
Python 迴圈 Loop (I)

臺北科技大學資訊工程系

for

```
Statement 1  
for var in sequence object :  
    Body statement  
Statement 2
```

□ for in

- 序列物件(Sequence object)，有順序可數的元素
- 控制變數var又稱迴圈變數/迴圈索引

for

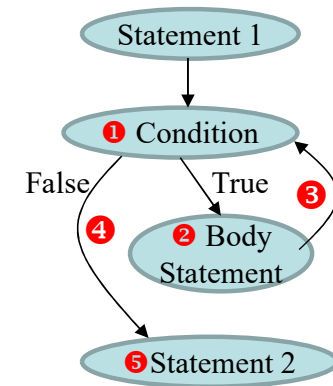
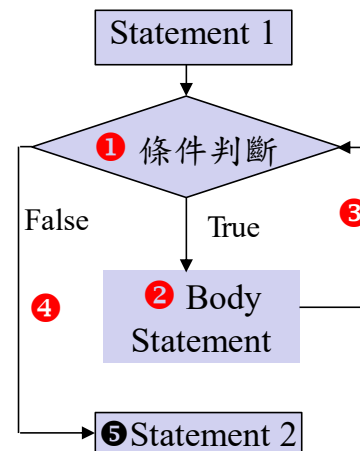
```
Statement 1  
for var in sequence object :  
    Body statement  
Statement 2
```

for in 迴圈執行流程

- ❶ 條件判斷
 - True (從序列物件中**找到**下一個元素)，取出給var ❷
 - False (從序列物件中**找不到**下一個元素)，❸跳出迴圈
- ❷ 執行 **loop 本體**所有指令(**Body Statement**)
- ❸ 回到 ❶，

依次取出**代入**的動作，稱為**疊代**

break 和 continue 可在 for 迴圈中出現



for - range()

□ range()函式回傳序列物件 range

使用方法	範例	執行結果
range(終止值) 數字串列到「終止值」的前一個數字為止，沒有指定起始值，預設起始值為0，沒有指定遞增值，預設為遞增1。	for i in range(5): print(i)	0 1 2 3 4
range(起始值, 終止值) 數字串列由「起始值」開始到「終止值」的前一個數字為止，沒有指定遞增值，預設為遞增1	for i in range(2, 6): print(i)	2 3 4 5
range(起始值, 終止值, 遞增(減)值) 數字串列由「起始值」開始到「終止值」的前一個數字為止，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

for - range()

❑ 序列物件 (Sequence)

○ Tuple, String, range(), List

❑ range(3) , 產生 0, 1, 2

```
def iterOp():  
    myRange = range(8)  
    print(myRange)  
    print(type(myRange))  
    print(myRange[2])  
    myRange = range(3,8,2)  
    print(myRange)  
    print(myRange[2])
```

```
range(0, 8)  
<class 'range'>  
2  
  
range(3, 8, 2)  
7
```

for - range()

□ range(0, 3, 1)，產生 0, 1, 2

○ ❶ 條件判斷，True

➢ 找到 0, 1, 2 的第1個，指定給 i，i = 0

○ ❷ 印出 i， ❸ 回到 ❶

○ ❶ 條件判斷，True

➢ 找到 0, 1, 2 的第2個，指定給 i，i = 1

○ ❷ 印出 i， ❸ 回到 ❶

○ ❶ 條件判斷，True

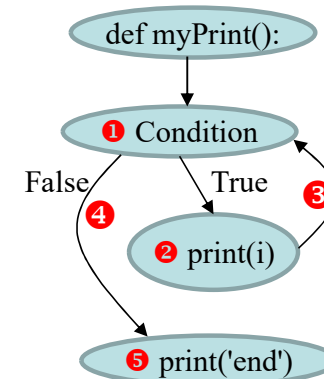
➢ 找到 0, 1, 2 的第3個，指定給 i，i = 2

○ ❷ 印出 i， ❸ 回到 ❶

○ ❶ 條件判斷，False ❹

➢ 序列物件找不到下一個資料，跳出迴圈 ❺

```
01. def myPrint():  
02.     for i in range(0, 3, 1):  
03.         print(i)  
04.     print('end')
```

