

# Python 程式設計

林奇賦 [daky1983@gmail.com](mailto:daky1983@gmail.com)

# Outline

- ▶ 流程控制
- ▶ 迴圈

# 布林運算

- ▶ 有三種布林運算 and, or, not


運算	範例	結果
or	$2==3$ or $3 < 7$	True
and	$2==3$ and $3 < 7$	False
not	not $3 < 7$	False



# 比較運算子

運算符號	描述
<	小於
<=	小於或等於
>	大於
>=	大於或等於
==	比較值是否相等
!=	比較值是否不相等
is	比較是否同一個物件(id)
is not	比較是否不同個物件(id)

# 邏輯運算

- ▶ 邏輯運算是針對真假值(布林值)的運算
- ▶ 布林型態只有兩種值: **True** 跟 **False** 
- ▶ 布林語境(**Boolean context**): 談論真假, 運算真假的情境
- ▶ 在布林語境中, **0**和任何的**空資料**代表**False**, 其他代表**True**(通常**True**會 跟**1**連結), **None**在布林語境中也是**False**

# if (如果...就...)

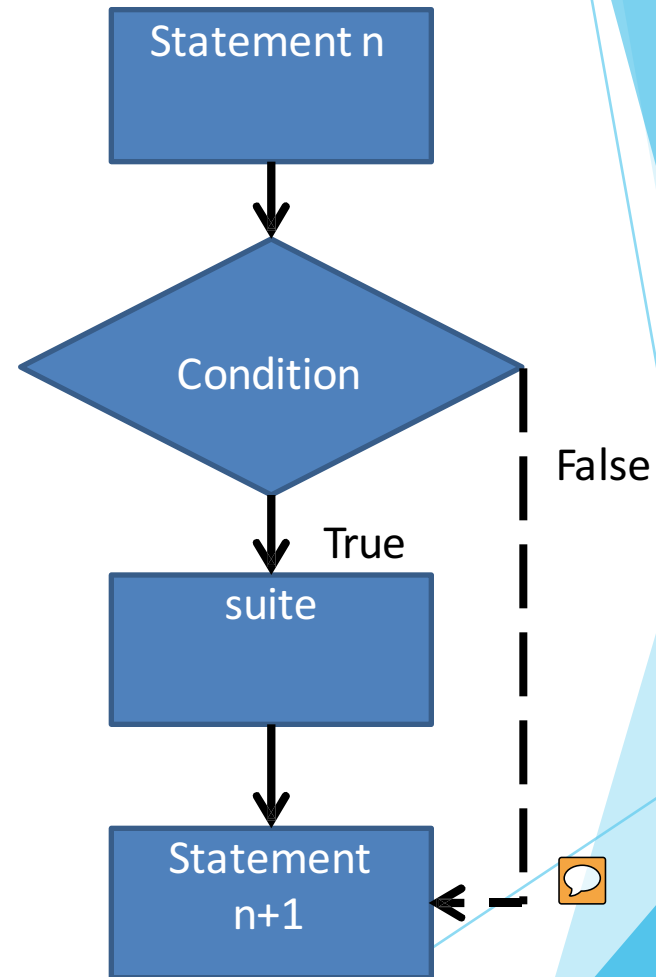
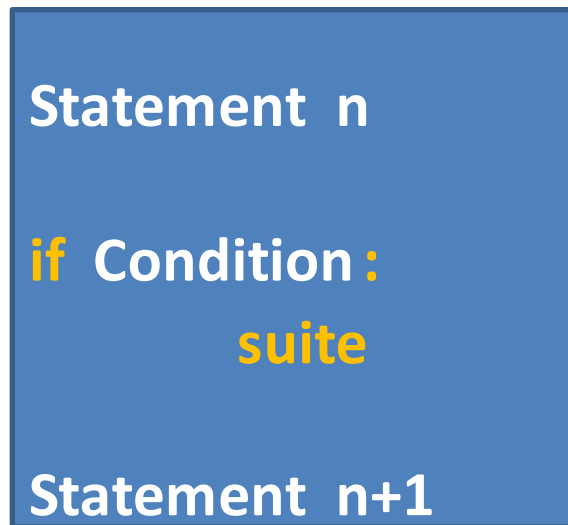
- if是最單純的一種條件分支敘述句
- If 後面接 condition，最後需要加上冒號：
- 冒號:之後的下一行程式碼記得要縮排
- 當條件(condition)成立時(True)，執行冒號後面的suite程式碼，若條件不成立(False)，略過整個suite開始執行suite之後一行程式碼
- 用condition來選擇suite程式碼做或不做



