3.2 判斷式

在日常生活中，我們經常會遇到一些需要做決策的情況，然後再依決策結果進行不同的事件，例如：暑假到了，如果所有學科都及格的話，媽媽就提供經費讓自己與朋友出國旅遊；如果有某些科目當掉，暑假就要到校重修了！程式設計也一樣，常會依不同情況進行不同處理方式，這就是「判斷式」。

3.2.1 程式流程控制

程式的執行方式有循序式及跳躍式兩種，循序式是程式碼由上往下依序一列一列的執行，到目前為止的範例都是這種模式。程式設計也和日常生活雷同，常會遇到一些需要做決策的情況，再依決策結果執行不同的程式碼，這種方式就是跳躍式執行。

Python 流程控制命令分為兩大類：

- **判斷式**：根據關係運算或邏輯運算的條件式來判斷程式執行的流程，若條件式結果為 True，就執行跳躍。判斷式命令只有一個：

```
if…elif…else
```

- **迴圈**：根據關係運算或邏輯運算條件式的結果為 True 或 False 來判斷，以決定是否重複執行指定的程式。迴圈指令包括下列兩種：（迴圈將在第 4 章詳細說明）

```
for
while
```

3.2.2 單向判斷式（if…）

「if…」為單向判斷式，是 if 指令中最簡單的型態，語法為：

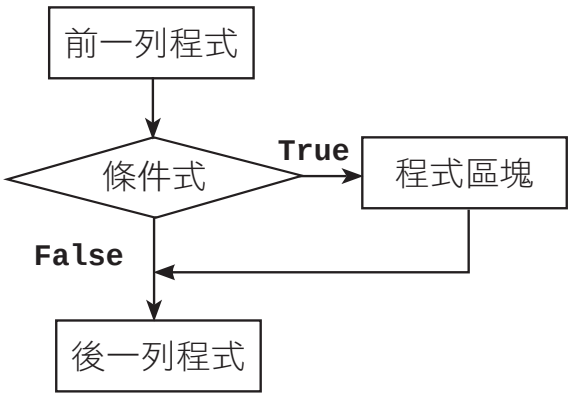
```
if 條件式：
    程式區塊
```

「條件式」允許加上括號，即「if (條件式):」。當條件式為 True 時，就會執行程式區塊的敘述；當條件式為 False 時，則不會執行程式區塊的敘述。

條件式可以是關係運算式，例如：「x>2」；也可以是邏輯運算式，例如：「x>2 or x<5」，如果程式區塊只有一列程式碼，也可以將兩列合併為一列，直接寫成：

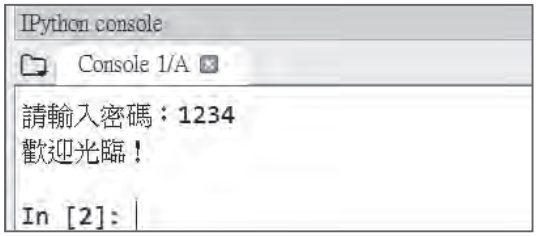
```
if 條件式 : 程式碼
```

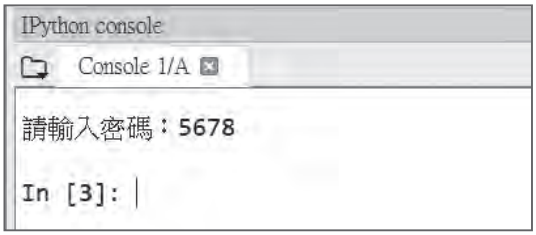
以下是單向判斷式流程控制的流程圖：



範例實作：密碼輸入判斷

小杰設計了一個通關密碼的程式，訪客必須輸入正確密碼才能登入，如果輸入的密碼正確（1234），會顯示「歡迎光臨！」；如果輸入的密碼錯誤，則不會顯示任何訊息。(<password1.py>)





程式碼：ch03\password1.py

```
1 pw = input("請輸入密碼:")
2 if pw=="1234":
3     print("歡迎光臨!")
```

程式說明

- 2-3　預設的密碼為「1234」，若輸入的密碼正確，就執行第 3 列程式列印「歡迎光臨！」訊息；若輸入的密碼錯誤就結束程式。
- 3　if 條件成立的程式區塊，必須以 Tab 鍵或空白鍵向右縮排，本例是以 4 個空白鍵做縮排。

