Python
迴圈 Loop (I)

臺北科技大學資訊工程系

for

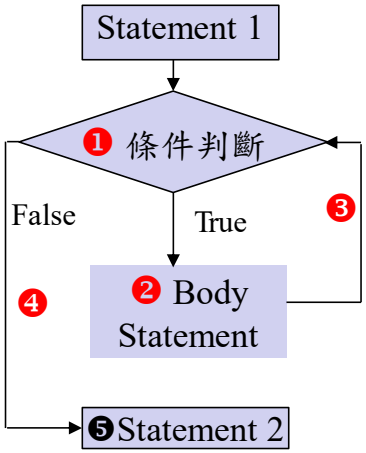
```
Statement 1
for var in sequence object :
    Body statement
Statement 2
```

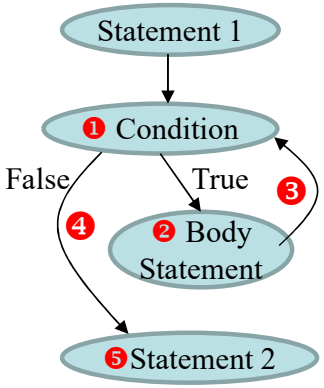
- for … in …
  - 序列物件(Sequence object)，有順序可數的元素
  - 控制變數 var 又稱迴圈變數/迴圈索引

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)，❺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
- range(0, 8)　，產生 0, 1, 2, 3, 4, 5, 6, 7
- range(3, 8, 2)，產生 3, 5, 7

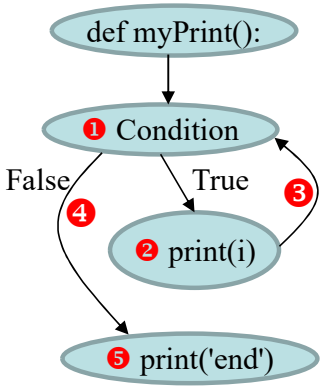
```
def iterOp():
    myRange = range(8)
    print(myRange)
    print(type(myRange))
    print(myRange[2])
    myRange = range(3,8,2)
    print(myRange)
    print(myRange[2])
```

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

range(3, 8, 2)
7
```

for - range()

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

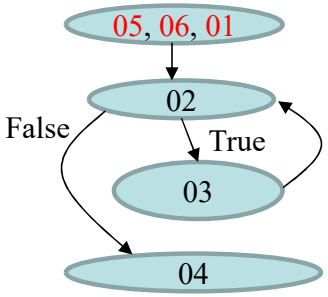
- range(0, 3, 1)，產生 0, 1, 2
  - ❶條件判斷，True
    - 找到 0, 1, 2 的第1個，指定給 i，i = 0
  - ❷印出 i， ❸回到❶
  - ❶條件判斷，True
    - 找到 0, 1, 2 的第2個，指定給 i，i = 1
  - ❷印出 i， ❸回到❶
  - ❶條件判斷，True
    - 找到 0, 1, 2 的第3個，指定給 i，i = 2
  - ❷印出 i， ❸回到❶
  - ❶條件判斷，False ❹
    - 序列物件找不到下一個資料，跳出迴圈❺

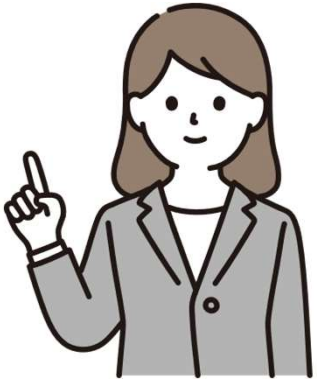


for - range()

- 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)
```



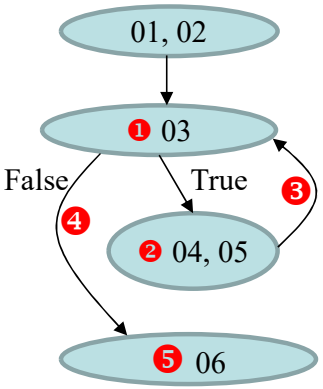


考試使用此種畫法

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 + ... +10 的計算結果 55
- 輸出 1 * 2 * 3 * 4 * 5 * ... *10 的計算結果 3628800

