

變數的宣告、算術運算與輸出入

本章大綱

- 2-1 變數的宣告
- 2-2 算術運算子與算術運算
- 2-3 資料的種類
- 2-4 程式的輸出入
- 2-5 亂數
- 2-6 程式的除錯
- 2-7 發聲

2-1 變數的宣告

程式設計

相信大家都有使用掌上型計算機計算長方形的面積與周長的經驗，那可要按一長串的按鍵，且邊長在算面積時輸入一次，算周長還要再輸入一次。又例如，使用計算機計算一堆數字的平均，也是要不斷鍵入一長串按鍵，且若有按鍵錯誤的疑慮，還要重複按一次。程式設計就是要解決這些輸入的不便。其次，若使用程式設計工具，那資料只要輸入一次，我們還可繼續深入求這些資

料的平均、極大值、不及格個數、排序等統計資料。還有，此一程式還可儲存，下次只要再輸入資料，就可進行此計算。例如，上一章已經完成以上長方形面積計算程式如下：

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

只要指派 a,b 的值一次，就可計算面積與周長，且此程式還可存檔，任何時機都可再拿來做相同的計算。本章即要開始介紹如何編寫這些程式，體驗『程式設計』的神奇功效。

變數的命名

前面我們已經用 a,b 等符號，代表長方形的邊長，這樣就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，我們就稱為變數，電腦會安排記憶體儲存這些『值』。為什麼這些符號稱為變數呢？那是因為這些符號的『值』，在程式執行階段，都可以再任意改變，所以就稱為變數。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。例如，以下是合法的變數名稱。

```
Aa
i
sum
_sum
```

以下是一些非法的變數識別字。

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

- 變數由字母、底線開頭後，僅可由字母、底線及數字組合而成，但不得包含空白。例如，以下是一些合法的變數。

```
a123
AB5566
_a_b
```

以下是一些非法的變數。

```
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
```

- 變數的大小寫均視為不同，例如 Score、score 及 SCORE 皆代表不同的變數。
- 變數不得使用保留字，如 if、for 等。以下是 Python 的保留字：（附註：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，當然不能再使用這些單字。就如同現實社會裡，『電視』、『電腦』已經有特定意義，所以就沒有人取名爲『電視』或『電腦』。

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

- 變數雖然可用中文，但因爲所有程式指令都是英文，若自己使用中文當變數，那還要切換輸入法，真是自找麻煩了。

變數的宣告

大部分的程式語言變數宣告的同時也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，型態與精密度就通通不用管，反正其精密度通通使用最高標準，這樣就可以滿足所有需求了。請鍵入以下程式，並觀察執行結果。

```
a=2 #整數
print(2**100)
b=3.4 #實數
print(b**100)
```

以上都是一個敘述佔用一行，若要將兩個敘述放在同一行，請用分號『;』，例如：

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

但是以下程式，雖然沒有錯，但卻是脫褲子放屁，因為這不是 C 或 Java，每一行不用以分號『;』做結束。

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

變數不用宣告的困境

變數只要放在左邊，就等於宣告，這樣就可在右邊使用，這樣也會有缺點。請鍵入以下程式，並觀察執行結果。

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

以上第二列程式指派運算子『=』右邊的 stutent 拚字錯誤，Python 可以馬上幫忙找出錯誤（請留意解譯器會出現錯誤提示），很快就可以修正如下：

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

得到正確結果。但以下程式您就哭不完了。

```
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 幫您找出錯誤，這次錯在指派運算子左邊，那就麻煩了。因為指派運算子『=』右邊的變數要先宣告，這樣可預防打字拚字錯誤，左邊就是您說了算，沒有幫忙除錯，所以指派運算子左邊的變數要自己特別留意了。又例如，

```
Score=1
score=Score+1
print(Score)
```

也是沒有錯誤信息，但結果也是錯誤，請特別留意運算子左邊的變數名稱不能錯誤。

2-2 算術運算子與算術運算

算術運算子(Arithmetic operators)

小型計算機有加、減、乘、除等鍵，這樣就可計算加、減、乘、除的結果，程式語言要能計算加減乘除，也要有這些算術運算子。下表是 Python 的算術運算子列表：

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

指派運算子 (Assignment operators)

指派運算子的符號為 (=)，其作用為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 `sum=a+b` 是將 `a+b` 的值指派給 `sum`。

