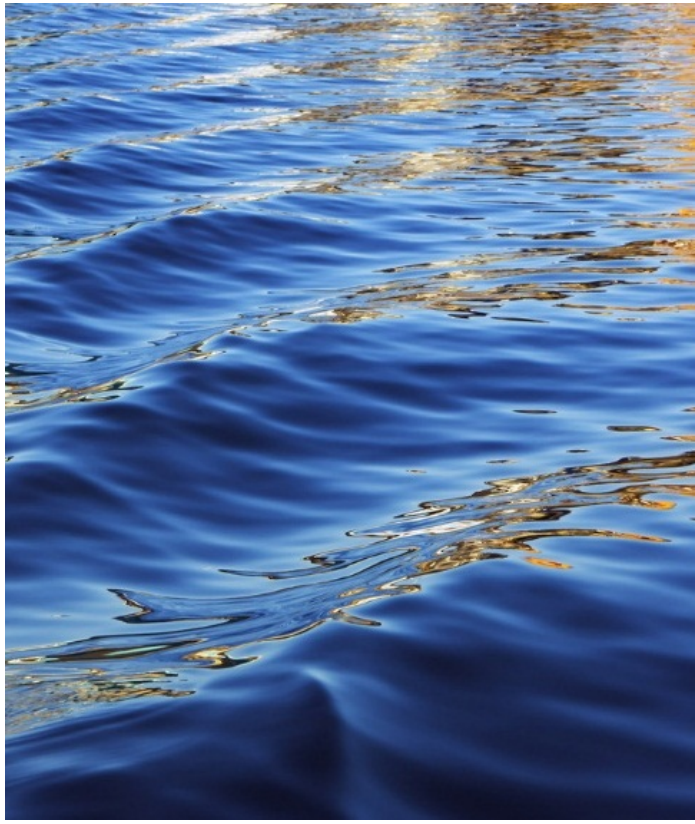
Python

Ricky

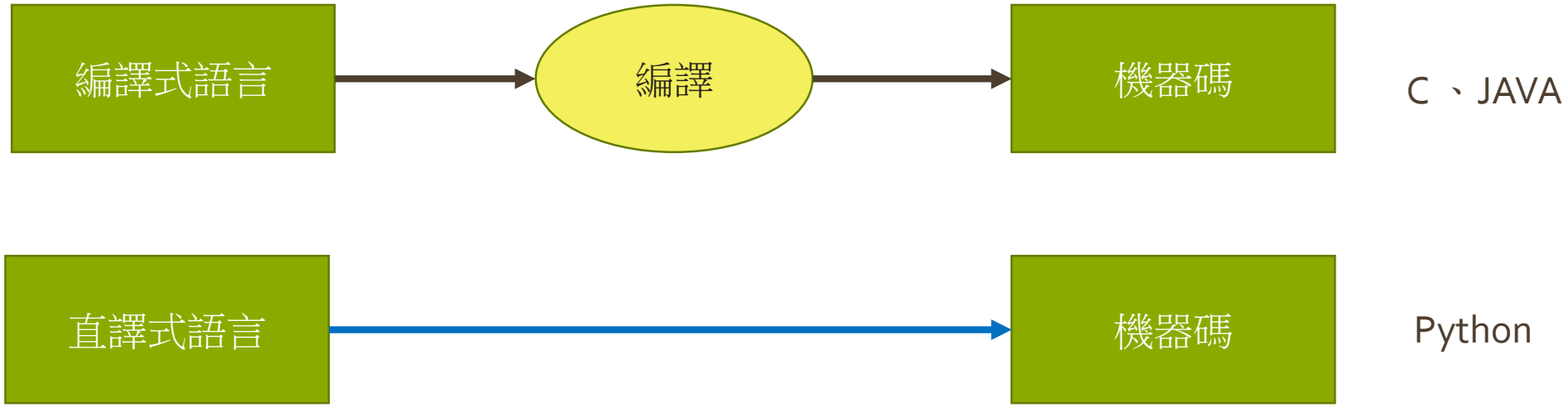
1.基本觀念

1. 認識Python
2. 應用範圍
3. Python的優缺點
4. 安裝Python



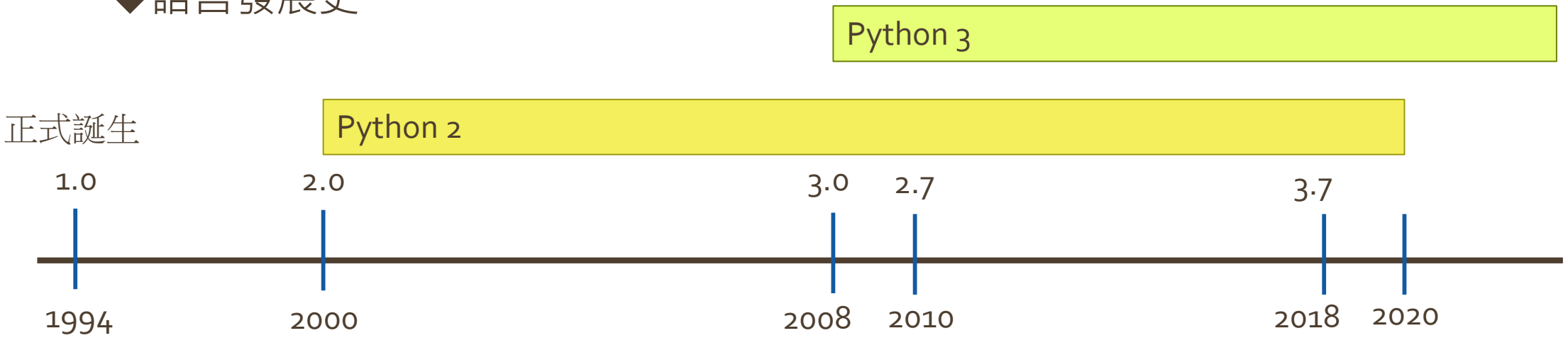
認識Python

- 直譯式語言、物件導向語言
  - 直譯式代表，直譯器(interpretor)會將程式碼一句一句直接執行，不會經過編譯(compile)動作，將語言先轉成機器碼再執行。
  - CPython，是由C語言編寫而成



認識Python

- ◆動態高階語言
  - ➢垃圾回收(garbage collection)，主動回收不需要的動態記憶體空間
- ◆開放原始碼(Open Source)
- ◆套件(Package)、模組(Module)非常多
- ◆語言發展史



應用範圍

- ✓設計動畫遊戲
- ✓支援圖形化介面(GUI,Graphical User Interface)
- ✓資料庫開發與設計動態網頁
- ✓科學計算與大數據分析
- ✓人工智慧與機器學習重要模組，ex TensorFlow、keres、Pytorch