因為此處 if 程式區塊的程式碼只有一列，所以第 2-3 列可改寫為：

```
if pw=="1234" : print("歡迎光臨！")
```

3.2.3 雙向判斷式（if…else）

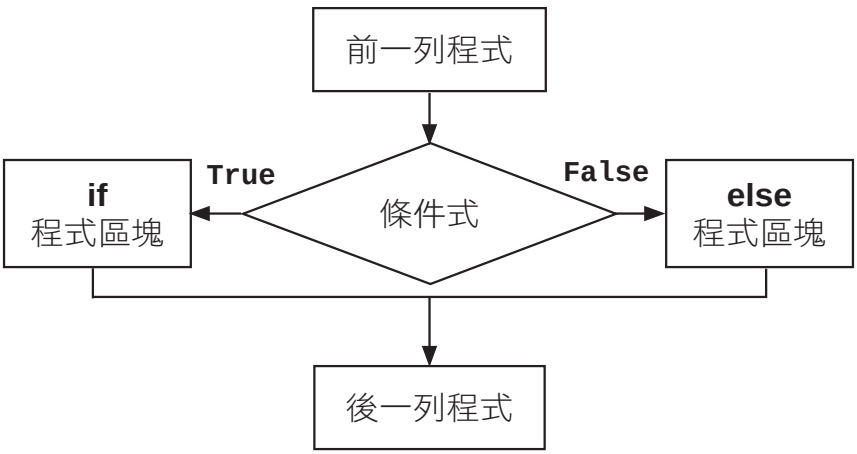
感覺上「if」語法並不完整，因為如果條件式成立就執行程式區塊內的內容，如果條件式不成立也應該做某些事來告知使用者。例如密碼驗證時，若密碼錯誤應顯示訊息告知使用者，此時就可使用「if…else…」雙向判斷式。

「if…else…」為雙向判斷式，語法為：

```
if 條件式：
    程式區塊一
else:
    程式區塊二
```

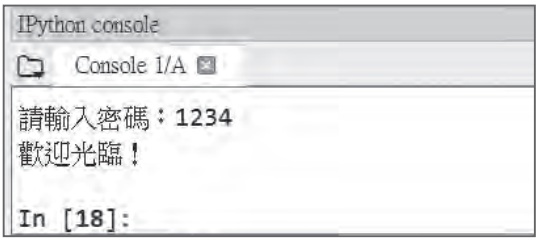
當條件式為 True 時，會執行 if 後的程式區塊一；當條件式為 False 時，會執行 else 後的程式區塊二，程式區塊中可以是一列或多列程式碼，如果程式區塊中的程式碼只有一列，可以合併為一列。

以下是雙向判斷式流程控制的流程圖：



範例實作：進階密碼判斷

小杰程式設計的功力進步許多，現在他改進了通關密碼程式，如果訪客輸入的密碼正確（1234），會顯示「歡迎光臨！」；如果訪客輸入的密碼錯誤，則會顯示「密碼錯誤！」。(<password2.py>)



程式碼：ch03\password2.py

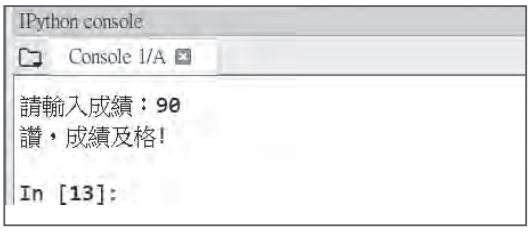
```
1  pw = input("請輸入密碼:")
2  if pw=="1234":
3      print("歡迎光臨!")
4  else:
5      print("密碼錯誤!")
```

程式說明

- 2-3　若輸入的密碼正確，就執行第 3 列程式，顯示歡迎訊息。
- 4-5　若輸入的密碼錯誤，就執行第 5 列程式，顯示密碼錯誤訊息。注意第 4 列要由開頭處輸入「else:」。

延伸練習

資訊小楷模阿梅幫老師設計一個程式，讓老師輸入學生的成績，若學生成績大於等於 60 分，顯示「讚，成績及格!」，否則顯示「成績不及格，加油喔!」。(<score.py>)



8.2 排序

將一串列的值由小至大或由大至小排列，稱為排序。在程式設計中常需要對資料進行排序，例如要計算學生成績的名次時，可先將成績總分由大到小排序，再於排序後的資料填入 1、2 、3、……等名次即可。

