高中資訊科技
使用Python程式設計

谷町、洪國勝　編著

谷町簡介

1. 台灣大學資訊工程博士
2. 曾經擔任大學多項程式設計競賽命題工作 (CPE、PUPC、TUPC、NCPC、ICPC 等)
3. 曾經擔任 TOI 臺灣國際資訊奧林匹亞競賽選訓營培訓教練 (IOI 隨隊教練)

泉勝出版有限公司
www.goodbooks.com.tw

編輯大意

目前高中資訊科技課綱有點太多、也太深入，有些章節，例如樹與圖形，是大學資工的精華，但卻硬生生塞給高一學生，高一學生連程式都不會，卻硬要學樹與圖形，有點不會走路就要學飛行，造成老師與學生困擾，本書則抽取資訊科技課綱精華—程式設計，給予加強闡述，若此一實用章節能學有成就，高一學會程式設計運算思維，其實也就夠了。

教師配套

1. 提供電腦批改程式功能：以往學生考程式設計，老師必須不斷走動幫學生看執行結果，批改成績，每考一題，每個學生就要走動看一次，全班 30 人，就是看 30 次，老師真的很累。本系統特色是，老師不用走動，老師只要勾選教材範例或自我練習題目，或依照我們給的教學方式，完成命題，學生就可以開始考試，電腦就會自動批改對或錯。
2. 程式考試時，有將學生電腦的複製 / 貼上功能取消，學生無法作弊。
3. 程式考完，每個學生程式也存在資料庫，此時可以統計每個學生程式使用的字數，若發現有學生字數相同，表示有作弊嫌疑，可以認真比對程式，揪出抄襲者。
4. 線上題庫評量系統：提供每一個學生都有帳號，且此系統所有的考試，包含程式設計題、選擇、填充的成績都在同一系統，都可以儲存考試成果，學期末老師只要指定每次考試的比重，就可以算出學期成績。（備註 1：也可以新增欄位，填入作業與職業道德分數。備註 2：學生可使用電腦、手機考試，或輸出紙本考試）。
5. 隨時更新的題庫 : 作者已經將教材範例與自我練習做成題庫，也可由任課老師與作者隨時更新，老師只要勾選題目，學生即可使用電腦作答，程式設計題也是，電腦就可以批改程式或選擇與填充題。

6. 不定期提供最新 APCS 術科試題解答：APCS 題目眞的難，其主要功能是選拔有天分的學生參加奧林匹亞比賽，本系統則有谷町老師主動協助，提供一些歷屆試題影音教學影片，讓有興趣的老師與學生觀摩與學習，期望挖掘更多軟體優秀人才。或者老師也可以將 APCS 試題丟給出版公司，我們會請谷町教授提供影音教學連結檔案。
7. 教師可任選泉勝圖書 5 本，放在研究室，讓優秀學生參考。

學生配套

1. 每一學生都有帳號，都可使用線上題庫評分系統。

谷町、洪國勝

www.goodbooks.com.tw

目錄

▶第 1 章　程式語言基本概念

1-1　程式語言與程式設計的演算法 1-1
1-2　程式語言開發環境的操作 1-4

▶第2章　程式組成與語法操作

2-1　程式基本架構 2-1
2-2　程式語法規則 2-3
2-3　資料型態 2-5

▶第3章　資料的基本運算

3-1　數值的運算 3-1
3-2　字串運算 3-13
3-3　時間的處理 3-16
3-4　聲音的處理 3-18
3-5　標準輸出輸入的操作 3-21
3-6　運算子與運算元的應用 3-26
3-7　程式的除錯（Debug） 3-34

▶第4章　選擇結構程式設計

4-1　結構化程式設計架構 4-1
4-2　單一選擇結構敘述與撰寫 4-2
4-3　多重選擇結構敘述與撰寫 4-6
4-4　巢狀選擇結構敘述與撰寫 4-11
4-5　選擇結構程式實作演練 4-15

▶第5章　重覆結構程式設計

5-1　迴圈基本架構 5-1
5-2　計數迴圈敘述與撰寫 5-1
5-3　變更迴圈流程的程式語法撰寫 5-6
5-4　條件迴圈敘述與撰寫 5-8
5-5　巢狀迴圈敘述與撰寫 5-18
5-6　重覆結構程式實作演練 5-24

▶第6章　串列程式設計

6-1　串列（list）..... 6-1
6-2　一維串列 6-1
6-3　二維串列 6-12
6-4　串列函式與串列生成式 6-23

▶第7章　函式應用

7-1　模組（Module）..... 7-1
7-2　內建函式庫的認識與應用 7-2
7-3　自定函式 7-26
7-4　函式應用實例演練 7-33

▶第8章　綜合應用實例

Chapter
01

程式語言基本概念

1-1 程式語言與程式設計的演算法

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）

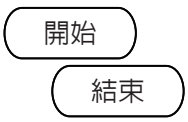
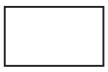
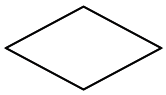
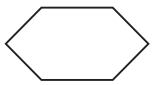
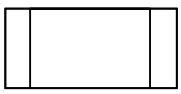
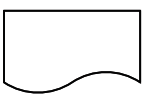
在自然語言中，有時嵌入一些慣用程式敘述，如 if 、switch 或 match 代表決策（第四章介紹）for、while 代表迴圈（第五章介紹），如此可縮短文字長度，也會讓敘述更加清楚易懂。例如，以下是求 9 開根號的虛擬碼演算法：

```
 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		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。
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 1-1 是求 9 開根號的流程圖演算法。

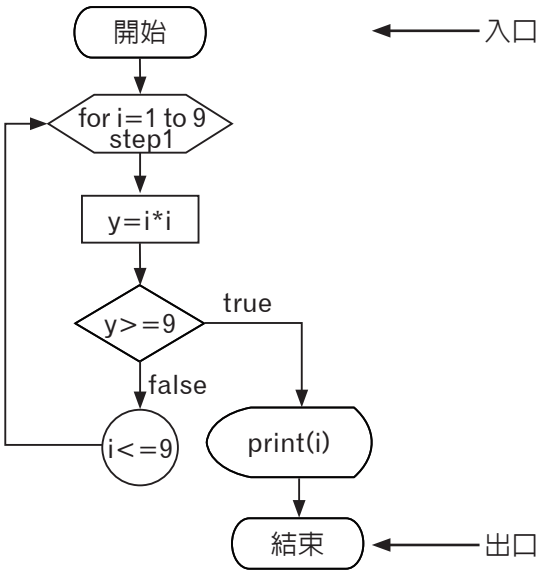


圖1-1 開根號求解流程圖

▶ 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，但因爲 Anaconda 已經幫忙蒐集與預置初學者常用模組，所以卻是目前最實用、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 1-2，請依下圖點選「Free 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。

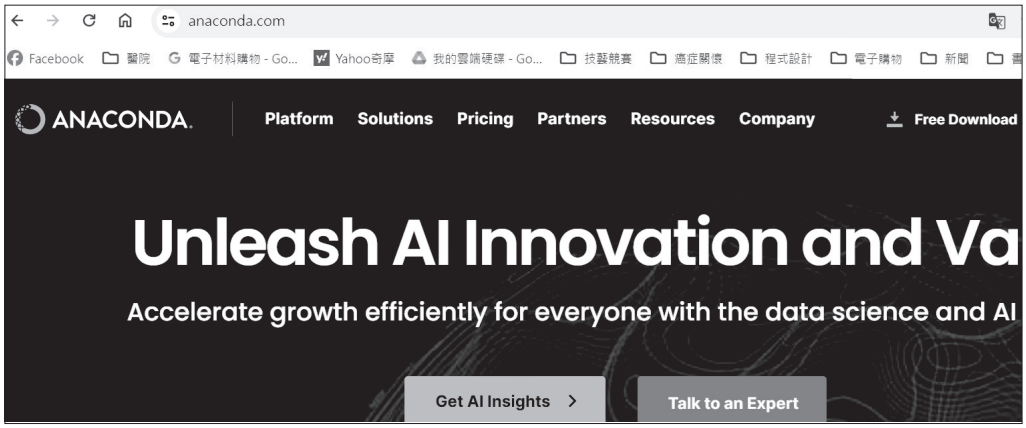


圖1-2　Anaconda官網首頁

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 1-3。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。

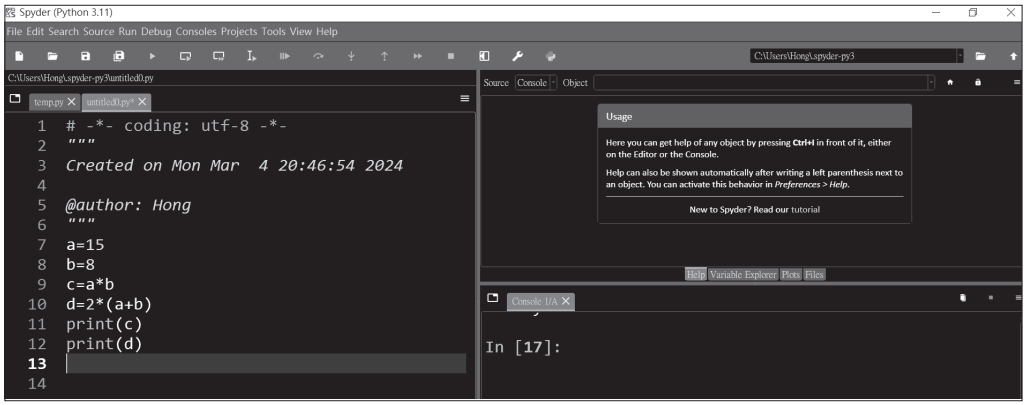


圖1-3　Spyder整合視窗

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 1-4 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

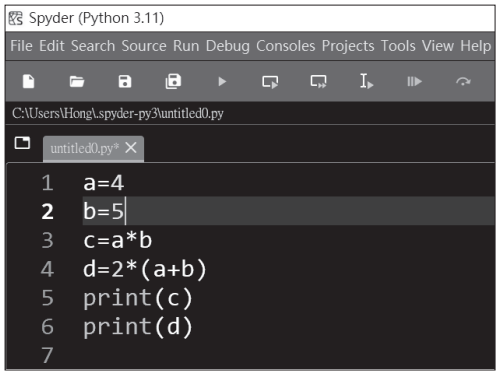


圖1-4　Spyder編輯視窗

程式的執行

圖 1-5 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。

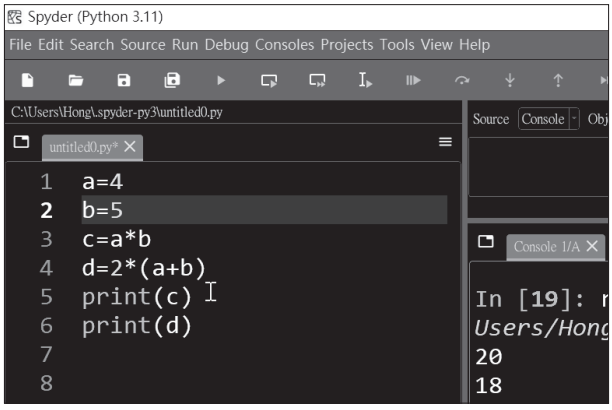


圖15　Spyder執行結果畫面

修改編輯環境

點選功能表的「Tools/Preferences/Appearance」，畫面如圖 1-6，可在此修改程式字型的大小。

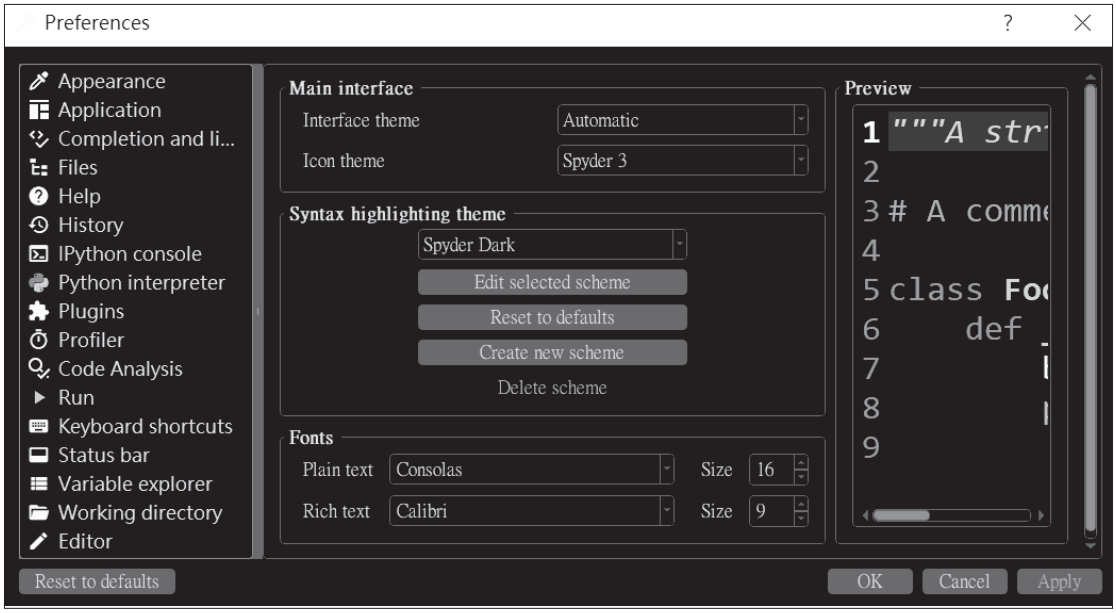


圖1-6　Preferences微調視窗

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 )
```

MEMO

Chapter
02

程式組成與語法操作

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 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：

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=-1234567890123456789012345678901234567890
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)

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。

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"
```

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)                                    _________
```

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。

Chapter
03

資料型態與運算

我們日常生活常見的資料有數值、字串、時間、與聲音，本節將介紹這些資料的基本運算。

3-1 數值運算

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

表3-1 Python算術運算子（續）

運算子	定義	優先順序	結合律
//	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，可以指派一個四位數字，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。
5. 找錢程式。

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
`<<, >>`	Shifts	8
`+, –`	Addition and subtraction	7
`*, @, /, //, %`	Multiplication, matrix multiplication, division, floor division, remainder 5	6
`+x, -x, ~x`	Positive, negative, bitwise NOT	5
`**`	次方	4

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 Python 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。

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 模組的 randint() 方法。使用 random 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稱。例如：

```
import random as a # a是程式設計者自己取的物件名
b=a.randint(1,6) #產生1～6整數亂數
print(b)
```

以上 randint() 稱爲物件的方法 (method)，僅附屬在該物件。

► 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

前面已經介紹運算子與函數，Python 是物件導向語言，所以也提供物件的方法來寫程式，使用物件的方法，語法如下：

```
物件.方法()
```

例如：以上 a 即爲物件。

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。

3-2 字串運算

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

字串常需要依照某些條件分割，字串的分割是使用字串物件的 split 方法。例如：

```
a="113/3/4"
y,m,d=a.split("/")
print(y,m,d) #113 3 4
```

又例如：

```
b="13:20:12"
h,m,s=b.split(":")
print(h,m,s) #13 20 12
```

以上 a="113/3/4"，a 雖然是變數，但物件導向語言也視爲物件。

3-3 時間的處理

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() 方法分解字串。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)                                  _____
```

以上分割的結果是字串，若要數值處理，則應使用 int() 函數。例如：

```
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。

```
a="113/315"
y,m,d=map(int,a.split('/'))
print(y+m+d)                                    _____
```

範例3-3a

請計算以下兩個時間相差的時間間隔。

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

運算思維

時間的運算，國小數學的方法是先處理秒，再處理分與時，秒不夠時，向分借 1，分不夠時，向時借 1，但電腦因爲計算能力強，所以通常是將時間通通轉爲秒，然後直接相減，減完再轉換爲「時：分：秒」的格式，程式如下：

```
a="18:30:22"
b="20:10:42"
h1,m1,s1=map(int,a.split(":"))
t1=h1*60*60+m1*60+s1
h2,m2,s2=map(int,b.split(":"))
t2=h2*60*60+m2*60+s2
t=t2-t1
h=t//3600
t=(t-h*3600)
m=t//60
s=t %60
print("%2d:%2d:%2d"%(h,m,s))
```

自我練習

1. 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"
```

2. 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"
```

3-4 聲音的處理

要讓電腦發聲，要先載入 ctypes 模組，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。

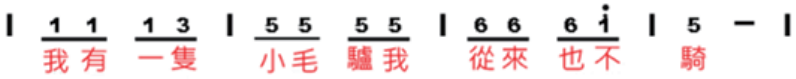


圖3-1　小毛驢簡譜

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
```

格式化的字串文本 (Formatted String Literals)

格式化的字串文本（簡稱爲 f- 字串），透過在字串加入前綴 f 或 F，並將運算式編寫爲 {expression}，讓你可以在字串內加入 Python 運算式的值。例如：以下四個變數，都佔 6 格，字串預設靠左，數值預設靠右。

```
import math#載入math模組
a="Mary"
b,c=1,200
d=math.pi #使用math模組的pi值
print(f'{a:6s} b={b:4d} c={c:4d} d={d:6.2f}')
```

輸出結果如下圖：

```
12345678901234567890123456789012
a=Mary  b=     1c=   200d=  3.14
```

字串的 format() method

以上格式化的字串文本，亦可使用字串的 format() 方法，例如以上程式使用字串的 format() 方法，程式如下：

