Python Tutorial

He-Zhe Lin 林鶴哲

2020.07.16 @ CSCamp

Before the class: 關於我

資工系大二升大三

只參加過一次資訊營

可以接受夏威夷披薩 :-)

最重要的: 我是辣個擁有最多道具卡的小摩爾！

Before the class: 關於電腦

裡面有裝還原卡, 會動到助教們辛辛苦苦設定的設定檔

所以***不要關機！不要關機！不要關機！***

離開時請聽講師指示動作～

About the class

講師將會努力和大家互動, 不斷丟問題

懂了就請一直點頭, 不懂請擺出疑惑表情

歡迎隨時打斷發問, 努力**善待**每個問題, 課堂上也沒有**笨問題**

不要不理我, 我上不下去 TAT

Overview

Chapter 1: Introduction

Chapter 2: Variables

Chapter 3: Input / Output

~~Chapter 4: if / else~~

Learning 4 chapters in 2 hours, interesting :-)

Introduction

程式

程式的英文是 Program，簡單地說就是劇本。

這個劇本是給計算機看的，劇本裡包含一條條的指令。

當計算機看到這一條條的指令，就會執行這些指令所要求的動作。於是藉由設計這一條一條的指令，我們就可以命令計算機執行我們想要做的事情。

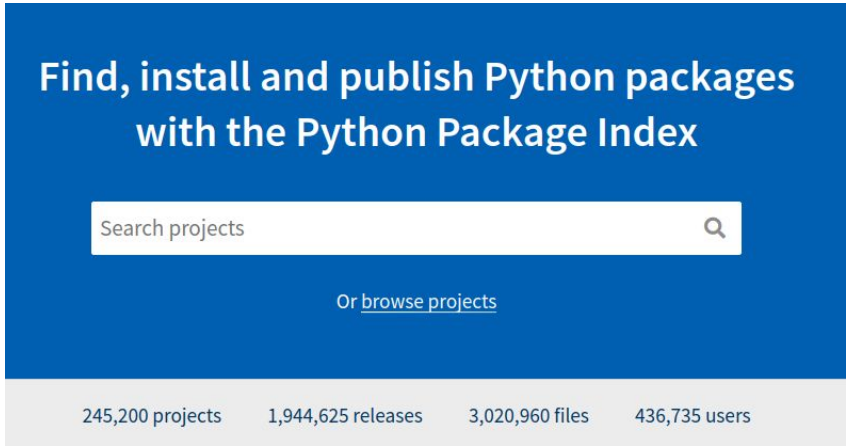
-- 劉邦鋒 (2012)。用片語學習 C 程式設計 。

Why Python?

簡單易上手

```
Zen of Python
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
```

廣大的社群



開始寫程式

Interactive mode: 寫一行跑一行

- 執行方式:直接在 powershell 上打 `python`

```
→  ~ python
Python 3.6.9 (default, Apr 18 2020, 01:56:04)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

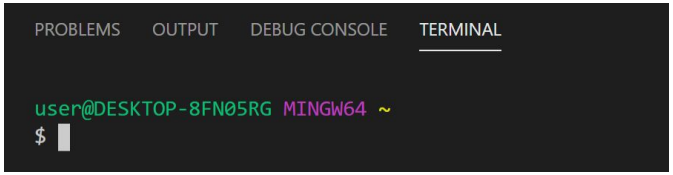
Script mode: 寫整段一次跑

- 執行方式:
  把程式碼放在一個 `.py` 檔內，在 powershell 上打 `python [檔名].py`

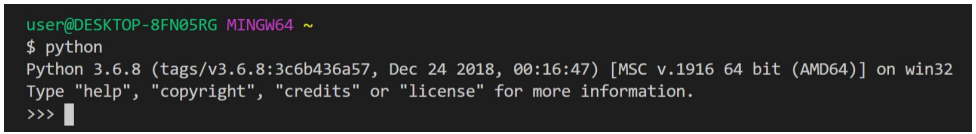
```
→  0 python hello.py
hello world!
→  0
```

手把手教學 - Interactive mode

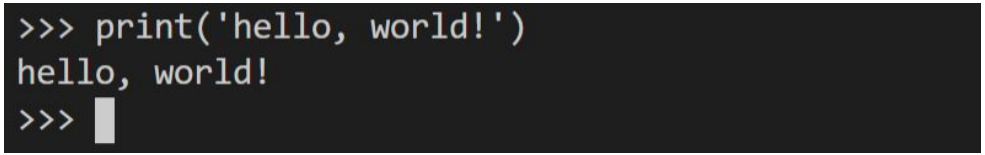
1. 開啟 VS code，按 `ctrl + shift + ` ` 進入 Terminal



2. 在 terminal 打上 `python`，跑出互動式的畫面。



3. 一字不漏的打上 `print('hello world!')`



退出：`ctrl + Z` or `exit()`

第一個 Python 程式: Hello world

【懸賞】

第一個全組用 Script mode 寫完 "Hello, world!" 的組別。

手把手教學 - Script mode

1. 在桌面新增一個自己 team_id 的資料夾（如:0）
2. 在資料夾裡面新增一個叫做 hello.py 的檔案
3. 用 Visual Studio Code 開啟 hello.py，一字不漏的打上

