python 程式設計

第 一 講

型 別 與 迴 圈

整數與浮點數

- 整數：無位數限制，沒有數字誤差

```
>>> a = 12
>>> b = 9876543210
>>> c = -123456789
```

- 浮點數：有小數點的數字，僅有 15 位有效數字

```
a=-2.3      # -2.3
b=4.5e3     # 4500.     e 或 E 代表10次方
c=3.7E-3    # 0.0037
```

❖ python 將井號(#)之後的文字當作註解

截去誤差 (一)

- 由於計算機使用二進位，其所儲存的數值經常與實際數有些差距，此差距稱為截去誤差 (round-off error)。

  例如 0.1 以二進位表示為循環數 $0.0\overline{0011}_2$ 取有限位數後，差距即為截去誤差。這些差距往往造成計算機的運算結果與實際結果不一樣。

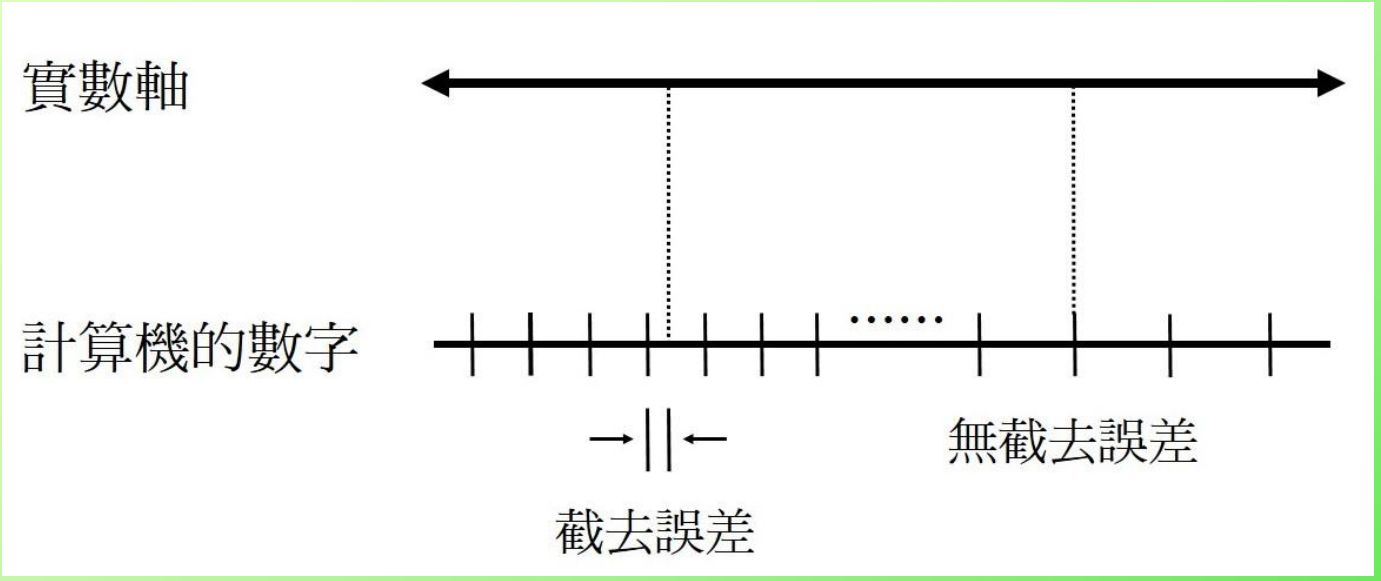
```
0.1 + 0.2 - 0.3          ----> 5.551115123125783e-17
1. + 1.e-20 - 1.         ----> 0
0.1 + 0.1 + 0.1 - 0.3    ----> 5.551115123125783e-17
```

- 有些數沒有截去誤差，例如 0.25 為 1×2^{-2} ，二進位為 0.01_2 ，所以：

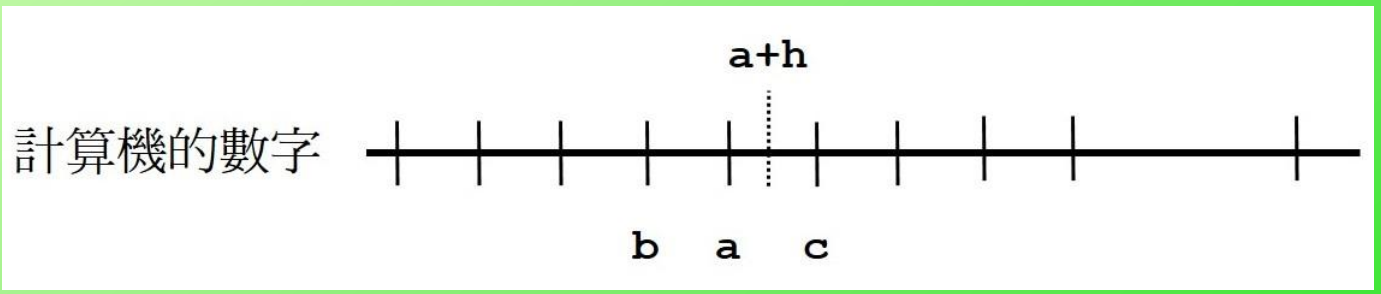
```
0.25 + 0.25 + 0.25 + 0.25 - 1 ---> 0
```

