Chapter 4

運算子與運算式

上一章已經介紹資料的分類、資料的儲存，本章則要介紹運算子，有了運算子，就可以開始各種資料的運算了。

4-1 運算子

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。運算子 Arduino 分爲五大類：算術（Arithmetic）運算子、比較（Comparison）運算子、布林（Boolean）運算子、位元操作（Bitwise）運算子及複合（Compound）運算子。每一種運算子都可以再細分爲一元（Unary）運算子與二元（Binary）運算子。一元運算子只需要一個運算元就可以操作，而二元運算子則需要兩個運算元才能夠操作。在以下單元中，我們會檢視各種不同的運算子。除此之外，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時的執行順序。

算術運算子（Arithmetic operators）

算術運算子用來執行一般的數學運算，包括取正負數(+/-)、加(+)、減(-)、乘(*)、除(/)、取餘數(%)等，下表是 Arduino 語言的算術運算子列表：

運算子	定義	優先順序	結合律
=	指派	15	由右至左
+/-	正負號，一元運算子	2	由右至左
*	乘法運算	4	由左至右
/	除法運算（商的型態同被除數）	4	由左至右
%	求餘數（Modulus）	4	由左至右
+/-	加法/減法運算	5	由左至右

『=』符號爲指派運算子，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 sum=a+b 是將 a+b 的值指派給 sum。

```
int sum = 0,a = 3,b = 5;
sum = a + b;
```

上式與數學的等號是不同的，所以不要一直懷疑爲什麼 0 會等於 8。其次，你是不能將常數放在指派運算子的左邊，例如：

```
8 = x ;
```

爲一個不合法的敘述，但以下敘述將常數 8 指派給變數 x 是合法的。

```
x = 8 ;
```

四則運算

以下是一些簡單四則運算：

```
int a=5,b=4;
float a1=5,b1=4;
Serial.println(a+b);//9
```

```
Serial.println(a-b);//1
Serial.println(a*b);//20
Serial.println(a%b);//1
```

整數除法或實數除法

Arduino/C/C++的除法運算，只有被除數與除數的型態均爲整數，才是整數除法，商的型態爲整數；否則即爲實數除法，得到實數商。例如，

```
int x=5, y=4,z;
float xf=5,yf=4;
Serial.println(x/y); // 1 ，被除數與除數的型態均為整數
Serial.println(xf/y);//1.25
Serial.println(x/yf);//1.25
Serial.println(xf/yf);//1.25
```

整數除法與取餘的應用

整數除法與取餘可以將一個整數分解爲數個數字。例如，若要使用七段顯示器顯示 152，那就要將此數字分解爲 1,5,2 三個數字，其方法如下:

```
int a=152;
int a1=152/100;//1 百位數
int a2=(152-a1*100)/10;//5 十位數
int a3=a %10;//個位數
```

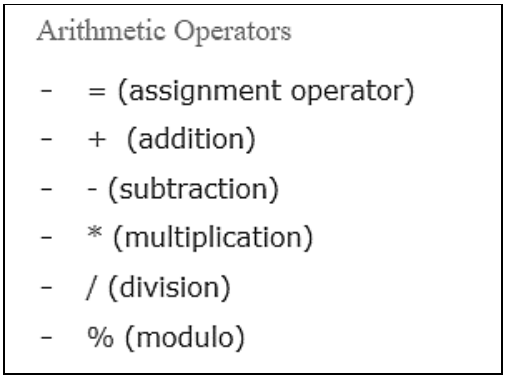
取餘

若要讓一個數字在累加的過程中，保持一定的循環，那也是用取餘。例如，

```
int i;
void loop() {
   i=(i+1) %4;
}
```

那 i 就永遠在 0,1,2,3 循環。

以上算術運算子，Arduino 手冊有詳細介紹，如下圖：



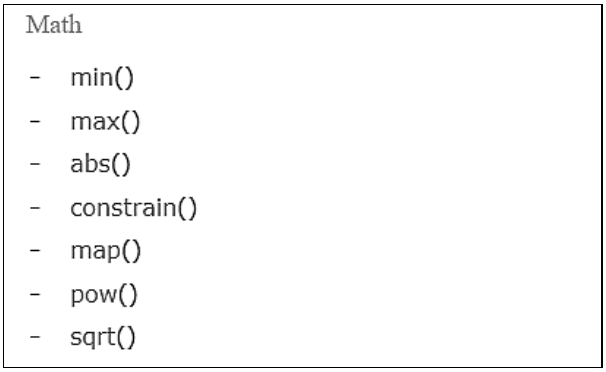
還有，算術運算子只有以上+, -, *, /, %，那如果所需運算沒有運算子，那該怎麼辦？答案是使用數學函式。例如，您想要次方運算，那就使用數學（Math）處理函式中的 pow()函式，如以下敘述，爲計算 a 的平方：

```
Serial.println(pow(a,2));//25.00
```