```
a=5;b=4;sum=0
sum=a+b
```

上式與數學的等號是不同的，所以不要一直糾結為什麼 0 會等於 9。其次，你是不能將常數放在指派運算子的左邊，例如，

```
8 = a
```

此為一個不合法的敘述，但以下敘述，將常數 8 指派給變數 `a` 是合法的。

```
a = 8
```

算術運算

以下是一些簡單算術運算，請自行鍵入，並觀察結果。

```
a=5;b=4
print(a+b)
```

```
print(a-b)
print(a*b)
print(a/b)
print(a//b)
print(a%b)
print(a**b)
print(9**(1/2))
print(27**(1/3))
print(10**2)
print(10**-2)
print(1/10**2)
```

整數除法與取餘的應用

我們平常的算術運算，對於整數除法與取餘並沒有特別留意。但在程式設計的領域，這兩個運算子有其獨到用途。因為整數除法與取餘可以將一個整數分解為數個數字，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為 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)
```

或者，也有人如以下程式這樣分割。

```
a=152
a1=152//100 #1百位數
a2=(152-a1*100)//10 #十位數
a3=a %10 #個位數
print(a1,a2,a3)
```

數值函式

Python 的算術運算子只有以上 $+$, $-$, $*$, $/$, $//$, $%$, $**$ ，若所需運算沒有運算子，例如，取絕對值運算，那要如何處理呢？答案是要使用數值處理函式。例如，若要執行取絕對值運算，則可使用 `abs()` 函式，程式如下：

```
x=-3
print(abs(x))
```

自我練習

1. 請輸入一個四位數，並將其反向輸出此四個數字。例如，輸入 1234，那可以輸出 4 3 2 1。（數字之間留空白）
2. 同上題，可得到數字和。本例答案是 10。
3. 請輸入一個四位數，並將其分解輸出為四個數字。例如，輸入 1234，那可以輸出 1 2 3 4。（數字之間留空白）
4. 若有一個 5 位數，請問如何分解為兩個一組。例如， $a=25678$ ，那如何分別得到 25,56,67,78 等 4 個數字；其次，要如何得到 2,56,78 三組數字呢。
5. 若有一個 5 位數，那如何得到最左邊與最右邊數字的和。例如，輸入 12345，那結果是 6。
6. 若有一個 5 位數，那如何得到最左邊兩位數與最右邊兩位數字的和。例如，輸入 12345，那結果是 $12+45=57$ 。
7. 若有一個 6 位數，那如何得到最左邊三位數與最右邊三位數字的和。例如，輸入 123456，那結果是 $123+456=579$ 。

補充說明

1. 以上方法不只一種，方法沒有優劣，只要自己想出來的都是最好的。

2. 以上題目的數字長度為已知，這樣程式設計者可依照數字長度決定分割次數，後續介紹迴圈後，就可處理任意長度整數，這樣即為 APCS 考試相近試題。

運算子的優先順序(Precedence)

