahoo.com/d/quotes.csv?s={names}
↪&f={fields}')
for line in yahoo.open(names='IBM,AAPL,FB', fields='sl1c1v'):
    print(line.decode('utf-8'))
```

è£ŹäÿłçśżåŕŕäžěèćńäÿĂäÿłæŻť'çőĂå■ŢçŽĎåĞ¡æŢřæİěäžčæŻ£ïijŽ

```
def urltemplate(template):
    def opener(**kwargs):
        return urlopen(template.format_map(kwargs))
    return opener

# Example use
yahoo = urltemplate('http://finance.yahoo.com/d/quotes.csv?s={names}
↪&f={fields}')
```

```
for line in yahoo(names='IBM,AAPL,FB', fields='sl1c1v'):
    print(line.decode('utf-8'))
```

èőÍèőž

åď ğéČĺåĹĘæČĚåĘţäÿŃïijÑä¡ăæŃěæIJL'äÿĂäÿłå■ŢæŰźæşŢçśżçŽĎåŐ§åŻăæŸŕéIJĂèęĄå■ŸåĆĺæ§ŘäžZ
æŕŤåęĆïijÑåőŽäźL'UrlTemplateçśżçŽĎåŤŕäÿĂçŻőçŽĎåřśæŸŕåĚĹåIJĺæ§Řäÿłå IJřæŰźå■ŸåĆĺæĺąæİ£åĂijïijÑ

ä¡£çŤĺäÿĂäÿłåĘĚéČĺåĞ¡æŢřæĹŰèĂĚéŮ■åŇĚçŽĎæŰźæąĹéĂŽåÿÿäijŽæŻť'äijŸéŻĚäÿĂäžŽãĂĆçőĂå■
åŘłäÿ■è£ĞåIJĺåĞ¡æŢřåĘĚéČĺåÿęäÿŁäžĘäÿĂäÿłécİåď ŰçŽĎåŘŸéĞŘçŐŕåćČãĂĆéŮ■åŇĚåĚşéŤőçL'źçĆźåřśa
åŽăæ■d'ïijÑåIJĺæĹŚäżñçŽĎèğčåĘşæŰźæąĹäÿ■ïijÑopener() åĞ¡æŢřèőřä¡ŘäžĘ
template åŘĆæŢřçŽĎåĂijïijÑåźűåIJĺæŐěäÿÑæİěçŽĎèřČçŤĺäÿ■ä¡£çŤĺåőČãĂĆ

äżżä¡ŢæŮűåĂŹåŘłèęĄä¡ăçćřåĹřéIJĂèęĄçżŹæ§Řäÿłå Ğ¡æŢřåćđåŁäécİåď ŰçŽĎçŁűæĂĄä£ąææĄřçŽĎéŮ
çŻÿæŕŤårĘä¡ăçŽĎåĞ¡æŢřè¡ñæ■ćæĹŘäÿĂäÿłçśżèĂŇèĺĂïijŇéŮ■åŇĚéĂŽåÿÿæŸŕäÿĂçğ■æŻť åŁăçőĂæť ĄåŠ

7.10 åÿęécİåď ŰçŁűæĂĄä£ąææĄřçŽĎåŻđèřČåĞ¡æŢř

éŮőécŸ

ä¡ăçŽĎäżčçăĄäÿ■éIJĂèęĄä¿İèţŰåĹřåŻđèřČåĞ¡æŢřçŽĎä¡£çŤĺ(æŕŤåęĆäžŇäżűåď ĎçŘĘåŹĺãĂĄç■L'å¿Ě
åźűäÿŤä¡ăè£ŸéIJĂèęĄèől'åŻđèřČåĞ¡æŢřæŇěæIJL'écİåď ŰçŽĎçŁűæĂĄåĂijïijÑäżěä¿£åIJĺåőČçŽĎåĘĚéČĺä

èğčåĘşæŰźæąĹ

è£ŹäÿĂåřŔèŁĆäÿżèęĄèőĺèőžçŽĎæŸŕéĆčäžŻåĞžçŐřåIJĺå¿Ĺåď ŽåĞ¡æŢřåžŞåŠŇæąĘæđűäÿ■çŽĎåŻđèř
äÿžäžĘæijŤçď žäÿŐæţŃèřŢïijÑæĹŚäżñåĚĹåőŽäźL'åęĆäÿŃäÿĂäÿłéIJĂèęĄèřČçŤĺåŻđèřČåĞ¡æŢřçŽĎåĞ¡æŢ

```
def apply_async(func, args, *, callback):
    # Compute the result
    result = func(*args)

    # Invoke the callback with the result
    callback(result)
```

åőđéŹĚäÿŁïijÑè£Źæőţäžčçă ĄåŘŕäžěåĄŽäżżä¡ŢæŻť'énŸçžğçŽĎåď ĎçŘĘïijÑåŇĚæŃňçž£çÍŃãĂĄè£Żçĺ
æĹŚäżñäžĚäżĚåŘłéIJĂèęĄåĚşæşĺåŻđèřČåĞ¡æŢřçŽĎèřČçŤĺãĂĆäÿŃéİćæŸŕäÿĂäÿłæijŤçď žæĂŐæăůä¡£çŤĺä

```
>>> def print_result(result):
...     print('Got:', result)
...
>>> def add(x, y):
...     return x + y
...
>>> apply_async(add, (2, 3), callback=print_result)
```

```
Got: 5
>>> apply_async(add, ('hello', 'world'), callback=print_result)
Got: helloworld
>>>
```

æşÍæĎŔåĹř `print_result()` åĞ¡æŢřäžĚäžĚåŔłæŐěåŔŮäÿĂäÿłåŔĆæŢř `result` ãĂĆäÿ■èČ¡åĘ■äijååĚěåĚűäžŰä£qæĄřãĂĆ èĂŇå¡Şä¡ăæČşèőľ åŻđěřČåĞ¡æŢřèő£éŮőåĚűäžŰåŔŸéĞŔæĹŰè

äÿžäžĘèőľ åŻđěřČåĞ¡æŢřèő£éŮőåđ' ŰéČĺä£qæĄřïijŇäÿĂçğ■æŰźæşŢæŸŕä¡£çŤĺäÿĂäÿłçżŚåőŽæŰźæşŢ æŕŤåęĆïijŇäÿŃéÍćè£ŹäÿłçśžäijŽä£Íå■ŸäÿĂäÿłåĘĚéČĺåžŔåĹŮåŔůïijŇærŔæňqæŐěæŤűåĹřäÿĂäÿł `result` çŽĎæŮűåĂŹåžŔåĹŮåŔůåŁă1ïijŽ

```
class ResultHandler:

    def __init__(self):
        self.sequence = 0

    def handler(self, result):
        self.sequence += 1
        print('[{}] Got: {}'.format(self.sequence, result))
```

ä¡£çŤĺè£ŹäÿłçśžçŽĎæŮűåĂŹïijŇä¡ăåĚĹåĹŻåžžäÿĂäÿłçśžçŽĎåőđä¿ŃïijŇçĎűåŘŐçŤĺåőĆçŽĎ `handler()` çżŚåőŽæŰźæşŢæÍěåĄŽäÿžåŻđěřČåĞ¡æŢřïijŽ

```
>>> r = ResultHandler()
>>> apply_async(add, (2, 3), callback=r.handler)
[1] Got: 5
>>> apply_async(add, ('hello', 'world'), callback=r.handler)
[2] Got: helloworld
>>>
```

çňňäžŇçğ■æŰźåijŔïijŇä¡IJäÿžçśžçŽĎæŻ£äžčïijŇåŕŕäžěä¡£çŤĺäÿĂäÿłéŮ■åŇĚæ■ŢèŐůçŁűæĂĄåĂijïijŇ

```
def make_handler():
    sequence = 0
    def handler(result):
        nonlocal sequence
        sequence += 1
        print('[{}] Got: {}'.format(sequence, result))
    return handler
```

äÿŃéÍćæŸŕä¡£çŤĺéŮ■åŇĚæŰźåijŔçŽĎäÿĂäÿłä¿Ńå■ŘïijŽ

```
>>> handler = make_handler()
>>> apply_async(add, (2, 3), callback=handler)
[1] Got: 5
>>> apply_async(add, ('hello', 'world'), callback=handler)
[2] Got: helloworld
>>>
```

è£ŸæIJL'åŔęåđ' ŰäÿĂäÿłæŻt'énŸçžğçŽĎæŰźæşŢïijŇåŕŕäžěä¡£çŤĺå■ŔçĺŃæÍěåőŇæĹŘåŘŇæăůçŽĎäžŇ

```
def make_handler():
    sequence = 0
    while True:
        result = yield
        sequence += 1
        print('[{}] Got: {}'.format(sequence, result))
```

åŕźäžŐå■ŔçÍŃïijŇä¡äéIJĂèęĄä¡£çŤĺåőČçŽĎ send() æŰźæşŢä¡IJäÿžåŻđèřČåĞ¡æŢřïijŇåęĆäÿŃæĽĂç

```
>>> handler = make_handler()
>>> next(handler) # Advance to the yield
>>> apply_async(add, (2, 3), callback=handler.send)
[1] Got: 5
>>> apply_async(add, ('hello', 'world'), callback=handler.send)
[2] Got: helloworld
>>>
```

èőÍèőž

å§žäžŐåŻđèřČåĞ¡æŢřçŽĎè¡ŕäžűéĂŽåÿÿéČ¡æIJĽåŔŕèČ¡åŔŸå¿Ůéİđåÿÿåď■æİĆãĂĆäÿĂéČĺåĹĘåŐ§åŻ
åŻăæ■ď¨ïijŇèŕůæśĆæĽĝèąŇåŠŇåď ĎçŘĘçżŞæđIJäźŃéŮťçŽĎæĽĝèąŇçŐŕåćČåőđéŹĚäÿŁåůšçżŔäÿćåď śäžĘ
éČčä¡ääŕšå£ĚéążåŐżèğčåĘşåęĆä¡Ţä£İå■ŸåŠŇæĄćåď■çŻÿåĚşçŽĎçĽűæĂĄä£ąæĄŕäžĘãĂĆ

èĞşåřŚæIJĽäÿď çğ■äÿżèęĄæŰźåijŔæİěæ■ŢèŐůåŠŇä£İå■ŸçŁűæĂĄä£ąæĄŕïijŇä¡ääŔŕäžěåIJÍäÿĂäÿłåŕź
äÿď çğ■æŰźåijŔçŻÿæŕŤïijŇéŮ■åŇĚæĹŰèőÿæŸŕæŻť åŁăè¡žéĞŘçžğåŠŇèĞłçĎűäÿĂçĆźïijŇåŻăäÿžåőČäžñå
åőČäžñè£ŸèČ¡èĞłåŁĺæ■ŢèŐůæĽĂæIJĽèćńä¡£çŤĺåĹřçŽĎåŔŸéĞŘãĂĆåŻăæ■ď¨ïijŇä¡ăæŮăéIJĂåŐżæŃĚå£

åęĆæđIJä¡£çŤĺéŮ■åŇĚïijŇä¡äéIJĂèęĄæşĺæĎŘåŕźéĆčäžŻåŘŕä£őæŤźåŔŸéĞŘçŽĎæŞ■ä¡IJãĂĆåIJÍäÿŁé
nonlocal åčřæŸÖèŕ■åŘěçŤĺæİěæŇĞçď žæÖèäÿŃæİěçŽĎåŔŸéĞŘäijŻåIJĺåŻđèřČåĞ¡æŢřäÿ■èćńä£őæŤźã

èĂŇä¡£çŤĺäÿĂäÿłå■ŔçÍŃæİěä¡IJäÿžäÿĂäÿłåŻđèřČåĞ¡æŢřåŕśæŻť æIJĽèűčäžĘïijŇåőČèů§éŮ■åŇĚæŰźa
æ§Řçğ■æĎŘäźĽäÿŁæİěèőšïijŇåőČæŸ¿å¿ŮæŻť åŁăçőĂæť ĄïijŇåŻăäÿžæĂżåĚśåŕśäÿĂäÿłåĞ¡æŢřèĂŇåůšãĂ
åźűäÿŤïijŇä¡ääŔŕäžěå¿ĹèĞłçŤśçŽĎä£őæŤźåŔŸéĞŘèĂŇæŮăéIJĂåŐżä¡£çŤÍ nonlocal
åčřæŸÖãĂĆ è£Źçğ■æŰźåijŔåŤŕäÿĂçijžçĆźåŕśæŸŕçŻÿåŕźäžÖåĚűäžŰPythonæŁĂæIJŕèĂŇåůšæĹŰèőÿæŕŤè¿
åŘęåď Úè£ŸæIJĽäÿĂäžŻæŕŤè¿ČéŽ¿æĞĆçŽĎéČĺåĹĘïijŇæŕŤåęĆä¡£çŤĺäźŃåĽ■éIJĂèęĄèřČçŤÍ
next() ïijŇåődéŹĚä¡£çŤĺæŮűè£Źäÿłæ■ěéłď å¿Ĺåőźæ ŸŞèćńå£Ÿèőřã ĂĆ
ař¡çőąåęĆæ■ď¨ïijŇå■ŔçÍŃè£ŸæIJĽåĚűäžŰçŤĺåď ĎïijŇæŕŤåęĆä¡IJäÿžäÿĂäÿłåĘĚèĄŤåŻđèřČåĞ¡æŢřçŽĎåőŻ

åęĆæđIJä¡ääżĚäżĚåŔłéIJĂèęĄçżŹåŻđèřČåĞ¡æŢřäijăéĂŠéİåď ŰçŽĎåĂijçŽĎèŕİïijŇè£ŸæIJĽäÿĂçğ■ä¡
partial() çŽĎæŰźåijŔäź§å¿ĹæIJĽçŤĺãĂĆ åIJÍæšąæIJĽä¡£çŤÍ partial()
çŽĎæŮűåĂŹïijŇä¡ääŔŕèČ¡çżŔåÿÿçIJŃåĹřäÿŃéİćè£Źçğ■ä¡£çŤĺlambdaèąĺè¿¿åijŔçŽĎåď■æİĆäžčçĂąijŽ

```
>>> apply_async(add, (2, 3), callback=lambda r: handler(r, seq))
[1] Got: 5
>>>
```

åŔŕäžěåŔĆèĂČ7.8åřŔèŁĆçŽĎåĞääÿłçď žä¿ŃïijŇæŢŹä¡ääęĆä¡Ţä¡£çŤÍ partial()
æİěæŻť æŤźåŔĆæŢřç■¿åŘ■æİěçőĂåŇŰäÿŁè£řäžčçăĄãĂĆ

7.11 åĘĚèĄŤåŻđèřČåĞ¡æŢř

éŮőécŸ

å¡Şä¡äçijŰåĘŹä¡£çŤĺåŻđèřČåĞ¡æŢřçŽĎäžčçăĄçŽĎæŮűåĂŹïijŇæŃĚå£Čå¿Ĺåd'ŽåřŔåĞ¡æŢřçŽĎæĽl'å
ä¡ăåÿŇæIJŻæĽ¿åĹřæ§ŘäÿłæŭźæşŢæİèèől'äžčçăĄçIJŃäÿŁåŐżæŻt'åČŔæŸŕäÿĂäÿłæŹőéĂŽçŽĎæĽğèąŇåžF

èğčåĘşæŰźæąĹ

éĂŽè£Ğä¡£çŤĺçŤ§æĹŘåŹĺåŠŇå■ŔçÍŃåŔŕäžěä¡£å¿ŮåŻđèřČåĞ¡æŢřåĘĚèĄŤåIJÍæ§ŘäÿłåĞ¡æŢřäÿ■ãĂĆ
äÿžäžĘæijŤçd'žèŕt'æŸŐïijŇåĄĞèő¿ä¡ăæIJL'åęĆäÿŃæĽĂçd'žçŽĎäÿĂäÿłæĽğèąŇæ§Řçğ■èőąçőŮäżżåŁąçĎű

```
def apply_async(func, args, *, callback):
    # Compute the result
    result = func(*args)

    # Invoke the callback with the result
    callback(result)
```

æŐěäÿŃæİèèől'æĹŚäžňçIJŃäÿĂäÿŃäÿŃéİćçŽĎäžčçăĄïijŇåőČåŇĚåŘŕäžĘäÿĂäÿł
`Async` çśżåŠŇäÿĂäÿł `inlined_async` èčĚéěřåŹĺïijŽ

```
from queue import Queue
from functools import wraps

class Async:
    def __init__(self, func, args):
        self.func = func
        self.args = args

def inlined_async(func):
    @wraps(func)
    def wrapper(*args):
        f = func(*args)
        result_queue = Queue()
        result_queue.put(None)
        while True:
            result = result_queue.get()
            try:
                a = f.send(result)
                apply_async(a.func, a.args, callback=result_queue.
↪put)
            except StopIteration:
                break
    return wrapper
```

è£Źäÿd'äÿłäžčçăĄçL'Ğæőţå ĚĄèőÿä¡ăä¡£çŤĺ `yield` èr■åŘěåĘĚèĄŤåŻđèřČæ■ěéłd'ãĂĆærŤåęĆïijŽ

```
def add(x, y):
    return x + y

@inlined_async
def test():
    r = yield Async(add, (2, 3))
    print(r)
    r = yield Async(add, ('hello', 'world'))
    print(r)
    for n in range(10):
        r = yield Async(add, (n, n))
        print(r)
    print('Goodbye')
```

åęĆæđIJä¡ăèřČçŤÍ `test()` ïijŇä¡ăäijŽå¿ŮåĹřçśžäijijåęĆäÿŃçŽĎè¿ŞåĞžïijŽ

```
5
helloworld
0
2
4
6
8
10
12
14
16
18
Goodbye
```

ä¡ăäijŽåŔŚçŐřïijŇéŹď äžĘéĆčäÿłçĽźåĹńçŽĎèčĚéěřåŹĺåŠŇ `yield` èŕ■åŔěåď ŰïijŇåĚűäžŮåIJřæŰźåźűæšąæIJĽåĞžçŐřäżżä¡ŢçŽĎåŻđèřČåĞ¡æŢř(åĚűåőđæŸŕåIJĺåŘŐåŔřåőŽäź

èőĺèőž

æIJňåřŔèŁĆäijŽåőđåőđåIJĺåIJĺçŽĎæţŃèŕŢä¡ääĚşäžŐåŻđèřČåĞ¡æŢřãĂąČŤ§æĹŘåŹĺåŠŇæŐğåĹűæţĄçŽ

éęŰåĚĹïijŇåIJĺéIJĂèęĄä¡£çŤĺåĹřåŻđèřČçŽĎäżčçăĄäÿ■ïijŇåĚşéŤőçĆźåIJĺäžŐå¡ŞåĽ■èőąçőŮåůëä¡IJäij
å¡Şèőąçő ŮéĞ■åŘŕæŮűïijŇåŻđèřČåĞ¡æŢřèćńèřČçŤĺæİěçżğçż■åď ĎçŘĘçżŞæđIJãĂĆapply_async()
åĞ¡æŢřæijŤçď žäžĘæĽğèąŇåŻđèřČçŽĎåőđéŹĚéĂżè¿Śïij Ň år¡çőąåőđéŹĚæČĚåĘţäÿ■åőČåŔřèČ¡äijŽæŻť åŁă

èőąçőŮçŽĎæŽĆåĄIJäÿŐéĞ■åŘŕæĂİèůřèů§çŤ§æĹŘåŹĺåĞ¡æŢřçŽĎæĽğèąŇæĺąåđŃäÿ■èřŃèĂŇåŘĹãĂ
åĚůä¡ŞæİěèőšïijŇyield æŞ■ä¡IJäijŽä¡£äÿĂäÿłçŤ§æĹŘåŹĺåĞ¡æŢřäžğçŤ§äÿĂäÿłåĂijåźűæŽĆåĄIJãĂĆ
æŐěäÿŃæİěèřČçŤĺçŤ§æĹŘåŹĺçŽĎ `__next__()` æĹŰ `send()`
æŰźæşŢåŔĹäijŽèőľåőČäžŐæŽĆåĄIJåď ĎçżğçżæĽğèąŇãĂĆ

æăźæ■őè£ŹäÿłæĂİèůřïijŇè£ŹäÿĂåřŔèŁĆçŽĎæäÿå£Čåřšå IJĺ `inline_async()`
èčĚéěřåŹĺåĞ¡æŢřäÿ■äžĘãĂĆ åĚşéŤőçĆźåřśæŸŕïijŇèčĚéěřåŹĺäijŽéĂŘæ■ěéĄ■åŐĘçŤ§æĹŘåŹĺåĞ¡æŢřçŽĎæ
`yield` èŕ■åŔěijŇæŕŔäÿĂæňąäÿĂäÿłãĂĆ äÿžäžĘè£ŹæăůåĄŽïijŇåĹŽåijĂåğŃçŽĎæŮűåĂŹåĹŽåžžäžĘäÿĂä
`result` éŸ§åĹŮåźűåŘŚéĞŇéİćæŤ¿åĚěäÿĂäÿł `None` åĂijãĂĆ

çĎűåŘŐåijĂåğŃäÿĂäÿłå¿łçŐŕæŞ■ä¡IJïijŇäžŐéŸ§åĹŮäÿ■åŘŰåĞžçżŞæđIJåĂijåźűåŔŚéĂĄçżŹçŢ§æĹŘåŹĺï
`yield` èŕ■åŘëïijŇ åIJĺè£ŹéĞŃäÿĂäÿł `Async` çŽĎåőđä¿ŃècńæŐěåŘŮåĹřãĂĆçĎűåŘŐå¿łçŐŕåijĂåğŃæčĂæ
`apply_async()` ãĂĆ çĎűèĂŃïijŇè£Źäÿłèőąçő̊ŮæIJL'äÿłæIJĂèŕąåijĆéČĺåĹĘæŸŕåőČåźűæšąæIJL'ä¡£çŢĺäy
`put()` æŰźæşŢæİěåŻđèřČãĂĆ

è£ŹæŮűåĂŹïijŇæŸŕæŮűåĂŹèŕęçżĘèğčéĞŁäÿŃåĹřåžŢåŔŚçŢ§äžĘäżĂäźĹäžĘãĂĆäÿżå¿łçŐŕçńŃå■şè£
`get()` æŞ■ä¡IJãĂĆ åęĆæđIJæŢřæ■őå■ŸåIJĺïijŇåőČäÿĂåőŽæŸŕ `put()`
åŻđèřČå■ŸæŢ¿çŽĎçżŞæđIJãĂĆåęĆæđIJæšąæIJL'æŢřæ■őïijŇéĆčäźĹåĚĹæŽĆåĄIJæŞ■ä¡IJåźűç■L'å¿ĚçżŞæ
è£ŹäÿłåĚůä¡ŞæĂŐæăůåőđçŐřæŸŕçŢś `apply_async()` åĞ¡æŢřæİěåĘşåőŽçŽĎãĂĆ
åęĆæđIJä¡ăäÿ■çŻÿä£ąäijŽæIJL'è£ŹäźĹçěđåěĞçŽĎäžŇæČĚïijŇä¡ăåŕŕäżěä¡£çŢĺ
`multiprocessing` åžŞæİèèŕŢäÿĂäÿŃïijŇ åIJĺå■ŢçŃňçŽĎè£ŽçĺŃäÿ■æL'ĝèąŇåijĆæ■ěèőąçő̊ŮæŞ■ä¡IJïijŇ

```
if __name__ == '__main__':
    import multiprocessing
    pool = multiprocessing.Pool()
    apply_async = pool.apply_async

    # Run the test function
    test()
```

åőđéŹĚäÿŁä¡ăäijŽåŔŚçŐřè£ŹäÿłçIJ§çŽĎåřśæŸŕè£ŹæăůçŽĎïijŇä¡ĘæŸŕèęĄèğčéĞŁæÿĚæěŽåĚůä¡ŞçŽĺ

åřĘåd'■æİĆçŽĎæŐğåĹűæţĄéŽŘèŮŔåĹřçŢ§æĹŘåŹĺåĞ¡æŢřèČŃåŘŐçŽĎä¿Ńå■ŘåIJĺæăĞåĞĘåžŞåŠŃç
ærŤåęĆïijŇåIJĺ `contextlib` äÿ■çŽĎ `@contextmanager`
èčĚéěřåŹĺä¡£çŢĺäžĘäÿĂäÿłäżd'äžžèt'źèğčçŽĎæŁĂåůğïijŇ éĂŽè£ĞäÿĂäÿł `yield`
èŕ■åŘěåřĘè£ŽåĚěåŠŃçężåijĂäÿŁäÿŃæŰĞçőąçŘĘåŹĺçšŸåŘĹåIJĺäÿĂèţůãĂĆ
åŘęåd'ŰéİđåÿÿæţĄèąŇçŽĎ `Twisted` åŃĚäÿ■äź§åŃĚåŘńäžĘéİđåÿÿçśżäijijçŽĎåĘĚèĄŤåŻđèřČãĂĆ

7.12 èő£éŮőéŮ■åŃĚäÿ■åőŽäźĽçŽĎåŔŸéĞŔ

éŮőécŸ

ä¡ăæČşèęĄæL'l'åśŢåĞ¡æŢřäÿ■çŽĎæ§ŘäÿłéŮ■åŃĚïijŇåĚĄèőÿåőČèČ¡èő£éŮőåŠŃä£őæŤźåĞ¡æŢřçŽĎå

èğčåĘşæŰźæąĹ

éĂŽåÿÿæİèèőšïijŇéŮ■åŃĚçŽĎåĘĚéČĺåŔŸéĞŔåŕźäžŐåd'ŰçŢŃæİèèőšæŸŕåőŃåĚĺéŽŘèŮŔçŽĎãĂĆ
ä¡ĘæŸŕïijŇä¡ăåŕŕäžěéĂŽè£ĞçijŰåĘŹèő£éŮőåĞ¡æŢřåźűåřĘåĚűä¡IJäÿžåĞ¡æŢřåśđæĂğçżŠåőŽåĹřéŮ■åŃĚäy

```
def sample():
    n = 0
    # Closure function
    def func():
        print('n=', n)

    # Accessor methods for n
    def get_n():
```

```
        return n

    def set_n(value):
        nonlocal n
        n = value

    # Attach as function attributes
    func.get_n = get_n
    func.set_n = set_n
    return func
```

äÿŃéÌćæŸŕä¡£çŤÍçŽĎä¿Ńå■Ř:

```
>>> f = sample()
>>> f()
n= 0
>>> f.set_n(10)
>>> f()
n= 10
>>> f.get_n()
10
>>>
```

èőÍèőž

äÿžäžĘèŕť æŸŐæÿĚæěŽåőČåęĆä¡Ţåůëä¡IJçŽĎïijŇæIJL'äÿd'çĆźéIJĂèęĄèğčéĞŁäÿĂäÿŃãĂĆéęŰåĚĹïij
åčřæŸŐåŔŕäżěèől'æĹŠäžňçijŰåĘŹåĞ¡æŢřæİěä‡őæŤźåĘĚéČĺåŔŸéĞŔçŽĎåĂijãĂĆ
åĚűæňąïijŇåĞ¡æŢřåšđæĂğåĚĄèőÿæĹŚäžňçŤĺäÿĂçğ■å¿ĹçőĂå■ŢçŽĎæŰźåijŔåřĘèő£éŮőæŰźæşŢçżŚåőŽåĹ

è£ŸåŔŕäżěè£ŻäÿĂæ■ěçŽĎæL'l'åśŢïijŇèől'éŮ■åŇĚæĺąæŃ§çśżçŽĎåőđä¿ŃãĂĆä¡äèęĄåĄŽçŽĎäżĚäżĚa

```
import sys
class ClosureInstance:
    def __init__(self, locals=None):
        if locals is None:
            locals = sys._getframe(1).f_locals

        # Update instance dictionary with callables
        self.__dict__.update((key,value) for key, value in locals.
↪items()
                             if callable(value) )
    # Redirect special methods
    def __len__(self):
        return self.__dict__['__len__']()

# Example use
def Stack():
    items = []
    def push(item):
```

```
        items.append(item)

    def pop():
        return items.pop()

    def __len__():
        return len(items)

    return ClosureInstance()
```

äÿŃéİćæŸŕäÿĂäÿłäžď äžŠåijŔäijŽèŕİæİěæijŤçď žåőČæŸŕåęĆä¡Ţåůëä¡IJçŽĎïijŽ

```
>>> s = Stack()
>>> s
<__main__.ClosureInstance object at 0x10069ed10>
>>> s.push(10)
>>> s.push(20)
>>> s.push('Hello')
>>> len(s)
3
>>> s.pop()
'Hello'
>>> s.pop()
20
>>> s.pop()
10
>>>
```

æIJL'èűčçŽĎæŸŕïijŃè£ŹäÿłäžčçăĄè£ŘèąŇèţůæİěäijŽærŤäÿĂäÿłæŹőéĂŽçŽĎçśżåőŽäźL'èęĄå£ńå¿Ĺåď

```
class Stack2:
    def __init__(self):
        self.items = []

    def push(self, item):
        self.items.append(item)

    def pop(self):
        return self.items.pop()

    def __len__(self):
        return len(self.items)
```

åęĆædIJè£ŹæăůåĄŽïijŃä¡ăäijŽå¿ŮåĹřçśżäijijåęĆäÿŃçŽĎçżŞædIJïijŽ

```
>>> from timeit import timeit
>>> # Test involving closures
>>> s = Stack()
>>> timeit('s.push(1);s.pop()', 'from __main__ import s')
0.9874754269840196
>>> # Test involving a class
```

```
>>> s = Stack2()
>>> timeit('s.push(1);s.pop()', 'from __main__ import s')
1.0707052160287276
>>>
```

çżŞæđIJæŸ¿çd'žïijŇéŮ■åŇĚçŽĎæŰźæąĹè£ŘèąŇètůæİěèęĄå£ńåd'ğæęĆ8%ïijŇåd'ğéČĺåĹĘåŐ§åŻăæŸ
éŮ■åŇĚæŻt'å£ńæŸŕåŻääÿžäÿ■äijŽæűL'åŔŁåĹřéćİåd'ŰçŽĎselfåŔŸéĞŔãĂĆ

Raymond HettingerårźäžŐè£ŹäÿłéŮőécŸèő¿èőąåĞžäžĘæŻt'åŁäéŽ¿äżěçŘĘèğčçŽĎæŤźè£ŻæŰźæąĹãĂ
èĂŇäÿŤåőČåŔłæŸŕçIJ§åőđçśżçŽĎäÿĂäÿłåěĞæĂłçŽĎæŻ£æ■ćèĂŇåůšïijŇä¿ŃåęĆïijŇçśżçŽĎäÿżèęĄçĽźæ
åźűäÿŤä¡ăèęĄåĄŽäÿĂäžŻåĚűäżŰçŽĎåůěä¡IJæĽ■èČ¡èőľäÿĂäžŻçĽźæőŁæŰźæşŢçŤ§æŢĹ(æŕŤåęĆäÿŁéIć
`ClosureInstance` äÿ■éĞ■åĘŹè£ĞçŽĎ `__len__()` åőđçŐřãĂĆ)

æIJĂåŘŐïijŇä¡ăåŔŕèČ¡è£ŸäijŽèőľåĚűäżŰéŸĚèŕża¡ääžčçăĄçŽĎäžžæĎ§åĹřçŰŚæČŚïijŇäÿžäżĂäźĹåőČ
(å¡ŞçĎűïijŇäżŰäżňäź§æČşç§ěéĄŞäÿžäżĂäźĹåőČè£ŘèąŇètůæİěäijŽæŻt'å£ń)ãĂĆår¡çőąåęĆæ■d'ïijŇè£Źåŕźä

æĂżä¡ŞäÿŁèőšïijŇåIJĺéĚ■ç¡őçŽĎæŮűåĂŹçżŹéŮ■åŇĚæůżåŁăæŰźæşŢäijŽæIJL'æŻt'åd'ŽçŽĎåőđçŤĺåŁ
æŕŤåęĆä¡ăéIJĂèęĄéĞ■ç¡őåĘĚéČĺçŁűæĂĄãĂĄåĹůæŰřçijŞåĘšåŇžãĂĄæŸĚéŹd'çijŞå■ŸæĹŰåĚűäżŰçŽĎåŔ

CHAPTER 10

çňňåĚńçńäïijŽçśżäÿŐåŕźèśą

æIJňçńääÿżèęĄåĚşæşÍçĆźçŽĎæŸŕåŠŇçśżåőŽäźĽæIJĽåĚşçŽĎåÿÿèğĄçijŰçÍŃæÍąåđŃãĂĆåŇĚæŃňèőľçśżåřĄèčĚæŁĂæIJŕãĂĄçżğæĽ£ãĂĄåĘĚå■Ÿçőąç ŘĘäżěåŔŁæIJĽçŤÍçŽĎèő¿èőąæÍąåijŔãĂĆ

Contents:

8.1 æŤźåŔŸåŕźèśąçŽĎå■ŮçňęäÿšæŸ¿çď'ž

éŮőécŸ

ä¡ăæČşæŤźåŔŸåŕźèśąåőđä¿ŃçŽĎæĽŞå■řæĹŰæŸ¿çď'žè¿ŞåĞžïijŇèőľåőČäžňæŻť'åĚůåŔŕèŕźæĂğãĂĆ

èğčåĘşæŰźæąĹ

èęĄæŤźåŔŸäÿĂäÿłåőđä¿ŃçŽĎå■Ůçňęäÿšèąĺçď'žïijŇåŔŕéĞ■æŰřåőŽäźĽ'åőČçŽĎ `__str__()` åŠŇ `__repr__()` æŰźæşŢãĂĆä¿ŃåęĆïijŽ

```
class Pair:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __repr__(self):
        return 'Pair({0.x!r}, {0.y!r})'.format(self)

    def __str__(self):
        return '({0.x!s}, {0.y!s})'.format(self)
```

__repr__() æŰźæşŢè£ŤåŻđäÿĂäÿłåőđä¿ŃçŽĎäžčçăĄèąÍçď'žå¡ćåijŔïijŇéĂŽåÿÿçŤÍæİěéĞ■æŰřædĎ åĘĚç¡őçŽĎ repr() åĞ¡æŢřè£ŤåŻđè£Źäÿłå■Ůçňęäÿšïij Ňèů§æĹŚäžňä¡£çŤÍäžď'äžŠåijŔèğčéĞŁåŹÍæŸ¿çď'žc __str__() æŰźæşŢåřĘåőđä¿Ńè¡ňæ■ćäÿžäÿĂäÿłå■Ůçňęäÿšïij Ňä¡£çŤÍ str() æĹŰ print() åĞ¡æŢřäijŽè¿ŞåĞžè£Źäÿłå■Ůçňęäÿšã ĂĆærŤåęĆïijŽ

```
>>> p = Pair(3, 4)
>>> p
Pair(3, 4) # __repr__() output
>>> print(p)
(3, 4) # __str__() output
>>>
```

æĹŚäžňåIJÍè£ŹéĞŇè£ŸæijŤçď'žäžĘåIJÍæăijåijŔåŇŰçŽĎæŮűåĂŹæĂŐæăůä¡£çŤÍäÿ■åŘŇçŽĎå■Ůçňęäÿ çĽźåĹńæİěèőšïijŇ!r æăijåijŔåŇŰäžčçăĄæŇĞæŸŐè¿ŞåĞžä¡£çŤÍ __repr__() æİěäžčæŻ£éżŸèőď'çŽĎ __str__() ãĂĆ ä¡ăåŔŕäžěçŤÍåĽ■éÍćçŽĎçśźæİěèŕŢçİĂæţŃèŕŢäÿŇïijŽ

```
>>> p = Pair(3, 4)
>>> print('p is {0!r}'.format(p))
p is Pair(3, 4)
>>> print('p is {0}'.format(p))
p is (3, 4)
>>>
```

èőÍèőž

èĞłåőŽäźĽ __repr__() åŠŇ __str__() éĂŽåÿÿæŸŕå¿Ĺåě¡çŽĎäźăæČŕïijŇåŻääÿžåőČèČ¡çőĂåŇŰä¿ŃåęĆïijŇåęĆæđIJäžĚäžĚåŔłæŸŕæĽŞå■řè¿ŞåĞžæĹŰæŮěå£Ůè¿ŞåĞžæ§Răÿłåőđä¿ŃïijŇéĆčäźĹçÍŃåžŔåŚ

__repr__() çŤ§æĹŘçŽĎæŰĞæIJňå■ŮçňęäÿšæăĞåĞĘåĄŽæşŢæŸŕéIJĂèęĄèőľ eval(repr(x)) == x äÿžçIJ§ãĂĆ åęĆæđIJåőđåIJÍäÿ■èČ¡è£ŹæäůåŇŰåĄŽïijŇåžŤèŕěåĹŻåžžäÿĂäÿłæIJL < åŠŇ > æŃňèţůæİěãĂĆærŤåęĆïijŽ

```
>>> f = open('file.dat')
>>> f
<_io.TextIOWrapper name='file.dat' mode='r' encoding='UTF-8'>
>>>
```

åęĆæđIJ __str__() æšąæIJL'èćńåőŽäźĽïijŇéĆčäźĹåřšäijŽä¡£çŤÍ __repr__() æİěäžčæŻ£è¿ŞåĞžãĂĆ

äÿŁéÍćçŽĎ format() æŰźæşŢçŽĎä¡£çŤÍçIJŃäÿŁåŐžå¿ĹæIJL'èűčïijŇæăijåijŔåŇŰäžčçăĄ {0.x} åŕźåžŤçŽĎæŸŕçňň1äÿłåŔĆæŢřçŽĎxåšđæĂğãĂĆ åŽäæ■d'ïijŇåIJÍäÿŃéÍćçŽĎåĞ¡æŢřäÿ■ïijŇ0åőđéŹ self æIJňèžńïijŽ

```
def __repr__(self):
    return 'Pair({0.x!r}, {0.y!r})'.format(self)
```

ä¡IJäÿžè£Źçğ■åőđçŐřçŽĎäÿĂäÿłæŻ£äžčïijŇä¡ăäź§åŔŕäžěä¡£çŤÍ % æŞ■ä¡IJçňęïijŇåřšåČŔäÿŃéÍćè£ŹæăůïijŽ

```
def __repr__(self):
    return 'Pair(%r, %r)' % (self.x, self.y)
```

8.2 èĞłåőŽäźĽå■Ůçňęäÿšç ŽĎæăijåijŔåŇŰ

éŮőécŸ

ä¡ăæČşéĂŽè£Ğ format() åĞ¡æŢřåŠŇå■Ůçňęäÿšæ ŰźæşŢä¡£å¿ŮäÿĂäÿłåŕźèśąèČ¡æŤŕæŇĄèĞłåőŽäźI

èğčåĘşæŰźæąĹ

äÿžäžĘèĞłåőŽäźĽå■Ůçňęäÿšç ŽĎæăijåijŔåŇŰïijŇæĹŚäżňéIJĂèęĄåIJĺçśžäÿŁéĺćåőŽäźĽ __format__() æŰźæşŢãĂĆä¿ŃåęĆïijŽ

```
_formats = {
    'ymd' : '{d.year}-{d.month}-{d.day}',
    'mdy' : '{d.month}/{d.day}/{d.year}',
    'dmy' : '{d.day}/{d.month}/{d.year}'
    }

class Date:
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day

    def __format__(self, code):
        if code == '':
            code = 'ymd'
        fmt = _formats[code]
        return fmt.format(d=self)
```

çŐřåIJĺ Date çśžçŽĎåőđä¿ŃåŔŕäżěæŤŕæŇĄæăijåijŔåŇŰæŞ■ä¡IJäžĘïijŇåęĆåŘŇäÿŃéĺćè£ŹæăůïijŽ

```
>>> d = Date(2012, 12, 21)
>>> format(d)
'2012-12-21'
>>> format(d, 'mdy')
'12/21/2012'
>>> 'The date is {:ymd}'.format(d)
'The date is 2012-12-21'
>>> 'The date is {:mdy}'.format(d)
'The date is 12/21/2012'
>>>
```

èőĺèőž

__format__() æŰźæşŢçżŹPythonçŽĎå■ŮçňęäÿšæăijåijŔåŇŰåŁ§èČ¡æŔŘä¿ŻäžĘäÿĂäÿłéŠľå■ŘãĂ
è£ŹéĞŇéIJĂèęAçİĂéĞ■åijžèřČçŽĎæŸŕæăijåijŔåŇŰäžčçăAçŽĎèğčæđŘåůëä¡IJåőŇåĚĺçŤśçśżèĞłåůśåEşåőŽ
ä¿ŃåęĆïijŇåŔĆèĂČäÿŃéİćæİëèĞł datetime æĺąåİŮäÿ■çŽĎäžčçĂąïijŽ

```
>>> from datetime import date
>>> d = date(2012, 12, 21)
>>> format(d)
'2012-12-21'
>>> format(d,'%A, %B %d, %Y')
'Friday, December 21, 2012'
>>> 'The end is {:%d %b %Y}. Goodbye'.format(d)
'The end is 21 Dec 2012. Goodbye'
>>>
```

åŕźäžŐåĘĚç¡őçśżåđŃçŽĎæăijåijŔåŇŰæIJL'äÿĂäžŻæăĞåĞĘçŽĎçžęåőŽãĂĆ
åŔŕäžěåŔĆèĂČ stringæĺąåİŮæŰĞæąč èŕť'æŸŐãĂĆ

8.3 èőľåŕźèśąæŤŕæŇĄäÿŁäÿŃæŰĞçőąçŘĘå■Ŕèőő

éŮőécŸ

ä¡ăæČşèőľä¡ăçŽĎåŕźèśąæŤŕæŇĄäÿŁäÿŃæŰĞçőąçŘĘå■Ŕèőő(withèŕ■åŘě)ãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘèőľäÿĂäÿłåŕźèśąåĚijåőź with èŕ■åŘěïijŇä¡ăéIJĂèęĄåőđçŐř __enter__()
åŠŇ __exit__() æŰźæşŢãĂĆ ä¿ŃåęĆïijŇèĂČèŹŚåęĆäÿŃçŽĎäÿĂäÿłçśżïijŇåőČèČ¡äÿžæĹŚäžňåĹŻåžžäÿ

```
from socket import socket, AF_INET, SOCK_STREAM

class LazyConnection:
    def __init__(self, address, family=AF_INET, type=SOCK_STREAM):
        self.address = address
        self.family = family
        self.type = type
        self.sock = None

    def __enter__(self):
        if self.sock is not None:
            raise RuntimeError('Already connected')
        self.sock = socket(self.family, self.type)
        self.sock.connect(self.address)
        return self.sock
```

```
    def __exit__(self, exc_ty, exc_val, tb):
        self.sock.close()
        self.sock = None
```

è£ŹäÿłçśžçŽĎåĚşéŤőçĽźçĆźåIJĺäžŐåőČèąĺçď žäžĘäÿĂäÿłç¡ŚçżIJè£đæŐěïijÑä¡ĘæŸŕåĹİåğŃåŇŰçŽĎa
è£đæŐěçŽĎåżžçńŃåŠŇåĚşéŮ■æŸŕä¡£çŤĺ with èŕ■åŘěèĞłåŁĺåőŇæĹŘçŽĎïijÑä¿ŃåęĆïijŽ

```
from functools import partial

conn = LazyConnection(('www.python.org', 80))
# Connection closed
with conn as s:
    # conn.__enter__() executes: connection open
    s.send(b'GET /index.html HTTP/1.0\r\n')
    s.send(b'Host: www.python.org\r\n')
    s.send(b'\r\n')
    resp = b''.join(iter(partial(s.recv, 8192), b''))
    # conn.__exit__() executes: connection closed
```

èőĺèőž

çijŰåĘŹäÿŁäÿŃæŰĞçőąçŘĘåŹĺçŽĎäÿżèęĄåŐ§çŘĘæŸŕä¡ăçŽĎäžčçăĄäijŽæŤ¿åĹř
with èŕ■åŘěåİŮäÿ■æĽĝèąŇãĂĆ å¡ŞåĞžçŐř with èŕ■åŘěçŽĎæŰűåĂŹïijŇářźèśąçŽĎ
__enter__() æŰźæşŢèćńèğęåŘŚïijŇ åőČè£ŤåŽđçŽĎåĂij(åęĆæđIJæIJĽçŽĎèŕÍ)äijŽèćńètŃåĂijçżŹ
as åčřæŸŐçŽĎåŘŸéĞŘãĂĆçĎűåŘŐïijŇwith èŕ■åŘěåİŮéĞŃéĺćçŽĎäžčçĄåijĂåğŃæĽĝèąŇãĂĆ
æIJĂåŘŐïijŇ__exit__() æŰźæşŢèćńèğęåŘŚè£ŽèąŇæÿĚçŘĘåůěä¡IJãĂĆ

äÿ■çőą with äžčçĄåİŮäÿ■åŘŚçŤ§äžĂäźĹïijŇäÿŁéİćçŽĎæŐğåĹűæţĄéČ¡äijŽæĽĝèąŇåőŃïijŇåřśçőŰ
äžŃåőđäÿŁïijŇ__exit__() æŰźæşŢçŽĎçňňäÿĽäÿłåŘĆæŢřåŃĚåŘńäžĘåijĆåÿÿçśžåđŃãĂĄåijĆåÿÿåĂijåŚ
__exit__() æŰźæşŢèČ¡èĞłåůśåĘşåőŽæĂŐæăůåĹ'çŤĺè£ŹäÿłåijĆåÿÿä£qæĄŕïijŇæĹŰèĂĚå£¡çŢěåőČåźűě
åęĆæđIJ __exit__() è£ŤåŽđ True ïijŃéĆčäźĹåijĆåÿÿäijŽèćńæÿĚçľžïijŇářśåě¡åČŘäžĂäźĹéČ¡æšąåŘŚçŤ
with èŕ■åŘěåŘŐéİćçŽĎçĺŃäžŘçżğçż■åIJĺæ■čåÿÿæĽĝèąŇãĂĆ

è£ŸæIJĽäÿĂäÿłçżĘèŁĆéŮőécŸåřśæŸŕ LazyConnection
çśžæŸŕåŘęåĚĄèŏÿåď ŽäÿŁ with èŕ■åŘěæİěåţŇåěŮä¡£çŤĺè£đæŐěãĂĆ
å¿ĹæŸ¿çĎűïijŇäÿŁéİćçŽĎåőŽäźĽäÿ■äÿĂæňąåŘŁèČ¡åĚĄèŏÿäÿĂäÿłsocketè£đæŐěïijŇåęĆæđIJæ■čåIJĺä¡£çŢ
with èŕ■åŘěïijŇ åřśäijŽäžğçŤ§äÿĂäÿłåijĆåÿÿäžĘãĂĆäÿ■è£Ğä¡ăåŘŕäžěåČŘäÿŃéİćè£ŹæăůäŁőæŤźäÿŃäÿŁé

```
from socket import socket, AF_INET, SOCK_STREAM

class LazyConnection:
    def __init__(self, address, family=AF_INET, type=SOCK_STREAM):
        self.address = address
        self.family = family
        self.type = type
        self.connections = []

    def __enter__(self):
        sock = socket(self.family, self.type)
```

```
        sock.connect(self.address)
        self.connections.append(sock)
        return sock

    def __exit__(self, exc_ty, exc_val, tb):
        self.connections.pop().close()

# Example use
from functools import partial

conn = LazyConnection(('www.python.org', 80))
with conn as s1:
    pass
    with conn as s2:
        pass
        # s1 and s2 are independent sockets
```

åIJÍçňňäžŇäÿłçĽĺæIJňäÿ■ïijŇLazyConnection çśžåŔŕäžěèćńçIJŃåĄŽæŸŕæ§Řäÿłè£đæŐěåůěåŐĆã
æŕŔæňą __enter__() æŰźæşŢæĽ'ğèąŇçŽĎæŮűåĂŹïijŇåőČåd'■åĹűåĹŻåžžäÿĂäÿłæŰřçŽĎè£đæŐěåźűåì
__exit__() æŰźæşŢçőĂå■ŢçŽĎäžŐæăĹäÿ■åijźåĞžæIJĂåŘŐäÿĂäÿłè£đæŐěåźűåĚşéÙ■åőČãĂĆ
è£ŹéĞŇçĺ■å¿õæIJĽçĆźéŽ¿çŘĘèğčïijŇäÿ■è£ĞåőČèČ¡åĚĄèőÿåţŇåěŮä¡£çŢÍ with
èŕ■åŘěåĹŻåžžåd'Žäÿłè£đæŐěïijŇåřśåęĆäÿŁéİćæijŢçd'žçŽĎéĆčæăůãĂĆ

åIJĺéIJĂèęĄçőąçŘĘäÿĂäžŻèţĎæžŘæŕŤåęĆæŰĞäżűãĂĄç¡ŚçżIJè£đæŐěåŠŇéŤĄçŽĎçijŰçĺŃçŐŕåćČäÿ■
è£ŹäžŻèţĎæžŘçŽĎäÿĂäÿłäÿżèęĄçĽźå¿ĄæŸŕåőČäżňå£ĚéążèćńæĽŃåŁĺçŽĎåĚşéÙ■æĹŰéĞŁæŤ¿æİěçąőä£
ä¿ŃåęĆïijŇåęĆæđIJä¡ăèŕůæśĆäžĘäÿĂäÿłéŤĄïijŇéĆčäźĹä¡ăå£Ěéążçąőä£İäźŃåŘŐéĞŁæŤ¿äžĘåőČïijŇåŘęåĹ
éĂŽè£ĞåőđçŐř __enter__() åŠŇ __exit__() æŰźæşŢåźűä¡£çŢÍ with
èŕ■åŘěåŕŕäžěå¿Ĺåőźæ ŸŞçŽĎéĄ£åĚ■è£ŹäžŻéŮőécŸïijŇ åŻääÿž __exit__()
æŰźæşŢåŕŕäžěèőľä¡ăæŮăéIJĂæŃĚå£Čè£ŹäžŻäžĘãĂĆ

åIJĺ contextmanager æĺąåİŮäÿ■æIJĽäÿĂäÿłæăĞåĞĘçŽĎäÿŁäÿŃæŰĞçőąçŘĘæŰźæąĹæĺąæİ£ïijŇå
åŘŇæŮűåIJĺ12.6åřŔèŁĆäÿ■è£ŸæIJĽäÿĂäÿłåŕźæIJňèŁĆçd'žä¿ŃçĺŃåžŘçŽĎçž£çĺŃåőĽåĚĺçŽĎä£őæŤźçĽĹ

8.4 åĹŻåžžåď'ğéĞŔåŕźèśąæŮűèŁĆçIJĄåĘĚå■ŸæŰźæşŢ

éŮőécŸ

ä¡ăçŽĎçĺŃåžŔèęĄåĹŻåžžåď'ğéĞŔ(åŔŕèČ¡äÿŁçŹ¿äÿĞ)çŽĎåŕźèśąïijŇåŕijèĞť'å■ăçŤĺå¿Ĺåď'ğçŽĎåĘĚå■Ÿ

èğčåĘşæŰźæąĹ

åŕźäžŐäÿżèęĄæŸŕçŤĺæİěå¡ŞæĹŘçőĂå■ŢçŽĎæŢřæ■őçżŞædĎçŽĎçśżèĂŇèĺĂïijŇä¡ååŔŕäžěéĂŽè£ĞçżŹ
__slots__ åśđæĂğæİěædĄåď'ğçŽĎåĞŘåřŚåőđä¿ŃæL'Ăå■ăçŽĎåĘĚå■ŸãĂĆæŕŤåęĆïijŽ

```
class Date:
    __slots__ = ['year', 'month', 'day']
```

```
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day
```

å¡Şä¡ăåőŽäźĽ' __slots__ åŘŐïijŇPythonåřśäijŽäÿžåőđä¿Ńä¡£çŤĺäÿĂçğ■æŻť'åŁăçť'ğåĞŚçŽĎåĘĚéČ
åőđä¿ŃéĂŽè£ĞäÿĂäÿłå¿ĹåřŔçŽĎåŽžåőŽåđ'ğåřŔçŽĎæŢřçżĎæİěæđĎåżžïijŇèĂŇäÿ■æŸŕäÿžæŕŔäÿłåőđä¿Ń
åIJÍ __slots__ äÿ■åĹŮåĞžçŽĎåśđæĂğåŘ■åIJĺåĘĚéČĺèćńæŸăåřĎåĹřè£ŹäÿłæŢřçżĎçŽĎæŇĞåőŽåřŔæăČ
ä¡£çŤĺslotsäÿĂäÿłäÿ■åě¡çŽĎåIJřæŰźåřśæŸŕæĹŚäžňäÿ■èČ¡åĘ■çżŹåőđä¿ŃæůżåŀăæŰřçŽĎåśđæĂğäžĘïijŇåĹ
__slots__ äÿ■åőŽäźĽçŽĎéĆčäžŽåśđæĂğåŘ■ãĂĆ

èőĺèőž

ä¡£çŤĺslotsåŘŐèŁĆçIJAçŽĎåĘĚå■ŸäijŽèů§å■ŸåĆĺåśđæĂğçŽĎæŢřéĞŔåŠŇçśżåđŃæIJL'åĚşãĂĆ
äÿ■è£ĞïijŇäÿĂèĹňæİěèőšïijŇä¡£çŤĺåĹřçŽĎåĘĚå■ŸæĂżéĞŔåŠŇårĘæŢřæ■őå■ŸåĆĺåIJĺäÿĂäÿłåĚČçżĎäÿ■å
äÿžäžĘçżŹä¡ăäÿĂäÿłçŻť'èğĆèőď'èrĘïijŇåĄĞèő¿ä¡ăäÿ■ä¡£çŤĺslotsçŻť'æŐěå■ŸåĆĺäÿĂäÿłDateåőđä¿ŃïijŇ
åIJĺ64ä¡■çŽĎPythonäÿŁéİćèęĄå■ăçŤĺ428å■ŮèŁĆïijŇèĂŇåęĆæđIJä¡£çŤĺäžĘslotsïijŇåĘĚå■Ÿå■ăçŤĺäÿŃéŹ■
åęĆæđIJçĺŃåžŔäÿ■éIJĂèęĄåŘŇæŮűåĹŻåžžåď'ğéĞŔçŽĎæŮěæIJ§åőđä¿ŃïijŇéĆčäźĹè£ŹäÿłåřśèČ¡æđĄåď'ğ

år¡çőąslotsçIJŃäÿŁåŐżæŸŕäÿĂäÿłå¿ĹæIJL'çŤĺçŽĎçL'źæĂğïijŇå¿Ĺåď'ŽæŮűåĂŹä¡ăè£ŸæŸŕå¿ŮåĞŔårS
PythonçŽĎå¿Ĺåď'ŽçL'źæĂğéČ¡ä¿İèţŰäžŐæŹőéĂŽçŽĎå§žäžŐå■ŮåĚÿçŽĎåőđçŐřãĂĆ
åŔęåď'ŰïijŇåőŽäźĽäžĘslotsåŘŐçŽĎçśżäÿ■åĘ■æŤŕæŇĄäÿĂäžŽæŹőéĂŽçśżçL'źæĂğäžĘïijŇæŕŤåęĆåď'Žçżğ
åď'ğåď'ŽæŢřæČĚåĘţäÿŃïijŇä¡ăåžŤèřěåŔťåIJĺéĆčäžŽçżŔåÿÿèćńä¡£çŤĺåĹřçŽĎçŤĺä¡IJæŢřæ■őçżŞæđĎçŽĎçś
(æŕŤåęĆåIJĺçĺŃåžŔäÿ■éIJĂèęĄåĹŽåžžæ§ŔäÿłçśżçŽĎåĞăçŹ¿äÿĞäÿłåőđä¿Ńårźèśą)ãĂĆ

åĚşäžŐ __slots__ çŽĎäÿĂäÿłåÿÿèğĄèŕŕåŇžæŸŕåőČåŔŕäžěä¡IJäÿžäÿĂäÿłåŕĄèčĚåůěåĚůæİěéŸšæ■ć
år¡çőąä¡£çŤĺslotsåŔŕäžěè¿¿åĹřè£ŹæăůçŽĎçŻőçŽĎïijŇä¡ĘæŸŕè£Źäÿłåźűäÿ■æŸŕåőČçŽĎåĹİèąůãĂĆ
__slots__ æŻť'åď'ŽçŽĎæŸŕçŤĺæİěä¡IJäÿžäÿĂäÿłåĘĚå■ŸäijŸåŇŰåůěåĚůãĂĆ

8.5 åIJĺçśżäÿ■åŕĄèčĚåśđæĂğåŘ■

éŮőécŸ

ä¡ăæČşåŕĄèčĚçśżçŽĎåőđä¿ŃäÿŁéİćçŽĎâĂIJçğĄæIJL'âĂİæŢřæ■őïijŇä¡ĘæŸŕPythonèŕ■èĺĂåźűæšąæIJ

èğčåĘşæŰźæąĹ

PythonçĺŃåžŔåŚŸäÿ■åŐżä¿İèţŰèŕ■èĺĂçL'źæĂğåŐżåŕĄèčĚæŢřæ■őïijŇèĂŇæŸŕéĂŽè£ĞéĄţå¿łäÿĂåőŽ
çňňäÿĂäÿłçžęåőŽæŸŕäżżä¡Ţäžěå■ŢäÿŃåĹŠçž£_åijĂåď'ťçŽĎåŘ■å■ŮéČ¡åžŤèřěæŸŕåĘĚéČĺåőđçŐřãĂĆæŕŤă

```
class A:
    def __init__(self):
        self._internal = 0 # An internal attribute
        self.public = 1 # A public attribute
```

```
    def public_method(self):
        '''
        A public method
        '''
        pass

    def _internal_method(self):
        pass
```

Pythonåźűäÿ■äijŽçIJ§çŽĎéŸżæ■ćåĹńäžžèő£éŮőåĘĚéČĺåŘ■çğřãĂĆä¡ĘæŸŕåęĆæđIJä¡ãè£ŹäźĹåĄŽèĆŕ
åŘŇæŮűè£ŸèęĄæşĺæĎŔåĹřïijŇä¡£çŤĺäÿŇåĹŠçž£åijĂåď'ťçŽĎçžęåőŽåŘŇæăůéĂĆçŤĺäžŐæĺąåİŮåŘ■åŠŇæ
ä¿ŇåęĆïijŇåęĆæđIJä¡ăçIJŇåĹřæ§ŘäÿłæĺąåİŮåŘ■äžěå■ŢäÿŇåĹŠçž£åijĂåď'ť(æŕŤåęĆ_socket)ïijŇéĆčáőČářš
çśżäijijçŽĎïijŇæĺąåİŮçžğåĹńåĞ¡æŢřæŕŤåęĆ sys._getframe()
åIJĺä¡£çŤĺçŽĎæŮűåĂŹářšå¿ŮåŁäåĂ■ářŔå£ČäžĘãĂĆ

ä¡ãè£ŸåŘŕèČ¡äijŽéĄĞåĹřåIJĺçśżåőŽäźL'äÿ■ä¡£çŤĺäÿď'äÿłäÿŇåĹŠçž£(__)åijĂåď'ťçŽĎåŚ¡åŘ■ãĂĆæŕŤå

```
class B:
    def __init__(self):
        self.__private = 0

    def __private_method(self):
        pass

    def public_method(self):
        pass
        self.__private_method()
```

ä¡£çŤĺåŔŇäÿŇåĹŠçž£åijĂåğŇäijŽåŕijèĞť'èő£éŮőåŘ■çğřåŔŸæĹŘåĚűäżŰå¡ćåijŔãĂĆ
æŕŤåęĆïijŇåIJĺåL'■éİćçŽĎçśżBäÿ■ïijŇçğĄæIJL'åśđæĂğäijŽècńåĹĘåĹńéĞ■åŚ¡åŘ■äÿž
_B__private åŠŇ _B__private_method ãĂĆ è£ŹæŮűåĂŹä¡ăåŔŕèČ¡äijŽéŮőè£ŹæăůéĞ■åŚ¡åŘ■çŽĎ

```
class C(B):
    def __init__(self):
        super().__init__()
        self.__private = 1 # Does not override B.__private

    # Does not override B.__private_method()
    def __private_method(self):
        pass
```

è£ŹéĞŇïijŇçğĄæIJL'åŘ■çğř __private åŠŇ __private_method
èćńéĞ■åŚ¡åŘ■äÿž _C__private åŠŇ _C__private_method
ïijŇè£ŹäÿłèůçĹűçśżBäÿ■çŽĎåŘ■çğřæŸŕåőŇåĚĺäÿ■åŔŇçŽĎãĂĆ

èőĺèőž

äÿŁéİćæŔŘåĹřæIJL'äÿď'çğ■äÿ■åŘŇçŽĎçijŰçăĄçžęåőŽ(å■ŢäÿŇåĹŠçž£åŠŇåŔŇäÿŇåĹŠçž£)æİěåŚ¡åŕ
åď'ğåď'ŽæŢřèĂŇèĺĂïijŇä¡ăåžŤèŕéèőľ'ä¡ăçŽĎéİđåĚñåĚśåŘ■çğřäžěå■ŢäÿŇåĹŠçž£åijĂåď'ťãĂĆä¡ĘæŸŕïijŇåę

åźűäÿŤæIJL'äžŻåĘĚéČĺåšđæĂğåžŤèŕěåIJĺå■Řçśżäÿ■éŽŘèŮŔèţůæİěïijŇéĆčäźĹæL'■èĂČèŹŚä¡£çŤĺåŔŇäÿŃ

è£ŸæIJL'äÿĂçĆźèęĄæşĺæĎŔçŽĎæŸŕïijŇæIJL'æŮűåĂŹä¡ăåőŽäźL'çŽĎäÿĂäÿłåŔŸéĞŔåŠŇæ§Řäÿłä£İç

```
lambda_ = 2.0 # Trailing _ to avoid clash with lambda keyword
```

è£ŹéĞŇæĹŚäžňåźűäÿ■ä¡£çŤĺå■ŢäÿŃåĹŠçž£åL'■çijĂçŽĎåŐ§åŻăæŸŕåőČéĄ£åĚ■èŕŕèğčåőČçŽĎä¡£çŤĺ
(åęĆä¡£çŤĺå■ŢäÿŃåĹŠçž£åL'■çijĂçŽĎçŻőçŽĎæŸŕäÿžäžĘéŸšæ■ćåŚ¡åŘ■åĘšçłĄèĂŇäÿ■æŸŕæŇĞæŸŐè£Źä
éĂŽè£Ğä¡£çŤĺå■ŢäÿŃåĹŠçž£åŘŐçijĂåŔŕäžěèğčåĘşè£ŹäÿłéŮőécŸãĂĆ

8.6 åĹŻåžžåŔŕçőąçŘĘçŽĎåšđæĂğ

éŮőécŸ

ä¡ăæČşçżŹæ§Řäÿłåőđä¿ŃattributeåćđåŀăéŹđ'èő£éŮőäÿŐä£őæŤźäźŃåđ'ŰçŽĎåĚűäžŰåđ'ĎçŘĘéĂżè¿Śi

èğčåĘşæŰźæąĹ

èĞłåőŽäźL'æ§ŘäÿłåšđæĂğçŽĎäÿĂçğ■çőĂå■ŢæŰźæşŢæŸŕåřĘåőČåőŽäźL'äÿžäÿĂäÿłpropertyãĂĆ
ä¿ŃåęĆïijŇäÿŃéİćçŽĎäžčçăĄåőŽäźL'äžĘäÿĂäÿłpropertyïijŇåćđåŀăářźäÿĂäÿłåšđæĂğçőĂå■ŢçŽĎçśżåđŃæ

```
class Person:
    def __init__(self, first_name):
        self.first_name = first_name

    # Getter function
    @property
    def first_name(self):
        return self._first_name

    # Setter function
    @first_name.setter
    def first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._first_name = value

    # Deleter function (optional)
    @first_name.deleter
    def first_name(self):
        raise AttributeError("Can't delete attribute")
```

äÿŁè£řäžčçăĄäÿ■æIJL'äÿL'äÿłçŻÿåĘşèĄŢçŽĎæŰźæşŢïijŇè£ŹäÿL'äÿłæŰźæşŢçŽĎåŘ■å■ŮéČ¡å£Ěéążäÿ
çňňäÿĂäÿłæŰźæşŢæŸŕäÿĂäÿł getter åĞ¡æŢřïijŇåőČä¡£å¿Ů first_name
æĹŘäÿžäÿĂäÿłåšđæĂğãĂĆ åĚűäżŰäÿd'äÿłæŰźæşŢçżŹ first_name åšđæĂğæůżåŁäžĘ
setter åŠŇ deleter åĞ¡æŢřãĂĆ éIJĂèęĄåijžèřČçŽĎæŸŕåŔłæIJL'åIJÍ first_name

åśđæĂğèćńåĹŻåžžåŘŐïijŇ åŘŐéİćçŽĎäÿď'äÿłèčĚéěřåŹÍ `@first_name.setter` åŠŇ `@first_name.deleter` æĽ■èČ¡èćńåőŽäźĽãĂĆ

propertyçŽĎäÿĂäÿłåĚşéŤőçĽźå¿ĄæŸŕåőĆçIJŃäÿŁåŐżèů§æŹőéĂŽçŽĎattributeæšąäžĂäźĹäÿď'æăůïijŇä¡ĘæŸŕèő£éŮőåőĆçŽĎæŮűåĂŹäijŽèĞłåŁĺèğęåŔŚ `getter` ãĂĄ`setter` åŠŇ `deleter` æŰźæşŢãĂĆä¿ŃåęĆïijŽ

```
>>> a = Person('Guido')
>>> a.first_name # Calls the getter
'Guido'
>>> a.first_name = 42 # Calls the setter
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "prop.py", line 14, in first_name
        raise TypeError('Expected a string')
TypeError: Expected a string
>>> del a.first_name
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: can`t delete attribute
>>>
```

åIJĺåőđçŐřäÿĂäÿłpropertyçŽĎæŮűåĂŹïijŇåžŢåśĆæŢřæ■ő(åęĆæđIJæIJL'çŽĎèŕİ)äż■çĎűéIJĂèęĄå■ŸåĆåŻăæ■d'ïijŇåIJĺgetåŠŇsetæŰźæşŢäÿ■ïijŇä¡ăäijŽçIJŃåĹřåŕź `_first_name` åśđæĂğçŽĎæŞ■ä¡IJïijŇè£Źäź§æŸŕåőđéŹĚæŢřæ■őä£Ĺå■ŸçŽĎåIJřæŰźãĂĆåŔęåd'ŰïijŇä¡ăåŔŕèČ¡è£ŸäijŽéŮőäÿżäżĂäźĹ `__init__()` æŰźæşŢäÿ■èő¿ç¡őäžĘ `self.first_name` èĂŇäÿ■æŸŕ `self._first_name` ãĂĆåIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇæĹŚäżňåĹŻåžžäÿĂäÿłpropertyçŽĎçŽőçŽĎåřśæŸŕåIJÍèő¿ç¡őattributeçŽĎæŮűåĂŹåŻăæ■d'ïijŇä¡ăåŔŕèČ¡æČşåIJĺåĹĺåğŃåŇŰçŽĎæŮűåĂŹäź§è£ŻèąŇè£Źçğ■çśžåđŃæčĂæ§ěãĂĆéĂŽè£Ğèő¿ç¡ `self.first_name` ïijŇèĞłåŁĺèřĆçŤĺ `setter` æŰźæşŢïijŇè£ŹäÿłæŰźæşŢéĞŇéİćäijŽè£ŻèąŇåŔĆæŢřçŽĎæčĂæ§ěïijŇåŘęåĹŹåřśæŸŕçŻť'æŐèèő£éŮő `self._first_name` äžĘãĂĆ

è£ŸèČ¡åIJĺåůšå■ŸåIJÍçŽĎgetåŠŇsetæŰźæşŢå§žçąĂäÿŁåőŽäźĽ'propertyãĂĆä¿ŃåęĆïijŽ

```
class Person:
    def __init__(self, first_name):
        self.set_first_name(first_name)

    # Getter function
    def get_first_name(self):
        return self._first_name

    # Setter function
    def set_first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._first_name = value

    # Deleter function (optional)
    def del_first_name(self):
```

```
        raise AttributeError("Can't delete attribute")

    # Make a property from existing get/set methods
    name = property(get_first_name, set_first_name, del_first_name)
```

èőÍèőž

äÿĂäÿłpropertyåśđæĂğåĚűåőđåřśæŸŕäÿĂçşżåĹŮçŻÿåĚşçżŚåőŽæŰźæşŢçŽĎéŽĘåŘĹãĂĆåęĆæđIJä¡ăå
åřśäijŽåŘŚçŐřpropertyæIJňèžńçŽĎfgetãĂĄfsetåŠŇfdelåśđæĂğåřśæŸŕçśžéĞŇéİćçŽĎđæŹőéĂŽæŰźæşŢãĂĆ

```
>>> Person.first_name.fget
<function Person.first_name at 0x1006a60e0>
>>> Person.first_name.fset
<function Person.first_name at 0x1006a6170>
>>> Person.first_name.fdel
<function Person.first_name at 0x1006a62e0>
>>>
```

éĂŽåÿÿæİèèőšïijŇä¡ăäÿ■äijŽçŻť æŐèåŔŰèřČçŤĺfgetæĹŰèĂĚfsetïijŇåőČäžňäijŽåIJĺèő£éŰőpropertyçŽ

åŘłæIJL'å¡Şä¡ăçąőåőđéIJĂèęĄåŕźattributeæL'ğèąŇåĚűäžŰéćİåđ'ŰçŽĎđæŞ■ä¡IJçŽĎđæŮűåĂŹæL'■åžŤèŕé
æIJL'æŮűåĂŹäÿĂäžŽäżŐåĚűäžŰçijŰçÍŃèŕ■èÍĂ(æŕŤåęĆJava)è£ĞæİěçŽĎçÍŃåžŔåŚŸæĂżèőđ'äÿžæL'ĂæIJL
æL'Ăäżěäž Űäžňèőđ'äÿžäżčçăĄåžŤèŕěåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
class Person:
    def __init__(self, first_name):
        self.first_name = first_name

    @property
    def first_name(self):
        return self._first_name

    @first_name.setter
    def first_name(self, value):
        self._first_name = value
```

äÿ■èęĄåĘŹè£Źçğ■æšąæIJL'åĄŽäżżä¡ŢåĚűäžŰéćİåđ'ŰæŞ■ä¡IJçŽĎpropertyãĂĆ
éęŰåĚĹïijŇåőČäijŽèől'ä¡ăçŽĎđżčçăĄåŔŸå¿Ůå¿ĹèĞČèĆ£ïijŇåźűäÿŤè£ŸäijŽè£ůæČŠéŸĚèŕżèĂĚãĂĆ
åĚűæňąïijŇåőČè£ŸäijŽèől'ä¡ăçŽĎđçÍŃåžŔè£ŘèąŇètůæİěåŔŸæĚćå¿Ĺåđ'ŽãĂĆ
æIJĂåŘŐïijŇè£ŹæăůçŽĎèő¿èőąåźűæšąæIJL'åÿęæİěäżżä¡ŢçŽĎåě¡åđ'ĐãĂĆ
çL'źåĹńæŸŕå¡Şä¡ääżěåŘŐæČşçżŹæŹőéĂŽattributeèè£éŰőæůżåŁăéćİåđ'ŰçŽĎđ'ĐçŘĘéĂżè¿ŠçŽĎæŮűåĂŹ
ä¡ăåŔŕäžěåřĘåőČåŔŸæĹŘäÿĂäÿłpropertyèĂŇæŮăéIJĂæŤźåŔŸåŐ§æİěçŽĎäžčçăĄãĂĆ
åŽääÿžèő£éŰőattributeçŽĎäžčçăĄè£ŸæŸŕä£ÍæŇĄåŐ§æäůãĂĆ

Propertiesè£ŸæŸŕäÿĂçğ■åőŽäźL'åŁĺæĂĄèőąçőŮattributeçŽĎæŰźæşŢãĂĆ
è£Źçğ■çśžåđŃçŽĎattributesåźűäÿ■äijŽèćńåőđéŹĚçŽĎå■ŸåĆĺïijŇèĂŇæŸŕåIJĺéIJĂèęĄçŽĎæŮűåĂŹèőąçőŮ

```
import math
class Circle:
```

```
    def __init__(self, radius):
        self.radius = radius

    @property
    def area(self):
        return math.pi * self.radius ** 2

    @property
    def diameter(self):
        return self.radius ** 2

    @property
    def perimeter(self):
        return 2 * math.pi * self.radius
```

åIJÍè£ŹéĞŇïijŇæĹŚäžňéĂŽè£Ğä¡£çŤÍpropertiesïijŇåřĘæĽĂæIJL'çŽĎèő£éŮőæŐěåŔčå¡ćåijŔçż§äÿĂèţ
åŕźå■Łå¿ĎãĂĄçŻť'å¿ĎãĂĄåŚÍéŢ£åŠŇéİćçğŕçŽĎèő£éŮőéČ¡æŸŕéĂŽè£ĞåšđæĂğèő£éŮőïijŇåŕšèů§èő£éŮőç
åęĆæđIJäÿ■è£Źæăůå ĄŽçŽĎèŕİïijŇéĆčäźĹåřśèęĄåIJÍäžčçăĄäÿ■æůůåŘĹä¡£çŤÍçŐÅå■ŢåšđæĂğèő£éŮőåŠŇæ
äÿŃéİćæŸŕä¡£çŤÍçŽĎåőđä¿ŃïijŽ

```
>>> c = Circle(4.0)
>>> c.radius
4.0
>>> c.area  # Notice lack of ()
50.26548245743669
>>> c.perimeter  # Notice lack of ()
25.132741228718345
>>>
```

år¡çőąpropertiesåŔŕäžěåőđçŐřäijŸéŻĚçŽĎçijŰçÍŃæŐěåŔčïijŇä¡ĘæIJL'äžŻæŮűåĂŹä¡ăè£ŸæŸŕäijŽæČ

```
>>> p = Person('Guido')
>>> p.get_first_name()
'Guido'
>>> p.set_first_name('Larry')
>>>
```

è£Źçğ■æČĚåĘţçŽĎåĞžçŐřéĂŽåÿÿæŸŕåŻääÿžPythonäžčçăĄèćńéŻĘæĹŘåĹřäÿĂäÿłåď'ğåđŃå§žçąĂáźş
ä¿ŃåęĆïijŇæIJL'åŔŕèČ¡æŸŕäÿĂäÿłPythonçśżåĞĘåď'ĞåŁăåĚěåĹřäÿĂäÿłå§žäžŐè£IJçÍŃè£ĞçÍŃěřČçŤÍçŽĎåđ
è£Źçğ■æČĚåĘţäÿŃïijŇçŻť'æŐëä¡£çŤÍget/setæŰźæşŢ(æŹőéĂŽæŰźæşŢèřČçŤÍ)èĂŇäÿ■æŸŕpropertyæĹŰèőÿ

æIJĂåŘŐäÿĂçĆźïijŇäÿ■èęĄåČŔäÿŃéİćè£ŹæăůåĘŹæIJL'åď'ğéĞŔéĞ■åď'■äžčçăĄçŽĎpropertyåőŽäźL'ï

```
class Person:
    def __init__(self, first_name, last_name):
        self.first_name = first_name
        self.last_name = last_name

    @property
    def first_name(self):
        return self._first_name
```

```
    @first_name.setter
    def first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._first_name = value

    # Repeated property code, but for a different name (bad!)
    @property
    def last_name(self):
        return self._last_name

    @last_name.setter
    def last_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._last_name = value
```

éĞ■åď■äžčçăĄäijŽåŕijèĞťèĞČèĆ£ãĂĄæŸŞåĞžéŤŹåŠŇäÿŚéŹŃçŽĎçĺŃåžŔãĂĆåě¡æűĹæĄŕæŸŕîijŇéĂ
åŔŕäžěåŔĆèĂČ8.9åŠŇ9.21åřŔèŁĆçŽĎåĘĚåőźãĂĆ

8.7 èřČçŤĺçĹűçśżæŰźæşŢ

éůőécŸ

ä¡ăæČşåIJĺå■Řçśżäÿ■èřČçŤĺçĹűçśżçŽĎæ§ŘäÿłåůšçżŕèćńèęĘçŻŰçŽĎæŰźæşŢãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘèřČçŤĺçĹűçśż(èűĚçśż)çŽĎäÿĂäÿłæŰźæşŢïijŇåŕŕäžěä¡£çŤĺ `super()` åĞ¡æŢřîijŇæŕŤåęĆïijŽ

```
class A:
    def spam(self):
        print('A.spam')

class B(A):
    def spam(self):
        print('B.spam')
        super().spam()  # Call parent spam()
```

`super()` åĞ¡æŢřçŽĎäÿĂäÿłåÿÿèğĄçŤĺæşŢæŸŕåIJĺ `__init__()` æŰźæşŢäÿ■çąőä£ĺçĹűçśżèćńæ■čçąőçŽĎåĹİåğŃåŇŰäžĘïijŽ

```
class A:
    def __init__(self):
```

```
        self.x = 0

class B(A):
    def __init__(self):
        super().__init__()
        self.y = 1
```

super() çŽĎåŕęåď ŰäÿĂäÿłåÿÿèğĄçŤĺæşŢåĞžçŐřåIJĺèęEçŻŰPythonçĽźæőŁæŰźæşŢçŽĎäžčçăĄäÿ

```
class Proxy:
    def __init__(self, obj):
        self._obj = obj

    # Delegate attribute lookup to internal obj
    def __getattr__(self, name):
        return getattr(self._obj, name)

    # Delegate attribute assignment
    def __setattr__(self, name, value):
        if name.startswith('_'):
            super().__setattr__(name, value) # Call original __
↪setattr__
        else:
            setattr(self._obj, name, value)
```

åIJĺäÿŁéİćäžčçăĄäÿ■ïijŇ__setattr__() çŽĎåőđçŐřåŇĚåŕńäÿĂäÿłåŕ■å■ŮæčĂæ§ěãĂĆ åęĆæđIJæ§ŘäÿłåśđæĂğåŕ■äžěäÿŇåĹŠçž£(_)åijĂåď't'ïijŇåŕśéĂŽè£Ğ super() èřČçŤĺåŐ§åğŇçŽĎ __setattr__() ïijŇ åŘęåĹŹçŽĎèŕĺåŕśåğŤæť¿çżŹåĘĚéČĺçŽĎäžčçŘĘåŕźèśą self._obj åŐžåď ĎçŘĘãĂĆ è£ŹçIJŃäÿŁåŐżæIJL'çĆźæĎŔæĂİïijŇåŻääÿžåŕśçőŮæšąæIJL'æŸ¿åijŔçŽĎæ super() äż■çĎűåŔŕäžěæIJL'æŢĹçŽĎåůěä¡IJãĂĆ

èőĺèőž

åőđéŹĚäÿŁïijŇåď'gåőűåŕźäžŐåIJĺPythonäÿ■åęĆä¡Ţæ■čçąőä¡£çŤĺ super() åĞ¡æŢřæŹőéĄ■ç§ěäźŃçŤŽåřŠãĂĆ ä¡ăæIJL'æŮűåĂŹäijŽçIJŃåĹřåČŔäÿŃéİćè£ŹæăůçŻť'æŐěěřČçŤĺçĹűçśżç

```
class Base:
    def __init__(self):
        print('Base.__init__')

class A(Base):
    def __init__(self):
        Base.__init__(self)
        print('A.__init__')
```

åř¡çőąåŕźäžŐåď'géČĺåĹEäžčçăĄèĂŇèĺĂè£ŹäźĹåĄŽæšąäžĂäźĹéŮőécŸïijŇä¡ĘæŸřåIJĺæŻť'åď'■æİĆçŽĎ æŕŤåęĆïijŇèĂČèŹŠåęĆäÿŃçŽĎæČĚåEţïijŽ

```
class Base:
    def __init__(self):
        print('Base.__init__')

class A(Base):
    def __init__(self):
        Base.__init__(self)
        print('A.__init__')

class B(Base):
    def __init__(self):
        Base.__init__(self)
        print('B.__init__')

class C(A,B):
    def __init__(self):
        A.__init__(self)
        B.__init__(self)
        print('C.__init__')
```

åęĆæđIJä¡ăè£ŘèąŇè£ŹæőţäžčçăĄåřšäijŽåŔŚçŐř `Base.__init__()` èćńèřČçŤĺäÿď'æňąïijŇåęĆäÿŇæĽĂçď'žïijŽ

```
>>> c = C()
Base.__init__
A.__init__
Base.__init__
B.__init__
C.__init__
>>>
```

åŕŕèČ¡äÿď'æňąèřČçŤĺ `Base.__init__()` æšąäžĂäźĹåİŕåď'ĎïijŇä¡ĘæIJL'æŮűåĂŹå■t'äÿ■æŸŕãĂĆ åŔęäÿĂæŰźéİćïijŇåĄĞèő¿ä¡ăåIJĺäžčçăĄäÿ■æ■ćæĹŘä¡£çŤĺ `super()` ïijŇçžŞæđIJåřšå¿ĹåőŇç¿ŐäžĘïijŽ

```
class Base:
    def __init__(self):
        print('Base.__init__')

class A(Base):
    def __init__(self):
        super().__init__()
        print('A.__init__')

class B(Base):
    def __init__(self):
        super().__init__()
        print('B.__init__')

class C(A,B):
```

```
    def __init__(self):
        super().__init__()  # Only one call to super() here
        print('C.__init__')
```

è£ŘèąŇè£ŹäýłæŰřçĽĹæIJňåŘŐïijŇä¡ăäijŽåŔŚçŐřæŕŔäýł __init__() æŰźæşŢåŔłäijŽèćńèřČçŢĺäýĂæňąäžĘïijŽ

```
>>> c = C()
Base.__init__
B.__init__
A.__init__
C.__init__
>>>
```

äýžäžĘåijĎæÿĚåőČçŽĎåŐ§çŘĘïijŇæĹŚäžňéIJĂèęĄèŁśçĆźæŮűéŮť'ègčéĞŁäýŃPythonæŸŕåęĆä¡Ţåőđ åŕźäžŐä¡ăåőŽäźL'çŽĎæŕŔäýĂäýłçśžïijŇPythonäijŽèőąçőŮåĞžäýĂäýłæL'ĂèřŞçŽĎæŰźæşŢègčæđŘéąžåžŔ(I è£ŹäýłMROåĹŮèąĺåŕśæŸŕäýĂäýłçőĂå■ŢçŽĎæL'ĂæIJL'å§žçśžçŽĎçž£æĂğéąžåžŔèąĺãĂĆä¿ŃåęĆïijŽ

```
>>> C.__mro__
(<class '__main__.C'>, <class '__main__.A'>, <class '__main__.B'>,
<class '__main__.Base'>, <class 'object'>)
>>>
```

äýžäžĘåőđçŐřçżğæL'£ïijŇPythonäijŽåIJÍMROåĹŮèąĺäýŁäžŐåůęåĹřåŘşåijĂåğŃæ§ěæL'¿å§žçśžïijŇçŽt'

èĂŇè£ŹäýłMROåĹŮèąÍçŽĎæđĎéĂăæŸŕéĂŽè£ĞäýĂäýłC3çž£æĂğåŇŰçőŮæşŢæİěåőđçŐřçŽĎãĂĆ æĹŚäžňäý■åŐżæůśçl'ûè£ŹäýłçőŮæşŢçŽĎæŢřå■ęåŐ§çŘĘïijŇåőČåőđéŹĚäýŁářśæŸŕåŘĹåźűæL'ĂæIJL'çĹűçs

- å■ŘçśżäijŽåĚĹäžŐçĹűçśżècńæčĂæ§ě
- åd'ŽäýłçĺűçśżäijŽæăźæ■őåőČäžňåIJĺåĹŮèąĺäý■çŽĎéąžåžŔècńæčĂæ§ě
- åęĆæđIJåŕźäýŇäýĂäýłçśżå■ŸåIJĺäýd'äýłåŘĹæşŢçŽĎéĂL'æŇl'ïijŇéĂL'æŇl'çňňäýĂäýłçĹűçśż

èĂĄåőđèŕt'ïijŇä¡ăæL'ĂèęAç§ěéĄŞçŽĎåřśæŸŕMROåĹŮèąĺäý■çŽĎçśżéąžåžŔäijŽèől'ä¡ăåőŽäźL'çŽĎäżz

å¡Şä¡ää¡£çŢĺ super() åĞ¡æŢřæŮűïijŇPythonäijŽåIJÍMROåĹŮèąĺäýŁçżğçż■æŘIJçt'ćäýŇäýĂäýłçśžãĂ åŘłèęĄæŕŔäýłéĞ■åőŽäźL'çŽĎæŰźæşŢçż§äýĂä¡£çŢĺ super() åźűåŘłèřČçŢĺåőČäýĂæňąïijŇ éĆčäźĹæŐğåĹűæţĄæIJĂçżĹäijŽéĄ■åŐĘåőŇæŢt'äýłM-ROåĹŮèąĺïijŇæŕŔäýłæŰźæşŢäź§åŘłäijŽèćńèřČçŢĺäýĂæňąäĂĆ è£Źäź§æŸŕäýžäžĂäźĹåIJÍçňňäžŇäýłä¿Ńå■Řäý■ä¡ääý■äijŽèřČçŢĺäýd'æňą Base.__init__() çŽĎåŐ§åŽãäĂĆ

super() æIJL'äýłäžd'äžžåŘČæČŁçŽĎåIJřæŰźæŸŕåőČźűäý■äýĂåőŽåŐżæ§ěæL'¿æ§Řäýłçśżå IJÍMRO ä¡ăçŢŽèĞşåŘŕäžěåIJÍäýĂäýłæšąæIJL'çŻt'æŐęĹűçśżçŽĎçśżäý■ä¡£çŢĺåőČãĂĆä¿ŃåęĆïijŇèĂČèŹŠåęĆäýŇè

```
class A:
    def spam(self):
        print('A.spam')
        super().spam()
```

åęĆæđIJä¡ăèŕŢçİĂçŻt'æŐëä¡£çŢĺè£ŹäýłçśżåřšäijŽåĞžéŤŹïijŽ

```
>>> a = A()
>>> a.spam()
A.spam
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "<stdin>", line 4, in spam
AttributeError: 'super' object has no attribute 'spam'
>>>
```

ä¡ĘæŸŕïijŇåęĆæđIJä¡ää¡£çŤĺåď'ŽçżğæL'£çŽĎèŕİçIJŃçIJŃäijŽåŔŚçŤ§äžĂäźĹïijŽ

```
>>> class B:
...     def spam(self):
...         print('B.spam')
...
>>> class C(A,B):
...     pass
...
>>> c = C()
>>> c.spam()
A.spam
B.spam
>>>
```

ä¡ääŔŕäžěçIJŃåĹřåIJĺçśżAäÿ■ä¡£çŤĺ `super().spam()` åŏđéŹĚäÿŁèřČçŤĺçŽĎæŸŕèů§çśżAæŕńæŮăåĚşçşżçŽĎçśżBäÿ■çŽĎ `spam()` æŰźæşŢãĂĆ è£ŹäÿłçŤĺçśżcçŽĎMROåĹŮèąĺářśåŔŕäžěåőŇåĚĺèğčéĞŁæÿĚæěŽäžĘïijŽ

```
>>> C.__mro__
(<class '__main__.C'>, <class '__main__.A'>, <class '__main__.B'>,
<class 'object'>)
>>>
```

åIJĺåőŽäźL'æůůåĚěçśżçŽĎæŮűåĂŹè£Źæăůä¡£çŤĺ `super()` æŸŕå¿ĹæŹőéĄ■çŽĎãĂĆåŔŕäžěåŔĆèĂČ8.13åŠŇ8.18åřŔèŁĆãĂĆ

çĎűèĂŇïijŇçŤśäžŐ `super()` åŔŕèČ¡äijŽèřČçŤĺäÿ■æŸŕä¡ăæČşèęĄçŽĎæŰźæşŢïijŇä¡ääžŤèŕěéĄţå¿łäÿ éęŮåĚĹïijŇçąőä£ÍåIJĺçżğæL'£ä¡Şçşżäÿ■æL'ĂæIJL'çŻÿåŘŇåŘ■å■ŮçŽĎæŰźæşŢæŃěæIJL'åŔŕåĚijåőźçŽĎåŔ è£Źæăůåŕäžěçąőä£Í `super()` èřČçŤĺäÿĂäÿłéİđçŻt'æŐěçĹűçśżæŰźæşŢæŮűäÿ■äijŽåĞžéŤŹãĂĆ åĚűæňąïijŇæIJĂåě¡çąőä£ÍæIJĂéąűåśĆçŽĎçśżæŔŘä¿ŻäžĘè£ŹäÿłæŰźæşŢçŽĎåőđçŐřïijŇè£ŹæăůçŽĎèĺåIJĺ

åIJĺPythonçď¿åŇžäÿ■åŕźäžŐ `super()` çŽĎä¡£çŤĺæIJL'æŮűåĂŹäijŽåijŢæÍěäÿĂäžŻäžL'èőőãĂĆ år¡çőąåęĆæ■d'ïijŇåęĆæđIJäÿĂåĹĞéąžåĹl'çŽĎèŕÍïijŇä¡ääžŤèŕěåIJĺä¡ăæIJĂæŰřäžčçăĄäÿ■ä¡£çŤĺåőČãĂĆ Raymond Hettingeräÿžæ■d'åĘŹäžĘäÿĂçŕĞéÍđåÿÿåě¡çŽĎæŰĞçńă âĂIJPythonâĂŹs super() Considered Super!âĂİ ïijŇ éĂŽè£Ğåď'ğéĞŔçŽĎä¿Ńå■ŘåŘŚæĹŚäžňèğčéĞŁäžĘäÿžäžĂäźĹ `super()` æŸŕæđĄåě¡çŽĎãĂĆ

8.8 å■Řçśżäÿ■æĽľåśŢproperty

éůőécŸ

åIJĺå■Řçśżäÿ■ïijŇä¡ăæČşèęĄæĽľåśŢåőŽäźĽåIJĺçĹűçśżäÿ■çŽĎpropertyçŽĎåŁ§èČ¡ãĂĆ

èğčåĘşæŰźæąĹ

èĂČèŹŚåęĆäÿŃçŽĎäžčçăĄïijŇåőČåőŽäźĽäžĘäÿĂäÿłpropertyïijŽ

```
class Person:
    def __init__(self, name):
        self.name = name

    # Getter function
    @property
    def name(self):
        return self._name

    # Setter function
    @name.setter
    def name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._name = value

    # Deleter function
    @name.deleter
    def name(self):
        raise AttributeError("Can't delete attribute")
```

äÿŃéİćæŸŕäÿĂäÿłçd'žä¿ŃçśżïijŇåőČçżğæĽ£èĞłPersonåźűæĽľåśŢäžĘ name åśđæĂğçŽĎåŁ§èČ¡ïijŽ

```
class SubPerson(Person):
    @property
    def name(self):
        print('Getting name')
        return super().name

    @name.setter
    def name(self, value):
        print('Setting name to', value)
        super(SubPerson, SubPerson).name.__set__(self, value)

    @name.deleter
    def name(self):
```

```
        print('Deleting name')
        super(SubPerson, SubPerson).name.__delete__(self)
```

æŐěäÿŃæİě䡣çŤĺè£ŹäÿłæŰřçśżïijŽ

```
>>> s = SubPerson('Guido')
Setting name to Guido
>>> s.name
Getting name
'Guido'
>>> s.name = 'Larry'
Setting name to Larry
>>> s.name = 42
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 16, in name
        raise TypeError('Expected a string')
TypeError: Expected a string
>>>
```

åęĆædIJä¡ääžĚäžĚåŘłæČşæĽľåśŢpropertyçŽĎæ§ŘäÿĂäÿłæŰźæşŢïijŇéĆčäźĹåŘŕäžěåČŘäÿŃéİćè£Źæ

```
class SubPerson(Person):
    @Person.name.getter
    def name(self):
        print('Getting name')
        return super().name
```

æĹŰèĂĚïijŇä¡ääŘłæČşä£őæŤźsetteræŰźæşŢïijŇářśè£ŹäźĹåĘŹïijŽ

```
class SubPerson(Person):
    @Person.name.setter
    def name(self, value):
        print('Setting name to', value)
        super(SubPerson, SubPerson).name.__set__(self, value)
```

èőÍèőž

åIJÍå■Řçśżäÿ■æĽľåśŢäÿĂäÿłpropertyåŘŕèČ¡äijŽåijŢètůå¿Ĺåď'Žäÿ■æŸŞår§èğL'çŽĎéŮőécŸïijŇ åŻääÿžäÿĂäÿłpropertyåĚűåőđæŸŕ getterãĂĄsetter åŠŇ deleter æŰźæşŢçŽĎéŻĘåŘĹïijŇèĂŇäÿ■æŸŕå■ŢäÿłæŰźæşŢãĂĆ åŻăæ■d'ïijŇä¡Ęä¡ăæĽľåśŢäÿĂäÿłpropertyçŽĎæŮűåĂŹïijŇä¡ăéIJĂèęĄåĚĹçąőåőŽä¡ăæŸŕåŘęèęĄéĞ■æŰřåĉ

åIJÍçňňäÿĂäÿłä¿Ńå■Řäÿ■ïijŇæĽ'ĂæIJL'çŽĎpropertyæŰźæşŢéČ¡èćńéĞ■æŰřåőŽäźL'ãĂĆ åIJÍæŕŔäÿĂäÿłæŰźæşŢäÿ■ïijŇä¡£çŤĺäžĘ super() æİèèřČçŤĺçĹűçśżçŽĎåőđçŐřãĂĆ åIJÍ setter åĞ¡æŢřäÿ■ä¡£çŤĺ super(SubPerson, SubPerson).name.__set__(self, value) çŽĎèŕ■åŘęæŸŕæšąæIJL'éŤŹçŽĎãĂĆ äÿžäžĘåğŤæL'ŸçżŹäźŃåL'■åőŽäźL'çŽĎsetteræŰźæşŢïijŇéIJĂèęĄåřĘæŐğåĹűæİČäijäéĂŠçżŹäźŃåL'■åőŽäź __set__() æŰźæşŢãĂĆ äÿ■è£ĞïijŇèŐůåŘŰè£ŹäÿłæŰźæşŢçŽĎåŤŕäÿĂéĂŤå¿ĎæŸŕä¡£çŤĺçśżåŘŸéĞŔèĂ

è£Źäź§æŸŕäÿžäżĂäźĹæĹŚäżňèęĄä¡£çŤĺ `super(SubPerson, SubPerson)` çŽĎåŐ§åŻããĂĆ

åęĆæđIJä¡ăåŔłæČşéĞ■åőŽäźL'åĚűäÿ■äÿĂäÿłæŰźæşŢïijŇéĆčåŔłä¡£çŤĺ @property æIJňèžńæŸŕäÿ■åđ'§çŽĎãĂĆæŕŤåęĆïijŇäÿŃéİćçŽĎäżčçăĄåřśæŮăæşŢåůěä¡IJïijŽ

```
class SubPerson(Person):
    @property  # Doesn't work
    def name(self):
        print('Getting name')
        return super().name
```

åęĆæđIJä¡ăèŕŢçİĂè£ŘèąŇäijŽåŔŚçŐřsetteråĞ¡æŢřæŢt'äÿłæűĹåđ'śäžĘïijŽ

```
>>> s = SubPerson('Guido')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 5, in __init__
        self.name = name
AttributeError: can't set attribute
>>>
```

ä¡ăåžŤèřěåČŔäźŃåL'■èŕt'è£ĞçŽĎéĆčæăůä£őæŤźäżčçăĄïijŽ

```
class SubPerson(Person):
    @Person.getter
    def name(self):
        print('Getting name')
        return super().name
```

è£ŹäźĹåĘŹåŘŐïijŇpropertyäźŃåL'■åůšçżŔåőŽäźL'è£ĞçŽĎæŰźæşŢäijŽèćńåđ'■åĹűè£ĞæÌěïijŇèĂŇget

```
>>> s = SubPerson('Guido')
>>> s.name
Getting name
'Guido'
>>> s.name = 'Larry'
>>> s.name
Getting name
'Larry'
>>> s.name = 42
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 16, in name
        raise TypeError('Expected a string')
TypeError: Expected a string
>>>
```

åIJĺè£ŹäÿłçĽźåĹńçŽĎèğčåĘşæŰźæąĹäÿ■ïijŇæĹŚäżňæšąåŁđæşŢä¡£çŤĺæŻť'åŁäéĂŽçŤĺçŽĎæŰźåijŔåČ `Person` çśżåŘ■ãĂĆ åęĆæđIJä¡ăäÿ■ç§ěéĄŞåĽřåžŢæŸŕåŞłäÿłå§žçśżåőŽäźL'äžEpropertyïijŇ éČčä¡ăåŔłèČ¡éĂŽè£ĞéĞ■æŰřåőŽäźL'æL'ĂæIJL'propertyåźűä¡£çŤĺ `super()` æÌěårĘæŐğåĹűæİČäijäéĂŠçżŹåL'■éİćçŽĎåőđçŐřãĂĆ

åĂijçŽĎæşĺæĎŔçŽĎæŸŕäÿŁéİćæijŤçd'žçŽĎçňňäÿĂçğ■æŁĂæIJŕè£ŸåŔŕäžèèćńçŤĺæİèæL'l'åśŢäÿĂäÿła

```
# A descriptor
class String:
    def __init__(self, name):
        self.name = name

    def __get__(self, instance, cls):
        if instance is None:
            return self
        return instance.__dict__[self.name]

    def __set__(self, instance, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        instance.__dict__[self.name] = value

# A class with a descriptor
class Person:
    name = String('name')

    def __init__(self, name):
        self.name = name

# Extending a descriptor with a property
class SubPerson(Person):
    @property
    def name(self):
        print('Getting name')
        return super().name

    @name.setter
    def name(self, value):
        print('Setting name to', value)
        super(SubPerson, SubPerson).name.__set__(self, value)

    @name.deleter
    def name(self):
        print('Deleting name')
        super(SubPerson, SubPerson).name.__delete__(self)
```

æIJĂåŘŐåĂijçŽĎæşĺæĎŔçŽĎæŸŕïijŇèŕżåĹŕè£ŹéĞŇæŮűïijŇä¡ăåžŤèŕěäijŽåŔŚçŐŕå■ŘçśżåŇŰ
`setter` åŠŇ `deleter` æŰźæşŢåĚűåőđæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆ
è£ŹéĞŇæijŤçd'žçŽĎèğčåĘşæŰźæąĹåŘŇæăůéĂĆçŤĺïijŇä¡ĘæŸŕåIJÍ PythonçŽĎissueéąţéİć
æŁěåŚŁçŽĎäÿĂäÿłbugïijŇæĹŰèőÿäijŽä¡£å¿ŮåřĘæİèçŽĎPythonçL'ĹæIJñäÿ■åĞžçŐřäÿĂäÿłæŻť'åŁăçőĂæť

8.9 åĹŻåžžæŰřçŽĎçšžæĹŰåőđä¿ŃåšđæĂğ

éŮőécŸ

ä¡ăæČşåĹŻåžžäÿĂäÿłæŰřçŽĎæŃěæIJĽäÿĂäžŻécĺåď´ŰåŁ§èČ¡çŽĎåőđä¿ŃåšđæĂğçšžåđŃïijŃæŕŤåęÇç

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşåĹŻåžžäÿĂäÿłåĚĺæŰřçŽĎåőđä¿ŃåšđæĂğïijŃåŕŕäžěéĂŽè£ĞäÿĂäÿłæŔŕè£řåŹĺçšžçŽĎ

```
# Descriptor attribute for an integer type-checked attribute
class Integer:
    def __init__(self, name):
        self.name = name

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return instance.__dict__[self.name]

    def __set__(self, instance, value):
        if not isinstance(value, int):
            raise TypeError('Expected an int')
        instance.__dict__[self.name] = value

    def __delete__(self, instance):
        del instance.__dict__[self.name]
```

äÿĂäÿłæŔŕè£řåŹĺåřśæŸŕäÿĂäÿłåőđçŐřäžĘäÿL'äÿłæäÿå£ČçŽĎåšđæĂğèő£éŮőæŞ■ä¡IJ(get, set, delete)çŽĎçšžïijŃ åĹĘåĹńäÿž `__get__()` ãĂĄ`__set__()` åŠŇ `__delete__()` è£ŹäÿL'äÿłçL'źæőŁçŽĎæŰźæşŢãĂĆ è£ŹäžŻæŰźæşŢæŐěåŔŮäÿĂäÿłåőđä¿Ńä¡IJäÿžè¿ŞåĚěïijŃäźŃåŘŐçŽÿåžŤçŽĎæŞ■ä¡IJåőđä¿ŃåžŢåśĆçŽĎå■

äÿžäžĘä¡£çŤĺäÿĂäÿłæŔŕè£řåŹĺïijŃéIJĂåřĘè£ŹäÿłæŔŕè£řåŹĺçŽĎåőđä¿Ńä¡IJäÿžçšžåšđæĂğæŤ¿åĹřäÿĂ

```
class Point:
    x = Integer('x')
    y = Integer('y')

    def __init__(self, x, y):
        self.x = x
        self.y = y
```

å¡Şä¡ăè£ŹæăůåĄŽåŘŐïijŃæL'ĂæIJL'åŕźæŔŕè£řåŹĺåšđæĂğ(æŕŤåęĆxæĹŰy)çŽĎèő£éŮőäijŽècń `__get__()` ãĂĄ`__set__()` åŠŇ `__delete__()` æŰźæşŢæ■ŢèŐůåĽřãĂĆä¿ŃåęĆïijŽ

```
>>> p = Point(2, 3)
>>> p.x # Calls Point.x.__get__(p,Point)
2
>>> p.y = 5 # Calls Point.y.__set__(p, 5)
>>> p.x = 2.3 # Calls Point.x.__set__(p, 2.3)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "descrip.py", line 12, in __set__
        raise TypeError('Expected an int')
TypeError: Expected an int
>>>
```

ä¡IJäÿžè¿ŞåĚěïijŇæŔŘè£řåŹĺçŽĎæŕŘäÿĂäÿłæŰźæşŢäijŽæŐěåŘŮäÿĂäÿłæŞ■ä¡IJåőđä¿ŃãĂĆ äÿžäžĘåőđçŐřèŕůæśĆæŞ■ä¡IJïijŇäijŽçŻÿåžŤçŽĎæŞ■ä¡IJåőđä¿ŃåžŢåśĆçŽĎå■ŮåĚÿ(__dict__åśđæĂğ)ãĂĆ æŔŘè£řåŹĺçŽĎ `self.name` åśđæĂğå■ŸåĆĺäžĘåIJĺåőđä¿Ńå■ŮåĚÿäÿ■èćńåőđéŹĚä¡£çŤĺåĹřçŽĎkeyãĂĆ

èőĺèőž

æŔŘè£řåŹĺåŘřåőđçŐřåd'ğéČĺåĹĘPythonçśžçL'źæĂğäÿ■çŽĎåžŢåśĆé■ŤæşŢïijŇ åŇĚæŃň `@classmethod` ãĂĄ`@staticmethod` ãĂĄ`@property` ïijŇçŤŽèĞşæŸŕ `__slots__` çL'źæĂğãĂĆ

éĂŽè£ĞåőŽäźL'äÿĂäÿłæŔŘè£řåŹĺïijŇä¡ăåŘŕäžěåIJĺåžŢåśĆæ■ŢèŐůæäÿå£ČçŽĎåőđä¿ŃæŞ■ä¡IJ(get, set, delete)ïijŇåźűäÿŤåŘŕåőŇåĚĺèĞłåőŽäźL'åőČäžňçŽĎèąŇäÿžĂĆ è£ŹæŸŕäÿĂäÿłåijžåd'ğçŽĎåůěåĚůïijŇæIJL'äžĘåőČä¡ăåŘŕäžěåőđçŐřå¿Ĺåd'ŽénŸçžğåŁ§èČ¡ïijŇåźűäÿŤåőČä

æŔŘè£řåŹĺçŽĎäÿĂäÿłæŕŤè¿ČåŽřæČŚçŽĎåIJřæŰźæŸŕåőČåŘłèČ¡åIJĺçśžçžğåĹńèćńåőŽäźL'ïijŇèĂŇäÿ■

```
# Does NOT work
class Point:
    def __init__(self, x, y):
        self.x = Integer('x') # No! Must be a class variable
        self.y = Integer('y')
        self.x = x
        self.y = y
```

åŘŇæŮűïijŇ`__get__()` æŰźæşŢåőđçŐřęţůæİěæŕŤçIJŃäÿŁåŐżèęĄåd'■æİĆå¿Ůåd'ŽïijŽ

```
# Descriptor attribute for an integer type-checked attribute
class Integer:

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return instance.__dict__[self.name]
```

`__get__()` çIJŃäÿŁåŐżæIJL'çĆźåd'■æİĆçŽĎåŐ§åŽää¡ŠçżŞäžŐåőđä¿ŃåŔŸéĞŔåŠŇçśžåŔŸéĞŔçŽĎ åęĆæđIJäÿĂäÿłæŔŘè£řåŹĺèćńå¡ŞåĄŽäÿĂäÿłçśžåŔŸéĞŔæİèèő£éŰőïijŇéĆčäźĹ `instance` åŘĆæŢřèćńèő¿ç¡őæĹŘ `None` ãĂĆ è£ŹçğæČĚåĘţäÿŃïijŇæăĞåĞĘåĄŽæşŢåřśæŸŕçőĂå■ŢçŽĎè£ŤåŽđè£Źä

```
>>> p = Point(2,3)
>>> p.x # Calls Point.x.__get__(p, Point)
2
>>> Point.x # Calls Point.x.__get__(None, Point)
<__main__.Integer object at 0x100671890>
>>>
```

æŔŘè£řåŹĺéĂŽåÿÿæŸŕéĆčäžŻä¡£çŤĺåĹřèčĚéěřåŹĺæĹŮåĚČçśżçŽĎåď˙ğåđŃæąĘæđűäÿ■çŽĎäÿĂäÿłçżĎ
äÿ¿äÿłä¿Ńå■ŘïijŇäÿŃéÍćæŸŕäÿĂäžŻæŻť´énŸçžğçŽĎå§žäžŐæŔŘè£řåŹĺçŽĎäżčçăĄïijŇåźűæűĽ˙åŔŁåĹřäÿĂ

```
# Descriptor for a type-checked attribute
class Typed:
    def __init__(self, name, expected_type):
        self.name = name
        self.expected_type = expected_type
    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return instance.__dict__[self.name]

    def __set__(self, instance, value):
        if not isinstance(value, self.expected_type):
            raise TypeError('Expected ' + str(self.expected_type))
        instance.__dict__[self.name] = value
    def __delete__(self, instance):
        del instance.__dict__[self.name]

# Class decorator that applies it to selected attributes
def typeassert(**kwargs):
    def decorate(cls):
        for name, expected_type in kwargs.items():
            # Attach a Typed descriptor to the class
            setattr(cls, name, Typed(name, expected_type))
        return cls
    return decorate

# Example use
@typeassert(name=str, shares=int, price=float)
class Stock:
    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

æIJĂåŘŐèęĄæŇĞåĞžçŽĎäÿĂçĆźæŸŕïijŇåęĆæđIJä¡ăåŔłæŸŕæČşçőĂå■ŢçŽĎèĞłåőŽäźĽ˙æ§Răÿłçśżçž
è£Źçğ■æČĚåĘţäÿŃä¡£çŤĺ8.6åřŔèŁĆäżŃçż■çŽĎpropertyæŁĂæIJŕäijŽæŻť˙åŁăåőźæŸŞãĂĆ
å¡ŞçĺŇåžŔäÿ■æIJL˙å¿Ĺåď˙ŽéĞ■åď˙■äžčçăĄçŽĎæŮűåĂŹæŔŘè£řåŹĺåřśå¿ĹæIJL˙çŤĺäžĘ
(æŕŤåęĆä¡ăæČşåIJĺä¡ăäžčçăĄçŽĎå¿Ĺåď˙ŽåIJřæŰźä¡£çŤĺæŔŘè£řåŹĺæŔŘä¿ŻçŽĎåŁ§èČ¡æĹŮèĂĚåřĘåőČä¡Ł

8.10 ä¡£çŤĺåžűè£§èőąçőŮåšđæĂğ

éŮőécŸ

ä¡ăæČşåřĘäÿĂäÿłåŔłèŕžåšđæĂğåőŽäźĽ'æĹŘäÿĂäÿłpropertyïijŇåźűäÿŤåŔłåIJĺèő£éŮőçŽĎæŮűåĂŹæĽ'ı
ä¡ĘæŸŕäÿĂæŮęèćńèő£éŮőåŘŐïijŇä¡ăåÿŇæIJŻçżŞæđIJåĂijèćńçijŞå■ŸèţůæİěïijŇäÿ■çŤĺærŔæňąéČ¡åŐżèőą

èğčåĘşæŰźæąĹ

åőŽäźĽäÿĂäÿłåžűè£§åšđæĂğçŽĎäÿĂçğ■énŸæŢĹæŰźæşŢæŸŕéĂŽè£Ğä¡£çŤĺäÿĂäÿłæŔŔè£řåŹĺçśżïijŇ

```
class lazyproperty:
    def __init__(self, func):
        self.func = func

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            value = self.func(instance)
            setattr(instance, self.func.__name__, value)
            return value
```

ä¡ăéIJĂèęĄåČŔäÿŃéİćè£ŹæăůåIJĺäÿĂäÿłçśżäÿ■ä¡£çŤĺåőČïijŽ

```
import math

class Circle:
    def __init__(self, radius):
        self.radius = radius

    @lazyproperty
    def area(self):
        print('Computing area')
        return math.pi * self.radius ** 2

    @lazyproperty
    def perimeter(self):
        print('Computing perimeter')
        return 2 * math.pi * self.radius
```

äÿŃéİćåIJĺäÿĂäÿłäžd'äžŠçŐŕåćČäÿ■æijŤçd'žåőČçŽĎä¡£çŤĺïijŽ

```
>>> c = Circle(4.0)
>>> c.radius
4.0
>>> c.area
Computing area
```

```
50.26548245743669
>>> c.area
50.26548245743669
>>> c.perimeter
Computing perimeter
25.132741228718345
>>> c.perimeter
25.132741228718345
>>>
```

äžŤçżĘèğĆår§ä¡ääijŽåŔŚçŐřæűĹæĄŕ `Computing area` åŠŇ `Computing perimeter` äžĚäžĚåĞžçŐřäÿĂæňąãĂĆ

èőĺèőž

å¿Ĺåď˙ŽæŮűåĂŹïijŇæđĎéĂääÿĂäÿłåžűè£§èőąçőŮåšđæĂğçŽĎäÿżèęĄçŽőçŽĎæŸŕäÿžäžĘæŔŘå■ĞæĂ
ä¿ŃåęĆïijŇä¡ääŔŕäżěéĄ£åĚ■èőąçőŮè£ŹäžŻåšđæĂğåĂijïijŇéŹď˙éÍđä¡ăçIJ§çŽĎéIJĂèęĄåőČäžñãĂĆ
è£ŹéĞŇæijŤçď˙žçŽĎæŰźæąĹářśæŸŕçŤĺæÍěåőđçŐřè£ŹæäůçŽĎæŢĹæđIJçŽĎïijŇ
åŘłäÿ■è£ĞåőČæŸŕéĂŽè£ĞäżěéÍđåÿÿéńŸæŢĹçŽĎæŰźåijŔä¡£çŤĺæŔŔè£řåŹĺçŽĎäÿĂäÿłçš¿åęŹçL˙źæĂğæÍě

æ■čåęĆåIJĺåĚűäżŮåřŔèŁĆ(åęĆ8.9åřŔèŁĆ)æL˙Ăèőšç ŽĎéĆčæäůïijŇå¡ŞäÿĂäÿłæŔŔè£řåŹĺèćńæŤ¿åĚěäÿ
æŕŔæňąèő£éŮőåšđæĂğæŮűåőČçŽĎ `__get__()` ãĂĄ `__set__()` åŠŇ `__delete__()` æŰźæşŢåŕšäijŽèćńèğęåŔŚãĂĆ äÿ■è£ĞïijŇåęĆæđIJäÿĂäÿłæŔŔè£řåŹĺäžĚäžĚåŘłåőŽäźL˙äžĘäÿĂäÿł `__get__()` æŰźæşŢçŽĎèŕÍïijŇåőČæŕŤéĂŽåÿÿçŽĎåĚůæIJL˙æŻť˙åijśçŽĎçżŠåőŽãĂĆ çL˙źåĹńåIJŕïijŇåŘłæIJL˙å¡Şèćńèő£éŮőåšđæĂğäÿ■åIJĺåőđä¿ŃåžŢåśĆçŽĎå■ŮåĚÿäÿ■æŮű `__get__()` æŰźæşŢæL˙■äijŽèćńèğęåŔŚãĂĆ

`lazyproperty` çśżåĹl˙çŤĺè£ŹäÿĂçĆźïijŇä¡£çŤĺ `__get__()` æŰźæşŢåIJĺåőđä¿ŃäÿĂ■å■ŸåĆĺèőąçőŮåĞžæĬęçŽĎåÄijïijŇ è£Źäÿłåőđä¿Ńä¡£çŤĺçŻÿåŘŇçŽĎåŘ■å■Ůä¡IJäÿžä
è£ŹæåůäÿĂæĬěïijŇçżŞæđIJåĂijèćńå■ŸåĆĺåIJĺåőđä¿Ńå■ŮåĚÿäÿ■åźűäÿŤäžěåŘŐårśäÿ■éIJĂèęĄåĘ■åŐżèőąçő
ä¡ääŔŕäžěåŕĬèŕŢæŻť˙æůśåĚěçŽĎä¿Ńå■ŘæĬèèğĆår§çżŞæđIJïijŽ

```
>>> c = Circle(4.0)
>>> # Get instance variables
>>> vars(c)
{'radius': 4.0}

>>> # Compute area and observe variables afterward
>>> c.area
Computing area
50.26548245743669
>>> vars(c)
{'area': 50.26548245743669, 'radius': 4.0}

>>> # Notice access doesn't invoke property anymore
>>> c.area
50.26548245743669

>>> # Delete the variable and see property trigger again
```

```
>>> del c.area
>>> vars(c)
{'radius': 4.0}
>>> c.area
Computing area
50.26548245743669
>>>
```

è£Źçğ■æŰźæąĹæIJL'äÿĂäÿłåřŔçijžéŹůåřśæŸŕèőąçőŮåĞžçŽĎåĂijèćńåĹŻåžžåŘŐæŸŕåŔŕäžěèćńä£őæŤ

```
>>> c.area
Computing area
50.26548245743669
>>> c.area = 25
>>> c.area
25
>>>
```

åęĆædIJä¡ăæŃĚå£Čè£ŹäÿłéŮőécŸïijŇéĆčäźĹåŔŕäžěä¡£çŤĺäÿĂçğ■çĺ■å¿őæšąéĆčäźĹéńŸæŢĹçŽĎåőđç

```
def lazyproperty(func):
    name = '_lazy_' + func.__name__
    @property
    def lazy(self):
        if hasattr(self, name):
            return getattr(self, name)
        else:
            value = func(self)
            setattr(self, name, value)
            return value
    return lazy
```

åęĆædIJä¡ăä¡£çŤĺè£ŹäÿłçL'ĹæIJñïijŇåřśäijŽåŔŚçŐřçŐřåIJĺä£őæŤźæŞ■ä¡IJåůšçżŔäÿ■èćńåĚĄèőÿäžĘïij

```
>>> c = Circle(4.0)
>>> c.area
Computing area
50.26548245743669
>>> c.area
50.26548245743669
>>> c.area = 25
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>>
```

çĎűèĂŇïijŇè£Źçğ■æŰźæąĹæIJL'äÿĂäÿłçijžçĆźåřśæŸŕæL'ĂæIJL'getæŞ■ä¡IJéČ¡å£ĚéążèćńåőŽåŘŚåĹřă
`getter` åĞ¡æŢřäÿŁåŐżãĂĆ è£Źäÿłèů§äźŃåL'■çőÅå■ŢçŽĎåIJĺåőđä¿Ńå■ŮåĚÿäÿ■æ§ěæL'¿åĂijçŽĎæŰźæąĺ
åęĆædIJæČşèŐůåŔŰæŻt'åd'ŽåĚşäžŐpropertyåŠŇåŔŕçőąçŘĘåšđæĂğçŽĎä£ąæĄŕïijŇåŔŕäžěåŔĆèĂČ8.6åřŔ

8.11 çőĂåŇŰæŢřæ■őçżŞæđĎçŽĎåĹİåğŃåŇŰ

éŮőécŸ

ä¡ăåĘŹäžĘå¿Ĺåď˙ŽäžĚäżĚçŤĺä¡IJæŢřæ■őçżŞæđĎçŽĎçśżïijŇäÿ■æČşåĘŹåď˙łåď˙ŽçČęäžžçŽĎ __init__() åĞ¡æŢř

èğčåĘşæŰźæąĹ

åŔŕäžěåIJĺäÿĂäÿłå§žçśżäÿ■åĘŹäÿĂäÿłåĚňçŤĺçŽĎ __init__() åĞ¡æŢřijŽ

```
import math

class Structure1:
    # Class variable that specifies expected fields
    _fields = []

    def __init__(self, *args):
        if len(args) != len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))
        # Set the arguments
        for name, value in zip(self._fields, args):
            setattr(self, name, value)
```

çĎűåŘŐä¡£ä¡ăçŽĎçśżçżğæL'£èĞłè£Źäÿłåšžçśż:

```
# Example class definitions
class Stock(Structure1):
    _fields = ['name', 'shares', 'price']

class Point(Structure1):
    _fields = ['x', 'y']

class Circle(Structure1):
    _fields = ['radius']

    def area(self):
        return math.pi * self.radius ** 2
```

ä¡£çŤĺè£ŹäžŽçśżçŽĎçď˙žä¿ŇïijŽ

```
>>> s = Stock('ACME', 50, 91.1)
>>> p = Point(2, 3)
>>> c = Circle(4.5)
>>> s2 = Stock('ACME', 50)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
```

```
    File "structure.py", line 6, in __init__
        raise TypeError('Expected {} arguments'.format(len(self._
↪fields)))
TypeError: Expected 3 arguments
```

åęĆæđIJè£ŸæČşæŤŕæŇĄåĚşéŤőå■ŮåŔĆæŢřïijŇåŔŕäžěářĘåĚşéŤőå■ŮåŔĆæŢřèő¿ç¡őäÿžåőđä¿ŃåšđæĂ

```
class Structure2:
    _fields = []

    def __init__(self, *args, **kwargs):
        if len(args) > len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))

        # Set all of the positional arguments
        for name, value in zip(self._fields, args):
            setattr(self, name, value)

        # Set the remaining keyword arguments
        for name in self._fields[len(args):]:
            setattr(self, name, kwargs.pop(name))

        # Check for any remaining unknown arguments
        if kwargs:
            raise TypeError('Invalid argument(s): {}'.format(','.
↪join(kwargs)))
# Example use
if __name__ == '__main__':
    class Stock(Structure2):
        _fields = ['name', 'shares', 'price']

    s1 = Stock('ACME', 50, 91.1)
    s2 = Stock('ACME', 50, price=91.1)
    s3 = Stock('ACME', shares=50, price=91.1)
    # s3 = Stock('ACME', shares=50, price=91.1, aa=1)
```

ä¡ăè£ŸèČ¡åřĘäÿ■åIJÍ `_fields` äÿ■çŽĎåŘ■çğřåŁăåĚěåĹřåšđæĂğäÿ■åŐżïijŽ

```
class Structure3:
    # Class variable that specifies expected fields
    _fields = []

    def __init__(self, *args, **kwargs):
        if len(args) != len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))

        # Set the arguments
        for name, value in zip(self._fields, args):
            setattr(self, name, value)
```

```
        # Set the additional arguments (if any)
        extra_args = kwargs.keys() - self._fields
        for name in extra_args:
            setattr(self, name, kwargs.pop(name))

        if kwargs:
            raise TypeError('Duplicate values for {}'.format(','.
↪join(kwargs)))

# Example use
if __name__ == '__main__':
    class Stock(Structure3):
        _fields = ['name', 'shares', 'price']

    s1 = Stock('ACME', 50, 91.1)
    s2 = Stock('ACME', 50, 91.1, date='8/2/2012')
```

èőĺèőž

å¡Şä¡ăéIJĂèęĄä¡£çŤĺåď ğéĞŕå¿ĹåřŕçŽĎæŢřæ■őçżŞæđĎçśżçŽĎæŮűåĂŹïijŇ çŻÿæŕŤæĽŃåůěäÿĂäÿłäÿłåőŽäźĽ __init__() æŰźæşŢèĂŇåůšïijŇä¡£çŤĺè£ŹçğæŰźåijŕåŕŕäžěåď ğåď ğ

åIJĺäÿŁéİćçŽĎåőđçŐřäÿ■æĹŚäžňä¡£çŤĺäžĘ setattr() åĞ¡æŢřçśżèő¿ç¡őåśđæĂğåĂijïijŇ ä¡ăåŕŕèČ¡äÿ■æČşçŤĺè£ŹçğæŰźåijŕïijŇèĂŇæŸŕæČşçŻťæŐěæŻťæŰřåőć

```
class Structure:
    # Class variable that specifies expected fields
    _fields= []
    def __init__(self, *args):
        if len(args) != len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))

        # Set the arguments (alternate)
        self.__dict__.update(zip(self._fields,args))
```

ař¡çőąè£ŹäźŞåŕŕäžěæ■čåÿÿåůëä¡IJïijŇä¡ĘæŸŕå¡ŞåőŽäźĽå■ŘçśżçŽĎæŮűåĂŹéŮőécŸåřśæİěäžĘãĂĆ å¡Şäÿ Ăäÿłå■ŘçśżåőŽäźĽäžĘ __slots__ æĹŰèĂĚéĂŽè£Ğproperty(æĹŰæŔŕè£řåŹĺ)æİěåŇĚècĚæ§Řäÿłåş éĆčäźĹçŻťæŐěö£éŮőåőđä¿Ńå■ŮåĚÿåřśäÿ■ètůä¡IJçŤĺäžĘãĂĆæĹŚäžňäÿŁéİćä¡£çŤĺ setattr() äijŽæŸ¿å¿ŮæŻťéĂŽçŤĺäžŽïijŇåŽääÿžåőČäź§éĂĆçŤĺäžŐå■ŘçśżæČĚåĘţãĂĆ

è£Źçğ■æŰźæşŢåŤŕäÿĂäÿ■åě¡çŽĎåIJŕæŰźåŕśæŸŕåŕźæ§ŘäžZIDEèĂŇåůšïijŇåIJĺæŸ¿çd'žåÿőåŁ'åĞ¡æŢ

```
>>> help(Stock)
Help on class Stock in module __main__:
class Stock(Structure)
...
| Methods inherited from Structure:
```

```
|
| __init__(self, *args, **kwargs)
|
...
>>>
```

åŕŕäžěåŕĆèĂČ9.16åřŕèŁĆæİěåijžåĹűåIJĺ `__init__()` æŰźæşŢäÿ■æŇĞåőŽåŕĆæŢřçŽĎçśżåđŃç■¿åŘ■ãĂĆ

8.12 åőŽäźĽæŐěåŕčæĹŰèĂĚæŁ¡èśąå§žçśż

éŮőécŸ

ä¡ăæČşåőŽäźĽäÿĂäÿłæŐěåŕčæĹŰæŁ¡èśąçśżïijŇåźűäÿŤéĂŽè£ĞæĽğèąŇçśżåđŃæčĂæ§ěæİěçąőä£İå■Ŕ

èğčåĘşæŰźæąĹ

ä¡£çŤĺ `abc` æĺąåİŮåŕŕäžěå¿Ĺè¡żæİ¿çŽĎåőŽäźĽæŁ¡èśąå§žçśżïijŽ

```
from abc import ABCMeta, abstractmethod

class IStream(metaclass=ABCMeta):
    @abstractmethod
    def read(self, maxbytes=-1):
        pass

    @abstractmethod
    def write(self, data):
        pass
```

æŁ¡èśąçśżçŽĎäÿĂäÿłçĽźçĆźæŸŕåőČäÿ■èČ¡çŻť'æŐèèćńåőđä¿ŃåŇŰïijŇæŕŤåęĆä¡ăæČşåČŔäÿŃéİćè£Ź

```
a = IStream() # TypeError: Can't instantiate abstract class
              # IStream with abstract methods read, write
```

æŁ¡èśąçśżçŽĎçŻőçŽĎåřśæŸŕèőľ'åĹńçŽĎçśżçżğæĽ'£åőČåźűåőđçŐřçĽ'źåőŽçŽĎæŁ¡èśąæŰźæşŢïijŽ

```
class SocketStream(IStream):
    def read(self, maxbytes=-1):
        pass

    def write(self, data):
        pass
```

æŁ¡èśąå§žçśżçŽĎäÿĂäÿłäÿżèęĄçŤĺéĂŤæŸŕåIJĺäžčçăĄäÿ■æčĂæ§ěæ§ŘäžŽçśżæŸŕåŘęäÿžçĽ'źåőŽçśżåđ

```
def serialize(obj, stream):
    if not isinstance(stream, IStream):
        raise TypeError('Expected an IStream')
    pass
```

éŹď'äžĘçżğæĽ'£è£ŹçğæŰźåijŕåď'ŰïijŇè£ŸåŕŕäžééĂŽè£ĞæşĺåĘŇæŰźåijŕæİèèőľ'æ§ŘäÿłçśžåőđçŐřæ

```
import io

# Register the built-in I/O classes as supporting our interface
IStream.register(io.IOBase)

# Open a normal file and type check
f = open('foo.txt')
isinstance(f, IStream) # Returns True
```

@abstractmethod è£ŸèČ¡æşĺèğčéİŹæĂĄæŰźæşŢãĂĄçżæŰźæşŢåŠŇ properties ãĂĆ ä¡ääŕŧéIJĂä£İèŕĄè£Źäÿłæşĺèğčçť'ğéİääIJĺåĞ¡æŢřåőŽäźĽ'åĽ'■å■şåŕřïijŽ

```
class A(metaclass=ABCMeta):
    @property
    @abstractmethod
    def name(self):
        pass

    @name.setter
    @abstractmethod
    def name(self, value):
        pass

    @classmethod
    @abstractmethod
    def method1(cls):
        pass

    @staticmethod
    @abstractmethod
    def method2():
        pass
```

èőĺèőž

æăĞåĞĘåžŞäÿ■æIJL'å¿Ĺåď'ŽçŤĺåĽřæŁ¡èśąå§žçśżçŽĎåIJřæŰźãĂĆcollections æĺąåİŮåőŽäźĽ'äžĘå¿Ĺåď'Žèů§åőźåŹĺåŠŇè£■äžčåŹĺ(åžŕåĽŮãĂĄæŸäářĎãĂĄéŻĘåŕĹç■Ľ')æIJL'åĚşçŽĎæŁ numbers åžŞåőŽäźĽ'äžĘèů§æŢřå■Ůåŕźèśą(æŢť'æŢřãĂĄæţőçĆźæŢřãĂĄæIJL'çŘĘæŢřç■Ľ')æIJL'åĚşçŽĎå§ åžŞåőŽäźĽ'äžĘå¿Ĺåď'Žèů§I/OæŞ■ä¡IJçŻÿåĚşçŽĎå§žçśżãĂĆ

ä¡ääŕŕäžëä¡£çŤĺécĎåőŽäźĽ'çŽĎæŁ¡èśąçśżæİëæĽ'ğèąŇæŻť'éĂŽçŤĺçŽĎçśżåđŃæčĂæ§ěijŇä¿ŃåęĆïijŽ

```
import collections

# Check if x is a sequence
if isinstance(x, collections.Sequence):
...

# Check if x is iterable
if isinstance(x, collections.Iterable):
...

# Check if x has a size
if isinstance(x, collections.Sized):
...

# Check if x is a mapping
if isinstance(x, collections.Mapping):
```

åř¡çőąABCsåŔŕäžěèőľæĹŚäžňå¿ĹæŰźä¿£çŽĎåĄŽçśžåđŃæčĂæ§ěïijŇä¡ĘæŸŕæĹŚäžňåIJĺäžčçăĄäÿ■æĹ
åŻääÿžPythonçŽĎæIJňèťÍæŸŕäÿĂéŮĺåŁÍæĂĄçijŰçÍŃèŕ■èÍĂïijŇåĚűçŽőçŽĎåřśæŸŕçżŹä¡ăæŽťåď˙ŽçĄţæťżą
åijžåĹűçśžåđŃæčĂæ§ěæĹŰèőľä¡ăäžčçăĄåŔŸå¿ŮæŽťåď˙■æİĆïijŇè£ŹæăůåĄŽæŮăäijĆäžŐèĹ■æIJňæśĆæIJ

8.13 åőđçŐřæŢřæ■őæÍąåđŃçŽĎçśžåđŃçžęæİ§

éŮőéćŸ

ä¡ăæČşåőŽäźĽæ§ŘäžŽåIJĺåśđæĂğèţŃåĂijäÿŁéİćæIJĽéŹŘåĹűçŽĎæŢřæ■őçżŞæđĎãĂĆ

èğčåĘşæŰźæąĹ

åIJÍè£ŹäÿłéŮőéćŸäÿ■ïijŇä¡ăéIJĂèęĄåIJĺåŕźæ§ŘäžŽåőđä¿ŃåśđæĂğèţŃåĂijæŮűè£ŻèąŇæčĂæ§ěãĂĆ
æĽĂäżěä¡ăèęĄèĞłåőŽäźĽåśđæĂğèţŃåĂijåĞ¡æŢřïijŇè£Źçğ■æČĚåĘţäÿŃæIJĂåě¡ä¡£çŤĺæŔŘè£řåŹĺãĂĆ

äÿŃéİćçŽĎäžčçăĄä¡£çŤĺæŔŘè£řåŹĺåőđçŐřäžĘäÿĂäÿłçşżçż§çśžåđŃåŠŇèţŃåĂijéłŇèŕĄæąĘæđűïijŽ

```
# Base class. Uses a descriptor to set a value
class Descriptor:
    def __init__(self, name=None, **opts):
        self.name = name
        for key, value in opts.items():
            setattr(self, key, value)

    def __set__(self, instance, value):
        instance.__dict__[self.name] = value


# Descriptor for enforcing types
```

```
class Typed(Descriptor):
    expected_type = type(None)

    def __set__(self, instance, value):
        if not isinstance(value, self.expected_type):
            raise TypeError('expected ' + str(self.expected_type))
        super().__set__(instance, value)


# Descriptor for enforcing values
class Unsigned(Descriptor):
    def __set__(self, instance, value):
        if value < 0:
            raise ValueError('Expected >= 0')
        super().__set__(instance, value)


class MaxSized(Descriptor):
    def __init__(self, name=None, **opts):
        if 'size' not in opts:
            raise TypeError('missing size option')
        super().__init__(name, **opts)

    def __set__(self, instance, value):
        if len(value) >= self.size:
            raise ValueError('size must be < ' + str(self.size))
        super().__set__(instance, value)
```

è£ŹäžŻçśžåřśæŸŕäą¡äèęĄåĹŻåžžçŽĎæŢřæ■őæĺąåđŃæĹŰçśżåđŃçşżçż§çŽĎå§žçąĂæđĎåžžæĺąåİŮãĂĆäÿŃéİćåřśæŸŕæĹŚäžňåőđéŹĚåőŽäźĽçŽĎåŘĎçğ■äÿ■åŘŇçŽĎæŢřæ■őçśżåđŃïijŽ

```
class Integer(Typed):
    expected_type = int

class UnsignedInteger(Integer, Unsigned):
    pass

class Float(Typed):
    expected_type = float

class UnsignedFloat(Float, Unsigned):
    pass

class String(Typed):
    expected_type = str

class SizedString(String, MaxSized):
    pass
```

çĎűåŘŐä¡£çŤĺè£ŹäžŻèĞłåőŽäźĽ'æŢřæ■őçśżåđŃïijŇæĹŚäžňåőŽäźĽ'äÿĂäÿłçśżïijŽ

```
class Stock:
    # Specify constraints
    name = SizedString('name', size=8)
    shares = UnsignedInteger('shares')
    price = UnsignedFloat('price')

    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

çĎűåŘŐæţŃèŕŢè£ŹäÿłçśżçŽĎåšđæĂğètŃåĂijçžęæİ§ïijŇåŕŕåŔŚçŐřåŕźæ§ŘäžŽåšđæĂğçŽĎètŃåĂijè£İ

```
>>> s.name
'ACME'
>>> s.shares = 75
>>> s.shares = -10
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 17, in __set__
        super().__set__(instance, value)
    File "example.py", line 23, in __set__
        raise ValueError('Expected >= 0')
ValueError: Expected >= 0
>>> s.price = 'a lot'
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 16, in __set__
        raise TypeError('expected ' + str(self.expected_type))
TypeError: expected <class 'float'>
>>> s.name = 'ABRACADABRA'
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 17, in __set__
        super().__set__(instance, value)
    File "example.py", line 35, in __set__
        raise ValueError('size must be < ' + str(self.size))
ValueError: size must be < 8
>>>
```

è£ŸæIJL'äÿĂäžŽæŁĂæIJŕåŔŕäžěçőĂåŇŰäÿŁéİćçŽĎäžčçăĄïijŇåĚűäÿ■äÿĂçğ■æŸŕä¡£çŤĺçśżèčĚéěřåŹĺ

```
# Class decorator to apply constraints
def check_attributes(**kwargs):
    def decorate(cls):
        for key, value in kwargs.items():
            if isinstance(value, Descriptor):
                value.name = key
                setattr(cls, key, value)
            else:
                setattr(cls, key, value(key))
```

```
        return cls

    return decorate

# Example
@check_attributes(name=SizedString(size=8),
                  shares=UnsignedInteger,
                  price=UnsignedFloat)
class Stock:
    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

åŕęåď Űäÿ Äçğ■æŰźåijŕæŸŕä¡£çŤĺåĚČçśżïijŽ

```
# A metaclass that applies checking
class checkedmeta(type):
    def __new__(cls, clsname, bases, methods):
        # Attach attribute names to the descriptors
        for key, value in methods.items():
            if isinstance(value, Descriptor):
                value.name = key
        return type.__new__(cls, clsname, bases, methods)

# Example
class Stock2(metaclass=checkedmeta):
    name = SizedString(size=8)
    shares = UnsignedInteger()
    price = UnsignedFloat()

    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

èőĺèőž

æIJňèŁĆä¡£çŤĺäžĘå¿Ĺåď ŽénŸçžğæŁÄæIJŕïijŇåŇĚæŃňæŔŕè£řåŹĺãĂĄæůůåĚěçśżãĂĄsuper()çŽĎä¡£çŤĺãĂĄçśżèčĚéěřåŹĺåŠŇåĚČçśżãĂĆ äÿ■åŕŕèČ¡åIJĺè£ŹéĞŇäÿĂäÿĂèŕęçżĘåśŢåijÄæİěèőšïijŇä¡ĘæŸä¡ĘæŸŕïijŇæĹŚåIJĺè£ŹéĞŇè£ŸæŸŕèęĄæŔŘäÿĂäÿŃåĞääÿłéIJÄèęĄæşĺæĎŔçŽĎçĆźãĂĆ

éęŰåĚĹïijŇåIJÍ Descriptor å§žçśżäÿ■ä¡ăäijŽçIJŃåĹřæIJL'äÿł__set__() æŰźæşŢïijŇå■ť'æšąæIJL'çŻÿåžŤçŽĎ __get__() æŰźæşŢãĂĆ åęĆædIJäÿĂäÿłæŔŘè£řäżĚäżĚæŸŕäžŐåžŢåśĆåőđä¿Ńå■ŮåĚÿäÿ■èŐůåŔŰæ§ŘäÿłåśđæĂğåĂijçŽĎèŕÏïijŇéĆ__get__() æŰźæşŢãĂĆ

æL'ĂæIJL'æŔŘè£řåŹĺçśżéČ¡æŸŕå§žäžŐæůůåĚěçśżæİěåőđçŐřçŽĎãĂĆæŕŤåęĆ Unsigned åŠŇ MaxSized èęĄèů§åĚűäżŰçżğæL'£èĞł Typed çśżæůůåĚěãĂĆ

è£ŹéĞŇåĹľçŤĺåď˙ŽçżğæĽ£æİěåőđçŐřçŻÿåžŤçŽĎåŁ§èČ¡ãĂĆ

æůůåĚěçśžçŽĎäÿĂäÿłærŤè¿ČéŽ¿çŘĘèğčçŽĎåIJřæŰźæŸŕïijŇèřČçŤĺ
`super()` åĞ¡æŢřæŮűïijŇä¡ääźűäÿ■ç§ěéĄŞçľűçń§èęĄèřČçŤĺåŞłäÿłåĚůä¡Şçśžã ĂĆ
ä¡ăéIJĂèęĄèů§åĚűäžŰçśžçżŞåŘĹåŘŐæĽ■èČ¡æ■čçąőçŽĎä¡£çŤĺïijŇäź§ářśæŸŕå£Ěéążå ŘĹä¡IJæĽ■èČ¡äžğçŤ

ä¡£çŤĺçśżèčĚéěřåŹĺåŠŇåĚČçśżéĂŽåÿÿåŘŕäžěçőĂåŇŰäžčçăĄãĂĆäÿŁéİćäÿd'äÿłä¿Ńå■Řäÿ■ä¡ääijŽåŔŚ

```
# Normal
class Point:
    x = Integer('x')
    y = Integer('y')

# Metaclass
class Point(metaclass=checkedmeta):
    x = Integer()
    y = Integer()
```

æĽĂæIJL'æŰźæşŢäÿ■ïijŇçśżèčĚéěřåŹĺæŰźæąĹåžŤèřěæŸŕæIJĂçĄţæť˙żåŠŇæIJĂéńŸæŸŐçŽĎãĂĆ
éęŰåĚĹïijŇåőČåźűäÿ■ä¿İèţŰäżżä¡ŢåĚűäžŰæŰřçŽĎæŁĂæIJŕïijŇæŕŤåęĆåĚČçśžãĂĆåĚűæňąïijŇèčĚéěřåŹĺå

æIJĂåŘŐïijŇèčĚéěřåŹĺè£ŸèČ¡ä¡IJäÿžæůůåĚěçśžçŽĎæŻ£äžčæŁĂæIJŕæİěåőđçŐřåŘŇæăůçŽĎæŢĹæđIJ

```
# Decorator for applying type checking
def Typed(expected_type, cls=None):
    if cls is None:
        return lambda cls: Typed(expected_type, cls)
    super_set = cls.__set__

    def __set__(self, instance, value):
        if not isinstance(value, expected_type):
            raise TypeError('expected ' + str(expected_type))
        super_set(self, instance, value)

    cls.__set__ = __set__
    return cls


# Decorator for unsigned values
def Unsigned(cls):
    super_set = cls.__set__

    def __set__(self, instance, value):
        if value < 0:
            raise ValueError('Expected >= 0')
        super_set(self, instance, value)

    cls.__set__ = __set__
    return cls


# Decorator for allowing sized values
```

```
def MaxSized(cls):
    super_init = cls.__init__

    def __init__(self, name=None, **opts):
        if 'size' not in opts:
            raise TypeError('missing size option')
        super_init(self, name, **opts)

    cls.__init__ = __init__

    super_set = cls.__set__

    def __set__(self, instance, value):
        if len(value) >= self.size:
            raise ValueError('size must be < ' + str(self.size))
        super_set(self, instance, value)

    cls.__set__ = __set__
    return cls


# Specialized descriptors
@Typed(int)
class Integer(Descriptor):
    pass


@Unsigned
class UnsignedInteger(Integer):
    pass


@Typed(float)
class Float(Descriptor):
    pass


@Unsigned
class UnsignedFloat(Float):
    pass


@Typed(str)
class String(Descriptor):
    pass


@MaxSized
class SizedString(String):
    pass
```

è£ŹçğæŰźåijŔåőŽäźĽçŽĎçśżèů§äźŃåĽçŽĎæŢĹæđIJäÿĂæăůïijŇèĂŇäÿŤæĽğèąŇéĂ§åžęäijŽæŻť'å
èő¿ç¡őäÿĂäÿłçőĂå■ŢçŽĎçśżåđŃåśđæĂğçŽĎåĂijïijŇèčĚéěřåŹĺæŰźåijŔèęĄæŕŤäźŃåĽ■çŽĎæůůåĚěçśżçŽĎ
çŐřåIJĺä¡ääžŤèŕěåžĘáźÿèĞłåůśèŕżåőŇäžĘæIJňèŁĆåĚĺéČĺåĘĚåőźäžĘåŘğïij§^_^

8.14 åőđçŐřèĞłåőŽäźĽåőźåŹĺ

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłèĞłåőŽäźĽçŽĎçśżæİěæĺąæŃ§åĘĚç¡őçŽĎåőźåŹĺçśżåŁ§èČ¡ïijŇærŤåęĆåĹŮèąĺåŠ

èğčåĘşæŰźæąĹ

`collections` åőŽäźĽäžĘå¿Ĺåd'ŽæŁ¡èśąå§žçśżïijŇå¡Şä¡ăæČşèĞłåőŽäźĽåőźåŹĺçśżçŽĎæŮűåĂŹåőČ
ærŤåęĆä¡ăæČşèől'ä¡ăçŽĎçśżæŤŕæŇĄè£■äžčïijŇéĆčářśèől'ä¡ăçŽĎçśżçžğæL'£
`collections.Iterable` å■şåŔřïijŽ

```
import collections
class A(collections.Iterable):
    pass
```

äÿ■è£Ğä¡ăéIJĂèęĄåőđçŐř `collections.Iterable` æL'ĂæIJL'çŽĎæŁ¡èśąæŰźæşŢïijŇåŘęåĹŹäijŽæŁěéŤŹ:

```
>>> a = A()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: Can't instantiate abstract class A with abstract methods␣
 ↪__iter__
>>>
```

ä¡ääŔłèęĄåőđçŐř `__iter__()` æŰźæşŢářśäÿ■äijŽæŁěéŤŹäžĘ(åŔĆèĂČ4.2åŠŇ4.7ářŔèŁĆ)ãĂĆ

ä¡ääŔŕäžěåĚĹèŕŢçİĂåŐžåőđä¿ŇåŇŰäÿĂäÿłářźèśąïijŇåIJĺéŤŹèŕŕæŔŘçd'žäÿ■åŔŕäžěæL'¿åĹřéIJĂèęĄåő

```
>>> import collections
>>> collections.Sequence()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: Can't instantiate abstract class Sequence with abstract␣
 ↪methods \
__getitem__, __len__
>>>
```

äÿŇéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎçd'žä¿ŃïijŇçżğæL'£èĞłäÿŁéİćSequenceæŁ¡èśąçśżïijŇåźűäÿŤåőđçŐřåĚČ

```
class SortedItems(collections.Sequence):
    def __init__(self, initial=None):
```

```
        self._items = sorted(initial) if initial is not None else []

    # Required sequence methods
    def __getitem__(self, index):
        return self._items[index]

    def __len__(self):
        return len(self._items)

    # Method for adding an item in the right location
    def add(self, item):
        bisect.insort(self._items, item)


items = SortedItems([5, 1, 3])
print(list(items))
print(items[0], items[-1])
items.add(2)
print(list(items))
```

åŔŕäžěçIJŃåĹřïijŇSortedItemsèů§æŹőéĂŽçŽĎåžŔåĹŮæšąäžĂäźĹäÿď æăůïijŇæŤŕæŇĄæĽ ĂæIJĽåÿÿç

è£ŹéĞŇéÍćä¡£çŤĺåĹřäžĘ `bisect` æĺąåİŮïijŇåőČæŸŕäÿĂäÿłåIJĺæŐŠåžŔåĹŮèąĺäÿ■æŔŠåĚěåĚČçť ăçŽ

èőÍèőž

ä¡£çŤÍ `collections` äÿ■çŽĎæŁ¡èśąå§žçśżåŔŕäžěçąőä£Íä¡ăèĞłåőŽäźĽçŽĎåőźåŹĺåőđçŐřäžĘæĽĂæI
ä¡ăçŽĎèĞłåőŽäźĽåőźåŹĺäijŽæżąèűşåď ğéČĺåĹĘçśżåđŃæčĂæ§ěéIJĂèęĄïijŇåęĆäÿŃæĽĂçď žïijŽ

```
>>> items = SortedItems()
>>> import collections
>>> isinstance(items, collections.Iterable)
True
>>> isinstance(items, collections.Sequence)
True
>>> isinstance(items, collections.Container)
True
>>> isinstance(items, collections.Sized)
True
>>> isinstance(items, collections.Mapping)
False
>>>
```

`collections` äÿ■å¿Łåď ŽæŁ¡èśąçśżäijŽäÿžäÿĂäžŽåÿÿèğĄåőźåŹĺæŞ■ä¡IJæŔŘä¿ŻéżŸèőď çŽĎåőđçČ
è£ŹæăůäÿĂæÍěä¡ăåŔłéIJĂèęĄåőđçŐřéĆčäžŽä¡ăæIJĂæĎ§åĚt èűčçŽĎæŰźæşŢå■şåŔŕãĂĆåĄĞèő¿ä¡ăçŽĎçśż
`collections.MutableSequence` ïijŇåęĆäÿŃïijŽ

```
class Items(collections.MutableSequence):
    def __init__(self, initial=None):
```

```
        self._items = list(initial) if initial is not None else []

    # Required sequence methods
    def __getitem__(self, index):
        print('Getting:', index)
        return self._items[index]

    def __setitem__(self, index, value):
        print('Setting:', index, value)
        self._items[index] = value

    def __delitem__(self, index):
        print('Deleting:', index)
        del self._items[index]

    def insert(self, index, value):
        print('Inserting:', index, value)
        self._items.insert(index, value)

    def __len__(self):
        print('Len')
        return len(self._items)
```

åęĆædIJä¡ăåĹŻåžž `Items` çŽĎåőđä¿ŃïijÑä¡ăäijŽåŔŚçŐřåőČæŤŕæŇĄåĞäážŐæĽĂæIJL'çŽĎæäÿå£Čå
äÿŃéÍćæŸŕä¡£çŤĺæijŤçd'žïijŽ

```
>>> a = Items([1, 2, 3])
>>> len(a)
Len
3
>>> a.append(4)
Len
Inserting: 3 4
>>> a.append(2)
Len
Inserting: 4 2
>>> a.count(2)
Getting: 0
Getting: 1
Getting: 2
Getting: 3
Getting: 4
Getting: 5
2
>>> a.remove(3)
Getting: 0
Getting: 1
Getting: 2
Deleting: 2
>>>
```

æIJňåřŕèŁĆåŕłæŸŕåŕźPythonæŁ¡èśąçśżåŁ§èČ¡çŽĎæŁŻçăŰåijŢçŐĽ'ãĂĆnumbers æĺąåİŮæŔŘä¿ŻäžĘäÿĂäÿłçśżäijijçŽĎèů§æŢť'æŢřçśżåđŃçŻÿåĚşçŽĎæŁ¡èśąçśżåđŃéŻĘåŘĹãĂĆ åŔŕäžěåŔĆèĂČ8.12åřŕèŁĆæİěæđĎéĂăæŻť'åđ'ŽèĞłåőŽäźĽ'æŁ¡èśąå§žçśżãĂĆ

8.15 åśđæĂğçŽĎäżčçŘĘèő£éŮő

éŮőécŸ

ä¡ăæČşåřĘæ§Řäÿłåőđä¿ŃçŽĎåśđæĂğèő£éŮőäżčçŘĘåĹřåĘĚéČĺåŔęäÿĂäÿłåőđä¿Ńäÿ■åŐżïijŇçŻőçŽĎå

èğčåĘşæŰźæąĹ

çőĂå■Ţæİěèŕť'ïijŇäżčçŘĘæŸŕäÿĂçğ■çijŰçĺŃæĺąåijŔïijŇåőČåřĘæ§Řäÿłæş■ä¡IJè¡ńçğżçżŹåŔęåđ'ŰäÿĂ æIJĂçőĂå■ŢçŽĎå¡ćåijŔåŔŕèČ¡æŸŕåČŔäÿŃéİćè£ŹæăůïijŽ

```
class A:
    def spam(self, x):
        pass

    def foo(self):
        pass


class B1:
    """çőĂå■ŢçŽĎäżčçŘĘ"""

    def __init__(self):
        self._a = A()

    def spam(self, x):
        # Delegate to the internal self._a instance
        return self._a.spam(x)

    def foo(self):
        # Delegate to the internal self._a instance
        return self._a.foo()

    def bar(self):
        pass
```

åęĆæđIJäžĚäžĚåřśäÿđ'äÿłæŰźæşŢéIJĂèęĄäżčçŘĘïijŇéĆčäźĹåČŔè£ŹæăůåĘŹåřśèűşåđ'§äžĘãĂĆä¡ĘæŸ éĆčäźĹä¡£çŤĺ `__getattr__()` æŰźæşŢæĹŰèőÿæĹŰæŻť'åě¡äžŻïijŽ

```
class B2:
    """ä¡£çŤĺ__getattr__çŽĎäżčçŘĘïijŇäżčçŘĘæŰźæşŢæŕŤè¿Čåd'ŽæŮűåĂŹ"""

    def __init__(self):
```

```
        self._a = A()

    def bar(self):
        pass

    # Expose all of the methods defined on class A
    def __getattr__(self, name):
        ""
↪"è£ŹäÿłæŰźæşŢåIJĺèő£éŮőçŽĎattributeäÿ■å■ŸåIJĺçŽĎæŮűåĂŹèćńèřČçŢĺ
        the __getattr__() method is actually a fallback method
        that only gets called when an attribute is not found"""
        return getattr(self._a, name)
```

__getattr__ æŰźæşŢæŸŕåIJĺèő£éŮőattributeäÿ■å■ŸåIJĺçŽĎæŮűåĂŹèćńèřČçŢĺïijÑä¡£çŢĺæijŢçď'ži

```
b = B()
b.bar() # Calls B.bar() (exists on B)
b.spam(42) # Calls B.__getattr__('spam') and delegates to A.spam
```

åŔęåď'ŰäÿĂäÿłäżčçŘĘä¿Ńå■ŘæŸŕåőđçŐřäżčçŘĘæĺąåijŔïijÑä¿ŃåęĆïijŽ

```
# A proxy class that wraps around another object, but
# exposes its public attributes
class Proxy:
    def __init__(self, obj):
        self._obj = obj

    # Delegate attribute lookup to internal obj
    def __getattr__(self, name):
        print('getattr:', name)
        return getattr(self._obj, name)

    # Delegate attribute assignment
    def __setattr__(self, name, value):
        if name.startswith('_'):
            super().__setattr__(name, value)
        else:
            print('setattr:', name, value)
            setattr(self._obj, name, value)

    # Delegate attribute deletion
    def __delattr__(self, name):
        if name.startswith('_'):
            super().__delattr__(name)
        else:
            print('delattr:', name)
            delattr(self._obj, name)
```

ä¡£çŢĺè£ŹäÿłäżčçŘĘçśżæŮűïijÑä¡ăåŔłéIJĂèęĄçŢĺåőČæĺěåŇĚèčĔäÿŃåĔűäżŰçśżå■şåŔŕïijŽ

```
class Spam:
    def __init__(self, x):
        self.x = x

    def bar(self, y):
        print('Spam.bar:', self.x, y)

# Create an instance
s = Spam(2)
# Create a proxy around it
p = Proxy(s)
# Access the proxy
print(p.x)  # Outputs 2
p.bar(3)  # Outputs "Spam.bar: 2 3"
p.x = 37  # Changes s.x to 37
```

éĂŽè£ĞèĞłåőŽäźL'åśđæĂğèő£éŮőæŰźæşŢïijŇä¡ääŔŕäżěçŤĺäÿ■åŘŇæŰźåijŔèĞłåőŽäźL'äžčçŘĘçśżèąŇ

èőĺèőž

äžčçŘĘçśżæIJL'æŮűåĂŹåŔŕäżěä¡IJäÿžçżğæL'£çŽĎæŻ£äžčæŰźæąĹãĂĆä¿ŃåęĆïijŇäÿĂäÿłçőĂå■ŢçŽĎ

```
class A:
    def spam(self, x):
        print('A.spam', x)
    def foo(self):
        print('A.foo')

class B(A):
    def spam(self, x):
        print('B.spam')
        super().spam(x)
    def bar(self):
        print('B.bar')
```

ä¡£çŤĺäżčçŘĘçŽĎèŕĬïijŇåřśæŸŕäÿŃéĺćè£ŹæăůïijŽ

```
class A:
    def spam(self, x):
        print('A.spam', x)
    def foo(self):
        print('A.foo')

class B:
    def __init__(self):
        self._a = A()
    def spam(self, x):
        print('B.spam', x)
        self._a.spam(x)
```

```
    def bar(self):
        print('B.bar')
    def __getattr__(self, name):
        return getattr(self._a, name)
```