Condition是一個完整的敘述，並且python會在布林語境中解讀，所以會是一個真假敘述，最常使用的是比較運算式

# if

- 以下是流程示意圖



# if 範例

```
1 #-*-coding:utf-8 -*-
2 #範例程式 EX02_01.py
3 #判斷 2的10次方是否等於1024
4
5 if 2**10 == 1024:
6     print("2^10=1024")
```

原始碼:

[https://gist.github.com/chifu/f044779487741c829734#file-ex02\\_01-py](https://gist.github.com/chifu/f044779487741c829734#file-ex02_01-py)



# if, else (2選1)

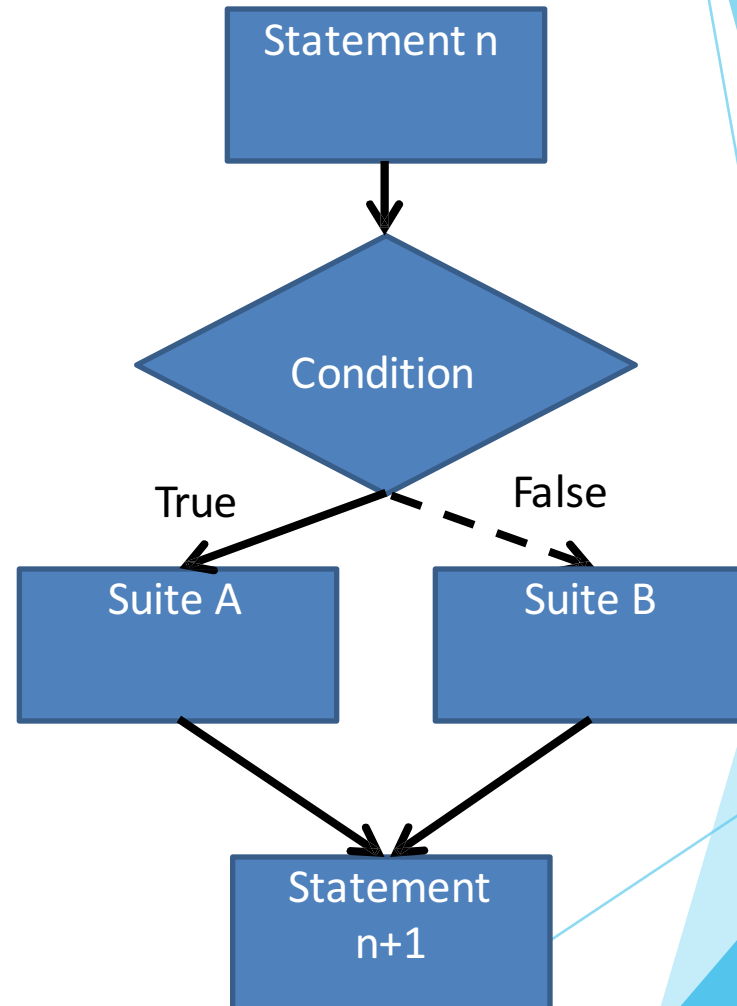
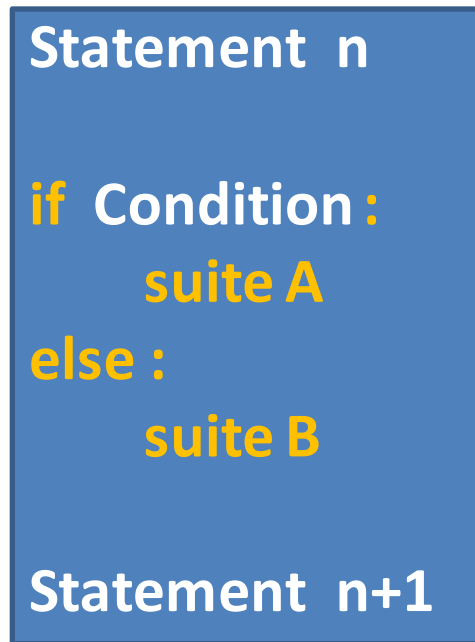
- 利用if/else敘述可以根據條件選擇執行區塊A或B

```
if Condition :  
    suite A  
else :  
    suite B
```

