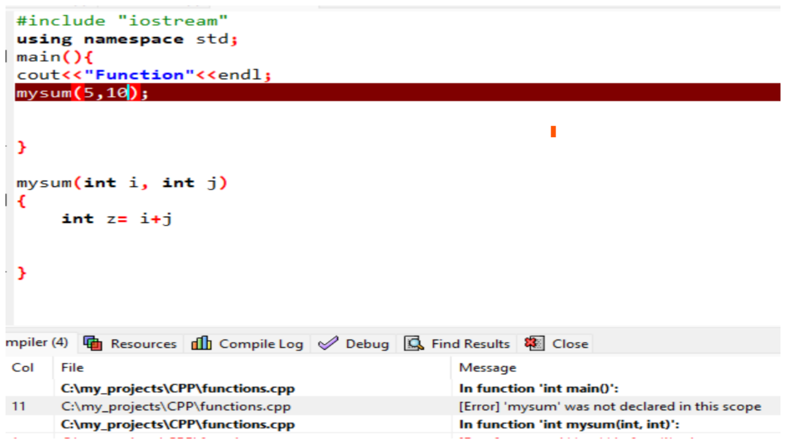
Summary CPP

Sessions No 9(01-11-2022)

- **Function/Method** - A method is a block of code that only runs when it is called. Now, this block of code we can call at any number of times according to our use. Functions are a good way to manage our code, For each different functionality that we think we will use multiple times, we can create a function for that.

- CPP will always call the **main()** function because the *main* function is the entry point of our complete code.

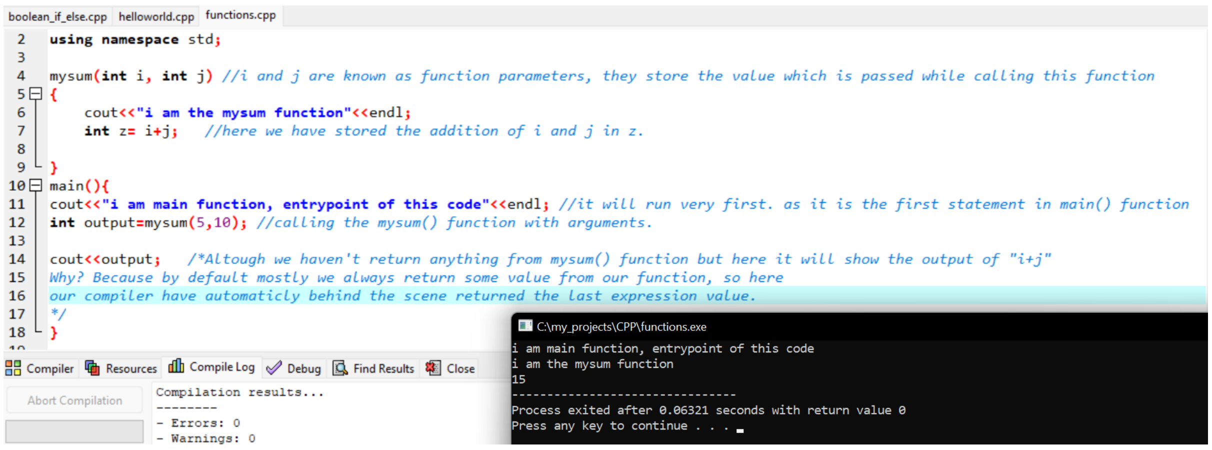
- Especially in CPP, we have to declare our user-defined function before the **main()** function, Hence normally we define the *main()* function in the last of the code file. If we defined the *main()* function first and then our user-defined function, further if we try to call our user-defined function from the *main*() function, Then it will through an error, that ***function is not declared in this scope.***



```
#include "iostream"
using namespace std;
main(){
cout<<"Function"<<endl;
mysum(5,10);


}

mysum(int i, int j)
{
    int z= i+j


}
```

```
Col  File                                   Message
     C:\my_projects\CPP\functions.cpp       In function 'int main()':
11   C:\my_projects\CPP\functions.cpp       [Error] 'mysum' was not declared in this scope
     C:\my_projects\CPP\functions.cpp       In function 'int mysum(int, int)':
```

