

---今日---期考---進行方式

(上課前，請先交 **Homework** 紙上作業與自行用 **Email** 寄 **Lab#**給老師)

二、考試方式：

(1)**Close Book.** (暫時不可以使用電腦相關設備)

(2)時間 **18:30** 開始，預定到 **19:30**(若時間仍不足，可視情況延長到 **20:00**，最多到 **20:30**)

(3)請先在題目卷上，寫上學號姓名。交卷時一齊交回。

(4)儘量在試卷上留下解答的過程。讓我知道解題方法，而不是只有最後的答案。

(5)其他：

a.每個問題，除非特別說明，否則有任何缺點就視為不正確。

Good Luck.

Name: _____ Student ID: _____

※儘量都要有計算過程，每小題若有兩項錯，該小題就幾乎是 0 分。(※儘量都要有計算過程，每小題若有兩項錯，該小題就幾乎是 0 分。)

(注意:請務必要選擇指定題: 1, 2, 8, 12, 13, 14, 16, 17, 18, 29, 34, 37,38, 40, 41 有加●記號，其他任選 1 題即可)

1. ●Matching(配合題):

	1	2	3	4	5	6	7	8	9	10	11	12
選項												

Sol:

- python 最重要的保留字原來有 33 個在 py3.5 以後增加為 35 個，是增加 `async` 和那個指令: `---`
`await`.
- python 要加入行註解的特殊符號是 `---` #
- 要一行分為多行的特殊符號是 `---` \
- 要多行合一行的特殊符號是 `---` ;
- Existing Modules(現有可用的模塊)有很多，請問要導入的保留字是? `---` `import`,
- 如果有某些模塊(Modules)無法直接匯入，要自行安裝的指令是: `---` `pip`,
- 這個函數，是不需匯入就可直接用的函數 `---` `print()`,
- 字典(dict)類在宣告新實例代表符號是 `---` 大括號 { }
- 串列(list)宣告符號是 `---` 中括號 []
- 元組(tuple)宣告符號 `---` 小括號 ()
- python 要寫程式印出 python 關鍵字的寫法是? `->` `import keyword; lst=keyword.kwlist; print(lst)`
- python 要寫程式印出 python 內建函數(builtin function)的寫法是? `->` `funs=dir(__builtins__); f2=[f for f in funs if 'a'<=f[0]<='z']; print(f2)`
- 如果程式任務有下列特徵:「任務是根據另一 list 處理後，再產生 list 當作輸出」，此時就可考慮用什麼語法? `->` List Comprehension
- 一個自訂函數最後是 `return`，正常是傳回一個變數，但是在 python 可以寫成 `return a,b,c`，請問這是那種 python 的那種特性? `->` 自動打包(automatic packing)為 tuple
- 開發 javascript，想要使用外掛程式庫又可免下載的技術是? `->` CDN(Content Delivery Network) 內容傳遞網路。
- python 與 JavaScript 有很多相似之處，請問 Python 的 dict(字典)，相當於 JavaScript 的什麼? `->` object(物件對象)
- 可放數據的地方，主要有那兩類，第一類是資料檔，第二類是? `->` 資料庫。
- CSV 格式的特點是? `->` 以「逗號為分隔而且第一列是欄名稱」
- 用 python 要寫 json 資料到檔案的指令是? `->` `json.dump`
- 用 python 要到 json 檔案讀資料的指令是? `->` `json.load`
- mongoDB 預設的 port 是? `->` 27017
- jupyter notebook 預設的 port 是? `->` 8888

●2: 因為 iris 是知名的入門資料庫，是否有簡單方法就可印出 iris 的官方版介紹？如果有，請寫出這段程式。

Ans:

有很多方法，例如：只要下列三行 python 即可：

```
p3/35    dataViz_pyjs
from sklearn import datasets
iris = datasets.load_iris()
print(iris['DESCR'])
```

Quiz01 (2025au)

By Jiun-Ting Jiang

3: Seaborn 是基于 matplotlib 的 Python 可視化庫。提供高級界面來繪制有吸引力的統計圖形。他的簡稱比較奇怪！在官方網站上建議的縮寫是什麼？以 iris 為例，如何快速畫出散點圖矩陣（PairsPlots）？在圖上，可見到任兩屬性的關係，至少可看出三種關係是「正相關」「無關」還有是什麼關係？

Ans:

(1)正式的縮寫是 sns。

```
import seaborn as sns
```

(2)

在引入 seaborn 後，只要四行程式碼，就有可視化了！

以 iris 的散點圖矩陣（Pairs Plots）為例：

```
import seaborn as sns
```

```
sns.set(style="ticks")
```

```
df = sns.load_dataset("iris")
```

```
sns.pairplot(df, hue="species")
```

(3)

到任兩屬性的關係，至少可看出三種關係是「正相關」「無關」還有就是「負相關」。

4: 關於 RESTful web API，(a) REST(**R**epresentational **s**tate **t**ransfer)是什麼，簡單說明。(b)請問「REST」與「RESTful」的 ful，代表什麼？

Ans:

(a) REST 是一種軟件體系結構樣式，它定義了一組用於創建 Web 服務的約束。主要有六個指導性約束定義：客戶-伺服器結構，無狀態協議，Web 緩存，分層系統，客戶端腳本，統一界面

(b) 就像英文字 beauty 與 beautiful, 用 ful 的字尾代表形容詞，所以具有 REST 特徵的網站，就可稱為 RESTful web.

5: 用 python 完成爬蟲工作有很多方法，(1)如果是一般爬蟲，已經有很多相關工具,請舉出至少兩種。(2)如果使用爬蟲框架(Crawler framework)來解決較複雜的爬蟲，請舉出具工業等級的框架一種。

Ans:

(1)Python 已內建有不少好用的爬蟲工具，簡單情況，可直接使用(urllib 庫或 requests 庫)爬網頁，再配合 bs4(Beautiful Soup)就可用 python 式的函式來處理導航、搜尋、修改分析樹等問題，再麻煩點的情況，可使用 selenium 解決 js 的異步加載問題。

(2)有一個好用的爬蟲框架 Scrapy: Scrapy 是一個為了爬取網站資料，提取結構性資料而編寫的應用框架。可以應用在包括資料探勘，資訊處理或儲存歷史資料等一系列的程式中。

6: 基本上，要開發 python 是可以不必使用 IDE(整合開發環境)的，但是如果有合適的 IDE，是可以事半功倍的。請舉出 3 個好用的 IDE。

Ans:

官方版 IDLE, Pycharm, Eclipse+pydev, Spyde, 也有雲端可直接寫 python 的地方:如 kaggle 或 google

7：用 python 開發網站的框架(web framework)有很多，請簡單畫出兩種並簡單說明。

Ans:

在此，只舉出 5 個最實用的框架。

1. Django，是有助於構建高質量 Web 應程序的框架。它被廣泛用於快速發展的 API 和 Web 應程序。而且，Django 是一個免費開源的 Web 開發框架。也是 python 網站開發中最常用的語言。包括 Instagram, Mozilla Firefox, National Geographic, Pinterest, NASA 等在內的大型機構、公司，都在使用 Django 作為網站開發框架
2. CherryPy 是僅次於 Django 的網站開發框架，它最大的特性就是能夠同時運行好幾個 web 服務。
3. TurboGears 是一種 MVC 架構的網站開發框架。同時它具有 widgets 控制項，可以非常方便的生成一些特定的功能。
4. Pyramid 最大的特點就是靈活性和可擴展性（尤其和 Django 相比）
5. Flask 是一個輕量級框架，簡單易學，支持所有的 DBMS 和模板引擎。Flask 的優勢在於擁有眾多的擴展工具來創建基於網站的應用（適合初學者），flask 的團隊同時也開發了一個模板語言套件 Jinja2。

●8: 網路上有很多在線 JavaScript 編輯器，(a)可以在不啟動服務器測試 JavaScript 代碼，這種撰寫程式的方式是否有何缺點？ (b)若安裝好 anaconda，請寫出可簡單打開網頁服務器(Server)的一行指令。

Ans:

(a)

網路上有很多在線 JavaScript 編輯器，可以在不啟動服務器測試 JavaScript 代碼，所以很方便簡單測試。但是也有缺點，主要是失去了您最喜歡的編輯環境的功能，包括某些程式分析工具或熟悉的鍵盤快捷鍵等，而且，只適合於簡單問題，而不適合於複雜的多檔案專案。

(b)而且啟動自己服務器也不是很難的工作，以 python 為例，只要一行指令就可打開服務器了！例如：可先執行 anacond prompt 再打下列指令

```
$python -m http.server 8000
```

※注意，要先切換到沒安全顧慮的資料夾，再執行，這個資料夾就當作臨時 server 的根目錄。

9: Python 與 JavaScript 在邏輯運算上有何異同？特別要注意那個大小寫問題？

Ans:

JavaScript 與 C 語言的寫法相同。而 python 則是直接使用英文字。

而 python 是改用英文文字定義，特別要注意的大小寫問題是：pyhton 的保留字是 True , False, 而 JavaScript 則是 true, false.

參考下列比較。

python:	not and or	True False
javascript:	! &&	true false

10: 在 OOP 設計時，Python 的 self， JavaScript 的 this 有何異同？

Ans:

乍一看，很容易以為 Python 的 self 和 JavaScript 的 this，似乎是相同的，後者可以是前者的隱式版本，提供給所有類實例方法。

實際上，this 和 self 有很大的不同！

比較如下：

Python 的 self 是提供給每個類方法的變量：可以隨意命名，但不建議取其他名稱，代表該類實例。但是 JS 的 this 是一個關鍵字，它引用調用該方法的物件對象。此調用對象可以與方法的對象實例不同，並且 JavaScript 提供了調用和應用函數的方法。

11: 在 OOP 設計時，python 與 JS 各是以什麼為基礎？請簡單比較其差異？

Ans:

python 是以物件類別為基礎(Class-based)，而 JavaScript 是以原型為基礎(Prototype-based)。

對於習慣 C++/Java 或 Python 的設計師，在學 JS 的 OO 時，有可能引起混亂的原因，是必須注意 JS 是以原型為基礎 (Prototypes-based)。

使用 python 的 Class 與 JS 的 prototype 來設計 OO 時，在想法上的差異是什麼？

Raganwald Braithwaite 曾在 raganwald.com 指出這兩者的差別：「The difference between a prototype and a class is similar to the difference between a model home and a blueprint for a home.」

也就是說：用 Python 的類別來實例化時要先有一個「藍圖(blueprint)」，創建一個對象並在繼承樹中調用其各種構造函數。

※ 換句話說，要從頭開始設計，並構建原始的新類別實例。

反之，用 JavaScript 原型，您是從現有的樣品屋模型開始。

一開始就有屋子的模型！如果您只想要一個新客廳，則可以用油漆換成喜歡的顏色代替舊客廳。

如果您想要一個新大音樂學院，則需進行擴展。

※ 換句話說，一開始就有一些房間物件，已經可住人。仍可繼續修改和擴展現有物件。不需要從頭開始全新設計。

●12: 關於浮點數，python 與 JS 都是遵守那個規範？取整數的函數，在 JS 的 parseInt()與 python 的 int()，都是把浮點數作怎樣的處理？關於超大整數 JS 有最大範圍是多少？而 python 有特別處理，所以範圍會是多少？

Ans:

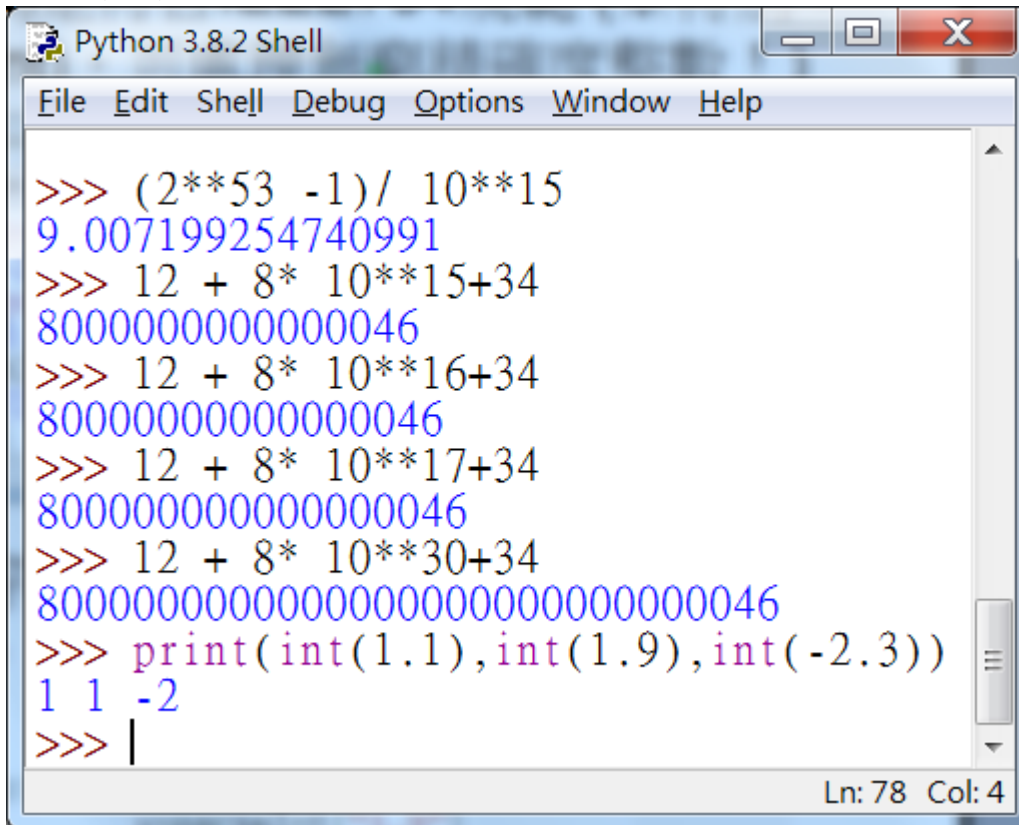
兩個語言的數值浮點數，都是符合 IEEE754 規範(其中,在分數部分則是有(52 bit))

※用 64 位二進位數來表示一個浮點數。其中，52 位用於表示有效數字，11 位用於表示指數，1 位用於表示符號。

parseInt 與 int()都是直接去尾數。不論正負也不論四捨五入！

JS 有最大範圍是 $2^{53}-1 \sim -2^{53}+1$ ，而 python3 的整數則是無窮大。

※ $2^{53}-1 = 9007199254740991$



```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help

>>> (2**53 -1)/ 10**15
9.007199254740991
>>> 12 + 8* 10**15+34
80000000000000046
>>> 12 + 8* 10**16+34
80000000000000046
>>> 12 + 8* 10**17+34
80000000000000046
>>> 12 + 8* 10**30+34
80000000000000000000000000000000046
>>> print(int(1.1),int(1.9),int(-2.3))
1 1 -2
>>> |
```

Ln: 78 Col: 4

```
>>> (2**53 -1)/ 10**15
>>> 12 + 8* 10**15+34
>>> 12 + 8* 10**16+34
>>> 12 + 8* 10**17+34
>>> 12 + 8* 10**30+34
80000000000000000000000000000000046
>>>
```

javascript 裡面，超過 2 的 53 次方後，就會因為有效數字而發生誤差！

```

> console.log((2**53 -1)/ 10**15)
9.007199254740991
< undefined
> console.log( 12 + 8* 10**15+34)
80000000000000046
< undefined
> console.log( 12 + 8* 10**16+34)
80000000000000050
< undefined
> console.log( 12 + 8* 10**17+34)
80000000000000000
< undefined
> console.log( 12 + 8* 10**18+34)
800000000000000000
< undefined
> x=parseInt("1.1")
y=parseInt("1.999")
z=parseInt("2.001")
z2=parseInt("-1.999")
z3=parseInt("-2.001")
console.log(x + "," + y + "," + z2 + "," + z3)
1,1,-1,-2

```

```

console.log((2**53 -1)/ 10**15)
console.log( 12 + 8* 10**15+34)
console.log( 12 + 8* 10**18+34)

```

●13: Python 與 JavaScript 在 function 的定義上，分別是使用那個保留字？又是如何指明函數區塊的範圍？

Ans:

Python:用保留字 def 定義，用冒號「：」與「縮格」來標示區塊。

JS:用保留字 function 定義，用大括號{}來標示區塊。

※JS 有另一種定義方式：函式運算式(function expression):(本課程幾乎都是採用此定義法！)

```

: def fun加到N(n):
    tot=0
    for i in range(n+1):
        tot+=i
    return tot
print(fun加到N(10))

```

55

def fun 加到 N(n):

tot=0

for i in range(n+1):

tot+=i

return tot

print(fun 加到 N(10))

```

> function fun加到N(n){
  var tot=0
  for (var i =0;i<=n; i++){
    tot+=i
  }
  return tot;
}
console.log(fun加到N(10))
55

< undefined

> var fun加到N2=function(n){
  var tot=0
  for (var i =0;i<=n; i++){
    tot+=i
  }
  return tot;
}
console.log(fun加到N2(10))
55

< undefined

> var fun加到N3=(function(n){
  var tot=0
  for (var i =0;i<=n; i++){
    tot+=i
  }
  return tot;
})
console.log(fun加到N3(10))
55

< undefined

> var fun加到N4=n => { //另外的寫法: => 箭頭函數。
  var tot=0
  for (var i =0;i<=n; i++){
    tot+=i
  }
  return tot;
}
console.log(fun加到N4(10))
55

< undefined

```

```

function fun 加到 N(n){
    var tot=0
    for (var i =0;i<=n; i++){
        tot+=i
    }
    return tot;
}
console.log(fun 加到 N(10))
//////////
var fun 加到 N2=function(n){
    var tot=0
    for (var i =0;i<=n; i++){
        tot+=i
    }
    return tot;
}
console.log(fun 加到 N2(10))
//////////
var fun 加到 N4=n => { //JS 的另外寫法: => 箭頭函數。
    var tot=0
    for (var i =0;i<=n; i++){
        tot+=i
    }
    return tot;
}
console.log(fun 加到 N4(10))

```

●14: 兩種語言都允許對字串使用單引號和雙引號。如果要在字串印出單引號，就可以用兩個雙引號來指明。反之要印出雙引號，就用兩個單引號來指明。那麼，如果同時要印出單引與雙引或者要印出反斜線字元，是否會有問題？

Ans:

如果同時要印出單引號與雙引號，就無法用單引或雙引來解決，而必須用到逸出字元「\」，所以要印出反斜字元，就必須用兩個反斜字元來代表一個反斜字元哦！

例如：

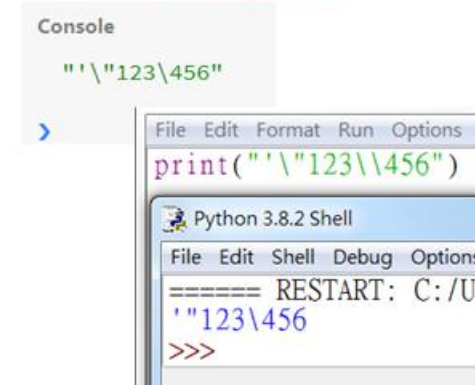
參考附圖！

```
console.log("\"123\\456")
```

或

```
print("\"123\\456")
```

```
console.log("\"123\\456")
```



15: 請從數據可視化機器的實務角度來看，python 很常用，但是 javascript 卻很少使用的技術？反之是否有 javascript 比較常用，而 python 卻很少用的技術呢？

Ans:

例如：JavaScript 常見的習慣用法是方法鏈接(method chaining)：方法鏈接是涉及使用「點」表示法，從其自己的方法返回一個對象，以便在結果上調用另一個方法，例如 D3.select 就有很高的使用率。

同時，在 python 則很少有這種用法。

反之。python 很常見到元組的反打包(Tuple Unpacking)。

例如，兩數的交換或多參數的傳遞...等。這些，JavaScript 都不可這樣使用。

●16: 若有一個像陣列的容器，請問要如何印出全部的項？請就 python 與 JavaScript 分別寫出這個簡單又重要的程式。

Ans:

通常，在追蹤列表(list)的同時，追蹤項目的索引(item's index)，常常會很有用。因此，Python 具有非常方便的內置列舉函數(enumerate)：(左圖程式)

JavaScript 的列表方法（例如 forEach 和函數的 map, reduce, filter），並將疊代的項目及其索引提供給回調函數(callback function)：(右圖程式)

```
File Edit Format Run Options Window Help
names = ['Alice', 'Bob', 'Carol']
for i, n in enumerate(names):
    print('%d: %s'%(i, n))

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:/Users/rr/AppData
0: Alice
1: Bob
2: Carol
>>>
```

```
JavaScript + No-Library (pure JS)
1 var names = ['Alice', 'Bob', 'Carol'];
2 names.forEach(function(n, i){
3     console.log(i + ': ' + n);
4 });
5
6
```

Running fiddle

```
"0: Alice"
"1: Bob"
"2: Carol"
>_
```

```
names = ['aa', 'bb', 'cc']
for i, n in enumerate(names):
    print('%d: %s'%(i, n))
```

```
=====
var names = ['aa', 'bb', 'cc'];
names.forEach(function(n, i){
    console.log(i + ': ' + n);
});
```

※補充 JavaScript 其他可行的類似寫法：

- callback 的另一種寫法
- for ... of 的寫法，原本只拿到物件 item，再配合 array 的方法：entries()就可拿到 index.