å¡Şåőđç Őřäžčç ŘĘæĺąåijŔæŮűïijŇè£ŸæIJL'äžŽçżĘèŁĆéIJĂèęĄæşĺæĎŔãĂĆ éęŰåĚĹïijŇ__getattr__() åőđéŹĚæŸŕäÿĂäÿłåŘŐåđ'ĞæŰźæşŢïijŇåŔłæIJL'åIJĺášđæĂğäÿ■å■ŸåIJĺæŮű åŽăæ■đ'ïijŇåęĆæđIJäžčçŘĘçśžåőđä¿ŃæIJňèžńæIJL'è£ŹäÿłášđæĂğçŽĎèŕĭïijŇéĆčäźĹäÿ■äijŽèğęåŔŚè£ŹäÿłåŘęåđ'ŰïijŇ__setattr__() åŠŇ __delattr__() éIJĂèęĄéĆìåđ'ŰçŽĎé■ŤæşŢæİěåŇžåĹĘäžčçŘĘåőđ _obj çŽĎåšđæĂğãĂĆ äÿĂäÿłéĂŽåÿÿçŽĎçžęåőŽæŸŕåŘłäžčçŘĘéĆčäžŹäÿ■äžěäÿŃåĹŠçž£ _ åijĂåđ't'çŽĎåšđæĂğ(äžčçŘĘçśžåŘłæŽt'éIJšèćńäžčçŘĘçśžçŽĎåĚňåĚšåšđæĂğ)ãĂĆ

è£ŸæIJL'äÿĂçĆźéIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇ__getattr__() åŕźäžŐåđ'ğéČĺåĹĘäžěåŘŇäÿŃåĹŠçž£(__)åijĂåğŃåŠŇçżŞår¿çŽĎåšđæĂğåźűäÿ■éĂĆçŤĺãĂĆ æŕŤåęĆïijŇèĂČèŹŠåęĆäÿŃçŽĎçśžïijŽ

```
class ListLike:
    """__getattr__
↪åŕźäžŐåŘŇäÿŃåĹŠçž£åijĂåğŃåŠŇçżŞår¿çŽĎæŰźæşŢæŸŕäÿ■èČ¡çŤĺçŽĎïijŇéIJĂèęĄäÿĂäÿłäÿł
↪"""

    def __init__(self):
        self._items = []

    def __getattr__(self, name):
        return getattr(self._items, name)
```

åęĆæđIJæŸŕåĹŻåžžäÿĂäÿłListLikeåŕźèśąïijŇäijŽåŘŞçŐřåőČæŤŕæŇĄæŹőéĂŽçŽĎåĹŮèąĺæŰźæşŢïijŇ ä¡ĘæŸŕå■t'äÿ■æŤŕæŇĄlen()ãĂĄåĚČçt'äæ§ěæL'¿ç■L'ãĂĆä¿ŃåęĆïijŽ

```
>>> a = ListLike()
>>> a.append(2)
>>> a.insert(0, 1)
>>> a.sort()
>>> len(a)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: object of type 'ListLike' has no len()
>>> a[0]
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: 'ListLike' object does not support indexing
>>>
```

äÿžäžĘèől'åőČæŤŕæŇĄè£ŹäžŽæŰźæşŢïijŇä¡åå£ĚéąžæL'ŃåŁĺçŽĎåőđçŐřè£ŹäžŽæŰźæşŢäžčçŘĘïijŽ

```
class ListLike:
    """__getattr__
↪åŕźäžŐåŘŇäÿŃåĹŠçž£åijĂåğŃåŠŇçżŞår¿çŽĎæŰźæşŢæŸŕäÿ■èČ¡çŤĺçŽĎïijŇéIJĂèęĄäÿĂäÿłäÿł
↪"""
```

```
    def __init__(self):
        self._items = []

    def __getattr__(self, name):
        return getattr(self._items, name)

    # Added special methods to support certain list operations
    def __len__(self):
        return len(self._items)

    def __getitem__(self, index):
        return self._items[index]

    def __setitem__(self, index, value):
        self._items[index] = value

    def __delitem__(self, index):
        del self._items[index]
```

11.8åřŔèŁĆè£ŸæIJL'äÿĂäÿłåIJÍè£IJçÍŃæŰźæşŢèřČçŤÍçŐŕåćČäÿ■ä¡£çŤÍäžčçŘEçŽĎä¿Ńå■ŘãĂĆ

8.16 åIJÍçśžäÿ■åőŽäźĽåď'ŽäÿłæđĎéĂăåŹÍ

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłçśżïijŇéŹď'äžĘä¡£çŤÍ `__init__()` æŰźæşŢåď'ŰïijŇè£ŸæIJL'åĚűäžŰæŰźåijŔåŔŕäžěåĹİåğŃåŇŰåőČãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘåőđçŐřåď'ŽäÿłæđĎéĂăåŹÍïijŇä¡ăéIJĂèęĄä¡£çŤÍåĹřçśżæŰźæşŢãĂĆä¿ŃåęĆïijŽ

```
import time

class Date:
    """æŰźæşŢäÿĂïijŽä¡£çŤÍçśżæŰźæşŢ"""
    # Primary constructor
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day

    # Alternate constructor
    @classmethod
    def today(cls):
```

```
        t = time.localtime()
        return cls(t.tm_year, t.tm_mon, t.tm_mday)
```

çŻťæŐěèřČçŤĺçśżæŰźæşŢå■şåŕřïijŇäÿŃéİćæŸŕä¡£çŤĺçđ'žä¿ŃïijŽ

```
a = Date(2012, 12, 21) # Primary
b = Date.today() # Alternate
```

èőĺèőž

çśżæŰźæşŢçŽĎäÿĂäÿłäÿżèęAçŤĺéĂŤåřśæŸŕåőŽäźL'åđ'ŽäÿłæđĎéĂäåŹĺãĂĆåőČæŐěåŔŮäÿĂäÿł class ä¡IJäÿžçňňäÿĂäÿłåŔĆæŢř(cls)ãĂĆ ä¡ääžŤèŕěæşĺæĎŔåĹřäžĘè£Źäÿłçśżèćńç ŤĺæİěåĹŻåżžåźűè£ŤåŻđ

```
class NewDate(Date):
    pass

c = Date.today() # Creates an instance of Date (cls=Date)
d = NewDate.today() # Creates an instance of NewDate (cls=NewDate)
```

8.17 åĹŻåżžäÿ■èřČçŤĺnitæŰźæşŢçŽĎåőđä¿Ń

éŮőécŸ

ä¡ăæČşåĹŻåżžäÿĂäÿłåőđä¿ŃïijŇä¡ĘæŸŕåÿŇæIJŻçżŢè£ĞæĽğèąŇ __init__() æŰźæşŢãĂĆ

èğčåĘşæŰźæąĹ

åŕŕäżěéĂŽè£Ğ __new__() æŰźæşŢåĹŻåżžäÿĂäÿłæIJłåĹİåğŃåŇŰçŽĎåőđä¿ŃãĂĆä¿ŃåęĆèĂČèŹŚåę

```
class Date:
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day
```

äÿŃéİćæijŤçđ'žåęĆä¡Ţäÿ■èřČçŤĺ __init__() æŰźæşŢæİěåĹŻåżžè£ŹäÿłDateåőđä¿ŃïijŽ

```
>>> d = Date.__new__(Date)
>>> d
<__main__.Date object at 0x1006716d0>
>>> d.year
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
```

```
AttributeError: 'Date' object has no attribute 'year'
>>>
```

çżŞæđĲåŔŕäżěçĲŃåĹřïijŇè£Źäÿł Dateåőđä¿ŃçŽĎåśđæĂğyearè£Ÿäÿ■å■ŸåĲĺïijŇæĽĂäżěä¡ăéĲĂèęĄæ

```
>>> data = {'year':2012, 'month':8, 'day':29}
>>> for key, value in data.items():
...     setattr(d, key, value)
...
>>> d.year
2012
>>> d.month
8
>>>
```

èőĺèőž

å¡ŞæĹŚäżňåĲĺåŔ■åžŔåĹŮåŕźèśąæĹŰèĂĚåőđçŐřæ§Ŗäÿłçśżæŕźæş ţæđĎéĂăåĞ¡æŢřæŮűéĲĂèęĄçżŢè __init__() æŰźæşŢæİěåĹŻåżžåŕźèśąãĂĆ ä¿ŃåęĆïijŇåŕźäžŐäÿŁéİćçŽĎDateæİěæİěèőšïijŇæĲĽæŮűåĂ today() ïijŽ

```
from time import localtime

class Date:
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day

    @classmethod
    def today(cls):
        d = cls.__new__(cls)
        t = localtime()
        d.year = t.tm_year
        d.month = t.tm_mon
        d.day = t.tm_mday
        return d
```

åŘŇæăůïijŇåĲĺä¡ăåŔ■åžŔåĹŮåŇŰJSONæŢřæ■őæŮűäžğçŤ§äÿĂäÿłåęĆäÿŃçŽĎå■ŮåĚÿåŕźèśąïijŽ

```
data = { 'year': 2012, 'month': 8, 'day': 29 }
```

åęĆæđĲä¡ăæČşåřĘåőČè¡ňæ■ćæĹŘäÿĂäÿłDateçśżåđŃåőđä¿ŃïijŇåŔŕäżěä¡£çŤĺäÿŁéİćçŽĎæŁĂæĲŕãĂĆ

å¡Şä¡ăéĂŽè£Ğè£Źçğ■éİđåÿÿèğĎæŰźåijŘæİěåĹŻåżžåőđä¿ŃçŽĎæŮűåĂŹïijŇæĲĂåě¡äÿ■èęĄçŻť æŐěåŘęåĹŹçŽĎèŕİïijŇåęĆæđĲè£Źäÿłçśżä¡£çŤĺäžĘ __slots__ ãĂĄproperties ãĂĄdescriptors æĹŰåĚűäžŰéńŸçžğæŁĂæĲŕçŽĎæŮűåĂŹäżčçăĄåřśäijŽåď śæŢĹãĂĆ èĂŇè£ŹæŮűåĂŹä¡£çŤĺ setattr() æŰźæşŢäijŽèőľä¡ăçŽĎäżčçăĄåŔŸå¿ŮæŻť åŁăéĂŽçŤĺãĂĆ

8.18 åĹľçŤÍMixinsæĽľåśŢçśżåŁ§èČ¡

éŮőécŸ

ä¡ăæIJL'å¿Ĺåd'ŽæIJL'çŤÍçŽĎæŰźæşŢïijŇæČşä¡£çŤĺåőČäżňæİěæL'l'åśŢåĚűäżŰçśżçŽĎåŁ§èČ¡ãĂĆä¡Ęą
åŻăæ■d'ä¡ääÿ■èČ¡çőĂå■ŢçŽĎåřĘè£ŹäžŻæŰźæşŢæŤ¿åĚëäÿĂäÿłå§žçśżïijŇçĎůåŘŐèćńåĚűäżŰçśżçżğæL'£ã

èğčåĘşæŰźæąĹ

éĂŽåÿÿå¡Şä¡ăæČşèĞłåőŽäźL'çśżçŽĎæŮűůåĂŹäijŽçćřäÿŁè£ŹäžŻéŮőécŸãĂĆåŔŕèČ¡æŸŕæ§Řäÿłåž§æŔ
ä¡ăåŔŕäžěåĹľçŤĺåőČäżňæİěædĎéĂää¡äèĞłåůśçŽĎçśżãĂĆ

åĄĞèő¿ä¡ăæČşæL'l'åśŢæŸăårĎårźèśąïijŇçżŹåőČäżňæůżåŁăæŮěå£ŮãĂĄåŤŕäÿĂæĂğèő¿ç¡őãĂĄçśżådł

```
class LoggedMappingMixin:
    """
    Add logging to get/set/delete operations for debugging.
    """
    __slots__ = ()  #␣
↪ůůåĚěçśżéČ¡æšąæIJL'åőđä¿ŃåŔŸéĞŔïjŇåŻääÿžçŻt'æŐěåőđä¿ŃåŇŰæůůåĚěçśżæšąæIJL'äżż

    def __getitem__(self, key):
        print('Getting ' + str(key))
        return super().__getitem__(key)

    def __setitem__(self, key, value):
        print('Setting {} = {!r}'.format(key, value))
        return super().__setitem__(key, value)

    def __delitem__(self, key):
        print('Deleting ' + str(key))
        return super().__delitem__(key)


class SetOnceMappingMixin:
    '''
    Only allow a key to be set once.
    '''
    __slots__ = ()

    def __setitem__(self, key, value):
        if key in self:
            raise KeyError(str(key) + ' already set')
        return super().__setitem__(key, value)


class StringKeysMappingMixin:
    '''
```

```
    Restrict keys to strings only
    '''
    __slots__ = ()

    def __setitem__(self, key, value):
        if not isinstance(key, str):
            raise TypeError('keys must be strings')
        return super().__setitem__(key, value)
```

è£ŹäžŻçśżå■ŢçŃňä¡£çŤĺèţůæĺěæšąæIJL'äżżä¡ŢæĎŕäźL'ïijŇäžŃåőđäÿŁåęĆæđIJä¡ăåŐżåőđä¿ŃåŇŰäżżä
åőČäžňæŸŕçŤĺæĺěéĂŽè£Ğåď'ŽçżğæL'£æĺěåŠŇåĚűäżŰæŸäářĎářźèśąæůůåĚěä¡£çŤĺçŽĎãĂĆä¿ŃåęĆïijŽ

```
class LoggedDict(LoggedMappingMixin, dict):
    pass

d = LoggedDict()
d['x'] = 23
print(d['x'])
del d['x']

from collections import defaultdict

class SetOnceDefaultDict(SetOnceMappingMixin, defaultdict):
    pass


d = SetOnceDefaultDict(list)
d['x'].append(2)
d['x'].append(3)
# d['x'] = 23  # KeyError: 'x already set'
```

è£Źäÿŀä¿Ńå■Řäÿ■ïijŇåŕźěçIJŃåĹřæůůåĚěçśżèů§åĚűäżŰåůšå■ŸåIJĺçŽĎçśż(æŕŤåęĆdictãĂĄdefaultd
çżŞåŘĹåŘŐåřśèČ¡åŔŚæŇěæ■čåÿÿåŁ§æŢĹäžĘãĂĆ

èőĺèőž

æůůåĚěçśżåIJĺæăĞåĞĘåžŞäÿ■å¿Ĺåď'ŽåIJřæŰźéČ¡åĞžçŐřè£ĞïijŇéĂŽåÿÿéČ¡æŸŕçŤĺæĺěåČŔäÿŁéĺćéĆč
åőČäžňäź§æŸŕåď'Žçżğæ L'£çŽĎäÿĂäÿłäÿżèęĄçŤĺéĂŤãĂĆærŤåęĆïijŇå¡Şä¡äçijŰåĘŹç¡ŚçżIJäżčçăĄæŮűåĂŹ
ä¡ăäijŽçżŔåÿÿä¡£çŤĺ socketserver æĺąåĺŮäÿ■çŽĎ ThreadingMixIn
æĺěçżŹåĚűäżŰç¡ŚçżIJçŽÿåĚşçśżåćđåŁăåď'Žçž£çĺŃæŤŕæŇĄãĂĆ
ä¿ŃåęĆïijŇäÿŃéĺćæŸŕäÿĂäÿłåď'Žçž£çĺŃçŽĎXML-RPCæIJ■åŁąïijŽ

```
from xmlrpc.server import SimpleXMLRPCServer
from socketserver import ThreadingMixIn
class ThreadedXMLRPCServer(ThreadingMixIn, SimpleXMLRPCServer):
    pass
```

åŘŇæŮűåIJĺäÿĂäžŻåď'ğđŃåžŞåŠŇæąĘæđűäÿ■äź§äijŽåŔŚçŐřæůůåĚěçśżçŽĎä¡£çŤĺïijŇçŤĺéĂŤåŘŇæ

åŕźäžŐæůůåĚěçśżïijŇæIJL'åĞăçĆźéIJĂèęĄèőřä¡ŕãĂĆéęŰůĚĹæŸŕïijŇæůůåĚěçśżäÿ■èČ¡çŻť'æŐèèćńåő
åĚűæňąïijŇæůůåĚěçśżæšąæIJL'èĞłåůśçŽĎçŁűæĂĄä£ąæĄŕïijŇäź§åřśæŸŕèřť'åőČäżňåźűæšąæIJL'åőŽäźĽ
__init__() æŰźæşŢïijŇåźűäÿŤæšąæIJL'åőđä¿ŃåšđæĂğãĂĆ
è£Źäź§æŸŕäÿžäżĂäźĹæĹŚäżňåIJĺäÿŁéİćæŸŐçąőåőŽäźĽ'äžĘ __slots__ = () ãĂĆ

è£ŸæIJL'äÿĂçğ■åőđçŐřæůůåĚěçśżçŽĎæŰźåijŔåřśæŸŕä¡£çŤĺçśżèčĚéěřåŹĺïijŇåęĆäÿŃæL'Ăçd'žïijŽ

```
def LoggedMapping(cls):
    """çňňäžŇçğ■æŰźåijŔïijŽä¡£çŤĺçśżèčĚéěřåŹĺ"""
    cls_getitem = cls.__getitem__
    cls_setitem = cls.__setitem__
    cls_delitem = cls.__delitem__

    def __getitem__(self, key):
        print('Getting ' + str(key))
        return cls_getitem(self, key)

    def __setitem__(self, key, value):
        print('Setting {} = {!r}'.format(key, value))
        return cls_setitem(self, key, value)

    def __delitem__(self, key):
        print('Deleting ' + str(key))
        return cls_delitem(self, key)

    cls.__getitem__ = __getitem__
    cls.__setitem__ = __setitem__
    cls.__delitem__ = __delitem__
    return cls


@LoggedMapping
class LoggedDict(dict):
    pass
```

è£ŹäÿłæŢĹædIJèů§äźŃåL'■çŽĎæŸŕäÿĂæăůçŽĎïijŇèĂŇäÿŤäÿ■åĘ■éIJĂèęĄä¡£çŤĺåď'ŽçżğæL'£äžĘãĂĆ
åŔĆèĂČ8.13åřŔèŁĆæ§ěçIJŃæŻť'åď'ŽæůůåĚěçśżåŠŇçśżèčĚéěřåŹĺçŽĎä¿Ńå■ŘãĂĆ

8.19 åőđçŐřçŁűæĂĄåŕźèśąæĹŰèĂĚçŁűæĂĄælJž

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłçŁűæĂĄæIJžæĹŰèĂĚæŸŕåIJĺäÿ■åŘŇçŁűæĂĄäÿŃæL'ğèąŇæŞ■ä¡IJçŽĎåŕźèśąïijĺ

èğčåĘşæŰźæąĹ

åIJĺå¿Ĺåď'ŽçĺŃåžŔäÿ■ïijŇæIJL'äžŻåŕźèśąäijŽæăźæ■őçŁűæĂĄçŽĎäÿ■åŘŇæİěæL'ğèąŇäÿ■åŘŇçŽĎæŞı

```
class Connection:
    """æŹőéĂŽæŰźæąĹïijŇåě¡åd'ŽäÿłåĹd'æŰ■èí■åŔěïijŇæŢĹçŐĞä¡ŐäÿŃ~~"""

    def __init__(self):
        self.state = 'CLOSED'

    def read(self):
        if self.state != 'OPEN':
            raise RuntimeError('Not open')
        print('reading')

    def write(self, data):
        if self.state != 'OPEN':
            raise RuntimeError('Not open')
        print('writing')

    def open(self):
        if self.state == 'OPEN':
            raise RuntimeError('Already open')
        self.state = 'OPEN'

    def close(self):
        if self.state == 'CLOSED':
            raise RuntimeError('Already closed')
        self.state = 'CLOSED'
```

è£Źæăůå ĘŹæIJL'å¿Ĺåd'ŽçijžçĆźïijŇéęŰåĚĹæŸŕäžčçăĄåd'łåd'■æİĆäžĘïijŇåě¡åd'ŽçŽĎæİąäžűåĹd'æŰ■åŽääÿžäÿĂäžŽåÿÿèğĄçŽĎæŞ■ä¡IJæŕŤåęĆread()ãĂĄwrite()æŕŔæňąæL'ğèąŇåL'■éČ¡éIJĂèęĄæL'ğèąŇæčĂæş

äÿĂäÿłæŻť'åě¡çŽĎåŁđæşŢæŸŕäÿžæŕŔäÿłçŁűæĂĄåőŽäźL'äÿĂäÿłåŕźèśąïijž

```
class Connection1:
    """æŰřæŰźæąĹâĂŤâĂŤåŕźæŕŔäÿłçŁűæĂĄåőŽäźL'äÿĂäÿłçśż"""

    def __init__(self):
        self.new_state(ClosedConnectionState)

    def new_state(self, newstate):
        self._state = newstate
        # Delegate to the state class

    def read(self):
        return self._state.read(self)

    def write(self, data):
        return self._state.write(self, data)

    def open(self):
        return self._state.open(self)

    def close(self):
```

```
        return self._state.close(self)


# Connection state base class
class ConnectionState:
    @staticmethod
    def read(conn):
        raise NotImplementedError()

    @staticmethod
    def write(conn, data):
        raise NotImplementedError()

    @staticmethod
    def open(conn):
        raise NotImplementedError()

    @staticmethod
    def close(conn):
        raise NotImplementedError()


# Implementation of different states
class ClosedConnectionState(ConnectionState):
    @staticmethod
    def read(conn):
        raise RuntimeError('Not open')

    @staticmethod
    def write(conn, data):
        raise RuntimeError('Not open')

    @staticmethod
    def open(conn):
        conn.new_state(OpenConnectionState)

    @staticmethod
    def close(conn):
        raise RuntimeError('Already closed')


class OpenConnectionState(ConnectionState):
    @staticmethod
    def read(conn):
        print('reading')

    @staticmethod
    def write(conn, data):
        print('writing')
```

```
    @staticmethod
    def open(conn):
        raise RuntimeError('Already open')

    @staticmethod
    def close(conn):
        conn.new_state(ClosedConnectionState)
```

äÿŃéÌćæŸŕä¡£çŤÍæijŤçď žïijŽ

```
>>> c = Connection()
>>> c._state
<class '__main__.ClosedConnectionState'>
>>> c.read()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 10, in read
        return self._state.read(self)
    File "example.py", line 43, in read
        raise RuntimeError('Not open')
RuntimeError: Not open
>>> c.open()
>>> c._state
<class '__main__.OpenConnectionState'>
>>> c.read()
reading
>>> c.write('hello')
writing
>>> c.close()
>>> c._state
<class '__main__.ClosedConnectionState'>
>>>
```

èőÍèőž

åęĆæđIJäžčçăĄäÿ■åĞžçŐřåď'łåď'ŽçŽĎæİąäžűåĹď'æŰ■èŕ■åŔěçŽĎèŕİïijÑäžčçăĄåřšäijŽåŔŸå¿ŮéŽ¿äžě
è£ŹéĞŇçŽĎèğčåĘşæŰźæąĹæŸŕåřĘæŕŔäÿłçŁűæĂĄæĽ¡åŔŰåĞžæİěåőŽäźĽ'æĹŘäÿĂäÿłçśžãĂĆ

è£ŹéĞŇçIJŃäÿŁåŐżæIJL'çĆźåěĞæĂłïijÑæŕŔäÿłçŁűæĂĄåŕźèśąéČ¡åŔłæIJL'éİŹæĂĄæŰźæşŢïijÑåźűæè
åőđéŹĚäÿŁïijÑæL'ĂæIJL'çŁűæĂĄä£ąæĄŕéČ¡åŔłå■ŸåĆĺåIJÍ Connection
åőđä¿Ńäÿ■ãĂĆ åIJĺå§žçśžäÿ■åőŽäźĽ'çŽĎ NotImplementedError
æŸŕäÿžäžĘçąőä£Ìå■ŘçśžåőđçŐřäžĘçŻÿåžŤçŽĎæŰźæşŢãĂĆ è£ŹéĞŇä¡äæĹŰèőÿè£ŸæČşä¡£çŤÍ8.12åřŔèŁĆ

èő¿èőąæÍąåijŔäÿ■æIJL'äÿĂçğ■æÍąåijŔåŔńçŁűæĂĄæÍąåijŔïijÑè£ŹäÿĂåřŔèŁĆçőŮæŸŕäÿĂäÿłåĹİæ■ěåĚ

8.20 éĂŽè£Ğå■ŮçňęäÿšèřČçŤĺářźèśąæŰźæşŢ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłå■Ůçňęäÿšå¡ćåijŔçŽĎæŰźæşŢåŘ■çğřïijŇæČşéĂŽè£ĞåőČèřČçŤĺæ§Řäÿłářźèśąç ŽĎåŕźå

èğčåĘşæŰźæąĹ

æIJĂçőĂå■ŢçŽĎæČĚåĘţïijŇåŔŕäżěä¡£çŤĺ `getattr()` ïijŽ

```
import math

class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __repr__(self):
        return 'Point({!r:},{!r:})'.format(self.x, self.y)

    def distance(self, x, y):
        return math.hypot(self.x - x, self.y - y)


p = Point(2, 3)
d = getattr(p, 'distance')(0, 0)  # Calls p.distance(0, 0)
```

åŔęåď'ŰäÿĂçğ■æŰźæşŢæŸŕä¡£çŤĺ `operator.methodcaller()` ïijŇä¿ŃåęĆïijŽ

```
import operator
operator.methodcaller('distance', 0, 0)(p)
```

å¡Şä¡ăéIJĂèęĄéĂŽè£ĞçŻÿåŘŇçŽĎåŔĆæŢřåď'ŽæňąèřČçŤĺæ§Řäÿłæ ŰźæşŢæŮűïijŇä¡£çŤĺ `operator.methodcaller` åřśå¿ĹæŰźä¿£äžĘãĂĆ æŕŤåęĆä¡ăéIJĂèęĄæŐŠåžŔäÿĂçşżåĹŮçŽĎçĆźïijŇår

```
points = [
    Point(1, 2),
    Point(3, 0),
    Point(10, -3),
    Point(-5, -7),
    Point(-1, 8),
    Point(3, 2)
]
# Sort by distance from origin (0, 0)
points.sort(key=operator.methodcaller('distance', 0, 0))
```

èőĺèőž

èřČçŤĺäÿĂäÿłæŰźæşŢåőđéŹĚäÿŁæŸŕäÿd'éČĺçŃňçńŃæŞ■ä¡IJïijŇçňňäÿĂæ■ěæŸŕæ§ěæL'¿åšđæĂğïijŇçåŻăæ■d'ïijŇäÿžäžĘèřČçŤĺæ§ŘäÿłæŰźæşŢïijŇä¡ăåŕŕäżěéęŰåĚĹéĂŽè£Ğ `getattr()` æĺěæ§ěæL'¿åĹřè£ŹäÿłåšđæĂğïijŇçĎűåŘŐåĘ■åŐżäžěåĞ¡æŢřæŰźåijŔèřČçŤĺåőČå■şåŕŕãĂĆ

`operator.methodcaller()` åĹŻåžžäÿĂäÿłåŕŕèřČçŤĺåŕźèśąïijŇåźűåŘŇæŮűæŔŘä¿ŻæL'ĂæIJL'å£çĎűåŘŐèřČçŤĺçŽĎæŮűåĂŹåŔłéIJĂèęĄåřĘåőđä¿ŃåŕźèśąäijăéĂŠçżŹåőČå■şåŕŕïijŇæŕŤåęĆïijŽ

```
>>> p = Point(3, 4)
>>> d = operator.methodcaller('distance', 0, 0)
>>> d(p)
5.0
>>>
```

éĂŽè£ĞæŰźæşŢåŘ■çğřå■ŮçňęäÿšæİèèřČçŤĺæŰźæşŢéĂŽåÿÿåĞžçŐřåIJĺéIJĂèęĄæĺąæŃ§ `case` èŕ■åŔěæĹŰåőđçŐřèő£éŮőèĂĚæĺąåijŔçŽĎæŮűåĂŹãĂĆ åŔĆèĂČäÿŃäÿĂåřŔèŁĆèŐůåŔŰæŻťåd'ŽéńŸçžğä¿Ńå■ŘãĂĆ

8.21 åőđçŐřèő£éŮőèĂĚæĺąåijŔ

éŮőécŸ

ä¡ăèęĄåd'ĎçŘĘçŤšåd'ğéĞŔäÿ■åŘŃçśżåđŃçŽĎåŕźèśąçżĎæĹŘçŽĎåd'■æİĆæŢřæ■őçżŞæđĎïijŇæŕŔäÿĂ æŕŤåęĆïijŇéĄ■åŐĘäÿĂäÿłæăŚå¡ćçżŞæđĎïijŇçĎűåŘŐæăźæ■őæŕŔäÿłèŁĆçĆźçŽĎçŻÿåžŤçŁűæĂĄæL'ğèąŃ

èğčåĘşæŰźæąĹ

è£ŹéĞŇéĄĞåĹřçŽĎéŮőécŸåIJĺçijŰçĺŃécĘå§§äÿ■æŸŕå¿ĹæŹőéĄ■çŽĎïijŇæIJL'æŮűåĂŹäijŽæđĎåžžä åĄĞèő¿ä¡ăèęĄåĘŹäÿĂäÿłèąĺçd'žæŢřå■ęèąĺè¿¿åijŔçŽĎçĺŃåžŔïijŇéĆčäźĹä¡ăåŔřèČ¡éIJĂèęĄåőŽäźL'åęĆäÿŃ

```
class Node:
    pass

class UnaryOperator(Node):
    def __init__(self, operand):
        self.operand = operand

class BinaryOperator(Node):
    def __init__(self, left, right):
        self.left = left
        self.right = right

class Add(BinaryOperator):
    pass
```

```
class Sub(BinaryOperator):
    pass

class Mul(BinaryOperator):
    pass

class Div(BinaryOperator):
    pass

class Negate(UnaryOperator):
    pass

class Number(Node):
    def __init__(self, value):
        self.value = value
```

çĎűåŘŐåĹľçŤĺè£ŹäžŽçśżæđĎåžžåţŇåěŮæŢřæ■őçżŞæđĎïijŇåęĆäÿŃæĽ˙Ăçđ˙žïijŽ

```
# Representation of 1 + 2 * (3 - 4) / 5
t1 = Sub(Number(3), Number(4))
t2 = Mul(Number(2), t1)
t3 = Div(t2, Number(5))
t4 = Add(Number(1), t3)
```

è£ŹæăůåĄŽçŽĎéŮőécŸæŸŕåŕźäžŐæŕŔäÿłèąĺè¿¿åijŔïijŇæŕŔæňąéČ¡èęĄéĞ■æŰřåőŽäźĽäÿĂéĄ■ïijŇæl
è£ŹéĞŇæĹŚäžňä¡£çŤĺèő£éŮőèĂĚæĺąåijŔåŔŕäžěè¿¿åĹře£ŹæăůçŽĎçŽőçŽĎïijŽ

```
class NodeVisitor:
    def visit(self, node):
        methname = 'visit_' + type(node).__name__
        meth = getattr(self, methname, None)
        if meth is None:
            meth = self.generic_visit
        return meth(node)

    def generic_visit(self, node):
        raise RuntimeError('No {} method'.format('visit_' +␣
↪type(node).__name__))
```

äÿžäžĘä¡£çŤĺè£ŹäÿłçśżïijŇåŔŕäžěåőŽäźĽäÿĂäÿłçśżçżğæĽ£åőČåźűäÿŤåőđçŐřåŘĎçğ■
visit_Name() æŰźæşŢïijŇåĚűäÿ■NameæŸŕnodeçśżåđŃãĂĆ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăæČşæśĆèąĺè¿¿åijŔçŽĎåĂijïijŇåŔŕäžěè£ŹæăůåĘŹïijŽ

```
class Evaluator(NodeVisitor):
    def visit_Number(self, node):
        return node.value

    def visit_Add(self, node):
        return self.visit(node.left) + self.visit(node.right)

    def visit_Sub(self, node):
```

```
        return self.visit(node.left) - self.visit(node.right)

    def visit_Mul(self, node):
        return self.visit(node.left) * self.visit(node.right)

    def visit_Div(self, node):
        return self.visit(node.left) / self.visit(node.right)

    def visit_Negate(self, node):
        return -node.operand
```

ä¡£çŤĺçd'žä¿ŃïijŽ

```
>>> e = Evaluator()
>>> e.visit(t4)
0.6
>>>
```

ä¡IJäÿžäÿĂäÿłäÿ■åŘŇçŽĎä¿Ńå■ŘïijŇäÿŃéİćåőŽäźL'äÿĂäÿłçśżåIJĺäÿĂäÿłæăĹäÿŁéİćårĘäÿĂäÿłèąĺè¿¿å

```
class StackCode(NodeVisitor):
    def generate_code(self, node):
        self.instructions = []
        self.visit(node)
        return self.instructions

    def visit_Number(self, node):
        self.instructions.append(('PUSH', node.value))

    def binop(self, node, instruction):
        self.visit(node.left)
        self.visit(node.right)
        self.instructions.append((instruction,))

    def visit_Add(self, node):
        self.binop(node, 'ADD')

    def visit_Sub(self, node):
        self.binop(node, 'SUB')

    def visit_Mul(self, node):
        self.binop(node, 'MUL')

    def visit_Div(self, node):
        self.binop(node, 'DIV')

    def unaryop(self, node, instruction):
        self.visit(node.operand)
        self.instructions.append((instruction,))

    def visit_Negate(self, node):
```

```
        self.unaryop(node, 'NEG')
```

ä¡£çŤĺçd'žä¿ŃïijŽ

```
>>> s = StackCode()
>>> s.generate_code(t4)
[('PUSH', 1), ('PUSH', 2), ('PUSH', 3), ('PUSH', 4), ('SUB',),
('MUL',), ('PUSH', 5), ('DIV',), ('ADD',)]
>>>
```

èőĺèőž

åĹŽåijĂåğŃçŽĎæŮűåĂŹä¡ăåŔŕèČ¡äijŽåĘŹåd'ğéĞŔçŽĎif/elseèŕ■åŔěæİěåőđçŐřïijŇ è£ŹéĞŇèő£éŮőèĂĚæĺąåijŔçŽĎåě¡åd'ĎåŕśæŸŕéĂŽè£Ğ `getattr()` æİèèŐůåŔŰçŻÿåžŤçŽĎæŰźæşŢïijŇåźűåĹl'çŤĺéĂŠå¡ŠæİéęĄ■åŐĘæL'ĂæIJL'çŽĎèŁĆçĆźïijŽ

```
def binop(self, node, instruction):
    self.visit(node.left)
    self.visit(node.right)
    self.instructions.append((instruction,))
```

è£ŸæIJL'äÿĂçĆźéIJĂèęĄæŇĞåĞžçŽĎæŸŕïijŇè£Źçğ■æŁĂæIJŕäź§æŸŕåőđçŐřåĚűäżŰèŕ■èĺĂäÿ■switch æŕŤåęĆïijŇåęĆæđIJä¡ăæ■čåIJĺåĘŹäÿĂäÿłHTTPæąĘæđűïijŇä¡ăåŔŕèČ¡äijŽåĘŹè£ŹæăůäÿĂäÿłèŕůæśĆåĹĘåŔ

```
class HTTPHandler:
    def handle(self, request):
        methname = 'do_' + request.request_method
        getattr(self, methname)(request)
    def do_GET(self, request):
        pass
    def do_POST(self, request):
        pass
    def do_HEAD(self, request):
        pass
```

è£éŮőèĂĚæĺąåijŔäÿĂäÿłçijžçĆźåřśæŸŕåőČäÿěéĞ■ä¿İèţŰéĂŠå¡ŠïijŇåęĆæđIJæŢřæ■őçżŞæđĎåţŇåěŮ æIJL'æŮűåĂŹäijŽèűĚè£ĞPythonçŽĎéĂŠå¡ŠæůśåžçéŹŔåĹű(åŔĆèĂČ `sys.getrecursionlimit()`)ãĂĆ

åŔŕäžěåŔĆçĚğ8.22åřŔèŁĆïijŇåĹl'çŤĺçŤ§æĹŘåŹĺæĹŰè£■äžčåŹĺæİěåőđçŐřéİđéĂŠå¡ŠéĄ■åŐĘçőŮæşŢ

åIJĺèů§èğčæđŘåŠŇçijŰèŕŚçŻÿåĚşçŽĎçijŰçĺŃäÿ■ä¡£çŤĺèő£éŮőèĂĚæĺąåijŔæŸŕéİđåÿÿåÿÿèğĄçŽĎãĂĆ PythonæIJňèžńçŽĎ `ast` æĺąåİŮåĂijçŽĎåĚşæşĺäÿŃïijŇåŔŕäžěåŐžçIJŃçIJŃæžŘçăĄãĂĆ 9.24åřŔèŁĆæijŤçd'žäžĘäÿĂäÿłåĹl'çŤĺ `ast` æĺąåİŮæİěåd'ĎçŘĘPythonæžŘäžčçăĄçŽĎä¿Ńå■ŘãĂĆ

8.22 äÿ■çŤĺéĂŠå¡Šåőđç Őřèő£éŮőèĂĚæĺąåijŔ

éŮőécŸ

ä¡ăä¡£çŤĺèő£éŮőèĂĚæĺąåijŔéĄ■åŐĘäÿĂäÿłå¿Ĺæůśç ŽĎåţŇåěŮæăŠå¡ćæŢřæ■őçżŞæđĎïijŇåźűäÿŤåŻăää¡ăæČşæűĹéŹd'éĂŠå¡Šïij Ňåźűå ŘŇæŮűä£ÍæŇĄèő£éŮőèĂĚçijŰçÍŃæĺąåijŔãĂĆ

èğčåĘşæŰźæąĹ

éĂŽè£Ğåůğåę ZçŽĎä¡£çŤĺçŤ§æĹŘåŹĺåŔŕäžěåIJĺæăŠéĄ■åŐĘæĹŰæŘIJçt'ćçőŮæşŢäÿ■æűĹéŹd'éĂŠå¡Šåij Í8.21åřŔèŁĆäÿ■ïijŇæĹŚäžňçžŹåĞžäžĘäÿĂäÿłèő£éŮőèĂĚçśžãĂĆäÿŃéÍćæĹŚäžňåĽ'çŤĺäÿĂäÿłæăĹåŠŇçŤ§æĹŘåŹĺéĞ■æŰřåődçŐřè£ŹäÿłçśžïijŽ

```
import types

class Node:
    pass

class NodeVisitor:
    def visit(self, node):
        stack = [node]
        last_result = None
        while stack:
            try:
                last = stack[-1]
                if isinstance(last, types.GeneratorType):
                    stack.append(last.send(last_result))
                    last_result = None
                elif isinstance(last, Node):
                    stack.append(self._visit(stack.pop()))
                else:
                    last_result = stack.pop()
            except StopIteration:
                stack.pop()

        return last_result

    def _visit(self, node):
        methname = 'visit_' + type(node).__name__
        meth = getattr(self, methname, None)
        if meth is None:
            meth = self.generic_visit
        return meth(node)

    def generic_visit(self, node):
        raise RuntimeError('No {} method'.format('visit_' +␣
↪type(node).__name__))
```

åęĆæđIJä¡ää¡£çŤĺè£Źäÿłçśżïij Ňäź§èČ¡è¿¿åĹřçŻÿåŘŃçŽĎæŢĹæđIJãĂĆäžŃåőđäÿŁä¡ääőŇåĚĺåŔŕäżěåř
èĂČèŹŚåęĆäÿŃäżčçăĄïijŇéĄ■åŐĘäÿĂäÿłèąĺè¿¿åijŔçŽĎæăŚïijŽ

```
class UnaryOperator(Node):
    def __init__(self, operand):
        self.operand = operand

class BinaryOperator(Node):
    def __init__(self, left, right):
        self.left = left
        self.right = right

class Add(BinaryOperator):
    pass

class Sub(BinaryOperator):
    pass

class Mul(BinaryOperator):
    pass

class Div(BinaryOperator):
    pass

class Negate(UnaryOperator):
    pass

class Number(Node):
    def __init__(self, value):
        self.value = value

# A sample visitor class that evaluates expressions
class Evaluator(NodeVisitor):
    def visit_Number(self, node):
        return node.value

    def visit_Add(self, node):
        return self.visit(node.left) + self.visit(node.right)

    def visit_Sub(self, node):
        return self.visit(node.left) - self.visit(node.right)

    def visit_Mul(self, node):
        return self.visit(node.left) * self.visit(node.right)

    def visit_Div(self, node):
        return self.visit(node.left) / self.visit(node.right)

    def visit_Negate(self, node):
        return -self.visit(node.operand)
```

```
if __name__ == '__main__':
    # 1 + 2*(3-4) / 5
    t1 = Sub(Number(3), Number(4))
    t2 = Mul(Number(2), t1)
    t3 = Div(t2, Number(5))
    t4 = Add(Number(1), t3)
    # Evaluate it
    e = Evaluator()
    print(e.visit(t4))  # Outputs 0.6
```

åęĆæđIJåţŇåěŮåśĆæňąåď'łæůśéĆčäźĹäÿŁè£řçŽĎEvaluatorårŕšäijŽåď'śæŢĹïijŽ

```
>>> a = Number(0)
>>> for n in range(1, 100000):
... a = Add(a, Number(n))
...
>>> e = Evaluator()
>>> e.visit(a)
Traceback (most recent call last):
...
    File "visitor.py", line 29, in _visit
return meth(node)
    File "visitor.py", line 67, in visit_Add
return self.visit(node.left) + self.visit(node.right)
RuntimeError: maximum recursion depth exceeded
>>>
```

çŐřåIJĺæĹŚäżňçĺ■å¿ő㣩æŤźäÿŃäÿŁéİćçŽĎEvaluatorïijŽ

```
class Evaluator(NodeVisitor):
    def visit_Number(self, node):
        return node.value

    def visit_Add(self, node):
        yield (yield node.left) + (yield node.right)

    def visit_Sub(self, node):
        yield (yield node.left) - (yield node.right)

    def visit_Mul(self, node):
        yield (yield node.left) * (yield node.right)

    def visit_Div(self, node):
        yield (yield node.left) / (yield node.right)

    def visit_Negate(self, node):
        yield - (yield node.operand)
```

åĘ■æňąè£ŘèąŇïijŇåŕšäÿ■äijŽæŁěéŤŹäžĘïijŽ

```
>>> a = Number(0)
>>> for n in range(1,100000):
...     a = Add(a, Number(n))
...
>>> e = Evaluator()
>>> e.visit(a)
4999950000
>>>
```

åęĆæđIJä¡ăè£ŸæČşæůżåŁăåĚűäżŰèĞłåőŽäźĽéĂżè¿Śäź§æšąéŰőécŸïijŽ

```
class Evaluator(NodeVisitor):
    ...
    def visit_Add(self, node):
        print('Add:', node)
        lhs = yield node.left
        print('left=', lhs)
        rhs = yield node.right
        print('right=', rhs)
        yield lhs + rhs
    ...
```

äÿŃéÌćæŸŕçőĂå■ŢçŽĎæţŃèŕŢïijŽ

```
>>> e = Evaluator()
>>> e.visit(t4)
Add: <__main__.Add object at 0x1006a8d90>
left= 1
right= -0.4
0.6
>>>
```

èőÍèőž

è£ŹäÿĂåřŔèŁĆæĹŚäžňæijŤçd'žäžĘçŤ§æĹŘåŹÍåŠŇå■ŔçÍŃåIJÍçÍŃåžŔæŐğåĽűæţĄæŰźéÌćçŽĎåijžåd'ğ
éĄ£åĚ■éĂŠå¡ŠçŽĎäÿĂäÿłéĂŽåÿÿæŰźæşŢæŸŕä¡£çŤÍäÿĂäÿłæăĹæĹŰéŸ§åĹŮçŽĎæŢřæ■őçżŞædĎãĂĆ
ä¿ŃåęĆïijŇæůšåžęäijŸåĚĹçŽĎéĄ■åŐĘçőŮæşŢïijŇçňňäÿĂæňąçćřåĽřäÿĂäÿłèŁĆçĆźæŮűåřĘåĚűåŐŃåĚěæă
æŰźæşŢçŽĎæäÿå£ČæĂÌèůŕårśæŸŕè£Źæăůã ĂĆ

åŔęåd'ŰäÿĂäÿłéIJĂèęĄçŘĘèğčçŽĎårśæŸŕçŤ§æĹŘåŹÍäÿ■yieldèŕ■åŔěãĂĆå¡ŞçćřåĽřyieldèŕ■åŔěæŮűïij
äÿŁéÌćçŽĎä¿Ńå■Ră¡£çŤÍè£ŹäÿłæŁĂæIJŕæÌěäžčæŻ£äžĘéĂŠå¡ŠãĂĆä¿ŃåęĆïijŇäźŃåL'■æĹŚäžňæŸŕè£Źæăů

```
value = self.visit(node.left)
```

çŐřåIJÍæ■ćæĹŘyieldèŕ■åŔëijŽ

```
value = yield node.left
```

åőČäijŽåřĘ `node.left` è£ŤåŻđçżŹ `visit()` æŰźæşŢïijŇçĎűåŘŐ `visit()` æŰźæşŢèřČçŤĺéĆčäÿłèŁĆçĆźçŻÿåžŤçŽĎ `visit_Name()` æŰźæşŢãĂĆ yield-æŽĆæŮűåřĘçĺŃåžŔæŐğåĹűåŹĺèőľ åĞžçżŹèřČçŤĺèĂËïijŇå¡ŞæĽğèąŇåőŇåŘŐïijŇçżŞæđIJäijŽèţŃåĂijçżŹv

çIJŃåőŇè£ŹäÿĂåřŔèŁĆïijŇä¡ăáź§èőÿæČşåŐżåŕżæĽ¿åĚűåőČæšąæIJL'yieldèŕ■åŘěçŽĎæŰźæąĹãĂĆä¡Ł ä¿ŃåęĆïijŇäÿžäžĘæűĹéŹď'éĂŠå¡ŠïijŇä¡ăå£ĚéążèęĄçżt'æŁď'äÿĂäÿłæăĹçżŞæđĎïijŇåęĆæđIJäÿ■ä¡£çŤĺçŤ§a åőđéŹĚäÿŁïijŇä¡£çŤĺyieldèŕ■åŘěåŔŕäžěèőľ'ä¡ăåĘŹåĞžéİđåÿÿæijĆäžőçŽĎäżčçăĄïijŇåőČæűĹéŹď'äžĘéĂŠå¡

8.23 å¿łçŐŕåijŢçŤĺæŢřæ■őçżŞæđĎçŽĎåĘĚå■ŸçőąçŘĘ

éŮőécŸ

ä¡ăçŽĎçĺŃåžŔåĹŻåżžäžĘå¿Ĺåď'Žå¿łçŐŕåijŢçŤĺæŢřæ■őçżŞæđĎ(æŕŤåęĆæăŚãĂĄåŻ¿ãĂĄèğĆår§èĂĚæ

èğčåĘşæŰźæąĹ

äÿĂäÿłçőĂå■ŢçŽĎå¿łçŐŕåijŢçŤĺæŢřæ■őçżŞæđĎä¿Ńå■ŘåŕśæŸŕäÿĂäÿłæăŚå¡ćçżŞæđĎïijŇåŔŇäžšèŁĆ è£Źçğ■æČĚåĘţäÿŃïijŇåŔŕäżěèĂČèŹŚä¡£çŤĺ `weakref` åžŞäÿ■çŽĎåijśåijŢçŤĺãĆĆä¿ŃåęĆïijŽ

```
import weakref

class Node:
    def __init__(self, value):
        self.value = value
        self._parent = None
        self.children = []

    def __repr__(self):
        return 'Node({!r:})'.format(self.value)

    # property that manages the parent as a weak-reference
    @property
    def parent(self):
        return None if self._parent is None else self._parent()

    @parent.setter
    def parent(self, node):
        self._parent = weakref.ref(node)

    def add_child(self, child):
        self.children.append(child)
        child.parent = self
```

è£Źçğ■æŸŕæČşæŰźåijŔåĚĄèőÿparentéİŹéżŸçżĹæ■ćãĂĆä¿ŃåęĆïijŽ

```
>>> root = Node('parent')
>>> c1 = Node('child')
```

```
>>> root.add_child(c1)
>>> print(c1.parent)
Node('parent')
>>> del root
>>> print(c1.parent)
None
>>>
```

èőĺèőž

å¿łçŐŕåijŢçŤĺçŽĎæŢřæ■őçżŞæđĎåIJíPythonäÿ■æŸŕäÿĂäÿłå¿ĹæčŸæĽŃçŽĎéŮőécŸïijŇåŽääÿžæ■čåÿä¿ŃåęĆèĂČèŹŚåęĆäÿŃäžčçăĄïijŽ

```
# Class just to illustrate when deletion occurs
class Data:
    def __del__(self):
        print('Data.__del__')

# Node class involving a cycle
class Node:
    def __init__(self):
        self.data = Data()
        self.parent = None
        self.children = []

    def add_child(self, child):
        self.children.append(child)
        child.parent = self
```

äÿŃéIćæĹŚäžňä¡£çŤĺè£Źäÿłäžčçă Ąæİěå ĄŽäÿĂäžŽåđČåIJ¿åŽđæŤűèŕŢétŇïijŽ

```
>>> a = Data()
>>> del a # Immediately deleted
Data.__del__
>>> a = Node()
>>> del a # Immediately deleted
Data.__del__
>>> a = Node()
>>> a.add_child(Node())
>>> del a # Not deleted (no message)
>>>
```

åŔŕäžěçIJŃåĹřïijŇæIJĂåŘŐäÿĂäÿłçŽĎåĹăéŹđ'æŮűæL'Şå■řèŕ■åŘěæšąæIJL'åĞžçŐřãĂĆåŐ§åŽăæŸŕPy
å¡ŞäÿĂäÿłårźèšąçŽĎåijŢçŤĺæŢřåŘŸæĹŘ0çŽĎæŮűåĂŹæL'■äijŽçńŇå■şåĹăéŹđ'æŐL'ãĂĆèĂŇårźäžŐå¿łçŐŕ
åŽăæ■đ'ïijŇåIJĺäÿŁéIćä¿Ńå■Řäÿ■æIJĂåŘŐéČĺåĹĘïijŇçĹűèŁĆçĆźåŠŇå■l'å■ŘèŁĆçĆźäžŠçŽÿæŃěæIJL'årźa

PythonæIJL'åŔęåđ'ŰçŽĎåđČåIJ¿åŽđæŤűåŹĺæİěäÿŞéŮĺéŠĹårźå¿łçŐŕåijŢçŤĺçŽĎïijŇä¡ĘæŸŕä¡ăæřÿè£IJ
åŔęåđ'Űä¡ăè£ŸåŔŕäžěæL'ŃåŁĺçŽĎèğęåŔŠåőČïijŇä¡ĘæŸŕäžčçăĄçIJŃäÿŁåŐžå¿ĹæŇńïijŽ

```
>>> import gc
>>> gc.collect() # Force collection
Data.__del__
Data.__del__
>>>
```

åęĆæđIJå¿łçŐŕåijŢçŤĺçŽĎåŕźèśąèĞłåůśè£ŸåőŽäźĽäžEèĞłåůśçŽĎ __del__() æŰźæşŢïijŇéĆčäźĹäijŽèőľæČĚåĘţåŔŸå¿ŮæŻť'çş§çşŢãĂĆ åĄĞèő¿ä¡ääČŔäÿŃéİćè£ŹæăůçżŹNodeåőŽäźĽèĞłåůśçŽĎ __del__() æŰźæşŢïijŽ

```
# Node class involving a cycle
class Node:
    def __init__(self):
        self.data = Data()
        self.parent = None
        self.children = []

    def add_child(self, child):
        self.children.append(child)
        child.parent = self

    # NEVER DEFINE LIKE THIS.
    # Only here to illustrate pathological behavior
    def __del__(self):
        del self.data
        del.parent
        del.children
```

è£Źçğ■æČĚåĘţäÿŃïijŇåđČåIJ¿åŻđæŤűæřÿè£IJéČ¡äÿ■äijŽåŐżåŻđæŤűè£ŹäÿłåŕźèśąçŽĎïijŇè£ŸäijŽåŕij åęĆæđIJä¡äèŕŢçİĂåŐżè£ŘèąŇåőČäijŽåŔŚçŐřïijŇData.__del__ æűĹæĄŕæřÿè£IJäÿ■äijŽåĞžçŐřäžE,çŤŽèĞşåIJĺä¡ääijžåĹűåĘĚå■ŸåŻđæŤűæŮűïijŽ

```
>>> a = Node()
>>> a.add_child(Node())
>>> del a # No message (not collected)
>>> import gc
>>> gc.collect() # No message (not collected)
>>>
```

åijśåijŢçŤĺæűĹéŹď'äžEåijŢçŤĺå¿łçŐŕçŽĎè£ŹäÿłéŮőécŸïijŇæIJňèť'ĺæİèèőšïijŇåijśåijŢçŤĺåřśæŸŕäÿĂäÿ ä¡ääŔŕäžěéĂŽè£Ğ weakref æİèåĹŽåžžåijśåijŢçŤĺãĂĆä¿ŃåęĆïijŽ

```
>>> import weakref
>>> a = Node()
>>> a_ref = weakref.ref(a)
>>> a_ref
<weakref at 0x100581f70; to 'Node' at 0x1005c5410>
>>>
```

äÿžäžĘèő£éŮőåijśåijŢçŤĺæL'ĂåijŢçŤĺçŽĎåŕźèśąïijŇä¡ääŔŕäžěåČŔåĞ¡æŢřäÿĂæăůåŐżèřČçŤĺåőČå■şåŔ

çŤšäžŐåŐ§åğŃåŕźèšąçŽĎåijŢçŤĺèőąæŢřæšąæIJL'åćđåŁäïijŇéĆčäźĹåŕšåŔŕäžěåŐżåĹäéŹđ'åőČäžĘãĂĆä¿Ńå

```
>>> print(a_ref())
<__main__.Node object at 0x1005c5410>
>>> del a
Data.__del__
>>> print(a_ref())
None
>>>
```

éĂŽè£Ğè£ŹéĞŇæijŢçđ'žçŽĎåijśåijŢçŤĺæŁĂæIJŕïijŇä¡ääijŽåŔŚçŐřäÿ■åĘ■æIJL'å¿łçŐŕåijŢçŤĺéŮőécŸ
ä¡ăè£ŸèČ¡åŔĆèĂČ8.25åřŔèŁĆåĚşäžŐåijśåijŢçŤĺçŽĎåŔęåđ'ŰäÿĂäÿłä¿Ńå■ŘãĂĆ

8.24 èőľçśżæŤŕæŇĄæŕŤè¿ČæŞ■ä¡IJ

éŮőécŸ

ä¡ăæČşèőľæ§ŘäÿłçśżçŽĎåőđä¿ŃæŤŕæŇĄæăĞåĞĘçŽĎæŕŤè¿Čè£ŘçőŮ(æŕŤåęĆ>=,!=,<=,<ç■L')ïijŇä¡Ę

èğčåĘşæŰźæąĹ

PythonçśżåŕźæŕŕŔäÿłæŕŤè¿ČæŞ■ä¡IJéČ¡éIJĂèęĄåőđçŐřäÿĂäÿłçL'źæőŁæŰźæşŢæİěæŤŕæŇĄãĂĆ
ä¿ŃåęĆäÿžäžĘæŤŕæŇĄ>=æŞ■ä¡IJçňęïijŇä¡ăéIJĂèęĄåőŽäźL'äÿĂäÿł __ge__()
æŰźæşŢãĂĆ åř¡çőąåőŽäźL'äÿĂäÿłæŰźæşŢæšąäžĂäźĹéŮőécŸïijŇä¡ĘåęĆæđIJèęĄä¡ăåőđçŐřæL'ĂæIJL'åŔŕè

èčĚéěřåŹÍ `functools.total_ordering` åřśæŸŕçŤĺæİěçőĂåŇŰè£Źäÿłåđ'ĎçŘĘçŽĎãĂĆ
ä¡£çŤĺåőĆæİěèčĚéěřäÿĂäÿłæİěïijŇä¡ăåŔłéIJĂåőŽäźL'äÿĂäÿł __eq__() æŰźæşŢïijŇ
åđ'ŰåŁăåĚűäžŰæŰźæşŢ(__lt__, __le__, __gt__, or __ge__)äÿ■çŽĎäÿĂäÿłå■şåŔŕãĂĆ
çĎűåŘŐèčĚéěřåŹĺäijŽèĞłåŁĺäÿžä¡ăąąńåĚĚåĚűåőĆærŤè¿ČæŰźæşŢãĂĆ

ä¡IJäÿžä¿Ńå■ŘïijŇæĹŚäžñædĎåžžäÿĂäžŻæĹ£å■ŘïijŇçĎűåŘŐçżŹåőĆäžñáćđåŁääÿĂäžŻæĹ£éŮt'ïijŇæ

```
from functools import total_ordering

class Room:
    def __init__(self, name, length, width):
        self.name = name
        self.length = length
        self.width = width
        self.square_feet = self.length * self.width

@total_ordering
class House:
    def __init__(self, name, style):
        self.name = name
        self.style = style
        self.rooms = list()
```

```
    @property
    def living_space_footage(self):
        return sum(r.square_feet for r in self.rooms)

    def add_room(self, room):
        self.rooms.append(room)

    def __str__(self):
        return '{}: {} square foot {}'.format(self.name,
                self.living_space_footage,
                self.style)

    def __eq__(self, other):
        return self.living_space_footage == other.living_space_
↪footage

    def __lt__(self, other):
        return self.living_space_footage < other.living_space_
↪footage
```

è£ŹéĞŇæĹŚäżňåŕŁæŸŕçżŹHouseçśżåőŽäźĽ'äžĘäÿd'äÿłæŰźæşŢïijŽ__eq__() åŠŇ __lt__() ïijŇåőČåřśèČ¡æŤŕæŇĄæL'ĂæIJL'çŽĎæŕŤè¿ČæŞ■ä¡IJïijŽ

```
# Build a few houses, and add rooms to them
h1 = House('h1', 'Cape')
h1.add_room(Room('Master Bedroom', 14, 21))
h1.add_room(Room('Living Room', 18, 20))
h1.add_room(Room('Kitchen', 12, 16))
h1.add_room(Room('Office', 12, 12))
h2 = House('h2', 'Ranch')
h2.add_room(Room('Master Bedroom', 14, 21))
h2.add_room(Room('Living Room', 18, 20))
h2.add_room(Room('Kitchen', 12, 16))
h3 = House('h3', 'Split')
h3.add_room(Room('Master Bedroom', 14, 21))
h3.add_room(Room('Living Room', 18, 20))
h3.add_room(Room('Office', 12, 16))
h3.add_room(Room('Kitchen', 15, 17))
houses = [h1, h2, h3]
print('Is h1 bigger than h2?', h1 > h2) # prints True
print('Is h2 smaller than h3?', h2 < h3) # prints True
print('Is h2 greater than or equal to h1?', h2 >= h1) # Prints False
print('Which one is biggest?', max(houses)) # Prints 'h3: 1101-
↪square-foot Split'
print('Which is smallest?', min(houses)) # Prints 'h2: 846-square-
↪foot Ranch'
```

èőĺèőž

åĚűåőđ total_ordering èčĚéěřåŹĺäź§æšąéĆčäźĹçěđçğŸãĂĆ
åőČåřśæŸŕåőŽäźĽäžĘäÿĂäÿłäžŐæŕŔäÿłæŕŤè¿ČæŤŕæŇĄæŰźæşŢåĹřæĽĂæIJĽéIJĂèęĄåőŽäźĽçŽĎåĚűäžŰ
æŕŤåęĆä¡ăåőŽäźĽäžĘ __le__() æŰźæşŢïijŇéĆčäźĹåőČåřśèćńçŤĺæĺěæđĎåžžæĽĂæIJĽåĚűäžŰçŽĎéIJĂè
åőđéŹĚäÿŁåřśæŸŕåIJĺçśžéĞŇéİćåČŔäÿŃéİćè£ŹæăůåőŽäźĽäžĘäÿĂäžŽçĽźæőŁæŰźæşŢïijŽ

```
class House:
    def __eq__(self, other):
        pass
    def __lt__(self, other):
        pass
    # Methods created by @total_ordering
    __le__ = lambda self, other: self < other or self == other
    __gt__ = lambda self, other: not (self < other or self == other)
    __ge__ = lambda self, other: not (self < other)
    __ne__ = lambda self, other: not self == other
```

å¡ŞçĎűïijŇä¡äèĞłåůśåŐžåĘŹäź§å¿ĹåőźæŸşïijŇä¡ĘæŸŕä¡£çŤĺ @total_ordering
åŔŕäžěçőĂåŇŰäžčçăĄïijŇä¡ŢäźŘèĂŇäÿ■äÿžåŚćãĂĆ

8.25 åĹŻåžžçijŞå■Ÿåőđä¿Ń

éŮőécŸ

åIJĺåĹŻåžžäÿĂäÿłçśžçŽĎåŕźèśąæŮűïijŇåęĆæđIJäźŃåL'■ä¡£çŤĺåŘŇæăůåŔĆæŢřåĹŻåžžè£Ğè£Źäÿłåŕźè
ä¡äæČşè£ŤåŽđåőČçŽĎçijŞå■ŸåijŢçŤĺãĂĆ

èğčåĘşæŰźæąĹ

è£Źçğ■éĂŽåÿÿæŸŕåŽääÿžä¡ăåÿŇæIJŻçŽÿåŘŇåŔĆæŢřåĹŻåžžçŽĎåŕźèśąæŮűå■Ţä¿ŃçŽĎãĂĆ
åIJĺå¿Ĺåď'ŽåžŞäÿ■éČ¡æIJĽăőđéŹĚçŽĎä¿Ńå■ŘïijŇæŕŤåęĆ logging
æĺąåİŮïijŇä¡£çŤĺçŽÿåŘŇçŽĎåŘ■çğřåĹŻåžžçŽĎ logger åőđä¿Ńæřÿè£IJåŔłæIJĽäÿĂäÿłãĂĆä¿ŃåęĆïijŽ

```
>>> import logging
>>> a = logging.getLogger('foo')
>>> b = logging.getLogger('bar')
>>> a is b
False
>>> c = logging.getLogger('foo')
>>> a is c
True
>>>
```

äÿžäžĘè¿¿åĹře£ŹæăůçŽĎæŢĹæđIJïijŇä¡äéIJĂèęĄä¡£çŤĺäÿĂäÿłåŠŇçśžæIJňèžńåĹĘåijĂçŽĎåůěåŐĆåĞ

```
# The class in question
class Spam:
    def __init__(self, name):
        self.name = name

# Caching support
import weakref
_spam_cache = weakref.WeakValueDictionary()
def get_spam(name):
    if name not in _spam_cache:
        s = Spam(name)
        _spam_cache[name] = s
    else:
        s = _spam_cache[name]
    return s
```

çĎűåŘŐåĄŽäÿĂäÿłæţŃèŕŢïijŇä¡ăäijŽåŔŞçŐřèů§äźŃåL’■éĆčäÿłæŮěå£ŮåŕźèśąçŽĎåĹŻåžžèąŇäÿžæŸŕ

```
>>> a = get_spam('foo')
>>> b = get_spam('bar')
>>> a is b
False
>>> c = get_spam('foo')
>>> a is c
True
>>>
```

èőĺèőž

çijŰåĘŹäÿĂäÿłåůěåŐĆåĞ¡æŢřæİěä£őæŤźæŹőéĂŽçŽĎåőđä¿ŃåĹŻåžžèąŇäÿžéĂŽåÿÿæŸŕäÿĂäÿłæŕŤè¿ä¡ĘæŸŕæĹŚäžňè£ŸèČ¡åŘęæL’¿åĹřæŻť’äijŸéŻĚçŽĎèğčåĘşæŰźæąĹåŚćïij§

ä¿ŃåęĆïijŇä¡ăåŔŕèČ¡äijŽèĂČèŹŚéĞ■æŰřåőŽäźL’çśżçŽĎ `__new__()` æŰźæşŢïijŇářšåČŔäÿŃéİćè£Źæăůïij Ž

```
# Note: This code doesn't quite work
import weakref

class Spam:
    _spam_cache = weakref.WeakValueDictionary()
    def __new__(cls, name):
        if name in cls._spam_cache:
            return cls._spam_cache[name]
        else:
            self = super().__new__(cls)
            cls._spam_cache[name] = self
            return self
    def __init__(self, name):
        print('Initializing Spam')
```

```
        self.name = name
```

åĹİçIJŃèţůæİěåě¡åČŔåŔŕäžěè¿¿åĽřéćĎæIJ§æŢĹæđIJïijŇä¡ĘæŸŕéŮőéćŸæŸŕ
__init__() æŕŘæňąéČ¡äijŽèćńěřČçŤĺïijŇäÿ■çőąè£Źäÿłåőđä¿ŃæŸŕåŘęèćńçijŞå■ŸäžĘãĂĆä¿ŃåęĆïijŽ

```
>>> s = Spam('Dave')
Initializing Spam
>>> t = Spam('Dave')
Initializing Spam
>>> s is t
True
>>>
```

è£ŹäÿłæĹŰèőÿäÿ■æŸŕä¡ăæČşèęĄçŽĎæŢĹæđIJïijŇåŽăæ■d'è£Źçğ■æŰźæşŢåźűäÿ■åŔŕåŔŰãĂĆ

äÿŁéİćæĹŚäžňä¡£çŤĺåĽřäžĘåijśåijŢçŤĺèőąæŢřïijŇåŕźäžŐåđČåIJ¿åŻđæŤűæİèèőšæŸŕå¿ĹæIJL'åÿőåŁl'çŽ
å¡ŞæĹŚäžňä£ĺæŇĄåőđä¿ŃçijŞå■ŸæŮűïijŇä¡ääŔŕèČ¡åŔłæČşåIJÍçÍŃåžŔäÿ■ä¡£çŤĺåĽřåőČäžňæŮűæL'■ä£Ĺå■
äÿĂäÿł WeakValueDictionary åőđä¿ŃåŔłäijŽä£İå■ŸéĆčäžŽåIJĺåĚűåőČåIJřæŰźè£ŸåIJÍèćńä¡£çŤÍçŽĎå
åŘęåĹŹçŽĎèŕİïijŇåŔłèęĄåőđä¿Ńäÿ■åĘ■èćńä¡£çŤĺäžĘïijŇåőČåřśäżŐå■ŮåĚÿäÿ■èćńçğżéŹd'äžĘãĂĆèğĆåŕ§ä

```
>>> a = get_spam('foo')
>>> b = get_spam('bar')
>>> c = get_spam('foo')
>>> list(_spam_cache)
['foo', 'bar']
>>> del a
>>> del c
>>> list(_spam_cache)
['bar']
>>> del b
>>> list(_spam_cache)
[]
>>>
```

åŕźäžŐåd'ğéČĺåĹEçÍŃåžŔèĂŇåůšïijŇè£ŹéĞŇäżčçăĄåůšçżŔåd'§çŤĺäžĘãĂĆäÿ■è£Ğè£ŸæŸŕæIJL'äÿĂäž

éęŰåĚĹæŸŕè£ŹéĞŇä¡£çŤĺåĽřäžĘäÿĂäÿłåĚĺåśĂåŔŸéĞŔïijŇåźűäÿŤåůěåŐĆåĞ¡æŢřèů§çśżæŤ¿åIJĺäÿĂ

```
import weakref

class CachedSpamManager:
    def __init__(self):
        self._cache = weakref.WeakValueDictionary()

    def get_spam(self, name):
        if name not in self._cache:
            s = Spam(name)
            self._cache[name] = s
        else:
            s = self._cache[name]
        return s
```

```
    def clear(self):
            self._cache.clear()

class Spam:
    manager = CachedSpamManager()
    def __init__(self, name):
        self.name = name

    def get_spam(name):
        return Spam.manager.get_spam(name)
```

è£Źæăůç ŽĎèŕÍäžčçăĄæŻť æÿĚæŹřïijŇåźűäÿŤäź§æŻť çĄţæť żïijŇæĹŚäžňåŕŕäžěåćđåŁăæŻť åď ŽçŽĎçij

è£ŸæIJL'äÿĂçĆźåřśæŸŕřïijŇæĹŚäžňæŻť éIJšäžĘçśžçŽĎåőđä¿ŃåŇŰçżŹçŤÍæĹůïijŇçŤÍæĹůå¿Ĺåőźæ ŸŞ

```
>>> a = Spam('foo')
>>> b = Spam('foo')
>>> a is b
False
>>>
```

æIJL'åĞăçğ■æŰźåijŔåŕŕäžěéŸšæ■ćçŤÍæĹůè£Źæăůå ĄŽïijŇçňňäÿĂäÿłæŸŕårĘçśžçŽĎåŘ■å■Ůä£őæŤźäÿ çňňäžŇçğ■åřśæŸŕèőľè£ŹäÿłçśžçŽĎ `__init__()` æŰźæşŢæŁŻåĞžäÿĂäÿłåijĆåÿÿïijŇèőľåőČäÿ■èČ¡èćńåĹ

```
class Spam:
    def __init__(self, *args, **kwargs):
        raise RuntimeError("Can't instantiate directly")

    # Alternate constructor
    @classmethod
    def _new(cls, name):
        self = cls.__new__(cls)
        self.name = name
```

çĎűåŘŐä£őæŤźçijŞå■ŸçőąçŘĘåŹĺäžčçăĄïijŇä¡£çŤÍ `Spam._new()` æĺěåĹŻåžžåőđä¿ŃïijŇèĂŇäÿ■æŸŕçŻť æŐěěřČçŤÍ `Spam()` æđĎéĂäåĞ¡æŢřïijŽ

```
# ----------------------æIJĂåŘŐçŽĎä£őæ■čæŰźæąĹ---------------------
 ↪---
class CachedSpamManager2:
    def __init__(self):
        self._cache = weakref.WeakValueDictionary()

    def get_spam(self, name):
        if name not in self._cache:
            temp = Spam3._new(name)  # Modified creation
            self._cache[name] = temp
        else:
            temp = self._cache[name]
        return temp
```

```
    def clear(self):
            self._cache.clear()

class Spam3:
    def __init__(self, *args, **kwargs):
        raise RuntimeError("Can't instantiate directly")

    # Alternate constructor
    @classmethod
    def _new(cls, name):
        self = cls.__new__(cls)
        self.name = name
        return self
```

æIJĂåŘŐè£Źæăůç ŽĎæŰźæąĹåřšåůšçżŕèűşåď§åě¡äžĘãĂĆ
çijŞå■ŸåŠŇåĚűäżŰæđĎéĂäæĺąåijŕè£Ÿåŕřäžěä¡£çŤĺ9.13åřŕèŁĆäÿ■çŽĎåĚČçśżåőđçŐřçŽĎæŻť äijŸéŻĚäÿ

CHAPTER 11

çňňäźİçńäïijŽåĚČçijŰçĺŃ

è¡ŕäżűåijĂåŔŚéćĘå§§äÿ■æIJĂçżŔåĚÿçŽĎåŔčåď'ťçęĚåřśæŸŕâĂIJdonâĂŹt repeat yourselfâĂİãĂĆ äź§åřśæŸŕèŕť'ïijŇäżżä¡ŢæŮűåĂŹå¡Şä¡ăçŽĎçĺŃåžŔäÿ■å■ŸåIJĺénŸåžęéĞ■åď■(æĹŰèĂĚæŸŕéĂåIJĺPythonå¡Şäÿ■ïijŇéĂŽåÿÿéČ¡åŔŕäżěéĂŽè£ĞåĚČçijŰçĺŃæİěèğčåĘşè£ŹçśżéŮőécŸãĂĆçőĂèĂŇèĺĂäźŃïijŇåĚČçijŰçĺŃåřśæŸŕåĚşäžŐåĹŻåżžæŞ■ä¡IJæžŘäżčçăĄ(æŕŤåęĆä£őæŤźãĂĄçŢ§æĹŘæĹŰäÿżèęĄæŁĂæIJŕæŸŕä¡£çŤĺèčĚéěřåŹĺãĂĄçśżèčĚéěřåŹĺåŠŇåĚČçśżãĂĆäÿ■è£Ğè£ŸæIJL'äÿĂäžŻåĚűäżŰæŁĂåŇĚæŃñç■¿åŘ■åŕźèśąãĂĄä¡£çŤĺ `exec()` æL'ğèąŇäżčçăĄäżěåŔŁåŕźåĘĚéČĺåĞ¡æŢřåŠŇçśżçŽĎåŔ■åřĎæŁĂæIJńçńäçŽĎäÿżèęĄçŽőçŽĎæŸŕåŘŚåď'ğåőűäżŃçż■è£ŹäžŻåĚČçijŰçĺŃæŁĂæIJŕïijŇåźűäÿŤçżŹåĞžåőđä¿Ńæ

Contents:

9.1 åIJĺåĞ¡æŢřäÿŁæůżåŁäåŇĚèčĚåŹĺ

éŮőécŸ

ä¡ăæČşåIJĺåĞ¡æŢřäÿŁæůżåŁääÿĂäÿłåŇĚèčĚåŹĺïijŇåćđåŁäécİåď'ŰçŽĎæŞ■ä¡IJåď'ĎçŘĘ(æŕŤåęĆæŮěå‡

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşä¡£çŤĺécİåď'ŰçŽĎäżčçăĄåŇĚèčĚäÿĂäÿłåĞ¡æŢřïijŇåŔŕäżěåőŽäźL'äÿĂäÿłèčĚéěřåŹĺåĞ

```
import time
from functools import wraps

def timethis(func):
    '''
    Decorator that reports the execution time.
```

```
    '''
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.time()
        result = func(*args, **kwargs)
        end = time.time()
        print(func.__name__, end-start)
        return result
    return wrapper
```

äÿŃéÍćæŸŕä¡£çŤÍèčĚéěřåŹÍçŽĎä¿Ńå■ŘïijŽ

```
>>> @timethis
... def countdown(n):
...     '''
...     Counts down
...     '''
...     while n > 0:
...         n -= 1
...
>>> countdown(100000)
countdown 0.008917808532714844
>>> countdown(10000000)
countdown 0.87188299392912
>>>
```

èőÍèőž

äÿĂäÿłèčĚéěřåŹÍåřśæŸŕäÿĂäÿłåĞ¡æŢřïijŇåőČæŐěåŔŮäÿĂäÿłåĞ¡æŢřä¡IJäÿžåŔĆæŢřåźűè£ŤåŻđäÿĂäÿå¡Şä¡ăåČŔäÿŃéÍćè£ŹæăůåĘŹïijŽ

```
@timethis
def countdown(n):
    pass
```

èů§åČŔäÿŃéÍćè£ŹæăůåĘŹåĚűåőđæŢĹæđIJæŸŕäÿĂæăůçŽĎïijŽ

```
def countdown(n):
    pass
countdown = timethis(countdown)
```

éąžä¿£èŕť'äÿĂäÿŃïijŇåĘĚç¡őçŽĎèčĚéěřåŹÍæŕŤåęĆ `@staticmethod`, `@classmethod`,`@property` åŐ§çŘĘäź§æŸŕäÿĂæăůçŽĎãĂĆä¿ŃåęĆïijŇäÿŃéÍćè£ŹäÿđäÿłäžčçăĄçL'ĞæőţæŸŕç■L'äžůçŽĎïijŽ

```
class A:
    @classmethod
    def method(cls):
        pass
```

```
class B:
    # Equivalent definition of a class method
    def method(cls):
        pass
    method = classmethod(method)
```

åIJĺäÿŁéİćçŽĎ `wrapper()` åĞ¡æŢřäÿ■ïijŇ èčĚéěřåŹĺåĘĚéČĺåőŽäźĽäžĘäÿĂäÿłä¡£çŢĺ `*args` åŠŇ `**kwargs` æİěæŐěåŔŮäżżæĎŔåŔĆæŢřçŽĎåĞ¡æŢřãĂĆ åIJĺè£ŹäÿłåĞ¡æŢřéĞŇéİćèřČçŢĺäžĘąŐ§ågŃåĞ¡æŢřåźűårĘąĚűçżŞædIJè£ŤåŻđïijŇäÿ■è£Ğä¡ăè£ŸåŔřäžěæůżçĎűåŘŐè£ŹäÿłæŰřçŽĎåĞ¡æŢřåŇĚèčĚåŹĺèćńä¡IJäÿžçżŞædIJè£ŤåŻđæİěäžčæŻ£åŐ§ågŃåĞ¡æŢřãĂĆ

éIJĂèęĄåijžèřČçŽĎæŸŕèčĚéěřåŹĺåźűäÿ■äijŽä£őæŤźåŐ§ågŃåĞ¡æŢřçŽĎåŔĆæŢřç■¿åŘ■äžěåŔŁè£ŤåŻ ä¡£çŢĺ `*args` åŠŇ `**kwargs` çŻőçŽĎåřśæŸŕçąőä£İäżżä¡ŢåŔĆæŢřéČ¡èČ¡éĂĆçŢĺãĂĆ èĂŇè£ŤåŻđçżŞædIJåĂijå§žæIJňéČ¡æŸŕèřČçŢĺåŐ§ågŃåĞ¡æŢř `func(*args, **kwargs)` çŽĎè£ŤåŻđçżŞædIJïijŇåĚűäÿ■funcåřśæŸŕåŐ§ågŃåĞ¡æŢřãĂĆ

åĹŽåijĂågŃå■ęäźäèčĚéěřåŹĺçŽĎæŮűåĂŹïijŇäijŽä¡£çŢĺäÿĂäžŽçőĂå■ŢçŽĎä¿Ńå■Řæİèèŕť æŸŐïijŇæŕ äÿ■è£Gåőđé ŹĚåIJžæŹŕä¡£çŢĺæŮűïijŇè£ŸæŸŕæIJL'äÿĂäžŽçżĘèŁĆéŮőécŸèęĄæşĺæĎŔçŽĎãĂĆ æŕŤåęĆäÿŁéİćä¡£çŢĺ `@wraps(func)` æşĺèğčæŸŕå¿ĹéĞ■èęĄçŽĎïijŇ åőČèČ¡ä£İçŢŹåŐ§ågŃåĞ¡æŢřçŽĎåĚČæŢřæ■ő(äÿŇäÿĂåřŔèŁĆäijŽèőšåĹř)ïijŇæŰřæL'ŃçżŘåÿÿäijŽå£¡çŢěè æŐěäÿŇæİěçŽĎåĞääÿłåřŔèŁĆæĹŚäžňäijŽæŻť åŁăæůśåĚěçŽĎèőšèğčèčĚéěřåŹĺåĞ¡æŢřçŽĎçżĘèŁĆéŮőécŸ

9.2 åĹŽåżžèčĚéěřåŹĺæŮűä£İçŢŹåĞ¡æŢřåĚČä£ąæĄŕ

éŮőécŸ

ä¡ăåĘŹäžĘäÿĂäÿłècĚéěřåŹĺä¡IJçŤĺåIJĺæ§ŘäÿłåĞ¡æŢřäŁïijŇä¡ĘæŸŕè£ŹäÿłåĞ¡æŢřçŽĎéĞ■èęĄçŽĎåĚ

èğčåĘşæŰźæąĹ

äżżä¡ŢæŮűåĂŹä¡ăåőŽäźĽèčĚéěřåŹĺçŽĎæŮűåĂŹïijŇéČ¡åžŤèŕěä¡£çŢĺ `functools` åžŞäÿ■çŽĎ `@wraps` èčĚéěřåŹĺæİěæşĺèğčåžŢåśĆåŇĚèčĚåĞ¡æŢřãĂĆä¿ŃåęĉïijŽ

```
import time
from functools import wraps
def timethis(func):
    '''
    Decorator that reports the execution time.
    '''
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.time()
        result = func(*args, **kwargs)
        end = time.time()
        print(func.__name__, end-start)
```

```
        return result
    return wrapper
```

äÿŃéİćæĹŚäžňä¡£çŤĺè£ŹäÿłèćńåŇĚèčĚåŘŐçŽĎåĞ¡æŢřåźűæčĂæ§ěåőČçŽĎåĚČä£ąæĄŕïijŽ

```
>>> @timethis
... def countdown(n:int):
...     '''
...     Counts down
...     '''
...     while n > 0:
...         n -= 1
...
>>> countdown(100000)
countdown 0.008917808532714844
>>> countdown.__name__
'countdown'
>>> countdown.__doc__
'\n\tCounts down\n\t'
>>> countdown.__annotations__
{'n': <class 'int'>}
>>>
```

èőĺèőž

åIJĺçijŰåĘŹèčĚéěřåŹĺçŽĎæŮűåĂŹåď■åĹűåĚČä£ąæĄŕæŸŕäÿĂäÿłéİđåÿÿéĞ■èęĄçŽĎéČĺåĹĘãĂĆåęĆæ
@wraps ïijŇ éĆčäźĹä¡ääijŽåŘŚçŐřèćńèčĚéěřåĞ¡æŢřäÿćåď śäžĘæĽ ĂæIJL'æIJL'çŤĺçŽĎä£ąæĄŕãĂĆæŕŤåęĆ
@wraps åŘŐçŽĎæŢĹæđIJæŸŕäÿŃéİćè£ŹæăůçŽĎïijŽ

```
>>> countdown.__name__
'wrapper'
>>> countdown.__doc__
>>> countdown.__annotations__
{}
>>>
```

@wraps æIJL'äÿĂäÿłéĞ■èęĄçĽźå¿ĄæŸŕåőČèČ¡èőľä¡äéĂŽè£ĞåšđæĂğ
__wrapped__ çŻť'æŐèèő£éŮőèćńåŇĚèčĚåĞ¡æŢřãĂĆä¿ŃåęĆ:

```
>>> countdown.__wrapped__(100000)
>>>
```

__wrapped__ åšđæĂğè£ŸèČ¡èőľèćńèčĚéěřåĞ¡æŢřæ■čçąőæŽť'éIJšåžŢåšĆçŽĎåŔĆæŢřç■¿åŘ■ä£ąæA

```
>>> from inspect import signature
>>> print(signature(countdown))
(n:int)
>>>
```

äÿĂäÿłå¿ĹæŹőéĄ■çŽĎéŮőécŸæŸŕæĂŐæăůèőľ'èčĚéěřåŹĺåŐżçŻť'æŐěåď'■åĹűåŐ§ågŃåĞ¡æŢřçŽĎåŔĆ
åęĆædIJæČşèĞłåůśæĽ'ŃåŁĺåőđçŐřçŽĎèŕÍéIJĂèęĄåĄŽåď'ğéĞŔçŽĎåůěä¡IJïijŇæIJĂåě¡åřśçőĂå■ŢçŽĎä¡£çŤ
__wrapped__ èčĚéěřåŹĺãĂĆ éĂŽè£ĞåžŢåśĆçŽĎ __wrapped__
åśđæĂğèő£éŮőåĹřåĞ¡æŢřç■¿åŘ■ä£ąæĄŕãĂĆæŻť'åď'ŽåĚşäžŐç■¿åŘ■çŽĎåĘĚåőźåŔŕäżěåŔĆèĂČ9.16åřŔèŁ

9.3 èğčéŹď'äÿĂäÿłèčĚéěřåŹĺ

éŮőécŸ

äÿĂäÿłèčĚéěřåŹĺåůšçżŔä¡IJçŤĺåIJĺäÿĂäÿłåĞ¡æŢřäŸŁïijŇä¡ăæČşæŠď'éŤĂåőČïijŇçŻť'æŐèèő£éŮőåŐ§åğ

èğčåĘşæŰźæąĹ

åĄĞèő¿èčĚéěřåŹĺæŸŕéĂŽè£Ğ `@wraps` (åŔĆèĂČ9.2åřŔèŁĆ)æİěåőđçŐřçŽĎïijŇéĆčäźĹä¡ăåŔŕäżěéĂŽè
__wrapped__ åśđæĂğæİèèő£éŮőåŐ§ågŃåĞ¡æŢřïijŽ

```
>>> @somedecorator
>>> def add(x, y):
...     return x + y
...
>>> orig_add = add.__wrapped__
>>> orig_add(3, 4)
7
>>>
```

èőĺèőž

çŻť'æŐèèő£éŮőæIJłåŇĚèčĚçŽĎåŐ§ågŃåĞ¡æŢřåIJĺèřČèŕŢãĂĄåĘĚçIJĄåŠŇåĚűäżŮåĞ¡æŢřæŞ■ä¡IJæŮ
ä¡ĘæŸŕæĹŚäżňè£ŹéĞŇçŽĎæŰźæąĹäžĚäżĚéĂĆçŤĺäžŐåIJĺåŇĚèčĚåŹĺäÿ■æ■čçąőä¡£çŤĺäžĘ
`@wraps` æĹŰèĂĚçŻť'æŐèèő¿ç¡őäžĘ __wrapped__ åśđæĂğçŽĎæČĚåĘţãĂĆ

åęĆædIJæIJL'åď'Žäÿłå ŇĚèčĚåŹĺïijŇéĆčäźĹèőž£éŮő __wrapped__
åśđæĂğçŽĎèąŇäÿžæŸŕäÿ■åŕŕécĎç§ěçŽĎïijŇåžŤèŕěéĄ£åĚ■è£ŹæăůåĄŽãĂĆ
åIJĺPython3.3äÿ■ïijŇåőČäijŽçŢèè£ĞæĽ'ĂæIJL'çŽĎåŇĚèčĚåśĆïijŇærŤåęĆïijŇåĄĞåęĆä¡ăæIJL'åęĆäÿŇçŽĎ

```
from functools import wraps

def decorator1(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        print('Decorator 1')
        return func(*args, **kwargs)
    return wrapper

def decorator2(func):
```

```
    @wraps(func)
    def wrapper(*args, **kwargs):
        print('Decorator 2')
        return func(*args, **kwargs)
    return wrapper

@decorator1
@decorator2
def add(x, y):
    return x + y
```

äÿŃéİćæĹŚäżňåIJÍPython3.3äÿŃæţŃèŕŢïijŽ

```
>>> add(2, 3)
Decorator 1
Decorator 2
5
>>> add.__wrapped__(2, 3)
5
>>>
```

äÿŃéİćæĹŚäżňåIJÍPython3.4äÿŃæţŃèŕŢïijŽ

```
>>> add(2, 3)
Decorator 1
Decorator 2
5
>>> add.__wrapped__(2, 3)
Decorator 2
5
>>>
```

æIJĂåŘŐèęĄèŕť'çŽĎæŸŕïijŇåźűäÿ■æŸŕæĽĂæIJL'çŽĎèčĚéěřåŹĺéČ¡ä¡£çŤĺäžĘ `@wraps` ïijŇåŻäæ■d'è£ŹéĞŇçŽĎæŰźæąĹåźűäÿ■åĚĺéČĺéĂĆçŤĺãĂĆçĽ'źåĹńçŽĎïijŇåĘĚç¡őçŽĎèčĚéěřåŹĺ `@staticmethod` åŠŇ `@classmethod` åřśæšąæIJL'éĄţå¿łè£Źäÿłçžęåőž (åőČäżňæŁŁåŐ§åğŃåĞ¡æŢřå■ŸåĆĺåIJĺåśđæĂğ `__func__` äÿ■)ãĂĆ

9.4 åőŽäźĽäÿĂäÿłåÿęåŔĆæŢřçŽĎèčĚéěřåŹĺ

éŮőécŸ

ä¡ăæČşåőŽäźĽäÿĂäÿłåŔŕäžěæŐěåŘŮåŔĆæŢřçŽĎèčĚéěřåŹĺ

èğčåĘşæŰźæąĹ

æĹŚäžňçŤĺäÿĂäÿłä¿Ńå■Řèŕęçżĕ́ÝŘè£řäÿŃæŐěåŔŮåŔĆæŢřçŽĎåď˙ĎçŘĘè£ĞçÍŃãĂĆ åĄĞèő¿ä¡ăæČşåĔŹäÿĂäÿłèčĚéěřåŹÍïijŇçżŹåĞ¡æŢřæůžåŁăæŮěå£ŮåŁ§èČ¡ïijŇå¡ŞæŮűåĚĄèőÿçŤĺæĹůæŇĞ äÿŃéIćæÝŕè£ŹäÿłèčĚéěřåŹÍçŽĎåőŽäźĽ˙åŠŇä¡£çŤĺçď˙žä¿ŃïijŽ

```
from functools import wraps
import logging

def logged(level, name=None, message=None):
    """
    Add logging to a function. level is the logging
    level, name is the logger name, and message is the
    log message. If name and message aren't specified,
    they default to the function's module and name.
    """
    def decorate(func):
        logname = name if name else func.__module__
        log = logging.getLogger(logname)
        logmsg = message if message else func.__name__

        @wraps(func)
        def wrapper(*args, **kwargs):
            log.log(level, logmsg)
            return func(*args, **kwargs)
        return wrapper
    return decorate

# Example use
@logged(logging.DEBUG)
def add(x, y):
    return x + y

@logged(logging.CRITICAL, 'example')
def spam():
    print('Spam!')
```

åĹİçIJŃèţůæİěïijŇè£Źçğ■åőđçŐřçIJŃäÿŁåŐżå¿Ĺåď˙■æİĆïijŇä¡ĘæŸŕæăÿå£ČæĂİæČşå¿ĹçőĂå■ŢãĂĆ æIJĂåď˙ŰåśĆçŽĎåĞ¡æŢř `logged()` æŐěåŔŮåŔĆæŢřåźűåřĘåőČäžňä¡IJçŤĺåIJĺåĘĚéČĺçŽĎèčĚéěřåŹĺåĞ¡æŢ åĘĚåśĆçŽĎåĞ¡æŢř `decorate()` æŐěåŔŮäÿĂäÿłåĞ¡æŢřä¡IJäÿžåŔĆæŢřïijŇçĎűåŘŐåIJĺåĞ¡æŢřäŸŁéIćæŢ è£ŹéĞŇçŽĎåĔşéŤőçĆźæŸŕåŇĚèčĚåŹĺæŸŕåŕřäžěä¡£çŤĺäijäéĂŠçżŹ `logged()` çŽĎåŔĆæŢřçŽĎãĂĆ

èőĺèőž

åőŽäźĽäÿĂäÿłæŐěåŔŮåŔĆæŢřçŽĎåŇĚèčĚåŹĺçIJŃäÿŁåŐżæŕŤè¿Čåď˙■æİĆäÿžèęĄæŸŕåŽääÿžåžŢåśĆç

```
@decorator(x, y, z)
def func(a, b):
```

```
    pass
```

èčĚéěřåŹĺåď ĎçŘĘè£ĞçÍŃèů§äÿŃéİćçŽĎèřČçŤÍæŸŕç■Ľ'æŢĹçŽĎ;

```
def func(a, b):
    pass
func = decorator(x, y, z)(func)
```

decorator(x, y, z) çŽĎè£ŤåŻđçżŞæđIJå£ĚéążæŸŕäÿĂäÿłåŔŕèřČçŤÍĺŕźèśąïijŇåőČæŐěåŔŮäÿĂ åŔŕäžěåŔĆèĂČ9.7åřŔèŁĆäÿ■åŔęåď ŰäÿĂäÿłåŔŕæŐěåŔŮåŔĆæŢřçŽĎåŇĚèčĚåŹĺä¿Ńå■ŘãĂĆ

9.5 åŔŕèĞłåőŽäźĽåśđæĂğçŽĎèčĚéěřåŹÍ

éŮőécŸ

ä¡ăæČşåĘŹäÿĂäÿłèčĚéěřåŹÍæİěåŇĚèčĚäÿĂäÿłåĞ¡æŢřïijŇåźűäÿŤåĚĄèőÿçŤÍæĹůæŔŘä¿ŻåŔĆæŢřåIJÍè

èğčåĘşæŰźæąĹ

åijŢåĚěäÿĂäÿłèő£éŮőåĞ¡æŢřïijŇä¡£çŤÍ nolocal æİěä£őæŤźåĘĚéČĺåŔŸéĞŔãĂĆ çĎűåŘŐè£Źäÿłèő£éŮőåĞ¡æŢřèćńä¡IJäÿžäÿĂäÿłåśđæĂğètŃåĂijçżŹåŇĚèčĚåĞ¡æŢřãĂĆ

```
from functools import wraps, partial
import logging
# Utility decorator to attach a function as an attribute of obj
def attach_wrapper(obj, func=None):
    if func is None:
        return partial(attach_wrapper, obj)
    setattr(obj, func.__name__, func)
    return func

def logged(level, name=None, message=None):
    '''
    Add logging to a function. level is the logging
    level, name is the logger name, and message is the
    log message. If name and message aren't specified,
    they default to the function's module and name.
    '''
    def decorate(func):
        logname = name if name else func.__module__
        log = logging.getLogger(logname)
        logmsg = message if message else func.__name__

        @wraps(func)
        def wrapper(*args, **kwargs):
            log.log(level, logmsg)
```

```
            return func(*args, **kwargs)

        # Attach setter functions
        @attach_wrapper(wrapper)
        def set_level(newlevel):
            nonlocal level
            level = newlevel

        @attach_wrapper(wrapper)
        def set_message(newmsg):
            nonlocal logmsg
            logmsg = newmsg

        return wrapper

    return decorate

# Example use
@logged(logging.DEBUG)
def add(x, y):
    return x + y

@logged(logging.CRITICAL, 'example')
def spam():
    print('Spam!')
```

äÿŃéİćæŸŕäžd'äžŠçŐŕåćČäÿŃçŽĎä¡£çŤĺä¿Ńå■ŘïijŽ

```
>>> import logging
>>> logging.basicConfig(level=logging.DEBUG)
>>> add(2, 3)
DEBUG:__main__:add
5
>>> # Change the log message
>>> add.set_message('Add called')
>>> add(2, 3)
DEBUG:__main__:Add called
5
>>> # Change the log level
>>> add.set_level(logging.WARNING)
>>> add(2, 3)
WARNING:__main__:Add called
5
>>>
```

èőÍèőž

è£ŹäÿĂåřŔèŁĆçŽĎåĚşéŤőçĆźåIJÍäžŐèő£éŮőåĞ¡æŢř(åęĆ `set_message()` åŠŇ `set_level()`)ïijŇåőČäžňèćńä¡IJäÿžåśđæĂğèţŃçżŹåŇĚèčĚåŹÍãĂĆ

æŕŕäÿłèő£éŮőåĞ¡æŢřåĚĄèőÿä¡£çŤĺ nonlocal æĺěä£őæŤźåĞ¡æŢřåĘĚéČĺçŽĎåŔŸéĞŔãĂĆ

è£ŸæIJL'äÿĂäÿłäžd'äžžåŘČæČŁçŽĎåIJřæŰźæŸŕèő£éŮőåĞ¡æŢřäijŽåIJĺåd'ŽåśĆèčĚéěřåŹĺéŮt'äijäæŠ■
@functools.wraps æşĺèğč)ãĂĆ ä¿ŃåęĆïijŇåĄĞèő¿ä¡ăåijŢåĚěåŔęåd'ŰäÿĂäÿłèčĚéěřåŹĺïijŇæŕŤåęĆ9.2å
@timethis ïijŇåČŔäÿŃéIćè£Źæăůïijž

```
@timethis
@logged(logging.DEBUG)
def countdown(n):
    while n > 0:
        n -= 1
```

ä¡ăäijŽåŔŚçŐřèő£éŮőåĞ¡æŢřä¿ĺæŮğæIJL'æŢĹïijž

```
>>> countdown(10000000)
DEBUG:__main__:countdown
countdown 0.8198461532592773
>>> countdown.set_level(logging.WARNING)
>>> countdown.set_message("Counting down to zero")
>>> countdown(10000000)
WARNING:__main__:Counting down to zero
countdown 0.8225970268249512
>>>
```

ä¡ăè£ŸäijŽåŔŚçŐřå■şä¡£èčĚéěřåŹĺåČŔäÿŃéIćè£Źæăůäžěçżÿå Ŕ■çŽĎæŰźåŘŚæŐŠæŤ¿ïijŇæŢĹæđIJäź

```
@logged(logging.DEBUG)
@timethis
def countdown(n):
    while n > 0:
        n -= 1
```

è£ŸèČ¡éĂŽè£Ğä¡£çŤĺlambdaèąĺè¿¿åijŔäžčçăĄæĺěől'èő£éŮőåĞ¡æŢřçŽĎè£ŤåŻđäÿ■åŘŇçŽĎèő¿åőŽåA

```
@attach_wrapper(wrapper)
def get_level():
    return level

# Alternative
wrapper.get_level = lambda: level
```

äÿĂäÿłæŕŤè¿ČéŽ¿çŘEèğčçŽĎåIJřæŰźåřśæŸŕåŕźäžŐèő£éŮőåĞ¡æŢřçŽĎéęŰæňąä¡£çŤĺãĂĆä¿ŃåęĆïijŇ

```
@wraps(func)
def wrapper(*args, **kwargs):
    wrapper.log.log(wrapper.level, wrapper.logmsg)
    return func(*args, **kwargs)

# Attach adjustable attributes
wrapper.level = level
wrapper.logmsg = logmsg
wrapper.log = log
```

è£ŹäÿłæŰźæşŢäź§åŔŕèČ¡æ■čåÿÿåůëä¡IJïijŇä¡ĘåĽ■æŔŘæŸŕåőČå£Ěéążæ ŸŕæIJĂåď ŰåśÇçŽĎèčĚéěřåŹ
åęĆæđIJåőČçŽĎäÿŁéİćè£ŸæIJĽåĶęåď ŰçŽĎèčĚéěřåŹĺ(æŕŤåęĆäÿŁéİćæŔŘåĹřçŽĎ
`@timethis` ä¿Ńå■Ř)ïijŇéĆčäźĹåőČäijŽéŽŘèŮŔåžŢåśĆåśđæĂğïijŇä¡£å¿Ůä£őæŤźåőČäžñæšąæIJĽäżżä¡Ţ
èĂŇéĂŽè£Ğä¡£çŤĺèő£éŮőåĞ¡æŢřåŕśèČ¡éĄ£åĚ■è£ŹæäůçŽĎåśĂéŹŘæĂğãĂĆ

æIJĂåŘŐæŔŘäÿĂçĆźïijŇè£ŹäÿĂåřŔèŁĆçŽĎæŰźæąĹäź§åŔŕäżěä¡IJäÿž9.9åřŔèŁĆäÿ■èčĚéěřåŹĺçśżçŽĺ

9.6 åÿęåŘŕéĂĽåŔĆæŢřçŽĎèčĚéěřåŹĺ

éŮőécŸ

ä¡ăæČşåĘŹäÿĂäÿłèčĚéěřåŹĺïijŇæŮćåŔŕäżěäÿ■äijăåŔĆæŢřçżŹåőČïijŇæŕŤåęĆ
`@decorator` ïijŇ äź§åŔŕäżěäijăéĂŠåŔŕéĂĽåŔĆæŢřçżŹåőČïijŇæŕŤåęĆ
`@decorator(x,y,z)` ãĂĆ

èğčåĘşæŰźæąĹ

äÿŃéİćæŸŕ9.5åřŔèŁĆäÿ■æŮěå£ŮèčĚéěřåŹĺçŽĎäÿĂäÿłä£őæŤźçĽĹæIJñïijŽ

```
from functools import wraps, partial
import logging

def logged(func=None, *, level=logging.DEBUG, name=None,␣
↪message=None):
    if func is None:
        return partial(logged, level=level, name=name,␣
↪message=message)

    logname = name if name else func.__module__
    log = logging.getLogger(logname)
    logmsg = message if message else func.__name__

    @wraps(func)
    def wrapper(*args, **kwargs):
        log.log(level, logmsg)
        return func(*args, **kwargs)

    return wrapper

# Example use
@logged
def add(x, y):
    return x + y

@logged(level=logging.CRITICAL, name='example')
def spam():
    print('Spam!')
```

åŘŕäžěçIJŃåĹřïijŇ@logged èčĚéěřåŹĺåŘŕäžěåŘŇæŮűäÿ■åÿęåŘĆæŢřæĹŰåÿęåŘĆæŢřãĂĆ

èőĺèőž

è£ŹéĞŇæŘŘåĹřçŽĎè£ŹäÿłéŮőécŸåřśæŸŕéĂŽåÿÿæĽĂèŕť çŽĎçijŰçĺŃäÿĂèĞť æĂğéŮőécŸãĂĆ
å¡ŞæĹŚäžňä¡£çŤĺèčĚéěřåŹĺçŽĎæŮűåĂŹïijŇåď ğéČĺåĹĘçĺŃåžŘåŚŸäźăæČŕäžĘèęĄäźĹäÿ■çżŹåőČäžňäijäéĂ
åĚűåődäżŐæŁĂæIJŕäÿŁæİěèőšïijŇæĹŚäžňåŘŕäžěåőŽäźĽäÿĂäÿłæĽĂæIJĽåŘĆæŢřéČ¡æŸŕåŘŕéĂĽçŽĎèčĚ

```
@logged()
def add(x, y):
    return x+y
```

ä¡ĘæŸŕïijŇè£Źçğ■åĘŹæşŢåźűäÿ■çňęåŘĹæĹŚäžňçŽĎäźăæČŕïijŇæIJĽæŮűåĂŹçĺŃåžŘåŚŸå£Ÿèőŕåŀăä
è£ŹéĞŇæĹŚäžňåŘŚä¡ăåśŢçd'žäžĘåęĆä¡Ţäžěäÿ ĂèĞť çŽĎçijŰçĺŃéčŐæäijæİěåŘŇæŮűæżąèűşæšąæIJĽæŃňă

äÿžäžĘçŘĘèğčäžčçăĄæŸŕåęĆä¡Ţåůěä¡IJçŽĎïijŇä¡ăéIJĂèęĄéİđåÿÿçĘ§æĆĽèčĚéěřåŹĺæŸŕåęĆä¡Ţä¡IJçŽ
åŕźäžŐäÿĂäÿłåČŔäÿŃéİćè£ŹæăůçŽĎçőĂå■ŢèčĚéěřåŹĺïijŽ

```
# Example use
@logged
def add(x, y):
    return x + y
```

è£Źäÿłèřčç ŤĺåžŘåĹŮèů§äÿŃéİćç■Ľäžůïij Ž

```
def add(x, y):
    return x + y

add = logged(add)
```

è£ŹæŮűåĂŹïijŇèćńèčĚéěřåĞ¡æŢřäijŽèćńå¡ŞåĄŽçňňäÿĂäÿłåŘĆæŢřçŻť æŐëäijăéĂŠçżŹ
logged èčĚéěřåŹĺãĂĆ åŽăæ■ď ïijŇlogged() äÿ■çŽĎçňňäÿĂäÿłåŘĆæŢřåŕśæŸŕèćńåŇĚèčĚåĞ¡æŢřæIJňè

èĂŇåŕźäžŐäÿĂäÿłäÿŃéİćè£ŹæăůæIJĽåŘĆæŢřçŽĎèčĚéěřåŹĺïijŽ

```
@logged(level=logging.CRITICAL, name='example')
def spam():
    print('Spam!')
```

èřČçŤĺåžŘåĹŮèů§äÿŃéİćç■Ľäžůïij Ž

```
def spam():
    print('Spam!')
spam = logged(level=logging.CRITICAL, name='example')(spam)
```

åĹİåğŃèřČçŤĺ logged() åĞ¡æŢřæŮűïijŇèćńåŇĚèčĚåĞ¡æŢřåźűæšąæIJĽäijăéĂŠè£ŽæİěãĂĆ
åŽăæ■ď åIJĺèčĚéěřåŹĺåĘĚïijŇåőČå£Ěéążæ ŸŕåŘŕéĂĽçŽĎãĂĆè£ŹäÿłåŘ■è£ĞæİěäijŽè£ńä¡£åĚűäžŮåŘĆæŢřå
åźűäÿŤïijŇä¡Ęè£ŹäžŽåŘĆæŢřèćńäijăéĂŠè£ŽæİěåŘŐïijŇèčĚéěřåŹĺèęĄè£ŤåŽđäÿĂäÿłæŐěåŘŮäÿĂäÿłåĞ¡æ
äÿžäžĘè£ŹæăůåĄŽïijŇæĹŚäžňä¡£çŤĺäžĘäÿĂäÿłæŁĂåůğïijŇåŕśæŸŕåĽçŤĺ functools.
partial ãĂĆ åőČäijŽè£ŤåŽđäÿĂäÿłæIJłåőŇåĚĺåĹİåğŇåŃŮçŽĎèĞłèžńïijŇéŹď äžĘèćńåŇĚèčĚåĞ¡æŢřåd'
åŘŕäžěåŘĆèĂČ7.8åŕŘèŁĆèŐůåŘŮæŻť åď Ž partial() æŰźæşŢçŽĎç§ěèŕĘãĂĆ

9.7 åĹľçŤĺèčĚéěřåŹĺåijžåĹűåĞ¡æŢřäÿŁçŽĎçśžåđŃæčĂæ§ě

éŮőécŸ

ä¡IJäÿžæ§ŘçğĺçijŰçÍŃèğĎçžęïijŇä¡ăæČşåIJĺářźåĞ¡æŢřåŔĆæŢřè£ŻèąŇåijžåĹűçśžåđŃæčĂæ§ěãĂĆ

èğčåĘşæŰźæąĹ

åIJĺæijŤçd'žåőđéŹĚäżčçăĄåL'ïijŇåĚĹèŕt'æŸŐæĹŚäžňçŽĎçŻőæăĞïijŽèČ¡ářźåĞ¡æŢřåŔĆæŢřçśžåđŃè

```
>>> @typeassert(int, int)
... def add(x, y):
...     return x + y
...
>>>
>>> add(2, 3)
5
>>> add(2, 'hello')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "contract.py", line 33, in wrapper
TypeError: Argument y must be <class 'int'>
>>>
```

äÿŃéİćæŸŕä¡£çŤĺèčĚéěřåŹĺæŁĂæIJŕæİěåőđçŐř `@typeassert` ïijŽ

```
from inspect import signature
from functools import wraps

def typeassert(*ty_args, **ty_kwargs):
    def decorate(func):
        # If in optimized mode, disable type checking
        if not __debug__:
            return func

        # Map function argument names to supplied types
        sig = signature(func)
        bound_types = sig.bind_partial(*ty_args, **ty_kwargs).
↪arguments

        @wraps(func)
        def wrapper(*args, **kwargs):
            bound_values = sig.bind(*args, **kwargs)
            # Enforce type assertions across supplied arguments
            for name, value in bound_values.arguments.items():
                if name in bound_types:
                    if not isinstance(value, bound_types[name]):
                        raise TypeError(
```

```
                                'Argument {} must be {}'.format(name,␣
↪bound_types[name])
                                )
            return func(*args, **kwargs)
        return wrapper
    return decorate
```

åŕŕäžěçIJŃåĞžè£ŹäÿłèčĕéěřåŹĺéİđåÿÿçĄţæť żïijŇæŮćåŕŕäžěæŇĞåőŽæĽĂæIJĽåŔĆæŢřçśżåđŃïijŇäź
åźűäÿŤåŕŕäžěéĂŽè£Ğä¡■ç¡őæĹŰåĚşéŤőå■ŮæİěæŇĞåőŽåŔĆæŢřçśżåđŃãĂĆäÿŃéİćæŸŕä¡£çŤĺçď žä¿ŃïijŽ

```
>>> @typeassert(int, z=int)
... def spam(x, y, z=42):
...     print(x, y, z)
...
>>> spam(1, 2, 3)
1 2 3
>>> spam(1, 'hello', 3)
1 hello 3
>>> spam(1, 'hello', 'world')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "contract.py", line 33, in wrapper
TypeError: Argument z must be <class 'int'>
>>>
```

èőĺèőž

è£ŹèŁĆæŸŕéńŸçžğèčĔéěřåŹĺçď žä¿ŃïijŇåijŢåĚěäžĘå¿Ĺåď ŽéĞ■èęĄçŽĎæęĆå£ţãĂĆ

éęŰåĚĹïijŇèčĔéěřåŹĺåŔłäijŽåIJĺåĞ¡æŢřåőŽäźĽæŮűèćńèřČçŤĺäÿĂæňąãĂĆ
æIJĽæŮűåĂŹä¡ăåŐżæŐĽèčĔéěřåŹĺçŽĎåŁ§èČ¡ïijŇéĆčäźĹä¡ăåŔłéIJĂèęĄçőĂå■ŢçŽĎè£ŤåŻđèćńèčĔéěřåĞ¡
äÿŃéİćçŽĎäžčçăĄäÿ■ïijŇåęĆæđIJåĚĺåśĂåŔŸéĞŘãĂĂ__debug__
èćńèő¿ç¡őæĹŘäžĘFalse(å¡Şä¡ää¡£çŤĺ-OæĹŰ-OOåŔĆæŢřçŽĎäijŸåŇŰæĺąåijŔæĽ'ğèąŇçĺŃåžŔæŮű)ïijŇ
éĆčäźĹåřśçŻť'æŐèè£ŤåŻđæIJłä£őæŤźè£ĞçŽĎåĞ¡æŢřæIJňèžńïijŽ

```
def decorate(func):
    # If in optimized mode, disable type checking
    if not __debug__:
        return func
```

åĚűæňąïijŇè£ŹéĞŇè£ŸåŕźèćńåŇĚèčĔåĞ¡æŢřçŽĎåŔĆæŢřç■¿åŘ■è£ŻèąŇäžĘæčĂæ§ěïijŇæĹŚäžňä¡£çŤ
inspect.signature() åĞ¡æŢřãĂĆ çőĂå■ŢæİěëőšïijŇåőČè£ŘèąŇä¡ăæŔŘåŔŰäÿĂäÿłåŔŕěŘçŤĺåŕźèśą

```
>>> from inspect import signature
>>> def spam(x, y, z=42):
...     pass
...
>>> sig = signature(spam)
>>> print(sig)
```

```
(x, y, z=42)
>>> sig.parameters
mappingproxy(OrderedDict([('x', <Parameter at 0x10077a050 'x'>),
('y', <Parameter at 0x10077a158 'y'>), ('z', <Parameter at␣
↪0x10077a1b0 'z'>)]))
>>> sig.parameters['z'].name
'z'
>>> sig.parameters['z'].default
42
>>> sig.parameters['z'].kind
<_ParameterKind: 'POSITIONAL_OR_KEYWORD'>
>>>
```

èčĚéěřåŹĺçŽĎåijĂåğŃéČĺåĹĘijŇæĹŚäžňä¡£çŤĺäžĘ `bind_partial()` æŰźæşŢæİěæĽğèąŇäżŐæŇĞåőŽçśžåđŃåĹřåŘ■çğřçŽĎéČĺåĹĘçżŚåőŽãĂĆ äÿŃéIćæŸŕä¿Ńå■ŘæijŤçď'žïijŽ

```
>>> bound_types = sig.bind_partial(int,z=int)
>>> bound_types
<inspect.BoundArguments object at 0x10069bb50>
>>> bound_types.arguments
OrderedDict([('x', <class 'int'>), ('z', <class 'int'>)])
>>>
```

åIJĺè£ŹäÿłéČĺåĹĘçżŚåőŽäÿ■ïijŇä¡ääŘŕäžěæşĺæĎŘåĹřçijžåď'śçŽĎåŔĆæŢřèćńå£¡çŢěäžĘ(æŕŤåęĆåźűæš äÿ■è£ĞæIJĂéĞ■èęĄçŽĎæŸŕåĹŻåžžäžĘäÿĂäÿłæIJL'ážŘå■ŮåĚÿ `bound_types.arguments` ãĂĆ è£Źäÿłå■ŮåĚÿäijŽåŕĘåŔĆæŢřåŘ■äžěåĞ¡æŢřç■¿åŘ■äÿ■çŽÿåŘŇéąžážŘæŸĂåřĎåĹřæŇĞå åIJĺæĹŚäžňçŽĎèčĚéěřåŹĺä¿Ńå■ŘäÿïijŇè£ŹäÿłæŸĂåřĎåŇĚåŘńäžĘæĹŚäžňèęĄåijžåĹűæŇĞåőŽçŽĎçśžåđŃ

åIJĺèčĚéěřåŹĺåĹŻåžžçŽĎåőđéŹĚåŇĚèčĚåĞ¡æŢřäÿ■ä¡£çŤĺåĹřäžĘ `sig.bind()` æŰźæşŢãĂĆ `bind()` èů§ `bind_partial()` çśžäijijïijŇä¡ĘæŸŕåőČäÿ■åĚĄèőÿå£¡çŢěäžžä¡ŢåŔĆæŢřãĂĆåŽäæ■ď'æIJL'ážĘäÿŃéIćçŽĎçżŞæđIJïijŽ

```
>>> bound_values = sig.bind(1, 2, 3)
>>> bound_values.arguments
OrderedDict([('x', 1), ('y', 2), ('z', 3)])
>>>
```

ä¡£çŤĺè£ŹäÿłæŸĂåřĎæĹŚäžňåŕŕäžěå¿Ĺè¡żæİ¿çŽĎåőđçŐřæĹŚäžňçŽĎåijžåĹűçśžåđŃæčĂæ§ěïijŽ

```
>>> for name, value in bound_values.arguments.items():
...     if name in bound_types.arguments:
...         if not isinstance(value, bound_types.arguments[name]):
...             raise TypeError()
...
>>>
```

äÿ■è£Ğè£ŹäÿłæŰźæąĹè£ŸæIJL'çĆźåřŔçŚŢçŰţijŇåőČåŕźäžŐæIJL'éżŸèőď'åĂijçŽĎåŔĆæŢřåźűäÿ■éĂĆ æŕŤåęĆäÿŃéIćçŽĎäžčçăĄåŔŕäžěæ■čåÿÿåůěä¡IJïijŇår¡çőąitemsçŽĎçśžåđŃæŸŕéŤŹèŕŕçŽĎïijŽ

```
>>> @typeassert(int, list)
... def bar(x, items=None):
...     if items is None:
...         items = []
...     items.append(x)
...     return items
>>> bar(2)
[2]
>>> bar(2,3)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "contract.py", line 33, in wrapper
TypeError: Argument items must be <class 'list'>
>>> bar(4, [1, 2, 3])
[1, 2, 3, 4]
>>>
```

æIJĂåŘŐäÿĂçĆźæŸŕåĚşäžŐéĂĆçŤĺèČéěřåŹĺåŔĆæŢřåŠŇåĞ¡æŢřæşĺèğčäźŃéŮťçŽĎäžL'èőžãĂĆ
ä¿ŃåęĆïijŇäÿžäżĂäźĹäÿ■åČŔäÿŃéİćè£ŹæăůåĘŹäÿĂäÿłèČĚéěřåŹĺæİěæ§ěæL'¿åĞ¡æŢřäÿ■çŽĎæşĺèğčåŚćïij§

```
@typeassert
def spam(x:int, y, z:int = 42):
    print(x,y,z)
```

äÿĂäÿłåŔŕèČ¡çŽĎåŐ§åŻăæŸŕåęĆæđIJä¡£çŤĺäžĘåĞ¡æŢřåŔĆæŢřæşĺèğčïijŃéĆčäźĹåřśèćńéŹŘåĹűäžĘãĂ
åęĆæđIJæşĺèğčèćńçŤĺæİěåĄŽçśżåđŃæčĂæ§ěåřśäÿ■èČ¡åĄŽåĚűäžŰäžŃæČĚäžĘãĂĆèĂŇäÿŤ
`@typeassert` äÿ■èČ¡åĘ■çŤĺäžŐä¡£çŤĺæşĺèğčåĄŽåĚűäžŰäžŃæČĚçŽĎåĞ¡æŢřäžĘãĂĆ
èĂŇä¡£çŤĺäÿŁéİćçŽĎèčĚéěřåŹĺåŔĆæŢřçĄţæť'żæĂğåď'ğåď'ŽäžĘïijŇäź§æŻť'åŁäéĂŽçŤĺãĂĆ

åŕřäżěåIJĺPEP 362äżěåŔŁ `inspect` æĺąåİŮäÿ■æL'¿åĹřæŻť'åď'ŽåĚşäžŐåĞ¡æŢřåŔĆæŢřářźèśąçŽĎä£ąą

9.8 åřĘèčĚéěřåŹĺåőŽäźL'äÿžçśżçŽĎäÿĂéČĺåĹĘ

éŮőécŸ

ä¡äæČşåIJĺçśżäÿ■åőŽäźL'èčĚéěřåŹĺïijŇåźűåřĘåĚűä¡IJçŤĺåIJĺåĚűäžŰåĞ¡æŢřæĹŰæŰźæşŢäÿŁãĂĆ

èğčåĘşæŰźæąĹ

åIJĺçśżéĞŇéİćåőŽäźL'èčĚéěřåŹĺå¿ĹçőĂå■ŢïijŇä¡ĘæŸŕä¡äéęŰåĚĹèęĄçąőèőď'åőČçŽĎä¡£çŤĺæŰźåijŔã
äÿŃéİćæĹŚäżňçŤĺä¿Ńå■ŘæİěéŸŘè£řåőČäżňçŽĎäÿ■åŘŇïijŽ

```
from functools import wraps

class A:
    # Decorator as an instance method
```

```
    def decorator1(self, func):
        @wraps(func)
        def wrapper(*args, **kwargs):
            print('Decorator 1')
            return func(*args, **kwargs)
        return wrapper

    # Decorator as a class method
    @classmethod
    def decorator2(cls, func):
        @wraps(func)
        def wrapper(*args, **kwargs):
            print('Decorator 2')
            return func(*args, **kwargs)
        return wrapper
```

äÿŃéİćæŸŕäÿĂä¡£çŤĺä¿Ńå■ŘïijŽ

```
# As an instance method
a = A()
@a.decorator1
def spam():
    pass
# As a class method
@A.decorator2
def grok():
    pass
```

äžŤçżĘèğĆåŕ§åŔŕäžěåŔŚçŐřäÿĂäÿłæŸŕåőđä¿ŃèřČçŤĺïijŇäÿĂäÿłæŸŕçśżèřČçŤĺãĂĆ

èőĺèőž

åIJÍçśżäÿ■åőŽäźL'èčĚéěřåŹĺåĹIçIJŃäÿŁåŐżåě¡åČŔå¿ĹåěĞæĂłïijŇä¡ĘæŸŕåIJÍæăĞåĞĘåžŞäÿ■æIJL'å¿Ĺ
çL'źåĹńçŽĎïijŇ@property èčĚéěřåŹĺåőđéŹĚäÿŁæŸŕäÿĂäÿłçśżïijŇåőČéĞŇéİćåőŽäźL'äžĘäÿL'äÿłæŰźæşŢ
getter(), setter(), deleter()，æŕŔäÿĂäÿłæŰźæşŢéČ¡æŸŕäÿĂäÿłèčĚéěřåŹĺãĂĆä¿ŃåęĆïijŽ

```
class Person:
    # Create a property instance
    first_name = property()

    # Apply decorator methods
    @first_name.getter
    def first_name(self):
        return self._first_name

    @first_name.setter
    def first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
```

```
        self._first_name = value
```

åőČäÿžäżĂäźĹèęĄè£ŹäźĹåőŽäźL'çŽĎäÿżèęĄåŐ§åŽäæŸŕåŘĎçğ■äÿ■åŘŇçŽĎèčĚéěřåŹĺæŰźæşŢäijŽåI
property åőđä¿ŃäÿŁæŞ■ä¡IJåőČçŽĎçŁűæĂĄãĂĆ åŽäæ■d'ïijŇäżżä¡ŢæŮűåĂŹåŘłèęĄä¡ăçćřåĹřéIJĂèęĄå

åIJÍçśžäÿ■åőŽäźL'èčĚéěřåŹĺæIJL'äÿłéŽ¿çŘĘèğčçŽĎåIJřæŰźåŕśæŸŕåŕźäžŐéćÍåd'ŰåŘĆæŢř
self æĹŰ cls çŽĎæ■čçąőä¡£çŤĺãĂĆ åř¡çőąæIJĂåd'ŰåśĆçŽĎèčĚéěřåŹĺåĞ¡æŢřæŕŤåęĆ
decorator1() æĹŰ decorator2() éIJĂèęĄæŘŘä¿ŻäÿĂäÿł self
æĹŰ cls åŘĆæŢřijŇ ä¡ĘæŸŕåIJÍäÿd'äÿłèčĚéěřåŹĺåĘĚéČÍèćńåĹŻåżžçŽĎ
wrapper() åĞ¡æŢřåźűäÿ■éIJĂèęĄåŇĚåŘńè£Źäÿł self åŘĆæŢřãĂĆ
ä¡ääŤŕäÿĂéIJĂèęĄè£ŹäÿłåŘĆæŢřæŸŕåIJÍä¡ăçąőåőđèęĄèő£éŮőåŇĚèčĚåŹÍäÿ■è£Źäÿłåőđä¿ŃçŽĎæ§ŘäžŻéČ

åŕźäžŐçśžéĞŇéİćåőŽäźL'çŽĎåŇĚèčĚåŹÍè£ŸæIJL'äÿĂçĆźæŕŤè¿ČéŽ¿çŘĘèğčïijŇåŕśæŸŕåIJÍæűL'åŘŁåÍ
ä¿ŃåęĆïijŇåĄĞèő¿ä¡ăæČşèőľåIJÍAäÿ■åőŽäźL'çŽĎèčĚéěřåŹÍä¡IJçŤĺåIJÍå■ŘçśžBäÿ■ãĂĆä¡ăéIJĂèęĄåČŘäÿŇ

```
class B(A):
    @A.decorator2
    def bar(self):
        pass
```

äź§åŕśæŸŕèŕťïijŇèčĚéěřåŹÍèęĄècńåőŽäźL'æĹŘçśżæŰźæşŢåźűäÿŤä¡ăå£ĚéążæŸ¿åijŔçŽĎä¡£çŤÍçĹűçśż
ä¡ăäÿ■èČ¡ä¡£çŤÍ @B.decorator2 ïijŇåŽääÿžåIJÍæŰźæşŢåőŽäźL'æŮűïijŇè£ŹäÿłçśžBè£ŸæšąæIJL'èćńåĹŽ

9.9 åřĘèčĚéěřåŹĺåőŽäźL'äÿžçśż

éŮőécŸ

ä¡ăæČşä¡£çŤÍäÿĂäÿłèčĚéěřåŹĺåŐżåŇĚèčĚåĞ¡æŢřijŇä¡ĘæŸŕåÿŇæIJŻè£ŤåŽđäÿĂäÿłåŔŕèřČçŤÍçŽĎåő
ä¡ăéIJĂèęĄèőľä¡ăçŽĎèčĚéěřåŹĺåŔŕäżěåŘŇæŮűåůěä¡IJåIJÍçśżåőŽäźL'çŽĎåĘĚéČĹåŠŇåd'ŰéČÍãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘåřĘèčĚéěřåŹĺåőŽäźL'æĹŘäÿĂäÿłåőđä¿ŃïijŇä¡ăéIJĂèęĄçąőä£ÍåőČåőđçŐřäžĘ
__call__() åŠŇ __get__() æŰźæşŢãĂĆ ä¿ŃåęĆïijŇäÿŃéİćçŽĎäżčçăĄåőŽäźL'äžĘäÿĂäÿłçśżïijŇåőČå

```
import types
from functools import wraps

class Profiled:
    def __init__(self, func):
        wraps(func)(self)
        self.ncalls = 0

    def __call__(self, *args, **kwargs):
        self.ncalls += 1
        return self.__wrapped__(*args, **kwargs)
```

```
    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return types.MethodType(self, instance)
```

ä¡ăåŔŕäžěåřĘåőČå¡ŞåĄŽäÿĂäÿłæŹőéĂŽçŽĎèčĚéěřåŹĺæİěä¡£çŤĺïijŇåIJĺçśžéĞŇéİćæĹŰåď ŰéİćéČ¡åŔŕ

```
@Profiled
def add(x, y):
    return x + y

class Spam:
    @Profiled
    def bar(self, x):
        print(self, x)
```

åIJĺäžď'äžŠçŐŕåćČäÿ■çŽĎä¡£çŤĺçď'žä¿ŃïijŽ

```
>>> add(2, 3)
5
>>> add(4, 5)
9
>>> add.ncalls
2
>>> s = Spam()
>>> s.bar(1)
<__main__.Spam object at 0x10069e9d0> 1
>>> s.bar(2)
<__main__.Spam object at 0x10069e9d0> 2
>>> s.bar(3)
<__main__.Spam object at 0x10069e9d0> 3
>>> Spam.bar.ncalls
3
```

èőĺèőž

ařĘèčĚéěřåŹĺåőŽäźĽ'æĹŘçśžéĂŽåÿÿæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆä¡ĘæŸŕè£ŹéĞŇè£ŸæŸŕæIJL'äÿĂäžŽçżĘ

éęŰåĚĹïijŇä¡£çŤĺ `functools.wraps()` åĞ¡æŢřçŽĎä¡IJçŤĺèů§äźŇåĽ■è£ŸæŸŕäÿĂæăůïijŇařĘèćńå

åĚűæňąïijŇéĂŽåÿÿå¿ĹåőźæŸŞäijŽå£¡èğĘäÿŁéİćçŽĎ `__get__()` æŰźæşŢãĂĆåęĆæđIJä¡ăå£¡çŢěåőČïijŇä£ÍæŇĄåĚűäžŰäžčçăĄäÿ■åŔŸåĘ■æňąè£ŘèąŇïijŇ ä¡ăäijŽåŔŞçŐřå¡Şä¡ăåŐżèřČçŤĺèćńèčĚéěřåőđä¿ŃæŰźæşŢæŮűåĞžçŐřå¿ĹåěĞæĂłçŽĎéŮőécŸãĂĆä¿ŃåęĆï

```
>>> s = Spam()
>>> s.bar(3)
Traceback (most recent call last):
```

```
...
TypeError: bar() missing 1 required positional argument: 'x'
```

åĞžéŤŹåŐ§åŻäæŸŕå¡ŞæŰźæşŢåĞ¡æŢřåIJĺäÿĂäÿłçśżäÿ■èćńæ§ěæL'¿æŮűïijŇåőČäžňçŽĎ
__get__() æŰźæşŢä¿Ìæ■őæŔŔè£řåŹĺå■ŔèőőèćńěřČçŢĺïijŇ
åIJĺ8.9åřŔèŁĆåůšçżŔèőšè£řè£ĞæŔŔè£řåŹĺå■Ŕèőőäž ĘãĂĆåIJĺè£ŹéĞŇïijŇ__get__()
çŽĎçŻőçŽĎæŸŕåĹŻåžžäÿĂäÿłçżŚåőŽæŰźæşŢåŕźèśą (æIJĂçżĹäijŽçżŹè£ŹäÿłæŰźæşŢäijäéĂŠselfåŔĆæŢř)ã

```
>>> s = Spam()
>>> def grok(self, x):
...     pass
...
>>> grok.__get__(s, Spam)
<bound method Spam.grok of <__main__.Spam object at 0x100671e90>>
>>>
```

__get__() æŰźæşŢæŸŕäÿžäžEçąőä£ĺçżŚåőŽæŰźæşŢåŕźèśąèČ¡èćńæ■čçąőçŽĎåĹŻåžžãĂĆ
type.MethodType() æL'ŃåŁĺåĹŻåžžäÿĂäÿłçżŚåőŽæŰźæşŢæĺěä¡£çŤĺãĂĆåŔłæIJL'å¡Şåőđä¿Ńèćńä¡£çŤ
åęĆæđIJè£ŹäÿłæŰźæşŢæŸŕåIJĺçśżäÿŁéĺćæĺěèő£éŮőïijŇ éĆčäźĹ __get__() äÿ■çŽĎin-
stanceåŔĆæŢřäijŽèćńèő¿ç¡őæĹŘNoneåźűçŻť'æŐěè£ŤåŻđ Profiled åőđä¿ŃæIJňèžńãĂĆ
è£ŹæăůçŽĎèŕĺæĹŚäžňářšåŔŕäžěæŔŘåŔŰåőČçŽĎ ncalls åšđæĂğäžEãĂĆ

åęĆæđIJä¡ăæČşéĄ£åĚ■äÿĂäžŻæůůäźśïijŇäź§åŔŕäžěèĂČèŹŚåŔęåď'ŰäÿĂäÿłä¡£çŤĺéŮ■åŇĚåŠŇ
nonlocal åŔŸéĞŔåőđçŐřçŽĎèčĚéěřåŹĺïijŇè£ŹäÿłåIJĺ9.5åřŔèŁĆæIJL'èőšåĹřãĂĆä¿ŃåęĆïijŽ

```
import types
from functools import wraps

def profiled(func):
    ncalls = 0
    @wraps(func)
    def wrapper(*args, **kwargs):
        nonlocal ncalls
        ncalls += 1
        return func(*args, **kwargs)
    wrapper.ncalls = lambda: ncalls
    return wrapper

# Example
@profiled
def add(x, y):
    return x + y
```

è£ŹäÿłæŰźåijŔèů§äźŃåL'■çŽĎæŢĹæđIJåĞääźŐäÿĂæăůïijŇéŹď'äžĘåŕźäžŐ ncalls
çŽĎèő£éŮőçŐřåIJĺæŸŕéĂŽè£ĞäÿĂäÿłèćńçżŚåőŽäÿžåšđæĂğçŽĎåĞ¡æŢřæĺěåőđçŐřïijŇä¿ŃåęĆïijŽ

```
>>> add(2, 3)
5
>>> add(4, 5)
9
>>> add.ncalls()
```

```
2
>>>
```

9.10 äÿžçśżåŠŇéİŹæĂĄæŰźæşŢæŔŘä¿ŻèčĚéěřåŹÍ

éŮőécŸ

ä¡ăæČşçżŹçśżæĹŰéİŹæĂĄæŰźæşŢæŔŘä¿ŻèčĚéěřåŹÍãĂĆ

èğčåĘşæŰźæąĹ

çżŹçśżæĹŰéİŹæĂĄæŰźæşŢæŔŘä¿ŻèčĚéěřåŹÍæŸŕå¿ĹçőĂå■ŢçŽĎïijŇäÿ■è£ĞèęAçąőä£ÌèčĚéěřåŹÍåIJ
`@classmethod` æĹŰ `@staticmethod` äźŃåĽ■ãĂĆä¿ŃåęĆïijŽ

```
import time
from functools import wraps

# A simple decorator
def timethis(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.time()
        r = func(*args, **kwargs)
        end = time.time()
        print(end-start)
        return r
    return wrapper

# Class illustrating application of the decorator to different␣
↪kinds of methods
class Spam:
    @timethis
    def instance_method(self, n):
        print(self, n)
        while n > 0:
            n -= 1

    @classmethod
    @timethis
    def class_method(cls, n):
        print(cls, n)
        while n > 0:
            n -= 1

    @staticmethod
    @timethis
```

```
    def static_method(n):
        print(n)
        while n > 0:
            n -= 1
```

èčĚéěřåŘŐçŽĎçśžåŠŇéİŹæĂĄæŰźæşŢåŕŕæ■čåÿÿåůěä¡IJïijŇåŕłäÿ■è£Ğåćđå£ääžĘéćİåď˙ŰçŽĎèőąæŮ

```
>>> s = Spam()
>>> s.instance_method(1000000)
<__main__.Spam object at 0x1006a6050> 1000000
0.11817407608032227
>>> Spam.class_method(1000000)
<class '__main__.Spam'> 1000000
0.11334395408630371
>>> Spam.static_method(1000000)
1000000
0.11740279197692871
>>>
```

èőÍèőž

åęĆæđIJä¡ăæŁŁèčĚéěřåŹÍçŽĎéąžåžŔåĘŹéŤŹäžĘåřśäijŽåĞžéŤŹãĂĆä¿ŃåęĆïijŇåĄĞèő¿ä¡ääČŔäÿŃéİć

```
class Spam:
    @timethis
    @staticmethod
    def static_method(n):
        print(n)
        while n > 0:
            n -= 1
```

éĆčäźĹä¡ăèřČçŤÍè£ŹäÿłéŢIJåď˙æŰźæşŢæŮűåřśäijŽæŁěéŤŹïijŽ

```
>>> Spam.static_method(1000000)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "timethis.py", line 6, in wrapper
start = time.time()
TypeError: 'staticmethod' object is not callable
>>>
```

éŮőécŸåIJÍäžŐ `@classmethod` åŠŇ `@staticmethod` åőđéŹĚäÿŁåźűäÿ■äijŽåĹŻåžžåŔŕçŻť˙æŐěřČçŤÍçŽĎåŕźèśąïijŇ
èĂŇæŸŕåĹŻåžžçL’źæőŁçŽĎæŔŔè£řåŹÍåŕźèśą(åŔĆèĂČ8.9åřŔèŁĆ)ãĂĆåŻăæ■ď˙å¡Şä¡ăèŕŢçİĂåIJÍåĚűäžŰèč
çąőä£Ìè£Źçğ■èčĚéěřåŹÍåĞžçŐřåIJÍèčĚéěřåŹÍéŞ¿äÿ■çŽĎçňňäÿĂäÿłä¡■ç¡őåŔŕäžëä£őåď˙■è£ŹäÿłéŰőécŸãĂĆ

å¡ŞæĹŚäžňåIJÍæŁ¡èśąå§žçśżäÿ■åőŽäźL’çśżæŰźæşŢåŠŇéİŹæĂĄæŰźæşŢ(åŔĆèĂČ8.12åřŔèŁĆ)æŮűïijŃ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăæČşåőŽäźL’äÿĂäÿłæŁ¡èśąçśżæŰźæşŢïijŇåŕŕäžëä¡£çŤÍçśżäijijäÿŃéİćçŽĎäžčçăĄïijŽ

```
from abc import ABCMeta, abstractmethod
class A(metaclass=ABCMeta):
    @classmethod
    @abstractmethod
    def method(cls):
        pass
```

åIJĺè£ŹæőţäžčçăĄäÿ■ïijŇ@classmethod èů§ @abstractmethod äÿd'èĂĚçŽĎéąžåžŔæŸŕæIJL'èőšçl'űçŽĎïijŇåęĆæđIJä¡ăèřČæ■ćåőČäžňçŽĎéąžåžŔåřšäijŽåĞžéŤŹãĂĆ

9.11 èčĚéěřåŹĺäÿžèćńåŇĚèčĚåĞ¡æŢřåćđåŁăåŔĆæŢř

éŮőécŸ

ä¡ăæČşåIJĺèčĚéěřåŹĺäÿ■çżŹèćńåŇĚèčĚåĞ¡æŢřåćđåŁăécİåd'ŰçŽĎåŔĆæŢřïijŇä¡ĘæŸŕäÿ■èČ¡å¡śåŞ■è£Ź

èğčåĘşæŰźæąĹ

åŔŕäžěä¡£çŤĺåĚşéŤőå■ŮåŔĆæŢřæİěçżŹèćńåŇĚèčĚåĞ¡æŢřåćđåŁăécİåd'ŰåŔĆæŢřãĂĆèĂČèŹŚäÿŃéİćç

```
from functools import wraps

def optional_debug(func):
    @wraps(func)
    def wrapper(*args, debug=False, **kwargs):
        if debug:
            print('Calling', func.__name__)
        return func(*args, **kwargs)

    return wrapper
```

```
>>> @optional_debug
... def spam(a,b,c):
... print(a,b,c)
...
>>> spam(1,2,3)
1 2 3
>>> spam(1,2,3, debug=True)
Calling spam
1 2 3
>>>
```

èőĺèőž

éĂŽè£ĞècĚéěřåŹĺæİěçżŹècńåŇĚècĚåĞ¡æŢřåćđåŁääŔĆæŢřçŽĎåĄŽæşŢåźűäÿ■åÿÿèğĄãĂĆ
ař¡çőąåęĆæ■ď'ïijŇæIJL'æŮűåĂŹåőČåŔŕäžěéĄ£åĚ■äÿĂäžŽéĞ■åď'■äžčçăĄãĂĆä¿ŃåęĆïijŇåęĆæđIJä¡ăæIJL'

```
def a(x, debug=False):
    if debug:
        print('Calling a')

def b(x, y, z, debug=False):
    if debug:
        print('Calling b')

def c(x, y, debug=False):
    if debug:
        print('Calling c')
```

éĆčäźĹä¡ăåŔŕäžěářĘåĚűéĞ■æđĎæĹŘè£Źæăůïij Ž

```
from functools import wraps
import inspect

def optional_debug(func):
    if 'debug' in inspect.getargspec(func).args:
        raise TypeError('debug argument already defined')

    @wraps(func)
    def wrapper(*args, debug=False, **kwargs):
        if debug:
            print('Calling', func.__name__)
        return func(*args, **kwargs)
    return wrapper

@optional_debug
def a(x):
    pass

@optional_debug
def b(x, y, z):
    pass

@optional_debug
def c(x, y):
    pass
```

è£Źçğ■åőđçŐřæŰźæąĹäźŃæL'Ăäžěèą Ňå¿ŮéĂŽïijŇåIJĺäžŐåijžåĹűåĚşéŤőå■ŮåŔĆæŢřå¿Ĺåőźæ ŸŞèćńa
`*args` åŠŇ `**kwargs` åŔĆæŢřçŽĎåĞ¡æŢřäÿ■ãĂĆ éĂŽè£Ğä¡£çŤĺåijžåĹűåĚşéŤőå■ŮåŔĆæŢřïijŇåőČèćńä
åźűäÿŤæŐěäÿŃæİěäžĚäżĚä¡£çŤĺåL'l'ä¡ŹçŽĎä¡■ç¡őåŠŇåĚşéŤőå■ŮåŔĆæŢřåŐżèřČçŤĺè£ŹäÿłåĞ¡æŢřæŮűïij
äź§åřśæŸŕèřťïijŇåőČåźűäÿ■äijŽèćńçžşåĚěåĹř `**kwargs` äÿ■åŐžãĂĆ

è£ŸæIJL'äÿĂäÿłéŽ¿çĆźåřśæŸŕåęĆä¡ŢåŐžåď'ĎçŘĘèćńæůżåŁăçŽĎåŔĆæŢřäÿŐèćńåŇĚècĚåĞ¡æŢřåŔĆ

ä¿ŃåęĆïijŇåęĆæđIJèčĖéěřåŹĺ `@optional_debug` ä¡IJçŤĺåIJĺäÿĂäÿłåůšçżŕæŃěæIJL'äÿĂäÿł `debug` åŕĆæŢřçŽĎåĞ¡æŢřäÿŁæŮűäijŽæIJL'éŮőécŸãĂĆ è£ŹéĞŇæĹŚäžňáćđåŁääžĘäÿĂæ■ěåŘ■å■ŮæčĂæ

äÿŁéİćçŽĎæŰźæąĹè£ŸåŕŕäżěæŽť åőŇç¿ŐäÿĂçĆźïijŇåŽääÿžçš¿æŸŐçŽĎçĺŃåžŕåŚŸåžŤèŕěåŕŚçŐřäž

```
>>> @optional_debug
... def add(x,y):
...     return x+y
...
>>> import inspect
>>> print(inspect.signature(add))
(x, y)
>>>
```

éĂŽè£ĞąęĆäÿŃçŽĎä£őæŤźïijŇåŕŕäżěèğčåĘşè£ŹäÿłéŮőécŸïijŽ

```
from functools import wraps
import inspect

def optional_debug(func):
    if 'debug' in inspect.getargspec(func).args:
        raise TypeError('debug argument already defined')

    @wraps(func)
    def wrapper(*args, debug=False, **kwargs):
        if debug:
            print('Calling', func.__name__)
        return func(*args, **kwargs)

    sig = inspect.signature(func)
    parms = list(sig.parameters.values())
    parms.append(inspect.Parameter('debug',
                 inspect.Parameter.KEYWORD_ONLY,
                 default=False))
    wrapper.__signature__ = sig.replace(parameters=parms)
    return wrapper
```

éĂŽè£Ğè£ŹæăůçŽĎä£őæŤźïijŇåŇĚèčĚåŘŐçŽĎåĞ¡æŢřç■¿åŘ■åřśèČ¡æ■čçąőçŽĎæŸ¿çď'ž `debug` åŕĆæŢřçŽĎå■ŸåIJĺäžĘãĂĆä¿ŃåęĆïijŽ

```
>>> @optional_debug
... def add(x,y):
...     return x+y
...
>>> print(inspect.signature(add))
(x, y, *, debug=False)
>>> add(2,3)
5
>>>
```

åŕĆèĂČ9.16åřŔèŁĆèŐůåŘŰæŽť åď'ŽåĚşäžŐåĞ¡æŢřç■¿åŘ■çŽĎä£ąæĄŕãĂĆ

9.12 ä¡£çŤĺèčĚéěřåŹĺæĽľåĚĚçśżçŽĎåŁ§èČ¡

éŮőécŸ

ä¡ăæČşéĂŽè£ĞåŔ■çIJĄæĹŰèĂĚéĞ■åĘŹçśżåőŽäźL'çŽĎæ§ŘéČĺåĹĘæİěä£őæŤźåőČçŽĎèąŇäÿžïijŇä¡E

èğčåĘşæŰźæąĹ

è£Źçğ■æČĚåĘţåŔŕèČ¡æŸŕçśżèčĚéěřåŹĺæIJĂåě¡çŽĎä¡£çŤĺåIJžæŹŕäžĘãĂĆä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿĂä
__getattribute__ çŽĎçśżèčĚéěřåŹĺïijŇ åŔŕäżěæĽŞå■ŕæŮěå£ŮïijŽ

```
def log_getattribute(cls):
    # Get the original implementation
    orig_getattribute = cls.__getattribute__

    # Make a new definition
    def new_getattribute(self, name):
        print('getting:', name)
        return orig_getattribute(self, name)

    # Attach to the class and return
    cls.__getattribute__ = new_getattribute
    return cls

# Example use
@log_getattribute
class A:
    def __init__(self,x):
        self.x = x
    def spam(self):
        pass
```

äÿŃéİćæŸŕä¡£çŤĺæŢĹæđIJïijŽ

```
>>> a = A(42)
>>> a.x
getting: x
42
>>> a.spam()
getting: spam
>>>
```

èőĺèőž

çśżèčĚéěřåŹĺéĂŽåÿÿåŔŕäżěä¡IJäÿžåĚűäżŰéńŸçžğæŁĂæIJŕæŕŤåęĆæůůåĚěæĹŰåĚČçśżçŽĎäÿĂçğ■éİđă
ærŤåęĆïijŇäÿŁéİćçd'žä¿Ńäÿ■çŽĎåŔęåd'ŰäÿĂçğ■åőđçŐŕä¡£çŤĺåĹŕçžğæL'£ïijŽ

```
class LoggedGetattribute:
    def __getattribute__(self, name):
        print('getting:', name)
        return super().__getattribute__(name)

# Example:
class A(LoggedGetattribute):
    def __init__(self,x):
        self.x = x
    def spam(self):
        pass
```

è£Źçğ■æŰźæąĹäź§èąŇå¿ŮéĂŽïijŇä¡ĘæŸŕäÿžäžĘåŐżçŘĘèğčåőČïijŇä¡ääŕšå£Ěéążç§ěéĄŞæŰźæşŢèřČ
äžěåŔŁåĚűåőČ8.7åŕŔèŁĆäżŃçż■çŽĎçżğæL'£ç§ěèŕĘãĂĆ æ§Řçğ■çÍŃåžęäÿŁæİěèőšïijŇçśżèčĚéěřåŹÍæŰźæą
åŻääÿžäžŰåźűäÿ■ä¿ÌètŰ `super()` åĞ¡æŢřãĂĆ

åęĆæđIJä¡ăçşżæČşåIJÍäÿĂäÿłçśżäÿŁéÍćä¡£çŤÍåd'ŽäÿłçśżèčĚéěřåŹÍïijŇéĆčäźĹåŕšéIJĂèęĄæşÍæĎŔäÿŃé
ä¿ŃåęĆïijŇäÿĂäÿłèčĚéěřåŹÍAäijŽåřĘåĚűèčĚéěřçŽĎæŰźæşŢåőŇæŢt'æŻ£æ■ćæĹŘåŔęäÿĂçğ■åőđçŐřïijŇ
èĂŇåŔęäÿĂäÿłèčĚéěřåŹÍBåŔłæŸŕçőĂå■ŢçŽĎåIJÍåĚűèčĚéěřçŽĎæŰźæşŢäÿ■æůżåŁăçĆźéĆÍåd'ŰéĂżè¿ŚãĂ
éĆčäźĹè£ŹæŮűåĂŹèčĚéěřåŹÍAåŕšéIJĂèęĄæŤ¿åIJÍèčĚéěřåŹÍBçŽĎåL'■éÍćãĂĆ

ä¡ăè£ŸåŔŕäžěåŻđéą¿äÿĂäÿŃ8.13åřŔèŁĆåŔęåd'ŰäÿĂäÿłåĚşäžŐçśżèčĚéěřåŹÍçŽĎæIJL'çŤÍçŽĎä¿Ńå■Ř

9.13 ä¡£çŤÍåĚČçśżæŐğåĹűåőđä¿ŃçŽĎåĹŻåžž

éŮőécŸ

ä¡ăæČşéĂŽè£ĞæŤźåŔŸåőđä¿ŃåĹŻåžžæŰźåijŔæÌěåőđçŐřå■Ţä¿ŃãĂĄçijŞå■ŸæĹŰåĚűäžŰçśżäijijçŽĎ

èğčåĘşæŰźæąĹ

PythonçÍŃåžŔåŚŸéČ¡ç§ěéĄŞïijŇåęĆæđIJä¡ääőŽäźL'äžĘäÿĂäÿłçśżïijŇåŕšèČ¡åČŔåĞ¡æŢřäÿĂæăůçŽĎè

```
class Spam:
    def __init__(self, name):
        self.name = name

a = Spam('Guido')
b = Spam('Diana')
```

åęĆæđIJä¡ăæČşèĞłåőŽäźL'è£Źäÿłæ■ěéłd'ïijŇä¡ääŔŕäžěåőŽäźL'äÿĂäÿłåĚČçśżåźűèĞłåůšåőđçŐř
`__call__()` æŰźæşŢãĂĆ

äÿžäžĘæijŤçd'žïijŇåĄĞèő¿ä¡ääÿ■æČşäżżä¡ŢäžžåĹŻåžžè£ŹäÿłçśżçŽĎåőđä¿ŃïijŽ

```
class NoInstances(type):
    def __call__(self, *args, **kwargs):
```

```
        raise TypeError("Can't instantiate directly")

# Example
class Spam(metaclass=NoInstances):
    @staticmethod
    def grok(x):
        print('Spam.grok')
```

è£Źæåůç ŽĎèřÍïijŇçŤÍæĹůåŔłèČ¡èřČçŤÍè£ŹäÿłçśżçŽĎéİŹæĂĄæŰźæşŢïijŇèĂŇäÿ■èČ¡ä¡£çŤÍéĂŽåÿÿç

```
>>> Spam.grok(42)
Spam.grok
>>> s = Spam()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example1.py", line 7, in __call__
        raise TypeError("Can't instantiate directly")
TypeError: Can't instantiate directly
>>>
```

çŐřåIJÍïijŇåĄĞąęĆä¡ăæČşåőđçŐřå■Ţä¿ŃæÍąåijŔïijĹåŔłèČ¡åĹŻåżžåŤŕäÿĂåőđä¿ŃçŽĎçśżïijL'ïijŇåőđçĆ

```
class Singleton(type):
    def __init__(self, *args, **kwargs):
        self.__instance = None
        super().__init__(*args, **kwargs)

    def __call__(self, *args, **kwargs):
        if self.__instance is None:
            self.__instance = super().__call__(*args, **kwargs)
            return self.__instance
        else:
            return self.__instance

# Example
class Spam(metaclass=Singleton):
    def __init__(self):
        print('Creating Spam')
```

éĆčäźĹSpamçśżåřśåŔłèČ¡åĹŻåżžåŤŕäÿĂçŽĎåőđä¿ŃäžĘïijŇæijŤçd'žąęĆäÿŃïijŽ

```
>>> a = Spam()
Creating Spam
>>> b = Spam()
>>> a is b
True
>>> c = Spam()
>>> a is c
True
>>>
```

æIJĂåŘŐïijŇåĄĞèő¿ä¡ăæČşåĹŻåžž8.25åřŔèŁĆäÿ■éĆčæăůçŽĎçijŞå■Ÿåőđä¿ŃãĂĆäÿŃéİćæĹŚäžňåŕŕä

```
import weakref

class Cached(type):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.__cache = weakref.WeakValueDictionary()

    def __call__(self, *args):
        if args in self.__cache:
            return self.__cache[args]
        else:
            obj = super().__call__(*args)
            self.__cache[args] = obj
            return obj

# Example
class Spam(metaclass=Cached):
    def __init__(self, name):
        print('Creating Spam({!r})'.format(name))
        self.name = name
```

çĎűåŘŐæĹŚäź§æİěæţŃèŕŢäÿĂäÿŃïijŽ

```
>>> a = Spam('Guido')
Creating Spam('Guido')
>>> b = Spam('Diana')
Creating Spam('Diana')
>>> c = Spam('Guido') # Cached
>>> a is b
False
>>> a is c # Cached value returned
True
>>>
```

èőÍèőž

åĹľçŤĺåĚČçśžåőđçŐřåď'Žçğ■åőđä¿ŃåĹŻåžžæÍąåijŔéĂŽåÿÿèęĄæŕŤäÿ■ä¡£çŤĺåĚČçśžçŽĎæŰźåijŔäijŸ

åĄĞèő¿ä¡ääÿ■ä¡£çŤĺåĚČçśžïijŇä¡ăåŔřèČ¡éIJĂèęĄåřĘçśžéŽŘèŮŔåIJÍæ§ŘäžŽåůěåŐĆåĞ¡æŢřåŘŐéİćãĂ
æŕŤåęĆäÿžäžĘåőđçŐřäÿĂäÿłå■Ţä¿ŃïijŇä¡ää¡ăåŔřèČ¡äijŽåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
class _Spam:
    def __init__(self):
        print('Creating Spam')

_spam_instance = None

def Spam():
```

```
    global _spam_instance

    if _spam_instance is not None:
        return _spam_instance
    else:
        _spam_instance = _Spam()
        return _spam_instance
```

åř¡çőąä¡£çŢĺåĚČçśżåŕŕèČ¡äijŽæűĽ'åŔŁåĹřæŕŤè¿ČéńŸçžğçĆźçŽĎæŁĂæIJŕïijÑä¡ĘæŸŕåőČçŽĎäżčçăĄ

æŻť'åď'ŽåĚşäžŐåĹŻåżžçijŞå■Ÿåőđä¿ÑãĂĄåijśåijŢçŢĺç■Ľ'åĘĚåőźïijÑèŕůåŔĆèĂČ8.25åŕŔèŁĆãĂĆ

9.14 æ■ŢèŐůçśżçŽĎåśđæĂğåőŽäźĽéąžåžŔ

éŮőécŸ

ä¡ăæČşèĞłåŁĺèőřå¡ŢäÿĂäÿłçśżäÿ■åśđæĂğåŠÑæŰźæşŢåőŽäźĽçŽĎéąžåžŔïijÑçĎűåŘŐåŔŕäżčåĹľçŢĺåőČæİěåĄŽå¿Ĺåď'ŽæŞ■ä¡IJïijĹæŕŤåęĆåžŔåĹŮåŇŰãĂĄæŸäářĎåĹřæŢřæ■őåžŞç■Ľ'ç

èğčåĘşæŰźæąĹ

åĹľçŢĺåĚČçśżåŔŕäżčå¿ĹåőźæŸŞçŽĎæ■ŢèŐůçśżçŽĎåőŽäźĽ'ä£ąæĄŕãĂĆäÿŇéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijÑ

```
from collections import OrderedDict

# A set of descriptors for various types
class Typed:
    _expected_type = type(None)
    def __init__(self, name=None):
        self._name = name

    def __set__(self, instance, value):
        if not isinstance(value, self._expected_type):
            raise TypeError('Expected ' + str(self._expected_type))
        instance.__dict__[self._name] = value

class Integer(Typed):
    _expected_type = int

class Float(Typed):
    _expected_type = float

class String(Typed):
    _expected_type = str

# Metaclass that uses an OrderedDict for class body
class OrderedMeta(type):
```

```
    def __new__(cls, clsname, bases, clsdict):
        d = dict(clsdict)
        order = []
        for name, value in clsdict.items():
            if isinstance(value, Typed):
                value._name = name
                order.append(name)
        d['_order'] = order
        return type.__new__(cls, clsname, bases, d)

    @classmethod
    def __prepare__(cls, clsname, bases):
        return OrderedDict()
```

åIJĺè£Źäÿłå ĚČçśżäÿ■ïijŇæL'ğèąŇçśżäÿżä¡ŞæŮűæŔŕè£řåŹĺçŽĎåőŽäźL'éąžåžŕäijŽèćńäÿĂäÿł
OrderedDict``æ■ŢèŐůåĹřïijŇ çŤŞæĹŘçŽĎæIJL'åžŕåŕ■çğřäżŐå■ŮåĚÿäÿ■æŔŘåŔŰåĞžæİě
``_order äÿ■ãĂĆè£Źæăůç ŽĎèŕĺçśżäÿ■çŽĎæŰźæşŢåŕřäžěéĂŽè£Ğåď'Žçğ■æŰźåijŕæİěä¡£çŤĺåőČãĂĆ
ä¿ŇåęĆïijŇäÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎçśżïijŇä¡£çŤĺè£ŹäÿłæŐŠåžŕå■ŮåĚÿæİěåőđçŐřårĘäÿĂäÿłçśżåőđă

