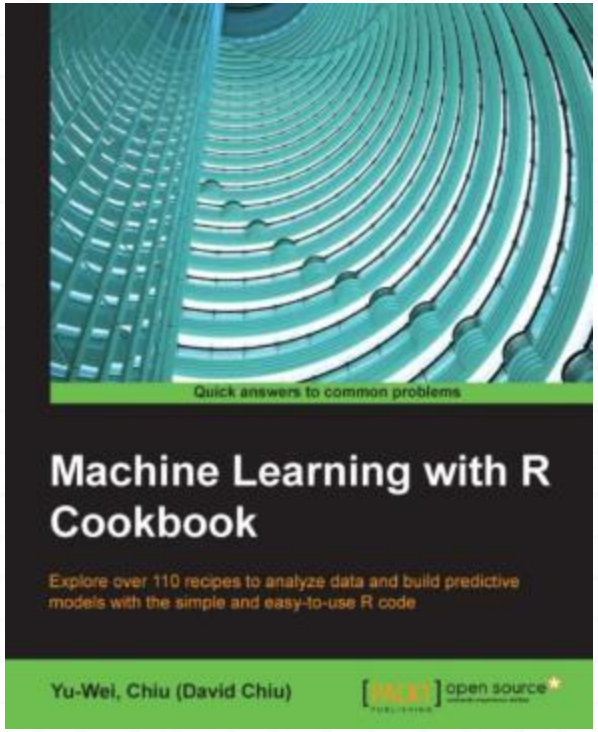
Python 基礎

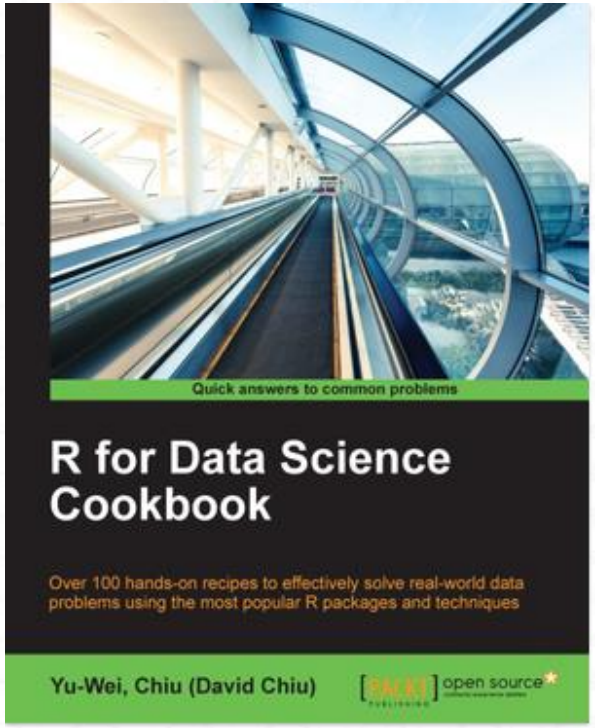
David Chiu

關於我

- 大數軟體有限公司創辦人
- 前趨勢科技工程師
- ywchiu.com
- 大數學堂 http://www.largitdata.com/
- 粉絲頁 https://www.facebook.com/largitdata
- R for Data Science Cookbook
  https://www.packtpub.com/big-data-and-business-intelligence/r-data-science-cookbook
- Machine Learning With R Cookbook
  https://www.packtpub.com/big-data-and-business-intelligence/machine-learning-r-cookbook

Machine Learning With R Cookbook (机器学习与R语言实战) & R for Data Science Cookbook (数据科学：R语言实现)





Author: David (YU-WEI CHIU) Chiu

課程資料

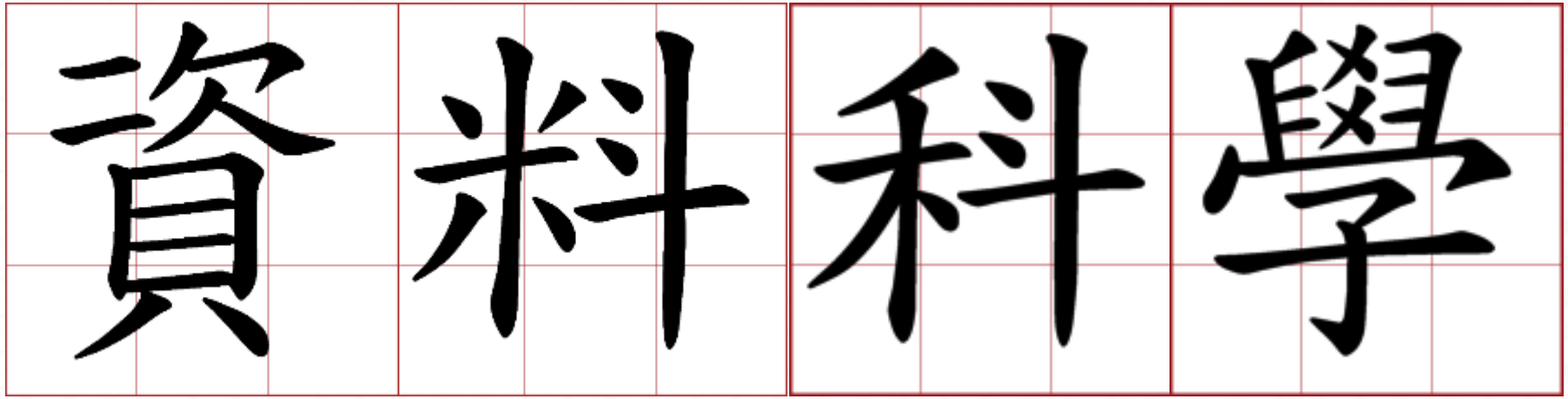
- 所有課程補充資料、投影片皆位於
  - https://github.com/ywchiu/ctbcpy

