Notes for the Programming Problems (Homework 5)

Problem 1: leafCount method to count only leafs on a tree, and to test (test code in homework)

Helpful code in Pg 550-555 size() for a tree in textbook

STUDY THIS CODE MORE !!

Psuedocode

**Recursive solution**

Check *for a null at the tree root* ***(empty tree)***

*return 0 if so*

*Check for Base Case:*

*if node.getLeft && node.getRight are null*

*return 1* ***(leaf found)***

*General Case:*

*call with rec(node.getLeft) + rec(node.getRight)* ***recursive smaller case***

**Iterative solution**

*Create variable* ***(counter for leafs)***

*Check for null at tree root* ***(empty tree)***

*Create a stack and a BSTNode object and push the root node onto the stack*

*Iterate using a while loop: (now the stack can do the work)*

*Set currNOde to the correct node via stacks top command*

*Pop*

*Check whether it is a leaf using getLeft and getRight (should be null)*

*Increment the count for leafs*

*Check if children exist in left branch*

*Push the next node attached to currNode on the left onto stack*

*Check if children exist in right branch*

*Repeat above for right*

Problem 2: Applications of the binary search, x3. Really 2 concepts in total.

-use an BST to alphabetically sort the words in a sentence by first letter

Specs: Take input from user in the form of a Scanner, which will be stored internally as string data. Then assign each word to a node in the tree (SOMEHOW?). Then use the inorder search to get the words in alphabetical order.

-Integer/number store in BST (of 1000 nodes), then test code to check 20 random using contains

-Building on previous calculate average of all nodes, and finds searching number less than ave

Before doing this problem finish all reading for the rest of chapter.

And special attention to the application code at end of chapter Word Frequency Generator (pg 594-596) with test code (pg. 596-597)

More information about testing the BST code is on pg 575-577

Read the chapter summary to make sure it all fits together before attempting.

Following completion:

1. Balancing a binary search tree (should be straightforward)
2. Go back over Lists ADT for examples and begin in Chapter 7, work back to chap 6 as needed.
3. Read section 6.4 about By Copy or By Reference for ADTs pg 407-411, and review ArrayIndexList