



作者序

雲端運算、網路爬蟲、大數據、物聯網等技術都和 Python 習習相關，進而讓 Python 成為 AI 與大數據時代最熱門的程式語言。同時 Python 擁有簡潔易學、免費開源、高階程式語言、直譯式與可移植性、膠水語言與可嵌入性、腳本語言、物件導向程式設計、功能豐富的函式庫等優點。不論是小型或大型的程式，相較於 Java 或 C/C++，Python 能讓開發人員可以使用更少的程式碼完成撰寫，且該語言簡短結構清晰明瞭。

有鑑於市面 Python 書籍眾多，入門書籍簡單無實例、專題書籍類型太多太雜，且缺少循序漸進方式打下程式設計基礎，不利初學者上手與教師教學。因此本書由學校教師（僑光科技大學－蔡文龍、曾芷琳）、資策會補教名師（蔡捷雲）與微軟最有價值專家（歐志信）共同編著，撰寫書籍的同時進行試教，精選出適合的章節與技能，由 Python 基礎程式設計開始、經由流程控制、迴圈、串列、函式、字典、檔案操作等章節，逐漸邁向資料爬蟲技能：存取開放資料 JSON 與 CSV、爬蟲網頁資訊進行彙整，最後提供五個實務案例讓初學者練習套用。書中範例圖文並茂，且使用淺顯易懂的語法與豐富的實際範例，是一本自學與教授 Python 程式設計與爬蟲應用的好書。本書備有提供教師使用的教學投影片與習題解答，採用本書的授課教師可向基峰業務索取，以供教學使用。

編著	蔡文龍	歐志信	蔡捷雲	微軟最有價值專家 (MVP)
	曾芷琳			僑光科技大學多遊系副教授
	黃承威	卓宏逸		大才全資訊科技資訊工程師

中華民國 109 年 5 月

06

重複結構

選擇結構是依條件指定要執行特定區塊內的敘述，而重覆結構是指重覆執行某一區塊內的敘述，善用這兩種結構有助於訓練初學者的邏輯。本章將介紹 Python 所提供的重複結構，開發者只要能善用選擇與重複結構即能靈活的進行程式的流程控制。





6.1 for 迴圈敘述

在程式執行過程中，遇到需要多次重複執行特定的區塊敘述時，則需使用「重複結構」敘述來達成此目的。在 Python 中所提供的重複結構敘述中，大致分為兩種敘述：在重複執行次數確定的情況下，可透過 for 迴圈敘述來完成；在重複執行的次數無法確定的情況下，需透過當下的條件式判斷來決定時，則須透過 while 敘述來達成。

