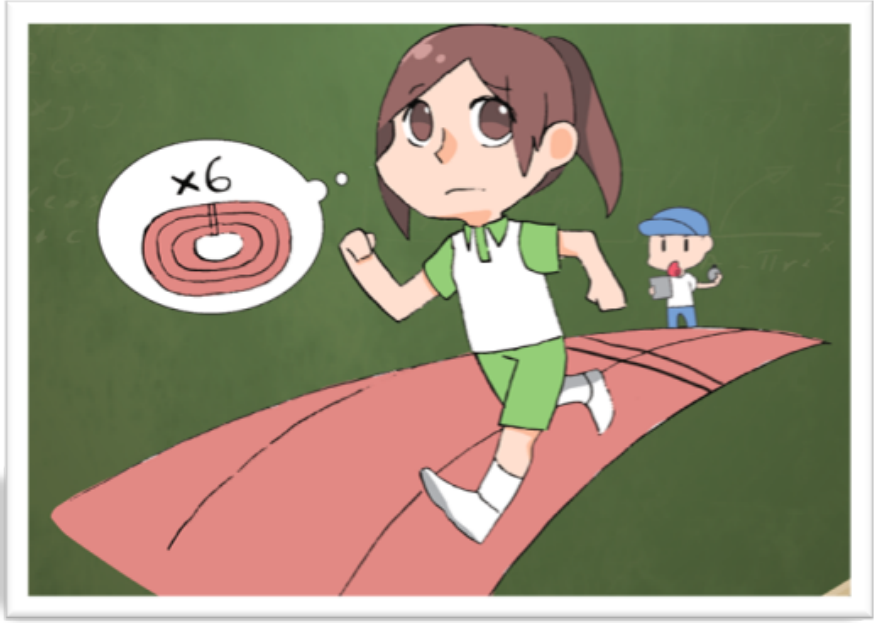
重複主題（二）：計數迴圈

by 田弘華 **Hung-Hua Tien**

1. 重複迴圈

我們在聽音樂時，如果聽到喜歡的歌曲，可以設定讓這首歌持續播放，而不用每次動手重新播放音樂。也好比上體育課時，老師叫我們跑操場，而且一跑就是十圈，重複跑十次操場。



例如：假設要撰寫程式產生**1000** 個「**Hello**」。

方法一： 寫1000 個「print('Hello')」。

產生 1000 個「Hello」的程式碼

```
print('Hello')
print('Hello')
print('Hello')
...
print('Hello')
```

1000 個 print('Hello')

方法二： 使用迴圈結構，簡化程式碼，達成相同功能。

產生 1000 個「Hello」的程式碼 (：ch5\5-for.py)

```
for i in range(1000):
    print('Hello')
```

