AEA Developers student club

Functions

C++ & Problem-solving Course

23 – 1 – 2025

AKHBAR EL-YOUM
ACADEMY COMMUNITY

CONTENT

01 Introduction

02 Parameter-less functions

03 Parameters

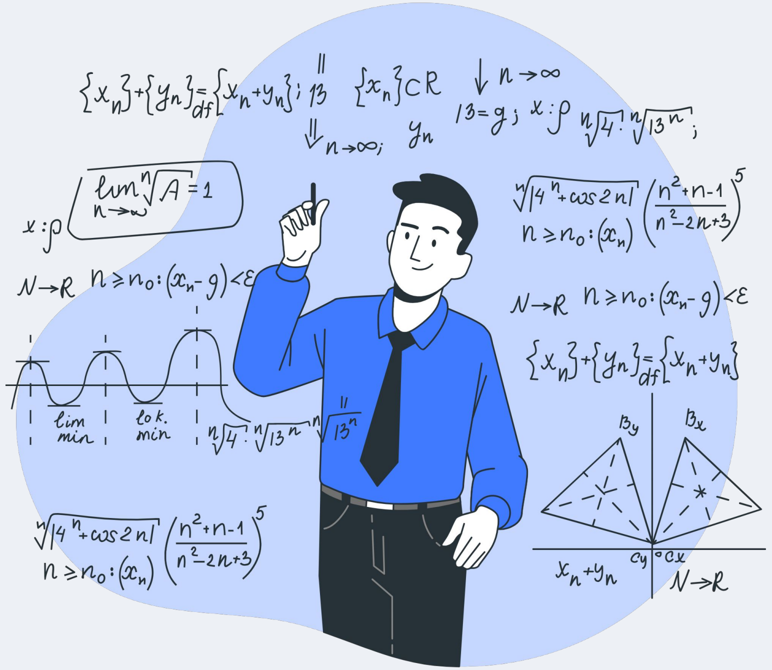
04 Pass by reference

05 return values

01

Introduction

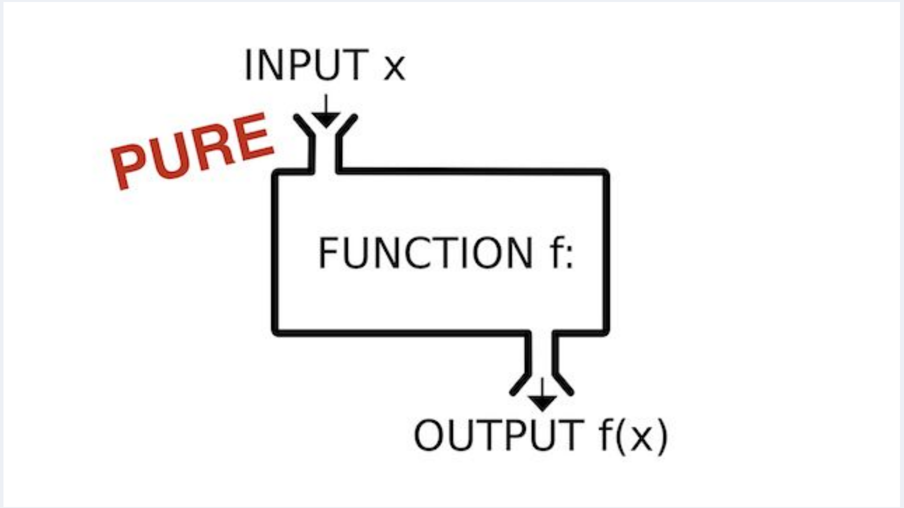
You can enter a subtitle here if you need it