Python語言與資料分析

AlphaGO
使用深度學習技術打敗頂尖棋手


Tesla
讓自動駕駛不再是夢想


人工智慧
取代重複性高的工作
創造更多想像與新可能



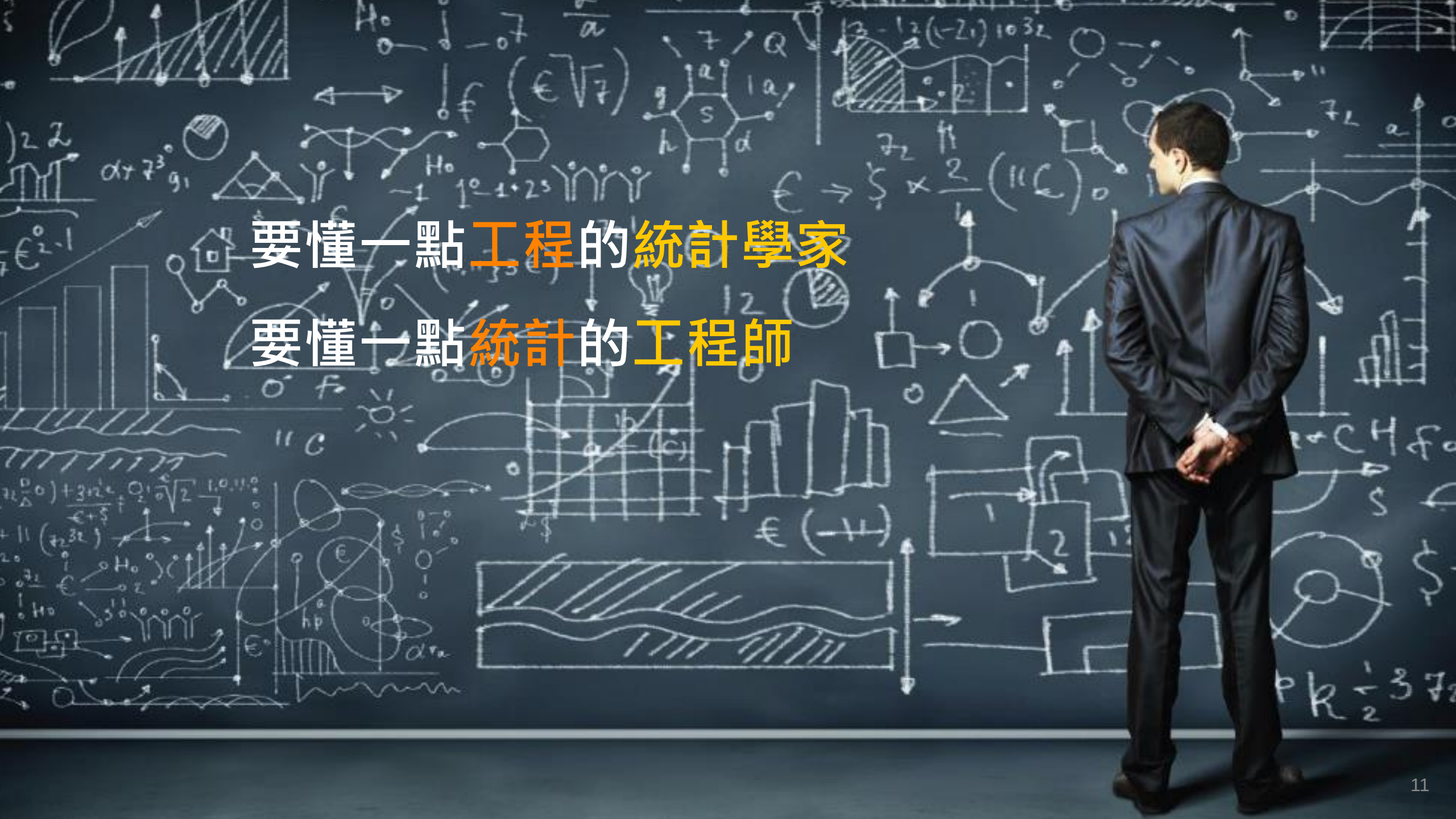
從資料鑒往

從資料知來

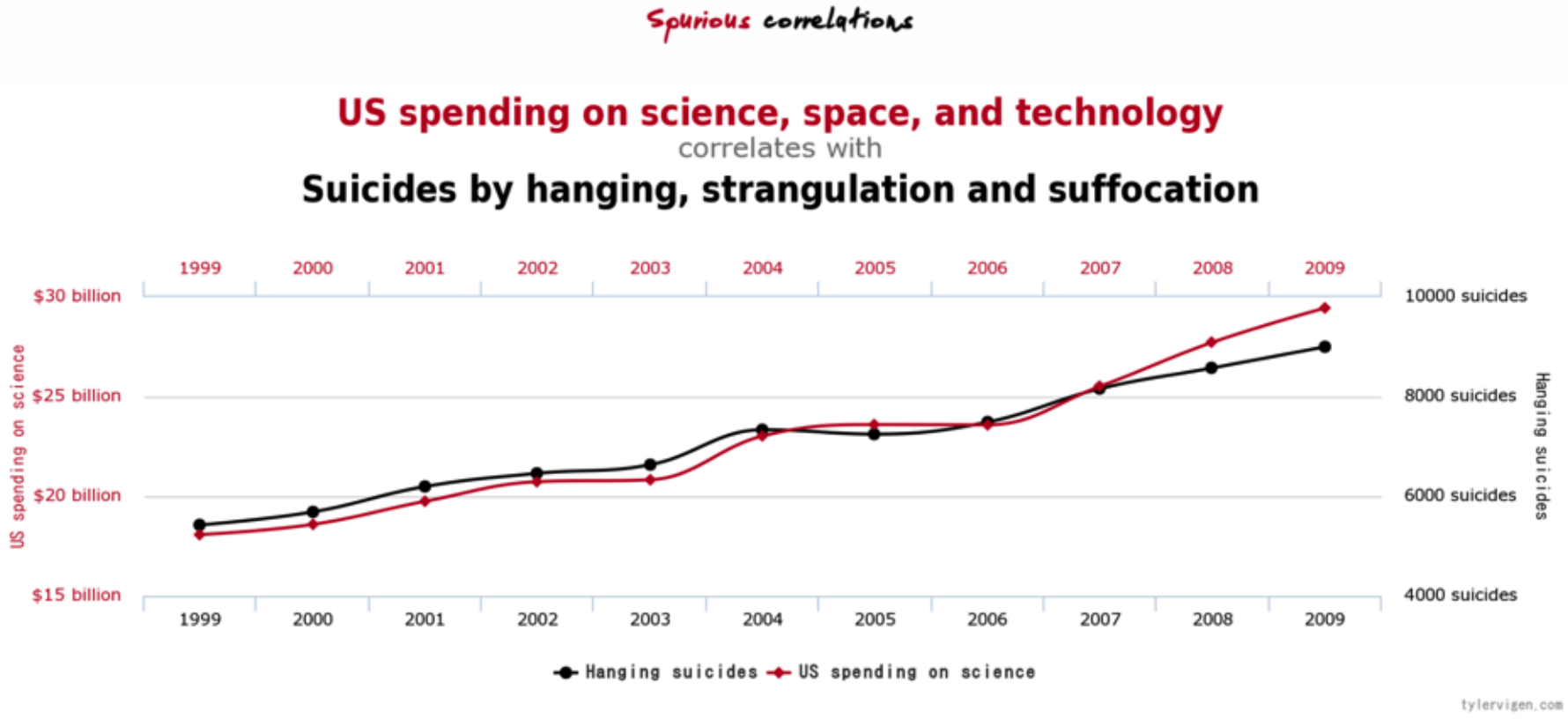
加一點數學統計

加一點工程

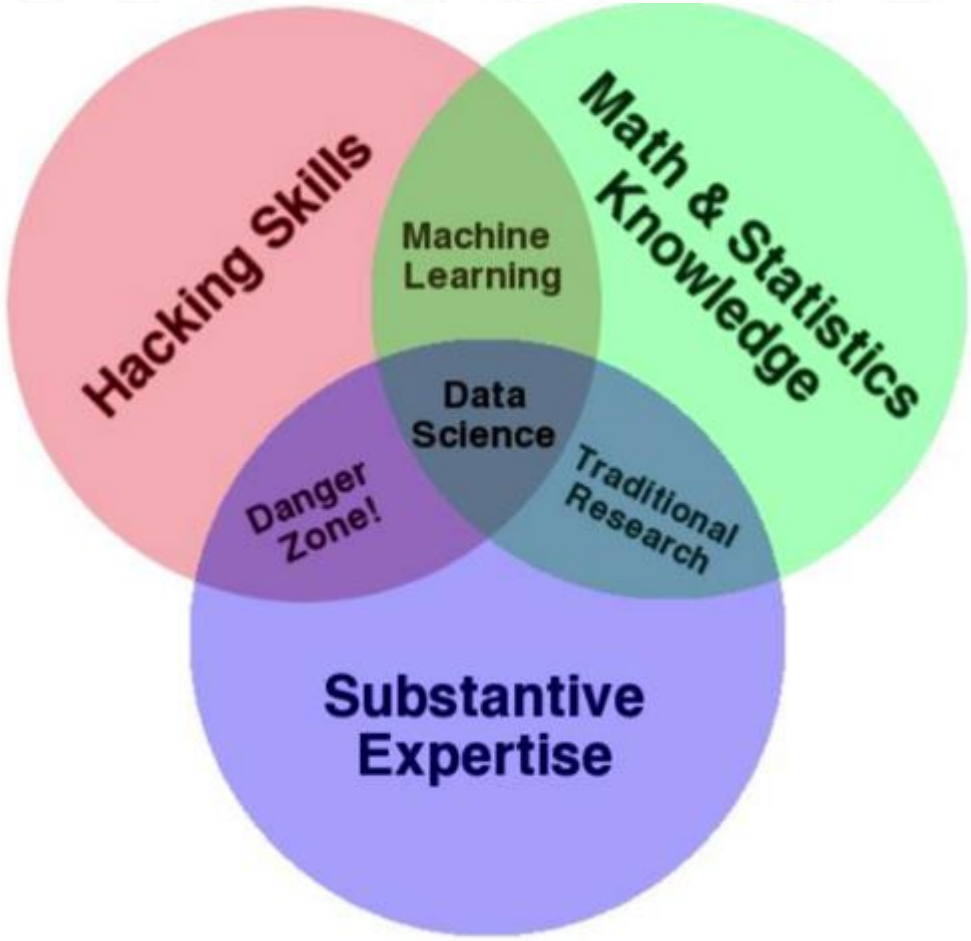
產生資料科學


要懂一點工程的統計學家
要懂一點統計的工程師

只有工程統計還不夠


Spurious correlations
US spending on science, space, and technology
correlates with
Suicides by hanging, strangulation and suffocation
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
$30 billion
$25 billion
$20 billion
$15 billion
US spending on science
10000 suicides
8000 suicides
6000 suicides
4000 suicides
Hanging suicides
Hanging suicides
US spending on science
tylervigen.com

資料科學


Hacking Skills
Math & Statistics Knowledge
Machine Learning
Data Science
Danger Zone!
Traditional Research
Substantive Expertise

資料科學能力

- 統計 (Statistic)

  單變數分析、多變數分析、變異數分析

- 資料處理 (Data Munging)

  抓取資料、清理資料、轉換資料