- 當條件(condition)成立時(True), 執行if冒號後面的suite A程式碼, 執行完後跳到suite B之後一行敘述執行
  - 若條件不成立(False), 則執行suite B之程式碼, 執行完後跳到suite B之後一行敘述執行
- 用condition來選擇做suite A還是suite B

# if, else

- 流程示意圖



# if, else 範例

```
1 #-*-coding:UTF-8 -*-  
2 #範例程式 EX02_02.py  
3 #判斷輸入的數字是奇數還是偶數  
4  
5 num = int(input('Please input a num:'))  
6  
7 if num % 2 == 0:  
8     print(num, '是偶數')  
9 else:  
10    print(num, '是奇數')
```

原始碼:

[https://gist.github.com/chifu/f044779487741c829734#file-ex02\\_02-py](https://gist.github.com/chifu/f044779487741c829734#file-ex02_02-py)

# if, elif, else (多選1)

- 當選擇超過兩種的時候使用if, elif(**else if**), else的語法來決定執行區塊A或B或C...

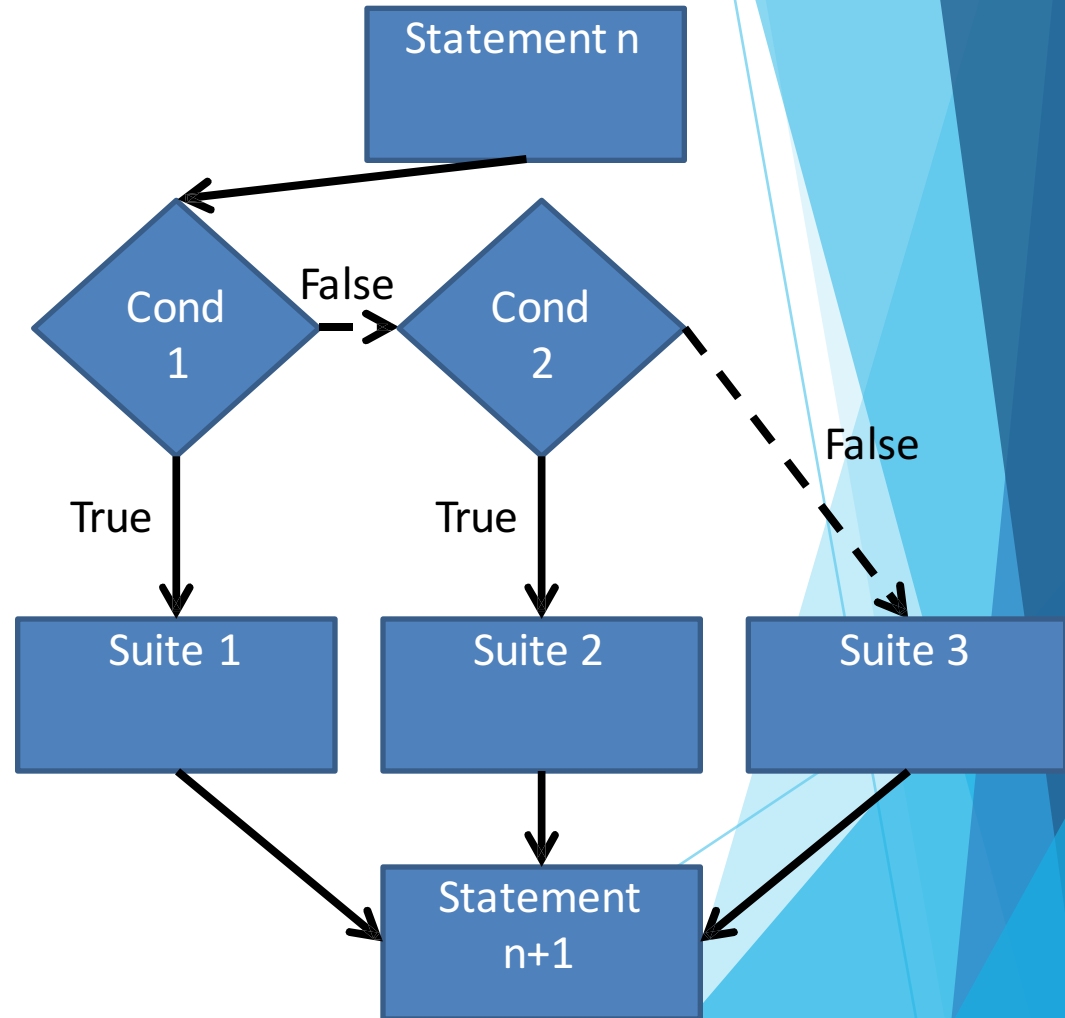
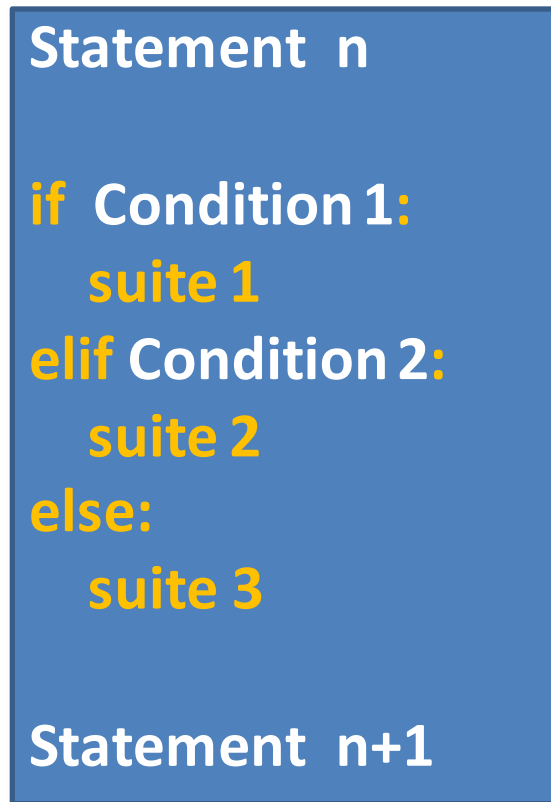
```
if Condition 1:  
    suite 1  
elif Condition 2:  
    suite 2  
    ...  
elif Condition k:  
    suite K  
    ...  
else:  
    suite N
```

