Python程式設計

流程語法

高師大數學系

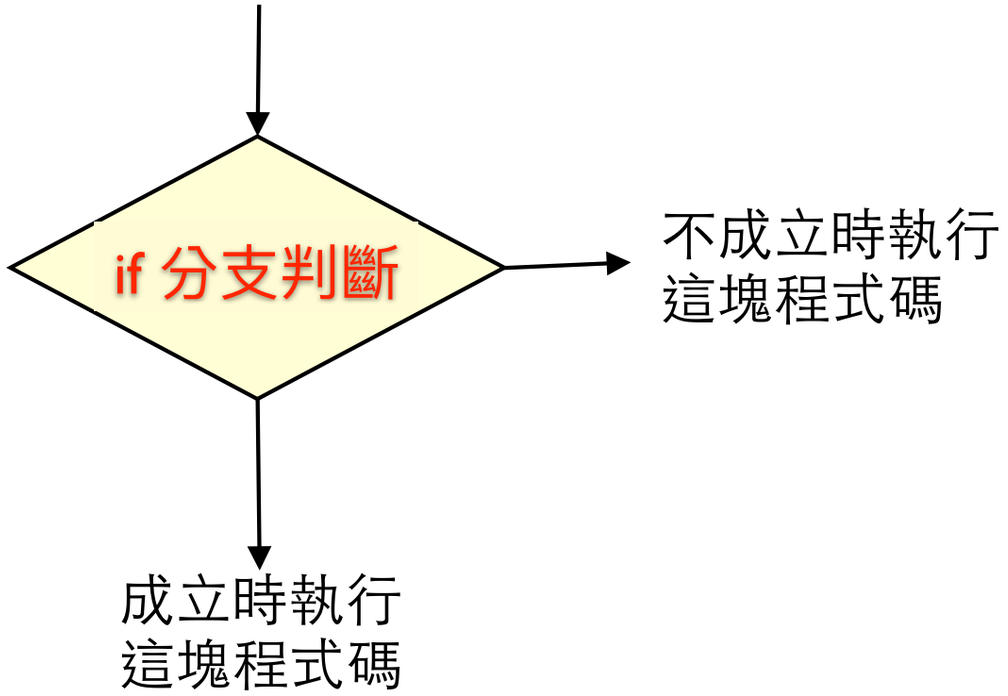
葉倚任

if 分支判斷

- 流程與法中最簡單也最常見的是 if 分支判斷

```
In [2]: x = input('Your score: ')
        num = int(x)
        if num > 60:
            print('You Pass!')
```

```
Your score: 80
You Pass!
```



if 分支判斷：注意事項

```
In [2]: x = input('Your score: ')
        num = int(x)
        if num > 60:
            print('You Pass!')

Your score: 80
You Pass!
```

程式區塊使用冒號「:」開頭

同一區塊範圍要有相同的縮排

- 不可混用不同空白數量
- 不可混用空白與Tab

Python 的建議是使用四個空白作為縮排

if 分支判斷：if-else

- 在上一頁的例子，我們可以搭配 else 使用。

In [4]:
```
x = input('Your score: ')
num = int(x)
if num >= 60:
    print('You Pass!')
else:
    print('You Fail!')
```

```
Your score: 38
You Fail!
```

練習

- 讓使用者輸入一個整數，若是偶數則輸出此數為偶數，若是奇數則輸出此數為奇數。

if 分支判斷：if-else配對問題

先來看一段 C 語言

```c
#include <stdio.h>

int main(void){

    int num = 38;
    if (num<60)
        if (num>=40)
            printf("Hahaha\n");
    else
        printf("Hehehe\n");

    return 0;
}
```

else 與最近的 if 配對

if 分支判斷：if-else配對問題

- Python 中的區塊定義就沒有上一頁的問題

```
In [6]: x = input('Your score: ')
num = int(x)
if num < 60:
    if num>= 40:
        print('Re-Exam!')
else:
    print('You Pass!')
```

