十二年國民基本教育

技術型高級中等學校商業與管理群

程式語言與設計

（上冊）

洪國勝、蔡懷文
蔡懷介、陳蘊慈 編著

泉勝出版有限公司
www.goodbooks.com.tw

編輯大意

一、本書係依據中華民國 107 年 8 月教育部發布之十二年國民基本教育技術型高級中等學校群科課程綱要 - 商業與管理群「程式語言與設計」編輯而成。

二、本書分上、下冊，供技術型高級中等學校商業與管理群第一學年第一學期及第二學期，每周上課二節二學分教學之用。

三、本書選擇 Python 作爲程式語言與設計的工具。

四、因爲學生的起點是國中畢業生，且技高學生 PR 值大多分佈在中後段，我們希望學生從作中學，所以很多地方的執行結果，我們不直接寫出，而是留給學生實作再填入結果，這樣印象才會深刻。例如：

```
a=5;b=4 #指派或稱賦值
print(a/b) #實數除法，得到實數商_________
print(a//b) #整數除法，得到整數商_________
```

五、本書教材範例豐富，所有指令都有範例解說，讓程式不只是程式，而是可以解決日常生活問題的工具。

六、充分的自我練習。我們有充分的自我練習，這樣學生才能充分練習，達到舉一反三的效果。

七、跨領域學習。本書的範例涵蓋數學、英文、資訊科技與生活科技，完全呼應與配合 108 課綱的跨領域教學與學習。

八、本書章節前加註 ※ 者，爲較深入內容，教師可依實際教學進度酌予刪減。

九、本書雖力求嚴謹，但疏漏之處在所難免，尚祈先進惠予指正。

本書學習表現如下：

1. 了解程式語言在生活上的應用與對科技創新的影響。
2. 具備撰寫程式語言基本能力，運用資訊科技方法解決問題。
3. 具備程式設計之邏輯思考能力，展現系統思考、分析與探索之素養。
4. 具備實作程式設計能力，展現程式語言跨域應用之軟實力。
5. 能思辨勞動法令規章與相關議題，省思自我的社會責任。

本書教學注意事項：

1. 本科目爲技能領域實習科目。
2. 教學活動宜引用流程圖作爲程式設計步驟的輔助說明。
3. 教學不宜偏重或強調單一軟體之功能，宜引導學生多認識與使用不同作業系統或雲端平台之相關應用軟體，增加學習之多元性與適應性。

目錄

上冊

Chapter
01

程式語言基本概念

學習綱要

- 1-1 程式語言與程式設計的演算法
- 1-2 程式語言開發環境的操作
- 1-3 物件導向程式設計的認識
- 1-4 程式設計相關議題
- 1-5 本章內容摘要

學習表現

- 1. 了解程式語言在生活上的應用與對科技創新的影響。
- 2 能思辨勞動法令規章與相關議題，省思自我的社會責任。

1-1 程式語言與程式設計的演算法

我們人人手上的手機與電腦之所以讓人愛不釋手，主要是這些設備功能非常豐富，現代人每天生活都已經離不開手機與電腦的程式應用服務之中，例如：

1. 手機與電腦的作業系統與應用程式。隨著資訊科技的進步，現在已經是人人有手機，家家有電腦或筆電，手機與電腦的作業系統與應用程式都是程式的應用。

2. 通訊軟體。隨著網路的普及，在家用手機與電腦，搭配通訊軟體 Facebook、Line 等等，就能得到全世界所有的資料、影片，這些網路傳輸軟體也是程式應用。例如，可以聊天、互傳訊息、看線上球類比賽、網紅直播等。

3. 工業自動化（Industrial Automation）。現代的工業生產線，已經大部分由機器人取代人力，機器人可自動焊接、組裝、運送，機器人與機器人也可自動溝通，整個生產線的人力僅維護這些機器人的運轉，這些機器人的運轉與溝通也都是程式應用。

4. 商業自動化。現在所有的商業進銷存運轉、股票買賣、銀行金流、線上轉帳、網路購物等都是靠程式應用。圖 1-1 是新光銀行的轉帳系統，讓您不用出門就可以辦理查詢交易明細、轉帳等功能。

圖1-1 銀行網路轉帳系統

5. 醫院自動化。現代醫院也不用人工傳送病歷了，所有的檢驗報告、醫囑、處方、收費、掛號等也都是程式應用。圖 1-2 是高雄榮總的網路掛號系統。讓您不用出門就可以完成看診預約。

圖1-2　醫院網路掛號系統

6. 個人穿戴式裝置。隨著智慧型裝置的快速發展，手機將不是被動裝置，日後手機將會協助監控我們的身體健康，發出訊息提醒自己、或直接將訊息傳給醫療單位，作爲協助就醫的依據等，這也是程式應用。圖 1-3 是一些常用個人穿戴式裝置。

圖1-3　是一些常用個人穿戴式裝置。（摘自PCHOME購物網）

7. 物聯網（Internet of Things，簡稱 IoT）時代來臨。現在每個人都有手機，人與人都可溝通，下一步將是物與物直接溝通，冰箱東西吃完了，冰箱會直接叫貨，機器人可自駕，也可直接操作電梯將東西送到我們家；地上髒了、機器人也可以自動打掃等等，這些也都是程式應用。

能讓手機與電腦有如此豐富的功能，表示這些設備有預先安裝應用程式，這些應用程式隨時可依照使用者的需求提供服務，有如孔明的錦囊妙計。而這些應用程式都需要程式語言與程式設計，此即爲本章的重點，以下說明什麼是程式語言、程式設計與程式設計演算法。

程式語言（Programming language）

人與人之間溝通的工具稱爲語言，人類是由不同民族組成，因其發源地不同，所以就有許多語言。例如，華語、英語及德語等。其次，人與電腦溝通的工具，則稱爲電腦程式語言。那麼爲什麼沒有電視語言、冰箱語言或冷氣語言呢？那是因爲這些機器的功能較爲簡單，只要幾個按鈕就能發揮其功能。但是電腦的功能就非常多，能處理所有的數值與字串計算、可以同人類有判斷功能、可以無限重複或依某些條件重複某些事件等等智慧功能，而完成以上銀行轉帳、醫院掛號、股票買賣、穿戴式裝置等應用程式，這些功能多到連用整個鍵盤的所有按鍵都無法表現其功能，所以必須使用一些類似單字所組成的片語與敘述來發揮其所有功能，這些單字與片語的集合就稱爲電腦程式語言，簡稱程式語言。就如同人類也無法用 26 個字母表達所有感受與思維，必須藉助這些字母的排類組合，先組成單字，再由單字組合成片語與句子，才能充分表達其思維。

程式設計（Programming）

使用程式語言命令電腦工作稱爲程式設計。

程式設計的演算法

演算法（Algorithm）是指電腦程式完成一項工作所需要『步驟』的集合。演算法嚴謹定義如下：

在有限 (finite) 的步驟 (step) 所構成的集合中，依照給定輸入 (input) ，依序執行每個明確 (definite) 且有效 (effective) 的步驟，以便能夠解決特定的問題，而且步驟的執行必定會終止 (terminate)，並產生輸出 (output)。

演算法的流程表示

常見的演算法流程表示有三種，分別是自然語言、虛擬碼與流程圖。分別說明如下：

▶ 自然語言（Nature Language）

自然語言就是使用我們日常生活的文字表示或已經熟悉的數學語言。例如，以下是求 9 開根號的自然語言演算法。

1. 使用迴圈從 1 到 9 分別求其平方。例如，1 的平方是 1，2 的平方是 4，3 的平方是 9…，此稱爲循序法。
2. 若其平方大於等於 9，此數即爲所求。例如 3 的平方已經大於等於 9，3 即爲所求。

▶ 虛擬碼（Pseudo Code）

在自然語言中，有時嵌入一些慣用程式敘述，如 if 、switch 或 match 代表決策（第四章介紹）for、while 代表迴圈（第五章介紹），如此可縮短文字長度，也會讓敘述更加清楚易懂。例如，以下是求 9 開根號的虛擬碼演算法：

```
 for i=1 to 9{  #1,2,3,4,5,6,7,8,9
    y=i*i
    if y>=9 then
        print(i)
}
```

▶ 流程圖（Flow Chart）

流程圖是利用各種方塊圖形、線條及箭頭等符號來表達演算流程的順序，常用的流程符號如表 (1-1)。

表1-1　常用流程圖符號表

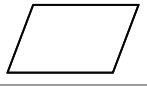
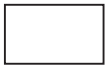
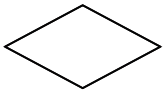
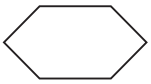
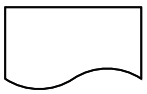
編號	符號	意義
1	開始 結束	起迄符號。 代表流程圖的開始或結束，此符號若是開始，則僅有一出口，若是結束則僅有一入口。
2		輸入與輸出符號。 用來填入輸入與輸出的符號，此符號有一入口與一出口。
3		處理符號。 用來填入處理程序，此符號有一入口與一出口。

表1-1 常用流程圖符號表（續）

編號	符號	意義
4		決策符號。 用來表示決策分歧點，此符號的出口至少有兩個，分別是 true 或 false。
5		重複符號。 聲明重複指令的起點與離開條件，通常配合下面的連結符號表示重複的範圍。
6		連結符號。 可配合上面的重複符號或移至另一頁的起點。
7		副程式。(副程式又稱函式) 用來表示另一個副程式。
8		程式流向符號。 用來連結以上符號，說明程式的流向。
9		代表輸出至螢幕。 將資料由螢幕輸出。
10		代表輸出至印表機。 將資料由印表機輸出。

例如，圖 1-4 是求 9 開根號的流程圖演算法。

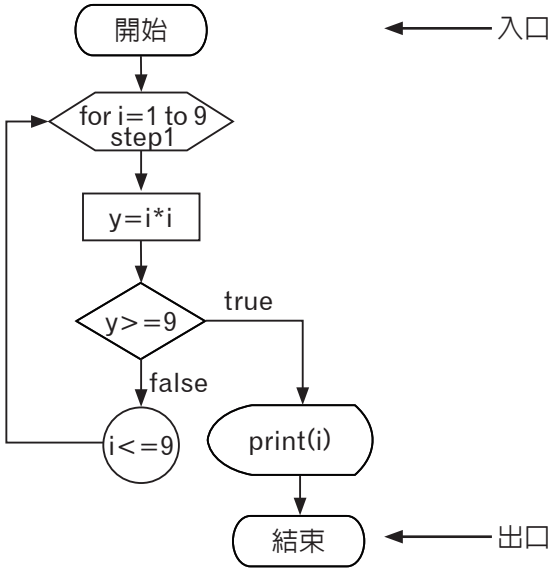


圖1-4 開根號求解流程圖

▶ 演算法比較

使用自然語言文字描述演算法，適合已經有程式設計經驗者，因爲隨著問題的複雜度增加，初學者較難體會精簡的自然語言描述。虛擬碼使用類似程式碼的文字描述演算法，但不能夠直接執行，雖然能快速轉換成程式碼，適合已經有程式基礎的人員，因爲有時過於精簡，初學者也難體會。流程圖使用鉅細靡遺的流程圖符號，雖然較耗時，但最能精準表現程式運算與流程，最適合初學者學習程式設計。

1-2 程式語言開發環境的操作

目前流行的程式語言，依其產出年份排列，有C、C++、Visual Basic、Java、C#、Python等語言。因爲Python是目前最新的物件導向程式設計語言，且其抽象化等級最爲先進、語言規定最少、功能也最進步、性能價值比(C/P值)最高、初學者也最容易入門。所以本書選擇Python作爲技術型高中商管群『程式語言與設計』的語言。

Python 整合開發環境的操作

所謂的整合程式開發環境，是指可以在同一個畫面鍵入程式、存檔、執行程式、看到執行結果，若有錯誤也可在同一畫面除錯、再執行，直到滿意爲止。Anaconda雖不是Python官方正式出版的整合程式開發環境，但卻是目前最實用、最強的Python外掛整合開發環境，所以本書選用此編譯程式。Anaconda官網(anaconda.com)如圖1-5，請依下圖點選「Download」下載安裝執行檔。另外，因爲是執行檔，所以只要按照指示即可完成安裝。

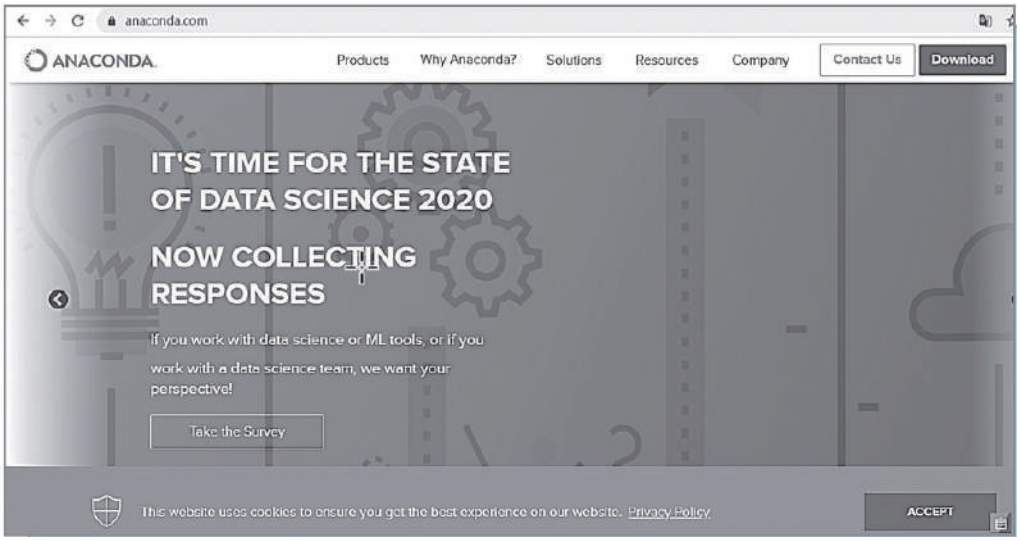


圖1-5 Anaconda官網首頁

以上執行檔安裝完成後，會在程式集出現「Anaconda」資料夾，請點選資料夾裡面的「Spyder」即可進入整合開發環境，如圖 1-6。以下是 Spyder 開新檔案（點選功能表的「File/NewFile」）的畫面。左邊窗格是程式編寫區，右上方窗格是輔助說明區，右下方窗格是程式執行輸出區。

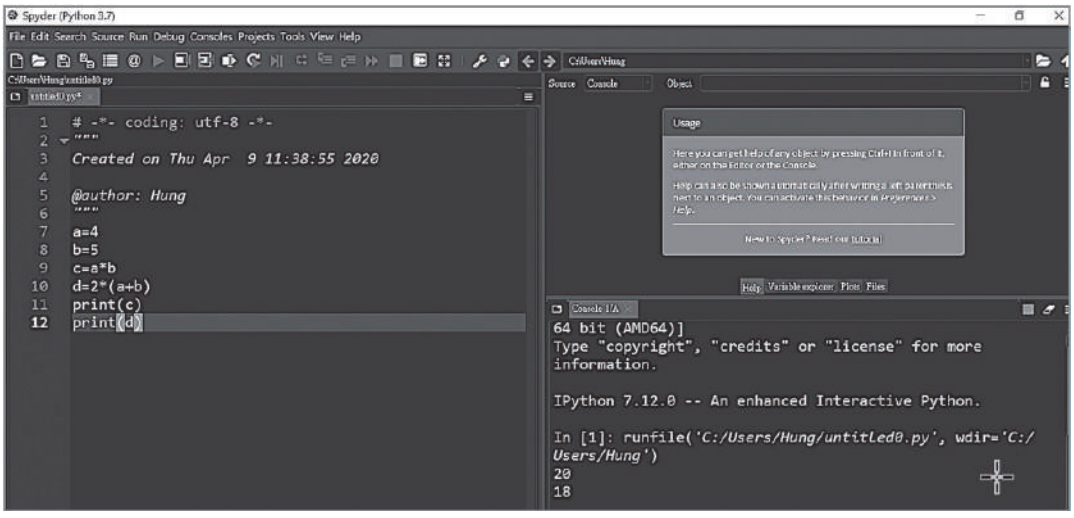


圖1-6 Spyder整合視窗

左邊的程式編寫區中，第一行的井號「#」與兩個「三雙引號（"""）」之間都是註解，此註解記載此程式開發的時間與作者姓名。這些文字僅給人看，電腦都不解譯。

程式編輯

圖 1-7 是一個已知長方形邊長，計算面積與周長的程式，請同學練習鍵入這些程式敘述。圖 1-8 則是寫入此程式的畫面。

```
a=4
b=5
c=a*b
d=2*(a+b)
print(c)
print(d)
```

圖1-7　計算長方形面積程式

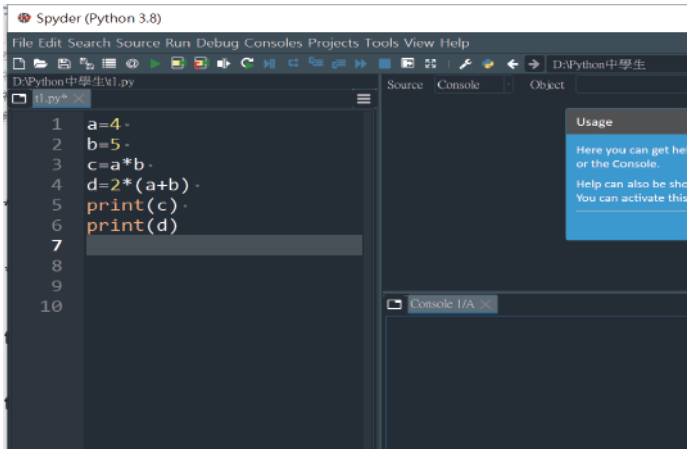


圖1-8　Spyder編輯視窗

程式的執行

圖 1-9 是點選工具列的 ▶「Run file (F5)」的畫面，或點選功能表的 Run/Run」亦可執行程式。因爲是第一次執行，電腦會出現存檔對話框，要求先存檔，請點選資料夾，填入主檔名就好（例如可填入 t1），附檔名預設爲「.py」，不用鍵入。執行結果就在右下窗格。若有任何錯誤，請自己修改，然後再執行程式即可。

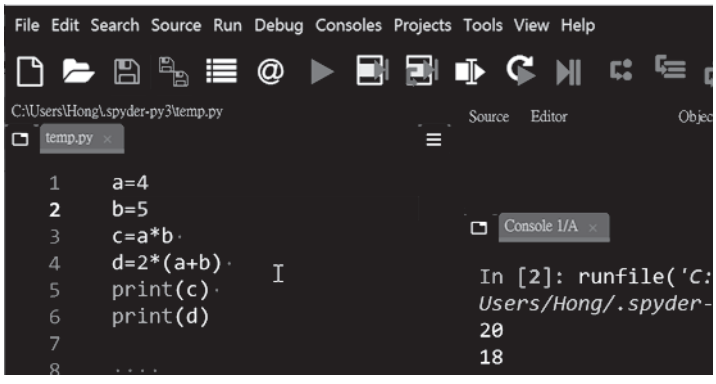


圖1-9　Spyder執行結果畫面

修改編輯環境

點選功能表的「Tools/Preferences/Appearance」，畫面如圖 1-10，可在此修改程式字型的大小。

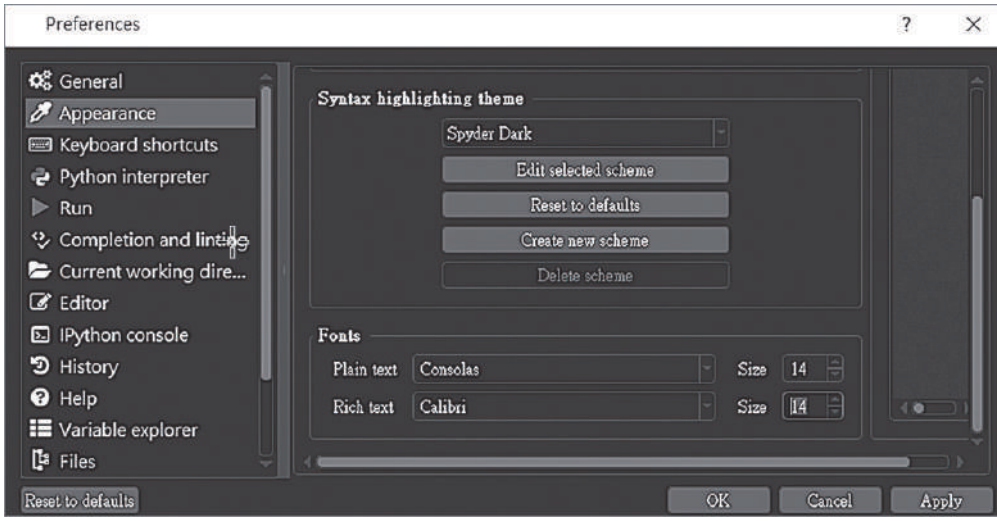


圖1-10　Preferences微調視窗

自我練習

以下是解一元二次方程式 $x^2+4x-5=0$ 的程式，請自行使用 Python 整合開發環境輸入，並觀察輸出結果是否爲 1 與 -5。

```
a = 1 ; b = 4 ; c = - 5
d = ( b ** 2 - 4 * a * c ) ** ( 1 / 2 )
x1 = ( - b + d ) / ( 2 * a )
x2 = ( - b - d ) / ( 2 * a )
print ( x1 )
print ( x2 )
```

1-3 物件導向程式設計的認識

人類之所以會是萬物之靈，其中一個主要原因是人類可以在錯誤中成長。物件導向的程式設計（Object Oriented Programming, OOP）也是人類在程式語言中逐漸累積的成果，這個觀念在 1970 年代就已提出，只是當時時機未到，未受到任何程式語言支持。現在，OOP 則已是所有程式語言的基本設計理念，爲了說明 OOP 大行其道的原因，我們將程式語言的發展分爲 3 個時期，分別是非程序導向、程序導向及物件導向，三者分別說明如下。

非程序導向（NonProcedure Oriented）

早期的程式語言，並沒有內儲副程式（Subroutine）。當開發新的應用程式時，如果某一功能與之前寫過的程式相近，則會將此段已完成的程式整段複製，並稍加修改即可重新加以利用。但是，這些程式的分身包括本尊，自從複製出來以後就開始以各自的方式發展，結果造成各版本的差異越來越大，這些程式很難分辨誰複製誰，彼此之間也難再共用某些程式碼，當遇到錯誤或欲新增功能時，更是很難逐一修改所有程式。

例如：以組合計算C_n^m爲例，$C_n^m = \frac{m!}{n!(m-n)!}$，$m!$=1*2*3…*$m$（此爲階乘計算），因爲非程式導向時代沒有函式，階乘計算共要寫 3 次，所以程式如下：

```
m=5
n=2
#計算階乘
s1=1
for i in range(1,m+1):
    s1=s1*i
#計算階乘
s2=1
for i in range(1,n+1):
    s2=s2*i
#計算階乘
s3=1
for i in range(1,m-n+1):
    s3=s3*i
print(s1/(s2*s3))#10
```

以上計算階乘共要寫 3 次，程式有點冗長，若要修改階乘運算，更要在 3 個地方同時修改，若沒有完全修改，將造成錯誤結果。

程序導向（Procedure Oriented）

爲了解決以上程式共用問題，各編譯器廠商便將一些常用功能寫成函式。較有規模的軟體設計公司亦會將常用的函式集中在一個函式庫，旗下的軟體產品一律呼叫這些標準的函式庫，而不是從函式庫複製出來修改，此即爲程序導向的程式設計。程序導向與非程序導向相比，的確解決了程式共用的問題，但是，人們並不以此爲自滿，有些問題還是不夠順暢。例如：有些函式庫會隨著人們需求的增加而有不同版本，當某些函式功能增加時，只好重新取函式庫的函式加以修改，並賦予新的名稱。如此日積月累，我們的函式庫已有許多函式，這些函式有的功能相近、有的是前後版本不同、有的函式裡的變數來龍去脈不明，造成使用者的混亂。爲了突破以上瓶頸，於是有物件導向的發展，以解決以上程序導向的不足。

以前面計算C_n^m爲例，階乘計算共 3 次，則先以函式完成階乘計算，往後所有程式都可使用此函式，這樣若要修改函式內容，只要統一在函式內修改，所有程式都有相同結果。以上程序導向程式如下：

```
#先以函式完成階乘計算
def fsum(a):
    s=1
    for i in range(1,a+1):
        s=s*i
    return s
#主程式
m=5
n=2
s1=fsum(m) #呼叫fsum函式
s2=fsum(n) #呼叫fsum函式
s3=fsum(m-n) #呼叫fsum函式
print(s1/(s2*s3))#10.0
```

物件導向（Object Oriented）

程序導向中的函式，存有許多解決問題的函式（函式在目前的物件導向程式設計中稱爲『方法』），它是偏重在方法的解決。但是，人類的生活方式不僅是單純的行爲描述，更存在著屬性的記載。例如：當在描述一個人時，通常描述如：『他的名字是洪國勝，身高 172、體重 70，具有滾進、游泳及跑步的能力』; 又例如：描述一輛車子時，其描述如：『 它的名字是 SENTRA，排氣量是 1600c.c.，耗油量是每公里 0.1 公升，且具有每小時 120 公里的移動能力』。以上人與車即稱爲『 物件』，名字、身高、排氣量則稱爲『屬性』或稱『資料』，而滾進、游泳、跑步、移動則稱爲『方法』。既然眞實世界的事件都有『資料』與『方法』，程式設計亦不應侷限在狹隘的函式，僅偏重解決問題的方法，而是應以物件的宏觀角度撰寫程式。所以，基於物件導向的新觀念，程式開發工具即制訂一種新的型態，此型態稱爲『類別』。每一個類別都有屬於自己的『資料』與『方法』。有了『類別』，我們可將眾多的函式依照類別存放，如此可解決目前日益龐大的函式命名與函式取用的困擾。例如：程序導向的時代 , 關於開門的函式即有『電梯開門』、『汽車開門』、『房子開門』等數種開門的『方法』，如此徒增命名與取用的困擾。但在物件導向的領域裡，開門這個方法是附在相對應的類別裡。例如：於電梯類別裡有電梯的開門，汽車類別有汽車的開門，房子類別裡有房子的開門方法，大家的方法名稱都叫『 開門』，撰寫程式時也是『電梯 . 開門』，『汽車 . 開門』，或是『房子 . 開門』（補充說明:物件與方法、屬性之間以點「.」運算子連結），如此既可簡化程式的撰寫，亦可減少程式出錯的機會。但是，在程序導向的領域裡，所有的函式都集中，就有可能用錯方法。例如：用開電梯門的方法去開汽車的門，結果當然是錯的。

