Udemy\_Python筆記

### Chapter1(基本概念, GitHub以及開發專案/寫程式觀念)

1. declare宣告就是創作，只有第一次遇到才會創作，若有遇到過，就只是改變內容而已。x和y在此就是variable變數意思，5,1000,10,2就是value值的意思。一個等號即為設值assignment，即把右邊存到左邊。
2. Data Structure有七種: int, float, bool(只有True, False), str(字串), list, dictionary, tuple.
3. Print()一次也可印很多個，記得用逗點分開就好，且這樣會印在同一行，否則他會自動換行。
4. input(“讓使用者輸入”)，input不會讓程式結束，需等使用者輸入。且input return是一個str。
5. Coding Style:   
   x = 5 (好)

x=5 (不好)

print('hi', x) (好)

print('hi',x) (不好)

變數的命名:

命名建議要貼切和直覺，不要只有x,y,z這樣。命名也幾乎都小寫開頭。

另外，若命名有好幾個單字的，可用，例如:my\_name，而不是myname。

命名開頭不要也不要數字開頭。

python的預留字不能當變數名稱，例如:if,lambda,try等等很多......。

1. 註解放在上面的話，用來解釋後面那一段的code，註解放在一行code的右邊的話，就是解釋那一行而已。，註解可以用三個引號的多行註解，或用#的單行註解。
2. 上傳到GitHub流程 :  
   A. 到GitHub新增專案(按new repository)

B. 桌面建新資料夾

C. cmd中cd進去新資料夾

D. 在cmd複製GitHub的那6行指令

E. 第一行 : 產生README檔，echo就是print，看你想print啥進去README。

F. 第二行 : 把資料夾初始化成git的資料夾(以後才可以在這資料夾做git指令)

G. 第三行 : 把README檔加入追蹤清單

H. 第四行 : 建立版本

I. 第五行 : 把資料夾跟雲端上的GitHub專案做連結

J. 第六行 : 上傳版本到雲端的GitHub專案

(此時就在GitHub上建立專案和README了，README就是說明書而已，接下來要看你想放啥py檔上去)

(六行指令中，最重要的是:

git init (開始可用git指令)

git add 檔名 (上傳想要的檔案)

git commit -m "版本訊息" (設定版本)

git push -u origin master (真正上傳GitHub的動作)(有沒有-u都沒差))

K. 把想放的py檔放到新資料夾

L. 上傳py檔案要執行以下 :

(git指令三部曲)

git add 檔名

git commit -m "版本訊息"

git push origin master

補充:  
git add .

git commit -m ""

git push origin master

以上不行就用以下:

git branch -M main

git push -u origin main

1. 良好的專案開發模式，為:

A. 先建立GitHub專案(一樣要複製那六行code)

B. 開始在subline寫py檔

C. 用Git指令三部曲來新增py檔

D. 當需要更新檔案的時候，一樣用Git指令三部曲去更新，檔名不用變，版本部分改變就好了，來提示自己做了啥更改。

1. 如何寫自己的程式:

【怎麼寫自己的程式】其實你們已經有所概念了，學到這邊，相信你們對寫程式已經有一定的了解。基礎會了，第三方套件的強大也見識到了。其實【怎麼寫自己的程式】沒有什麼祕密SOP，

我只能盡量給大家大格局的流程模擬：  
A. 想清楚自己的程式要有什麼功能

先有願景 ，input output是什麼，先稍微思考開發的階段性，程式的發展性，想做到什麼程度，例如思考要不要先做基本款，以後再增加網頁介面，這時候在思考的是一種【步驟】的概念

