

# 序

Python 程式語言是在西元 1989 年，由創始人吉多范羅蘇姆（Guido van Rossum）所設計，Python 是一種直譯式的電腦程式語言，近幾年受到廣大程式設計師與教育單位的喜愛，擠身十大程式語言排行榜的第 1 名。Python 除了原本功能就相當完備的標準函式庫，能夠完成相關基礎程式設計需求外，還能夠整合第三方函式庫套件，提升不同類型應用程式的開發效率，其主要特色包括：免費且開源、語法結構簡單易學、移植性較高以及豐富的第三方套件。

本書一共分為 12 章，在章節安排上由淺而深，以循序漸進的方式來介紹 Python 程式語言最核心的知識，是一本相當適合教學或自學的書籍。另外，本書特別規劃第 0 章，幫助讀者瞭解運算思維與電腦解題的概念。

筆者從事資訊教育工作已經多年，在教學的過程中，發現很多學生學習程式設計時，往往失敗在基礎的觀念沒有學好，或是對於程式的運作流程無法掌握，進而發生學習狀況，因此本書規劃了超過百題的實用 Python 程式範例，有效提升學習樂趣並降低學習障礙，讓讀者們可以不斷地實際思考與練習，以「做中學」的方式來學習 Python 程式語言。

本書的程式範例架構明確，將程式範例分為「程式設計目標」、「參考程式碼」和「程式碼解說」等三個部分，希望讀者先從程式設計目標開始瞭解題目要求，自行思考並設計解題策略，如果遇到困難，再參考本書的程式碼，最後，可以從程式碼解說部分得到詳細的說明。

本書特別融入國際知名專業證照認證機構 Certiport 的資訊科技 Python 專家認證（IT Specialist Certification, 簡稱 ITS）考試重點，考試的重點在官方網站有介紹（[https://certiport.filecamp.com/s/ITS\\_OD\\_303\\_Python.pdf/fi](https://certiport.filecamp.com/s/ITS_OD_303_Python.pdf/fi)），讀者只要確實理解本書內容，通過 ITS Python 認證考試，取得國際證照的機會將會大為增加。最後，特別感謝碁峰的伙伴們，對於本書的出版，奉獻無比心力，使得本書更加完善 ^^

敬祝大家

健康快樂！幸福滿滿！

李啟龍

謹識

# 運算思維與電腦解題

# 0

## CHAPTER

電腦是人類進行問題解決的好幫手，由於電腦具有運算速度快、容量大、計算精確、可以處理大量資料、重複作業…等特性，相當適合幫助人類來解決各種問題，只要規劃出正確的解題方法，就可以透過電腦的輔助來處理。

在我們的日常生活中，處處可見運用電腦來解決實際的生活問題，小至個人的通訊活動或資料處理，大至國家實驗室或企業的儀器設備，都需要使用電腦來達成各項操作。

基本上，電腦不像人類會自主思考解決問題，但如果我們能以電腦處理問題的方式，給予電腦正確的指令，那電腦就能按照我們的指示來處理問題。運算思維（Computational Thinking）就是指能構思一個有條理的程序，應用各種運算方法或工具來解決問題的思維能力。

## 0-1 運算思維

---

運算思維能力是指每個人除了聽、說、讀、寫等基本素養外，亦應具備之基本能力，此能力並非專屬於電腦科學家，而是人人在資訊時代所需要的能力；運算思維就是利用歸納、嵌入、轉換或模擬等方法，將複雜問題轉為我們所熟悉之模式，以利問題的解決。

因此，運算思維具有以下特性：

- 是種基本的素養，並非死記硬背之技能
- 非指撰寫電腦程式
- 是種人類解決問題之方法或策略
- 結合數學及工程之思維
- 是種概念或構想，並非指相關作品
- 適用於每個人與每個地方，人人都需具備的能力

美國的電腦科學教師協會（Computer Science Teacher Association, CSTA）將運算思維定義為電腦可執行之問題解決策略，並包含資料蒐集、資料分析、資料表示、問題分解、抽象化、演算法與程序、自動化、模擬及並行化等概念。所提出之電腦科學核心能力指標中，將運算思維視為貫穿整個課程的重要理念，透過運算思維，以期能培養學生解決問題、設計系統、創造新知識及瞭解現今社會中資訊科技的能力與限制，該協會之網站網址為：<https://www.csteachers.org/>。