物件（Object）

物件導向的程式設計是先建立類別，就如同先建立產品模型，然後我們使用類別建立物件，並且可以將此物件指派給變數，物件亦稱爲類別的實現或類別的樣例化（Instance）。也就是說，類別就像是一個模型，在這些模型裡面已設計好他所具備的資料與方法，當您需要物件時，只要依照這些模型樣例去產生一個或多個物件即可。

再以前面計算C_n^m爲例，函式導向僅在乎函式，物件導向則包含資料成員與函式成員，這樣才能完整展現此物件樣貌。因爲階乘的計算，包括輸入、輸出等資料成員與計算階乘的函式成員，所以程式如下：

```
#建立 Factorial 類別
class Factorial:
    a=1 #資料成員 階乘輸入
    b=1 #資料成員 階乘輸出
    def sum(self): #函式成員，計算階乘
        s=1
        b=self.a
        for i in range(1,b+1):
            s=s*i
        self.b=s  #將計算結果指派給b
#主程式
m=5
n=2
c=Factorial() #以c變數樣力產生1個物件
c.a=m  #指派輸入值
c.sum()  #計算階乘
s1=c.b  #傳回結果
c.a=n   #指派輸入值
c.sum()  #計算階乘
s2=c.b  #傳回結果
c.a=m-n  #指派輸入值
c.sum()  #計算階乘
s3=c.b  #傳回結果
print(s1/(s2*s3)) #10.0
```

抽象化（Abstraction）

什麼是抽象化呢？例如，拍照與攝影是一種具體的表現，但是繪畫就需要抽象化。因爲人類不可能、也沒有需要一五一十的具體描繪，所以需要抽象化。優秀的抽象化，更要能以很簡潔的描述就能夠傳達繪畫者的意念。所以畢卡索的抽象畫就很精簡，但卻能讓普羅大眾讚賞，甚至不同層面的士農工商與販夫走卒都能感受到自己想要的喜悅。程式設計也是相同的道理，亦要依照不同的需求給予抽象化，讓程式設計者都能很簡單的描述，編譯器就會依照不同的需求編譯與連結不同的程式，而完成不同使用

者的需求。物件導向所提供的多型（Polymorphism）、封裝（Encapsulation）、繼承（Inheritance）等，都可以實現不同等級的抽象化，以解決程序導向的不足。以上多型、封裝、繼承說明如下：

▶ 多型

同一種形式，但編譯器能依照使用者的需求，進行適當的處理。例如：以下同樣是加法運算子「+」，但編譯器卻能依照使用者的需求，找到對應的編譯方式，完成數值相加或字串串接，此即爲編譯器多型的表現。

```
print(3+4)       #結果是7
print("3"+"4")    #結果是字串串接34
```

▶ 封裝

程式設計的開發與其它的工業 , 如汽車、電視、收音機等機械、電子產品相較，可說是起步較晚的領域，在汽車、電視及收音機等產品上，我們發現這些產品按鈕或旋扭有不同的封裝等級，有些是留給使用者操作用，所以就放在明顯的地方讓使用者操作機器，有些則是出廠微調後就隱藏或稱封裝起來，僅供日後技術人員維修用，這樣才不會被使用者任意操作而破壞原先功能。程式設計也是這樣，有些功能可讓使用者操作，有些功能則應封裝，才不會被任意破壞。

▶ 繼承

任何工業或電子產品的開發，均不是無中生有，而是從舊有的產品中繼承某些特性，再加入新的零件或修改部份零件而成一項新的產品。例如：SENTRA180 正是繼承 NEW SENTRA 而來，只是排氣量提高了、內裝豪華了，但是原來的輪胎、方向盤及座椅還是用原來 SENTRA 的東西，這就是繼承的道理，使得新產品的開發得以縮短時程。程式設計也是相同的道理，先建立類別，往後都可以繼承一個或數個類別，再新增或修改方法，這樣才能節省程式開發流程。

1-4 程式設計相關議題

程式語言對科技創新的影響

電動車、無人駕駛等技術日益成熟、線上購物網路交易金額日益蓬勃發展、單晶全球鬧缺貨、5G 網路日益普及，這些創新科技都需要程式設計的支援，所以學會程式設計就能站在科技創新的尖端。

資料保護及資訊安全

網路的發展可說無遠弗界，全世界的電腦都網網相連，國家發展委員會爲規範個人資料之蒐集、處理及利用，以避免人格權受侵害，並促進個人資料之合理利用，特制定「個人資料保護法」，如圖 1-11 所示：

圖1-11 個人資料保護法（摘自法規資料庫）

例如，您寫程式時，將會面對很多客戶的資料，這些資料程式設計人員也不得任意使用、公開、販賣，請資訊相關從業人員，應該先行瀏覽，才能保護自己，以免執業過程不甚觸法，得不償失。

妨害電腦使用罪

我國一位某大學工學院的學生，曾經因爲在 1998 年設計了名爲「 CIH 」的電腦病毒，這種病毒在當年造成全球 6000 萬台電腦受到破壞，該名學生當年散佈電腦病毒後「逍遙法外」，主要因爲當時我國刑法並未針對「散佈電腦病毒」之行爲加以明確規範。

隨著資訊科技日新月異的發展，利用電腦及相關設備的犯罪也日益增加。因此，我國於 民國九十二年六月二十五日 增定刑法第三十六章「妨害電腦使用罪」，以便有效赫阻類似的犯罪行爲，其中第三百六十條之「干擾他人電腦使用罪」，便是明確處罰散佈「電腦病毒」之規定，該條規定：「無故以電腦程式或其他電磁方式干擾他人電腦或其相關設備，致生損害於公眾或他人者，處三年以下有期徒刑、拘役或科或併科十萬元以下罰金。」

次外，鑑於電腦病毒這種惡意的電腦程式，對於電腦系統安全性危害甚鉅，往往造成重大的財產損失，致生損害於公眾或他人，鉅額損失往往難以估計，因此，實有必要對於此類電腦病毒的程式設計者，加以處罰的必要，故刑法第三百六十二條規定：「製作專供犯本章之罪之電腦程式，而供自己或他人犯本章之罪，致生損害於公眾或他人者，處五年以下有期徒刑、拘役或科或併科二十萬元以下罰金。」。刑法第三十六章妨害電腦使用罪如圖 1-12（以上摘自教育部資訊教育司 https://depart.moe.edu.tw/）

圖1-12　妨害電腦使用罪（摘自全國法規資料庫）

勞動基準法

勞動部爲規定勞工勞動條件最低標準，保障勞工權益，加強勞雇關係，促進社會與經濟發展，特制定勞動基準法，請開啓全國法規資料庫(https://law.moj.gov.tw/LawClass/LawAll.aspx?PCode=N0030001)，如圖 1-13。

內有許多勞工勞動權益，請同學自己瀏覽，才能保障自己勞動權益。例如，程式設計容易被歸納爲「包工制」，有時有些突發狀況，上班時間是否過長、長時間沒有休假等，這都會影響身體健康，請從事程式設計工作人員翻閱勞基法，適度的向公司反應。

圖1-13 勞動基準法（摘自全國法規資料庫）

1-5 本章內容摘要

1. 常見的演算法流程表示有三種，分別是自然語言、虛擬碼與流程圖，其中以流程圖最適合初學者。
2. 程式語言的發展分爲 3 個時期，分別是非程序導向、程序導向及物件導向。
3. 物件導向的開發程序是先建立類別，再由類別產生 1 個或多個物件。
4. 物件導向所提供的多型（Polymorphism）、封裝（Encapsulation）、繼承（Inheritance）等，都可以實現不同等級的抽象化，以解決程序導向的不足。
5. 程式設計師設計程式時，必然看到很多個人隱私資料，請瀏覽「個人資料保護法」，不能散播、販賣個人資料，以免觸法。
6. 刑法第三十六章，有訂定「妨害電腦使用罪」，內有一些妨害他人使用電腦的刑法。
7. 勞動基礎法內有保障勞工基本勞動安全的規章。

課後評量

一、填充題

1. 寫出 3 種常用演算法的表示。_____，_____，_____。
2. 寫出流程圖符號的開始符號。_____
3. 寫出流程圖符號的輸出、輸入符號。_____
4. 寫出流程圖符號的運算處理符號。_____
5. 寫出流程圖符號的決策符號。_____
6. 寫出流程圖符號的迴圈重複符號。_____
7. 寫出流程圖符號的連結符號。_____
8. 同樣是「+」運算子，但編譯器能依照使用者的需求，進行數值相加或字串串接，此種特性在物件導向稱為_____。
9. 開發一個方法，但某些參數不想讓使用者任意更動，在物件導向裡稱為_____。
10. 開發新的類別不是從零開始，而是從現有類別新增與修改方法，在物件導向程式設計稱為_____。

Chapter
02

程式組成與語法操作

學習綱要

學習表現

1. 具備撰寫程式語言基本能力，運用資訊科技方法解決問題。

2-1 程式基本架構

前面第一章，我們已經給一個計算長方形面積的程式，讓大家鍵入程式，觀察執行結果，本單元就要開始介紹如何『設計』程式。

範例2-1a

已知長方形的長與寬，請寫一個程式，求其面積。

程式設計解題步驟

1. 資料的數位化與變數設計
   (1) 本題若使用掌上型計算機，其方法是直接將鍵入「長 * 寬 =」，就可得到答案。例如，鍵入「15*8=」，就可得到答案「120」。
   (2) 但使用電腦計算，必須先將所要計算的值先數位化，然後以一個英文代號儲存，例如 a, b, length, width，以上代號在程式設計的領域，稱爲「變數」。例如：

```
a=15
b=8
```

   設定變數的目的是將資料與程式分開，這樣當資料改變時，還可重複計算。而且，計算過程都是以變數與程式指令描述，此稱爲程式設計。此一程式設計只要第一次對，往後都對，計算結果有一致性，不用像計算機要按兩次，才能確認計算結果是否正確。
2. 寫出演算法。本例是輸入長方形的長與寬來計算面積，所以演算法如下：

```
面積=長*寬
```

3. 使用變數與電腦程式指令，完成以上演算法。本例是計算面積，所以是

```
c=a*b
```

   以上「a」代表長，「b」代表寬，而「c」則代表面積。

4. 輸出結果。

```
print(c)
```

5. 全部程式如下：

```
a=15
b=8
c = a * b
print(c)
```

6. 請於 Spyder 鍵入以上程式，並觀察執行結果。以下單元即開始介紹如何編寫這些程式，體驗「程式設計」的神奇功用。

執行結果

```
120
```

程式基本架構

以上範例已經寫出一個程式，程式基本架構如下：

1. 輸入或指派輸入。	`a=15` `b=8`
2. 使用程式語言寫出所有運算。	`c = a * b`
3. 輸出運算結果。	`print(c)`

2-2 程式語法規則

以上範例已經完成一個程式，以下先介紹一些 Python 程式基本語法規則，分別是變數的命名規則、保留字等。更多的程式語法規則，將會在以下單元陸續介紹。

變數的命名

前面已經用 a,b 等符號，代表長方形的邊長，這樣便可求得長方形的面積與周長，「a,b」這些符號在程式設計的領域中稱爲變數，電腦會安排記憶

體儲存這些「值」。爲什麼這些符號稱爲變數呢？因爲這些符號的「值」，在程式執行階段都可以再任意改變，所以就稱爲變數。數學通常用 a,b,c 代表已知數，x,y,z 代表未知數，物理通常以 t 代表時間，v 代表速率。Python 的應用領域涵蓋所有科學、工程與商業計算，所以變數種類也多，以下則是 Python 變數命名取用規則。

1. 變數僅能以大小寫的英文字母或底線「_」開頭。以下是可以辨識的變數名稱。

```
a
i
sum
_sum
```

以下是無法辨識的變數名稱。

```
7eleven     # 不能由數字開頭
%as    # 不能由%符號開頭
```

2. 變數由字母、底線開頭後，僅可由字母、底線及數字組合而成，也不得包含空白。例如，以下是可以辨識的變數。

```
a123
AB5566
_a_b
```

以下是無法辨識的變數名稱。

```
A= # 不能含有 = 號
sum! # 不能含有 ! 號
Age#3 # 不能含有 # 號
a c # 不能含空白
c+3 # 不得含有加號
```

3. 變數的大小寫均視爲不同，例如 Score、score 及 SCORE 皆代表不同的變數。

4. 變數不得使用保留字，如 if、for 等（補充說明：所有程式語言都有保留字，也就是某些單字已經事先定義一些功能，爲了避免混淆，故不能再使用這些單字）。表 2-1 是 Python 的保留字。

表2-1　Python保留字

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
async	elif	if	or	yield

請鍵入以下程式，即可輸出所有 Python 保留字。以下程式輸入程式的過程，請留意程式的「縮排」，遇到縮排時，請按一下鍵盤的「Tab」鍵，而不是刻意按四個空白鍵。

```
import keyword
for a in keyword.kwlist:
    print (a)
```

5. 變數雖然可用中文，但因爲所有程式指令都是英文，若使用中文當變數，還要切換輸入法，鍵入程式的過程較不順，所以不建議使用中文當變數。

6. 變數要盡量依其性質取有意義的英文字，這樣更能充分表達程式內涵。例如：num、total、average。

7. 不要單獨使用 l,o,O 等字母當作變數名稱，因爲這些字母與阿拉伯數字 1,0 很相近，很容易讓人誤判。

自我練習

以下程式可以認識 Python 有哪些關鍵保留字，它會逐一輸出每一個保留字，請讀者跟著輸入，程式最後會統計答對題數。

```
import keyword
r=0
for a in keyword.kwlist:
    print (a)
    b=input("Please type above word:")
    if b==a :
        r=r+1
print(r)
```

2-3 本章內容摘要

1. 程式基本架構是「輸入」、「運算」與「輸出」。
2. 變數的命名規則如下：
   (1) 變數僅能以大小寫的英文字母或底線「_」開頭。
   (2) 變數由字母、底線開頭後，僅可由字母、底線及數字組合而成，也不得包含空白。
   (3) 變數的大小寫均視爲不同。
   (4) 變數不得使用保留字。
   (5) Python 變數可用中文。

課後評量

一、是非題

以下變數命名，對的打○，錯的打 X。

題號	題目	對或錯
1	a	
2	1a	
3	a1	
4	-a	
5	_a	
6	sum	
7	if	
8	IF	
10	and	
11	薪水	

MEMO

Chapter
03

資料型態與運算

學習大綱

- 3-1 資料型態
- 3-2 常數與變數
- 3-3 運算元與運算子
- 3-4 基本指令的操作
- 3-5 標準輸出輸入的操作
- 3-6 運算元與運算子的應用
- 3-7 程式的除錯
- 3-8 本章內容摘要

學習表現

1. 具備撰寫程式語言基本能力，運用資訊科技方法解決問題。
2. 具備程式設計之邏輯思考能力，展現系統思考、分析與探索之素養。
3. 能思辨勞動法令規章與相關議題，省思自我的社會責任。

3-1 資料型態

我們平常處理的資料依其型態可分爲整數、浮點實數、字元、字串與布林值，分別說明如下：

3-1-1 整數（Integer）

2,-3 等稱爲整數。大部分的程式語言必須依照整數的長度（例如，1000 或 100000000 等長度就不同）選用適當的資料型態，但 Python 則不用。例如 : 請鍵入以下程式，觀察執行結果。(本書強調由作中學，所以刻意沒有寫出執行結果，請同學務必自己動手鍵入程式，在下面底線寫出執行結果，這樣印象才會深刻)

```
a=1
b=12345678901234567890123456789
c=-123456789012345678901234567890123456789
print(a)
print(b)
print(c)
```

Python 可以處理的整數有兩種進位方式，分別是十進位 (Decimal) 與十六進位 (Hexadecimal)。其中十進位則以我們平常書寫數字的方式即可。十六進位應以 0X 或 0x 開頭，且以 0,1,2,3,4,5,7,8,9,a,b,c,d,e,f 代表 0 到 15。例如 0x1a 或 0XB 均爲十六進制，分別代表十進位的 26 與 11。請鍵入以下程式，寫出執行結果。

```
a=0x1a
print(a)#26
```

代表 $1*16^1+10=26$(十六進位的 a,b,c,d,e,f 等字元，大小寫都可以)

3-1-2 浮點實數（Float）

數字中含有小數點或指數的稱爲浮點數、實數或浮點實數。以指數爲例，E 或 e 表示 10 的次方，例如 0.0023、2.3E-3 及 2.3e-3 都是表示相同的浮點數；又例如 2.3E+2 則代表 230。浮點數可使用標準寫法或科學符號法表示，例如 321.123 即爲標準寫法，1.23e+4 即爲科學符號表示法。大部分的程

式語言必須依照浮點數的精密程度選用不同的資料型態，但 Python 則不用。例如：

```
a=5.0 #強調這個5,已經精確到小數點1位。
b=0.0023
c=2.3E2
d=2.3e-2
e=-2.3e2
f=-2.3e-2
print(a)                                    _____
print(b)                                    _____
print(c)                                    _____
print(d)                                    _____
print(e)                                    _____
print(f)                                    _____
```

Python 已經簡化資料型態的學習，如此就能降低學習門檻，減輕使用者負擔，就如同現代使用手機拍照，只要按下快門就好，所有光圈、快門、焦距等，電腦都能自動處理。

3-1-3　字元（Char）與字串（String）

Python 不分字元或字串，通通使用單引號或雙引號表示字元與字串。例如：

```
a="a"
b='ab'
c='Mary'
d="國勝"
```

但以下就不行

```
d='aa"
```

3-1-4　布林值（Boolean Value）

人類對於事情的正確與否的表達方式有『對』與『錯』、『正確』與『錯誤』、『是』與『非』。電腦爲了讓此事件有一致性，就特別有一個資料型態，稱爲布林型態，且只有兩個值，分別是『True』與『False』。例如：

```
print(3>2)
print(5<0)
```

3-2 常數與變數

所謂「常數」(Constant)是指程式執行的過程都不會也不需要改變的值，所以稱爲常數；「變數」（Variable）則有可能變動。因爲程式經常帶入不同的數值或必須經過多次反覆運算才能得到結果，這些值就必須用變數儲存，這樣才能跟隨反覆的計算而跟著變動，所以稱爲「變數」。但是 Python 並不用特別區分此兩個名詞，也就是常數不用特別使用 const 標注，都是直接用變數名稱代表以上整數、實數或布林值就可以，但是常數通常全大寫，這樣比較好區分。例如：

```
PI=3.14
E=2.61
RATE=0.062
```

3-3 運算子與運算元

前面有談到關於面積的計算 c=a*b，以上「a,b」稱爲運算元，「*，=」稱爲運算子，「a*b」稱爲運算式，c=a*b 稱爲「敘述」。所謂運算子 (Operator)，指的是可以對運算元 (Operand) 執行特定功能的特殊符號。Python 的運算子分爲四大類，分別是算術 (Arithmetic) 運算子、複合指派運算子、關係運算子、邏輯運算子。除此之外，我們還會討論運算子的優先順序 (Precedence) 與結合律 (Associativity)。優先順序用來決定同一式子擁有多個運算子時，每一個運算子進行運算的優先順序；而結合律則決定了在同一敘述中，相鄰的運算子有相同優先順序的運算子執行的順序。

算術運算子（Arithmetic operators）

小型計算機有加、減、乘、除等鍵，而程式語言要能計算加減乘除，也要有這些算術運算子。表 3-1 是 Python 的算術運算子列表：

表3-1　Python算術運算子

運算子	定義	優先順序	結合律
**	次方	1	由左至右
+/-	正負號	2	由右至左
*	乘法運算	3	由左至右
/	除法，得實數商	3	由左至右
//	除法，得整數商	3	由左至右
%	求餘數 (Modulus)	3	由左至右
+/-	加法 / 減法運算	4	由左至右
=	指派	14	由右至左

以上運算子的運算結果、運算子優先順序、運算子結合律等，分別說明如下：

指派運算子（Assignment operators）

指派運算子的符號爲「=」，又稱爲「賦值」運算子，其作用爲將運算符號右邊運算式的值指派給運算符號左邊的運算元。所以，以下敘述 s=a+b 是將 a+b 的值先計算，然後指派給 s，或稱賦值給 s。

```
s=0 ;a=3 ;b=5
s= a +b
print(s)
```

上式與數學的「等號」意義是不同的，所以不要一直糾結爲什麼 0 會等於 a+b，而是 a+b 先運算，然後指派或稱賦值給 s。其次，不能將常數放在指派運算子的左邊，例如：

```
8=a
```

此爲一個錯誤的敘述。但以下敘述，將常數 8 指派給變數 a 是正確的。

```
a=8
```

▶ 算術運算

以下是簡單的算術運算，請自行鍵入，並寫出執行結果。

```
a=5;b=4 #指派或稱賦值
print(a+b)                                    _____
print(a-b)                                    _____
print(a*b)                                    _____
print(a/b) #實數除法，得到實數商                  _____
print(a//b) #整數除法，得到整數商                 _____
print(a%b)                                    _____
print(a**2)                                   _____
print(9**(1/2)) #開根號                       _____
print(27**(1/3))#開立方                       _____
print(10**2)                                  _____
print(10**-2)                                 _____
print(1/10**2)                                _____
```

▶ 複合運算

程式設計常需要計算

```
s=s+a;s=s-a;s=s*a;s=s//a;s=s/a;s=s%a
```

所以以上可以寫成如下：

```
s+=a;s-=a;s*=a;s//=a;s/=a;s%=a
```

此稱爲複合運算子。請鍵入以下程式，並寫出執行結果。

```
s=0;a=3;b=4
s+=a
print(s)                                      _____
s-=b
print(s)                                      _____
```

▶ 整數除法與取餘的應用

平常的算術運算對於整數除法與取餘並無特別著重。但在程式設計的領域，這兩個運算子有其獨到用途。因爲整數除法與取餘可以將一個整數分解爲單一個數字，且很多應用都會用到此「分解」運算。例如，若要在顯

示器顯示 152，那就要將此數字先分解爲 1、5、2 三個數字，請自行鍵入以下程式，並觀察結果。

```
a=152
a3=a%10 #2 個位數
a=a//10
a2=a%10 #5 十位數
a=a//10
a1=a    #1 百位數
print(a1,a2,a3)
```

▶ 程式語言與數學的差異

數學有

```
19/10=1…9
```

的書寫習慣，但程式語言並沒有此表示方式，因此以上運算方式，一定要先取餘數，再求整數商如下：

```
a=19;b=10
c=a%b;#餘數
a=a//b; #整數商
print(a,c)
```

順序錯也不行，以下程式，先除再取餘數，結果就不對。

```
a=19;b=10
a=a//b;
c=a%b;
print(a,c)
```

自我練習

1. 請指派一個四位數，並將其反向輸出此四個數字且求其數字和。例如，指派 1234(a=1234)，那可以輸出 4 3 2 1 與 10。(數字之間留空白，且先不要用迴圈)
2. 請寫一個程式，可以指派 4 個數字，然後將其兜成一個 4 位數輸出。例如，指派 a=3,b=4,c=5,d=6，那可以輸出「3456」。

3. 請寫一個程式，可以指派 1 個 0 到 86399 的整數，然後分解爲「時:分:秒」的輸出格式。
4. 請寫一個程式，可以指派 1 個 0 到 999999 的整數，然後從右邊兩兩分解與輸出。例如，指派輸入 123456 則輸出 56,34,12。

關係運算子(Relational Operators)

關係運算子又稱爲比較 (Comparison) 運算子，用於資料之間的大小比較，比較的結果可得到布林值的 True 或 False，表 3-2 是 Python 中的關係運算子符號。

表3-2 Python 關係運算子

運算子	定義	優先順序	結合律
<	小於	9	由左至右
>	大於	9	由左至右
<=	小於等於	9	由左至右
>=	大於等於	9	由左至右
==	等於	9	由左至右
!=	不等於	9	由左至右

例如，請鍵入以下程式，寫出執行結果。

```
x=3;y=4
print(x==y) #False
print(x!=y)#True
print(x<y)#True
print(x=y)#錯，此爲指派運算子，不是關係運算子
```

邏輯運算子(Logical Operators)

邏輯運算子 (Logical Operators) 又稱爲布林 (Boolean) 運算子。當同一個運算式要同時存在兩個以上的關係運算子時，每兩個關係運算子之間必須使用邏輯運算子連結，例如，您要找『男生』且『年齡大於 40』，此一選擇就同時含有兩個關係運算式，此時就要使用邏輯運算子。下表是 Python 的邏輯運算子，C/C++ 用符號『！ ,&&,||』表示，Python 直接用單字表示，如表 3-3 所示，這樣反而比較好記。

表3-3　Python 邏輯運算子

運算子	定義	優先順序	結合律
not	邏輯否定運算	10	由右至左
and	邏輯 and 運算	11	由左至右
or	邏輯 or 運算	12	由左至右

▶ not

邏輯not稱爲否定運算，可將『True』轉爲『False』，『False』轉爲『True』。例如：

```
a=False
print(not a)
```

▶ and

邏輯 and 是兩件事都 True，結果才是 True，其餘都是 False，其眞值表如表 3-4。

表3-4　邏輯 and 真值表

事件1	事件2	結果
True	True	True
True	False	False
False	True	False
False	False	False

例如：

```
x=3;y=4
print(x>0 and y>0) #True
print(x>0 and x==y) #False
```

▶ or

邏輯 or 是兩件事只要有一件爲 True，那就爲 True，只有兩件事全爲 False，才爲 False，其眞值表如表 3-5。

表3-5 邏輯 or 真值表

事件1	事件2	結果
True	True	True
True	False	True
False	True	True
False	False	False

例如：

```
x=3;y=4
print(x>0 or x==y) #True
print(x<0 or x==y) #False
```

又例如：若有一數學式，判斷 x 是否滿足 1<x<=6 , 請轉換爲 Python 語言敘述。因爲 1<x<=6 是數學語言，Python 是

```
x>1 and x<=6
```

