



# Python一天學會

吳佳謙 老師





1. 下載及安裝Python軟體
2. Python直譯器與計算機
3. 資料結構
4. 控制結構
5. 函數
6. 類別
7. 繼承
8. 異常或錯誤處理
9. 使用matplotlib畫圖





# 1. 下載及安裝Python軟體

到[python.org](https://www.python.org)下載軟體

到Aanconda[下載Python組合](https://www.anaconda.com)



# 到python.org下載軟體

The screenshot shows the Python Software Foundation website. At the top, there's a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below the navigation bar is the Python logo and a search bar with a magnifying glass icon and a 'GO' button. A socialize button is also present. The main content area features a dark blue sidebar on the right containing a 'Compound Data Types' section with text about lists and code examples. On the left, there's a large code editor window showing Python code related to list comprehensions and the enumerate function. Below the sidebar, a quote reads: "Python is a programming language that lets you work quickly and integrate systems more effectively. [» Learn More](#)". At the bottom, a yellow banner encourages users to join the "Python Developers Survey 2017" and win prizes, with a "Start the survey!" button.

Get Started

Whether you're new to  
programming or an experienced

Download

Python source code and installers  
are available for download for all

Docs

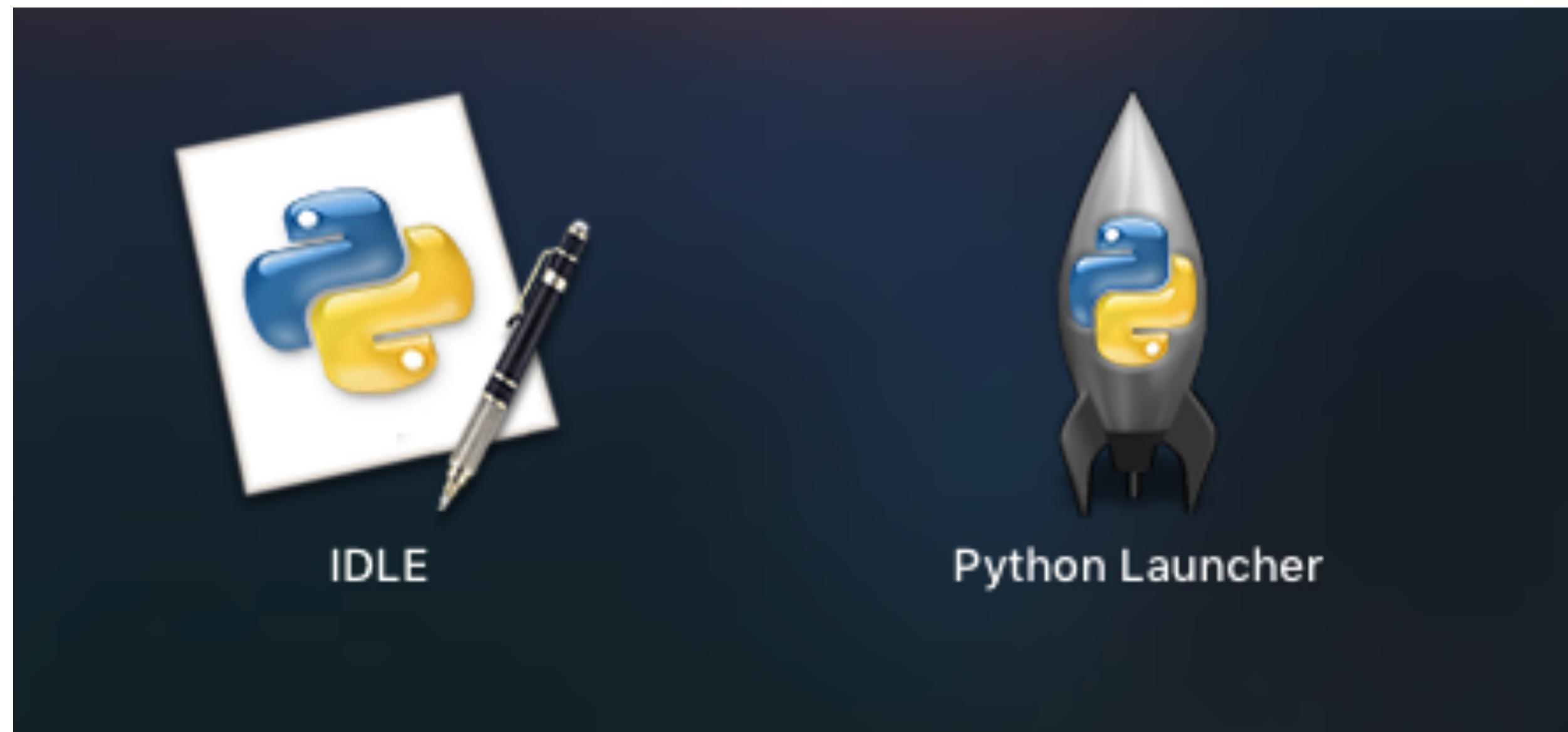
Documentation for Python's  
standard library, along with tutorials

Jobs

Looking for work or have a Python  
related position that you're trying to

# Python安裝

- 這是在Mac上安裝完的Python Launcher



# 這是安裝完Python的直譯環境

```
Python 3.6.2 Shell
Python 3.6.2 (v3.6.2:5fd33b5926, Jul 16 2017, 20:11:06)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> WARNING: The version of Tcl/Tk (8.5.9) in use may be unstable.
Visit http://www.python.org/download/mac/tcltk/ for current information.

>>>
```

# 到pypi.python.org下載 easy\_install程式並且安裝

The screenshot shows the homepage of the Python Package Index (PyPI). At the top left is the Python logo and the word "python™". To its right is a search bar with a "search" button. Below the search bar, the text "» Package Index" is visible. On the far right, there's a "Not Logged In" sidebar with links for "Login", "Register", "Lost Login?", "Login with OpenID", and "Login with Google". Below the sidebar, a "Status" section says "Nothing to report". The main content area has a heading "PyPI - the Python Package Index" and a sub-heading "The Python Package Index is a repository of software for the Python programming language. There are currently **119382** packages here. To contact the PyPI admins, please use the [Support](#) or [Bug reports](#) links." To the left of the main content is a sidebar with sections for "PACKAGE INDEX", "ABOUT", "NEWS", "DOCUMENTATION", "DOWNLOAD", "COMMUNITY", "FOUNDATION", and "CORE DEVELOPMENT". Each section contains a list of links. The "PACKAGE INDEX" section is currently active, showing links like "Browse packages", "List trove classifiers", "RSS (latest 40 updates)", "RSS (newest 40 packages)", "Terms of Service", "PyPI Tutorial", "PyPI Security", "PyPI Support", "PyPI Bug Reports", "PyPI Discussion", and "PyPI Developer Info". The "ABOUT" section includes links for "ABOUT", "NEWS", "DOCUMENTATION", "DOWNLOAD", "COMMUNITY", "FOUNDATION", and "CORE DEVELOPMENT". The "NEWS" section has links for "ABOUT", "NEWS", "DOCUMENTATION", "DOWNLOAD", "COMMUNITY", "FOUNDATION", and "CORE DEVELOPMENT". The "DOCUMENTATION" section has links for "ABOUT", "NEWS", "DOCUMENTATION", "DOWNLOAD", "COMMUNITY", "FOUNDATION", and "CORE DEVELOPMENT". The "DOWNLOAD" section has links for "ABOUT", "NEWS", "DOCUMENTATION", "DOWNLOAD", "COMMUNITY", "FOUNDATION", and "CORE DEVELOPMENT". The "COMMUNITY" section has links for "ABOUT", "NEWS", "DOCUMENTATION", "DOWNLOAD", "COMMUNITY", "FOUNDATION", and "CORE DEVELOPMENT". The "FOUNDATION" section has links for "ABOUT", "NEWS", "DOCUMENTATION", "DOWNLOAD", "COMMUNITY", "FOUNDATION", and "CORE DEVELOPMENT". The "CORE DEVELOPMENT" section has links for "ABOUT", "NEWS", "DOCUMENTATION", "DOWNLOAD", "COMMUNITY", "FOUNDATION", and "CORE DEVELOPMENT". Below the sidebar, there are three boxes: "Get Packages" (instructions for using pip), "Package Authors" (instructions for submitting packages), and "Infrastructure" (information about interoperation and interfaces). At the bottom, there's a table showing recently updated packages:

Updated	Package	Description
2017-10-16	pyqybe 0.0.2	
2017-10-16	cyanite-utils 0.0.13	Cyanite Utils
2017-10-16	Jupyter 0.250	Productive Simple Intuitive Jupyter Python Utilities