6.1.1 range 函式

在 Python 提供的 range() 內建函式可產生一個數字串列，其語法如下：

寫法 1：

```
range( 終值 )
```

寫法 2：

```
range( 初值 , 終值 [, 間隔值 ] )
```

1. 初值：

為串列的初始值，可省略。若省略設定初值則預設串列初值為 0。

2. 終值：

為串列的終止值，不可省略。

3. 間隔值：

為串列的間隔值，可以是正值、負值或省略不寫。若省略則預設間隔值為 1；若間隔值為負值，則串列即是遞減序列。

例 建立 0 ~ 4 串列，寫法如下：

```
range(5)
```

例 建立 5 ~ 15 串列，寫法如下：

```
range(5, 16)
```

例 建立 15、12、9、6、3 串列，寫法如下：

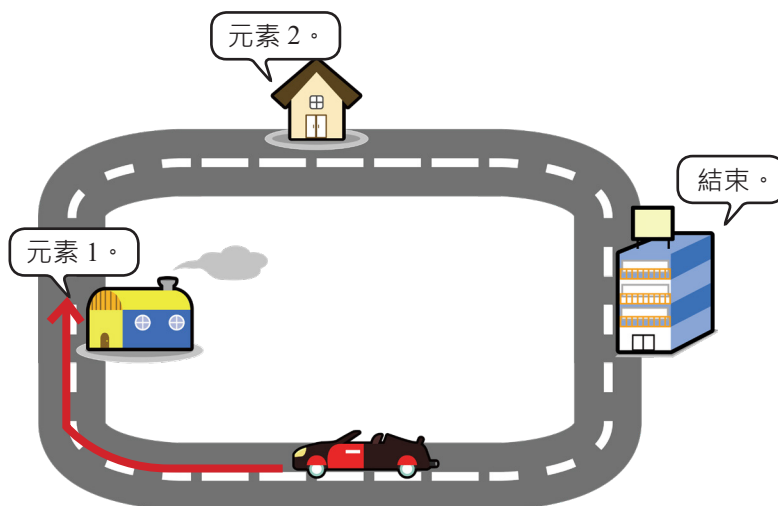
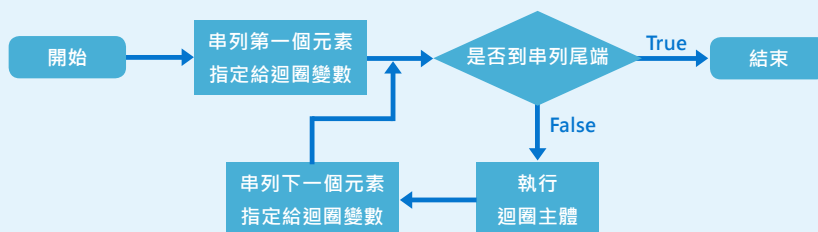
```
range(15, 0, -3)
```

6.1.2 for 敘述

在撰寫程式過程中，如果已經確定要重複執行的次數時，則可透過 for 敘述來達成。Python 的 for 迴圈敘述會逐一取出串列元素並指定給迴圈變數，接著再進入迴圈主體執行，一直到串列沒有元素即停止。要注意的是 for 迴圈敘述要使用「:」符號當結尾，迴圈主體要進行縮排。語法如下：

for 迴圈變數 in 串列：

迴圈主體





範例演練 (for01.py)

練習 for 迴圈敘述的使用，完整程式碼請參考 for01.py。

1. 使用 for 迴圈敘述印出「1,2,3,4,5,」，寫法如下：

```
for i in range(1,6):  
    print(i, end=",")
```

2. 使用 for 迴圈敘述印出「5,4,3,2,1,」，寫法如下：

```
for i in range(5,0,-1):  
    print(i, end=",")
```

3. 使用 for 迴圈敘述印出「1+3+5+...<100=2500」的結果，寫法如下：

```
sum=0  
for i in range(1,100,2):  
    sum += i  
print("1+3+5+...<100=%d" %sum)
```

4. 使用 for 迴圈敘述印出 program 串列中所有元素的資料，寫法如下：

```
program = ["Python", "Java", "C#", "C++"]  
for s in program:  
    print(s, end=",")
```

執行結果

```
IPython console  
Console 1/A  
In [18]: runfile('C:/PythonEx/ch06/  
for01.py', wdir='C:/PythonEx/ch06')  
1,2,3,4,5,  
5,4,3,2,1,  
1+3+5+...<100=2500  
Python,Java,C#,C++,
```

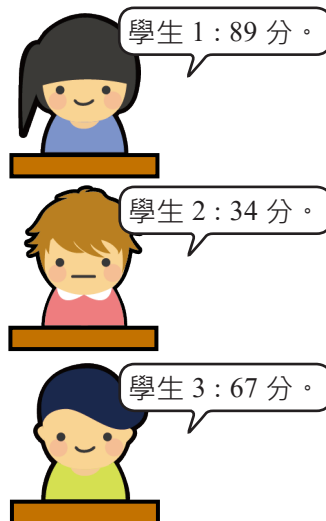


實例挑戰 (for03.py)


先輸入學生數，接著連續輸入每位學生的成績，最後使用串列的 `sort()` 和 `reverse()` 方法來印出成績遞增排序與遞減排序的結果。

執行結果

```
請輸入學生數：3   
第 1 位學生：  
89   
第 2 位學生：  
34   
第 3 位學生：  
67   
成績列表：[89, 34, 67]  
遞增排序：[34, 67, 89]  
遞減排序：[89, 67, 34]
```



完整程式碼

```
01 listScore = [] # 建立 listScore 為空串列  
02 count = int(input(" 請輸入學生數：")) # 指定學生數  
03 # 輸入學生成績並逐一放入 listScore 串列，append() 方法可將資料附加到串列中  
04 for i in range(count):  
05     print(" 第 %d 位學生： " %(i+1), end="")  
06     listScore.append(int(input("")))   
07  
08 print(" 成績列表：", listScore) # 顯示 listScore 所有元素  
09 listScore.sort() # 由小到大排序 listScore 串列  
10 print(" 遞增排序：", listScore) # 印出 listScore 由小到大排序的結果  
11 listScore.reverse() # listScore 串列反轉  
12 print(" 遞減排序：", listScore) # 印出 listScore 由大到小排序的結果
```