```
class Structure(metaclass=OrderedMeta):
    def as_csv(self):
        return ','.join(str(getattr(self,name)) for name in self._
 ↪order)

# Example use
class Stock(Structure):
    name = String()
    shares = Integer()
    price = Float()

    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

æĹŚäžňåIJĺäžď'äžŠåijŕçŐŕåćČäÿ■æţŃèŕŢäÿĂäÿŃè£ŹäÿłStockçśżïijŽ

```
>>> s = Stock('GOOG',100,490.1)
>>> s.name
'GOOG'
>>> s.as_csv()
'GOOG,100,490.1'
>>> t = Stock('AAPL','a lot', 610.23)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "dupmethod.py", line 34, in __init__
TypeError: shares expects <class 'int'>
>>>
```

èőĺèőž

æIJňèŁĆäÿĂäÿłåĚşéŤőçĆźåřśæŸŕOrderedMetaåĚČçśżäÿ■åőŽäźĽçŽĎ ‘‘ __prepare__()‘‘ æŰźæşŢãĂĆ è£ŹäÿłæŰźæşŢäijŽåIJĺåijĂåğŃåőŽäźĽçśżåŠŇåőČçŽĎçĽűçśżçŽĎæŮűåĂŹèćńæĽğè æĹŚäžňè£ŹéĞŇéĂŽè£Ğè£ŤåŻđäžĘäÿĂäÿłOrderedDictèĂŇäÿ■æŸŕäÿĂäÿłæŹőéĂŽçŽĎå■ŮåĚÿïijŇåŕŕäžěå

åęĆæđIJä¡ăæČşæđĎéĂăèĞłåůśçŽĎçśżå■ŮåĚÿåŕźèśąïijŇåŕŕäžěå¿Ĺåőźæ ŸŞçŽĎæĽ'ľåśŢè£ŹäÿłåŁ§èČ¡ã

```
from collections import OrderedDict

class NoDupOrderedDict(OrderedDict):
    def __init__(self, clsname):
        self.clsname = clsname
        super().__init__()
    def __setitem__(self, name, value):
        if name in self:
            raise TypeError('{} already defined in {}'.format(name,␣
↪self.clsname))
        super().__setitem__(name, value)

class OrderedMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        d = dict(clsdict)
        d['_order'] = [name for name in clsdict if name[0] != '_']
        return type.__new__(cls, clsname, bases, d)

    @classmethod
    def __prepare__(cls, clsname, bases):
        return NoDupOrderedDict(clsname)
```

äÿŃéÍćæĹŚäžňæţŃèŕŢéĞ■åď■çŽĎåőŽäźĽäijŽåĞžçŐřäžĂäźĹæČĚåĘţïijŽ

```
>>> class A(metaclass=OrderedMeta):
...     def spam(self):
...         pass
...     def spam(self):
...         pass
...
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "<stdin>", line 4, in A
    File "dupmethod2.py", line 25, in __setitem__
        (name, self.clsname))
TypeError: spam already defined in A
>>>
```

æIJĂåŘŐè£ŸæIJL'äÿĂçĆźå¿ĹéĞ■èęĄïijŇåřśæŸŕåIJĺ __new__() æŰźæşŢäÿ■åŕźäžŐåĚČçśżäÿ■èćńä£őæŤźå■ŮåĚÿçŽĎåď'ĎçŘĘãĂĆ åř¡çőąçśżä¡£çŤĺäžĘåŔęåď'ŰäÿĂäÿłå■ŮåĚÿæÌěåőŽäźĽ'ïijŇåIJĺæđĎéĂăæIJĂçźĹçŽĎ class åŕźèśąçŽĎæŮűåĂŹïijŇ æĹŚäžňäż■çĎűéIJĂèęĄåřĘè£Źäÿłå■ŮåĚÿè¡ňæ■ćäÿžäÿĂäÿłæ■ĉçąőçŽĎ dict åőđä¿ŃãĂĆ éĂŽè£ĞèŕåŔě d = dict(clsdict)

æİěåőŇæĹŘè£ŹäÿłæŢĹæđIJãĂĆ

åŕźäžŐå¿Ĺåď˙ŽåžŤçŤĺçĺŃåžŔèĂŇåůšïijŇèČ¡åď§æ■ŢèŐůçśżåőŽäźĽçŽĎéąžåžŔæŸŕäÿĂäÿłçIJŃäijijäÿ■ä¿ŃåęĆïijŇåIJĺåŕźèśąåĚşçşżæŸăåřĎäÿ■ïijŇæĹŚäžňéĂŽåÿÿäijŽçIJŃåĹřäÿŃéİćè£Źçğ■æŰźåijŔåőŽäźĽçŽĎç

```
class Stock(Model):
    name = String()
    shares = Integer()
    price = Float()
```

åIJĺæąĘæđűåžŢåśĆïijŇæĹŚäžňå£Ěéążæ■ŢèŐůåőŽäźĽçŽĎéąžåžŔæİěåřĘåŕźèśąæŸăåřĎåĹřåĚČçżĎæĹÚ
`as_csv()` çŽĎåŁ§èČ¡ïijĽãĂĆ è£ŹèŁĆæijŤçď˙žçŽĎæŁĂæIJŕéİđåÿÿçőĂå■ŢïijŇåźűäÿŤéĂŽåÿÿäijŽæŕŤåĚú

9.15 åőŽäźĽælJĽåŔŕéĂĽåŔĆæŢřçŽĎåĚČçśż

éŮőécŸ

ä¡ăæČşåőŽäźĽäÿĂäÿłåĚČçśżïijŇåĚĄèőÿçśżåőŽäźĽæŮűæŔŘä¿ŻåŔŕéĂĽåŔĆæŢřïijŇè£ŹæăůåŔŕäžěæĆ

èğčåĘşæŰźæąĹ

åIJĺåőŽäźĽçśżçŽĎæŮűåĂŹïijŇPythonåĚĄèőÿæĹŚäžňä¡£çŤĺ
“metaclass“åĚşéŤőå■ŮåŔĆæŢřæİěæŇĞåőŽçĽźåőŽçŽĎåĚČçśżãĂĆ
ä¿ŃåęĆä¡£çŤĺæŁ¡èśąå§žçśżïijŽ

```
from abc import ABCMeta, abstractmethod
class IStream(metaclass=ABCMeta):
    @abstractmethod
    def read(self, maxsize=None):
        pass

    @abstractmethod
    def write(self, data):
        pass
```

çĎűèĂŇïijŇåIJĺèĞłåőŽäźĽçŽĎéąžäÿ■æĹŚäžňè£ŸåŕźèæŔŘä¿ŻåĚűäžŰçŽĎåĚşéŤőå■ŮåŔĆæŢřïijŇå

```
class Spam(metaclass=MyMeta, debug=True, synchronize=True):
    pass
```

äÿžäžĘä¡£åĚČçśżæŤŕæŇąè£ŹäžŻåĚşéŤőå■ŮåŔĆæŢřïijŇä¡ăå£Ěéążçąőä£ĺåIJĺ
`__prepare__()` , `__new__()` åŠŇ `__init__()` æŰźæşŢäÿ■
éČ¡ä¡£çŤĺåijžåĹűåĚşéŤőå■ŮåŔĆæŢřãĂĆářšåČŕäÿŃéİćè£ŹæăůïijŽ

```
class MyMeta(type):
    # Optional
    @classmethod
```

```
    def __prepare__(cls, name, bases, *, debug=False,␣
↪synchronize=False):
        # Custom processing
        pass
        return super().__prepare__(name, bases)

    # Required
    def __new__(cls, name, bases, ns, *, debug=False,␣
↪synchronize=False):
        # Custom processing
        pass
        return super().__new__(cls, name, bases, ns)

    # Required
    def __init__(self, name, bases, ns, *, debug=False,␣
↪synchronize=False):
        # Custom processing
        pass
        super().__init__(name, bases, ns)
```

èőĺèőž

çżŹäÿĂäÿłåĚČçśżæůżåŁăåŕŕéĂĽåĚşéŤőå■ŮåŔĆæŢřéIJĂèęĄä¡ăåőŇåĚĺåijĎæĞĆçśżåĹŻåžžçŽĎæĽĂa
åŻääÿžè£ŹäžŻåŔĆæŢřäijŽèćńäijăéĂŠçżŹæŕŔäÿĂäÿłçŻÿåĚşçŽĎæŰźæşŢãĂĆ
`__prepare__()` æŰźæşŢåIJĺæĽĂæIJĽçśżåőŽäźĽåijĂåğŇæĽğèąŇåĽ■éęŰåĚĹèćńèřČçŤĺïijŇçŤĺæİěåĹŻ
éĂŽåÿÿæİěèőšïijŇè£ŹäÿłæŰźæşŢåŔłæŸŕçőĂå■ŢçŽĎè£ŤåŻđäÿĂäÿłå■ŮåĚÿæĹŰåĚűäžŰæŸäåřĎåŕźèśąãĂĆ
`__new__()` æŰźæşŢèćńçŤĺæİěåőđä¿ŃåŇŰæIJĂçżĹçŽĎçśżåŕźèśąãĂĆåőČåIJĺçśżçŽĎäÿżä¡Şèćńæ Ľğèą Ňåő
`__init__()` æŰźæşŢæIJĂåŘŐèćńèřČçŤĺïijŇçŤĺæİěæĽğèąŇåĚűäžŰçŽĎäÿĂäžŻåĹİåğŃåŇŰåůěä¡IJãĂĆ

å¡ŞæĹŚäžňæđĎéĂäåĚČçśżçŽĎæŮűåĂŹïijŇéĂŽåÿÿåŔłéIJĂèęĄåőŽäźĽäÿĂäÿł
`__new__()` æĹŰ `__init__()` æŰźæşŢïijŇä¡Ęäÿ■æŸŕäÿđ'äÿłéČ¡åőŽäźĽãĂĆ
ä¡ĘæŸŕïijŇåęĆæđIJéIJĂèęĄæŐěåŘŮåĚűäžŰçŽĎåĚşéŤőå■ŮåŔĆæŢřçŽĎĐérİïijŇè£ŹäÿđäÿłæŰźæşŢåřśèęĄå
éżŸèőđ'çŽĎ `__prepare__()` æŰźæşŢæŐěåŘŮäžżæĎŔçŽĎåĚşéŤőå■ŮåŔĆæŢřïijŇä¡ĘæŸŕäijŽå£¡çŢěåő
æĽĂäžěåŘłæIJĽå¡Şè£ŹäžŻécİåđ'ŰçŽĎåŔĆæŢřåŕŕèČ¡äijŽå¡śåŞ■åĹřçśżåŚ¡åŘ■çľžéŮt'çŽĎåĹŻåžżæŮűä¡ăa
`__prepare__()` æŰźæşŢãĂĆ

éĂŽè£ĞäĄ£çŤĺåijžåĹűåĚşéŤőå■ŮåŔĆæŢřïijŇåIJĺçśżçŽĎåĹŻåžžè£ĞçĺŃäÿ■æĹŚäžňå£ĚéąžéĂŽè£ĞåĚş

ä¡£çŤĺåĚşéŤőå■ŮåŔĆæŢřéĚ■ç¡őäÿĂäÿłåĚČçśżè£ŸåŕŕäžěèğĘä¡IJåŕźçśżåŔŸéĞŔçŽĎäÿĂçğ■æŻ£äžčæ

```
class Spam(metaclass=MyMeta):
    debug = True
    synchronize = True
    pass
```

åřĘè£ŹäžŻåśđæĂğåőŽäźĽäÿžåŔĆæŢřçŽĎåě¡åđ'ĎåIJĺäžŐåőČäžňäÿ■äijŽæśąæ§ŞçśżçŽĎåŘ■çğřçľžéŮt'
è£ŹäžŻåśđæĂğäžĚäžĚåŔłäžŐåśđäžŐçśżçŽĎåĹŻåžžéŸűæőţïijŇèĂŇäÿ■æŸŕçśżäÿ■çŽĎèŕ■åŘèæĽğèąŇéŸűa
åŘęåđ'ŰïijŇåőČäžňåIJĺ `__prepare__()` æŰźæşŢäÿ■æŸŕåŕŕäžěèćńèő£éŮőçŽĎïijŇåŻääÿžè£ŹäÿłæŰźæş

ä¡ĘæŸŕçśżåŔŸéĞŔåŔłèČ¡åIJĺåĚČçśżçŽĎ `__new__()` åŠŇ `__init__()` æŰźæşŢäÿ■åŔŕèğĄãĂĆ

9.16 *argsåŠŇ**kwargsçŽĎåijžåĹűåŔĆæŢřç■¿åŘ■

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłåĞ¡æŢřæĹŰæŰźæşŢïijŇåőČä¡£çŤĺ*argsåŠŇ**kwargsä¡IJäÿžåŔĆæŢřïijŇè£ŹæăůäĄ£å¿Ů
ä¡ĘæIJL'æŮűåĂŹä¡ăæČşæčĂæ§ěäijăéĂŠè£ŻæİěçŽĎåŔĆæŢřæŸŕäÿ■æŸŕæ§ŘäÿłäĄăæČşèęĄçŽĎçśžåđŇãĂĆ

èğčåĘşæŰźæąĹ

åŕźäżżäĄŢæűL'åŔŁåĹřæŞ■äĄIJåĞ¡æŢřèřČçŤĺç■¿åŘ■çŽĎéŮőécŸïijŇäĄăéČ¡åžŤèŕěäĄ£çŤĺ `inspect` æĺąåİŮäÿ■çŽĎç■¿åŘ■çL'źæĂğãĂĆ æĹŠäżňæIJĂäÿżèęĄåĚşæşĺäÿd'äÿłçśżïijŽ`Signature` åŠŇ `Parameter` ãĂĆäÿŃéIćæŸŕäÿĂäÿłåĹŻåżžåĞ¡æŢřåL'■éIćçŽĎäžd'äžŠä¿Ńå■ŘïijŽ

```
>>> from inspect import Signature, Parameter
>>> # Make a signature for a func(x, y=42, *, z=None)
>>> parms = [ Parameter('x', Parameter.POSITIONAL_OR_KEYWORD),
...           Parameter('y', Parameter.POSITIONAL_OR_KEYWORD,␣
 ↪default=42),
...           Parameter('z', Parameter.KEYWORD_ONLY, default=None) ]
>>> sig = Signature(parms)
>>> print(sig)
(x, y=42, *, z=None)
>>>
```

äÿĂæŮęä¡ăæIJL'äžĘäÿĂäÿłç■¿åŘ■åŕźèśąïijŇä¡ăåřšåŔŕäżěä¡£çŤĺåőČçŽĎ `bind()` æŰźæşŢå¿ĹåőźæŸ§çŽĎåřĘåőČçżŚåőŽåĹř `*args` åŠŇ `**kwargs` äÿŁåŐżãĂĆ äÿŃéIćæŸŕäÿĂäÿłçőÅå■ŢçŽĎæijŤçd'žïijŽ

```
>>> def func(*args, **kwargs):
...     bound_values = sig.bind(*args, **kwargs)
...     for name, value in bound_values.arguments.items():
...         print(name,value)
...
>>> # Try various examples
>>> func(1, 2, z=3)
x 1
y 2
z 3
>>> func(1)
x 1
>>> func(1, z=3)
x 1
z 3
>>> func(y=2, x=1)
```

```
x 1
y 2
>>> func(1, 2, 3, 4)
Traceback (most recent call last):
...
    File "/usr/local/lib/python3.3/inspect.py", line 1972, in _bind
        raise TypeError('too many positional arguments')
TypeError: too many positional arguments
>>> func(y=2)
Traceback (most recent call last):
...
    File "/usr/local/lib/python3.3/inspect.py", line 1961, in _bind
        raise TypeError(msg) from None
TypeError: 'x' parameter lacking default value
>>> func(1, y=2, x=3)
Traceback (most recent call last):
...
    File "/usr/local/lib/python3.3/inspect.py", line 1985, in _bind
        '{arg!r}'.format(arg=param.name))
TypeError: multiple values for argument 'x'
>>>
```

åŕŕäżěçIJŃåĞžæÌěïijŇéĂŽè£ĞåřEç■¿åŘ■åŠŇäijäéĂŠçŽĎåŔĆæŢřçżŚåőŽèţůæÌěïijŇåŕŕäżěåijžåĹűåĞ¡

äÿŃéÌćæŸŕäÿĂäÿłåijžåĹűåĞ¡æŢřç■¿åŘ■æŻťåĚůä¡ŞçŽĎä¿Ńå■ŘãĂĆåIJÍäżčçăĄäÿ■ïijŇæĹŚäżňåIJÍå§žç
`__init__()` æŰźæşŢïijŇ çĎűåŘŐæĹŚäżňåijžåĹűæĽĂæIJL'çŽĎå■Řçśżå£Ěéążæŕŕä¿ŻäÿĂäÿłçL'źåőŽçŽĹ

```
from inspect import Signature, Parameter

def make_sig(*names):
    parms = [Parameter(name, Parameter.POSITIONAL_OR_KEYWORD)
             for name in names]
    return Signature(parms)

class Structure:
    __signature__ = make_sig()
    def __init__(self, *args, **kwargs):
        bound_values = self.__signature__.bind(*args, **kwargs)
        for name, value in bound_values.arguments.items():
            setattr(self, name, value)

# Example use
class Stock(Structure):
    __signature__ = make_sig('name', 'shares', 'price')

class Point(Structure):
    __signature__ = make_sig('x', 'y')
```

äÿŃéÌćæŸŕä¡£çŤÍè£ŹäÿłStockçśżçŽĎçd'žä¿ŃïijŽ

```
>>> import inspect
>>> print(inspect.signature(Stock))
(name, shares, price)
>>> s1 = Stock('ACME', 100, 490.1)
>>> s2 = Stock('ACME', 100)
Traceback (most recent call last):
...
TypeError: 'price' parameter lacking default value
>>> s3 = Stock('ACME', 100, 490.1, shares=50)
Traceback (most recent call last):
...
TypeError: multiple values for argument 'shares'
>>>
```

èőĺèőž

åIJĺæĹŚäžňéIJĂèęĄæđĎåžžéĂŽçŤĺåĞ¡æŢřåžŞãĂĄçijŰåĘŹèčĚéěřåŹĺæĹŰåőđçŐřäžčçŘęçŽĎæŮűåĂŹ*args åŠŇ **kwargs çŽĎä¡£çŤĺæŸřå¿ĹæŹőéĄ■çŽĎãĂĆä¡ĘæŸřïijŇè£ŹæăůçŽĎåĞ¡æŢřæIJL'äÿĂäÿłçijžçĆźåřśæŸřå¡Şä¡ăæČşèęĄåőđçŐřèĞłåůśçŽĎåŔĆæŢřæčĂéłÑè£ŹæŮűåĂŹæĹŚäžňåŔřäžěéĂŽè£ĞäÿĂäÿłç■¿åŘ■åŕźèśąæİěçőĂåŇŰåőČãĂĆ

åIJĺæIJĂåŘŐçŽĎäÿĂäÿłæŰźæąĹåőđä¿Ńäÿ■ïijŇæĹŚäžňè£ŸåŔřäžěéĂŽè£Ğä¡£çŤĺèĞłåőŽäźL'åĚČçśżæİ

```
from inspect import Signature, Parameter

def make_sig(*names):
    parms = [Parameter(name, Parameter.POSITIONAL_OR_KEYWORD)
             for name in names]
    return Signature(parms)

class StructureMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        clsdict['__signature__'] = make_sig(*clsdict.get('_fields',
 ↪[]))
        return super().__new__(cls, clsname, bases, clsdict)

class Structure(metaclass=StructureMeta):
    _fields = []
    def __init__(self, *args, **kwargs):
        bound_values = self.__signature__.bind(*args, **kwargs)
        for name, value in bound_values.arguments.items():
            setattr(self, name, value)

# Example
class Stock(Structure):
    _fields = ['name', 'shares', 'price']

class Point(Structure):
    _fields = ['x', 'y']
```

å¡ŞæĹŚäžňèĞłåőŽäźĽç■¿åŘ■çŽĎæŮűåĂŹïijŇářĘç■¿åŘ■å■ŸåĆĺåIJĺçĽźåőŽçŽĎåšđæĂğ `__signature__` äÿ■éĂŽåÿÿæŸŕå¿ĹæIJL'çŤĺçŽĎãĂĆ è£ŹæăůçŽĎèŕĺïijŇåIJĺä¡£çŤĺ `inspect` æĺąåİŮæL'ğèąŇåĘĚçIJAçŽĎäžčçăĄářśèČ¡åŔŚçŐřç■¿åŘ■åźűářĘåőČä¡IJäÿžèřČçŤĺçžęåőŽãĂĆ

```
>>> import inspect
>>> print(inspect.signature(Stock))
(name, shares, price)
>>> print(inspect.signature(Point))
(x, y)
>>>
```

9.17 åIJĺçśżäÿŁåijžåĹűä¡£çŤĺçijŰçĺŃèğĎçžę

éŮőécŸ

ä¡ăçŽĎçĺŃåžŔåŇĚåŔńäÿĂäÿłå¿Ĺåd'ğçŽĎçśżçżğæL'£ä¡ŞçşżïijŇä¡ääÿŇæIJŻåijžåĹűæL'ğèąŇæ§ŘäžŻçij

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăæČşçŻŚæŐğçśżçŽĎåőŽäźL'ïijŇéĂŽåÿÿåŔŕäżěéĂŽè£ĞåőŽäźL'äÿĂäÿłåĚČçśżãĂĆäÿĂäÿłå§ `type` åźűéĞ■åőŽäźL'ååČçŽĎ `__new__()` æŰźæşŢ æĹŰèĂĚæŸŕ `__init__()` æŰźæşŢãĂĆæŕŤåęĆïijŽ

```
class MyMeta(type):
    def __new__(self, clsname, bases, clsdict):
        # clsname is name of class being defined
        # bases is tuple of base classes
        # clsdict is class dictionary
        return super().__new__(cls, clsname, bases, clsdict)
```

åŔęäÿĂçğ■æŸŕïijŇåőŽäźL' `__init__()` æŰźæşŢïijŽ

```
class MyMeta(type):
    def __init__(self, clsname, bases, clsdict):
        super().__init__(clsname, bases, clsdict)
        # clsname is name of class being defined
        # bases is tuple of base classes
        # clsdict is class dictionary
```

äÿžäžĘä¡£çŤĺè£ŹäÿłåĚČçśżïijŇä¡ăéĂŽåÿÿèęĄářĘåőČæŤ¿åĹřåĹřäÿĂäÿłéąűçžğçĹűçśżåőŽäźL'äÿ■ïijŇçĽ

```
class Root(metaclass=MyMeta):
    pass

class A(Root):
    pass
```

```
class B(Root):
    pass
```

åĚČçśžçŽĎäÿĂäÿłåĚşéŤőçĽźçĆźæŸŕåőČåĚĄèőÿä¡ääIJĺåőŽäźĽçŽĎæŮűåĂŹæčĂæ§ěçśžçŽĎåĘĚåőźã
__init__() æŰźæşŢäÿ■ïijŇ ä¡ääŘŕäžěå¿Ĺè¡żæİ¿çŽĎæčĂæ§ěçśžå■ŮåĚÿãĂĄçĹűçśžç■Ľ'ç■Ľ'ãĂĆåźűäÿŤï
åŽäæ■ď'ïijŇäÿĂäÿłæąĘæđűçŽĎæđĎåžžèĂĚářśèČ¡åIJĺåď'ğåđŃçŽĎçžğæĽ'£ä¡Şçşžäÿ■éĂŽè£ĞçźŹäÿĂäÿłéąű

ä¡IJäÿžäÿĂäÿłåĚůä¡ŞçŽĎåžŤçŤĺä¿Ńå■ŘïijŇäÿŃéİćåőŽäźĽäžĘäÿĂäÿłåĚČçśžïijŇåőČäijŽæŃŠçżİäżżä¡Ţ

```
class NoMixedCaseMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        for name in clsdict:
            if name.lower() != name:
                raise TypeError('Bad attribute name: ' + name)
        return super().__new__(cls, clsname, bases, clsdict)

class Root(metaclass=NoMixedCaseMeta):
    pass

class A(Root):
    def foo_bar(self): # Ok
        pass

class B(Root):
    def fooBar(self): # TypeError
        pass
```

ä¡IJäÿžæŻťéńŸçžğåŠŇåőđçŤĺçŽĎä¿Ńå■ŘïijŇäÿŃéİćæIJĽäÿĂäÿłåĚČçśžïijŇåőČçŤĺæİěæčĂæţŃéĞ■è¡¡

```
from inspect import signature
import logging

class MatchSignaturesMeta(type):

    def __init__(self, clsname, bases, clsdict):
        super().__init__(clsname, bases, clsdict)
        sup = super(self, self)
        for name, value in clsdict.items():
            if name.startswith('_') or not callable(value):
                continue
            # Get the previous definition (if any) and compare the␣
↪signatures
            prev_dfn = getattr(sup,name,None)
            if prev_dfn:
                prev_sig = signature(prev_dfn)
                val_sig = signature(value)
                if prev_sig != val_sig:
                    logging.warning('Signature mismatch in %s. %s !
↪= %s',
                                    value.__qualname__, prev_sig,␣
↪val_sig)
```

```
# Example
class Root(metaclass=MatchSignaturesMeta):
    pass

class A(Root):
    def foo(self, x, y):
        pass

    def spam(self, x, *, z):
        pass

# Class with redefined methods, but slightly different signatures
class B(A):
    def foo(self, a, b):
        pass

    def spam(self,x,z):
        pass
```

åęĆæđIJä¡ăè£Řèą Ňè£Źæőţäžčçă Ąïij Ňářšäij Žå¿ŮåĹřäÿŃéİćè£ŹæăůçŽĎè¿ŞåĞžçżŞæđIJïij Ž

```
WARNING:root:Signature mismatch in B.spam. (self, x, *, z) != (self,
↪ x, z)
WARNING:root:Signature mismatch in B.foo. (self, x, y) != (self, a,␣
↪b)
```

è£Źçğ■è■ęåŚŁä£ąæĄŕåŕźäžŐæ■ŢèŐůäÿĂäžŽå¿őåęŹçŽĎçÍŃåžŔbugæŸŕå¿ĹæIJL'çŤÍçŽĎãĂĆä¿ŃåęĆïij éĆčäźĹå¡Şå■ŘçśżæŤźåŔŸåŔĆæŢřåŘ■å■ŮçŽĎæŮűåĂŹářšäijŽèřČçŤÍåĞžéŤŹãĂĆ

èőÍèőž

åIJÍåď ğåđŃéİćåŘŚářźèśąçŽĎçÍŃåžŔäÿ■ïijŃéĂŽåÿÿářĘçśżçŽĎåőŽäźL'æŤ¿åIJÍåĚČçśżäÿ■æŐğåĹűæŸŕå åĚČçśżåŔŕäžěçŻŚæŐğçśżçŽĎåőŽäźL'ïijŃè■ęåŚŁçijŰçÍŃäžžåŚŸæ§ŘäžŻæšąæIJL'æşÍæĎŔåĹřçŽĎåŔŕèČ¡åĞ

æIJL'äžžåŔŕèČ¡äijŽèŕť'ïijŃåČŔè£ŹæăůçŽĎéŤŹèŕŕåŔŕäžěéĂŽè£ĞçÍŃåžŔåĹĘæđŘåůěåĚůæĹŰIDEåŐżå ä¡ĘæŸŕïijŃåęĆæđIJä¡ăåIJÍæđĎåžžäÿĂäÿłæąĘæđűæĹŰåĞ¡æŢřåžŞä¿ŻåĚűäžŰäžžä¡£çŤÍïijŃéĆčäźĹä¡ăæšąåĽ åŻăæ■d'ïijŃåŕźäžŐè£Źçğ■çśżåđŃçŽĎçÍŃåžŔïijŃåęĆæđIJåŘŕäžěåIJÍåĚČçśżäÿ■åĄŻæčĂæţŃæĹŰèőÿåŘŕäžěś

åIJÍåĚČçśżäÿ■éĂL'æŃl'éĞ■æŰřåőŽäźL' __new__() æŰźæşŢè£ŸæŸŕ __init__() æŰźæşŢåŘŰåEşäžŐä¡ăæČşæĂŐæăůä¡£çŤÍçżŞæđIJçśżãĂĆ __new__() æŰźæşŢåIJÍçśżåĹŻåžžäźŃåL'■èćńèřČçŤÍïijŃéĂŽåÿÿçŤÍäžŐéĂŽè£ĞæŞŘçğ■æŰźåijŔïijĹærŤåęĆéĂŽè£ĞæŤ èĂŃ __init__() æŰźæşŢæŸŕåIJÍçśżèćńåĹŻåžžäźŃåŘŐèćńèřČçŤÍïijŃå¡Şä¡ăéIJĂèęĄåőŃæŢť'æđĎåžžçśż åIJÍæIJĂåŘŐäÿĂäÿłä¿Ńå■Řäÿ■ïijŃè£ŹæŸŕå£ĚèęĄçŽĎïijŃåŽääÿžåőČä¡£çŤÍäžĘ super() åĞ¡æŢřæİěæŘIJçť'ćäźŃåL'■çŽĎåőŽäźL'ãĂĆ åőČåŔłèČ¡åIJÍçśżçŽĎåőđä¿Ńèćńå ĹŻåžžäźŃåŘŐïijŃåźűäÿŤçŻ

æIJĂåŘŐäÿĂäÿłä¿Ńå■Řè£ŸæijŤçď'žäžĘPythonçŽĎåĞ¡æŢřç■¿åŘ■åŕźèśąçŽĎä¡£çŤÍãĂĆ åőđéŹĚäÿŁïijŃåĚČçśżäijŽçőąçŘĘäÿ■ærŔäÿłäÿĂäÿłèŕČçŤÍåőŽäźL'ïijŃæŘIJçť'ćåL'■äÿĂäÿłåőŽäźL'ïijĹąęĆæđ çĎűåŘŐéĂŽè£Ğä¡£çŤÍ inspect.signature() æİěçőĂå■ŢçŽĎærŤè¿ČåőČäžňçŽĎèŕČçŤÍç■¿åŘ■ãĂĆ

æIJĂåŘŐäÿĂçĆźïijŇäžčçăĄäÿ■æIJL'äÿĂèąŇä¡£çŤĺäžĘ `super(self, self)` åźűäÿ■æŸŕæŐŠçL'ĹéŤŹèŕŕãĂĆ å¡Şä¡£çŤĺåĚČçśżçŽĎæŮűåĂŹïijŇæĹŚäžňèęĄæŮűåĽżèőŕä¡ŔäÿĂçĆźåřśæŸ `self` åőđéŹĚäÿŁæŸŕäÿĂäÿłçśżåŕźèśąãĂĆ åŽăæ■d'ïijŇè£Źæİąéŕ■åŔěåĚűåőđåřśæŸŕçŤĺæİěåŕżæL'¿ä¡■äžŐçż `self` çĹűçśżçŽĎåőŽäźL'ãĂĆ

9.18 äžěçijŰçÍŃæŰźåijŔåőŽäźL'çśż

éŮőécŸ

ä¡ăåIJĺåĘŹäÿĂæőţäžčçăĄïijŇæIJĂçżĹéIJĂèęĄåĹŻåžžäÿĂäÿłæŰřçŽĎçśżåŕźèśąãĂĆä¡äèĂČèŹŚåřĘçśżçŽ åźűäÿŤä¡£çŤĺåĞ¡æŢřæŕŤåęĆ `exec()` æİěæL'ğèąŇåőČïijŇä¡ĘæŸŕä¡äæČşåŕżæL'¿äÿĂäÿłæŻť'åŁääijŸéŽĚçŽ

èğčåĘşæŰźæąĹ

ä¡ăåŔŕäžěä¡£çŤĺåĞ¡æŢř `types.new_class()` æİěåĹİåğŃåŇŰæŰřçŽĎçśżåŕźèśąãĂĆ ä¡ăéIJĂèęĄåĄŽçŽĎåŔłæŸŕæŔŘä¿ŻçśżçŽĎåŕ■å■ŮãĂĄçĹűçśżåĚČçżĎãĂĄåĚşéŤőå■ŮåŔĆæŢřïijŇäžěåŔŁ

```
# stock.py
# Example of making a class manually from parts

# Methods
def __init__(self, name, shares, price):
    self.name = name
    self.shares = shares
    self.price = price
def cost(self):
    return self.shares * self.price

cls_dict = {
    '__init__' : __init__,
    'cost' : cost,
}

# Make a class
import types

Stock = types.new_class('Stock', (), {}, lambda ns: ns.update(cls_
↪dict))
Stock.__module__ = __name__
```

è£Źçğ■æŰźåijŔäijŽædĎåžžäÿĂäÿłæŹőéĂŽçŽĎçśżåŕźèśąïijŇåźűäÿŤæŇL'çĚğä¡ăçŽĎæIJ§æIJŻåůěä¡IJïj

```
>>> s = Stock('ACME', 50, 91.1)
>>> s
<stock.Stock object at 0x1006a9b10>
>>> s.cost()
```

```
4555.0
>>>
```

è£Źçğ■æŰźæşŢäÿ■ïijŇäÿĂäÿłæŕŤè¿ČéŽ¿çŘĘèğčçŽĎåIJřæŰźæŸŕåIJÍèřČçŤĺåőŇ types.new_class() åŕź Stock.__module__ çŽĎèţŃåĂijãĂĆ æŕŔæňąå¡ŞäÿĂäÿłçśżèćńåőŽäźĽåŘŐïijŇåőČçŽĎ __module__ åśđæĂğåŇĚåŘńåőŽäźĽåőČçŽĎæĺąåİŮåŘ■ãĂĆ è£ŹäÿłåŘ■å■ŮçŤĺäžŐçŤ§æĹŘ __repr__() æŰźæşŢçŽĎè¿ŞåĞžãĂĆåőČåŘŇæăůäź§èćńçŤĺäžŐå¿Ĺåď ŽåžŞïijŇæŕŤåęĆ pickle ãĂĆ åŽăæ■ď ïijŇäÿžäžĘèőľ ä¡ääĹŻåżžçŽĎçśżæŸŕâĂIJæ■čçąőâĂİçŽĎïijŇä¡ăéIJĂèęĄçąőä£İè£Źäÿ

åęĆæđIJä¡ăæČşåĹŻåżžçŽĎçśżéIJĂèęĄäÿĂäÿłäÿ■åŘŇçŽĎåĚČçśżïijŇåŔŕäżěéĂŽè£Ğ types.new_class() çňňäÿĽäÿłåŔĆæŢřäijăéĂŠçżŹåőČãĂĆä¿ŃåęĆïijŽ

```
>>> import abc
>>> Stock = types.new_class('Stock', (), {'metaclass': abc.ABCMeta},
...                         lambda ns: ns.update(cls_dict))
...
>>> Stock.__module__ = __name__
>>> Stock
<class '__main__.Stock'>
>>> type(Stock)
<class 'abc.ABCMeta'>
>>>
```

çňňäÿĽäÿłåŔĆæŢřè£ŸåŔŕäżěåŇĚåŘńåĚűäżŰçŽĎåĚşéŤőå■ŮåŔĆæŢřãĂĆæŕŤåęĆïijŇäÿĂäÿłçśżçŽĎåő

```
class Spam(Base, debug=True, typecheck=False):
    pass
```

éĆčäźĹåŔŕäżěåřĘåĚűç£żèŕŚæĹŘåęĆäÿŃçŽĎ new_class() èřČçŤĺå¡ćåijŔïijŽ

```
Spam = types.new_class('Spam', (Base,),
                        {'debug': True, 'typecheck': False},
                        lambda ns: ns.update(cls_dict))
```

new_class() çňňåŻŻäÿłåŔĆæŢřæIJĂçěđçğŸïijŇåőČæŸŕäÿĂäÿłçŤĺæİěæŐěåŔŮçśżåŚ¡åŘ■çľ žéŮť ç éĂŽåÿÿè£ŹæŸŕäÿĂäÿłæŹőéĂŽçŽĎå■ŮåĚÿïijŇä¡ĘæŸŕåőČåőđéŹĚäÿŁæŸŕ __prepare__() æŰźæşŢè£ŤåŻđçŽĎäżżæĎŔåŕźèśąïijŇè£Źäÿłåľ9.14årŔèĽĆåůšçżŔäžŃçż■è£ĞäžĘãĂĆ è£ŹäÿłåĞ¡æŢřéIJĂèęĄä¡£çŤĺäÿŁéİćæijŤçď žçŽĎ update() æŰźæşŢçżŹåŚ¡åŘ■çľ žéŮť åćđåŁăåĘĚåőźãĂĆ

èőĺèőž

å¿Ĺåď ŽæŮűåĂŹåęĆæđIJèČ¡æđĎéĂăæŰřçŽĎçśżåŕźèśąæŸŕå¿ĹæIJĽçŤĺçŽĎãĂĆ æIJĽäÿłå¿ĹçĘ§æĆĽçŽĎä¿Ńå■ŘæŸŕèřČçŤĺ collections.namedtuple() åĞ¡æŢřïijŇä¿ŃåęĆïijŽ

```
>>> Stock = collections.namedtuple('Stock', ['name', 'shares',
↪'price'])
>>> Stock
<class '__main__.Stock'>
>>>
```

namedtuple() ä¡£çŤĺ exec() èĂŇäÿ■æŸŕäÿŁéĺćäżŃçż■çŽĎæŁĂæIJŕãĂĆä¡ĘæŸŕïijŇäÿŃéĺćéĂŽè æĹŚäžňçŻť'æŐěåĹŽåžžäÿĂäÿłçśżïijŽ

```
import operator
import types
import sys

def named_tuple(classname, fieldnames):
    # Populate a dictionary of field property accessors
    cls_dict = { name: property(operator.itemgetter(n))
                 for n, name in enumerate(fieldnames) }

    # Make a __new__ function and add to the class dict
    def __new__(cls, *args):
        if len(args) != len(fieldnames):
            raise TypeError('Expected {} arguments'.
↪format(len(fieldnames)))
        return tuple.__new__(cls, args)

    cls_dict['__new__'] = __new__

    # Make the class
    cls = types.new_class(classname, (tuple,), {},
                          lambda ns: ns.update(cls_dict))

    # Set the module to that of the caller
    cls.__module__ = sys._getframe(1).f_globals['__name__']
    return cls
```

è£Źæőţäžčçă AçŽĎæIJĂåŘŐéČĺåĹĘä¡£çŤĺäžĘäÿĂäÿłæĽĂèřŞçŽĎ”æąĘæđűé■ŤæşŢ”ïijŇéĂŽè£ĞèřČçŤ sys._getframe() æİěèŐůåŔŰèřČçŤĺèĂĚçŽĎæĺąåİŮůåŘ■ãĂĆ åŔęåď'ŰäÿĂäÿłæąĘæđűé■ŤæşŢä¿Ńå■ŘåIJĺ2.15åŕŔèŁĆäÿ■æIJL'äžŃçż■è£ĞãĂĆ

äÿŇéİćçŽĎä¿Ńå■ŘæijŤçď'žäžĘåL'■éİćçŽĎäžčçăĄæŸŕåęĆä¡Ţåůěä¡IJçŽĎïijŽ

```
>>> Point = named_tuple('Point', ['x', 'y'])
>>> Point
<class '__main__.Point'>
>>> p = Point(4, 5)
>>> len(p)
2
>>> p.x
4
>>> p.y
5
```

```
>>> p.x = 2
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>> print('%s %s' % p)
4 5
>>>
```

è£Źéąźæ£ĂæIJŕäÿĂäÿłå¿ĹéĞ■èęĄçŽĎæŰźéİćæŸŕåőČåŕźäžŐåĚČçśżçŽĎæ■čçąőä¡£çŤĺãĂĆ ä¡ăåŘŕèČ¡åČŔéĂŽè£ĞçŻť'æŐěåðđä¿ŃåŇŰäÿĂäÿłåĚČçśżæİěçŻť'æŐěåĹŻåžžäÿĂäÿłçśżïijŽ

```
Stock = type('Stock', (), cls_dict)
```

è£Źçğ■æŰźæşŢçŽĎéŮőécŸåIJĺäžŐåőČå£¡çŢëäžĘäÿĂäžŻåĚşéŤőæ■ěéłd'ïijŇæŕŤåęĆåŕźäžŐåĚČçśżäÿ■ `__prepare__()` æŰźæşŢçŽĎèŕČçŤĺãĂĆ éĂŽè£ĞäĄ£çŤĺ `types.new_class()` ïijŇä¡ăåŘŕäžëä£İèŕĄæL'ĂæIJL'çŽĎå£ĚèęĄåĹİåğŃåŇŰæ■ěéłd'éČ¡èČ¡å¿ŮåĹřæL'ğèąŇãĂĆ æŕŤåęĆïijŇ`types.new_class()` çňňåŻŻäÿłåŔĆæŢřçŽĎåŻđèŕČåĞ¡æŢřæŐěåŔŮ `__prepare__()` æŰźæşŢè£ŤåŻđçŽĎæŸăărĎåŕźèśąãĂĆ

åęĆæđIJä¡ăäżĚäżĚåŘłæŸŕæČşæL'ğèąŇåĞĘåd'Ğæ■ěéłd'ïijŇåŘŕäžëä¡£çŤĺ `types.prepare_class()` ãĂĆä¿ŃåęĆïijŽ

```
import types
metaclass, kwargs, ns = types.prepare_class('Stock', (), {'metaclass
↪': type})
```

åőČäijŽæ§ěæL'¿åŘĹéĂĆçŽĎåĚČçśżåźűèŕČçŤĺåőČçŽĎ `__prepare__()` æŰźæşŢãĂĆ çĎűåŘŐè£ŹäÿłåĚČçśżä£İå■ŸåőČçŽĎåĚşéŤőå■ŮåŘĆæŢřïijŇåĞĘåd'ĞåŚ¡åŘ■çl'žéŮt'åŘŐèćńè

æŻt'åd'Žä£qæĄf, èŕůåŘĆèĂČ PEP 3115 , äžěåŘŁ Python documentation .

9.19 åIJĺåőŽäźL'çŽĎæŮűåĂŹåĹİåğŃåŇŰçśżçŽĎæĹŘåŚŸ

éŮőécŸ

ä¡ăæČşåIJĺçśżèćńåőŽäźL'çŽĎæŮűåĂŹåŕśåĹİåğŃåŇŰäÿĂéČĺåĹĘçśżçŽĎæĹŘåŚŸïijŇèĂŇäÿ■æŸŕèęĄç

èğčåĘşæŰźæąĹ

åIJĺçśżåőŽäźL'æŮűåŕśæL'ğèąŇåĹİåğŃåŇŰæĹŰèő¿ç¡őæŞ■ä¡IJæŸŕåĚČçśżçŽĎäÿĂäÿłåĚÿåđŃåžŤçŤĺåIJ è£ŹæŮűåĂŹä¡ăåŘŕäžëæL'ğèąŇäÿĂäžŻéĆİåd'ŰçŽĎæŞ■ä¡IJãĂĆ

äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijŇåĹl'çŤĺè£ŹäÿłæĂİèůŕæİěåĹŻåžžçśżäijijäžŐ `collections` æĺąåİŮäÿ■çŽĎåŚ¡åŘ■åĚČçżĎçŽĎçśżïijŽ

```
import operator
```

```
class StructTupleMeta(type):
    def __init__(cls, *args, **kwargs):
        super().__init__(*args, **kwargs)
        for n, name in enumerate(cls._fields):
            setattr(cls, name, property(operator.itemgetter(n)))

class StructTuple(tuple, metaclass=StructTupleMeta):
    _fields = []
    def __new__(cls, *args):
        if len(args) != len(cls._fields):
            raise ValueError('{} arguments required'.format(len(cls.
↪_fields)))
        return super().__new__(cls,args)
```

è£Źæőţäžčçă Ąåŕŕäžěç ŤÍæİěåőŽäźĽçőĂå■ŢçŽĎå§žäžŐåĚČçżĎçŽĎæŢřæ■őçżŞæđĎïijŇåęĆäÿŃæĽ Ăç

```
class Stock(StructTuple):
    _fields = ['name', 'shares', 'price']

class Point(StructTuple):
    _fields = ['x', 'y']
```

äÿŃéİćæijŤçd' žåőČåęĆä¡Ţåůëä¡IJïijŽ

```
>>> s = Stock('ACME', 50, 91.1)
>>> s
('ACME', 50, 91.1)
>>> s[0]
'ACME'
>>> s.name
'ACME'
>>> s.shares * s.price
4555.0
>>> s.shares = 23
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>>
```

èőÍèőž

è£ŹäÿĂåřŔèŁĆäÿ■ïijŇçśż `StructTupleMeta` èŐůåŔŰåĹřçśžåśđæĂğ `_fields` äÿ■çŽĎåśđæĂğåŘ■å■ŮåĹŮèąÍïijŇ çĎűåŘŐåřĘåőČäžňè¡ňæ■ćæĹŘçŻÿåžŢçŽĎåŔŕèő£éŮőçĽźåőŽåĚČçżĎæ `operator.itemgetter()` åĹŻåžžäÿĂäÿłèő£éŮőåŹĺåĞ¡æŢřïijŇ çĎűåŘŐ `property()` åĞ¡æŢřårĘåĚűè¡ňæ■ćæĹŘäÿĂäÿłåśđæĂğãĂĆ

æIJňèŁĆæIJĂéŽ¿æĞĆçŽĎéČĺåĹĘæŸŕç§ěéĄŞäÿ■åŘŇçŽĎåĹİåğŃåŇŰæ■ěéłď'æŸŕäżĂäźĹæŮűåĂŹåŔŞ `StructTupleMeta` äÿ■çŽĎ `__init__()` æŰźæşŢåŔłåIJĺæŕŔäÿłçśżèćńåőŽäźĽ'æŮűèćněŕČçŤĺäÿĂæňąãĂ `cls` åŔĆæŢřårśæŸŕéĆčäÿłèćńåőŽäźĽçŽĎçśżãĂĆåőđéŹĚäÿŁïijŇäÿŁè£ŕäžčçăĄä¡£çŤĺäžĘ

_fields çśżåŔŸéĞŔæİěä£Ìå■ŸæŰřçŽĎèćńåőŽäźĽçŽĎçśżïijŇ
çĎűåŘŐçżŹåőČåĘ■æůżåŁääÿĂçĆźæŰřçŽĎäÿIJèě£ãĂĆ

StructTuple çśżä¡IJäÿžäÿĂäÿłæŹőéĂŽçŽĎå§žçśżïijŇä¿ŻåĚűäžŰä¡£çŤĺèĂĚæİěçżğæĽ£ãĂĆ è£Źäÿłçśżäÿ■çŽĎ __new__() æŰźæşŢçŤĺæİěæđĎéĂäæŰřçŽĎåőđä¿ŃãĂĆ è£ŹéĞŇä¡£çŤĺ __new__() åźűäÿ■æŸŕå¿Ĺåÿÿè§ĄïijŇäÿżèęĄæŸŕåŽääÿžæĹŚäžňèęĄä£őæŤźåĚČçżĎçŽĎèřČçŤĺç■¿åŘ■ïijĺ ä¡£å¿ŮæĹŚäžňåŔŕäžěåČŔæŹőéĂŽçŽĎåőđä¿ŃèřČçŤĺéĆčæůåĹŻåżžåőđä¿ŃãĂĆåřśåČŔäÿŃéİćè£ŹæůüijŽ

```
s = Stock('ACME', 50, 91.1) # OK
s = Stock(('ACME', 50, 91.1)) # Error
```

èů§ __init__() äÿ■åŘŇçŽĎæŸŕïijŇ__new__() æŰźæşŢåIJĺåőđä¿ŃèćńåĹŻåżžäźŃåĽ■èćńèğęåŔŚ çŤśäžŐåĚČçżĎæŸŕäÿ■åŔŕä£őæŤźçŽĎïijŇæĽĂäžěäÿĂæŮęåőČäžňèćńåĹŻåżžäžĘåřśäÿ■åŔŕèČ¡åŕźåőČåĄŽä __init__() äijŽåIJĺåőđä¿ŃåĹŻåżžçŽĎæIJĂåŘŐèćńèğęåŔŚïijŇ è£ŹæůçŽĎèŕİæĹŚäžňåřśåŔŕäžěåĄŽæĹŚäžňæČşåĄŽçŽĎäžĘãĂĆè£Źäź§æŸŕäÿžäžĂäźĹ __new__() æŰźæşŢåůšçżŔèćńåőŽäźĽäžĘãĂĆ

år¡çőąæIJňèŁĆå¿Ĺç§■ïijŇè£ŸæŸŕéIJĂèęĄä¡ãèČ¡äžŤçżĘçăŤèŕżïijŇæůśåĚěæĂİèĂČPythonçśżæŸŕåęĆä è£ŸæIJĽåŕśæŸŕåĚČçśżåŠŇçśżçŽĎåŘĎäÿłäÿ■åŘŇçŽĎæŰźæşŢçľűçń§åIJĺäžĂäźĹæŮűåĂŹèćńèŕČçŤĺãĂĆ

PEP 422 æŔŘä¿ŻäžĘäÿĂäÿłèğčåĘşæIJňèŁĆéŮőécŸçŽĎåŔęåď ŰäÿĂçğ■æŰźæşŢãĂĆ ä¡ĘæŸŕïijŇæĹłæ■ćåĽřæĹŚåĘŹè£ŹæIJňäźęçŽĎæŮűåĂŹïijŇåőČè£ŸæšąèćńéĞĞçžşåŠŇæŐěåŔŮãĂĆ år¡çőąåęĆæ■ď¨ïijŇåęĆæđIJä¡ää¡£çŤĺçŽĎæŸŕPython 3.3æĹŰæŽť énŸçŽĎçĽĹæIJňïijŇéĆčäźĹè£ŸæŸŕåĂijå

9.20 åĹľçŤĺåĞ¡æŢřæşĺèğčåőđçŐřæŰźæşŢéĞ■è¡¡

éŮőécŸ

ä¡ääůšçżŔå■ęè£ĞæĂŐæăůä¡£çŤĺåĞ¡æŢřåŔĆæŢřæşĺèğčïijŇéĆčäźĹä¡ääŔŕèČ¡äijŽæČşåĹľçŤĺåőČæİěåőđ ä¡ĘæŸŕä¡ääÿ■çąőåőŽåžŤèŕěæĂŐæăůåŐżåőđçŐřïijĹæĹŰèĂĚåĹřåžŢèąŇå¿ŮéĂŽäÿ■ïijĽãĂĆ

èğčåĘşæŰźæąĹ

æIJňåřŔèŁĆçŽĎæŁĂæIJŕæŸŕå§žäžŐäÿĂäÿłçőĂå■ŢçŽĎæŁĂæIJŕïijŇéĆčåřśæŸŕPythonåĚĄèőÿåŔĆæŢ

```
class Spam:
    def bar(self, x:int, y:int):
        print('Bar 1:', x, y)

    def bar(self, s:str, n:int = 0):
        print('Bar 2:', s, n)

s = Spam()
s.bar(2, 3) # Prints Bar 1: 2 3
s.bar('hello') # Prints Bar 2: hello 0
```

äÿŃéİćæŸŕæĹŚäžňçňňäÿĂæ■ęçŽĎåřİèŕŢïijŇä¡£çŤĺåĹřäžĘäÿĂäÿłåĚČçśżåŠŇæŔŘè£řåŹĺïijŽ

```
# multiple.py
import inspect
import types

class MultiMethod:
    '''
    Represents a single multimethod.
    '''
    def __init__(self, name):
        self._methods = {}
        self.__name__ = name

    def register(self, meth):
        '''
        Register a new method as a multimethod
        '''
        sig = inspect.signature(meth)

        # Build a type signature from the method's annotations
        types = []
        for name, parm in sig.parameters.items():
            if name == 'self':
                continue
            if parm.annotation is inspect.Parameter.empty:
                raise TypeError(
                    'Argument {} must be annotated with a type'.
↪format(name)
                )
            if not isinstance(parm.annotation, type):
                raise TypeError(
                    'Argument {} annotation must be a type'.
↪format(name)
                )
            if parm.default is not inspect.Parameter.empty:
                self._methods[tuple(types)] = meth
            types.append(parm.annotation)

        self._methods[tuple(types)] = meth

    def __call__(self, *args):
        '''
        Call a method based on type signature of the arguments
        '''
        types = tuple(type(arg) for arg in args[1:])
        meth = self._methods.get(types, None)
        if meth:
            return meth(*args)
        else:
            raise TypeError('No matching method for types {}'.
↪format(types))
```

```
    def __get__(self, instance, cls):
        '''
        Descriptor method needed to make calls work in a class
        '''
        if instance is not None:
            return types.MethodType(self, instance)
        else:
            return self

class MultiDict(dict):
    '''
    Special dictionary to build multimethods in a metaclass
    '''
    def __setitem__(self, key, value):
        if key in self:
            # If key already exists, it must be a multimethod or␣
↪callable
            current_value = self[key]
            if isinstance(current_value, MultiMethod):
                current_value.register(value)
            else:
                mvalue = MultiMethod(key)
                mvalue.register(current_value)
                mvalue.register(value)
                super().__setitem__(key, mvalue)
        else:
            super().__setitem__(key, value)

class MultipleMeta(type):
    '''
    Metaclass that allows multiple dispatch of methods
    '''
    def __new__(cls, clsname, bases, clsdict):
        return type.__new__(cls, clsname, bases, dict(clsdict))

    @classmethod
    def __prepare__(cls, clsname, bases):
        return MultiDict()
```

äÿžäžĘä¡£çŤĺè£ŹäÿłçśżïijŇä¡ååŔŕäžěåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
class Spam(metaclass=MultipleMeta):
    def bar(self, x:int, y:int):
        print('Bar 1:', x, y)

    def bar(self, s:str, n:int = 0):
        print('Bar 2:', s, n)

# Example: overloaded __init__
```

```
import time

class Date(metaclass=MultipleMeta):
    def __init__(self, year: int, month:int, day:int):
        self.year = year
        self.month = month
        self.day = day

    def __init__(self):
        t = time.localtime()
        self.__init__(t.tm_year, t.tm_mon, t.tm_mday)
```

äÿŃéÍćæŸŕäÿĂäÿłäžďäžŠçďža¿ŃæÍěéłŃèŕĄåőČèČ¡æ■čçąőçŽĎåůěä¡IJïijŽ

```
>>> s = Spam()
>>> s.bar(2, 3)
Bar 1: 2 3
>>> s.bar('hello')
Bar 2: hello 0
>>> s.bar('hello', 5)
Bar 2: hello 5
>>> s.bar(2, 'hello')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "multiple.py", line 42, in __call__
        raise TypeError('No matching method for types {}'.
 ↪format(types))
TypeError: No matching method for types (<class 'int'>, <class 'str
 ↪'>)
>>> # Overloaded __init__
>>> d = Date(2012, 12, 21)
>>> # Get today's date
>>> e = Date()
>>> e.year
2012
>>> e.month
12
>>> e.day
3
>>>
```

èőÍèőž

åÍęçŹ¡æÍěèőšïijÑçŽÿåŕźäžŐéĂŽåÿÿçŽĎäžčçăĄèĂŇåůšæIJňèŁĆä¡£çŤĺåŁřäžĘå¿Ĺåď'ŽçŽĎé■ŤæşŢäžčçă
ä¡ĘæŸŕïijŇåőČå■t'èČ¡èől'æĹŚäžňæůšåĚěçŘĘèğčåĚČçśżåŠŇæŔŔè£řåŹĺçŽĎåžŢåśĆåůěä¡IJåŐ§çŘĘïijŇ
åźűèČ¡åŁäæůšåŕźè£ŹäžŽæęĆå£tçŽĎå■řèśąãĂĆåŽäæ■d'ïijŇåřśçőŮä¡ääźűäÿ■äijŽçńŇå■şåŐżäžŤçŤĺæIJňèŁĆ
åőČçŽĎäÿĂäžŽäžŢåśĆæĂÍæČşå■t'äijŽå¡šåŞ■åĹřåĚűåőČæűĽ'åŔŁåĹřåĚČçśżãĂĄæŔŔè£řåŹĺåŠŇåĞ¡æŢřæş

æIJňèŁĆçŽĎåőđçŐřäÿ■çŽĎäÿżèęĄæĂÍèůřåĚűåőđæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆMutipleMeta

åĚČçśżä¡£çŤĺåőČçŽĐ __prepare__() æŰźæşŢ æİěæŔŘä¿Žäÿ Ăäÿłä¡ĲäÿžMultiDict åőđä¿ŃçŽĐèĞłåőŽäźĽå■ŮåĚÿãĂĆè£Źäÿłèů§æŹőéĂŽå■ŮåĚÿäÿ■äÿ Ăæăůç ŽĐæŸŕïijŇ MultiDict äijŽåIJİåĚÇçť ăèćńèő¿ç¡őçŽĐæŮűåĂŹæčĂæ§ěæŸŕåŘęåůšçžŔå■ŸåIJíïijŇåęĆæđIJå■ŸåIJíçŽĐ MultiMethod åőđä¿Ńäÿ■åŘĹåźűãĂĆ

MultiMethod åőđä¿ŃéĂŽè£ĞæđĐåżžäżŐçśżåđŃç■¿åŘ■åĽřåĞ¡æŢřçŽĐæŸăåřĐæİěæŤűéŽĘæŰźæşŢåIJİè£ŹäÿłæđĐåżžè£ĞçĺŃäÿ■ïijŇåĞ¡æŢřæşĺèğčèćńçŤĺæİěæŤűéŽĘç¿åŘ■çĐűåŘŐæđĐåżžè£Źäÿłæ èŁŹäÿłè£ĞçĺŃåIJĺ MultiMethod.register() æŰźæşŢäÿ■åőđçŐřãĂĆ è£Źçğ■æŸăåřĐçŽĐäÿ Ăäÿłå ĚşéŤőçĽźçĆźæŸŕåŕźäžŐåď ŽäÿłæŰźæşŢïijŇæĽ ĂæIJĽåŘĆæŢřçśżåđŃéČ¡å£Ěé

äÿžäžĘèőľ MultiMethod åőđä¿ŃæĺąæŃ§äÿĂäÿłèřČçŤĺïijŇåőČçŽĐ __call__() æŰźæşŢèćńåőđçŐřäžĘãĂĆ è£Źäÿłæ ŰźæşŢäžŐæĽ ĂæIJĽæŐŠéŹď slef çŽĐåŘĆæŢřäÿ■æđĐåżžäÿĂäÿłçśżåđŃåĚČçżĐïijŇåIJĺåĘĚéČÍmapäÿ■æ§ěæĽ¿è£Źäÿłæ ŰźæşŢïijŇ çĐűåŘŐèřČçŤĺçŻÿåžŤçŽĐæŰźæşŢãĂĆäÿžäžĘèČ¡èőľ MultiMethod åőđä¿ŃåIJĺçśżåőŽäźĽæŮűæ■čçąőæŞ■ä¡ĲïijŇ __get__() æŸŕå£Ěéążå¿Ůåőđ çŐřçŽĐãĂĆ åőČèćńçŤĺæİěæđĐåżžæ■čçąőçŽĐçżŠåőŽæŰźæşŢãĂĆæŕŤåęĆïijŽ

```
>>> b = s.bar
>>> b
<bound method Spam.bar of <__main__.Spam object at 0x1006a46d0>>
>>> b.__self__
<__main__.Spam object at 0x1006a46d0>
>>> b.__func__
<__main__.MultiMethod object at 0x1006a4d50>
>>> b(2, 3)
Bar 1: 2 3
>>> b('hello')
Bar 2: hello 0
>>>
```

äÿ■è£ĞæIJňèŁĆçŽĐåőđçŐřè£ŸæIJĽäÿĂäžŻéŹŘåĹűïijŇåĚűäÿ■äÿĂäÿłæŸŕåőČäÿ■èČ¡ä¡£çŤĺåĚşéŤőå■

```
>>> s.bar(x=2, y=3)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: __call__() got an unexpected keyword argument 'y'

>>> s.bar(s='hello')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: __call__() got an unexpected keyword argument 's'
>>>
```

äź§èőÿæIJĽåĚűäżŰçŽĐæŰźæşŢèČ¡æůżåŁăè£Źçğ■æŤŕæŇĄïijŇä¡ĘæŸŕåőČéIJĂèęĄäÿĂäÿłåőŇåĚĺäÿ■ éŮőéćŸåIJĺäžŐåĚşéŤőå■ŮåŘĆæŢřçŽĐåĞžçŐřæŸŕæšąæIJĽéąžåžŔçŽĐãĂĆå¡Şåőčéů§ä¡■ç¡őåŘĆæŢřæůůåIJ éĆčä¡ăçŽĐåŘĆæŢřåřśäijŽåŘŸå¿ŮæŕŤè¿ČæůůäźśäžĘïijŇè£ŹæŮűåĂŹä¡ăäÿ■å¿Ůäÿ■åIJĺ __call__() æŰźæşŢäÿ■åĚĹåŐżåĄŽäÿłæŐŠåžŔãĂĆ

åŘŇæăůåřźäžŐçżğæĽ£äź§æŸŕæIJĽéŹŘåĹűçŽĐïijŇä¿ŃåęĆïijŇçśżäijijäÿŃéİćè£Źçğ■äżčçăĄåřśäÿ■èČ¡

```
class A:
    pass
```

```
class B(A):
    pass

class C:
    pass

class Spam(metaclass=MultipleMeta):
    def foo(self, x:A):
        print('Foo 1:', x)

    def foo(self, x:C):
        print('Foo 2:', x)
```

åŐ§åŻäæŸŕåŻääÿž x : A æşÍèğčäÿ■èČ¡æĹŘåŁ§åŇźéĚ■å■Řçśžåőđä¿ŃïijĹæŕŤåęĆBçŽĎåőđä¿ŃïijĽïijŇå

```
>>> s = Spam()
>>> a = A()
>>> s.foo(a)
Foo 1: <__main__.A object at 0x1006a5310>
>>> c = C()
>>> s.foo(c)
Foo 2: <__main__.C object at 0x1007a1910>
>>> b = B()
>>> s.foo(b)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "multiple.py", line 44, in __call__
        raise TypeError('No matching method for types {}'.
↪format(types))
TypeError: No matching method for types (<class '__main__.B'>,)
>>>
```

ä¡IJäÿžä¡£çŤĺåĚČçśžåŠŇæşÍèğčçŽĎäÿĂçğ■æŻ£äżčæŰźæąĹïijŇåŕŕäżěéĂŽè£ĞæŔŔè£ŕåŹĺæİěåőđçŐřç

```
import types

class multimethod:
    def __init__(self, func):
        self._methods = {}
        self.__name__ = func.__name__
        self._default = func

    def match(self, *types):
        def register(func):
            ndefaults = len(func.__defaults__) if func.__defaults__␣
↪else 0
            for n in range(ndefaults+1):
                self._methods[types[:len(types) - n]] = func
            return self
        return register
```

```
    def __call__(self, *args):
        types = tuple(type(arg) for arg in args[1:])
        meth = self._methods.get(types, None)
        if meth:
            return meth(*args)
        else:
            return self._default(*args)

    def __get__(self, instance, cls):
        if instance is not None:
            return types.MethodType(self, instance)
        else:
            return self
```

äÿžäžĘä¡£çŤĺæŔŕè£řåŹĺçĽĹæIJňïijŇä¡äéIJĂèęĄåČŕäÿŃéİćè£ŹæăůåĘŹïijŽ

```
class Spam:
    @multimethod
    def bar(self, *args):
        # Default method called if no match
        raise TypeError('No matching method for bar')

    @bar.match(int, int)
    def bar(self, x, y):
        print('Bar 1:', x, y)

    @bar.match(str, int)
    def bar(self, s, n = 0):
        print('Bar 2:', s, n)
```

æŔŕè£řåŹĺæŰźæąĹåŘŇæăůäź§æIJL'åL'■éİćæŔŕåĹřçŽĎéŹŕåĹűïijĹäÿ■æŤŕæŇĄåĚşéŤőå■ŮåŔĆæŢřåŠ

æL'ĂæIJL'äžŃçL'ľéČ¡æŸŕåźşç■L'çŽĎïijŇæIJL'åě¡æIJL'åIŕïijŇäź§èőÿæIJĂåě¡çŽĎåŁđæşŢåřśæŸŕåIJĺæŹ
äÿ■è£ĞæIJL'äžŻçL'źæőŁæČĚåĘţäÿŃè£ŸæŸŕæIJL'æĎŔäźL'çŽĎïijŇærŤåęĆå§žäžŐæĺąåijŔåŇźéĚ■çŽĎæŰźä
äÿ¿äÿłä¿Ňå■ŘïijŇ8.21åřŔèŁĆäÿ■çŽĎèő£éŮőèĂĚæĺąåijŔåŕŕäžëä£őæŤźäÿžäÿĂäÿłä¡£çŤĺæŰźæşŢéĞ■è¡¡çŽĺ
ä¡ĘæŸŕïijŇéŹđ'äžĘè£Źäÿłäžëåđ'ŰïijŇéĂŽåÿÿäÿ■åžŤèŕëä¡£çŤĺæŰźæşŢéĞ■è¡¡ïijĹåřśçőĂå■ŢçŽĎä¡£çŤĺäÿ■å

åIJĺPythonçđ'¿åŇžåŕźäžŐåőđçŐřæŰźæşŢéĞ■è¡¡çŽĎèőĺèőžåůšçżŔçŤśæİèåůšäźĚãĂĆ
åŕźäžŐåijŢåŘŚè£ŹäÿłäžL'èőžçŽĎåŐ§åŻäïijŇåŕŕäžëåŔĆèĂČäÿŃGuido van RossumçŽĎè£ŹçŕĞå■Žåőćïijž Five-Minute Multimethods in Python

9.21 éĄ£åĚ■éĞ■åđ'■çŽĎåśđæĂğæŰźæşŢ

éŮőécŸ

ä¡ăåIJĺçśżäÿ■éIJĂèęĄéĞ■åđ'■çŽĎåőŽäźL'äÿĂäžŻæL'ğèąŇçŻÿåŘŇéĂżè¿ŚçŽĎåśđæĂğæŰźæşŢïijŇærŤ

èğčåĘşæŰźæąĹ

èĂČèŹŚäÿŃäÿĂäÿłçőĂå■ŢçŽĎçśżïijŇåőČçŽĎåšđæĂğçŤšåšđæĂğæŰźæşŢåŇĚèčĚïijŽ

```
class Person:
    def __init__(self, name ,age):
        self.name = name
        self.age = age

    @property
    def name(self):
        return self._name

    @name.setter
    def name(self, value):
        if not isinstance(value, str):
            raise TypeError('name must be a string')
        self._name = value

    @property
    def age(self):
        return self._age

    @age.setter
    def age(self, value):
        if not isinstance(value, int):
            raise TypeError('age must be an int')
        self._age = value
```

åŔŕäžěçIJŃåĹřïijŇäÿžäžĘåőđçŐřåšđæĂğåĂijçŽĎçśżåđŃæčĂæ§ěæĹŚäžňåĘŹäžĘå¿Ĺåď'ŽçŽĎéĞ■åď'■ä åŔłèęĄä¡ääžěåŘŐçIJŃåĹřçśżäijijè£ŹæăůçŽĎäžčçăĄïijŇä¡ăéČ¡åžŤèŕěæČşåŁđæşŢåŐżçőĂåŇŰåőČãĂĆ äÿĂäÿłåŔŕèąŇçŽĎæŰźæşŢæŸŕåĹŻåžžäÿĂäÿłåĞ¡æŢřçŤĺæİěåőŽäźĽ'åšđæĂğåźűè£ŤåŽđåőČãĂĆä¿ŃåęĆïijŽ

```
def typed_property(name, expected_type):
    storage_name = '_' + name

    @property
    def prop(self):
        return getattr(self, storage_name)

    @prop.setter
    def prop(self, value):
        if not isinstance(value, expected_type):
            raise TypeError('{} must be a {}'.format(name, expected_
↪type))
        setattr(self, storage_name, value)

    return prop

# Example use
class Person:
```

```
    name = typed_property('name', str)
    age = typed_property('age', int)

    def __init__(self, name, age):
        self.name = name
        self.age = age
```

èőÍèőž

æIJňèŁĆæĹŚäžňæijŤçď žåĘĚéČÍåĞ¡æŢřæĹŰèĂĚéŮ■åŇĚçŽĎäÿĂäÿłéĞ■èęĄçL'źæĂğïijŇåőČäžňå¿ĹåC
`typed_property()` çIJŃäÿŁåŐżæIJL'çĆźéŽ¿çŘĘèğčïijŇåĚűåőđåőČæL'ĂåĄŽçŽĎäžĚäžĚářśæŸŕäÿžä¡ăç
åŻăæ■ď'ïijŇå¡ŞåIJÍäÿĂäÿłçśžäÿ■ä¡£çŤÍåőČçŽĎæŮűåĂŹïijŇæŢĹæđIJèů§ářĘåőČéĞŇéİćçŽĎäžčçăĄæŤ¿åĹř
ář¡çőąåśđæĂğçŽĎ `getter` åŠŇ `setter` æŰźæşŢèő£éŮőäžĘæIJňåIJřåŔŸéĞŔåęĆ `name` ,
`expected_type` äžěåŔŁ `storate_name` ïijŇè£ŹäÿłåżĹæ■čåÿÿïijŇè£ŹäžŻåŔŸéĞŔçŽĎåĂijäijŽä£İå■Ÿ

æĹŚäžňè£Ÿåŕŕäžěä¡£çŤÍ `functools.partial()` æİěçÍ■çÍ■æŤźåŔŸäÿŃè£ŹäÿłäżŃå■ŘïijŇå¿ĹæIJI

```
from functools import partial

String = partial(typed_property, expected_type=str)
Integer = partial(typed_property, expected_type=int)

# Example:
class Person:
    name = String('name')
    age = Integer('age')

    def __init__(self, name, age):
        self.name = name
        self.age = age
```

åĚűåőđä¡ăåŕŕäžěåŔŚçŐřïijŇè£ŹéĞŇçŽĎäžčçăĄèů§8.13åřŔèŁĆäÿ■çŽĎçśžåđŃçşžçż§æŔŔè£řåŹĺäžčçă

9.22 åőŽäźĽäÿŁäÿŃæŰĞçőąçŘĘåŹĺçŽĎçőĂå■ŢæŰźæşŢ

éŮőécŸ

ä¡ăæČşèĞłåůšåŐżåőđçŐřäÿĂäÿłæŰřçŽĎäÿŁäÿŃæŰĞçőąçŘĘåŹĺïijŇäžěä¿£ä¡£çŤÍwithèŕ■åŘěãĂĆ

èğčåĘşæŰźæąĹ

åőđçŐřäÿĂäÿłæŰřçŽĎäÿŁäÿŃæŰĞçőąçŘĘåŹĺçŽĎæIJĂçőĂå■ŢçŽĎæŰźæşŢåřśæŸŕä¡£çŤÍ
`contexlib` æĺąåİŮäÿ■çŽĎ `@contextmanager` èčĚéěřåŹĺãĂĆ
äÿŃéİćæŸŕäÿĂäÿłåőđçŐřäžĘäžčçăĄåİŮèőąæŮűåŁ§èČ¡çŽĎäÿŁäÿŃæŰĞçőąçŘĘåŹĺä¿Ńå■ŘïijŽ

```
import time
from contextlib import contextmanager

@contextmanager
def timethis(label):
    start = time.time()
    try:
        yield
    finally:
        end = time.time()
        print('{}: {}'.format(label, end - start))

# Example use
with timethis('counting'):
    n = 10000000
    while n > 0:
        n -= 1
```

åIJÍåĞ¡æŢř `timethis()` äÿ■ïijŇ`yield` äźŃåĽ'■çŽĎäžčçăĄäijŽåIJÍäÿŁäÿŃæŰĞçőąçŘĘåŹÍäÿ■ä¡IJäÿ `__enter__()` æŰźæşŢæĽ'ğèąŇïijŇ æĽ'ĂæIJĽ'åIJÍ `yield` äźŃåŘŐçŽĎäžčçăĄäijŽä¡IJäÿž `__exit__()` æŰźæşŢæĽ'ğèąŇãĂĆ åęĆæđIJåĞžçŐřäžĘåijĆåÿÿïijŇåijĆåÿÿäijŽåIJÍyield-èŕ■åŘéĆčéĞŇæŁŻåĞžãĂĆ

äÿŃéÍćæŸŕäÿĂäÿłæŻť'åŁăéńŸçžğäÿĂçĆźçŽĎäÿŁäÿŃæŰĞçőąçŘĘåŹÍïijŇåőđçŐřäžĘåĹŮèąĺåŕźèśąäÿŁç

```
@contextmanager
def list_transaction(orig_list):
    working = list(orig_list)
    yield working
    orig_list[:] = working
```

è£ŹæőţäžčçăĄçŽĎä¡IJçŤÍæŸŕäżżä¡ŢåŕźåĹŮèąĺçŽĎä£őæŤźåŘłæIJĽ'å¡ŞæĽ'ĂæIJĽ'äžčçăĄè£ŘèąŇåőŇæ
äÿŃéÍćæĹŚäżñæÍěæijŤçd'žäÿĂäÿŃïijŽ

```
>>> items = [1, 2, 3]
>>> with list_transaction(items) as working:
...     working.append(4)
...     working.append(5)
...
>>> items
[1, 2, 3, 4, 5]
>>> with list_transaction(items) as working:
...     working.append(6)
...     working.append(7)
...     raise RuntimeError('oops')
...
Traceback (most recent call last):
    File "<stdin>", line 4, in <module>
RuntimeError: oops
>>> items
[1, 2, 3, 4, 5]
```

```
>>>
```

èőĺèőž

éĂŽåÿÿæČĚåĘţäÿŃĭijŇåęĆæđIJèęĄåĘŹäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺĭijŇä¡ăéIJĂèęĄåőŽäźĽäÿĂäÿłç
__enter__() åŠŇäÿĂäÿł __exit__() æŰźæşŢĭijŇåęĆäÿŃæĽĂçd'žĭijŽ

```
import time

class timethis:
    def __init__(self, label):
        self.label = label

    def __enter__(self):
        self.start = time.time()

    def __exit__(self, exc_ty, exc_val, exc_tb):
        end = time.time()
        print('{}: {}'.format(self.label, end - self.start))
```

åř¡çőąè£Źäÿłäź§äÿ■éŽ¿åĘŹĭijŇä¡ĘæŸŕçŻÿæŕŤè¿ČåĘŹäÿĂäÿłçőĂå■ŢçŽĎä¡£çŤĺ
@contextmanager æşĺèğčçŽĎåĞ¡æŢřèĂŇèĺĂè£ŸæŸŕçĺ■æŸ¿äźŔåŠşãĂĆ

@contextmanager åžŤèŕëäżĚäżĚçŤĺæİěåĘŹèĞłåŇĚåŘńçŽĎäÿŁäÿŃæŰĞçőąçŘĘåĞ¡æŢřãĂĆ
åęĆæđIJä¡ăæIJL'äÿĂäžŻåŕźèšą(æŕŤåęĆäÿĂäÿłæŰĞäžűãĂĄç¡ŚçżIJè£đæŐěæĹŰéŤĄ)ïijŇéIJĂèęĄæŤŕæŇĄ
with èŕ■åŘëïijŇéĆčäźĹä¡ăářšéIJĂèęĄå■ŢçŃňåőđçŐř __enter__() æŰźæşŢåŠŇ
__exit__() æŰźæşŢãĂĆ

9.23 åIJĺåšĂéČĺåŔŸéĞŔå§§äÿ■æĽğèąŇäżčçăĄ

éŮőécŸ

ä¡ăæČşåIJĺä¡£çŤĺèŇČåŻť'åĘĚæĽğèąŇæ§Řäÿłäżčçă AçĽĞæőţïijŇåźűäÿŤåÿŇæIJŻåIJĺæĽğèąŇåŘŐæĽĂ

èğčåĘşæŰźæąĹ

äÿžäžĘçŘĘèğčè£ŹäÿłéŮőécŸïijŇåĚĹèŕŢèŕŢäÿĂäÿłçőĂå■ŢåIJżæŹŕãĂĆéęŰåĚĹïijŇåIJĺåĚĺåśĂåŚ¡åŘ■ç

```
>>> a = 13
>>> exec('b = a + 1')
>>> print(b)
14
>>>
```

çĎűåŘŐïijŇåĘ■åIJÍäÿĂäÿłåĞ¡æŢřäÿ■æĽ§èąŇåŘŇæăůçŽĎäżčçăĄïijŽ

```
>>> def test():
...     a = 13
...     exec('b = a + 1')
...     print(b)
...
>>> test()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "<stdin>", line 4, in test
NameError: global name 'b' is not defined
>>>
```

åŔŕäżěçIJŃåĞžïijŇæIJĂåŘŐæŁŻåĞžäžĘäÿĂäÿłNameErroråijĆåÿÿïijŇåřśèů§åIJÍ exec() èŕ■åŔěäżŐæšąæĽ§èąŇè£ĞäÿĂæăůãĂĆ èęĄæŸŕä¡ăæČşåIJÍåŘŐéİćçŽĎèőąçőŮäÿ■ä¡£çŤÍåĹř exec() æĽ§èąŇçżŞæđIJçŽĎèŕİåřśäijŽæIJĽéŮőéćŸäžĘãĂĆ

äÿžäžĘä£őæ■čè£ŹæăůçŽĎéŤŹèŕŕïijŇä¡ăéIJĂèęĄåIJÍèřČçŤÍ exec() äźŃåĽ■ä¡£çŤÍ locals() åĞ¡æŢřæİěå¿ŮåĹřäÿĂäÿłåśĂéČÍåŔŸéĞŔå■ŮåĚÿãĂĆ äźŃåŘŐä¡ăåřśèČ¡äżŐåśĂéČÍå■ŮåĚÿäÿ■èŐůåŔŰä£őæŤźè£ĞåŘŐçŽĎåŔŸéĞŔåĂijäžĘãĂĆä¿Ńåęĺïijž

```
>>> def test():
...     a = 13
...     loc = locals()
...     exec('b = a + 1')
...     b = loc['b']
...     print(b)
...
>>> test()
14
>>>
```

èőĺèőž

åőđéŹĚäÿŁåŕźäžŐ exec() çŽĎæ■čçąőä¡£çŤÍæŸŕæŕŤè¿ČéŽ¿çŽĎãĂĆåď§åď§ŽæŢřæČĚåĘţäÿŃå¡Şä¡ăè exec() çŽĎæŮűåĂŹïijŇ è£ŸæIJĽåŔęåď Űæŀť åĕ¡çŽĎèğčåĘşæŰźæąĹïijĹæŕŤåęĆèčĚééřåŹĺãĂĄéŮ■åŇĚã

çĎűèĂŇïijŇåęĆæđIJä¡ăäž■çĎűèęĄä¡£çŤÍ exec() ïijŇæIJňèŁĆåĹŮåĞžäžĘäÿĂäžŻåęĆä¡Ţæ■čçąőä¡£ç éżŸèőď æČĚåĘţäÿŃïijŇexec() äijŽåIJÍèřČçŤÍèĂĚåśĂéČÍåŠŇåĚÍåśĂèŇČåŹť åĘĚæĽ§èąŇäżčçăĄãĂĆçĎű äijăéĂŠçżŹ exec() çŽĎåśĂéČÍèŇČåŹť æŸŕæŃůèť İåőđéŹĚåśĂéČÍåŔŸéĞŔçżĎæĹŘçŽĎäÿĂäÿłå■ŮåĚÿã åŽăæ■ď ïijŇåęĆæđIJ exec() åęĆæđIJæĽ§èąŇäžĘä£őæŤźæŞ■ä¡IJïijŇè£Źçğ■ä£őæŤźåŘŐçŽĎçżŞæđIJåŕźåőđ äÿŃéİćæŸŕåŔęåď ŰäÿĂäÿłæijŤçď žåőČçŽĎä¿Ńå■ŘïijŽ

```
>>> def test1():
...     x = 0
...     exec('x += 1')
...     print(x)
...
>>> test1()
```

```
0
>>>
```

äÿŁéİćäżčçăĄéĞŇïijŇå¡Şä¡ăèřČçŤÍ `locals()` èŐůåŔŰåśĂéČĺåŔŸéĞŔæŮűïijŇä¡ăèŐůå¿ŮçŽĎæŸŕäi
`exec()` çŽĎåśĂéČĺåŔŸéĞŔçŽĎäÿĂäÿłæŃůèťÍãĂĆ éĂŽè£ĞåIJĺäżčçăĄæĽğèąŇåŘŐåőąæ§ěè£ŹäÿłåńŮåĚ

```
>>> def test2():
...     x = 0
...     loc = locals()
...     print('before:', loc)
...     exec('x += 1')
...     print('after:', loc)
...     print('x =', x)
...
>>> test2()
before: {'x': 0}
after: {'loc': {...}, 'x': 1}
x = 0
>>>
```

äżŤçżĘèğĆåŕ§æIJĂåŘŐäÿĂæńěçŽĎè¿ŞåĞžïijŇéŹďéİđä¡ăåřĘ `loc`
äÿńèćńä£őæŤźåŘŐçŽĎåĂijæĽŃåŁĺèţŃåĂijçżŹxïijŇåŗęåĹŹxåŔŸéĞŔåĂijæŸŕäÿńäijŽåŔŸçŽĎãĂĆ

åIJĺä¡£çŤÍ `locals()` çŽĎæŮűåĂŹïijŇä¡ăéIJĂèęĄæşĺæĎŔæŞńä¡IJéąžåžŔãĂĆærŔæňąåőČèćńèřČçŤÍçŽ
`locals()` äijŽèŐůåŔŰåśĂéČĺåŔŸéĞŔåĂijäÿńçŽĎåĂijåźűèęĘçŽŰå■ŮåĚÿäÿńçŻÿåžŤçŽĎåŔŸéĞŔãĂĆ
èŕůæşĺæĎŔèğĆåŕ§äÿŃäÿŃéİćè£ŹäÿłèŕŢéłŇçŽĎè¿ŞåĞžçżŞæđIJïijŽ

```
>>> def test3():
...     x = 0
...     loc = locals()
...     print(loc)
...     exec('x += 1')
...     print(loc)
...     locals()
...     print(loc)
...
>>> test3()
{'x': 0}
{'loc': {...}, 'x': 1}
{'loc': {...}, 'x': 0}
>>>
```

æşĺæĎŔæIJĂåŘŐäÿĂæňąèřČçŤÍ `locals()` çŽĎæŮűåĂŹxçŽĎåĂijæŸŕåęĆä¡ŢèćńèęĕçŻŰæŐĽçŽĎãA

ä¡IJäÿž `locals()` çŽĎäÿĂäÿłæŻ£äżčæŰźæąĹïijŇä¡ăåŔŕäžëä¡£çŤÍä¡ăèĞłåůśçŽĎå■ŮåĚÿïijŇåźűåřĘåő
`exec()` ãĂĆä¿ŃåęĆïijŽ

```
>>> def test4():
...     a = 13
...     loc = { 'a' : a }
...     glb = { }
```

```
...     exec('b = a + 1', glb, loc)
...     b = loc['b']
...     print(b)
...
>>> test4()
14
>>>
```

åď ğéČĺåĹĘæČĚåĘţäÿŃïijŇè£Źçğ■æŰźåijŔæŸŕä¡£çŤĺ exec() çŽĎæIJĂä¡şåőđèůţãĂĆ ä¡ăåŔłéIJĂèęĄä£ÌèŕĄåĚĺåśĂåŠŇåśĂéČĺå■ŮåĚÿåIJĺåŘŐéÍćäżčçăĄèő£éŮőæŮűåůšçżŔèćńåĹÌåğŃåŇŰãĂĆ

è£ŸæIJL'äÿĂçĆźïijŇåIJĺä¡£çŤĺ exec() äźŃåL'■ïijŇä¡ăåŔŕèČ¡éIJĂèęĄéŮőäÿŃèĞłåůśæŸŕåŘęæIJL'åĚű åď ğåď ŽæŢřæČĚåĘţäÿŃå¡Şä¡ăèęĄèĂČèŹŠä¡£çŤĺ exec() çŽĎæŮűåĂŹïijŇ è£ŸæIJL'åŔęåď ŰæŻť'åě¡çŽĎèğčåĘşæŰźæąĹïijŇæŕŤåęĆèčĚéěřåŹĺãĂĄéŮ■åŇĚãĂĄåĚČçśżïijŇæĹŰåĚűäżŰ

9.24 èğčæđŘäÿŐåĹĘæđŘPythonæžŘçăĄ

éŮőécŸ

ä¡ăæČşåĘŹèğčæđŘåźűåĹĘæđŘPythonæžŘäżčçăĄçŽĎçÍŃåžŔãĂĆ

èğčåĘşæŰźæąĹ

åď ğéČĺåĹĘçÍŃåžŔåŚŸç§ěéĄŞPythonèČ¡åď §èőąçőŮæĹŰæL'ğèąŇå■Ůçňęäÿšå¡ćåijŔçŽĎæžŘäżčçăĄãĂ

```
>>> x = 42
>>> eval('2 + 3*4 + x')
56
>>> exec('for i in range(10): print(i)')
0
1
2
3
4
5
6
7
8
9
>>>
```

ař¡çőąåęĆæ■d'ïijŇast æĺąåİŮèČ¡èćńçŤĺæİěářĘPythonæžŘçăĄçijŰèŕŚæĹŘäÿĂäÿłåŔŕèćńåĹĘæđŘçŽĎa

```
>>> import ast
>>> ex = ast.parse('2 + 3*4 + x', mode='eval')
>>> ex
```

```
<_ast.Expression object at 0x1007473d0>
>>> ast.dump(ex)
"Expression(body=BinOp(left=BinOp(left=Num(n=2), op=Add(),
right=BinOp(left=Num(n=3), op=Mult(), right=Num(n=4))), op=Add(),
right=Name(id='x', ctx=Load())))"

>>> top = ast.parse('for i in range(10): print(i)', mode='exec')
>>> top
<_ast.Module object at 0x100747390>
>>> ast.dump(top)
"Module(body=[For(target=Name(id='i', ctx=Store()),
iter=Call(func=Name(id='range', ctx=Load()), args=[Num(n=10)],
keywords=[], starargs=None, kwargs=None),
body=[Expr(value=Call(func=Name(id='print', ctx=Load()),
args=[Name(id='i', ctx=Load())], keywords=[], starargs=None,
kwargs=None))], orelse=[])])"
>>>
```

åĹĘæđŘæžŘçăĄæăŚéIJĂèęĄä¡äèĞłåůśæŻt'åd'ŽçŽĎå■ęäźăïijŇåőČæŸŕçŤśäÿĂçşžåĹŮASTèŁĆçĆźçżĎ
åĹĘæđŘè£ŹäžŻèŁĆçĆźæIJĂçőĂå■ŢçŽĎæŰźæşŢářśæŸŕåőŽäźL'äÿĂäÿłèő£éŮőèĂĚçśżïijŇåőđçŐřå¿Łåd'Ž
`visit_NodeName()` æŰźæşŢïijŇ `NodeName()` åŇźéĚ■éĆčäžŻä¡äæĎ§åĚt'èűčçŽĎèŁĆçĆźãĂĆäÿŃéIćæ

```
import ast

class CodeAnalyzer(ast.NodeVisitor):
    def __init__(self):
        self.loaded = set()
        self.stored = set()
        self.deleted = set()

    def visit_Name(self, node):
        if isinstance(node.ctx, ast.Load):
            self.loaded.add(node.id)
        elif isinstance(node.ctx, ast.Store):
            self.stored.add(node.id)
        elif isinstance(node.ctx, ast.Del):
            self.deleted.add(node.id)

# Sample usage
if __name__ == '__main__':
    # Some Python code
    code = '''
    for i in range(10):
        print(i)
    del i
    '''

    # Parse into an AST
    top = ast.parse(code, mode='exec')
```

```
    # Feed the AST to analyze name usage
    c = CodeAnalyzer()
    c.visit(top)
    print('Loaded:', c.loaded)
    print('Stored:', c.stored)
    print('Deleted:', c.deleted)
```

åęĆæđIJä¡ăè£ŘèąŇè£ŹäÿłçÍŃåžŔïijŇä¡ăäijŽå¿ŮåĹřäÿŃéİćè£ŹæăůçŽĎè¿ŞåĞžïijŽ

```
Loaded: {'i', 'range', 'print'}
Stored: {'i'}
Deleted: {'i'}
```

æIJĂåŘŐïijŇASTåŔřäžěéĂŽè£Ğ `compile()` åĞ¡æŢřæİěçijŰèŕŚåźűæĽğèąŇãĂĆä¿ŃåęĆïijŽ

```
>>> exec(compile(top,'<stdin>', 'exec'))
0
1
2
3
4
5
6
7
8
9
>>>
```

èőÍèőž

å¡Şä¡ăèČ¡åď§åĹĘæđŘæžŘäžčçăĄåźűäžŐäÿ■èŐůåŔŰä£ąæĄŕçŽĎæŮűůåĂŹïijŇä¡ăåřśèČ¡åĘŹå¿Ĺåď Žäžč
ä¿ŃåęĆïijŇçŻÿæŕŢçŽšçŽőçŽĎäijăéĂŠäÿĂäžŻäžčçăĄçĽĞæőţåĹřçśžäijij
`exec()` åĞ¡æŢřäÿ■ïijŇä¡ăåŔŕäžěåĚĹåřĘåőČè¡ňæ■ćæĹŘäÿĂäÿłASTïijŇ
çĎűåŘŐèğĆår§åőČçŽĎçżĘèŁĆçIJŃåőČåĹřåžŢæŸŕæĂŐæăůåĄŽçŽĎãĂĆ
ä¡ăè£ŸåŔŕäžěåĘŹäÿĂäžŻåůěåĚůæİěæ§ěçIJŃæ§ŘäÿłæĺąåİŮçŽĎåĚĺéČĺæžŘçăĄïijŇåźűäÿŤåIJĺæ■ď å§žçąĂäÿ

éIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇåęĆæđIJä¡ăç§ěéĄŞèĞłåůšåIJĺåźšåŢěïijŇä¡ăè£ŸèČ¡åď §éĞ■åĘŹASTæİěèą
äÿŃéİćæŸŕäÿĂäÿłèčĚéěřåŹĺä¿Ńå■ŘïijŇåŔŕäžěéĂŽè£ĞéĞ■æŰřèğčæđŘåĞ¡æŢřä¡ŞæžŘçăĄãĂĄ
éĞ■åĘŹASTåźűéĞ■æŰřåĹŽåžžåĞ¡æŢřäžčçăĄåŕźèśąæİěåřĘåĚĺåśĂèő£éŮőåŔŸéĞŔéŹ■äÿžåĞ¡æŢřä¡Şä¡IJçŤ

```
# namelower.py
import ast
import inspect

# Node visitor that lowers globally accessed names into
# the function body as local variables.
class NameLower(ast.NodeVisitor):
    def __init__(self, lowered_names):
        self.lowered_names = lowered_names
```

```
    def visit_FunctionDef(self, node):
        # Compile some assignments to lower the constants
        code = '__globals = globals()\n'
        code += '\n'.join("{0} = __globals['{0}']".format(name)
                          for name in self.lowered_names)
        code_ast = ast.parse(code, mode='exec')

        # Inject new statements into the function body
        node.body[:0] = code_ast.body

        # Save the function object
        self.func = node

# Decorator that turns global names into locals
def lower_names(*namelist):
    def lower(func):
        srclines = inspect.getsource(func).splitlines()
        # Skip source lines prior to the @lower_names decorator
        for n, line in enumerate(srclines):
            if '@lower_names' in line:
                break

        src = '\n'.join(srclines[n+1:])
        # Hack to deal with indented code
        if src.startswith((' ','\t')):
            src = 'if 1:\n' + src
        top = ast.parse(src, mode='exec')

        # Transform the AST
        cl = NameLower(namelist)
        cl.visit(top)

        # Execute the modified AST
        temp = {}
        exec(compile(top,'','exec'), temp, temp)

        # Pull out the modified code object
        func.__code__ = temp[func.__name__].__code__
        return func
    return lower
```

äÿžäžĘä¡£çŤĺè£ŹäÿłäżčçăĄïijŇä¡ăåŔŕäžěåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
INCR = 1
@lower_names('INCR')
def countdown(n):
    while n > 0:
        n -= INCR
```

èčĚéěřåŹĺäijŽåřĘ `countdown()` åĞ¡æŢřéĞ■åĘŹäÿžçśżäijijäÿŃéİćè£ŹæăůåŕŔïijŽ

```
def countdown(n):
    __globals = globals()
    INCR = __globals['INCR']
    while n > 0:
        n -= INCR
```

åIJÍæĂğèČ¡æţŃèŕŢäÿ■ïijŇåőČäijŽèőľåĞ¡æŢřè£ŘèąŇå£ń20%

çŐřåIJĺïijŇä¡ăæŸŕäÿ■æŸŕæČşäÿžä¡ăæĽĂæIJL'çŽĎåĞ¡æŢřéČ¡åŁääÿŁè£ŹäÿłèčĕéěřåŹĺåŚćïij§æĹŰèőÿä ä¡ĘæŸŕïijŇè£Źå■t'æŸŕåŕźäžŐäÿĂäžŻéńŸçžğæŁĂæIJŕæŕŤåęĆASTæŞ■ä¡IJãĂĄæžŘçăĄæŞ■ä¡IJç■Ľç■Ľç■ŽĎ

æIJňèŁĆåŔŮåŔęåd'ŰäÿĂäÿłåIJÍ `ActiveState` äÿ■åd'ĎçŘĘPythonå■ŮèŁĆçăĄçŽĎçńäèŁĆçŽĎåŘŕçd ä¡£çŤÍASTæŸŕäÿĂäÿłæŻt'åŁäéńŸçžğçĆźçŽĎæŁĂæIJŕïijŇåźűäÿŤäź§æŻt'çőĂå■ŢäžŻãĂĆåŔĆèĂČäÿŃéÍćäÿ

9.25 æŃĘèğčPythonå■ŮèŁĆçăĄ

éŮőécŸ

ä¡ăæČşéĂŽè£ĞåřĘä¡ăçŽĎäžčçăĄåŘ■çijŰèŕŚæĹŘä¡ŐçžğçŽĎå■ŮèŁĆçăĄæÍěæ§ěçIJŃåőČåžŢåśĆçŽĎå

èğčåĘşæŰźæąĹ

`dis` æÍąåİŮåŘŕäżěèćńçŤÍæÍěè¿ŞåĞžäżżä¡ŢPythonåĞ¡æŢřçŽĎåŘ■çijŰèŕŚçżŞæđIJãĂĆä¿ŃåęĆïijŽ

```
>>> def countdown(n):
...     while n > 0:
...         print('T-minus', n)
...         n -= 1
...     print('Blastoff!')
...
>>> import dis
>>> dis.dis(countdown)
...
>>>
```

èőÍèőž

å¡Şä¡ăæČşèęĄç§ěéĄŞä¡ăçŽĎçÍŃåžŘåžŢåśĆçŽĎè£ŘèąŇæIJžåĹűçŽĎæŮűåĂŹïijŇ`dis` æÍąåİŮæŸŕå¿ĹæIJL'çŤÍçŽĎãĂĆæŕŤåęĆåęĆæđIJä¡ăæČşèŕŢçİĂçŘĘèğčæĂğèČ¡çL'źå¿ĄãĂĆ èćń `dis()` åĞ¡æŢřèğčæđŘçŽĎåŐ§ågŃå■ŮèŁĆçăĄåęĆäÿŃæL'Ăçđ'žïijŽ

```
>>> countdown.__code__.co_code
b"x
 ↪'\x00|\x00\x00d\x01\x00k\x04\x00r)\x00t\x00\x00d\x02\x00|\x00\x00\x83
\x02\x00\x01|\x00\x00d\x03\x008}
 ↪\x00\x00q\x03\x00Wt\x00\x00d\x04\x00\x83
```

```
\x01\x00\x01d\x00\x00S"
>>>
```

åęĆæđIJä¡ăæČşèĞłåůśèğčéĞŁè£Źæőţäžčçă Ąïij Ñä¡ăéIJĂèęĄä¡£çŤĺäÿĂäžŽåIJĺ opcode æĺąåİŮäÿ■åőŽäźĽçŽĎåÿýéĞŕãĂĆä¿ŃåęĆïijŽ

```
>>> c = countdown.__code__.co_code
>>> import opcode
>>> opcode.opname[c[0]]
>>> opcode.opname[c[0]]
'SETUP_LOOP'
>>> opcode.opname[c[3]]
'LOAD_FAST'
>>>
```

åěĞæĂłçŽĎæŸŕïijŃåIJĺ dis æĺąåİŮäÿ■åźűæšąæIJL'åĞ¡æŢřèőľä¡ääžěçijŰçÍŃæŰźåijŕå¿Ĺåőźæŷ§çŽĎ äÿ■è£ĞïijŃäÿŃéİćçŽĎçŤ§æĹŘåŹĺåĞ¡æŢřåŕŕäžěářĘåŐ§ågŃå■ŮèŁĆçăĄåžŕåĹŮè¡ñæ■ćæĹŘ opcodes åŠŇåŕĆæŢřãĂĆ

```
import opcode

def generate_opcodes(codebytes):
    extended_arg = 0
    i = 0
    n = len(codebytes)
    while i < n:
        op = codebytes[i]
        i += 1
        if op >= opcode.HAVE_ARGUMENT:
            oparg = codebytes[i] + codebytes[i+1]*256 + extended_arg
            extended_arg = 0
            i += 2
            if op == opcode.EXTENDED_ARG:
                extended_arg = oparg * 65536
                continue
        else:
            oparg = None
        yield (op, oparg)
```

ä¡£çŤĺæŰźæşŢåęĆäÿŃïijŽ

```
>>> for op, oparg in generate_opcodes(countdown.__code__.co_code):
...     print(op, opcode.opname[op], oparg)
```

è£Źçğ■æŰźåijŕå¿ĹåřŚæIJL'äžžç§ěéĄŞïijŃä¡ăåŕŕäžěåĹl'çŤĺåőĆæŻ£æ■ćäżżä¡Ţä¡äæČşèęĄæŻ£æ■ćçŽĎ äÿŃéİćæĹŚäżñçŤĺäÿĂäÿłçd'žä¿ŃæİěæijŤçd'žæŢt'äÿłè£ĞçĺŃïijŽ

```
>>> def add(x, y):
...     return x + y
...
>>> c = add.__code__
```

```
>>> c
<code object add at 0x1007beed0, file "<stdin>", line 1>
>>> c.co_code
b'|\x00\x00|\x01\x00\x17S'
>>>
>>> # Make a completely new code object with bogus byte code
>>> import types
>>> newbytecode = b'xxxxxxx'
>>> nc = types.CodeType(c.co_argcount, c.co_kwonlyargcount,
...     c.co_nlocals, c.co_stacksize, c.co_flags, newbytecode, c.co_
 ↪consts,
...     c.co_names, c.co_varnames, c.co_filename, c.co_name,
...     c.co_firstlineno, c.co_lnotab)
>>> nc
<code object add at 0x10069fe40, file "<stdin>", line 1>
>>> add.__code__ = nc
>>> add(2,3)
Segmentation fault
```

ä¡ăåŔŕäžěåČŔè£ŹæăůèĂ■åď ğæŃŻèőľèğčéĞŁåŹĺåěŤæžČãĂĆä¡ĘæŸŕïijŇåŕźäžŐçijŰåĘŹæŻťéńŸçžğäi
äżŰäžňåŔŕèČ¡çIJ§çŽĎéIJĂèęĄéĞ■åĘŹå■ŮèŁĆçăĄãĂĆæIJňèŁĆæIJĂåŘŐçŽĎéČĺåĹĘæijŤçď žäžĘè£ŹäÿłæŸ
this code on ActiveState

CHAPTER 12

çňňå■Ąçńäïij Žæĺąå İŮäÿŐåŇĚ

æĺąåİŮäÿŐåŇĚæŸŕäżżä¡Ţåď ğåđŃçÍŃåžŔçŽĎæäÿå£ČïijŇåřśè£đPythonåőL'èčĚçÍŃåžŔæIJňèžńäź§æŸŕ

Contents:

10.1 æđĎåżžäÿĂäÿłæĺąåİŮçŽĎåśĆçžğåŇĚ

éŮőécŸ

ä¡ăæČşåřĘä¡ăçŽĎäżčçăĄçżĎçżĞæĹŘçŤśå¿Ĺåď ŽåĹĘåśĆæĺąåİŮæđĎæĹŘçŽĎåŇĚãĂĆ

èğčåĘşæŰźæąĹ

åřĄèčĚæĹŘåŇĚæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆåIJÍæŰĞäżűçşżçż§äÿŁçżĎçżĞä¡ăçŽĎäżčçăĄïijŇåźűçąőä£İæŕ
ä¿ŃåęĆïijŽ

```
graphics/
    __init__.py
    primitive/
        __init__.py
        line.py
        fill.py
        text.py
    formats/
        __init__.py
        png.py
        jpg.py
```

äÿĂæŮęä¡ăåĄŽåĹřäžĘè£ŹäÿĂçĆźïijŇä¡ăåžŤèŕêèČ¡åd'§æĽğèąŇåŘĎçğ■importèŕ■åŕëïijŇåęĆäÿŃïijŽ

```
import graphics.primitive.line
from graphics.primitive import line
import graphics.formats.jpg as jpg
```

èőÍèőž

åőŽäźĽæÍąåİŮçŽĎåśĆæňąçżŞæđĎåřšåČŕåIJÍæŰĞäžűçşżçż§äÿŁåžžçńŃçŽőå¡ŢçżŞæđĎäÿĂæăůåőźæŸ
æŰĞäžű__init__.pyçŽĎçŽőçŽĎæŸŕèęĄåŇĚåŘńäÿ■åŘŇè£ŘèąŇçžğåĹńçŽĎåŇĚçŽĎåŔŕéĂĽçŽĎåĹİåğŃåŇ
äÿ¿äÿłä¿Ńå■ŘïijŇåęĆæđIJä¡ăæĽğèąŇäžĘèŕ■åŔëimport graphicsïijŇ æŰĞäžűgraph-
ics/__init__.pyåřĘèćńåŕijåĚë,åżžçńŃgraphicsåŚ¡åŘ■çľžéŮť çŽĎåĘĚåőźãĂĆåČŔimport
graphics.format.jpgè£ŹæăůåŕijåĚëïijŇæŰĞäžűgraphics/__init__.pyåŠŇæŰĞäžűgraphics/graphics/formats/__

çżİåd'ğéČĺåĹĘæŮűåĂŹèőľ__init__.pyçľžçİĂåřšåě¡ãĂĆä¡ĘæŸŕæIJL'äžŽæČĚåĘţäÿŃåŔŕèČ¡åŇĚåŘńäżč
äÿ¿äÿłä¿Ńå■ŘïijŇ__init__.pyèČ¡åd'§çŤĺæİěèĞłåŁĺåŁăè¡¡å■ŘæĺąåİŮ:

```
# graphics/formats/__init__.py
from . import jpg
from . import png
```

åČŔè£ŹæăůäÿĂäÿłæŰĞäžű,çŤĺæĹůåŔŕäžëäžĚäžĚéĂŽè£Ğimport grah-
pics.formatsæİěäžčæŽ£import graphics.formats.jpgäžěåŔŁimport graphics.formats.pngãĂĆ

__init__.pyçŽĎåĚűäžŰåÿÿçŤĺçŤĺæşŢåŇĚæŃňåřĘåd'ŽäÿłæŰĞäžűåŘĹåźűåĹŕäÿĂäÿłéĂżè¿ŚåŚ¡åŘ■çľžé

æŢŔéŤŘçŽĎçĺŃåžŔåŚŸäijŽåŘŚçŐřïijŇå■şä¡£æšąæIJL'__init__.pyæŰĞäžűå■ŸåIJĺïijŇpythonäż■çĎűä

10.2 æŐğåĹűæĺąåİŮèćńåĚĺéČĺåŕijåĚëçŽĎåĘĚåőź

éŮőécŸ

å¡Şä¡£çŤĺ'from module import *' èŕ■åŔëæŮűïijŇåÿŇæIJŽåŕźäžŐæĺąåİŮæĹŰåŇĚåŕijåĞžçŽĎçňęåŔůè£Z

èğčåĘşæŰźæąĹ

åIJĺä¡ăçŽĎæĺąåİŮäÿ■åőŽäźĽäÿĂäÿłåŔŸéĞŔ __all__ æİěæŸŐçąőåIJřåĹŮåĞžéIJĂèęĄåŕijåĞžçŽĎåĘĚå

äÿ¿äÿłä¿Ńå■Ř:

```
# somemodule.py
def spam():
    pass

def grok():
    pass
```

```
blah = 42
# Only export 'spam' and 'grok'
__all__ = ['spam', 'grok']
```

èőÍèőž

åř¡çőąåijžçČĹåŔ■åŕźä¡£çŤÍ 'from module import *', ä¡ĘæŸŕåIJĺåőŽäźĽäžĘåď ğéĞŔåŔŸéĞŔåŘ■çŽĎæĺąåİŮäÿ■écŚçźĄä¡£çŤĺãĂĆ åęĆæđIJä¡ääÿ■åĄŽäżżä¡ŢäžŃ, è£ŹæăůçŽĎåŕijåĚěåřĘäijŽåŕijåĚěæĽĂæIJL'äÿ■äžěäÿŃåĹŠçž£åijĂåď't'çŽĎã åŘęäÿĂæŰźéİć,åęĆæđIJåőŽäźĽäžĘ __all__ , éĆčäźĹåŔłæIJL'èćńåĹŮäÿ¿åĞžçŽĎäÿIJèě£äijŽèćńåŕijåĞžãĂĆ

åęĆæđIJä¡ăåřĘ __all__ åőŽäźĽæĹŘäÿĂäÿłçľžåĹŮèąĺ, æšąæIJL'äÿIJèě£åřĘèćńåŕijåĞžãĂĆ åęĆæđIJ __all__ åŃĚåŘńæIJłåőŽäźĽçŽĎåŘ■å■Ů, åIJĺåŕijåĚěæŮűåijŢètůAttributeErrorãĂĆ

10.3 ä¡£çŤÍçŻÿåŕźèůŕå¿ĎåŘ■åŕijåĚěåŃĚäÿ■å■ŘæĺąåİŮ

éŮőécŸ

åřĘäžčçăĄçżĎçżĞæĹŘåŃĚ,æČşçŤĺimportèŕ■åŔëäžŐåŔęäÿĂäÿłåŃĚåŘ■æšąæIJL'çąńçijŰçăĄè£ĞçŽĎå

èğčåĘşæŰźæąĹ

ä¡£çŤĺåŃĚçŽĎçŻÿåŕźåŕijåĚëijŃä¡£äÿĂäÿłçŽĎæĺąåİŮåŕijåĚěåŘŃäÿĂäÿłåŃĚçŽĎåŔęäÿĂäÿłæĺąåİŮ äÿ¿äÿłä¿Ńå■ŘijŃåĄĞèő¿åIJĺä¡ăçŽĎæŰĞäžűçşżçż§äÿŁæIJL'mypackageåŃĚijŃçżĎçżĞåęĆäÿŃijŽ

```
mypackage/
    __init__.py
    A/
        __init__.py
        spam.py
        grok.py
    B/
        __init__.py
        bar.py
```

åęĆæđIJæĺąåİŮmypackage.A.spamèęĄåŕijåĚěåŘŃçŻőå¡ŢäÿŃçŽĎæĺąåİŮgrokïijŃåőĆåžŤèŕěåŃĚæŃńçŽ

```
# mypackage/A/spam.py
from . import grok
```

åęĆæđIJæĺąåİŮmypackage.A.spamèęĄåŕijåĚëäÿ■åŘŃçŻőå¡ŢäÿŃçŽĎæĺąåİŮB.barïijŃåőĆåžŤèŕěä¡£çŤĺ

```
# mypackage/A/spam.py
from ..B import bar
```

äÿď'äÿłimportèŕ■åŔěéČ¡æšąåŇĚåŘńéąűåśĆåŇĚåŘ■ïijŇèĂŇæŸŕä¡£çŤĺäžĘspam.pyçŽĎçŻÿåŕźèůŕå¿ĎãĂ

èőĺèőž

åIJĺåŇĚåĘĚïijŇæŮćåŔŕäžěä¡£çŤĺçŻÿåŕźèůŕå¿Ďäź§åŔŕäžěä¡£çŤĺçżÌåŕźèůŕå¿ĎæÌěåŕijåĚěãĂĆ äÿ¿äÿłä¿Ńå■ŘïijŽ

```
# mypackage/A/spam.py
from mypackage.A import grok # OK
from . import grok # OK
import grok # Error (not found)
```

åČŔmypackage.Aè£Źæăůä¡£çŤĺçżÌåŕźèůŕå¿ĎåŘ■çŽĎäÿ■åĹl'äźŃåd'ĎæŸŕè£ŹåřĘéąűåśĆåŇĚåŘ■çąňçijU äÿ¿äÿłä¿Ńå■ŘïijŇåęĆædIJä¡ăæŤźåŔŸäžĘåŇĚåŘ■ïijŇä¡ăåřśå£ĚéążæčĂæ§ěæL'ĂæIJL'æŰĞäžűæÌěä£őæ■čæż åŘŇæăůïijŇçąňçijŰçăĄçŽĎåŘ■çğřäijŽä¡£çğżåŁĺäžčçăĄåŔŸå¿ŮåŽŕéŽ¿ãĂĆäÿ¿äÿłä¿Ńå■ŘïijŇäź§èőÿæIJL'ä åęĆædIJä¡£çŤĺçŻÿåŕźåŕijåĚěïijŇéĆčäÿĂåĹĞéČ¡okïijŇçĎűèĂŇä¡£çŤĺçżÌåŕźèůŕå¿ĎåŘ■å¿ĹåŔŕèČ¡äijŽåĞžéŮ

importèŕ■åŔěçŽĎ . åŠŇ "..''çIJŃètůæÌěå¿ĹæżŚçĺ¡, ä¡ĘåőČæŇĞåőŽçŻőå¡ŢåŘ■.äÿžå¡ŞåL'■çŽőå¡ŢïijŇ..BäÿžçŽőå¡Ţ../BãĂĆè£Źçğ■èŕ■æşŢåŔłéĂĆçŤĺäžŐimport äÿ¿äÿłä¿Ńå■ŘïijŽ

```
from . import grok # OK
import .grok # ERROR
```

åř¡çőąä¡£çŤĺçŻÿåŕźåŕijåĚěçIJŃètůæÌěåČŔæŸŕæţŔèğĹæŰĞäžűçşżçż§ïijŇä¡ĘæŸŕäÿ■èČ¡åĹřåőŽäźL'åŇĚ

æIJĂåŘŐïijŇçŻÿåŕźåŕijåĚěåŔłéĂĆçŤĺäžŐåIJĺåŘĹéĂĆçŽĎåŇĚäÿ■çŽĎæĺąåÌŮãĂĆård'åĚűæŸŕåIJĺéąűås ä¿ŃåęĆïijŽ

```
% python3 mypackage/A/spam.py # Relative imports fail
```

åŔęäÿĂæŰźéÌćïijŇåęĆædIJä¡ăä¡£çŤĺPythonçŽĎ-méĂL'éąźæÌěæL'ğèąŇåĚĹåL'■çŽĎèĎŽæIJñïijŇçŻÿåŕ ä¿ŃåęĆïijŽ

```
% python3 -m mypackage.A.spam # Relative imports work
```

æŻt'åd'ŽçŽĎåŇĚçŽĎçŻÿåŕźåŕijåĚěçŽĎèČŇæŹŕç§ěèŕĘ,èŕůçIJŃ PEP 328 .

10.4 åřĘæĺąåÌŮåĹĘåL'šæĹŘåd'Žäÿłæ ŰĞäžű

éŮőécŸ

ä¡ăæČşåřĘäÿĂäÿłæĺąåÌŮåĹĘåL'šæĹŘåd'Žäÿłæ ŰĞäžűãĂĆä¡ĘæŸŕä¡ăäÿ■æČşåřĘåĹĘçężçŽĎæŰĞäžűçż

èğčåĘşæŰźæąĹ

çĺŃåžŔæĺąåİŮåŔŕäžěéĂŽè£ĞåŔŸæĹŘåŇĚæİěåĹĘåĽ'šæĹŘåd'Žäÿłç Ńňçń ŃçŽĎæŰĞäžűãĂĆèĂČèŹŚäÿ

```
# mymodule.py
class A:
    def spam(self):
        print('A.spam')

class B(A):
    def bar(self):
        print('B.bar')
```

åĄĞèő¿ä¡ăæČşmymodule.pyåĹĘäÿžäÿd'äÿłæŰĞäžűïijŇæŕŔäÿłåőŽäźL'çŽĎäÿĂäÿłçśžãĂĆèęĄåĄŽåĹřè
è£Źè£ŹäÿłçŻőå¡ŢäÿŇïijŇåĹŻåžžäžěäÿŇæŰĞäžűïijŽ

```
mymodule/
    __init__.py
    a.py
    b.py
```

åIJĺa.pyæŰĞäžűäÿ■æŔŠåĚěäžěäÿŇäžčçăĄïijŽ

```
# a.py
class A:
    def spam(self):
        print('A.spam')
```

åIJĺb.pyæŰĞäžűäÿ■æŔŠåĚěäžěäÿŇäžčçăĄïijŽ

```
# b.py
from .a import A
class B(A):
    def bar(self):
        print('B.bar')
```

æIJĂåŘŐïijŇåIJĺ __init__.py äÿ■ïijŇářĘ2äÿłæŰĞäžűçšŸåŘĹåIJĺäÿĂèţůïijŽ

```
# __init__.py
from .a import A
from .b import B
```

åęĆædIJæŇL'çĚğè£ŹäžŻæ■ěéłd'ïijŇæL'Ăäžğç Ť§çŽĎåŇĚMyModuleářĘä¡IJäÿžäÿĂäÿłå■ŢäÿĂçŽĎéĂż

```
>>> import mymodule
>>> a = mymodule.A()
>>> a.spam()
A.spam
>>> b = mymodule.B()
>>> b.bar()
```

```
B.bar
>>>
```

èőĺèőž

åIJĺè£Źäÿłçńäèŀ Ćäÿ■çŽĎäÿżèęĄéŮőécŸæŸŕäÿĂäÿłèő¿èőąéŮőécŸïijŇäÿ■çőąä¡ăæŸŕåŘęåÿŇæIJŻçŤĺæ

```
from mymodule.a import A
from mymodule.b import B
...
```

è£ŹæăůèČ¡åůëä¡IJïijŇä¡Ęè£Źèőľ çŤĺæĹůæL'£åŔŮæŻť'åd'ŽçŽĎèť §æŃĚïijŇçŤĺæĹůèęĄç§ěéĄŞäÿ■åŘŇ

```
from mymodule import A, B
```

åŕźåŘŐèĂĚèĂŇèĺĂïijŇèőľ mymoduleæĹŘäÿžäÿĂäÿłåd'ğçŽĎæžŘæŰĞäżűæŸŕæIJĂåÿÿèğĄçŽĎãĂĆä¡Ł
è£ŹæăůåĄŽçŽĎåĚşéŤőæŸŕåĹŻåžžäÿĂäÿłåŇĚçŽőå¡ŢïijŇä¡£çŤĺ __init__.py
æŰĞäżűæİěåřĘæŕŔéČĺåĹĘçšŸåŘĹåIJĺäÿĂèţůãĂĆ

å¡ŞäÿĂäÿłæĺąåİŮèćńåĹĘåL'šïijŇä¡äéIJĂèęĄçL'źåĹńæşĺæĎŔäžd'åŔL'åijŢçŤĺçŽĎæŰĞäżűåŘ■ãĂĆäÿ¿äÿ
from .a import A æİěèŐůåŔŰãĂĆ

æŢť'äÿłçńäèŀĆéČ¡ä¡£çŤĺåŇĚçŽĎçŻÿåŕźåŕijåĚěæİěéĄ£åĚ■åřĘéąűåśĆæĺąåİŮåŘ■çąňçijŰçăĄåĹřæžŘäżč

ä¡IJäÿžè£ŹäÿĂçńäèŀĆçŽĎåžűäijÿïijŇåřĘäżŃçż■åžűè£§åŕijåĚěãĂĆåęĆåŻ¿æL'Ăçd'žïijŇ__init__.pyæŰ
èęĄåĄŽåĹřè£ŹäÿĂçĆźïijŇ__init__.pyæIJL'çżĘå¿őçŽĎåŔŸåŇŰïijŽ

```
# __init__.py
def A():
    from .a import A
    return A()

def B():
    from .b import B
    return B()
```

åIJĺè£Źäÿłç L'ĹæIJñäÿ■ïijŇçśżAåŠŇçśżBècńæŻ£æ■ćäÿžåIJĺçňňäÿĂæňąèő£éŮőæŮűåŁäè¡¡æL'ĂéIJĂçŽĎ
ä¿ŃåęĆïijŽ

```
>>> import mymodule
>>> a = mymodule.A()
>>> a.spam()
A.spam
>>>
```

åžűè£§åŁäè¡¡çŽĎäÿżèęĄçijžçĆźæŸŕçżğæL'£åŠŇçśżåđŇæčĂæ§ěåŔŕèČ¡äijŽäÿ■æŰ■ãĂĆä¡ăåŔŕèČ¡äijŽč

```
if isinstance(x, mymodule.A): # Error
...
```

```
if isinstance(x, mymodule.a.A): # Ok
...
```

åžűè£§åŁäè¡¡çŽĎçIJ§åőđä¿Ńå■Ř, èğĄæăĞåĞĘåžŞ multiprocessing/__init__.py çŽĎæžŘçăĄ.

10.5 åĹľçŤĺåŚ¡åŘ■çľžéŮťåŕijåĚěçŻőå¡ŢåĹĘæŢčçŽĎäżčçăĄ

éŮőécŸ

ä¡ăåŔŕèČ¡æIJL'åd'ğéĞŔçŽĎäżčçăĄïijŇçŤšäÿ■åŘŇçŽĎäžžæİěåĹĘæŢčåIJřçżť'æŁd'ãĂĆæŕŔäÿłéČĺåĹĘè

èğčåĘşæŰźæąĹ

äžŐæIJňèť'ĺäÿŁèőšïijŇä¡ăèęĄåőŽäźL'äÿĂäÿłéąűçžğPythonåŇĚïijŇä¡IJäÿžäÿĂäÿłåd'ğéŻĘåŘĹåĹĘåijĂç

åIJĺçż§äÿĂäÿ■åŘŇçŽĎçŻőå¡ŢéĞŇçż§äÿĂçŻÿåŘŇçŽĎåŚ¡åŘ■çľžéŮť'ïijŇä¡ĘæŸŕèęĄåĹăåŐżçŤĺæİěåřH

```
foo-package/
    spam/
        blah.py

bar-package/
    spam/
        grok.py
```

åIJĺè£Ź2äÿłçŻőå¡ŢéĞŇïijŇéČ¡æIJL'çİĂåĚśåŘŇçŽĎåŚ¡åŘ■çľžéŮť'spamãĂĆåIJĺäżžä¡ŢäÿĂäÿłçŻőå¡ŢéĞ

èőľ'æĹŚäžňçIJŃçIJŃïijŇåęĆædIJåřĘfoo-packageåŠŇbar-packageéČ¡åŁăåĹřpythonæĺąåIJŮèůŕå¿ĎåźűåřİèŕŢåŕijåĚěäijŽåŔŚçŤ§äžĂäźĹ

```
>>> import sys
>>> sys.path.extend(['foo-package', 'bar-package'])
>>> import spam.blah
>>> import spam.grok
>>>
```

äÿd'äÿłäÿ■åŘŇçŽĎåŇĚçŻőå¡ŢèćńåŕĺåźűåĹřäÿĂèţůïijŇä¡ăåŔŕäžěåŕijåĚěspam.blahåŠŇspam.grokïijŇå

èőĺèőž

åIJĺè£ŹéĞŇåůěä¡IJçŽĎæIJžåĹűèćńçğřäÿžâĂIJåŇĚåŚ¡åŘ■çľžéŮť'âĂİçŽĎäÿĂäÿłçL'źå¿ĄãĂĆäżŐæIJňè

åŇĚåŚ¡åŘ■çľžéŮť'çŽĎåĚşéŤőæŸŕçąőä£ÌéąűçžğçŻőå¡Ţäÿ■æšąæIJL'__init__.pyæŰĞäžűæİěä¡IJäÿžåĚśäÿ¿äÿłä¿Ńå■ŘïijŽ

```
>>> import spam
>>> spam.__path__
_NamespacePath(['foo-package/spam', 'bar-package/spam'])
>>>
```

åIJĺåőŽä¡■åŇĚçŽĎå■ŘçżĎäžűæŮűïijŇçŻőå¡Ţ__path__åřĘèćńçŤĺåĹř(ä¿ŃåęĆ, å¡ŞåŕijåĚěspam.grokæĹŰèĂĚspam.blahçŽĎæŮűåĂŹ).

åŇĚåŚ¡åŘ■çl'žéŮt'çŽĎäÿĂäÿłéĞ■èęAçL'źçĆźæŸŕäżżä¡Ţäžžé Č¡åŕŕäžěçŤĺèĞłåůšçŽĎäžčçăĄæİěæL'l'åš

```
my-package/
    spam/
        custom.py
```

åęĆæđIJä¡ăåřĘä¡ăçŽĎäžčçăAçŻőå¡ŢåŠŇåĚűäžŰåŇĚäÿĂętůæůżåŁăåĹřsys.pathïijŇè£ŹåřĘæŮăçijİåIJřå

```
>>> import spam.custom
>>> import spam.grok
>>> import spam.blah
>>>
```

äÿĂäÿłåŇĚæŸŕåŘęèćńä¡IJäÿžäÿĂäÿłåŇĚåŚ¡åŘ■çl'žéŮt'çŽĎäÿżèęĄæŰźæşŢæŸŕæčĂæ§ěåĚű__file__åš

```
>>> spam.__file__
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: 'module' object has no attribute '__file__'
>>> spam
<module 'spam' (namespace)>
>>>
```

æŻt'åd'ŽçŽĎåŇĚåŚ¡åŘ■çl'žéŮt'ä£ąæĄŕåŕŕäžěæ§ěçIJŃ PEP 420.

10.6 éĞ■æŰřåŁăè¡¡æĺąåİŮ

éŮőécŸ

ä¡ăæČşéĞ■æŰřåŁăè¡¡åůšçżŕåŁăè¡¡çŽĎæĺąåİŮïijŇåŻääÿžä¡ăåŕźåĚűæžŘçăĄè£ŻèąŇäžĘä£őæŤźãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤĺimp.reload()æİěéĞ■æŰřåŁăè¡¡åĚĹåL'■åŁăè¡¡çŽĎæĺąåİŮãĂĆäÿ¿äÿłä¿Ńå■ŘïijŽ

```
>>> import spam
>>> import imp
>>> imp.reload(spam)
```

```
<module 'spam' from './spam.py'>
>>>
```

èőÍèőž

éĞ■æŰřåŁäè¡¡æĺąåİŮåIJĺåijĂåŔŚåŠŇèřČèŕŢè£ĞçÍŃäÿ■åÿÿåÿÿå¿ĹæIJĽçŤĺãĂĆä¡ĘåIJĺçŤ§äžğçŐŕåćČäÿ

reload()æŞęéŹđ'äžĘæĺąåİŮåžŢåśĆå■ŮåËÿçŽĎåĘĚåőźïijŇåźűéĂŽè£ĞéĞ■æŰřæĽ'ğèąŇæĺąåİŮçŽĎæžŘ

åř¡çőąåęĆæ■d'ïijŇreload()æšąæIJĽæŻť'æŰřåČŔ"from module import name"è£ŹæăůäĄ£çŤĺimportèŕ■åŘěåŕijåĚěçŽĎåőŽäźĽ'ãĂĆäÿ¿äÿłä¿Ńå■ŘïijŽ

```
# spam.py
def bar():
    print('bar')

def grok():
    print('grok')
```

çŐřåIJĺåŘřåŁĺäžd'äžŠåijŔäijŽèŕİïijŽ

```
>>> import spam
>>> from spam import grok
>>> spam.bar()
bar
>>> grok()
grok
>>>
```

äÿ■éĂĂåĞžPythonä£őæŤźspam.pyçŽĎæžŘçăĄïijŇářĘgrok()åĞ¡æŢřæŤźæĹŘè£ŹæåůïijŽ

```
def grok():
    print('New grok')
```

çŐřåIJĺåŻđåĹřäžd'äžŠåijŔäijŽèŕİïijŇéĞ■æŰřåŁäè¡¡æĺąåİŮïijŇářİèŕŢäÿŃè£ŹäÿłåőđéłŇïijŽ

```
>>> import imp
>>> imp.reload(spam)
<module 'spam' from './spam.py'>
>>> spam.bar()
bar
>>> grok() # Notice old output
grok
>>> spam.grok() # Notice new output
New grok
>>>
```

åIJĺè£Źäÿłä¿Ńå■ŘäÿïijŇä¡äçIJŃåĹřæIJĽ2äÿłçĽĹæIJňçŽĎgrok()åĞ¡æŢřèćńåŁäè¡¡ãĂĆéĂŽåÿÿæİèèŕť'i

åŻăæ■d'ïijŇåIJĺçŤ§äžğçŐŕåćČäÿ■åŘŕèČ¡éIJĂèęĄéĄ£åĚ■éĞ■æŰřåŁäè¡¡æĺąåİŮãĂĆåIJĺäžd'äžŠçŐŕåćČä

10.7 è£ŘèąŇçŽőå¡ŢæĹŰåŐŃçijľæŰĞäžű

éŮőécŸ

æĆĺæIJL'äÿĂäÿłåůšæĹŘéŢ£äÿžåŇĚåŘńåd'ŽäÿłæŰĞäžűçŽĎåžŤçŤĺïijŇåőČåůšè£IJäÿ■åĘ■æŸŕäÿĂäÿłçő

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăçŽĎåžŤçŤĺçĺŃåžŔåůšçżŔæIJL'åd'ŽäÿłæŰĞäžűïijŇä¡ăåŔŕäžěæŁŁä¡ăçŽĎåžŤçŤĺçĺŃåžŔæŤ¿äÿ¿äÿłä¿Ńå■ŘïijŇä¡ăåŔŕäžěåČŔè£ŹæăůåĹŻåžžçŻőå¡ŢïijŽ

```
myapplication/
    spam.py
    bar.py
    grok.py
    __main__.py
```

åęĆædIJ__main__.pyå■ŸåIJĺïijŇä¡ăåŔŕäžěçőĂå■ŢåIJřåIJĺéąűçžğçŻőå¡Ţè£ŘèąŇPythonèğčéĞŁåŹĺïijŽ

```
bash % python3 myapplication
```

èğčéĞŁåŹĺåřĘæL'ğèąŇ__main__.pyæŰĞäžűä¡IJäÿžäÿżçĺŃåžŔãĂĆ

åęĆædIJä¡ăåřĘä¡ăçŽĎäžčçăĄæL'ŞåŇĚæĹŘzipæŰĞäžűïijŇè£Źçğ■æŁĂæIJŕåŘŇæăůäź§éĂĆçŤĺïijŇäÿ¿ă

```
bash % ls
spam.py bar.py grok.py __main__.py
bash % zip -r myapp.zip *.py
bash % python3 myapp.zip
... output from __main__.py ...
```

èőĺèőž

åĹŻåžžäÿĂäÿłçŻőå¡ŢæĹŰzipæŰĞäžűåźűæůžåŁă__main__.pyæŰĞäžűæİěåřĘäÿĂäÿłæŻť'åd'ğçŽĎPytho

çŤśäžŐçŻőå¡ŢåŠŇzipæŰĞäžűäÿŐæ■čåÿÿæŰĞäžűæIJL'äÿĂçĆźäÿ■åŘŇïijŇä¡ăåŔŕèČ¡è£ŸéIJĂèęĄåćđå

```
#!/usr/bin/env python3 /usr/local/bin/myapp.zip
```

10.8 èŕžåŔŰä¡■äžŐåŇĚäÿ■çŽĎæŢřæ■őæŰĞäžű

éŮőécŸ

ä¡ăçŽĎåŇĚäÿ■åŇĚåŘńäžčçăĄéIJĂèęĄåŐżèŕžåŔŰçŽĎæŢřæ■őæŰĞäžűãĂĆä¡ăéIJĂèęĄåř¡åŔŕèČ¡åIJřçŤ

èğčåĘşæŰźæąĹ

åĄĞèő¿ä¡ăçŽĎåŇĚäÿ■çŽĎæŰĞäžűçżĎçżĞæĹŘåęĆäÿŃïijŽ

```
mypackage/
    __init__.py
    somedata.dat
    spam.py
```

çŐřåIJĺåĄĞèő¿spam.pyæŰĞäžűéIJĂèęĄèŕžåŔŰsomedata.datæŰĞäžűäÿ■çŽĎåĘĚåőźãĂĆä¡ääŔŕäžěçŤĺ

```
# spam.py
import pkgutil
data = pkgutil.get_data(__package__, 'somedata.dat')
```

çŤśæ■d'äžğçŤ§çŽĎåŔŸéĞŔæŸŕåŇĚåŘńèřěæŰĞäžűçŽĎåŐ§ågŃåĘĚåőźçŽĎå■ŮèŁĆå■Ůçňęäÿšã ĂĆ

èőĺèőž

èęĄèŕžåŔŰæŢřæ■őæŰĞäžűïijŇä¡ääŔŕèČ¡äijŽåĂ¿åŘŚäžŐçijŰåĘŹä¡£çŤĺåĘĚç¡őçŽĎI/
OåŁ§èČ¡çŽĎäžčçăĄïijŇåęĆopen()ãĂĆä¡ĘæŸŕè£Źçğ■æŰźæşŢäź§æIJL'äÿĂäžŻéŮőécŸãĂĆ

éęŰåĚĹïijŇäÿĂäÿłåŇĚåŕźèğčéĞŁåŹĺçŽĎå¡ŞåL'■åůëä¡IJçŻőå¡ŢåĞääźŐæšąæIJL'æŐğåĹűæİČãĂĆåŻäæ

çňňäžŇïijŇåŇĚéĂŽåÿÿåőL'èčĚä¡IJäÿž.zipæĹŰ.eggæŰĞäžűïijŇè£ŹäžŻæŰĞäžűåźűäÿ■åČŘåIJĺæŰĞäžű

pkgutil.get_data()åĞ¡æŢřæŸŕäÿĂäÿłèŕžåŔŰæŢřæ■őæŰĞäžűçŽĎéńŸçžğåůěåĚůïijŇäÿ■çŤĺçőąåŇĚæŸŕå

get_data()çŽĎçňňäÿĂäÿłåŔĆæŢřæŸŕåŇĚåŘńåŇĚåŘ■çŽĎå■Ůçňęäÿšã ĂĆä¡ääŔŕäžěçŻť'æŐěä¡£çŤĺåŇĚ

10.9 åřĘæŰĞäžűåď źåŁääĚěåĹřsys.path

éŮőécŸ

ä¡ăæŮăæşŢåŕijåĚěä¡ăçŽĎPythonäžčçăĄåŻăäÿžåőČæL'ĂåIJĺçŽĎçŻőå¡Ţäÿ■åIJĺsys.pathéĞŇãĂĆä¡ăæČş

èğčåĘşæŰźæąĹ

æIJL'äÿď'çğ■åÿÿçŤĺçŽĎæŰźåijŔåřĘæŰřçŻőå¡ŢæůžåŁääĹřsys.pathãĂĆçňňäÿĂçğ■ïijŇä¡ääŔŕäžěä¡£çŤĺ

```
bash % env PYTHONPATH=/some/dir:/other/dir python3
Python 3.3.0 (default, Oct 4 2012, 10:17:33)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more␣
↪information.
>>> import sys
>>> sys.path
```

```
['', '/some/dir', '/other/dir', ...]
>>>
```

åIJÍèĞłåőŽäźĽåžŤçŤÍçÍŃåžŔäÿ■ïijŇè£Źæăůç ŽĎçŐŕåćČåŔŸéĞŔåŔŕåIJÍçÍŃåžŔåŘŕåŁÍæŮűèő¿ç¡őæĹŰ

çňňäžŇçğ■æŰźæşŢæŸŕåĹŻåžžäÿĂäÿł.pthæŰĞäžűïijŇåřĘçŽőå¡ŢåĹŮäÿ¿åĞžæÍěïijŇåČŔè£Źæăůïijž

```
# myapplication.pth
/some/dir
/other/dir
```

è£Źäÿł.pthæŰĞäžűéIJĂèęĄæŤ¿åIJÍæ§ŘäÿłPythonçŽĎsite-packagesçŻőå¡ŢïijŇéĂŽåÿÿä¡■äžŐ/usr/local/lib/python3.3/site-packages æĹŰèĂĚ ~/.local/lib/python3.3/sitepackagesãĂĆå¡Şèğčé GŁåŹÍåŘŕåŁÍæŮűïijŇ.pthæŰĞäžűéĞŇåĹŮäÿ¿åĞžæÍěçŽĎå■ŸåIJ

èőÍèőž

æŕŤèţůèť źåŁŽåIJřæĽ¿æŰĞäžűïijŇä¡ăåŔŕèČ¡äijŽåĄ¿åŘŚäžŐåĘŹäÿĂäÿłäžčçăĄæĽŃåŁÍèřČèŁĆsys.path

```
import sys
sys.path.insert(0, '/some/dir')
sys.path.insert(0, '/other/dir')
```

èŹ¡çĎűè£ŹèČ¡âĂIJåůěä¡IJâĂÍïijŇåőČæŸŕåIJÍåőđèůţäÿ■æđĄäÿžèĐĘåijśïijŇážŤår¡éĞŔéĄ£åĚ■ä¡£çŤÍãĂ

```
import sys
from os.path import abspath, join, dirname
sys.path.insert(0, join(abspath(dirname('__file__')), 'src'))
```

è£ŹářĘsrccŻőå¡ŢæůžåŁăăĹřpathéĞŇïijŇåŠŇæĽğèąŇæŔŠåĚèæ■ěéłd'çŽĎäžčçăĄåIJÍåŘŇäÿĂäÿłçŻőå¡Ţ

site-packagesçŻőå¡ŢæŸŕçňňäÿĽæŰźåŇĚåŠŇæÍąåİŮåőĽèčĚçŽĎçŻőå¡ŢãĂĆåęĆæđIJä¡ăæĽŃåŁÍåőĽèč packagesçŻőå¡ŢãĂĆèŹ¡çĎűçŤÍäžŐéĚ■ç¡őpathçŽĎ.pthæŰĞäžűå£ĚéąžæŤ¿ç¡őåIJÍsite-packageséĞŇïijŇä¡ĘåőČéĚ■ç¡őçŽĎèůŕå¿ĎåŔŕäžěæŸŕçşżçż§äÿŁäžżä¡Ţä¡ăåÿŇæIJŻçŽĎçŻőå¡ŢãĂĆåŽăæ■d'

10.10 éĂŽè£Ğå■Ůçňęäÿšå Ř■åŕijåĚěæÍąåİŮ

éŮőécŸ

ä¡ăæČşåŕijåĚěäÿĂäÿłæÍąåİŮïijŇä¡ĘæŸŕæÍąåİŮçŽĎåŘ■å■ŮåIJÍå■ŮçňęäÿšéĞŇãĂĆä¡ăæČşåŕźå■Ůçňęäÿ

èğčåĘşæŰźæąĹ

ä¡£çŤÍimportlib.import_module()åĞ¡æŢřæÍěæĽŃåŁÍåŕijåĚěåŘ■å■Ůäÿžå■ŮçňęäÿšçżźåĞžçŽĎäÿĂäÿłæ

```
>>> import importlib
>>> math = importlib.import_module('math')
>>> math.sin(2)
0.9092974268256817
>>> mod = importlib.import_module('urllib.request')
>>> u = mod.urlopen('http://www.python.org')
>>>
```

import_moduleåŔłæŸŕçőĂå■ŢåIJřæĽ'ğèąŇåŠŇimportçŽÿåŘŇçŽĎæ■ěéłd'ïijŇä¡ĘæŸŕè£ŤåŻđçŤ§æĹŘç

åęĆæđIJä¡ăæ■čåIJĺä¡£çŤĺçŽĎåŇĚïijŇimport_module()äźšåŔŕçŤĺäžŐçŽÿåŕźåŕijåĚěãĂĆä¡ĘæŸŕïijŇä¡äé

```
import importlib
# Same as 'from . import b'
b = importlib.import_module('.b', __package__)
```

èőĺèőž

ä¡£çŤĺimport_module()æĽŃåŁĺåŕijåĚěæĺąåİŮçŽĎéŮőécŸéĂŽåÿÿåĞžçŐřåIJĺäžěæ§Řçğ■æŰźåijŔçijŰå

åIJĺæŮğçŽĎäžčçăĄïijŇæIJL'æŮűä¡ääijŽçIJŃåĹřçŤĺäžŐåŕijåĚěçŽĎåĘĚåžžåĞ¡æŢř__import__()ãĂĆår¡ç éĂŽåÿÿæŻť'åőźæŸŞä¡£çŤĺãĂĆ

èĞłåőŽäźL'åŕijåĚèè£ĞçĺŃçŽĎéńŸçžğåődä¿ŃèğĄ10.11årŔèŁĆ

10.11 éĂŽè£ĞéŠľå■Řè£IJçĺŃåŁäè¡¡æĺąåİŮ

éŮőécŸ

ä¡äæČşèĞłåőŽäźL'PythonçŽĎimportèŕ■åŔëïijŇä¡£å¿ŮåőČèČ¡äžŐè£IJçĺŃæIJžåŹĺäÿŁéİćéĂŔæŸŐçŽĎå

èğčåĘşæŰźæąĹ

éęŰåĚĹèęĄæŔŘåĞžæİěçŽĎæŸŕåőL'åĚĺéŮőécŸãĂĆæIJňåśŁèőĺèőžçŽĎæĂİæČşåęĆæđIJæšąæIJL'äÿĂä äźš§årśæŸŕèŕt'ïijŇæĹŚäžňçŽĎäÿżèęĄçŽőçŽĎæŸŕæůšåĚěåĹĘæđŘPythonçŽĎimportèŕ■åŔëæIJžåĹűãĂĆ åęĆæđIJä¡ăçŘĘèğčäžĘæIJňèŁĆåĘĚéČĺåŐ§çŘęïijŇä¡ăårśèČ¡åd'§äÿžåĚűäžŰäžžä¡ŢçŽőçŽĎèĂŇèĞłåőŽäźL'i æIJL'äžĘè£ŹäžŻïijŇèől'æĹŚäžňçżğçż■åŘŚåL'■èţřãĂĆ

æIJňèŁĆæăÿå£ČæŸŕèő¿èőąåŕijåĚèèŕ■åŔëçŽĎæL'l'åśŢåŁ§èČ¡ãĂĆæIJL'å¿Ĺåd'Žçğ■æŰźæşŢåŘŕäžěåĄŽ äÿ■è£ĞäÿžäžĘæijŤçd'žçŽĎæŰźä¿£ïijŇæĹŚäžňåijĂågŃåĚĹædĎéĂăäÿŃéİćè£ŹäÿłPythonäžčçăĄçżŞædĎïijŽ

```
testcode/
    spam.py
    fib.py
    grok/
```

```
        __init__.py
        blah.py
```

è£ŹäžŽæŰĞäžűçŽĎåĘĚåőźåźűäÿ■éĞ■èęĄïijŇäÿ■è£ĞæĹŚäžňåIJÍæŕŔäÿłæŰĞäżűäÿ■æŤ¿åĚěäžĘåřŚéĞ
è£Źæěůä¡ăåŔŕäžěæţŃèŕŢåőČäžňåźűæ§ěçIJŃå¡ŞåőČäžňèćńåřijåĚěæŮűçŽĎè¿ŞåĞžãĂĆä¿ŃåęĆïijŽ

```
# spam.py
print("I'm spam")

def hello(name):
    print('Hello %s' % name)

# fib.py
print("I'm fib")

def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n-1) + fib(n-2)

# grok/__init__.py
print("I'm grok.__init__")

# grok/blah.py
print("I'm grok.blah")
```

è£ŹéĞŇçŽĎçŽőçŽĎæŸŕåĚĄèőÿè£ŹäžŽæŰĞäżűä¡IJäÿžæÍąåİŮèćńè£IJçÍŃèő£éŮőãĂĆ
äźş§èőÿæIJĂçőĂå■ŢçŽĎæŰźåijŔåřśæŸŕåřĘåőČäžňåŔŚåÿČåĹřäÿĂäÿłwebæIJ■åŁąåŹĺäÿŁéİćãĂĆåIJÍtestcode

```
bash % cd testcode
bash % python3 -m http.server 15000
Serving HTTP on 0.0.0.0 port 15000 ...
```

æIJ■åŁąåŹÍè£ŘèąŇètůæİěåŘŐåĘ■åŘŕåŁĺäÿĂäÿłå■ŢçŃňçŽĎPythonènğčéĞŁåŹÍãĂĆ
çąőä£Ìä¡ăåŔŕäžěä¡£çŤÍ `urllib` èő£éŮőåĹře£IJçÍŃæŰĞäżűãĂĆä¿ŃåęĆïijŽ

```
>>> from urllib.request import urlopen
>>> u = urlopen('http://localhost:15000/fib.py')
>>> data = u.read().decode('utf-8')
>>> print(data)
# fib.py
print("I'm fib")

def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n-1) + fib(n-2)
>>>
```

äžŐè£ŹäÿłæIJ■åŁąåŹĺåŁäè¡¡æžŘäžčçăĄæŸŕæŐěäÿŃæİěæIJňèŁĆçŽĎå§žçąĂãĂĆ äÿžäžĘæŻ£äžčæĽ'ŃåŁĺçŽĎéĂŽè£Ğ urlopen() æİěæŤűéŻĘæžŘæŰĞäžűïijŇ æĹŚäžňéĂŽè£ĞèĞłåőŽäźĽ'importèŕ■åŘěæİěåIJĺåŘŐåŕřèĞłåŁĺåÿőæĹŚäžňåĄŽåĹřãĂĆ

åŁäè¡¡è£IJçĺŃæĺąåİŮçŽĎçňňäÿĂçğ■æŰźæşŢæŸŕåĹŻåžžäÿĂäÿłæŸ¿çd'žçŽĎåŁäè¡¡åĞ¡æŢřæİěåőŇæĹŘ

```
import imp
import urllib.request
import sys

def load_module(url):
    u = urllib.request.urlopen(url)
    source = u.read().decode('utf-8')
    mod = sys.modules.setdefault(url, imp.new_module(url))
    code = compile(source, url, 'exec')
    mod.__file__ = url
    mod.__package__ = ''
    exec(code, mod.__dict__)
    return mod
```

è£ŹäÿłåĞ¡æŢřäijŽäÿŃè¡¡æžŘäžčçăĄïijŇåźűä¡£çŤĺ compile() åřĘåĚűçijŰèŕŚåĹřäÿĂäÿłäžčçăĄåŕźèśąäÿ■ïijŇ çĎűåŘŐåIJĺäÿĂäÿłæŰřåĹŻåžžçŽĎæĺąåİŮåŕźèśąçŽĎå■ŮåĚÿä

```
>>> fib = load_module('http://localhost:15000/fib.py')
I'm fib
>>> fib.fib(10)
89
>>> spam = load_module('http://localhost:15000/spam.py')
I'm spam
>>> spam.hello('Guido')
Hello Guido
>>> fib
<module 'http://localhost:15000/fib.py' from 'http://
↪localhost:15000/fib.py'>
>>> spam
<module 'http://localhost:15000/spam.py' from 'http://
↪localhost:15000/spam.py'>
>>>
```

æ■čåęĆä¡ăæL'ĂèğĄïijŇåŕźäžŐçőĂå■ŢçŽĎæĺąåİŮè£ŹäÿłæŸŕèąŇå¿ŮéĂŽçŽĎãĂĆ äÿ■è£ĞåőČåźűæšąæIJL'åţŇåĚěåĹřéĂŽåÿÿçŽĎimportèŕ■åŘěäÿ■ïijŇåęĆæđIJèęĄæŤŕæŇĄæŻt'énŸçžğçŽĎçż

äÿĂäÿłæŻt'éĚůçŽĎåĄŽæşŢæŸŕåĹŻåžžäÿĂäÿłèĞłåőŽäźĽ'åŕijåĚěåŹĺãĂĆçňňäÿĂçğ■æŰźæşŢæŸŕåĹŻåž

```
# urlimport.py
import sys
import importlib.abc
import imp
from urllib.request import urlopen
from urllib.error import HTTPError, URLError
from html.parser import HTMLParser
```

```
# Debugging
import logging
log = logging.getLogger(__name__)

# Get links from a given URL
def _get_links(url):
    class LinkParser(HTMLParser):
        def handle_starttag(self, tag, attrs):
            if tag == 'a':
                attrs = dict(attrs)
                links.add(attrs.get('href').rstrip('/'))
    links = set()
    try:
        log.debug('Getting links from %s' % url)
        u = urlopen(url)
        parser = LinkParser()
        parser.feed(u.read().decode('utf-8'))
    except Exception as e:
        log.debug('Could not get links. %s', e)
    log.debug('links: %r', links)
    return links

class UrlMetaFinder(importlib.abc.MetaPathFinder):
    def __init__(self, baseurl):
        self._baseurl = baseurl
        self._links = { }
        self._loaders = { baseurl : UrlModuleLoader(baseurl) }

    def find_module(self, fullname, path=None):
        log.debug('find_module: fullname=%r, path=%r', fullname,␣
↪path)
        if path is None:
            baseurl = self._baseurl
        else:
            if not path[0].startswith(self._baseurl):
                return None
            baseurl = path[0]
        parts = fullname.split('.')
        basename = parts[-1]
        log.debug('find_module: baseurl=%r, basename=%r', baseurl,␣
↪basename)

        # Check link cache
        if basename not in self._links:
            self._links[baseurl] = _get_links(baseurl)

        # Check if it's a package
        if basename in self._links[baseurl]:
            log.debug('find_module: trying package %r', fullname)
            fullurl = self._baseurl + '/' + basename
```

```
            # Attempt to load the package (which accesses __init__.
↪py)
            loader = UrlPackageLoader(fullurl)
            try:
                loader.load_module(fullname)
                self._links[fullurl] = _get_links(fullurl)
                self._loaders[fullurl] = UrlModuleLoader(fullurl)
                log.debug('find_module: package %r loaded',␣
↪fullname)
            except ImportError as e:
                log.debug('find_module: package failed. %s', e)
                loader = None
            return loader
        # A normal module
        filename = basename + '.py'
        if filename in self._links[baseurl]:
            log.debug('find_module: module %r found', fullname)
            return self._loaders[baseurl]
        else:
            log.debug('find_module: module %r not found', fullname)
            return None

    def invalidate_caches(self):
        log.debug('invalidating link cache')
        self._links.clear()

# Module Loader for a URL
class UrlModuleLoader(importlib.abc.SourceLoader):
    def __init__(self, baseurl):
        self._baseurl = baseurl
        self._source_cache = {}

    def module_repr(self, module):
        return '<urlmodule %r from %r>' % (module.__name__, module._
↪_file__)

    # Required method
    def load_module(self, fullname):
        code = self.get_code(fullname)
        mod = sys.modules.setdefault(fullname, imp.new_
↪module(fullname))
        mod.__file__ = self.get_filename(fullname)
        mod.__loader__ = self
        mod.__package__ = fullname.rpartition('.')[0]
        exec(code, mod.__dict__)
        return mod

    # Optional extensions
    def get_code(self, fullname):
        src = self.get_source(fullname)
```

```
        return compile(src, self.get_filename(fullname), 'exec')

    def get_data(self, path):
        pass

    def get_filename(self, fullname):
        return self._baseurl + '/' + fullname.split('.')[-1] + '.py'

    def get_source(self, fullname):
        filename = self.get_filename(fullname)
        log.debug('loader: reading %r', filename)
        if filename in self._source_cache:
            log.debug('loader: cached %r', filename)
            return self._source_cache[filename]
        try:
            u = urlopen(filename)
            source = u.read().decode('utf-8')
            log.debug('loader: %r loaded', filename)
            self._source_cache[filename] = source
            return source
        except (HTTPError, URLError) as e:
            log.debug('loader: %r failed. %s', filename, e)
            raise ImportError("Can't load %s" % filename)

    def is_package(self, fullname):
        return False

# Package loader for a URL
class UrlPackageLoader(UrlModuleLoader):
    def load_module(self, fullname):
        mod = super().load_module(fullname)
        mod.__path__ = [ self._baseurl ]
        mod.__package__ = fullname

    def get_filename(self, fullname):
        return self._baseurl + '/' + '__init__.py'

    def is_package(self, fullname):
        return True

# Utility functions for installing/uninstalling the loader
_installed_meta_cache = { }
def install_meta(address):
    if address not in _installed_meta_cache:
        finder = UrlMetaFinder(address)
        _installed_meta_cache[address] = finder
        sys.meta_path.append(finder)
        log.debug('%r installed on sys.meta_path', finder)

def remove_meta(address):
```

```
    if address in _installed_meta_cache:
        finder = _installed_meta_cache.pop(address)
        sys.meta_path.remove(finder)
        log.debug('%r removed from sys.meta_path', finder)
```

äÿŃéÌćæŸŕäÿĂäÿłäžd'äžŠäijŽèŕÌïijŇæijŤçd'žäžĘåęĆä¡Ţä¡£çŤĺĺL'■éÌćçŽĎäżčçăĄïijŽ

```
>>> # importing currently fails
>>> import fib
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>> # Load the importer and retry (it works)
>>> import urlimport
>>> urlimport.install_meta('http://localhost:15000')
>>> import fib
I'm fib
>>> import spam
I'm spam
>>> import grok.blah
I'm grok.__init__
I'm grok.blah
>>> grok.blah.__file__
'http://localhost:15000/grok/blah.py'
>>>
```

è£ŹäÿłçĽźæőŁçŽĎæŰźæąĹäijŽåőĽèčĔäÿĂäÿłçĽźåĹńçŽĎæ§ěæĽ¿åŹÍ UrlMetaFinder åőđä¿ŃïijŇ ä¡IJäÿž sys.meta_path äÿ■æIJĂåŘŐçŽĎåőđä¡ŞãĂĆ å¡ŞæĺąåİŮèćńåŕijåĚěæŮűïijŇäijŽä¿İæ■ő sys.meta_path äÿ■çŽĎæ§ěæĽ¿åŹĺåőŽä¡■æĺąåİŮãĂĆ åIJÍè£Źäÿłä¿Ńå■ŘäÿïijŇUrlMetaFinder åőđä¿ŃæŸŕæIJĂåŘŐäÿĂäÿłæ§ěæĽ¿åŹĺæŰźæąĹïijŇ å¡ŞæĺąåİŮåIJĺäżä¡ŢäÿĂäÿłæŹőéĂŽåIJřæŰźéČ¡æĽ¿äÿ

ä¡IJäÿžåÿÿèğĄçŽĎåőđçŐřæŰźæąĹïijŇUrlMetaFinder çśżåŇĚèčĚåIJĺäÿĂäÿłçŤĺæĹůæŇĞåőŽçŽĎURLäÿŁãĂĆ åIJĺåĘĚéČĺïijŇæ§ěæĽ¿åŹléĂŽè£ĞæŁŞåŔŰæŇĞåđ åŕijåĚěçŽĎæŮűåĂŹïijŇæĺąåİŮåŘ■äijŽèů§åůšæIJĽçŽĎéŞ¿æŐěä¡IJåŕźæŕŤãĂĆåęĆæđIJæĽ¿åĹřäžĘäÿĂäÿłåŇ äÿĂäÿłå■ŢçŃňçŽĎ UrlModuleLoader çśżèćńçŤĺæİěäżŐè£IJçĺŃæIJžåŹĺäÿŁåŁãè¡¡æžŘäžčçăĄåźűåĹŽåżž è£ŹéĞŇçijŞå■ŸéŞ¿æŐěçŽĎäÿĂäÿłåŐ§åŻăæŸŕéĄ£åĚ■äÿ■å£ĚèęĄçŽĎHTTPèŕůæśĆéĞ■åd'■åŕijåĚěãĂĆ