```
print('Hello world!')
```

4. 按 `ctrl + shift + ` ` 進入 Terminal, 一字不漏的打上

```
cd ~/Desktop/[team_id]
```

5. 在 terminal 打上 `python hello.py`，執行第一份 python 程式。

```
user@DESKTOP-8FN05RG MINGW64 ~/Desktop/0
$ python hello.py
hello, world!
```

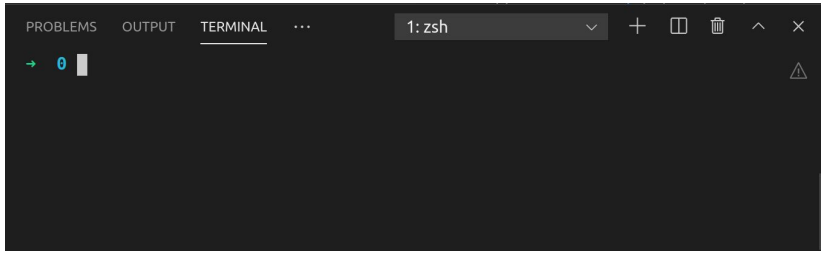
你會寫程式了！

VS code 快捷鍵

```
ctrl + 'N': 開新檔案
ctrl + 'O': 開啟檔案
ctrl + 'S': 存檔
ctrl + shift + ` (左上角的波浪符號那格): 開啟 Terminal
ctrl + '=': 放大文字
ctrl + '-': 縮小文字
```

快捷鍵按下去, 程式寫出來, ？ 小發大財！(請自行帶入)

Terminal 快捷鍵




```
cd ~/Desktop                # change directory to Desktop(切換至桌面)
cd [資料夾名稱]              # 切換至某個資料夾內
python                      # 用 interactive mode 執行 python
python [檔名]               # 用 script mode 執行該檔案
ctrl + shift + 'C'          # 從 terminal 複製文字到其他地方
ctrl + shift + 'V'          # 從其他地方貼上文字到 terminal
```

快捷鍵按下去, 程式寫出來, ? 小發大財！(請自行帶入)

怎麼寫「好」程式

```
class Num:
    def __init__(self,i,j):
        self.i,self.j=i,j
    def getdata(self):
        return (self.i**2+self.j**2)**0.5
    def method1(self,num):
        self.i,self.j=self.i+num.i,self.j+num.j
    def method2(self,num):
        self.i,self.j=self.i*num.i-self.j*num.j,
self.i*num.j+self.j*num.i
```

怎麼寫「好」程式

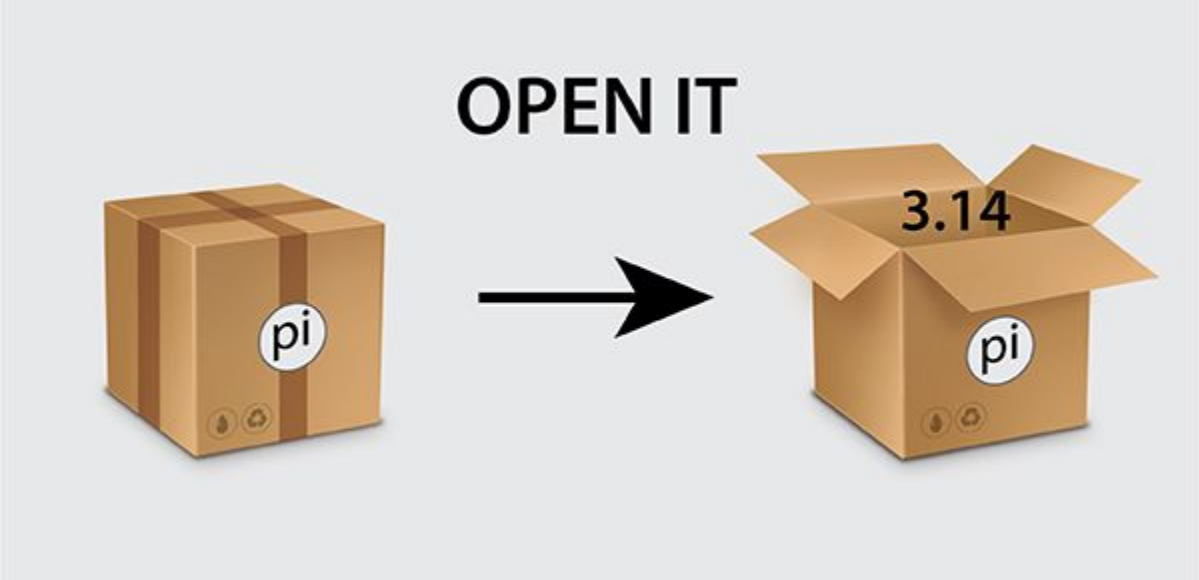
```
class Complex:
    def __init__(self, real, img):
        self.real, self.img = real, img

