嵌入式系統
Lab 04: Python 程式設計

銘傳大學 電腦與通訊工程學系 羅嘉寧

簡介 2

- Script Program Language
- Object-Oriented Program Language
- General-Purpose Program Language
- Easy to learn
- 誰在使用Python呢?
  - 大神Google
  - 美國太空總署(NASA)
  - ...
- [How to Become a Hacker] 一文中推薦使用

使用Python 3

- 有兩種主要使用python的方法
  - 使用互動式命令列
    - e.g. 直接鍵入python就會進入python的互動式命令列
  - 將程式寫成檔案，再由python執行
    - 直在將程式碼寫在檔案內，然後再執行python去讀取該檔案
      - Ex: python hello.py
    - 或是在檔案的第一個行寫著 #!/usr/bin/python，然後在第二行之後輸入程式碼，如此可以直接執行該檔案
      - Ex: ./hello.py
- 作業平台
  - Linux、FreeBSD ...
  - Windows

安裝Python 4

- 執行下列指令安裝Python執行環境

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install python2.7
```

您的第一個python程式 – Hello World 5

- 使用互動式命令列

```
>>> print "Hello World"
Hello World
>>>
```

- 放在檔案裡

```
#!/usr/bin/python
print "Hello World"
```

  - 記得將檔案改成可執行 chmod a+x <檔名>

基本概念 6

- 語法特色
  - 以冒號(:)做為敘述的開始
  - 不必使用分號(;)做為結尾
  - 井字號(#)做為註解符號，同行井字號後的任何字將被忽略
  - 使用tab鍵當為區塊的依據
  - 不必指定變數型態 (runtime時才會進行binding)

變數(Variables)和表示式(Expressions) 7

- 表示式

```
3 + 5
3 + (5 * 4)
3 ** 2              // 3的2次方(平方)
'hello' + 'World'   // 字串串接'helloWorld'
```

- 變數指定

```
a = 4 << 3
b = a * 8.5
c = (a+b)/2.5
s = "Hello World"   // s變成字串類別
```

  - 型別是動態的，會根據指定時的物件來決定型別
  - 變數單純只是物件的名稱，並不會和記憶體綁在一起。e.g.和記憶體綁在一起的是物件，而不是物件名稱。

條件式敘述(Conditional Statements) Part I 8

- if..else

```
if a < b:
    z = b
else:
    z = a
```

- pass 敘述 – 不做任何事時使用

```
if a < b:
    pass
else:
    z = a
```

條件式敘述(Conditional Statements) Part II 9

- elif敘述

```
if a == '+':
    op = PLUS
elif a == '-':
    op = MINUS
else:
    op = UNKNOWN
```

  - 沒有像C語言一樣，有switch的語法
- 布林表示式 – and, or, not

```
if b >= a and b <= c:
    print 'b is between a and c'
if not (b < a or c < b):
    print 'b is still between a and c'
```

基本型態 (Numbers and String) 10

- Numbers (數)

```
a = 3               # Integer (整數)
b = 4.5             # Float point (浮點數)
c = 5172588883938   # Long integer (精準度無限)
d = 4 + 2j          # Complex number (複數)
```

- Strings (字串)

```
a = 'Hello'                 # Single quotes
b = "World"                 # Double quotes
c = "Bob said 'Hey there.'" # A mix of both
d = """A triple quoted string # 跨越多行的字串(用於執行檔案中)
can span multiple lines
like this"""
e = """Also works for double quotes""" # 跨越多行的字串(用於執行檔案中)
```

基本型態 – 串列(Lists) 11

- 任意物件的串列

```
a = [2, 3, 4]           # A list of integer
b = [2, 7, 3.5, "Hello"]  # A mixed list
c = []                  # An empty list
d = [2, [a, b]]         # A list containing a list
e = a + b               # Join two lists
```

- 串列的操作

```
x = a[1]        # Get 2nd element (0 is first)
y = b[1:3]      # Return a sub-list
z = d[1][0][2]  # Nested lists (z is 4)
b[0] = 42       # Change an element
```

基本型態 – 固定有序列(Tuples) 12

- Tuples

```
f = (2,3,4,5)                 # A tuple of integers
g = ()                        # An empty tuple
h = (2, [3,4], ((2,),),(2))   # A tuple containing mixed objects
```

- Tuples的操作