8.2.1 泡沫排序

排序的方法有很多種，泡沫排序是最簡單且最常用的排序方法，其原理為逐一比較兩個資料，如果符合指定的排序原則，就將兩個資料對調，如此反覆操作，就可完成排序工作。

泡沫排序的演算法

氣泡排序以由小到大排序為例，演算法的運作如下：

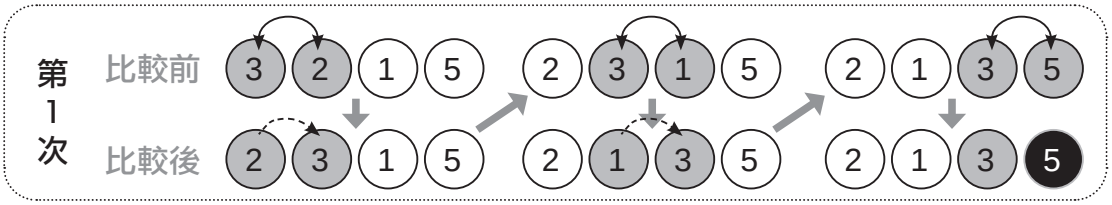
1. 比較相鄰的元素，如果第一個比第二個大，就將兩元素進行交換。
2. 對每一對相鄰元素作同樣的工作，從開始第一對到結尾的最後一對。這一步做完後，最後的元素就是最大的數。
3. 針對所有的元素重複以上的步驟，除了最後一個。
4. 持續每次對越來越少的元素重複上面的步驟，直到沒有任何一對數字需要比較。

由於它的簡潔，氣泡排序通常用來作為程式設計入門學生介紹演算法的概念。

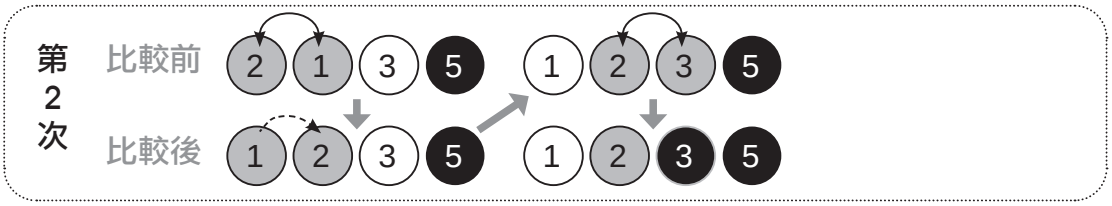
泡沫排序的運作實例

例如有一序列的值是 3,5,2,1 要由小至大排序，泡沫排序的過程如下 (n 為串列元素個數，本例 n=4)：

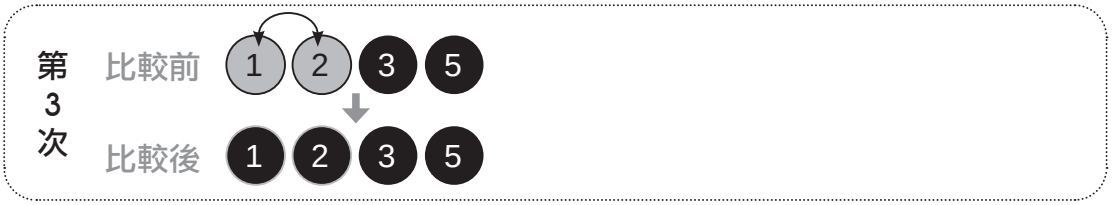
1. 自序列第 1 個數開始至第 3 個數 (n-1) 為止，分別和它右邊的數比較，如果比右邊的數大，則兩數互相交換，也就是將較大數向右移，完成後可看到最大的數 5 已移到最右邊。如下：



2. 現在 5 已是最大數不用再比了，再自序列第 1 個數開始至第 2 個數 (n-2) 為止，分別和它右邊的數比較，如果比右邊的數大，則兩數互相交換，完成後可看到第二大的數 3 已移到最右邊倒數第二的位置。如下：



3. 現在 5、3 已是第 1、2 大的數不用再比了。自序列第 1 個數開始至第 1 個數 (n-3) 為止，和它右邊的數比較，如果比右邊的數大，則兩數互相交換，完成後可看到第三大的數 2 已冒到最右邊倒數第三的位置。如下：