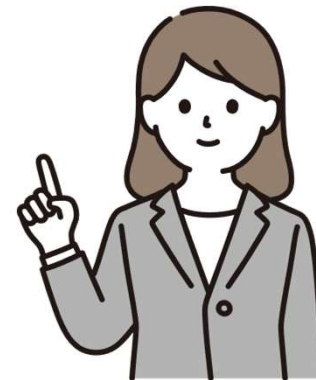
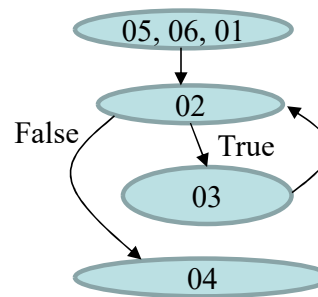


for - range()

□ 使用者輸入 n，產生 0, 1, 2, ..., n

- 程式
- 程式編號
- 程式編號執行順序

```
01 def myPrint(num):  
02     for i in range(0, num+1):  
03         print(i)  
04         print('end')  
05 n = int(input('number:'))  
06 myPrint(n)
```



考試使用此種畫法

for - range()

□ range(1, 4)，產生 1, 2, 3

○ ❶ 條件判斷，True

➤ 找到 1, 2, 3 的第1個，指定給 i，i=1

○ ❷ 印出 i， $s = s + i = 0 + 1 = 1$ ，❸ 回到 ❶

○ ❶ 條件判斷，True

➤ 找到 1, 2, 3 的第2個，指定給 i，i=2

○ ❷ 印出 i， $s = s + i = 1 + 2 = 3$ ，❸ 回到 ❶

○ ❶ 條件判斷，True

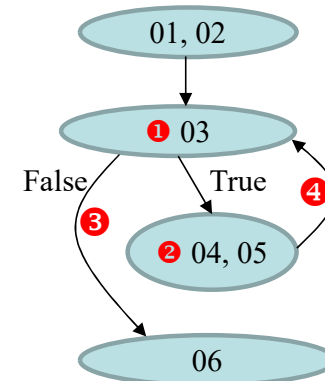
➤ 找到 1, 2, 3 的第3個，指定給 i，i=3

○ ❷ 印出 i， $s = s + i = 3 + 3 = 6$ ，❸ 回到 ❶

○ ❶ 條件判斷，False ❹

➤ 序列物件找不到下一個資料，跳出迴圈 ❺

```
01 def myPrint():
02     s = 0
03     for i in range(1, 4):
04         print(i)
05         s = s + i
06     print('sum=', s)
```



for 迴圈

- ❑ 輸出 $1 + 2 + 3 + 4 + 5 + \dots + 10$
- ❑ 輸出 $1 * 2 * 3 * 4 * 5 * \dots * 10$

編號每一行程式
劃出流程圖
寫下執行編號順序
寫下輸出內容

```
def getSum(num):  
    sumValue = 0  
    for i in range(num):  
        sumValue = sumValue + (i+1)  
    return sumValue  
  
def main():  
    num = int(input("Input a number:"))  
    myPrint(num)  
    getSum(num)  
    getProduct(num)  
  
main()
```

```
def myPrint(n):  
    for i in range(0, n, 1):  
        print(i)
```

```
def getProduct(num):  
    product = 1  
    for i in range(1, num):  
        product = product * i  
    return product
```

Exercise

- 輸入開始值m、結束值n與遞增值step，計算數值加總結果
 - 例如 $3 + 6 + 9 + 12$ ，輸入3為開始值，12為結束值，3為遞增值。

```
def mySum(m, n, step):
```

```
    return sum
```

```
def main():  
    m = int(input("Input a min number:"))  
    n = int(input("Input a max number:"))  
    step = int(input("Input a step number:"))  
    main_sum = mySum(m, n, step)  
    print('sum (%d ~ %d)= %d' %(m, n, main_sum))  
  
main()
```

編號每一行程式
劃出流程圖
寫下執行編號順序
寫下輸出內容

Exercise

□ 輸入 m, n ，計算 $m \sim n$ 的偶數，相加總合、相乘總和？

□ 輸出兩數

○ $m + (m+2) + (m+4) + (m+6) + \dots + n = ?$

○ $m * (m+3) * (m+6) * (m+9) * \dots * n = ?$

```
def forOps(m, n):  
    for index in range(_____):
```

```
        print(index)
```

編號每一行程式
劃出流程圖
寫下執行編號順序
寫下輸出內容

Exercise

□ 改寫以下程式

- 印出myString中大寫字母
- 計算myString有幾個字元?
- 計算myString有幾個大寫字元?

```
01 def forOps():  
02     myString = "ATCgATAgcTCGaTCG"  
03     for index in myString:  
04         if index.isupper():  
05             print(index)
```