請鍵入以下程式，並觀察執行結果。

```
x=3
print(x>1 and x<=6)                    _____
x=8
print(x>1 and x<=6)                    _____
```

但太多初學者習慣將以上數學語言直接用。例如：

```
x=8
print(1<x<=6)                          _____
```

以上寫法，Python 竟然也可以接受，其它語言都不行，這也是物件導向抽象化的特性。

自我練習

1. 表決器。假設有一項評審工作，有 3 位評審，當其中兩人或以上同意，則表示過關，請設計此程式。
2. 請指派 1 個整數 x，判斷 x 是否滿足 x>3 or x<-2 ？

3. 請指派 3 個線段長度，判斷這 3 個線段，是否可圍一個三角形？
   (提示：任兩邊之和大於第三邊的數學語言是 :a+b > c and a+c > b and b+c > a，請將以上數學語言轉爲 Python。測試資料 3,4,5 可以；1,1,3 不可以)
4. 若有一數學式，如何同時判斷六個數字是否滿足 $\frac{a_1}{a_2}=\frac{b_1}{b_2}=\frac{c_1}{c_2}$ ，請轉換爲 Python 敘述。例如，a1,b1,c1=1,2,3 與 a2,b2,c2=2,4,6 爲 True；1,2,3 與 1,2,5 爲 False。
5. 若有一數學式，如何同時判斷六個數字是否滿足 $\frac{a_1}{a_2}=\frac{b_1}{b_2}\neq\frac{c_1}{c_2}$ ？

運算子的優先順序（Precedence）

同一敘述，若同時含有多個運算子，即須定義運算子的優先順序。例如：

```
x=a+b*c
```

在數學裡，是定義先乘除再加減，程式語言也是相同，也必須定義這些運算子的優先順序，茲將使用手冊的運算子優先順序摘錄如表 3-6，有些還沒介紹，那就先瀏覽。

表3-6　運算子優先順序表

Operator	Description	優先順序（1最優先）
`lambda`	Lambda expression	17
`if – else`	Conditional expression	16
`or`	Boolean OR	15
`and`	Boolean AND	14
`not x`	Boolean NOT	13
`in, not in, is, is not, <, <=, >, >=, !=, ==`	Comparisons, including membership tests and identity tests	12
`|`	Bitwise OR	11
`^`	Bitwise XOR	10
`&`	Bitwise AND	9

表3-6 運算子優先順序表（續）

Operator	Description	優先順序（1最優先）
<<, >>	Shifts	8
+, -	Addition and subtraction	7
*, @, /, //, %	Multiplication, matrix multiplication, division, floor division, remainder 5	6
+x, -x, ~x	Positive, negative, bitwise NOT	5
**	Exponentiation 6	4
await x	Await expression	3
x[index], x[index:index], x(arguments...), x.attribute	Subscription, slicing, call, attribute reference	2
(expressions...), [expressions...], {key: value...}, {expressions...}	Binding or tuple display, list display, dictionary display, set display	1

請自行鍵入以下程式，並寫出執行結果。

```
print(3**2)                                    _____
print(-3**2)                                   _____
print((-3)**2)                                 _____
print(9**(1/2))                                _____
print(9**1/2)                                  _____
print((3/10)**2)                               _____
print(3/10**2)                                 _____
print(4+3**2%2+1)                              _____
```

運算子的結合律（Associativity）

當同一敘述，相鄰的運算子擁有相同優先順序的運算子，即須定義運算子是「由左至右」的左結合或「由右至左」的右結合。例如：

```
x=a-b-c
```

同樣是減號「-」，優先順序相同，此時就要靠定義結合律，減法結合律是由左至右，所以以上同義於

```
x=((a-b)-c))
```

而

```
a=b=c=2
```

指派運算子的結合律是由右至左，所以以上式子同義於：

```
(a=(b=(c=2)))
```

請鍵入以下程式，寫出執行結果。

```
a,b,c=2,3,4
x=a+b-c
print(x)                                    _____
a=b=c=3
print(a,b,c)                                _____
```

函式

所有的編譯器爲了節省有限的記憶體，都是將最常用的運算寫成運算子，如以上的算術、關係、邏輯運算子。比較不常用的運算，例如，取絕對值就以函式 abs() 表示。函式將會在第七章詳細介紹，以下先將常用函式介紹如下：

▶ abs(x)

傳回數值 x 所對應的絕對值。例如：

```
a=-3
print(abs(a))#3
```

▶ round(a)

將數值 a 四捨五入到整數。例如：

```
print(round(255.5))#256
```

▶ str(a)

將數值 a 轉爲字串。例如：

```
a=123
b=456
print(a+b)
```

此爲數值相加，結果是 579，若要將這些數字進行字串串接，則要先轉爲字串，再串接，程式如下：

```
print(str(a)+str(b))  #字串串接123456
int(x)
```

將字串 x 轉爲數值。例如：

```
a="123"
b="456"
print(a+b)  #此爲字串串接123456
print(int(a)+int(b))  #先轉數值，則爲數值相加
```

▶ ord(x)

傳回字元 x 的 ASCII 編碼。例如：

```
print(ord("A"))  #65
```

補充說明

什麼是 ASCII 編碼呢？因爲資料要由電腦運算，所以每一個數字、符號、大小寫英文字元，都要數位化，才能用電腦儲存，再進行運算。這些常用字元沒有超過 127 個，所以使用 7 位元編碼，就可以儲存常用字元，此稱爲 ASCII 編碼。例如：字元 A 編碼爲「01000001」=65，叫到 65 號就是字元「A」;字元「B」的編碼是「01000010」=66，叫到 66 號，就是「B」;字元「1」的編碼是「00110001」=49，叫到 49 號就是「1」。就如同每個同學都有座號，所以就有對應位置。

▶ chr(x)

傳回數值 x 所對應的字元。例如：

```
a=65
print(chr(a))#A
```

▶ len(a)

傳回字串 a 的長度。例如：

```
a="abcd"
print(len(a))#4
```

▶ max(x)、min(x)

傳回參數 x 的極大或極小值。例如：

```
b=6;c=3;d=9
print(max(b,c))#6
print(min(b,c,d))#3
```

▶ eval(x)

傳回字串 x 算術運算式的運算結果。例如：

```
print(eval("3+2*4"))#11
```

這也很神，例如：用來製作計算器就很方便，只要讓使用者按完一個運算式，Python 就幫您自動計算結果。

3-4 基本指令的操作

以上已經介紹一些程式語法規則，以下介紹一些 Python 基本指令的操作，例如：變數的宣告、指令的排列、註解等。更多的指令介紹，將在後續章節陸續介紹。

變數的宣告

大部分的程式語言在變數宣告的同時要指派其型態是整數、實數或字串，但是 Python 只要指派其初值就好，編譯器會依照實際情況指派記憶體的多寡。請鍵入以下程式，並觀察執行結果。

```
a=2  # 整數
print ( 2 ** 100 ) # 2 的100 次方
b = 3.4   # 浮點實數
print ( b ** 100 ) # b 的100 次方
```

程式指令的排列

以上我們都將是同一列僅放置一個敘述，若要將兩個敘述放在同一列，請用分號「;」，例如：

```
a=1;b=2;print(a+b)
```

而以下程式，雖然沒有錯，但卻是多此一舉，因爲這不是 C 或 Java，每一列不用以分號「;」做結束。

```
a=1; b=2;
print(a+b);
```

▶ 變數不用宣告的困境

變數只要放在指派運算子「=」左邊就等於宣告，這樣就可在指派運算子右邊使用，但也會有其缺點。請鍵入以下程式，並觀察執行結果

```
student=0
student=stutent+1  # 本例表示s t u d e n t 變數要執行累加的動作
print(student)
```

以上第二列程式指派運算子「=」右邊的 stutent 拚字錯誤（因爲此變數未曾在左邊出現過），Python 解譯器可以馬上幫忙找出錯誤，就可以修正如下：

```
student=0
student=student+1   # 本例表示s t u d e n t 變數要執行累加的動作
print(student)
```

得到正確結果。但以下程式可就沒那麼順利了。

```
student=0
stutent=student+1
print(student)
```

請問錯在哪了？剛剛是錯在指派運算子「=」的右邊，Python 找出了錯誤，這次則是錯在指派運算子左邊。因爲指派運算子「=」右邊的變數要先宣告，這樣可預防打字錯誤，左邊則是你說了算，電腦不會幫忙除錯，所以指派運算子左邊的變數要特別留意。又例如：

```
Score=1
score=Score+1 （本例表示同一變數要執行累加的動作）
print(Score)
```

沒有出現錯誤訊息，但結果也是錯誤的，因爲大小寫不同，表示不同的變數名稱，請特別留意運算子左邊的變數名稱不能錯誤。以下程式也不行，不小心重複使用同一變數，造成嚴重錯誤。

```
a=[3,4,5];print(a)#宣告a爲串列
a=6;print(a)#又宣告a爲單一變數
if a>3:
    print("OK")#OK
if a[2]>2:#錯誤，因a已經是單一變數了
    b=3
```

▶ 變數同時交換內容

變數同時交換內容是程式設計常需使用的運算，例如，變數 a,b 要交換內容，大部分的程式語言都要先找一個臨時變數 t，程式如下：

```
a=1;b=2
t=a
a=b
b=t
print(a,b)#2 1
```

Python 就允許使用者同時交換變數內容。請鍵入以下程式，並觀察與寫出執行結果。

```
a,b=1,2
a,b=b,a
print(a,b)#2,1
a,b,c=1,2,3
a,b,c=b,c,a
print(a,b,c) #2,3,1
```

註解 (Comments)

適當的程式註解才能增加程式的可讀性，註解是僅給人看，電腦不會理會。Python 的註解有兩種方式。第一，使用『"""』當註解開頭，直到遇到『"""』，兩個『"""』中間的文字通通是註解。

```
"""我是註解
我也是註解"""
```

上式『""" 』符號以後的字串視爲註解，編譯程式不予處理，直到遇到『 """ 』爲止。其次，同一列中，井號『#』後面的也視爲註解，編譯器均不予處理。例如：

```
# 我是註解
```

前者，因爲可超過兩列，較適合編寫較長的註解，後者，則僅能寫在同一列。

亂數

電腦常常需要模擬一些執行結果，例如，樂透開獎、紙牌遊戲或擲骰子，此時就需要產生亂數，Python 產生亂數的方法是使用 random 模組。使用這些模組前，請先載入此模組，程式如下：

```
import random
a=random.randint(1,6)#產生1～6整數亂數
print(a)
```

每次用『類別名稱 . 方法』有點麻煩，可以取一個簡單的物件名。例如：

```
import random as a # a是物件名
b=a.randint(1,6) #產生1～6整數亂數
print(b)
```

▶ randint(min,max)

傳回 min(含) 與 max(含) 之間的亂數。例如，以下程式可模擬擲骰子的結果。

```
import random
a=random.randint(1,6)
print(a)
```

（補充說明：以上 randint 在物件導向的程式設計稱爲方法。）

小寫字元的 ASCII 是 97～122，所以以下程式，可產生一個小寫字元。

```
import random
a=random.randint(97,122) #產生97～122整數亂數
print(a)
print(chr(a))#傳回對應字元
print("%c" %a)#將整數以字元形式輸出
```

ord() 函數可取得傳入字元的 ASCII 值，所以以下程式也可以傳回一個小寫字元。

```
import random
a=random.randint(ord('a'),ord('z'))#ord傳回對應整數
print(a)
print(chr(a))
print("%c" %a)
```

自我練習

1. 請寫一個程式，可以產生大寫字元。
2. 請寫一個程式，可以產生二位數的整數。
3. 請寫一個程式，可以產生一個介於 0~1 且包含 0，但不含 1 的浮點實數，小數點取 2 位。

字串運算子

Python 是一個物件導向程式語言。物件導向的抽象化，就是希望程式語法能越來越接近人類語言，所以，Python 開發『+, *, in』等運算子 , 希望字串的運算也與人類的運算習慣相同。請鍵入以下程式：

```
a="aa"
b='Mary'
print (a+b)#合併 aaMary
c=a*3
print(c)#三次  aaaaaa
print ('a' in b) #True
print ( 'a'  not in b)#False
```

補充說明

以上「+」、「*」都是物件導向「多型」的表現，使用者都是使用相同的運算子，但編譯器就能依照實際情形，完成整數、浮點數與字串等運算。

字串與數值互轉

a=12 是數值，b="34" 是字串，但是有時候輸入時是字串，但要改爲數值來運算；或某些數值希望轉爲字串處理，此時就要先用 str() 將數值轉爲字串，用 int() 將字串轉爲數值，如此才能進行相關運算。例如：

```
a=12;b=34
print(a+b)#46
print(str(a)+str(b))#1234
c='12';d='34'
print(c+d)#1234
print(int(c)+int(d))#46
```

但是，不是數字的字串，例如：強制將 "Mary" 轉爲數值，也是沒意義，且發生錯誤訊息。

```
b="Mary"
c=int(b)#錯
print (c)
```

字串的長度

字串的長度也是依使用者習慣，英文字元與中文字元都能順利按照其習慣，計算其長度。(以前的程式語言，一個中文字長度是 2，這樣雖然對，但是反而不好資料處理，這也是物件導向「多型」的表現。)

```
a='aa'
b='泉勝'
print(len(a))#2
print(len(b))#2
```

字串的字元處理

字串是由字元組成，若我們要取個別字元，也可直接用串列 (list) 處理，串列請看第六章，例如：

```
a="a"
b='ab'
c='Mary'
d="國勝"
print(a[0])# a
print(b[0],b[1])# a b
print(d[0],d[1])#國 勝
```

字串的分割

Python 並沒有時間與日期等資料型態，關於日期與字串都是以字串表示，例如：

```
a="112/3/2"
b="18:30:22"
```

此時若要進一步處理日期與時間相關運算問題，則需要分解。Python 提供 split() 函式分解字串。例如：請鍵入以下程式，寫出執行結果。

```
a="112/3/2"
y,m,d=a.split('/')#分解後資料型態爲字串
print(y,m,d)                                    _____
print(y+m+d)                                    _____
```

```
y,m,d=int(y),int(m),int(d)#轉爲數值
print(y,m,d)                                    _____
print(y+m+d)                                    _____
```

以上分解 (split)、轉數值 (int) 兩個動作，亦可使用 map() 函式。例如：請鍵入以下程式。

```
y,m,d=map(int,a.split('/'))
print(y+m+d)                                    _____
```

自我練習

1. 請計算以下兩個時間相差的時間間隔。（提示：通通轉爲秒數）

```
b1="18:30:22"
b2="20:10:42"
```

2. 請判斷以下兩個時間何者在前。（提示：通通轉爲秒數）

```
b1="18:30:22"
b2="20:10:42"
```

3. 請判斷以下時間 b 有沒有在時間 b1、b2 的中間。（提示：通通轉爲秒數）

```
b="17:40:52"
b1="18:30:22"
b2="20:10:42"
```

發聲

要讓電腦發聲，要先載入 ctypes 類別，且取 ctypes.windll.kernel32 物件。例如：

```
import ctypes
p = ctypes.windll.kernel32
```

電子琴鍵盤頻率對照表如表 3-7。

表3-7　電子琴鍵盤頻率對照表

	n	1	2	3	4	5	6	7	8	9	10	11	12
音階	音符	C (Do)	C# (Do#)	D (Re)	D# (Re#)	E (Mi)	F (Fa)	F# (Fa#)	G (So)	G# (So#)	A (La)	A# (La#)	B (Si)
低音	頻率 (Hz)	262	277	294	311	330	349	370	392	415	440	466	494
中音	頻率 (Hz)	523	554	587	622	659	698	740	784	831	880	932	988
高音	頻率 (Hz)	1046	1109	1175	1245	1318	1397	1480	1568	1661	1760	1865	1976

以下程式可讓電腦發出 Do、Re、Mi、Fa…等 8 個音，523、587 就是震盪的頻率。

```
import ctypes
p = ctypes.windll.kernel32
p.Beep(523,200)
p.Beep(587,200)
p.Beep(659,200)
p.Beep(698,200)
p.Beep(784,200)
p.Beep(880,200)
p.Beep(998,200)
p.Beep(1046,200)
```

以上的數值 200，表示發聲的延遲的時間，單位是 ms。

演奏音樂

要讓電腦演奏音樂，那就要先下載一個簡譜，如圖 3-1。

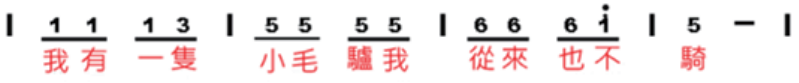


圖3-1　小毛驢簡譜

本例一分鐘 80 拍，所以 1 拍的時間是，60000ms/80，以一個四分音符爲 1 拍，所以上圖一個八分音符所佔的時間是 60000ms/(2*80)，每小節是 2 拍。其次，上圖，所有音符的最小延遲時間是八分音符，所以就取八分音

符的時間爲 1 個單位。第 1 音『1 我』半拍，取 1 個單位;『5- 騎』是 2 拍，取 4 個單位。

資料的數位化

圖 3-1 的小毛驢簡譜，所有音符的最小延遲時間是八分音符，所以就取八分之一音符的時間爲 1 個單位，以下程式可演奏『我有一隻小毛驢，我從來也不騎』。

```
import ctypes
p = ctypes.windll.kernel32
t=60000//160
p.Beep(523,t)
p.Beep(523,t)
p.Beep(523,t)
p.Beep(659,t)
p.Beep(784,t)
p.Beep(784,t)
p.Beep(784,t)
p.Beep(784,t)
p.Beep(880,t)
p.Beep(880,t)
p.Beep(880,t)
p.Beep(1046,t)
p.Beep(784,4*t)
p.Beep(784,t)
```

若加上歌詞，就有點像 KTV 了，程式如下：

```
import ctypes
p = ctypes.windll.kernel32
t=60000//160
print('我')
p.Beep(523,t)
print('有')
p.Beep(523,t)
print('一')
p.Beep(523,t)
print('隻')
p.Beep(659,t)
```

```
print('小')
p.Beep(784,t)
print('毛')
p.Beep(784,t)
print('驢')
p.Beep(784,t)
print('我')
p.Beep(784,t)
print('從')
p.Beep(880,t)
print('來')
p.Beep(880,t)
print('也')
p.Beep(880,t)
print('不')
p.Beep(1046,t)
print('騎')
p.Beep(784,4*t)
p.Beep(784,t)
```

但是以上程式有點冗長，等到介紹串列與迴圈，程式就會簡化。

自我練習

1. 請自行下載一個簡譜，撰寫程式演奏音樂（請含歌詞）。

3-5 標準輸出輸入的操作

Python 使用 print() 函式輸出，使用 input() 函式輸入，分別說明如下：

print()

前面我們已經使用 print 函式輸出結果，每一個 print() 預設輸出後跳列。例如：

```
print('aa')
print('bb')
```

輸出結果是：

```
aa
bb
```

若您不想讓它跳列，請自行指派 end 值，程式如下：

```
print('aa',end='')
print('bb')
```

輸出結果是：

```
aabb
```

又例如：

```
print('aa',end=',')
print('bb')
```

輸出結果是：

```
aa,bb
```

print() 內若是接變數，就輸出變數的內容，請鍵入以下程式，並觀察與寫出輸出結果。

```
a,b,c=1,2,3
print(a)                                    _________
print(b)                                    _________
print(c)                                    _________
```

若您不想讓它跳列，也是請自行指派 end 值。請鍵入以下程式，並觀察與寫出輸出結果。

```
a,b,c=1,2,3
print(a,end=' ')
print(b,end=',')
print(c)                                    _________
```

若要在輸出結果套用文字說明，如「a=2」，則可使用『%』當作控制字元。使用「%」控制資料輸出時，還要依資料的型態使用對應的符號，整數請用「d」，浮點數使用「f」，字元用「c」，字串請用「s」，如表 3-8。

表3-8　資料型態列印格式對照表

資料型態	列印格式
整數	d
浮點數	f
字元	c
字串	s

請鍵入以下程式，並觀察與寫出輸出結果。

```
a=2
print("a=%d" %(a))
```

又例如：

```
a,b,c=1,2,3
print("a=%d,b=%d,c=%d" %(a,b,c))
```

以上「a=」是字串直接輸出，控制字元「%」會到後面變數區 (a,b,c) 依照順序找對應的變數。所以，輸出會是 :

```
 a=1,b=2,c=3
```

又例如：

```
a="永松";b=56
print("帥哥%s 年齡是%d" %(a,b))
```

輸出結果是：

```
帥哥永松 年齡是56
```

浮點數輸出是 %f。請鍵入以下程式，並觀察與寫出輸出結果。

```
a=3.4
print("%f" % a)
print("%d" % a)#以整數輸出
```

若是控制字元 % 接數字，則表示此欄位的寬度且靠右排列，剩的以空白表示。例如：

```
a="Mary"
b,c=1,2
print("123456789012345678901234")
print("a=%6s,b=%3d,c=%4d" %(a,b,c))
```

輸出結果如下圖：

```
123456789012345678901234
a=  Mary,b=  1,c=   2
```

input()

前面我們一直都用指派的方式指派變數值，input() 可以讓使用者於程式執行後輸入資料。例如，以下程式可以讓我們輸入資料。

```
a=input("input a:")
print(a)
```

其次，Python 將輸入的資料一律視爲字串，所以，若您要進行數值運算，那請先用 int() 函數轉爲數值。請鍵入以下程式、輸入數值，並觀察結果。

```
a=input("input a:")
b=input("input b:")
print(a+b)
a=int(input("input a:"))
b=int(input("input b:"))
print(a+b)
```

eval()

前面使用 input() 每次僅能輸入一筆資料，善用 eval() 函式可以讓我們同時輸入多筆資料，資料的分隔採用逗號「,」。請鍵入以下程式，輸入『12,3,4』，並觀察輸出結果。

```
a,b,c=eval(input("input a,b,c:")) #數字請用逗號『,』隔開，例如 12,3,4
print(a,b,c)                                                    _____
print(a+b+c)                                                    _____
```

請留意其資料型態是數值。eval 還可輸入一個數學運算式，例如：

```
a=eval("3+4*2" )
print(a)                                                        _____
```

結果是 11。請鍵入以下程式，並輸入『3+4*2』，並觀察輸出結果。

```
a=eval(input("input a 數學運算式:" ))
print(a)                                                        _____
```

所以利用 eval() 函式，待學完視窗輸出入元件，很快就可以自行作出具有按鍵功能的小型計算機功能。

範例3-5a

請寫一程式，可以輸入長方形的長與寬，並計算周長與面積，且輸出結果。

程式列印

```
a=int(input("input a:"))
b=int(input("input b:"))
c=a*b
d=2*(a+b)
print("面積=%d ,周長=%d" %(c,d))
```

執行結果

```
input a:3

input b:4
面積=12 ,周長=14
```

補充說明

1. 本例若用

```
a,b=eval(input("input a,b:"))
```

則 a,b 的資料型態是數值，所以程式如下：

```
a,b=eval(input("input a,b:"))
c=a*b
d=2*(a+b)
print("面積=%d ,周長=%d" %(c,d))
```

但使用 input 所輸入的資料型態卻是字串，此時要先轉數值再運算，請特別留意。

自我練習

1. 請分四次輸入 1 個 0 到 9 的整數，並將它合併爲 1 個整數。例如，輸入 1，輸入 2，輸入 3，輸入 4，則輸出 1234。
2. 請先輸入 3 個 0 到 9 的整數，再輸入兩個 0 到 9 的整數，並將其合併爲 1 個浮點數。例如，輸入 1，輸入 2，輸入 3，輸入 4，輸入 5，則輸出 123.45。

3-6 運算子與運算元的應用

以上已經介紹算術運算子與運算元，有了運算子與運算元，我們就可以先將國小與國中的數學問題以程式語言爲工具，撰寫程式，而由電腦求出解答，請看以下範例。

範例3-6a

請寫一程式，可以任意輸入兩個點，計算其距離。

程式設計步驟

1. 將資料數位化與變數設計。本例指派兩個座標如下：

```
x1=3;y1=0
x2=0;y2=4
```

2. 寫出演算法。本例已知兩點座標分別是 (x1,y1),(x2,y2)，則其距離的公式的數學語言是：

$$d=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

3. 以程式語言完成演算法。以上的距離公式以 Python 語言可表示如下：

```
d=((x1-x2)**2+(y1-y2)**2)**(1/2)#此爲Python語言表示法
```

4. 輸出此兩點距離。

```
print(d)#5.0
```

5. 完整程式列印如下：

```
x1=3;y1=0
x2=0;y2=4
d=((x1-x2)**2+(y1-y2)**2)**(1/2)#此爲Python語言表示法
print(d)#5.0
```

執行結果

```
5.0
```

補充說明

1. 本例所有變數內容先使用『指派』，請練習改爲輸入。
2. 因爲有運算子優先順序問題，所以請留意括號不能省略。

自我練習

1. 輸入三角形三邊長 a、b、c，求其面積。（提示：先計算 d=(a+b+c)/2，則三角形面積＝$\sqrt{d(d-a)(d-b)(d-c)}$，本例假設所輸入的三角形三邊長可圍成三角形，例如指派 a=3;b=4;c=5 則得三角形面積 6。其次，並不是任意三條線都可圍成三角形，若要判斷是否可圍成三角形，請繼續研讀下一章）
2. 請寫一程式，可以指派三點座標，輸出其面積。提示：三角形面積公式如下。

$$=\frac{1}{2}\begin{vmatrix} x_1 & x_2 & x_3 & x_1 \\ y_1 & y_2 & y_3 & y_1 \end{vmatrix}$$ 其中↘乘爲正，↙乘為負

$$=\frac{1}{2}|(x_1y_2+x_2y_3+x_3y_1)-(x_2y_1+x_3y_2+x_1y_3)|$$

例如：x1=0;y1=0;x2=3,y2=0;x3=0;y3=4，執行結果是 6。

資料的數位化與變數設計

前面都是指定數值，完成某些功能運算，但有時候，資料比較複雜，這時就需要將資料抽象出來，此資料抽象的過程，又稱爲資料的數位化或資料的抽象。例如：若有一直線方程式是 ax+by+c=0，點 p(m,n) 到直線的距離是

$$d=\frac{|am+bn+c|}{\sqrt{a^2+b^2}}$$ (提示：絕對值函式是 abs())

