

程式設計-第五章-函式

洪麗玲



Wireless Access Technologies & Software Engineering

5.2 C語言中的程式模組



- C裡面的模組稱為**函式 (function)**。C程式的撰寫，通常是將程式設計師所寫的新函式，與位於**C標準函式庫 (C standard library)** 中事先寫好的函式結合起來構成一個程式。本章將討論這兩種函式。
- 函式經由**函式呼叫 (function call)** 的方式**調用 (invoked)**。函式指明了欲調用之函式的名稱，並提供所需的資訊(當作**引數, argument**)給受呼叫函式，以執行其工作。與此十分類似的是管理的階層形式。老闆(**呼叫函式或呼叫者, calling function 或 caller**)要求某位員工(**受呼叫的函式, called function**)去執行某項工作，並在工作完成後回報(圖5.1)。

Wireless Access Technologies & Software Engineering

5.3 數學函式庫函式



- 數學函式庫函式讓你能夠執行某些常用的數學運算。在此我們將使用部分的數學函式庫函式來介紹函式的觀念。
- 函式在程式裡調用的方法，通常是先寫函式名稱，接著寫一個左括號，然後跟著寫此函式的**引數 (argument)** (或是由逗號分隔的一個引數列)，最後再寫一個右括號。

函式	說明	範例
<code>sqrt(x)</code>	x 的平方根	<code>sqrt(900.0)</code> is 30.0 <code>sqrt(9.0)</code> is 3.0
<code>exp(x)</code>	指數函式 e^x	<code>exp(1.0)</code> is 2.718282 <code>exp(2.0)</code> is 7.389056
<code>log(x)</code>	x 的自然對數 (底為 e)	<code>log(2.718282)</code> is 1.0 <code>log(7.389056)</code> is 2.0
<code>log10(x)</code>	x 的對數 (底為 10)	<code>log10(1.0)</code> is 0.0 <code>log10(10.0)</code> is 1.0 <code>log10(100.0)</code> is 2.0
<code>fabs(x)</code>	x 的絕對值	<code>fabs(13.5)</code> is 13.5 <code>fabs(0.0)</code> is 0.0 <code>fabs(-13.5)</code> is 13.5
<code>ceil(x)</code>	不小於 x 的最小整數	<code>ceil(9.2)</code> is 10.0 <code>ceil(-9.8)</code> is -9.0
<code>floor(x)</code>	不大於 x 的最大整數	<code>floor(9.2)</code> is 9.0 <code>floor(-9.8)</code> is -10.0
<code>pow(x, y)</code>	x 的 y 次方 (x^y)	<code>pow(2, 7)</code> is 128.0 <code>pow(9, .5)</code> is 3.0
<code>fmod(x, y)</code>	x/y 的浮點餘數	<code>fmod(13.657, 2.333)</code> is 1.992
<code>sin(x)</code>	x 的正弦值 (x 的單位為弧度)	<code>sin(0.0)</code> is 0.0
<code>cos(x)</code>	x 的餘弦值 (x 的單位為弧度)	<code>cos(0.0)</code> is 1.0
<code>tan(x)</code>	x 的正切值 (x 的單位為弧度)	<code>tan(0.0)</code> is 0.0

圖5.2 常用的數學函式庫函式



5.4 函式



- 函式讓你能夠模組化一個程式。所有宣告在函式定義裡的變數都是**區域變數 (local variable)**——只有定義它們的函式才知道這些變數的存在。大多數的函式都有一列**參數 (parameter)**。參數提供了函式間交換資訊的管道。函式的參數也是區域變數。

5.5 函式定義



- 我們介紹過的每個程式都含有一個稱為**main**的函式，它負責呼叫標準函式庫函式來完成程式的工作。請看圖5.3的程式，此程式使用了一個稱為**square**的函式來計算1到10之整數的平方。

```

1 // Fig. 5.3: fig05_03.c
2 // Creating and using a programmer-defined function.
3 #include <stdio.h>
4
5 int square( int y ); // function prototype
6

```

圖5.3 建立並使用程式設計師自訂函式(1/2)



```

7 // function main begins program execution
8 int main( void )
9 {
10     int x; // counter
11
12     // loop 10 times and calculate and output square of x each time
13     for ( x = 1; x <= 10; ++x ) {
14         printf( "%d ", square( x ) ); // function call
15     } // end for
16
17     puts( "" );
18 } // end main
19
20 // square function definition returns the square of its parameter
21 int square( int y ) // y is a copy of the argument to the function
22 {
23     return y * y; // returns the square of y as an int
24 } // end function square