- 當條件(condition k)成立時(True), 執行冒號後面的suite K 程式碼, 執行完後跳到suite N 之後一行敘述執行
- 若條件(condition k)不成立(False), 則往下開始判斷 condition K+1
- 所有condition皆不成立則執行 suite N 程式碼

- 用多個condition來決定做suite K!

# if, elif, else (多選1)

- 流程示意圖



# if, elif, else 範例(比大小)

```
1 #-*-coding:UTF-8 -*-
2 #範例程式 EX02_03.py
3 #輸入兩個數字比大小
4
5 num1 = int(input('Please input a num1:'))
6 num2 = int(input('Please input a num2:'))
7
8 if num1 == num2:
9     print(num1, '等於', num2)
10 elif num1 < num2:
11     print(num1, '小於', num2)
12 else:
13     print(num1, '大於', num2)
```



原始碼:

<https://gist.github.com/chifu/f044779487741c829734> - file-ex02\_03.py

# if, elif, else 範例

```
# 計算BMI並輸出分級值  
# BMI = 體重 (kg) / 身高 (m^2)
```

分 級	身體質量指數
體重過輕	$\text{BMI} < 18.5$
正常範圍	$18.5 \leq \text{BMI} < 24$
過 重	$24 \leq \text{BMI} < 27$
輕度肥胖	$27 \leq \text{BMI} < 30$
中度肥胖	$30 \leq \text{BMI} < 35$
重度肥胖	$\text{BMI} \geq 35$

原始碼:

<https://gist.github.com/chifu/f044779487741c829734> - file-ex02\_04.py

參考網頁：<http://www.scpo.nccu.edu.tw/show/part1/b/B2/bmi.htm>

# for 迴圈

- for迴圈是另外一個可以重複進行運算的結構, 以下是for-in的基本語法

```
for Var in iterable object :  
    suite
```

