CSE 373 Section Handout #4

1. This problem will give you some practice with the basic operations on AVL Trees. Show the result of inserting 43, 8, 5, 10, 4, 7, 32, 2, 1, 3 in that order into an initially empty AVL tree. Show your work.

2. Draw an AVL tree of height 4 that contains the minimum possible number of nodes.

3. Given a binary search tree, describe how you could convert it into an AVL tree with worst-case time O(n lg(n)). What is the best case runtime of your algorithm?

4. Write pseudocode to determine if an AVLTree is balanced. You may assume that the nodes of the tree keep track of the height of their subtree. What would change if the height wasn't stored?

   You may assume the nodes of the tree look like:

```
class AVLNode {
        int key;
        V value;
        int height;
        AVLNode left;
        AVLNode right;
 }
```

Assume we are implementing the following IntTree class, assuming it is a binary search tree.

```
public class IntTree {
        private IntTreeNode root;

        private class IntTreeNode {
                IntTreeNode left;
                IntTreeNode right;
                int data;
        } // Add your methods here....
}
```

5. Implement the contains method for the IntTree class. What is the runtime of this method? How does this compare to the contains method for a heap/priority queue?

6. Implement printSortedOrder, a method which given the root of a BST, print all elements in sorted order.