CSCI 200 - Fall 2023
Foundational Programming Concepts & Design

Lab XC - Binary Search Trees

Home | HW Sets | Schedule | Files | Help | Links

This lab is due by Thursday, December 07, 2023, 11:59 PM.
As with all labs you may, and are encouraged, to pair program a solution to this lab. If you choose to pair program a solution, be sure that you individually understand how to generate the correct solution.

Jump To: **Rubric Submission**

The BinarySearchTree Class

We are going to now make a new templated class called `BinarySearchTree`. This will necessitate creating a new struct for a `TreeNode` to store the left and right pointers as appropriate.

The `BinarySearchTree` class needs to have the following public methods created:

- `BinarySearchTree()` - Constructor that sets the root pointer to be null and set the size to be zero
- `BinarySearchTree(const BinarySearchTree& OTHER)` - Copy Constructor that performs a deep copy of the other tree
- `~BinarySearchTree()` - Destructor that deallocates the entire tree, sets the root pointer to be null, sets the size to be zero
- `BinarySearchTree& operator=(const BinarySearchTree& OTHER)` - Copy Assignment Operator that deallocates the existent tree (if any) then performs a deep copy of the other tree

RMP-H

- `void insert(const T VAL)` - inserts a node for the given `VAL`. If `VAL` is less than the root, recursively attempt to add to the left subtree. If `VAL` is greater or equal to the root, recursively attempt to add to the right subtree. Increments the size by one. (*Note: We are not going to be implementing the balanced binary search tree.*)
- `int size() const` - returns the current total number of nodes in the tree

To test your implementation, perform in `main.cpp` the following steps in order:

1. Create a BinarySearchTree of integers
2. Add the value 6
3. Add the value 5
4. Add the value 7
5. Add the value 1
6. Add the value 2
7. Add the value 9
8. Add the value 3

You'll verify your `insert()` is working correctly in the corresponding assignment.

Grading Rubric

Your submission will be graded according to the following rubric:

Points	Requirement Description
0.70	Fully meets specifications
0.15	Submitted correctly by Thursday, December 07, 2023, 11:59 PM
0.15	**Best Practices** and **Style Guide** followed
1.00	**Total Points**

Lab Submission

Always, **always**, **ALWAYS** update the header comments at the top of your `main.cpp` file. And if you ever get stuck, remember that there is LOTS of **help** available.

Zip together your `BinarySearchTree.hpp, main.cpp, Makefile` files and name the zip file `LXC.zip`. Upload this zip file to Canvas under LXC.

This lab is due by Thursday, December 07, 2023, 11:59 PM.

As with all labs you may, and are encouraged, to pair program a solution to this lab. If you choose to pair program a solution, be sure that you individually understand how to generate the correct solution.

Last Updated: 04/08/23 10:14

Any questions, comments, corrections, or request for use please contact jpaone {at} mines {dot} edu.

[Jump to Top] [Site Map]