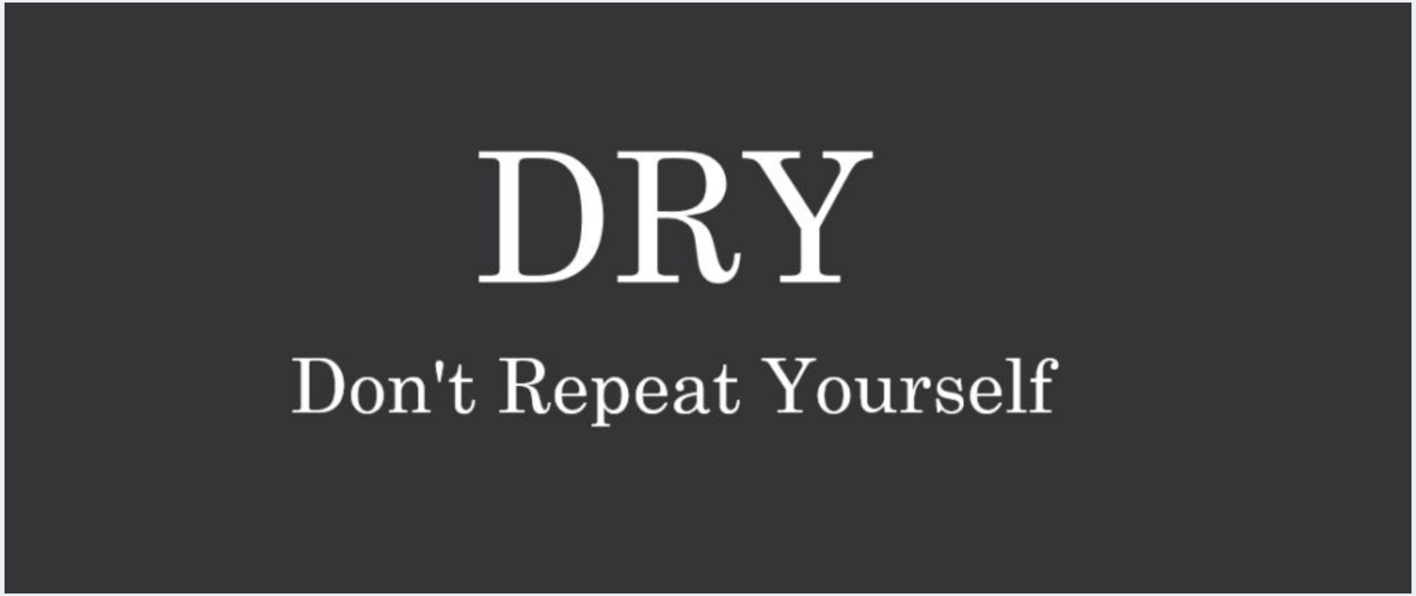
What are functions?

A **function** is a set of statements that takes input, does some specific computation, and produces output. The idea is to put some commonly or repeatedly done tasks together to make a function so that instead of writing the same code again and again for different inputs, we can call this function.

In simple terms, a function is a block of code that runs only when it is called.



So why functions?



DRY

Don't Repeat Yourself

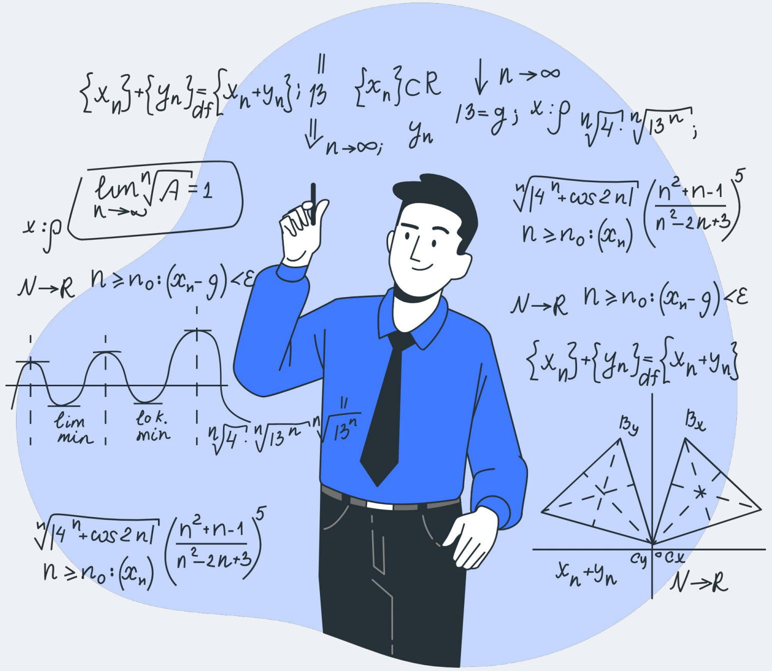
Syntax

```
#include <iostream>
using namespace std;

returnType functionName (parameter1,
parameter2,...)
{
    // function body
}
```

