程式語言與設計

劉和師 老師 part 2

2025/8/20

資料型態進階

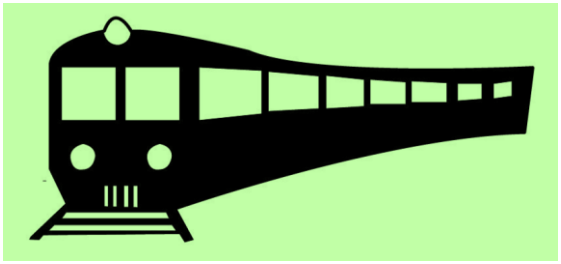
- Python除了基本資料型態，還有以下容器型態：
  - 清單或稱串列(list)(其它語言也稱陣列)
  - 元組或稱序對(tuple)
  - 集合(set)
  - 字典(dict)
- 各型態比較：

資料型態	tuple	list	dict	set
中文名稱	序對	串列	字典	集合
使用符號	()	[]	{}	{}
具順序性	有序	有序	無序	無序
可變 / 不可變	不可	可	可	可
舉例	(1, 2, 3)	[1,2,3]	{'word1':'apple'}	{1, 2, 3}

清單(List)

- 在過去的章節，我們的做法是每次讀入一個資料，就立刻去處理，當讀入的資料一多就很不方便。
- 可以先將讀入的資料儲存至記憶體，以方便後續處理。最常用的就是清單(List)了。

- 目的：
  - 減少變數的個數，使程式的編寫更容易。
  - 清單可配合迴圈來處理大量資料。



清單(List)

- 建立清單的語法：
  - 直接指定清單內容：

```
A = [1, 2, 3, 4, 5]
B = ['a', 12, 3.14, 'Yes']  #可以不同型態
```

  - 建立一個空的清單，之後再加入元素：

```
A = []  #之後用append()方法添加
```

  - 建立一個指定大小及初值的清單：

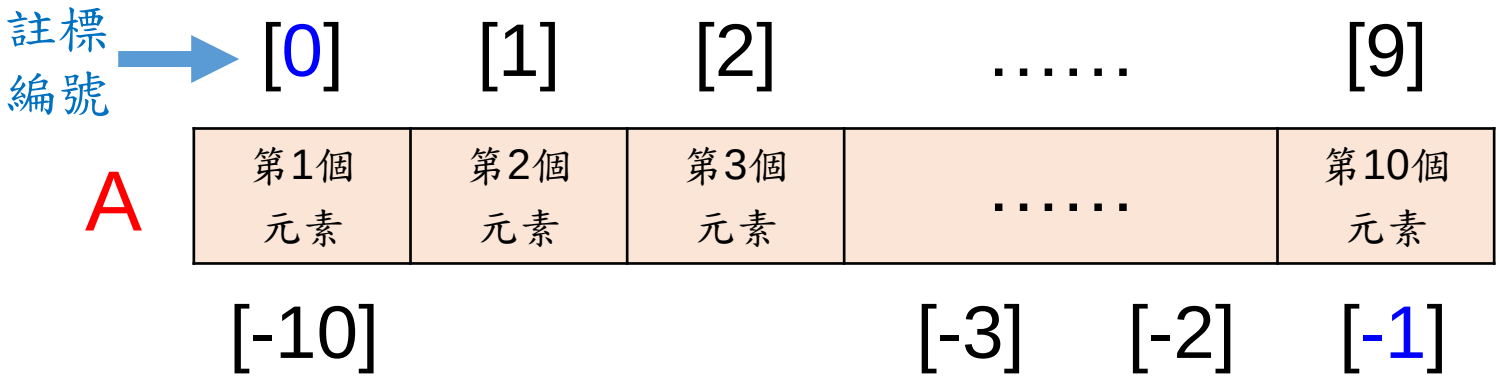
```
A = [0] * 10 或 A = [0 for _ in range(10)]
#設定A清單有10個元素，初值皆為0
```

因為清單建立完此變數就不需要了，所以用_(一個底線)即可。

清單(List)

- 下圖為一維清單的概念，清單名稱為A，每個元素都有一個編號，依編號[n]來存取。
- 註標也稱索引(Index)。

注意：清單註標為正整數,而且是從0開始



注意：使用切片功能時可以從後數起，從-1開始

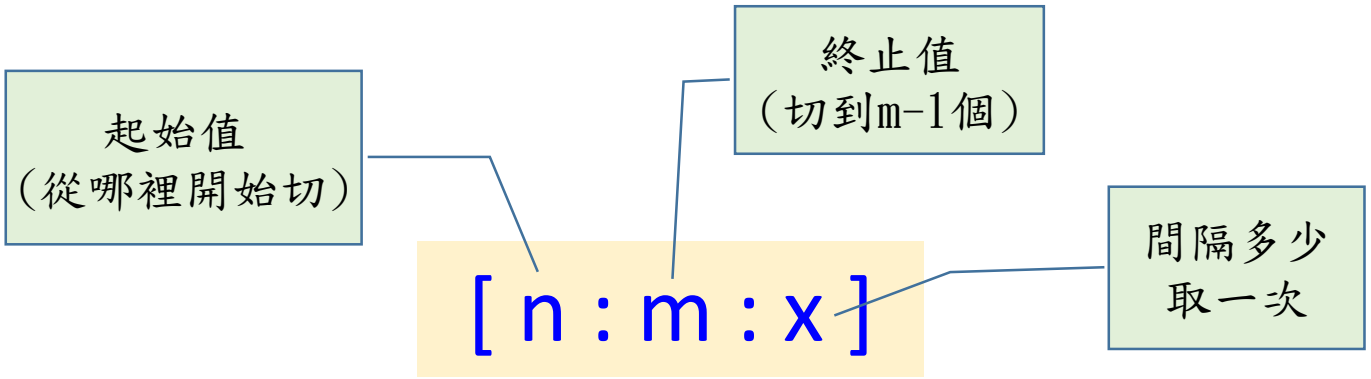
清單(List)

- 清單存取：
- 設有一清單為： `A = [1, 2, 3, 4, 5]`

  - `A[2] = 8` #將8存入A[2]位置內。
    ➔ A成為[1, 2, 8, 4, 5]
  - `A.append(9)` #在A最後面添加一個元素9。
    ➔ A成為[1, 2, 8, 4, 5, 9]
  - `A.remove(8)` #移除元素8。(注意不是[8]這個位置)
    ➔ A成為[1, 2, 4, 5, 9]
  - `x = A[3]` #取出A[3]的內容存入x ➔ x內容為5
  - `y = A[-1]` #取出A[-1]的內容存入y ➔ y內容為9

清單(List)

- 索引切片運算：
  - 只要具有索引特性，基本上都能進行切片運算，如list、字串、tuple等。



[0] [1] [2] [3] [4]

- 設 A = [11, 12, 13, 14, 15]
- 若 X = A[1:4]，則X內容為[12, 13, 14]

#取出的片段

清單(List)

- 清單切片範例：
- 範例：
  - 設：

```
A = [3, 5, 7, 9, 11, 13]
```

```
A[0:3]   得到  [3, 5, 7]
A[3:]    得到  [9, 11, 13]
A[:4]    得到  [3, 5, 7, 9]
A[1:4]   得到  [5, 7, 9]
A[::2]   得到  [3, 7, 11]
A[-5:-1] 得到 [5, 7, 9, 11]
A[::-1] 得到  [13, 11, 9, 7, 5, 3] #反轉清單
```

字串切片功能

- 在Python中字串也被視為一個清單，所以可以使用清單的功能去對字串做動作。

- 例如： `A = "HelloPython"`

- 則儲存情況如下：

註標(正數)	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
A =	H	e	l	l	o	P	y	t	h	o	n
註標(倒數)	[-11]	[-10]	[-9]	[-8]	[-7]	[-6]	[-5]	[-4]	[-3]	[-2]	[-1]

字串切片功能

- 字串切片範例：
- 範例：
  - 設： `A = 'HelloPython'`

```
A[1:4]   得到 'ell'
A[5:]    得到 'Python'
A[:5]    得到 'Hello'
A[7:2:-1] 得到 'tyPol' (倒著取)
A[::3]   得到 'Hlyo'
A[::-1] 得到 'nohtyPolleH' (倒轉整個字串)
```

- 注意：字串可以"切片"取出，但不能更改原內容。

切片功能

- 切片功能除了能取出清單裡的元素，也可以插入元素。
- 範例：
  - 設：

```
A = [3, 5, 7, 9, 11, 13]
```

指定一個值：

A[4] = 99 則A成為 [3, 5, 7, 9, 99, 13]

指定一個範圍：

A[1:3] = 77, 88 ➔ [3, 77, 88, 9, 99, 13]

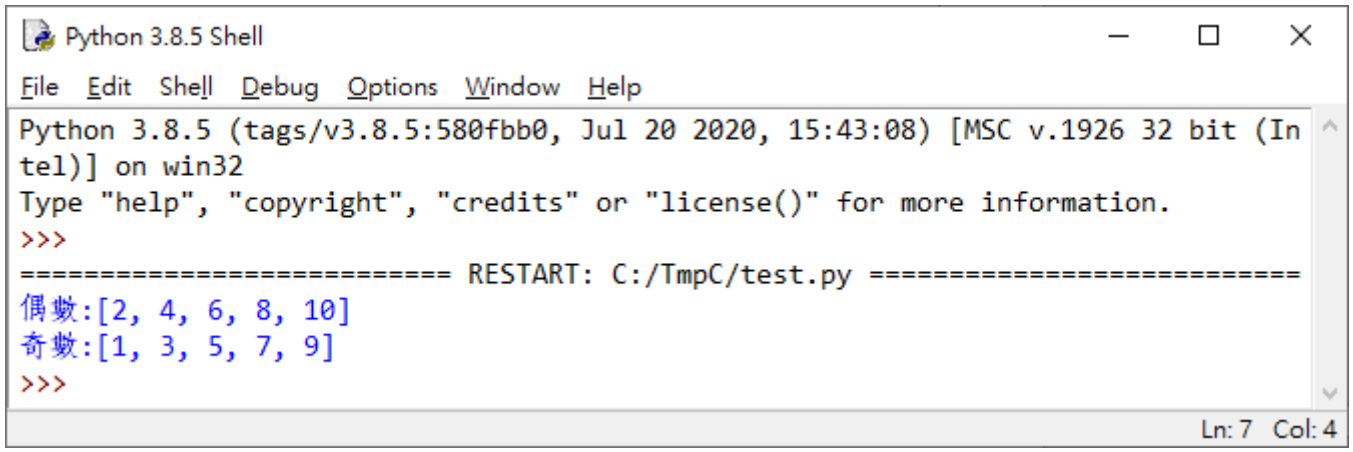
指定範圍並間隔插入：

A[::2] = 1, 10, 20 ➔[1, 77, 10, 9, 20, 13]

清單(List)

- 利用切片功能將奇偶數分離：

```
number = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
even_num = number[1::2]
odd_num = number[0::2]
print(f"偶數:{even_num}\n奇數:{odd_num}")
```

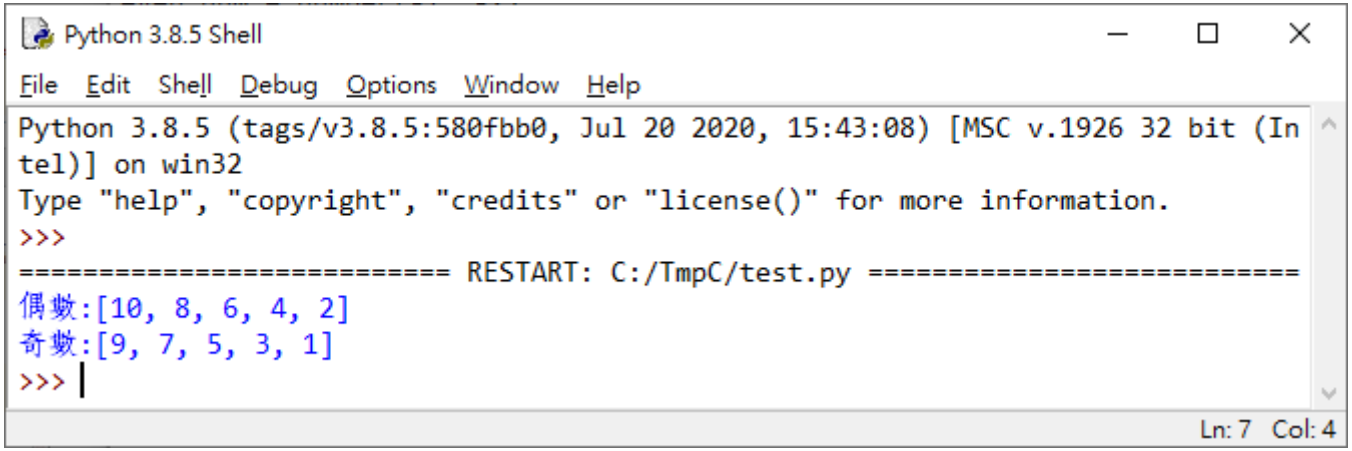


- "切片"是清單很有用的功能。

清單(List)

- 利用倒數切片方式取出奇偶數：

```
number = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
even_num = number[-1::-2]
odd_num = number[-2::-2]
print(f"偶數:{even_num}\n奇數:{odd_num}")
```



- 不但取出奇偶數，還反向排序。

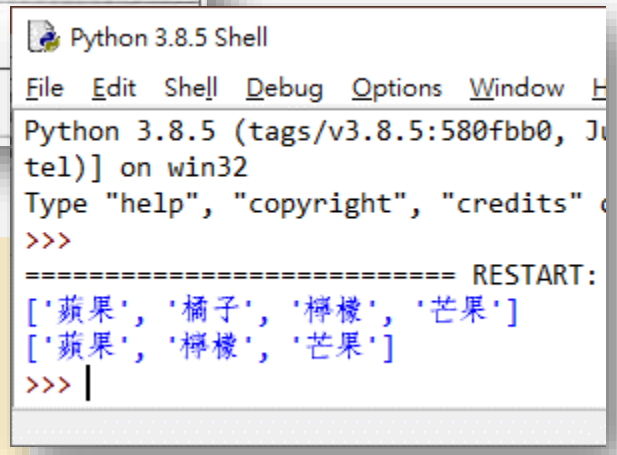
清單(List)

- 除了傳統的用法，list也提供一些常用的方法：

方法	說明
append()	附加新元素
count(x)	計算 list 串列中 x 出現的次數
insert()	插入新元素
pop()	移除元素，預設是最後一個
remove()	移除元素
reverse()	倒轉元素的順序
sort()	排序

- 例：

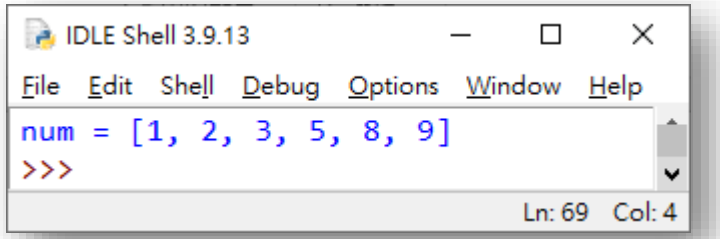
```
fruit = ["蘋果" , "橘子", "檸檬"]
fruit.append("芒果")
print(fruit)
fruit.remove("橘子")
print(fruit)
```



清單(List)

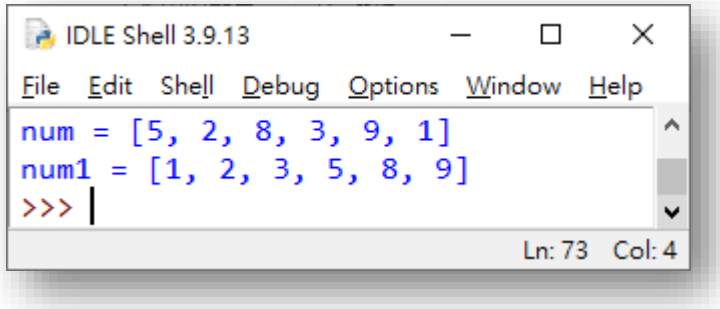
- 排序，直接改變原清單內容：

```
num = [5, 2, 8, 3, 9, 1]
num.sort()
print('num =', num)
```



- 排序，產生新清單，原清單內容不變：

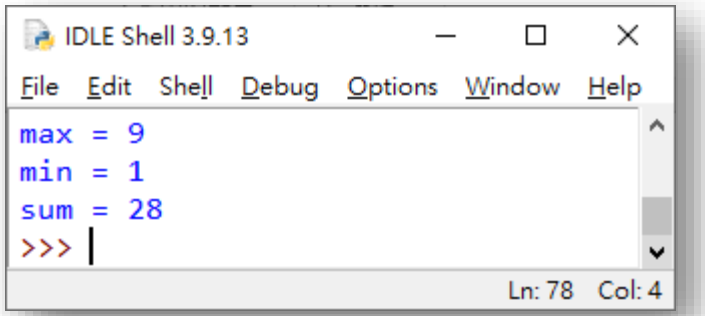
```
num = [5, 2, 8, 3, 9, 1]
num1 = sorted(num)
print('num =', num)
print('num1 =', num1)
```



清單(List)

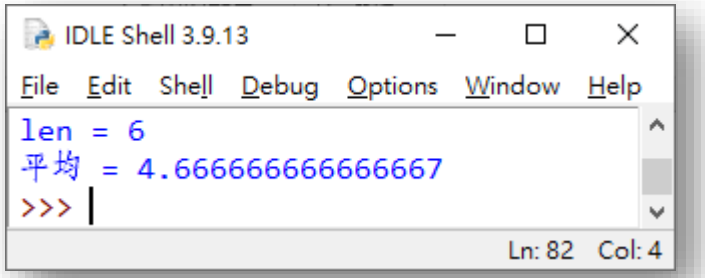
- 取最大值及最小值及總和：

```
num = [5, 2, 8, 3, 9, 1]
print('max =', max(num))
print('min =', min(num))
print('sum =', sum(num))
```



- 取得清單元素個數及平均值：

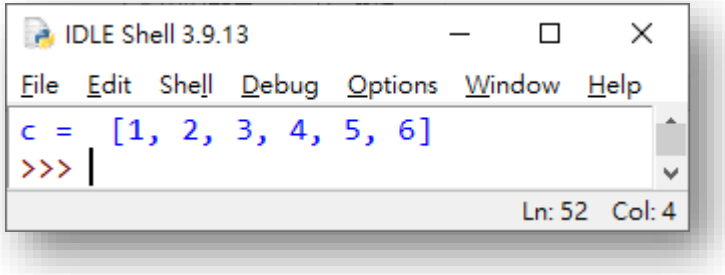
```
num = [5, 2, 8, 3, 9, 1]
print('len =', len(num))
print('平均 =', sum(num)/len(num))
```



清單(List)

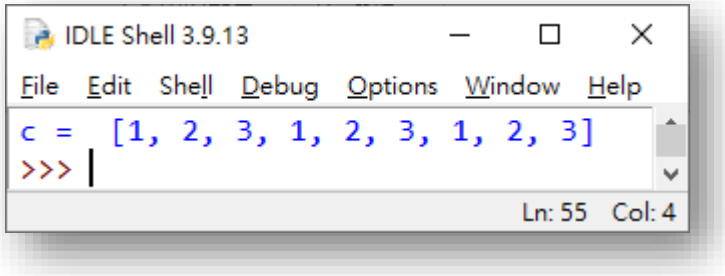
- 清單也可以相加：

```
a = [1, 2, 3]
b = [4, 5, 6]
c = a + b
print('c = ', c)
```



- 清單也可以乘以一個數字：

```
a = [1, 2, 3]
c = a * 3
print('c = ', c)
```



清單(List)

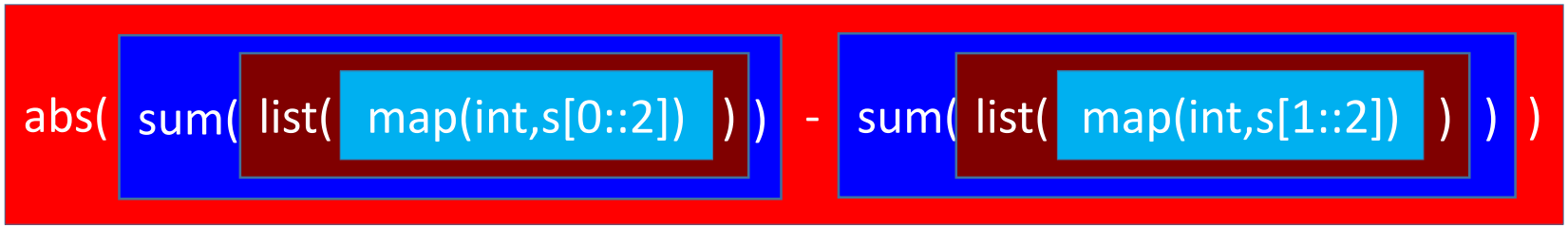
- 練習：
  將一個十進位正整數的奇數位數的和稱為A，偶數位數的和稱為B，則A與B的絕對差值
  |A－B|稱為這個正整數的秘密差。
  例如：263541的
  奇數位數的和A = 6+5+1 = 12，
  偶數位數的和B = 2+3+4 = 9，
  所以263541的秘密差是|12－9| = 3。
  現在給定一個十進位正整數X，請找出X的秘密差。
- (提示：利用切片功能)

練習題

- 6-6.參考程式：

```
s = input("請輸入一個十進位正整數:")

print( abs(sum(list(map(int, s[0::2])))) -  \
           sum(list(map(int, s[1::2]))))  )
```

- 函式層次圖示：

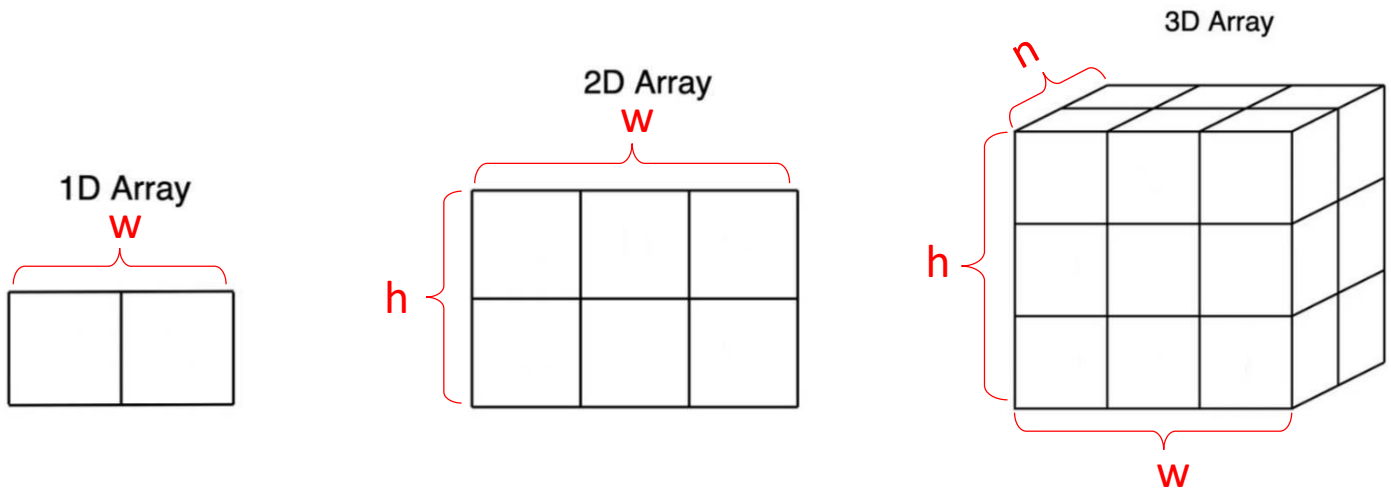


休息一下~

清單(List)

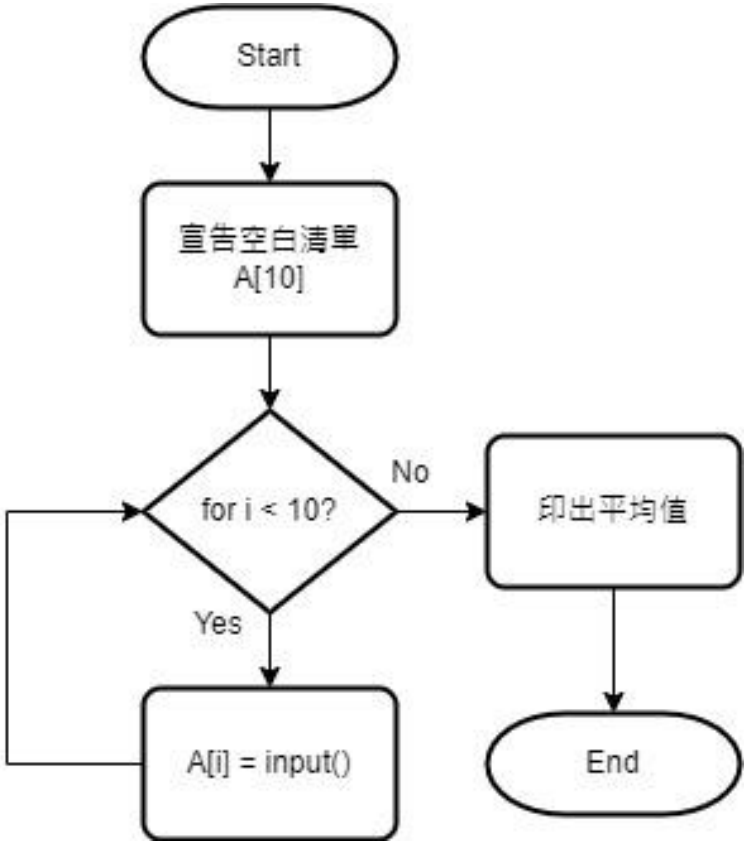
- 清單宣告方式：
- 空的清單：
```
A = []
```
- 一維清單：
```
A = [初值] * w
```
- 二維清單：
```
A = [[初值]*w for _ in range(h)]
```
- 三維清單：

```
A = [[[初值]*w for _ in range(h)] for _ in range(n)]
```



清單(List)

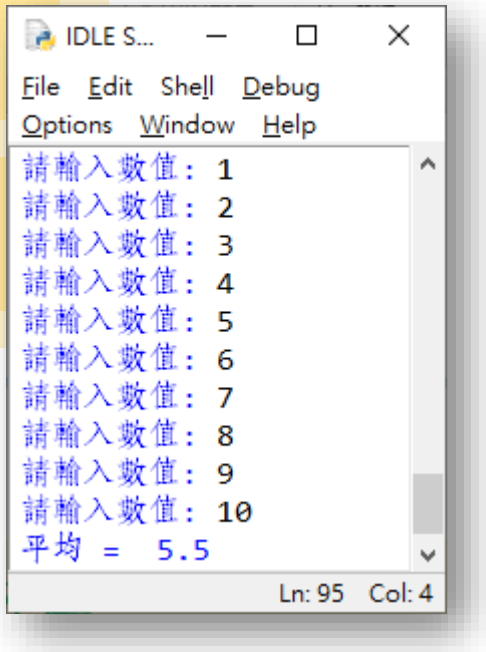
- 利用一維清單計算10個使用者輸入的數字的平均值。
- 流程圖：



清單(List)

- 參考程式一：

```
#宣告一維清單A，有10個元素，皆填入初值0
A = [0] * 10

#將輸入的數值存入清單
for i in range(10):
    A[i] = int(input("請輸入數值: "))

#計算平均，印出結果
print("平均 = ", sum(A) / len(A))
```



清單(List)

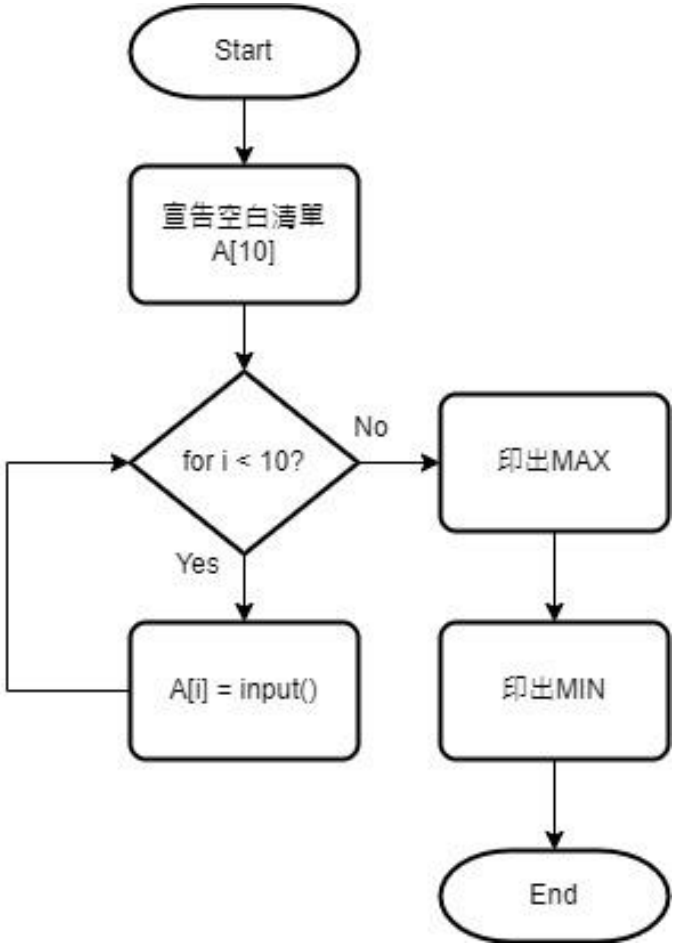
- 參考程式二，你也可以寫得再精簡一點：

```
A = []  #宣告一個空的清單

for i in range(10):
    A.append(int(input("請輸入數值: ")))

print("平均 =", sum(A) / 10)
```

- 之後的練習自己要去精簡程式喔~

清單(List)

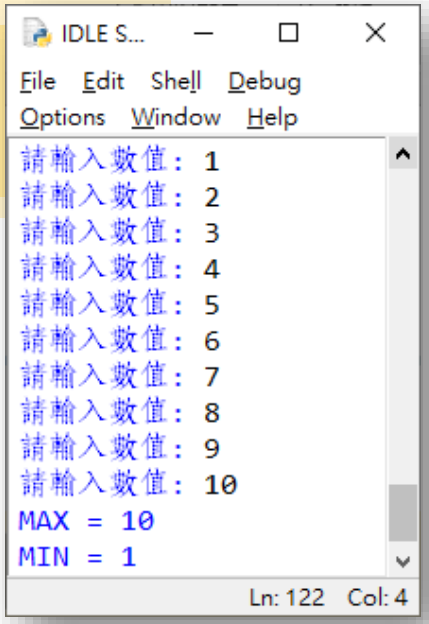
- 利用一維清單求10個數字的最大值及最小值。
- 流程圖：



清單(List)

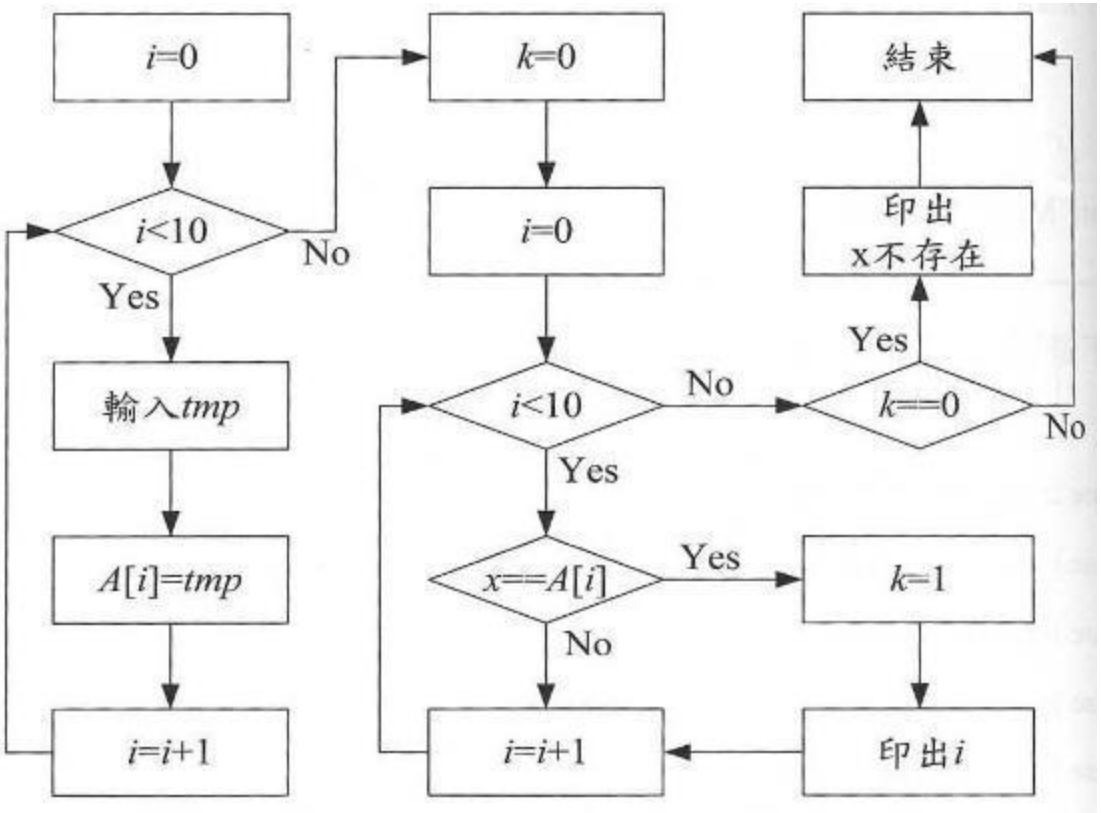
- 參考程式：

```
A = [0] * 10    #宣告清單

for i in range(10):     #輸入10個數值
    A[i] = int(input("請輸入數值: "))

print("MAX =", max(A)) #使用函式即可
print("MIN =", min(A))
```



清單(List)

- 搜尋問題：輸入10個數字至A[]，再輸入x，判斷x是否存在於A清單中，如果存在，輸出所在的註標(索引)值，若不存在，則告知不存在。
- 流程圖：



清單(List)

- 參考程式一：

```
A = [0] * 10

for i in range(10):
    A[i]= int(input("請輸入數值: "))

x = int(input("請輸入你要搜尋的數值: "))
k = 0
for i in range(10):  #依序查找
    if x == A[i]:
        k = 1
        print(x, "在索引位置", i)

if k == 0:
    print(x, "不存在")
```

清單(List)

- 參考程式二，使用index()：
  - index()會傳回特定元素內容第一次出現的索引值，若搜尋值不存在，會傳回錯誤訊息。

```
A = [0] * 10

for i in range(10):
    A[i]= int(input("請輸入數值: "))

x = int(input("請輸入你要搜尋的數值: "))

#先判斷x是否存在於A中再使用index()，以避免錯誤訊息。
if x in A:
    print(f"{x}在索引值{A.index(x)}的位置")
else:
    print(f"{x}不存在")
```

使用try敘述

- 使用try...except...敘述：
  - 在程式中若擔心或預期可能發生錯誤而導致程式中斷的話(例如除以0或資料為NULL等)，可以使用try敘述，請程式"試"一下，沒問題就繼續往下執行，出現錯誤就去執行except之後的敘述，以避免程式突兀的中斷。
- 語法：

```
try:
        :
    程式區塊
        :
except 錯誤資訊(可省略):
        :
    發生錯誤時要執行的敘述
        :
```

程式區塊有任何錯誤發生時，跳到except執行

安全執行完畢，離開try敘述

繼續except之後的指令

使用try敘述

- 常見的幾種錯誤資訊：

錯誤資訊	說明
NameError	使用沒有被定義的對象
IndexError	索引值超過了序列的大小
TypeError	數據類型（type）錯誤
SyntaxError	Python 語法規則錯誤
ValueError	傳入值錯誤
KeyboardInterrupt	當程式被手動強制中止
AssertionError	程式 asset 後面的條件不成立
KeyError	鍵發生錯誤
ZeroDivisionError	除以 0
AttributeError	使用不存在的屬性
IndentationError	Python 語法錯誤（沒有對齊）
IOError	Input/output異常
UnboundLocalError	區域變數和全域變數發生重複或錯誤

例：偵測除以0的錯誤是否發生

```
  :
try:
      ans = x / y
      print(ans)
except ZeroDivisionError:
      #若y為0會觸發此錯誤
      #而跳至此執行
      print('除數不得為0')
  :
```

註：可以有很多個except敘述，以抓出各種不同的錯誤。

使用try敘述

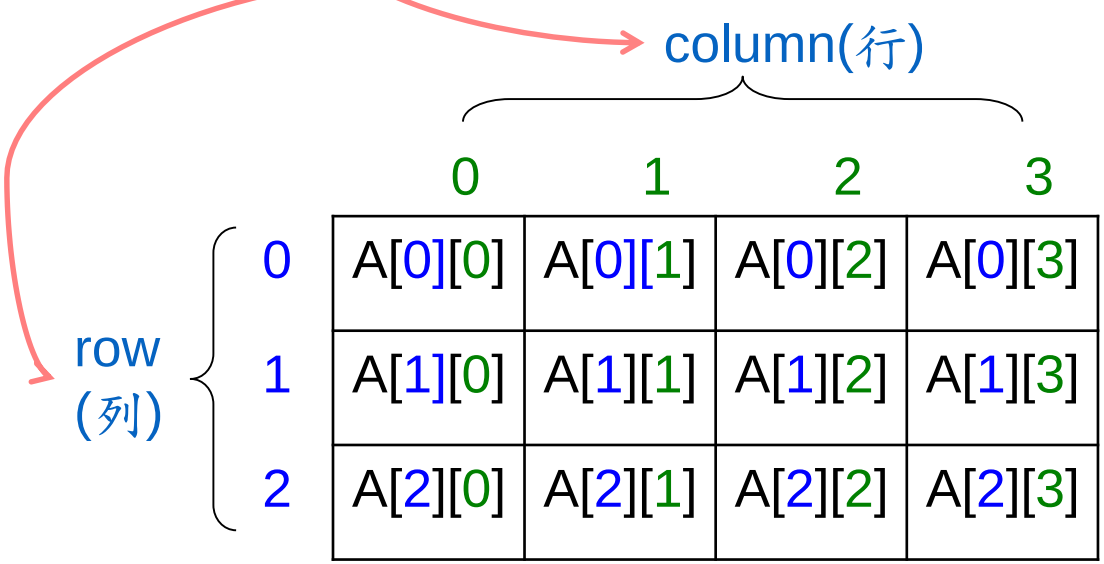
- 參考程式三，使用try...except...：

```
A = [0] * 10

for i in range(10):
    A[i]= int(input("請輸入數值: "))

x = int(input("請輸入你要搜尋的數值: "))

#當A.index(x)找不到而傳回錯誤訊息時，就會跳去執行except
try:
    print(f"{x}在索引值{A.index(x)}的位置")
except:
    print(f"{x}不存在")
```

清單(List)

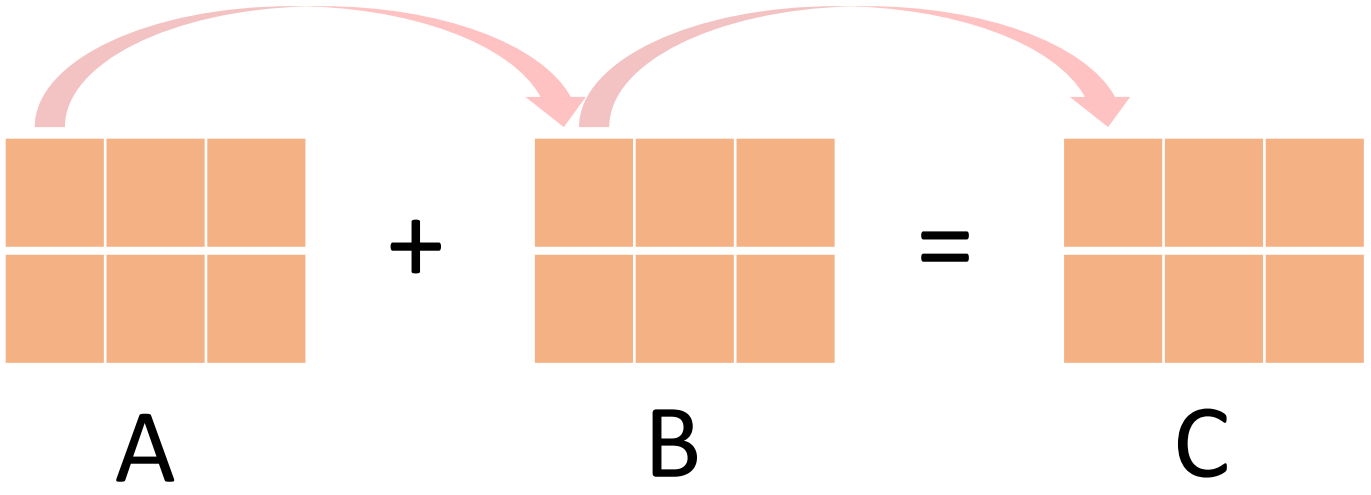
- 二維清單：使用兩個註標來宣告並使用清單。
- 宣告並指定初值：

```
A = [[0] * 4  for _ in range(3)]
```



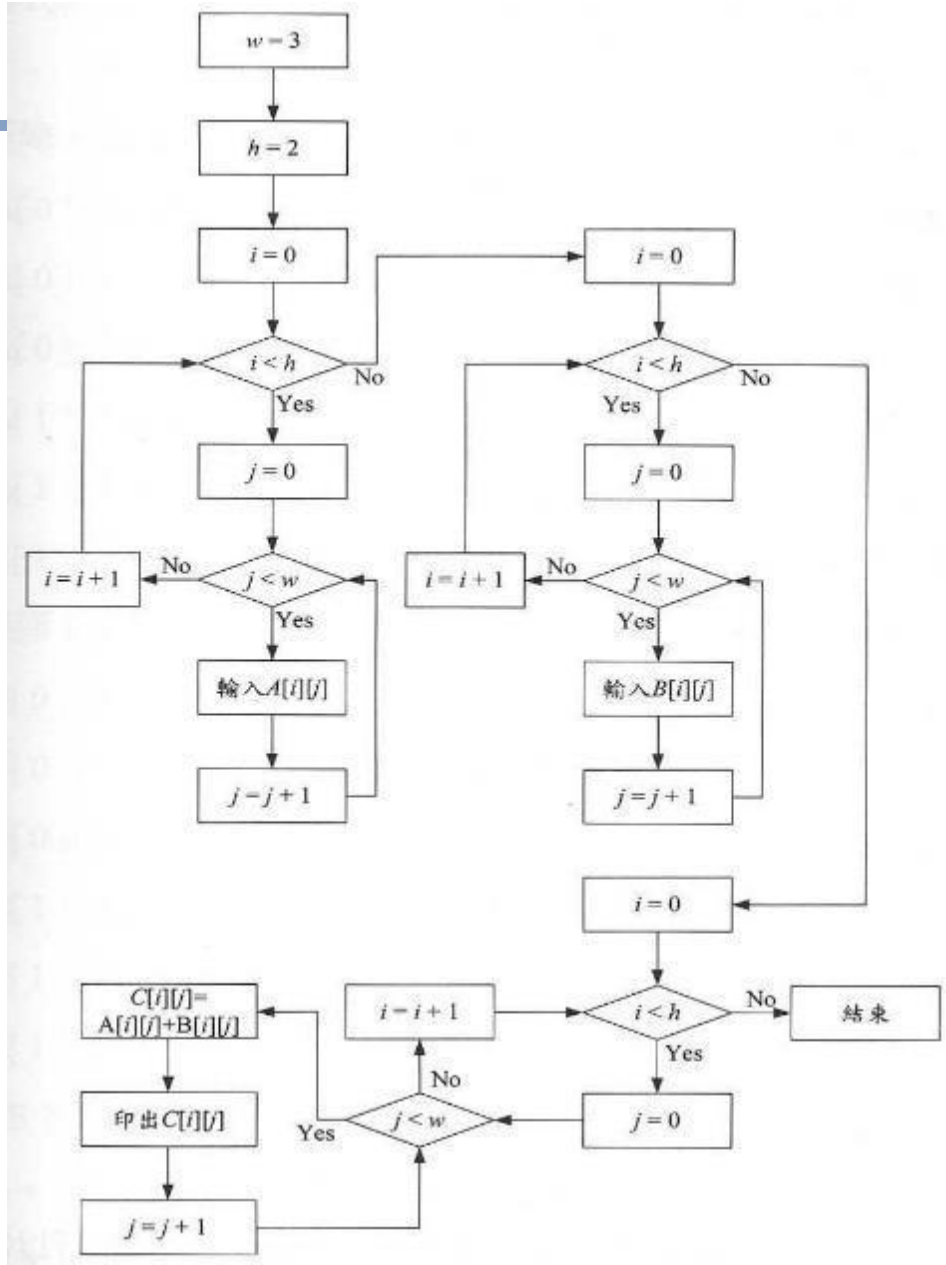
清單(List)

- 兩個二維矩陣相加。以清單模擬矩陣，求兩個2乘3的二維矩陣相加之結果，第一個及第二個矩陣分別以A及B表示，相加之結果存入C矩陣，最後將C矩陣內容顯示出來。
- 圖示：



清單(List)

- 流程圖：


w = 3
h = 2
i = 0
i < h
No
Yes
j = 0
j < w
No
Yes
輸入A[i][j]
j = j + 1
i = i + 1
i = 0
i < h
No
Yes
j = 0
j < w
No
Yes
輸入B[i][j]
j = j + 1
i = i + 1
i = 0
i < h
No
Yes
結束
j = 0
j < w
No
Yes
C[i][j]=
A[i][j]+B[i][j]
印出C[i][j]
j = j + 1
i = i + 1

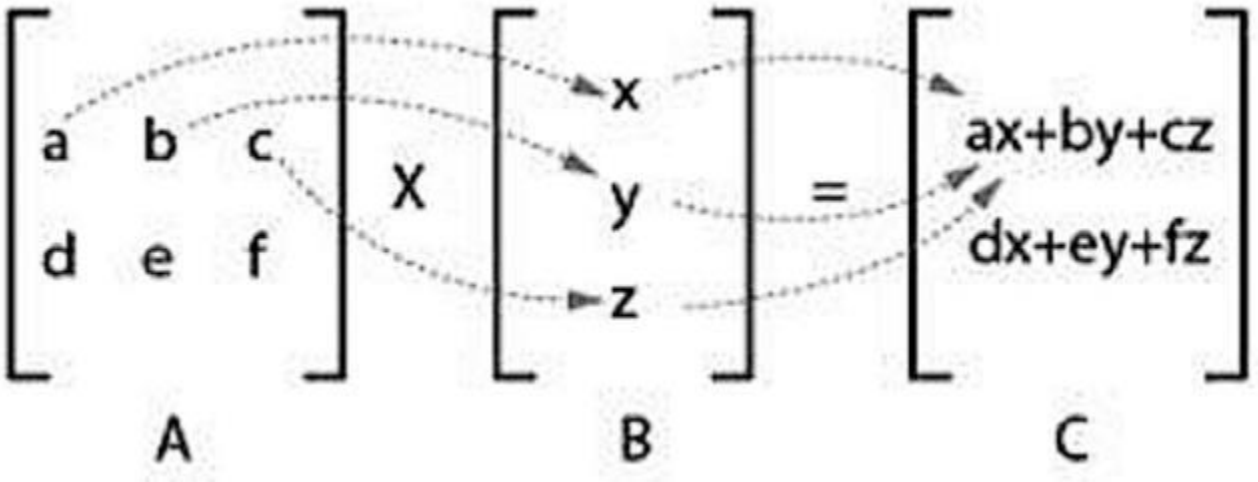
清單(List)

- 參考程式：

```
w = 3
h = 2
#宣告A是一個h*w = 2*3的清單
A = [[0]*w for _ in range(h)]
print("請輸入矩陣A的值...")
for i in range(h):
    for j in range(w):
        print(f"請輸入A[{i}][{j}]的值: ", end="")
        A[i][j] = int(input())
#宣告B是一個h*w = 2*3的清單
B = [[0]*w for _ in range(h)]
print("請輸入矩陣B的值...")
for i in range(h):
    for j in range(w):
        print(f"請輸入B[{i}][{j}]的值: ", end="")
        B[i][j] = int(input())
print("矩陣A、B相加結果儲存於矩陣C中: ")
#宣告C是一個h*w = 2*3的清單
C = [[0]*w for _ in range(h)]
for i in range(h):
    for j in range(w):
        #計算A[i][j]+B[i][j]，將結果存入C[i][j]並印出
        C[i][j] = A[i][j] + B[i][j]
        print(f"C[{i}][{j}] =", C[i][j])
```

清單(List)

- 兩個矩陣相乘。求2乘3的A矩陣乘以3乘1的B矩陣，結果存入2乘1的C矩陣，最後將C矩陣的內容顯示出來。
- 圖示：



清單(List)

- 參考程式：

```
#ha、wa代表清單A的行列數目
ha = 2
wa = 3
#宣告A是一個ha*wa = 2*3的清單
A = [[0]*wa for _ in range(ha)]
print("請輸入矩陣A的值...")
for i in range(ha):
    for j in range(wa):
        print("請輸入A[",i,"][",j,"]的值: ",end=" ")
        A[i][j] = int(input())
#hb、wb代表清單B的行列數目
hb = 3
wb = 1
#宣告B是一個hb*wb = 3*1的清單
B = [[0]*wb for _ in range(hb)]
print("請輸入矩陣B的值...")
for i in range(hb):
    for j in range(wb):
        print("請輸入B[",i,"][",j,"]的值: ",end=" ")
        B[i][j] = int(input())
```

清單(List)

- 參考程式(續)：

```
print("矩陣A、B相乘結果儲存於矩陣C中： ")

#宣告C是一個ha*wb = 2*1的清單
C = [[0]*wb for _ in range(ha)]
for i in range(ha):
    for j in range(wb):
        C[i][j] = 0
        for k in range(wa):
            #將A[i][k]*B[k][j]的結果累加入C[i][j]
            C[i][j] = C[i][j] + A[i][k] * B[k][j]
            #最後印出C[i][j]
        print("C[",i,"][",j,"] = ",C[i][j])
```

練習題

- 6-1.寫一程式，將10個數字讀入A清單，然後逐一檢查此清單，如A[i]>5，則令A[i]=A[i]-5，否則A[i]=A[i]+5。

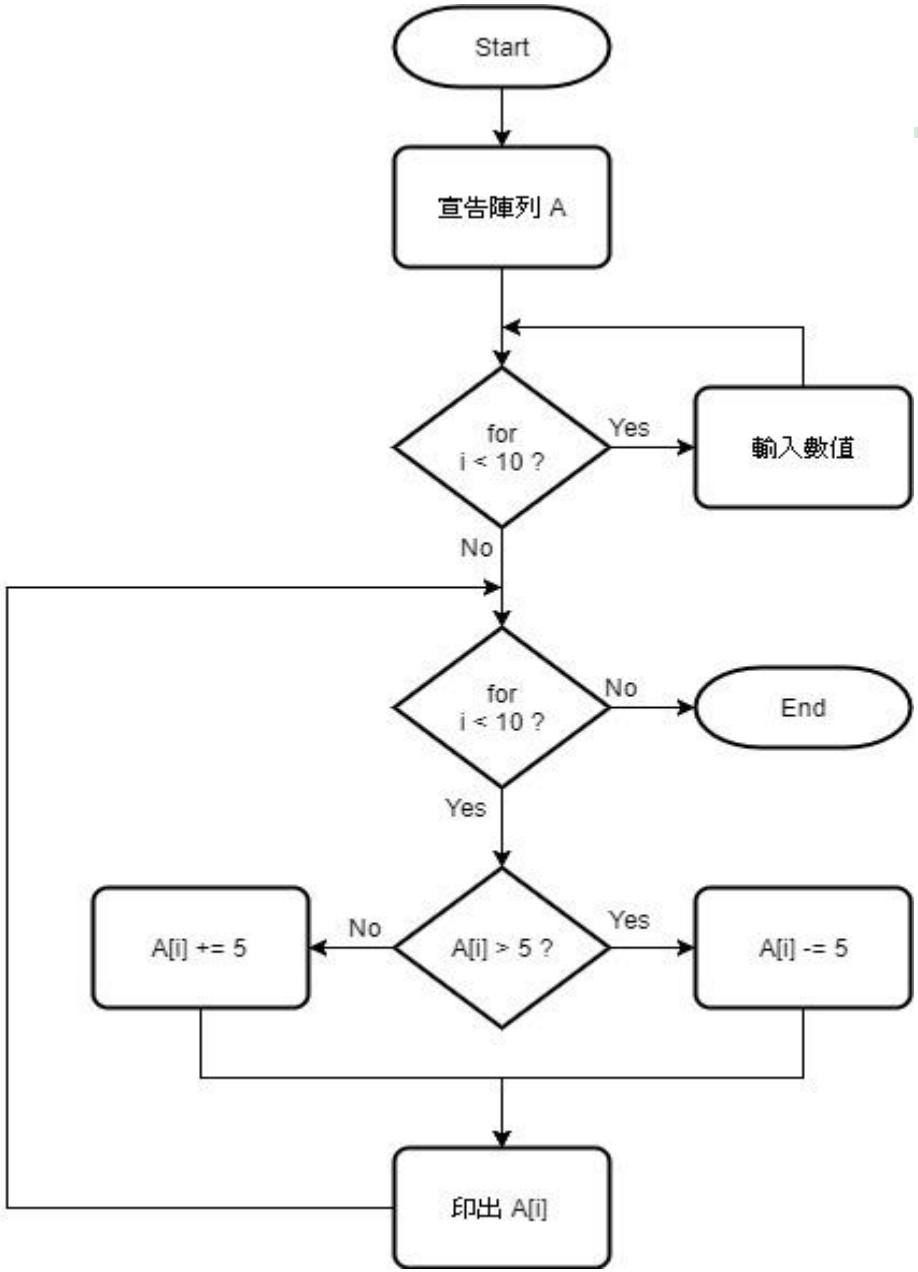
- 6-2.寫一程式，將10個數字讀入A清單，對每一個數字，令A[i]=A[i]+i。

- 6-3.寫一程式，將10個數字讀入A清單，並建立一個B清單，如A[i]≥0，令B[i]=1，否則令B[i]=0。

- 6-4.寫一程式，將15個數字存入3×5的二維清單A中，求每一行及每一列數字的和。

練習題

- 6-1流程圖：


Start
宣告陣列 A
for
i < 10 ?
Yes
輸入數值
No
for
i < 10 ?
No
End
Yes
A[i] += 5
No
A[i] > 5 ?
Yes
A[i] -= 5
印出 A[i]

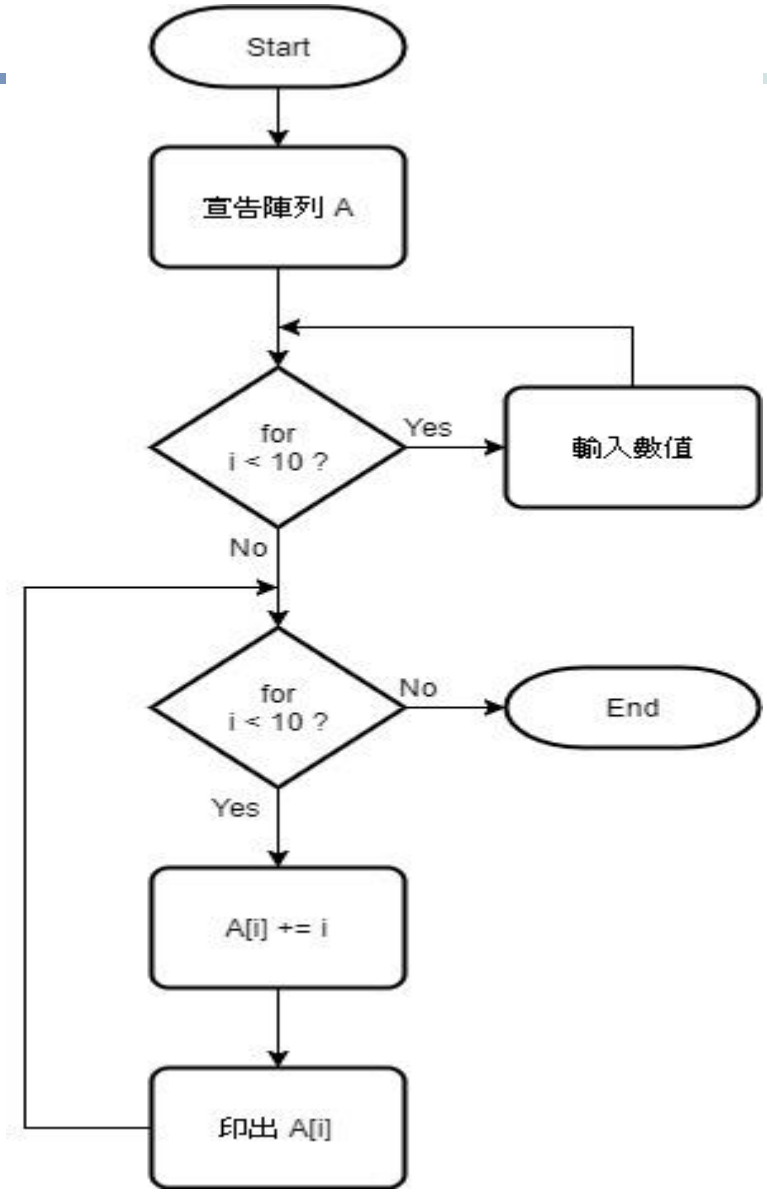
練習題

- 6-1.參考程式：

```
A = [0] * 10

for i in range(10):
    A[i] = int(input("請輸入數值: "))

for i in range(10):
    if A[i] > 5:
        A[i] -= 5
    else:
        A[i] += 5
    print("A[",i,"] = ", A[i])
```

練習題

- 6-2流程圖：



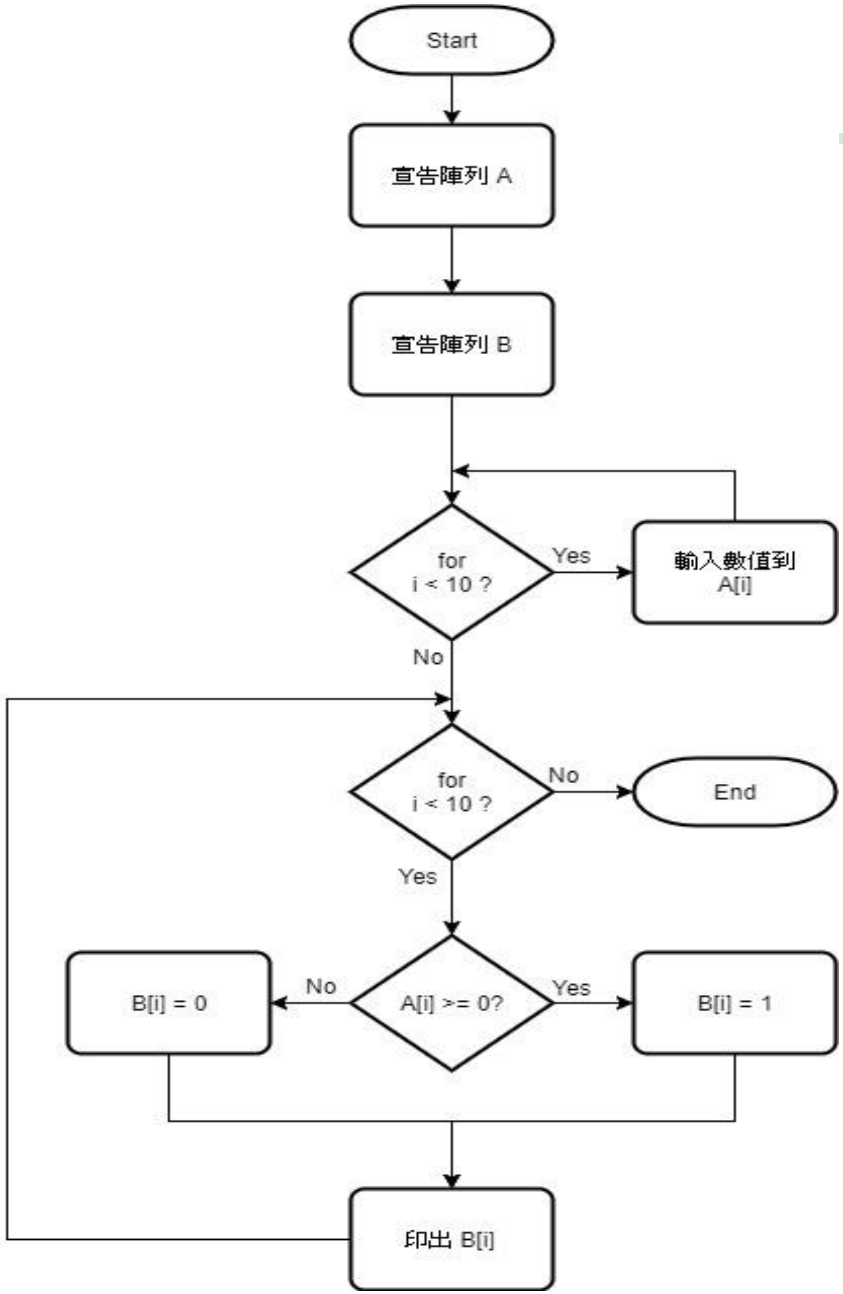
練習題

- 6-2.參考程式：

```
A = [0] * 10

for i in range(10):
    A[i] = int(input("請輸入數值: "))

for i in range(10):
    A[i] += i
    print("A[",i,"] = ", A[i])
```

練習題

- 6-3流程圖：


Start
宣告陣列 A
宣告陣列 B
for
i < 10 ?
Yes
輸入數值到
A[i]
No
for
i < 10 ?
No
End
Yes
B[i] = 0
No
A[i] >= 0?
Yes
B[i] = 1
印出 B[i]

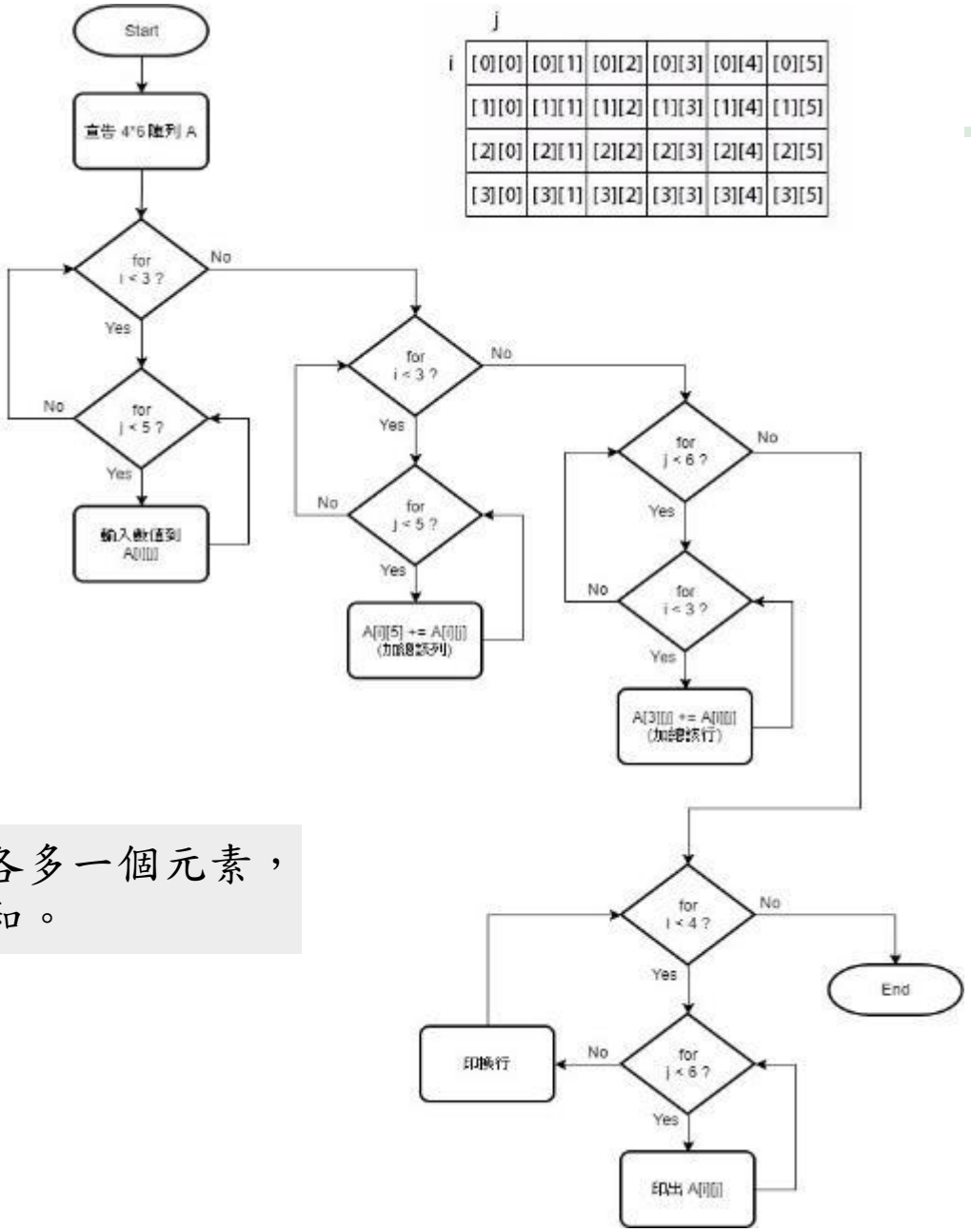
練習題

- 6-3.參考程式：

```
A = [0] * 10
B = [0] * 10

for i in range(10):
    A[i] = int(input("請輸入數值: "))

for i in range(10):
    B[i] = 1 if A[i] >= 0 else 0
    print("B[",i,"] = ", B[i])
```

練習題

- 6-4流程圖：



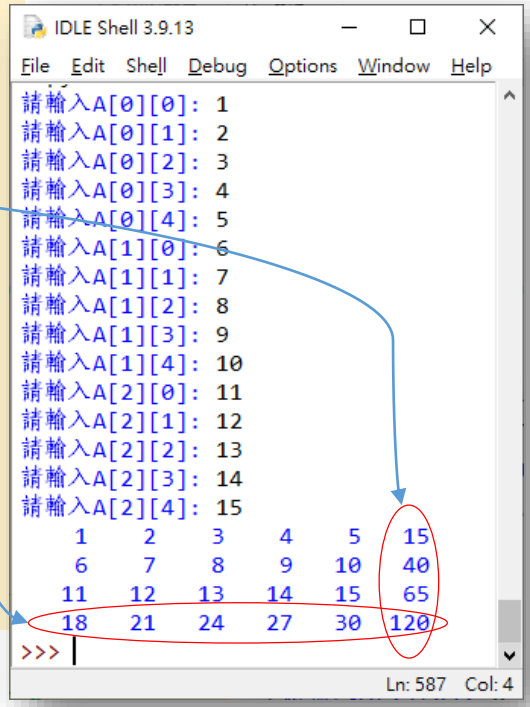
i \ j	[0]	[1]	[2]	[3]	[4]	[5]
[0]	[0][0]	[0][1]	[0][2]	[0][3]	[0][4]	[0][5]
[1]	[1][0]	[1][1]	[1][2]	[1][3]	[1][4]	[1][5]
[2]	[2][0]	[2][1]	[2][2]	[2][3]	[2][4]	[2][5]
[3]	[3][0]	[3][1]	[3][2]	[3][3]	[3][4]	[3][5]

清單宣告時，行列各多一個元素，用來儲存行和列的和。

練習題

- 6-4.參考程式：

```
A = [[0]*6 for _ in range(4)]
#請求輸入
for i in range(3):
    for j in range(5):
        print(f"請輸入A[{i}][{j}]: ", end='')
        A[i][j] = int(input())
#計算列，將列的和存入每列最後一個元素
for i in range(3):
    A[i][5] = sum(A[i][:])
#計算行，將行的和存入每行最後一個元素
for j in range(6):
    A[3][j] = sum([A[0][j], A[1][j], A[2][j]])
#印出矩陣和結果
for i in range(4):
    for j in range(6):
        print("%5d" % A[i][j], end='')
    print()
```



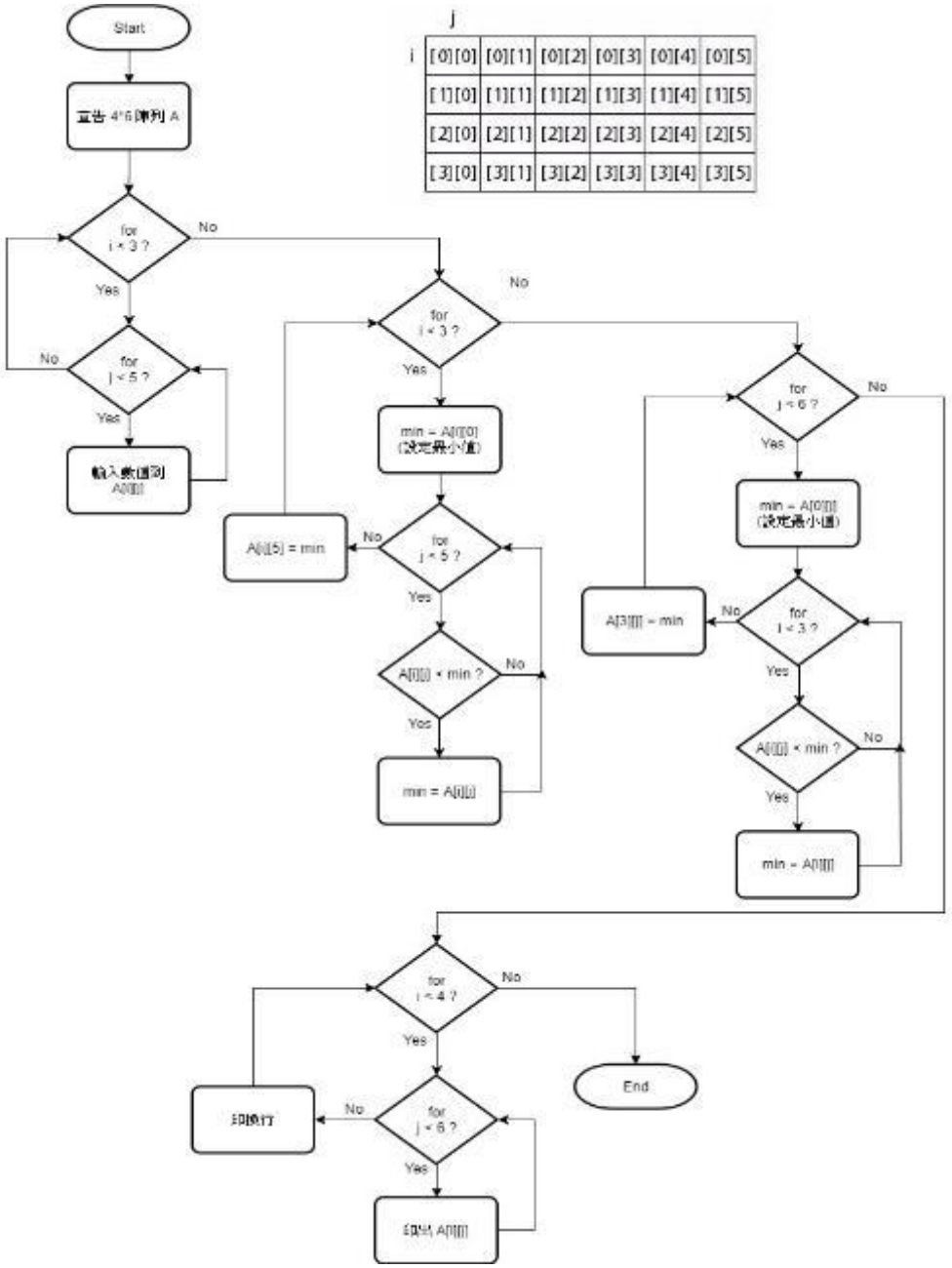
練習題

- 6-5.寫一程式，將15數字存入3×5的二維清單A中，求每一行及每一列數字的最小值。

- 6-6.輸入一個字串，將輸入的字串依相反的字元順序印出來。

練習題

- 6-5流程圖：

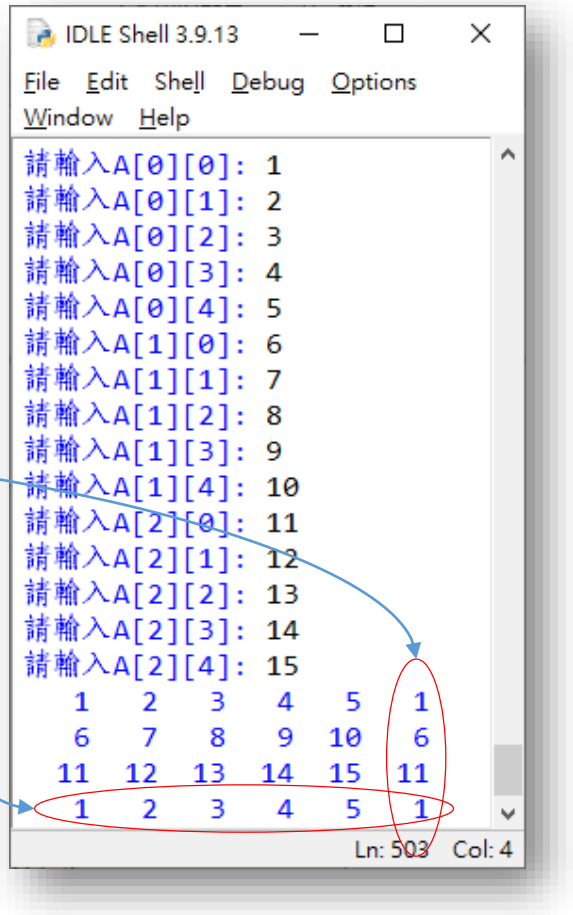


i \ j	[0]	[1]	[2]	[3]	[4]	[5]
[0]	[0][0]	[0][1]	[0][2]	[0][3]	[0][4]	[0][5]
[1]	[1][0]	[1][1]	[1][2]	[1][3]	[1][4]	[1][5]
[2]	[2][0]	[2][1]	[2][2]	[2][3]	[2][4]	[2][5]
[3]	[3][0]	[3][1]	[3][2]	[3][3]	[3][4]	[3][5]

練習題

- 6-5.參考程式：

```
A = [[0]*6 for _ in range(4)]
#請求輸入
for i in range(3):
    for j in range(5):
        print(f"請輸入A[{i}][{j}]: ", end='')
        A[i][j] = int(input())
#尋找列，將列的最小值存入每列最後一個元素
for i in range(3):
    A[i][5] = min(A[i][:5])
#尋找行，將行的最小值存入每行最後一個元素
for j in range(6):
    A[3][j] = min(A[0][j], A[1][j], A[2][j])
#印出矩陣和結果
for i in range(4):
    for j in range(6):
        print("%4d" % A[i][j], end='')
    print()
```



練習題

- 6-6.參考程式：

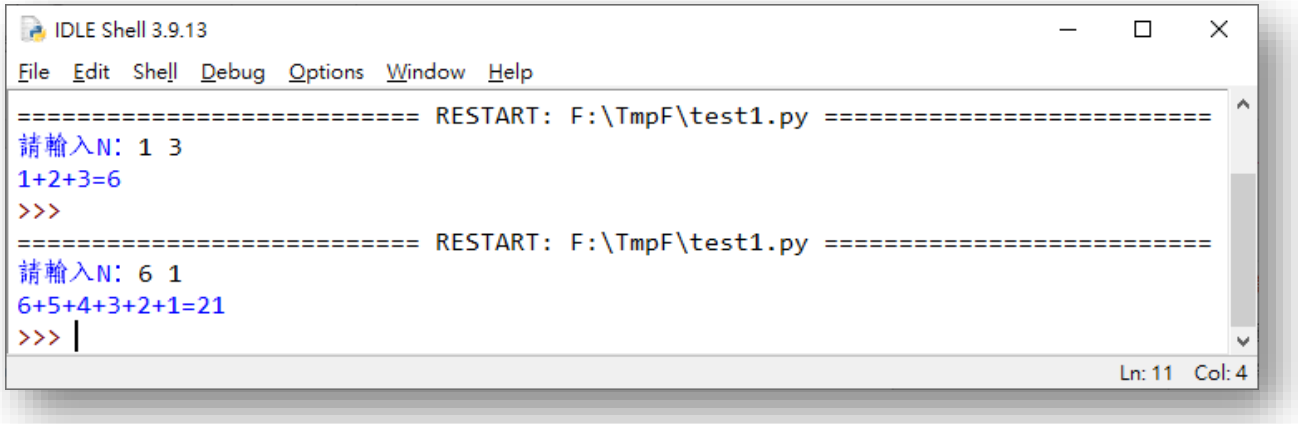
```
s = input("請輸入字串:")

print(s[-1::-1])  #利用切片倒轉功能
```

練習題

- 6-7.給你兩個正整數A、B(A和B不一定誰比較大)，請你把A到B之間所有的整數加起來，並印出它的算式以及結果。

- 例如：輸入 `1 3` ，則輸出 `1+2+3=6`

  輸入 `6 1` ，則輸出 `6+5+4+3+2+1=21`



練習題

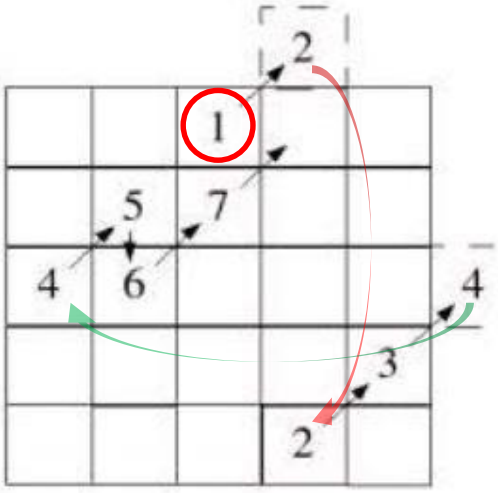
- 6-7.參考程式：

```
#輸入兩數字
A, B = map(int, input("請輸入N：").split())

#將A、B調整成A比B小，並記錄s
A, B, s = (A, B, 1) if A < B else (B, A, -1)

#產生範圍內正數的數字序列清單
N = [i for i in range(A, B+1)]

#將清單依s是正或負來正取或倒取出來，加上總和
print('+'.join(map(str, N[::s])) + '=' + str(sum(N)))
```

綜合練習

- 魔術方陣
  - 有一n*n的方陣，其中n為奇數，請在這個方陣中將1到n^2的整數填入其中，使其各行、各列及對角線之和皆相等。規則很簡單，最上面一行的中間為1，之後依下列規則填入：

1. 每次往右上方那一格填入加1的數字，直到n^2。
2. 若該位置已有值，則填在原位置的下方。
3. 若超出方陣，則往最下面或最左邊填。
4. 最右上角那一格要接著填在原位置下方。



綜合練習

- 參考程式：

```
n = int(input("請輸入方陣邊長(奇數，須>=3):"))

if n < 3 or n % 2 == 0:
    print("輸入不符要求")
else:
    #建立空清單及初值
    A = [[0]*n for _ in range(n)]
    #起始位置及值
    x = 0
    y = n // 2
    A[x][y] = 1
    e = n*n  #結尾的數字
```

綜合練習

- 參考程式(續)：

```
#填入方陣
for i in range(2, e+1):
    x = x - 1
    y = y + 1
    if x<0 and y==n:   #若為右上角
        x = x + 2
        y = y - 1
    elif x < 0 :       #若為上邊界
        x = n - 1
    elif y == n:       #若為右邊界
        y = 0
    if A[x][y] != 0:   #若該位置已有值
        x = x + 2
        y = y - 1
    A[x][y] = i        #設定新位置值
```

綜合練習

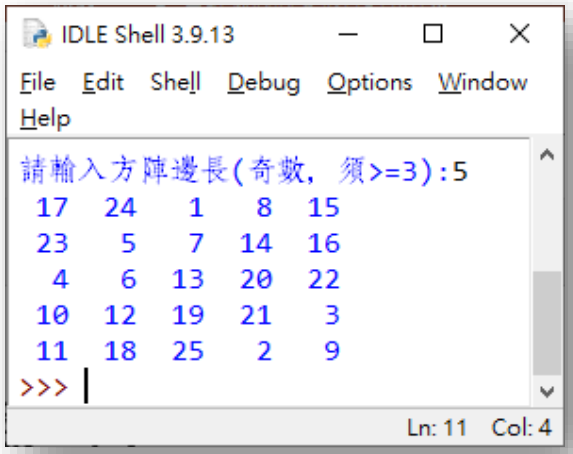
- 參考程式(續)：

```
    #印出結果
    L = len(str(e))
    for i in range(n):
        for j in range(n):
            #為不同大小的方陣預留不同寬度
            if L <= 3:
                print("%3d " % A[i][j], end='')
            else:
                print("%5d " % A[i][j], end='')
        print()
```

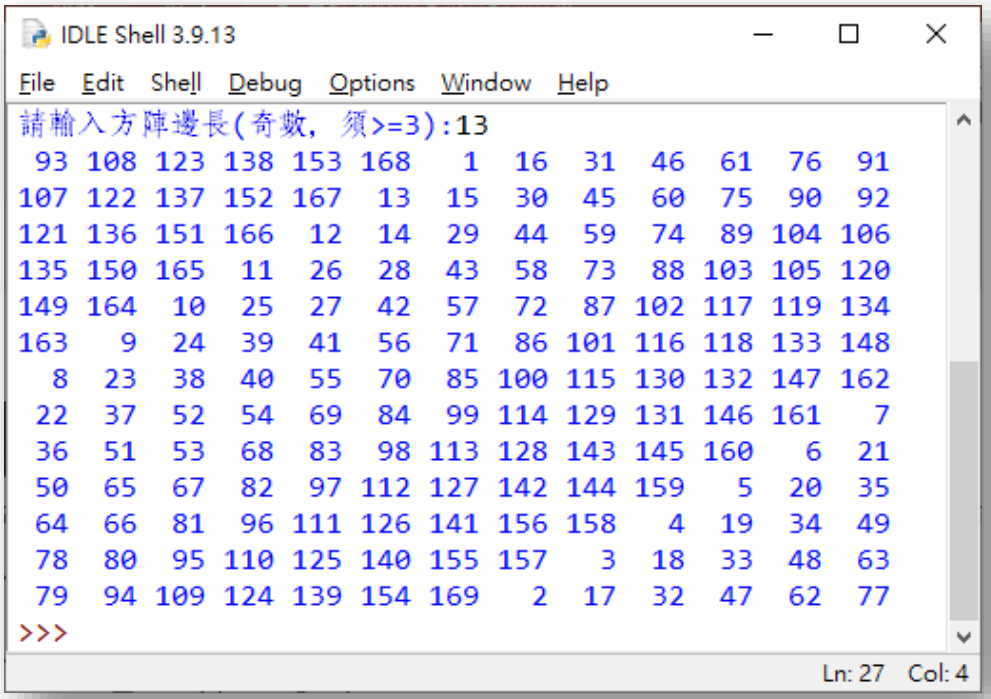
綜合練習

- 執行結果：

5 x 5



13 x 13



休息一下~

元組(Tuple)

- Tuple是有序物件，類似list串列，差別是tuple是不可變物件，一旦建立後，序對中的元素不能任意更改其位置和值。

- tuple由括號( )建立，可以包含不同型態的資料，以逗點分隔。
- Ex： `tuple1 = (1, 'sky', 3.14)`

- 以索引值取出資料。
- Ex： `print( tuple1[1] )` => 取得 'sky'

元組(Tuple)

- 可以將tuple轉成list：
  - 例： list1 = list(tuple1)
  - 之後可以對list1增減元素。但若將list轉成tuple後，該tuple就不能再增減元素。

- Ex：

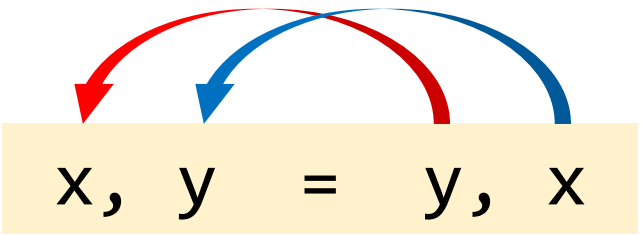
```
tuple1 = (1, 2, 3, 4, 5)
list1 = list(tuple1)
list1.append(6)
print(list1)    #印出 [1,2,3,4,5,6]

tuple2 = (1, 2, 3, 4, 5)
tuple2.append(6)   #錯誤，無法執行
```

元組(Tuple)

- 拆解：
  - 各元素會依序置入變數，此方式List也適用

```
data = ("Orion", "25", "花蓮")

name, age, addr = data
print(name)    #印出Orion
```

- 交換：
  - 交換兩變數，超方便



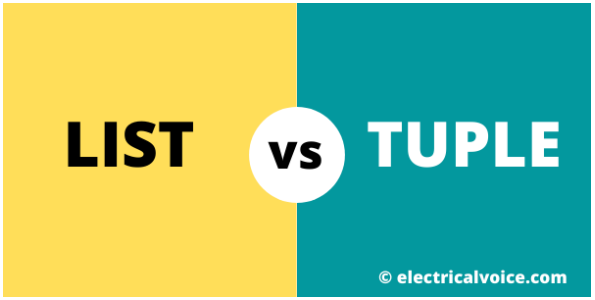
元組(Tuple)

- Tuple的好處：
  - 可以更安全的保護資料：

    程式中可能會碰到有些資料是永遠不會改變的，將它存成元組(tuple)可以安全的被保護。

  - 增加程式執行速度：

    元組(tuple)的結構比清單(list)簡單，占用較少系統資源，程式執行速度較快。



字典(Dict)

- 以大括號{ }來建立，它是使用「鍵」和 「值」的方式儲存資料。
- 「鍵」必須是唯一的，而「值」可以相同。

- 例：

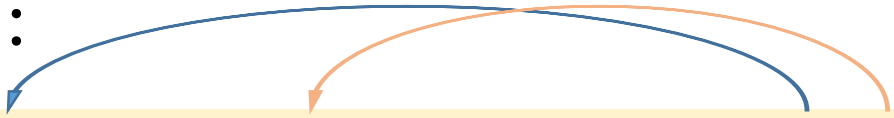
```
d = {'name':'Andy', 'age':18, 'city':'台北'}
    #  鍵   :  值       鍵  :  值     鍵  :  值

print(d['name'])    #印出Andy
print(d['age'])     #印出18
print(d['city'])    #印出台北
```

字典(Dict)

- 修改字典的元素：
- 例：

```
d['name'] = 'Tom'    #原來的Andy變成Tom
```

- 新增元素：
- 例：



```
d['hobby'] = '籃球'    #直接新增「鍵:值」對即可
```

- 刪除元素：
- 例：

```
del d['hobby']  #刪除一個元素

del d    #刪除整個字典
```

字典(Dict)

- 字典常用函式：

方法	說明
clear()	清空 dict 物件
copy()	複製 dict 物件
get()	以 key 來搜尋資料
pop()	移除元素
update()	合併或更新 dict 物件
keys()	取出 key 以 dict_items 物件型態回傳
values()	取出 value 以 dict_items 物件型態回傳

字典(Dict)

- 練習：
  - 公元前50年凱薩發明了凱薩密碼，就是將每個英文字母往後移若干字，形成新的字串，未來就可以依此還原，例如將每個字母往後移三個字，則A對應D、B對應E、...，依此類推。

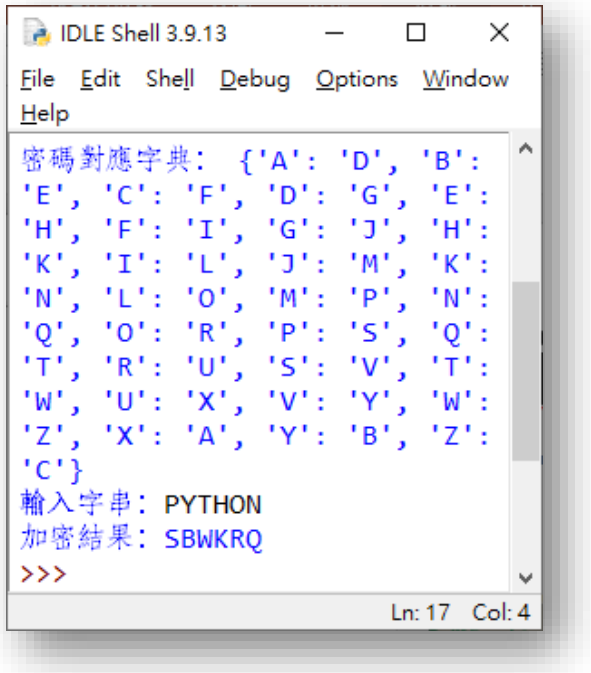
A	B	C	D	...	V	W	X	Y	Z
D	E	F	G	...	Y	Z	A	B	C

  - 例如"PYTHON"會加密成"SBWKRQ"。
  - 寫一程式，輸入原始字串後輸出加密後的字串

字典(Dict)

- 參考程式：

```
#先建立對應表
a = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
b = 'DEFGHIJKLMNOPQRSTUVWXYZABC'
#將a和b打包建立成字典
code = dict(zip(a, b))
print("密碼對應字典：", code)
#輸入要轉成大寫
txt = input("輸入字串：").upper()
txtout = ''
#找出對應的字，加入字串
for i in txt:
    txtout += code[i]
#將字串輸出
print("加密結果：" + txtout)
```



集合(Set)

- 集合(set)與字典(dict)一樣都是把元素放在大括號{ }裏。不過集合只有「鍵」，沒有「值」。
- 集合裏的元素沒有順序之分，而且不可以重複。
- 例：
```
fruitlist = {'Apple', 'Orange', 'Lemon'}
```
- 集合可以做以下運算：

集合運算	範例	說明
聯集 (|)	A|B	存在集合 A 或存在集合 B
交集 (&)	A&B	存在集合 A 也存在集合 B
差集 (-)	A-B	存在集合 A 但不存在集合 B
互斥或 (^)	A^B	排除相同元素

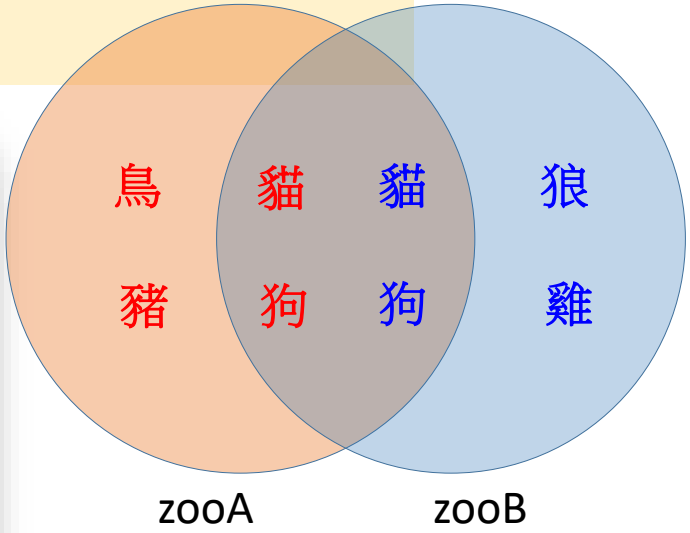
集合(Set)

- Set的聯集(|)、交集(&)、差集(-)與互斥或(^)：
- 例：

```
zooA = {'鳥', '貓', '狗', '豬'}
zooB = {'狼', '貓', '狗', '雞'}

print(zooA | zooB)  #重複的元素只會出現一次
print(zooA & zooB)
print(zooA - zooB)
print(zooA ^ zooB)
```

- 執行：

```
Python 3.8.5 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020,
on win32
Type "help", "copyright", "credits" or "license
>>>
========== RESTART: C:/TmpC/te
{'狼', '雞', '狗', '鳥', '貓', '豬'}
{'貓', '狗'}
{'鳥', '豬'}
{'狼', '雞', '鳥', '豬'}
>>>
```



集合(Set)

- Set常用的分法：

方法	說明
add()	新增元素
remove()	移除元素
update()	合併或更新 set 物件
clear()	清空 set 集合

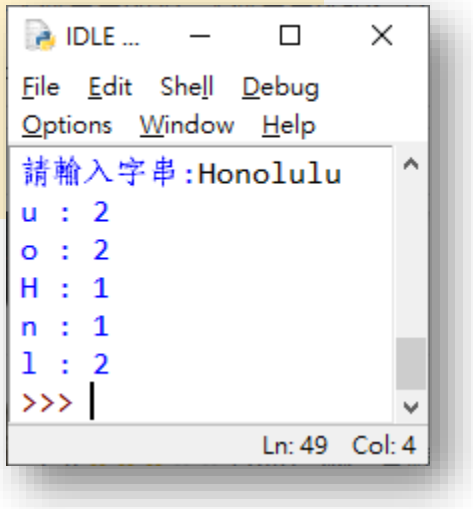
- 例：

```
zooA.add('魚')          #增加了'魚'
zooB.remove('狗')       #移除了'狗'
```

集合(Set)

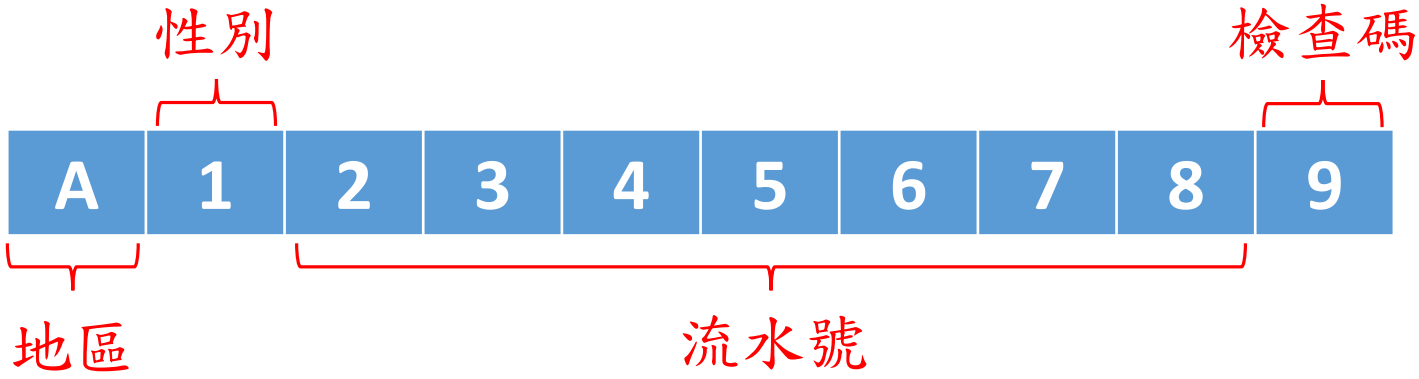
- 練習：計算字串中每個字母出現的次數。
  - 利用set()將重複的字去除，然後計算每個字母在原字串中出現的次數，再用dict()做成字典。

```
s = input('請輸入字串:')

n = dict([x, s.count(x)] for x in set(s))

#將字典印出來，注意字典是無序的
for i, v in n.items():
    print(i, ':', v)
```



綜合練習

- 身份證字號檢查
  - 身分證字號的每個字元代表的意義如下，例如：



  - (1)第一個字元代表地區。
  - (2)第二個字元代表性別，1代表男性，2代表女性
  - (3)第三個字元到第九個字元為流水號碼。
  - (4)第十個字元為檢查號碼。

綜合練習

- 身份證字號檢查
  - 第一個字元代表地區，轉換方式為：A轉換成1,0兩個數值，B轉換成1,1……，依此類推。

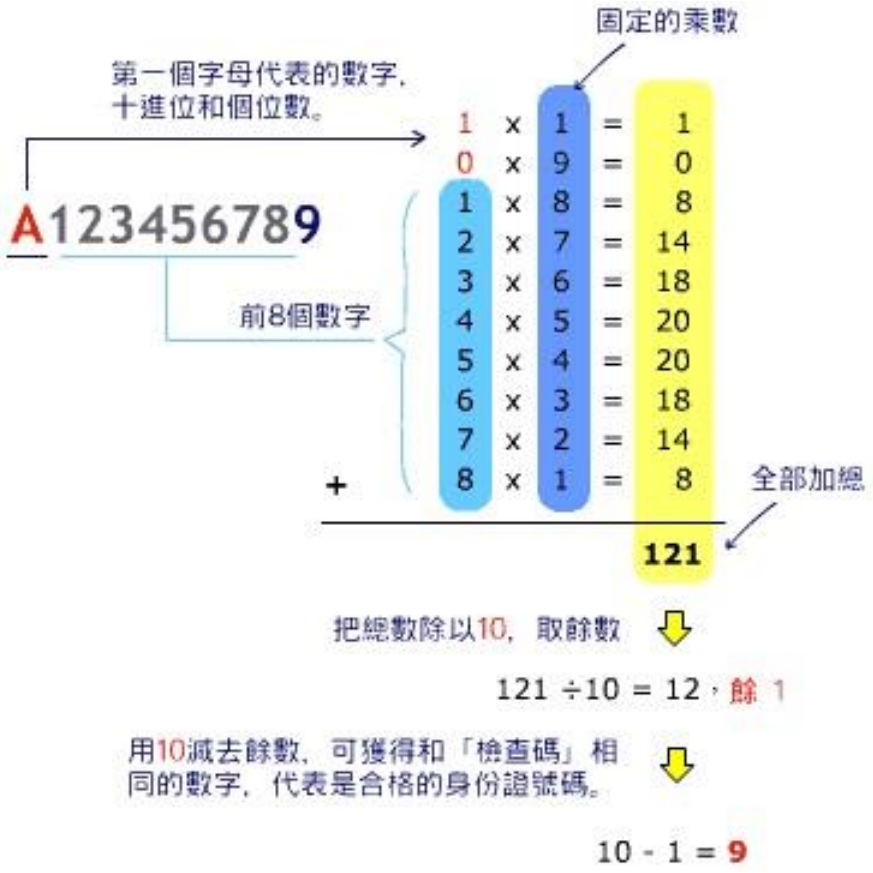
字母	轉換字元	縣市
A	10	臺北市
B	11	臺中市
C	12	基隆市
D	13	臺南市
E	14	高雄市
F	15	新北市
G	16	宜蘭縣
H	17	桃園市
I	34	嘉義市
J	18	新竹縣
K	19	苗栗縣

字母	轉換字元	縣市
M	21	南投縣
N	22	彰化縣
O	35	新竹市
P	23	雲林縣
Q	24	嘉義縣
T	27	屏東縣
U	28	花蓮縣
V	29	臺東縣
W	32	金門縣
X	30	澎湖縣
Z	33	連江縣

字母	轉換字元	原行政區	停發日期	現行行政區
L	20	臺中縣	2010年12月25日	臺中市
R	25	臺南縣	2010年12月25日	臺南市
S	26	高雄縣	2010年12月25日	高雄市
Y	31	陽明山管理局	1975年	臺北市

綜合練習

- 身份證字號檢查
  - 第十個字元為檢查號碼，檢查碼產生的規則為：



綜合練習

- 身份證字號檢查
  - 檢查步驟：

    1.檢查長度，須為十個字元。

    2.檢查格式，第一碼須為字母，後九碼須為數字。

    3.檢查性別，第一個數字須為1(男性)或2(女性)。

    4.計算檢查碼是否正確。

  - 以上檢查都正確即為正確之身分證號碼，但不能確定是否已分配出去或尚在使用，僅能確定其「正確」。

綜合練習

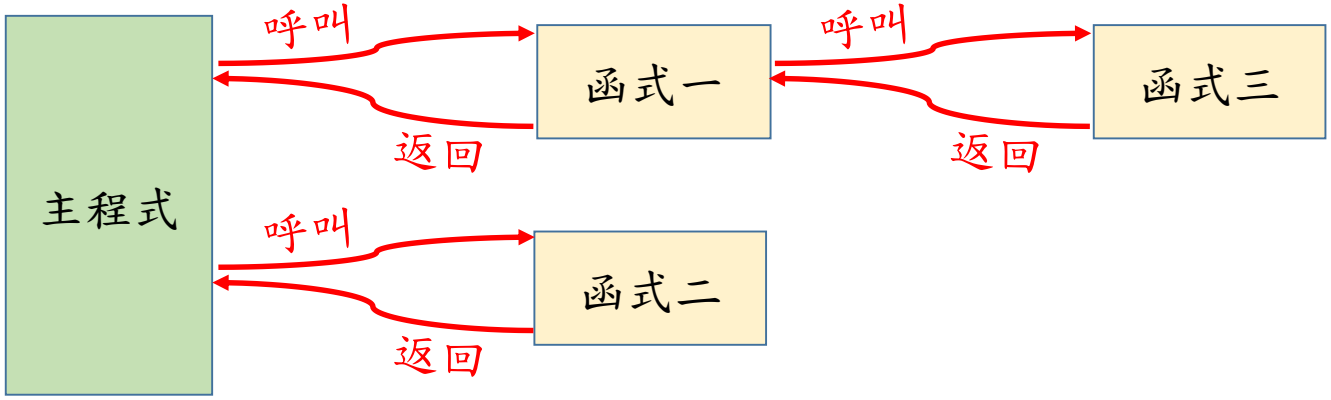
- 參考程式：

```
ID = input("請輸入身份證字號:").upper()
if(len(ID) != 10):
    print("錯誤，身份證字號須為10碼")
elif(not ID[0].isalpha()):
    print("錯誤，第一碼須為英文字母")
elif(not ID[1:].isdigit()):
    print("錯誤，後九碼須為數字")
elif(ID[1]<'1' or ID[1]>'2'):
    print("錯誤，第一個數字須為1或2")
else:
    X = {'A':10,'B':11,'C':12,'D':13,'E':14,'F':15,'G':16,'H':17,'I':34,
         'J':18,'K':19,'L':20,'M':21,'N':22,'O':35,'P':23,'Q':24,'R':25,
         'S':26,'T':27,'U':28,'V':29,'W':32,'X':30,'Y':31,'Z':33 }
    num = X[ID[0]] // 10 + (X[ID[0]] % 10) * 9
    for i in range(2,10):
        num += int(ID[-i]) * (i-1)
    ans = 10 - num % 10
    if(ans == int(ID[-1])):
        print(ID + " 是正確的身分證字號")
    else:
        print(ID + " 不是正確的身分證字號")
```

休息一下~

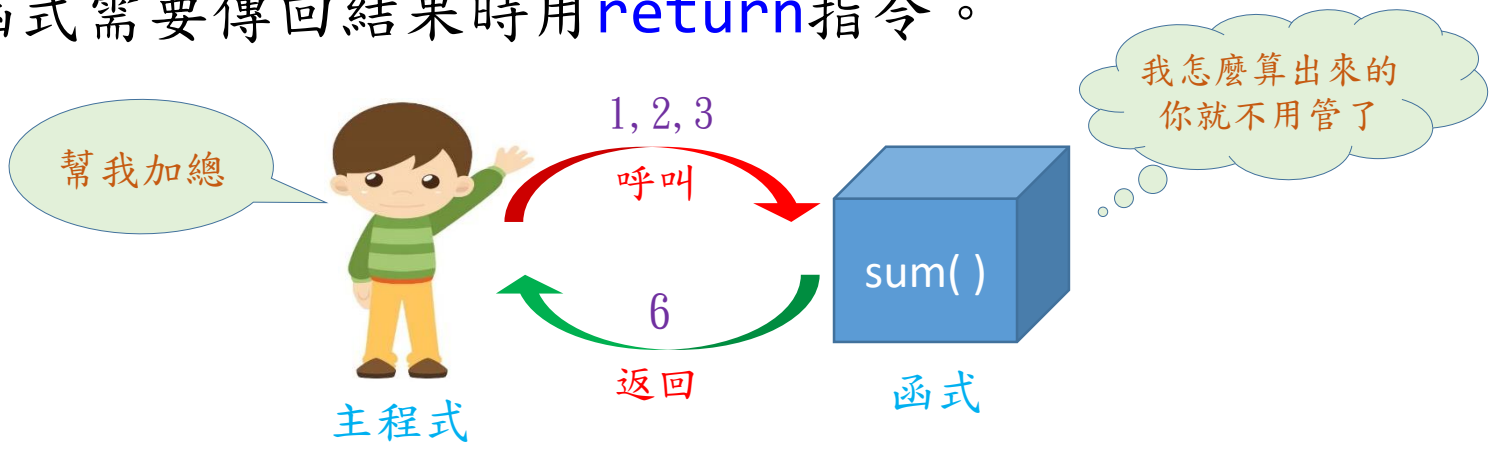
函式(Function)

- 函式(Function)有時也稱副程式。
- 將程式的步驟依功能分成幾個小部分，再由主程式去呼叫，可使較大的程式簡單明瞭並容易維護。
- 程式中重複的段落可以寫成函式的方式，代入不同的參數重複呼叫。
- 函式中可以再呼叫其他函式。



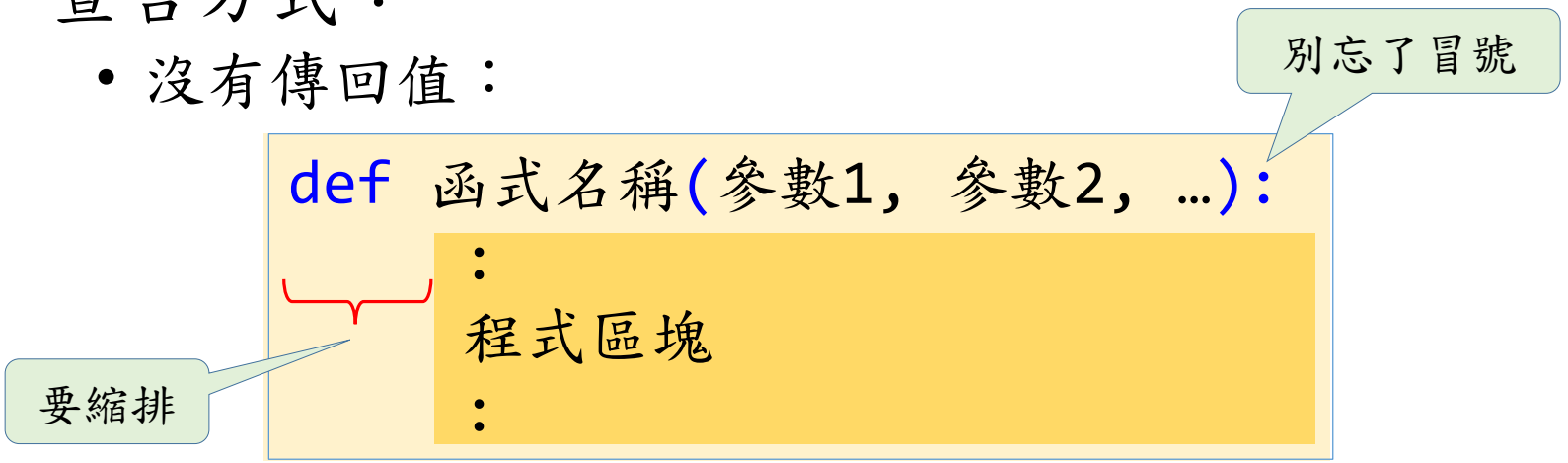
函式(Function)

- 你可以想像函式就像一個黑盒子，它有特定功能，你呼叫它時不用管它執行的細節，只要它完成工作、傳回你要的東西就可以了。
- 如果一個程式中有兩個或以上相同或類似的功能段落，就應該考慮將這個段落寫成一個函式，再透過傳遞參數呼叫的方式來簡化程式。
- 可以的話盡量使用內建函式。
- 函式需要傳回結果時用return指令。



函式(Function)

- 宣告方式：
  - 沒有傳回值：



  - 有傳回值：(傳回的值要有相應的變數接收)

```
def 函式名稱(參數1, 參數2, …):
    :
    程式區塊
    :
    return 值1, 值2, ...
```

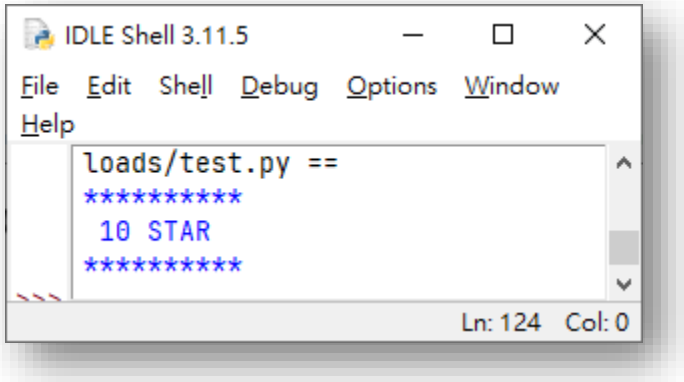
函式(Function)

- 沒有傳回值，單純的呼叫例子：

```
def printStar():
    print('*' * 10)

#---main---
printStar()
print(" 10 STAR  ")
printStar()
```

呼叫

printStar()
被呼叫了兩次



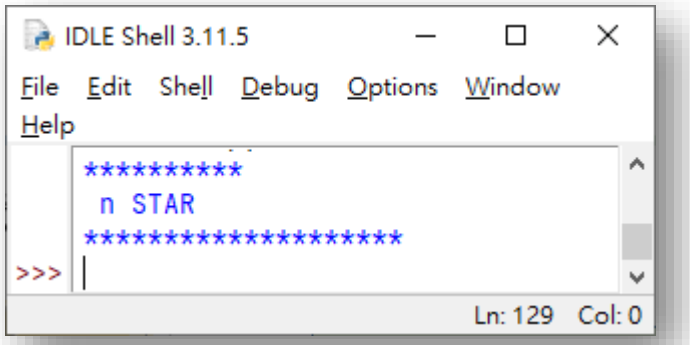
函式(Function)

- 沒有傳回值，但呼叫時傳遞參數例子：

```
def printStar(n):
    print('*' * n)

#---main---
printStar(10)
print(" n STAR  ")
printStar(20)
```

呼叫

printStar(n)被呼叫了兩次，每次隨傳入的n值印出n個'*'

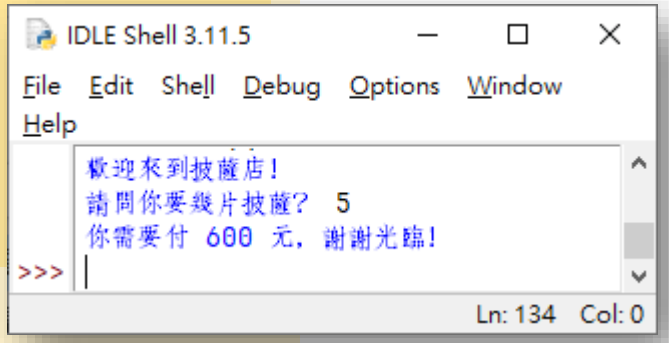


函式(Function)

- 有傳回值的例子：
  - 寫一個函式，計算訂購幾片披薩的總價。每片披薩是120元。
  - 有使用return傳回值時一定要有變數接收傳回值。

```
# 定義一個函式，計算總價
def countPizza(n):
    price = 120
    total = price * n
    return total   # 傳回總價

# 主程式
print("歡迎來到披薩店！")
num = int(input("請問你要幾片披薩？ "))

pay = countPizza(num)   # 呼叫函式
print("你需要付", pay, "元，謝謝光臨！")
```

呼叫　返回　傳遞



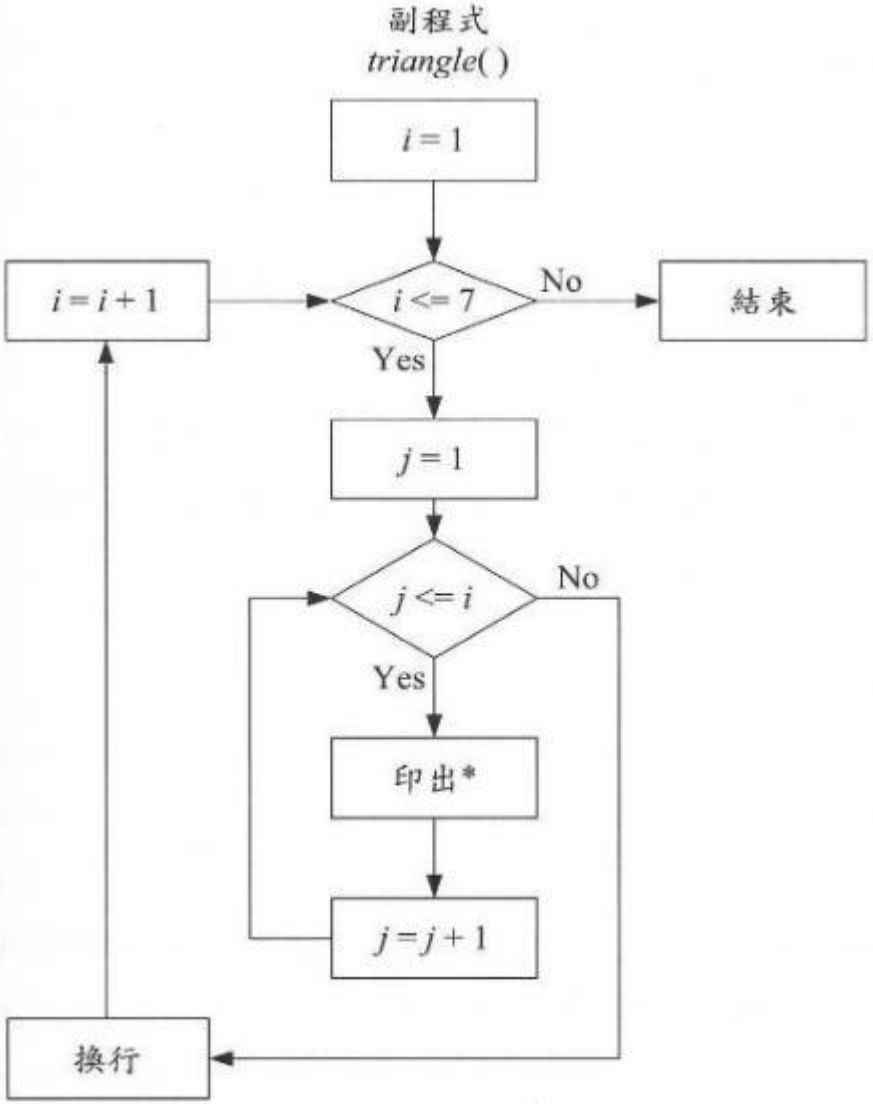
函式(Function)

- 練習輸出「＊」三角形，如圖。
- 印完後要顯示"輸入x鍵離開,其他任何鍵繼續:"，如果使用者按x鍵，程式就要停止，其他任何鍵就會再印一次星號。

```
＊
＊ ＊
＊ ＊ ＊
＊ ＊ ＊ ＊
＊ ＊ ＊ ＊ ＊
＊ ＊ ＊ ＊ ＊ ＊
```

函式(Function)

- 流程圖：



函式(Function)

- 參考程式：

```
def triangle():
    for i in range(7):
        for j in range(i):
            print("*", end="")
        print()
#main
triangle()
ch = input("輸入x鍵離開,其他任何鍵繼續:")
while ch != "x":
    triangle()
    ch = input("輸入x鍵離開,其他任何鍵繼續:")
```

- print()會單純的換行而已

函式(Function)

- 練習將兩個一維清單相加：
  - 1.將一組數字讀入A清單。
  - 2.將一組數字讀入B清單。
  - 3.將A清單與B清單相加，成為C清單，列印C清單。

- 此例我們可以寫三個函式：
  - 1.將數字讀入一個清單的函式。
  - 2.將兩個清單相加的函式。
  - 3.列印一個清單的函式。

- 函式通常會放在主程式前面。

函式(Function)

- 1.將數字讀入一個清單的函式。

```
def input_array(x, n):
    print("---開始讀入清單---")
    for i in range(n):
        x[i]= int(input("輸入數字: "))
```

- 函式名稱為input_array，並需要傳入x及n兩個參數，x是清單名稱，n是要輸入的數字的數目。

函式(Function)

- 2.將兩個清單相加的函式。

```
def add_array(x, y, z, n):
    print("---開始清單相加---")
    for i in range(n):
        z[i] = x[i] + y[i]
```

- 函式名稱為add_array，並需要傳入x、y、z三個清單名稱及數字的數目n。

函式(Function)

- 3.列印一個清單的函式。

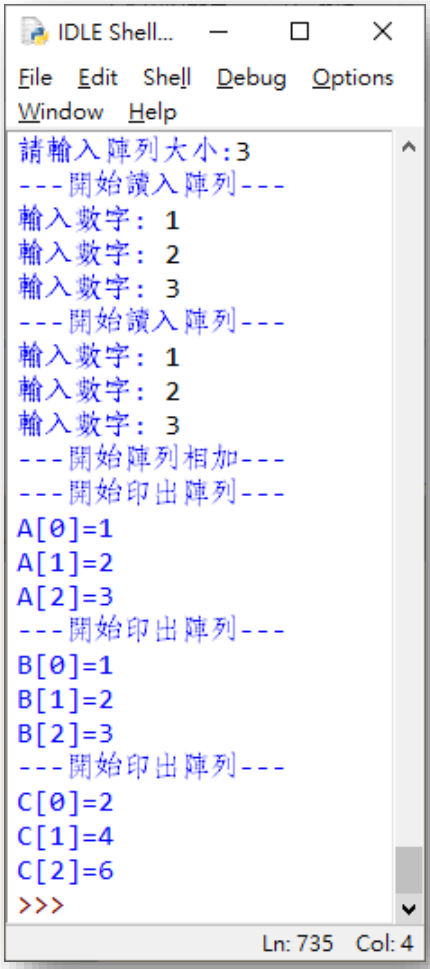
```
def print_array(x, name, n):
    print("---開始印出清單---")
    for i in range(n):
        print(f"{name}[{i}]={x[i]}")
```

- 函式名稱為print_array，並需要傳入清單名稱x，及要印出的清單名稱的字母name，及數字的數目n。

函式(Function)

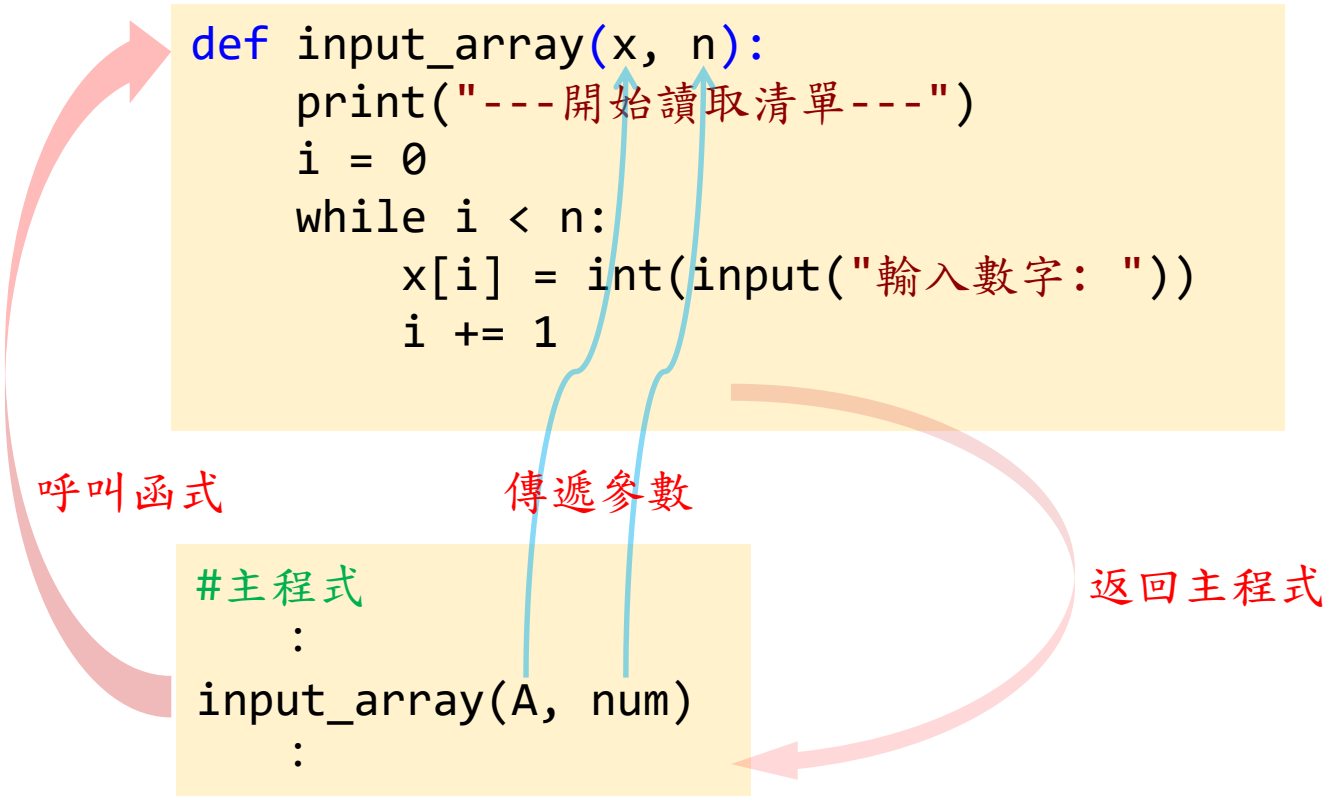
- 主程式：

```
num = int(input("請輸入清單大小:"))

#宣告清單
A = [0] * num
B = [0] * num
C = [0] * num

#依序呼叫各個動作的函式
input_array(A, num)
input_array(B, num)
add_array(A, B, C, num)
print_array(A, "A", num)
print_array(B, "B", num)
print_array(C, "C", num)
```



函式(Function)

- 呼叫函式方式：



函式(Function)

- 完整程式：

```
def input_array(x, n):
    print("---開始讀入清單---")
    for i in range(n):
        x[i]= int(input("輸入數字: "))

def add_array(x, y, z, n):
    print("---開始清單相加---")
    for i in range(n):
        z[i] = x[i] + y[i]

def print_array(x, name, n):
    print("---開始印出清單---")
    for i in range(n):
        print(f"{name}[{i}]={x[i]}")
```

```
#main
num = int(input("請輸入清單大小:"))
A = [0] * num
B = [0] * num
C = [0] * num

input_array(A, num)
input_array(B, num)
add_array(A, B, C, num)
print_array(A, "A", num)
print_array(B, "B", num)
print_array(C, "C", num)
```

函式(Function)

- 求清單中最大之數：將5個數字讀入清單，求清單中最大之數。我們需要兩個函式。
  - 1.將5個數字讀入一個清單的函式。
  - 2.求這個清單中數字最大值的函式。

- 這裡使用return敘述將結果回傳給呼叫它的敘述，所以呼叫這個函式時需準備接收傳回值。

- Python可以直接傳遞清單，與傳遞變數相同。
- len()函式會傳回清單或字串的長度(個數)
- 本例不使用內建的max()函式。

函式(Function)

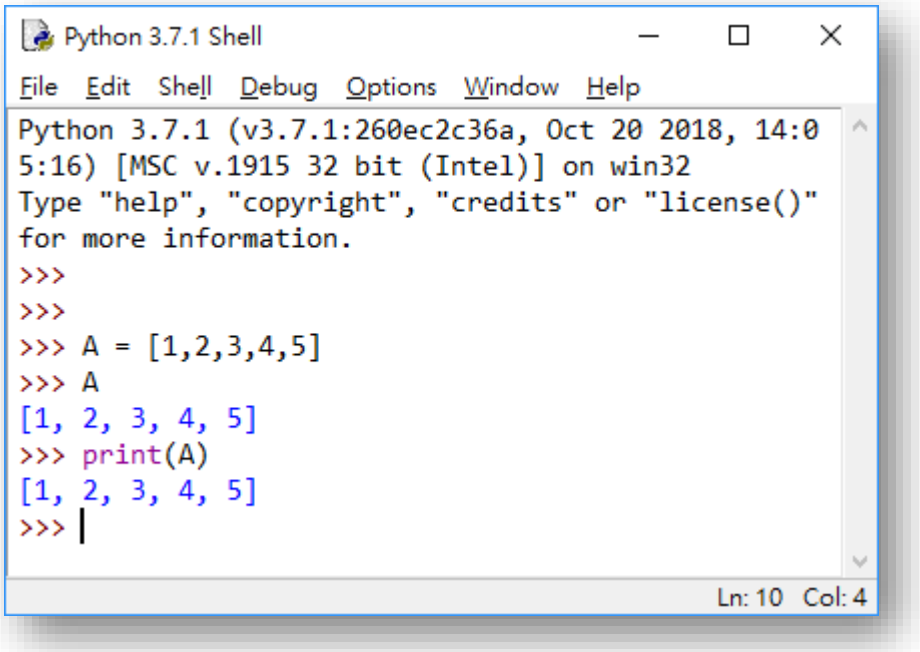
- 參考程式：

```
def createArray():
    A = [0] * 5
    for i in range(5):
        A[i] = int(input("請輸入數字:"))
    return A

def getMax(A):
    max = 0
    for i in range(len(A)):
        if A[i] > max:
            max = A[i]
    return max

#main
array = createArray()
print(getMax(array))
```

函式(Function)

- Array + 1：
  - 1.讀入5個數字，將這5個數字放入一個清單中。
  - 2.對清單中的每一個數字都加1，然後列印出來。
- 注意Python使用print( )可以直接列印清單，不須使用迴圈。

例：



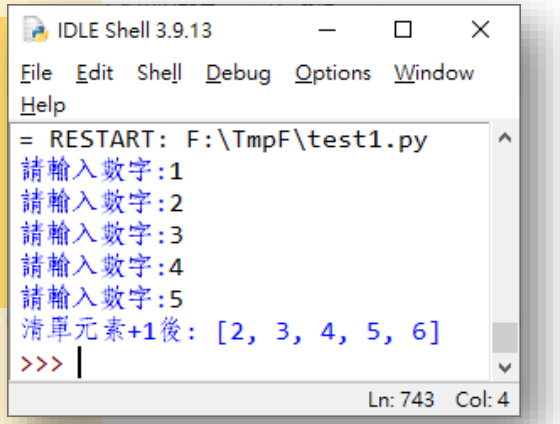
函式(Function)

- 參考程式：

```
def createArray():
    A = [0] * 5
    for i in range(5):
        A[i] = int(input("請輸入數字:"))
    return A
```

```
def addition(A):
    for i in range(len(A)):
        A[i] += 1
    return A
```

```
#main
array = createArray()
print("清單元素+1後:",addition(array))
```



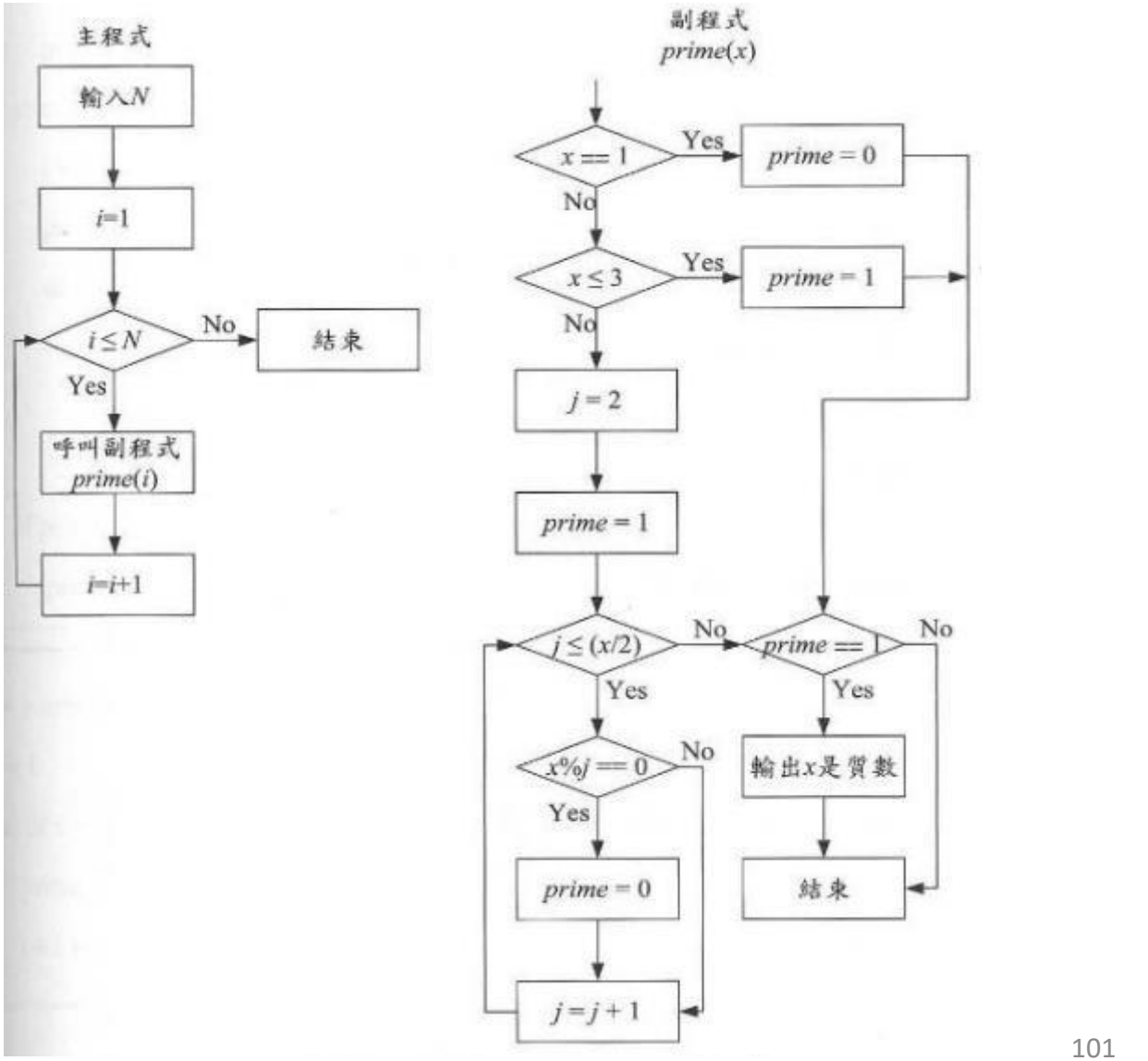
函式(Function)

- 求小於N的所有質數。
  - 質數（Prime number），又稱素數，指在大於1的自然數中，除了1和該數自身外，無法被其他自然數整除的數。
  - 例如假設N=20，則小於20的質數有：
    2，3，5，7，11，13，17，19。

- 我們需要一個函式，這個函式輸入一個正整數x，然後判斷x是否是一個質數。

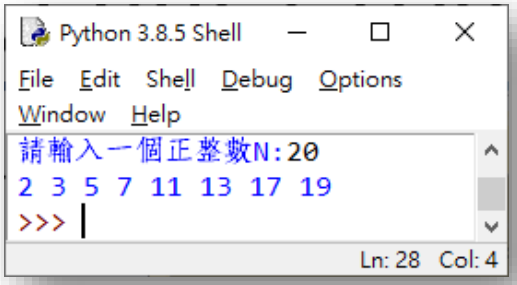
函式(Function)

- 流程圖：



函式(Function)

- 參考程式：




```
def prime(x):
    if x == 1:
        prime = 0
    elif x <= 3:
        prime = 1
    else:
        j = 2
        prime = 1
        while j <= (x**0.5):
            if x % j == 0:
                prime = 0
            j += 1
    if prime == 1:
        print(x, end=" ")
#main
N = int(input("請輸入一個正整數N:"))
i = 1
while i <= N:
    prime(i)
    i += 1
```

1不列入質數

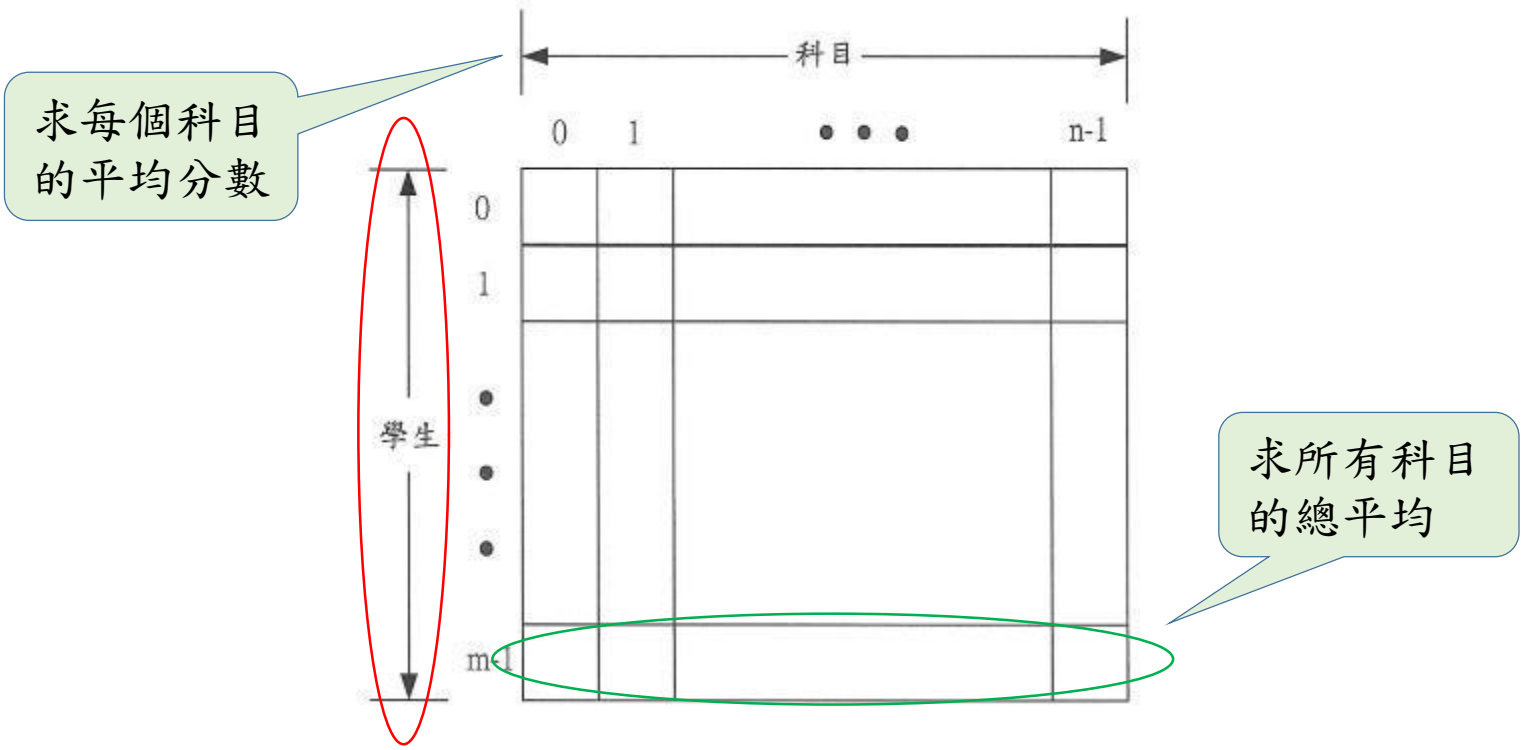
2,3是質數，不用計算

算到x的開根號即可，再加上去也不可能整除x了

不能輸入0或1，會沒有輸出訊息

函式(Function)

- 求出各科平均分數及總平均分數。
  - 我們有m個學生，每一個學生有n門課，我們的任務是對每一門課，求這門課的平均分數，然後求所有科目的總平均。



函式(Function)

- 參考程式：

```
def subjectAverage(i, m, Array):  #求第i科平均成績
    sum = 0
    for j in range(m):
        sum += Array[j][i]
    sum /= m
    print("第", i+1, "科平均為:", sum)
    return sum
def getAverage(m, n, Array):  #求各科總平均
    sum = 0
    for i in range(n):
        sum = sum + subjectAverage(i, m, Array)
    sum /= n
    print("總平均:", sum)
def inputArray(n):  #輸入某位學生n科成績
    B = [0] * n
    for i in range(n):
        print("輸入第", i+1, "科:")
        B[i] = int(input())
    return B
```

函式(Function)

- 參考程式：

```
def input2DArray(m, n):  #輸入m個學生的n科成績
    A = [[0]*n for _ in range(m)]
    for i in range(m):
        print("--請輸入第",i+1,"位學生的成績--")
        A[i] = inputArray(n)
    return A

#main
m = int(input("請輸入學生數目:"))
n = int(input("請輸入科目數:"))
Array = input2DArray(m, n)
getAverage(m, n, Array)
```

- 函式裡可以再呼叫其他函式，也可以呼叫自己(這種情況稱遞迴(Recursive))

函式(Function)

- 判斷日期先後順序。
  - 輸入日期A與日期B，若：
  - 日期A在日期B之前，輸出「日期A在日期B之前」。
  - 日期A在日期B之後，輸出「日期A在日期B之後」。
  - 日期A在日期B相同，輸出「日期A在日期B相同」。
- 假設輸入格式為(日/月/年)「21/01/2018」存入變數temp，則使用split()方法可將字串拆開。

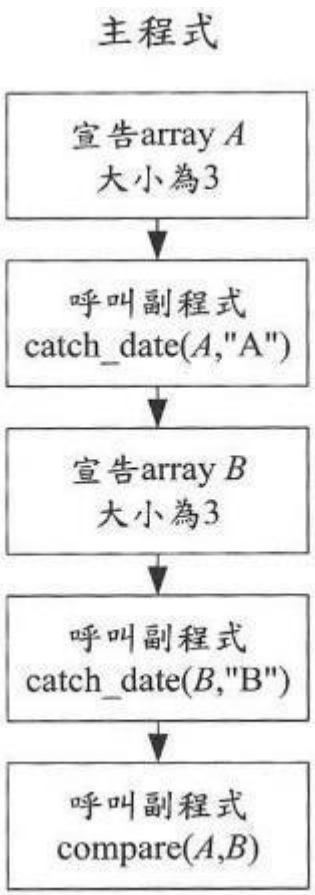
```
date = temp.split("/")
```

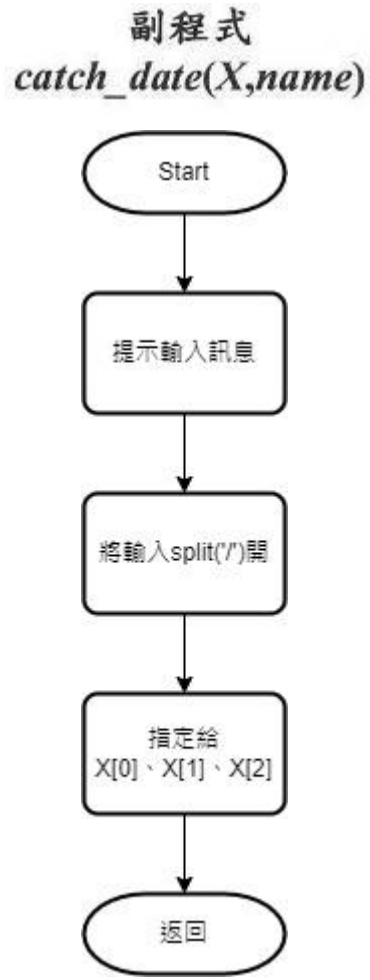
- 以「/」為分隔，拆成三組字串，存入date清單，使得

```
date[0]="21"，date[1]="01"，date[2]="2018"
```

函式(Function)

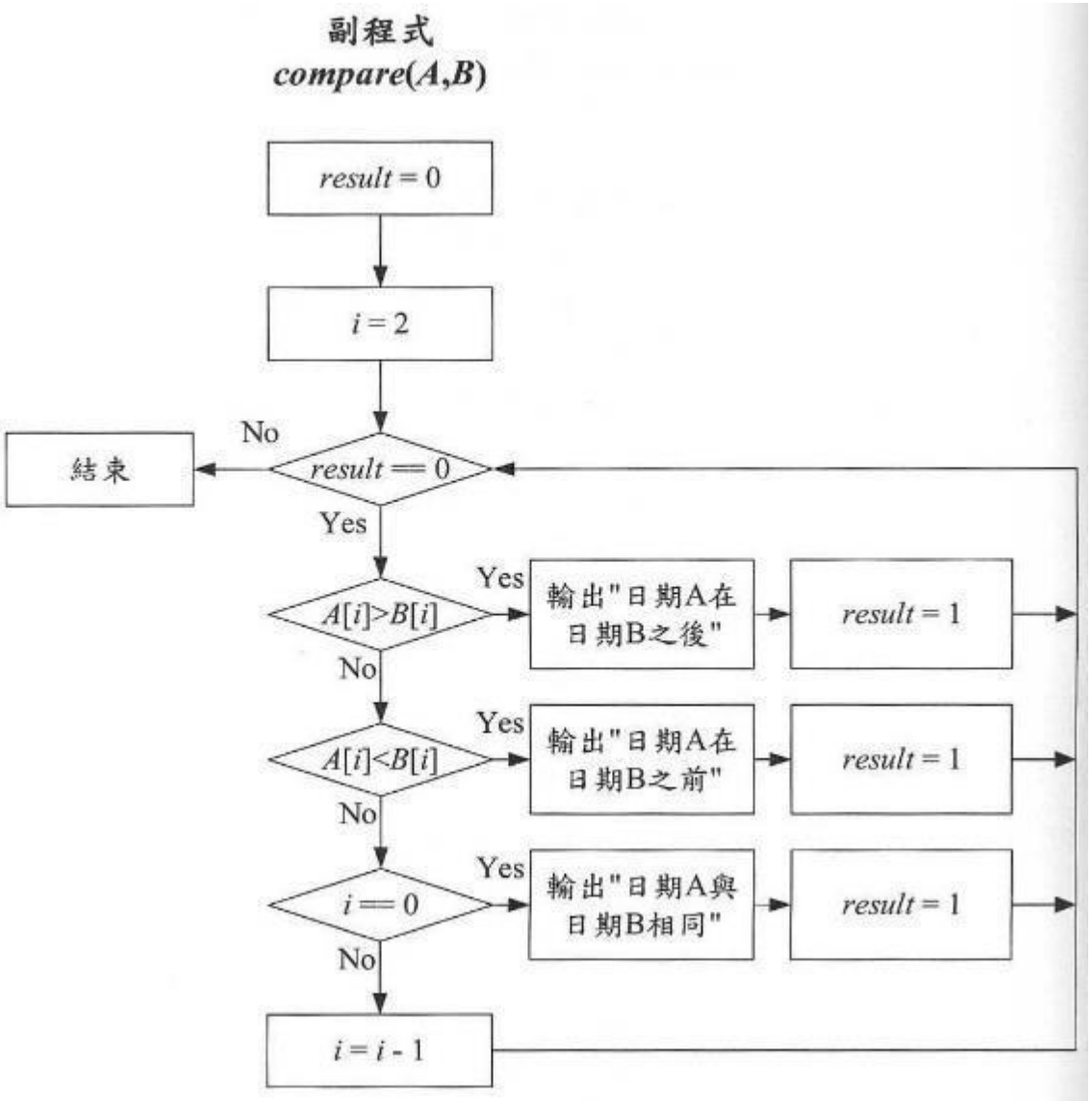
- 流程圖：





函式(Function)

- 流程圖：



函式(Function)

- 參考程式：

- 函式要寫在前面，最後再寫主程式。

```
def catch_date(X, day):  #將日期分開存入清單函式
    print("請輸入日期",day,"(日/月/年):",end="")
    X[0], X[1], X[2] = input().split("/")

def compare(A,B):  #比較日期函式
    result = 0
    i = 2
    while result == 0:
        if A[i] > B[i]:
            print("日期A在日期B之後")
            result = 1
        elif A[i] < B[i]:
            print("日期A在日期B之前")
            result = 1
        elif i == 0:
            print("日期A與日期B相同")
            result = 1
        else:
            i -= 1
```

```
#main
A = [0] * 3
catch_date(A, "A")
B = [0] * 3
catch_date(B, "B")
compare(A, B)
```

函式(Function)

- 關於分割字串函式split()的一些說明：
  - 依照指定的字元分割字串：

```
temp = "Hello World".split() #以空白字元分割字串
```

  - 則temp為 ['Hello', 'World']

```
temp = "2023-01-10".split('-') #以-符號分割字串
```

  - 則temp為 ['2023', '01', '10']

```
temp = input().split()    #假設輸入"12 34"
```

  - 則temp為 ['12', '34']

```
temp = input().split('/') #假設輸入"2023/1/12"
```

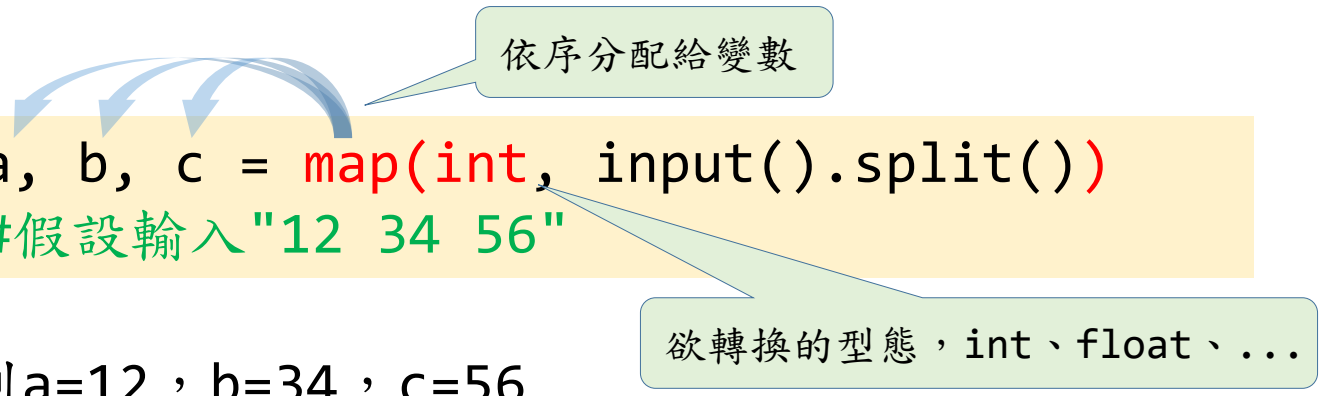
  - 則temp為 ['2023', '1', '12']

函式(Function)

- 關於分割字串函式split()的一些說明：
  - 若一次輸入多個資料，要用多個變數接收：

```
Y, M, D = input().split('/')
#假設輸入"2023/01/12"
```

  - 則Y="2023"，M="01"，D="12"
  - 將多個輸入轉成數值(使用map()函式)：



  - 則a=12，b=34，c=56

函式(Function)

- 關於分割字串函式split()的一些說明：
  - 若不確定有多少資料要接收，就用一個變數來接收：

```
#假設輸入"1 2 3 4"
temp = map(int, input().split())
for i in temp:  #再依序取出輸入的數值
    print(i)
```

  - 此時temp是map型態的物件。

  - 你也可以將temp轉成list形式：

```
#假設輸入"1 2 3 4"
temp = list(map(int, input().split()))
print(temp)
```

  - 則temp = [1, 2, 3, 4]

函式(Function)

- ord()和chr()內建函式：
- 在ASCII碼中，字母A的編碼為65_{10}，則函式

```
ord("A") 會傳回A的編碼65
chr(65) 會傳回字母A
```

- 所以字母也可以加或減，會得到另一個字母。

函式(Function)

- 凱薩密碼另一種寫法：
- 為了保密，可以將原文加密，加密的方法有很多種，最簡單的英文加密方法是將每一個字母做位移，假設只考慮26個英文大寫字母，且位移規則如下：

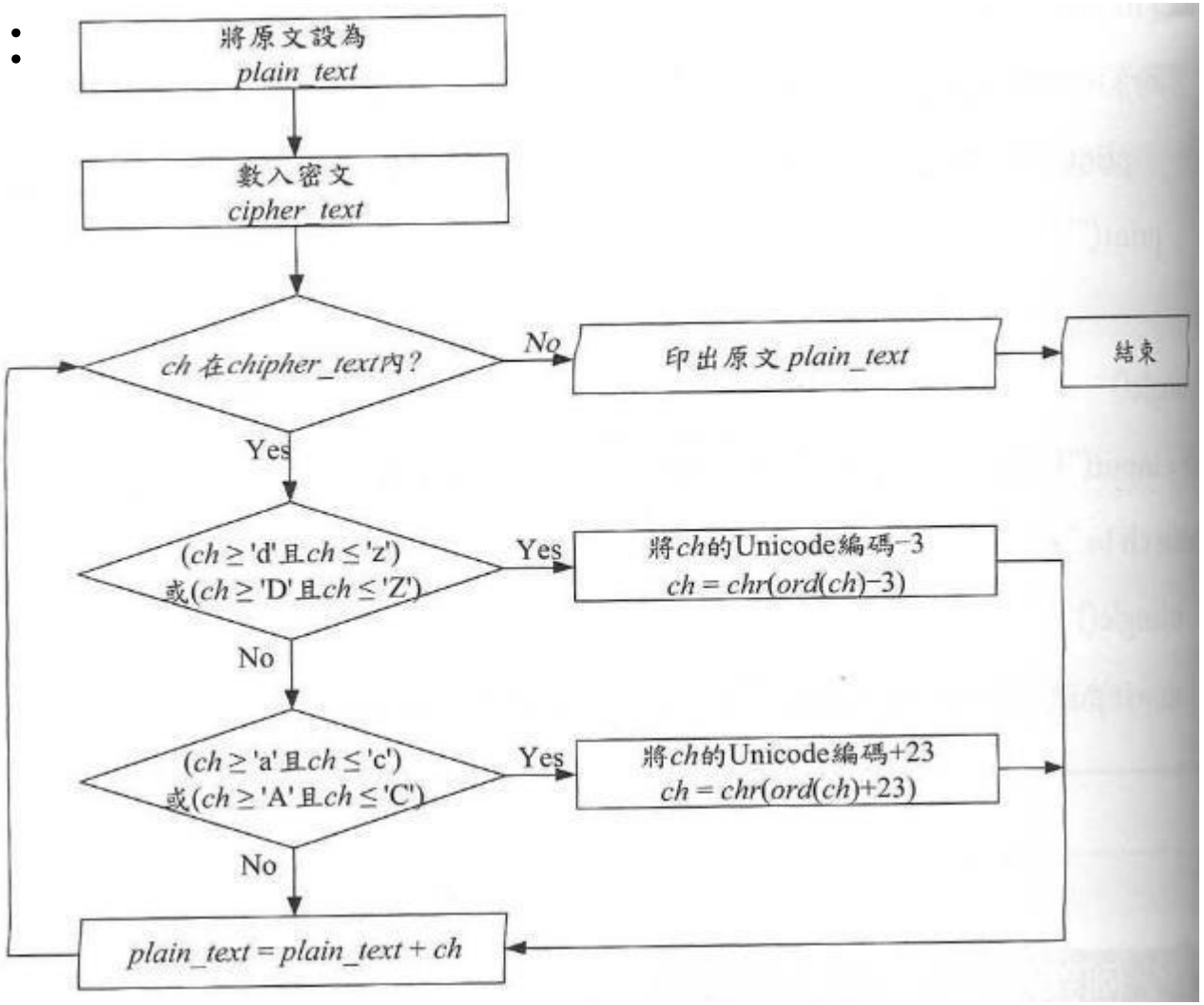
原來字母	A	B	C	…	X	Y	Z
轉換後字母	D	E	F	…	A	B	C

小寫字母規則亦同。可以看出皆是往後移三個字母。

- 還原的動作叫解密，假設加密後的文字為DSSOH和CRR，解密後的文字就是APPLE和ZOO。

函式(Function)

- 流程圖：



函式(Function)

- 參考程式：

```
plain_text = ""
cipher_text = input("請輸入密文:")

for ch in cipher_text:
    if (ch>="d" and ch<="z") or (ch>="D" and ch<="Z"):
        ch = chr(ord(ch) - 3)
    elif (ch>="a" and ch<="c") or (ch>="A" and ch<="C"):
        ch = chr(ord(ch) + 23)
    plain_text += ch

print("原文為:" + plain_text)
```

  - 此做法不建立字典，而是用計算，所以大小寫皆可加密和解密。

函式(Function)

- 常用內建函式：Python對許多常用的功能提供了函式。

		內建函式		
abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

函式(Function)

- 常用內建函式：

函式	意義	範例	運算結果
abs(x)	取得數值 x 的絕對值	abs(-3)	3
		abs(5.6)	5.6
bool (x)	將 x 轉成布林值	bool(1)	True
		bool(3>5)	False
chr(x)	取得整數 x 的字元	chr(65)	A
		chr(97)	a
float(x)	將 x 轉成浮點數	float(3)	3.0
hex(x)	將數值 x 轉成 16 進位	hex(17)	0x11
max(串列)	取得串列中的最大值	max(1,2,3,4,5)	5
min(串列)	取得串列中的最小值	min(1,2,3,4,5)	1
oct(x)	將數值 x 轉成 8 進位	oct(17)	0o21
ord(x)	取得字元 x 的 Unicode 編碼值	ord('李')	26446
		ord('a')	97
pow(x,y)	計算 x 的 y 次方值	pow(6,3)	216
sorted(串列)	將串列由小到大排序	sorted([1,3,5,2,4])	[1, 2, 3, 4, 5]
str(x)	將 x 轉換成字串	str(35)	'35'
sum(串列)	將串列加總的和	sum([1,3,5])	9

函式(Function)

- 傳入不定數目的參數給函式：
  - 方式一：
    - *號位置接受任意多個非關鍵字（non-keyword）參數，在函式中將其轉化為元組

```
def 函式名稱( *參數 ):
```

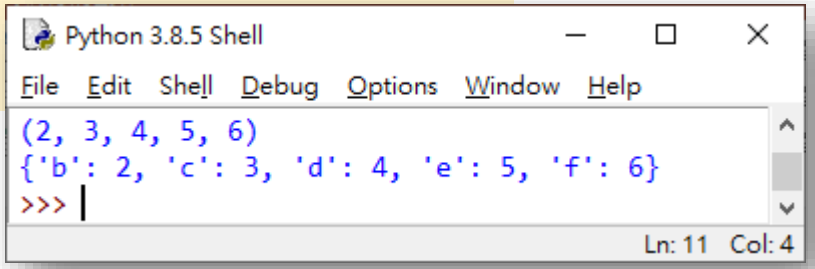
  - 方式二：
    - **號位置接受任意多個關鍵字（keyword）參數，在函式**位置上轉化為詞典[key:value, key:value]

```
def 函式名稱( **參數 ):
```

函式(Function)

- 例如：

```
#a是一個普通傳入參數，*b是一個非關鍵字星號參數
def one(a, *b):
    print(b)

one(1,2,3,4,5,6)

#a是一個普通關鍵字參數，**b是一個關鍵字雙星號參數
def two(a,**b):
    print(b)

two(a=1,b=2,c=3,d=4,e=5,f=6)
```



休息一下~

變數的生命週期及可視範圍

- 變數的生命週期：
  - 是指其存在於記憶體內的時間。
  - 全域變數(global variables)，從程式開始到程式結束為止。
  - 區域變數(local variables)，從其宣告開始至區塊結束為止，例如某函式內。
- 變數的可視範圍：
  - 指的是哪些程式段可以存取該變數。
  - 全域變數，所有程式段都可以存取它。
  - 區域變數，只有宣告它的那個程式區塊可以存取它。

變數的生命週期及可視範圍

變數a於func1執行時產生，函式結束時消失。
只有func1可以存取它，其它人都不行。

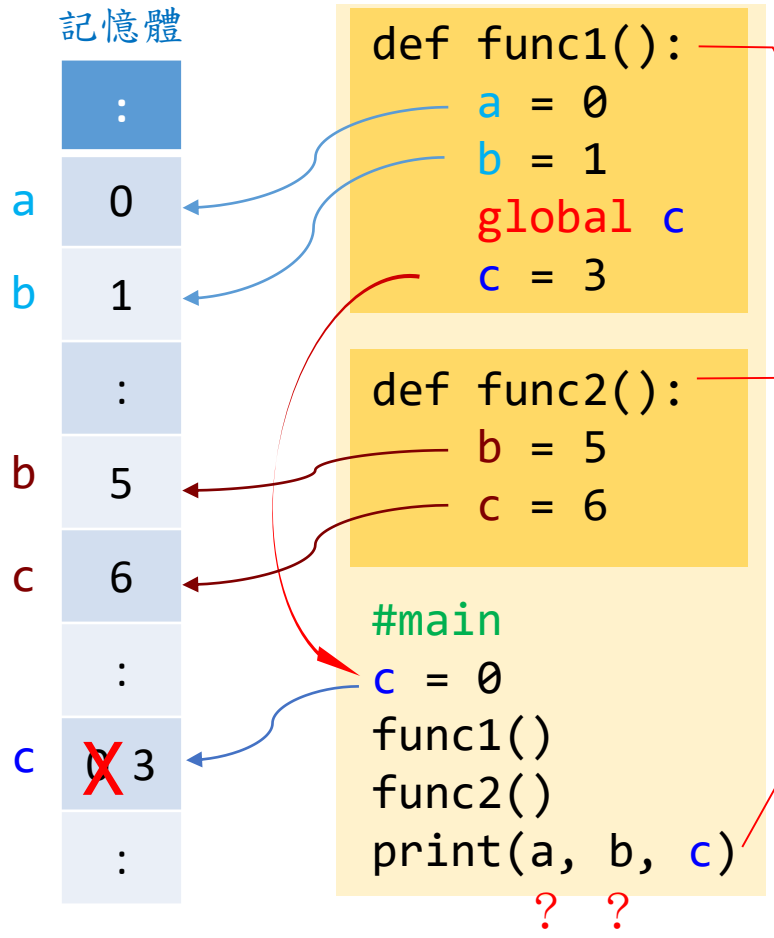
變數b於func2執行時產生，函式結束時消失。
只有func2可以存取它，其它人都不行。

變數c於程式執行時產生，程式結束時消失。
所有人都可以存取它。

```
def func1( ):
    a = 0   #區域變數
    :

def func2( ):
    b = 0   #區域變數
    :

#main
c = 0   #全域變數
:
```

變數的生命週期及可視範圍

- Ex：下列程式a、b、c內容為何?



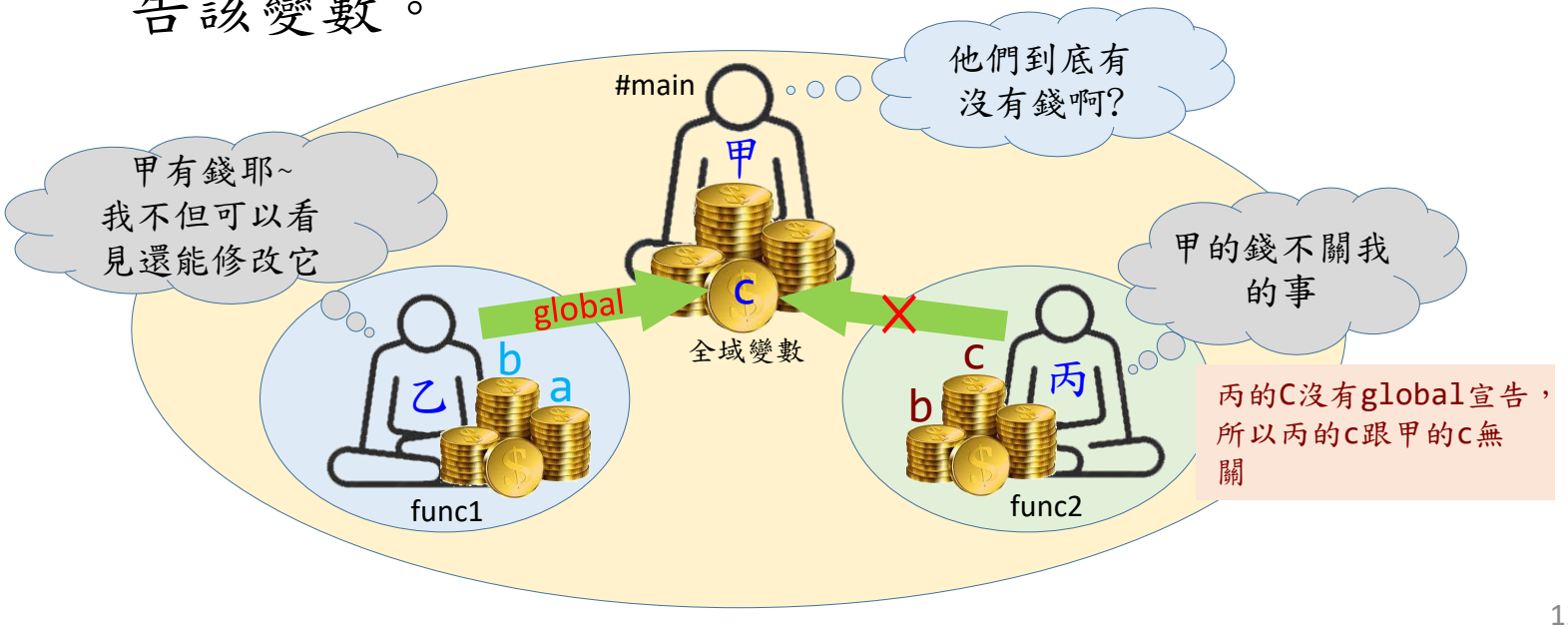
func1有自己變數a,b，但c因為有global宣告，所以是用全域變數的c。(若全域變數裡並無c，則這個變數會成為全域變數)
這裡的b和func2的b沒有關係。

func2有自己變數b,c，和全域變數的c沒有關係。
這裡的b和func1的b沒有關係。

這一行會有錯誤訊息，因為它並不知道a,b是什麼。
func1和func2執行完返回後它們的區域變數就不存在了。

變數的生命週期及可視範圍

- 注意事項：
  - 全域變數和函式內的區域變數盡量不要用相同的名字，以免混淆。
  - 若區域變數與全域變數同名，Pyton會視其為完全不同的變數。
  - 若在函式中要存取全域變數，需先用global指令宣告該變數。



練習題

- 7-1.在主程式中接受使用者輸入梯形之上底、下底與高的值，並呼叫一函式，將之上底、下底與高的值傳入該函式後，在函式中計算並印出梯形面積的值。

- 7-2.寫一函式，接受主程式傳進的清單，分別計算清單上奇數元素之平均值與偶數元素之平均值並印出。

- 7-3.在主程式中接受使用者輸入A、B、C的值並呼叫一函式，將A、B、C的值傳入函式中，在函式中判斷|A|、|B|、|C|之大小順序，回傳絕對值最大者並在主程式中印出，|A|、|B|、|C|之最大值。

練習題

- 7-1.參考程式：

```
def trapezoid(A,B,H):
    return ((A + B) * H) / 2

a = float(input("請輸入上底:"))
b = float(input("請輸入下底:"))
h = float(input("請輸入高:"))
print("此梯形面積為:", trapezoid(a, b, h))
```

- 函式trapezoid()的傳回值直接交給print()列印。

練習題

- 7-2.參考程式：

```
def averageElement(x):
    odd_avg = sum(x[::2]) / len(x[::2])
    even_avg = sum(x[1::2]) / len(x[1::2])
    return odd_avg, even_avg

#main
L = int(input("請輸入有多少個元素:"))
A = [0] * L
for i in range(L):
    print("請輸入元素", i+1, ":", end="")
    A[i] = int(input())

a, b = averageElement(A)
print("奇數元素平均:", a, "\n偶數元素平均:", b)
```

練習題

- 7-3.參考程式：

abs()函式會傳回數值的絕對值

```
def maxABS(a, b, c):
    return max(abs(a), abs(b), abs(c))

#main
A = int(input("請輸入數值A:"))
B = int(input("請輸入數值B:"))
C = int(input("請輸入數值C:"))
print("|最大值| =", maxABS(A, B, C))
```

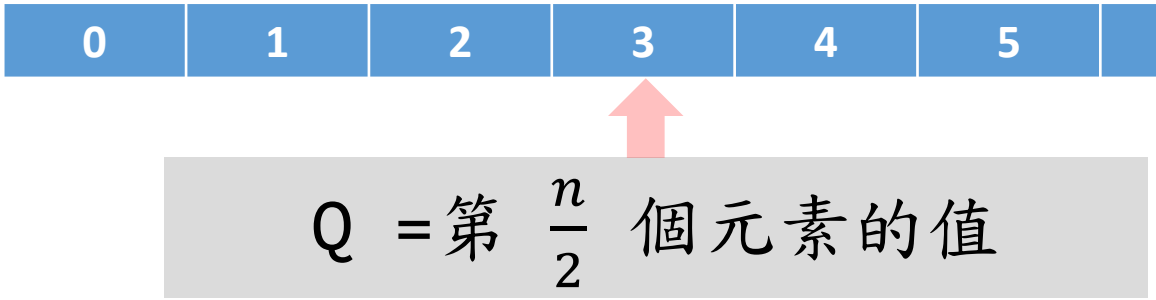
練習題

- 7-4.在主程式中接受使用者輸入首項a_1，公比r與項數n，呼叫一函式並將a_1、r、n的值傳入，在函式中計算等比級數第n項的值並回傳，在主程式中印出該值。
  註：等比級數第n項的值為：$a_n = a_1 r^{n-1}$

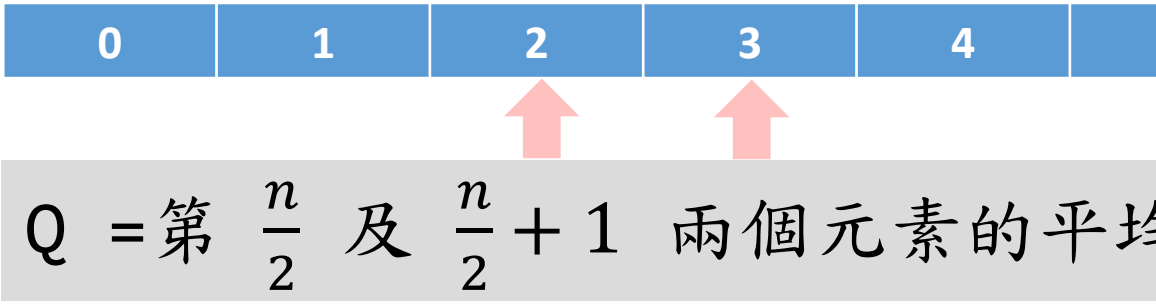
- 7-5.在主程式中接受使用者輸入一個清單的值，將清單的值與清單的大小傳入一函式中，此函式將會計算該清單之中位數並回傳，主程式在收到此函式的回傳值之後印出。

練習題

- 7-5.說明：
- 若清單元素個數為奇數，則中位數為正中間元素的值。



- 若清單元素個數為偶數，則中位數為中間兩個元素的平均值。



練習題

- 7-4.參考程式：

```
def progression(a1, r, n):
    return a1 * (r**(n-1))

#main
A1 = int(input("請輸入首項:"))
R = int(input("請輸入公比:"))
N = int(input("請輸入項數:"))
print(progression(A1, R, N))
```

- 註：等比級數第n項的值為：

$$a_n = a_1 r^{n-1}$$

練習題

- 7-5.參考程式：

```
def median(x, L):
    x.sort()   #先將清單排序
    if L % 2 != 0:
        return x[int(L/2)]
    else:
        return (x[int(L/2-1)] + x[int(L/2)]) / 2

#main
L = int(input("請輸入清單長度:"))
X = [0] * L
for i in range(L):
    print("請輸入數值", i+1, ":", end="")
    X[i] = int(input ())
print("中數:", median(X, L))
```

休息一下~

遞迴(recursive)

- 函式可以被呼叫，當函式裡又呼叫自己時，這種情況稱為遞迴。
- 有些問題用遞迴可以得到更簡潔的程式，但不一定會有高效率。
- 使用遞迴時要注意：
  - 1.傳入的參數：要能符合要解決的問題。
  - 2.終止條件：錯誤或沒有終止條件，將造成無盡迴圈甚至當機。
- 經典題型：最大公因數(GCD)、費波納西數列(Fibonacci Sequence)、河內塔(Hanoi Tower)、N個字元的排列組合等。

遞迴(recursive)

- 求1 + 2 + 3 + ….. + N的和。

```
def sum(N):
    if N == 1:
        value = 1
    else:
        value = N + sum(N-1)
    return value

#main
x = int(input("計算1到N的總和，請輸入N:"))
y = sum(x)
print(f"1到{x}的總和為{y}")
```

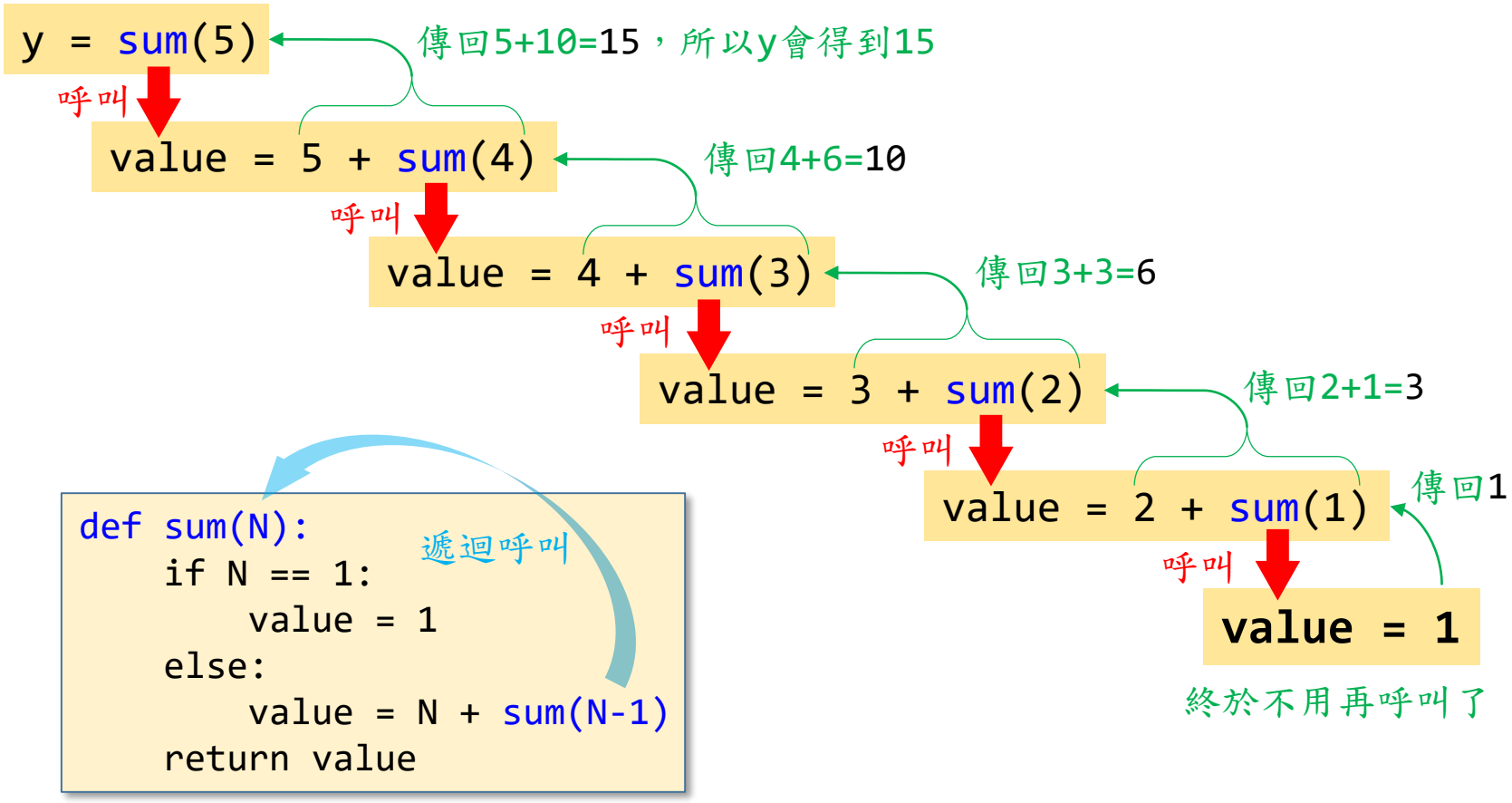
終止條件

結束呼叫，回傳確定值

傳遞參數，呼叫自己

遞迴(recursive)

- 圖解(假設輸入5)：



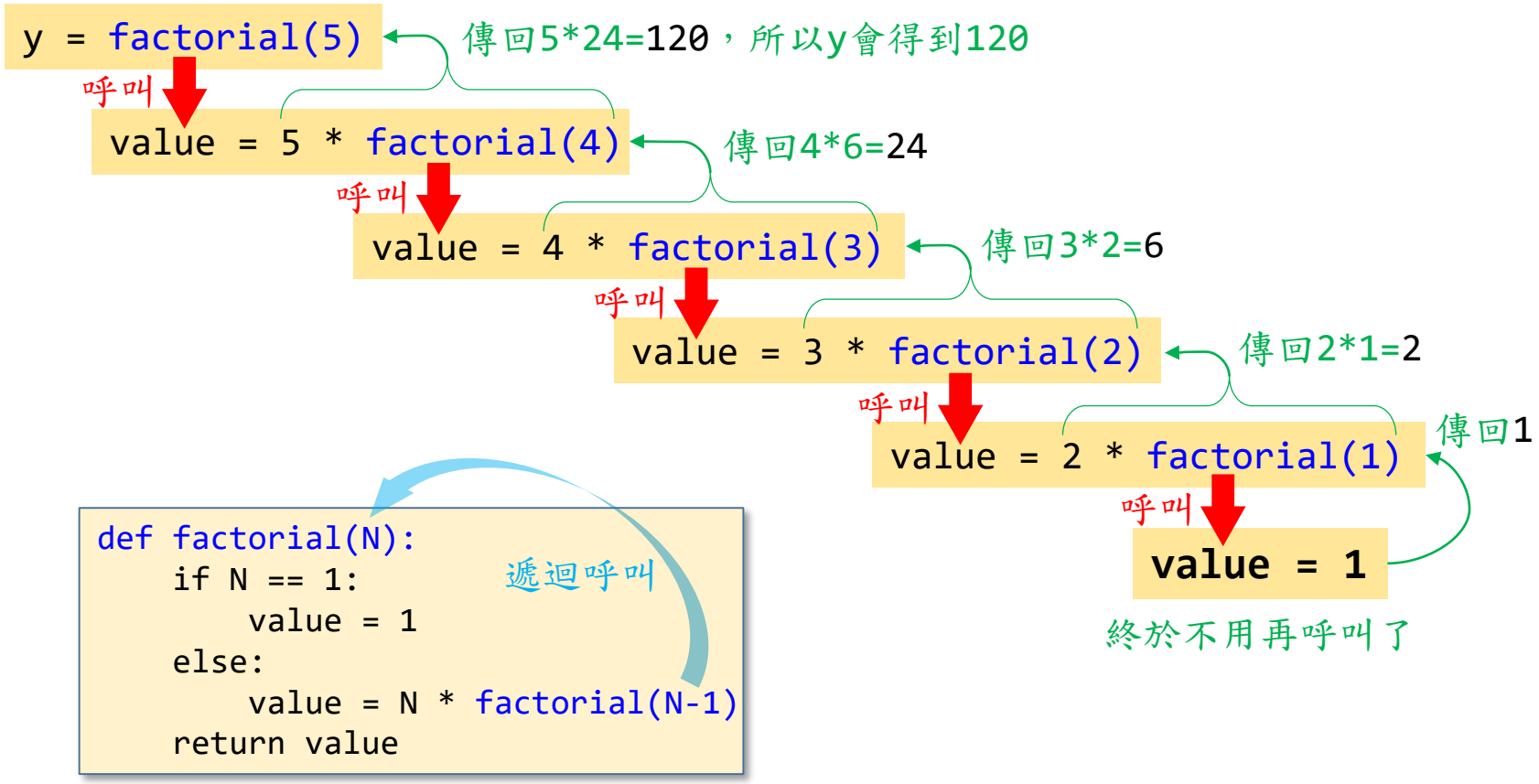
遞迴(recursive)

- 計算N!。

```
def factorial(N):
    if N == 1:
        value = 1
    else:
        value = N * factorial(N-1)
    return value

#main
x = int(input("要計算幾階層："))
y = factorial(x)
print(f"{x}! = {y}")
```

遞迴(recursive)

- 圖解(假設輸入5)：



遞迴(recursive)

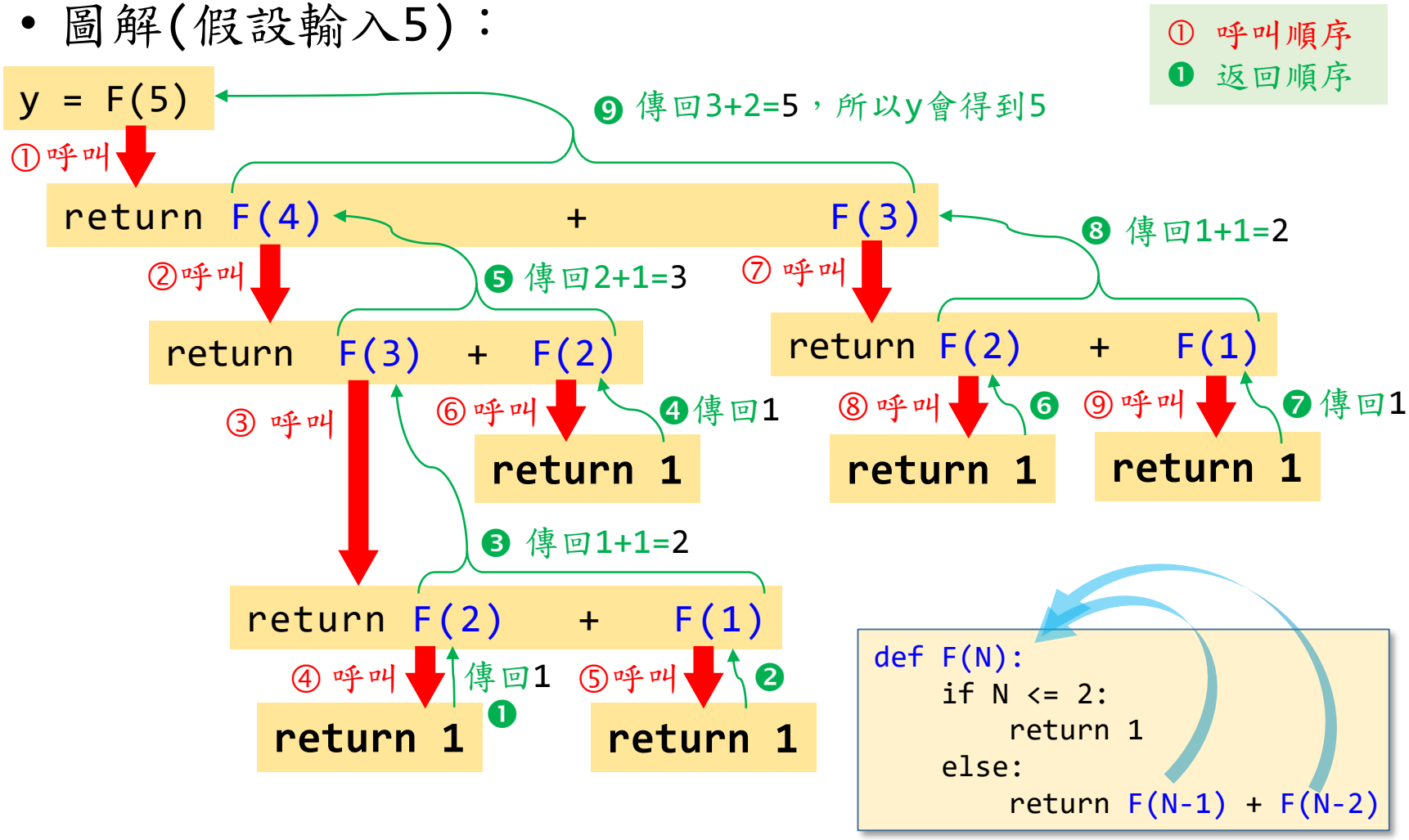
- 求費氏數列F(n)。

```
def F(N):
    if N <= 2:
        return 1
    else:
        return F(N-1) + F(N-2)

#main
x = int(input("請輸入求F(N)的N為:"))
y = F(x)
print(f"F({x}) = {y}")
```

終止條件

遞迴(recursive)

- 圖解(假設輸入5)：



練習題

- 10-1.以遞迴方式求 $1+b+b^2+\ldots+b^n$。
- 10-2.以遞迴方式求 $1\times2+2\times3+\ldots n(n+1)$。
- 10-3.以遞迴方式求
  $a+(a+b)+(a+2b)+\ldots.+(a+(n-1)b)$。
  (注意b的係數在增加)

練習題

- 10-1.以遞迴方式求$1+b+b^2+\ldots+b^n$。

```
def nF(b, n):
    if n == 0:
        return 1
    else:
        return b**n + nF(b, n-1)

#main
b = int(input("請輸入基底b:"))
n = int(input("請輸入次方n:"))
y = nF(b, n)
print("1+b+b**2+.....+b**n= "+str(y))
```

練習題

- 10-2.以遞迴方式求1×2+2×3+…n(n+1)。

```
def f(n):
    if n == 2:
        return 2
    else:
        return n*(n-1) + f(n-1)

#main
n = int(input("請輸入N:"))
x = f(n+1)
print("(1*2)+(2*3)+...+n*(n+1)= "+str(x))
```

練習題

- 10-3.以遞迴方式求 (注意b的係數在增加)

  a+(a+b)+(a+2b)+….+(a+(n-1)b)。

```
def f(a, b, n):
    if n == 0:
        return a
    else:
        return a+n*b + f(a, b, n-1)

#main
a = int(input("請輸入a:"))
b = int(input("請輸入b:"))
n = int(input("請輸入n:"))
x = f(a, b, n-1)
print("結果為:"+str(x))
```

休息一下~

模組與套件

- 在Python程式中可以輕易地加入由許多高手熱心設計的模組，使得許多複雜功能的程式，只要短短幾行就可以了，節省許多自行開發的時間與精力。

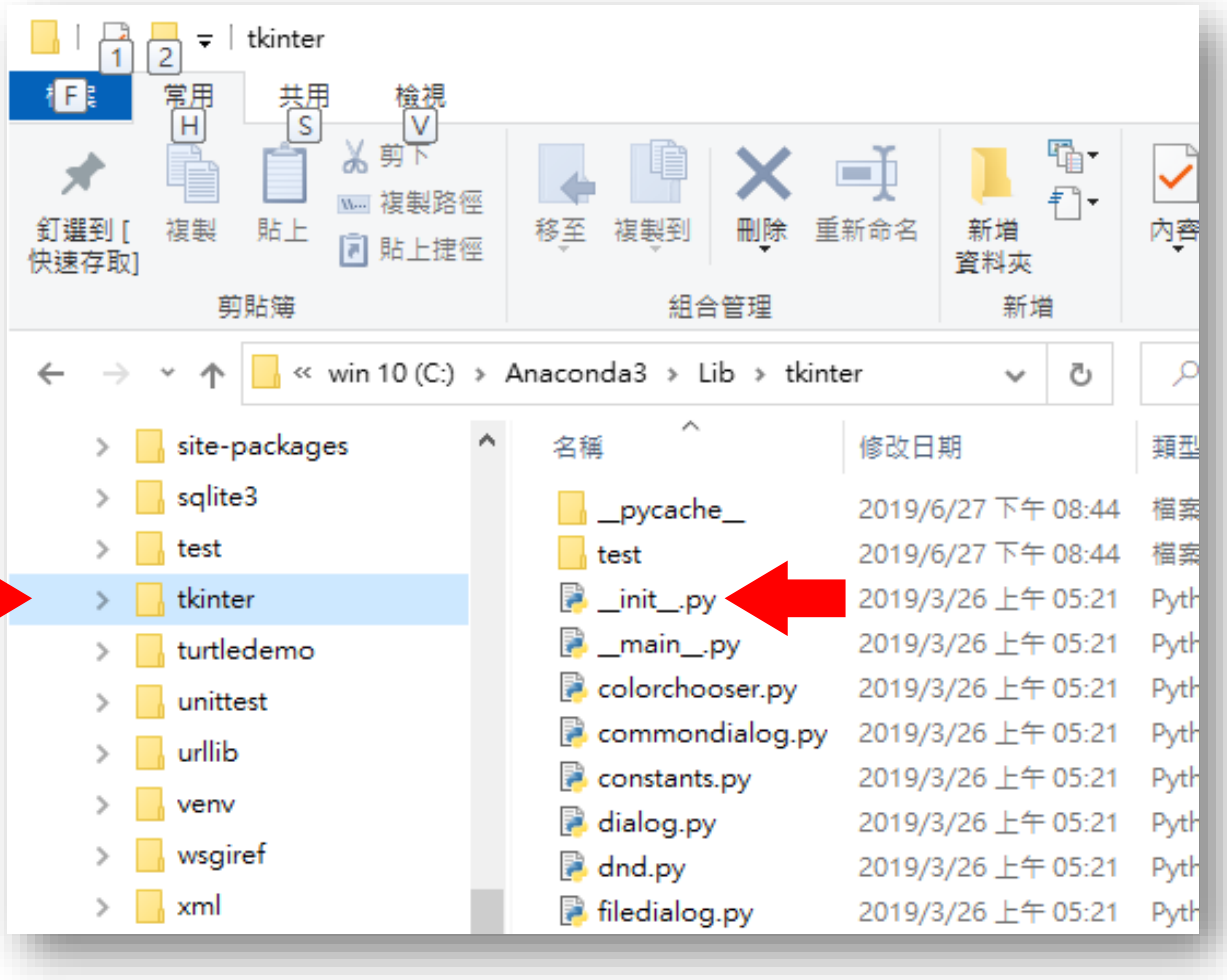
- 所謂模組指的是已經寫好的Python程式，也就是一個*.py檔案，模組中可以包含可執行的敘述、定義好的資料、函式、類別等。

- 許多個模組組合在一起就稱為「套件」(Package)，如果說模組是一個檔案，那套件就是一個資料夾。

模組與套件

- 擁有__init__.py檔案的資料夾就會被視為一個套件。

例如：



模組與套件

- __init__.py是Python套件(package)資料夾中的一個特殊檔案，用來讓 Python 把該資料夾視為一個「套件」。
- 它裡面可以是空的，也可以放程式碼，取決於你要它做什麼。
- 在套件被import時，__init__.py 會執行一次，所以你可以放一些初始化邏輯。

- 目前我們還不會自己建立套件，所以先知道一些概念就可以了。

模組與套件

- 匯入模組：
  - 使用import指令匯入模組。
  - 要放在使用到它提供的功能之前，通常會在程式最前面。
  - 除了Python本身附的套件外，如果有安裝Anaconda，也已包含了許許多多的套件，非常方便。

  - 匯入模組的方式：
    - 1. 匯入整個模組。
    - 2. 替模組取個別名。
    - 3. 只匯入特定函式。

模組與套件

- 匯入整個模組/套件：

```
import 模組 或 套件名稱
```

然後就可以用「模組名稱.函式名稱」來呼叫要使用的函式。

Ex：匯入亂數套件，使用randint()函式產生1~100之間的亂數。

```
import random  #匯入亂數套件
random.randint(1, 100)  #亂數套件提供的函式
```

- 或是一次全部匯入：

```
from 模組/套件名稱 import *
```

Ex：

```
from random import *
randint(1, 100)  #這時模組/套件名稱就可以省略不寫。
```

模組與套件

- 有時若覺得模組名稱太長，可以在匯入時給它一個別名。

```
import 套件名稱 as 別名
```

- 例：用r取代了random。

```
import random as r
r.randint(1, 100)
```

- 也可以一次匯入多個套件：

```
import 套件名稱1, 套件名稱2, ...
```

亂數套件

- Python的亂數產生需載入random套件：

```
import random
```

- 亂數起始種子(可以省略此敘述)

```
random.seed()
```

- 然後用下列敘述取得整數亂數：

```
X = random.randint(a, b)
```

傳回隨機整數X，其值介於： a <= X <= b

亂數套件

- 模擬擲骰子，隨機產生2個1~6之間的亂數。

```
import random as rd    #引用套件

x1 = rd.randint(1, 6)  #產生第一個亂數
x2 = rd.randint(1, 6)  #產生第二個亂數

print(x1, x2)    #印出內容
```

亂數套件

- 隨機產生5個1~40之間不重複的亂數，排序後印出。

```
import random as rd    #引用套件

num = set()     #宣告一個空set
while len(num) <= 5:
    x = rd.randint(1, 40)  #產生一個亂數
    #加入到set，若該數值已存在則不會加入
    num.add(x)

print(sorted(num))    #印出排序後結果
```

- 想想看，為甚麼要用set？

亂數套件

- 同前例，但使用random內建函式sample()。

```
import random as rd

#使用sample()自range()範圍內取出6個不同的數字
a = sorted(rd.sample(range(1, 41), 5))

#使用map()將a轉成字元後以join()連接後印出
print(','.join(map(str, a)))
```

練習題

- 8-1. 印出100個1~50之間的亂數，並統計你的幸運數字(假設是25)出現幾次。

- 8-2. 產生20個0~100之間的亂數，印出最小值及最大值、平均值。

- 8-3. 寫一個簡易猜數字大小的遊戲：產生一個1~100之間的亂數，請玩家猜大小，要提示猜得太大或太小了。

練習題

- 8-1.參考程式一：

```
import random as rd

m = 0
for i in range(100):
    n = rd.randint(1, 50)
    print(n, end=' ')
    if n == 25:
        m += 1

print("\n 25出現了 " , m , " 次")
```

練習題

- 8-1.參考程式二：

```
import random as rd

m = []
for i in range(100):
    m.append(rd.randint(1, 50))

print(m, "\n25出現了" , m.count(25) , "次")
```

計算清單m裡有多少個25

- 善用清單(list)及內建函式可以簡化程式。

練習題

- 8-2.參考程式：

```
import random as rd

n = []
for i in range(20):
    n.append(rd.randint(0, 100))

print("n =", n)
print("最大值: ", max(n))
print("最小值: ", min(n))
print("平均值: ", sum(n) / len(n))
```

練習題

- 8-3.參考程式：

```
import random

ans = random.randint(1, 100)
n = 0
while ans != n:
    n = int(input("請輸入數字: "))
    if n < ans:
        print(n, "太小了")
    if n > ans:
        print(n, "太大了")
    if n == ans:
        print(n, "答對了!")
```

休息一下~

檔案讀寫

- 關於檔案，要注意下面幾點：
  - 要使用磁碟內的檔案，必須先將檔案內容複製到記憶體，假設檔案叫做test.txt，要讀取這個檔案，使用以下指令：

```
f = open("test.txt", "r")
```

  - open是開啟檔案的指令。
  - f可自行命名，它僅是一個資源代碼，程式結束或關機就不存在了。
  - 必要時須加上路徑名稱，否則會找不到檔案。
  - 在程式結束的地方或不再使用此檔時，需下達關閉檔案的指令。

```
f.close()
```

檔案讀寫

- 檔案開啟模式參數：

參數	說　明
r	讀取模式(預設)。
w	寫入模式，會先清空檔案內容。
x	只在檔案不存在時才建立新檔，並開啟為寫入模式，若檔案已存在會引發FileExistsError錯誤。
a	附加模式，若檔案已存在，寫入的內容會附加至檔案尾端。
b	二進位模式。

檔案讀寫

- 常用檔案處理函式：

函式	說明
flush( )	將緩衝區的資料寫入檔案中，然後清除緩衝區內容。
next(檔案變數)	將檔案指標移到下一行。
read([size])	依指定的 size 大小讀取檔案，如未指定 size，則讀取檔案所有字元。
readable( )	測試檔案是否可讀取，可讀取回傳 True，不可讀取回傳 False。
readline([size])	依指定的 size 大小讀取所在行，如未指定 size，則讀取一整行。
readlines( )	讀取所有行的內容，會回傳一個串列。
seek(位址)	移動檔案指標到指定的位址，seek(0) 為移動至檔案的開頭。
tell( )	傳回檔案指標在文件中的位址。
writable( )	測試檔案是否可寫入，可寫入回傳 True，不可寫入回傳 False。
write(字串)	將字串內容寫入檔案之中。

檔案讀寫

- 讀取檔案並印出資料。

```
f = open("test.txt", "r")

for line in f:
    print(line)

f.close()
```

以唯讀方式開啟test.txt，暫存檔名稱為f

For迴圈會自檔案f中一次讀入一行放入line變數，然後用print()印出來

程式結束要關閉檔案

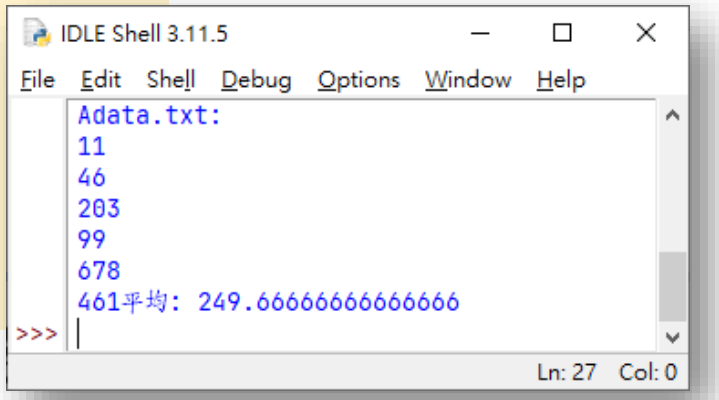
檔案讀寫

- 讀取一個檔案內的數字，求其平均值。

```
sum = 0
flen = 0

f = open("Adata.txt", "r")
print("Adata.txt:")

for line in f:
    sum += int(line)
    print(line, end='')
    flen += 1

print("平均:", sum/flen)
f.close()
```