Google 於西元 2010 年推出 Exploring Computational Thinking 網站，其網址為：<https://edu.google.com/resources/programs/exploring-computational-thinking/>，網站上收集了許多的教材與資源，提供世人參考與利用。Google 認為具體的運

算思維技能應包含分解問題、模式識別、模式一般化與抽象化、演算法設計及資料分析與視覺化等。

## 0-2 電腦解題的特性

電腦解題的特性就是會依我們設計的步驟，循序漸進的執行，每次執行都會獲得一致的結果。由於垂直式思考的邏輯推理結論較具正確性、系統性及普遍性，大都能轉換成可以循序漸進執行的步驟，來解決各項問題。

當我們要解決的問題比較複雜或是龐大時，可以採取循序漸進解決問題的方式，將大問題分成幾個較小的問題，擬定小問題的解決方案，循序漸進的執行解題方案。

在我們的日常生活中，有許多應用循序漸進流程來設計的例子，例如：烹飪食物的食譜就要求使用者，依照一定的步驟來料理食物；進行網路購物時，也需要循序漸進的完成各項購物程序，先選擇商品、填寫資料後完成付款如圖所示；還有使用自動提款機進行交易時，也需要循序輸入密碼、選擇金融交易方式與輸入相關金額。



循序漸進的網路購物流程

循序漸進的流程就是會依循一定次序的執行步驟，逐步完成各個步驟，最後獲得可預期的結果。而我們規劃出的解題步驟，由於過程明確，次序清楚，非常適合運用電腦來解決問題。

## 0-3 電腦解題之應用

電腦解題應用的領域相當廣泛，只要是電腦所提供的服務，背後都可以觀察到電腦解題的過程。常見的電腦解題在各領域上之應用實例，包括：網路購物系統、電子商務系統、搜尋引擎系統、醫學工程系統、氣象預測系統、校務行政系統、電子地圖應用、各種數學問題解決…等，以下以生活中的電子地圖規劃路線例子，說明隱藏在系統背後的電腦解題應用。

電子地圖之規劃路線功能，就是電腦解題的一個應用，電腦會根據使用者輸入的起點與終點位置，來規劃可行的路線，並且讓使用者還可以選擇交通方式，包括：自行開車、乘坐大眾運輸工具、步行等方式。此處是以總統府為起點，台北 101 大樓為終點，並且選擇自行開車的方式，來測試電子地圖之規劃路線功能，如圖所示。



Google 地圖規劃出來的路線如圖所示，它會在地圖上標示出路線，我們在使用時，可以放大或縮小顯示比例，以方便我們看清楚交通路線。



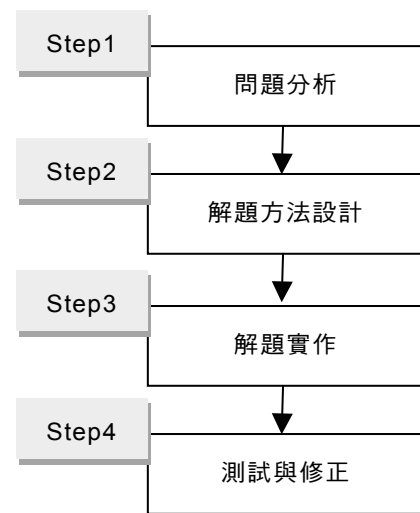
除了顯示出路線路徑之外，知道路名與路線距離也是非常重要的資訊，Google 地圖會把規劃出來的路線，詳細地顯示出這些資訊，包括：會經過哪些路線、每段路線的距離…等，如圖所示。

Google 地圖根據使用者的輸入資料，幫使用者規劃建議路線，解決從起點到終點的路線問題，是一個用電腦來解決生活中問題的例子。

開車到 台北101 的路線指示	
☰ 建議路線	
台5線 7.2 公里	18 分鐘
台5線和仁愛路四段 7.2 公里	21 分鐘
 總統府 100台北市中正區重慶南路一段122號	
1. 朝南，走重慶南路一段，往凱達格蘭大道前進	3 公尺
2. 於第一個路口轉左，走凱達格蘭大道	450 公尺
3. 於中山南路/台9線口向右轉	27 公尺
4. 微靠左行後，繼續在中山南路/台9線前進	87 公尺
5. 向右轉後，繼續走中山南路/台9線	150 公尺
6. 叉路處繼續靠左	450 公尺
7. 微靠右行後，繼續在中山南路/台9線上前進	150 公尺
8. 於忠孝東路一段/台5線口向右轉	4.5 公里

