

C++ FUNCTION CALL BY REFERENCE

The **call by reference** method of passing arguments to a function copies the reference of an argument into the formal parameter. Inside the function, the reference is used to access the actual argument used in the call. This means that changes made to the parameter affect the passed argument.

To pass the value by reference, argument reference is passed to the functions just like any other value. So accordingly you need to declare the function parameters as reference types as in the following function **swap**, which exchanges the values of the two integer variables pointed to by its arguments.

```
// function definition to swap the values.
void swap(int &x, int &y)
{
    int temp;
    temp = x; /* save the value at address x */
    x = y; /* put y into x */
    y = temp; /* put x into y */

    return;
}
```

For now, let us call the function **swap** by passing values by reference as in the following example:

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

// function declaration
void swap(int &x, int &y);

int main ()
{
    // local variable declaration:
    int a = 100;
    int b = 200;

    cout << "Before swap, value of a :" << a << endl;
    cout << "Before swap, value of b :" << b << endl;

    /* calling a function to swap the values using variable reference.*/
    swap(a, b);

    cout << "After swap, value of a :" << a << endl;
    cout << "After swap, value of b :" << b << endl;

    return 0;
}
```

When the above code is put together in a file, compiled and executed, it produces the following result:

```
Before swap, value of a :100
Before swap, value of b :200
After swap, value of a :200
After swap, value of b :100
```