98. Validate Binary Search Tree

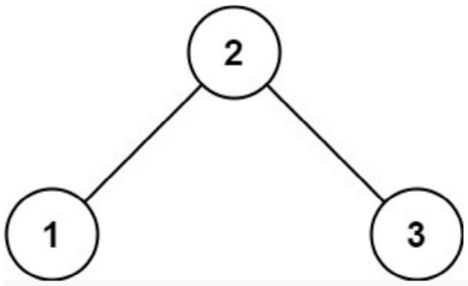
02 April 2022 07:58 PM

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

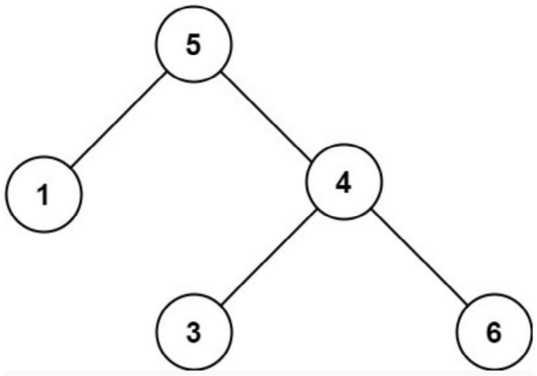
A **valid BST** is defined as follows:

- 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:

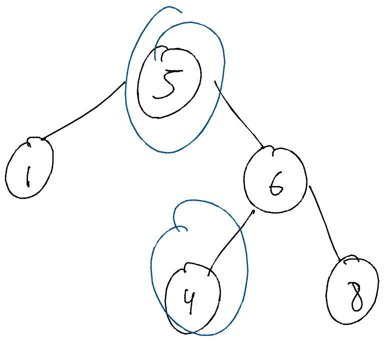


```
Input: root = [2,1,3]
Output: true
```



```
Input: root = [5,1,4,null,null,3,6]
Output: false
Explanation: The root node's value is 5 but its right
child's value is 4.
```