Adata.txt內容

```
11
46
203
99
678
461
```



檔案讀寫

- 如果要讀寫的檔案不在目前的資料夾裡，就要把路徑也寫出來。
- 例：

```
f = open("C:/test/Adata.txt", "r")
```

路徑　檔名

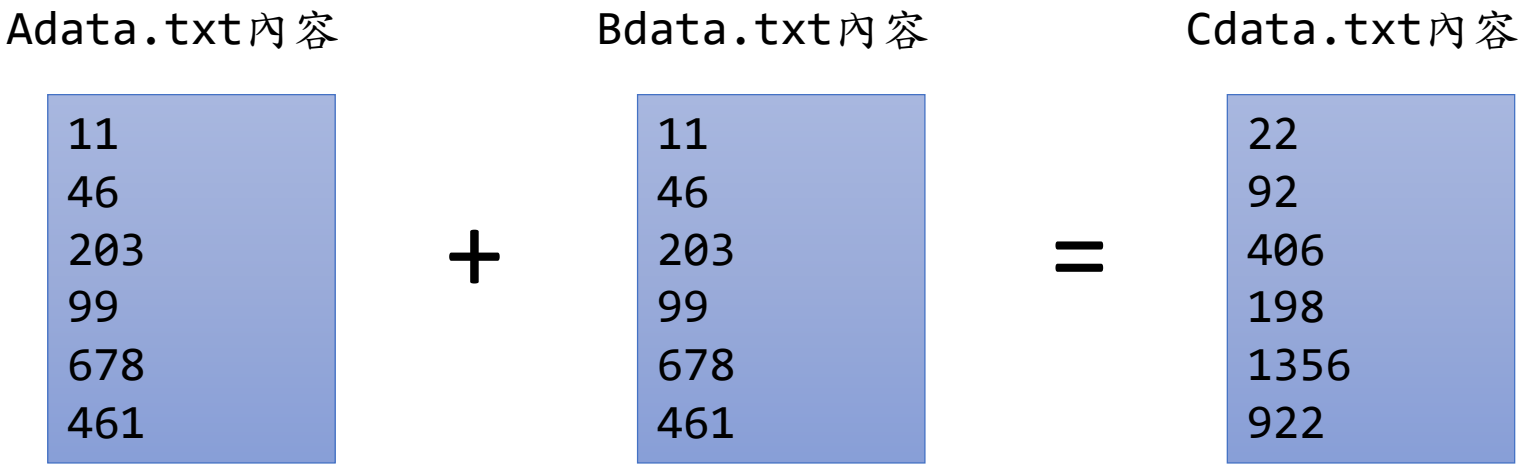
- 在Windows及UNIX下皆可使用"/"路徑符號。
- 若要使用"\"符號，需寫成下列方式，以免被認為是"\t"跳脫字元。

```
f = open("C:\\test\\Adata.txt", "r")
```

路徑　檔名

檔案讀寫

- 將兩個檔案中的數字加起來，再將結果存入第三個檔案。(為方便測試，Adata.txt與Bdata.txt內容相同)。
- f.readline() 可以一次讀入一整行。
- f.write() 是將資料寫入檔案。



檔案讀寫

- 參考程式：

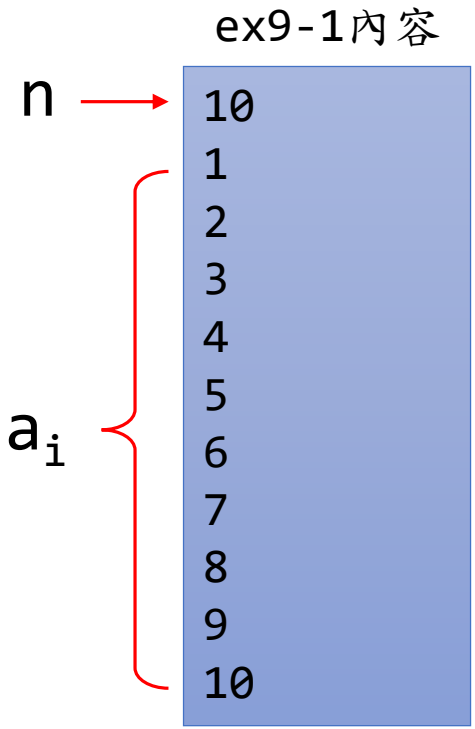
```
def open_and_print_file(filename):
    f = open(filename, "r")
    print(filename)
    for line in f:  #印出檔案內容
        print(int(line))
    f.close()

def addfile(x, y, z):
    f1 = open(x, "r")  #r是讀入檔案
    f2 = open(y, "r")
    f3 = open(z, "w")  #w是寫入檔案
    u = f1.readline()  #一次讀取一行
    v = f2.readline()  #一次讀取一行
    while u and v:
        sum = int(u) + int(v)
        f3.write(str(sum)+"\n")
        u = f1.readline()
        v = f2.readline()
    f1.close()
    f2.close()
    f3.close()
```

```
#main
open_and_print_file("Adata.txt")
open_and_print_file("Bdata.txt")
addfile("Adata.txt","Bdata.txt","Cdata.txt")
open_and_print_file("Cdata.txt")
```

任一檔案讀不到東西時(u或v是空值時)，迴圈便會結束。

練習題

- 9-1. 寫一程式，從檔案中讀入n及a_1，a_2，…a_n。n和所有的a_i都是正整數，計算出a_1^2，a_2^2，…a_n^2。再將a_i，a_i^2，i=1到i=n寫到另一個檔案上。



ex9-1a內容

1	1
2	4
3	9
4	16
5	25
6	36
7	49
8	64
9	81
10	100

練習題

- 9-1.參考程式：

```
f1 = open("ex9-1.txt", "r")
f2 = open("ex9-1a.txt", "w")

n = int(f1.readline())
for i in range(n):
    u = int(f1.readline())
    v = u**2
    print(str(u)+"\t"+str(v)) #顯示在螢幕
    f2.write(str(u)+"\t"+str(v)+"\n") #寫入檔案

f1.close()
f2.close()
```

檔案讀寫

- 另一種開啟檔案的方式：

```
with open(檔名) as 檔案物件:
        :
    相關指令
        :
```

- 例：

```
with open("ex9-1.txt", "r") as fn:
    for line in fn:
        print(line.rstrip())
```

- 它的好處是with指令結束不必另外關閉檔案，它會自動關閉。

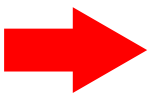
檔案讀寫

- 使用with改寫練習9-1：

```
with open('ex9-1.txt') as f1, \  #可一次開啟多個檔案
     open('ex9-1a.txt', 'w') as f2 :
    for i in f1:
        v = int(i) ** 2
        print(int(i), '\t', v)
        f2.write(i.rstrip() + '\t' + str(v) + '\n')
```

ex9-1內容

```
1
2
3
4
5
6
7
8
9
10
```



ex9-1a內容

```
1       1
2       4
3       9
4       16
5       25
6       36
7       49
8       64
9       81
10      100
```

檔案讀寫

- 更完整的寫法：
  - 因為檔案名稱、位置、權限、屬性等都可能造成檔案開啟失敗，這時with會丟出錯誤訊息並中斷程式，我們應該自己去捕捉並解決這些錯誤，以避免程式中斷。

```
try:
    with open('ex9-1.txt') as f1, \
         open('ex9-1a.txt', 'w') as f2 :
        for i in f1:
            v = int(i) ** 2
            print(int(i), '\t', v)
            f2.write(i.rstrip() + '\t' + str(v) + '\n')
except:
    print('檔案開啟錯誤')
```

練習題

- 9-3. 寫一程式，從檔案中讀入一個3x5的矩陣，求此矩陣的transpose(轉置)，然後將此結果寫到另一個檔案。檔案內數字以跳格符號(Tab)分開。

ex9-3內容

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15

ex9-3a內容

1	6	11
2	7	12
3	8	13
4	9	14
5	10	15

練習題

- 9-3.參考程式：

```
f1 = open("ex9-3.txt", "r")    #開啟檔案
f2 = open("ex9-3a.txt", "w")
A = [[0]*5 for i in range(3)]  #宣告3x5清單
B = [[0]*3 for i in range(5)]  #宣告5x3清單

def print_matrix(x, a, b):  #印出清單函式
    for i in range(a):
        for j in range(b):
            print(str(x[i][j])+"\t", end="")
        print("")

#讀取檔案並放入矩陣A
i = 0
for line in f1:
    line = line.replace("\n", "")   #將換行符號消去
    A[i] = line.split("\t")  #以跳格當分界，將數字存入清單
    i += 1
print_matrix(A, 3, 5)
print("")    #印一個換行
```

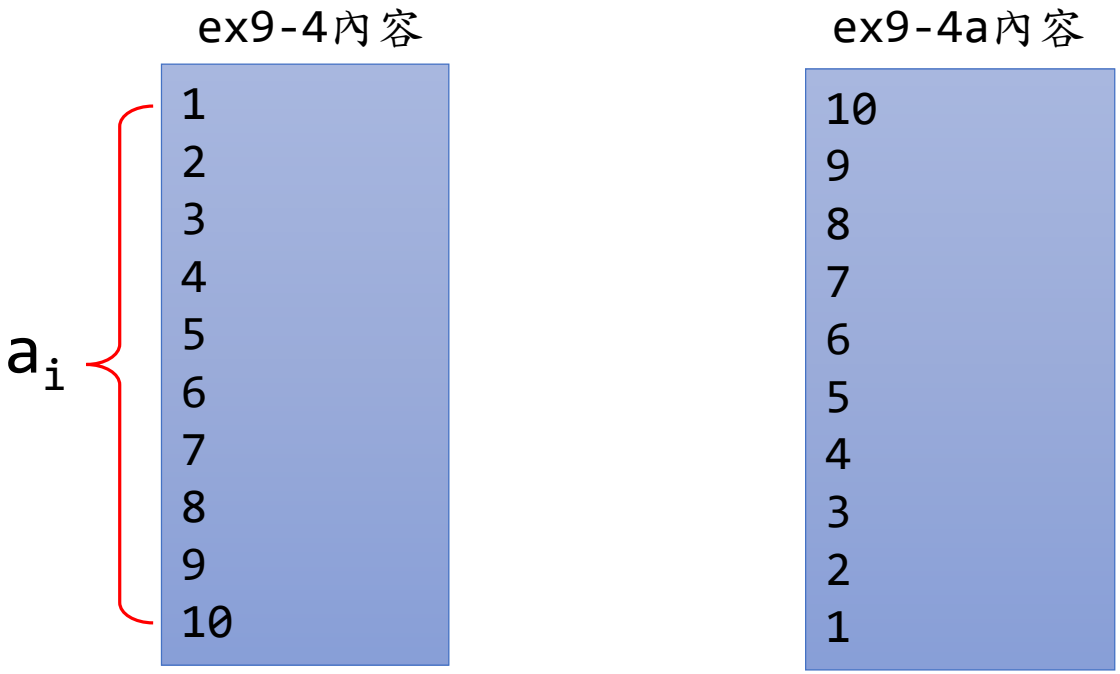
練習題

- 9-3.參考程式(續)：

```
#轉換矩陣A至矩陣B，並寫入檔案
for j in range(5):
    s = ""
    for i in range(3):
        B[j][i] = A[i][j]
        s += str(B[j][i]) + "\t"
    f2.write(s + "\n")

print_matrix(B, 5, 3)   #顯示置換後結果

f1.close()  #關閉檔案
f2.close()
```

練習題

- 9-4. 寫一程式，從檔案中讀入a_1，a_2，…a_n。所有的a_i都是正整數，將a_n，a_{n-1}，…，a_1寫到另一個檔案上去(就是將順序倒過來)。



練習題

- 9-4.參考程式：

```
with open("ex9-4.txt") as f1, \
     open("ex9-4a.txt", "w") as f2:

    A = []
    for i in f1:    #一次一行讀入檔案
        A.append(i.rstrip())

    for i in A[::-1]:    #倒過來寫入檔案
        f2.write(i + "\n")
```

檔案操作

- 除了檔案內容的讀寫，還有對資料夾(目錄)及檔案本身的相關操作，例如更改檔名、檢查是否為檔案或資料夾等。
- 我們需要使用Python的os及glob模組：
  - OS:最基礎的操作，包山包海，但使用起來較為繁瑣。
  - glob:非常簡單的檔案搜尋，可使用正規表示式。

```
#基本檔案、路徑功能都在這裡面
import os
#如果需要使用到萬用字元*和?就需要這個
import glob

#或直接：
import os, glob
```

檔案操作

- os模組常用函式：

方法	參數	說明
getcwd()		取得目前程式的工作資料夾路徑
chdir()	path	改變程式的工作資料夾路徑
mkdir()	folder	建立資料夾
rmdir()	folder	刪除空資料夾
listdir()	folder	列出資料夾裡的內容
open()	file, mode	開啟檔案
write()	string	寫入內容到檔案
rename()	old, new	重新命名檔案
remove()	file	刪除檔案
stat()	file	取得檔案的屬性
close()	file	關閉檔案
path		取得檔案的各種屬性
system		執行系統命令 (等同使用 cmd 或終端機輸入指令)

檔案操作

- os.path模組常用函式：

函式	說明
abspath( )	取得檔案的絕對路徑。
basename( )	取得路徑最後的檔案名稱。
dirname( )	取得檔案的目錄路徑，如要取得目前 Python 檔案所在的目錄路徑，其語法為 os.path.dirname(__file__)。
exists( )	檢查檔案或路徑是否存在，其回傳值為 True 或 False。
getsize( )	取得檔案的大小，其回傳單位是 Bytes。
isabs( )	檢查該路徑是否為完整路徑。
isdir( )	檢查該路徑是否為目錄。
isfile( )	檢查該路徑是否為檔案。
join( )	合併檔案路徑和檔案名稱。
split( )	分割該路徑為目錄路徑與檔案名稱。
splitdrive( )	分割該路徑為磁碟名稱與檔案路徑。

檔案操作

- 檢查檔案是否存在再開啟，避免開啟錯誤：

```
import os

if os.path.exists("E:/TmpE/ex9-9.txt"):
    with open("E:/TmpE/ex9-9.txt") as f1:
            :
            :
            :
else:
    print("檔案E:/TmpE/ex9-9.txt不存在")
```

- exists()如果檔案存在則傳回真，否則傳回假。

檔案操作

- 目錄的操作範例：

這些內建函式使用上都很直覺且簡單。

```
import os

#建立目錄
path = 'C:/tmp'
if os.path.exists(path):
    print(path + '已存在')
else:
    os.mkdir(path)
    print(path + '已建立')

#重新命名
new_path = 'C:/new_tmp'
os.rename(path, new_path)
print('已重新命名為' + new_path)

#刪除目錄(需為空目錄)
os.rmdir(new_path)
print(new_path + '已刪除')
```

檔案操作

- 目錄的操作範例：

要使用萬用字元需glob模組。

```
import os

#列出資料夾內容
path = 'D:/'
for file in os.listdir(path):
    print(file)

#列出指定資料夾下所有副檔名為txt的檔案
import glob
fname = '*.txt'
for file in glob.glob(path + fname):
    print(file)
```

休息一下~

圖形化使用者介面(GUI)

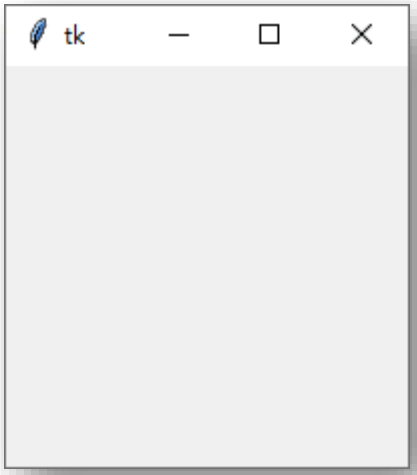
- 由於在視窗環境下，視窗程式有使用上的便利性，所以Python也有套件支援寫出簡單的視窗程式。
- 通常會使用tkinter套件(代表Tk interface)，是Python中最基本的Tk圖形化工具標準模組，在多數的UNIX/Linux、Mac OS以及Windows系統都可以使用。
- tkinter仍算簡陋，如有興趣可另外參考PyQt5等套件。
- 目前Python沒有像Visual Basic那樣有方便的IDE來寫視窗程式，也許在發展時就不是以寫視窗程式為主吧，以後看看會不會有更方便的環境來寫Python視窗程式。

圖形化使用者介面(GUI)

- 使用時匯入tkinter套件即可。

```
import tkinter as tk  #以tk取代tkinter

win = tk.Tk()  #建立視窗
win.mainloop()     #執行視窗直到關閉
```

我們自己命名叫win，稱為「容器物件」



- 或

```
from tkinter import *  #此方式之後可省略套件名稱

win = Tk()         #建立視窗
win.mainloop()     #執行視窗直到關閉
```

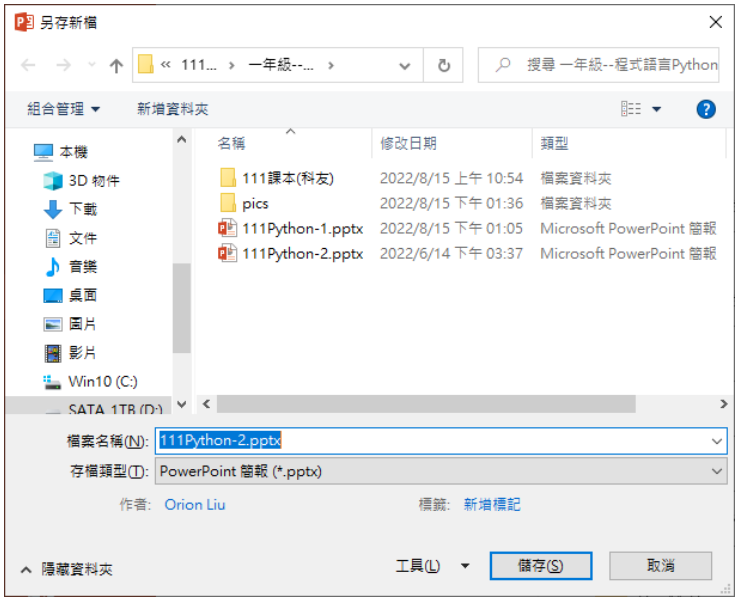
圖形化使用者介面(GUI)

- 建立一個大小為500x300的視窗：

```
import tkinter as tk
#呼叫Tk()建立一個名稱為win的視窗，注意T是大寫，k是小寫
win = tk.Tk()
win.title("我的GUI視窗")          #視窗標題
win.geometry("500x300")          #視窗大小，注意"500x300"是字串
win.mainloop()                   #執行視窗
```

圖形化使用者介面(GUI)

- 跟其它軟體工具一樣(例如微軟Visual Studio的VB)，tkinter套件也提供了一些基礎元件。
- 基礎元件：
  - 標籤(Label)
  - 按鈕(Button)
  - 文字方塊(Entry)
  - 文字區域(Text)
  - 捲軸(Scrollbar)
  - 核取按鈕(Checkbutton)
  - 選項按鈕(Radiobutton)
  - 圖形(Photoimage)
  - 功能表(Menu)



視窗範例：
這些元件常出現在視窗裡

圖形化使用者介面(GUI)

- 基礎元件 - 標籤(Label)：
  - 語法：`tkinter.Label(容器物件, 參數1=值1, 參數2=值2, ...)`
  - 參數：

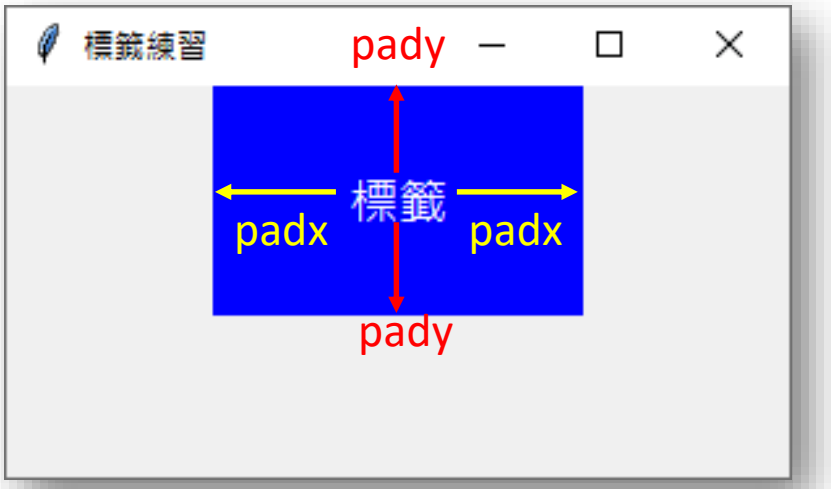
參數	說明
text	標籤文字
width	標籤寬度
height	標籤高度
background	標籤的背景顏色，簡稱 bg
foreground	標籤的文字顏色，簡稱 fg
padx	標籤文字與標籤物件邊緣的水平間距
pady	標籤文字與標籤物件邊緣的垂直間距
justify	對齊方式，有靠左（Left）、置中（Center）、靠右（Right）
font	標籤文字字體與大小，例如：font=('新細明體',14)

圖形化使用者介面(GUI)

- 基礎元件 - 標籤(Label)：

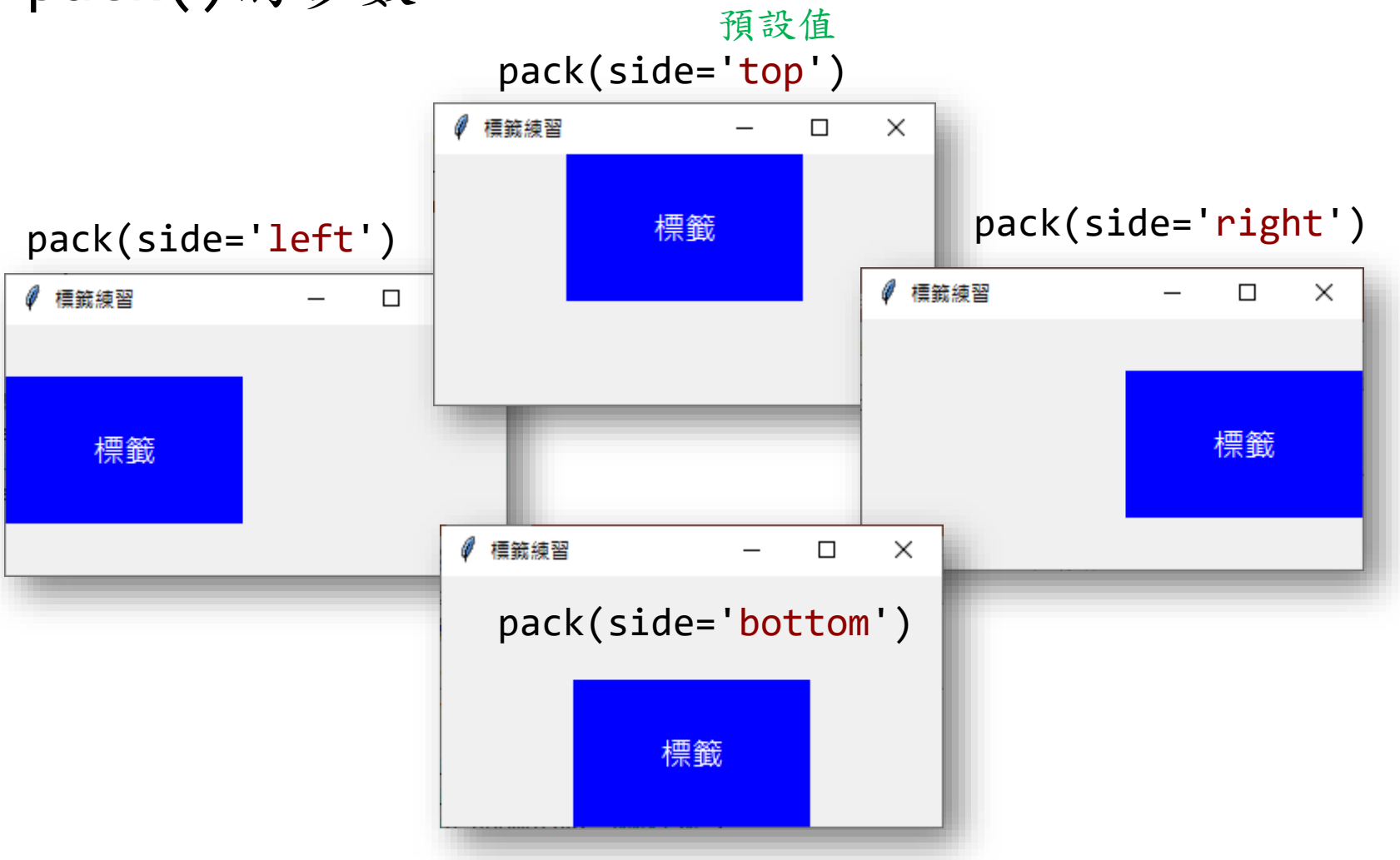
```
import tkinter as tk
win = tk.Tk()
win.title("標籤練習")
win.geometry("300x150")
label = tk.Label(win, text='標籤', bg='blue',
        fg='white', font=('微軟正黑體',14), padx=50, pady=30)
label.pack()  #配置label到win
win.mainloop()
```

- 執行結果：



圖形化使用者介面(GUI)

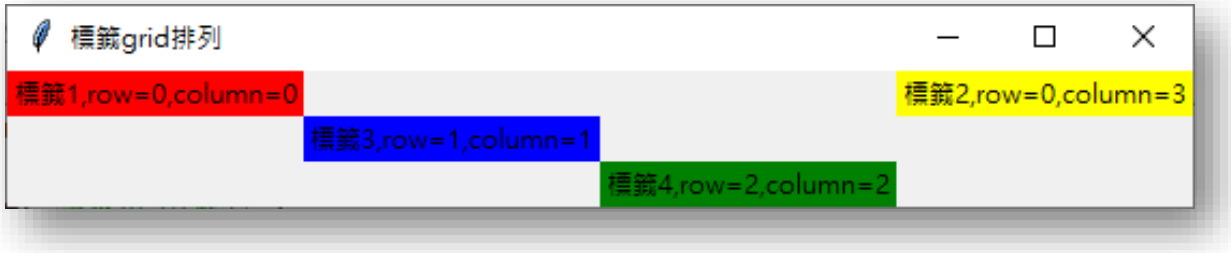
- pack()的參數：



圖形化使用者介面(GUI)

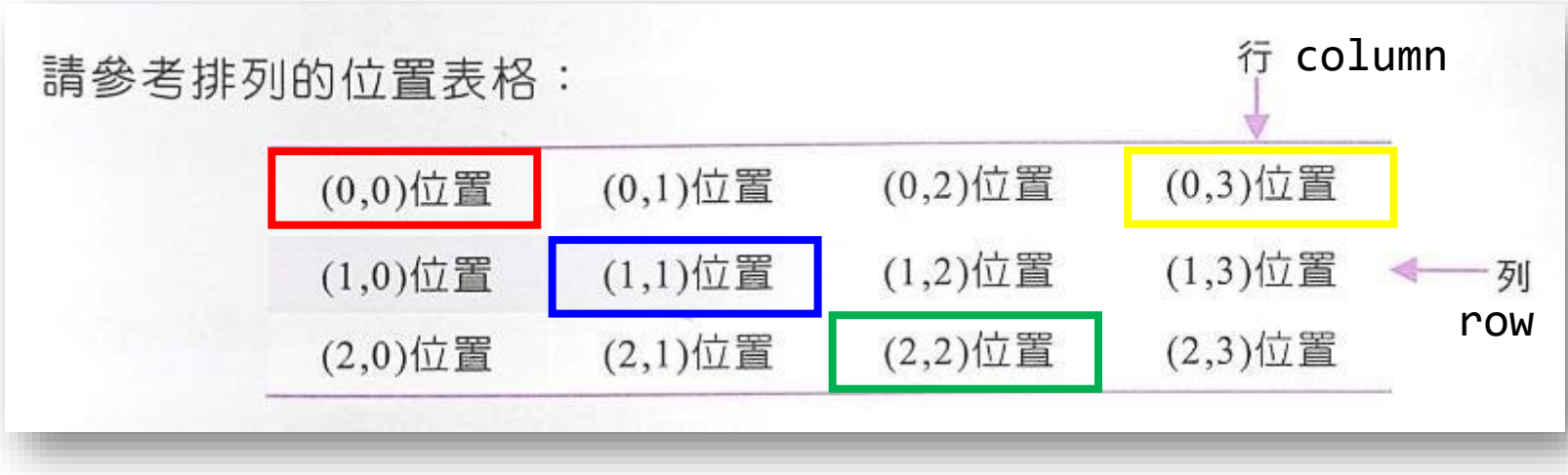
- 使用grid()來安排標籤位置。

```
from tkinter import *
win = Tk()
win.title("標籤grid排列")
label1 = Label(win, text="標籤1,row=0,column=0", bg="red")
label2 = Label(win, text="標籤2,row=0,column=3", bg="yellow")
label3 = Label(win, text="標籤3,row=1,column=1", bg="blue")
label4 = Label(win, text="標籤4,row=2,column=2", bg="green")
label1.grid(row=0, column=0)
label2.grid(row=0, column=3)
label3.grid(row=1, column=1)
label4.grid(row=2, column=2)
win.mainloop()
```



圖形化使用者介面(GUI)

- grid()的位置：
  - 根據row和column的座標來放置標籤。
  - 前例的四個label的位置：



圖形化使用者介面(GUI)

- 基礎元件 - 按鈕(Button)：
  - 語法：`tkinter.Button(容器物件, 參數1=值1, 參數2=值2, ...)`
  - 參數：

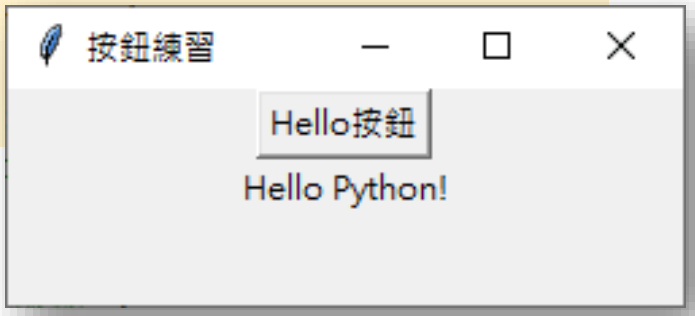
參數	說明
text	按鈕文字
width	按鈕寬度
height	按鈕高度
background	按鈕的背景顏色，簡稱 bg
foreground	按鈕的文字顏色，簡稱 fg
padx	按鈕文字與按鈕物件邊緣的水平間距
pady	按鈕文字與按鈕物件邊緣的垂直間距
justify	對齊方式，有靠左（Left）、置中（Center）、靠右（Right）
font	按鈕文字字體與大小，例如：font=('新細明體',14)
command	當使用者按下按鈕時，呼叫 command 所指定的函式
textvariable	按鈕文字之變數，可用作設定或取得按鈕的文字內容
underline	按鈕文字加上底線，預設值為-1，代表全部不加底線，0 表示第 1 個字元，1 表示第 2 個字元，2 表示第 3 個字元，依此類推

圖形化使用者介面(GUI)

- 基礎元件 - 按鈕(Button)：

```
from tkinter import *

def HelloMsg():
    label["text"] = "Hello Python!"

win = Tk()
win.title("按鈕練習")
win.geometry("250x80")
btn = Button(win, text="Hello按鈕", command=HelloMsg)
label = Label(win)  #label一開始並未指定文字
btn.pack()  #配置按鈕
label.pack()  #配置標籤
win.mainloop()
```

按下按鈕後執行指定的函式



圖形化使用者介面(GUI)

- 基礎元件 - 按鈕(Button)：
  - 如果要向呼叫的函式傳遞參數，有兩種方法：
  - 1.使用lambda命令：

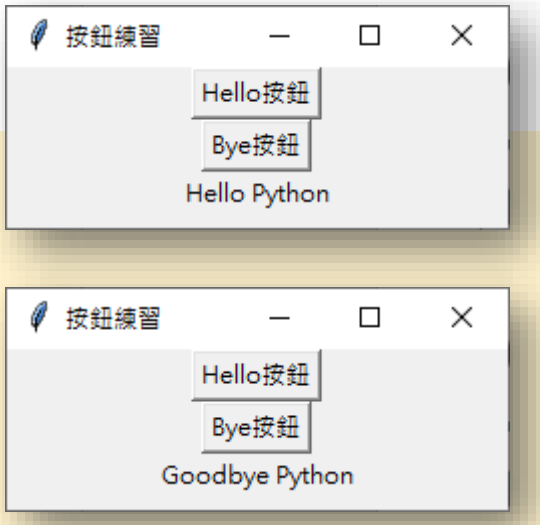
```
     :
def HelloMsg(x):
    label["text"] = x
     :
btn = Button(win, text = "Hello按鈕",
             command = lambda:HelloMsg("Hello Python"))
```

  - 2.使用partial()函式：

```
from functools import partial
        :
def HelloMsg(x):
    label["text"] = x
        :
btn = Button(win, text = "Hello按鈕",
             command = partial(HelloMsg, "Hello Python"))
```

圖形化使用者介面(GUI)

- 基礎元件 - 按鈕(Button)：
  - 按鈕傳遞參數示範：




```
from tkinter import *

def HelloMsg(x):
    label["text"] = x

win = Tk()
win.title("按鈕練習")
win.geometry("250x80")
btn1 = Button(win, text = "Hello按鈕",
              command = lambda:HelloMsg("Hello Python")).pack()
btn2 = Button(win, text = "Bye按鈕",
              command = lambda:HelloMsg("Goodbye Python")).pack()
label = Label(win)
label.pack()
win.mainloop()
```

圖形化使用者介面(GUI)

- 基礎元件 - 文字方塊(Entry)：
  - 是用來讓使用者輸入資料的元件。
  - 語法：`tkinter.Entry(容器物件, 參數1=值1, 參數2=值2, ...)`
  - 參數：

參數	說明
text	文字方塊文字
width	文字方塊寬度
background	文字方塊的背景顏色，簡稱 bg
foreground	文字方塊的文字顏色，簡稱 fg
state	文字方塊的輸入狀態，預設值是 normal；如為 disabled 則無法輸入；如為 readonly 則為唯讀
textvariable	文字方塊文字之變數，可用作設定或取得文字方塊的文字內容

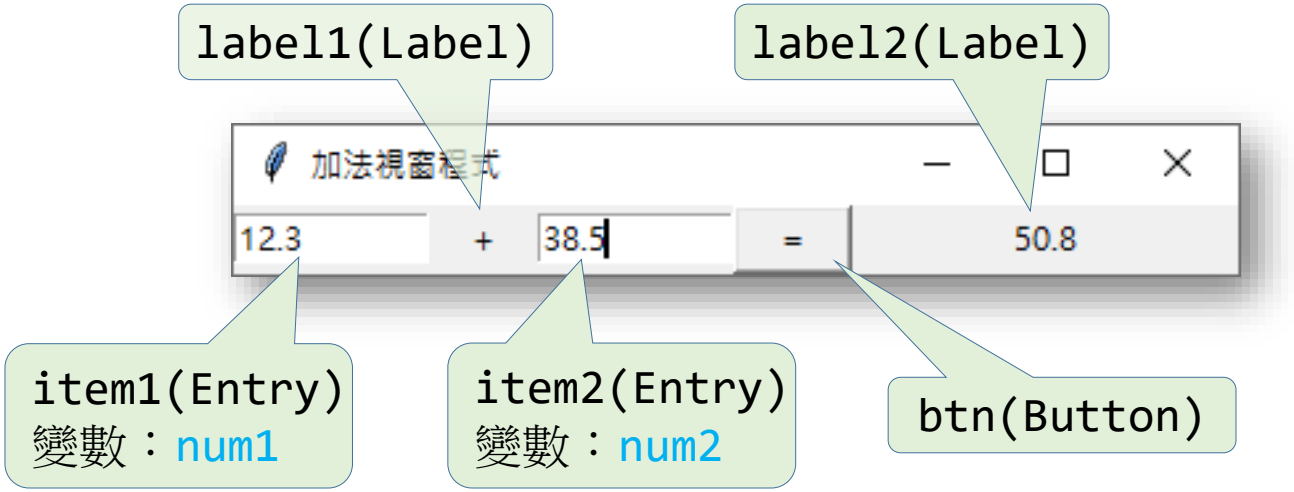
圖形化使用者介面(GUI)

- 基礎元件 - 文字方塊(Entry)：
  - 一個簡易加法視窗程式：

```
import tkinter as tk
def add_num():
    label2["text"] = str(round(num1.get() + num2.get(), 4))
win = tk.Tk()
win.title('加法視窗程式')
num1 = tk.DoubleVar()   #設定num1和num2為Entry元件輸入儲存資料用
num2 = tk.DoubleVar()
result = 0
item1 = tk.Entry(win, width=10, textvariable=num1).pack(side='left')
label1 = tk.Label(win, width=5, text='+').pack(side='left')
item2 = tk.Entry(win, width=10, textvariable=num2).pack(side='left')
btn = tk.Button(win, width=5, text='=', command=add_num).pack(side='left')
label2 = tk.Label(win, width=20)
label2.pack(side='left')
win.mainloop()
```

圖形化使用者介面(GUI)

- 基礎元件 - 文字方塊(Entry)：
  - 執行結果：



  - Entry的輸入會存入變數num1和num2，再用get()方法取出。
  - 由於Double型態的二進制會有誤差，所以用round()取四捨五入。(你可以試試3.3 + 6.6 = ?)

```
>>> 3.3+6.6
9.899999999999999
```

圖形化使用者介面(GUI)

- 基礎元件 - 文字方塊(Entry)：
- textvariable 參數有四種型態可設定：
  - tk.StringVar()：資料型態為「字串」，預設值為空字串。
  - tk.IntVar()：資料型態為「整數」，預設值為 0。
  - tk.DoubleVar()：資料型態為「浮點數」，預設值為 0.0。
  - tk.BooleanVar()：資料型態為「布林」，預設值為 0(假)。

- 設定與取得動態變數值：
  - textvariable變數.set()：設定動態文字變數內容。
  - textvariable變數.get()：取得動態文字變數內容。

圖形化使用者介面(GUI)

- 基礎元件 - 文字區域(Text)：
  - 語法：`tkinter.Text(容器物件, 參數1=值1, 參數2=值2, ...)`
  - 參數：

參數	說明
width	文字區域寬度
height	文字區域高度
background	文字區域的背景顏色，簡稱 bg
foreground	文字區域的文字顏色，簡稱 fg
state	文字區域的輸入狀態，預設值是 normal；如為 disabled 則無法輸入；如為 readonly 則為唯讀
padx	文字區域的文字與文字物件邊緣的水平間距
pady	文字區域的文字與文字物件邊緣的垂直間距
wrap	文字換行的方式，預設值是「char」，當文字超過文字區域的寬度時，會切斷單字進行換行；如參數值為「word」，則不會切斷單字換行；如參數值為「none」，則不換行，但必須搭配開啟水平捲軸
xscrollcommand	水平捲軸
yscrollcommand	垂直捲軸

圖形化使用者介面(GUI)