接下來可能思考的問題：

1. 程式需要互動嗎？
2. 這個程式需要互動操作嗎？（例如透過CMD輸入資料）
3. 還是只要執行就好了 （例如執行後就自動開始下載某些資料）
4. 程式需要哪些功能？
5. 例如需要儲存資料嗎 (阿我可能就要想我要用哪種格式儲存，文字檔嗎還是需要使用到資料庫(基礎課程沒有教資料庫，但同學可以上網自學也可，先從什麼是資料庫查起)
6. 把問題拆解 (你想達到的功能就是待解決的問題):例如你想寫程式來定期自動貼文到臉書，聽到這樣的問題，我會想到兩個部分：
7. 如何貼文到臉書
8. 如何定期
9. 先把程式的想法拋到GOOGLE看看有什麼東西，說不定已經有現成的程式或是教學，完全跟你想做的功能一樣，先google 'python facebook 貼文' 就會找到不少教學，再來就要解決如何定期貼文，google 'python 定期' 又會查到不少教學。那你這兩個都達的到，不是組合在一起就完成了嗎！
10. 尋找相關套件: google你的想法時，出現很多現成的套件，例如跟臉書互動的套件，這時你就該去看看這個套件的＂官網＂，先稍微看一下介紹，安裝，跟使用範例等等。看起來很順眼，就準備好使用它了！
11. 初始能查的東西查得差不多了，可以開始動工了！動工的第一步就是建立GitHub專案，隨著你的開發，做好辦本控管。如果使用到一些套件，也許是邊看套件的教學，邊做。如果覺得程式行數開始越來越多了，可以看refactor（重構程式）那些影片看看怎麼把程式碼用function收納好。這部分比較屬於嘗試，修改，嘗試，修改。沒有任何公認的SOP表示，寫程式一定要直接先寫function，只要你能把你的程式做到你想要的功能，就已經任務達成了，當你發現一件事情要重複做一百次，你自然而然會發現太麻煩，所以必須寫成function或是用到迴圈，所以沒有任何＂標準步驟＂，都是很自然的a不行就試b, b不行就換c。一般現實生活中，每個寫程式的人寫法習慣等等其實都相差許多，有的人喜歡用這個軟體，有的人喜歡那個，所以越是了解這個領域，越是會發現很多很多的東西都在激辯當中，各種事情都各有優缺，很少很少事情已經有 公布說＂這就是最好的做法＂。所以大家勇於去嘗試就對了！

### Chapter2(邏輯比較, flow control和型別轉換)

x = 5

print(x == 5) #是否等於

print(x != 5) #是否不等於

print(x > 2) #是否大於

print(x < 2) #是否小於

print(x >= 2) #是否大於等於

print(x <= 2) #是否小於等於

1. if 架構，if後面一定要是個會算成布林值:True/False的問題，然後一定要記得有冒號(可以想像冒號是如果有....的話)。
2. else(if架構延伸)，延伸指的是他們都是同一個block，也就是if和else只會執行一個。
3. elif(if架構延伸)，elif和else都只是if的延伸，整個是1個if的架構，也就是說他們都必須要接在if底下才行。還有else一定是放在if架構的最後面，他捕捉所有剩下的可能性。if後面可以接無限個elif和只有一個else。
4. 什麼是區塊(block)?  
   python是以block來區分程式碼的執行，類似C/C++的{}，開頭的判斷式判斷要不要進去block裡面。只要是空格數相同的程式碼就叫做同一個block，空格的話都是4個空格或1個tab為單位。block用意是要知道code是在裡面還是外面，執行結果可能會因此而差很多，像是會有明明if條件沒過不會執行if內的block的print，但因你print寫在if的block外面，他就會去執行。另外，你可以用tab或四個空白鍵去空格，但不能混著用，而subline的幫你自動換行是用tab。你可以ctrl+a全選去看到底是空格鍵還是用tab，一條線就是tab，有點點的就是空白鍵。另外，google公布的寫法指南是用空白鍵去空格。
5. 型別轉換(Casting): 比較判斷符號不能把字串和整數做比較，因此需要型別轉換。重點補充: input()永遠都會存成字串，所以就算你輸入的是數值，也會是字串的30。
6. while 非無窮迴圈(跑操場，每跑完一次就問教官可不可以不要跑了)
7. while True: 無窮迴圈，當程式在執行無窮迴圈想終止時，可按Ctrl+c。
8. time.sleep()。用來暫停程式碼執行，單位是秒，例如time.sleep(1)就是暫停1秒。使用時要先import time。time是python本身有的模組。