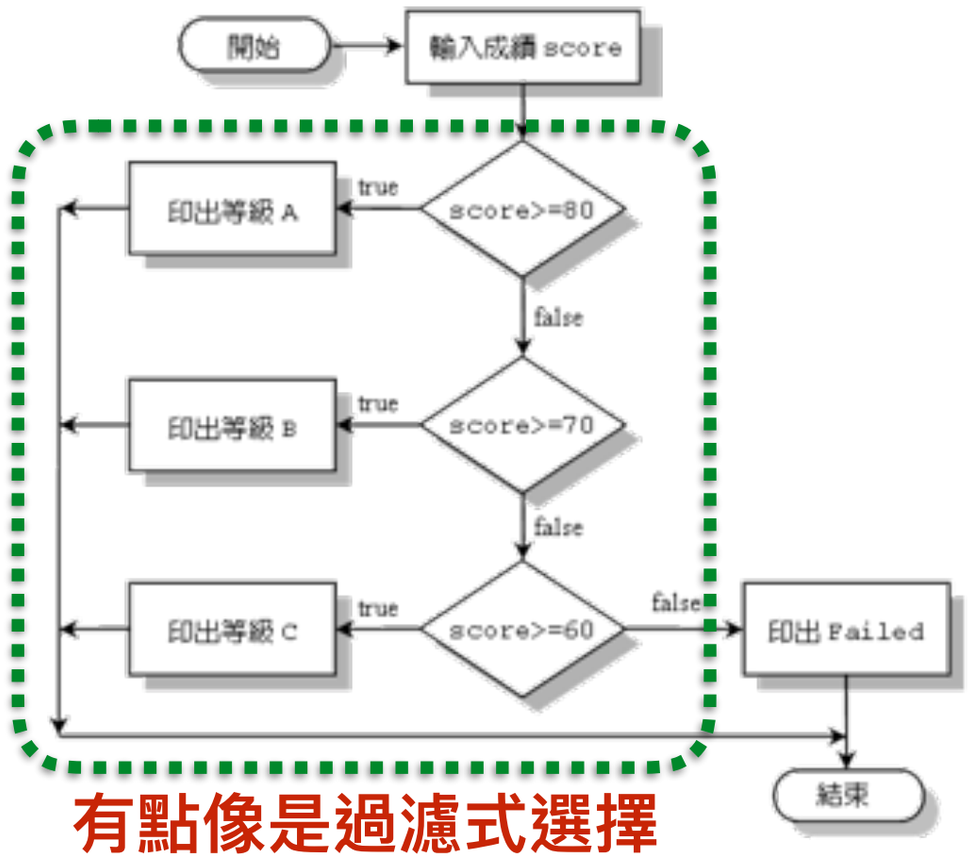
這兩個配對，以對齊縮排為主

```
Your score: 30
```

if 分支判斷：多重判斷(if..elif..else)

In [8]:
```
x = input('Your score: ')
num = int(x)
if num>=80:
    print(num,'is A')
elif num>=70:
    print(num,'is B')
elif num>=60:
    print(num,'is C')
else:
    print('You Fail!')
```

```
Your score: 75
75 is B
```



有點像是過濾式選擇

練習

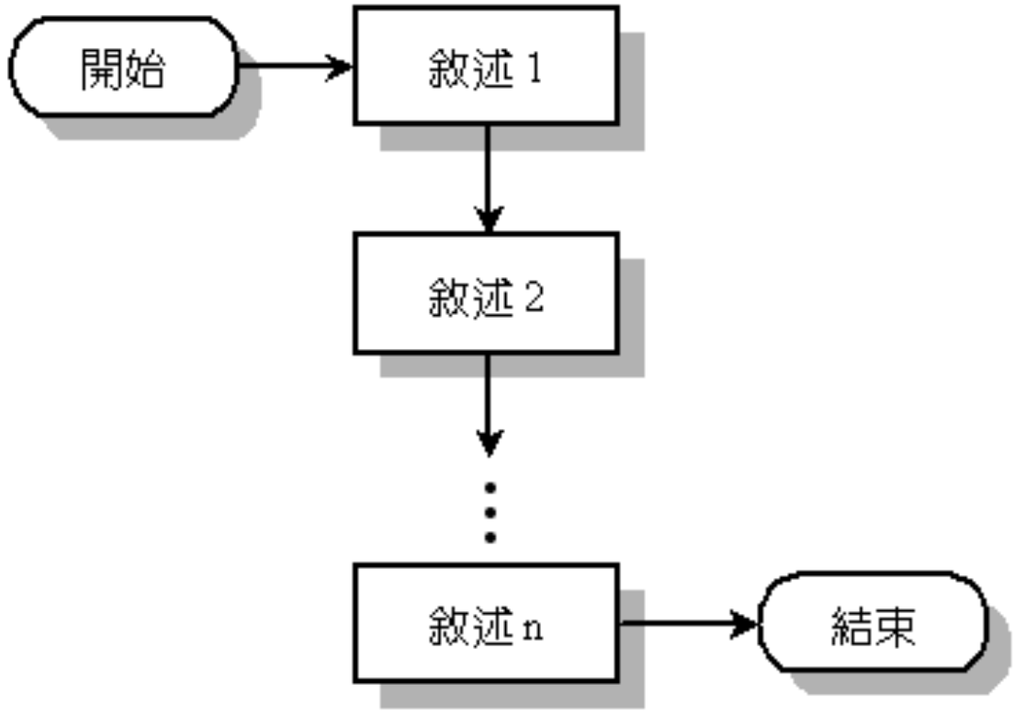
- 請使用者輸入兩個數，輸入下列三種狀況其中一種
  - 第一個數大於第二個數
  - 第二個數大於第一個數
  - 第一個數等於第二個數

