CS3610/5610D

Homework #2: (90 points, 4.5 final pts.)
due date: Oct. 14, Monday, 11:59pm
Review on Oct. 15, Tuesday
Midterm Exam on Oct. 17, Thursday.

Turn in your answers for the following questions

1. (Binary Tree, 25pts) Read through the `binaryTreeType` class defined on pages 609-616 and 628-632. Add a recursive implementation of the function, **singleParent()**, that returns the number of the nodes in a binary tree that have only one child. Convert it to an iterative version. Turn in your source code for these two functions.

2. (BT, 15pts) Exercise 14, page 679.

3. (BT, 5pts) Exercise 15, page 679.

4. (BT, 10pts) Exercise 16, page 679.

5. (AVL tree, 10pts) Use a diagram to explain when a right-left double rotation (right rotation followed by a left rotation) is needed in AVL **insertion**, and how it's conducted.

6. (AVL tree, 10pts) Exercise 20, page 681.

7. (BST, 15pts) In Chapter 10, we will learn that every comparison-based algorithm used to sort n elements require at least $\Omega(n\log n)$ comparisons in the worst case. Based on this information, what would be the complexity of constructing an n-node binary search tree, and why? (Hint: Start by establishing a connection between BSTs and sorting algorithms.)

Self-practice; no need to turn in your answers.

1. (Binary Tree, 15pts)

   Add a recursive function, `leafCount()`, that returns the number of leaf nodes in a binary tree. Convert this function to a non-recursive version `nr-leafCount()`.

2. (AVL tree, 5pts) Exercise 19, page 680.