編號每一行程式
劃出流程圖
寫下執行編號順序
寫下輸出內容

for 迴圈

□ in 在串列中，一個一個依序取出

```
def forOps():  
    myList = ["asm", "C", "C++", "Java", "iOS", "Ruby", "perl", "delphi", "python"]  
    for index in myList:  
        print(index)
```

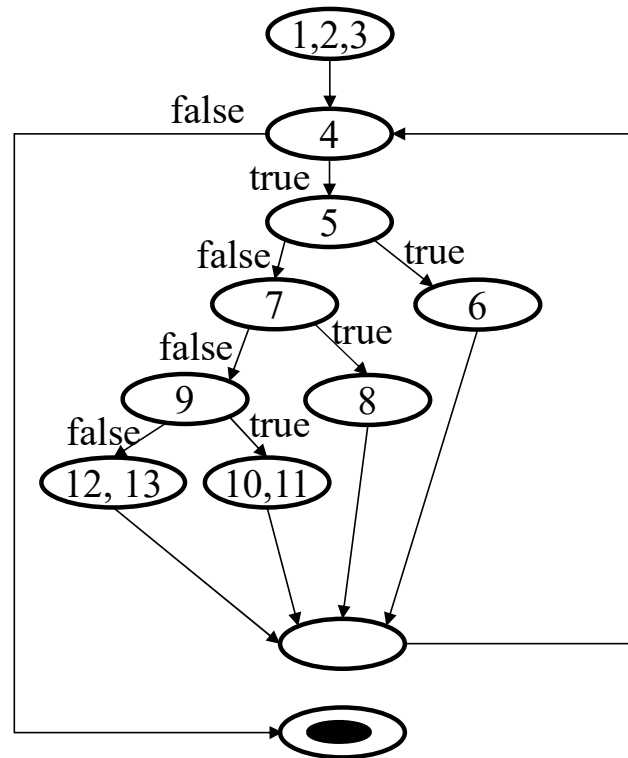
```
01 def forOps():  
02     i = 1  
03     myList = ["asm", "C", "python", "C++", "Java", "iOS", "Ruby", "perl", "delphi"]  
04     for index in myList :  
05         if (index == "python"):  
06             print(i,index)  
07         elif (index == "Java"):  
08             print(i, index)  
09         elif (i%3 != 0):  
10             print(i, index)  
11             i = i+1  
12         else:  
13             i = i+1  
14     forOps()
```

編號每一行程式
劃出流程圖
寫下執行編號順序
寫下輸出內容

for 迴圈

	i	index	i % 3	output
14, 1, 2, 3, 4	1	asm	1	1 asm
5, 7, 9, 10, 11, 4	2	C	2	
5, 7, 9, 10	3	python	0	2 C
11, 4				3 python
5, 6	4	C++	1	4 Java
4,		Java		
12, 13		iOS		
4,				
9, 10	5	Ruby	2	4 iOS
11, 4				5 Ruby
9, 10	6	perl	0	
11, 4				
12, 13	7		1	7 delphi
4	delphi			
10,				
11				

for 迴圈



for 迴圈

□ 輸入 **N** 和 **N** 個整數，輸出其中最大的數。

○ 例如 N = 5，5 個整數 11, 45, 8, 13, 22，

```
01 def getMax(N):  
02     num = int(input())           #輸入第1個值  
03     maxVal = minVal = num;  
04     for i in range(N-1):  
05         num = int(input())       #輸入第2~N個值  
06         if (num>maxVal):  
07             maxVal = num  
08     return maxVal  
09  
10 def main():  
11     Num = int(input("Input a number:"))  
12     x = getMax(Num)  
13     print(x)  
14  
15 main()
```

編號每一行程式
劃出流程圖
寫下執行編號順序
寫下輸出內容

for 迴圈

□ 輸入 **N** 和 **N** 個整數，輸出其中**最大**和**最小**的數。

○ 例如 N = 5，5個整數 11, 45, 8, 13, 22，

```
01 def getMaxMin(N):  
02     num = int(input())  
03     maxVal = minVal = num;  
04     for i in range(N-1):  
05         num = int(input())  
06         if (num>maxVal):  
07             maxVal = num  
08         if (num<minVal):  
09             minVal = num  
10     return maxVal, minVal  
11  
12 def main():  
13     num = int(input("Input a number:"))  
14     x, y = getMaxMin(num)  
15     print(x, y)  
16  
17 main()
```