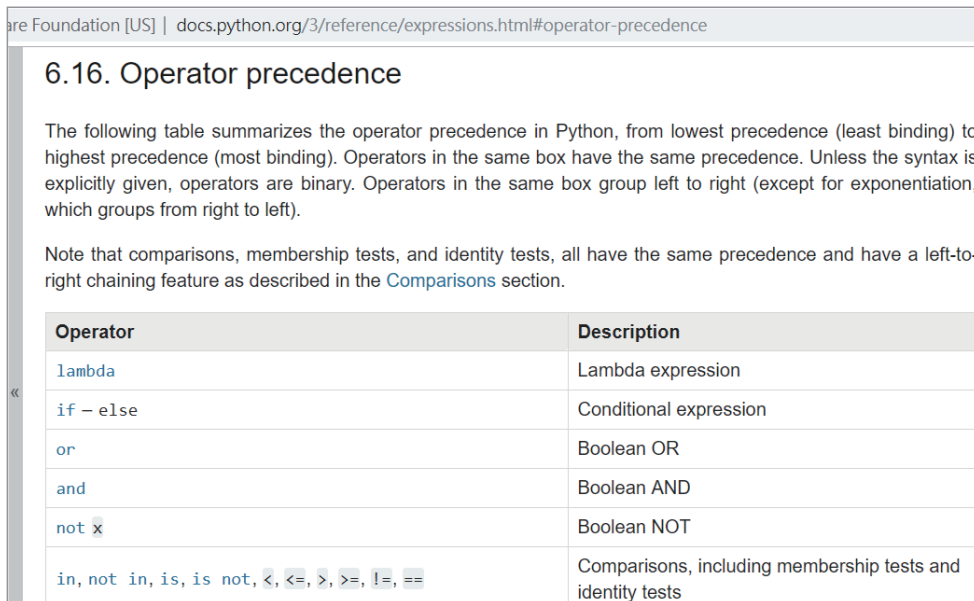
同一敘述，若同時含有多個運算子，此時即需定義運算子的優先順序。例如：

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

在數學裡，我們定義先乘除再加減，程式語言也是相同，必須定義這些運算子的優先順序，請自行鍵入以下程式，並觀察結果。

```
print(3**2)
print(-3**2)
print(4+3**2%2+1)
```

更多的運算子優先順序，請查閱『Language Reference/ 6.16. Operator precedence』，如下圖所示：



Operator	Description
<code>lambda</code>	Lambda expression
<code>if - else</code>	Conditional expression
<code>or</code>	Boolean OR
<code>and</code>	Boolean AND
<code>not x</code>	Boolean NOT
<code>in, not in, is, is not, <, <=, >, >=, !=, ==</code>	Comparisons, including membership tests and identity tests

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，此時即需定義運算子是『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)))
```

以上式子，a、b、c的結果都是2。請鍵入以下程式，並觀察結果。

```
print(6/2*3%2+1)
a=b=c=3
print(a,b,c)
```

變數同時多重指派

Python 允許使用者同時多重指派。請鍵入以下程式，並觀察結果。

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

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，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，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，註解是僅給給人看，電腦不予理會。

```
"""我是註解
我也是註解"""
```

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

```
# 我是註解
```

前者，因為可超過兩列，較適合編寫較長的註解，後者，則僅能寫在同一列。

範例2-2a

請寫一程式，滿足以下條件。

1. 可以指派兩個座標。
2. 計算此兩點座標距離。
3. 輸出此兩點距離。

演算法

1. 已知兩點座標分別是 $(x_1, y_1), (x_2, y_2)$ ，則其距離的公式的數學語言是：

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

程式列印

Python 有次方運算子，所以開根號就是 1/2 次方。其次，因為有運算子優先順序問題，所以請留意括號不能省略。

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

補充說明

1. 本例所有變數內容先使用『指派』，等待學完下一節，再改爲輸入。
2. 因為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

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，例如輸入 3，4，5 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)

資料的抽象化

若有一直線方程式是 $ax+by+c=0$ ，點 $p(m,n)$ 到直線的距離是

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

這件事要由電腦作，那就是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

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

約定此為直線

```
3x+4y+5=0
```

然後以

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

約定此為 $p(1,2)$ ，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，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 ，我們就約定此為 $ax+by+c=0$ ，此稱為資料的抽象化。

範例2-2b

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）。

運算思維

國中解一元二次方程式是先用十字交叉乘法，十字交叉乘法需要一點判斷，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) 設有一元二次方程式如下：

$$ax^2 + bx + c = 0$$

- (2) 資料的抽象化，本例指派 a,b,c 三個整數（本例假設整係數方程式），我們就約定此 a,b,c 代表 $ax^2 + bx + c = 0$ 。

- (3) 令 $d = \sqrt{b^2 - 4ac}$ 。提示：此為數學語言，Python 是 $d=(b*b-4*a*c)**(1/2)$ ，括號、乘號均不能省。

- (4) 則其二解分別為 $x_1 = \frac{-b+d}{2a}$ ， $x_2 = \frac{-b-d}{2a}$ 。（提示：此為數學語言，Python 是 $x_1=(-b+d)/(2*a)$ ，括號、乘號均不能省。）

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

程式列印