# 使用easy\_install安裝pip套件

- easy\_install 套件名稱
- 安裝pip
- easy\_install install pip

# 使用pip install 套件名稱

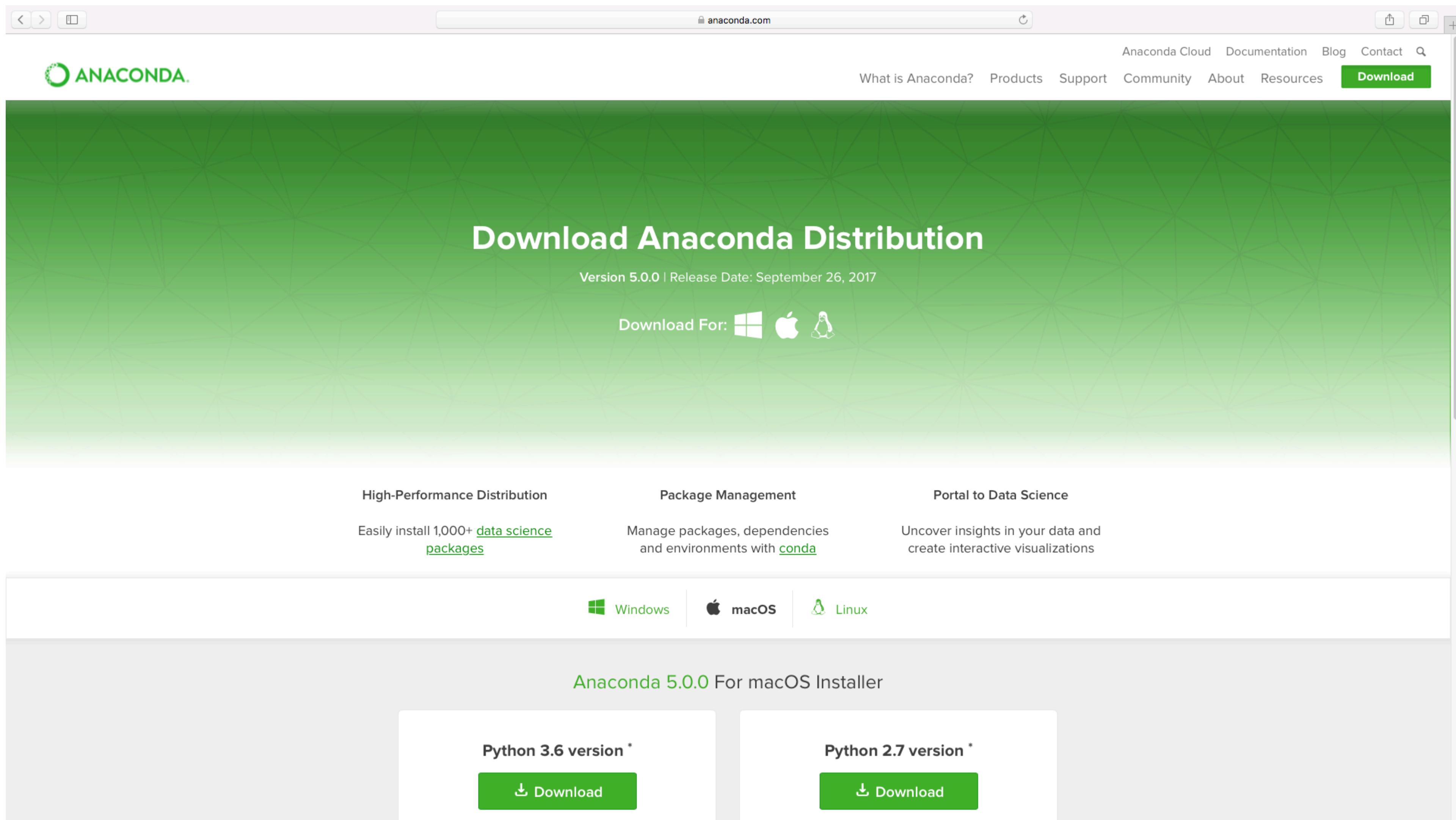
- 安裝套件
- pip install 套件名稱
- 解除安裝套件
- pip uninstall 套件名稱
- 檢視目前系統已經安裝的套件
- pip list

# 安裝

## Anaconda,Numpy,Matplotlib

- 安裝Anaconda組合包
- 此包含Python,spyder,和科學大數據計算軟體
- 安裝Numpy,Matplotlib,Scipy

- 到Anaconda下載Python組合包<https://www.anaconda.com/>



# 使用conda升級套件

```
Last login: Mon Oct 16 07:12:00 on console
[60-250-191-81:~ justinwu$ pip install scipy
Requirement already satisfied: scipy in ./anaconda3/lib/python3.6/site-packages
[60-250-191-81:~ justinwu$ pip install -U scipy
Requirement already up-to-date: scipy in ./anaconda3/lib/python3.6/site-packages
[60-250-191-81:~ justinwu$ conda install numpy
Fetching package metadata .....
Solving package specifications: .

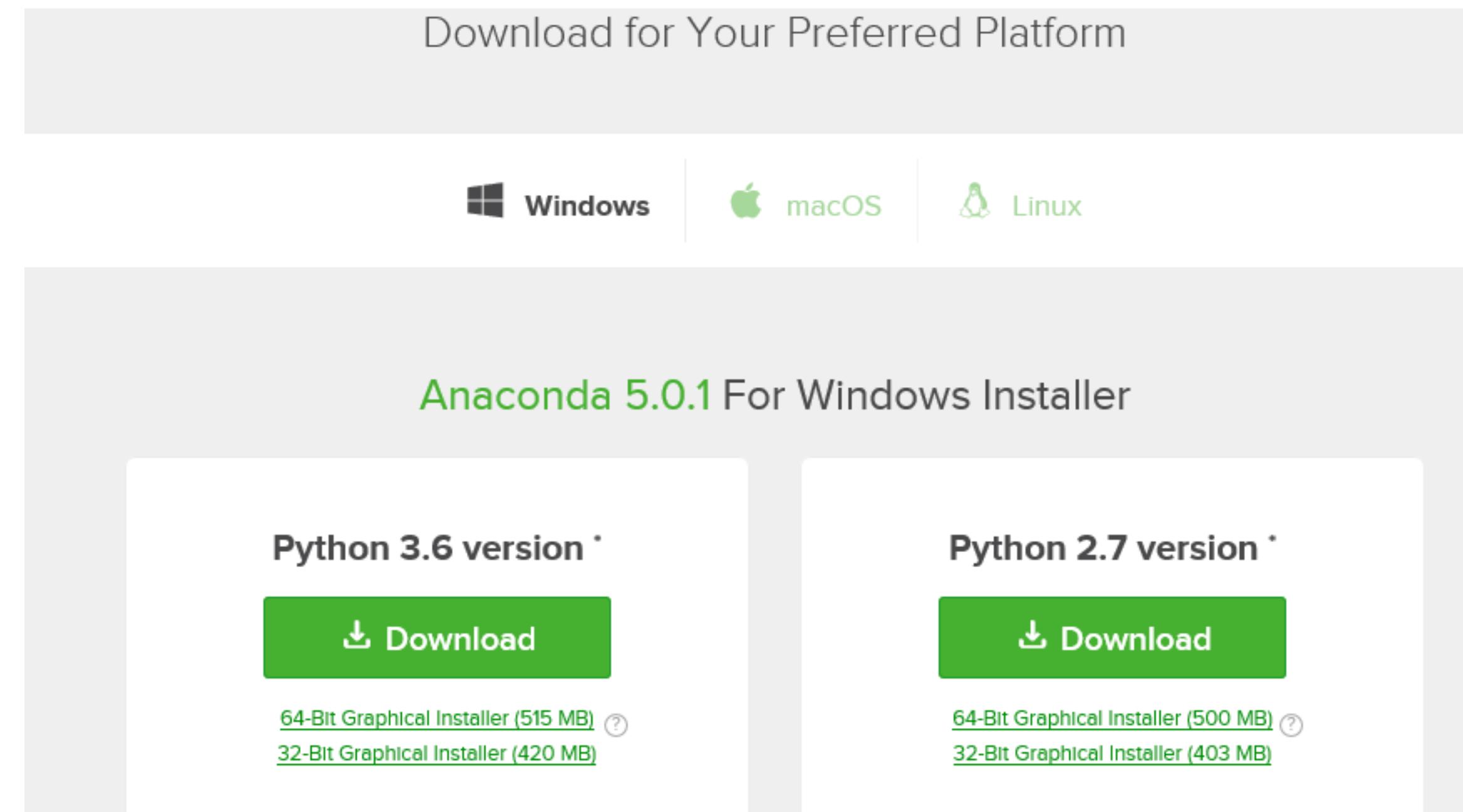
Package plan for installation in environment /Users/justinwu/anaconda3:

The following packages will be UPDATED:

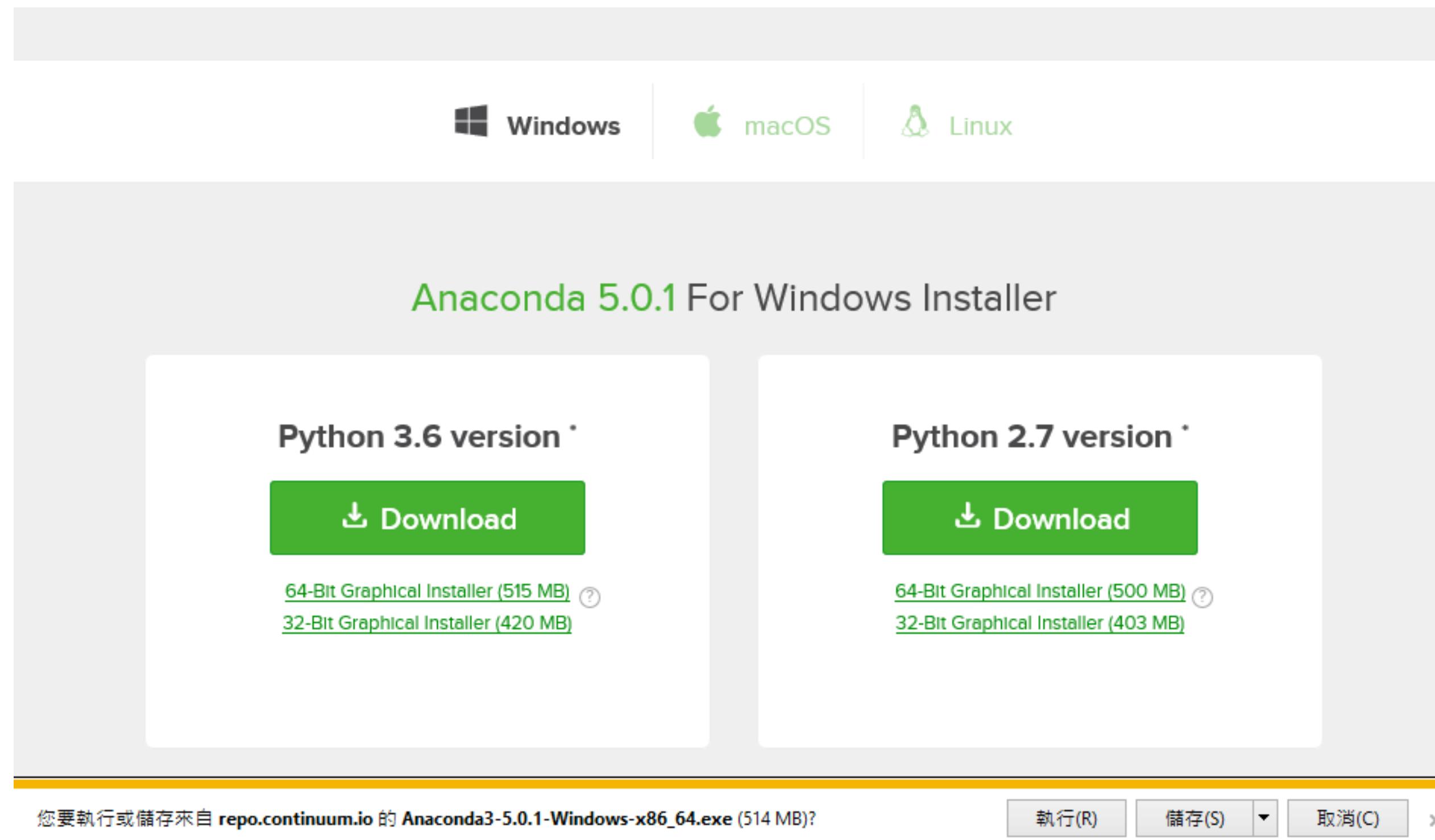
  anaconda: 5.0.0-py36hd9bc8a5_0 --> custom-py36_0
    numpy:    1.13.1-py36h93d791d_2 --> 1.13.3-py36h2cdce51_0

Proceed ([y]/n)?
anaconda-custo 100% [########################################] Time: 0:00:00 238.23 kB/s
numpy-1.13.3-p 100% [########################################] Time: 0:00:00 11.34 MB/s
60-250-191-81:~ justinwu$ ]
```

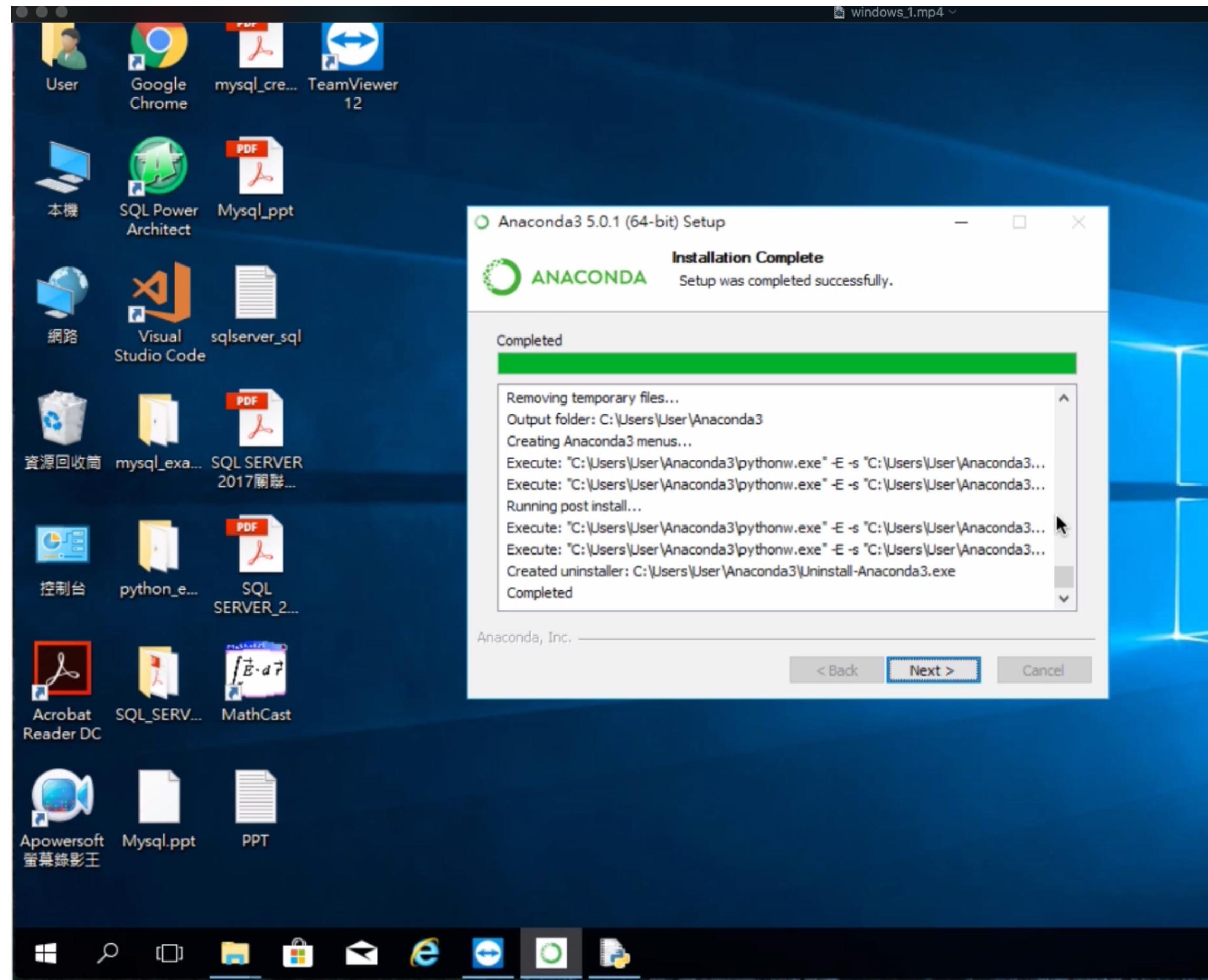
# 在Windows上安裝



# 下載並且安裝Anaconda



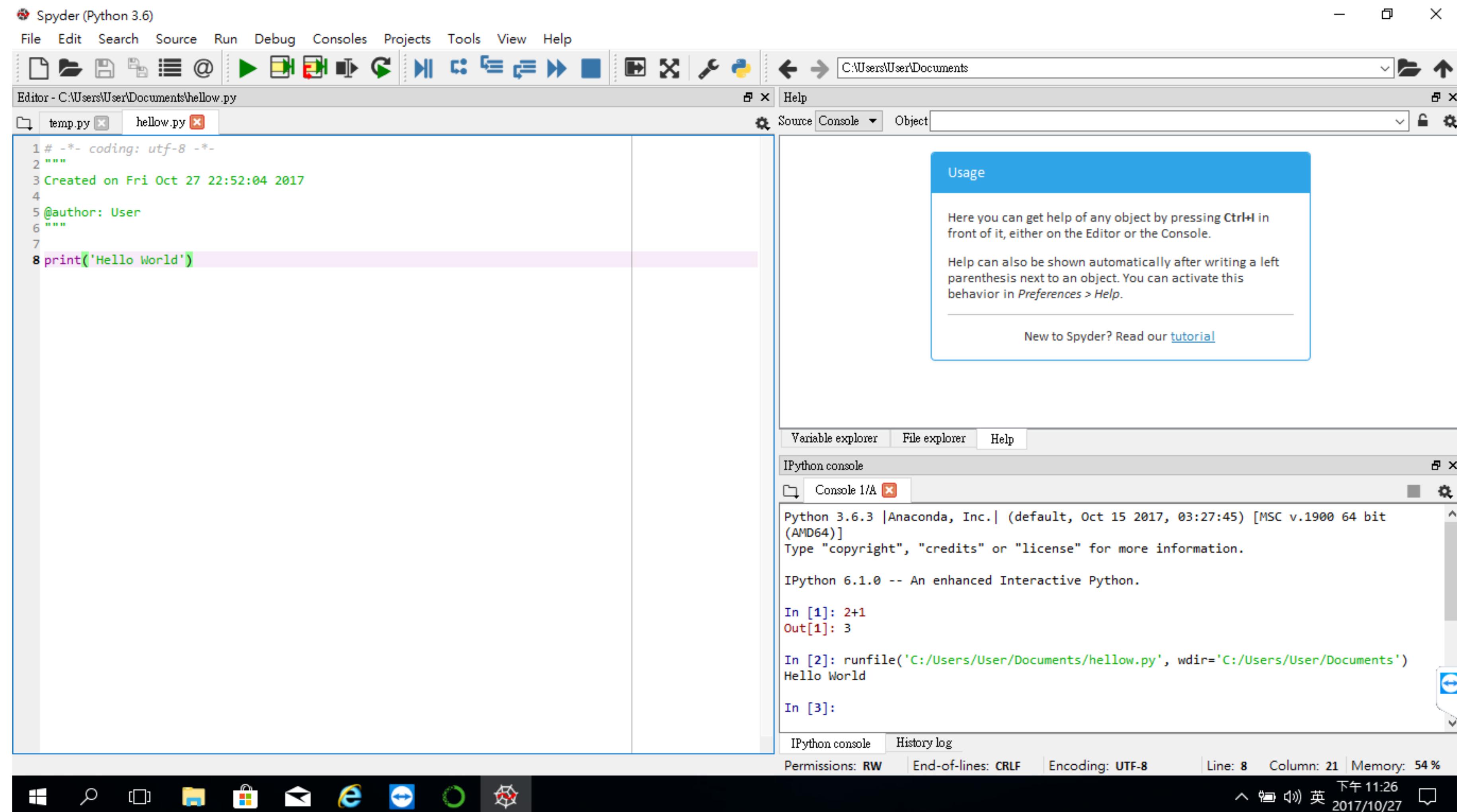
# 安裝軟體Anaconda



# 選取Anaconda Navigator和Spider



# 執行Spider





## 2. Python直譯器與計算機

- Mac電腦/usr/local/bin
- Windows電腦C:\python36
- set path=%path%;C:\python36



# 輸入python執行

```
$ python
Python 3.6.2 |Anaconda, Inc.| (default,
Sep 21 2017, 18:29:43)
[GCC 4.2.1 Compatible Clang 4.0.1 (tags/
RELEASE_401/final)] on darwin
Type "help", "copyright", "credits" or
"license" for more information.
```

>>>

# UTF-8編碼

```
# -*- coding: encoding -*-
```

這是設定utf-8-\*編碼

```
#-*- coding: utf-8 -*-
```

# 註解

- #是註解符號

# python計算機

```
[>>> 1+2
3
[>>> 2-1
1
[>>> 3*2
6
[>>> 3/2
1.5
[>>> 3%2
1
[>>> 50-5/6
49.16666666666664
[>>> (50-5*6)/4
5.0
[>>> 5**2
25
[>>> 2**5
32
>>> ]
```



# 3. 資料結構

- 變數
- 運算式與運算子
- 串列
- 堆疊
- 倷列



# 變數

- 資料型態
  - 整數
  - 浮點數
  - 字串

```

1#!/usr/bin/env python3
2#_*_coding:utf-8_*
3"""
4Created on Thu Oct 26 07:40:42 2017
5
6@author: justinwu
7"""
8#這是註解
9
10#這是註解
111+2
12#print(1+2)
13x=2-1
14print(x)
15y=3+2
16print(y)
17z=3.2*2
18print(z)
19str='大家好'
20print(str)
21str2='Python'
22mychar=str[0]
23print(mychar)

```

Name	Type	Size	
mychar	str	1	大
str	str	1	大家好
str2	str	1	Python
x	int	1	1
y	int	1	5
z	float	1	6.4