## 0-4 電腦解題程序

使用電腦來解題，其程序大致可以分為：問題分析、解題方法設計、解題實作、測試與修正等四步驟，電腦解題程序如圖所示。



電腦解題程序

### 一、問題分析階段

使用電腦來解題之前，需要先對問題進行分析，問題分析是一種思考過程，我們對於要解決的問題，需要從不同的角度來思考問題，確定問題的定義及範疇。在問題分析的過程中，需要先釐清「輸入規範」、「輸出規範」及「輸入與輸出對應關係」等要素。

若我們要用 `print()` 函式來顯示 100 次 Hello，一一列的寫，將會需要 100 列的「`print(' Hello')`」程式敘述，這樣的程式實在太過繁雜且撰寫耗時，幸好 Python 語言提供了迴圈（Loop），可以簡化重複動作的撰寫，只要幾行的程式碼，就可顯示 100 次的 Hello。

迴圈的重複結構使得程式語言更具威力，且善用了電腦的好處，可以不厭其煩的重複執行特定程式敘述，以完成指定的動作。迴圈就像是一條圓型的道路，從原點開始走，走一圈會回到原點，當回到原點時，可以依條件選擇要不要繼續走，或是完成指定的重複次數。

## 5-1 迴圈結構之 for 敘述

---

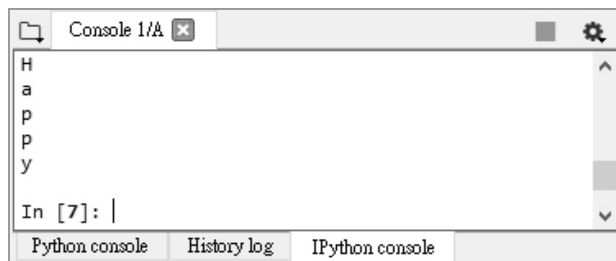
`for` 迴圈會依序走訪序列（Sequence）內的元素（Item），直到序列結束為止，其基本語法如下：

```
for 變數名稱 in 序列:  
    for 的程式區塊
```

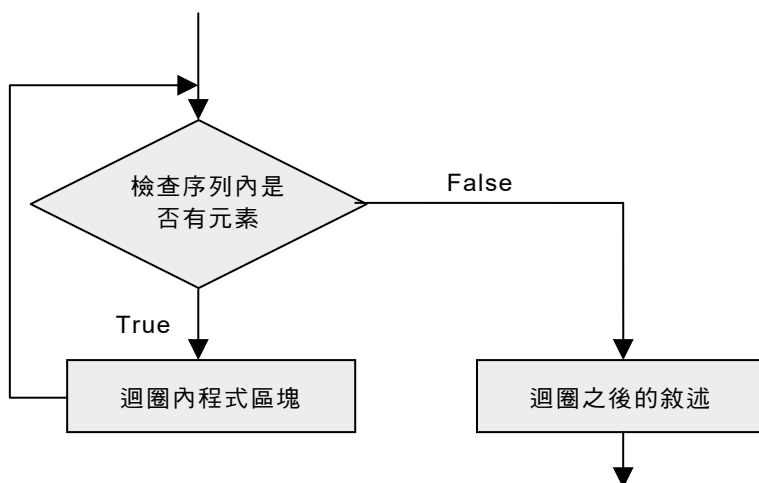
請參考程式範例：

```
word = 'Happy'  
for x in word:  
    print(x)
```

此例中 for 迴圈的序列為字串「Happy」，以變數「x」依序走訪序列內的元素，每次執行迴圈，就印出變數「x」的內容，也就是「H」、「a」、「p」、「p」、「y」，該 for 迴圈會執行 5 次，其輸出結果如下：



for 迴圈的流程圖表示法如下：



迴圈可以重複執行程式敘述，藉由執行次數的控制，可以完成我們需要的運算，更可以設計許多較為複雜的程式，for 迴圈常搭配 range( ) 函式來使用，其基本語法如下：

```
for 變數名稱 in range(參數):
    for 的程式區塊
```

range( ) 函式主要用來建立整數序列，總共有 3 個參數，其起始值與增減值為非必備參數，使用格式如下：

```
range([起始值], 終止值[, 增減值])
```