編號每一行程式
劃出流程圖
寫下執行編號順序
寫下輸出內容

for 迴圈

- 輸入 **N** 和 **N** 個整數，輸出其中 **第二大的數**。
 - 例如 $N = 5$ ，5個整數 11, 45, 8, 13, 22，

```
01 def inputData(data: list):
02     n = int(input())
03     for i in range(n):
04         data.append(int(input()))
05
06 def compute(data):
07     print(data)
08     r = sorted(data, reverse=True)
09     print(r)
10     print(r[1])
11
12 def testCompute():
13     data = [25, 48, 57, 79, 68]
14     compute(data)
15
16 def testInput():
17     data = []
18     inputData(data)
19     print(data)
20
21 def main():
22     data = []
23     inputData(data)
24     compute(data)
25
26 testInput()
27 #testCompute()
28 #main()
```

for 迴圈

□ function有迴圈

```
01 def myPrint01():
02     for i in range(0, 10, 1):
03         print(i)
04
05 def myPrint02(m, n):
06     for j in range(m, n, -1):
07         print(j, end=")      #不換行
08
09 def myPrint03(m):
10     for j in range(0, 2*m-1, 1):
11         print(j, end=")      #不換行
12
13 def main():
14     num = 5
15     myPrint01()
16     myPrint02(8, num)
17     myPrint03(num)
18     print()                  #預設換行
19 main()
```

```
01 def myPrint04(listData):
02     for i in listData:
03         print(i)
04
05 def main():
06     listData = ['a', 'b', 'c', 'd']
07     myPrint04(listData)
08
09 main()
```

編號每一行程式
劃出流程圖
寫下執行編號順序
寫下輸出內容

for - range

□ 要印出 1 個 '*'

□ 要印出 2 個 '*'

□ 要印出 3 個 '*'

□ 要印出 n 個 '*'

○ n 是 function 參數

○ n 從鍵盤輸入

寫成程式

```
def myPrint():  
    for i in range(1):  
        print('*', end="")
```

```
def myPrint():  
    for i in range(2):  
        print('*', end="")
```

```
def myPrint():  
    for i in range(3):  
        print('*', end="")
```

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

```
n = int(input())  
for i in range(n):  
    print('*', end="")
```

for - range

- 要印出 n 個 '*'
 - n 是 function 參數

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

記住這個



- 要印出
 - 第一行 1 個 '*'
 - 第一行 2 個 '*'
 - 第一行 3 個 '*'

寫成程式

```
def myPrintS():  
    myPrint(1)  
    print("")  
    myPrint(2)  
    print("")  
    myPrint(3)  
    print("")
```

- 寫成 LOOP 變成這樣

```
def myPrintS():  
    for i in range(1, 4):  
        myPrint(i)  
        print()
```

for - range

❑ 合起來寫

- 使用兩個 LOOP
- 不使用 myPrint()

```
def myPrintS():  
    for i in range(1, 4):  
        for j in range(i):  
            print('*', end="")  
        print("")
```

2個loop
↙ 程式太複雜



for - range

- 要印出 n 個 '*'
 - n 是 function 參數

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

記住這個



- 要印出

```
****  
***  
**  
*
```

寫成程式

```
def myPrintT():  
    myPrint(4)  
    print()  
    myPrint(3)  
    print()  
    myPrint(2)  
    print()  
    myPrint(1)  
    print()
```

- 寫成 LOOP 變成這樣

```
def myPrintS():  
    for i in range(4, 0, -1):  
        myPrint(i)  
        print()
```

for - range

□ 要印出 n 個自訂符號 mark

- mark 可以是 '*', '!'
- n, mark 是 function 參數

```
def myPrint(n, mark):  
    for i in range(n):  
        print(mark, end="")
```