Console 1/A

In [28]: runfile('/Users/justinwu/Desktop/test.py')  
1  
5  
6.4  
大家好  
大

In [29]:

# 運算式與運算子

- 運算式是由運算子與運算元組成
- +加-減\*乘/除是運算子,先乘除後加減的結合優先順序
- 運算元是變數,數字,字串和資料結構
- =是分配符號,將右邊的值分配給左邊變數

```
1#!/usr/bin/env python3
2#_*_coding:utf-8_*
3"""
4Created on Thu Oct 26 07:40:42 2017
5
6@author: justinwu
7"""
8#這是註解
9
10#這是註解
111+2
12#print(1+2)
13x=2-1
14print(x)
15y=3+2
16print(y)
17z=3.2*2
18print(z)
19str='大家好'
20print(str)
21
```

Name	Type	Size	
str	str	1	大家好
x	int	1	1
y	int	1	5
z	float	1	6.4

In [22]: runfile('/Users/justinwu/Desktop/test.py')
1
5
6.4
大家好

In [23]:

# 串列

```
>>> fruits = ['orange','apple','pear','banana','kiwi','apple','banana']
>>> fruits.count('apple')
2
>>> fruits.index('banana')
3
>>> fruits.index('banana',4)#Finding next banana starting a position 4
6
>>> fruits.reverse()
>>> fruits
['banana', 'apple', 'kiwi', 'banana', 'pear', 'apple', 'orange']
>>> fruits.append('grape')
>>> fruits
['banana', 'apple', 'kiwi', 'banana', 'pear', 'apple', 'orange', 'grape']
>>> fruits.sort()
>>> fruits
['apple', 'apple', 'banana', 'banana', 'grape', 'kiwi', 'orange', 'pear']
>>> fruits.pop()
'pear'
>>>
```

# 堆疊

```
Python 3.6.2 Shell
Python 3.6.2 (v3.6.2:5fd33b5926, Jul 16 2017, 20:11:06)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> WARNING: The version of Tcl/Tk (8.5.9) in use may be unstable.
Visit http://www.python.org/download/mac/tcltk/ for current information.

>>> stack=[3,4,5]
>>> stack.append(6)
>>> stack.append(7)
>>> stack
[3, 4, 5, 6, 7]
>>> stack.pop()
7
>>> stack.pop()
6
>>> stack
[3, 4, 5]
>>>
```

# 佇列

```
8 from collections import deque
9 queue = deque(['阿呆', 'Eric', 'John', 'Michael', '小寶', '小文'])
10 queue.append("Terry")
11 queue.append("Graham")
12 print(queue.popleft())
13 print(queue.popleft())
14 print(queue)
15
16
```

```
In [1]: runfile('/Users/justinwu/Desktop/queue.py', wdir='/Us
阿呆
Eric
deque(['John', 'Michael', '小寶', '小文', 'Terry', 'Graham'])

In [2]:
```

# 數組tuple,集合set和字典

- 可以用數組tuple來儲存固定的元素,使用小括號()來建立一數組tuple
- 集合的元素放置沒有按照順序,可以使用{}大括號來建立一集合Set
- 集合加上索引就是字典{索引:值}

# Tuple數組

- 也可以從字串中建立數組
- `tp5 = tuple('Ivy Lin')`
- 從數組得到串列
- `list1 = list(tp5)`

```
8
9 tp1=()
10 print(tp1)
11 tp2=(1,2,3,4,5,6,7,8)
12 print(tp2)
13 print(sum(tp2))
14 print('-----')
15 print(tp2[2:5])#切割運算子
16 print(tp2[-1])
17 tp3 =tuple([2*x for x in range(1,8)])
18 print(tp3)
19 print('-----')
20 tp4=tuple('Ivy Lin')
21 print(tp4)
22 tp5=("John",'小寶','小文')
23 print(tp5)
24 print(len(tp5))
25 print(tp4+tp5)
26 print('-----')
27 tp6=tuple([1,2,3,4,5,6,7,8,9])
28 print(tp6)
29 print(max(tp6))
30 print(min(tp6))
31
```

```
In [6]: runfile('/Users/justinwu/Desktop/tuple.py', wdir='Desktop')
()
(1, 2, 3, 4, 5, 6, 7, 8)
36
-----
(3, 4, 5)
8
(2, 4, 6, 8, 10, 12, 14)
-----
('I', 'v', 'y', ' ', 'L', 'i', 'n')
('John', '小寶', '小文')
3
('I', 'v', 'y', ' ', 'L', 'i', 'n', 'John', '小寶', '小文')
-----
(1, 2, 3, 4, 5, 6, 7, 8, 9)
9
1

In [7]:
```

```
8  
9 tp6=tuple([66,22,3,46,5,65,7,83,19])  
10 print(tp6)  
11 list1 = list(tp6)  
12 list1.sort()#排序串列  
13 print(list1)  
14 print('-----')  
15 tp8=tuple(list1)  
16 tp9=tuple(list1)  
17 print(tp8)  
18 print(tp8 == tp9)#比較兩個數組tuple  
19  
20  
21
```

```
In [15]: runfile('/Users/justinwu/Desktop')  
(66, 22, 3, 46, 5, 65, 7, 83, 19)  
[3, 5, 7, 19, 22, 46, 65, 66, 83]  
-----  
(3, 5, 7, 19, 22, 46, 65, 66, 83)  
True  
In [16]:
```

# Set集合

- 集合(set)用來儲存沒有重複的元素.
- 集合的元素是不可以複製的,元素放置也沒有按照順序
- 可以使用{}大括號來建立一集合Set

# Set集合

```
8  
9 st1=set()#建立一個空集合  
10 st2=set([1,2,3,4,5])  
11 print(st2)  
12 st3={'a','b','c','d','e'}  
13 print(st3)  
14 print('-----')  
15 st3.add('f')  
16 print(st3)  
17 st3.remove('d')  
18 print(st3)  
19 print('-----')  
20 print(st3.union(st2))  
21 st5={'a','b','c','d','e'}  
22 print(st3.intersection(st5))  
23 print(st3.difference(st5))
```

```
In [30]: runfile('/Users/justinwu/Desktop')  
{1, 2, 3, 4, 5}  
{'d', 'a', 'e', 'c', 'b'}  
-----  
{'d', 'f', 'a', 'e', 'c', 'b'}  
{'f', 'a', 'e', 'c', 'b'}  
-----  
{1, 2, 3, 4, 5, 'f', 'a', 'e', 'c', 'b'}  
{'b', 'a', 'c', 'e'}  
{'f'}  
  
In [31]:
```

# 字典

- 集合加上索引就是字典{索引:值}

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Fri Oct 27 19:40:39 2017
5
6@author: justinwu
7"""
8tel={'Justin':'0920909872', 'Ivy':'0922876895'}
9tel['Johny']='0920885356'
10print(tel)
11print(tel['Johny'])
12del tel['Johny']
13tel['Mary']='0922865255'
14print(tel)
15print(list(tel.keys()))
16print(sorted(tel.keys()))
17print('Ivy' in tel)
18print('Johny' in tel)
```

```
In [5]: runfile('/Users/justinwu/Desktop/TOP/python/
example/dictionary_1.py', wdir='/Users/justinwu/Desktop/
TOP/python/example')
{'Justin': '0920909872', 'Ivy': '0922876895', 'Johny':
'0920885356'}
0920885356
{'Justin': '0920909872', 'Ivy': '0922876895', 'Mary':
'0922865255'}
['Justin', 'Ivy', 'Mary']
['Ivy', 'Justin', 'Mary']
True
False