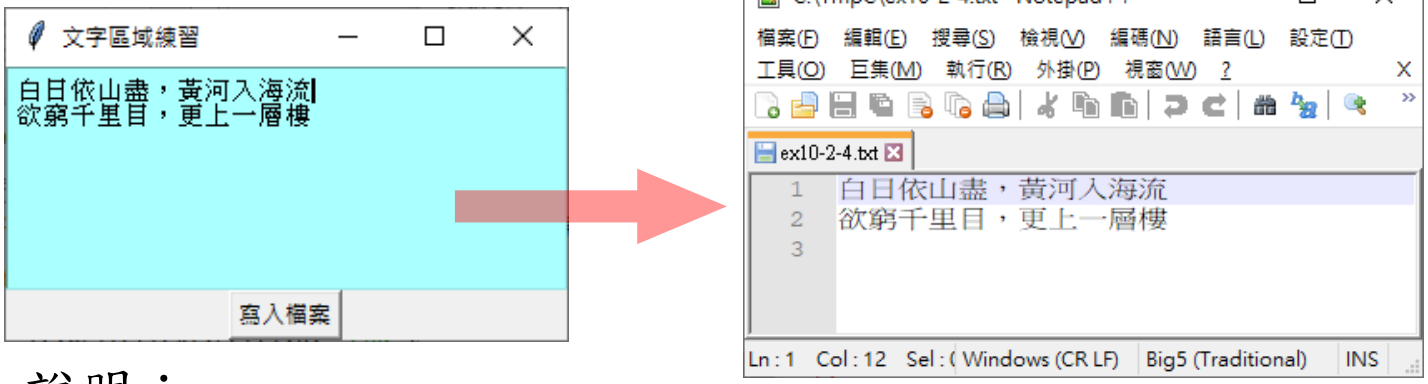
- 基礎元件 - 文字區域(Text)：
  - 練習，將文字區域輸入的內容寫入檔案。

```
import tkinter as tk

def TextToFile():
    f2 = open("ex10-2-4.txt", "w")
    f2.write(text.get(1.0, "end"))  #寫入指定檔案
    f2.close()

win = tk.Tk()
win.title('文字區域練習')
txt = ""
text = tk.Text(win, width=40, height=8, padx=5, pady=5, bg='#AAFFFF')
text.insert(1.0, txt)   #將游標至於起始處
text.pack()
btn = tk.Button(win, text="寫入檔案", command=TextToFile).pack()
win.mainloop()
```

也可以用rgb方式表示色彩

圖形化使用者介面(GUI)

- 基礎元件 - 文字區域(Text)：
  - 執行結果：



  - 說明：
    - text.get(起始位置, 結束位置)，取得文字區域內容：
      - 元件中第一個字元的位置是 1.0，可以用數字 1.0 或字串 "1.0"來表示。"end"表示它將讀取直到文字框的結尾的輸入。例：text.get(1.0, "end")
    - text.insert(位置, 字串)，插入文字至文字區域：
      - text.insert("insert", 字串)：在目前游標位置插入字串。
      - text.insert("end", 字串)：在結尾處插入字串。

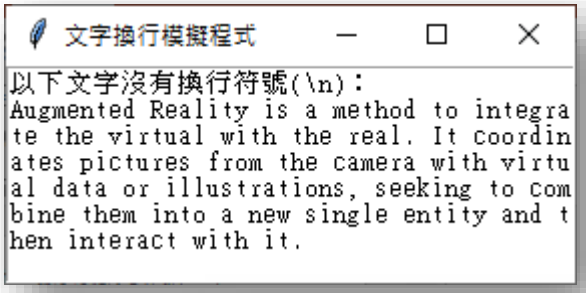
圖形化使用者介面(GUI)

- 基礎元件 - 文字區域(Text)：
  - 文字換行的模式：

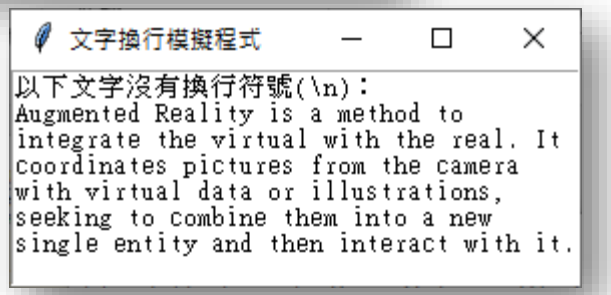
```
import tkinter as tk
txt='Augmented Reality is a method to integrate the virtual with the real. \
It coordinates pictures from the camera with virtual data or illustrations, \
seeking to combine them into a new single entity and then interact with it.'
win = tk.Tk()
win.title('文字換行模擬程式')
choice = input('換行模式(1:依文字區域寬度換行 2:依單字換行 3:不換行)：')
if(choice == '1'):
    text = tk.Text(win, width=40, height=8, wrap='char')
elif(choice == '2'):
    text = tk.Text(win, width=40, height=8, wrap='word')
elif(choice == '3'):
    text = tk.Text(win, width=40, height=8, wrap='none')
text.insert('end', '以下文字沒有換行符號(\\n)：\n')
text.insert('end', txt)
text.pack()
win.mainloop()
```

圖形化使用者介面(GUI)

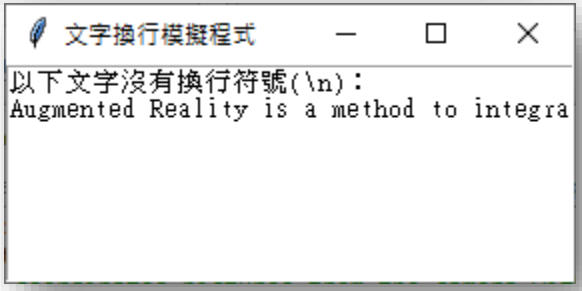
- 基礎元件 - 文字區域(Text)：
  - 文字換行的模式：
    - wrap='char'：
    - 依照字元，到底就換行。



    - wrap='word'：
    - 依照單字換行，單字不會被切斷。



    - wrap='none'：
    - 無換行，會一直延續下去到畫面外，
    - 須配合卷軸才行。



圖形化使用者介面(GUI)

- 基礎元件 - 捲軸(Scrollbar)：
  - 須配合文字區域(Text)或其他元件來使用。
  - 語法：`tkinter.Scrollbar(容器物件, 參數1=值1, 參數2=值2, ...)`
  - 參數：

參數	說明
width	捲軸寬度
background	捲軸的背景顏色，簡稱 bg
borderwidth	捲軸的框線寬度，簡稱 bd
orient	垂直捲軸或水平捲軸，預設值為垂直捲軸「vertical」，水平捲軸為「horizontal」
command	當使用者移動捲軸時，呼叫 command 所指定的函式

圖形化使用者介面(GUI)

- 基礎元件 - 捲軸(Scrollbar)：
  - 替剛才的文字區域(Text)加上垂直捲軸。

```
import tkinter as tk
txt='Augmented Reality is a method to integrate the virtual with the real. \
It coordinates pictures from the camera with virtual data or illustrations, \
seeking to combine them into a new single entity and then interact with it.'
win = tk.Tk()
win.title('垂直捲軸應用程式')
#建立Text元件
text = tk.Text(win, width=40, height=5, wrap='word')
text.insert('end', 'Augmented Reality (AR)\n')  #插入文字
text.insert('end', txt)
text.pack(side='left', fill='y')  #靠左，垂直方向填滿
#建立Scrollbar元件
sbar = tk.Scrollbar(win)
sbar.pack(side='left', fill='y')   #靠左，垂直方向填滿
sbar["command"] = text.yview   #捲軸的動作設為Text的垂直軸(yview)動作
text["yscrollcommand"] = sbar.set     #將sbar連結到Text的yscrollcommand動作
win.mainloop()
```

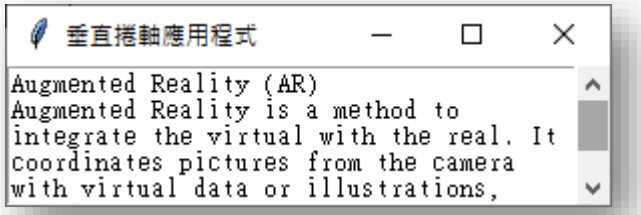
圖形化使用者介面(GUI)

- 基礎元件 - 捲軸(Scrollbar)：
  - 再加上水平捲軸。注意，各元件放置(pack)的順序很重要：

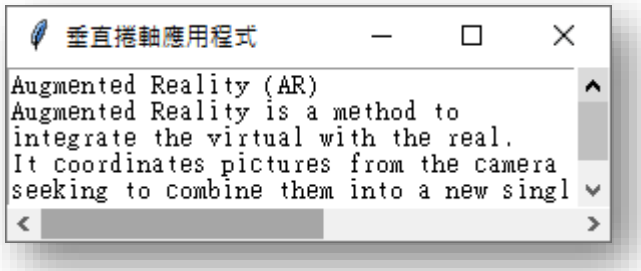
```
import tkinter as tk
txt='Augmented Reality is a method to integrate the virtual with the real. \
It coordinates pictures from the camera with virtual data or illustrations, \
seeking to combine them into a new single entity and then interact with it.'
win = tk.Tk()
win.title('垂直捲軸應用程式')
text = tk.Text(win, width=40, height=5, wrap='none')  #建立Text元件，不換行
text.insert('end', 'Augmented Reality (AR)\n')  #插入文字
text.insert('end', txt)
sbarX = tk.Scrollbar(win, orient='horizontal') #建立水平Scrollbar元件
sbarX.pack(side='bottom', fill='x')   #置放水平捲軸，位置是bottom
sbarX["command"] = text.xview
text["xscrollcommand"] = sbarX.set
text.pack(side='left', fill='y')      #置放Text元件，靠左
sbarY = tk.Scrollbar(win)                      #建立垂直Scrollbar元件
sbarY.pack(side='left', fill='y')     #置放垂直捲軸，位置是left
sbarY["command"] = text.yview
text["yscrollcommand"] = sbarY.set
win.mainloop()
```

圖形化使用者介面(GUI)

- 基礎元件 - 捲軸(Scrollbar)：
  - 執行結果：
  - 垂直捲軸：



  - 垂直+水平捲軸：
  - 注意水平捲軸要加上 `orient = 'horizontal'`。



  - 注意和被控制的元件要相互指定動作的連結。
  - 注意各元件放置(pack)的順序很重要。

圖形化使用者介面(GUI)

- 基礎元件 - 對話方塊(messagebox)：
  - 用來顯示對話方塊，並取得使用者回應。
  - 語法：

```
tkinter.messagebox.方法(標題, 對話文字, 參數1=值1, 參數2=值2, ...)
```

  - 方法：

方法	說明	圖例
showerror	出現錯誤圖示視窗，使用者需按下「確定」。	
showinfo	出現資訊圖示視窗，使用者需按下「確定」。	
showwarning	出現警告圖示視窗，使用者需按下「確定」。	

圖形化使用者介面(GUI)

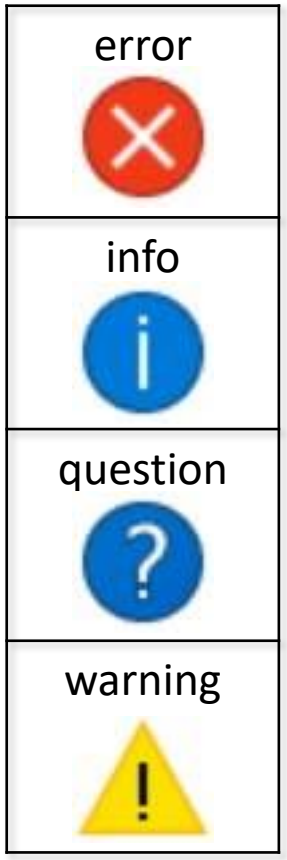
- 基礎元件 - 對話方塊(messagebox)：
  - 方法：

方法	說明	圖例
askokcancel	詢問「確定」或「取消」，使用者選擇「確定」會回傳「True」，選擇「取消」會回傳「False」。	
askquestion	詢問「是」或「否」，使用者選擇「是」會回傳「yes」，選擇「否」會回傳「no」。	
askretrycancel	詢問「重試」或「取消」，使用者選擇「重試」會回傳「True」，選擇「取消」會回傳「False」。	
askyesno	詢問「是」或「否」，使用者選擇「是」會回傳「True」，選擇「否」會回傳「False」。	

圖形化使用者介面(GUI)

- 基礎元件 - 對話方塊(messagebox)：
  - 參數：

參數	說明
default	預設的按鈕項目，如：askokcancel 方法的預設按鈕是「確定」，可以透過 default 參數的設定改為「取消」按鈕，其語法參考如下： `messagebox.askokcancel('參數','Hi!',default='cancel')` 其執行結果如圖所示，預設按鈕改為「取消」。 
icon	設定對話方塊內的圖示，其參數值有：「error」、「info」、「question」、「warning」等。如修改 icon 參數的圖示為「info」，其語法參考如下： `messagebox.askokcancel('參數','Hi!',icon='info')` 其執行結果如圖所示，圖示改為「資訊」圖示。 



圖形化使用者介面(GUI)

- 基礎元件 - 對話方塊(messagebox)：
  - 寫一程式詢問使用者是某已成年。

圖形化使用者介面(GUI)

- 基礎元件 - 對話方塊(messagebox)：
  - 呼叫對話方塊之年齡判斷程式：

```
import tkinter
from tkinter import messagebox
def showMsg():
    Ans=messagebox.askquestion('年齡問題', '你已滿18歲了嗎？')
    if(Ans=='yes'):  #依呼叫方法，會傳回yes或no，True或False
        messagebox.showinfo('恭喜', '您已成年！')
    else:
        messagebox.showinfo('很抱歉', '您尚未成年喔！')
win = tkinter.Tk()
win.title('年齡判斷程式')
win.geometry('250x80')
btn = tkinter.Button(win, text='跳出對話方塊', command=showMsg)
btn.pack()
win.mainloop()
```

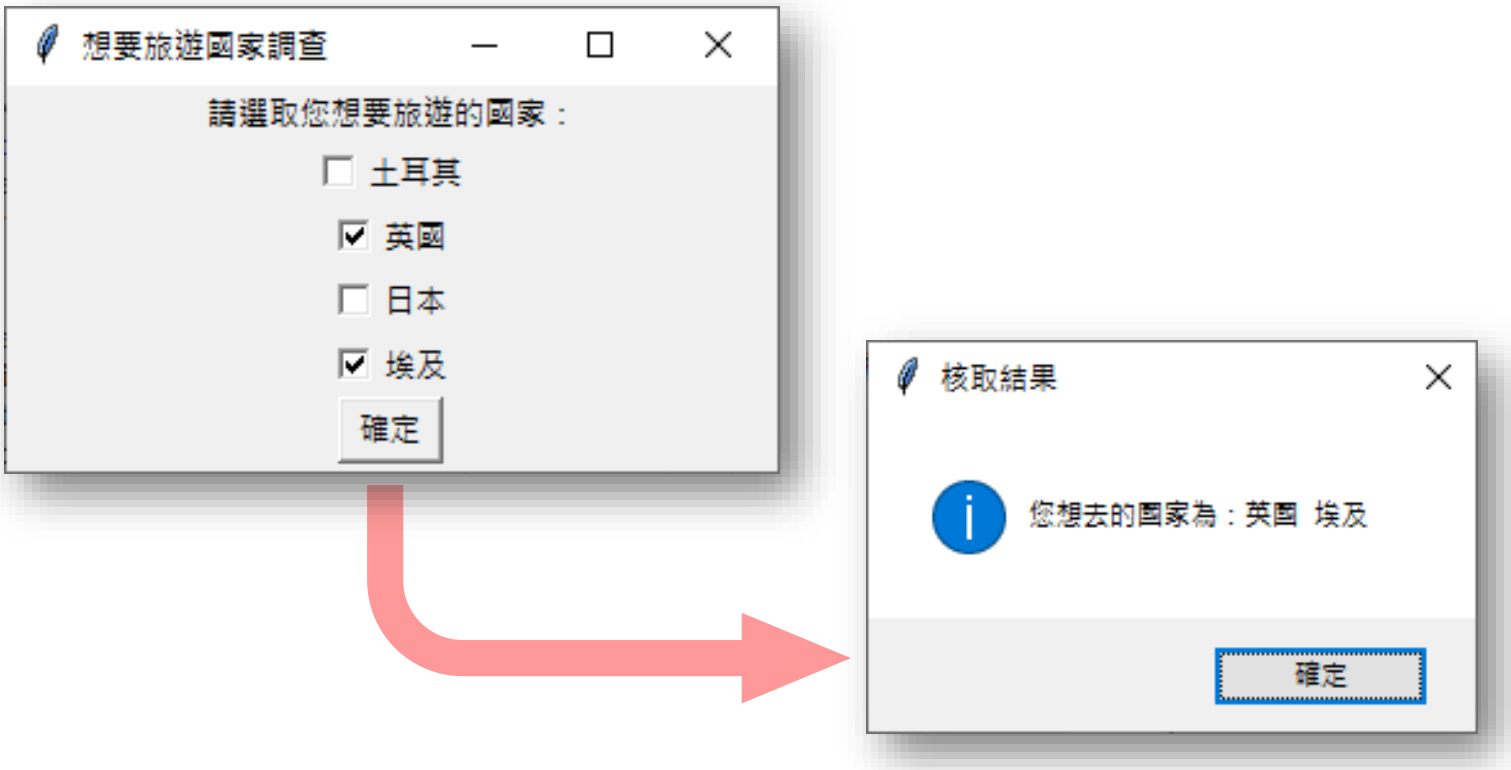
圖形化使用者介面(GUI)

- 基礎元件 -核取按鈕(Checkbutton)：
  - 提供複選的選項。
  - 語法：`tkinter.Checkbutton(容器物件, 參數1=值1, 參數2=值2, ...)`
  - 參數：

參數	說明
text	核取按鈕文字
width	核取按鈕寬度
height	核取按鈕高度
background	核取按鈕的背景顏色，簡稱 bg
foreground	核取按鈕的文字顏色，簡稱 fg
command	當核取按鈕的狀態改變時，呼叫 command 所指定的函式
textvariable	核取按鈕文字之變數，可用作設定或取得核取按鈕的文字內容
variable	可以取得或設定核取按鈕的狀態

圖形化使用者介面(GUI)

- 基礎元件 -核取按鈕(Checkbutton)：
  - 寫一程式讓使用者核取想要去旅遊的國家。
  - 執行畫面：



圖形化使用者介面(GUI)

- 基礎元件 -核取按鈕(Checkbutton)：

```
import tkinter
from tkinter import messagebox
def showMsg():
    result = ''
    for i in check_v:  #依check_v清單內容，若該元素為True，表示有勾選
        if check_v[i].get() == True:
            result = result + country[i] + '  '  #串接相對的country[i]元素
    messagebox.showinfo('核取結果', '您想去的國家為：' + result)
win=tkinter.Tk()
win.title('想要旅遊國家調查')
win.geometry('300x150')
label=tkinter.Label(win, text='請選取您想要旅遊的國家：').pack()
country = {0:'土耳其', 1:'英國', 2:'日本', 3:'埃及'}
check_v = {}
for i in range(len(country)):
    check_v[i] = tkinter.BooleanVar()
    tkinter.Checkbutton(win, text=country[i], variable=check_v[i]).pack()
tkinter.Button(win, text=' 確定 ', command=showMsg).pack()
win.mainloop()
```

圖形化使用者介面(GUI)

- 基礎元件 -核取按鈕(Checkbutton)：
  - 使用dict型態來儲存鍵值與地名的對應。

```
country = {0:'土耳其', 1:'英國', 2:'日本', 3:'埃及'}
```

  - 使用dict來存放是否勾選。用迴圈先置入BooleanVar型態，預設值是False。

```
check_v = {}
for i in range(len(country)):
    check_v[i] = tkinter.BooleanVar()
```

  - 執行後check_v為：

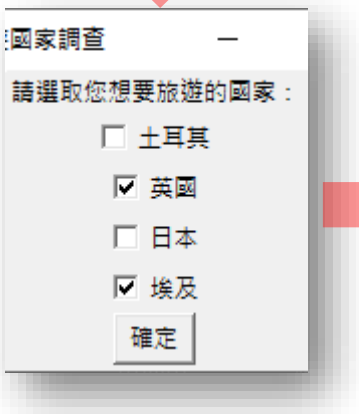
```
check_v = {0:False, 1:False, 2:False, 3:False}
```

  - 使用 `if check_v[i].get() == True:` 取出該核取按鈕的值來判斷是否勾選。

圖形化使用者介面(GUI)

- 基礎元件 -核取按鈕(Checkbutton)：
  - 建立Checkbutton元件：

```
tkinter.Checkbutton(win, text=country[i], variable=check_v[i])
```



[i]	country[i]	check_v[i]
0	土耳其	False
1	英國	True
2	日本	False
3	埃及	True

- 核取按鈕對應到如表格，勾選的選項其內容會變成True，檢查check_v[]就知道哪些選項以勾選。
- 若預設要勾選可用：`check_v[i].set(True)`，讓該選項預設值為True，該選項就會先打勾。

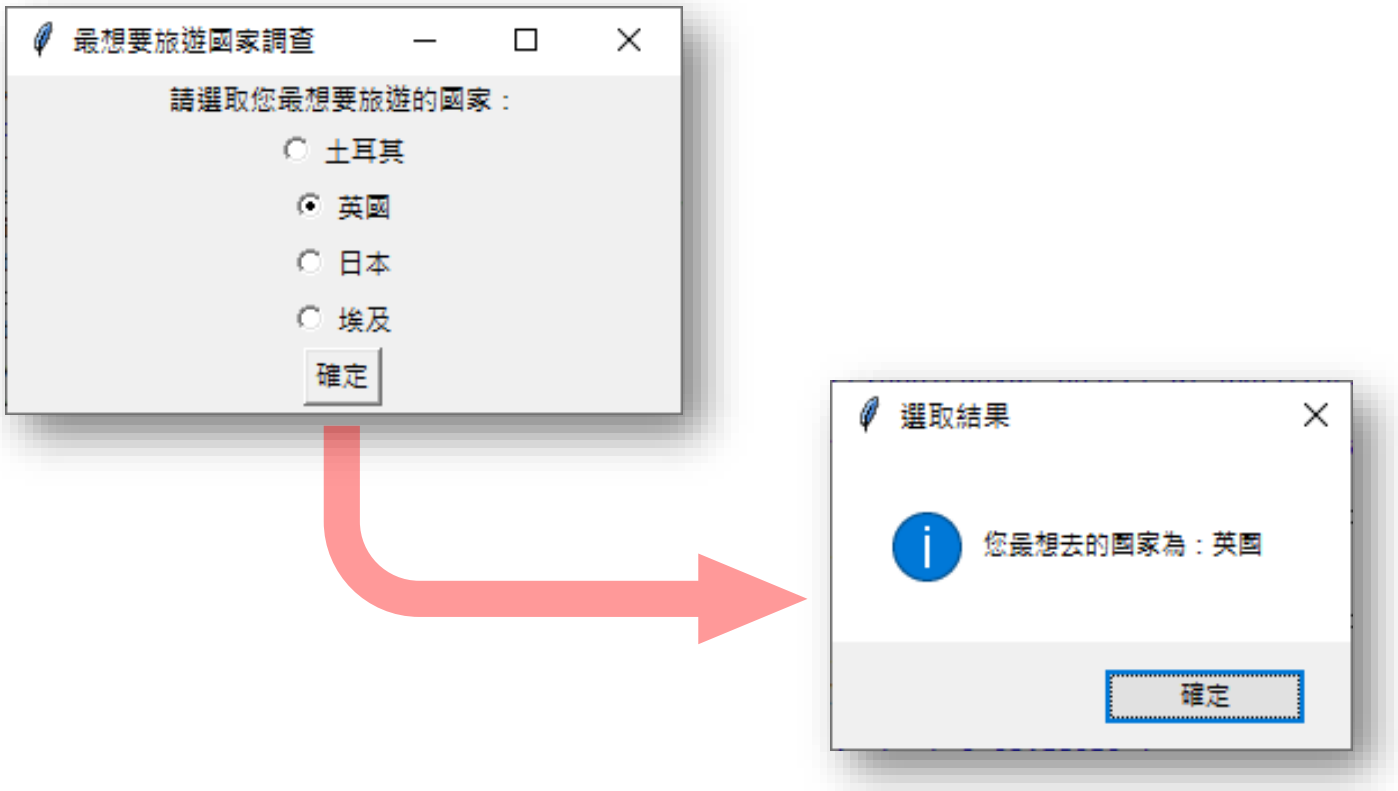
圖形化使用者介面(GUI)

- 基礎元件 -選項按鈕(Radiobutton)：
  - 提供單選的選項。
  - 語法：`tkinter.Radiobutton(容器物件, 參數1=值1, 參數2=值2, ...)`
  - 參數：

參數	說明
text	選項按鈕文字
width	選項按鈕寬度
height	選項按鈕高度
background	選項按鈕的背景顏色，簡稱 bg
foreground	選項按鈕的文字顏色，簡稱 fg
value	選項按鈕的值
command	當選項按鈕的狀態改變時，呼叫 command 所指定的函式
textvariable	選項按鈕文字之變數，可用作設定或取得選項按鈕的文字內容
variable	可以取得或設定選項按鈕的狀態
image	用圖片來當作選項按鈕的內容

圖形化使用者介面(GUI)

- 基礎元件 -選項按鈕(Radiobutton)：
  - 寫一程式讓使用者核取最想要去旅遊的國家。
  - 執行畫面：



圖形化使用者介面(GUI)

- 基礎元件 -選項按鈕(Radiobutton)：

```
#最想要旅遊的國家調查程式
import tkinter as tk
from tkinter import messagebox
def showMsg():
    i=radio_v.get()
    messagebox.showinfo('選取結果', '您最想去的國家為：'+country[i])

win=tk.Tk()
win.title('最想要旅遊國家調查')
win.geometry('300x150')
label=tk.Label(win, text='請選取您最想要旅遊的國家：').pack()
country={0:'土耳其', 1:'英國', 2:'日本', 3:'埃及'}
radio_v = tk.IntVar()   #用來儲存點選到的選項值
radio_v.set(0)   #將預設設為0，是第一個選項，改變此值可改變預設選項
for i in range(len(country)):
    tk.Radiobutton(win,text=country[i],value=i,variable=radio_v).pack()
tk.Button(win, text = "確定", command=showMsg).pack()
win.mainloop()
```

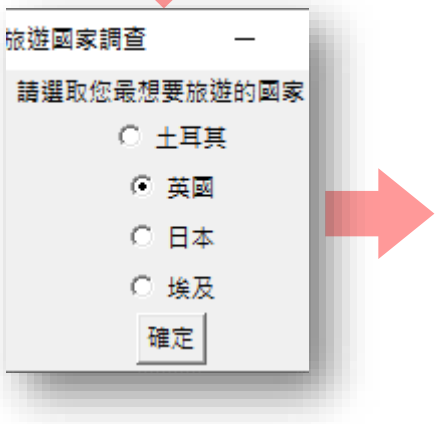
圖形化使用者介面(GUI)

- 基礎元件 -選項按鈕(Radiobutton)：
  - 使用dict型態來儲存鍵值與地名的對應。

```
country = {0:'土耳其', 1:'英國', 2:'日本', 3:'埃及'}
```

  - 使用radio_v以IntVar()型態來存放點選到的值，並先設為0。點選到的選項其value會存入此變數。
  - 以radio_v.get()取出點選到的值，再對應country{}取得國家名稱。
  - 建立按鈕選項：

```
tk.Radiobutton(win,text=country[i],value=i,variable=radio_v)
```

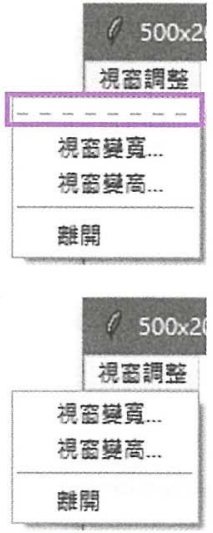


[i]	country[i]	value	radio_v
0	土耳其	0	1 (點選到的選項其value會存入此變數)
1	英國	1	
2	日本	2	
3	埃及	3	

圖形化使用者介面(GUI)

- 基礎元件 -功能表(Menu)：
  - 語法：`tkinter.Menu(容器物件, 參數1=值1, 參數2=值2, ...)`
  - 參數：

參數	說明
background	功能表的背景顏色，簡稱 bg。
foreground	功能表的文字顏色，簡稱 fg。
	第一個選項上方的分隔線是否顯示，預設值為顯示，其執行畫面如右圖所示。
tearoff	如想要不顯示該分隔線，設定「tearoff=0」，其執行畫面如右圖所示。

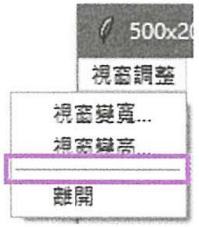


圖形化使用者介面(GUI)

- 基礎元件 -功能表(Menu)：
  - 參數：

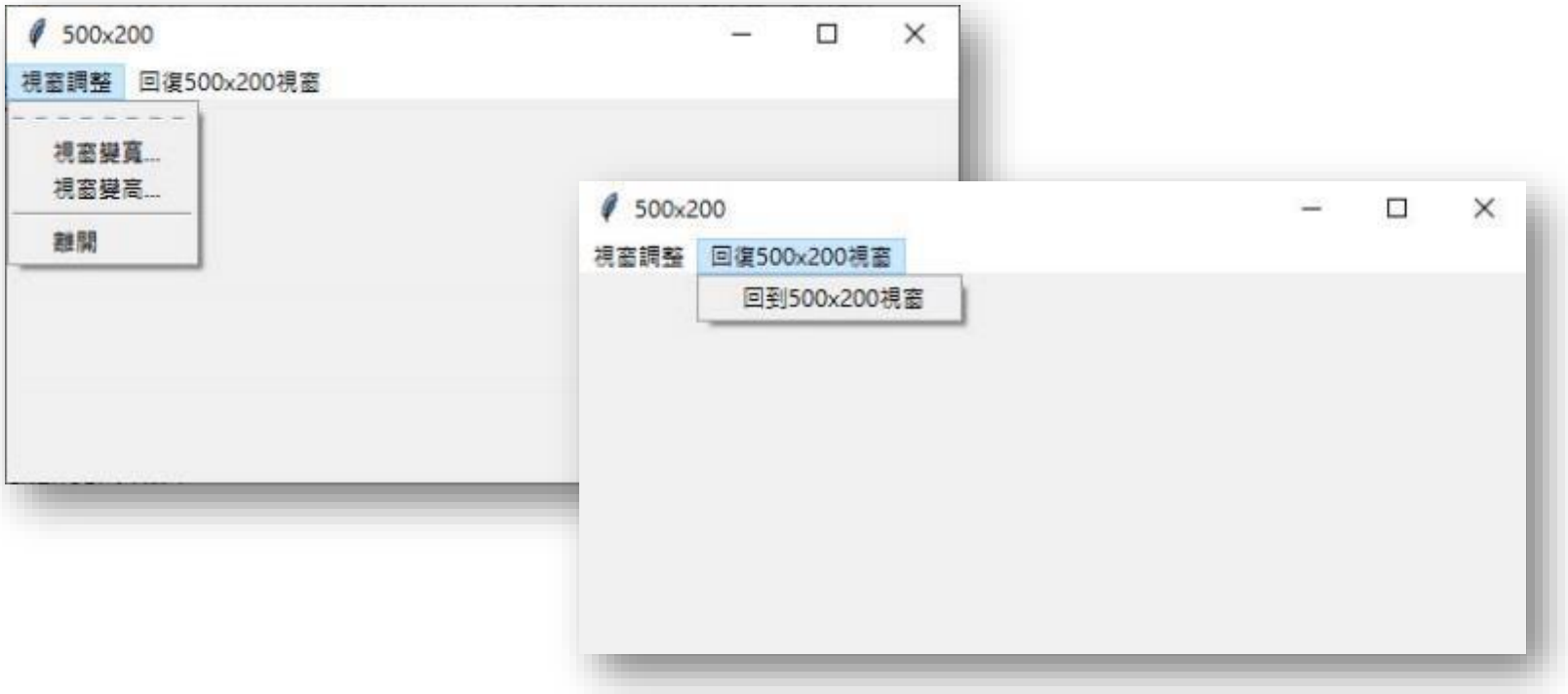
參數	說明

add_cascade(參數)	在功能表內加入子功能表，常用參數為 label 與 menu，label 是用來顯示在子功能表內的文字，menu 是用來指定產生關聯的功能表元件名稱，使用範例如下：
	`filemenu = tkinter.Menu(menu) #功能表元件`
	`menu.add_cascade(label="視窗調整", menu=filemenu)`

add_command(參數)	在子功能表內加入項目，常用參數為 label 與 command，label 是用來顯示在子功能表內項目的文字，command 是用來指定要呼叫的函式，使用範例如下： `filemenu.add_command(label="變寬", command=width)`
add_separator(參數)	在子功能表內加入分隔線，使用範例如下： `filemenu.add_separator() #子功能表內分隔線` 

圖形化使用者介面(GUI)

- 基礎元件 -功能表(Menu)：
  - 設計一程式，功能表為「回復500x200視窗」與「視窗調整」。
  - 「視窗調整」含下拉選單「視窗變寬」、 「視窗變高」與「離開」。



圖形化使用者介面(GUI)

- 基礎元件 -功能表(Menu)：
  - 程式(1/2)：

```
#使用功能表修改視窗的長寬像素
import tkinter as tk

def width():
    win.title('800x200')
    win.geometry('800x200')
def height():
    win.title('500x500')
    win.geometry('500x500')
def back():
    win.title('500x200')
    win.geometry('500x200')

win = tk.Tk()
win.geometry('500x200')
win.title('500x200')
```

圖形化使用者介面(GUI)

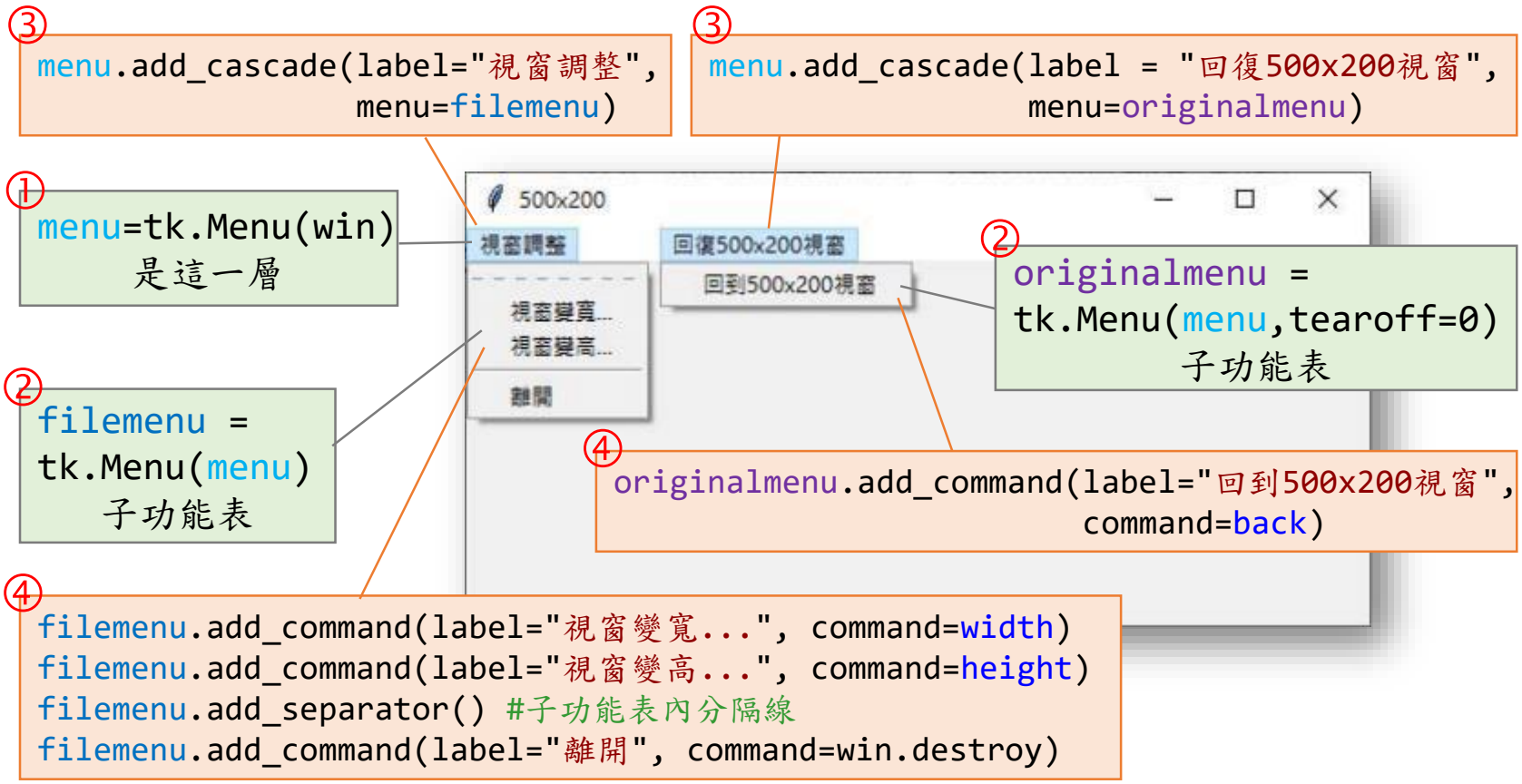
- 基礎元件 -功能表(Menu)：
  - 程式(2/2)：

```
menu = tk.Menu(win)
win["menu"] = menu

filemenu = tk.Menu(menu) #建立子功能表
menu.add_cascade(label="視窗調整", menu=filemenu)
filemenu.add_command(label="視窗變寬...", command=width)
filemenu.add_command(label="視窗變高...", command=height)
filemenu.add_separator() #子功能表內分隔線
filemenu.add_command(label="離開", command=win.destroy)

originalmenu = tk.Menu(menu, tearoff=0) #建立取消預設線的子功能表
menu.add_cascade(label = "回復500x200視窗", menu=originalmenu)
originalmenu.add_command(label="回到500x200視窗", command=back)

win.mainloop()
```

點選該功能後
要執行的函式

圖形化使用者介面(GUI)