結構化程式設計

- 結構化的程式設計包含有下面三種結構：
  - 循序性結構（sequence structure）
  - 選擇性結構（selection structure）
  - 重複性結構（iteration structure）

循序性結構(sequence structure)

- 程式的執行流程是由上而下，一個接著一個敘述依序執行

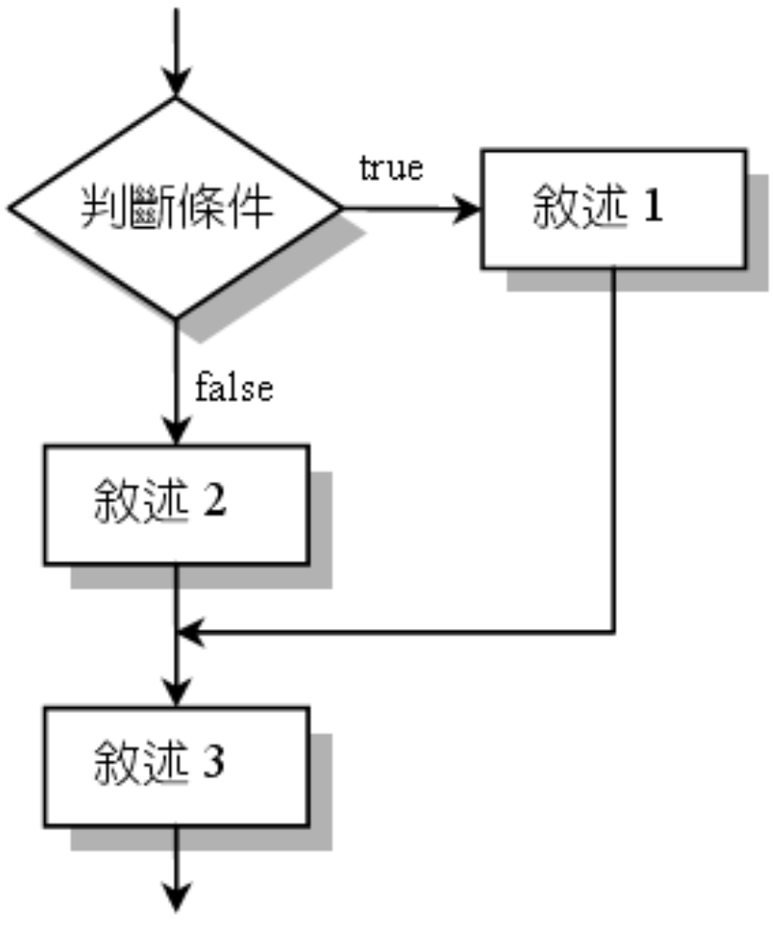



```
In [35]: num = [1,2,3,4,5]
         print(num[:])
         print(num[0:6:2])
         print(num[::-1])
```

```
[1, 2, 3, 4, 5]
[1, 3, 5]
[5, 4, 3, 2, 1]
```

選擇性結構(selection structure)

