**Data Structures (2080C) – Lab 7**

***Topics covered: Working with Binary Search Trees***

**Objective:**

The objective of this Lab is to create an implementation of a Binary Search Tree.

**Requirements:**

1. Create a Binary Search Tree class with the following methods. This class should contain a node that is defined separately to hold a string.
   1. Constructor
   2. Destructor
   3. Insert – accepts a node that is created outside this class. This will find the correct place to insert the node and insert it.
   4. Find – accepts a value, locates the value in the tree and returns a pointer to the node. If the value isn’t in the tree, it will return a null pointer.
   5. Size – returns the number of elements in the tree as an integer.
   6. GetAllAscending – returns an array with each node stored in order from smallest to largest (based on the sorting value, not the other data in the node).
   7. GetAllDescending – returns an array with each node stored in order from largest to smallest (based on the sorting value, not the other data in the node).
   8. EmptyTree – removes all nodes in the tree in a way to avoid memory leaks.
   9. Remove – accepts a value, finds the value and removes it from the tree. A pointer to the removed node is returned. The returned node has all pointers set to nullptr. Additionally, if the deleted node has a precedessor, set the predecessor to point left child if present, otherwise right child. In the case of both a left and right child, have the replacement node be the left child, and then have the right child be child of the left <I’ll draw this in class>. In short make the tree still legal 😊 If the value can’t be found, it will return a null pointer.
2. Write the code in your main to test the above functions. Include in your lab report what the tree looks like (you can draw it by hand) after inserting the following nodes in the following orders as well as screen shots of the results from calling GetAllAscending and GetAllDescending:
   1. “Star Wars”, “Star Trek”, “Space Balls”, “Galaxy Quest”
   2. “Cars”, “Monsters , Inc”, “The Incredibles”, “Wall-E”
   3. “Halloween”, “A Nightmare On Elm Street”, “Hocus Pocus”, “Beetlejuice”

Discuss why your tree has the structure it has and what can be done to give it a better structure.

**Lab Submission:**

1. Package all files in a single zip folder and submit the file as a group via Blackboard.
2. **Lab report of tree structure and discussion (paragraph or two) on how to improve it’s structure (recall about ideal case for BST (binary search Tree)**

**Lab Grading:**

1. 20% - Lab attendance
2. 25% - Task 1 has been correctly implemented and meets all requirements.
3. 35% - Task 2 has been correctly implemented and meets all requirements.
4. 20% - Lab report contains all required information and is well written.

If program fails to compile, 0% will be given for that Task.