**In Class Lab 06 - Sorted List**

**Sorted List and Testing Lab**

**Introduction**

For this lab, you will need to implement the UNIT TESTS for the list, as well as the sorted list that it will test.

You will have to spend some time figuring out test cases for the sorted list, and then you will implement them.

**Sorted List**

For the sorted list, this is an average Linked List, except that when an item is inserted, it will be placed in the linked list according to its value. For example,

1 2 3 5

insert 4

1 2 3 4 5

You will need to implement a **Node** class and your **SortedList** class. In the Sorted List, you will need:

* Constructor
* Destructor
* Display function
* SortedPush function, which takes the data and places it in the list in its appropriate place.
* GetAt function, which takes an index and returns the data
* Size function, which returns the amount of items in the list

For your private member variables, you should at least have a pointer to the first Node in the list, and an item counter.

Your **SortedList** and **Node** objects should both be templates.

**Tests**

You will need to implement tests to test the following:

* SortedPush
* GetAt
* Size

The idea behind unit tests is that you know the expected output for some given inputs. **You're not re-implementing logic for Push, GetAt, Size, etc. in your tests!**

You're only saying...

*"If I add two items to the list, then Size should return 2."*

*"If I add 5 and then 1 to the list, then 1 should be at position 0."*

And so on.

Your tests will be what is run from main(), and there are lots of ways to write tests.

It can be as simple as adding functions within your *main.cpp* file that return true or false if the tests all pass, or at least one fails...

bool TestSize()

{

// Test 1

SortedList s1;

if ( s1.Size() != 0 )

{

cout << "Test failed: Size is not 0." << endl;

return false;

}

// Test 2

SortedList s2;

s2.SortedPush( 3 );

if ( s2.Size() != 1 )

{

cout << "Test failed: Size is not 1." << endl;

return false;

}

// Test 3

SortedList s3;

s3.SortedPush( 3 );

s3.SortedPush( 4 );

s3.SortedPush( 5 );

if ( s2.Size() != 3 )

{

cout << "Test failed: Size is not 3." << endl;

return false;

}

}

int main()

{

cout << "TestSize result: " << TestSize() << endl;

return 0;

}

If you want something more sophisticated, you can also look at my [cuTEST framework](https://github.com/Rachels-Courses/cuTEST), which has a TestTemplate that you can use, but that's only if you're feeling adventurous.