- 選擇性結構是根據條件的成立與否，再決定要執行哪些敘述的結構。




```
In [6]: x = input('Your score: ')
        num = int(x)
        if num < 60:
            if num>= 40:
                print('Re-Exam!')
        else:
            print('You Pass!')
```

```
Your score: 30
```

介紹重複性結構，先來看一個例子

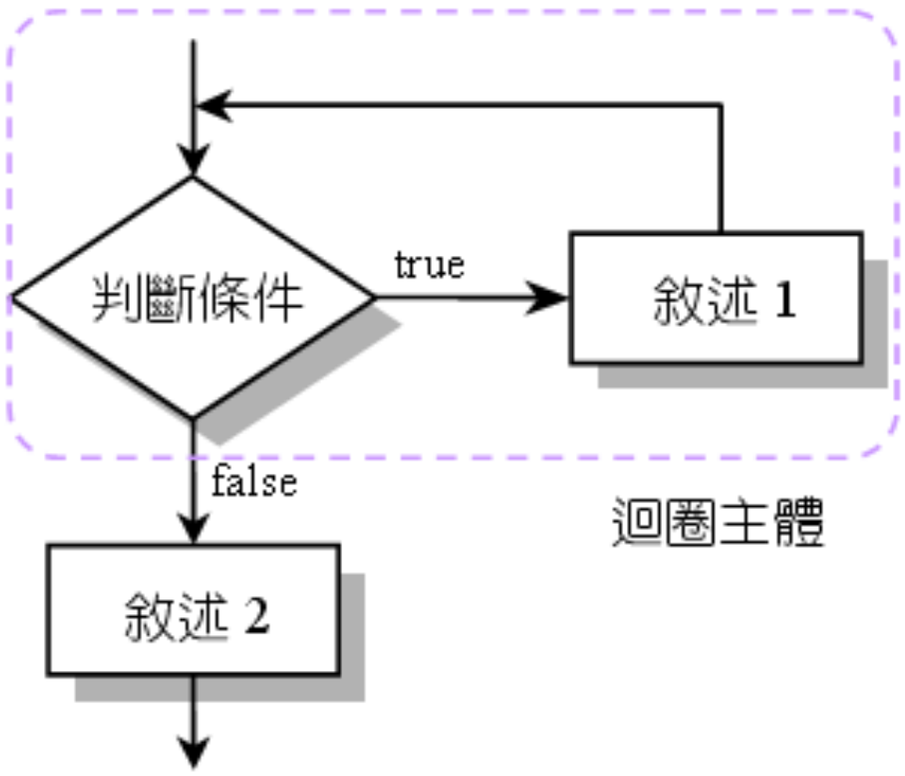
```
In [9]: print('You cannot PASS!')
        You cannot PASS!
```

```
In [10]: print('You cannot PASS!')
         print('You cannot PASS!')
         print('You cannot PASS!')
```


```
In [11]: print('You cannot PASS!')
         print('You cannot PASS!')
         print('You cannot PASS!')
         print('You cannot PASS!')
         print('You cannot PASS!')
         print('You cannot PASS!')
         print('You cannot PASS!')
         print('You cannot PASS!')
         print('You cannot PASS!')
         print('You cannot PASS!')
```

重複性結構(iteration structure)

- 重複性結構是根據判斷條件成立與否，決定程式段落的執行次數
- 也就是程式在某些敘述區塊反覆執行，直到符合測試條件時才離開
- C 語言提供的重複性結構可用迴圈敘述來完成。
- 迴圈有 for、while



while 迴圈

while 條件式:
陳述句

在條件式成立時，會執行 while 區塊。

```
In [13]: cnt=1
         while cnt <=10:
             print('You cannot PASS!')
             cnt = cnt+1
```

```
i=1  You cannot PASS!
i=2  You cannot PASS!
i=3  You cannot PASS!
i=4  You cannot PASS!
i=5  You cannot PASS!
i=6  You cannot PASS!
i=7  You cannot PASS!
i=8  You cannot PASS!
i=9  You cannot PASS!
i=10 You cannot PASS!
i=11
```

while 迴圈-另一個範例

- 重複抽數字

```
In [16]: import random          ← import package
         drawNum = random.randint(1,10)  ← 隨機產生 1~10 之整數亂數
         while drawNum!=7:
             print('You got {0:2d}. Please re-draw!'.format(drawNum))
             drawNum = random.randint(1,10)
         print('Nice! You got',drawNum)
```

```
You got  9. Please re-draw!
You got  9. Please re-draw!
You got  4. Please re-draw!
You got  9. Please re-draw!
You got  3. Please re-draw!
You got  2. Please re-draw!
You got 10. Please re-draw!
Nice! You got 7
```