```
a=2;b=-7;c=3#變數先指派，可簡化問題。  
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
```

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. 運算思維：近來運算思維廣為被大家討論，我的淺見是，人有人的特性與優勢，電腦有電腦的特性與優勢，所謂運算思維，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。以本範例為例，人類可能用十字交叉法較快，但是電腦是使用公式法較快，此即為電腦的運算思維，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

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

運算思維

1. 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，這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) 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 $a_1, b_1, c_1, a_2, b_2, c_2$ 六個整數。(本例假設整係數方程式)

(3) 令 $d = \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix} = a_1b_2 - a_2b_1$ 。(此稱為行列式表示法)

(4) 則其解分別是 $x = \frac{\begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix}}{d} = (c_1b_2 - c_2b_1) / d$
(x 的地方用 c_1, c_2 代替)

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

(y 的地方用 c_1, c_2 代替)

(5) 例如， $3x+y=5$

$$x-2y=-3$$

則其解為 $x=1$ $y=2$

2. 請寫一個程式，可以將時速 (km/h) 轉為每秒幾公尺 (m/s)。
3. 請寫一個程式，可以將每秒幾公尺 (m/s) 轉為時速 (km/h)。
4. 找零問題。請寫一程式，可以輸入消費金額與付款鈔票面值，計算所需找回的錢，並以最節省鈔票與硬幣數量的方式找錢，例如，輸入
26,1000
可以計算所需找 500,100,50,10,5,1 元的鈔票數量。(APCS105 試題)

範例2-2c

假設某次考試成績資料如下：

55、66、77、88、99

- (1) 請寫一程式儲存以上資料。
- (2) 計算總和。
- (3) 輸出總和與平均。

程式列印

本例 5 筆資料，先用 5 個變數儲存以上資料，所以程式如下：

```
a1=55;a2=66;a3=77;a4=88;a5=99
s=a1+a2+a3+a4+a5
print(s)#385
print(s/5)#77.0
```

補充說明

本例只有 5 筆資料，程式就落落長，那如果有 50 筆，甚至 10000 筆，那不就操死程式設計師，請大家放心，待學習第四、六兩章的迴圈與 list，就會豁然開朗。

自我練習

每一個人請挑一個物理或數學公式，設計一個程式，計算其結果。例如，攝氏轉華氏溫度的公式是『華氏 = 攝氏 *(9/5)+32』，請寫一程式，可以指派攝氏溫度，並輸出華氏溫度。

2-3 資料的種類

前面我們先從生活經驗裡的計算機或計算器談程式設計的功能，計算機的常用功能是幫我們處理整數與實數運算，這些功能電腦程式設計也都能幫我們處理。但是其實我們的生活中所面對的資料還有實數、複數（高中數學）、字元、字串、布林值（True or False）、時間，這些資料型態，傳統計算機就無法處理了，但是電腦的程式設計，竟然通通可以處理，此即為本節重點。

實數

前面我們專注在整數的運算，其實 Python 也可處理實數與複數。請鍵入以下程式，並觀察輸出結果。

```
a=5
b=5.0 #強調這個5,已經精確到小數點1位。
c=3
print(a*c)#15
print(b*c)#15.0
```

字元與字串

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'
```

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，這樣才能進行相關運算。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。)

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

布林值

人類的資料處理常有『對』與『錯』，電腦爲了方便表示與作出決策，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』。

2-4 程式的輸出入

print()

前面我們已經使用 `print()` 指令輸出結果，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
```

以上 `print('aa')` 是輸出字串，`print` 內若是接變數，那就輸出變數的內容，請鍵入以下程式，並觀察輸出結果。

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