èĞłåőŽäźĽåŕijåĚěçŽĎçňňäžŇçg■æŰźæşŢæŸŕçijŰåĘŹäÿĂäÿłéŠľå■ŘçŻt'æŐěåţŇåĚěåĹř sys.path åŘŸéĞŘäÿ■åŐżïijŇ èŕĘåĹńæ§ŘäżŽçŽőå¡ŢåŚ¡åŘ■æĺąåijŔãĂĆ åIJÍ urlimport.py äÿ■æůżåŁäåęĆäÿŃçŽĎçśżåŠŇæŤŕæŇĄåĞ¡æŢřijŽ

```
# urlimport.py
# ... include previous code above ...
# Path finder class for a URL
class UrlPathFinder(importlib.abc.PathEntryFinder):
    def __init__(self, baseurl):
        self._links = None
        self._loader = UrlModuleLoader(baseurl)
        self._baseurl = baseurl
```

```
    def find_loader(self, fullname):
        log.debug('find_loader: %r', fullname)
        parts = fullname.split('.')
        basename = parts[-1]
        # Check link cache
        if self._links is None:
            self._links = [] # See discussion
            self._links = _get_links(self._baseurl)

        # Check if it's a package
        if basename in self._links:
            log.debug('find_loader: trying package %r', fullname)
            fullurl = self._baseurl + '/' + basename
            # Attempt to load the package (which accesses __init__.
↪py)
            loader = UrlPackageLoader(fullurl)
            try:
                loader.load_module(fullname)
                log.debug('find_loader: package %r loaded',␣
↪fullname)
            except ImportError as e:
                log.debug('find_loader: %r is a namespace package',␣
↪fullname)
                loader = None
            return (loader, [fullurl])

        # A normal module
        filename = basename + '.py'
        if filename in self._links:
            log.debug('find_loader: module %r found', fullname)
            return (self._loader, [])
        else:
            log.debug('find_loader: module %r not found', fullname)
            return (None, [])

    def invalidate_caches(self):
        log.debug('invalidating link cache')
        self._links = None

# Check path to see if it looks like a URL
_url_path_cache = {}
def handle_url(path):
    if path.startswith(('http://', 'https://')):
        log.debug('Handle path? %s. [Yes]', path)
        if path in _url_path_cache:
            finder = _url_path_cache[path]
        else:
            finder = UrlPathFinder(path)
            _url_path_cache[path] = finder
        return finder
```

```
    else:
        log.debug('Handle path? %s. [No]', path)

def install_path_hook():
    sys.path_hooks.append(handle_url)
    sys.path_importer_cache.clear()
    log.debug('Installing handle_url')

def remove_path_hook():
    sys.path_hooks.remove(handle_url)
    sys.path_importer_cache.clear()
    log.debug('Removing handle_url')
```

èęĄä¡£çŤĺè£Źäÿłèůŕå¿Ďæ§ěæL'¿åŹĺïijŇä¡ăåŕłéIJĂèęĄåIJĺ sys.path äÿ■åŁăåĚěURLéŞ¿æŐěãĆĆä¿ŃåęĆïijŽ

```
>>> # Initial import fails
>>> import fib
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'

>>> # Install the path hook
>>> import urlimport
>>> urlimport.install_path_hook()

>>> # Imports still fail (not on path)
>>> import fib
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'

>>> # Add an entry to sys.path and watch it work
>>> import sys
>>> sys.path.append('http://localhost:15000')
>>> import fib
I'm fib
>>> import grok.blah
I'm grok.__init__
I'm grok.blah
>>> grok.blah.__file__
'http://localhost:15000/grok/blah.py'
>>>
```

åĚşéŤőçĆźářśæŸŕ handle_url() åĞ¡æŢřïijŇåőČèćńæůżåŁăåĹřäžĘ sys.path_hooks åŔŸéĞŔäÿ■ãĂĆ å¡Ş sys.path çŽĎåőđä¡Şèćńåď'ĎçŘĘæŮűïijŇäijŽèřČçŤĺ sys.path_hooks äÿ■çŽĎåĞ¡æŢřãĂĆ åęĆæđIJäżżä¡ŢäÿĂäÿłåĞ¡æŢřè£ŤåŽđäžĘäÿĂäÿłæ§ěæL'¿åŹĺářźèśą sys.path åőđä¡ŞåŁăè¡¡æĺąåİŮãĂĆ

è£IJçĺŃæĺąåİŮåŁăè¡¡èů§åĚűäżŰçŽĎåŁăè¡¡ä¡£çŤĺæŰźæşŢåĞääźŐæŸŕäÿĂæăůçŽĎãĂĆä¿ŃåęĆïijŽ

```
>>> fib
<urlmodule 'fib' from 'http://localhost:15000/fib.py'>
>>> fib.__name__
'fib'
>>> fib.__file__
'http://localhost:15000/fib.py'
>>> import inspect
>>> print(inspect.getsource(fib))
# fib.py
print("I'm fib")

def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n-1) + fib(n-2)
>>>
```

èőÍèőž

åIJÍèŕęçżĘèőÍèőžäźŃåĽ'■ïijŇæIJL'çĆźèęĄåijžèřČçŽĎæŸŕïijŇPythonçŽĎæÍąåİŮãĂĄåŇĚåŠŇåŕijåĚěæIJ
å■şä¡£çżŔéłŇäÿřåŕŇçŽĎPythonçÍŃåžŔåŚŸäź§å¿ĹåřŚèČ¡çš¿éĂŽåőČäžňãĂĆ
æĹŚåIJÍè£ŹéĞŇæŐÍè■ŘäÿĂäžŽåĂijçŽĎåŐżèŕżçŽĎæŰĞæąčåŠŇäźęçś■ïijŇåŇĚæŃň im-
portlib module åŠŇ PEP 302. æŰĞæąčåĘĚåőźåIJÍè£ŹéĞŇäÿ■äijŽèćńéĞ■åd'■æŔŘåĹřïijŇäÿ■è£ĞæĹŚåIJÍè£Z

éęŰåĚĹïijŇåęĆædIJä¡ăæČşåĹŽåžžäÿĂäÿłæŰřçŽĎæÍąåİŮåŕźèśąïijŇä¡£çŤÍ imp.
new_module() åĞ¡æŢřïijŽ

```
>>> import imp
>>> m = imp.new_module('spam')
>>> m
<module 'spam'>
>>> m.__name__
'spam'
>>>
```

æÍąåİŮåŕźèśąéĂŽåÿÿæIJL'äÿĂäžŽæIJ§æIJŽåśđæĂğïijŇåŇĚæŃň __file__
ïijÍè£ŘèąŇæÍąåİŮåŁãè¡¡èí■åŘěçŽĎæŰĞäžűåŘ■ïijL' åŠŇ __package__ (åŇĚåŘ■)ãĂĆ

åĚűæňąïijŇæÍąåİŮäijŽèćńèğčéĞŁåŹÍçijŞå■ŸèţůæİěãĂĆæÍąåİŮçijŞå■ŸåŘŕäžěåIJÍå■ŮåĚÿ
sys.modules äÿ■èćńæL'¿åĹřãĂĆ åŽääÿžæIJL'äžĘè£ŹäÿłçijŞå■ŸæIJžåĹűïijŇéĂŽåÿÿåŘŕäžěårĘçijŞå■ŸåŠ

```
>>> import sys
>>> import imp
>>> m = sys.modules.setdefault('spam', imp.new_module('spam'))
>>> m
<module 'spam'>
>>>
```

åęĆæđIJçżŹåőŽæĺąåİŮåůšçżŔå■ŸåIJĺéĆčäźĹářšäijŽçŻť æŐěèŐůå¿ŮåůšçżŕèćńåĹŻåžžè£ĞçŽĎæĺąåİŮï

```
>>> import math
>>> m = sys.modules.setdefault('math', imp.new_module('math'))
>>> m
<module 'math' from '/usr/local/lib/python3.3/lib-dynload/math.so'>
>>> m.sin(2)
0.9092974268256817
>>> m.cos(2)
-0.4161468365471424
>>>
```

çŤśäžŐåĹŻåžžæĺąåİŮå¿ĹçőĂå■ŢïijŇå¿Ĺåőźæŷ§çijŰåĘŹçőĂå■ŢåĞ¡æŢřæŕŤåęĆçňňäÿĂéČĺåĹĘçŽĎ load_module() åĞ¡æŢřãĂĆ è£ŹäÿłæŰźæąĹçŽĎäÿĂäÿłçijžçĆźæŸŕå¿ĹéŽ¿åď ĎçŘĘåď■æİĆæČĚåĘţæŕŤäÿžäžĘåď ĎçŘĘäÿĂäÿłåŇĚïijŇä¡ăèęĄéĞ■æŰřåőđçŐřæŹőéĂŽimportèŕ■åŘěçŽĎåžŢåśĆéĂżè¿ŚïijĹæŕŤåęĆæĽğèąŇéĆčäžŽæŰĞäžűïijŇèő¿ç¡őèůŕå¿Ďç■Ľ'ïijĽãĂĆè£Źäÿłåď ■æİĆæĂğåřśæŸŕäÿžäżĂäźĹæIJĂåě¡çŻť æŐ

æĽľåśŢimportèŕ■åŘěå¿ĹçőĂå■ŢïijŇä¡ĘæŸŕäijŽæIJĽå¿Ĺåď ŽçğżåŁĺæŞ■ä¡IJãĂĆ æIJĂéńŸåśĆäÿŁïijŇåŕijåĚěæŞ■ä¡IJèćńäÿĂäÿłä¡■äžŐsys.meta_pathåĹŮèąĺäÿ■çŽĎâĂIJåĚČèůŕå¿ĎâĂİæ§ěæ åęĆæđIJä¡ăè¿ŞåĞžåőČçŽĎåĂijïijŇäijŽçIJŃåĹřäÿŃéIćè£Źæăůïijž

```
>>> from pprint import pprint
>>> pprint(sys.meta_path)
[<class '_frozen_importlib.BuiltinImporter'>,
<class '_frozen_importlib.FrozenImporter'>,
<class '_frozen_importlib.PathFinder'>]
>>>
```

å¡ŞæĽğèąŇäÿĂäÿłèŕ■åŘěæŕŤåęĆ import fib æŮűïijŇèğčéĞŁåŹĺäijŽéĄ■åŐĘsys.mata_pathäÿ■çŽĎ èřČçŤĺåőČäžňçŽĎ find_module() æŰźæşŢåőŽä¡■æ■čçąőçŽĎæĺąåİŮåŁăè¡¡åŹĺãĂĆ åŘŕäžěéĂŽè£Ğåőđé!ŇæİěçIJŃçIJŃïijž

```
>>> class Finder:
...     def find_module(self, fullname, path):
...         print('Looking for', fullname, path)
...         return None
...
>>> import sys
>>> sys.meta_path.insert(0, Finder()) # Insert as first entry
>>> import math
Looking for math None
>>> import types
Looking for types None
>>> import threading
Looking for threading None
Looking for time None
Looking for traceback None
Looking for linecache None
Looking for tokenize None
Looking for token None
>>>
```

æşĺæĎŕçIJŃ `find_module()` æŰźæşŢæŸŕæĂŐæăůåIJĺæŕŕäÿĂäÿłåŕijåĚěåřśèćńèğęåŕŚçŽĎãĂĆ è£ŹäÿłæŰźæşŢäÿ■çŽĎpathåŕĆæŢřçŽĎä¡IJçŤĺæŸŕåď˙ĎçŘĘåŃĚãĂĆ åď˙ŽäÿłåŃĚèćńåŕijåĚěïijŃåřśæŸŕäÿĂäÿłåŕŕåIJĺåŃĚçŽĎ `__path__` åśđæĂğäÿ■æĽ¿åĹřçŽĎèůŕå¿ĎåĹŮèąĺãĂĆ èęĄæĽ¿åĹřåŃĚçŽĎå■ŘçżĎäžűåřśèęĄæčĂæ§ěè£ŹäžŻèůŕå¿ĎãĂ æŕŤåęĆæşĺæĎŕåŕźäžŐ `xml.etree` åŠŃ `xml.etree.ElementTree` çŽĎèůŕå¿ĎéĚ■ç¡őïijŽ

```
>>> import xml.etree.ElementTree
Looking for xml None
Looking for xml.etree ['/usr/local/lib/python3.3/xml']
Looking for xml.etree.ElementTree ['/usr/local/lib/python3.3/xml/
↪etree']
Looking for warnings None
Looking for contextlib None
Looking for xml.etree.ElementPath ['/usr/local/lib/python3.3/xml/
↪etree']
Looking for _elementtree None
Looking for copy None
Looking for org None
Looking for pyexpat None
Looking for ElementC14N None
>>>
```

åIJĺ `sys.meta_path` äÿŁæ§ěæĽ¿åŹĺçŽĎä¡■ç¡ő å¿ĹéĞ■èęĄïijŃåřĘåőČäžŐéŸ§åď˙t’çğžåĹřéŸ§år¿ïijŃ

```
>>> del sys.meta_path[0]
>>> sys.meta_path.append(Finder())
>>> import urllib.request
>>> import datetime
```

çŐřåIJĺä¡ăçIJŃäÿ■åĹřäžża¡Ţè¿ŞåĞžäžĘïijŃåŻääÿžåŕijåĚèèćńsys.meta_pathäÿ■çŽĎåĚűäžŰåőđä¡Şåď˙Ďç è£ŹæŮűåĂŹïijŃä¡ääŘłæIJL’åIJĺåŕijåĚěäÿ■å■ŸåIJĺæĺąåİŮçŽĎæŮűåĂŹæL’■èČ¡çIJŃåĹřåőČèćńèğęåŔŚïijŽ

```
>>> import fib
Looking for fib None
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>> import xml.superfast
Looking for xml.superfast ['/usr/local/lib/python3.3/xml']
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'xml.superfast'
>>>
```

ä¡ääźŃåL’■åőL’èčĚè£ĞäÿĂäÿłæ■ŢèŐůæIJłç§ěæĺąåİŮçŽĎæ§ěæL’¿åŹĺïijŃè£ŹäÿłæŸŕ `UrlMetaFinder` çśżçŽĎåĚşéŤőãĂĆ äÿĂäÿł `UrlMetaFinder` åőđä¿Ńèćńæůżåŀääĺř `sys.meta_path` çŽĎæIJńår¿ïijŃä¡IJäÿžæIJĂåŘŐäÿĂäÿłæ§ěæL’¿åŹĺæŰźæąĹãĂĆ åęĆæđIJèćńèřůæśĆçŽĎæĺąåİŮåŘ■äÿ■èČ¡åőŽä¡■ïijŃåřśäijŽèćńè£Źäÿłæ§ěæL’¿åŹĺåď˙ĎçŘĘæŐL’ãĂĆ åď˙ĎçŘĘåŃĚçŽĎæŮűåĂŹéIJĂèęĄæşĺæĎŔïijŃåIJĺpathåŔĆæŢřäÿ■æŇĞåőŽçŽĎåĂijéIJĂèęĄèćńæčĂæ§ěïijŃ åęĆæđIJäÿ■æŸŕïijŃèřěå■ŘæĺąåİŮå£ĚéąžåĄŠåśđäžŐåĚűäžŰæ§ěæL’¿åŹĺåźűèćńå£¡çŢěæŐL’ãĂĆ

åŕźäžŐåŇĚçŽĎåĚűäžŰåď˙ĎçŘĘåŔŕåIJĺ UrlPackageLoader çśżäÿ■èćńæĽ¿åĹřãĂĆ è£Źäÿłçśżäÿ■äijŽåŕijåĚěåŇĚåŘ■ïijŇèĂŇæŸŕåŐżåŁãè¡¡åŕźåžŤçŽĎ __init__.py æŰĞäżűãĂĆ åőČäź§äijŽèő¿ç¡őæĺąåİŮçŽĎ __path__ åśđæĂğïijŇè£ŹäÿĂæ■ěå¿ĹéĞ■èęĄïijŇ åŻăäÿžåIJĺåŁãè¡¡åŇĚçŽĎå■ŘæĺąåİŮæŮűè£ŹäÿłåĂijäijŽèćńäijäçżŹåfind_module() èřČçŤĺãĂĆ å§žäžŐèůŕå¿ĎçŽĎåŕijåĚěéŠĽ˙å■ŘæŸŕè£ŹäžŽæĂİæČşçŽĎäÿĂäÿłæĽĽ˙åśŢïijŇæĹŚäžňéČ¡ç§ěéĄŞïijŇsys.path æŸŕäÿĂäÿłPythonæ§ěæĽ¿æĺąåİŮçŽĎçŻő¡ŢåĹŮèąĺïijŇä¿ŃåęĆïijŽ

```
>>> from pprint import pprint
>>> import sys
>>> pprint(sys.path)
['',
'/usr/local/lib/python33.zip',
'/usr/local/lib/python3.3',
'/usr/local/lib/python3.3/plat-darwin',
'/usr/local/lib/python3.3/lib-dynload',
'/usr/local/lib/...3.3/site-packages']
>>>
```

åIJĺ sys.path äÿ■çŽĎærŔäÿĂäÿłåőđä¡ŞéČ¡äijŽèćńécĺåď˙ŰçŽĎçżŚåőŽåĹřäÿĂäÿłæ§ěæĽ¿åŹĺåŕźèšąäÿä¡ăåŔŕäžěéĂŽè£Ğæ§ěçIJŃ sys.path_importer_cache åŐżçIJŃäÿŃè£ŹäžŽæ§ěæĽ¿åŹĺïijŽ

```
>>> pprint(sys.path_importer_cache)
{'.': FileFinder('.'),
'/usr/local/lib/python3.3': FileFinder('/usr/local/lib/python3.3'),
'/usr/local/lib/python3.3/': FileFinder('/usr/local/lib/python3.3/
↪'),
'/usr/local/lib/python3.3/collections': FileFinder('...python3.3/
↪collections'),
'/usr/local/lib/python3.3/encodings': FileFinder('...python3.3/
↪encodings'),
'/usr/local/lib/python3.3/lib-dynload': FileFinder('...python3.3/
↪lib-dynload'),
'/usr/local/lib/python3.3/plat-darwin': FileFinder('...python3.3/
↪plat-darwin'),
'/usr/local/lib/python3.3/site-packages': FileFinder('...python3.3/
↪site-packages'),
'/usr/local/lib/python33.zip': None}
>>>
```

sys.path_importer_cache æŕŤ sys.path äijŽæŻt˙åď˙ğçĆźïijŇ åŻăäÿžåőČäijŽäÿžæĽĂæIJĽèćńåŁãè¡¡äžčçăĄçŽĎçŽőå¡Ţèőŕå¡ŢåőČäžňçŽĎæ§ěæĽ¿åŹĺãĂĆ è£ŹåŇĚæŃňåŇĚçŽĎå■ŘçŽőå¡ŢïijŇè£ŹäžŽéĂŽåÿÿåIJĺ sys.path äÿ■æŸŕäÿ■å■ŸåIJĺçŽĎãĂĆ

èęĄæĽ˙ğèąŇ import fib ïijŇäijŽéąžåžŔæčĂæ§ě sys.path äÿ■çŽĎçŽőå¡ŢãĂĆ åŕźäžŐærŔäÿłçŽőå¡ŢïijŇåŘ■çğŕâĂIJfibâĂİäijŽèćńäijäçżŹçŻÿåžŢçŽĎ sys.path_importer_cache äÿ■çŽĎæ§ěæĽ¿åŹĺãĂĆ è£ŹäÿłåŔŕäžěèöľä¡ăåĹŻåžžèĞłåůśçŽĎæ§ěæĽ¿åŹĺåźűå

```
>>> class Finder:
... def find_loader(self, name):
```

```
...     print('Looking for', name)
...     return (None, [])
...
>>> import sys
>>> # Add a "debug" entry to the importer cache
>>> sys.path_importer_cache['debug'] = Finder()
>>> # Add a "debug" directory to sys.path
>>> sys.path.insert(0, 'debug')
>>> import threading
Looking for threading
Looking for time
Looking for traceback
Looking for linecache
Looking for tokenize
Looking for token
>>>
```

åIJĺè£ŹéĞŇïijŇä¡ăåŔŕäžëäÿžåŘ■å■ŮâĂIJdebugâĂİåĹŻåžžäÿĂäÿłæŰřçŽĎçijŞå■Ÿåőđä¡Şåźűář€åőČèő¿
sys.path äÿŁçŽĎçňňäÿĂäÿłãĂĆ åIJĺæĽ'ĂæIJL'æŐěäÿŃæİěçŽĎåŕijåĚěäÿ■ïijŇä¡ăäijŽçIJŃåĹřä¡ăçŽĎæ§ěa
äÿ■è£ĞïijŇçŤśäžŐåőČè£ŤåŻđ (None, [])ïijŇéĆčäźĹåđ'ĎçŘĘè£ŻçÍŃäijŽçżğçż■åđ'ĎçŘĘäÿŃäÿĂäÿłåőđä¡ŞãA

sys.path_importer_cache çŽĎä¡£çŤĺèćńäÿĂäÿłå■ŸåĆĺåIJĺ sys.path_hooks
äÿ■çŽĎåĞ¡æŢřåĹŮèąĺæŐğåĹűãĂĆ èŕŢèŕŢäÿŃéİćçŽĎä¿Ńå■ŘïijŇåőČäijŽæÿĚéŹđ'çijŞå■Ÿåźűçżź
sys.path_hooks æůżåŁääÿĂäÿłæŰřçŽĎèůŕå¿ĎæčÄæ§ěåĞ¡æŢř

```
>>> sys.path_importer_cache.clear()
>>> def check_path(path):
...     print('Checking', path)
...     raise ImportError()
...
>>> sys.path_hooks.insert(0, check_path)
>>> import fib
Checked debug
Checking .
Checking /usr/local/lib/python33.zip
Checking /usr/local/lib/python3.3
Checking /usr/local/lib/python3.3/plat-darwin
Checking /usr/local/lib/python3.3/lib-dynload
Checking /Users/beazley/.local/lib/python3.3/site-packages
Checking /usr/local/lib/python3.3/site-packages
Looking for fib
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>>
```

æ■čåęĆä¡ăæĽ'ĂèğĄïijŇcheck_path() åĞ¡æŢřèćńæŕŔäÿł sys.path
äÿ■çŽĎåőđä¡ŞěřČçŤĺãĂĆ äÿ■éą¿ïijŇçŤśäžŐæŁŻåĞžäžĘ ImportError åijĆåÿÿïijŇ
åŢěéČ¡äÿ■äijŽåŔŠçŤ§äžĘïijĹäžĚäžĚåřĘæčÄæ§ěè¡ńçğżåĹřsys.path_hooksçŽĎäÿŃäÿĂäÿłåĞ¡æŢřïijL'ãĂĆ

ç§ěéĄŞäžĘæĂŐæăůsys.pathæŸŕæĂŐæăůèćńåđ'ĎçŘĘçŽĎïijŇä¡ăåŕśèČ¡æđĎåžžäÿĂäÿłèĞłåőŽäźL'èůŕå¿

```
>>> def check_url(path):
...     if path.startswith('http://'):
...         return Finder()
...     else:
...         raise ImportError()
...
>>> sys.path.append('http://localhost:15000')
>>> sys.path_hooks[0] = check_url
>>> import fib
Looking for fib # Finder output!
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'

>>> # Notice installation of Finder in sys.path_importer_cache
>>> sys.path_importer_cache['http://localhost:15000']
<__main__.Finder object at 0x10064c850>
>>>
```

è£ŹåřśæŸŕæIJňèŁĆæIJĂåŘŐéČĺåĹĘçŽĎåĚşéŤőçĆźãĂĆäžŃåðđäÿŁïijŇäÿĂäÿłçŤĺæİěåIJĺsys.pathäÿ■æ
å¡ŞåőČäżňèćńçćřåĹřçŽĎæŮűåĂŹïijŇäÿĂäÿłæŰřçŽĎ `UrlPathFinder`
åðđä¿ŃèćńåĹŻåżžåźűèćńæŤ¿åĚě `sys.path_importer_cache`.
äźŃåŘŐïijŇæĽĂæIJĽéIJĂèęĄæčĂæ§ě `sys.path` çŽĎåŕijåĚěèŕ■åŔéČ¡äijŽä¡£çŤĺä¡ăçŽĎèĞłåőŽäźĽæ§ě

å§žäžŐèůŕå¿ĎåŕijåĚěçŽĎåŇĚåd'ĎçŘEçĺ■å¿őæIJĽçĆźåd'■æİĆïijŇåźűäÿŤèů§
`find_loader()` æŰźæşŢè£ŤåŻđåĂijæIJĽåĚşãĂĆ åŕźäžŐçőĂå■ŢæĺąåİŮïijŇ`find_loader()`
è£ŤåŻđäÿĂäÿłåĚČçżĎ(loader, None)ïijŇ åĚűäÿ■çŽĎload-
eræŸŕäÿĂäÿłçŤĺäžŐåŕijåĚěæĺąåİŮçŽĎåŁăè¡¡åŹĺåőđä¿ŃãĂĆ

åŕźäžŐäÿĂäÿłæŹőéĂŽçŽĎåŇĚïijŇ`find_loader()` è£ŤåŻđäÿĂäÿłåĚČçżĎ(loader,
path)ïijŇ åĚűäÿ■çŽĎloaderæŸŕäÿĂäÿłçŤĺäžŐåŕijåĚěåŇĚïijĹåźűæĽğèąŇ__init__.pyïijĽçŽĎåŁăè¡¡åŹĺåőđä¿
pathæŸŕäÿĂäÿłäijŽåĹİåğŃåŇŰåŇĚçŽĎ `__path__` åśđæĂğçŽĎçŽőå¡ŢåĹŮèąĺãĂĆ
ä¿ŃåęĆïijŇåęĆæđIJå§žçąĂURLæŸŕ http://localhost:15000 åźűäŸŤäÿĂäÿłçŤĺæĹůæĽğèąŇ
`import grok`, éĆčäźĹ `find_loader()` è£ŤåŻđçŽĎpathåřśäijŽæŸŕ ['http://localhost:
15000/grok']

`find_loader()` è£ŸèęĄèČ¡åđ'ĎçŘEäÿĂäÿłåŚ¡åŘ■çľžéŮť'åŇĚãĂĆ
äÿĂäÿłåŚ¡åŘ■çľžéŮť'åŇĚäÿ■æIJĽäÿĂäÿłåŘĹæşŢçŽĎåŇĚçŽőå¡ŢåŘ■ïijŇä¡ĘæŸŕäÿ■å■ŸåIJĺ__init__.pyæŰ
è£ŹæăůçŽĎèŕİïijŇ`find_loader()` å£ĚéążèŁŤåŻđäÿĂäÿłåĚČçżĎ(None, path)ïijŇ
pathæŸŕäÿĂäÿłçŽőå¡ŢåĹŮèąĺïijŇçŤśåőČæİěæđĎåžžåŇĚçŽĎåőŽäźĽæIJĽ__init__.pyæŰĞäžűçŽĎ__path__
åŕźäžŐè£Źçğ■æČĚåĘţïijŇåŕijåĚěæIJžåĹűäijŽçżğçż■åĽ■èąŇåŐżæčĂæ§ěsys.pathäÿ■çŽĎçŽőå¡ŢãĂĆ
åęĆæđIJæĽ¿åĹŕäžĘåŚ¡åŘ■çľžéŮť'åŇĚïijŇæĽĂæIJĽçŽĎçżŞæđIJèůŕå¿ĎèćńåŁăåĹŕäÿĂèţůæİěæđĎåžžæIJĂ
åĚşäžŐåŚ¡åŘ■çľžéŮť'åŇĚçŽĎæŻť'åd'Žä£qæĄŕèŕůåŔĆèĂČ10.5åřŔèŁĆãĂĆ

æĽĂæIJĽçŽĎåŇĚéČ¡åŇĚåŘńäžĘäÿĂäÿłåĘĚéČĺèůŕå¿Ďèő¿ç¡őïijŇåŔŕäžěåIJĺ__path__åśđæĂğäÿ■çIJŃ

```
>>> import xml.etree.ElementTree
>>> xml.__path__
['/usr/local/lib/python3.3/xml']
>>> xml.etree.__path__
['/usr/local/lib/python3.3/xml/etree']
```

```
>>>
```

äźŃåĽ■æŔŘåĹřïijŇ__path__çŽĎèő¿ç¡őæŸŕéĂŽè£Ğ find_loader() æŰźæşŢè£ŤåŻđåĂijæŐğåĹűçŽĎãĂĆ äÿ■è£ĞïijŇ__path__æŐëäÿŃæİëäź§èćńsys.path_hooksäÿ■çŽĎåĞ¡æŢåŻăæ■d'ïijŇä¡EąĂŇĚçŽĎå■ŘçżĎäžűèćńåŁăè¡¡åŘŐïijŇä¡■äžŐ__path__äÿ■çŽĎåőđä¡ŞäijŽèćń handle_url() åĞ¡æŢřæčĂæ§ěãĂĆ è£ŹäijŽåŕijèĞt'æŰřçŽĎ UrlPathFinder åőđä¿ŃèćńåĹŻåžžåźűäÿŤèćńåŁăåĚěåĹř sys.path_importer_cache äÿ■ãĂĆ

è£ŸæIJL'äÿłéŽ¿çĆźåřśæŸŕ handle_url() åĞ¡æŢřäžěåŘŁåőČèů§åĘĚéČĺä¡£çŤĺçŽĎ _get_links() åĞ¡æŢřäźŃéŮt'çŽĎäžd'äžŠãĂĆ åęĆædIJä¡äçŽĎæ§ěæL'¿åŹĺåőđçŐŕéIJĂèęĄä¡£çŤĺåĹřåĚű æIJL'åŘŕèČ¡è£ŹäžŻæĺąåİŮäijŽåIJĺæ§ěæL'¿åŹĺæŞ■ä¡IJæIJ§éŮt'è£ŻèąŇæŻt'åd'ŽçŽĎåŕijåĚěãĂĆ åőČåŘŕäžěåŕijèĞt' handle_url() åŠŇåĚűäžŰæ§ěæL'¿åŹĺéČĺåĹĘéŹůåĚěäÿĂçg■éĂŠå¡Šå¿łçŐŕçŁűæĂĄã äÿžäžĘèğčéĞŁè£Źçg■åŘŕèČ¡æĂğïijŇåődçŐřäÿ■æIJL'äÿĂäÿłèćńåĹŻåžžçŽĎæ§ěæL'¿åŹĺçijŞå■ŸïijĹæŕŘäÿĂ åőČåŘŕäžěéĄ£åĚ■åĹŻåžžéĞ■åd'■æ§ěæL'¿åŹĺçŽĎéŮőéćŸãĂĆ åŘęåd'ŰïijŇäÿŃéİćçŽĎäžčçăĄçL'ĞæőţåŘŕäžěçąőä£İæ§ěæL'¿åŹĺäÿ■äijŽåIJĺåĹİåğŃåŇŰéŞ¿æŐěéŻĘåŘĹçŽĎ

```
# Check link cache
if self._links is None:
    self._links = [] # See discussion
    self._links = _get_links(self._baseurl)
```

æIJĂåŘŐïijŇæ§ěæL'¿åŹĺçŽĎ invalidate_caches() æŰźæşŢæŸŕäÿĂäÿłåůěåĚůæŰźæşŢïijŇçŤĺæİěæÿĚçŘĘåĘĚéČĺçijŞå■ŸãĂĆ è£ŹäÿłæŰźæşŢåĘ■çŤĺæĹůèřČçŤĺ importlib.invalidate_caches() çŽĎæŮűåĂŽèćńèğęåŔŠãĂĆ åęĆædIJä¡äæČşèől'URLåŕijåĚěèĂĚéĞ■æŰřèŕžåŘŰéŞ¿æŐěåĹŮèąĺçŽĎèŕĺåŘŕä

åŕźæŕŤäÿŃäÿd'çğ■æŰźæąĹïijĹä£őæŤźsys.meta_pathæĹŰä¡£çŤĺäÿĂäÿłèůŕå¿ĎéŠl'å■ŘïijL'ãĂĆ ä¡£çŤĺsys.meta_pathçŽĎåŕijåĚěèĂĚåŘŕäžěæŇL'çĚğèĞłåůśçŽĎéIJĂèęĄèĞłçŤšåd'ĎçŘĘæĺąåİŮãĂĆ ä¿ŃåęĆïijŇåőČäžňåŘŕäžěäžŐæŢřæ■őåžŞäÿ■åŕijåĚěæĹŰäžěäÿ■åŘŇäžŐäÿĂèĹňæĺąåİŮ/åŇĚåd'ĎçŘĘæŰźåi è£ŹçğèĞłçŤšåŘŇæăůæĎŘåŠşçİĂåŕijåĚěèĂĚéIJĂèęĄèĞłåůśè£ŻèąŇåĘĚéČĺçŽĎäÿĂäžŽçőąçŘĘãĂĆ åŘęåd'ŰïijŇå§žäžŐèůŕå¿ĎçŽĎéŠl'å■ŘåŘłæŸŕéĂĆçŤĺäžŐåŕźsys.pathçŽĎåd'ĎçŘĘãĂĆ éĂŽè£Ğè£ŹçğæL'l'åśŢåŁăè¡¡çŽĎæĺąåİŮèů§æŹőéĂŽæŰźåijŔåŁăè¡¡çŽĎçL'źæĂğæŸŕäÿĂæăůçŽĎãĂĆ

åęĆædIJåĹřçŐřåIJĺäÿžæ■ćä¡ăè£ŸæŸŕäÿ■æŸŕå¿ĹæŸŐçŹ¡ïijŇéĆčäźĹåŘŕäžěéĂŽè£ĞåćđåŁăäÿĂäžŽæŮ

```
>>> import logging
>>> logging.basicConfig(level=logging.DEBUG)
>>> import urlimport
>>> urlimport.install_path_hook()
DEBUG:urlimport:Installing handle_url
>>> import fib
DEBUG:urlimport:Handle path? /usr/local/lib/python33.zip. [No]
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>> import sys
>>> sys.path.append('http://localhost:15000')
>>> import fib
DEBUG:urlimport:Handle path? http://localhost:15000. [Yes]
DEBUG:urlimport:Getting links from http://localhost:15000
DEBUG:urlimport:links: {'spam.py', 'fib.py', 'grok'}
```

```
DEBUG:urlimport:find_loader: 'fib'
DEBUG:urlimport:find_loader: module 'fib' found
DEBUG:urlimport:loader: reading 'http://localhost:15000/fib.py'
DEBUG:urlimport:loader: 'http://localhost:15000/fib.py' loaded
I'm fib
>>>
```

æIJĂåŘŐïijŇåžžèőőä¡ăèŁśçĆźæŮűéŮt'çIJŃçIJŃ PEP 302 äżěåŘŁimportlibçŽĎæŰĞæąčãĂĆ

10.12 åŕijåĚěæĺąåİŮçŽĎåŘŇæŮűä£őæŤźæĺąåİŮ

éŮőécŸ

ä¡ăæČşçżŹæ§Řäÿłåůšå■ŸåIJĺæĺąåİŮäÿ■çŽĎåĞ¡æŢřæůżåŁăèčĚéěřåŹĺãĂĆ äÿ■è£ĞïijŇåĽ■æŔŘæŸŕè£ŹäÿłæĺąåİŮåůšçżŔèćńåŕijåĚěåźűäÿŤèćńä¡£çŤĺè£ĞãĂĆ

èğčåĘşæŰźæąĹ

è£ŹéĞŇéŮőécŸçŽĎæIJňèťĺåřśæŸŕä¡ăæČşåIJĺæĺąåİŮèćńåŁăè¡¡æŮűæĽğèąŇæ§ŘäÿłåŁĺä¡IJãĂĆ åŔŕèČ¡æŸŕä¡ăæČşåIJĺäÿĂäÿłæĺąåİŮèćńåŁăè¡¡æŮűèğęåŔŚæ§ŘäÿłåŻđèřČåĞ¡æŢřæİěéĂŽç§ëä¡ããĂĆ

è£ŹäÿłéŮőécŸåŔŕäžëä¡£çŤĺ10.11åřŔèŁĆäÿ■åŘŇæăůçŽĎåŕijåĚěéŠľå■ŘæIJžåĹűæİěåőđçŐřãĂĆäÿŃéİć

```
# postimport.py
import importlib
import sys
from collections import defaultdict

_post_import_hooks = defaultdict(list)

class PostImportFinder:
    def __init__(self):
        self._skip = set()

    def find_module(self, fullname, path=None):
        if fullname in self._skip:
            return None
        self._skip.add(fullname)
        return PostImportLoader(self)

class PostImportLoader:
    def __init__(self, finder):
        self._finder = finder

    def load_module(self, fullname):
```

```
        importlib.import_module(fullname)
        module = sys.modules[fullname]
        for func in _post_import_hooks[fullname]:
            func(module)
        self._finder._skip.remove(fullname)
        return module

def when_imported(fullname):
    def decorate(func):
        if fullname in sys.modules:
            func(sys.modules[fullname])
        else:
            _post_import_hooks[fullname].append(func)
        return func
    return decorate

sys.meta_path.insert(0, PostImportFinder())
```

è£ŹæăůïijŇä¡ăářšåŕŕäžěä¡£çŤĺ `when_imported()` èčĚéěřåŹĺäžĘïijŇä¿ŃåęĆïijŽ

```
>>> from postimport import when_imported
>>> @when_imported('threading')
... def warn_threads(mod):
...     print('Threads? Are you crazy?')
...
>>>
>>> import threading
Threads? Are you crazy?
>>>
```

ä¡IJäÿžäÿĂäÿłæŻť'åőđéŹĚçŽĎä¿Ńå■ŘïijŇä¡ăåŕŕèČ¡æČşåIJĺåůšå■ŸåIJĺçŽĎåőŽäźĽ'äÿŁéÍćæůżåŁăèčĚé

```
from functools import wraps
from postimport import when_imported

def logged(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        print('Calling', func.__name__, args, kwargs)
        return func(*args, **kwargs)
    return wrapper

# Example
@when_imported('math')
def add_logging(mod):
    mod.cos = logged(mod.cos)
    mod.sin = logged(mod.sin)
```

èőĺèőž

æIJňèŁĆæŁĂæIJŕä¿ÌèţŰäžŐ10.11åřŔèŁĆäý■èőšè£řè£ĞçŽĎåŕijåĚééŠľå■ŘïijŇåźűçĺ■ä¡IJä£őæŤźãĂĆ

`@when_imported` èčĚéěřåŹĺçŽĎä¡IJçŤĺæŸŕæşĺåĘŇåIJĺåŕijåĚěæŮűèćńæ£Ăæť żçŽĎåď ĎçŘĘåŹĺåĞ¡æèŕěèčĚéěřåŹĺæčĂæ§ěsys.modulesæÌěæ§ěçIJŃæĺąåÌŮæŸŕåŘęçIJ§çŽĎåůšçżŔèćńåŁäè¡¡äžĘãĂĆ åęĆæđIJæŸŕçŽĎèŕÍïijŇèŕěåď ĎçŘĘåŹĺèćńçńŇå■şèŕĈçŤĺãĂĆäý■çĎűïijŇåď ĎçŘĘåŹĺèćńæůżåŁäåĽř `_post_import_hooks` å■ŮåĚÿäý■çŽĎäýĂäýłåĹŮèąĺäý■åŐżãĂĆ `_post_import_hooks` çŽĎä¡IJçŤĺåřśæŸŕæŤűéŽĘæĽĂæIJĽçŽĎäýžæŕŔäýłæĺąåÌŮæşĺåĘŇçŽĎåď ĎçŘĘäýĂäýłæĺąåÌŮåŔŕäžěæşĺåĘŇåď Žäýłåď ĎçŘĘåŹĺãĂĆ

èęĄèőľ æĺąåÌŮåŕijåĚěåŘŐèğęåŔŚæůżåŁäçŽĎåŁĺä¡IJïijŇ`PostImportFinder` çśżèćńèő¿ç¡őäýžsys.meta_pathçňňäýĂäýłåĚČçť ããĂĆ åőČäijŽæ■ŢèŐůæĽĂæIJĽæĺąåÌŮåŕijåĚěæŞ■ä¡IJãĂĆ

æIJňèŁĆäý■çŽĎ `PostImportFinder` çŽĎä¡IJçŤĺåźűäý■æŸŕåŁäè¡¡æĺąåÌŮïijŇèĂŇæŸŕèĞłåýęåŕijåĚ åőđéŹĚçŽĎåŕijåĚěèćńåğŤæť ¿çžŹä¡■äžŐsys.meta_pathäý■çŽĎåĚűäžŮæ§ěæĽ¿åŹĺãĂĆ `PostImportLoader` çśżäý■çŽĎ `imp.import_module()` åĞ¡æŢřèćńéĂŠå¡ŞçŽĎèřČçŤĺãĂĆ äýžäžĘéĄ£åĚ■éŹůåĚěæŮăçž£å¿łçŐŕïijŇ`PostImportFinder` ä£ÌæŇĄäžĘäýĂäýłæĽĂæIJĽèćńåŁäè¡¡è£ĞçŽĎæĺąåÌŮéŽĘåŘĹãĂĆ åęĆæđIJäýĂäýłæĺąåÌŮåŘ■å■ŸåIJĺåřśäijŽçŻť æŐěèćńå£¡çŢěæŐĽãĂĆ

å¡ŞäýĂäýłæĺąåÌŮèćń `imp.import_module()` åŁäè¡¡åŘŐïijŇ æĽĂæIJĽåIJĺ_post_import_hooksèćńæşĺåĘŇçŽĎåď ĎçŘĘåŹĺèćńèřČçŤĺïijŇä¡£çŤĺæŰřåŁäè¡¡æĺąåÌŮä¡IJäýžä

æIJĽäýĂçĆźéIJĂèęĄæşĺæĎŔçŽĎæŸŕæIJňæIJžäý■éĂĆçŤĺäžŐéĆčäžŹéĂŽè£Ğ `imp.reload()` èćńæŸ¿åijŔåŁäè¡¡çŽĎæĺąåÌŮãĂĆ äź§åřśæŸŕèŕťïijŇåęĆæđIJä¡ääŁäè¡¡äýĂäýłäźŇåĽ■åůšèćńåŁä åŔęåď ŰïijŇèęĄæŸŕä¡ääžŐsys.modulesäý■åĽäéŹď æĺąåÌŮçĎűåŘŐåĘ■éĞ■æŰřåŕijåĚëijŇåď ĎçŘĘåŹĺåŔĹä

æŽť åď ŽåĚşäžŐåŕijåĚěåŘŐéŠľå■Ră£ąæĄŕèřůåŔĆèĂČ PEP 369.

10.13 åőĽèčĚçğĄæIJĽçŽĎåŇĚ

éŮőécŸ

ä¡ăæČşèęĄåőĽèčĚäýĂäýłçňňäýĽæŰźåŇĚïijŇä¡ĘæŸŕæšąæIJĽæİČéŹŘårĘåőČåőĽèčĚåĹřçşżçż§Python æĹŰèĂĚïijŇä¡ääŔŕèČ¡æČşèęĄåőĽèčĚäýĂäýłä¿ŻèĞłåůśä¡£çŤĺçŽĎåŇĚïijŇèĂŇäý■æŸŕçşżçż§äýŁéÍćæĽĂä

èğčåĘşæŰźæąĹ

PythonæIJĽäýĂäýłçŤĺæĹůåőĽèčĚçŽőå¡ŢïijŇéĂŽåÿÿçśżäijij”~/.local/lib/python3.3/site-packages”ãĂĆ èęĄäijžåĹűåIJĺè£ŹäýłçŽőå¡Ţäý■åőĽèčĚåŇĚïijŇåŔŕä¡£çŤĺåőĽèčĚéĂĽéąźâĂIJ–userâĂİãĂĆä¿ŃåęĆïijŽ

```
python3 setup.py install --user
```

æĹŰèĂĚ

```
pip install --user packagename
```

åIJĺsys.pathäÿ■çŤĺæĹůçŽĎâĂIJsite-packagesâĂİçŽőå¡Ţä¡■äžŐçşżçż§çŽĎâĂIJsite-packagesâĂİçŽőå¡Ţäźŕ̊åL'■ãĂĆ åŻäæ■d'ïijŇä¡ăåőL'èčĚåIJĺéĞŇéİćçŽĎåŇĚåřśæŕŤçşżçż§åůšåőL'èčĚçŽĎåŇïijĹåř¡çőąåźűäÿ■æĂżæŸŕè£ŹæăůïijŇèęĄåŔŰåĘşäžŐçňňäÿL'æŰźåŇĚçőąçŘĘåŹĺïijŇæŕŤåęĆdistributeæĹŰpi

èőĺèőž

éĂŽåÿÿåŇĚäijŽèćńåőL'èčĚåĹřçşżçż§çŽĎsite-packagesçŻőå¡Ţäÿ■åŐżïijŇèůŕå¿ĎçśżäijijâĂIJ/usr/local/lpackagesâĂİãĂĆ äÿ■è£ĞïijŇè£ŹæăůåĄŽéIJĂèęĄæIJL'çőąçŘĘåŚŸæİČéŹŘåźűäÿŤä¡£çŤĺsudoåŚ¡äžd'ãĂĆ åřśçőŰä¡ăæIJL'è£ŹæăůçŽĎæİČéŹŘåŐżæL'ğèąŇåŚ¡äžd'ïijŇä¡£çŤĺsudoåŐżåőL'èčĚäÿĂäÿłæŰřçŽĎïijŇåŘřèČ

åőL'èčĚåŇĚåĹřçŤĺæĹůçŻőå¡Ţäÿ■éĂŽåÿÿæŸŕäÿĂäÿłæIJL'æŢĹçŽĎæŰźæąĹïijŇåőČåĚĄèőÿä¡ăåĹŽåžžä

åŘęåd'ŰïijŇä¡ăè£ŸåŘŕäžěåĹŽåžžäÿĂäÿłèŹŽæŃ§çŐŕåćČïijŇè£ŹäÿłæĹŚäžňåIJĺäÿŇäÿĂèŁĆäijŽèőšåĹřa

10.14 åĹŽåžžæŰřçŽĎPythonçŐŕåćČ

éŮőécŸ

ä¡ăæČşåĹŽåžžäÿĂäÿłæŰřçŽĎPythonçŐŕåćČïijŇçŤĺæİěåőL'èčĚæĺąåİŮåŠŇåŇĚãĂĆ äÿ■è£ĞïijŇä¡ăäÿ■æČşåőL'èčĚäÿĂäÿłæŰřçŽĎPythonåĚŃéŽĘïijŇäź§äÿ■æČşåŕźçşżçż§PythonçŐŕåćČäžğçŤ§

èğčåĘşæŰźæąĹ

ä¡ăåŘŕäžěä¡£çŤĺ `pyvenv` åŚ¡äžd'åĹŽåžžäÿĂäÿłæŰřçŽĎâĂIJèŹŽæŃ§âĂİçŐŕåćČãĂĆ è£ŹäÿłåŚ¡äžd'èćńåőL'èčĚåIJĺPythonèğčéĞŁåŹĺåŘŇäÿĂçŻőå¡ŢïijŇæĹŰWindowsäÿŁéİćçŽĎScriptsçŻőå¡Ţäÿ

```
bash % pyvenv Spam
bash %
```

äijăçżŹ `pyvenv` åŚ¡äžd'çŽĎåŘ■å■ŮæŸŕåřĘèęĄèćńåĹŽåžžçŽĎçŻőå¡ŢåŘ■ãĂĆå¡ŞèćńåĹŽåžžåŘŐïijŇS

```
bash % cd Spam
bash % ls
bin include lib pyvenv.cfg
bash %
```

åIJĺbinçŻőå¡Ţäÿ■ïijŇä¡ăäijŽæL'¿åĹřäÿĂäÿłåŘŕäžěä¡£çŤĺçŽĎPythonèğčéĞŁåŹĺïijŽ

```
bash % Spam/bin/python3
Python 3.3.0 (default, Oct 6 2012, 15:45:22)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more␣
↪information.
```

```
>>> from pprint import pprint
>>> import sys
>>> pprint(sys.path)
['',
'/usr/local/lib/python33.zip',
'/usr/local/lib/python3.3',
'/usr/local/lib/python3.3/plat-darwin',
'/usr/local/lib/python3.3/lib-dynload',
'/Users/beazley/Spam/lib/python3.3/site-packages']
>>>
```

è£Źäÿłèğčé ĞŁåŹĺçŽĎçĽźçĆźåřśæŸŕäżŰçŽĎsite-packagesçŻőå¡Ţèćńèő¿ç¡őäÿžæŰřåĹŻåżžçŽĎçŐŕáćČ åęĆæđIJä¡ãèęĄåőĽèčĚçňňäÿĽæŰźåŇĚïijŇåőČäžňäijŽèćńåőĽèčĚåIJĺéĆčéĞŇïijŇèĂŇäÿ■æŸŕéĂŽåÿÿçşżçż packagesçŻőå¡ŢãĂĆ

èőĺèőž

åĹŻåżžèŹŽæŃ§çŐŕáćČéĂŽåÿÿæŸŕäÿžäžĘåőĽèčĚåŠŇçőąçŘĘçňňäÿĽæŰźåŇĚãĂĆ æ■čåęĆä¡ăåIJĺä¿Ńå■Řäÿ■çIJŃåĹřçŽĎéĆčæăůïijŇsys.path åŘŸéĞŔåŇĚåŘńæİèèĞłäžŐçşżż§PythonçŽĎçŻőå¡ŢïijŇ èĂŇ site-packagesçŻőå¡ŢåůšçżŕèćńéĞ■åőŽä¡■åĹřäÿĂäÿłæŰřçŽĎçŻőå¡ŢãĂĆ

æIJL'äžĘäÿĂäÿłæŰřçŽĎèŹŽæŃ§çŐŕáćČïijŇäÿŃäÿĂæ■ěårśæŸŕåőL'èčĚäÿĂäÿłåŇĚçőąçŘĘåŹĺïijŇærŤå ä¡ĘåőL'èčĚè£Źæăůç ŽĎåůěåĚůåŠŇåŇĚçŽĎæŮűåĂŹïijŇä¡ăéIJĂèęĄçąőä£İä¡ää¡£çŤĺçŽĎæŸŕèŹŽæŃ§çŐŕáćČ åőČäijŽårĘåŇĚåőL'èčĚåĹřæŰřåĹŻåżžçŽĎsite-packagesçŻőå¡Ţäÿ■åŐžãĂĆ

år¡çőąäÿĂäÿłèŹŽæŃ§çŐŕáćČçIJŃäÿŁåŐżæŸŕPythonåőL'èčĚçŽĎäÿĂäÿłåd'■åĹűïijŇ äÿ■è£ĞåőČåőđéŹĚäÿŁåŘłåŇĚåŘńäžĘårŠéĞŔåĞääÿłæŰĞäžűåŠŇäÿĂäžŻçňęåŘůéŞ¿æŐěãĂĆ çt'ăæIJL'æăĞåĞĘåžŞåĞ¡æŰĞäžűåŠŇåŘŕæL'ğèąŇèğčéĞŁåŹĺéČ¡æİèèĞłåŐ§æİęçŽĎPythonåőL'èčĚãĂĆ åŽăæ■d'ïijŇåĹŻåżžè£Źæăůç ŽĎçŐŕáćČæŸŕå¿Ĺåőźæ ŸŞçŽĎïijŇåźűäÿŤåĞääźŐäÿ■äijŽæűĹèĂŮæIJžåŹĺèţĎæ

éżŸèőd'æČĚåĘţäÿŃïijŇèŹŽæŃ§çŐŕáćČæŸŕçl'žçŽĎïijŇäÿ■åŇĚåŘńäžżä¡ŢéćIåd'ŰçŽĎçňňäÿĽæŰźåžŞ åŘŕäžěä¡£çŤĺâĂIJ–system-site-packagesâĂİéĂL'éąźæİèåĹŻåżžèŹŽæŃ§çŐŕáćČïijŇä¿ŃåęĆïijŽ

```
bash % pyvenv --system-site-packages Spam
bash %
```

èů§åd'ŽåĚşäžŐ `pyvenv` åŠŇèŹŽæŃ§çŐŕáćČçŽĎä£qæĄŕåŘŕäžěåŔĆèĂČ PEP 405.

10.15 åĹĘåŘŚåŇĚ

éŮőécŸ

ä¡ăåůšçżŘçijŰåĘŹäžĘäÿĂäÿłæIJL'çŤĺçŽĎåžŞïijŇæČşåřĘåőČåĹĘäžńçżŹåĚűäżŰäžžãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşåĹĘåŔŚä¡ăçŽĎäžčçăĄïijŇçňňäÿĂäžűäžŇåřśæŸŕçżŹåőČäÿĂäÿłåŤŕäÿĂçŽĎåŘ■å■ŮïijŇä¿ŇåęĆïijŇäÿĂäÿłåĚÿåđŃçŽĎåĞ¡æŢřåžŞåŇĚäijŽçśżäijijäÿŃéİćè£ŹæăůïijŽ

```
projectname/
    README.txt
    Doc/
        documentation.txt
    projectname/
        __init__.py
        foo.py
        bar.py
        utils/
            __init__.py
            spam.py
            grok.py
    examples/
        helloworld.py
        ...
```

èęĄèőľä¡ăçŽĎåŇĚåŔŕäžěåŔŚåÿČåĞžåŐżïijŇéęŰåĚĹä¡ăèęĄçijŰåĘŹäÿĂäÿł `setup.py` ïijŇçśżäijijäÿŃéİćè£ŹæăůïijŽ

```
# setup.py
from distutils.core import setup

setup(name='projectname',
    version='1.0',
    author='Your Name',
    author_email='you@youraddress.com',
    url='http://www.you.com/projectname',
    packages=['projectname', 'projectname.utils'],
)
```

äÿŃäÿĂæ■ěïijŇåřśæŸŕåĹŻåžžäÿĂäÿł `MANIFEST.in` æŰĞäžűïijŇåĹŮåĞžæĽĂæIJL'åIJĺä¡ăçŽĎåŇĚäÿ

```
# MANIFEST.in
include *.txt
recursive-include examples *
recursive-include Doc *
```

çąőä£İ `setup.py` åŠŇ `MANIFEST.in` æŰĞäžűæŤ¿åIJĺä¡ăçŽĎåŇĚçŽĎæIJĂéąűçžğçŻőå¡Ţäÿ■ãĂĆäÿĂæŮęä¡ăåůšçżŔåĄŽäžEè£ŹäžŻïijŇä¡ăåřśåŔŕäžěåČŔäÿŃéİćè£ŹæăůæĽğèąŇåŚ¡äžd'æİěåĹŻåžžäÿĂäÿłæžŘ

```
% bash python3 setup.py sdist
```

åőČäijŽåĹŻåžžäÿĂäÿłæŰĞäžűæŕŤåęĆ"projectname-1.0.zip" æĹŰ âĂIJprojectname-1.0.tar.gzâĂİ, åĚůä¡Şä¿ÌètŰäžŐä¡ăçŽĎçşżçż§åźşåŔŕãĂĆåęĆæđIJäÿĂåĹĞæ■čåÿÿïijŇè£ŹäÿłæŰĞäžűåřśåŔŕäžěåŔŚéĂĄçżŹåĹńäžžä¡£çŤĺæĹŰèĂĚäÿŁäijăèĞş Python Package Index.

èőĺèőž

åŕźäžŐçžŕPythonäżčçăĄïijŇçijŰåĘŹäÿĂäÿłæŹőéĂŽçŽĎ `setup.py` æŰĞäżűéĂŽåÿÿå¿ĹçőĂå■ŢãĂĆ äÿĂäÿłåŔŕèČ¡çŽĎéŮőéćŸæŸŕä¡ăå£ĚéążæĽ́ŃåŁĺåĹŮåĞžæĽĂæIJĽæđĎæ äÿĂäÿłåÿÿèğĄéŤŹèŕŕåřśæŸŕäżĚäżĚåŔłåĹŮåĞžäÿĂäÿłåŇĚçŽĎæIJĂéąűçžğçŻőå¡ŢïijŇå£ŸèőřäžĘåŇĚåŘńåŇ è£Źäź§æŸŕäÿžäżĂäźĹåIJĺ `setup.py` äÿ■åŕźäžŐåŇĚçŽĎèŕťæŸŐåŇĚåŘńäžĘåĹŮèąĺ

```
packages=['projectname', 'projectname.utils']
```

åď ğéČĺåĹĘPythonçĺŃåžŔåŚŸéČ¡ç§ěéĄŞïijŇæIJĽå¿Ĺåď ŽçňňäÿĽæŰźåŇĚçőąçŘĘåŹĺä¿ŻéĂĽæŃľïijŇ æIJĽäžŻæŸŕäÿžäžEæŻ£äżčæăĞåĞĘåžŞäÿ■çŽĎdistutilsãĂĆæşĺæĎŔåęĆæđIJä¡ää¿ÌètŰè£ŹäžŻåŇĚïijŇ çŤĺæĹůåŔŕèČ¡äÿ■èČ¡åőĽèčĚä¡ăçŽĎè¡ŕäżűïijŇéŹď éİđäžŰäżňåůšçżŔäžŃåĚĹåőĽèčĚè£ĞæĽ´ĂéIJĂèęĄçŽĎå æ■čåŻăåęĆæ■ď ïijŇä¡ăæŻťåžŤèŕěæŮűåĹżèőřä¡ŔèűŁçőĂå■Ţèűŀåě¡çŽĎéĄŞçŘĘãĂĆ æIJĂåě¡èőľä¡ăçŽĎäżčçăĄä¡£çŤĺæăĞåĞĘçŽĎPython 3åőĽèčĚãĂĆ åęĆæđIJåĚűäžŰåŇĚäź§éIJĂèęĄçŽĎèŕİïijŇåŔŕäžěéĂŽè£ĞäÿĂäÿłåŔŕéĂĽéąźæİěæŤŕæŇĄãĂĆ

åŕźäžŐæűĽåŘŁåĹřCæĽľåśŢçŽĎäżčçăĄæĽŞåŇĚäÿŐåĹĘåŔŚåřśæŻťåď ■æİĆçĆźäžĘãĂĆ çňň15çńååŕźåĚşäžŐCæĽľåśŢçŽĎè£ŹæŰźéİćç§ěèŕĘæIJĽäÿĂäžŻèŕęçżĘèőšèğčïijŇçĽźåĹńæŸŕåIJĺ15.2åřŔè

CHAPTER 13

çňňå■ĄäÿĂçńäïijŽç¡ŚçżIJäÿŐWebçijŰçĺŃ

æIJňçńăæŸŕåĚşäžŐåIJĺç¡ŚçżIJåžŤçŤĺåŠŇåĹĘåÿČåijŔåžŤçŤĺäÿ■ä¡£çŤĺçŽĎåŘĎçğ■äÿżécŸãĂĆäÿżécŸ

Contents:

11.1 ä¡lJäÿžåőćæĹůçńŕäÿŐHTTPælJ■åŁąäžď'äžŠ

éŮőécŸ

ä¡ăéIJĂèęĄéĂŽè£ĞHTTPå■ŔèőőäžěåőćæĹůçńŕçŽĎæŰźåijŔèő£éŮőåď'Žçğ■æIJ■åŁąãĂĆä¿ŇåęĆïijŇäÿ

èğčåĘşæŰźæąĹ

åŕźäžŐçőĂå■ŢçŽĎäžŇæČĚæİěèŕť'ïijŇéĂŽåÿÿä¡£çŤĺ `urllib.request` æĺąåİŮåřšåď§äžĘãĂĆä¿ŇåęĆïijŇåŘŚéĂĄäÿĂäÿłçőĂå■ŢçŽĎHTTP GETèřůæśĆåĹřè£IJçĺŃçŽĎæIJ■åŁąäÿŁïijŇåŘŕäžěè£ŹæăůåĄŽïijŽ

```
from urllib import request, parse

# Base URL being accessed
url = 'http://httpbin.org/get'

# Dictionary of query parameters (if any)
parms = {
   'name1' : 'value1',
   'name2' : 'value2'
}
```

```
# Encode the query string
querystring = parse.urlencode(parms)

# Make a GET request and read the response
u = request.urlopen(url+'?' + querystring)
resp = u.read()
```

åęĆæđIJä¡ăéIJĂèęĄä¡£çŤĺPOSTæŰźæşŢåIJĺèřůæśĆäÿżä¡Şäÿ■åŔŚéĂĄæ§ěèŕćåŔĆæŢřïijŇåŔŕäžěåřĘåŔ
urlopen() åĞ¡æŢřïijŇářšåČŔè£ŹæăůïijŽ

```
from urllib import request, parse

# Base URL being accessed
url = 'http://httpbin.org/post'

# Dictionary of query parameters (if any)
parms = {
   'name1' : 'value1',
   'name2' : 'value2'
}

# Encode the query string
querystring = parse.urlencode(parms)

# Make a POST request and read the response
u = request.urlopen(url, querystring.encode('ascii'))
resp = u.read()
```

åęĆæđIJä¡ăéIJĂèęĄåIJĺåŔŚåĞžçŽĎèřůæśĆäÿ■æŔŘä¿ŻäÿĂäžŻèĞłåőŽäźL'çŽĎHTTPåd'ť'ïijŇä¿ŃåęĆä£
user-agent å■Ůæőţ,åŔŕäžěåĹŻåžžäÿĂäÿłåŇĚåŘńå■ŮæőţåĭijçŽĎå■ŮåĚÿïijŇáźűåĹŻåžžäÿĂäÿłRequestå
urlopen() ïijŇåęĆäÿŃïijŽ

```
from urllib import request, parse
...

# Extra headers
headers = {
    'User-agent' : 'none/ofyourbusiness',
    'Spam' : 'Eggs'
}

req = request.Request(url, querystring.encode('ascii'),␣
↪headers=headers)

# Make a request and read the response
u = request.urlopen(req)
resp = u.read()
```

åęĆæđIJéIJĂèęĄäžd'äžŠçŽĎæIJ■åŁąæŕŤäÿŁéĺćçŽĎä¿Ńå■ŘéČ¡èęĄåd'■æĺĆïijŇäź§èőÿåžŤèŕěåŐżçIJŃç

requests åžŞïijĹhttps://pypi.python.org/pypi/requestsïijĽãĂĆä¿ŃåęĆïijŇäÿŃéĺćè£Źäÿłçđ'žä¿ŃéĞĞçŤĺreques

```
import requests

# Base URL being accessed
url = 'http://httpbin.org/post'

# Dictionary of query parameters (if any)
parms = {
   'name1' : 'value1',
   'name2' : 'value2'
}

# Extra headers
headers = {
    'User-agent' : 'none/ofyourbusiness',
    'Spam' : 'Eggs'
}

resp = requests.post(url, data=parms, headers=headers)

# Decoded text returned by the request
text = resp.text
```

åĚşäžŐrequestsåžŞïijŇäÿĂäÿłåĂijå¿ŮäÿĂæŔŘçŽĎçĽźæĂğářśæŸŕåőČèČ¡äžěåđ'ŽçğæŰźåijŔäžŐèřůa
`resp.text` åÿęçżŹæĹŚäžńçŽĎæŸŕäžěUnicodeèğčçăĄçŽĎåŞåžŤæŰĞæIJňãĂĆä¡ĘæŸŕïijŇåęĆæđIJåŐżèć
`resp.content` ïijŇářśäijŽå¿ŮåĹřåŐ§ågŃçŽĎäžŇè£ŻåĹűæŢřæ■õãĂĆåŔęäÿĂæŰźéĺćïijŇåęĆæđIJèő£éŮć
`resp.json` ïijŇéĆčäźĹářśäijŽå¿ŮåĹřJSONæăijåijŔçŽĎåŞåžŤåĘĚåőźãĂĆ

äÿŃéĺćè£Źäÿłçđ'žä¿ŃåĹl'çŤĺ `requests` åžŞåŘŚèţůäÿĂäÿłHEAD-èřůæśĆïijŇåźűäžŐåŞåžŤäÿ■æŔŘåŘŰåĞžäÿĂäžŻHTTPåđ't'æŢřæ■őçŽĎå■Ůæőţïijž

```
import requests

resp = requests.head('http://www.python.org/index.html')

status = resp.status_code
last_modified = resp.headers['last-modified']
content_type = resp.headers['content-type']
content_length = resp.headers['content-length']
```

äÿŃéĺćæŸŕäÿĂäÿłåĹl'çŤĺrequestséĂŽè£Ğå§žæIJňèőđ'èŕĄçŹžå¡ŢPypiçŽĎä¿Ńå■Řïijž

```
import requests

resp = requests.get('http://pypi.python.org/pypi?:action=login',
                    auth=('user','password'))
```

äÿŃéĺćæŸŕäÿĂäÿłåĹl'çŤĺrequestsåřĘHTTP cookiesäžŐäÿĂäÿłèřůæśĆäijåéĂŠåĹřåŔęäÿĂäÿłçŽĎä¿Ńå■

```
import requests

# First request
resp1 = requests.get(url)
...

# Second requests with cookies received on first requests
resp2 = requests.get(url, cookies=resp1.cookies)
```

æIJĂåŘŐä¡Ęåźűéİđæ IJĂäÿ■éĞ■èęAçŽĎäÿĂäÿłä¿Ńå■ŘæŸŕçŤĺrequestsäÿŁäijååĘĚåőźïijŽ

```
import requests
url = 'http://httpbin.org/post'
files = { 'file': ('data.csv', open('data.csv', 'rb')) }

r = requests.post(url, files=files)
```

èőĺèőž

åŕźäžŐçIJ§çŽĎå¿ĹçőĂå■ŢHTTPåőćæĹůçńŕäžčçăĄïijŇçŤĺåĘĚç¡őçŽĎ urllib æĺąåİŮéĂŽåÿÿåřśèűşåď§äžĘãĂĆä¡ĘæŸŕïijŇåęĆæđIJä¡äèęĄåĄŽçŽĎäÿ■äżĚäžĚåŔłæŸŕçőĂå■ŢçŽĎGETæĹ requests åď ğæŸ¿èžńæĽŃçŽĎæŮűåĂŹäžĘãĂĆ

ä¿ŃåęĆïijŇåęĆæđIJä¡ääĘşåőŽåİŽæŇĄä¡£çŤĺæăĞåĞĘçŽĎçĺŃåžŔåžŞèĂŇäÿ■èĂČèŹŚåČŔ requests è£ŹæäůçŽĎçňňäÿĽæŰźåžŞïijŇéĆčäźĹäź§èőÿåřśäÿ■å¿Ůäÿ■ä¡£çŤĺåžŢåśĆçŽĎ http.client æĺąåİŮæİěåőđçŐřèĞłåůśçŽĎäžčçăĄãĂĆærŤæŰźèŕť ïijŇäÿŃéİćçŽĎäžčçăĄåśŢçď žäžĘåęĆä

```
from http.client import HTTPConnection
from urllib import parse

c = HTTPConnection('www.python.org', 80)
c.request('HEAD', '/index.html')
resp = c.getresponse()

print('Status', resp.status)
for name, value in resp.getheaders():
    print(name, value)
```

åŘŇæăůåIJřïijŇåęĆæđIJå£Ěéąžçij ŰåĘŹæűĽåŔŁäžčçŘĘãĂĄèőď èŕĄãĂĄcookiesäžěåŔŁåĚűäžŰäÿĂäžŽ urllib åřśæŸ¿å¿ŮçĽźåĹńåĹńæĽ■åŠŇåŢřåŮęãĂĆærŤæŰźèŕť ïijŇäÿŃéİćè£Źäÿłçđ žä¿Ńåőđç ŐřåIJĺPython

```
import urllib.request

auth = urllib.request.HTTPBasicAuthHandler()
auth.add_password('pypi','http://pypi.python.org','username',
 ↪'password')
opener = urllib.request.build_opener(auth)

r = urllib.request.Request('http://pypi.python.org/pypi?
 ↪:action=login')
```

```
u = opener.open(r)
resp = u.read()

# From here. You can access more pages using opener
...
```

åİęçŹ¡èŕťˇïijŇæĽĂæIJĽçŽĎè£ŹäžŻæŞ■ä¡IJåIJĺ requests åžŞäÿ■éČ¡åŔŸå¿ŮçőĂå■ŢçŽĎåď˙ŽãĂĆ

åIJĺåijĂåŔŚè£ĞçĺŃäÿ■æţŃèŕŢHTTPåőćæĹůçńŕäžčçăĄåÿÿåÿÿæŸŕå¿Ĺäžď˙äžžæšöäÿğçŽĎïijŇåŻääÿžæĽ //httpbin.orgïijĽãĂĆè£ŹäÿłçńŹçĆźäijŽæŐěæŤůåŔŚåĞžçŽĎèřůæśĆïijŇçĎűåŘŐäžěJSONçŽĎå¡ćåijŔářĘçŻÿ

```
>>> import requests
>>> r = requests.get('http://httpbin.org/get?name=Dave&n=37',
...     headers = { 'User-agent': 'goaway/1.0' })
>>> resp = r.json
>>> resp['headers']
{'User-Agent': 'goaway/1.0', 'Content-Length': '', 'Content-Type': '
 ↪',
'Accept-Encoding': 'gzip, deflate, compress', 'Connection':
'keep-alive', 'Host': 'httpbin.org', 'Accept': '*/*'}
>>> resp['args']
{'name': 'Dave', 'n': '37'}
>>>
```

åIJĺèęĄåŘŇäÿĂäÿłçIJ§æ■čçŽĎçńŹçĆźè£ŻèąŇäžď˙äžŠåĽ■ïijŇåĚĹåIJĺ httpbin.org è£ŹæăůçŽĎç¡ŚçńŹäÿŁåĄŽåőđéłŇåÿÿåÿÿæŸŕåŔŕåŔŰçŽĎåŁđæşŢãĂĆåŕď˙åĚűæŸŕå¡ŞæĹŚäžňéİćåŕź3æňąçŹ

åř¡çőąæIJňèŁĆæšąæIJĽæűĽåŔŁïijŇ `request` åžŞè£Ÿåŕźèőÿåď˙ŽénŸçžğçŽĎHTTPåőćæĹůçńŕå■Ŕèőő `requests` æĺąåİŮçŽĎæŰĞæąčïijĹhttp://docs.python-requests.org)èť˙léĞŔå¿ĹénŸïijĹåİęçŹ¡èŕťˇærŤåIJĺè£Źç§■ç§■çŽĎäÿĂèŁĆçŽĎçŕĞáźĚäÿ■æĽ˙ĂæŔŘä¿ŽçŽĎäżżä¡Ţä£ą

11.2 åĹŻåżžTCPæIJ■åŁąåŹĺ

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłæIJ■åŁąåŹĺïijŇéĂŽè£ĞTCPå■ŔèőőåŠŇåőćæĹůçńŕéĂŽä£ąãĂĆ

èğčåĘşæŰźæąĹ

åĹŻåżžäÿĂäÿłTCPæIJ■åŁąåŹĺçŽĎäÿĂäÿłçőĂå■ŢæŰźæşŢæŸŕä¡£çŤĺ `socketserver` åžŞãĂĆä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎåžŤç■ŤæIJ■åŁąåŹĺïijŽ

```
from socketserver import BaseRequestHandler, TCPServer

class EchoHandler(BaseRequestHandler):
    def handle(self):
```

```
        print('Got connection from', self.client_address)
        while True:

            msg = self.request.recv(8192)
            if not msg:
                break
            self.request.send(msg)

if __name__ == '__main__':
    serv = TCPServer(('', 20000), EchoHandler)
    serv.serve_forever()
```

åIJĺè£Źæőţäžčçă Ąäÿ■ïijŇä¡ääőŽäźL'äžĘäÿĂäÿłçL'źæőŁçŽĎåď'ĎçŘĘçśżïijŇåőđçŐřäžĘäÿĂäÿł handle() æŰźæşŢïijŇçŤĺæİěäÿžåőćæĹůçńŕè£đæŐěæIJ■åŁąãĂĆ request åśđæĂğæŸŕåőćæĹůçńŕsocketïijŇclient_address æIJL'åőćæĹůçńŕåIJřåİĂãĂĆ äÿžäžĘæţŃèŕŢè£ŹäÿłæIJ■åŁąåŹĺïijŇè£ŘèąŇåőČåźűæL'Şåij ÄåŔęåď'ŰäÿĂäÿłPythonè£ŻçĺŃè£đæŐěè£Źäÿłæ

```
>>> from socket import socket, AF_INET, SOCK_STREAM
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s.connect(('localhost', 20000))
>>> s.send(b'Hello')
5
>>> s.recv(8192)
b'Hello'
>>>
```

å¿Ĺåď'ŽæŮűåĂŹïijŇåŔŕäžěå¿Ĺåőźæ ŸŞçŽĎåőŽäźL'äÿĂäÿłäÿ■åŘŇçŽĎåď'ĎçŘĘåŹĺãĂĆäÿŃéİćæŸŕäÿĂ StreamRequestHandler å§žçśżåŕĘäÿĂäÿłçśżæŰĞäžűæŐěåŔčæŤ¿ç¡őåIJĺåžŢåśĆsocketäÿŁçŽĎä¿Ńå■Ř

```
from socketserver import StreamRequestHandler, TCPServer

class EchoHandler(StreamRequestHandler):
    def handle(self):
        print('Got connection from', self.client_address)
        # self.rfile is a file-like object for reading
        for line in self.rfile:
            # self.wfile is a file-like object for writing
            self.wfile.write(line)

if __name__ == '__main__':
    serv = TCPServer(('', 20000), EchoHandler)
    serv.serve_forever()
```

èőĺèőž

socketserver åŔŕäžěèőľ'æĹŚäžňå¿Ĺåőźæ ŸŞçŽĎåĹŻåžžçőĂå■ŢçŽĎTCPæIJ■åŁąåŹĺãĂĆ ä¡ĘæŸŕïijŇä¡ăéIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇéżŸèőď'æČĚåĘţäÿŃè£Źçğ■æIJ■åŁąåŹĺæŸŕå■Ţçž£çĺŃçŽĎïijŇäÿ åęĆæđIJä¡ăæČşåď'ĎçŘĘåď'ŽäÿłåőćæĹůçńŕïijŇåŔŕäžěåĹİåğŃåŇŰäÿĂäÿł

ForkingTCPServer æĹŰèĂĚæŸŕ ThreadingTCPServer åŕźèśąãĂĆä¿ŃåęĆïijŽ

```
from socketserver import ThreadingTCPServer


if __name__ == '__main__':
    serv = ThreadingTCPServer(('', 20000), EchoHandler)
    serv.serve_forever()
```

ä¡£çŤĺforkæĹŰçž£çĺŃæIJ■åŁąåŹĺæIJL'äÿłæ¡IJåIJĺéŮőécŸåřśæŸŕåőČäžňäijŽäÿžæŕŔäÿłåőćæĹůçńrè£đæçŤśäžŐåőćæĹůçńrè£đæŐěæŢřæŸŕæšąæIJL'éŹŘåĹűçŽĎïijŃåŽäæ■đ'äÿĂäÿłæĄűæĎŔçŽĎéŻŠåőćåŔŕäžěåŘŃ

åęĆæđIJä¡ăæŃĚå£Čè£ŹäÿłéŮőécŸïijŃä¡ăåŔŕäžěåĹŻåžžäÿĂäÿłécĎåĚĹåĹĘéĚ■åđ'gåřŔçŽĎåůěä¡IJçž£çä¡ăåĚĹåĹŻåžžäÿĂäÿłæŹőéĂŽçŽĎéIđçž£çĺŃæIJ■åŁąåŹĺïijŃçĎűåŘŐåIJĺäÿĂäÿłçž£çĺŃæśăäÿ■ä¡£çŤĺ serve_forever() æŰźæşŢæİěåŘŕåŁĺåőČäžňãĂĆ

```
if __name__ == '__main__':
    from threading import Thread
    NWORKERS = 16
    serv = TCPServer(('', 20000), EchoHandler)
    for n in range(NWORKERS):
        t = Thread(target=serv.serve_forever)
        t.daemon = True
        t.start()
    serv.serve_forever()
```

äÿĂèĹňæİěèőšïijŃäÿĂäÿł TCPServer åIJĺåőđä¿ŃåŃŰçŽĎæŮűåĂŹäijŽçżŚåőŽåźűæ£ÄæťżçŻÿåžŤçŽ socket ãĂĆ äÿ■è£ĞïijŃæIJL'æŮűåĂŹä¡ăæČşéĂŽè£Ğèő¿ç¡őæ§ŘäžŻéĂL'éąźåŐżèřČæŢť'åžŢäÿŃçŽĎ *socket*' ïijŃåŘŕäžěèő¿ç¡őåŔĆæŢř bind_and_activate=False ãĂĆåęĆäÿŃïijŽ

```
if __name__ == '__main__':
    serv = TCPServer(('', 20000), EchoHandler, bind_and_
↪activate=False)
    # Set up various socket options
    serv.socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR,␣
↪True)
    # Bind and activate
    serv.server_bind()
    serv.server_activate()
    serv.serve_forever()
```

äÿŁéİćçŽĎ socket éĂL'éąźæŸŕäÿĂäÿłéİđåÿÿæŹőéĄ■çŽĎéĚ■ç¡őéąźïijŃåőČåĚĄèõÿæIJ■åŁąåŹĺéĞ■æçŤśäžŐèęĄèćńçżŔåÿÿä¡£çŤĺåĹřïijŃåőČèćńæŤ¿ç¡őåĹřçśżåŔŸéĞŔäÿ■ïijŃåŘŕäžěçŻťæŐěåIJĺ TCPServer äÿŁéİćèő¿ç¡õãĂĆ åIJĺåőđä¿ŃåŃŰæIJ■åŁąåŹĺçŽĎæŮűåĂŹåŐżèő¿ç¡őåőČçŽĎåĂijïijŃåęĆäÿŃ

```
if __name__ == '__main__':
    TCPServer.allow_reuse_address = True
    serv = TCPServer(('', 20000), EchoHandler)
    serv.serve_forever()
```

åIJĺäÿŁéİćçd'žä¿Ńäÿ■ïijŃæĹŚäžňæijŤçd'žäžĘäÿd'çğ■äÿ■åŘŃçŽĎåd'ĎçŘĘåŹĺå§žçśżïijĹ

BaseRequestHandler åŠŇ StreamRequestHandler ïijĽãĂĆ StreamRequestHandler æŻťåŁăçĄţætťżçĆźïijŇèČ¡éĂŽè£Ğèŏżç¡ŏåĚűäżŰçŽĎçśžåŔŸéĞŕæİěæŤŕæŇA

```
import socket

class EchoHandler(StreamRequestHandler):
    # Optional settings (defaults shown)
    timeout = 5                         # Timeout on all socket␣
↪operations
    rbufsize = -1                       # Read buffer size
    wbufsize = 0                        # Write buffer size
    disable_nagle_algorithm = False  # Sets TCP_NODELAY socket␣
↪option
    def handle(self):
        print('Got connection from', self.client_address)
        try:
            for line in self.rfile:
                # self.wfile is a file-like object for writing
                self.wfile.write(line)
        except socket.timeout:
            print('Timed out!')
```

æIJĂåŘŐïijŇè£ŸéIJÄèęĄæşĺæĎŔçŽĎæŸŕåůĺåď ğéČĺåĹĘPythonçŽĎéńŸåśĆç¡ŚçżIJæĺąåİŮïijĹæŕŤåęĆI
RPCç■L'ïijL'éČ¡æŸŕåžžçńŇåIJĺ socketserver åŁ§èČ¡äźŇäÿŁãĂĆ
äź§åřśæŸŕèŕťïijŇçŻťæŐěä¡£çŤĺ socket åžŞæİěåőđçŐřæIJ■åŁąåŹĺäź§åźűäÿ■æŸŕå¿ĹéŽ¿ãĂĆ
äÿŃéĺćæŸŕäÿĂäÿłä¡£çŤĺ socket çŻťæŐěçijŰçĺŃåőđçŐřçŽĎäÿĂäÿłæIJ■åŁąåŹĺçőĂå■Ţä¿Ńå■ŘïijŽ

```
from socket import socket, AF_INET, SOCK_STREAM

def echo_handler(address, client_sock):
    print('Got connection from {}'.format(address))
    while True:
        msg = client_sock.recv(8192)
        if not msg:
            break
        client_sock.sendall(msg)
    client_sock.close()

def echo_server(address, backlog=5):
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(address)
    sock.listen(backlog)
    while True:
        client_sock, client_addr = sock.accept()
        echo_handler(client_addr, client_sock)

if __name__ == '__main__':
    echo_server(('', 20000))
```

11.3 åĹŻåžžUDPæIJ■åŁąåŹÍ

éůőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłå§žäžŐUDPå■ŔèőőçŽĎæIJ■åŁąåŹÍæİěäÿŐåőćæĹůçńŕéĂŽä£ąãĂĆ

èğčåĘşæŰźæąĹ

èů§TCPäÿĂæăůïijŇUDPæIJ■åŁąåŹÍäź§åŔŕäżěéĂŽè£Ğä¡£çŤÍ socketserver åžŞå¿ĹåőźæŸŞçŽĎèćńåĹŻåžžãĂĆ ä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎæŮűéŮť æIJ■åŁąåŹÍïijŽ

```
from socketserver import BaseRequestHandler, UDPServer
import time

class TimeHandler(BaseRequestHandler):
    def handle(self):
        print('Got connection from', self.client_address)
        # Get message and client socket
        msg, sock = self.request
        resp = time.ctime()
        sock.sendto(resp.encode('ascii'), self.client_address)

if __name__ == '__main__':
    serv = UDPServer(('', 20000), TimeHandler)
    serv.serve_forever()
```

èů§äźŃåL'■äÿĂæăůïijÑä¡ăåĚĹåőŽäźL'äÿĂäÿłåőđçŐř handle() çL'źæőŁæŰźæşŢçŽĎçśžïijÑäÿžåőćæĹůçńŕè£đæŐěæIJ■åŁąãĂĆ è£ŹäÿłçśżçŽĎ request åśđæĂğæŸŕäÿĂäÿłåŇĚåŘńäžĘæŢřæ■őæŁěåŠŇåžŢåśĆsocketåŕźèśąçŽĎåĚČçżĎãĂĆclient_address åŇĚåŘńäžĘåőćæĹůçńŕåIJřåİĂãĂĆ

æĹŚäźňæİěæţŃèŕŢäÿŃè£ŹäÿłæIJ■åŁąåŹÍïijŇéęŰåĚĹè£ŘèąŇåőČïijŇçĎűåŘŐæL'ŞåijĂåŔęåd'ŰäÿĂäÿł

```
>>> from socket import socket, AF_INET, SOCK_DGRAM
>>> s = socket(AF_INET, SOCK_DGRAM)
>>> s.sendto(b'', ('localhost', 20000))
0
>>> s.recvfrom(8192)
(b'Wed Aug 15 20:35:08 2012', ('127.0.0.1', 20000))
>>>
```

èőÍèőž

äÿĂäÿłåĚÿåđŃçŽĎUPDæIJ■åŁąåŹÍæŐěæŤűåĹřè¿¿çŽĎæŢřæ■őæŁě(æűĹæĄŕ)åŠŇåőćæĹůçńŕåIJřåİĂãĂ åőČèęĄçżŹåőćæĹůçńŕåŽđåŘŚäÿĂäÿłæŢřæ■őæŁěãĂĆåŕźäžŐæŢřæ■őæŁęçŽĎäijăéĂĄïijŇ ä¡ååžŤèŕěä¡£çŤÍsocketçŽĎ sendto() åŠŇ recvfrom() æŰźæşŢãĂĆ

åř¡çőąäijăçż§çŽĎ `send()` åŠŇ `recv()` äźšåŔŕäžěè¿¿åĹřåŘŇæăůçŽĎæŢĹæđIJïijŇ ä¡ĘæŸŕåĽ■éİćçŽĎäÿď'äÿłæŰźæşŢåŕźäžŐUDPè£đæŐěèĂŇèÍĂæŻť'æŹőéĄ■ãĂĆ

çŤśäžŐæšąæIJL'åžŢåśĆçŽĎè£đæŐěïijŇUPDæIJ■åŁąåŹÍçŻÿåŕźäžŐTCPæIJ■åŁąåŹÍæİèèőšåőđçŐřètůæİ äÿ■è£ĞïijŇUDPåď'l'çŤ§æŸŕäÿ■åŔŕéİăçŽĎïijĹåŽääÿžéĂŽä£ąæšąæIJL'åžžçńŇè£đæŐěïijŇæűĹæĄŕåŘŕèČ¡äÿ åŽăæ■ď'éIJĂèęĄçŤśä¡äèĞłåůśæİěåEşåőŽèŕěæĂŐæăůåď'ĎçŘEäÿćåď'śæűĹæĄŕçŽĎæČĚåEţãĂĆè£Źäÿłåůšçżl äÿ■è£ĞéĂŽåÿÿæİèèŕť'ïijŇåęĆæđIJåŔŕéİăæĂğåŕźäžŐä¡äçÍŃåžŘå¿ĹéĞ■èęĄïijŇä¡äéIJĂèęĄåĂ§åŁl'äžŐåžŘåÍ UDPéĂŽåÿÿèćńçŤĺåIJĺéĆčäžŽåŕźäžŐåŘŕéİăäijăè¿Şèę Ąæśćäÿ■æŸŕå¿ĹéńŸçŽĎåIJžåŘĹãĂĆä¿ŃåęĆïijŇåIJĺå æŮăéIJĂè£ŤåŻđæĄćåď'■äÿćåď'śçŽĎæŢřæ■őåŇĚïijĹçÍŃåžŘåŘłéIJĂçőĂå■ŢçŽĎå£¡çŢěåőČåźűçżğçż■åŘŚåL

`UDPServer` çśżæŸŕå■Ţçž£çÍŃçŽĎïijŇäź§åřśæŸŕŕèŕť'äÿĂæňąåŘłèČ¡äÿžäÿĂäÿłåőćæĹůçńŕè£đæŐěæIJ■ åőđéŹĚä¡£çŤĺäÿ■ïijŇè£ŹäÿłæŮăèőžæŸŕåŕźäžŐUDPè£ŸæŸŕTCPéČ¡äÿ■æŸŕäżĂäźĹåď'ğéŮőécŸãĂĆ åęĆæđIJä¡äæČşèęĄåźűåŔŚæŞ■ä¡IJïijŇåŘŕäžěåőđä¿ŃåŇŰäÿĂäÿł `ForkingUDPServer` æĹŰ `ThreadingUDPServer` åŕźèśąïijŽ

```
from socketserver import ThreadingUDPServer

   if __name__ == '__main__':
    serv = ThreadingUDPServer(('',20000), TimeHandler)
    serv.serve_forever()
```

çŻť'æŐěä¡£çŤĺ `socket` æİěæŸŕæČşäÿĂäÿłUDPæIJ■åŁąåŹÍäź§äÿ■éŽ¿ïijŇäÿŃéİćæŸŕäÿĂäÿłä¿Ńå■Řïij

```
from socket import socket, AF_INET, SOCK_DGRAM
import time

def time_server(address):
    sock = socket(AF_INET, SOCK_DGRAM)
    sock.bind(address)
    while True:
        msg, addr = sock.recvfrom(8192)
        print('Got message from', addr)
        resp = time.ctime()
        sock.sendto(resp.encode('ascii'), addr)

if __name__ == '__main__':
    time_server(('', 20000))
```

11.4 éĂŽè£ĞCIDRåIJŕåİĂçŤ§æĹŘåŕźåžŤçŽĎIPåIJŕåİĂéŻĘ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłCIDRç¡ŚçżIJåIJŕåİĂæŕŤåęĆâĂIJ123.45.67.89/27âĂİïijŇä¡ăæČşåřĘåĚűè¡ňæ■ćæĹŘåőČa ïijĹæŕŤåęĆïijŇâĂIJ123.45.67.64âĂİ, âĂIJ123.45.67.65âĂİ, âĂę, âĂIJ123.45.67.95âĂİ)ïijL'

èğčåĘşæŰźæąĹ

åŔŕäżěä¡£çŤÍ ipaddress æĺąåİŮå¿ĹåőźæŸŞçŽĎåőđçŐřè£ŹæăůçŽĎèőąçőŮãĂĆä¿ŃåęĆïijŽ

```
>>> import ipaddress
>>> net = ipaddress.ip_network('123.45.67.64/27')
>>> net
IPv4Network('123.45.67.64/27')
>>> for a in net:
...     print(a)
...
123.45.67.64
123.45.67.65
123.45.67.66
123.45.67.67
123.45.67.68
...
123.45.67.95
>>>

>>> net6 = ipaddress.ip_network('12:3456:78:90ab:cd:ef01:23:30/125')
>>> net6
IPv6Network('12:3456:78:90ab:cd:ef01:23:30/125')
>>> for a in net6:
...     print(a)
...
12:3456:78:90ab:cd:ef01:23:30
12:3456:78:90ab:cd:ef01:23:31
12:3456:78:90ab:cd:ef01:23:32
12:3456:78:90ab:cd:ef01:23:33
12:3456:78:90ab:cd:ef01:23:34
12:3456:78:90ab:cd:ef01:23:35
12:3456:78:90ab:cd:ef01:23:36
12:3456:78:90ab:cd:ef01:23:37
>>>
```

Network äź§åĚĄèőÿåČŔæŢřçżĎäÿĂæăůçŽĎçť´ćåijŢåŔŰåĂijïijŇä¿ŃåęĆïijŽ

```
>>> net.num_addresses
32
>>> net[0]

IPv4Address('123.45.67.64')
>>> net[1]
IPv4Address('123.45.67.65')
>>> net[-1]
IPv4Address('123.45.67.95')
>>> net[-2]
IPv4Address('123.45.67.94')
>>>
```

åŔęåď ŰïijŇä¡ăè£ŸåŔŕäżěæĽ ğèąŇç¡ŚçżIJæĹŘåŠŸæčĂæ§ěäźŃçśżçŽĎæŞ■ä¡IJïijŽ

```
>>> a = ipaddress.ip_address('123.45.67.69')
>>> a in net
True
>>> b = ipaddress.ip_address('123.45.67.123')
>>> b in net
False
>>>
```

äÿĂäÿłIPåIJřåİĂåŠŇç¡ŚçżIJåIJřåİĂèČ¡éĂŽè£ĞäÿĂäÿłIPæŐěåŔčæİěæŇĞåőŽïijŇä¿ŃåęĆïijŽ

```
>>> inet = ipaddress.ip_interface('123.45.67.73/27')
>>> inet.network
IPv4Network('123.45.67.64/27')
>>> inet.ip
IPv4Address('123.45.67.73')
>>>
```

èőÍèőž

ipaddress æĺąåİŮæIJĽå¿Ĺåď ŽçśżåŔŕäżěèąĺçď žIPåIJřåİĂãĂĄç¡ŚçżIJåŠŇæŐěåŔčãĂĆ å¡Şä¡ăéIJĂèęĄæŞ■ä¡IJç¡ŚçżIJåIJřåİĂïijĹæŕŤåęĆèğčæđŘãĂĄæĽŞå■řãĂĄéłŇèŕĄç■ĽïijĽçŽĎæŮűåĂŹïijŽå

èęĄæşĺæĎŔçŽĎæŸŕïijŇipaddress æĺąåİŮèů§åĚűäżŰäÿĂäžŻåŠŇç¡ŚçżIJçŻÿåĚşçŽĎæĺąåİŮæŕŤåęĆ socket åžŞäżď éŻĘå¿ĹåřŠãĂĆ æĽĂäżěïijŇä¡ăäÿ■èČ¡ä¡£çŤĺ IPv4Address çŽĎåőđä¿ŃæİěäżčæŻ£äÿĂäÿłåIJřåİĂå■ŮçňęäÿšïijŇä¡ăéęŰåĚĹå¿ŮæŸ¿åijŔçŽĎä¡£çŤĺ str() è¡ňæ■ćåőČãĂĆä¿ŃåęĆïijŽ

```
>>> a = ipaddress.ip_address('127.0.0.1')
>>> from socket import socket, AF_INET, SOCK_STREAM
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s.connect((a, 8080))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Can't convert 'IPv4Address' object to str implicitly
>>> s.connect((str(a), 8080))
>>>
```

æŻť åď ŽçŻÿåĚşåĘĚåőźïijŇèŕůåŔĆèĂČ An Introduction to the ipaddress Module

11.5 åĹŻåżžäÿĂäÿłçőĂå■ŢçŽĎRESTæŐěåŔč

éŮőéćŸ

ä¡ăæČşä¡£çŤĺäÿĂäÿłçőĂå■ŢçŽĎRESTæŐěåŔčéĂŽè£Ğç¡ŚçżIJè£IJçĺŃæŐğåĹűæĹŰèő£éŮőä¡ăçŽĎåžŤ

èğčåĘşæŰźæąĹ

æđĎåžžäÿĂäÿłRESTéčŐæăijçŽĎæŐěåŔčæIJĂçőĂå■ŢçŽĎæŰźæşŢæŸŕåĹŽåžžäÿĂäÿłå§žäžŐWSGIæă 3333ïijĽçŽĎå¿ĹåřŔçŽĎåžŞïijŇäÿŃéÍćæŸŕäÿĂäÿłä¿Ńå■ŘïijŽ

```
# resty.py

import cgi

def notfound_404(environ, start_response):
    start_response('404 Not Found', [ ('Content-type', 'text/plain
↪') ])
    return [b'Not Found']

class PathDispatcher:
    def __init__(self):
        self.pathmap = { }

    def __call__(self, environ, start_response):
        path = environ['PATH_INFO']
        params = cgi.FieldStorage(environ['wsgi.input'],
                                  environ=environ)
        method = environ['REQUEST_METHOD'].lower()
        environ['params'] = { key: params.getvalue(key) for key in␣
↪params }
        handler = self.pathmap.get((method,path), notfound_404)
        return handler(environ, start_response)

    def register(self, method, path, function):
        self.pathmap[method.lower(), path] = function
        return function
```

äÿžäžĘä¡£çŤÍè£ŹäÿłèřČåžęåŹÍïijŇä¡ääŔłéIJÄèęAçijŰåĘŹäÿ■åŘŇçŽĎåď ĎçŘĘåŹÍïijŇářšåČŔäÿŃéÍćè£

```
import time

_hello_resp = '''\
<html>
  <head>
     <title>Hello {name}</title>
   </head>
   <body>
     <h1>Hello {name}!</h1>
   </body>
</html>'''

def hello_world(environ, start_response):
    start_response('200 OK', [ ('Content-type','text/html')])
    params = environ['params']
    resp = _hello_resp.format(name=params.get('name'))
    yield resp.encode('utf-8')
```

```
_localtime_resp = '''\
<?xml version="1.0"?>
<time>
  <year>{t.tm_year}</year>
  <month>{t.tm_mon}</month>
  <day>{t.tm_mday}</day>
  <hour>{t.tm_hour}</hour>
  <minute>{t.tm_min}</minute>
  <second>{t.tm_sec}</second>
</time>'''

def localtime(environ, start_response):
    start_response('200 OK', [ ('Content-type', 'application/xml')␣
↪])
    resp = _localtime_resp.format(t=time.localtime())
    yield resp.encode('utf-8')

if __name__ == '__main__':
    from resty import PathDispatcher
    from wsgiref.simple_server import make_server

    # Create the dispatcher and register functions
    dispatcher = PathDispatcher()
    dispatcher.register('GET', '/hello', hello_world)
    dispatcher.register('GET', '/localtime', localtime)

    # Launch a basic server
    httpd = make_server('', 8080, dispatcher)
    print('Serving on port 8080...')
    httpd.serve_forever()
```

èęĄæţŃèŕŢäÿŃè£ŹäÿłæIJ■åŁąåŹĺïijŇä¡ăåŕŕäžěä¡£çŤĺäÿĂäÿłæţŕèğĹåŹĺæĹŰ `urllib` åŠŇåőČäžďäžŠãĂĆä¿ŃåęĆïijŽ

```
>>> u = urlopen('http://localhost:8080/hello?name=Guido')
>>> print(u.read().decode('utf-8'))
<html>
  <head>
     <title>Hello Guido</title>
   </head>
   <body>
     <h1>Hello Guido!</h1>
   </body>
</html>

>>> u = urlopen('http://localhost:8080/localtime')
>>> print(u.read().decode('utf-8'))
<?xml version="1.0"?>
<time>
```

```
  <year>2012</year>
  <month>11</month>
  <day>24</day>
  <hour>14</hour>
  <minute>49</minute>
  <second>17</second>
</time>
>>>
```

èőĺèőž

åIJÍçijŰåĘŹRESTæŐěåŔčæŮűïijŇéĂŽåÿÿéČ¡æŸŕæIJ■åŁąäžŐæŹőéĂŽçŽĎHTTPèŕůæśĆãĂĆä¡ĘæŸŕè
è£ŹäžŻæŢřæ■őäžěåŘĎçğ■æäĞåĞĘæäijåijŔçijŰçăĄïijŇæŕŤåęĆXMLãĂĄJSONæĹŰCSVãĂĆ
åř¡çőąçÍŃåžŔçIJŃäÿŁåŐżå¿ĹçőĂå■ŢïijŇä¡ĘæŸŕäžèè£Źçğ■æŰźåijŔæŔŘä¿ŻçŽĎAPIåŕźäžŐå¿Ĺåď'ŽåžŢçŤÍç

ä¿ŃåęĆïijŇéŢ£æIJ§è£ŘèąŇçŽĎçÍŃåžŔåŔŕèČ¡äijŽä¡£çŤÍäÿĂäÿłREST
APIæİěåőđçŐřçŻŚæŐğæĹŰèŕŁæŰ■ãĂĆ åď'ğæŢřæ■őåžŤçŤÍçÍŃåžŔåŔŕäžěä¡£çŤÍRESTæİěæđĎåžžäÿĂäÿłæ
RESTè£ŸèČ¡çŤÍæİěæŐğåĹűçąňäžűěő¿åď'ĞæŕŤåęĆæIJžåŹÍäžžãĂĄäijäæĎ§åŹÍãĂĄåůěåŐĆæĹŰçĄŕæşąãĂĆ
æŻť éĞ■èęĄçŽĎæŸŕïijŇREST APIåůšçżŔèćńåď'ğéĞŘåőćæĹůçńŕçijŰçÍŃçŐŕåćČæĽ'ĂæŤŕæŇĄïijŇæŕŤåęĆ-
Javascript, Android, iOSç■Ľ'ãĂĆ åŻăæ■d'ïijŇåĹl'çŤÍè£Źçğ■æŐěåŔčåŔŕäžèèől'ä¡ääijĂåŘŚåĞžæŻť åŁääď'■æ

äÿžäžĘåőđçŐřäÿĂäÿłçőĂå■ŢçŽĎRESTæŐěåŔčïijŇä¡ääŔłéIJĂèől'ä¡ăçŽĎçÍŃåžŔäžčçĂąæżąèűşPython
WSGIèćńæăĞåĞĘåžŞæŤŕæŇĄïijŇåŘŇæŮűäź§èćńçżİåď'ğéČĺåĹEçňňäÿĽ'æŰźwebæąĘæđűæŤŕæŇĄãĂĆ
åŻăæ■d'ïijŇåęĆædIJä¡ăçŽĎäžčçĂąéĄţå¿łè£ŹäÿłæăĞåĞĘïijŇåIJĺåŘŐéİćçŽĎä¡£çŤÍè£ĞçÍŃäÿ■åŕśäijŽæŻť åŁ

åIJÍWSGIäÿ■ïijŇä¡ääŔŕäžěåČŔäÿŃéİćè£ŹæăůçžęåőŽçŽĎæŰźåijŔäžěäÿĂäÿłåŔŕèŘçŤÍåŕźèśąå¡ćåijŔæİ

```
import cgi

def wsgi_app(environ, start_response):
    pass
```

environ åśđæĂğæŸŕäÿĂäÿłå■ŮåĚÿïijŇåŇĚåŘńäžĘäžŐwebæIJ■åŁąåŹĺåęĆA-
pache[åŘĆèĂČInternet RFC 3875]æŔŘä¿ŻçŽĎCGIæŐěåŔčäÿ■èŐůåŘŰçŽĎåĂijãĂĆ
èęĄåŕĘè£ŹäžŻäÿ■åŘŇçŽĎåĂijæŔŘåŘŰåĞžæİěïijŇä¡ääŔŕäžěåČŔè£ŹäźĹè£ŹæăůåĘŹïijŽ

```
def wsgi_app(environ, start_response):
    method = environ['REQUEST_METHOD']
    path = environ['PATH_INFO']
    # Parse the query parameters
    params = cgi.FieldStorage(environ['wsgi.input'],␣
↪environ=environ)
```

æĹŚäžňåśŢçd'žäžĘäÿĂäžŻåÿÿèğĄçŽĎåĂijãĂĆenviron['REQUEST_METHOD']
äžčèąĺèŕůæśĆçśžåđŃåęĆGETãĂĄPOSTãĂĄHEADç■L'ãĂĆ environ['PATH_INFO']
èąĺçd'žèćńèŕůæśĆèţĎæžŘçŽĎèůŕå¿ĎãĂĆ èŕČçŤÍ cgi.FieldStorage()
åŔŕäžěäžŐèŕůæśĆäÿ■æŔŘåŘŰæ§ěèŕćåŔĆæŢřåźűåŕĘåőČäžňæŤ¿åĚěäÿĂäÿłçśžå■ŮåĚÿåŕźèśąäÿ■äžěä¿£åŘŐ

start_response åŔĆæŢřæŸŕäÿĂäÿłäÿžäžĘåĹìåğŃåŇŰäÿĂäÿłèŕůæśĆåŕźèśąèĂŇå£Ěéąžèćńèŕ̆ĆçŤÍ
çňňäÿĂäÿłåŔĆæŢřæŸŕè£ŤåŻđçŽĎHTTPçŁűæĂĄåĂijïijŇçňňäžŇäÿłåŔĆæŢřæŸŕäÿĂäÿł(åŘ■,åĂij)åĚČçżĎ

```
def wsgi_app(environ, start_response):
    pass
    start_response('200 OK', [('Content-type', 'text/plain')])
```

äÿžäžĘè£ŤåŻđæŢřæ■őïijŇäÿĂäÿłWSGIçÍŃåžŔå£Ěéążè£ŤåŻđäÿĂäÿłå■ŮèŁĆå■Ůçňęäÿšåž
ŔåĹŮãĂĆåĚ

```
def wsgi_app(environ, start_response):
    pass
    start_response('200 OK', [('Content-type', 'text/plain')])
    resp = []
    resp.append(b'Hello World\n')
    resp.append(b'Goodbye!\n')
    return resp
```

æĹŰèĂĚïijŇä¡ăè£ŸåŔŕäžěä¡£çŤÍ `yield` ïijŽ

```
def wsgi_app(environ, start_response):
    pass
    start_response('200 OK', [('Content-type', 'text/plain')])
    yield b'Hello World\n'
    yield b'Goodbye!\n'
```

è£ŹéĞŇèęĄåijžèřČçŽĎäÿĂçĆźæŸŕæIJĂåŘŐè£ŤåŻđçŽĎå£ĚéążæŸŕå■ŮèŁĆå■Ůçňęäÿšã
å¡ŞçĎűïijŇáźűæšąæIJL'èęĄæśĆä¡ăè£ŤåŻđçŽĎäÿĂçĆźæŸŕæŰĞæIJñïijŇä¡ăåŔŕäžěå¿Ĺè¡żæİ¿çŽĎçijŰåĘŹäÿ

ář¡çőąWSGIçÍŃåžŔéĂŽåÿÿècńåőŽäźĽæĹŘäÿĂäÿłåĞ¡æŢřijŇäÿ■è£Ğä¡ääź§åŔŕäžěä¡£çŤÍçśžåőđä¿ŃæI
`__call__()` æŰźæşŢãĂĆä¿ŃåęĆïijŽ

```
class WSGIApplication:
    def __init__(self):
        ...
    def __call__(self, environ, start_response)
       ...
```

æĹŚäžňåůšçżŔåIJÍäÿŁéİćä¡£çŤÍè£Źçğ■æŁĂæIJŕåĹŻåžž `PathDispatcher` çśžãĂĆ
è£ŹäÿłåĹĘåŔŚåŹÍäżĚäżĚåŘłæŸŕçőąçŘĘäÿĂäÿłå■ŮåĚÿïijŇářĘ(æŰźæşŢ,èůŕå¿Ď)åŕźæŸĂářĎåĹŕåď'ĎçŘĘåŹÍ
å¡ŞäÿĂäÿłèŕůæśĆåĹŕæİěæŮűïijŇåőČçŽĎæŰźæşŢåŠŇèůŕå¿Ďèćńæŕŕ̊åŔŰåĞžæİěïijŇçĎűåŘŐèćńåĹĘåŔŚåĹ
åŔęåď'ŰïijŇäżżä¡Ţæ§ěèŕćåŔŸéĞŔäijŽèćńèğčæđŘåŘŐæŤ¿åĹŕäÿĂäÿłå■ŮåĚÿäÿ■ïijŇäžě
`environ['params']` å¡ćåijŔå■ŸåĆĺãĂĆ åŘŐéİćè£Źäÿłæ■ěéłď'åd'łåÿÿèğĄïijŇæL'Ääžěåžžèőőä¡ăåIJĺåĹĘ
ä¡£çŤĺåĹĘåŔŚåŹÍçŽĎæŮűåĂŹïijŇä¡ăåŔłéIJĂçőÄå■ŢçŽĎåĹŻåžžäÿĂäÿłåődä¿ŃïijŇçĎűåŘŐéĂŽè£ĞåőČæş
çijŰåĘŹè£ŹäžŽåĞ¡æŢřåžŤèŕěèűĚçžğçőÅå■ŢäžĘïijŇåŔłèęĄä¡äéĄţå¿ł
`start_response()` åĞ¡æŢřçŽĎçijŰåĘŹèğĎåĹŹïijŇáźűäÿŤæIJĂåŘŐè£ŤåŻđå■ŮèŁĆå■Ůçňęäÿšå■şåŔŕã

å¡ŞçijŰåĘŹè£Źçğ■åĞ¡æŢřçŽĎæŮűåĂŹè£ŸéIJĂæşĺæĎŔçŽĎäÿĂçĆźåŕśæŸŕåŕźäžŐå■Ůçňęäÿšæĺąæİ£çŽ
æšąäžžæĎ£æĎŔåĘŹéĆčçğ■åĹŕåď'ĎæůůåŘĹçİĂ `print()` åĞ¡æŢř ãĂĄXM-
LåŠŇåď'ğéĞŔæäijåijŔåŇŰæŞ■ä¡IJçŽĎäžčçăĄąĂĆ æĹŚäžňäÿŁéİćä¡£çŤĺäžĘäÿL'åijŢåŕůåŇĚåŘńçŽĎéćĎåĚ
è£Źçğ■æŰźåijŔçŽĎåŔŕäžěèőľ'æĹŚäžňå¿Ĺåőźæ ŸŞçŽĎåIJĺäžěåŘŐä£őæŤźè¿ŞåĞžæäijåijŔ(åŕłéIJĂèęĄä£őæ

æIJĂåŘŐïijŇä¡£çŤÍWSGIè£ŸæIJL'äÿĂäÿłå¿ĹéĞ■èęĄçŽĎéČĺåĹĘåŕśæŸŕæšąæIJL'äžĂäźĹåIJŕæŰźæŸŕé
åŻääÿžæăĞåĞĘåŕźäžŐæIJ■åŁąåŹĺåŠŇæąĘæđűæŸŕäÿ■çńÑçŽĎïijŇä¡ăåŔŕäžěåŕĘä¡äçŽĎçÍŃåžŔæŤ¿åĚäżžä
æĹŚäžňä¡£çŤĺäÿŃéİćçŽĎäžčçăĄąętŃèŕŢætŃèŕŢæIJñèŁĆäžčçăĄïijŽ

```
if __name__ == '__main__':
    from wsgiref.simple_server import make_server

    # Create the dispatcher and register functions
    dispatcher = PathDispatcher()
    pass

    # Launch a basic server
    httpd = make_server('', 8080, dispatcher)
    print('Serving on port 8080...')
    httpd.serve_forever()
```

äÿŁéÍćäžčçăĄåĹŻåžžäžĘäÿĂäÿłçőĂå■ŢçŽĎæIJ■åŁąåŹĺïijŇçĎűåŘŐä¡ååřšåŕŕäžěæİěæţŃèŕŢäÿŃä¡äçŽĎ
æIJĂåŘŐïijŇå¡Şä¡ăåĞĘåd'Ğè£ŻäÿĂæ■ěæL'ľåśŢä¡ăçŽĎçÍŃåžŔçŽĎæŮűåĂŹïijŇä¡ååŔŕäżěä£őæŤźè£Źäÿłäžĕ

WSGIæIJňèžńæŸŕäÿĂäÿłå¿ĹåřŔçŽĎæăĞåĞĘãĂĆåŻäæ■d'åőČåźűæšąæIJL'æŔŘä¿ŻäÿĂäžŻéńŸçžğçŽĎ
è£ŹäžŻä¡äèĞłåůšåőđçŐřèţůæİěäź§äÿ■éŽ¿ãĂĆäÿ■è£ĞåęĆæđIJä¡äæČşèęĄæŻt'åd'ŽçŽĎæŤŕæŇĄïijŇåŔŕäžěè
WebOb æĹŰèĂĚ Paste

11.6 éĂŽè£ĞXML-RPCåőđçŐřçőĂå■ŢçŽĎè£IJçÍŃèřČçŤÍ

éŮőécŸ

ä¡äæČşæL'¿åĹřäÿĂäÿłçőĂå■ŢçŽĎæŰźåijŔåŐżæL'ğèąŇè£ŘèąŇåIJÍè£IJçÍŃæIJžåŹÍäÿŁéÍćçŽĎPythonçÍ

èğčåĘşæŰźæąĹ

åőđçŐřäÿĂäÿłè£IJçÍŃæŰźæşŢèřČçŤÍçŽĎæIJĂçőĂå■ŢæŰźåijŔæŸŕä¡£çŤÍXML-
RPCãĂĆäÿŃéÍćæĹŠäžňæijŤçd'žäÿĂäÿŃäÿĂäÿłåőđçŐřäžĘéŤő-
åĂijå■ŸåĆÍåŁ§èČ¡çŽĎçőĂå■ŢæIJ■åŁąåŹĺïijŽ

```
from xmlrpc.server import SimpleXMLRPCServer

class KeyValueServer:
    _rpc_methods_ = ['get', 'set', 'delete', 'exists', 'keys']
    def __init__(self, address):
        self._data = {}
        self._serv = SimpleXMLRPCServer(address, allow_none=True)
        for name in self._rpc_methods_:
            self._serv.register_function(getattr(self, name))

    def get(self, name):
        return self._data[name]

    def set(self, name, value):
        self._data[name] = value
```

```
    def delete(self, name):
        del self._data[name]

    def exists(self, name):
        return name in self._data

    def keys(self):
        return list(self._data)

    def serve_forever(self):
        self._serv.serve_forever()

# Example
if __name__ == '__main__':
    kvserv = KeyValueServer(('', 15000))
    kvserv.serve_forever()
```

äÿŃéÌćæĹŚäżňäżŐäÿĂäÿłåőćæĹůçńŕæIJžåŹÍäÿŁéÌćæÌěè£éŮőæIJ■åŁąåŹÍïijŽ

```
>>> from xmlrpc.client import ServerProxy
>>> s = ServerProxy('http://localhost:15000', allow_none=True)
>>> s.set('foo', 'bar')
>>> s.set('spam', [1, 2, 3])
>>> s.keys()
['spam', 'foo']
>>> s.get('foo')
'bar'
>>> s.get('spam')
[1, 2, 3]
>>> s.delete('spam')
>>> s.exists('spam')
False
>>>
```

èőÍè0ž

XML-RPC åŔŕäżěèőľ'æĹŚäżňå¿Ĺåőźæ ŸŞçŽĎæđĎéĂääÿĂäÿłçőĂå■ŢçŽĎè£IJçÍŃèřČçŤÍæIJ■åŁąãĂĆä¡éĂŽè£ĞåőČçŽĎæŰźæşŢ `register_function()` æÌěæşÍåĘŇåĞ¡æŢřïijŇçĎůåŘŐä¡£çŤÍæŰźæşŢ `serve_forever()` åŘŕåŁÍåőČãĂĆ åIJÍäÿŁéÌćæĹŚäżňářĘè£ŹäžŻæ■ěéłď'æŤ¿åIJÍäÿĂètůåĘŹåĹřäÿĂäÿłçśż

```
from xmlrpc.server import SimpleXMLRPCServer
def add(x,y):
    return x+y

serv = SimpleXMLRPCServer(('', 15000))
serv.register_function(add)
serv.serve_forever()
```

XML-RPCæŽť'éIJšåĞžæİěçŽĎåĞ¡æŢřåŘłèČ¡éĂĆçŤĺäžŐéČĺåĹĘæŢřæ■őçśżåđŃïijŇæŕŤåęĆå■Ůçňęäÿš
åŕźäžŐåĚűäżŰçśżåđŃåřšå¿ŮéIJĂèęĄåĄŽäžŽéĆİåď'ŰçŽĎåŁ§èŕ¿äžĘãĂĆ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăæČşéĂŽè£Ğ XML-RPC äijăéĂŠäÿĂäÿłåŕźèśąåőđä¿ŃïijŇåőđéŹĚäÿŁåŘłæIJL'äżŰçŽĎ

```
>>> class Point:
...     def __init__(self, x, y):
...             self.x = x
...             self.y = y
...
>>> p = Point(2, 3)
>>> s.set('foo', p)
>>> s.get('foo')
{'x': 2, 'y': 3}
>>>
```

çśżäijijçŽĎïijŇåŕźäžŐäžŇè£ŻåĹűæŢřæ■őçŽĎåď'ĎçŘĘäź§èů§ä¡ăæČşèśąçŽĎäÿ■åď'łäÿĂæăůïijŽ

```
>>> s.set('foo', b'Hello World')
>>> s.get('foo')
<xmlrpc.client.Binary object at 0x10131d410>

>>> _.data
b'Hello World'
>>>
```

äÿĂèĹňæİěèőšïijŇä¡ääÿ■åžŤèŕěårĘ XML-RPC æIJ■åŁąäžěåĚňåĚśAPIçŽĎæŰźåijŔæŽť'éIJšåĞžæİěãĂĆ
åŕźäžŐè£Źçğ■æČĚåĘţïijŇéĂŽåÿÿåĹĘåÿČåijŔåžŤçŤĺçÍŇåžŔäijŽæŸŕäÿĂäÿłæŻť'åě¡çŽĎéĂĽ'æŃľãĂĆ

XML-RPCçŽĎäÿĂäÿłçijžçĆźæŸŕåőČçŽĎæĂğèČ¡ãĂĆ`SimpleXMLRPCServer`
çŽĎåőđçŐřæŸŕå■Ţçž£çÍŃçŽĎïijŇ æĽĂäžěåőČäÿ■éĂĆåŘĹäžŐåď'ğåđŃçÍŃåžŔïijŇår¡çőąæĹŚäžňåIJÍ11.2åřł
åŔęåď'ŰïijŇçŤśäžŐ XML-RPC åřĘæĽĂæIJL'æŢřæ■őéČ¡åžŔåĹŮåŇŰäÿžXMLæäijåijŔïijŇæĽĂäžěåőČäijŽ
ä¡ĘæŸŕåőČäź§æIJL'äijŸçĆźïijŇè£Źçğ■æŰźåijŔçŽĎçijŰçĂąåŕäžěèćńçżÍåď'ğéČĺåĹĘåĚűäżŰçijŰçÍŃèŕ■èÍĂ
éĂŽè£Ğä¡£çŤÍè£Źçğ■æŰźåijŔïijŇåĚűäżÛèŕ■èÍĂçŽĎåőćæĹůçńŕçÍŃåžŔéČ¡èČ¡è0£éŮőä¡ăçŽĎæIJ■åŁąãĂĆ

èŹ¡çĎűXML-RPCæIJL'å¿Ĺåď'ŽçijžçĆźïijŇä¡ĘæŸŕåęĆæđIJä¡ăéIJĂèęĄå£ńéĂ§æđĎåžžäÿĂäÿłçőĂå■Ţè£
æIJL'æŮűåĂŹïijŇçőĂå■ŢçŽĎæŰźæąĹåŕšåůšçżŔèűşåď'§äžĘãĂĆ

11.7 åIJÍäÿ■åŘŇçŽĎPythonèğčéĞŁåŹÍäźŃéŮť'äžď'äžŠ

éŮőécŸ

ä¡ăåIJÍäÿ■åŘŇçŽĎæIJžåŹÍäÿŁéÍć è£ŘèąŇçİĂåď'Žäÿł Pythonèğčé ĞŁåŹÍåőđä¿ŃïijŇåźűåÿŇæIJŻèČ¡åď'§å

èğčåĘşæŰźæąĹ

éĂŽè£Ğä¡£çŤÍ `multiprocessing.connection` æÍąåİŮåŘŕäžěå¿ĹåőźæŸŞçŽĎåőđçŐřèğčéĞŁåŹÍ
äÿŃéÍćæŸŕäÿĂäÿłçőĂå■ŢçŽĎåžŤç■ŤæIJ■åŁąåŹÍä¿Ńå■ŘïijŽ

```
from multiprocessing.connection import Listener
import traceback

def echo_client(conn):
    try:
        while True:
            msg = conn.recv()
            conn.send(msg)
    except EOFError:
        print('Connection closed')

def echo_server(address, authkey):
    serv = Listener(address, authkey=authkey)
    while True:
        try:
            client = serv.accept()

            echo_client(client)
        except Exception:
            traceback.print_exc()

echo_server(('', 25000), authkey=b'peekaboo')
```

çĎűåŘŐåőćæĹůçńŕè£đæŐěæIJ■åŁąåŹĺåźűåŔŚéĂĄæűĹæĄŕçŽĎçőĂå■Ţçđ'žä¿ŃïijŽ

```
>>> from multiprocessing.connection import Client
>>> c = Client(('localhost', 25000), authkey=b'peekaboo')
>>> c.send('hello')
>>> c.recv()
'hello'
>>> c.send(42)
>>> c.recv()
42
>>> c.send([1, 2, 3, 4, 5])
>>> c.recv()
[1, 2, 3, 4, 5]
>>>
```

èů§åžŢåśĆsocketäÿ■åŘŇçŽĎæŸŕïijŇæŕŔäÿłæűĹæĄŕäijŽåőŇæŢt'ä£Ĺå■ŸïijĹæŕŔäÿĂäÿłéĂŽè£ĞsendåŔęåđ'ŰïijŇæĽ'ĂæIJL'åŕźèśąäijŽéĂŽè£ĞpickleåžŔåĹŮåŇŰãĂĆåŽăæ■đ'ïijŇäżżä¡ŢåĚijåőźpickleçŽĎåŕźèśą

èőĺèőž

çŽőåL'■æIJL'å¿Ĺåđ'ŽçŤĺæĺěåőđçŐřåŘĎçğ■æűĹæĄŕäijăè¿ŞçŽĎåŇĚåŠŇåĞ¡æŢřåžŞïijŇæŕŤåęĆZeroMQ
ä¡ăè£ŸæIJL'åŔęåđ'ŰäÿĂçğ■éĂĽ'æŃľ'åŕśæŸŕèĞłåůśåIJĺåžŢåśĆsocketå§žçąĂäźŃäÿŁæĺěåőđçŐřäÿĂäÿłæűĹæ
ä¡ĘæŸŕä¡ăæČşèęĄçőĂå■ŢäÿĂçĆźçŽĎæŰźæąĹïijŇéĆčäźĹè£ŹæŮűåĂŹ
multiprocessing.connection åŕśæť¿äÿŁçŤĺåIJžäžĘãĂĆ
äžĚäžĚä¡£çŤĺäÿĂäžŽçőĂå■ŢçŽĎèŕ■åŔěå■şåŔřåőđçŐřåđ'ŽäÿłèğčéĞŁåŹĺäźŃéŮť'çŽĎæűĹæĄŕéĂŽä£ąãĂĆ

åęĆæđIJä¡ăçŽĎèğčéĞŁåŹÍè£ŘèąŇåIJÍåŘŇäÿĂåŕřæIJžåŹÍäÿŁéİćïijŇéĆčäźĹä¡ăåŕŕäžëä¡£çŤÍåŔęåď Ůç
èęĄæČşä¡£çŤÍUNIXå§§åěŮæŐěå■ŮæÍěåĹŻåžžäÿĂäÿłè£đæŐěïijŇåŘłéIJĂçőĂå■ŢçŽĎåřĘåIJřåİĂæŤźåĘŹäÿ

```
s = Listener('/tmp/myconn', authkey=b'peekaboo')
```

èęĄæČşä¡£çŤÍWindowsåŚ¡åŘ■çőąéĄŞæÍěåĹŻåžžè£đæŐěïijŇåŘłéIJĂåČŔäÿŃéİćè£ŹæăůäĄ£çŤÍäÿĂäÿła

```
s = Listener(r'\\.\pipe\myconn', authkey=b'peekaboo')
```

äÿĂäÿłéĂŽçŤÍåĞĘåĹŹæŸŕïijŇä¡ääÿ■èęĄä¡£çŤÍ multiprocessing æÍěåőđçŐřäÿĂäÿłåŕźåď ŮçŽĎåĚňåĚśæIJ■åŁąãĂĆ Client() åŠŇ Listener() äÿ■çŽĎ authkey åŔĆæŢřçŤÍæÍèèőď èŕĄåŔŚèţůè£đæŐěçŽĎçżĹçńŕçŤÍæĹůãĂĆ åęĆæđIJåŕĘéŠěäÿ■åŕźäijŽäžğçŤ§äÿĂäÿłåijĆåÿÿãĂĆæ■ď åď ŮïijŇèŕěæĺąåİŮæIJĂéĂĆåŘĹçŤÍæÍěåžžçńŇéŢ£
ä¿ŇåęĆïijŇäÿď äÿłèğčéĞŁåŹÍäźŇéŮť åŘŕåŁĺåŘŐåŕśåijĂägŇåžžçńŇè£đæŐěåźűåIJĺåď ĎçŘĘæ§ŘäÿłéŮőécŸ

åęĆæđIJä¡ăéIJĂèęĄåŕźåžŢåśĆè£đæŐěåĄŽæŽť åď ŽçŽĎæŐğåĹűïijŇæŕŤåęĆéIJĂèęĄæŤŕæŇĄèűĚæŮűã
ä¡ăæIJĂåě¡ä¡£çŤÍåŘęåď ŮçŽĎåžŞæĹŰèĂĚæŸŕåIJĺéńŸåśĆsocketäÿŁæÍěåőđçŐřè£ŹäžŽçĽźæĂğãĂĆ

11.8 åőđçŐřè£IJçÍŃæŰźæşŢèřČçŤÍ

éŮőécŸ

ä¡ăæČşåIJÍäÿĂäÿłæúĹæĄŕäijăè¿ŞåśĆåęĆ sockets ãĂĄmultiprocessing connections æĹŰ ZeroMQ çŽĎå§žçąĂäźŇäÿŁåőđçŐřäÿĂäÿłçőĂå■ŢçŽĎè£IJçÍŃè£ĞçÍŃèřČçŤÍïijĹRPC

èğčåĘşæŰźæąĹ

åřĘåĞ¡æŢřèřůæśĆãĂĄåŔĆæŢřåŠŇè£ŤåŽđåĂija¡£çŤÍpickleçijŰçăĄåŘŐïijŇåIJÍäÿ■åŘŇçŽĎèğčéĞŁåŹ
äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎPRCåď ĎçŘĘåŹÍïijŇåŘŕäžěèćńæŢť åŘĹåĽřäÿĂäÿłæIJ■åŁąåŹÍäÿ■åŐžïijŽ

```
# rpcserver.py

import pickle
class RPCHandler:
    def __init__(self):
        self._functions = { }

    def register_function(self, func):
        self._functions[func.__name__] = func

    def handle_connection(self, connection):
        try:
            while True:
                # Receive a message
                func_name, args, kwargs = pickle.loads(connection.
↪recv())
                # Run the RPC and send a response
```

```
                try:
                    r = self._functions[func_name](*args,**kwargs)
                    connection.send(pickle.dumps(r))
                except Exception as e:
                    connection.send(pickle.dumps(e))
        except EOFError:
            pass
```

èęĄä¡£çŤĺè£Źäÿłåd'ĎçŘĘåŹĺïijŇä¡ăéIJĂèęĄåřĘåőČåŁăåĚěåĹřäÿĂäÿłæűĹæĄŕæIJ■åŁąåŹĺäÿ■ãĂĆä¡ăæä¡ĘæŸŕä¡£çŤĺ multiprocessing åžŞæŸŕæIJĂçőĂå■ŢçŽĎãĂĆäÿŃéĺćæŸŕäÿĂäÿłR-PCæIJ■åŁąåŹĺä¿Ńå■Řïijž

```
from multiprocessing.connection import Listener
from threading import Thread

def rpc_server(handler, address, authkey):
    sock = Listener(address, authkey=authkey)
    while True:
        client = sock.accept()
        t = Thread(target=handler.handle_connection, args=(client,))
        t.daemon = True
        t.start()

# Some remote functions
def add(x, y):
    return x + y

def sub(x, y):
    return x - y

# Register with a handler
handler = RPCHandler()
handler.register_function(add)
handler.register_function(sub)

# Run the server
rpc_server(handler, ('localhost', 17000), authkey=b'peekaboo')
```

äÿžäžĘäżŐäÿĂäÿłè£IJçÍŃåőćæĹůçńŕèő£éŮőæIJ■åŁąåŹĺïijŇä¡ăéIJĂèęĄåĹŻåžžäÿĂäÿłåŕźåžŤçŽĎçŤĺæĺè

```
import pickle

class RPCProxy:
    def __init__(self, connection):
        self._connection = connection
    def __getattr__(self, name):
        def do_rpc(*args, **kwargs):
            self._connection.send(pickle.dumps((name, args,␣
↪kwargs)))
            result = pickle.loads(self._connection.recv())
```

```
            if isinstance(result, Exception):
                raise result
            return result
        return do_rpc
```

èęĄä¡£çŤĺè£ŹäÿłäžčçŘĘçśżïijŇä¡äéIJĂèęĄåřĘåĚűåŇĚèčĚåĹřäÿĂäÿłæIJ■åŁąåŹĺçŽĎè£đæŐěäÿŁéĺćïijŇ

```
>>> from multiprocessing.connection import Client
>>> c = Client(('localhost', 17000), authkey=b'peekaboo')
>>> proxy = RPCProxy(c)
>>> proxy.add(2, 3)

5
>>> proxy.sub(2, 3)
-1
>>> proxy.sub([1, 2], 4)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "rpcserver.py", line 37, in do_rpc
    raise result
TypeError: unsupported operand type(s) for -: 'list' and 'int'
>>>
```

èęĄæşĺæĎŘçŽĎæŸŕå¿Ĺåď ŽæűĹæĄŕåśĆïijĹæŕŤåęĆ `multiprocessing` ïijĽåůšçżŔä¡£çŤĺpickleåžŘåĹŮåŇŰäžĘæŢřæ■õãĂĆ åęĆæđIJæŸŕè£ŹæăůçŽĎèŕĺïijŇåŕź `pickle.dumps()` åŠŇ `pickle.loads()` çŽĎèřČçŤĺèęĄåŐżæŐĽ˜ãĂĆ

èőĺèőž

`RPCHandler` åŠŇ `RPCProxy` çŽĎå§žæIJňæĂİèůŕæŸŕå¿ĹæŕŤè¿ČçőĂå■ŢçŽĎãĂĆ åęĆæđIJäÿĂäÿłåőćæĹůçńŕæČşèęĄèřČçŤĺäÿĂäÿłè£IJçĺŇåĞ¡æŢřïijŇæŕŤåęĆ `foo(1, 2, z=3)` ,äžčçŘĘçśżåĹŻåžžäÿĂäÿłåŇĚåŘńäžĘåĞ¡æŢřåŘ■åŠŇåŔĆæŢřçŽĎåĚČçżĎ `('foo', (1, 2), {'z': 3})` ãĂĆ è£ŹäÿłåĚČçżĎèćńpickleåžŘåĹŮåŇŰåŘŐéĂŽè£Ğç¡ŚçżIJè£đæŐěåŘŚçŤ§åČ è£ŹäÿĂæ■ěåIJĺ `RPCProxy` çŽĎ `__getattr__()` æŰźæşŢè£ŤåŽđçŽĎ `do_rpc()` éŮ■åŇĚäÿ■åőŇæĹŘãĂĆ æIJ■åŁąåŹĺæŐěæŤűåŘŐéĂŽè£ĞpickleåŘ■åžŘåĹŮåŇŰæűĹæĄŕïijŇæ§ěæL˝¿åĞ¡æ æL˝ğèąŇçżŞædIJ(æĹŰåijĆåÿÿ)èćńpickleåžŘåĹŮåŇŰåŘŐè£ŤåŽđåŘŚéĂĄçżŹåőćæĹůçńŕãĂĆæĹŚäžňçŽĎåő `multiprocessing` è£ŽèąŇéĂŽä£ąãĂĆ äÿ■è£ĞïijŇè£ŹçğæŰźåijŔåŕŕäžěéĂĆçŤĺäžŐåĚűäžŰäžžä¡ŢæűĹ äžĚäžĚåŘłéIJĂèęĄåřĘè£đæŐěåŕźèśąæ■ćæĹŘåŘĹéĂĆçŽĎZeroMQçŽĎsocketåŕźèśąå■şåŘŕãĂĆ

çŤśäžŐåžŢåśĆéIJĂèęĄä¿ÌètŰpickleïijŇéĆčäźĹåőL˝åĚĺéŮőécŸåŕśéIJĂèęĄèĂČèŹŚäžĘ ïijĹåŽääÿžäÿĂäÿłèĄłæŸŐçŽĎéżŚåőćåŘŕäžěåĹŽåžžçL˝źåőŽçŽĎæűĹæĄŕïijŇèČ¡åď §èőľ˝äžžæĎŘåĞ¡æŢřéĂŽ åŽäæ■ď˝ä¡äæřÿè£IJäÿ■èęĄåĚĄèõÿæİěèĞłäÿ■ä£ąäžžæĹŰæIJłèőď˝èŕĄçŽĎåőćæĹůçńŕçŽĎRPCãĂĆçL˝źåĹńæ è£Źçğ■åŘłèČ¡åIJĺåĘĚéĆĺèćńä¡£çŤĺïijŇä¡■äžŐéŸšçĄńáćŹåŘŐéİćåźűäÿŤäÿ■èęĄåŕźåď ŰæŽť éIJšãĂĆ

ä¡IJäÿžpickleçŽĎæŻ£äžčïijŇä¡ääź§èőÿåŘŕäžěèĂČèŹŚä¡£çŤĺJSONãĂĄXMLæĹŰäÿĂäžŻåĚűäžŰçŽĎçi ä¿ŃåęĆïijŇæIJňæIJžåőđä¿ŃåŘŕäžěå¿Ĺåőźæ˜ŞçŽĎæŤźåĘŹæĹŘJSONçijŰçĂĄæŰźæąĹãĂĆè£ŸéIJĂèęĄåřĘ `pickle.loads()` åŠŇ `pickle.dumps()` æŻ£æ■ćæĹŘ `json.loads()` åŠŇ `json.dumps()` å■şåŘŕïijŽ

```
# jsonrpcserver.py
import json

class RPCHandler:
    def __init__(self):
        self._functions = { }

    def register_function(self, func):
        self._functions[func.__name__] = func

    def handle_connection(self, connection):
        try:
            while True:
                # Receive a message
                func_name, args, kwargs = json.loads(connection.
↪recv())
                # Run the RPC and send a response
                try:
                    r = self._functions[func_name](*args,**kwargs)
                    connection.send(json.dumps(r))
                except Exception as e:
                    connection.send(json.dumps(str(e)))
        except EOFError:
             pass

# jsonrpcclient.py
import json

class RPCProxy:
    def __init__(self, connection):
        self._connection = connection
    def __getattr__(self, name):
        def do_rpc(*args, **kwargs):
            self._connection.send(json.dumps((name, args, kwargs)))
            result = json.loads(self._connection.recv())
            return result
        return do_rpc
```

åőđçŐřRPCçŽĎäÿĂäÿłæŕŤè¿Čåď■æİĆçŽĎéŮőécŸæŸŕåęĆä¡ŢåŐžåď ĎçŘĘåijĆåÿÿãĂĆèĞşåřŚïijŇå¡Ş
åŻžæ■ď ïijŇè£ŤåŻđçżŹåőćæĹůçńŕçŽĎåijĆåÿÿæĽĂäžčèąÍçŽĎåŘńäźĽåřśèęĄåě¡åě¡èő¿èőąäžĘãĂĆ
åęĆæđIJä¡ăä¡£çŤÍpickleïijŇåijĆåÿÿåŕźèśąåőđä¿ŃåIJÍåőćæĹůçńŕèČ¡ècńåŘ■ážŘåĹŮåŇŰåźűæŁŻåĞžãĂĆåęĆ
äÿ■è£ĞèĞşåřŚïijŇä¡ăåžŤèŕěåIJÍåŞ■ážŤäÿ■è£ŤåŻđåijĆåÿÿå■Ůçňęäÿšã ĂĆæĹŚäžňåIJÍJSONçŽĎä¿Ńå■Řäÿ■å

åŕźäžŐåĚűäżŰçŽĎRPCåőđçŐřä¿Ńå■ŘïijŇæĹŚæŐÍè■Řä¡ăçIJŃçIJŃåIJÍXML-RPCäÿ■ä¡£çŤÍçŽĎ `SimpleXMLRPCServer` åŠŇ `ServerProxy` çŽĎåőđçŐřïijŇäźş§åřśæŸŕ11.6åřŔèŁĆäÿ■çŽĎåĘĚåőźãĂĆ

11.9 çőĂå■ŢçŽĎåőćæĹůçńŕèőď'èŕĄ

éŮőécŸ

ä¡ăæČşåIJĺåĹĘåÿČåijŔçşżçż§äÿ■åőđçŐřäÿĂäÿłçőĂå■ŢçŽĎåőćæĹůçńŕè£đæŐěèőď'èŕĄåŁ§èČ¡ïijŇåŔĹä

èğčåĘşæŰźæąĹ

åŔŕäžěåĹl'çŤĺ hmac æĺąåİŮåőđçŐřäÿĂäÿłè£đæŐěæŔąæL'ŃïijŇäžŐèĂŇåőđçŐřäÿĂäÿłçőĂå■ŢèĂŇénŸ

```
import hmac
import os

def client_authenticate(connection, secret_key):
    '''
    Authenticate client to a remote service.
    connection represents a network connection.
    secret_key is a key known only to both client/server.
    '''
    message = connection.recv(32)
    hash = hmac.new(secret_key, message)
    digest = hash.digest()
    connection.send(digest)

def server_authenticate(connection, secret_key):
    '''
    Request client authentication.
    '''
    message = os.urandom(32)
    connection.send(message)
    hash = hmac.new(secret_key, message)
    digest = hash.digest()
    response = connection.recv(len(digest))
    return hmac.compare_digest(digest,response)
```

å§žæIJňåŐ§çŘĘæŸŕå¡Şè£đæŐěåžžçńŇåŘŐïijŇæIJ■åŁąåŹĺçżŹåőćæĹůçńŕåŔŚéĂĄäÿĂäÿłéŽŔæIJžçŽĎ os.urandom() è£ŤåŽđåĂijïijL'ãĂĆ åőćæĹůçńŕåŠŇæIJ■åŁąåŹĺåŘŇæŮűåĹl'çŤĺh-macåŠŇäÿĂäÿłåŔŁæIJL'åŔŇæŰźç§ěéĄŞçŽĎåŕĘéŠěæİèèőąçőŮåĞžäÿĂäÿłåŁăåŕĘåŞĹåÿŇåĂijãĂĆçĎűåŘŐåæIJ■åŁąåŹĺéĂŽè£ĞæŕŤè¿Čè£ŹäÿłåĂijåŠŇèĞłåůśèőąçőŮçŽĎæŸŕåŘęäÿĂèĞt'æİěåĘşåőŽæŐěåŔŮæĹŰæŃŠ hmac.compare_digest() åĞ¡æŢřãĂĆ ä¡£çŤĺè£ŹäÿłåĞ¡æŢřåŔŕäžěéĄ£åĚ■éĄ■åĹřæŮűéŮt'åĹĘæđŘæŤ äÿžäžĘä¡£çŤĺè£ŹäžŽåĞ¡æŢřijŇä¡ăéIJĂèęĄåřĘåőČéŽĘæĹŘåĹřåůšæIJL'çŽĎç¡ŚçżIJæĹŰæűĹæĄŕäžčçăĄäÿ■

```
from socket import socket, AF_INET, SOCK_STREAM

secret_key = b'peekaboo'
def echo_handler(client_sock):
    if not server_authenticate(client_sock, secret_key):
        client_sock.close()
```

```
        return
    while True:

        msg = client_sock.recv(8192)
        if not msg:
            break
        client_sock.sendall(msg)

def echo_server(address):
    s = socket(AF_INET, SOCK_STREAM)
    s.bind(address)
    s.listen(5)
    while True:
        c,a = s.accept()
        echo_handler(c)

echo_server(('', 18000))

Within a client, you would do this:

from socket import socket, AF_INET, SOCK_STREAM

secret_key = b'peekaboo'

s = socket(AF_INET, SOCK_STREAM)
s.connect(('localhost', 18000))
client_authenticate(s, secret_key)
s.send(b'Hello World')
resp = s.recv(1024)
```

èőÍèőž

hmac èőd'èŕĄçŽĎäÿĂäÿłåÿÿèğĄä¡£çŤĺåIJžæŹŕæŸŕåĘĚéČĺæűĹæĄŕéĂŽä£ąçşżçż§åŠŇè£ŽçÍŃéŮt'éĂŽ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăçijŰåĘŹçŽĎçşżçż§æűĽ'åŔŁåĹŕäÿĂäÿłéŻĘç¿d'äÿ■åd'Žäÿłåd'ĎçŘĘåŹĺäźŃéŮt'çŽĎéĂ
ä¡ăåŔŕäžěä¡£çŤĺæIJňèŁĆæŰźæąĹæİěçąőä£İåŔłæIJL'èćńåĚĄèőÿçŽĎè£ŽçÍŃäźŃéŮt'æL'■èČ¡å¡ijæ■d'éĂŽä£ą
äžŃåőđäÿŁïijŇå§žäžŐ hmac çŽĎèőd'èŕĄèćń multiprocessing
æĺąåİŮä¡£çŤĺæİěåőđçŐřå■Řè£ZçÍŃçŻt'æŐěçŽĎéĂŽä£ąãĂĆ

è£ŸæIJL'äÿĂçĆźéIJĂèęĄåijžèřČçŽĎæŸŕè£đæŐěèőd'èŕĄåŠŇåŁăåŕĘæŸŕäÿd'çăĄäžŃãĂĆ
èőd'èŕĄæĹŘåŁ§äźŃåŘŐçŽĎéĂŽä£ąæűĹæĄŕæŸŕäžěæŸŐæŰĞå¡ćåijŕåŘŚéĂĄçŽĎïijŇäżżä¡Ţäžžåŕłèę
ĄæČ

hmacèőd'èŕĄçőŮæşŢå§žäžŐåŞĹåÿŇåĞ¡æŢřåęĆMD5åŠŇSHA-
1ïijŇåĚşäžŐè£ŹäÿłåIJĺIETF RFC 2104äÿ■æIJL'èŕęçżĘäżŃçż■ãĂĆ

11.10 ålJĺç¡ŚçżlJælJ■åŁąäÿ■åŁååĚěSSL

éŮőécŸ

ä¡äæČşåőđçŐřäÿĂäÿłå§žäžŐsocketsçŽĎç¡ŚçżIJæIJ■åŁąïijŇåőćæĹůçńřåŠŇæIJ■åŁąåŹĺéĂŽè£ĞSSLå■Ŕ

èğčåĘşæŰźæąĹ

ssl æĺąåİŮèČ¡äÿžåžŢåśĆsocketè£đæŐěæůżåŁăSSLçŽĎæŤŕæŇĄãĂĆ ssl.wrap_socket() åĞ¡æŢřæŐěåŔŮäÿĂäÿłåůšå■ŸåIJĺçŽĎsocketä¡IJäÿžåŔĆæŢřåźűä¡£çŤĺSSLåśĆæİěåŇĚèč ä¿ŇåęĆïijŇäÿŇéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎåžŤç■ŤæIJ■åŁąåŹĺïijŇèČ¡åIJĺæIJ■åŁąåŹĺçńŕäÿžæĽĂæIJL'åőćæĹ

```
from socket import socket, AF_INET, SOCK_STREAM
import ssl

KEYFILE = 'server_key.pem'    # Private key of the server
CERTFILE = 'server_cert.pem' # Server certificate (given to client)

def echo_client(s):
    while True:
        data = s.recv(8192)
        if data == b'':
            break
        s.send(data)
    s.close()
    print('Connection closed')

def echo_server(address):
    s = socket(AF_INET, SOCK_STREAM)
    s.bind(address)
    s.listen(1)

    # Wrap with an SSL layer requiring client certs
    s_ssl = ssl.wrap_socket(s,
                            keyfile=KEYFILE,
                            certfile=CERTFILE,
                            server_side=True
                            )
    # Wait for connections
    while True:
        try:
            c,a = s_ssl.accept()
            print('Got connection', c, a)
            echo_client(c)
        except Exception as e:
            print('{}: {}'.format(e.__class__.__name__, e))

echo_server(('', 20000))
```

äÿŃéİćæĹŚäžňæijŤçď'žäÿĂäÿłåőćæĹůçńŕè£đæŐěæIJ■åŁąåŹĺçŽĎäžď'äžŠä¿Ńå■ŘãĂĆåőćæĹůçńŕäijŽèŕů

```
>>> from socket import socket, AF_INET, SOCK_STREAM
>>> import ssl
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s_ssl = ssl.wrap_socket(s,
                            cert_reqs=ssl.CERT_REQUIRED,
                            ca_certs = 'server_cert.pem')
>>> s_ssl.connect(('localhost', 20000))
>>> s_ssl.send(b'Hello World?')
12
>>> s_ssl.recv(8192)
b'Hello World?'
>>>
```

è£Źçğ■çŻt'æŐěåď'ĎçŘĘåžŢåśĆsocketæŰźåijŔæIJL'äÿłéŮőéćŸåřśæŸŕåőČäÿ■èČ¡å¿Ĺåě¡çŽĎèů§æăĞåČ
ä¿ŃåęĆïijŇçżĺåď'ğéČĺåĹĘæIJ■åŁąåŹĺäžčçăĄïijĹHTTPãĂĄXML-
RPCç■L'ïijL'åőđéŹĚäÿŁæŸŕå§žäžŐ socketserver åžŞçŽĎãĂĆ
åőćæĹůçńŕäžčçăĄåIJĺäÿĂäÿłè¿ČéńŸåśĆäÿŁåőđçŐřãĂĆæĹŚäžňéIJĂèęĄåŘęåď'ŰäÿĂçğ■çĺ■å¿őäÿ■åŘŇçŽĎ

éęŰåĚĹïijŇårźäžŐæIJ■åŁąåŹĺèĂŇèĺĂïijŇåŘŕäžěéĂŽè£ĞåČŔäÿŃéİćè£Źæăůä¡£çŤĺäÿĂäÿłmixinçśžæİě

```
import ssl

class SSLMixin:
'''
Mixin class that adds support for SSL to existing servers based
on the socketserver module.
'''
def __init__(self, *args,
             keyfile=None, certfile=None, ca_certs=None,
             cert_reqs=ssl.NONE,
             **kwargs):
    self._keyfile = keyfile
    self._certfile = certfile
    self._ca_certs = ca_certs
    self._cert_reqs = cert_reqs
    super().__init__(*args, **kwargs)

def get_request(self):
    client, addr = super().get_request()
    client_ssl = ssl.wrap_socket(client,
                                 keyfile = self._keyfile,
                                 certfile = self._certfile,
                                 ca_certs = self._ca_certs,
                                 cert_reqs = self._cert_reqs,
                                 server_side = True)
    return client_ssl, addr
```

äÿžäžĘä¡£çŤĺè£ŹäÿłmixinçśžïijŇä¡ăåŘŕäžěåřĘåőČèů§åĚűäžŰæIJ■åŁąåŹĺçśžæůůåŘĹãĂĆä¿ŃåęĆïijŇäÿ
RPCæIJ■åŁąåŹĺä¿Ńå■ŘïijŽ

```
# XML-RPC server with SSL

from xmlrpc.server import SimpleXMLRPCServer

class SSLSimpleXMLRPCServer(SSLMixin, SimpleXMLRPCServer):
    pass

Here's the XML-RPC server from Recipe 11.6 modified only slightly␣
↪to use SSL:

import ssl
from xmlrpc.server import SimpleXMLRPCServer
from sslmixin import SSLMixin

class SSLSimpleXMLRPCServer(SSLMixin, SimpleXMLRPCServer):
    pass

class KeyValueServer:
    _rpc_methods_ = ['get', 'set', 'delete', 'exists', 'keys']
    def __init__(self, *args, **kwargs):
        self._data = {}
        self._serv = SSLSimpleXMLRPCServer(*args, allow_none=True,␣
↪**kwargs)
        for name in self._rpc_methods_:
            self._serv.register_function(getattr(self, name))

    def get(self, name):
        return self._data[name]

    def set(self, name, value):
        self._data[name] = value

    def delete(self, name):
        del self._data[name]

    def exists(self, name):
        return name in self._data

    def keys(self):
        return list(self._data)

    def serve_forever(self):
        self._serv.serve_forever()

if __name__ == '__main__':
    KEYFILE='server_key.pem'     # Private key of the server
    CERTFILE='server_cert.pem'   # Server certificate
    kvserv = KeyValueServer(('', 15000),
                            keyfile=KEYFILE,
                            certfile=CERTFILE),
```

```
    kvserv.serve_forever()
```

ä¡£çŤĺè£ŹäÿłæIJ■åŁąåŹĺæŮűïijŇä¡ăåŔŕäžëä¡£çŤĺæŹőéĂŽçŽĎ `xmlrpc.client` æĺąåİŮæİêè£đæŐěåőČãĂĆ åŔłéIJĂèęĄåIJĺURLäÿ■æŇĞåőŽ `https:` å■şåŔŕïijŇä¿ŃåęĆïijŽ

```
>>> from xmlrpc.client import ServerProxy
>>> s = ServerProxy('https://localhost:15000', allow_none=True)
>>> s.set('foo','bar')
>>> s.set('spam', [1, 2, 3])
>>> s.keys()
['spam', 'foo']
>>> s.get('foo')
'bar'
>>> s.get('spam')
[1, 2, 3]
>>> s.delete('spam')
>>> s.exists('spam')
False
>>>
```

åŕźäžŐSSLåőćæĹůçńŕæİêèőšäÿĂäÿłæŕŤè¿Čåď■æİĆçŽĎéŮőécŸæŸŕåęĆä¡Ţçąőèőď'æIJ■åŁąåŹĺèŕĄäźęäÿ■åźÿçŽĎæŸŕïijŇæŽĆæŮűè£Ÿæšąæ IJL'äÿĂäÿłæăĞåĞĘæŰźæşŢæİêèğčåĘşè£ŹäÿłéŮőécŸïijŇéIJĂèęĄèĞäÿ■è£ĞïijŇäÿŃéİćçżŹåĞžäÿĂäÿłä¿Ńå■ŘïijŇçŤĺæİêåžžçńŃäÿĂäÿłåőL'åĚĺçŽĎXML-RPCè£đæŐěæİêçąőèőď'æIJ■åŁąåŹĺèŕĄäźęïijŽ

```
from xmlrpc.client import SafeTransport, ServerProxy
import ssl

class VerifyCertSafeTransport(SafeTransport):
    def __init__(self, cafile, certfile=None, keyfile=None):
        SafeTransport.__init__(self)
        self._ssl_context = ssl.SSLContext(ssl.PROTOCOL_TLSv1)
        self._ssl_context.load_verify_locations(cafile)
        if cert:
            self._ssl_context.load_cert_chain(certfile, keyfile)
        self._ssl_context.verify_mode = ssl.CERT_REQUIRED

    def make_connection(self, host):
        # Items in the passed dictionary are passed as keyword
        # arguments to the http.client.HTTPSConnection()␣
↪constructor.
        # The context argument allows an ssl.SSLContext instance to
        # be passed with information about the SSL configuration
        s = super().make_connection((host, {'context': self._ssl_
↪context}))

        return s

# Create the client proxy
s = ServerProxy('https://localhost:15000',
```

```
                transport=VerifyCertSafeTransport('server_cert.pem
↪'),
                allow_none=True)
```

æIJ■åŁąåŹĺåřĘèŕĄäźęåŘŚéĂĄçżŹåőćæĹůçńřïijŇåőćæĹůçńřæİěçąőèőď'åőČçŽĎåŘĹæşŢæĂğãĂĆè£ŹçğåęĆæđIJæIJ■åŁąåŹĺæČşèęĄçąőèőď'åőćæĹůçńřïijŇåŘŕäžěåřĘæIJ■åŁąåŹĺåŘŕåŁĺäžčçăĄä£őæŤźåęĆäÿŃïijŽ

```
if __name__ == '__main__':
    KEYFILE='server_key.pem'    # Private key of the server
    CERTFILE='server_cert.pem' # Server certificate
    CA_CERTS='client_cert.pem' # Certificates of accepted clients

    kvserv = KeyValueServer(('', 15000),
                            keyfile=KEYFILE,
                            certfile=CERTFILE,
                            ca_certs=CA_CERTS,
                            cert_reqs=ssl.CERT_REQUIRED,
                            )
    kvserv.serve_forever()
```

äÿžäžĘèől'XML-RPCåőćæĹůçńřåŘŚéĂĄèŕĄäźęïijŇä£őæŤźçŽĎåĹİåğŃåŇŰäžčçăĄåęĆäÿŃïijŽ

```
# Create the client proxy
s = ServerProxy('https://localhost:15000',
                transport=VerifyCertSafeTransport('server_cert.pem',
                                                  'client_cert.pem',
                                                  'client_key.pem'),
                allow_none=True)
```

èőĺèőž

èŕŢçİĂåŐżè£ŘèąŇæIJňèŁĆçŽĎäžčçăĄèČ¡æţŃèŕŢä¡ăçŽĎçşżçż§éĚ■ç¡őèČ¡åŁŻåŠŇçŘĘèğčSSLãĂĆ åŘŕèČ¡æIJĂåď'ğçŽĎæŇŚæĹŸæŸŕåęĆä¡ŢäÿĂæ■ěæ■ěçŽĎèŐůåŘŰåĹİåğŃéĚ■ç¡őkeyãĂĄèŕĄäźęåŠŇåĚűäžŰ

æĹŚèğčéĞŁäÿŃåĹřåžŢéIJĂèęĄåŢëïijŇærŔäÿĂäÿłSSLè£đæŐěçżĹçńŕäÿĂèĹňéČ¡äijŽæIJL'äÿĂäÿłçğĄé è£ŹäÿłèŕĄäźęåŇĚåŘńäžĘåĚňéŠěåźűåIJĺæŕŔäÿĂæňąè£đæŐěçŽĎæŮűåĂŹéČ¡äijŽåŘŚéĂĄçżŹåŕźæŰźãĂĆ åŕźäžŐåĚňåĚśæIJ■åŁąåŹĺïijŇåőČäžňçŽĎèŕĄäźęéĂŽåÿÿæŸŕèćńæİČåĺĄèŕĄäźęæIJžædĎæŕŤåęĆVerisignãĂA äÿžäžĘçąőèőď'æIJ■åŁąåŹĺç■¿åŘ■ïijŇåőćæĹůçńřåŽđä£Ìå■ŸäÿĂäż¡åŇĚåŘńäžĘä£ąäżżæŐĹæİČæIJžædĎçŽĎ ä¿ŃåęĆïijŇwebæţŔèğĹåŹĺä£Ìå■ŸäžĘäÿżèęĄçŽĎèőď'èŕĄæIJžædĎçŽĎèŕĄäźęïijŇåźűä¡£çŤĺåőČæİěäÿżærŔä åŕźæIJňåŕŔèŁĆçď'žä¿ŃèĂŇèĺĂïijŇåŘłæŸŕäÿžäžĘæţŃèŕŢïijŇæĹŚäžňåŘŕäžěåĹŻåżżèĞłç■¿åŘ■çŽĎèŕĄäźęïij

bash % openssl req -new -x509 -days 365 -nodes -out server_cert.pem
-keyout server_key.pem

Generating a 1024 bit RSA private key .....++++++
...++++++

writing new private key to 'server_key.pem'

You are about to be asked to enter information that will be incorporated into your certificate request. What you are about to enter is what is called a Distinguished Name or a DN. There are quite a few fields but you can leave some blank For some fields there will be a default value, If you enter '.', the field will be left blank.

