Valid a BST

Given a binary tree, determine if it is a valid binary search tree (BST).

Assume a 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.

/\*\*

\* Definition for a binary tree node.

\* public class TreeNode {

\* int val;

\* TreeNode left;

\* TreeNode right;

\* TreeNode(int x) { val = x; }

\* }

\*/

class Solution **{**

long num **=** Long**.**MIN\_VALUE**;**

public boolean isValidBST**(**TreeNode root**)** **{**

**if(**root**==null)** //if node null does´t affects the inorder traversal

**return** **true;**

**if(!**isValidBST**(**root**.**left**))** //Check the left branch

**return** **false;**

**if(**root**.**val**<=**num**)** // if the definition of inOrder traversal is broken, then

**return** **false;**

num **=** root**.**val**;**

**if(!**isValidBST**(**root**.**right**))** //Check the right breanch

**return** **false;**

**return** **true;** // Return true id the inOrder traversal y this subTree is find

**}**

**}**