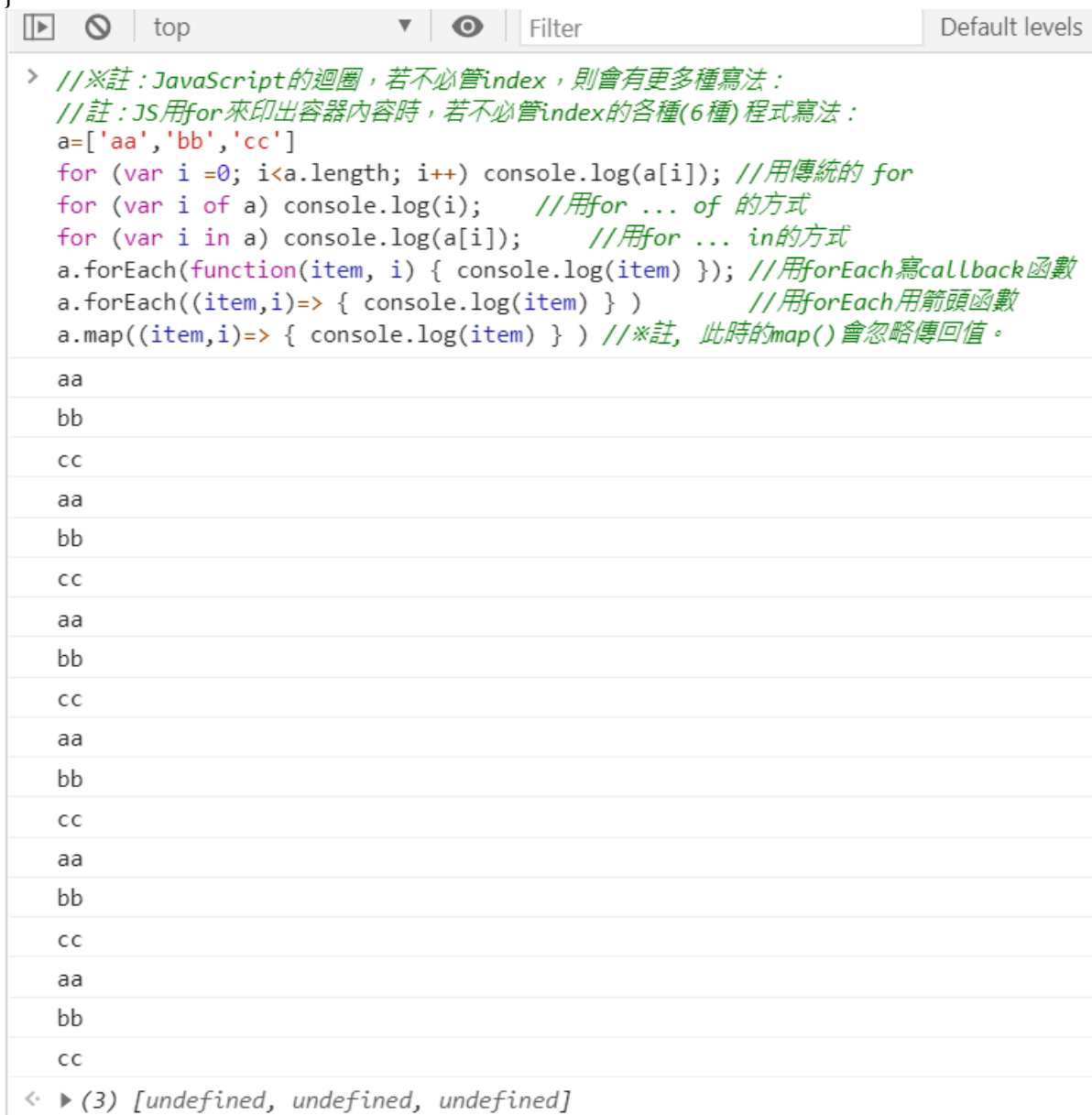
※entries() 方法會回傳一個包含陣列中每一個索引之鍵值對 (key/value pairs) 的新陣列迭代器 (Array Iterator) 物件。

```
Console Sources Elements Network Appl
top Filter
> const callback = (item,index) => {
  console.log(index + ': ' + item);
}
['aa', 'bb', 'cc'].forEach(callback)
0: aa
1: bb
2: cc
< undefined
> for (const [i, n] of ['aa', 'bb', 'cc'].entries()) {
  console.log(i + ': ' + n);
}
0: aa
1: bb
2: cc
< undefined
top Filter
> a=['aa','bb','cc']
a.map((item,i)=> { console.log(i+": "+ item) } )
0:aa
1:bb
2:cc
< (3) [undefined, undefined, undefined]
> |
```

//△定義 callback 函數的寫法：

```
const callback = (item,index) => {
    console.log(index + ': ' + item);
}
```

//△for .. of 的寫法+再搭配 array 的內建函數
for (const [i, n] of ['aa', 'bb', 'cc'].entries()) {
 console.log(i + ':' + n);
}



The screenshot shows a web browser's developer console. At the top, there are icons for running, stopping, and filtering, along with a 'Filter' input field and a 'Default levels' dropdown. Below this, a code editor displays the following JavaScript code with green comments:

```
> //※註：JavaScript的迴圈，若不必管index，則會有更多種寫法：  
//註：JS用for來印出容器內容時，若不必管index的各種(6種)程式寫法：  
a=[ 'aa', 'bb', 'cc' ]  
for (var i =0; i<a.length; i++) console.log(a[i]); //用傳統的 for  
for (var i of a) console.log(i); //用for ... of 的方式  
for (var i in a) console.log(a[i]); //用for ... in的方式  
a.forEach(function(item, i) { console.log(item) }); //用forEach寫callback函數  
a.forEach((item,i)=> { console.log(item) } ) //用forEach用箭頭函數  
a.map((item,i)=> { console.log(item) } ) //※註，此時的map()會忽略傳回值。
```

The output of the code is displayed below the code editor, showing the values 'aa', 'bb', and 'cc' printed multiple times, corresponding to the different loop methods used. At the bottom, the console shows the result of the map function: (3) [undefined, undefined, undefined].

//※註：JavaScript 的迴圈，若不必管 index，則會有更多種寫法：

//註：JS 用 for 來印出容器內容時，若不必管 index 的各種(6 種)程式寫法：

```
a=[ 'aa','bb','cc' ]
```

```
for (var i =0; i<a.length; i++) console.log(a[i]); //用傳統的 for
```

```
for (var i of a) console.log(i); //用 for ... of 的方式
```

```
for (var i in a) console.log(a[i]); //用 for ... in 的方式
```

```
a.forEach(function(item, i) { console.log(item) }); //用 forEach 寫 callback 函數
```

```
a.forEach((item,i)=> { console.log(item) } ) //用 forEach 用箭頭函數
```

```
a.map((item,i)=> { console.log(item) } ) //※註，此時的 map()會忽略傳回值，所以見到 3 個 undefined.
```

●17: Filter, Map, Reduce,是在『函數式程式設計』中常用的指令。請分別用 python 與 javascript, 解決相同的問題：問題是先有一個容器物件內容是 1~5, 然後 filter(過濾)出奇數項, 再使用 map(對應函數)將每項值都平方, 最後再 Reduce(減少)成只有一項的總和。

Ans:

JS:

```
var nums = [1, 2, 3, 4, 5];
var sum = nums.filter(function(o){ return o%2 })
    .map(function(o){ return o * o})
    .reduce(function(a, b){return a+b});
console.log('Sum of the odd squares is ' + sum);
```

或

```
var l=[1,2,3,4,5]
var isOdd = function(x){ return x%2; };
var sq    = function(x){ return x*x; };
var total = function(a,b){ return a+b; };
z=l.filter(isOdd).map(sq).reduce(total);
console.log(z);
```

#####

在 python:

```
#Python's list comprehensions
nums = [1,2,3,4,5]
odd_squares = [x * x for x in nums if x%2]
sum(odd_squares)
```

或

```
from functools import reduce
nums = [ 1, 2, 3, 4, 5]
odds = filter(lambda x: x % 2, nums)
odds_sq = map(lambda x: x * x, odds)
reduce(lambda x, y: x + y, odds_sq)
```

●18: 一個簡單問題同時寫出 py 與 js 的程式：重覆結構的實驗，計算 $1+2+\dots+10$ 的總和的實驗，實驗程式中，請把 10 改為自己的學號後四碼。

(a)使用 for 的版本，(b)使用 while 的寫法。

Ans:

(a)使用 for 的版本：參考下列程式與附圖。

```
tot=0
for i in range(10+1):
    tot+= i
print(tot)
```

```

tot=0
for i in range(10+1):
    tot+= i
print(tot)

```

Python 3.8.2 Shell

```

File Edit Shell Debug Options
===== RESTART: C:/USE
55
>>>

```

```

> var tot=0;
  var i;
  for (i=1;i<=10; i++){
    tot+=i
  }
  console.log(tot)
55
< undefined
>

```

```

var tot=0;
var i;
for (i=1;i<=10; i++){
    tot+=i
}
console.log(tot)

```

(b)while 版

```

t=0
i=1
while True:
    if i>10:
        break
    t+=i
    i+=1
print( t)
=====
t=0; i=1;
while (true){
    t+=i;
    if (i>=10) break;
    i+=1;
}
console.log(t)

```

19: 一個簡單問題同時寫出py與js的程式:先自訂小型數據假設有兩個學生的名字與一系列成績，請計算平均並印出。

Ans:

參考附圖。

```
student_data=[
  {'name':'Bob',
   'scores':[90,80,70,60]},
  {'name':'Alice',
   'scores':[45,75,45,96]},
]
```

```
s_data=student_data
print("before:",s_data)
for data in s_data:
    av=sum(data['scores'])/ float(len(data['scores']))
    data['average']=av
print("after:",s_data)
```

```
JavaScript
studentData=[
  {'name':'Bob',
   'scores':[90,80,70,60]},
  {'name':'Alice',
   'scores':[45,75,45,96]},
]

console.log("before-----")
studentData.forEach(function(sdata){
  console.log(sdata.name+ "--scored-->" + sdata.average);
})

studentData.forEach(function(sdata){
  var av=sdata.scores
    .reduce(function(prev,current){
      return prev+current;
    },0) / sdata.scores.length;

  sdata.average=av;
})

console.log("after-----")
studentData.forEach(function(sdata){
  console.log(sdata.name+ "--scored-->" + sdata.average);
})
```

```
Console
"before-----"
"Bob--scored-->undefined"
"Alice--scored-->undefined"
"after-----"
"Bob--scored-->75"
"Alice--scored-->65.25"
```

```
Python 3.8.2 Shell
student_data=[
  {'name':'Bob',
   'scores':[90,80,70,60]},
  {'name':'Alice',
   'scores':[45,75,45,96]},
]

s_data=student_data
print("before:",s_data)
for data in s_data:
    av=sum(data['scores'])/ float(len(data['scores']))
    data['average']=av
print("after:",s_data)

before: [{'name': 'Bob', 'scores': [90, 80, 70, 60]}, {'name': 'Alice', 'scores': [45, 75, 45, 96]}]
after: [{'name': 'Bob', 'scores': [90, 80, 70, 60], 'average': 75.0}, {'name': 'Alice', 'scores': [45, 75, 45, 96], 'average': 65.25}]
>>>
```

```
studentData=[
  {'name':'Bob',
   'scores':[90,80,70,60]},
  {'name':'Alice',
   'scores':[45,75,45,96]},
]
```

```
console.log("before-----")
```

```
studentData.forEach(function(sdata){
  console.log(sdata.name+ "--scored-->" + sdata.average);
})
```

```
studentData.forEach(function(sdata){
  var av=sdata.scores
  .reduce(function(prev,current){
    return prev+current;
  },0) / sdata.scores.length; //reduce 的初始值是 0

  sdata.average=av;
})
```

```
console.log("after-----")
studentData.forEach(function(sdata){
  console.log(sdata.name+ "--scored-->" + sdata.average);
})
```