### Chapter3(import, for loop, list, 如何開啟檔案和range())

1. import(載入，裝別人寫好的功能進我們的程式)(可當成工具箱或一本書)。import時你要先下載你想import的檔案(模組)，才能import進來。random不用下載是因為python本身已經有了，在他的library裡(可當成圖書館)。專業術語: module(模組)是一個檔案，package(套件)就是有多個module裝在一起(資料夾)。import時可以載入整個package或module。想要有import是因為想要讓你的程式輕一點。
2. List(清單)(可想像成火車，火車有車廂編號，車廂內有人)。清單的符號是中括號，且可以混裝不同資料型別。index索引(代號)是從0開始(可想像0號車廂)。可用 .append()來加東西進入清單。用len()來取長度。用in來檢查東西在不在裡面。另外也可以清單內還有清單，變成很像多維度的array。
3. for loop的第一個叫做暫時的變數。也就是說用在for loop，在for loop外就不會使用。List在for loop的變數為把list內的項目一個一個取出來，str在for loop的變數為把一個一個字母取出來。另外，字串可以用in,len()，但不能用append，因為append()是list的method。
4. 讀取檔案 :   
   下面的code印出來會中間多一行，是因為在檔案內我們有按到enter去換行，所以其實有存到一個換行符號，也就是"倒斜線n"。然後print會幫你換行，所以pasta換行，然後又讀到倒斜線n又換行。另外，讀取檔案在for loop是以一行一行的去讀取為字串出來。

補充: r是讀取模式的意思，寫入模式是w。 as f就是當作是file的意思，意思就是把這個food.txt當作是f，因此從此以後只要需要用到這檔案，就只要打f就好。 with是自動關閉檔案的功能，以前沒with的時候，你開檔案都要自己手動關閉，現在有with就不用了。with那一block結束後就會自動關閉。

‘’’  
data = []  
with open('chapter4\_read.txt', 'r') as f:

for line in f:

print(line)

data.append(line)

print(data)

‘’’

補充: strip就是專門給字串用的，他會把換行或多餘的空格給去掉。append是專門給清單用的，所以兩者不能互用。利用strip可把左右兩邊多的換行或空白去掉。

1. range(): python內鍵功能:清單產生器。range通常都跟for loop搭配，不會單獨寫出來。range(5) 就是[0, 1, 2, 3, 4] 自動從0開始，但結尾值不包含
2. rang()之延伸用法:  
   A. range(2, 5) #[2, 3, 4]

B. range(2, 10, 3) #[2, 5, 8]

C. range(10, 3, -2) #[10, 8, 6, 4]

### Chapter4(寫入檔案, split(), continue用法, function和main function)

1. 字串可以做加與乘:  
   加法: 'abc' + '123' = 'abc123'   
   乘法: 'abc' \* 3 = 'abcabcabc'
2. 寫入檔案: (#'w'和'r'差在'w'可以覆寫檔案或若本來無檔案會新增此檔案。)

‘’’  
with open('products.csv', 'w', encoding = 'utf-8') as f:

f.write('商品,價格\n')

for p in products:

f.write(p[0] + ',' + p[1] + '\n')

‘’’

補充: CSV檔案，會以逗號來做資料的區隔，所以若你少了逗號，那他會把p[0]和p[1]塞在同一格。把CSV檔案拉到subline不會出現亂碼，但直接打開excel會有是因為excel沒設定用utf-8編碼去開啟，所以你要去excel那邊設定。從excel的資料那邊設定，導入txt檔。然後設定就好。utf-8是世界最有名的編碼方式，盡量都用這個

1. 寫入檔案+split():

‘’’

products = []

with open('products.csv', 'r', encoding = 'utf-8') as f:

for line in f:

s = line.strip().split(',')

name = s[0]

price = s[1]

products.append([name, price])

print(products)

‘’’

補充: 讀取檔案都是一行一行去讀，寫的話不是，想一行一行寫要+換行符。split()是字串用法，用來切割字串，切割點就看你想怎樣切，如果你想遇到','就切那就打split(',')。記住split完的東西會變成清單(list)。

1. continue用法(跳過一次迴圈，跳到下一次迴圈)。記住，continue和break一樣都只能寫在迴圈裡面。continue的使用時機通常都放在loop比較高的位置，因為這樣才有意義，才有東西跳過。
2. function (函式): function是用來收納程式碼的，它是個功能。要執行function要先define才能執行。寫程式時，盡量把程式碼都收納在function內，是良好的作法，這樣程式才清楚架構，而且有能重複使用的功能。
3. function的parameter (參數)(想像成投幣孔)的重點:

