Python輕鬆上手學

目錄

目錄

Python簡介與環境設定

Python語言

- 動態語言(Dynamic Language)
  - 於執行期間(Runtime)執行程式碼(不需編譯)
  - Dynamic Type：函式與變數都不需要宣告型態
- 直譯式語言(Interpreted Language)
  - 每次執行後可以直接看到結果
- 物件導向語言(OOP)
- 可執行於多平台(Python VM)

Python如何執行

- 原始碼
  - Code.py
  - 使用者的程式
- 位元碼
  - Code.pyc
  - Python編譯器，會轉成位元碼加速執行
- 執行期
  - PVM
  - 由Python Virtual Machine執行位元碼

Python 2.x v.s. Python 3.x

- Python 2.x is legacy,Python 3.x is the present and future of the language.
- Python 3.x
  - 更簡潔的語法，Unicode支援與強大函式庫
  - 尚在持續更新與開發中
- Python 2.x
  - 支援主要作業環境
  - 第三方函式庫主要支援版本

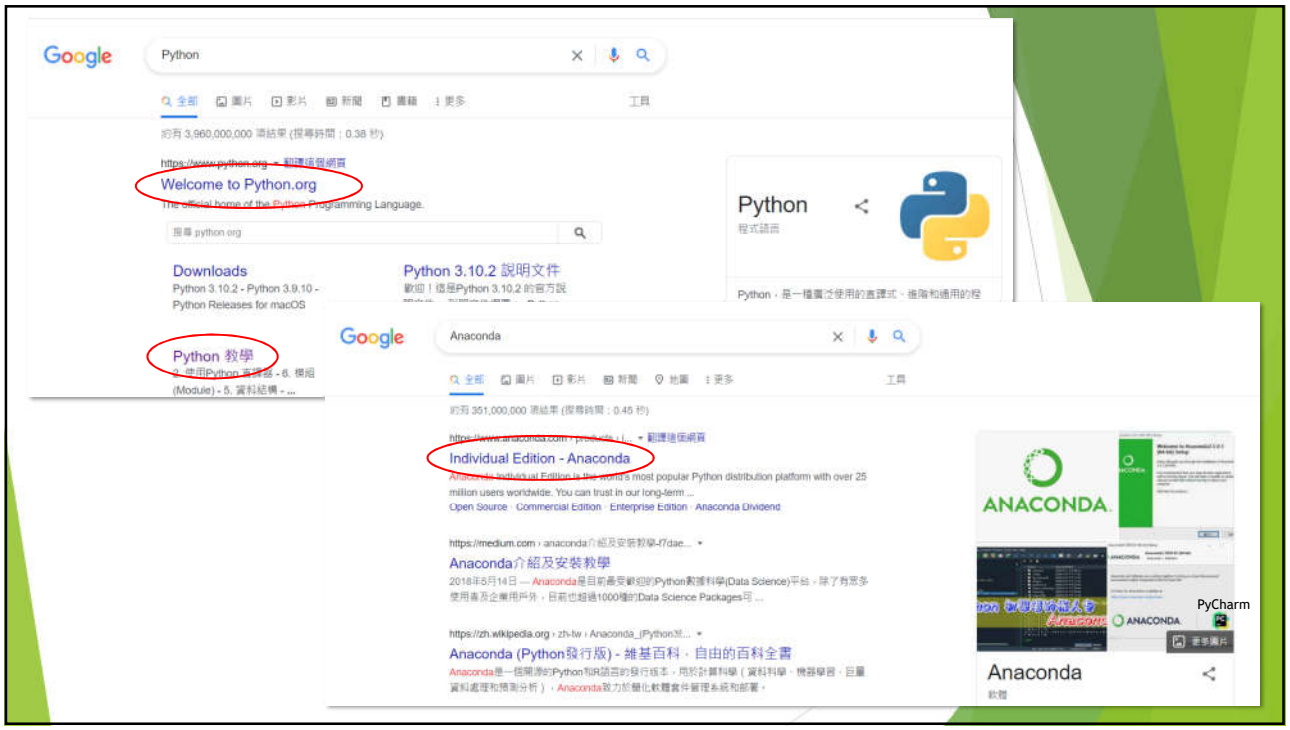
Python v.s. Java

Java

- 執行速度較Python為快
- 使用{}分隔區塊
- 需要宣告變數型態
- 可以透過Compiler檢查錯誤
- 使用/**/做註解

Python

- 開發速度較快
- 使用indent替代{}
- 不需宣告變數型態
- 只能在runtime檢查錯誤
- 以#與”或””做註解



目錄

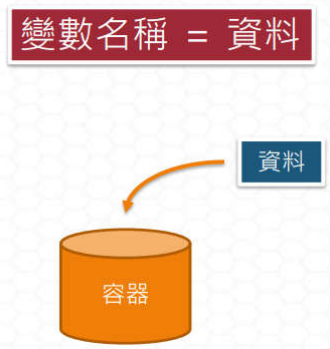
- Python簡介與環境設定
- Python物件與資料結構
- Python運算式與陳述
- Python方法與函式
- Python物件導向程式設計
- Python模組與套件

Python物件、資料型態與資料結構

基本算術功能

- a = 5
  a + a

- # 重新賦值
  a = 10
  a
  # 重新定義 a
  a = a + a
  a



進階算術功能

- 指數運算
  - 2**3
- 開根號
  - 4**0.5
- 先乘除後加減
  - 2+10*10+3
- 使用小括號指定運算順序
  - (2+10)*(10+3)

整數、浮點數

- A=2
- type(A) #善用type()檢查型態

- str='Python'
- A+str

- type(A*3.14)
- float('3.2')

字串

- # 使用單引號標註一段話
  'This is a string'
- 'I'm using single quotes, but will create an error'

  "Now I'm ready to use the single quotes inside a string!"

- a="Hello“
- a[0]
- a[:2]

- dir(a)
