Project 3

Binary Search Trees with Lazy Deletion

Implement binary search tree class with **lazy deletion** that has TreeNode as nested class in Java.

Design the class, TreeNode to have following class variables:

int key; // All keys are in the range 1 to 99

TreeNode leftChild;

TreeNode rightChild;

boolean deleted;

Your program method must have routines to do the following operations.

- insert
  //Should insert a new element to a leaf node. If new element is a duplicate then do nothing. If the new element is previously deleted one, then do not add other copy just mark the previous deleted as valid now
- delete
  //Should not remove the element from the tree. It should just mark the element as deleted.
- findMin
  //Should return the minimum element, but if it is marked deleted return appropriate minimum
- findMax
  //Should return the maximum element, but if it is marked deleted return appropriate maximum
- contains
  //Should return true if a particular element is in the tree and is not marked as deleted
- In order tree Traversal
  //Should print the in order traversal of the tree. Indicating with * symbol for elements that are marked deleted
- Height (returns the height of the tree)
  //Return the height of the tree, count all the elements even the ones that are marked as deleted
- No Of nodes (returns number of nodes + number of deleted nodes)
- //Print size of the tree, count all the elements even the ones that are marked as deleted. And also return the number of deleted elements.

  You may have other routines that may be necessary.
  The program should prompt user with options to do one of the above routines.

Sample Run:

Binary Search Trees

1. Insert
2. Delete
3. Find Max
4. Find Min
5. Contains
6. In order Tree Traversal
7. Height
8. No Of Nodes

Your Option: 1
Enter element: 11
Element inserted

Your Option: 1
Enter Element: 36
Element Inserted

Your option: 1
Enter Element: 5
Element Inserted

Your Option: 6
In order Traversal
5 11 36

Your Option: 7
Height = 1

Your Option: 8
No Of Nodes = 3
No Of Deleted Nodes = 0

Note you will be doing lazy deletion, meaning you will not be deleting the node but only mark it for deletion. It is ok to display the deleted node put an * before it to indicate it is deleted.
Example:
In order Traversal = 2 *5 30 45 47 50 *60
(Indicating 5 and 60 are deleted node)

Do appropriate error checking.

Make sure your source file has your name as a comment at the top. Include a README.txt file explaining the IDE/compiler that you used, and a statement of what works and what doesn't work. Zip your files into one file before submitting. Use good style and layout and comment your code well. For Zip file use the following naming convention
FirstInitial_FullLastName_courseID_sectionNumber.zip

Grading Policy /rubric

1. Insert ---- 10 Points
2. Delete ---- 5 Points
3. Find Max ---- 20 Points
4. Find Min ---- 20 Points
5. Contains ---- 5 Points
6. In order Tree Traversal ---- 10 Points
7. Height ---- 10 Points
8. No Of Nodes ---- 5 Points

Error Checking --- 15 Points