```
import math
a="Mary"
b,c=1,200
d=math.pi
print("12345678901234567890123456789012")
print('a={0:6s}b={1:6d}c={2:6d}d={3:6.2f}'.format(a,b,c,d))
```

以上大刮號內的 0,1,2,3 將會分別對應後面 format() 方法，以上執行結果與上一程式相同。

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：

   $ax^2 + bx + c = 0$

   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）。

提示：

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

(6) 請根據以上運算思維，寫出程式。

提示：以上程式比較冗長，等學習迴圈就能簡化程式，體會程式設計的樂趣。

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#輾轉相除法 b=9 r=a%b print(r) a=b b=r r=a%b print(a) print(b) print(r)	

2	a=6 #十進位轉二進位
	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 #計算數字和與數字長度
	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)	

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 3-2 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 3-3「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。

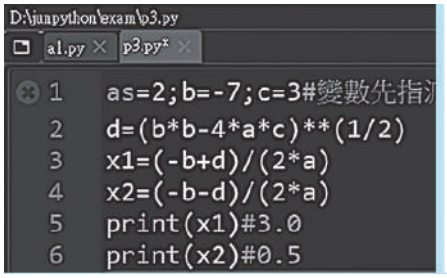


圖3-2 Spyder解譯指令畫面

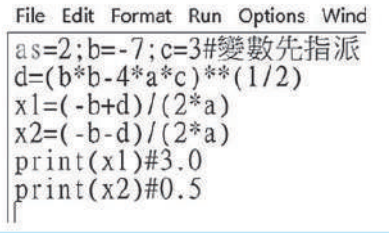


圖3-3 Python IDLE解譯指令畫面

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 3-4，請留意第 1 行敘述出現箭號。

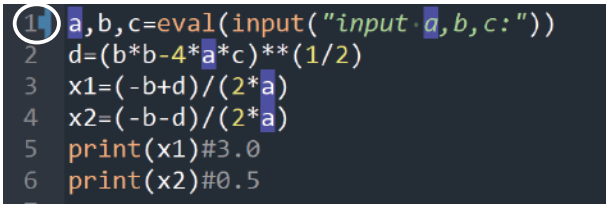


圖3-4　Spyder Debug畫面

逐一點選工具列的「Run current line」，即可逐一執行箭號所在敘述。圖 3-5 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 3-6，此時即可觀察每個變數的內容。

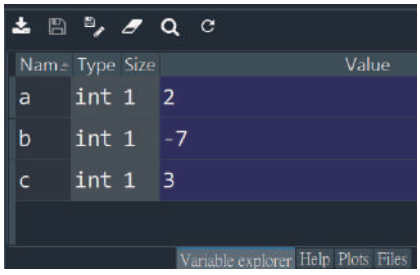


圖3-5　變數值窗格（一）

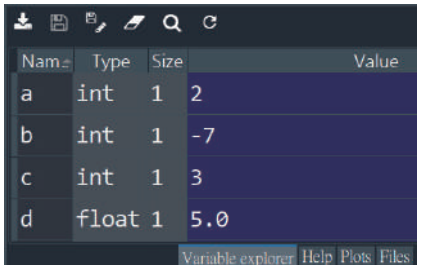


圖3-6　變數值窗格（二）

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。

Chapter
04

選擇結構程式設計

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：

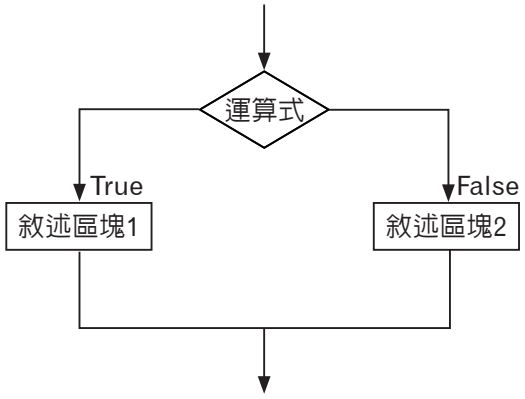


圖4-1 if ～ else~流程圖

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(:)

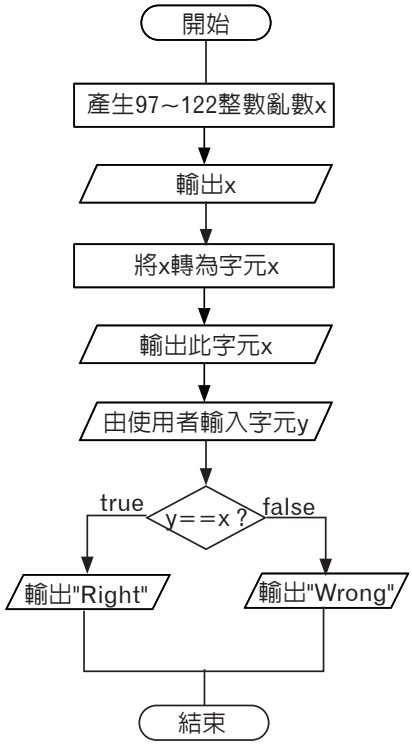
冒號 (:) 也是 Python 特有的符號，往後的 for、while、函式等，都需要這個符號。

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。

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：

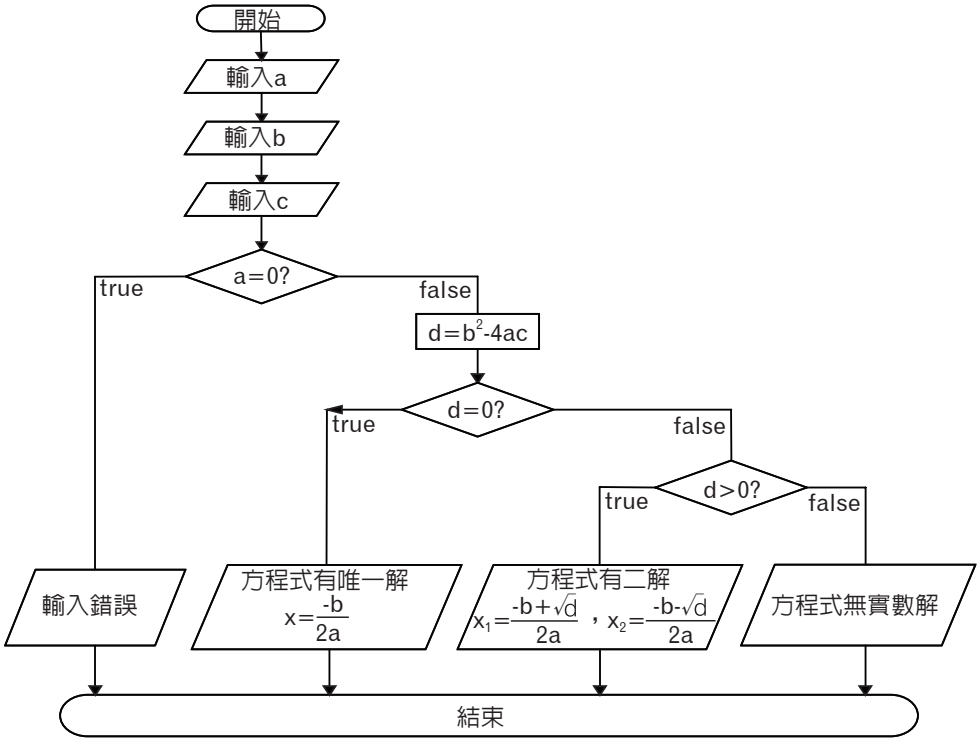
```
aa,60
bb,30
cc,80
```

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。

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

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

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 $d=\dfrac{1}{2}(a+b+c)$

(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。
2. 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下表。

重量(公克)	不逾 20	21~50	51~100	101~250	251~500	501~1000	1001~2000
郵資	8	16	24	40	72	112	160

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。

3. 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。

Chapter
05

重覆結構程式設計

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

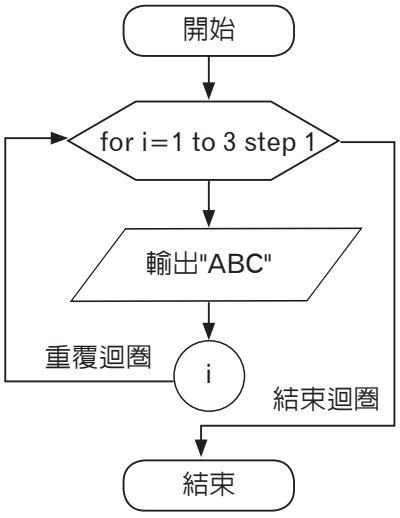
Python 計數迴圈指令爲 for，for 指令的基本語法如下：

```
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，$a_1+a_2 2^1+a_3 2^2+a_4 2^3\cdots$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。
   $(a)_{10}=(a_n\cdots a_3 a_2 a_1 a_0)_2$

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
```

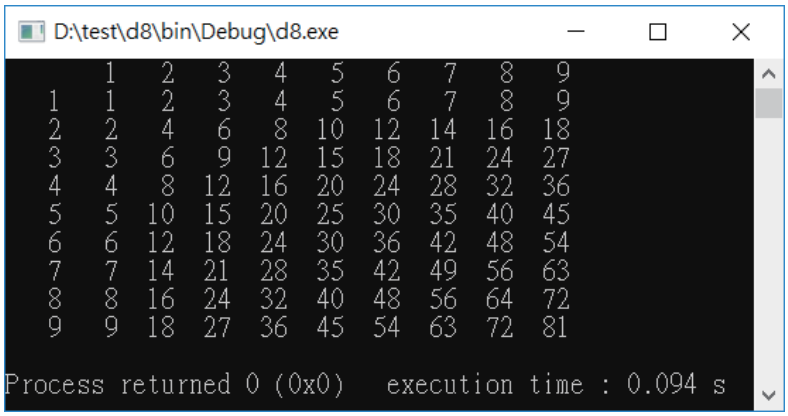
程式列印

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。

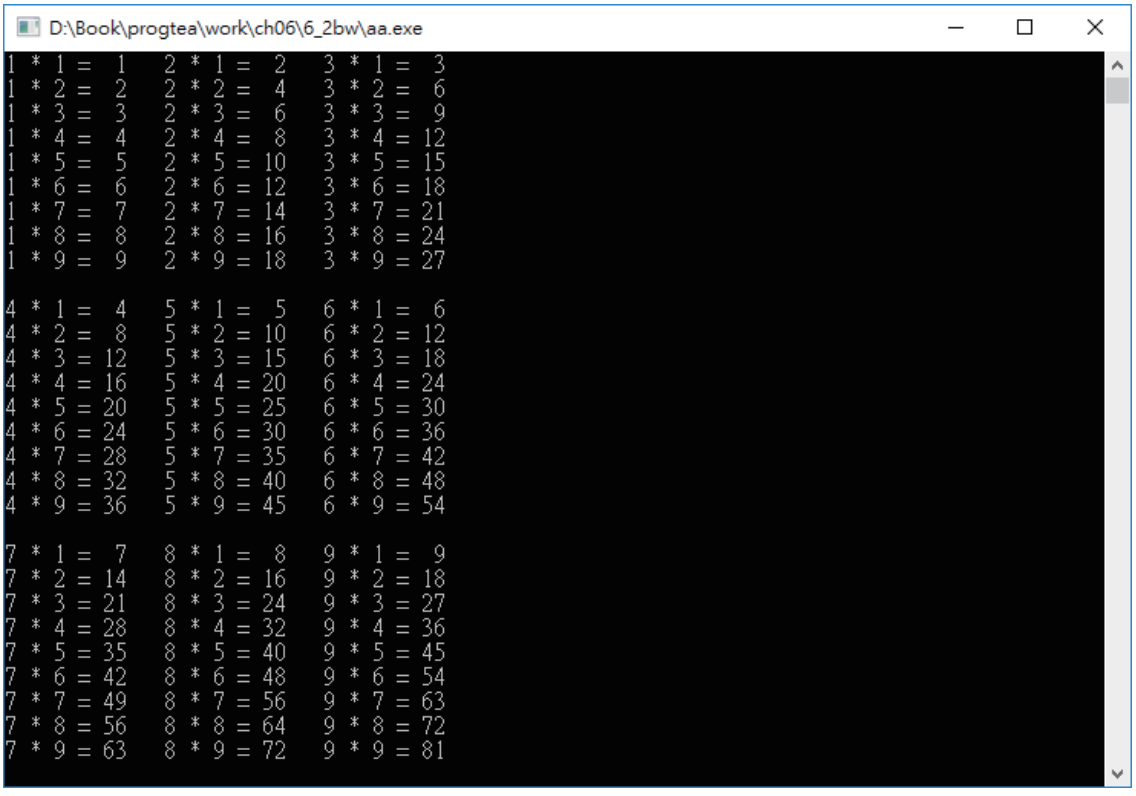
```
for i in range(1,9+1):
    for j in range(1,9+1):
        print("%d*%d=%2d"% (i,j,i*j),end=" ")
    print()
```

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

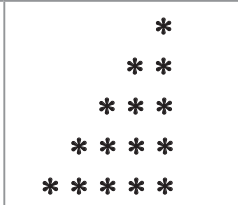
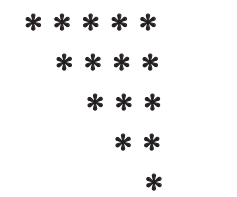
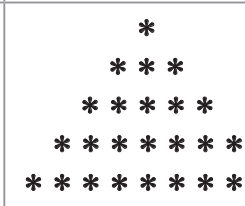
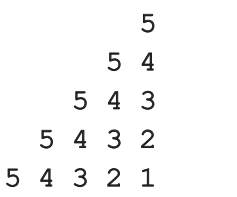
1. 本例的主要目的是訓練學生，熟練雙迴圈的內外迴圈解析，也就是要找其關係，這樣可以減少變數的數量，有助於後續複雜程式的簡化，例如往後串列與排序的應用。

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	1 1 1 1 1 2 2 2 2 3 3 3 4 4 5	6	

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 $b_1b_2b_3$…。
      (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))
```

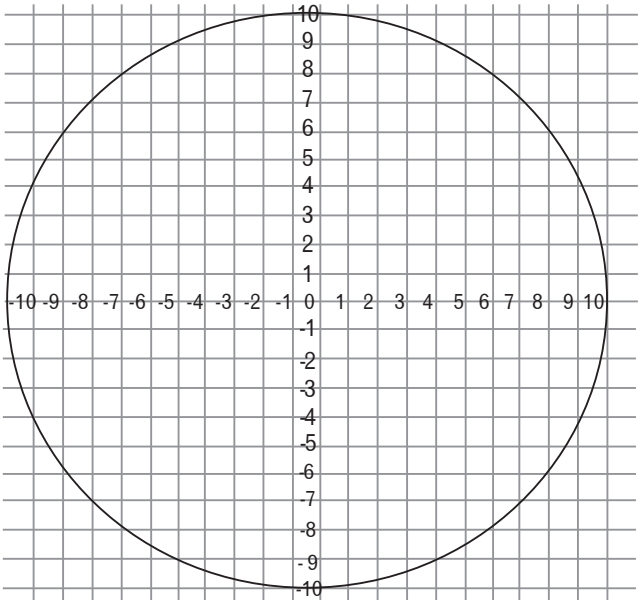
範例5-6i

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。

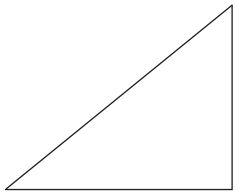
範例5-6j

積分的剖析。

演算法

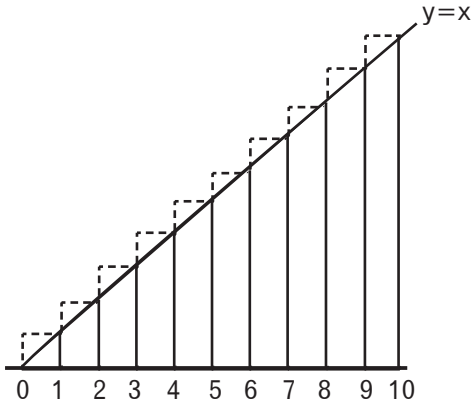
積分的計算就如同將一個不規則的圖形，先切割成一條條很細的長條矩形，因爲每個長條矩形很細，其面積趨近於底乘以高，再累加這些長條矩形面積，即爲其總面積，例如，若有積分式如下：

$$\int^{10} xdx$$



此積分式就如同求右圖邊長爲 10 的等腰三角形面積，則其面積依照公式是（底 * 高）/2 ＝ 50。此公式的由來，解析如下：

1. 若 dx=1，也就是將三角形切爲 10 個矩形，每個矩形的寬爲 1，高度則遵循 y=x 的關係，隨著當時的 x 而變化，如下圖所示：



每一矩形面積如下：

```
dx*y
```

有兩個變數，但此兩變數相依，其關係是 y=x，所以要先簡化變數個數如下：

```
dx*x
```

共有 10 個矩形，dx=1，
第 1 個矩形 x=1，所以面積 =1*1（x=1 時，y=1）
第 2 個矩形 x=2，所以面積 =1*2（x=2 時，y=2）
第 3 個矩形 x=3，所以面積 =1*3（x=3 時，y=3）
第 4 個矩形 x=4，所以面積 =1*4（x=2 時，y=2）
第 5 個矩形 x=5，所以面積 =1*5（x=5 時，y=5）
第 6 個矩形 x=6，所以面積 =1*6（x=6 時，y=6）
第 7 個矩形 x=7，所以面積 =1*7（x=7 時，y=7）
第 8 個矩形 x=8，所以面積 =1*8（x=8 時，y=8）
第 9 個矩形 x=9，所以面積 =1*9（x=9 時，y=9）
第 10 個矩形 x=10，所以面積 =1*10（x=10 時，y=10）
三角形面積就是累加以上 10 個矩形面積，所以其面積 =55，那也合理，因爲由上圖可知，我們多算了 10 個邊長爲 1 的直角三角形。根據以上推演，程式撰寫如下：

```
sum=0
dx=1
for x in range(1,10+1):
        y=x
        sum=sum+y*dx
print(sum)#55
```

2. 若 dx=0.1，也就是將三角形切爲 100 個矩形，則其面積如下：

```
sum=0
s=10
```

```
dx=1/s
for x in range(1,(10*s+1)):
        y=x/s
        sum=sum+y*dx
print(sum)#50.5
```

結果是 50.5。

3. 若 dx=0.01，也就是將三角形切爲 1000 個矩形，則其面積如下

```
sum=0
s=100
dx=1/s
for x in range(1,(10*s+1)):
    y=x/s
    sum=sum+y*dx
print(sum)#50.05
```

結果是 50.05。

4. 若 dx 取的非常小，也就是切爲無限多個矩形，則其結果將會是 50，這就是積分的道理。

自我練習

1. 使用以上積分法，求解四分之一圓面積，四分之一圓面積 = $\int_0^1 \sqrt{1-x^2}dx$ 。（圓方程式是 $x^2+y^2=1$）

2. 使用以上積分法求解 sin(x) 正半週面積，sin(x) 正半週面積 $\int_0^{\pi} \sin x dx = 2$ 。（此爲高職基本電學一年級題目，sinx 可使用 import math; math.sin(x)）

※ 3. 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。

4. 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。
5. 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。
6. 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。
7. 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。
8. 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。

Chapter
06

串列程式設計

6-1 串列（List）

處理少量的資料，可以個別設定一些單一變數，但是如果資料龐大，例如，若有資料 3,8,4,7,2,9 要求極大、極小、平均等運算，若還是使用 6 個變數儲存，程式將會很冗長，本章將會介紹另一種資料結構『串列』，然後配合上一章的 for 或 while 迴圈，就可以以非常精簡的程式，處理以上大批資料。其次，串列還有維度之分，分別是一維、二維、三維等等，就要看您資料運算的方式，若只有同批資料的關係，例如，上例就是一維資料串列就可以，請看 6-2 節。但是，在一個班級學生裡，每個人都有國、英、數、自然、社會等 5 科成績，我們不僅要求縱向的各科平均，還要求橫向的每人平均，那就需要使用二維串列，請看 6-3 節。（補充說明：C/C++、Java 等連續資料的集合稱爲 Array 翻譯爲陣列，Python 則稱爲 List，翻譯爲串列。）