這件事要由電腦作，就要將資料從人類約定的書寫習慣 ax+by+c=0 獨立抽象出來。例如，直線方程式是 3x+4y+5=0，求點 p(1,2) 到直線的距離，首先，以變數

```
a=3;b=4;c=5
```

約定此爲直線 3x+4y+5=0。然後以變數

```
m=1;n=2
```

約定此爲點 p(1,2)，以上過程稱爲資料的抽象化與變數的設計。所以全部程式如下：

```
a=3;b=4;c=5
m=1;n=2
d=abs(a*m+b*n+c)/((a*a+b*b)**(1/2))
print(d)
```

又例如，您要判斷 q(1,-2) 是否在直線上，也是將資料抽象出來以變數來表示，程式如下：（if 請看下一章）

```
a=3;b=4;c=5
m=1;n=-2
if (a*m+b*n+c)==0:
    print("yes")
else:
    print("no")
```

以上 3x+4y+5=0 是人類在數學上書寫的約定，電腦只要給 a,b,c，我們就約定此爲 ax+by+c=0，此稱爲資料的抽象化與變數的設計。

範例 3-6b

寫一個程式，可以輸入一個一元二次方程式，並求其解（本例假設所輸入的方程式恰有二解）。

程式設計步驟

1. 資料的數位化與變數設計。設有一元二次方程式如下：

```
ax² + bx + c = 0
```

本例指派 a,b,c 三個整數如下：

```
a=2;b=-7;c=3
```

即表示此爲方程式 $2x^2-7x+3=0$。

2. 寫出演算法。

國中解一元二次方程式是先用配方法，配方法需要一點判斷，可減少計算。但是本範例使用公式法，公式法雖計算較多，但完全不用判斷，最適合由電腦完成。這種強調多計算少判斷，就是電腦與人類不同的運算思維，人類計算能力較差，所以希望使用一些技巧來簡化計算，但是電腦則是計算能力強，判斷能力差，所以強調用計算來簡化程式。解一元二次方程式的公式法，演算步驟如下：

(1) 令 d= $\sqrt{b^2-4ac}$ 。提示：此爲數學語言，以 Python 語言表示則是 d=(b*b-4*a*c)**(1/2) ，括號、乘號均不能省。)

(2) 則其二解分別爲 $x_1=\frac{-b+d}{2a}$ ， $x_2=\frac{-b-d}{2a}$ 。

（提示：此爲數學語言，以 Python 語言表示則是 x1=(-b+d)/(2*a)，括號、乘號均不能省。)

(3) 例如，$2x^2-7x+3=0$。其解爲 $x_1=3.0$,$x_2=0.5$。

3. 以程式語言完成演算法。本例程式參考程式如下：

```
a=2;b=-7;c=3
d=(b*b-4*a*c)**(1/2)
x1=(-b+d)/(2*a)
x2=(-b-d)/(2*a)
print(x1)#3.0
print(x2)#0.5
```

執行結果

```
3.0
0.5
```

補充說明

1. 本例先假設所輸入的係數一定有實數解，待下一章再判斷所輸入係數有沒有實數解。
2. 電腦運算思維：人有人的特性與優勢，電腦有電腦的特性與優勢，所謂電腦運算思維，就是要學習電腦有哪些特性與優勢，然後使用電腦的運算思維寫程式。以本範例爲例，人類可能用配方法較快，但是電腦是使用公式法較快，此即爲電腦的運算思維，往後各章還有更多電腦運算思維，學習這些運算思維，就可增加程式設計功力，請繼續研讀以下各章，即可瞭解。

自我練習

1. 寫一個程式，可以輸入一個二元一次方程式，並求其解（本例假設所輸入的方程式恰有一解）。

運算思維

解二元一次方程式，國中會先教代入消去法與加減消去法，這都需要一點判斷，但可以簡化計算。本題使用公式法，可完全不用判斷，直接計算而得，這樣最適合計算機了，此也是電腦的運算思維。解二元一次方程式的克拉馬公式演算過程如下：

(1) 假設二元一次方程式如下：

$a_1x + b_1y = c_1$

$a_2x + b_2y = c_2$

(2) 指派或輸入 a1,b1,c1,a2,b2,c2 六個整數。（本例假設整係數方程式）

(3) 令 $d = \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix} = a_1b_2 - a_2b_1$。

(4) 則其解分別是 $x = \dfrac{\begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix}}{d} = (c_1b_2 - c_2b_1)/d$　　$y = \dfrac{\begin{vmatrix} a_1 & c_1 \\ a_2 & c_2 \end{vmatrix}}{d} = (a_1c_2 - a_2c_1)/d$

(5) 例如：

3x+y=5

x-2y=-3

則其解爲 x=1 y=2

計數器

日常生活常需要計數，例如，景點通常有一個計數器，累計今天進園人數。程式設計也不例外，常常需要統計某些敘述的執行幾次，此件事由電腦做的程式如下：首先，先宣布與清除計數值變數。

```
q=0;
```

計數值加 1 的程式如下：

```
q=q+1;
print(q)
```

_____

請鍵入以下程式，並觀察與寫出執行結果：

```
q=0
a=8
b=3
a=a-b
q=q+1
print(a,q)
a=a-b
q=q+1
print(a,q)
```

_____

_____

累加器

生活上也常面對累加問題，程式設計的累加程式如下：

```
a=2
s=0
s=s+a
print(s)
s=s+a
print(s)
```

自我練習

1. 請鍵入以下程式，寫出執行結果。

題號	程式	執行結果
1	a=21 b=9 r=a%b print(r) a=b b=r r=a%b print(a) print(b) print(r)	
2	a=6 n=2 r=0 d='' r=a % n d=str(r)+d print(d) a=a//n print(a) r=a%n d=str(r)+d print(d) a=a//n print(a) r=a%n d=str(r)+d print(d)	

3	a=542 s=0 n=0 r=a%10 n=n+1 s=s+r a=a//10 r=a%10 n=n+1 s=s+r a=a//10 r=a%10 n=n+1 s=s+r print(s) print(n)	
4	a,b="12","34" d="" d=d+a print(d) d=d+b print(d) d=b+d print(d) d=a+d print(d)	

3-7 程式的除錯（Debug）

程式除錯是所有程式設計者的惡夢。程式設計常見的錯誤是指令拚字錯誤、演算法錯誤等，分別說明如下：

拚字錯誤

拚字錯誤是初學者最容易發生的錯誤，但是現在很多整合開發環境已經將程式解析，給予提示。圖 3-1 是 Spyder 的解譯指令畫面，圖 3-2 是 Python 官版解譯指令畫面，兩者都能將指令、函式、常數解析給予不同的顏色。所以當鍵入程式並打完關鍵字時，若沒有變色，此時就要馬上更正，直到指令變色爲止。 Spyder 更強的是，還會給指令提示，幫忙補冒號、將

對稱的括號換色。例如：圖 3-2「as」前已經出現錯誤訊息，Spyder 爲其標示顏色，表示「as」是保留字，不能拿來當識別字。

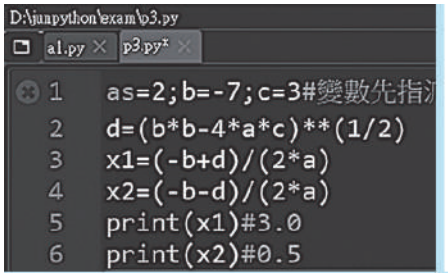


圖3-1 Spyder解譯指令畫面

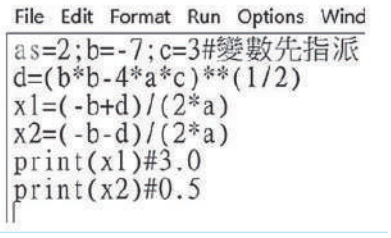


圖3-2 Python IDLE解譯指令畫面

演算法錯誤

若程式沒有錯誤訊息，結果卻錯誤，此時就要一步一步執行與觀察變數的變化。觀察的方法有兩種，分別是自行使用 print() 方法與使用整合編輯環境的 Debug 功能，分別說明如下：

▶ print()方法

在程式中，我們可以自行使用 print() 輸出變數的內容。例如，程式原本如下：

```
a,b,c=eval(input("input a,b,c:"))
d=(b*b-4*a*c)**(1/2)
x1=(-b+d)/(2*a)
x2=(-b-d)/(2*a)
print(x1)#3.0
print(x2)#0.5
```

我們可以中途先輸出變數內容 a,b,c,d，這樣可以縮小範圍，找出程式哪裡錯。

```
a,b,c=eval(input("input a,b,c:"))
print(a);print(b);print(c)
d=(b*b-4*a*c)**(1/2)
print(d)
x1=(-b+d)/(2*a)
```

```
x2=(-b-d)/(2*a)
print(x1)#3.0
print(x2)#0.5
```

▶ Debug

現今大部分整合開發環境都有 Debug 功能，Spyder 也不例外，點選 Spyder 功能表的「Debug/Debug」，畫面如圖 3-3，請留意第 1 行敘述出現箭號。

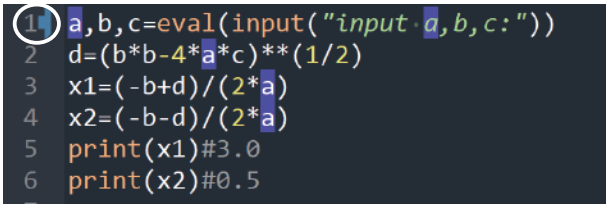


圖3-3　Spyder Debug畫面

逐一點選工具列的「Run current line」，即可逐一執行箭號所在敘述。圖 3-4 是執行 input() 且輸入資料後，於輔助說明區點選「Variable explorer」，此窗格會自動出現變數內容，如圖 3-5，此時即可觀察每個變數的內容。

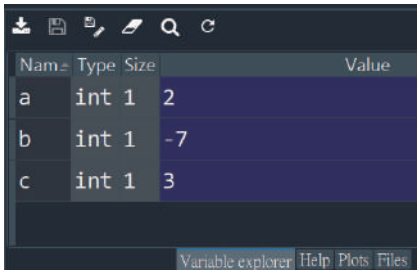


圖3-4　變數值窗格（一）

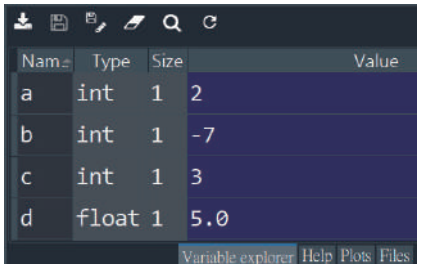


圖3-5　變數值窗格（二）

自我練習

1. 請將上一節的自我練習，以 Debug 追蹤每一個變數的結果。
2. 請同學們兩個人一組，每個人將自己的程式設定 2 個錯誤，然後交換電腦，由對方更正這些錯誤。

3-8 本章內容摘要

1. 我們平常處理的資料依其型態可分爲整數、浮點實數、字元、字串、與布林值。
2. Python 的運算子分爲四大類，分別是算術運算子、複合指派運算子、關係運算子、邏輯運算子。
3. 優先順序用來決定同一式子擁有多個運算子時，每一個運算子進行運算的優先順序。
4. 結合律則決定了在同一敘述中，相鄰的運算子有相同優先順序的運算子執行的順序。
5. Python 算術運算子有次方 (**)、乘法 (*)、除法 (/)、整數除法 (//)、取餘 (%)、加法 (+)、減法 (-)、指派 (=) 等。
6. Python 關係運算子有 <、>、<=、>=、==、!= 。
7. Python 邏輯運算子有 not、and、or。
8. abs() 是絕對值函式，int() 可將字串轉數值，str() 可將數值轉字串，round() 可將數值取四捨五入。
9. Python 變數的宣告語法只要宣告其初值如下：

```
a=0
```

10. Python 變數可同時交換。例如：

```
a,b=b,a
```

11. Python 的註解有兩種方式。第一，使用『"""』當註解開頭，直到遇到『"""』，兩個『"""』中間的文字通通是註解。第二，同一列中，井號『#』後面的也視爲註解。
12. Python 產生亂數的方法如下：

```
import random
a=random.randint(1,6)#產生1～6整數亂數
```

13. Python 數值相加與字串串接分別如下：

```
a=3;b=4
print(a+b)  #數值相加_________
print(str(a)+str(b))  #字串串接_________
```

14 字串可用串列直接取子字串，串列將於第 6 章介紹。例如：

```
a='0123'
print(a[0])  #_______
```

15. map() 與 split() 函式可協助字串分割。例如：

```
a="112/3/2"
y,m,d=a.split('/')            #得到字串型態
y,m,d=map(int,a.split('/'))    #得到數值型態
```

16. 要讓電腦發聲，要先載入 ctypes 類別，且取 ctypes.windll.kernel32 物件。例如：

```
import ctypes
p = ctypes.windll.kernel32
p.Beep(523,200)
```

17. Python 使用 print() 函式輸出，使用 input() 函式輸入。
18. 計數器的程式如下：

```
p=0；  p=p+1
```

19. 累加器的程式如下：

```
a=2；    s=0
s=s+a
```

課後評量

一、填充題（請寫出以下程式執行結果）

題號	題目	輸出結果
1	a=0x3b; print(a)	
2	a=3.9E3 print(a)	
3	a=8;b=3;c=2 print(a**b) print(a//b) print(a/b) print(a%b) print(a**(1/3)) print((1/2)**2) print(5**-2)	
4	a=3;s=0 s+=a print(a) a=a-1 print(a)	
5	a=162 b=a%10 print(b) a=a//10 print(a)	
6	print(3==2) print(4!=2)	
7	print(3>=2 or 4==5)	
8	print(not(3>=2 and 4==5))	
9	print(9**(1/2)) print(9**1/2) print(1/3**2) print(1/2**2/4)	
10	print(3-2-2) print(2**2**3)	

11	money=3 momey=money+3 print(money)	
12	a=3;b=4 a,b=b,a print(a)	

13	a=3
	#a=a+1
	print(a)	

14	a=123;b=456 print(a+b) print(str(a)+str(b))	
15	a='123' print(a+a) print(a*3) print(int(a)+2)	
16	a='1234' print(len(a)) print(a[0]+a[1]) print(int(a[0])+int(a[1]))	
17	a='東西南北' print(len(a)) print(a[0]+a[1])	
18	a="112/11/2" y,m,d=a.split('/') print(y+m+d) y,m,d=map(int,a.split('/')) print(y+m+d)	
19	print('aaa',end='') print('bbb')	
20	a,b,c=1,2,3 print("123456789012345678 90") print("a=%2d,b=%3d,c=%4d" %(a,b,c))	
21	a=65 print("%d"%a) print("%c"%a)	
22	a="老師";b=56 print("帥哥%s,年齡是%d" %(a,b))	

23	a=eval("3+4**2//4") print(a)	
24	s=0 s=10*s+1 s=10*s+2 s=10*s+3 print(s)	
25	x1=3;y1=0 x2=0;y2=-4 d=((x1-x2)**2+(y1-y2)**2)**(1/2) print(d)	
26	a,b,c=3,4,5 d=(a+b+c)/2 s=(d*(d-a)*(d-b)*(d-c))**(1/2) print(s)	
27	x1=0;y1=0;x2=3;y2=0;x3=0;y3=4 r=abs((x1*y2+x2*y3+x3*y1)- (x2*y1+x3*y2+x1*y3))/2 print(r)	
28	a=3;b=4;c=5 m=4;n=1 d=abs(a*m+b*n+c)/((a*a+b*b)**(1/2)) print(d)	
29	a=1;b=5;c=-14 d=(b*b-4*a*c)**(1/2) x1=(-b+d)/(2*a) x2=(-b-d)/(2*a) print(x1) print(x2)	
30	a1,b1,c1=3,1,5 a2,b2,c2=1,-2,-3 d=a1*b2-a2*b1 x=(c1*b2-c2*b1)/d y=(a1*c2-a2*c1)/d print(x,y)	

本章習題

1. 本章已經介紹任意 3 點所圍的面積公式如下：

$$=\frac{1}{2}\begin{vmatrix} x_1 & x_2 & x_3 & x_1 \\ y_1 & y_2 & y_3 & y_1 \end{vmatrix}$$ 其中↘乘為正，↙乘為負

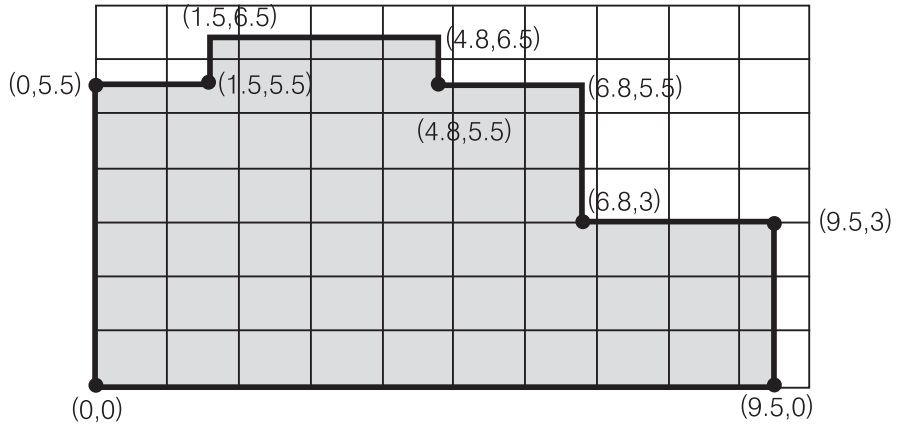
$$=\frac{1}{2}[(x_1y_2+x_2y_3+x_3y_1)-(x_2y_1+x_3y_2+x_1y_3)]$$

任意凸多邊形面積可推廣如下：

$$\text{多邊形面積}=\frac{1}{2}\begin{vmatrix} x_1 & x_2 & \cdots & x_n & x_1 \\ y_1 & y_2 & \cdots & y_n & y_1 \end{vmatrix}$$ 其中↘乘為正，↙乘為負

請依此公式求 (0,0),(5,0),(3,4),(2,4) 的面積。

2. 實例探討。請於 Goole 地圖下載自己學校的地圖，將學校的角落都座標化，然後計算學校面積。例如，以下是某學校的地圖，於左下角指派為座標 (0,0)，將學校等距離畫水平、垂直方格直線，估計每個點的座標，如下圖：



蒐集以上座標 (0,0),(9.5),(9.5,3),(6.8,3),(6.8,5.5),(4.8,5.6),(4.8,6.5),(1.5,6.5),(1.5,5.5),(0,5.5)，代入以上公式，再乘以比例尺比例的平方，即為所求。

3. 日幣的買入。日幣的匯率每天都在變動，請寫一個程式，可以輸入一個匯率、輸入一個台幣金額，得到一筆日圓金額。

4. 同上第 3 題，但是日鈔最小金額是千元，請將日圓千元以下捨棄，再換算回來台幣金額。例如，匯率當日若是每 1 元新台幣可換 4.443 元日圓，表示 10000 元新台幣可換 44430，但日鈔最小的面額是 1000 元，表示僅可換 44000 元日幣，然後請再換算回來台幣多少錢。

5. 同上第 4 題。銀行日鈔僅提供萬元、5 千元與千元紙鈔，請寫一個程式，可以將以上日幣換算各種面額鈔票的張數。
6. 銀行利息計算。假設銀行年利率是百分之 1.2，請寫一個程式，可以輸入定存金額，計算每年利息。
7. 營業稅的計算。營業稅若外加，則營業稅 = 定價 *0.05，若內含，則營業稅 = 定價 /21，且都四捨五入到整數。請寫一程式，可以輸入定價，並分別計算稅外加、與稅內含的營業稅。

Chapter
04

選擇結構程式設計

學習綱要

學習表現

1. 具備撰寫程式語言基本能力，運用資訊科技方法解決問題。
2. 具備程式設計之邏輯思考能力，展現系統思考、分析與探索之素養。
3. 具備實作程式設計能力，展現程式語言跨域應用之軟實力。

4-1 結構化程式設計架構

任何程式皆可由循序結構、選擇結構與重複結構等三種結構組合起來，且避免使用 Goto 指令，此稱爲「結構化程式設計架構」。以上三種結構分別說明如下：

循序結構

程式執行順序與程式出現順序完全相同者，稱爲循序結構。例如，本書目前爲止的所有程式都稱爲循序結構。

選擇結構

程式執行順序可以某些條件而改變者，此稱爲選擇結構，例如，肚子餓了就執行「吃飯」，否則執行「繼續工作」等。Python 的選擇結構有「if ~else」、「if ~elif~else」此爲本單元將介紹的內容。

重複結構

我們人類很多事都具有重複性。例如，重複做 10 個題目，重複連續上七節課等，程式設計是用來解決人類問題的工具，當然也有此重複結構。Python 的重複結構有 for 與 while 等指令，兩者的主要差別主要是——若程式設計階段就知道執行次數，可使用 for，請看 5-2 節；若於程式設計階段不知道執行次數，要等到程式執行階段，才能依照執行結果判斷何時結束迴圈，此時可用 while，請看 5-4 節。

4-2 單一選擇結構敘述與撰寫

if～else ～

生活上常會面對『假如～否則～』，Python 的指令是『if~else~』，其語法如下：

```
if運算式:
    敘述區塊1；
```

```
[else :
    敘述區塊2；]
```

請特別留意冒號『:』與『程式縮排』，因爲冒號『:』是 Python 特有;『縮排』在其他語言是美觀性質，但 Python 是語法也就是規定，因爲縮排才是屬於 if else 要執行的敘述子區塊（若使用 Spyder 編寫程式，Spyder 會幫您補上冒號與縮排，若編譯器沒有幫您自動縮排，請按一下鍵盤的「Tab」鍵，而不是刻意按四個空白鍵。其次，縮排的空格數沒有強制性，但是在同一檔案中有相同空格數的強制性。），以上程式其流程圖如圖 4-1：

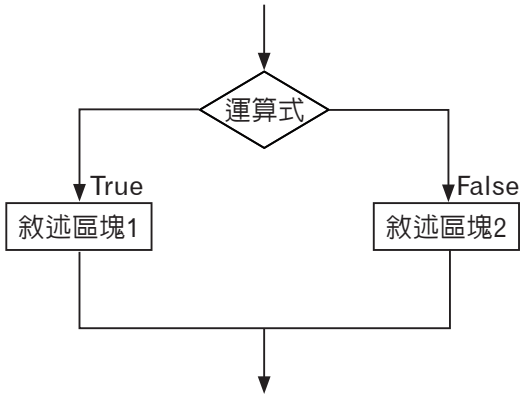


圖4-1　if ～ else~流程圖

例如：

```
a=66
if a>=60:
    b="Pass"
else:
    b="Fail"
print(b)
```

以上程式一定要縮排。因爲，程式縮排在其它語言是美觀問題，但 Python 是語法，所以沒有縮排就不行。例如，以下程式就不行。

```
a=66
if a>=60:
b="Pass"
```

```
else:
b="Fail"
print(b)
```

但若是僅執行一個敘述，這樣也行，但是視覺效果很差。

```
a=55
if a>=60:b="Pass"
else:b="Fail"
print(b)
```

請留意，以下兩個程式的 print(b)，因爲縮排不同，歸屬就不同，執行結果也不相同。

```
a=66
if a>=60:
    b="Pass"
else:
    b="Fail"
print(b)
```

```
a=66
if a>=60:
    b="Pass"
else:
    b="Fail"
    print(b)
```

C/C++ 用大括號『{}』來表示程式區塊的範圍，Python 直接用縮排表示，那就更簡潔。其次，也可以將否則的部分放在前面當預設值，程式如下：

```
a=46;b="Fail"
if a>=60:
    b="Pass"
print(b)
```

或

```
a=66;b="Fail"
if a>=60:b="Pass"
print(b)
```

if 內可否放 if，當然可以，此稱爲巢狀 if，請看 4-4 節。

冒號(:)

冒號 (:) 也是 Python 特有的符號，往後的 for、while、函式等，都需要

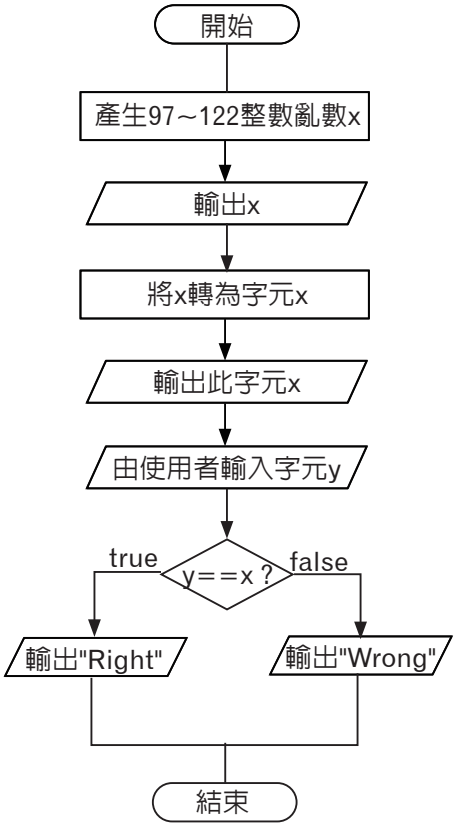
這個符號。

範例 4-2a

請寫一個程式，可以產生一個小寫字元，由使用者輸入此小寫字元，電腦評判是否正確。

運算思維

1. 本例以流程圖分析如下：



2. 由以上流程圖分析，我們發現 if~ else~ 剛好可解決此問題。
3. 電腦產生字元、使用者如何輸入字元，前面已經說明，再套上 if~else~ 語法，所以全部程式如下：

```
import random
x=random.randint(97,122) #產生97～122的整數
print(x)#輸出此數字
```

```
x=chr(x)#轉爲此數字所代表字元
print(x)
y=input('input the char:')
if y==x:
    print("Right")
else:
    print("Wrong")
```