- a.upper() #善用鍵盤”Tab”鍵
- a.split('A')
- a.split('a')
- type(a.split('a'))

Python清單(List)

有沒有一個容器能同時記錄多個人的資料?

- 電話
- 姓名
- 性別

- 前面介紹的數值與字串在容器均為單一值
- 如果要存放多個值時，可以用什麼資料結構？
  - List
- 其它的程式語言都有陣列的概念(Array)；Python的List比較接近於Java語言的ArrayList
- 宣告List 的方式
  a = []

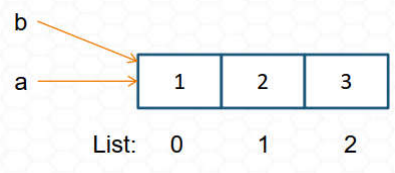
- list=[“1”,2.5,3] #List資料結構可以裝載不同類型的元素(不同資料型態的資料)
- type(list)
- dir(list)
- print(list[-2]) #indexing
- a=[5,6,7,8]
- a.pop()
- a.append(2)
- a.sort()
- a.reverse() #print(a)

- a = [1, 2, 3] #指定(Assignment)陣列
  b = a
  a[1] = 2000
  a

[1, 2000, 3]

b

[1, 2000, 3]



- 複製List - 解決方法
- a = [1, 2, 3]
  import copy
  b = a.copy() # Deep Copy
  a[1] = 2000
  b

Python字典(Dictionary)

- 其它語言有相同操作
  - Java：HashMap,HashTable…
  - C++：hashmap
  - C#：Dictionary…
- **dic={ key : value }**
- **其中，key 為唯一不重複值**

```
dic={"a":10,"b":"OK","c":0.25}
print(dic)
print(dic.keys())
print(dic.values())
print(dic["a"])
print(dic.get("b"))  #預設值 (dir(dic))
```

Python元組(Tuple)

- **跟List不同，在Tuple內的值無法被更改**
  - 在記憶體中有固定大小
- 建立Tuple的方式
  - tuple1=(1,2,3)
  - tuple2=1,2,3
  - tuple3=tuple([1,2,3])
- 檢視tuple跟list的差異
  - dir(tuple)
  - dir(list)

當希望資料能維持一致性，而內容值不能被修改時(例如資料庫回傳資料)，可以使用Tuple。

目錄

Python運算式與陳述

陳述式

- 其他語言的陳述式
  - if(a>b){
    a=2;
    b=4;
    }
  - 使用{}表述陳述式
  - 使用；表示指令的結尾
- Python語言陳述式
  - if a > b:
    a=2
    b=4
  - **使用：表示陳述式**
  - **使用indent (使用Tab或空白區隔程式執行區塊)**

