Lecture 4: C++ Functions (Local & global variables)

Functions in C++:

A function is a block of code that performs a specific task. You can pass data, known as parameters, into a function.

Every C++ program has at least one function, which is **main()**, and all the most trivial programs can define additional functions.

There are two types of function:

1. Standard Library Functions: Predefined in C++

2. User-defined Function: Created by users

Create Function (Function Declaration):

Tells the compiler about a function's name, return type, and parameters.

A function definition provides the actual body of the function.

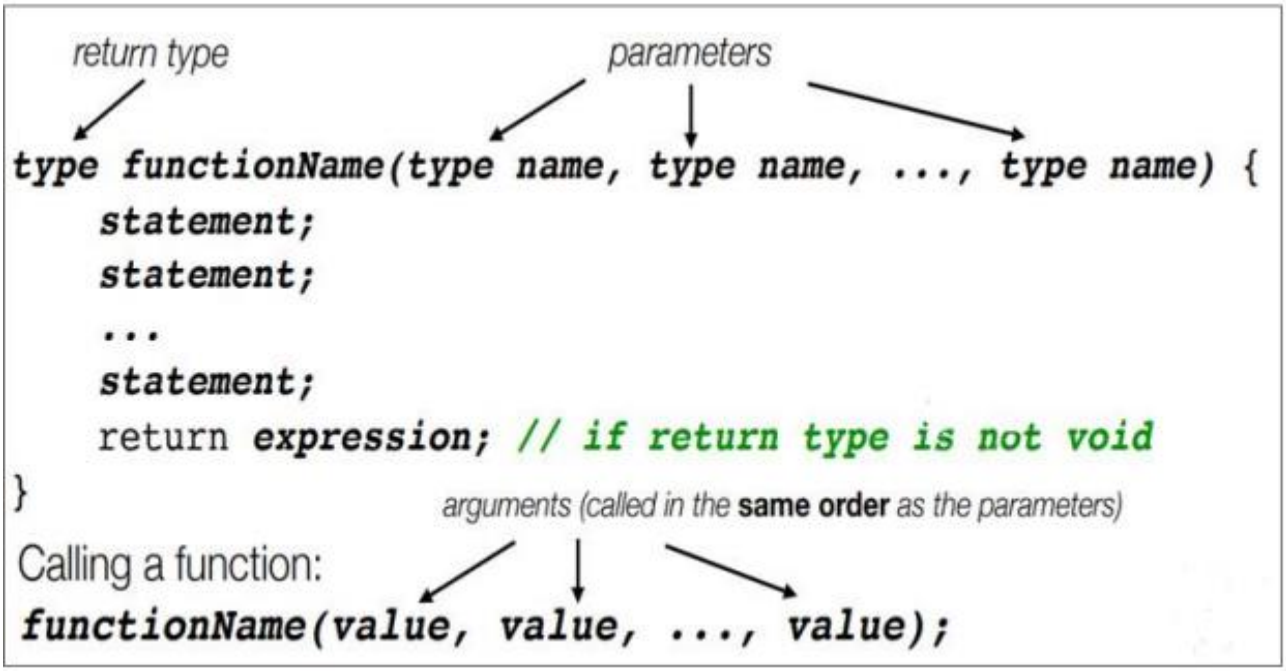
Syntax

```
void myFunction() {
  // code to be executed
}
```

• myFunction() is the name of the function.

• void means that the function does not have a return value.

• inside the function (the body), add code that defines what the function should do.



Note: Parameters are optional so function may contain no parameters.

Variables:

Variables are of two types:

1-Local Variables:

Variables that are declared inside a function or block are local variables. They can be used only by statements that are inside that function or block of code. Local variables are not known to functions outside their own.

Following the examples using local variables.

Example 1:

```
#include <iostream>
using namespace std;
int main () {
 int a, b, c;              // Local variable declaration:
 a = 5;
 b = 4;
 c = a + b;
 cout << c;
 return 0;
}
```

Output:

```
9
```

Example 2:

```
#include <iostream>
using namespace std;
void func1(){
        int x = 4;
        cout << x << endl;
}
void func2(){
        int x = 5;
        cout << x << endl;
}

int main(){
        func1();    //call the function1 to execute it in the main function
        func2();
        return 0;
}
```

Output:

```
4
5
```

2-Global Variables:

Global variables are defined outside of all the functions, usually on top of the program. The global variables will hold their value throughout the life-time of your program. A global variable can be accessed by any function. That is, a global variable is available for use throughout your entire program after its declaration.

Following the examples using global and local variables.

Example 1:

```
#include <iostream>
using namespace std;
// Global variable declaration:
int g;
int main () {
// Local variable declaration:
int a, b;
a = 10;
b = 20;
g = a + b;
cout << g;
return 0;
}
```

Output:

```
30
```

Example 2:

```
#include <iostream>
using namespace std;
int g =10;
void function1();              // Global variable declaration:
int main(){
function1();                   // call function
g = 30;
cout << g << endl;
return 0;
}
void function1()               // function definition
{
g = 20;
cout << g << endl;
}
```

Output:

```
20
30
```

Example 3:

```
#include <iostream>
using namespace std;
// Global variable declaration:
int g = 20;
int main () {
// Local variable declaration:
int g = 10;
cout << g<< endl;    // Local
cout << ::g;            // Global
return 0;
}
```

Output :

```
10
20
```

Note :A program can have the same name for local and global variables but the value of a local variable inside a function will take preference. For accessing the global variable with same rame, you'll have to use the scope resolution operator(::) to call the global variable.