Python一天學會

吳佳諺 老師

一天學會 python

Python課程架構一目了然!

吳老師教學中心

1. Python 程式基礎

1. Python 軟體
2. Python 直譯器與計算機
   1. 資料型態
   2. 運算式與運算子
   3. 運算子結合優先順序

```
print(1+2*3-6/2+5)
```

   4. 直譯器解析區塊,空四格

2. Python 資料結構

1. 串列,集合set和字典

```
alist=〔1,3,5,8,12〕
```

2. 數組

3. Python 控制流程與函數

1. 控制結構
   1. if...elif...else...
   2. for...
   3. while

```
x = 5
if x > 90:
    print("x為A")
elif x > 60 and x < 90 :
    print("x為OK")
else :
    print("PASS")

alist=[1,3,5,8,12]
for i in alist :
    print(i)
```

2. 函數
   1. 參數與引數
   2. 遞迴函數
   3. 套件模組

```
def add(x,y):
    print('ok')
    return x+y

print(add(3,5))
```

```
def factorial(n):
    if n == 1 :
        return 1
    elif n == 0 :
        return 1
    else :
        return n*factorial(n-1)

print(factorial(3))
```

4. Python 物件導向

1. 類別(封裝)
   1. 建構函_init_(self)
   2. 解構函數_del_(self)
2. 繼承
   1. 子類別繼承父類別
   2. 多重繼承
3. 多型
   1. 同名異式
   2. 類別繼承自object

```
class MyClass(object):
    def __init__(self):
        print('Constructor')

class ChildClass(object):
    def __init__(self):
        print('建構函數')

    def __del__(self):
        print('解構函數')


MyObject = MyClass()
MyChild = ChildClass()
del MyChild
```

5. Python 異常或錯誤處理

1. try...except

```
try:
    x = int(input("請輸入非0數字:"))
    z = 2/x
    print(z)
except :
    print("輸入0,除法異常")
```

吳老師教學中心

了解詳情:https://www.justinwu.com.tw/

1.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

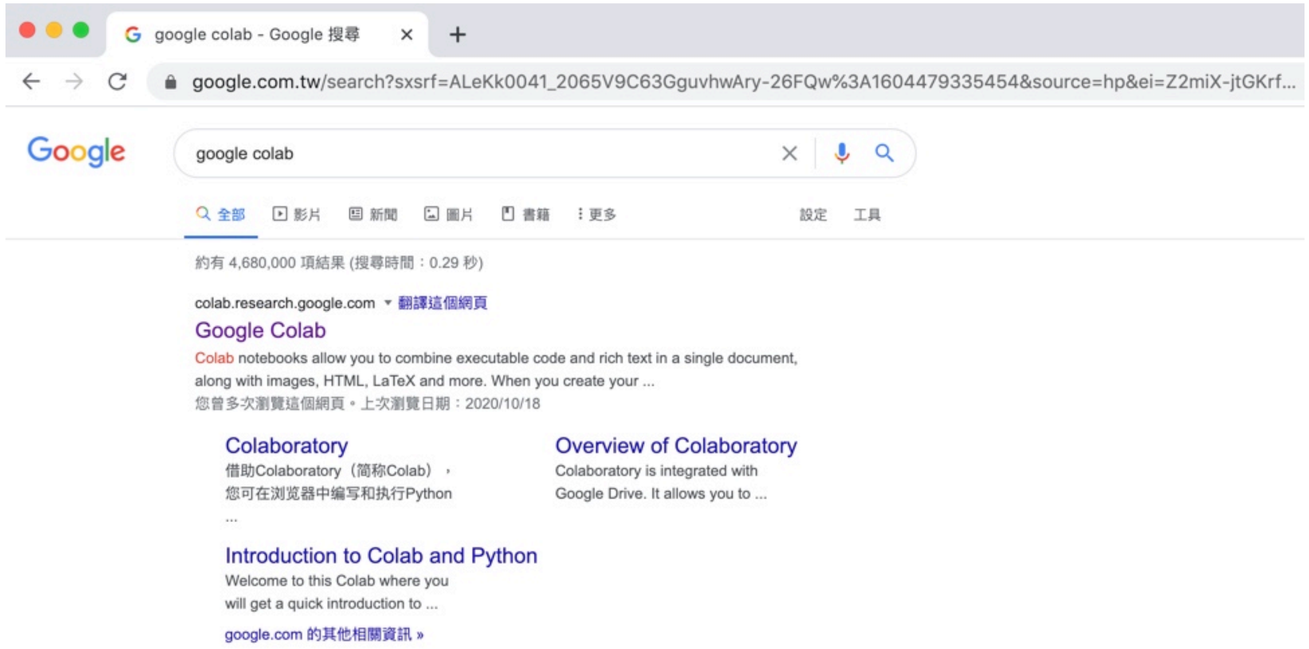
1.Python軟體

- 1-1 Google Colab
- 1-2 下載及安裝Python軟體
- 1-3 Mac安裝Anaconda
- 1-4 Windows安裝Anaconda

1-1Google Colab平台

- Colaboratory (簡稱為「Colab」) 可在瀏覽器上撰寫及執行 Python。
- Colab 可使用熱門 Python 程式庫，對資料進行分析並以視覺化方式呈現。
- 只要有Google gmail 帳號就可以使用。
- 可執行Python機器學習範例
- 可執行Python深度學習範例

搜尋Google Colab



選取檔案->新增筆記本


歡迎使用 Colaboratory - Colab
colab.research.google.com/notebooks/intro.ipynb#scrollTo=lSrWNr3MuFUS
歡迎使用 Colaboratory
檔案 編輯 檢視畫面 插入 執行階段 工具 說明
共用
新增筆記本
開啟筆記本 ⌘/Ctrl+O
上傳筆記本
重新命名筆記本
移至垃圾桶
在雲端硬碟中儲存副本
將副本另存為 GitHub Gist
在 GitHub 中儲存副本
儲存 ⌘/Ctrl+S
儲存及固定修訂版本 ⌘/Ctrl+M S
修訂版本記錄
下載 .ipynb
下載 .py
更新雲端硬碟中的預覽畫面
程式碼 + 文字 複製到雲端硬碟
連線
編輯
什麼是 Colaboratory?
olaboratory (簡稱為「Colab」) 可讓你在瀏覽器上撰寫及執行 Python，且具備下列優點：
不必進行任何設定
免費使用 GPU
輕鬆共用
無論你是學生、數據資料學家或是 AI 研究人員，Colab 都能讓你的工作事半功倍。請觀看 Colab 的簡介影片瞭解詳情，或是直接瀏覽以下的新手入門說明！
開始使用
你正在閱讀的文件並非靜態網頁，而是名為 Colab 筆記本的互動式環境，可讓你撰寫和執行程式碼。
舉例來說，以下是包含簡短 Python 指令碼的程式碼儲存格，可進行運算、將值儲存至變數中並列印運算結果：

輸入print(1+1)


Hello.ipynb - Colaboratory
colab.research.google.com/drive/1bLj_Dk0wIkiVu60gbKhvws4ClpAn0LaM
Hello.ipynb
檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更
+ 程式碼 + 文字
1 print(1+1)

執行程式

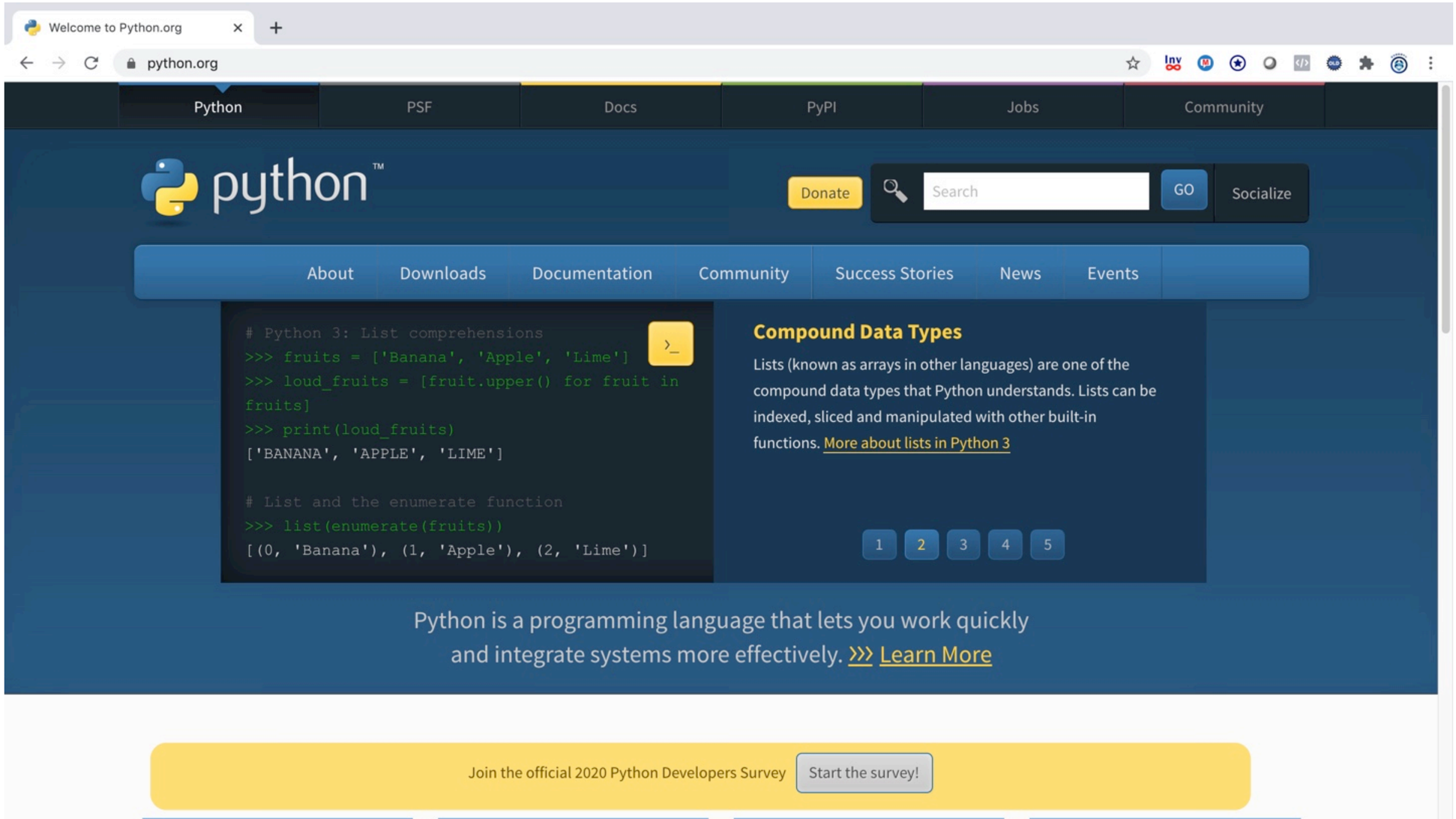

Hello.ipynb - Colaboratory
colab.research.google.com/drive/1bLj_Dk0wIkiVu60gbKhvws4ClpAn0LaM
Hello.ipynb
檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更
+ 程式碼 + 文字
1 print(1+1)
2

1-2.下載及安裝Python軟體

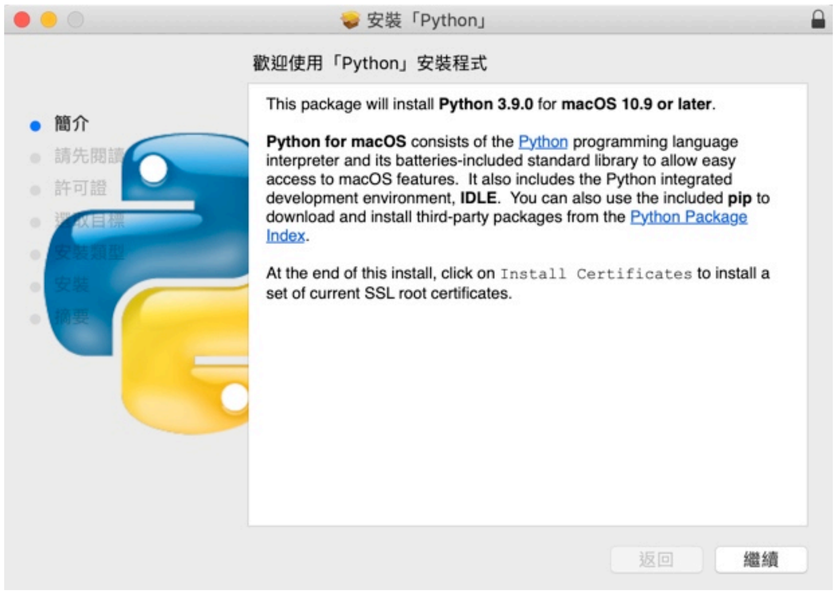
到python.org下載軟體

到Aanconda下載Python組合

到python.org下載軟體



Mac安裝過程


安裝「Python」
歡迎使用「Python」安裝程式
簡介
請先閱讀
許可證
選取目標
安裝類型
安裝
摘要
This package will install Python 3.9.0 for macOS 10.9 or later.
Python for macOS consists of the Python programming language interpreter and its batteries-included standard library to allow easy access to macOS features. It also includes the Python integrated development environment, IDLE. You can also use the included pip to download and install third-party packages from the Python Package Index.
At the end of this install, click on Install Certificates to install a set of current SSL root certificates.
返回
繼續

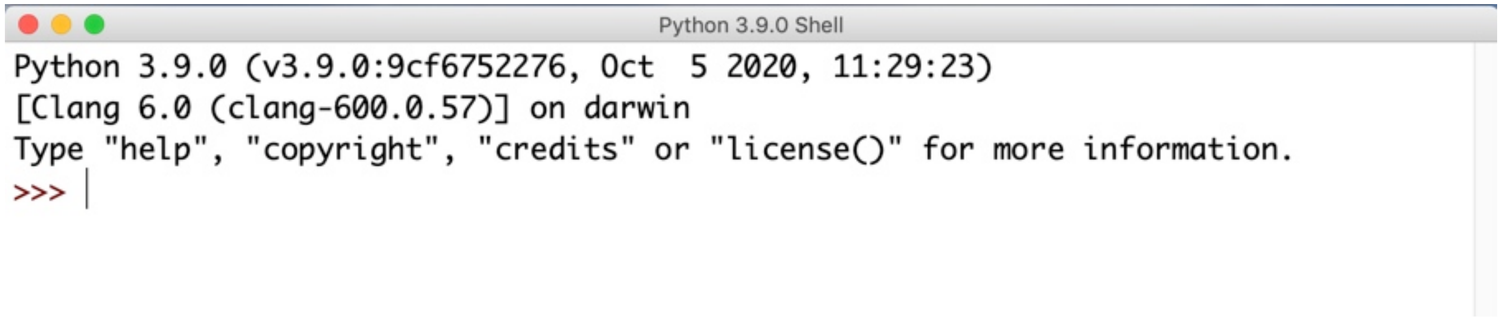

安裝「Python」
重要資訊
簡介
請先閱讀
許可證
選取目標
安裝類型
安裝
摘要
This package will install Python 3.9.0 for macOS 10.9 or later for the following architecture(s): x86_64.
Certificate verification and OpenSSL
This package includes its own private copy of OpenSSL 1.1.1. The trust certificates in system and user keychains managed by the Keychain Access application and the security command line utility are not used as defaults by the Python ssl module. A sample command script is included in /Applications/Python 3.9 to install a curated bundle of default root certificates from the third-party certifi package (https://pypi.org/project/certifi/). Double-click on Install Certificates to run it.
The bundled pip has its own default certificate store for verifying download connections.
Using IDLE or other Tk applications
This package includes its own private version of Tcl/Tk 8.6. It does not use any system-supplied or third-party supplied versions of Tcl/Tk.
列印⋯
儲存⋯
返回
繼續


安裝「Python」
在「Macintosh HD」上進行標準安裝
簡介
請先閱讀
許可證
選取目標
安裝類型
安裝
摘要
這將會佔用您電腦的 119.5 MB 空間。
請按一下「安裝」來在磁碟「Macintosh HD」上執行此軟體的標準安裝。
更改安裝位置⋯
自定
返回
安裝