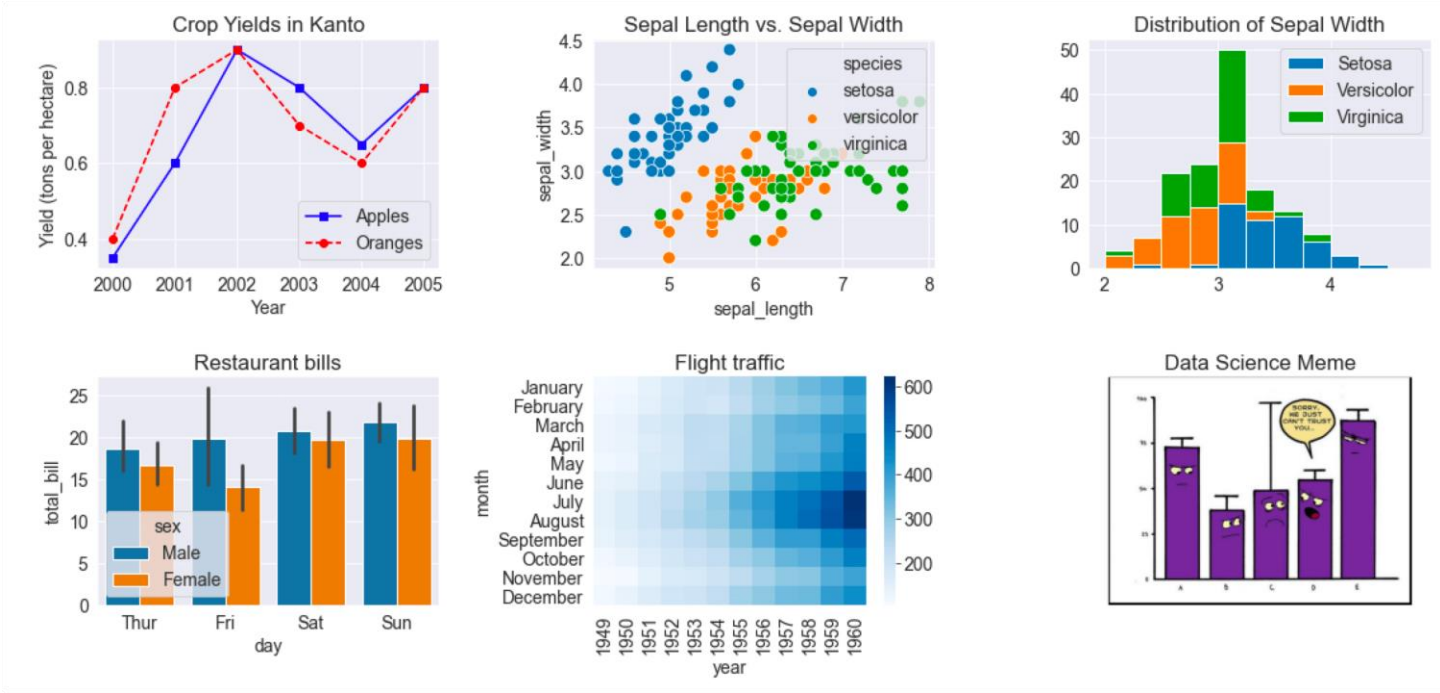
- 基礎元件 -功能表(Menu)：
  - 說明：



休息一下~

圖表繪製

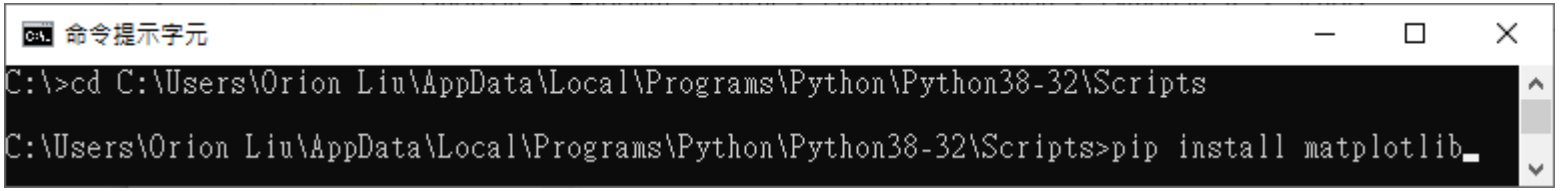
- 以圖形來顯示資訊總是比文字敘述更快讓人接收。
- matplotlib套件提供Python進行繪圖的功能，提供使用者將數據圖形化的方法。



圖表繪製

- 由於matplotlib套件不一定附在Python裡，那就需要先安裝它，安裝方式：
  - 找到Python附的pip.exe程式。
  - 執行 `pip install matplotlib` ，會自動完成安裝。
- 如果安裝Python時有將路徑參數加入系統，就直接執行即可，如圖：

- 如果沒有，就得切換到pip所在的目錄下去執行，例如：



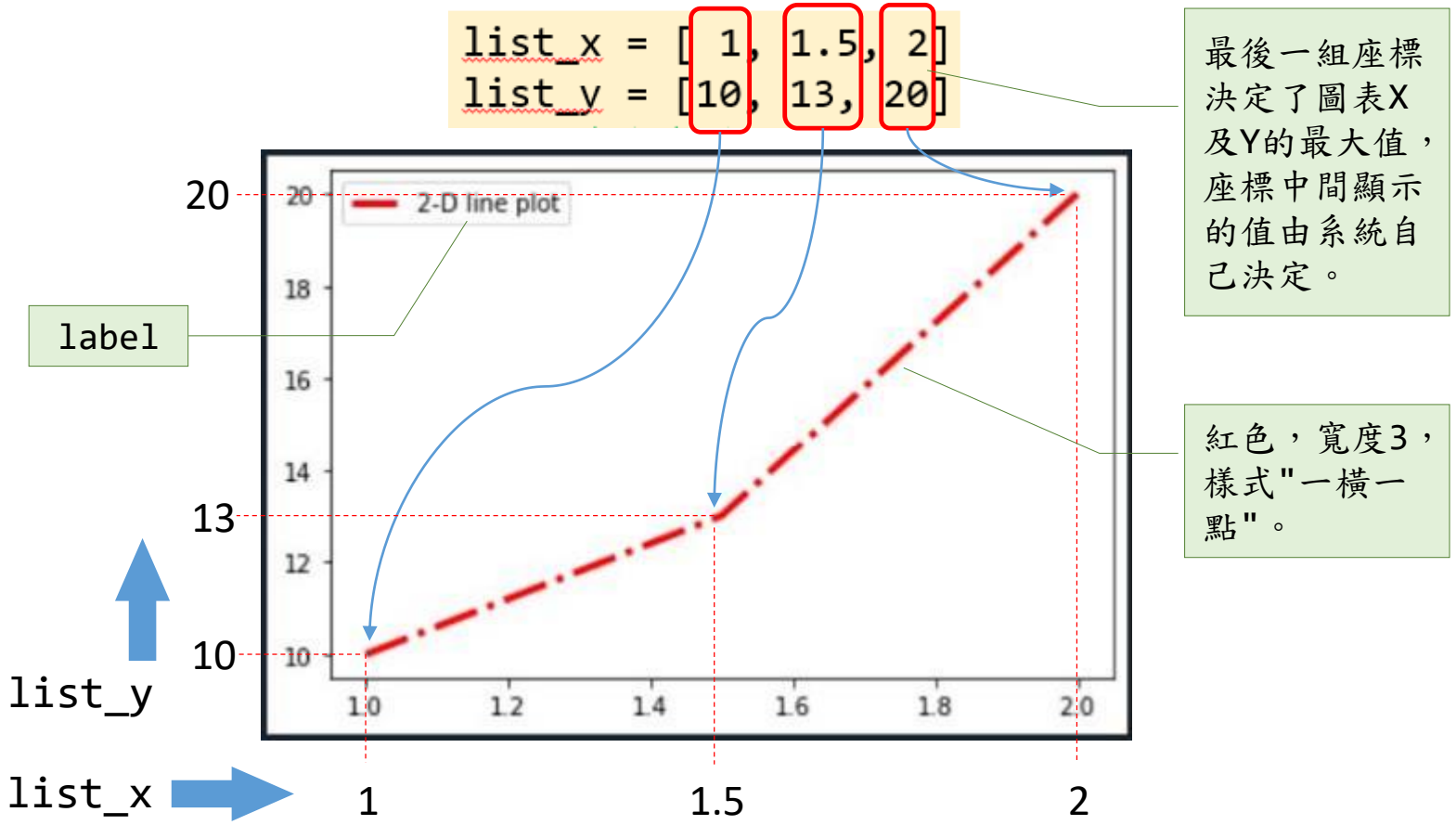
圖表繪製

- 繪製簡單折線圖：
  - 匯入matplotlib套件的pyplot模組即可。

```
import matplotlib.pyplot as plt

list_x = [ 1, 1.5, 2]   #x座標資料清單
list_y = [10, 13, 20]   #y座標資料清單
#設定線條資料
plt.plot(list_x, \   #x座標資料
         list_y, \   #y座標資料
         color = 'red', \   #線條顏色
         ls = '-.', \  #線條樣式
         lw = 3, \    #線條寬度
         label = '2-D line plot')  #線條說明

plt.legend()  #顯示圖例說明(就是上一行的label)
plt.show()  #繪出圖表
```

圖表繪製

- 執行結果：


list_x = [1, 1.5, 2]
list_y = [10, 13, 20]
最後一組座標決定了圖表X及Y的最大值，座標中間顯示的值由系統自己決定。
label
2-D line plot
紅色，寬度3，樣式"一橫一點"。
20
13
10
list_y
list_x
1
1.5
2

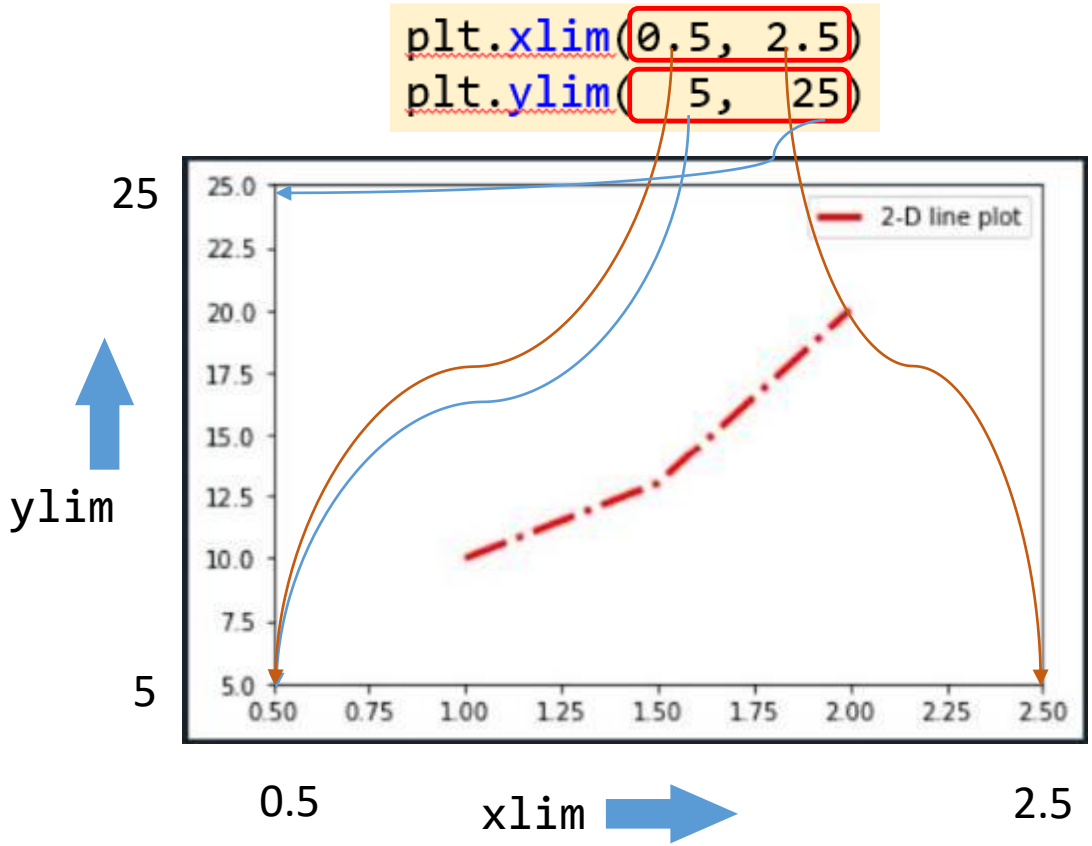
圖表繪製

- 也可以自己指定X軸和Y軸的起始和結束座標。

```
import matplotlib.pyplot as plt
list_x = [ 1, 1.5, 2]   #x座標資料清單
list_y = [10, 13, 20]   #y座標資料清單
#設定線條資料
plt.plot(list_x, \   #x座標資料
         list_y, \   #y座標資料
         color = 'red', \   #線條顏色
         ls = '-.', \  #線條樣式
         lw = 3, \    #線條寬度
         label = '2-D line plot')  #線條說明

plt.xlim(0.5, 2.5)  #指定X軸起始和結束座標值
plt.ylim(  5,  25)     #指定X軸起始和結束座標值

plt.legend()  #顯示圖例說明(就是上一行的label)
plt.show()  #繪出圖表
```

圖表繪製

- 執行結果：


plt.xlim(0.5, 2.5)
plt.ylim(5, 25)
25
ylim
5
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
5.0
0.50
0.75
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2-D line plot
0.5
xlim
2.5

圖表繪製

- 以下資料是近幾年男性與女性初婚的平均年齡統計，請將它用折線圖繪製出來，男性用藍色實線，女性用紅色虛線。

年度	2011	2014	2015	2016	2018	2020
男性	30.7	31.8	32.1	32.2	32.4	33.1
女性	27.8	29.4	29.9	30.0	30.5	31.4

（註：非真實資料）

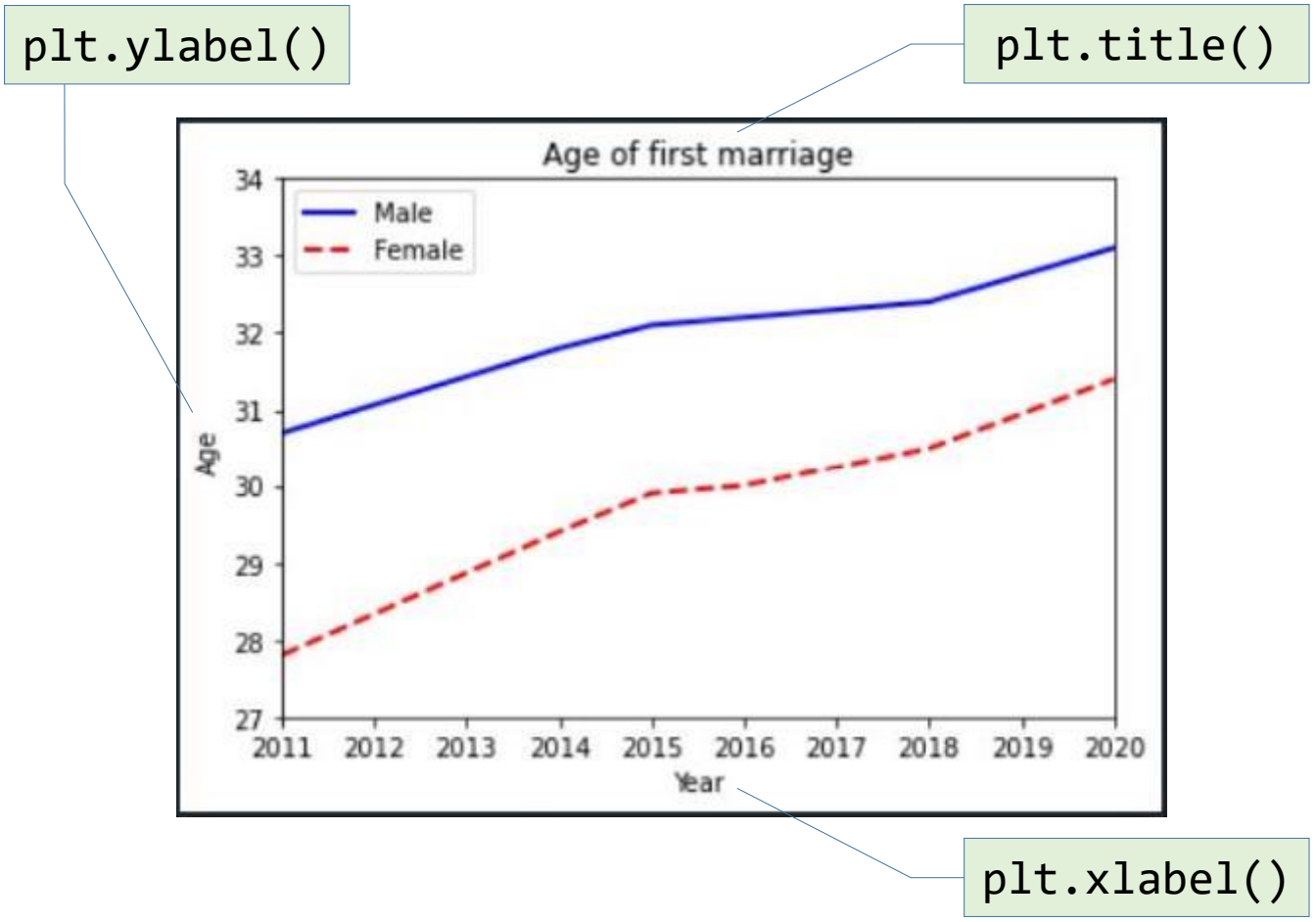
圖表繪製

- 參考程式：

```
import matplotlib.pyplot as plt
#男性資料
list_x1=[2011, 2014, 2015, 2016, 2018, 2020]
list_y1=[30.7, 31.8, 32.1, 32.2, 32.4, 33.1]
plt.plot(list_x1, list_y1, color='blue', lw=2, label='Male')
#女性資料
list_x2=[2011, 2014, 2015, 2016, 2018, 2020]
list_y2=[27.8, 29.4, 29.9, 30.0, 30.5, 31.4]
plt.plot(list_x2, list_y2, color='red', ls='--', lw=2, label='Female')

plt.xlim(2011, 2020)   #自訂X軸座標
plt.ylim(27, 34)       #自訂Y軸座標
plt.legend()
plt.title("Age of first marriage")   #顯示圖表標題
plt.xlabel("Year")    #顯示X軸標題
plt.ylabel("Age")     #顯示Y軸標題
plt.show()
```

圖表繪製

- 執行結果：


plt.ylabel()
plt.title()
Age of first marriage
Male
Female
Age
34
33
32
31
30
29
28
27
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
Year
plt.xlabel()

圖表繪製

- 顯示中文：
  - 當我們將圖表訊息換成中文時卻發現會變成亂碼(沒辦法，外國人寫的嘛)，但開源的好處就是容許度很大，只要再加上一行指定字型的函式就可以了。

```
import matplotlib.pyplot as plt

#用來正常顯示中文標籤
plt.rcParams['font.sans-serif'] = ['DFKai-SB']
#用來正常顯示負號
plt.rcParams['axes.unicode_minus'] = False
```

標楷體

  - 不過連英文字型也會改變喔。

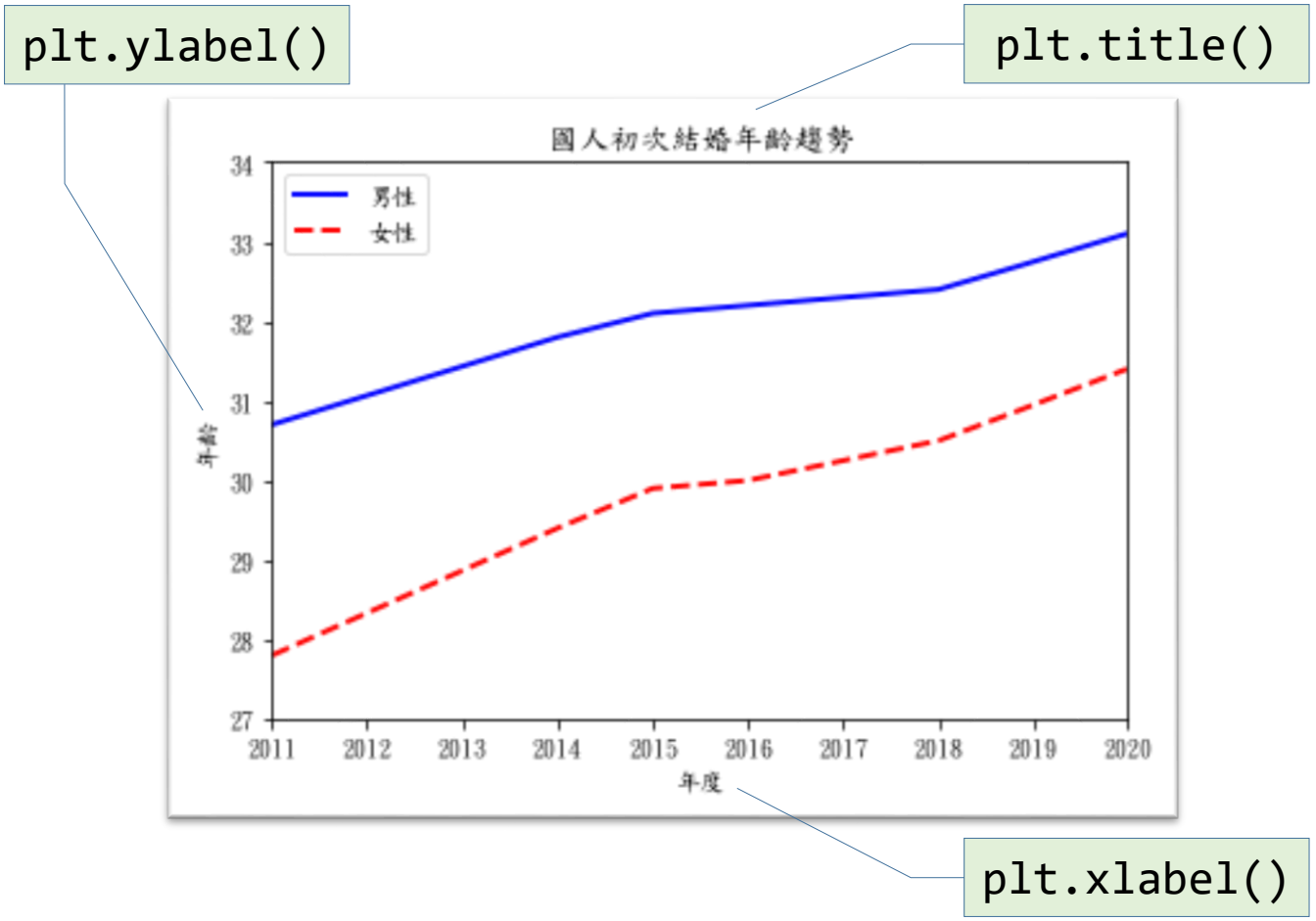
圖表繪製

- 將訊息換成中文顯示：

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['DFKai-SB'] #用來正常顯示中文標籤
plt.rcParams['axes.unicode_minus']=False #用來正常顯示負號
list_x1=[2011, 2014, 2015, 2016, 2018, 2020]
list_y1=[30.7, 31.8, 32.1, 32.2, 32.4, 33.1]
plt.plot(list_x1, list_y1, color='blue', lw=2, label='男性')
list_x2=[2011, 2014, 2015, 2016, 2018, 2020]
list_y2=[27.8, 29.4, 29.9, 30.0, 30.5, 31.4]
plt.plot(list_x2, list_y2, color='red', ls='--', lw=2, label='女性')
plt.xlim(2011,2020)
plt.ylim(27,34)
plt.legend()
plt.title("國人初次結婚年齡趨勢")
plt.xlabel("年度")
plt.ylabel("年齡")
plt.show()
```

圖表繪製

- 執行結果：可以顯示中文了。



圖表繪製

- 常用中文字型的名稱：

字　型	名　稱
標楷體	DFKai-SB
宋體	SimSun
微軟雅黑體	Microsoft YaHei
微軟正黑體	Microsoft JhengHei
細明體	MingLiU

- .plot(...,ls='')線條樣式的參數可以是：

-	:	'dashed'
--	' '(空白)	'dashdot'
-.	'solid'	'dotted'

圖表繪製

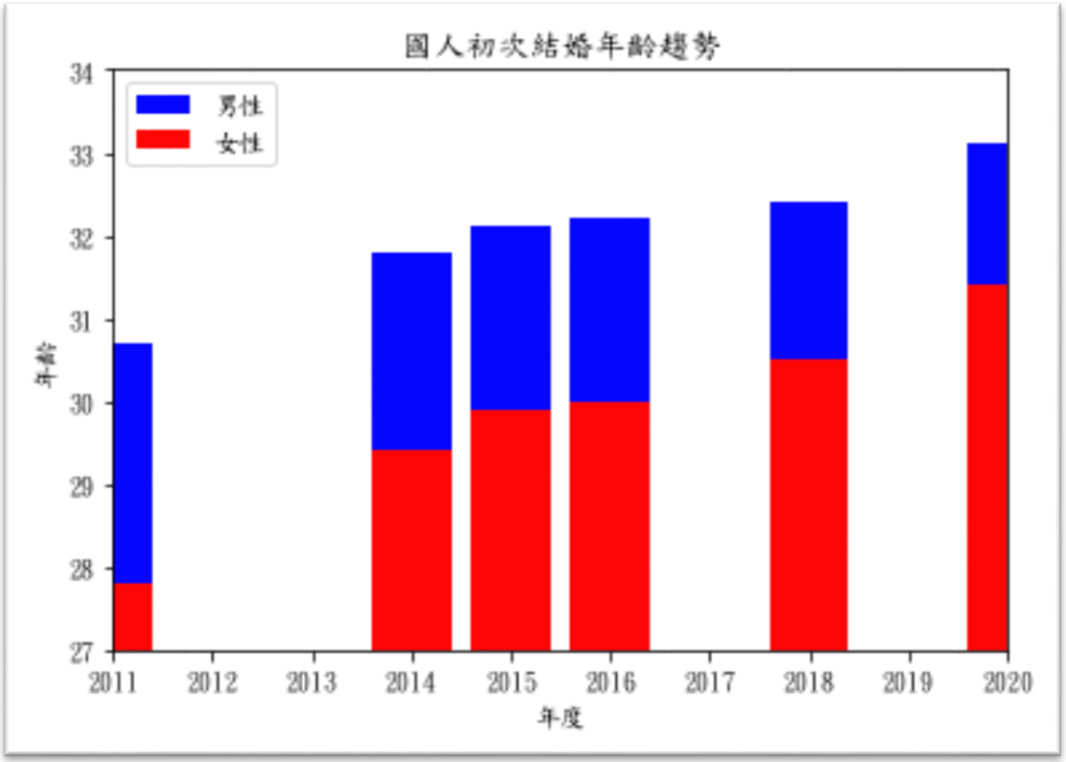
- 繪製柱狀圖：

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['DFKai-SB'] #用來正常顯示中文標籤
plt.rcParams['axes.unicode_minus']=False #用來正常顯示負號
list_x1=[2011, 2014, 2015, 2016, 2018, 2020]
list_y1=[30.7, 31.8, 32.1, 32.2, 32.4, 33.1]
plt.bar(list_x1, list_y1, color='blue', label='男性')
list_x2=[2011, 2014, 2015, 2016, 2018, 2020]
list_y2=[27.8, 29.4, 29.9, 30.0, 30.5, 31.4]
plt.bar(list_x2, list_y2, color='red', label='女性')
plt.xlim(2011, 2020)
plt.ylim(27, 34)
plt.legend()
plt.title("國人初次結婚年齡趨勢")
plt.xlabel("年度")
plt.ylabel("年齡")
plt.show()
```

其實只是把
plot()改成bar()
而已，然後參數
只有顏色和標籤。

圖表繪製

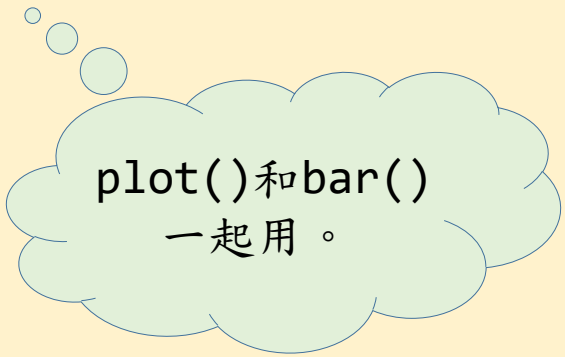
- 執行結果：


國人初次結婚年齡趨勢
男性
女性
年齡
年度
34
33
32
31
30
29
28
27
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020

圖表繪製

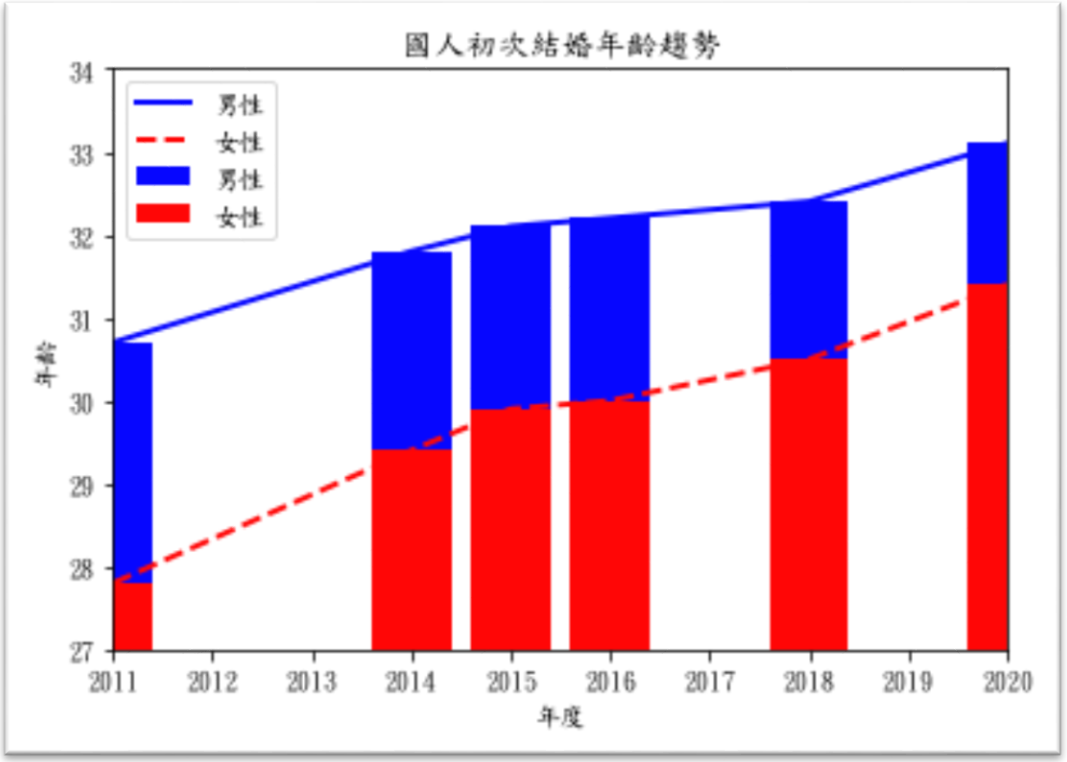
- 也可以同時繪製折現圖及柱狀圖：

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['DFKai-SB'] #用來正常顯示中文標籤
plt.rcParams['axes.unicode_minus']=False #用來正常顯示負號
list_x1=[2011, 2014, 2015, 2016, 2018, 2020]
list_y1=[30.7, 31.8, 32.1, 32.2, 32.4, 33.1]
plt.bar(list_x1, list_y1, color='blue', label='男性')
plt.plot(list_x1, list_y1, color='blue', lw=2, label='男性')
list_x2=[2011, 2014, 2015, 2016, 2018, 2020]
list_y2=[27.8, 29.4, 29.9, 30.0, 30.5, 31.4]
plt.bar(list_x2, list_y2, color='red', label='女性')
plt.plot(list_x2, list_y2, color='red', ls='--', lw=2, label='女性')
plt.xlim(2011, 2020)
plt.ylim(27, 34)
plt.legend()
plt.title("國人初次結婚年齡趨勢")
plt.xlabel("年度")
plt.ylabel("年齡")
plt.show()
```



圖表繪製

- 執行結果：


國人初次結婚年齡趨勢
男性
女性
男性
女性
年齡
年度
34
33
32
31
30
29
28
27
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020

圖表繪製

- 繪製圓餅圖：

```
import matplotlib.pyplot as plt
my_size=[35.35, 23, 26.65, 15]
my_labels=["小明", "小華", "魯夫", "悟空"]
my_colors=["red", "blue", "yellow", "purple"]
my_explode=[0.1, 0, 0, 0]
plt.pie(my_size,\     #圓餅的各部分數字
        labels=my_labels, \    #圓餅各部份的標籤
        colors=my_colors, \    #圓餅各部份的顏色
        explode=my_explode, \ #圓餅各部份的突出的比例
        labeldistance=1.1, \   #圓餅各標籤與餅的距離
        autopct="%3.2f%%", \   #圓餅各部份數字的型態
        pctdistance=0.6, \     #圓餅各部份的數字與圓心的距離
        startangle=0, \        #圓餅起始角度
        shadow=True )          #圓餅是否要有陰影
plt.axis("equal")    #設為正圓形(預設是橢圓形)
plt.legend()
plt.show()
```

圖表繪製

- 數據說明：
  - 要個別指定大小、標籤、顏色、跳出距離。

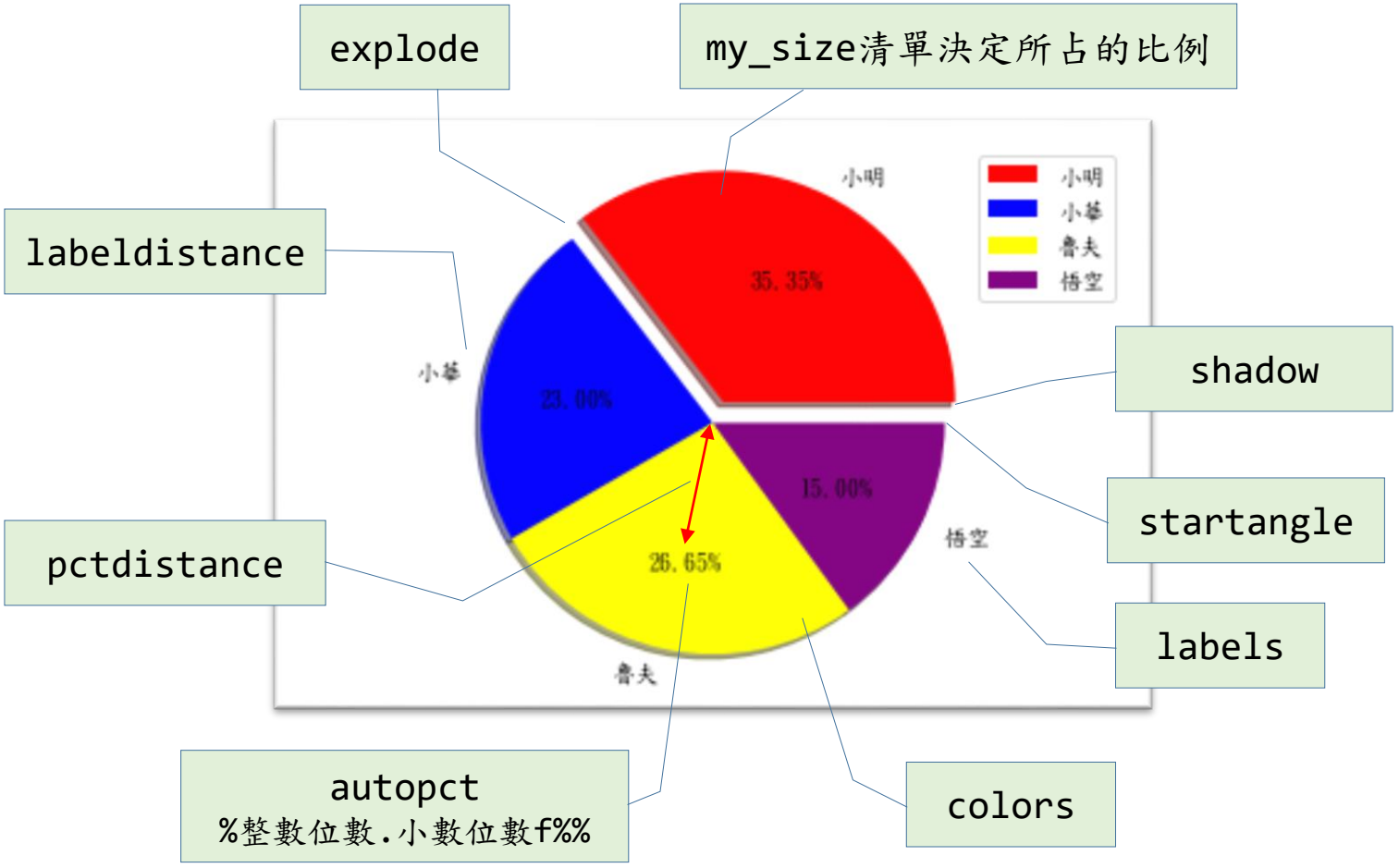
```
my_size=    [ 35.35,    23,     26.65,    15      ]
my_labels=  [ "小明",  "小華",  "魯夫" ,  "悟空"   ]
my_colors=  [ "red",   "blue",  "yellow", "purple" ]
my_explode=[  0.1,     0,       0,        0       ]
```

圓餅圖會有四個部分

  - 其他參數是全體適用，如果不指定也沒關係，都會有各自的預設值。

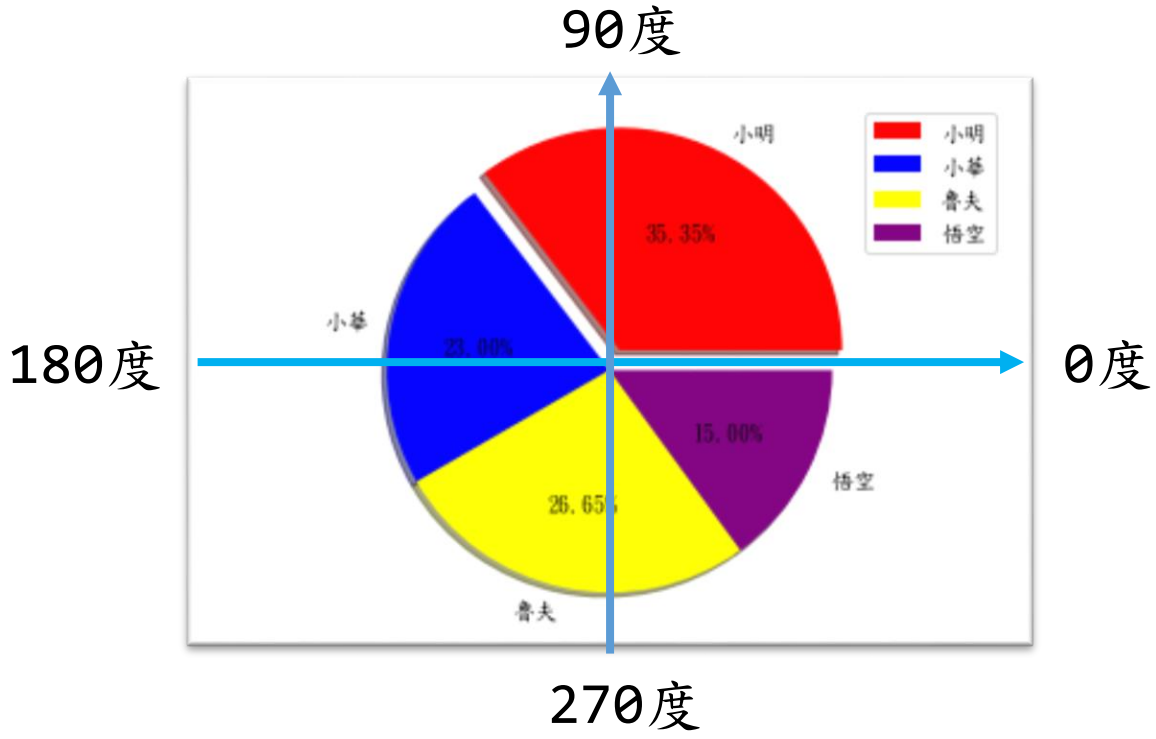
圖表繪製

- 執行結果：各參數說明。



圖表繪製

- startangle(起始角度說明)：
  - 第一個數據從哪一個角度開始繪製。
  - 因為第一個是小明，所以小明從0度開始。



下課~

資料來源：

1. 本講義部分圖片取自網路。
2. 部分取自「程式語言與設計」課本(碁峰、科友)。
3. 專門為中學生寫的程式語言設計(使用Python)，李家同。