若要執行開根號運算，則應使用數學處理函式的 sqrt() 函式。如以下敘述是計算 a 的開根號。

```
   Serial.println(sqrt(a));//2.24
```

更多的數學函式請自行檢視參考文件，如下圖。



➲ 自我練習

1. 請將以上運算式於 Arduino 環境鍵入、編譯、執行，觀察執行結果。

程式	執行結果

void setup() {
  Serial.begin(9600);
  int a=5,b=4;
  float a1=5,b1=4;
  Serial.println(a+b);
  Serial.println(a-b);
  Serial.println(a*b);
  Serial.println(a/b);//
  Serial.println(a1/b);//
  Serial.println(a/b1);//
  Serial.println(a%b);//
  Serial.println(pow(a,2));//
  Serial.println(sqrt(a));//
}
void loop() {}	

2. 範例 2-3b 的 LED 電路，有取餘數(%)的應用程式，請複習，此即爲「%」運算子的應用。
3. 若有一個秒數是 12345，那如何分解爲幾時幾分幾秒。例如，本例要能得到 3 時 25 分 45 秒。
4. 若有一個 5 位數，請問如何分解爲兩個一組。例如，a=25678，那如何分別得到 25,56,67,78 等四組數字；其次，要如何得到 2,56,78 字等三組數字呢？

比較運算子（Comparison Operators）

比較運算子又稱爲關係（Relational）運算子，用於資料之間的大小比較，比較的結果可得到 bool 型態的 1 (true)或 0 (false)，下表是 Arduino 語言中的關係運算子符號，這些都和 C/C++相同。例如，以下敘述用來比較 a 與 b 是否相等。

```
if (a == b)
    Serial.println("Equal");
```

運算子	定義	優先順序	結合律
<	小於	7	由左至右
>	大於	7	由左至右
<=	小於等於	7	由左至右
>=	大於等於	7	由左至右
==	等於	8	由左至右
!=	不等於	8	由左至右

例如，

```
int a=5,b=3;
float c=5;
Serial.println(a>b);//1
Serial.println(a>=b);//1
Serial.println(a==b);//0
Serial.println(a!=b);//1
Serial.println(a=!b);//0 小心不要打錯，且沒有錯誤信息
Serial.println(a==c);//0 變數型態要相同才能比較
Serial.println(c>b);//1 變數型態要相同才能比較，但這一運算式勉強正確
Serial.println(a=b);//3 單個等號是指派，請小心
```

➲ 自我練習

1. 請將以上運算式於 Arduino 環境鍵入、編譯、執行，觀察執行結果。

布林運算子（Boolean Operators）

布林運算子又稱邏輯（Logical）運算子。當同一個運算式要同時存在兩個以上的比較運算時，則每兩個比較運算子之間必須使用布林運算子連結。例如，您要找『男生』且『年齡大於 40』，此一選擇就同時含有兩個比較運算式，此時就要運用布林運算子連結。Arduino 布林運算子如下表所示：

運算子	定義	優先順序	結合律
!	布林邏輯 not 運算	2	由右至左
&&	布林邏輯 and 運算	12	由左至右
||	布林邏輯 or 運算	13	由左至右

邏輯 not 是將『true』變『false』,『false』變『true』; 邏輯 and 是兩件事都 true，結果才是 true，其餘都是 false，其眞值表如下：

事件 1	事件 2	結果
true	true	true
true	false	false
false	true	false
false	false	false

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，其眞值表如下：

事件 1	事件 2	結果
true	true	true
true	false	true
false	true	true
false	false	false

例如：

```
int a=5,b=3;
Serial.println(!(a>b));  //0
Serial.println((a>b) && (b>=4));//0
Serial.println((a>b) || (b>=4)); //1
```

➲ 自我練習

1. 若有一數學式，判斷 x 是否滿足 1<x<=6，請使用 Arduino 敘述表示。

```
int x=5;Serial.println((x>1) &&(x<=6));//1
int x=8;Serial.println((x>1) &&(x<=6));//0
```

2. 若有一數學式，判斷 x 是否滿足 x>=3 或 x<1，請使用 Arduino 敘述表示。
3. 若有一數學式，同時判斷六個變數是否滿足 $\frac{a_1}{a_2}=\frac{b_1}{b_2}=\frac{c_1}{c_2}$，請使用 Arduino 敘述表示。那 $\frac{a_1}{a_2}=\frac{b_1}{b_2}\neq\frac{c_1}{c_2}$ 又要如何表示呢？

4. 輸入三角形三邊長 a,b,c，圍成三角形的條件是，任兩邊之和要大於第三邊，那如何以 Aduino 語言表示？（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 Aduino 語言）

位元（Bitwise）運算子

位元（Bitwise）運算子可以分爲兩類：位移（Shift）運算子與布林運算子。位移運算子可以用來把一個整數的各個二進位元向左或是向右移；布林運算子則可以逐位元進行布林運算。下面是 Arduino 語言位元操作運算子的完整列表：

