Functions

- Function is a way to break our code into chunks so that it is possible for a programmer to reuse them.
- A function is a block of code which performs a particular task.
- It can be used any number of time in a given program.

Example and syntax of function:

```
#include<stdio.h>
void display();     //Function prototype
int main(){
    display();      //Function call
    return 0;
}
void display(){     //Function definition
    printf("A function is called.");
}
```

Function prototype:

Function prototype is a way to tell the computer compiler about the function we are going to define in the program. Here void indicates that the function returns nothing.

Function call:

Function call is a way to tell the compiler to execute the function body at the time the call is made.

NOTE: The program execution starts from the main function in the sequence the instructions are written.

Function definition:

This part contains the exact set of instructions which are executed during the function call. When a function is called from main(), the main function falls asleep and gets temporally suspended.

During this time the control goes to the function being called. When the function body is done executing main() resumes.

Important points:

- Execution of a C program starts from main().
- A C program can have more than one function.
- Every function gets called directly or indirectly from main().

Types of functions:

1. **Library functions:** Commonly require function grouped together in a library file on disk.
2. **User defined function:** These are the functions declared and defined by the user.

Passing value to functions:

We can pass values to a function and can get a value in return from a function.

```
int sum(int a, int b)
```

The above prototype means that sum is a function which takes values **a** (of type *int*) and **b** (of type *int*) and returns a value of type *int*.

Function definition of sum can be:

```
int sum(int a,int b){
    int c;
    c = a+b;     // a and b are parameters
    return c;
}
```

Now we can call **sum(2,3);** from main to get 5 in return. Here 2 and 3 are arguments.

```
int d = sum(2,3);
```

Important points:

1. Parameters are the values or variable placeholders in the function definition.
2. Arguments are the actual values passed to the function to make a call.
3. A function can return only one value at a time.
4. If the passed variable is changed inside the function, the function call doesn't change the value in the calling function.

```
void change(int a){
    a = 77;
}
```

Change is a function which changes a to 77 if we call it from main like this.

```
int b = 22;
change(b);
printf("%d",b);
```

The output will be: 22.

This happens because a copy of b is passed to the change function.