記住這個



□ 輸入 N = 4，要印出

```
...*  
..**  
.****  
****
```

切割處理



寫成程式

- 第一行印3個.，1個*
- 第一行印2個.，2個*
- 第一行印1個.，3個*
- 第一行印0個.，4個*

□ 寫成LOOP變成?

```
def myPrintT(N):  
    myPrint(N-1, '!')  
    myPrint(1, '*')  
    print("")  
    myPrint(N-2, '!')  
    myPrint(2, '*')  
    print("")  
    myPrint(N-3, '!')  
    myPrint(3, '*')  
    print("")  
    myPrint(N-4, '!')  
    myPrint(4, '*')  
    print("")
```


for - range

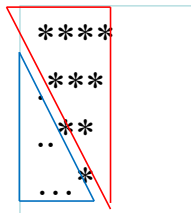
□ 寫成 LOOP 變成這樣

- 假設 $N = 4$
- 請將 for 展/拆開，看是否跟上面程式一樣

```
def myPrintT(N):  
    for i in range(1, N+1):  
        myPrint(N-i, '.')  
        myPrint(i, '*')  
        print("")
```

for - range

□ 輸入 $N = 4$ ，要印出



切割處理



寫成程式

```
def myPrintT(N):  
    myPrint(0, '.')  
    myPrint(4, '*')  
    print("")  
    myPrint(1, '.')  
    myPrint(3, '*')  
    print("")  
    myPrint(2, '.')  
    myPrint(2, '*')  
    print("")  
    myPrint(3, '.')  
    myPrint(1, '*')  
    print("")
```

□ 寫成 LOOP 變成?

寫成程式

for - range

□ 要印出 1 個 '1'

□ 要印出 12

□ 要印出 123

□ 要印出 123, .. n

○ n 是 function 參數

○ n 從鍵盤輸入

寫成程式

```
def myPrint():  
    for i in range(1,2):  
        print(i, end="")
```

```
def myPrint():  
    for i in range(1, 3):  
        print(i, end="")
```

```
def myPrint():  
    for i in range(1, 4):  
        print(i, end="")
```

```
def myPrint(n):  
    for i in range(1, n+1):  
        print(i, end="")
```

```
n = int(input())  
for i in range(1, n+1):  
    print(i, end="")
```

for - range

- ❑ 要印出 123, .. n
 - n 是 function 參數
- ❑ 要印出

```
1
12
123
1234
```

寫成程式

```
def myPrint(n):  
    for i in range(1, n+1):  
        print(i, end="")
```

```
def myPrintT():  
    myPrint(1)  
    print("")  
    myPrint(2)  
    print("")  
    myPrint(3)  
    print("")  
    myPrint(4)  
    print("")
```

```
def myPrintS():  
    for i in range(1, 5)  
        myPrint(i)  
        print()
```

記住這個



- ❑ 寫成 LOOP
- ❑ 寫成兩層 LOOP，不在一個 loop 內 call 另一個 function?

for - range

- ❑ 要印出 123,..n
 - n 是 function 參數
- ❑ 寫成 Loop

```
1
12
123
1234
```

寫成程式

```
def myPrint(n):
    for i in range(1, n+1):
        print(i, end="")
```

記住這個



```
def myPrintS():
    for i in range(1, 5)
        myPrint(i)
    print()
```

- ❑ 寫成兩層 Loop，不用 function?
 - 合併前兩個程式
 - 兩個 Loop 變數不能用同一個 i
 - 一個用 i, 一個用 j

寫成程式

```
def myPrintS():
    for i in range(1, 5):
        myPrint(i)
    for j in range(1, i+1):
        print(j)
    print()
```

for - range

- 要印出 123, ..n
 - n 是 function 參數
- 要印出

```
1234
123
12
1
```

寫成程式

```
def myPrint(n):
    for i in range(1, n+1):
        print(i, end="")
```

```
def myPrintT():
    myPrint(4)
    print("")
    myPrint(3)
    print("")
    myPrint(2)
    print("")
    myPrint(1)
    print("")
```

```
def myPrintS():
    for i in range(4, 0, -1)
        myPrint(i)
    print()
```

記住這個



- 寫成 LOOP

for - range

- 要印出 1357, ..n
 - n 是 function 參數
- 要印出

```
1
13
135
1357
```

寫成程式

```
def myPrint(n):
    for i in range(1, 2*n+1, 2):
        print(i, end="")
```

```
def myPrintT():
    myPrint(1)
    print("")
    myPrint(2)
    print("")
    myPrint(3)
    print("")
    myPrint(4)
    print("")
```

```
def myPrintS():
    for i in range(1, 5):
        myPrint(i)
        print()
```