電腦擁有強大的計算能力，每秒鐘可以執行幾億次的指令，程式中的迴圈結構可以重複執行某個程式區塊許多次，更能善加利用電腦的計算能力。

Question 1: 什麼是迴圈？

迴圈是讓一段程式碼，重複執行很多次的結構。

Question 2: 迴圈有幾種？

Python 有兩種常用的迴圈，分別是計次迴圈（for）與條件迴圈（while）。

(1)計次迴圈

在for迴圈中，變數依據計數變數初值到終值的變化，反覆執行區塊程式碼，當變數等於終值時離開迴圈。

～for 迴圈結構通常用於已知重複次數的程式。已知重複次數，不一定是直接控制迴圈的次數。

在撰寫程式時，程式設計師常常無法明確知道迴圈要執行的次數。

(2)條件迴圈：

在while迴圈中，測試條件是否成立，決定要不要繼續或跳出迴圈。

～while迴圈通常用於不固定次數的迴圈，只要條件符合就繼續做。

例如：猜數字遊戲

兩人（A 與B）玩猜數字遊戲，一人(A)心中想一個數，另一人(B)去猜。
A就B所猜數字回答「猜大一點」或「猜小一點」，直到B猜到A所想數字。

2. for迴圈

2-1 for迴圈與range()函數

range函數通常和for迴圈一起搭配使用。

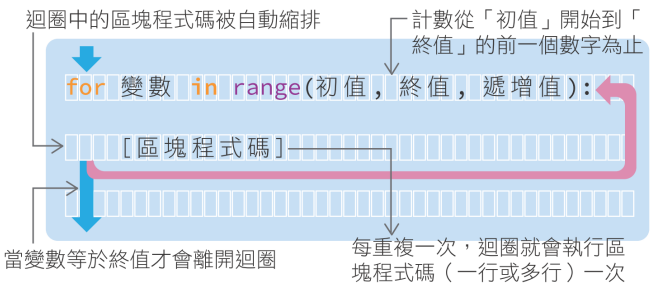


圖 4-3.4　for 迴圈語法說明

- 透過range函數，for迴圈可以指定迴圈變數的初值、終值與遞增(減) 值。
    - 迴圈變數由初值到終值的前一個數字為止，不包括終值。
    - 每重複執行一次，迴圈變數就依照遞增(減) 值遞增或遞減，並執行迴圈內程式。
- for迴圈指令的最後面要加「冒號：」，代表底下敘述區塊的開始。

- 在Python是用「上下對齊的縮排」方式，判斷是否在同一程式區塊。
    - 其他程式語言大多是利用大括號 { } 代表執行區塊。

範例：數字或文字的列印

例1: 數字列印，印出range()函數值的範圍

使用方法	範例	執行結果
range(終止值) range 函式指定「終止值」，數字串列會到「終止值」的前一個數字爲止，沒有指定起始值，預設起始值爲 0，沒有指定遞增值，預設爲遞增 1。	for i in range(5): print(i)	0 1 2 3 4
range(起始值 , 終止值) range 函式指定「起始值」與「終止值」，數字串列由「起始值」開始到「終止值」的前一個數字爲止，沒有指定遞增值，預設爲遞增 1。	for i in range(2,6): print(i)	2 3 4 5

使用方法	範例	執行結果
range(起始值 , 終止值 , 遞增 (減) 值) range 函式指定「起始值」、「終止值」與「遞增 (減) 值」，數字串列由「起始值」開始到「終止值」的前一個數字爲止，每次遞增或遞減「遞增 (減) 值」。	for i in range(2,10,2): print(i)	2 4 6 8
	for i in range(100,90,-3): print(i)	100 97 94 91

例2：文字列印，印出1000個Hello。

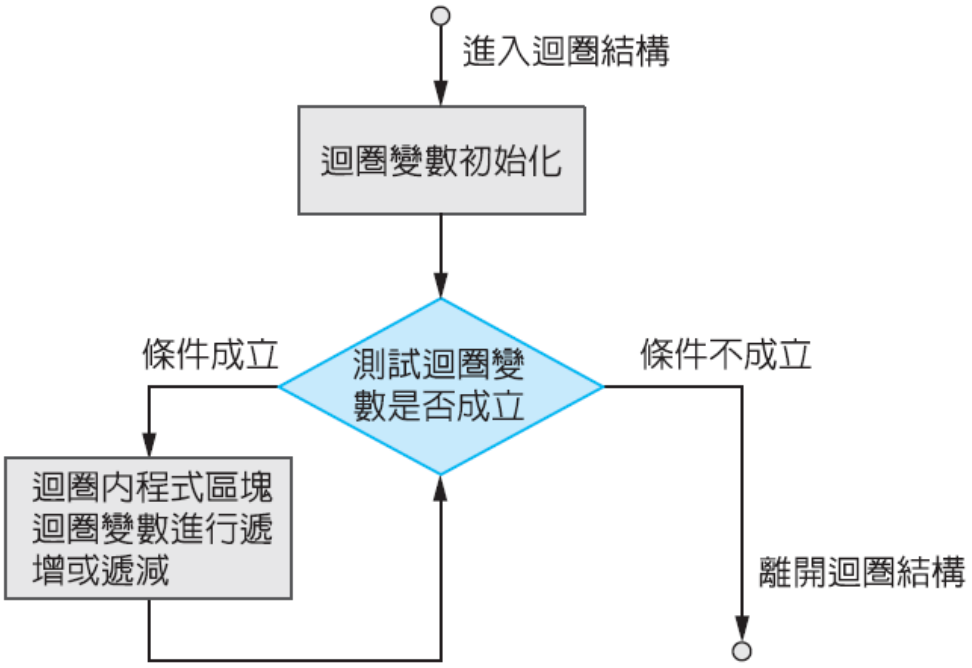


圖 A-11　流程圖

表 5-1　for 迴圈結構

for 程式語法	程式範例 (印出 1000 個 Hello)
for 迴圈變數 in range(起始值 , 終止值 , 遞增 (減) 值): 　重覆的程式	for i in range(0, 1000, 1): 　print('Hello')
說明	
for 迴圈內迴圈變數由起始值變化到終止值的前一個數字，每重複執行一次程式迴圈變數就會遞增 (減) 值，重複執行迴圈內程式。	