//※ JS 的 reduce:

//array.reduce(callback[accumulator, currentValue, currentIndex, array], initialValue)

//※ .forEach 裡的 reduce 也可改寫為：

```
const redu_fun444 = (prev,current) => {
  return prev+current;
}
studentData.forEach(function(sdata){
  var av=sdata.scores
  .reduce(redu_fun444) / sdata.scores.length;
  sdata.average=av;
})
```

20: 一個簡單問題同時寫出 py 與 js 的程式：宣告一個容器，假定原來的數據是 1,2,3,4，後來加入字串'foo'到容器又拿出來。然後印出這個容器的全部內容。

Ans:

參考附圖。

```
l=[1,2,3,4] ; print(l)
l.append('foo'); print(l)
l.pop() ; print(l)
print(l[1:3])
print(l[-3:-1])
print(l[0:4:2])
```

```
z=[o*o for o in l]
print('z=>', z)
```

```
d={'a': 1, 'b':2,'c':3}
print('d=>', d)
```

```
print(d['a'])
print(d.get('a'))
```

```
print(d.get('z')) #None
#print(d['z']) #KeyError: 'z'
```

```
for key,value in d.items():
    print(key,value)
for key in d:
    print(key)
```

```
for value in d.values():
    print(value)
```

Lab#2-5-3a_python的容器list放入移走印出的實驗

■ 按一下

```

l=[1,2,3,4] ; print(l)
l.append('foo'); print(l)
l.pop() ; print(l)
print(l[1:3])
print(l[-3:-1])
print(l[0:4:2])

z=[0]*len(l)
print('z=>', z)

d={'a': 1, 'b':2, 'c':3}
print('d=>', d)

print(d['a'])
print(d.get('a'))

print(d.get('z')) #None
#print(d['z']) #KeyError: 'z'

for key,value in d.items():
    print(key,value)
for key in d:
    print(key)

for value in d.values():
    print(value)
```

Lab#2-5-3b_JS的容器list放入移走印出的實驗

方法一:用jsfiddle.net/測試

```

JavaScript + Processing.js 1.7
1 var l=[1,2,3,4] ; console.log(l);
2 l.push('foo') ; console.log(l);
3 l.pop() ; console.log(l);
4 console.log(l.slice(1,3))
5 console.log(l.slice(-3,-1))
6 console.log(l.slice(0,4,2))
7
8 z=l.map(function(o){ return o*o;})
9 console.log('z=>', z)
10
11 d={'a': 2, 'b':5, 'c':9}
12 console.log('d=>', d)
13
14 console.log(d['a'])
15 console.log(d['a'] === d.a) //兩種寫法 true
16 console.log(d.z) //undefined
17 console.log(d['z']) //undefined
18
19 //較舊的寫法 會取出key的index字串 的寫法
20 console.log("print dictionary...")
21 for ( var key in d ) {
22     if (d.hasOwnProperty(key)){
23         var item=d[key];
24         console.log(key, ":", item);
25     }
26 }
27 //較新的寫法
28 console.log("obj=>", Object.keys(d));
29 console.log("values =>", Object.values(d));
30
31 Object.keys(d).forEach(function (item,i) {
32     console.log(i,":")
33     console.log(item); // key
34     console.log(d[item]); // value
35 });
```

方法二:用JSbin.com/測試

```

JavaScript
var l=[1,2,3,4] ; console.log(l);
l.push('foo') ; console.log(l);
l.pop() ; console.log(l);
console.log(l.slice(1,3))
console.log(l.slice(-3,-1))
console.log(l.slice(0,4,2))

z=l.map(function(o){ return o*o;})
console.log('z=>', z)

d={'a': 2, 'b':5, 'c':9}
console.log('d=>', d)

console.log(d['a'])
console.log(d['a'] === d.a) //兩種寫法 true
console.log(d.z) //undefined
console.log(d['z']) //undefined

//較舊的寫法 會取出key的index字串 的寫法
console.log("print dictionary...")
for ( var key in d ) {
    if (d.hasOwnProperty(key)){
        var item=d[key];
        console.log(key, ":", item);
    }
}

//較新的寫法
console.log("obj=>", Object.keys(d));
console.log("values =>", Object.values(d));

Object.keys(d).forEach(function (item,i) {
    console.log(i,":")
    console.log(item); // key
    console.log(d[item]); // value
});
```

```

var l=[1,2,3,4] ; console.log(l);
l.push('foo') ; console.log(l);
```



```
p17/35      dataViz_pyjs
l.pop()      ; console.log(l);
console.log(l.slice(1,3) )
console.log(l.slice(-3,-1) )
console.log(l.slice(0,4,2) )
```

Quiz01 (2025au)

By Jiun-Ting Jiang

```
z=l.map(function(o){ return o*o;})
console.log('z=>', z)
```

```
d={'a': 2, 'b':5,'c':9}
console.log('d=>', d)
```

```
console.log(d['a'])
console.log(d['a'] === d.a) //兩種寫法 true
console.log(d.z ) //undefined
console.log(d['z'] ) //undefined
```

//較舊的寫法，會取出 key 的 index 字串 的寫法

```
console.log("print dictionary...")
for ( var key in d ) {
    if (d.hasOwnProperty(key)){
        var item=d[key];
        console.log(key, ":", item);
    }
}
```

//較新的寫法

```
console.log("obj=>", Object.keys(d));
console.log("values =>", Object.values(d));
```

```
Object.keys(d).forEach(function (item,i) {
    console.log(i,":")
    console.log(item); // key
    console.log(d[item]); // value
});
```

21: 關於 py 與 js，請就他們相同點、相異點至少各舉出 3 點。

Ans:

相同：

- 1.都是直釋式語言(interpreter)
- 2.都都有垃圾回收(garbage collection)機制。
- 3.都只要有文字編輯器(text editor)就可進行開發工作。
- 4.語言裡的函數都是一等公民(first-class citizens)，所以函數可當作參數。
- 5.※備註：資料容器 JS 的 object 就相當於 python 的 dict，而 JS 的 array 就相當於 python 的 list，而且操作方式也很相似！
- 6.※備註：兩種語言的字串都是單引號，或雙引號來包含。如果要在字串印出單引號，就可以用兩個雙引號來指明。如果同時要印出單引號與雙引號，就無法用單引或雙引來解決，而必須用到逸出字元「\」，所以要印出反斜字元，就必須用兩個反斜字元來代表一個反斜字元。

相異：

1. Python 是多執行緒，(※但是，並不適合進行平行加速)，對於檔案、網路與裝置通訊等等，執行緒能夠協助我們處理阻斷式 I/O，檔案存取時，不需回呼函數，只要像一般語言的打開檔案緩衝區，

就可由串流讀取資料。

反之，JavaScript 是單一執行緒執行環境、非阻斷式函數呼叫(single-threaded and non-blocking)，非同步式 I/O (asynchronous I/O) 。就是簡單檔案存取，都會有回呼函式(callback function)。

2. JS 原本是只出現在 Web，而且相比 python 僅有較少的輔助物件程式，不像 python 出現在各領域且有完善的標準庫。

3. OO 設計上，Python 有相當典型的物件導向的類別(object-oriented classes)，而 JS 是以原型為基礎(Prototype-based programming)去定義物件的鏈結關係，也就是說 JS 是沒有類別(classless)的語言。

4. JS 相比於 python，只有較少的通用數據處理庫(general-purpose data-processing)，所以 python 比較適合大量數據的處理與分析。

5. JS 是網路瀏覽器上的唯一語言，反之，python 想在瀏覽器上工作，則要靠翻譯程式協助。

6. JS 的整數是有最大值的限制必須小於 $2^{53}-1$ ，而 python 則是無窮大！

7. ※函數區塊。JS 是用 {} 來界定，而 python 則是用:與縮格來界定。

22. 在 python，如果想得到實驗的執行時間。可以加入 time 的模組，請問

(a)這個模組的函數 time()的時間，是由那一年開始計算的？

(b)請寫出可以測試實驗執行時間的簡單程式。

Ans:

(a)

裡面的 time()的時間是由是由 1970 開始計算的總秒數

(b)

寫法很多種。以下是一種簡單的寫法：

```
import time
```

```
start_time = time.time()
```

```
#main()
```

```
s=0
```

```
for i in range(1000):
```

```
    for j in range(1000):
```

```
        s+=1
```

```
print( (time.time() - start_time), "秒" )
```

輸出結果是：

0.6340363025665283 秒

```
>>>
```

23: 關於 py 與 js 都把函數當作一等公民，而且也都有一級物件(first-class object)，都可以把函數直接當作參數。同時這兩種語言也都可以用省略明確定義的匿名函數(Anonymous Function)來設計程式。請同時用這兩種語言，寫出可以用匿名函數(Anonymous Function)示範來解決相同問題的程式。

Ans:

參考：Lab#2-5_匿名函數(Anonymous Function)

的簡易實驗

Lab#2-5_匿名函數(Anonymous Function) 的簡易實驗

- 兩種語言的函數，都是一級物件(first-class object)，所以可以當作參數。
- 在JS是把'function'直接寫在參數區，
- Python則有lambad的保留字。

```
JavaScript
//有函數名稱的寫法
/*function say(word) {
  console.log(word);
}

function execute(someFunction, value) {
  someFunction(value);
}
execute(say, "Hello");*/

//匿名函數(Anonymous Function)的寫法
function execute(someFunction, value) {
  someFunction(value);
}
execute(function(word){console.log(word);}, "Hello");
```

```
Console
"Hello"
>
```

```
tttt.py - C:/Users/rr/AppData/Local/Programs/Python/Python38-32/tttt.py
File Edit Format Run Options Window Help

'''
###有函數名稱的寫法
def say(word):
    print(word)

def execute(someFunction, value):
    someFunction(value)
execute(say, "Hello");

###匿名函數(Anonymous Function)的寫法
'''
def execute(someFunction, value):
    someFunction(value)
execute(lambda word: print(word), "Hello")

Python 3.8.2 Shell
File Edit Shell
= RESTART:
on38-32/tt
Hello
>>>
```

```
//有函數名稱的寫法
/*function say(word) {
  console.log(word);
}

function execute(someFunction, value) {
  someFunction(value);
}
execute(say, "Hello");*/

//匿名函數(Anonymous Function)的寫法
function execute(someFunction, value) {
  someFunction(value);
}
execute(function(word){ console.log(word);}, "Hello");

=====
'''

###有函數名稱的寫法
def say(word):
    print(word)

def execute(someFunction, value):
    someFunction(value)
execute(say, "Hello");
```

```
##//匿名函數(Anonymous Function)的寫法
"""
def execute(someFunction, value):
    someFunction(value)
execute(lambda word: print(word), "Hello")
```

24. Python 語言真正影響傳值效果的，並不僅是看資料型態是否"immutable"?而是改變參數的操作！
例如有下列兩個簡單函數 `clear_a()`與 `clear_b()`。請問呼叫後的結果，那一個函數會真正把 `z` 清

```
>>> def clear_a(x):
>>>     x=[]
>>>
>>> def clear_b(x):
>>>     while x: x.pop()
>>>
>>> z=[1,2,3]
>>> clear_a(z)
>>> z=[1,2,3]
>>> clear_b(z)
```

為空的 list? 請說明理由。

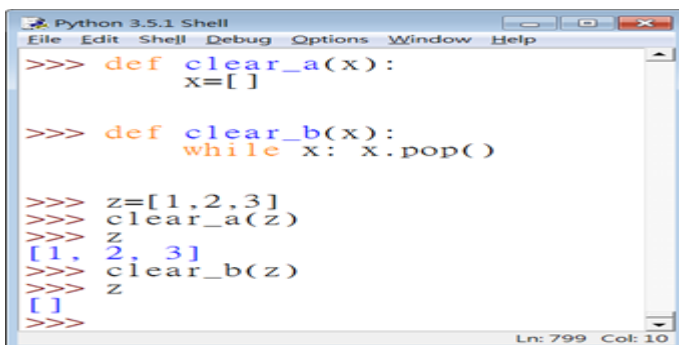
Ans:

參考圖。

`clear_b()`，則會真的會把原來的 list 清為空白。

但是 `clear_a()`，在返回後，並沒改變 `a` 原來的內容！

因為在函數內的 `x=[]`，只會改變別名的新配到的內容，而不會更改實際參數！



25. python 的 built-in function 都是很常用的函數。例如 print, max.. 其中可以查詢字元符的 Unicode 的函數是？可以把 Unicode 轉為字元符的函數是？

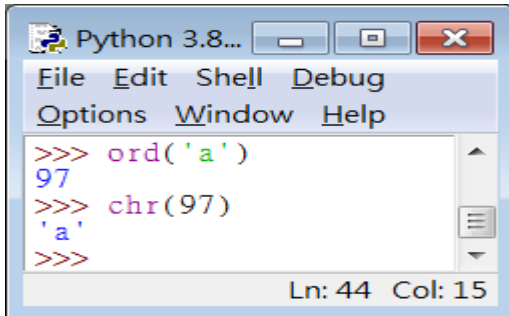
Ans:

這兩個是互為反函數：

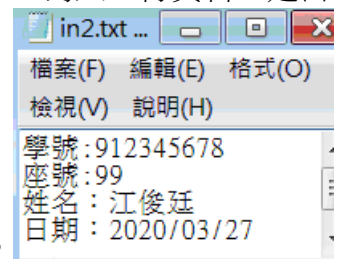
a)可以查詢字元符 Unicode 的函數是 ord(char)

b)可以把 Unicode 轉為字元符的函數是 chr(integer)

請參考下面實驗照相確認：



26: 請問要產生簡單文字檔的(a)簡單三步驟是什麼？(b)如果要在文字檔中，寫入三行資料，是自



己的學號、座號、姓名、日期 (如右圖)。可執行的完整程式要如何寫？

Ans:

(a)產生文字檔的簡單三步驟

(1)open 「可寫入」檔，

(2)Print 轉向，

(3)close

(b)

```
import sys
```

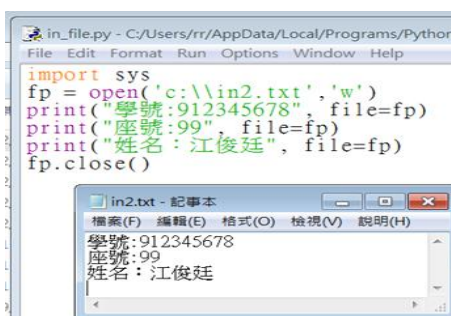
```
fp = open('c:\\in2.txt','w')
```

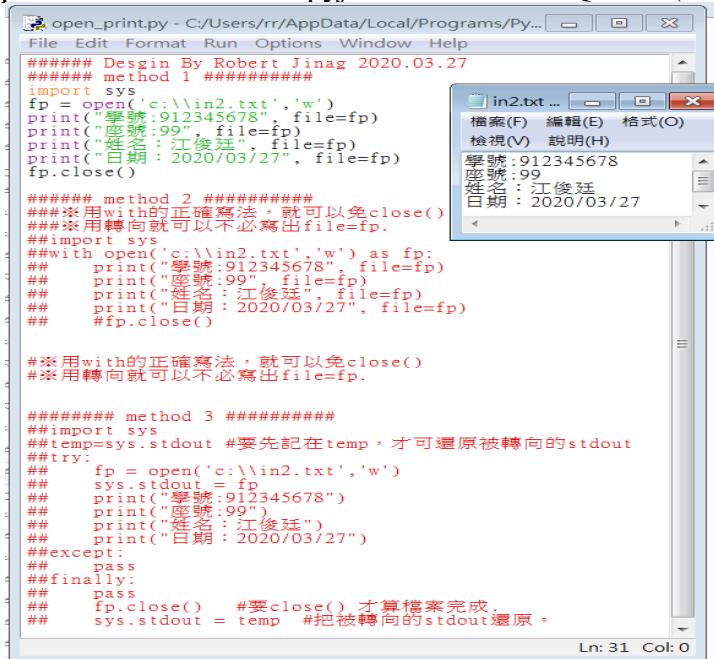
```
print("學號:912345678", file=fp)
```

```
print("座號:99", file=fp)
```

```
print("姓名:江俊廷", file=fp)
```

```
fp.close()
```





```

##### Desgin By Robert Jinag 2020.03.27
##### method 1 #####
import sys
fp = open('c:\\in2.txt','w')
print("學號:912345678", file=fp)
print("座號:99", file=fp)
print("姓名:江俊廷", file=fp)
print("日期:2020/03/27", file=fp)
fp.close()

##### method 2 #####
#####用with的正確寫法，就可以免close()
#####用轉向就可以不必寫出file=fp.
import sys
##with open('c:\\in2.txt','w') as fp:
##    print("學號:912345678", file=fp)
##    print("座號:99", file=fp)
##    print("姓名:江俊廷", file=fp)
##    print("日期:2020/03/27", file=fp)
##    #fp.close()

#####用with的正確寫法，就可以免close()
#####用轉向就可以不必寫出file=fp.

##### method 3 #####
import sys
temp=sys.stdout #要先記在temp，才可還原被轉向的stdout
##try:
##    fp = open('c:\\in2.txt','w')
##    sys.stdout = fp
##    print("學號:912345678")
##    print("座號:99")
##    print("姓名:江俊廷")
##    print("日期:2020/03/27")
##except:
##    pass
##finally:
##    pass
##    fp.close() #要close() 才算檔案完成.
##    sys.stdout = temp #把被轉向的stdout還原。

```

Ln: 31 Col: 0

in2.txt ...

檔案(F) 編輯(E) 格式(O)
檢視(V) 說明(H)

學號:912345678
座號:99
姓名:江俊廷
日期:2020/03/27

27. 一般在進行排序演算法設計時，都會用到 **swap**(交換)，請問一般程式語言(例如 **c++**)，要兩變數交換的寫法是什麼？而在 **python**，因為有自動打包(**Packing**)與解包(**unpacking**)，所以兩變數交換的寫法是什麼？

Ans:

c++語言，要兩變數交換的寫法要三行指定是：`t=a ; a=b;b=t`

而 **pyhton** 則是一個指定 `a,b = b,a`

28. 請自由選擇使用「理解語法(**comprehension syntax**)」或傳統的「重覆結構」來寫下列問題的程式 (下列 4 題選 2 題即可)。

- 「有一個 n ，計算所有小於 n 的正整數的平方和」的問題。
- 「有一個 n ，計算所有小於 n 的奇數的平方和」的問題。
- 要「產生 $[0, 2, 6, 12, 20, 30, 42, 56, 72, 90]$ 的 list」。
- 要「產生 $[a, b, c, ..., z]$ 的 list」

Ans:

a) 對於「有一個 n ，計算所有小於 n 的正整數的平方和」的問題，請寫出「有」與「沒有」使用理解語法(**comprehension syntax**)的程式。

```

1 #-----R1-4 Similar(R1-5) -----
2 #Ex1-R1.4: 小函數，輸入一個正整數n，返回所有小於n的正整數的平方和。
3 def sumsquares(k):#一般Loop的寫法
4     sum=0
5     for x in range(1,k):
6         sum+= x*x
7     return sum
8
9 print (sumsquares(2)) #want to add up 1*1
10 print (sumsquares(10)) # want to add up 1*1, 2*2, etc.
11 print (sumsquares(0))
12 print (sumsquares(-5))

```

```

1
285
0
0

```

```

1 #-----R1-5( similar as R1-4)-----
2 def sumsquares(k): #用理解語法(comprehension 的語法)
3     return (sum(x*x for x in range(1, k)))
4
5 print (sumsquares(2)) #want to add up 1*1
6 print (sumsquares(10)) # want to add up 1*1, 2*2, etc.
7 print (sumsquares(0))
8 print (sumsquares(-5))

```

```

1
285
0
0

```

b): 對於「有一個 n，計算所有小於 n 的奇數的平方和」的問題

```

1 #-----R1-7(( similar as R1-6)-----
2 def sumoddsquares(k): #用理解語法(comprehension 的語法)
3     return sum(x*x for x in range (1,k) if x%2==1)
4
5 print (sumoddsquares(2)) #want to add up 1*1
6 print (sumoddsquares(10)) # want to add up 1*1, 3*3, etc.
7 print (sumoddsquares(0))
8 print (sumoddsquares(4))
9 print (sumoddsquares(-5))
10 for x in range (10):
11     print (f'{x}: {sumoddsquares(x)}')

```

```

1
165
0
10
0
0: 0
1: 0
2: 1
3: 1
4: 10
5: 10
6: 35
7: 35
8: 84
9: 84

```

c) 如果要「產生[0, 2, 6, 12, 20, 30, 42, 56, 72, 90]的 list」。

d)如果要「產生[a, b, c, ...,z]的 list」請用理解語法(comprehension syntax)來完成。

```

1 #-----C1-18-----
2 #如何用「列表理解」語法來生成n*(n+1)的列表
3 #本題找出規律 是 n*(n+1)
4 print( [ n*(n+1) for n in range(0,10)] )

```

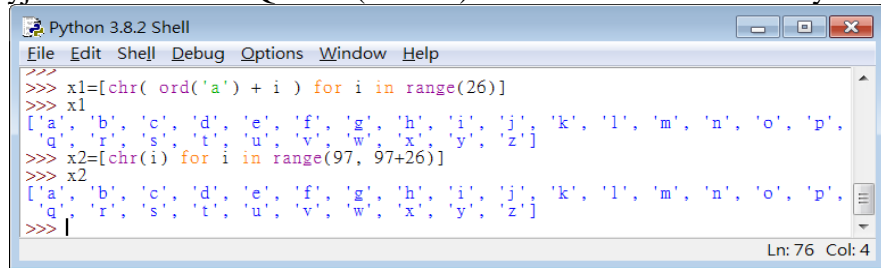
```
[0, 2, 6, 12, 20, 30, 42, 56, 72, 90]
```

```

1 #-----C1-19-----
2 #如何用「列表理解」語法來生成a~z 的列表
3 #chr(97)=>'a'          ord('a')=>97
4 print([chr( ord('a')+x) for x in range(26)])

```

```
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l',
'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x',
'y', 'z']
```



```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
>>> x1=[chr( ord('a') + i ) for i in range(26)]
>>> x1
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p',
 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']
>>> x2=[chr(i) for i in range(97, 97+26)]
>>> x2
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']
>>>
```

本題的其他類似寫法：

●29: 如何傳參數給範圍建構函數(**range**),可生成值分別為 **50,60,70,80,90** 的範圍資料？

ans:

#如果要產生 50, 60, 70, 80,90 的數列，則 range 裡面的參數，有三個：start, stop step

#step=10, start=50, 而 stop 只要剛好超過 90 即可，例如 91~100 之間(含)就可以。

print(list(range(50,90,10))) #這是錯誤示範

print(list(range(50,81,10))) #這是錯誤示範

print(list(range(50,100,10))) #這是正確

The Output is:

```
[50, 60, 70, 80]
```

```
[50, 60, 70, 80]
```

```
[50, 60, 70, 80, 90]
```

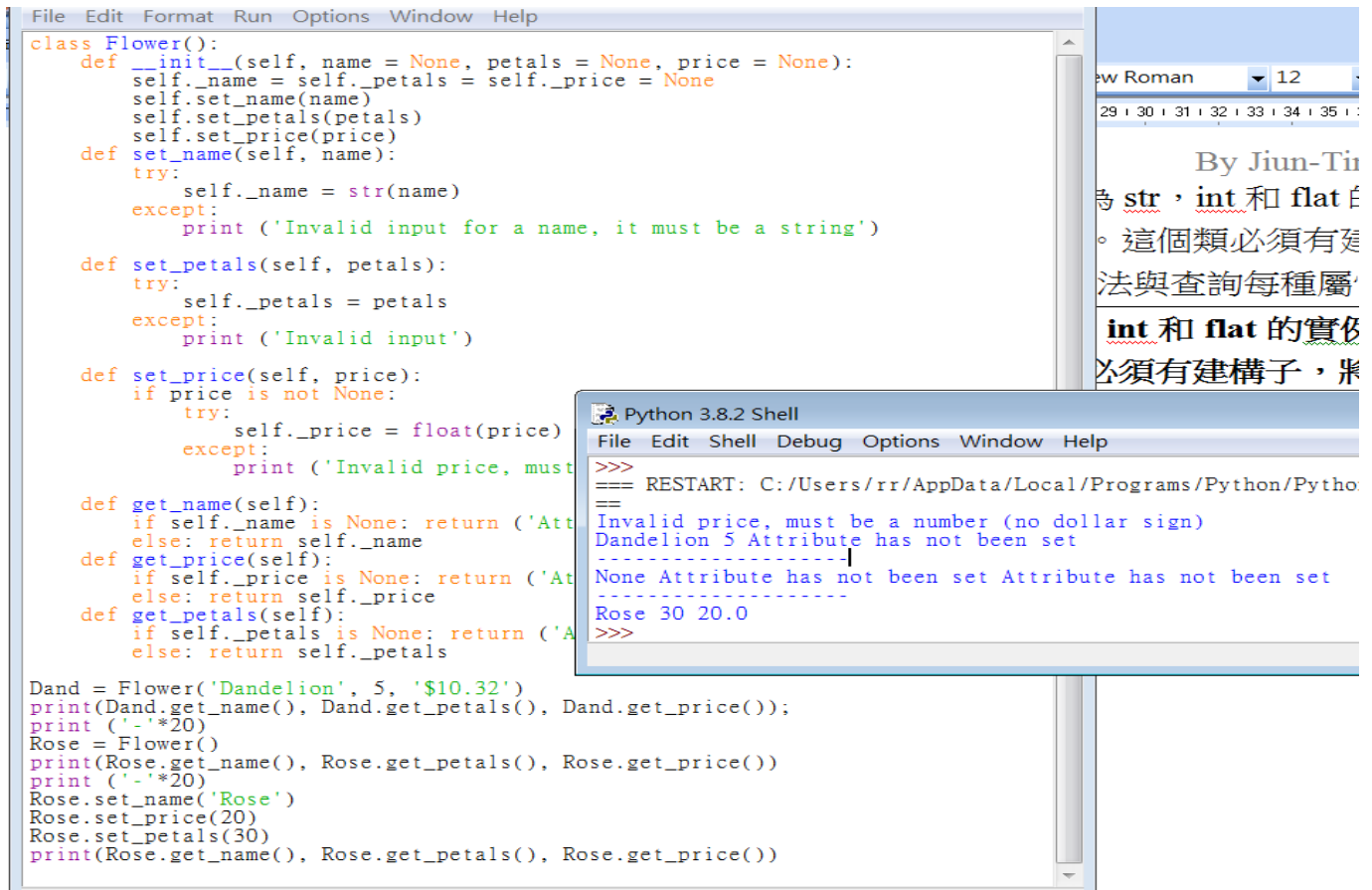
參考畫面：

```
#-----R1-9-----
#如果要產生 50, 60, 70, 80的數列，則range裡面的參數，有三個：start, stop step
#step= 10, start=50, 而stop只要大於80 又不會大於100 就可以。
print(list(range(50,90,10))) #這是 錯誤示範
print(list(range(50,81,10))) #這是 錯誤示範
print(list(range(50,100,10)))

[50, 60, 70, 80]
[50, 60, 70, 80]
[50, 60, 70, 80, 90]
```

※注意：因為 range 是一個 generator, 所以要 list(), 轉入 串列，才方便查看得到的全部數列。

30: 設計一個稱為"Flower"的類別(class)，它具有三個屬性分別為 **str**，**int** 和 **float** 的實例變量，分別代表 **flower** 的名稱(name)，花瓣數量(petals)和價格(price)。這個類必須有建構子，將每個變量「初始化為適當值」，並且有用於「設置」每種類型屬性值的方法與「查詢」每種屬性的值的方法。而且在初始化時，最好還有「預設值」的設計。



```
File Edit Format Run Options Window Help
class Flower():
    def __init__(self, name = None, petals = None, price = None):
        self._name = self._petals = self._price = None
        self.set_name(name)
        self.set_petals(petals)
        self.set_price(price)
    def set_name(self, name):
        try:
            self._name = str(name)
        except:
            print ('Invalid input for a name, it must be a string')
    def set_petals(self, petals):
        try:
            self._petals = petals
        except:
            print ('Invalid input')
    def set_price(self, price):
        if price is not None:
            try:
                self._price = float(price)
            except:
                print ('Invalid price, must be a number')
    def get_name(self):
        if self._name is None: return ('Attribute has not been set')
        else: return self._name
    def get_price(self):
        if self._price is None: return ('Attribute has not been set')
        else: return self._price
    def get_petals(self):
        if self._petals is None: return ('Attribute has not been set')
        else: return self._petals

Dand = Flower('Dandelion', 5, '$10.32')
print(Dand.get_name(), Dand.get_petals(), Dand.get_price());
print ('-'*20)
Rose = Flower()
print(Rose.get_name(), Rose.get_petals(), Rose.get_price())
print ('-'*20)
Rose.set_name('Rose')
Rose.set_price(20)
Rose.set_petals(30)
print(Rose.get_name(), Rose.get_petals(), Rose.get_price())
```

```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
>>>
=== RESTART: C:/Users/rr/AppData/Local/Programs/Python/Python38-Shell
>>>
Invalid price, must be a number (no dollar sign)
Dandelion 5 Attribute has not been set
-----
None Attribute has not been set Attribute has not been set
-----
Rose 30 20.0
>>>
```

31. python 開啟檔案，用 `with` 取代 `open` 主要好處有那些，至少寫出兩點？

Ans:

讀檔或寫檔時，都有可能產生 `IOError`，一旦出錯，後面的 `f.close()` 就不會呼叫。所以，為了保證無論是否出錯都能正確地關閉檔案，我們可以使用 `try ... finally` 來實現這種檢查機制。但是每次都這麼寫實在太繁瑣，所以，Python 引入了 `with` 語句來自動幫我們呼叫 `close()` 方法。

換言之，用 `with` 的開檔方式，會在 `with` 結束後，自動關檔，就不必再 `close()`，而且在具有 `try` 效果的同時，又可以比 `try` 的結構簡潔。

32. 關於數據庫的 SQL 與 NoSQL 請簡單說明與比較，而且指出 python 在處理這些數據時常用的工具庫是什麼？

Ans:

關於數據庫：

(第 1 類)SQL 資料庫--關聯式資料庫(Relational database)：

可用標準的資料結構化查詢語言(SQL, Structured Query Language) 結構化查詢語言),對關聯式資料庫中資料的檢索和操作。python 有一些很棒的庫可以與 SQL 數據庫進行交互，例如 `SQLAlchemy`，是不錯的入門指南。

(第 2 類)NoSQL 非關聯式資料庫(Not Only SQL)：

數據儲存可以不需要固定的表格模式，也可認為就是「無綱要 SQL (No Schema SQL)」(或「無表格架構資料庫」)

`MongoDB` 是一種較新的 NoSQL 資料庫。配合 Python 的 `pymongo` 庫，適合新手。

33. CSV 檔的問題(a)如果在 python 已經有一個符合 JSON 格式的資料物件稱為 `nobel_winners`,請寫出可以將資料寫入 CSV 檔的程式碼。(b) python 程式設計，請參考 `iris`，自行設計類似 `iris` 的資料 3 筆以上即可。需可以將這些資料寫入 csv 的檔案，並且讀入 csv 檔並印出來。(c)如果有 csv 的網路資料，打算先存成 csv 檔，再解碼成字典物件 `dict`，請完成這段簡單的 python 爬蟲程式。(d)為何在處理數據資料時，即使 python 可輕易處理的 csv 格式,仍建議使用 `pandas` 而不是用內建的程式庫 `csv`?

Ans:

(a) 參考 Lab#3-13 建立一個簡單的資料檔案 CSV-以諾貝爾獎得主為例(`nobel`)

```
cols = nobel_winners[0].keys()
with open('./nobel_winners.csv', 'w') as f:
    f.write(','.join(cols) + '\n')
    for o in nobel_winners:
        row = [str(o[col]) for col in cols]
        f.write(','.join(row) + '\n')
```

(b)

`import csv` ##※注意 csv 裡「,」逗號前後都不可有多餘的空白哦！

注意 aaa 前面有空白，讀入後，名稱就會是「aaa」哦！

```
str_irir_txt="""花萼長,花萼寬,花瓣長,花瓣寬,花名
```

```
1.1,2.1,2.1,3.1,aaa
```

```
2.2,3.2,4.2,5.2,bbb
```

```
7.2,7.2,7.2,7.2,ccc"""
```

```
with open('./iris.txt', 'w', newline='') as csvfile:
    print(str_irir_txt,file=csvfile)
```

開啟 CSV 檔案

```
with open('./iris.txt', newline='') as csvfile:
```

```
    # 讀取 CSV 檔案內容
```

```
    rows = csv.reader(csvfile)
```

```
    #rows = csv.reader(csvfile, delimiter=',') #用:來分隔資料，例如 linux 的 passwd
```

```
    # 以迴圈輸出每一列
```

```
    for row in rows:
```

```
        print(row)
```

(c)

##網路上 opendata 的 CSV 用爬蟲方法

```
import requests
```

```
http='..'
```

```
response = requests.get(http) ##直接 get 會自動生成一個 session()
```

```
dicts=response.text
```

```
with open('in.csv', 'w') as f:
```

```
    f.write(dicts)
```

讀入後，並直接轉為 Dictionary 的字典物件。

```
import csv
```

```
with open('in.csv') as csvfile:
```

```
    # 讀取 CSV 檔內容，將每一列轉成一個 dictionary
```

```
    rows = csv.DictReader(csvfile)
```

```
    #print(rows) # type : csv.DictReader object
```

```
    # 以迴圈輸出指定欄位
```

```
    for row in rows:
```

```
        print(row)
```

(d)

因為 pandas 處理時，可以更聰明的判斷各種合理的格式與檔案類型，可說比直接用 csv 程式庫來處理 csv 會更方便更正確！例如：內建的 csv.DictReader 閱讀器，能讀檔並轉為字典物件，但是「不會從文件判斷數據類型」，會將所有內容都解釋為字串。此時，如果是用 Pandas 來讀資料，則 Python 「通常會成功地猜測出正確的數據類型」。

●34. python 在用 json.dumps 轉換 dict(字典)為 json 時，

(a)對於 dict 裡的 pair_data : (key:value), 裡的 key, 與 value 是否有特別要注意的地方？

(b) json 執行 dumps 後的結果：參考下列程式，寫出會印出來的結果。

```
import json
```

```
t2={"today": '9', 'is': "7", "nice": 8, "day": [6,5,.4,True,None] }
```

```
t2 = json.dumps(t2)
```

```
t2 #print(t2)
```

(c)加分題：如果已有 Json 的網路資料，打算直接爬下 JSON 數據，並且解碼成字典物件 dict，請完成這段簡單的 python 爬蟲程式。

Ans:

(a)

用 json.dumps()的指令將其轉換成 Json 格式。輸出與原 data 幾乎相同！

##注意 1：python 在處理 json 時，特別是對於 dict 裡的 pair_data : (key:value), 裡的 key, 會轉成雙引號！

##注意 2: 其中的 value, 數值就沒有雙引號，字串會有雙引號！

(b)

除了(a)要注意。還要注意保留字的定義與大小寫的不同。

```
'{"today": "9", "is": "7", "nice": 8, "day": [6, 5, 0.4, true, null]}'
```

(c)加分題

簡單的參考程式：

##網路上 opendata 的 JSON 用爬蟲方法

```
import requests
```

```
http='....'
```

```
response = requests.get(http) ##直接 get 會自動生成一個 session()
```

```
dic=response.json()
```

```
for i in range(3):
```

```
    print(dic[i])
```

35. SQL 資料庫要解決 CRUD 的問題，請問 CRUD 是那四個英文字分別代表那四種工作？請配合 Sql 說明。

Ans:

SQL 資料庫要知道的基本 CRUD 四種問題

(Create)新增資料

```
insert into 表名 values('資料 1'，'資料 2'，'資料 3')；
```

```
insert into 表名 (資料項 1，資料項 2，資料項 3) values ('資料 1'，'資料 2'，'資料 3')；
```

(Read)讀取資料

```
select 資料項 from 表名 where 條件；
```

(Update)更新資料

```
update 表名 set 屬性項 where 表名；
```

(Delete)刪除資料

```
delete from 表名 where 條件；
```

```
delete from 表名； //表中資料全部刪除
```

36. MongoDB 是非關聯式資料庫 (NoSQL)，(a)相對於 RDBMS 有那些特點？ (b)用 MongoDB 建立資料時，與一般資料庫加入自動增量的 id 不同，而是採用 ObjectID 的設計，請問這樣有何好處？ (c)是 NoSQL 中很適合入門數據師採用，可能的適用場合有那些？ (d)是否有那些場合，比較不適合 mongoDB 呢？ (e)Google 有推出 Firebase 的資料庫服務，也是 NoSQL，與 mongoDB 有那些比較明顯的差異？

Ans:

(a)

NoSQL 是非關聯式資料庫，當中有許多的設計方式，會與關聯式 DBMS 有所不同。主要差別有下列幾點： 不需要事先定義好資料綱要(schema) 以及資料之間的關聯、可以自由新增欄位，不需要回頭修改過去的資料文件 (document)、可以自由定義資料文件 (document) 的結構。

(b)

MongoDB 採用 ObjectId，而不是其他常規做法（比如自動增加的主鍵）的主要原因，是在多伺服器上同步自動增加主鍵值既費力又費時！另外，也有在 ObjectId 加入 Timestam

ps 與機器行程相關的訊息，方便在分散式場景的操作。

(c)

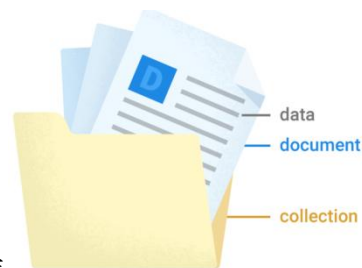
適合考慮用 **MongoDB** 的場合，主要有下列幾點：想快速啟動小專案測試 idea、資料格式不確定 (unstable schema)，而未來很有可能調整、資料表格之間沒有複雜的關聯、或未來讀取資料時不需要使用 JOIN 的功能、著重在快速讀取資料與可用性，而非 ACID.

(d)

NoSQL 的資料庫為了要達到快速讀資料與可用性，普遍都只強調「最終一致性，在過程中可能就有一段時間資料不同步」！不像 SQL 資料庫都會要求確保 Transaction 交叉執行的可靠！所以比較不適合需要交易(Transaction)的場景。另外 SQL 資料庫，大都會強調正規畫分割資料表，而 mongoDB 則不需先規畫，也可自由新增欄位，所以也比較不適合要多資料表格 Join 的應用場合。

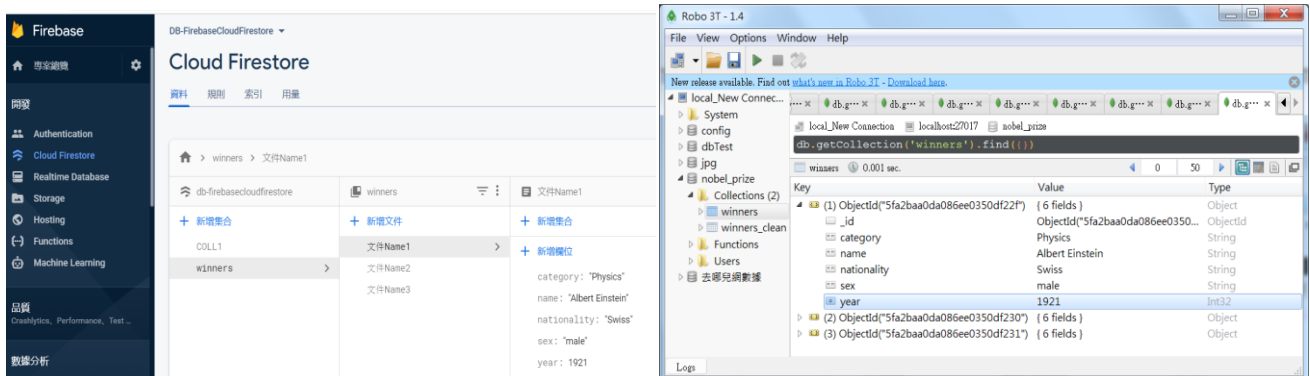
(e)

Google 的 Firebase 資料庫，也是 NoSQL，而且又有「Real-time 資料庫」與「使用者身份認證」，所以方便身份認證，而且在資料庫發生更新時可主動推送到客戶端，適合開發手機 App 時的雲端應用。



※本題補充：mongoDB 與 Firebase 都是由 collection 組成 (此時的 collection 類似 SQL 裡的 table)

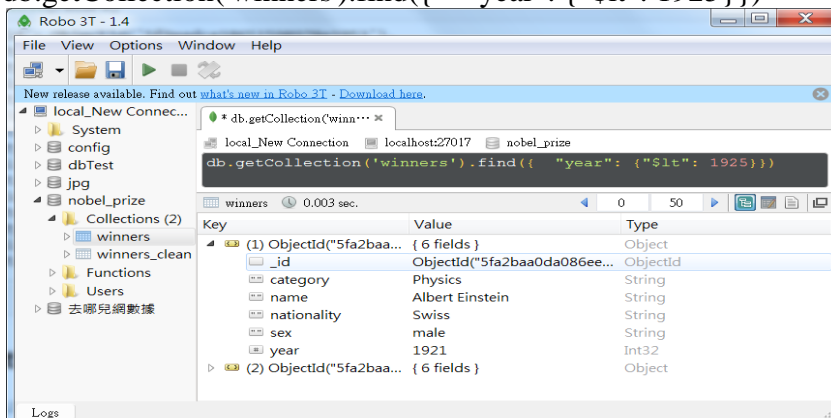
左圖是 Firebase，右圖是 mongoDB



關於 mongoDB，要執行查詢，可在 find ()裡面加入條件即可：

下圖是要查詢「得獎者的年度 小於(less than) 1925 年」的寫法：

`db.getCollection('winners').find({ "year": { "$lt": 1925 } })`



※ numpy 的重要資料結構是 ndarray, 裡面可以表達矩陣是多少乘多少的關鍵屬性是? --> 形狀(shape)

numpy

● 37. Numpy (a) Numpy 的基本操作如下，結果為何？

```
a=np.arange(6).reshape([2,3])
z=( (a+1) *2-2)/2.0
print(z)
```

(b)有以下三個指令，請問執行後的矩陣結果是？

```
a = np.ones([5,7])
a[:5:3] = 6
a[:,2:5]=np.nan
```

(c)若有以下指令，請問執行後的結果，u,v,w,x 分別是什麼？

```
a=np.arange(16)
a=a.reshape([2,2,4])
u=a[1]
v=a[1,0]
w=a[0, :, 1:3]
x=a[1,1, :-1]
print("u=",u, "\nv=",v, "\nw=\n",w,"\nx=",x, )
```

Ans:

```
(a)
# rray([[0., 1., 2.],
#       [3., 4., 5.]])
```

```
(b)
# array([[ 6.,  6., nan, nan, nan,  6.,  6.],
#        [ 1.,  1., nan, nan, nan,  1.,  1.],
#        [ 1.,  1., nan, nan, nan,  1.,  1.],
#        [ 6.,  6., nan, nan, nan,  6.,  6.],
#        [ 1.,  1., nan, nan, nan,  1.,  1.]])
[[ 6.  6. nan nan nan  6.  6.]
 [ 1.  1. nan nan nan  1.  1.]
 [ 1.  1. nan nan nan  1.  1.]
 [ 6.  6. nan nan nan  6.  6.]
 [ 1.  1. nan nan nan  1.  1.]
```

```
(c)
# u= [[ 8  9 10 11]
#      [12 13 14 15]]
# v= [ 8  9 10 11]
# w=
# [[1 2]
#  [5 6]]
# x= [12 13 14]
```

●38. Numpy 的 slice 操作時，「省略號...」與「冒號:」，都是可用又重要的切片操作，請問分別代表什麼效果？又有那些要注意的地方？

Ans:

省略號...」與「冒號:」，都是可用的符號！

(a)冒號: 的操作 是 slice 對每個維度軸的指定方法, 與 list 的 slice 操作方式相同, 就是 start: end: step
 注意 1: 其中的 start: end: step 表示是 start: end, 由 0 開始, 不包含 end, 若只一個冒號,
 注意 2: 如果在冒號指定資料數少於維數軸時, 則假定其餘沒有指定的維, 都是視為, 全部選擇 (:).
 (b)「省略號...」的操作 是表示省略多個軸的指定, 代表這些省略的維, 都是視為, 全部選擇 (:).
 注意 3: 只想直接選最後一個維的軸, 就適合用「省略號...」

39. 移動平均值是基於最近 n 個值的移動窗口的一系列平均值, 也稱為移動平均值或滾動平均值。

(a)請利用 numpy 設計這個函數。

(b1)如果 $a = \text{np.arange}(3, 9)$, 而且 $n=3$ 的時候, 請問 a 是什麼? $\text{moving_average}(a, n=3)$ 的結果是?

(b2)另外, 如果 $a = \text{np.arange}(10)$, 而且 $n=4$, 請問 a 是什麼? $\text{moving_average}(a, n=4)$ 的結果是?

Ans:

(a) NumPy 的移動平均線

```
def moving_average(a, n=3):
    ret = np.cumsum(a)
    ret[n:] = ret[n:] - ret[:-n]
    return ret[n - 1:] / n
```

(b1) $a = \text{np.arange}(3, 9)$. 時

$a = \text{array}([3, 4, 5, 6, 7, 8])$

$\text{moving_average}(a, n=3) = \text{array}([4., 5., 6., 7.])$

(b2) $a = \text{array}([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])$

$\text{moving_average}(a, n=4) = \text{array}([1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5])$

● 40. 要引入 NumPy 的擴充程式庫, (a)慣例寫法是什麼? (b)是否可以直接寫 `import numpy` 或 `from numpy import *`, 若這樣引用, 可能分別會有什麼問題呢?

Ans:

(a)慣用的寫法是: `import numpy as np`

(b1)若只寫成「`import numpy`」則使用時要寫成 `numpy.array()`, 比較不方便。

(b2)若直接用 `from numpy import *`, 雖然可以省略寫模組名稱, 但是也會有問題, 因為會把模組內的變數名稱, 全部都加到全域命名空間! 這樣就浪費了 python 在模組命名空間的管理機制!

所以在使用 numpy 相關的功能時, 儘可能就是用「`np.`」來引用!

```
In [1]: 1 c=sin(3.14/2)
        2 c

-----
NameError                                Tr
<ipython-input-1-d4141900c4ce> in <module>
----> 1 c=sin(3.14/2)
        2 c

NameError: name 'sin' is not defined

In [2]: 1 from numpy import *
        2 a = array([1, 2, 3])
        3 a + a

Out[2]: array([2, 4, 6])

In [3]: 1 b=sin(3.14/2)
        2 b

Out[3]: 0.9999996829318346
```

雖然可以省略寫
模組名稱, 但是
也會有問題, 因
為, 會把模組內的
變數名稱, 全部
都加到全域命名
空間!

●41. (a) 知名 AI 工具已是百花齊放。關於「AI 聊天機器人」目前可免費，產生程式碼效果又很好的工具，請舉出 3 個。 (b) 數據視覺化，若概分為兩部份。第一是取得資料、第二是視覺化，任務一：「爬蟲程式」若有一個 json 的網址。請用 AI 工具。設計一個可執行的「可由 json 網址完成爬蟲的 python」參考：政府資料開放平臺網址：<https://data.gov.tw/> 任務二：「視覺化程式」若有一個已放在 dataframe 的資料框的資料，例如：某班同學的月考成績，請自行產生假設性資料。並用 javascript 與 python 呈現兩種不同的視覺化效果。請自選一任務，簡單寫出「爬蟲程式」「視覺化程式」或「其他」相關，的 Prompt(提示詞)。

Ans:

(a)各種知名 AI 工具的特點與比較。包括 **ChatGPT** (由 OpenAI 開發)，**Claude**(由 Anthropic 開發)，**Copilot** (由 OpenAI+ms)。**Gemini**：(由 Google 開發)，**Perplexity**(由 Perplexity 開發) 與其他。

(b)

例如 1:

請幫忙製作一組數據，模擬某次月考全班的成績有「英文、數學、計概」三科，用 python 來寫，並且可放在 dataframe，然後，增加修改程式，可讓第一位學生姓名改為「江俊廷」，學號是「987654321」，成績可修改為 99 分,98 分,97 分。完成後，全部放在一個 JSON 的檔案中。

例如 2:

請幫忙製作 web-game。使用 html + css + javascript (js) ,寫一個可直接在 chrome 執行的 web-game 小遊戲. 例如 tic-tac-toe 圈叉井字遊戲。在遊戲開始之前，要輸入玩家姓名。而且要顯示玩家勝負的次數，畫面最上方，要有訊息顯示：「創作者:江俊廷」。並且有一個超連結，可以連到文化大學(<https://www.pccu.edu.tw/>)

(※請將完整的 html, css,javascript 放在一個 html 檔)

ChatGPT 的示例

今天

Tic Tac Toe 網頁遊戲 ...

CodeCombat 比賽準備技巧

請幫忙製作web-game。使用html + css + javascript (js) ,寫一個可直接在chrome執行的web-game小遊戲。例如tic-tac-toe圖又井字遊戲。在遊戲開始之前，要輸入玩家姓名。而且要顯示玩家勝負的次數，畫面最上方，要有訊息顯示：「創作者:江俊廷」。並且有一個超連結，可以連到文化大學(<https://www.pccu.edu.tw/>)

記憶已更新

Here's a simple Tic-Tac-Toe web game using HTML, CSS, and JavaScript, including the requested features like player name input, win-loss tracking, and the display of creator information.

html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Tic-Tac-Toe</title>
</head>
```

複製程式碼

傳訊息給 ChatGPT

Claude 的示例

Claude Tic-Tac-Toe Web Game with Player Names and Scoreboard

請幫忙製作web-game。使用html + css + javascript (js) ,寫一個可直接在chrome執行的web-game小遊戲。例如tic-tac-toe圖又井字遊戲。在遊戲開始之前，要輸入玩家姓名。而且要顯示玩家勝負的次數，畫面最上方，要有訊息顯示：「創作者:江俊廷」。並且有一個超連結，可以連到文化大學(<https://www.pccu.edu.tw/>)

我會幫你製作一個井字遊戲(Tic-tac-toe)的網頁遊戲，包含您要求的所有功能。

井字遊戲
Click to open website

這個井字遊戲包含以下功能：

1. 頁面頂部顯示創作者名字：江俊廷
2. 文化大學的超連結
3. 玩家名稱輸入功能
4. 完整的遊戲功能：
 - 兩位玩家輪流下棋
 - 自動判斷勝負
 - 顯示當前玩家
 - 記錄並顯示雙方勝場次數
5. 重新開始功能

井字遊戲

創作者：江俊廷
文化大學

現在輪到：bb (O)

O		
O	X	
X		X

aa: 4 勝 | bb: 1 勝

重新開始

Perplexity 的示例

The screenshot shows the Perplexity AI chat interface. The user's query is: "請問程式語言家教的行情是多少？" (What is the market price of programming tutors?). The AI's response is in Chinese and describes a Tic-Tac-Toe game. It mentions that the game will require the player to input a name and display the win/loss count. It also states that the game will display the creator's name and have a link to PCCU (https://www.pccu.edu.tw/). The AI provides the following HTML, CSS, and JavaScript code for the game:

```
xml
<!DOCTYPE html>
<html lang="zh-Hant">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Tic Tac Toe</title>
  <style>
    body {
```

Gemini 的示例

Gemini ▾

✦ 試用 Gemini Advan



請幫忙製作web-game。使用html + css + javascript (js),寫一個可直接在chrome執行的web-game小遊戲。例如tic-tac-toe圈叉井字遊戲。在遊戲開始之前，要輸入玩家姓名。而且要顯示玩家勝負的次數，畫面最上方，要有訊息顯示：「創作者:江俊廷」。並且有一個超連結，可以到文化大學(https://www.pccu.edu.tw/)。我要一個完整的html檔案，並且把用到的函數完成，而不是只有註解。

顯示草稿 ▾



好的，這是一個結合了之前回答的優點，並且功能更完整的圈叉井字遊戲 HTML 檔案，包含了完整的 HTML、CSS 和 JavaScript 程式碼，可以直接在 Chrome 中執行：

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>圈叉井字遊戲</title>
  <style>
    body {
      text-align: center;
      font-family: Arial, sans-serif;
    }
```

問問 Gemini



Copilot 的示例

**Copilot**

請幫忙製作web-game。使用html + css + javascript (js) ,寫一個可直接在chrome執行的web-game小遊戲。例如 tic-tac-toe 圈叉井字遊戲。在遊戲開始之前，要輸入玩家姓名。而且要顯示玩家勝負的次數，畫面最上方，要有訊息顯示：「創作者:江俊廷」。並且有一個超連結，可以連到文化大學(<https://www.pccu.edu.tw/>)請將完整的html, css,javascript放在一個html檔

新增聊天

請幫忙製作web-game。使用html + css + javascript (js) ,寫一個可直接在chrome執行的web-game小遊戲。例如 tic-tac-toe 圈叉井字遊戲。在遊戲開始之前，要輸入玩家姓名。而且要顯示玩家勝負的次數，畫面最上方，要有訊息顯示：「創作者:江俊廷」。並且有一個超連結，可以連到文化大學(<https://www.pccu.edu.tw/>)請將完整的html, css,javascript放在一個html檔

**Copilot**

當然可以！以下是一個完整的 HTML 檔案，包含 HTML、CSS 和 JavaScript，實現了一個簡單的圈叉井字遊戲 (Tic-Tac-Toe)。這個遊戲會在開始時要求輸入玩家姓名，並顯示玩家的勝負次數。畫面最上方會顯示「創作者: 江俊廷」，並且有一個連結到文化大學的超連結。

HTML

```
<!DOCTYPE html>
<html lang="zh-TW">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Tic-Tac-Toe 圈叉井字遊戲</title>
</head>
<body>
  <div>創作者: 江俊廷</div>
  <div>勝負次數: 0</div>
  <div>請輸入您的姓名: <input type="text"/></div>
  <div><button>開始遊戲</button></div>
  <div><table>
    <tr><td></td><td></td><td></td></tr>
    <tr><td></td><td></td><td></td></tr>
    <tr><td></td><td></td><td></td></tr>
  </table></div>
  <div><button>重置</button></div>
</body>
</html>
```

聊天

請幫忙製作web-game。...

下午9:49

關於視覺化，請協助寫pyth...

10/17

A

牛