

Python研習課程 (II)

程式設計競賽研習

2018/10/20 ~ 21

第五類：函式 (Function)

- ❑ 函式類型
- ❑ 自訂函式
- ❑ 參數與回傳值
- ❑ import 套件

函式 (Function)

- ❑ 有獨特的功能和目的

- ❑ Python 函式:

- ❑ 內建函式 (Built-in Functions)

- ❑ print(), input(), eval(), abs()

- ❑ import 套件中的函式

- ❑ random.randint(a,b)

- ❑ 自訂函式

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)

❑ 為什麼要自訂函式？

- ❑ 沒有符合需要的內建或套件的函式
- ❑ 需要重複使用的程式
- ❑ 有易於維護程式碼的優點

```
# abs() 回傳絕對值
```

```
a=-1.5
```

```
print("a = ",abs(a))
```

```
# 不使用函式
```

```
a=-1.5
```

```
if a < 0:
```

```
    a=-a
```

```
print("a = ",a)
```

函式 (Function)

- ❑ 沒有參數, 也沒有回傳值

- ❑ 例: `print()` 輸出換行

- ❑ 有參數但沒有回傳值

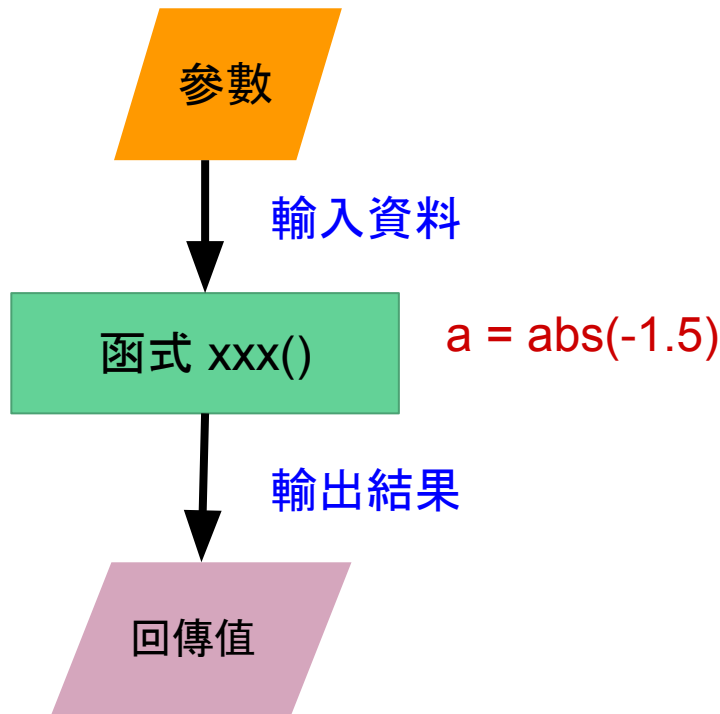
- ❑ 例: `print("Hello World!")`

- ❑ 有回傳值但沒有參數

- ❑ 例: `random.random()`

- ❑ 有參數也有回傳值

- ❑ 例: `random.randint(1,10)`



函式 (Function)

❑ 自訂函式方式

❑ `def` - 必要

❑ `:-` 必要

❑ 函式名() - 必要

❑ 參數 - 非必要

❑ 回傳值 - 非必要, 且可多個回傳值

```
def myFuntion(arg1, arg2,...):  
    execute statement(s)  
    return r1,r2,...
```

函式 (Function)

❑ 自訂函式 - 沒有參數, 也沒有回傳值

輸出5個星號並換行

```
def print5Stars():
```

```
    print("*****")
```

```
print5Stars()
```

練習:輸出加總1~10的總和

```
def add1_10():
```

```
    ....
```

```
add1_10()
```

函式 (Function)

❑ 自訂函式 - 有參數, 但沒有回傳值

輸出n個星號並換行

```
def printNStars(n):
```

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

```
        print("*",end="")
```

```
    print()
```

```
printNStars()
```

練習: 輸出加總1~n的總和

```
def add1_n(n):
```

```
    ....
```

```
add1_n()
```


函式 (Function)

❑ 自訂函式 - 有回傳值

回傳加總1~10的總和

```
def add1_10():  
    sum=0  
    for i in range(1,11):  
        sum += i  
    return sum
```

```
print(add1_10())
```

練習：回傳加總1~n的總和

```
def add1_n(n):  
    ....
```

函式 (Function)

❑ 自訂函式 - 預設參數值

❑ 「預設參數值」- 若不特別指定參數時 ...

❑ 例 : `print(a,b,sep="",end="")`

❑ 定義函數時設定參數值即為「預設參數值」

❑ 有多個參數時, 預設參數值不可以在沒有預設值的參數前面

❑ 例 : `abc(a,b,c=1)` 正確

❑ `abc(a=1,b,c)` 不正確

輸出n個星號並換行 (預設5個星號)

```
def printNStars(n=5):  
    for i in range(n):  
        print("*",end="")  
    print()  
printNStars()
```

函式 (Function)

❑ 自訂函式 - 預設參數值

```
# 練習:第n期未來值計算  
def myFv(p,n,r=0.01):  
    # FV = PV(1+r)**n 第n期未來值計算公式  
    return p*(1+r)**n  
  
print(myFv(1000,1))  
print(myFv(1000,1,0.02))
```

函式 (Function)

❑ 自訂函式 - 多個回傳值

- ❑ Python 函式可回傳多個回傳值
- ❑ `return r1, r2, ...`

計算2數的總和及平均

```
def sum_average(n1,n2):  
    sum=n1+n2  
    average=(n1+n2)/2  
    return sum, average
```

```
s,a = sum_average(5,10)
```

```
print("sum = %d, average = %.2f" % (s,a))
```

變數範圍 Scope of variables

- ❑ 變數有效範圍: 能存取該變數的範圍

- ❑ Local variable (區域變數)

- ❑ 定義在一個函式 中的變數, 有效範圍是在 該函式內

- ❑ Global variable (全域變數)

- ❑ 定義在函式 外的變數, 其有效範圍是 整個 Python 檔案

```
def add_ab(n1,n2):  
    n3=n1+n2  
    return n3
```

```
print(add_ab(1,2))  
print(n3)
```

變數範圍 Scope of variables

❑ 區域變數 & 全域變數 - 5個基本原則

- ❑ A variable must be local or global; it cannot be both.
- ❑ Code in the global scope cannot use any local variables.
- ❑ Global variables can be read from a local scope.
- ❑ Code in a function's local scope cannot use variables in any other local scope.
- ❑ You can use the same name for different variables if they are in different scopes.