運算子	定義	優先順序	結合律
~	位元 not 運算	2	由右至左
&	位元 and 運算	9	由左至右
^	位元 xor 運算	10	由左至右
|	位元 or 運算	11	由左至右
<<	逐位元向左位移	6	由左至右
>>	逐位元向右位移	6	由左至右

not 運算是將 0 變 1，1 變 0。and 的位元操作眞值表如下，當 a 與 b 同時爲 1，c 才得到 1。

```
c = a and b
```

a	b	c (and)
0	0	0
0	1	0
1	0	0
1	1	1

or 的位元操作眞值表如下，當 a 與 b 有一爲 1，c 就得到 1。

```
c = a or b
```

a	b	c (or)
0	0	0
0	1	1
1	0	1
1	1	1

xor 的位元操作眞值表如下，當 a 與 b 不同時，c 才得到 1。

```
c = a xor b
```

a	b	c (xor)
0	0	0
0	1	1
1	0	1
1	1	0

例如，

```
int a=1,b=0;
Serial.println(~a);// -2
Serial.println(a&b);//0
Serial.println(a|b);//1
Serial.println(a^b);//1
Serial.println(a<<1);//2
Serial.println(a>>1);//0
```

上面~a= not 00000001=11111110，首位元是 1 表示負數，那到底負多少，再取 2 補數（先取 1 補數 00000001，再加 1，00000010），所以是-2，此爲二補數的觀念。

➲ 自我練習

1. 請將以上運算式於 Arduino 環境鍵入、編譯、執行，觀察執行結果。
2. 請使用範例 2-3b 的 LED 電路，鍵入以下程式，觀察執行結果。

```
byte i=1;
void setup() {
  DDRA=B11111111;  //29~22
}
```

```
void loop() {
  PORTA=i;
  i=i<<1;
  delay(1000);
}
```

請留意移位 8 次後，1 就移出去了，若要旋轉，請繼續研讀第四章。

3. 請使用範例 2-3b 的 LED 電路，鍵入以下程式，觀察執行結果。

```
byte i=B11000000;
void setup() {
  DDRA=B11111111;
}
void loop() {
  PORTA=i;
  i=i>>2;
  delay(1000);
}
```

複合指派運算子

結合指派與算術、比較及位元運算的運算子稱爲複合指派運算子。例如，程式設計者常會鍵入 sum=sum+5，爲了簡化此敘述，乃制定此一複合指派運算子+=，所以上述 sum=sum+5，即可寫成 sum + = 5，下表是 Arduino 語言常用的複合指派運算子：

運算子	定義	優先順序	結合律
+=	相加之後再指派內容	15	由右至左
-=	相減之後再指派內容	15	由右至左
*=	相乘之後再指派內容	15	由右至左
/=	相除之後再指派內容	15	由右至左
&=	位元 and 運算之後再指派內容	15	由右至左
|=	位元 or 運算之後再指派內容	15	由右至左
%=	取餘再指派內容	15	由右至左
++	遞增。	1	由右至左
--	遞減	1	由右至左

➲ 遞增（++）及遞減（--）

遞增（例如，a++）及遞減（例如，a--）運算子的功能同

```
a=a+1;
a=a-1;
```

但又分爲前置與後置，前置是運算子在運算元之前，如

```
a=1;++a; Serial.println(a);//2
```

後置是運算子在運算元之後，如

```
a=1;a++; Serial.println(a);//2
```

原則上不論++a 或 a++都是將 a 值加 1 並放回 a，但若是

```
a=1;b=++a; //a=a+1,b=a, a=2  b=2
```

和

```
a=1;b= a++ ;  //b=a, a=a+1, a=2  b=1
```

則其 a 值均會加 1，但 b 值會有差異，前置 b 值會得到加 1 的結果，後者只能得原 a 值。

➲ 自我練習

程式	執行結果

void setup() {
  Serial.begin(9600);
  int a=1;
  int b;
  ++a;
  Serial.println(a);//2
  a=1;
  a++;
  Serial.println(a);//2
  a=1;
  b=++a;	

```
  Serial.print("a=");
  Serial.println(a);//2
  Serial.print("b=");
  Serial.println(b);//2
  a=1;
  a++;
  Serial.println(a);
  a=1;
  b=a++;
  Serial.print("a=");
  Serial.println(a);//2
  Serial.print("b=");
  Serial.println(b);//1
}
void loop() {}
```

➲ 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;
```

由以上各運算子的『優先順序』可知，乘號(*)的優先順序是第 4，而加號(+)則是第 5，指派的優先順序是 15。所以，上式的結果與

