Function - Call by Value

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.
The parameters passed to the function are called ***actual parameters*** whereas the parameters received by the function are called ***formal 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 of Call by Value

The following example demonstrates the call-by-value method of parameter passing

```
// C program to illustrate call by value
#include <stdio.h>

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

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

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

    printf("In the Caller:\na = %d b = %d\n", a, b);

    return 0;
}

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

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

    printf("Inside Function:\nx = %d y = %d\n", x, y);
}
```

Output

```
Inside Function:

x = 20 y = 10

In the Caller:

a = 10 b = 20
```

Thus actual values of a and b remain unchanged even after exchanging the values of x and y in the function.