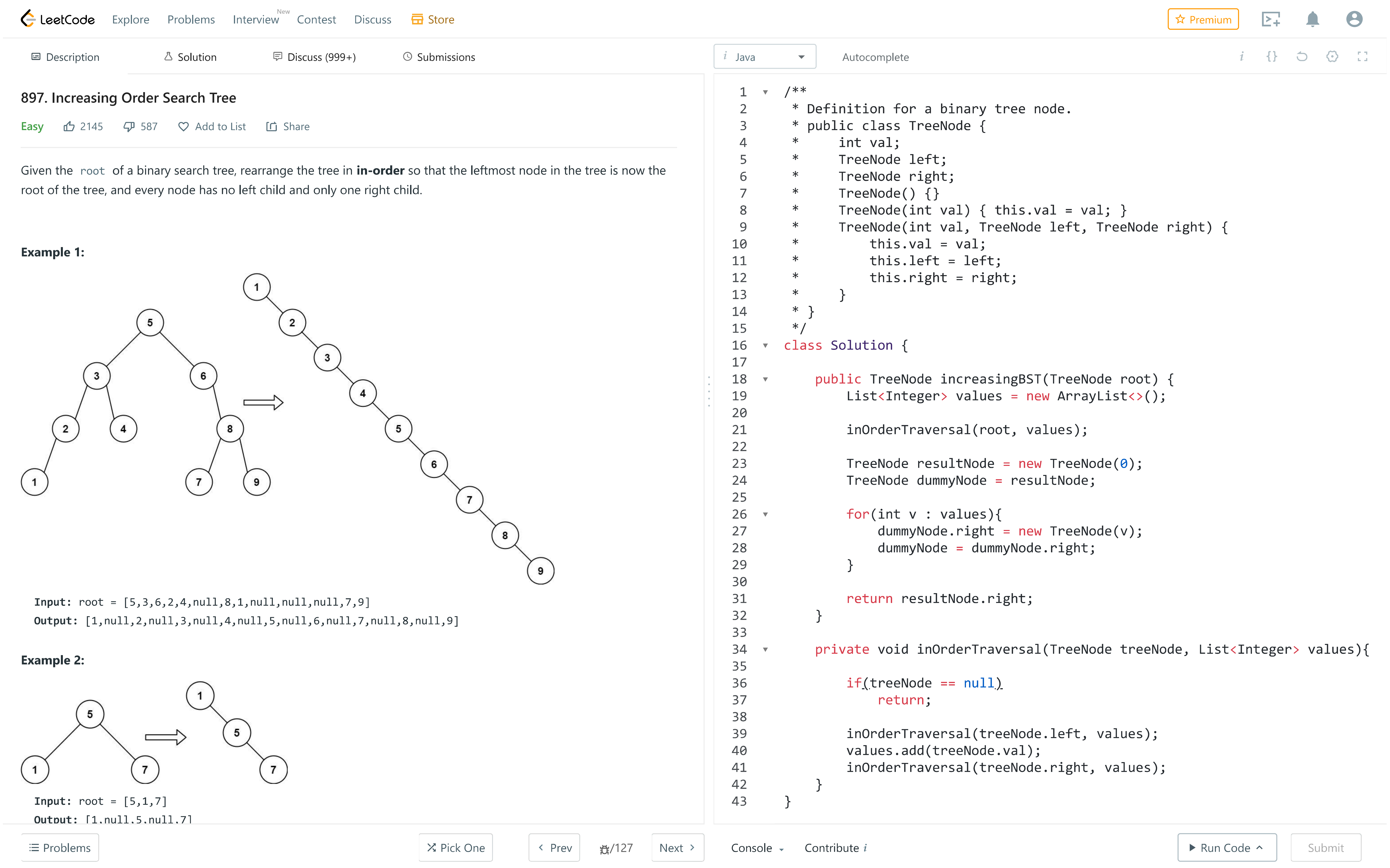
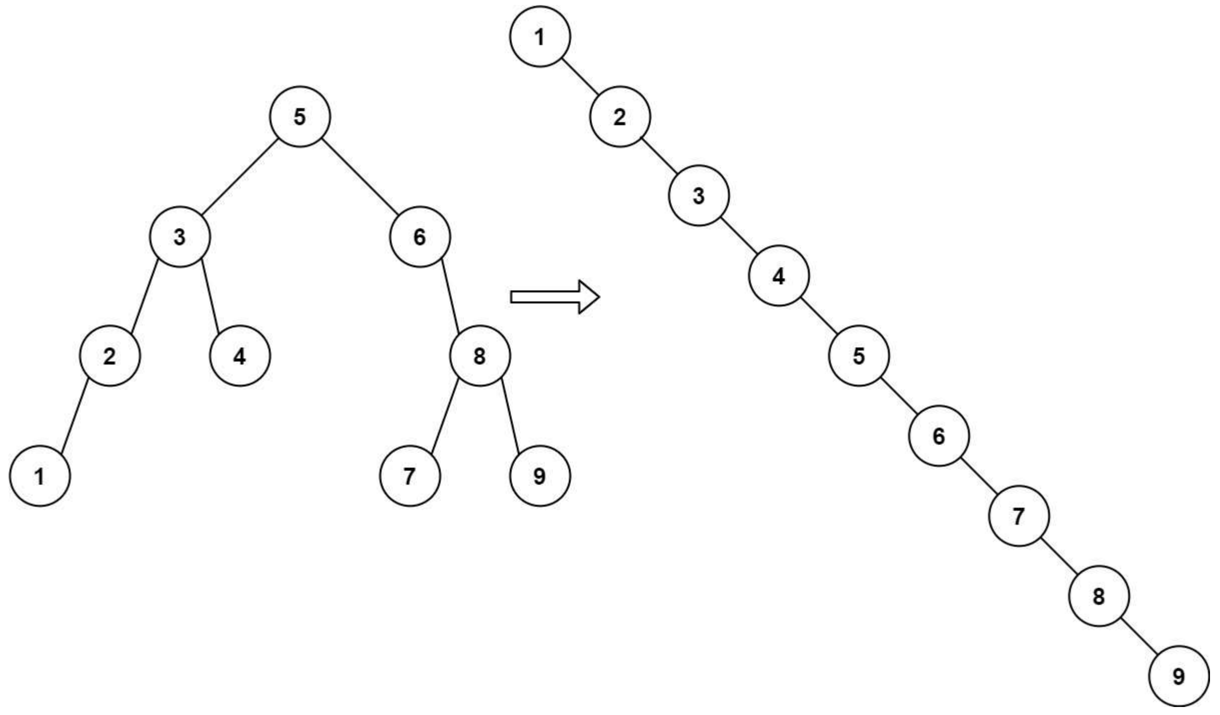


