**Experiment 2.1**

**Student Name:** TUSHAR GUPTA **UID:** 20BCS9268

**Branch:** CSE **Section/Group:** 607-A

**Semester:** 6th **Date of Performance:** 12/03/23

**Subject Name:** Competitive Coding-II **Subject Code:** 20CSP-351

**Aim:** To demonstrate the concept of trees

# OBJECTIVE 1: Same Tree

Given the roots of two binary trees p and q, write a function to check if they are the same or not.

Two binary trees are considered the same if they are structurally identical, and the nodes have the same value.

# Approach: Recursive approach

**Complexity Analysis**

# Time Complexity: O(N)

* **Space Complexity:** O(1)

# Code :

class Solution {

public static boolean isSameTree(TreeNode p, TreeNode q) { return tree(p,q);

}

public static boolean tree(TreeNode root1, TreeNode root2){

if(root1==null && root2==null){ return true;

}

if(root1==null){ return false;

}

if(root2==null){ return false;

}

if(root1.val!=root2.val){ return false;

}

boolean b1=tree(root1.left,root2.left); if(b1==false){

return false;

}

boolean b2=tree(root1.right,root2.right); if(b2==false){

return false;

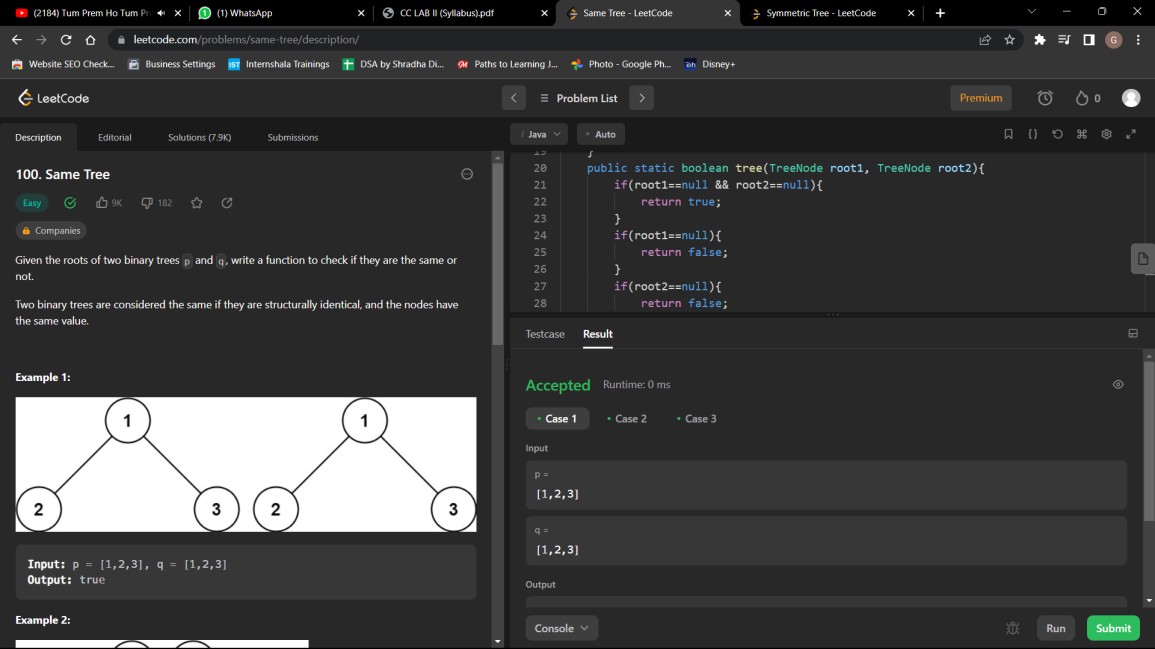
}

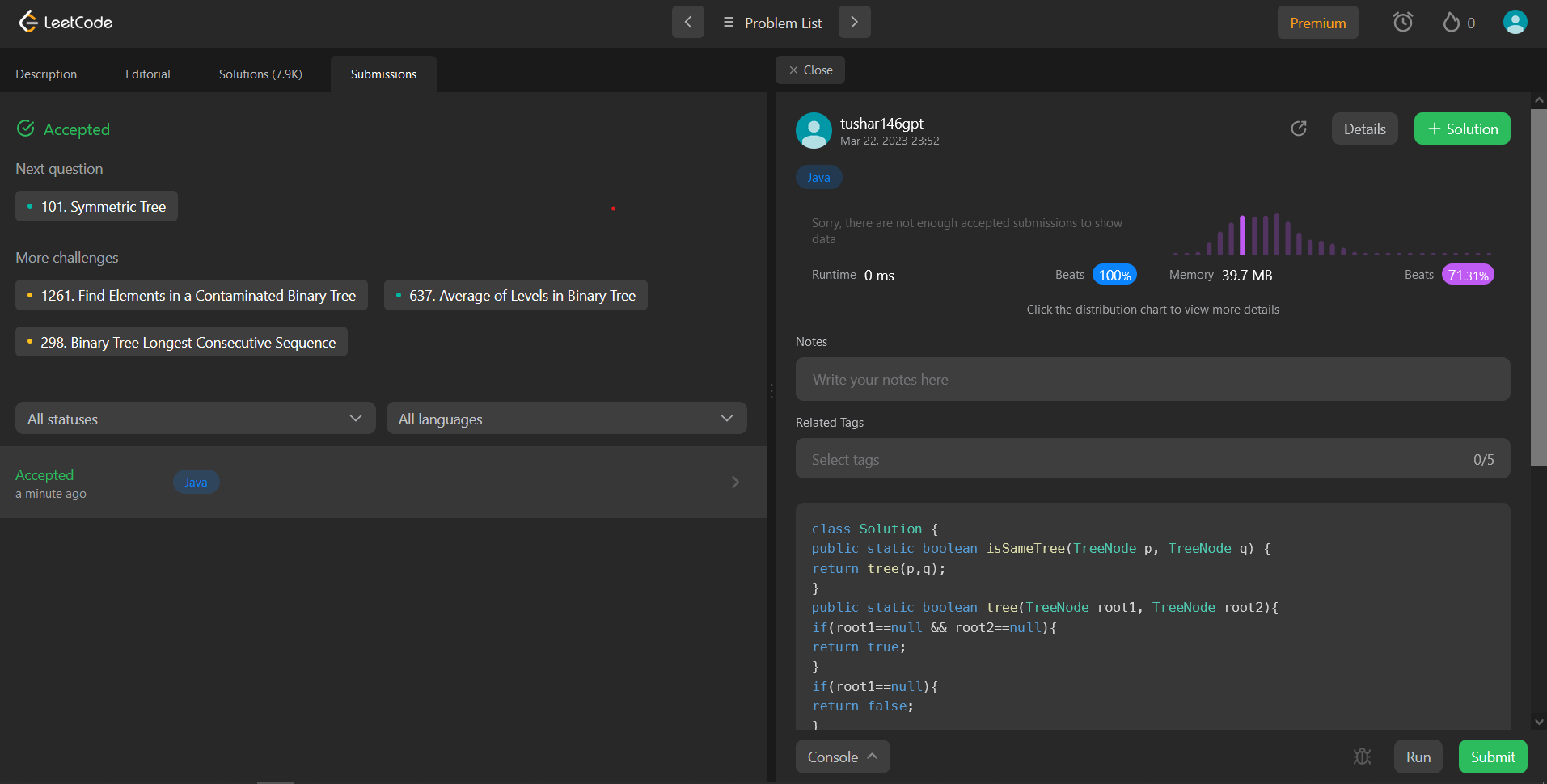
return true;

}

}

# Output:



****

**Question 2: Symmetric Tree**

# Approach: Recursive approach Complexity Analysis:

* Time Complexity: O(n)
* Space Complexity: O(n) Stack space

# Code:

class Solution {

public static boolean symmetric(TreeNode left,TreeNode right){ if(left==null || right==null){

return left==right;

}

if(left.val!=right.val){ return false;

}

return symmetric(left.left,right.right) && symmetric(left.right,right.left);

