Programming Fundamentals Lab

Lab # 07

Functions

Instructor: Fariba Laiq

Email: fariba.laiq@nu.edu.pk

Course Code: CL1002

Semester Fall 2022

Department of Computer Science,

National University of Computer and Emerging Sciences FAST
Peshawar Campus

Contents

Function

- A function is a collection of statements that performs a specific task and it runs only when called.
- They let you divide complicated programs into manageable pieces.
- Functions are commonly used to break a problem down into small manageable pieces. Instead of writing one long function that contains all of the statements necessary to solve a problem, several small functions that each solve a specific part of the problem can be written. These small functions can then be executed in the desired order to solve the problem.

Advantage of functions

● By using functions, we can avoid rewriting same logic/code again and again in a program.

● We can call C functions any number of times in a program and from any place in a program.

● We can track a large C program easily when it is divided into multiple functions.

Types of Functions

There are two types of functions in C programming:

1. Library Functions:

They are the functions which are declared in the C header files such as scanf(), printf(), etc.

2. User-defined functions:

They are the functions which are created by the C programmer, so that he/she can use it many times. It reduces the complexity of a big program and optimizes the code.

Components of a Function

There are three components of a C function.

i. Function declaration

A function must be declared globally in a c program to tell the compiler about the function name, function parameters, and return type. But in case you define a function before main(), in that case you don't need to declare the function.

ii. Function call

Function can be called from anywhere in the program. The parameter list must not differ in function calling and function declaration. We must pass the same number of arguments as it is declared in the function declaration.

iii. Function definition

It contains the actual statements which are to be executed. It is the most important aspect to which the control comes when the function is called. Here, we must notice that only one value can be returned from the function.

Creating a Function in C

The syntax of creating function in c language is given below:

return_type function_name (data_type parameter...)

{

    // body of the function containing the code to be executed

 }

Return Value

A C function may or may not return a value from the function. If you don't have to return any value from the function, use void for the return type. Let's see a simple example of C function that doesn't return any value from the function.

Example without return value:

```
void hello()

{

    printf("hello c");

}
```

If you want to return any value from the function, you need to use any data type such as int, long, char, etc. The return type depends on the value to be returned from the function. Let's see a simple example of C function that returns int value from the function.

Example with return value:

```
int get()

{

    return 10;

}
```

In the above example, we have to return 10 as a value, so the return type is int. If you want to return floating-point value (e.g., 10.2, 3.1, 54.5, etc), you need to use float as the return type of the method.

```
float get()

{

    return 10.2;

}
```

Now, you need to call the function, to get the value of the function.

Different ways to define a Function

A function may or may not accept any argument. It may or may not return any value. Based on these facts, there are four different aspects of function.

1. function without arguments and without return value

2. function with arguments and without return value

3. function without arguments and with return value

4. function with arguments and with return value

1. Function without argument and return value

Example 1

```c
#include <stdio.h>
void printme();
int main ()
{
        printf("Hello\n");
        printme();
        return 0;
}
void printme()
{
        printf("Function without any parameter and return type\n");
}
```

Output

```
Hello

Function without any parameter and return type
```

2. Function with argument and without return value

Example 2

```c
#include<stdio.h>
void sum(int a, int b);
int main()
{
        sum(4,3);
        return 0;
}
void sum(int a, int b)
{
        printf("The sum is %d\n",a+b);
}
```

Output

```
The sum is 7
```

3. Function without argument and with return value

Example 3

```c
#include<stdio.h>
int sum();
int main()
{
        int result;
        result = sum();
        printf("Result = %d\n",result);
        return 0;
}
int sum()
{
        int a =4,b=3;
        return (a+b);
}
```

Output

```
Result = 7
```

4. Function with argument and with return value

Example 4

```c
#include<stdio.h>
int sum(int a, int b);
int main()
{
        int result;
        result = sum(2,3);
        printf("Result = %d\n",result);
        return 0;
}
int sum(int a, int b)
{
        int c=a+b;
        return c;
}
```

Output

```
Result = 5
```