02

Parameter-less functions

You can enter a subtitle here if you need it

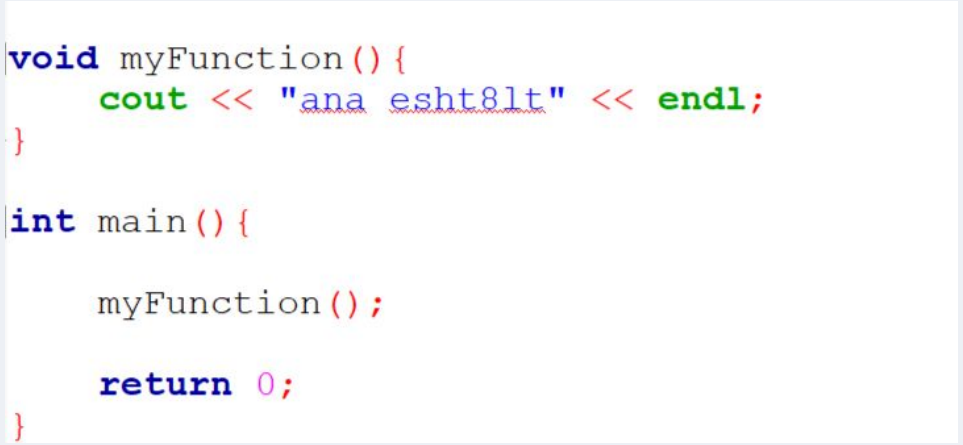


Call a Function

Declared functions are not executed immediately. They are "saved for later use", and will be executed later, when they are called.

To call a function, write the function's name followed by two parentheses () and a semicolon ;

In the following example, myFunction() is used to print a text (the action), when it is called:

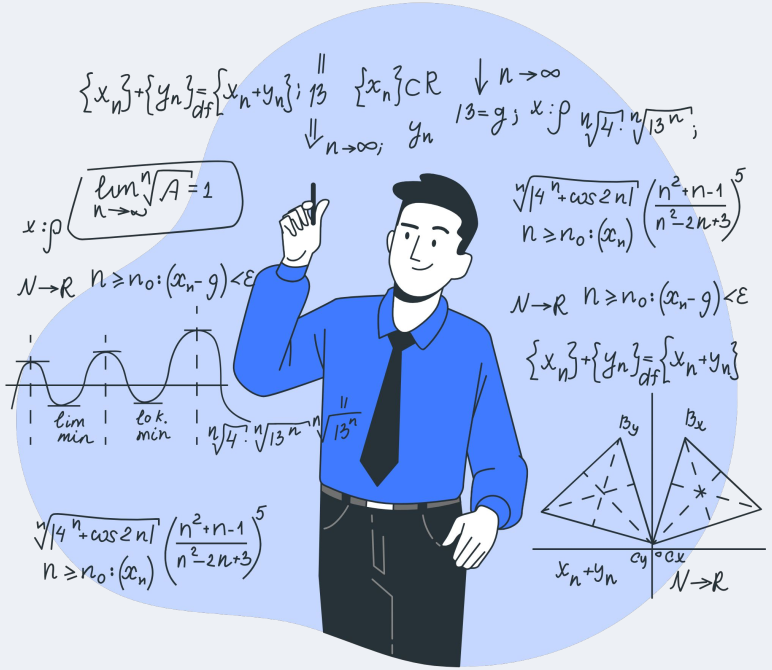


```
void myFunction(){
    cout << "ana esht8lt" << endl;
}

int main(){

    myFunction();

    return 0;
}
```

03

Parameters

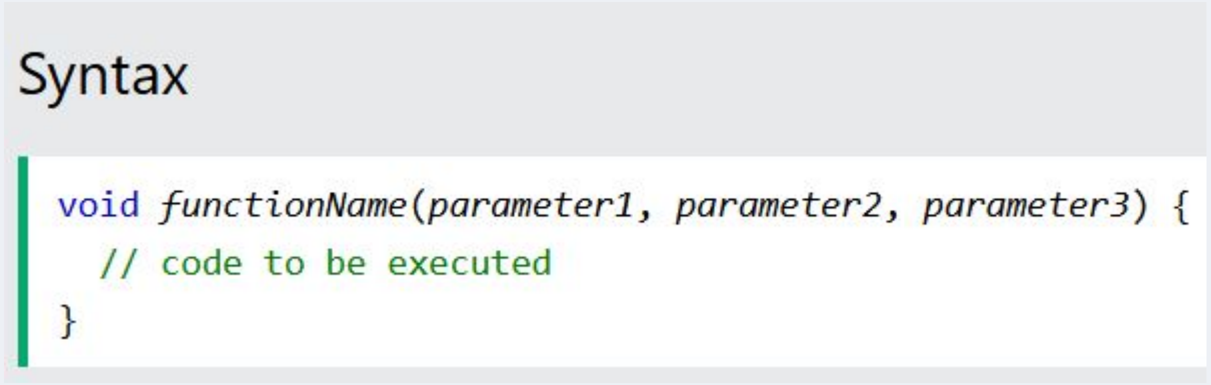
You can enter a subtitle here if you need it



