# Homework Assignment #7 – AVL Tree Implementation

by Evan Olds

Cpt S 223

## Submission Instructions:

Submit source code (zipped) to Angel BEFORE the due date/time. If the Angel submission is not working, then submit to TA via email BEFORE the due date/time. No late submissions will be accepted. "Angel wasn't working" is never an excuse. Optional: Include a readme.txt file in the zip with any relevant information that you want the grader to be aware of.

## Assignment Instructions:

**Read all the instructions *carefully* before you write any code.**

Download the zip file from Angel and open the Visual Studio project included within it. Complete the implementation of the AVL tree class functions.

1. Implement Add function in AVLTree class:

```
bool Add(int dataValue)
```

This function adds the specified value to the tree. If the tree is empty then the value becomes the root node. True is returned on success and false on failure. If the number already exists in the tree or memory cannot be allocated for a new node, then false should be returned. Remember to ensure balance in the tree before returning.

2. Implement Remove function in AVLTree class:

```
bool Remove(int dataValue)
```

This function removes the specified value from the tree, if it exists. If it does exist and is removed, then true is returned. Remember to ensure balance in the tree before returning. If it does not exist in the tree then false is returned and the tree is not modified.

Additional Notes:

Remember the rules for each node in an AVL tree:

1. Entire left subtree is less (same as BST)
2. Entire right subtree is greater (same as BST)
3. abs(DepthOfLeftSubtree - DepthOfRightSubtree) <= 1

You may add a Balance field to the Node struct or alternatively two integer fields that keep track of left and right subtree heights. You can also do the implementation without these values if you wish.