Python 程式設計 II

Mosky 2011/4/30

過去簡報、範例檔案下載

1. j.mp/pyintro
2. j.mp/py-I

快速複習 Quick Review

注意事項 Notice

1. 以換行字元 (\n) 做為敘述結尾，分號 (;) 是選用的。
2. 縮排協定，同一個區塊 (block) 的縮排必須一致。習慣上使用 4 個空白。
3. 可執行腳本宣告 (unix-like only)，需放在第一行。

```
#!/usr/bin/env python
```

4. 編碼宣告(Python 2.x only)，若使用到非 ASCII 字元，需加上編碼宣告，並放在檔案的最開頭。

```
# -*- encoding: utf-8 -*-
# -*- encoding: big5 -*-
```

5. PEP 8: Style Guide for Python Code [1]。

內建資料型態 Native Data Type

A) Numeric Types: **`int`**, **`float`**, `long`, `complex`

B) Sequence Types: **`str`**, **`unicode`**, `list`, `tuple`

C) Set Types: `set`, `frozenset`

D) Mapping Types: **`dict`**

types	iterable	mutable	order	mapping
`list`	yes	yes	yes	

`str`
`unicode`
`tuple`
	yes		yes	

`set`	yes	yes		
`frozenset`	yes			
`dict`	yes	yes		yes

1 請見 http://www.python.org/dev/peps/pep-0008/

流程控制敘述 Control Flow Statement

```
if /條件式 A/:
    /條件式 A 成立時執行的區塊/
elif /條件式 B[1]/:
    /條件式 B[1]成立時執行的區塊/
elif /條件式 B[2]/:
    /條件式 B[2]成立時執行的區塊/
…
else:
    /當條件式 A 與 B[1...n]都不成立時執行的區塊/

for /項目/ in /可迭代(列舉)物件/:
    /每次迭代執行的區塊/
else:
    /當迴圈沒有被 break 中斷時執行/

while /條件式 W/:
    /當條件式 W 成立時執行的區塊/
else:
    /當迴圈沒有被 break 中斷時執行/
```

自訂函數 User-defined Function

基本函數寫法:

```
def /函數名稱/(/參數列/):
    /函數執行時執行的區塊/
    return /表達式/
```

參數預設值 (Default Arguments Value):

```
def /函數名稱/(x=10):
    /.../
```

注意: 預設值盡量使用 immutable 的資料型態。

收集任意長度參數 (Arbitrary Arguments):

```
def /函數名稱/(a, b, *all):
    /.../
```

收集所有鍵參數 (Keyword Arguments):

```
def /函數名稱/(a, b, **keyword):
    /.../
```

解開序列 (Unpack Sequence):

```
def f(x, y, z): ...
my_args = (1, 2, 3)
result = f(*my_args)
```

解開映射 (Unpack Mapping):

```
def f(x, y, z): ...
my_kargs = {'x': 1, 'y': 2, 'z': 3}
result = f(**my_kargs)
```

函數一定有回傳值，若無定義 return，會回傳 None。

＃1：已知費式數列定義如下： fib(0) = 0; fib(1) = 1; fib(n) = fib(n-1) + fib(n-2)，請利用遞迴的方式寫出 fib 函數。

理解(建構)式 Comprehension

串列 (list) 建構式

```
[/表達式/ for /項目 1/ in /可迭代物件 1/
          if /條件式 1/
          for /項目 2/ in /可迭代物件 1/
          if /條件式 2/
          … ]
```

產生器 (generator) 建構式

```
(/表達式/ for /項目 1/ in /可迭代物件 1/
          if /條件式 1/
          for /項目 2/ in /可迭代物件 1/
```

```
        if /條件式 2/
        … )
```

寫在函數呼叫內可以省略圓括號，如:

```
any(i % 2 == 0 for i in A)
```

```
#2: 請利用 any 與 generator comprehension 撰寫一個判斷輸入參數是否為質數的函數。
```

習題講解 Pratice Review

使用 .format 製作金字塔

1. format string syntax 可以使用巢層撰寫
2. (額外) 使用 **locals() 技巧將區域變數解開映射成為 .format 的鍵參數

使用 comprehesion 撰寫矩陣相加、減

1. 使用巢層的 list comprehesion
2. (額外) 使用 operator 模組
3. (額外) 使用 functools.partial 特定化函數
4. (額外) 使用 jump table 技巧來對應使用者輸入的命令與函數

使用 re 過濾正確的手機號碼

1. 正確的使用正規表示式 (regular expression or regex)[2]