Country Name (2 letter code) [AU]:US State or Province Name (full name) [Some-State]:Illinois Locality Name (eg, city) []:Chicago Organization Name (eg, company) [Internet Widgits Pty Ltd]:Dabeaz, LLC Organizational Unit Name (eg, section) []: Common Name (eg, YOUR name) []:localhost Email Address []: bash %

åIJĺåĹŻåžžèŕĄäźęçŽĎæŮűåĂŹïijŇåŘĎäÿłåĂijçŽĎèő¿åőŽåŔŕäżěæŸŕäżżæĎŔçŽĎïijŇä¡ĘæŸŕâĂİComm
NameâĂIJçŽĎåĂijéĂŽåÿÿèęĄåŇĚåŘńæIJ■åŁąåŹĺçŽĎDNSäÿżæIJžåŘ■ãĂĆ
åęĆæđIJä¡ääŔłæŸŕåIJĺæIJňæIJžæţŃèŕŢïijŇéĆčäźĹåřśä¡£çŤĺâĂİlocalhostâĂIJïijŇåŘęåĹŹä¡£çŤĺæIJ■åŁąåŹĺç

——BEGIN RSA PRIVATE KEY—— MIICXQIBAAKBgQCZr-
CNLoEyAKF+f9UNcFaz5Osa6jf7qkbUl8si5xQrY3ZYC7juu
nL1dZLn/VbEFIITaUOgvBtPv1qUWTJGwga62VSG1oFE0ODIx3g2Nh4sRf+rySsx2
L4442nx0z4O5vJQ7k6eRNHAZUUnCL50+YvjyLyt7ryLSjSuKhCcJsbZgPwIDAQAB
AoGAB5evrr7eyL4160tM5rHTeATlaLY3UBOe5Z8XN8Z6gLiB/ucSX9AysviVD/6F
3oD6z2aL8jbeJc1vHqjt0dC2dwwm32vVl8mRdyoAsQpWmiqXrkvP4Bsl04VpBeHw
Qt8xNSW9SFhceL3LEvw9M8i9MV39viih1ILyH8OuHdvJyFECQQDLEjl2d2ppxND9
PoLqVFAirDfX2JnLTdWbc+M11a9Jdn3hKF8TcxfEnFVs5Gav1MusicY5KB0ylYPb
YbTvqKc7AkEAwbnRBO2VYEZsJZp2X0IZqP9ovWokkpYx+PE4+c6MySDgaMcigL7v
WDIHJG1CHudD09GbqENasDzyb2HAIW4CzQJBAKDdkv+xoW6gJx42Auc2WzTcUHCA
eXR/+BLpPrhKykzbvOQ8YvS5W764SUO1u1LWs3G+wnRMvrRvlMCZKgggBjkCQQCG
Jewto2+a+WkOKQXrNNScCDE5aPTmZQc5waCYq4UmCZQcOjkUOiN3ST1U5iuxRqfb
V/yX6fw0qh+fLWtkOs/JAkA+okMSxZwqRtfgOFGBfwQ8/iKrnizeanTQ3L6scFXI
CHZXdJ3XQ6qUmNxNn7iJ7S/LDawo1QfWkCfD9FYoxBlg ——END RSA PRI-
VATE KEY——

æIJ■åŁąåŹĺèŕĄäźęæŰĞäżűserver_cert.pemåĘĚåőźçśżäijijäÿŃéİćè£Źæăůïijž

——BEGIN CERTIFICATE—— MIIC+DCCAmGgAwIBAgIJAPMd+vi45js3MA0GCSqGSIb3DQEBB
BAYTAlVTMREwDwYDVQQIEwhJbGxpbm9pczEQMA4GA1UEBxMHQ2hpY2FnbzEUMBIG
A1UEChMLRGFiZWF6LCBMTEMxEjAQBgNVBAMTCWxvY2FsaG9zdDAeFw0xMzAxMTEx
ODQyMjdaFw0xNDAxMTExODQyMjdaMFwxCzAJBgNVBAYTAlVTMREwDwYDVQQIEwhJ
bGxpbm9pczEQMA4GA1UEBxMHQ2hpY2FnbzEUMBIGA1UEChMLRGFiZWF6LCBMTEMx
EjAQBgNVBAMTCWxvY2FsaG9zdDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
mawjS6BMgChfn/VDXBWs+TrGuo3+6pG1JfLIucUK2N2WAu47rpy9XWS5/1WxBSCE
2lDoLwbT79alFkyRsIGutlUhtaBRNDgyMd4NjYeLEX/q8krMdi+OONp8dM+DubyU

O5OnkTRwGVFJwi+dPmL48i8re68i0o0rioQnCbG2YD8CAwEAAaOBwTCBvjAdBgNV
HQ4EFgQUrtoLHHgXiDZTr26NMmgKJLJLFtIwgY4GA1UdIwSBhjCBg4AUrtoLHHgX
iDZTr26NMmgKJLJLFtKhYKReMFwxCzAJBgNVBAYTAlVTMREwDwYDVQQIEwhJbGxp
bm9pczEQMA4GA1UEBxMHQ2hpY2FnbzEUMBIGA1UEChMLRGFiZWF6LCBMTEMxEjAQ
BgNVBAMTCWxvY2FsaG9zdIIJAPMd+vi45js3MAwGA1UdEwQFMAMBAf8wDQYJKoZI
hvcNAQEFBQADgYEAFci+dqvMG4xF8UTnbGVvZJPIzJDRee6Nbt6AHQo9pOdAIMAu

WsGCplSOaDNdKKzl+b2UT2Zp3AIW4Qd51bouSNnR4M/gnr9ZD1ZctFd3jS+C5XRp
D3vvcW5lAnCCC80P6rXy7d7hTeFu5EYKtRGXNvVNd/06NALGDflrrOwxF3Y=
——END CERTIFICATE——

åIJÍæIJ■åŁąåŹÍçńŕäžčçăĄäÿ■ïijŇçğĄéŠěåŠŇèŕĄäźęæŰĞäžűäijŽèćńäijăçżŹSSLçŻÿåĚşçŽĎåŇĚèčĚåĞ¡çğĄéŠěåžŤèŕěåIJÍä£Ìå■ŸåIJÍæIJ■åŁąåŹÍäÿ■ïijŇåźűåŁääžěåőĽåĚÍä£ÌæŁd'ãĂĆ

åIJÍåőćæĹůçńŕäžčçăĄäÿ■ïijŇéIJĂèęĄä£Ìå■ŸäÿĂäÿłåŘĹæşŢèŕĄäźęæŐĹæİČæŰĞäžűæİěçąőèőd'æIJ■åŁąęĆæđIJä¡ăæšąæIJL'è£ŹäÿłæŰĞäžűïijŇä¡ääŔŕäžěåIJÍåőćæĹůçńŕåd'■åĹűäÿĂäż¡æIJ■åŁąåŹÍçŽĎèŕĄäźęåźűä¡è£đæŐěåžžçńŇåŘŐïijŇæIJ■åŁąåŹÍäijŽæŔŘä¿ŻåőČçŽĎèŕĄäźęïijŇçĎůåŘŐä¡ääřśèČ¡ä¡£çŤĺåůšçżŔä£Ìå■ŸçŽ

æIJ■åŁąåŹÍäź§èČ¡éĂL'æŃl'æŸŕåŘęèęĄçąőèőd'åőćæĹůçńŕçŽĎèžńäż¡ãĂĆåęĆæđIJèęĄè£Źæăůå ĄŽçŽĎ
æIJ■åŁąåŹÍäź§éIJĂèęĄä£Ìå■ŸäÿĂäÿłèćńä£qäżżèŕĄäźęæŐĹæİČæŰĞäžűæİěçąőèőd'åőćæĹůçńŕèŕĄäźęãĂĆ

åęĆæđIJä¡ăèęĄåIJÍçIJ§åőđçŐŕåćČäÿ■äÿžä¡ăçŽĎç¡ŚçżIJæIJ■åŁąåŁääÿŁSSLçŽĎæŤŕæŇĄïijŇè£ŹåřŔèŁ
ä¡ăè£ŸåžŤèŕěåŔĆèĂČåĚűäżŰçŽĎæŰĞæąčïijŇåĄŽåě¡èŁśèť źäÿ■åřŚæŮűéŮť'æİěæţŃèŕŢåőČæ■čåÿÿåůěä¡IJç
^_^

11.11 è£ŻçÍŃéŮť'äijăéĂŠSocketæŰĞäžűæŔŘè£řçňę

éŮőécŸ

ä¡ăæIJL'åd'ŽäÿłpythonèğčéĞŁåŹÍè£ŻçÍŃåIJÍåŘŇæŮűè£ŘèąŇïijŇä¡ăæČşářĘæ§Řäÿłæ L'ŞåijĂçŽĎæŰĞä
ærŤåęĆïijŇåĄĞèő¿æIJL'äÿłæIJ■åŁąåŹÍè£ŻçÍŃçŻÿåžŤè£đæŐěèřůæśĆïijŇä¡ĘæŸŕåőđéŹĚçŽĎçŻÿåžŤéĂżè¿Ś

èğčåĘşæŰźæąĹ

äÿžäžĘåIJÍåd'Žäÿłè£ŻçÍŃäÿ■äijăéĂŠæŰĞäžűæŔŘè£řçňęïijŇä¡ăéęŰåĚĹéIJĂèęĄářĘåőČäžňè£đæŐěåĹř
èĂŇåIJÍwindowsäÿŁéÍćä¡ăéIJĂèęĄä¡£çŤĺåŚ¡åŘ■çőąéĄŞãĂĆäÿ■è£Ğä¡ăæŮäéIJĂçIJ§çŽĎéIJĂèęĄåŐżæŞ■ä¡
éĂŽåÿÿä¡£çŤÍ `multiprocessing` æĺąåİŮæİěåĹŻåžžè£ŹæăůçŽĎè£đæŐěäijŽæŻťåőźæŸŞäÿĂäžŽãĂĆ

äÿĂæŮęäÿĂäÿłè£đæŐěèćńåĹŻåžžïijŇä¡ăåŔŕäžěä¡£çŤÍ `multiprocessing.reduction` äÿ■çŽĎ `send_handle()` åŠŇ `recv_handle()` åĞ¡æŢřåIJÍäÿ■åŘŇçŽĎåd'ĎçŘĘåŹÍçŻť'æŐěäijăéĂŠæŰĞäžűæŔŘè£řçňęãĂĆ
äÿŃéÍćçŽĎä¿Ńå■ŘæijŤçd'žäžĘæIJĂå§žæIJňçŽĎçŤÍæşŢïijŽ

```
import multiprocessing
from multiprocessing.reduction import recv_handle, send_handle
import socket

def worker(in_p, out_p):
    out_p.close()
    while True:
        fd = recv_handle(in_p)
        print('CHILD: GOT FD', fd)
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM,␣
↪fileno=fd) as s:
            while True:
```

```
                msg = s.recv(1024)
                if not msg:
                    break
                print('CHILD: RECV {!r}'.format(msg))
                s.send(msg)

def server(address, in_p, out_p, worker_pid):
    in_p.close()
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, True)
    s.bind(address)
    s.listen(1)
    while True:
        client, addr = s.accept()
        print('SERVER: Got connection from', addr)
        send_handle(out_p, client.fileno(), worker_pid)
        client.close()

if __name__ == '__main__':
    c1, c2 = multiprocessing.Pipe()
    worker_p = multiprocessing.Process(target=worker, args=(c1,c2))
    worker_p.start()

    server_p = multiprocessing.Process(target=server,
                   args=(('', 15000), c1, c2, worker_p.pid))
    server_p.start()

    c1.close()
    c2.close()
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijÑäÿd'äÿłè£ŻçÍÑèćńåĹŻåżžåźűéĂŽè£ĞäÿĂäÿł
`multiprocessing` çőąéĄŞè£đæŐěèţůæİěãĂĆ æIJ■åŁąåŹÍè£ŻçÍÑæL'ŞåijĂäÿĂäÿłsocketåźűç■L'å¿Ěåőćæĺ
åůěä¡IJè£ŻçÍÑäżĚäżĚä¡£çŤÍ `recv_handle()` åIJÍçőąéĄŞäÿŁéÍćç■L'å¿ĚæŐěæŤűäÿĂäÿłæŰĞäżűæŔŘè£řç
å¡ŞæIJ■åŁąåŹÍæŐěæŤűåĹřäÿĂäÿłè£đæŐëïijŇåőČåřĘäžğçŤ§çŽĎsocketæŰĞäżűæŔŘè£řçňęéĂŽè£Ğ
`send_handle()` äijäéĂŠçżŹåůěä¡IJè£ŻçÍÑãĂĆ åůěä¡IJè£ŻçÍÑæŐěæŤűåĹřsocketåŘŐåŘŠåőćæĹůçńŕåŽđå

åęĆæđIJä¡ăä¡£çŤÍTelnetæĹŰçśżäijijåůěåĚůè£đæŐěåĹřæIJ■åŁąåŹÍïijÑäÿŃéÍćæŸŕäÿĂäÿłæijŤçd'žä¿Ńå■

bash % python3 passfd.py SERVER: Got connection from ('127.0.0.1', 55543)
CHILD: GOT FD 7 CHILD: RECV b'Hellorn' CHILD: RECV b'Worldrn'

æ■d'ä¿ŃæIJĂéĞ■èęĄçŽĎéČÍåĹĘæŸŕæIJ■åŁąåŹÍæŐěæŤűåĹřçŽĎåőćæĹůçńŕsocketåőđéŹĚäÿŁèćńåŘęå
æIJ■åŁąåŹÍäżĚäżĚåŔłæŸŕářĘåĚűè¡ňæL'ŃåźűåĕşéŮ■æ■d'è£đæŐěïijÑçĎűåŘŐç■L'å¿ĚäÿŃäÿĂäÿłè£đæŐěãĂ

èőÍèőž

åŕźäžŐåd'ğéČÍåĹĘçÍÑåžŔåŚŸæİèèőšåIJÍäÿ■åŘÑè£ŻçÍÑäźŃéŮt'äijäéĂŠæŰĞäżűæŔŘè£řçňęåě¡åČŔæšą
ä¡ĘæŸŕijÑæIJL'æŮűåĂŹåőĆæŸŕædĎåžžäÿĂäÿłåŔŕæL'l'åśŢçşżçż§çŽĎå¿ĹæIJL'çŤÍçŽĎåůěåĚůãĂĆä¿ŃåęĆ
ä¡ååŔŕäžěæIJL'åd'ŽäÿłpythonèğčéĞŁåŹÍåőđä¿ŃïijÑářĘæŰĞäżűæŔŘè£řçňęäijäéĂŠçżŹåĚűåőĆèğčéĞŁåŹÍæ

send_handle() åŠŇ recv_handle() åĞ¡æŢřåŔłèČ¡åď§çŤĺäžŐ multiprocessing è£đæŐěãĂĆ ä¡£çŤĺåőČäžňæİěäžčæŻ£çőąéĄŞçŽĎä¡£çŤĺïijĹåŔĆèĂČ11.7èŁĆïijĽïijŇä¿ÑåęĆïijŇä¡ăåŔŕäżěèőľ'æIJ■åŁąåŹĺåŠŇåůěä¡IJèĂĚåŘĎèĞłäžěå■ŢçŃňçŽĎçÍŇåžŔæİěåŕŕåŁĺãĂĆäÿŇéİćæŸ

```
# servermp.py
from multiprocessing.connection import Listener
from multiprocessing.reduction import send_handle
import socket

def server(work_address, port):
    # Wait for the worker to connect
    work_serv = Listener(work_address, authkey=b'peekaboo')
    worker = work_serv.accept()
    worker_pid = worker.recv()

    # Now run a TCP/IP server and send clients to worker
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, True)
    s.bind(('', port))
    s.listen(1)
    while True:
        client, addr = s.accept()
        print('SERVER: Got connection from', addr)

        send_handle(worker, client.fileno(), worker_pid)
        client.close()

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 3:
        print('Usage: server.py server_address port', file=sys.
↪stderr)
        raise SystemExit(1)

    server(sys.argv[1], int(sys.argv[2]))
```

è£ŘèąŇè£ŹäÿłæIJ■åŁąåŹĺïijŇåŔłéIJĂèęĄæĽğèąŇ *python3 servermp.py /tmp/servconn 15000* ïijŇäÿŇéİćæŸŕçŻÿåžŢçŽĎåůěä¡IJèĂĚäžčçăĄïijŽ

```
# workermp.py

from multiprocessing.connection import Client
from multiprocessing.reduction import recv_handle
import os
from socket import socket, AF_INET, SOCK_STREAM

def worker(server_address):
    serv = Client(server_address, authkey=b'peekaboo')
    serv.send(os.getpid())
    while True:
        fd = recv_handle(serv)
```

```
        print('WORKER: GOT FD', fd)
        with socket(AF_INET, SOCK_STREAM, fileno=fd) as client:
            while True:
                msg = client.recv(1024)
                if not msg:
                    break
                print('WORKER: RECV {!r}'.format(msg))
                client.send(msg)

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 2:
        print('Usage: worker.py server_address', file=sys.stderr)
        raise SystemExit(1)

    worker(sys.argv[1])
```

èęĄè£ŘèąŇåůěä¡IJèĂĚïijŇæĽ'ğèąŇæĽ'ğèąŇåŚ¡äžď' *python3 workermp.py /tmp/servconn* . æŢĹædIJèů§ä¡£çŤĺPipe()ä¿Ńå■ŘæŸŕåőŇåĚĺäÿĂæăůçŽĎãĂĆ æŰĞäżűæŔŔè£ŕçňęçŽĎäijăéĂŠäijŽæűĽ'åŔŁåĹřUNIXå§§åěŮæŐěå■ŮçŽĎåĹŻåžžåŠŇåěŮæŐěå■ŮçŽĎ `sendmsg()` æŰźæşŢãĂĆ äÿ■è£Ğè£ŹçğæŁĂæIJŕåźűäÿ■åÿÿèğĄïijŇäÿŃéĺćæŸŕä¡£çŤĺåěŮæŐěå■Ůæİěäijăé

```
# server.py
import socket

import struct

def send_fd(sock, fd):
    '''
    Send a single file descriptor.
    '''
    sock.sendmsg([b'x'],
                 [(socket.SOL_SOCKET, socket.SCM_RIGHTS, struct.
↪pack('i', fd))])
    ack = sock.recv(2)
    assert ack == b'OK'

def server(work_address, port):
    # Wait for the worker to connect
    work_serv = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
    work_serv.bind(work_address)
    work_serv.listen(1)
    worker, addr = work_serv.accept()

    # Now run a TCP/IP server and send clients to worker
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, True)
    s.bind(('',port))
    s.listen(1)
    while True:
```

```
        client, addr = s.accept()
        print('SERVER: Got connection from', addr)
        send_fd(worker, client.fileno())
        client.close()

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 3:
        print('Usage: server.py server_address port', file=sys.
↪stderr)
        raise SystemExit(1)

    server(sys.argv[1], int(sys.argv[2]))
```

äÿŃéİćæŸŕä¡£çŤĺåěŮæŐěå■ŮçŽĎåůěä¡IJèĂĚåőđçŐřïijŽ

```
# worker.py
import socket
import struct

def recv_fd(sock):
    '''
    Receive a single file descriptor
    '''
    msg, ancdata, flags, addr = sock.recvmsg(1,
                                    socket.CMSG_LEN(struct.
↪calcsize('i')))

    cmsg_level, cmsg_type, cmsg_data = ancdata[0]
    assert cmsg_level == socket.SOL_SOCKET and cmsg_type == socket.
↪SCM_RIGHTS
    sock.sendall(b'OK')

    return struct.unpack('i', cmsg_data)[0]

def worker(server_address):
    serv = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
    serv.connect(server_address)
    while True:
        fd = recv_fd(serv)
        print('WORKER: GOT FD', fd)
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM,␣
↪fileno=fd) as client:
            while True:
                msg = client.recv(1024)
                if not msg:
                    break
                print('WORKER: RECV {!r}'.format(msg))
                client.send(msg)
```

```
if __name__ == '__main__':
    import sys
    if len(sys.argv) != 2:
        print('Usage: worker.py server_address', file=sys.stderr)
        raise SystemExit(1)

    worker(sys.argv[1])
```

åęĆæđIJä¡ăæČşåIJĺä¡ăçŽĎçÍŃåžŔäÿ■äijäéĂŠæŰĞäžűæŔŔè£řçňęïijŇåžžèőőä¡ååŔĆéŸĚåĚűäžŰäÿĂäžŻ
æŕŤåęĆ `Unix Network Programming by W. Richard Stevens (Prentice Hall, 1990)`. åIJÍWindowsäÿŁäijäéĂŠæŰĞäžűæŔŔè£řçňęèů§UnixæŸŕäÿ■äÿĂæăůçŽĎïijŇåžžèőőä¡ăçăŤ
`multiprocessing.reduction` äÿ■çŽĎæžŔäžčçăĄçIJŃçIJŃåĚűåůëä¡IJåŐ§çŘĘãĂĆ

11.12 çŘĘèğčäžŃäžűéľśåŁíçŽĎIO

éŮőécŸ

ä¡ăåžŤèŕěåůšçżŔåŘňè£Ğå§žäžŐäžŃäžűéľśåŁíæĹŰåijĆæ■ëI/OçŽĎåŇĚïijŇä¡ĘæŸŕä¡ăè£Ÿäÿ■èČ¡åőŇåŁ
æĹŰèĂĚæŸŕåęĆæđIJä¡£çŤĺåőČçŽĎèŕĪäijŽåŕźä¡ăçŽĎçÍŃåžŔäžğçŤ§äžĂäźĹå¡śåŞ■ãĂĆ

èğčåĘşæŰźæąĹ

äžŃäžűéľśåŁÍI/OæIJňèť´ĺäÿŁæİěèőšåřśæŸŕåřĘå§žæIJňI/OæŞ■ä¡IJïijĹæŕŤåęĆèŕżåŠŇåĘŹïijL'è¡ňåŇŰäÿž
ä¿ŃåęĆïijŇå¡ŞæŢřæ■őåIJĺæ§ŘäÿłsocketäÿŁèćńæŐěåŔŮåŘŐïijŇåőČäijŽè¡ňæ■ćæĹŘäÿĂäÿł
`receive` äžŃäžűïijŇçĎűåŘŐèćńä¡ăåőŽäźĽçŽĎåŻđèřČæŰźæşŢæĹŰåĞ¡æŢřæİěåď´ĎçŘĘãĂĆ
ä¡IJäÿžäÿĂäÿłåŔŕèČ¡çŽĎèţůåğŇçĆźïijŇäÿĂäÿłäžŃäžűéľśåŁíçŽĎæąĘæđűåŔŕèČ¡äijŽäžěäÿĂäÿłåőđçŐřäžĘä

```
class EventHandler:
    def fileno(self):
        'Return the associated file descriptor'
        raise NotImplemented('must implement')

    def wants_to_receive(self):
        'Return True if receiving is allowed'
        return False

    def handle_receive(self):
        'Perform the receive operation'
        pass

    def wants_to_send(self):
        'Return True if sending is requested'
        return False

    def handle_send(self):
```

```
        'Send outgoing data'
        pass
```

è£Źäÿłçśžç ŽĎåőđä¿Ńä¡IJäÿžæŔŠäžűèćńæŤ¿åĚěçśżäijijäÿŃéIćè£Źæăůç ŽĎäžŃäžűå¿łçŐŕäÿ■ïijŽ

```
import select

def event_loop(handlers):
    while True:
        wants_recv = [h for h in handlers if h.wants_to_receive()]
        wants_send = [h for h in handlers if h.wants_to_send()]
        can_recv, can_send, _ = select.select(wants_recv, wants_
↪send, [])
        for h in can_recv:
            h.handle_receive()
        for h in can_send:
            h.handle_send()
```

äžŃäžűå¿łçŐŕçŽĎåĚşéŤőéČĺåĹĘæŸý select() `` èřČçŤĺïijŇåőČäijŽäÿ■æŰ■è¡õèŕćæŰĞ
åIJÍèřČçŤÍ ``select() äźŃåĽ■ïijŇæŮűéŮť å¿łçŐŕäijŽèŕćéŮőæĽĂæIJĽçŽĎåď ĎçŘĘåŹÍæÍěåĘşåőŽ
çĎűåŘŐåőČåřĘçżŞæđIJåĹŮèąÍæŔŘä¿ŻçżŹ select() ãĂĆçĎűåŘŐ select()
è£ŤåŽđåĞĘåď ĞæŐěåŔŮæĹŰåŘŠéĂĄçŽĎåŕźèśąçżĎæĹŘçŽĎåĹŮèąÍãĂĆ
çĎűåŘŐçŻÿåžŤçŽĎ handle_receive() æĹŰ handle_send()
æŰźæşŢèćńèğęåŘŠãĂĆ

çijŰåĘŹåžŤçŤĺçÍŃåžŔçŽĎæŮűåĂŹïijŇEventHandler
çŽĎåőđä¿ŃäijŽèćńåĹŻåžžãĂĆä¿ŃåęĆïijŇäÿŃéIćæŸŕäÿď äÿłçőĂå■ŢçŽĎå§žäžŐUDPç¡ŚçżIJæIJ■åŁąçŽĎåd

```
import socket
import time

class UDPServer(EventHandler):
    def __init__(self, address):
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self.sock.bind(address)

    def fileno(self):
        return self.sock.fileno()

    def wants_to_receive(self):
        return True

class UDPTimeServer(UDPServer):
    def handle_receive(self):
        msg, addr = self.sock.recvfrom(1)
        self.sock.sendto(time.ctime().encode('ascii'), addr)

class UDPEchoServer(UDPServer):
    def handle_receive(self):
        msg, addr = self.sock.recvfrom(8192)
        self.sock.sendto(msg, addr)
```

```
if __name__ == '__main__':
    handlers = [ UDPTimeServer(('',14000)), UDPEchoServer(('',
↪15000))  ]
    event_loop(handlers)
```

æţŃèŕŢè£Źæőţäžčçă Ąïij ŃèŕŢçİĂäžŐåŔęåd'ŰäÿĂäÿłPythonèğčéĞŁåŹĺè£đæŐěåőČïijŽ

```
>>> from socket import *
>>> s = socket(AF_INET, SOCK_DGRAM)
>>> s.sendto(b'',('localhost',14000))
0
>>> s.recvfrom(128)
(b'Tue Sep 18 14:29:23 2012', ('127.0.0.1', 14000))
>>> s.sendto(b'Hello',('localhost',15000))
5
>>> s.recvfrom(128)
(b'Hello', ('127.0.0.1', 15000))
>>>
```

åőđçŐřäÿĂäÿłTCPæIJ■åŁąåŹĺäijŽæŻt'åŁăåd'■æİĆäÿĂçĆźïijŃåŻääÿžæŕŔäÿĂäÿłåőćæĹůçńŕéČ¡èęĄåĹİäÿŃéİćæŸŕäÿĂäÿłTCPåžŢç■ŤåőćæĹůçńŕä¿Ńå■ŘïijŽ

```
class TCPServer(EventHandler):
    def __init__(self, address, client_handler, handler_list):
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_
↪STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR,
↪ True)
        self.sock.bind(address)
        self.sock.listen(1)
        self.client_handler = client_handler
        self.handler_list = handler_list

    def fileno(self):
        return self.sock.fileno()

    def wants_to_receive(self):
        return True

    def handle_receive(self):
        client, addr = self.sock.accept()
        # Add the client to the event loop's handler list
        self.handler_list.append(self.client_handler(client, self.
↪handler_list))

class TCPClient(EventHandler):
    def __init__(self, sock, handler_list):
        self.sock = sock
        self.handler_list = handler_list
        self.outgoing = bytearray()
```

```
    def fileno(self):
        return self.sock.fileno()

    def close(self):
        self.sock.close()
        # Remove myself from the event loop's handler list
        self.handler_list.remove(self)

    def wants_to_send(self):
        return True if self.outgoing else False

    def handle_send(self):
        nsent = self.sock.send(self.outgoing)
        self.outgoing = self.outgoing[nsent:]

class TCPEchoClient(TCPClient):
    def wants_to_receive(self):
        return True

    def handle_receive(self):
        data = self.sock.recv(8192)
        if not data:
            self.close()
        else:
            self.outgoing.extend(data)

if __name__ == '__main__':
    handlers = []
    handlers.append(TCPServer(('',16000), TCPEchoClient, handlers))
    event_loop(handlers)
```

TCPä¿Ńå■ŘçŽĎåĚşéŤőçĆźæŸŕäżŐåď'ĎçŘĘåŹĺäÿ■åĹŮèąĺåćđåŁäåŠŇåĹăéŹđ'åőćæĹůçńŕçŽĎæŞ■ä¡IJã
åŕźæŕŔäÿĂäÿłè£đæŐěïijŇäÿĂäÿłæŰřçŽĎåď'ĎçŘĘåŹĺèćńåĹŻåžžåźűåŁäåĹřåĹŮèąĺäÿ■ãĂĆå¡Şè£đæŐěèćńåĚ
åęĆæđIJä¡äè£ŘèąŇçĺŃåžŘåźűèŕŢçİĂçŤĺTelnetæĹŰçśžäijijåůěåĚůè£đæŐěïijŇåőČäijŽårĘä¡ääŔŠéĂĄçŽĎæű

èőĺèőž

åőđéŹĚäÿŁæĽ'ĂæIJL'çŽĎäžŃäžűél'śåŁĺæąĘæđűåŐ§çŘĘèů§äÿŁéĺćçŽĎä¿Ńå■ŘçŻÿåůőæŮääĞãăĂĆåőć
ä¡ĘæŸŕåIJĺæIJĂæăÿå£ČçŽĎéČĺåĹĘïijŇéČ¡äijŽæIJL'äÿĂäÿłè¡őèŕćçŽĎå¿łçŐŕæİěæčĂæ§ěæť'żåŁĺsocketïijŇå

äžŃäžűél'śåŁĺI/OçŽĎäÿĂäÿłåŔŕèČ¡åě¡åď'ĎæŸŕåőČèČ¡åď'ĎçŘĘéİđåÿÿåď'ğçŽĎåźűåŔŚè£đæŐěïijŇèĂŇ
äź§åřśæŸŕèŕť'ïijŇselect() èřČçŤĺïijĹæĹŰåĚűäžŰç■L'æŢĹçŽĎïijL'èČ¡çŻŚåŘňåď'ğéĞŘçŽĎsocketåźűåŞ■
åIJĺå¿łçŐŕäÿ■äÿĂæňąåď'ĎçŘĘäÿĂäÿłäžŃäžűïijŇåźűäÿ■éIJĂèęĄåĚűäžŰçŽĎåźűåŔŚæIJžåĹűãĂĆ

äžŃäžűél'śåŁĺI/OçŽĎçijžçĆźæŸŕæšąæIJL'çIJ§æ■čçŽĎåŘŇæ■ěæIJžåĹűãĂĆ
åęĆæđIJäżżä¡ŢäžŃäžűåď'ĎçŘĘåŹĺæŰźæşŢéŸżåąđæĹŰæL'ğèąŇäÿĂäÿłèĂŮæŮűèőąçőŮïijŇåőČäijŽéŸżåąđa
èřČçŤĺéĆčäžŽåźűäÿ■æŸŕäžŃäžűél'śåŁĺéčŐæăijçŽĎåžŞåĞ¡æŢřäź§äijŽæIJL'éŮőécŸïijŇåŘŇæăůèęĄæŸŕæ§Ł

åŕźäžŐéŸżåąđæĹŰèĂŮæŮűèőąçőŮçŽĎéŮőécŸåŔŕäżěéĂŽè£ĞårĘäžŃäžűåŔŚéĂĄäÿłåĚűäžŰå■ŢçŃňç

äÿ■è£ĞïijŇåIJÍäžŃäžűå¿łçŐŕäÿ■åijŢåĚěåď'ŽçžŁçÍŃåŠŇåď'Žè£ŻçÍŃæŸŕæŕŤè¿ČæčŸæĽ'ŃçŽĎïijŇäÿŃéİćçŽĎä¿Ńå■ŘæijŤçď'žäžĘåęĆä¡Ţä¡£çŤÍ `concurrent.futures` æÍąåİŮæİěåőđçŐřïijŽ

```
from concurrent.futures import ThreadPoolExecutor
import os

class ThreadPoolHandler(EventHandler):
    def __init__(self, nworkers):
        if os.name == 'posix':
            self.signal_done_sock, self.done_sock = socket.
↪socketpair()
        else:
            server = socket.socket(socket.AF_INET, socket.SOCK_
↪STREAM)
            server.bind(('127.0.0.1', 0))
            server.listen(1)
            self.signal_done_sock = socket.socket(socket.AF_INET,
                                                  socket.SOCK_
↪STREAM)
            self.signal_done_sock.connect(server.getsockname())
            self.done_sock, _ = server.accept()
            server.close()

        self.pending = []
        self.pool = ThreadPoolExecutor(nworkers)

    def fileno(self):
        return self.done_sock.fileno()

    # Callback that executes when the thread is done
    def _complete(self, callback, r):

        self.pending.append((callback, r.result()))
        self.signal_done_sock.send(b'x')

    # Run a function in a thread pool
    def run(self, func, args=(), kwargs={},*,callback):
        r = self.pool.submit(func, *args, **kwargs)
        r.add_done_callback(lambda r: self._complete(callback, r))

    def wants_to_receive(self):
        return True

    # Run callback functions of completed work
    def handle_receive(self):
        # Invoke all pending callback functions
        for callback, result in self.pending:
            callback(result)
            self.done_sock.recv(1)
        self.pending = []
```

åIJĺäžčçăĄäÿ■ïijŇrun() æŰźæşŢèćńçŤĺæİěåřĘåůěä¡IJæŔŘäžď˙çżŹåŻđèřČåĞ¡æŢřæśäïijŇåď˙ĎçŘĘåőŇ
åőđéŹĚåůěä¡IJèćńæŔŘäžď˙çżŹ ThreadPoolExecutor åőđä¿ŃãĂĆ
äÿ■è£ĞäÿĂäÿłéŽ¿çĆźæŸŕå■ŔèřČèőąçőŮçżŞæđIJåŠŇäžŃäžűå¿łçŐŕïijŇäÿžäžĘèğčåĘşåőČïijŇæĹŚäżňåĹŻåż
å¡Şçž£çĺŃæśăåőŇæĹŘåůěä¡IJåŘŐïijŇåőČäijŽæĽğèąŇçśžäÿ■çŽĎ _complete()
æŰźæşŢãĂĆ è£ŹäÿłæŰźæşŢåĘ■æ§Řäÿłsocketäÿ˙ŁåĘŹåĚěå■ŮèŁĆäźŃåĽ■äijŽèőšæŇĆèţůçŽĎåŻđèřČåĞ¡æŢ
fileno() æŰźæşŢè£ŤåŻđåŘęåď˙ŮçŽĎéĆčäÿłsocketãĂĆ åŻäæ■d'ïijŇè£ŹäÿłåŮèŁĆèćńåĘŹåĚěæŮűïijŇå
çĎűåŘŐ handle_receive() æŰźæşŢèćńæ£Ăæť˙žåźűäÿžæĽ˙ĂæIJĽäźŃåĽ■æŔŘäžď˙çŽĎåůěä¡IJæĽğèąŇ
åİęçŹ¡èőšïijŇèŕť˙äžĘè£ŹäźĹåď˙Žè£đæĹŚèĞłåůśéČ¡æŹŢäžĘãĂĆ
äÿŃéĺćæŸŕäÿĂäÿłçőÅå■ŢçŽĎæIJ■åŁąåŹĺïijŇæijŤçď˙žäžĘåęĆä¡Ţä¡£çŤĺçž£çĺŃæśăæİěåőđçŐřèĂŮæŮűçŽĎè

```
# A really bad Fibonacci implementation
def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n - 1) + fib(n - 2)

class UDPFibServer(UDPServer):
    def handle_receive(self):
        msg, addr = self.sock.recvfrom(128)
        n = int(msg)
        pool.run(fib, (n,), callback=lambda r: self.respond(r,␣
↪addr))

    def respond(self, result, addr):
        self.sock.sendto(str(result).encode('ascii'), addr)

if __name__ == '__main__':
    pool = ThreadPoolHandler(16)
    handlers = [ pool, UDPFibServer(('',16000))]
    event_loop(handlers)
```

è£ŘèąŇè£ŹäÿłæIJ■åŁąåŹĺïijŇçĎűåŘŐèŕŢçİĂçŤĺåĚűåőČPythonçĺŃåžŔæİěæţŃèŕŢåőČïijŽ

```
from socket import *
sock = socket(AF_INET, SOCK_DGRAM)
for x in range(40):
    sock.sendto(str(x).encode('ascii'), ('localhost', 16000))
    resp = sock.recvfrom(8192)
    print(resp[0])
```

ä¡ăăžŤèŕěèČ¡åIJĺäÿ■åŘŇçłŮåŔčäÿ■éĞ■åď˙■çŽĎæL˙ğèąŇè£ŹäÿłçĺŃåžŔïijŇåźűäÿŤäÿ■äijŽå¡śåŞ■åĹřåĚű

åůšçżŔéŸĚèŕżåőŇäžĘè£ŹäÿĂåŕŔèŁĆïijŇéĆčäźĹä¡ăăžŤèŕěä¡£çŤĺè£ŹéĞŇçŽĎäžčçăĄåŘŮïij§äź§èőÿäÿ■
äÿ■è£ĞïijŇåęĆæđIJä¡ăçŘĘèğčäžĘå§žæIJňåŐ§çŘĘïijŇä¡ăăŕśèČ¡çŘĘèğčè£ŹäžŻæąĘæđűæL˙Ää¡£çŤĺçŽĎæäÿ
ä¡IJäÿžåŕźåŻđèřČåĞ¡æŢŕçijŰçĺŃçŽĎæŻ£äžčïijŇäžŇäžűéľ śåŁĺçijŰçăĄæIJL˙æŮűåĂŹäijŽä¡£çŤĺåĹřå■ŔçĺŃïi

11.13 åŕŚéĂĄäÿŐæŐěæŤűåď˙gåđŃæŢřçżĎ

éŮőécŸ

ä¡ăèęĄéĂŽè£Ğç¡ŚçżIJè£đæŐěåŔŚéĂĄåŠŇæŐěåŔŮè£đçż■æŢřæ■őçŽĎåď˙gåđŃæŢřçżĎïijŇåźűår¡éĞŔ

èğčåĘşæŰźæąĹ

äÿŃéÍćçŽĎåĞ¡æŢřåĽľçŤÍ `memoryviews` æÍěåŔŚéĂĄåŠŇæŐěåŔŮåď˙gæŢřçżĎïijŽ

```
# zerocopy.py

def send_from(arr, dest):
    view = memoryview(arr).cast('B')
    while len(view):
        nsent = dest.send(view)
        view = view[nsent:]

def recv_into(arr, source):
    view = memoryview(arr).cast('B')
    while len(view):
        nrecv = source.recv_into(view)
        view = view[nrecv:]
```

äÿžäžĘæţŃèŕŢçÍŃåžŔïijŇéęŰåĚĹåĹŻåžžäÿĂäÿłéĂŽè£ĞsocketèŁđæŐěçŽĎæIJ■åŁąåŹĺåŠŇåőćæĹůçńŕç

```
>>> from socket import *
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s.bind(('', 25000))
>>> s.listen(1)
>>> c,a = s.accept()
>>>
```

åIJÍåőćæĹůçńŕïijĹåŔęåď˙ŰäÿĂäÿłèğčéĞŁåŹÍäÿ■ïijL'ïijŽ

```
>>> from socket import *
>>> c = socket(AF_INET, SOCK_STREAM)
>>> c.connect(('localhost', 25000))
>>>
```

æIJňèŁĆçŽĎçŻőæăĞæŸŕä¡ăèČ¡éĂŽè£Ğè£đæŐěäijăè¿ŞäÿĂäÿłèűĚåď˙gæŢřçżĎãĂĆè£Źçğ■æČĚåĘţçŽĎ
`array` æĺąåİŮæĹŰ `numpy` æĺąåİŮæÍěåĹŻåžžæŢřçżĎïijŽ

```
# Server
>>> import numpy
>>> a = numpy.arange(0.0, 50000000.0)
>>> send_from(a, c)
>>>
```

```
# Client
>>> import numpy
>>> a = numpy.zeros(shape=50000000, dtype=float)
>>> a[0:10]
array([ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.])
>>> recv_into(a, c)
>>> a[0:10]
array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.])
>>>
```

èőĺèőž

åIJÍæŢřæ■őåŕĘéŻĘåđŃåĹĘåÿČåijŔèőąçőŮåŠŇåźşèąŇèőąçőŮçÍŃåžŔäÿ■ïijŇèĞłåůśåĘŹçÍŃåžŔæİěåőđç
äÿ■è£ĞïijŇèęĄæŸŕä¡ăçąőåőđæČşè£Źæăůå ĄŽïijŇä¡ăåŔŕèČ¡éIJĂèęĄåŕĘä¡ăçŽĎæŢřæ■ôè¡ňæ■ćæĹŘåŐ§ågŃ
ä¡ăåŔŕèČ¡è£ŸéIJĂèęĄåŕĘæŢřæ■őåĹĞåL'šæĹŘåđ'ŽäÿłåİŮïijŇåŽăäÿžåđ'ğéČĺåĹĘåŠŇç¡ŚçżIJçŻÿåĚşçŽĎåĞ¡

äÿĂçğ■æŰźæşŢæŸŕä¡£çŤĺæ§Řçğ■æIJžåĹűåžŔåĹŮåŇŰæŢřæ■ôâĂŤâĂŤåŔŕèČ¡åŕĘåĚűè¡ňæ■ćæĹŘäÿĂ
äÿ■è£ĞïijŇè£ŹæăůæIJĂçżĹäijŽåĹŻåżžæŢřæ■őçŽĎäÿĂäÿłåđ'■åĹűãĂĆ
åŕśçőŮä¡ăåŔłæŸŕéŽűçćŐçŽĎåĄŽè£ŹäžŻïijŇä¡ăçŽĎäżčçăĄæIJĂçżĹè£ŸæŸŕäijŽæIJL'åđ'ğéĞŔçŽĎåŕŔåđŃå

æIJňèŁĆéĂŽè£Ğä¡£çŤĺåĘĚå■ŸèğĘåŻ¿åśŢçđ'žäžĘäÿĂäžŻé■ŤæşŢæŞ■ä¡IJãĂĆ
æIJňèť'ĺäÿŁïijŇäÿĂäÿłåĘĚå■ŸèğĘåŻ¿åŕśæŸŕäÿĂäÿłåůšå■ŸåIJÍæŢřçżĎçŽĎèęĘçŽŰåśĆãĂĆäÿ■äżĚäżĚæŸŕé
åĘĚå■ŸèğĘåŻ¿è£ŸèČ¡äžëäÿ■åŘŇçŽĎæŰźåijŔè¡ňæ■ćæĹŘäÿ■åŘŇçśžåđŃæİěèąĺçŐřæŢřæ■őãĂĆ
è£ŹäÿłåŕśæŸŕäÿŃéİćè£Źäÿłèŕ■åŘëçŽĎçŽőçŽĎïijŽ

```
view = memoryview(arr).cast('B')
```

åőČæŐěåŔŮäÿĂäÿłæŢřçżĎ arråźűåŕĘåĚűè¡ňæ■ćäÿžäÿĂäÿłæŮăçňęåŔůå■ŮèŁĆçŽĎåĘĚå■ŸèğĘåŻ¿ãĂĆ
æŕŤåęĆ `socket.send()` æĹŰ `send.recv_into()` ãĂĆ
åIJĺåĘĚéČĺïijŇè£ŹäžŻæŰźæşŢèČ¡åđ'§çŻť'æŐěæŞ■ä¡IJè£ŹäÿłåĘĚå■ŸåŇžå§§ãĂĆä¿ŃåęĆïijŇsock.
send() çŻť'æŐěäżŐåĘĚå■Ÿäÿ■åŔŞŤ§æŢřæ■ôèĂŇäÿ■éIJĂèęĄåđ'■åĹűãĂĆ `send.
recv_into()` ä¡£çŤĺè£ŹäÿłåĘĚå■ŸåŇžå§§ä¡IJäÿžæŐěåŔŮæŞ■ä¡IJçŽĎè¿ŞåĚěçijŞåĘšåŇžãĂĆ

åL'l'äÿŃçŽĎäÿĂäÿłéŽ¿çĆźåŕśæŸŕsocketåĞ¡æŢřåŔŕèČ¡åŘłæŞ■ä¡IJéČĺåĹĘæŢřæ■őãĂĆ
éĂŽåÿÿæİěèőšïijŇæĹŚäžňå¿Ůä¡£çŤĺå¿Ĺåđ'Žäÿ■åŘŇçŽĎ `send()` åŠŇ `recv_into()`
æİěäijåè¿ŞæŢť'äÿłæŢřçżĎãĂĆ äÿ■çŤĺæŃĚå£ČïijŇæŕŔæňąæŞ■ä¡IJåŘŐïijŇèğĘåŻ¿äijŽéĂŽè£ĞåŔŚéĂĄæĹŰ
æŰřçŽĎèğĘåŻ¿åŘŇæăůäźş§æŸŕåĘĚå■ŸèęĘçŽŰåśĆãĂĆåŽăæ■đ'ïijŇè£ŸæŸŕæšąæIJL'äżżä¡ŢçŽĎåđ'■åĹűæŞ

è£ŹéĞŇæIJL'äÿłéŮőécŸåŕśæŸŕæŐěåŔŮèĂĚå£ĚéążäžŇåĚĹç§ěéĄŞæIJL'åđ'ŽåŕŚæŢřæ■őèęĄèćńåŔŚéĂ
äżěä¿£åőČèČ¡écĎåĹĘéĚ■äÿĂäÿłæŢřçżĎæĹŰèĂĚçąőä£İåőČèČ¡åŕĘæŐěåŔŮçŽĎæŢřæ■őæŤ¿åĚěäÿĂäÿłåůš
åęĆædIJæšąåŁđæşŢç§ěéĄŞçŽĎèŕÍïijŇåŔŚéĂĄèĂĚåŕśå¿ŮåĚĹåŕĘæŢřæ■ôåđ'ğåŕŔåŔŚéĂĄè£ĞæİěïijŇçĎűå

CHAPTER 14

çňňå■ĄäžŇçńäïijŽåźűåŔŚçijŰçĺŃ

åŕźäžŐåźűåŔŚçijŰçĺŃ, PythonæIJL'åď'Žçğ■éŢ£æIJ§æŤŕæŇĄçŽĎæŰźæşŢ, åŇĚæŃňåď'Žçž£çĺŃ, èřČçŤĺĺ■Řè£ŽçĺŃ, äžěåŔŁåŘĎçğ■åŘĎæăůçŽĎåĚşäžŐçŤ§æĹŘåŹĺåĞ¡æŢřçŽĎæŁĂåů è£ŹäÿĂçńăářĘäijŽçżŹåĞžåźűåŔŚçijŰçĺŃåŘĎçğ■æŰźéİćçŽĎæŁĂåůğ, åŇĚæŃňéĂŽçŤĺçŽĎåď'Žçž£çĺŃæŁĂæIJŕäžěåŔŁåźűèąŇèőąçőŮçŽĎåőđçŐřæŰźæşŢ.

åČŔçżŔélŇäÿřárŇçŽĎçĺŃåžŔåŚŸæL'Ăç§ěéĄŞçŽĎéĆčæăů, åď'ğåőűæŃĚå£Čåźűåŕ̊ŚçŽĎçĺŃåžŔæIJL'æ¡IJåIJĺçŽĎå■śéŹl'. åŻăæ■d', æIJňçńăçŽĎäÿżèęĄçŻőæăĞäźŃäÿĂæŸŕçżŹåĞžæŻt'åŁăåŔŕä£ąèţŰåŠŇæŸŞèřČèŕŢçŽĎäžčçăĄ.

Contents:

12.1 åŘŕåŁĺäÿŐåĄIJæ■ćçž£çĺŃ

éŮőécŸ

ä¡ăèęĄäÿžéIJĂèęĄåźűåŔŚæL'ğèąŇçŽĎäžčçăĄåĹŻåžž/éŤĂæŕAçž£çĺŃ

èğčåĘşæŰźæąĹ

`threading` åžŞåŔŕäżěåIJĺå■ŢçŃňçŽĎçž£çĺŃäÿ■æL'ğèąŇäżża¡ŢçŽĎåIJĺ Python äÿ■åŔŕäžěèřČçŤĺçŽĎåŕźèśąãĂĆä¡ăåŔŕäžěåĹŻåžžäÿĂäÿł `Thread` åŕźèśąåźűåřĘä¡ăèęĄæL'ğèąŇçŽĎåŕźèśąäžě target åŔĆæŢřçŽĎå¡ćåijŔæŔŘä¿ŻçżŹèřěåŕźèśąãĂĆ äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎä¿Ńå■ŘïijŽ

```
# Code to execute in an independent thread
import time
```

```
def countdown(n):
    while n > 0:
        print('T-minus', n)
        n -= 1
        time.sleep(5)

# Create and launch a thread
from threading import Thread
t = Thread(target=countdown, args=(10,))
t.start()
```

å¡Şä¡ääĹŻåžžåě¡äÿĂäÿłçž£çĺŃåŕźèšąåŘŐïijŇèŕěåŕźèšąåźűäÿ■äijŽçńŃå■şæĽğèąŇïijŇéŹd'éİđä¡ăèřČçŤĺ
start() æŰźæşŢïijĹå¡Şä¡ăèřČçŤÍ start() æŰźæşŢæŮűïijŇåőČäijŽèřČçŤĺä¡ääijäéĂŠè£ŻæİěçŽĎåĞ¡æ
POSIX çž£çĺŃæĹŰèĂĚäÿĂäÿł Windows çž£çĺŃïijĽïijŇè£ŹäžŻçž£çĺŃåřĘçŤśæŞ■ä¡IJçşżçż§æİěåĚĺæİČçőąçĺ

```
if t.is_alive():
    print('Still running')
else:
    print('Completed')
```

ä¡ääź§åŔŕäžěåřĘäÿĂäÿłçž£çĺŃåŁăåĚěåĹřå¡ŞåĽ■çž£çĺŃïijŇåźűç■Ľå¿ĚåőČçżĹæ■ćïijŽ

```
t.join()
```

PythonèğčéĞŁåŹĺåIJÍæĽĂæIJĽçž£çĺŃéČ¡çżĹæ■ćåŘŐæĽ■çżğçż■æĽğèąŇäžčçăĄåĽľä¡ŹçŽĎéČĺåĹĘã
ä¿ŃåęĆïijŽ

```
t = Thread(target=countdown, args=(10,), daemon=True)
t.start()
```

åŘŐåŔŕçž£çĺŃæŮăæşŢç■Ľå¿ĚïijŇäÿ■è£ĞïijŇè£ŹäžŻçž£çĺŃäijŽåIJĺäÿżçž£çĺŃçżĹæ■ćæŮűèĞłåŁĺéŤĂa
éŹd'äžĘåęĆäÿŁæĽĂçd'žçŽĎäÿd'äÿłæŞ■ä¡IJïijŇåźűæšąæIJĽåd'łåd'ŽåŔŕäžěåŕźçž£çĺŃåĄŽçŽĎäžŃæČĚãĂĆ

```
class CountdownTask:
    def __init__(self):
        self._running = True

    def terminate(self):
        self._running = False

    def run(self, n):
        while self._running and n > 0:
            print('T-minus', n)
            n -= 1
            time.sleep(5)

    c = CountdownTask()
    t = Thread(target=c.run, args=(10,))
    t.start()
    c.terminate() # Signal termination
    t.join()      # Wait for actual termination (if needed)
```

åęĆæđIJçž£çÍŃæĽğèąŇäÿĂäžŻåČŔI/Oè£Źæăůç ŽĎéŸżåąđæŞ■ä¡IJïijŇéĆčäźĹéĂŽè£Ğè¡őèŕćæİěçžĹæ■ä¿Ńå■ŘåęĆäÿŃïijŽ

```
class IOTask:
    def terminate(self):
        self._running = False

    def run(self, sock):
        # sock is a socket
        sock.settimeout(5)          # Set timeout period
        while self._running:
            # Perform a blocking I/O operation w/ timeout
            try:
                data = sock.recv(8192)
                break
            except socket.timeout:
                continue
            # Continued processing
            ...
        # Terminated
        return
```

èőÍèőž

çŤśäžŐåĚĺåśĂèğčéĞŁéŤĄïijĹGILïijĽçŽĎåŐ§åŻăïijŇPython çŽĎçž£çÍŃèćńéŹŘåĹűåĹřåŘŇäÿĂæŮűåĹżåŔłåĚĄèőÿäÿĂäÿłçž£çÍŃæĽğèąŇè£ŹæăůäÿĂäÿłæĽğèąŇæĺąåđŇçŽĎçž£çÍŃæŻť'éĂĆçŤĺäžŐåď'ĎçŘĘI/OåŠŇåĚűäżŮéIJĂèęĄåźűåŔŚæĽğèąŇçŽĎéŸżåąđæŞ■ä¡IJïijĹærŤåęĆ

æIJĽæŮűä¡ääijŽçIJŃåĹřäÿŃè¿źè£ŹçğéĂŽè£ĞçżğæĽ£ `Thread` çśżæİěåőđçŐřçŽĎçž£çÍŃïijŽ

```
from threading import Thread

class CountdownThread(Thread):
    def __init__(self, n):
        super().__init__()
        self.n = 0
    def run(self):
        while self.n > 0:

            print('T-minus', self.n)
            self.n -= 1
            time.sleep(5)

c = CountdownThread(5)
c.start()
```

åř¡çőąè£Źæăůäź§åŔŕäżěåůěä¡IJïijŇä¡Ęè£Źä¡£å¿Ůä¡ăçŽĎäžčçăĄä¿İèţŰäžŐ

threading åžŞïijŇæĽĂäżěä¡äçŽĎè£ŹäžŻäżčçăĄåŔłèČ¡åIJÍçž£çÍŃäÿŁäÿŃæŰĞäÿ■ä¡£çŤÍãĂĆäÿŁæŰĞæ
threading åžŞæŮäåËşçŽĎïijŇè£Źæăůåřśä¡£å¿Ůè£ŹäžŻäżčçăĄåŔŕäżěèćńçŤÍåIJÍåĚűäžŰçŽĎäÿŁäÿŃæŰČ
multiprocessing æÍąåİŮåIJÍäÿĂäÿłå■ŢçŃňçŽĎè£ŻçÍŃäÿ■æĽğèąŇä¡äçŽĎäżčçăĄïijŽ

```
import multiprocessing
c = CountdownTask(5)
p = multiprocessing.Process(target=c.run)
p.start()
```

åĘ■æňąéĞ■çŤşïijŇè£ŹæőţäżčçăĄäżĚéĂĆçŤÍäžŐ CountdownTask
çśżæŸŕäżěçŃňçńŃäžŐåőđéŹĚçŽĎåźűåŔŚæĽŃæőţïijĹåd'Žçž£çÍŃãĂĄåd'Žè£ŻçÍŃç■Ľç■Ľ'ïijĽ'åőđçŐřçŽĎæ

12.2 åĹd'æŰ■çž£çÍŃæŸŕåŘęåůšçżŔåŘŕåŁÍ

éŮőéćŸ

ä¡ăåůšçżŔåŘŕåŁÍäžĘäÿĂäÿłçž£çÍŃïijŇä¡ĘæŸŕä¡ăæČşç§ěéĄŞåőČæŸŕäÿ■æŸŕçIJ§çŽĎåůšçżŔåijĂägŃè£

èğčåĘşæŰźæąĹ

çž£çÍŃçŽĎäÿĂäÿłåĚşéŤőçĽźæĂğæŸŕæŕŔäÿłçž£çÍŃéČ¡æŸŕçŃňçńŃè£ŘèąŇäÿŤçŁűæĂĄäÿ■åŕŕécĎæţ
threading åžŞäÿ■çŽĎ Event åŕźèśąãĂĆ Event åŕźèśąåŇĚåŘńäÿĂäÿłåŔŕçŤśçž£çÍŃèő¿ç¡őçŽĎä£ąåŔůæ
åŕźèśąäÿ■çŽĎä£ąåŔůæĞå£Ůèćńèő¿ç¡őäÿžåĄĞãĂĆåęĆæđIJæIJL'çž£çÍŃç■Ľå¿ĚäÿĂäÿł
event åŕźèśąïijŇèĂŇè£Źäÿł event åŕźèśąçŽĎæăĞå£ŮäÿžåĄĞïijŇéĆčäźĹè£Źäÿłçž£çÍŃåřĘäijŽèćńäÿĂçŻť'éŸ
event åŕźèśąçŽĎä£ąåŔůæăĞå£Ůèő¿ç¡őäÿžçIJ§ïijŇåőČåřĘåŤd'éĘŠæĽĂæIJL'ç■Ľå¿Ěè£Źäÿł
event åŕźèśąçŽĎçž£çÍŃãĂĆåęĆæđIJäÿĂäÿłçž£çÍŃç■Ľå¿ĚäÿĂäÿłåůšçżŔèćńèő¿ç¡őäÿžçIJ§çŽĎ
event åŕźèśąïijŇéĆčäźĹåőČåřĘå£¡çŢěè£ŹäÿłäžŃäżűïijŇçžğçž■æĽğèąŇãĂĆ
äÿŃè¿źçŽĎäžčçăĄåśŢçd'žäžĘåęĆä¡Ţä¡£çŤÍ Event æİěå■ŔěŘČçž£çÍŃçŽĎåŘŕåŁÍïijŽ

```
from threading import Thread, Event
import time

# Code to execute in an independent thread
def countdown(n, started_evt):
    print('countdown starting')
    started_evt.set()
    while n > 0:
        print('T-minus', n)
        n -= 1
        time.sleep(5)

# Create the event object that will be used to signal startup
started_evt = Event()

# Launch the thread and pass the startup event
print('Launching countdown')
```

```
t = Thread(target=countdown, args=(10,started_evt))
t.start()

# Wait for the thread to start
started_evt.wait()
print('countdown is running')
```

å¡Şä¡äæĽğèąŇè£ŹæőţäžčçăĄïijŇâĂIJcountdown is runninggâĂİ æĂżæŸŕæŸ¿çd'žåIJÍ âĂIJcountdown startinggâĂİ äźŃåŘŐæŸ¿çd'žãĂĆè£ŹæŸŕçŤśäžŐä¡£çŤÍ event æİěå■ŔèřČçž£çÍŃïijŇä¡£å¿Ůäÿżçž£çÍŃèęĄç■Ľ'åĹř `countdown()` åĞ¡æŢřè¿ŞåĞžåŘŕåŁĺä£qæĄŕåŘŐïijŇæĽ'■èČ¡çżğçż■æĽ'ğèąŇãĂĆ

èőĺèőž

event åŕźèśąæIJĂåě¡å■Ţæňąä¡£çŤĺïijŇåřśæŸŕèŕť'ïijŇä¡ăåĹŻåžžäÿĂäÿł event åŕźèśąïijŇèőľæ§Řäÿł£çÍŃç■Ľ'å¿Ěè£ŹäÿłåŕźèśąïijŇäÿĂæŮęè£Źäÿłåŕźèśąèćńèő¿ç¡őäÿžçIJ§ïijŇä¡ăåřśåžŤèŕè `clear()` æŰźæşŢæİěéĞ■ç¡ő event åŕźèśąïijŇä¡ĘæŸŕå¿ĹéŽ¿çąőä£İåőĽåĚĺåIJřæÿĚçŘĘ event åŕźèśąåźűåŕźåőČéĞ■æŰřètŃåĂijãĂĆå¿ĹåŘŕèČ¡äijŽåŔŚçŤ§éŤŹè£ĞäžŃäžűãĂĄæ■żéŤĄæĹŰèĂĚåĚűä event åŕźèśąçŽĎäžčçăĄäijŽåIJĺçž£çÍŃåĘ■æňąç■Ľ'å¿Ěè£ŹäÿłåŕźèśąäźŃåĽ'■æĽ'ğèąŇïijĽãĂĆåęĆæđIJäÿĂäÿłçž£çÍŃéIJĂèęĄäÿ■åĄIJåIJřéĞ■åd'■ä¡£çŤÍ event åŕźèśąïijŇä¡ăæIJĂåě¡ä¡£çŤÍ `Condition` åŕźèśąæİěäžčæŽ£ãĂĆäÿŃéİççŽĎäžčçăĄä¡£çŤÍ `Condition` åŕźèśąåődçŐřäžĘäÿĂäÿłåŚĺæIJ§åőŽæŮűåŹĺïijŇærŔå¡ŞåőŽæŮűåŹĺèűĚæŮűçŽĎæŮűåĂŹïijŇåĹ

```
import threading
import time

class PeriodicTimer:
    def __init__(self, interval):
        self._interval = interval
        self._flag = 0
        self._cv = threading.Condition()

    def start(self):
        t = threading.Thread(target=self.run)
        t.daemon = True

        t.start()

    def run(self):
        '''
        Run the timer and notify waiting threads after each interval
        '''
        while True:
            time.sleep(self._interval)
            with self._cv:
                self._flag ^= 1
                self._cv.notify_all()
```

```
    def wait_for_tick(self):
        '''
        Wait for the next tick of the timer
        '''
        with self._cv:
            last_flag = self._flag
            while last_flag == self._flag:
                self._cv.wait()

# Example use of the timer
ptimer = PeriodicTimer(5)
ptimer.start()

# Two threads that synchronize on the timer
def countdown(nticks):
    while nticks > 0:
        ptimer.wait_for_tick()
        print('T-minus', nticks)
        nticks -= 1

def countup(last):
    n = 0
    while n < last:
        ptimer.wait_for_tick()
        print('Counting', n)
        n += 1

threading.Thread(target=countdown, args=(10,)).start()
threading.Thread(target=countup, args=(5,)).start()
```

eventåŕźèśąçŽĎäÿĂäÿłéĞ■èęĄçĽźçĆźæŸŕå¡ŞåőČèćńèő¿ç¡őäÿžçIJ§æŮűäijŽåŤd'éĘŠæĽĂæIJL'ç■L'å¿Ě
Condition åŕźèśąæİěæŻ£äžčãĂĆèĂČèŹŚäÿĂäÿŃè£Źæőţä¡£çŤĺä£ąåŔůéĞŔåőđçŐřçŽĎäžčçăĄïijŽ

```
# Worker thread
def worker(n, sema):
    # Wait to be signaled
    sema.acquire()

    # Do some work
    print('Working', n)

# Create some threads
sema = threading.Semaphore(0)
nworkers = 10
for n in range(nworkers):
    t = threading.Thread(target=worker, args=(n, sema,))
    t.start()
```

è£ŘèąŇäÿŁè¿źçŽĎäžčçăĄåřĘäijŽåŘŕåŁĺäÿĂäÿłçž£çĺŃæśäïijŇä¡ĘæŸŕåźűæšąæIJL'äżĂäźĹäžŃæČĚåŔŚ

```
>>> sema.release()
Working 0
>>> sema.release()
Working 1
>>>
```

çijŰåĘŹæűĽåŔŁåĹřåď˙ğéĞŔçŽĎçž£çÍŃéŮt˙åŘŇæ■ěéŮőécŸçŽĎäžčçăĄäijŽèőľä¡ăçŮŻäÿ■æňšçŤ§ãĂĆ

12.3 çž£çÍŃéŮt˙éĂŽä£ą

éŮőécŸ

ä¡ăçŽĎçÍŃåžŔäÿ■æIJL˙åď˙ŽäÿłçžçÍŃïijŇä¡ăéIJĂèęĄåIJÍè£ŹäžŻçž£çÍŃäźŃéŮt˙åőL˙åĚÍåIJřäžď˙æ■ćä£ąa

èğčåĘşæŰźæąĹ

äžŐäÿĂäÿłçž£çÍŃåŘŚåŔęäÿĂäÿłçž£çÍŃåŘŚéĂĄæŢřæ■őæIJĂåőL˙åĚÍçŽĎæŰźåijŔåŔŕèČ¡åřśæŸŕä¡£çŤÍ
queue åžŞäÿ■çŽĎéŸ§åĹŮäžĘãĂĆåĹŻåžžäÿĂäÿłèćńåď˙Žäÿłçž£çÍŃåĚśäžńçŽĎ
Queue åŕźèśąïijŇè£ŹäžŻçž£çÍŃéĂŽè£Ğä¡£çŤÍ put() åŠŇ get()
æŞ■ä¡IJæİěåŘŚéŸ§åĹŮäÿ■æůżåŁăæĹŰèĂĚåĹăéŹď˙åĚČçt˙ăãĂĆ ä¿ŃåęĆïijŽ

```
from queue import Queue
from threading import Thread

# A thread that produces data
def producer(out_q):
    while True:
        # Produce some data
        ...
        out_q.put(data)

# A thread that consumes data
def consumer(in_q):
    while True:
# Get some data
        data = in_q.get()
        # Process the data
        ...

# Create the shared queue and launch both threads
q = Queue()
t1 = Thread(target=consumer, args=(q,))
t2 = Thread(target=producer, args=(q,))
t1.start()
t2.start()
```

Queue åŕźèśąåůšçżŔåŃĚåŘńäžĘå£ĚèęAçŽĎéŤĄïijŇæĽ Ăäżěä¡ăåŘŕäžěéĂŽè£ĞåőČåIJÍåď Žäÿłçž£çÍŃé
å¡Şä¡£çŤĺéŸ§åĹŮæŮűïijŇå■ŔèřČçŤ§äžğèĂĚåŠŇæűĹèť źèĂĚçŽĎåĚşéŮ■éŮőéćŸåŔŕèČ¡äijŽæIJĽäÿĂäžŽé

```
from queue import Queue
from threading import Thread

# Object that signals shutdown
_sentinel = object()

# A thread that produces data
def producer(out_q):
    while running:
        # Produce some data
        ...
        out_q.put(data)

    # Put the sentinel on the queue to indicate completion
    out_q.put(_sentinel)

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()

        # Check for termination
        if data is _sentinel:
            in_q.put(_sentinel)
            break

        # Process the data
        ...
```

æIJňä¿Ńäÿ■æIJĽäÿĂäÿłçĽźæőŁçŽĎåIJřæŰźïijŽæűĹèť źèĂĚåIJÍèŕżåĹřè£ŹäÿłçĽźæőŁåĂijäźŃåŘŐçńŃ
åř¡çőąéŸ§åĹŮæŸŕæIJĂåÿÿèğĄçŽĎçž£çÍŃéŮť éĂŽä£ąæIJžåĹűïijŇä¡ĘæŸŕäż■çĎűåŔŕäžěèĞłåůśéĂŽè£ĞåĹŽ
Condition åŔŸéĞŔæİĕåŃĚèčĚä¡ăçŽĎæŢřæ■őçżŞæđĎãĂĆäÿŃè¿źè£Źäÿłä¿Ńå■ŘæijŤçď žäžĘåęĆä¡ŢåĹŽ

```
import heapq
import threading

class PriorityQueue:
    def __init__(self):
        self._queue = []
        self._count = 0
        self._cv = threading.Condition()
    def put(self, item, priority):
        with self._cv:
            heapq.heappush(self._queue, (-priority, self._count,␣
↪item))
            self._count += 1
            self._cv.notify()
```

```
    def get(self):
        with self._cv:
            while len(self._queue) == 0:
                self._cv.wait()
            return heapq.heappop(self._queue)[-1]
```

ä¡£çŤĺéŸ§åĹŮæİĕè£ŻèąŇçž£çĺŃéŮťéĂŽä£ąæŸŕäÿĂäÿłå■ŢåŘŚãĂĄäÿ■çąőåőŽçŽĎè£ĞçĺŃãĂĆéĂŽåÿ
task_done() åŠŇ join() ïijŽ

```
from queue import Queue
from threading import Thread

# A thread that produces data
def producer(out_q):
    while running:
        # Produce some data
        ...
        out_q.put(data)

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()

        # Process the data
        ...
        # Indicate completion
        in_q.task_done()

# Create the shared queue and launch both threads
q = Queue()
t1 = Thread(target=consumer, args=(q,))
t2 = Thread(target=producer, args=(q,))
t1.start()
t2.start()

# Wait for all produced items to be consumed
q.join()
```

åęĆæđIJäÿĂäÿłçž£çĺŃéIJĂèęĄåIJĺäÿĂäÿłâĂIJæűĹèťźèĂĚâĂİçž£çĺŃåďĎçŘĘåőŇçĽźåőŽçŽĎæŢřæ■ő
Event æŤ¿åĽřäÿĂèţůå¡£çŤĺïijŇè£ŹæăůâĂIJçŤ§äžğèĂĚâĂİåřśåŔŕäžěéĂŽè£Ğè£ŹäÿłEventåŕźèśąæİěçŽŚæ

```
from queue import Queue
from threading import Thread, Event

# A thread that produces data
def producer(out_q):
    while running:
```

```
        # Produce some data
        ...
        # Make an (data, event) pair and hand it to the consumer
        evt = Event()
        out_q.put((data, evt))
        ...
        # Wait for the consumer to process the item
        evt.wait()

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data, evt = in_q.get()
        # Process the data
        ...
        # Indicate completion
        evt.set()
```

èőÍèőž

å§žäžŐçőĂå■ŢéŸ§åĹŮçijŰåĘŹåď Žçž£çÍŃçÍŃåžŔåIJÍåď ŽæŢřæČĚåĘţäÿŃæŸŕäÿĂäÿłæŕŤè¿ČæŸŐæŹż
ä¡£çŤÍçž£çÍŃéŸ§åĹŮæIJL'äÿĂäÿłèęĄæşÍæĎŔçŽĎéŮőécŸæŸŕïijŇåŘŚéŸ§åĹŮäÿ■æůžåŁăæŢřæ■őéąźæŮűå

```
from queue import Queue
from threading import Thread
import copy

# A thread that produces data
def producer(out_q):
    while True:
        # Produce some data
        ...
        out_q.put(copy.deepcopy(data))

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()
        # Process the data
        ...
```

Queue åŕźèśąæŔŘä¿ŻäÿĂäžŽåIJÍå¡ŞåL'■äÿŁäÿŃæŰĞå¿ÍæIJL'çŤÍçŽĎéŹĎåŁăçL'źæĂğãĂĆæŕŤåęĆåIJ
Queue åŕźèśąæŮűæŔŘä¿ŻåŘŕéĂL'çŽĎ size åŘĆæŢřæİěéŹŘåĹűåŘŕäžěæůžåŁäåĹřéŸ§åĹŮäÿ■çŽĎåĚĆçť'ă
âĂIJæűĹèť'źâĂİçŽĎéĂ§åžęå£ńïijŇéĆčäźĹä¡£çŤĺåŻžåőŽåď ğåřŔçŽĎéŸ§åĹŮåřśåŘŕäžěåIJÍéŸ§åĹŮåůšæžąçŽ
get() åŠŇ put() æŰźæşŢéČ¡æŤřæŇĄéİđéŸżåąđæŰźåijŔåŠŇèő¿åőŽèűĚæŮűïijŇä¿ŃåęĆïijŽ

```
import queue
q = queue.Queue()

try:
    data = q.get(block=False)
except queue.Empty:
    ...

try:
    q.put(item, block=False)
except queue.Full:
    ...

try:
    data = q.get(timeout=5.0)
except queue.Empty:
    ...
```

è£ŹäžŻæŞ■ä¡IJéČ¡åŔŕäżěçŤĺæİěéĄ£åĚ■å¡ŞæĽğèąŇæ§ŘäžŻçĽźåőŽéŸ§åĹŮæŞ■ä¡IJæŮűåŔŚçŢ§æŮăé
put() æŰźæşŢåŠŇäÿĂäÿłåŻžåőŽåď ğåřŔçŽĎéŸ§åĹŮäÿĂèţůä¡£çŢĺïijŇè£Źæăůå¡ŞéŸ§åĹŮåůšæżąæŮűåřšá

```
def producer(q):
    ...
    try:
        q.put(item, block=False)
    except queue.Full:
        log.warning('queued item %r discarded!', item)
```

åęĆæđIJä¡ăèŕŢåŻ¿èőľ æűĹèť źèĂĚçž£çĺŃåIJĺæĽ ğèąŇåČŔ q.get() è£ŹæăůçŽĎæŞ■ä¡IJæŮűïijŇèűĚæŮűèĞłåŁĺçżĹæ■ćäžěä¿£æčĂæ§ěçžĹæ■ćæăĞå£ŮïijŇä¡ăåžŢèŕěä¡£çŢĺ q.get() çŽĎåŔŕéĂĽåŔĆæŢř timeout ïijŇåęĆäÿŃïijŽ

```
_running = True

def consumer(q):
    while _running:
        try:
            item = q.get(timeout=5.0)
            # Process item
            ...
        except queue.Empty:
            pass
```

æIJĂåŘŐïijŇæIJĽ q.qsize() ïijŇ q.full() ïijŇ q.empty() ç■Ľåőđç ŢĺæŰźæşŢåŔŕäżèèŐůåŔŰäÿĂäÿłéŸ§åĹŮçŽĎå¡ŞåĽ■åď ğåřŔåŠŇçŁűæĂĄãĂĆä¡Ęèę Ąæşĺæ ĎŔïijŇè
empty() åĹď æŰ■åĞžè£ŹäÿłéŸ§åĹŮäÿžçľžïijŇä¡ĘåŘŇæŮűåŔęåď ŰäÿĂäÿłçž£çĺŃåŔŕèČ¡åůšçżŔåŘŚè£Źä

12.4 çžŹåĚşéŤőéČĺåĹĘåŁăéŢĄ

éŮőécŸ

ä¡ăéIJĂèęĄåŕźåď˙ŽçžŁçĺŃçĺŃåžŔäÿ■çŽĎäÿť˙çŢŇåŇžåŁăéŢĄäżěéĄŁåĚ■çńđäžĽ˙æİąäżűãĂĆ

èğčåĘşæŰźæąĹ

èęĄåIJĺåď˙Žçž£çĺŃçĺŃåžŔäÿ■åőĽ˙åĚĺä¡£çŢĺåŔŕåŔŸåŕźèśąïijŇä¡ăéIJĂèęĄä¡£çŢĺ threading åžŞäÿ■çŽĎ `Lock` åŕźèśąïijŇåŕśåČŔäÿŃè¿źè£ŹäÿŁä¿Ńå■Řè£ŹæăůïijŽ

```
import threading

class SharedCounter:
    '''
    A counter object that can be shared by multiple threads.
    '''
    def __init__(self, initial_value = 0):
        self._value = initial_value
        self._value_lock = threading.Lock()

    def incr(self,delta=1):
        '''
        Increment the counter with locking
        '''
        with self._value_lock:
             self._value += delta

    def decr(self,delta=1):
        '''
        Decrement the counter with locking
        '''
        with self._value_lock:
             self._value -= delta
```

`Lock` åŕźèśąåŠŇ `with` èŕ■åŔěåİŮäÿĂètůä¡£çŢĺåŔŕäžëä£ÌèŕĄäžŠæŰěæĽ˘ğèąŇïijŇåŕśæŸŕæŕŔæňąåŔŁæI
with èŕ■åŔěåŇĚåŘńçŽĎäžčçăĄåİŮãĂĆwith èŕ■åŔëäijŽåIJĺè£ŹäÿŁäžčçăĄåİŮæĽ˘ğèąŇåĽ˘■èĞŁåŁĺèŐůåŔŰéŢ

èőĺèőž

çž£çĺŃèřČåžęæIJňèť˙ĺäÿŁæŸŕäÿ■çąőåőŽçŽĎïijŇåŽăæ■d˘ïijŇåIJĺåď˙Žçž£çĺŃçĺŃåžŔäÿ■éŢŹèŕŕåIJŕä¡£çŢ
åIJĺäÿĂäžŽâĂIJèĂĄçŽĎâĂİ Python äžčçăĄäÿ■ïijŇæŸ¿åijŔèŐůåŔŰåŠŇéĞŁæŢ¿éŢĄæŸŕå¿Ĺåÿÿèğ ĄçŽĎãĂ

```
import threading

class SharedCounter:
```

```
    '''
    A counter object that can be shared by multiple threads.
    '''
    def __init__(self, initial_value = 0):
        self._value = initial_value
        self._value_lock = threading.Lock()

    def incr(self,delta=1):
        '''
        Increment the counter with locking
        '''
        self._value_lock.acquire()
        self._value += delta
        self._value_lock.release()

    def decr(self,delta=1):
        '''
        Decrement the counter with locking
        '''
        self._value_lock.acquire()
        self._value -= delta
        self._value_lock.release()
```

çŽÿæŕŤäžŐè£Źçğ■æŸ¿åijŔèřČçŤĺçŽĎæŰźæşŢïijŇwith èŕ■åŔěæŻť åŁääijŸéŻĚïijŇäź§æŻť äÿ■åőźæŸŞ release() æŰźæşŢæĹŰèĂĚçĺŃåžŔåIJĺèŐůå¿ŮéŤĄäźŃåŘŐäžğçŤ§åijĆåÿÿè£Źäÿď çğ■æČĚåĘţïijĹä¡£çŤĺ with èŕ■åŔěåŔŕäžěä£ÌèŕĄåIJĺè£Źäÿď çğ■æČĚåĘţäÿŃäż■èČ¡æ■čçąőéĞŁæŤ¿éŤĄïijĽãĂĆ äÿžäžĘéĄ£åĚ■åĞžçŐřæ■żéŤĄçŽĎæČĚåĘţïijŇä¡£çŤĺéŤĄæIJžåĹűçŽĎçĺŃåžŔåžŤèŕěèő¿åőŽäÿžæŕŔäÿłçž£çĺ åIJĺ `threading` åžŞäÿ■è£ŸæŔŘä¿ŻäžĘåĚűäžŰçŽĎåŘŇæ■ěåŐ§èŕ■ïijŇæŕŤåęĆ `RLoct` åŠŇ `Semaphore` åŕźèśąãĂĆä¡ĘæŸŕæăźæ■őäžěå¿ĂçżŔéłŇïijŇè£ŹäžŻåŐ§èŕ■æŸŕçŤĺäžŐäÿĂäžŻçĽźæőĹçŽ `RLock` ïijĹåŔŕéĞ■åĚěéŤĄïijĽåŔŕäžěèćńåŘŇäÿĂäÿłçž£çĺŃåď ŽæňąèŐůåŔŰïijŇäÿżèęĄçŤĺæİěåőđçŐřå§žäžĞ SharedCounter çśżïijŽ

```
import threading

class SharedCounter:
    '''
    A counter object that can be shared by multiple threads.
    '''
    _lock = threading.RLock()
    def __init__(self, initial_value = 0):
        self._value = initial_value

    def incr(self,delta=1):
        '''
        Increment the counter with locking
        '''
        with SharedCounter._lock:
            self._value += delta

    def decr(self,delta=1):
```

```
        '''
        Decrement the counter with locking
        '''
        with SharedCounter._lock:
            self.incr(-delta)
```

åIJĺäÿŁè¿źè£ŹäÿŁä¿Ńå■Řäÿ■ïijŇæšąæIJL'åŕźæŕŔäÿĂäÿłåőđä¿Ńäÿ■çŽĎåŔŕåŔŸåŕźèšąåŁăéŤĄïijŇåŔŰè
decr æŰźæşŢãĂĆ è£ŹçğåőđçŐřæŰźåijŔçŽĎäÿĂäÿłçĽźçĆźæŸřïijŇæŮăèőžè£ŹäÿłçśżæIJL'åđŽåřŚäÿłåőđä¿
ä£ąåŔůéĞŔåŕźèšąæŸřäÿĂäÿłåżžçńŇåIJĺåĚśäžńèőąæŢřåŹĺå§žçąĂäÿŁçŽĎåŘŇæ■ěåŐ§èŕ■ãĂĆåęĆæđIJèőąæŢ
èŕ■åŘěåřĘèőąæŢřåŹĺåĞŔ1ïijŇçž£çĺŃèćńåĚĄèőÿæL'ğèąŇãĂĆwith
èŕ■åŘěæL'ğèąŇçżŞæİ§åŘŐïijŇèőąæŢřåŹĺåŁäïijŚãĂĆåęĆæđIJèőąæŢřåŹĺäÿž0ïijŇçž£çĺŃåřĘèćńéŸżåąđïijŇçŽ

```
from threading import Semaphore
import urllib.request

# At most, five threads allowed to run at once
_fetch_url_sema = Semaphore(5)

def fetch_url(url):
    with _fetch_url_sema:
        return urllib.request.urlopen(url)
```

åęĆæđIJä¡ăåŕźçž£çĺŃåŘŇæ■ěåŐ§èŕ■çŽĎåžŢåśĆçŘĘèőžåŠŇåőđçŐřæĎ§åĚt'èűčïijŇåŔŕäžěåŔĆèĂČæŞ

12.5 éŸšæ■ćæ■żéŤĄçŽĎåŁăéŤĄæIJžåĹű

éŮőécŸ

ä¡ăæ■čåIJĺåĘŹäÿĂäÿłåd'Žçž£çĺŃçĺŃåžŔïijŇåĚűäÿ■çž£çĺŃéIJĂèęĄäÿĂæňąèŐůåŔŰåd'ŽäÿłéŤĄïijŇæ■d

èğčåĘşæŰźæąĹ

åIJĺåd'Žçž£çĺŃçĺŃåžŔäÿ■ïijŇæ■żéŤĄéŮőécŸå¿Ĺåd'ğäÿĂéČĺåĹĘæŸŕçŤšäžŐçž£çĺŃåŘŇæŮűèŐůåŔŰå
æŮűåĂŹåŔŠçŤ§éŸżåąđïijŇéĆčäźĹè£ŹäÿłçžçĺŃåřšåŔŕèČ¡éŸżåąđåĚűäżŰçž£çĺŃçŽĎæL'ğèąŇïijŇäżŐèĂŇå
èğčåĘşæ■żéŤĄéŮőécŸçŽĎäÿĂçğ■æŰźæąĹæŸŕäÿžçĺŃåžŔäÿ■çŽĎærŔäÿĂäÿłéŤĄåĹĘéĚ■äÿĂäÿłåŤŕäÿĂçŽ
æŸŕéİđåÿÿåőźæŸŞåőđçŐřçŽĎïijŇçd'žä¿ŇåęĆäÿŇïijŽ

```
import threading
from contextlib import contextmanager

# Thread-local state to stored information on locks already acquired
_local = threading.local()

@contextmanager
def acquire(*locks):
    # Sort locks by object identifier
```

```
    locks = sorted(locks, key=lambda x: id(x))

    # Make sure lock order of previously acquired locks is not␣
↪violated
    acquired = getattr(_local,'acquired',[])
    if acquired and max(id(lock) for lock in acquired) >=␣
↪id(locks[0]):
        raise RuntimeError('Lock Order Violation')

    # Acquire all of the locks
    acquired.extend(locks)
    _local.acquired = acquired

    try:
        for lock in locks:
            lock.acquire()
        yield
    finally:
        # Release locks in reverse order of acquisition
        for lock in reversed(locks):
            lock.release()
        del acquired[-len(locks):]
```

åęĆä¡Ţä¡£çŤĺè£Źäÿłäÿ ŁäÿŃæŰĞçőąçŘĘåŹĺåŚćïij§ä¡ăåŕŕäžěæŇĽçĚğæ■čåÿÿéĂŤå¿ĎåĹŻåžžäÿĂäÿłéŤ
acquire() åĞ¡æŢřæİěçŤşèřůéŤĄïijŇ çd'žä¿ŃåęĆäÿŃïijŽ

```
import threading
x_lock = threading.Lock()
y_lock = threading.Lock()

def thread_1():
    while True:
        with acquire(x_lock, y_lock):
            print('Thread-1')

def thread_2():
    while True:
        with acquire(y_lock, x_lock):
            print('Thread-2')

t1 = threading.Thread(target=thread_1)
t1.daemon = True
t1.start()

t2 = threading.Thread(target=thread_2)
t2.daemon = True
t2.start()
```

åęĆæđIJä¡ăæL'ğèąŇè£Źæőţäžčçă Ąïij Ňä¡ăäijŽåŔŚçŐřåőČå■şä¡£åIJĺäÿ■åŘŇçŽĎåĞ¡æŢřäÿ■äžěäÿ■åŘŇç
åĚűåĚşéŤőåIJĺäžŐïijŇåIJĺçňňäÿĂæőţäžčçăĄäÿ■ïijŇæĹŚäžňářžè£ŹäžŻéŤĄè£ŻèąŇäžĘæŐŠåžŔãĂĆéĂŽè£Ğ

åęĆæđIJæIJĽåď'Žäÿł acquire() æŞ■ä¡IJèćńåţŇåěŮèřČçŤĺïijŇåŘŕäžěéĂŽè£ĞçžçĺŃæIJňåIJřå■ŸåĆĺïijĹT
åĄĞèő¿ä¡ăçŽĎäžčçăĄæŸŕè£ŹæăůåĘŹçŽĎïijŽ

```
import threading
x_lock = threading.Lock()
y_lock = threading.Lock()

def thread_1():

    while True:
        with acquire(x_lock):
            with acquire(y_lock):
                print('Thread-1')

def thread_2():
    while True:
        with acquire(y_lock):
            with acquire(x_lock):
                print('Thread-2')

t1 = threading.Thread(target=thread_1)
t1.daemon = True
t1.start()

t2 = threading.Thread(target=thread_2)
t2.daemon = True
t2.start()
```

åęĆæđIJä¡ăè£ŘèąŇè£ŹäÿłçĽĹæIJňçŽĎäžčçăĄïijŇå£ĚåőŽäijŽæIJĽäÿĂäÿłçž£çĺŃåŘŠçŤ§åť'ľæžČïijŇåi

```
Exception in thread Thread-1:
Traceback (most recent call last):
  File "/usr/local/lib/python3.3/threading.py", line 639, in _
↪bootstrap_inner
    self.run()
  File "/usr/local/lib/python3.3/threading.py", line 596, in run
    self._target(*self._args, **self._kwargs)
  File "deadlock.py", line 49, in thread_1
    with acquire(y_lock):
  File "/usr/local/lib/python3.3/contextlib.py", line 48, in __
↪enter__
    return next(self.gen)
  File "deadlock.py", line 15, in acquire
    raise RuntimeError("Lock Order Violation")
RuntimeError: Lock Order Violation
>>>
```

åŘŠçŤ§åť'ľæžČçŽĎåŐ§åŽäåIJĺäžŐïijŇærŔäÿłçž£çĺŃéČ¡èőřå¡ŢçİĂèĞłåůšåůšçżŔèŐůåŘŰåĹřçŽĎéŤĄã
acquire() åĞ¡æŢřäijŽæčĂæ§ěäźŃåĽ■åůšçżŔèŐůåŘŰçŽĎéŤĄåĹŮèąĺïijŇ
çŤšäžŐéŤĄæŸŕæŇĽçĚğå■ĞåžŔæŐŠåĹŮèŐůåŘŰçŽĎïijŇæĽĂäžěåĞ¡æŢřäijŽèőď'äÿžäźŃåĽ■åůšèŐůåŘŰç

èőĺèőž

æ■žéŤĄæŸŕæŕŔäÿĂäÿłåď'ŽçŽ£çĺŃçĺŃåžŔéČ¡äijŽéİcäÿť'çŽĎäÿĂäÿłéŮőécŸïijĹåŕśåČŔåőČæŸŕæŕŔäÿĂ
çž£çĺŃåŔłèČ¡åŘŇæŮűä£ÍæŇĄäÿĂäÿłéŤĄïijŃè£ŹæăůçĺŃåžŔåŕśäÿ■äijŽècńæ■žéŤĄéŮőécŸæĽ'ĂåŽřæĽřãĂ

æ■žéŤĄçŽĎæčĂæţŃäÿŐæĄcåď'■æŸŕäÿĂäÿłåĞăăźŐæšąæIJL'äijŸéŽĚçŽĎè§čåĘşæŰźæąĹçŽĎæĽ'ľåśŢ
è£ŘèąŇçŽĎæŮűåĂŹäijŽæŕŔéŽŤäÿĂæőţæŮűéŮť'éĞ■ç¡õèőąæŢřåŹĺïijŃåIJÍæšąæIJL'åŘŚçŢ§æ■žéŤĄçŽĎæČ
èűĚæŮűïijŃè£ŹæŮűçĺŃåžŔäijŽéĂŽè£ĞéĞ■åŘŕèĞłèžńæĄcåď'■åĹřæ■čåÿÿçŁűæĂĄãĂĆ

éĄ£åĚ■æ■žéŤĄæŸŕåŖęåď'ŰäÿĂçğ■èğčåĘşæ■žéŤĄéŮőécŸçŽĎæŰźåijŔïijŃåIJÍè£ŻçĺŃèŐůåŔŰéŤĄçŽ
æ■žéŤĄçŁűæĂĄãĂĆèŕĄæŸŐåŕśçŢŹçżŹèŕźèĂĚä¡IJäÿžçżČäźăäžĘãĂĆéĄ£åĚ■æ■žéŤĄçŽĎäÿżèęĄæĂİæČşa
æ■žéŤĄçŽĎäÿĂäÿłå£ĚèęĄæİąäžűïijŃäżŐèĂŇéĄ£åĚ■çĺŃåžŔè£ŻåĚěæ■žéŤĄçŁűæĂĄãĂĆ

äÿŃéİcäżěäÿĂäÿłåĚşäžŐçž£çĺŃæ■žéŤĄçŽĎçżŔåĚÿéŮőécŸïijŽâĂIJåŞšå■ęåőűåřśéď'ŘéŮőécŸâĂİïijŃä
éİcåĽ■æIJL'äÿĂçćŮéě■åŠŃäÿĂåŘłç■ůå■ŘãĂĆåIJÍè£ŹéĞŇærŔäÿłåŞšå■ęåőűåŔŕäžěçIJŃåĄŽæŸŕäÿĂäÿłçŃ
æĂİèĂČãĂĄåŘČéě■äÿĽçğ■çŁűæĂĄäÿ■çŽĎäÿĂäÿłãĂĆéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijŃærŔäÿłåŞšå■ęåőűåŘČ
éĆčäźĹäżŰäžňäžŤäÿłéČ¡åŘłèČ¡æŃ£çİĂäÿĂåŘłç■ůå■ŘåİŘåIJÍéĆčåĎ£ïijŃçŻť'åĹřéě£æ■žãĂĆæ■ď'æŮűäżŰä
äÿŃéİcæŸŕäÿĂäÿłçőĂå■ŢçŽĎä¡£çŤÍæ■žéŤĄéĄ£åĚ■æIJžåĹűèğčåĘşâĂIJåŞšå■ęåőűåřśéď'ŘéŮőécŸâĂİçŽĎå

```
import threading

# The philosopher thread
def philosopher(left, right):
    while True:
        with acquire(left,right):
             print(threading.currentThread(), 'eating')

# The chopsticks (represented by locks)
NSTICKS = 5
chopsticks = [threading.Lock() for n in range(NSTICKS)]

# Create all of the philosophers
for n in range(NSTICKS):
    t = threading.Thread(target=philosopher,
                         args=(chopsticks[n],chopsticks[(n+1) %␣
↪NSTICKS]))
    t.start()
```

æIJĂåŘŐïijŃèęĄçL'źåĹńæşÍæĎŔåĹřïijŃäÿžäźĘéĄ£åĚ■æ■žéŤĄïijŃæL'ĂæIJL'çŽĎåŁăéŤĄæŞ■ä¡IJå£Ěé
`acquire()` åĞ¡æŢřãĂĆåęĆæđIJäżčçăĄäÿ■çŽĎæ§ŘéČĺåĹĘçżŢè£Ğacquire
åĞ¡æŢřçŻť'æŐěçŤşèřůéŤĄïijŃéĆčäźĹæŢť'äÿłæ■žéŤĄéĄ£åĚ■æIJžåĹűåřśäÿ■ètůä¡IJçŤĺäžĘãĂĆ

12.6 ä£İå■Ÿçž£çĺŃçŽĎçŁűæĂĄä£ąæĄŕ

éŮőécŸ

ä¡ăéIJĂèęĄä£İå■Ÿæ■čåIJÍè£ŘèąŇçž£çĺŃçŽĎçŁűæĂĄïijŃè£ŹäÿłçŁűæĂĄåŕźäžŐåĚűäžŰçŽĎçž£çĺŃæŸ

èğčåĘşæŰźæąĹ

æIJL'æŮűåIJĺåď'Žçž£çĺŃçijŰçĺŃäÿ■ïijŇä¡ăéIJĂèęĄåŔłä£İå■Ÿå¡ŞåL'■è£ŘèąŇçž£çĺŃçŽĎçŁűæĂĄãĂĆ èęĄè£ŹäźĹåĄŽïijŇåŔŕä¡£çŤĺ `thread.local()` åĹŻåžžäÿĂäÿłæIJňåIJřçž£çĺŃå■ŸåĆĺåŕźèśąãĂĆ åŕźè£ŹäÿłåŕźèśąçŽĎåšđæĂğçŽĎä£İå■ŸåŠŇèŕžåŔŰæŞ■ä¡IJéČ¡åŔłäijŽåŕźæL'ğèąŇçž£çĺŃåŔŕèğĄïijŇèĂŇåĚ

ä¡IJäÿžä¡£çŤĺæIJňåIJřå■ŸåĆĺçŽĎäÿĂäÿłæIJL'èűčçŽĎåőđéŹĚä¿Ńå■ŘïijŇ èĂČèŹŚåIJĺ8.3åŕŔèŁĆåőŽäźL'è£ĞçŽĎ `LazyConnection` äÿŁäÿŇæŰĞçőąçŘĘåŹĺçśžãĂĆ äÿŃéİćæĹŚäżňåŕźåőČè£ŽèąŇäÿĂäžŽåŕŘçŽĎä£őæŤźä¡£å¿ŮåőČåŘŕäžěéĂĆçŤĺäžŐåď'Žçž£çĺŃïijŽ

```
from socket import socket, AF_INET, SOCK_STREAM
import threading

class LazyConnection:
    def __init__(self, address, family=AF_INET, type=SOCK_STREAM):
        self.address = address
        self.family = AF_INET
        self.type = SOCK_STREAM
        self.local = threading.local()

    def __enter__(self):
        if hasattr(self.local, 'sock'):
            raise RuntimeError('Already connected')
        self.local.sock = socket(self.family, self.type)
        self.local.sock.connect(self.address)
        return self.local.sock

    def __exit__(self, exc_ty, exc_val, tb):
        self.local.sock.close()
        del self.local.sock
```

äžčçăĄäÿ■ïijŇèĞłåůśèğĆår§åŕźäžŐ `self.local` åśđæĂğçŽĎä¡£çŤĺãĂĆ åőČèćńåĹİåğŃåŇŰår¿äÿĂäÿł `threading.local()` åőđä¿ŃãĂĆ åĚűäžŰæŰźæşŢæŞ■ä¡IJèćńå■ŸåĆĺäÿž `self.local.sock` çŽĎåěŮæŐěå■ŮåŕźèśąãĂĆ æIJL'äžĘè£ŹäžŽåŕśåŔŕäžěåIJĺåď'Žçž£çĺŃäÿ■åőL'åĚĺçŽĎä¡£çŤĺ `LazyConnection` åőđä¿ŃäžĘãĂĆä¿ŃąĆïijŽ

```
from functools import partial
def test(conn):
    with conn as s:
        s.send(b'GET /index.html HTTP/1.0\r\n')
        s.send(b'Host: www.python.org\r\n')

        s.send(b'\r\n')
        resp = b''.join(iter(partial(s.recv, 8192), b''))

    print('Got {} bytes'.format(len(resp)))

if __name__ == '__main__':
    conn = LazyConnection(('www.python.org', 80))
```

```
    t1 = threading.Thread(target=test, args=(conn,))
    t2 = threading.Thread(target=test, args=(conn,))
    t1.start()
    t2.start()
    t1.join()
    t2.join()
```

åőČäźŃæĽĂäžěèąŇå¿ŮéĂŽçŽĎåŐ§åŽäæŸŕæŕŘäÿłçž£çĺŃäijŽåĹŽåžžäÿĂäÿłèĞłåůśäÿŞåśđçŽĎåěŮæŐ
åŽäæ■d'ïijŇå¡Şäÿ■åŘŇçŽĎçž£çĺŃæĽğèąŇåěŮæŐěå■ŮæŞ■ä¡IJæŮűïijŇçŤśäžŐæŞ■ä¡IJçŽĎæŸŕäÿ■åŘŇçŽ

èőĺèőž

åIJĺåď ğéČĺåĹEçĺŃåžŔäÿ■åĹŽåžžåŠŇæŞ■ä¡IJçž£çĺŃçL'źåőŽçŁűæĂĄåźűäÿ■äijŽæIJL'äžĂäźĹéŮőécŸã
äÿ■è£ĞïijŇå¡ŞåĞžäžEéŮőécŸçŽĎæŮűůåĂŹïijŇéĂŽåÿÿæŸŕåŽääÿžæ§Řäÿłåŕźèśąèćńåď Žäÿłçž£çĺŃä¡£çŤĺåĹ
æŕŤåęĆäÿĂäÿłåěŮæŐěå■ŮæĹŰæŰĞäžűãĂĆä¡ääÿ■èČ¡èőľæL'ĂæIJL'çž£çĺŃèt'ąçŇőäÿĂäÿłå■ŢçŃňåŕźèśąïijĺ
åŽääÿžåď Žäÿłçž£çĺŃåŘŇæŮűèŕżåŠŇåĘŹçŽĎæŮűůåĂŹäijŽäžğçŤ§æůůäźśãĂĆ
æIJňåIJřçž£çĺŃå■ŸåĆĺéĂŽè£Ğèőľè£ŹäžŽèţĎæžŘåŔłèČ¡åIJĺèćńä¡£çŤĺçŽĎçž£çĺŃäÿ■åŔŕèğĄæİèèğčåĘşè£Ź

æIJňèŁĆäÿ■ïijŇä¡£çŤĺ `thread.local()` åŘŕäžěèőľ `LazyConnection` çśżæŤŕæŇĄäÿĂäÿłçž£çĺŃäÿĂäÿłè£đæŐěïijŇ
èĂŇäÿ■æŸŕåŕźäžŐæL'ĂæIJL'çŽĎè£ŽçĺŃéČ¡åŔłæIJL'äÿĂäÿłè£đæŐěãĂĆ

åĚűåŐ§çŘĘæŸŕïijŇæŕŘäÿł `threading.local()` åőđä¿ŃäÿžæŕŘäÿłçž£çĺŃçżt'æŁd'çİĂäÿĂäÿłå■Ţç
æL'ĂæIJL'æŹőéĂŽåőđä¿ŃæŞ■ä¡IJæŕŤåęĆèŐůåŔŰãĂĄä£őæŤźåŠŇåĹäéŹd'åĂijäžĚäžĚæŞ■ä¡IJè£Źäÿłåůůåĺ
æŕŘäÿłçž£çĺŃä¡£çŤĺäÿĂäÿłçŃňçńŃçŽĎå■ŮåĚÿåŕśåŔŕäžěä£İèŕĄæŢŕæ■őçŽĎéŽŤçęžäžĘãĂĆ

12.7 åĹŽåžžäÿĂäÿłçž£çĺŃæśă

éŮőécŸ

ä¡ăåĹŽåžžäÿĂäÿłåůěä¡IJèĂĚçž£çĺŃæśăïijŇçŤĺæİěçŻÿåžŤåőćæĹůçńŕèŕůæśĆæĹŰæL'ğèąŇåĚűäžŰçŽĎå

èğčåĘşæŰźæąĹ

`concurrent.futures` åĞ¡æŢŕåžŞæIJL'äÿĂäÿł `ThreadPoolExecutor` çśżåŔŕäžěèćńçŤĺæİěåőŇæĹŘè£ŹäÿłäžžåŁąãĂĆ äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎTCPæIJ■åŁąåŹĺijŇä¡£çŤĺäžE

```
from socket import AF_INET, SOCK_STREAM, socket
from concurrent.futures import ThreadPoolExecutor

def echo_client(sock, client_addr):
    '''
    Handle a client connection
    '''
    print('Got connection from', client_addr)
```

```
    while True:
        msg = sock.recv(65536)
        if not msg:
            break
        sock.sendall(msg)
    print('Client closed connection')
    sock.close()

def echo_server(addr):
    pool = ThreadPoolExecutor(128)
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(addr)
    sock.listen(5)
    while True:
        client_sock, client_addr = sock.accept()
        pool.submit(echo_client, client_sock, client_addr)

echo_server(('',15000))
```

åęĆædIJä¡ăæČşæĽŃåŁĺåĹŻåżžä¡äèĞłåůśçŽĎçž£çĺŃæśäïijŇ
éĂŽåÿÿåŕŕäžěä¡£çŤĺäÿĂäÿłQueueæİěè¡żæİ¿åőđçŐřãĂĆäÿŃéİćæŸŕäÿĂäÿłçĺ■å¿őäÿ■åŘŇä¡ĘæŸŕæĽŃåŁĺåő

```
from socket import socket, AF_INET, SOCK_STREAM
from threading import Thread
from queue import Queue

def echo_client(q):
    '''
    Handle a client connection
    '''
    sock, client_addr = q.get()
    print('Got connection from', client_addr)
    while True:
        msg = sock.recv(65536)
        if not msg:
            break
        sock.sendall(msg)
    print('Client closed connection')

    sock.close()

def echo_server(addr, nworkers):
    # Launch the client workers
    q = Queue()
    for n in range(nworkers):
        t = Thread(target=echo_client, args=(q,))
        t.daemon = True
        t.start()

    # Run the server
```

```
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(addr)
    sock.listen(5)
    while True:
        client_sock, client_addr = sock.accept()
        q.put((client_sock, client_addr))

echo_server(('',15000), 128)
```

ä¡£çŤÍ ThreadPoolExecutor çŻÿåŕźäžŐæĽŃåŁĺåőđçŐřçŽĎäÿĂäÿłåě¡åď ĎåIJĺäžŐåőČä¡£å¿Ů äżżåŁąæŔŘäžď èĂĚæŻť æŰźä¿£çŽĎäżŐèćńèřČçŤĺåĞ¡æŢřäÿ■èŐůåŔŰè£ŤåŻđåĂijãĂĆä¿ŃåęĆïijŇä¡ăåŔŕèČ

```
from concurrent.futures import ThreadPoolExecutor
import urllib.request

def fetch_url(url):
    u = urllib.request.urlopen(url)
    data = u.read()
    return data

pool = ThreadPoolExecutor(10)
# Submit work to the pool
a = pool.submit(fetch_url, 'http://www.python.org')
b = pool.submit(fetch_url, 'http://www.pypy.org')

# Get the results back
x = a.result()
y = b.result()
```

ä¿Ńå■Řäÿ■è£ŤåŻđçŽĎhandleåŕźèśąäijŽåÿőä¡ăåď ĎçŘĘæĽĂæIJL'çŽĎéŸżåąđäÿŐå■Řä¡IJïijŇçĎűåŘŐä çĽźåĹńçŽĎïijŇa.result() æŞ■ä¡IJäijŽéŸżåąđè£ŻçÍŃçŻť åĹřåŕźåžŤçŽĎåĞ¡æŢřæĽğèąŇåőŇæĹŘåźűè£Ť

èőÍèőž

éĂŽåÿÿæÌěèőšïijŇä¡ăåžŤèŕěéĄ£åĚ■çijŰåĘŹçž£çÍŃæŢřéĞŘåŔŕäžěæŮäéŹŘåĹűåćđéŢ£çŽĎçÍŃåžŘãĂĆ

```
from threading import Thread
from socket import socket, AF_INET, SOCK_STREAM

def echo_client(sock, client_addr):
    '''
    Handle a client connection
    '''
    print('Got connection from', client_addr)
    while True:
        msg = sock.recv(65536)
        if not msg:
            break
        sock.sendall(msg)
```

```
    print('Client closed connection')
    sock.close()

def echo_server(addr, nworkers):
    # Run the server
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(addr)
    sock.listen(5)
    while True:
        client_sock, client_addr = sock.accept()
        t = Thread(target=echo_client, args=(client_sock, client_
↪addr))
        t.daemon = True
        t.start()

echo_server(('',15000))
```

åř¡çőąè£ŹäÿłäźŞåŘŕäžěåůěä¡IJïijŇ ä¡ĘæŸŕåőČäÿ■èČ¡æŁţå¿ąæIJL'äžžèŕŢåŻ¿éĂŽè£ĞåĹŻåžžåď'ğéĞŔçž
éĂŽè£Ğä¡£çŤĺécĎåĚĹåĹİåğŇåŇŰçŽĎçž£çĺŃæśäïijŇä¡ääŘŕäžěèő¿ç¡őåŘŇæŮűè£ŘèąŇçž£çĺŃçŽĎäÿŁéŹŘa

ä¡ääŘŕèČ¡äijŽåĚşå£ČåĹŻåžžåď'ğéĞŔçž£çĺŃäijŽæIJL'äžĂäźĹåŘŐæđIJãĂĆ
çŐřäžčæŞ■ä¡IJçşżçż§åŘŕäžěå¿Ĺè¡żæİ¿çŽĎåĹŻåžžåĞää■ČäÿłçžçĺŃçŽĎçž£çĺŃæśäãĂĆ
çŤŽèĞşïijŇåŘŇæŮűåĞää■Čäÿłçž£çĺŃç■L'å¿Ěåůěä¡IJåźűäÿ■äijŽåŕźåĚűäžŰäžčçăĄäžğçŤ§æĂğèČ¡å¡śå§■ãĂ
å¡ŞçĎűäžĘïijŇåęĆæđIJæL'ĂæIJL'çž£çĺŃåŘŇæŮűèćńåŤd'éĘŠåźűçńŇå■şåIJĺCPUäÿŁæL'ğèąŇïijŇéĆčåřśäÿ■
éĂŽåÿÿïijŇä¡ääžŤèŕěåŘłåIJĺI/Oåď'ĎçŘĘçŻÿåĚşäžčçăĄäÿ■ä¡£çŤĺçž£çĺŃæśäãĂĆ

åĹŻåžžåď'ğçŽĎçž£çĺŃæśäçŽĎäÿĂäÿłåŕŕèČ¡éIJĂèęĄåĚşæşĺçŽĎéŮőécŸæŸŕåĘĚå■ŸçŽĎä¡£çŤĺãĂĆ
ä¿ŇåęĆïijŇåęĆæđIJä¡ääIJĺOS Xçşżçż§äÿŁéİćåĹŻåžž2000äÿłçž£çĺŃïijŇçşżçż§æŸ¿çď'žPythonè£ŻçĺŃä¡£çŤĺã
äÿ■è£ĞïijŇè£ŹäÿłèőąçőŮéĂŽåÿÿæŸŕæIJL'èŕŕåůőçŽĎãĂĆå¡ŞåĹŻåžžäÿĂäÿłçž£çĺŃæŮűïijŇæŞ■ä¡IJçşżçż§äi
æŤ¿ç¡őçž£çĺŃçŽĎæL'ğèąŇæăĹïijĹéĂŽåÿÿæŸŕ8MBåď'ğåŕŘïijL'ãĂĆä¡ĘæŸŕè£ŹäÿłåĘĚå■ŸåŘłæIJL'äÿĂåŕŔ
åŽăæ■d'ïijŇPythonè£ŻçĺŃä¡£çŤĺåĹřçŽĎçIJ§åőđåĘĚå■ŸåĚűåőđå¿ĹåŕŘ
ïijĹæŕŤåęĆïijŇåŕźäžŐ2000äÿłçž£çĺŃæİèèőšïijŇåŘłä¡£çŤĺåĹřäžĘ70MBçŽĎçIJ§åőđåĘĚå■ŸïijŇèĂŇäÿ■æŸŕ
åęĆæđIJä¡ääŃĚå£ČèŹŽæŃ§åĘĚå■Ÿåď'ğåŕŘïijŇåŘŕäžěä¡£çŤĺ `threading.stack_size()` åĞ¡æŢřæİěéŹ■ä¡ŐåőČãĂĆä¿ŃåęĆïijŽ

```
import threading
threading.stack_size(65536)
```

åęĆæđIJä¡ääŁääÿŁè£ŹæIąèŕ■åŘěåźűåĘ■æňąè£ŘèąŇåL'■éİćçŽĎåĹŻåžž2000äÿłçž£çĺŃèŕŢéłŇïijŇ
ä¡ääijŽåŔŠçŐřPythonè£ŻçĺŃåŘłä¡£çŤĺåĹřäžĘåď'ğæęĆ210MBçŽĎèŹŽæŃ§åĘĚå■ŸïijŇèĂŇçIJ§åőđåĘĚå■Ÿ
æşĺæĎŘçž£çĺŃæăĹåď'ğåŕŘå£ĚéąžèĞşåŕŠäÿž32768å■ŮèŁĆïijŇéĂŽåÿÿæŸŕçşżçż§åĘĚå■Ÿéąţåď'ğåŕŘïijĹ409

12.8 çőĂå■ŢçŽĎåźűèąŇçijŰçĺŃ

éŮőécŸ

ä¡ääIJL'äÿłçĺŃåžŘèęĄæL'ğèąŇCPUåŕĘéŻĘåđŃåůěä¡IJïijŇä¡ääČşèől'äžŰåĹl'çŤĺåď'ŽæäÿCPUçŽĎäijŸå

èğčåĘşæŰźæąĹ

concurrent.futures åžŞæŔŘä¿ŻäžĘäÿĂäÿł ProcessPoolExecutor çśżïijŇ åŔŕèćńçŤĺæİěåIJĺäÿĂäÿłå■ŢçŃňçŽĎPythonèğčéĞŁåŹĺäÿ■æĽğèąŇèőąçőŮåŕĘéŻĘåđŃåĞ¡æŢřãĂĆäÿ■è£ĞïijŇèęĄä¡£çŤĺåőČïijŇä¡ăéęŰåĚĹèęĄæIJL'äÿĂäžŻèőąçőŮåŕĘéŻĘåđŃçŽĎäżżåŁąãĂĆæĹŚäžňéĂŽè£ĞäÿĂäÿłçőĂå■ŢèĂŇåőđéŹĚçŽĎä¿Ńå■ŘæİěæijŤçd'žåőČãĂĆåĄĞåőŽä¡ăæIJL'äÿłApache webæIJ■åŁąåŹĺæŮěå£ŮçŻőå¡ŢçŽĎgzipåŐŃçijľåŇĚïijŽ

```
logs/
   20120701.log.gz
   20120702.log.gz
   20120703.log.gz
   20120704.log.gz
   20120705.log.gz
   20120706.log.gz
   ...
```

è£ŻäÿĂæ■ěåĄĞèő¿æŕŔäÿłæŮěå£ŮæŰĞäžűåĘĚåőźçśżäijijäÿŃéİćè£Źæăůïijž

```
124.115.6.12 - - [10/Jul/2012:00:18:50 -0500] "GET /robots.txt ..."␣
↪200 71
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /ply/ ..." 200␣
↪11875
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /favicon.ico ..
↪." 404 369
61.135.216.105 - - [10/Jul/2012:00:20:04 -0500] "GET /blog/atom.xml␣
↪..." 304 -
...
```

äÿŃéİćæŸŕäÿĂäÿłèĎŽæIJňïijŇåIJĺè£ŹäžŻæŮěå£ŮæŰĞäžűäÿ■æ§ěæĽ¿åĞžæĽĂæIJL'èő£éŮőè£Ğrobot

```
# findrobots.py

import gzip
import io
import glob

def find_robots(filename):
    '''
    Find all of the hosts that access robots.txt in a single log␣
↪file
    '''
    robots = set()
    with gzip.open(filename) as f:
        for line in io.TextIOWrapper(f,encoding='ascii'):
            fields = line.split()
            if fields[6] == '/robots.txt':
                robots.add(fields[0])
    return robots

def find_all_robots(logdir):
```

```
    '''
    Find all hosts across and entire sequence of files
    '''
    files = glob.glob(logdir+'/*.log.gz')
    all_robots = set()
    for robots in map(find_robots, files):
        all_robots.update(robots)
    return all_robots

if __name__ == '__main__':
    robots = find_all_robots('logs')
    for ipaddr in robots:
        print(ipaddr)
```

åĽ■éİćçŽĎçÍŃåžŔä¡£çŤÍäžĘéĂŽåÿÿçŽĎmap-reduceéčŐæäijæİěçijŰåĘŹãĂĆ åĞ¡æŢř `find_robots()` åIJÍäÿĂäÿłæŰĞäżűåŘ■éŻĘåŘĹäÿŁåĄŽmapæŞ■ä¡IJïijŇåźűåřĘçżŞæđIJæśĞæĂżäÿžäÿĂääź§åřśæŸŕ `find_all_robots()` åĞ¡æŢřäÿ■çŽĎ `all_robots` éŻĘåŘĹãĂĆçŐřåIJÍïijŇåĄĞèő¿ä¡ăæČşèęĄä£őæŤźè£ŹäÿłçÍŃåžŔèőľåőČä¡£çŤÍåđŽæäÿCPUãĂĆå¿ĹçőĂå■ŢâĂŤâĂŤåŘłéIJĂèęĄåřĘmap()æŞ■ä¡IJæŻ£æ■ćäÿžäÿĂäÿł `concurrent.futures` åžŞäÿ■çŤ§æĹŘçŽĎçśżäijijæŞ■ä¡IJå■şåŕŕãĂĆäÿŃéİćæŸŕäÿĂäÿłçőĂå■Ţä£őæŤźçĽĹæIJñïijŽ

```
# findrobots.py

import gzip
import io
import glob
from concurrent import futures

def find_robots(filename):
    '''
    Find all of the hosts that access robots.txt in a single log␣
↪file

    '''
    robots = set()
    with gzip.open(filename) as f:
        for line in io.TextIOWrapper(f,encoding='ascii'):
            fields = line.split()
            if fields[6] == '/robots.txt':
                robots.add(fields[0])
    return robots

def find_all_robots(logdir):
    '''
    Find all hosts across and entire sequence of files
    '''
    files = glob.glob(logdir+'/*.log.gz')
    all_robots = set()
    with futures.ProcessPoolExecutor() as pool:
```

```
        for robots in pool.map(find_robots, files):
            all_robots.update(robots)
    return all_robots

if __name__ == '__main__':
    robots = find_all_robots('logs')
    for ipaddr in robots:
        print(ipaddr)
```

éĂŽè£Ğè£Źäÿŀä£őæŤźåŘŐïijŇè£ŘèąŇè£ŹäÿłèĎŽæIJňäžğçŤ§åŘŇæăůçŽĎçżŞæđIJïijŇä¡ĘæŸŕåIJĺåŻŻ
åőđéŹĚçŽĎæĂğèČ¡äijŸåŇŰæŢĹæđIJæăźæ■őä¡äçŽĎæIJžåŹĺCPUæŢřéĞŘçŽĎäÿ■åŘŇèĂŇäÿ■åŘŇãĂĆ

èőĺèőž

ProcessPoolExecutor çŽĎåĚÿåđŃçŤĺæşŢåęĆäÿŃïijŽ

```
from concurrent.futures import ProcessPoolExecutor

with ProcessPoolExecutor() as pool:
    ...
    do work in parallel using pool
    ...
```

åĚűåŐ§çŘĘæŸŕïijŇäÿĂäÿł ProcessPoolExecutor
åĹŻåžžNäÿłçŃňçńŃçŽĎPythonèğčéĞŁåŹĺïijŃ NæŸŕçşżçż§äÿŁéİćåŘŕçŤĺCPUçŽĎäÿłæŢřãĂĆä¡ååŘŕäżěéĂŽ
ProcessPoolExecutor(N) æİëä£őæŤź åď˙ĎçŘĘåŹĺæŢřéĞŘãĂĆè£Źäÿłåď˙ĎçŘĘæśääijŽäÿĂçŻť˙è£Řèą
çĎűåŘŐåď˙ĎçŘĘæśäècńåĚşéŮ■ãĂĆäÿ■è£ĞïijŇçĺŃäžŘäijŽäÿĂçŻť˙ç■Ľ˙å¿ĚçŻť˙åĹřæĽ˙ĂæIJĽ˙æŔŘäžď˙çŽĎå

èćńæŔŘäžď˙åĹřæśääÿ■çŽĎåůěä¡IJå£ËéążèćńåőŽäźĽ˙äÿžäÿĂäÿłåĞ¡æŢřãĂĆæIJĽ˙äÿď˙çğ■æŰźæşŢåŐżæ
åęĆæđIJä¡äæČşèőľ˙äÿĂäÿłåĹŮèąĺæŐĺårijæĹŰäÿĂäÿł map()
æŞ■ä¡IJåźűèąŇæĽ˙ğèąŇçŽĎèŕÏïijŇåŔŕä¡£çŤĺ pool.map() :

```
# A function that performs a lot of work
def work(x):
    ...
    return result

# Nonparallel code
results = map(work, data)

# Parallel implementation
with ProcessPoolExecutor() as pool:
    results = pool.map(work, data)
```

åŘęåď˙ŰïijŇä¡ääŔŕäżěä¡£çŤĺ pool.submit() æİěæĽ˙ŃåŁĺçŽĎæŔŘäžď˙å■ŢäÿłäżžåŧąïijŽ

```
# Some function
def work(x):
    ...
```

```
    return result

with ProcessPoolExecutor() as pool:
    ...
    # Example of submitting work to the pool
    future_result = pool.submit(work, arg)

    # Obtaining the result (blocks until done)
    r = future_result.result()
    ...
```

åęĆæđIJä¡ăæĽŃåŁĺæŔŘäžd'äÿĂäÿłäżżåŁąïijŇçżŞæđIJæŸŕäÿĂäÿł Future åőđä¿ŃãĂĆ èęĄèŐůåŔŰæIJĂçżĹçżŞæđIJïijŇä¡ăéIJĂèęĄěŕĈçŤĺáőĈçŽĎ result() æŰźæşŢãĂĆ åőČäijŽéŸżåąđè£ŽçĺŃçŻť åĽřçżŞæđIJèćńè£ŤåŻđæİěãĂĆ

åęĆæđIJäÿ■æČşéŸżåąđïijŇä¡ăè£ŸåŔŕäžěä¡£çŤĺäÿĂäÿłåŻđěŕČåĞ¡æŢřïijŇä¿ŃåęĆïijŽ

```
def when_done(r):
    print('Got:', r.result())

with ProcessPoolExecutor() as pool:
     future_result = pool.submit(work, arg)
     future_result.add_done_callback(when_done)
```

åŽđèřČåĞ¡æŢřæŐěåŔŮäÿĂäÿł Future åőđä¿ŃïijŇèćńçŤĺæİěèŐůåŔŰæIJĂçżĹçŽĎçżŞæđIJïijĹæŕŤåęĆ
år¡çőąåd'ĎçŘĘæśăå¿Łåőźæ Ÿşä¡£çŤĺïijŇåIJĺèő¿èőąåd'ğçĺŃåžŔçŽĎæŮűåĂŹè£ŸæŸŕæIJL'å¿Łåd'ŽéIJĂèęĄæ

- è£Źçğ■åźűèąŇåd'ĎçŘĘæŁĂæIJŕåŔłéĂĆçŤĺäžŐéĆčäžŻåŔŕäžěèćńåĹĘèğčäÿžäžŠçŻÿçŃňçńŃéČĺåĹĘçŽ
- èćńæŔŘäžd'çŽĎäżżåŁąå£Ěéążæ ŸŕçőĂå■ŢåĞ¡æŢřå¡ćåijŔãĂĆåŕźäžŐæŰźæşŢãĂĄéŮ■åŇĚåŠŇåĚűäżŮ
- åĞ¡æŢřåŔĆæŢřåŠŇè£ŤåŻđåĂijå£ĚéążåĕijåőźpickleïijŇåŻääÿžèęĄä¡£çŤĺåĹrè£ŻçĺŃéŮť'çŽĎéĂŽä£ąï
- èćńæŔŘäžd'çŽĎäżżåŁąåĞ¡æŢřäÿ■åžŤä£ĺçŢŹçŁűæĂĄæĹŰæIJL'åL'ŕä¡IJçŤĺãĂĆéŹd'äžĘæL'Şå■řæŮěå

  äÿĂæŮęåŘŕåŁĺä¡ääÿ■èČ¡æŐğåĹűå■Řè£ŻçĺŃçŽĎäżżä¡Ţèą Ñäÿžïij ŇåŽăæ■d'æIJĂåě¡ä£ĺæŇĄçőĂå■ŢåŠ

- åIJĺUnixäÿŁè£ŻçĺŃæśăéĂŽè£ĞěŕČçŤĺ fork() çşżçż§ěŕČçŤĺèćńåĹŽåżžïijŇ

åőČäijŽåĚŃéŽĘPythonèğčéĞŁåŹĺïijŇåŇĚæŃňforkæŮűçŽĎæL'ĂæIJL'çĺŃåžŔçŁűæĂĄãĂĆ èĂŇåIJĺWindowsäÿŁïijŇåĚŃéŽĘèğčéĞŁåŹĺæŮűäÿ■äijŽåĚŃéŽĘçŁűæĂĄãĂĆ åőđéŹĚçŽĎ-forkæŞ■ä¡IJäijŽåIJĺçňňäÿĂæňąěŕČçŤĺ pool.map() æĹŰ pool.submit() åŘŐåŔŠçŢ§ãĂĆ

- å¡Şä¡ăæůůåŘĹä¡£çŤĺè£ŻçĺŃæśăåŠŇåd'Žçž£çĺŃçŽĎæŮűåŹèęĄçL'źåĹńåřŕå£ČãĂĆ

  ä¡ăåžŤèŕěåIJĺåĹŻåżžäżżä¡Ţçž£çĺŃäźŃåL'■åĚĹåĹŻåżžåźűæ£Ăæť'è£ŻçĺŃæśäïijĹæŕŤåęĆåIJĺçĺŃåžŔåŘŕ

12.9 PythonçŽĎåĚĺåśĂéŤĄéŮőécŸ

éŮőécŸ

ä¡ăåůšçżŕåŘňèŕť è£ĞåĚĺåśĂèğčéĞŁåŹĺéŤĄGILïijŇæŃĚå£ČåőČäijŽå¡śåŞ■åĹřåď Žçž£çĺŃçĺŃåžŕçŽĎæ

èğčåĘşæŰźæąĹ

åř¡çőąPythonåőŇåĚĺæŤŕæŇĄåď Žçž£çĺŃçijŰçĺŃïijŇ ä¡ĘæŸŕèğčéĞŁåŹĺçŽĎCèŕ■èĺĂåőđçŐřéČĺåĹĘåIJĺ
åőđéŹĚäÿŁïijŇèğčéĞŁåŹĺèćńäÿĂäÿłåĚĺåśĂèğčéĞŁåŹĺéŤĄä£ÍæŁď çİĂïijŇåőČçąőä£Íäżżä¡ŢæŮűåĂŹéČ¡åŕ
GILæIJĂåď ğçŽĎéŮőécŸåřśæŸŕPythonçŽĎåď Žçž£çĺŃçĺŃåžŕåźűäÿ■èČ¡åĹľçŤĺåď Žæăÿ CPUçŽĎäijŸåŁ£
ïijĹærŤåęĆäÿĂäÿłä¡£çŤĺäžĘåď Žäÿłçž£çĺŃçŽĎèőąçőŮåŕĘéŻĘåđŃçĺŃåžŕåŘłäijŽåIJĺäÿĂäÿłå■ŢCPUäÿŁéÍć

åIJĺèőĺèőžæŹőéĂŽçŽĎGILäźŃåL’■ïijŇæIJL’äÿĂçĆźèęĄåijžèřČçŽĎæŸŕGILåŘłäijŽå¡śåŞ■åĹréĆčäžŽäÿ
åęĆæđIJä¡ăçŽĎçĺŃåžŕåď ğéČĺåĹĘåŘłäijŽèő¿èőąåĹřI/OïijŇærŤåęĆç¡ŚçżIJäžď äžŠïijŇéĆčäźĹä¡£çŤĺåď Žçž£
åŻääÿžåőČäžňåď ğéČĺåĹĘæŮűéŮt´éČ¡åIJĺç■L’å¿ĚãĂĆåőđéŹĚäÿŁïijŇä¡ăåőŇåĚĺåŕäžěæŤ¿å£ČçŽĎåĹŻåžžå
çŐřäžčæŞ■ä¡IJçşżçż§è£ŘèąŇè£ŹäźĹåď Žçž£çĺŃæšąæIJL’äżżä¡ŢåŐŃåŁŻïijŇæšąåŢěåŕræŃĚå£ČçŽĎãĂĆ

èĂŇåŕźäžŐä¿ÍètŰ CPUçŽĎçĺŃåžŕïijŇä¡ăéIJĂèęĄåijĐæÿĚæěŽæL’ğèąŇçŽĎèőąçőŮçŽĎçL’źçĆźãĂĆ
ä¿ŃåęĆïijŇäijŸåŇŰåžŢåśĆçőŮæşŢèęĄæŕŤä¡£çŤĺåď Žçž£çĺŃè£ŘèąŇå£ńå¿Ůåď ŽãĂĆ
çśżäijijçŽĎïijŇçŤśäžŐPythonæŸŕèğčéĞŁæL’ğèąŇçŽĎïijŇåęĆæđIJä¡ăåřĘéĆčäžŽæĂğèČ¡çŞűéĆĹäžčçăĄçğżå
éĂ§åžęäź§äijŽæŔŘå■ĞçŽĎå¿Ĺå£ńãĂĆåęĆæđIJä¡ăèęĄæŞ■ä¡IJæŢřçŽĎïijŇéĆčäźĹä¡£çŤĺNumPyè£ŹæăůçŽĎ
æIJĂåŘŐïijŇä¡ăè£ŸåŔŕäžěèĂČèŹŠäÿŃåĚűäžŰåŘŕéĂL’ăődçŐřæŰźæąĹïijŇærŤåęĆPyPyïijŇăőČéĂŽè£ĞäÿĂ
ïijĹäÿ■è£ĞåIJĺåĘŹè£ŹæIJñäźęçŽĎæŮűåĂŹăőČè£Ÿäÿ■èČ¡æŤŕæŇĄPython 3ïijL’ãĂĆ

è£ŸæIJL’äÿĂçĆźèęĄæşĺæĐŔçŽĎæŸŕïijŇçž£çĺŃäÿ■æŸŕäÿŞéŮĺçŤĺæİěäijŸåŇŰæĂğèČ¡çŽĎãĂĆ
äÿĂäÿłCPUä¿ÍètŰåđŃçĺŃåžŕåŔŕèČ¡äijŽä¡£çŤĺçž£çĺŃæİěçőąçŘĘäÿĂäÿłåŻ¿å¡ćçŤĺæĹůçŢŇéÍćãĂĄäÿĂäÿłç¡
è£ŹæŮűåĂŹïijŇGILäijŽäžğçŤ§äÿĂäžŽéŮőécŸïijŇåŽääÿžåęĆæđIJäÿĂäÿłçž£çĺŃéŢ£æIJ§æŇĄæIJL’GILçŽĐ
äžŃåődäÿŁïijŇäÿĂäÿłåĘçŽĎäÿ■åě¡çŽĎCèŕ■èĺĂæL’ľåśŢäijŽåŕijèĞt’è£ŹäÿłéŮőécŸæŻť åŁääÿěéĞ■ïijŇ
åř¡çőąäžčçăĄçŽĎèőąçőŮéČĺåĹĘäijŽærŤäźŃåL’■è£ŘèąŇçŽĎæŻť å£ńäžŽãĂĆ

èŕť äžĘè£ŹäźĹåď ŽïijŇçŐřåIJĺæČşèŕť çŽĎæŸŕæĹŚäžňæIJL’äÿd’çğ■ç■ŰçŢěæİěèğčåĘşGILçŽĎçijžçĆźãĂ
éęŰåĚĹïijŇåęĆæđIJä¡ăåőŇåĚĺåůěä¡IJäžŐPythonçŐŕåćČäÿ■ïijŇä¡ăåŔŕäžěä¡£çŤĺ
`multiprocessing` æĺąåİŮæİěåĹŻåžžäÿĂäÿłè£ŻçĺŃæšäïijŇ
åźűåČŕå■ŔåŘŇåď ĎçŘĘåŹĺäÿĂæăůçŽĎä¡£çŤĺåőČãĂĆä¿ŃåęĆïijŇåŁăåĚěä¡ăæIJL’åęĆäÿŃçŽĎçž£çĺŃäžčçă

```
# Performs a large calculation (CPU bound)
def some_work(args):
    ...
    return result

# A thread that calls the above function
def some_thread():
    while True:
        ...
        r = some_work(args)
    ...
```

ä£őæŤźäžčçăĄïijŇä¡£çŤĺè£ŻçÍŃæśăïijŽ

```
# Processing pool (see below for initiazation)
pool = None

# Performs a large calculation (CPU bound)
def some_work(args):
    ...
    return result

# A thread that calls the above function
def some_thread():
    while True:
        ...
        r = pool.apply(some_work, (args))
        ...

# Initiaze the pool
if __name__ == '__main__':
    import multiprocessing
    pool = multiprocessing.Pool()
```

è£ŹäÿłéĂŽè£Ğäĵ£çŤĺäÿĂäÿłæŁĂåůğåĹ'çŤĺè£ŻçÍŃæśăèğčåĘşäžĘGILçŽĎéŮőécŸãĂĆ
å¡ŞäÿĂäÿłçž£çÍŃæČşèęĄæL'ğèąŇCPUåŕĘéŻĘåđŃåůëä¡IJæŮűïijŇäijŽåřĘäżżåŁąåŔŞçżŹè£ŻçÍŃæśãĂĆ
çĎűåŘŐè£ŻçÍŃæśääijŽåIJĺåŔęåď'ŰäÿĂäÿłè£ŻçÍŃäÿ■åŘŕåŁĺäÿĂäÿłå■ŢçŃňçŽĎPythonèğčéĞŁåŹĺæİěåůëä¡I
å¡Şçž£çÍŃç■L'å¿ĚçżŞæđIJçŽĎæŮűåĂŹäijŽéĞŁæŤ¿GILãĂĆ åźűäÿŤïijŇçŤśäžŐèőąçőŮäżżåŁąåIJĺå■ŢçŃňèğ
åIJĺäÿĂäÿłåď'Žæäÿçşżçż§äÿŁéİćïijŇä¡ääijŽåŔŞçŐřè£ŹäÿłæŁĂæIJŕåŔŕäžěèől'ä¡åå¿Łåě¡çŽĎåĹl'çŤĺåď'ŽCPU

åŘęåď'ŰäÿĂäÿłèğčåĘşGILçŽĎç■ŰçŢěæŸŕä¡£çŤĺCæL'l'åśŢçijŰçÍŃæŁĂæIJŕãĂĆ
äÿżèęĄæĂİæČşæŸŕåřĘèőąçőŮåŕĘéŻĘåđŃäżżåŁąè¡ňçğżçżŹCïijŇèů§PythonçŃňçńŃïijŇåIJĺåůëä¡IJçŽĎæŮűş
è£ŹåŔŕäžěéĂŽè£ĞåIJĺCäžčçăĄäÿ■æŔŠåĚëäÿŃéİćè£ŹæăůçŽĎçL'źæőŁåőŔæİěåőŇæĹŘïijŽ

```
#include "Python.h"
...

PyObject *pyfunc(PyObject *self, PyObject *args) {
   ...
   Py_BEGIN_ALLOW_THREADS
   // Threaded C code
   ...
   Py_END_ALLOW_THREADS
   ...
}
```

åęĆæđIJä¡ăä¡£çŤĺåĚűäžŰåůěåĚůèő£éŮőCèŕ■èĺĂïijŇærŤåęĆåŕźäžŐCythonçŽĎctypesåžŞïijŇä¡ăäÿ■éIJĂ
ä¿ŇåęĆïijŇctypesåIJĺèřČçŤĺCæŮűäijŽèĞłåŁĺéĞŁæŤ¿GILãĂĆ

èőĺèőž

èőÿåď˙ŽçĺŃåžŔåŚŸåIJĺéİćåŕźçž£çĺŃæĂğèČ¡éŮőécŸçŽĎæŮűåĂŹïijŇél˙ňäýŁåřśäijŽæĂłç¡łGILïijŇäżĂ
åĚűåőđè£Źæăůå■Řåď˙łäÿ■åŐŽéĄŞäź§åď˙łåď˙l˙çIJ§äžĘçĆźãĂĆ
ä¡IJäÿžäýĂäýłçIJ§åőđçŽĎä¿Ńå■ŘïijŇåIJĺåď˙Žçž£çĺŃçŽĎç¡ŚçżIJçijŰçĺŃäÿ■çěđçğŸçŽĎ
`stalls`åŔŕèČ¡æŸŕåŻääÿžåĚűäżŰåŐ§åŽäæŕŤåęĆäýĂäýłDNSæ§ěæL˙¿åžűæŮűïijŇèĂŇèů§GILæŕńæŮăåĚş
æIJĂåŘŐä¡ăçIJ§çŽĎéIJĂèęĄåĚĹåŐżæŘđæĞĆä¡ăçŽĎäžčçăĄæŸŕåŘęçIJ§çŽĎèćńGILå¡śåŞ■åĹřãĂĆ
åŘŇæŮűè£ŸèęĄæŸŐçŹ¡GILåď˙ğéČĺåĹĘéČ¡åžŤèŕěåŔłåĚşæşĺCPUçŽĎåď˙ĎçŘĘèĂŇäÿ■æŸŕI/O.

åęĆæđIJä¡ăåĞĘåď˙Ğä¡£çŤĺäÿĂäýłåď˙ĎçŘĘåŹĺæśăïijŇæşĺæĎŔçŽĎæŸŕè£ŹæăůåĄŽæűL˙åŘŁåĹřæŢřæ■ô
èćńæL˙ğèąŇçŽĎæŞ■ä¡IJéIJĂèęĄæŤ¿åIJĺäÿĂäýłéĂŽè£Ğdefèŕ■åŘěåőŽäźL˙çŽĎPythonåĞ¡æŢřäÿ■ïijŇäÿ■èČ¡
åźüäÿŤåĞ¡æŢřåŔĆæŢřåŠŇè£ŤåŽđåĂijå£ĚéążèęĄåĚijåőźpickleãĂĆ
åŘŇæăůïijŇèęĄæL˙ğèąŇçŽĎäżżåŁąéĞŔå£Ěéążèűşåď˙§åď˙ğäžěåijèèąěéćİåď˙ŰçŽĎéĂŽåőţåijĂéŤĂãĂĆ

åŘęåď˙ŰäÿĂäýłéŽ¿çĆźæŸŕå¡Şæůůåŕĺä¡£çŤĺçž£çĺŃåŠŇè£ŽçĺŃæśăçŽĎæŮűåĂŹäijŽèől˙ä¡ăå¿Ĺåď˙t˙çŰi
åęĆæđIJä¡ăèęĄåŘŇæŮűä¡£çŤĺäÿd˙èĂĚïijŇæIJĂåě¡åIJĺçĺŃåžŔåŕŕåŁĺæŮűïijŇåĹŽåžžäżżä¡Ţçž£çĺŃäźŃåL˙■å
çĎűåŘŐçž£çĺŃä¡£çŤĺåŘŇæăůçŽĎè£ŽçĺŃæśæİěè£ŽèąŇåőČäżňçŽĎèőąçőŮårĘéŽĘåđŃåůěä¡IJãĂĆ

CæL˙l˙åśŢæIJĂéĞ■èęĄçŽĎçL˙źå¿ĄæŸŕåőČäżňåŠŇPythonèğčéĞŁåŹĺæŸŕä£ĺæŇĄçŃňçńŃçŽĎãĂĆ
äź§åŕśæŸŕèŕť˙ïijŇåęĆæđIJä¡ăåĞĘåď˙ĞåŕĘPythonäÿ■çŽĎäżżåŁąåĹĘéĚ■åĹřCäÿ■åŐżæL˙ğèąŇïijŇ
ä¡ăéIJĂèęĄçąőä£ĺCäżčçăĄçŽĎæŞ■ä¡IJèů§Pythonä£ĺæŇĄçŃňçńŃïijŇ
è£ŹåŕśæĎŔåŠşçİĂäÿ■èęĄä¡£çŤĺPythonæŢřæ■őçżŞæđĎäżěåŔŁäÿ■èęĄèŕČçŤĺPythonçŽĎC
APIãĂĆ åŘęåď˙ŰäÿĂäýłåŕśæŸŕä¡ăèęĄçąőä£ĺCæL˙l˙åśŢæL˙ĂåĄŽçŽĎåůěä¡IJæŸŕèűşåď˙§çŽĎïijŇåĂijå¿Ůä¡ăè
äź§åŕśæŸŕèŕť˙CæL˙l˙åśŢæŃĚèt˙§ètűäžĘåď˙ğéĞŔçŽĎèőąçőŮäżżåŁąïijŇèĂŇäÿ■æŸŕåŕŚæŢřåĞääýłèőąçőŮãĂĆ

è£ŹäžŽèğčåĘşGILçŽĎæŰźæąĹåźűäÿ■èČ¡éĂĆçŤĺäžŐæL˙ĂæIJL˙éŰőécŸãĂĆ
ä¿ŃåęĆïijŇæ§ŘäžŽçśžåđŃçŽĎåžŤçŤĺçĺŃåžŔåęĆæđIJèćńåĹĘèğčäÿžåď˙Žäÿłè£ŽçĺŃåď˙ĎçŘĘçŽĎèŕİåźűäÿ■è
äź§äÿ■èČ¡åŕĘåőČçŽĎéČĺåĹĘäžčçĂąæŤźæĹŘCèŕ■èĺĂæL˙ğèąŇãĂĆ
åŕźäžŐè£ŹäžŽåžŤçŤĺçĺŃåžŔïijŇä¡ăåŕśèęĄèĞłåůśéIJĂæśĆèğčåĘşæŰźæąĹäžĘ
ïijĹæŕŤåęĆåď˙Žè£ŽçĺŃèő£éŮőåĚśäžńåĘĚå■ŸåŇžïijŇåď˙ŽèğčæđŕåŹĺè£ŔèąŇäžŐåŘŇäÿĂäýłè£ŽçĺŃç■L˙ïijL
æĹŰèĂĚïijŇä¡ăè£ŸåŔŕäžěèĂČèŹŠäÿŃåĚűäżŰçŽĎèğčéĞŁåŹĺåőđçŐŕïijŇæŕŤåęĆPyPyãĂĆ

äžĘèğčæŽť˙åď˙ŽåĚşäžŐåIJĺCæL˙l˙åśŢäÿ■éĞŁæŤ¿GILïijŇèŕůåŔĆèĂČ15.7åŠŇ15.10åŕŔèŁĆãĂĆ

12.10 åőŽäźĽäÿĂäýłActoräżżåŁą

éŮőécŸ

ä¡ăæČşåőŽäźL˙èů§actoræĺąåijŔäÿ■çśžäijijâĂIJactorsâĂİèğŠèL˙šçŽĎäżżåŁą

èğčåĘşæŰźæąĹ

actoreæĺąåijŔæŸŕäÿĂçğ■æIJĂåŔď˙èĂĄçŽĎäź§æŸŕæIJĂçőĂå■ŢçŽĎåźűèąŇåŠŇåĹĘåÿČåijŔèőąçőŮèğ
äžŃåődäÿŁïijŇåőČåď˙l˙çŤ§çŽĎçőĂå■ŢæĂğæŸŕåőČåęĆæ■ď˙åŘŮæńćè£ŐçŽĎéĞ■èęĄåŐ§åŽääźŃäÿĂãĂĆ
çőĂå■ŢæİěèőšïijŇäÿĂäýłactoråŕśæŸŕäÿĂäýłåźűåŘŚæL˙ğèąŇçŽĎäżżåŁąïijŇåŘłæŸŕçőĂå■ŢçŽĎæL˙ğèąŇåŔ
åŞ■åžŤè£ŹäžŽæűĹæĄŕæŮűïijŇåőČåŔŕèČ¡è£ŸäijŽçżŹåĚűäżŰactoråŘŚéĂĄæŽť˙è£ŽäÿĂæ■ęçŽĎæűĹæĄŕãĂ

actoräźŃéŮťçŽĎéĂŽä£ąæŸŕå■ŢåŘŚåŠŇåijĆæ■ěçŽĎãĂĆåŻäæ■ď˙ijŇæűĹæĄŕåŘŚéĂĄèĂĚäÿ■ç§ěéĄŞæűĹäźŞäÿ■äijŽæŐěæŤűåĹřäÿĂäÿłæúĹæĄŕåůšèćńåď˙ĎçŘĘçŽĎåŻđåžŤæĹŰéĂŽç§ěãĂĆ

çżŞåŘĹä¡£çŤĺäÿĂäÿłçž£çÍŃåŠŇäÿĂäÿłéŸ§åĹŮåŔŕäżěå¿Ĺåőźæ ŸŞçŽĎåőŽäźĽ'actorïijŇä¿ŃåęĆïijŽ

```
from queue import Queue
from threading import Thread, Event

# Sentinel used for shutdown
class ActorExit(Exception):
    pass

class Actor:
    def __init__(self):
        self._mailbox = Queue()

    def send(self, msg):
        '''
        Send a message to the actor
        '''
        self._mailbox.put(msg)

    def recv(self):
        '''
        Receive an incoming message
        '''
        msg = self._mailbox.get()
        if msg is ActorExit:
            raise ActorExit()
        return msg

    def close(self):
        '''
        Close the actor, thus shutting it down
        '''
        self.send(ActorExit)

    def start(self):
        '''
        Start concurrent execution
        '''
        self._terminated = Event()
        t = Thread(target=self._bootstrap)

        t.daemon = True
        t.start()

    def _bootstrap(self):
        try:
            self.run()
        except ActorExit:
```

```
            pass
        finally:
            self._terminated.set()

    def join(self):
        self._terminated.wait()

    def run(self):
        '''
        Run method to be implemented by the user
        '''
        while True:
            msg = self.recv()

# Sample ActorTask
class PrintActor(Actor):
    def run(self):
        while True:
            msg = self.recv()
            print('Got:', msg)

# Sample use
p = PrintActor()
p.start()
p.send('Hello')
p.send('World')
p.close()
p.join()
```

è£Źäÿłä¿Ńå■Řäÿ■ïijŇä¡ää¡£çŤÍactoråőđä¿ŃçŽĎ send() æŰźæşŢåŔŚéĂĄæűĹæĄŕçżŹåőČäžñăĂĆ åĚűæIJžåĹűæŸŕïijŇè£ŹäÿłæŰźæşŢäijŽåřĘæűĹæĄŕæŤ¿åĚëäÿĂäÿçĎűůåŘŐåřĘåĚűè¡ñäžd'çżŹåd'ĎçŘĘèćńæŐëåŔŮæűĹæĄŕçŽĎäÿĂäÿłåĘĚéČÍçž£çÍŇãĂĆ close() æŰźæşŢéĂŽè£ĞåIJÍéŸ§åĹŮäÿ■æŤ¿åĚëäÿĂäÿłçĽźæőŁçŽĎåŞÍåĚţåĂijïijÍActorExitïijĽæÌěåĚşéçŤÍæĹůåŔŕäžëéĂŽè£ĞçżğæĽ£ActoråźűåőŽäźĽåőđçŐřèĞłåůšåd'ĎçŘĘéĂżè¿Śrun()æŰźæşŢæÌěåőŽäźĽæŰřActorExit åijĆåÿÿçŽĎä¡£çŤÍåřśæŸŕçŤÍæĹůèĞłåőŽäźĽäžčçĂąåŔŕäžěåIJÍéIJĂèęĄçŽĎæŮűåĂŹæÌěæ■ŢèĆïijĹåijĆåÿÿèćńget()æŰźæşŢæŁŻåĞžåźűäijäæŠ■åĞžåŐżïijĽãĂĆ

åęĆædIJä¡ăæŤ¿åő¡åŕźäžŐåŘŇæ■ěåŠŇåijĆæ■ěæűĹæĄŕåŘŚéĂĄçŽĎèęĄæśĆïijŇ çśżactoråŕźèśąè£ŸåŔŕäžëéĂŽè£ĞçŤ§æĹŘåŹÍæÌěçőĂåŇŮåőŽäźĽãĂĆä¿ŃåęĆïijŽ

```
def print_actor():
    while True:

        try:
            msg = yield       # Get a message
            print('Got:', msg)
        except GeneratorExit:
            print('Actor terminating')

# Sample use
```

```
p = print_actor()
next(p)      # Advance to the yield (ready to receive)
p.send('Hello')
p.send('World')
p.close()
```

èőÍèőž

actoræÍąåijŔçŽĎé■ĚåŁŻåřśåIJÍäžŐåőČçŽĎçőĂå■ŢæĂğãĂĆ
åőđéŹĚäÿŁïijŇè£ŹéĞŇäżĚäżĚåŔłæIJĽäÿĂäÿłæăÿå£ČæŞ■ä¡IJ send() .
çŤŽèĞşïijŇåŕźäžŐåIJÍå§žäžŐactorçşżçż§äÿ■çŽĎâĂIJæűĹæĄŕâĂİçŽĎæşŻåŇŰæęĆå£ţåŔŕäżěåůšåď Žçğ■æŰ
ä¿ŇåęĆïijŇä¡ăåŔŕäżěäżěåĚČçżĎå¡ćåijŔäijăéĂŠæăĞç■¿æűĹæĄŕïijŇèőľactoræĽ ğèąŇäÿ■åŘŇçŽĎæŞ■ä¡IJïij

```
class TaggedActor(Actor):
    def run(self):
        while True:
            tag, *payload = self.recv()
            getattr(self,'do_'+tag)(*payload)

    # Methods correponding to different message tags
    def do_A(self, x):
        print('Running A', x)

    def do_B(self, x, y):
        print('Running B', x, y)

# Example
a = TaggedActor()
a.start()
a.send(('A', 1))      # Invokes do_A(1)
a.send(('B', 2, 3))   # Invokes do_B(2,3)
```

ä¡IJäÿžåŔęåď ŰäÿĂäÿłä¿Ńå■ŘïijŇäÿŃéİćçŽĎactoråĚĄèőÿåIJÍäÿĂäÿłåůěä¡IJèĂĚäÿ■è£ŘèąŇäżżæĎŔçŽ
åźűäÿŤéĂŽè£ĞäÿĂäÿłçĽźæőŁçŽĎResultåŕźèśąè£ŤåŻđçżŞæđIJïijŽ

```
from threading import Event
class Result:
    def __init__(self):
        self._evt = Event()
        self._result = None

    def set_result(self, value):
        self._result = value

        self._evt.set()

    def result(self):
        self._evt.wait()
        return self._result
```

```
class Worker(Actor):
    def submit(self, func, *args, **kwargs):
        r = Result()
        self.send((func, args, kwargs, r))
        return r

    def run(self):
        while True:
            func, args, kwargs, r = self.recv()
            r.set_result(func(*args, **kwargs))

# Example use
worker = Worker()
worker.start()
r = worker.submit(pow, 2, 3)
print(r.result())
```

æIJĂåŘŐïijÑâĂIJåŔŚéĂĄâĂİäÿĂäÿłäżżåŁąæűĹæĄſçŽĎæęĆå£ţåŕſäżěèćńæL'ľåśŢåĹřåď'Žè£ŻçÍÑçŤŽ
ä¿ÑåęĆïijÑäÿĂäÿłçśżactorårźžèśąçŽĎ send() æŰźæşŢåŕſäżěèćńçijŰçÍŃèőľåőČèČ¡åIJÍäÿĂäÿłåěŮæŐěå■Ů
æĹŰéĂŽè£Ğæ§ŘäžŻæűĹæĄſäÿ■éŮt'äżűïijĹæſŤåęĆAMQPãĂĄZMQç■L'ïijL'æİěåŔŚéĂĄãĂĆ

12.11 åőđçŐřæűĹæĄſåŔŚåÿČ/èőćéŸĚæĺąåđŃ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłå§žäžŐçž£çÍŃéĂŽä£ąçŽĎçÍŃåžŕïijŇæČşèőľåőČäžňåőđçŐřåŔŚåÿČ/èőćéŸĚæĺąåijŔçŽİ

èğčåĘşæŰźæąĹ

èęĄåőđçŐřåŔŚåÿČ/èőćéŸĚçŽĎæűĹæĄſéĂŽä£ąæĺąåijŔïijŇ
ä¡ăéĂŽåÿÿèęĄåijŢåĚěäÿĂäÿłå■ŢçŃňçŽĎâĂIJäžd'æ■ćæIJžâĂİæĹŰâĂIJç¡ŚåĚşâĂİåſźèśąä¡IJäÿžæL'ĂæIJL'æ
äź§åřśæŸſèſt'ïijÑäÿ■çŽt'æŐěåřĘæűĹæĄſäżŐäÿĂäÿłäżżåŁąåŔŚéĂĄåĹřåŔęäÿĂäÿłïijÑèĂŇæŸſåřĘåĚűåŔŚé
çĎűåŘŐçŤśäžd'æ■ćæIJžåřĘåőČåŔŚéĂĄçżŹäÿĂäÿłæĹŰåd'ŽäÿłèćńåĕşèĄŤäżżåŁąãĂĆäÿŃéİćæŸſäÿĂäÿłéİđ

```
from collections import defaultdict

class Exchange:
    def __init__(self):
        self._subscribers = set()

    def attach(self, task):
        self._subscribers.add(task)

    def detach(self, task):
        self._subscribers.remove(task)
```

```
    def send(self, msg):
        for subscriber in self._subscribers:
            subscriber.send(msg)

# Dictionary of all created exchanges
_exchanges = defaultdict(Exchange)

# Return the Exchange instance associated with a given name
def get_exchange(name):
    return _exchanges[name]
```

äÿĂäÿłäžď'æ■ćæIJžåřśæŸŕäÿĂäÿłæŹőéĂŽåŕźèśąïijŇèt'§èt'čçżt'æŁď'äÿĂäÿłæt'żèůČçŽĎèőćéŸĚèĂĚéŹĺ
æŕŔäÿłäžď'æ■ćæIJžéĂŽè£ĞäÿĂäÿłåŘ■çğřåőŽä¡■ïijŇget_exchange()
éĂŽè£ĞçżŹåőŽäÿĂäÿłåŘ■çğřè£ŤåŽđçŽýåžŤçŽĎ Exchange åőđä¿ŃãĂĆ

äÿŃéÍćæŸŕäÿĂäÿłçőĂå■Ţä¿Ńå■ŘïijŇæijŤçď'žäžĘåęĆä¡Ţä¡£çŤĺäÿĂäÿłäžď'æ■ćæIJžïijŽ

```
# Example of a task.  Any object with a send() method

class Task:
    ...
    def send(self, msg):
        ...

task_a = Task()
task_b = Task()

# Example of getting an exchange
exc = get_exchange('name')

# Examples of subscribing tasks to it
exc.attach(task_a)
exc.attach(task_b)

# Example of sending messages
exc.send('msg1')
exc.send('msg2')

# Example of unsubscribing
exc.detach(task_a)
exc.detach(task_b)
```

åř¡çőąåŕźäžŐè£ŹäÿłéŮőécŸæIJL'å¿Ĺåď'ŽçŽĎåŔŸçğ■ïijŇäÿ■è£ĞäÿĞåŔŸäÿ■çężåĚűåőŮãĂĆ
æűĹæĄŕäijŽècńåŔŠéĂĄçżŹäÿĂäÿłäžď'æ■ćæIJžïijŇçĎűåŘŐäžď'æ■ćæIJžäijŽářĘåőČäžňåŔŠéĂĄçżŹècńçżŠå

èőĺèőž

éĂŽè£ĞéŸ§åĹŮåŘŚéĂĄæűĹæĄŕçŽĎäżżåŁąæĹŰçž£çĺŃçŽĎæĺąåijŔå¿Ĺåőźæ
äÿ■è£ĞïijŇä¡£çŤĺåŘŚåÿČ/èőćéŸĔæĺąåijŔçŽĎåě¡åď'ĎæŻť'åŁäæŸŐæŸ¿ãĂĆ

éęŰåĚĹïijŇä¡£çŤĺäÿĂäÿłäžď'æ■ćæIJžåŔŕäżěçőĂåŇŰåď'ğéČĺåĹĘæűĽ'åŘŁåĹřçž£çĺŃéĂŽä£ąçŽĎåůěä¡Ł
æŮăéIJĂåŐżåĘźéĂŽè£Ğåď'Žè£ŻçĺŃæĺąåİŮæİěæŞ■ä¡IJåď'Žäÿłçž£çĺŃïijŇä¡ăåŘłéIJĂèęĄä¡£çŤĺè£Źäÿłäžď'æ
æ§Řçğ■çĺŃåžęäÿŁïijŇè£Źäÿłåřśèů§æŮěå£ŮæĺąåİŮçŽĎåůěä¡IJåŐ§çŘĘçśżäijijãĂĆ
åőđéŹĔäÿŁïijŇåőČåŘŕäżěè¡żæİ¿çŽĎèğčèĂęçĺŃåžŔäÿ■åď'ŽäÿłäżżåŁąãĂĆ

åĔűæňąïijŇäžď'æ■ćæIJžåź£æŠ■æűĹæĄçżŹåď'ŽäÿłèőćéŸĔèĂĔçŽĎèČ¡åŁŻåÿęæİěäžĘäÿĂäÿłåĔĺæŰřçŽ
ä¿ŃåęĆïijŇä¡ăåŘŕäżěä¡£çŤĺåď'ŽäżżåŁąçşżçż§ãĂĄåź£æŠ■æĹŰæĽ'ĞåĞžãĂĆ
ä¡ăè£ŸåŘŕäżěéĂŽè£ĞäżěæŹőéĂŽèőćéŸĔèĂĔèžńäż¡çżŚåőŽæİěædĎåżžèřČèŕŢåŠŇèŕŁæŰ■åůěåĔůãĂĆ
ä¿ŃåęĆïijŇäÿŇéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎèŕŁæŰ■çśžïijŇåŘŕäżěæŸ¿çď'žèćńåŘŚéĂĄçŽĎæűĹæĄŕïijŽ

```
class DisplayMessages:
    def __init__(self):
        self.count = 0
    def send(self, msg):
        self.count += 1
        print('msg[{}]: {!r}'.format(self.count, msg))

exc = get_exchange('name')
d = DisplayMessages()
exc.attach(d)
```

æIJĂåŘŐïijŇèŕěåődçŐřçŽĎäÿĂäÿłéĞ■èęĄçL'źçĆźæŸŕåőČèČ¡åĔijåőźåď'Žäÿłâ
likeâĂİåŕźèşąãĂĆ ä¿ŃåęĆïijŇæűĹæĄŕæŐěåŘŮèĂĔåŘŕäżěæŸŕactorïijĹ12.10åřŔèŁĆäżŃçż■ïijL'ãĂĄå■ŘçĺŃ
`send()` æŰźæşŢçŽĎäÿIJèě£ãĂĆ

åĔşäžŐäžď'æ■ćæIJžçŽĎäÿĂäÿłåŘŕèČ¡éŮőécŸæŸŕåŕźäžŐèőćéŸĔèĂĔçŽĎæ■čçąőçżŚåőŽåŠŇèğčçżŚãĂ
äÿžäžĘæ■čçąőçŽĎçőąçŘĘęţĎæžŘïijŇæŕŘäÿĂäÿłçżŚåőŽçŽĎèőćéŸĔèĂĔå£ĔéążæIJĂçźĹèęĄèğčçżŚãĂĆ
åIJĺäżčçăĄäÿ■éĂŽåÿÿäijŽæŸŕåČŘäÿŃéİćè£ŹæăůçŽĎæĺąåijŔïijŽ

```
exc = get_exchange('name')
exc.attach(some_task)
try:
    ...
finally:
    exc.detach(some_task)
```

æ§Řçğ■æĎŘäźL'äÿŁïijŇè£ŹäÿłåŠŇä¡£çŤĺæŰĞäżűãĂĄéŤĄåŠŇçśžäijijåŕźèśąå¿ĹåČŘãĂĆ
éĂŽåÿÿå¿Ĺåőźæ ŸŞäijŽå£ŸèőřæIJĂåŘŐçŽĎ `detach()` æ■ěéłd'ãĂĆ
äÿžäžĘçőĂåŇŰè£ŹäÿłïijŇä¡ăåŘŕäżěèĂČèŹŠä¡£çŤĺäÿŁäÿŃæŰĞçőąçŘĘåŹĺå■ŔèőőãĂĆ
ä¿ŃåęĆïijŇåIJĺäžď'æ■ćæIJžåŕźèśąäÿŁåćđåŁääÿĂäÿł `subscribe()`
æŰźæşŢïijŇåęĆäÿŃïijŽ

```
from contextlib import contextmanager
from collections import defaultdict

class Exchange:
```

```
    def __init__(self):
        self._subscribers = set()

    def attach(self, task):
        self._subscribers.add(task)

    def detach(self, task):
        self._subscribers.remove(task)

    @contextmanager
    def subscribe(self, *tasks):
        for task in tasks:
            self.attach(task)
        try:
            yield
        finally:
            for task in tasks:
                self.detach(task)

    def send(self, msg):
        for subscriber in self._subscribers:
            subscriber.send(msg)

# Dictionary of all created exchanges
_exchanges = defaultdict(Exchange)

# Return the Exchange instance associated with a given name
def get_exchange(name):
    return _exchanges[name]

# Example of using the subscribe() method
exc = get_exchange('name')
with exc.subscribe(task_a, task_b):
     ...
     exc.send('msg1')
     exc.send('msg2')
     ...

