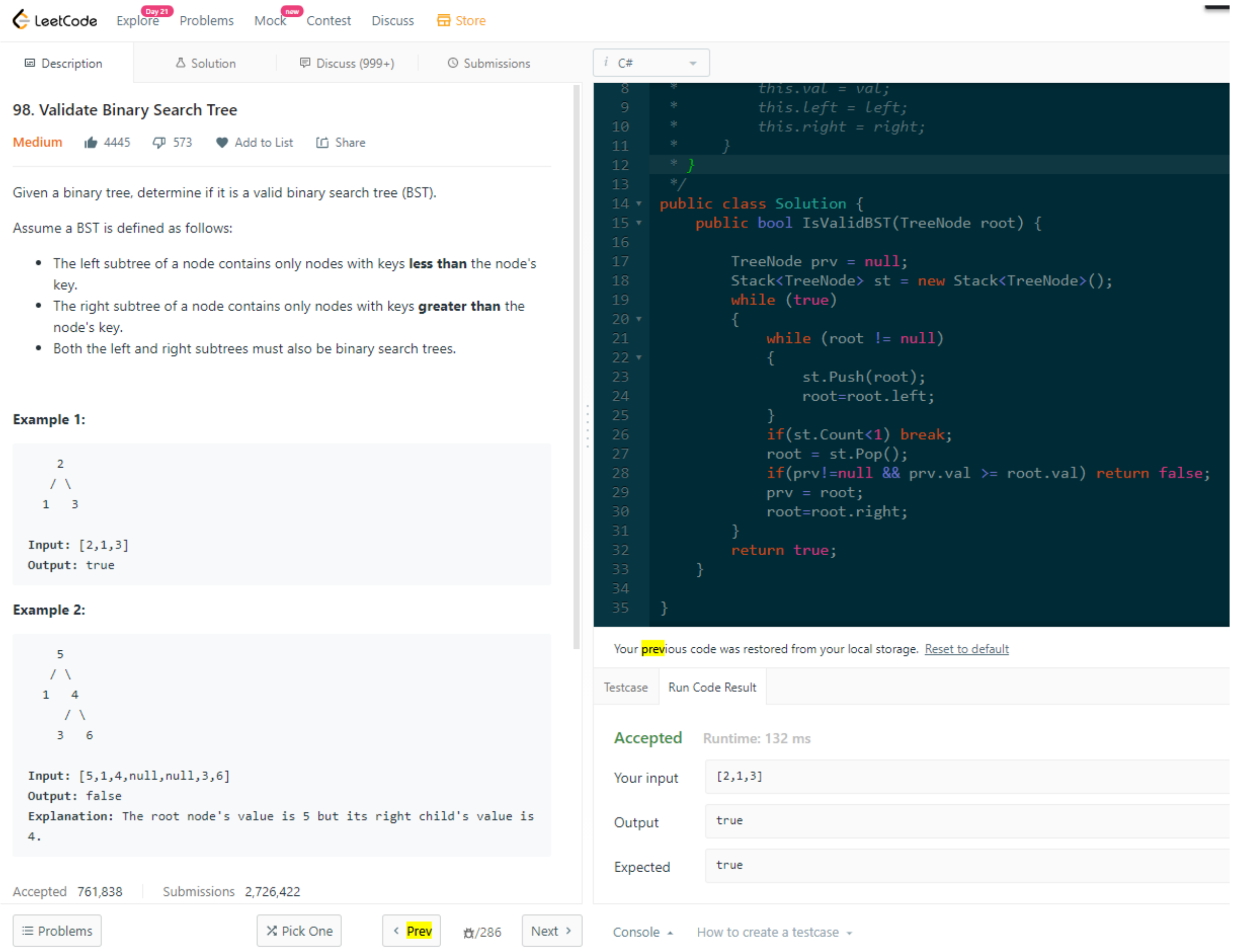
// https://leetcode.com/problems/validate-binary-search-tree/



98. Validate Binary Search Tree

Medium 4445 573 Add to List Share

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.

Example 1:

```
    2
   / \
  1   3

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

Example 2:

```
    5
   / \
  1   4
     / \
    3   6

Input: [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.
```

Accepted 761,838 Submissions 2,726,422

```
 *          this.val = val;
 *          this.left = left;
 *          this.right = right;
 *      }
 * }
 */
public class Solution {
    public bool IsValidBST(TreeNode root) {

        TreeNode prv = null;
        Stack<TreeNode> st = new Stack<TreeNode>();
        while (true)
        {
            while (root != null)
            {
                st.Push(root);
                root=root.left;
            }
            if(st.Count<1) break;
            root = st.Pop();
            if(prv!=null && prv.val >= root.val) return false;
            prv = root;
            root=root.right;
        }
        return true;
    }