變數範圍 Scope of variables

❑ 區域變數 & 全域變數 - 使用建議

- ❑ Using global variables in small programs is fine.
- ❑ It is a bad habit to rely on global variables as your programs get larger and larger.
- ❑ [Avoid using local variables that have the same name](#) as a global variable or other local variables.
- ❑ [Avoid using global variables directly in functions](#) is encouraged.

變數範圍 Scope of variables


Code in the global scope cannot use any local variables.

```
def candyAmount():  
    myCandy=5      # local variable  
  
sharedCandy=10    # global variable  
  
candyAmount()  
  
print('Total candy=', sharedCandy+myCandy)
```


變數範圍 Scope of variables

Code in the global scope cannot use any local variables.

```
def candyAmount():  
    myCandy=5 # local variable  
sharedCandy=10 # global variable  
candyAmount()  
print('Total candy=', sharedCandy+myCandy)
```



```
def candyAmount():  
    myCandy=5  
    return myCandy  
sharedCandy=10  
print('Total candy=', sharedCandy+candyAmount())
```

變數範圍 Scope of variables

Global variables can be read from a local scope.

```
def candyAmount():  
    myCandy=5 # local variable  
    print('Total candy=', sharedCandy+myCandy)  
  
sharedCandy=10# global variable  
  
candyAmount()
```

變數範圍 Scope of variables

Local scopes cannot use variables in other local scopes.

```
def farm():  
    apple=50      # local variable in farm()  
    harvest()  
    print('apple=',apple)  
def harvest():  
    apple=10      # local variable in harvest()  
    orange=20     # local variable in harvest()  
farm()
```

變數範圍 Scope of variables

Local and global variables with same name.

```
def farm1():  
    apple='Apple in farm1'      # local variable in farm1()  
    print(apple)  
def farm2():  
    apple='Apple in farm2'      # local variable in farm2()  
    farm1()  
    print(apple)  
apple='Apple in global'        # global variable  
farm1()  
farm2()  
print(apple)
```

變數範圍 Scope of variables

global Statement - global 敘述

- If you need to modify a global variable from within a function.

```
def farm():  
    global apple                # global statement  
    apple = 'Apple sets in fram()' # modify global variable  
apple='Apple in global'  
farm()  
print(apple)
```

例題: 502 乘積函式

- 請撰寫一程式，將使用者輸入的兩個數字作為參數傳遞給一個名為`compute(x, y)`的函式，此函式將回傳`x`和`y`的乘積。

範例輸入

56

11

範例輸出

616

502 乘積函式

```
def compute(a, b):
```

```
    return a * b
```

```
num1 = eval(input())
```

```
num2 = eval(input())
```

```
print(compute(num1, num2))
```

例題: 504 次方計算函式

- 請撰寫一程式，讓使用者輸入兩個整數，接著呼叫函式 `compute()`，此函式接收兩個參數 `a`、`b`，並回傳 a^b 的值。

範例輸入

14
3

範例輸出

2744

```
# 次方計算函式
def compute(a, b):
    return a**b

a = eval(input())
b = eval(input())
print(compute(a, b))
```

例題: 508 最大公因數函式

- ❑ 請撰寫一程式，讓使用者輸入兩個正整數x,y (半形逗號分隔)，並將x與y傳遞給名為 `compute()` 的函式，此函式回傳x和y的最大公因數 (Greatest Common Divisor)。
 - ❑ 窮舉法: 逐一列出可整除各整數的所有因數，找出最大的公因數。

範例輸入1

12,8

範例輸出1

4

```
# TQC+ 程式語言 508 最大公因數
def compute(a,b):
    gcd=1          # 預設公因數為1
    k=2            # 最小可能因數
    while k<=a and k<=b:
        if a%k==0 and b%k==0:
            gcd=k
        k+=1
    return gcd    # 回傳gcd
x,y=eval(input())
print(compute(x,y))
```


例題：質數函式

- 請撰寫一程式，讓使用者輸入一個整數x，並將x傳遞給名為 `compute()` 的函式，此函式將回傳x是否為質數(Prime number)的布林值，接著再將判斷結果輸出。如輸入值為質數顯示【Prime】，否則顯示【Not Prime】。

- 試除法：若一自然數n，無法被2與 \sqrt{n} 之間的任一整數整除，則n為質數。

範例輸入1

3

範例輸出1

Prime

```
def compute(n):  
    for i in range(2,int(n**0.5)+1):  
        if n % i == 0:    # n被整除  
            return False # n不是質數,回傳False  
    return True          # 無法被整除,n是質數,回傳  
True  
x=eval(input())  
if x>1:  
    if compute(x):  
        print("Prime")  
    else:  
        print("Not Prime")  
else:  
    print("Not Prime")
```

例題：閏年判斷函式

❑ 請撰寫一程式，在 `main()` 函式輸入一年份 `year`，將此整數傳給 `isLeap()` 函式，用以顯示 `year` 是否為閏年。

❑ 提示：閏年判斷規則為，每四年一閏，每百年不閏，但每四百年也一閏。

```
# 閏年判斷函式
def isLeap(y):
    if y%400==0 or (y%4==0 and y%100!=0):
        print("%d is leap year." % (y))
    else:
        print("%d is not leap year." % (y))
def main():
    year=eval (input())
    isLeap(year)
main()
```

例題：費氏數列函式

❑ 請撰寫一程式，計算費氏數列(Fibonacci numbers)，使用者輸入一正整數 num，傳遞給名為 `compute()` 函式，此函式將回傳費氏數列前num個數值。

❑ 提示：費氏數列的某項數字是其前兩項的和，而且第 0項為0，第一項為1，表示方式如下：

❑ $F_0=0$

❑ $F_1=1$

❑ $F_n=F_{n-1}+F_{n-2}$

範例輸入1

10

範例輸出1

0 1 1 2 3 5 8 13 21 34

n1,n2的移動順序可否變動？

```
def compute(n):  
    n1=0    # F0=0  
    n2=1    # F1=1  
    print("%d %d" % (n1,n2),end=" ")  
    for i in range(3,n+1):  
        n3=n1+n2  
        print("%d" % n3,end=" ")  
        n1=n2    # 各向後移動一位  
        n2=n3    # 各向後移動一位  
num=eval(input())  
compute(num)
```

練習：最簡分數

- 請撰寫一程式，讓使用者輸入二個分數，分別是 x/y 和 m/n ，計算這兩個分數的和為 p/q ，再建立 `compute()` 函式回傳 p 和 q 的最大公因數。並將 p 和 q 各除以其最大公因數，最後輸出的結果必須以最簡分數表示。