練習

- 利用 while 印出九九乘法表

for 迴圈

- 想要循序迭代某個序列，例如字串、list、tuple，可以使用 **for in** 陳述句。

```
In [21]: numlist = [2,3,1,4,5]
         for i in numlist:
             print(i)
```

```
2
3
1
4
5
```

要被迭代的序列，是放在 in 之後，for in 會依索引順序逐一取出元素，並指丟給 in 之前的變數。

for 迴圈

- 回到 You cannot PASS! 的例子，亦即要重複10次

```
In [22]: for i in range(1,11):
             print('You cannot PASS!')
```

使用 range() 函式來產生指定的數字範圍

```
You cannot PASS!
You cannot PASS!
You cannot PASS!
You cannot PASS!
You cannot PASS!
You cannot PASS!
You cannot PASS!
You cannot PASS!
You cannot PASS!
You cannot PASS!
```

range() 函式的形式是 range(start, stop[, step])

- start 省略時，預設是0
- step 是步進值，預設是1
- 上述的例子會產生1~10的數字

練習

- 利用 for 印出九九乘法表

pass、break、continue

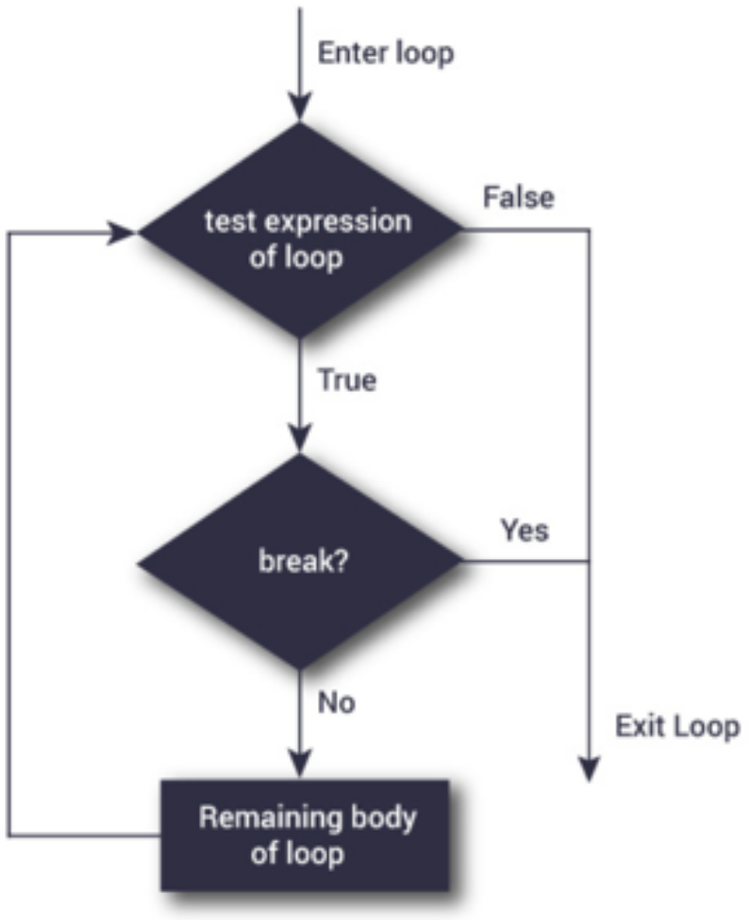
- 在某個區塊中，並不想做任何的事情，或者是稍候才會寫些什麼，可以先放個 pass。