```
x=(a+(b*c));
```

的效果相同。同理，

```
x=a>b & b>z;
```

『>』運算子優先順序是 7，而『&&』的優先順序是 12，所以上式同義於

```
x=(a>b) && (b>z);
```

➲ 自我練習

1. 請鍵入、執行以下程式，並寫出執行結果。

程式	執行結果

void setup() {
  Serial.begin(9600);
  int a=4;
  int b=2;
  Serial.println(!a>b);
  Serial.println(a<<2+2);
  Serial.println(!(a>2)||(b>1));
  Serial.println(!((a>2)||(b>1)));
}
void loop() {}	

➲ 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左結合或右結合。例如：

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

連續兩個減號『-』，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

```
x=y=z=2;
```

連續三個指派運算子『=』，指派運算子的結合律是由右至左，所以以上式子同義於

```
 (x=(y=(z=2)));
```

所以，以上敘述，x、y、z 的結果都是 2。

➲ 自我練習

1. 鍵入、執行以下程式，並觀察執行結果。

```
void setup() {
   Serial.begin(9600);
   byte x,y,z;
   x=y=z=2;
   Serial.println(x);
   Serial.println(y);
   Serial.println(z);
}void loop() {}
```

4-2 運算式

運算式（Expression）

任何一個可求得一個值的式子，都稱爲一個運算式，例如，5+3 會傳回一個數值，所以 5+3 是一個運算式，以下是一些合法的運算式，這些運算式通常放在 Serial.print（運算式）、第五章的 if（運算式）、第六章的 for（；運算式；）及 while（運算式）。

```
6          // 傳回 6
4 + 5      // 傳回 9
sum > 3    // 傳回 true or false
a==b       // 傳回 true or false
a + b      // 傳回 8
```

敘述（Statement）

凡是控制敘述執行的順序、對運算式取值或不作任何事，均可稱爲敘述。所有的敘述都要以分號（;）作結束，例如以下式子即是一個敘述。於 Arduino 語言中，若前一個敘述未以分號結束，則錯誤訊息通常出現在下一個敘述的開頭。

```
   sum = sum + 1 ;
```

敘述區塊（Block Statement）或複合敘述（Compound Statement）

在任何可以放上單一敘述的地方，你都能放上敘述區塊，敘述區塊亦稱複合敘述，一個複合敘述是由兩個大括號組合而成，如下所示，但大括號之後不可再加分號。

```
{
    t = a ;
    a = b ;
    b = t ;
}
```

註解（Comments）

適當的程式註解才能增加程式的可讀性，Arduino 語言的註解方式有兩種，分別表示如下：

```
/* 我是註解 */
//我是註解
```

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。例如，

```
/* This program write by Gwosheng
The function is calculate the area of triangle */
```

還有，同一列中，雙斜線『(//)』後面的也視爲註解，編譯器均不處理。例如，

```
sum = sum + y;      // 將 y 之值加至 sum
```

因爲前者可超過兩列，較適合編寫較長的註解；後者，則僅能寫在同一列。

縮排與空行

除了適當的註解，程式設計應善用縮排與空行，才能提高程式可讀性。因爲有了縮排（請使用 TAB 鍵，而不是自行按空白鍵，因爲空白鍵很費力，且無法完全對齊），程式才有層次感。其次，空行則可使段落更加明顯。所以，本書範例都遵守這些規定，請讀者自行觀察。

程式架構介紹

Arduino 的程式架構如下：大致分爲三大區塊，第一區塊是類別引用區與全域變數宣告區；第二區塊是程式初始化區；第三區塊是程式區。

```
//第一區塊
類別引用區與全域變數宣告區
//第二區塊
void setup() {
  // put your setup code here, to run once:
}
//第三區塊
void loop() {
  // put your main code here, to run repeatedly:
}
```

例如，