# task_a and task_b detached here
```

æIJĂåŘŐè£ŸåžŤèŕěæşĺæĎŔçŽĎæŸŕåĚşäžŐäžď'æ■ćæIJžçŽĎæĂİæČşæIJL'å¿Ĺåď'ŽçğçŽĎæL'l'åśŢåőđ
ä¿ŃåęĆïijŇäžď'æ■ćæIJžåŘŕäžěåőđçŐřäÿĂæŢt'äÿłæűĹæĄŕéĂŽéĄŞéŻĘåŘĹæĹŰæŔŘä¿Żäžď'æ■ćæIJžåŘçğ
äžď'æ■ćæIJžè£ŸåŔŕäžěèćńæL'l'åśŢåĹřåĹĘåÿČåijŔèőąçőŮçÍŃåžŔäÿ■ïijĹæŕŤåęĆïijŇářĘæűĹæĄŕèůŕçŤśåĹřä

12.12 ä¡£çŤĺçŤ§æĹŘåŹĺäžčæŻ£çž£çĺŃ

éŮőécŸ

ä¡ăæČşä¡£çŤĺçŤ§æĹŘåŹĺïijĹå■ŔçĺŃïijĽæŻ£äžčçşżçż§çž£çĺŃæİěåőđçŐřåźűåŔŚãĂĆè£ŹäÿłæIJĽæŮűåŀ

èğčåĘşæŰźæąĹ

èęĄä¡£çŤĺçŤ§æĹŘåŹĺåőđçŐřèĞłåůśçŽĎåźűåŔŚïijŃä¡ăéęŰåĚĹèęĄåŕźçŤ§æĹŘåŹĺåĞ¡æŢřåŠŇ `yield` èŕ■åŔěæIJĽæůśåĹżçŘĘèğčãĂĆ `yield` èŕ■åŔěäijŽèőľäÿĂäÿłçŤ§æĹŘåŹĺæŇĆèţůåőČçŽĎæĽğèąŇïi åřĘçŤ§æĹŘåŹĺå¡ŞåĄŽæ§Řçğ■âĂIJäžżåŁąâĂİåźűä¡£çŤĺäžżåŁąå■Ŕä¡IJåĹĞæ■ćæİěæŻ£æ■ćåőČäžňçŽĎæĽğ èęĄæijŤçđ'žè£Źçğ■æĂİæČşïijŇèĂČèŹŚäÿŃéİćäÿđ'äÿłä¡£çŤĺçőĂå■ŢçŽĎ `yield` èŕ■åŔěçŽĎçŤ§æĹŘåŹĺåĞ¡æŢřïijŽ

```
# Two simple generator functions
def countdown(n):
    while n > 0:
        print('T-minus', n)
        yield
        n -= 1
    print('Blastoff!')

def countup(n):
    x = 0
    while x < n:
        print('Counting up', x)
        yield
        x += 1
```

è£ŹäžŻåĞ¡æŢřåIJĺåĘĚéČĺä¡£çŤĺyieldèŕ■åŔěïijŃäÿŃéİćæŸŕäÿĂäÿłåőđçŐřäžEçőĂå■ŢäżżåŁąèřČåžęåŹĺ

```
from collections import deque

class TaskScheduler:
    def __init__(self):
        self._task_queue = deque()

    def new_task(self, task):
        '''
        Admit a newly started task to the scheduler

        '''
        self._task_queue.append(task)

    def run(self):
        '''
        Run until there are no more tasks
        '''
```

```
        while self._task_queue:
            task = self._task_queue.popleft()
            try:
                # Run until the next yield statement
                next(task)
                self._task_queue.append(task)
            except StopIteration:
                # Generator is no longer executing
                pass

# Example use
sched = TaskScheduler()
sched.new_task(countdown(10))
sched.new_task(countdown(5))
sched.new_task(countup(15))
sched.run()
```

TaskScheduler çśžåIJÍäÿĂäÿłå¿łçŐŕäÿ■è£ŘèąŇçŤ§æĹŘåŹĺéŻĘåŘĹâĂŤâĂŤæŕŔäÿłéČ¡è£ŘèąŇåĹřç
è£ŘèąŇè£Źäÿłä¿Ńå■ŘïijŇè¿ŞåĞžåęĆäÿŃïijŽ

```
T-minus 10
T-minus 5
Counting up 0
T-minus 9
T-minus 4
Counting up 1
T-minus 8
T-minus 3
Counting up 2
T-minus 7
T-minus 2
...
```

åĹřæ■d'äÿžæ■ćïijŇæĹŚäżňåðđéŹĚäÿŁåůšçżŔåðđçŐřäžĘäÿĂäÿłâĂIJæŞ■ä¡IJçşżçż§âĂİçŽĎæIJĂářŔæă
çŤ§æĹŘåŹĺåĞ¡æŢřářśæŸŕèðd'äÿžïijŇèĂŇyieldèŕ■åŘěæŸŕäżżåŁąæŇĆèţůçŽĎä£ąåŔůãĂĆ
èřČåžęåŹĺå¿łçŐŕæčÄæ§ěäżżåŁąåĹŮèąĺçŻť'åĹřæšąæIJL'äżżåŁąèęĄæL'ğèąŇäÿžæ■ćãĂĆ

åðđéŹĚäÿŁïijŇä¡ăåŔŕèČ¡æČşèęĄä¡£çŤĺçŤ§æĹŘåŹĺæİěåðđçŐřçőĂå■ŢçŽĎåźűåŔŚãĂĆ
éĆčäźĹïijŇåIJĺåðđçŐřactoræĹŰç¡ŚçżIJæIJ■åŁąåŹĺçŽĎæŮűåĂŹä¡ăåŔŕäžěä¡£çŤĺçŤ§æĹŘåŹĺæİěæŻ£äżčçž£ç

äÿŃéİćçŽĎäżčçăĄæijŤçd'žäžĘä¡£çŤĺçŤ§æĹŘåŹĺæİěåðđçŐřäÿĂäÿłäÿ■ä¿İèţŰçž£çĺŃçŽĎactorïijŽ

```
from collections import deque

class ActorScheduler:
    def __init__(self):
        self._actors = { }          # Mapping of names to actors
        self._msg_queue = deque()   # Message queue

    def new_actor(self, name, actor):
        '''
```

```
        Admit a newly started actor to the scheduler and give it a␣
↪name
        '''
        self._msg_queue.append((actor,None))
        self._actors[name] = actor

    def send(self, name, msg):
        '''
        Send a message to a named actor
        '''
        actor = self._actors.get(name)
        if actor:
            self._msg_queue.append((actor,msg))

    def run(self):
        '''
        Run as long as there are pending messages.
        '''
        while self._msg_queue:
            actor, msg = self._msg_queue.popleft()
            try:
                 actor.send(msg)
            except StopIteration:
                 pass

# Example use
if __name__ == '__main__':
    def printer():
        while True:
            msg = yield
            print('Got:', msg)

    def counter(sched):
        while True:
            # Receive the current count
            n = yield
            if n == 0:
                break
            # Send to the printer task
            sched.send('printer', n)
            # Send the next count to the counter task (recursive)

            sched.send('counter', n-1)

    sched = ActorScheduler()
    # Create the initial actors
    sched.new_actor('printer', printer())
    sched.new_actor('counter', counter(sched))

    # Send an initial message to the counter to initiate
```

```
    sched.send('counter', 10000)
    sched.run()
```

åőŇåĚĺåijĎæĞĆè£ŹæőţäżčçăĄéIJĂèęĄæŻť'æůśåĚěçŽĎå■ęäźäïijŇä¡ĘæŸŕåĚşéŤőçĆźåIJĺäžŐæŤűéŻĘæ
æIJňèť'ĺäÿŁïijŇèřČåžęåŹĺåIJĺæIJL'éIJĂèęĄåŘŠéĂĄçŽĎæűĹæĄŕæŮűäijŽäÿĂçŻť'è£ŘèąŇçİĂãĂĆ
èőąæŢřçŤ§æĹŘåŹĺäijŽçżŹèĞłåůśåŘŠéĂĄæűĹæĄŕåźűåIJĺäÿĂäÿłéĂŠå¡Šå¿łçŐŕäÿ■çżŞæİ§ãĂĆ

äÿŃéİćæŸŕäÿĂäÿłæŻť'åŁăéńŸçžğçŽĎä¿Ńå■ŘïijŇæijŤçď'žäžĘä¡£çŤĺçŤ§æĹŘåŹĺæİěåőđçŐŕäÿĂäÿłåźűå

```
from collections import deque
from select import select

# This class represents a generic yield event in the scheduler
class YieldEvent:
    def handle_yield(self, sched, task):
        pass
    def handle_resume(self, sched, task):
        pass

# Task Scheduler
class Scheduler:
    def __init__(self):
        self._numtasks = 0        # Total num of tasks
        self._ready = deque()     # Tasks ready to run
        self._read_waiting = {}   # Tasks waiting to read
        self._write_waiting = {} # Tasks waiting to write

    # Poll for I/O events and restart waiting tasks
    def _iopoll(self):
        rset,wset,eset = select(self._read_waiting,
                                self._write_waiting,[])
        for r in rset:
            evt, task = self._read_waiting.pop(r)
            evt.handle_resume(self, task)
        for w in wset:
            evt, task = self._write_waiting.pop(w)
            evt.handle_resume(self, task)

    def new(self,task):
        '''
        Add a newly started task to the scheduler
        '''

        self._ready.append((task, None))
        self._numtasks += 1

    def add_ready(self, task, msg=None):
        '''
        Append an already started task to the ready queue.
        msg is what to send into the task when it resumes.
        '''
```

```
        self._ready.append((task, msg))

    # Add a task to the reading set
    def _read_wait(self, fileno, evt, task):
        self._read_waiting[fileno] = (evt, task)

    # Add a task to the write set
    def _write_wait(self, fileno, evt, task):
        self._write_waiting[fileno] = (evt, task)

    def run(self):
        '''
        Run the task scheduler until there are no tasks
        '''
        while self._numtasks:
            if not self._ready:
                self._iopoll()
            task, msg = self._ready.popleft()
            try:
                # Run the coroutine to the next yield
                r = task.send(msg)
                if isinstance(r, YieldEvent):
                    r.handle_yield(self, task)
                else:
                    raise RuntimeError('unrecognized yield event')
            except StopIteration:
                self._numtasks -= 1

# Example implementation of coroutine-based socket I/O
class ReadSocket(YieldEvent):
    def __init__(self, sock, nbytes):
        self.sock = sock
        self.nbytes = nbytes
    def handle_yield(self, sched, task):
        sched._read_wait(self.sock.fileno(), self, task)
    def handle_resume(self, sched, task):
        data = self.sock.recv(self.nbytes)
        sched.add_ready(task, data)

class WriteSocket(YieldEvent):
    def __init__(self, sock, data):
        self.sock = sock
        self.data = data
    def handle_yield(self, sched, task):

        sched._write_wait(self.sock.fileno(), self, task)
    def handle_resume(self, sched, task):
        nsent = self.sock.send(self.data)
        sched.add_ready(task, nsent)
```

```
class AcceptSocket(YieldEvent):
    def __init__(self, sock):
        self.sock = sock
    def handle_yield(self, sched, task):
        sched._read_wait(self.sock.fileno(), self, task)
    def handle_resume(self, sched, task):
        r = self.sock.accept()
        sched.add_ready(task, r)

# Wrapper around a socket object for use with yield
class Socket(object):
    def __init__(self, sock):
        self._sock = sock
    def recv(self, maxbytes):
        return ReadSocket(self._sock, maxbytes)
    def send(self, data):
        return WriteSocket(self._sock, data)
    def accept(self):
        return AcceptSocket(self._sock)
    def __getattr__(self, name):
        return getattr(self._sock, name)

if __name__ == '__main__':
    from socket import socket, AF_INET, SOCK_STREAM
    import time

    # Example of a function involving generators.  This should
    # be called using line = yield from readline(sock)
    def readline(sock):
        chars = []
        while True:
            c = yield sock.recv(1)
            if not c:
                break
            chars.append(c)
            if c == b'\n':
                break
        return b''.join(chars)

    # Echo server using generators
    class EchoServer:
        def __init__(self,addr,sched):
            self.sched = sched
            sched.new(self.server_loop(addr))

        def server_loop(self,addr):
            s = Socket(socket(AF_INET,SOCK_STREAM))

            s.bind(addr)
            s.listen(5)
```

```
            while True:
                c,a = yield s.accept()
                print('Got connection from ', a)
                self.sched.new(self.client_handler(Socket(c)))

        def client_handler(self,client):
            while True:
                line = yield from readline(client)
                if not line:
                    break
                line = b'GOT:' + line
                while line:
                    nsent = yield client.send(line)
                    line = line[nsent:]
            client.close()
            print('Client closed')

    sched = Scheduler()
    EchoServer(('',16000),sched)
    sched.run()
```

è£Źæőţäžčçă Ąæ IJL'çĆžåď■æİĆãĂĆäÿ■è£ĞïijŇåőČåőđçŐřäžĘäÿĂäÿłářŔåđŃçŽĎæŞ■ä¡IJçşžçż§ãĂĆ æIJL'äÿĂäÿłářšçżłçŽĎäżżåŁąéŸ§åĹŮïijŇåźűäÿŤè£ŸæIJL'åŻăI/OäijŚçIJăçŽĎäżżåŁąç■L'å¿ĚåŇžå§§ãĂĆ è£ŸæIJL'å¿Ĺåď'ŽèřČåžęåŹĺeť§èť čåIJĺářšçżłéŸ§åĹŮåŠŇI/Oç■L'å¿ĚåŇžå§§äźŃéŮť çğżåŁĺäżżåŁąãĂĆ

èőĺèőž

åIJĺæđĎåżžå§žäžŐçŤ§æĹŘåŹĺçŽĎåźűåŔŚæąĘæđűæŮűïijŇéĂŽåÿÿäijŽä¡£çŤĺæŻť åÿÿèğAçŽĎyieldå¡ćå

```
def some_generator():
    ...
    result = yield data
    ...
```

ä¡£çŤĺè£Źçğ■å¡ćåijŔçŽĎyieldèí■åŘěçŽĎåĞ¡æŢřéĂŽåÿÿèćńçğřäÿžâĂIJå■ŔçĺŃâĂİãĂĆ éĂŽè£ĞèřČåžęåŹĺïijŇyieldèí■åŘěåIJĺäÿĂäÿłå¿łçŐřäÿ■èćńåď'ĎçŘĘïijŇåęĆäÿŃïijŽ

```
f = some_generator()

# Initial result. Is None to start since nothing has been computed
result = None
while True:
    try:
        data = f.send(result)
        result = ... do some calculation ...
    except StopIteration:
        break
```

è£ŹéĞŇçŽĎéĂżè¿ŞçĺĺåżőæIJL'çĆžåď'■æİĆãĂĆäÿ■è£ĞïijŇèćńäijăçżŹ

send() çŽĎåĂijåőŽäźĽäžĘåIJÍyieldèŕ■åŔěéĘŠæĺěæŮűçŽĎè£ŤåŻđåĂijãĂĆ
åŻäæ■d'ïijŇåęĆæđIJäÿĂäÿłyieldåĞĘåd'ĞåIJÍåŕźäźŃåĽ■yield-
æŢřæ■őçŽĎåŻđåžŤäÿ■è£ŤåŻđçżŞæđIJæŮűïijŇäijŽåIJÍäÿŃäÿĂæňą send()
æŞ■ä¡IJè£ŤåŻđãĂĆ åęĆæđIJäÿĂäÿłçŤ§æĹŔåŹĺåĞ¡æŢřåĽŽåijĂåğŃè£ŘèąŇïijŇåŔŚéĂĄäÿĂäÿłNoneåĂijäij

éŹd'äžĘåŔŚéĂĄåĂijåd'ŰïijŇè£ŸåŔŕäžěåIJÍäÿĂäÿłçŤ§æĹŔåŹĺäÿŁéĺćæĽ'ğèąŇäÿĂäÿł
close() æŰźæşŢãĂĆ åőČäijŽåŕijèĞť'åIJÍæĽ'ğèąŇyieldèŕ■åŔěæŮűæŁŻåĞžäÿĂäÿł
GeneratorExit åijĆåÿÿïijŇäżŐèĂŇçżĹæ■ćæĽ'ğèąŇãĂĆ
åęĆæđIJè£ŻäÿĂæ■ěèő¿èőąïijŇäÿĂäÿłçŤ§æĹŔåŹĺåŔŕäžěæ■ŢèŐůè£ŹäÿłåijĆåÿÿåźűæĽ'ğèąŇæÿĚçŘĘæŞ■ä¡I
åŘŇæåůè£ŸåŔŕäžěä¡£çŤĺçŤ§æĹŔåŹĺçŽĎ throw() æŰźæşŢåIJÍyield-
èŕ■åŔěæĽ'ğèąŇæŮűçŤ§æĹŘäÿĂäÿłäżżæĎŘçŽĎæĽ'ğèąŇæŇĞäžd'ãĂĆ
äÿĂäÿłäżżåŁąèřČåžęåŹĺåŔŕåĽ'çŤĺåőČæĺěåIJÍè£ŘèąŇçŽĎçŤ§æĹŔåŹĺäÿ■åd'ĎçŘĘéŤŹèŕŕãĂĆ

æIJĂåŘŐäÿĂäÿłä¿Ńå■Řäÿ■ä¡£çŤĺçŽĎ yield from èŕ■åŔèćńçŤĺæĺěåőđçŐřå■ŔçĺŇïijŇåŔŕäžěèćńåĚ
æIJňèťĺäÿŁåřśæŸŕåŗĘæŐğåĹűæİČéĂŔæŸŐçŽĎäijàè¿ŞçżŹæŰřçŽĎåĞ¡æŢřãĂĆ
äÿ■åČŔæŹőéĂŽçŽĎçŤ§æĹŘåŹĺïijŇäÿĂäÿłä¡£çŤĺ yield from
èćńèřČçŤĺçŽĎåĞ¡æŢřåŔŕäžěè£ŤåŻđäÿĂäÿłä¡IJäÿž yield from
èŕ■åŔężŞæđIJçŽĎåĂijãĂĆ åĚşäžŐ yield from çŽĎæŻťåd'Žä£ąæĄŕåŔŕäžěåIJÍ PEP
380 äÿ■æĽ¿åĹřãĂĆ

æIJĂåŘŐïijŇåęĆæđIJä¡£çŤĺçŤ§æĹŘåŹĺçijŰçĺŃïijŇèęĄæŔŘéĘŠä¡ăçŽĎæŸŕåőČè£ŸæŸŕæIJL'å¿Ĺåd'Žç
çĽźåĹńæŸŕïijŇä¡ăå¿Ůäÿ■åĹřäžżä¡Ţçž£çĺŃåŔŕäžěæŔŘä¿ŻçŽĎåě¡åd'ĐãĂĆä¿ŃåęĆïijŇåęĆæđIJä¡ăæĽ'ğèąŇ
åőČäijŽåřĘæŢt'äÿłäżżåŁąæŇĆètůç§ěéĄŞæŞ■ä¡IJåőŇæĹŘãĂĆäÿžäžĘèğčåĘşè£ŹäÿłéŮőécŸïijŇ
ä¡ăåŔłèČ¡éĂĽæŇl'åřĘæŞ■ä¡IJåğŤæt'¿çżŹåŖęåd'ŰäÿĂäÿłåŔŕäžěçŇñçńŃè£ŘèąŇçŽĎçž£çĺŃæĹŰè£ŻçĺŃãĂ
åŖęåd'ŰäÿĂäÿłéŹŘåĹűæŸŕåd'ğéČĺåĹĘPythonåžŞåźűäÿ■èČ¡å¿Ĺåě¡çŽĎåĚijåőźå§žäžŐçŤ§æĹŘåŹĺçŽĎçž£çĺ
åęĆæđIJä¡ăéĂĽæŇl'è£ŹäÿłæŰźæąĹïijŇä¡ăäijŽåŔŚçŐŕä¡ăéIJĂèęĄèĞłåůśæŤźåĘŹå¿Ĺåd'ŽæăĞåĞĘåžŞåĞ¡æ
ä¡IJäÿžæIJňèŁĆæŘŘåĽřçŽĎå■ŔçĺŃåŠŃçŻÿåĚşæŁĂæIJŕçŽĎäÿĂäÿłå§žçąĂèČŃæŹřïijŇåŔŕäžěæ§ěçIJŃ
PEP 342 åŠŃ âĂIJå■ŔçĺŃåŠŃåźűåŔŚçŽĎäÿĂéŮĺæIJL'èűčèŕ¿çĺŃâĂİ

PEP 3156 åŘŇæåůæIJL'äÿĂäÿłåĚşäžŐä¡£çŤĺå■ŔçĺŃçŽĎåijĆæ■ěI/OæĺąåđŃãĂĆ
çĽźåĹńçŽĎïijŇä¡ăäÿ■åŔŕèČ¡èĞłåůśåŐžåőđçŐřäÿĂäÿłåžŢåśĆçŽĎå■ŔçĺŃèřČåžęåŹĺãĂĆ
äÿ■è£ĞïijŇåĚşäžŐå■ŔçĺŃçŽĎæĂĺæČşæŸŕå¿Ĺåd'ŽæţĄęąŃåžŞçŽĎå§žçąĂïijŇ åŇĚæŃň
gevent, greenlet, Stackless Python äžěåŔŁåĚűäžŰçśžäijijåůěçĺŃãĂĆ

12.13 åd'Žäÿłçž£çĺŃéŸ§åĹŮè¡őèŕć

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłçž£çĺŃéŸ§åĹŮéŻĘåŘĹïijŇæČşäÿžåĹřæĺěçŽĎåĚČçť'ăè¡őèŕćåőČäžñïijŇ
åŕśèů§ä¡ăäÿžäÿĂäÿłåőćæĹůçńŕèřůæśĆåŐżè¡őèŕćäÿĂäÿłç¡ŞçżIJè£đæŐěéŻĘåŘĹçŽĎæŰźåijŔäÿĂæăůãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐè¡őèŕćéŮőécŸçŽĎäÿĂäÿłåÿÿèğĄèğčåĘşæŰźæąĹäÿ■æIJL'äÿłå¿ĹåřŚæIJL'äžžç§ěéĄŞçŽĎæŁĂåů
æIJňèťĺäÿŁèőšåĚűæĂİæČşåřśæŸŕïijŽåŕźäžŐæŕŔäÿłä¡ăæČşèęĄè¡őèŕćçŽĎéŸ§åĹŮïijŇä¡ăåĹŻåžžäÿĂåŕźè£đa
çĎűåŘŐä¡ăåIJÍåĚűäÿ■äÿĂäÿłåěŮæŐěå■ŮäÿŁéĺćçijŮåĘŹäžčçĂĄæİěæăĞèŕĘå■ŸåIJÍçŽĎæŢřæ■őïijŇ
åŖęåd'ŰäÿĂäÿłåěŮæŐěå■Ůèćńäijăçžź select() æĹŰçśžäijijçŽĎäÿĂäÿłè¡őèŕćæŢřæ■őåĹřè¿¿çŽĎåĞ¡æŢř

```
import queue
import socket
import os

class PollableQueue(queue.Queue):
    def __init__(self):
        super().__init__()
        # Create a pair of connected sockets
        if os.name == 'posix':
            self._putsocket, self._getsocket = socket.socketpair()
        else:
            # Compatibility on non-POSIX systems
            server = socket.socket(socket.AF_INET, socket.SOCK_
↪STREAM)
            server.bind(('127.0.0.1', 0))
            server.listen(1)
            self._putsocket = socket.socket(socket.AF_INET, socket.
↪SOCK_STREAM)
            self._putsocket.connect(server.getsockname())
            self._getsocket, _ = server.accept()
            server.close()

    def fileno(self):
        return self._getsocket.fileno()

    def put(self, item):
        super().put(item)
        self._putsocket.send(b'x')

    def get(self):
        self._getsocket.recv(1)
        return super().get()
```

åIJÍè£Źäÿłäžčçă Ąäÿ■ïijŇäÿĂäÿłæŰřçŽĎ `Queue` åőđä¿Ńçśžåđ Ńèćńåő Žäź L'ïijŇåžŢåśĆæŸŕäÿĂäÿłèćńè£
åIJÍUnixæIJžåŹĺäÿŁçŽĎ `socketpair()` åĞ¡æŢřèČ¡è¡żæİ¿çŽĎåĹŽåžžè£Źæăůç ŽĎåěÛæŐěå■ŮãĂĆ
åIJÍWindowsäÿŁéİćïijŇä¡åå£Ěéążä¡£çŤĺçśžäijijäžčçă ĄæİěæĺąæŃ§åőČãĂĆ
çĎűåŘŐåőŽäź L'æŹőéĂŽçŽĎ `get()` åŠŇ `put()` æŰźæşŢåIJÍè£ŹäžŽåěÛæŐěå■Ůäÿ ŁéİćæİěæL'ğèąŇI/OæŞ
`put()` æŰźæşŢåE■åřĘæŢřæ■őæŤ¿åĚéé Ÿ§åĹŮåŘŐäijŽåĘŹäÿĂäÿłå■Ţå■ÛèŁĆåĹřæ§Řäÿłåě ÛæŐěå■Ůäÿ■å
èĂŇ `get()` æŰźæşŢåIJÍäžŐéŸ§åĹŮäÿ■çğżéŹd'äÿĂäÿłåĚČçť'ăæŮűäijŽäžŐåŕęåd'ŰäÿĂäÿłåěÛæŐěå■Ůäÿ■

`fileno()` æŰźæşŢä¡£çŤĺäÿĂäÿłåĞ¡æŢřæŕŤåęĆ `select()` æİèèőľè£ŹäÿłéŸ§åĹŮåŕŕäžèèćńè¡őèŕćãĂĆ åőČäżĚäžĚåŘłæŸŕæŽť'éIJšäžĘåžŢåśĆèćń `get()` åĞ¡æŢřä¡£çŤĺåĹřçŽĎsocketçŽĎæŰĞäžűæŔŔè£řçňęèĂŇåůšãĂĆ

äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijŇåőŽäźL'äžĘäÿĂäÿłäÿžåĹřæİěçŽĎåĚČçť'ăçŽŚæŐĝåd'ŽäÿłéŸ§åĹŮçŽĎæű

```
import select
import threading

def consumer(queues):
    '''
```

```
    Consumer that reads data on multiple queues simultaneously
    '''
    while True:
        can_read, _, _ = select.select(queues,[],[])
        for r in can_read:
            item = r.get()
            print('Got:', item)

q1 = PollableQueue()
q2 = PollableQueue()
q3 = PollableQueue()
t = threading.Thread(target=consumer, args=([q1,q2,q3],))
t.daemon = True
t.start()

# Feed data to the queues
q1.put(1)
q2.put(10)
q3.put('hello')
q2.put(15)
...
```

åęĆædIJä¡ăèŕŢçİĂè£ŘèąŇåőČïijŇä¡ăäijŽåŔŚçŐřè£ŹäÿłæűĹèť źèĂĚäijŽæŐěåŔŮåĹřæĽĂæIJĽçŽĎèćń

èőĺèőž

åŕźäžŐè¡őèŕćéİđçśżæŰĞäżűåŕźèśąïijŇæŕŤåęĆéŸ§åĹŮéĂŽåÿÿéČ¡æŸŕæŕŤè¿ČæčŸæĽŃçŽĎéŮőécŸãĂ
ä¿ŇåęĆïijŇåęĆædIJä¡ăäÿ■ä¡£çŤĺäÿŁéİćçŽĎåěŮæŐěå■ŮæŁĂæIJŕïijŇ
ä¡ăåŤŕäÿĂçŽĎéĂĽæŃľåŕśæŸŕçijŰåĘŹäžčçăĄæİěå¿łçŐŕéĄ■åŐĘè£ŹäžŻéŸ§åĹŮåźűä¡£çŤĺäÿĂäÿłåőŽæŮűå

```
import time
def consumer(queues):
    while True:
        for q in queues:
            if not q.empty():
                item = q.get()
                print('Got:', item)

        # Sleep briefly to avoid 100% CPU
        time.sleep(0.01)
```

è£ŹæăůåĄŽåĚűåőđäÿ■åŘĹçŘĘïijŇè£ŸäijŽåijŢåĚěåĚűäžŰçŽĎæĂğèČ¡éŮőécŸãĂĆ
ä¿ŇåęĆïijŇåęĆædIJæŰŕçŽĎæŢŕæ■őècńåŁăåĚěåĹŕäÿĂäÿłéŸ§åĹŮäÿ■ïijŇèĞşåřŚèęĄèŁś10æŕńçğŠæĽ■èČ¡è
åęĆædIJä¡ăäźŇåĽ■çŽĎè¡őèŕćè£ŸèęĄåŐżè¡őèŕćåĚűäžŰåŕźèśąïijŇæŕŤåęĆç¡ŚçżIJåěŮæŐěå■ŮéĆčè£ŸäijŽæ
ä¿ŇåęĆïijŇåęĆædIJä¡ăæČşåŘŇæŮűè¡őèŕćåěŮæŐěå■ŮåŠŇéŸ§åĹŮïijŇä¡ăåŔŕèČ¡èęĄåČŔäÿŇéİćè£ŹæăůäĄ£

```
import select

def event_loop(sockets, queues):
```

```
    while True:
        # polling with a timeout
        can_read, _, _ = select.select(sockets, [], [], 0.01)
        for r in can_read:
            handle_read(r)
        for q in queues:
            if not q.empty():
                item = q.get()
                print('Got:', item)
```

è£Źäÿ łæŰźæąĹéĂŽè£ĞåřĘéŸ§åĹŮåŠŇåěŮæŐěå■Ůç■L'åŘŇåŕźå¿ĚæİêèğčåĘşäžĘåd'ğéČĺåĹĘçŽĎéŮőe
äÿĂäÿłå■ŢçŃňçŽĎ select() èřČçŤĺåŔŕèćńåŘŇæŮűçŤĺæİêè¡őèŕćãĂĆ
ä¡£çŤĺèűĚæŮűæĹŮåĚűäžŰå§žäžŐœŮűéŮt'çŽĎæIJžåĹűæİěæL'ğèąŇåŚĺæIJ§æĂğæčÄæ§ěåźűæšąæIJL'å£Ěè
çŤŽèĞşïijŇåęĆæđIJæŢřæ■őèćńåŁääĚěåĹřäÿĂäÿłéŸ§åĹŮïijŇæűĹèt'źèĂĚåĞääźŐåŔŕäžěåðđæŮűçŽĎèćńéĂ
år¡çőąäijŽæIJL'äÿĂçĆźçĆźåžŢåśĆçŽĎI/Oæ■§èĂŮïijŇä¡£çŤĺåőČéĂŽåÿÿäijŽèŐůå¿ŮæŻt'åě¡çŽĎåŞ■åžŤæŮ

12.14 åIJĺUnixçşžçż§äÿŁéİćåŘŕåŁĺåőĹæŁd'è£ŽçĺŃ

éŮőécŸ

ä¡ăæČşçijŰåĘŹäÿĂäÿłä¡IJäÿžäÿĂäÿłåIJĺUnixæĹŰçśżUnixçşžçż§äÿŁéİcè£ŘèąŇçŽĎåőĹæŁd'è£ŽçĺŃè£Ł

èğčåĘşæŰźæąĹ

åĹŽåžžäÿĂäÿłæ■čçąőçŽĎåőĹæŁd'è£ŽçĺŃéIJĂèęĄäÿĂäÿłçš¿çąőçŽĎçşžçż§èřČçŤĺåžŔåĹŮäžěåŔŁåŕźäž
äÿŃéİćçŽĎäžčçăĄåśŢçd'žäžĘæĂŐæăůåőŽäźL'äÿĂäÿłåőĹæŁd'è£ŽçĺŃïijŇåŔŕäžěåŘŕåŁĺåŘŐå¿Ĺåőźæ ŸŞçŽĎ

```
#!/usr/bin/env python3
# daemon.py

import os
import sys

import atexit
import signal

def daemonize(pidfile, *, stdin='/dev/null',
                          stdout='/dev/null',
                          stderr='/dev/null'):

    if os.path.exists(pidfile):
        raise RuntimeError('Already running')

    # First fork (detaches from parent)
    try:
        if os.fork() > 0:
```

```
            raise SystemExit(0)    # Parent exit
    except OSError as e:
        raise RuntimeError('fork #1 failed.')

    os.chdir('/')
    os.umask(0)
    os.setsid()
    # Second fork (relinquish session leadership)
    try:
        if os.fork() > 0:
            raise SystemExit(0)
    except OSError as e:
        raise RuntimeError('fork #2 failed.')

    # Flush I/O buffers
    sys.stdout.flush()
    sys.stderr.flush()

    # Replace file descriptors for stdin, stdout, and stderr
    with open(stdin, 'rb', 0) as f:
        os.dup2(f.fileno(), sys.stdin.fileno())
    with open(stdout, 'ab', 0) as f:
        os.dup2(f.fileno(), sys.stdout.fileno())
    with open(stderr, 'ab', 0) as f:
        os.dup2(f.fileno(), sys.stderr.fileno())

    # Write the PID file
    with open(pidfile,'w') as f:
        print(os.getpid(),file=f)

    # Arrange to have the PID file removed on exit/signal
    atexit.register(lambda: os.remove(pidfile))

    # Signal handler for termination (required)
    def sigterm_handler(signo, frame):
        raise SystemExit(1)

    signal.signal(signal.SIGTERM, sigterm_handler)

def main():
    import time
    sys.stdout.write('Daemon started with pid {}\n'.format(os.
↪getpid()))
    while True:
        sys.stdout.write('Daemon Alive! {}\n'.format(time.ctime()))
        time.sleep(10)

if __name__ == '__main__':
    PIDFILE = '/tmp/daemon.pid'
```

```
    if len(sys.argv) != 2:
        print('Usage: {} [start|stop]'.format(sys.argv[0]),␣
↪file=sys.stderr)
        raise SystemExit(1)

    if sys.argv[1] == 'start':
        try:
            daemonize(PIDFILE,
                      stdout='/tmp/daemon.log',
                      stderr='/tmp/dameon.log')
        except RuntimeError as e:
            print(e, file=sys.stderr)
            raise SystemExit(1)

        main()

    elif sys.argv[1] == 'stop':
        if os.path.exists(PIDFILE):
            with open(PIDFILE) as f:
                os.kill(int(f.read()), signal.SIGTERM)
        else:
            print('Not running', file=sys.stderr)
            raise SystemExit(1)

    else:
        print('Unknown command {!r}'.format(sys.argv[1]), file=sys.
↪stderr)
        raise SystemExit(1)
```

èęĄåŘŕåŁĺè£ŹäÿłåőĹæŁď'è£ŻçĺŃïijŇçŤĺæĹůéIJĂèęĄä¡£çŤĺåęĆäÿŃçŽĎåŠ¡äžď'ïijŽ

```
bash % daemon.py start
bash % cat /tmp/daemon.pid
2882
bash % tail -f /tmp/daemon.log
Daemon started with pid 2882
Daemon Alive! Fri Oct 12 13:45:37 2012
Daemon Alive! Fri Oct 12 13:45:47 2012
...
```

åőĹæŁď'è£ŻçĺŃåŔŕäžěåőŇåĚĺåIJĺåŘŐåŔřè£ŘèąŇïijŇåŽäæ■d'è£ŹäÿłåŠ¡äžď'äijŽçńŇå■şè£ŤåŻđãĂĆäÿ■è£ĞïijŇä¡ääŔŕäžěåČŔäÿŁéİćéĆčæăůæ§ěçIJŃäÿŐåőČçŻÿåĚşçŽĎpidæŰĞäžűåŠŇæŮěå£ŮãĂĆèęĄåĄIJæ

```
bash % daemon.py stop
bash %
```

èőĺèőž

æIJňèŁĆåőŽäźĽäžĘäÿĂäÿłåĞ¡æŢř daemonize() ïijŇåIJĺçĺŃåžŕåŘŕåŁĺæŮűèćńèřČçŤĺä¡£å¿ŮçĺŃåžŔ
daemonize() åĞ¡æŢřåŔłæŐěåŔŮåĚşéŤőå■ŮåŔĆæŢřïijŇè£ŹæăůçŽĎèŕĺåŔŕéĂĽ'åŔĆæŢřåIJĺèćńä¡£çŤĺæŮ
åőČäijŽåijžåĹűçŤĺæĹůåČŔäÿŃéİćè£Źæăůä¡£çŤĺåőČïijŽ

```
daemonize('daemon.pid',
          stdin='/dev/null,
          stdout='/tmp/daemon.log',
          stderr='/tmp/daemon.log')
```

èĂŇäÿ■æŸŕåČŔäÿŃéİćè£Źæăůå Řńçş Łäÿ■æÿĚçŽĎèřČçŤĺïijŽ

```
# Illegal. Must use keyword arguments
daemonize('daemon.pid',
          '/dev/null', '/tmp/daemon.log','/tmp/daemon.log')
```

åĹŻåžžäÿĂäÿłåőĹæŁd'è£ŽçĺŃçŽĎæ■ěéłd'çIJŃäÿŁåŐžäÿ■æŸŕå¿ĹæŸŞæĞĆïijŇä¡ĘæŸŕådğä¡ŞæĂİæČ
éęŮåĚĹïijŇäÿĂäÿłåőĹæŁd'è£ŽçĺŃå£ĚéąžèęĄäžŐçĹűè£ŽçĺŃäÿ■èĎšçężãĂĆ è£ŹæŸŕçŤš
os.fork() æŞ■ä¡IJæİěåőŇæĹŘçŽĎïijŇåźűçńŇå■şèćńçĹűè£ŽçĺŃçżĹæ■ćãĂĆ

åIJĺå■Řè£ŽçĺŃåŔŸæĹŘå■d'åĎ£åŘŐïijŇèřČçŤĺ os.setsid()
åĹŻåžžäžĘäÿĂäÿłåĚĺæŰřçŽĎè£ŽçĺŃäijŽèŕİïijŇåźűèő¿ç¡őå■Řè£ŽçĺŃäÿžéęŮéćĘãĂĆ
åőČäijŽèő¿ç¡õè£ŹäÿłåRè£ŽçĺŃäÿžæŰřçŽĎè£ŽçĺŃçżĎçŽĎéęŮéćĘïijŇåźűçąőä£İäÿ■äijŽåĘ■æIJL'æŐğåĹűç
åęĆæđIJè£ŹäžŽåŘňäÿŁåŐžåd'lé■Ťåźžïij ŇåŽääÿžåőČéIJĂèęĄåřĘåőĹæŁd'è£ŽçĺŃåŘŇçżĹçńŕåĹĘçęžåijĂåźű
èřČçŤĺ os.chdir() åŠŇ os.umask(0) æŤźåŔŸäžĘå¡ŞåL'■åůëä¡IJçŽőå¡Ţåźűé Ğ■ç¡őæŮĞäžűæİČéŹŘæ
ä£őæŤźçŽőå¡ŢéĂŽåÿÿæŸŕäÿłåě¡äÿżæĎŘïijŇåŽääÿžè£Źæăůåŕŕäžëä¡£å¿ŮåőČäÿ■åĘ■åůëä¡IJåIJĺèćńåŘŕåŁĺæ

åŘęåd'ŰäÿĂäÿłèřČçŤĺ os.fork() åIJĺè£ŹéĞŇæŽt'åŁăçěđçğŸçĆźãĂĆ
è£ŹäÿĂæ■ëä¡£å¿ŮåőĹæŁd'è£ŽçĺŃåd'śåŐžäžĘèŐůåŘŰæŰřçŽĎæŐğåĹűçżĹçńŕçŽĎèČ¡åŁŽåźűäÿŤèől'åőČæ
ïijĹæIJňèt'ĺäÿŁïijŇèŕědaemonæŤ¿åijČäžĘåőČçŽĎäijŽèŕİéęŮéćĘä¡Őä¡■ïijŇåŽăæ■d'åĘ■äź§æšąæIJL'æİČéŹŘ
år¡çőąä¡ååŕŕäžěå£¡çŢèè£ŹäÿĂæ■ëïijŇä¡ĘæŸŕæIJĂåě¡äÿ■èęĄè£ŹäźĹåĄŽãĂĆ

äÿĂæŮęåőĹæŁd'è£ŽçĺŃèćńæ■čçąőçŽĎåĹĘçężïijŇåőČäijŽéĞ■æŰřåĹİåğŃåŇŰæăĞåĞĘI/OæţĄæŇĞåĚ
è£ŹäÿĂéČĺåĹĘæIJL'çĆźéŽ¿æĞĆãĂĆèů§æăĞåĞĘI/OæţĄçŽÿåĚşçŽĎæŰĞäžűåŕźèśąçŽĎåijŢçŤĺåIJĺèğčéĞŁå
ïijĹsys.stdout, sys.__stdout__ç■L'ïijL'ãĂĆ äźĚäźĚçőĂå■ŢçŽĎåĚşéŮ■
sys.stdout åźűéĞ■æŰřæŇĞåőŽåőČæŸŕèąŇäÿ■éĂŽçŽĎïijŇ
åŽääÿžæšąåŁđæşŢç§ééĄŞåőČæŸŕåŘęåĚĺéČĺéČ¡æŸŕçŤĺçŽĎæŸŕ sys.stdout ãĂĆ
è£ŹéĞŇïijŇæĹŚäžňæL'ŞåijĂäžĘäÿĂäÿłå■ŢçŃňçŽĎæŰĞäžűåŕźèśąïijŇåźűèŕČçŤĺ os.
dup2() ïijŇ çŤĺåőČæİěäžčæŽ£èćń sys.stdout ä¡£çŤĺçŽĎæŰĞäžűæŔŘè£řçňęãĂĆ
è£ŹæăůïijŇsys.stdout ä¡£çŤĺçŽĎåŐ§åğŃæŰĞäžűäijŽèćńåĚşéŮ■åźűçŤśæŰřçŽĎæİèæŽ£æ■ćãĂĆ
è£ŸèęĄåijžèŕČçŽĎæŸŕäžżä¡ŢçŤĺäžŐæŰĞäžűçijÚçĄæĹŰæŰĞæIJňåd'ĎçŘĘçŽĎæăĞåĞĘI/OæţĄè£ŸäijŽä

åőĹæŁd'è£ŽçĺŃçŽĎäÿĂäÿłéĂŽåÿÿåőđèůţæŸŕåIJĺäÿĂäÿłæŰĞäžűäÿ■åĘźåĚèè£ŽçĺŃIDïijŇåŕäžěèćńåĚ
daemonize() åĞ¡æŢřçŽĎæIJĂåŘŐéČĺåĹĘåĘźäžĘè£ŹäÿłæŰĞäžűïijŇä¡ĘæŸŕåIJĺçĺŃåžŘçžĹæ■ćæŰűåĹăè
atexit.register() åĞ¡æŢřæşĺåĘŇäžĘäÿĂäÿłåĞ¡æŢřåIJĺPythonèğčéĞŁåŹĺçžĹæ■ćæŰűæL'ğèąŇãĂĆ
äÿĂäÿłårźäžŐSIGTERMçŽĎä£ąåŘůåd'ĎçŘĘåŹĺçŽĎåőŽäźL'åŘŇæăůéIJĂèęĄèćńäijŸéŽĚçŽĎåĚşéŮ■ãĂĆ
ä£ąåŘůåd'ĎçŘĘåŹĺçőĂå■ŢçŽĎæŁŽåĞžäžĘ SystemExit() åijĆåÿÿãĂĆ
æĹŰèőÿè£ŹäÿĂæ■ěçIJŃäÿŁåŐžæšąå£ĚèęĄïijŇä¡ĘæŸŕæšąæIJL'åőČïijŇ
çžĹæ■ćä£ąåŘůäijŽä¡£å¿Ůäÿ■æL'ğèąŇ atexit.register()
æşĺåĘŇçŽĎæÿĚçŘĘæŞ■ä¡IJçŽĎæŰůåĂŹårśæİĂæŐL'äžĘèğčéĞŁåŹĺãĂĆ

äÿĂäÿłæİĂæŐĽè£ŻçÍŃçŽĎä¿Ńå■ŘäžčçăĄåŔŕäżěåIJÍçÍŃåžŔæIJĂåŘŐçŽĎ `stop` åŚ¡äżď'çŽĎæŞ■ä¡IJäÿ■çIJŃåĹřãĂĆ

æŻť'åď'ŽåĚşäžŐçijŰåĘŹåőĹæŁď'è£ŻçÍŃçŽĎä£ąæĄŕåŔŕäżěæ§ěçIJŃãĂŁUNIX çŐŕåćČéńŸçžğçijŰçÍŃãĂŃ, çňňäžŇçĽĹ by W. Richard Stevens and Stephen A. Rago (Addison-Wesley, 2005)ãĂĆ åř¡çőąåőČæŸŕåĚşæşÍäÿŐCèŕ■èÍĂçijŰçÍŃïijŇä¡ĘæŸŕæĽĂæIJL'çŽĎåĘĚåőźéČ¡éĂĆçŤĺäžŐPythonïijŇ åŻăäÿžæĽĂæIJL'éIJĂèęĄçŽĎPOSIXåĞ¡æŢřéČ¡åŔŕäżěåIJÍæăĞåĞĘåžŞäÿ■æĽ¿åĹřãĂĆ

CHAPTER 15

çñňå■ĄäÿĽçńäïijŽèĎŽæIJňçijŰçÍÑäÿŐçşżçż§çőąçŘĘ

èőÿåď Žäžža¡£çŤÍPythonä¡IJäÿžäÿĂäÿłshellèĎŽæIJňçŽĎæŻ£äżčïijŇçŤÍæİěåőđçŐřåÿÿçŤÍçşżçż§äżżåŁą

Contents:

13.1 éĂŽè£ĞéĞ■åőŽåŘŚ/çőąéĄŞ/æŰĞäžűæŐěåŔŮè¿ŞåĚě

éŮőécŸ

ä¡ăåÿŇæIJŻä¡ăçŽĎèĎŽæIJňæŐěåŔŮäżżä¡ŢçŤÍæĹůèőď äÿžæIJĂçőĂå■ŢçŽĎè¿ŞåĚěæŰźåijŔãĂĆåŇĚæ
éĞ■åőŽåŘŚæŰĞäžűåĹřèŕěèĎŽæIJňïijŇæĹŰåIJÍåŚ¡äżď èąŇäÿ■äijåéĂŠäÿĂäÿłæŰĞäžűåŘ■æĹŰæŰĞäžűåŘ■

èğčåĘşæŰźæąĹ

PythonåĘĚç¡őçŽĎ `fileinput` æÍąåİŮèőľ è£ŹäÿłåŔŸå¿ŮçőĂå■ŢãĂĆåęĆæđIJä¡ăæIJL’äÿĂäÿłäÿŇéİćè

```
#!/usr/bin/env python3
import fileinput

with fileinput.input() as f_input:
    for line in f_input:
        print(line, end='')
```

éĆčäźĹä¡ăåřśèČ¡äżěåL’■éİćæŔŘåĹřçŽĎæL’ĂæIJL’æŰźåijŔæİěäÿžæ■ď èĎŽæIJňæŔŘä¿Żè¿ŞåĚěãĂĆåĄ
`filein.py` åźűåřĘåĚűåŔŸäÿžåŔŕæL’ğèąŇæŰĞäžűïijŇ éĆčäźĹä¡ăåŔŕäžěåČŔäÿŇéİćè£ŹæăůèřČçŤÍåőČïijŇ

```
$ ls | ./filein.py          # Prints a directory listing to stdout.
$ ./filein.py /etc/passwd   # Reads /etc/passwd to stdout.
$ ./filein.py < /etc/passwd # Reads /etc/passwd to stdout.
```

èőÍèőž

fileinput.input() åĹŻåžžåźűè£ŤåŻđäÿĂäÿł FileInput çśżçŽĎåőđä¿ŃãĂĆ
èŕěåőđä¿ŃéŹd'äžĘæŃěæIJL'äÿĂäžŻæIJL'çŤĺçŽĎåÿőåŁl'æŰźæşŢåd'ŰïijŇåőČè£ŸåŘŕèćńå¡ŞåĄŽäÿĂäÿłäÿŁă
åŽăæ■d'ïijŇæŢt'åŘĹèţůæİěïijŇåęĆæđIJæĹŚäžňèęĄåĘŹäÿĂäÿłæL'Şå■řåd'Žäÿłæ ŰĞäžűè¿ŞåĞžçŽĎèĎŽæIJń

```
>>> import fileinput
>>> with fileinput.input('/etc/passwd') as f:
>>>     for line in f:
...         print(f.filename(), f.lineno(), line, end='')
...
/etc/passwd 1 ##
/etc/passwd 2 # User Database
/etc/passwd 3 #

<other output omitted>
```

éĂŽè£ĞåřĘåőČä¡IJäÿžäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺä¡£çŤĺïijŇåŘŕäžěçąőä£İåőČäÿ■åĘ■ä¡£çŤĺæŮűæŰ
èĂŇäÿŤæĹŚäžňåIJĺäźŃçśżè£ŸæijŤçd'žäžĘ FileInput çŽĎäÿĂäžŻæIJL'çŤĺçŽĎåÿőåŁl'æŰźæşŢæİěèŐůå

13.2 çżĹæ■ćçĺŃåžŔåźűçżŹåĞžéŤŹèŕŕä£ąæĄŕ

éŮőécŸ

ä¡ăæČşåŘŚæăĞåĞĘéŤŹèŕŕæL'Şå■řäÿĂæİąæűĹæĄŕåźűè£ŤåŻđæ§ŘäÿłéİđéŽűçŁűæĂĄçăĄæİěçżĹæ■ćçĺ

èğčåĘşæŰźæąĹ

ä¡ăæIJL'äÿĂäÿłçĺŃåžŔåČŔäÿŃéİćè£ŹæăůçżĹæ■ćïijŇæŁŻåĞžäÿĂäÿł SystemExit
åijĆåÿÿïijŇä¡£çŤĺéŤŹèŕŕæűĹæĄŕä¡IJäÿžåŔĆæŢřãĂĆä¿ŃåęĆïijŽ

```
raise SystemExit('It failed!')
```

åőČäijŽåřĘæűĹæĄŕåIJĺ sys.stderr äÿ■æL'Şå■řïijŇçĎűåŘŐçĺŃåžŔäžěçŁűæĂĄçăĄ1éĂĂåĞžãĂĆ

èőÍèőž

æIJňèŁĆèŹ¡çĎűå¿Łç§■åřŔïijŇä¡ĘæŸŕåőČèČ¡èğčåĘşåIJĺåĘŹèĎŽæIJňæŮűçŽĎäÿĂäÿłåÿÿèğĄéŰőécŸã
äź§åřśæŸŕèŕt'ïijŇå¡Şä¡ăæČşèęĄçŹĹæ■ćæ§ŘäÿłçĺŃåžŔæŮűïijŇä¡ăåŔŕèČ¡äijŽåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
import sys
sys.stderr.write('It failed!\n')
raise SystemExit(1)
```

åęĆæđIJä¡ăçŻť'æŐěåřĘæűĹæĄŕä¡IJäÿžåŔĆæŢřäijăçżŹ SystemExit() ïijŇéĆčäźĹä¡ăåŔŕäžěçIJĄçŢěåĚűäžŰæ■ěéłđ'ïijŇ æŕŤåęĆimportèŕ■åŔěæĹŰåřĘéŤŹèŕŕæűĹæĄŕåĘŹåĚě sys.stderr

13.3 èğčæđŘåŚ¡äžđ'èąŇéĂĽ'éąź

éŮőécŸ

ä¡ăçŽĎçÍŃåžŘåęĆä¡ŢèČ¡åđ'§èğčæđŘåŚ¡äžđ'èąŇéĂĽ'éąźïijĹä¡■äžŐsys.argväÿ■ïijĽ'

èğčåĘşæŰźæąĹ

argparse æĺąåİŮåŔŕèćńçŤĺæİěèğčæđŘåŚ¡äžđ'èąŇéĂĽ'éąźãĂĆäÿŃéİćäÿĂäÿłçőĂå■Ţä¿Ńå■ŘæijŤçđ'ž

```
# search.py
'''
Hypothetical command-line tool for searching a collection of
files for one or more text patterns.
'''
import argparse
parser = argparse.ArgumentParser(description='Search some files')

parser.add_argument(dest='filenames',metavar='filename', nargs='*')

parser.add_argument('-p', '--pat',metavar='pattern', required=True,
                    dest='patterns', action='append',
                    help='text pattern to search for')

parser.add_argument('-v', dest='verbose', action='store_true',
                    help='verbose mode')

parser.add_argument('-o', dest='outfile', action='store',
                    help='output file')

parser.add_argument('--speed', dest='speed', action='store',
                    choices={'slow','fast'}, default='slow',
                    help='search speed')

args = parser.parse_args()

# Output the collected arguments
print(args.filenames)
```

```
print(args.patterns)
print(args.verbose)
print(args.outfile)
print(args.speed)
```

èŕěçÍŃåžŔåőŽäźĽäžĘäÿĂäÿłåęĆäÿŃä¡£çŤÍçŽĎåŚ¡äžd'èąŇèğčæđŘåŹÍïijŽ

```
bash % python3 search.py -h
usage: search.py [-h] [-p pattern] [-v] [-o OUTFILE] [--speed {slow,
↪fast}]
                 [filename [filename ...]]

Search some files

positional arguments:
  filename

optional arguments:
  -h, --help            show this help message and exit
  -p pattern, --pat pattern
                        text pattern to search for
  -v                    verbose mode
  -o OUTFILE            output file
  --speed {slow,fast}   search speed
```

äÿŃéÍćçŽĎéČĺåĹĘæijŤçd'žäžĘçÍŃåžŔäÿ■çŽĎæŢřæ■őéČĺåĹĘãĂĆäżŤçżĘèğĆåŕ§print()èí■åŔěçŽĎæL'Ş

```
bash % python3 search.py foo.txt bar.txt
usage: search.py [-h] -p pattern [-v] [-o OUTFILE] [--speed {fast,
↪slow}]
                 [filename [filename ...]]
search.py: error: the following arguments are required: -p/--pat

bash % python3 search.py -v -p spam --pat=eggs foo.txt bar.txt
filenames = ['foo.txt', 'bar.txt']
patterns  = ['spam', 'eggs']
verbose   = True
outfile   = None
speed     = slow

bash % python3 search.py -v -p spam --pat=eggs foo.txt bar.txt -o␣
↪results
filenames = ['foo.txt', 'bar.txt']
patterns  = ['spam', 'eggs']
verbose   = True
outfile   = results
speed     = slow

bash % python3 search.py -v -p spam --pat=eggs foo.txt bar.txt -o␣
↪results \
            --speed=fast
```

```
filenames = ['foo.txt', 'bar.txt']
patterns  = ['spam', 'eggs']
verbose   = True
outfile   = results
speed     = fast
```

åŕźäžŐéĂĽéąźåĂijçŽĎè£ŻäÿĂæ■ěåď˙ĎçŘĘçŤśçÍŃåžŔæÌěåĘşåőŽïijŇçŤÍä¡äèĞłåůśçŽĎéĂżè¿ŚæÌěæŻ£
`print()` åĞ¡æŢřãĂĆ

èőÍèőž

`argparse` æÍąåİŮæŸŕæăĞåĞĘåžŞäÿ■æIJĂåď˙ğçŽĎæÍąåİŮäźŃäÿĂïijŇæŃěæIJĽåď˙ğéĞŔçŽĎéĚ■ç¡őé
æIJňèŁĆåŔłæŸŕæijŤçď˙žäžĘåĚűäÿ■æIJĂå§žçąĂçŽĎäÿĂäžŻçĽźæĂğïijŇåÿőåŁľä¡ääĚěéŮÍãĂĆ

äÿžäžĘèğčæđŘåŚ¡äžď˙èąŇéĂĽéąźïijŇä¡äéęŰåĚÍèęĄåĹŻåžžäÿĂäÿł
`ArgumentParser` åőđä¿ŃïijŇ åźűä¡£çŤÍ `add_argument()`
æŰźæşŢåčřæŸŐä¡äæČşèęĄæŤŕæŇĄçŽĎéĂĽéąźãĂĆ åIJÍæŕŔäÿł `add-argument()`
èřČçŤÍäÿ■ïijŇ`dest` åŔĆæŢřæŇĞåőŽèğčæđŘçżŞæđIJèćńæŇĞæť¿çżŹåśđæĂğçŽĎåŘ■å■ŮãĂĆ
`metavar` åŔĆæŢřèćńçŤÍæÌěçŤ§æĹŘåÿőåŁľä£ąæĄŕãĂĆ`action`
åŔĆæŢřæŇĞåőŽèů§åśđæĂğåŕźåžŤçŽĎåď˙ĎçŘĘéĂżè¿ŚïijŇ éĂŽåÿÿçŽĎåĂijäÿž `store`
,èćńçŤÍæÌěå■ŸåĆÍæ§ŘäÿłåĂijæĹŰèőšåď˙ŽäÿłåŔĆæŢřåĂijæŤűéŻĘåĹřäÿĂäÿłåĹŮèąÍäÿ■ãĂĆ
äÿŃéÌćçŽĎåŔĆæŢřæŤűéŻĘæĽĂæIJĽåĽľä¡ŹçŽĎåŚ¡äžď˙èąŇåŔĆæŢřåĹřäÿĂäÿłåĹŮèąÍäÿ■ãĂĆåIJÍæIJñä¿Ň

```
parser.add_argument(dest='filenames',metavar='filename', nargs='*')
```

äÿŃéÌćçŽĎåŔĆæŢřæăźæ■őåŔĆæŢřæŸŕåŘęå■ŸåIJÍæÌěèő¿ç¡őäÿĂäÿł `Boolean`
æăĞå£ŮïijŽ

```
parser.add_argument('-v', dest='verbose', action='store_true',
                    help='verbose mode')
```

äÿŃéÌćçŽĎåŔĆæŢřæŐěåŔŮäÿĂäÿłå■ŢçŃňåĂijåźűåřĘåĚűå■ŸåĆÍäÿžäÿĂäÿłå■ŮçňęäÿšïijŽ

```
parser.add_argument('-o', dest='outfile', action='store',
                    help='output file')
```

äÿŃéÌćçŽĎåŔĆæŢřèŕť˙æŸŐåĚĄèőÿæ§ŘäÿłåŔĆæŢřéĞ■åď˙■åĞžçŐŕåď˙ŽæňąïijŇåźűåřĘåőČäžňè£¡åŁäåÍ
`required` æăĞå£ŮèąÍçď˙žèŕěåŔĆæŢřèĞşåřŚèęĄæIJĽäÿĂäÿłãĂĆ`-p` åŠŇ `--pat`
èąÍçď˙žäÿď˙äÿłåŔĆæŢřåŘ■å¡ćåijŔéČ¡åŔŕä¡£çŤÍãĂĆ

```
parser.add_argument('-p', '--pat',metavar='pattern', required=True,
                    dest='patterns', action='append',
                    help='text pattern to search for')
```

æIJĂåŘŐïijŇäÿŃéÌćçŽĎåŔĆæŢřèŕť˙æŸŐæŐěåŔŮäÿĂäÿłåĂijïijŇä¡ĘæŸŕäijŽåřĘåĚűåŠŇåŔŕèČ¡çŽĎéĂ

```
parser.add_argument('--speed', dest='speed', action='store',
                    choices={'slow','fast'}, default='slow',
                    help='search speed')
```

äÿĂæŮęåŔĆæŢřéĂĽéąźèćńæŇĞåőŽïijŇä¡ăåřśåŕřäžěæĽğèąŇ `parser.parse()` æŰźæşŢäžĘãĂĆ åőČäijŽåď˙ĎçŘĘ `sys.argv` çŽĎđĂijåźűè£ŤåŻđäÿĂäÿłçżŞæđIJåőđä¿ŃãĂĆ æŕŔäÿłåŔĆæŢřåĂijäijŽèćńèő¿ç¡őæĹŘèŕěåőđä¿Ńäÿ■‘‘add_ar æŰźæşŢçŽĎ‘‘dest‘‘ åŔĆæŢřæŇĞåőŽçŽĎåśđæĂğåĂijãĂĆ

è£Ÿå¿Ĺåď˙Žçğ■åĚűäžŰæŰźæşŢèğčæđŘåŚ¡äżď˙èąŇéĂĽéąźãĂĆ ä¿ŃåęĆïijŇä¡ăåŔŕèČ¡äijŽæĽ˙ŃåŁĺçŽĎđď˙ĎçŘĘ `sys.argv` æĹŰèĂĚä¡£çŤĺ `getopt` æĺąåİŮãĂĆ ä¡ĘæŸŕïijŇåęĆæđIJä¡áéĞĞçŤĺæIJňèŁĆçŽĎđæŰźåijŔïijŇårĘäijŽåĞŔårŚå¿Ĺåď˙ŽåĘŮä¡Źäžčçăąï `argparse` æĺąåİŮåůšçżŔåÿőä¡ăåď˙ĎçŘĘäžĘãĂĆ ä¡ăåŔŕèČ¡è£ŸäijŽçćřåĹřä¡£çŤĺ `optparse` åžŞèğčæđŘéĂĽéąźçŽĎäžčçăąãĂĆ år¡çőą `optparse` åŠŇ `argparse` å¿ĹåČŔïijŇä¡ĘæŸŕåŘŐèĂĚæŻť˙åĚĹè£ŻïijŇåŻăæ■d˙åIJĺæŰřçŽĎçĺŃåžŔäÿ■ä¡ăåžŤèŕěä¡£çŤĺåőČãĂĆ

13.4 è£ŘèąŇæŮűůåijźåĞžårĘçăĄè¿ŞåĚěæŔŘçď˙ž

éŮőécŸ

ä¡ăåĘŹäžĘäÿłèĎŽæIJňïijŇè£ŘèąŇæŮűéIJĂèęĄäÿĂäÿłårĘçăĄãĂĆæ■d˙èĎŽæIJňæŸŕäžď˙äžŠåijŔçŽĎïij èĂŇæŸŕéIJĂèęĄåijźåĞžäÿĂäÿłårĘçăĄè¿ŞåĚěæŔŘçď˙žïijŇèőľçŤĺæĹůèĞłåůśè¿ŞåĚěãĂĆ

èğčåĘşæŰźæąĹ

è£ŹæŮűůåĂŹPythonçŽĎ `getpass` æĺąåİŮæ■čæŸŕä¡ăæĽ˙ĂéIJĂèęĄçŽĎđãĂĆä¡ăåŔŕäžěèőľä¡ăå¿Ĺè¡żæİ¿ åźűäÿŤäÿ■äijŽåIJĺçŤĺæĹůçżĹçńŕåŽđæŸ¿årĘçăĄãĂĆäÿŃéİćæŸŕåĚůä¡Şäžčçăąïijž

```
import getpass

user = getpass.getuser()
passwd = getpass.getpass()

if svc_login(user, passwd):     # You must write svc_login()
   print('Yay!')
else:
   print('Boo!')
```

åIJĺæ■d˙äžčçăąäÿ■ïijŇ`svc_login()` æŸŕä¡ăèęĄåőđçŐrçŽĎåď˙ĎçŘĘårĘçăĄçŽĎđĞ¡æŢřïijŇåĚůä¡Şç

èőĺèőž

æşĺæĎŔåIJĺåĽ˙■éİćäžčçăąäÿ■ `getpass.getuser()` äÿ■äijŽåijźåĞžçŤĺæĹůåŘ■çŽĎè¿ŞåĚěæŔŘçď˙žãĂĆ åőČäijŽæäźæ■őèŕěçŤĺæĹůçŽĎshel-lçŐŕáćČæĹŰèĂĚäijŽä¿Íæ■őæIJňåIJŕçşżçż§çŽĎårĘçăĄåžŞïijĹæŤŕæŇĄ *pwd* æĺąåİŮçŽĎåźşåŔřïijĽ˙æİěä¡£çŤĺå¡ŞåĽ˙■çŤĺæĹůçŽĎçŹżå¡ŢåŘ■ïijŇ

åęĆæđIJä¡ăæČşæŸ¿çď˙žçŽĎåijźåĞžçŤĺæĹůåŘ■è¿ŞåĚěæŔŘçď˙žïijŇä¡£çŤĺåĘĘç¡őçŽĎ `input` åĞ¡æŢřïijž

```
user = input('Enter your username: ')
```

è£ŸæIJL'äÿĂçĆźå¿ĹéĞ■èęĄïijŇæIJL'äžŻçşżçż§åŕŕèČ¡äÿ■æŤŕæŇĄ `getpass()` æŰźæşŢéŽŘèŮŔè¿ŞåĚěåŕĘçĂĄãĂĆ è£ŹçĝæČĚåĘţäÿŃïijŇPythonäijŽæŔŘåL'■è■ęåŚŁä¡ăè£ŹäžŻéŮőécŸïi

13.5 èŐůåŔŰçżĹçńŕçŽĎåď'ĝåřŔ

éŮőécŸ

ä¡ăéIJĂèęĄç§ěéĄŞå¡ŞåL'■çżĹçńŕçŽĎåď'ĝåřŔäžěä¿£æ■čçąőçŽĎæăijåijŔåŇŰè¿ŞåĞžãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤÍ `os.get_terminal_size()` åĞ¡æŢřæİěåĄŽåĹřè£ŹäÿĂçĆźãĂĆ

äžčçăĄçď'žä¿ŃïijŽ

```
>>> import os
>>> sz = os.get_terminal_size()
>>> sz
os.terminal_size(columns=80, lines=24)
>>> sz.columns
80
>>> sz.lines
24
>>>
```

èőÍèőž

æIJL'åď'łåď'ŽæŰźåijŔæİěå¿Ůç§ěçżĹçńŕåď'ĝåřŔäžĘïijŇäžŐèŕżåŔŰçŐŕåćČåŔŸéĞŔåĹřæL'ĝèąŇåžŢåśĆç `ioctl()` åĞ¡æŢřç■L'ç■L'ãĂĆ äÿ■è£ĞïijŇäÿžäżĂäźĹèęĄåŐżçăŤçľ'űè£ŹäžŻåď'■æİĆçŽĎåŁđæşŢèĂŇäÿ■æ

13.6 æL'ĝèąŇåď'ŰéČĺåŚ¡äžď'åźűèŐůåŔŰåőČçŽĎè¿ŞåĞž

éŮőécŸ

ä¡ăæČşæL'ĝèąŇäÿĂäÿłåď'ŰéČĺåŚ¡äžď'åźűäžěPythonå■ŮçňęäÿšçŽĎå¡ćåijŔèŐůåŔŰæL'ĝèąŇçżŞæđIJãĂ

èğčåĘşæŰźæąĹ

ä¡£çŤÍ `subprocess.check_output()` åĞ¡æŢřãĂĆä¿ŃåęĆïijŽ

```
import subprocess
out_bytes = subprocess.check_output(['netstat','-a'])
```

è£ŹæőţäžčçăĄæĽğèąŇäÿĂäÿłæŇĞåőŽçŽĎåŚ¡äžď åźűårĘæĽğèąŇçżŞædIJäžěäÿĂäÿłå■ŮèŁĆå■Ůçňęäÿ åęĆædIJä¡ăéIJĂèęĄæŰĞæIJňå¡ćåijŔè£ŤåŻđïijŇåŁääÿĂäÿłèğčçăĄæ■ěéłď å■şåŘřãĂĆä¿ŃåęĆïijŽ

```
out_text = out_bytes.decode('utf-8')
```

åęĆædIJècńæĽğèąŇçŽĎåŚ¡äžď äžěéİđéŻűçăĄè£ŤåŻđïijŇářšäijŽæŁŻåĞžåijĆåÿÿãĂĆ äÿŃéIćçŽĎä¿Ńå■Řæ■ŢèŐůåĹřéŤŹèŕřåźűèŐůåŘŰè£ŤåŻđçăĄïijŽ

```
try:
    out_bytes = subprocess.check_output(['cmd','arg1','arg2'])
except subprocess.CalledProcessError as e:
    out_bytes = e.output       # Output generated before error
    code      = e.returncode   # Return code
```

éżŸèőď æČĚåĘţäÿŃïijŇcheck_output() äžĚäžĚè£ŤåŻđè¿ŞåĚěåĹřæăĞåĞĘè¿ŞåĞžçŽĎåĂijãĂĆ åęĆædIJä¡ăéIJĂèęĄåŘŇæŮűæŤűéŻĘæăĞåĞĘè¿ŞåĞžåŠŇéŤŹèŕřè¿ŞåĞžïijŇä¡£çŤÍ `stderr` åŘĆæŢřïijŽ

```
out_bytes = subprocess.check_output(['cmd','arg1','arg2'],
                                    stderr=subprocess.STDOUT)
```

åęĆædIJä¡ăéIJĂèęĄçŤÍäÿĂäÿłèűĚæŮűæIJžåĹűæİěæĽğèąŇåŚ¡äžď ïijŇä¡£çŤÍ `timeout` åŘĆæŢřïijŽ

```
try:
    out_bytes = subprocess.check_output(['cmd','arg1','arg2'],␣
↪timeout=5)
except subprocess.TimeoutExpired as e:
    ...
```

éĂŽåÿÿæİěèőšïijŇåŚ¡äžď çŽĎæĽğèąŇäÿ■éIJĂèęĄä¡£çŤÍåĹřåžŢåśĆshellçŐŕåćČïijĹæŕŤåęĆshãĂĄbash äÿĂäÿłå■ŮçňęäÿšåĹŮèąÍäijŽèćńäijăéĂŠçżŹäÿĂäÿłä¡Őçžğçşżçż§åŚ¡äžď ïijŇæŕŤåęĆ `os.execve()` ãĂĆ åęĆædIJä¡ăæČşèőĺ åŚ¡äžď èćńäÿĂäÿłshellæĽğèąŇïijŇäijăéĂŠäÿĂäÿłå■ŮçňęäÿšåŔĆæŢřïij `shell=True`. æIJĽæŮűåĂŹä¡ăæČşèęĄPythonåŐžæĽğèąŇäÿĂäÿłåď■æİĆçŽĎshellåŚ¡äžď çŽĎæŮűåĂŹè

```
out_bytes = subprocess.check_output('grep python | wc > out',␣
↪shell=True)
```

éIJĂèęĄæşÍæĎŔçŽĎæŸŕåIJÍshelläÿ■æĽğèąŇåŚ¡äžď äijŽå■ŸåIJÍäÿĂåőŽçŽĎåőĽåĚÍéčŐéŹĺ ïijŇçĽźåĹ è£ŹæŮűåĂŹåŔŕäžěä¡£çŤÍ `shlex.quote()` åĞ¡æŢřæİěèőšåŔĆæŢřæ■čçąőçŽĎçŤÍåŔŇåijŢçŤÍåijŢèţůæİěã

èőĺèőž

ä¡£çŤĺ `check_output()` åĞ¡æŢřæŸŕæĽ˘ğèąŇåď˙ŰéČĺåŚ¡äžď˙åźűèŐůåŘŰåĚűè£ŤåŻđåĂijçŽĎæIJĂçĉ
ä¡ĘæŸŕřijŇåęĆæđIJä¡ăéIJĂèęĄåŕźå■Řè£ŻçÍŃåĄŽæŻť˙åď˙■æİĆçŽĎäžď˙äžŠïijŇæŕŤåęĆçżŹåőČåŘŚéĂĄè¿Şå
è£ŹæŮűåĂŹåŘŕçŻť˙æŐěä¡£çŤĺ `subprocess.Popen` çśżãĂĆä¿ŃåęĆïijŽ

```
import subprocess

# Some text to send
text = b'''
hello world
this is a test
goodbye
'''

# Launch a command with pipes
p = subprocess.Popen(['wc'],
          stdout = subprocess.PIPE,
          stdin = subprocess.PIPE)

# Send the data and get the output
stdout, stderr = p.communicate(text)

# To interpret as text, decode
out = stdout.decode('utf-8')
err = stderr.decode('utf-8')
```

`subprocess` æÍąåİŮåŕźäžŐä¿İèţŰTTYçŽĎåď˙ŰéČĺåŚ¡äžď˙äÿ■åŘĹéĂĆçŤĺãĂĆ
ä¿ŃåęĆïijŇä¡ăäÿ■èČ¡ä¡£çŤĺåőČæİěèĞłåŁĺåŇŰäÿĂäÿłçŤĺæĹůè¿ŞåĚěåŕĘçăĄçŽĎäżžåŁąïijĹæŕŤåęĆäÿĂäÿłss
è£ŹæŮűåĂŹïijŇä¡ăéIJĂèęĄä¡£çŤĺåĹřçňňäÿĽ˙æŰźæÍąåİŮäžĘïijŇæŕŤåęĆå§žäžŐèŠŮåŘ■çŽĎ
`expect` åőűæŮŘçŽĎåůěåĚůïijĹpexpectæĹŰçśżäijijçŽĎïijĽ

13.7 åď■åĹűæĹŰèĂĚçğżåŁĺæŰĞäżűåŠŇçŻőå¡Ţ

éŮőécŸ

ä¡ăæČşèęĄåď■åĹűæĹŰçğżåŁĺæŰĞäżűåŠŇçŻőå¡ŢïijŇä¡ĘæŸŕåŘĹäÿ■æČşèřČçŤĺshellåŚ¡äžď˙ãĂĆ

èğčåĘşæŰźæąĹ

`shutil` æÍąåİŮæIJĽå¿Ĺåď˙Žä¿£æ■ůçŽĎåĞ¡æŢřåŘŕäžěåď˙■åĹűæŰĞäżűåŠŇçŻőå¡ŢãĂĆä¡£çŤĺèţůæİěé

```
import shutil

# Copy src to dst. (cp src dst)
shutil.copy(src, dst)
```

```
# Copy files, but preserve metadata (cp -p src dst)
shutil.copy2(src, dst)

# Copy directory tree (cp -R src dst)
shutil.copytree(src, dst)

# Move src to dst (mv src dst)
shutil.move(src, dst)
```

è£ŹäžŽåĞ¡æŢřçŽĎåŔĆæŢřéČ¡æŸŕå■Ůçňęäÿšå¡ćåijŔçŽĎæŰĞäżűæĹŰçŻőå¡ŢåŘ■ãĂĆ åžŢåśĆèŕ■äźĽ'æĺąæŃ§äžEçśżäijijçŽĎUnixåŚ¡äžd'ïijŇåęĆäÿŁéİćçŽĎæşĺéĞŁéČĺåĹEąĂĆ

éżŸèődʼæČĚåEţäÿŃïijŇåŕźäžŐçňęåŘůéŞ¿æŐěèĂŇåůšè£ŹäžŽåŚ¡äžd'åd'ĎçŘEçŽĎæŸŕåőČæŇĞåŘŚçŽ ä¿ŃåęĆïijŇåęĆæđIJæžŘæŰĞäżűæŸŕäÿĂäÿłçňęåŘůéŞ¿æŐěïijŃéČčäźĹçŻőæăĞæŰĞäżűåřĘäijŽæŸŕçňęåŘůé åęĆæđIJä¡ăåŘłæČşåd'■åĹűçňęåŘůéŞ¿æŐěæIJňèžńïijŃéČčäźĹéIJĂèęĄæŇĞåőŽåĚşéŤőå■ŮåŔĆæŢř `follow_symlinks` ,åęĆäÿŃïijŽ

åęĆæđIJä¡ăæČşä£İçŢŹèćńåd'■åĹűçŻőå¡Ţäÿ■çŽĎçňęåŘůéŞ¿æŐěïijŇåČŔè£ŹæăůåĄŽïijŽ

```
shutil.copytree(src, dst, symlinks=True)
```

`copytree()` åŕŕäžěèőľä¡ăåIJĺåd'■åĹűè£ĞçĺŃäÿ■éĂĽ'æŃľæĂğçŽĎå£¡çŢěæ§ŘäžŽæŰĞäżűæĹŰçŻőå ä¡ăåŕŕäžěæŔŘä¿ŻäÿĂäÿłå£¡çŢěåĞ¡æŢřïijŇæŐěåŔŮäÿĂäÿłçŻőå¡ŢåŘ■åŠŇæŰĞäżűåŘ■åĹŮèąĺä¡IJäÿžè¿ŞåĚ

```
def ignore_pyc_files(dirname, filenames):
    return [name in filenames if name.endswith('.pyc')]

shutil.copytree(src, dst, ignore=ignore_pyc_files)
```

Since ignoring filename patterns is common, a utility function ignore_patterns() has already been provided to do it. For example:

shutil.copytree(src, dst, ignore=shutil.ignore_patterns('~','.pyc'))

èőĺèőž

ä¡£çŤĺ `shutil` åd'■åĹűæŰĞäżűåŠŇçŻőå¡Ţäź§å£ŠçőĂå■ŢäžEçĆźåŘğãĂĆ äÿ■è£ĞïijŇåŕźäžŐæŰĞäżűåĚČæŢřæ■őä£qæĄŕïijŇ`copy2()` è£ŹæăůçŽĎåĞ¡æŢřåŔłèČ¡år¡èĞłåůśæIJĂåd'ğè èőŕéŮőæŮűéŮt'ãĂĄåĹŻäžžæŮűéŮt'åŠŇæİČéŹŘè£ŹäžŽå§žæIJñä£qæĄŕäijŽèćńä£ĺçŢŹïijŇ ä¡ĘæŸŕåŕźäžŐæL'ĂæIJL'èĂĚãĂĄACLsãĂĄètĎæžŘforkåŠŇåĚűäżŰæŻt'æůśåśĆæňąçŽĎæŰĞäżűåĚČä£qæA è£ŹäÿłèŸå¿Ůä¿ÌètŰäžŐåžŢåśĆæŞ■ä¡IJçşżçż§çśžåđŃåŠŇçŤĺæĹůæL'ĂæŇèæIJL'çŽĎèőŕéŮőæİČéŹŘãĂĆ ä¡ăéĂŽåÿÿäÿ■äijŽåŐżä¡£çŤĺ `shutil.copytree()` åĞ¡æŢřæİěæL'ğèąŇçşżçż§åd'ĞäŻ¡ãĂĆ å¡Şåd'ĎçŘEæŰĞäżűåŘ■çŽĎæŮűåĂŹïijŇæIJĂåě¡ä¡£çŤĺ `os.path` äÿ■çŽĎåĞ¡æŢřæİěçąőä£ĺæIJĂåd'ğçŽĎåŔŕçĞżæd'■æĂğïijĹçL'źåĹńæŸŕåŘŇæŮűèęĄéĂĆçŤĺäžŐUnixåŠŇWi ä¿ŃåęĆïijŽ

```
>>> filename = '/Users/guido/programs/spam.py'
>>> import os.path
```

```
>>> os.path.basename(filename)
'spam.py'
>>> os.path.dirname(filename)
'/Users/guido/programs'
>>> os.path.split(filename)
('/Users/guido/programs', 'spam.py')
>>> os.path.join('/new/dir', os.path.basename(filename))
'/new/dir/spam.py'
>>> os.path.expanduser('~/guido/programs/spam.py')
'/Users/guido/programs/spam.py'
>>>
```

ä¡£çŤĺ `copytree()` åď■åĹűæŰĞäžűåď źçŽĎäÿĂäÿłæčŸæĽŃçŽĎéŮőécŸæŸŕåŕźäžŐéŤŹèŕŕçŽĎåď
ä¿ŃåęĆïijŇåIJĺåď■åĹűè£ĞçĺŃäÿ■ïijŇåĞ¡æŢřåŔŕèČ¡äijŽçćřåĹřæ■§åİŔçŽĎçňęåŘůéŞ¿æŐěïijŇåŻääÿžæİČéŹ
äÿžäžEèğčåĘşè£ŹäÿłéŮőécŸïijŇæĽĂæIJL'çćřåĹřçŽĎéŮőécŸäijŽèćńæŤűéŻĘåĹřäÿĂäÿłåĹŮèąĺäÿ■åźűæĽŞ
äÿŃéIćæŸŕäÿĂäÿłä¿Ńå■ŘïijŽ

```
try:
    shutil.copytree(src, dst)
except shutil.Error as e:
    for src, dst, msg in e.args[0]:
        # src is source name
        # dst is destination name
        # msg is error message from exception
        print(dst, src, msg)
```

åęĆædIJä¡ăæŔŘä¿ŻåĚşéŤőå■ŮåŔĆæŢř `ignore_dangling_symlinks=True` ïijŇ
è£ŹæŮűåĂŹ `copytree()` äijŽå£¡çŢěæŐĽæŮăæŢĹçňęåŘůéŞ¿æŐěãĂĆ

æIJňèŁĆæijŤçď žçŽĎè£ŹäžŻåĞ¡æŢřéČ¡æŸŕæIJĂåÿÿèğĄçŽĎãĂĆäÿ■è£ĞïijŇshutil
è£ŸæIJĽæŻťåď ŽçŽĎåŠŇåď ■åĹűæŢřæ■őçŻÿåĚşçŽĎæŞ■ä¡IJãĂĆ
åőĈçŽĎæŰĞæąčå¿ĹåĂijå¿ŮäÿĂçIJŃïijŇåŔĆèĂČ Python documentation

13.8 åĹŻåžžåŠŇèğčåŐŃå¡Šæąčæ ŰĞäžű

éŮőécŸ

ä¡ăéIJĂèęĄåĹŻåžžæĹŰèğčåŐŃåÿÿèğĄæăijåijŔçŽĎå¡Šæąčæ ŰĞäžűïijĹæŕŤåęĆ.tar,
.tgzæĹŰ.zipïijL'

èğčåĘşæŰźæąĹ

shutil æĺąåİŮæŃěæIJL'äÿď äÿłåĞ¡æŢřâĂŤâĂŤ `make_archive()` åŠŇ
`unpack_archive()` åŘŕæť¿äÿŁçŤĺåIJžãĂĆ ä¿ŃåęĆïijŽ

```
>>> import shutil
>>> shutil.unpack_archive('Python-3.3.0.tgz')

>>> shutil.make_archive('py33','zip','Python-3.3.0')
'/Users/beazley/Downloads/py33.zip'
>>>
```

make_archive() çŽĎçňňäžŇäÿłåŔĆæŢřæŸŕæIJ§æIJŻçŽĎè¿ŞåĞžæăijåijŔãĂĆ åŔŕäžëä¡£çŤĺ get_archive_formats() èŐůåŔŰæĽĂæIJL'æŤŕæŇĄçŽĎå¡Šæąčæăijåij ŔåĹÙèąĺãĂĆä¿Ň

```
>>> shutil.get_archive_formats()
[('bztar', "bzip2'ed tar-file"), ('gztar', "gzip'ed tar-file"),
 ('tar', 'uncompressed tar file'), ('zip', 'ZIP file')]
>>>
```

èőĺèőž

Pythonè£ŸæIJL'åĚűäžŰçŽĎæĺąåİŮåŔŕçŤĺæİěåď'ĎçŘĘåď'Žçğ■å¡Šæąčæăijåij ŔïijĹæŕŤåęĆtarfile, zipfile, gzip, bz2ïijL'çŽĎåžŢåśĆçżĘèŁĆãĂĆ äÿ■è£ĞïijŇåęĆæđIJä¡ääžĚäžĚåŘłæŸŕèęĄåĹŻåžžæĹŰæŔŘåŔŰ åŔŕäžëçŻť'æŐëä¡£çŤĺ shutil äÿ■çŽĎè£ŹäžŻéńŸåśĆåĞ¡æŢřãĂĆ

è£ŹäžŻåĞ¡æŢřè£ŸæIJL'å¿Ĺåď'ŽåĚűäžŰéĂL'éąźïijŇçŤĺäžŐœŮěå£ŮæL'Şå■řãĂĄécĎæčĂãĂĄæŰĞäžűa åŔĆèĂČ shutilæŰĞæąč

13.9 éĂŽè£ĞæŰĞäžűåŘ■æ§ěæL'¿æŰĞäžű

éŮőécŸ

ä¡ăéIJĂèęĄåĘŹäÿĂäÿłæűL'åŔŁåĹřæŰĞäžűæ§ěæL'¿æŞ■ä¡IJçŽĎèĎŽæIJñïijŇæŕŤåęĆåŕźæŮěå£Ůå¡Šæą ä¡ăäÿ■æČşåIJĺPythonèĎŽæIJñäÿ■èřČçŤĺshellïijŇæĹŰèĂĚä¡ăèęĄåőđçŐřäÿĂäžŽshelläÿ■èČ¡åĄŽçŽĎåŁ§èČ

èğčåĘşæŰźæąĹ

æ§ěæL'¿æŰĞäžűïijŇåŔŕä¡£çŤĺ os.walk() åĞ¡æŢřïijŇäijăäÿĂäÿłéąűçžğçŻőå¡ŢåŘ■çżŹåőČãĂĆ äÿŇéĺćæŸŕäÿĂäÿłä¿Ňå■ŘïijŇæ§ěæL'¿çL'źåőŽçŽĎæŰĞäžűåŘ■åźűç■ŤåžŤæL'ĂæIJL'çňęåŘĹæĺąäžűçŽĎæŰ

```
#!/usr/bin/env python3.3
import os

def findfile(start, name):
    for relpath, dirs, files in os.walk(start):
        if name in files:
            full_path = os.path.join(start, relpath, name)
            print(os.path.normpath(os.path.abspath(full_path)))
```

```
if __name__ == '__main__':
    findfile(sys.argv[1], sys.argv[2])
```

ä£Ìå■ŸèĎŽæIJňäÿžæŰĞäžűfindfile.pyïijŇçĎűåŘŐåIJĺåŚ¡äžd'èąŇäÿ■æĽğèąŇåőČãĂĆ
æŇĞåőŽåĹÌåğŃæ§ěæĽ¿çŻőå¡ŢäžěåŕŁå Ř■å■Ůä¡IJäÿžä¡■ç¡őåŔĆæŢřïijŇåęĆäÿŃïijŽ

èőÍèőž

os.walk() æŰźæşŢäÿžæĹŚäžňéĄ■åŐEçŻőå¡ŢæăŚïijŇ
æŕŔæňąè£ŽåĚěäÿĂäÿłçŻőå¡ŢïijŇåőČäijŽè£ŤåŽđäÿĂäÿłäÿĽåĚČçžĎïijŇåŇĚåŘńçŻÿåŕźäžŐæ§ěæĽ¿çŻőå¡Ţ
äžěåŔŁéĆčäÿłçŻőå¡ŢäÿŃéÌćçŽĎæŰĞäžűåŘ■åĹŮèąĺãĂĆ

åŕźäžŐæŕŔäÿłåĚČçžĎïijŇåŔłéIJĂæčĂæţŃäÿĂäÿŃçŻőæăĞæŰĞäžűåŘ■æŸŕåŘęåIJĺæŰĞäžűåĹŮèąĺäÿ■
os.path.join() åŘĹåźűèůŕå¿ĎãĂĆ äÿžäžEéĄ£åĚ■åěĞæĂłçŽĎèůŕå¿ĎåŘ■æŕŤåęĆ ./
./foo//bar ïijŇä¡£çŤĺäžĘåŔęåd'Ûäÿd'äÿłåĞ¡æŢřæÌěä£őæ■čçżŞædIJãĂĆ çňňäÿĂäÿłæŸŕ
os.path.abspath() ,åőČæŐěåŘŮäÿĂäÿłèůŕå¿ĎïijŇåŔŕèČ¡æŸŕçŻÿåŕźèůŕå¿ĎïijŇæIJĂåŘŐè£ŤåŽđçŻÌå
çňňäžŇäÿłæŸŕ os.path.normpath() ïijŇçŤĺæÌěè£ŤåŽđæ■čåÿÿèůŕå¿ĎïijŇåŔŕäžěèğčåĘşåŘŇæŰIJæÌĘã

ař¡çőąè£ŹäÿłèĎŽæIJňçŻÿåŕźäžŐUNIXåźşåŘřäÿŁéÌćçŽĎå¿Ĺåd'Žæ§ěæĽ¿åĚňäžd'æÌěèőšèęAçőĂå■Ţå¿Ì
åźűäÿŤïijŇè£ŸèČ¡å¿Ĺè¡żæÌ¿çŽĎåŁăåĚěåĚűäžŰçŽĎåŁ§èČ¡ãĂĆ
æĹŚäžňåĘ■æijŤçd'žäÿĂäÿłä¿Ńå■ŘïijŇäÿŃéÌćçŽĎåĞ¡æŢřæĽŞå■řæĽĂæIJL'æIJĂè£Śèćńä£őæŤźè£ĞçŽĎæŰ

```
#!/usr/bin/env python3.3

import os
import time

def modified_within(top, seconds):
    now = time.time()
    for path, dirs, files in os.walk(top):
        for name in files:
            fullpath = os.path.join(path, name)
            if os.path.exists(fullpath):
                mtime = os.path.getmtime(fullpath)
                if mtime > (now - seconds):
                    print(fullpath)

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 3:
        print('Usage: {} dir seconds'.format(sys.argv[0]))
        raise SystemExit(1)

    modified_within(sys.argv[1], float(sys.argv[2]))
```

åIJĺæ■d'åĞ¡æŢřçŽĎå§žçąĂäźŃäÿŁïijŇä¡£çŤĺos,os.path,globç■L'çśžäijijæĺąåİŮïijŇä¡ååřśèČ¡åőđçŐřæŻt
åŔřåŔĆèĂČ5.11åřŔèŁĆåŠŇ5.13åřŔèŁĆç■L'çŻÿåĚşçńäèŁĆãĂĆ

13.10 èŕżåŔŰéĚ■ç¡őæŰĞäżű

éŮőécŸ

æĂŐæăůèŕżåŔŰæŹőéĂŽ.iniæäijåijŔçŽĎéĚ■ç¡őæŰĞäżűïij§

èğčåĘşæŰźæąĹ

configparser æĺąåİŮèČ¡èćńçŤĺæİěèŕżåŔŰéĚ■ç¡őæŰĞäżűãĂĆä¿ŃåęĆïijŇåĄĞèő¿ä¡ăæIJĽåęĆäÿŃç

```
; config.ini
; Sample configuration file

[installation]
library=%(prefix)s/lib
include=%(prefix)s/include
bin=%(prefix)s/bin
prefix=/usr/local

# Setting related to debug configuration
[debug]
log_errors=true
show_warnings=False

[server]
port: 8080
nworkers: 32
pid-file=/tmp/spam.pid
root=/www/root
signature:
    ================================
    Brought to you by the Python Cookbook
    ================================
```

äÿŃéİćæŸŕäÿĂäÿłèŕżåŔŰåŠŇæŔŘåŔŰåĚűäÿ■åĂijçŽĎä¿Ńå■ŘïijŽ

```
>>> from configparser import ConfigParser
>>> cfg = ConfigParser()
>>> cfg.read('config.ini')
['config.ini']
>>> cfg.sections()
['installation', 'debug', 'server']
>>> cfg.get('installation','library')
'/usr/local/lib'
>>> cfg.getboolean('debug','log_errors')

True
>>> cfg.getint('server','port')
```

```
8080
>>> cfg.getint('server','nworkers')
32
>>> print(cfg.get('server','signature'))

\==================================
Brought to you by the Python Cookbook
\==================================
>>>
```

åęĆæđIJæIJL'éIJĂèęĄïijŇä¡ăè£ŸèČ¡ä£őæŤźéĚ■ç¡őåźűä¡£çŤÍ `cfg.write()` æŰźæşŢåřĘåĚűåĘŹåŽđåĹřæŰĞäžűäÿ■ãĂĆä¿ŃåęĆïijŽ

```
>>> cfg.set('server','port','9000')
>>> cfg.set('debug','log_errors','False')
>>> import sys
>>> cfg.write(sys.stdout)
```

```
[installation]
library = %(prefix)s/lib
include = %(prefix)s/include
bin = %(prefix)s/bin
prefix = /usr/local

[debug]
log_errors = False
show_warnings = False

[server]
port = 9000
nworkers = 32
pid-file = /tmp/spam.pid
root = /www/root
signature =
          ==================================
          Brought to you by the Python Cookbook
          ==================================
>>>
```

èőÍèőž

éĚ■ç¡őæŰĞäžűä¡IJäÿžäÿĂçğ■åŕřèřżæĂğå¿Ĺåě¡çŽĎæăijåijŔïijŇéİđåÿÿéĂĆçŤÍäžŐå■ŸåĆÍçÍŃåžŔäÿ■çŽ
åIJÍæŕŔäÿłéĚ■ç¡őæŰĞäžűäÿ■ïijŇéĚ■ç¡őæŢřæ■őäijŽèćńåĹEçżĎïijĹæŕŤåęĆä¿Ńå■ŘäÿçŽĎâĂIJinstallationâ
âĂIJdebugâĂİ åŠŇ âĂIJserverâĂİïijL'ãĂĆ æŕŔäÿłåĹEçżĎåIJÍåĚűäÿ■æŇĞåőŽåŕźåžŢçŽĎåŘĎäÿłåŔŸéĞŔåĂ

åŕźäžŐåŔřåőđçŐřåŘŇæăůåŁ§èČ¡çŽĎéĚ■ç¡őæŰĞäžűåŠŇPythonæžŘæŰĞäžűæŸřæIJL'å¿Ĺåđ'ğçŽĎäÿ■
éęŰåĚĹïijŇéĚ■ç¡őæŰĞäžűçŽĎèŕ■æşŢèęĄæŽť'èĞłçŤśäžŽïijŇäÿŃéİćçŽĎèţŃåĂijèŕ■åŘěæŸŕç■L'æŢĹçŽĎïij

```
prefix=/usr/local
prefix: /usr/local
```

éĚ■ç¡őæŰĞäžűäÿ■çŽĎåŘ■å■ŮæŸŕäÿ■åŇžåĹĘåď ğářŕåĘźçŽĎãĂĆä¿ŃåęĆïijŽ

```
>>> cfg.get('installation','PREFIX')
'/usr/local'
>>> cfg.get('installation','prefix')
'/usr/local'
>>>
```

åIJĺèğčæđŘåĂijçŽĎæŮűåĂŹïijŇgetboolean() æŰźæşŢæ§ěæĽ¿äžżä¡ŢåŕŕèąŇçŽĎåĂijãĂĆä¿ŃåęĆ

```
log_errors = true
log_errors = TRUE
log_errors = Yes
log_errors = 1
```

æĹŰèőÿéĚ■ç¡őæŰĞäžűåŠŇPythonäżčçăĄæIJĂåď ğçŽĎäÿ■åŘŇåIJĺäžŐïijŇåőČåźűäÿ■æŸŕäżŐäÿŁèĂŇ
æŰĞäžűæŸŕåőĽèčĚäÿĂäÿłæŢt'ä¡Şèćńèŕżå ŔŰçŽĎãĂĆåęĆæđIJçćřåĹřäžĘåŔŸéĞŔæŻ£æ■ćïijŇåőČåőđéŹĚä
ä¿ŃåęĆïijŇåIJĺäÿŃéÍćè£ŹäÿłéĚ■ç¡őäÿ■ïijŇprefix åŔŸéĞŔåIJĺä¡£çŤĺåőČçŽĎåŔŸéĞŔäźŃåĽ■åŘŐäźŃåŘ

```
[installation]
library=%(prefix)s/lib
include=%(prefix)s/include
bin=%(prefix)s/bin
prefix=/usr/local
```

ConfigParser æIJĽäÿłåőźæŸŞèćńå£¡èğĘçŽĎçĽźæĂğæŸŕåőČèČ¡äÿĂæňąèŕżåŔŰåď ŽäÿłéĚ■ç¡őæŰ
ä¿ŃåęĆïijŇåĄĞèő¿äÿĂäÿłçŤĺæĹůåČŔäÿŃéÍćè£ŹæăůæđĎéĂääžĘäžŰäžňçŽĎéĚ■ç¡őæŰĞäžűïijŽ

```
; ~/.config.ini
[installation]
prefix=/Users/beazley/test

[debug]
log_errors=False
```

èŕżåŔŰè£ŹäÿłæŰĞäžűïijŇåőČåřśèČ¡èů§äźŃåĽ■çŽĎéĚ■ç¡őåŘĹåźűèţůæÍěãĂĆåęĆïijŽ

```
>>> # Previously read configuration
>>> cfg.get('installation', 'prefix')
'/usr/local'

>>> # Merge in user-specific configuration
>>> import os
>>> cfg.read(os.path.expanduser('~/.config.ini'))
['/Users/beazley/.config.ini']

>>> cfg.get('installation', 'prefix')
'/Users/beazley/test'
```

```
>>> cfg.get('installation', 'library')
'/Users/beazley/test/lib'
>>> cfg.getboolean('debug', 'log_errors')
False
>>>
```

äžŤçżĘèğĆåŕ§äÿŃ `prefix` åŔŸéĞŔæŸŕæĂŐæăůèęEçŻŰåĚűäžŰçŻÿåĚşåŔŸéĞŔçŽĎïijŇæŕŤåęĆ `library` çŽĎèő¿åőŽåĂijãĂĆ äžğçŤ§è£Źçğ■çżŞæđIJçŽĎåŐ§åŻăæŸŕåŔŸéĞŔçŽĎæŤźåĘŹéĞĞåŔŰçŽĎæ ä¡ăåŔŕäžěåČŔäÿŃéİćè£Źæăůå ĄŽèŕŢéłŇïijŽ

```
>>> cfg.get('installation','library')
'/Users/beazley/test/lib'
>>> cfg.set('installation','prefix','/tmp/dir')
>>> cfg.get('installation','library')
'/tmp/dir/lib'
>>>
```

æIJĂåŘŐè£ŸæIJL'å¿ĹéĞ■èęĄäÿĂçĆźèęĄæşÍæĎŔçŽĎæŸŕPythonåźűäÿ■èČ¡æŤŕæŇĄ.iniæŰĞäžűåIJĺå çąőä£İä¡ăåůšçżŔåŔĆéŸĚäžĘconfigparseræŰĞæąčäÿ■çŽĎèŕ■æşŢèŕęæČĚäžěåŔŁæŤŕæŇĄçL'źæĂğãĂĆ

13.11 çżŹçőĂå■ŢèĎŽælJňåćđåŁäæŮěå£ŮåŁ§èČ¡

éŮőécŸ

ä¡ăåÿŇæIJŻåIJĺèĎŽæIJňåŠŇçĺŇåžŔäÿ■ărĘèŕŁæŰ■ä£ąæĄŕåĘŹåĚěæŮěå£ŮæŰĞäžűãĂĆ

èğčåĘşæŰźæąĹ

The easiest way to add logging to simple programs is to use the logging module. For example: æL'Şå■řæŮěå£ŮæIJĂçőĂå■ŢæŰźåijŔæŸŕä¡£çŤĺ `logging` æĺąåİŮãĂĆä¿ŃåęĆïijŽ

```
import logging

def main():
    # Configure the logging system
    logging.basicConfig(
        filename='app.log',
        level=logging.ERROR
    )

    # Variables (to make the calls that follow work)
    hostname = 'www.python.org'
    item = 'spam'
    filename = 'data.csv'
    mode = 'r'
```

```
    # Example logging calls (insert into your program)
    logging.critical('Host %s unknown', hostname)
    logging.error("Couldn't find %r", item)
    logging.warning('Feature is deprecated')
    logging.info('Opening file %r, mode=%r', filename, mode)
    logging.debug('Got here')

if __name__ == '__main__':
    main()
```

äýŁéĺćäžŤäýłæŮěå£ŮèřČçŤĺïijĹcritical(), error(), warning(), info(), debug()ïijĽäžěéŹ■åžŕæŰźåijŔèąĺçď žäÿ■åŘŇçŽĎäÿěéĞ■çžğåĹńãĂĆ `basicConfig()` çŽĎ `level` åŔĆæŢřæŸŕäÿĂäýłè£ĞæżďåŹĺãĂĆ æĽĂæIJĽçžğåĹńä¡ŐäžŐæ■ď çžğåĹńçŽĎæŮěå£ŮæűĹæĄŕéČ¡äijŽèćńå£¡çŢěæŐĽãĂĆ æŕŔäýłloggingæŞ■ä¡IJçŽĎåŔĆæŢřæŸŕäÿĂäýłæűĹæĄŕå■ŮçňęäÿšïijŇåŘŐéĺćåĘ■èů§äÿĂäýłæĹŮåď ŽäýłåŔĆ æđĎéĂăæIJĂçžĹçŽĎæŮěå£ŮæűĹæĄŕçŽĎæŮűåĂŹæĹŚäžňä¡£çŤĺäžĘ%æŞ■ä¡IJçňęæÌěæäijåijŔåŇŰæűĹæĄ

è£ŘèąŇè£ŹäýłçÍŃåžŔåŘŐïijŇåIJĺæŰĞäžű `app.log` äÿ■çŽĎåĘĚåőźåžŤèŕěæŸŕäÿŃéĺćè£ŹæăůïijŽ

```
CRITICAL:root:Host www.python.org unknown
ERROR:root:Could not find 'spam'
```

If you want to change the output or level of output, you can change the parameters to the basicConfig() call. For example: åęĆæđIJä¡ăæČşæŤźåŔŸè¿ŞåĞžç■L'çžğïijŇä¡ăåŔŕäžěä£őæŤź `basicConfig()` èřČçŤĺäÿ■çŽĎåŔĆæŢřãĂĆä¿ŃåęĆïijŽ

```
logging.basicConfig(
     filename='app.log',
     level=logging.WARNING,
     format='%(levelname)s:%(asctime)s:%(message)s')
```

æIJĂåŘŐè¿ŞåĞžåŔŸæĹŘåęĆäýŃïijŽ

```
CRITICAL:2012-11-20 12:27:13,595:Host www.python.org unknown
ERROR:2012-11-20 12:27:13,595:Could not find 'spam'
WARNING:2012-11-20 12:27:13,595:Feature is deprecated
```

äýŁéĺćçŽĎæŮěå£ŮéĚ■ç¡őéČ¡æŸŕçąňçijŰçăĄåĹřçÍŃåžŔäÿ■çŽĎãĂĆåęĆæđIJä¡ăæČşä¡£çŤĺéĚ■ç¡őæŰČ åŔŕäžěåČŔäýŃéĺćè£ŹæăůäőæŤź `basicConfig()` èřČçŤĺïijŽ

```
import logging
import logging.config

def main():
    # Configure the logging system
    logging.config.fileConfig('logconfig.ini')
    ...
```

åĹŻåžžäÿĂäýłäýŃéĺćè£ŹæăůçŽĎæŰĞäžűïijŇåŘ■å■ŮåŔń `logconfig.ini` ïijŽ

```
[loggers]
keys=root

[handlers]
keys=defaultHandler

[formatters]
keys=defaultFormatter

[logger_root]
level=INFO
handlers=defaultHandler
qualname=root

[handler_defaultHandler]
class=FileHandler
formatter=defaultFormatter
args=('app.log', 'a')

[formatter_defaultFormatter]
format=%(levelname)s:%(name)s:%(message)s
```

åęĆæđIJä¡ăæČşä£őæŤźéĚ■ç¡őïijŇåŔŕäžěçŻť'æŐěçijŰè¿ŚæŰĞäžűlogconfig.iniå■şåŔŕãĂĆ

èőĺèőž

åř¡çőąåŕźäžŐ `logging` æĺąåİŮèĂŇåůšæIJL'å¿Ĺåď'ŽæŻť'énŸçžğçŽĎéĚ■ç¡őéĂL'éąźïijŇ äÿ■è£Ğè£ŹéĞŇçŽĎæŰźæąĹåŕźäžŐçőĂå■ŢçŽĎçÍŃåžŔåŠŇèĎŽæIJňåůšçżŔèűşåď'§äžĘãĂĆ åŘłæČşåIJÍèřČçŤÍæŮěå£ŮæŞ■ä¡IJåL'■åĚĹæL'ğèąŇäÿŃbasicConfig()åĞ¡æŢřæŰźæşŢïijŇä¡ăçŽĎçÍŃåžŔåřśè

åęĆæđIJä¡ăæČşèęĄä¡ăçŽĎæŮěå£ŮæűĹæĄŕåĘŹåĹřæăĞåĞĘéŤŹèŕŕäÿ■ïijŇèĂŇäÿ■æŸŕæŮěå£ŮæŰĞäż `basicConfig()` æŮűäÿ■äijăæŰĞäžűåŘ■åŔĆæŢřå■şåŔŕãĂĆä¿ŃåęĆïijŽ

```
logging.basicConfig(level=logging.INFO)
```

`basicConfig()` åIJÍçÍŃåžŔäÿ■åŘłèČ¡èćńæL'ğèąŇäÿĂæňąãĂĆåęĆæđIJä¡ăçÍ■åŘŐæČşæŤźåŔŸæŮěå åŕśéIJĂèęĄåĚĹèŐůåŘŰ `root logger` ïijŇçĎűåŘŐçŻť'æŐëä£őæŤźåőČãĂĆä¿ŃåęĆïijŽ

```
logging.getLogger().level = logging.DEBUG
```

éIJĂèęĄåijžěřČçŽĎæŸŕæIJňèŁĆåŘłæŸŕæijŢçd'žäžĘ `logging` æĺąåİŮçŽĎäÿĂäžŻå§žæIJňçŤÍæşŢãĂĆ åőČåŔŕäžěåĄŽæŻť'åd'ŽæŻť'énŸçžğçŽĎåőŽåĹűãĂĆ åĚşäžŐæŮěå£ŮåőŽåĹűåŇŰäÿĂäÿłå¿Ĺåě¡çŽĎèţĎæžŘæŸý Logging Cookbook

13.12 çžŹåĞ¡æŢřåžŞåćđåŁäæŮěå£ŮåŁ§èČ¡

éŮőécŸ

ä¡ăæČşçżŹæ§ŘäÿłåĞ¡æŢřåžŞåćđåŁäæŮěå£ŮåŁ§èČ¡ïijŇä¡ĘæŸŕåŔĹäÿ■èČ¡å¡śåŞ■åĹřéĆčäžŻäÿ■ä¡£çŤĺ

èğčåĘşæŰźæąĹ

åŕźäžŐæČşèęĄæĽğèąŇæŮěå£ŮæŞ■ä¡IJçŽĎåĞ¡æŢřåžŞèĂŇåůšïijŇä¡ăåžŤèŕěåĹŻåżžäÿĂäÿłäÿŞåśđçŽĎ
`logger` åŕźèśąïijŇåźűäÿŤåČŔäÿŃéİćè£ŹæăůåĹİåğŃåŇŰéĚ■ç¡őïijŽ

```
# somelib.py

import logging
log = logging.getLogger(__name__)
log.addHandler(logging.NullHandler())

# Example function (for testing)
def func():
    log.critical('A Critical Error!')
    log.debug('A debug message')
```

ä¡£çŤĺè£ŹäÿłéĚ■ç¡őïijŃézŸèőd'æČĚåĘţäÿŃäÿ■äijŽæĽŞå■řæŮěå£ŮãĂĆä¿ŃåęĆïijŽ

```
>>> import somelib
>>> somelib.func()
>>>
```

äÿ■è£ĞïijŇåęĆæđIJéĚ■ç¡őè£ĞæŮěå£Ůçşżçż§ïijŇéĆčäźĹæŮěå£ŮæűĹæĄŕæĽŞå■řåŕšåijĂåğŃçŤ§æŢĹï

```
>>> import logging
>>> logging.basicConfig()
>>> somelib.func()
CRITICAL:somelib:A Critical Error!
>>>
```

èőĺèőž

éĂŽåÿÿæİěèőšïijŇä¡ääÿ■åžŤèŕěåIJĺåĞ¡æŢřåžŞäžčçăĄäÿ■èĞłåůśéĚ■ç¡őæŮěå£Ůçşżçż§ïijŇæĹŰèĂĚæŸ

èřČçŤĺ `getLogger(__name__)` åĹŻåżžäÿĂäÿłåŠŃèřČçŤĺæĺąåİŮåŘŃåŘ■çŽĎlog-
geræĺąåİŮãĂĆ çŤšäžŐæĺąåİŮéČ¡æŸŕåŤŕäÿĂçŽĎïijŇåŻäæ■d'åĹŻåżžçŽĎloggeräź§åřĘæŸŕåŤŕäÿĂçŽĎãĂĆ

`log.addHandler(logging.NullHandler())` æŞ■ä¡IJåřĘäÿĂäÿłçl'žåd'ĎçŘĘåŹĺçżŚåőŽåĹřåĹ
äÿĂäÿłçl'žåd'ĎçŘĘåŹĺéżŸèőd'äijŽå£¡çŢěèřČçŤĺæL'ĂæIJL'çŽĎæŮěå£ŮæűĹæĄŕãĂĆ
åŻăæ■d'ïijŇåęĆæđIJä¡£çŤĺèŕěåĞ¡æŢřåžŞçŽĎæŮűåĂŹè£ŸæšąæIJL'éĚ■ç¡őæŮěå£ŮïijŇéĆčäźĹåřĘäÿ■äijŽæl

è£ŸæIJL'äÿĂçĆźåřśæŸŕåŕźäžŐåŘĎäÿłåĞ¡æŢřåžŞçŽĎæŮěå£ŮéĚ■ç¡őåŔŕäžěæŸŕçŻÿäžŠçŃňçńŃçŽĎïij
ä¿ŃåęĆïijŇåŕźäžŐåęĆäÿŃçŽĎäžčçăĄïijŽ

```
>>> import logging
>>> logging.basicConfig(level=logging.ERROR)

>>> import somelib
>>> somelib.func()
CRITICAL:somelib:A Critical Error!

>>> # Change the logging level for 'somelib' only
>>> logging.getLogger('somelib').level=logging.DEBUG
>>> somelib.func()
CRITICAL:somelib:A Critical Error!
DEBUG:somelib:A debug message
>>>
```

åIJÍè£ŹéĞŇïijŇæăźæŮěå£ŮèćńéĚ■ç¡őæĹŘäžĚäžĚè¿ŞåĞžERRORæĹŰæŻť'énŸçžğåĹńçŽĎæűĹæĄŕãĂ
äÿ■è£Ğ ïijŇsomelib çŽĎæŮěå£ŮçžğåĹńèćńå■ŢçŃňéĚ■ç¡őæĹŘåŔŕäžěè¿ŞåĞždebugçžğåĹńçŽĎæűĹæĄŕïi
åČŔè£ŹæăůæŻť'æŤźå■ŢçŃňæĺąåİŮçŽĎæŮěå£ŮéĚ■ç¡őåŕźäžŐèřČèŕŢæİěèőšæŸŕå¿ĹæŰźä¿£çŽĎïijŇ
åŻääÿžä¡äæŮäéIJĂåŐżæŻť'æŤźäžžä¡ŢçŽĎåĚĺåšĂæŮěå£ŮéĚ■ç¡őâĂŤâĂŤåŔłéIJĂèęĄä£őæŤźä¡äæČşèęĄæŻ

Logging HOWTO èŕęçżĘäžŃçż■äžĘåęĆä¡ŢéĚ■ç¡őæŮěå£ŮæĺąåİŮåŠŇåĚűäžŰæIJL'çŤĺæŁĂåůğïijŇåŔŕä

13.13 åőđçŐřäÿĂäÿłèőąæŮűåŹĺ

éŮőécŸ

ä¡äæČşèőřå¡ŢçĺŃåžŔæL'ğèąŇåd'ŽäÿłäžžåŀąæL'ĂèŁśèt'źçŽĎæŮűéŮt'

èğčåĘşæŰźæąĹ

time æĺąåİŮåŃĚåŘńå¿Ĺåd'ŽåĞ¡æŢřæİěæL'ğèąŇèů§æŮűéŮt'æIJL'åĚşçŽĎåĞ¡æŢřãĆ
år¡çőąåęĆæ■d'ïijŇéĂŽåÿÿæĹŚäžñäijŽåIJĺæ■d'å§žçąĂäźŃäÿŁæđĎéĂääÿĂäÿłæŻť'énŸçžğçŽĎæŐěåŔčæİěæ

```
import time

class Timer:
    def __init__(self, func=time.perf_counter):
        self.elapsed = 0.0
        self._func = func
        self._start = None

    def start(self):
        if self._start is not None:
            raise RuntimeError('Already started')
        self._start = self._func()
```

```
    def stop(self):
        if self._start is None:
            raise RuntimeError('Not started')
        end = self._func()
        self.elapsed += end - self._start
        self._start = None

    def reset(self):
        self.elapsed = 0.0

    @property
    def running(self):
        return self._start is not None

    def __enter__(self):
        self.start()
        return self

    def __exit__(self, *args):
        self.stop()
```

è£Źäÿłçśżåő Žäź L'äž Ęäÿ Ăäÿłå Ŕŕäżěèćńç Ť Ĺæ Ĺůæăźæ■őéIJĂèęĄåŘŕåŁĺãĂĄåĄIJæ■ćåŠŇéĞ■ç¡őçŽĎèőąą åőČäijŽåIJÍ elapsed åśđæĂğäÿ■èőřå¡ŢæŢt'äÿłæűĹèĂŮæŮűéÚt'ãĂĆ äÿŃéÍćæŸŕäÿĂäÿłä¿Ńå■ŘæÍěæijŤçd'žæĂŐæăůä¡£çŤĺåőČïijŽ

```
def countdown(n):
    while n > 0:
        n -= 1

# Use 1: Explicit start/stop
t = Timer()
t.start()
countdown(1000000)
t.stop()
print(t.elapsed)

# Use 2: As a context manager
with t:
    countdown(1000000)

print(t.elapsed)

with Timer() as t2:
    countdown(1000000)
print(t2.elapsed)
```

èőĺèőž

æIJňèŁĆæŔŘä¿ŻäžĘäÿĂäÿłçőĂå■ŢèĂŇåőđçŤĺçŽĎçśżæİěåőđçŐřæŮűéŮť'èőřå¡ŢäžěåŔŁèĂŮæŮűèőąçő
åŘŇæŮűäź§æŸŕåŕźä¡£çŤĺwithèŕ■åŔëäžěåŔŁäÿŁäÿŃæŰĞçőąçŘĘåŹĺå■Ŕèőőç ŽĎäÿĂäÿłå¿Ĺåě¡çŽĎæijŤçd'ž

åIJĺèőąæŮűäÿ■èęĄèĂČèŹŚäÿĂäÿłåžŢåśĆçŽĎæŮűéŮť'åĞ¡æŢřéŮőécŸãĂĆäÿĂèĹňæİěèŕť'ïijÑ
ä¡£çŤĺ `time.time()` æĹŰ `time.clock()` èőąçőŮçŽĎæŮűéŮť'çš¿åžęåŻăæŞ■ä¡IJçşżç§çŽĎäÿ■åŘŇäijŽ
èĂŇä¡£çŤĺ `time.perf_counter()` åĞ¡æŢřåŔŕäžěçąőä£Ia¡£çŤĺçşżç§äÿŁéĺćæIJĂçš¿çąőçŽĎèőąæŮűåŹ

äÿŁè£řäžčçăĄäÿ■çŤś `Timer` çśżèőřå¡ŢçŽĎæŮűéŮť'æŸŕéŠ§èąĺæŮűéŮť'ïijÑåźűåŇĚåŘńäžĘæL'ĂæIJL'ä
åęĆæđIJä¡ååŔłæČşèőąçőŮèŕěè£ŻçĺŃæL'ĂèŁśèť'źçŽĎCPUæŮűéŮť'ïijÑåžŤèŕěä¡£çŤĺ `time.process_time()` æİěäžčæŻ£ïijŽ

```
t = Timer(time.process_time)
with t:
    countdown(1000000)
print(t.elapsed)
```

`time.perf_counter()` åŠŇ `time.process_time()` éČ¡äijŽè£ŤåŻđářŔæŢřå¡ćåijŔçŽĎçğŠæŢřæŮűéŮť'ãĂĆ åőđéŹĚçŽĎæŮűéŮť'åĂijæšąæIJL'äżżä¡ŢæĎŔäźL'ïijl

æŻť'åđ'ŽåĚşäžŐèőąæŮűåŠŇæĂğèČ¡åĹĘæđŘçŽĎä¿Ńå■ŘèŕůåŔĆèĂČ14.13åřŔèŁĆãĂĆ

13.14 éŹŘåĹűåĘĚå■ŸåŠŇCPUçŽĎä¡£çŤĺéĞŔ

éŮőécŸ

ä¡ăæČşåŕźåIJĺUnixçşżç§äÿŁéİćè£ŘèąŇçŽĎçĺŃåžŔèő¿ç¡őåĘĚå■ŸæĹŰCPUçŽĎä¡£çŤĺéŹŘåĹűãĂĆ

èğčåĘşæŰźæąĹ

`resource` æĺąåİŮèČ¡åŘŇæŮűæL'ğèąŇè£Źäÿd'äÿłäżżåŁąãĂĆä¿ŃåęĆïijŇèęĄéŹŘåĹűCPUæŮűéŮť'ïijl

```
import signal
import resource
import os

def time_exceeded(signo, frame):
    print("Time's up!")
    raise SystemExit(1)

def set_max_runtime(seconds):
    # Install the signal handler and set a resource limit
    soft, hard = resource.getrlimit(resource.RLIMIT_CPU)
    resource.setrlimit(resource.RLIMIT_CPU, (seconds, hard))
    signal.signal(signal.SIGXCPU, time_exceeded)
```

```
if __name__ == '__main__':
    set_max_runtime(15)
    while True:
        pass
```

çÍŃåžŔè£ŘèąŇæŮűĭijŇSIGXCPU ä£ąåŔůåIJÍæŮűéŮt'è£ĞæIJ§æŮűèćńçŤ§æĹŘïijŇçĎűåŘŐæL'ğèąŇæÿ

èęĄéŹŘåĹűåĘĚå■Ÿä¡£çŤĺĭijŇèő¿ç¡őåŔŕä¡£çŤĺçŽĎæĂżåĘĚå■ŸåĂijå■şåŔŕĭijŇåęĆäÿŃïijŽ

```
import resource

def limit_memory(maxsize):
    soft, hard = resource.getrlimit(resource.RLIMIT_AS)
    resource.setrlimit(resource.RLIMIT_AS, (maxsize, hard))
```

åČŔè£Źæăůèő¿ç¡őäžĘåĘĚå■ŸéŹŘåĹűåŘŐĭijŇçÍŃåžŔè£ŘèąŇåĹřæšąæIJL'åd'Žä¡ŹåĘĚå■ŸæŮűäijŽæŁŽ
MemoryError åijĆåÿÿãĂĆ

èőÍèőž

åIJÍæIJňèŁĆä¿Ńå■Řäÿ■ïijŇsetrlimit() åĞ¡æŢřèćńçŤĺæİěèő¿ç¡őçL'źåőŽètĎæžŘäÿŁéİćçŽĎè¡ŕéŹŘ
è¡ŕéŹŘåĹűæŸŕäÿĂäÿłåĂijïijŇå¡ŞèűĚè£Ğè£ŹäÿłåĂijçŽĎæŮűåĂŹæŞ■ä¡IJçşżçż§éĂŽåÿÿäijŽåŔŠéĂĄäÿĂäÿł
çąňéŹŘåĹűæŸŕçŤĺæİěæŇĞåőŽè¡ŕéŹŘåĹűèČ¡èő¿åőŽçŽĎæIJĂåd'ğåĂijãĂĆéĂŽåÿÿæİèèőšïijŇè£ŹäÿłçŤśçşż
ař¡çőąçąňéŹŘåĹűåŔŕäžěæŤźařŘäÿĂçĆźïijŇä¡ĘæŸŕæIJĂåě¡äÿ■èęĄä¡£çŤĺçŤĺæĹůè£ŽçĺŃåŐżä£őæŤźãĂĆ

setrlimit() åĞ¡æŢřè£ŸèČ¡èćńçŤĺæİěèő¿ç¡őå■Řè£ŽçĺŃæŢřéĞŔãĂĄæL'ŞåijĂæŰĞäžűæŢřäžěåŔŁç
æŽt'åd'Žèŕęæ ČĚèŕůåŔĆèĂČ resource æĺąåİŮçŽĎæŰĞæąčãĂĆ

éIJĂèęĄæşĺæĎŔçŽĎæŸŕæIJňèŁĆåĘĚåőźåŔłèČ¡éĂĆçŤĺäžŐUnixçşżçż§ïijŇåźűäÿŤäÿ■ä£İèŕĄæL'ĂæIJL
æŕŤåęĆæĹŚäžňåIJĺæţŃèŕŢçŽĎæŮűåĂŹïijŇåőČèČ¡åIJĺLinuxäÿŁéİćæ■čåÿÿè£ŘèąŇïijŇä¡ĘæŸŕåIJĺOS
XäÿŁå■t'äÿ■èČ¡ãĂĆ

13.15 åŘŕåŁĺäÿĂäÿłWEBæţŔèğĹåŹĺ

éŮőécŸ

ä¡ăæČşéĂŽè£ĞèĎŽæIJňåŘŕåŁĺæţŔèğĹåŹĺåźűæL'ŞåijĂæŃĞåőŽçŽĎURLç¡Śéąţ

èğčåĘşæŰźæąĹ

webbrowser æĺąåİŮèČ¡èćńçŤĺæİěåŘŕåŁĺäÿĂäÿłæţŔèğĹåŹĺïijŇåźűäÿŤäÿŐåźşåŘŕæŮăåĚşãĂĆä¿ŃåęĆ

```
>>> import webbrowser
>>> webbrowser.open('http://www.python.org')
True
>>>
```

åőČäijŽä¡£çŤĺéżŸèőd'æţŔèğĹåŹĺæĽŞåijĂæŇĞåőŽç¡Śéąţã ĂĆåęĆæđIJä¡ăè£ŸæČşåŕźç¡ŚéąţæĽŞåijĂæ

```
>>> # Open the page in a new browser window
>>> webbrowser.open_new('http://www.python.org')
True
>>>

>>> # Open the page in a new browser tab
>>> webbrowser.open_new_tab('http://www.python.org')
True
>>>
```

è£ŹæăůåřśåŔŕäžěæĽŞåijĂäÿĂäÿłæŰřçŽĎæţŔèğĹåŹĺçłŮåŔčæĹŰèĂĚæăĞç■¿ïijŇåŔłèęĄæţŔèğĹåŹĺæŤ

åęĆæđIJä¡ăæČşæŇĞåőŽæţŔèğĹåŹĺçśžåđŃïijŇåŔŕäžěä¡£çŤĺ `webbrowser.get()` åĞ¡æŢřæİěæŇĞåőŽæ§ŘäÿłçĽźåőŽæţŔèğĹåŹĺãĂĆä¿ŃåęĆïijŽ

```
>>> c = webbrowser.get('firefox')
>>> c.open('http://www.python.org')
True
>>> c.open_new_tab('http://docs.python.org')
True
>>>
```

åŕźäžŐæŤŕæŇĄçŽĎæţŔèğĹåŹĺåŘ■çğřåĹŮèąĺåŔŕæ§ěéŸĚ'PythonæŰĞæąč <http://docs.python.org/3/library/webbrowser.html>'_

èőĺèőž

åIJĺèĎŽæIJňäÿ■æĽŞåijĂæţŔèğĹåŹĺæIJL'æŮűåĂŹäijŽå¿ĹæIJL'çŤĺãĂĆä¿ŃåęĆïijŇæ§ŘäÿłèĎŽæIJňæL'
ä¡ăæČşå£ńéĂ§æĽŞåijĂäÿĂäÿłæţŔèğĹåŹĺæİěçąőä£ĺåőČåůšçżŔæ■čåÿÿè£ŘèąŇäžĘãĂĆ
æĹŰèĂĚæŸŕæ§ŘäÿłçĺŃåžŔäžěHTMLç¡ŚéąţæäijåijŔè¿ŞåĞžæŢřæ■őïijŇä¡ăæČşæĽŞåijĂæţŔèğĹåŹĺæ§ěçIJŃ
äÿ■çőąæŸŕäÿŁéĺćåŞłçğ■æČĚåĘţïijŇä¡£çŤĺ `webbrowser` æĺąåİŮéČ¡æŸŕäÿĂäÿłçőÅå■Ţåðđç̀ĺçŽĎèğčåĘşa

CHAPTER 16

çñňå■ĄåŻŻçńäïijŽæţŃèŕŢãĂĄèřČèŕŢåŠŇåijĆåÿÿ

èŕŢéłŇè£ŸæŸŕå¿ĹæčŠçŽĎïijÑä¡ĘæŸŕèřČèŕŢïij§åřśæšąéĆčäźĹæIJL'èűčäžĘãĂĆäžŃåőđæŸŕïijÑåIJÍPytl

Contents:

14.1 æţŃèŕŢstdoutè¿ŞåĞž

éŮőécŸ

ä¡ăçŽĎçÍŃåžŔäÿ■æIJL'äÿłæŰźæşŢäijŽè¿ŞåĞžåĽřæăĞåĞĘè¿ŞåĞžäÿ■ïijĹsys.stdoutïijL'ãĂĆäź§åřśæŸŕè
ä¡ăæČşåĘŹäÿłæţŃèŕŢæÌěèŕĄæŸŐåőČïijÑçżŹåőŽäÿĂäÿłè¿ŞåĚěïijÑçŽÿåžŢçŽĎè¿ŞåĞžèČ¡æ■čåÿÿæŸ¿çď'žå

èğčåĘşæŰźæąĹ

ä¡£çŤÍ `unittest.mock` æĺąåİŮäÿ■çŽĎ `patch()` åĞ¡æŢřïijÑ
ä¡£çŤÍèţůæİěéİđåÿÿçőĂå■ŢïijÑåŔŕäżěäÿžå■ŢäÿłæţŃèŕŢæĺąæÑ§ `sys.stdout`
çĎűåŘŐåŽđæżŽïijÑ åźűäÿŤäÿ■äžğçŢ§åď'ğéĞŔçŽĎäÿť'æŮűåŘŸéĞŔæĹŰåIJÍæţŃèŕŢçŤÍä¿ŃçŽť'æŐěæŽť'éIJ

ä¡IJäÿžäÿĂäÿłä¿Ńå■ŘïijÑæĹŚäżňåIJÍ `mymodule` æĺąåİŮäÿ■åőŽäźL'åęĆäÿŃäÿĂäÿłåĞ¡æŢřïijŽ

```
# mymodule.py

def urlprint(protocol, host, domain):
    url = '{}://{}.{}'.format(protocol, host, domain)
    print(url)
```

éżŸèőď'æČĚåĘţäÿŃåĘĚç¡őçŽĎ `print` åĞ¡æŢřäijŽårĘè¿ŞåĞžåŔŚéĂĄåĹř `sys.stdout` ãĂĆ äÿžäžĘæţŃèŕŢè¿ŞåĞžçIJ§çŽĎåIJÍéĆčéĞŃïijÑä¡ăåŔŕäżěä¡£çŤÍäÿĂäÿłæŽ£èžńårźèśąæİěæĺąæŃ

ä¡£çŤĺ `unittest.mock` æĺąåİŮçŽĎ `patch()` æŰźæşŢåŘŕäžěå¿ĹæŰźä¿£çŽĎåIJĺæţŃèŕŢè£ŘèąŇçŽĎäÿŁ
åźűäÿŤå¡ŞæţŃèŕŢåőŇæĹŘæŮűåĂŹèĞłåŁĺè£ŤåŻđåőČäžňçŽĎåŐ§æIJL'çŁűæĂĄãĂĆäÿŃéĺćæŸŕåŕź
`mymodule` æĺąåİŮçŽĎæţŃèŕŢäžčçăĄïijŽ

```
from io import StringIO
from unittest import TestCase
from unittest.mock import patch
import mymodule

class TestURLPrint(TestCase):
    def test_url_gets_to_stdout(self):
        protocol = 'http'
        host = 'www'
        domain = 'example.com'
        expected_url = '{}://{}.{}\n'.format(protocol, host, domain)

        with patch('sys.stdout', new=StringIO()) as fake_out:
            mymodule.urlprint(protocol, host, domain)
            self.assertEqual(fake_out.getvalue(), expected_url)
```

èőĺèőž

`urlprint()` åĞ¡æŢřæŐěåŔŮäÿL'äÿłåŔĆæŢřijŇæţŃèŕŢæŰźæşŢåijĂåğŃäijŽåĚĹèő¿ç¡őæŕŔäÿĂäÿłåĺ
`expected_url` åŔŸéĞŔèćńèő¿ç¡őæĹŔåŇĚåŘńæIJ§æIJŻçŽĎè¿ŞåĞžçŽĎå■Ůçňęäÿšã ĂĆ

`unittest.mock.patch()` åĞ¡æŢřèćńçŤĺä¡IJäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺïijŇä¡£çŤĺ
`StringIO` åŕźèśąæİěäžčæŻ£ `sys.stdout` . `fake_out`
åŔŸéĞŔæŸŕåIJĺèřěè£ŻçĺŃäÿ■èćńåĹŻåžžçŽĎæĺąæŃ§åŕźèśąãĂĆ åIJĺwith-
èŕ■åŘěäÿ■ä¡£çŤĺåőČåŕäžěæL'ğèąŇåŘĎçğ■æčĂæ§ěãĂĆå¡Şwithèŕ■åŘężŞæİ§æŮűïijŃpatch
äijŽářĘæL'ĂæIJL'äÿIJèě£æĄćåd'■åĹŕæţŃèŕŢäijĂåğŃåL'■çŽĎçŁűæĂĄãĂĆ
æIJL'äÿĂçĆźéIJĂèęĄæşĺæĎŔçŽĎæŸŕæ§ŘäžŻåŕźPythonçŽĎCæL'l'åśŢåŘŕèČ¡äijŽå£¡çŢěæŐL'
`sys.stdout` çŽĎéĚ■ç¡őäžŇçŻť'æŐěåĘŹåĚěåĹřæăĞåĞĘè¿ŞåĞžäÿ■ãĂĆ
éŹŘäžŐçŕĞåźĚïijŇæIJňèŁĆäÿ■äijŽæűL'åŘŁåĹřè£ŹæŰźéĺćçŽĎèőšèğčïijŇåőČéĂĆçŤĺäžŐçžŕPythonäżčçăĄã
åęĆædIJä¡äçIJ§çŽĎéIJĂèęĄåIJĺCæL'l'åśŢäÿ■æ■ŢèŐůI/OïijŇä¡ääŘŕäžěåĚĹæL'ŞåijĂäÿĂäÿłäÿťæŮűæŰĞäžű
æŻť'åd'ŽåĚşäžŐæ■ŢèŐůäžěå■Ůçňęäÿšå¡ćåijŔæ■ŢèŐůI/OåŠŇ `StringIO`
åŕźèśąèŕůåŔĆéŸĚ5.6åŕŘèŁĆãĂĆ

14.2 åIJĺå■ŢåĚČæţŃèŕŢäÿ■çżŹåŕźèśąæL'Şèąěäÿą

éŮőécŸ

ä¡ääĘŹçŽĎå■ŢåĚČæţŃèŕŢäÿ■éIJĂèęĄçżŹæŇĞåőŽçŽĎåŕźèśąæL'Şèąěäÿą ïijŇ
çŤĺæİěæŰ■èĺĂåőČäžňåIJĺæţŃèŕŢäÿ■çŽĎæIJ§æIJŻèąŇäÿžïijĹæŕŤåęĆïijŇæŰ■èĺĂèćńèřČçŤĺæŮűçŽĎåŔĆæ

èğčåĘşæŰźæąĹ

unittest.mock.patch() åĞ¡æŢřåŔŕèćńçŤĺæİěèğčåĘşè£ŹäÿłéŮőécŸãĂĆ patch() è£ŸåŔŕèćńçŤĺä¡IJäÿĂäÿłèčĚéěřåŹĺãĂĄäÿŁäÿŃæŰĞçőąçŘĘåŹĺæĹŰå■ŢçŃňä¡£çŤĺïijŇář¡çőąåźűä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿĂäÿłářĘåőČå¡ŞåĄŽèčĚéěřåŹĺä¡£çŤĺçŽĎä¿Ńå■ŘïijŽ

```
from unittest.mock import patch
import example

@patch('example.func')
def test1(x, mock_func):
    example.func(x)       # Uses patched example.func
    mock_func.assert_called_with(x)
```

åőČè£ŸåŔŕäżěèćńå¡ŞåĄŽäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺïijŽ

```
with patch('example.func') as mock_func:
    example.func(x)      # Uses patched example.func
    mock_func.assert_called_with(x)
```

æIJĂåŘŐïijŇä¡ăè£ŸåŔŕäżěæĽŃåŁĺçŽĎä¡£çŤĺåőČæĽŞèąčäÿĄïijŽ

```
p = patch('example.func')
mock_func = p.start()
example.func(x)
mock_func.assert_called_with(x)
p.stop()
```

åęĆæđIJåŔŕèČ¡çŽĎèŕİïijŇä¡ăèČ¡åd'§åŘäåŁäèčĚéěřåŹĺåŠŇäÿŁäÿŃæŰĞçőąçŘĘåŹĺæİěçżŹåd'Žäÿłářźèś

```
@patch('example.func1')
@patch('example.func2')
@patch('example.func3')
def test1(mock1, mock2, mock3):
    ...

def test2():
    with patch('example.patch1') as mock1, \
         patch('example.patch2') as mock2, \
         patch('example.patch3') as mock3:
    ...
```

èőĺèőž

patch() æŐěåŔŮäÿĂäÿłåůšå■ŸåIJĺářźèśąçŽĎåĚĺèůŕå¿ĎåŘ■ïijŇářĘåĚűæŻ£æ■ćäÿžäÿĂäÿłæŰřçŽĎåĂ åŐ§æİěçŽĎåĂijäijŽåIJĺèčĚéěřåŹĺåĞ¡æŢřæĹŰäÿŁäÿŃæŰĞçőąçŘĘåŹĺåőŇæĹŘåŘŐèĞłåŁĺæĄćåd'■åŻđæİěãéžŸèőd'æČĚåĘţäÿŃïijŇæL'ĂæIJL'åĂijäijŽèćń MagicMock åőđä¿ŃæŻ£äžčãĂĆä¿ŃåęĆïijŽ

```
>>> x = 42
>>> with patch('__main__.x'):
...     print(x)
...
<MagicMock name='x' id='4314230032'>
>>> x
42
>>>
```

äÿ■è£ĞïijŇä¡ăåŕŕäžěéĂŽè£ĞçżŹ patch() æŔŘä¿ŻçňňäžŇäÿłåŔĆæŢřæİěåřĘåĂijæŻ£æ■ćæĹŘäżżä¡Ţ

```
>>> x
42
>>> with patch('__main__.x', 'patched_value'):
...     print(x)
...
patched_value
>>> x
42
>>>
```

èćńçŤĺæİěä¡IJäÿžæŻ£æ■ćåĂijçŽĎ MagicMock åðđä¿ŃèČ¡åđ'§æĺąæŃ§åŕŕèřČçŤĺåŕźèśąåŠŇåðđä¿ŃãĂ äżŰäžňèőřå¡ŢåŕźèśąçŽĎä¡£çŤĺä£ąæĄŕåźűåĚĄèőÿä¡ăæL'ğèąŇæŰ■èĺĂæčĂæ§ěïijŇä¿ŃåęĆïijŽ

```
>>> from unittest.mock import MagicMock
>>> m = MagicMock(return_value = 10)
>>> m(1, 2, debug=True)
10
>>> m.assert_called_with(1, 2, debug=True)
>>> m.assert_called_with(1, 2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File ".../unittest/mock.py", line 726, in assert_called_with
    raise AssertionError(msg)
AssertionError: Expected call: mock(1, 2)
Actual call: mock(1, 2, debug=True)
>>>

>>> m.upper.return_value = 'HELLO'
>>> m.upper('hello')
'HELLO'
>>> assert m.upper.called

>>> m.split.return_value = ['hello', 'world']
>>> m.split('hello world')
['hello', 'world']
>>> m.split.assert_called_with('hello world')
>>>

>>> m['blah']
<MagicMock name='mock.__getitem__()' id='4314412048'>
```

```
>>> m.__getitem__.called
True
>>> m.__getitem__.assert_called_with('blah')
>>>
```

äÿĂèĹňæÌěèőšïijŇè£ŹäžŻæŞ■ä¡IJäijŽåIJĺäÿĂäÿłå■ŢåĚČæţŃèŕŢäÿ■åőŇæĹŘãĂĆä¿ŃåęĆïijŇåĄĞèő¿ä¡ä

```
# example.py
from urllib.request import urlopen
import csv

def dowprices():
    u = urlopen('http://finance.yahoo.com/d/quotes.csv?s=@^DJI&f=sl1
↪')
    lines = (line.decode('utf-8') for line in u)
    rows = (row for row in csv.reader(lines) if len(row) == 2)
    prices = { name:float(price) for name, price in rows }
    return prices
```

æ■čåÿÿæÌěèőšïijŇè£ŹäÿłåĞ¡æŢřäijŽä¡£çŤĺ urlopen() äžŐWebäÿŁéÍćèŐůåŔŰæŢřæ■őåźűèğčædŘåőČãĂĆ åIJĺå■ŢåĚČæţŃèŕŢäÿ■ïijŇä¡ăåŔŕäžèçżŹåőČäÿĂäÿłécĎåĚĹåőŽ

```
import unittest
from unittest.mock import patch
import io
import example

sample_data = io.BytesIO(b'''\
"IBM",91.1\r
"AA",13.25\r
"MSFT",27.72\r
\r
''')

class Tests(unittest.TestCase):
    @patch('example.urlopen', return_value=sample_data)
    def test_dowprices(self, mock_urlopen):
        p = example.dowprices()
        self.assertTrue(mock_urlopen.called)
        self.assertEqual(p,
                         {'IBM': 91.1,
                          'AA': 13.25,
                          'MSFT' : 27.72})

if __name__ == '__main__':
    unittest.main()
```

æIJňä¿Ńäÿ■ïijŇä¡■äžŐ example æĺąåİŮäÿ■çŽĎ urlopen() åĞ¡æŢřèćńäÿĂäÿłæĺąæŃ§åŕźèśąæŻ£äżčïijŇ èŕěåŕźèśąäijŽè£ŤåŻđäÿĂäÿłåŇĚåŘńæţŃèŕŢæŢřæ■őçŽĎ ByteIO().

è£ŸæIJL'äÿĂçĆźïijŇåIJÍæĽŞèąěäÿĄæŮűæĹŚäžňä¡£çŤÍäžĘ `example.urlopen` æİěäžčæŻ£ `urllib.request.urlopen` ãĂĆ å¡Şä¡ăåĹŻåžžèąěäÿĄçŽĎæŮűåĂŹïijŇä¡ăå£Ěéążä¡£çŤÍåőČäžňåIJÍæţŃèŕŢäžčçăĄäÿ■çŽĎåŘ■çğřãĂĆ çŤśäžŐæţŃèŕŢäžčçăĄä¡£çŤÍäžĘ `from urllib.request import urlopen` ,éĆčäźĹ `dowprices()` åĞ¡æŢř äÿ■ä¡£çŤÍçŽĎ `urlopen()` åĞ¡æŢřåőđéŹĚäÿŁåŕśä¡■äžŐ `example` æÍąåİŮäžĘãĂĆ

æIJňèŁĆåőđéŹĚäÿŁåŔłæŸŕåŕź `unittest.mock` æÍąåİŮçŽĎäÿĂæňąæţĚåřİè¿Ďæ■ćãĂĆ æŻťåď'ŽæŻťéńŸçžğçŽĎçL'źæĂğïijŇèŕůåŔĆèĂČ åőŸæŰźæŰĞæąč

14.3 åIJÍå■ŢåĚČæţŃèŕŢäÿ■æţŃèŕŢåijĆåÿÿæČĚåĘţ

éŮőécŸ

ä¡ăæČşåĘŹäÿłæţŃèŕŢçŤÍä¿ŃæİěåĞEçąőçŽĎåĹd'æŰ■æ§ŘäÿłåijĆåÿÿæŸŕåŘęècńæŁŻåĞžãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐåijĆåÿÿçŽĎæţŃèŕŢåŔŕä¡£çŤÍ `assertRaises()` æŰźæşŢãĂĆ ä¿ŃåęĆïijŇåęĆæđIJä¡ăæČşæţŃèŕŢæ§ŘäÿłåĞ¡æŢřæŁŻåĞžäžĘ `ValueError` åijĆåÿÿïijŇåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
import unittest

# A simple function to illustrate
def parse_int(s):
    return int(s)

class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        self.assertRaises(ValueError, parse_int, 'N/A')
```

åęĆæđIJä¡ăæČşæţŃèŕŢåijĆåÿÿçŽĎåĚůä¡ŞåĂijïijŇéIJĂèęAçŤÍåĹřåŔęåď'ŰäÿĂçğ■æŰźæşŢïijŽ

```
import errno

class TestIO(unittest.TestCase):
    def test_file_not_found(self):
        try:
            f = open('/file/not/found')
        except IOError as e:
            self.assertEqual(e.errno, errno.ENOENT)

        else:
            self.fail('IOError not raised')
```

èőĺèőž

assertRaises() æŰźæşŢäÿžæţŃèŕŢåijĆåÿÿå■ŸåIJĺæĂğæŔŘä¿ŻäžĘäÿĂäÿłçőÄä¿£æŰźæşŢãĂĆ äÿĂäÿłåÿÿèğĄçŽĎéŹůéŸśæŸŕæĽŃåŁĺåŐżè£ŻèąŇåijĆåÿÿæčĂæţŇãĂĆæŕŤåęĆïijŽ

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        try:
            r = parse_int('N/A')
        except ValueError as e:
            self.assertEqual(type(e), ValueError)
```

è£Źçğ■æŰźæşŢçŽĎéŮőécŸåIJĺäžŐåőČå¿ĹåőźæŸ§éĄŮæijŔåĚűäżŰæČĚåĘţijjŇæŕŤåęĆæšąæIJL'äżżä¡Ť éĆčäźĹä¡ãè£Ÿå¿ŮéIJĂèęĄåćđåŁãåŔęåd'ŰçŽĎæčĂæţŃè£ĞçÍŇïijŇåęĆäÿŃéÍćè£ŹæăůïijŽ

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        try:
            r = parse_int('N/A')
        except ValueError as e:
            self.assertEqual(type(e), ValueError)
        else:
            self.fail('ValueError not raised')
```

assertRaises() æŰźæşŢäijŽåd'ĎçŘĘæL'ĂæIJL'çżĘèŁĆïijŇåŽäæ■d'ä¡ääžŤèŕěä¡£çŤĺåőČãĂĆ

assertRaises() çŽĎäÿĂäÿłçijžçĆźæŸŕåőČæţŃäÿ■äžĘåijĆåÿÿåĚůä¡ŞçŽĎåĂijæŸŕåd'ŽåřŠãĂĆ äÿžäžĘæţŃèŕŢåijĆåÿÿåĂijïijŇåŔŕäżěä¡£çŤÍ assertRaisesRegex() æŰźæşŢïijŇ åőČåŔŕåŘŇæŮűæţŃèŕŢåijĆåÿÿçŽĎå■ŸåIJĺäžěåŔŁéĂŽè£Ğæ■čåĹŹåijŔåŇźéĚ■åijĆåÿÿçŽĎå■Ůçňęäÿšèąĺçd

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        self.assertRaisesRegex(ValueError, 'invalid literal .*',
                               parse_int, 'N/A')
```

assertRaises() åŠŇ assertRaisesRegex() è£ŸæIJL'äÿĂäÿłåőźæŸŞå£¡çŢěçŽĎåIJřæŰźåřśæŸŕåőČäžňè£ŸèČ¡èćńå¡ŞåĄŽäÿŁäÿŃæŰĞçőąçŘĘåŹĺä¡£çŤĺ

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        with self.assertRaisesRegex(ValueError, 'invalid literal .*
↪'):
            r = parse_int('N/A')
```

ä¡Eä¡äçŽĎæţŃèŕŢæűL'åŔŁåĹřåd'Žäÿłæl'ğèąŇæ■ěéłd'çŽĎæŮűåĂŹè£Źçğ■æŰźæşŢåřšå¿ĹæIJL'çŤĺäžĘ

14.4 åřĘæţŃèŕŢè¿ŞåĞžçŤĺæŮěå£Ùèőřå¡ŢåĹřæŰĞäžűäÿ■

éŮőécŸ

ä¡ăåÿŇæIJŻåřĘå■ŢåĚČæţŃèŕŢçŽĎè¿ŞåĞžåĘŹåĹřåĹřæ§ŘäÿłæŰĞäžűäÿ■åŐżïijŇèĂŇäÿ■æŸŕæĽŞå■řå

èğčåĘşæŰźæąĹ

è£ŘèąŇå■ŢåĚČæţŃèŕŢäÿĂäÿłåÿÿèğĄæŁĂæIJŕåřśæŸŕåIJĺæţŃèŕŢæŰĞäžűåžŢéČĺåŁăåĚěäÿŃéIćè£Źæő

```
import unittest

class MyTest(unittest.TestCase):
    pass

if __name__ == '__main__':
    unittest.main()
```

è£ŹæăůçŽĎèŕÍæţŃèŕŢæŰĞäžűåřśæŸŕåŔŕæĽğèąŇçŽĎïijŇåźűäÿŤäijŽåřĘè£ŘèąŇætŃèŕŢçŽĎçżŞæđIJæ
åęĆæđIJä¡ăæČşéĞ■åőŽåŘŚè¿ŞåĞžïijŇåřśéIJĂèęĄåČŔäÿŃéIćè£Źæăůä£őæŤź `main()`
åĞ¡æŢřïijŽ

```
import sys

def main(out=sys.stderr, verbosity=2):
    loader = unittest.TestLoader()
    suite = loader.loadTestsFromModule(sys.modules[__name__])
    unittest.TextTestRunner(out,verbosity=verbosity).run(suite)

if __name__ == '__main__':
    with open('testing.out', 'w') as f:
        main(f)
```

èőÍèőž

æIJňèŁĆæĎ§åĚt'èűčçŽĎéČĺåĹĘåźűäÿ■æŸŕåřĘætŃèŕŢçżŞæđIJéĞ■åőŽåŘŚåĹřäÿĂäÿłæŰĞäžűäÿ■ïijŇ
èĂŇæŸŕéĂŽè£ĞèŁŹæăůåĄŽåŘŚä¡ăåśŢçd'žäžĘ `unittest`
æÍąåİŮäÿ■äÿĂäžŻåĂijå¿ŮåĚşæşÍçŽĎåĘĚéČĺåůěä¡IJåŐ§çŘĘãĂĆ

`unittest` æÍąåİŮéęŰåĚĹäijŽçżĎèčĚäÿĂäÿłætŃèŕŢåěŮäžűãĂĆ
è£ŹäÿłætŃèŕŢåěŮäžűåŇĚåŘńäžĘä¡ăåőŽäźL'çŽĎåŘĎçğ■æŰźæşŢãĂĆäÿĂæŮęåěŮäžűçżĎèčĚåőŇæĹŘïijŇå

è£Źäÿd'æ■ěæŸŕåĹĘåijĂçŽĎïijŇ`unittest.TestLoader`
åőđä¿ŃèćńçŤĺæİěçżĎèčĚætŃèŕŢåěŮäžűãĂĆ `loadTestsFromModule()`
æŸŕåőČåőŽäźL'çŽĎæŰźæşŢäźŃäÿĂïijŇçŤĺæİěæŤűéŻĘætŃèŕŢçŤĺä¿ŃãĂĆ åőČäijŽäÿž
`TestCase` çśżæĽńæŔŘæ§ŘäÿłæĺąåİŮåžűåřĘåĚűäÿ■çŽĎætŃèŕŢæŰźæşŢæŘŘåŘŮåĞžæİěãĂĆ
åęĆæđIJä¡ăæČşè£ŻèąŇçżĘçšŠåžęçŽĎæŐğåĽűïijŇ åŔŕäžěä¡£çŤÍ

loadTestsFromTestCase() æŰźæşŢæÌěäžŐæ§ŘäÿłçžğæĽ£TestCaseçŽĎçśžäÿ■æŔŘåŔŰæţŃèŕŢæŰźæ
TextTestRunner çśžæŸŕäÿĂäÿłæţŃèŕŢè£ŘèąŇçśžçŽĎä¿Ńå■ŘïijŇ
è£ŹäÿłçśžçŽĎäÿžèęĄçŤĺéĂŤæŸŕæĽğèąŇæ§ŘäÿłæţŃèŕŢåěŮäžűäÿ■åŇĚåŘńçŽĎæţŃèŕŢæŰźæşŢãĂĆ
è£Źäÿłçśžèů§æĽğèąŇ unittest.main() åĞ¡æŢřæĽĂä¡£çŤĺçŽĎæţŃèŕŢè£ŘèąŇåŹĺæŸŕäÿĂæăůçŽĎãĂĆ
äÿ■è£ĞïijŇæĹŚäžňåIJĺè£ŹéĞŇåŕźåőČè£ŻèąŇäžĘäÿĂäžŻåĹŮåžŢåśĆéĚ■ç¡őïijŇåŇĚæŃňè¿ŞåĞžæŰĞäžűåŠ
åř¡çőąæIJňèŁĆä¿Ńå■ŘäžčçăĄå¿ĹåřŚïijŇä¡ĘæŸŕèČ¡æŇĞåŕijä¡ăåęĆä¡Ţåŕź
unittest æąĘæđűè£ŻèąŇæŻť è£ŻäÿĂæ■ěçŽĎèĞłåőŽäźĽãĂĆ
èęĄæČşèĞłåőŽäźĽæţŃèŕŢåěŮäžűçŽĎèčĚéĚ■æŰźåijŔïijŇä¡ăåŔŕäžěåŕź TestLoader
çśžæĽğèąŇæŻť åď ŽçŽĎæŞ■ä¡IJãĂĆ äÿžäžĘèĞłåőŽäźĽæţŃèŕŢè£ŘèąŇïijŇä¡ăåŔŕäžěæđĎéĂääÿĂäÿłèĞłåůś
TextTestRunner çŽĎåŁ§èČ¡ãĂĆ èĂŇè£ŹäžŻåůšçżŔèűĚåĞžäžĘæIJňèŁĆçŽĎèŇČåŻť ãĂĆunittest
æĺąåİŮçŽĎæŰĞæąčářźåžŢåśĆåőđçŐřåŐ§çŘĘæIJĽæŻť æůšåĚěçŽĎèőšèğčïijŇåŔŕäžěåŐžçIJŃçIJŃãĂĆ

14.5 å£¡çŢěæĹŰæIJ§æIJŻæţŃèŕŢåď śèť ě

éŮőéćŸ

ä¡ăæČşåIJĺå■ŢåĚČæţŃèŕŢäÿ■å£¡çŢěæĹŰæăĞèőŕæ§ŘäžŻæţŃèŕŢäijŽæŇĽçĚğécĎæIJ§è£ŘèąŇåď śèť ěã

èğčåĘşæŰźæąĹ

unittest æĺąåİŮæIJĽèčĚéěřåŹĺåŔŕçŤĺæÌěæŐğåĹűåŕźæŇĞåőŽæţŃèŕŢæŰźæşŢçŽĎåď ĎçŘĘïijŇä¿Ń

```
import unittest
import os
import platform

class Tests(unittest.TestCase):
    def test_0(self):
        self.assertTrue(True)

    @unittest.skip('skipped test')
    def test_1(self):
        self.fail('should have failed!')