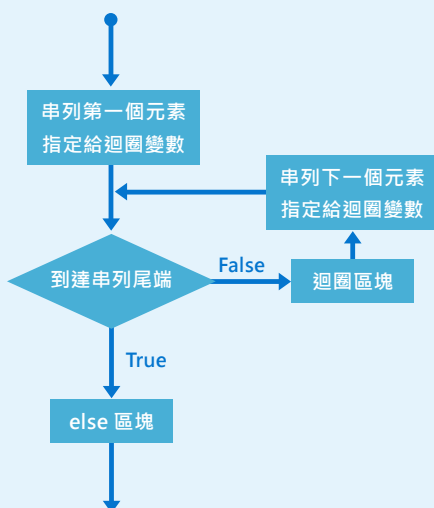
程式說明

04~06 行：使用 for 迴圈逐一將每位學生的成績放入 listScore 串列中。

6.1.3 for...else 敘述

Python 的 for 迴圈敘述另外可以再指定 else 敘述區塊。當 for 正常結束後，若有指定 else，此時即會執行 else 敘述區塊一次，else 敘述區塊可進行迴圈最後的處理或配合 break 敘述使用。語法如下。

```
for 迴圈變數 in 串列:
    # 迴圈區塊
else:
    # else 敘述區塊
```



實例挑戰 (forelse01.py)


練習撰寫可以驗證帳密三次的程式。當使用者輸入帳號等於 "dte" 且 密碼 "168" 時即顯示 "帳密正確，歡迎進入系統"；當輸入三次帳密都失敗時即顯示 "3 次帳密錯誤，無法使用系統" 訊息。






執行結果

第 1 次帳密驗證：

帳號：dtc 

密碼：168 


帳密正確，歡迎進入系統


▲ 帳密正確執行結果

3 次帳密錯誤，無法使用系統。





第 1 次帳密驗證：

帳號：jasper 


密碼：520 


第 2 次帳密驗證：

帳號：gotop 

密碼：168 

第 3 次帳密驗證：

帳號：dtc 

密碼：520 

3 次帳密錯誤，無法使用系統

▲ 三次帳密錯誤執行結果

完整程式碼

```
01 # 帳號密碼驗證三次
02 for i in range(3):
03     print(" 第 %d 次帳密驗證：" % (i+1), end="")
04     uid = input(" 帳號：")          # 將帳號指定給 uid
05     pw = input(" 密碼：")          # 將密碼指定給 pw
06     if uid=="dtc" and pw=="168":  # 判斷帳密是否為 "dtc" 與 "168"
07         print(" 帳密正確，歡迎進入系統 ")
08         break
09 else :
10     print()
11     print("3 次帳密錯誤，無法使用系統 ")
```


程式說明

02~11 行：for 迴圈執行 3 次。

03~05 行：讓使用者輸入 3 次帳號與密碼。

06~08 行：判斷帳號是否為 "drc" 與 "168"，若成立則印出 "帳號正確，歡迎進入系統" 訊息，接著執行 break 敘述離開迴圈。

09~11 行：當 for 迴圈 3 次執行完成會執行 else 敘述區塊。

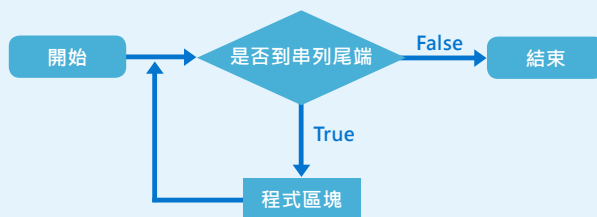
6.2 while 迴圈敘述

6.2.1 while 敘述

while 之後會接上條件式，當條件式為 True 時即進入迴圈內執行程式區塊，執行後再回到條件式判斷是否繼續執行迴圈內的程式區塊，一直到條件式為 False 時才離開 while 迴圈。所以迴圈內的程式區塊必須有設定條件式為 False 的程式敘述，否則程式會無法離開迴圈而形成無窮迴圈。由於 while 迴圈是先判斷條件式再決定是否執行迴圈內的程式區塊，因此該迴圈有可能一次都不會執行。while 敘述語法與流程圖如下：

while(條件式)：

程式區塊迴圈主體





實例挑戰 (while01.py)