範例輸入1

```
1,2  
1,6
```

範例輸出1

```
1/2 + 1/6 = 2/3
```

練習：最簡分數

範例輸入1

```
1,2  
1,6
```

範例輸出1

```
1/2 + 1/6 = 2/3
```

範例輸入2

```
12,16  
18,32
```

範例輸出2

```
12/16 + 18/32 = 21/16
```

最簡分數

```
def compute(a,b):
```

```
    gcd=1      # 預設公因數為1
```

```
    k=2        # 最小可能因數
```

```
    while k<=a and k<=b:
```

```
        if a%k==0 and b%k==0:
```

```
            gcd=k
```

```
            k+=1
```

```
    return gcd # 回傳gcd
```

```
x,y=eval(input())
```

```
m,n=eval(input())
```

```
p=x*n + m*y      # 分子*對方分母相加
```

```
q=y*n             # 分母互乘
```

```
gcd=compute(p,q) # 相加後分數最大公因數
```

```
print("%d/%d + %d/%d = %d/%d" % (x,y,m,n,p/gcd,q/gcd))
```

練習：階乘函式

- 請撰寫一程式，在 `main()` 函式輸入整數 `n`，將此整數傳給 `factor()` 函式，用以顯示 $1 \sim n$ 的階乘。

```
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800
```

練習：階乘函式

- ❑ 請撰寫一程式，在 `main()` 函式輸入整數 `n`，將此整數傳給 `factor()` 函式，用以顯示 $1 \sim n$ 的階乘。

```
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800
```

```
# 階乘函式
def factor(k):
    for i in range(1,k+1):
        factor=1
        print("%2d! = " % (i),end="")
        for j in range(2,i+1):
            factor*=j
        print(factor)

def main():
    n=eval(input())
    factor(n)

main()
```

第六類：串列 (List)

- ❑ 一維、二維串列
- ❑ List functions 和應用

Python List

❑ List 串列

- ❑ A list is an ordered **sequence** of **multiple** values.
- ❑ 和 Java, C 陣列 (array)不同, Python list 不用事先宣告長度和維度, 可不斷增加。
- ❑ 相當於 Java ADTs ArrayList, LinkedList。

❑ 建立串列

- ❑ 以 [] 建立
- ❑ list(sequence object), 將 sequence object 轉為串列回傳

```
list_a=[]  
list_b=[1,2,3]  
list_c=['1','2','3']  
list_d=list('Python')
```

Python List

❏ List 串列的操作

+	Concatenation 連接	list_a + list_b 連接兩個串列
*	Repetition 複製	list_a * 2 = list_a + list_a
[], [:]	Slice 索引運算子	list_a[] 以索引操作串列
in, not in	Membership 存在判斷	2 in [1,2,3,4,5], 7 not in [1,2,3,4,5]

Python List

❑ Slice [], [beg:end], 以索引運算子取得串列內容

```
list_a=['apple','banana','orange','broccoli','carrot','cherry']
```

```
list_b=[1,3,5,7,9,11]
```

```
str_a="Python"
```

list_a	apple	banana	orange	broccoli	carrot	cherry
list_b	1	3	5	7	9	11
str_a	P	y	t	h	o	n
index	0 -6	1 -5	2 -4	3 -3	4 -2	5 -1

```
list_a[1]
```

```
list_b[0]
```

```
list_a[-1]
```

```
list_b[-3:]
```

```
list_a[1:3]
```

```
list_b[-3:-1]
```

```
str_a[-3:]
```

```
list_a[:-3]
```

```
list_a[:]
```

```
len(list_a)
```

```
len(list_b[:3])
```

Python List

❏ List 串列的連接 Concatenation

```
list_a=['apple','banana','orange','broccoli','carrot']  
list_b=[1,3,5,7,9,11,13]  
list_c=list_a+list_b  
list_d=list_a+['onion','fig']  
print(list_c)  
print(list_d)
```

Python List

❏ List 串列的複製 Repetition

```
list_a=['apple','banana','orange','broccoli','carrot']  
list_b=[1,3,5,7,9,11,13]  
list_c=list_a*2  
list_d=list_b*3  
print(list_c)  
print(list_d)
```

Python List

❑ 判斷 List 串列中是否存在某個值

Whether a value is or is not in a list.

```
list_a=['apple','banana','orange','broccoli','carrot']  
list_b=[1,3,5,7,9,11,13]  
print('orange' in list_a)           # if 'orange' is in list_a  
print('onion' not in list_a)        # if 'onion' is not in list_a  
print(5 in list_b)                  # if 5 is in list_b  
print(2 not in list_b)              # if 2 is not in list_b
```

Python List

❏ List 串列的迴圈控制

```
list_a=['apple','banana','orange','broccoli','carrot']
list_b=[1,3,5,7,9,11,13]
for i in range(len(list_a)):
    print(list_a[i],end=' ')
print()
for i in list_a:
    print(i,end=' ')
print()
for i in list_b:
    print(i,end=' ')
```

Python List

❑ 常用的 Lists 函式

- ❑ `len(lst)` - 回傳 lst 串列長度
- ❑ `max(lst)` - 回傳 lst 串列中最大項目
- ❑ `min(lst)` - 回傳 lst 串列中最小項目
- ❑ `sum(lst)` - 回傳 lst 串列加總數值
- ❑ `list.index(x[, start[, end]])` - 回傳串列中出現 x 的最小索引值, 若不存在則 `ValueError`
- ❑ `list.count(x)` - 回傳串列中出現 x 的次數

Python List

❑ 常用的 Lists 函式

- ❑ `list.sort(key=None, reverse=False)` - 將串列以 `key` 項目進行排序, `reverse` 控制遞增/遞減
- ❑ `list.reverse()` - 將串列中的項目順序反轉 list.sort() 和 sorted() 的差異？
- ❑ `list.append(x)` - 將 `x` 項目加在串列最尾端
- ❑ `list.extend(iterable)` - 連接 `iterable` 和串列, 和 `(+)` 連接較接近
- ❑ `list.insert(i, x)` - 將 `x` 項目加入串列索引 `i` 處, `insert(len(list),x) = append(x)`
- ❑ `list.remove(x)` - 將串列中第一個出現的 `x` 項目刪除, 若無該項目會產生 `ValueError`
- ❑ `list.pop([i])` - 回傳並刪除串列中索引 `i` 項目, 若無指定 `i` 則為最後一項

Python List

❏ 常用的 Lists 函式

```
list_a=['apple','banana','orange','broccoli','carrot']  
list_b=[1,3,5,7,9]  
list_c=list_a + list_b  
print(list_c)  
list_a.append('cherry')  
list_a.extend(list_b)  
print(list_a)  
list_a.append(list_b)  
print(list_a)
```