執行結果

```
119
w

input the char:w
Right
```

自我練習

1. 請寫一程式，電腦自動產生 1~6 的整數亂數，並判斷是奇數或是偶數。
2. 直線。直線標準式爲 ax+by+c=0，請寫一程式，可以指派一直線係數 a,b,c。其次，可再輸入任一點座標，並判斷所輸入點是否在直線上。例如，指派 a,b,c 分別是 1,2,-4，那方程式就是 x+2y-4=0，其次，輸入點若是 (2,1)，那就是在直線上，若是 (1,2)，那就不在直線上。
3. 請寫一程式，可以自動產生一個小寫字元，請判斷其是否爲母音。說明，字元 a,e,i,o,u 稱爲母音，其餘爲子音。
4. 心算練習。請寫一個程式，可分別產生 2 個 1 位數整數亂數，由使用者輸入相加結果，電腦評判是否正確。
5. 音感練習。請寫一個程式，由電腦產生 1 ～ 8 的亂數，依照亂數發出 Do、Re、Mi 到高音 Do 的音，並由使用者輸入 1~8，電腦評判是否正確。

4-3 多重選擇結構敘述與撰寫

match case

大部分程式語言都有 switch case 解決多重選擇的決策問題，早期 Python 沒有 switch case 指令，但到了 Python 3.10 版本時，也順應潮流，使用 match case 解決多重選擇分歧問題，match case 簡易語法如下：

```
a=數值、字元或字串
match a:
    case 數值1、字元1或字串1:
        敘述區塊1
    case 數值2、字元2或字串2:
        敘述區塊2
    case …
    case _ :#以上沒有符合的條件時，自動執行此敘述區塊
        敘述區塊
```

例如，以下程式，可將 1,2,3,4 轉爲春、夏、秋、冬。

```
a=1
match a:
    case 1:
        b='春'
    case 2:
        b='夏'
    case 3:
        b='秋'
    case 4:
        b='冬'
    case _:
        b='輸入錯誤'
print(b)
```

請留意以上程式僅能在 Python 3.10 以後版本執行，且同 if 指令，請特別留意縮排與冒號問題。若沒有一個條件符合，就有可能空手而回，因爲 b 沒有宣告而輸出，導致程式錯誤，如以下程式。

```
a=3
match a:
    case 1:
        b='春'
    case 2:
        b='夏'
print(b)
```

應該在 match 前面先宣告變數「a=3;b=' 輸入錯誤 '」，或在 match 裡面加入以下程式：

```
 case_ :
    b=' 輸入錯誤'
```

範例 4-3a

請寫一個程式，完成以下要求：

1. 輸入一個 0~100 的分數。
2. 當分數大於 90 分時，輸出 A。
3. 當分數介於 80~89 時，輸出 B。
4. 當分數介於 70~79 時，輸出 C。
5. 當分數介於 0~69 時，輸出 D。

運算思維

1. 此一問題可先將分數除以 10 得到一個整數商，此整數商的結果僅剩 0~10 的 11 個整數，所以每個分數同時有 11 個選擇，此即爲多重選擇問題，可以適用 match case 解決此問題。
2. 當不同的選擇條件，有相同的結果，可將這些選擇以「|」共用結果，例如，9 與 10 都是得到「A」，可以「9|10」表示，所以全部程式如下：

```
a=int(input("input a grade: "))
a=a//10; b='輸入錯誤'
match a:
    case 9|10:
        b='A' #大於等於90分爲A
```

```
    case 8:
        b='B' #大於等於80分且小於90分爲B
    case 7: #大於等於70分且小於80分爲C
        b='C'
    case 6|5|4|3|2|1|0:
        b='D'
    case _:
        b='輸入錯誤'
print(b)
```

執行結果

```
input a grade: 98
A
```

自我練習

1. 請寫一程式，將所輸入的 0、1、2…6，轉爲 ' 星期日 '、' 星期一 '…' 星期六 ' 等字串。
2. 請寫一個程式，可以產生一個 0 到 25 的亂數，且依以下分數顯示燈號

```
21～25：五個燈。
16～20：四個燈。
11～15：三個燈。
6～10：兩個燈。
1～5：一個燈。
0：零個燈。
```

範例4-3b

猜拳遊戲。請寫一個程式，可以由人和電腦猜拳，並評定勝負。

運算思維

1. 這一任務就是寫人工智慧程式的入門了，寫程式前先想一下您和一個不懂猜拳的人猜拳，那您們如何猜拳與評判勝負？首先，我們先規定有三種拳，分別是『剪刀、石頭與布』，每人每次僅能出一種拳法，且定義『剪刀贏布，石頭贏剪刀、布贏石頭，兩者相同則平手等』。『人工智慧』就是要把這些規定以程式語言表示，並由以上規定來評判勝負。

2. 資料的數位化與變數的設計。我們人類猜拳是直接用手勢表示『剪刀、石頭與布』，但是如何讓電腦瞭解您的拳法呢，可以將以上『剪刀、石頭與布』以『1,2,3』表示，此即爲資料的數位化。因爲用『1,2,3』表示，只要 1byte 就可以，若您用『剪刀、石頭與布』表示，當然也可以，但一個中文字就佔用 2byte，且往後的資料處理也比較複雜。
3. 電腦同時有三種拳法，所以適用 match，人也同時有三種拳法，也適用 match，所以共有 9 種情況。
4. 您要讓電腦也能思考，也就是將以上九種情況的結果，以程式語言先規定好，所以程式如下：。

```
import random
a=int(input("input 1,2,3:"))#記得要用int()轉型
b=random.randint(1,3)#產生1～3的整數亂數
astr='' #a代表的拳法
bstr='' #b代表的拳法
r=''    #猜拳結果
match a:#people
    case 1:
        astr='剪刀'
        match b: #computer
            case 1:
                bstr='剪刀'
                r='平手'
            case 2:
                bstr='石頭'
                r='computer win'
            case 3:
                bstr='布'
                r='people win'
    case 2:
        astr='石頭'
        match b:
            case 1:
                bstr='剪刀'
                r='people win'
            case 2:
                bstr='石頭'
```

```
                r='平手'
            case 3:
                bstr='布'
                r='computer win'
    case 3:
        astr='布'
        match b:
            case 1:
                bstr='剪刀'
                r='computer win'
            case 2:
                bstr='石頭'
                r='people win'
            case 3:
                bstr='布'
                r='平手'
print("您出 %s,電腦出 %s,結果是 %s" % (astr,bstr,r))
```

執行結果

```
input 1,2,3:2
您出 石頭,電腦出 剪刀,結果是 people win
```

補充說明

這一題輸入資料時，若忘了將資料轉爲數值，如以下程式，因爲沒有錯誤訊息，但就是沒結果。

```
a=input("input 1,2,3:")#請留意a是字串型態
```

自我練習

1. 請寫一個程式，讓三個人同時猜拳。本例由電腦產生兩個亂數，由您和電腦猜，且評判輸贏。（提示：兩個人共 9 種情況，用兩層決策，三個人共 27 種情況，可用 3 層決策）

4-4 巢狀選擇結構敘述與撰寫

選擇結構內還有選擇結構稱爲巢狀選擇結構，請看以下範例說明。

範例4-4a

同範例 4-3a，但改用 if else 巢狀迴圈重做。

執行結果

```
input a grade: 88
the grade is B
```

運算思維

1. 前面我們已經用 match case 完成此問題，但也可以使用巢狀迴圈，程式如下：

```
a=int(input("input a grade: "))#請留意input傳回字串型態
if(a>=90) : #大於等於90分爲A
    r='A'
else :   #迴圈內允許還有迴圈
    if(a>=80):#大於等於80分且小於90分爲B
        r='B'
    else :
        if(a>=70):
            r='C'
        else :
            r='D'
print("the grade is %c" % r)
```

2. 以上逐漸縮排，有時太多的縮排，程式容易產生斷行，不易閱讀，所以也可以使用 if~elif~ 代替，程式如下：

```
a=int(input("input a grade: "))#請留意input傳回字串型態
if(a>=90) : #大於等於90分爲A
    r='A'
elif(a>=80):#大於等於80分且小於90分爲B
    r='B'
elif(a>=70):
```

```
    r='C'
else :
    r='D'
print("the grade is %c" % r)
```

3. 以下程式就不行。因爲 if-elif-else 語法在進行條件比對時是由上往下逐一比對，只要一有條件滿足就進入該對應區塊執行，執行完畢後不會對於下方的剩餘條件繼續比對，所以比對條件的順序需要特別注意，這是許多學生在學習 if-elif-else 語法時常犯的錯誤。

```
a=int(input("input a grade: "))#請留意input傳回字串型態
if(a>=70) : #大於等於70分爲C
    r='C'
elif(a>=80):#大於等於80分爲B
    r='B'
elif(a>=90):
    r='A'
else :
    r='D'
print("the grade is %c" % r)
```

4. 考試成績的分布，通常中間分數的人較多，此時可以使用二分法調整執行順序，程式如下，這樣不管分數爲何，至多比較 2 次，所以可以減少比較次數，提升程式執行效率，程式如下：

```
a=70
if a>=80:
    if a>=90:
        b='A'
    else:
        b='B'
else:
    if a>=70:
        b='C'
    else:
        b='D'
print(b)
```

自我練習

1. 以範例 4-4a 爲例，請將「a=int(input("input a grade: "))」改爲「a=input("input a grade: ")」，並觀察執行結果。
2. 請寫一程式，電腦自動產生 -5~5 的整數亂數，並判斷是正數、0 或負數。
3. 某一貨品定價 100 元，若購買 10 件 (含) 以上打 9 折，若購買 30 ～ 99 件則打 8 折，若購買 100 件以上則打 7 件，試寫一程式可以輸入購買件數而得總價。測試資料如下：

輸入	輸出
5	500
10	900(100*10*0.9)
30	2400(100*30*0.8)
100	7000(100*100*0.7)

4. 請寫一個程式，可以產生一個 0 到 25 的亂數，且依以下分數顯示燈號

   21 ～ 25：五個燈。
   16 ～ 20：四個燈。
   11 ～ 15：三個燈。
   6 ～ 10：兩個燈。
   1 ～ 5：一個燈。
   0：零個燈。

5. 同範例 4-4a，但程式一執行，要求先輸入密碼，密碼若爲『abcd』，那才可執行本程式。
6. 同範例 4-4a，但程式一執行，要求先輸入密碼，密碼若爲『aa11』、『bb22』、『cc33』或『abcd』，才可執行本程式。

範例 4-4b

請寫一個程式，可以判斷所輸入座標的所在象限。爲了簡化問題，本例先不考慮 x ＝ 0 或 y ＝ 0 的座標軸與原點，而是留到本範例自我練習，再補齊程式。

運算思維

當 x>0 時，還要繼續以 y 判斷在第 1，或第 4 象限，此即爲選擇內還要選擇，所以可套用巢狀選擇，程式如下：

```
x,y=eval(input("input x,y:")) #數字請用逗號『,』隔開，例如輸入 3,4
if x>0 :
    if y>0:
        b="I"
    else:
        b="IV"
else:
    if y>0:
        b="II"
    else:
        b="III"
print(b)
```

執行結果

```
input x,y:3,-2
IV
```

補充說明

1. 請留意 eval() 傳回數值型態。
2. 此題目有人會寫成

```
x,y=eval(input("input x,y:")) #數字請用逗號『,』隔開，例如 3,4
if (x>0 and y>0) :
    b="I"
if (x<0 and y>0) :
    b="II";
```

```
if (x<0 and y<0) :
    b="III"
if (x>0 and y<0 ):
    b="IV"
print(b)
```

這樣雖然沒有錯，但是執行效率非常差，因爲電腦要不斷的比較。

自我練習

1. 同上範例 4-4b，但增加先判斷是否在原點或 x、y 軸上。測試資料如下

輸入(x,y)	輸出
(0,0)	原點
(0,3)	y 軸
(3,4)	I

4-5 選擇結構程式實作演練

極大與極小

極大與極小是日常資料處理最常見的問題，以下我們先介紹 3 筆資料的極大值的求法，4~5 筆資料的極大或極小請自行擴充。6 筆以上資料，程式就冗長了，資料就需要以串列儲存，並以迴圈撰寫，待學完串列就能理解迴圈與串列的特異功能。

範例 4-5a

假如有 3 筆資料如下：

3,8,2,

請寫一程式，找出極大值。

設計步驟

1. 資料的數位化與變數設計。本例以單一變數儲存如下：

```
a=3;b=8;c=2
```

2. 寫出演算法則。

(1) 設定極大值 (max) 爲第一數。

```
max=a
```

(2) 當第二數 (b) 大於極大值時，極大值即以 b 取代。

```
if (b>max):
    max=b
```

(3) 當第三數 (c) 大於極大值時，極大值即以 c 取代。

```
if (c>max):
    max=c
```

(4) 輸出極大值。

```
print(max)
```

程式列印

```
a=4;b=5;c=1
max=a
if b>max:
    max=b
if c>max:
    max=c
print(max)
```

執行結果

```
5
```

補充說明

1. 以上是求極大值的演算法，但 Python 已經有 max() 函式，所以本例也可以寫成如下：

```
a=4;b=5;c=1
max=max(a,b,c)
print(max)
```

自我練習

1. 假如有 4 筆資料如下：

   4,8,2,5

   如何找出極大值。
2. 有一批資料，含有 3 個人的人名與成績，請寫一程式，可以找出最低分的人名與成績。例如：

```
aa,60
bb,30
cc,80
```

3. 請寫一程式，可以指派 5 個數值，請去掉最大值與最小值，再求其平均。

排序

將資料由小而大、或由大而小排列稱爲排序。

氣泡排序法

爲了充分理解氣泡排序的原理，我們先逐一探討 2 筆、3 筆、4 筆、5 筆資料的排序，這樣循序漸進，就能找出規律性，等待學習迴圈與串列才能很快寫出程式。首先，我們先探討 2 筆資料，若有 2 筆資料如下：(備註：所有的排序程式，測試資料一定要用最遭的情況，也就是要與排序後完全相反，這樣才能觀察排序是否完成。)

```
5  4
```

我們只要比較 1 次就好。若第 1 筆大於第 2 筆，則 2 者交換，排序完成。程式如下：

```
a=5;b=4
if a> b:
    a,b=b,a
print(a,b)
```

以上程式，比較過程如下表：

階次	5	4	比較方式	比較次數
1	4	5	(5,4)	1

若有 3 筆資料如下

```
5  4   3
```

則需要兩個階次，程式如下：

```
a=5;b=4;c=3
#階次1，比較2次
if a> b:
    a,b=b,a
if b>c:
    b,c=c,b
print(a,b,c)#c 此時最大，不用再比較
#階次2，比較1次
if a> b:
    a,b=b,a
print(a,b,c)
```

以上程式，比較過程如下表：

階次	5	4	3	比較方式	比較次數
1	4	3	5	(5,4)(5,3)	2
2	3	4		(4,3)	1

若有 4 筆資料，共需要 3 個階次，排序方法如下：

```
a=5;b=4;c=3;d=2
#階次1，比較3次
if a> b:
    a,b=b,a
if b>c:
    b,c=c,b
if c>d:
    c,d=d,c
print(a,b,c,d)#d 已經最大，不用再比較
```

```
#階次2，比較2次
if a> b:
    a,b=b,a
if b>c:
    b,c=c,b
print(a,b,c,d)#c 已經次大，不用再比較
#階次3，比較1次
if a> b:
    a,b=b,a
print(a,b,c,d)#排序完成
```

以上程式，比較過程如下表：

階次	5	4	3	2	比較方式	比較次數
1	4	3	2	5	(5,4)(5,3)(5,2)	3
2	3	2	4		(4,3)(4,2)	2
3	2	3			(3,2)	1

若有 5 筆資料，共需要 4 個階次，排序方法如下：

```
a=5;b=4;c=3;d=2;e=1
#階次1，比較4次
if a> b:
    a,b=b,a
if b>c:
    b,c=c,b
if c>d:
    c,d=d,c
if d>e:
    d,e=e,d
print(a,b,c,d,e)#e 已經最大，不用再比較
#階次2，比較3次
if a> b:
    a,b=b,a
if b>c:
    b,c=c,b
if c>d:
    c,d=d,c
print(a,b,c,d,e)#d 已經次大，不用再比較
#階次3，比較2次
```

```
if a> b:
    a,b=b,a
if b>c:
    b,c=c,b
print(a,b,c,d,e)
#階次4，比較1次
if a> b:
    a,b=b,a
print(a,b,c,d,e)
```

以上程式，比較過程如下表：

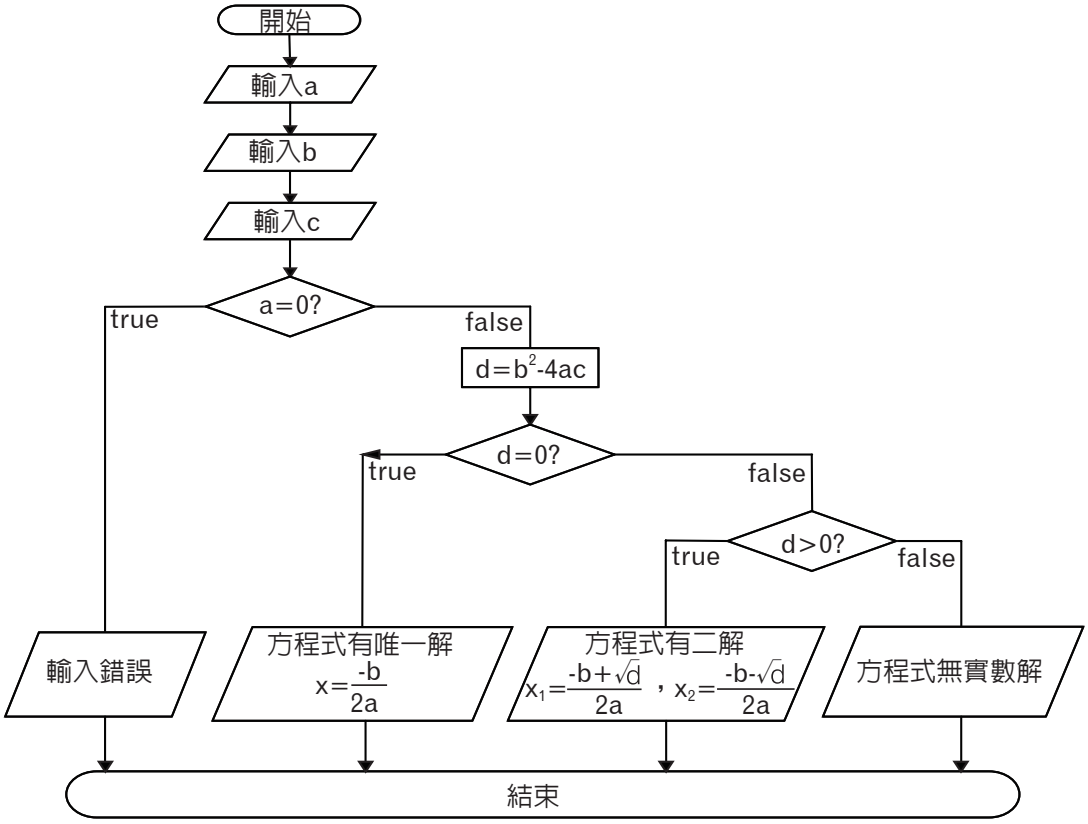
階次	5	4	3	2	1	比較方式	比較次數
1	4	3	2	1	5	(5,4)(5,3)(5,2)(5,1)	4
2	3	2	1	4		(4,3)(4,2)	3
3	2	1	3			(3,2)	2
4	1	2				(2,1)	1

以上 5 筆資料，程式就很冗長了，那如果 100 筆、1000 筆呢，程式是不是更冗長？所幸，答案是否定的，因爲後續的迴圈與串列就可解決此問題。

一元二次方程式

前面我們直接假設所輸入的一元二次方程式有解，但並不是隨便給三個係數，一個方程式就通通有解，本例則要加上判斷了。解一元二次方程式的演算法如下：

1. 設有一元二次方程式如下：
   $ax^2 + bx + c = 0$
2. 若 a=0 則輸出“輸入錯誤”。
3. 令 $d=b^2 - 4ac$。
4. 若 d=0，則方程式有唯一解 $x = \frac{-b}{2a}$；否則，若 d>0，則方程式有二解 $x_1 = \frac{-b+\sqrt{d}}{2a}$，$x_2 = \frac{-b-\sqrt{d}}{2a}$；否則，無實數解。以上演算分析，以流程圖說明如下：



範例 4-5b

請設計一個程式，可以解一元二次方程式 $ax^2 + bx + c = 0$。

程式列印

```
a,b,c=eval(input("input a,b,c:"))
if(a==0):
    print("input error")# 若a=0，則列印錯誤訊息
else:
    d=b**2-4*a*c# 計算d值 */
    if(d==0):
        print("only one answer,x= %d" % (-b/(2*a)))
    elif d>0 :
        d=d**(1/2)
        x1=(-b+d)/(2*a)
        x2=(-b-d)/(2*a)
        print("two answer,x1= %f,x2=%f"%(x1,x2))
    else:
        print("no real answer")
```

執行結果

```
input a,b,c:1,-5,6
two answer,x1= 3.000000,x2=2.000000
```

自我練習

1. 解二元一次方程式。解二元一次方程式的演算法如下：
   (1) 設二元一次方程式如下：

   $a_1x + b_1y = c_1$

   $a_2x + b_2y = c_2$

   (2) 令 $d = \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}$（表示 $d = a_1b_2 - a_2b_1$）

   (3) 假如 $\frac{a_1}{a_2} = \frac{b_1}{b_2} = \frac{c_1}{c_2}$，則方程式無限多解，且程式結束。（代表兩重疊直線）

   (4) 否則，假如 $\frac{a_1}{a_2} = \frac{b_1}{b_2} \neq \frac{c_1}{c_2}$，則方程式無解，且程式結束。（代表兩平行直線）

   (5) $x = \frac{\begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix}}{d} = (c_1b_2 - c_2b_1)/d$

   (6) $y = \frac{\begin{vmatrix} a_1 & c_1 \\ a_2 & c_2 \end{vmatrix}}{d} = (a_1c_2 - a_2c_1)/d$

2. 請設計一個程式，可以解二元一次方程式。

範例4-5c

三角形判斷（APCS105 第二梯次試題）

若已知三角形三邊長，判斷是否構成三角形、評定三角形種類與計算三角形面積的演算法如下：

(1) 輸入三角形的三邊長 a、b、c。

(2) 任兩邊之和要大於第三邊，才能繼續以下計算，否則輸出『無法構成三角形』，且程式結束。

(3) 若 c 是最長邊，假如 $a^2+b^2>c^2$ 則爲銳角三角形；否則，假如 $a^2+b^2=c^2$，則爲直角三角形；否則此三角形爲鈍角三角形。

(4) 令 $d=\frac{1}{2}(a+b+c)$

(5) 三角形面積 $=\sqrt{d(d-a)(d-b)(d-c)}$

請輸入三角形三邊長，首先判斷是否構成三角形、其次判別三角形的種類，最後計算其面積。

程式列印

任兩邊之和要大於第三邊的程式是「 if a+b>c and b+c>a and c+a>b :」，本例就將最大邊找出來，只要最小兩邊之和大於第三邊，這樣就一石二鳥，可以方便判斷是否形成三角形，也能判斷三角形種類。

```
a,b,c=eval(input("input a,b,c:"))
#以爲氣泡排序法將a,b,c三個數字排序
if a>b:
    a,b=b,a
if b>c:
    b,c=c,b
#c已經是最大
if (a+b>c)  :
    c2=c**2
    b2=b**2
    a2=a**2
    if ((a2+b2)>c2):
        t="銳角三角形"
    elif ((a2+b2)==c2):
        t="直角三角形"
    else:
        t="鈍角三角形"
    d=(a+b+c)/2
    area=(d*(d-a)*(d-b)*(d-c))**(1/2)#留意括號的對稱
    print(t)
    print(area)
else:
    print("無法構成三角形")
```

執行結果

```
input a,b,c:3,4,5
直角三角形
6.0
```

※ 範例4-5d

示範實作電子琴。

程式列印

1. 前面已經介紹電腦如何發音，本例希望按鍵盤「1」發「Do」，按「2」發「Re」，按「3」發「Mi」，以上即是選擇結構的應用，程式如下：
2. thinter 可實作視窗介面，不是本書介紹範圍，請在此先體驗就好。

```
from tkinter import *#載入視窗模組
import ctypes #載入發音模組
p = ctypes.windll.kernel32
def aa(e):
    k=e.char
    tk.title(k)
    if k=='1':
        p.Beep(523,200)#發Do音
    elif k=='2':
        p.Beep(587,200)
    elif k=='3':
        p.Beep(659,200)
    elif k=='4':
        p.Beep(698,200)
    elif k=='5':
        p.Beep(784,200)
    elif k=='6':
        p.Beep(880,200)
    elif k=='7':
        p.Beep(988,200)
    elif k=='8':
        p.Beep(1046,200)
tk=Tk()#產生視窗
tk.geometry("300x100+50+70")#視窗大小
```

```
eny1=Entry(tk)#產生輸入盒
eny1.bind('<Key>',aa)#綁定aa函式
eny1.pack()#版面配置
tk.mainloop()
```

執行結果

自我練習

1. 同範例 4-5d，但改爲以 match 重作。

範例4-5e

實例探討。郵局信件郵資計算。

說明

中華郵政普通信函郵資依照信件重量費用如圖 4-1。

圖4-1 中華郵政普通信件資費表

請寫一個程式，可以輸入重量，而得到資費。

程式列印

此爲巢狀選擇問題，程式如下：

```
w=12
m=0
if w<=20:
    m=8
elif w<=50:
    m=16
elif w<=250:
    m=40
elif w<=500:
    m=72
elif w<=1000:
    m=112
elif w<=2000:
    m=160
else :
    m="超重"
print(m)
```