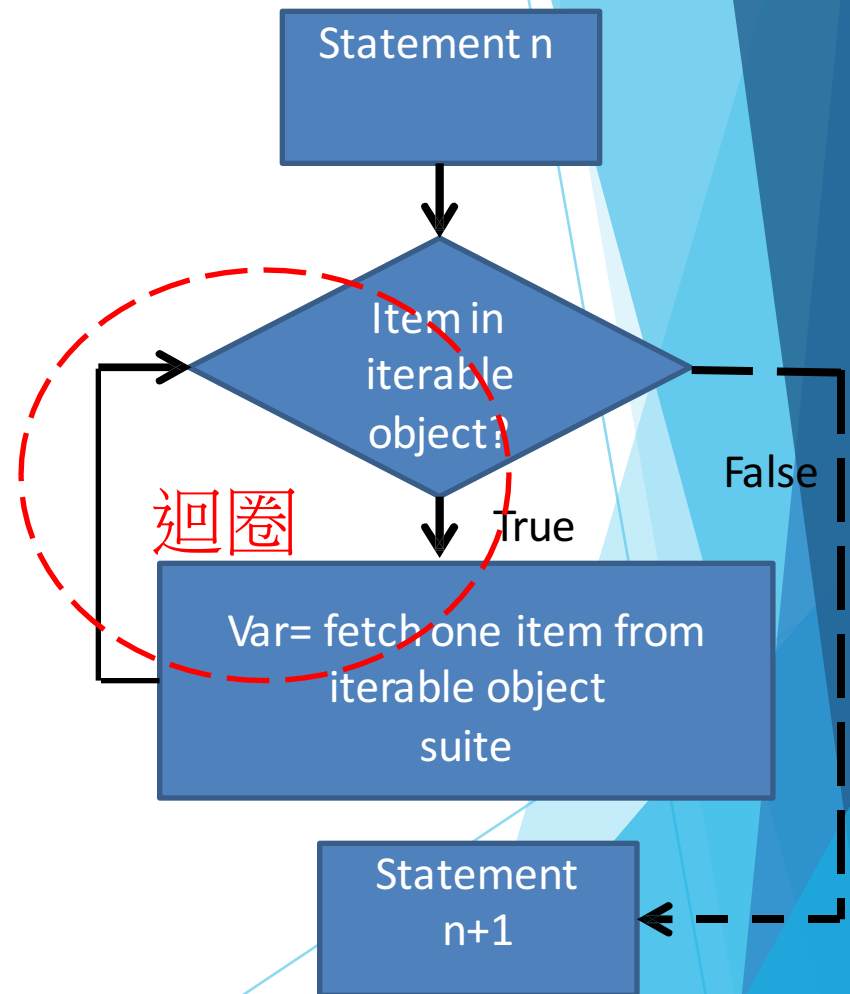
- Iterable object 是指可迭代物件, 可以想像成這種物件裡面有可數的項目可依特定順序一個一個取出
- Var我們稱控制變數又或迴圈變數
- for迴圈的執行流程
  - ① 自可迭代物件中取出一個項目, 代入至Var中
  - ② 執行suite
  - ③ 回到第一步直到可迭代物件中的項目盡皆取出
    - 這種依次取出(探訪)並且進行代入的動作稱為迭代
- 當然, break 和 continue 也可以在 for 迴圈中出現



# for 迴圈

- 以下是流程示意圖

```
Statement n
for Var in iterable object :
    suite
Statement n+1
```