範例：數字加總

例1:寫一個程式計算數字1到10之和。

```
# 1. 數字1到10之和。
sum = 0  # 起始條件
for i in range(1, 11, 1): #數字串列[1,2,3,4,5,6,7,8,9,10]
    sum = sum + i   # 加總運算,變更條件
print("1+2+3+4+5+6+7+8+9+10=", sum)

# 2. 數字1到10奇數之和。
sum = 0 # 起始值
for i in range(1, 11, 2): # 數字串列[1,3,5,7,9]
    sum = sum + i # 加總運算,變更條件
print("1+3+5+7+9=", sum)
```

例2:寫一個程式計算數字3到13之間，每隔3個數字之和。

```
sum = 0
for i in range(3, 13, 3):
    sum = sum + i
```

i 值	sum 加總過程	sum 加總後
i=3	sum=0 + 3	sum=3
i=6	sum=3 + 6	sum=9
i=9	sum=9 + 9	sum=18
i=12	sum=18+12	sum=30

隨堂練習：寫一個程式允許使用者輸入加總的初值、終值與遞增值，計算數值加總的結果。

解題想法

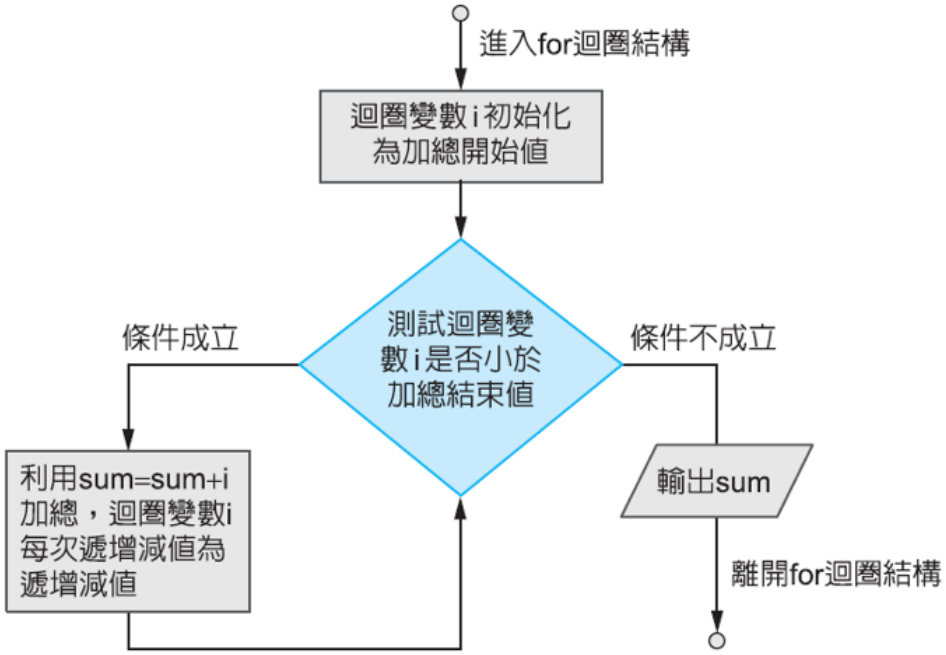


圖 A-13　流程圖

- 可以使用for 迴圈結構撰寫程式。
- 迴圈變數取決於range()函數。其起始值為輸入的加總起始值，迴圈變數終止值為輸入的加總終止值，迴圈每執行一次迴圈變數就會依照輸入的遞增(減) 值進行遞增(減)。
- 迴圈內使用「sum = sum + 迴圈變數」進行數值的加總，顯示加總的過程。
    - 在Python 語言中，等號右邊「sum + i」的算式會先計算，結果回存到等號左邊(sum)。
    - sum要先給初值。

```
# 給數字串列，將數字串列中的元素加總。
## 數字串列的初值、終值、遞增值
start = int(input('請輸入加總開始值？'))
end = int(input('請輸入加總終止值？'))
step = int(input('請輸入遞增減值？'))

## 用for迴圈做加總 + range()
sum = 0          #初始條件
for i in range(start, end, step):  #判斷條件
    sum = sum + i            #更新條件
    print('i為', i, '加總結果為', sum)
```