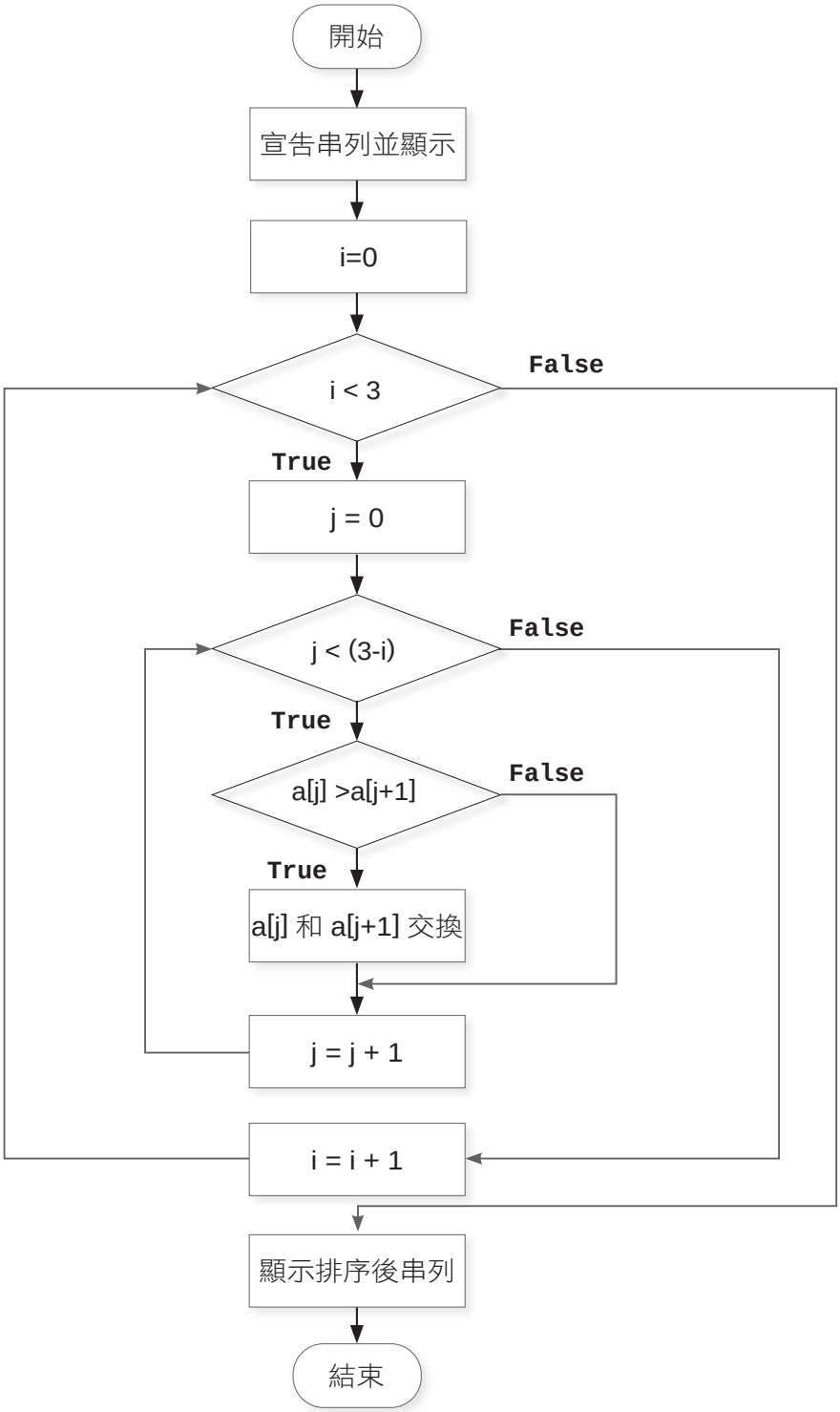


是否感覺排序過程中，較大的數就像泡沫般往後冒出。整理出它的規則為：

1. 將串列索引值第 i 的串列元素分別和第 i+1 的串列元素比較，如果較大，則互相交換，第一次 i 的範圍由 0 到 n-1 (不包含 n-1，即 n 由 0~2)。
2. 相同的操作，將串列索引第 i 的串列元素分別和第 i+1 的串列元素比較，如果比第 i+1 的數大，則互相交換，第二次 i 的範圍由 0 到 n-2 (不包含 n-2，即 n 由 0~1)。
3. 重複的操作，直到全部比完為止。

