認識 Python 基本語法

4
CHAPTER

4-1 輸出輸入指令

通常程式在執行的時候都需要輸入資料，要把資料送到程式中有兩種方法：第一種方法就是讓使用者從鍵盤輸入資料 input() 指令；另外一種方法就是在程式中設定，如：m=5；n=10…等等，本書在範例中盡量在程式中設定資料數值，以免使用者輸入錯誤造成困擾。

4.1.1 print 輸出指令

其實 print 是一個函數，後面會解釋什麼叫做函數，print 指令提供三個參數，格式如下：

```
print（變數：Sep="分隔符號"，end="結束符號"）
```

一般使用上，會輸出多個變數，就用逗號隔開，例如：

```
a=3
b=2
print(a,b)
print(a+b)
print(a,b,sep="%")
```

輸出結果是：

```
3 2
5
3%2
```

這代表著分隔符號的預設是空白，另一個結束符號的預設是 '\n' 代表換行。print() 會讓列印自動跳下一行，等於 print ("\n") ，九九乘法表程式是讓你了解資料不跳行跟著印 (end='') 和印出後跳下一行最好的練習，為了精準控制畫面欄位列印格式，format 指令也都需要清楚了解。

下面舉出五個九九乘法的程式供各位觀摩練習（下載程式：e-5-9x9.py）：

```
for i in range(1,10):
    for j in range (1,10):
        print (i*j , end=' ')
    print()
```

```
for i in range(1,10):
    for j in range (1,10):
        print ("%3d" %(i*j) , end='')
    print()
```

```
for i in range(1,10):
    for j in range (1,10):
        print (i*j , end=' ')
    print(end ='\n')
```

```
for i in range(1,10):
    for j in range (1,10):
        if i*j < 10 :
            print ('', i*j, end=' ')
        else :
            print (i*j , end=' ')
    print(end ='\n')
```

```
for i in range(1,10):
    for j in range (1,10):
        if i*j < 10 :
            print (i,'x',j,'= ',i*j, end=' ')
        else :
            print (i,'x',j,'=',i*j , end=' ')
    print()
```

如果要在字串中放進變數輸出，可使用下列的三個方式。假設有下列輸入：

【練習範例】下載程式：阿珠的成績.py

```
a=65
b=92
print('阿珠的成績是:' +str(a)+' 阿花的成績是：'+str(b))
print('阿珠的成績是:%3d,阿花的成績是:%3d' %(a,b))
print('阿珠的成績是: {}, 阿花的成績是: {}'.format(str(a),str(b)))
```

【執行結果】

```
阿珠的成績是:65 阿花的成績是：92
阿珠的成績是: 65,阿花的成績是: 92
阿珠的成績是: 65, 阿花的成績是: 92
```

【指令說明】

1. 用"+"連結字串輸出

```
print('阿珠的成績是: "+str(a)+"，阿花的成績是：'+str(b))
```

2. 用參數格式化的方式：%d 代表整數輸出

```
print('阿珠的成績是: %3d , 阿花的成績是: %3d' %(a,b))
```

3. 用字串的 format() 函數輸出

```
print('阿珠的成績是: {}, 阿花的成績是: {}'.format(str(a),str(b)))
```

4.1.2 跳脫字元（Escape）

和 C 語言類似，Python 也有跳脫字元，所謂的跳脫字元就是 \ 再加上一些特定符號，如：\n 用來列印打不出來或會被誤解的特殊符號。以下列出各種跳脫（Escape）字元：

跳脫字元	代表字元	跳脫字元	代表字元
\\	反斜線\	\n	換行
\'	單引號'	\r	Return
\"	雙引號"	\t	定位
\a	喇叭嗶聲	\ooo	8 進制
\b	退位	\xhh	16 進制
\f	下跳一頁		

【指令練習】

```
>>> print('1/4')
1/4
>>> print('1\\4')
1\4
>>> print('\name')                # \n 印出時跳下一行

ame
>>> print('\\name')
\name
>>> c='abcde'
>>> print('\tJerome %s' % c)      # %s 字串格式
  Jerome abcde
>>> print('\tJerome')
  Jerome
>>> print("It\'s my \"pleasure\")

SyntaxError: EOL while scanning string literal
>>> print("It\'s my \"pleasure\"")

It's my "pleasure"
>>> print('印出單引號\'')


印出單引號'
>>> print("\\101 is \101.\n")     # \101(八進位) = 65(十進位)

\101 is A.

>>> print("\\x41 is \x41")

\x41 is A
```