In [6]:
```



# 4. 控制結構

- 選取結構if
- 迴圈結構while,for

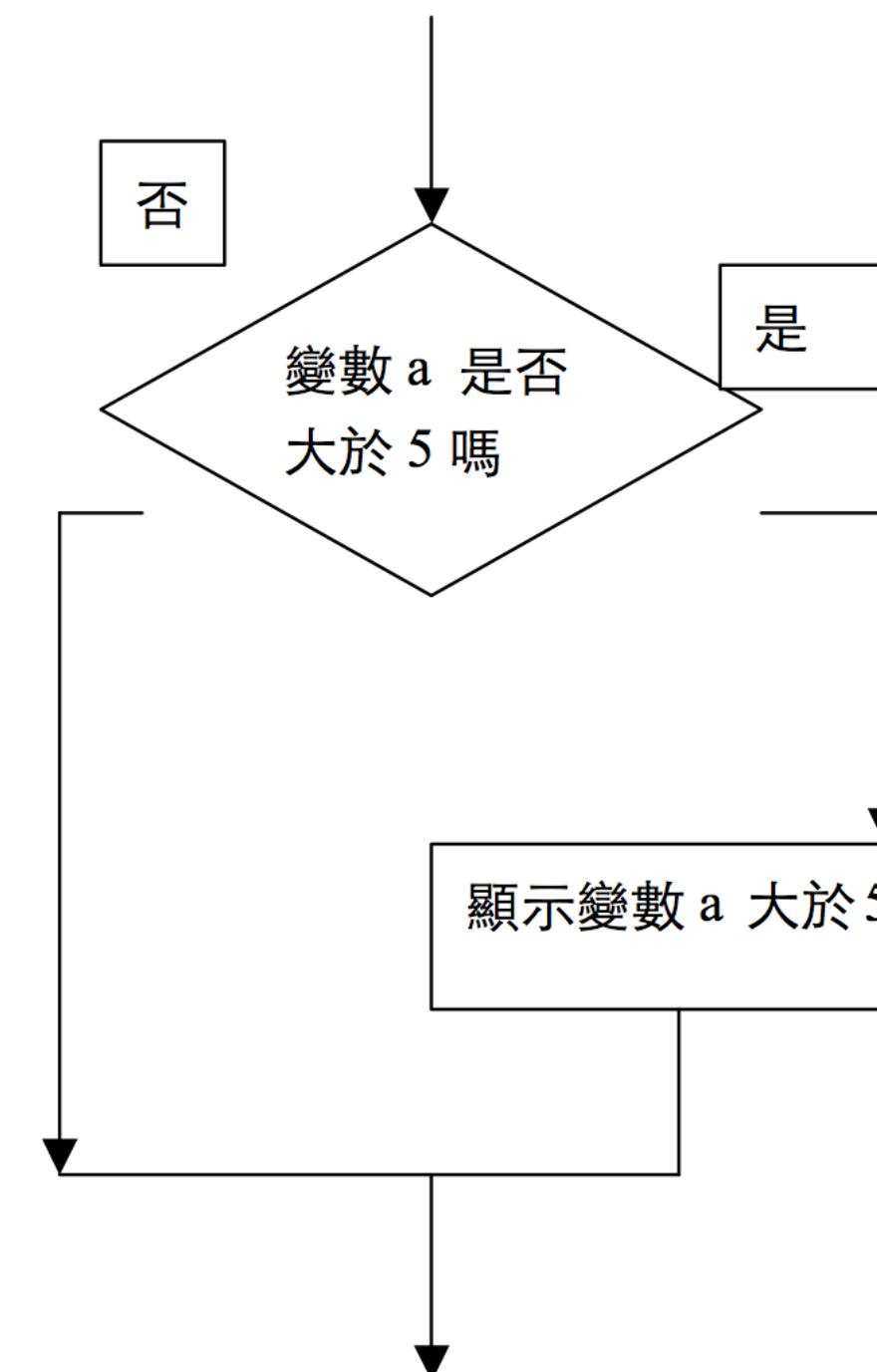


# 選取結構if

- 語法if:
- if 條件運算式:
  - 程式敘述1
- else:
  - 程式敘述2

# 布林運算式

- 如何來選擇流程前進的方向，我們必須經過測試條件，例如，當條件成立時往左方，當條件不成立時往右方。我們使用布林表示式來測試工作。
- 布林Boolean代數定義在一個二元素的集合上，即 $B=\{\text{true},\text{false}\}$ ，true為真，false為假。我們可以使用這個值的結果來決定我們行進的方向。
- 當下列菱形四邊形成立true時會執行右方的流程，當下列菱形四邊行的條件不成立false時會執行左方的流程。true和false就是屬於布林代數，這是用在if判別式。

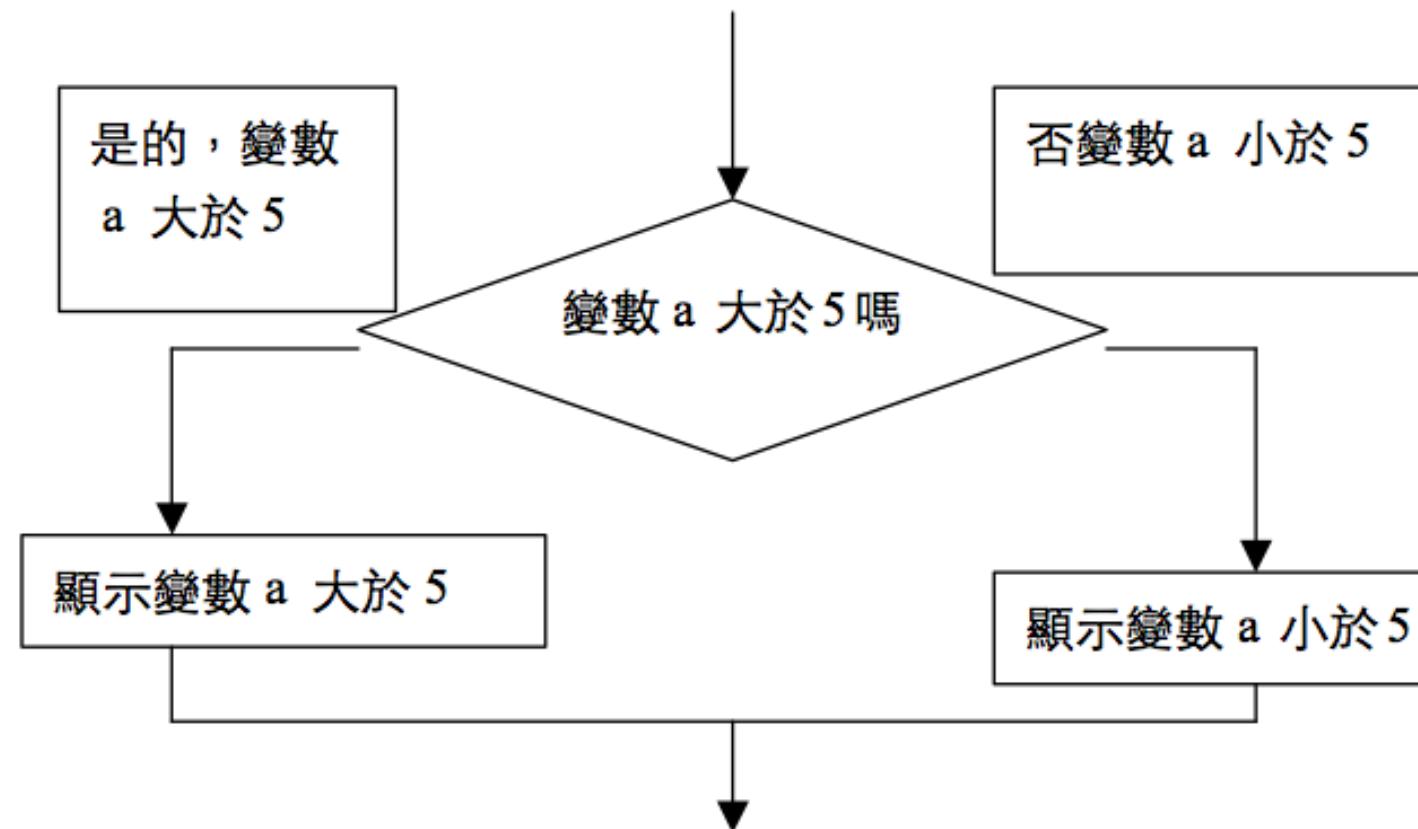


- 當下列菱形四邊形成立true時會執行右方的流程，當下列菱形四邊形的條件不成立false時會執行下方或右方的流程。  
True和false就是屬於布林代數，這是運用在迴圈結構。

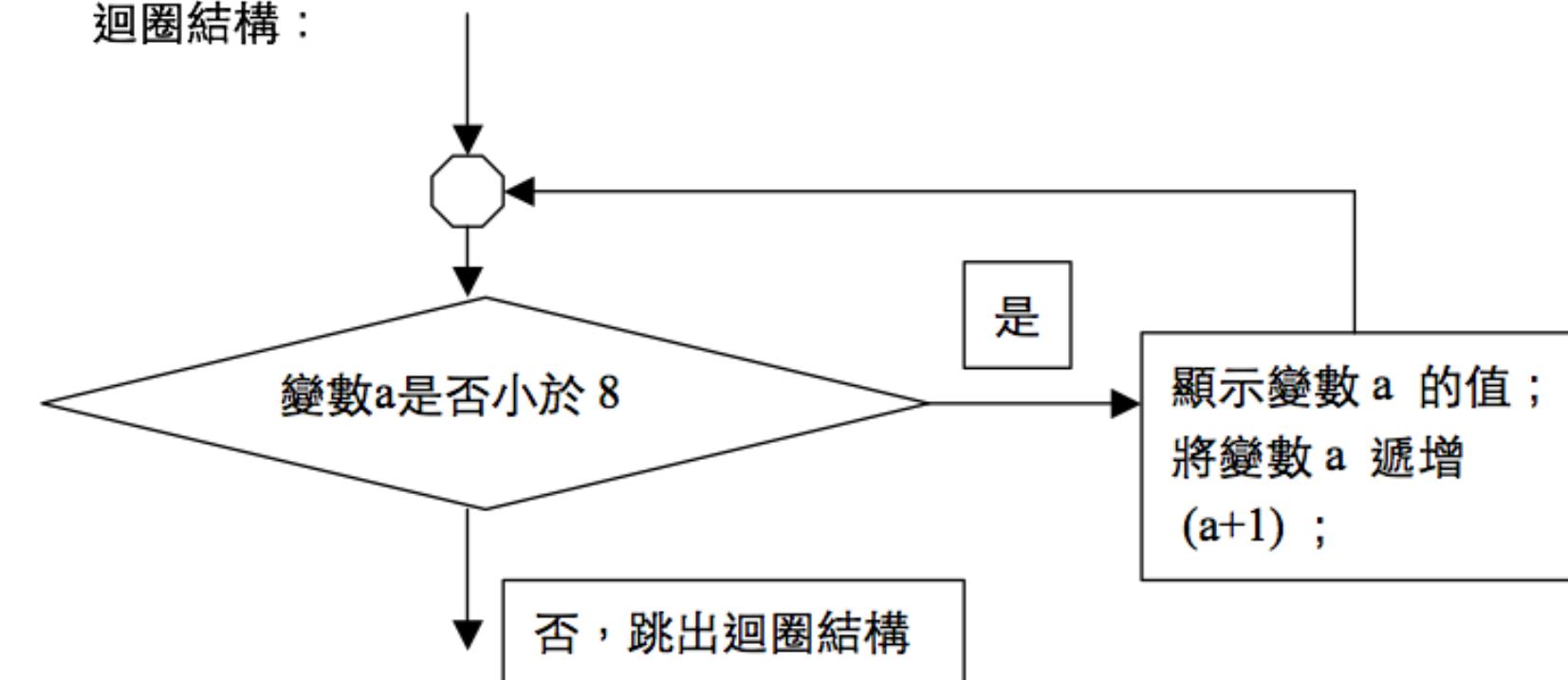
循序結構：  
程式碼第一行；  
程式碼第二行；  
程式碼第三行；  
……  
……  
……

循序結構，就是程式一行一行的由上而下循序執行。

選取結構：



迴圈結構：



# 選取結構if

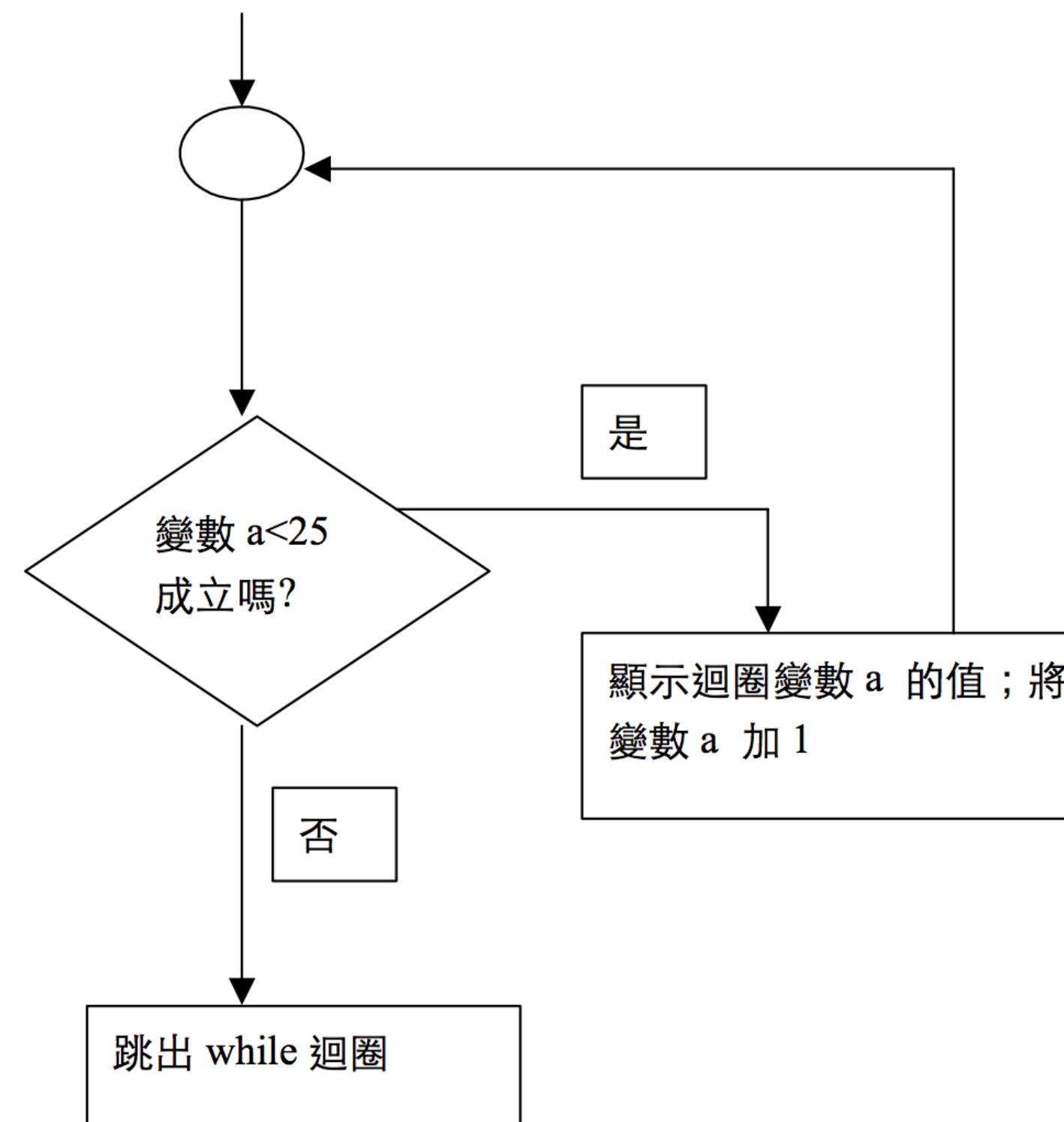
```
8  
9 a = int(input("請輸入薪水:"))  
10 if a < 50000:  
11     print("薪水小於50000")  
12 else:  
13     print("薪水大於50000")
```

The screenshot shows a Jupyter Notebook interface with a single cell containing Python code. The code prompts the user for an input, checks if it's less than 50000, and prints a corresponding message. The cell has been run twice, with the second run's output highlighted in pink. The output window shows the prompt, the user's input '38000', and the resulting output '薪水小於50000'.