6-2 一維串列

一維串列宣告

建立一個連續數值的一維串列的方法很多種，最常見語法如下：

```
串列名稱=[初值 for i in range(串列長度)]
```

例如：

```
n=5
a=[0 for i in range(n)]
```

則可使用 a[0],a[1],a[2],a[3],a[4] 等共 5 個記憶體位置，且其初值均爲 0，以上 0~4 稱爲串列索引，往後都是使用索引存取串列元素。又例如：

```
n=4
b=[i for i in range(n)]
print(b)#輸出串列[0,1,2,3]
```

以上語法也可簡化如下：

```
串列名稱=[初值]*串列長度
```

例如：

```
a=[0]*5
```

也是同時指派 a[0],a[1]..a[4] 的初值爲 0。

串列的宣告與預設初值

變數可以宣告的同時給予初值，串列也是。例如：

```
a=[3,2,7,6,8]
print(a)
```

即可宣告串列也給予初值。

串列值的存取

串列的每個元素都可使用索引存取其值。例如：

```
a=[0,1,2]
print(a[0]) #0 輸出串列索引0的值
a[0]=3  #修改串列索引0的值
b=a[0]  #提取串列索引0的值
print(b)
```

串列索引可以是負值，索引負值代表從串列末端存取。例如：

```
a=[0,1,2,3]
print(a[-1]) #3
print(a[-2]) #2
```

串列的輸出

串列的輸出採用 print() 方法，以下三種方式均可輸出串列。 方法一：使用 print(串列名稱)，請鍵入以下程式，寫出執行結果。

```
a=[1,2,3]
print(a)
```

方法二：使用索引值。請鍵入以下程式，寫出執行結果。

```
a=[1]*5
for i in range(len(a)):  #len()是內儲函式，可得到串列a的長度，
                           請看7-2節。
    print(a[i],end=" ")
```

方法三：針對 Iterable(可疊代的) 物件（如串列、tuple、dict、set 等，詳見 6-6 ～ 6~8 節），Python 的 for-loop 提取每一元素的語法如下：

```
a=[1,2,3]
for a1 in a:
    print(a1)
```

以上 a1 可以是一個任意合法的變數名稱，例如 a1,aa,item 等。請鍵入以下程式，寫出執行結果。

```
a=[i for i in range(5)]
for a1 in a:  #對於每一個a1 in a
    print(a1,end=' ')  #輸出a1
```

串列的複製

串列的複製有兩種方式。第一種是取別名，兩者共用位址，也就是兩者互相連通，只要一個改變，另外一個也跟著改變。例如：

```
a=[3,2,7,6,8]
print(a)
b=a #串列的複製,共用位址
print(b)
a[4]=0
```

```
print(b)#[3,2,7,6,0]
b[0]=0
print(a)#[0,2,7,6,0]
```

第二種使用 b=a[:] 複製串列，此一複製彼此獨立，互不干擾。例如：

```
a=[3,2,7,6,8]
print(a)
b=a[:] #彼此獨立
print(b)
a[4]=0
print(b)#[3,2,7,6,8]
b[0]=0
print(b)#[0,2,7,6,8]
print(a)#[3,2,7,6,0]
```

子串列

從一個串列提取部分元素到另一串列，稱爲提取子串列。提取子字串可使用範圍運算子 (:)，格式爲「(開始索引 : 結束索引)」，但不含結束索引。若開始索引省略，則表示從頭提取；若結束索引省略，則表示提取到串列末端；索引同樣也是可以使用負值，請看以下程式。

```
a=[0,1,2,3,4]
b=a[1:3]  #從索引1開始到索引3，不含索引3
print(b) #[1,2]
c=a[1:]  #從索引1開始到結束
print(c) #[1,2,3,4]
d=a[:2]  #從索引0開始到索引2，不含索引2
print(d) #[0,1]
e=a[1:-2] #從索引1開始到索引-2，不含索引-2
print(e) #[1,2]
f=a[:-2] #從索引0開始到索引-2，不含索引-2
print(f) #[0,1,2]
g=a[-3:] #從索引-3開始到結束
print(g) #[2,3,4]
```

範例6-2a

設有資料如下

77, 66, 99, 44, 55

請寫一個程式，可以計算平均、最高分及最低分。

執行結果

```
Sum=341
Avg=68
Max=99
Min=44
```

運算思維

1. 此題我們在第四章已經介紹，我們使用 5 個單一變數儲存以上資料，但是程式有點冗長。本例爲了能使用迴圈來簡化程式，資料結構採用一維串列儲存以上資料。

   a = [7 7 , 6 6 , 9 9 , 4 4 , 5 5]

2. 有了串列，就有串列索引，有了串列索引，就可使用迴圈存取資料，參考程式如下：

```
a=[77, 66, 99, 44, 55]  #以串列儲存資料
#先指派第0筆資料爲和、極大與極小值
sum=a[0] #和
max=a[0] #極大值
min=a[0] #極小值
#使用迴圈逐一從第1筆資料開始處理每一筆資料
for i in range(1,5):
      sum=sum+a[i]  #累加每一筆資料
      if a[i]> max:  #若該筆資料大於極大值
          max=a[i]  #極大值以該筆資料取代
      if a[i]<min: #若該筆資料小於極小值
          min=a[i]  #極小值以該筆資料取代
print("Sum=%d" % sum)  #輸出和
print("Avg=%d" % (sum/5))  #輸出平均， (sum/5)括號不可省
print("Max=%d" % max) #輸出極大值
prnt("Min=%d" % min) #輸出極小值
```

自我練習

1. 設有資料如下：

   -1, 0, 3, 4, -5, 8

   (1) 統計正數、0、負數的個數。

   (2) 請問您如何儲存以上資料？

2. 假設有資料如下：

   3,6,2,1,2,3,5,6,1

   (1) 計算並輸出共有多少個 3。(2) 計算 1~6 各幾個。(3) 哪一個數字出現最多次

3. 假設有資料如下：

   11,22,33,44,55,66,77,88,99,12,23,25,31,32,33,42,72,82,99

   統計與輸出 0~9,10~19,20~29,30~39,40~49,50~59,60~69,70~79,80~89,90~100 等區間的人數。

範例6-2b

請寫一程式，可讀入 5 位學生的國文成績，並可由使用者輸入座號來查詢成績。

執行結果

```
1
input score:11
2
input score:22
3
input score:33
4
input score:44
5
input score:55
輸入完畢，可開始輸入座號查詢,輸入0結束
input number:2
座號：2 成績:22
```

運算思維

1. 本例資料使用陣列儲存資料，這樣才能使用迴圈，簡化程式。
2. 使用 a=[0]*6 宣告 a 陣列，則可使用 a[0]..a[5]，但索引 0 可以不用。
3. 輸入 1~5 號成績。
4. 進入查詢，且要能重複查詢，所以使用不定數量 while 迴圈。

程式列印

```
n=5+1  #宣告學生人數 ，本例索引0不用，所以加1
a=[0]*n  #宣告a[0]..a[5]，但本例索引0不用
#使用迴圈，逐一輸入1號到5號成績
for i in range(1,n):
    print(i,end=' ')
    b=int(input("input score:"))
    a[i]=b
print("輸入完畢，可開始輸入座號查詢,輸入0結束",end='')
#進入查詢，且要能重複查詢，且爲不定數量，所以使用不定數量while迴圈
c=int(input("input number:")) #記得轉爲int
while c!=0:  #只要c!=0 重複迴圈
    print("座號：%d 成績:%d" %(c,a[c]),end='')
    #繼續輸入座號
    c=int(input("input number:")) #記得轉爲int
```

自我練習

1. 假設班上共有 12 人，請寫一程式，可以由使用者自行輸入座號，電腦並隨時輸出共有哪些人未到。（提示：可用串列的值表示到與未到，例如串列值 1 表到，0 表未到）

範例6-2c

英文打字練習。假設有單字如下：

look、at、one、two、tree、and、day、book、from、go

請寫一個程式，可以出現 10 題英文單字，讓使用者可鍵入此單字，並評判正確與錯誤題數和使用時間。

執行結果 (以下僅摘取前面兩題)

```
look :

look
at :

at
```

程式列印

1. 將單字用串列儲存如下：

```
import time #使用time.time()
a=['look','at','one','two','tree','and','day','book','from','go']
n=10
right=0
wrong=0
t1=int(time.time())#取得系統時間的秒數
#使用迴圈逐一輸出每一題單字
for i in range(n):
    print(a[i],':')#每次取一個輸出
    b=input()#由使用者鍵入此單字
    if b==a[i]:#比較是否相同
        right=right+1
    else:
        wrong=wrong+1
t2=int(time.time())
t=t2-t1
print('right=%d'% right)
print('wrong=%d'% wrong)
print('total time=%d second'% t)
```

自我練習

1. 同範例 6-2c，但使用者答題後，馬上可回應正確或錯誤，若是錯誤也要回應正確的單字。
2. 同範例 6-2c，但增加爲 100 個單字，每次測驗依亂數出現 10 個單字，且不能重複。

   提示：題目取亂數，直覺的方法如下：

```
import random as r
a=['look','at','one','two','tree','and','day','book','from','go']
print(a[r.randint(1, 9)])
```

但是，因爲亂數會重複，以上方法會挑到重複的題目，所以應該修改如下，即可將題目先依照亂數排列。

```
import random as r
a=['look','at','one','two','tree','and','day','book','from','go']
for i in range(10):
    c=r.randint(0,9)
    a[i],a[c]=a[c],a[i]#每一次與取到的亂數交換
```

3. 同範例 6-2c，但增加爲 100 個單字，每次測驗 3 分鐘，題目依照亂數出現，統計使用者可答對題數。亂數的取法請複習上冊 3-4 節。
4. 同範例 6-2c，但增加爲 100 個單字，題目依照亂數出現，使用者答錯了才停止，統計使用者可連續答對題數。亂數的取法請複習上冊 3-4 節。

查表

寫程式有時候會用到查表問題，例如，十進位轉爲十六進位時，當除以 16 時，其餘數的對應值如表 6-1：

表 6-1　十六進位符號對照表

餘數	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
對應值	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

此時即可使用陣列。

範例6-2d

請寫一程式，可將任意十進位數轉爲 N 進位數。（2<=N<=16）

運算思維

1. 將 16 進位的 16 個符號以串列儲存如下：

```
c=["0","1","2","3","4","5","6","7","8","9","A","B","C","D","E","F"]
```

2. 只要待轉換數字大於 0，將待轉換數字除以欲轉換的基底，每得到 1 個餘數，就查表。例如，餘數是 10，那就得到『A』。
3. 向左串接餘數，即爲所求。例如：待轉換數字若爲 18，則運算過程如下：

```
d= ''   (先指派一個變數作爲轉換後的十六進位數)
18/16=1..2 (商是1，餘數p='2')
d=p+d   (d= '2'，商大於0，程式繼續)
1/16=0..1 (商是0，餘數p='1')
d=p+d   (d='12'，商沒有大於0，程式結束)
```

程式完成，18 的十六進位是 '12'。

程式列印

```
#將十六進位的16個符號以串列儲存
c=["0","1","2","3","4","5","6","7","8","9","A","B","C","D","E","F"]
a=18 #待轉換數字
b=16  #轉換基底
d="" #轉換結果
#因爲數字長度不定，所以使用不定while迴圈，逐一求餘數
while (a>0):   #只要待轉換數字a還大於0，則重複迴圈
    r=a%b;  #以基底爲除數，求餘數
    p=c[r]  #將餘數查表轉爲字元
    d=p+d  #向前串接所求得字元
    a=a//b  #除以基底
print(d)  #輸出結果 12
```

補充說明

1. 先產生的餘數是放在右邊，所以本例是 d=p+d 向左串接，請自行修改爲 d=d+p，並觀察執行結果。

自我練習

1. 假設有一個大富翁遊戲，共有 10 個位置，每個位置的獎罰如表 6-2，遊戲一開始假設有 10000 元，請寫程式讓使用者玩，每次擲一顆骰子，決定其行走距離、落腳位置、增減金額，且記錄每次剩餘金額。

表 6-2　大富翁遊戲賞罰表

0	1	2	3	4	5	6	7	8	9
中發票得200元	闖紅燈罰200元	中樂透得10000元	任意停車，罰1000元	繳汽車稅2000元	繳房屋稅3000元	停玩一次	得到股息1000元	抽獎得到3000元	繳停車費30元

2. 以串列重做上冊範例 4-5d 的電子琴。

3. 假設有兩筆資料如下：

```
3,4,7,9,12
1,5,6,8,15,
```

請先以兩個串列儲存，再合併成一個串列。

4. 假設有兩筆資料如下：(每筆資料都已經是由小而大排列)

```
3,4,7,9,12
1,5,6,8,15,16,17,18
```

請以兩個串列儲存以上資料，再幫忙合併，且合併後還是由小而大。

※ 5. 假如有兩筆資料如下：

```
3,1,4,5,6,7
5,1,7,4,3
```

(1) 請問如何儲存以上資料？
(2) 請比較與分析有哪些資料重複。
(3) 將重複的資料放到第三個串列。(交集)
(4) 將兩筆資料合併，放到第四個串列，但資料不可重複。(聯集)

※ 6. 假如有資料如下

3,1,4,5,6,7,1,1,3

請將重複的資料去掉。

6-3 二維串列

前面範例 6-2c 的單字

```
look、at、one、two、tree、and、day、book、from、go
```

此稱爲一維資料，我們使用一維串列儲存如下：

```
a=['look','at','one','two','tree','and','day','book','from','go']
```

這樣可以用迴圈存取每個單字，現在若有英文單字的詞性與中文資料如下：希望能出現中文，讓使用者輸入英文，並檢查正確與否？

編號	英文	詞性	中文
1	delighted	adj	高興的
2	gear	n	服裝
3	behave	v	表現

以上每一個單字都還有詞性與中文註解，此稱爲二維資料，此時就需要使用二維串列儲存以上二維資料。本例可宣告二維串列如下：

```
a=[["delighted", "adj", "高興的"],
   ["gear", "n", "服裝"],
   ["behave", "v", "表現"]]
```

則以上單字、詞性、中文的索引編號（以下簡稱索引）如下：

英文	詞性	中文
delighted (0,0)	adj (0,1)	高興的 (0,2)
gear (1,0)	n (1,1)	服裝 (1,2)
behave (2,0)	v (2,1)	表現 (2,2)

因爲是二維資料，此時就需要二維索引 (列索引 , 行索引)，例如，a[0][2] 的 0 稱爲列索引，2 稱爲行索引。有了索引，往後就可以使用變數名稱與索引存取以上資料。例如：

```
print(a[0][2]) #_____________
print(a[2][1]) #_____________
```

以上即爲本節所要介紹的二維串列。

二維串列的宣告

二維串列的宣告常用語法如下：

```
串列名稱=[[初值 for i in range (行數)]for j in range (列數)]
```

例如：以下程式可快速初始化一個三行四列的二維串列，其初值爲 0。

```
a=[[0 for i in range (3)]for j in range (4)]
```

以上程式亦可簡化如下：

```
b=[[0]*3 for i in range(4)]
print(a)  #_____________
print(b)  #_____________
```

二維串列初始化

單一變數、一維串列都可宣告變數同時給予初值，二維串列也可以，以下程式可初始化一個二維串列，且預設初值。

```
c=[[1,2,3],[4,5,6]]
print(c)
```

以上二維串列的索引如表 6-3。

表 6-3　二維串列資料與索引對照表

c[0][0]=1	c[0][1]=2	c[0][2]=3
c[1][0]=4	c[1][1]=5	c[1][2]=6

請鍵入以下程式，寫出執行結果。

```
c=[[1,2,3],[4,5,6]]
print(c)
c[0][1]=12
d=c[0][1]
print(d)
print(c)
```

範例6-3a

假設有單字的詞性與中文資料如下：請寫一個程式，此程式會連續 3 次出現中文與詞性，讓使用者輸入其英文，並統計答對與答錯題數及答題花費時間。

編號	英文	詞性	中文
1	delighted	adj	高興的
2	gear	n	服裝
3	behave	v	表現

輸出結果

```
高興的 adj :
delighted
服裝 n :
gear
表現 v :
beha
right=2
wrong=1
total time=23 second
```

運算思維

1. 將以上二維資料以二維串列儲存，本例使用 a 串列。
2. 使用串列儲存資料，就有索引，有索引就可使用迴圈操作以上資料。

程式列印

```
import time #使用time()函式
a=[["delighted", "adj", "高興的"],
   ["gear", "n", "服裝"],
   ["behave","v","表現"]]
n=3  #題數
right=0 #答對題數
wrong=0 #答錯題數
t1=int(time.time())#取得系統時間的秒數
#使用for迴圈逐一輸出每一題的中文與詞性
for i in range(n):
    print(a[i][2],a[i][1],':',end='') #輸出每一題的中文、詞性
    b=input()#等待使用者輸入英文
    if b==a[i][0]: #將使用者輸入與原題目比對
        right=right+1
    else:
        wrong=wrong+1
t2=int(time.time()) #取得系統時間的秒數
t=t2-t1
print('right=%d'% right)
print('wrong=%d'% wrong)
print('total time=%d second'% t)
```

補充說明

1. t2=int(time.time()) # 可取得系統時間，單位是秒數，使用前需要先載入 time 模組，請看 7-2 節。待程式結束時，再取系統時間，兩者相減，即可計算使用者使用的秒數。

自我練習

1. 同範例 6-3a，但增加當使用者答題後，馬上告知正確與否；若是錯誤，也告知正確答案。
2. 同範例 6-3a，但增加爲 100 個單字，每次測驗依亂數出現 10 個單字，同次測驗中所出現單字不能重複，統計使用者答對題數。亂數的取法請複習上冊 3-4 節。
3. 同範例 6-3a，但增加爲 100 個單字，每次測驗 3 分鐘，題目依照亂數出現，同次測驗中所出現單字不能重複，統計使用者答對題數。亂數的取法請複習上冊 3-4 節。

範例6-3b

選擇題測驗。假設英文單字的選擇題選項與答案資料如下：

編號	單字	類別	答案	選項 1	選項 2	選項 3	選項 4
1	delighted	adj	1	高興的	悲傷的	生氣的	憤怒的
2	gear	n	3	背包	褲子	服裝	鞋子
3	behave	v	2	生氣	表現	難過	看到