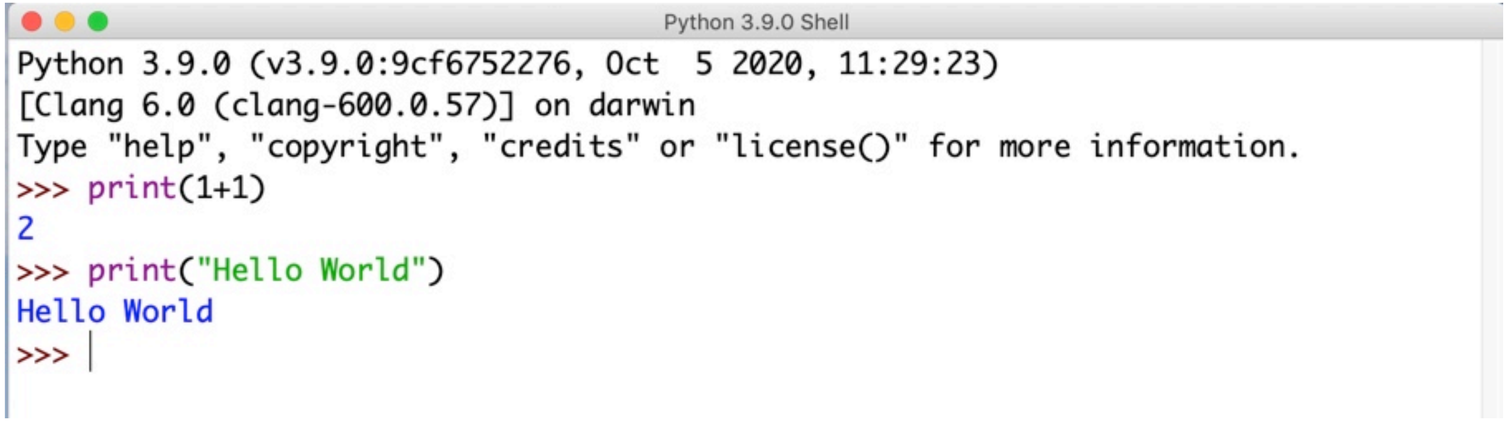
Python安裝

- 這是在Mac上安裝完的Python Launcher

這是安裝完Python的直譯環境

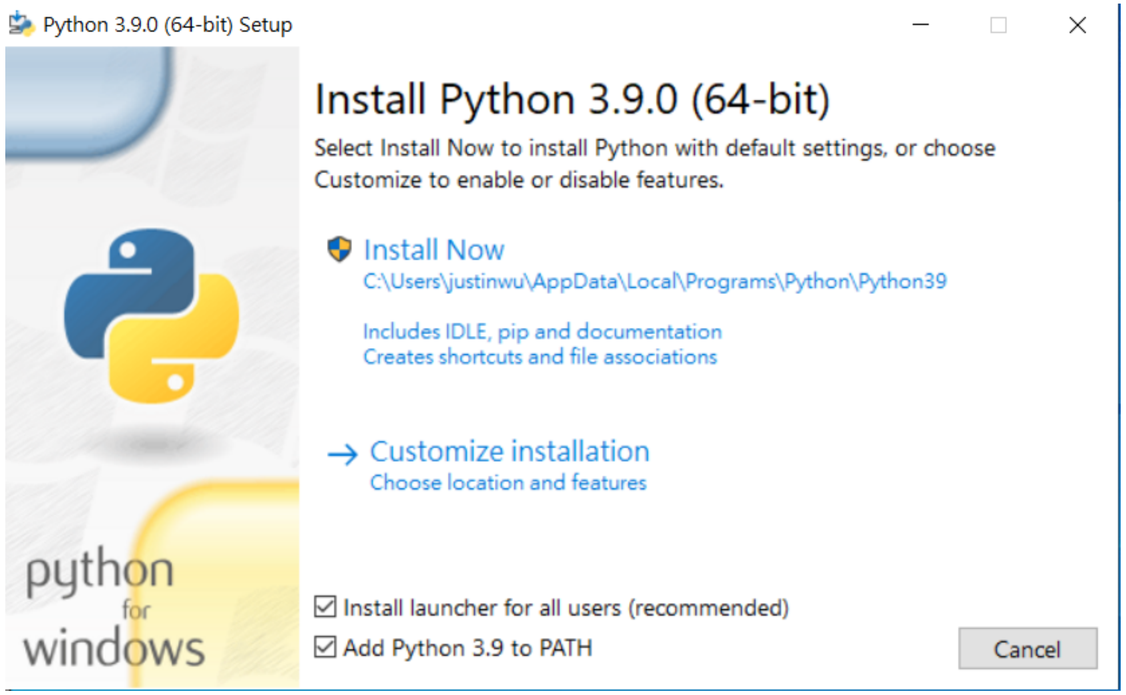


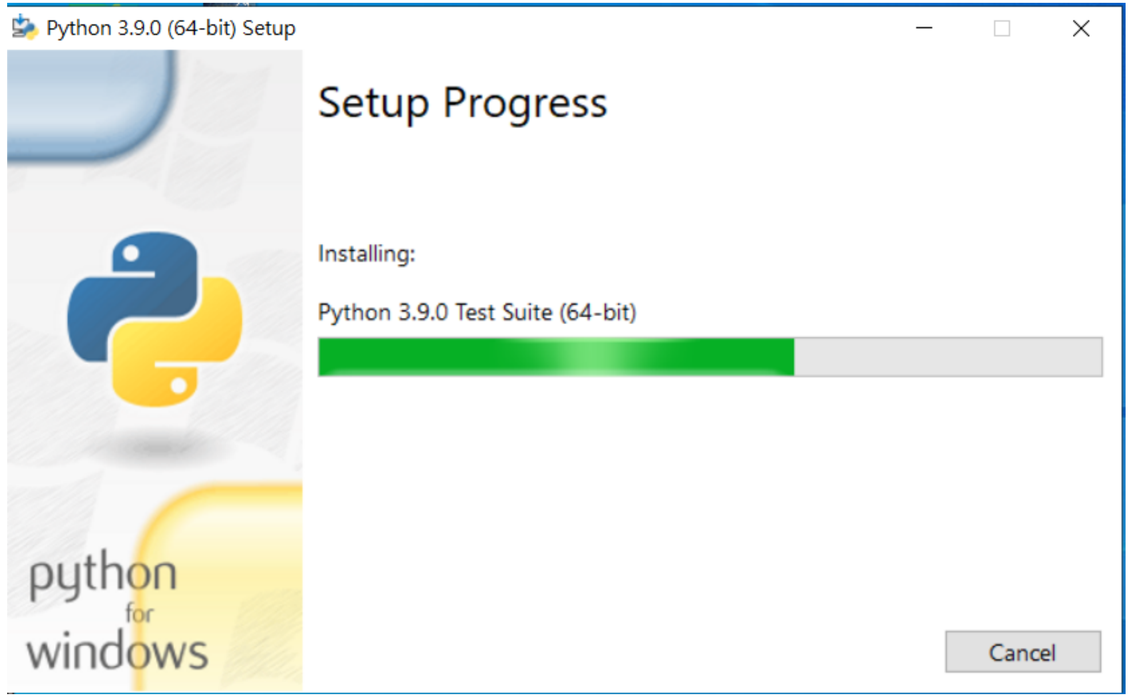
第一個程式”Hello World”

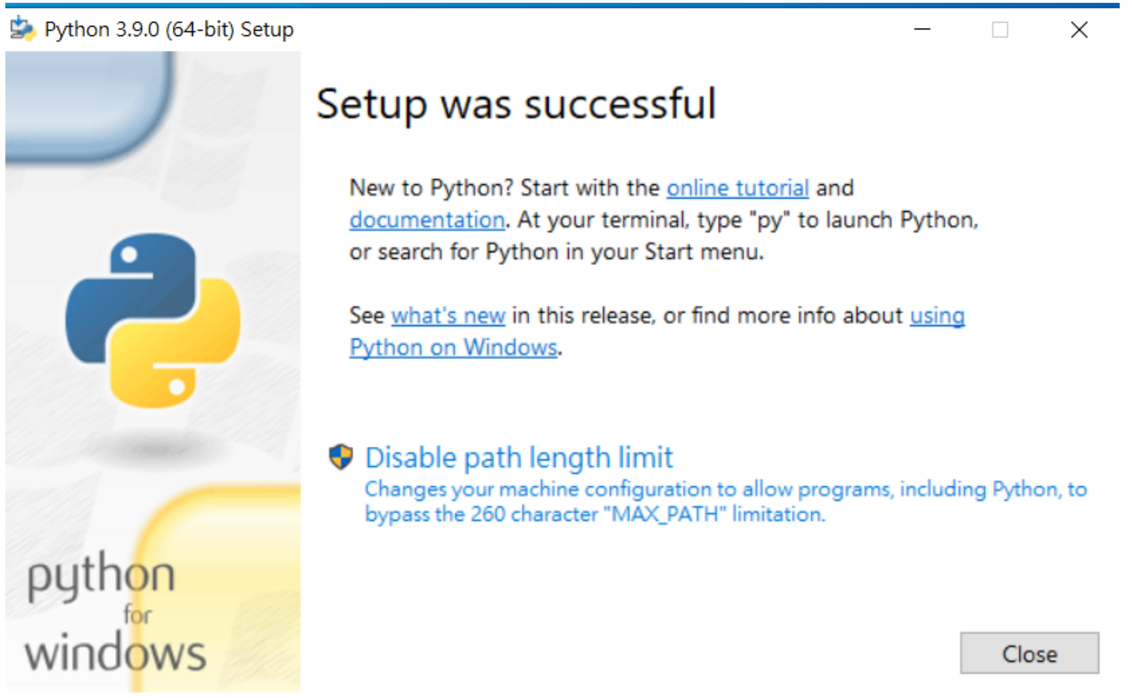


Windows安裝

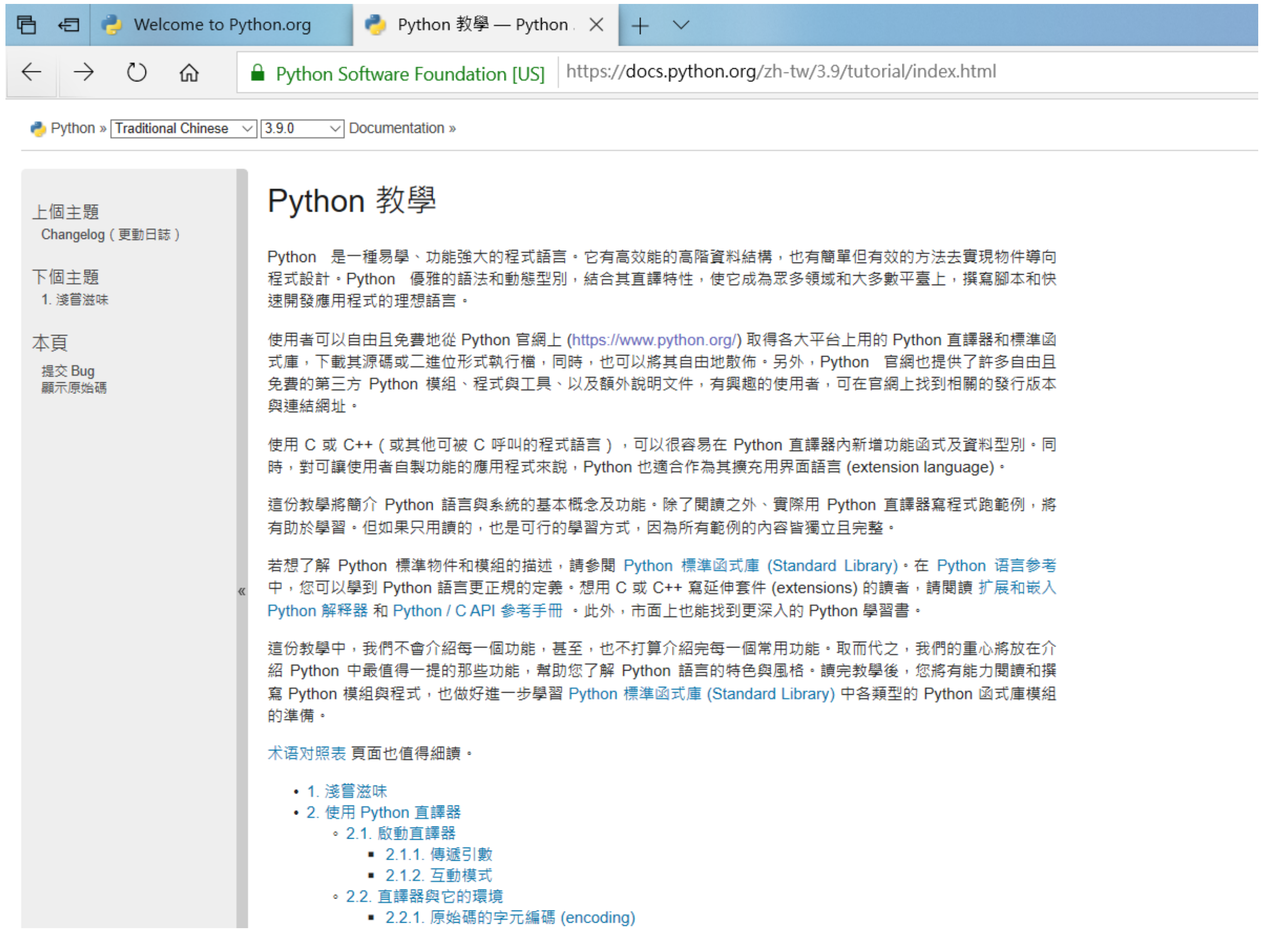
- 將Python路徑加入



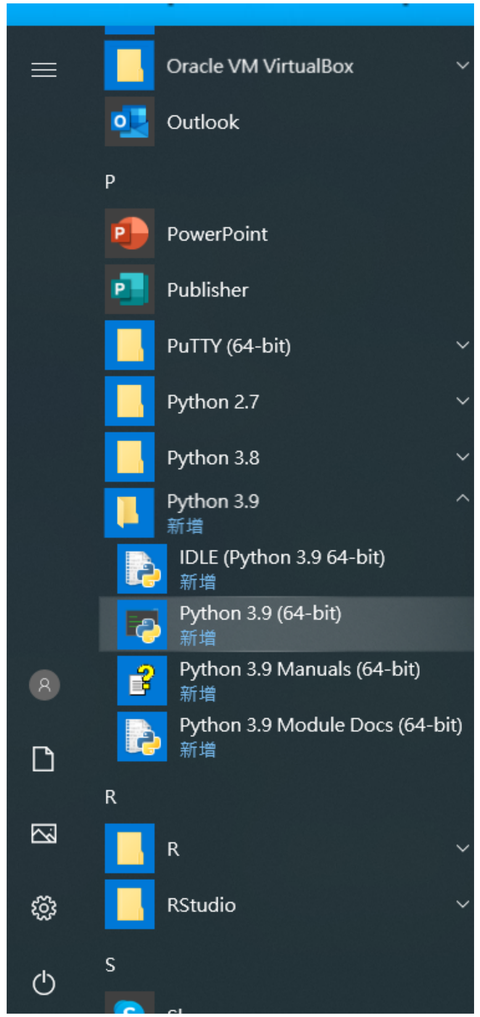

Python 3.9.0 (64-bit) Setup
Setup Progress
Installing:
Python 3.9.0 Test Suite (64-bit)
Cancel
python
for
windows


Python 3.9.0 (64-bit) Setup
Setup was successful
New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.
See what's new in this release, or find more info about using Python on Windows.
Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.
Close
python
for
windows

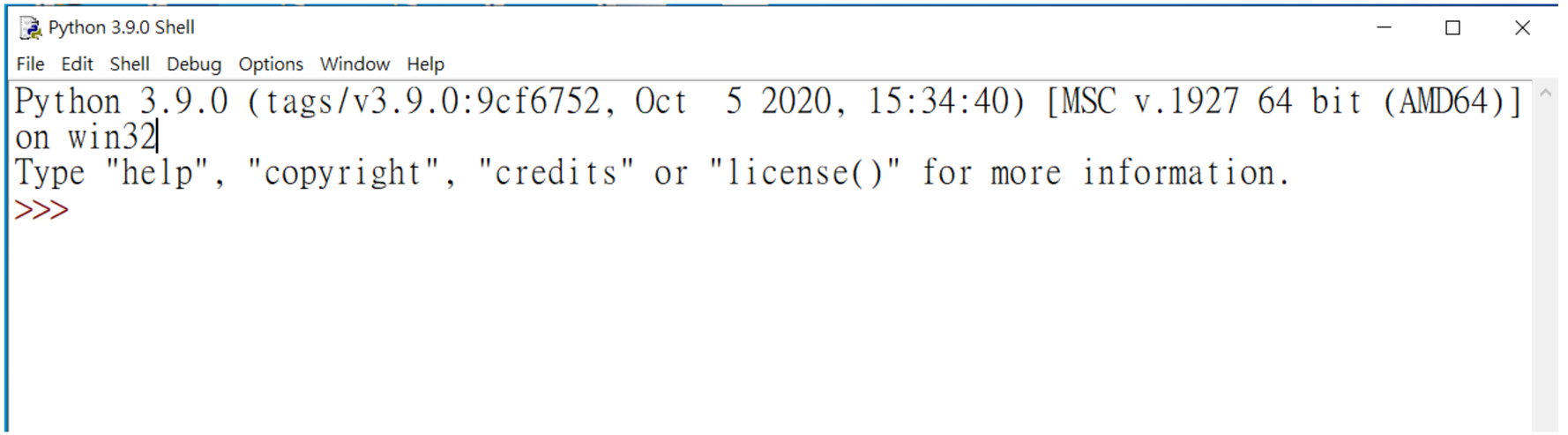
成功安裝Python

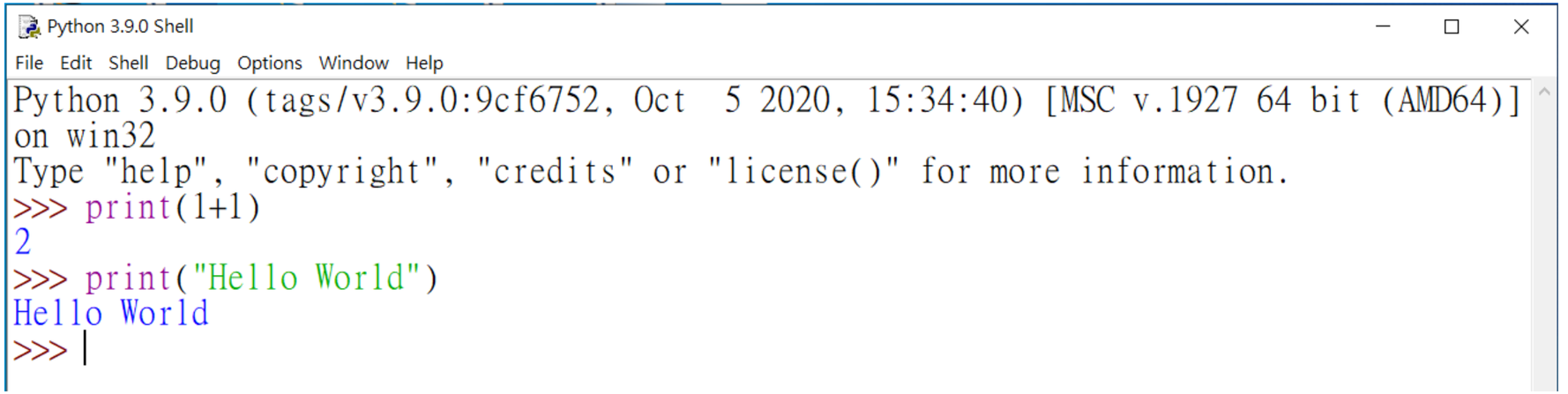

Welcome to Python.org
Python 教學 — Python
Python Software Foundation [US]
https://docs.python.org/zh-tw/3.9/tutorial/index.html
Python » Traditional Chinese 3.9.0 Documentation »
上個主題
Changelog（更動日誌）
下個主題
1. 淺嘗滋味
本頁
提交 Bug
顯示原始碼
Python 教學
Python 是一種易學、功能強大的程式語言。它有高效能的高階資料結構，也有簡單但有效的方法去實現物件導向程式設計。Python 優雅的語法和動態型別，結合其直譯特性，使它成為眾多領域和大多數平臺上，撰寫腳本和快速開發應用程式的理想語言。
使用者可以自由且免費地從 Python 官網上 (https://www.python.org/) 取得各大平台上用的 Python 直譯器和標準函式庫，下載其源碼或二進位形式執行檔，同時，也可以將其自由地散佈。另外，Python 官網也提供了許多自由且免費的第三方 Python 模組、程式與工具、以及額外說明文件，有興趣的使用者，可在官網上找到相關的發行版本與連結網址。
使用 C 或 C++（或其他可被 C 呼叫的程式語言），可以很容易在 Python 直譯器內新增功能函式及資料型別。同時，對可讓使用者自製功能的應用程式來說，Python 也適合作為其擴充用界面語言 (extension language)。
這份教學將簡介 Python 語言與系統的基本概念及功能。除了閱讀之外、實際用 Python 直譯器寫程式跑範例，將有助於學習。但如果只用讀的，也是可行的學習方式，因為所有範例的內容皆獨立且完整。
若想了解 Python 標準物件和模組的描述，請參閱 Python 標準函式庫 (Standard Library)。在 Python 语言参考 中，您可以學到 Python 語言更正規的定義。想用 C 或 C++ 寫延伸套件 (extensions) 的讀者，請閱讀 扩展和嵌入 Python 解释器 和 Python / C API 参考手冊 。此外，市面上也能找到更深入的 Python 學習書。
這份教學中，我們不會介紹每一個功能，甚至，也不打算介紹完每一個常用功能。取而代之，我們的重心將放在介紹 Python 中最值得一提的那些功能，幫助您了解 Python 語言的特色與風格。讀完教學後，您將有能力閱讀和撰寫 Python 模組與程式，也做好進一步學習 Python 標準函式庫 (Standard Library) 中各類型的 Python 函式庫模組的準備。
术语对照表 頁面也值得細讀。
1. 淺嘗滋味
2. 使用 Python 直譯器
2.1. 啟動直譯器
2.1.1. 傳遞引數
2.1.2. 互動模式
2.2. 直譯器與它的環境
2.2.1. 原始碼的字元編碼 (encoding)

在選單啟動Python3.9



執行Python程式

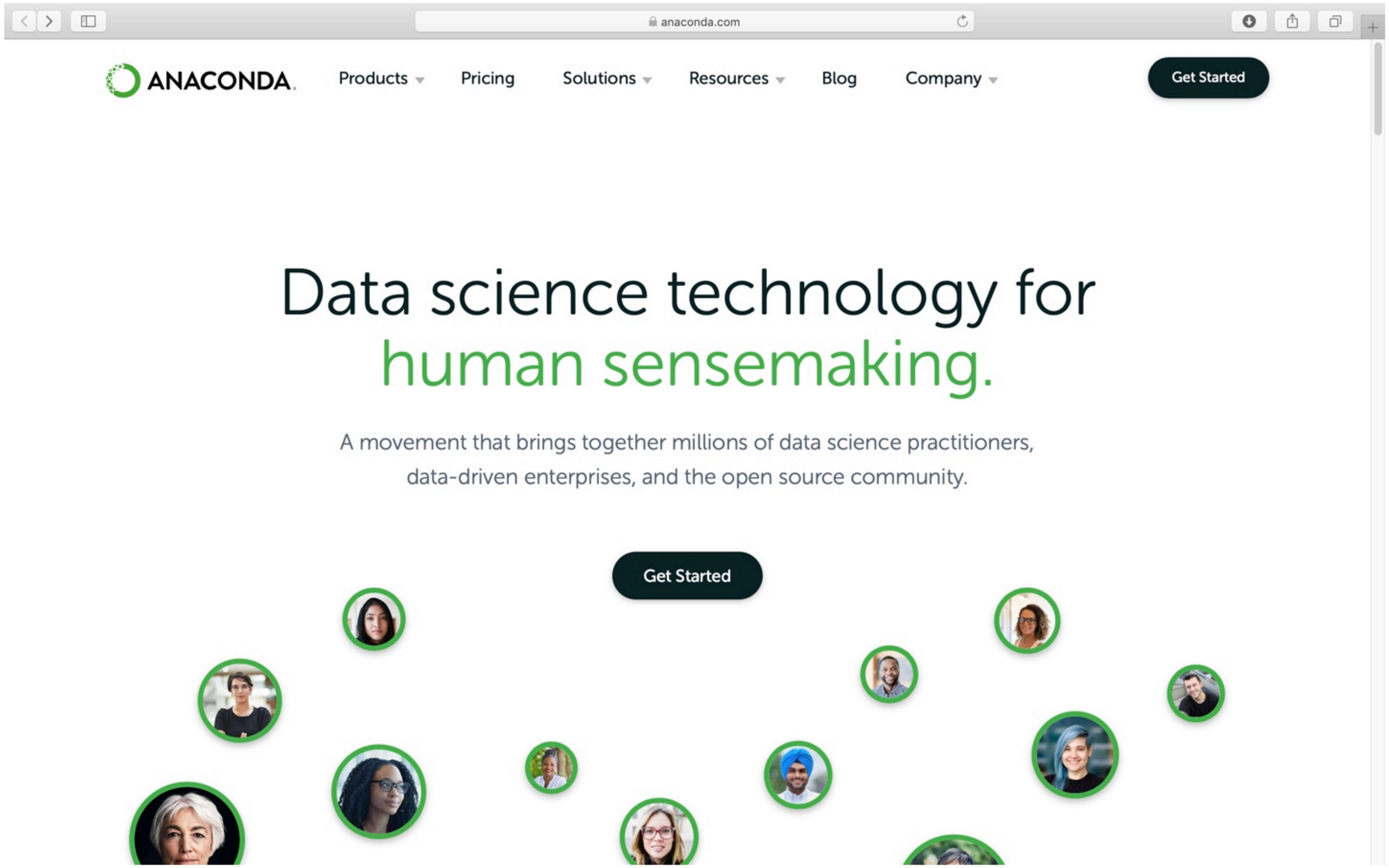

Python 3.9.0 Shell
File Edit Shell Debug Options Window Help
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct 5 2020, 15:34:40) [MSC v.1927 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>


Python 3.9.0 Shell
File Edit Shell Debug Options Window Help
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct 5 2020, 15:34:40) [MSC v.1927 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print(1+1)
2
>>> print("Hello World")
Hello World
>>>

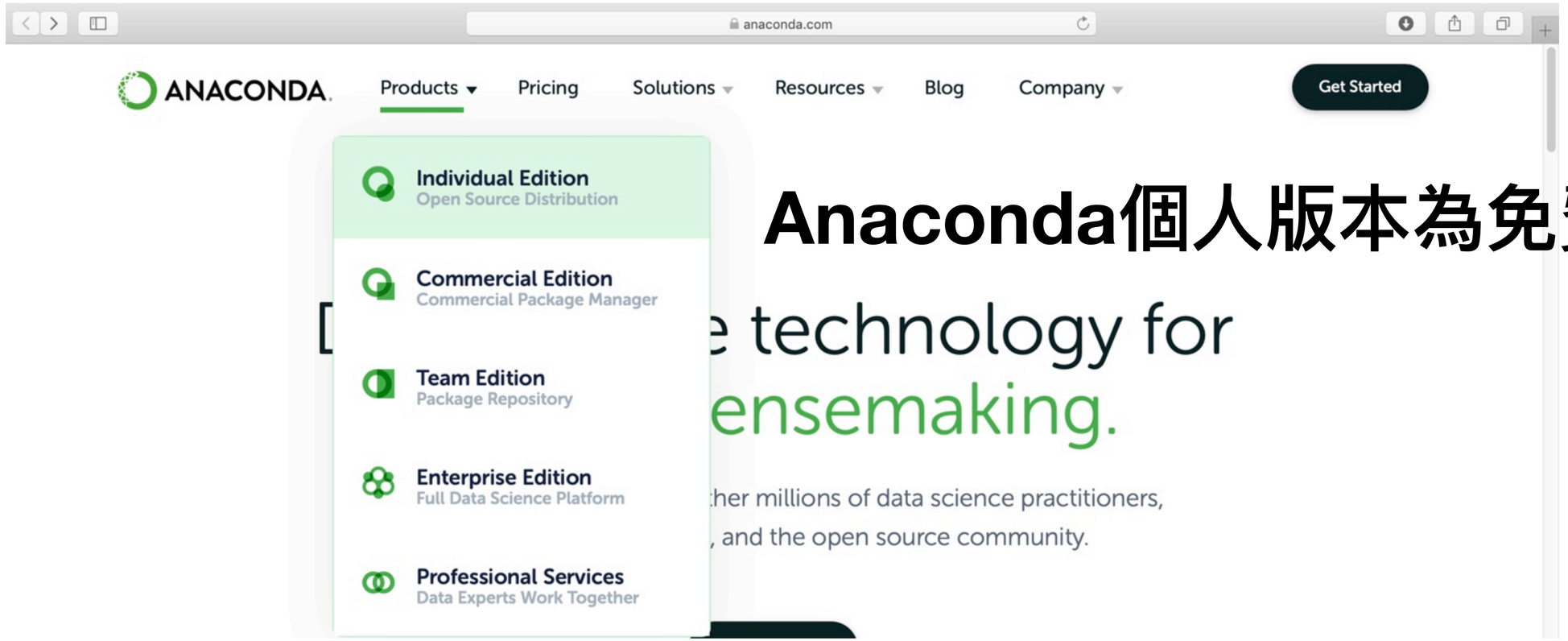
安裝
Anaconda,Numpy,Matplotilib

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

- 到Anconda下載Python組合包https://www.anaconda.com/



選取Products->Individual Edition

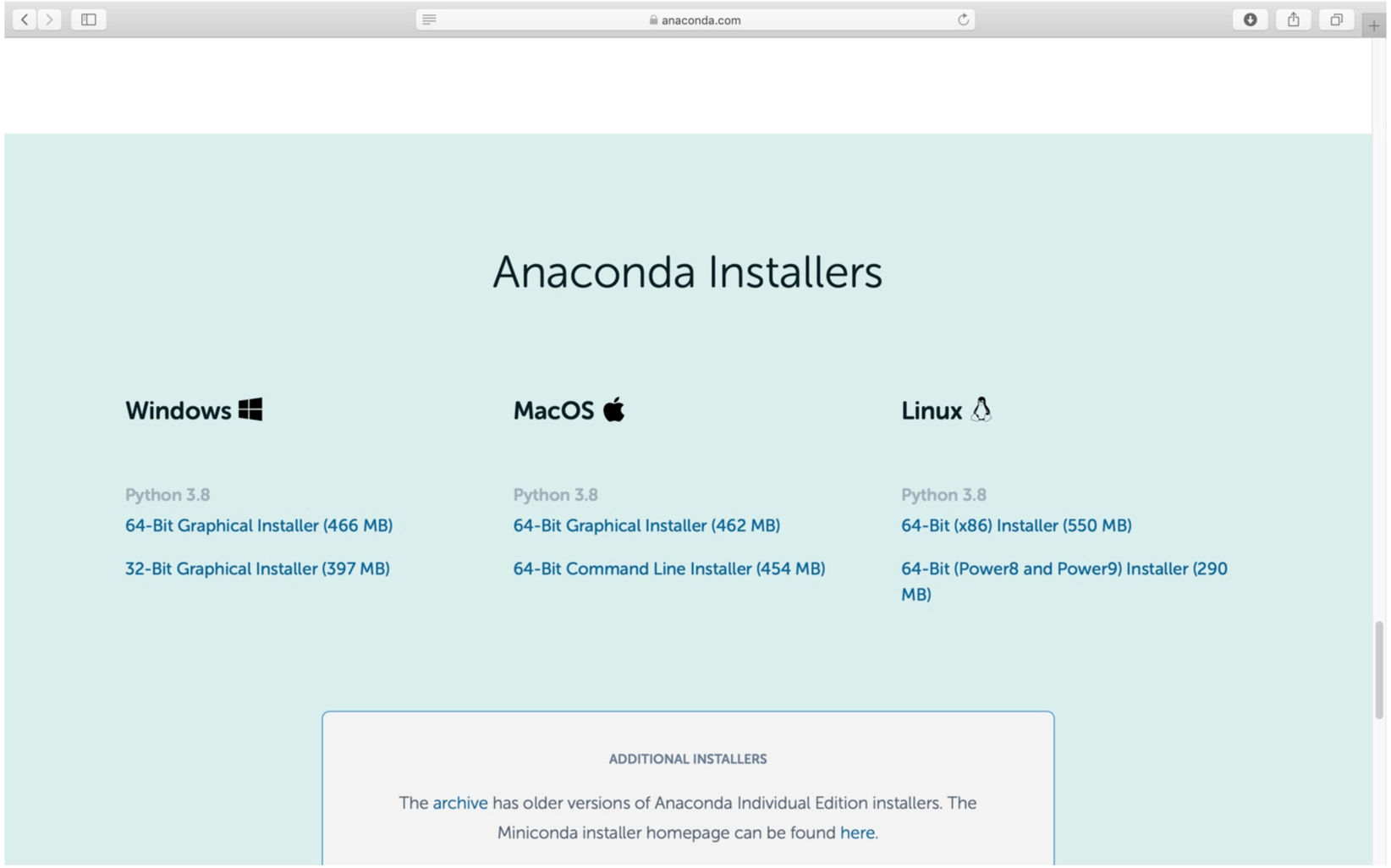


Anaconda個人版本為免費

1-3Mac安裝Anaconda

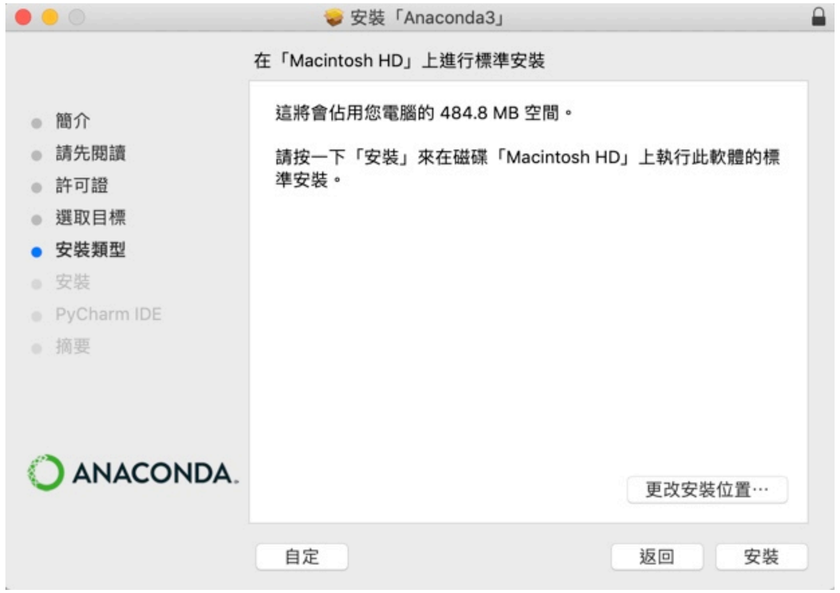
- 下載Anaconda安裝Installers下載Anaconda安裝Installers
- 執行Spider

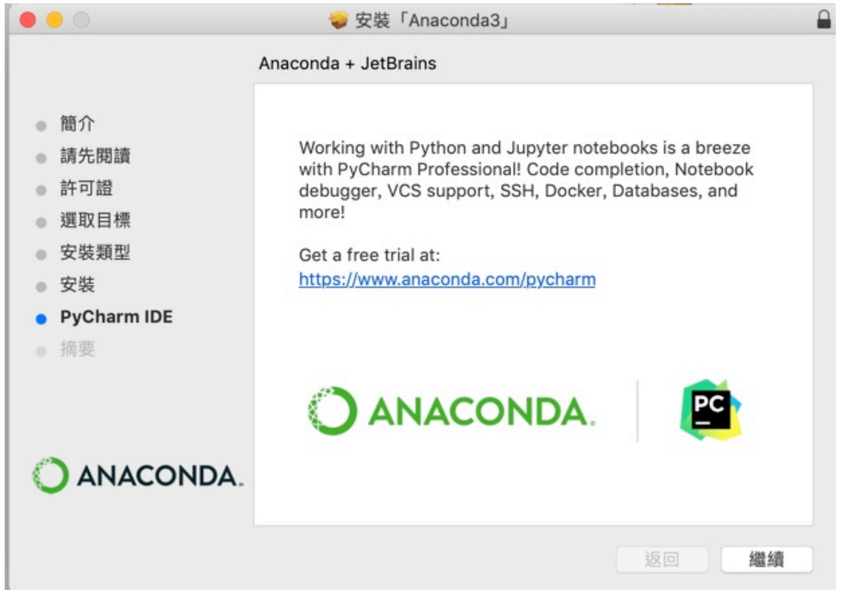
下載Anaconda安裝Installers



安裝Anaconda

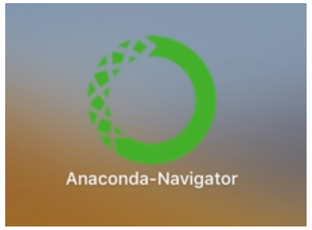

安裝「Anaconda3」
歡迎使用「Anaconda3」安裝程式
簡介
請先閱讀
許可證
選取目標
安裝類型
安裝
PyCharm IDE
摘要
將會引導您完成安裝此軟體的所需步驟。
ANACONDA.
返回
繼續


安裝「Anaconda3」
在「Macintosh HD」上進行標準安裝
簡介
請先閱讀
許可證
選取目標
安裝類型
安裝
PyCharm IDE
摘要
這將會佔用您電腦的 484.8 MB 空間。
請按一下「安裝」來在磁碟「Macintosh HD」上執行此軟體的標準安裝。
更改安裝位置…
ANACONDA.
自定
返回
安裝


安裝「Anaconda3」
Anaconda + JetBrains
簡介
請先閱讀
許可證
選取目標
安裝類型
安裝
PyCharm IDE
摘要
Working with Python and Jupyter notebooks is a breeze with PyCharm Professional! Code completion, Notebook debugger, VCS support, SSH, Docker, Databases, and more!
Get a free trial at:
https://www.anaconda.com/pycharm
ANACONDA.
PC
ANACONDA.
返回
繼續

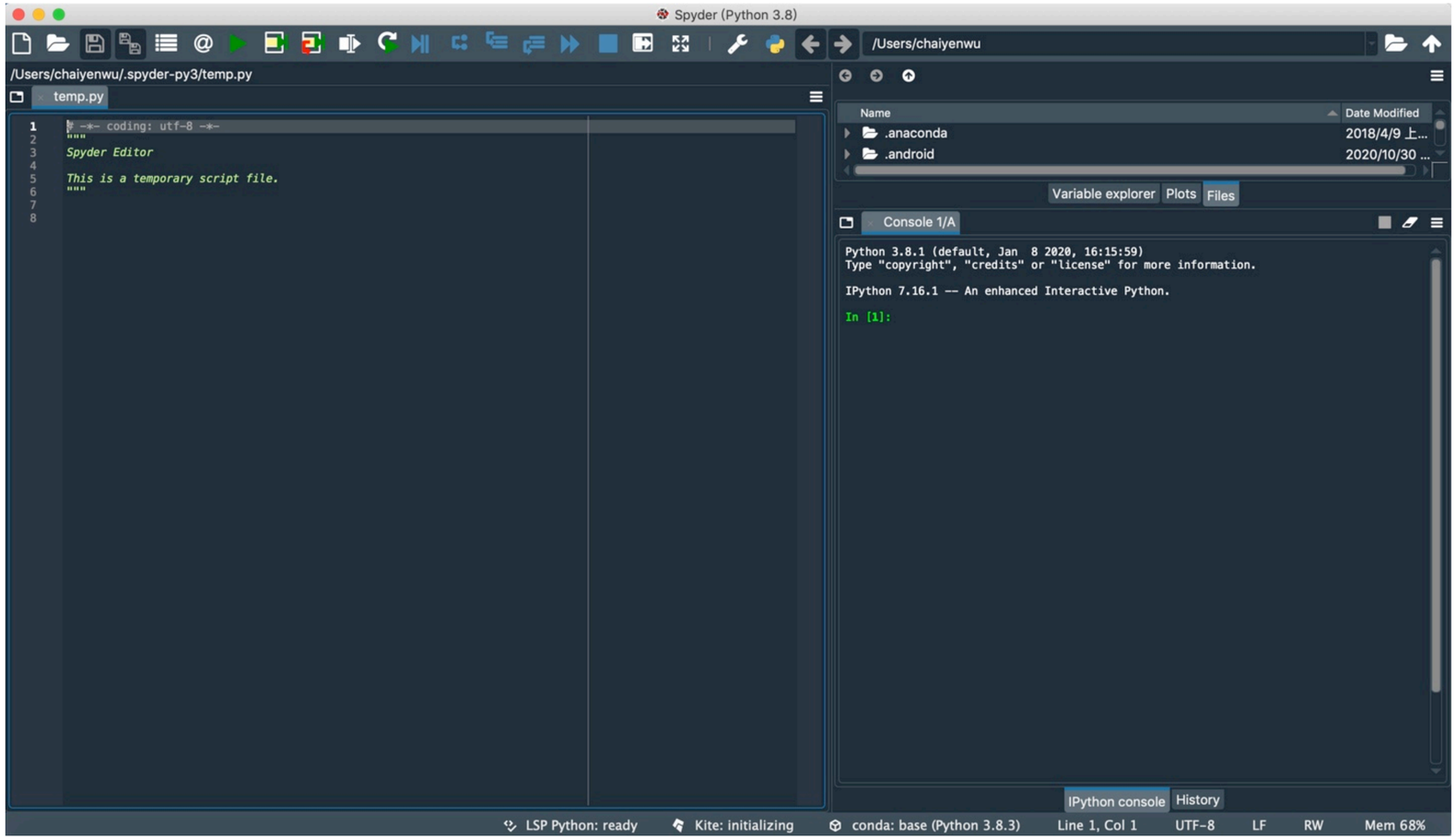
選取Launchpad

- 執行Anaconda



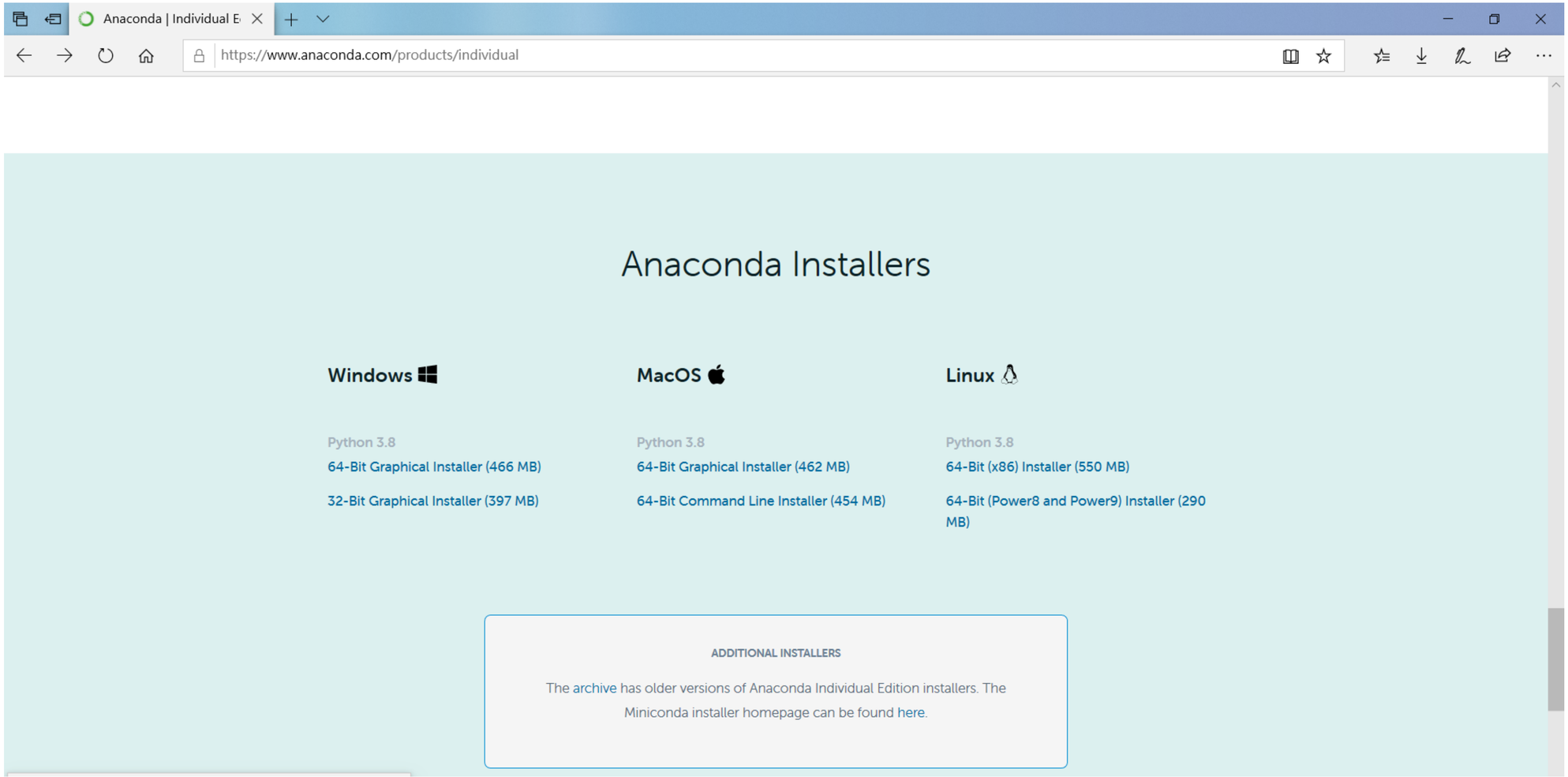
執行Spyder

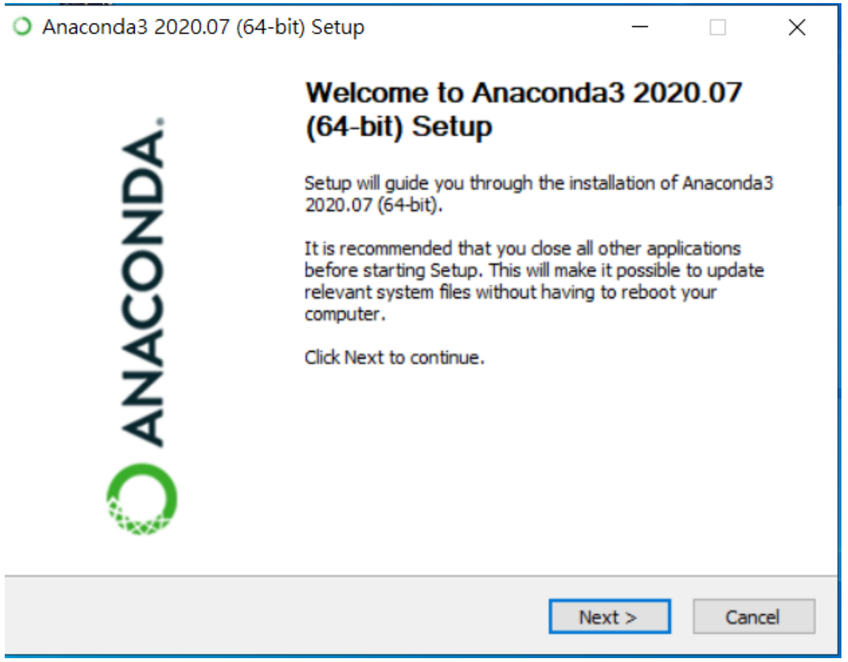

Spyder
4.1.4
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Launch

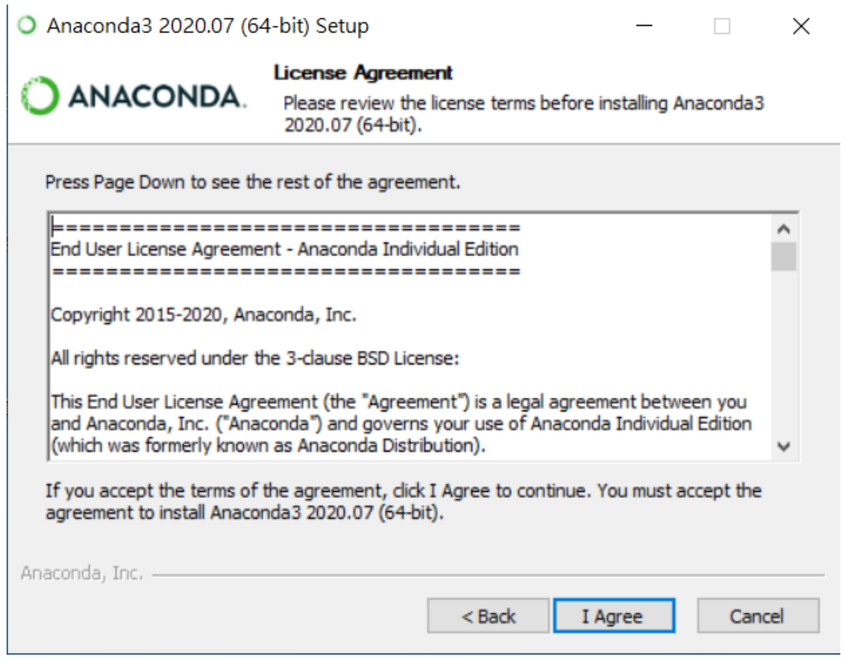

Spyder (Python 3.8)
/Users/chaiyenwu
/Users/chaiyenwu/.spyder-py3/temp.py
temp.py
-*- coding: utf-8 -*-
"""
Spyder Editor
This is a temporary script file.
"""
Name
Date Modified
.anaconda
2018/4/9 上...
.android
2020/10/30 ...
Variable explorer
Plots
Files
Console 1/A
Python 3.8.1 (default, Jan 8 2020, 16:15:59)
Type "copyright", "credits" or "license" for more information.
IPython 7.16.1 -- An enhanced Interactive Python.
In [1]:
IPython console
History
LSP Python: ready
Kite: initializing
conda: base (Python 3.8.3)
Line 1, Col 1
UTF-8
LF
RW
Mem 68%

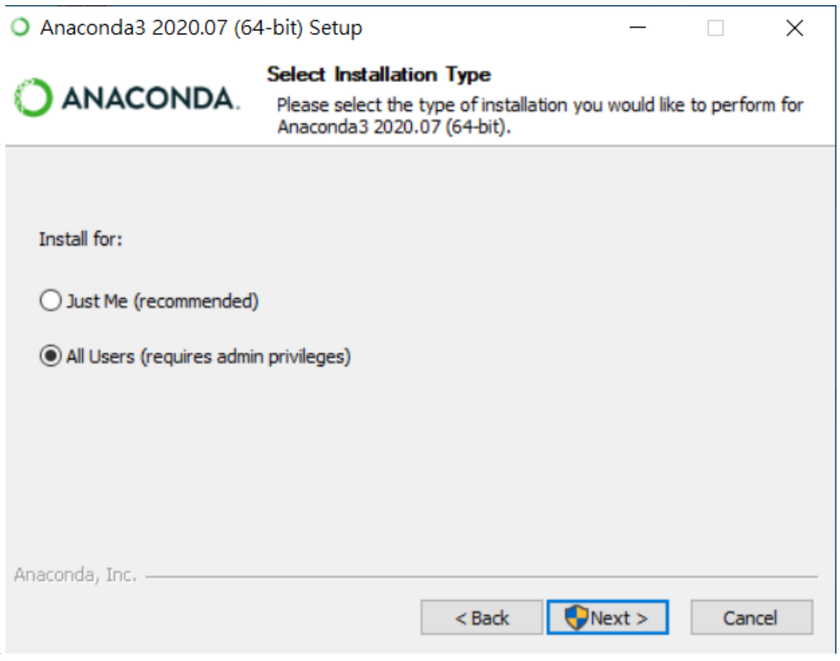
1-4Windows安裝Anaconda

- Windows安裝
- 設定Anaconda安裝路徑
- 將Anaconda3加入系統路徑

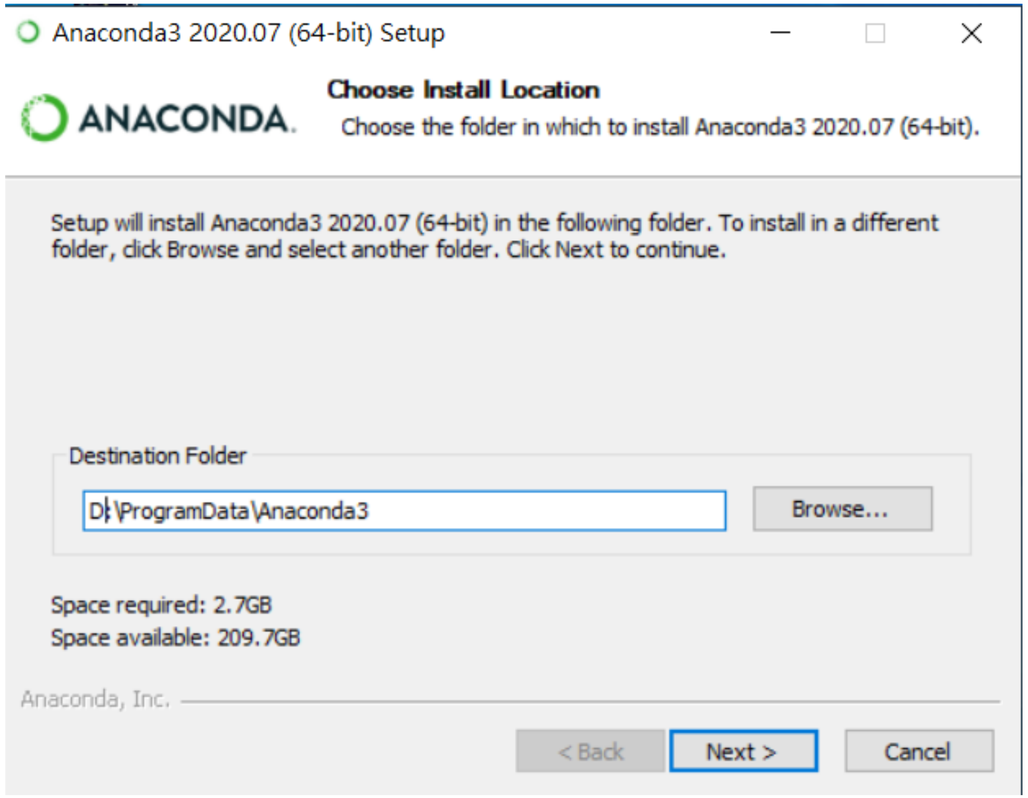

Anaconda | Individual E
https://www.anaconda.com/products/individual
Anaconda Installers
Windows
Python 3.8
64-Bit Graphical Installer (466 MB)
32-Bit Graphical Installer (397 MB)
MacOS
Python 3.8
64-Bit Graphical Installer (462 MB)
64-Bit Command Line Installer (454 MB)
Linux
Python 3.8
64-Bit (x86) Installer (550 MB)
64-Bit (Power8 and Power9) Installer (290 MB)
ADDITIONAL INSTALLERS
The archive has older versions of Anaconda Individual Edition installers. The Miniconda installer homepage can be found here.


Anaconda3 2020.07 (64-bit) Setup
ANACONDA.
Welcome to Anaconda3 2020.07 (64-bit) Setup
Setup will guide you through the installation of Anaconda3 2020.07 (64-bit).
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.
Click Next to continue.
Next >
Cancel

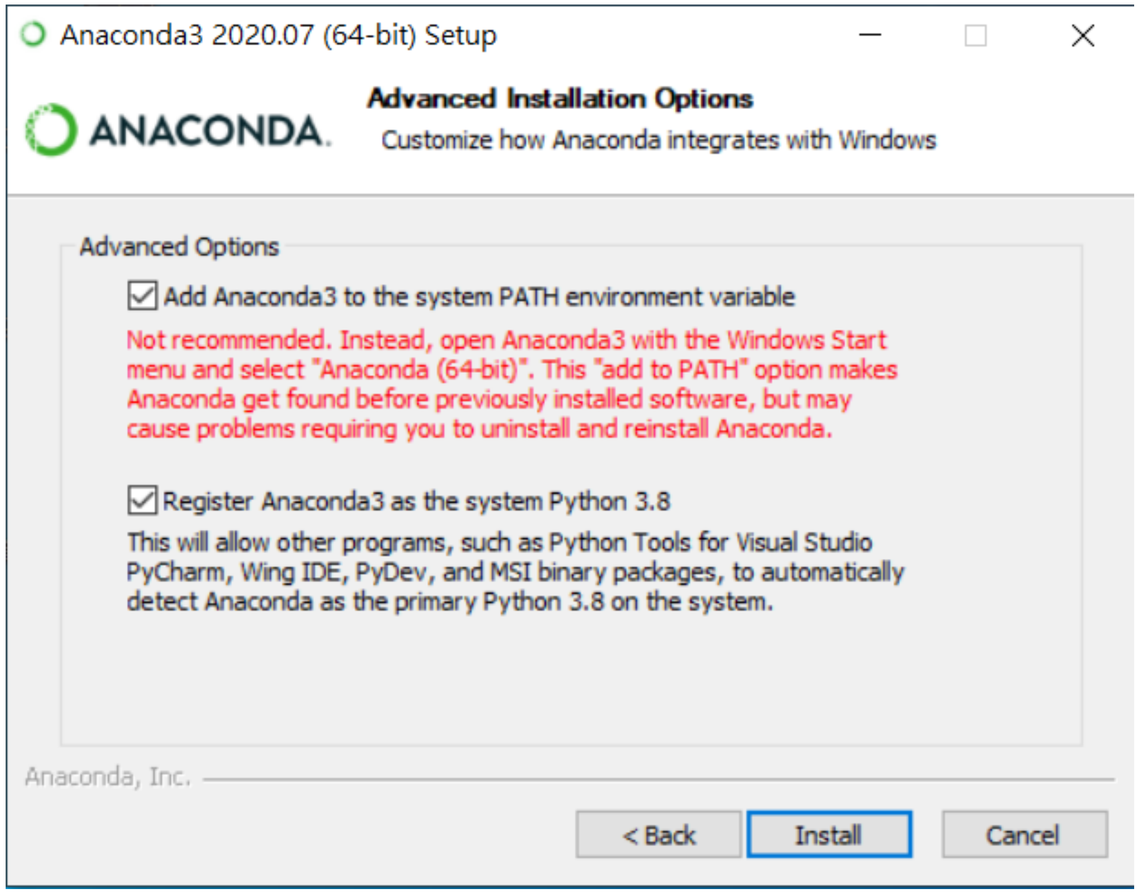

Anaconda3 2020.07 (64-bit) Setup
ANACONDA.
License Agreement
Please review the license terms before installing Anaconda3 2020.07 (64-bit).
Press Page Down to see the rest of the agreement.
End User License Agreement - Anaconda Individual Edition
Copyright 2015-2020, Anaconda, Inc.
All rights reserved under the 3-clause BSD License:
This End User License Agreement (the "Agreement") is a legal agreement between you and Anaconda, Inc. ("Anaconda") and governs your use of Anaconda Individual Edition (which was formerly known as Anaconda Distribution).
If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Anaconda3 2020.07 (64-bit).
Anaconda, Inc.
< Back
I Agree
Cancel

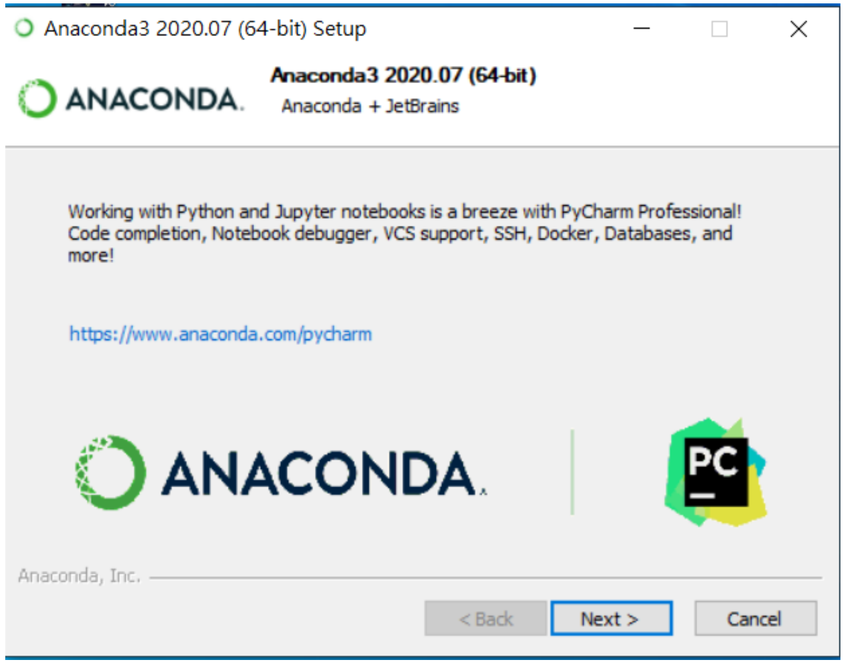

Anaconda3 2020.07 (64-bit) Setup
ANACONDA.
Select Installation Type
Please select the type of installation you would like to perform for Anaconda3 2020.07 (64-bit).
Install for:
Just Me (recommended)
All Users (requires admin privileges)
Anaconda, Inc.
< Back
Next >
Cancel

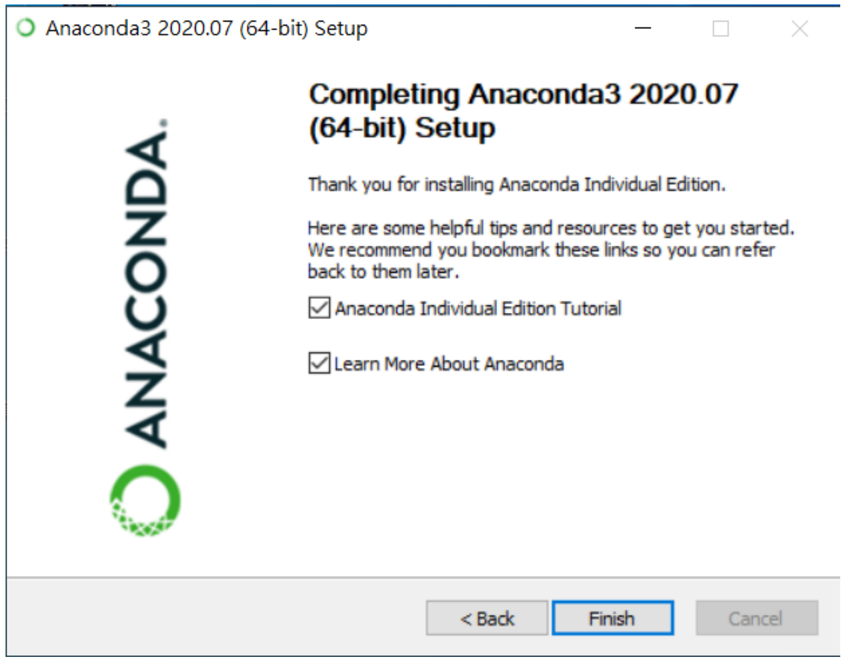
設定Anaconda安裝路徑



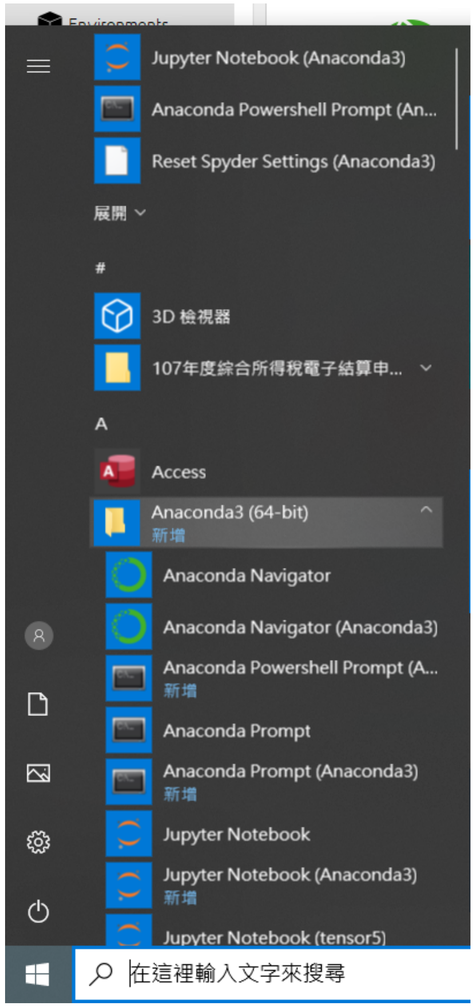
將Anaconda3加入系統路徑



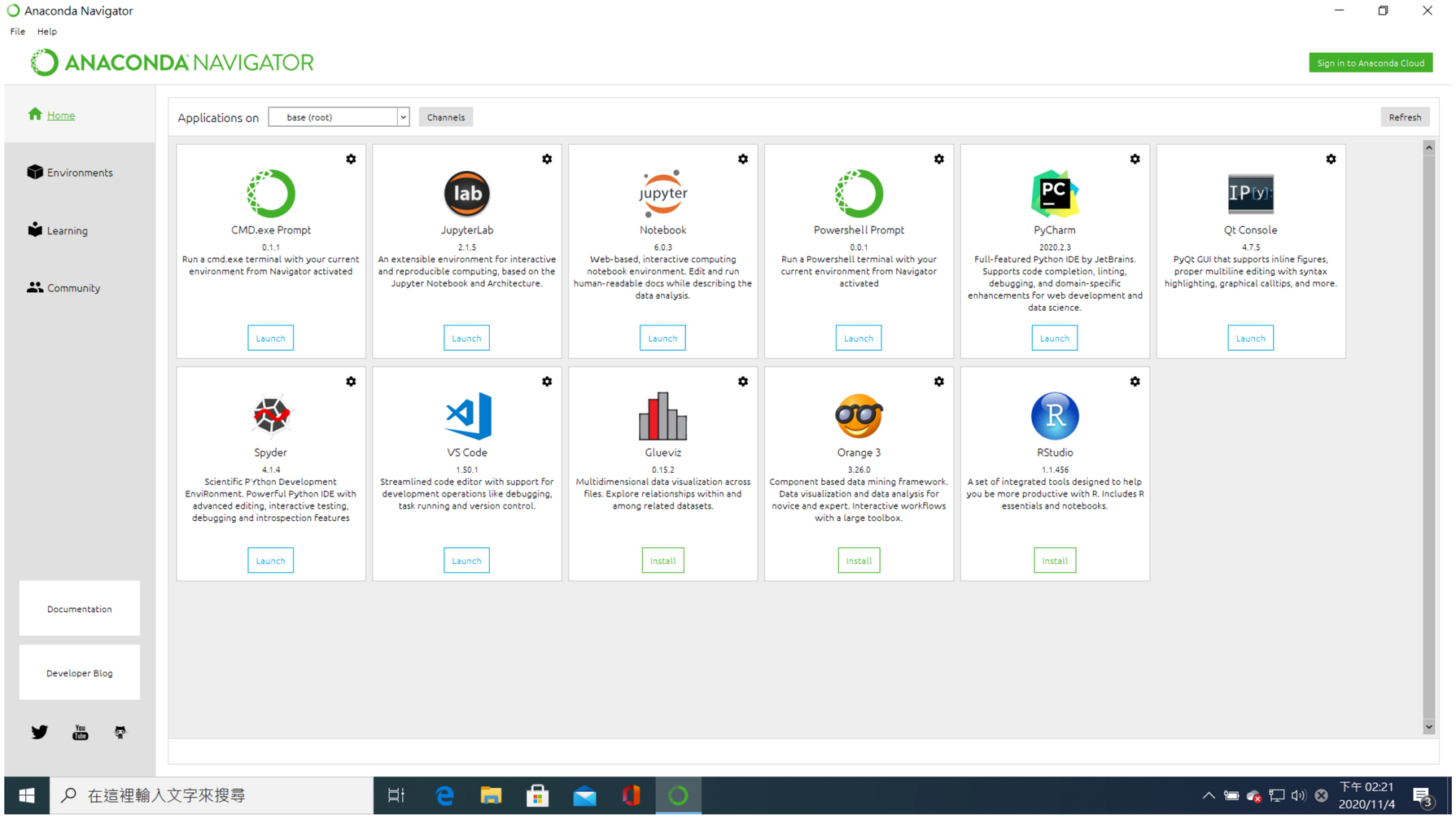

Anaconda3 2020.07 (64-bit) Setup
ANACONDA.
Anaconda3 2020.07 (64-bit)
Anaconda + JetBrains
Working with Python and Jupyter notebooks is a breeze with PyCharm Professional! Code completion, Notebook debugger, VCS support, SSH, Docker, Databases, and more!
https://www.anaconda.com/pycharm
ANACONDA.
PC
Anaconda, Inc.
< Back
Next >
Cancel

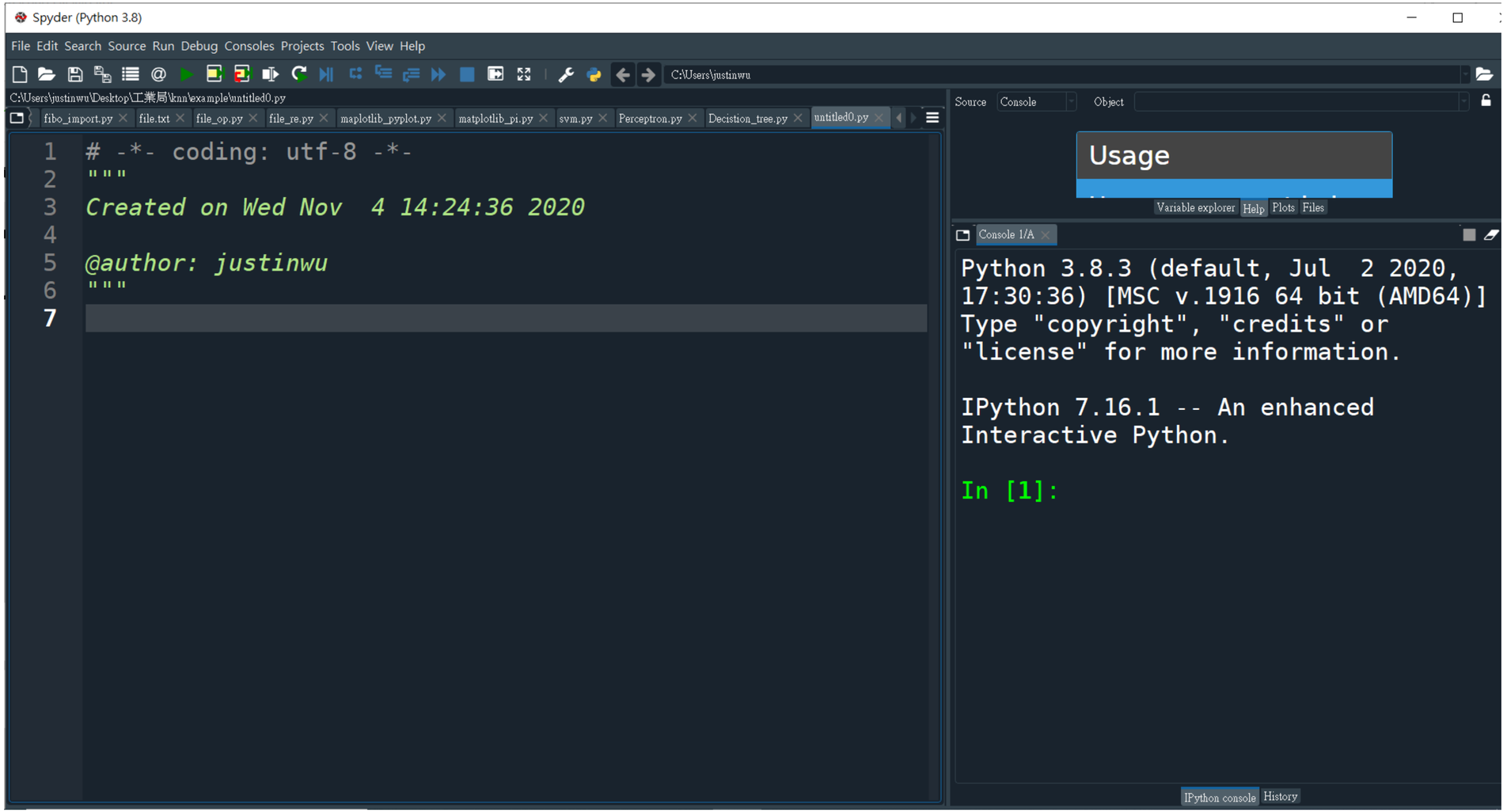

Anaconda3 2020.07 (64-bit) Setup
ANACONDA.
Completing Anaconda3 2020.07 (64-bit) Setup
Thank you for installing Anaconda Individual Edition.
Here are some helpful tips and resources to get you started. We recommend you bookmark these links so you can refer back to them later.
Anaconda Individual Edition Tutorial
Learn More About Anaconda
< Back
Finish
Cancel

在選單啟動Anaconda

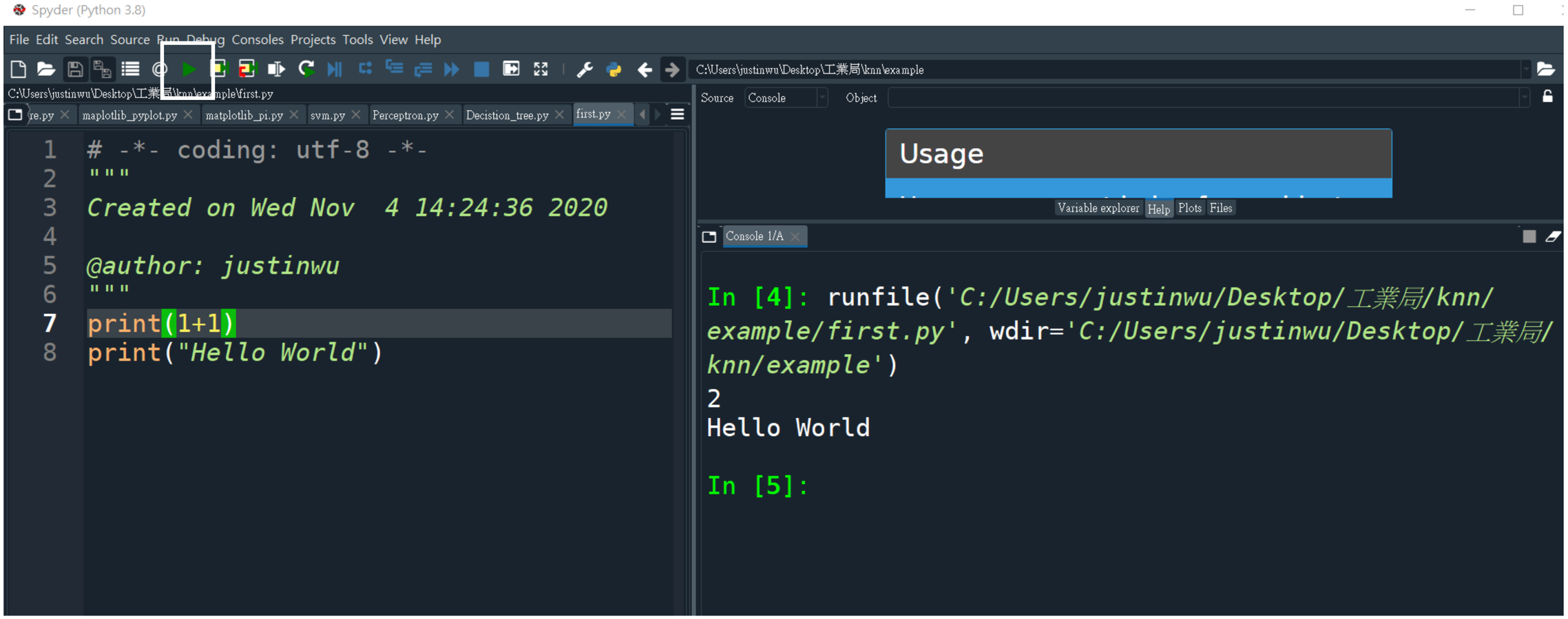


啟動Spyder


Anaconda Navigator
File Help
ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
Home
Environments
Learning
Community
Documentation
Developer Blog
Applications on
base (root)
Channels
Refresh
CMD.exe Prompt
0.1.1
Run a cmd.exe terminal with your current environment from Navigator activated
Launch
JupyterLab
2.1.5
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch
Notebook
6.0.3
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch
Powershell Prompt
0.0.1
Run a Powershell terminal with your current environment from Navigator activated
Launch
PyCharm
2020.2.3
Full-featured Python IDE by JetBrains. Supports code completion, linting, debugging, and domain-specific enhancements for web development and data science.
Launch
Qt Console
4.7.5
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch
Spyder
4.1.4
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Launch
VS Code
1.50.1
Streamlined code editor with support for development operations like debugging, task running and version control.
Launch
Glueviz
0.15.2
Multidimensional data visualization across files. Explore relationships within and among related datasets.
Install
Orange 3
3.26.0
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.
Install
RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.
Install
在這裡輸入文字來搜尋
下午 02:21
2020/11/4


Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\justinwu
C:\Users\justinwu\Desktop\工業局\knn\example\untitled0.py
fibo_import.py
file.txt
file_op.py
file_re.py
maplotlib_pyplot.py
matplotlib_pi.py
svm.py
Perceptron.py
Decistion_tree.py
untitled0.py
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Nov 4 14:24:36 2020
4
5 @author: justinwu
6 """
7
Source Console Object
Usage
Variable explorer Help Plots Files
Console 1/A
Python 3.8.3 (default, Jul 2 2020, 17:30:36) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.
IPython 7.16.1 -- An enhanced Interactive Python.
In [1]:
IPython console History

執行程式


Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\justinwu\Desktop\工業局\knn\example
C:\Users\justinwu\Desktop\工業局\knn\example\first.py
re.py
maplotlib_pyplot.py
matplotlib_pi.py
svm.py
Perceptron.py
Decistion_tree.py
first.py
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Nov 4 14:24:36 2020
4
5 @author: justinwu
6 """
7 print(1+1)
8 print("Hello World")
Source
Console
Object
Usage
Variable explorer
Help
Plots
Files
Console 1/A
In [4]: runfile('C:/Users/justinwu/Desktop/工業局/knn/example/first.py', wdir='C:/Users/justinwu/Desktop/工業局/knn/example')
2
Hello World
In [5]:

2.Python直譯器與計算機

- Mac電腦/usr/local/bin
- Windows電腦C:\python3
- set path=%path%;C:\python3

Python直譯器所在目錄

輸入python執行

```
$ /usr/local/bin/python3.9
```

或

```
$python
Python 3.9.0 (v3.9.0:9cf6752276, Oct  5 2020, 11:29:23)
[Clang 6.0 (clang-600.0.57)] 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

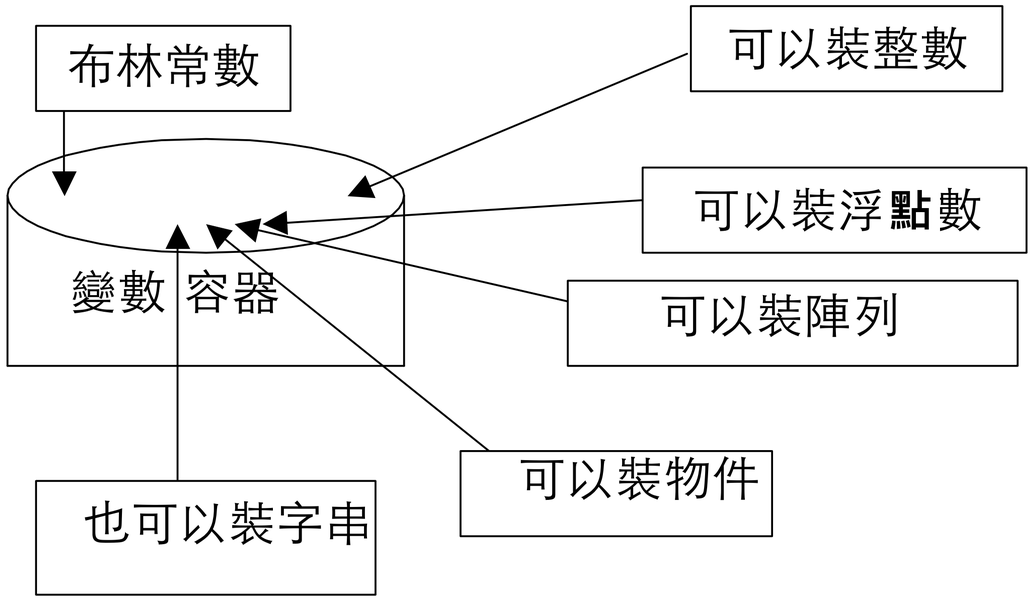
- #是註解符號
- >>> #註解#

python計算機

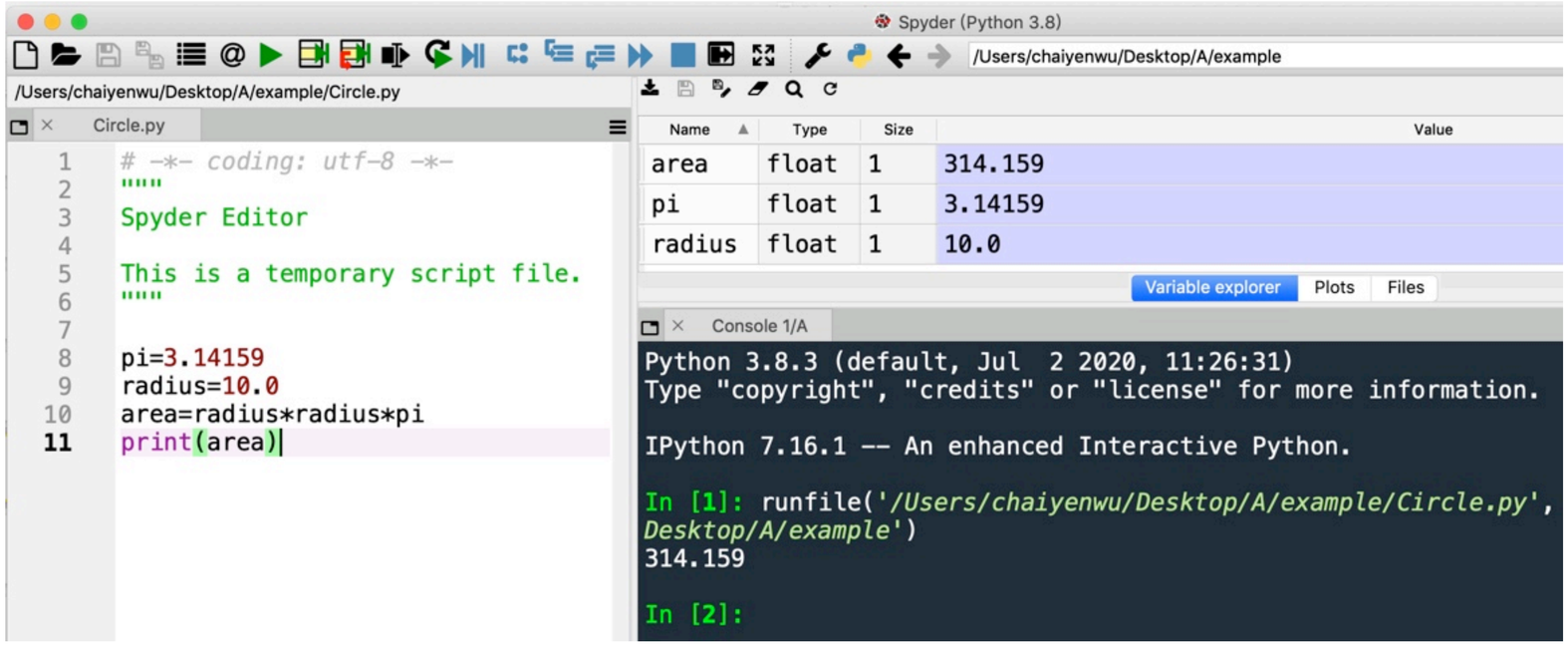
-
```
>>> 1+2
3
>>> 2-1
1
>>> 3*2
6
>>> 3/2
1.5
>>> 3%2
1
>>> 50-5/6
49.166666666666664
>>> (50-5*6)/4
5.0
>>> 5**2
25
>>> 2**5
32
```

2-1簡單的程式

- Python支援的資料型態:
- 浮整數(floating point)
- 字元(char)
- 整數(integer)
- 物件(object)
- 布林常數(Boolean)
- 空值(null)
- 字串(string)。



- 範例：Circle.py
- 第一行宣告了pi的資料型態為雙精度浮點數，並且給予初始值3.14159。
- 第二行宣告了radius半徑的資料型態為雙精度浮點數。
- 第三行宣告了area面積的資料型態為雙精度浮點數。



2-2識別名稱

- 每一個變數都有識別名稱。
- 我們變數宣告時，就是給該變數一個識別名稱。
- 所有的識別名稱是由字元、數字、下底線(_)所組成。
- Python是有大小寫的區隔，a和A是不一樣的變數。

2-3數值資料型態與運算子

- 變數a=1+1,c，c=5，變數c的值為5在這個中
- a=1+1是一個運算式
- c=5是一個運算式
- +和=是運算子，變數a和數值5，和數值1是運算元。
- 運算式就是由運算子和運算元所組成。
-

運算子優先順序,先乘除後加減

Arithmatic.py