將 for02.py 印出三筆產品的範例，改使用 while 來達成，本例執行結果與 for02.py 相同。



完整程式碼

```
01 #name 串列存放產品名稱
02 name = ["PS4 Slim 主機 CUH-2117",
03         "任天堂 Nintendo Switch", "Xbox One S 500G 同捆組 "]
04 #price 串列存放產品單價
05 price = [9980, 12999, 11000]
06 #len() 函式取得 name 串列個數並指定給 count
07 count = len(name)
08 #count 等於 3，range(count) 會產生 0, 1, 2 串列，for 迴圈中的 i 會逐一被指定為 0, 1, 2
09 i=0                                     # i 起始為 0
10 while (i<count):
11     print("%s \t" %name[i],end="")      # 印出產品
12     print(" 單價 %d 元 " %price[i])    # 印出單價
13     i+=1
```

程式說明

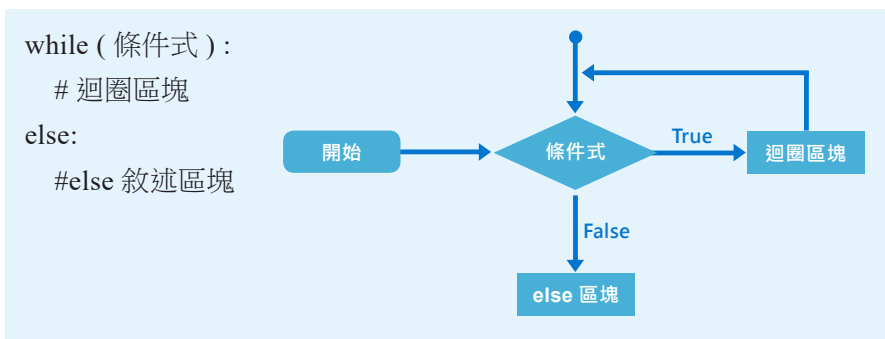
10~13 行：使用 for 迴圈將 name 產品名稱串列和 price 單價串列的所有內容印出，即印出 name[0]~name[2] 和 price[0]~price[2] 的串列內容。

13 行：將 i 加 1，若省略此行，則 i 會有不符合 i<3 的情形，省略此行即會永遠離不開迴圈。



6.2.2 while...else 敘述

當 while 正常結束後，若有指定 else，此時即會執行 else 敘述區塊一次，else 敘述區塊可進行迴圈最後處理使用。語法如下。



實例挑戰 (whileelse01.py)

練習將 forelse01.py 的帳號密碼驗證程式改使用 while...else... 敘述。本例執行結果與 forelse01.py 相同。

完整程式碼

```
01 i = 1 # i 起始值為 1  
02 # 執行三次  
03 while (i<=3) :  
04     print(" 第 %d 次帳密驗證：" %i, end="")  
05     uid = input(" 帳號：" ) # 將帳號指定給 uid  
06     pw = input(" 密碼：" ) # 將密碼指定給 pw  
07     if uid=="dtc" and pw=="168":  
08         print(" 帳密正確，歡迎進入系統 ")  
09         break  
10     i+=1  
11 else :  
12     print()  
13     print("3 次帳密錯誤，無法使用系統 ")
```

程式說明

- 03 行： `i` 小於等於 3 會執行 04~10 行。
- 04~06 行：讓使用者輸入 3 次帳號與密碼。
- 07~09 行：判斷帳號是否為 "dte" 與 "168"，若成立則印出 "帳號密碼正確，歡迎進入系統" 訊息，接著執行 `break` 敘述離開迴圈。
- 10 行：將 `i` 加 1，若省略此行，則 `i` 會有不符合 `i<3` 的情形，省略此行即會永遠離不開迴圈。
- 11~13 行：當 `while` 迴圈 3 次執行完成會執行 `else` 敘述區塊。

6.3 break 與 continue 敘述

6.3.1 break 敘述

在迴圈區塊中若碰遇到 `break` 敘述時會忽略 `break` 後面的程式敘述而直接跳離迴圈，繼續往下執行。`break` 敘述的流程和語法如下，如下流程可知迴圈區塊 B 皆不會執行到。

for 變數 in 串列：

迴圈區塊 A

`break`

迴圈區塊 B

迴圈外敘述

while(條件式):

迴圈區塊 A

`break`

迴圈區塊 B

迴圈外敘述



實例挑戰 (break01.py)

