Pass by Value in Functions

In this lesson, you will learn a way to pass the value of actual parameters to the function.

We'll cover the following

- Introduction
  - Basic syntax
  - Example program
  - Explanation
  - passValue function
  - main function

Introduction

Suppose you have sent an email with an attached file to your friend. Your friend has downloaded the file and then made some changes to it. The original document do not be changed by any of the changes made by your friend because your friend has a copy of the original file.

Pass by value is just like sending a copy of the file to the person.

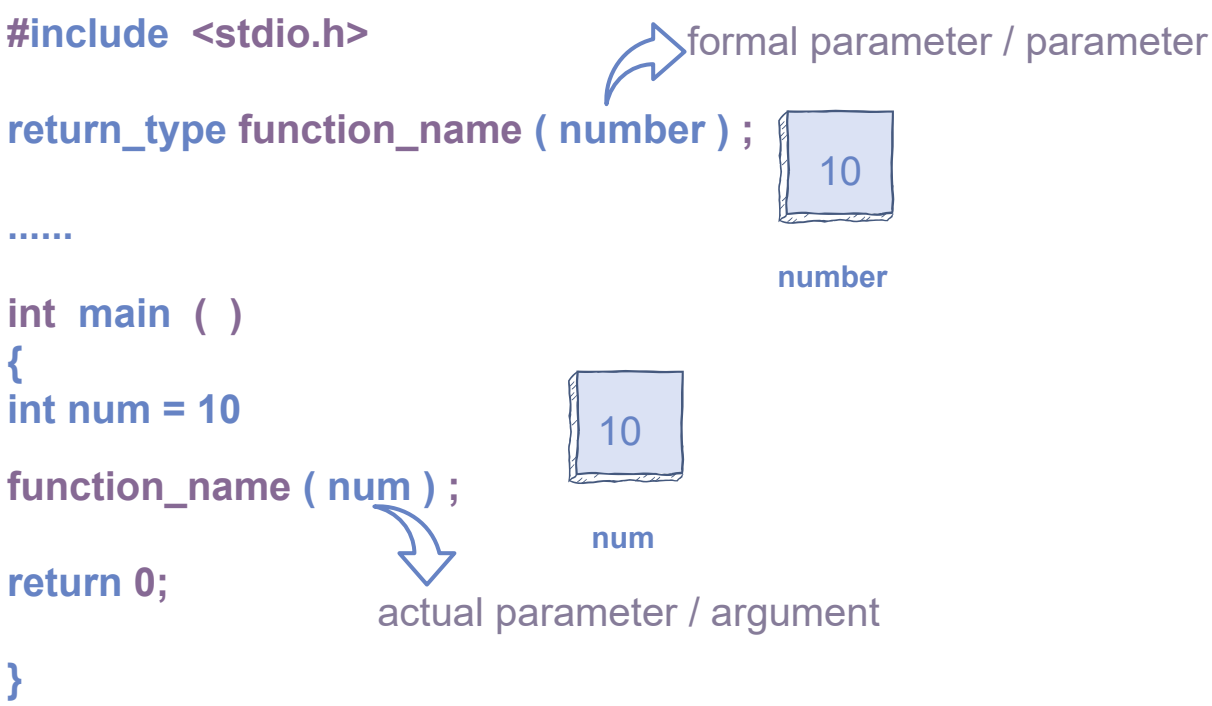
---

*In **pass by value**, when we call a function, we pass the copy of the actual parameters to the formal parameters in the function.*

In pass by value, the actual and formal parameters are stored in different memory locations. Any changes made in the formal parameters inside the function will not affect the values of actual parameters in the main function. In C++, by default, actual parameters are passed by value to the function.

Basic syntax

The general syntax for passing the value of a variable by value is given below:



Example program

Press the **RUN** button and see the output!

```
#include <iostream>

using namespace std;
// function definition
void passValue(int number) {
  // Multiply the number by 10
  number = number * 10;
  cout << "Value of number inside the function = " << number << endl;
}

int main() {
  // Initialize variable
  int number = 10;
  cout << "Value of number before function call = " << number << endl;
  // Call function
  passValue(number);
  cout << "Value of number after function call = " << number << endl;

  return 0;
}
```

Explanation

In the code above, we have two functions:

- passValue function
- main function

passValue function

Line No. 5: The `passValue` function takes a `number` of type `int`. It will perform its task and then returns nothing in output.

Line No. 7: Multiplies the `number` by `10` and stores the result in the `number`

Line No. 8: Prints the updated value of the `number`

main function

Line No. 13: Initializes a variable `number`

Line No. 14: Prints the value of the `number` before the function call

Line No. 16: Calls a function `passValue`. The program execution control is transferred to **Line No. 5**.

Line No. 17: Prints the value of the `number` after the function call

What is the output of the following code?

```
void cube(int number) {
  number = number * number * number;
  cout << "number = " << number << endl;
}

int main() {
  int number = 5;
  cube(number);
  cout << "number = " << number << endl;

  return 0;
}
```

Retake Quiz

This sums up our discussion of passing values to the functions. In the upcoming lesson, let’s explore how we can pass a reference of value to the function.

Stay tuned!