## What is FUNCTION

Function is a piece of code written to perform a specific task. Every C++ program must have at least one function to execute. We have already used main function in the previous tutorials. If a C++ program doesn't have main function then the program cannot be executed.

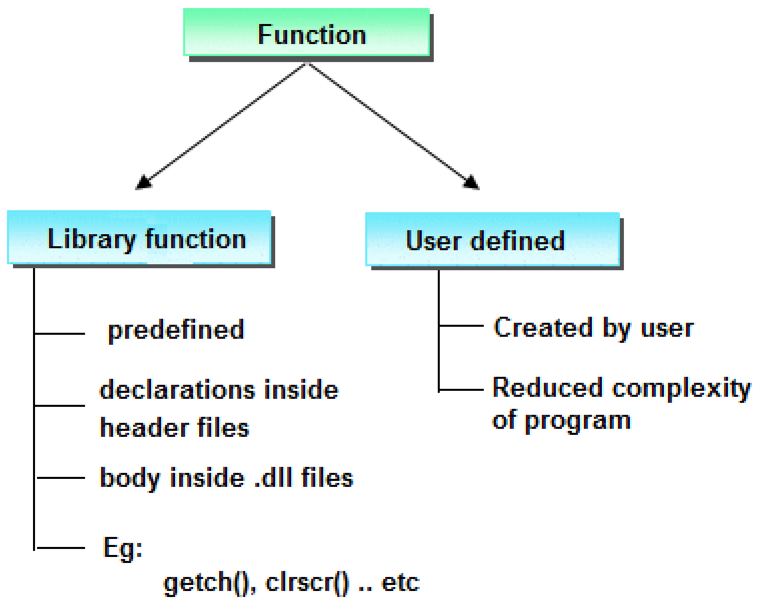
## Advantage of Function

1. Code Re-usability
2. Develop an application in module format.
3. Easily to debug the program.
4. Code optimization: No need to write lot of code.

ccatpreparation.com

## Types of Functions

There are two types of functions in C programming:


Function
Library function
User defined
predefined
declarations inside header files
body inside .dll files
Eg: getch(), clrscr() .. etc
Created by user
Reduced complexity of program

**1. Library Functions:** are the functions which are declared in the C++ header files such as ceil(x), cos(x), exp(x), etc.

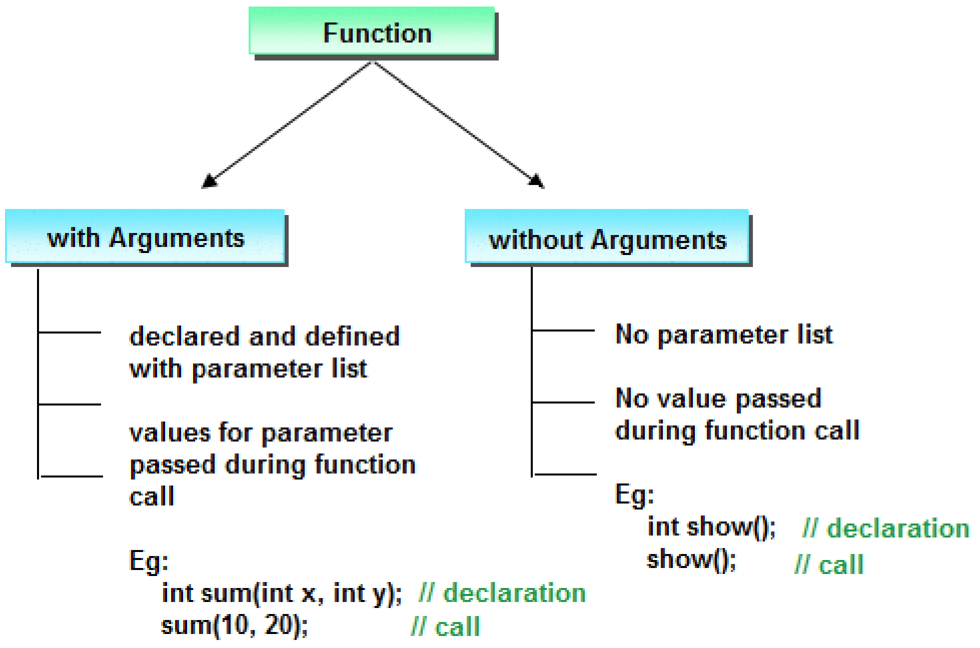
**2. User-defined functions:** are the functions which are created by the C++ programmer, so that he/she can use it many times. It reduces complexity of a big program and optimizes the code.

