Assignment 5

In this assignment you are asked to create a Java that builds and operates on a Binary Search Tree (BST). This will involve building and operating one or more trees in succession.

Overview

The project zip file contains two files (in addition to the assignment document):

1. BinarySearchTree.java
2. TestBST.java

Files Overview and requirements

BinarySearchTree.java

This is the data definition class for the binary search tree. This class is partially implemented using linked list. This simple implementation builds BST where each node contains a integer key value and two (left and right) children.

Several of the methods declared in the BST class are "stubbed" only to the point that they will satisfy the compiler, but are not truly concrete. Your job will be to fully define these methods so that the class truly fulfills the expectations of the interface. You can also add extra methods if you deem them necessary.

This BST should support following methods:

- Add(int k): This method will add a integer key '*k*' into the BST. If the insertion is successful the method will return the inserted key; if the key is already in the tree it should throw an exception. This method has been implemented.

- Search(int k): The method will search for the key '*k*' the BST starting from root. If the key is found, return the key; otherwise throw an exception. *You need to implement this method.*

- Remove(int k): The method will delete the key 'k' from the tree. If the deletion is successful return the deleted key, otherwise throw an exception. *You need to implement all the three cases:*
  - Case 1: Delete a node with zero child
  - Case 2: Delete a node with only one child
  - Case 3: Delete a node with both children

  For case 1 and 2 see lecture slide and for case 3 you need to adopt the option of finding and replacing the minimum of right sub-tree.

- displayInorder(): Implement the inorder tree traversal and print all the keys of the tree using the inorder traversal.
- findMin(): the method will return the minimum key of the tree if the tree is not empty. *You need to implement this method.*
- findMax(): the method will return the maximum key of the tree if the tree is not empty. *You need to implement this method.*
- isEmpty(): the method will check if the tree is empty. *You need to implement this method.*
- size(): this method will return the number of nodes of the tree. *You need to implement this method.*

TestBST.java

These are a variety of test operations on a series of BST that you can use to test your implementation. If required update the test operation to adapt exception handling that you implemented in your code. For example: if you throw an exception in your search () method, you should provide the try/catch block in the test file to handle the exception.

Requirements

Provide complete implementation of all the methods described above.

Deliverables

You should submit a zip file containing your java source code.