Python List

❏ 二維串列的建立

```
# 建立一個 3*5 二維串列
import random
list_a=[]
for i in range(3):      # 可視為有 3 列
    list_a.append([])  # 增加一個維度(行)
    for j in range(5): # 可視為有 5 行
        list_a[i].append(random.randint(1,20))
print(list_a)
```

Python List

❏ 二維串列的建立

```
# 建立一個不定長度二維串列
import random
list_a=[]
for i in range(random.randint(1,5)):      # 隨機列數(1~5)
    list_a.append([])                    # 增加一個維度(行)
    for j in range(random.randint(1,5)):  # 隨機行數(1~5)
        list_a[i].append(random.randint(1,20))
print(list_a)
```

Python List

❑ 二維串列的操作

```
list_a=[[1,2,3],[4,5,6],[7,8,9]]  
print(list_a)  
print(list_a[0])  
print(list_a[0:2])  
print(list_a[0][2])
```

❑ 多維串列

- ❑ 不見得要用維度去思考，串列中的每個元素都可以是串列，或其他物件

例題：撲克牌總和

❑ 請撰寫一程式，讓使用者輸入52張牌中的5張，計算並輸出其總和。

❑ 提示：J、Q、K以及A分別代表11、12、13以及1。

範例輸入

5
10
K
3
A

大小寫怎麼處理？

範例輸出

32

```
cards=[] # 存放牌的串列
result=0
for i in range(5):
    cards.append(input()) # 輸入5張牌放入串列
for i in cards:
    if i == 'A': result += 1
    elif i == 'J': result += 11
    elif i == 'Q': result += 12
    elif i == 'K': result += 13
    else: result += eval(i) # 數字牌
print(result)
```

例題：眾數

- ❑ 請撰寫一程式，讓使用者輸入十個整數作為樣本數，輸出眾數(樣本中出現最多次的數字)及其出現的次數。

- ❑ 提示：假設樣本中只有一個眾數。

範例輸入

```
34
18
22
32
18
29
30
38
42
18
```

範例輸出

```
18
3
```

```
size = 10
sample = []
count = [0]*size
for i in range(size):
    num = int(input())
    sample.append(num)
    count[sample.index(num)] += 1
num_occu = max(count)
print(sample[count.index(num_occu)])
print(num_occu)
```

例題：串列數字排序

- 請撰寫一程式，要求使用者輸入十個數字並存放在串列中。接著由大到小的順序顯示最大的3個數字。

範例輸入

```
40  
32  
12  
29  
20  
19  
38  
48  
57  
44
```

範例輸出

```
57 48 44
```

```
# 串列數字排序
```

```
lst=[]          # 建立 lst串列
```

```
for i in range(10):
```

```
    lst.append(eval(input())) # 輸入10個數字放入串列
```

```
lst.sort()      # 將串列排序
```

```
print(lst[-1],lst[-2],lst[-3])
```

使用append()

使用sort()

例題：二維串列成績計算

- ❑ 請撰寫一程式，讓使用者輸入三位學生各五筆成績，將資料儲存在二維串列 `score_lst`，接著再計算並輸出每位學生的總分及平均分數。

❑ 提示：平均分數輸出到小數點後第二位。

增加維度的方法

```
score_lst=[]                # 學生成績串列
stu_lst=["1st","2nd","3rd"] # 區別學生串列
for i in range(3):          # 分別取得3個學生的成績
    print("The %s student:" % stu_lst[i])
    score_lst.append([])     # 將 score_lst 串列增加一個維度
    for j in range(5):       # 取得5筆成績
        score_lst[i].append(eval(input()))
for i in range(3):
    print("Student %d" % (i+1))
    print("#Sum %d" % (sum(score_lst[i])))          # 計算總分
    print("#Average %.2f" % (sum(score_lst[i])/5))  # 計算平均
```

例題：串列最大值最小值索引

❑ 請撰寫一程式，讓使用者建立一個3*3的矩陣，其內容為從鍵盤輸入的整數 (不重複)，接著輸出矩陣最大值與最小值的索引。

❑ 提示：注意輸出格式。

```
size=3      # 設定陣列大小
matrix=[]   # 宣告 n*n 陣列
for i in range(size):
    matrix.append([])
    for j in range(size):
        matrix[i].append(eval(input()))
max_num=min_num=matrix[0][0] # 預設最大最小值
max_index=min_index=[0,0]   # 預設索引值
for i in range(size):
    for j in range(size):
        if matrix[i][j] > max_num:    # 尋找最大值
            max_num=matrix[i][j]
            max_index=[i,j]
        elif matrix[i][j] < min_num:  # 尋找最小值
            min_num=matrix[i][j]
            min_index=[i,j]
print("Index of the largest number %d is: (%d, %d)" % (max_num,max_index[0],max_index[1]))
print("Index of the smallest number %d is: (%d, %d)" % (min_num,min_index[0],min_index[1]))
```

例題：矩陣相加

❑ 請撰寫一程式，讓使用者建立兩個 2*2 矩陣，其內容為從鍵盤輸入的整數，接著輸出這兩個矩陣的內容以及它們相加的結果。

❑ 提示：數字間以空白分隔。

```
def getMatrix(mat,num,rows,cols):
    print("Enter matrix %d:" % num)
    for i in range(rows):
        mat.append([])
        for j in range(cols):
            print("[%d, %d]: " % (i+1,j+1),end="")
            mat[i].append(eval(input()))

def showMatrix(mat,num,rows,cols):
    print("Matrix %d:" % num)
    for i in range(rows):
        for j in range(cols):
            print("%d" % mat[i][j],end=" ")
        print()

mat_rows=mat_cols=2 # 設定陣列大小
mat1=[]
mat2=[]
getMatrix(mat1,1,mat_rows,mat_cols)
getMatrix(mat2,2,mat_rows,mat_cols)
showMatrix(mat1,1,mat_rows,mat_cols)
showMatrix(mat2,2,mat_rows,mat_cols)
print("Sum of 2 matrices:")
for i in range(mat_rows):
    for j in range(mat_cols):
        print("%d" % (mat1[i][j] + mat2[i][j]),end=" ")
    print()
```

練習：平均溫度

- ❑ 請撰寫一程式，讓使用者輸入四週各三天的溫度，儲存到二維串列，接著計算並輸出這四週的平均溫度及最高、最低溫度。
 - ❑ 提示：平均溫度輸出到小數點後第二位。

練習：平均溫度

```
num_week=4
num_day=3
temp=[]
for i in range(num_week):
    temp.append([])
    print("Week %d:" % (i+1))
    for j in range(num_day):
        temp[i].append(eval(input("Day %d:" % (j+1))))
comb=[]
for i in range(num_week):
    comb.extend(temp[i])           # 將二維合併為一維
avg=sum(comb)/(num_week*num_day) # sum()計算總和
print("Average: %.2f" % avg)      # 計算平均
print("Highest:", max(comb))      # max()找最大值
print("Lowest:", min(comb))       # min()找最小值
```