截去誤差 (二)

截去誤差為實數與計算機所代表數之間的差距



當 a 與 c 之間沒有其它數，如果 h 太小，a+h 計算後仍可能等於 a 。



字串

- 字串：被雙引號或單引號框住的字元，跨列字元需使用三個雙(單)引號夾住。

```
a = 'abc'                  # abc
b = "I'll be back"         # I'll be back 雙引號可夾住單一個單引號

c = """ they               # they\nare cats 這裡 \n 為換行字元
are cats"""

d = 'a"b' "c'd"            # a"bc'd 字串可自動合併

e = 'abc' * 3              # abcabcabc 使用乘法複製
f = 'abc' + 'def'          # abcdef    使用加法合併
```

❖ 獨立型式的一組三引號可將數列程式碼夾住即可用來註解程式碼，但第一個三引號仍須遵循縮排規定

資料轉型

- 轉型：以上可使用 **int, float, str** 互相轉型：

```
a = int(57.3)          # 浮點數 --> 整數 57
b = int("34")          # 字串   --> 整數 34

c = float(23)          # 整數   --> 浮點數 23.
d = float('3.23')      # 字串   --> 浮點數 3.23

e = str(35)            # 整數   --> 字串 "35"
f = str(3.14)          # 浮點數 --> 字串 "3.14"
```

字串長度：len 函式

```
a = len("abc")              # len(字串) 回傳字串長度

n = 2349
b = len(str(n))             # b 為整數 n 的位數
```

指定多筆資料

- 指定多筆資料:使用逗號分離資料

```
a , b = 5 , 2                  # a = 5 , b = 2
a , b = a + b , a - b          # a = 7 , b = 3

c , d , e = 2 ,7.5 , "cat"     # c = 2 ,d = 7.5 , e = "cat"

x , y = 4 , "four"             # x = 4 , y = "four"
x , y = y , x                  # 對調 x 與 y 資料
                               # x = "four" , y = 4
```

❖ python 變數型別可隨時更動，並不是固定不變的

輸出

- **print**：輸出資料，預設為印完後自動換列

```
print()                          # 空印一列
print(3)                         # 輸出 3，印完後自動換列
print(3,end="")                  # 輸出 3，印完後不換列
print(3,end="cats")              # 輸出 3cats，印完後不換列
print(3,end="\n\n")              # 輸出 3，印完後多換一列
```

多筆資料列印：

```
print(3,5,7)                     # 輸出 3 5 7，資料間有空格分開
print(3,5,7,sep='')              # 輸出 357，資料擠在一起
print(3,5,sep='-',end="")        # 輸出 3-5，資料有橫線分開，印完後不換行
print('/'+'\\'*3)                # 輸出 斜線與三個反斜線
```

❖ 以上反斜線(\)為特殊字元，使用時需多加一個反斜線字元

輸入

- **input:** 讀取資料成為字串

```
a = input()                    # 將輸入字串資料存入 a
b = input(">>>")               # 先輸出 >>> 於螢幕，之後將輸入字串存於 b
c = int( input("> ") )         # 將輸入的資料轉型為整數後存於 c
```

輸入

- **input: 一次讀入多筆資料**
  - 將 **eval** 函式包裹 **input** 式子
  - 輸入的資料要用「**逗號**」分離
  - 輸入資料量不限，但等號左側要有同等數量的變數
  - 資料經過處理後會自動轉型
  - 若資料包含字串時，字串要用單(雙)引號夾住

```
>>> a , b , c = eval( input("> ") )
> 3 , "cat" , 2.8                    # 字串要有單(雙)引號
>>> a                                 # a 為整數 3
3
>>> b                                 # b 為字串 'cat'
'cat'
>>> c                                 # c 為浮點數 2.8
2.8
```

❖ **若資料是以空格分離則可參考第 ?? 頁中的用法**

基本運算符號 (一)

符　號	運 算 子	範　例
+ - * /	加 減 乘 除	3+4 = 7
%	餘數運算	7%3 = 1
//	floor 除法	7//3 = 2, 7.5//3 = 2
**	指數運算	3**2 = 9
+= -= *= /= %=	複合運算	a += 4 → a = a + 4
<< >>	位元左右位移	1<<2 = 4 , 7>>1 = 3

❖ // 為特殊的除法運算，回傳去除小數部份的計算結果。

基本運算符號 (二)