```
1  a = 1+1
2  b = 2-1
3  div=6/2
4  pi=3.14159
5  radius=10
6  area=radius*radius*pi
7  c=5
8  x=2*3+4/2+1-3%2+5
9  print(a)
10 print(b)
11 print(div)
12 print(area)
13 print(c)
14 print(x)
```

thmatic

```
/Users/justinwu/PycharmProjects/Pythc
2
1
3.0
314.159
5
13.0
```

2-4運算子結合優先順序

Arithmatic.py | operator_preference.py

```
1  x=2**3+5*2-5/5+5%2
2  y=1<<2
3  print(x)
4  print(y)
5  b=3
6  c=3
7  d=b&c
8  print(d)
9  e=b^c
10 print(e)
11 f=b|c
12 print(f)
13 g=1<5
14 print(g)
15 h=1>5
16 print(h)
```

```
18.0
4
3
0
3
True
False
```

3.資料結構

- 變數
- 運算式與運算子
- 串列
- 堆疊
- 佇列

變數

- 資料型態
  - 整數
  - 浮點數
  - 字串

```python
#!/usr/bin/env python3
#_*_coding:utf-8_*_
"""
Created on Thu Oct 26 07:40:42 2017

@author: justinwu
"""
#這是註解

#這是註解
1+2
#print(1+2)
x=2-1
print(x)
y=3+2
print(y)
z=3.2*2
print(z)
str='大家好'
print(str)
str2='Python'
mychar=str[0]
print(mychar)
```

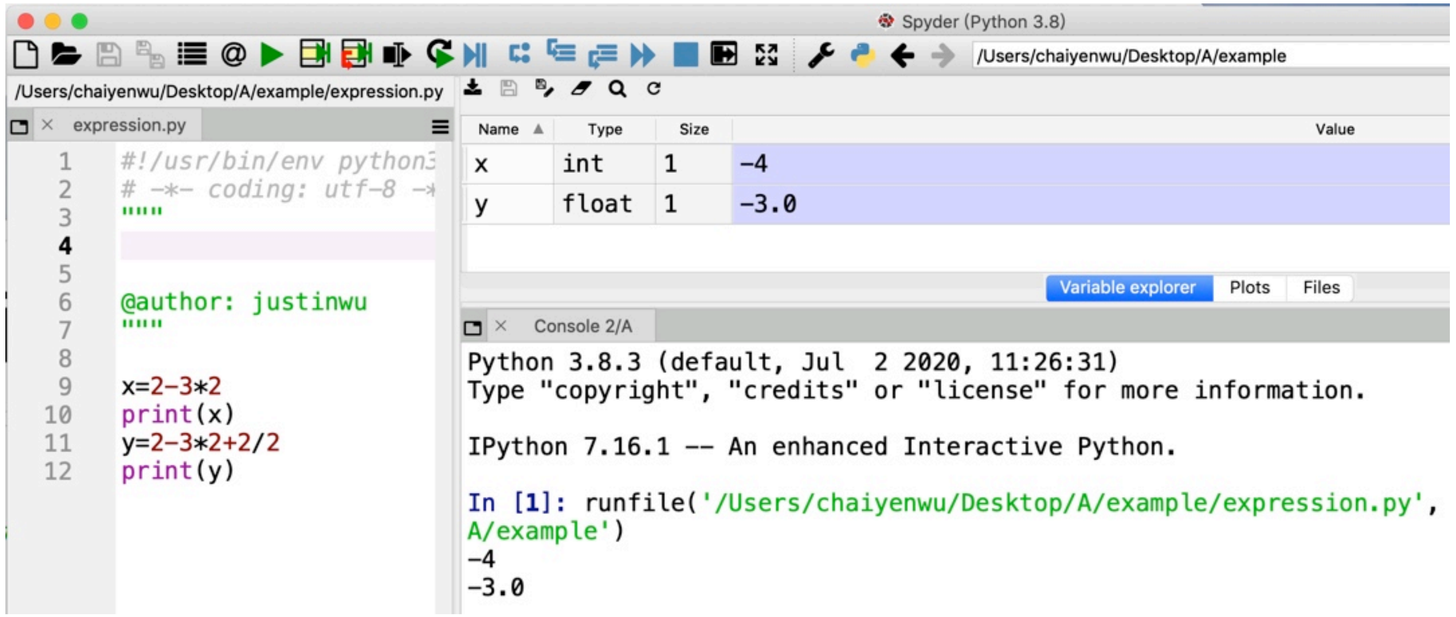
Name	Type	Size	
mychar	str	1	大
str	str	1	大家好
str2	str	1	Python
x	int	1	1
y	int	1	5
z	float	1	6.4

Console 1/A

```
In [28]: runfile('/Users/ju
wdir='/Users/justinwu/Deskt
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


Spyder (Python 3.8)
/Users/chaiyenwu/Desktop/A/example
/Users/chaiyenwu/Desktop/A/example/expression.py
expression.py
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*
3 """
4
5
6 @author: justinwu
7 """
8
9 x=2-3*2
10 print(x)
11 y=2-3*2+2/2
12 print(y)
Name
Type
Size
Value
x
int
1
-4
y
float
1
-3.0
Variable explorer
Plots
Files
Console 2/A
Python 3.8.3 (default, Jul 2 2020, 11:26:31)
Type "copyright", "credits" or "license" for more information.
IPython 7.16.1 -- An enhanced Interactive Python.
In [1]: runfile('/Users/chaiyenwu/Desktop/A/example/expression.py',
A/example')
-4
-3.0

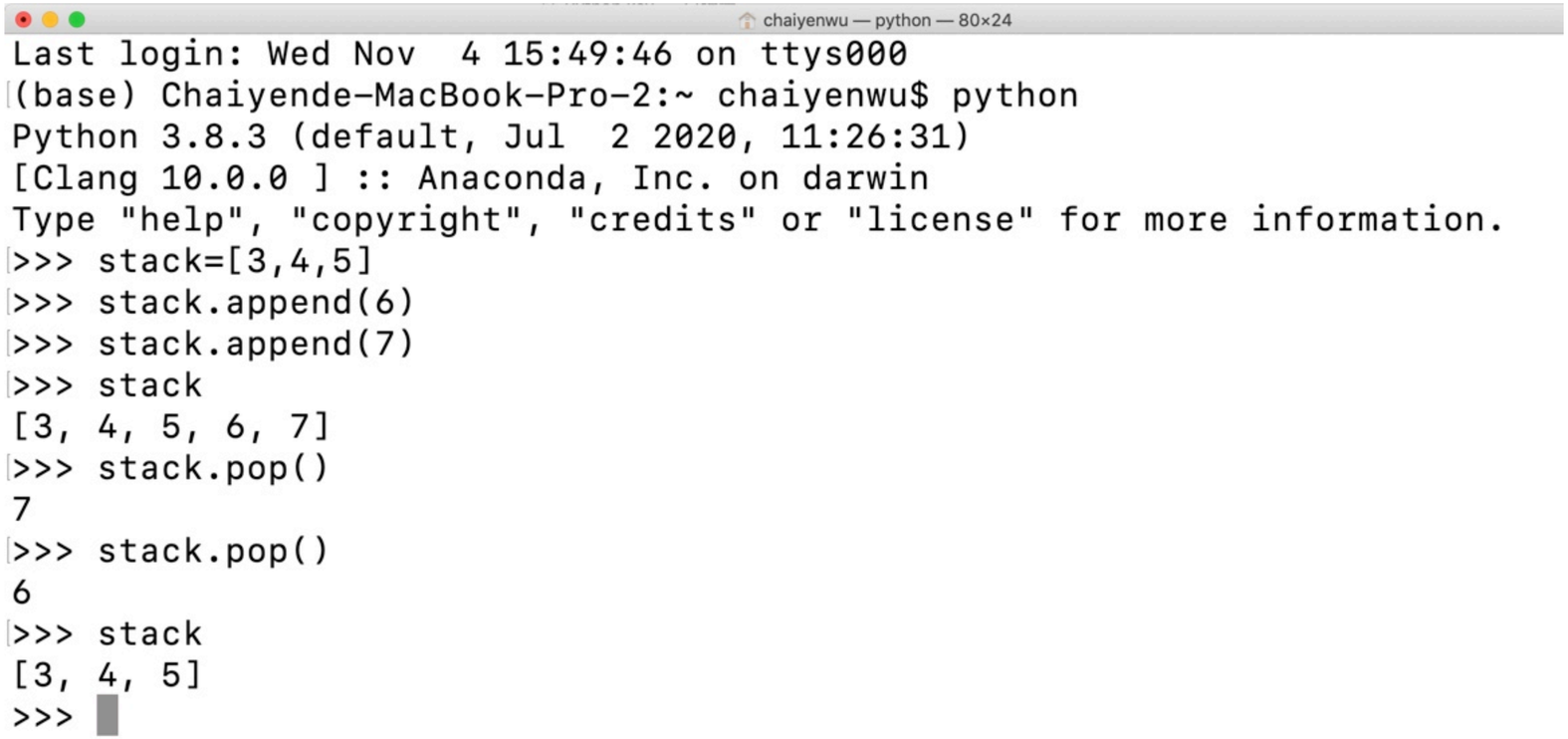
串列

- 中括號[]包起來的為串列。
- 儲存的元素放在 [] 括號中，不同元素之間使用逗號來隔開。
- 串列的操作有push放入元素和pop取出元素。

```
>>> 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'
>>>
```

堆疊使用串列

- 先進後出


chaiyenwu — python — 80×24
Last login: Wed Nov 4 15:49:46 on ttys000
(base) Chaiyende-MacBook-Pro-2:~ chaiyenwu$ python
Python 3.8.3 (default, Jul 2 2020, 11:26:31)
[Clang 10.0.0] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more 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]
>>>

佇列使用串列

- 先進先出
-

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

```python

tp1=()
print(tp1)
tp2=(1,2,3,4,5,6,7,8)
print(tp2)
print(sum(tp2))
print('--------------------')
print(tp2[2:5])#切割運算子
print(tp2[-1])
tp3 =tuple([2*x for x in range(1,8)])
print(tp3)
print('---------------')
tp4=tuple('Ivy Lin')
print(tp4)
tp5=("John",'小寶','小文')
print(tp5)
print(len(tp5))
print(tp4+tp5)
print('-------------')
tp6=tuple([1,2,3,4,5,6,7,8,9])
print(tp6)
print(max(tp6))
print(min(tp6))

```

```
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/Deskto
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}

```
 8 tel={'Justin':'0920909872','Ivy':'0922876895'}
 9 tel['Johny']='0920885356'
10 print(tel)
11 print(tel['Johny'])
12 del tel['Johny']
13 tel['Mary']='0922865255'
14 print(tel)
15 print(list(tel.keys()))
16 print(sorted(tel.keys()))
17 print('Ivy' in tel)
18 print('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

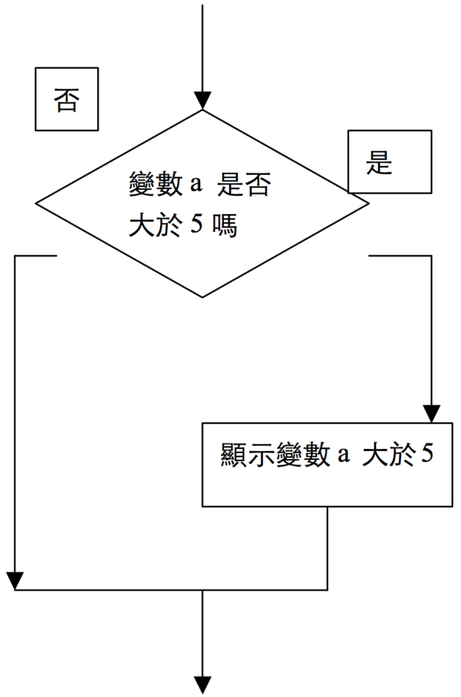
- 選取結構if
- 迴圈結構while，for

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={true,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。



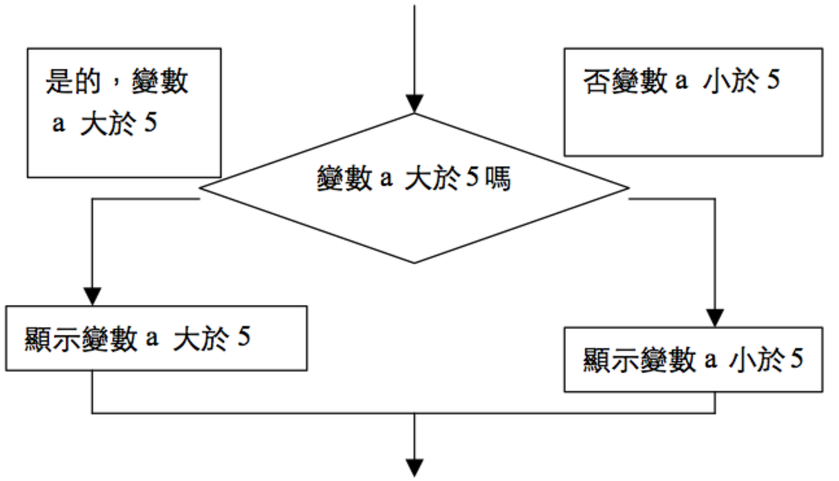
- 當下列菱形四邊形成立true時會執行右方的流程，當下列菱形四邊行的條件不成立false時會執行下方或右方的流程。True和false就是屬於布林代數，這是運用在迴圈結構。

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

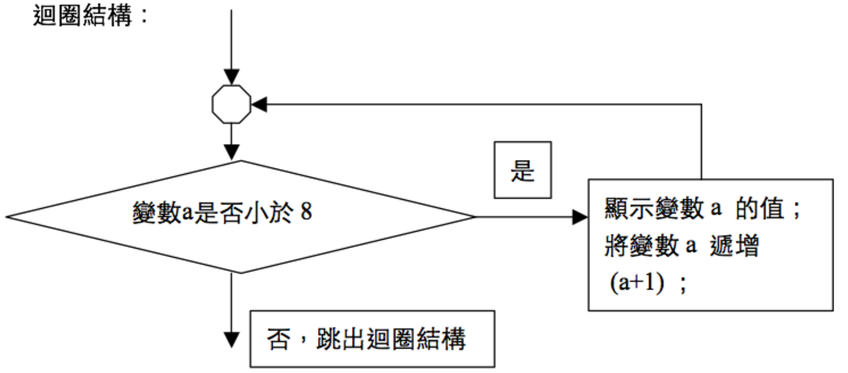
循序結構，就是程式一行一行的由上而下循序執行。

選取結構：

選取結構：



迴圈結構：



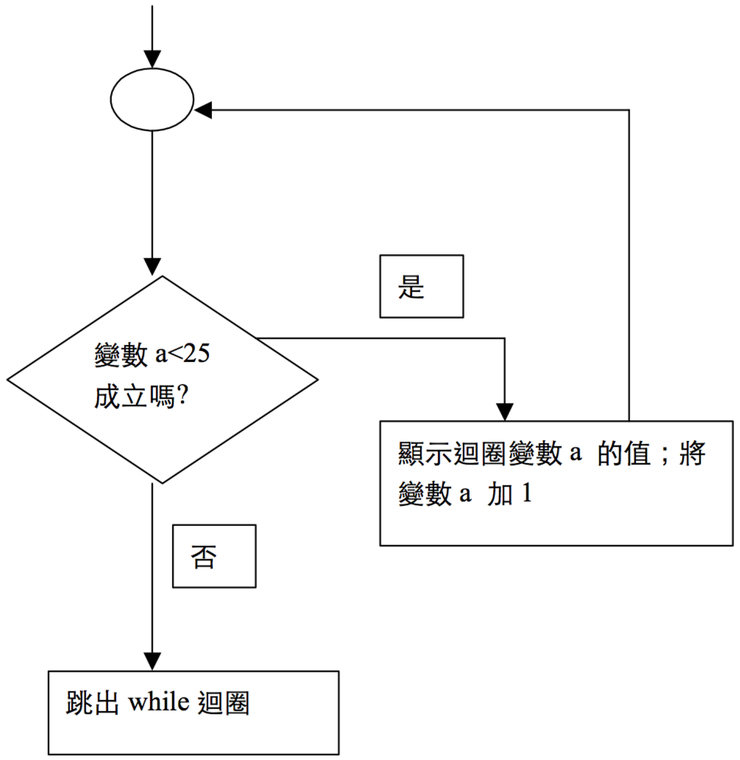
選取結構if

```
 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")
```

```
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

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