- ✓Google、Yahoo!、Youtube、Instagram、NASA等公司功能開發主要語言
- ✓網路爬蟲、駭客攻防、搜尋引擎..等

Python的優缺點

◆Python的優點很多，簡單的可以總結為以下幾點。

- 簡單明了，學習曲線低，比很多編程語言都容易上手。
- 開放源代碼，擁有強大的社區和生態圈，尤其是在數據分析和機器學習領域。
- 解釋型語言，天生具有平台可移植性，代碼可以工作於不同的操作系統。
- 代碼規範程度高，可讀性強，適合有代碼潔癖和強迫症的人群。

◆Python的缺點主要集中在以下幾點。

- 執行效率稍低，對執行效率要求高的部分可以由其他語言（如：C、C++）編寫。
- 代碼無法加密，但是現在很多公司都不銷售賣軟件而是銷售服務，這個問題會被弱化。
- 在開發時可以選擇的框架太多（如Web框架就有100多個），有選擇的地方就有錯誤。

安裝Python

◆單純安裝Python

- 官方網站:　https://www.python.org/
- 檔案小，但繁瑣，套件安裝容易遺漏

◆安裝組合包 Anaconda

- 開源、免費以及跨平台
- 內含SpyderIDE 與Jupyternotebook 環境
- 支援Python 2.x, 3.x 與R 語言
- 額外的加速、優化要收費，但學術用途可以申請免費
- 檔案大，但統一窗口，套件安裝會連帶一起安裝

