Module 3

Function:

1. A function is a group of statements that together perform a task. A function can also be referred as a method or a sub-routine or a procedure, etc.
2. Every C program has at least one function, which is main(), from where the execution begins.
3. Functions are useful when a block of statements has to be executed again and again. Also they are used to decompose the large program into small segments which makes it easy to understand, maintain and debug.

Types of functions:

1. Inbuit/Library function: Library functions are the predefined functions in C. Programmer has to just call a function in order to use it.
   Eg.:
   main() The execution of every C program starts from this main() function.
   printf() prinf() is used for displaying output in C.
   scanf() scanf() is used for taking input in C.
   sqrt() is used to calculate square root of a number in C
2. User defined function: C allows programmer to define their own function according to their requirement. These types of functions are known as user-defined functions that contains block of statements which are written by the user to perform a task. There are three parts of user-defined function.
   a. Function Declaration (prototype): Every function in C programming should be declared before they are used called as function prototype. Function prototype gives compiler information about function name, type of parameter to be passed and return type.
      Syntax: returntype function_name (parameter types);
      Note: Function prototype is not needed if function definition is written before main() function.
   b. Function definition: A function definition provides the actual body of the function that contains statements to perform specific task.
      Syntax:
      ```
      returntype function_name( parameter_list ){
              set of statements;
      }
      ```
      parameter_list refers to the type, order and number of the parameters of a function. A function may or may not return a value. Return statement is used for returning a value from function definition to calling function.
   c. Calling a Function: To use a function, you will have to call that function to perform the defined task. When a program calls a function, the program control is transferred to the called function. A called function performs a defined task and when its return statement is executed or when closing brace is reached, it returns the program control back to the main program.
      Syntax: function_name(parameter_list);

Actual & Formal Arguments:

The arguments listed in the function calling statements are referred to as actual arguments. These actual values are passed to a function to perform a task. The arguments used in the function definition are referred as formal arguments.
Note: The number and data types of actual and formal arguments should be same.

Recursion/Recursive function:

1. Recursion is the process of repeating items in a self-similar way. A function that calls itself is known as recursive function and this technique is known as recursion in C programming.
2. But while using recursion, programmers need to be careful to define an exit condition from the function; otherwise it will go into an infinite loop.
3. Recursive functions are very useful to solve many mathematical problems, such as calculating the factorial of a number, generating Fibonacci series, etc.
4. Advantages:

a. Recursion is more elegant and requires few variables which make program clean.
b. Recursion can be used to replace complex nesting code in the program.

5. Disadvantages:
   a. It is hard to think the logic of a recursive function.
   b. It is also difficult to debug the code containing recursion.
6. Syntax:

```
void main()
{
recursion();
}
void recursion()
{
recursion();        // function calls itself
}
```

7. Example: Program to find Factorial of a number using recursion