Regis University CC&IS CS310 Data Structures Programming Assignment 6: Binary Search Trees

Problem Scenario

The IT manager of the real estate office was at a barbeque over the weekend. He was talking to a programmer friend who was learning about trees and who said they were really cool. Now the IT manager would like or you to convert the data structures in your program to binary search trees.

Program Requirements

You will replace your **RealtorLogImpl** and **PropertyLogImpl** classes from Assn 5 (to work with binary search trees, instead of hash tables).

NOTE: Make sure your input data file is not ordered in any way, or you will end up with very unbalanced binary search trees.

For the **RealtorLogImpl** implementation you will build your own Binary Search Tree. You cannot use any existing Java Collection classes.

For the **PropertyLogImpl**, you will be implementing the TreeMap from the Java Collection.

You will need to create any secondary classes, such as **Nodes**, for each of the implementations.

Use the Realtor license number and the Property MLS number as the keys for each implementation.

The inputs will remain the same as for assn 5 – the Realtor/Property input file, and the RealtorRequests file. You will also provide the same report as last week.

You will add a **traverseDisplay()** method to both **RealtorLogImpl** and **PropertyLogImpl** (similar to the traverse in Assn 3, but this time providing an "in order" recursive traversal of the binary search tree).

The methods will display

Realtor List: OR Property List:

and will then traverse the list being implemented, using the **toString()** method to display each object in the list.

Use these methods to display each list, before you process the RealtorRequests file.

The program must follow all **CS310 Coding Standards** from Content section 1.9.

Additional Requirements

• Your original input data file (containing Realtor and Property data to build the binary trees from) will still be read from the **input** folder in your project.

Place all test data files that you create to test your program in the **input** folder of your project, and name them as follows:

assn6input1.txt
assn6input2.txt

(i.e. number each data file after the filename of assn6input.txt)

Your second input data file will also be read from the input folder in your project.

Place all test data files that you create to test your program in the **input** folder of your project, and name them as follows:

realtorRequests1.txt realtorRequests2.txt

(i.e. number each data file after the filename of realtorRequests.txt)

As a group, all of your test data files should demonstrate that you have tested every possible execution path within your code, including erroneous data which causes errors or exceptions.

- Your output report will still be written to a salesReport.txt file in the output folder in your project.
- Create and/or modify **Javadoc headers**, and generate **Javadoc files**
- Add screen shots of **clean compile** of your classes to the documentation folder.

WARNING: Submittals without the clean compile screenshots will **not** be accepted. (This means that programs that do not compile will **not** be accepted)

Program Submission

This programming assignment is due by midnight of the date listed on the **Course Assignments by Week** page.

- Export your project from NetBeans using the same method as you did for previous weeks.
 - o Name your export file in the following format:

CS310<lastname>Assn<x>.zip

For example: CS310SmithAssn6.zip

• Submit your .zip file to the **Prog Assn 6** Submission Folder (located under **Assignments** tab in online course).

Warning: Only NetBeans export files will be accepted.

Do not use any other kind of archive or zip utility.

Grading

This program will be graded using the **rubric** that is linked under **Student Resources** page.

WARNING:

Programs submitted more than 5 days past the due date will **not** be accepted, and will receive a grade of 0.