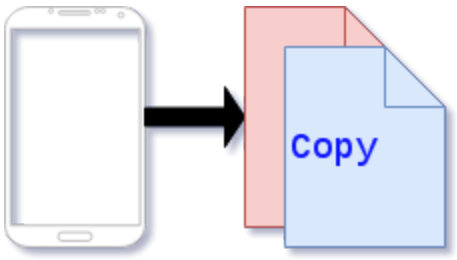
Call by Value

Call by Value is when a function makes a copy of the incoming argument. That is, when a function invocation takes place, the argument variable's value is copied to the function's variable. Therefore, the original variable is not modified. Here is an example to clarify:




```
int function(int number)
{
    number = 50; //Pink to denote function's local variable
    return number;
}
int main()
{
    int number = 100; //Green to denote main's local variable
    cout << number << endl;
    cout << function(number) << endl;
    cout << number << endl;
}
```

There are three cout statements to examine.

The first cout statement will display 100, as that's what number has been assigned.

In the second cout statement, there is a function invocation with the argument number. The contents of number is 100. This value is copied and placed into the variable number. Now both local variables contain the same value but are different entities. The variable number is then set to 50 and then returned. Therefore, 50 will display in the second cout statement. Again, 100 was inputted as the argument and that value was copied to number. There is no direct correlation between number and number. Since number is what was actually returned, that is the value that will be printed.

The last cout statement will print the number 100. The variable number has in no way been altered and will stay 100. A variable can be altered in a function if it is called by reference (look at next page).

Remember, the argument to the function may be a literal value. This method is also known in CSI as pass by value as you are passing only the value to the function.