```
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
```

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

In [2]:
```

布林運算式

```
 8
 9 x=True
10 y=False
11 print(x&y)
12 print(x|y)
13 if (x&y):
14     print(x&y)
15 else:
16     print(x&y)
17 z=True
18 print(z&y)
19 print(z|y)
20
```

```
In [30]: runfil
Users/justinwu/
False
True
False
False
True

In [31]:
```

continue繼續執行迴圈

```
 8
 9 for 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

```
 8
 9 i=5
10 x =True
11 while 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]:
```

4-1邏輯運算子

- 邏輯運算子可以結合條件，以一個表達式判斷許多條件，而這些條件的結果不是真True就是假False。
- and稱為”與邏輯運算子”，只有當所有條件都成立時才會回傳真True，否則回傳假False。
- or稱為或邏輯運算子，只要運算式中一個條件成立就會回傳真True，只有當所有的條件都為假Fasle時，才會回傳假False。
- not為相反邏輯運算子，真True的條件加上not相反邏輯運算子時，就會變成假False；當假False的條件加上not相反邏輯運算子時，就會變成真True。

邏輯運算子

```
1   x=True
2   y=False
3   z=x and y
4   print(z)
5   z1=x or y
6   print(z1)
7   z2=not x
8   z3=not y
9   print(z2)
10  print(z3)
```

```
/Users/justinwu/PycharmProjects/Pyth
False
True
False
True

Process finished with exit code 0
```

4-2一個選擇的if敘述

- if 條件:

  敘述

一個選擇的if敘述

```
1   x=8
2   if x>5 :
3       print("x is bigger than 5")
```

trol_if

```
/Users/justinwu/PycharmProjects/Pytho
x is bigger than 5

Process finished with exit code 0
```

python

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)
```

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

In [2]:
```

遞迴函數

- 自己呼叫自己
- 有最終條件，並且開始回傳

遞迴函數

```
 8
 9 def 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
17 print(factorial(1))
18 print(factorial(2))
19 print(factorial(3))
```

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

In [3]:
```

費式係數

```
 8
 9 def 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
17 print(fibonaci(1))
18 print(fibonaci(2))
19 print(fibonaci(3))
20 print(fibonaci(10))
21 print(fibonaci(20))
22
```

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

6.類別

- 成員屬性與成員方法
- 類別和實體變數
- __init__(self)建構物件
- __del__(self)解構物件

python

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
- self為自己這個物件

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


>>> 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

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

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

In [5]:
```

7.繼承

- 子類別繼承自父類別，程式碼重複使用
- __為私有存取控制修飾,只有該類別方法才能存取
- 多重繼承
- 多型
- 因為有繼承所以有多型，同名異式

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

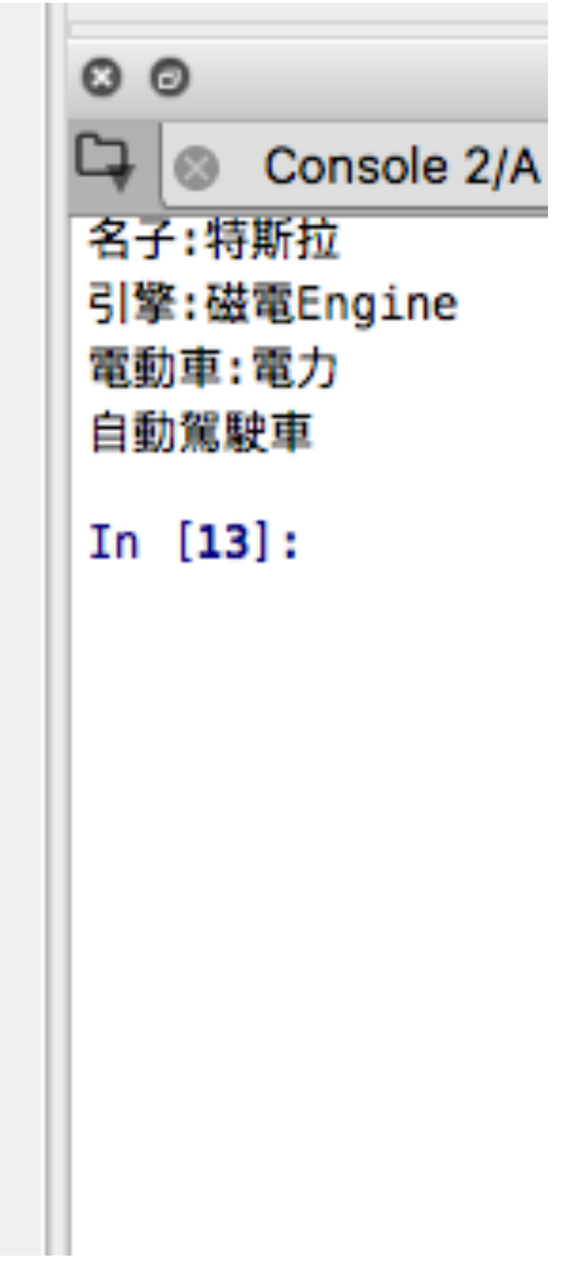
```
 7
 8 class Vehicle:
 9     def __init__(self,name,engine):
10         self.__name = name
11         self.__engine = engine
12
13     def getName(self):
14         return self.__name
15
16     def getEngine(self):
17         return self.__engine
18
19     def setEngine(self,engine):
20         self.__engine = 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())
```

```
名子特斯拉
引擎磁電Engine
電動車電力
自動駕駛車
None
```

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
```



```
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

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

class Vehicle:
    def __init__(self,name,engine):
        self.__name=name
        self.__engine=engine

    def getName(self):
        return self.__name
    def getEngine(self):
        return self.__engine

class Car(Vehicle):
    def __init__(self,name,engine,electric):
        super().__init__(name,engine)
        self.__electric=electric

    def getEngine(self):
        return ("超級")

    def getAuto(self):
        print("自動駕駛車")

myCar=Car("特斯拉","磁電Engine","電力")
myCar.getAuto()
print(myCar.getEngine())
```

```
In [17]: r
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

```
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
```

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

```
Console 2/A
Unexpected error: <class 'SyntaxError'>

In [14]: runfile('/Users/justinwu/Deskto

請輸入薪水:80000
薪水=80000

In [15]: runfile('/Users/justinwu/Deskto

請輸入薪水:x
Unexpected error: <class 'NameError'>

In [16]: runfile('/Users/justinwu/Deskto

請輸入薪水:-10000
錯誤:

In [17]: runfile('/Users/justinwu/Deskto

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+	從檔案尾巴開使讀寫	建立讀寫檔案

開啟,關閉及寫入檔案

```
 8 fp=open('file.txt','w')
 9 if fp !=None:
10     print('檔案開啟成功')
11 fp.close()
12
13 fp=open('file.txt','w')
14 if fp !=None:
15     fp.write("小白")
16 fp.close()
17
18 fp=open('file.txt','w')
19 if fp !=None:
20     fp.write("宇哲")
21 fp.close()
```

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

In [9]:
```

讀取檔案

```
 8
 9 fp=open('file.txt','r')
10 if fp !=None:
11     str=fp.read()
12     print(str)
13 fp.close()
14
15 fp=open('file.txt','r')
16 if fp !=None:
17     strList=fp.readlines()
18     print(strList)
19 fp.close()
```

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

In [12]:
```

8-1讀寫資料

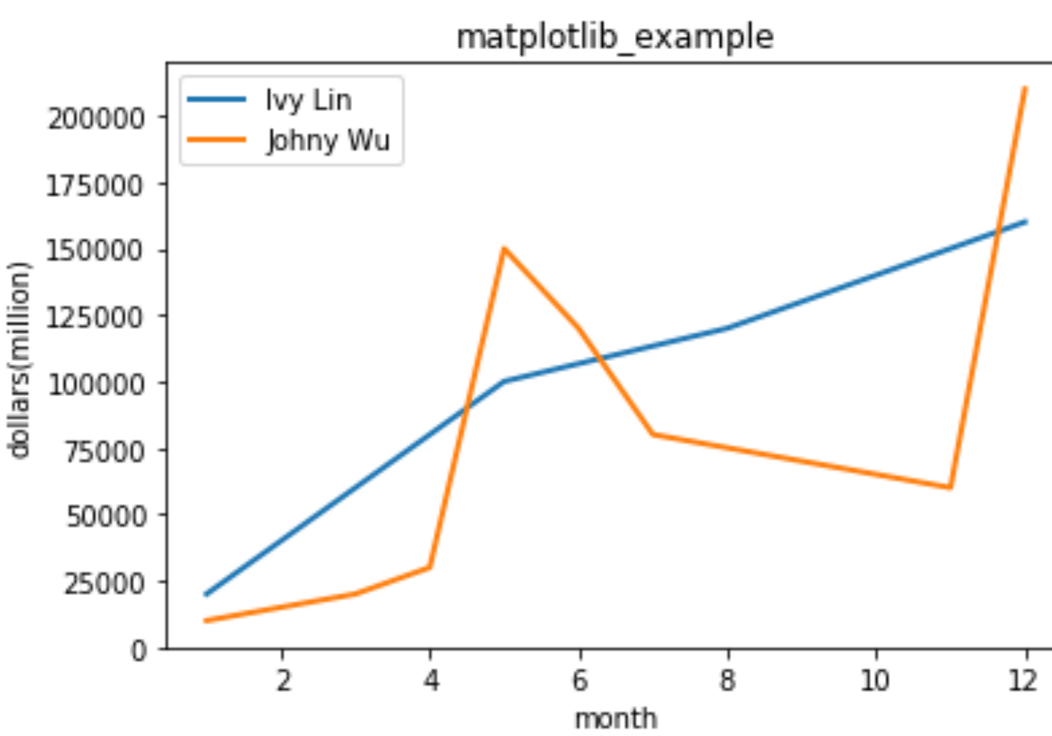
- 檔案輸入輸出
- mode第一個字母代表操作
- mode第二個字母是檔案的類型
- t或無代表文字，b代表二進位

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

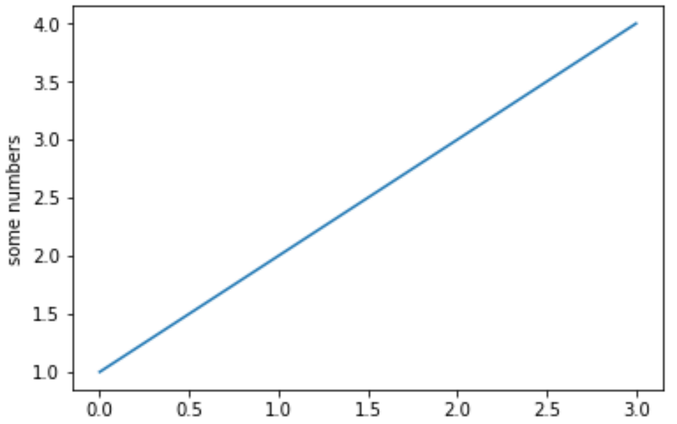
```python
@author: justinwu
"""
import matplotlib.pyplot as mpt

month1 = [1,2,3,4,5,8,10,12]
month2 = [1,3,4,5,6,7,11,12]
sales1 = [20000,40000,60000,80000,100000,120000,140000,160000]
sales2 = [10000,20000,30000,150000,120000,80000,60000,210000]

mpt.plot(month1,sales1,lw=2,label='Ivy Lin')
mpt.plot(month2,sales2,lw=2,label='Johny Wu')
mpt.xlabel('month')
mpt.ylabel('dollars(million)')
mpt.legend()
mpt.title('matplotlib_example')
mpt.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].

```
 8
 9 import matplotlib.pyplot as plt
10 plt.plot([1, 2, 3, 4])
11 plt.ylabel('some numbers')
12 plt.show()
13
14
```

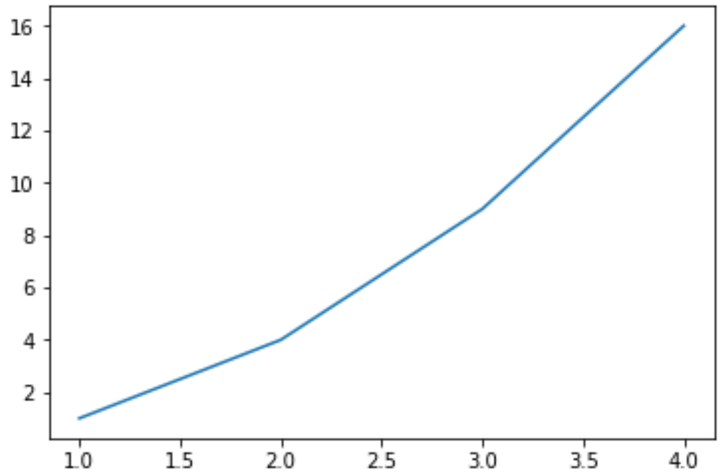
```
In [19]: runfile('/Users/
maplotlib_pyplot.py', wd:
```



第一個[1,2,3,4]參數是X軸,第二個參數是Y軸

```
 8
 9 import matplotlib.pyplot as plt
10
11
12 plt.plot([1, 2, 3, 4], [1, 4, 9, 16])
13 plt.show()
14
```

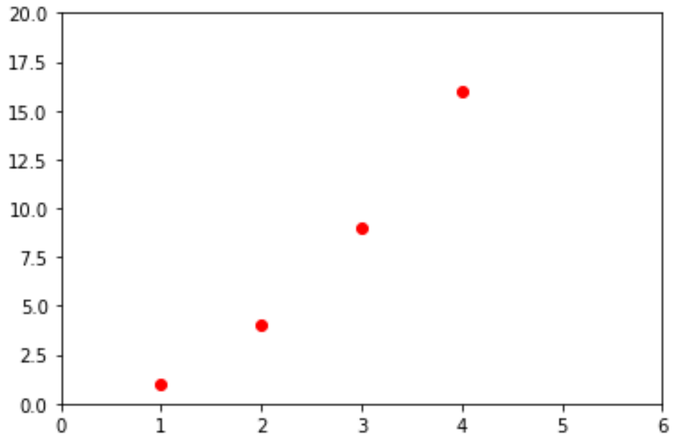
```
In [24]: runfile('/Use
wdir='/Users/justinwu/
```



plot()第三個參數是格式字串點
plot,'ro'為顯示紅色圓圈

```
 8 import 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
10 plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro')
11 plt.axis([0, 6, 0, 20])
12 plt.show()
```

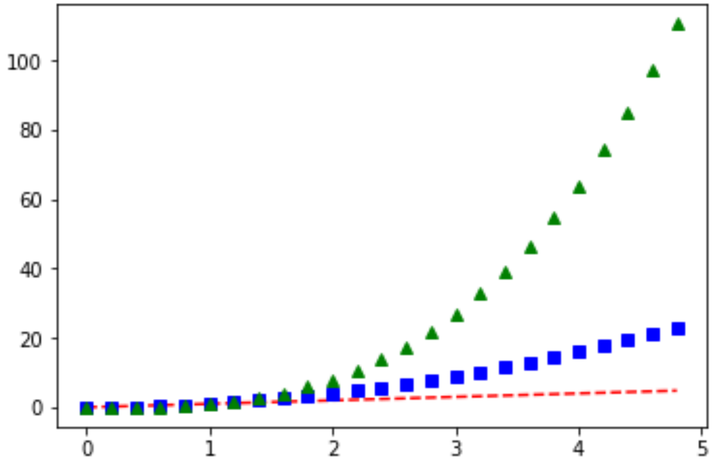
```
In [30]: runfile('/Users/j
plot_ro.py', wdir='/Users/
```



'r--'紅色虛線,'bs'藍色矩形,'g^'綠色三角形

```
 8
 9 import numpy as np
10
11 # 0到5每步0.2
12 t = 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
16 plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3,
17 plt.show()
```

```
In [33]: runfile('/Users/ju
plot_s.py', wdir='/Users/ju
```



- Thanks.