```
#include <LiquidCrystal.h>//引用類別檔 LiquidCrystal.h
LiquidCrystal lcd(7,6,5,4,3,2);//lcd 是全域物件
int a;//a 是全域變數
void setup() {//only one time
  lcd.begin(16,2);
  lcd.write("Hello ");//only write char
}
void loop() {// to run repeatedly
   lcd.setCursor(0,1);//col,row , The index start (0,0)
   lcd.print(millis()/1000);//char,byte,int ,long,string
}
```

4-3 演算法基本概念與實例探討

演算法基本概念

演算法是指電腦程式完成一項工作所需要『步驟』的集合。若從計算的角度，演算法嚴謹定義如下：

由有限(finite)的步驟(step)所構成的集合，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；而步驟的執行必定會終止(terminate)，並產生輸出(output)。

演算法的流程控制

常見的演算法流程控制有三種，分別是自然語言、虛擬碼與流程圖。分別說明如下：

➲ 自然語言

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，範例 4-3d 解一元二次方程式的演算法。

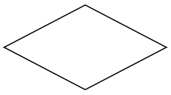
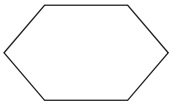
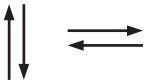
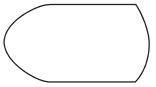
➲ 虛擬碼

在自然語言中，有時嵌入一些程式敘述，那會縮短文字長度，也會讓敘述更加清楚易懂。例如，範例 4-3d 的兩數交換演算法與範例 7-4a 的排序演算法。這些演算法裡面除了有文字敘述外，也包括了簡單的迴圈重複指令，那會使得演算步驟更加簡潔又明瞭。

➲ 流程圖

流程圖(flow chart)是利用各種方塊圖形、線條及箭頭等符號來表達解決問題的步驟及進行的順序，常用的流程符號如下表：

編號	符號	意義
1	開始 結束	起迄符號。 代表流程圖的開始或結束，此符號若是開始，則僅有一出口，若是結束則僅有一入口。

2		輸入與輸出符號。 用來填入輸入與輸出的符號，此符號有一入口與一出口。
3		處理符號。 用來填入處理程序，此符號有一入口與一出口。
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。 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。

使用流程圖表示演算法的有範例 5-1a、範例 6-1a。以下本書的所有範例的演算法，也都依照範例性質，精選最適當的方法來表示。

▶範例 4-3a

請寫一個程式，可以指派長方形的長與寬，並計算周長與面積。

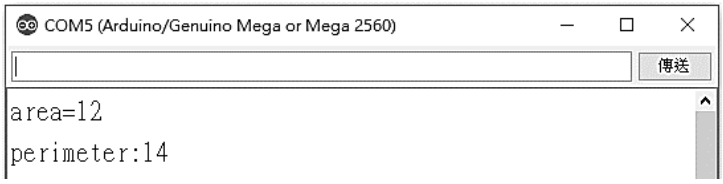
➲ 題目分析

1. 需要兩個數值輸入。請選擇輸入整數還是浮點數，本例選擇整數。
2. 需要兩個輸出。請選擇整數或是浮點數，本例選擇整數。

3. 先使用變數指派方式，指派變數的值，程式如下。這樣可以省略冗長的變數輸入，先專注於演算法的實現。

```
void setup() {
    Serial.begin(9600);
    int a,b,area,perimeter;
    a=3;
    b=4;
    area= a*b;
    perimeter=2*(a+b);
    Serial.print("area=");
    Serial.println(area);
    Serial.print("perimeter:");
    Serial.println(perimeter);
}void loop() {}
```

4. 以上程式執行結果如下：

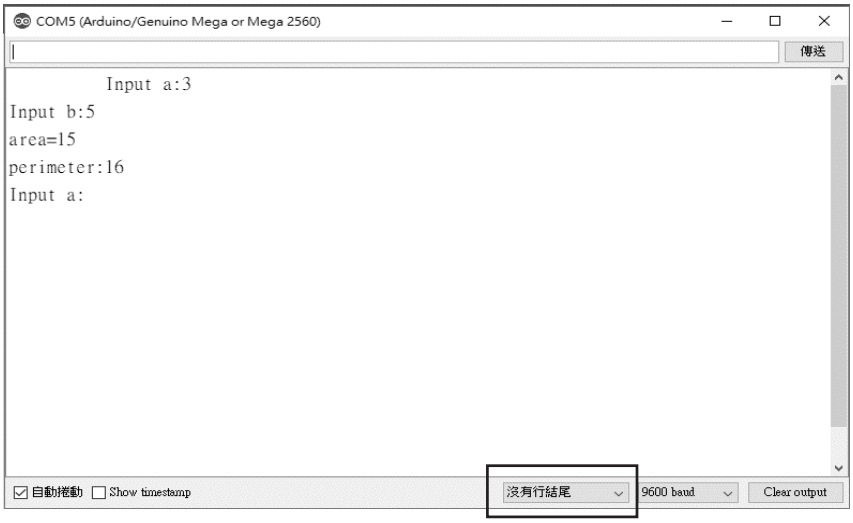


5. 使用電腦的鍵盤輸入變數的數值，程式如下：

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    int a,b,area,perimeter;
    Serial.print("Input a:");
    while(Serial.available() ==0) {}//等待使用者輸入資料
    a=Serial.parseInt();
    Serial.println(a);
    Serial.print("Input b:");
    while(Serial.available() ==0) {}//wait for user's input data
    b=Serial.parseInt();
    Serial.println(b);
    area= a*b;
```

```
  perimeter=2*(a+b);
  Serial.print("area=");
  Serial.println(area);
  Serial.print("perimeter:");
  Serial.println(perimeter);
}
```

6. 以上程式執行結果如下：(請將輸入方式點選『沒有行結尾』，如下圖，不然無法輸入第二個變數)



➲ 補充說明

Serial 物件的所有輸入函式 read()、readString()、parseInt()，都不會刻意等待使用者輸入完畢才執行下一敘述，所以要利用 available()方法，自行撰寫 while 迴圈，等待使用者輸入，只要還沒輸入資料，available()就傳回 0，所以只要 available()==0 這件事情 true，就要繼續執行 while(){}迴圈，這樣就達到繼續偵測與等待使用者是否有輸入資料。其次，此處大括號{}一定不可漏掉，漏掉就錯了，關於 while 迴圈請看第六章。