- 數據視覺化(Data Visualization)

  圖表、商業智慧系統

資料科學家主要工作內容

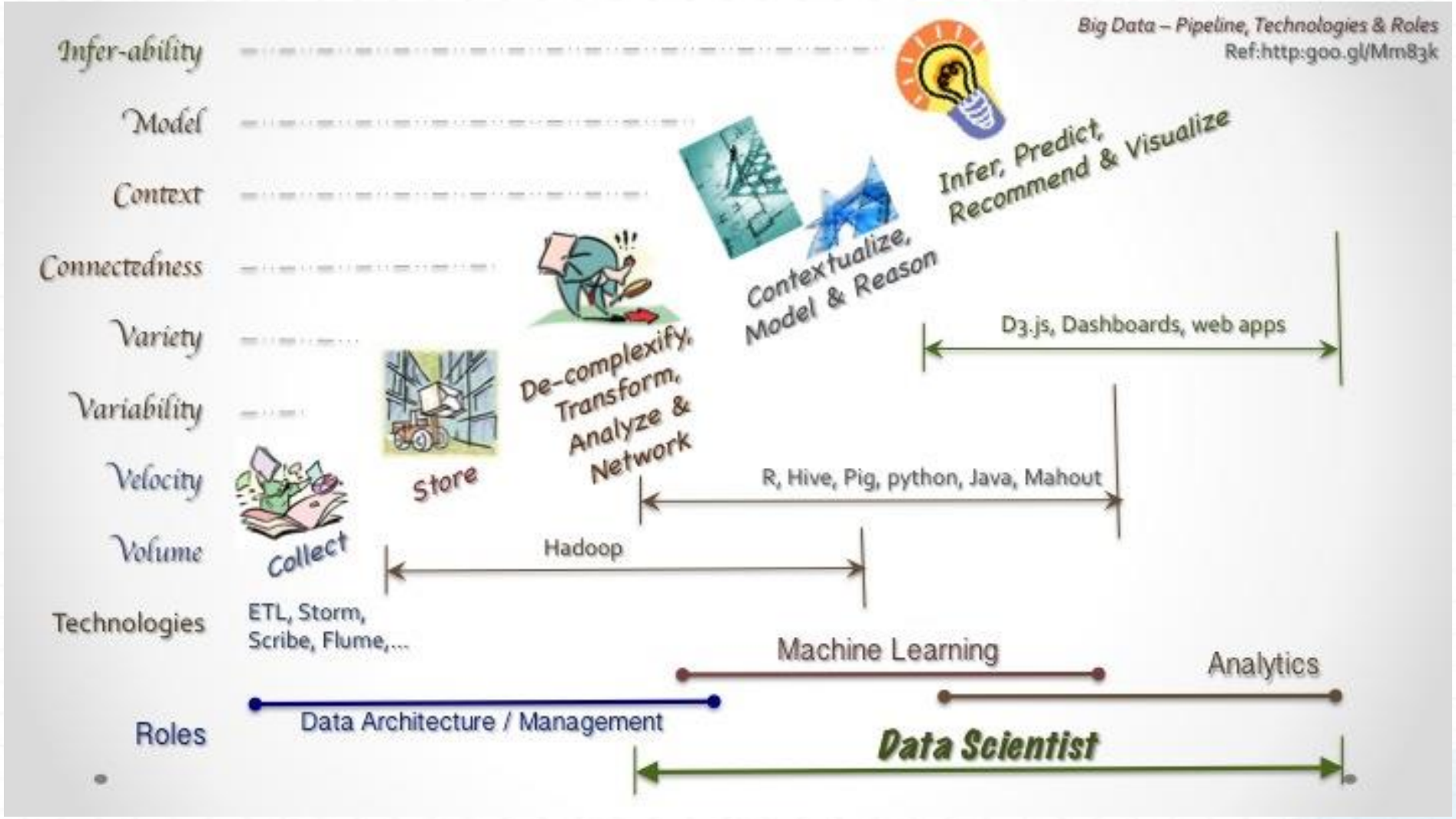
“80% 都在做加總與平均”

工作內容

1. 資料處理 (Data Munging)
2. 資料分析 (Data Analysis)
3. 詮釋結果 (Interpret Result)

真正能用在資料分析的時間很少，必須要能善用工具

資料科學分析步驟


Big Data – Pipeline, Technologies & Roles
Ref:http:goo.gl/Mm83k
Infer-ability
Model
Context
Connectedness
Variety
Variability
Velocity
Volume
Technologies
Roles
Infer, Predict, Recommend & Visualize
Contextualize, Model & Reason
De-complexify, Transform, Analyze & Network
Store
Collect
D3.js, Dashboards, web apps
R, Hive, Pig, python, Java, Mahout
Hadoop
ETL, Storm, Scribe, Flume,...
Machine Learning
Analytics
Data Architecture / Management
Data Scientist

資料分析工具

Python 語言

Python 語言

- 動態語言 (Dynamic Language)
  - 於執行時期(Runtime)執行程式碼 (不用編譯)
  - Dynamic Type: 函式與變數都不需要宣告類型
- 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)
- 可執行於多平臺 (Python VM)