- `//` 為特殊的除法運算，回傳去除小數部份的計算結果
- `a += b` 是 `a = a + b` 的省寫法
- `a << n` 是將 `a` 的二進位表示方式向左移動 `n` 個位置，例如：

```
5 << 2 等同 101 << 2 等同 10100 等同 20
6 >> 1 等同 110 >> 1 等同    11 等同 3
```

以上 5 的二進位表示為 101_2 $(1\times2^2+0\times2^1+1\times2^0)$，
向左移兩位等同 10100_2 $(1\times2^4+0\times2^3+1\times2^2+0\times2^1+0\times2^0)$，化成十進位後得 20。

跨列式子：

- 使用小括號

```
a = ( 1  + 1 +  2 +  3 +
      5  + 8 + 13 + 21 +
      34 + 55 )
```

- 使用反斜線於列尾

```
a = 1  + 1 +  2 +  3 + \
    5  + 8 + 13 + 21 + \
    34 + 55
```

❖ 反斜線之後不得有任何空格

range (一)

- **range：可用來產生等差整數數列**
  - **range(a)：可依次產生 {0,1, ··· ,a-1} 等整數，共有 a 個數**
  - **range(a, b)：可依次產生 {a,a+1, ···,b-1} 等整數，共有 b-a 個數**
  - **range(a, b, c)：可依次產生 {a,a+c,a+2c, ···} 等數字，若為遞增數列，最大數字比 b 小，若為遞減數列，最小數字比 b 大。**

range (二)

```
range(4)          ---> 0 1 2 3   共四個數
range(5,8)        ---> 5 6 7     共三個數 (8-5)
range(1,5,2)      ---> 1 3
range(1,6,2)      ---> 1 3 5
range(5,-1,-1)  ---> 5 4 3 2 1 0
range(5,0,-1)   ---> 5 4 3 2 1

range(1,4,0.5) ---> 錯誤，僅能產生整數
```

for 迴圈 (一)

- **for 迴圈：重複執行式子**
  - **for 迴圈經常與 range 合併使用**

```
for i in range(1,4) : print(i,i*i)
```

輸出：

```
1 1
2 4
3 9
```

  - **迴圈末尾有冒號，之後才是要重複執行的式子。**
  - **若有多個式子，則須跳列並使用定位鍵(tab 鍵）加以縮排：**

```
for i in range(1,4) :
    j = i**2                          # j 的前面是使用定位鍵
    print( i , '平方 =' , j )  # print 的前面是使用定位鍵
```

for 迴圈 (二)

- 產生前 n 個 Fibonacci 數字

```
n = int(input("> "))
a , b = 1 , 1
print( a , b , end =" " )

for i in range(n-2) :
    a , b = b , a + b
    print( b , end=" " )
```

輸出:

```
> 15
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
```

for 迴圈 (三)

- 鑽石圖形:兩個迴圈

```
n = int(input("> "))
for i in range(n) :
    print( ' ' * (n-1-i)+ '*' * (2*i+1) )

for i in range(n-2,-1,-1) :
    print( ' ' * (n-1-i)+ '*' * (2*i+1) )
```

輸出為:

```
> 3
  *
 ***
*****
 ***
  *
```

```
> 4
   *
  ***
 *****
*******
 *****
  ***
   *
```

多層迴圈 (一)

- 性質:
  - 迴圈內另有迴圈，各層迴圈需要縮排
  - 迭代速度：外層迴圈迭代一步，內層迴迭次一圈
  - 外層迴圈：迭代慢，如同時針
  - 內層迴圈：迭代快，如同分針
  - 各層迴圈的執行次數可根據問題需要自由變更
  - 多數問題的內外層迴圈通常不能互換

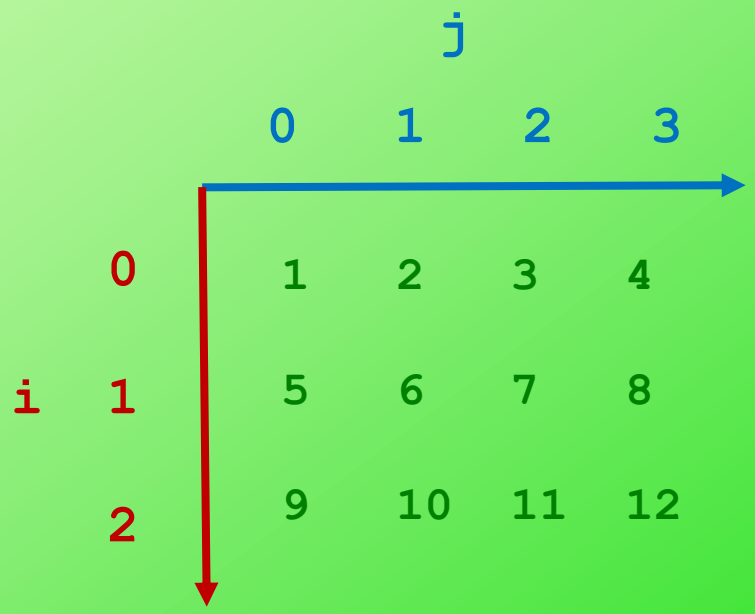
多層迴圈 (二)