- 起始值：此為非必備參數，其預設值為 0。
- 終止值：此為必備參數。



- 增減值：此為非必備參數，其預設值為 1。

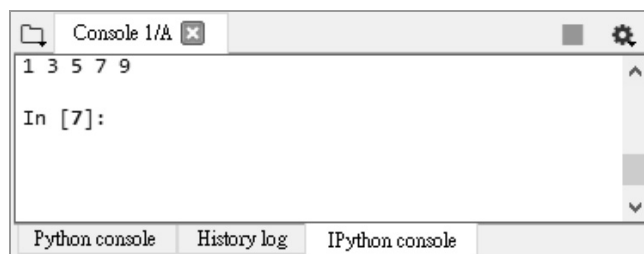
請參考 `range()` 函式的範例：

- `range(6)`：起始值參數為空，其預設值為 0；終止值為 6；其增減值參數為空，預設值為 1，故其 `range()` 範圍為索引值 0、1、2、3、4、5 共 6 個元素，索引值 6 不包括在內。
- `range(1, 11)`：起始值參數為 1；終止值為 11；其增減值參數為空，預設值為 1，故其 `range()` 範圍為索引值 1、2、3、4、5、6、7、8、9、10 共 10 個元素，索引值 11 不包括在內。
- `range(3, 10, 2)`：起始值參數為 3；終止值為 10；其增減值為 2，故其 `range()` 範圍為索引值 3、5、7、9 共 4 個元素，索引值 10 不包括在內。

請參考 `for` 迴圈搭配 `range()` 函式的程式範例：

```
for x in range(1, 11, 2): #以增值 2 來產生整數序列
    print(x, end=' ')
```

`for` 迴圈搭配使用 `range()` 函式，起始值為 1，終止值為 11，增減值為 2，輸出時每個項目之間間隔一個空格，`for` 迴圈會執行 5 次，其輸出結果如下：



```
Console 1/A x [Settings]
1 3 5 7 9
In [7]:
Python console History log IPython console
```

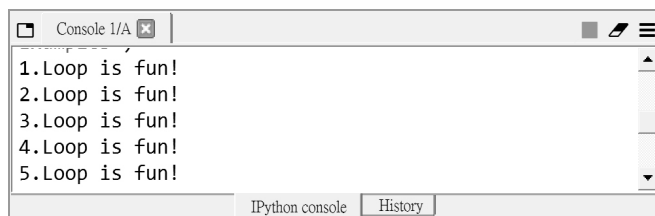
## 程式範例：連續印出字串程式

📄 參考檔案：5-1-1.py

📌 學習重點：熟悉 `for` 迴圈的使用

### 一、程式設計目標

運用 `for` 迴圈，寫出一個程式，連續印出 5 次「Loop is fun!」，並在輸出內容的前方加上序號，執行結果如圖所示。



```

1. Loop is fun!
2. Loop is fun!
3. Loop is fun!
4. Loop is fun!
5. Loop is fun!

```

## 二、參考程式碼

列數	程式碼
1	<code># 連續印出字串的範例</code>
2	<code>for x in range(5):</code>
3	<code>    print('%d.Loop is fun!' % (x+1))</code>

## 三、程式碼解說

- 第 2~3 行：for 迴圈搭配 `range()` 函式，設定終止值為 5，因此索引值範圍為 0~4，此處將迴圈變數值加 1，以利序號的計數。for 迴圈共執行 5 次，因此 `print()` 函式會列印 5 次「Loop is fun!」字串。

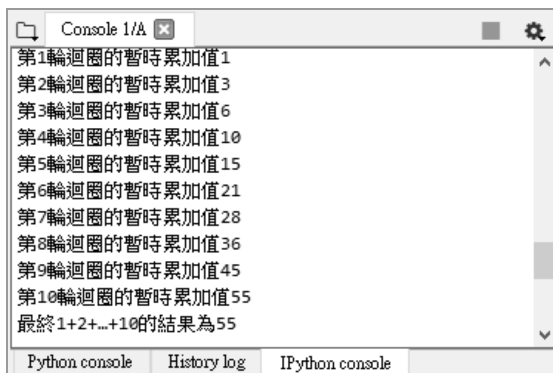
### 程式範例：累加程式 $1+2+\dots+10$

📄 參考檔案：5-1-2.py

📌 學習重點：熟悉 for 迴圈的使用

## 一、程式設計目標

運用 for 迴圈，寫出一個程式，計算  $1+2+\dots+10$  的結果，其中間的累加過程會一併顯示出來，其執行結果如下圖所示。



```

第1輪迴圈的暫時累加值1
第2輪迴圈的暫時累加值3
第3輪迴圈的暫時累加值6
第4輪迴圈的暫時累加值10
第5輪迴圈的暫時累加值15
第6輪迴圈的暫時累加值21
第7輪迴圈的暫時累加值28
第8輪迴圈的暫時累加值36
第9輪迴圈的暫時累加值45
第10輪迴圈的暫時累加值55
最終1+2+...+10的結果為55