迴圈與控制流程

- FOR與IF-ELIF-ELSE迴圈控制巢狀流程

```
a = 9
b = 8
for i in range(1,10):   # i from 1 to 9  (i.e., for(i=1; i<10; i++){})
    if i == a:
        print(str(i) + ' found!')
        break
    elif i == b:
        print str(i) + ' found'
    else:
        print(str(i) + ' was not in the list')
```

TypeError: 'str' object is not callable

目錄

Python方法與函式

Python函式

- Python**物件導向語言(OOP)**
- 在Python中，**任何東西都是物件**
- **每個物件都有自己的函式(方法)，封裝對物件的操作，減少使用者花費時間撰寫重複的程式碼**
- 希望撰寫程式一次，但在不同情境下可以重複呼叫該程式好幾次
  - 遞迴
  - Python 定義函式的語句: def

比較Lambda與函式

一般函式

```
def square(num):
    result=num**2
    return result
print(square(2))

def square(num):
    return num**2
print(square(2))

def square(num): return num**2
print(square(2))
```

Lambda函式

```
square=lambda num:num**2
print(square(2))
```

範圍搜尋順序

LEGB規則

- Local
- Enclosing functions
- Global(模組)
- Built-in(Python)

```
x=20
def func(x):
    print("x is",x)
    x=10
    print("Change local x to",x)

func(x)
print("x is still",x)
```

```
x=30
def func():
    global x
    print("This function now is using the global x!")
    print("Because of global x is",x)
    x=5
    print("Run func(),change global x to",x)

print("Before calling func(),x is",x)
func()
print("Value of x (outside of func()) is",x)
```

目錄

Python物件導向程式設計

不變性與可變性

- 整數、浮點數、字串、Tuple 是不可變
- 當物件不再被使用時，Python 會自動做垃圾收集(Garbage Collection)釋放記憶體空間
- 串列(List)與字典(Dictionary)屬於可變型態
- 兩組可變的資料，即使資料相同，則會對應到各自的id
- 兩組不可變的資料，資料內容相同時，會對應到同id

- **class**
  - 使用者可以使用**class**定義自己的**類別**
- 物件包含**屬性(attributes)** 與**方法(methods)**
  - 屬性: 物件的特性
  - 方法: 對物件的操作
    例如: 可以建立一隻”狗”的物件，包含”品種”為其屬性，”叫” 則為狗的方法

特殊方法

- 每個物件都有基本方法如：__init__()，__str__()，__len__()以及__del__()
- 特殊方法通常都以底線(e.g. __init__)包覆
- 在定義類別時可以修改這些特殊方法

目錄

Python模組與套件

模組

- **Python只是以 .py 做為副檔名的Python程式**
- **通常模組內包覆多組函式可供使用者使用**
- **使用者可以透過import指令匯入模組**
- 所有模組只會被載入一次，重複import不會導致函式重複被宣告

套件

- **套件就是目錄而已**
- **必須在目錄下加上__init__.py(空檔案)來啟用套件**
- 如果要限制引用模組，則在__init__.py中使用__all__即可限制匯入

Pip操作

- 套件安裝
  - pip install requests
- 套件升級
  - pip install –U requests
- 套件移除
  - pip uninstall requests
- 搜尋套件
  - pip search requests
- 列出pip說明
  - pip help
- 列出所有已安裝的套件
  - pip freeze

https://pypi.org/

END

NumPy

- 讓兩個List元素進行相乘?

```
a = [1, 3, 5, 7, 9]
b = [2, 4, 6, 8, 10]

print zip(a, b)
[(1, 2), (3, 4), (5, 6), (7, 8), (9, 10)]

c = [i*j for i, j in zip(a, b)]
print c
[2, 12, 30, 56, 90]
```


```
import numpy as np
na = np.array(a)
nb = np.array(b)

nc = na * nb

print nc
[ 2 12 30 56 90]
```

- na = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])
- na[0, 0]
- na.T

```
array([[ 1, 6],
       [ 2, 7],
       [ 3, 8],
       [ 4, 9],
       [ 5, 10]])
```

Pandas

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa