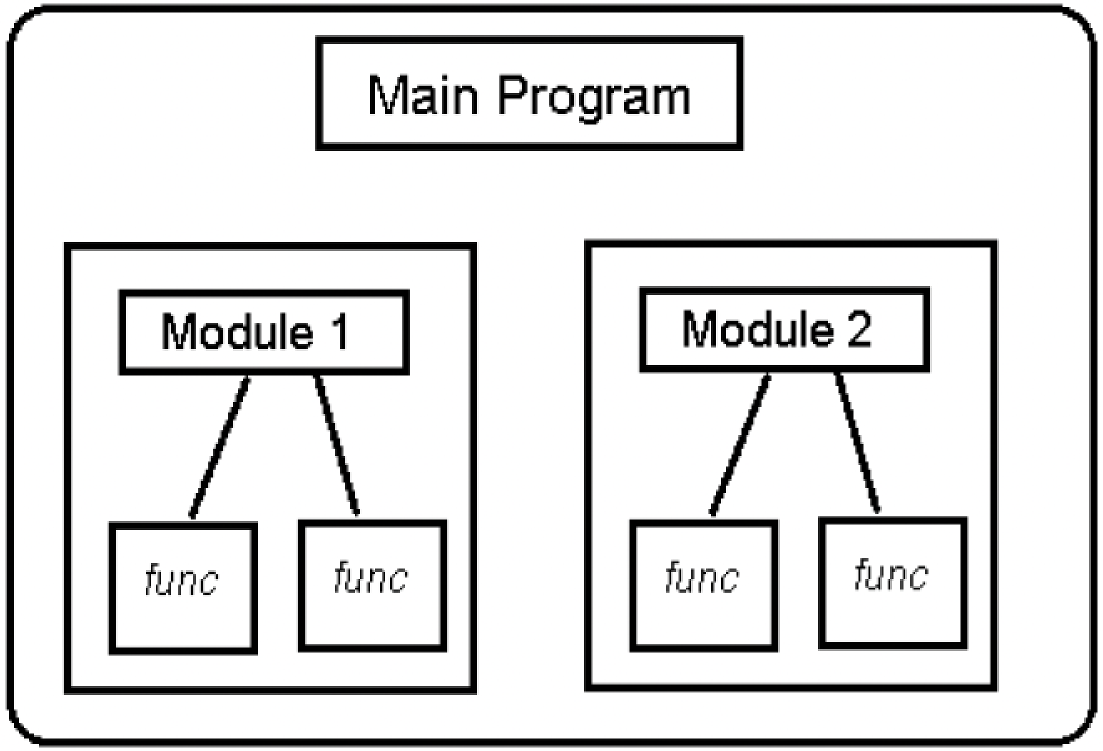
Fundamentals of Computer Programming

CS-110

Course Instructor: Dr. Momina Moetesum

Function Calls

Week 6-c



Learning Objectives

01

To understand what are function parameters

02

To understand the difference between call-by-value and call-by-reference

03

To practice call-by-value and call-by-reference

Function Parameters

Actual Parameters:

- The parameters passed to the function are called

Formal Parameters:

- The parameters received by the function are called formal parameters.

Calling a Function

Functions can be invoked in two ways:

- Call by Value or
- Call by Reference.

These two ways are generally differentiated by the type of values passed to them as parameters.

Call-by-Value

In call by value method of parameter passing, the values of actual parameters are copied to the function’s formal parameters.

There are two copies of parameters stored in different memory locations.

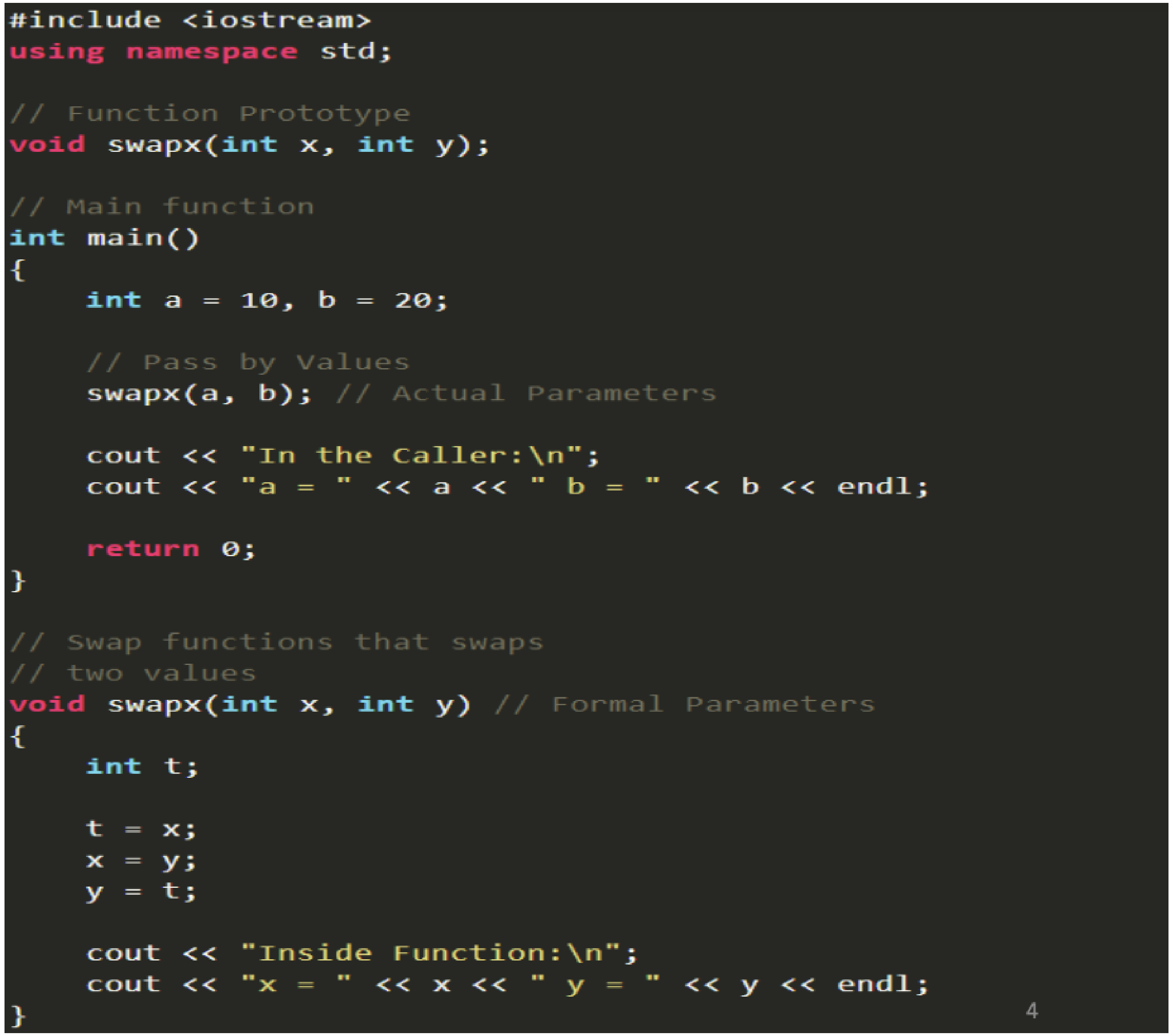
One is the original copy and the other is the function copy.

Any changes made inside functions are not reflected in the actual parameters of the caller.

Example

Output

```
Inside Function:
x = 20 y = 10
In the Caller:
a = 10 b = 20
```



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

// Function Prototype
void swapx(int x, int y);

// Main function
int main()
{
    int a = 10, b = 20;

    // Pass by Values
    swapx(a, b); // Actual Parameters

    cout << "In the Caller:\n";
    cout << "a = " << a << " b = " << b << endl;

    return 0;
}

// Swap functions that swaps
// two values
void swapx(int x, int y) // Formal Parameters
{
    int t;

    t = x;
    x = y;
    y = t;

