程式設計
(Programming)

真理大學 資訊工程系 吳汶涓老師

CH05

函式 (function)

本章綱要

5.1 簡介

- 大部分電腦程式都比前幾章所介紹的程式大很多
  - 無法管理
  - 無法分工

- 各個擊破 (divide and conquer)
  - 以一些較小的單元來建構整個程式
    - 這些小單元稱爲模組 (module)
  - 這些小單元要比整個大程式好管理多了

5.2 C語言中的程式模組

- 函式 (function)
  - C的模組
  - 程式包含：使用者定義的函式 、函式庫函式
    - C標準函式庫提供了包羅萬象的函式，包括數學運算、字串處理、字元處理、輸入／輸出等等。
    - 例如：**printf( ), scanf( ), pow( )** 函式

```
/* calculate new amount for specified year */
amount = principal * pow( 1.0 + rate, year );
```

軟體工程的觀點 5.1

避免「重新發明輪子」。儘量使用 ANSI C 標準函式庫的函式，而不要自己撰寫功能相同的新函式。這將可以減少程式發展的時間。

函式呼叫 (function call)

- 引用函式 (invoking function)
  - 提供函式名稱、引數(資料)
  - 函式是用來執行運算或操作
  - 函式可以傳回運算結果

```
/* calculate new amoun   or specified year */
amount = principal * pow( 1.0 + rate, year );
```

- 函式呼叫的比喻
  - 老闆要求員工去執行某項工作
  - 員工得到資訊 → 執行任務→ **回報結果(return)**
  - 老闆不曉得執行的細節

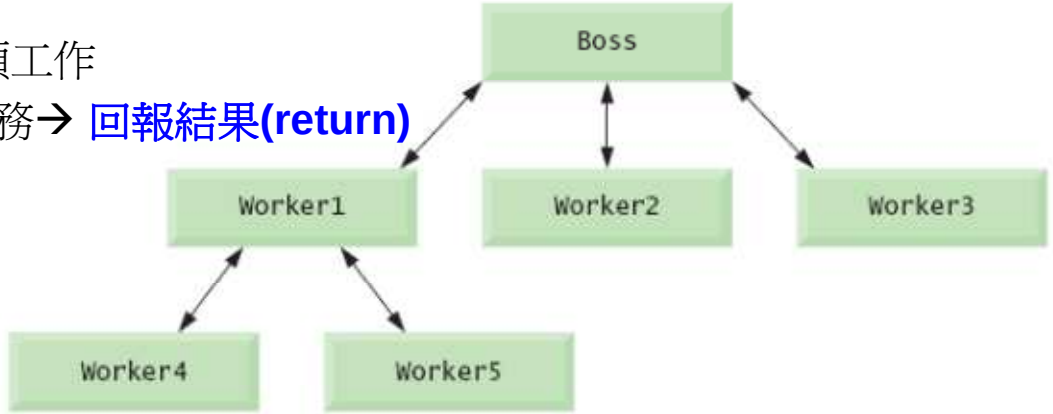
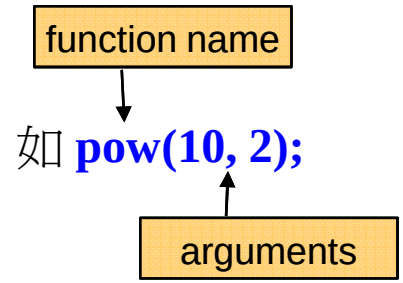


圖 5.1　階層式的老闆函式／員工函式關係圖

5.3 數學函式庫函式

- 數學函式庫函式：
  - 提供常用的數學運算
  - **#include <math.h>**
- 呼叫函式的格式
  - **FunctionName(argument);**
    - 若有多個引數，使用逗號隔開，如 **pow(10, 2);**



  - printf(“%.2f”, sqrt(900.0));
    - 呼叫 sqrt 函式，回傳引數的平方根
    - 所有數學函式**傳回值的型態都是 double**
  - 引數可以是**常數、變數或運算式**