練習撰寫依產品編號進行查詢產品的程式。

建立 pid、name、price 三個串列用來記錄三筆產品的產品編號、品名、單價。如下：



A01 |

第一筆記錄 →

第二筆記錄 →

第三筆記錄 →

pid	name	price
"A01"	"PS4 特價包 "	9980
"A02"	" Switch "	12999
"A03"	"Xbox One"	11000

執行結果

Switch
單價:12999 元。

A02 |

找不到資料。

B02 |

請輸入欲查詢的產品編號：A02

編號 品名 單價

A02 Switch 12999

▲ 依產品編號找到產品資訊

請輸入欲查詢的產品編號：B02

找不到資料

▲ 沒有查詢到產品資訊

完整程式碼

```
01 #pid 串列存放產品編號
02 pid = ["A01", "A02", "A03"]
03 #name 串列存放產品名稱
04 name = ["PS4 特價包 ", "Switch", "Xbox One"]
05 #price 串列存放產品單價
06 price = [9980, 12999, 11000]
07
08 inputId = input(" 請輸入欲查詢的產品編號：")
09
10 index=-1                # index 串列索引為 -1 表示找不到
11 count = len(pid)         # len() 函式取得 name 串列個數並指定給 count
12 #count 等於 3，因此 range(count) 會產生 [0, 1, 2] 串列
13 #for 迴圈中的 i 會逐一被指定為 0, 1, 2
14 for i in range(count):
15     if(inputId==pid[i]):
16         index=i          # 若有找到資料將 i 指定給 index
17         break            # 離開迴圈
18 # 若 index 等於 -1 表示找不到資料
19 if index== -1:
20     print(" 找不到資料 ")
21 else:
22     print(" 編號\t品名\t單價 ")
23     print("%s\t%s\t%d" %(pid[index], name[index], price[index]))
```

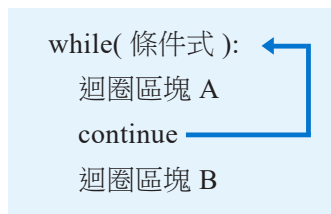
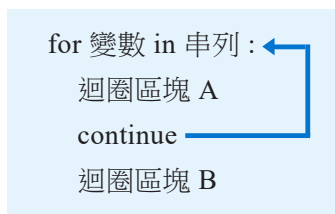


程式說明

- 08 行： 輸入欲查詢的產品編號並指定給變數 `inputId`。
- 10 行： 另 `index` 等於 `-1`，`-1` 表示找不到產品編號的索引。
- 14~17 行： `inputId` 逐一比對是否等於 `pid[0]~pid[2]` 的串列元素，若有找到符合的產品編號即將該索引指定給 `index`，接著再執行 `break` 離開迴圈。
- 19~23 行： 若 `index` 等於 `-1` 表示找不到產品編號，此時執行 20 行；否則執行 21~23 行將產品編號、品名、單價印出。

6.3.2 continue 敘述

在某些情況下，迴圈區塊中要忽略後面的敘述，需要跳回迴圈開頭繼續執行，此時就要使用 `continue` 敘述。`continue` 敘述的流程和語法如下：

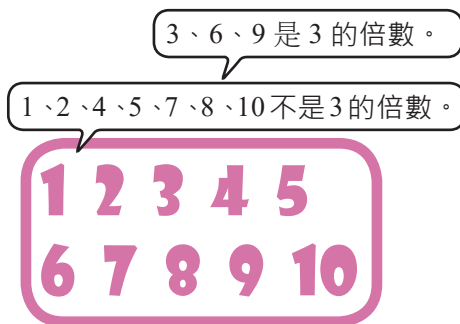


實例挑戰 (continue01.py)

練習使用 `continue` 敘述配合 `while` 顯示 1~10 數字中不是 3 的倍數的整數。

執行結果

1 不是 3 的倍數
2 不是 3 的倍數
4 不是 3 的倍數
5 不是 3 的倍數
7 不是 3 的倍數
8 不是 3 的倍數
10 不是 3 的倍數



完整程式碼

```
01 i = 0
02 while(i<10):
03     i += 1
04     if(i % 3 == 0):
05         continue
06     print("%d 不是 3 的倍數" % (i))
```