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)
    print(getSum(num))
    print(getProduct(num))

main()
```

```
def myPrint(n):
    for i in range(0, n, 1): # 0, 1, …, 9
        print(i, end = '')
```

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

Exercise

- 輸入開始值 m、結束值 n (包含結束值)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):
    sum = 0
    for i in range(m, n+1, step):
        Sum += i
    print("數值相加總合：", sum)
    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 ~ n 的偶數，相加總合、相乘總和?
  - 你可以假設 m 輸入的是偶數
- 輸出兩數(m ~ n 的偶數，相加總合、相乘總和)
  - m + (m+2) + (m+4) + (m+6) + ... + n = ?
  - m * (m+2) * (m+4) * (m+6) * ... * n = ?

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 forOps(m, n):
    sum = 0
    multi = 1
    for index in range(m, n+1, 2):
        sum += index
        multi *= index
        print(index, end = ', ')
    print("\n相加總合：", sum)
    print("\n相乘總合：", multi)

forOps(6, 12)
```

Exercise

字串：ATCgATAgcTCGaTCG
字元個數： 16

大寫字母：ATCATATCGTCG
大寫字母個數： 12

- 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有幾個大寫字母?

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

```
01 def forOps():
02     myString = "ATCgATAgcTCGaTCG"
03     for index in myString:
04         if index.isupper():
05             print(index, end = '')
06
07 forOps()
```

```
ATCATATCGTCG
```

```
01 def forOps2():
02     myString = "ATCgATAgcTCGaTCG"
03     bigCount = 0
04     allCount = 0
05     print("大寫字母：", end='')
06
07     for index in myString:
08         allCount += 1
09         if index.isupper():
10             bigCount += 1
11         print(index, end = "")
12
13     print("\n有幾個字母：", allCount)
14     print("有幾個大寫字母：", bigCount)
15
16 forOps2()
```

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

```
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()
```

寫下執行編號順序

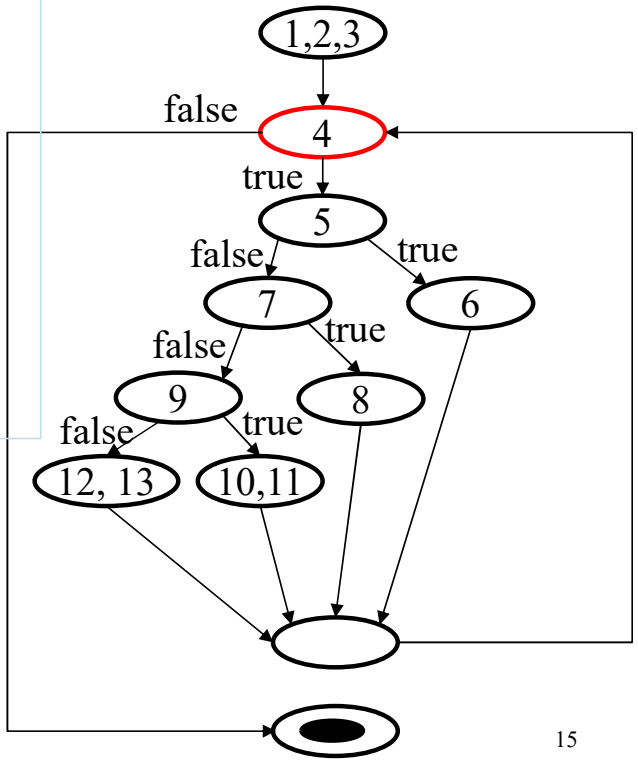
寫下輸出內容

寫下執行編號順序	i	index	i % 3	output
14, 1, 2, 3, 4	1	asm	1	
5, 7, 9, 10,				1 asm
11, 4	2	C	2	
5, 7, 9, 10				2 C
11, 4	3	python	0	
5, 6				3 python
4,		C++		
12, 13	4		1	
4,		Java		
7, 8				4 Java
4,		iOS		
9, 10				4 iOS
11, 4	5	Ruby	2	
9, 10				5 Ruby
11, 4	6	perl	0	
12, 13	7		1	
4		delphi		
10,				7 delphi
11				