安裝Python

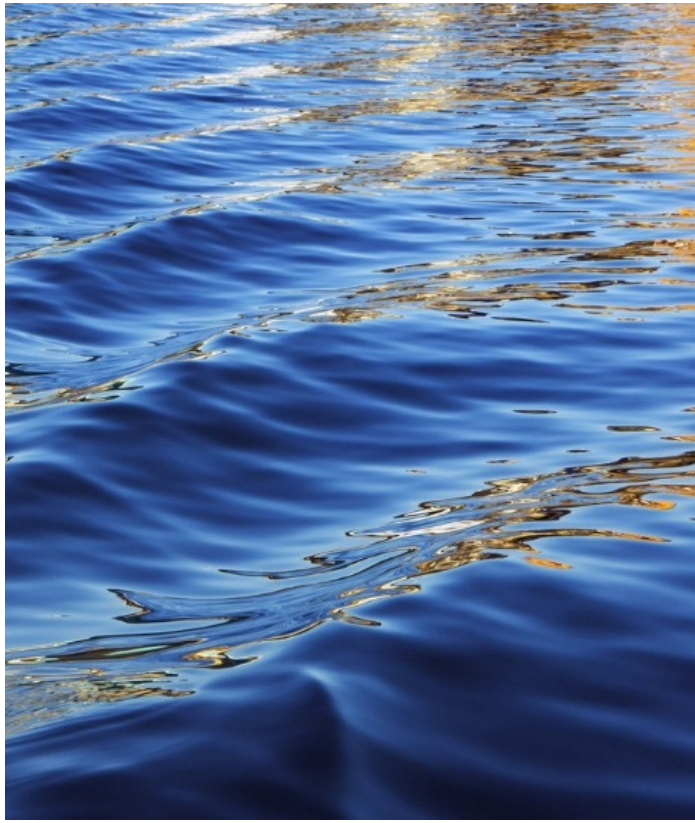
◆Anaconda 平台

➢Spyder 圖形化編輯器

➢Jupyter Notebook 線上編輯器

2.認識變數

1. 用Python計算
2. 認識變數
3. 認識變數地址
4. 認識程式
5. 變數命名規則



用Python計算

- 假設到7-11打工，一小時120元時薪，如果想計算一天八小時的薪水，可以得到多少呢?
- 計算機大家都會按!那Python 怎麼用呢?

```
In [1]: 120 *8
Out[1]: 960
```

- **Q:那365天的總共薪水怎麼算呢?**
- **Q:假設因為表現優秀每小時加薪20元，而每個月花掉3000元的手機費用，與日常支出10000元，那請問一年可以存到多少呢?**

認識變數

- 變數是一個暫時儲存資料的地方，如剛剛計算，你會發現當要計算新的算式時，都要重新計算。所以為了解決這個問題就需要建立一個變數來儲存你要計算的時薪。
- 在Pyhton中 可以用 “= “ 來賦予變數內容，在這個例子我們用變數x，設定為時薪

```
In [3]: x=120

In [4]: print(x)
120
```

- 加薪可以這樣表示

```
In [5]: x = x+20

In [6]: print (x)
140
```

認識變數

- 所以一天8小時，365天的年薪可以寫成如下，

```
In [9]: x=120

In [10]: y= x*8 *365

In [11]: print(x)
120

In [12]: print(y)
350400
```

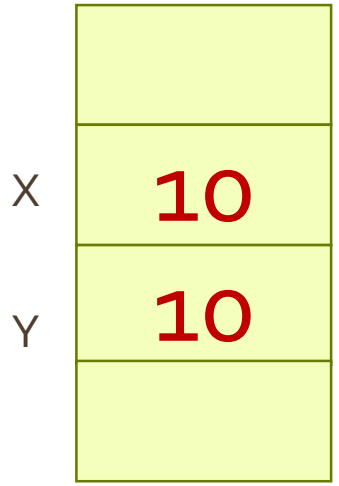
認識變數

- 從剛剛的設計可以很清楚的知道，我們可以用變數來當作暫時儲存資料的地方。
- 所以
  - **時薪:hourly_salary 來代替x**
  - **年薪:annual_salary 來代替y**
  - **支出:expenses 可以來代替 每個 月的消費變數**
  - **積蓄:accumulation 可以代替一年存下來的積蓄**