Python 語言優點

- 可執行於多平臺 (Python VM)
- 擁有相當多的協力廠商資源
  (資料分析、圖形介面、網頁開發)
- 語法簡潔，編寫快速

簡單易用

JAVA

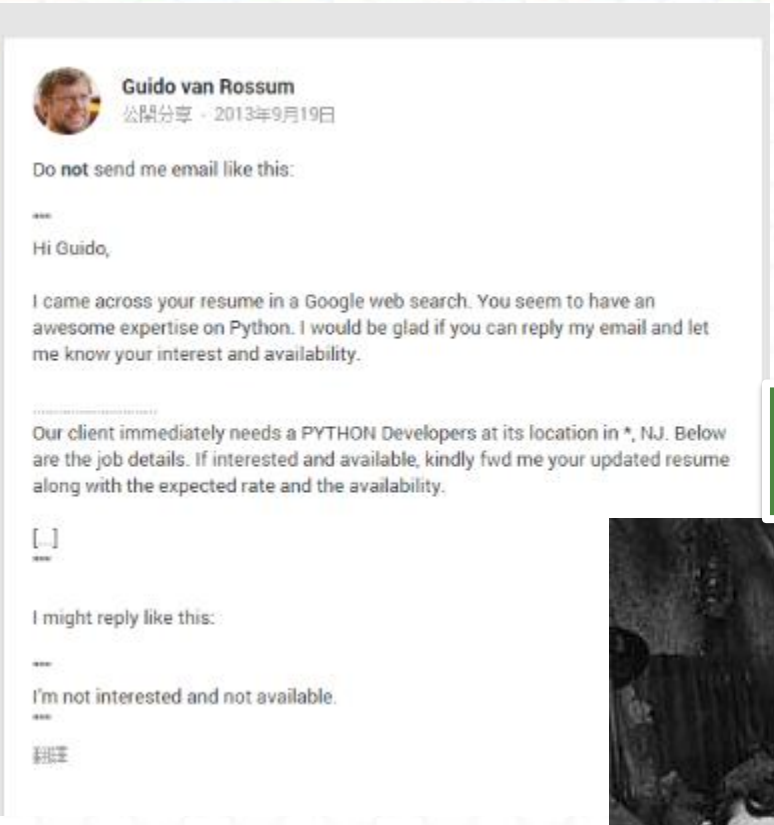

```
class test{
    public static void main(String args[]){
        System.out.println("Hello World");
    }
}
```

PYTHON


```
print("Hello World")
```

Guido van Rossum – Python 之父

Guido Van Rossum
(https://goo.gl/2kul7Y)



Monty Python
(https://goo.gl/dTjCxR)

豐富的函式庫讓Python 無所不在

物聯網
(http://goo.gl/2j45Nk)

網頁製作
(https://goo.gl/2304w7)

資料分析
(https://goo.gl/2304w7)

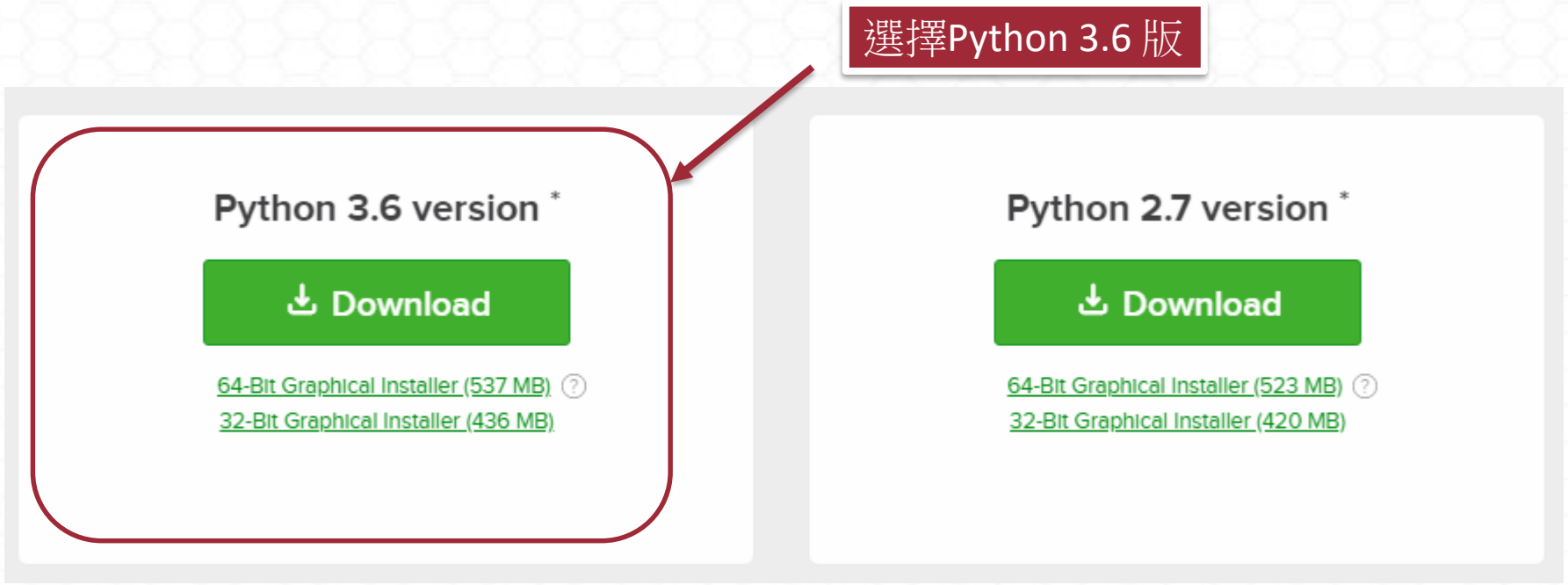
完整的資料分析套件

- 統計科學計算
  - Numpy
  - Scipy
  - statsmodels
- 結構化資料處理與分析
  - Pandas
- 資料探索編輯器
  - Jupyter Notebook
- 深度學習
  - TensorFlow
  - MXNet
- 大數據處理
  - PySpark
- 機器學習
  - Scikit-learn

使用Python 的公司

Python 開發工具簡介

安裝Anaconda



https://www.anaconda.com/download/

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 支援與強大內建函式庫
  - 尚在持續更新與開發中
- *Python2.x*
  - 支援主要作業環境
  - 協力廠商函式庫主要支援版本

從命令列執行Python

- 互動式Shell – 即時看到結果，難以編修程式

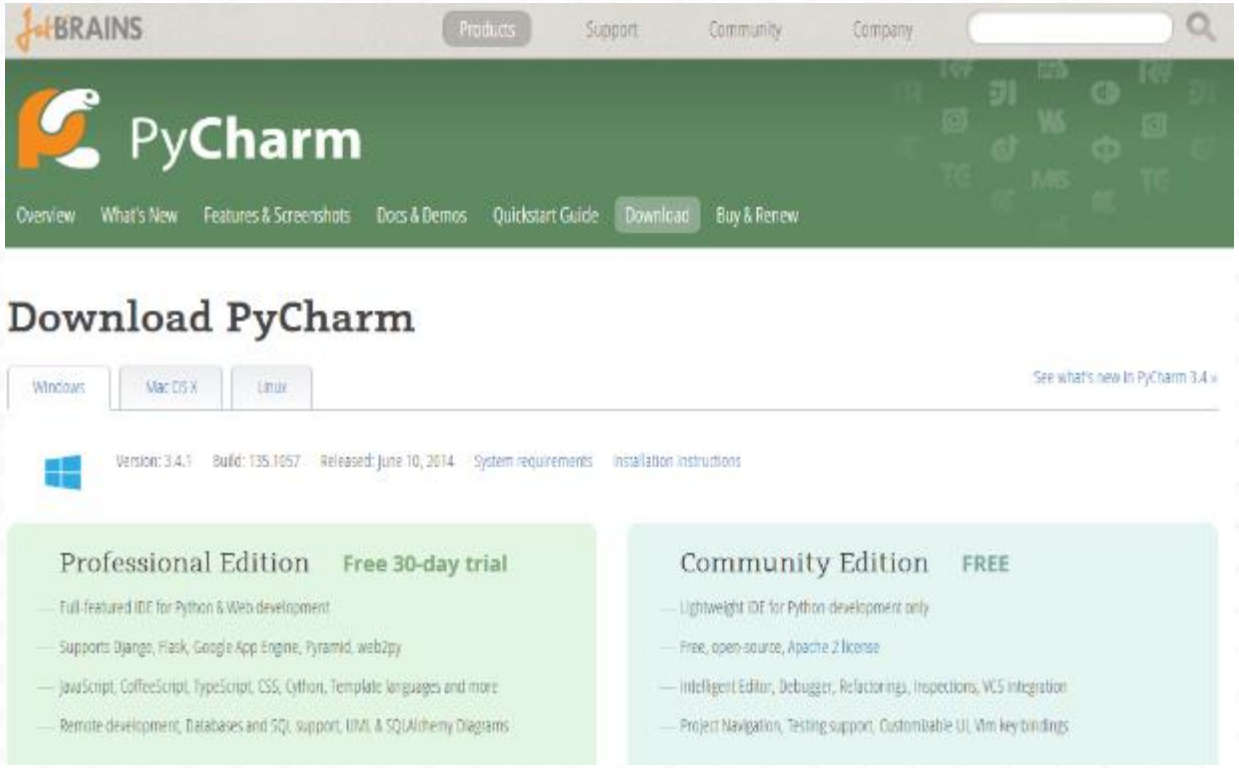
```
C:\Users\david>python
```

- 從檔案執行Python – 無法即時看到結果，容易編修程式

```
C:\Users\david>python test.py
```

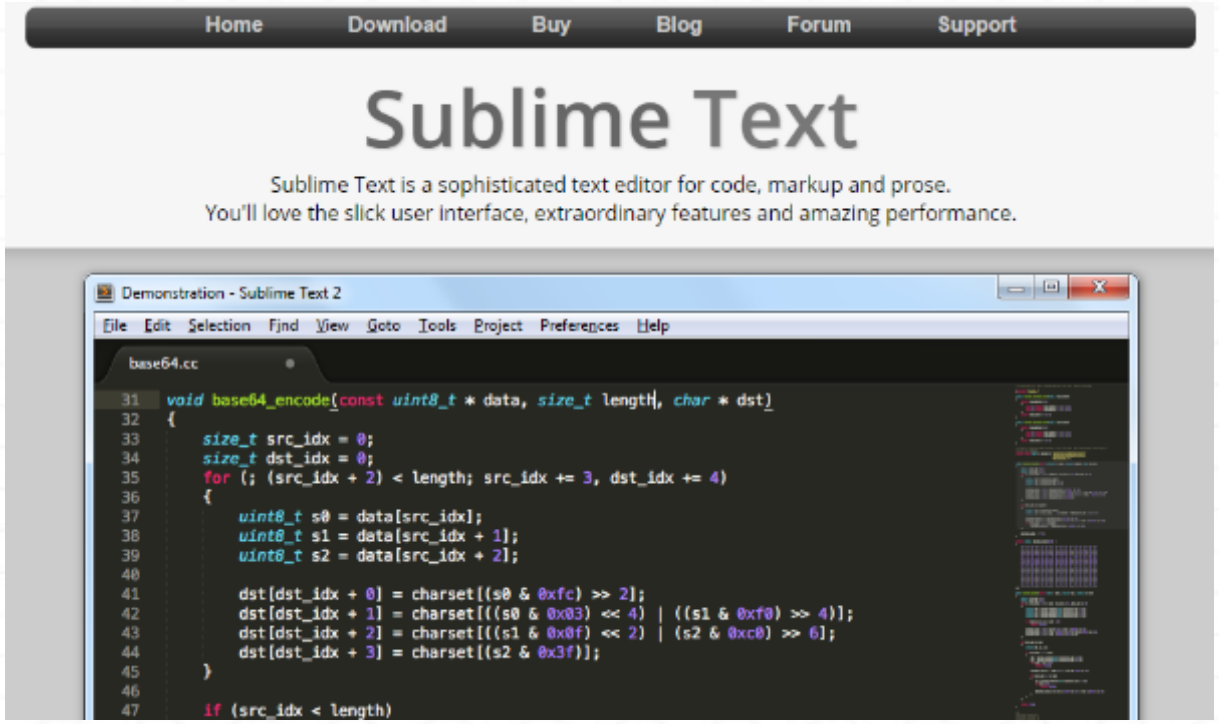
Python IDE - PyCharm

- 建議使用該工具做專案開發
  - https://www.jetbrains.com/pycharm/download/



Sublime Text

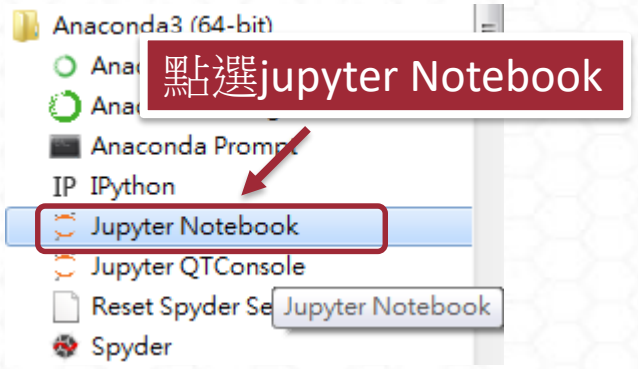
- MAC 使用者建議使用該工具做專案開發
  - https://www.sublimetext.com/