程式解說

- 第1 行：輸出「請輸入加總開始值？」，經由input 與int 函式，將輸入值轉換成整數，指定給變數start。
- 第2 行：輸出「請輸入加總終止值？」，經由input 與int 函式，將輸入值轉換成整數，指定給變數end。
- 第3 行：輸出「請輸入遞增減值？」，經由input 與int 函式，將輸入值轉換成整數，指定給變數inc。
- 第4 行：初始化變數sum 為0。
- 第5 到6 行：使用for 迴圈，其中i 值變化由使用者輸入的「加總開始值(s)」到「加總終止值(e)」的前一個數字，每次依所輸入的「遞增減值(inc)」進行遞增減，利用「sum=sum+i」計算加總（第6 行），將i 值與sum 值顯示於螢幕（第7 行）。

2-2 for 迴圈與資料容器

若要取出資料容器(字串、tuple、串列、字典) 的所有元素，可以使用「for」一個一個取出該資料容器中每一個元素，再對每一個元素進行運算。

```
for 暫時名字 in 資料容器：
    操作暫時名字 （逐一取出、逐一處理）
```

for...in保證兩件事：

(1)每次從資料容器（串列）裡只拿一個東西

(2)每個東西都會依序拿一次。

暫時名字 = 變數名稱

(1)如果不給這個暫時的名字，電腦會不知道如何操作這個拿出來的東西。

(2)暫時名字 = 這次拿出來的東西。

使用for讀取串列中的每一個元素，for迴圈把串列中的東西一個個拿出來。

～ 兩種 for迴圈的應用：列印與加總

```
# for迴圈示範碼
```

```
for i in range(5): # for 迴圈 + range()函數
for i in [0,1,2,3,4]: # for 迴圈  + 資料容器（數字串列）

# 典型程式：列印
for i in range(5):
    print("逐一取得清單中的元素列印", i)
for i in [0,1,2,3,4]:  # 串列
    print("逐一取得清單中的元素列印", i)

# for迴圈應用：加總
## 1 + 2 + 3 + ... + 10 = ?
sum = 0  #初始條件
for i in range(1,11) # 判斷條件 [1,2,3,4,5,6,7,8,9,10]
    sum = sum + i  #運算式，變更條件
print("Total is" , sum)
```

範例：讓迴圈跑過串列

～變數i輪流等於串列清單中的每個項目，執行程式碼時是i = alex，接著i = bob，以此類推。

```
# 串列資料容器
names = ['alex', 'bob', 'sue', 'dave', 'emily']
## 計數迴圈：把names串列中的東西一個一個直接拿出來
for i in names:   #每次拿出來的東西暫時叫它i
    print('Welcome to Class!', i)  #清單東西都拿完就結束

## 用range處理串列
sheet = ['牛奶', '蛋', '咖啡豆']
for i in range(0,len(sheet)): #位置索引從0開始 [0,1,2]
    print (i, sheet[i])  #i是索引值，sheet[i]才是項目

# 數字串列資料容器
numbers = [123, 456, 789]
for num in numbers:
    print(num)  #印出拿的東西
print(numbers[0], numbers[1], numbers[2]) #與上述結果比較
```

2-3 巢狀迴圈

巢狀迴圈是多層迴圈，為迴圈內又有迴圈的程式結構。
從外層迴圈來看，內層迴圈只是外層迴圈內的動作，因此外層迴圈作用一次，內層迴圈全部都需要執行一次。

範例： 九九乘法表

寫一個程式印出九九乘法表。

解題想法

以輸出九九乘法表為例，當外層迴圈作用一次，內層迴圈要執行九次，當外層迴圈作用九次，內層迴圈總共執行八十一次。

~巢狀迴圈的外層迴圈使用迴圈變數 i，內層迴圈使用迴圈變數 j。

- 外層迴圈 i 等於1，內層迴圈 j 由1 變化到9，印出「1 * 1 = 1，1 * 2 = 2，1 * 3 = 3，…，1 * 9 = 9」。
- 外層迴圈 i 遞增1，外層迴圈 i 等於2，內層迴圈 j再次由1 變化到9，印出「2 * 1 = 2，2 * 2 = 4，2 * 3 = 6，…，2 * 9 = 18」。
- 依此類推，直到外層迴圈 i 等於9，內層迴圈 j 由1 變化到9，印出「9 * 1 = 9，9 * 2 = 18，9 * 3 = 27，…，9 * 9 = 81」。