for 迴圈

```
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()
```

劃出流程圖



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     maxValue = minValue = num;
04     for i in range(N - 1):
05         num = int(input())                #輸入第2~N個值
06         if (num>maxValue):
07             maxValue = num
08     return maxValue
09
10 def main():
11     Num = int(input("Input a number:"))
12     x = getMax(Num)
13     print('max = ', 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     maxValue = minValue = num;
04     for i in range(N-1):
05         num = int(input())
06         if (num>maxValue):
07             maxValue = num
08         if (num<minValue):
09             minValue = num
10     return maxValue, minValue
11
12 def main():
13     num = int(input("Input a number:"))
14     x, y = getMaxMin(num)
15     print('Max, Min = ', 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('Number of numbers: '))
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      compute(data)
21
22  def main():
23    data = []
24    inputData(data)
25    compute(data)
26
27  testInput()
28  #testCompute()
29  #main()
```

for 迴圈

- function有迴圈

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

876

012345678
```

```
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()
```

```
a
b
c
d
```

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='')
myPrint(6)
```

```
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('')
```

不要用這個

- 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('')
```

不要用這個

- 寫成 LOOP 變成這樣

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

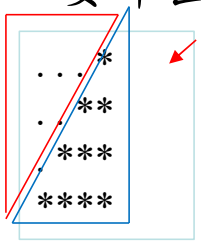
for - range

- 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

  - 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*

寫成程式

```
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('')
myPrintT(4)
```

不要用這個

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

- 寫成LOOP變成?

for - range

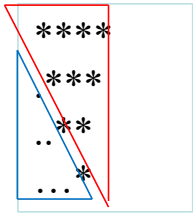
- 寫成 LOOP 變成這樣
  - 假設 N = 4
  - 請將 for 展/拆開，看是否跟上面程式一樣

```
def myPrintT(N):
    for i in range(1, N+1):
        myPrint(N-i, '.')
        myPrint(i, '*')
        print('')
myPrintT(4)
```

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

for - range

- 輸入 N = 4，要印出



切割處理

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

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='')
```

記住這個

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

- 寫成 LOOP

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

- 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='')
```

記住這個

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

- 寫成 LOOP

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

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='')
```

記住這個

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

- 寫成 LOOP

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

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='')
```

記住這個

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

- 寫成 LOOP

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

for 迴圈

- function 有迴圈

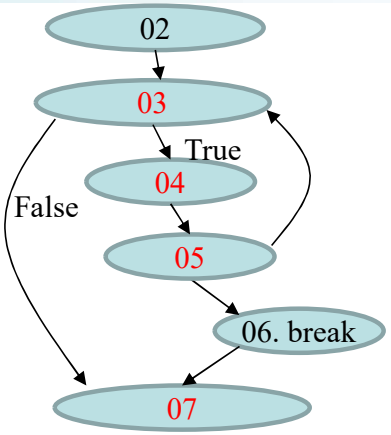
```
def myPrint01(m, n):
    for x in range(m, n, 1):
        print(x, end='')

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
5432
1357
```

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

```
def printMultiples(n: int):
    for i in range(1, n+1):          #印 n 個
        print(' %d ' %(i), end='')   # 從 1 開始，每次印 加 1
    print()

def printAntiTriangle(m: int):
    for i in range(m, 0, -1):        # 印 m 層，第一層印m個，逐層遞減
        printMultiples(i)            # 每層印 i 個，間格 1

def main():
    printAntiTriangle(3)             # 印 3 層，第一層印 3 個，逐層遞減，
    printAntiTriangle(4)             # 印 4 層，第一層印 4 個，逐層遞減，

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

```
def printMultiples(n: int, step: int):
    for i in range(1, n+1):                 # 印 n 個
        print(' %d ' %(i*step), end='')     # 從 step 開始，每次印 step 倍數
    print()

def printAntiTriangle(m: int, step: int):
    for i in range(m, 0, -1):               # 印 m 層，第一層印m個，逐層遞減
        printMultiples(i, step)             # 每層印 i 個，間格 step .

def main():
    printAntiTriangle(3, 6)                 # 印 3 層，第一層印 3 個，逐層遞減，
                                            # 每一層從 6 開始，間格 6 的倍數
    printAntiTriangle(4, 8)                 # 印 4 層，第一層印 4 個，逐層遞減，
                                            # 每一層從 8 開始，間格 8 的倍數
main()
```

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

END