- 固定的迴圈執行次數
  - 雙重迴圈

```
k = 1
for i in range(3) :

    for j in range(4) :
        print(k,end=" ")
        k += 1

    print()
```



❖ 此例中當橫列 i 數值變化時(即 j 迴圈結束後)才進行換行動作

多層迴圈 (三)

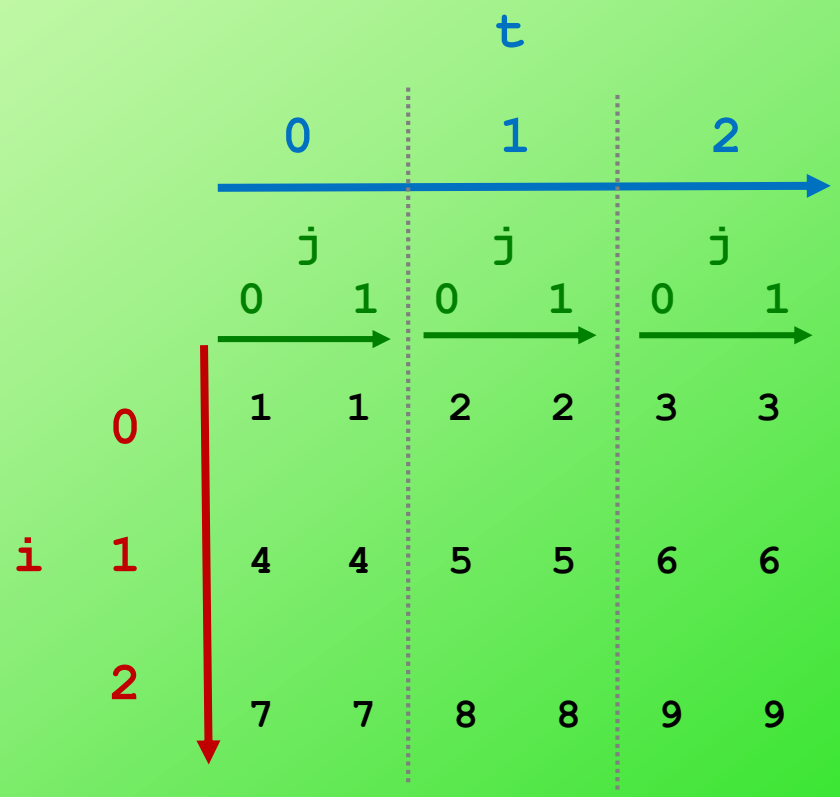
- 三重迴圈:

```
k = 1
for i in range(3) :

    for t in range(3) :

        for j in range(2) :
            print(k,end=" ")
        k += 1

    print()
```



❖ 此例中當橫列 i 數值變化時(即 t 迴圈結束後)才進行換行動作

多層迴圈 (四)

- 四重迴圈:

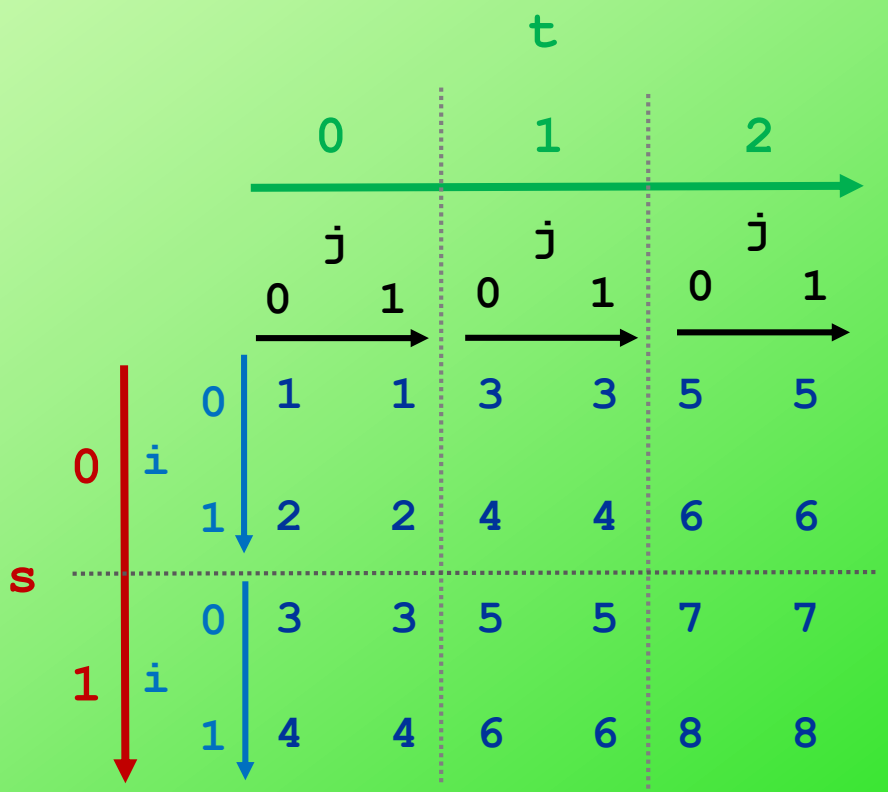
```
for s in range(2) :

    for i in range(2) :

        k = 2*s + i + 1
        for t in range(3) :

            for j in range(2) :
                print(k,end=" ")
            k+=2

        print()
```