練習：大樂透號碼

- ❑ 請撰寫一程式，以 `lotto()` 產生大樂透號碼，並以 `main()` 函式呼叫五次 `lotto()` 函式，亦即產生五組大樂透號碼。請將產生的樂透號碼由小至大排序。

練習：大樂透號碼

```
import random
def lotto():
    lotto_lst=[]
    for i in range(7):
        lottoNum=random.randint(1,49)
        if lottoNum not in lotto_lst:    # 樂透號碼不重覆
            lotto_lst.append(lottoNum) # 加入list中
    lotto_lst.sort()
    print(lotto_lst)
def main():
    for i in range(6):
        lotto()
main()
```

第七類：數組(Tuple)、集合(Set)及詞典(Dictionary)

- ❑ 定義、建立
- ❑ 方法和應用

Python Tuple

❑ Tuple 數組

- ❑ tuple 和 list 很相近, 差異在於 list 是可改變的 (mutable), tuple 是不可改變 (immutable)。
- ❑ 可以刪除 (del) 整個 tuple, 但不可刪除個別元素和改變資料 內容。
- ❑ 不能 append() 和 insert(), 但可使用 (+) 連接和 (*) 複製, 其它串列的函式和運算子 tuple 都可以使用。
- ❑ 只連結一個元素時, 要在連結的元素後加上逗號 (,)。

❑ 建立數組

- ❑ 以 () 建立
- ❑ tuple(sequence object), 將 sequence object 轉為數組回傳

Python Tuple

❏ Tuple 數組的操作

```
tuple_a=()
tuple_b=("A","B")
tuple_a+=(1,2,3)
tuple_a+=(4,)          # 只連結一個元素時要加上(,)
str_a="This is Python"
list_a=[1,2,3,4]
tuple_c=tuple(str_a)   # tuple()轉換型態回傳
tuple_d=tuple(list_a)  # tuple()轉換型態回傳
print(tuple_a)
print(tuple_b)
print(tuple_c)
print(tuple_d)
```

Python Set

❑ Set 集合

- ❑ 集合是不可重複的資料。
- ❑ 集合是無序的 (unordered), 元素的順序對 set 沒有意義。
- ❑ 使用 add(x) 將 x 加入集合, remove(x) 將 x 從集合中刪除。
- ❑ 其它串列的函式和運算子 set 也都可以使用。

❑ 建立集合

- ❑ 以 {} 建立集合內容。
- ❑ 以 set(sequence object), 將 sequence object 轉為集合回傳。
- ❑ 以 set() 建立空集合。

Python Set

❑ Set 集合的操作

- ❑ 聯集 (union): $\text{set_a} \mid \text{set_b}$ - 集合的相加
- ❑ 交集 (intersection): $\text{set_a} \& \text{set_b}$ - a,b集合共有的項目
- ❑ 差集 (difference): $\text{set_a} - \text{set_b}$ - a集合有但b集合沒有的項目
- ❑ 對稱差集 (symmetric difference): $\text{set_a} \wedge \text{set_b}$ - a,b集合所有項目去掉共有的項目
- ❑ 子集合 (subset)、超集合 (superset) 判斷: 若a集合的項目b集合裡全都有, b集合中有a集合沒有的項目, 則稱a是b的子集合, b是a的超集合
- ❑ 可以用 $==$ 、 $!=$ 判斷兩集合是否相等或不相等 (無關順序)

Python Set

❏ Set 集合的操作

```
set_1={2,1,7,4}
set_2=set()    # 建立空集合
set_3=set([x for x in range(8)])
set_4=set("Python")
set_5={1,2,4,7}
print(set_1)
print(set_2)
print(set_3)
print(set_4)
print(set_1.issubset(set_3))  # 子集合判斷
print(set_1==set_5)  # 集合相等判斷
```

Python Dictionary

❑ Dictionary 詞典

- ❑ 以「鍵值對(key-value pair)」的結構來定義和存取資料。
- ❑ key(鍵)就是它的索引, value 是 key 對應的值, key 和 value 間用冒號 (':', colon)分隔, 並以大括號 ('{}', brace)包起來。
- ❑ key(鍵)可以是數值或字串, 但 key 是不可改變的 (immutable), value可以改變。

❑ 建立詞典

- ❑ 以 {} 建立詞典, 但讀取和使用時還是用中括號 ([]).
- ❑ 以 dict(sequences of key-value pairs), 將 sequences object 轉為詞典回傳。

Python Dictionary

❑ 常用的 Dictionary 詞典函式

- ❑ `del` - 刪除詞典中的某一 key-value pair
- ❑ `dict.keys()` - 回傳 dict 詞典中的鍵(key)
- ❑ `dict.values()` - 回傳 dict 詞典中的值(value)
- ❑ `dict.items()` - 回傳 dict 詞典中的鍵值對(key-value pair)
- ❑ `dict.copy()` - 複製 dict 詞典
- ❑ `dict.update(dict_1)` - 將 dict_1 合併到 dict 裡

Python Dictionary

❏ Dictionary 詞典的操作

```
dict_1={'name':'Jack','age':20} # 設定一個 dictionary
dict_2={} # 宣告一個空詞典
print(dict_1['name']) # 讀取dict_1中key='name'的值/資料內容(value)
print(dict_1['age']) # 讀取dict_1中key='age'的值/資料內容(value)
print(dict_1[0]) # 錯誤, 不能用位置索引來存取, 會導致 KeyError
dict_1['gender'] # 錯誤, 不存在的key, 會導致 KeyError
dict_1['age']=21 # 改變dict_1中key='age'的值/資料內容(value)
dict_1['gender']='male' # 加入一組新的key-value pair
dict_1['zodiac']='Pisces' # 加入一組新的key-value pair
print(dict_1)
del dict_1['gender'] # 刪除key='gender'元素
print(dict_1)
```


例題：數組合併排序

- ❑ 請撰寫一程式，輸入並建立兩組數組，各以-9999為結束點(數組中不包含-9999)。將此兩數組合併並從小到大排序之，顯示排序前和排序後的數組。

Create tuple1:

9

0

-1

3

8

-9999

Create tuple2:

28

16

39

56

78

88

-9999

Combined tuple before sorting: (9, 0, -1, 3, 8, 28, 16, 39, 56, 78, 88)

Combined list after sorting: [-1, 0, 3, 8, 9, 16, 28, 39, 56, 78, 88]