    def abs(self):          # get the absolute value of a complex number
        return (self.real ** 2 + self.img ** 2) ** 0.5

    def add(self, other):
        self.real, self.img = self.real + other.real, self.img + other.img

    def multiply(self, other):
        self.real, self.img = self.real * other.real - self.img * other.img, \
                              self.real * other.img + self.img * other.real
```

變數命名

註解:# 至行末

換行 / 空白

Variables



變數就是「名字」+「值」

名字 (name)

- 有名字才能稱呼他
- 包含英文字母 (a-zA-Z)、數字 (0-9)、底線 (_)
- 如：x, math_score, money ...

值 (value)

- 有值才有料，不然變數空空的
- 根據不同**型態**的變數有不同的值
- 如：3.14, 'pizza', 6 ...

Python 當中有哪些型態？

型態 (type)：變數的身份

字串 (str)

浮點數 (float)

多元組 (tuple)

整數 (int)

字典 (dict)

序列 (list)

複數 (complex)

迭代器 (iterator)

只要我想要，就做得到！

賦值:賦予某變數某值

語法:[變數名稱] = [值]

`a = 1` 代表的是:

- 讓 `a` 這個變數擁有整數 `1` 的值。
- 把 `1` 這個整數放到 `a` 變數當中。

```
>>> a = 1
>>> b = 1.414
>>> c = "Welcome to CSIE camp!"
```

int / float: 就是數字

```
>>> a = 7
>>> b = 3
>>> a + b
>>> a - b
>>> a * b
>>> a / b
>>> a ** b
>>> a // b
>>> a % b
>>> 2 ** 2000
```

技法:修改變數值

方法一

重新賦值

方法二

用 +=, -=, ... 更改變數值

```
>>> x = 0
>>> x = x + 3
>>> x += 4
>>> x = x - 2
>>> x -= 1
>>> x *= 3.5
>>> x /= 0.7
>>> x
20
```

Exercise: 一元二次方程式

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

請找出方程式「$x^2 - 1433\,x + 513372 = 0$」的解

Step 1：將方程式化成 $ax^2 + bx + c = 0$ 的形式，找出 a, b, c

Step 2：用 Python 宣告 a, b, c 三個變數，用公式解得到答案

Hint

注意四則運算（小括弧）

```
>>> (-b + (b**2 - 4ac)**0.5) / (2a)
  File "<stdin>", line 1
    (-b + (b**2 - 4ac)**0.5) / (2a)
                       ^
SyntaxError: invalid syntax
```

Solution: 一元二次方程式

請找出方程式「$x^2 - 1433\ x + 513372 = 0$」的解

```
>>> a = 1
>>> b = -1433
>>> c = 513372
>>> (-b + (b * b - 4 * a * c) ** 0.5) / (2 * a)
717.0
>>> (-b - (b * b - 4 * a * c) ** 0.5) / (2 * a)
716.0
```

多元素變數:有很多元素的變數

一個變數

一個元素

字串 (str)

序列 (list)

多元組 (tuple)

Q: 要怎麼拿到變數中的元素？

取得變數中的元素

取得元素**個數**	取得一個元素	取得一串元素 (start 到 stop-1)
`len(object_name)`	`object_name[index]`	object_name[start:stop]

【注意】【顛覆世界】資工的世界, index 從 0 開始數！

```
>>> A = 'abcdefghij'
       # 0123456789
>>> len(A)
>>> A[0]
>>> A[5]
>>> A[10]
>>> A[-1]
>>> A[0:2]
```

字串 (str) - 很多字元組起來

語法:以兩個單引號 ' 或是兩個雙引號 " 包夾住, ' 和 " 是重點。

```
>>> str1 = '廁所'
>>> str2 = '門'
>>> str1[0]

>>> str1[0] = '便'                    # 不可以單獨更改某個字元
```

【注意】字串是 **immutable** 的, 不能單獨更改變數當中的某個元素。

字串 (str) - 「+」和「*」

```
>>> str1 = '便所'
>>> str1 = '便' + str1[1]          # 另一種修改方式

>>> str1 + str2                    # 字串加法:把兩個字串串接起來

>>> str2 * 8 + str1 + str2         # 字串乘法:複製之後串接


>>> '1' + '1'