In [24]:
```
num=55;
if num>=60:
    print('You PASS!')
else:
    pass
```

← pass 就真的是 pass，什麼都不做

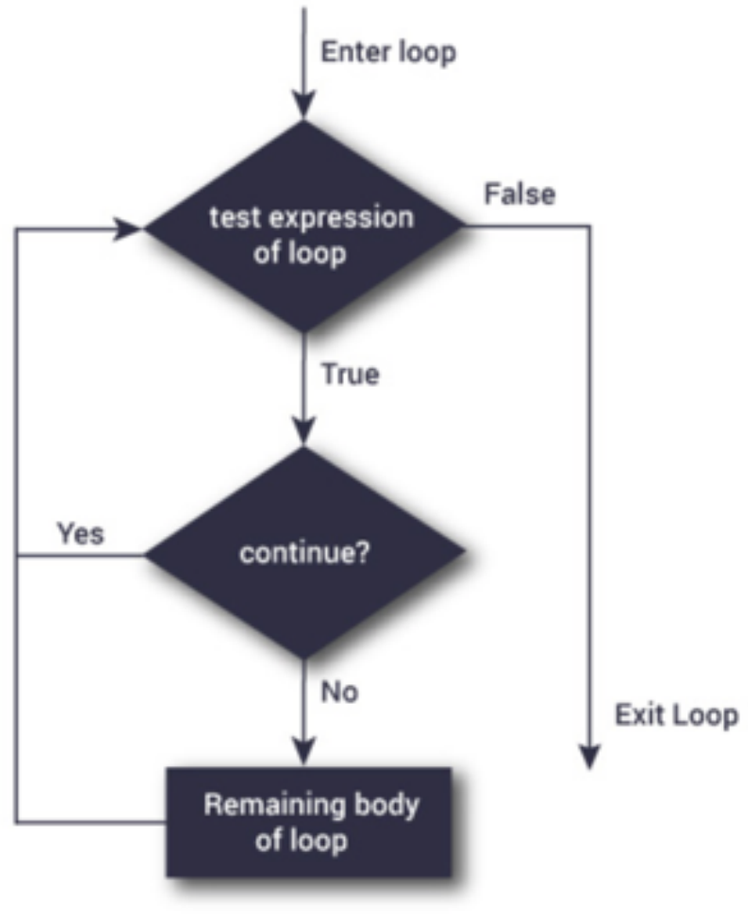
pass、break、continue

break



略過迴圈主體的其餘部分，
執行迴圈之後的敘述

continue



略過迴圈主體的其餘部分，
直接開始下一個迴圈循環

pass、break、continue

In [26]:
```
cnt=0
for i in range(1,11):
    if i==7:
        break
    cnt=cnt+1
print(cnt)
```

6

In [27]:
```
cnt=0
for i in range(1,11):
    if i==7:
        continue
    cnt=cnt+1
print(cnt)
```

9

練習

- 在之前抽號碼的例子，我們是用while來實作。請換成用 for 與 break 實作抽號碼的例子。

先再看一個問題

- 給一 list 為 num =[1,3,5,2,4]
- 若要將這些數字全部進行平方運算該處如何處理？

```
num=[1,3,5,2,4]
numSQ = [];
for item in num:
    numSQ.append(item ** 2)
print('New list is',numSQ)
```

```
New list is [1, 9, 25, 4, 16]
```

將一個 list 轉成另一個 list 是很常見的操作，python 提供了 for comprehension 語法滿足這類需求。

for Comprehension

- 上一頁之程式碼可以改成如下：

```
num=[1,3,5,2,4]

numSQ=[ item**2 for item in num ]

print('New list is',numSQ)
```

```
New list is [1, 9, 25, 4, 16]
```

逐一迭代出num之內容給item變數，之後執行 for 左方的 item ** 2。

使用 [] 包含起來，表示每次迭代的運算結果，會被收集成一個 list

for Comprehension 與條件式結合

- for comprehension 也可以與條件式結合，構成一個過濾的功能。
- 例如想收集某個 list 中的奇數元素至另一 list。

不使用 for comprehension

```
num=[2,1,3,4,5,8,6,7]
odds=[]
for item in num:
    if item%2:
        odds.append(item)
print('odds list is',odds)
```

```
odds list is [1, 3, 5, 7]
```

使用 for comprehension

```
num=[2,1,3,4,5,8,6,7]

odds=[item for item in num if item%2]

print('odds list is',odds)
```

```
odds list is [1, 3, 5, 7]
```

if 的條件成立時，for左邊的運算式才會被執行

練習

- 給定一 list ，將其偶數變成0
  例如：num = [1,2,3,4,5] => newnum =[1,0,3,0,5]
- 不使用 for comprehension 來實作
- 使用 for comprehension 來實作