請寫一程式，可以讓使用者使用選擇題測驗以上 3 個單字，使用者每回答 1 題，電腦馬上回應正確或錯誤，結束測驗並統計答對與答錯題數及答題所花費時間。

執行結果

```
1 . delighted,adj :(1) 高興的 :(2)  悲傷的 :(3) 生氣的 :(4) 憤怒的 :
1
Right
2 . gear,n :(1) 背包 :(2) 褲子 :(3) 服裝 :(4) 鞋子 :
3
Right
3 . behave,v :(1) 生氣 :(2) 表現 :(3) 難過 :(4) 看到 :
1
Wrong
The Right number is :2
The wrong number is :1
total time=12 second
```

運算思維

1. 以上英文測驗題目是二維資料，所以使用二維串列儲存，如以下 a 串列，每筆資料第三個 (索引 2) 是答案 ，如以下第 1 題答案是 2。
2. 有了串列，就有索引編號，即可使用迴圈操作資料，程式如下：

程式列印

```
import time
a=[["book","n","2","學校","書","老師","校長"],
   ["delighted", "adj", "1","高興的", " 悲傷的", "生氣的", "憤怒的"],
        ["gear", "n","3" , "背包", "褲子","服裝", "鞋子"]];
n=3#題數
```

```
c=0
d=0
t1=int(time.time())
for i in range(0,n):
    print("%d . %s,%s :" % (i+1,a[i][0],a[i][1]),end='')
                                                    #輸出題號,英文單字,詞性
    for j in range(1,4+1):
        print("(%d) %s :" % (j,a[i][j+2]),end='')#輸出四個選項
    b=input()#等待使用者輸入答案
    if (b==a[i][2]): #比對
        c=c+1
        print("Right")
    else:
        print("Wrong")
        d=d+1
t2=int(time.time())
t=t2-t1
print("The Right number is :%d"%c)
print("The wrong number is :%d"%d)
print('total time=%d second'% t)
```

自我練習

1. 同範例 6-3b，但增加當使用者答題後，馬上告知正確與否；若是錯誤，也告知正確答案。
2. 同範例 6-3b，但增加爲 100 個單字，每次測驗依亂數出現 10 個單字，同次測驗中所出現題目不能重複，統計使用者答對題數。

※ 3. 同範例 6-3b，但增加爲 100 個單字，每次測驗 3 分鐘，題目依照亂數出現，同次測驗中所出現題目不能重複，還有四個選項的順序也是亂數，請留意，答案也要跟著變動，才能核對是否正確，統計使用者答對題數。

範例6-3c

前面 3-4 節的「小毛驢」程式有點冗長。請問，資料應該如何數位化，程式才會精簡呢？

執行結果 (以下僅摘一小段)

運算思維

1. 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，所以一個八分音符所占的時間是 60000ms/(2*80)，每小節是 2 拍。
2. 小毛驢所有音符的最小速度是八分音符，所以我就取八分音符的時間爲 1 個單位。
3. 簡譜的 1，是 Do，對應頻率是 523，所以，以 a 串列爲對照表。(前面索引 0，我不用，所以放 0)

```
a = [ 0 , 5 2 3 , 5 8 7 , 6 5 9 , 6 9 8 , 7 8 4 , 8 8 0 , 9 9 8 , 1 0 4 6 ]
```

4. 將音階、歌詞、延遲時間數位化。以下僅作 4 小節，「我有一隻小毛驢我從來也不騎」。

```
song = [[1,'我',1],[1,'有',1],[1 ,'一',1],[3 ,'隻',1 ],
[5 ,'小' ,1 ],[5 ,'毛' ,1 ],[5 ,'驢' ,1 ],[5 ,'我' ,1 ],
[6 , '從' ,1 ],[6 , '來' ,1 ],[6 , '也' ,1 ],[8 , '不' ,1 ],[5 , '騎' ,4 ]]
```

以上每一列是由 [音階 , 歌詞 , 延遲時間] 所構成。

5. 使用迴圈依序輸出每一個列，參考程式如下：

程式列印

```
import ctypes
p = ctypes.windll.kernel32
a = [ 0 , 523 , 587 , 659 , 698 , 784 , 880 , 998 , 1046 ]
t = 60000//160
song = [[1,'我',1],[1,'有',1],[1 ,'一',1],[3 ,'隻',1 ],
    [5 ,'小' ,1 ],[5 ,'毛' ,1 ],[5 ,'驢' ,1 ],[5 ,'我' ,1 ],
    [6 , '從' ,1 ],[6 , '來' ,1 ],[6 , '也' ,1 ],[8 , '不' ,1 ],[5 , '騎' ,4 ]]
#使用迴圈，每次輸出二維串列的一列
for i in song:
    print(i[1])#歌詞
    p.Beep(a[i[0]],i[2]*t)# 音階, 時間
```

範例6-3d

假設有兩個矩陣 A、B 如下：

$$A=\begin{pmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{pmatrix} \quad B=\begin{pmatrix}1 & 2 & 0\\ 2 & 1 & 3\end{pmatrix}$$

請求其相加的結果如下：

$$\begin{pmatrix}2 & 4 & 3\\ 6 & 6 & 9\end{pmatrix}$$

請問要使用何種資料結構較省事？

執行結果

```
[[1, 2, 3], [4, 5, 6]]
[[1, 2, 0], [2, 1, 3]]
[[2, 4, 3], [6, 6, 9]]
2  4  3
6  6  9
```

程式列印

1. 本題因爲是二維資料，若用二維串列儲存，就可以使用雙迴圈處理縱向與橫向二維資料。

```
#逐一初始化每一串列
a=[[1,2,3],[4,5,6]]
b=[[1,2,0],[2,1,3]]
c=[[0,0,0],[0,0,0]]
#使用雙迴圈，將兩個二維串列a與b，逐一將每一行、列資料相加，放到c串列
for i in range(0,2): #逐一處理每一列
    for j in range(0,3): #逐一處理每一行
        c[i][j]=a[i][j]+b[i][j]
print(a) #使用函式輸出a串列
print(b) #使用函式輸出b串列
print(c) #使用函式輸出c串列
#自行使用雙迴圈輸出c串列
for i in range(0,2):
    for j in range(0,3):
        print(c[i][j],end="  ")
    print() #跳列
```

自我練習

1. 假設有矩陣如下：

$$\begin{pmatrix} 1 & 2 & 0 \\ 0 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 0 \end{pmatrix}$$

(1) 請問以何種資料結構儲存。

(2) 請檢查幾個 0。

(3) 請統計所有元素和。

(4) 再將每一元素乘以 2 後輸出。

範例6-3e

假設有學生成績如下：

座號	國文	英文	數學	平均	不及格科數
1	50	60	70		
2	30	40	50		
3	70	80	90		
4	66	77	88		
5	22	33	44		
平均					

1. 請選用適當資料結構儲存以上資料。
2. 計算每人平均。
3. 統計每人不及格科數。
4. 統計各科平均。

執行結果

```
[1, 50, 60, 70, 60, 1]
[2, 30, 40, 50, 40, 3]
[3, 70, 80, 90, 80, 0]
[4, 66, 77, 88, 77, 0]
[5, 22, 33, 44, 33, 3]
[0, 47, 58, 68, 58, 0]
```

運算思維

1. 本例雖可使用 5 個一維串列儲存以上資料，這樣只簡化縱向的各科成績計算，但橫向每一位學生的平均計算還是無法使用迴圈，若使用二維串列，那就可以直向、縱向通通配合迴圈，使得程式非常精簡。
2. 使用二維 a 串列儲存以上二維資料

```
a=[[1,50,60,70,0,0],
   [2,30,40,50,0,0],
   [3,70,80,90,0,0],
   [4,66,77,88,0,0],
   [5,22,33,44,0,0],
   [0,0,0,0,0,0]]
```

3. 以上資料的列、行索引如下表：

座號	國文	英文	數學	平均	不及格科數
1 (0,0)	50 (0,1)	60 (0,2)	70 (0,3)	(0,4)	(0,5)
2 (1,0)	30 (1,1)	40 (1,2)	50 (1,3)	(1,4)	(1,5)
3 (2,0)	70 (2,1)	80 (2,2)	90 (2,3)	(2,4)	(2,5)
4 (3,0)	66 (3,1)	77 (3,2)	88 (3,3)	(3,4)	(3,5)
5 (4,0)	22 (4,1)	33 (4,2)	44 (4,3)	(4,4)	(4,5)
科目平均	(5,1)	(5,2)	(5,3)	(5,4)	(5,5)

4. 每個人國英數的橫向資料累加。此一累加過程爲「行」索引先變化，例如 (0,1)+(0,2)+(0,3) 先累加，然後要放到 (0,4)，所以內迴圈 j 要要從 1 到 3，且運算後放在「行」索引 4。然後，外迴圈 i 要從 0 到 4，分別計算 1 號到 5 號的成績和，變數 i 爲則爲「列」索引。所以程式如下：

```
for i in range(0,5):#每個學生
        for j in range(1,4):#累加1,2,3等3個科目
            a[i][4]=a[i][4]+a[i][j]
```

5. 每個科目國、英、數的縱向資料累加。此一累加過程爲「列」索引先變化，例如，(0,1)+(1,1)+(2,1)+(3,1)+(4,1) 先累加，然後放在 (5,1)，所以，內迴圈 j 要從 0 到 4，且放在「列」索引 5。然後，外迴圈 i 要從 1 到 4，分別計算『國、英、數、平均』等 4 個科目的和，此變數 i 則爲「行」索引，所以程式如下：

```
    for i in range(1,5):#每個科目
        for j in range(0,5):#累加每個學生的成績
            a[5][i]=a[5][i]+a[j][i]
```

程式列印

```
#使用二維串列儲存二維資料
a=[[1,50,60,70,0,0],
   [2,30,40,50,0,0],
   [3,70,80,90,0,0],
   [4,66,77,88,0,0],
   [5,22,33,44,0,0],
   [0,0,0,0,0,0]]
#逐一計算每個人總分、不及格科數
for i in range(0,5):
    for j in range(1,4):
        a[i][4]=a[i][4]+a[i][j] #計算每個人總分
        #計算不及格科數
        if a[i][j]<60:
            a[i][5]=a[i][5]+1 #計算不及格科數
    a[i][4]=a[i][4]//3 #計算每個人平均
#逐一計算每一科目總分
for i in range(1,5):
    for j in range(0,5):
        a[5][i]=a[5][i]+a[j][i] #累加每一個人的分數
    a[5][i]=a[5][i]//5 #計算每一科目的平均
#輸出每一個人成績
for i in range(0,6):
    print(a[i])
```

自我練習

1. 假設某一公司有三個業務員，其每季的業績如下：

編號	第一季	第二季	第三季	第四季	和	零的個數
1	0	2	4	6		
2	8	0	2	1		
3	7	0	0	9		
和						
零的個數						

(1) 計算每一個人的業績總和及零的個數。
(2) 計算每一季的業績總和及零的個數。
(3) 請問資料結構爲何？

6-4 串列函式與串列生成式

串列函式

前面 6-2、6-3 節是由程式開發者自行使用迴圈操作串列，Python 的特色是增加一些處理串列的函式，這樣就可以減輕程式設計師的工作。以下是一些常用的串列函式：

▶ sum()

前面範例 6-2a 已經介紹如何使用迴圈計算串列的和。Python 則提供 sum() 函式計算串列和。例如，

```
a=[3,2,1]
print(sum(a))#6
```

也可指定範圍，也就是可跳過非數值的資料。例如：

```
a=["aa",3,2,1]
print(sum(a[1:4]))#6 [1:4]表示取索引1~4，但不含索引4
```

二維陣列也可以，例如：

```
a=[[1,50,60,70,0,0],
    [2,30,40,50,0,0],
```

```
    [3,70,80,90,0,0],
    [4,66,77,88,0,0],
    [5,22,33,44,0,0],
    [0,0,0,0,0,0]]
a[0][4]=sum(a[0][1:4])
print(a[0]) #1,50,60,70,180,0
```

▶ max()、min()

求串列極大與極小，與 sum() 用法相同，請自行練習。

▶ append()

在串列末端加入元素。例如：

```
a=[1,2,3]
a.append(4)
print(a)#1,2,3,4
```

▶ insert()

在指定索引加入資料，原索引資料則後退。例如：

```
a=[0,1,2,3]
a.insert(2,4)
print(a)#0,1,4,2,3
```

又例如，以下程式可將資料加在最前面，這樣就如同佇列的操作。

```
a=[1,2,3]
a.insert(0,0)
print(a)#0,1,2,3
```

▶ pop()

取出指定索引元素，且刪除此元素。例如：

```
a=[0,1,2,3]
b=a.pop(1)
print(b)#1
print(a)#[0,2,3]
```

若沒有指派索引值，那就是刪除串列末端元素，所以可以拿來當作佇列用。

```
a=[0,1,2,3]
b=a.pop()
print(b)#3
print(a)#[0,1,2]
```

▶ remove()

取出指定元素，且刪除此元素。例如；

```
a=[1,2,3]
a.remove(2)
print(a)#[1,3]
```

▶ sort()

排序串列，例如：

```
a=[1,8,3]
a.sort()#由小而大
print(a)#1,3,8
a.sort(reverse=True)#由大到小
print(a)#8,3,1
```

二維串列也可以使用 sort() 排序，但要使用 lambda 指派欄位。例如：

```
a=[[1,50,60,70,0,0],
   [2,30,40,50,0,0],
   [3,70,80,90,0,0],
   [4,66,77,88,0,0],
   [5,22,33,44,0,0],]
a.sort(key=lambda x:x[1])#欄位1
for i in range(5):
    print(a[i])
```

```
[5, 22, 33, 44, 0, 0]
[2, 30, 40, 50, 0, 0]
[1, 50, 60, 70, 0, 0]
[4, 66, 77, 88, 0, 0]
[3, 70, 80, 90, 0, 0]
```

▶ sorted()

前面 sort 是將串列排序，sorted() 則將排序結果複製到另一串列，且原串列不變。例如：

```
a=[1,8,3]
b=sorted(a)#由小而大
print(b)#1,3,8
b=sorted(a,reverse=True)#由大到小
print(b)#8,3,1
print(a)#1,8,3
```

▶ extend()

合併兩個串列。例如：

```
a=[1,2,3]
b=[4,5]
a.extend(b)
print(a)#1,2,3,4,5
```

▶ in

判斷元素是否在物件裡面。物件可以是 List、Tuple、Dict 或 Set。例如：

```
a=[1,2,3]
print( 1 in a)#True
```

▶ not in

判斷元素是否『不在』物件裡面，用法同 in。

▶ len()

傳回串列長度。例如：

```
a=[1,2,3]
print( len(a))#3
```

▶ cout()

傳回指定元素出現次數。例如：

```
a=[1,2,3,2]
print(a.count(2))#2
```

▶ index()

傳回指定元素第一次出現的索引。例如：

```
a=[2,3,1,3]
print(a.index(3))#1
```

串列生成式

前面 6-2、6-3 節是由程式開發者自行使用迴圈操作串列，串列生成式 (List Generator) 也是 Python 很神的方法，其方式是可以在串列的括號內使用運算式與迭代物件產生新的串列。其語法如下：

```
new串列=[運算式 for 元素 in 迭代物件]
```

例如，以下程式可以將整個串列元素都乘以 2，再放入另一串列。

```
a=[1,2,3]
b=[a1*2 for a1 in a]#以上兩個a1，變數名稱要相同，且遵守變數命名規則
print(b)#2 4 6
```

以下程式可快速初始化串列：

```
c=[0 for i in range(5)]
print (c)#0,0,0,0,0
d=[i for i in range(5)]
print (d)#0,1,2,3,4
```

又例如，以下串列 a 的元素都是字串，那要如何通通轉爲數值呢？程式如下：

```
a=['1','2','3']
print(a[0]+a[1])#12
b=[int(a1) for a1 in a]
print(b[0]+b[1])#3
```

條件生成

Python 眞的非常富有想像力，串列生成式還可加條件。例如，以下可將及格的分數放到 b 串列。

```
a=[22,66,33,88]
b=[a1 for a1 in a if a1>=60]
print(b)#66 88
```

又例如，以下程式可直接找出指定數字 12 的因數。

```
a=[i for i in range(1,12+1) if 12 % i==0]
print(a)
```

又例如，以下程式就可直接完成三位數阿姆斯壯數。

```
a=[[100*i+10*j+k]for i in range(1,9+1) for j in range (0,9+1)
for k in range(0,9+1) if (100*i+10*j+k)== (i**3+j**3+k**3)]
print (a)
```

又例如，以下程式可將二維陣列每一個元素加 1。

```
a=[[1,1,1,0,1],[0,1,0,1,1],[0,0,0,0,0],[0,0,0,1,0]]
b=[[ a11+1 for a11 in a1] for a1 in a]
print(b)
```

自我練習

1. 請鍵入以下程式，並觀察執行結果。

```
a = [[i for i in range(3)] for j in range(4)]
print(a)
print(a[0][0])
b=[sum(a1) for a1 in a ]
print (b)
```

2. 假設有資料如下，單位是公斤

   2,5,6,8,10

   請使用串列儲存，且將它轉爲台斤數且放到另一串列。

3. 請寫一程式，可以初始化一個 100 個元素的一維串列，每一個元素是 1 到 6 的亂數，其次，將偶數放到另一串列。
4. 請寫一程式，將二維陣列初始化如下：

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

範例6-4a

編碼（106/10 APCS 試題）

1. 輸入。
   第一列爲取碼長度 m。例如：

```
1
```

   第二列輸入若干數字，以 0 結束。例如：

```
252 373 28 0
```

2. 處理
   將第二列的每一項數字從左邊取長度 m 的數字字元，然後比較這些取出的數字大小，取極大值輸出。
   將第二列的每一項數字從右邊取長度 m 的數字字元，然後比較這些取出的數字大小，取極小值輸出。
3. 範例

資料編號	輸入	輸出
1	1　（取碼長度） 252 373 28 0	2 3 2 max=3（取左邊 1 個） 2 3 8 min=2（取右邊 1 個） 3 2
2	2　（取碼長度） 258 0	25 max=25（取左邊 1 個） 58 min=58（取右邊 1 個） 25 58

3	2 （取碼長度） 123 58 612 0	12 58 61 max=61 取左邊 2 個） 23 58 12 min=12（取右邊 2 個） 61 12

執行結果

```
['252', '373', '28']
['2', '3', '2']
3
['2', '3', '8']
[2, 3, 8]
2
```

程式列印

```
m=1 #取碼長度，測試資料1
a=[252,373,28] #先將資料以串列儲存
#m=2 #取碼長度，測試資料2
#a=[258]
#m=2 #取碼長度，測試資料3
#a=[123,58,612]
b=[str(a1) for a1 in a] #逐一將串列每一元素轉字串
print(b) #輸出資料，確認是否正確
c=[b1[0:m] for b1 in b]  #串列每一元素取左邊m個位數
print(c) #['2', '3', '2']
d=[int(c1) for c1 in c] #將取到的資料轉回數值
print(max(d))  # 3，依題目要求，取資料的極大值，再輸出
#取右邊指定m位數，然後取極小值
#print(min([int(b1[-m:]) for b1 in b]))  #用這一行就代表下面全部
e=[b1[-m:] for b1 in b] #取右邊指定位數
print(e)
f=[int(e1) for e1 in e] #轉回數值
print(f)
print(min(f)) #取極小值
```