Parameters and arguments

Information can be passed to **functions** as a **parameter**. **Parameters** act as **variables** inside the function.

Parameters are specified after the function name, inside the parentheses. You can add as many parameters as you want, just separate them with a comma:



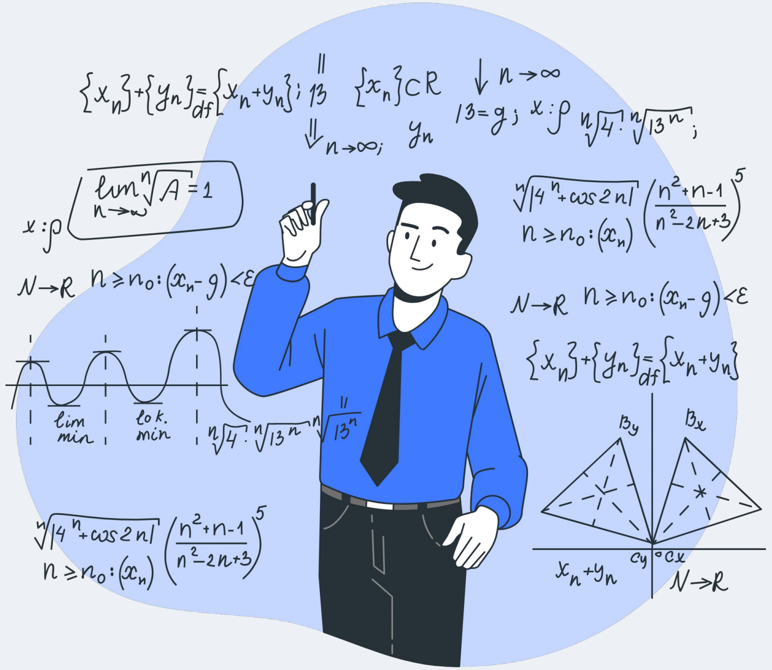
Syntax

```
void functionName(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

04

Pass by reference

You can enter a subtitle here if you need it

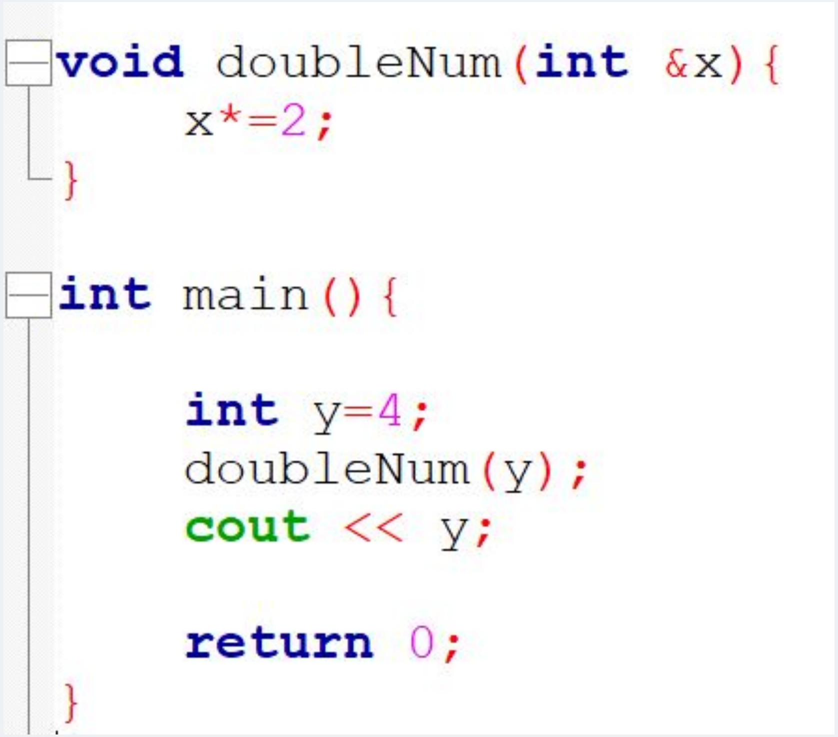


Passing parameters

There are two common ways to pass parameters:

1. **Pass by Value**: Copies the actual parameter's value to the function's formal parameter. Since they occupy separate memory locations, changes in the function don't affect the actual parameter.
2. **Pass by Reference**: Both parameters share the same memory location, so changes in the function directly affect the actual parameter.

Passing by reference:

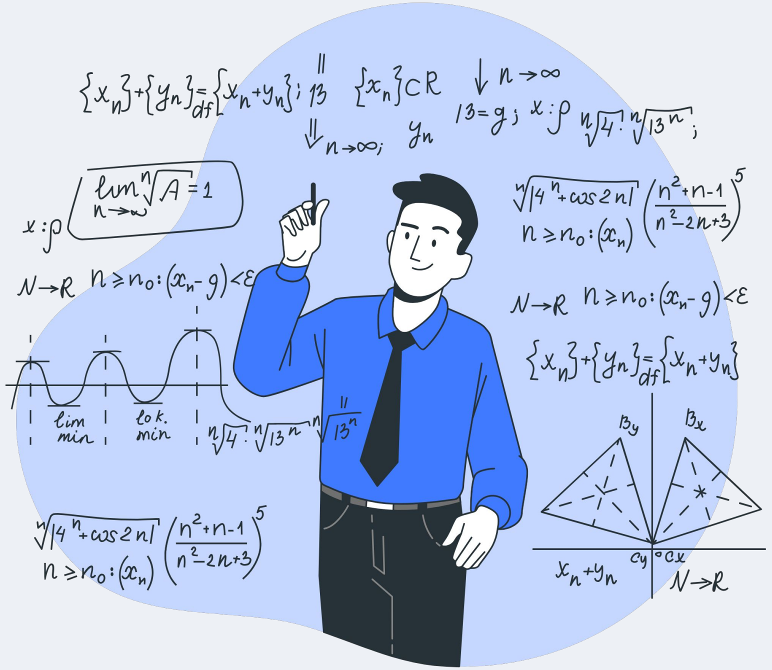