Python 的動態定型 Dynamic Typing in Python

- `a = 1`
  1. 製作物件
  2. 建立變數
  3. 將變數繫結到物件
- 沒有宣告變數為什麼型態
- 型態在於物件而不是變數

2 詳細請見 http://docs.python.org/library/re.html

進階函數設計 **More on Function**

基礎概念

1. Python treats function as first-class objects.
2. 在 Python 中，函數可以像變數一樣隨意地被傳遞。
3. LEGB (Local Enclose Global Built-in) 規則。

lambda (λ) 敘述

1. 匿名 (anonymous) 函數
2. 單行
3. 表達式即回傳值

```
f = lambda x: x**2
```

```
# 3: 請嘗試使用 lambda 敘述撰寫計算階乘的遞迴函數。
```

閉包 (Closure)

1. 保留狀態
2. 會有巢層函數或普通函數包 lambda
3. 應用：工廠函數 (factory function)

```
# 4: 請撰寫一個 mkrepeat 函數，接受一個數字作為參數，回傳值為一個函數，其可以將傳入的值進行乘法或重複運算。
```

修飾器 (Decorator)

1. `@g; def f(): pass`
2. 相當於 `f = g(f)`

```
# 5: 首先請先撰寫一個函數，接受一個參數 n，並直接回傳；再在其上方加上一個修飾器函數，會將這個函數的回傳值修改成 n!。
```

產生器 (Generator)

1. 支援迭代器協定，是一種可迭代物件 (.next)
2. 節省記憶體
3. 互動式的函數 (.send)
4. 協同函數 (Coroutine)

```
def /函數名稱/(/參數列/):
    /函數執行時執行的區塊/
    yield /表達式/
```

```
x = (yield x) #先送出 x 的值，再將 .send 送入的值存入 x
```

#6: 請將#1的 fib 函數修改為產生器版本。也就是說它可以直接被迭代，[0...n]的值就是[fib(0)...fib(n)]的值。

自訂模組 User-defined Module

內建模組示範請參考範例檔案。

1. 除了內建模組，每個 .py 檔案都可以當作是模組匯入
2. 在資料夾內加入 __init__.py，可以下層檔案都能夠被匯入
3. 匯入語法:

```
import module
import module as m
from module import something
from module import something as s
```

4. 可以利用 a.b.c 的方式進行相對路徑的匯入

#7: 在 fractions 模組中有個 gcd 函數，它接受兩個數字，並傳回兩者的最大公因數。請試著使用 fractions.gcd, functools.partial, reduce 與 unpacking 技巧，製作一個可以接受任意長度參數的 gcd 函數。

#8: 請利用 urllib 且/或 urllib2 與 re 模組，過濾搜尋引擎的查詢結果，找出所有合法的 e-mail 帳號並列出所有不重複的帳號與數量。

自訂類別 User-defined Class

```
class /類別名稱/(/父類別/):
    /定義類別屬性 Class Attribute/

    def /方法名稱/(/實例/, /參數列/):
        /方法區塊/

    @classmethod
    def /類別方法名稱/(/類別/, /參數列/):
        /類別方法區塊/

    @staticmethod
    def /靜態方法名稱/(/參數列/):
```

/靜態方法區塊/

物件 Object

Everything is object in Python

類別 Class

類似產品的模型

實例 Instance

使用模型製作出的產品

```
instance_of_a_class = AClass()
```

屬性 Attribute

繫結於實例的變數

方法 Method

繫結於實例的函數

特殊方法 Special Method

1. 有雙底線包著的方法：__specialmethod__
2. 經常使用 __init__ 來初始化實例，參考 Python 的資料模型以取得更多資訊[3]。

綁定與未綁定 Bound/ Unbound Method

1. 實例方法尚未被實體化時，屬於未綁定方法，也就是可以自由指定要用於哪個實例。
2. 被實體化時，第一個變數會被綁定至一個實例。

類別屬性與方法 Class attribute and method

繫結於類別的屬性或方法。類別方法的第一個參數會是該類別物件。

靜態方法 Static method

與普通函數相同，只是被繫結在類別上。

鴨子定型 Duck Typing

1. 「當看到一隻鳥走起來像鴨子、游泳起來像鴨子、叫起來也像鴨子，那麼這隻鳥就可以被稱為鴨子。」

3 Python 的資料模型 http://docs.python.org/reference/datamodel.html

2. 不關注物件本身的型別，而是物件能夠如何被使用。

繼承 Inheritance

迭代 (列舉) 器協定 Iterator Protolcol

1. `container.__iter__()`
2. `iterator.__iter__()`
3. `iterator.next()`
4. `StopIteration`

#9: 請撰寫一個 Person 類別，有 name、email 與 cpnumber 屬性，並可以利用實例方法 display 印出。及可以利用類別屬性 count 得知總共實體化了幾個 Person 實例。

#10: 請撰寫一個提供 read, write, truncate, seek, tell, close 方法，並可以和使用者互動操作的 InteractiveFile 類別。

#11: 請利用繼承的方式撰寫一個不會有重複元素的 list。

#12: 請利用繼承的方式撰寫一個有序的辭典。