## Declaration of a function

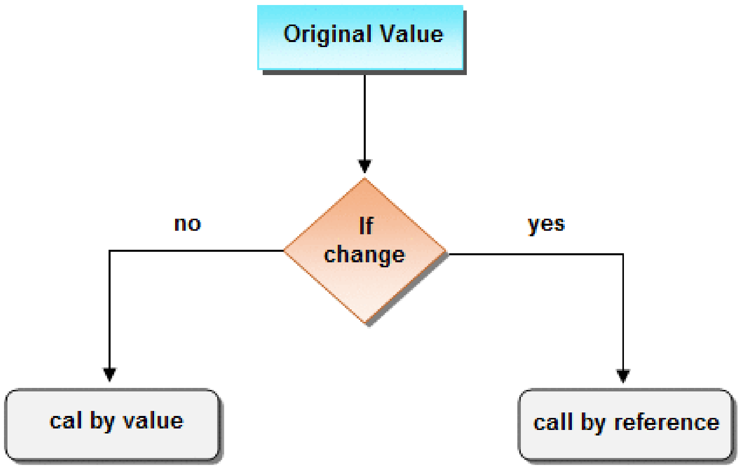
```
return-type function_name(parameters1, parameter2)
{

      //code to be executed
         funtion's body
}
```

## Function Arguments


Function
with Arguments
declared and defined with parameter list
values for parameter passed during function call
Eg:
int sum(int x, int y); // declaration
sum(10, 20); // call
without Arguments
No parameter list
No value passed during function call
Eg:
int show(); // declaration
show(); // call

ccat

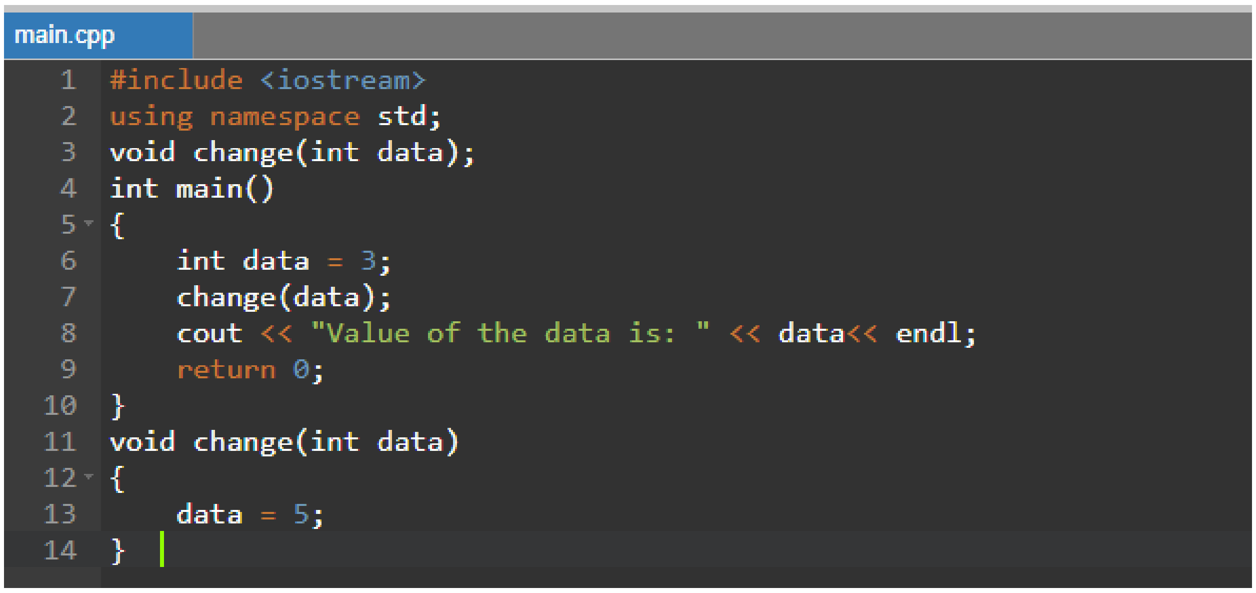


## Call by value in C++

In call by value, **original value is not modified.**

In call by value, value being passed to the function is locally stored by the function parameter in stack memory location. If you change the value of function parameter, it is changed for the current function only. It will not change the value of variable inside the caller method such as main().