```
Python 3.6.2 |Anaconda, Inc.|  
Type "copyright", "credits" or  
IPython 6.1.0 -- An enhanced I  
薪水小於50000請輸入薪水:38000  
  
In [2]:  
  
In [2]: runfile('/Users/justin/  
請輸入薪水:85000  
薪水大於50000  
  
In [3]:
```

# while迴圈

- 在if敘述中，條件後的敘述只執行一次，而在while敘述中，則可執行一次以上。While敘述的程序圖形中.選取結構和循序結構，都只執行程式敘述一次，如果我們要讓同一行程式重複執行好幾遍則要用迴圈敘述。迴圈敘述可以重複執行某一段程式好幾遍，直到條件的不成立才跳出這個迴圈。迴圈敘述：while、do.....while。



## 迴圈結構for,while

迴圈結構for

迴圈結構while

## 迴圈結構for

- 語法:
- for 計數變數 in range(起始值, 終始值):
  - 程式敘述

# range(8,19)為8到18的數值

The image shows a Jupyter Notebook interface with two panes. The left pane contains Python code, and the right pane shows the resulting output in a console window.

**Code (Left Pane):**

```
8
9 for i in range(8,19):
10     print("i的值:",i)
```

**Output (Right Pane):**

```
Python 3.6.2 |Anaconda 4.2.0| (64-bit)
Type "copyright" for copyright info

IPython 6.1.0 -- 

In [1]: runfile(
i的值: 8
i的值: 9
i的值: 10
i的值: 11
i的值: 12
i的值: 13
i的值: 14
i的值: 15
i的值: 16
i的值: 17
i的值: 18
```

## 迴圈結構while

```
8  
9 i=5  
10 while i<=10:  
11     print("i:",i)  
12     i=i+1
```

Console 6/A

Python 3.6.2 |Ana  
Type "copyright",  
IPython 6.1.0 --

In [1]: runfile('  
i: 5  
i: 6  
i: 7  
i: 8  
i: 9  
i: 10

In [2]:

# 布林運算式

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Thu Oct 26 14:30:21 2017
5
6@author: justinwu
7"""
8
9x=True
10y=False
11print(x&y)
12print(x|y)
13if (x&y):
14    print(x&y)
15else:
16    print(x&y)
17z=True
18print(z&y)
19print(z|y)
20
```

In [30]: runfile  
Users/justinwu/  
False  
True  
False  
False  
True

In [31]:

# continue繼續執行迴圈

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Thu Oct 26 14:07:44 2017
5
6@author: justinwu
7"""
8
9for i in range(8,19):
10    if(i%2==1):
11        continue
12    print("i的值:",i)
```

In [19]: ru  
Users/justi  
i: 5  
i: 6  
i: 7  
i: 8  
i: 9

In [20]:

# break跳出while迴圈

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Thu Oct 26 14:09:11 2017
5
6@author: justinwu
7"""
8
9i=5
10x =True
11while x:
12    print("i:",i)
13    i=i+1
14    if(i>=10):
15        break;
16
```

```
In [19]: runfile('
Users/justinwu/Des
i: 5
i: 6
i: 7
i: 8
i: 9
```

```
In [20]:
```



# 5. 函數

```
8 i = 10
9 def f():
10    print(i)
11
12 i = 42
13 f()
```



# 函數與參數

```
14  
15 def addf(x,y):  
16     print(x+y)  
17  
18 i1=23  
19 i2=12  
20 addf(23,12)  
21  
22 
```

# return回傳

```
8 i = 10
9 def f():
10    print(i)
11
12 i = 42
13 f()
14
15 def addf(x,y):
16    print(x+y)
17
18 i1=23
19 i2=12
20 addf(23,12)
21
22 def myreturn(x,y):
23    return x*y
24
25 i3=2
26 i5=5
27 i8=myreturn(i3,i5)
28 print(i8)
29
30 def myreturn(a,x=2,y=3):
31    return a*x*y
32
33 i3=2
34 i5=5
35 i8=myreturn(8,i3,i5)
36 print(i8)
```

The screenshot shows a Jupyter Notebook interface with two code cells. Cell [1] contains the code from the left panel. Cell [2] shows the output:

```
In [1]: runfile(
42
35
10
80

In [2]:
```

# 遞迴函數

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Thu Oct 26 16:25:09 2017
5
6@author: justinwu
7"""

8
9def factorial(n):
10    if(n==0):
11        return 0
12    if(n==1):
13        return 1
14    else:
15        return n*factorial(n-1)
16
17print(factorial(1))
18print(factorial(2))
19print(factorial(3))
```

```
In [2]: runfile('/Us
wdir='/Users/justinw
1
2
6
```

```
In [3]:
```

# 費式係數

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Thu Oct 26 16:28:50 2017
5
6@author: justinwu
7"""
8
9def fibonaci(n):
10    if(n==0):
11        return 0
12    if(n==1):
13        return 1
14    else:
15        return fibonaci(n-1)+fibonaci(n-2)
16
17print(fibonaci(1))
18print(fibonaci(2))
19print(fibonaci(3))
20print(fibonaci(10))
21print(fibonaci(20))
22
```

```
In [2]: runfile
      wdir='/Users/ju
      1
      2
      6
```

```
In [3]:
```



# 6. 類別



```
8 class MyClass:  
9     #範例屬性參考  
10    i=12345  
11  
12  
13 print(MyClass.i)  
14  
15  
16 class Complex:  
17     #實體建構  
18     def __init__(self,realpart,imagpart):  
19         self.r = realpart  
20         self.i = imagpart  
21  
22 x=Complex(3.0,-4.5)  
23 print(x.r,x.i)  
24  
25
```

# 成員屬性與成員方法

```
8  
9 class MyClass2:  
10     #範例屬性參考  
11     i=12345  
12     def f(self):  
13         return 'hello world'  
14  
15 x=MyClass2()  
16 print(x.i)  
17 print(x.f())  
18  
19
```

# 類別和實體變數

- `__init__(self,..)`為建構函數, 實體化物件時會呼叫它

```
>>> class Dog:  
        kind = 'small dog'  
        def __init__(self, name):  
            self.name = name
```

- `self`為自己這個物件

```
>>> d = Dog('small dog')  
>>> e = Dog('very small dog')  
>>> print(d.kind)  
small dog  
>>> print(e.kind)  
small dog  
>>> print(d.name)  
small dog  
>>> print(e.name)  
very small dog  
>>>
```

# `__init__(self)`建構物件, `__del__(self)`解構物件

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Fri Oct 27 04:52:38 2017
5
6@author: justinwu
7"""

8
9class MyClass:
10    i=12345
11
12print(MyClass.i)
13
14class Complex:
15    def __init__(self,realpart,imagepart):
16        self.r=realpart
17        self.i=imagepart
18    def __del__(self):
19        print('delete object')
20x=Complex(3.0,-4.5)
21print(x.r,x.i)
22x=None
```

```
In [4]: runfile('/
wdir=/Users/justi
12345
3.0 -4.5
delete object
```

```
In [5]:
```



## 7. 繼承

- class 子類別(父類別):
  - 敘述1
  - 敘述2



# \_\_為私有存取控制修飾,只有該類別方法才能存取