```
print(b)
print(c)
```

以上每輸出一個結果就跳列，若您不想讓它跳列，請自行指派 `end` 值。請鍵入以下程式，並觀察輸出結果。

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

`%`

『`%`』是輸出控制字元。以上僅輸出結果 `1,2,3`，若要在輸出結果套用文字說明，例如『`a=1,b=2,c=3`』，則可用『`%`』當作控制字元。請鍵入以下程式，並觀察輸出結果。

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

以上變數 `a,b,c` 會按照順序對應到前面的控制字元『`%`』，『`d`』表示把變數 `a` 當作整數輸出。

`d,f,c,s`

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`』，例如：

```
a=3.4
print("%f" % a)#3.400000
print("%d" % a)#3 以整數輸出
```

字元是『`c`』，例如：

```
a=65
print("%d" % a)#65
print("%c" % a)#A
```

字串是『s』，例如：

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

輸出結果是

```
帥哥Horng 年齡是56
```

格式化輸出

若有多筆資料一起輸出，爲了控制每個欄位的寬度，那還可在d,s,f前將上一個數字，代表欄位寬度。例如：

```
a=5
b=628
print("%6d%6d" % (a,b))#      5      628
```

每欄寬度都是6。又例如：

```
pi=3.14159
print("%6.2f" % pi)#  3.14
```

寬度是6，小數取2位，小數點一位，整數剩3位。以上輸出都是預設靠右，若要左靠，那要加上『-』號。例如：

```
a=5
b=628
print("%-6d%-6d" % (a,b))#5      628
```

以上數值是正數時，沒有輸出正負號，若要強制輸出符號，那就加上『+』，例如：

```
a=1
b=-2
c=3
print('%dx%+dy%+d=0'% (a,b,c))
```

輸出結果是

```
1x-2y+3=0
```

input()

前面我們一直都用指派的方式指派變數值，這樣可以簡化問題，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，『input a:』是輸入提示。

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

其次，Python 將輸入的東西一律視為字串，所以，若您要進行數值運算，那請先用 int() 函數轉為數值。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) #字串相加
a=int(input("input a:"))
b=int(input("input b:"))
print(a+b) #數值相加
```

eval()

前面每次僅輸入一筆資料，善用 eval() 函式可以讓我們同時輸入多筆資料。請鍵入以下程式，並輸入『12,3,4』，並觀察輸出結果，請留意資料與資料之間是用『,』隔開。

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

請留意使用 eval() 函數，其資料型態是數值，所以就可直接進行數值運算。例如：

```
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)#19
```

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:")) #輸入資料請用逗號分隔，例
如， 3,5
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2-4a

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。

輸出結果

```
input a:3
input b:5
面積=15 ,周長=16
```

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))
```


補充說明

本例若用

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

那 a,b 的資料型態是數值，這請讀者自行練習。請鍵入以下程式，並觀察執行結果。

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

自我練習

1. 範例 2-2a、2-2b、2-2c 的資料都是指派，請改為輸入模式。
2. 請分四次輸入 1 個 0 到 9 的整數，並將它合併為 1 個整數。例如，輸入 1，輸入 2，輸入 3，輸入 4，則輸出 1234。
3. 同上習題 2，請先輸入 3 個 0 到 9 的整數，再輸入兩個 0 到 9 的整數，並將其合併為 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。
4. 請輸入 1 個 4 位數，並將其分解輸出如 $a_1*10^3+a_2*10^2+a_3*10^1+a_4$ 。例如，輸入 1234，則輸出 $1*10^3+2*10^2+3*10^1+4$ 。
5. 攝氏轉華氏溫度的公式是華氏 = 攝氏 $*(9/5)+32$ ，請寫一程式，可以輸入攝氏溫度，並輸出華氏溫度，輸出時，欄位寬度設為 5，小數取一位。

範例2-4b

兩點式，兩點可決定一直線，直線方程式為 $(y-y_1)/(x-x_1)=(y_2-y_1)/(x_2-x_1)$ 。請寫一程式，可輸入兩個二維座標，並求出此直線方程式。例如，輸入 (2,1),(4,6)，則直線方程式為 $5x-2y-8=0$ 。

設計步驟

1. 假設輸入為 $(x_1,y_1),(x_2,y_2)$ 。
2. 由兩點式，直線方程式為 $(y-y_1)/(x-x_1)=(y_2-y_1)/(x_2-x_1)$
3. 先計算 $d=y_2-y_1, f=x_2-x_1$
4. 代入 $(y-y_1)/(x-x_1)=d/f$
5. 乘開 $d(x-x_1)-f(y-y_1)=0$
6. 整理上式得方程式為 $dx-fy-dx_1+fy_1=0$

輸出結果

