**Functions in C**

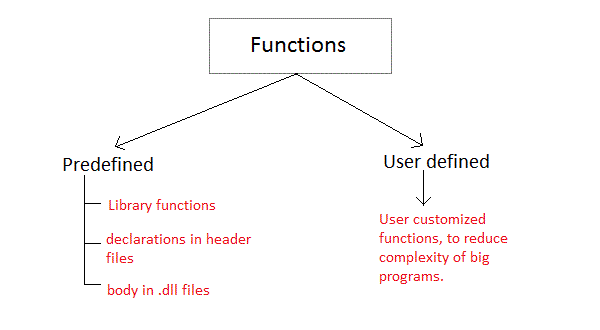
A **function** is a block of code that performs a particular task.

There are many situations where we might need to write same line of code for more than once in a program. This may lead to unnecessary repetition of code, bugs and even becomes boring for the programmer. So, C language provides an approach in which you can declare and define a group of statements once in the form of a function and it can be called and used whenever required.

These functions defined by the user are also know as **User-defined Functions**

C functions can be classified into two categories,

1. **Library functions (input/output/string handling)**
2. **User-defined functions**

****

**Library functions** are those functions which are already defined in C library, example printf(), scanf(), strcat() etc. You just need to include appropriate header files to use these functions. These are already declared and defined in C libraries.

A **User-defined functions** on the other hand, are those functions which are defined by the user at the time of writing program. These functions are made for code **reusability** and for saving time and space.

**Benefits of Using Functions**

1. It provides modularity to your program's structure.
2. It makes your code reusable. You just have to call the function by its name to use it, wherever required.
3. In case of large programs with thousands of code lines, debugging and editing becomes easier if you use functions.
4. It makes the program more readable and easy to understand.

**General syntax for function declaration is,**

Return type function Name(type1 parameter1, type2 parameter2,...);

Function declaration informs the compiler about the function name, parameters is accept, and its return type.

**Return type**

When a function is declared to perform some sort of calculation or any operation and is expected to provide with some result at the end, in such cases, a return statement is added at the end of function body. Return type specifies the type of value(int, float, char, double) that function is expected to return to the program which called the function.

**Note:** In case your function doesn't return any value, the return type would be void.

**Function Name**

Function name is an identifier and it specifies the name of the function. The function name is any valid C identifier and therefore must follow the same naming rules like other variables in C language.

### **Calling a function**

When a function is called, control of the program gets transferred to the function.

**functionName(argument1, argument2,...);**

**Parameter list**

The parameter list declares the type and number of arguments that the function expects when it is called. Also, the parameters in the parameter list receives the argument values when the function is called. They are often referred as formal parameters.

Ex : Multiplication of two numbers

#include<stdio.h>  
**int multiply(int a, int b);** // function declaration  
int main()   
{  
 int i, j, result;  
 printf("Please enter 2 numbers you want to multiply...");  
 scanf("%d%d", &i, &j);  
 result = multiply(i, j); // function call  
 printf("The result of multiplication is: %d", result);   
 return 0;  
}  
  
**int multiply(int a, int b)  
{  
 return (a\*b);** // function defintion, this can be done in one line **}**

**Input** :

Please enter 2 numbers you want to multiply...

10

20

Output : 200

**Ex : addition**

* Define the function before calling it.

#include <stdio.h>

float addition(int num1, float num2)

{

float sum;

sum = num1+num2;

return sum;

}

int main()

{

float res = addition(5, 5.5);

printf ("Output: %f", res);

return 0;

}