4.1.3 格式化輸出及%用法

參數	格式描述	實例練習

%%	輸出一個%	>>> a=123.45
		>>> print('%6.2f %%' %a)
		123.45 %

%c	字符或 ASCII 碼	>>> print ("%c" % 65) A
%s	字符串	>>> print("I'm %s. I'm %d year old" % ('Hom', 30)) I'm Hom. I'm 30 year old
%o	八進制整數	>>> print ("%4o" % 10) 12
%d	十進制整數	>>> print ("%4d" % 10) 10

參數	格式描述	實例練習
		>>> print ("%04d" % 10) 0010
%x	十六進制整數	>>> print ("%+10x" % 10) +a
%f	浮點數	>>> print ("%6.3f" % 2.3) 2.300
%e	浮點數（科學計數法）	>>> print('%e' % 1.11) 1.110000e+00

% 為修飾格式輸出符號

【資料格式】

a=1234；b=12.34

資料	格式	印出結果									
a	%8d					1	2	3	4		
a	%+d	+	1	2	3	4					
a	%-8d	1	2	3	4						
a	%d	1	2	3	4						
a	% d		1	2	3	4					
a	%010d	0	0	0	0	0	0	1	2	3	4
b	%7.2f			1	2	.	3	4			
b	%010.3f	0	0	0	0	1	2	.	3	4	0
b	%+10.4f			+	1	2	.	3	4	0	0

【指令練習】

```
>>> print("%8d" %a)
1234
>>> print("%+d" %a)
+1234
>>> print("%-8d" %a)
1234
>>> print("%d" %a)
1234
```

```
>>> print("% d" %a)
1234
>>> print("%010d" %a)
0000001234
>>> print("%7.2f" %b)
  12.34
>>> print("%010.3f" %b)
000012.340
>>> print("%+10.4f" %b)
  +12.3400
>>>
```

4.1.4 input 輸入指令

input 指令又有兩種輸入方法：「隱性輸入」和「顯性輸入」。隱性輸入就是 n= input('') 的指令，執行後畫面會出現游標 | 等待輸入，一般人會不知道要輸入什麼資料；顯性輸入在輸入時會先列印出提醒文字，例如：

1. 隱性輸入

```
s = input( " )          # 這個時候一直輸入的是字串
n = int(input( " ))     # 輸入的是整數
```

2. 顯性輸入

```
s = input('請輸入一個數值( 0<s<100)')          # 這個時候一直輸入的是字串
n=int( input('請輸入一個數值( 0<s<100)')       # 這個時候一直輸入的是數值
```

【範例說明】

(1) 有提示字的輸入

【程式範例】

```
name = input('請輸入你的名字：')
print('歡迎 ', name)
```

【執行結果】

```
請輸入你的名字：Jerome
歡迎  Jerome
```

(2) 沒有提示字的輸入，計算總和（sum-0.py）

【程式範例】

```
n=int(input())  # 執行後畫面會出現游標 | 等待輸入
s=0
for i in range(n+1):
    s=s+i
print(s)
```

【執行結果】

```
10    (請輸入 10)
55
```

3. 這個程式也可以修改成輸出入都有提示字元（sum-1.py）

【程式範例】

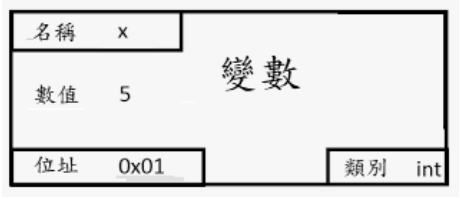
```
n=int(input("請輸入一整數 n="))
s=0
for i in range(n+1):
    s=s+i
print(" 1 到 ",n, "的總和=", s)
```

【執行結果】

```
請輸入一整數 n=10    (請輸入 10)
 1 到 10 的總和= 55
```

4-2 Python 程式內涵簡介

內涵所指的是資料和指令，要練習撰寫 Python 程式之前，希望能夠先對 Python 語言的元素有一個粗淺的認識，下面每一項在最後面的篇章中都有更詳細探討。在這一節中各位只要了解初步概念，知道有這些元素存在，至於詳細的使用方法後面會再討論。電腦的資料就是靠變數的傳遞，變數的屬性重要的如下圖所示，包括變數位址、名稱、類別、數值等。