函式	說明	範例
sqrt(x)	x的平方根	sqrt(900.0) is 30.0 sqrt(9.0) is 3.0
exp(x)	指數函式 e^x	exp(1.0) is 2.718282 exp(2.0) is 7.389056
log(x)	x的自然對數 (底爲 e)	log(2.718282) is 1.0 log(7.389056) is 2.0
log10(x)	x的對數 (底爲 10)	log10(1.0) is 0.0 log10(10.0) is 1.0 log10(100.0) is 2.0
fabs(x)	x的絕對值	fabs(5.0) is 5.0 fabs(0.0) is 0.0 fabs(-5.0) is 5.0
ceil(x)	不小於 x的最小整數	ceil(9.2) is 10.0 ceil(-9.8) is -9.0
floor(x)	不大於 x的最大整數	floor(9.2) is 9.0 floor(-9.8) is -10.0
pow(x, y)	x的 y次方 (x^y)	pow(2, 7) is 128.0 pow(9, .5) is 3.0
fmod(x, y)	x/y的浮點餘數	fmod(13.657, 2.333) is 1.992
sin(x)	x的正弦值 (x的單位爲弧度)	sin(0.0) is 0.0
cos(x)	x的餘弦值 (x的單位爲弧度)	cos(0.0) is 1.0
tan(x)	x的正切值 (x的單位爲弧度)	tan(0.0) is 0.0

課本pp. 5-5

圖 5.2 常用的數學函式庫函式

練習

- 課本pp.5-52, Ex 5.8
  請求出下列每項敘述式執行後，x的值為何。
  - x = fabs(7.5); ← ans: 7.5
  - x = floor(7.5); ← ans: 7.0
  - x = fabs(0.0); ← ans: 0.0
  - x = ceil(0.0);
  - x = fabs(-6.4);
  - x = ceil(-6.4);
  - x = ceil(-fabs(-8 + floor(-5.5)));

5.4 函式

- 函式
  - 模組化一個程式
  - 宣告在函式中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
- 使用函式的好處
  - 各個擊破
    - 程式更好管理
  - 軟體可以重複使用性 (software reusability)
    - 抽象化: 隱藏內部細節
  - 減少重複的程式碼

軟體工程的觀點 5.2

main 通常被寫成一群執行程式工作的函式呼叫

```
int main()
{
    processA();
    processB();
    processC();
    ...
}
```

軟體工程的觀點 5.5

一個函式其大小不應超過一頁。最好是不要超過半頁。小型函式可以提升軟體再使用性。

軟體工程的觀點 5.3

每一個函式限制在只執行一項定義明確的工作，函式名稱能充反映出它所執行的工作。

軟體工程的觀點 5.4

將工作切割爲數個較小的函式，又稱爲函式分解 (decomposition)。

課本pp. 5-8

5.5 函式定義

- 計算1~10整數的平方
  - 函式自行定義

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

```c
/* Fig. 5.3: fig05_03.c */

#include <stdio.h>
int square( int y ); /* function prototype */

int main( void )
{
   int x;

   for ( x = 1; x <= 10; x++ ) {
      printf( "%d  ", square( x ) );
   }

   printf( "\n" );
   return 0;
}

/* square function definition returns square */
int square( int y )
{
   return y * y;
}
```

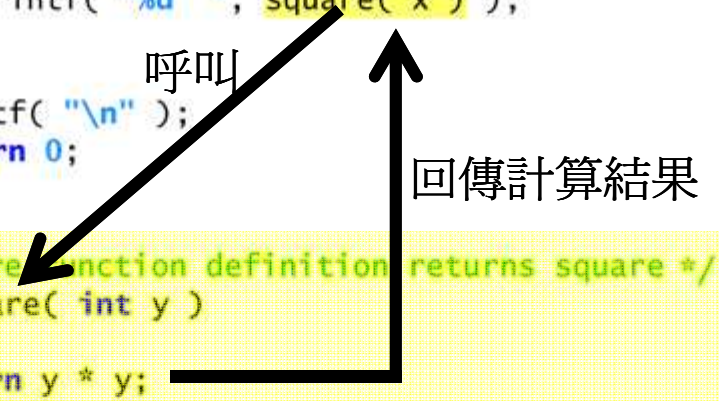


圖 5.3　使用程式設計師自訂函式

函式定義的格式

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

- function-name: 任何合法識別字
- return-value-type: 計算結果的資料型態
  - **void 是用來表示函式沒有回傳值**
- parameter-list: 宣告參數
  - 每個參數的型別要明確指出

```
/* square function definition returns square */
int square( int y )
{
   return y * y;
}
```