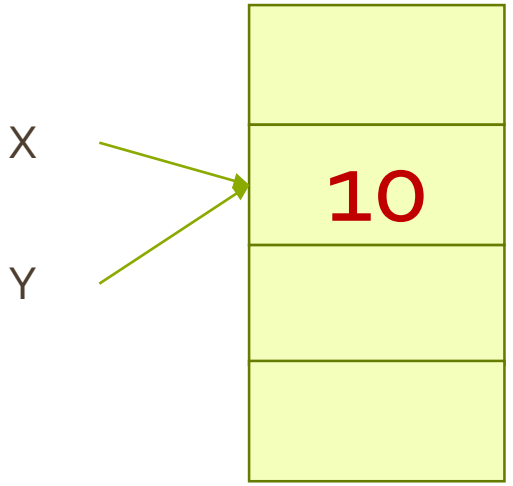
- **Q:請重新將一開始的打工，加薪，支出，存下來的積蓄重新設定一次。**

認識變數地址

- Python是一個動態語言，她處理變數的觀念與一般語言不同，對於靜態語言，如C、C++當宣告時記憶體會預留空間給變數去儲存。
- 但在Python中變數是參照地址的觀念，當設定變數x = 10時， Python會在記憶體某地址內儲存10，而這時x 就是一個標誌(tag)，標誌內容就是儲存10的記憶體位置(是不是就像之前說的指標呢?)，若設定了y =10 ，那y的標誌內容也是儲存10的記憶體位址。



C、C++



Python

認識變數地址

- Python可以使用id()函數，來取得變數的地址

```
In [14]: x=10

In [15]: y=10

In [16]: z=15

In [17]: id(x)
Out[17]: 140705203200688

In [18]: id(y)
Out[18]: 140705203200688

In [19]: id(z)
Out[19]: 140705203200848
```

認識程式

- 剛剛這樣計算完，發現如果要改變時薪、改變天數、改變消費，每次都要重頭再設定，非常麻煩。
- 所以我們可以將剛剛那些計算儲存在一個檔案內，這個檔案就叫做**程式**
- **Q:請將剛剛打的東西，打在IDE上，打工一小時時薪120元，365天的年薪，扣掉每個月消費的手機費3000元與日常支出10000元，那剩下的積蓄是多少?**
  - **Tip:最後顯示請用print(變數)，做顯示**

```python
hourly_salar =120
annual_salar = hourly_salar *8 *365
expenses= 3000+10000
accumulation = annual_salar - (expenses*12)
print(accumulation)
```

名称	类型	大小	
accumulation	int	1	194400
annual_salar	int	1	350400
expenses	int	1	13000
hourly_salar	int	1	120

```
In [1]: runfile('C:/Users/Chen/.spyder-py3/Test1.py', wdir='C:/Users/Chen/.spyder-py3')
194400
```

認識程式

- 初步建立一個小程式後，我們也設計了正確名稱的變數
- 但……但……但……十年後……你還記得嗎?
- **所以我們要幫程式 加上 筆記，也就是 上註解(comment)**
- **註解的好處是提高閱讀性與增加除錯便利性**

- **非常重要**
- **非常重要**
- **非常重要**

認識程式

- **單行註解符號 #**
- **在Python中 一旦文字前面加上了# 符號，程式中就會自動忽略 #右邊那行所有的文字意義。**

```
In [2]: print("今天開始入門了!!!!") # 註解說明
今天開始入門了!!!!
```

- **三個單引號 ‘ 或雙引號 “ 包起來的上下兩行 ，就是做為多行註解**

```
'''
    早餐沒吃飽
    好想睡覺
'''
```

```
"""
    開始入門的註解說明
    今天下班下大雨非常哀傷
"""
```

認識程式

- 所以加上註解後的程式碼是這樣子

```
"""
Created on Thu Jul  2 00:49:55 2020

@author: Chen

@version: 0.01

@brief : calculate my salary and expenses for a year


"""

# 時薪
hourly_salar =120
# 年薪
annual_salar = hourly_salar *8 *365
# 支出 月
expenses= 3000+10000
# 積蓄 年
accumulation = annual_salar - (expenses*12)
#列印出來積蓄
print(accumulation)
```

變數命名規則

- 在Python中，變數命名需要遵循以下這些必須遵守硬性規則和強烈建議遵守的非硬性規則。
- **硬性規則：**
  - 必須由字母（廣義的Unicode字符，不包括特殊字符）、數字和下底線構成，數字不能開頭。
  - 大小寫敏感（大寫的a和小寫的A是兩個不同的變量）
  - 不要跟關鍵字（有特殊含義的單詞，後面會講到）和系統保留字（如函數、模塊等的名字）衝突