```
7
8 class Vehicle:
9     def __init__(self, name, engine):
10        self.__name = name
11        self.__engine = engine
12
13    def getName(self):
14        return self.__name
15
16    def getEngine(self):
17        return self.__engine
18
19    def setEngine(self, engine):
20        self.__engine = engine
21
22
23 class Car(Vehicle):
24     def __init__(self, name, engine, electric):
25         super().__init__(name, engine)
26         self.__electric = electric
27
28     def getCarName(self):
29         print("名子"+self.getName())
30         print("引擎"+self.getEngine())
31         print("電動車"+self.__electric)
32
33     def getAuto(self):
34         print("自動駕駛車")
35
36 myCar = Car("特斯拉", "磁電Engine", "電力")
37 myCar.getCarName()
38 print(myCar.getAuto())
```

The screenshot shows a Jupyter Notebook interface with two code cells and their outputs.

**In [2]:**

```
Python 3.6.2 |Anaconda  
Type "copyright", "c  
IPython 6.1.0 -- An  
名子特斯拉  
引擎磁電Engine  
電動車電力  
自動駕駛車  
None
```

**In [2]:**

```
In [2]:
```

# 多重繼承

- class 子類別(父類別1,父類別2,父類別3,...):
  - 敘述1
  - 敘述2
- 當子類別繼承 (inheritance) 超過一個來源的時候，會以寫在最左邊的父類別優先繼承，多個父類別如果有相同名稱的屬性 (attribute) 與方法 (method)，就會以最左邊的父類別優先。

```
8 class Vehicle:
9     def __init__(self, name, engine):
10        self.__name = name
11        self.__engine = engine
12
13    def getName(self):
14        return self.__name
15
16    def getEngine(self):
17        return self.__engine
18
19    def setEngine(self, engine):
20        self.__engine = engine
21
22 class Electric:
23     def __init__(self, PowerElectric):
24        self.__PowerElectric = PowerElectric
25
26     def getPower(self):
27        return self.__PowerElectric
28
29     def setPower(self, PowerElectric):
30        self.__PowerElectric = PowerElectric
```

名子: 特斯拉  
引擎: 磁電 Engine  
電動車: 電力  
自動駕駛車  
In [13]:

```
33
34 class Car(Vehicle,Electric):
35     def __init__(self,name,engine,PowerElectric,auto):
36         super().__init__(name,engine)
37         self.setPower(PowerElectric)
38         self.__Auto = auto
39
40     def getCarName(self):
41         print("名子:"+self.getName())
42         print("引擎:"+self.getEngine())
43         print("電動車:"+self.getPower())
44
45     def getAuto(self):
46         return self.__Auto
47
48
49 myCar = Car("特斯拉","磁電Engine","電力","自動駕駛車")
50 myCar.getCarName()
51 print(myCar.getAuto())
```

# 多型

子類別和父類別  
有同名的  
getEngine()名稱

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3
4class Vehicle:
5    def __init__(self, name, engine):
6        self.__name=name
7        self.__engine=engine
8
9    def getName(self):
10       return self.__name
11    def getEngine(self):
12       return self.__engine
13
14class Car(Vehicle):
15    def __init__(self, name, engine, electric):
16        super().__init__(name, engine)
17        self.__electric=electric
18
19    def getEngine(self):
20       return ("超級")
21
22    def getAuto(self):
23       print("自動駕駛車")
24
25myCar=Car("特斯拉", "磁電Engine", "電力")
26myCar.getAuto()
27print(myCar.getEngine())
```

In [17]: `car_in.py'`  
自動駕駛車  
超級

In [18]:



# 8.異常或錯誤處理

```
>>> while True:  
    try:  
        x = int(input("Please enter a number: "))  
        break  
    except ValueError:  
        print("input error")
```

```
Please enter a number: f  
input error  
Please enter a number: f  
input error  
Please enter a number: ffffff  
input error  
Please enter a number:
```



# 異常或錯誤處理

The screenshot shows a Jupyter Notebook environment with two main panes. The left pane displays a Python script named `exception.py` containing code for handling file operations and integer conversion errors. The right pane shows the output of the code execution, including a variable explorer and a console window.

**Code (exception.py):**

```
1 import sys
2 try:
3     #f = open('myle.txt')
4     f = open('mysql.py')
5     s = f.readline()
6     i = int(s.strip())
7 except OSError as err:
8     print("OS error: {0}".format(err))
9 except ValueError:
10    print("Could not convert data to an integer.")
11 except:
12    print("Unexpected error:", sys.exc_info()[0])
13
```

**Variable Explorer:**

Name	Type	Size	Value
s	str	1	<code>#!/usr/bin/python</code>

**Console Output:**

```
In [5]: runfile('/Users/justinwu/Desktop/exception.py', wdi
OS error: [Errno 2] No such file or directory: 'myle.txt'

In [6]: runfile('/Users/justinwu/Desktop/exception.py', wdi
Could not convert data to an integer.

In [7]:
```

# 使用raise關鍵字丟出例外

```
8 import sys
9
10 def displaySalary(salary):
11     if salary<0:
12         raise ValueError("薪水為正")
13     print("薪水="+str(salary))
14
15 try:
16     #f = open('myle.txt')
17     Salary = eval(input("請輸入薪水:"))
18     displaySalary(Salary)
19 except OSError as err:
20     print("OS error: {0}".format(err))
21 except ValueError:
22     print("錯誤:輸入薪水值為正")
23 except:
24     print("Unexpected error:", sys.exc_info()[0])
25
```

The screenshot shows a Jupyter Notebook interface with a code cell and its corresponding output in the console.

**Code Cell:**

```
Unexpected error: <class 'SyntaxError'>
In [14]: runfile('/Users/justinwu/Desktop/test.py')
請輸入薪水:80000
薪水=80000

In [15]: runfile('/Users/justinwu/Desktop/test.py')
請輸入薪水:x
Unexpected error: <class 'NameError'>
In [16]: runfile('/Users/justinwu/Desktop/test.py')
請輸入薪水:-10000
錯誤:

In [17]: runfile('/Users/justinwu/Desktop/test.py')
請輸入薪水:-10000
錯誤:輸入薪水值為正
```

**Output:**

- In [14]: runfile('/Users/justinwu/Desktop/test.py')  
請輸入薪水:80000  
薪水=80000
- In [15]: runfile('/Users/justinwu/Desktop/test.py')  
請輸入薪水:x  
Unexpected error: <class 'NameError'>
- In [16]: runfile('/Users/justinwu/Desktop/test.py')  
請輸入薪水:-10000  
錯誤:
- In [17]: runfile('/Users/justinwu/Desktop/test.py')  
請輸入薪水:-10000  
錯誤:輸入薪水值為正

# 檔案處理

- `fp=open('檔案名稱','檔案開啟模式')`

模式字串	當開啟檔案已存在	當開啟檔案不存在
r	開啟唯獨的檔案	產生異常錯誤
w	清除檔案內容後寫入	建立寫入檔案
a	開啟檔案從檔尾後開始寫入	建立寫入檔案
r+	開啟讀寫的檔案	產生錯誤
w+	清除檔案內容後讀寫內容	建立讀寫檔案
a+	從檔案尾巴開始讀寫	建立讀寫檔案

# 開啟, 關閉及寫入檔案

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Sat Oct 28 06:39:38 2017
5
6@author: justinwu
7"""
8fp=open('file.txt','w')
9if fp !=None:
10    print('檔案開啟成功')
11fp.close()
12
13fp=open('file.txt','w')
14if fp !=None:
15    fp.write("小白")
16fp.close()
17
18fp=open('file.txt','w')
19if fp !=None:
20    fp.write("宇哲")
21fp.close()
```

In [8]: runfile  
Users/justinwu/  
檔案開啟成功

In [9]:

# 讀取檔案

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Sat Oct 28 06:46:03 2017
5
6@author: justinwu
7"""
8
9fp=open('file.txt','r')
10if fp !=None:
11    str=fp.read()
12    print(str)
13fp.close()
14
15fp=open('file.txt','r')
16if fp !=None:
17    strList=fp.readlines()
18    print(strList)
19fp.close()
```

```
In [11]: runfile
Users/justinwu/
宇哲
['宇哲']

In [12]:
```