4.2.1 數

數就是數值，也就是一般認為的數字。「常數」和「變數」的差別是：常數用在程式裡面，並沒有存放在記憶體，用完了就沒有了；變數會存在記憶體裡面，存放位置為了方便使用，就用變數來表示。

1. 常數：是固定不變的數，在寫程式的時候所輸進去的數字都是常數。
2. 變數：就是用英文字母代替數值的變數，像在代數學有 x 和 y 都是變數。

當各位在參考國外的程式時喜歡用很長的名字或者底線 _ 當變數名稱的字元，主要是他們熟悉英文單字，國內學生對名稱比較長的變數不太適應，也會和保留字混用，所以本書在用變數的時候都是用最精簡的字母，例如：a,b,c,,,i,j,k,,,m,n,,,x,y,z,, 等等。

4.2.2 字

電腦可以處理的字包括：數字、英文字母、中文字、符號等等。Python 定義字元是將字元和字串用單引號 ' 或雙引號 " 括弧起來，在 IDLE 編輯器上呈現的是綠色，這樣有助於你了解程式結構是否正確。

1. 字元：一個字就是字元，像：'1', '2', '3',,,,, 'a', 'b', 'c'
2. 字串：很多字元會組成字串，像：'abc', '123', '水果', '星期一'

字串字元也有常數跟變數，例如：

1. 常數： "my name is Jerome" 就是字串常數。
2. 變數：s="my name is Jerome"，s 就是字串變數。

4.2.3 邏輯

Python 中也提供幾個基本布林運算所需的「邏輯運算子（logical operator）」，分別為「且（and ）」、「或（or ）」及「反相（not）」三個運算子。

1. 常數邏輯的值只有 True 或者 False 兩者之一。

2. 變數 a=True，那 a 就是存放邏輯值的變數，下面這個例子可以說明 a 所存放的邏輯值是真（True）。

【程式範例】

```
a=True
if (a) :
    print(a)
```

【執行結果】

```
True
```

4.2.4 運算

運算子（operator）的功用是結合物件，組成運算式（expression），以計算某些結果；物件又稱為運算元（operand）。

小學時候大家都學過算術四則運算加減乘除，現在到了電腦，大部分程式語言都提供 7 則運算：加（+）、減（-）、乘（*）、除（/）、餘（%）、整商（//）、次方（**）等，下面這行式子代表這算術運算其中有運算子和運算元：

```
a=b+c*2-8
```

1. 運算子：就像 +、 *、 - 等等運算符號
2. 運算元：b、c、2、8 就是運算的元素

4.2.5 運算的種類

一般程式語言所執行的運算種類包括：數值運算、字串運算、邏輯運算和比較運算。

1. 數值運算：+、 -、 *、/、%、//、**
2. 字串運算：+、*、[]
3. 邏輯運算：and 、or、not
4. 比較運算：< >、==、 ! =、< >、>=、<=
5. 位元（布林）運算：&、|、~

4.2.6 指令

不少人認為程式設計很難，其實程式語言除運算之外就只有：判斷和迴圈二種指令，至於函數也是由上述三種指令完成。

1. 設定：例如：a=6 就是設定把右邊的 6 存到左邊的 a 變數中。
2. 判斷：if 指令，運算式也是用來決定一則運算式在成立的時候要執行哪些程式，不成立的時候又要執行哪些程式。
3. 迴圈：有 for 和 while 兩個指令。for 一般用在已經知道起始值和終止值的情況，while 用在不確定執行次數的情況，在程式中使用 continue 和 break 決定何時跳出迴圈。

4.2.7 資料型別

Python 在處理這些資料型別時還包括下列幾項：

- 數字（digit）：如 1234
- 字元（character）：如 'a'、'b'、'c'、'1'、'2'、'3'
- 字串（string）：如 'asd3234'

有一個資料型別是有次序性的資料又稱容器（container），資料型態如下，在其他語言就是陣列：

- 列表（list）：如 [1，'abc',(1,3),[2,3]]
- 元組（tuple）：如 (1，'abc',[1,3])
- 字典（dictionary）：如 {'key1'：'values','key2':'very good'}
- 集合（set）：如 {1，'abc',(1,3)}

4.2.8 函數

函數在數學中為兩集合間的一種對應關係，輸入值的每項元素皆能對應一項輸出值。實數 x 對應到其平方 x^2 的關係就是一個函數，若以 3 作為此函數的輸入值，pow(3,2) 所得的輸出值便是 9。