A. 當function需要外部資料，我們就設計投幣孔(參數)，把資料投進function。

B. 如果function有投幣孔，你就一定要投東西。除非你有預設值。

C. 投東西的時候，自動是按照參數(孔)的順序

D. 參數可以有預設值，那就不一定要投它了。

E. 投東西給參數的時候，可以明確指定要投到哪個參數。

1. function的return回傳: function如果有return，就可以把function執行的結果給存下來。可以想像成function運行完後會化身成為return的東西。
2. function可以有很多個return，但她遇到第一個return就會結束了。
3. function定義的時候記得要加冒號。
4. python規定function參數的部分，"沒預設值的"　一定要在　"有預設值的"　的前面。
5. 影片中我們提到，最好有一個main function來作為程式的進入點。我們在最下面寫上main()來呼叫(執行)這個 main function 但，許多程式碼你會看到​

‘’’

if \_\_name\_\_ == '\_\_main\_\_':​

main()

‘’’

他們多寫了一個if 。這個if因為不好解釋＋比較進階，我放到這邊來講，（這是可有可無的，最理想是有）最簡單的解釋就是，如果你沒有寫這個if，這個python檔被import的時候，會直接執行了main function。但通常這不是我們要的，我們如果import別的模組（也就是python檔），我們不希望有東西立刻執行，而是我們叫它執行再執行。所以加了這個if的意思就是要確認　這個python檔是被主要運行的python檔，而不是被import，也就是說我們是打python 這個檔案的名字來執行這個檔案，那這樣我們就希望main function執行。如果這個檔案是被import的話，就不要執行main function了。

舉一個例子好了:所謂的被import就是說我們寫了另一個檔案來import這個檔案，例如現在寫的是 a.py，後來又寫一個b.py　裡面寫到import a。如果a 裡面有main()　然後沒有寫那個if來檢查是不是被import，那b.py執行的時候一遇到import a，竟然會直接執行了a 的main function。

### Chatper5(dictionary)

1. 字典想像成真正的字典，去查單字那樣。字典的術語為，左邊那些就是key，對應的就叫做value。key功能主要用來查找。字典在寫的時候，都會寫成有多行式，最後也可以有逗點也沒差。記得去查的時候，一定都是用key去查。
2. 增加字典新的key以及value。沒有等號的時候會去查找，有等號會去增加。另外，如果本身字典已經有這個key了，那你去做這個動作就等於去更新key的value的意思。
3. 字典應用情形，用來把東西的屬性名稱記錄下來，因為清單很難做到這樣。而且通常所有網站的儲存資料都是用字典，不然就是字典和清單混用(很常混用)。

### Chapter6(cmd和sublime操作技巧)

1. cmd部分:
2. 按'方向鍵上'可以回到上一個指令
3. 調整CMD字大小 (在白色框視窗那裡滑鼠右鍵按內容。)
4. cd .. 可以回到上一層
5. cls 可以清掉畫面
6. 按'tab'可以自動完成未打完指令(一直按tab可以選擇你要哪個指令)
7. 反白選取文字 (如果是非win10可能不行，那就記得去內容那裡要去打勾'快速編輯模式')
8. 複製貼上(win10可以直接ctrl+c去複製，ctrl+v去貼上。也可以用滑鼠右鍵)(如果是非win10可能不行，那就記得去內容那裡打勾'快速編輯模式'，就可以用滑鼠右鍵作複製貼上，滑鼠右鍵是複製，然後去打字那裡再按一下右鍵可以貼上。)
9. insert鍵可以切換模式要不要取代文字。
10. sublime部分:
11. ctrl++ 放大。 ctrl-- 縮小。
12. ctrl+s 存檔。(檢查存檔沒，檔案名字旁邊是不是圈圈還是xx，是xx就代表存了。)
13. ctrl+n 開新檔案。
14. ctrl+a 全選。
15. 如何在程式碼中遊走:

* 按住ctrl可以一次跳一個字(詞)
* home鍵可以跳到開頭
* end鍵可以跳到結尾

1. 如何選取字:

* 選取單字: shift+左(或右)
* 選取整個詞: shift+ctrl+左(或右)
* 選取整行: shift+home(或end)

1. 善用自動完成加速打字。(打一打會自動出來字可以去選)
2. 如何快速解決tab跟空白鍵混用的問題:

* ctrl+h開啟底下的取代工具列
* find 輸入\t(也就是tab的符號)
* replace 輸入四個空白鍵
* regular expression記得啟動 (就是左上的.\*符號)
* 按replace all 來取代全部
* 所有的tab就變成4個空白鍵了

1. sublime如何更改預設值: 如果覺得Sublime時常發生tab跟space(空白鍵) 混用的錯誤，除了用Sublime操作技巧裡面提到的 ctrl+h 取代的方式去全部取代，還有一個方法是把Sublime的預設值改掉，讓它變成 你按下tab的時候，一定給你4個空白鍵，不要給tab。請看以下做法：

* 按下 Preferences -> Settings
* 會跳出一個新的Sublime視窗長這樣： (分成左右兩半)
* 此時你要在右手邊增加一行東西，記得第一行要補上逗點，變成這樣：
* 最後按ctrl + s來存檔 然後關掉這個視窗就可以了!

### Chapter7(常見的Error)

1. 空格錯誤(IndentationError/TabError)，IndentationError可能是多空格少空格的錯誤。TabError是Tab和空格混用的錯誤。

2. 語法錯誤(SyntaxError)(可能忘記括號，或是忘記打冒號等等.....)

3. 名稱錯誤(NameError)(沒先宣告就要執行他等等......)

### Chapter8(class)

1. class 類別是什麼

python中所有東西都是物件，他們都有class，差別在於有些是python本身已寫好的，像是int和string。有些你可以自己創造。

x = 5 是物件，它的class為int。

物件也有屬性，你可以打dir(x)去查屬性。

2. 如何寫class

class寫法:

‘’’

class <class名稱>:

def \_\_init\_\_(self):

.....

def <其他function>(self):

.....

‘’’

3. class之實際寫法與操作

‘’’

class Students: #字首大寫，前面有講過，小寫通常是function

def \_\_init\_\_(self):

#initialize 初始化之意，通常class都會先寫這個，把class初始設定都先做好。這是python內有的function，不是你自己創造的。只要這個class生出來或說創造時(被做成物件，或是直接使用)都會自動執行\_\_init\_\_()。

print('我誕生了')

def do\_hw(self):

print('我在做作業')

def study(self):

print('我在讀書')

print('執行 : s = Students()')

s = Students() #創造出物件，名為s，類別是students，

print('執行 : s.do\_hw()')

s.do\_hw() #使用class的do\_hw()。注意這邊不會執行\_\_init\_\_()，因為這是創造時才會執行。

print('執行 : Students().do\_hw()')

Students().do\_hw()

#也可以用這方式來執行，只是這個和上面那個的比起來的缺點就是，它沒有被存下來，所以每次呼叫他都會重新創造一個，所以\_\_init\_\_()都會重新被執行。

print('執行 : s.study()')

s.study()

print('---------')

‘’’

#補充: function是用來收納程式碼的, class可以想成收納function的。

4. self是什麼:

class內的每個function你都可以透過self去共用

想像成一個房子，每間房間你都可以透過鑰匙(self)去開並拿裡面的所有東西。

功能:

* + 使用self來存取自身上的function，而且self這個參數你不用去投它，python會幫你投。
  + 使用self來增加身上的屬性(非function的屬性)。
  + 典型的初始化參數設計。
  + 用同一個class來創作不同的物件。

5. self之實際操作: 使用self來增加身上的屬性，包括function和非function。

‘’’

class Students:

def \_\_init\_\_(self):

print('我誕生了')

self.name = 'Allen'

#增加身上的屬性。此非function的屬性，要注意屬性要有等號去設值才會創作出來，不能只寫self.name 這樣會當掉。

self.do\_hw('英文')

self.study()

def do\_hw(self, hw):

print('我在做作業', hw)

def study(self):

print('我在讀書')

s = Students()

print(dir(s)) #可以看到有個屬性叫做name

print('------')

print(s.name)

print('------')

‘’’

6. 將self.name變成為一個浮動屬性。即典型的初始化參數設計。

‘’’

class Students:

def \_\_init\_\_(self, n):

print('我誕生了')

self.name = n

self.do\_hw('英文')

self.study()

def do\_hw(self, hw):

print('我在做作業', hw)

def study(self):

print('我在讀書')

s1 = Students('John')