例題：數組合併排序

- ❑ 請撰寫一程式，輸入並建立兩組數組，各以-9999為結束點(數組中不包含-9999)。將此兩數組合併並從小到大排序之，顯示排序前和排序後的數組。

```
tup1=()
tup2=()
print("Create tuple1:")
num=eval(input())
while num!=-9999:
    tup1+=(num,) # tuple 連接
    num=eval(input())

print("Create tuple2:")
num=eval(input())
while num!=-9999:
    tup2+=(num,) # tuple 連接
    num=eval(input())

tup_comb=tup1+tup2 # tuple 合併
print("Combined tuple before sorting:",tup_comb)
lst_comb=list(tup_comb)
print("Combined list after sorting:",sorted(lst_comb))
```

例題：集合條件判斷

- ❑ 請撰寫一程式，輸入數個整數並儲存至集合，以輸入-9999為結束點(集合中不包含-9999)，最後顯示該集合的長度(Length)、最大值(Max)、最小值(Min)、總和(Sum)。

範例輸入

```
34
-23
29
7
0
-1
-9999
```

範例輸出

```
Length: 6
Max: 34
Min: -23
Sum: 46
```

```
num_set=set() # 指定 set()
n=eval(input())
while n!=-9999:
    num_set.add(n) # 集合以add()連接
    n=eval(input())
print("Length:",len(num_set)) # set len()長度
print("Max:",max(num_set)) # set max()最大值
print("Min:",min(num_set)) # set min()最小值
print("Sum:",sum(num_set)) # set sum()總和
```

例題：串列數組轉換

- ❑ 請撰寫一程式，輸入數個整數並儲存至串列中，以輸入-9999為結束點(串列中不包含-9999)，再將此串列轉換成數組，最後顯示該數組以及其長度(Length)、最大值(Max)、最小值(Min)、總和(Sum)。

```
num_lst=[]
while True:
    n=int(input())
    if n== -9999:
        break
    num_lst.append(n)      # 建立為串列
num_tuple=tuple(num_lst)  # tuple()轉換
print(num_tuple)
print("Length:",len(num_tuple)) # tuple len()數量
print("Max:",max(num_tuple))    # tuple max()最大值
print("Min:",min(num_tuple))    # tuple min()最小值
print("Sum:",sum(num_tuple))    # tuple sum()總和
```

例題：共同科目-集合判斷

- ❑ 請撰寫一程式，輸入X組和Y組各自的科目至集合中，以字串"end"作為結束點(集合中不包含字串"end")。請依序分行顯示(1)X組和Y組的所有科目、(2)X組和Y組的共同科目、(3)Y組有但X組沒有的科目，以及(4)X組和Y組彼此沒有的科目(不包含相同科目)。

- ❑ 提示：科目須參考範例輸出樣本，依字母由小至大進行排序。

```
X=set() # 集合宣告
Y=set()
print("Enter group X's subjects:")
subject=input()
while subject!="end":
    X.add(subject)
    subject=input()
```

```
print("Enter group Y's subjects:")
subject=input()
while subject!="end":
    Y.add(subject)
    subject=input()
print(sorted(X | Y)) # set聯集
print(sorted(X & Y)) # set交集
print(sorted(Y - X)) # set差集
print(sorted(X ^ Y)) # set對稱差集
```

練習：詞典合併

- ❑ 請撰寫一程式，自行輸入兩個詞典(以輸入鍵值"end"作為輸入結束點，詞典中將不包含鍵值"end")，將此兩詞典合併，並根據key值字母由小到大排序輸出，如有重複key值，後輸入的key值將覆蓋前一key值。

練習：詞典合併

Create dict1:

Key: a

Value: apple

Key: b

Value: banana

Key: d

Value: durian

Key: end

Create dict2:

Key: c

Value: cat

Key: e

Value: elephant

Key: end

a: apple

b: banana

c: cat

d: durian

e: elephant

```
def compute():
    dic={}
    while True:
        key=input("Key: ")
        if key=="end":
            return dic
        value=input("Value: ")
        dic[key]=value

print("Create dict1:")
dict1=compute()
print("Create dict2:")
dict2=compute()
merge_dict=dict1.copy()    # 將dict1複製到merge_dict
merge_dict.update(dict2)   # 將dict2合併到merge_dict
sorted_dict=sorted(merge_dict)
for i in sorted_dict:
    print("%s: %s" % (i,merge_dict[i]))
```

練習：詞典搜尋

- ❑ 請撰寫一程式，為一詞典輸入資料(以輸入鍵值"end"作為輸入結束點，詞典中將不包含鍵值"end")，再輸入一鍵值並檢視此鍵值是否存在於該詞典中。

練習：詞典搜尋

- ❑ 請撰寫一程式，為一詞典輸入資料(以輸入鍵值"end"作為輸入結束點，詞典中將不包含鍵值"end")，再輸入一鍵值並檢視此鍵值是否存在於該詞典中。

```
Key: 123-4567-89
Value: Jennifer
Key: 987-6543-21
Value: Tommy
Key: 246-8246-82
Value: Kay
Key: end
Search key: 246-8246-82
True
```

```
dic={}
while True:
    key=input("Key: ")
    if key=="end":
        break
    value=input("Value: ")
    dic[key]=value
search_key=input("Search key: ")
print(search_key in dic)
```

第八類:字串(String)

- ❑ 建立字串
- ❑ 字串處理函式與應用

Python String

❑ Strings 字串

- ❑ Python 基本型態不區分字串 (string) 和字元 (character)

❑ 建立 string

- ❑ (') / (") 單/雙引號 (quote) 包括起來就是字串的設定 (assignment)
- ❑ str(object) 將object轉換成string後回傳

```
str_1='string'
```

```
str_2="string"
```

```
str_3=str(12.34)
```

Python String

❏ Strings 字串的操作

+	Concatenation 連接	str_a + str_b 連接兩個字串
*	Repetition 複製	str_a="*" * 5 → str_4="*****"
[], [:]	Slice 索引運算子	str_a="Python", str_a[0]=="P", 使用方式同串列 list
in, not in	Membership 存在判斷	"P" in "Python" == True, "a" not in "Python" == True
r/R	Raw string 原字串	抑制字串內的 Escape characters, str_a="C:\python\n\\"
%	Format 格式化	%10s, %-10s

Python String

❑ Strings 字符串和 List 串列的不同

Strings are immutable, but lists are mutable.

```
list_a=['apple','banana','orange','broccoli','carrot']
```

```
list_b=[1,3,5,7,9,11,13]
```

```
str_A='applepie'
```

```
str_A[0]='A' # TypeError, strings are immutable
```

```
list_a[0]='cherry'
```

```
print(list_a)
```

```
list_b[3]=8
```

```
list_b[1:3]=[-3,-5]
```

```
print(list_b)
```