```
5x-2y-8=0
```

程式列印

```
x1=2
y1=1
x2=4
y2=6
d=y2-y1
f=x2-x1
print('%dx%+dy%+d=0'% (d,-f,-d*x1+f*y1))
```

自我練習

1. 點斜式。輸入一點 $p(x_1,y_1)$ 與斜率 m 可造出一條直線 $(y-y_1)/(x-x_1)=m$ ，請寫程式完成。例如，輸入 p 點是 (1,2), $m=3$ ，可得到直線 $3x-y-1=0$ 。
2. 截距式。輸入 x 軸與 y 軸截距，也可造出一條直線，由截距式 $x/a+y/b=1$ ，可得直線方程式為 $bx+ay-ab=0$ 。例如， x 截距是 4, y 截距是 3 直線方程式是 $3x+4y-12=0$ 。

2-5 亂數

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

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

每次用『類別名稱.方法』有點麻煩，可以取一個簡單的物件名。例如：

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

`randint(min,max)`

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

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

大寫字元的 ASCII 編碼是 65 到 90，所以以下程式可隨機產生一個大寫字元。

```
import random
a=random.randint(65,90)
print(a)#輸出數字
print("%c" % a)#輸出對應字元
```

`random` 類別還有一些 `list` 結構的方法，待第六章再說明。

自我練習

請寫一程式，可以得到任意 3 個小寫字元。

2-6 程式的除錯

程式的除錯是所有程式設計者的惡夢，本節就來分享一些程式除錯心得。程式設計常見的錯誤分別是指令與語法拚字錯誤、演算法錯誤等，分別說明如下：

指令拚字錯誤

拚字錯誤是初學者最容易發生的錯誤，但是很多整合編輯器已經幫您將程式解析，給予提示，下圖左是 Spyder 的解析畫面，下圖右是 Python 官版解譯畫面，兩者都能夠將指令、函式、常數解析，給予不同的顏色。其次，Spyder 更強，還會給指令提示，幫忙補冒號、幫忙將對稱的括號給顏色、幫忙將錯誤語法給提示。例如，下圖左 `as` 變數名稱就不行，因為誤用保留字了，且『`as`』前已經出現錯誤信息，由 Spyder 所標示顏色，您會發現『`as`』是保留字。

<pre> D:\junpython\exam\p3.py a1.py x p3.py* x 1 as=2;b=-7;c=3#變數先指派 2 d=(b*b-4*a*c)**(1/2) 3 x1=(-b+d)/(2*a) 4 x2=(-b-d)/(2*a) 5 print(x1)#3.0 6 print(x2)#0.5 7 </pre>	<pre> File Edit Format Run Options Wind as=2;b=-7;c=3#變數先指派 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 </pre>
--	---

演算法錯誤

以上程式都對了，但結果就是不對，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，分別是自己用 `print()` 輸出變數內容，或使用編譯器所提供的 `Debug` 工具，以上兩種方法說明如下：

print()

print 就是在程式中，認為可能發生錯誤的地方輸出變數的內容。
例如，程式原本如下：

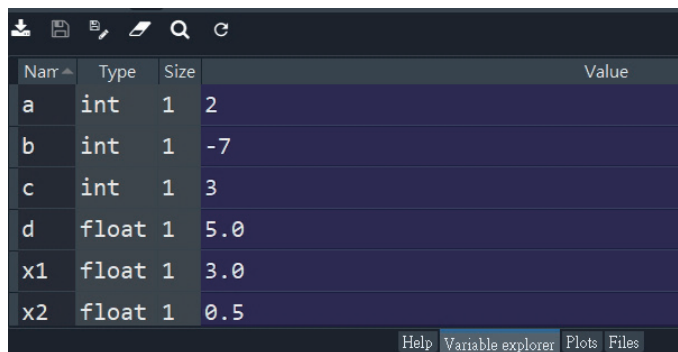
```
a,b,c=eval(input("input a,b,c:"))
d=(b*b-4*a*c)**(1/2)
x1=(-b+d)/(2*a)
x2=(-b-d)/(2*a)
print(x1)#3.0
print(x2)#0.5
```

那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,b,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
```