```

## 二、參考程式碼

列數	程式碼
1	# 累加程式
2	Sum = 0 # 將變數 Sum 的初值設為 0
3	for i in range(1, 11):
4	Sum = Sum + i # 將 Sum 的值再加上 i 的值
5	print('第%d 輪迴圈的暫時累加值%d' % (i, Sum))
6	print('最終1+2+...+10 的結果為%d' % (Sum)) # 印出 Sum 的值

## 三、程式碼解說

- 第 2 行：宣告變數 Sum，並將變數 Sum 的初值設為 0。
- 第 3~5 行：for 迴圈的起始值設為 1，終止值為 11，其索引值為 1~10，使用「Sum = Sum + i」敘述來累加各個索引值。
- 第 6 行：使用 print() 函式印出 1+2+...+10 的最終計算結果。



### TIPs for...else...迴圈

for 迴圈有時會搭配 else 敘述，其語法如下：

```
for 變數名稱 in 序列:
    for 的程式區塊
else:
    else 的程式區塊
```

當程式流程離開 for 迴圈時，會執行 else 部分的程式區塊，請參考 for 迴圈搭配 else 敘述的程式範例：

```
for i in range(1, 6):
    print(i, end=',')
else:
    print('for 迴圈結束！')
```

for 迴圈搭配使用 else 敘述，起始值為 1，終止值為 6，增減值為 1，輸出時每個項目之間以逗號「,」間隔，for 迴圈會執行 5 次（1、2、3、4、5），最後印出 else 部分的「for 迴圈結束！」，其輸出結果如下：

```
Console 1/A
1,2,3,4,5,for迴圈結束!
In [19]:
Python console History log IPython console
```

## 5-2 迴圈結構之 while 敘述

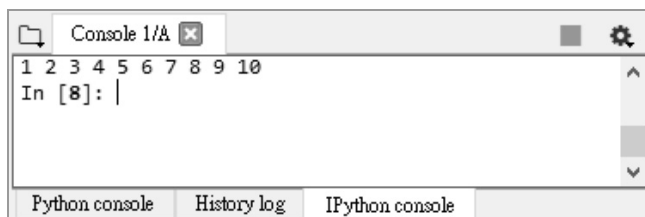
`while` 迴圈的結構與 `for` 迴圈相似，`while` 迴圈會先檢查條件式是否成立，條件式成立，則進入 `while` 迴圈的程式區塊，如條件式不成立，則離開 `while` 迴圈，其基本語法如下：

```
while(條件式):  
    while 迴圈程式區塊
```

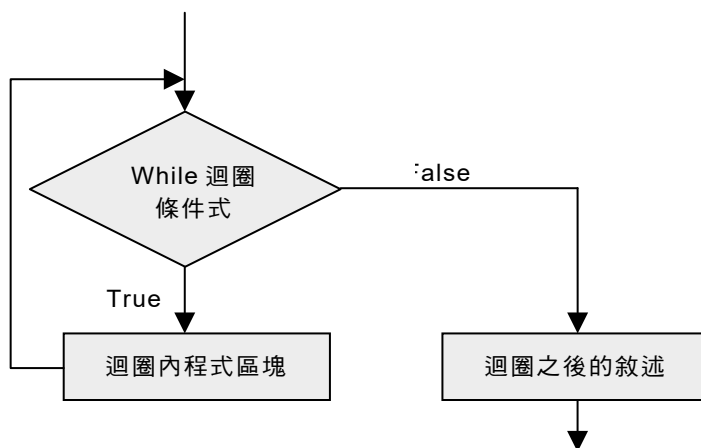
請參考程式範例：

```
i = 1  
while(i <= 10):  
    print(i, end=' ')  
    i += 1
```

`while` 迴圈的條件式為「`i<=10`」，當 `i` 值小於等於 10 時，條件式成立，故會持續在迴圈內執行；當 `i` 值超過 10 時，條件式不成立，故會離開迴圈，因此該程式會依序印出變數 `i` 從 1 到 10 的變化，其輸出結果如下：