Python String

❑ 常用的 Strings 函式

- ❑ `len(str)` - 回傳 `str` 字串長度
- ❑ `max(str)` - 回傳 `str` 字串中最大字元
- ❑ `min(str)` - 回傳 `str` 字串中最小字元
- ❑ `ord(s)` - 回傳該字元的 Unicode code 整數值
- ❑ `str.find(s[, start[, end]])` - 回傳 `str` 字串中出現 `s` 字串的最小索引值
- ❑ `str.index(s[, start[, end]])` - 回傳 `str` 字串中出現 `s` 字串的最小索引值, 若不存在則 `ValueError`
- ❑ `str.count(s)` - 回傳 `str` 字串中出現 `s` 字串的次數

Python String

❑ 常用的 Strings 函式

- ❑ `str.lower()` - 將 `str` 字串中所有字母變小寫回傳 # 原 `str` 字串有沒有改變？
- ❑ `str.upper()` - 將 `str` 字串中所有字母變大寫回傳 # 原 `str` 字串有沒有改變？
- ❑ `str.title()` - 將 `str` 字串中每個單字的第一個字母變大寫回傳
- ❑ `str.replace(old, new[, count])` - 將 `str` 字串中的 `old` 字串以 `new` 取代 `count` 次
- ❑ `str.strip()` - 將 `str` 字串兩側的空白去除回傳 (`lstrip()`, `rstrip()`)
- ❑ `str.center(w)` - 將 `str` 字串以 `w` 寬度置中回傳, 不足補空白 (`ljust()`, `rjust()`)
- ❑ `str.split(sep, maxsplit)` - 將 `str` 字串以 `sep` (預設空白)分隔成串列回傳, `maxsplit` 是分隔次數

例題：字元對應

- ❑ 請撰寫一程式，要求使用者輸入一字串，顯示該字串每個字元的對應ASCII碼及其總和。

範例輸入

Kingdom

範例輸出

```
ASCII code for 'K' is 75
ASCII code for 'i' is 105
ASCII code for 'n' is 110
ASCII code for 'g' is 103
ASCII code for 'd' is 100
ASCII code for 'o' is 111
ASCII code for 'm' is 109
713
```

```
# 字元對應
```

```
total = 0
```

```
string = input()
```

```
for i in range(len(string)):
```

```
    num = ord(string[i]) # return the Unicode code point
```

```
    print("ASCII code for '%s' is %d" % (string[i],num))
```

```
    total += num # 將ASCII碼加總
```

```
print(total)
```


例題：大寫轉換

- ❑ 請撰寫一程式，讓使用者輸入一字串，分別將該字串轉換成全部大寫以及每個字的第一個字母大寫。

範例輸入

```
learning python is funny
```

範例輸出

```
LEARNING PYTHON IS FUNNY  
Learning Python Is Funny
```

大寫轉換

```
myStr = input()
print(myStr.upper()) # 回傳變大寫的字串複本
print(myStr.title()) # 回傳字首大寫的字串複本
```

例題：字元次數計算

- ❑ 請撰寫一程式，讓使用者輸入一字串和一字元，並將此字串及字元作為參數傳遞給名為compute()的函式，此函式將回傳並輸出該字串中指定字元出現的次數，接著再輸出結果。

範例輸入

```
Our country is beautiful  
u
```

範例輸出

```
u occurs 4 time(s)
```

```
# 字元次數計算
```

```
def compute(str,w):
```

```
    return str.count(w) # 計算出現次數
```

```
myStr=input()
```

```
myW=input()
```

```
print("%s occurs %s time(s)" % (myW,compute(myStr,myW)))
```

例題：字串輸出

- 請撰寫一程式，要求使用者輸入一個長度為6的字串，將此字串分別置於10個欄位的寬度的左邊、中間和右邊，並顯示這三個結果，左右皆以直線 | (Vertical bar) 作為邊界。

範例輸入

```
python
```

範例輸出

```
|python|  
|  python  |  
|    python|
```

字串輸出

```
myStr=input()  
if len(myStr)==6:  
    print("|%-10s|" % (myStr))  
    # center()以指定長度將字串置中  
    print("|%s|" % (myStr.center(10)))  
    print("|%10s|" % (myStr))
```

例題：字串加總

- 請撰寫一程式，要求使用者輸入一字串，該字串為五個數字，以空白隔開。請將此五個數字加總(Total)並計算平均(Average)。

範例輸入

```
-2 34 18 29 -56
```

範例輸出

```
Total = 23  
Average = 4.6
```

字串加總

```
s=input()  
slist=[int(x) for x in s.split(" ")]  
print("Total =", sum(slist))  
print("Average =", sum(slist)/len(slist))
```

第九類：檔案與異常處理

- ❑ 文字/二進位檔案建立
- ❑ 檔案寫入與讀取
- ❑ 異常處理

Python File Input / Output

❑ file 處理流程

❑ 以 `open(file, mode, encoding)` 或 `with open() as` : 開啟檔案

❑ 進行檔案處理

❑ 以 `close()` 將檔案關閉 (一定要)

❑ 檔案有中文怎麼辦？

```
file_1 = open("fileName", "w")  
# read, write statement(s)  
file_1.close()
```

```
with open("fileName", "w") as file_2:  
    # read, write statement(s)  
file_2.close() # 不寫也沒關係
```

Python File Input / Output

❑ 檔案模式

- ❑ w: 由檔案開頭寫入, 即覆寫
- ❑ a: 由檔案尾寫入, 即附加
- ❑ +: 讀寫同時

文字檔模式	二進位模式	意義
w / w+	wb / wb+	寫入 / 讀寫
r / r+	rb / rb+	讀取 / 讀寫
a / a+	ab / ab+	附加 / 讀寫

Python File Input / Output

❑ 檔案存取函式

- ❑ `file.write()` - 寫入
- ❑ `file.read(n)` - 從檔案中讀取 `n` 個字元, 若未指定 `n` 則讀取檔案所有內容
- ❑ `file.readline()` - 從檔案一次讀取一行 (“\n”)
- ❑ `file.readlines()` - 讀取檔案所有內容, 以串列回傳檔案內容 (一行 (“\n”) 為一個項目)
- ❑ `file.seek(offset,where)` - 移動檔案指標, `file.seek(0,0)` 移動檔案指標到檔頭

Python File Input / Output

❏ 檔案的操作

```
file_1=open("read.txt","r")
data_1=file_1.read()      # 使用 read()
file_1.seek(0,0)          # read()後, 檔案指標在檔尾, 重新移動到檔頭
data_2=file_1.readlines() # 使用 readlines()
file_1.close()            # 關閉檔案
print(data_1)
print(data_2)             # 回傳串列檔案內容
```