Validate a BST

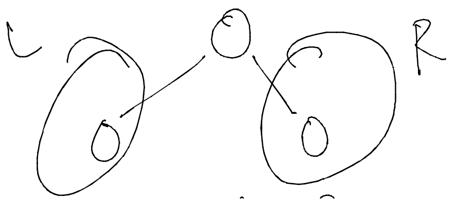


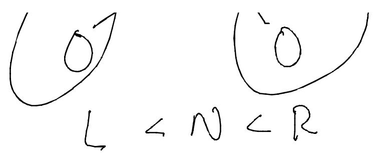
⟹ This is not a B.S.T

because 5 > 4

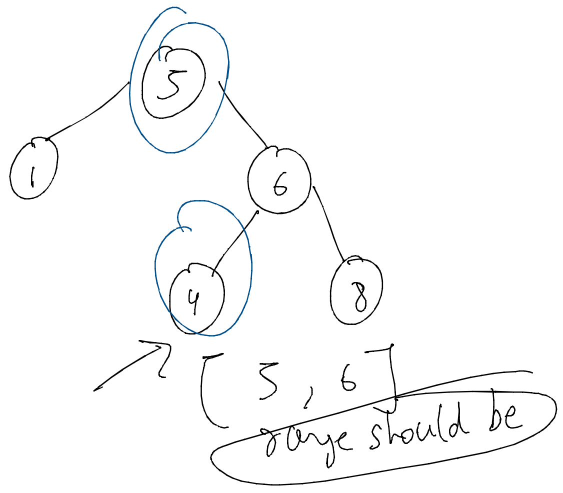
violates the condition

* Condition ⟹

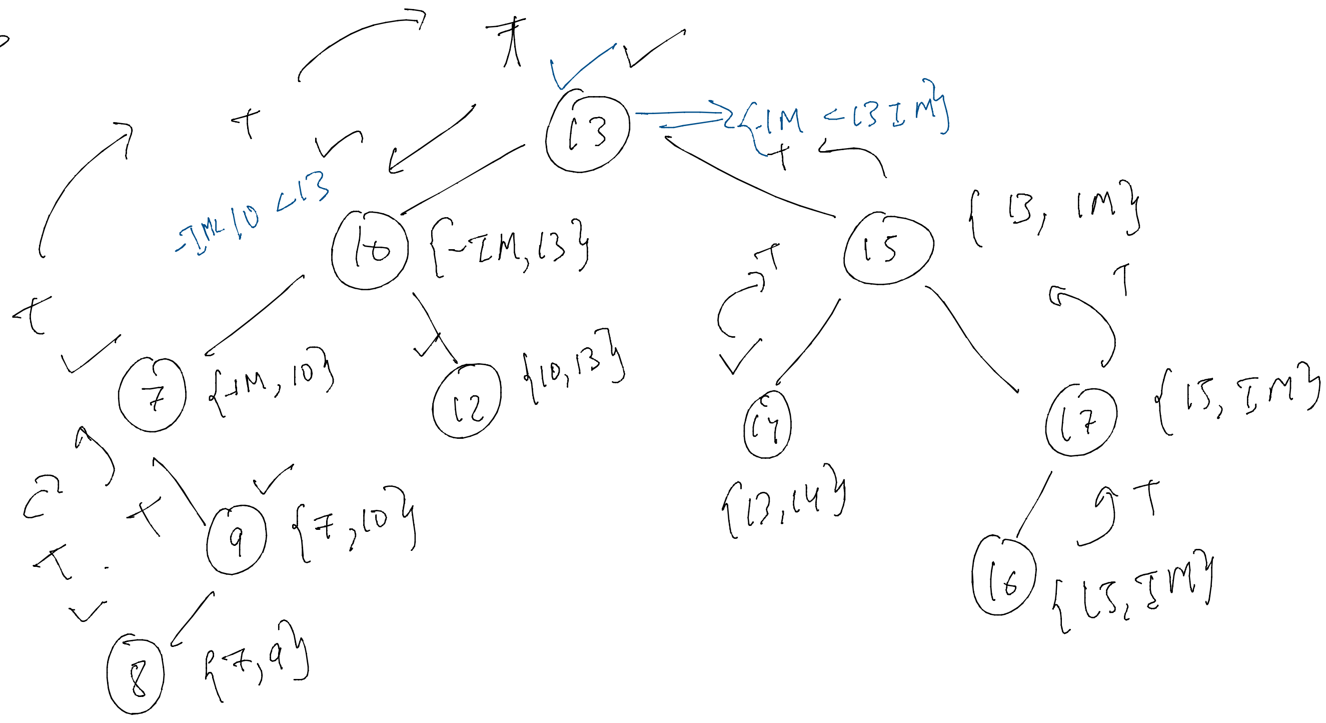




Intuition : for every node give a range for e.g.



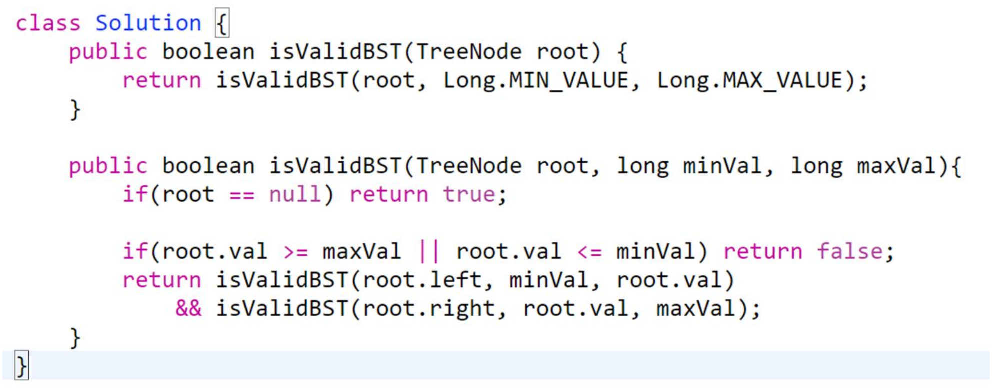
Recursion



* What is initial range ? {-INT MIN, INT MAX}
-IM IM

T.C ⇒ O(N)

S.C ⇒ O(1)



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

    public boolean isValidBST(TreeNode root, long minVal, long maxVal){
        if(root == null) return true;

        if(root.val >= maxVal || root.val <= minVal) return false;
        return isValidBST(root.left, minVal, root.val)
            && isValidBST(root.right, root.val, maxVal);
    }
}
```