自我練習

1. 以上資料的輸入，我們先使用變數a指派測試資料，如程式a=[252,373,28]，請同學依題目說明，自行改爲使用「input」指令輸入測試資料。

Chapter
07

函式應用

7-1 模組（Module）

工業化的產品有所謂模組化設計，例如，一部電腦分成主機、螢幕、硬碟、鍵盤、滑鼠等模組，產品的設計與維修都是模組化，彼此獨立，也就是鍵盤壞了，就換鍵盤，滑鼠壞了就換滑鼠。軟體工業是發展較晚的新興產業，也是遵循此道理，希望將問題模組化，每個模組也和硬體系統一樣，是可組合、可分解和可更換的單元。模組化程式設計是指解決一個複雜問題時由上而下逐層把軟體系統劃分成若干模組。每個模組完成一個特定的子功能，所有的模組的功能與介面要先定義，並可按某種方法組裝起來，完成整個系統所要求的功能。例如，要撰寫學校學生成績管理系統，就可先分成三個模組，分別是教務處的考試成績模組、學務處的操行系統模組、輔導處的輔導評量系統模組，這些子系統都可完成特定功能，且資料交換介面也已經定義且公開。教務處的考試成績系統模組則再分爲小考成績登錄模組、月考成績登錄模組、期末考成績登錄模組等系統，如圖 7-1 所示。

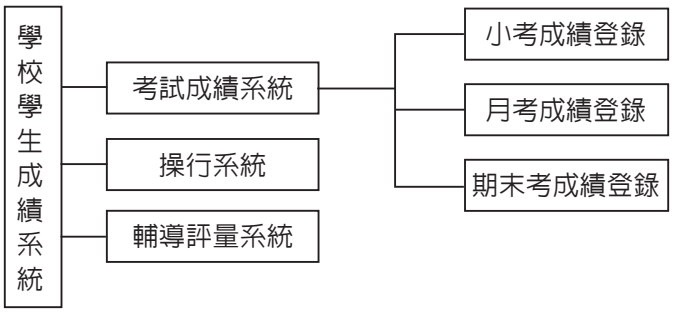


圖 7-1　學校學生成績系統示意圖

7-2 內建函式庫的認識與應用

Python 內建函式庫分爲內建函式與模組，分別說明如下：

內建函式（Built-in Function）

Python 雖然是以物件導向語言爲基礎所發展的語言，所有函式庫通通以模組分類存放，但仍保留使用者舊有習慣，將最常用的功能以函式存在，Python 內建公用函式如圖 7-2。

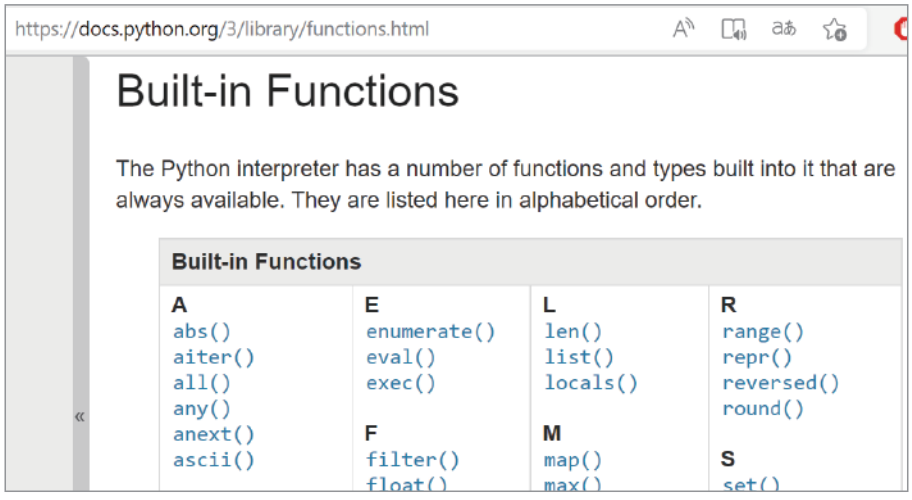


圖 7-2 Python 內建函式圖

以上常用函數的用法，請參考 3-1 節。

模組

物件導向則是依照函式功能分類，寫在一個資料夾，就稱爲一個模組。下圖則是 Python 的內建函式庫 (點選『開始 /Python/Python Module Docs』)，所有方法已經依照模組分類存放，如圖 7-3 所示：

圖 7-3　Python 內建模組

以上模組相當豐富與完整，以下我們摘錄一些高職生常用模組。熟悉這些模組以後，自然可自行探索其餘模組。

▶ math

開啓 math 模組，如圖 7-4 所示：

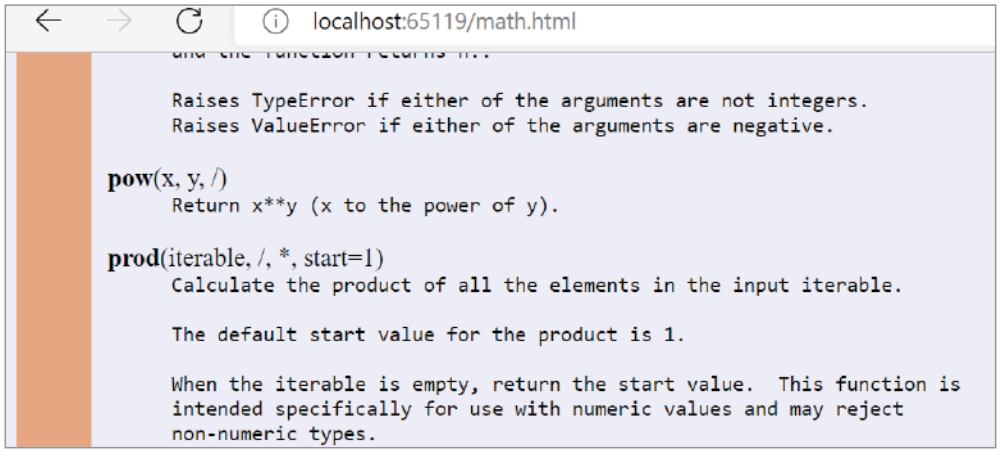


圖 7-4　math 模組可用方法

裡面有很多數值運算的方法。使用前應先載入此模組，例如：

```
import math
print(math.fabs(-3.14))#3.14
print(math.pow(2.1,3.1))#9.974
```

有時候模組名稱很長，此時也可將模組取一個別名，然後使用「別名 . 方法」撰寫程式。例如：

```
import math as m
print(m.fabs(-3.14))#3.14
print(m.pow(2.1,3.1))#9.974
```

▶ random

random 模組有一些隨機亂數的方法，例如，之前我們已經介紹產生亂數的方法 randint()，程式如下：

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

以下再介紹一些 random 模組常用方法。

▶ choice(串列)

於串列中，任選一個資料傳回。例如，先將獎品放在串列中，以下程式就可幫我們隨機抽選一個獎品。

```
import random
a=['電視','冰箱','手機','銘謝惠顧']
b=random.choice(a)
print(b)
```

▶ sample()

剛剛 choice() 是選一個，但是樂透開獎卻是選指定個數的球，sample() 就可以傳回指定個數的球數。例如，以下程式可從 1 ～ 42 號球任意取出 7 球，我們假設最後一個球就是特別號。

```
import random
a=random.sample(range(1,43),7)
print(a)
b=a.pop()
print('special number is %d'%b)
```

或是於 42 個人抽出 7 人，程式也是同上。

▶ time

time 模組內有一些關於時間的方法，以下篩選常用方法如下：

▶ localtime()

以 tuple 的方式傳回時間的個別資料。例如：

```
import time #載入時間模組
a=time.localtime()
print(a[0])#年
print(a[1]) #月
print(a[2]) #日
print(a[3]) #時
print(a[4]) #分
print(a[5]) #秒
print(a[6]) #星期幾
print(a[7]) #該年第幾天
print(a[8]) #是否夏令時間
print(time.asctime())#傳回現在時間Wed Nov 6 15:57:20 2022
print(time.time())#傳回1970/1/1起的秒數
```

▶ sleep(n)

強制讓電腦暫停 n 秒。例如，前面我們已經使用 sleep(1) 讓電腦暫停 1 秒。這樣就可簡單完成時鐘程式。

```
import time
t=0
while (1):
    t=(t+1)%(24*60*60)
    h=t//3600
    m=(t-h*3600)//60
    s=t %60
    print('%2d:%2d:%2d'% (h,m,s))
    time.sleep(1)#暫停1秒
```

twstock 模組

因爲 Python 是免費開源軟體（英語：open source software，縮寫：OSS）又稱開放原始碼軟體，也就是原始碼可以任意取用的電腦軟體，這

種軟體的著作權持有人允許使用者學習、修改且散播自己創作的軟體。所以熱心的人士與協力廠商就源源不絕開發能相容 Python 的實用模組。以股市分析程式爲例（可能是股票買賣公司，爲了鼓勵大家投資股市，就幫忙寫一些股票看盤套件），例如，以下程式，就可輸出近三十日『大立光』收盤價。

```
import twstock as a
b=a.Stock("3008")#3008大立光股票代碼
print(b.price)#近三十日收盤價
```

補充說明

1. 本書推薦與介紹使用 Anaconda 作爲 Python 整合開發環境，主要是 Anaconda 已經蒐集使用者常用模組且安裝。例如：以上 twstock 模組，雖然是第三方模組，但因爲熱門且常用，所以安裝完 Anaconda 就可以看到 twstock 模組，如圖 7-5：

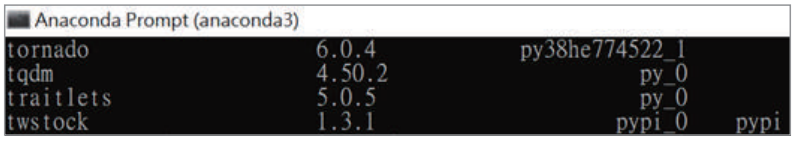


圖 7-5　Anaconda 所安裝模組

2. 開啓「Anaconda Prompt」(點選功能 Window/ 開始 /Anaconda/ Anaconda Prompt)，並鍵入「conda list」，如圖 7-6，再往下捲動，即可見圖 7-5。

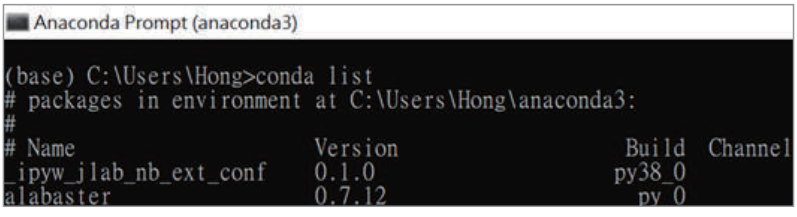


圖 7-6　Anaconda Prompt 視窗

3. 若所需模組未出現，請於圖 7-6Anaconda Prompt 視窗，鍵入「conda install 模組名稱」即可安裝該模組。

numpy模組

numpy 模組存在很多解國高中數學的方法，這些方法可以幫助理解或解決國中、高中的數學問題。

▶ roots()方法

解一元多次方程式是使用 roots() 方法，可解一元一次、一元二次、一元三次等方程式。例如，若有一元一次方程式如下：

```
x+2=0
```

則求其解的程式如下：

```
import numpy as np
b=np.roots([1,2])#x+2=0
print(b)#-2
```

又例如，一元二次方程式如下：

```
x²-4x-12=0
```

求其解的程式如下：

```
import numpy as np
b=np.roots([1,-4,-12])#x**2-4x-12=0
print(b)#6,-2
```

又例如，一元三次方程式如下：（一元三次方程式是一元二次方程式的推廣，高職商管數學無介紹，僅供參考）

```
x³-3x²-16x-12=0
```

求其解的程式如下：

```
import numpy as np
b=np.roots([1,-3,-16,-12])#x**3-3x**2-16x-12=0
print(b)#6,-2,-1
```

▶ linalg.solve()方法

二元一次或三元一次要使用 linalg 模組的 solve 方法。例如，若有二元一次方程式如下：

```
3x+y=5
x-2y=-3
```

則求其解的 Python 程式如下：

```
import numpy as np
c=np.array([[3,1],[1,-2]])
d=np.array([5,-3])
ans=np.linalg.solve(c,d)
print(ans)#1,2
```

又例如，三元一次方程式如下：（三元一次方程式是二元一次方程式的推廣，高職商管數學無介紹，僅供參考）

```
x+y-z=-2
x+z=2
x-y+2z=5
```

則求其解的 Python 程式如下：

```
import numpy as np
c=np.array([[1,1,-1],[1,0,1],[1,-1,2]])
d=np.array([-2,2,5])
ans=np.linalg.solve(c,d)
print(ans)#1,-2,1
```

matplotlib.pyplot模組

國高中數學有很多數學函數，本節就使用 matplotlib.pyplot 繪圖模組來繪製這些函數圖形，尤其是高一數學的指數、對數、三角函數，您只要多畫幾次，就如同圖像式學習，那就會更瞭解這些數學函數的內涵。其次，matplotlib.pyplot 繪圖模組還要使用 numpy 模組，來存放所需要的連續數據。（補充說明：若使用 Anaconda3，那以上模組都是預設模組，通通不用再額外安裝）

▶ 直線

我們人類繪製直線是使用直尺，但電腦並沒有直尺，電腦如何繪製直線呢？答案就是要先建立直線方程式，然後再根據直線方程式一點一點密集繪製，這些密集的點，看起來就是直線。例如，設有直線方程式如下：

```
y=ax+b
```

要繪製 x 從 -5 到 5 的直線，程式如下：

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(-5, 5, 1) #-5開始到5，間隔是1，不含結束點5，間隔可
                          以實數，且x爲串列
print(x) #[-5 -4 -3 -2 -1  0  1  2  3  4]
y = 2*x +1 #  y爲串列
print(y) #[-9 -7 -5 -3 -1  1  3  5  7  9]
plt.plot(x,y) #使用直線連接以上座標
plt.axhline(y=0) #繪出x軸
plt.axvline(x=0) #繪出y軸
plt.show() #於螢幕輸出結果
```

執行結果如圖 7-7：

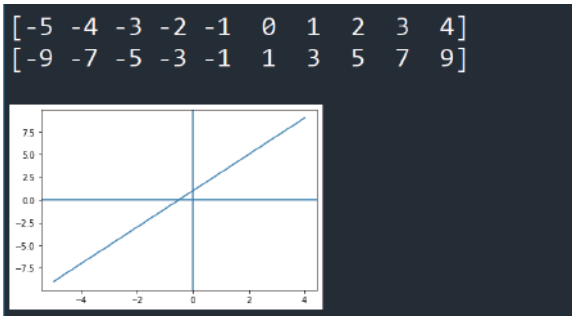


圖 7-7　直線圖

arange() 方法是產生一個 list（串列），本例從 -5 開始到 5，間隔是 1，不含結束點 5，且間隔可以實數。請留意圖 7-7 第一、第二列數字是以下程式的輸出結果。

```
x = np.arange(-5, 5, 1) #-5開始到5，間隔是1，不含結束點5，且x爲串列
print(x) #[-5 -4 -3 -2 -1  0  1  2  3  4]
y = 2*x +1 #  y爲串列
print(y) #[-9 -7 -5 -3 -1  1  3  5  7  9]
```

自我練習

1. 請分別繪製以下聯立直線方程式。

(1) 3x+y=4 6x+2y=8	(2) 3x+y=4 3x+y=8	(3) 3x+y=4 2x-y=1

▶ 二次曲線

前面 x 僅一次方，圖形是直線，若 x 是兩次、或三次，就形成曲線，x 二次會是拋物線，二次方係數爲正就凹向上，會有極小值；二次方係數爲負就凹向下，會有極大值，請觀察以下圖形。

$y=(x-3)^2+1$ (x=3有極小值1)
$=x^2-6x+10$

繪製以上圖形的程式如下：

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(-3, 10, 1) #數字是我多次修正的結果，這樣可以表現圖形
                           最有變化的部分
print(x)
y = x *x-6*x+10
print(y)
plt.plot(x,y)
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

以上程式執行結果如圖 7-8：（x 從 -3 到 10，這是我慢慢觀察圖形，所調整出來的範圍，這樣才能畫出函數最有變化的範圍）

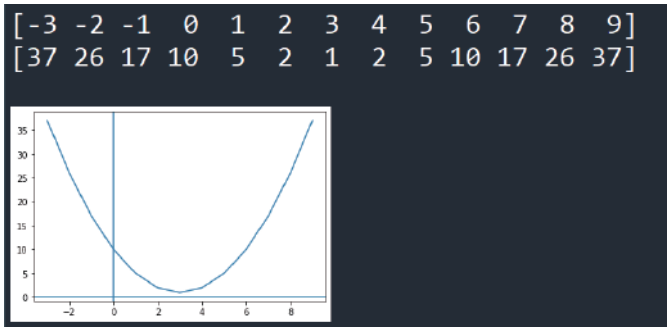


圖 7-8　一元二次方程式圖形

以下是一元三次方圖形，一元三次方圖形大部分有兩個臨界點，或稱相對極值點。例如：

$y=2x^3-3x^2-12x+3$

將以上對 x 微分，得到斜率方程式如下：

$dy/dx=6x^2-6x-12$

設其斜率爲零如下：

$dy/dx=6x^2-6x-12=0$

解出 x=-1 或 x=2 時斜率爲零，此點即爲臨界點，撰寫程式驗證如下：

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(-2, 3, 0.1)# 這是我慢慢觀察圖形，所調整出來的範圍
y=2*x**3-3*x**2-12*x-3
plt.plot(x,y)
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

執行結果如圖 7-9：

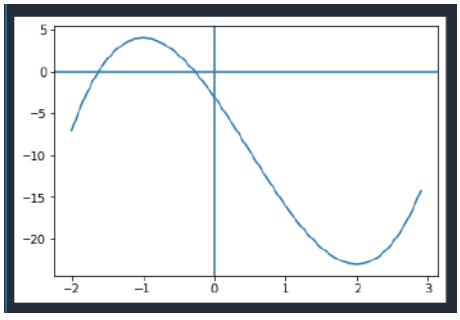


圖 7-9　一元三次方程式圖形

▶ 絕對值函數

方程式有一個決對值，就會有一個轉折點，請看以下程式。

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(-6, 6, 0.1)
y1=abs(x-2)+1#一個折點
y2=abs(x+3)+abs(x-1)+1#兩個折點
plt.plot(x,y1)
plt.plot(x,y2)
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

以上執行結果如圖 7-10：

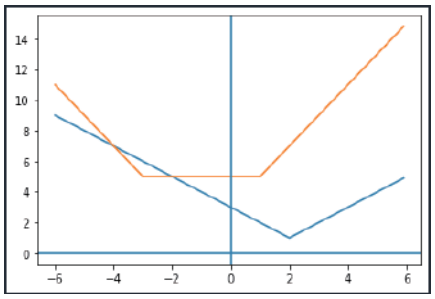


圖 7-10　絕對值曲線圖

▶ 指數函數

x 是指數，例如：