    cout << "Inside Function:\n";
    cout << "x = " << x << " y = " << y << endl;
}
```

Call-by-Reference

In call by reference method of parameter passing, the address of the actual parameters is passed to the function as the formal parameters.

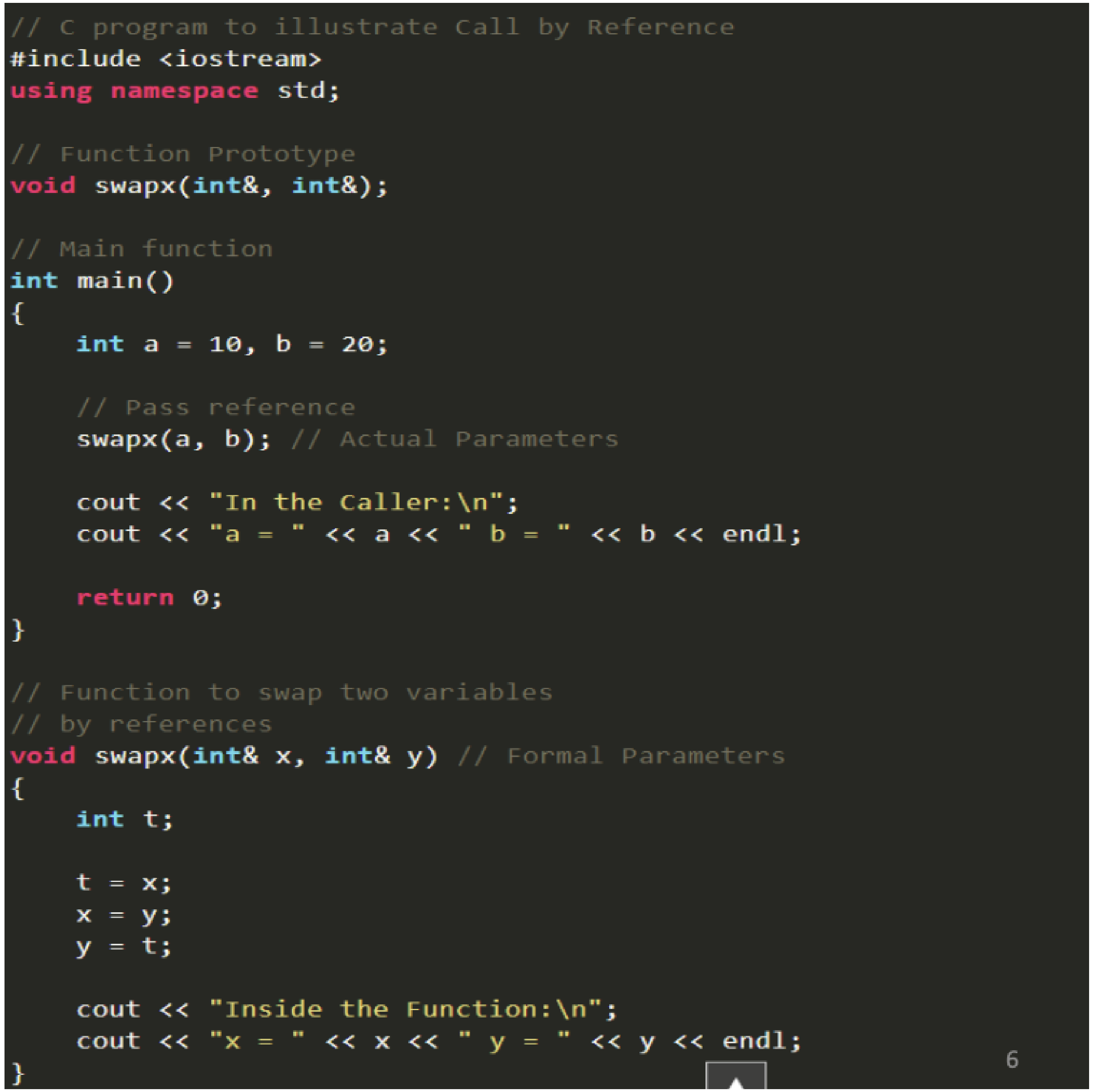
Both the actual and formal parameters refer to the same locations.

Any changes made inside the function are actually reflected in the actual parameters of the caller.

Example

Output

```
Inside the Function:
x = 20 y = 10
Inside the Caller:
a = 20 b = 10
```



```cpp
// C program to illustrate Call by Reference
#include <iostream>
using namespace std;

// Function Prototype
void swapx(int&, int&);

// Main function
int main()
{
    int a = 10, b = 20;

    // Pass reference
    swapx(a, b); // Actual Parameters

    cout << "In the Caller:\n";
    cout << "a = " << a << " b = " << b << endl;

    return 0;
}

// Function to swap two variables
// by references
void swapx(int& x, int& y) // Formal Parameters
{
    int t;

    t = x;
    x = y;
    y = t;

    cout << "Inside the Function:\n";
    cout << "x = " << x << " y = " << y << endl;
}
```

Call By Value	Call By Reference
While calling a function, we pass the values of variables to it. Such functions are known as “Call By Values”.	While calling a function, instead of passing the values of variables, we pass the address of variables(location of variables) to the function known as “Call By References.
In this method, the value of each variable in the calling function is copied into corresponding dummy variables of the called function.	In this method, the address of actual variables in the calling function is copied into the dummy variables of the called function.
With this method, the changes made to the dummy variables in the called function have no effect on the values of actual variables in the calling function.	With this method, using addresses we would have access to the actual variables and hence we would be able to manipulate them.
In call-by-values, we cannot alter the values of actual variables through function calls.	In call by reference, we can alter the values of variables through function calls.
Values of variables are passed by the Simple technique.	Pointer variables are necessary to define to store the address values of variables.
This method is preferred when we have to pass some small values that should not change.	This method is preferred when we have to pass a large amount of data to the function.

Acknowledgment

- Content of these slides are taken from:
  - https://www.geeksforgeeks.org/
  - https://www.tutorialspoint.com/
  - https://www.programiz.com/
  -