```
void doubleNum(int &x){
    x*=2;
}

int main(){

    int y=4;
    doubleNum(y);
    cout << y;

    return 0;
}
```

05

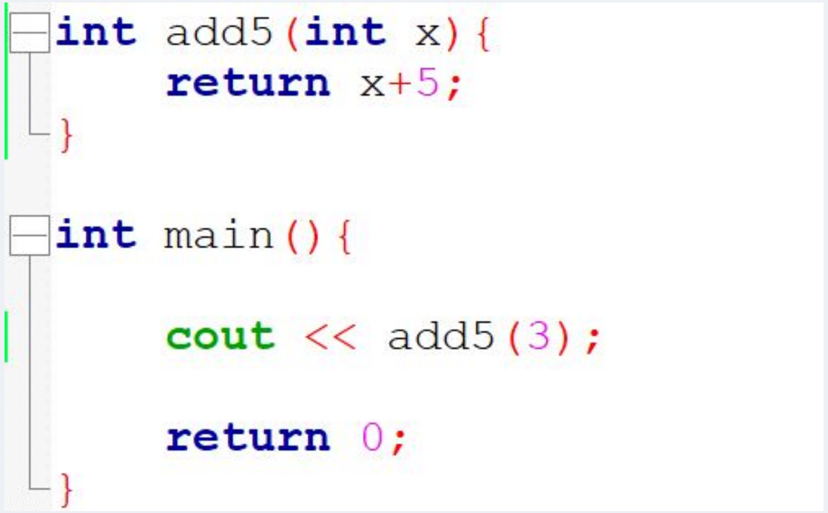
Return values

You can enter a subtitle here if you need it



Return values

The **void** keyword, used in the previous examples, indicates that the function **should not return a value**. If you want the function to return a value, you can use a data type (such as int, string, etc.) instead of void, and use the **return** keyword inside the function:



```
int add5(int x){
    return x+5;
}

int main(){

    cout << add5(3);

    return 0;
}
```

لَولا المَشَقَّةُ سادَ الناسُ كُلُّهُمُ   الجودُ يُفقِرُ وَالإِقدامُ قَتّالُ

–المتنبي