# for 迴圈 範例



- ▶ 輸入一個數字 $n$ ，計算 $1+2+3+\dots+n$ 的總和為多少？

# while 迴圈

- 當條件成立(True)時, 進行區塊(suite)運算



**while** Condition :

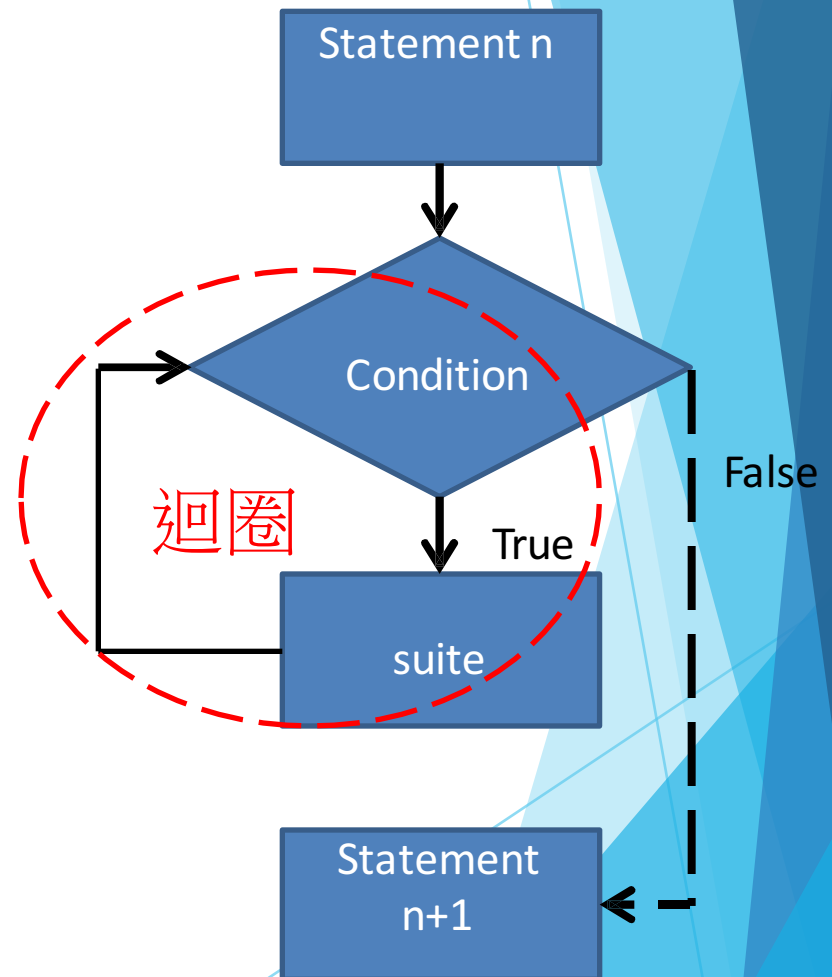
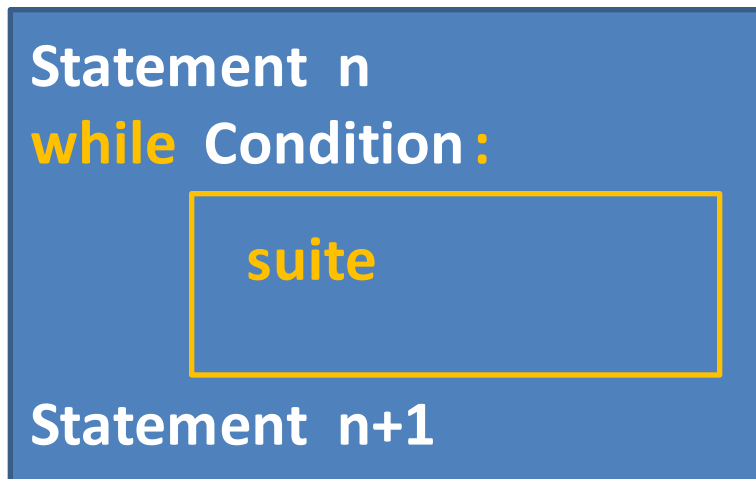
**suite**

- 區塊執行完畢後, 再次檢查條件, 若依然成立則執行suite否則開始執行區塊之後的敘述
- 這種重複的結構我們稱為迴圈
- 不再繼續執行區塊的動作稱為跳出迴圈或離開迴圈