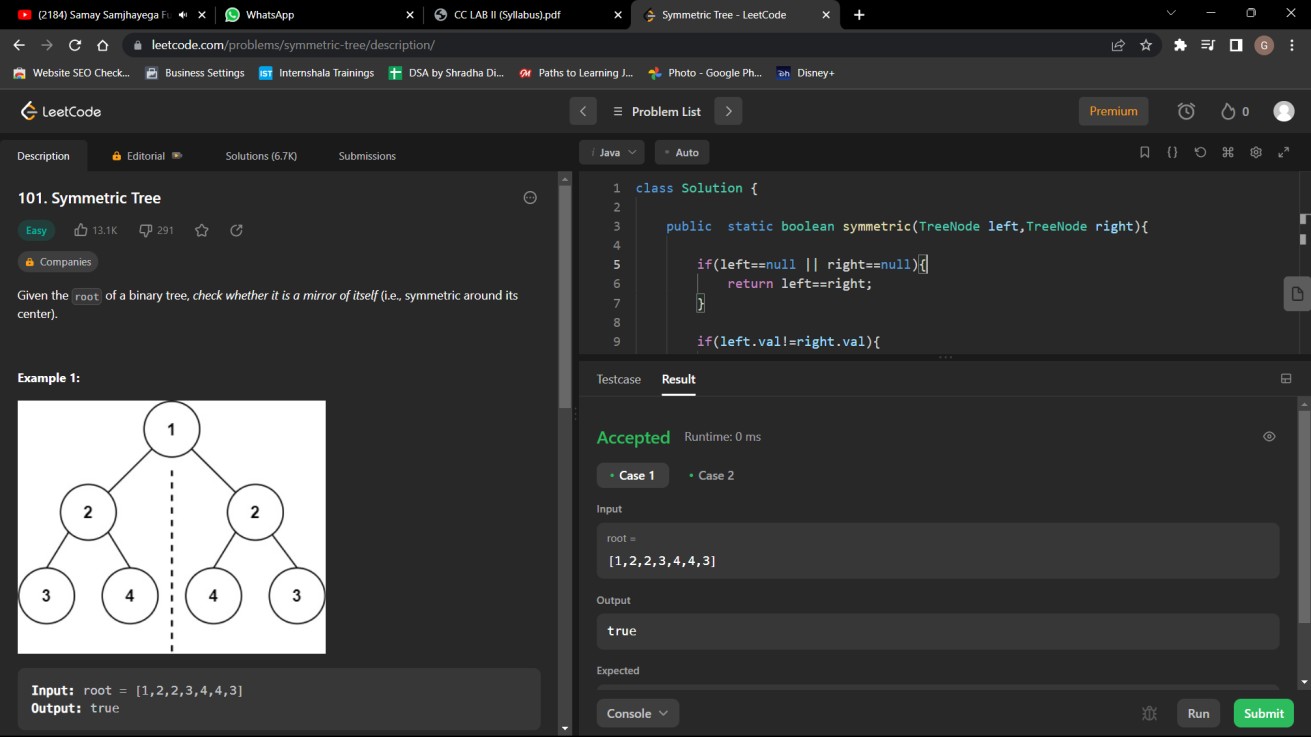
}

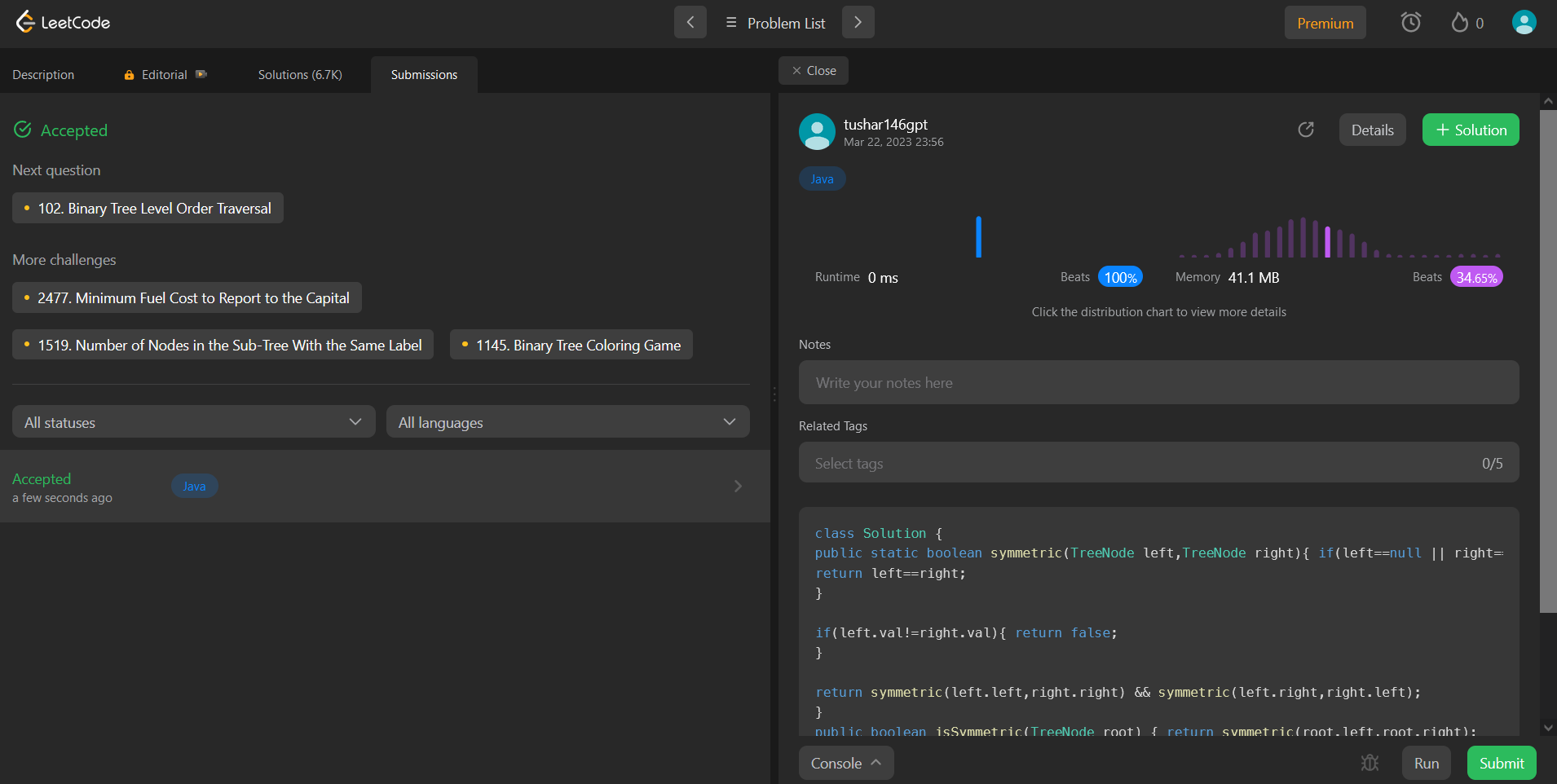
public boolean isSymmetric(TreeNode root) { return symmetric(root.left,root.right);

}

}

# Output:



****