- See the below example to learn about Functions, arguments, and parameters

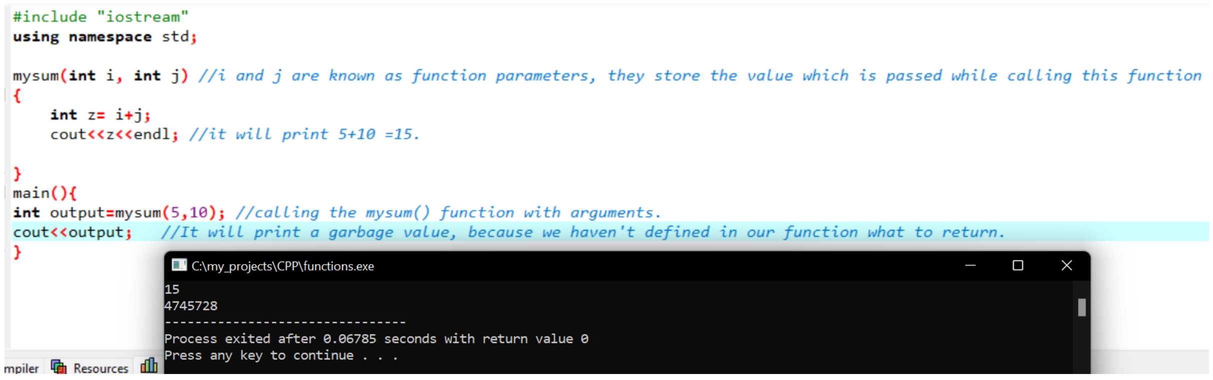


```cpp
using namespace std;

mysum(int i, int j) //i and j are known as function parameters, they store the value which is passed while calling this function
{
    cout<<"i am the mysum function"<<endl;
    int z= i+j;   //here we have stored the addition of i and j in z.

}
main(){
cout<<"i am main function, entrypoint of this code"<<endl; //it will run very first. as it is the first statement in main() function
int output=mysum(5,10); //calling the mysum() function with arguments.

cout<<output;    /*Altough we haven't return anything from mysum() function but here it will show the output of "i+j"
Why? Because by default mostly we always return some value from our function, so here
our compiler have automaticly behind the scene returned the last expression value.
*/
}
```

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
```

```
C:\my_projects\CPP\functions.exe
i am main function, entrypoint of this code
i am the mysum function
15
--------------------------------
Process exited after 0.06321 seconds with return value 0
Press any key to continue . . .
```

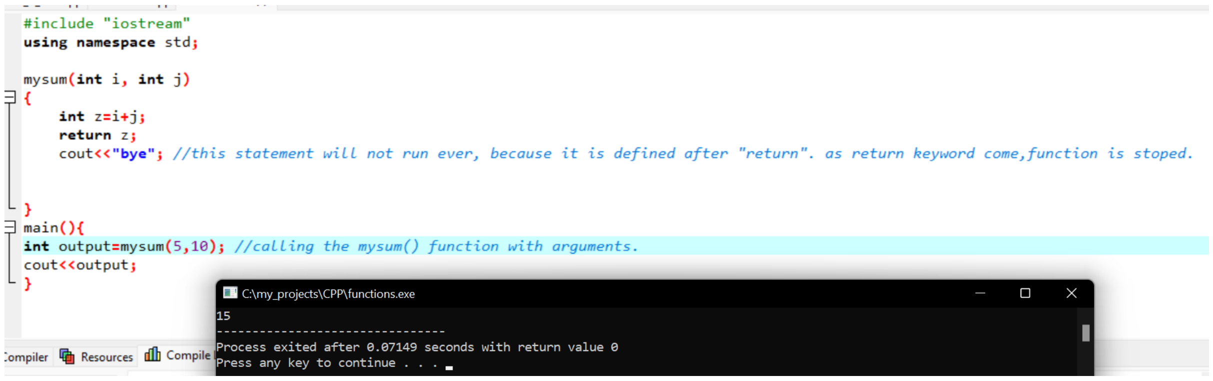
- Now, here we can see we have defined a function, which is using ***cout*** to print "z" on the console, It is printing the output as normal, But in the **main** function "*output*" variable is printing some *garbage value*. Why? Because in the function we haven't defined which value to return back to the *main* function.



```cpp
#include "iostream"
using namespace std;

mysum(int i, int j) //i and j are known as function parameters, they store the value which is passed while calling this function
{
    int z= i+j;
    cout<<z<<endl; //it will print 5+10 =15.

}
main(){
int output=mysum(5,10); //calling the mysum() function with arguments.
cout<<output;   //It will print a garbage value, because we haven't defined in our function what to return.
}
```

```
C:\my_projects\CPP\functions.exe
15
4745728
--------------------------------
Process exited after 0.06785 seconds with return value 0
Press any key to continue . . .
```

- So, the best way to define a function is, when we think the function has performed its duty then in last we can mention the *return* keyword with the value which you wanted to return.



```cpp
#include "iostream"
using namespace std;

mysum(int i, int j)
{
    int z=i+j;
    return z;
    cout<<"bye"; //this statement will not run ever, because it is defined after "return". as return keyword come,function is stoped.

}
main(){
int output=mysum(5,10); //calling the mysum() function with arguments.
cout<<output;
}
```

```
C:\my_projects\CPP\functions.exe
15
--------------------------------
Process exited after 0.07149 seconds with return value 0
Press any key to continue . . .
```

- Whenever a program is run, It becomes a process in RAM, Now in a process, we have mainly 3 parts **Code Section**, **Stack memory**, And **Heap memory**, Now *Code section* stores the code/instructions we have created in our CPP code file. *Stack* and

Heap memory provides space for our variables in RAM. Specially Stack memory is the one which is providing the space to our variables defined in the above examples. Whenever a function is called it is loaded on RAM, and then whatever space the function needs to store its variables, it takes from the stack memory.
Now, when the function job is done, It returns the expected result, and all the space of that function from stack memory will vanish out.