❖ 此例中當橫列 i 數值變化時(即 t 迴圈結束後)才進行換行動作

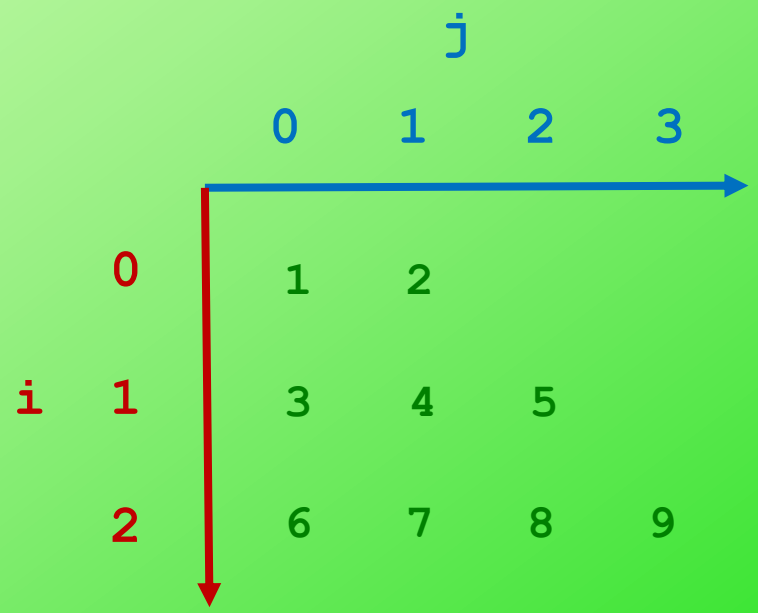
多層迴圈 (五)

- 變化的迴圈執行次數
  - 雙重迴圈：變化的內迴圈執行次數

```
k = 1
for i in range(3) :

    # 各列 j 迴圈執行次數是 i+2
    for j in range(i+2) :
        print(k,end=" ")
        k += 1

    print()
```



多層迴圈 (六)

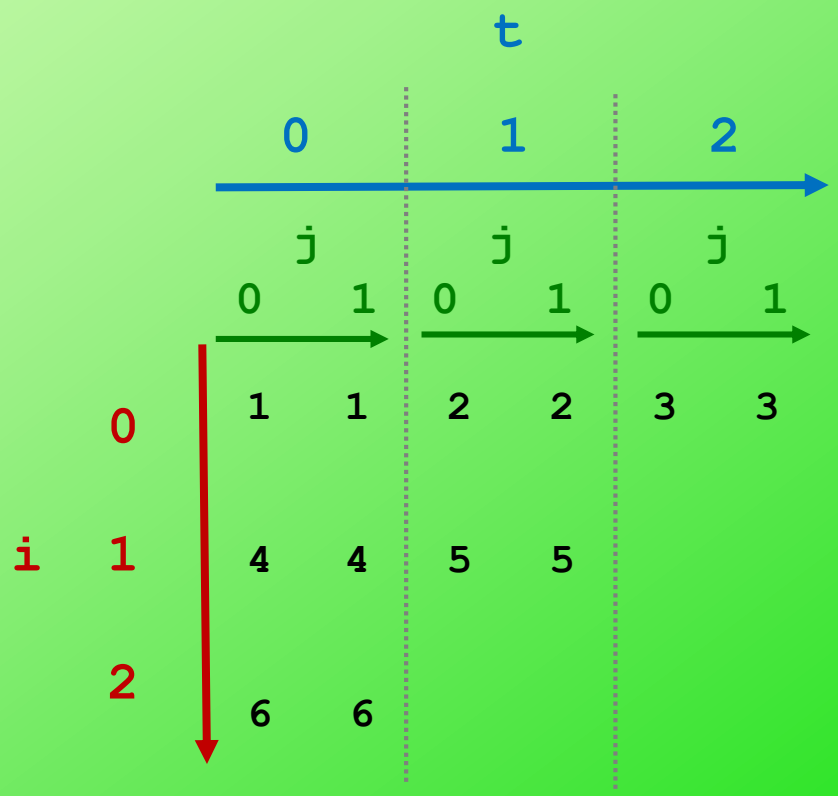
- 三重迴圈：變化的 t 迴圈執行次數

```
k = 1
for i in range(3) :

    # 各列 t 迴圈執行次數是 3 2 1
    for t in range(2-i,-1,-1) :

        for j in range(2) :
            print(k,end=" ")
        k += 1

    print()
```