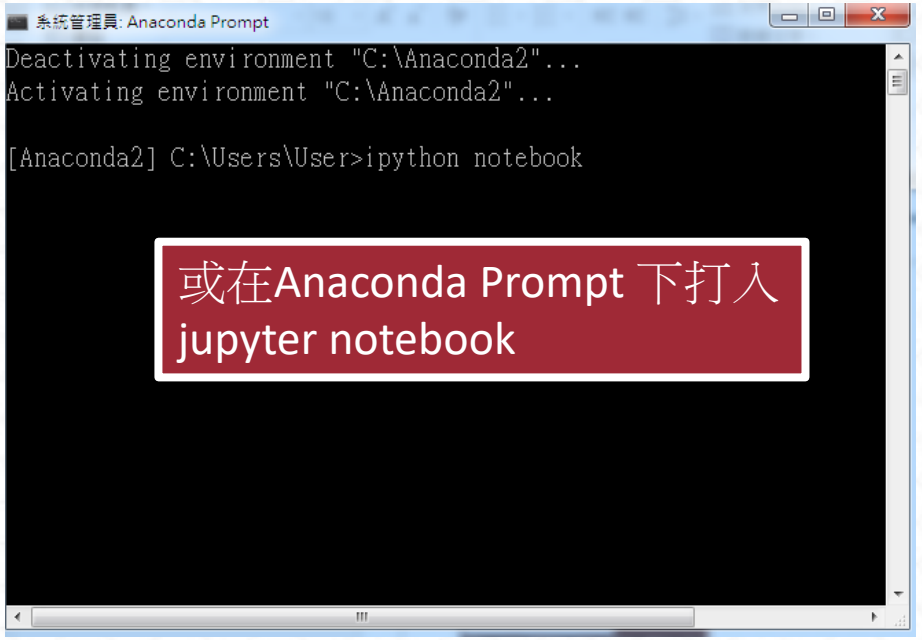


Jupyter Notebook

- 使用於教學與資料探索，不適合開發大型專案

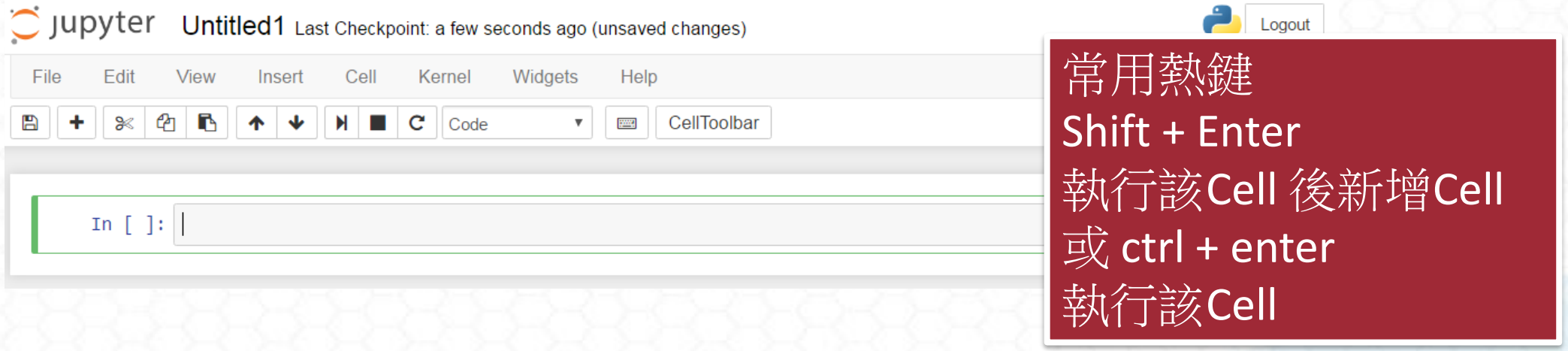
使用 Jupyter Notebook





啟用 Jupyter (jupyter Notebook)

- 在命令列下打:
  - jupyter notebook
  - 自動開啟瀏覽器後便可瀏覽 (預設為localhost:8888)
- 可匯出.ipynb, .py 各種不同格式檔案
- 瀏覽快捷鍵 Help -> Keyboard Shortcuts



Python 數值運算

算數還需倚賴計算機?

- 透過Python 可以直接計算數字

```
3 + 2 * 8
```

數值運算

```
>>> a = 3
>>> b = 2
>>> a + b
5
>>> a - b
1
```

整數加減後
還是整數

```
>>> c = 1.5
>>> d = 2.5
>>> c + d
4.0
>>> c - d
-1.0
```

浮點數運算

數值型態轉換

```
>>> int('2')
2
>>> int(2)
2
>>> float(2)
2.0
>>> float('3.2')
3.2
```

善用type 檢查型態

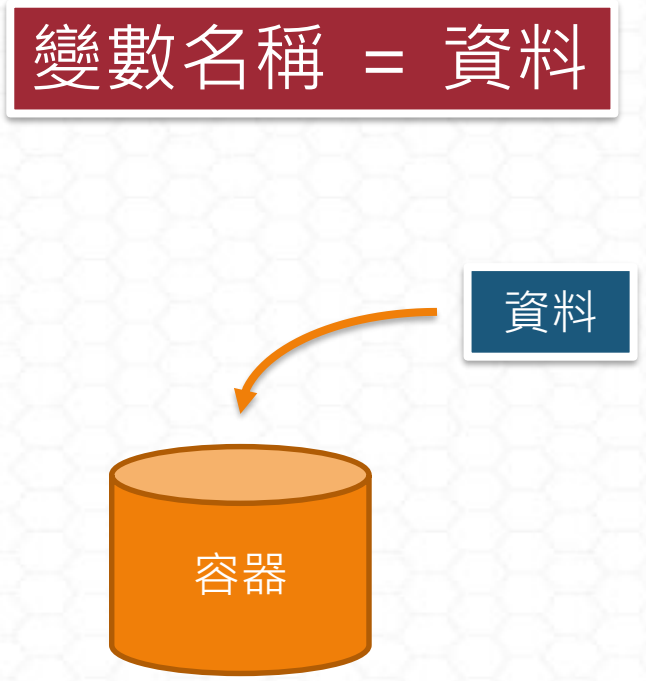
可以如何創建變數？

■ 使用等號

```
a = 5
# Python會將a視為5
a + a
```

■ 可重新命名

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



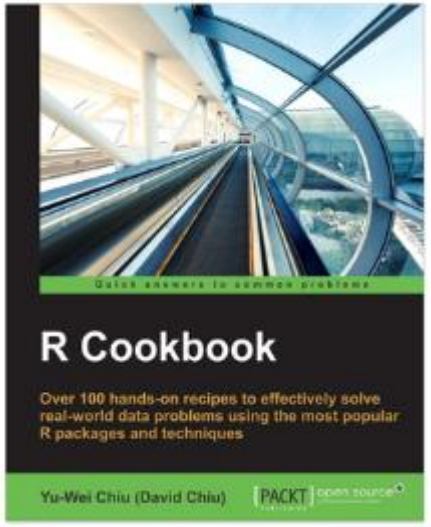
計算台幣價格

price = 25.6

exchange_rate = 32.33

ntd_price= price * exchange_rate

ntd_price



https://goo.gl/wPVmSK

Python 字串

Python 讓文字處理再簡單不過!

- 有沒有試過?
  - 從上千筆記錄將所有英文文字變成大寫
  - 將百篇文章中多餘的空白拿掉
  - 根據特定符號切割文字

http://goo.gl/CC8Qcx

創建字串

- 字串類別 (str) 是內建的

```
a = "hello world"
print(a.__str__)
```

- 使用單引號或雙引號

```
# 使用單引號標註一段話
'This is also a string'
# 使用雙引號標註一段話
"String built with double quotes"
```

```
' I'm using single quotes, but will create an error'
```

SyntaxError: invalid syntax

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

因為單引號停止了字串
因此可以使用雙引號和單引號
的組合表達完整字串

Print 函式

```
■ # 印出字詞
'Hello world'
■ # 中文也沒問題
'寶寶心理苦，但寶寶不說'
■ # 用Print列出所有結果
print('Hello world 1')
print('Hello world 2')
print('Use \n to print a new line')
print('\n')
print('See what i mean?')
```

Print – 特殊字元

- \n – 換行字元

```
print('Here is a new line \n and here is the second line')
```

- \t – 以表格、或表格形式排列資料

```
print('Here is a new line \t and here is the second line')
```

換行與多行文字處理

```
# 使用 \ 作為換行符號
a = 'hi this is a l\
ong text'
print(a)

# 使用三個single quote 做多行處理
a = '''hi this is a l
ong text'''
print(a)
```