`while` 迴圈的流程圖表示法如下：



運用 `while` 迴圈時要注意迴圈的跳出條件，萬一在條件設定上有問題，可能會形成無窮迴圈，造成程式不斷執行迴圈裡的程式區塊。

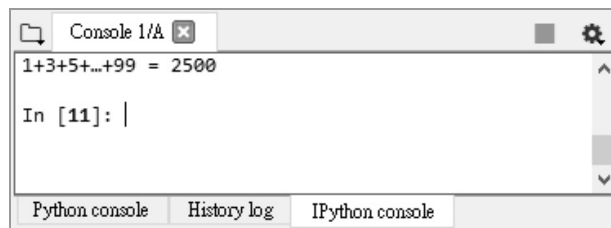
## 程式範例：累加程式 $1+3+5+\dots+99$

📄 參考檔案：5-2-1.py

📌 學習重點：熟悉 `while` 迴圈的使用

### 一、程式設計目標

運用 `while` 迴圈，寫出一個程式，計算級數  $1+3+5+\dots+99$  的結果，執行結果如下圖所示。



### 二、參考程式碼

列數	程式碼
1	<code># 1+3+5+...+99 累加程式</code>
2	<code>i = 1</code>
3	<code>Sum = 0</code>
4	<code>while(i &lt;= 99):</code>
5	<code>    Sum += i</code>
6	<code>    i += 2</code>
7	<code>print('1+3+5+...+99 = %d' % (Sum))</code>

### 三、程式碼解說

- 第 2、3 行：設定變數 `i` 的初值為 1（用於計算間隔為 2 的級數），設定變數 `Sum` 的初值為 0（用於計算累加總和）。
- 第 4~6 行：`while` 迴圈的條件式為「`i<=99`」，當 `i` 值小於等於 99 時，條件式成立，故會持續在迴圈內執行；當 `i` 值超過 99 時，條件式不成立，故會離開迴圈。`while` 迴圈的程式區塊為「`Sum+=i`」敘述和「`i+=2`」敘述。

述，所以， $i$  值的變化為 1、3、5、7...99，當  $i$  等於 101 的時候便不符合條件式的要求，因此只會加總 1、3、5、7...99 等數值。

- 第 7 行：印出加總後 Sum 的值，得到  $1+3+5+\dots+99$  的結果。

## 程式範例：捐款累加程式

📄 參考檔案：5-2-2.py

📌 學習重點：熟悉 while 迴圈的使用

### 一、程式設計目標

運用 while 迴圈，寫出一個程式，顯示使用者每次的捐款次數與金額，最後會顯示總捐款金額，執行結果如下圖所示。

```

Console 1/A
請輸入捐款金額(如要結束計算請按0)：100
存了1次款項，累計：100元

請輸入捐款金額(如要結束計算請按0)：150
存了2次款項，累計：250元

請輸入捐款金額(如要結束計算請按0)：0
總捐款金額合計：250元

In [21]: |
Python console History log IPython console
  
```

### 二、參考程式碼

列數	程式碼
1	<code># 捐款累加程式</code>
2	<code>i = 1</code>
3	<code>Sum = 0</code>
4	<code>money = int(input('請輸入捐款金額(如要結束計算請按0)：'))</code>
5	<code>while (money != 0):</code>
6	<code>    Sum += money</code>
7	<code>    print('存了%d 次款項，累計：%d 元' % (i, Sum))</code>
8	<code>    i += 1</code>
9	<code>    money = int(input('請輸入捐款金額(如要結束計算請按0)：'))</code>
10	<code>print('總捐款金額合計：%d 元' % (Sum))</code>

### 三、程式碼解說

- 第 2、3 行：設定變數 `i` 的初值為 1（用於計算捐款次數），設定變數 `Sum` 的初值為 0（用於計算捐款總金額）。
- 第 4 行：使用 `input()` 函式讀入使用者輸入的每次捐款金額，並且轉成整數型態後存入變數 `money`。
- 第 5~9 行：`while` 迴圈的條件式為「`money!=0`」，當 `money` 值不為「0」的時候，條件式成立，會進入 `while` 迴圈做計算；當 `money` 值為「0」時，條件式不成立，故會離開迴圈。
- 第 10 行：印出加總後 `Sum` 的值，得到總捐款金額的計算結果。



#### TIPS while...else...迴圈

`while` 迴圈有時會搭配 `else` 敘述，其語法如下：

```
while(條件式):
    while 迴圈程式區塊
else:
    while 條件不滿足時執行的程式區塊
```