```

```
1 4 9 16 25 36 49 64 81 100
```

圖5.3 建立並使用程式設計師自訂函式(2/2)

一 函式定義的格式如下

```

return-value-type function-name( parameter-list )
{
    definitions
    statements
}

```

- function-name 是任何合法的識別字。**return-value-type** 是指傳回給呼叫者之結果的資料型別。**return-value-type** 為 **void** 的話，表示此函式沒有傳回值。有時候把 **return-value-type**、**function-name** 以及 **parameter-list** 稱為函式標頭 (function header)。



– main的回傳型態

- 注意main有個整數回傳型態，main的回傳值是用來指出程式是否正確地執行。在早期的C語言，如下所示

```
return 0;
```

- 在main的結尾—0代表程式執行成功。在C語言標準裡如果省略了此一敘述，則隱含預設為0—本書範例都是如此。

– maximum函式

- 圖5.4是我們的第二個例子。此程式使用一個程式設計師自訂函式**maximum**，來判斷並傳回三個整數中最大的一個。



```
1 // Fig. 5.4: fig05_04.c
2 // Finding the maximum of three integers.
3 #include <stdio.h>
4
5 int maximum( int x, int y, int z ); // function prototype
6
7 // function main begins program execution
8 int main( void )
9 {
10     int number1; // first integer entered by the user
11     int number2; // second integer entered by the user
12     int number3; // third integer entered by the user
13
14     printf( "%s", "Enter three integers: " );
15     scanf( "%d%d%d", &number1, &number2, &number3 );
16 }
```

圖5.4 尋找三個整數中的最大值(1/3)



```

17 // number1, number2 and number3 are arguments
18 // to the maximum function call
19 printf( "Maximum is: %d\n", maximum( number1, number2, number3 ) );
20 } // end main
21
22 // Function maximum definition
23 // x, y and z are parameters
24 int maximum( int x, int y, int z )
25 {
26     int max = x; // assume x is largest
27
28     if ( y > max ) { // if y is larger than max,
29         max = y; // assign y to max
30     } // end if
31
32     if ( z > max ) { // if z is larger than max,
33         max = z; // assign z to max
34     } // end if

```

圖5.4 尋找三個整數中的最大值(2/3)



```

35
36     return max; // max is largest value
37 } // end function maximum

```

Enter three integers: 22 85 17
Maximum is: 85

Enter three integers: 47 32 14
Maximum is: 47

Enter three integers: 35 8 79
Maximum is: 79

圖5.4 尋找三個整數中的最大值(3/3)

5.6 函式原型：深入探討



- 圖5.4中(第5行)，函式**maximum**的函式原型為

```
int maximum( int x, int y, int z ); // function prototype
```

- 它指明了**maximum**有三個型別為**int**的引數，且傳回型別為**int**的呼叫結果。
請注意，函式原型和**maximum**函式定義的第一行是一樣的。



— 編譯錯誤

- 不合函式原型規定的函式呼叫將造成編譯錯誤。若是函式原型與函式定義不一致的話，也會造成錯誤。舉例來說，如果圖5.4中的函式原型變成了

```
void maximum( int x, int y, int z );
```

- 將會使編譯器產生一個錯誤，因為這個函式原型中的**void**回傳型別，與函式標頭中的**int**回傳型別不一致。



– 強制參數與常用算數轉換規則

- 函式原型的另一項功能是**強制的引數型別轉換 (coercion of arguments)**，亦即強迫引數變成恰當的型別。
- 常用算數轉換規則會自動應用到含有兩種(或更多)資料型別之數值的運算式，也稱為**混合型別運算式 (mixed-type expressions)**，會自動由編譯器處理。對至少包含一個浮點數值的混合型別運算式，常用算數轉換規則為：
 - 如果一個數值為**long double**，則其他的值都轉換成**long double**。
 - 如果一個數值為**double**，則其他的值都轉換成**double**。
 - 如果一個數值為**float**，則其他的值都轉換成**float**。



資料型別	printf 的轉換指定詞	scanf 的轉換指定詞
<i>Floating-point types</i>		
long double	%Lf	%Lf
double	%f	%lf
float	%f	%f
<i>Integer types</i>		
unsigned long long int	%llu	%llu
long long int	%lld	%lld
unsigned long int	%lu	%lu
long int	%ld	%ld
unsigned int	%u	%u
int	%d	%d
unsigned short	%hu	%hu
short	%hd	%hd
char	%c	%c