泡沫排序的流程圖

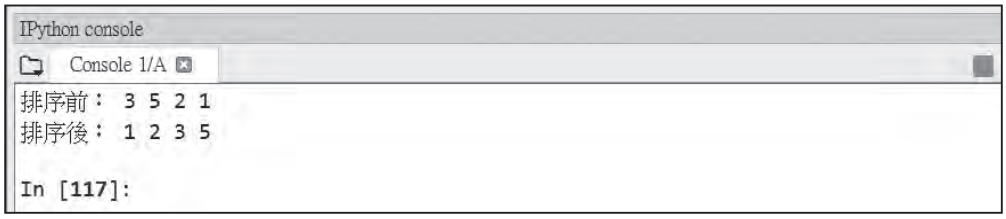
例如有一串列包含 4 個元素，由小至大排序，泡沫排序的流程如下：



▲ 氣泡排序流程圖

範例實作：由小到大排序（泡沫排序）

國隆發現泡沫排序是使用巢狀迴圈後，再利用自訂程序顯示串列，馬上就挑戰 datas=[3,5,2,1] 串列由小到大排序。(<bubble.py>)



程式碼：ch08\bubble.py

```
1   def disp_data(): # 顯示串列的自訂程序
2       for item in datas:
3           print(item,end=" ")
4       print()
5
6   # 主程式
7   datas=[3,5,2,1]
8   print("排序前:",end=" ")
9   disp_data() # 顯示排序前串列
10  n=len(datas)-1 # 串列長度-1
11
12  for i in range(0,n):
13      for j in range(0,n-i):
14          if (datas[j]>datas[j+1]): # 由小到大排序
15              datas[j],datas[j+1]=datas[j+1],datas[j] #兩數互換
16
17  print("排序後:",end=" ")
18  disp_data()  # 顯示排序後串列
```

Appendix

A

ITS 資訊科技專家
國際認證模擬試題

IT Specialist Certification - Python

一、使用資料類型和運算子執行操作

1. 你編寫了以下的程式碼：

對應章節 02

```
list_1 = [1, 2, 3]
list_2 = [4, 5, 6]
list_3 = list_1 + list_2
list_4 = list_3 * 2
print(list_4)
```

執行程式碼的輸出值是？

( )A. [[1, 2, 3], [4, 5, 6], [1, 2, 3], [4, 5, 6]]

( )B. [4, 10, 18]

( )C. [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6]

( )D. [[1, 2, 3, 4, 5, 6], [1, 2, 3, 4, 5, 6]]

2. 你是運動 App 的程式設計師。你必須製作一個函式為跑者計算步速，所謂步速就是每公里所花的時間就是步速。輸出結果必須盡可能精準。要如何完成程式碼？請在回答區中選擇適當的程式碼片段。其中距離轉換為浮點數，分秒的輸入值都要轉換為整數。

對應章節 02

```
# 步速計算器
distance =___(1)___(input("Enter the distance traveled in meters"))
distance_kms = distance / 1000 # convert to kilometers
time_minute =___(2)___(input("Enter the time elapsed in minutes"))
time_sec =____(3)____(input("Enter the time elapsed in seconds"))
time = time_minute*60 + time_sec
pace = time / distance_kms
print("The average velocity is" ,str((pace//60))+ ": str((pace%60)))
```

以上空格分別要填入的函式名為：

( )(1) A. int B. string C. float

( )(2) A. int B. string C. float

( )(3) A. int B. string C. float

6. 對應章節 02

你正在編寫一個計算使用者出生年份的程式。該程式詢問使用者的年齡和當前年份，然後輸出使用者的出生年份。你編寫以下程式碼。其中包含的行號只是做為參考。

```
01 age = input("Enter your age: ")
02 year = input("Enter the four digit year: ")
03 born = eval(year) - eval(age)
04 message = "You were born in " + str(born)
05 print(message)
```

請問下列何者是正確的？

( )A. 在 01 行中 year 的資料類型是 str?

( )B. 在 03 行中 born 的資料類型是 float?

( )C. 在 04 行中 message 的資料類型是 bool?

7. 對應章節 02

在 Python 資料類型的課程中創建以下三個程式碼片段：

```
# Code segment 1
x1 = "5"
y1 = 4
a = x1 * y1

# Code segment 2
x2 = 10
y2 = 4
b = x2 / y2

# Code segment 3
x3 = 5.5
y3 = 1
c = x3 / y3
```

你需要評估程式碼片段。請問下列何者是正確的？(可多選)

( )A. 執行程式碼片段 1 後，變數 a 的資料類型為 str。

( )B. 執行程式碼片段 2 後，變數 b 的資料類型是 float。

( )C. 執行程式碼片段 3 後，變數 c 的資料類型為 int。