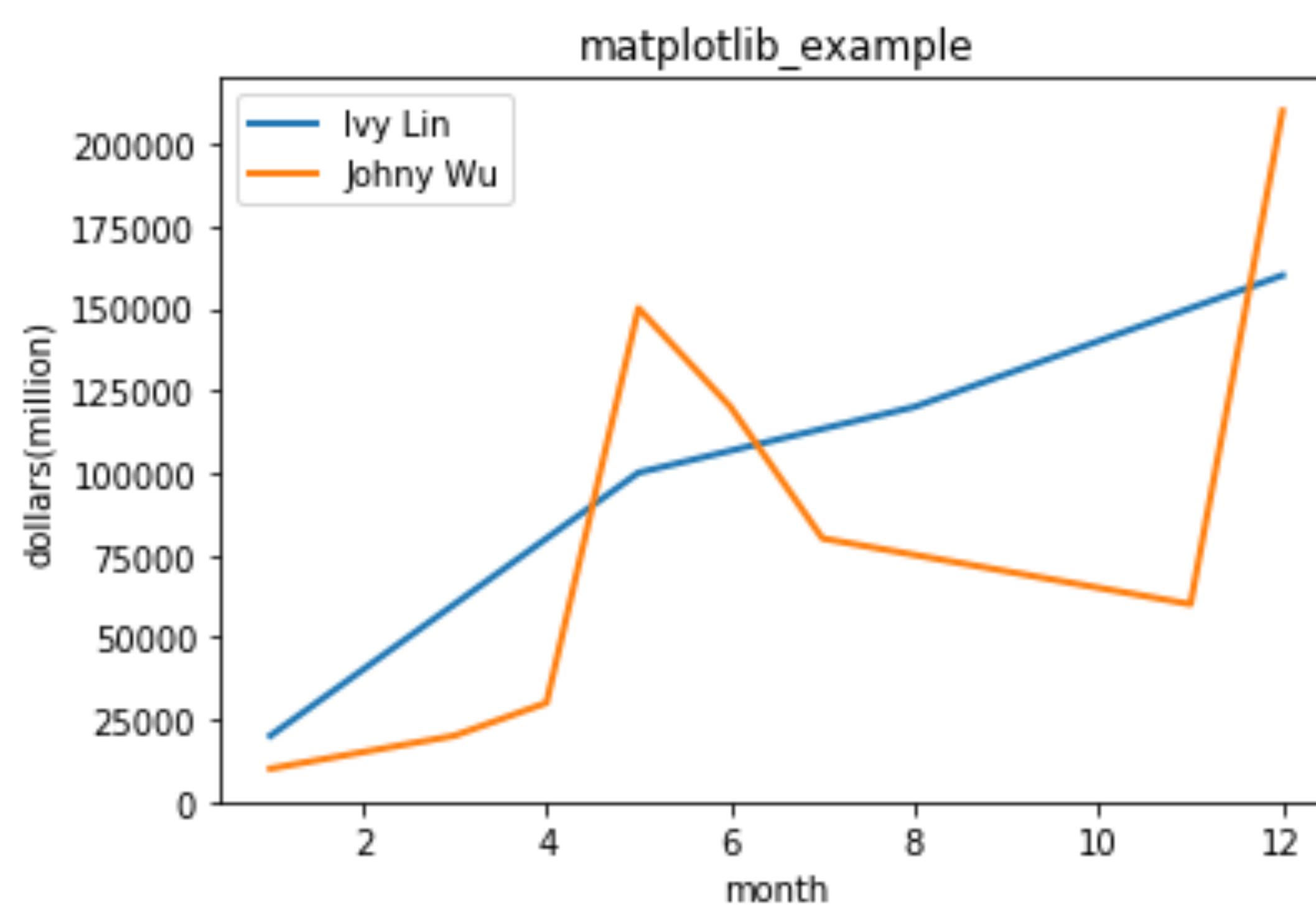
# 9. 使用matplotlib畫圖

- Matplotlib.pyplot是畫圖的命令集合函數.每一個pyplot函數可以建立或修改圖形



# 使用matplotlib畫圖

```
6 @author: justinwu
7 """
8 import matplotlib.pyplot as plt
9
10 month1 = [1,2,3,4,5,8,10,12]
11 month2 = [1,3,4,5,6,7,11,12]
12 sales1 = [20000,40000,60000,80000,100000,120000,140000,160000]
13 sales2 = [10000,20000,30000,150000,120000,80000,60000,210000]
14
15 plt.plot(month1,sales1,lw=2,label='Ivy Lin')
16 plt.plot(month2,sales2,lw=2,label='Johny Wu')
17 plt.xlabel('month')
18 plt.ylabel('dollars(million)')
19 plt.legend()
20 plt.title('matplotlib_example')
21 plt.show()
```



`plt.plot([1,2,3,4])`預設是X軸,而Y軸  
是我們輸入的資料串列[1,2,3,4].

The screenshot shows the Spyder IDE interface with the following details:

- Editor:** The code file is named `matplotlib_pyplot.py`. The code content is as follows:

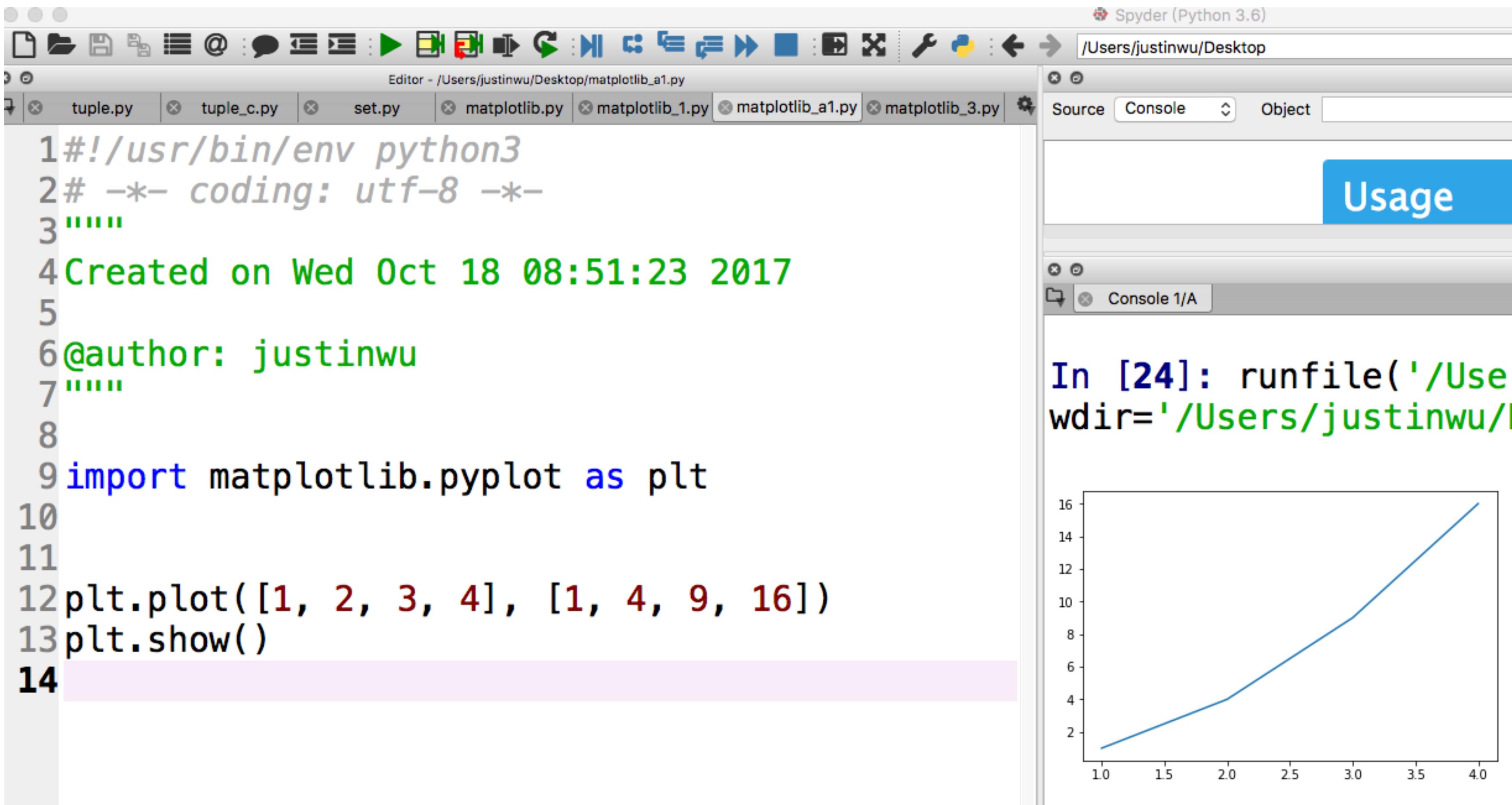
```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Wed Oct 18 09:20:28 2017
5
6@author: justinwu
7"""
8
9import matplotlib.pyplot as plt
10plt.plot([1, 2, 3, 4])
11plt.ylabel('some numbers')
12plt.show()
13
14
```

- Console:** The console output shows the command runfile and the resulting plot.

```
In [19]: runfile('/Users/justinwu/Desktop/matplotlib_pyplot.py', wd:
Usage
```

- Plot:** A line plot titled "some numbers" is displayed. The x-axis ranges from 0.0 to 3.0, and the y-axis ranges from 1.0 to 4.0. The plot shows a straight line starting at (0, 1) and ending at (3, 4).

第一個[1,2,3,4]參數是X軸,第二個  
參數是Y軸



The screenshot shows the Spyder Python IDE interface. The left pane is the code editor for 'matplotlib\_a1.py' with the following content:

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Wed Oct 18 08:51:23 2017
5
6@author: justinwu
7"""
8
9import matplotlib.pyplot as plt
10
11
12plt.plot([1, 2, 3, 4], [1, 4, 9, 16])
13plt.show()
14
```

The right pane shows the console output and a plot window. The console output includes:

```
In [24]: runfile('/Users/justinwu/Desktop/matplotlib_a1.py', wdir='/Users/justinwu/')
```

The plot window displays a line graph with X-axis values [1, 2, 3, 4] and Y-axis values [1, 4, 9, 16]. The plot area has a light gray background with white grid lines. The X-axis ranges from 1.0 to 4.0 with major ticks every 0.5 units. The Y-axis ranges from 2 to 16 with major ticks every 2 units. A single blue line connects the points (1, 1), (2, 4), (3, 9), and (4, 16).

# plot()第三個參數是格式字串點 plot,'ro'為顯示紅色圓圈

The screenshot shows the Spyder Python IDE interface. On the left, the code editor displays a file named `plot_ro.py` with the following content:

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Wed Oct 18 09:34:44 2017
5
6@author: justinwu
7"""
8import matplotlib.pyplot as plt
9#第三個參數是格式字串點plot,'ro'為顯示紅色圓圈
10plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro')
11plt.axis([0, 6, 0, 20])
12plt.show()
```

On the right, the console window shows the output of running the script:

```
In [30]: runfile('/Users/justinwu/Desktop/python/example/plot_ro.py', wdir='/Users/justinwu/Desktop/python/example')
```

A plot window is displayed, showing four red circular markers at the coordinates (1, 1), (2, 4), (3, 9), and (4, 16).

# # 'r--'紅色虛線,'bs'藍色矩形,'g^'綠色三角形

The screenshot shows the Spyder IDE interface with the following details:

- Editor:** The code editor window displays the file `plot_s.py` with the following content:

```
1#!/usr/bin/env python3
2# -*- coding: utf-8 -*-
3"""
4Created on Wed Oct 18 09:42:54 2017
5
6@author: justinwu
7"""

8
9import numpy as np
10
11# 0到5每步0.2
12t = np.arange(0., 5., 0.2)
13
14# red dashes, blue squares and green triangles
15# 'r--'紅色虛線,'bs'藍色矩形,'g^'綠色三角形
16plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3,
17plt.show()
```
- Console:** The console window shows the command `In [33]: runfile('/Users/justinwu/Desktop/python/example/plot_s.py', wdir='/Users/justinwu/Desktop/python/example')` and the resulting plot.
- Plot:** The plot shows three data series:
  - A red dashed line (`'r--'`) representing  $y = x$ .
  - Blue square markers (`'bs'`) representing  $y = x^2$ .
  - Green triangle markers (`'g^'`) representing  $y = x^3$ .The x-axis ranges from 0 to 5, and the y-axis ranges from 0 to 100.



- Thanks.