# while 迴圈

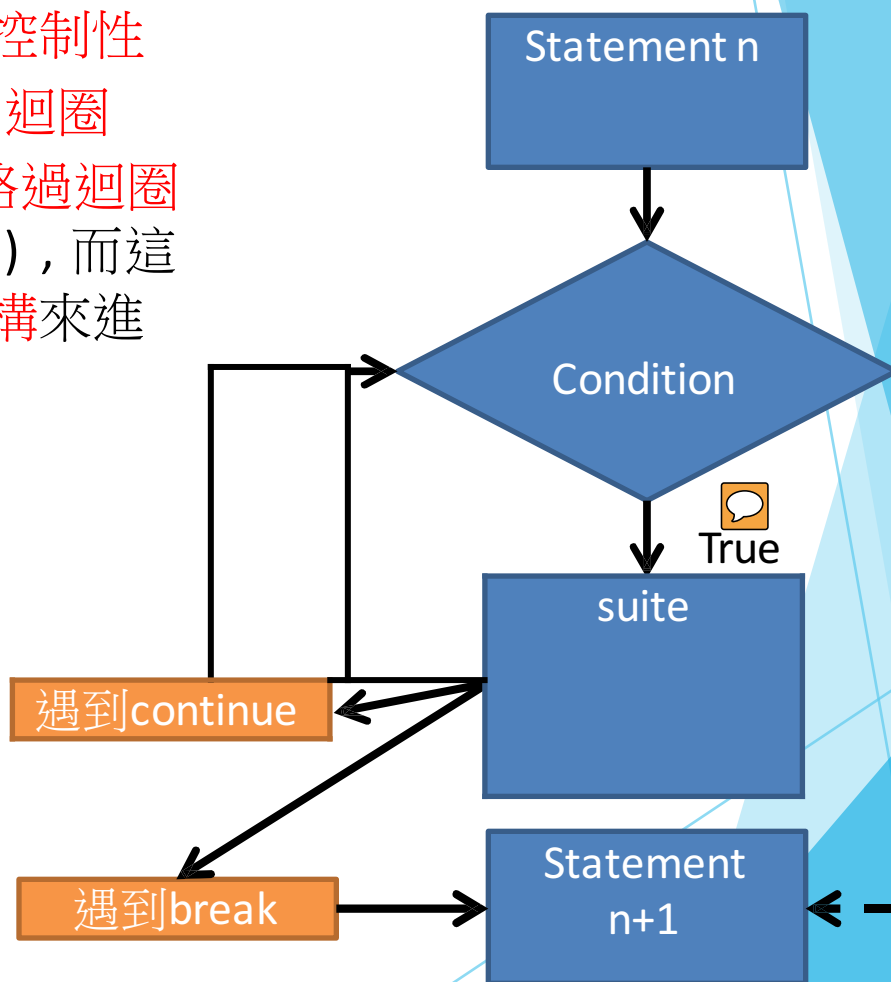
- 以下是流程示意圖




# break敘述與continue敘述



- 對於迴圈想要擁有更高的**控制性**
- 利用**break**在任何時候**跳出迴圈**
- 利用**continue**在任何時候**略過迴圈** (略過本次迴圈剩餘的運算)，而這些時機點通常**搭配選擇結構**來進行




# for/while的使用時機

- ▶ 當需要重複進行運算的時候使用迴圈(for/while)
- ▶ 當重複的次數可以清楚被計算或當迭代的表現明顯時使用for迴圈
- ▶ 當重複的次數難以計算(但條件清楚)或是有條件的重複時使用while 

# 額外的else敘述

- 重複結構while和for都支援額外的else敘述, 其語法如下

```
while Condition :   
    while_suite  
else:  
    else_suite
```

```
for Var in Condition :  
    for_suite  
else:  
    else_suite
```

- 當while迴圈或for迴圈不是因為break, return或例外終止時(指迴圈正常中止), else\_suite會被執行

# 巢狀結構

- ▶ 不論是if/elif/else結構,while迴圈或for迴圈都支援巢狀(層疊式)的撰寫,各層之間的縮排務必清楚,冒號也要記得加上
- ▶ 雙重迴圈(多重迴圈)是程式中重要的結構,是處理多層(多軌)迭代或是運算的必要手段
- ▶ 多重迴圈的運行次序可回到基本定義上想



# homework 2

上傳連結：

<https://goo.gl/zQT47x>

- ▶ 分別用for，while迴圈各寫一個 $n \times n$ 的乘法表 程式  
可以讀取使用者輸入的值  $n$ ,  $n > 1$
- ▶ 輸出樣式: ( $n=3$ )

```
1 * 1 = 1
1 * 2 = 2
1 * 3 = 3
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
>>>
```

