程式設計與實習(一)

BY 孫茂勛

EMAIL:JOHN85051232@GMAIL.COM

Function

函式(Function)

- C語言寫程式的模組單位。
- 將程式切割成不同功能模組的組合。
- 使程式碼容易閱讀、維護。
- 架構包含：函式型態、函式名稱、參數、程式碼。
- 例如：給一台機器（函式型態、名稱）電力（參數），他就可以執行工作（功能）。

Function

```c
#include <stdio.h>
#include <stdlib.h>

void printf_star(int star_number)
{
    for(int i = star_number ; i > 0 ; i--)
    {
        printf("*\n");
    }
}

int main()
{
    printf_star(5);//可以直接放入參數
    int n = 10;
    printf_star(n);//也可以放入變數
    system("PAUSE");
    return 0;
}
```

宣告方式:

函式型態 函數名稱(可能會用到的參數)

{

...

}

當編譯器讀到main中的printf_star時，他會去尋找程式碼中有這一個名字的函數開始執行，執行完才會回到main中繼續執行下一行指令。

Function

```c
#include <stdio.h>
#include <stdlib.h>

void printf_star(int star_number);

int main()
{
    printf_star(5);//可以直接放入參數
    int n = 10;
    printf_star(n);//也可以放入變數
    system("PAUSE");
    return 0;
}
void printf_star(int star_number)
{
    for(int i = star_number ; i > 0 ; i--)
    {
        printf("*\n");
    }
}
```

函式宣告在main()後面時，由於編譯器是由上往下讀，所以會看不懂main中的printf_star是什麼。

所以要在一開始先加入一個宣告，告訴編譯器這是我們自定義的函式。

練習

用*印出正三角形、倒三角形

```
   *                 ****
  **                  ***
 ***                   **
****                    *
```

Function

宣告方式:

函式型態 函數名稱(可能會用到的參數)
{
…
回傳值
}

不同類型的函式以及回傳值：

函式類型	回傳值
void	不用回傳
int	return int
float	return float
char	return char

Function

```c
int add_two(int input1,int input2)//一個將兩數相加的函式，參數是兩個整數
{
    return input1 + input2;//回傳一個整數，他是兩數相加的結果
}
int main()
{
    int input1 = 0,input2 = 0;
    scanf("%d %d",&input1,&input2);
    printf("%d + %d = %d\n",input1,input

    system("PAUSE");
    return 0;
}
```

```
3
99
3 + 99 = 102
請按任意鍵繼續 . . .
```

視域、生命週期(Scope)

視域：看的到該變數出現的地方
生命週期：該變數何時會消滅

變數類型	視域	生命週期
全域變數	整個程式	整個程式
區域變數	只在該函式、方法內	只在該函式、方法內
Static靜態變數	函式、方法內	整個程式

視域、生命週期(Scope)

```c
#include <stdio.h>
#include <stdlib.h>

int global_variable = 20;//寫在最外面的是區域變數
void test()
{
    printf("%d",local_variable);//無法判斷這個是什麼?
}
int main()
{
    int local_variable = 10;//位於main內的區域變數
    test();
    system("PAUSE");
    return 0;
}
```

視域、生命週期(Scope)

比較看看兩者的差別?

區域變數當該區塊執行結束後便會消失。
所以資料不會保存。

```c
#include <stdio.h>
#include <stdlib.h>

void test()
{
    int local_variable = 10;
    printf("%d\n",local_variable);
    local_variable = local_variable + 1;
}
int main()
{
    for(int i = 0 ; i < 10 ; ++i)
    {
        test();
    }
    system("PAUSE");
    return 0;
}
```

Static的變數只會生成一次，之後直到程式結束前都會一直存在。
所以資料可以一直保存。

```c
#include <stdio.h>
#include <stdlib.h>

void test()
{
    static int local_variable = 10;
    printf("%d\n",local_variable);
    local_variable = local_variable + 1;
}
int main()
{
    for(int i = 0 ; i < 10 ; ++i)
    {
        test();
    }
    system("PAUSE");
    return 0;
}
```

視域、生命週期(Scope)

當變數名稱重複時，視域小的優先。

```
#include <stdio.h>
#include <stdlib.h>
int c = 10;
int main()
{
    int c = 20;
    printf("%d\n",c);
    system("PAUSE");
    return 0;
}
```

```
20
請按任意鍵繼續 . . .
```

Define

符號常數(Define)

- 為了讀寫及修改方便經常將常數以符號常數代替。
- 通常放置在開頭。
- 用法：#define 常數名稱 常數

Define

```
4  #define PI 3.14159
5
6  float area(float);
7  int main(){
8
9      float radius = 2.3;
10
11     printf("半徑為%.1f 面積為 : %.2f\n" , radius , area(radius));
12
13     system("pause");
14     return 0;
15 }
16
17 float area(float radius){
18
19         float out = 0;
20
21         out = radius*radius*PI;
22
23         return out;
24
25 }
```

Random

rand()

- #include <stdlib.h>
- 傳回一個介於 0 到 RAND_MAX 之間的一個整數值

srand()

- #include <time.h>

Random

```c
#include <stdio.h>
#include <stdlib.h>



int main(){


    int i = 0;
    i = rand() % 6;

    printf("i = %d\n" , i);

    system("pause");
    return 0;
}
```

Random

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>



int main(){

    srand(time(NULL));

    printf("亂數1 = %d\n" , rand());
    printf("亂數2 = %d\n" , rand());

    int a = rand()%6+1;
    printf("a = %d\n" , a);

    system("pause");
    return 0;
}

```

Recursion

遞迴(Recursion)

- 在函式中不斷地呼叫自己，直到達成終止條件才會結束。
- 需要額外堆疊空間儲存。
- 程式碼通常都很短，但通常都不好寫。
- 常看到的：階乘、費式數列、GCD、河內塔...

Recursion

```
5  long factirial(int number);//階層
6  int main(void) {
7
8
9     int num = 0;
10    scanf("%d" , &num);
11
12    printf("%d! = %ld\n" ,num ,  factirial(num));
13   system("pause");
14   return 0;
15 }
16
17 long factirial(int number){
18
19       if(number <= 1){
20                 return 1;
21       }
22       else{
23            return (number * factirial(number-1));
24       }
25
26 }
```

Recursion

```c
#include <stdio.h>
#include <stdlib.h>

int fib(int num)
{
    if(num <= 1)
    {
        return num;
    }
    else
    {
        return fib(num-1) + fib(num-2);
    }
}
int main()
{
    for(int i = 1 ; i <= 10 ; ++i)
    {
        printf("%d ",fib(i));
    }
    return 0;
}
```

Recursion

```
3
4  int gcd(int, int); //最大公因數
5  int main(void) {
6      int m = 0;
7      int n = 0;
8
9      printf("輸入兩數(num1 num2)：");
10     scanf("%d %d", &m, &n);
11
12     printf("GCD: %d\n", gcd(m, n));
13
14     system("pause");
15     return 0;
16 }
17 int gcd(int m, int n) {
18     if(m % n == 0) {
19         return n;
20     }
21     else {
22         return gcd(n, m % n);
23     }
24 }
25
```