Project 2: Binary Search Trees

Description:
In this assignment, you will implement a Binary Search Tree (BST) that stores non-negative integers. You will create a BST.java file that will contain an inner node class, as well as methods for performing tree operations. All binary search tree properties should be maintained at all times, regardless of any operation performed.

BST.java Specifications

class Node:

Fields:

- private int key;
- private Node parent;
- private Node leftChild;
- private Node rightChild;

Constructors:

- Node():
  The default constructor. Key should be initialized to -1, and all fields should be initialized to null.
- Node(int key):
  Creates a new node where the key is initialized, but all other fields are null.
- Node(int key, Node parent, Node leftChild, Node rightChild):
  Creates a new node where all fields are initialized according to the parameters.

Methods:

- Getters and Setters for all fields.

class BST:

Fields:

- private Node root;
- private int counter;

Constructors:

- BST():
  The default constructor. root is initialized to null, and the counter is set to zero.

Public Methods:

- int size():
  This method should return the number of elements stored within the tree.
- void insert(int element):
  Inserts a new element into the tree. Duplicates are not allowed. If the element is already in the tree, print "Element is already in the tree!".
- void delete(int element):
  Removes the specified element from the tree. If the element is not currently stored in the tree, print "Element not found!".
- void preorder():
  Prints out all of the elements in the tree according to their order in a preorder traversal.
- void postorder():
  Prints out all of the elements in the tree according to their order in a postorder traversal.
- void inorder():
  Prints out all of the elements in the tree according to their order in an inorder traversal.

Private Methods:

- preorderHelper(Node current)
- postorderHelper(Node current)
- inorderHelper(Node current

Important Note:
For the traversal methods, create helper methods (i.e. preorderHelper, postorderHelper, and inorderHelper) that perform the recursive traversal. During testing, I should be able to call one of the above public methods without supplying any parameter and it should print the appropriate order of elements because it called its corresponding helper method.

Submission Instructions:

- Create a README text file that contains your name and any specific compilation instructions you may have for your program.
- Submit BST.java.
- Make sure you receive the submission email.

Two options for projects:

- Choose a partner – someone you know or meet in the class (at most a group of 3 students)
- Opt to work alone – not recommended, but available