```
y=a^x    a>0 ,a<>1 ,x可爲任意實數
```

稱爲指數函數，此爲高一數學。此一函數的第一個重點是，要討論定義域與值域，所有 x 值所成的集合稱爲定義域，其值可從負無限大到正無限大。第二個重點是，所有 y 值所成的集合稱爲值域，值域僅會大於 0，驗證程式如下：

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(-3, 3, 0.1) #x從-3~3，每次遞增0.1，不含結束點3
print(x) #-3,-2.9,-2.8…0…2.8,2.9
y=2**x #**次方運算子
plt.plot(x,y) #以x,y串列爲資料，繪出圖形
plt.axhline(y=0) #繪出x軸
plt.axvline(x=0) #繪出y軸
plt.show() #於螢幕顯示此圖形
```

執行結果如圖 7-11 所示，請留意 x=0 時 y 一定等於 1。

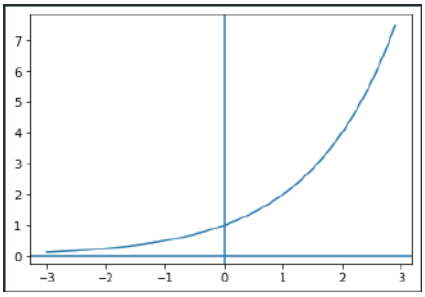


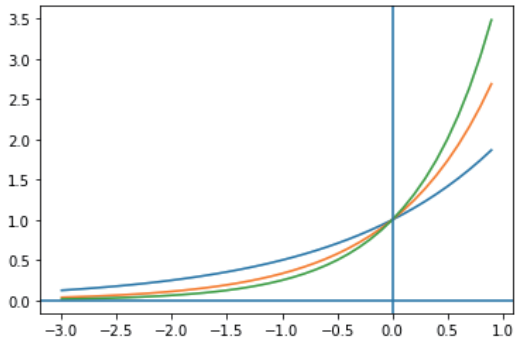
圖 7-11　指數函數圖

函數遞增與遞減

函數遞增與遞減是數學的考試重點，本節就是要用電腦繪圖，不斷繪圖，這樣就會是圖像式記憶法，這些圖形的特徵就會烙印腦海，考試很快可以寫出答案，把數學老師嚇昏。指數函數考試的重點在於 a>1 與 a<1，首先，先介紹 a>1 時，以下程式可比較 x>0 與 x<0，x>0，a 越大，當 x 值愈大時，y 值也愈大；x<0，a 越大，當 x 值愈大時，y 值反而愈小，驗證程式如下：

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(-3, 1, 0.1)
y1=2**x
y2=3**x
y3=4**x
plt.plot(x,y1)#藍
plt.plot(x,y2)#橘
plt.plot(x,y3)#綠
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

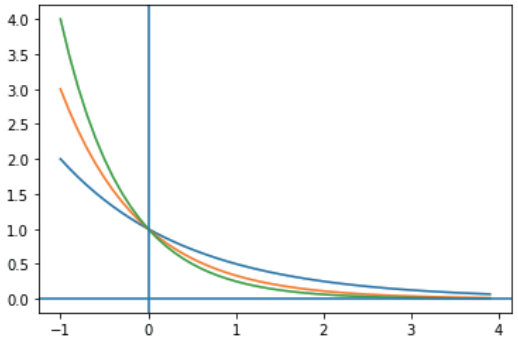
以上程式執行結果如下圖：（自己執行，就會有顏色）



當 a 小於 1 時，以下程式可比較 $x>0$ 與 $x<0$ 時，$y=(1/2)^x$, $y=(1/3)^x$, $y=(1/4)^x$ 的大小關係，此爲數學升學考試的重點，請自己將圖形映在腦海，那考試時，就很快可以寫出答案，這樣可以把數學老師嚇昏或冒冷汗，這就是圖像式記憶法。

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(-1, 4, 0.1)
y1=(1/2)**x
y2=(1/3)**x
y3=(1/4)**x
plt.plot(x,y1)#藍
plt.plot(x,y2)#橘
plt.plot(x,y3)#綠
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

以上程式執行結果如下：（自己執行，就會有色彩）



自我練習

1. 請繪圖，並觀察 $y=2^x$ 與 $y=(1/4)*x^2$ 的交點個數。
2. 請繪圖，並觀察 $y=2^x$ 與 $y=x-3$ 的交點個數。

3. 請繪圖，並觀察 $y=2^x$ 與 $y=x+3$ 的交點個數。

對數

指數函數的反函數，就稱爲對數函數（什麼是反函數，例如，由座號得到姓名若是一個函數，那由姓名得到座號，就稱爲反函數）。例如：

$$y=log_a x$$

以上式子與

$$a^y=x$$

相同，x 稱爲眞數，a 稱爲底，a 僅能大於 0，且不能等於 1（請回想前面 1 的任何次方都是 1，所以對數的底就不能 1 了，因爲底數 =1 的對數值不唯一

$$log_1 1=1$$
$$log_1 1=2$$
$$log_1 1=100$$

其次，因爲指數函數的值域大於 0，現在反過來看，那就變成眞數 x 的定義域僅能大於 0。以下是 log 函數的 Python 表示法，請先鍵入，並觀察執行結果。

```
import numpy as np
import math
print (np.log(math.e))#base is e
print(np.log10(10))#base is 10
print(np.log2(2))#base is 2
```

其次，Python 對數函數的底僅有 10、*e* 與 2，若要其他數字當底數，則要使用換底公式。換底公式如下：

$$log_a b=logb/loga$$

例如，要得到 $log_3 9$，則程式如下：

```
print(np.log(9)/np.log(3))#  2
```

根據以上定義，若有對數函數如下

$y=log_2x$

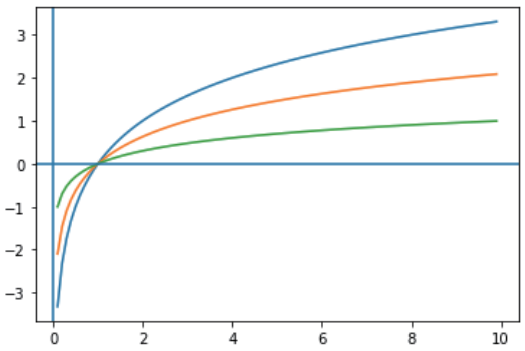
$y=log_3x$

$y=log_{10}x$

則其繪圖程式如下，請留意 x 要大於 0

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0.1, 10, 0.1)#不能從0開始
y1=np.log2(x)
y2=np.log(x)/np.log(3)#換底
y3=np.log10(x)
plt.plot(x,y1)#藍
plt.plot(x,y2)#橘
plt.plot(x,y3)#綠
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

以上程式執行結果如下，圖形一定經過 (1,0) 。

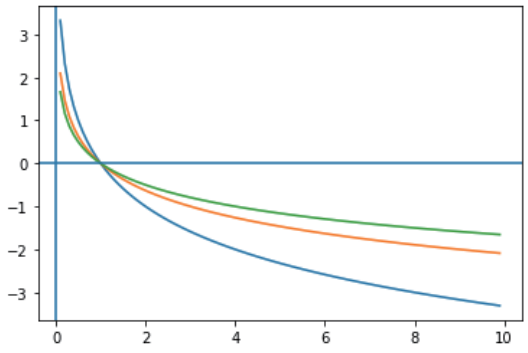


上圖是 $a>1$ 的圖形，請留意 $x>1$ 與 $x<1$ 時三個函數的大小，這就要記圖形的輸出結果，數學考試時才能很快寫出答案。以下程式可繪出 $y=log_{1/2}x$、$y=log_{1/3}x$、$y=log_{1/4}x$，這些都是 $a<1$ 的圖形。

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0.1, 10, 0.1)#不能從0開始
y1=np.log(x)/np.log(1/2)
y2=np.log(x)/np.log(1/3)
y3=np.log(x)/np.log(1/4)
```

```
plt.plot(x,y1)#藍
plt.plot(x,y2)#橘
plt.plot(x,y3)#綠
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

以上程式執行結果如下，圖形一定經過 (1,0) 。數學考試也是常考 $x>1$ 與 $x<1$ 時，三者的大小關係，也是記下以下執行結果，那考試很快就可寫出答案。

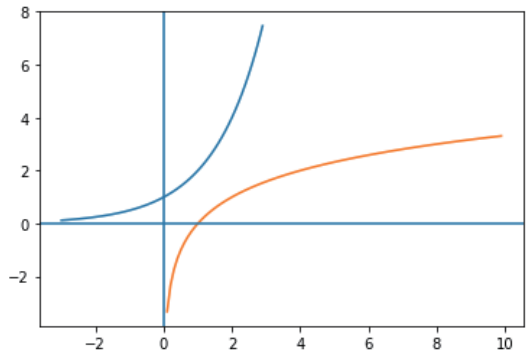


指數與對數的對稱驗證

以下程式可繪出 $y=2^x$（藍色）與 $y=log_2x$（橘色）的圖形，他們對稱 x-y=0。

```
import matplotlib.pyplot as plt
import numpy as np
x1 = np.arange(-3, 3, 0.1)
y1=2**x1
plt.plot(x1,y1)#藍
x2 = np.arange(0.1, 10, 0.1)#不能從0開始
y2=np.log2(x2)
plt.plot(x2,y2)#橘
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

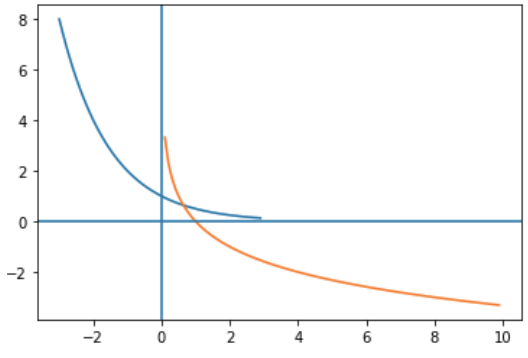
以上程式執行結果如下：



以下程式可繪出 $y=(1/2)^x$（藍色）與 $y=log_{1/2}x$（橘色）的圖形，他們對稱 x-y=0。

```
import matplotlib.pyplot as plt
import numpy as np
x1 = np.arange(-3, 3, 0.1)
y1=(1/2)**x1
plt.plot(x1,y1)#藍
x2 = np.arange(0.1, 10, 0.1)#不能從0開始
y2=np.log(x2)/np.log(1/2)
plt.plot(x2,y2)#橘
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

以上程式執行結果如下。



自我練習

1. 請寫程式，繪出 $y=log_2x$ 與 $y=x$ 的圖形，觀察有幾個交點。
2. 請寫程式，繪出 $y=log_2x$ 與 $y=x^2$ 的圖形，觀察有幾個交點。

3. 方程式 $|log_2x|=2-x$ 有幾個實數解？（提示：同時繪出 $y=|log_2x|$ 與 $y=2-x$，再用肉眼觀察）

▶ 三角函數

任意三角形有三個邊，三個角，科學家爲了找出其邊與角的關係，就定義出三角函數。例如已知三角形任兩邊與其所夾角度，就可透過餘弦定理找出第 3 邊的長度。所以現代人蓋六角庭，蓋橋樑、蓋房子，可以將所有木頭或鋼板長度通通事先裁切好，到了現場就是組裝而已。但是古代的工匠就窘了，他們沒有三角函數，所以是一面作，一面量下一根木頭長度，一面裁切，那速度當然慢。以下是 sin() 函數的定義，變數 x 是角度，單位是徑度量。

```
y=sin(x)
```

x 定義域是 0 到 2π，值域在正負 1 之間，所以撰寫程式如下：

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 2*np.pi+0.1, 0.1) #x從0~2pi遞增，每次遞增0.1
y1=np.sin(x) #計算sin(x)
y2=np.cos(x) #計算cos(x)
plt.plot(x,y1) #繪出sin(x)圖形
plt.plot(x,y2) #繪出sin(x)圖形
plt.axhline(y=0) #繪出x軸
plt.axvline(x=0) #繪出y軸
plt.show() #於螢幕輸出圖形
```

執行結果如圖 7-12。

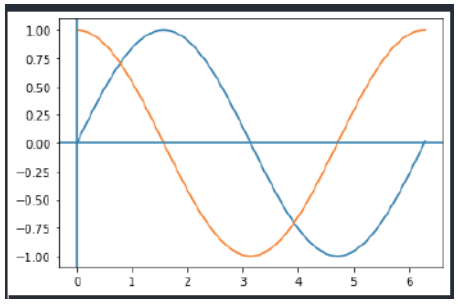


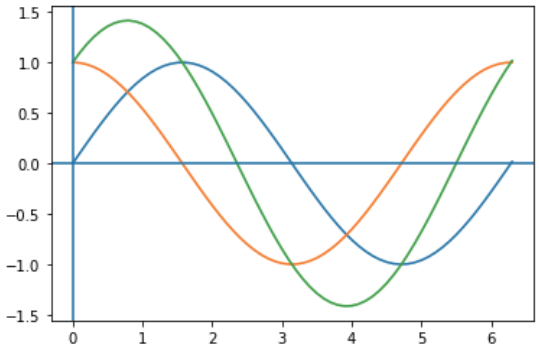
圖 7-12　三角函數圖

函數相加圖形

以下程式可繪出兩個函數的相加。

```
x = np.arange(0, 2*np.pi+0.1, 0.1)
y1 =np.sin(x)
y2=np.cos(x)
y3=np.sin(x)+np.cos(x)
plt.plot(x,y1)#藍色
plt.plot(x,y2) #橘色
plt.plot(x,y3)
```

執行結果如下：

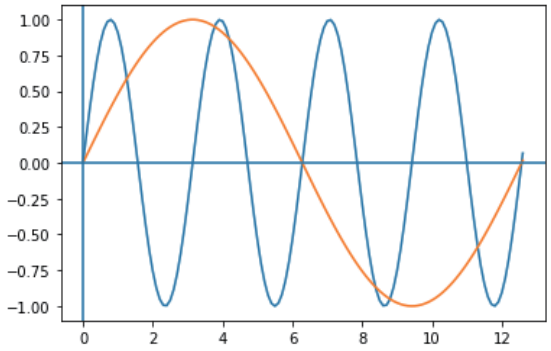


sin(2x)與sin(x/2)週期變化

sin(x) 週期是 2π ，那 sin(2x)、sin(x/2) 週期爲何呢？這也是多繪幾次函數圖，就很容易記起來。繪製 sin(2x) 與 sin(x/2) 函數圖的程式如下：

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 4*np.pi+0.1, 0.1)
y1 =np.sin(2*x)
y2=np.sin(x/2)
plt.plot(x,y1)#藍色
plt.plot(x,y2)#橘色
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

執行結果如下：



由上圖可知，sin(2*x*) 函數週期變一半，sin(*x*/2) 函數週期變兩倍，這些問題，都是數學老師常考的題目。

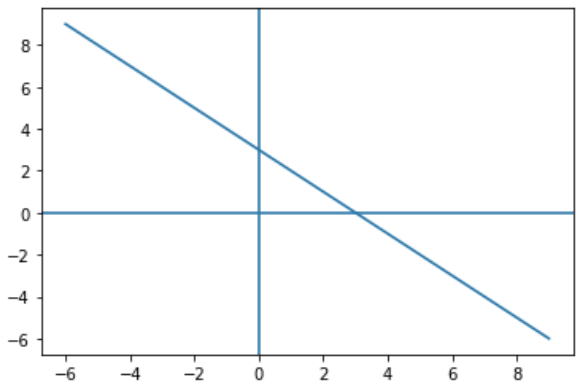
自我練習

1. 請自己練習繪出另 4 個三角函數圖形。
2. 請圖示 *y*=sin(*x*)，*y*=*x*+1，並觀察有幾個交點。

暴力法繪圖

所有實數，所有 *P*(*x*,*y*) 所成的圖形 , 滿足 *x*+*y*-3=0，將會是一直線，請寫程式證明。

執行結果



程式列印

數學老師說『所有實數，所有 P(x,y) 所成的圖形』這句話，眞有如文言文一樣簡潔，程式設計就要使用『雙迴圈』來表示，所以程式如下：

```
import matplotlib.pyplot as plt
x=[]
y=[]
for i in range(-10,10,1):
    for j in range(-10,10,1):
        if i+j-3==0:
            x.append(i)
            y.append(j)
plt.plot(x,y)#藍
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

自我練習

1. 所有實數，所有 $P(x,y)$ 所成的集合，F_1 是 (3,0)，F_2 是 (-3,0)，且 $PF_1+PF_2=10$，將會是橢圓，上式以方程式表示爲 $\sqrt{(x-3)^2+y^2}+\sqrt{(x+3)^2+y^2}=10$，請寫程式以暴力法繪出此圖形。
2. 所有實數，所有 $P(x,y)$ 所成的集合，F_1 是 (3,0)，F_2 是 (-3,0)，且 $|PF_1-PF_2|=4$，將會是雙曲線，請寫程式證明。

圓

人類必須靠圓規畫圓，但電腦沒有圓規，那電腦如何畫圓，此與直線一樣，也是要先造方程式，再依照方程式一點一點密集的點上去，看起來就是圓，圓的方程式有兩種，一種是直角座標。例如：