字串索引 (1/2)

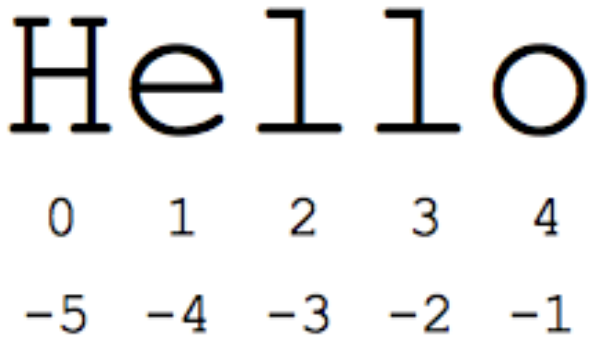
```
s = 'Hello'
print(s)
```

- Python的編制索引為從0開始

```
s[0]
s[1]
```

- 也可使用負號從後面開始索引

```
s[-1]
s[-2]
```



字串索引 (2/3)

- 用：來選定起始點或迄點等範圍

```
#  取得從第一個索引之後的所有元素
s[1:]
s #  並不會更改原字串

#  取到索引為第 3 個的元素
s[:3]
#  取得所有元素
s[:]
```

代表抓取從第0~第3個元素
但不包括第3個元素本身

字串索引 (3/3)

```
#  for(int i=0; i< len(s); i++)
s[::1]
#  for(int i=0; i< len(s); i+=2)
s[::2]
#  for(int i=len(s) -1 ; i > 0; i--)
s[::-1]
```

字串特性 (1)

- 使用+做字串的連接：

```
s
s + 'concatenate me!'
s = s + 'concatenate me!'
print(s)
s
```

字串特性 (2)

- 可用乘號重複創建同一元素：

```
letter = "?"
letter
letter *10
```

字串的其他功能

- .upper( ) 大寫

```
s.upper()
```

- .lower( )：小寫

```
s.lower()
```

- .split( )：分隔

```
s.split()
s.split('W')
```

- 使用函數len()來檢查的字串的長度：

```
len('Hello World')
```

可以用dir(s) 做功能的查詢

Python List

記錄多人的資料

http://goo.gl/TkRhfz

有沒有一個容器能同時記錄多個人的資料?

- 電話
- 姓名
- 性別

Python List

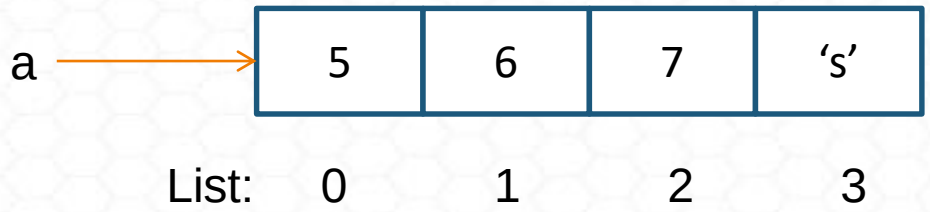
- 剛剛介紹到的數值與字串單為單一值，如果要存放多個值時，可以用設麼資料結構?
  - List
- 其他語言都有陣列的概念(Array)，Python 的List 比較接近於ArrayList
- 宣告List 的方式
  - a = []
  - a = list()

List 可以裝載不同類型元素

```
#List 可以裝載不同類型元素
print(['asap', 3])
print(list("word"))
print( [['list'], ['of', 'lists']] )
print(list(('a', 1)) )

a = ['heterogeneous', 3]
a[1] + 5
```

List 基本操作

```
a = [5, 6, 7, ‘s’] # a = [5 6 7 ‘s’]
print(a[0]) # 5
print(a[2:4]) # [7, ‘s’]
print(a[-1] ) # ‘s’
print(a[-2] ) # 7
print(a[2:] ) # [7, ‘s’]
print(a[::2] ) # [5, 7]
print(a[::-1] ) # [‘s’, 7, 6, 5]
print(len(a)) # 4
```



List 元素增減

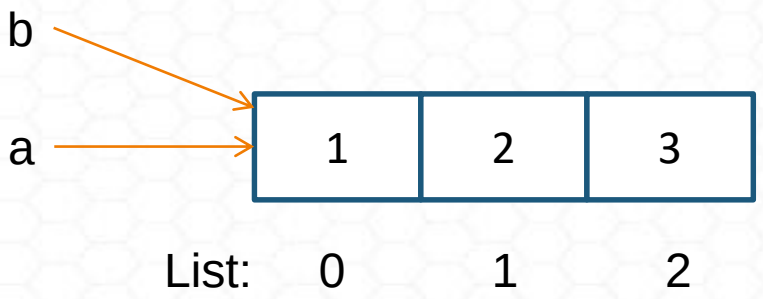
```
a = [5, 6, 7, 8]
a.pop() # a = [5, 6, 7]
a.append(2) # a = [5, 6, 7, 2]
a.sort() # a = [2, 5, 6, 7]
a.reverse() # a = [7, 6, 5, 2]
```

套用List 於字串中

```
>>> list('a')
['a']
>>>hello =  list('hello world')
>>>print(hello)
['h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd']
>>> 'e' in hello
True
>>> 'a' in hello
False
```

指定(Assignment)陣列

```
a = [1, 2, 3]
b = a
a[1] = 2000
b
```



[1, 2000, 3]

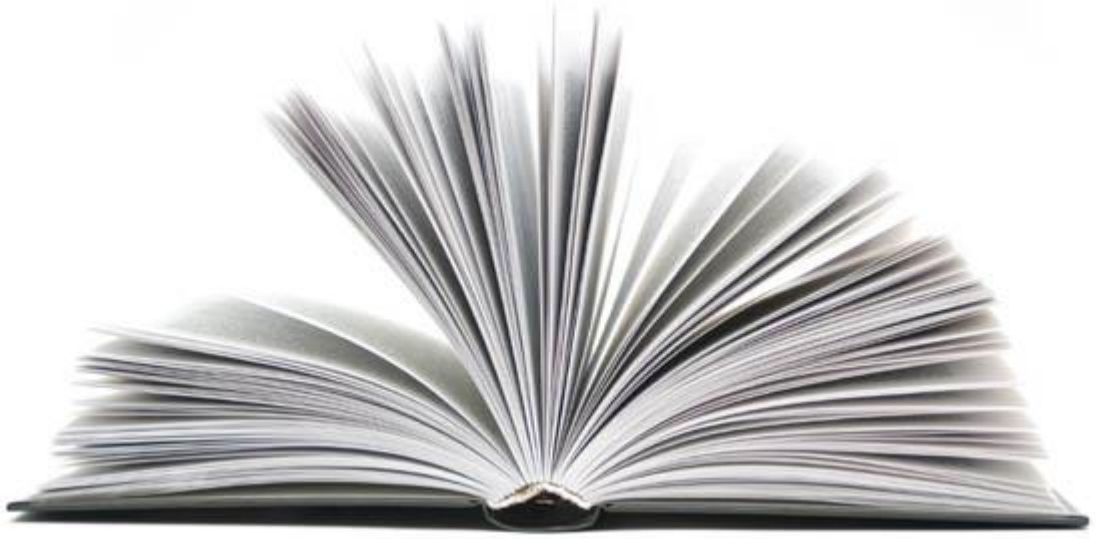
複製List – 解決方法

```
a = [1, 2, 3]
import copy
aa = copy.deepcopy(a) # Deep Copy
a[1] = 2000
aa
```

字典 (Dictionary)

從書中快速找到想要的資料?