Variable explorer

也可以直接點選『Variable explorer』，就可觀察所有中間變數的內容，如下圖。




Var	Type	Size	Value
a	int	1	2
b	int	1	-7
c	int	1	3
d	float	1	5.0
x1	float	1	3.0
x2	float	1	0.5

Debug


若有迴圈，如下範例，那變數的執行過程就很重要。例如，若有 while 迴圈程式如下：

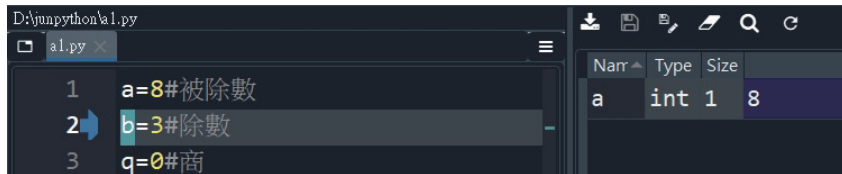
```
a=8#被除數
b=3#除數
q=0#商
while(a>=b):#只要(被除數>除數) 就執行迴圈
    a=a-b    # (被除數)-(除數)
    q=q+1    #商每次遞增1
print(q)# 商數
print(a)#餘數
```

這個程式是，假如沒有除法運算子，那如何得到商與餘數，那就是用減法，只要被除數大於除數，那就減掉除數，每能減一次，商就加一。這在第四章再介紹，現在就請大家用 Debug 工具觀察變數內容。

1. 請先清除 Variable explorer 窗格內容（請留意右邊有橡皮擦圖像）。
2. 點選功能表的『Debug file』畫面如下圖：請留意首行『a=8』前有『大箭號』，這表示這是下一步要執行的指令。

```
7 ▶ a=8#被除數
8   b=3#除數
9   q=0#商
10 ▼ while(a>=b):#只要(被除數>除數) 就執行迴圈
11     a=a-b    # (被除數)-(除數)
12     q=q+1    #商每次遞增1
13     print(q)# 商數
14     print(a)#餘數
```

3. 按一下工具列的『Run current line』圖項，畫面如下圖：請留意『大箭號』跑到下一行，且右邊變數區出現『a=8』。



4. 請繼續按『Run current line』圖項，即可繼續追蹤程式，並觀察變數內容的變化。
5. 點選工具列的『Stop debugging』，即可停止追蹤程式。

2-7 發聲

以下程式可讓電腦發出 Do、Re、Me、Fa…等 8 個音，523、587 就是震盪的頻率，200 的單位是 ms，表示發音的時間。

```
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)
```

請自行下載一個音樂簡譜，讓電腦演奏音樂。例如，下圖是小毛驢簡譜。（摘自 <http://blog.roodo.com/midiland/archives/1679070.html>，特別致謝）以下程式可演奏『我有一隻小毛驢，我從來也不騎』。

本例一分鐘 80 拍，所以 1 拍的時間是， $60000\text{ms}/80$ ，以一個四分之一音符為 1 拍，但實際上每小節是 2 拍，所以一個四分之一音符所佔的時間是 $60000\text{ms}/160$ 。

```
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)
```

請先熟悉以上電腦的發聲方式，下一章介紹決策指令後，就可作出電子琴與電子琴教學機。其次，以上程式待學習迴圈與串列，也可簡化，這樣才能體會後續迴圈與串列的優點。

自我練習

請自行下載簡譜，製作一個 KTV 導唱機。

MEMO