Homework 9

This homework uses a copy constructor and a Clone method to make a copy of a tree. The copy has the following properties:

1. The data associated with nodes in the original tree and the copy have the same values. The data is a single int.
2. The structure of the tree is the same, to be more precise, assume a treeOriginal and a soon to be created treeCopy. Node n in treeCopy with data d should have the same path from the root of the tree to n in the copy as it had in the original.
3. No Node object in tree copy is the same as a Node object in the original. By the same I mean they do not occupy the same memory location.

The function boolean checkStructure(Node t1, Node t2) in HW9.java checks these properties and returns true if they are met, and false otherwise.

What you need to do:

You are given a Tree.java and HW9.java files. HW9.java should not need to be changed. The tree class maintains a root and some functions for manipulating the tree. You should implement a copy constructor and clone method for the class. Note that if you have implemented the copy constructor, the Clone function implementation is trivial.

You will write a Node class. Node maintains the nodes that make up the tree, and is responsible for copying these nodes and inserting new nodes into the tree. It has the following functions:

1. Node(int), which constructs a node with the given data
2. Node(Node), which is a copy constructor
3. static void addNode(Node node, int d), which starting a the root of a tree, or a sub-tree, inserts a node with data whose value is d into the tree. If the tree or sub-tree already contains a node with data d, nothing is done. If the value of the d is less than the value of the data at the current node, the data will be placed in a node in the left subtree of the current node, and *vice versa* for when the vaue of d is larger. An in-order traversal should print a sorted list.
4. static void printInOrder(Node node) does an in-order traversal of the tree or sub-tree rooted at node, and prints out the value of the nodes. The statement *System.out.print(""+node.data+" ");* can be used to print the value. There is no need for new lines.
5. Node getLeft( ), Node getRight( ), int getData( ) are public getters for the different attributes of the tree. The checkStructure function uses these.
6. static void copy(Node toRoot, Node fromRoot), where fromRoot is the tree being copied, and toRoot is the tree that will be a copy of the tree or sub-tree rooted at fromRoot. The copy function is called from the copy constructor of Tree to copy a tree, and recursively from copy.

Your output should match the example output. You may get different numbers if your java.util.Random generates a different sequence than mine, but I've specified a seed in HW9.java to try and not have this happen.

What to turn in:

Turn in your code in a directory called <*userid*>. It should compile by executing *javac HW9.java* in the <*userid*> directory, and executing *java HW9* should run it.

Points for this homework:

2 points for the “Print of tree” data to be sorted.

2 points for the “Print of tree2” output to match the “Print of tree” output.

2 points for the “Print of tree3” output to match the “Print of tree2” output.

2 points for the first “Copy is ok” printed

2 points for the second “Copy is ok” to be printed