98. Validate Binary Search Tree

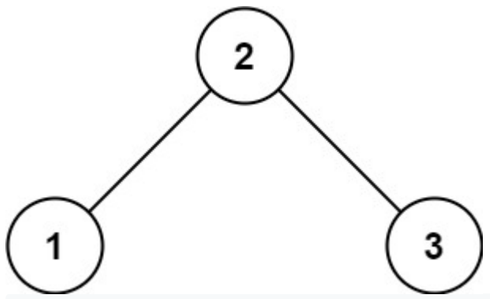
Medium 12770 1026 Add to List Share

Given the `root` of a binary tree, *determine if it is a valid binary search tree (BST)*.

A **valid BST** is defined as follows:

- The left subtree of a node contains only nodes with keys **less than** the node's key.
- The right subtree of a node contains only nodes with keys **greater than** the node's key.
- Both the left and right subtrees must also be binary search trees.

Example 1:




```
Input: root = [2,1,3]
Output: true
```

Example 2:

Java Autocomplete

```java
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     int val;
 *     TreeNode left;
 *     TreeNode right;
 *     TreeNode() {}
 *     TreeNode(int val) { this.val = val; }
 *     TreeNode(int val, TreeNode left, TreeNode right) {
 *         this.val = val;
 *         this.left = left;
 *         this.right = right;
 *     }
 * }
 */
class Solution {

    private boolean isValidBST(TreeNode root, long left, long right){

        if(root == null){
            return true;
        }else if(!(root.val > left && root.val < right)){
            return false;
        }

        return isValidBST(root.left, left, root.val) &&
isValidBST(root.right, root.val, right);
    }

    public boolean isValidBST(TreeNode root) {
        return isValidBST(root, Long.MIN_VALUE, Long.MAX_VALUE);
    }
}
```

Your previous code was restored from your local storage. Reset to default

Testcase Run Code Result Debugger

Accepted Runtime: 0 ms

Your input `[2,1,3]`

Output `true` Diff

Expected `true`