系統保留字

and	elif	import	raise	global
as	else	in	return	nonlocal
assert	except	is	try	True
break	finally	lambda	while	False
class	for	not	with	None
continue	from	or	yield	
def	if	pass	del	

變數命名規則

Q:請說明以下變數是否合乎規格

1. Sum,1
2. 3y
3. SUM
4. _fillg
5. X$3
6. X_5
7. And
8. Joe_Mary
9. Joe&Mary

Q:何者變數為同一個變數

1. Sum
2. sum
3. SUM

變數命名規則

- 非硬性規則-PEP 8 為大多數Python Coding Style
- 變數名稱用小寫
- 兩個英文文字之間用底線連接
- 執行運算時，運算符號左右會空格
- 可參考 http://python.org/dev/peps/pep-0008


Python »» Python Developer's Guide »» PEP Index »» PEP 8 -- Style Guide for Python Code

PEP 8 -- Style Guide for Python Code

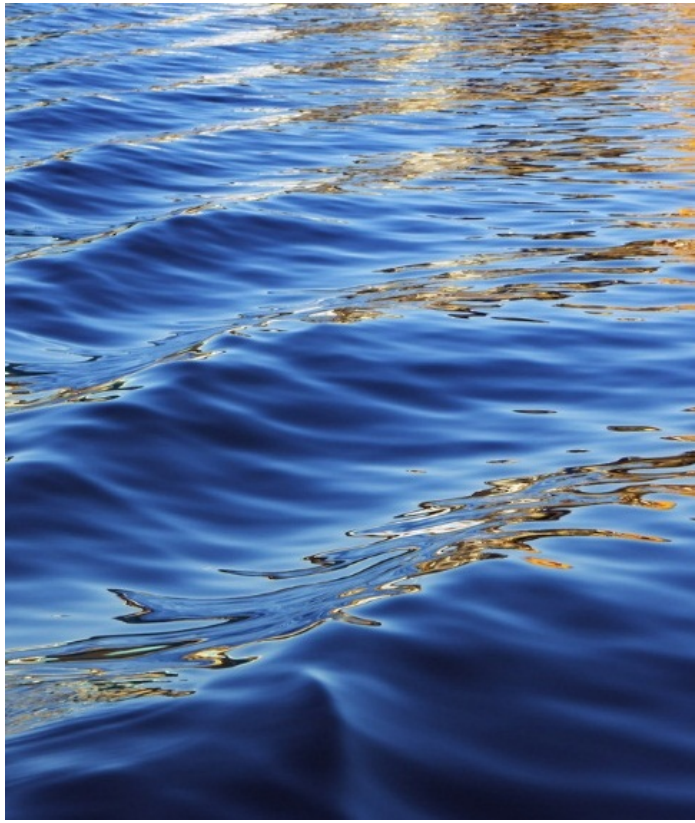
PEP:	8
Title:	Style Guide for Python Code
Author:	Guido van Rossum <guido at python.org>, Barry Warsaw <barry at python.org>, Nick Coghlan <ncoghlan at gmail.com>
Status:	Active
Type:	Process
Created:	05-Jul-2001
Post-History:	05-Jul-2001, 01-Aug-2013

變數命名規則-底線開頭與結尾的變數

1. 變數名稱有前單底線，例如 _test
   - 這是一種私有變數、函數或方法，可能是在測試中或一般應用中不想被直接引用的變數，可以利用此方式命名
2. 變數名稱有後單底線，例如 dict_
   - 這種方式主要是避免與Python同名的關鍵字與內建函數互相衝突
3. 變數名稱前後有雙底線，例如 __test__
   - 這是保留給Python內建的變數或方法使用
4. 變數名稱前有雙底線，例如 __test
   - 這也是私有方法或變數的命名，無法直接使用本名存取

3.基本數學運算

1. 四則運算
2. 餘數與整除
3. 次方
4. 優先順序
5. 指派運算子
6. 多重指定
7. 刪除變數
8. 斷行



四則運算

- 加 +
- 減 –
- 乘 *
- 除 /

```
In [6]: x = 3 + 6

In [7]: print(x)
9

In [8]:  y = x - 10

In [9]: print (y)
-1
```

```
In [10]: a = 10 * 5

In [11]: print (a)
50

In [12]: b = a / 2

In [13]: print (b)
25.0
```