    @unittest.skipIf(os.name=='posix', 'Not supported on Unix')
    def test_2(self):
        import winreg

    @unittest.skipUnless(platform.system() == 'Darwin', 'Mac␣
↪specific test')
    def test_3(self):
        self.assertTrue(True)

    @unittest.expectedFailure
    def test_4(self):
        self.assertEqual(2+2, 5)
```

```
if __name__ == '__main__':
    unittest.main()
```

åęĆæđIJä¡ăåIJĺMacäÿŁè£ŘèąŇè£Źæőţäžčçă Ąïij Ňä¡ăäijŽå¿ŮåĹřåęĆäÿŃè¿ŞåĞžïijŽ

```
bash % python3 testsample.py -v
test_0 (__main__.Tests) ... ok
test_1 (__main__.Tests) ... skipped 'skipped test'
test_2 (__main__.Tests) ... skipped 'Not supported on Unix'
test_3 (__main__.Tests) ... ok
test_4 (__main__.Tests) ... expected failure

----------------------------------------------------------------------
↪--
Ran 5 tests in 0.002s

OK (skipped=2, expected failures=1)
```

èőĺèőž

skip() èčĚéěřåŹĺèČ¡èćńçŤĺæİěå£¡çŢěæ§Řäÿłä¡ääÿ■æČşè£ŘèąŇçŽĎæţŃèŕŢãĂĆ skipIf() åŠŇ skipUnless() åŕźäžŐä¡ăåŔłæČşåIJĺæ§ŘäÿłçĽźåőŽåźşåŘřæĹŰPythonçĽĹæIJňæĹŰåĚû ä¡£çŤĺ @expected çŽĎåď´śèť ěèčĚéěřåŹĺæİěæăĞèőřéĆčäžŻçąőåőŽäijŽåď´śèť ěçŽĎæţŃèŕŢïijŇåźűäÿŤåŕźè£

å£¡çŢěæŰźæşŢçŽĎèčĚéěřåŹĺè£ŸåŘŕäžěèćńçŤĺæİěèčĚéěřæŢť äÿłæţŃèŕŢçśžïijŇærŤåęĆïijŽ

```
@unittest.skipUnless(platform.system() == 'Darwin', 'Mac specific␣
↪tests')
class DarwinTests(unittest.TestCase):
    pass
```

14.6 åď ĎçŘĘåď ŽäÿłåijĆåÿÿ

éŮőéćŸ

ä¡ăæIJL'äÿĂäÿłäžčçăĄçL'ĞæőţåŔŕèČ¡äijŽæŁŻåĞžåď Žäÿłäÿ■åŘŇçŽĎåijĆåÿÿïijŇæĂŐæăůæL'■èČ¡äÿ■

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăåŔŕäžěçŤĺå■Ţäÿłäžčçă ĄåİŮåď ĎçŘĘäÿ■åŘŇçŽĎåijĆåÿÿïijŇåŔŕäžěåřĘåőČäžňæŤ¿åĚěäÿĂä

```
try:
    client_obj.get_url(url)
```

```
except (URLError, ValueError, SocketTimeout):
    client_obj.remove_url(url)
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇåĚČçěŰäÿ■äżżä¡ŢäÿĂäÿłåijĆåÿÿåŔŚçŤ§æŮűéČ¡äijŽæĽğèąŇ `remove_url()` æŰźæşŢãĂĆ åęĆædIJä¡ăæČşåŕźåĚűäÿ■æ§Řäÿłåij Ćåÿÿè£ŻèąŇäÿ■åŘŇçŽĎåď ĎçŘĘïijŇåF `except` èŕ■åŔěäÿ■ïijŽ

```
try:
    client_obj.get_url(url)
except (URLError, ValueError):
    client_obj.remove_url(url)
except SocketTimeout:
    client_obj.handle_url_timeout(url)
```

å¿Ĺåď ŽçŽĎåijĆåÿÿäijŽæIJĽåśĆçžğåĚşçşżïijŇåŕźäžŐè£Źçğ■æČĚåĘţïijŇä¡ăåŔŕèČ¡ä¡£çŤĺåőČäżňçŽĎäÿ

```
try:
    f = open(filename)
except (FileNotFoundError, PermissionError):
    pass
```

åŔŕäżěèćńéĞ■åĘŹäÿžïijŽ

```
try:
    f = open(filename)
except OSError:
    pass
```

`OSError` æŸŕ `FileNotFoundError` åŠŇ `PermissionError` åijĆåÿÿçŽĎå§žçśżãĂĆ

èőĺèőž

åř¡çőąåď ĎçŘĘåď ŽäÿłåijĆåÿÿæIJňèžńåźűæšąäżĂäźĹçĽźæőŁçŽĎïijŇäÿ■è£Ğä¡ăåŔŕäżěä¡£çŤĺ `as` åĚşéŤőå■ŮæİěèŐůå¿Ůèćńæ ŁŻåĞžåijĆåÿÿçŽĎåijŢçŤĺïijŽ

```
try:
    f = open(filename)
except OSError as e:
    if e.errno == errno.ENOENT:
        logger.error('File not found')
    elif e.errno == errno.EACCES:
        logger.error('Permission denied')
    else:
        logger.error('Unexpected error: %d', e.errno)
```

è£Źäÿłä¿Ńå■Řäÿ■ïijŇ `e` åŔŸéĞŔæŇĞåŘŚäÿĂäÿłèćńæŁŻåĞžçŽĎ `OSError` åijĆåÿÿåőđä¿ŃãĂĆ è£ŹäÿłåIJĺä¡ăæČşæŻť è£ŻäÿĂæ■ěåĹĘædŘè£ŹäÿłåijĆåÿÿçŽĎæŮűåĂŹäijŽå¿ĹæIJĽçŤĺï

åŘŇæŮűè£ŸèęĄæşĺæĎŔçŽĎæŮűåĂŹ except èí■åŔěæŸŕéąžåžŔæčĂæ§ěçŽĎïijŇçňňäÿĂäÿłåŇźéĚ■ç ä¡ääŔŕäžěå¿ĹåőźæŸ§çŽĎæđĎéĂääď'Žäÿł except åŘŇæŮűåŇźéĚ■çŽĎæČĚå¡ćïijŇæŕŤåęĆïijŽ

```
>>> f = open('missing')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: 'missing'
>>> try:
...     f = open('missing')
... except OSError:
...     print('It failed')
... except FileNotFoundError:
...     print('File not found')
...
It failed
>>>
```

è£ŹéĞŇçŽĎ FileNotFoundError èí■åŔěåźűæšąæIJL'æL'ğèąŇçŽĎåŐ§åŻăæŸŕ OSError æŻt'äÿĂèĹňïijŇäőČåŔŕåŇźéĚ■ FileNotFoundError åijĆåÿÿïijŇ äžŐæŸŕåřśæŸŕçňňäÿĂäÿłåŇźéĚ■çŽĎãĂĆ åIJĺèřČèŕŢçŽĎæŮűåĂŹïijŇåęĆædIJä¡ääŕźæ§ŘäÿłçL'źåőŽåijĆåÿÿ ä¡ääŔŕäžěéĂŽè£Ğæ§ěçIJŃèŕěåijĆåÿÿçŽĎ __mro__ åśđæĂğæİěå£ńéĂ§æţŔèğĹãĂĆæŕŤåęĆïijŽ

```
>>> FileNotFoundError.__mro__
(<class 'FileNotFoundError'>, <class 'OSError'>, <class 'Exception'>
 ↪,
 <class 'BaseException'>, <class 'object'>)
>>>
```

äÿŁéİćåĹŮèąĺäÿ■äżżä¡ŢäÿĂäÿłçŽt'åĹř BaseException çŽĎçśžéČ¡èČ¡èćńçŢĺäžŐ except èí■åŔěãĂĆ

14.7 æ■ŢèŐůæL'ĂæIJL'åijĆåÿÿ

éŮőécŸ

æĂŐæăůæ■ŢèŐůäžčçăĄäÿ■çŽĎæL'ĂæIJL'åijĆåÿÿïij§

èğčåĘşæŰźæąĹ

æČşèęĄæ■ŢèŐůæL'ĂæIJL'çŽĎåijĆåÿÿïijŇåŕŕäžěçŽt'æŐěæ■ŢèŐů Exception å■şåŔŕïijŽ

```
try:
    ...
except Exception as e:
    ...
    log('Reason:', e)        # Important!
```

è£ŹäÿłåřĘäijŽæ■ŢèŐůéŹd'äžĘ `SystemExit` ãĂĄ `KeyboardInterrupt` åŠŇ `GeneratorExit` äźŃåd'ŰçŽĎæĽĂæIJL'åijĆåÿÿãĂĆ åęĆædIJä¡ãè£ŸæČşæ■ŢèŐůè£ŹäÿĽäÿłåijĆåÿÿïijŇåřĘ `Exception` æŤźæĹŘ `BaseException` å■şåŔŕãĂĆ

èőĺèőž

æ■ŢèŐůæĽĂæIJL'åijĆåÿÿéĂŽåÿÿæŸŕçŤśäžŐçĺŃåžŔåŚŸåIJĺæ§ŘäžŽåd'■æİĆæŞ■ä¡IJäÿ■åźűäÿ■èČ¡èőř åęĆædIJä¡ãäÿ■æŸŕå¿ĹçżĘå£ČçŽĎäžžïijŇè£ŹäźæŸŕçijŰåĘŹäÿ■æŸŞèřČèŕŢäžčçăĄçŽĎäÿĂäÿłçőĂå■ŢæŰź

æ■čåŽäåęĆæ■d'ïijŇåęĆædIJä¡ãéĂL'æŃl'æ■ŢèŐůæĽĂæIJL'åijĆåÿÿïijŇéĆčäźĹåIJĺæ§ŘäÿłåIJřæŰźïijĹæ åęĆædIJä¡ãæšąæIJL'è£Źæăůå˛ŽïijŇæIJL'æŮűåĂŹä¡ăçIJŃåĹřåijĆåÿÿæĽŞå■řæŮűåŔŕèČ¡æŠÿäÿ■çİĂåd'ť'èĎ

```
def parse_int(s):
    try:
        n = int(v)
    except Exception:
        print("Couldn't parse")
```

èŕŢçİĂè£ŘèąŇè£ŹäÿłåĞ¡æŢřïijŇçżŞædIJåęĆäÿŃïijŽ

```
>>> parse_int('n/a')
Couldn't parse
>>> parse_int('42')
Couldn't parse
>>>
```

è£ŹæŮűåĂŹä¡ăåřšäijŽæŇăåd'ť'æČşïijŽâĂIJè£ŹåŠŃåŽđäžŃåŢŁïij§âĂİ åĄĞåęĆä¡ăåČŔäÿŃéİćè£ŹæăůéĞ■åĘŹè£ŹäÿłåĞ¡æŢřïijŽ

```
def parse_int(s):
    try:
        n = int(v)
    except Exception as e:
        print("Couldn't parse")
        print('Reason:', e)
```

è£ŹæŮűåĂŹä¡ăèČ¡èŐůåŔŰåęĆäÿŃè¿ŞåĞžïijŇæŇĞæŸŐäžĘæIJL'äÿłçijŰçĺŃéŤŹèŕřïijŽ

```
>>> parse_int('42')
Couldn't parse
Reason: global name 'v' is not defined
>>>
```

å¿ĹæŸŐæŸ¿ïijŇä¡ăåžŤèŕěår¡åŔŕèČ¡årĘåijĆåÿÿåd'ĎçŘĘåŹĺåőŽäźL'çŽĎçš¿åĞĘäÿĂäžŽãĂĆ äÿ■è£ĞïijŇèęĄæŸŕä¡ăå£Ěéążæ■ŢèŐůæĽĂæIJL'åijĆåÿÿïijŇçąőä£İæL'Şå■řæ■čçąőçŽĎèŕŁæŰ■ä£ąæĄŕæĹŰå

14.8 åĹŻåžžèĞłåőŽäźĽåijĆåÿÿ

éŮőécŸ

åIJĺä¡ăæđĎåžžçŽĎåžŤçŤĺçĺŃåžŕäÿ■ïijŇä¡ăæČşářĘåžŢåśĆåijĆåÿÿåŇĚècĚæĹŕèĞłåőŽäźĽçŽĎåijĆåÿÿ

èğčåĘşæŰźæąĹ

åĹŻåžžæŰřçŽĎåijĆåÿÿå¿ĹçőĂå■ŢâĂŤâĂŤåőŽäźĽæŰřçŽĎçśżïijŇèőľåőČçżğæĽ£èĞł
`Exception` ïijĹæĹŰèĂĚæŸŕäžżä¡ŢäÿĂäÿłåůšå■ŸåIJĺçŽĎåijĆåÿÿçśżåđŇïijĽãĂĆ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăçijŰåĘŹç¡ŚçżIJçŻÿåĚşçŽĎçĺŃåžŕïijŇä¡ăåŕŕèČ¡äijŽåőŽäźĽäÿĂäžŻçśżäijijåęĆäÿŃçŽ

```
class NetworkError(Exception):
    pass

class HostnameError(NetworkError):
    pass

class TimeoutError(NetworkError):
    pass

class ProtocolError(NetworkError):
    pass
```

çĎűåŘŐçŤĺæĹůåřśåŕŕäžěåČŕéĂŽåÿÿéĆčæăůä¡£çŤĺè£ŹäžŻåijĆåÿÿäžĘïijŇä¿ŃåęĆïijŽ

```
try:
    msg = s.recv()
except TimeoutError as e:
    ...
except ProtocolError as e:
    ...
```

èőĺèőž

èĞłåőŽäźĽåijĆåÿÿçśżåžŤèŕěæĂżæŸŕçżğæĽ£èĞłåĘĚç¡őçŽĎ `Exception`
çśżïijŇ æĹŰèĂĚæŸŕçżğæĽ£èĞłéĆčäžŻæIJňèžńåřśæŸŕäžŐ `Exception`
çżğæĽ£èĂŇæİěçŽĎçśżãĂĆ åř¡çőąæĽ'ĂæIJL'çśżåŘŇæŮűäź§çżğæĽ£èĞł `BaseException`
ïijŇä¡Ęä¡ääÿ■åžŤèŕěä¡£çŤĺè£ŹäÿłåšžçśżæİěåőŽäźĽæŰřçŽĎåijĆåÿÿãĂĆ `BaseException`
æŸŕäÿžçşżçż§éĂĂåĞžåijĆåÿÿèĂŇä£ĺçŢŹçŽĎïijŇærŤåęĆ `KeyboardInterrupt` æĹŰ
`SystemExit` äžěåŘŁåĚűäžŰéĆčäžŻäijŽçżŹåžŤçŤĺåŚéĂĄä£ąåŕůèĂŇéĂĂåĞžçŽĎåijĆåÿÿãĂĆ
åŻăæ■ď'ïijŇæ■ŢèŐůè£ŹäžŻåijĆåÿÿæIJňèžńæšąäžĂäźĹæĎŕäźĽ'ãĂĆ
è£ŹæăůçŽĎèŕİïijŇåĄĞåęĆä¡ăçżğæĽ'£ `BaseException` åŕŕèČ¡äijŽåŕijèĞt'ä¡ăçŽĎèĞłåőŽäźĽåijĆåÿÿäÿ■ä

åIJĺçĺŃåžŕäÿ■åijŢåĚěèĞłåőŽäźĽåijĆåÿÿåŕŕäžěä¡£å¿Ůä¡ăçŽĎäžčçăĄæŻt'åĚůåŕŕèŕżæĂğïijŇèČ¡æÿĚæ
è£ŸæIJL'äÿĂçğ■èő¿èőąæŸŕårĘèĞłåőŽäźĽåijĆåÿÿéĂŽè£ĞçżğæĽ£çżĎåŘĹèţůæİěãĂĆåIJĺåd'■æİĆåžŤçŤĺçĺŃ

ä¡£çŤĺå§žçśżæİěåĹĘçżĎåŘĎçğ■åijĆåÿÿçśżäź§æŸŕå¿ĹæIJL'çŤĺçŽĎãĂĆåőČåŕŕäžěèőľ'çŤĺæĹůæ■ŢèŐůäÿĂä

```
try:
    s.send(msg)
except ProtocolError:
    ...
```

ä¡ăè£ŸèČ¡æ■ŢèŐůæŻt'åd'ğèŇČåŻt'çŽĎåijĆåÿÿïijŇářšåČŕäÿŃéİćè£ŹæăůïijŽ

```
try:
    s.send(msg)
except NetworkError:
    ...
```

åęĆæđIJä¡ăæČşåőŽäźL'çŽĎæŰřåijĆåÿÿéĞ■åĘŹäžĘ `__init__()` æŰźæşŢïijŇçąőä£İä¡ää¡£çŤĺæL'ĂæIJL'åŔĆæŢřèřČçŤĺ `Exception.__init__()` ïijŇä¿ŃåęĆïijŽ

```
class CustomError(Exception):
    def __init__(self, message, status):
        super().__init__(message, status)
        self.message = message
        self.status = status
```

çIJŃäÿŁåŐżæIJL'çĆźåěĞæĂłïijŇäÿ■è£ĞExceptionçŽĎéżŸèőd'èąŇäÿžæŸŕæŐěåŔŮæL'ĂæIJL'äijăéĂŠç `.args` åšđæĂğäÿ■. å¿Ĺåd'ŽåĚűäžŰåĞ¡æŢřåžŞåŠŇéČĺåĹĘPythonåžŞéżŸèőd'æL'ĂæIJL'åijĆåÿÿéČ¡å£Ěéąża `.args` åšđæĂğïijŇ åŽăæ■d'åęĆæđIJä¡ăå£¡çŢëäžĘè£ŹäÿĂæ■ëïijŇä¡ääijŽåŔŞçŐřæIJL'äžŽæŮűåĂŹä¡ååőŽäź äÿžäžĘæijŤçd'ž `.args` çŽĎä¡£çŤĺïijŇèĂČèŹŚäÿŃäÿŃéİćè£ŹäÿłäĄ£çŤĺåĘĚç¡őçŽĎ *RuntimeError'* åijĆåÿÿçŽĎäžd'äžŠäijŽèŕĪïijŇ æşĺæĎŔçIJŃraiseèŕ■åŔëäÿ■ä¡£çŤĺçŽĎåŔĆæŢřäÿłæŢřæŸŕæĂŐæă

```
>>> try:
...     raise RuntimeError('It failed')
... except RuntimeError as e:
...     print(e.args)
...
('It failed',)
>>> try:
...     raise RuntimeError('It failed', 42, 'spam')
... except RuntimeError as e:

...     print(e.args)
...
('It failed', 42, 'spam')
>>>
```

åĚşäžŐåĹŻåżżèĞłåőŽäźL'åijĆåÿÿçŽĎæŻt'åd'Žä£ąæĄŕïijŇèŕůåŔĆèĂČ'Pythonåő ŸæŰźæŰĞæąč <https://docs.python.org/3/tutorial/errors.html>'_

14.9 æ■ŢèŐůåijĆåÿÿåŘŐæŁŻåĞžåŕęåď ŰçŽĎåijĆåÿÿ

éŮőécŸ

ä¡ăæČşæ■ŢèŐůäÿĂäÿłåijĆåÿÿåŘŐæŁŻåĞžåŕęåď ŰäÿĂäÿłäÿ■åŘŇçŽĎåijĆåÿÿïijŇåŘŇæŮűè£Ÿå¿ŮåIJ

èğčåĘşæŰźæąĹ

äÿžäžĘéŞ¿æŐěåijĆåÿÿïijŇä¡£çŤÍ `raise from` èŕ■åŕěæİěäžčæŻ£çőĂå■ŢçŽĎ `raise` èŕ■åŕěãĂĆ åőČäijŽèőľä¡ăåŘŇæŮűä£İçŢŹäÿď äÿłåijĆåÿÿçŽĎä£ąæĄŕãĂĆä¿ŃåęĆïijŽ

```
>>> def example():
...     try:
...             int('N/A')
...     except ValueError as e:
...             raise RuntimeError('A parsing error occurred') from␣
 ↪e
>>>
example()
Traceback (most recent call last):
  File "<stdin>", line 3, in example
ValueError: invalid literal for int() with base 10: 'N/A'
```

äÿŁéIćçŽĎåijĆåÿÿæŸŕäÿŃéIćçŽĎåijĆåÿÿäžğçŤ§çŽĎçŻť æŐěåŐ§åŻäïijŽ

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in example
RuntimeError: A parsing error occurred
>>>
```

åIJÍåŻđæžŕäÿ■çğŚéŹćçIJŃåĹřïijŇäÿď äÿłåijĆåÿÿéČ¡èćńæ■ŢèŐůãĂĆ èęĄæČşæ■ŢèŐůè£ŹæăůçŽĎåijĆåÿÿïijŇä¡ăåŕŕäžěä¡£çŤÍäÿĂäÿłçőĂå■ŢçŽĎ `except` èŕ■åŕěãĂĆ äÿ■è£ĞïijŇä¡ăè£ŸåŕŕäžěéĂŽè£Ğæ§ěçIJŃåijĆåÿÿårźèśąçŽĎ `__cause__` åśđæĂğæİěèů§èÿłåijĆåÿÿéŞ¿ãĂĆä¿ŃåęĆïijŽ

```
try:
    example()
except RuntimeError as e:
    print("It didn't work:", e)

    if e.__cause__:
        print('Cause:', e.__cause__)
```

å¡ŞåIJÍ `except` åİŮäÿ■åŔĹæIJL'åŕęåď ŰçŽĎåijĆåÿÿèćńæŁŻåĞžæŮűäijŽårijèĞť äÿĂäÿłéŽŘèŮŔçŽĎå

```
>>> def example2():
...     try:
```

```
...             int('N/A')
...     except ValueError as e:
...             print("Couldn't parse:", err)
...
>>>
>>> example2()
Traceback (most recent call last):
  File "<stdin>", line 3, in example2
ValueError: invalid literal for int() with base 10: 'N/A'
```

åIJĺåď ĎçŘĘäÿŁè£řåijĆåÿÿçŽĎæŮűåĂŹïijŇåŔęåď ŰäÿĂäÿłåijĆåÿÿåŔŚçŢ§äžĘïijŽ

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in example2
NameError: global name 'err' is not defined
>>>
```

è£ŹäÿłäżŃå■Řäÿ■ïijŇä¡ääŘŇæŮűèŐůå¿ŮäžĘäÿď äÿłåijĆåÿÿçŽĎä£ąæĄŕïijŇä¡ĘæŸŕåŕźåijĆåÿÿçŽĎèğč è£ŹæŮűåĂŹïijŇNameError åijĆåÿÿèćńä¡IJäÿžçÍŃäžŔæIJĂçżĹåijĆåÿÿèćńæŁŻåĞžïijŇèĂŇäÿ■æŸŕä¡■äžŐ

åęĆæđIJïijŇä¡ăæČşå£¡çŢěæŐĽåijĆåÿÿéŞ¿ïijŇåŔŕä¡£çŤĺ raise from None:

```
>>> def example3():
...     try:
...             int('N/A')
...     except ValueError:
...             raise RuntimeError('A parsing error occurred') from
 ↪None...
>>>
example3()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in example3
RuntimeError: A parsing error occurred
>>>
```

èőĺèőž

åIJĺèő¿èőąäžčçăĄæŮűïijŇåIJĺåŔęåď ŰäÿĂäÿł except äžčçăĄåİŮäÿ■ä¡£çŤĺ raise èŕ■åŔěçŽĎæŮűåĂŹä¡äèęĄçĽźåĹńåřŔå£ČäžĘãĂĆ åď ğåď ŽæŢřæČĚåĘţäÿŇïijŇè£Źçğ■ raise èŕ■åŔěéČ¡åžŤèŕěèćńæŤźæĹŘ raise from èŕ■åŔěãĂĆäź§åřśæŸŕèŕťä¡äåžŤèŕěä¡£çŤĺäÿŃéİćè£Źçğ

```
try:
    ...
except SomeException as e:
    raise DifferentException() from e
```

è£ŹæăůåĄŽçŽĎåŐ§åŻäæŸŕä¡ääžŤèŕěæŸ¿çd'žçŽĎåřĘåŐ§åŻäéŞ¿æŐěèţůæİěãĂĆ

äź§åřśæŸŕèŕ'ïijŇDifferentException æŸŕçŻť æŐěäżŐ SomeException èą■çŤ§èĂŃæİěãĂĆ è£Źçğ■åĘşçşżåŔŕäżěäżŐåŻđæžŕçżŞæđIJäÿ■çIJŃåĞžæİěãĂĆ

åęĆæđIJä¡ääČŔäÿŃéİćè£Źæăůå̄ĘŹäżčçăĄïijŇä¡ääż■çĎűäijŽå¿ŮåĹřäÿĂäÿłéŞ¿æŐěåijĆåÿÿïijŇ äÿ■è£Ğè£ŹäÿłåźűæšąæIJĽå¿ĹæÿĚæŹřçŽĎèŕťæŸŐè£Źäÿłåij Ćåÿÿé Ş¿åĹřåžŢæŸŕåĘĚéČĺåijĆåÿÿè£ŸæŸŕæ§

```
try:
    ...
except SomeException:
    raise DifferentException()
```

å¡Şä¡ää¡£çŤĺ raise from èŕ■åŔěçŽĎèŕİïijŇåŕśå¿ĹæÿĚæěŽçŽĎèąĺæŸŐæŁŻåĞžçŽĎæŸŕçňňäžŇäÿłå

æIJĂåŘŐäÿĂäÿłä¿Ńå■Řäÿ■éŽŘèŮŔåijĆåÿÿéŞ¿ä£ąæĄŕãĂĆ åř¡çőąéŽŘèŮŔåijĆåÿÿéŞ¿ä£ąæĄŕäÿ■åĹľäžŐåŻđæžŕïijŇåŘŇæŮűåőČäź§äÿćåđ'śäžĘå¿Ĺåď'ŽæIJĽçŤĺçŽĎèř äÿ■è£ĞäÿĞäžŃçŽĘåźşç■L'ïijŇæIJĽæŮűåĂŹåŔłä£İçŢŹéĂĆå¡ŞçŽĎä£ąæĄŕäź§æŸŕå¿ĹæIJĽçŤĺçŽĎãĂĆ

14.10 éĞ■æŰřæŁŻåĞžèćńæ■ŢèŐůçŽĎåijĆåÿÿ

éŮőéćŸ

ä¡ääIJĺäÿĂäÿł except åİŮäÿ■æ■ŢèŐůäžĘäÿĂäÿłåijĆåÿÿïijŇçŐřåIJĺæČşéĞ■æŰřæŁŻåĞžåőČãĂĆ

èğčåĘşæŰźæąĹ

çőĂå■ŢçŽĎä¡£çŤĺäÿĂäÿłå■ŢçŃňçŽĎ rasie èŕ■åŔěå■şåŔŕïijŇä¿Ńåę ĆïijŽ

```
>>> def example():
...     try:
...             int('N/A')
...     except ValueError:
...             print("Didn't work")
...             raise
...

>>> example()
Didn't work
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 3, in example
ValueError: invalid literal for int() with base 10: 'N/A'
>>>
```

èőĺèőž

è£ŹäÿłéŭőécŸéĂŽåÿÿæŸŕå¡Şä¡ăéIJĂèęĄåIJĺæ■ŢèŐůåijĆåÿÿåŘŐæĽğèąŇæ§Řäÿłæş■ä¡IJïijĹæŕŤåęĆèć
äÿĂäÿłå¿Ĺåÿÿèğ ĄçŽĎçŤĺæşŢæŸŕåIJĺæ■ŢèŐůæĽĂæIJL'åijĆåÿÿçŽĎåď'ĎçŘĘåŹĺäÿ■ïijŽ

```
try:
    ...
except Exception as e:
    # Process exception information in some way
    ...

    # Propagate the exception
    raise
```

14.11 è¿ŞåĞžè■ęåŚĽä£ąæĄŕ

éŮőécŸ

ä¡ăåÿŇæIJŻèĞłåůśçŽĎçÍŃåžŔèČ¡çŤ§æĹŘè■ęåŚĽä£ąæĄŕïijĹæŕŤåęĆåž§åijČçĽźæĂğæĹŰä¡£çŤĺéŮőéć

èğčåĘşæŰźæąĹ

èęĄè¿ŞåĞžäÿĂäÿłè■ęåŚĽæűĹæĄŕïijŇåŔŕä¡£çŤÍ `warning.warn()` åĞ¡æŢřãĂĆä¿ŃåęĆïijŽ

```
import warnings

def func(x, y, logfile=None, debug=False):
    if logfile is not None:
         warnings.warn('logfile argument deprecated',␣
↪DeprecationWarning)
    ...
```

`warn()` çŽĎåŔĆæŢřæŸŕäÿĂäÿłè■ęåŚĽæűĹæĄŕåŠŇäÿĂäÿłè■ęåŚĽçśżïijŇè■ęåŚĽçśżæIJL'åęĆäÿŃåĞă
DeprecationWarning, SyntaxWarning, RuntimeWarning, ResourceWarning, æĹŰ FutureWarning.

åŕźè■ęåŚĽçŽĎåď'ĎçŘĘåŔŰåĘşäžŐä¡ăåęĆä¡Ţè£ŘèąŇèğčéĞŁåŹĺäžěåŔŁäÿĂäžŻåĚűäżŰéĚ■ç¡õãĆ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăä¡£çŤÍ `-W all` éĂL'éąźåŐżè£ŘèąŇPythonïijŇä¡äăijŽå¿ŮåĹřåęĆäÿŃçŽĎè¿ŞåĞžïijŽ

```
bash % python3 -W all example.py
example.py:5: DeprecationWarning: logfile argument is deprecated
  warnings.warn('logfile argument is deprecated',␣
↪DeprecationWarning)
```

éĂŽåÿÿæİèèőšïijŇè■ęåŚläijŽè¿ŞåĞžåĹřæăĞåĞĘéŤŹèŕŕäÿŁãĂĆåęĆæđIJä¡ăæČşèőšè■ęåŚLè¡ňæ■ćäÿžå
`-W error` éĂĽéąźïijŽ

```
bash % python3 -W error example.py
Traceback (most recent call last):
  File "example.py", line 10, in <module>
    func(2, 3, logfile='log.txt')
  File "example.py", line 5, in func
    warnings.warn('logfile argument is deprecated',␣
 ↪DeprecationWarning)
DeprecationWarning: logfile argument is deprecated
bash %
```

èőÍèőž

åIJÍä¡ăçžť'æŁd'è¡ŕäžűïijŇæŔŘçd'žçŤÍæĹůæ§ŘäžŻä£ąæĄŕïijŇä¡ĘæŸŕåŔĹäÿ■éIJĂèęĄåřĘåĚűäÿŁå■ĞäÿÍ
ä¿ŇåęĆïijŇåĄĞèő¿ä¡ăåĞĘåd'Ğä£őæŤźæ§ŘäÿłåĞ¡æŢřåžŞæĹŰæąĘæđűçŽĎåŁ§èČ¡ïijŇä¡ăåŔŕäžěåĚĹäÿžä¡ăè
ä¡ăè£ŸåŔŕäžěè■ęåŚLçŤÍæĹůäÿĂäžŻåŕźäžčçăĄæIJL'éŮőécŸçŽĎä¡£çŤÍæŰźåijŘãĂĆ

ä¡IJäÿžåŘęåd'ŰäÿĂäÿłåĘĚç¡őåĞ¡æŢřåžŞçŽĎè■ęåŚLä¡£çŤÍä¿Ńå■ŘïijŇäÿŃéİćæijŤçd'žäžĘäÿĂäÿłæšąæ

```
>>> import warnings
>>> warnings.simplefilter('always')
>>> f = open('/etc/passwd')
>>> del f
__main__:1: ResourceWarning: unclosed file <_io.TextIOWrapper name=
 ↪'/etc/passwd'
 mode='r' encoding='UTF-8'>
>>>
```

éżŸèőd'æČĚåĘţäÿŃïijŇåźűäÿ■æŸŕæL'ĂæIJL'è■ęåŚLæűĹæĄŕéČ¡äijŽåĞžçŐřãĂĆ`-W`
éĂĽéąźèČ¡æŐğåĹűè■ęåŚLæűĹæĄŕçŽĎè¿ŞåĞžãĂĆ `-W all`
äijŽè¿ŞåĞžæL'ĂæIJL'è■ęåŚLæűĹæĄŕïijŇ`-W ignore` å£¡çŢěæŐL'æL'ĂæIJL'è■ęåŚLïijŇ`-W error` åřĘè■ęåŚLè¡ňæ■ćæĹŘåijĆåÿÿãĂĆ åŘęåd'ŰäÿĂçğ■éĂĽ'æŃl'ïijŇä¡ăè£ŸåŔŕäžěä¡£çŤÍ `warnings.simplefilter()` åĞ¡æŢřæŐğåĹűè¿ŞåĞžãĂĆ `always` åŔĆæŢřäijŽèől'æL'ĂæIJL'è■ęåŚLæűĹæĄŕåĞžçŐřïijŇ`ignore` å£¡çŢěèřČæL'ĂæIJL'çŽĎè■ęåŚLïijŇ`error` åřĘè■ęåŚLè¡ňæ■ćæĹŘåijĆåÿÿãĂĆ

åŕźäžŐçőĂå■ŢçŽĎçŤ§æĹŘè■ęåŚLæűĹæĄŕçŽĎæČĚåĘţè£ŹäžŻåůšçżŘèűşåd'§äžĘãĂĆ
`warnings` æĺąåİŮåŕźè£Ğæżd'åŠŇè■ęåŚLæűĹæĄŕåd'ĎçŘĘæŔŘä¿ŻäžĘåd'ğéĞŔçŽĎæŻť'énŸçžğçŽĎéĚ■ç¡
æŻť'åd'Žä£ąæĄŕèřůåŔĆèĂČ PythonæŰĞæąč

14.12 èřČèŕŢå§žælJňçŽĎçÍŃåžŔåť'ľæžČéŤŹèŕŕ

éŮőécŸ

ä¡ăçŽĎçÍŃåžŔåěŤæžČåŘŐèřěæĂŐæăůåŐżèřČèŕŢåőČïij§

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăçŽĎçÍŃåžŔåŻääÿžæ§Řäÿłåij Ćåÿÿè ĂŇåěŤæžČïijŇè£ŘèąŇ `python3 -i someprogram.py` åŔŕæĽğèąŇçőĂå■ŢçŽĎèřČèŕŢãĂĆ `-i` éĂĽéąźåŔŕèőľçÍŃåžŔçżŞæİ§åŘŐæĽŞåijĂäÿĂäÿłäžđ'äžŠåijŔshellãĂĆ çĎűåŘŐä¡ääřśèČ¡æ§ěçIJŃçŐŕåćČïijŇä¿ŃåęĆïijŇåĄĞèő¿ä¡ăæIJL'äÿŃéÍćçŽĎäžčçăĄïijŽ

```
# sample.py

def func(n):
    return n + 10

func('Hello')
```

è£ŘèąŇ `python3 -i sample.py` äijŽæIJL'çśżäijijåęĆäÿŃçŽĎè¿ŞåĞžïijŽ

```
bash % python3 -i sample.py
Traceback (most recent call last):
  File "sample.py", line 6, in <module>
    func('Hello')
  File "sample.py", line 4, in func
    return n + 10
TypeError: Can't convert 'int' object to str implicitly
>>> func(10)
20
>>>
```

åęĆæđIJä¡ăçIJŃäÿ■åĹřäÿŁéÍćè£ŹæăůçŽĎïijŇåŔŕäžěåIJÍçÍŃåžŔåěŤæžČåŘŐæĽŞåijĂPythonçŽĎèřČèŕŢ

```
>>> import pdb
>>> pdb.pm()
> sample.py(4)func()
-> return n + 10
(Pdb) w
  sample.py(6)<module>()
-> func('Hello')
> sample.py(4)func()
-> return n + 10
(Pdb) print n
'Hello'
(Pdb) q
>>>
```

åęĆæđIJä¡ăçŽĎäžčçăĄæĽĂåIJÍçŽĎçŐŕåćČå¿ĹéŽ¿èŐůåŔŰäžđ'äžŠshellïijĹæŕŤåęĆåIJÍæ§ŘäÿłæIJ■åŁą éĂŽåÿÿåŔŕäžěæ■ŢèŐůåijĆåÿÿåŘŐèĞłåůśæĽŞå■řèů§èÿłä£ąæĄŕãĂĆä¿ŃåęĆïijŽ

```
import traceback
import sys

try:
    func(arg)
```

```
except:
    print('**** AN ERROR OCCURRED ****')
    traceback.print_exc(file=sys.stderr)
```

èęĄæŸŕä¡ăçŽĎçÍŃåžŔæšąæIJL'åěŤæžČïijŇèĂŇåŘłæŸŕäžğçŤ§äžEäÿĂäžŻä¡ăçIJŃäÿ■æĞĆçŽĎçżŞæđI
ä¡ååIJÍæĎ§åĚt'èűčçŽĎåIJřæŰźæŔŠåĚěäÿĂäÿŃ `print()` èí■åŘěäź§æŸŕäÿłäÿ■éŤŹçŽĎéĂL'æŃl'ãĂĆ
äÿ■è£ĞïijŇèęĄæŸŕä¡ăæL'ŞçőÙè£ŹæăůåĄŽïijŇæIJL'äÿĂäžŻářŔæŁĂåůğåŔŕäžěåÿőåŁl'ä¡ããĂĆ
éęŰåĚĹïijŇ`traceback.print_stack()` åĞ¡æŢřäijŽä¡ăçÍŃåžŔè£ŘèąŇåĹřéĆčäÿłçĆźçŽĎæŮűåĂŹåĹŻ

```
>>> def sample(n):
...     if n > 0:
...             sample(n-1)
...     else:
...             traceback.print_stack(file=sys.stderr)
...
>>> sample(5)
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 3, in sample
  File "<stdin>", line 3, in sample
  File "<stdin>", line 3, in sample
  File "<stdin>", line 3, in sample
  File "<stdin>", line 3, in sample
  File "<stdin>", line 5, in sample
>>>
```

åŔęåd'ŰïijŇä¡ãè£ŸåŔŕäžěåČŔäÿŃéÍćè£ŹæăůäĄ£çŤÍ `pdb.set_trace()`
åIJÍäżžä¡ŢåIJřæŰźæL'ŃåŁÍçŽĎåŘŕåŁÍěŕČèŕŢåŹĺïijŽ

```
import pdb

def func(arg):
    ...
    pdb.set_trace()
    ...
```

å¡ŞçÍŃåžŔæŕŤè¿Čåd'ğäžŇä¡ăæČşěŕČèŕŢæŐğåĹűæţĄçÍŃäżěåŔŁåĞ¡æŢřåŔĆæŢřçŽĎæŮűåĂŹè£Źäÿłåŕs
ä¿ŃåęĆïijŇäÿĂæŮęèŕČèŕŢåŹÍåijĂåğŃè£ŘèąŇïijŇä¡ååŕśèČ¡åd'§ä¡£çŤÍ
`print` æÍěèğĆæţŃåŘŸéĞŔåĂijæĹŰæŢšåĞżæ§Řäÿłåŝ¡äžd'æŕŤåęĆ `w`
æÍěèŐůåŔŰè£¡èÿłä£ąæĄŕãĂĆ

èőÍèőž

äÿ■èęĄåřĘèřČèŕŢåijĎçŽĎè£ĞäžŐåd'■æÍĆåŇŰãĂĆäÿĂäžŻçőĂå■ŢçŽĎéŤŹèŕŕåŔłéIJĂèęĄèğĆår§çÍŃåž
åőđéŹĚçŽĎéŤŹèŕŕäÿĂèĹňæŸŕååĘæăĹçŽĎæIJĂåŘŐäÿĂèąŇãĂĆ
ä¡ååIJÍåijĂåŔŚçŽĎæŮűåĂŹïijŇäź§åŔŕäžěåIJÍä¡ăéIJĂèęĄèřČèŕŢçŽĎåIJřæŰźæŔŠåĚěäÿĂäÿŃ
`print()` åĞ¡æŢřæÍèèŕŁæŰ■ä£ąæĄŕïijĹåŔłéIJĂèęĄæIJĂåŘŐåŔŚäÿČçŽĎæŮűåĂŹåĹăéŹd'è£ŹäžŻæL'Şå■ř

èřČèŕŢåŹÍçŽĎäÿĂäÿłåÿÿèğĄçŤÍæşŢæŸŕèğĆæţŃæ§ŘäÿłåůšçżŔåěŤæžČçŽĎåĞ¡æŢřäÿ■çŽĎåŔŸéĞŔãĂ
ç§ěéĄŞæĂŐæăůåIJÍåĞ¡æŢřåěŤæžČåŘŐè£ŻåĚèřČèŕŢåŹÍæŸŕäÿĂäÿłå¿ĹæIJL'çŤÍçŽĎæŁĂèČ¡ãĂĆ

å¡Şä¡ăæČşèğčåĽŰäÿĂäÿłéİđåÿÿåď■æİĆçŽĎçÍŃåžŔïijŇåžŢåśĆçŽĎæŐğåĹűéĂżè¿Śä¡ăäÿ■æŸŕå¿ĹæÿĚæŔŠåĚě `pdb.set_trace()` è£ŹæăůçŽĎèŕ■åŘěåřšå¿ĹæIJĽçŤĺäžĘãĂĆ

åőđéŹĚäÿŁïijŇçÍŃåžŔäijŽäÿĂçŻťè£ŘèąŇåĹřçćřåĹř `set_trace()` èŕ■åŘëä¡■ç¡őïijŇçĎűåŘŐçńŃél'ňè£ŻåĚëèřČèŕŢåŹĺãĂĆ çĎűåŘŐä¡ăářšåŔŕäžěåĄŽæŻť'åď'ŽçŽĎäžŃäžĘãĂĆ

åęĆæđIJä¡ăä¡£çŤÍIDEæÌěåĄŽPythonåijĂåŔŚïijŇéĂŽåÿÿIDEéČ¡äijŽæŔŘä¿ŻèĞłåůśçŽĎèřČèŕŢåŹĺæÌěäæŻť'åď'Žè£ŹæŰźéÌćçŽĎä£ąæĄŕåŔŕäžěåŔĆèĂČä¡ăä¡£çŤÍçŽĎIDEæĽ'ŃåĘŇãĂĆ

14.13 çżŹä¡äçŽĎçÍŃåžŔåĄŽæĂğèČ¡æţŃèŕŢ

éŮőécŸ

ä¡ăæČşæţŃèŕŢä¡äçŽĎçÍŃåžŔè£ŘèąŇæĽ'ĂèŁśèť'źçŽĎæŮűéŮť'åźűåĄŽæĂğèČ¡æţŃèŕŢãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăåŔłæŸŕçőĂå■ŢçŽĎæČşæţŃèŕŢäÿŃä¡äçŽĎçÍŃåžŔæŢť'ä¡ŞèŁśèť'źçŽĎæŮűéŮť'ïijŇéĂŽåÿÿä¡£çŤÍUnixæŮűéŮť'åĞ¡æŢřåřšèąŇäžĘïijŇærŤåęĆïijŽ

```
bash % time python3 someprogram.py
real 0m13.937s
user 0m12.162s
sys  0m0.098s
bash %
```

åęĆæđIJä¡ăè£ŸéIJĂèęĄäÿĂäÿłçÍŃåžŔåŘĎäÿłçżĘèŁĆçŽĎèŕęçżĘæŁěåŠŁïijŇåŔŕäžěä¡£çŤÍ `cProfile` æĺąåİŮïijŽ

```
bash % python3 -m cProfile someprogram.py
         859647 function calls in 16.016 CPU seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall␣
↪filename:lineno(function)
   263169    0.080    0.000    0.080    0.000 someprogram.
↪py:16(frange)
      513    0.001    0.000    0.002    0.000 someprogram.
↪py:30(generate_mandel)
   262656    0.194    0.000   15.295    0.000 someprogram.py:32(
↪<genexpr>)
        1    0.036    0.036   16.077   16.077 someprogram.py:4(
↪<module>)
   262144   15.021    0.000   15.021    0.000 someprogram.py:4(in_
↪mandelbrot)
        1    0.000    0.000    0.000    0.000 os.py:746(urandom)
```

```
        1    0.000    0.000    0.000    0.000 png.py:1056(_readable)
        1    0.000    0.000    0.000    0.000 png.py:1073(Reader)
        1    0.227    0.227    0.438    0.438 png.py:163(<module>)
      512    0.010    0.000    0.010    0.000 png.py:200(group)
    ...
bash %
```

äÿ■è£ĞéĂŽåÿÿæČĚåĘţæŸŕäżŃäžŐè£ŹäÿďäÿłæđĄçńŕäźŃéŮťăĂĆæŕŤåęĆä¡ååůšçżŔç§ěéĄŞäżčçăĄè£Ł
åŕźäžŐè£ŹäžŽåĞ¡æŢřçŽĎæĂğèČ¡æţŃèŕŢïijŇåŔŕäżěä¡£çŤĺäÿĂäÿłçŐĂå■ŢçŽĎèčĚéěřåŹĺïijŽ

```
# timethis.py

import time
from functools import wraps

def timethis(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.perf_counter()
        r = func(*args, **kwargs)
        end = time.perf_counter()
        print('{}.{} : {}'.format(func.__module__, func.__name__,␣
↪end - start))
        return r
    return wrapper
```

èęĄä¡£çŤĺè£ŹäÿłèčĚéěřåŹĺïijŇåŔŕéIJĂèęĄåřĘåĚűæŤ¿ç¡őåIJĺä¡ăèęĄè£ŻèąŇæĂğèČ¡æţŃèŕŢçŽĎåĞ¡æŢ

```
>>> @timethis
... def countdown(n):
...     while n > 0:
...             n -= 1
...
>>> countdown(10000000)
__main__.countdown : 0.803001880645752
>>>
```

èęĄæţŃèŕŢæ§Řäÿłäžčçă Ąå İŮè£ŘèąŇæŮűéŮťïijŇä¡ăåŔŕäżěåőŽäźĽäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺïijŇ

```
from contextlib import contextmanager

@contextmanager
def timeblock(label):
    start = time.perf_counter()
    try:
        yield
    finally:
        end = time.perf_counter()
        print('{} : {}'.format(label, end - start))
```

äÿŃéİćæŸŕä¡£çŤĺè£ŹäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺçŽĎä¿Ńå■ŘïijŽ

```
>>> with timeblock('counting'):
...     n = 10000000
...     while n > 0:
...             n -= 1
...
counting : 1.5551159381866455
>>>
```

åŕźäžŐæţŃèřŢå¿ĹåřŔçŽĎäžčçăĄçĽĞæőţè£ŘèąŇæĂğèČ¡ïijŇä¡£çŤÍ timeit æĺąåİŮäijŽå¿ĹæŰźä¿£ïijŇä¿ŃåęĆïijŽ

```
>>> from timeit import timeit
>>> timeit('math.sqrt(2)', 'import math')
0.1432319980012835
>>> timeit('sqrt(2)', 'from math import sqrt')
0.10836604500218527
>>>
```

timeit äijŽæĽğèąŇçňňäÿĂäÿłåŔĆæŢřäÿ■èŕ■åŘě100äÿĞæňąåźűèőąçőŮè£ŘèąŇæŮűéŮťăĂĆ çňňäžŇäÿłåŔĆæŢřæŸŕè£ŘèąŇæţŃèřŢäźŃåĽ■éĚ■ç¡őçŐŕåćČãĂĆåęĆæđIJä¡ăæČşæŤźåŔŸå¿łçŐŕæĽğèąŇæň åŘŕäžěåČŔäÿŃéİćè£Źæăůèő¿ç¡ő number åŔĆæŢřçŽĎåĂijïijŽ

```
>>> timeit('math.sqrt(2)', 'import math', number=10000000)
1.434852126003534
>>> timeit('sqrt(2)', 'from math import sqrt', number=10000000)
1.0270336690009572
>>>
```

èőĺèőž

å¡ŞæĽğèąŇæĂğèČ¡æţŃèřŢçŽĎæŮűåĂŹïijŇéIJÀèęĄæşĺæĎŔçŽĎæŸŕä¡ăèŐůåŔŰçŽĎçżŞæđIJéČ¡æŸŕè time.perf_counter() åĞ¡æŢřäijŽåIJĺçżŹåőŽåźşåŘŕäÿŁèŐůåŔŰæIJÄéńŸçš¿ážęçŽĎèőąæŮűåĂijãĂĆ äÿ■è£ĞïijŇåőČäż■çĎûè£ŸæŸŕå§žäžŐæŮűéŠ§æŮűéŮťïijŇå¿Ĺåď˙ŽåŻäçť˙ääijŽå¡šåŞ■åĹřåőČçŽĎçš¿çąőåžęï åęĆæđIJä¡ååŕźäžŐæĽğèąŇæŮűéŮťæŻť˙æĎ§åĚť˙èűčïijŇä¡£çŤÍ time.process_time() æİěäžčæŻ£åőČãĂĆä¿ŃåęĆïijŽ

```
from functools import wraps
def timethis(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.process_time()
        r = func(*args, **kwargs)
        end = time.process_time()
        print('{}.{} : {}'.format(func.__module__, func.__name__,␣
↪end - start))
        return r
    return wrapper
```

æIJĂåŘŐïijŇåęĆæđIJä¡ăæČşè£ŻèąŇæŻť æůśåĚěçŽĎæĂğèČ¡åĹĘæđŘïijŇéĆčäźĹä¡ăéIJĂèęĄèŕęçżĘéŸĖ
time ãĂĄtimeit åŠŇåĚűäžŰçŻÿåĚşæĺąåİŮçŽĎæŰĞæąčãĂĆ
è£Źæăůä¡ăåŘŕäžěçŘĘèğčåŠŇåźşåŘřçŻÿåĚşçŽĎåůőåijĆäžěåŘŁäÿĂäžŻåĚűäžŰéŹůéŸśãĂĆ
è£ŸåŘŕäžěåŘĆèĂČ13.13åřŔèŁĆäÿ■çŻÿåĚşçŽĎäÿĂäÿłåĹŻåżžèőąæŮűåŹĺçśżçŽĎä¿Ňå■ŘãĂĆ

14.14 åŁăéĂ§çĺŃåžŔè£ŘèąŇ

éŮőécŸ

ä¡ăçŽĎçĺŃåžŔè£ŘèąŇåd'łæĚćïijŇä¡ăæČşåIJĺäÿ■ä¡£çŤĺåd'■æİĆæŁĂæIJŕæŕŤåęĆCæL'l'åśŢæĹŰJITçijŰ

èğčåĘşæŰźæąĹ

åĚşäžŐçĺŃåžŔäijŸåŇŰçŽĎçňňäÿĂäÿłåĞĘåĹŹæŸŕâĂIJäÿ■èęĄäijŸåŇŰâĂİïijŇçňňäžŇäÿłåĞĘåĹŹæŸí
åęĆæđIJä¡ăçŽĎçĺŃåžŔè£ŘèąŇçijŞæĚćïijŇéęŰåĚĹä¡ăå¿Ůä¡£çŤĺ14.13åřŔèŁĆçŽĎæŁĂæIJŕåĚĹåŕźåőĆè£Żèă

éĂŽåÿÿæİěèőšä¡ăäijŽåŔŞçŐřä¡åå¿ŮçĺŃåžŔåIJĺåřŞæŢřåĞääÿłçĈ■çĆźåIJřæŰźèŁśèť źäžĘåd'ğéĞŔæŮűé
äÿ■çĎűåĘĚå■ŸçŽĎæŢřæ■őåd'ĎçŘĘå¿łçŐřãĂĆäÿĂæŮęä¡ăåőŽä¡■åĹrè£ŹäžŻçĆźïijŇä¡ăåřšåŔŕäžěä¡£çŤĺäÿŇ

ä¡£çŤĺåĞ¡æŢř

å¿Ĺåd'ŽçĺŃåžŔåŚŸåĹŽåijĂåğŃäijŽä¡£çŤĺPythonèŕ■èĺĂåĘŹäÿĂäžŻçőĂå■ŢèĎŽæIJñãĂĆ
å¡ŞçijŰåĘŹèĎŽæIJňçŽĎæŮűåĂŹïijŇéĂŽåÿÿäźăæČŕäžĘåĘŹæŕńæŮăçżŞædĎçŽĎäžčçăĄïijŇæŕŤåęĆïijŽ

```
# somescript.py

import sys
import csv

with open(sys.argv[1]) as f:
     for row in csv.reader(f):

          # Some kind of processing
          pass
```

å¿ĹåřŚæIJL'äžžç§ěéĄŞïijŇåČŔè£ŹæăůåőŽäźL'åIJĺåĚĺåśĂèŇČåŻť çŽĎäžčçăĄè£ŘèąŇèţůæİěèęĄæŕŤåőŽ
è£Źçğ■éĂ§åžęåůőåijĆæŸŕçŤśäžŐåśĂéČĺåŔŸéĞŔåŠŇåĚĺåśĂåŔŸéĞŔçŽĎåőđçŐřæŰźåijŔïijĹä¡£çŤĺåśĂéČĺ
åŽăæ■d'ïijŇåęĆæđIJä¡ăæČşèől'çĺŃåžŔè£ŘèąŇæŻť å£ńäžŻïijŇåŔłéIJĂèęĄåřĘèĎŽæIJñèŕ■åŘěæŤ¿åĚěåĞ¡æŢ

```
# somescript.py
import sys
import csv

def main(filename):
    with open(filename) as f:
         for row in csv.reader(f):
             # Some kind of processing
             pass
```

```
main(sys.argv[1])
```

éĂ§åžęçŽĎåůőåijĆåŔŰåĘşäžŐåőđéŹĚè£Řèą ŇçŽĎçÍŃåžŔïijŇäÿ■è£Ğæăźæ■őçżŔéłŇïijŇä¡£çŤĺåĞ¡æŢ 30%çŽĎæĂğèČ¡æŔŘå■ĞæŸŕå¿Ĺåÿÿè§ĄçŽĎãĂĆ

åř¡åŔŕèČ¡åŐżæŐĽåśđæĂğèő£éŮő

æŕŔäÿĂæňąä¡£çŤĺçĆź(.)æŞ■ä¡IJçňęæÌěèő£éŮőåśđæĂğçŽĎæŮűåĂŹäijŽåÿęæÌěéćÌåď ŰçŽĎåijĂéŤĂãĂ åőČäijŽèğęåŔŞçĽźåőŽçŽĎæŰźæşŢäijŽè£ŻèąŇå■ŮåĚÿæŞ■ä¡IJæŞ■ä¡IJãĂĆ

éĂŽåÿÿä¡ăåŔŕäžěä¡£çŤÍ `from module import name` è£Źæăůç ŽĎåŕijåĚěå¡ćåijŔïijŇäžěåŔŁä¡£çŤĺçżŞåőŽçŽĎæŰźæşŢãĂĆ åĄĞèő¿ä¡ăæIJL'ąęĆäÿŃçŽĎäžčçăĄçL'ĞæőţïijŽ

```
import math

def compute_roots(nums):
    result = []
    for n in nums:
        result.append(math.sqrt(n))
    return result

# Test
nums = range(1000000)
for n in range(100):
    r = compute_roots(nums)
```

åIJÍæĹŚäžñæIJžåŹÍäÿŁéÌćæţŃèŕŢçŽĎæŮűåĂŹïijŇè£ŹäÿłçÍŃåžŔèŁśèť źäžĘåď ğæęĆ40çğŠãĂĆçŐřåIJÍa `compute_roots()` åĞ¡æŢřåęĆäÿŃïijŽ

```
from math import sqrt

def compute_roots(nums):

    result = []
    result_append = result.append
    for n in nums:
        result_append(sqrt(n))
    return result
```

ä£őæŤźåŘŐçŽĎçL'ĹæIJňè£ŔèąŇæŮűéŮť åď ğæęĆæŸŕ29çğŠãĂĆåŤŕäÿĂäÿ■åŘŇäźŃåď ĎåřśæŸŕæűĹéçŤĺ `sqrt()` äžčæŽ£äžĘ `math.sqrt()` ãĂĆ `The result.append()` æŰźæşŢèćńètŃçżŹäÿĂäÿłåśĂéČĺåŔŸéĞŔ `result_append` ïijŇçĎűåŘŐåIJÍåĘĚéČĺå¿łçŐŕäÿ■ä¡£çŤĺåőČãĂĆ

äÿ■è£ĞïijŇè£ŹäžŽæŤźåŔŸåŔłæIJL'åIJÍåď ğéĞŔéĞ■åď ■äžčçăĄäÿ■æL'■æIJL'æĎŔäźL'ïijŇærŤåęĆå¿łçĆ åŽăæ■ď ïijŇè£ŹäžŽäijŸåŇŰäź§åŔłæŸŕåIJÍæ§ŘäžŽçL'źåőŽåIJřæŰźæL'■åžŤèŕĕèćńä¡£çŤĺãĂĆ

çŘĘèğčåśĂéČĺåŔŸéĞŔ

äźŃåĽ■æŔŘè£ĞïijŇåśĂéČĺåŔŸéĞŔäijŽæŕŤåĚĺåśĂåŔŸéĞŔè£ŘèąŇéĂ§åžęå£ńãĂĆ åŕźäžŐéćŚçźĄèő£éŮőçŽĎåŘ■çğřïijŇéĂŽè£ĞåřĘè£ŹäžŽåŘ■çğřåŔŸæĹŘåśĂéČĺåŔŸéĞŔåŔŕäžěåŁäéĂ§çĺŃä ä¿ŃåęĆïijŇçIJŃäÿŃäźŃåĽ■åŕźäžŐ `compute_roots()` åĞ¡æŢřè£ŽèąŇä£őæŤźåŘŐçŽĎçĽĹæIJñïijŽ

```
import math

def compute_roots(nums):
    sqrt = math.sqrt
    result = []
    result_append = result.append
    for n in nums:
        result_append(sqrt(n))
    return result
```

åIJĺè£ŹäÿłçĽĹæIJñäÿ■ïijŇ`sqrt` äžŐ `match` æĺąåĺŮècńæŃ£åĞžåźűæŤ¿åĚěäžĘäÿĂäÿłåśĂéČĺåŔŸéĞŔ åęĆædIJä¡äè£ŘèąŇè£ŹäÿłäžčçăĄïijŇåď ğæęĆèŁśèť ź25çğŠïijĹåŕźäžŐäźŃåĽ■29çğŠåŔĹæŸŕäÿĂäÿłæŤźè£Žż è£Źäÿłécĺåď ŰçŽĎåŁäéĂ§åŐ§åŽäæŸŕåŽääÿžåŕźäžŐåśĂéČĺåŔŸéĞŔ `sqrt` çŽĎæ§ěæL'¿èęĄå£ńäžŐåĚĺåśĂåŔŸéĞŔ `sqrt`

åŕźäžŐçśżäÿ■çŽĎåśđæĂğèő£éŮőäź§åŘŇæăůéĂĆçŤĺäžŐè£ŹäÿłåŐ§çŘĘãĂĆ éĂŽåÿÿæĺěèőšïijŇæ§ěæL'¿æ§ŘäÿłåĂijæŕŤåęĆ `self.name` äijŽæŕŤèő£éŮőäÿĂäÿłåśĂéČĺåŔŸéĞŔèęĄæĚćäÿĂäžŽãĂĆ åIJĺåĘĚéČĺå¿łçŐŕäÿ■ïijŇåŔŕäžěåŕĘæ§ŘäÿłéIJĂè

```
# Slower
class SomeClass:
    ...
    def method(self):
         for x in s:
             op(self.value)

# Faster
class SomeClass:

    ...
    def method(self):
         value = self.value
         for x in s:
             op(value)
```

éĄ£åĚ■äÿ■å£ĚèęĄçŽĎæŁ¡èšą

äžžä¡ŢæŮűåĂŹå¡Şä¡ăä¡£çŤĺécĺåď ŰçŽĎåď ĎçŘĘåśĆïijĹæŕŤåęĆèčĚéěřåŹĺãĂĄåśđæĂğèő£éŮőãĂĄæŔ æŕŤåęĆçIJŃäÿŃåęĆäÿŃçŽĎè£ŹäÿłçśżïijŽ

```
class A:
    def __init__(self, x, y):
        self.x = x
        self.y = y
    @property
    def y(self):
        return self._y
    @y.setter
```

```
    def y(self, value):
        self._y = value
```

çŐřåIJĺè£ŻèąŇäÿĂäÿłçőĂå■ŢæţŃèŕŢïijŽ

```
>>> from timeit import timeit
>>> a = A(1,2)
>>> timeit('a.x', 'from __main__ import a')
0.07817923510447145
>>> timeit('a.y', 'from __main__ import a')
0.35766440676525235
>>>
```

åŘŕäžěçIJŃåĹřïijŇèő£éŮőåśđæĂğyçŻÿæŕŤåśđæĂğxèĂŇèĺĂæĚćçŽĎäÿ■æ■ćäÿĂçĆźçĆźïijŇåď ğæęĆæ
åęĆæđIJä¡ăåIJĺæĎŘæĂğèČ¡çŽĎèŕĺïijŇéĆčäźĹåřśéIJĂèęĄéĞ■æŰřåőąèğĘäÿŃåŕźäžŐyçŽĎåśđæĂğèő£éŮőåŽ
åęĆæđIJæšąæIJL'å£ĚèęĄïijŇåřśä¡£çŤĺçőĂå■ŢåśđæĂğåŘğãĂĆ
åęĆæđIJäžĚäžĚæŸŕåŻääÿžåĚűäžŰçijŰçĺŃèŕ■èĺĂéIJĂèęĄä¡£çŤĺgetter/setteråĞ¡æŢřåřśåŐżä£őæŤźäžčçăĄéč

ä¡£çŤĺåĘĚç¡őçŽĎåőźåŹĺ

åĘĚç¡őçŽĎæŢřæ■őçśżåđŃæŕŤåęĆå■Ůçňęäÿšã ĂĄåĚČçżĎãĂĄåĹŮèąĺãĂĄéŻĘåŘĹåŠŇå■ŮåĚÿéČ¡æŸŕ
åęĆæđIJä¡ăæČşèĞłåůśåőđçŐřæŰřçŽĎæŢřæ■őçżŞæđĎïijĹæŕŤåęĆéŞ¿æŐěåĹŮèąĺãĂĄåźşèąąæăŚç■L'ïijL'ïijŇ
éĆčäźĹèęĄæČşåIJĺæĂğèČ¡äÿŁè¿¿åĹřåĘĚç¡őçŽĎéĂ§åžęåĞääźŐäÿ■åŘŕèČ¡ïijŇåŽăæ■d'ïijŇè£ŸæŸŕäźŰäź

éĄ£åĚ■åĹŻåžžäÿ■å£ĚèęĄçŽĎæŢřæ■őçżŞæđĎæĹŰåd'■åĹű

æIJL'æŮűåĂŹçĺŃåžŘåŚŸæČşæŸ¿æŚĘäÿŃïijŇæđĎéĂääÿĂäžŻåźűæšąæIJL'å£ĚèęĄçŽĎæŢřæ■őçżŞæđ

```
values = [x for x in sequence]
squares = [x*x for x in values]
```

äź§èőÿè£ŹéĞŇçŽĎæČşæşŢæŸŕéęŰåĚĹåřĘäÿĂäžŻåĂijæŤűéŽĘåĹřäÿĂäÿłåĹŮèąĺäÿ■ïijŇçĎűåŘŐä¡£çŤ
äÿ■è£ĞïijŇçňňäÿĂäÿłåĹŮèąĺåőŇåĚĺæšąæIJL'å£ĚèęĄïijŇåŘŕäžěçőĂå■ŢçŽĎåČŘäÿŃéÍćè£ŹæăůåĘŹïijŽ

```
squares = [x*x for x in sequence]
```

äÿŐæ■d'çŻÿåĚşïijŇè£ŸèęĄæşĺæĎŘäÿŃéĆčäžŻåŕźPythonçŽĎåĚšäžńæŢřæ■őæIJžåĹűè£ĞäžŐåĄŘæL'ğ
æIJL'äžŻäžžåźűæšąæIJL'å¿Ĺåě¡çŽĎçŘĘèğčæĹŰä£ąääżżPythonçŽĎåĘĚå■ŸæĺąåđŃïijŇæžěçŤĺ
`copy.deepcopy()` äźŃçšżçŽĎåĞ¡æŢřãĂĆ éĂŽåÿÿåIJĺè£ŹäžŻäžčçăĄäÿ■æŸŕåŘŕäžěåŐżæŐL'åd'■åĹűæŞ

èőĺèőž

åIJĺäijŸåŇŰäźŃåL'■ïijŇæIJL'å£ĚèęĄåĚĹçăŤçl'űäÿŃä¡£çŤĺçŽĎçőŮæşŢãĂĆ
éĂL'æŃl'äÿĂäÿłåd'■æİĆåžęäÿž O(n log n) çŽĎçőŮæşŢèęĄærŤä¡ăåŐżèŕČæŢt'äÿĂäÿłåd'■æİĆåžęäÿž
O(n**2) çŽĎçőŮæşŢæL'ĂåÿęæİěçŽĎæĂğèČ¡æŘŘå■ĞèęĄåd'ğå¿Ůåd'ŽãĂĆ

åęĆæđIJä¡ăèğL'å¿Ůä¡ăè£ŸæŸŕå¿Ůè£ŻèąŇäijŸåŇŰïijŇéĆčäźĹèŕůäžŐæŢt'ä¡ŞèĂČèŹŚãĂĆ
ä¡IJäÿžäÿĂèĹňåĞĘåĹŹïijŇäÿ■èęĄåŕźçĺŃåžŘçŽĎærŔäÿĂäÿłéČĺåĹĘéČ¡åŐżäijŸåŇŰ,åŻăäÿžè£ŹäžŻä£őæŤź
ä¡ăåžŤèŕěäÿŞæşĺäžŐäijŸåŇŰäžğçŤ§æĂğèČ¡çŞűécĹçŽĎåIJræŰźïijŇærŤåęĆåĘĚéČĺå¿łçŐŕãĂĆ

ä¡ăè£ŸèęĄæşĺæĎŘå¿őåřŔäijŸåŇŰçŽĎçżŞæđIJãĂĆä¿ŃåęĆèĂČèŹŚäÿŃéÍćåĹŻåžžäÿĂäÿłå■ŮåĚÿçŽĎă

```
a = {
    'name' : 'AAPL',
    'shares' : 100,
    'price' : 534.22
}

b = dict(name='AAPL', shares=100, price=534.22)
```

åŘŐéÍćäÿĂçğ■åĘŹæşŢæŻťçőĂæťĄäÿĂäžŻïijĹä¡ääÿ■éIJĂèęĄåIJÍåĚşéŤőå■ŮäÿŁè¿ŞåĚěåijŢåŔůïijĽãĂ
äÿ■è£ĞïijŇåęĆæđIJä¡ăářĘè£ŹäÿďäÿłäžčçãĄçĽĞæőţè£ŻèąŇæĂğèČ¡æţŃèŕŢářźæŕŤæŮűïijŇäijŽåŔŚçŐřä¡£ç
dict() çŽĎæŰźåijŔäijŽæĚćäžĘ3åĂ■ãĂĆ çIJŃåĹřè£ŹäÿłïijŇä¡ăæŸŕäÿ■æŸŕæIJĽåĘšåŁĺæŁŁæĽĂæIJĽä¡
dict() çŽĎäžčçãĄéČ¡æŻ£æ■ćæĹŘçňňäÿĂçğ■ãĂĆ äÿ■åď§ïijŇèĄłæŸŐçŽĎçÍŃåžŔåŚŸåŔłäijŽåĚşæşĺäžŰ

åęĆæđIJä¡ăçŽĎäijŸåŇŰèęĄæśĆæŕŤè¿ČéńŸïijŇæIJňèŁĆçŽĎè£ŹäžŻçŐÅå■ŢæŁĂæIJŕæżąèűşäÿ■äžĘïij
ä¿ŃåęĆïijŇPyPyåůěçÍŃæŸŕPythonèğčéĞŁåŹÍçŽĎåŔęåď ŰäÿĂçğ■åőđçŐřïijŇåőČäijŽåĹĘæđŘä¡ăçŽĎçÍŃåž
åőČæIJĽæŮűåĂŹèČ¡æđĄåď ğçŽĎæŔŘå■ĞæĂğèČ¡ïijŇéĂŽåÿÿåŔŕäžěæŐěè£ŚCäžčçãĄçŽĎéĂ§åžęãĂĆ
äÿ■è£ĞåŔŕæČIJçŽĎæŸŕïijŇåĹřåĘŹè£ŹæIJňäźęä¡■ç¡őïijŇPyPyè£Ÿäÿ■èČ¡åőŇåĚíæŤŕæŇĄPython3.
åŽăæ■ď ïijŇè£ŹäÿłæŸŕä¡ääřĘæİěéIJĂèęĄåŐžçãŢçľűçŽĎãĂĆä¡ăè£ŸåŔŕäžěèĂČèŹŚäÿŃNumbaåůěçÍŃïijŇ
NumbaæŸŕäÿĂäÿłåIJÍä¡ää¡£çŤÍèčĚéěřåŹÍæİěéĂĽ'æŃľPythonåĞ¡æŢřè£ŻèąŇäijŸåŇŰæŮűçŽĎåŁÍæĂĄçijŰ
è£ŹäžŻåĞ¡æŢřäijŽä¡£çŤÍLLVMèćńçijŰèŕŚæĹŘæIJňåIJřæIJžåŹÍçăĄãĂĆåőČåŘŇæăůåŔŕäžěæđĄåď ğçŽĎæŔ
ä¡ĘæŸŕïijŇèů§PyPyäÿĂæăůïijŇåőČåŕźäžŐPython 3çŽĎæŤŕæŇĄçŐřåIJÍè£ŸåĄIJçŢŹåIJÍåőđéłŇéŸűæőţãĂĆ

æIJĂåŘŐæĹŚåijŢçŤÍJohn Ousterhoutèŕť è£ĞçŽĎèŕÍä¡IJäÿžçżŞår¿ïijŽâĂIJæIJĂåě¡çŽĎæĂğèČ¡äijŸåŇŰ
çŻť åĹřä¡ăçIJ§çŽĎéIJĂèęĄäijŸåŇŰçŽĎæŮűåĂŹåĘ■åŐżèĂČèŹŚáőČãĂĆçąőä£Íä¡ăçÍŃåžŔæ■čçąőçŽĎè£Řè

CHAPTER 17

çňňå■ĄäžŤçńäïijŽCèŕ■èÍĂæĽľåśŢ

æIJňçńăçİĂçIJijäžŐäżŐPythonèő£éŮőCäžčçăĄçŽĎéŮőécŸãĂĆèőÿåď'ŽPythonåĘĚç¡őåžŞæŸŕçŤÍCåĘŹ
èő£éŮőCæŸŕèőľPythonçŽĎåŕźçŐřæIJL'åžŞè£ŻèąŇäžď'äžŠäÿĂäÿłéĞ■èęĄçŽĎçżĎæĹŘéČĺåĹĘãĂĆ
è£Źäź§æŸŕäÿĂäÿłå¡Şä¡ăéİćäÿť'äżŐPython 2 åĹř Python 3æĽľ'åśŢäžčçăĄçŽĎéŮőécŸãĂĆ
èŹ¡çĎűPythonæŘŘä¿ŻäžĘäÿĂäÿłáź£æşŽçŽĎçijŰçÍŃAPIïijŇåőđéŹĚäÿŁæIJL'å¿Ĺåď'ŽæŰźæşŢæİěåď'ĎçŘĘ
çŻÿæŕŤèŕŢåŻ¿çżŹåĞžåŕźäžŐæŕŔäÿĂäÿłåŘŕèČ¡çŽĎåůěåĚůæĹŰæŁĂæIJŕçŽĎèŕęçżĘåŔĆèĂČïijŇ
æĹŚäźĹéĞĞçŤÍçŽĎæŸŕæŸŕéŻĘäÿ■åIJÍäÿĂäÿłåřŔçL'Ğæőţç ŽĎC++äżčçăĄïijŇäžěåŘŁäÿĂäžŻæIJL'äžčèąÍæ
è£ŹäÿłçŽőæăĞæŸŕæŘŘä¿ŻäÿĂçşżåĹŮçŽĎçijŰçÍŃæÍąæİ£ïijŇæIJL'çżŔéłŇçŽĎçÍŃåžŘåŚŸåŘŕäžěæĽľ'åśŢè

è£ŹéĞŇæŸŕæĹŚäžňåřĘåIJÍåď'ğéČĺåĹĘçğŸçś■äÿ■åůěä¡IJçŽĎäžčçăĄïijŽ

```
/* sample.c */_method
#include <math.h>

/* Compute the greatest common divisor */
int gcd(int x, int y) {
    int g = y;
    while (x > 0) {
        g = x;
        x = y % x;
        y = g;
    }
    return g;
}

/* Test if (x0,y0) is in the Mandelbrot set or not */
int in_mandel(double x0, double y0, int n) {
    double x=0,y=0,xtemp;
    while (n > 0) {
        xtemp = x*x - y*y + x0;
        y = 2*x*y + y0;
```

```
        x = xtemp;
        n -= 1;
        if (x*x + y*y > 4) return 0;
    }
    return 1;
}

/* Divide two numbers */
int divide(int a, int b, int *remainder) {
    int quot = a / b;
    *remainder = a % b;
    return quot;
}

/* Average values in an array */
double avg(double *a, int n) {
    int i;
    double total = 0.0;
    for (i = 0; i < n; i++) {
        total += a[i];
    }
    return total / n;
}

/* A C data structure */
typedef struct Point {
    double x,y;
} Point;

/* Function involving a C data structure */
double distance(Point *p1, Point *p2) {
    return hypot(p1->x - p2->x, p1->y - p2->y);
}
```

è£Źæőţäžčçă Ąå ŇĚåŘńäžĘåď Žçğ■äÿ■åŘŇçŽĎCèŕ■èÍĂçijŰçÍŃçĽźæĂğãĂĆ éęŰåĚĹïijŇè£ŹéĞŇæIJĽå¿Ĺåď ŽåĞ¡æŢřæŕŤåęĆ `gcd()` åŠŇ `is_mandel()` ãĂĆ `divide()` åĞ¡æŢřæŸŕäÿĂäÿłè£ŤåŽđåď ŽäÿłåĂijçŽĎCåĞ¡æŢřä¿Ńå■ŘïijŇåĚűäÿ■æIJĽäÿĂäÿłæŸŕéĂŽè£Č `avg()` åĞ¡æŢřéĂŽè£ĞäÿĂäÿłCæŢřçżĎæĽğèąŇæŢřæ■őèĄŽéŻĘæŞ■ä¡IJãĂĆ`Point` åŠŇ `distance()` åĞ¡æŢřæűĽåŔŁåĹřäžĘCçžŞædĎä¡ŞãĂĆ

åŕźäžŐæŐěäÿŃæİěçŽĎæĽĂæIJĽåřŔèŁĆïijŇåĚĹåĄĞåőŽäÿŁéİćçŽĎäžčçăĄåůšçżŔèćńåĘŹåĚěäžĘąäÿĂ çĎűåŘŐåőČäžňçŽĎåőŽäźĽèćńåĘŹåĚěäÿĂäÿłåŘ■åŔńâĂIJsample.hâĂİçŽĎåď ť æŰĞäžűäÿ■ïijŇ åźűäÿŤèćńçijŰèŕŚäÿžäÿĂäÿłåžŞåŔńâĂIJlibsampleâĂİïijŇèČ¡èćńéŞ¿æŐěåĹřåĚűäžŰCèŕ■èÍĂäžčçăĄäÿ■ãĂĆ çijŰèŕŚåŠŇéŞ¿æŐěçŽĎçżĘèŁĆä¿İæ■őçşżçż§çŽĎäÿ■åŘŇèĂŇäÿ■åŘŇïijŇä¡ĘæŸŕè£Źäÿłäÿ■æŸŕæĹŚäžňåĚ åęĆæđIJä¡ăèęĄåď ĎçŘĘCäžčçăĄïijŇæĹŚäžňåĄĞåőŽè£ŹäžŽå§žçąĂçŽĎäÿIJèě£ä¡ăéČ¡æŐŇæŔąäžĘãĂĆ

Contents:

15.1 ä¡£çŤĺctypesèő£éŮőCäžčçăĄ

éŮőécŸ

ä¡ăæIJL'äÿĂäžŻCåĞ¡æŢřåůšçżŔèćńçijŰèŕŚåĹřåĚśäžńåžŞæĹŰDLLäÿ■ãĂCä¡ååÿŇæIJŻåŔŕäžěä¡£çŤĺçžíèĂŇäÿ■çŤĺçijŰåĘŹéćÍåd'ŰçŽĎCäžčçăĄæĹŰä¡£çŤĺçňňäÿĽ'æŰźæL'ľåśŢåůěåĚůãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐéIJĂèęĄěŕČçŤĺCäžčçăĄçŽĎäÿĂäžŻåŕŔçŽĎéŮőécŸïijŇéĂŽåÿÿä¡£çŤĺPythonæăĞåĞĘåžŞäÿ■çŽctypes æĺąåİŮåřśèűşåd'§äžĘãĂĆ èęĄä¡£çŤĺ ctypes ïijŇä¡ăéęŰåĚĹèęĄçąőä£İä¡ăèęĄèő£éŮőçŽĎCäžčçăĄïijĹåŘŇæăůçŽĎæđűædĎãĂĄå■Ůåd'ğåŕŔãĂĄçijŰèŕŚåŹĺç■L'ïijL'çŽĎæ§ŘäÿłåĚśäžńåžŞäÿ■äžĘãĂĆäÿžäžĘè£Żèą ŇæIJńèŁĆçŽĎæijŢçd'žïijŇåĄĞèő¿ä¡ăæIJL'äÿĂäÿłåĚśäžńåžŞåŘ■å■ŮåŔńlibsample.so ïijŇéĞŇéİćçŽĎåĘĚåőźåřśæŸŕ15çńääżŇçż■éČĺåĹĘéĆčæăůãĂĆåŔęåd'Ůè£ŸåĄĞèő¿è£ŹäÿłåĞ¡æŢřåžŞïijŇä¡ăèęĄåĚĹæđĎåžžäÿĂäÿłåŇĚèčĚåőČçŽĎPythonæĺąåİŮïijŇåęĆäÿŇè

```
# sample.py
import ctypes
import os

# Try to locate the .so file in the same directory as this file
_file = 'libsample.so'
_path = os.path.join(*(os.path.split(__file__)[:-1] + (_file,)))
_mod = ctypes.cdll.LoadLibrary(_path)

# int gcd(int, int)
gcd = _mod.gcd
gcd.argtypes = (ctypes.c_int, ctypes.c_int)
gcd.restype = ctypes.c_int

# int in_mandel(double, double, int)
in_mandel = _mod.in_mandel
in_mandel.argtypes = (ctypes.c_double, ctypes.c_double, ctypes.c_
↪int)
in_mandel.restype = ctypes.c_int

# int divide(int, int, int *)
_divide = _mod.divide
_divide.argtypes = (ctypes.c_int, ctypes.c_int, ctypes.
↪POINTER(ctypes.c_int))
_divide.restype = ctypes.c_int

def divide(x, y):
    rem = ctypes.c_int()
    quot = _divide(x, y, rem)
```

```
    return quot,rem.value

# void avg(double *, int n)
# Define a special type for the 'double *' argument
class DoubleArrayType:
    def from_param(self, param):
        typename = type(param).__name__
        if hasattr(self, 'from_' + typename):
            return getattr(self, 'from_' + typename)(param)
        elif isinstance(param, ctypes.Array):
            return param
        else:
            raise TypeError("Can't convert %s" % typename)

    # Cast from array.array objects
    def from_array(self, param):
        if param.typecode != 'd':
            raise TypeError('must be an array of doubles')
        ptr, _ = param.buffer_info()
        return ctypes.cast(ptr, ctypes.POINTER(ctypes.c_double))

    # Cast from lists/tuples
    def from_list(self, param):
        val = ((ctypes.c_double)*len(param))(*param)
        return val

    from_tuple = from_list

    # Cast from a numpy array
    def from_ndarray(self, param):
        return param.ctypes.data_as(ctypes.POINTER(ctypes.c_double))

DoubleArray = DoubleArrayType()
_avg = _mod.avg
_avg.argtypes = (DoubleArray, ctypes.c_int)
_avg.restype = ctypes.c_double

def avg(values):
    return _avg(values, len(values))

# struct Point { }
class Point(ctypes.Structure):
    _fields_ = [('x', ctypes.c_double),
                ('y', ctypes.c_double)]

# double distance(Point *, Point *)
distance = _mod.distance
distance.argtypes = (ctypes.POINTER(Point), ctypes.POINTER(Point))
distance.restype = ctypes.c_double
```

åęĆæđIJäÿĂåĹĞæ■čåÿÿïijŇä¡ăářśåŔŕäžëåŁãè¡¡åźűä¡£çŤĺéĞŇéİćåőŽäźĽçŽĎCåĞ¡æŢřäžĘãĂĆä¿ŃåęĆï

```
>>> import sample
>>> sample.gcd(35,42)
7
>>> sample.in_mandel(0,0,500)
1
>>> sample.in_mandel(2.0,1.0,500)
0
>>> sample.divide(42,8)
(5, 2)
>>> sample.avg([1,2,3])
2.0
>>> p1 = sample.Point(1,2)
>>> p2 = sample.Point(4,5)
>>> sample.distance(p1,p2)
4.242640687119285
>>>
```

èőĺèőž

æIJňåřŔèŁĆæIJL'å¿Ĺåď'ŽåĂijå¿ŮæĹŚäžňèŕęçżĘèőĺèőžçŽĎåIJřæŰźãĂĆ éęŰåĚĹæŸŕåŕźäžŐCåŠŇPythonäžčçăĄäÿĂèţůæĽ'ŞåŇĚçŽĎéŮőécŸïijŇåęĆæđIJä¡ăåIJĺä¡£çŤĺ `ctypes` æİèèő£éŮőçijŰèŕŚåŘŐçŽĎCäžčçăĄïijŇ éĆčäźĹéIJĂèęĄçąőä£Ìè£ŹäÿłåĚśäžńåžŞæŤ¿åIJĺ `sample.py` æĺąåİŮåŘŇäÿĂäÿłåIJřæŰźãĂĆ äÿĂçğ■åŔŕèČ¡æŸŕåřĘçŤ§æĹŘçŽĎ . `so` æŰĞäžűæŤ¿ç¡őåIJĺèęĄä¡£çŤĺåőĆçŽĎPythonäžčçăĄåŘŇäÿĂäÿłçŻőå¡ŢäÿŇãĂĆ æĹŚäžňåIJĺ `recipeâĂŤsample.py` äÿ■ä¡£çŤĺ `__file__` åŘŸéĞŔæİèæ§ěçIJŇåőČèćńåőL'èčĚçŽĎä¡■ç¡őïijŇ çĎűåŘŐædĐéĂääÿĂäÿłæŇĞåŘŚåŘŇäÿĂäÿłçŻőå¡Ţäÿ■ç `libsample.so` æŰĞäžűçŽĎèůŕå¿ĎãĂĆ

åęĆæđIJCåĞ¡æŢřäžŞèćńåőL'èčĚåĹřåĚűäžŰåIJřæŰźïijŇéĆčäźĹä¡ăářśèęĄä£őæŤźçŽŷåžŤçŽĎèůŕå¿ĎãĂ åęĆæđIJCåĞ¡æŢřäžŞåIJĺä¡ăæIJžåŹĺäÿŁèćńåőL'èčĚäÿžäÿĂäÿłæăĞåĞĘåžŞäžĘïijŇ éĆčäźĹåŔŕäžëä¡£çŤĺ `ctypes.util.find_library()` åĞ¡æŢřæİèæ§ěæL'¿ïijŽ

```
>>> from ctypes.util import find_library
>>> find_library('m')
'/usr/lib/libm.dylib'
>>> find_library('pthread')
'/usr/lib/libpthread.dylib'
>>> find_library('sample')
'/usr/local/lib/libsample.so'
>>>
```

äÿĂæŮęä¡ăç§ěéĄŞäžĘCåĞ¡æŢřäžŞçŽĎä¡■ç¡őïijŇéĆčäźĹářśåŔŕäžëåČŔäÿŇéİćè£Źæăůä¡£çŤĺ `ctypes.cdll.LoadLibrary()` æİèåŁãè¡¡åőČïijŇ åĚűäÿ■ `_path` æŸŕæăĞåĞĘåžŞçŽĎåĚĺèůŕå¿ĎïijŽ

```
_mod = ctypes.cdll.LoadLibrary(_path)
```

åĞ¡æŢřåžŞèćńåŁàè¡¡åŘŐïijŃä¡ăéIJĂèęĄçijŰåĘŹåĞăäÿłèŕ■åŘěæİěæŔŘåŘŰçĽźåőŽçŽĎçňęåŔůåźűæŇ
åřśåČŔäÿŃéİćè£Źäÿłäžčçă ĄçĽĞæőţäÿĂæăůïijŽ

```
# int in_mandel(double, double, int)
in_mandel = _mod.in_mandel
in_mandel.argtypes = (ctypes.c_double, ctypes.c_double, ctypes.c_
↪int)
in_mandel.restype = ctypes.c_int
```

åIJÍè£Źæőţäžčçă Ąäÿ■ïijŇ .argtypes åśđæĂğæŸŕäÿĂäÿłåĚČçżĎïijŇåŇĚåŘńäžĘæ§ŘäÿłåĞ¡æŢřçŽĎè
èĂŇ .restype åřśæŸŕçŻÿåžŤçŽĎè£ŤåŻđçśżåđŃãĂĆ ctypes
åőŽäźĽäžĘåď ğéĞŔçŽĎçśżåđŃåŕźèśąïijĹæŕŤåęĆc_double, c_int, c_short, c_floatç■ĽïijĽïijŇ
äżčèąĺäžĘåŕźåžŤçŽĎCæŢřæ■őçśżåđŃãĂĆåęĆæđIJä¡ăæČşèőľPythonèČ¡åď§äijăéĂŠæ■ćçąőçŽĎåŔĆæŢřçśż
éĆčäźĹè£ŹäžŻçśżåđŃç■¿åŘ■çŽĎçżŚåőŽæŸŕä¿ĹéG■èęĄçŽĎäÿĂæ■ěãĂĆåęĆæđIJä¡ăæšąæIJĽè£ŹäźĹåĄŽïi
è£Ÿåŕŕèč¡äijŽåŕijèĞť æŢť äÿłèğčéĞŁåŹİè£ŻçĺŃæŇĆæŐĽãĂĆ
ä¡£çŤĺctypesæIJĽäÿĂäÿłéžžçČęçĆźçŽĎåIJřæŰźæŸŕåŐ§çŤ§çŽĎCäžčçăĄä¡£çŤĺçŽĎæIJŕèŕ■åŔŕèČ¡èů§Pytho
divide() åĞ¡æŢřæŸŕäÿĂäÿłå¿Ĺåě¡çŽĎä¿Ńå■ŘïijŇåőČéĂŽè£ĞäÿĂäÿłåŔĆæŢřéŹď äžěåŔęäÿĂäÿłåŔĆæŢ
åř¡çőąè£ŹæŸŕäÿĂäÿłå¿ĹåÿÿèğĄçŽĎCæŁĂæIJŕïijŇä¡ĘæŸŕåIJÍPythonäÿ■å■ť äÿ■ç§ěéĄŞæĂŐæăůæÿĚæŹřç
ä¿Ńå ęĆïijŇä¡ äÿ■èČ¡åČŔäÿŃéİćè£Źæăůç őĂå■ŢçŽĎåĄŽïijŽ

```
>>> divide = _mod.divide
>>> divide.argtypes = (ctypes.c_int, ctypes.c_int, ctypes.
↪POINTER(ctypes.c_int))
>>> x = 0
>>> divide(10, 3, x)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ctypes.ArgumentError: argument 3: <class 'TypeError'>: expected LP_
↪c_int
instance instead of int
>>>
```

åřśçőÙè£Źäÿłèč¡æ■ćçąőçŽĎåůěä¡IJïijŇåőČäijŽè£İåŘ■Pythonåŕźäž Őæ Ţť æŢřçŽĎäÿ■åŔŕæŹť æŤźåŐ§å
åŕźäžŐæűĽåŘŁåĹřæŇĞéŠĹçŽĎåŔĆæŢřïijŇä¡ăéĂŽåÿÿéIJĂèęĄåĚĹæđĐåžžäÿĂäÿłçŻÿåžŤçŽĎctypesåŕźèśą

```
>>> x = ctypes.c_int()
>>> divide(10, 3, x)
3
>>> x.value
1
>>>
```

åIJÍè£ŹéĞŇïijŇäÿĂäÿł ctypes.c_int åőđä¿Ńèćńå ĹŻåžžåźűä¡IJäÿžäÿĂäÿłæŇĞéŠĹèćńäijăè£ŻåŐżã
èů§æŹőéĂŽPythonæŢť å¡ćäÿ■åŘŇçŽĎæŸŕïijŇäÿĂäÿł c_int
åŕźèśąæŸŕåŔŕäžěèćńä£őæŤźçŽĎãĂĆ .value åśđæĂğåŔŕèćńçŤĺæİěèŐůåŔŰæĹŰæŹť æŤźè£Źäÿłå Aij ãĂĆ

åŕźäžŐéĆčäžŻäÿ■åČŔPythonçŽĎCèřČçŤĺïijŇéĂŽåÿÿåŔŕäžěåĘŹäÿĂäÿłåřŔçŽĎåŇĚèčĚåĞ¡æŢřãĂĆ
è£ŹéĞŇïijŇæĹŠäžňèőľ divide() åĞ¡æŢřéĂŽè£ĞåĚČçżĎæİěè£ŤåŻđäÿď äÿłçżŞæđIJïijŽ

```
# int divide(int, int, int *)
_divide = _mod.divide
```

```
_divide.argtypes = (ctypes.c_int, ctypes.c_int, ctypes.
↪POINTER(ctypes.c_int))
_divide.restype = ctypes.c_int

def divide(x, y):
    rem = ctypes.c_int()
    quot = _divide(x,y,rem)
    return quot, rem.value
```

avg() åĞ¡æŢřåŔĹæŸŕäÿĂäÿłæŰřçŽĎæŇŚæĹŸãĂĆCäžčçăĄæIJ§æIJŻæŐěåŔŮåĹřäÿĂäÿłæŇĞéŠĹåŠ
ä¡ĘæŸŕïijŇåIJĺPythonäÿ■ïijŇæĹŚäžňå£Ěéążè ĂČèŹŚè£ŹäÿłéŮőécŸïijŽæŢřçžĎæŸŕåŢěïij§åőČæŸŕäÿĂäÿłåĹ
è£ŸæŸŕ array æĺąåĺŮäÿ■çŽĎäÿĂäÿłæŢřçžĎïij§è£ŸæŸŕäÿĂäÿł numpy
æŢřçžĎïij§è£ŸæŸŕèŕť æĽĂæIJL'éČ¡æŸŕïij§ åőđéŹĚäÿŁïijŇäÿĂäÿłPythonâĂIJæŢřçžĎâĂİæIJL'åď'Žçğ■å¡ćå

DoubleArrayType æijŤçď'žäžĘæĂŐæăůåď'ĎçŘĘè£Źçğ■æČĚåĘţãĂĆ
åIJĺè£Źäÿłçśžäÿ■åőŽäźL'äžĘäÿĂäÿłå■ŢäÿłæŰźæşŢ from_param() ãĂĆ
è£ŹäÿłæŰźæşŢçŽĎèğŠèL'šæŸŕæŐěåŔŮäÿĂäÿłå■ŢäÿłåŔĆæŢřçĎűåŘŐářĘåĚűåŘŚäÿŃè¡ňæ■ćäÿžäÿĂäÿłåŘĺ
ïijĹæIJňä¿Ńäÿ■æŸŕäÿĂäÿł ctypes.c_double çŽĎæŇĞéŠĹïijL'ãĂĆ
åIJĺ from_param() äÿ■ïijŇä¡ăåŔŕäžěåĄŽäżżä¡Ţä¡ăæČşåĄŽçŽĎäžŃãĂĆ
åŔĆæŢřçŽĎçśžåđŃåŘ■èćńæŔŘåŔŮåĞžæİěåźűèćńçŤĺäžŐåĹĘåŚŚåĹřäÿĂäÿłæŻť'åĚůä¡ŞçŽĎæŰźæşŢäÿ■åŐ
ä¿ŃåęĆïijŇåęĆæđIJäÿĂäÿłåĹŮèąĺèćńäijăéĂŠè£ĞæİěïijŇéĆčäźĹ typename åŕśæŸŕ list
ïijŇ çĎűåŘŐ from_list æŰźæşŢèćńèřĆçŤĺãĂĆ

åŕźäžŐåĹŮèąĺåŠŇåĚČçżĎïijŇfrom_list æŰźæşŢåřĘåĚűè¡ňæ■ćäÿžäÿĂäÿł ctypes
çŽĎæŢřçžĎåŕźèśąãĂĆ è£Źäÿłç IJŃäÿŁåŐżæIJL'çĆźåěĞæĂłïijŇäÿŃéİćæĹŚäžňä¡£çŤĺäÿĂäÿłäžď'äžŠåijŔä¿Ń
ctypes æŢřçžĎïijŽ

```
>>> nums = [1, 2, 3]
>>> a = (ctypes.c_double * len(nums))(*nums)
>>> a
<__main__.c_double_Array_3 object at 0x10069cd40>
>>> a[0]
1.0
>>> a[1]
2.0
>>> a[2]
3.0
>>>
```

åŕźäžŐæŢřçžĎåŕźèśąïijŇfrom_array() æŔŘåŔŮåžŢåśĆçŽĎåĘĚå■ŸæŇĞéŠĹåźűåřĘåĚűè¡ňæ■ćäÿž
ctypes æŇĞéŠĹåŕźèśąãĂĆä¿ŃåęĆïijŽ

```
>>> import array
>>> a = array.array('d',[1,2,3])
>>> a
array('d', [1.0, 2.0, 3.0])
>>> ptr_ = a.buffer_info()
>>> ptr
4298687200
>>> ctypes.cast(ptr, ctypes.POINTER(ctypes.c_double))
<__main__.LP_c_double object at 0x10069cd40>
```

```
>>>
```

from_ndarray() æijŤçď'žäžĘåŕźäžŐ numpy æŢřçżĎçŽĎè¡ňæ■ćæŞ■ä¡IJãĂĆ éĂŽè£ĞåőŽäźĽ' DoubleArrayType çśžåźűåIJÍ avg() çśžåđŃç■¿åŘ■äÿ■ä¡£çŤĺåőČïijŇ éĆčäźĹè£ŹäÿłåĞ¡æŢřåřśèČ¡æŐěåŔŮåď'Žäÿłäÿ■åŘŇçŽĎçśžæŢřçżĎè¿ŞåĚěäžĘïijŽ

```
>>> import sample
>>> sample.avg([1,2,3])
2.0
>>> sample.avg((1,2,3))
2.0
>>> import array
>>> sample.avg(array.array('d',[1,2,3]))
2.0
>>> import numpy
>>> sample.avg(numpy.array([1.0,2.0,3.0]))
2.0
>>>
```

æIJňèŁĆæIJĂåŘŐäÿĂéČĺåĹĘåŘŚä¡ăæijŤçď'žäžĘæĂŐæăůåď'ĎçŘĘäÿĂäÿłçőĂå■ŢçŽĎCçżŞædĎãĂĆ åŕźäžŐçżŞædĎä¡ŞïijŇä¡ăåŘłéIJÄèęĄåČŔäÿŃéÍćè£ŹæăůçőĂå■ŢçŽĎåőŽäźĽäÿĂäÿłçśžïijŇåŇĚåŘńçŻÿåžŤç

```
class Point(ctypes.Structure):
    _fields_ = [('x', ctypes.c_double),
                ('y', ctypes.c_double)]
```

äÿĂæŮęçśžèćńåőŽäźĽåŘŐïijŇä¡ăåřśåŔŕäžěåIJÍçśžåđŃç■¿åŘ■äÿ■æĹŰèĂĚæŸŕéIJÄèęĄåőđä¿ŃåŇŰçżŞ

```
>>> p1 = sample.Point(1,2)
>>> p2 = sample.Point(4,5)
>>> p1.x
1.0
>>> p1.y
2.0
>>> sample.distance(p1,p2)
4.242640687119285
>>>
```

æIJĂåŘŐäÿĂäžŽåřŔçŽĎæŔŘçď'žïijŽåęĆæđIJä¡ăæČşåIJÍPythonäÿ■èő£éŮőäÿĂäžŽåřŔçŽĎCåĞ¡æŢřïijŇ ctypes æŸŕäÿĂäÿłå¿ĹæIJL'çŤÍçŽĎåĞ¡æŢřåžŞãĂĆ år¡çőąåęĆæ■d'ïijŇåęĆæđIJä¡ăæČşèęĄåŐżèő£éŮőäÿĂă Swig (15.9èŁĆäijŽèőšåĹř) æĹŰ CythonïijĹ15.10èŁĆïijL'ãĂĆ

åŕźäžŐåď'ğåđŃåžŞçŽĎèő£éŮőæIJL'äÿłäÿżèęĄéŮőéćŸïijŇçŤśäžŐctypesåźűäÿ■æŸŕåőŇåĚÍèĞłåŁÍåŇŰï éĆčäźĹä¡ăåřśå£Ěéążè£śèť'źåď'ğéĞŔæŮűéŮť'æÍèçijŰåĘŹæL'ĂæIJL'çŽĎçśžåđŃç■¿åŘ■ïijŇåřśåČŔä¿Ńå■Řäÿ åęĆæđIJåĞ¡æŢřåžŞåď'§åď'■æÍĆïijŇä¡ăè£Ÿå¿ŮåŐżçijŰåĘŹå¿Ĺåď'ŽåřŔçŽĎåŇĚèčĚåĞ¡æŢřåŠŇæŤŕæŇĄçśž åŘęåď'ŰïijŇéŹď'éÍđä¡ăåůšçżŔåőŇåĚÍçš¿éĂŽäžĘæL'ĂæIJL'ăžŢåśĆçŽĎCæŐěåŔčçżĘèŁĆïijŇåŇĚæŃńåĘĚå éĂŽåÿÿäÿĂäÿłå¿ĹåřŔçŽĎäžčçăĄçijžéŹůãĂĄèő£éŮőèűŁçŢŇæĹŰåĚűäžŰçśžäijjéŤŹèŕŕåřśèČ¡èőľ'PythonçÍŃ

ä¡IJäÿž ctypes çŽĎäÿĂäÿłæŻ£äžčïijŇä¡ăè£ŸåŔŕäžěèĂČèŹŠäÿŃCFFIãĂĆCFFIæŔŘä¿ŻäžĘå¿Ĺåď'Žçś ä¡ĘæŸŕä¡£çŤÍCèŕ■æşŢåźűæŤŕæŇĄæŻt'åď'ŽéńŸçžğçŽĎCäžčçăĄçśžåđŃãĂĆ åĹřåĘŹè£ŹæIJñäźęäÿžæ■ćïijŃCFFIè£ŸæŸŕäÿĂäÿłçŻÿåŕźè¿ČæŰřçŽĎåůěçÍŃïijŇ

ä¡ĘæŸŕåőČçŽĎæţĄèąŇåžęæ■čåIJĺå£ńéĂ§äÿŁå■ĞãĂĆ çŤŽèĞşè£ŸæIJL'åIJĺèőĺèőžåIJĺPythonåřĘæĺěçŽĎçL'ĺ

15.2 çőĂå■ŢçŽĎCæĽľåśŢæĺąåİŮ

éŮőécŸ

ä¡ăæČşäÿ■ä¿ĺéĺăåĚűäżŰåůěåĚůïijŇçŻť'æŐěä¡£çŤĺPythonçŽĎæĽľåśŢAPIæĺěçijŰåĘŹäÿĂäžŽçőĂå■ŢçŽ

èğčåĘşæŰźæąĹ

åŕźäžŐçőĂå■ŢçŽĎCäżčçăĄïijŇæđĎåżžäÿĂäÿłèĞłåőŽäźĽæĽľåśŢæĺąåİŮæŸŕå¿ĹåőźæŸŞçŽĎãĂĆ
ä¡IJäÿžçňňäÿĂæ■ëïijŇä¡ăéIJĂèęĄçąőä£Ĺä¡ăçŽĎCäżčçăĄæIJL'äÿĂäÿłæ■čçąőçŽĎåď'æŰĞäžűãĂĆä¿ŇåęĆïij

```
/* sample.h */

#include <math.h>

extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

éĂŽåÿÿæĺěèőšïijŇè£Źäÿłåď'æŰĞäžűèęĄåŕźåžŤäÿĂäÿłåůšçżŔèćńå■ŢçŇňçijŰèŕŚè£ĞçŽĎåžŞãĂĆ
æIJL'äžĘè£ŹäžŻïijŇäÿŇéĺćæĹŚäžňæijŤçď'žäÿŇçijŰåĘŹæL'ľåśŢåĞ¡æŢřçŽĎäÿĂäÿłçőĂå■Ţä¿Ňå■ŘïijŽ

```
#include "Python.h"
#include "sample.h"

/* int gcd(int, int) */
static PyObject *py_gcd(PyObject *self, PyObject *args) {
  int x, y, result;

  if (!PyArg_ParseTuple(args,"ii", &x, &y)) {
    return NULL;
  }
  result = gcd(x,y);
  return Py_BuildValue("i", result);
}

/* int in_mandel(double, double, int) */
static PyObject *py_in_mandel(PyObject *self, PyObject *args) {
```

```
  double x0, y0;
  int n;
  int result;

  if (!PyArg_ParseTuple(args, "ddi", &x0, &y0, &n)) {
    return NULL;
  }
  result = in_mandel(x0,y0,n);
  return Py_BuildValue("i", result);
}

/* int divide(int, int, int *) */
static PyObject *py_divide(PyObject *self, PyObject *args) {
  int a, b, quotient, remainder;
  if (!PyArg_ParseTuple(args, "ii", &a, &b)) {
    return NULL;
  }
  quotient = divide(a,b, &remainder);
  return Py_BuildValue("(ii)", quotient, remainder);
}

/* Module method table */
static PyMethodDef SampleMethods[] = {
  {"gcd",  py_gcd, METH_VARARGS, "Greatest common divisor"},
  {"in_mandel", py_in_mandel, METH_VARARGS, "Mandelbrot test"},
  {"divide", py_divide, METH_VARARGS, "Integer division"},
  { NULL, NULL, 0, NULL}
};

/* Module structure */
static struct PyModuleDef samplemodule = {
  PyModuleDef_HEAD_INIT,

  "sample",           /* name of module */
  "A sample module",  /* Doc string (may be NULL) */
  -1,                 /* Size of per-interpreter state or -1 */
  SampleMethods       /* Method table */
};

/* Module initialization function */
PyMODINIT_FUNC
PyInit_sample(void) {
  return PyModule_Create(&samplemodule);
}
```

èęĄçżŚåőŽè£ŹäÿłæĽľåśŢæĺąåİŮïijŇåČŔäÿŃéİćè£ŹæăůåĹŻåžžäÿĂäÿł `setup.py` æŰĞäżűïijŽ

```
# setup.py
from distutils.core import setup, Extension
```

```
setup(name='sample',
      ext_modules=[
        Extension('sample',
                  ['pysample.c'],
                  include_dirs = ['/some/dir'],
                  define_macros = [('FOO','1')],
                  undef_macros = ['BAR'],
                  library_dirs = ['/usr/local/lib'],
                  libraries = ['sample']
                  )
        ]
)
```

äÿžäžĘæđĎåžžæIJĂçżĹçŽĎåĞ¡æŢřåžŞïijŇåŘłéIJĂçőĂå■ŢçŽĎä¡£çŤÍ `python3 buildlib.py build_ext --inplace` åŚ¡äžďå■şåŘřïijŽ

```
bash % python3 setup.py build_ext --inplace
running build_ext
building 'sample' extension
gcc -fno-strict-aliasing -DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-
 ↪prototypes
 -I/usr/local/include/python3.3m -c pysample.c
 -o build/temp.macosx-10.6-x86_64-3.3/pysample.o
gcc -bundle -undefined dynamic_lookup
build/temp.macosx-10.6-x86_64-3.3/pysample.o \
 -L/usr/local/lib -lsample -o sample.so
bash %
```

åęĆäÿŁæĽĂçďžïijŇåőČäijŽåĹŻåžžäÿĂäÿłåŘ■å■ŮåŘń `sample.so` çŽĎåĚśäžńåžŞãĂĆå¡Şèćńçij Ůèŕ ŚåŘŐïijŇä¡ääřśèČ¡ářĘåőČä¡IJäÿžäÿĂäÿłæĺąåİŮåŕijåĚèè£ŻæİěäžĘïijŽ

```
>>> import sample
>>> sample.gcd(35, 42)
7
>>> sample.in_mandel(0, 0, 500)
1
>>> sample.in_mandel(2.0, 1.0, 500)

0
>>> sample.divide(42, 8)
(5, 2)
>>>
```

åęĆædIJä¡ăæŸŕåIJĺWindowsæIJžåŹĺäÿŁéİćåřİèŕŢè£ŹäžŻæ■ěéłď¨ïijŇåŘŕèČ¡äijŽéĄĞåĹřåŘĎçğ■çŐŕáćČ PythonçŽĎäžŇè£ŻåĹűåĹĘåŘŚéĂŽåÿÿä¡£çŤĺäžĘMicrosoft Visual StudioæİěædĎåžžãĂĆ äÿžäžĘèőľè£ŹäžŻæĽ'ľåśŢèČ¡æ■čåÿÿåůěä¡IJïijŇä¡äéIJĂèęĄä¡£çŤĺåŘŇæăůæĹŰåĚijåőźçŽĎåůěåĚůæİěçijŰè åŘĆèĂČçŻÿäžŤçŽĎ PythonæŰĞæąč

èőĺèőž

åIJĺåřİèŕŢäżżä¡ŢæĽ'ŃåĘŹæĽ'ľåśŢäźŃåĽ'■ïijŇæIJĂåě¡èČ¡åĚĹåŔĆèĂČäÿŃPythonæŰĞæąčäÿ■çŽĎ
æĽ'ľåśŢåŠŇåţŇåĚěPythonèğčéĞŁåŹĺ . PythonçŽĎCæĽ'ľåśŢAPIå¿Ĺåď'ğïijŇåIJĺè£ŹéĞŇæŢť'äÿłåŐżèőšè£řåđ
äÿ■è£Ğåŕźäž0ŒæIJĂæăÿå£ČçŽĎéČĺåĹĘè£ŸæŸŕåŔŕäżěèőĺèőžäÿŃçŽĎãĂĆ

éęŰåĚĹïijŇåIJĺæĽ'ľåśŢæĺąåİŮäÿ■ïijŇä¡ååĘŹçŽĎåĞ¡æŢřéČ¡æŸŕåČŔäÿŃéİćè£ŹæăůçŽĎäÿĂäÿłæŹőéĂ

```
static PyObject *py_func(PyObject *self, PyObject *args) {
  ...
}
```

PyObject æŸŕäÿĂäÿłèČ¡èąĺçd'žäżżä¡ŢPythonåŕźèśąçŽĎCæŢřæ■őçśżåđŃãĂĆ
åIJĺäÿĂäÿłéńŸçžğåśĆéİćïijŇäÿĂäÿłæĽ'ľåśŢåĞ¡æŢřåřśæŸŕäÿĂäÿłæŐěåŔŮäÿĂäÿłPythonåŕźèśą
ïijĹåIJĺ PyObject *argsäÿ■ïijĽåĚČçżĎåźűè£ŤåŽđäÿĂäÿłæŰřPythonåŕźèśąçŽĎCåĞ¡æŢřãĂĆ
åĞ¡æŢřçŽĎ self åŔĆæŢřåŕźäžŐçőĂå■ŢçŽĎæĽ'ľåśŢåĞ¡æŢřæšąæIJL'èćńä¡£çŤĺåĹřïijŇ
äÿ■è£ĞåęĆæđIJä¡ăæČşåőŽäźĽ'æŰřçŽĎçśżæĹŰèĂĚæŸŕCäÿ■çŽĎåŕźèśąçśżåđŃçŽĎèŕİåřśèČ¡æť¿äÿŁçŤĺåIJž
éĆčäźĹ self åřśèČ¡åijŢçŤĺéĆčäÿłåődä¿ŃäžĘãĂĆ

PyArg_ParseTuple() åĞ¡æŢřèćńçŤĺæİěåřĘPythonäÿ■çŽĎåĂijè¡ňæ■ćæĹŘCäÿ■åŕźåžŤèąĺçd'žãĂĆ
åőČæŐěåŔŮäÿĂäÿłæŇĞåőŽè¿ŞåĚěæăijåijŔçŽĎæăijåijŔåŇŰå■Ůçňęäÿšä¡IJäÿžè¿ŞåĚěïijŇærŤåęĆâĂIJiâĂİ
åŘŇæăůè£ŸæIJL'å■ŸæŤ¿è¡ňæ■ćåŘŐçżŞæđIJçŽĎCåŔŸéĞŔçŽĎåIJřåİĂãĂĆ
åęĆæđIJè¿ŞåĚěçŽĎåĂijäÿ■åŇźéĚ■è£ŹäÿłæăijåijŔåŇŰå■ŮçňęäÿšïijŇářšäijŽæŁŻåĞžäÿĂäÿłåijĆåÿÿåźűè£Ť
éĂŽè£ĞæčĂæ§ěåźűè£ŤåŽđNULLïijŇäÿĂäÿłåŘĹéĂĆçŽĎåijĆåÿÿäijŽåIJĺèŕĈçŤĺäžčçăĄäÿ■èćńæŁŻåĞžãĂĆ

Py_BuildValue() åĞ¡æŢřèćńçŤĺæİěæăźæ■őCæŢřæ■őçśżåđŃåĹŻåžžPythonåŕźèśąãĂĆ
åőČåŘŇæăůæŐěåŔŮäÿĂäÿłæăijåijŔåŇŰå■ŮçňęäÿšæİěæŇĞåőŽæIJ§æIJŻçśżåđŃãĂĆ
åIJĺæĽ'ľåśŢåĞ¡æŢřäÿ■ïijŇåőČèćńçŤĺæİěè£ŤåŽđçżŞæđIJçŽŹPythonãĂĆ
Py_BuildValue() çŽĎäÿĂäÿłçĽźæĂğæŸŕåőČèČ¡æđĎåžžæŽť'åŁăåď'■æİĆçŽĎåŕźèśąçśżåđŃïijŇærŤåęĆ
åIJĺ py_divide() äžčçăĄäÿ■ïijŇäÿĂäÿłä¿Ńå■ŘæijŤçď'žäžĘæĂŐæăůè£ŤåŽđäÿĂäÿłåĚČçżĎãĂĆäÿ■è£Ğï

```
return Py_BuildValue("i", 34);       // Return an integer
return Py_BuildValue("d", 3.4);      // Return a double
return Py_BuildValue("s", "Hello"); // Null-terminated UTF-8 string
return Py_BuildValue("(ii)", 3, 4); // Tuple (3, 4)
```

åIJĺæĽ'ľåśŢæĺąåİŮåžŢéČĺïijŇä¡ăäijŽåŔŚçŐřäÿĂäÿłåĞ¡æŢřèąĺïijŇærŤåęĆæIJňèŁĆäÿ■çŽĎ
SampleMethods èąĺãĂĆ è£ŹäÿłèąĺåŔŕäžěåĽŮåĞžCåĞ¡æŢřãĂĄPythonäÿ■ä¡£çŤĺçŽĎåŘ■å■ŮãĂĄæŰĞæą
æĽ'ĂæIJĽæĺąåİŮéČ¡éIJĂèęĄæŇĞåőŽè£ŹäÿłèąĺïijŇåŽääÿžåőČåIJĺæĺąåİŮåĹİåğŃåŇŰæŮűèęĄęćńä¡£çŤĺåĹř

æIJĂåŘŐçŽĎåĞ¡æŢř PyInit_sample() æŸŕæĺąåİŮåĹİåğŃåŇŰåĞ¡æŢřïijŇä¡ĘèŕěæĺąåİŮçňňäÿĂæì
è£ŹäÿłåĞ¡æŢřçŽĎäÿżèęĄåůěä¡IJæŸŕåIJĺèğčéĞŁåŹĺäÿ■æşĺåĘŇæĺąåİŮåŕźèśąãĂĆ

æIJĂåŘŐäÿĂäÿłèęĄçĆźéIJĂèęĄæŔŘåĞžæİěïijŇä¡£çŤĺCåĞ¡æŢřæİěæĽ'ľåśŢPythonèęĄèĂČèŹŚçŽĎäžl
ïijĹåőđéŹĚäÿŁïijŇC APIåŇĚåŘńäžĘèűĚè£Ğ500äÿłåĞ¡æŢřïijL'ãĂĆä¡ăåžŤèŕěåŕĘæIJňèŁĆå¡ŞåĄŽæŸŕäÿĂäÿ
æŻť'åď'ŽéńŸçžğåĘĚåőźïijŇåŔŕäžěçIJŃçIJŃ PyArg_ParseTuple() åŠŇ
Py_BuildValue() åĞ¡æŢřçŽĎæŰĞæąčïijŇ çĎűåŘŐè£ŻäÿĂæ■ěæĽ'ľåśŢåijĂãĂĆ

15.3 çijŰåĘŹæĽľåśŢåĞ¡æŢřæŞ■ä¡IJæŢřçžĎ

éůőécŸ

ä¡ăæČşçijŰåĘŹäÿĂäÿłCæĽľåśŢåĞ¡æŢřæİěæŞ■ä¡IJæŢřçžĎïijŇåŕřèČ¡æŸŕèćńarrayæĺąåİŮæĹŰçśžäijijN
äÿ■è£ĞïijŇä¡ăæČşèőľä¡ăçŽĎåĞ¡æŢřæŻť åŁăéĂŽçŤĺïijŇèĂŇäÿ■æŸŕéŠĹåŕźæ§ŘäÿłçĽźåőŽçŽĎåžŞæĽĂçŤ

èğčåĘşæŰźæąĹ

äÿžäžĘèČ¡èőľæŐěåŔŮåŠŇåď ĎçŘĘæŢřçžĎåĚůæIJĽåŔŕçğżæď■æĂğïijŇä¡ăéIJĂèęĄä¡£çŤĺåĹř *Buffer Protocol* . äÿŃéİćæŸŕäÿĂäÿłæĽŃåĘŹçŽĎCæĽľåśŢåĞ¡æŢřä¿Ńå■ŘïijŇ çŤĺæİěæŐěåŔŮæŢřçžĎæŢřæ■őåźűèřČçŤĺæIJňçńăåijĂçŕĞéČĺåĹĘçŽĎ `avg(double *buf, int len)` åĞ¡æŢřïijŽ

```
/* Call double avg(double *, int) */
static PyObject *py_avg(PyObject *self, PyObject *args) {
  PyObject *bufobj;
  Py_buffer view;
  double result;
  /* Get the passed Python object */
  if (!PyArg_ParseTuple(args, "O", &bufobj)) {
    return NULL;
  }

  /* Attempt to extract buffer information from it */

  if (PyObject_GetBuffer(bufobj, &view,
      PyBUF_ANY_CONTIGUOUS | PyBUF_FORMAT) == -1) {
    return NULL;
  }

  if (view.ndim != 1) {
    PyErr_SetString(PyExc_TypeError, "Expected a 1-dimensional array
↪");
    PyBuffer_Release(&view);
    return NULL;
  }

  /* Check the type of items in the array */
  if (strcmp(view.format,"d") != 0) {
    PyErr_SetString(PyExc_TypeError, "Expected an array of doubles
↪");
    PyBuffer_Release(&view);
    return NULL;
  }

  /* Pass the raw buffer and size to the C function */
  result = avg(view.buf, view.shape[0]);
```

```
  /* Indicate we're done working with the buffer */
  PyBuffer_Release(&view);
  return Py_BuildValue("d", result);
}
```

äÿŃéÌćæĹŚäžňæijŤçd'žäÿŃè£ŹäÿłæĽl'åśŢåĞ¡æŢřæŸŕåęĆä¡Ţåůëä¡IJçŽĎïijŽ

```
>>> import array
>>> avg(array.array('d',[1,2,3]))
2.0
>>> import numpy
>>> avg(numpy.array([1.0,2.0,3.0]))
2.0
>>> avg([1,2,3])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'list' does not support the buffer interface
>>> avg(b'Hello')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Expected an array of doubles
>>> a = numpy.array([[1.,2.,3.],[4.,5.,6.]])
>>> avg(a[:,2])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: ndarray is not contiguous
>>> sample.avg(a)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Expected a 1-dimensional array
>>> sample.avg(a[0])

2.0
>>>
```

èőÍèőž

åřĘäÿĂäÿłæŢřçžĎåŕźèśąäijåçžŹCåĞ¡æŢřåŔŕèČ¡æŸŕäÿĂäÿłæĽl'åśŢåĞ¡æŢřåĄŽçŽĎæIJĂåÿÿèğĄçŽĎäž
å¿Ĺåď'ŽPythonåžŤçŤÍçÍŃåžŔïijŇäżŐåŽ¿åČŔåď'ĎçŘĘåĹřçğŚå■ęèőąçőŮïijŇéČ¡æŸŕå§žäžŐéńŸæĂğèČ¡çŽĎ
éĂŽè£ĞçijŰåĘŹèČ¡æŐěåŔŮåźűæŞ■ä¡IJæŢřçžĎçŽĎäžčçĂąïijŇä¡ăåŔŕäžěçijŰåĘŹå¿Ĺåě¡çŽĎåĚijåőźè£ŹäžŽ
èĂŃäÿ■æŸŕåŔłèČ¡åĚijåőźä¡äèĞłåůśçŽĎäžčçĂąãĂĆ

äžčçĂąçŽĎåĚşéŤőçĆźåIJĺäžŐ `PyBuffer_GetBuffer()` åĞ¡æŢřãĂĆ
çžŹåőŽäÿĂäÿłäžżæĎŔçŽĎPythonåŕźèśąïijŇåőČäijŽèŕŢçİĂåŐżèŐůåŔŰåžŢåśĆåĘĚå■Ÿä£ąæĄŕïijŇåőČçőĂå
1. äijăçžŹ `PyBuffer_GetBuffer()` çŽĎçĽźæőŁæăĞå£ŰçżŹåĞžäžĘæĽĂéIJĂçŽĎåĘĚå■ŸçijŞåĘšçśžåč
ä¿ŃåęĆïijŇ`PyBUF_ANY_CONTIGUOUS` èąÍçd'žæŸŕäÿĂäÿłèĄŢçşžçŽĎåĘĚå■ŸåŇžå§§ãĂĆ

åŕźäžŐæŢřçžĎãĂĄå■ŮèŁĆå■Ůçňęäÿšå$ŇåĚűäžŰçśžäijijåŕźèśąèĂŇèÍĂïijŇäÿĂäÿł

Py_buffer çżŞæđĎä¡ŞåŇĚåŘńäžĘæĽĂæIJL'åžŢåśĆåĘĚå■ŸçŽĎä£ąæĄŕãĂĆ
åőČåŇĚåŘńäÿĂäÿłæŇĞåŘŚåĘĚå■ŸåIJŕåİĂãĂĄåď ğåŕŔãĂĄåĚČçť ăåď ğåŕŔãĂĄæăijåijŔåŠŇåĚűäžŰçżĘèŁ

```
typedef struct bufferinfo {
    void *buf;              /* Pointer to buffer memory */
    PyObject *obj;          /* Python object that is the owner */
    Py_ssize_t len;         /* Total size in bytes */
    Py_ssize_t itemsize;    /* Size in bytes of a single item */
    int readonly;           /* Read-only access flag */
    int ndim;               /* Number of dimensions */
    char *format;           /* struct code of a single item */
    Py_ssize_t *shape;      /* Array containing dimensions */
    Py_ssize_t *strides;    /* Array containing strides */
    Py_ssize_t *suboffsets; /* Array containing suboffsets */
} Py_buffer;
```

æIJňèŁĆäÿ■ïijŇæĹŚäžňåŔłåĚşæşĺæŐěåŔŮäÿĂäÿłåŔŇçš¿åžęæţőçĆźæŢřæŢřçżĎä¡IJäÿžåŔĆæŢřãĂĆ
èęĄæčĂæ§ěåĚČçť ăæŸŕåŘęæŸŕäÿĂäÿłåŔŇçš¿åžęæţőçĆźæŢřïijŇåŔłéIJĂéłŇèŕĄ
format åśđæĂğæŸŕäÿ■æŸŕå■Ůçňęäÿš"d". è£ŹäÿłäźšæŸŕ struct
æĺąåİŮçŤĺæİěçijŰçăĄäžŇè£ŻåĹűæŢřæ■őçŽĎãĂĆ éĂŽåÿÿæİěèõšïijŇformat
åŘŕäžěæŸŕäżä¡ŢåĚijåőź struct æĺąåİŮçŽĎæăijåijŔåŇŰå■ŮçňęäÿšïijŇ
åźűäÿŤåęĆæđIJæŢřçżĎåŇĚåŘńäžĘCçżŞæđĎçŽĎèŕİåőČåŘŕäžěåŇĚåŘńåď Žäÿłå Äijã ĂĆ
äÿĂæŮęæĹŚäžňåůšçżŔçąőåőŽäžĘåžŢåśĆçŽĎçijŞå■ŸåŇžä£ąæĄŕïijŇéĆčåŘłéIJĂèęĄçőĂå■ŢçŽĎårĘåőČäij
åődéŹĚäÿŁïijŇæĹŚäžňäÿ■å£ĚæŃĚå£ĆæŸŕæĂŐæăůçŽĎæŢřçżĎçśžđŃæĹŮèĂĚåőĆæŸŕèćńäžĂäźĹåžŞåĹŽ
è£ŹäźšæŸŕäÿžäżĂäźĹè£ŹäÿłåĞ¡æŢřèČ¡åĚijåőź array æĺąåİŮäźš§èČ¡åĚijåőź numpy
æĺąåİŮäÿ■çŽĎæŢřçżĎäžĘãĂĆ

åIJÍè£ŤåŻđæIJĂçżĹçżŞæđIJäźŃåL'■ïijŇäžŢåśĆçŽĎçijŞåĘšåŇžèğĘåŻ¿å£Ěéążä¡£çŤÍ
PyBuffer_Release() éĞŁæŤ¿æŐL'ãĂĆ äźŃæL'Ăäżěèę Ąè£ŹäÿĂæ■ěæŸŕäÿžäžĘèČ¡æ■ĉçąőçŽĎçőąçŘĚ

åŘŇæăůïijŇæIJňèŁĆäź§äžĚäžĚåŔłæŸŕæijŤçď žäžĘæŐěåŘŮæŢřçżĎçŽĎäÿĂäÿłåŕŔçŽĎäżĉçăĄçL'Ğæő
åęĆæđIJä¡ăçIJ§çŽĎèęĄåď ĎçŘĘæŢřçżĎïijŇä¡ăåŕèČ¡äijŽçćřåĹřåď Žçźť æŢřæ■őãĂĄåď ğæŢřæ■őãĂĄäÿ■å
éĆĉäźĹåřšå¿ŮåŐżå■ęæŻť énŸçžğçŽĎäÿIJèě£äžĘãĂĆä¡áéIJĂèęĄåŔĆèĂČåőŸæŰźæŰĞæąćæİěèŐůåŘŰæŻť

åęĆæđIJä¡ăéIJĂèęĄçijŰåĘŹæűL'åŔŁåĹřæŢřçżĎåď ĎçŘĘçŽĎåď Žäÿłæl'l'åśŢïijŇéĆčäźĹéĂŽè£ĞCytho

15.4 åIJÍCæL'ľåśŢæĺąåİŮäÿ■æŞ■ä¡IJéŽŘå¡ćæŇĞéŠĹ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłæL'l'åśŢæĺąåİŮéIJĂèęĄåď ĎçŘĘCçżŞæđĎä¡Şäÿ■çŽĎæŇĞéŠĹïijŇ
ä¡ĘæŸŕä¡ăåŔĹäÿ■æČşæŽť éIJšçżŞæđĎä¡Şäÿ■äżżä¡ŢåĘĚéČĺçżĘèŁĆçżŹPythonãĂĆ

èğčåĘşæŰźæąĹ

éŽŘå¡ćçżŞæđĎä¡Şåŕŕäžěå¿ĹåőźæŸ§çŽĎéĂŽè£ĞářĘåőČäžňåŇĚèčĚåIJĺèČűåŽŁåŕźèśąäÿ■æİěåď ĎçŘĘ
èĂČèŹŚæĹŚäžňä¿Ňå■Řäžčçăąäÿ■çŽĎäÿŇåĹŮCäžčçăĄçL'Ğæőţïijž

```
typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

äÿŃéİćæŸŕäÿĂäÿłä¡£çŤĺèČűåŻŁåŇĚèčĚPointçżŞæđĎä¡ŞåŠŇ `distance()` åĞ¡æŢřçŽĎæĽľåśŢäžčçăĄåőđä¿ŃïijŽ

```
/* Destructor function for points */
static void del_Point(PyObject *obj) {
  free(PyCapsule_GetPointer(obj,"Point"));
}

/* Utility functions */
static Point *PyPoint_AsPoint(PyObject *obj) {
  return (Point *) PyCapsule_GetPointer(obj, "Point");
}

static PyObject *PyPoint_FromPoint(Point *p, int must_free) {
  return PyCapsule_New(p, "Point", must_free ? del_Point : NULL);
}

/* Create a new Point object */
static PyObject *py_Point(PyObject *self, PyObject *args) {

  Point *p;
  double x,y;
  if (!PyArg_ParseTuple(args,"dd",&x,&y)) {
    return NULL;
  }
  p = (Point *) malloc(sizeof(Point));
  p->x = x;
  p->y = y;
  return PyPoint_FromPoint(p, 1);
}

static PyObject *py_distance(PyObject *self, PyObject *args) {
  Point *p1, *p2;
  PyObject *py_p1, *py_p2;
  double result;

  if (!PyArg_ParseTuple(args,"OO",&py_p1, &py_p2)) {
    return NULL;
  }
  if (!(p1 = PyPoint_AsPoint(py_p1))) {
    return NULL;
  }
  if (!(p2 = PyPoint_AsPoint(py_p2))) {
    return NULL;
```

```
  }
  result = distance(p1,p2);
  return Py_BuildValue("d", result);
}
```

åIJĺPythonäÿ■åŔŕäžěåČŔäÿŃéIćè£ŹæăůæIěä¡£çŤĺè£ŹäžŻåĞ¡æŢřïijŽ

```
>>> import sample
>>> p1 = sample.Point(2,3)
>>> p2 = sample.Point(4,5)
>>> p1
<capsule object "Point" at 0x1004ea330>
>>> p2
<capsule object "Point" at 0x1005d1db0>
>>> sample.distance(p1,p2)
2.8284271247461903
>>>
```

èőĺèőž

èČűåŻŁåŠŇCæŇĞéŠĹçśżäijijãĂĆåIJĺåĘĚéČĺïijŇåőČäžňèŐůåŔŰäÿĂäÿłéĂŽçŤĺæŇĞéŠĹåŠŇäÿĂäÿłåŔ
`PyCapsule_New()` åĞ¡æŢřå¿Ĺåőźæŷ§çŽĎèćńåĹŻåžžãĂĆ
åŔęåď ŰïijŇäÿĂäÿłåŔŕéĂĽçŽĎæđŘæđĎåĞ¡æŢřèČ¡èćńçžŠåőŽåĹřèČűåŻŁäÿŁïijŇçŤĺæIěåIJĺèČűåŻŁåŕźèśąè

èęĄæŔŘåŔŰèČűåŻŁäÿ■çŽĎæŇĞéŠĹïijŇåŔŕä¡£çŤĺ `PyCapsule_GetPointer()`
åĞ¡æŢřåźűæŇĞåőŽåŘ■çğřãĂĆ åęĆæđIJæŔŘä¿ŻçŽĎåŘ■çğřåŠŇèČűåŻŁäÿ■åŇźéĚ■æĹŰåĚűäžŰéŤŹèŕŕåĞž

æIJňèŁĆäÿ■ïijŇäÿĂåŕźåůěåĚůåĞ¡æŢřâĂŤâĂŤ `PyPoint_FromPoint()`
åŠŇ `PyPoint_AsPoint()` èćńçŤĺæIěåĹŻåžžåŠŇäžŐèČűåŻŁåŕźèśąäÿ■æŔŘåŔŰ-
Pointåőđä¿ŃãĂĆ åIJĺäžżä¡ŢæĽľåśŢåĞ¡æŢřäÿ■ïijŇæĹŚäžňäijŽä¡£çŤĺè£ŹäžŻåĞ¡æŢřèĂŇäÿ■æŸŕçŻť æŐěä¡
è£Źçğ■èő¿èőąä¡£å¿ŮæĹŚäžňåŔŕäžěå¿Ĺåőźæŷ§çŽĎåžŤåŕźåŕĘæIěåŕźPointåžŢäÿŃçŽĎåŃĚèčĚçŽĎæŻť æŤź
ä¿ŃåęĆïijŇåęĆæđIJä¡ăåEşåőŽä¡£çŤĺåŔęåď ŰäÿĂäÿłèČűåŻŁäžĘïijŃéČčäźĹåŘłéIJĂèęĄæŻť æŤźè£Źäÿd'äÿłå

åŕźäžŐèČűåŻŁåŕźèśąäÿĂäÿłéŽ¿çĆźåIJĺäžŐåđČåIJ¿åŻđæŤűåŠŇåĘĚå■Ÿçőąç ŘĘãĂĆ
`PyPoint_FromPoint()` åĞ¡æŢřæŐěåŘŮäÿĂäÿł `must_free`
åŘĆæŢřïijŇ çŤĺæIěæŇĞåőŽå¡ŞèČűåŻŁèćńéŤĂæŕĄæŮűåžŢåśĆPoint *
çżŞæđĎä¡ŞæŸŕåŘęåžŤèŕěèćńåŻđæŤűãĂĆ åIJĺæ§ŘäžŻCäžčçăĄäÿ■ïijŇå¡ŠåśđéŭőécŸéĂŽåÿÿå¿ĹéŽ¿èćńåď Ì
çĺŃåžŘåŠŸåŔŕäžěä¡£çŤĺ `extra` åŘĆæŢřæIěæŐğåĹűïijŇèĂŇäÿ■æŸŕå■ŢæŰźéIćçŽĎåĘşåőŽåđČåIJ¿åŻđæŤ
èęĄæşĺæĎŘçŽĎæŸŕåŠŇçŐřæIJL'èČűåŻŁæIJL'åĚşçŽĎæđŘæđĎåŹĺèČ¡ä¡£çŤĺ
`PyCapsule_SetDestructor()` åĞ¡æŢřæIěæŽť æŤźãĂĆ

åŕźäžŐæűL'åŘŁåĹřçżŞæđĎä¡ŞçŽĎCäžčçăĄèĂŇèĺĂïijŇä¡£çŤĺèČűåŻŁæŸŕäÿĂäÿłærŤè¿ČåŘĹçŘĘçŽĎè
ä¿ŃåęĆïijŇæIJL'æŮűåĂŹä¡ăåźűäÿ■åĚşå£ČæŽť éIJšçżŞæđĎä¡ŞçŽĎåĘĚéČĺä£qæĄŕæĹŰèĂĚářĘåĚűè¡ñæ■ćæ
éĂŽè£Ğä¡£çŤĺèČűåŻŁïijŇä¡ăåŘŕäžěåIJĺåőČäÿŁéIćæŤ¿äÿĂäÿłè¡žéĞŘçžğçŽĎåŇĚèčĚåŹĺïijŇçĎűåŘŐářĘåőČ

15.5 äžŐæĽľåijäæĺąåİŮäÿ■åőŽäźĽåŠŇåŕijåĞžCçŽĎAPI

éŮőécŸ

ä¡äæIJL'äÿĂäÿłCæL'ľ'åśŢæĺąåİŮïijŇåIJĺåĘĚéČĺåőŽäźL'äžĘå¿Ĺåď'ŽæIJL'çŤĺçŽĎåĞ¡æŢřïijŇä¡äæČşåřĘå
APIä¿ŻåĚűäżŮåIJřæŰźä¡£çŤĺãĂĆ ä¡äæČşåIJĺåĚűäżŮæL'ľ'åśŢæĺąåİŮäÿ■ä¡£çŤĺè£ŹäžŻåĞ¡æŢřïijŇä¡ĘæŸŕäÿ
åźűäÿŤéĂŽè£ĞCçijŰèŕŚåŹĺ/éŞ¿æŐěåŹĺæİěåĄŽçIJŃäÿŁåŐżçL'źåĹńåď'■æİĆïijĹæĹŰèĂĚäÿ■åŔŕèČ¡åĄŽåĹř

èğčåĘşæŰźæąĹ

æIJňèŁĆäÿżèęĄéŮőécŸæŸŕåęĆä¡Ţåď'ĎçŘĘ15.4åřŔèŁĆäÿ■æŔŘåĹřçŽĎPointåŕźèśąãĂĆäżŤçżĘåŻđäÿ

```
/* Destructor function for points */
static void del_Point(PyObject *obj) {

  free(PyCapsule_GetPointer(obj,"Point"));
}

/* Utility functions */
static Point *PyPoint_AsPoint(PyObject *obj) {
  return (Point *) PyCapsule_GetPointer(obj, "Point");
}

static PyObject *PyPoint_FromPoint(Point *p, int must_free) {
  return PyCapsule_New(p, "Point", must_free ? del_Point : NULL);
}
```

çŐřåIJĺçŽĎéŮőécŸæŸŕæĂŐæăůåřĘ `PyPoint_AsPoint()` åŠŇ `Point_FromPoint()` åĞ¡æŢřä¡IJäÿžAPIåŕijåĞžïijŇ è£ŹæăůåĚűäżŮæL'ľ'åśŢæĺąåİŮèČ¡ä¡£çŤĺázűéŞ¿æŐěåőČäžñïijŇæŕŤåęĆåęĆædIJä¡äæIJL'åĚűäżŮæL'ľ'åśŢäź§

èęĄèğčåĘşè£ŹäÿłéŮőécŸïijŇéęŰåĚĹèęĄäÿž `sample` æL'ľ'åśŢåĘŹäÿłæŰřçŽĎåď'ť'æŰĞäžűåŘ■åŔń `pysample.h` ïijŇåęĆäÿŃïijŽ

```
/* pysample.h */
#include "Python.h"
#include "sample.h"
#ifdef __cplusplus
extern "C" {
#endif

/* Public API Table */
typedef struct {
  Point *(*aspoint)(PyObject *);
  PyObject *(*frompoint)(Point *, int);
} _PointAPIMethods;

#ifndef PYSAMPLE_MODULE
```

```
/* Method table in external module */
static _PointAPIMethods *_point_api = 0;

/* Import the API table from sample */
static int import_sample(void) {
  _point_api = (_PointAPIMethods *) PyCapsule_Import("sample._point_
↪api",0);
  return (_point_api != NULL) ? 1 : 0;
}

/* Macros to implement the programming interface */
#define PyPoint_AsPoint(obj) (_point_api->aspoint)(obj)
#define PyPoint_FromPoint(obj) (_point_api->frompoint)(obj)
#endif

#ifdef __cplusplus
}
#endif
```

è£ŹéĞŇæIJĂéĞ■èęĄçŽĎéČĺåĹĘæŸŕåĞ¡æŢřæŇĞéŠĹèąĺ `_PointAPIMethods` . åőČäijŽåIJĺåŕijåĞžæĺąåİŮæŮűèćńåĹĺåğŃåŇŰïijŇçĎűåŘŐåŕijåĚěæĺąåİŮæŮűèćńæ§ěæĽ¿åĹřãĂĆä£őæŤźåŐ§åğŃçŽĎæĽľåśŢæĺąåİŮæİěåąńåĚĚèąĺæäijåźűåřĘåőČåČŔäÿŃéİćè£ŹæăůåŕijåĞžïijŽ

```
/* pysample.c */

#include "Python.h"
#define PYSAMPLE_MODULE
#include "pysample.h"

...
/* Destructor function for points */
static void del_Point(PyObject *obj) {
  printf("Deleting point\n");
  free(PyCapsule_GetPointer(obj,"Point"));
}

/* Utility functions */
static Point *PyPoint_AsPoint(PyObject *obj) {
  return (Point *) PyCapsule_GetPointer(obj, "Point");
}

static PyObject *PyPoint_FromPoint(Point *p, int free) {
  return PyCapsule_New(p, "Point", free ? del_Point : NULL);
}

static _PointAPIMethods _point_api = {
  PyPoint_AsPoint,
  PyPoint_FromPoint
};
...
```

```
/* Module initialization function */
PyMODINIT_FUNC
PyInit_sample(void) {
  PyObject *m;
  PyObject *py_point_api;

  m = PyModule_Create(&samplemodule);
  if (m == NULL)
    return NULL;

  /* Add the Point C API functions */
  py_point_api = PyCapsule_New((void *) &_point_api, "sample._point_
↪api", NULL);
  if (py_point_api) {
    PyModule_AddObject(m, "_point_api", py_point_api);
  }
  return m;
}
```

æIJĂåŘŐïijŇäÿŃéÍćæŸŕäÿĂäÿłæŰřçŽĎæĽľåśŢæĺąåİŮä¿Ńå■ŘïijŇçŤĺæİěåŁăè¡¡åźűä¡£çŤĺè£ŹäžŻAPIå

```
/* ptexample.c */

/* Include the header associated with the other module */
#include "pysample.h"

/* An extension function that uses the exported API */
static PyObject *print_point(PyObject *self, PyObject *args) {
  PyObject *obj;
  Point *p;
  if (!PyArg_ParseTuple(args,"O", &obj)) {
    return NULL;
  }

  /* Note: This is defined in a different module */
  p = PyPoint_AsPoint(obj);
  if (!p) {
    return NULL;
  }
  printf("%f %f\n", p->x, p->y);
  return Py_BuildValue("");
}

static PyMethodDef PtExampleMethods[] = {
  {"print_point", print_point, METH_VARARGS, "output a point"},
  { NULL, NULL, 0, NULL}
};

static struct PyModuleDef ptexamplemodule = {
```

```
  PyModuleDef_HEAD_INIT,
  "ptexample",           /* name of module */
  "A module that imports an API",  /* Doc string (may be NULL) */
  -1,                 /* Size of per-interpreter state or -1 */
  PtExampleMethods       /* Method table */
};

/* Module initialization function */
PyMODINIT_FUNC
PyInit_ptexample(void) {
  PyObject *m;

  m = PyModule_Create(&ptexamplemodule);
  if (m == NULL)
    return NULL;

  /* Import sample, loading its API functions */
  if (!import_sample()) {
    return NULL;
  }

  return m;
}
```

çijŰèŕŚè£ŹäÿłæŰřæĺąåİŮæŮűïijŇä¡ăçŤŽèĞşäÿ■éIJĂèęĄåŐżèĂČèŹŚæĂŐæăůåřĘåĞ¡æŢřåžŞæĹŰäżčçà
ä¿ŇåęĆïijŇä¡ääŔŕäżěåČŔäÿŇéİćè£ŹæăůåĹŻåžžäÿĂäÿłçőĂå■ŢçŽĎ `setup.py` æŰĞäžűïijŽ

```
# setup.py
from distutils.core import setup, Extension

setup(name='ptexample',
      ext_modules=[
        Extension('ptexample',
                  ['ptexample.c'],
                  include_dirs = [],  # May need pysample.h␣
↪directory
                  )
        ]
)
```

åęĆædIJäÿĂåĹĞæ■čåÿÿïijŇä¡ääijŽåŔŚçŐřä¡ăçŽĎæŰřæĽľåśŢåĞ¡æŢřèČ¡åŠŇåőŽäźĽåIJĺåĚűäžŰæĺąåİ
APIåĞ¡æŢřäÿĂèţůè£ŘèąŇçŽĎå¿Ĺåě¡ãĂĆ

```
>>> import sample
>>> p1 = sample.Point(2,3)
>>> p1
<capsule object "Point *" at 0x1004ea330>
>>> import ptexample
>>> ptexample.print_point(p1)
2.000000 3.000000
```

```
>>>
```

èőÍèőž

æIJňèŁĆå§žäžŐäÿĂäÿłåĽ■æŔŘåřśæŸŕřijŇèČűåŻŁårźèśąèČ¡èŐůåŔŰäżża¡Ţä¡ăæČşèęĄçŽĎårźèśąçŽĎ
è£Źæăůç ŽĎèŕÏijŇåőŽäźĽæÍąåİŮäijŽåąńåĚĚäÿĂäÿłåĞ¡æŢřæŇĞéŠĹçŽĎçżŞæđĎä¡ŞijŇåĹŻåżžäÿĂäÿłæŇČ
ä¿ŃåęĆ `sample._point_api`.

