CS 314 Final Review — Binary Search Trees

Verify a Binary Search Tree

Suppose we were given an implementation of a BinarySearchTree but were doubtful of its correctness. Let's write an instance method for a BinarySearchTree class which will verify if `this` is a valid BinarySearchTree.

You will need to ensure that all nodes in `this` follows the BinarySearchTree rule that, for any given node, all nodes' values in its left subtree are less than the node's value and all nodes' values in its right subtree is greater than the node's value. Also, make sure that the `size` instance variable correctly stores the number of nodes in the tree.

Complete the following method.

```
// pre: none
// post: returns true iff the BST is valid
//       this object is unaltered as a result of this call
public boolean verifyBST() {
```

Here are some sample calls to `isSubtree`:

You may use the following `BinarySearchTree` implementation

```
public class BinarySearchTree<E extends Comparable<? super E>>{
  private BSTNode<E> root;
  private int size;

  private static class BSTNode<E extends Comparable<? super E>>{
    private E data;
    private BSTNode left, right;
  }
}
```

You may create a single int array of size 1. **Do not create any other data structures or use any other Java classes or methods.**

```
// pre: none
// post: returns true iff the BST is valid
//       this object is unaltered as a result of this call
public boolean verifyBST() {
```