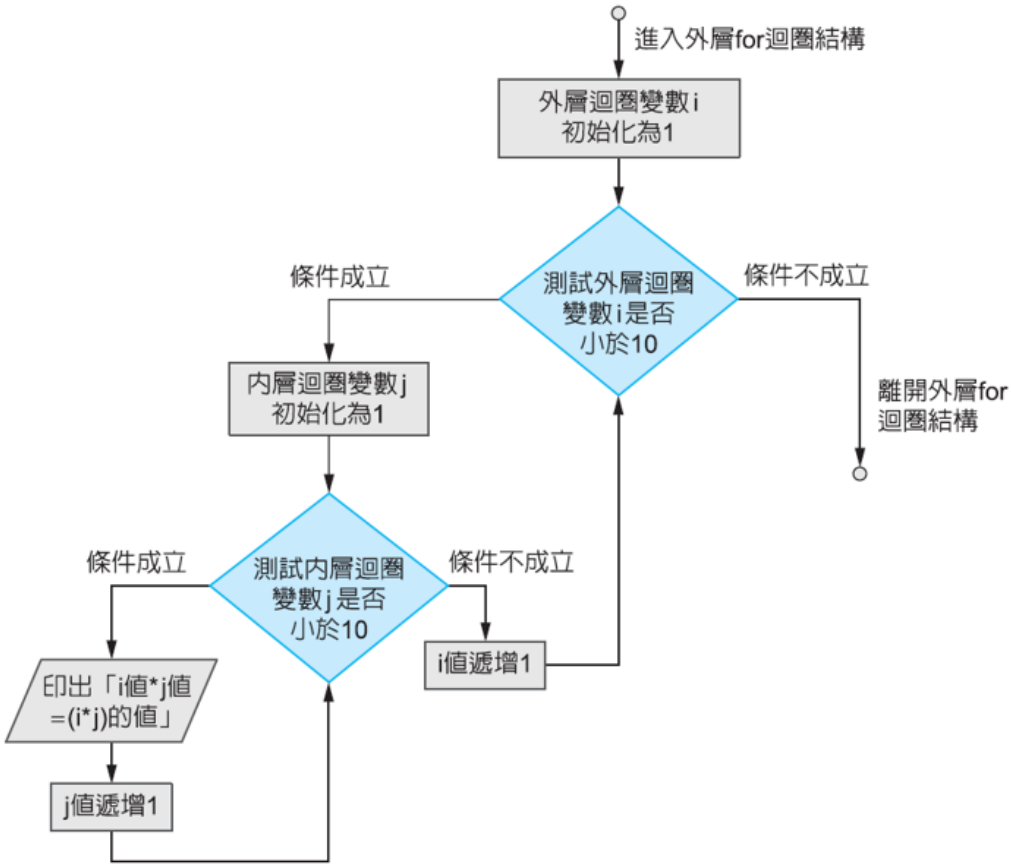


圖 A-18　流程圖

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

i 值	j 值	輸出結果
i=1	j=1,2,3,4,5,6,7,8,9	1*1=1 1*2=2 1*3=3 1*4=4 1*5=5 1*6=6 1*7=7 1*8=8 1*9=9
i=2	j=1,2,3,4,5,6,7,8,9	2*1=2 2*2=4 2*3=6 2*4=8 2*5=10 2*6=12 2*7=14 2*8=16 2*9=18
i=3	j=1,2,3,4,5,6,7,8,9	3*1=3 3*2=6 3*3=9 3*4=12 3*5=15 3*6=18 3*7=21 3*8=24 3*9=27
…	…	…
i=9	j=1,2,3,4,5,6,7,8,9	9*1=9 9*2=18 9*3=27 9*4=36 9*5=45 9*6=54 9*7=63 9*8=72 9*9=81

```python
# 雙迴圈，for巢狀迴圈
for i in range(1,10):  # 控制輸出幾行
    for j in range(1,10): # 每行輸出多少個
        # 依序將「變數i* 變數j = 變數i 與j 相乘結果」顯示在螢幕
        # 串接tab，設定sep與end為空字串。
        print(i, '*', j, '=', i*j, '\t', sep='',end='')
    print() # 每行輸出後換行
```