➲ 自我練習

1. 指派長方體的長、寬、高，計算其表面積與體積。
2. 使用電腦的鍵盤輸入長方體的長、寬、高，計算其表面積與體積。

▶範例 4-3b

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。

➲ 演算法則

1. 已知兩點座標，此兩點座標(x_1,y_1)、(x_2,y_2)，則其距離 d 的數學語言如下：

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

➲ 程式列印

```
void setup() {
   Serial.begin(9600);
   int x1=3,y1=0,x2=0,y2=4;
   float d=sqrt(pow((x1-x2),2)+pow((y1-y2),2));
   Serial.println(d);
}void loop() {}
```

➲ 補充說明

1. 算術運算子只有+,-,*,/,%（取餘數），其餘算術運算都要使用函數，本例pow()是次方，sqrt()是開根號。

➲ 自我練習

1. 同上範例，但可用序列埠視窗輸入兩點座標。
2. 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）
3. 同上題，但使用序列埠視窗輸入變數。

▶範例 4-3c

解一元二次方程式。寫一個程式，可以指派的方式輸入一個一元二次方程式的係數，並求其解（本例假設所輸入的方程式恰有二解）。

➲ 演算法則

解一元二次方程式演算法有循序代入法、配方法與公式法，循序代入法於第六章介紹；配方法判斷較多、計算較少，較適合人腦，待學完第五章的決策，請自己練習；公式法則純粹計算，較適合電腦運算，演算法如下：。

(1) 設有一元二次方程式如下：

$ax^2 + bx + c = 0$

(2) 輸入 a, b, c 三個整數。（本例假設整係數方程式，且用指派的方式輸入變數值）

(3) 令 $d = \sqrt{b^2 - 4ac}$。提示：此爲數學語言，Arduino 語言是 d = sqrt(b*b-4*a*c)，括號、乘號均不能省略。）

(4) 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，Arduino 語言是 x1=(-b+d)/(2*a)，括號、乘號均不能省。）

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

➲ 程式列印

```
void setup() {
    Serial.begin(9600);
    int a=2,b=-7,c=3;
    float d,x1,x2;
    d=sqrt(b*b-4*a*c);
    x1=(-b+d)/(2*a);//The parentheses can not to leave out
    x2=(-b-d)/(2*a);
    Serial.print("x1=");Serial.println(x1);
    Serial.print("x2=");Serial.println(x2);
}void loop() {}
```

➲ 自我練習

1. 同上範例，但改爲可用序列埠視窗輸入係數 a,b,c。
2. 解二元一次方程式。寫一個程式，可以指派一個二元一次聯立方程式的係數，並求其解（本例假設所輸入的方程式恰有一解）。

   提示：

   解二元一次方程式的方法有代入消去法、加減消去法、或本例的公式法。前面兩個方法是判斷比較多、計算比較少，比較適合人腦運算；本例的公式法則是純計算，這就非常適合使用電腦來運算，以下是其演算法（此稱克拉瑪公式）。

   (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 a1, b1, c1, a2, b2, c2 等六個整數係數。（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

▶範例 4-3d

請寫一程式，滿足以下條件。

1. 可以使用兩個變數指派兩個數值。
2. 交換此兩個變數的數值。
3. 輸出交換的結果。

➲ 演算法則

兩個變數的內容要交換，就如同兩隻手的東西要交換位置。所以假設你有 a,b 兩隻手，各拿一樣東西，那要交換其位置的方法如下：

1. 先找來第 3 隻手 t。
2. 將 a 的東西交給 t 暫存。

```
    t=a;
```

3. 將 b 的東西交給 a。

```
    a=b;
```

4. 將 t 的東西交給 b，而完成兩隻手東西的交換。

```
    b=t;
```

5. 其次，若未先找來第三之手 t，電腦並沒有這個能力，同時拋出雙手東西，再同時接住另一手的東西，如以下敘述：

```
    a=b;
    b=a;
```

➲ 程式列印

```
void setup() {
   Serial.begin(9600);
   int a=3,b=4;
   int t;
   t=a;
   a=b;
   b=t;
   Serial.print("a=");
   Serial.println(a);
   Serial.print("b=");
   Serial.println(b);
}void loop() {}
```

➲ 自我練習

1. 輸入三個數，並將 1 交給 2，2 交給 3，3 交給 1，並輸出。

▶範例 4-3e

請寫一程式，可以讓範例 2-3b 的走馬燈，有每次亮一個燈，且向左旋轉移動的點亮方式。

➲ 演算法則

Arduino 並沒有左旋指令，只有左移指令，所以要先取位元 7，然後再執行左移，最後再將位元 7 放在最右邊，完成左旋。取位元 7 的方法有兩種，分別是除以 B10000000 的整數，或使用 bitRead(a,7)函式，請看以下程式。