餘數與整除

- 餘數(mod) % ，可計算除法運算中的餘數
- 整除 //，只保留整除的商，去掉餘數

```
In [14]: x = 9 % 5

In [15]: print (x)
4

In [16]: y = 9 // 2

In [17]: print (y)
4
```

次方

- 次方的符號是 **

```
In [19]: x = 3 **2

In [20]: print(x)
9

In [21]: y = 2 ** 3

In [22]: print(y)
8
```

優先順序

- 當一堆運算符號出現在程式中，**括號( ) 永遠都是第一順位**，由左至右。
- 優先順序如下
  1. **次方**
  2. **乘法、除法、求餘數、求整數**
  3. **加、減**

```
In [24]: print(x)
86

In [25]: y = 5 + 6 * 8 -2

In [26]: print(y)
51

In [27]: z  = 2* 3**3 * 2
```

答案是甚麼?

指派運算子

運算子	意義	範例	範例結果
+=	相加後再指定給原變數	i += 5	15
-=	相減後再指定給原變數	i -= 5	5
*=	相乘後再指定給原變	i *= 5	50
/=	相除後再指定給原變數	i /= 5	2
%=	相除得到餘數後再指定給原變數	i %= 5	0
//=	相除得到整除商數後再指定給原變數	i //= 5	2
**=	做指數運算後再指定給原變數	i **= 3	1000

多重指定

- 在Python 指定變數可以一次設定多個變數數值

```
In [30]: x = y = z = 10

In [31]: print (x)
10

In [32]: print (y)
10

In [33]: print (z)
10
```

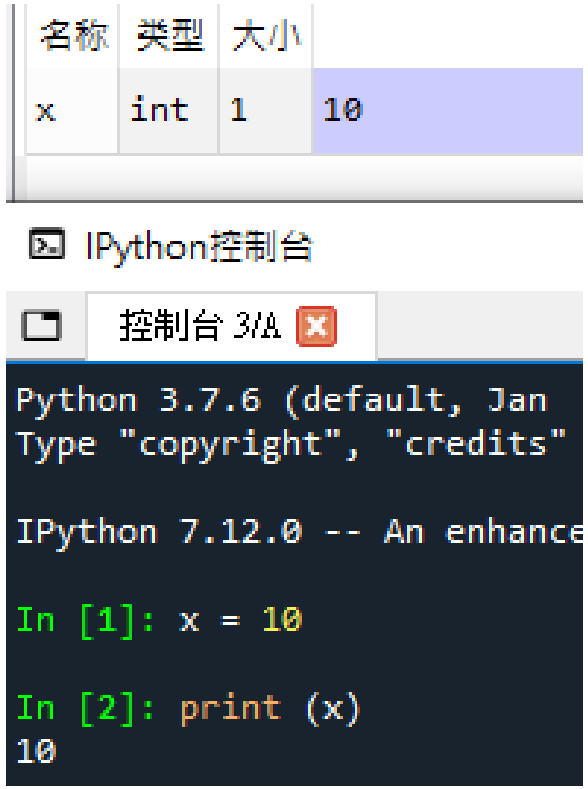
```
In [34]: x ,y ,z = 1 , 3 ,5

In [35]: print (x)
1

In [36]: print (y)
3

In [37]: print (z)
5
```