åĚűäžŰæÍąåİŮèČ¡åď§åIJÍårijåĚěæŮűèŐůåŔŰåĹřè£ŹäÿłåśđæĂğåźűæŔŘåŔŰåžŢåśĆçŽĎæŇĞéŠĹãĂĆ
äžŃåőđäÿŁïijŇPythonæŔŘä¿ŻäžĘ `PyCapsule_Import()`
åůěåĚůåĞ¡æŢřijŇäÿžäžĘåőŇæĹŘæĽĂæIJĽçŽĎæ■ěélď ãĂĆ
ä¡ääŔłéIJĂæŔŘä¿Żåśđæ ĂğçŽĎåŘ■å■Ůå■şåŔŕijĹæŕŤåęĆsample._point_apiïijĽïijŇçĎűåŘŐäžŰårśäijŽäÿĂ

åIJÍårĘèćńårijåĞžåĞ¡æŢřåŔŸäÿžåĚűäžŰæÍąåİŮäÿ■æŹőéĂŽåĞ¡æŢřæŮűïijŇæIJL'äÿĂäžŻCçijŰçÍŃéŹů
åIJÍ `pysample.h` æŰĞäžűäÿ■ïijŇäÿĂäÿł `_point_api`
æŇĞéŠĹèćńçŤÍæİěæŇĞåŘŚåIJÍårijåĞžæÍąåİŮäÿ■èćńåĹİåğŃåŇŰçŽĎæŰźæşŢèąÍãĂĆ
äÿĂäÿłçŻÿåĚşçŽĎåĞ¡æŢř `import_sample()` èćńçŤÍæİěæŇĞåŘŚèČűåŻŁårijåĚěåźűåĹİåğŃåŇŰè£Źäÿłæ
è£ŹäÿłåĞ¡æŢřå£Ěéążå IJÍäżża¡ŢåĞ¡æŢřèćńä¡£çŤÍäźŃåĽ■èćńèřÇçŤÍãĂĆéĂŽåÿÿæİěèőšïijŇåőČäijŽåIJÍæÍąåİ
æIJĂåŘŐïijŇCçŽĎéćĎåď ĎçŘĘåőŔèćńåőŽäźĽïijŇèćńçŤÍæİěéĂŽè£ĞæŰźæşŢèąÍåŐżåĹĘåŔŚè£ŹäžŻAPIåĞ
çŤÍæĹůåŔłéIJĂèęĄä¡£çŤÍè£ŹäžŽåŐ§ågŃåĞ¡æŢřåŘ■çğřå■şåŔŕijŇäÿ■éIJĂèęĄéĂŽè£ĞåőŘåŐżäžĘèğčåĚűä

æIJĂåŘŐïijŇè£ŸæIJĽäÿĂäÿłéĞ■èęĄçŽĎåŐ§åŻäèőľä¡ăåŐżä¡£çŤÍè£ŹäÿłæŁĂæIJŕæİěéŞ¿æŐěæÍąåİŮâ
åęĆæđIJä¡ăäÿ■æČşä¡£çŤÍæIJñæIJžçŽĎæŁĂæIJŕïijŇéĆčä¡ăåřśå£Ěéążä¡£çŤÍåĚśäžńåžŞçŽĎéńŸçžğçL'źæĂğå
ä¿ŃåęĆïijŇåřĘäÿĂäÿłæŹőéĂŽçŽĎAPIåĞ¡æŢřæŤ¿åĚěäÿĂäÿłåĚśäžńåžŞåźűçąőä£ÍæĽĂæIJĽæĽľåśŢæÍąåİŮ
è£ŹçğæŰźæşŢçąőåőđåŔŕèąŇïijŇä¡ĘæŸŕåőĆçŽÿåŕźćźĄçŘŘïijŇçĽźåĹńæŸŕåIJÍåď ğåđŃçşżçž§äÿ■ãĂĆ
æIJňèŁĆæijŤçď žäžĘåęĆä¡ŢéĂŽè£ĞPythonçŽĎæŹőéĂŽåŕijåĚěæIJžåĹűåŠŇäżĚäżĚåĞääÿłèČűåŻŁèřĆçŤÍæİ
åŕźäžŐæÍąåİŮçŽĎçijŰèŕŚïijŇä¡ăåŘłéIJĂèęĄåőŽäźĽåď ťæŰĞäżűïijŇèĂŇäÿ■éIJĂèęĄèĂČèŹŚåĞ¡æŢřåžŞçŽ

æŻťåď ŽåĚşäžŐåĹľçŤÍC APIæİěæđĎéĂäæĽľåśŢæÍąåİŮçŽĎä£qæĄŕåŔŕäżěåŔĆèĂČ
PythonçŽĎæŰĞæąč

15.6 äžŐCèŕ■èÍĂäÿ■èřČçŤÍPythonäžčçăĄ

éŮőécŸ

ä¡äæČşåIJÍCäÿ■åőĽåĚÍçŽĎæĽğèąŇæ§ŘäÿłPythonèřČçŤÍåźűè£ŤåŻđçżŞæđIJçżŹCãĂĆ
ä¿ŃåęĆïijŇä¡äæČşåIJÍCèŕ■èÍĂäÿ■ä¡£çŤÍæ§ŘäÿłPythonåĞ¡æŢřä¡IJäÿžäÿĂäÿłåŻđèřČãĂĆ

èğčåĘşæŰźæąĹ

åIJÍCèŕ■èÍĂäÿ■èřČçŤÍPythonéİđåÿÿçőĂå■ŢïijŇäÿ■è£Ğèő¿èőąåĹřäÿĂäžŻårŔçł■éŮÍãĂĆ
äÿŃéİćçŽĎCäžčçăĄåŚŁèŕĽä¡äæĂŐæăůåőĽåĚÍçŽĎèřČçŤÍïijŽ

```
#include <Python.h>

/* Execute func(x,y) in the Python interpreter.  The
   arguments and return result of the function must
   be Python floats */

double call_func(PyObject *func, double x, double y) {
  PyObject *args;
  PyObject *kwargs;
  PyObject *result = 0;
  double retval;

  /* Make sure we own the GIL */
  PyGILState_STATE state = PyGILState_Ensure();

  /* Verify that func is a proper callable */
  if (!PyCallable_Check(func)) {
    fprintf(stderr,"call_func: expected a callable\n");
    goto fail;
  }
  /* Build arguments */
  args = Py_BuildValue("(dd)", x, y);
  kwargs = NULL;

  /* Call the function */
  result = PyObject_Call(func, args, kwargs);
  Py_DECREF(args);
  Py_XDECREF(kwargs);

  /* Check for Python exceptions (if any) */
  if (PyErr_Occurred()) {
    PyErr_Print();
    goto fail;
  }

  /* Verify the result is a float object */
  if (!PyFloat_Check(result)) {
    fprintf(stderr,"call_func: callable didn't return a float\n");
    goto fail;
  }

  /* Create the return value */
  retval = PyFloat_AsDouble(result);
  Py_DECREF(result);

  /* Restore previous GIL state and return */
  PyGILState_Release(state);
  return retval;

fail:
```

```
  Py_XDECREF(result);
  PyGILState_Release(state);
  abort();    // Change to something more appropriate
}
```

èęĄä¡£çŤĺè£ŹäÿłåĞ¡æŢřïijŇä¡ăéIJĂèęĄèŐůåŔŰäijăéĂŠè£ĞæİěçŽĎæ§Řäÿłåůšå■ŸåIJÍPythonèřČçŤĺçŽ
æIJĽå¿Ĺåď Žçğ■æŰźæşŢåŔŕäžěèőľä¡ăè£ŹæăůåĄŽïijŇ æŕŤåęĆåřĘäÿĂäÿłåŔŕèřČçŤĺåŕźèśąäijăçżŹäÿĂäÿłæĽ

äÿŃéİćæŸŕäÿĂäÿłçőĂå■Ţä¿Ńå■ŘçŤĺæİěæŐl'éěřäžŐäÿĂäÿłåţŇåĚěçŽĎPythonèğčéĞŁåŹĺäÿ■èřČçŤĺäÿ

```
#include <Python.h>

/* Definition of call_func() same as above */
...

/* Load a symbol from a module */
PyObject *import_name(const char *modname, const char *symbol) {
  PyObject *u_name, *module;
  u_name = PyUnicode_FromString(modname);
  module = PyImport_Import(u_name);
  Py_DECREF(u_name);
  return PyObject_GetAttrString(module, symbol);
}

/* Simple embedding example */
int main() {
  PyObject *pow_func;
  double x;

  Py_Initialize();
  /* Get a reference to the math.pow function */
  pow_func = import_name("math","pow");

  /* Call it using our call_func() code */
  for (x = 0.0; x < 10.0; x += 0.1) {
    printf("%0.2f %0.2f\n", x, call_func(pow_func,x,2.0));
  }
  /* Done */
  Py_DECREF(pow_func);
  Py_Finalize();
  return 0;
}
```

èęĄæđĎåżžä¿Ńå■ŘäżčçăĄïijŇä¡ăéIJĂèęĄçijŰèŕŚCåźűåřĘåőČéŞ¿æŐěåĹřPythonèğčéĞŁåŹĺãĂĆ
äÿŃéİćçŽĎMakefileåŔŕäžěæŢŹä¡ăæĂŐæăůåĄŽïijĹäÿ■è£ĞåIJĺä¡ăæIJžåŹĺäÿŁéİćéIJĂèęĄäÿĂäžŻéĚ■ç¡őïijĽ

```
all::
        cc -g embed.c -I/usr/local/include/python3.3m \
          -L/usr/local/lib/python3.3/config-3.3m -lpython3.3m
```

çijŰèŕŚåźűè£ŘèąŇäijŽäžğçŤ§çśżäijijäÿŃéİćçŽĎè¿ŞåĞžïijŽ

```
0.00 0.00
0.10 0.01
0.20 0.04
0.30 0.09
0.40 0.16
...
```

äÿŃéİćæŸŕäÿĂäÿłçĺ■å¿őäÿ■åŘŇçŽĎä¿Ńå■ŘïijŇåśŢçď žäžĘäÿĂäÿłæĽ'ĺ åśŢåĞ¡æŢřïijŇ åőČæŐěåŘŮäÿĂäÿłåŔŕèřČçŤĺåŕźèśąåŠŇåĚűäžŰåŔĆæŢřïijŇåźűårĘåőČäžňäijăéĂŠçžŹ `call_func()` æİěåĄŽæţŃèŕŢïijŽ

```
/* Extension function for testing the C-Python callback */
PyObject *py_call_func(PyObject *self, PyObject *args) {
  PyObject *func;

  double x, y, result;
  if (!PyArg_ParseTuple(args,"Odd", &func,&x,&y)) {
    return NULL;
  }
  result = call_func(func, x, y);
  return Py_BuildValue("d", result);
}
```

ä¡£çŤĺè£ŹäÿłæĽ'ĺ åśŢåĞ¡æŢřïijŇä¡ăèęĄåČŔäÿŃéİćè£ŹæăůæţŃèŕŢåőČïijŽ

```
>>> import sample
>>> def add(x,y):
...     return x+y
...
>>> sample.call_func(add,3,4)
7.0
>>>
```

èőĺèőž

åęĆæđIJä¡ăåIJíCèŕ■èĺĂäÿ■èřČçŤĺPythonïijŇèęĄèőřä¡ŔæIJĂéĞ■èęĄçŽĎæŸŕCèŕ■èĺĂäijŽæŸŕäÿżä¡ŞãĂ äźş§åřśæŸŕèŕťïijŇCèŕ■èĺĂèť§èť čæđĎéĂăåŔĆæŢřãĂĄèřČçŤĺPythonåĞ¡æŢřãĂĄæčĂæ§ěåijĆåÿÿãĂĄæčĂæ

ä¡IJäÿžçňňäÿĂæ■ěïijŇä¡ăå£ĚéąžåĚĹæIJL'äÿĂäÿłèąĺçď žä¡ăårĘèęĄèřČçŤĺçŽĎPythonåŔŕèřČçŤĺåŕźèśąãĂ è£ŹåŔŕäžěæŸŕäÿĂäÿłåĞ¡æŢřãĂĄçśžãĂĄæŰźæşŢãĂĄåĘĚç¡őæŰźæşŢæĹŰåĚűäžŰäżżæĎŔåőđçŐřäžĘ `__call__()` æŞ■ä¡IJçŽĎäÿIJèě£ãĂĆ äÿžäžĘçąőä£ĺæŸŕåŔŕèřČçŤĺçŽĎïijŇåŔŕäžěåČŔäÿŃéİćçŽĎäžčçăĄè `PyCallable_Check()` åĄŽæčĂæ§ěijŽ

```
double call_func(PyObject *func, double x, double y) {
  ...
  /* Verify that func is a proper callable */
  if (!PyCallable_Check(func)) {
    fprintf(stderr,"call_func: expected a callable\n");
    goto fail;
```

```
  }
  ...
```

åIJÍCäžčçăĄéĞŇåď ĎçŘĘéŤŹèŕŕä¡ăéIJĂèęĄæăijåď ŰçŽĎåřŔå£ČãĂĆäÿĂèĹňæİěèőšïijŇä¡ăäÿ■èČ¡äžĚä
éŤŹèŕŕåžŤèŕěä¡£çŤÍCäžčçăĄæŰźåijŔæİěèćńåď ĎçŘĘãĂĆåIJÍè£ŹéĞŇïijŇæĹŚäžňæĽŞçőŮåřĘåŕźéŤŹèŕŕçŽĎ
abort() çŽĎéŤŹèŕŕåď ĎçŘĘåŹÍãĂĆ åőČäijŽçžŞæİ§æŐĽæŢť äÿłçÍŃåžŔïijŇåIJÍçIJ§åőđçŐŕåćČäÿŃéİćä¡ăă
ä¡ăèęĄèőŕä¡ŔçŽĎæŸŕåIJÍè£ŹéĞŇCæŸŕäÿżèğŠïijŇåŽăæ■ď åźűæšąæIJL'èů§æŁŽåĞžåijĆåÿÿçŽÿåŕźåžŤçŽĎæ
éŤŹèŕŕåď ĎçŘĘæŸŕä¡ăåIJÍçijŰçÍŃæŮűå£Ëéążèę Ą èĂČèŹŚçŽĎäžŃæČĚãĂĆ

èřČçŤÍäÿĂäÿłåĞ¡æŢřçŽÿåŕźæİěèőšå¿ĹçőĂå■ŢâĂŤâĂŤåŔłéIJĂèęĄä¡£çŤÍ
PyObject_Call() ïijŇ äijăäÿĂäÿłåŔŕèřČçŤÍåŕźèśąçżŹåőČãĂĄäÿĂäÿłåŔĆæŢřåĚČçžĎåŠŇäÿĂäÿłåŔŕéĂ
èęĄæđĎåžžåŔĆæŢřåĚČçžĎæĹŰå■ŮåËÿïijŇä¡ăåŔŕäžëä¡£çŤÍ Py_BuildValue()
,åęĆäÿŃïijŽ

```
double call_func(PyObject *func, double x, double y) {
  PyObject *args;
  PyObject *kwargs;

  ...
  /* Build arguments */
  args = Py_BuildValue("(dd)", x, y);
  kwargs = NULL;

  /* Call the function */
  result = PyObject_Call(func, args, kwargs);
  Py_DECREF(args);
  Py_XDECREF(kwargs);
  ...
```

åęĆæđIJæšąæIJL'åĚşéŤőå■ŮåŔĆæŢřijŇä¡ăåŔŕäžëäijăéĂŠNULLãĂĆå¡Şä¡ăèęĄèřČçŤÍåĞ¡æŢřæŮűïijŇ
éIJĂèęĄçąőä£Ìä¡£çŤÍäžĘ Py_DECREF() æĹŰèĂĚ Py_XDECREF() æÿĚçŘĘåŔĆæŢřãĂĆ
çňňäžŇäÿłåĞ¡æŢřçŽÿåŕźåőL'åĚÍçĆźïijŇåŽääÿžåőČåĚĄèőÿäijăéĂŠNULLæŇĞéŠĹïijÍçŻt'æŐěå£¡çŢěåőČïijÍ
è£Źäź§æŸŕäÿžäžĂäźĹæĹŚäžňä¡£çŤÍåőČæÌěæÿĚçŘĘåŔŕéĂL'çŽĎåĚşéŤőå■ŮåŔĆæŢřãĂĆ

èřČçŤÍäÿĞPythonåĞ¡æŢřäźŃåŘŐïijŇä¡ăå£ËéążæčĂæ§ěæŸŕåŘęæIJL'åijĆåÿÿåŔŚçŤ§ãĂĆ
PyErr_Occurred() åĞ¡æŢřåŔŕèćńçŤÍæÌěåĄŽè£Źäžűäž ŃãĂĆ
ăŕźåŕźäžŐåijĆåÿÿçŽĎåď ĎçŘĘåřśæIJL'çĆźžżçČęäžĘïijŇçŤśäžŐæŸŕçŤÍCèŕ■èÍĂåĘźçŽĎïijŇä¡ăæšąæIJL'åČ
åŽăæ■ď ïijŇä¡ăå£Ëéążèę Ąèő¿ç¡őäÿĂäÿłåijĆåÿÿçŁűæĂĄçĂąïijŇæL'Şå■ŕåijĆåÿÿä£ąæĄŕæĹŰåĚűäžŰçŻÿåžŤ
åIJÍè£ŹéĞŇïijŇæĹŚäžňéĂL'æŃľäžĘçőĂå■ŢçŽĎ abort()
æÌěåď ĎçŘĘãĂĆåŘęåď ŰïijŇäijăçż§CçÍŃåžŔåŚŸåŔŕèČ¡äijŽçŻt'æŐěěľ'çÍŃåžŔåěŤæžČãĂĆ

```
...
/* Check for Python exceptions (if any) */
if (PyErr_Occurred()) {
  PyErr_Print();
  goto fail;
}
...
fail:
  PyGILState_Release(state);
```

```
  abort();
```

äžŐèřČçŤÍPythonåĞ¡æŢřçŽĎè£ŤåŽđåĂijäÿ■æŔŘåŔŰä£ąæĄŕéĂŽåÿÿèęĄè£ŻèąŇçśžåđŃæčĂæ§ěåŠŇæ
èęĄè£ŹæăůåĄŽçŽĎèŕÏïijŇä¡ăå£Ěéąžä¡£çŤÍPythonåŕźèśąåśĆäÿ■çŽĎåĞ¡æŢřãĂĆ
åIJÍè£ŹéĞŇæĹŚäžňä¡£çŤÍäžĘ PyFloat_Check() åŠŇ PyFloat_AsDouble()
æÌěæčĂæ§ěåŠŇæŔŘåŔŰPythonæţőçĆźæŢřãĂĆ

æIJĂåŘŐäÿĂäÿłéŮőécŸæŸŕåŕźäžŐPythonåĚĺåśĂéŤĄçŽĎçőąçŘęãĂĆ
åIJÍCèŕ■èÍĂäÿ■èő£éŮőPythonçŽĎæŮűåĂŹïijŇä¡ăéIJĂèęĄçąőä£ÌGILèćńæ■čçąőçŽĎèŐůåŔŰåŠŇéĞŁæŤ¿äž
äÿ■çĎűçŽĎèŕÏïijŇåŔŕěČ¡äijŽåŕijèĞť`èğčéĞŁåŹÍè£ŤåŽđéŤŹèŕŕæŢřæ■őæĹŰèĂĚçŻť`æŐěåěŤæžČãĂĆ
èřČçŤÍ PyGILState_Ensure() åŠŇ PyGILState_Release()
åŔŕäžěçąőä£ÌäÿĂåĹĞéČ¡èČ¡æ■čåÿÿãĂĆ

```
double call_func(PyObject *func, double x, double y) {
  ...
  double retval;

  /* Make sure we own the GIL */
  PyGILState_STATE state = PyGILState_Ensure();
  ...
  /* Code that uses Python C API functions */
  ...
  /* Restore previous GIL state and return */
  PyGILState_Release(state);
  return retval;

fail:
  PyGILState_Release(state);
  abort();
}
```

äÿĂæŮęè£ŤåŽđïijŇPyGILState_Ensure() åŔŕäžěçąőä£ÌèřČçŤÍçž£çÍŃçŃňå■ăPythonèğčéĞŁåŹÍãĂ
åřśçőŮCäžčçăĄè£ŘèąŇäžŐåŔęåď`ŰäÿĂäÿłèğčéĞŁåŹÍäÿ■ç§ěéĄŞçŽĎçž£çÍŃäź§æšąäžŃãĂĆ
è£ŹæŮűåĂŹïijŇCäžčçăĄåŔŕäžěèĞłçŤśçŽĎä¡£çŤÍäżżä¡ŢåőČæČşèęĄçŽĎPython C-API
åĞ¡æŢřãĂĆ èřČçŤÍæĹŘåŁ§åŘŐïijŇPyGILState_Release()èćńçŤÍæÌèèőšèğčéĞŁåŹÍæĄćåď`■åĹřåŐ§ågŃçŁűa

èęĄæşÍæĎŔçŽĎæŸŕæŕŔäÿĂäÿł PyGILState_Ensure()
èřČçŤÍå£ĚéąžèůçÌĂäÿĂäÿłåŇźéĚ■çŽĎ PyGILState_Release()
èřČçŤÍâĂŤâĂŤå■şä¿£æIJL'éŤŹèŕŕåŔŚçŤ§ãĂĆ åIJÍè£ŹéĞŇïijŇæĹŚäžňä¡£çŤÍäÿĂäÿł goto
èŕ■åŘęçIJŃäÿŁåŐżæŸŕäÿłåŔŕæĂŢçŽĎèő¿èőąïijŇ ä¡ĘæŸŕåődéŹĚäÿŁæĹŚäžňä¡£çŤÍåőČæÌèèőšæŐğåĹűæÌČè
åIJÍ fail: æăĞç■¿åŘŐéÍćçŽĎäžčçăĄåŠŇPythonçŽĎ fianl:
åİŮçŽĎçŤÍéĂŤæŸŕäÿĂæăůçŽĎãĂĆ

åęĆædIJä¡ăä¡£ŢÍæL'ĂæIJL'è£ŹäžŻçžęåőŽæÌěçijŰåĘŹCäžčçăĄïijŇåŇĚæŃňåŕźGILçŽĎçőąçŘĘãĂĄåij
ä¡ăäijŽåŔŚçŐřäžŐCèŕ■èÍĂäÿ■èřČçŤÍPythonèğčéĞŁåŹÍæŸŕåŔŕéÍăçŽĎâĂŤâĂŤåřśçőŮåĘ■åď`■æÌĆçŽĎçÍŃåž

15.7 äžŐCæĽľåśŢäÿ■éĞŁæŤ¿åĚĺåśĂéŤĄ

éŮőécŸ

ä¡ăæČşèőľCæĽľåśŢäžčçăĄåŠŇPythonèğčéĞŁåŹĺäÿ■çŽĎåĚűäžŰè£ŻçÍŃäÿĂètůæ■čçąőçŽĎæĽğèąŇïij
éĆčäźĹä¡ăåřśéIJĂèęĄåŐżéĞŁæŤ¿åźűéĞ■æŰřèŐůåŔŰåĚĺåśĂèğčéĞŁåŹĺéŤĄïijĹGILïijĽãĂĆ

èğčåĘşæŰźæąĹ

åIJÍCæĽľåśŢäžčçăĄäÿ■ïijŇGILåŔŕäžěéĂŽè£ĞåIJĺäžčçăĄäÿ■æŔŠåĚěäÿŃéÍćè£ŹæăůçŽĎåőŔæÍěéĞŁæ

```
#include "Python.h"
...

PyObject *pyfunc(PyObject *self, PyObject *args) {
   ...
   Py_BEGIN_ALLOW_THREADS
   // Threaded C code.  Must not use Python API functions
   ...
   Py_END_ALLOW_THREADS
   ...
   return result;
}
```

èőÍèőž

åŔłæIJL'å¡Şä¡ăçąőä£ÍæšąæIJL'Python C APIåĞ¡æŢřåIJÍCäÿ■æĽğèąŇçŽĎæŮűåĂŹä¡ăæĽ■èČ¡åőL'åĚĺçŽ
GILéIJĂèęĄèćńéĞŁæŤ¿çŽĎåÿÿèğĄçŽĎåIJžæŹŕæŸŕåIJÍèőąçőŮårĘéŻĘåđŇäžčçăĄäÿ■éIJĂèęĄåIJÍCæŢřçżĎ
æĹŰèĂĚæŸŕèęĄæĽğèąŇéŸżåąđçŽĎI/OæŞ■ä¡IJæŮűïijĹæŕŤåęĆåIJÍäÿĂäÿłæŰĞäžűæŔŔè£řçňęäÿŁèŕżåŔŰa

å¡ŞGILèćńéĞŁæŤ¿åŘŐïijŇåĚűäžŰPythonçž£çÍŃæĽ■èćńåĚĄęőÿåIJÍèğčéĞŁåŹĺäÿ■æĽğèąŇãĂĆ
`Py_END_ALLOW_THREADS` åőŔäijŽéŸżåąđæĽğèąŇçŻťåĹřěřĆçŤÍçž£çÍŃéĞ■æŰřèŐůåŔŰäžĘGILãĂĆ

15.8 CåŠŇPythonäÿ■çŽĎçž£çÍŃæůûçŤÍ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłçÍŃåžŔéIJĂèęĄæůůåŘĹä¡£çŤÍCãĂĄPythonåŠŇçž£çÍŃïijŇ
æIJL'äžŻçž£çÍŃæŸŕåIJÍCäÿ■åĹŻåžžçŽĎïijŇèűĚåĞžäžĘPythonèğčéĞŁåŹÍçŽĎæŐğåĹűèŇČåŻť'ãĂĆ
åźűäÿŤäÿĂäžŻçž£çÍŃè£Ÿä¡£çŤÍäžĘPython C APIäÿ■çŽĎåĞ¡æŢřãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşåřĘCãĂĄPythonåŠŇçž£çĺŃæůůåŘĹåIJĺäÿĂètůïijŇä¡ăéIJĂèęĄçąőä£Ìæ■čçąőçŽĎåĹİåğŇ
èęĄæČşè£Źæăůå ĄŽïijŇåŕŕäžéåřĘäÿŇåĽŮäžčçăĄæŤ¿åĽřä¡ăçŽĎCäžčçăĄäÿ■åźűçąőä£ÌåőČåIJĺäżżä¡Ţçž£çĺŇ

```
#include <Python.h>
  ...
  if (!PyEval_ThreadsInitialized()) {
    PyEval_InitThreads();
  }
  ...
```

åŕźäžŐäżżä¡ŢèřČçŤĺPythonåŕźèśąæĹŰPython C APIçŽĎCäžčçăĄïijŇçąőä£Ìä¡ăéęŰåĚĹåůšçżŔæ■čçąőå
è£ŹåŔŕäžěçŤĺ `PyGILState_Ensure()` åŠŇ `PyGILState_Release()`
æÌěåĄŽåĹřïijŇåęĆäÿŇæĽĂçd'žïijŽ

```
...
/* Make sure we own the GIL */
PyGILState_STATE state = PyGILState_Ensure();

/* Use functions in the interpreter */
...
/* Restore previous GIL state and return */
PyGILState_Release(state);
...
```

æŕŔæňąèřČçŤĺ `PyGILState_Ensure()` éČ¡èęĄçŻÿåžŢçŽĎèřČçŤĺ
`PyGILState_Release()`.

èőĺèőž

åIJĺæűĽ'åŔŁåĹřCåŠŇPythonçŽĎéńŸçžğçĺŃåžŔäÿ■ïijŇå¿Ĺåd'ŽäžŃæČĚäÿĂètůåĄŽæŸŕå¿Ĺåÿÿèğ
åŔŕèČ¡æŸŕåŕźCãĂĄPythonãĂĄCçž£çĺŃãĂĄPythonçž£çĺŃçŽĎæůůåŘĹä¡£çŤĺãĂĆ
åŔłèęĄä¡ăçąőä£Ìèğčé ĞŁåŹĺèćńæ■čçąőçŽĎåĹİåğŃåŇŰïijŇåźűäÿŤæűĽ'åŔŁåĹřèğčéĞŁåŹĺçŽĎCäžčçăĄæĽ'ğ

èęĄæşĺæĎŘçŽĎæŸŕèřČçŤĺ `PyGILState_Ensure()`
åźűäÿ■äijŽçńŃåĹżæŁćå■ăæĹŰäÿ■æŰ■èğčéĞŁåŹĺãĂĆ åęĆæđIJæIJL'åĚűäžŰäžčçăĄæ■čåIJĺæĽ'ğèąŇïijŇè£Ź
åIJĺåĘĚéČĺïijŇèğčéĞŁåŹĺäijŽæĽ'ğèąŇåŚĺæIJ§æĂğçŽĎçž£çĺŃåĹĞæ■ćïijŇåŻăæ■d'åęĆæđIJåĚűäžŰçž£çĺŃå
èřČçŤĺèĂĚæIJĂçżĹè£ŸæŸŕåŕźèè£ŘèąŇçŽĎïijĹår¡çőąåŔŕèČ¡èęĄåĚĹç■L'äÿĂäijŽïijL'ãĂĆ

15.9 çŤĺWSIGåŇĚèčĚCäžčçăĄ

éŮőécŸ

ä¡ăæČşèől'ä¡ăåEŹçŽĎCäžčçăĄä¡IJäÿžäÿĂäÿłCæL'l'åśŢæĺąåİŮæİèèő£éŰőïijŇæČşéĂŽè£Ğä¡£çŤĺ
SwigåŇĚèčĚçŤ§æĹŘåŹĺ æÌěåőŇæĹŘãĂĆ

èğčåĘşæŰźæąĹ

SwigéĂŽè£ĞèğčædŘCåď'ť'æŰĞäžűåźűèĞłåŁĺåĹŻåžžæĽ'ľ'åśŢäžčçăĄæİěæŞ■ä¡IJãĂĆ
èęĄä¡£çŤĺåőČïijŇä¡ăåĚĹèęĄæIJL'äÿĂäÿłCåď'ť'æŰĞäżűãĂĆä¿ŃåęĆïijŇæĹŚäžňçď'žä¿ŃçŽĎåď'ť'æŰĞäžűåę

```
/* sample.h */

#include <math.h>
extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

äÿĂæŮęä¡ăæIJL'äžĘè£Źäÿłåď'ť'æŰĞäžűïijŇäÿŃäÿĂæ■ěářśæŸŕçijŰåĘŹäÿĂäÿłSwig"æŐěåŔč"æŰĞäžű
æŇL'çĚğçžęåőŽïijŇè£ŹäžŻæŰĞäžűäžě".i"åŘŐçijĂåźűäÿŢçśžäijijäÿŃéÍćè£Źæăůïijž

```
// sample.i - Swig interface
%module sample
%{
#include "sample.h"
%}

/* Customizations */
%extend Point {
    /* Constructor for Point objects */
    Point(double x, double y) {
        Point *p = (Point *) malloc(sizeof(Point));
        p->x = x;
        p->y = y;
        return p;
   };
};

/* Map int *remainder as an output argument */
%include typemaps.i
%apply int *OUTPUT { int * remainder };

/* Map the argument pattern (double *a, int n) to arrays */
%typemap(in) (double *a, int n)(Py_buffer view) {
  view.obj = NULL;
  if (PyObject_GetBuffer($input, &view, PyBUF_ANY_CONTIGUOUS |␣
↪PyBUF_FORMAT) == -1) {
    SWIG_fail;
  }
  if (strcmp(view.format,"d") != 0) {
```

```
    PyErr_SetString(PyExc_TypeError, "Expected an array of doubles
↪");
    SWIG_fail;
  }
  $1 = (double *) view.buf;
  $2 = view.len / sizeof(double);
}

%typemap(freearg) (double *a, int n) {
  if (view$argnum.obj) {
    PyBuffer_Release(&view$argnum);
  }
}

/* C declarations to be included in the extension module */

extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

äÿĂæŮęä¡ăåĘŹåě¡äžĘæŐěåŔčæŰĞäžűïijÑářśåŔŕäžěåIJĺåŚ¡äžd'èąŇåůěåĚůäÿ■èřČçŤĺSwigäžĘïijŽ

```
bash % swig -python -py3 sample.i
bash %
```

swigçŽĎè¿ŞåĞžářśæŸŕäÿd'äÿłæŰĞäžűïijÑsample_wrap.cåŠŇsample.pyãĂĆ åŘŐéÍćçŽĎæŰĞäžűářśæŸŕçŤĺæĹůéIJĂèęĄåŕijåĚěçŽĎãĂĆ èĂŇsample_wrap.cæŰĞäžűæŸŕéIJĂèęĄèćńçijŰèŕŚåĹřåŘ■åŔń _sample çŽĎæŤŕæŇĄæĺąåİŮçŽĎCäžčçăĄãĂĆ è£ŹäÿłåŔŕäžěéĂŽè£Ğèů§æŹőéĂŽæL'l'åśŢæĺąåİŮäÿĂæăůçŽĎæŁĂæä¿ŇåęĆïijŇä¡ăåĹŻåžžäžĘäÿĂäÿłåęĆäÿŇæL'Ăçd'žçŽĎ `setup.py` æŰĞäžűïijŽ

```
# setup.py
from distutils.core import setup, Extension

setup(name='sample',
      py_modules=['sample.py'],
      ext_modules=[
        Extension('_sample',
                  ['sample_wrap.c'],
                  include_dirs = [],
                  define_macros = [],

                  undef_macros = [],
```

```
                    library_dirs = [],
                    libraries = ['sample']
                    )
        ]
)
```

èęAçijŰèŕŚåŠŇæţŃèŕŢïijŇåIJĺsetup.pyäÿŁæL'ğèąŇpython3ïijŇåęĆäÿŃïijŽ

```
bash % python3 setup.py build_ext --inplace
running build_ext
building '_sample' extension
gcc -fno-strict-aliasing -DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-
 ↪prototypes
-I/usr/local/include/python3.3m -c sample_wrap.c
 -o build/temp.macosx-10.6-x86_64-3.3/sample_wrap.o
sample_wrap.c: In function âĂŸSWIG_InitializeModuleâĂŹ:
sample_wrap.c:3589: warning: statement with no effect
gcc -bundle -undefined dynamic_lookup build/temp.macosx-10.6-x86_64-
 ↪3.3/sample.o
 build/temp.macosx-10.6-x86_64-3.3/sample_wrap.o -o _sample.so -
 ↪lsample
bash %
```

åęĆædIJäÿĂåĹĞæ■čåÿÿçŽĎèŕİïijŇä¡ăäijŽåŔŚçŐřä¡ăåřšåŔŕäžěå¿ĹæŰźä¿£çŽĎä¡£çŤĺçŤ§æĹŘçŽĎCæĽ

```
>>> import sample
>>> sample.gcd(42,8)
2
>>> sample.divide(42,8)
[5, 2]
>>> p1 = sample.Point(2,3)
>>> p2 = sample.Point(4,5)
>>> sample.distance(p1,p2)
2.8284271247461903
>>> p1.x
2.0
>>> p1.y
3.0
>>> import array
>>> a = array.array('d',[1,2,3])
>>> sample.avg(a)
2.0
>>>
```

èőĺèőž

SwigæŸŕPythonåŐĘåŔšäÿ■æđĎåžžæL'ľåśŢæĺąåİŮçŽĎæIJĂåŔd'èĂĄçŽĎåůěåĚůäźŃäÿĂãĂĆ SwigèČ¡èĞłåŁĺåŇŰå¿Ĺåd'ŽåŇĚèčĚçŤ§æĹŘåŹĺçŽĎåd'ĎçŘĘãĂĆ

æĽĂæIJĽSwigæŐěåŔčéČ¡äžěçśžäijijäÿŃéİćè£Źæăůç ŽĎäÿžåijĂåď ťïijŽ

```
%module sample
%{
#include "sample.h"
%}
```

è£ŹäÿłäżĚäżĚåŔłæŸŕåčřæŸŐäžĘæĽ'ľåśŢæĺąåİŮçŽĎåŘ■çğřåźűæŇĞåőŽäžĘCåď ťæŰĞäżűïijŇ
äÿžäžĘèČ¡èőľçijŰèŕŚéĂŽè£Gå£Ěéążèę Ąå ŇĚåŔńè£ŹäžŽåď ťæŰĞäżűïijĹä¡■äžŐ %{ åŠŇ %}
çŽĎäżčçăĄïijĽïijŇ åřĘåőČäžňäźŃéŮť åď ■åĹűçšŸèť ťåĹřè¿ŞåĞžäžčçăĄäÿ■ïijŇè£Źäź§æŸŕä¡äèę ĄæŤ¿ç¡őæ

SwigæŐěåŔčçŽĎåžŢäÿŃéČĺåĹĘæŸŕäÿĂäÿłCåčřæŸŐåĹŮèąĺïijŇä¡äéIJĂèę ĄåIJĺæĽ'ľåśŢäÿ■åŇĚåŔńåő
è£ŹéĂŽåÿÿäżŐåď ťæŰĞäżűäÿ■ècńåď ■åĹűãĂĆåIJĺæĹŚäžňçŽĎä¿Ńå■Řäÿ■ïijŇæĹŚäžňäżĚäżĚåČŔäÿŃéİćè£

```
%module sample
%{
#include "sample.h"
%}
...
extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

æIJĽäÿĂçĆźéIJĂèę Ąåijžěřčç ŽĎæŸŕè£ŹäžŽåčřæŸŐäijŽåŚŁèŕĽSwigä¡ăæČşèę ĄåIJĺPythonæĺąåİŮäÿ■å
éĂŽåÿÿä¡ăéIJĂèę AçijŰè¿Śè£ŹäÿłåčřæŸŐåĹŮèąĺæĹŮçŽÿåžŢçŽĎä£őæŤźäÿŃåőĆăĂĆ
ä¿ŃåęĆïijŇåęĆæđIJä¡ääÿ■æČşæ§ŘäžŽåčřæŸŐècńåŇĚåŔńè£ŽæİěïijŇä¡äèę ĄårĘåőČäžŐåčřæŸŐåĹŮèąĺäÿ■

ä¡£çŤĺSwigæIJĂåď ■æİĆçŽĎåIJřæŰźæŸŕåőČèČ¡çżŹCäžčçăĄæŔŘä¿Žåď ğéĞŔçŽĎèĞłåőŽäźĽæŞ■ä¡IJ
è£ŹäÿłäÿżécŸåď łåď ğïijŇè£ŹéĞŇæŮăæşŢåśŢåijĂïijŇä¡ĘæŸŕæĹŚäžňåIJĺæIJńèŁĆè£ŸåĽľåśŢçď žäžĘäÿĂäź

çňňäÿĂäÿłèĞłåőŽäźĽæŸŕ `%extend` æŇĞäžď åĚĄèőÿæŰźæşŢècńéŹĎåŁääĹřåůšå■ŸåIJĺçŽĎçżŞæđĎä
æĹŚä¿Ńå■Řäÿ■ïijŇè£ŹäÿłèćńçŤĺæİěæůžåŁääÿĂäÿłPointçżŞæđĎä¡ŞçŽĎæđĎéĂääŹĺæŰźæşŢãĂĆ
åőĆåŔŕäžěèőľä¡ääČŔäÿŃéİćè£Źæăůä¡£çŤĺè£ŹäÿłçżŞæđĎä¡ŞïijŽ

```
>>> p1 = sample.Point(2,3)
>>>
```

åęĆæđIJçŢěè£ĞçŽĎèŕĺïijŇPointåŕźèśąåřśå£ĚéążäžěæŻť åŁääď ■æİĆçŽĎæŰźåijŔæİěècńåĹŽåžžïijŽ

```
>>> # Usage if %extend Point is omitted
>>> p1 = sample.Point()
>>> p1.x = 2.0
>>> p1.y = 3
```

çňňäžŇäÿłèĞłåőŽäźĽæűĽåŔŁåĹřåŕź `typemaps.i` åžŞçŽĎåijŢåĚěåŠŇ

%apply æŇĞäžđ'ïijŇ åőČäijŽæŇĞçđ'žSwigåŔĆæŢřç■¿åŘ■ int *remainder èęĄèćńå¡ŞåĄŽæŸŕè¿ŞåĞžåĂijãĂĆ è£ŹäÿłåőđéŹĚäÿŁæŸŕäÿĂäÿłæĺąåijŔåŇźéĚ■èğĎåĹŹãĂĆ åIJĺæŐĕäÿŇæİěçŽĎæĽĂæIJL'åčřæŸŐäÿ■ïijŇäżżä¡ŢæŮűåĂŹåŘłèęĄçćřäÿŁ int *remainder ïijŇäżŰåřśäijŽèćńä¡IJäÿžè¿ŞåĞžãĂĆ è£ŹäÿłèĞłåőŽäźĽ'æŰźæşŢåŔŕäžĕèőľ' divide() åĞ¡æŢřè£ŤåŻđäÿđ'äÿłåĂijãĂĆ

```
>>> sample.divide(42,8)
[5, 2]
>>>
```

æIJĂåŘŐäÿĂäÿłæűĽ'åŘŁåĹř %typemap æŇĞäżđ'çŽĎèĞłåőŽäźĽ'åŔŕèČ¡æŸŕè£ŹéĞŇåśŢçđ'žçŽĎæIJĂ äÿĂäÿłtypemapåřśæŸŕäÿĂäÿłåIJĺè¿ŞåĚĕäÿ■çĽźåőŽåŔĆæŢřæĺąåijŔçŽĎèğĎåĹŹãĂĆ åIJĺæIJňèŁĆäÿ■ïijŇäÿĂäÿłtypemapèćńåőŽäźĽäÿžåŇźéĚ■åŔĆæŢřæĺąåijŔ (double *a, int n). åIJĺtypemapåĘĚéČĺæŸŕäÿĂäÿłCäżčçăĄçĽĞæőţïijŇåőČåŚŁèŕĽSwigæĂŐæăůåřĘäÿĂäÿłPythonår æIJňèŁĆäżčçăĄä¡£çŤĺäžĘPythonçŽĎçijŞå■Ÿå■ŔèőőåŏžåŇźéĚ■äżżä¡ŢçIJŃäÿŁåŐżçśżäijijåŔŇçš¿åžęæŢřçż ïijĹæŕŤåęĆNumPyæŢřçżĎãĂĄarrayæĺąåİŮåĹŽåżżçŽĎæŢřçżĎç■Ľ'ïijĽ'ïijŇæŻť'åđ'Žèŕůåŕ ĆèĂČ15.3åřŔèŁĆ

åIJĺtypemapäżčçăĄåĘĚéČĺïijŇ$1åŠŇ$2è£ŹæăůçŽĎåŔŸéĞŔæŻ£æ■ćäijŽèŐůåŔŰtypemapæĺąåijŔçŽĎC ïijĹæŕŤåęĆ$1æŸăåřĎäÿž double *a ïijĽãĂĆ$inputæŇĞåŘŚäÿĂäÿłä¡IJäÿžè¿ŞåĚĕçŽĎ PyObject * åŔĆæŢřïijŇ èĂŇ $argnum åřśäżčèąĺåŔĆæŢřçŽĎäÿłæŢřãĂĆ

çijŰåĘŹåŠŇçŘĘèğčtypemapsæŸŕä¡£çŤĺSwigæIJĂå§žæIJňçŽĎåĽ■æŔŘãĂĆ äÿ■äżĚæŸŕèŕťäżčçăĄæŻť'çđđçğŸïijŇèĂŇäÿŤä¡äéIJĂèęĄçŘĘèğčPython C APIåŠŇSwigåŠŇåőČäžđ'äžŠçŽĎæŰźåijŔãĂĆ SwigæŰĞæąčæIJL'æŻť'åđ'Žè£ŹæŰźéİćçŽĎçżĘèŁĆïijŇåŔŕäż

äÿ■è£ĞïijŇåęĆæđIJä¡ăæIJL'åđ'ğéĞŔçŽĎCäżčçăĄéIJĂèęĄèćńæŽť'éIJšäÿžæĽ'ľ'åśŢæĺąåİŮãĂĆ SwigæŸŕäÿĂäÿłéİđåÿÿåijžåđ'ğçŽĎåůěåĚůãĂĆåĚşéŤőçĆźåIJĺäžŐSwigæŸŕäÿĂäÿłåđ'ĎçŘĘCåčřæŸŐçŽĎçij éĂŽè£Ğåijžåđ'ğçŽĎæĺąåijŔåŇźéĚ■åŠŇèĞłåőŽäźĽçżĎäżűïijŇåŔŕäżèőľä¡äæŻť'æŤźåčřæŸŐæŇĞåőŽåŠŇçś æŻť'åđ'Žä£ąæĄŕèŕůåŐżæ§ěéŸĚ Swigç¡ŚçńŹ ïijŇ è£ŸæIJL' çĽźåőŽäžŐPythonçŽĎçŻÿåĚşæŰĞæąč

15.10 çŤĺCythonåŇĚèčĚCäżčçăĄ

éŮőécŸ

ä¡äæČşä¡£çŤĺCythonæİěåĹŽåżžäÿĂäÿłPythonæĽ'ľ'åśŢæĺąåİŮïijŇçŤĺæİěåŇĚèčĚæ§Řäÿłåůšå■ŸåIJĺçŽĎ

èğčåĘşæŰźæąĹ

ä¡£çŤĺCythonæđĎåżžäÿĂäÿłæĽ'ľ'åśŢæĺąåİŮçIJŃäÿŁåŐżå¿ĹæĽ'ŃåĘŹæĽ'ľ'åśŢæIJL'äžŽçśżäijijïijŇ åŻääÿżä¡äéIJĂèęĄåĹŽåżžå¿Ĺåđ'ŽåŇĚèčĚåĞ¡æŢřãĂĆäÿ■è£ĞïijŇèů§åĽ'■éİćäÿ■åŘŇçŽĎæŸŕïijŇä¡ääÿ■éIJĂ

ä¡IJäÿžåĞĘåđ'ĞïijŇåĄĞèő¿æIJňçńääżŃçż■éČĺåĹĘçŽĎçđ'žä¿ŃäžčçăĄåůšçżŔèćńçijŰèŕŠåĹřæ§Řäÿłåŕ libsample çŽĎCåĞ¡æŢřåžŞäÿ■äžĘãĂĆ éęŰåĚĹåĹŽåżžäÿĂäÿłåŘ■åŔń csample.pxd çŽĎæŰĞäżűïijŇåęĆäÿŃæĽ'Äçđ'žïijŽ

```
# csample.pxd
#
# Declarations of "external" C functions and structures

cdef extern from "sample.h":
    int gcd(int, int)
    bint in_mandel(double, double, int)
    int divide(int, int, int *)
    double avg(double *, int) nogil

    ctypedef struct Point:
         double x
         double y

    double distance(Point *, Point *)
```

è£ŹäÿłæŰĞäžűåIJÍCythonäÿ■çŽĎä¡IJçŤÍåřśèů§CçŽĎåď'ťæŰĞäžűäÿĂæăůãĂĆ åĹİåğŃåčřæŸŐ `cdef extern from "sample.h"` æŇĞåőŽäžĘæĽĂå■ęçŽĎCåď'ťæŰĞäžűãĂĆ æŐěäÿŃæİěçŽĎåčřæŸŐéČ¡æŸŕæİěèĞłäžŐéĆčäÿłåď'ťæŰĞäžűãĂĆæŰĞäžűåŘ■æŸŕ `csample.pxd` ïijŇèĂŇäÿ■æŸŕ `sample.pxd` âĂŤâĂŤè£ŹçĆźå¿ĹéĞ■èęĄãĂĆ

äÿŃäÿĂæ■ěïijŇåĹŻåżžäÿĂäÿłåŘ■äÿž `sample.pyx` çŽĎéŮőécŸãĂĆ èřěæŰĞäžűäijŽåőŽäźĽåŇĚèčĚåŹĺïijŇçŤÍæİěæąěæŐěPythonèğčéĞŁåŹĺåĹř `csample.pxd` äÿ■åčřæŸŐçŽĎCäžčçăĄãĂĆ

```
# sample.pyx

# Import the low-level C declarations
cimport csample

# Import some functionality from Python and the C stdlib
from cpython.pycapsule cimport *

from libc.stdlib cimport malloc, free

# Wrappers
def gcd(unsigned int x, unsigned int y):
    return csample.gcd(x, y)

def in_mandel(x, y, unsigned int n):
    return csample.in_mandel(x, y, n)

def divide(x, y):
    cdef int rem
    quot = csample.divide(x, y, &rem)
    return quot, rem

def avg(double[:] a):
    cdef:
        int sz
        double result
```

```
    sz = a.size
    with nogil:
        result = csample.avg(<double *> &a[0], sz)
    return result

# Destructor for cleaning up Point objects
cdef del_Point(object obj):
    pt = <csample.Point *> PyCapsule_GetPointer(obj,"Point")
    free(<void *> pt)

# Create a Point object and return as a capsule
def Point(double x,double y):
    cdef csample.Point *p
    p = <csample.Point *> malloc(sizeof(csample.Point))
    if p == NULL:
        raise MemoryError("No memory to make a Point")
    p.x = x
    p.y = y
    return PyCapsule_New(<void *>p,"Point",<PyCapsule_Destructor>
↪del_Point)

def distance(p1, p2):
    pt1 = <csample.Point *> PyCapsule_GetPointer(p1,"Point")
    pt2 = <csample.Point *> PyCapsule_GetPointer(p2,"Point")
    return csample.distance(pt1,pt2)
```

èŕřæŰĞäżűæŻťåď ŽçŽĎçżĘèŁĆéČĺåĹĘäijŽåIJĺèőĺèőžéČĺåĹĘèŕęçżĘåśŢåijĂãĂĆ æIJĂåŘŐïijŇäÿžäžĘæđĎåžžæĽľåšŢæĺąåİŮïijŇåČŔäÿŃéİćè£ŹæăůåĹŻåžžäÿĂäÿł `setup.py` æŰĞäżűïijŽ

```
from distutils.core import setup
from distutils.extension import Extension
from Cython.Distutils import build_ext

ext_modules = [
    Extension('sample',

              ['sample.pyx'],
              libraries=['sample'],
              library_dirs=['.'])]
setup(
  name = 'Sample extension module',
  cmdclass = {'build_ext': build_ext},
  ext_modules = ext_modules
)
```

èęĄæđĎåžžæĹŚäžňæţŃèŕŢçŽĎçŻőæăĞæĺąåİŮïijŇåČŔäÿŃéİćè£ŹæăůåĄŽïijŽ

```
bash % python3 setup.py build_ext --inplace
running build_ext
cythoning sample.pyx to sample.c
building 'sample' extension
gcc -fno-strict-aliasing -DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-
 ↪prototypes
 -I/usr/local/include/python3.3m -c sample.c
 -o build/temp.macosx-10.6-x86_64-3.3/sample.o
gcc -bundle -undefined dynamic_lookup build/temp.macosx-10.6-x86_64-
 ↪3.3/sample.o
  -L. -lsample -o sample.so
bash %
```

åęĆæđIJäÿĂåĹĞéąžåĹľçŽĎèŕĺïijÑä¡ääžŤèŕěæIJL'äžĘäÿĂäÿłæL'ľåśŢæĺąåİŮ `sample.so` ïijÑåŔŕåIJĺäÿŃéİćä¿Ńå■Řäÿ■ä¡£çŤĺïijŽ

```
>>> import sample
>>> sample.gcd(42,10)
2
>>> sample.in_mandel(1,1,400)
False
>>> sample.in_mandel(0,0,400)
True
>>> sample.divide(42,10)
(4, 2)
>>> import array
>>> a = array.array('d',[1,2,3])
>>> sample.avg(a)
2.0
>>> p1 = sample.Point(2,3)
>>> p2 = sample.Point(4,5)
>>> p1
<capsule object "Point" at 0x1005d1e70>
>>> p2
<capsule object "Point" at 0x1005d1ea0>
>>> sample.distance(p1,p2)
2.8284271247461903
>>>
```

èőĺèőž

æIJňèŁĆåŇĚåŘńäžĘå¿Ĺåď'ŽåL'■éİćæL'Ăèőšç'ĎďéńŸçžğçL'źæĂğïijŇåŇĚæŃňæŢřçżĎæŞ■ä¡IJãĂĄåŇĚæŕŔäÿĂéČĺåĹĘéČ¡äijŽéĂŘäÿłèćńèőšè£řåĹřïijÑä¡ĘæŸŕæĹŚäžňæIJĂåě¡èČ¡åď'■äźääÿĂäÿŇåL'■éİćåĞåářŔèŁåIJĺéąűåśĆïijÑä¡£çŤĺCythonæŸŕå§žäžŐCäźŃäÿŁãĂĆ.pxdæŰĞäżűäžĚäżĚåŔłåŇĚåŘńCåőŽäźL'ïijĹçśżäijij.hæ.pyxæŰĞäżűåŇĚåŘńäžĘåőđçŐřïijĹçśżäijij.cæŰĞäżűïijL'ãĂĆcimportèŕ■åŔèèćńCythonçŤĺæİěåŕijåĚě.pxdæŰĞäżűäÿ■çŽĎåőŽäźL'ãĂĆåőČèů§ä¡£çŤĺæŹőéĂŽçŽĎåŁäè¡¡PythonæĺąåİŮçŽĎåŕijåĚèŕ■åŔèæŸŕäÿ■åŘŇçŽĎãĂĆ

åř¡çőą *.pxd* æŰĞäżűåŇĚåŘńäžĘåőŽäźL'ïijÑä¡ĘåőČäżňåźűäÿ■æŸŕçŤĺæİèèĞłåŁĺåĹŻåżžæL'ľåśŢäżčçăĄç

åŽžæ■d'ïijŇä¡ãè£ŸæŸŕèęĄåĘŹåŇĚèčĚåĞ¡æŢřãĂĆä¿ŃåęĆïijŇårśçőŮ `csample.pxd` æŰĞäžűåčřæŸŐäžĘ `int gcd(int, int)` åĞ¡æŢřïijŇ ä¡ääž■çĎűéIJĂèęĄåIJĺ `sample.pyx` äÿ■äÿžåőČåĘŹäÿĂäÿłåŇĚèčĚåĞ¡æŢřãĂĆä¿ŃåęĆïijŽ

```
cimport csample

def gcd(unsigned int x, unsigned int y):
    return csample.gcd(x,y)
```

åŕźäžŐçőĂå■ŢçŽĎåĞ¡æŢřïijŇä¡ääźűäÿ■éIJĂèęĄåŐżåĄŽåd'łåd'ŽçŽĎæŮűãĂĆ CythonäijŽçŢ§æĹŘåŇĚèčĚäžčçăĄæÌěæ■čçąőçŽĎè¡ñæ■ćåŔĆæŢřåŠŇè£ŤåŻđåĂijãĂĆ çżŚåőŽåĹřåśđæĂğäÿŁçŽĎCæŢřæ■őçśżåđŇæŸŕåŘŕéĂL'çŽĎãĂĆäÿ■è£ĞïijŇåęĆæđIJä¡ääŇĚåŘńäžĘåőČäžñ ä¿ŃåęĆïijŇåęĆæđIJæIJL'äžžä¡£çŤĺèt'§æŢřæÌěěřČçŤĺè£Źäÿłåĝ¡æŢřïijŇäijŽæŁŽåĞžäÿĂäÿłåijĆåÿÿïijŽ

```
>>> sample.gcd(-10,2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "sample.pyx", line 7, in sample.gcd (sample.c:1284)
    def gcd(unsigned int x,unsigned int y):
OverflowError: can't convert negative value to unsigned int
>>>
```

åęĆæđIJä¡ăæČşåŕźåŇĚèčĚåĞ¡æŢřåĄŽåŘęåd'ŰçŽĎæčĂæ§ëïijŇåŘłéIJĂèęĄä¡£çŤĺåŘęåd'ŰçŽĎåŇĚèčĚ

```
def gcd(unsigned int x, unsigned int y):
    if x <= 0:
        raise ValueError("x must be > 0")
    if y <= 0:
        raise ValueError("y must be > 0")
    return csample.gcd(x,y)
```

åIJĺcsample.pxdæŰĞäžűäÿ■çŽĎ‘‘in_mandel()‘‘ åčřæŸŐæIJL'äÿłå¿ĹæIJL'èűčä¡ĘæŸŕæŕŤè¿ČéŽ¿çŘĘèğ åIJĺè£ŹäÿłæŰĞäžűäÿ■ïijŇåĞ¡æŢřèćńåčřæŸŐäÿžçĎűåŘŐäÿĂäÿłbintèĂŇäÿ■æŸŕäÿĂäÿłintãĂĆ åőČäijŽèől'åĞ¡æŢřåĹŽåžžäÿĂäÿłæ■čçąőçŽĎBooleanåĂijèĂŇäÿ■æŸŕçőĂå■ŢçŽĎæŢt'æŢřãĂĆ åŽžæ■d'ïijŇè£ŤåŽđåĂij0èąĺçd'žFalseèĂŇ1èąĺçd'žTrueãĂĆ

åIJĺCythonåŇĚèčĚåŹĺäÿ■ïijŇä¡ååŕŕäžěéĂL'æŃl'åčřæŸŐCæŢřæ■őçśżåđŇïijŇäź§åŘŕäžěä¡£çŤĺæL'ĂæIJ åŕźäžŐ `divide()` çŽĎåŇĚèčĚåŹĺåśŢçd'žäžĘè£ŹæăůäÿĂäÿłä¿Ńå■ŘïijŇåŘŇæŮűè£ŸæIJL'åęĆä¡ŢåŐžåd'Ď

```
def divide(x,y):
    cdef int rem
    quot = csample.divide(x,y,&rem)
    return quot, rem
```

åIJĺè£ŹéĞŇïijŇ`rem` åŘŸéĞŘèćńæŸ¿çd'žçŽĎåčřæŸŐäÿžäÿĂäÿłCæŢt'ådŃåŘŸéĞŘãĂĆ å¡ŞåőČèćńäijăåĚě `divide()` åĞ¡æŢřçŽĎæŮűåĂŹïijŇ`&rem` åĹŽåžžäÿĂäÿłèů§CäÿĂæăůçŽĎæŇĞåŘŚåőČçŽĎæŇĞéŠĹãĂĆ `avg()` åĞ¡æŢřçŽĎäžčçăĄæijŤçd'žäžĘCythonæŻt'énŸçžğçŽĎçL'źæĂğãĂĆ éęŰåĚĹ `def avg(double[:] a)` åčřæŸŐäžĘ `avg()` æŐěåŘŮäÿĂäÿłäÿĂçżt'çŽĎåŘŇçš¿åžęåĘĚå■ŸèğĘåŻ¿ãĂĆ æIJĂæČŁåěĞçŽĎéČĺåĹĘæŸŕè£ŤåŽđçŽĎçżŞæđ

```
>>> import array
>>> a = array.array('d',[1,2,3])
>>> import numpy
>>> b = numpy.array([1., 2., 3.])
>>> import sample
>>> sample.avg(a)
2.0
>>> sample.avg(b)
2.0
>>>
```

åIJÍæ■d'åŇĚèčĚåŹÍäÿ■ïijŇa.size0 åŠŇ &a[0] åĹĘåĹńåijŢçŤÍæŢřçżĎåĚČçť'ääÿłæŢřåŠŇåžŢåśĆæŇèŕ■æşŢ <double *> &a[0] æŢŹä¡ăæĂŐæăůåřĘæŇĞéŠĹè¡ňæ■ćäÿžäÿ■åŘŇçŽĎçśžåđŃãĂĆ åL'■æŔŘæŸŕCäÿ■çŽĎ avg() æŐěåŔŮäÿĂäÿłæ■čçąőçśžåđŃçŽĎæŇĞéŠĹãĂĆ åŔĆèĂČäÿŃäÿĂèŁĆåĚşäžŐCythonåĘĚå■ŸèğĘåŻ¿çŽĎæŻť'énŸçžğèőšè£řãĂĆ

éŹď'äžĘåď'ĎçŘĘéĂŽåÿÿçŽĎæŢřçżĎåď'ŰïijŇavg() çŽĎè£Źäÿłä¿Ńå■Řè£ŸåśŢçď'žäžĘåęĆä¡Ţåď'ĎçŘèŕ■åŔě with nogil: åčřæŸŐäžĘäÿĂäÿłäÿ■éIJĂèęĄGILåřśèČ¡æL'ğèąŇçŽĎäżčçăĄåİŮãĂĆ åIJÍè£ŹäÿłåİŮäÿ■ïijŇäÿ■èČ¡æIJL'äżżä¡ŢçŽĎæŹőéĂŽPythonåŕźèśąâĂŤâĂŤåŔłèČ¡ä¡£çŤÍèćńåčřæŸŐäÿž cdef çŽĎåŕźèśąåŠŇåĞ¡æŢřãĂĆ åŘęåď'ŰïijŇåď'ŰéČÍåĞ¡æŢřå£Ěéążç Őřåőđç ŽĎåčřæŸŐåőĆäžňèČ¡äÿ■ä¿Ìè ęåŽăæ■d'ïijŇåIJÍcsample.pxdæŰĞäżűäÿ■ïijŇavg() èćńåčřæŸŐäÿž double avg(double *, int) nogil.

åŕźPointçżŞæđĎä¡ŞçŽĎåď'ĎçŘĘæŸŕäÿĂäÿłæŇŚæĹŸãĂĆæIJňèŁĆä¡£çŤÍèČűåŽŁåŕźèśąåŕĘPointåŕźèśą èęĄè£ŹæăůåĄŽçŽĎèŕÌïijŇåžŢåśĆCythonäžčçăĄçÍ■å¿őæIJL'çĆźåď'■æİĆãĂĆ éęŰůĚĹïijŇäÿŇéİćçŽĎåŕijåĚèècńçŤÍæÌěåijŢåĚěCåĞ¡æŢřåžŞåŠŇPython C APIäÿ■åőŽäźL'çŽĎåĞ¡æŢřïijŽ

```
from cpython.pycapsule cimport *
from libc.stdlib cimport malloc, free
```

åĞ¡æŢř del_Point() åŠŇ Point() ä¡£çŤÍè£ŹäÿłåŁ§èČ¡æÌěåĹŽåžžäÿĂäÿłèČűåŽŁåŕźèśąïijŇ åőČäijŽåŇĚèčĚäÿĂäÿł Point * æŇĞéŠĹãĂĆcdef del_Point() åřĘ del_Point() åčřæŸŐäÿžäÿĂäÿłåĞ¡æŢřïijŇ åŘłèČ¡éĂŽè£ĞCythonèő£éŮőïijŇèĂŇäÿ■èČ¡äżŐPythonäÿ■èő£éŮőãĂĆ åŽăæ■d'ïijŇè£ŹäÿłåĞ¡æŢřåŕźåď'ŰéČÍæŸŕäÿ■åŘŕèğĄçŽĎâĂŤâĂŤåőČèćńçŤÍæÌěå¡ŞåĄŽäÿĂäÿłåŽđěřČåĞ¡æ åĞ¡æŢřěŘçŤÍæŕŤåęĆ PyCapsule_New() ãĂĄPyCapsule_GetPointer() çŻť'æŐěæÌèèĞłPython C APIåźűäÿŤäžěåŘŇæäůçŽĎæŰźåijŘèćńä¡£çŤÍãĂĆ

distance åĞ¡æŢřäžŐ Point() åĹŽåžžçŽĎèČűåŽŁåŕźèśąäÿ■æŔŘåŔŰæŇĞéŠĹãĂĆ è£ŹéĞŇèęĄæşÍæĎŘçŽĎæŸŕä¡ääÿ■éIJĂèęĄæŃĚå£ČåijĆåÿÿåď'ĎçŘĘãĂĆ åęĆæđIJäÿĂäÿłéŤŹèŕŕçŽĎåŕźèśąèćńäijăè£ŻæÌěïijŇPyCapsule_GetPointer() äijŽæŁŽåĞžäÿĂäÿłåijĆåÿÿïijŇ ä¡ĘæŸŕCythonåůšçżŘç§ěéĄŞæĂŐäźĹæ§ěæL'¿åĹřåőČïijŇåźűåřĘåőČäžŐ distance() äijăéĂŠåĞžåŐžãĂĆ

åď'ĎçŘĘPointçżŞæđĎä¡ŞäÿĂäÿłçijžçĆźæŸŕåőČçŽĎåőđçŐřæŸŕäÿ■åŔŕèĞĄçŽĎãĂĆ ä¡ăäÿ■èČ¡èő£éŮőäżžä¡ŢåśđæĂğæÌěæ§ěçIJŃåőČçŽĎåĘĚéČĺãĂĆ è£ŹéĞŇæIJL'åŘęåď'ŰäÿĂçğ■æŰźæşŢåŐžåŇĚèčĚåőČïijŇåřśæŸŕåőŽäźL'äÿĂäÿłæL'l'åśŢçśžåđŃïijŇåęĆäÿŃ

```
# sample.pyx

cimport csample
```

```
from libc.stdlib cimport malloc, free
...

cdef class Point:
    cdef csample.Point *_c_point
    def __cinit__(self, double x, double y):
        self._c_point = <csample.Point *> malloc(sizeof(csample.
↪Point))
        self._c_point.x = x
        self._c_point.y = y

    def __dealloc__(self):
        free(self._c_point)

    property x:
        def __get__(self):
            return self._c_point.x
        def __set__(self, value):
            self._c_point.x = value

    property y:
        def __get__(self):
            return self._c_point.y
        def __set__(self, value):
            self._c_point.y = value

def distance(Point p1, Point p2):
    return csample.distance(p1._c_point, p2._c_point)
```

åIJÍè£ŹéĞŇïijŇcdifçśż `Point` åřĘPointåčřæŸŐäÿžäÿĂäÿłæĽľåśŢçśżåđŃãĂĆçśżåśđæĂğ `cdef csample.Point *_c_point` åčřæŸŐäžĘäÿĂäÿłåőđä¿ŃåŔŸéĞŔïijŇæŃěæIJĽäÿĂäÿłæŇĞåŘŠåžŢåśĆPointçżŞæđĎä¡ŞçŽĎæŇĞéŠĹãĂĆ `__cinit__()` åŠŇ `__dealloc__()` æŰźæşŢéĂŽè£Ğ `malloc()` åŠŇ `free()` åĹŻåżžåźűéŤĂæŕĄåžŢåśĆCçżŞæđĎä¡ŞãĂĆxåŠŇyåśđæĂğçŽĎåčřæŸŐèőľä¡ăèŐůåŔŰåŠŇèő¿ç¡őåžŢåśĆçżŞæđĎä¡ŞçŽĎåśđæĂğåÄijãĂĆ `distance()` çŽĎåŇĚèčĚåŹĺè£Ÿåŕäžěèćńä£őæŤźïijŇä¡£å¿ŮåőČèČ¡æŐěåŔŮ `Point` æĽľåśŢçśżåđŃåőđä¿Ńä¡IJäÿžåŔĆæŢřïijŇ èĂŇäijäéĂŠåžŢåśĆæŇĞéŠĹçżŹĆåĞ¡æŢřãĂĆ

åĄŽäžĘè£ŹäÿłæŤźåŔŸåŘŐïijŇä¡ääijŽåŔŞçŐřæŞ■ä¡IJPointåŕźèśąåřśæŸ¿å¿ŮæŻť åŁäèĞłçĎűäžĘïijŽ

```
>>> import sample
>>> p1 = sample.Point(2,3)
>>> p2 = sample.Point(4,5)
>>> p1
<sample.Point object at 0x100447288>
>>> p2
<sample.Point object at 0x1004472a0>
>>> p1.x
2.0
>>> p1.y
```

```
3.0
>>> sample.distance(p1,p2)
2.8284271247461903
>>>
```

æIJňèŁĆåůšçżŔæijŤçd'žäžĘå¿Ĺåd'ŽCythonçŽĎæäÿå£ČçĽźæĂğïijŇä¡ääŔŕäžěäżěæ■d'äÿžå§žåĞĘæİěæ
äÿ■è£ĞïijŇä¡ăæIJĂåě¡åĚĹåŐžéŸĚèŕżäÿŃåőŸæŰźæŰĞæąčæİěäžĘèğčæŻt'åd'Žä£qæĄŕãĂĆ

æŐěäÿŃæİěåĞăèŁĆè£ŸäijŽçżğçż■æijŤçd'žäÿĂäžŻCythonçŽĎåĚűäžŰçĽźæĂğãĂĆ

15.11 çŤÍCythonåĘŹéńŸæĂğèČ¡çŽĎæŢřçżĎæŞ■ä¡IJ

éŮőécŸ

ä¡ăèęĄåĘŹéńŸæĂğèČ¡çŽĎæŞ■ä¡IJæİěèĞłNumPyäźŃçśżçŽĎæŢřçżĎèőąçőŮåĞ¡æŢřãĂĆ
ä¡ăåůšçżŔç§ěéĄŞäžĘCythonè£ŹæăůçŽĎåůěåĚůäijŽèőľ'åőČåŔŸå¿ŮçőĂå■ŢïijŇä¡ĘæŸŕåźűäÿ■çąőåőŽèŕěæĂ

èğčåĘşæŰźæąĹ

ä¡IJäÿžäÿĂäÿłä¿Ńå■ŘïijŇäÿŃéİćçŽĎäžčçăĄæijŤçd'žäžĘäÿĂäÿłCythonåĞ¡æŢřïijŇçŤÍæİěä£őæŢt'äÿĂä

```
# sample.pyx (Cython)

cimport cython

@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip(double[:] a, double min, double max, double[:] out):
    '''
    Clip the values in a to be between min and max. Result in out
    '''
    if min > max:
        raise ValueError("min must be <= max")
    if a.shape[0] != out.shape[0]:
        raise ValueError("input and output arrays must be the same␣
↪size")
    for i in range(a.shape[0]):
        if a[i] < min:
            out[i] = min
        elif a[i] > max:
            out[i] = max
        else:
            out[i] = a[i]
```

èęĄçijŰèŕŚåŠŇædĎåžžè£ŹäÿłæL'l'åśŢïijŇä¡ăéIJĂèęĄäÿĂäÿłåČŔäÿŃéİćè£ŹæăůçŽĎ
`setup.py` æŰĞäžű ïijĹä¡£çŤÍ `python3 setup.py build_ext --inplace`
æİěædĎåžžåőČïijL'ïijŽ

```
from distutils.core import setup
from distutils.extension import Extension
from Cython.Distutils import build_ext

ext_modules = [
    Extension('sample',
              ['sample.pyx'])
]

setup(
  name = 'Sample app',
  cmdclass = {'build_ext': build_ext},
  ext_modules = ext_modules
)
```

ä¡ăäijŽåŔŚçŐřçżŞæđIJåĞ¡æŢřçąőåőđåŕźæŢřçżĎè£ŻèąŇçŽĎä£őæ■čïijŇåźűäÿŤåŔŕäżěéĂĆçŤĺäžŐåď'Žç

```
>>> # array module example
>>> import sample
>>> import array
>>> a = array.array('d',[1,-3,4,7,2,0])
>>> a

array('d', [1.0, -3.0, 4.0, 7.0, 2.0, 0.0])
>>> sample.clip(a,1,4,a)
>>> a
array('d', [1.0, 1.0, 4.0, 4.0, 2.0, 1.0])

>>> # numpy example
>>> import numpy
>>> b = numpy.random.uniform(-10,10,size=1000000)
>>> b
array([-9.55546017,  7.45599334,  0.69248932, ...,  0.69583148,
       -3.86290931,  2.37266888])
>>> c = numpy.zeros_like(b)
>>> c
array([ 0.,  0.,  0., ...,  0.,  0.,  0.])
>>> sample.clip(b,-5,5,c)
>>> c
array([-5.        ,  5.        ,  0.69248932, ...,  0.69583148,
       -3.86290931,  2.37266888])
>>> min(c)
-5.0
>>> max(c)
5.0
>>>
```

ä¡ăè£ŸäijŽåŔŚçŐřè£ŘèąŇçŤ§æĹŘçżŞæđIJéİđåÿÿçŽĎå£ńãĂĆ
äÿŃéİćæĹŚäżňåřĘæIJňä¿ŃåŠŇnumpyäÿ■çŽĎåůšå■ŸåIJĺçŽĎ `clip()`
åĞ¡æŢřåĄŽäÿĂäÿłæĂğèČ¡åŕźærŤïijŽ

```
>>> timeit('numpy.clip(b,-5,5,c)','from __main__ import b,c,numpy',
 ↪number=1000)
8.093049556000551
>>> timeit('sample.clip(b,-5,5,c)','from __main__ import b,c,sample
 ↪',
...           number=1000)
3.760528204000366
>>>
```

æ■čåęĆä¡ăçIJŃåĹřçŽĎïijŇåőČèęĄå£ńå¿Ĺåď ŽâĂŤâĂŤè£ŹæŸŕäÿĂäÿłå¿ĹæIJL'èűčçŽĎçżŞæđIJïijŇåŻă

èőĺèőž

æIJňèŁĆåĹl'çŤĺäžĘCythonçśžåđŃçŽĎåĘĚå■ŸèğĘåŻ¿ïijŇæđĄåď ğçŽĎçőĂåŇŰäžĘæŢřçżĎçŽĎæŞ■ä¡I
`cpdef clip()` åčřæŸŐäžĘ `clip()` åŘŇæŮűäÿžCçžğåĹńåĞ¡æŢřäžěåŔŁPythonçžğåĹńåĞ¡æŢřãĂĆ
åIJĺCythonäÿ■ïijŇè£ŹäÿłæŸŕå¿ĹéĞ■èęĄçŽĎïijŇåŻääÿžåőČèąĺçď žæ■ď åĞ¡æŢřěřČçŤĺèęĄæŕŤåĚűäžŰCytho
ïijĹæŕŤåęĆä¡ăæČşåIJĺåŔęåď Űäÿ Ääÿłäÿ■åŘŇçŽĎCythonåĞ¡æŢřäÿ■èřČçŤĺclip()ïijL'ãĂĆ

çśžåđŃåŔĆæŢř `double[:] a` åŠŇ `double[:] out` åčřæŸŐè£ŹäžŻåŔĆæŢřäÿžäÿĂçżť çŽĎåŔŇçš¿åžęæŢřçżĎãĂĆ
ä¡IJäÿžè¿ŞåĚěïijŇåőČäžñäijŽèő£éŮőäżżä¡Ţåőđç ŐřäžĘåĘĚå■ŸèğĘåŻ¿æŐěåŔčçŽĎæŢřçżĎåŕźèśąïijŇè£ŹäÿŁ
3118æIJL'èŕęçżĘåőŽäźL'ãĂĆ åŇĚæŃňäžĘNumPyäÿ■çŽĎæŢřçżĎåŠŇåĘĚç¡őçŽĎarrayåžŞãĂĆ

å¡Şä¡ăçijŰåEŹçŤ§æĹŘçżŞæđIJäÿžæŢřçżĎçŽĎäżčçăĄæŮűïijŇä¡ääžŤèŕěéĄţå¿łäÿŁéİćçď žä¿ŃéĆčæăůè
åőČäijŽářĘåĹŻåżžè¿ŞåĞžæŢřçżĎçŽĎèť čäżżçżŹèřČçŤĺèĂĚïijŇäÿ■éIJĂèęĄç§ěéĄŞä¡ăæŞ■ä¡IJçŽĎæŢřçżĎç
ïijĹåőČäžĚäżĚåĄĞèő¿æŢřçżĎåůšçżŔåĞĘåď Ğåě¡äžĘïijŇåŔłéIJĂèęĄåĄŽäÿĂäžŽářŘçŽĎæčÄæ§ěæŕŤåęĆçą
åIJĺåČŔNumPyäźŃçśžçŽĎåžŞäÿ■ïijŇä¡£çŤĺ `numpy.zeros()` æĹŰ `numpy.zeros_like()` åĹŽåżžè¿ŞåĞžæŢřçżĎçŻÿåŕźèĂŃèĺĂæŕŤè¿Čåőźæ ŸŞãĂĆåŔęåď ŰïijŃèęĄåĹŽåżžæIJłåĹİ
ä¡ääŔŕäžěä¡£çŤĺ `numpy.empty()` æĹŰ `numpy.empty_like()` .
åęĆæđIJä¡ăæČşèęĘçŽŰæŢřçżĎåĘĚåőźä¡IJäÿžçżŞæđIJçŽĎèŕÍéĂĽ'æŃľè£Źäÿd'äÿłäijŽæŕŤè¿Čå£ńçĆźãĂĆ

ä¡ää¡ăçŽĎåĞ¡æŢřåőđçŐřäÿ■ïijŇä¡ääŔłéIJĂèęĄçőĂå■ŢçŽĎéĂŽè£ĞäÿŃæăĞè£ŘçőŮåŠŇæŢřçżĎæ§ěæ
CythonäijŽèť §èť čäÿžä¡ăçŤ§æĹŘéńŸæŢĹçŽĎäžčçăĄãĂĆ

`clip()` åőŽäźL'äźŃåL'■çŽĎäÿd'äÿłèčĚéěřåŹĺåŔŕäžěäijŸåŇŰäÿŃæĂğèČ¡ãĂĆ
`@cython.boundscheck(False)` çIJĄåŐżäžĘæL'ĂæIJL'çŽĎæŢřçżĎèűŁçŢŇæčÄæ§ěïijŇ
å¡Şä¡ăç§ěéĄŞäÿŃæăĞèő£éŮőäÿ■äijŽèűŁçŢŇçŽĎæŮűåĂŹåŔŕäžěä¡£çŤĺåőČãĂĆ
`@cython.wraparound(False)` æűĹéŹd'äžĘçŽÿåŕźæŢřçżĎår¿éČĺçŽĎèť §æŢřäÿŃæăĞçŽĎåd'ĎçŘĘïijĺ
åijŢåĚèè£Źäÿd'äÿłèčĚéěřåŹĺåŔŕäžěædĄåď ğçŽĎæŔŘå■ĞæĂğèČ¡ïijĹæţŃèŕŢè£Źäÿłä¿Ńå■ŘçŽĎæŮűåĂŹåd'

äžżä¡ŢæŮűåĂŹåď ĎçŘĘæŢřçżĎæŮűïijŇçăŤçl'űåźűæŤźåŰĎåžŢåśĆçőŮæşŢåŘŇæăůåŔŕäžěædĄåď ğçŽ
ä¿ŃåęĆïijŇèĂČèŹŠåŕź `clip()` åĞ¡æŢřçŽĎåęĆäÿŃä£őæ■čïijŇä¡£çŤĺæĮąäžűèąĺè¿¿åijŔïijŽ

```
@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip(double[:] a, double min, double max, double[:] out):
    if min > max:
        raise ValueError("min must be <= max")
    if a.shape[0] != out.shape[0]:
        raise ValueError("input and output arrays must be the same␣
 ↪size")
```

```
    for i in range(a.shape[0]):
        out[i] = (a[i] if a[i] < max else max) if a[i] > min else␣
↪min
```

åőđéŹĚætŃèŕŢçżŞæđIJæŸŕïijŇè£ŹäÿłçĽĹæIJňçŽĎäžčçăĄè£ŘèąŇéĂ§åžęèęĄå£ń50%äžëäÿŁïijĹ2.44çğ
`timeit()` ætŃèŕŢçŽĎ3.76çğŠïijĽãĂĆ

åĹřè£ŹéĞŇäÿžæ■ćïijŇä¡ååŔřèČ¡æČşç§ěéĄŞè£Źçğ■äžčçăĄæĂŐäźĹèČ¡èů§æĽŃåĘŹCèŕ■èÍĂPKåŠćïij§
ä¿ŃåęĆïijŇä¡ååŔřèČ¡åĘŹäžĘåęĆäÿŇçŽĎCåĞ¡æŢřåźűä¡£çŤĺåĽ■éÌćåĞäèŁĆçŽĎæŁĂæIJŕæÌěæĽŃåĘŹæĽĽ

```
void clip(double *a, int n, double min, double max, double *out) {
  double x;
  for (; n >= 0; n--, a++, out++) {
    x = *a;

    *out = x > max ? max : (x < min ? min : x);
  }
}
```

æĹŠäžňæšąæIJL'åśŢçd'žè£ŹäÿłçŽĎæL'l'åśŢäžčçăĄïijŇä¡ĘæŸŕèŕŢétŇäźŃåŘŐïijŇæĹŠäžňåŘŠçŐřäÿĂäĵ
æIJĂåžŢäÿŇçŽĎäÿĂèąŇæŕŤä¡ăæČşèśąçŽĎè£ŘèąŇçŽĎå£ńå¿Ĺåd'ŽãĂĆ

ä¡ååŔřäžěåŕźåőđä¿ŃäžčçăĄæđĎåžžåd'ŽäÿłæĽ'l'åśŢãĂĆ åŕźäžŐæ§ŘäžŻæŢřçżĎæŞ■ä¡IJïijŇæIJĂåě¡èęĄ
èęĄè£ŹæăůåĄŽçŽĎèŕÍïijŇéIJĂèęĄä£őæŤźäžčçăĄïijŇä¡£çŤÍ `with nogil:` èŕ■åŘěïijŽ

```
@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip(double[:] a, double min, double max, double[:] out):
    if min > max:
        raise ValueError("min must be <= max")
    if a.shape[0] != out.shape[0]:
        raise ValueError("input and output arrays must be the same␣
↪size")
    with nogil:
        for i in range(a.shape[0]):
            out[i] = (a[i] if a[i] < max else max) if a[i] > min␣
↪else min
```

åęĆædIJä¡ăæČşåĘŹäÿĂäÿłæŞ■ä¡IJäžŇçžťæŢřçżĎçŽĎçĽĹæIJňïijŇäÿŃéÌćæŸŕåŔŕäžěåŔĆèĂČäÿŃïijŽ

```
@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip2d(double[:,:] a, double min, double max, double[:,:]␣
↪out):
    if min > max:
        raise ValueError("min must be <= max")
    for n in range(a.ndim):
        if a.shape[n] != out.shape[n]:
            raise TypeError("a and out have different shapes")
    for i in range(a.shape[0]):
        for j in range(a.shape[1]):
```

```
            if a[i,j] < min:
                out[i,j] = min
            elif a[i,j] > max:
                out[i,j] = max
            else:
                out[i,j] = a[i,j]
```

åÿŇæIJŻèŕžèĂĚäÿ■èęĄå£ŸäžĘæIJňèŁĆæĽĂæIJĽäžčçăĄéČ¡äÿ■äijŽçżŚåőŽåĹřæ§ŘäÿłçĽźåőŽæŢřçżĎ
è£Źæăůäžčçă Ąåŕśæ Żť æIJĽçĄţæť żæĂğãĂĆ äÿ■è£ĞïijŇèęĄæşĺæĎŔçŽĎæŸŕåęĆæđIJåď ĎçŘĘæŢřçżĎèęĄæ
è£ŹäžŽåĘĚåőźåůšçżŔèűĚåĞžæIJňèŁĆèŇČåŻť ïijŇæŻť ådˇŽä£ąæĄŕèŕůåŔĆèĂČ PEP
3118 ïijŇ åŘŇæŮű CythonæŰĞæąčäÿ■åĚşäžŐâĂIJçśżåđŃåĘĚå■ŸèğĘåŻ¿âĂİ
çŕĞäź§åĂijå¿ŮäÿĂèŕžãĂĆ

15.12 åřĘåĞ¡æŢřæŇĞéŠĹè¡ňæ■ćäÿžåŔŕèřČçŤĺåŕźèśą

éŮőécŸ

ä¡ăåůšçżŔèŐůå¿ŮäžĘäÿĂäÿłèćńçijŰèŕŠåĞ¡æŢřçŽĎåĘĚå■ŸåIJŕåİĂïijŇæČşåřĘåőČè¡ňæ■ćæĹŘäÿĂäÿłP
è£ŹæăůçŽĎèŕİä¡ăåŕśåŔŕäžěåřĘåőČä¡IJäÿžäÿĂäÿłæĽľåśŢåĞ¡æŢřä¡£çŤĺäžĘăĂĆ

èğčåĘşæŰźæąĹ

ctypes æĺąåİŮåŔŕèćńçŤĺæİěåĹŻåžžåŇĚèčĚäżżæĎŔåĘĚå■ŸåIJŕåİĂçŽĎPythonåŔŕèřČçŤĺåŕźèśąăĂĆ
äÿŇéİćçŽĎä¿Ńå■ŘæijŤçď žäžĘæĂŐæăůèŐůåŔŰCåĞ¡æŢřçŽĎåŐ§ågŇãĂĄåžŢåśĆåIJŕåİĂïijŇäžěåŔŁåęĆä¡

```
>>> import ctypes
>>> lib = ctypes.cdll.LoadLibrary(None)
>>> # Get the address of sin() from the C math library
>>> addr = ctypes.cast(lib.sin, ctypes.c_void_p).value
>>> addr
140735505915760

>>> # Turn the address into a callable function
>>> functype = ctypes.CFUNCTYPE(ctypes.c_double, ctypes.c_double)
>>> func = functype(addr)
>>> func
<CFunctionType object at 0x1006816d0>

>>> # Call the resulting function
>>> func(2)
0.9092974268256817
>>> func(0)
0.0
>>>
```

èőĺèőž

èęĄæđĎåżžäÿĂäÿłåŔŕèřČçŤĺåŕźèśąïijŇä¡ăéęŰåĚĹéIJĂèęĄåĹŻåżžäÿĂäÿł
CFUNCTYPE åőđä¿ŃãĂĆ CFUNCTYPE() çŽĎçňňäÿĂäÿłåŔĆæŢřæŸŕè£ŤåŻđçśžåđŃãĂĆ
æŐěäÿŃæİěçŽĎåŔĆæŢřæŸŕåŔĆæŢřçśžåđŃãĂĆäÿĂæŮęä¡ăáőŽäźĽäžĘåĞ¡æŢřçśžåđŃïijŇä¡ăářśèČ¡ářĘáőČ
çŤ§æĹŘçŽĎåŕźèśąèćńå¡ŞåĄŽæŹőéĂŽçŽĎåŔŕéĂŽè£Ğ ctypes
èő£éŮőçŽĎåĞ¡æŢřæİěä¡£çŤĺãĂĆ

æIJňèŁĆçIJŃäÿŁåŐżåŔŕèČ¡æIJĽçĆźçěđçğŸïijŇåĄŔåžŢåśĆäÿĂçĆźãĂĆ
ä¡ĘæŸŕïijŇä¡ĘæŸŕåőČèćńáź£æşŻä¡£çŤĺäžŐåŘĎçğ■énŸçžğäžčçăĄçŤ§æĹŘæŁĂæIJŕæŕŤåęĆå■şæŮűçijŰèŕŚ

ä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿĂäÿłä¡£çŤĺ llvmpy æĽľåśŢçŽĎçőÅå■Ţä¿Ńå■ŘïijŇçŤĺæİěæđĎåżžäÿĂäÿłåřI
åźűåřĘåĚűè¡ñæ■ćäÿžäÿĂäÿłPythonåŔŕèřČçŤĺåŕźèśąãĂĆ

```
>>> from llvm.core import Module, Function, Type, Builder
>>> mod = Module.new('example')
>>> f = Function.new(mod,Type.function(Type.double(), \
                     [Type.double(), Type.double()], False), 'foo')
>>> block = f.append_basic_block('entry')
>>> builder = Builder.new(block)
>>> x2 = builder.fmul(f.args[0],f.args[0])
>>> y2 = builder.fmul(f.args[1],f.args[1])
>>> r = builder.fadd(x2,y2)
>>> builder.ret(r)
<llvm.core.Instruction object at 0x10078e990>
>>> from llvm.ee import ExecutionEngine
>>> engine = ExecutionEngine.new(mod)
>>> ptr = engine.get_pointer_to_function(f)
>>> ptr
4325863440
>>> foo = ctypes.CFUNCTYPE(ctypes.c_double, ctypes.c_double, ctypes.
↪c_double)(ptr)

>>> # Call the resulting function
>>> foo(2,3)
13.0
>>> foo(4,5)
41.0
>>> foo(1,2)
5.0
>>>
```

åźűäÿ■æŸŕèŕť'åIJĺè£ŹäÿłåśĆéİćçŁŕäžĘäżżä¡ŢéŤŹèŕŕåřśäijŽåŕijèĞť'Pythonèğčé ĞŁåŹĺæŇĆæŐL'ãĂĆ
èęĄèőřå¿ŮçŽĎæŸŕä¡ăæŸŕåIJĺçŻť'æŐèèů§æIJžåŹĺçžğåĹńçŽĎåĘĚå■ŸåIJřåİĂåŠŇæIJňåIJřæIJžåŹĺçăĄæL'Şä

15.13 äijăéĂŠNULLçżŞår¿çŽĎå■ŮçňęäÿšçżŹCåĞ¡æŢřåžŞ

éŮőécŸ

ä¡ăèęĄåĘŹäÿĂäÿłæĽ'ĺåśŢæÍąåİŮïijŇéIJĂèęĄäijăéĂŠäÿĂäÿłNULLçżŞår¿çŽĎå■ŮçňęäÿšçżŹCåĞ¡æŢřåäÿ■è£ĞïijŇä¡ăäÿ■æŸŕå¿Ĺçąőåő ŽæĂŐæăůä¡£çŤÍPythonçŽĎUnicodeå■Ůçňęäÿšå Őżåőđç Őřåő ČãĂĆ

èğčåĘşæŰźæąĹ

èőÿåď ŽCåĞ¡æŢřåžŞåŇĚåŘńäÿĂäžŻæŞ■ä¡IJNULLçżŞår¿çŽĎå■ŮçňęäÿšïijŇèćńåčřæŸŐçśżåđŃäÿž `char *`. èĂČèŹŚåęĆäÿŃçŽĎCåĞ¡æŢřïijŇæĹŚäžňçŤÍæİěåĄŽæijŤçď žåŠŇætŃèřŢçŤÍçŽĎïijŽ

```
void print_chars(char *s) {
    while (*s) {
        printf("%2x ", (unsigned char) *s);

        s++;
    }
    printf("\n");
}
```

æ■ď åĞ¡æŢřäijŽæĽŞå■řèćńäijàè£Żæİěĺ■ŮçňęäÿšçŽĎæŕŔäÿłå■ŮçňęçŽĎå■ĄåĚ■è£ŻåĹűèąÍçď žïijŇè£Ź

```
print_chars("Hello");   // Outputs: 48 65 6c 6c 6f
```

åŕźäžŐåIJÍPythonäÿ■èřČçŤÍè£ŹæăůçŽĎCåĞ¡æŢřïijŇä¡ăæIJL'åĞăçğ■éĂĽ'æŃľãĂĆ éęŰåĚĹïijŇä¡ăåŔŕäżěéĂŽè£ĞèřČçŤÍ `PyArg_ParseTuple()` åźűæŇĞåőŽâĂİyâĂIJè¡ňæ■ćçăĄæİěéŹŘåĹűåőČåŔłèČ¡æŞ■ä¡IJå■ŮèŁĆïijŇåęĆäÿŃïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s;

  if (!PyArg_ParseTuple(args, "y", &s)) {
    return NULL;
  }
  print_chars(s);
  Py_RETURN_NONE;
}
```

çżŞæđIJåĞ¡æŢřçŽĎä¡£çŤÍæŰźæşŢåęĆäÿŃãĂĆäžŤçżĘèğĆár§åţŇåĚěäžĘNULLå■ŮèŁĆçŽĎå■Ůçňęäÿš

```
>>> print_chars(b'Hello World')
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>> print_chars(b'Hello\x00World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be bytes without null bytes, not bytes
>>> print_chars('Hello World')
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' does not support the buffer interface
>>>
```

åęĆæđIJä¡ăæČşäijăéĂŠUnicodeå■ŮçňęäÿšïijŇåIJĺ PyArg_ParseTuple() äÿ■ä¡£çŤĺâĂİsâĂIJæäijåijŔçăĄïijŇåęĆäÿŃïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s;

  if (!PyArg_ParseTuple(args, "s", &s)) {
    return NULL;
  }
  print_chars(s);
  Py_RETURN_NONE;
}
```

å¡Şèćńä¡£çŤĺçŽĎæŮűåĂŹïijŇåőČäijŽèĞłåŁĺåřĘæĽĂæIJL'å■Ůçňęäÿšè¡ňæ■ćäÿžäżěNULLçżŞår¿çŽĎU8çijŰçăĄãĂĆä¿ŃåęĆïijŽ

```
>>> print_chars('Hello World')
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>> print_chars('Spicy Jalape\u00f1o')  # Note: UTF-8 encoding
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> print_chars('Hello\x00World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be str without null characters, not str
>>> print_chars(b'Hello World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be str, not bytes
>>>
```

åęĆæđIJåŽääÿžæ§ŘäžŽåŐ§åŽäïijŇä¡ăèęĄçŽt'æŐěä¡£çŤĺ PyObject * èĂŇäÿ■èČ¡ä¡£çŤĺ PyArg_ParseTuple() ïijŇäÿŃéĺćçŽĎä¿Ńå■ŘåŘŚä¡åáśŢçd'žäžĘæĂŐæăůäżŐå■ŮèŁĆåŠŇå■Ůçňęäÿšářźèśąäÿ■æčÄæ§ěåŠŇæŔŘåŔŰäÿchar * åijŢçŤĺïijŽ

```
/* Some Python Object (obtained somehow) */
PyObject *obj;

/* Conversion from bytes */
{
   char *s;
   s = PyBytes_AsString(o);
   if (!s) {
      return NULL;    /* TypeError already raised */
   }
```

```
   print_chars(s);
}

/* Conversion to UTF-8 bytes from a string */
{
   PyObject *bytes;
   char *s;
   if (!PyUnicode_Check(obj)) {
       PyErr_SetString(PyExc_TypeError, "Expected string");
       return NULL;
   }
   bytes = PyUnicode_AsUTF8String(obj);
   s = PyBytes_AsString(bytes);
   print_chars(s);
   Py_DECREF(bytes);
}
```

åĽ■éÍćäÿď˙çğ■è¡ňæ■ćéČ¡åŔŕäžěçąőä£ÍæŸŕNULLçżŞář¿çŽĎæŢřæ■őïijŇ
ä¡ĘæŸŕåőČäžňåźűäÿ■æčĂæ§ěå■Ůçňęäÿšäÿ■éŮt˙æŸŕåŘęåţŇåĚěäžĘNULLå■ŮèŁĆãĂĆ
åŽäæ■ď˙ïijŇåęĆæđIJè£Źäÿłå¿ĹéĞ■èęĄçŽĎèŕÍïijŇéĆčä¡ăéIJĂèęĄèĞłåůśåŐžåĄŽæčĂæ§ěäžĘãĂĆ

èőĺèőž

åęĆæđIJåŔŕèČ¡çŽĎèŕÍïijŇä¡ăåžŤèŕěéĄ£åĚ■åŐżåĘŹäÿĂäžŻä¿ÍèţŰäžŐNULLçżŞář¿çŽĎå■ŮçňęäÿšïijŇ
æIJĂåě¡çżŞåŘĹä¡£çŤĺäÿĂäÿłæŇĞéŠĹåŠŇéŢ£åžęåĂijæÍěåď˙ĎçŘĘå■Ůçňęäÿšã ĂĆ
äÿ■è£ĞïijŇæIJL˙æŮűåĂŹä¡åå£ĚéążåŐżåď˙ĎçŘĘCèŕ■èĺĂéĄŮçŢŹäžčçăĄæŮűåřśæšąå¿ŮéĂL˙æŃl˙äžĘãĂĆ

ař¡çőąå¿Ĺåőźæ˙ŸŞä¡£çŤĺïijŇä¡ĘæŸŕå¿Ĺåőźæ˙ŸŞå£¡èğĘçŽĎäÿĂäÿłéŮőéĆŸæŸŕåIJÍ
`PyArg_ParseTuple()` äÿ■ä¡£çŤĺâĂIJsâĂÍæăijåijŔåŇŰçăĄäijŽæIJL˙åĘĚå■Ÿæ■§èĂŮãĂĆ
ä¡Ęä¡ăéIJĂèęĄä¡£çŤĺè£Źçğ■è¡ňæ■ćçŽĎæŮűåĂŹïijŇäÿĂäÿłUTF-
8å■ŮçňęäÿšèćńåĹŽåžžåźűæřÿäźĚéŹĎåŁååIJĺåŐ§åğŃå■ŮçňęäÿšårźèšąäÿŁéÍćãĂĆ
åęĆæđIJåŐ§åğŃå■Ůçňęäÿšå ŇĚåŘńéÍđASCIIå■ŮçňęçŽĎèŕÍïijŇårśäijŽårijèĞt˙å■ŮçňęäÿšçŽĎåržárÿåćđåĹřäÿ

```
>>> import sys
>>> s = 'Spicy Jalape\u00f1o'
>>> sys.getsizeof(s)
87
>>> print_chars(s)      # Passing string
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> sys.getsizeof(s)    # Notice increased size
103
>>>
```

åęĆæđIJä¡ăåIJÍäźŐè£ŹäÿłåĘĚå■ŸçŽĎæ■§èĂŮïijŇä¡ăæIJĂåě¡éĞ■åĘŹä¡ăçŽĎCæL˙l˙åśŢäžčçăĄïijŇèől˙å
`PyUnicode_AsUTF8String()` åĞ¡æŢřãĂĆåęĆäÿŃïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  PyObject *o, *bytes;
  char *s;
```

```
  if (!PyArg_ParseTuple(args, "U", &o)) {
    return NULL;
  }
  bytes = PyUnicode_AsUTF8String(o);
  s = PyBytes_AsString(bytes);
  print_chars(s);
  Py_DECREF(bytes);
  Py_RETURN_NONE;
}
```

éĂŽè£Ğè£ŹäÿŀäŁőæŤźïijŇäÿĂäÿłUTF-8çijŰçăAçŽĎå■Ůçňęäÿšæăźæ■őéIJĂèęĄèćńåĹŻåžžïijŇçĎűåŘĆ

```
>>> import sys
>>> s = 'Spicy Jalape\u00f1o'
>>> sys.getsizeof(s)
87
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> sys.getsizeof(s)
87
>>>
```

åęĆæđIJä¡ăèŕŢçİĂäijăéĂŠNULLçżŞår¿å■ŮçňęäÿšçżŹctypesåŇĚèčĚè£ĞçŽĎåĞ¡æŢřïijŇ
èęĄæşĺæĎŔçŽĎæŸŕctypesåŘŀèČ¡åĚĄęőÿäijăéĂŠå■ŮèŁĆïijŇåźűäÿŤåőČäÿ■äijŽæčĂæ§ĕäÿ■éŮť åţŇåĚěçŽĎ

```
>>> import ctypes
>>> lib = ctypes.cdll.LoadLibrary("./libsample.so")
>>> print_chars = lib.print_chars
>>> print_chars.argtypes = (ctypes.c_char_p,)
>>> print_chars(b'Hello World')
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>> print_chars(b'Hello\x00World')
48 65 6c 6c 6f
>>> print_chars('Hello World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ctypes.ArgumentError: argument 1: <class 'TypeError'>: wrong type
>>>
```

åęĆæđIJä¡ăæČşäijăéĂŠå■ŮçňęäÿšèĂŇäÿ■æŸŕå■ŮèŁĆïijŇä¡ăéIJĂèęĄåĚĹæĽğèąŇæĽŃåŁĺçŽĎUTF-8çijŰçĂąãĂĆä¿ŃåęĆïijŽ

```
>>> print_chars('Hello World'.encode('utf-8'))
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>>
```

åŕźäžŐåĚűäżŰæĽľåśŢåůěåĚůïijĹæŕŤåęĆSwigãĂĄCythonïijĽïijŇ
åIJĺä¡ää¡£çŤĺåőČäżňäijăéĂŠå■ŮçňęäÿšçżŹCäžčçăĄæŮűèęĄåĚĹåě¡åě¡å■ęäźăçŻÿåžŤçŽĎäÿIJèě£äžĘãĂĆ

15.14 äijăéĂŠUnicodeå■ŮçňęäÿšçżŹCåĞ¡æŢřåžŞ

éŮőécŸ

ä¡ăèęĄåĘŹäÿĂäÿłæĽ'ľåśŢæÍąåİŮïijŇéIJĂèęĄåřĘäÿĂäÿłPythonå■ŮçňęäÿšäijăéĂŠçżŹCçŽĎæ§Řäÿłåžş

èğčåĘşæŰźæąĹ

è£ŹéĞŇæĹŚäžňéIJĂèęĄèĂČèŹŚå¿Ĺåď'ŽçŽĎéŮőécŸïijŇä¡ĘæŸŕæIJĂäÿżèęĄçŽĎéŮőécŸæŸŕçŐřå■Ÿç
åŻăæ■ď'ïijŇä¡ăçŽĎæŇŚæĹŸæŸŕåřĘPythonå■Ůçňęäÿšè¡ňæ■ćäÿžäÿĂäÿłèČ¡èćńCçŘĘèğčçŽĎå¡ćåijŔãĂĆ

äÿžäžĘæijŤçď'žçŽĎçŻőçŽĎïijŇäÿŃéIćæIJĽäÿď'äÿłCåĞ¡æŢřïijŇçŤĺæİěæŞ■ä¡IJå■ŮçňęäÿšæŢřæ■őåźűè
äÿĂäÿłä¡£çŤĺå¡ćåijŔäÿž `char *, int` å¡ćåijŔçŽĎå■ŮèŁĆïijŇ
èĂŇåŘęäÿĂäÿłä¡£çŤĺå¡ćåijŔäÿž `wchar_t *, int` çŽĎåő¡å■Ůçňęå¡ćåijŔïijŽ

```
void print_chars(char *s, int len) {
  int n = 0;

  while (n < len) {
    printf("%2x ", (unsigned char) s[n]);
    n++;
  }
  printf("\n");
}

void print_wchars(wchar_t *s, int len) {
  int n = 0;
  while (n < len) {
    printf("%x ", s[n]);
    n++;
  }
  printf("\n");
}
```

åŕźäžŐéIćåŘŚå■ŮèŁĆçŽĎåĞ¡æŢř `print_chars()` ïijŇä¡ăéIJĂèęĄåřĘPythonå■Ůçňęäÿšè¡ňæ■ćäÿžä
8. äÿŃéIćæŸŕäÿĂäÿłè£ŹæăůçŽĎæĽ'ľåśŢåĞ¡æŢřä¿Ńå■ŘïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s;
  Py_ssize_t  len;

  if (!PyArg_ParseTuple(args, "s#", &s, &len)) {
    return NULL;
  }
  print_chars(s, len);
  Py_RETURN_NONE;
}
```

åŕźäžŐéĆčäžŻéIJĂèęĄåď˙ĎçŘĘæIJžåŹĺæIJňåIJř wchar_t çśżåđŃçŽĎåžŞåĞ¡æŢřïijŇä¡ääŘŕäżěåČŔäÿŃéÍćè£ŹæăůçijŰåĘŹæĽ˙ľåśŢäžčçăĄïijŽ

```
static PyObject *py_print_wchars(PyObject *self, PyObject *args) {
  wchar_t *s;
  Py_ssize_t  len;

  if (!PyArg_ParseTuple(args, "u#", &s, &len)) {
    return NULL;
  }
  print_wchars(s,len);
  Py_RETURN_NONE;
}
```

äÿŃéÍćæŸŕäÿĂäÿłäžd˙äžŠäijŽèŕÍæÍěæijŤçd˙žè£ŹäÿłåĞ¡æŢřæŸŕåęĆä¡Ţåůěä¡IJçŽĎïijŽ

```
>>> s = 'Spicy Jalape\u00f1o'
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> print_wchars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 f1 6f
>>>
```

äžŤçżĘèğĆåŕ§è£ŹäÿłéÍćåŘŚå■ŮèŁĆçŽĎåĞ¡æŢř print_chars() æŸŕæĂŐæăůæŐěåŘŮUTF-8çijŰçăĄæŢřæ■őçŽĎïijŇ äžěåŘŁ print_wchars() æŸŕæĂŐæăůæŐěåŘŮUnicodeçijŰçăĄåĂijçŽĎ

èőÍèőž

åIJÍçżğçż■æIJňèŁĆäźŃåL˙■ïijŇä¡ääžŤèŕěéęŰåĚĹå■ęäźää¡ăèő£éŮőçŽĎCåĞ¡æŢřåžŞçŽĎçL˙źå¿ĄãĂĆ åŕźäžŐå¿Ĺåď˙ŽCåĞ¡æŢřåžŞïijŇéĂŽåÿÿäijăéĂŠå■ŮèŁĆèĂŇäÿ■æŸŕå■ŮçňęäÿšäijŽæŕŤè¿Čåě¡äžŽãĂĆèęĄè£

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s;
  Py_ssize_t  len;

  /* accepts bytes, bytearray, or other byte-like object */
  if (!PyArg_ParseTuple(args, "y#", &s, &len)) {
    return NULL;
  }
  print_chars(s, len);
  Py_RETURN_NONE;
}
```

åęĆæđIJä¡ääż■çĎűè£ŸæŸŕæČşèęĄäijăéĂŠå■ŮçňęäÿšïijŇ ä¡ăéIJĂèęĄç§ěéĄŞPython 3åŘŕä¡£çŤĺäÿĂäÿłåŘĹéĂĆçŽĎå■ŮçňęäÿšèąÍçd˙žïijŇ åőČåźűäÿ■çŻt˙æŐěæŸăărĎåĹřä¡£çŤĺæăĞåĞĘçśžåđŃ char * æĹŰ wchar_t * ïijĹæŻt˙åď˙ŽçżĘèŁĆåŔĆèĂČPEP 393ïijL˙çŽĎCåĞ¡æŢřåžŞãĂĆ åŻăæ■d˙ïijŇèęĄåIJÍCäÿ■èąÍçd˙žè£Źäÿłå■ŮçňęäÿšæŢřæ■őïijŇäÿĂäžŻè¡ňæ■ćè£ŸæŸŕå£ĚéążèęĄç ŽĎãĂĆ åIJÍ PyArg_ParseTuple() äÿ■ä¡£çŤĺ"s#" åŠŇ"u#"æăijåijŔåŇŰçĂĄåŔŕäžěåőL˙åĚĺçŽĎæL˙ğèąŇè£ŹæăůçĄ

äÿ■è£Ğè£Źçğ■è¡ňæ■ćæIJĽäÿłçijžçĆźåřśæŸŕåőČåŔŕèČ¡äijŽåŕijèĞťåŐ§åğŃå■ŮçňęäÿšåŕźèśąçŽĎåřžåŕÿ
äÿĂæŮęè¡ňæ■ćè£ĞåŘŐïijÑäijŽæIJĽäÿĂäÿłè¡ňæ■ćæŢřæ■őçŽĎåd'■åĹűéŹĎåŁääĹřåŐ§åğŃå■Ůçňęäÿšåŕźèś
ä¡ääŔŕäžěèğĆår§äÿŃè£Źçğ■æŢĹædIJïijŽ

```
>>> import sys
>>> s = 'Spicy Jalape\u00f1o'
>>> sys.getsizeof(s)
87
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> sys.getsizeof(s)
103
>>> print_wchars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 f1 6f
>>> sys.getsizeof(s)
163
>>>
```

åŕźäžŐåřŠéĞŔçŽĎå■ŮçňęäÿšåŕźèśąïijŇåŔŕèČ¡æšąäżĂäźĹå¡śåŞ■ïijŇ
ä¡ĘæŸŕåęĆædIJä¡ăéIJĂèęĄåIJÍæĽ'ľåśŢäÿ■åd'ĎçŘĘåd'ğéĞŔçŽĎæŰĞæIJñïijŇä¡ääŔŕèČ¡æČşéĄ£åĚ■è£Źäÿł
äÿŃéÍćæŸŕäÿĂäÿłä£őèőćçĽĹæIJñåŔŕäžěéĄ£åĚ■è£Źçğ■åĘĚå■Ÿæ■§èĂŮïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  PyObject *obj, *bytes;
  char *s;
  Py_ssize_t   len;

  if (!PyArg_ParseTuple(args, "U", &obj)) {
    return NULL;
  }
  bytes = PyUnicode_AsUTF8String(obj);
  PyBytes_AsStringAndSize(bytes, &s, &len);
  print_chars(s, len);
  Py_DECREF(bytes);
  Py_RETURN_NONE;
}
```

èĂŇåŕź `wchar_t` çŽĎåd'ĎçŘĘæŮűæČşèęĄéĄ£åĚ■åĘĚå■Ÿæ■§èĂŮåŕśæŻť'åŁäéŽ¿åŁđäžĘãĂĆ
åIJĺåĘĚéČĺïijŇPythonä¡£çŤĺæIJĂéńŸæŢĹçŽĎèąĺçd'žæİěå■ŸåĆĺå■Ůçňęäÿšã ĂĆ
ä¿ŃåęĆïijŇåŔłåŇĚåŘńASCIIçŽĎå■Ůçňęäÿšèćńå■ŸåĆĺäÿžå■ŮèŁĆæŢřçżĎïijŇ
èĂŇåŇĚåŘńèŇČåŻť'äžŐU+0000åĹřU+FFFFçŽĎå■ŮçňęçŽĎå■Ůçňęäÿšä¡£çŤĺåŔŇå■ŮèŁĆèąĺçd'žãĂĆ
çŤśäžŐåŕźäžŐæŢřæ■őçŽĎèąĺçd'žå¡ćåijŔäÿ■æŸŕå■ŢäÿĂçŽĎïijŇä¡ääÿ■èČ¡årĘåĘĚéČĺæŢřçżĎè¡ňæ■ćäÿž
`wchar_t *` çĎűåŘŐæIJ§æIJŻåőČèČ¡æ■čçąőçŽĎåůěä¡IJãĂĆ ä¡ääžŤèŕěåĹŻåžžäÿĂäÿł
`wchar_t` æŢřçżĎåźűåŘŚåĚűäÿ■åd'■åĹűæŰĞæIJñãĂĆ `PyArg_ParseTuple()`
çŽĎ"u#"æäijåijŔçăĄåŔŕäžěåÿőåŁľä¡äéńŸæŢĹçŽĎåőŇæĹŘåőČïijĹåőČårĘåd'■åĹűçżŞædIJéŹĎåŁääĹřå■Ůç

åęĆædIJä¡ăæČşéĄ£åĚ■éŢ£æŮűéŮť'åĘĚå■Ÿæ■§èĂŮïijŇä¡ääŤŕäÿĂçŽĎéĂĽ'æŃľ'ářśæŸŕåd'■åĹűUnicoc
ářĘåőČäijăéĂŠçżŹCåĞ¡æŢřïijŇçĎűåŘŐåŻđæŤűè£ŹäÿłæŢřçżĎçŽĎåĘĚå■ŸãĂĆäÿŃéÍćæŸŕäÿĂäÿłåŔŕèČ¡çŽ

```
static PyObject *py_print_wchars(PyObject *self, PyObject *args) {
  PyObject *obj;
```

```
  wchar_t *s;
  Py_ssize_t len;

  if (!PyArg_ParseTuple(args, "U", &obj)) {
    return NULL;
  }
  if ((s = PyUnicode_AsWideCharString(obj, &len)) == NULL) {
    return NULL;
  }
  print_wchars(s, len);
  PyMem_Free(s);
  Py_RETURN_NONE;
}
```

åIJÍè£ŹäÿłåőđçŐřäÿ■ïijŇPyUnicode_AsWideCharString()
åĹŻåžžäÿĂäÿłäÿťæŮűçŽĎwchar_tçijŞåĘšåźűåď■åĹűæŢřæ■őè£ŻåŐżãĂĆ
è£ŹäÿłçijŞåĘšècńäijăéĂŠçżŹCçĎűåŘŐècńéĞŁæŤ¿æŐĽ˜ãĂĆ ä¡ĘæŸŕæĹŚåĘŹè£ŹæIJňäźęçŽĎæŮűåĂŹïijŇè

åęĆæđIJä¡ăç§ěéĄŞCåĞ¡æŢřåžŞéIJĂèęĄçŽĎå■ŮèŁĆçijŰçăĄåźűäÿ■æŸŕUTF-8ïijŇ
ä¡ăåŔŕäžěåijžåĹűPythonä¡£çŤĺæĽ˙ľåśŢçăĄæİěæĽ˙ğèąŇæ■čçąőçŽĎè¡ňæ■ćïijŇåřšåČŔäÿŇéİćè£Źæăůïijž

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s = 0;
  int   len;
  if (!PyArg_ParseTuple(args, "es#", "encoding-name", &s, &len)) {
    return NULL;
  }
  print_chars(s, len);
  PyMem_Free(s);
  Py_RETURN_NONE;
}
```

æIJĂåŘŐïijŇåęĆæđIJä¡ăæČşçŻť˙æŐěåď˙ĎçŘĘUnicodeå■ŮçňęäÿšïijŇäÿŇéİćçŽĎæŸŕä¿Ńå■ŘïijŇæijŢç

```
static PyObject *py_print_wchars(PyObject *self, PyObject *args) {
  PyObject *obj;
  int n, len;
  int kind;
  void *data;

  if (!PyArg_ParseTuple(args, "U", &obj)) {
    return NULL;
  }
  if (PyUnicode_READY(obj) < 0) {
    return NULL;
  }

  len = PyUnicode_GET_LENGTH(obj);
  kind = PyUnicode_KIND(obj);
  data = PyUnicode_DATA(obj);
```

```
  for (n = 0; n < len; n++) {
    Py_UCS4 ch = PyUnicode_READ(kind, data, n);
    printf("%x ", ch);
  }
  printf("\n");
  Py_RETURN_NONE;
}
```

åIJÍè£Źäÿłäżčçă Ąäÿ■ïijŇPyUnicode_KIND() åŠŇ PyUnicode_DATA() è£Źäÿd'äÿłåőŘåŠŇUnicodeçŽĎåŘŕåŘŸåő¡åžęå■ŸåĆÍæIJL'åĚşïijŇè£ŹäÿłåIJÍPEP 393äÿ■æIJL'æŘŔè£řãĂĆ kind åŘŸéĞŘçijŰçăĄåžŢåśĆå■ŸåĆÍïijĹ8ä¡■ãĂĄ16ä¡■æĹŰ32ä¡■ïijL'äžčåŘŁæŇ åIJÍåőđéŹĚæČĚåĘţäÿ■ïijŇä¡ăáźűäÿ■éIJĂèęĄç§ěéĄŞäżżä¡Ţèů§è£ŹäžŻåĂijæIJL'åĚşçŽĎäÿIJèě£ïijŇ åŘłéIJĂèęĄåIJÍæŘŘåŘŰå■ŮçňęçŽĎæŮűåĂŹårĘåőČäžňäijăçżŹ PyUnicode_READ() åőŔãĂĆ

è£ŸæIJL'æIJĂåŘŐåĞäåŔěïijŽå¡ŞäžŐPythonäijăéĂŠUnicodeå■ŮçňęäÿšçżŹCçŽĎæŮűåĂŹïijŇä¡ăážŤèŕè åęĆædIJæIJL'UTF-8åŠŇåő¡å■Ůçňęäÿd'çğ■éĂL'æŃl'ïijŇèŕůéĂL'æŃl'UTF-8. åŕźUTF-8çŽĎæŤŕæŇĄæŻt'åŁäæŹőéĄ■äÿĂäžŻïijŇäźŧ§äÿ■åőźæŸŞçŁŕéŤŹïijŇèğčéĞŁåŹĺäź§èČ¡æŤŕæŇĄçŽĎæŻt'åě æIJĂåŘŐïijŇçąőä£Ła¡ăáżŤçżĘéŸĚèŕżäžĘ åĚşäžŐåd'ĎçŘĘUnicodeçŽĎçŻÿåĚşæŰĞæąč

15.15 Cå■Ůçňęäÿšè¡ňæ■ćäÿžPythonå■Ůçňęäÿš

éŮőécŸ

æĂŐæăůåřĘCäÿ■çŽĎå■Ůçňęäÿšè¡ňæ■ćäÿžPythonå■ŮèŁĆæĹŰäÿĂäÿłå■Ůçňęäÿšåŕźèśąïij§

èğčåĘşæŰźæąĹ

Cå■Ůçňęäÿšä¡£çŤĺäÿĂåŕź char * åŠŇ int æİěèąÍçd'žïijŇ ä¡ăéIJĂèęĄåĘşáőŽå■ŮçňęäÿšåĹŕåžŢæŸŕçŤĺäÿĂäÿłåŐ§ăgŃå■ŮèŁĆå■Ůçňęäÿšè£ŸæŸŕäÿĂäÿłUnicodeå■Ůç å■ŮèŁĆåŕźèśąåŘŕäžěåČŘäÿŃéİćè£ŹæăůäĄ£çŤĺ Py_BuildValue() æİěædĎåžžïijŽ

```
char *s;      /* Pointer to C string data */
int   len;    /* Length of data */

/* Make a bytes object */
PyObject *obj = Py_BuildValue("y#", s, len);
```

åęĆædIJä¡ăèęĄåĹŽåžžäÿĂäÿłUnicodeå■ŮçňęäÿšïijŇáźűäÿŤä¡ăç§ěéĄŞ s æŇĞåŘŚäžĘUTF-8çijŰçăĄçŽĎæŢŕæ■őïijŇåŘŕäžěä¡£çŤĺäÿŃéİćçŽĎæŰźåijŔïijŽ

```
PyObject *obj = Py_BuildValue("s#", s, len);
```

åęĆædIJ s ä¡£çŤĺåĚűäżŰçijŰçăĄæŰźåijŔïijŇéĆčäźĹåŘŕäžěåČŘäÿŃéİćä¡£çŤĺ PyUnicode_Decode() æİěædĎåžžäÿĂäÿłå■ŮçňęäÿšïijŽ

```
PyObject *obj = PyUnicode_Decode(s, len, "encoding", "errors");

/* Examples */
obj = PyUnicode_Decode(s, len, "latin-1", "strict");
obj = PyUnicode_Decode(s, len, "ascii", "ignore");
```

åęĆæđIJä¡ăæĄřåě¡æIJĽäÿĂäÿłçŤĺ `wchar_t *, len` åŕźèąĺçđ'žçŽĎåő¡å■ŮçňęäÿšïijŇæIJĽåĞăçğ■éĂĽæŃľæĂğãĂĆéęŰåĚĹä¡ăåŔŕäžěä¡£çŤĺ `Py_BuildValue()` ïijŽ

```
wchar_t *w;     /* Wide character string */
int len;        /* Length */

PyObject *obj = Py_BuildValue("u#", w, len);
```

åŔęåđ'ŰïijŇä¡ăè£ŸåŔŕäžěä¡£çŤĺ `PyUnicode_FromWideChar()`:

```
PyObject *obj = PyUnicode_FromWideChar(w, len);
```

åŕźäžŐåő¡å■ŮçňęäÿšïijŇåźűæšąæIJĽåŕźå■Ůçňęæţřæ■őè£ŻèąŇèğčæđŘâĂŤâĂŤåőČèćńåĄĞåőŽæŸŕåŐ§

èőĺèőž

åřĘCäÿ■çŽĎå■Ůçňęäÿšè¡ňæ■ćäÿžPythonå■Ůçňęäÿšéąţå¿łåŠŇI/OåŘŇæăůçŽĎåŐ§åĹŹãĂĆäź§åřśæŸŕèřťïijŇæİèèĞłCäÿ■çŽĎæţřæ■őå£Ěéążæăźæ■őäÿĂäžŻèğčçăĄåŹĺèćńæŸ¿åijŔçŽĎèğčçăĄäÿžäÿĂéĂŽåÿÿçijŰçăĄæăijåijŔåŇĚæŃňASCIIãĂĄLatin-1åŠŇUTF-8.
åęĆæđIJä¡ååźűäÿ■çąőåőŽçijŰçăĄæŰźåijŔæĹŰèĂĚæţřæ■őæŸŕäžŇè£ŻåĹűçŽĎïijŇä¡ăæIJĂåě¡åřĘå■Ůçňęäÿ
å¡ŞædĎéĂăäÿĂäÿłåŕźèśąçŽĎæŮűåĂŹïijŇPythonéĂŽåÿÿäijŽåđ'■åĹűä¡ăæŔŘä¿ŻçŽĎå■Ůçňęäÿšæţřæ■õãĂĆ
åęĆæđIJæIJĽå£ĚèęĄçŽĎèŕÍïijŇä¡ăéIJĂèęĄåIJĺåŘŐéİćåŐžéĞŁæŤ¿Cå■Ůçňęäÿšã ĂĆ
åŘŇæŮűïijŇäÿžäžĘèől'çĺŃåžŔæŻt'åŁăåĄěåčőïijŇä¡ăåžŤèŕěåŘŇæŮűä¡£çŤĺäÿĂäÿłæŇĞéŠĹåŠŇäÿĂäÿłåd'ğ
èĂŇäÿ■æŸŕä¿ÌètŰNULLçžŞář¿æţřæ■őæİěåĹŻåžžå■Ůçňęäÿšã ĂĆ

15.16 äÿ■çąőåőŽçijŰçăĄæăijåijŔçŽĎCå■Ůçňęäÿš

éŮőécŸ

ä¡ăèęĄåIJĺCåŠŇPythonçŻt'æŐěæİěåŻđè¡ňæ■ćå■ŮçňęäÿšïijŇä¡ĘæŸŕCäÿ■çŽĎçijŰçăĄæăijåijŔåźűäÿ■ç
ä¿ŃåęĆïijŇåŔŕèČ¡Cäÿ■çŽĎæţřæ■őæIJ§æIJŻæŸŕUTF-8ïijŇä¡ĘæŸŕåźűæšąæIJĽåijžåĹűåőČå£ĚéążæŸŕãĂĆ
ä¡ăæČşçijŰåĘŹäžčçăĄæİěäžěäÿĂçğ■äijŸéŻĚçŽĎæŰźåijŔåđ'ĎçŘĘè£ŹäžŻäÿ■åŘĹæăijæţřæ■õïijŇè£ŹæăůåĄ

èğčåĘşæŰźæąĹ

äÿŃéİćæŸŕäÿĂäžŻCçŽĎæţřæ■őåŠŇäÿĂäÿłåĞ¡æţřæİěæijŤçđ'žè£ŹäÿłéŮőécŸïijŽ

```
/* Some dubious string data (malformed UTF-8) */
const char *sdata = "Spicy Jalape\xc3\xb1o\xae";
int slen = 16;

/* Output character data */
void print_chars(char *s, int len) {
  int n = 0;
  while (n < len) {
    printf("%2x ", (unsigned char) s[n]);
    n++;
  }
  printf("\n");
}
```

åIJÍè£ŹäÿłäžčçăĄäÿ■ïijŇå■Ůçňęäÿš sdata åŇĚåŘńäžĘUTF-8åŠŇäÿ■åŘĹæăijæŢřæ■őãĂĆ äÿ■è£ĞïijŇåęĆæđIJçŤĺæĹůåIJÍCäÿ■èřČçŤÍ print_chars(sdata, slen) ïijŇåőČçijžèČ¡æ■čåÿÿåůëä¡IJãĂĆ çŐřåIJÍåĄĞèő¿ä¡ăæČşåřĘ sdata çŽĎåĘĚåőźè¡ňæ■ćäÿžäÿĂäÿłPythonå■Ůçňęäÿšã ĂĆ è£ŻäÿĂæ■ěåĄĞèő¿ä¡ăåIJÍåŘŐéÍćè£ŸæČşéĂŽè£ĞäÿĂäÿłæĽľåśŢåřĘéĆčäÿłå■Ůçňęäÿšäijăäÿł print_chars() åĞ¡æŢřãĂĆ äÿŃéÍćæŸŕäÿĂçğ■çŤĺæÍëä£ÍæŁd'åŐ§åğŃæŢřæ■őçŽĎæŰźæşŢïijŇåřśçőŮå

```
/* Return the C string back to Python */
static PyObject *py_retstr(PyObject *self, PyObject *args) {
  if (!PyArg_ParseTuple(args, "")) {
    return NULL;
  }
  return PyUnicode_Decode(sdata, slen, "utf-8", "surrogateescape");
}

/* Wrapper for the print_chars() function */
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  PyObject *obj, *bytes;
  char *s = 0;
  Py_ssize_t   len;

  if (!PyArg_ParseTuple(args, "U", &obj)) {
    return NULL;
  }

  if ((bytes = PyUnicode_AsEncodedString(obj,"utf-8",
↪"surrogateescape"))
        == NULL) {
    return NULL;
  }
  PyBytes_AsStringAndSize(bytes, &s, &len);
  print_chars(s, len);
  Py_DECREF(bytes);
  Py_RETURN_NONE;
}
```

åęĆæđIJä¡ăåIJĺPythonäÿ■åřÌèřŢè£ŹäžŻåĞ¡æŢřïijŇäÿŃéÌćæŸŕè£ŘèąŇæŢĹæđIJïijŽ

```
>>> s = retstr()
>>> s
'Spicy JalapeÃśo\udcae'
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f ae
>>>
```

äžŤçżĘèğĆåŕ§çżŞæđIJä¡ăäijŽåŘŠçŐřïijŇäÿ■åŘĹæăijå■ŮçňęäÿšèćńçijŰçăĄåĹřäÿĂäÿłPythonå■Ůçňęäÿ
åźűäÿŤå¡ŞåőČèćńåŽđäijăçżŹCçŽĎæŮűåĂŹïijŇèćńè¡ñæ■ćäÿžåŠŇäźŃåĽ■åŐ§ågŃCå■ŮçňęäÿšäÿĂæăůçŽĎå

èőĺèőž

æIJňèŁĆåśŢçđ'žäžĘåIJĺæĽl'åśŢæĺąåİŮäÿ■åđ'ĎçŘĘå■ŮçňęäÿšæŮűäijŽéĚ■åĹřçŽĎäÿĂäÿłæčŸæĽ'ŃåŘĺ
äźş§åřśæŸŕèŕt'ïijŇåIJĺæĽl'åśŢäÿ■çŽĎCå■ŮçňęäÿšåŕŕèČ¡äÿ■äijŽäÿěæäijéĄţå¿łPythonæĽ'ĂæIJ§æIJŻçŽĎUn
åŻăæ■đ'ïijŇå¿ĹåŘŕèČ¡äÿĂäžŽäÿ■åŘĹæăijCæŢřæ■őäijăéĂŠåĹřPythonäÿ■åŐžãĂĆ
äÿĂäÿłå¿Ĺåě¡çŽĎä¿Ńå■ŘåřśæŸŕæűĽ'åŘŁåĹřåžŢåśĆçşżçż§èřČçŤĺæŕŤåęĆæŰĞäžűåŘ■è£ŹæăůçŽĎå■Ůçňęäÿ
ä¿ŃåęĆïijŇåęĆæđIJäÿĂäÿłçşżçż§èřČçŤĺè£ŤåŻđçżŹèğčéĞŁåŹĺäÿĂäÿłæ■§åİŕçŽĎå■ŮçňęäÿšïijŇäÿ■èČ¡èćńa

äÿĂèĹñæİěèőšïijŇåŕŕäžěéĂŽè£ĞåĹűåőŽäÿĂäžŻéŤŹèŕŕç■ŰçŢěæŕŤåęĆäÿěæäijãĂĄå£¡çŢěãĂĄæŻ£äżč
äÿ■è£ĞïijŇè£ŹäžŻç■ŰçŢěçŽĎäÿĂäÿłçijžçĆźæŸŕåőČäžñæřÿäźĚæĂğçăť'åİŕäžĘåŐ§ågŃå■Ůçňęäÿšç
ä¿ŃåęĆïijŇåęĆæđIJä¿Ńå■Řäÿ■çŽĎäÿ■åŘĹæăijæŢřæ■őä¡£çŤĺè£ŹäžŻç■ŰçŢěäźŃäÿĂèğčçăĄïijŇä¡äăijŽå¿Ů

```
>>> raw = b'Spicy Jalape\xc3\xb1o\xae'
>>> raw.decode('utf-8','ignore')
'Spicy JalapeÃśo'
>>> raw.decode('utf-8','replace')
'Spicy JalapeÃśo?'
>>>
```

surrogateescape éŤŹèŕŕåđ'ĎçŘĘç■ŰçŢěäijŽåřĘæĽ'ĂæIJL'äÿ■åŕŕèğčçăĄå■ŮèŁĆè¡ńåŇŰäÿžäÿĂä
ä¿ŃåęĆïijŽ

```
>>> raw.decode('utf-8','surrogateescape')
'Spicy JalapeÃśo\udcae'
>>>
```

å■ŢçŃňçŽĎä¡Őä¡■äžčçŘĘå■Ůçňęæŕ̈ŤåęĆ \udcae åIJĺUnicodeäÿ■æŸŕéİđæşŢçŽĎãĂĆ åŻăæ■đ'ïijŇè£ŹäÿłåŮçňęäÿšåřśæŸŕäÿĂäÿłéİđæşŢèąĺçđ'žãĂĆ
åőđéŹĚäÿŁïijŇåęĆæđIJä¡ăåřĘåőČäijăäÿłäÿĂäÿłæĽ'ğèąŇè¿ŞåĞžçŽĎåĞ¡æŢřïijŇä¡ăäijŽå¿ŮåĹřäÿĂäÿłéŤŹèŕ

```
>>> s = raw.decode('utf-8', 'surrogateescape')
>>> print(s)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
UnicodeEncodeError: 'utf-8' codec can't encode character '\udcae'
in position 14: surrogates not allowed
>>>
```

çĎűèĂŇïijŇåĔĄèőÿäżčçŘĘè¡ňæ■ćçŽĎåĖşéŤőçĆźåĲĺäžŐäżŐCäijăçżŹPythonåŔĹåŽđäijăçżŹCçŽĎäÿ■
å¡Şè£Źäÿłå■Ůçňęäÿšå Ę■æňąä¡£çŤĺ surrogateescape çijŰçăĄæŮűïijŇäżčçŘĘå■Ůçňęäij Žè¡ňæ■ćåŽđåŐ

```
>>> s
'Spicy JalapeÃśo\udcae'
>>> s.encode('utf-8','surrogateescape')
b'Spicy Jalape\xc3\xb1o\xae'
>>>
```

ä¡ĲäÿžäÿĂèĹňåĞĘåĹŹïijŇæĲĂåě¡éĄ£åĔ■äżčçŘĘçijŰçăĄâĂŤâĂŤåęĆæđĲä¡ăæ■ćçąőçŽĎä¡£çŤĺäžĘçij
äÿ■è£ĞïijŇæĲĽæŮűåĂŹçąőåőđäijŽåĞžçŐřä¡ăåźűäÿ■èČ¡æŐğåĹűæŢřæ■őçijŰçăĄåźűäÿŤä¡ăåŔĹäÿ■èČ¡å£¡
éĆčäźĹåřśåŔŕäžĕä¡£çŤĺæĲňèŁĆçŽĎæŁĂæĲŕäžĘãĂĆ

æĲĂåŘŐäÿĂçĆźèçĄæşĺæĎŔçŽĎæŸŕïijŇPythonäÿ■èőÿåď ŽéİćåŘŚçşżçż§çŽĎåĞ¡æŢřïijŇçĽźåĹńæŸŕå
éČ¡äijŽä¡£çŤĺäżčçŘĘçijŰçăĄãĂĆä¿Ńåęĺ ïijŇåęĆæđĲä¡ää¡£çŤĺåČŔ os.listdir()
è£Źæăůçŏ ĎåĞ¡æŢřïijŇ äijååĔĕäÿĂäÿłåŇĔåŘńäžĘäÿ■åŔŕèğčçăĄæŰĞäżűåŘ■çŽĎçŻőå¡ŢçŽĎèŕĲïijŇåőČäijŽ
åŔĆèĂČ5.15çŽĎçŻÿåĔşçńăèŁĆãĂĆ

PEP 383 äÿ■æĲĽæŻťåď ŽåĔşäžŐæĲňæĲžæŔŘåĹřçŽĎäżĕåŔŁåŠŃsurrogateescapeéŤŹèŕŕåď ĎçŘĘçŻÿåĔşçŽĎä£ąæĄŕãĂĆ

15.17 äijăéĂŠæŰĞäżűåŘ■çżŹCæĽľåśŢ

éŮőéćŸ

ä¡ăéĲĂèęĄåŘŚCåžŞåĞ¡æŢřäijăéĂŠæŰĞäżűåŘ■ïijŇä¡ĘæŸŕéĲĂèęĄçąőä£ĬæŰĞäżűåŘ■æăźæ■őçşżçż§å

èğčåĘşæŰźæąĹ

åĘŹäÿĂäÿłæŐěåŔŮäÿĂäÿłæŰĞäżűåŘ■äÿžåŔĆæŢřçŽĎæĽľåśŢåĞ¡æŢřïijŇåęĆäÿŃè£Źæăůïij Ž

```
static PyObject *py_get_filename(PyObject *self, PyObject *args) {
  PyObject *bytes;
  char *filename;
  Py_ssize_t len;
  if (!PyArg_ParseTuple(args,"O&", PyUnicode_FSConverter, &bytes)) {
    return NULL;
  }
  PyBytes_AsStringAndSize(bytes, &filename, &len);
  /* Use filename */
  ...

  /* Cleanup and return */
  Py_DECREF(bytes)
  Py_RETURN_NONE;
}
```

åęĆæđIJä¡ăåůšçżŔæIJĽäžĘäÿĂäÿł PyObject * ïijŇåÿŇæIJŻåřĘåĚűè¡ñæ■ćæĹŘäÿĂäÿłæŰĞäžűåŘ■ï

```
PyObject *obj;    /* Object with the filename */
PyObject *bytes;
char *filename;
Py_ssize_t len;

bytes = PyUnicode_EncodeFSDefault(obj);
PyBytes_AsStringAndSize(bytes, &filename, &len);
/* Use filename */
...

/* Cleanup */
Py_DECREF(bytes);

If you need to return a filename back to Python, use the following␣
↪code:

/* Turn a filename into a Python object */

char *filename;       /* Already set */
int   filename_len;   /* Already set */

PyObject *obj = PyUnicode_DecodeFSDefaultAndSize(filename, filename_
↪len);
```

èőÍèőž

äžěåŘŕçğżæď■æŰźåijŔæÌěåď ĎçŘĘæŰĞäžűåŘ■æŸŕäÿĂäÿłå¿ĹæčŸæĽŃçŽĎéŮőécŸïijŇæIJĂåŘŐäžď
åęĆæđIJä¡ăåIJÍæĽľåśŢäžčçăĄäÿ■ä¡£çŤÍæIJňèŁĆçŽĎæŁĂæIJŕïijŇæŰĞäžűåŘ■çŽĎåď ĎçŘĘæŰźåijŔåŠŇåŠ
åŇĚæŃňçijŰçăA/çŢŇéÍćå■ŮèŁĆïijŇåď ĎçŘĘåÍŘå■Ůçňę̈ïijŇäžčçŘĘè¡ñæ■ćåŠŇåĚűäžŰåď ■æÌĆæČĚåĘţãĂ

15.18 äijäéĂŠåůšæĽŞåijĂçŽĎæŰĞäžűçżŹCæĽľåśŢ

éŮőécŸ

ä¡ăåIJÍPythonäÿ■æIJĽäÿĂäÿłæĽŞåijĂçŽĎæŰĞäžűåŕźèśąïijŇä¡ĘæŸŕéIJĂèęĄåřĘåőČäijăçżŹèęĄä¡£çŤÍ

èğčåĘşæŰźæąĹ

èęĄåřĘäÿĂäÿłæŰĞäžűè¡ñæ■ćäÿžäÿĂäÿłæŢť åđŃçŽĎæŰĞäžűæŔŔè£řçňę̈ïijŇä¡£çŤÍ PyFile_FromFd() ïijŇåęĆäÿŃïijŽ

```
PyObject *fobj;     /* File object (already obtained somehow) */
int fd = PyObject_AsFileDescriptor(fobj);
```

```
if (fd < 0) {
   return NULL;
}
```

çżŞæđIJæŰĞäžűæŔŕè£řçňęæŸŕéĂŽè£ĞèřČçŤĺ fobj äÿ■çŽĎ fileno() æŰźæşŢèŐůå¿ŮçŽĎãĂĆ åŻăæ■đ'ïijŇäžżä¡Ţäžěè£Źçğ■æŰźåijŔæŽť'éIJšçżŹäÿĂäÿłæŔŕè£řåŹĺçŽĎåŕźèšąéČ äÿĂæŮęä¡ăæIJL'äžĘè£ŹäÿłæŔŕè£řåŹĺïijŇåőČåŕšèČ¡èćńäijăéĂŠçżŹåđ'Žäÿłä¡ŐçžğçŽĎåŔŕåđ'ĎçŘĘæŰĞäžű

åęĆæđIJä¡ăéIJĂèęĄè¡ňæ■ćäÿĂäÿłæŢť'åđŇæŰĞäžűæŔŕè£řçňęäÿžäÿĂäÿłPythonåŕźèšąïijŇéĂĆçŤĺäÿŇé PyFile_FromFd() :

```
int fd;      /* Existing file descriptor (already open) */
PyObject *fobj = PyFile_FromFd(fd, "filename","r",-1,NULL,NULL,NULL,
↪1);
```

PyFile_FromFd() çŽĎåŔĆæŢřåŕźäžŤåĘĚç¡őçŽĎ open() åĞ¡æŢřãĂĆ NULL-LèąÍçđ'žçijŰçăĄãĂĄéŤŹèŕŕåŠŇæ■ćèąŇåŔĆæŢřä¡£çŤĺéżŸèőđ'åĂijãĂĆ

èőÍèőž

åęĆæđIJåřĘPythonäÿ■çŽĎæŰĞäžűåŕźèšąäijăçżŹCïijŇæIJL'äÿĂäžŻæşÍæĎŔäžŇéąźãĂĆ éęŰåĚĹïijŇPythonéĂŽè£Ğ io æÍąåİŮæL'ğèąŇèĞłåůšçŽĎI/OçijŞåĘšãĂĆ åIJĺäijăéĂŠäžżä¡Ţçšżåđ̊ŇçŽĎæŰĞäžűæŔŕè£řçňęçżŹCäźŇåL'■ïijŇä¡ăéČ¡èęĄéęŰåĚĹåIJÍçŻÿäžŤæŰĞäžűåŕź äÿ■çĎűçŽĎèŕİïijŇä¡ăäijŽæL'Şäźšæ ŰĞäžűçşżçż§äÿŁéÍćçŽĎæŢřæ■őãĂĆ

åĚűæňąïijŇä¡ăéIJĂèęĄçL'źåĹńæşÍæĎŔæŰĞäžűçŽĎå¡Šåšđè ĂĚäžěåŔŁåĚşéŮ■æŰĞäžűçŽĎèĄŇèť'čãĂĆ åęĆæđIJäÿĂäÿłæŰĞäžűæŔŕè£řçňęèćńäijăçżŹCïijŇä¡ĘæŸŕåIJÍPythonäÿ■è£ŸåIJÍèćńä¡£çŤĺçİĂïijŇä¡ăéIJĂèę çšżäijijçŽĎïijŇåęĆæđIJäÿĂäÿłæŰĞäžűæŔŕè£řçňęèćńè¡ňæ■ćäÿžäÿĂäÿłPythonæŰĞäžűåŕźèšąïijŇä¡ăéIJĂèę PyFile_FromFd() çŽĎæIJĂåŘŐäÿĂäÿłåŔĆæŢřèćńèő¿ç¡őæĹŘ1ïijŇçŤĺæİěæŇĞåĞžPythonåžŤèŕěåĚşéŮ

åęĆæđIJä¡ăéIJĂèęĄäżŐCæăĞåĞĘI/OåžŞäÿ■ä¡£çŤĺåęĆãĂĂfdopen() åĞ¡æŢřæİěåĹŻåžžäÿ■åŘŇçšżåđ̊ŇçŽĎæŰĞäžűåŕźèšąæŕŤåęĆ FILE * åŕźèšąïijŇ ä¡ăéIJĂèęĄçL'źåĹńåŕŔå£ČäžĘãĂĆè£ŹæăůåĄŽäijŽåIJÍI/OåăĘæăĹäÿ■äžğçŤ§äÿd'äÿłåőŇåĚĺäÿ■åŘŇçŽĎI/Oç ïijĹäÿĂäÿłæŸŕæİěèĞłPythonçŽĎ io æÍąåİŮïijŇåŘęäÿĂäÿłæİěèĞłCçŽĎ stdio ïijL'ãĂĆ åČŔCäÿ■çŽĎ fclose() äijŽåĚşéŮ■PythonèęĄ¡£çŤĺçŽĎæŰĞäžűãĂĆ åęĆæđIJèől'ä¡ăéĂL'çŽĎèŕİïijŇä¡ăäžŤèŕëäijŽéĂL'æŇl'åŐżædĎåžžäÿĂäÿłæL'l'åśŢäžčçăĄæİěåd'ĎçŘĘåžŢåśĆ èĂŇäÿ■æŸŕä¡£çŤĺæİěèĞł<stdio.h>çŽĎéŃŸåśĆæŁ¡èśąåŁ§èČ¡ãĂĆ

15.19 äžŐCèŕ■èÍÄäÿ■èŕżåŔŰçšżæŰĞäžűåŕźèšą

éŮőécŸ

ä¡äèęĄåĘŹCæL'l'åśŢæİěèŕżåŔŰæİěèĞłäžżä¡ŢPythonçšżæŰĞäžűåŕźèšąäÿ■çŽĎæŢřæ■őïijĹæŕŤåęĆæŹőé

èğčåĘşæŰźæąĹ

èęĄèŕżåŔŰäÿĂäÿłçśżæŰĞäżűåŕźèśąçŽĎæŢřæ■őïijŇä¡ăéIJĂèęĄéĞ■åd'■èřČçŤÍ read() æŰźæşŢïijŇçĎűåŘŐæ■čçąőçŽĎèğčçăĄèŐůå¿ŮçŽĎæŢřæ■őãĂĆ

äÿŃéÍćæŸŕäÿĂäÿłCæĽľåśŢåĞ¡æŢřä¿Ńå■ŘïijŇäżËäżĚåŔłæŸŕèŕżåŔŰäÿĂäÿłçśżæŰĞäżűåŕźèśąäÿ■çŽĎ

```
#define CHUNK_SIZE 8192

/* Consume a "file-like" object and write bytes to stdout */
static PyObject *py_consume_file(PyObject *self, PyObject *args) {
  PyObject *obj;
  PyObject *read_meth;
  PyObject *result = NULL;
  PyObject *read_args;

  if (!PyArg_ParseTuple(args,"O", &obj)) {
    return NULL;
  }

  /* Get the read method of the passed object */
  if ((read_meth = PyObject_GetAttrString(obj, "read")) == NULL) {
    return NULL;
  }

  /* Build the argument list to read() */
  read_args = Py_BuildValue("(i)", CHUNK_SIZE);
  while (1) {
    PyObject *data;
    PyObject *enc_data;
    char *buf;
    Py_ssize_t len;

    /* Call read() */
    if ((data = PyObject_Call(read_meth, read_args, NULL)) == NULL)
↪{
      goto final;
    }

    /* Check for EOF */
    if (PySequence_Length(data) == 0) {
      Py_DECREF(data);
      break;
    }

    /* Encode Unicode as Bytes for C */
    if ((enc_data=PyUnicode_AsEncodedString(data,"utf-8","strict
↪"))==NULL) {
      Py_DECREF(data);
      goto final;
    }
```

```
    /* Extract underlying buffer data */
    PyBytes_AsStringAndSize(enc_data, &buf, &len);

    /* Write to stdout (replace with something more useful) */
    write(1, buf, len);

    /* Cleanup */
    Py_DECREF(enc_data);
    Py_DECREF(data);
  }
  result = Py_BuildValue("");

 final:
  /* Cleanup */
  Py_DECREF(read_meth);
  Py_DECREF(read_args);
  return result;
}
```

èçĄæţŃèŕŢè£Źäÿłäžčç ăĄïijŇåĚĹæđĎéĂăäÿĂäÿłçśżæŰĞäżűåŕźèśąæŕŤåęĆäÿĂäÿłStringIOåőđä¿ŃïijŇç

```
>>> import io
>>> f = io.StringIO('Hello\nWorld\n')
>>> import sample
>>> sample.consume_file(f)
Hello
World
>>>
```

èőĺèőž

åŠŇæŹőéĂŽçşżçż§æŰĞäżűäÿ■åŔŇçŽĎæŸŕïijŇäÿĂäÿłçśżæŰĞäżűåŕźèśąåźűäÿ■éIJĂèęĄä¡£çŤĺä¡Őçžğ åŽäæ■đ’ïijŇä¡ääÿ■èČ¡ä¡£çŤĺæŹőéĂŽçŽĎCåžŞåĞ¡æŢřæİèèő£éŮőåőČãĂĆ ä¡ăéIJĂèęĄä¡£çŤĺPythonçŽĎC APIæİěåČŔæŹőéĂŽæŰĞäżűçśżäijijçŽĎéĆčæăůæŞ■ä¡IJçśżæŰĞäżűåŕźèśąãĂ

åIJĺæĹŠäżňçŽĎèğčåĘşæŰźæąĹäÿ■ïijŇread() æŰźæşŢäżŐèćńäijăéĂŠçŽĎåŕźèśąäÿ■æŔŘåŔŰåĞžæİě äÿĂäÿłåŔĆæŢřåĹŮèąĺèćńæđĎåżžçĎűåŔŐäÿ■æŰ■çŽĎèćńäijăçżŹ PyObject_Call() æİěèŕČçŤĺè£ŹäÿłæŰźæşŢãĂĆ èęĄæčĂæ§ěæŰĞäżűæIJńåř¿ïijĹEOFïijĽïijŇä¡£çŤĺäžĘ PySequence_Length() æİěæ§ěçIJŃæŸŕåŘęè£ŤåŻđåŕźèśąéŢ£åžęäÿž0.

åŕźäžŐæĽ’ÄæIJĽçŽĎI/OæŞ■ä¡IJïijŇä¡ăéIJĂèęĄåĚşæşĺåžŢåśĆçŽĎçijŰçăĄæäijåijŔïijŇè£ŸæIJĽå■ŮèŁ æIJňèŁĆæijŤçď’žäžĘåęĆä¡ŢäžěæŰĞæIJňæĺąåijŔèŕżåŔŰäÿĂäÿłæŰĞäżűåźűåřĘçżŞæđIJæŰĞæIJňèğčçăĄäÿž åęĆæđIJä¡ăæČşäžěäžŇè£ŻåĹűæĺąåijŔèŕżåŔŰæŰĞäżűïijŇåŔłéIJĂèęĄä£őæŤźäÿĂçĆźçĆźå■şåŔŕïijŇä¿ŃåęĆ

```
...
/* Call read() */
if ((data = PyObject_Call(read_meth, read_args, NULL)) == NULL) {
  goto final;
```

```
}

/* Check for EOF */
if (PySequence_Length(data) == 0) {
  Py_DECREF(data);
  break;
}
if (!PyBytes_Check(data)) {
  Py_DECREF(data);
  PyErr_SetString(PyExc_IOError, "File must be in binary mode");
  goto final;
}

/* Extract underlying buffer data */
PyBytes_AsStringAndSize(data, &buf, &len);
...
```

æIJňèŁĆæIJĂéŽ¿çŽĎåIJřæŰźåIJÍäžŐåęĆä¡Ţè£ŻèąŇæ■čçąőçŽĎåĘĚå■ŸçőąçŘĘãĂĆ å¡Şåď˙ĎçŘĘ`PyObject * `` åŔŸéĞŔçŽĎæŮűåĂŹïijÑéIJĂèęĄæşÍæĎŔçőąçŘĘåijŢçŤÍèőąæŢi åŕź ``Py_DECREF() çŽĎèřČçŤÍåřśæŸŕæİěåĄŽè£ŹäÿłçŽĎãĂĆ

æIJňèŁĆäžčçăĄäžěäÿĂçğ■éĂŽçŤÍæŰźåijŔçijŰåĘŹïijÑåŻăæ■ď˙äžŰäź§èČ¡éĂĆçŤÍäžŐåĚűäżŰçŽĎæŰČ ä¿ŃåęĆïijÑèęĄåĘŹæŢřæ■őïijÑåŔłéIJĂèęĄèŐůåŔŰçśżæŰĞäžűåŕźèśąçŽĎ `write()` æŰźæşŢïijÑářĘæŢřæ■őè¡ňæ■ćäÿžåŘĹéĂĆçŽĎPythonåŕźèśą ïijÍå■ŮèŁĆæĹŰŰnicodeïijL˙ïijÑçĎűåŘŐèřČçŤÍèŕěæŰźæşŢářĘè¿ŞåĚěåĘŹåĚěåĹŕæŰĞäžűãĂĆ

æIJĂåŘŐïijÑář¡çőąçśżæŰĞäžűåŕźèśąéĂŽåÿÿè£ŸæŔŘä¿ŻåĚűäżŰæŰźæşŢïijĹæŕŤåęĆreadline(), read_info()ïijL˙ïijÑ æĹŚäžňæIJĂåě¡åŔłä¡£çŤÍå§žæIJňçŽĎ `read()` åŠŇ `write()` æŰźæşŢãĂĆ åIJÍåĘŹCæL˙l˙åśŢçŽĎæŮűåĂŹïijÑèČ¡çőĂå■Ţářśář¡éĞŔçőĂå■ŢãĂĆ

15.20 åď˙ĎçŘĘCèŕ■èÍÄäÿ■çŽĎåŔŕè£■äžčåŕźèśą

éŮőécŸ

ä¡ăæČşåĘŹCæL˙l˙åśŢäžčçăĄåď˙ĎçŘĘæİěèĞłäżżä¡ŢåŔŕè£■äžčåŕźèśąåęĆåĹŮèąÍãĂĄåĚČçżĎãĂĄæŰĞäź

èğčåĘşæŰźæąĹ

äÿŃéİćæŸŕäÿĂäÿłCæL˙l˙åśŢåĞ¡æŢřä¿Ńå■ŘïijÑæijŤçď˙žäžĘæĂŐæăůåď˙ĎçŘĘåŔŕè£■äžčåŕźèśąäÿ■çŽĎ

```
static PyObject *py_consume_iterable(PyObject *self, PyObject␣
↪*args) {
  PyObject *obj;
  PyObject *iter;
  PyObject *item;

  if (!PyArg_ParseTuple(args, "O", &obj)) {
```

```
    return NULL;
  }
  if ((iter = PyObject_GetIter(obj)) == NULL) {
    return NULL;
  }
  while ((item = PyIter_Next(iter)) != NULL) {
    /* Use item */
    ...
    Py_DECREF(item);
  }

  Py_DECREF(iter);
  return Py_BuildValue("");
}
```

èőÍèőž

æIJňèŁĆäÿ■çŽĎäžčçăĄåŠŇPythonäÿ■åŕźåžŤäžčçăĄçśżäijijãĂĆ PyObject_GetIter() çŽĎèřČçŤÍåŠŇèřČçŤÍ iter() äÿĂæăůåŔŕèŐůå¿ŮäÿĂäÿłè£■äžčåŹÍãĂĆ PyIter_Next() åĞ¡æŢřèřČçŤÍ next æŰźæşŢè£ŤåŻđäÿŃäÿĂäÿłåĚČçť'ăæĹŰNULL(åęĆæđIJæšąæIJL'åĚČçť'ääžĘ)ãĂĆ èęĄæşÍæĎŔæ■čçąőçŽĎåĘĚå■Ÿçőąç ŘĘâĂŤâĂŤ Py_DECREF() éIJĂèęĄåŘŇæŮűůåIJÍäžğçŤ§çŽĎåĚČçť'ăåŠŇè£■äžčåŹÍåŕźèśąæIJňèžńäÿŁåŘŇæŮűèćńèřČçŤÍïijŇ äžěéĄ£åĚ■åĞžçŐřåĘĚå■ŸæşĎéIJšãĂĆ

15.21 èŕŁæŰ■åĹĘæőţéŤŹèŕŕ

éŮőécŸ

èğčéĞŁåŹÍåŻääÿžæ§ŘäÿłåĹĘæőţéŤŹèŕŕãĂĄæĂżçž£éŤŹèŕŕãĂĄèő£éŮőèűŁçŢŇæĹŰůĚűäžŰèĞť'åŚ¡éŤ ä¡ăæČşèŐůå¿ŮPythonåăĘæăĹä£ąæĄŕïijŇäžŐèĂŇæL'¿åĞžåIJÍåŘŠçŤ§éŤŹèŕŕçŽĎæŮűůåĂŹä¡ăçŽĎçÍŃåžŔè£

èğčåĘşæŰźæąĹ

faulthandler æÍąåİŮèČ¡èćńçŤÍæİěåÿőä¡äèğčåĘşè£ŹäÿłéŮőécŸãĂĆ åIJÍä¡ăçŽĎçÍŃåžŔäÿ■åijŢåĚëäÿŃåĹŮäžčçăĄïijŽ

```
import faulthandler
faulthandler.enable()
```

åŔęåd'Űè£ŸåŔŕäžěåČŔäÿŃéİćè£ŹæăůäĄçŤÍ -Xfaulthandler æİěè£ŘèąŇPythonïijŽ

```
bash % python3 -Xfaulthandler program.py
```

æIJĂåŘŐïijŇä¡ăåŕŕäżěëő¿ç¡ő PYTHONFAULTHANDLER çŐŕåćČåŔŸéĞŔãĂĆ åijĂåŘŕ-faulthandleråŘŐïijŇåIJÍCæĽľ'åśŢäÿ■çŽĎèĞť'åŚ¡éŤŹèŕŕäijŽåŕijèĞť'äÿĂäÿłPythonéŤŹèŕŕåäĘæăĹèćńæĽŞå■ŕ

```
Fatal Python error: Segmentation fault

Current thread 0x00007fff71106cc0:
  File "example.py", line 6 in foo
  File "example.py", line 10 in bar
  File "example.py", line 14 in spam
  File "example.py", line 19 in <module>
Segmentation fault
```

ař¡çőąè£Źäÿłåźűäÿ■èČ¡åŚŁèŕĽä¡ăCäżčçăĄäÿ■åŞłéĞŇåĞžéŤŹäžĘïijŇä¡ĘæŸŕèĞşåŕŚèČ¡åŚŁèŕĽä¡ăPyth

èőÍèőž

faulthandleräijŽåIJÍPythonäżčçăĄæĽğèąŇåĞžéŤŹçŽĎæŮűåĂŹåŘŚä¡ăåśŢçd'žèů§èÿłä£ąæĄŕãĂĆ èĞşåŕŚïijŇåőČäijŽåŚŁèŕĽä¡ăåĞžéŤŹæŮűèćńèřČçŤÍçŽĎæIJĂéąűçžğæĽľ'åśŢåĞ¡æŢřæŸŕåŞłäÿłãĂĆ åIJÍpdbåŠŇåĚűäżŰPythonèřČèŕŢåŹÍçŽĎåÿőåŁľäÿŇïijŇä¡ăåŕśèČ¡è£¡æăźæžŕæžŘæĽ¿åĹréŤŹèŕŕæĽĂåIJÍçŽĎ

faulthandleräÿ■äijŽåŚŁèŕĽä¡ăäżżä¡ŢCèŕ■èÍĂäÿ■çŽĎéŤŹèŕŕä£ąæĄŕãĂĆ åŻăæ■ď'ïijŇä¡ăéIJĂèęĄä¡£çŤÍäijăçż§çŽĎCèřČèŕŢåŹÍïijŇæŕŤåęĆgdbãĂĆ äÿ■è£ĞïijŇåIJÍfaulthandlerè£¡èÿłä£ąæĄŕåŘŕäżěëőĺ'ä¡ăåŐżåĹď'æŰ■äžŐåŞłéĞŇçİĂæĽŃãĂĆ è£ŸèęĄæşÍæĎŘçŽĎæŸŕåIJÍCäÿ■æ§ŘäžŽçśżåđŃçŽĎéŤŹèŕŕåŘŕèČ¡äÿ■åď'łåőźæŸŞæĄćåď'■ãĂĆ ä¿ŃåęĆïijŇåęĆæđIJäÿĂäÿłCæĽľ'åśŢäÿćåijČäžĘçÍŃåžŘåäĘæăĹä£ąæĄŕïijŇåőČäijŽèőĺ'faulthandleräÿ■åŘŕçŤ éĆčäźĹä¡ääź§å¿Ůäÿ■åĹřäżżä¡Ţè¿ŞåĞžïijĹéŹď'äžĘçÍŃåžŘåěŤæžČåď'ŰïijĽãĂĆ

CHAPTER 18

éŹĎå¡ŢA

åIJÍçž£èţĎæžŘ

http://docs.python.org

åęĆæđIJä¡ăéIJĂèęĄæůśåĚěäžĘèğčæŐćçľ˝űèŕ■èÍĂåŠŇæÍąåİŮçŽĎçżĘèŁĆïijŇéĆčäźĹäÿ■å£Ěèŕť˝ïijŇPyth
3 çŽĎæŰĞæąčèĂŇäÿ■æŸŕäžěåL’■çŽĎèĂĄçL’ĹæIJň

http://www.python.org/dev/peps

åęĆæđIJä¡ăåŘŠçŘĘèğčäÿžpythonèŕ■èÍĂæůżåŁăæŰřçL’źæĂğçŽĎåŁÍæIJžäžěåŔŁåőđçŐřçŽĎçżĘèŁĆïijŇ
Enhancement Proposals—-PythonåijĂåŔŠçijŰçăĄèğĎèŇČïijL’çżİåŕźæŸŕéİđåÿÿåőİèť˝ţçŽĎèţĎæžŘãĂĆåřd’å

http://pyvideo.org

è£ŹéĞŇæIJL’æİěèĞłæIJĂè£ŠçŽĎPyConåď˝ğäijŽãĂĄçŤÍæĹůçżĎèğĄéİćäijŽç■L’çŽĎåď˝ğéĞŔèğĘéćŚæi
3äÿ■æůżåŁăçŽĎçŽĎæŰřçL’źæĂğãĂĆ

http://code.activestate.com/recipes/langs/python

éŢ£æIJ§äžěæİěïijŇActiveStateçŽĎPythonçL’ĹåİŮåůšçżŔæĹŘäÿžäÿĂäÿłæL’¿åĹřæŢřäžěå■ČèőąçŽĎéŠĹ

http://stackoverflow.com/questions/tagged/python

Stack Overflow æIJÍåŹÍåŢŁæIJL’èűĚè£Ğ175,000äÿłéŮőécŸècńæăĞèőřäÿžPythonçŻÿåĚşïijĹèĂŇåĚűäÿ
3çŽĎïijL’ãĂĆår¡çőąéŮőécŸåŠŇåŻđç■ŤçŽĎèť˝ÍéĞŔäÿ■åŘŇïijŇä¡ĘæŸŕäż■çĎűèČ¡åŔŠçŐřå¿Ĺåď˝Žåě¡äijŸçğ

Pythonå■ęäźăäźęçś■

äÿŃéİćè£ŹäžŻäźęçś■æŔŘä¿ŻäžĘåŕźPythonçijŰçÍŃçŽĎåĚěéŮÍäžŃçż■ïijŇäÿŤéĞ■çĆźæŤ¿åIJÍäžĘPytho
3äÿŁãĂĆ

Beginning Python: From Novice to Professional, 2nd Edition, by Magnus Lie HetâĂŘ land, Apress (2008). Programming in Python 3, 2nd Edition, by Mark Summerfield, Addison-Wesley (2010).

- *Learning Python*ïijŇçňňåŻŻçĽĹ ïijŇä¡IJèĂĚ Mark LutzïijŇ OâĂŹReilly & Associates åĞžçĽĹ (2009)ãĂĆ
- *The Quick Python Book*ïijŇä¡IJèĂĚ Vernon CederïijŇ Manning åĞžçĽĹ(2010)ãĂĆ
- *Python Programming for the Absolute Beginner*ïijŇçňňäÿĽçĽĹïijŇä¡IJèĂĚ Michael DawsonïijŇCourse Technology PTR åĞžçĽĹ(2010).
- *Beginning Python: From Novice to Professional*ïijŇçňňäžŇçĽĹïijŇ ä¡IJèĂĚ Magnus Lie HetâĂŘ landïijŇ Apress åĞžçĽĹ(2008).
- *Programming in Python 3*ïijŇçňňäžŇçĽĹïijŇä¡IJèĂĚ Mark SummerfieldïijŇAddison-Wesley åĞžçĽĹ (2010).

énŸçžğäźęçś■

äÿŃéİćçŽĎè£ŹäžŻäźęçś■æŔŘä¿ŻäžĘæŻťåď ŽénŸçžğçŽĎèŇČåŻťïijŇäź§åŇĚåŘńPython 3æŰźéİćçŽĎåĘĚåőźãĂĆ

- *Programming Python*ïijŇçňňåŻŻçĽĹ, by Mark Lutz, OâĂŹReilly & Associates åĞžçĽĹ(2010).
- *Python Essential Reference*ïijŇçňňåŻŻçĽĹïijŇä¡IJèĂĚ David Beazley, Addison-Wesley åĞžçĽĹ(2009).
- *Core Python Applications Programming*ïijŇçňňäÿĽçĽĹïijŇä¡IJèĂĚ Wesley Chun, Prentice Hall åĞžçĽĹ(2012).
- *The Python Standard Library by Example* ïijŇ ä¡IJèĂĚ Doug HellmannïijŇAddison-Wesley åĞžçĽĹ(2011).
- *Python 3 Object Oriented Programming*ïijŇä¡IJèĂĚ Dusty Phillips, Packt Publishing åĞžçĽĹ(2010).
- *Porting to Python 3*ïijŇ ä¡IJèĂĚ Lennart RegebroïijŇCreateSpace åĞžçĽĹ(2011), http://python3porting.com.

CHAPTER 19

åĚşäžŐèŕŚèĂĚ

åĚşäžŐèŕŚèĂĚ

- åğŞåŘ■ïijŽ çĘŁèČ¡
- å¿őä£ąïijŽ yidao620
- EmailïijŽ yidao620@gmail.com
- å■Žåőćïij Ž http://yidao620c.github.io/
- GitHubïijŽ https://github.com/yidao620c

CHAPTER 20

Roadmap

2014/08/10 - 2014/08/31:

```
|    githubéąźçŻőæŘ■åżžïijŇreadthedocsæŰĞæąčçŤ§æĹŘãĂĆ
|    æŢt’äÿłéąźçŻőçŽĎæąĘæđűåőŇæĹŘ
```

2014/09/01 - 2014/10/31:

```
|    åL’■4çńăç£żèŕŚåőŇæĹŘ
```

2014/11/01 - 2015/01/31:

```
|    åL’■8çńăç£żèŕŚåőŇæĹŘ
```

2015/02/01 - 2015/03/31:

```
|    åL’■9çńăç£żèŕŚåőŇæĹŘ
```

2015/04/01 - 2015/05/31:

```
|    10çńăç£żèŕŚåőŇæĹŘ
```

2015/06/01 - 2015/06/30:

```
|    11çńăç£żèŕŚåőŇæĹŘ
```

2015/07/01 - 2015/07/31:

```
|    12çńăç£żèŕŚåőŇæĹŘ
```

2015/08/01 - 2015/08/31:

```
|    13çńäç£żèŕŚåőŇæĹŘ
```

2015/09/01 - 2015/11/30:

```
|    14çńäç£żèŕŚåőŇæĹŘ
```

2015/12/01 - 2015/12/20:

```
|    15çńäç£żèŕŚåőŇæĹŘ
```

2015/12/21 - 2015/12/31:

```
|    åŕźåĚĺéČĺç£żèŕŚè£ŻèąŇæăąåŕźäÿĂæňą
```

2016/01/01 - 2016/01/10:

```
|    åŕźåd'ŰåĚňåijĂåŔŚåÿČåőŇæŢt'çL'Ĺ1.
↪0ïijŇåŇĚæŃňè¡ňæ■ćåŘŐçŽĎPDFæŰĞäżű
```