

# Summary of Lesson 5

## Function

- . function is a group of task like a **patcher**
- . normally it need 1 or more input
- . normally it **return** a result of process, otherwise it's called **void function**
- . input for functions are called **parameters** from the view of a invoked function
- . input for functions are called **arguments** from the view of original function
- . output is called **return value**
- . function is a **black box**, a function which invokes another function can not comprehend the process of invoking function
- . parameters and arguments are **copy**, in principle, you can not change the value in the original function directly from a invoked function (but there is an exception = pointer)
- . the **scope** of variable is limited by **function** but you can refer all variables declared as **external variables** from everywhere
- . you have to declare **extern** before you use it in a function
- . all datas used in function will be removed after the process, but if you declare a variable with keyword **static**, the data remains in the variable
- . you can recursively call a function, it's called **recursive call**. this is the third way to make a loop in the program

---

### lesson5\_1.c

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

```
void japan(void);  
void spain(void);  
void germany(void);
```

```
int main (void) {  
    puts("I am in basel");  
    japan();  
    spain();  
    germany();  
    puts("I came back to basel");  
    return 0;  
}
```

```
void japan(void)  
{  
    puts("I am in japan.");  
}
```

```
void spain(void)  
{  
    puts("I am in spain.");  
}
```

```
}
```

```
void germany(void)
{
    puts("I am in germany.");
}
```

---

### **lesson5\_2.c**

```
#include <stdio.h>
void circleArea(float radius);

int main (void) {
    float radius;
    printf("input radius:");
    scanf("%f",&radius);
    circleArea(radius);

    return 0;
}

void circleArea(float radius)
{
    printf("%f", radius * radius * 3.14159265);
}
```

---

### **lesson5\_3.c**

```
#include <stdio.h>
float circleArea(float radius);

int main (void) {
    float radius;
    float area;

    printf("input radius:");
    scanf("%f",&radius);
    area = circleArea(radius);
    printf("area:%f",area);
    return 0;
}

float circleArea(float radius)
{
    float result;
    result = radius * radius * 3.14159265;
    return result;
}
```

---

### lesson5\_4.c

```
#include <stdio.h>
#include <math.h>
float pythagorean(float a, float b);

int main (void) {
    float x,y,z;

    printf("input length of two sides:");
    scanf("%f %f",&x,&y);
    z = pythagorean(x,y);
    printf("the other side is %f",z);
    return 0;
}

float pythagorean(float a, float b)
{
    float result;
    result = sqrtf(a*a + b*b);
    return result;
}
```

---

### lesson5\_5.c

```
#include <stdio.h>
#include <math.h>
float pythagorean(float a, float b);

int main (void) {
    float x,y,z;

    printf("input length of two sides:");
    scanf("%f %f",&x,&y);
    z = pythagorean(x,y);
    if(z < 0)
    {
        puts("wrong input");
        return 1;
    }
    printf("the other side %f",z);
    return 0;
}

float pythagorean(float a, float b)
{
    float result;
    if(a <= 0 || b <= 0)
        return -1;
}
```

```
    result = sqrtf(a*a + b*b);  
    return result;  
}
```

---

### **lesson5\_6.c**

```
#include <stdio.h>  
void test(int x);  
  
int main (void)  
{  
    int x = 3;  
    test(x);  
    printf("%d",x);  
    return 0;  
}  
  
void test(int x)  
{  
    x = 20;  
}
```

---

### **lesson5\_7.c**

```
#include <stdio.h>  
int g_value = 30;  
void test(void);  
  
int main (void)  
{  
    extern int g_value; // not necessary but recommended  
    test();  
    printf("in main %d\n",g_value);  
    return 0;  
}  
  
void test(void)  
{  
    extern int g_value; // not necessary but recommended  
    printf("in test %d\n",g_value);  
}
```

---

### **lesson5\_8.c**

```
#include <stdio.h>  
int g_value = 30;  
void test(void);
```

```
int main (void)
{
    extern int g_value; // not necessary but recommended
    test();
    printf("in main %d\n",g_value);
    return 0;
}
```

### **lesson5\_8\_sub.c**

```
void test(void)
{
    extern int g_value; // necessary !!
    printf("in test %d\n",g_value);
}
```

---

### **lesson5\_9.c**

```
#include <stdio.h>
void test(void);
```

```
int main (void)
{
    test();
    test();
    test();
    test();
    test();
    return 0;
}
```

```
void test(void)
{
    static int x = 1;
    printf("function test is called %d times\n", x);
    x++;
}
```

---

### **lesson5\_10.c**

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

```
void test(void);
```

```
int main (void)
{
    test();
    return 0;
}
```

```

}

void test(void)
{
    static int x = 1;
    printf("function test is called %d times\n", x);
    x++;
    if(x < 30)
        test();
}

```

---

### **lesson5\_11.c**

```

#include <stdio.h>
#include "myLib.h"

```

```

int main (void)
{
    float radius,area;
    radius = pythagorean(6,8) / 2.0;
    printf("radius %f\n",radius);
    area = circle(radius);
    printf("result %f\n",area);
    return 0;
}

```

### **myLib.h**

```

float pythagorean(float a, float b);
float circle(float radius);

```

### **myLib.c**

```

#include "myLib.h"
#include <math.h>

```

```

float pythagorean(float a, float b)
{
    return sqrtf(a*a + b*b);
}

```

```

float circle(float radius)
{
    return radius * radius * 3.14159265;
}

```