➲ 程式列印

```
void setup() {
    DDRA=0xFF;
    PORTA=0;
    //Serial.begin(9600);
    //Serial.print(bit(2));
}
byte a=1;
byte b;
void loop() {
    PORTA=a;
    //two way take the leftmost bit
    b=a / B10000000;//take the leftmost bit
    b=bitRead(a,7);//also take the leftmost bit
    a=a<<1  ;// bitshift left
    a=a+b;
    delay(500);
}
```

➲ 自我練習

1. 同本範例電路，請寫一程式，讓 LED 從最左邊每次亮兩個，依次向右或向左旋轉。

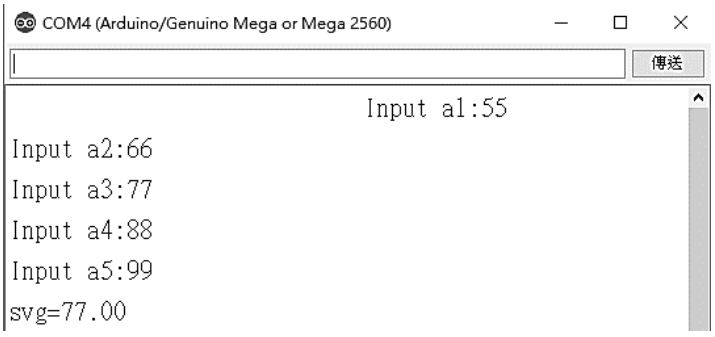
▶範例 4-3f

假設某次考試成績資料如下：

55、66、77、88、99

(1)請寫一程式輸入以上資料。
(2)輸出以上資料。
(3)計算總和。
(4)輸出平均。

➲ 輸出結果



➲ 程式列印

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    int a1,a2,a3,a4,a5;
    float sum=0,avg;
    Serial.print("Input a1:");
    while(Serial.available() ==0) {}
    a1=Serial.parseInt();
    Serial.println(a1);
    sum=sum+a1;
    Serial.print("Input a2:");
    while(Serial.available() ==0) {}
    a2=Serial.parseInt();
    Serial.println(a2);
    sum=sum+a2;
```

```
  Serial.print("Input a3:");
  while(Serial.available() ==0) {}
  a3=Serial.parseInt();
  Serial.println(a3);
  sum=sum+a3;
  Serial.print("Input a4:");
  while(Serial.available() ==0) {}
  a4=Serial.parseInt();
  Serial.println(a4);
  sum=sum+a4;
  Serial.print("Input a5:");
  while(Serial.available() ==0) {}
  a5=Serial.parseInt();
  Serial.println(a5);
  sum=sum+a5;
  avg=sum/5;
  Serial.print("avg=");
  Serial.println(avg);

}
```

➲ 補充說明

1. 本例只有 5 筆資料，程式就非常冗長，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，待學習迴圈與陣列的章節，就會豁然開朗。

➲ 自我練習

1. 假設您有三處溫度感測裝置，請寫一個程式，可以計算其平均溫度，並於序列埠視窗輸出。（補充說明，若使用本書實驗板，雖然一個實驗板僅有一個熱敏電阻，但可以取隔壁同學熱敏電阻的輸出來用，但記得兩塊版子的地線要接在一起。）
2. 請寫一程式，可以統計今天一整天所測得溫度的平均值。

▶範例 4-3g

電子時鐘。

➲ 執行結果

```
7:59:55
7:59:56
7:59:57
7:59:58
```

➲ 演算法則與資料結構

1. 時鐘就是每秒秒數加 1，所以資料結構就是指派秒數的變數爲 t，程式如下：

```
int t=0;
void loop() {
    t++;
    delay(1000);//延遲 1 秒
}
```

2. 考慮資料範圍與型態。1 天 24 小時，每 1 小時 60 分，每 1 分 60 秒，24*60*60=86400，int 的範圍是-32768~32767，已經超過 int 的 32767，所以要取 long。

```
    long t;
```

3. 整數常數的預設值。關於整數的運算，系統預設 int，所以

```
    long t=22*60*60 ;
```

就已經溢位，而且沒有錯誤訊息。這是初學者常常懊惱的地方，反正找不到錯誤，也沒有出現錯誤，但結果就是錯誤。也就是整數數值運算，其運算值的範圍預設爲 int，若要強調以 long 爲範圍，那就要加上『L』，如以下敘述。

```
    long t=22*60*60 L;
```

4. 時鐘就是每秒秒數加 1，秒數到深夜 12 點要歸零，所以程式如下：

```
void loop() {
    t=(t+1) % 86400L;//60*60*24=86400
    delay(1000);//延遲 1 秒
}
```

5. 時、分、秒的表示如下：

```
   byte  h=t/3600;
   byte  m=(t-h*3600L)/60
   byte  s=(t %60);
```

請特別留意，只要整數運算超過 32767 的，那就要加上『L』，強制改爲 long 運算，不然您會哭不完。

6. 輸出格式爲了整齊，所以當數字小於 10，要先加上 0，所以程式如下：