多層迴圈 (七)

- 印出 3 x 4 的乘法表：使用雙層迴圈

```
for i in range(1,4) :
    for j in range(1,5) :
        print( i , 'x' , j , '=' , i*j , end=' ' )
    print()
```

輸出：

```
1 x 1 = 1  1 x 2 = 2  1 x 3 = 3  1 x 4 = 4
2 x 1 = 2  2 x 2 = 4  2 x 3 = 6  2 x 4 = 8
3 x 1 = 3  3 x 2 = 6  3 x 3 = 9  3 x 4 = 12
```

多層迴圈 (八)

- 下三角乘法表：動態調整內層迴圈執行次數

```
for i in range(1,4) :
    for j in range(1,i+1) :
        print( i , 'x' , j , '=' , i*j , end=' ' )
    print()
```

輸出:

```
1 x 1 = 1
2 x 1 = 2  2 x 2 = 4
3 x 1 = 3  3 x 2 = 6  3 x 3 = 9
```

練習題 (一)

學好程式需要大量練習，在練習中才能由中開發適合於自己邏輯思維的一套解題方式。

本章的重點是迴圈，迴圈代表著「有規則的重複步驟」。一個程式問題通常是由一些有規則步驟與一些沒有規則步驟交錯組合而成，有規則步驟使用迴圈替代，沒有規則步驟則用一般式子。

在動手寫程式前先由紙筆推導入手，規則理清楚，轉為程式碼就簡單了。程式之道無他，就只是多多練習而已。

練習題 (二)

1. 輸入數字 n 產生以下階乘輸出:

```
> 5
1! = 1 = 1
2! = 1x2 = 2
3! = 1x2x3 = 6
4! = 1x2x3x4 = 24
5! = 1x2x3x4x5 = 120
```

練習題 (三)

2. 輸入數字 n 產生以下數字和輸出:

```
> 5
sum([1,5]) = 1+2+3+4+5 = 15
sum([1,4]) = 1+2+3+4 = 10
sum([1,3]) = 1+2+3 = 6
sum([1,2]) = 1+2 = 3
sum([1,1]) = 1 = 1
```

練習題 (四)

3. 輸入列數 n 產生 n×n 方形數字圖形:

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

```
> 4
1 2 3 4
5 6 7 8
9 0 1 2
3 4 5 6
```

練習題 (五)

4. 輸入列數 n 產生 n×n 方形空心數字圖形:

```
> 3
1 2 3
4   5
6 7 8
```

```
> 4
1 2 3 4
5     6
7     8
9 0 1 2
```

練習題 (六)

5. 輸入列數 n 產生 n×n 環形數字圖形:

```
> 4
1 1 1 2
4     2
4     2
4 3 3 3
```

```
> 5
1 1 1 1 2
4       2
4       2
4       2
4 3 3 3 3
```

練習題 (七)

6. 輸入列數 n 產生 n×n 環形遞增數字圖形:

```
> 4
1 2 3 4
2     5
1     6
0 9 8 7
```

```
> 5
1 2 3 4 5
6       6
5       7
4       8
3 2 1 0 9
```

練習題 (八)

7. 輸入列數產生右下三角數字圖案:

```
> 3
    1
  2 3
4 5 6
```

```
> 4
      1
    2 3
  4 5 6
7 8 9 0
```

練習題 (九)

8. 輸入列數產生以下數字圖案:

```
> 3
1           1
2 2       2 2
3 3 3 3 3 3
```

```
> 4
1               1
2 2           2 2
3 3 3       3 3 3
4 4 4 4 4 4 4 4
```

練習題 (十)

9. 輸入列數 n 產生數字圖形:

```
> 4
1 2 3 4
2 3 4 1
3 4 1 2
4 1 2 3
```

```
> 5
1 2 3 4 5
2 3 4 5 1
3 4 5 1 2
4 5 1 2 3
5 1 2 3 4
```

練習題 (十一)

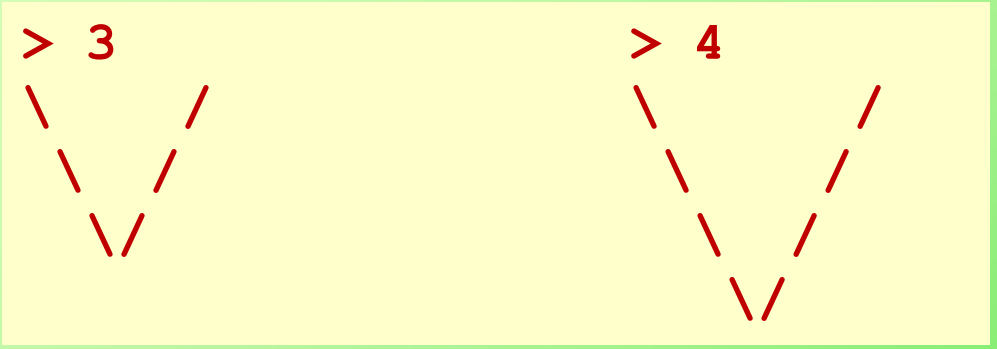
10. 輸入列數 n 產生數字排列圖形 :

```
> 3
|123|231|312|
|231|312|123|
|312|123|231|
```

```
> 4
|1234|2341|3412|4123|
|2341|3412|4123|1234|
|3412|4123|1234|2341|
|4123|1234|2341|3412|
```