程式說明

02 行：當 i 小於 10 即執行 03~06 行程式。

03 行：進入迴圈後 i 加上 1。

04 行： i 除於 3 餘數為 0 表示為 3 的倍數，此即執行 05 行。

05 行：執行到 `continue` 馬上回到 `while` 迴圈開頭。

06 行：印出目前 i 的值，印出的 i 值不是 3 的倍數。

6.4 巢狀迴圈

若迴圈內還有另一層迴圈，即由內而外形成「巢狀迴圈」(Nested loop)，`for` 與 `while` 迴圈都可以同時使用形成巢狀迴圈。開發人員要特別注意的是：撰寫巢狀迴圈程式時，迴圈縮排程式要正確才可正常執行。



範例演練 (nestLoop01.py)

使用巢狀迴圈列印九九乘法表。程式寫法如下：

```
01 for i in range(1,10):                # 外層迴圈
02     for j in range(1, 10):            # 內層迴圈
03         print("%d*%d=%2d" % (i, j, (i*j)), end='; ')
04     print() # 換行列印
```




程式說明

01 行：外層迴圈，i 外層迴圈控制變數範圍由 1~9。

02 行：外層迴圈，j 內層迴圈控制變數範圍由 1~9。

03 行：列印 $i*j=(i*j)$ 的格式與值。

執行結果

```
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;
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;
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;
4*1= 4; 4*2= 8; 4*3=12; 4*4=16; 4*5=20; 4*6=24; 4*7=28; 4*8=32; 4*9=36;
5*1= 5; 5*2=10; 5*3=15; 5*4=20; 5*5=25; 5*6=30; 5*7=35; 5*8=40; 5*9=45;
6*1= 6; 6*2=12; 6*3=18; 6*4=24; 6*5=30; 6*6=36; 6*7=42; 6*8=48; 6*9=54;
7*1= 7; 7*2=14; 7*3=21; 7*4=28; 7*5=35; 7*6=42; 7*7=49; 7*8=56; 7*9=63;
8*1= 8; 8*2=16; 8*3=24; 8*4=32; 8*5=40; 8*6=48; 8*7=56; 8*8=64; 8*9=72;
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;
```



實例挑戰 (nestLoop02.py)

使用一維串列記錄四位學生的姓名，使用二維串列記錄四位學生的國文、英文、數學的成績，最後再計算四位學生三科成績的總分。

執行結果

姓名	國文	英文	數學	總分
小明	77	66	88	231
小華	83	92	56	231
小莉	90	98	79	267
小呆	89	81	70	240





6.5

習題

1. 下列何者正確？
 - (1) 在重複執行次數確定的情況下，可透過 `while` 迴圈敘述來完成。
 - (2) 在重複執行的次數無法確定的情況下，需透過當下的條件式判斷來決定時，則須透過 `for` 敘述來達成。
 - (3) 在 Python 中所提供的重複結構敘述為 `loop` 敘述。
 - (4) 在程式執行過程中，遇到需要多次重複執行特定的區塊敘述時，則需使用重複結構敘述來達成此目的。
2. 欲建立 4~16 數字串列，程式碼為下列何者？
 - (1) `range(3, 15)`
 - (2) `range(4, 15)`
 - (3) `range(4, 16)`
 - (4) `range(4, 17)`
3. 欲建立 14、12、10、8 數字串列，程式碼為下列何者？
 - (1) `range(14,8,-3)`
 - (2) `range(14,8,-2)`
 - (3) `range(13,7,-2)`
 - (4) `range(14,7,-2)`
4. 下列程式碼印出結果為何？

```
for i in range(5,0,-1):  
    print(i, end=",")
```

 - (1) 5,4,3,2,1,0,
 - (2) 5,4,3,2,1
 - (3) 5,4,3,2,1,
 - (4) 6,5,4,3,2,



9. 下列何者有誤？

- (1) 在某些情況下，迴圈區塊中要忽略後面的敘述，需要跳回迴圈開頭繼續執行，此時就要使用 `continue` 敘述。
- (2) 若迴圈內還有另一層迴圈，即由內而外形成巢狀迴圈。
- (3) 只有 `for` 迴圈可以形成巢狀迴圈。
- (4) 在迴圈區塊中若碰遇到 `break` 敘述時會忽略 `break` 後面的程式敘述而直接跳離迴圈，繼續往下執行。

10. 在迴圈區塊中如果要跳離迴圈，可使用下列何種敘述？

- (1)`break`
- (2)`continue`
- (3)`else`
- (4)`range`

這章節結束囉，非常棒 ~!!
繼續加油 !!!