Python File Input / Output

❏ 檔案的操作

```
file_1=open("read.txt","r")
data_1=file_1.readline()      # 使用 readline()先讀一行
while data_1 != "":           # 如果回傳非空字串, 讀檔尚未結束
    print(repr(data_1))        # repr() 顯示跳脫字元
    data_1=file_1.readline()   # readline()讀下一行
file_1.close()                 # 讀檔結束, 關閉檔案
```

Python File Input / Output

❑ 檔案的操作

- ❑ 以 with open() as :
- ❑ Python 官方教材建議使用此敘述, with open() as : 會自動關閉檔案

```
with open("read.txt","r") as file_1:
    data_1=file_1.read()          # 使用 read()
    file_1.seek(0,0)             # read()後, 標案指標在檔尾, 重新移動到檔頭
    data_2=file_1.readlines()    # 使用 readlines()
file_1.close()                   # 關閉檔案, 不寫也可以
print(data_1)
print(data_2)                   # 回傳串列檔案內容
```

Python Exception Handling

❑ 異常處理 (Exception Handling)

程式執行時發生錯誤或異常會怎麼樣？

```
# print out quotient of a/b
```

```
def quotient(a,b):
```

```
    print(a//b)
```

```
quotient(4,2)
```

```
quotient(4,0)    # Division by zero, error! Program stops here.
```

```
quotient(5,2)
```

Python Exception Handling

❏ 異常處理 (Exception Handling)

```
# print out quotient of a/b
def quotient(a,b):
    try:
        print(a//b)
    except ZeroDivisionError:
        print('Error: parameter b can not be 0!')
quotient(4,2)
quotient(4,0)    # handle by excepty, program keeps going...
quotient(5,2)
```

Python Exception Handling

❑ try except else statement

print out quotient of a/b

```
def quotient(a,b):
```

```
    try:
```

```
        print(a//b)
```

```
    except ZeroDivisionError:
```

```
        print('Error: parameter b can not be 0!')
```

```
    except TypeError:
```

```
        ....
```

```
    except:
```

沒有被前述匹被到的異常

```
    else:
```

若沒有異常時執行的敘述

```
    finally:
```

最後一定會執行的敘述

例題：檔案資料加總

- ❑ 請撰寫一程式，讀取read.txt的內容 (內容為數字，以空白分隔) 並將這些數字加總，接著再顯示檔案內容和加總的結果。檔案讀取完成後要關閉。

read.txt:

11 22 33 22 33 44 33 44 55 44 55 66 55 66 77

```
f=open("read.txt","r")  
data=f.read()  
f.close()
```

```
num=data.split(" ") # 空白分隔取成list  
total=0
```

```
for i in range(len(num)):  
    total += eval(num[i])  
print(total)
```

例題：檔案資料計算

- ❑ 請撰寫一程式，讀取read.txt (每一列的格式為名字和身高、體重，以空白分隔)並顯示所有人的平均身高、平均體重以及最高者、最重者。
 - ❑ 提示：輸出浮點數到小數點後第二位。

範例輸出

```
Ben 175 65  
  
Cathy 155 55  
  
Tony 172 75  
Average height: 167.33  
Average weight: 65.00  
The tallest is Ben with 175.00cm  
The heaviest is Tony with 75.00kg
```

read.txt:

Ben 175 65

Cathy 155 55

Tony 172 75

例題：檔案資料計算

```
data=[]
with open("read.txt","r") as file:
    for line in file:
        print(line)
        tmp=line.strip("\n").split(" ")
        tmp=[tmp[0],eval(tmp[1]),eval(tmp[2])]
        data.append(tmp)

name=[data[x][0] for x in range(len(data))]
height=[data[x][1] for x in range(len(data))]
weight=[data[x][2] for x in range(len(data))]
print("Average height: %.2f" % (sum(height)/len(height)))
print("Average weight: %.2f" % (sum(weight)/len(weight)))
max_h=max(height)
max_w=max(weight)
print("The tallest is %s with %.2fcm" % (name[height.index(max_h)],max_h))
print("The heaviest is %s with %.2fkg" % (name[weight.index(max_w)],max_w))
```

例題：成績資料

- ❑ 請撰寫一程式，將使用者輸入的五筆資料寫入到write.txt (若不存在，則讓程式建立它)，每一筆資料為一行，包含學生名字和期末總分，以空白隔開。檔案寫入完成後要關閉。

範例輸入

```
Leon 87  
Ben 90  
Sam 77  
Karen 92  
Kelena 92
```

```
# 成績資料  
file=open("write.txt","w")  
for i in range(5):  
    data=input()  
    file.write(data+"\n")  
file.close()
```

例題：檔案新增資料

- ❑ 請撰寫一程式，要求使用者輸入五個人的名字並加入到data.txt的尾端。之後再顯示此檔案的內容。

範例輸入

```
Daisy  
Kelvin  
Tom  
Joyce  
Sarah
```

範例輸出

```
Append completed!  
Content of "data.txt":  
Ben  
Cathy  
Tony  
Daisy  
Kelvin  
Tom  
Joyce  
Sarah
```

檔案新增資料

```
file=open("data.txt","a+")  
  
for i in range(5):  
    file.write("\n"+input())  
  
print("Append completed!")  
print('Content of "data.txt":')  
  
file.seek(0,0) # seek(offset, from_what)位置控制  
print(file.read())  
  
file.close()
```

例題：學生基本資料

- ❑ 請撰寫一程式，要求使用者讀入read.dat (以UTF-8編)，第一列為欄位名稱，第二列之後是個人記錄。請輸出檔案內容並顯示男生人數和女生人數
- ❑ 提示："性別"欄位，0為女性、1為男性

範例輸出

```
學號 姓名 性別 科系
101 陳小華 0 餐旅管理
202 李小安 1 廣告
303 張小威 1 英文
404 羅小美 0 法文
505 陳小凱 1 日文
Number of males: 3
Number of females: 2
```

例題：學生基本資料

範例輸出

```
學號 姓名 性別 科系
101 陳小華 0 餐旅管理
202 李小安 1 廣告
303 張小威 1 英文
404 羅小美 0 法文
505 陳小凱 1 日文
Number of males: 3
Number of females: 2
```

```
# 學生基本資料
male_num=female_num=0
with open("read.dat","r",encoding="utf-8") as file_1:
    data=file_1.readline() # 使用 encoding="utf-8" 參數
    while data != "":
        print(data)
        row_data=data.split() # 以空白(預設)分隔取資料
        if row_data[2]=="0": # 0為女生
            female_num+=1
        elif row_data[2]=="1": # 1為男生
            male_num+=1
        data=file_1.readline() #讀取下一行
    print("Number of males: ",male_num)
    print("Number of females: ",female_num)
```