執行結果

```
8
```

範例4-5f

勞動基準法。

勞動基準法第 38 條內文如下：

第 38 條：勞工在同一雇主或事業單位，繼續工作滿一定期間者，應依下列規定給予特別休假：

一、六個月以上一年未滿者，三日。

二、一年以上二年未滿者，七日。

三、二年以上三年未滿者，十日。

四、三年以上五年未滿者，每年十四日。

五、五年以上十年未滿者，每年十五日。

六、十年以上者，每一年加給一日，加至三十日爲止。

請寫一個程式，可以依照勞工任職年資，計算其特休天數。

程式列印

此也是巢狀選擇經典問題，程式如下：

```
#測試資料，年資
#a=0.5 #b=0
#a=0.6 #b=3
#a=0.7 #b=3
#a=1 #b=7
#a=9.5 #b=15
#a=10.1 #b=16
a=11.2 #b=17
b=0 #特休天數
if a<0.6 :
    b=0
elif a<1:
    b=3
elif a<2:
    b=7
elif a<3:
    b=10
elif a<5:
    b=14
elif a<10:
    b=15
else :
    b=15+(a-9)//1 #取最小整數
    #每超過1年，年休增加1天
    if b>=30: #超過30天，以30天計算
        b=30
print(b)
```

執行結果

```
17
```

4-6 本章內容摘要

1. Python 選擇結構有 if~else、match case、if~elif~else，且請留意要有冒號與縮排問題。
2. if~else 用法如下：

```
a=66
if a>=60:
    b="Pass"
else:
    b="Fail"
print(b)
```

3. match case 用法如下：

```
a=1 ;b=''
match a:
    case 1:
        b='春'
    case 2:
        b='夏'
    case _:
        b='輸入錯誤'
print(b)
```

請留意以上程式僅能在 Python 3.10 以後的版本執行。

4. 巢狀的 if~else~if~else 可用 if~elif~else 代替，用法如下：

```
a=3
if a<0 :
    re='負數'
elif a==0:
    re='Zero'
else:
    re='正數'
print(b)
```

但請留意條件的擺放方式與順序，避免執行錯誤與浪費時間。

課後評量

一、填充題（請寫出以下程式執行結果）

題號	題目	執行結果

1	a=3
	if a%2==0:
	　　b='偶數'
	else:
	　　b='奇數'
	print(b)	

2	a,b,c=1,2,-4
	x,y=2,1
	if a*x+b*y+c==0:
	　　print('在直線上')
	else:
	　　print('不在直線上')	

3	x='a' x=input('input a char:') if x=='a' or x=='e' or x=='i' or x=='o' or x=='u': 　　print('母音') else: 　　print('子音')	
4	a=20 match a: 　　case 0: 　　　　b='0' 　　case 1|2|3|4|5: 　　　　b='1' 　　case 6|7|8|9|10: 　　　　b='2' 　　case 11|12|13|14|15: 　　　　b='3' 　　case 16|17|18|19|20: 　　　　b='4' 　　case 21|22|23|24|25: 　　　　b='5' print(b)	

5	x=20
	print(x)
	if x>=21:
	    print('five lights')
	else:
	    if x>=16:
	        print('four lights')
	    else:
	        if x>=11:
	            print('three lights')
	        else:
	            if x>=6:
	                print('two lights')
	            else:
	                if x>=1:
	                    print('one light')
	                else:
	                    print('zero light')	

6	x=4
	print(x)
	if x>=1:
	    print('five lights')
	else:
	    if x>=6:
	        print('four lights')
	    else:
	        if x>=11:
	            print('three lights')
	        else:
	            if x>=16:
	                print('two lights')
	            else:
	                if x>=20:
	                    print('one light')
	                else:
	                    print('zero light')	

7	a=75
	if a>=80:
	    if a>=90:
	        b='A'
	    else:
	        b='B'
	else:
	    if a>=70:
	        b='C'
	    else:
	        b='D'
	print(b)	

8	a=6;b=8;c=4;d=2;e=9 max=min=a if b>max:max=b if c>max:max=c if d>max:max=d if e>max:max=e if b<min:min=b if c<min:min=c if d<min:min=d if e<min:min=e sum=a+b+c+d+e-max-min avg=sum/3 print(avg)	
9	a=5;b=4;c=3;d=2 #階次1 if a>b: a,b=b,a if b>c: b,c=c,b if c>d: c,d=d,c print(a,b,c,d) #階次2 if a>b:a,b=b,a if b>c:b,c=c,b print(a,b,c,d)	

10	a1,b1,c1=3,1,5
	a2,b2,c2=1,-2,-3
	if a1/a2==b1/b2==c1/c2:
	　　print('二直線重疊')
	elif a1/a2==b1/b2!=c1/c2:
	　　print('二直線平行')
	else:
	　　d=a1*b2-a2*b1
	　　x=(c1*b2-c2*b1)/d
	　　y=(a1*c2-a2*c1)/d
	print(x,y)	

11	x=200　#度數
	if x<100:
	　　print(x*3)
	elif x<300:
	　　print(100*3+(x-100)*5)
	else:
	　　print(100*3+200*5+(x-300)*6)	

12	a='1';b=0 if a==1: 　　b=1 print(b)	
13	b=0 a=input("input 1:")#本例輸入1 if a==1: 　　b=1 print(b)	
14	a='1';b=0 match a: 　　case 1: 　　　　b=1 　　case '1': 　　　　b=2 　　case _: 　　　　b=3 print(b)	

15	b=0;a=2
	if a>0:
	　　b=1
	if a==2:
	　　b=2
	print(b)	

課後習題

1. 假設所得稅稅率累進法則如下：

   (1) 淨所得 30 萬以下繳納 6%。

   淨所得 30 ～ 80 萬之間，則前面 30 萬繳納 6%，超過 30 萬的部分繳納 13%。

   (3) 淨所得在 80 ～ 200 萬之間繳納 21%。（前面 30 萬繳 6%，30 ～ 80 萬之間繳 13%）

   (4) 淨所得超過 200 萬，超過的部分繳納 30%。

   試寫一程式可以輸入淨所得，並計算應繳納稅額。測資如下：

淨所得	納稅額
20 萬	20 萬＊ 6% =12000
40 萬	30 萬＊ 6% +10 萬＊ 13% = 31000
100 萬	30 萬＊ 6% +50 萬＊ 13% +20 萬＊ 21% =125000
220 萬	30 萬＊ 6% +50 萬＊ 13% +120 萬＊ 21% +20 萬＊ 30% =395000

2. 假設自來水費率如下：

   100 度以下，每度 3 元。

   100 ～ 300 度，超過 100 度的部分，每度 5 元。

   300 度以上，超過 300 度的部分，每度 6 元。

   根據以上條件，寫一程式，可輸入用水度數，得到水費。例如，測資如下表：

度數	水費
50	50 ＊ 3 ＝ 150
200	100 ＊ 3 ＋ 100 ＊ 5 ＝ 800
400	100 ＊ 3+200 ＊ 5+100 ＊ 6=1900

3. 寫一程式輸入 x 值，並印出其所對應的值，其函數對應如下：

$$y = f(x) = \begin{cases} x+3 & x > 100 \\ x^2 & 50 <= x <= 100 \\ \sqrt{x} & 0 < x < 50 \\ 0 & x <= 0 \end{cases}$$

4. 三點共線。請寫一個程式，任意指派 3 點座標，檢查是否共線，若未共線，輸出其面積。
   提示：任意 3 點面積若為 0，則共線。例如：(1,1),(2,2),(3,3) 共線；(1,1),(2,2),(5,2) 不共線。
5. 判斷任意點 D 是否在三角形 ABC 內或外。已知 ABC 三點座標，請寫一程式，可以輸入 D 點座標，並判斷任一點 D 是否在三角形 ABC 內或外。
   （提示：D 若在三角形內，則三角形 ABC 面積 = DAB + DBC + DAC 的面積）
   例如：三角形三點分別是 (0,0),(3,0),(0,4)，則 (1,1) 在三角形內，(5,0) 則在三角形外。
6. 衛福部國民健康署，對於體重身高的比例訂有 BMI 標準如圖 4-2：請寫一程式完成此一指數計算，並給予建議。

圖4-2　BMI 定義與建議表

7. 台北市計程車費率計算如下：
   (1) 計程運價：起程 1.25 公里 70 元，續程每 200 公尺 5 元（日間）。
   (2) 延滯計時運價：車速每小時 5 公里以下，累計每 1 分鐘 20 秒 5 元（日間）。
   (3) 夜間加成運價：自夜間 11 時起至翌晨 6 時止（遇跨夜間加成時段之情形，統一以「上車時間」為準），每趟次依日間運價加收 20 元（即起程運價 1.25 公里 90 元，續程及延滯計時運價與日間相同）。

請問如何規劃與設計此程式？

測試資料編號	搭乘時間(時:分)	行駛距離(公里)	延滯計時累計時間(分:秒)	費率
1	07:20	1	1:20	70+5=75
2	08:30	1.3	5:00	70+5*1+5*3=90
3	23:10	1.9	6:00	70+20+5*4+5*4=130
4	03:40	2	7:00	70+20+5*4+5*5=135

8. 營業稅的計算。營業稅若外加，則營業稅 = 定價 *0.05，若內含，則營業稅 = 定價 /21，且都四捨五入到整數。請寫一程式，程式一執行，先詢問是稅外加，還是內含，然後輸入定價，最後由電腦計算營業稅。

Chapter
05

重覆結構程式設計

學習綱要

學習表現

1. 具備撰寫程式語言基本能力，運用資訊科技方法解決問題。
2. 具備程式設計之邏輯思考能力，展現系統思考、分析與探索之素養。
3. 具備實作程式設計能力，展現程式語言跨域應用之軟實力。

5-1 迴圈基本架構

生活上常會遇到重複執行的事件，例如，一天要上七堂課，連續作答選擇題 10 題等等。所以電腦就有所謂的迴圈，幫您完成指定的重複次數。其次，所有的程式語言都有兩種基本迴圈，那就是 for 與 while，這兩種迴圈的主要差別是，前者是程式設計階段就知道迴圈執行次數，所以又稱爲計數迴圈，請看 5-2 節；而後者是設計階段不知道，要執行後才能知道執行次數，所以稱爲條件迴圈，請看 5-4 節。

5-2 計數迴圈敘述與撰寫

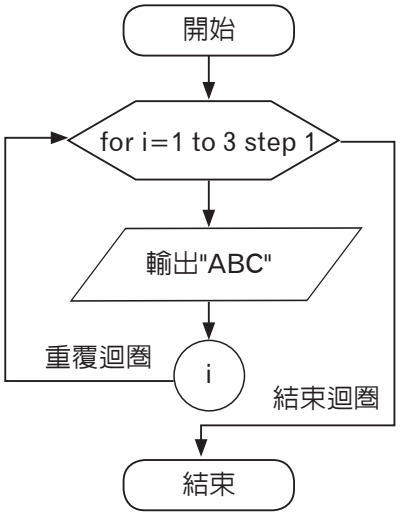
Python 計數迴圈指令爲 for，for 指令的基本語法如下：

```
for var in range(起始值,終值,遞增值)
    敘述區塊
    [break]
    [continue]
    [pass]
```

例如，以下程式可以輸出ABC三次。（請留意不含終值4，且一定要有縮排）

```
for i in range(1,4,1):
    print("ABC")
```

以上程式以流程圖說明如下：



程式縮排在 C/C++、Java 等是美觀，但 Python 是語法，所以一定要縮排。以下就不行。

```
for i in range(1,4,1):
print("ABC")
```

且留意要有冒號 (:)，以下也不行。

```
for i in range(1,4,1)
    print("ABC")
```

遞增值若是 1，也可省略，如以下程式。

```
for i in range(1,4):
    print("ABC")
```

起始值若是 0，也可省略，所以以下程式『ABC』輸出四次。

```
for i in range(4): #索引i之值分別是0,1,2,3
    print("ABC")
```

以下程式可以縱向輸出 1 2 3。（請留意不含終值 4）

```
for i in range(1,4,1):
    print(i)
```

既然 range 不含終值，所以本書習慣使用『+1』，表示含終值。例如：

```
for i in range(1,4+1,1):
    print(i)
```

這樣就會輸出 1,2,3,4。以下程式可以縱向輸出 2 4 6 8 10。（遞增值爲 2）

```
for i in range(2,10+1,2):
    print(i)
```

以下程式可以縱向輸出 5 4 3 2。（請留意不含終值 1，且每次是遞減 1，因爲是遞減，所以終值要小於起使值）

```
for i in range(5,1,-1):
    print(i)
```

遞減若要強調含終值，本書也是用『-1』表示，例如，以下程式輸出 5,4,3,2,1

```
for i in range(5,1-1,-1):
    print(i)
```

以下程式可以縱向輸出 10 7 4。（遞增值爲 -3）

```
for i in range(10,1,-3):
    print(i)
```

以下程式可連續產生 3 個 1~6 的亂數。

```
import random
for i in range(3):
    a=random.randint(1,6)
    print(a)
```

以下程式，可產生 3 個小寫字元。

```
import random
for i in range(3):
    a=random.randint(ord('a'),ord('z'))#ord傳回對應整數
    print(chr(a))
```

範例5-2a

請寫一個程式，可以由電腦產生小寫字元 5 次，每次由使用者鍵入此字元，電腦評判是否正確，統計正確與錯誤題數。

程式列印

前面已經說明如何產生亂數，如何輸入字元，如何比對字元是否相同，現在要重複 5 次，只要使用 for 迴圈，程式如下：

```
import random
r=0
w=0
for i in range(5):
    a=random.randint(ord('a'),ord('z'))#ord傳回對應整數
    print(chr(a))
    b=input()
    if b==chr(a):
        r=r+1
        print("Right")
    else:
        w=w+1
        print("Wrong")
print("Right=%d" % r)
print("Wrong=%d" % w)
```

自我練習

1. 請使用 Debug 工具，觀察以上程式執行流程。
2. 請產生 6 個 -3~3 的亂數，統計正數、0、負數的個數。
3. 請產生 10 個 1~6 的亂數，統計偶數與奇數個數。
4. 請產生 10 組 (x,y) 座標亂數，(x,y) 都介於 -2 與 2 的整數，並統計落在原點、x 軸、y 軸、四個象限的個數。
5. 請產生 10 個 1~6 的亂數，統計數字「1」出現的次數。
6. 請產生 10 個 1~6 的亂數，統計所有數字出現的次數。
7. 請產生 10 個 1~6 的亂數，統計哪一個數字出現的次數最多。
8. 心算練習。請連續產生 8 題兩個 1 位數，由使用者輸入相加結果，電腦回應對或錯，並統計正確與錯誤的題數。
9. 音感練習。請連續發出 10 個 Do 到高音 Do，由使用者輸入 1~8，電腦回應對或錯，並統計正確與錯誤的題數。
10. 請由電腦自動產生 30 個大於等於 0 且小於等於 100 的亂數 x，統計 0~59,60~69,70~79,80~89,90~100 的個數。
11. 請寫一個程式，電腦可以連續出現 50 個小寫字元，電腦統計出現母音 (a,e,i,o,u) 的個數。

範例5-2b

編譯器的乘法運算

演算法

我們人類的乘法運算式是先背誦九九乘法表，然後用以下直式乘法計算兩數相乘。

$$\begin{array}{rrrr} & & 8 & 2 \\ \times & & 3 & 6 \\ \hline & 4 & 9 & 2 \\ 2 & 4 & 6 & \\ \hline 2 & 9 & 5 & 2 \end{array}$$

但是電腦可不用如此麻煩，電腦的強項是使用循序法，可循序逐一累加計算。（電腦內部只有累加器與比較器），所以就用 for 迴圈實現循序累加如下：

```
a=82
b=36
s=0
for i in range(b):
    s=s+a
print(s)
```

範例5-2c

試寫一程式，可以輸入 2 至 9 的整數，並輸出此數的九九乘法表。

執行結果

```
input a:3
 3 *  1 =  3
 3 *  2 =  6
 3 *  3 =  9
 3 *  4 = 12
 3 *  5 = 15
 3 *  6 = 18
 3 *  7 = 21
 3 *  8 = 24
 3 *  9 = 27
```

程式列印

這是很典型重複的工作，所以使用 for 迴圈設計如下：

```
a=int(input("input a:"))
for i in range(1,9+1):
    print("%2d*%2d=%3d"% (a,i,a*i))
```

以上 "%2d*%2d=%3d" ，「%」代表控制字元，會分別到後面變數區依序找對應變數 a , i , a*i 。「2」「3」代表此對應變數輸出時所佔寬度，且數值向右對齊。

5-3 變更迴圈流程的程式語法撰寫

在結構化程式設計的前提下，程式設計不能再用 goto 變更迴圈流程，Python 則改用 break 與 continue 變更迴圈流程，分別說明如下：

break

break 可提早離開迴圈。例如，以下程式僅輸出 1 2 3 4

```
for i in range(1,10,1):
    if i==5:
        break
    print(i)#縱向1 2 3 4
```

continue

continue 可略過 continue 後面的迴圈內容，提早回到迴圈的起始點。例如，以下程式輸出 1 2 3 4 6 7 8 9，i==5 成立時，print(i) 就被略過，換下個 i, 此時 i=6，繼續執行迴圈。

```
for i in range(1,10,1):
    if i==5:
        continue
    print(i) #縱向1 2 3 4 6 7 8 9
```

pass

Python 不允許空迴圈、空函式、空類別，若一定要形成空迴圈、空函式、空類別，則裡面一定要放一個 pass。例如：

```
for i in range(4):
    pass
```

自我練習

1. 請使用 Debug 工具，觀察以上程式執行流程。
2. 請鍵入以下程式，並觀察寫出執行結果。

題號	程式	執行結果
1	`for i in range(1,-5,1):` `    print(i)`	

2	`for i in range(3,6,-1):`
	`    print(i)`
	`print(i)`	

3	`for i in range(3,6,1):`
	`    print(i)`
	`print(i)`	

4	`for i in range(6,3,-1):`
	`    print(i)`
	`print(i)`	

5	`for i in range(4):`
	`    pass`
	`print (i)`	

範例5-3a

求解輸入數值是否質數。

演算法

任一整數，除了 1 和本身外，若沒有任何數可以整除此數，則稱此數爲質數。所以，本例也可使用循序法，使用 2 至該數減 1 的數一一試除，若無一數可整除，則稱此數爲質數。其次，爲了提高程式執行效率，若找

到 1 個可以整除，那此數就不是質數，可以提早離開迴圈，這樣才不會浪費時間，可提高執行效率，因爲許多考試的程式執行時間也要算分數。

程式列印

```
i=12
b=True       #先假設所有整數都是質數
print(i)
for j in range(2,i):  #使用該數減1的數一一試除
    if i %j ==0 :     #找到一個可以整除，就不是質數，且提早離開
        b=False
        break
if b==True:
    print("prime")
else:
    print("not prime")
```

自我練習

1. 請寫一程式，可以判斷一個四位數是否全偶數。例如，2468 傳回『是』，1468 傳回『否』。

5-4 條件迴圈敘述與撰寫

上一節的 for 是用於程式設計階段已知迴圈次數，但有些情況，我們於程式設計階段並不知迴圈的執行次數，一定要等到程式執行後，才能知道何時離開迴圈，此時即可使用條件迴圈 while 指令。while 的迴圈語法如下：

```
while(條件運算式):
    程式區塊
    [break]
    [continue]
    [pass]
```

以上語法說明如下：

1. 同 for 迴圈，請留意冒號與縮排。
2. 當條件運算式值爲（True）時，繼續執行迴圈，運算式爲 (False) 時，離開迴圈。

3. while 程式區塊內亦適用 break 與 continue，前者爲強迫提早離開迴圈，後者可略過部份指令，提早進入條件運算式。
4. 若是空迴圈，也要放一個 pass。

範例5-4a

假如沒有除法運算子，請自行使用加減法，完成除法運算。

演算法

兩數相乘時，程式設計階段就知道迴圈執行次數，所以使用 for。但除法就是不知道，只能說，當被除數大於除數時，就連續減去除數，能減去的次數，就是商，剩下的就是餘數。以 8 除以 3 爲例，8 可以連續減 3 兩次，所以商就是 2，剩下的就是餘數。此爲設計階段不知重複次數，所以使用 while 迴圈。

程式列印

```
a=8#被除數
b=3#除數
q=0#商
while(a>=b):#只要(被除數>=除數)  就執行迴圈
    a=a-b    # (被除數)-(除數)
    q=q+1    #商每次遞增1
print(q)# 商數
print(a)#餘數
```

程式說明

1. a= 被除數。
2. b= 除數。
3. 商數 q=0。
4. 所謂的商就是被除數 a 共有幾個除數 b，也就是只要 a 大於等於 b，就要執行以下指令：

   a=a-b

   q=q+1

5. 本例以 8 除以 3，實際演練如下：

(1) a=8；b=3

a>=b 成立，所以執行迴圈指令

a=a-b=5

q=q+1=1

(2) a=5；b=3

a>=b 成立，所以執行迴圈指令

a=a-b=2

q=q+1=2

(3) a=2；b=3

a>=b 不成立，所以離開迴圈

q=2（商）

a=2（餘數）

自我練習

1. 請使用 Debug 工具，觀察程式執行流程。
2. 同範例，但可以求到小數點 1 位的實數除法。提示：將被除數先乘以 10，再運算，最後將商再除以 10。

範例5-4b

請寫一個擲骰子程式，滿足以下條件：

(1) 可以產生 1 至 6 的亂數。

(2) 累加以上亂數。

(3) 輸出此亂數與統計其和。

(4) 若亂數不爲 1，則重複（1）～（3），否則輸出其和並結束程式。

執行結果

```
2 3 3 2 3 5 2 4 6 1 30
```

程式列印

1. 因爲是重複事件，此爲迴圈應用，且一開始不知重複次數，所以要使用 while 迴圈。
2. 「只要 a!=1，就要重複執行迴圈，產生亂數」的程式語言，符合 while 的用法，程式如下：

```
while a!=1:
```

3. Python 並沒有後測試迴圈，所以有時候就要先執行一次，再進入迴圈，如以下程式。

```
import random
s=0
a=random.randint(1,6)
print(a,end=' ')
while a!=1:
    s=s+a;
    a=random.randint(1,6)
    print(a,end=' ')
print(s)
```

補充說明

這題也有人這樣出題『電腦一直產生 1 到 6 的亂數，直到產生 1 時停止』，所以 while 可以修改如下：

```
while not(a==1):
```

not 就想爲『直到』這樣就很好理解。

自我練習

1. 同上範例，但人和電腦玩，人與電腦每次產生 1 個亂數，點數大者爲贏，直到人贏爲止，輸出共產生幾次亂數。
2. 同上範例，但人和電腦玩，人與電腦每次產生 1 個亂數，點數大者爲贏，直到人贏 3 次爲止，輸出共產生幾次亂數。
3. 請寫一個程式，滿足以下條件：(擲骰子遊戲)
   (1) 可以產生兩個 1 至 6 的亂數。

(2) 累加以上亂數。
(3) 輸出此亂數與其和。
(4) 若亂數和大於 8，則重複（1）～（3），直到亂數和小於等於 8，則程式結束。

4. 電腦心算測驗。請寫一程式，可以無限次數，顯示題號、出現兩個 1 位數，並由使用者回答，直到答對 10 題，電腦顯示答錯題數。
5. 同上題，但加上直到答錯才停止，並顯示連續答題數目。
6. 請寫一猜拳遊戲程式，可以讓人與電腦猜拳，並輸出結果，直到任一方贏 3 次爲止。

範例5-4c

擲骰子遊戲。
請寫一程式，可以連續產生 3 個 1 ～ 6 的亂數，並輸出此 3 個亂數，直到有其中兩個亂數相等爲止，並輸出另一個不相同的數字。

執行結果

```
5 1 2
1 6 4
4 5 4
5
```

程式列印

1. 此也是設計階段不知迴圈重複次數，所以使用 while 迴圈。
2. 只有 Basic 有『直到』until 的迴圈，其實，『直到』同義於『not』，所以也是用 while 迴圈。
3. C/C++ 等還有後測試迴圈，Python 則沒有，遇到後測試迴圈，那就只好先做一次，再進入迴圈，所以程式如下：

```
import random as r #r是物件名
a=r.randint(1,6)
print(a,end=' ')
b=r.randint(1,6)
print(b,end=' ')
```

```
c=r.randint(1,6)
print(c)
while not ((a==b )or (b==c) or (a==c)):
    a=r.randint(1,6)
    print(a,end=' ')
    b=r.randint(1,6)
    print(b,end=' ')
    c=r.randint(1,6)
    print(c)
if a==b:
    print(c)
elif b==c:
    print(a)
else:
    print(b)
```

範例5-4d

擲骰子遊戲。

請寫一程式，可以連續產生 4 個 1 ～ 6 的亂數，並輸出此 4 個亂數，直到有其中兩個亂數相等爲止，並輸出此不相等的數字與和。例如，產生 6,4,5,1 則繼續產生亂數，若亂數爲 6,2,1,6 則其和爲 3。

程式列印

將上題推廣就可以，所以程式如下：

```
import random as r #r是物件名
a=r.randint(1,6)
print(a,end=' ')
b=r.randint(1,6)
print(b,end=' ')
c=r.randint(1,6)
print(c,end=' ')
d=r.randint(1,6)
print(d)
while not ((a==b )or (a==c) or (a==d) or (b==c) or (b==d) or (c==d)):
    a=r.randint(1,6)
    print(a,end=' ')
    b=r.randint(1,6)
```

```
    print(b,end=' ')
    c=r.randint(1,6)
    print(c,end=' ')
    d=r.randint(1,6)
    print(d)
if a==b:
    print(c+d)
elif a==c:
    print(b+d)
elif a==d:
    print(b+c)
elif (b==c):
    print(a+d)
elif (b==d):
    print(a+c)
else:
    print(a+b)
```

自我練習

1. 由範例可知，若是四顆骰子分別是 1,1,6,6 那就有點運氣了，其和有可能 12，或有可能 2，若是定義得分取大的，那應該其和是 12 才對，請自己思考如何完成。
2. 請寫一個程式，可以人和電腦玩以上遊戲，點數和大的贏。