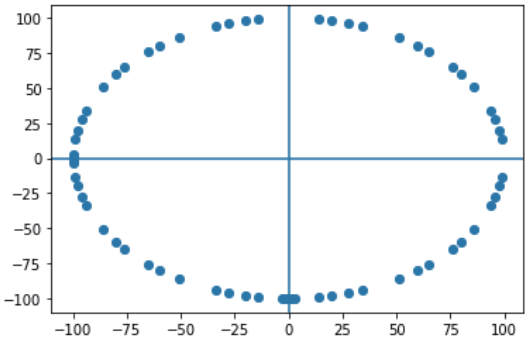
$$x^2+y^2=100$$

以上方程式有兩個變數，可以用以上暴力法繪製，程式如下：（若使用 if abs(i**2+j**2-10000)==0 則合乎條件的點太少了，請自己練習）

```
import matplotlib.pyplot as plt
x=[]
```

```
y=[]
for i in range(-100,100,1):
    for j in range(-100,100,1):
        if abs(i**2+j**2-10000)<10:
            x.append(i)
            y.append(j)
plt.scatter(x,y)#藍
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

以上程式執行結果如下。



其次，雖然有兩個變數，但是，因爲兩個變數有關係，所以可以簡化變數如下：

$$y = \pm\sqrt{10000 - x^2}$$

根據以上方程式，程式撰寫如下：

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(-100, 100+1, 1)
y1 = (10000-x*x)**(1/2)
y2 = -(10000-x*x)**(1/2)
plt.plot(x,y1)#藍色
plt.plot(x,y2)#橘色
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

執行結果如下：

參數式

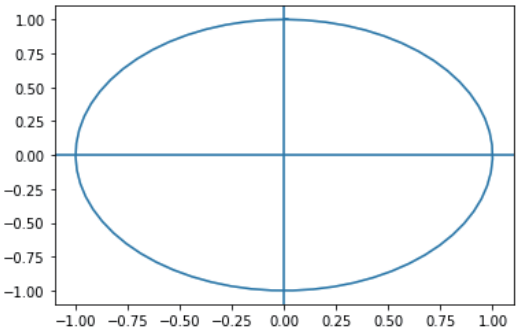
圓、橢圓、雙曲線都還有參數式，這通通可用來畫出圖形。例如，圓 $x^2+y^2=1$ 的參數式如下：

$$x=\cos(x)$$
$$y=\sin(x)\quad 0<x\leq=2\pi$$

利用以上參數式畫圖，程式如下：

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 2*np.pi+0.1, 0.1)
y1 =np.sin(x)
y2 =np.cos(x)
plt.plot(y1,y2)#藍色
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
```

執行結果如下：



自我練習

1. 假設橢圓的參數式如下，請寫一程式繪出圖形。

   x=2cos(x)

   y=3sin(x)

2. 假設雙曲線參數式如下，請寫一程式繪出圖形。

   x=atan(x)

   y=bsec(x)　$0<x\leq=2\pi$　x $\neq \frac{\pi}{2},\frac{3}{2}\pi$

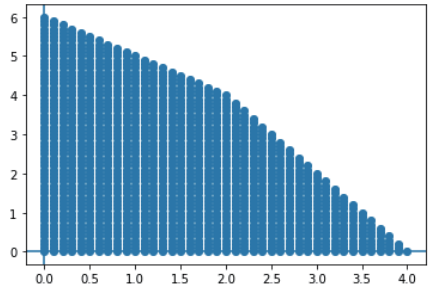
線性規劃

請繪出受制於 x>=0,y>=0,x+y<=6,2x+y<=8 條件下，並求出 f(x,y)=-2x+3y 的極大值。

程式列印

```
import matplotlib.pyplot as plt
x=[]
y=[]
s=10
max=-10000
for i in range(10*s):
    for j in range(10*s):
        if ((i/s+j/s)<=6) and ((2*i/s+j/s)<=8) :
            x.append(i/s)
            y.append(j/s)
            f=-2*i/s+3*j/s
            if f>max:
                max=f
plt.scatter(x,y)#藍色
plt.axhline(y=0)#x軸
plt.axvline(x=0)#y軸
plt.show()
print(max)
```

執行結果如下：




```
18.0
```

自我練習

1. 受制於 $x>=0,y>=0,x+y<=3,2x+y<=4$ 的條件下，請繪出滿足條件的區域，並求 $f(x,y)=x+3y$ 的最大值。(答案是 9)
2. 請寫程式圖示 x,y 爲正實數，且滿足 $x+y=4$ ，則繪出 $f(x,y)=1/x+1/y$ 的圖形，其最小值爲何？（答案是 1）
3. 請寫程式圖示 $|x-1|+|y-3|<=2$ 的區域，並求 $f(x,y)=x+2y$ 的最大值。（答案是 11）
4. 在座標平面上，設 S,T 爲不等式 $|x|+3|y|<=7$ 的圖形上任意兩點，請圖示其範圍，且求 S,T 距離最大值爲何？
5. 請寫一程式，圖示 $|x+y|<=8$,$|x-y|<=8$ 的區域。
6. 請寫程式，圖示不等式 $2|x+1|<=|x-2|$ 的範圍。
7. 請寫程式，圖示不等式 $1<=|2x-1|<3$ 的範圍。

7-3 自定函式

若內定函式沒有可用的函式，當然可以自己寫，這樣可以讓程式模組化，增加程式維修的便利性。例如，以下就是很簡單的範例，請先鍵入此程式，觀察執行結果。請留意 Python 的縮排有意義，也代表函式的範圍；冒號 (:) 也是 Python 的語法，不可省略。

```
#自訂函式
def add(a,b): #參數a接收m，參數b接收n
    print(a,b)
    c=a+b
    return c #將結果c傳回
#主程式
m=3;n=4
p=add(m,n) #主程式的m,n稱爲引數
#呼叫自訂函式add，m傳給函式的a，n傳給函式的b
#函式傳回值由主程式p接收
print(p)#7
```

此程式 add(a,b) 稱爲自訂函式，由主程式 add(m,n) 呼叫才執行，且接收傳遞而來兩個引數，a 接收 m 之值 ,b 接收 n 之值，主程式的 m,n 稱爲引

數（Argument），函式的 a,b 稱爲參數（Parameter），自訂函式 add，將兩個參數相加，由 c 儲存，並將結果 c 傳回，由主程式 p 接收。以下我們分別說明 Python 函式的宣告與定義、函式參數傳遞、函式傳回值。

函式的宣告與定義

Python 函式的宣告與定義語法如下：

```
def 函式名稱([參數1, 參數2, 參數3…]) :
    [敘述區塊]
    [pass]
    [return 回傳1,回傳2,回傳3…]
```

以上語法說明如下：

1. 函式名稱可以是任何合法識別字，例如：aa、a1、add、inputdata 等。但盡量用函式功能命名，可讀性才高，例如 : add、sum、inputdata 等。
2. 函式名稱當然不能用 Python 保留字，例如：for、if。
3. 函式的呼叫就直接使用函式名稱加上所需傳遞的引數就可以。例如，

```
add(3,4) #將(3,4)傳給以上add()函式，參數a得到3，參數b得到4
print(a)#7
m=3;n=4
add(m,n) #將(m,n)傳給以上add()函式，參數a得到m，參數b得到n
```

4. 函式的位置要放在主程式的前面，以下是全部程式列印。

```
#函式
def add(a,b):
    c=a+b
    return c
#主程式
p=add(3,4)
print(p)#7
```

5. Pathon 因爲縮排關係，也不允許空函式，所以若函式沒內容（有可能先定義好介面，待後續再寫入程式），也是先放一個 pass。例如：

```
def aa():
    pass
#主程式
aa()
```

6. 函式名稱若與 Python 內建函式同名，則此自訂函式可改寫內建函式功能。例如：len() 是內建函式，可求物件的長度，以下則又使用 len 定義使用者自訂函式，等於改寫 len() 函式功能。

```
def len(a,b):#a接收m，b接收n
    c=a+b
    return c #將結果c傳回
p=len(3,4)
print(p)#7
```

函式參數傳遞

1. 參數可以是任意數量或無，例如：def add()、def add(a)、def add(a,b)。
2. 引數的傳遞若不指名，則是按照引數位置順序接收。例如：

```
def add(a,b):#a接收m，b接收n
    print(a,b)
m=3;n=4
p=add(m,n)
```

3. 主程式引數若指名，則按照指定的名稱傳遞。例如：

```
def add(a,b):#a接收n，b接收m
    print(a,b)
m=3;n=4
p=add(a=n,b=m) #於主程式指定引數與參數的接收方式
```

4. 函式參數若有預設值，則當主程式沒給此引數時，可用預設值替代。

```
def add(a,b=0):#本例參數b有預設值0
    c=a+b
    return c
m=3;n=4
print(add(m,n)) #7    有給2個參數
print(add(m)) #3    第2個參數b沒給，參數b用預設值0
```

5. 引數若是不定數量，請在參數前加上（*）號，例如：

```
def add(*a):     # *代表可接收不定數量的引數
    c=0
    for a1 in a:
        c=c+a1
    return c
#主程式
print(add()) #0    可以沒有引數
print(add(3)) #3    1個引數
print(add(3,4)) #7   2個引數
```

6. 引數可以任何資料型態，單一變數、串列 (list) 等等都可以當作引數。而且資料型態是可變的，完全依照使用者傳入的資料型態（此即爲物件導向「多型」的概念）。例如：請鍵入以下程式，寫出執行結果。

```
def aa(a):
    print(a)
#主程式
aa(3) #數值
aa('a') #字元
aa("Mary") #字串
aa(True) #布林值
aa([3,4,5]) #list
```

7. 以下引數則是傳遞 list。

```
def add(a) :
    c=0
    for a1 in a:
        c=c+a1
    return c
#主程式
b=[3,4,5]
print(add(b)) #12
```

函式傳回值

Python 函式可用參數或 return 傳回值給主程式，分別說明如下：

▶ 引數傳回值

函式的參數值的改變不傳回者，稱爲傳值呼叫。單一變數、tuple、等都是傳值呼叫。例如：以下程式，參數 a 是單一變數，a 值雖在函數被修改，但並不回傳。

```
def aa(a):
    print(a) #3
    a=4
#主程式
m=3
aa(m)
print(m) #3
```

tuple 不論在主程式或函式都不能修改其值，所以是單向傳遞。

參數若是串列、dict、set，則是傳址呼叫，傳址呼叫表示此參數在主副程式共用位址，任一方的改變，都是改變同一變數，也就是參數在函式的改變，會傳回主程式，是一種雙向傳遞，此稱爲傳址呼叫。

以下程式是串列的傳址呼叫，串列若在函式有變更其值，也會將變更結果回傳。

```
def bb(a):
    print(a) # [3,4,5]
    a[1]=6 #修改串列的內容
#主程式
m=[3,4,5]  #list 本書翻譯爲串列
bb(m)
print(m) #  串列有傳回修改結果
#[3,6,5]
```

因爲串列是傳址呼叫，若不想接收被修改的結果，請改爲傳串列副本到函式，如以下程式的 bb(m[:])。

```
def bb(a):
    print(a) # [3,4,5]
    a[1]=6 #修改串列的內容
#主程式
```

```
m=[3,4,5]  #list 本書翻譯爲串列
bb(m[:]) #傳副本
print(m) #  本例是傳串列副本，所以沒傳回
#[3,4,5]
```

以下程式是 dict 的傳址呼叫。

```
def aa(a):
    print(a) # {'book': '書', 'pen': '筆'}
    a["book"]="黃金" #修改 dict的內容
#主程式
m={"book":"書","pen":"筆"}  #dict
aa(m)
print(m) #  dict 有傳回
#{'book': '黃金', 'pen': '筆'}
```

以下程式是 set 的傳址呼叫。

```
def aa(a):
    print(a) #{3,4}
    a.add(6)
#主程式
m={3,4}
aa(m)
print(m) #{3,4,6}
```

C/C++ 可用修飾字改變參數爲傳值或傳址，Python 則不行。也就是 Python 是由資料型態種類決定其爲傳值或傳址，而且沒有修飾字改變其傳遞方式。

▶ return 傳回值

以上是用參數傳回函式值，Python 也可用 return 傳回值，return 傳回值的語法如下：

```
[return 回傳1,回傳2,回傳3…]
```

1. 傳回值可以是任何資料型態，個數也沒有限制，傳回幾個都可以，但是主程式也要使用相同個數的變數接收回傳值。例如，以下程式未傳回任何值。

```
def aa():
    pass
#主程式
aa()
```

2. 以下程式傳回一個單一變數，主程式也要用一個變數接收回傳值。

```
def aa():
    b="Mary"
    return b
#主程式
c=aa()
print(c)
```

3. 以下程式傳回一個單一變數與一個串列，主程式也要用相同數量變數接收回傳值。

```
def aa()  :
    c=1
    d=[2,3]
    return c,d #傳回個數沒有限制
a,b=aa() #a,b個數要與return c,d一致
print(a) #1
print(b) #[2,3]
```

範例7-3a

試寫一程式，計算 C_n^m 的值。

提示：$C_n^m = \frac{m!}{n!(m-n)!}$ 。

m!=1*2*3…*m（此爲高中數學）

程式列印

方法一，未使用函式，階乘共需撰寫 3 次：

```
m=5
n=2
s1=1
for i in range(1,m+1):
```

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

方法二：使用函式。本例階乘共需寫三次，所以寫成函式 fsum。

```
def fsum(a):
    s=1
    for i in range(1,a+1):
        s=s*i
    return s
m=5
n=2
s1=fsum(m)
s2=fsum(n)
s3=fsum(m-n)
print(s1/(s2*s3))#10.0
```

自我練習

1. 請寫一個程式，內含四個函式，分別可執行兩個整數的加減乘除等四個功能。

7-4 函式應用實例演練

範例7-4a

判斷任意點 D 是在三角形 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 是在三角形 ABC 內或外？

執行結果

```
10.0 5.0 2.0 3.0
IN
```

運算思維

D 若在三角形內，則三角形 ABC 面積＝△ DAB ＋△ DBC ＋△ DAC 的面積。

因爲三角形面積要計算四次，所以，先以 aa() 函式寫好，每次計算面積，只要呼叫 aa() 函式就好，程式如下：

```
def aa(x1,y1,x2,y2,x3,y3):
    return abs((x1*y2+x2*y3+x3*y1-x2*y1-x3*y2-x1*y3))/2
#三角形ABC，A(x1,y1)，B(x2,y2),C(x3,y3)
x1=0;y1=0
x2=0;y2=5
x3=4;y3=0
x4=2;y4=1#D點
a=aa(x1,y1,x2,y2,x3,y3)#原來三角形面積
a1=aa(x4,y4,x1,y1,x2,y2)#DAB面積
a2=aa(x4,y4,x1,y1,x3,y3) #DAC面積
a3=aa(x4,y4,x2,y2,x3,y3) #DBC面積
print(a,a1,a2,a3)
if a==(a1+a2+a3):
    print("IN")
else :
    print("NOT IN")
```

補充說明

1. 本例變數的取法要有技巧，不然程式不好寫。
2. 本例若沒有使用函式，計算三角形面積要寫四次，程式不但很冗長，也沒有一致性，錯在哪裡也不曉得。

範例 7-4b

主客場籃球賽制。（108 年 APCS 第二梯次試題）

1. 連打兩場，主場球隊兩場皆贏輸出 Win。
2. 兩場皆輸，輸出 Loss。
3. 其餘輸出 Tie。
4. 輸入格式：每場 4 節得分，主隊先輸入。以下分別是主隊、客隊兩場比賽的四小節得分

```
15 20 25 30
15 16 17 18
5 10 15 20
5 6 7 8
```

執行結果

```
input a b c d:15 20 25 30

input a b c d:15 16 17 18

input a b c d:5 10 15 20

input a b c d:5 6 7 8
Win
```

程式列印

參考解答如下：輸入資料與計算總分共要執行四次，所以將輸入資料與計算總分寫成函式 inputdata()。

```
def inputdata():
    p=input('input a b c d:')
    q=p.split(' ')#本例測資是以空白分隔，所以要以split分解，請看8-1節
    print(q)
    r=[int(q1) for q1 in q]#將字串轉爲數值
    print(r)
    return sum(r)#計算4小節總和
a1=inputdata()
b1=inputdata()
a2=inputdata()
```

```
b2=inputdata()
num=0
if a1>b1 :
    num=num+1
if a2> b2:
    num=num+1
if num==2:
    print('Win')
else:
    if num==0:
        print('Loss')
    else:
        print('Tie')
```

Chapter
08

綜合應用實例

字串的運算

Python 並沒有細分字元或字串型態，用單引號『b='Horng'』或雙引號『a="Horng"』圍起的字元，通通是字串，例如：

```
a="Horng"
print (a)
```

要從字串取字元，可以同串列的取單一元素。例如：

```
print(a[0])#H
b='Horng'
print(b)
print(b[0])
```

⚙⚙字串與串列、字串與集合互轉

請鍵入以下程式，並觀察執行結果。

```
a="AABBCCC"
b=list(a) #字串轉串列
print(b) #['A', 'A', 'B', 'B', 'C', 'C', 'C']
c="".join(b) #串列轉字串
print(c) #AABBCCC
d=set(a) #字串轉集合
print(d) #{'A', 'C', 'B'} #已經剔除相同字元
e="".join(d) #集合轉字串
print(e) #ACB
```

子字串也可用子串列方法處理。例如，請鍵入以下程式，並觀察執行結果。

```
a='abcdef'
print(a[0]) #a取索引0
print(a[-1]) #f 取倒數第1個
print(a[-2]) #e取倒數第1個
print(a[0:3]) #abc 取索引0～2
print(a[2:]) #cdef 取索引2開始到結束
print(a[-2:]) #ef  取倒數兩個
print(len(a)) #6 計算長度
print(max(a)) #f 取最大元素
b=list(a)  #轉爲串列 ['a', 'b', 'c', 'd', 'e', 'f']
print(b)
c="".join(b) #轉爲字串 abcdef
print(c)
```

▶ split()

幾乎所有程式語言都是使用 split() 分割字串，Python 也不例外。例如：

```
a="There are four people in my family."
b=a.split(" ")#以空白爲分割依據。
print(b)
print(b[0])
print(b[1])
```

以上 b 是一個串列，輸出如下圖：

```
['There', 'are', 'four', 'people', 'in', 'my', 'family.']
There
are
```

▶ 測資含有空白

有些考試測試資料在同一列含有多筆資料，中間以空白隔開，例如

```
2 3 4
```

此時可用 split 分解資料，例如：

```
d=input("input a b c:") #本例假設輸入 2 3 4
print(d) #2 3 4
a,b,c=d.split(' ') #以空白分割資料
print(a+b+c) #字串型態 234
a=int(a);b=int(b);c=int(c)
print(a+b+c) # 9 數值型態輸出
```

▶ map()

map() 函式是 Python 內建資料型態轉換函式，其語法如下：

```
map(function, iterable, ...)
```

以上語法非常彈性，本書並不一一介紹，茲將以上字串分割與型態轉換程式使用 map() 重寫如下：

```
d=input("input a b c:") #本例假設輸入 2 3 4
a,b,c=map(int,d.split(' '))
print(a+b+c) # 9 數值型態輸出
```