當程式流程離開 `while` 迴圈時，會執行 `else` 部分的程式區塊，請參考 `while` 迴圈搭配 `else` 敘述的程式範例：

```
i = 6
while(i < 6):
    print(i, end=',')
    i += 1
else:
    print('while 迴圈結束!')
```

變數 `i` 的初始值為「6」，`while` 迴圈不滿足，所以直接印出 `else` 部分的「`while 迴圈結束!`」，其輸出結果如下：

```
Console 1/A
while迴圈結束!


In [25]:

Python console | History log | IPython console
```

### 三、程式碼解說

- 第 3 行：外迴圈，控制程式總共印幾列。
- 第 4 行：內迴圈，設計每一列的輸出，根據題目要求使用迴圈控制，第一次印出一個「\*」，第二次印出兩個「\*」…，如此就達到題目要求了。
- 第 6 行：每執行完畢內迴圈一次，就使用 `print()` 函式來換行。

#### 程式範例：印出星形圖樣 2

 參考檔案：5-5-3.py

 學習重點：巢狀 for 迴圈的使用

#### 一、程式設計目標

請運用巢狀 for 迴圈配合 `print()` 函式，印出如右圖排列的星形圖樣，可以讓使用者輸入要列印的列數，每一列的星星個數與列數相同。



#### 二、參考程式碼

列數	程式碼
1	<code># 印出星形圖樣2</code>
2	<code>line = int(input('請輸入要列印的星星列數：'))</code>
3	<code>for i in range(1, line+1):</code>
4	<code>    for j in range(line, i, -1):</code>
5	<code>        print(' ', end='') # 印出空白</code>
6	<code>    for k in range(1, i+1):</code>
7	<code>        print('*', end='') # 印出星號</code>
8	<code>    print() # 換印下一列</code>

### 三、程式碼解說

- 第 3 行：外迴圈，控制程式總共印幾列。
- 第 4 行：內迴圈，設計每一列輸出的空格個數。
- 第 6 行：內迴圈，設計每一列輸出的星號個數。
- 第 8 行：每執行完畢 2 個內迴圈一次，就使用 `print()` 函式來換行。



## 5-6 程式練習

### 練習題 1：印出兩數之間的所有質數

📄 參考檔案：5-6-1.py

📌 學習重點：巢狀 for 迴圈的使用

#### 一、程式設計目標

找出兩數之間的所有質數，並輸出至螢幕，下圖為起始數字「5」和結尾數字「88」的執行結果。

```

Console 1/A
請輸入起始數字：5
請輸入結尾數字：88
5是質數, 7是質數, 11是質數, 13是質數, 17是質數, 19是質數, 23是質數,
29是質數, 31是質數, 37是質數, 41是質數, 43是質數, 47是質數, 53是質數,
59是質數, 61是質數, 67是質數, 71是質數, 73是質數, 79是質數, 83是質數,

In [14]:
Python console | History log | IPython console
  
```

#### 二、參考程式碼

列數	程式碼
1	<code># 印出兩數之間的所有質數</code>
2	<code>p1 = int(input('請輸入起始數字：'))</code>
3	<code>p2 = int(input('請輸入結尾數字：'))</code>
4	<code>for i in range(p1, p2+1):</code>
5	<code>    flag = 1</code>
6	<code>    for j in range(2, i):</code>
7	<code>        if(not(i % j)):</code>
8	<code>            flag = 0</code>
9	<code>    if(flag):</code>
10	<code>        print('%2d 是質數' % (i), end=',')</code>

### 三、程式碼解說

- 第 4~10 行：利用兩層 for 迴圈配合餘數運算「%」，來檢查是否為質數。如果有因數，則表示不是質數，flag 值會被設為 0；如果沒有因數，則表示是質數，flag 值會被設為 1，並且印出質數的值。

### 練習題 2：累加程式 $1+2+4+7+11+\dots+106$

參考檔案：5-6-2.py

學習重點：while 迴圈的使用

#### 一、程式設計目標

寫出一個程式，計算級數  $1+2+4+7+11+\dots+106$  的過程與結果，執行結果如右圖所示。

```

i=1 Sum=1
i=2 Sum=3
i=4 Sum=7
i=7 Sum=14
i=11 Sum=25
i=16 Sum=41
i=22 Sum=63
i=29 Sum=92
i=37 Sum=129
i=46 Sum=175
i=56 Sum=231
i=67 Sum=298
i=79 Sum=377
i=92 Sum=469
i=106 Sum=575
Sum=575

In [16]:

```

#### 二、參考程式碼

列數	程式碼
1	<code># 非等距累加程式</code>
2	<code>Sum = 0</code>
3	<code>i = 1</code>
4	<code>j = 1</code>
5	<code>while (i &lt;= 106):</code>
6	<code>    Sum += i</code>
7	<code>    print('i=%d Sum=%d' % (i, Sum))</code>
8	<code>    i = i + j</code>
9	<code>    j = j + 1</code>
10	<code>print("Sum=%d" % (Sum))</code>

#### 三、程式碼解說

累加程式  $1+2+4+7+11+\dots+106$  與之前的範例不同點在於，每一項之間的差距，不是等距，其間距由 1 開始，依次變成 2、3、4...，此處要特別注意。

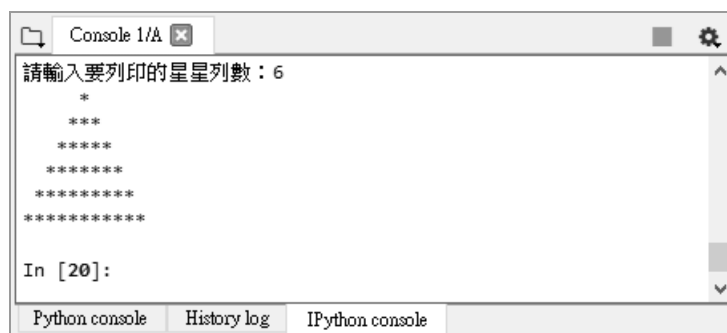
## 練習題 3：印出金字塔星形圖樣

參考檔案：5-6-3.py

學習重點：巢狀 for 迴圈的使用

### 一、程式設計目標

請運用巢狀 for 迴圈配合 `print()` 函式，印出如下圖排列的星形圖樣，可以讓使用者輸入要列印的列數，每一列的星星個數與列數相同。



### 二、參考程式碼

列數	程式碼
1	<code># 印出金字塔星形圖樣</code>
2	<code>line = int(input('請輸入要列印的星星列數：'))</code>
3	<code>for i in range(1, line + 1):</code>
4	<code>    for j in range(line, i, -1):</code>
5	<code>        print(' ', end='') # 印出空白</code>
6	<code>    for k in range(1, 2 * i):</code>
7	<code>        print('*', end='') # 印出星號</code>
8	<code>    print() # 換印下一列</code>

### 三、程式碼解說

- 第 6 行：此題在印出星號的部分有一些變化，每次會增加 2 個星號。



## 習題

### 選擇題

( ) 1. 下列何種敘述會略過接下來的程式碼，然後直接跳到下一輪迴圈的起始位置？

- (a) break      (b) next      (c) for      (d) continue

( ) 2. 執行下列程式後，其輸出內容為何？

```
i = 3
while(i < 6):
    print(i, end=' ')
    i += 1
else:
    print('5')
```

- (a) 3 4 5 5      (b) 3 4 5 6      (c) 1 2 3 4 5      (d) 以上皆非

( ) 3. 下列關於 range( ) 函式的敘述何者錯誤？

- (a) 起始值為非必備參數，其預設值為 1  
 (b) 終止值為必備參數  
 (c) 增減值為非必備參數，其預設值為 1  
 (d) 主要用來建立整數序列

( ) 4. 執行下列程式後，其輸出內容為何？

```
num = 54
while(num % 7 >= 0):
    print('%d' % (num), end=' ')
    if(num % 7 == 0):
        break
    num += 1
else:
    print(num)
```

- (a) 53 54 55      (b) 54 55      (c) 54 55 56      (d) 54

( ) 5. for 迴圈的增減值之預設值為何？

- (a) 1      (b) 0      (c) -1      (d) 無

( ) 6. 下列程式碼會搭配哪一個指令逐一檢查資料？

```
(index < 10):
```

- (a) if      (b) for      (c) while      (d) elif

( ) 14. 請問下列程式碼的輸出結果為何？

```
Sum = i = 0
while(i <= 5):
    Sum += i
    if(i >= 3):
        break
    i += 1
print(Sum)
```

(a) 5                      (b) 6                      (c) 8                      (d) 10

( ) 15. 請問下列程式碼的輸出結果為何？

```
i = 1
while(i <= 3):
    i += 1
    if(i >= 1):
        i += 1
        continue
print(i)
```

(a) 5                      (b) 6                      (c) 7                      (d) 8

## 問答題

1. 請說明迴圈的概念為何？
2. 請說明 for 迴圈的意義、語法與流程圖。