範例5-4e

請將 10 進位數轉爲 N 進位數（本例暫討論 N<=9，其餘 N>=11 時，待串列介紹之後再討論）。

演算法

1. 若要將十進位的 a 轉爲 2 進位，則以數學式表示如下：

   $(a)_{10}=a_0+a_1 2^1+a_2 2^2+a_3 2^3+a_4 2^4$---

   $=a_0+2(a_1+a_2 2^1+a_3 2^2+a_4 2^3$---)　　//$a_1$ 以後，提出 2
2. 上式的 a_0 爲 a 除以 2 的餘數，$a_1+a_2 2^1+a_3 2^2+a_4 2^3$…則爲 a 除以 2 的整數商，重複上式，直至整數商等於零爲止。因爲數字長度不限，此爲未知執行次數迴圈，所以應該使用「while a>0」的迴圈。

3. 最先出爐的餘數應放在 2 進位的最右邊。
   $(a)_{10}=(a_n\cdots a_3a_2a_1a_0)_2$

程式列印

```
a=13     #待轉換的十進位數
n=2
r=0      #餘數
d=''
while a>0:  #只要a>0則重覆迴圈
    r=a % n
    d=str(r)+d #d是上一次的餘數，要放右邊。str()是整數轉字串
    a=a//n
print(d)#1101
```

程式說明

1. a 爲待轉換的十進位數。
2. n 爲 N 進位數 2 到 9 都可以。
3. r 爲餘數。
4. 將 a 連續除以 n，直到整數商爲 0，其餘數的字串串接，即爲 n 進位數。
5. 本例進入迴圈之前，我們並無法預估迴圈次數，所以較適用 while 迴圈（亦可用 for，但程式架構較不漂亮）。且有可能迴圈一次均不執行，所以要用前測試迴圈。
6. 本例輸出時，先出爐的餘數要放右邊，所以是 d=str(c)+d, 請您改爲 d=d+str(c), 並觀察執行結果。

自我練習

1. 請寫一個程式，密碼爲「5566」，僅可以錯兩次，密碼正確可以執行以上範例，錯第 3 次，程式就結束。
2. 請寫一個程式，可以連續輸入阿拉伯數字，每次 1 個，當輸入「e」時結束，且組合成一個數字。例如：輸入 4,3,8,e 則輸出 438。

範例5-4f

請寫一程式，可以將所輸入的任意長度正整數，反向輸出。例如，輸入1234，輸出「4 3 2 1 」。

演算法

1. 數字的分解在 3-3 節已經介紹使用「%、//」運算子，現在數字長度不限，所以要使用 while 迴圈。
2. 使用變數 a 指派數字。只要 a>0 就要重複迴圈，所以程式如下：

```
a=1234
while a>0:
    b=a%10
    print(b, end=' ')
    a=a//10
```

自我練習

1. 同範例 5-4f，但求和與平均。
2. 同範例 5-4f，但是兩兩 1 組，反向輸出。例如，a=12345，輸出 45 23 1。
3. 同範例 5-4f，但是每次 2 個，反向輸出。例如，a=12345，輸出 45 34 23 12。
4. 請寫一個程式，可以輸入若干個整數，當輸入 0 時，程式結束，且求輸入數字的平均。

範例5-4g

請以輾轉相除法，求兩數的最大公因數。

演算法

兩個數字的最大公因數，方法一就是分別求出其因數，再挑一個最大的。方法二是轉轉相除法，這是古代數學家想到的，也是高中數學，所以這題考試常考，其演算法如下：

1. 輸入 a、b 兩數。
2. 假如 b>a，則兩者交換。
3. r= 餘數。
4. 將 a 除以 b，得餘數 r，假如餘數不爲 0，則以原除數爲被除數，原餘數爲除數，重複執行 a 除以 b，直到餘數爲 0，促使餘數爲 0 的除數，即爲最大公因數。
5. 本例以 a=21，b=9，實際演練如下：

   (1) a=21；b=9；

   r=a%b=3

   a=b=9

   b=r=3

   r>0 成立，所以繼續執行迴圈。

   (2) a=9；b=3；

   r=a%b=0

   a=b=3

   b=r=0

   r>0 不成立，所以離開迴圈，最大公因數爲 a=3。

程式列印

```
a=21
b=9
if b>a:
    a,b=b,a #a,b的值交換
r=a % b
while r>0:  #只要餘數大於0則重覆迴圈
    a=b
    b=r
    r=a%b
print(b)#3
```

程式說明

1. 輾轉相除法的執行之初並無法預估重覆執行次數，所以適用 while。
2. 離開迴圈時，當時的 b，就是餘數，就是最大公因數。

5-5 巢狀迴圈敘述與撰寫

迴圈中又有迴圈，稱爲巢狀迴圈。巢狀迴圈在程式設計的領域非常重要 (因爲很多問題都需要雙迴圈、甚至三層迴圈)，也是初學者最感頭疼的單元，本節將使用若干範例引領學生征服此運算思維。

範例5-5a

請寫一程式，印出如下的九九乘法表。

執行結果

```
1*1= 1 1*2= 2 1*3= 3 1*4= 4 1*5= 5 1*6= 6 1*7= 7 1*8= 8 1*9= 9
2*1= 2 2*2= 4 2*3= 6 2*4= 8 2*5=10 2*6=12 2*7=14 2*8=16 2*9=18
3*1= 3 3*2= 6 3*3= 9 3*4=12 3*5=15 3*6=18 3*7=21 3*8=24 3*9=27
4*1= 4 4*2= 8 4*3=12 4*4=16 4*5=20 4*6=24 4*7=28 4*8=32 4*9=36
5*1= 5 5*2=10 5*3=15 5*4=20 5*5=25 5*6=30 5*7=35 5*8=40 5*9=45
6*1= 6 6*2=12 6*3=18 6*4=24 6*5=30 6*6=36 6*7=42 6*8=48 6*9=54
7*1= 7 7*2=14 7*3=21 7*4=28 7*5=35 7*6=42 7*7=49 7*8=56 7*9=63
8*1= 8 8*2=16 8*3=24 8*4=32 8*5=40 8*6=48 8*7=56 8*8=64 8*9=72
9*1= 9 9*2=18 9*3=27 9*4=36 9*5=45 9*6=54 9*7=63 9*8=72 9*9=81
```

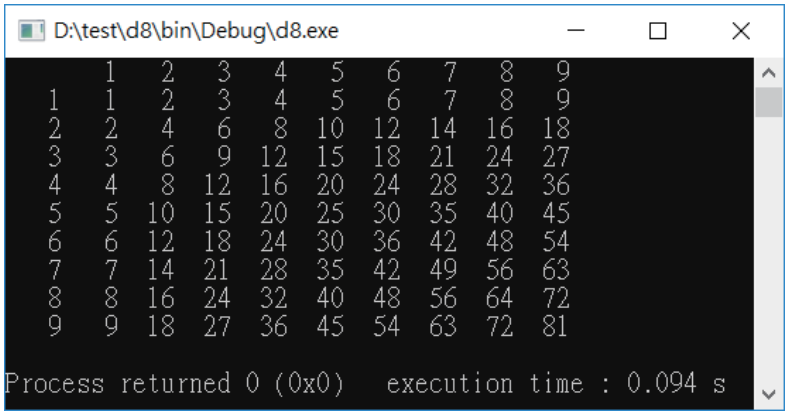
程式列印

前面我們已經用單一迴圈完成指派數字的九九乘法表，現在此數字也要從 2~9，所以在外面再加一個迴圈，程式如下，此迴圈中有迴圈，稱爲巢狀迴圈。

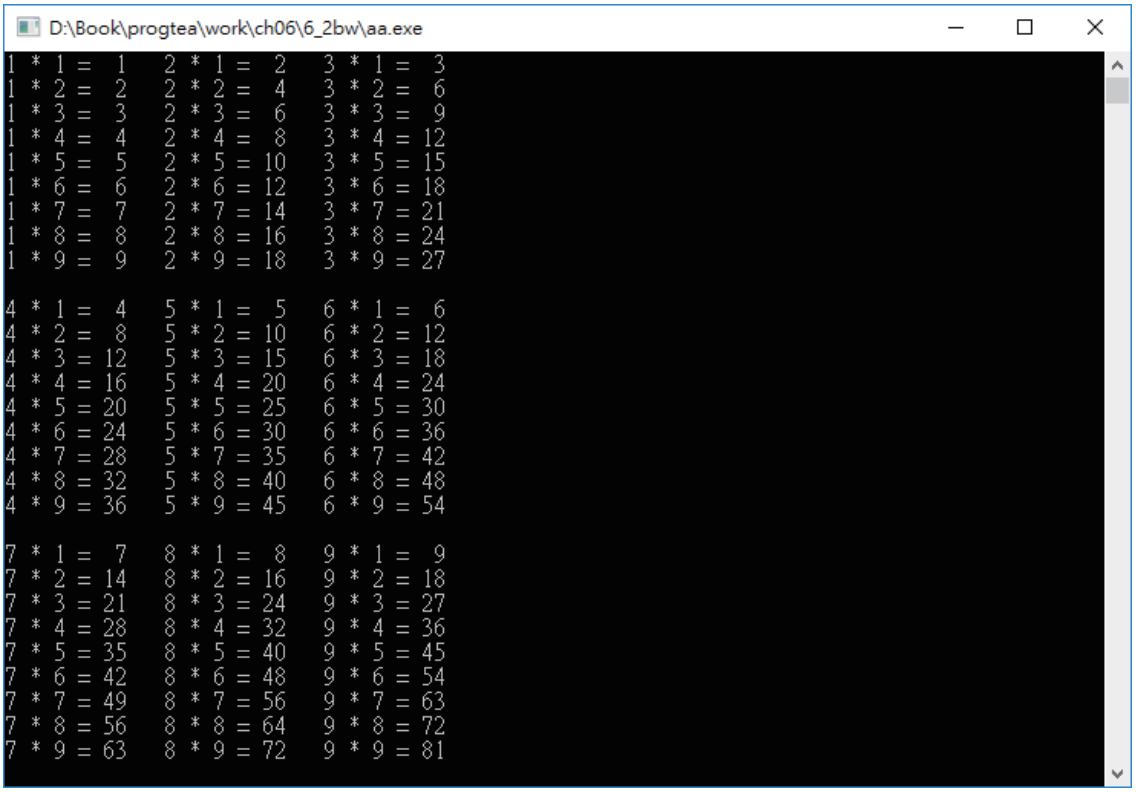
```
for i in range(1,9+1):
    for j in range(1,9+1):
        print("%d*%d=%2d"% (i,j,i*j),end=" ")
    print()
```

自我練習

1. 請寫一程式，嘗試使用兩層迴圈印出如下的九九乘法表。



2. 請寫一程式，嘗試使用三層迴圈印出如下的九九乘法表。



範例5-5b

請寫一程式，印出 2 至 100 的所有質數。

執行結果

```
  2  3  5  7 11 13 17 19 23 29
31 37 41 43 47 53 59 61 67 71 73
79 83 89 97
```

運算思維

前面我們已經使用一個迴圈判別一個整數是否爲質數，現在要找 2 ~100 的所有質數，也是再外加一個迴圈，所以程式如下：

程式列印

```
for i in range(2,100+1):
    b=True
    for j in range(2,i):
        if i %j ==0 :
            b=False
    if b==True:
        print("%3d" %i,end="")
```

自我練習

1. 同上範例，但請加上讓每列僅輸出 10 個數字。

※ 2. 質因數連乘積。請寫一個程式，輸入一正整數，將其質因數分解後印出其式子，例如：

```
輸入：319
輸出：319=11*29
輸入：19
輸出：19 =質數
輸入：521752
輸出：521752 = 2^3*7^2*11^3
```

範例5-5c

試寫一程式，找出三位數“阿姆斯壯數”。所謂阿姆斯壯數是指一數等於各個位數的立方和。例如，$153=1^3+5^3+3^3$。

執行結果

```
153
370
371
407
```

程式列印

1. 一個 3 位數，我們可以看成百位數 i 從 1 到 9，十位數 j 從 0 到 9，個位數 k 從 0 到 9，所以所有三位數的組合可用以下程式的 i,j,k 三個迴圈。
2. i,j,k 若代表一個三位數，則所代表的三位數如下：

```
s1=100*i+10*j+k
```

3. 此三位數所有數字的三次方如下：

```
s2=i**3+j**3+k**3
```

4. 全部程式如下：

```
for i in range(1,9+1):
    for j in range (0,9+1):
        for k in range(0,9+1):
            s1=100*i+10*j+k
            s2=i**3+j**3+k**3
            if s1==s2:
                print (s1)
```

5. 此三位數也可以使用 1 個迴圈，i 從 100 到 999，現在要得到百位數、十位數、個位數，則要將此三位數使用「//,%」技術先分解，程式如下:

```
    a=i//100          #百位數
    b=(i-a*100)//10   #十位數
    c=i%10            #個位數5-24
```

6. 全部程式如下：

```
for i in range(100,999+1):
    a=i//100            #百位數
    b=(i-a*100)//10      #十位數
    c=i%10             #個位數
    s=a**3+b**3+c**3
    if i==s:
        print(i)
```

自我練習

1. 試寫一程式，找出四位數“阿姆斯壯數”abcd=a^4+b^4+c^4+d^4。

範例5-5d

請嘗試使用雙迴圈，寫一程式輸出如下：

執行結果

```
*
**
***
****
*****
******
```

程式列印

```
for i in range(1,6+1):
    for j in range (1,i+1):
        print ("*",end='')
    print()
```

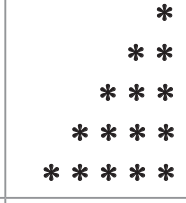
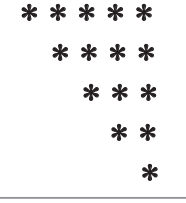
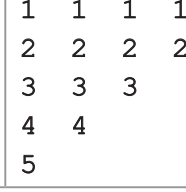
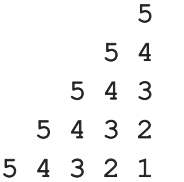
程式說明

1. 本例的主要目的是訓練學生，熟練雙迴圈的內外迴圈解析，也就是要找其關係，這樣可以減少變數的數量，有助於後續複雜程式的簡化，例如往後串列與排序的應用。
2. 本例的列數、每一列的星星個數，列表解析如下：

列編號	星星個數
1	1
2	2
3	3
4	4
5	5
6	6

3. 由上表可知，共 6 列，所以 i 從 1 到 6。
4. 共有兩個變數，我們就是要找關係，這樣才能簡化變數的個數，簡化問題，找出解答。本例內迴圈 j 與 i 的關係爲 i=j ，所以 j 從 1 到 i。

自我練習

題號	題目	題號	題目
1		2	
3	 （105APCS 試題）	4	 （105APCS 試題）
5		6	

7	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	8	A B B C C C D D D D E E E E E
9	1 1 2 1 2 3 1 2 3 4 1 2 3 4 5	10	1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
11	5 5 5 5 5 4 4 4 4 3 3 3 2 2 1	12	5 4 3 2 1 4 3 2 1 3 2 1 2 1 1
13	1 1 2 1 2 3 1 2 3 4 1 2 3 4 5	14	1 1 2 1 2 3 1 2 3 4 1 2 3 4 5
15	A AB ABC ABCD ABCDE	16	a bb ccc dddd eeeee

17. 請寫程式完成 1+(1+2)+(1+2+3)+(1+2+3+4)+(1+2+3+4+5) 的和。

5-6 重覆結構程式實作演練

循序法

循序法故名思義就是將所有的值一一嘗試。例如，前面範例 5-2b，我們已經介紹計算機的乘法，82*36 是 82 連加 36 完成。此即爲電腦的特性，計算能力超強且快，以下我們將繼續延伸此方法，解決等差級數與平方根的循序求解方法。程式語言的 for 就是用來實現循序法。請看以下範例。

範例 5-6a

等差數列的求和。例如：寫一個程式計算 1+2+3+…+100 的和，或計算 3+6+9+…+93 的和。

運算思維

本例人類通常使用等差級數求和的公式（首項 + 末項）* 項數 /2，但電腦有快速累加能力，所以不用如此麻煩，不用任何學習。電腦就直接一個一個累加，程式如下，此稱爲循序法。前面範例 5-3b 的乘法計算，我們也是使用循序法。

```
s=0
for i in range(100+1):
    s=s+i
print(s)
```

自我練習

1. 請寫一個程式計算 3+6+9+…93 的和。
2. 請寫一個程式計算 1+1/2+1/3+…+1/20 的和。
3. 3+6+12+24+…+3072 的和。
4. 5+5/2+5/4+…+5/1024 的和。

循序猜值法

所謂循序猜值法，就是將所有可能的解一一循序代入，又稱爲暴力猜值法。例如，您要求任一數的開平方，因爲開平方的結果一定在 0 與此數之間，那我們就從 0 開始，每次遞增 1、0.1、0.01 或 0.001…，至於是要遞增多少，那就依您要的精密度了。例如，要求整數解，就遞增 1，要求到小數 1 位，就遞增 0.1，要求到小數兩位，就遞增 0.01 等等等，請看以下範例說明。

範例5-6b

請用循序猜值法求任意數的平方根。

運算思維

1. 人類求開根號的方法，會使用二項和的二項式定理。
2. 二項和的二項式定理如下：

   $(10a+b)^2$（展開）

   $=100a^2+20ab+b^2$ (b 提出)

   $=100\ a^2+b(20a+b)$
3. 以 138384 爲例，開根號運算過程如下：
   (1) 由以上 $100\ a^2+b(20a+b)$，表示我們由左到右，每兩個一組，本例 138384 將分爲 13,83,84 三組如表 5-1，然後先找 a，再找 b。
   (2) 找出 a。從 9,8,7 到 1 的平方依序找出小於等於最左邊那一組的數字 (本例是 3)，因爲 81(9*9),64(8*8),49,36,25,16 都太大，所以找到 3，並扣掉 3 的平方 9，剩下 4，請看表 5-1 運算步驟 1。
   (3) 使用迴圈，從第二組數字開始逐一找 $b_1b_2b_3\cdots$。
      (a) 計算每一次的餘數。餘數 d = 前面餘數 × 100 + 這一組數字，本例 d 是 483，如表 5-1 步驟 2。
      (b) 用迴圈找 b。b 從 9,8,7 到 1 ，找 b(20a + b) 小於餘數 d。例如，本例 a 是 3，d 是 483，所以運算步驟如下：

      b=9,9*(20*3+9)=621 太大，不行

      b=8,8*(20*3+8)=544 太大，不行

      b=7,7*(20*3+7)=469 已經小於 483，所以可以，此迴圈結束，請看表 5-1 步驟 2。
      (c) 餘數 d 扣掉 b(20a + b) 。d = d – b(20a + b) 。
      (d) 把得到的 37 看成 a，繼續用 b(20a+b) 找下一位數的 b，如表 5-1 步驟 3。
   (4) 輸出 $ab_1b_2b_3\cdots$，即爲所求。

表5-1　以二項式定理求開根號

步驟		3　7　2
1		1　3　8　3　8　4 9
2	3×20+7=67 67×7=469	4　8　3 4　6　9
3	37×20+2=742 742×2=1484	1　4　8　4 1　4　8　4
4		0

4. 以上運用二項式定理可以減少計算，但是電腦計算能力強，所以可使用循序猜值法求解，此即爲電腦的運算思維。
5. 以下程式每次遞增 1。

```
a=9
for i in range(9):
    if i*i>=a:
        print(i)#3
        break
```

6. 以下程式每次遞增 0.1。因爲 range 的遞增值只能整數，不能是浮點實數 0.1，所以把他先放大 10 倍，以 9 爲例，變成 0,1,2,3..89,90，求結果時再除以 10，就變成 0,0.1,0.2,0.3…8.8,8.9,9。

```
a=9
s=10#放大倍數
for i in range(a*s+1):
    if (i/s)*(i/s)>=a:
        print(i/s)#3.0
        break
```

7. 同理，以下程式每次遞增 0.01。

```
a=9
s=100#放大倍數
for i in range(a*s+1):
    if (i*i/(s**2))>=(a):
        print(i/s)
        break
```

自我練習

1. 請用循序法求解某一正數的立方根。例如，輸入 27 可得到 3.0。本例小數點取 1 位。

2. 請用循序法求解兩數相除的結果。例如，輸入 27 與 9 可得到 3.0。本例小數點取 1 位。（本例假設除數是大於 1 的整數）

3. 假設有一函式 $y=f(x)=x^2-4x-5$ 請分別印出 x 從 -10 到 10 的值。

4. 同上題，請用循序猜值找出其整數解。提示：x 從 -10 到 10 一一代入，找出使函數爲零的值，此即爲暴力猜值法解題。

5. 同上題，請找出極小值。

6. 函數極值。請寫一程式，可以輸入一個一元二次函式，並求其極大或極小值。例如，輸入 $y=f(x)=x^2-2x+2$ 有極小值 1，輸入 $y=f(x)=-x^2-2x+2$ 有極大值 -1。

7. 假設一個一元多次方程式含有實數解，請寫一程式，可求其解。例如，$y=f(x)= x^2+x-0.75=0$ 的解是 0.5 和 -1.5。提示：浮點運算時，若無法得到 0，此時要使用接近 0 的判斷。例如，abs(y)<0.0001，即可視爲成立，0.0001 即是其精密度，請換成自己想要的精密度 0.1、0.01 或 0.001。但是 Python 竟然無此問題，請鍵入以下程式，並觀察結果。

```
s=10#放大倍數
for x in range(-10*s,10*s):
    y=((x/s)**2)+x/s-0.75
    if y==0:
        print (x/s)
```

8. 請用循序法求解任意整數的所有因數。

範例5-6c

請寫一個程式，找出 1 至 200 的整數中，找出含有 2 的數字，且統計共含有多少個 2。例如，122 有兩個 2。

演算法

這是高中排列組合的問題，解題要先分析 2 出現的位置，但電腦就神了，就通通列出來，再將這些數字分解、並計算 2 的個數，數字如何分解，請複習 3-3 節。

程式列印

```
a=200
s=0
for i in range(1,a+1):
    a1=i//100 #百位數
    a2=(i-a1*100)//10 #十位數
    a3=i %10 #個位數
    if a1==2:
        s=s+1
    if a2==2:
        s=s+1
    if a3==2:
        s=s+1
print(s)
```

二分猜值法

前面的循序法是循序一個一個猜，若沒猜中，每次僅減少一筆資料，這裡要介紹一個較有效率的猜值法，稱爲二分猜值法。二分猜值法是每次猜其可能範圍的中間值，若猜的太大，則調整猜值上限爲此中間值，表示後半部都刪除；若猜的太小，則調整猜值下限爲此中間值，表示前半部都刪除，每次都縮小範圍爲一半，所以可提升猜值效率，請看一下範例。

範例5-6d

猜數字。請您先默想一個 1 ～ 100 的數字，讓電腦猜，但每次要回應太大『3』或猜中『2』或太小『1』。

執行結果

以下是我默想 40，逐次回答電腦猜值的過程。

1. 電腦首次猜 1~100 的中間 50，我回答太大。
2. 電腦修正上限爲 49，所以第 2 次電腦猜 1~49 的中間 25，我回答太小。
3. 電腦修正下限爲 26，所以第 3 次電腦猜 26~49 的中間 37，我回答太小。
4. 電腦修正下限爲 38，所以第 4 次電腦猜 38~49 的中間 43，我回答太大。
5. 電腦修正上限爲 42，所以第 5 次電腦猜 38~42 的中間 40，我回答猜中。

```
電腦猜x=50
Please input 3(太大),2(猜中),1(太小):3
太大,電腦猜x=25
Please input 3(太大),2(猜中),1(太小):1
太小,電腦猜x=37
Please input 3(太大),2(猜中),1(太小):1
太小,電腦猜x=43
Please input 3(太大),2(猜中),1(太小):3
太大,電腦猜x=40
Please input 3(太大),2(猜中),1(太小):2
Bingo,The number is 40
```

運算思維

1. 讓電腦猜，電腦可從 1,2,3,4 一個一個開始猜，此稱爲循序猜值法，若數字是 1，那運氣好，1 次就猜到；若數字是 100，那就要猜 100 次，所以平均是 50 次才可猜到，這就由讀者練習。
2. 本範例要介紹二分猜值法，那就是每次猜其一半，例如，本例下限 x1=1，上限 x2=100，我就猜其中間值 x=(x1+x2)/2=50，然後您會告知猜中、太大或太小。

3. 若是太大，則調整上限x2=x-1=49;若是太小，則調整下限x1=x+1=51，然後重複步驟2與3，每次範圍縮小一半，很快就會猜到，程式如下：

程式列印

```
x1=1
x2=100
right=False
while not(right):
    x=(x1+x2)//2
    print('電腦猜x=%d'%x,end='')
    a=input("Please input 3(太大),2(猜中),1(太小):")
    if a=='2':
        right=True
        break
    elif a=='3':
        print('太大,',end='')
        x2=x-1
    else:
        print('太小,',end='')
        x1=x+1
print("Bingo,The number is %d"% x)
```

※ 範例5-6e

請以二分猜值法求解一正數的平方根。本例要精密度到小數點以下第二位。

執行結果

```
1:x1=0.00,x2=9.00
2:x1=0.00,x2=4.50
3:x1=2.25,x2=4.50
4:x1=2.25,x2=3.38
5:x1=2.81,x2=3.38
6:x1=2.81,x2=3.09
7:x1=2.95,x2=3.09
8:x1=2.95,x2=3.02
9:x1=2.99,x2=3.02
10:x1=2.99,x2=3.01
2.9970703125
```

演算法

本例以求 9 的評分根爲例。求解 9 的平方根，其解必在 0 到 9 之間，所以設定下界爲 0，上界爲 9。

1. 首先，先猜 4.5，如下圖步驟 1，但 4.5 的平方爲 20.25，大於 9，表示猜的太大，那就縮小範圍，將上界調爲 4.5，目前下界 0 ，上界 4.5 。
2. 第二次就猜 2.25，如下圖步驟 2。但是 2.25 的平方爲 5.025，小於 9，表示猜的太小，那就調整下界爲 2.25，目前下界 2.25 ，上界 4.5 。
3. 那要猜到何時呢？答案是設定一個精密度，例如，您要精密度達小數一位，那就是下界與上界之間的距離小於 0.1；若是要小數兩位，那就是下界與上界之間的距離小於 0.01，也就是只要距離大於 0.1(精密度小數 1 位)，或距離大於 0.01（精密度小數 2 位）通通要繼續猜；當離開迴圈時，此時下界或上界的值，就都是所要求的答案了。
4. 在一維座標裡，兩個數值相減，取絕對值的物理意義就是「距離」。Python 數值運算的絕對值是使用 abs() 函式。例如：

```
abs(3-4)=1
```