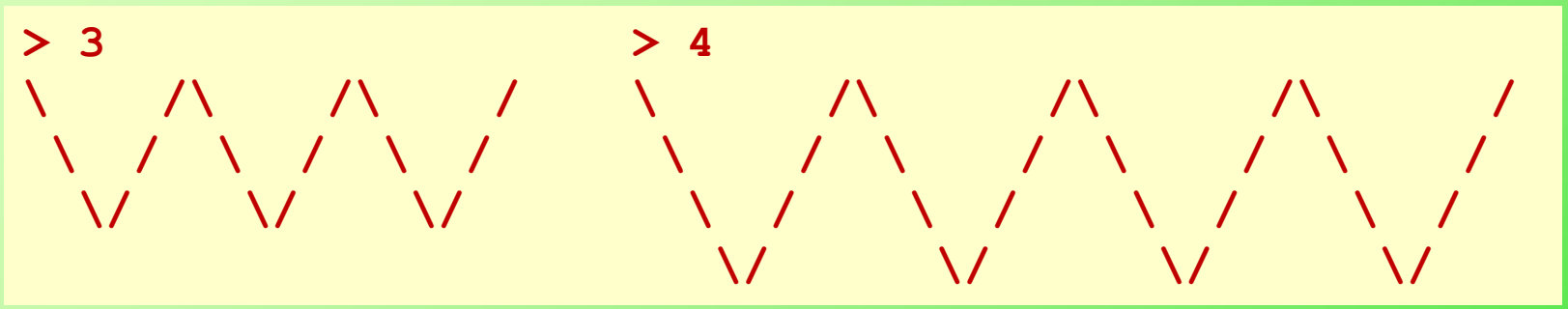
練習題 (十二)

11. 輸入列數產生以下 v 圖案:



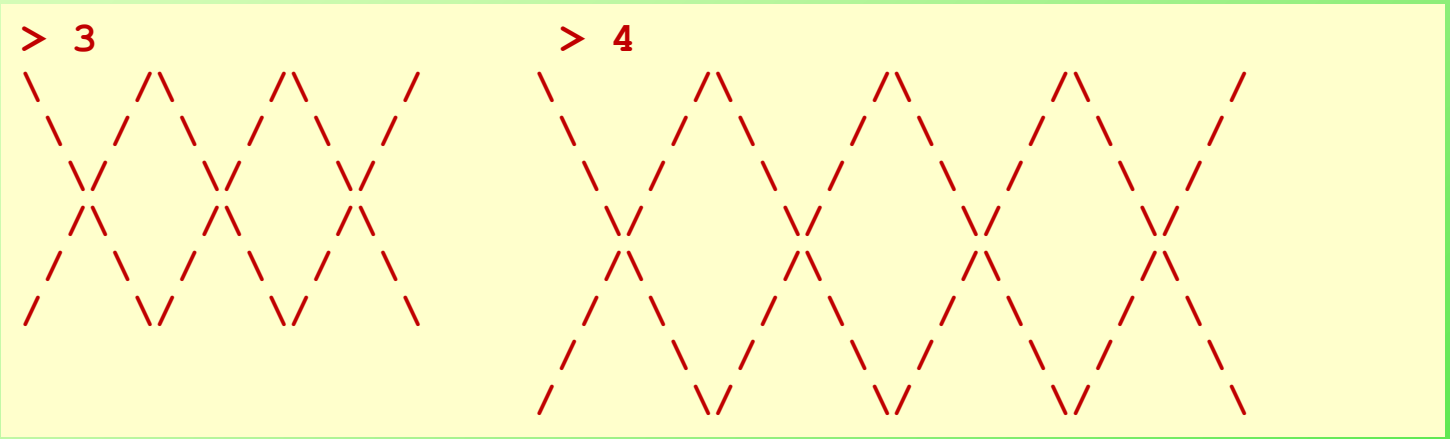
練習題 (十三)

12. 輸入列數 n 產生 n 個連在一起的 V 圖案:



練習題 (十四)

13. 輸入列數 n 產生 n 個連在一起的 X 圖案:



練習題 (十五)

