

Chapter 13

FUNCTIONS

Q1. What is a Function? Also write down its advantages.

Ans.

Function:

- A function is a complete independent program that performs a specific task.
- The main function call these type of functions according to requirement of a program.
- A program may have repetition of piece of code at various places and it is very difficult to understand and debug larger programs, therefore, a larger program is divided into simple and independent modules called functions and this type of programming is called modular programming.

Advantages of functions:

- Functions make the program easier to understand and maintain.
- They increase the readability of the program.
- Functions may be reused in multiple programs.
- The length of programs can be reduced using functions.
- Functions can be executed as many times as necessary in the program.
- When an error arises, rather than examining the whole program, the infected function is debugged only.

Q2. What are different types of Functions? Explain with the help of examples.

Ans.

Types of functions:

There are two types of functions in C.

- o Built-in functions
- o User defined functions.

Built-in functions:

- These are predefined functions provided by high level language. All built-in functions are available in library and therefore called library functions.
- All the C library functions are defined in header files which must be included in writing C program.

Examples are:

- o printf() and scanf() functions use stdio.h
- o sqrt() function uses math.h
- o getch(), getche(), clrscr() functions use conio.h

User defined functions:

- The functions created by user are called user defined functions.

Q3. How to create and use a user defined function?

Ans.

- There are three steps to create a user defined function
 - o Function prototype or function declaration.
 - o Function definition.
 - o Function calling.

Function Prototype or Function Declaration:

- It is a statement that provides the basic information that the compiler checks and uses a function correctly.
- The function prototype (declaration) consists of name of function its return type, number of parameters and type of parameters (arguments).
- Function prototype is usually written at the beginning of the source file (before main function).
- It is similar to variable declaration.

Syntax:

Return_type function_name (parameter list);

Return_type:

- It represents the data type returned by the function e.g. int, float.
- If the function does not return any value then the keyword void is used as return type.

Function name:

It specifies the name of the function e.g. sum, multiply etc.

Parameters (arguments):

- The information provided to the functions is called parameters or arguments.
- In function declaration we specify the type of parameters separated by comma.
- If no parameter is used the keyword void is used between the parenthesis.
- The parameters specified in the function header are called formal parameters.
- The parameters passed to a function in the function call are called actual parameters.

Example:

- o `int sum(int, int);` or `int sum(int x, int y);`
- o `void display(void);`
- o `float divide(float, int);` or `float divide(float x, int y);`
- Semicolon represents the end of function declaration.

Function Definition:

- Variable declaration and the function logic are implemented in function definition.
- The actual code of the function is called function definition.
- It is written outside the main function (before or after).
- There are two parts of the function definition
 - o Function header
 - o Function body.

Function Header:

- It is the first line of the function definition.
- It is similar to function prototype.
- it is not terminated with semicolon.

Function Body:

- Variable declaration and the function logic are implemented in function definition.
- It is enclosed in curly braces.

Return Statement:

- The return statement in the function body is used to specify the value returned by a function.
- It is the last statement in the function body.
- The general form of return statement is
- **return value/variable/expression;**
- When return statement is executed, expression is evaluated and returned as the value of the function.
- Execution of the function body stops when the return statement is executed.
- If the type of function is void then there is no need of return statement.

Function Calling:

- It is a process that is used to invoke (execute) a function to perform a specific task.
- A function can be called at any point in the main program.

- A function can be called with its name and correct sequence of parameters.
- When function call statement is executed. Then the control is transferred to the function body. And the statements in function body are executed.
- After executing the last statements in the function, the control is transferred to the calling function.

Q4. What are local and global variables?

Ans.

Local Variables:

- All variables that we have declared within a pair of curly braces are called local variables and their scope is local.
- The local variables can only be accessed in the function in which they are declared.
- They are not available outside the function.
- These are also called automatic variables. Variables in the functions are automatically created when the function is executed and destroyed when the function terminates.
- The lifetime of a variable is the duration in which a variable exists in memory (the creation and destruction of the memory variable).
- The scope of a variable refers to the region of a program in which it is accessible.
- The name of a variable is valid only within its scope not outside the scope.

Global Variables:

- The variables that we have declared outside all the blocks of the program are called global or external variables.
- The scope of a variable refers to the region of a program in which it is accessible. These variables can be accessed in any function at any time during the execution.
- They are available for all the functions in the program, following their point of declaration.
- The lifetime of a global variable is until the termination of the program.

Q.5 What are static variables?

Ans. Static Variables:

Static variables are special variables that are declared inside a function by using the keyword “static”. Like local variables, these can only be accessed in the function in which they are declared but they remain in existence for the lifetime of the program.

Another difference between local variables and static local variables is that the initialization in the static variables takes place only once when the function is called for the first time. Each time a function containing a static variable is called, it uses the values in the static variable assigned to it during the last execution of the function.

Since a static variable is initialized only once and is not destroyed, the function containing static variables runs faster. The static variables are usually used to count the number of times a function was called.

Q.6 How is data provided to the function?

Ans. Passing Arguments to Functions:

The arguments are used to provide data to the function. The function performs operations on the data and produces result. The arguments are placed in parentheses. The arguments are either constants or variables. They are written in the same sequence in which they are defined in the function declaration. The arguments in the function definition of the function are called formal arguments or formal parameters.

When a function is called, values are provided to the function. These values are called actual arguments or actual parameters. The data type of actual arguments in the function call must match the corresponding data types in the function declaration.

Actual arguments → temp (a, b, 16, 2); a, b 16, 2 are actual arguments

Formal arguments → void temp (int x, int y, float z) x, y, z are formal arguments

Program:

Write a program in C language that displays a message by using a function.

```
#include <stdio.h>
#include <conio.h>
void display(void);
void main( )
{
    clrscr( );
    display( );
    getch( );
}
void display(void)
{
    printf("Pakistan");
}
```

Output:

Pakistan

taleemcity.com

Program:

Write a program in C language that find maximum number from three numbers by using a function.

```
#include <stdio.h>
#include <conio.h>
void max(int, int, int);
void main( )
{
    int x,y,z;
    clrscr( );
    printf("Enter first number \t");
    scanf("%d",&x);
    printf("Enter second number \t");
    scanf("%d",&y);
    printf("Enter third number \t");
    scanf("%d",&z);
    max(x, y, z);
    getch( );
}
void max(int a, int b, int c)
{
    int m;
    m=a;
    if(b>m)
        m=b;
    if(c>m)
        m=c;
    printf("Maximum Number is %d",m);
}
```

Output:

```
Enter first number    10
Enter second number   20
Enter third number    5
Maximum number is    20
```

Program:

Write a program in C language that find minimum number from three numbers by using a function.

```
#include <stdio.h>
#include <conio.h>
void min(int, int, int);
void main( )
{
    int x,y,z;
    clrscr( );
    printf("Enter first number \t");
    scanf("%d",&x);
    printf("Enter second number \t");
    scanf("%d",&y);
    printf("Enter third number \t");
    scanf("%d",&z);
    min(x, y, z);
    getch( );
}
void min(int a, int b, int c)
{
    int m;
    m=a;
    if(b<m)
        m=b;
    if(c<m)
        m=c;
    printf("Minimum Number is %d",m);
}
```

Output:

```
Enter first number    10
Enter second number  20
Enter third number    5
Minimum number is    5
```