自我練習

1. 請寫一程式，讓使用者可以輸入任意數量的整數，整數與整數之間以空白隔開，輸入後可計算其和、平均、極大值與極小值。
2. 請寫一程式，可以計算所輸入句子的字數。例如，'There are four people in my family.' 表示共有 7 個字。
3. 請寫一程式，可以計算所輸入文章的句數。例如，' This is a book. There are four people in my family.' 表示共有 2 句。

▶ replace(ch1,ch2)

將 ch1 子字串由 ch2 取代。例如：以下程式可將全部的 are 都置換爲 is。

```
a="There are four people in my family."
b=a.replace("are","is")
print (b)#There is four people in my family.
```

若 ch2 是空白，則是刪除 ch1 字串。例如：以下程式可刪除 is 。

```
a="This is a book."
b=a.replace("is","") #將所有 is 全部刪除
print(b) #Th  a book.
a=a.replace(" is","") #僅刪除單字is，請留意有無空白的差異
print(a) #This a book
```

▶ startswith()、endswith()

傳回字串是否由某一子字串開頭或結束。

```
a="洪國勝"
print(a.startswith("洪"))#True
print(a.endswith("勝"))#True
```

範例8a

統一發票對獎。請寫一程式，可以幫使用者核對統一發票中獎情形。

執行結果

程式列印

1. 將中獎號碼以字串儲存，如以下程式的 a1~a4。
2. 以下我們僅核對 1 張發票，以變數 b 儲存。

```
a1='59647042'#特別獎號碼
a2='01260528'#特獎號碼
a3=['01616970','69921388','53451508']#頭獎號碼
a4=['710','585','633']#增開六獎的號碼
#b='31616970' #測試資料1，欲核對的發票號碼
b='11111710' #測試資料2，欲核對的發票號碼
c=''
if b==a1 :
    c='特別獎'
elif b==a2 :
    c='特獎'
else:
    #逐一核對頭獎
    for i in range(len(a3)): #本例頭獎有三個
```

```
        d=a3[i]
        if b==d:
            c='頭獎'
        elif b[1:]==d[1:]:#大家都取末七位數，末七位數相同稱爲二獎
            c='二獎'
        elif b[2:]==d[2:]: #大家都取末六位數，末六位數相同稱爲三獎
            c='參獎'
        elif b[3:]==d[3:]: #大家都取末五位數，末五位數相同稱爲四獎
            c='四獎'
        elif b[4:]==d[4:]: #大家都取末四位數，末四位數相同稱爲五獎
            c='五獎'
        elif b[5:]==d[5:]: #大家都取末三位數，末三位數相同稱爲六獎
            c='六獎'
    for i in range(len(a4)):#增開六獎
        if b[5:]==a4[i]:
            c='六獎'
print(c)
```

自我練習

1. 以上範例，我們僅核對一張發票，但我們通常很多發票，請將使用者的 6 張發票以串列儲存，並由電腦核對中獎情形。
2. 請寫一程式，可以找出個字串所含的大寫字元、小寫字元、數字的個數。例如，'AcFgk7' 傳回大寫 2，小寫 3，數字 1

範例8b

中華民國的身分證字號有其特定的編碼原則。第一個字是大寫的英文字母，其餘 9 個字必須爲數字，但在套用編碼原則時，第一個英文字母將會先依下表被轉換爲數字：

字母	A	B	C	D	E	F	G	H	J	K	L	M	N
數字	10	11	12	13	14	15	16	17	18	19	20	21	22
字母	P	Q	R	S	T	U	V	X	Y	W	Z	I	O
數字	23	24	25	26	27	28	29	30	31	32	33	34	35

轉換後的身分證字號 (共 11 位數字) 每一位數均有固定的權重 (Weight)，由左往右依序爲「1, 9, 8, 7, 6, 5, 4, 3, 2, 1, 1」。判斷身分證字號是否正確

的方法爲：各位數字與其相對應的權重相乘後再加總，加總後的結果若爲10的倍數則身分證字號即屬正確。請根據以上說明，寫一程式，驗證身份證號碼是否正確。

執行結果

程式列印

```
#a='O157387560' #測試號碼1
a='A157387560'#測試號碼2
#請留意，題目的字母沒有按照順序，以下我重新調整，由A～Z以串列儲存如下
b=[10,11,12,13,14,15,16,17,34,18,19,20,21,22,35,23,23,25,26,27,
   28,29,32,30,31,33]
c=[1,9,8,7,6,5,4,3,2,1,1]#權重
a0=ord(a[0])#取ASCII編碼
print(a0)#65 A的ASCII編碼是65
a1=b[a0-65]#將第一個字母轉爲數字a1
print(a1)#10
a2=a[1:]#取末9碼157387560
a=str(a1)+a2#成功將第一個字母轉爲數字
print(a)#10157387560 此時共11位數
#使用迴圈，依照11個數字的權重累計其和
s=0
for i in range(11):
    s=s+int(a[i])*c[i]
print(s)#170
if s %10 ==0:
    print("RIGHT")
else :
    print("Worng")
```

自我練習

1. 於範例 8-1b，我們僅檢查一個身份證號碼，請您將 6 個身份證號碼以串列儲存，並一一檢查其是否正確。

※ 2. 請問如何自動產生一個身份證號碼，且此號碼可以通過以上檢查。

範例8-1c

猜單字遊戲。

本例我寫一個猜單字遊戲，遊戲過程如下：

1. 輸出底線的個數提示單字的字母數，例如，單字是 apple，就輸出 5 個底線『_ _ _ _ _』。
2. 使用者每次僅能輸入一個 a 到 z 的字母。本例假設輸入『p』。
3. 電腦將單字中合乎此字母的通通現形，所以本例將會輸出『_ p p _ _』
4. 重複步驟 2 與 3，直到全部子母猜對爲止。

執行結果

```
_ _ _ _ _
請輸入一個字母：a
a _ _ _ _
請輸入一個字母：e
a _ _ _ e
請輸入一個字母：p
a p p _ e
請輸入一個字母：l
a p p l e
BinGo
```

程式列印

```
a='apple'
s=len(a)
b=[]
c=[]
#使用迴圈，逐一取a字串的每個字元放入b串列
#c串列則放入相對應數量的_
for i in range(s) :
    b.append(a[i])
    c.append('_') #c串列放入相對應數量的 _
#輸出c串列所有字元
for i in range(s) :
    print('%c '% c[i],end='')
s1=0  #答對字元計數器
while (s1<s) :  #只要沒有完全猜對，重複迴圈
    d=input('請輸入一個字母：')
```

```
    #使用迴圈，逐一核對，將相同的字元取代原來底線
    for i in range(s):
        if d==b[i]:
            c[i]=d
            s1=s1+1   #答對字元計數器
    #輸出c串列所有字元，以便使用者繼續猜測
    for i in range(s) :
        print('%c '% c[i],end='')
print()
print("BinGo")
```

自我練習

1. 以上範例僅用一個單字，請擴大此題目，找來 50 個單字，每次以亂數出現一個單字。

範例8d

英文默寫。英文要能進步，就是要強迫自己能將看過的句子覆誦一次，您想想看，中文別人講一次，我們通常可以覆誦一次，其實英文也要這樣自我要求。以下是我寫的程式，只要將一篇文章先輸入或複製貼上，程式就會逐一出現每一個句子、每個句子的第一個單字、並將該句剩餘的單字以底線『_』標示，讓使用者填空輸入，輸入完電腦還能自動批改。

執行結果

```
Action ____ ____ ____ ____
請輸入句子0：Action speak louder than word.
錯了1 單字
正確句子是Action speak louder than words.
Wasting ____ ____ ____ ____
請輸入句子1：Wasting time is robbing time.
錯了1 單字
正確句子是Wasting time is robbing oneself.
Never ____ ____
請輸入句子2：Never say die.
全對
```

運算思維

1. 將整篇文章複製，以字串儲存。
2. 將文章以點 (.) 分解爲若干句子。例如，3 個句字有 3 個點，那分解後有 4 句，最後一句是空白，所以本例程式撰寫就減 1。
3. 本例將每一個句子的第 1 個單字輸出，且每一個單字以底線『_』標示，這樣使用者比較好作答，電腦也比較方便批改。
4. 逐一由使用者輸入整個句子。
5. 本例假設使用者所輸入單字數與題目一樣。例如，題目提示 5 個單字，使用者就輸入 5 個單字。
6. 逐一按順序比對兩個句子的單字，並統計正確單字的個數，也就是第 1 個單字與第 1 個單字比對，第 2 個單字與第 2 個單字比對。
7. 若全部單字都對，輸出『全對』，否則輸出正確單字的個數。
8. 輸出正確的句子。

程式列印

```
a="Action speak louder than words.Wasting time is robbing
   oneself.Never say die."
b=a.split('.')#整篇文章以「.」分解
#print(len(b))
#print(b)#由輸出發現，3個點將文章分爲4句，最後1句空白，所以以下就減1
#使用迴圈，逐一輸出每一句英文
for i in range(len(b)-1):
    c=b[i]+'.'    #每個句子將'.'補回來
    #每個句字再以空白分解
    d=c.split(' ')
    dlen=len(d)
    #print(dlen)
    print(d[0],end=' ')#輸出第1個單字
    for j in range(1,dlen) :#依照單字個數，輸出底線
        print('____',end=' ')
    #print('.')
    e=input("請輸入句子%d："%i)#由使用者輸入整句
    f=e.split(' ')#分解使用者輸入的句子
    #print(len(f))
    s=0
```

```
#計算答對幾個單字
for j in range(dlen):#逐一每個單字比對
    if (d[j]==f[j]):
        s=s+1
if s==dlen :
    print("全對")
else:
    print("錯了%d 單字" %(dlen-s))
print("正確句子是%s"%c)
```

自我練習

1. 本例假設使用者輸入的單字數與提示的單字數一致，若使用者輸入太長或太短都會造成執行錯誤，請問您如何修改程式補救。

範例8e

英文聽寫程式的研究。

執行結果

將英文課文複製，然後貼到以下變數 a，電腦就會逐一唸出每一句，且電腦提示每一句的長度，使用者就可開始輸入聽到的單字，然後電腦還會檢查這些單字是否正確。以下是執行結果：

```
____ ____ ____ ____ ____
請輸入句子0：Action speak louder than words.
全對
正確句子是Action speak louder than words.

____ ____ ____ ____ ____
請輸入句子1：Wasting time is rob one.
錯了2 個單字
正確句子是Wasting time is robbing oneself.

____ ____ ____
請輸入句子2：Never say die.
全對
```

運算思維

1. 我們於第一章曾經強調，Python 之所以強，就是有源源不絕的好用套件不斷發表，上網搜尋『Python 文字轉語音』，找到套件如下

（https://codertw.com/%E7%A8%8B%E5%BC%8F%E8%AA%9E%E8%A8%80/368251/）：

```
import win32com.client
speaker = win32com.client.Dispatch("SAPI.SpVoice")
a="Hello, it works!"
speaker.Speak(a)
a="洪國勝，您好"
speaker.Speak(a)
```

2. 以上程式眞的可以發音，也就是視障者想聽某段文章，都可以將任何中英文文章複製，放到變數 a，就可以自動播放。或是，您也可將英文課本複製下來，貼到變數 a, 電腦就會用語音輸出，讓您不斷練習聽力。
3. 根據以上程式，我將以上程式改寫如下，就可完成英文聽寫程式，希望可以幫助大家學習英文。

```
import win32com.client
speaker = win32com.client.Dispatch("SAPI.SpVoice")
speaker.Speak("Hello, 各位同學大家好，現在開始聽寫測驗")
#將課文複製，然後貼來以下變數a,電腦就會逐一唸出每一句。
a="Action speak louder than words.Wasting time is robbing
   oneself.Never say die."
b=a.split('.')#整篇文章分解
#print(len(b))
#print(b)#由輸出發現，b的長度是4，所以以下就減1
#使用迴圈，逐一唸出每一句英文與輸出底線
for i in range(len(b)-1):
    c=b[i]+'.'  #每個句子將'.'補回來
    speaker.Speak(c)#您要念幾遍，就寫幾次
    speaker.Speak(c)
    #每個句字再以空白分解
    d=c.split(' ')
    dlen=len(d)
    #print(dlen)
    #print(d[0],end=' ')
    #使用迴圈，依照句子長度輸出底線
    for j in range(dlen) :
        print('____',end=' ')
    #print('.')
```

```
#由使用者輸入句子
e=input("請輸入句子%d："%i)
f=e.split(' ')
#print(len(f))
s=0
#計算答對幾個單字
for j in range(dlen):
    if (d[j]==f[j]):
        s=s+1
if s==dlen :
    print("全對")
else:
    print("錯了%d 個單字" %(dlen-s))
print("正確句子是%s"%c)
```

4. 讀檔功能。以上是資料和程式寫在一起，這樣對不會程式的人還是困擾，以下我們則要程式和資料分離，將文章先放在記事本，以 a0.txt 存檔（請與 Python 程式放在同一資料夾），如下圖：

5. 撰寫以下程式，就可讀檔，且整篇發音。

```
import win32com.client
speaker = win32com.client.Dispatch("SAPI.SpVoice")
f='a0.txt'
with open(f,'r',encoding='utf-8')as f1:
    data=f1.read()
f1.close()
print(data)
speaker.Speak(data)
```

自我練習

1. 請將以上英文聽寫測驗程式套用讀檔程式 (with open() as f1)，這樣就可以將程式與資料分開，每次只要改變資料就可以，不用改程式。

範例8f

猜數字

有一種遊戲稱爲幾 A 幾 B 的猜數字遊戲，兩個人對玩，互相猜對方預先寫下的四位數（四位數中的阿拉伯數字不可重覆），若所猜的數字與對方位置相同者爲 A，數字相同，位置不對，則稱爲 B。例如對方預寫的數字爲 6713，若猜 6731 則應回應 2A2B，若猜 7652 則應回應 0A2B。試寫一程式，電腦自動產生四位數的亂數，再讓使用者猜的一種遊戲程式，電腦應逐一回應使用者已猜的狀況。其次，四位數亂數，數字不能重複，例如，1001、5566、1022 等都不行。

執行結果

```
1709
輸入四位數：9071
0 a 4 b
輸入四位數：2345
0 a 0 b
輸入四位數：6789
2 a 0 b
輸入四位數：1790
2 a 2 b
輸入四位數：1709
4 a 0 b
BinGo
```

程式列印

```
import random
c=[0,1,2,3,4,5,6,7,8,9]
#將以上數字打亂
for i in range(len(c)):
    d=random.randint(0,9)
    c[i],c[d]=c[d],c[i]
#取前四個數字,此即爲亂數
e=str(c[0])+str(c[1])+str(c[2])+str(c[3])
print(e)
right=False
```

```
a=0
b=0
#使用while 迴圈由使用者重複猜測
while (not right):#只要沒完全答對，重複迴圈
    f=input('輸入四位數：')
    #使用for迴圈，計算a與b的數量
    for i in range(4):
        if f[i]==e[i]:
            a=a+1
        for j in range(4):
            if f[j]==e[i]:
                b=b+1
    b=b-a
    print('%d a %d b'%(a,b))
    if a==4:
        right=True
    a=0
    b=0
print("BinGo" )
```

※範例8g

人工智慧與刪除法。

先將所有可能的答案通通列出來，再逐步將所有不可能的答案一一刪掉，最後剩下的值即爲所求，此稱爲刪除法。例如，同上題，剛剛是人腦猜，人腦有人腦的運算思維，但是電腦有電腦的運算思維，既然是電腦程式設計，那就要利用電腦的強項，才能事半功倍，以下示範電腦用刪除法來猜，使用者必須逐一回應電腦已猜的狀況。電腦猜數字若使用以上刪除法，則演算法如下：

(1) 使用串列列出 0000 至 9999 的四位數。

(2) 逐一刪除阿拉伯數字重複者，例如 1001 或 3343 等。

(3) 於剩下的可能數字中，挑最小的當作猜值，本例爲 0123。（使用者應回應幾 A 幾 B，本例假設使用者的數字是 4237，所以回應 0A2B）

(4) 於剩下的可能值中，使用臆測值，本例爲 0123 逐一比對，將結果是 0A2B 者，蒐集起來，也就是不是 0A2B 者全部刪除。（爲什麼呢？因爲答案 4237 也會是 0A2B，將會被保留，此即爲刪除法，逐一將不可能的答案，先行刪除。）

(5) 重複 (3)、(4) 兩個步驟，直到剩下最後一個，就是答案了，或使用者回應 4A0B 爲止。

執行結果

下圖就是我預設 4237，電腦逐一猜值的過程。本例電腦 5 次就猜出結果，眞的很強。其次，若您中間有任一過程回答錯了，例如，1045 應該 1B，您回答爲 2B，那串列就會刪光，所以還要補足這一缺憾，避免程式當掉，這請讀者自行練習。

```
Computer guess:0123

input na , nb:0,2
Computer guess:1045

input na , nb:0,1
Computer guess:2356

input na , nb:0,2
Computer guess:3274

input na , nb:1,3
Computer guess:4237

input na , nb:4
```

程式列印

```
a=[]
b=[]
#使用串列列出0000、0001、0002至9999的四位數
for i in range(10):
    for j in range(10):
        for k in range(10):
            for l in range(10):
                a.append(str(i)+str(j)+str(k)+str(l))
print(len(a))
#輸出a串列
for i in range(len(a)):
    print(a[i])
#於a串列逐一蒐集數字不重複者
for i in range(len(a)):
```

```
    flag=True
    c=a[i]
    for j in range(3):
        for k in range(j+1,4):
            if c[j]==c[k]:
                flag=False
                break
    if flag:#沒有重複才蒐集
        b.append(c)
for i in range(len(b)):
    print(b[i])
a=b[:] #串列的複製，已經刪除數字重複者
b=[]#重新清爲空串列
print(len(b))
right=False
#使用while 迴圈，由使用者開始猜測
while (not right):
    #每次都從串列索引0開始與答案比對
    d=a[0]
    print ('Computer guess:%s' %d)
    a1,b1=eval(input('input na , nb:')) #使用者輸入符合a,b的個數
    if a1==4 :
        right=True
        print("Bingo")
        break
    #於a串列的每個數字與輸入答案比對，計算a,b的數量
    for i in range(1,len(a)):
        a2=0
        b2=0
        f=a[i]
        for j in range(4):
            if f[j]==d[j]:
                a2=a2+1
            for k in range(4):
                if f[j]==d[k]:
                    b2=b2+1
        b2=b2-a2
          #將相同a 且相同b的數字蒐集起來，放入b串列
        if a1==a2 and b1==b2 :
            b.append(f)
    #已經完成一階次的比對，b串列蒐集完畢
      a=b[:]  #將b串列取代a串列
    b=[] #b串列清空，重複迴圈，直到答對
```