- 建立索引頁或許是個比較好的方法



http://goo.gl/37cpjL

Python 字典(Dictionary)

- 其他語言有相同的操作
  - Java: HashMap, HashTable…
  - C++: hashmap
  - C#: Dictionary…
- dic = {key : value}
- 其中key 為唯一不重複值

Dictionary 範例

```
#declare dictionary
dic = {'a':100, 'b':"yes", 'c':0.98}
dic

#get keys in dictionary
dic.keys()

#get values in dictionary
dic.values()

#get value of given key
dic['a']

#get value of given key
dic.get('a')
```

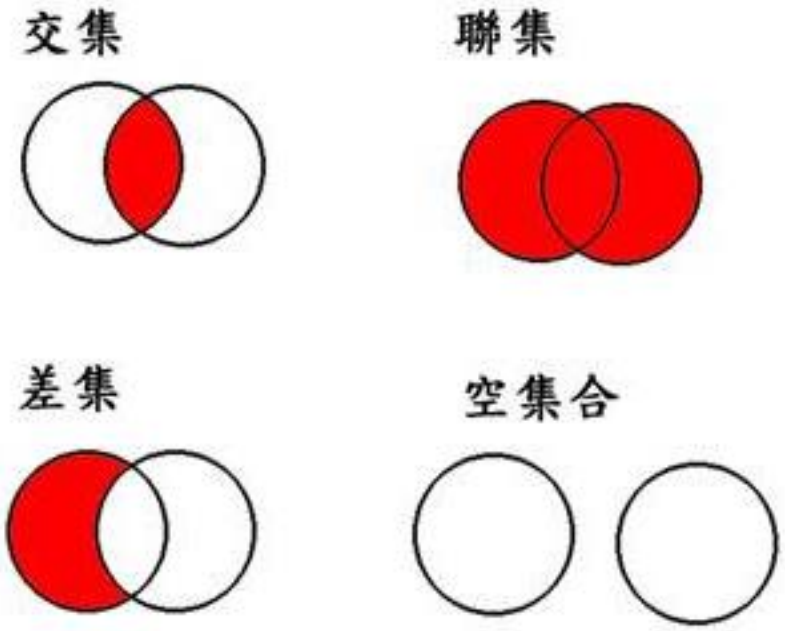
Dictionary 範例(二)

```
#add entry into dictionary
dic['d'] = 'new'
dic

#add entry into dictionary
dic.update({'e':123})
dic

# iter the dictionary
for rec in dic:
    print(rec, dic[rec])
```

集合(SET)

集合

- 當要從兩堆資料中找到之間的相同處與相異處時可以使用集合



http://goo.gl/xRiuwd

Sets

```
x = set()

# 使用 add() 增添資料
x.add(1)
x
x.add(2)
x

# 可以將一個list 轉為 set
l = [1,1,2,2,3,4,5,6,1,1]
set(l)
```

元組(TUPLES)

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])

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。

- 檢視tuple 跟 list 的差異
  - dir(tuple)
  - dir(list)

建立 Tuples

```
# 可以混和不同資料型態
t = ('one',2)

# 可以檢查Tuple 的長度
len(t)

# 根據位置取用值
t[0]
t[-1]
```

Packing 與 Unpacking

```
>>> a,b = 1,2
>>> 1
1
>>> a,b = 1,2
>>> a
1
>>> b
2
>>> c = a,b
>>> c
(1, 2)
>>> d = 1,2
>>> c ==d
True
```

迴圈與控制流程

程式縮排 Indentation

```
def main():
    print('Hello World')
```

- 使用 tab 或空白去區隔執行區塊(替代傳統的{})
  - Pep8規定使用4個空白http://legacy.python.org/dev/peps/pep-0008/#indentation
  - Google 使用2個空白

IF-ELIF-ELSE 控制流程

```
#if - else
a = 9
b = 8
for i in range(1,10):
    if i == a:
        print(str(a) + ' found!')
        break
    elif i == b:
        print str(b) + ' found'
    else:
        print(str(a) + ' was not in the list')
```

注意: 是用ELIF 而非ELSE IF

迴圈控制

```
# from 1 to 10
for i in range(1,10):
    print(i)
```

for(i=1; i< 10 ; i++){ }

```
# step by 2
for i in range(1,10,2):
    print(i)
```

for(i=1; i< 10 ; i+=2){ }

```
# Iterate over list
a = [1,2,3,4,5]
for i in a:
    print(i)
```

Iteration

使用while 迴圈

```
i =0
s = 0
while(i < 101):
    s += i
    i +=1
print(s)
```

Python 沒有do while 迴圈

Python 函式

函式

- 希望能寫一次，但在不同情境可以重複呼叫好幾次
- Python 定義函式的語句: def
  - Defining a function

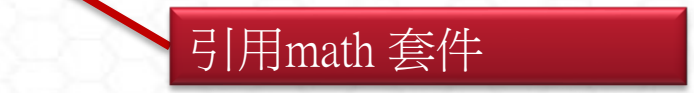
```
def say_hello():
    print(”hello world”)
```

定義簡單函式

```
def addNum(a, b):#Define Function
    return a+b #Return function

print(addNum(1,4))
```

引用套件

```
import math
a = math.ceil(3.6)
a
```



檢視套件與函式

```
import math #Importing Modules
print(math) #Print Content

import sys #Importing Modules
print(sys.argv) #Print Argument

print(range(1,len(sys.argv)))
```

錯誤與例外
(Errors and Exceptions)

錯誤訊息

- 寫程式出錯是不可避免的，但該怎麼處理錯誤情況呢？

除以0 的時候的處理：Try & Except

```
dividend = input("Dividend: ")
divisor = input("Divisor: ")

try:
    print(float(dividend) / float(divisor))
except ZeroDivisionError as detail:
    print("ZeroDvisionError", detail)
```

想像是try catch – 針對除以零時的對應動作

Try & Except

```
dividend = input("Dividend: ")
divisor = input("Divisor: ")

try:
    print(float(dividend) / float(divisor))
except:
    print(¨Error¨)
```

與上面例子的不同?

```
Dividend: "qoo"
Divisor: 3
error
```

Finally

```
dividend = input("Dividend: ")
divisor = input("Divisor: ")

try:
    print(float(dividend) / float(divisor))
except:
    print("Error")
finally:
    print('final block')
```

最後執行的區塊

檔案處理

操作檔案

- 透過檔案，我們可以保存與讀取資料

1, David, M
2, Mary, F
3, John, M

寫入檔案

```
#將Hello World 寫入檔案中
fid = open('test.txt', 'w')
fid.write('Hello\nWorld')
fid.close()

# 使用with 自動關檔
with open('test.txt', 'w') as f:
        f.write('Hello\nWorld')
```

讀取檔案

```
# 一行一行讀取檔案
with open('test.txt', 'r') as fid:
    for line in fid:
        print("Line: " + line.strip())

# 讀取檔案中所有內容
with open('test.txt', 'r') as fid:
    s = fid.read()
print("line",s)
```

計算行數

```
#計算行數
k = 0
with open('1.tsv') as fid:
    for line in fid:
        k = k + 1
print(k)

#或簡寫為:
print(len([ line for line in open('test.txt')]))
```

問題描述

試寫出一個Python 程式可以計算檔案中每個詞出現的頻率?

THANK YOU