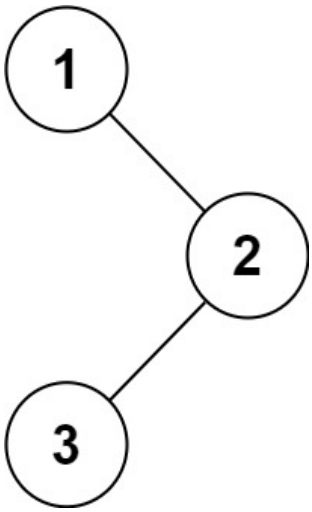
Leetcode Problem 2. (Easy)

Binary Tree Inorder Traversal

Given the root of a binary tree, return the inorder traversal of its nodes' values.

Example 1:



Input: root = [1,null,2,3]

Output: [1,3,2]

Example 2:

Input: root = []

Output: []

Example 3:

Input: root = [1]

Output: [1]

Constraints:

The number of nodes in the tree is in the range [0, 100].

-100 <= Node.val <= 100

Follow up: Recursive solution is trivial, could you do it iteratively?

Link:-

https://leetcode.com/problems/binary-tree-inorder-traversal/

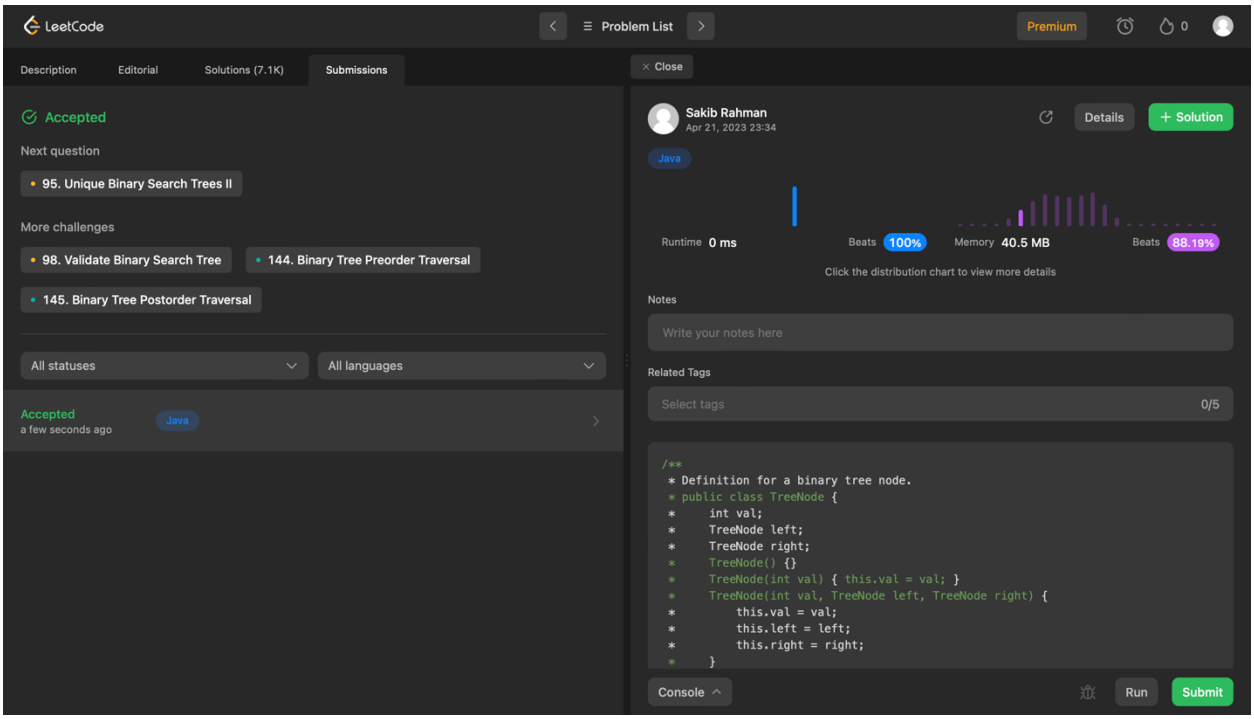
```
class Solution {
    public List<Integer> inorderTraversal(TreeNode root) {

    List<Integer> result = new ArrayList<>();
        Stack<TreeNode> stack = new Stack<>();

        while (root != null || !stack.isEmpty()) {
            while (root != null) {
                stack.push(root);
                root = root.left;
            }

            root = stack.pop();
            result.add(root.val);
            root = root.right;
        }

        return result;
    }
}
```


LeetCode
Problem List
Premium
0
Description
Editorial
Solutions (7.1K)
Submissions
Close
Accepted
Next question
95. Unique Binary Search Trees II
More challenges
98. Validate Binary Search Tree
144. Binary Tree Preorder Traversal
145. Binary Tree Postorder Traversal
All statuses
All languages
Accepted
a few seconds ago
Java
Sakib Rahman
Apr 21, 2023 23:34
Details
+ Solution
Java
Runtime 0 ms
Beats 100%
Memory 40.5 MB
Beats 88.19%
Click the distribution chart to view more details
Notes
Write your notes here
Related Tags
Select tags
0/5
/**
* 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;
*   }
Console
Run
Submit