表示座標 3 與座標 4 的距離爲 1。

猜值步驟	猜值內容
1	x1　　x　　x2 0　　4.5（太大）　　9
2	x1　　x　　x2（調整上界 x2） 0　　2.25（太小）　　4.5
3	（調整下界 x1）　x1　　x　　x2 3.374（太大）

設計步驟

根據以上運算思維，求解平方根的自然語言演算法如下：

(1) 設求解的正數爲 y，則其平方根必在 x1=0（下界）與 x2=y（上界）之間，猜值範圍爲 [x1,x2]。

(2) 首先猜 x1+x2 之和的一半 x。

(3) 若所猜 x 的平方小於 y，表示猜的太小，並縮小猜值範圍爲〔x，x2〕。

(4) 若所猜的 x 的平方大於 y，表示猜的太大，並縮小猜值範圍爲〔x1，x〕。

(5) 重覆步驟 (2)、(3)、(4)，只要〔x1，x2〕的範圍大於所要求的精密度（小數兩位 0.01 或小數三位 0.001），則要繼續猜；當離開迴圈時，此時的 x1 或 x2 即爲平方根。

程式列印

```
y=9.0
x1=0.0
x2=y
n=1
while (abs(x1-x2)>0.01):#距離>0.01繼續猜
    print("%d:x1=%2.2f,x2=%2.2f" %(n,x1,x2))
    x=(x1+x2)/2
    t=x*x
    if t<y :
        x1=x
    else:
        x2=x
    n=n+1
print(x)#2.99
```

自我練習

1. 請以二分猜值法，求解兩數相除的結果。
2. 請以二分猜值法，求解一正數的立方根。
3. 請寫一程式，由電腦產生一個 1 到 100 的亂數，由使用者用二分猜值法猜，電腦應逐次回答太大、太小、或猜中，且回應幾次猜中。

※ 範例5-6f

秘密差（106/10 APCS 試題）

將一個十進位正整數的奇數位數的和稱爲 A，偶數位數的和稱爲 B，則 A 與 B 的絕對差值 |A － B| 稱爲這個正整數的秘密差。例如：263541 的奇數位數的和 A = 6+5+1 = 12，偶數位數的和 B= 2+3+4 = 9，所以 263541 的秘密差是 |12 － 9|= 3。給定一個十進位正整數 X，請找出 X 的秘密差。

程式列印

1. 前面範例 5-4f 已經有數字拆解範例，這題也是將數字分解的經典題目，參考程式如下：

```
a=263541
a=261
a=2662
a=10000000000000000000000000000000000
c1=0
c2=0
#從個位數開始數字分解
while a>0:
    b=a%10    #得到1個數字
    c1=c1+b   #累加此數字
    a=a//10   #分解
    b=a %10   #再得到1個數字
    c2=c2+b   #累加此數字
    a=a//10    #再分解
print(abs(c1-c2)) #計算絕對值
```

2. 本例因爲測試資料有可能很大，例如 1000 位數，所以還是要以字串解題，程式如下：(本例使用「串列」（如以下的 a[2*i]）處理字串，串列將於下一章介紹。)

```
a="263541"
a="263"
b=len(a)
print(b)
c0=0
c1=0
```

```
if b%2 ==0:
    for i in range(b//2):
        c0=c0+eval(a[2*i])#位置0,2,4
        c1=c1+eval(a[2*i+1])#位置1,3,5
else:
    for i in range(b//2):
        c0=c0+eval(a[2*i])#位置0,2,4
        c1=c1+eval(a[2*i+1])#位置1,3,5
    c0=c0+eval(a[b-1])#最左邊
print(abs(c0-c1))
```

※ 範例5-6g

完全奇數（107/10 APCS 試題）

若一個整數由全部都是奇數組成，我們定義此數爲完全奇數。例如：13311,13199 稱爲完全奇數，13021 就不是。請寫一程式，可以指派一個整數 n，判斷此數是否爲完全奇數。

運算思維

1. 以變數 n 表示此數值。
2. 將此數 n 從個位數一一分解，並判斷此位數是否爲奇數。
3. 因爲數值 n 長度不定，所以應該配合 while 迴圈，只要 n>0 就要持續分解，所以程式如下：

```
n=13242
n=137915
prime=True
while n>0:
    t=n %10
    #只要一個不是，就離開了
    if t %2 ==0:
        prime=False
        break
    n=n//10
if prime:
    print("完全奇數")
else:
    print("不是完全奇數")
```

自我練習

1. 給定一個整數，請往上找出此數的最小完全奇數。
2. 給定一個整數，請往下找出此數的最小完全奇數。

※ 範例5-6h

遞增數。APCS 107 試題

12、345 等稱爲遞增數，53、55 等就不是，請寫一個程式，可以求出 1 到指定輸入數字的遞增數。

運算思維

1. 假設求 1~n 的整數。
2. 使用循序法，逐一列出 1~n 的整數。
3. 逐一將每一個數字從個位數分解。例如，325 分解爲 5,2,3。
4. 逐一兩兩比對，若

   前 1 位數 >= 後一位數

   則提前結束，此非遞增數。本例：

   2>=5

   爲 False，繼續比較。但是

   3>=2

   爲 True，所以比對結束，此非遞增數。以下程式 b2 是前 1 位數，b1 是後一位數，其位置是 b2,b1，比完了，b2 的值給 b1

   b1=b2

   b2 繼續往左邊取，如下表：

   b2=a%10

i	b2	b1
1	2	5
2	3	2

程式列印

1. 以下程式，先判斷一個數值是否全遞增。

```
inc=True
a=256
b1=a%10#先分解最右邊一個
a=a//10
while(a>0):
    b2=a%10  #再分解一個
    a=a//10
    #位置是b2在左邊，b2,b1
    if b2>=b1:   #若成立就沒遞增
        inc=False #找到1個，就離開
        break
    b1=b2#將b2放到b1，b2繼續往左取
print(inc)
```

2. 要找出 1 到 25，再外加一個迴圈，程式如下：

```
n=25
num=0
for i in range(1,n+1):
    inc=True
    a=i
    b1=a%10
    a=a//10
    while(a>0):
        b2=a%10
        a=a//10
        #位置是b2在左邊，b2,b1
        if b2>=b1:
            inc=False
            break
        b1=b2#將b2放到b1，b2繼續往左取
    if inc:
        num=num+1
        print("%d :%d" %(i,num))
```

範例5-6i

實例探討。資產折舊與殘值計算。

運算思維

資產折舊計算有兩種，分別是平均法與定率折算法，分別說明如下：

一、資產剩餘價值平均法公式如下：

殘價 ＝ 取得成本 ÷（ 耐用年數 ＋ 1)【大於耐用年數的只算到這一項】
折舊額 ＝（ 取得成本 － 殘價 ）× 1 /（ 耐用年數 ）×（ 使用年數 ）
扣除折舊後剩餘價值 ＝ 取得成本 － 折舊額

1. 請使用 Python 完成以上計算。

2. 因爲 Python 變數可用中文，所以使用中文變數，程式如下：

```
取得成本=30000
耐用年數=5
使用年數=3
if 使用年數> 耐用年數:
    殘價=取得成本/(耐用年數+1)
    剩餘價值=殘價
else:
    殘價=取得成本/(耐用年數+1)
    折舊額=(取得成本-殘價)*(使用年數/耐用年數)
    剩餘價值=取得成本-折舊額
print(剩餘價值)    #15000
```

3. 開啓司法院的折舊自動試算表 (http://gdgt.judicial.gov.tw/judtool/wkc/GDGT02.htm)，畫面如圖 5-1：可驗證以上計算是否正確。

圖5-1　司法院的折舊自動試算表

二、定率折舊法。

1. 假設汽車買進時 100 萬元，耐用 5 年，每年折舊千分之 369，所以此汽車每年的殘值計算如下：

第 1 年折舊值	100 萬 ×0.369=36.9 萬
第 1 年折舊後價值	100 萬 -36.9 萬 =63.1 萬
第 2 年折舊值	63.1 萬 ×0.369=23.3 萬
第 2 年折舊後價值	63.1 萬 -23.2 萬 =39.8 萬
第 3 年折舊值	39.8 萬 ×0.369=14.7 萬
第 3 年折舊後價值	39.8 萬 -14.7 萬 =25.1 萬
第 4 年折舊值	25.1 萬 ×0.369=9.3 萬
第 4 年折舊後價值	25.1 萬 -9.2 萬 =15.9 萬
第 5 年折舊值	15.9 萬 ×0.369=5.8 萬
第 5 年折舊後價值	15.8 萬 -5.8 萬 =10 萬

2. 請寫程式列出此汽車每年年底的殘值。

3. 以上都是重複連續的計算，每一年都重複計算折舊、計算殘值，使用 for 迴圈最適當了，參考程式如下：

```
a=100 #單位是萬
for i in range(5):
    b=a*0.369    #該年折舊
    a=a-b        #該年殘值
    print("第 %d 年的折舊=%3.1f萬,殘值=%3.1f 萬"%(i+1,b,a))
```

4. 執行結果如下：

```
第 1 年的折舊=36.9萬,殘值=63.1 萬
第 2 年的折舊=23.3萬,殘值=39.8 萬
第 3 年的折舊=14.7萬,殘值=25.1 萬
第 4 年的折舊=9.3萬,殘值=15.9 萬
第 5 年的折舊=5.8萬,殘值=10.0 萬
```

5. 開啓司法院的折舊自動試算表 (http://gdgt.judicial.gov.tw/judtool/wkc/GDGT02.htm)，畫面如圖 5-2：可驗證以上計算是否正確。

圖5-2　司法院的折舊自動試算表

6. 上圖往下捲動，還有詳細計算，如圖 5-3：

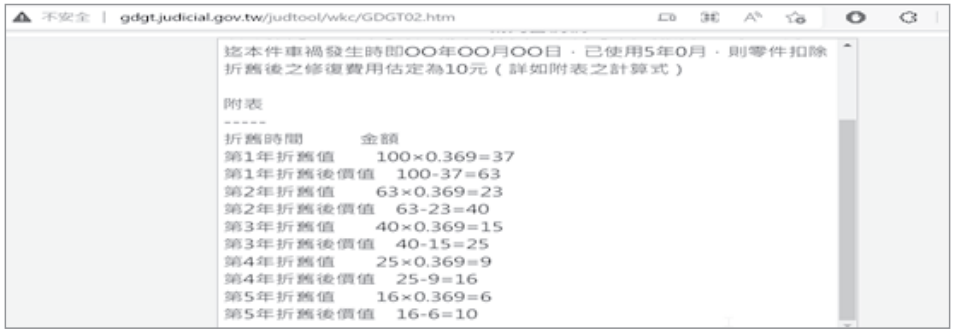


圖5-3 司法院的折舊自動試算表

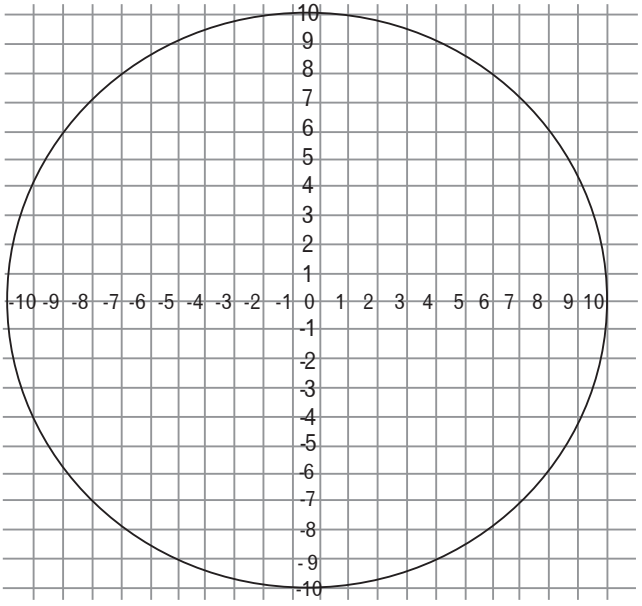
範例5-6j

圓形面積的探討

運算思維

圓形面積我們都知道是「3.14* 半徑 * 半徑」，爲什麼呢？我們已經學了迴圈，以下我們用迴圈闡述此一公式。

1. 取 1 個 20cm*20cm，單位長爲 1cm 的方格紙。
2. 將方格紙座標化，原點 (0,0) 取在中心點。這樣 x 從 -10~10，y 從 -10~10 共畫出 400 個正方形，每個正方形面積是 1，如下圖：
3. 以圓心 (0,0)，畫 1 個半徑爲 10 的圓，如下圖：



4. 一一檢驗所畫的圓共圈進幾個正方形。也就是檢驗每個正方形與圓心的距離平方 x**2+y**2 若滿足小於等於半徑平方 r**2，則此正方形在圓內。
5. 每個正方形的座標 x 軸從 -10~10，y 軸從 -10~10，共 400 個，這正是迴圈最強的運算。根據以上說明，程式如下：

```
s=1
r=10
a=0
for x in range (-10,10+1):#x軸
    for y in range(-10,10+1):#y軸
        if (x**2+y**2)<r**2:#檢驗每一個正方形是否在圓內
            a=a+1
print(a)#305
```

也就是共有 305 個正方形在圓內，所以圓面積推算爲 305。

6. 其次，將方格紙的單位縮小爲 0.1cm，這樣 x 從 -10~10，y 從 -10~10 共畫出 40000 個正方形，每個正方形面積是 0.01，統計圓內的正方形個數的程式調整如下：（Python 的 range 僅能整數，本例也是先將 range 放大 10 倍（10*s），計算時再縮小回去 (x/s)）也就是共有 305 個正方形在圓內，所以圓面積推算爲 305。

```
s=10  #切成0.1*0.1的正方形
a=0
for x in range (-10*s,10*s+1):
    for y in range(-10*s,10*s+1):
        if ((x/s)**2+(y/s)**2)<r**2:
            a=a+1
print(a*0.01)#313.97
```

也就是共有 31397 個正方形在圓內，每個正方形面積是 0.01，所以圓面積推算爲 313.97。

7. 方格紙單位繼續縮小爲 0.01，這樣 x 從 -10~10，y 從 -10~10 共畫出 4000000 個正方形，每個正方形面積是 0.0001，統計圓內的正方形個數程式調整如下：也就是共有 31397 個正方形在圓內，每個正方形面積是 0.01，所以圓面積推算爲 313.97。

```
s=100#切成0.01*0.01的正方形
a=0
for x in range (-10*s,10*s+1):
    for y in range(-10*s,10*s+1):
        if ((x/s)**2+(y/s)**2)<r**2:
            a=a+1
print(a*0.0001) #314.1529
```

8. 我們發現，只要將正方形分的更多更小，精密度就會出來，但分的越多，計算將會更繁瑣，所幸有電腦迴圈任勞任怨的快速計算功能，所以可解決很多數值分析問題。同學若繼續學習大學數值分析與積分問題，將會發現很多問題都可以用迴圈求解。

範例5-6k

實例探討。銀行借貸本金平均攤還實例探討。

銀行借貸，若採用本金平均攤還法，此方法的計算方式如下：

1. 計算平均每月應攤付本金金額。
   每月應攤付本金金額＝貸款本金 ÷ 還款總月數 (公式中：還款總月數＝貸款年期 ×12)
2. 計算每月應還利息。每月應付利息金額＝本金餘額 × 月利率。(月利率＝年利率 ÷12）
3. 每月應付本息金額＝每月應還本金＋每月應付利息
4. 以上計算結果可參考富邦銀行，如下圖所示。請同學自行開啓網頁，計算每月應該還款金額。

5. 以上計算，因爲每月固定還本金，每個月都要重複計算本金餘額與利息，這也是很典型迴圈的應用，程式如下：

```
a=1000000
b=0.01#年利率
c=1 #貸款總期數
d=round(1000000/(c*12))#每月還款本金
#round是四捨五入函式
rate=b/12
print("每月攤還本金 每月利息 每月攤還金額 剩餘本金")
for i in range(c*12):
    m=round(a*rate)   #每月利息
    a=a-d             #每月本金餘額
    print(("%11d%8d%7d%8d")%(d,m,d+m,a))
```

執行結果如下圖

```
每月攤還本金 每月利息 每月攤還金額 剩餘本金
      83333      833      84166   916667
      83333      764      84097   833334
      83333      694      84027   750001
      83333      625      83958   666668
      83333      556      83889   583335
```

5-7 本章內容摘要

1. Python 的重複結構有 for 與 while，這兩種迴圈的主要差別是，前者是程式設計階段就知道迴圈執行次數，所以又稱爲計數迴圈；而後者是設計階段不知道，要執行後才能知道執行次數，所以稱爲條件迴圈。
2. for 用法如下：

```
for i in range(1,4,1):
    print(i) #1,2,3
```

3. while 主要用於設計階段不知迴圈執行次數。例如：

```
a=8#被除數
b=3#除數
q=0#商
while(a>=b):#只要(被除數>=除數)  就執行迴圈
    a=a-b    # (被除數)-(除數)
    q=q+1    #商每次遞增1
print(q)# 商數
print(a)#餘數
```

6. break 可提早離開迴圈。
7. continue 可略過迴圈未執行部分，提早重回迴圈。
8. Python 不允許空迴圈、空函式，pass 可當作空敘述。

課後評量

一、填充題（請寫出以下程式執行結果）

題號	題目	執行結果
1	`for i in range(1,4,1):` `    print(i)`	
2	`for i in range(4,1):` `    print(i)`	
3	`for i in range(8,2,-2):` `    print(i)`	

4	`a=8;b=3;s=0`
	`for i in range(b):`
	`    s=s+a`
	`print(s)`	

5	`s=0`
	`for i in range(5):`
	`    s=s+i`
	`print(s)`	

6	`for i in range(1,10,1):`
	`    if i==4:`
	`        break`
	`    print(i)`	

7	`for i in range(1,10,1):`
	`    if i==6:`
	`        continue`
	`    print(i)`	

8	`i=11`
	`b=True`
	`print(i)`
	`for j in range(2,i):`
	`    if i %j ==0 :`
	`        b=False`
	`        break`
	`if b==True:`
	`    print("prime")`
	`else:`
	`    print("not prime")`	

9	a=8;b=3;q=0
	while(a>=b):
	    a=a-b
	    q=q+1
	print(q)
	print(a)	

10	a=13;n=2;r=0;d=''
	while a>0:
	    r=a % n
	    d=d+str(r)
	    a=a//n
	print(d)	

11	a=13;n=2;r=0;d=0
	while a>0:
	    r=a % n
	    d=r+d
	    a=a//n
	print(d)	

12	a=987
	while a>0:
	    b=a%10
	    print(b, end=' ')
	    a=a//10	

13	a=21;b=9
	while r>0:
	    a=b
	    b=r
	    r=a%b
	print(b)	

14	for i in range(2,12+1):
	    b=True
	    for j in range(2,i):
	        if i %j ==0 :
	            b=False
	    if b==True:
	        print("%3d" %i,end="")	

15	for i in range(5+1):
	    a=" " *(5-i)
	    b="*"*i
	    print(a,end='')
	    print(b)	

16	n=5
	for i in range(n):
	    for j in range(5-i):
	        print('%2d'%(i+1), end='')
	    print()	

17	n=5
	for i in range(n):
	    for j in range(i+1):
	        print(j+1,'', end='')
	    print()	

18	n=5
	for i in range(n):
	    for j in range(n-i-1):
	        print(' ', end='')
	    for k in range(i+1):
	        print(k+1,'', end='')
	    print()	

19	a=16
	for i in range(9):
	    if i*i>=a:
	        print(i)#3
	        break	

20	for x in range(-10,10+1,1):
	    if x**2-4*x-5==0:
	        print(x)	

21	x1=20;x2=40;right=False
	while not(right):
	  x=(x1+x2)//2
	  print('電腦猜x=%d'%x,end='')
	  a=input("Please input 3(太大),2(猜中),1(太小):")
	  if a=='2':
	  right=True
	  break
	  elif a=='3':
	  print('太大,',end='')
	  x2=x-1
	  else:
	  print('太小,',end='')
	  x1=x+1
	print("Bingo,The number is %d"% x)
	以上題目執行後，於所出現的輸入提示，依序輸入3,1,3,2，
	請問「Bingo,The number is」的數字爲何？	

22	y=16.0;x1=0.0;x2=y
	while (abs(x1-x2)>0.01):
	  x=(x1+x2)/2
	  t=x*x
	  if t<y :
	  x1=x
	  else:
	  x2=x
	print(x)	

23	a=263541;c1=0;c2=0
	while a>0:
	  b=a%10 #得到1個數字
	  c1=c1+b #累加此數字
	  a=a//10 #分解
	  b=a %10 #再得到1個數字
	  c2=c2+b #累加此數字
	  a=a//10 #再分解
	print(abs(c1-c2)) #計算絕對值	

24	n=137915;prime=True
	while n>0:
	　　t=n %10
	　　if t %2 ==0:
	　　　　prime=False
	　　　　break
	　　n=n//10
	if prime:
	　　print("完全奇數")
	else:
	　　print("不是完全奇數")	

25	inc=True
	a=256
	b1=a%10
	a=a//10
	while(a>0):
	　　b2=a%10
	　　a=a//10
	　　#位置是b2在左邊，b2,b1
	　　if b2>=b1:
	　　　　inc=False
	　　　　break
	　　b1=b2#將b2放到b1，b2繼續往左取
	print(inc)	

26	sum=0;dx=1 for x in range(1,10+1): 　　y=x 　　sum=sum+y*dx print(sum)	

本章習題

※ 1. e= 2.718 的由來

工程與科學的指數與對數計算，通常不是以 2 或 10 爲底，而是以 e 爲底，我簡單闡述如下：

(1) 假設借款金額爲一元，言明年利率爲 100%，每年複利一次，則一年後本利和爲 2 元。

$$1\times(1+1)^{1}=2$$

(2) 假設借款金額爲一元，言明年利率爲 100%，每月複利一次，則一年後本利和爲 2.613。

$$1\times(1+\frac{1}{12})^{12}=2.613$$

(3) 假設借款金額爲一元，言明年利率爲 100%，每日複利一次，則一年後本利和爲 2.714。

$$1\times(1+\frac{1}{365})^{365}=2.714$$

(4) 假設借款金額爲一元，言明年利率爲 100%，每秒複利一次，則一年後本利和爲 2.718。

$$1\times(1+\frac{1}{365\times24\times60\times60})^{365\times24\times60\times60}=2.718$$

(5) 細菌的繁殖、電容的充放電等，這些都是數量非常龐大的成長現象，都要用 e 來解釋才能理解。也就是數量非常龐大的物件，若其一年「平均」成長 2 倍，則一年後總數量會成長爲 2.718 倍，而不是 2 倍。

(6) 請根據以上演算法，求出 e 值。

2. 假設銀行利息採用單利計算，存款 1 萬元，年利率是百分之一，請列出今後十年，每年年底的本例和。

3. 假設銀行利息採用複利計算，每年複利 1 次，存款 1 萬元，年利率是百分之一，請列出今後十年，每年年底的本例和。
4. 假設銀行利息是單利計算，年利率百分之一，每年年初存 1 萬，請問今後十年的年底的本利和是多少。
5. 假設銀行利息是複利計算，年利率百分之一，每年年初存 1 萬，請問今後十年的年底的本利和是多少。
6. 定額扣款。假設欠款 100 萬，年利率百分之一，每月月底還款 2 萬，請問多久可還完，最後 1 期是還多少。

MEMO

附錄1：本書中英文索引對照表

英文	中文	頁碼
Abstraction	抽象化	1-14
Alrorithm	演算法	1-4
Arithmetic Operators	算術運算子	3-4
Assignment Operators	指派運算子	3-5
Associativity	結合律	3-12
Bool Value	布林值	3-3
Char	字元	3-3
Comments	註解	3-18
Constant	常數	3-4
Debug	除錯	3-37
Encapsulation	封裝	1-15
Float	浮點實數	3-2
Flow Char	流程圖	1-5
Inheritance	繼承	1-15
Instance	樣例化	1-13
Integer	整數	3-2
Logical Operators	邏輯運算子	3-8
Nature Language	自然語言	1-5
NonProcddure Oriented	非程序導向	1-11
Object	物件	1-13
Object Oriented	物件導向	1-13
Operand	運算元	3-4
Operator	運算子	3-4

英文	中文	頁碼
Polymorphism	多型	1-15
Procddure Oriented	程序導向	1-12
Procedence	優先順序	3-11
Programming	程式設計	1-4
Programming Language	程式語言	1-4
Pseudo Code	虛擬碼	1-5
Relational Operators	關係運算子	3-8
String	字串	3-3
Subroutine	副程式	1-5
Variable	變數	3-4

國家圖書館出版品預行編目 (CIP) 資料

程式語言與設計 / 洪國勝 , 蔡懷文 , 蔡懷介 , 陳蘊慈編著 . -- 初版 . -- 高雄市 : 泉勝出版有限公司 , 民 112.04
冊 ; 公分
技術型高級中等學校商業與管理群
ISBN 978-986-99632-4-4(上冊 : 平裝). --
ISBN 978-986-99632-5-1(下冊 : 平裝). --
ISBN 978-986-99632-6-8(全套 : 平裝)
1.CST: 電腦教育 2.CST: 電腦程式設計 3.CST: Python(電腦程式語言) 4.CST: 技職教育

528.831 112002894

十二年國民基本教育
技術型高級中等學校商業與管理群
程式語言與設計（上冊）
審定字號：技審字第 112006 號
書號：SB001
編著者：洪國勝、蔡懷文、蔡懷介、陳蘊慈
責任編輯：洪國勝
排版封面設計：陳彥慈（芒現）
發行所：泉勝出版有限公司
地址：高雄市左營區華夏路 708 號 17 樓
電話：0900757159 LINE ID：5abook
網址：www.goodbooks.com.tw
Email:aa163677@yahoo.com.tw
訂購方式：電話：0900757159 LINE ID：5abook Email:aa163677@yahoo.com.tw
初版日期：112 年 4 月
版次：初版第一刷
定價：280
ISBN：978-986-99632-4-4