COP3538 Project 4 – Binary Search Trees

Due Date: Wed, 11/22/2017 11:59 PM

Turn in:

Submit the zipped Eclipse program including at least Project4.java, BinarySearchTree.java, and States4.csv. The zip file should be named <your last name>_Project4.zip (for example, Liu_Project4.zip). The program should be well documented in the format of doc comments in Java. Detailed formats are found at http://www.oracle.com/technetwork/articles/java/index-147868.html.

Requirements:

Implement a binary search tree class. The binary search tree will be created from the same state data that was used in the last project. Use the following class definition for a node in the tree (put this inside your search tree class and do not change it, except to add comments):

```
private class Node {
   String stateName;
   int statePopulation;
   Node leftChild;
   Node rightChild;

   public Node(String state, int population) {
      stateName = state;
      statePopulation = population;
   }
   public void printNode() {
      System.out.printf("%-25s%,10d\n", stateName, statePopulation);
   }
}
```

(1) . Create the **BinarySearchTree** class should implement the following public methods:

1. A no-arg constructor that creates an empty tree.
2. The method: **public void insert(String state, int population)** that will insert a node into the proper position in the search tree based on state name.
3. The method: **public int find(String state)** that will search the tree for the state of the given name and if found will return the population or -1 if not found.
4. The method: **public void delete(String state)** that will find and delete the given state from the tree.
5. The method: **public void printInorder()** that will traverse the tree in using a Inorder traversal (LNR) and print each node.

6. The method: **public void printPreorder()** that will traverse the tree in using a Preorder traversal (NLR) and print each node.

7. The method: **public void printPostorder()** that will traverse the tree in using a Postorder traversal (LRN) and print each node.

8. The method: **public void printMinimum()** that will find and print the state with the minimum population.

9. The method: **public void printMaximum()** that will find and print the state with the maximum population.

(2) . Create a class called **Project4** that will

1. Read a file (csv) of states and create a binary search tree by calling the **insert** method.

2. Inorder traverse the tree and print all nodes using the **printNode** method as they are visited.

3. Delete states **California**, **Florida** and **Michigan** from the tree by calling the **delete** method; and preorder traverse the resulting tree printing all nodes as they are visited.

4. Search for states **Kentucky**, **Rhode Island** and **Florida** by calling the **find** method. For each search, print the population information of the found states, and not-found message if not found. Note, for each search, you will also print the number of non-leaf nodes visited to have or haven't found the target state.

5. Delete states **Delaware**,**Wyoming**, **West Virginia** and **South Dakota** from the tree, and postorder traverse the remaining tree printing all nodes as they are visited.

6. Print the states with the minimum and maximum population in the tree by calling the **printMinimum** and **printMaximum** methods, respectively.

Provide comments in this form for the **BinarySearchTree** classes:

Comments for the class:

```
/**
 * Detailed description of the class.
 *
 * @author <your name>
 * @version <date you last changed the class>
 */
```

Public method comments:

```
/**
 * Description of the purpose of the method, the meaning of the
```

```
 * input parameters (if any) and the meaning of the return values * (if any).
 *
 * @param  parameter  description of the parameter (one for each)
 * @return description of the return value
 */
```

Provide comments in this form for the **Project4** class.

```
/**
 * COP 3538: Project 4 – Binary Search Trees
 * <p>
 * Description of the class using as many lines as needed
 * with <p> between paragraphs. Including descriptions of the
 * input required and output generated.
 *
 * @author <your name>
 * @version <the date you last modified the program>
 */
public class Project4
{
```

Example:

```
COP3538 Project 4 – Xudong Liu

Binary Search Trees
Enter the file name: States4.csv

There were 50 state records put on the binary search tree.

Inorder Traversal:

State Name              State Population
-----------------------------------------
Alabama                 4,833,722
Alaska                  735,132
. . .
Wisconsin               5,742,713
Wyoming                 582,658

California has been deleted from tree
Florida has been deleted from tree
Michigan has been deleted from tree

Preorder Traversal

State Name              State Population
-----------------------------------------
Louisiana               4,625,470
Hawaii                  1,404,054
. . .
Virginia                8,260,405
Wyoming                 582,658

Kentucky is found with a population of 4,395,295
X nodes visited

Rhode Island is found with a population of 1,051,511
X nodes visited

Florida is not found
X nodes visited

Delaware has been deleted from tree
Wyoming has been deleted from tree
West Virginia has been deleted from tree
South Dakota has been deleted from tree

Postorder Traversal

State Name              State Population
-----------------------------------------
Alaska                  735,132
Alabama                 4,833,722
. . .
Wisconsin               5,742,713
Louisiana               4,625,470


State with the minimum population

State Name              State Population
-----------------------------------------
Vermont                 626,630

State with the maximum population

State Name              State Population
-----------------------------------------
Texas                   26,448,193
```