圖 5.5 昇數資料型別與轉換規則

4.10 邏輯運算子



- C提供了邏輯運算子 (logical operator)，可以將數個簡單條件式組合成一個複雜的條件式。
- 邏輯AND運算子&&
 - 如果我們希望在**兩種**條件都為真的情況下，才執行某項動作，則我們可以用邏輯運算子&&
- 邏輯OR運算子 ||
 - 現在讓我們來看看 || (邏輯OR) 運算子。如果我們希望在任一個或兩種條件皆為真的情況下，才執行某項動作，我們可以運用 || 運算子

練習



- 請將找最小值的程式改寫成函式的方式，並以呼叫函式的做法實作完成
 - 先說明程式要做什麼事，並適當指引要做什麼輸入
 - 收到輸入後要再輸出收到的資料以進行確認
 - 完成後輸出結果並詢問是否要再執行一次
 - 請想清楚那些寫在函式中，那些寫在主程式(main)中
- 其他要求參照課程要求”程式註解”

5.8 標頭



- 每個標準函式庫都有一個相對應的**標頭 (header)**，它含有此函式庫中所有函式的函式原型，以及這些函式所需之各種資料型別和常數的定義。圖5.10依字母的順序列出了可以含括進程式裡的標準函式庫標頭檔。



標準函式庫標頭	說明
<assert.h>	內含一些用來幫助程式偵錯的巨集和資訊。
<ctype.h>	內含一些檢測字元特性及大小寫字元轉換等函式的原型。
<errno.h>	定義了一些用來回報錯誤狀況的巨集。
<float.h>	內含此系統中對浮點數大小的限制。
<limits.h>	內含此系統中對整數大小的限制。
<locale.h>	內含一些能夠使程式區域化的函式原型與資訊。區域化的表示方式讓電腦系統能處理世界各地各種不同的資料 (如日期，時間，金額及大的數目) 表示習慣。
<math.h>	內含數學函式庫的函式原型。

圖5.10 標準函式庫標頭檔(1/2)



標準函式庫標頭	說明
<setjmp.h>	內含改變正常函式呼叫與回傳順序的函式原型。
<signal.h>	內含處理程式執行中各種狀況的函式原型和巨集。
<stdarg.h>	定義一些處理不確定型別及個數之引數列的函式。
<stddef.h>	內含一些在 C 執行運算時所常用到的型別。
<stdio.h>	內含標準輸出 / 入函式庫的函式原型以及所需的資訊。
<stdlib.h>	內含一些數字與文字間轉換，記憶體配置，亂數，及其他公用函式的原型。
<string.h>	內含字串處理函式的原型。
<time.h>	內含處理時間與日期的函式原型。

圖 5.10 標準函式庫標頭檔(2/2)

- 你也可以撰寫自己的標頭檔。程式設計師定義的標頭檔，也應以 **h** 做為副檔名。我們可以用前置處理器命令 **#include "ooo.h"** 含括進程式設計師定義的標頭檔。

- #include 引入標頭檔可分為下列兩種格式：

– #include <xxx.h> (

- **xxx.h** 為 C 編譯器提供的標頭檔，並且存放在編譯器內定的目錄中，使用此種格式，前置處理器會自動到內定目錄中找到標頭檔。

– #include "ooo.h" (

- **ooo.h** 不是編譯器提供的標頭檔，所以程式設計師必須標明該檔案所在目錄，以便前置處理器取得該檔案。

運算式 1	運算式 2	運算式 1 && 運算式 2
0	0	0
0	非零	0
非零	0	0
非零	非零	1

```
if ( gender == 1 && age >= 65 )
    ++seniorFemales;
```

運算式 1	運算式 2	運算式 1 運算式 2
0	0	0
0	非零	1
非零	0	1
非零	非零	1

```
if ( semesterAverage >= 90 || finalExam >= 90 )
    puts( "Student grade is A" );
```

Wireless Access Technologies & Software Engineering



回家作業

- 請撰寫一支程式功能如下：
 - 使用者輸入一個最多7位數的數字
 - 系統輸出該數的相反數字（我們稱倒數）
 - 並算出該輸入與倒數的平均值
 - 可以輸入很多次



Wireless Access Technologies & Software Engineering