>>> '1' * 4
```

序列 (list) - 多元素有順序的排列

置物櫃裡要放什麼都可以(甚至是另一個置物櫃)

語法:一個序列以 `[]` 包住, 不同元素以 `','` 隔開 (`[]` 是重點)。

```
>>> my_list = list()
>>> my_list = [ '就是', 87, 9.4, ['Another', 'list'] ]
>>> my_list[3][0]
>>> my_list.pop(3)
>>> my_list[2] = 94
>>> my_list.append('狂')
>>> my_list
[ '就是', 87, 94, '狂' ]
```

多元組 (tuple) - Immutable 的 list

語法:在一個 tuple 的不同元素間用逗點 , 隔開, 逗點是重點。

```
>>> my_tuple = tuple()
>>> my_tuple = ()

>>> my_tuple = (123)
>>> my_tuple = (123, )
>>> my_tuple = 123,

>>> my_tuple[0] = 0
```

技法:tuple 超好用

一次宣告多個變數

一次交換兩個變數

```
>>> a, b = 1, 3

>>> c, d = 'a', 'b', 'c'
>>> c, d = 'a', ('b', 'c')

>>> a, b = b, a
```

字典 (dict) - *key : value* 的對應關係

中英字典: 中文 -> 英文

在抽屜內的 value

key

可以是任何型態！

可以是 int, str

字典 (dict) - *key : value* 的對應關係

語法:用 {} 表示字典, 內部儲存許多 key : value 關係, 每組關係以 , 隔開。

```
>>> CSIE = { 'Founded' : 1978, \
             'Activity' : ['資訊營', '之夜', '週'],\
             204 : '電腦教室' }
>>> CSIE['Building'] = '德田館'
>>> CSIE['Building']
>>> CSIE['Activity'][0]
>>> CSIE['Activity'].append('期末爆肝')
>>> CSIE['Activity']
['資訊營', '之夜', '週', '期末爆肝']
```

type() - 看變數的型別

```
>>> my_int = 100
>>> type(my_int)
<class 'int'>

>>> my_str = 'my str'
>>> type(my_str)
<class 'str'>

>>> my_tuple, wrong_tuple = (1, ), (1)
>>> type(my_tuple)
<class 'tuple'>
>>> type(wrong_tuple)
<class 'int'>
```

為什麼 "data type" 很重要

```
>>> int_1 = 1
>>> int_1 + int_1
2

>>> str_1 = '1'
>>> str_1 + str_1
'11'
```

不同的 data type 會有不同的指令和不同的結果

你需要了解該變數於程式中的意義, 設定合適的型態

Input / Output

輸出:從電腦裡送到螢幕上

語法:print(var_1, var_2, ...), 可以印出一堆變數

```
>>> my_int, my_list, my_dict = 1, [1,3,4], {1:'one', 2:'two'}
>>> print('my_int is', my_int)
my_int is 1
>>> print(my_dict)
{1: 'one', 2: 'two'}
```

輸入:從鍵盤打字傳給電腦

語法:input('[輸入提示]'), 讀進一串「字串」

```
>>> name = input('Please enter your name: ')
Please enter your name: Uvuvwevwevwe Onyetenyevwe Ugwemuhwem Osas
>>> name
'Uvuvwevwevwe Onyetenyevwe Ugwemuhwem Osas'
>>>
```

那如果想要讀進整數當計算機用怎麼辦 TAT

轉型 (casting)

語法:在需要轉型的變數外面包上 int(), float(), str(), list(), tuple() 等

```
a = int(input('Please input a: '))
b = int(input('Please input b: '))
c = int(input('Please input c: '))
print((-b + (b**2 - 4 * a * c)**0.5) / (2 * a))
print((-b - (b**2 - 4 * a * c)**0.5) / (2 * a))
```

Input

```
1
-1433
513372
```

你會做給別人用的一元二次方程式計算機了！

Exercise: 台北 fun 旅行

```
Station(0, 0)
```

```
Hotel(0.7, 1.6)
```

```
Park(1.5, -1.8)
```

```
Dertian(2.1, -3)
```

給定四個座標點, 輸入兩個地點, 輸出它們的距離

1. 建一個 `(location):(x,y)` 的 dictionary
2. 讀進兩個地點 `src, dest`, 表示起訖
3. 把 src, dest 當 key, 讀取對應地點的座標
4. 計算距離並印出來

Solution

```
location = {'Station': (0, 0), 'Hotel': (0.7, 1.6),\
            'Park': (1.5, -1.8), 'Dertian': (2.1, -3)}
src, dest = input(), input()
distance = ( (location[src][0] - location[dest][0]) ** 2 +
            (location[src][1] - location[dest][1]) ** 2 ) ** 0.5

print(distance)
```

上半場結束