記住這個



- 寫成 LOOP

for - range

□ 要印出 1357... $2*n$

○ n 是 function 參數

□ 要印出

1357
135
13
1

寫成程式

```
def myPrint(n):  
    for i in range(1,  $2*n+1$ , 2):  
        print(i, end="")
```

```
def myPrintT():  
    myPrint(4)  
    print()  
    myPrint(3)  
    print()  
    myPrint(2)  
    print()  
    myPrint(1)  
    print()
```

```
def myPrintS():  
    for i in range(4, 0, -1):  
        myPrint(i)  
        print()
```

記住這個



□ 寫成LOOP


for 迴圈

□ function 有迴圈

```
def myPrint01(m, n):  
    for x in range(m, n, 1):  
        print(x, end="")  
  
def myPrint02(m, n):  
    for y in range(m, n, -1):  
        print(y, end="")  
  
def myPrint03(m, n):  
    for z in range(m, 2*n-1, 2):  
        print(z, end="")  
  
def main():  
    m, n = 1, 5  
    myPrint01(m, n)  
    print()  
    myPrint02(n, m)  
    print()  
    myPrint03(m, n)  
    print()  
  
main()
```

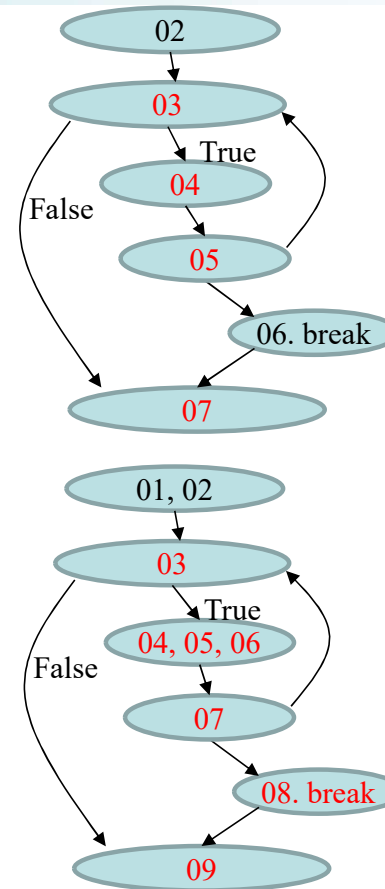
編號每一行程式
劃出流程圖
寫下執行編號順序
寫下輸出內容

1234



break

```
01 #當 i 數到5時就不做
02 def test01():
03     for i in range(1, 10):
04         number = number + i
05         if (i == 5):
06             break
07     print('end')
```



利用break在任何時候跳出迴圈

Encapsulation and generalization

□ Encapsulation 封裝

- 將單一功能包裝，
- 不同函式實作不同單一功能

```
01 def printMultiples(n: int):
02     for i in range(1, n+1):          #印 n 個
03         print(' %d ' %(i), end=")    # 從 1 開始，每次印 加 1
04     print()
05
06 def printAntiTriangle(m: int):
07     for i in range(m, 0, -1):        # 印 m 層，第一層印m個，逐層遞減
08         printMultiples(i)           # 每層印 i 個，間格 1
09
10 def main():
11     printAntiTriangle(3)              # 印 3 層，第一層印 3 個，逐層遞減，
12     printAntiTriangle(4)              # 印 4 層，第一層印 4 個，逐層遞減，
13
14 main()
```

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

Encapsulation and Generalization

□ Generalization 一般化

- 函式，可以設定參數，藉由調整參數增加功能
- 增加 **step**，可以調整開始的數字、每次印的間隔

```
01 def printMultiples(n: int, step: int):
02     for i in range(1, n+1):          # 印 n 個
03         print(' %d ' %(i*step), end='') # 從 step 開始，每次印 step 倍數
04     print()
05
06 def printAntiTriangle(m: int, step: int):
07     for i in range(m, 0, -1):        # 印 m 層，第一層印m個，逐層遞減
08         printMultiples(i, step)      # 每層印 i 個，間格 step .
09
10 def main():
11     printAntiTriangle(3, 6)          # 印 3 層，第一層印 3 個，逐層遞減，
12                                     # 每一層從 6 開始，間格 6 的倍數
13     printAntiTriangle(4, 8)          # 印 4 層，第一層印 4 個，逐層遞減，
14                                     # 每一層從 8 開始，間格 8 的倍數
15 main()
```

```
6 12 18
6 12
6
8 16 24 32
8 16 24
8 16
8
```

END