#將Allen投到\_\_init\_\_()的n參數。為啥不會投到do\_hw()的hw參數? 因為這邊是在創造出物件s，創造只會執行\_\_init\_\_()，所以當然是投在這個function。

s2 = Students('Peter')

print(s1.name, s2.name)

print('------')

‘’’

#補充: 通常我們會把n這個名稱取的跟你的屬性名字一樣，也就是self.name = name，但他們對應的參數位置和意義不一樣，別搞混。

7. 用同一個class來創造出不同的物件:

‘’’

class Students:

def \_\_init\_\_(self, name, score):

print('我誕生了')

self.name = name

self.score = score

def do\_hw(self, hw):

print('我在做作業', hw)

def study(self):

print('我在讀書')

self.score += 5 #它可以自己去\_\_init\_\_()去找出這個屬性，並且去做運算。

s1 = Students('John', 100)

s2 = Students('Peter', 95)

print(s1.name, s1.score)

print(s2.name, s2.score)

print('------')

s2.study() #會執行self.score那行，因為你呼叫了這個function。

print(s2.name, s2.score)

print('------')

‘’’

8. class的應用:

‘’’

students = [s1, s2] #等同一個清單裝成兩個字典

for s in students:

print(s.name, '的分數是', s.score)

‘’’

9. 寫class的三大好處:

1. 把相關function收納在一起 : 可做最大層級的收納。
2. 透過self來共用身上的屬性 : 當有許多function需用到同樣的參數時，可以透過self去共用。
3. 包裝程式碼 方便使用 : import時，可以很方便使用這些工具。

10. class總複習:

1. class名字第一個字要大寫
2. class裡的function(method)沒有順序的差別，沒有說\_\_init\_\_()一定要放在第一個，而且也可以還未先def就直接用self去使用這function。因為程式會直接先看過整個一遍再去執行class。
3. self.增加屬性不一定要在init裡，但是不這樣寫，不太好。因為想要創造出來時，就把非function的屬性都設定好。
4. class裡的function叫做method
5. method間隔一行, 而function間隔兩行
6. \_\_init\_\_()不一定要寫，但不寫的方法比較進階，這裡不講。
7. 繼承也是進階功能，這裡不講。

### Application: Line-Bot

1. 去 line 的 Messaging API 去啟動一個機器人

2. 建立新資料夾,建立GitHub專案 (打那六行指令)。

3. 新增程式並將程式命名為app.py，去 line的SDK的usage給copy下來放進程式內。(記得pip install該裝的)

4. 去 line 的 Messaging API，把channel secret和channel access token給複製下來到程式碼內對應的地方，另外把webhooks給啟用(沒有的話就不用啟用，他已經自動啟用了)。

5. 上傳程式碼到GitHub(三步驟)

6. CMD重開。 然後打heroku login來登錄heroku。(沒裝的話就先裝heroku的CLI)

7. 在heroku首頁去create new app。app name隨意你取，國家也隨意。

8. 去資料夾建立兩個檔案: Procfile 和 requirements.txt。檔案名必須打對，然後第一個檔案不用副檔名。在第一個檔案內打 web gunicorn app:app ，其中app:app當中第二個app是你的主要程式名字，即app.py只是.py不用打。第一個app是告知這是一個app的意思。第二個檔案內容為三行: line-bot-sdk, gunicorn, flask.

補充: Procfile目的是為了讓heroku去知道怎麼運行你的程式，還有要執行哪個主要程式檔案，requirements.txt目的是為了讓運行者(像是cmd)知道我們用到哪些套件，然後它會去找並安裝。

9. 把剛剛那個程式給放在heroku。去最下面Existing Git repository(因為我們已經有先在GitHub建立專案了)，先執行那行程式，再執行git push heroku master

10. 把資料夾內的檔案再新增到github(三步驟)。另外第一步驟可打 git add . 意思是把目前所在資料夾的東西全都新增進去。就不用git add 然後後面打一堆想加的檔名。

11. 把資料夾內的檔案再新增到heroku。只需打 git push heroku master 。

12. 去heroku的setting找到Domain有個app的網址，去把它複製，並到 line 的 Messaging API 的 webhooks的網址欄貼上去。最後記得網址最後結尾要是/callback，所以記得加上去。

補充: 可以在cmd打 heroku logs ，把遇到的程式問題都顯示出來。

13. 去LINE Messaging API那邊去掃QRcode加好友，即可開始測試。