}
```

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

Accepted Runtime: 132 ms

Your input [2,1,3]

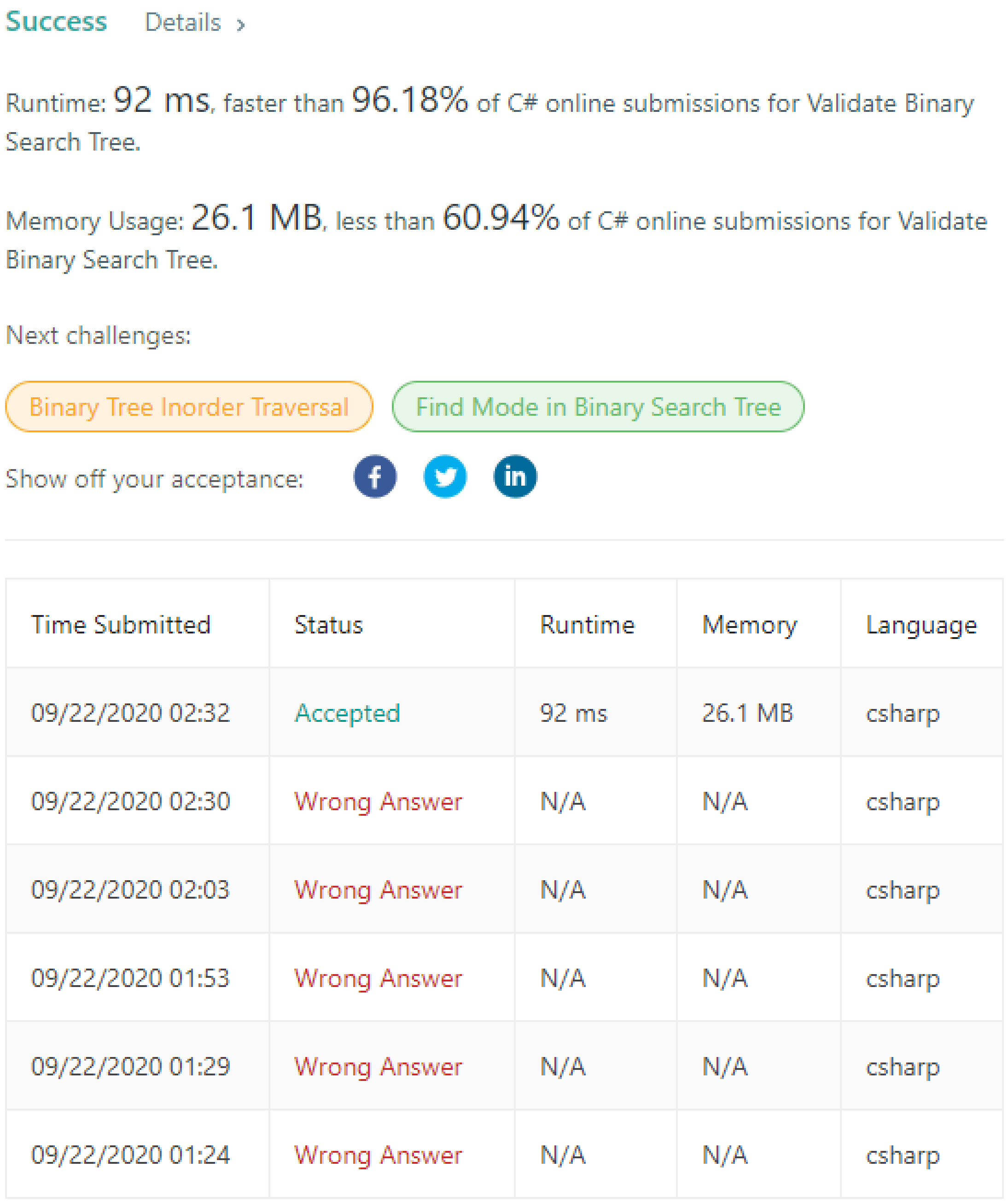
Output true

Expected true

```
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     public int val;
 *     public TreeNode left;
 *     public TreeNode right;
 *     public TreeNode(int val=0, TreeNode left=null, TreeNode right=null) {
 *         this.val = val;
 *         this.left = left;
 *         this.right = right;
 *     }
 * }
 */
public class Solution {
    public bool IsValidBST(TreeNode root) {

        TreeNode prv = null;
        Stack<TreeNode> st = new Stack<TreeNode>();
        while (true)
        {
            while (root != null)
            {
                st.Push(root);
                root=root.left;
            }
            if(st.Count<1) break;
            root = st.Pop();
            if(prv!=null && prv.val >= root.val) return false;
            prv = root;
            root=root.right;
        }
        return true;
    }

}
```



Runtime: 92 ms, faster than 96.18% of C# online submissions for Validate Binary Search Tree.

Memory Usage: 26.1 MB, less than 60.94% of C# online submissions for Validate Binary Search Tree.

Next challenges:

Binary Tree Inorder Traversal
Find Mode in Binary Search Tree

Show off your acceptance:

Time Submitted	Status	Runtime	Memory	Language
09/22/2020 02:32	Accepted	92 ms	26.1 MB	csharp
09/22/2020 02:30	Wrong Answer	N/A	N/A	csharp
09/22/2020 02:03	Wrong Answer	N/A	N/A	csharp
09/22/2020 01:53	Wrong Answer	N/A	N/A	csharp
09/22/2020 01:29	Wrong Answer	N/A	N/A	csharp
09/22/2020 01:24	Wrong Answer	N/A	N/A	csharp