897. Increasing Order Search Tree

Easy 2145 587 Add to List Share

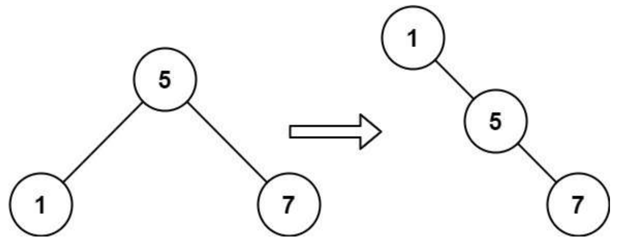
Given the `root` of a binary search tree, rearrange the tree in **in-order** so that the leftmost node in the tree is now the root of the tree, and every node has no left child and only one right child.

Example 1:



```
Input: root = [5,3,6,2,4,null,8,1,null,null,null,7,9]
Output: [1,null,2,null,3,null,4,null,5,null,6,null,7,null,8,null,9]
```

Example 2:



```
Input: root = [5,1,7]
Output: [1,null,5,null,7]
```

```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 {

    public TreeNode increasingBST(TreeNode root) {
        List<Integer> values = new ArrayList<>();

        inOrderTraversal(root, values);

        TreeNode resultNode = new TreeNode(0);
        TreeNode dummyNode = resultNode;

        for(int v : values){
            dummyNode.right = new TreeNode(v);
            dummyNode = dummyNode.right;
        }

        return resultNode.right;
    }

    private void inOrderTraversal(TreeNode treeNode, List<Integer> values){

        if(treeNode == null)
            return;

        inOrderTraversal(treeNode.left, values);
        values.add(treeNode.val);
        inOrderTraversal(treeNode.right, values);
    }
}
```