```
   Serial.print(h);
   Serial.print(":");
   if (m<10)
      Serial.print("0");
   Serial.print(m);
   Serial.print(":");
```

以上的 if 指令，請見下一章介紹。

7. 全部程式如下：

```
//long t=22*60*60+59*60+55;//22*60*60 default int, overflow
long t=22*60*60L+59*60+55;//Must add L to force the constant
into a long data format. see the Integer Constants page for
explanation of the 'L'
byte h,m,s;
void setup() {
   Serial.begin(9600);
}
void loop() {
   h=t/3600;
   m=(t-h*3600L)/60;
   s=t %60;
   Serial.print(h);
```

```
    Serial.print(":");
    if (m<10)
       Serial.print("0");
    Serial.print(m);
    Serial.print(":");
    if (s<10)
       Serial.print("0");
    Serial.println(s);
    delay(1000);
    t=(t+1) % (24*60*60L);
}
```

➲ 補充說明

1. 待下一章，再說明如何使用按鈕調整時間。
2. 待介紹陣列與迴圈，就可做出萬年曆。
3. 待介紹四位數七段顯示器或 LCD，就可使用這些元件輸出時間。
4. 關於時鐘程式，有人會先指派爲 h,m,s，這種資料結構的程式如下：

```
byte h=23,m=58,s=50;
void setup() {
  Serial.begin(9600);
}
void loop() {
    s=s+1;
    if (s>=60) {
       s=0;
       m=m+1;
       if (m>=60){
          m=0;
          h=h+1;
          if (h>=24)
             h=0;
       }
    }
    Serial.print(h);
    Serial.print(":");
    if (m<10)
       Serial.print("0");
```

```
    Serial.print(m);
    Serial.print(":");
    if (s<10)
       Serial.print("0");
    Serial.println(s);
    delay(100);
    //delay(1000);
}
```

但這是人類的運算思維，電腦可不用如此大費周章，因爲電腦的長整數可表示範圍是 0～4,294,967,295，這樣就可以表示 0~136 年，程式如下，對於人類經常性的時間處理，可說綽綽有餘。

```
void setup() {
    Serial.begin(9600);
    int a=32767;
    int b=32768;
    Serial.println(a);
    Serial.println(b);//over flow
    unsigned long t=4294967295L;
    unsigned long t1=365L*24L*60L*60L;
    Serial.println(t1);
    Serial.println(t/t1);//136 year
}
void loop() {}
```

所以關於時間的表示，請用一個長整數即可，若要給人看，那再將此 t 轉爲年、月、日、時、分、秒就可以。其次，還有一個問題，我們僅用一個長整數 t 代表 1 個時間點，那關於時間的長度計算與時間前後判斷，就會很容易，若您使用 y,m,d,hour,m,s 代表一個時間點，那關於以上時間長度與前後的計算，將會很複雜，此即爲程式設計者要先瞭解電腦的長處在哪裡、有哪些運算思維，然後愼選資料結構，才能簡化程式的撰寫。

➲ 自我練習

1. 請寫一程式，可以以指派的方式，指派一個時間，程式一執行，可以以倒數的方式，倒數輸出。

2. 同上題，但可以以鍵盤輸入一個時間，例如，20 分鐘，然後倒數計時輸出。
3. 同上題，可用 8 位元指撥開關輸入一個時間，例如，每撥一個，代表 1 分鐘，撥 8 個代表 8 分鐘，然後開始倒數計時。
4. 同上題，將 8 位元指撥開關當作二進位，例如，00000110，代表倒數 6 分鐘。

習　題

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. 請輸入 1 個 4 位數，並將其分解輸出如 $a1*10^3+a2*10^2+a3*10^1+a4$。例如，輸入 1234，則輸出 $1*10^3+2*10^2+3*10^1+4$。
3. 請輸入一個 0 到 15 的整數，並將其轉爲 2 進位輸出。例如，輸入 12，則輸出 $(12)_{10} = (1100)_2$。
4. 請設計一程式，可以計算兩個三維座標的內積運算。例如，P=(a,b,c)，Q=(d,e,f)，則 P 與 Q 的內積爲 ad+be+cf，內積運算結果爲純量。
5. 請設計一程式，可以計算兩個三維座標的外積運算（高中三維空間向量）。例如，P=(a,b,c)，Q=(d,e,f)，則 P 與 Q 的外積爲(bf−ec,cd−af,ae−bd)，外積運算結果仍爲向量。例如，P(x 軸)=(1,0,0)，Q(y 軸)=(0,1,0)，則 P 與 Q 的外積爲 z 軸(0,0,1)，方向即爲右手螺旋定則。
6. 點與直線的距離。請寫一程式，可與輸入一直線與一點，並求出此距離。例如，直線爲 3x+4y=5 與點 A(1,2)的距離爲 1.2。