```
x = f[1]        # Element access, x = 3
y = f[1:3]      # Slices, y = (3,4)
z = h[1][1]     # Nesting, z = 4
```

- 特色
  - 與list類似，最大的不同tuple是一種唯讀且不可變更的資料結構
  - 不可取代tuple中的任意一個元素，因為它是唯讀不可變更的

基本型態 – 字典 (Dictionaries)

13

- Dictionaries (關聯陣列)

```
a= {}              # An empty dictionary
b = { 'x' : 3, 'y' : 4 }
c = { 'uid': 105,
      'login': 'beazley',
      'name' : 'David Beazley'
    }
```

- Dictionaries的存取

```
d = c['uid']       # Get an element
d['shell'] = '/bin/csh'    # Set an element
if c.has_key('directory'): # Check for presence of an member
    d = c['directory']
else:
    d = None

d = c.get('directory',None)    #Same thing, more compact
```

迴圈 (Loops)

14

- while敘述

```
while a < b:
    # Do something
    a = a + 1
```

- for敘述 (走訪序列的元素)

```
for i in [3, 4, 10, 25]:
    print i

# Print characters one at a time
for c in "Hello World":
    print c

# Loop over a range of numbers
for i in range(0,100):
    print i
```

函式 (Functions)

15

- def敘述

```
# Return the remainder of a/b
def remainder(a,b):
    q = a/b
    r = a - q*b
    return r

# Now use it
z = remainder(42,5)        # z = 2
```

- 回傳一個以上的值

```
def divide(a,b):
    q = a/b
    r = a - q*b
    return q,r

xy = divide(42,5)          # x = 8, y = 2
```

類別 (Classes)

16

- class敘述

```
class Account:
    def __init__(self, initial):
        self.balance = initial
    def deposit(self, amt):
        self.balance = self.balance + amt
    def withdraw(self,amt):
        self.balance = self.balance - amt
    def getbalance(self):
        return self.balance
```

- 使用定義好的class

```
a = Account(1000.00)
a.deposit(550.23)
a.deposit(100)
a.withdraw(50)
print a.getbalance()
```

例外處理 (Exceptions)

17

- try敘述

```
try:
    f = open("foo")
except IOError:
    print "Couldn't open 'foo'. Sorry."
```

- raise敘述

```
def fact(n):
    if n < 0:
        raise ValueError, "Expected non-negative number"
    if (n == 1) or (n == 0):
        return 1
    else:
        return n*fact(n-1)
```

- 沒有處理的例外

```
>>> fact(-1)
Traceback (innermost last):
  File "<stdin>", line 1, in ?
  File "<stdin>", line 5, in fact
ValueError: Expected non-negative number
>>>
```

檔案處理

18

- open()函式

```
f = open("foo","w")  # Open a file for writing
g = open("bar","r")  # Open a file for reading
```

- 檔案的讀取/寫入

```
f.write("Hello World")
data = g.read()        # Read all data
line = g.readline()    # Read a single line
lines = g.readlines()  # Read data as a list of lines
```

- 格式化的輸入輸出
  - 使用%來格式化字串

```
for i in range(0,10):
    f.write("2 times %d = %d\n" % (i, 2*i))
```

模組 (Modules) 19

- 程式可分成好幾個模組
- import敘述

Python的標準模組函式庫 20

- Python本身就包含了大量的模組提供使用
  - String processing
  - Operating system interfaces
  - Networking
  - Threads
  - GUI
  - Database
  - Language services
  - Security.
- 使用模組

```
import string
...
a = string.split(s)
```

參考 21

- 官方網頁(英)
  - http://www.python.org
- Python教學文件(中)
  - http://www.freebsd.org.hk/html/python/tut_tw/tut.html

CHECK POINT 1 22

- 請寫一支程式輸出下列結果

```
*
**
***
****
*****
******
*******
********
*********
**********
```

CHECK POINT 2 23

- 請寫一支程式輸入一個字元後輸出下列結果

```
Please input a character: X
X
XX
XXX
XXXX
XXXXX
XXXXXX
XXXXXXX
XXXXXXXX
XXXXXXXXX
```