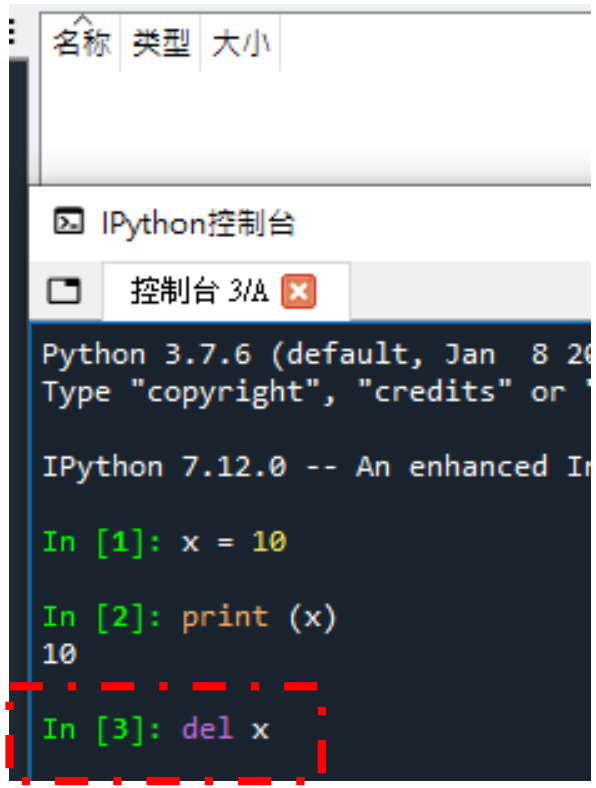
刪除變數

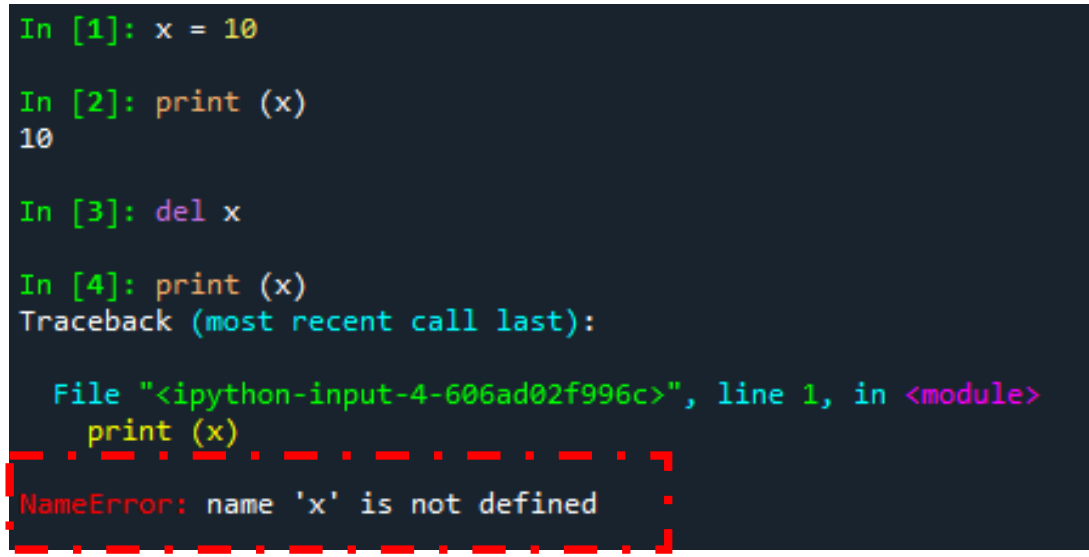
- 程式設計時，如有些變數不再需要，可以使用 del 指令來將此變數刪除

  del　　變數名稱

- 實驗，定義完X變數後，又移除，會怎樣呢?







斷行

- 在設計一個大型程式中很長會出現要寫一個落落長的程式，需要分成2行以上來說明，可以在要斷行的地方 **加上 \ **，PYTHON就會在程式直譯中自動續行
- 也可使用**括號 ( )**

```
a = b = c = 10
x = a + b + c + 12
print(x)
# 續行方法1      # PEP 8風格
y = a \
    + b \
    + c \
    + 12
print(y)
# 續行方法2      # PEP 8風格
z = ( a          # 此處可以加上註解
      + b
      + c
      + 12 )
print(z)
```

底數

- 在程式中，有時候會出現2進位、10進位、16進位。那我們要如何表示呢?
- 前置底數 0b 代表 告訴電腦後面為 2進位
- 前置底數 0x 代表告訴電腦後面為 16進位

```
In [7]: 0b11
Out[7]: 3

In [8]: 0x11
Out[8]: 17
```

```
In [9]: 0b99
  File "<ipython-input-9-3f11e18b3a4b>", line 1
    0b99
     ^
SyntaxError: invalid token


In [10]: 0xZZ
  File "<ipython-input-10-e20770b2de7d>", line 1
    0xZZ
     ^
SyntaxError: invalid token
```

HOMEWORK

請針對下列兩題，各寫一隻程式，並存成 兩個 .py的檔案，檔名為HW3_1，HW3_2

1. 圓形半徑 5 公分，試求 園面積 與 圓周長
    1. Pi = 3.14159

2. 請將 華式溫度 30度 轉成 攝氏溫度

3. 請列舉出 y = 3 * 5 / 2 的2進位寫法與16進位寫法，並給出結果