Let's try to understand the concept of call by value in C++ language by the example given below:



```cpp
#include <iostream>
using namespace std;
void change(int data);
int main()
{
    int data = 3;
    change(data);
    cout << "Value of the data is: " << data<< endl;
    return 0;
}
void change(int data)
{
    data = 5;
}
```

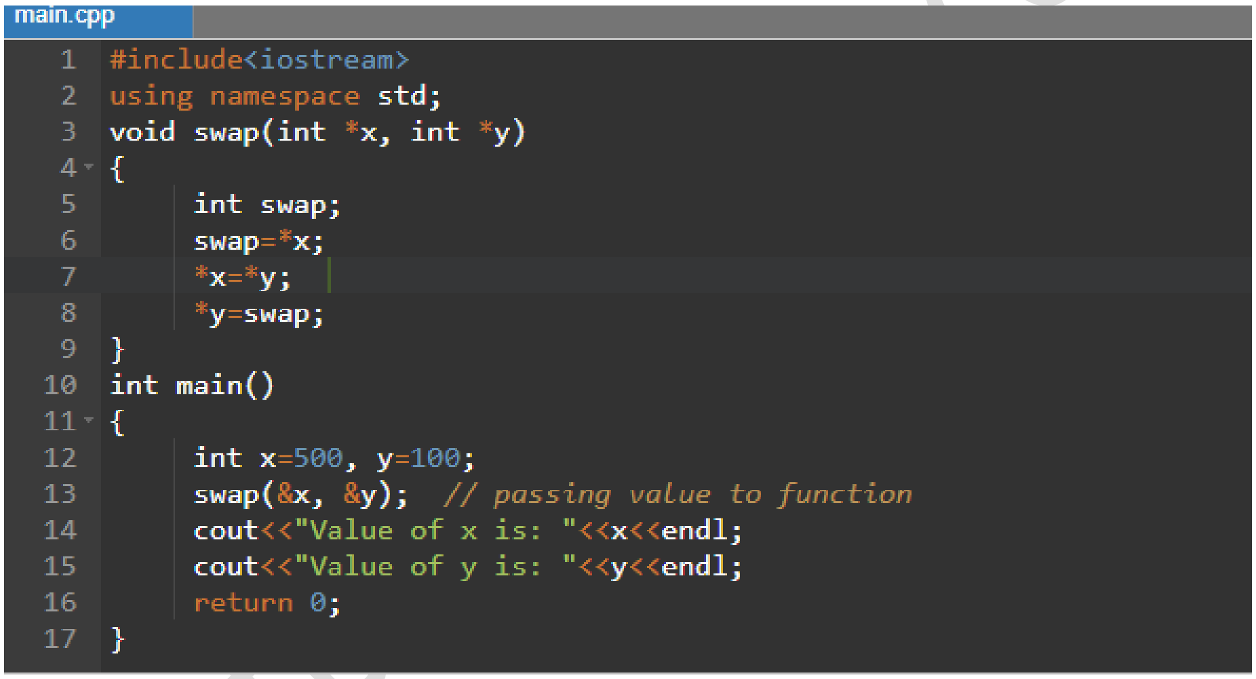
## Call by reference in C++

In call by reference, original value is modified because we pass reference (address).

Here, address of the value is passed in the function, so actual and formal arguments share the same address space. Hence, value changed inside the function, is reflected inside as well as outside the function.

**Note:** To understand the call by reference, you must have the basic knowledge of pointers.

Let's try to understand the concept of call by reference in C++ language by the example given below:



main.cpp

```
#include<iostream>
using namespace std;
void swap(int *x, int *y)
{
     int swap;
     swap=*x;
     *x=*y;
     *y=swap;
}
int main()
{
     int x=500, y=100;
     swap(&x, &y);  // passing value to function
     cout<<"Value of x is: "<<x<<endl;
     cout<<"Value of y is: "<<y<<endl;
     return 0;
}
```

ccatpreparation

Telegram Channel: @ccatpreparations

## Difference between call by value and call by reference in C++

| No. | Call by value | Call by reference |
|---|---|---|
| 1 | A copy of value is passed to the function | An address of value is passed to the function |
| 2 | Changes made inside the function is not reflected on other functions | Changes made inside the function is reflected outside the function also |
| 3 | Actual and formal arguments will be created in different memory location | Actual and formal arguments will be created in same memory location |

ccatpreparation