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

- definitions & statements: (函式本體、函式區塊)
  - 區塊中可以宣告變數
  - 函式不能定義在另一個函式之內
  - 本體最後一行需傳回執行的控制權
    - 沒有回傳值
      return;
    - 有回傳值
      return *expression or values;*

```c
/* Fig. 5.3: fig05_03.c */

#include <stdio.h>
int square( int y ); /* function prototype */

int main( void )
{
   int x;

   for ( x = 1; x <= 10; x++ ) {
      printf( "%d  ", square( x ) );
   }

   printf( "\n" );
   return 0;
}

/* square function definition returns square */
int square( int y )
{
   return y * y;
}
```

函式原型:
表示之後會定義此函式

呼叫square函式

函式定義

圖 5.3　使用程式設計師自訂函式

良好的程式設計習慣 5.2

請在函式與函式的定義之間加一行空行，增進程式的可讀性。

良好的程式設計習慣 5.4

請選用有意義的函式名稱和參數名稱，使程式更具可讀性。

常見的程式設計錯誤 5.1

忘了從一個必須傳回值的函式傳回數值，會造成非預期的錯誤。
從一個回傳型別宣告爲 void 的函式裡傳回數值，將導致編譯錯誤。

常見的程式設計錯誤 5.3

將具有相同型別的參數宣告成如 `double x, y`，會造成編譯錯誤。

常見的程式設計錯誤 5.5

將函式的參數在函式內重新定義爲區域變數，是一種編譯錯誤。

- Example: 尋找三個整數中的最大值 課本pp. 5-10

```
1  /* Fig. 5.4: fig05_04.c  */
2
3  #include <stdio.h>
5  int maximum( int x, int y, int z ); /* function prototype */
6
8  int main( void )
9  {
10    int number1;
11    int number2;
12    int number3;
13
14    printf( "Enter three integers: " );
15    scanf( "%d%d%d", &number1, &number2, &number3 );
16
19    printf( "Maximum is: %d\n", maximum( number1, number2, number3 ) );
20    return 0;
21 }
22
```

函式原型

呼叫函式

圖 5.4 尋找三個整數中的最大值

```
23  /* Function maximum definition */
24
25  int maximum( int x, int y, int z )
26  {
27     int max = x;
28
29     if ( y > max ) {
30        max = y;
31     }
32
33     if ( z > max ) {
34        max = z;
35     }
36
37     return max; /* max is largest value */
38  }
```

函式定義

圖 5.4　尋找三個整數中的最大值

```
Enter three integers: 22 85 17
Maximum is: 85
```

```
Enter three integers: 85 22 17
Maximum is: 85
```

```
Enter three integers: 22 17 85
Maximum is: 85
```

練習

- 改寫上面的程式，使能計算三個整數的最小值。

- 請撰寫一個程式輸入一連串的整數，並將他們一次一個傳給函式**even**，它將利用模數運算子來判斷傳進來的整數是否爲偶數。此函式應接收一個整數引數，並在整數爲偶數時傳回1，否則傳回0。
  pp.5-54, ex 5.18

5.6 函式原型 (function prototype)

- 函式原型：用來驗證函式
  - 函式名稱
  - 參數列
  - 回傳型別

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

- 當函式定義出現在使用函式的位置之後，才需要函式原型

```
19  printf("Maximum is: %d\n",maximum(number1, number2,number3 ));
    ...
25  int maximum( int x, int y, int z )
26  {
      ...
38  }
```

- 提昇規則
  - C語言會強制進行恰當的型別轉換，如 sqrt(4);
  - 將數值轉換成較低的型別，通常會得到不正確的值
    如圖5.3的 square(4.5) 將會回傳16，而非20.25

高

低

資料型別	printf 的轉換指定詞	scanf 的轉換指定詞
long double	%Lf	%Lf
double	%f	%lf
float	%f	%f
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　資料型別的提升階層

課本pp. 5-12

練習

- 請修改先前的進位轉換作業，將每個功能分別寫成函式(function)，最多會有3個函式。
  功能有:
  1. Dec -> Bin
  2. Bin -> Dec
  3. Dec -> Oct
  4. Bin -> Oct

練習

- 撰寫一個程式，此程式有兩個功能如下

(1) 指數計算:

函式 integerPower(base, exponent) 能計算出 $base^{exponent}$的值並回傳。兩個參數皆爲整數。
函式中不能用任何數學函式庫來做次方的計算。

(2) 倍數計算:

函式 multiple(v1, v2) 能判斷出v2是否爲v1的倍數並回傳true 或 false。兩個參數皆爲整數。

課本pp. 5-54, Ex. 5.16, 5.17