Marwadi University	Marwadi University Faculty of Technology Department of Information and Communication Technology	
Sem : 5	Name :Pushti Depani	
Day : 17	Date : 03/11/2022	Enrollment No: 92000133018

CP Club 365Days Challenge

Date – 03/11/2022
Programming language – only C/C++ language

Problem Statement

Write the code not just for one input but for many more.(Dynamic code)

Code must be in C OR C++ language only

Given the root of a binary tree. Check whether it is a BST or not.
Note: We are considering that BSTs can not contain duplicate Nodes.
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.

Example 1:

Input:

```
  2
 /   \
1     3
```

Output: 1

Explanation:

The left subtree of root node contains node
with key lesser than the root nodes key and
the right subtree of root node contains node
with key greater than the root nodes key.
Hence, the tree is a BST.

Example 2:

Input:

```
 2
  \
   7
    \
     6
      \
       5
        \
         9
          \
           2
            \
             6
```

Output: 0

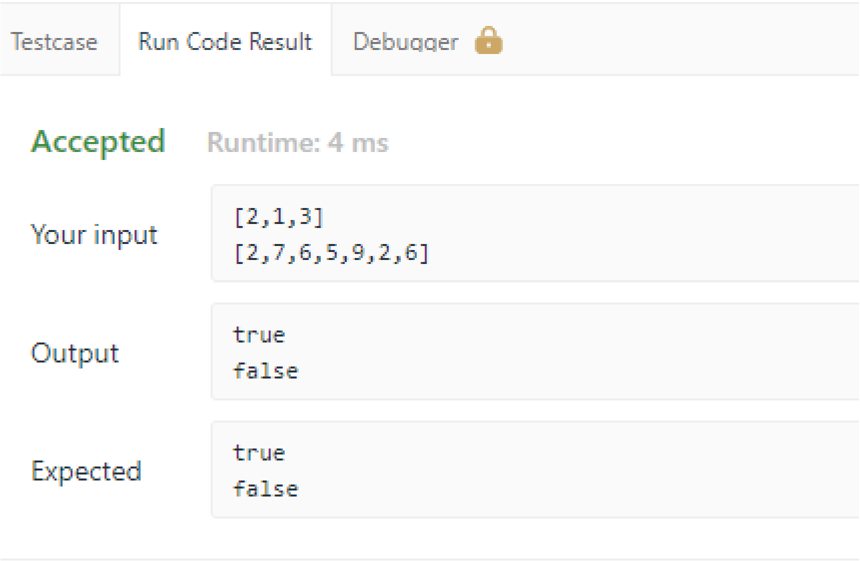
Explanation:

Since the node with value 7 has right subtree nodes with keys less than 7, this is not a BST.

Your Code:

```
class Solution {
public:
	bool isValidBST(TreeNode* root, long min, long max) {
		if(root == NULL){
			return true;
		}
		if(root->val > min && root->val < max) {
			return isValidBST(root->left, min, root->val) && isValidBST(root->right, root->val, max);
		}
		return false;
	}
	bool isValidBST(TreeNode* root) {
		return isValidBST(root, LONG_MIN, LONG_MAX);
	}
};
```

Output (Screen Shot):



Accepted Runtime: 4 ms

Your input

```
[2,1,3]
[2,7,6,5,9,2,6]
```

Output

```
true
false
```

Expected

```
true
false
```

Understanding about problem:

From this problem I have learnt how to validate binary search tree. We have to check if the max value in left subtree is smaller then the node and min value in right subtree is greater then the node. If the current node is null then return true. If the value of the left subtree of the node is greater than or equal to the current node then return false. If the value of the right subtree of the node is less than or equal to the current node then return false. If the left subtree or the right subtree is not a BST then return false. Else return true.

Note: If you can't understand the problem, feel free to contact us and we'll help you. Please don't copy and paste from anywhere.

ALL THE BEST

Team CP Club