14. 輸入列數 n 產生 n 個連在一起菱形圖案，並在各個菱形框內有數字:

```
> 3
\    /\    /\    /
1\  /22\  /33\  /4
11\/2222\/3333\/44
11/\2222/\3333/\44
1/  \22/  \33/  \4
/    \/    \/    \
```

```
> 4
 \      /\      /\      /\      /
1\    /22\    /33\    /44\    /5
11\  /2222\  /3333\  /4444\  /55
111\/222222\/333333\/444444\/555
111/\222222/\333333/\444444/\555
11/  \2222/  \3333/  \4444/  \55
1/    \22/    \33/    \44/    \5
/      \/      \/      \/      \
```

練習題 (十六)

15. 輸入列數 n 產生以下 n 個方塊數字:

```
> 3
111/2 333/4 555/6
11/22 33/44 55/66
1/222 3/444 5/666
```

```
> 4
1111/2 3333/4 5555/6 7777/8
111/22 333/44 555/66 777/88
11/222 33/444 55/666 77/888
1/2222 3/4444 5/6666 7/8888
```

練習題 (十七)

16. 撰寫程式利用四重迴圈，產生以下數字排列：

```
1 1 1 1 1  2 2 2 2 2  3 3 3 3 3  4 4 4 4 4
1 1 1 1 1  2 2 2 2 2  3 3 3 3 3  4 4 4 4 4
1 1 1 1 1  2 2 2 2 2  3 3 3 3 3  4 4 4 4 4

5 5 5 5 5  6 6 6 6 6  7 7 7 7 7  8 8 8 8 8
5 5 5 5 5  6 6 6 6 6  7 7 7 7 7  8 8 8 8 8
5 5 5 5 5  6 6 6 6 6  7 7 7 7 7  8 8 8 8 8
```

練習題 (十八)

17. 撰寫程式產生以下數字排列:

```
> 4
1111
1111 2222
1111 2222 3333
1111 2222 3333 4444
```

```
> 5
11111
11111 22222
11111 22222 33333
11111 22222 33333 44444
11111 22222 33333 44444 55555
```

練習題 (十九)

18. 撰寫程式產生以下數字排列:

```
> 4
1111 2222 3333 4444
     2222 3333 4444
          3333 4444
               4444
```

```
> 5
11111 22222 33333 44444 55555
      22222 33333 44444 55555
            33333 44444 55555
                  44444 55555
                        55555
```

練習題 (二十)

19. 撰寫程式產生以下方塊數字排列:

```
> 2
1 1
1 1

2 2  3 3
2 2  3 3
```

```
> 3
1 1 1
1 1 1
1 1 1

2 2 2  3 3 3
2 2 2  3 3 3
2 2 2  3 3 3

4 4 4  5 5 5  6 6 6
4 4 4  5 5 5  6 6 6
4 4 4  5 5 5  6 6 6
```

練習題 (二十一)

20. 輸入列數 n 產生以下個位數數字圖案:

```
> 4
1-----------2
  3--------4
    5---6
      7
```

```
> 5
1----------------2
  3------------4
    5--------6
      7---8
        9
```

練習題 (二十二)

21. 輸入列數 n 產生以下雙數字圖案:

```
> 4
1-----------2-----------3
  4-------5   6-------7
    8---9       0---1
      2           3
```

```
> 5
1---------------2---------------3
  4-----------5   6-----------7
    8-------9       0-------1
      2---3           4---5
        6               7
```

練習題 (二十三)

22. 輸入列數 n 產生以下 n 個數字塔:

```
> 3
  1     1     1
 222   222   222
33333 33333 33333
```

```
> 4
   1       1       1       1
  222     222     222     222
 33333   33333   33333   33333
4444444 4444444 4444444 4444444
```

練習題 (二十四)

23. 輸入列數 n 產生以下 n 個數字空心塔:

```
> 3
  1     1     1
 2 2   2 2   2 2
33333 33333 33333
```

```
> 4
   1       1       1       1
  2 2     2 2     2 2     2 2
 3   3   3   3   3   3   3   3
4444444 4444444 4444444 4444444
```

練習題 (二十五)

24. 輸入列數，產生以下一排山：

```
> 3
  /\    /\    /\
 /\/\  /\/\  /\/\
/\/\/\/\/\/\/\/\
```

```
> 4
    /\      /\      /\      /\
   /\/\    /\/\    /\/\    /\/\
  /\/\/\  /\/\/\  /\/\/\  /\/\/\
 /\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
```

練習題 (二十六)

25. 輸入列數，產生以下山與其倒影圖案:

```
> 3
  /\    /\    /\
 /\/\  /\/\  /\/\
/\/\/\/\/\/\/\/\/\
~~~~~~~~~~~~~~~~~~~
\/\/\/\/\/\/\/\/\/
 \/\/  \/\/  \/\/
  \/    \/    \/
```

```
> 4
    /\      /\      /\      /\
   /\/\    /\/\    /\/\    /\/\
  /\/\/\  /\/\/\  /\/\/\  /\/\/\
 /\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 \/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
  \/\/\/  \/\/\/  \/\/\/  \/\/\/
   \/\/    \/\/    \/\/    \/\/
    \/      \/      \/      \/
```