Call by value / call by reference in C++

In c++, when a function is called, the arguments can be passed too the function either by value or by reference:

1. Call by value : When a function is called by value, a copy of the argument value is made, and the function works on this copy. Any changes made to the argument inside the function are not reflected outside the function.

Example:

*void* increment(*int* *x*) {

*x*++; // changing the value of x, but only the copy inside the function

    cout<<"Inside function value is = "<<*x*<<endl;

}

*int* main() {

*int* a = 5;

    increment(a); // function call

    cout<<"outside function value of is : "<<a; // output: 5, as the original value is unchanged

    return 0;

}

1. Call by reference: When a function is called by reference, the function works on the original argument directly, rather than making a copy. Any changes made to the argument inside the function are reflected outside the function.

Example:

*void* increment(*int* &*x*) {

*x*++; // changing the value of x, which is the original argument

    cout<<"Inside function value is = "<<*x*<<endl;

}

*int* main() {

*int* a = 5;

    increment(a); // function call, which is passed by a reference

    cout<<"outside function value of is : "<<a; // output: 6, as the original value has been changed by the function

    return 0;

}