



## Experiment-5

**Student Name:** MEGHA SHREE

**UID:** 22BCS10381

**Branch:** BE-CSE

**Section/Group:** KPIT\_901 (B)

**Semester:** 6<sup>th</sup>

**Date of Performance:** 27/02/2025

**Subject Name:** AP LAB-II

**Subject Code:** 22CSP-351

### Problem- 1

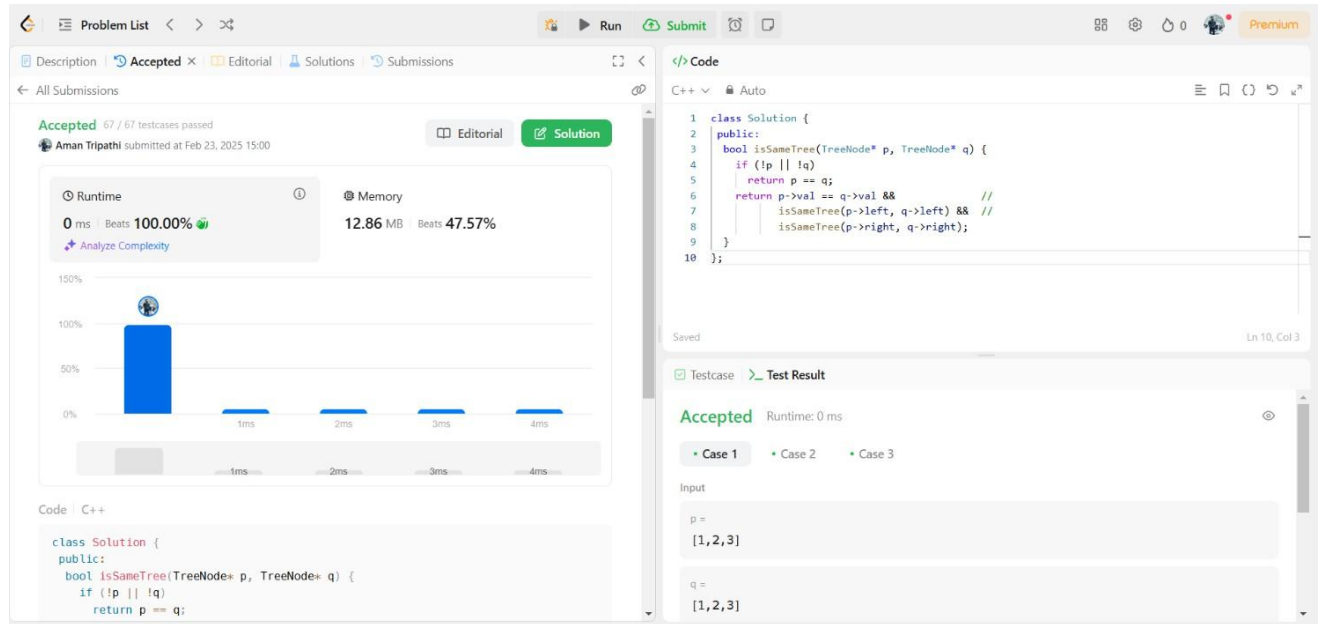
#### 1. Aim:

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.

#### 2. Implementation/Code: Backend:

```
class Solution { public:  
    bool isSameTree(TreeNode* p, TreeNode* q) { if  
        (!p || !q)  
            return p == q;  
        return p->val == q->val && isSameTree(p-  
            >left, q->left) && isSameTree(p-  
            >right, q->right);  
    }  
};
```

#### 3. Output:



## 4. Learning Outcomes:

- Understanding binary tree structure
- Implementing recursive tree traversal
- Comparing two trees for identical structure and values
- Handling edge cases like empty trees

## Problem- 2

### 1. Aim:

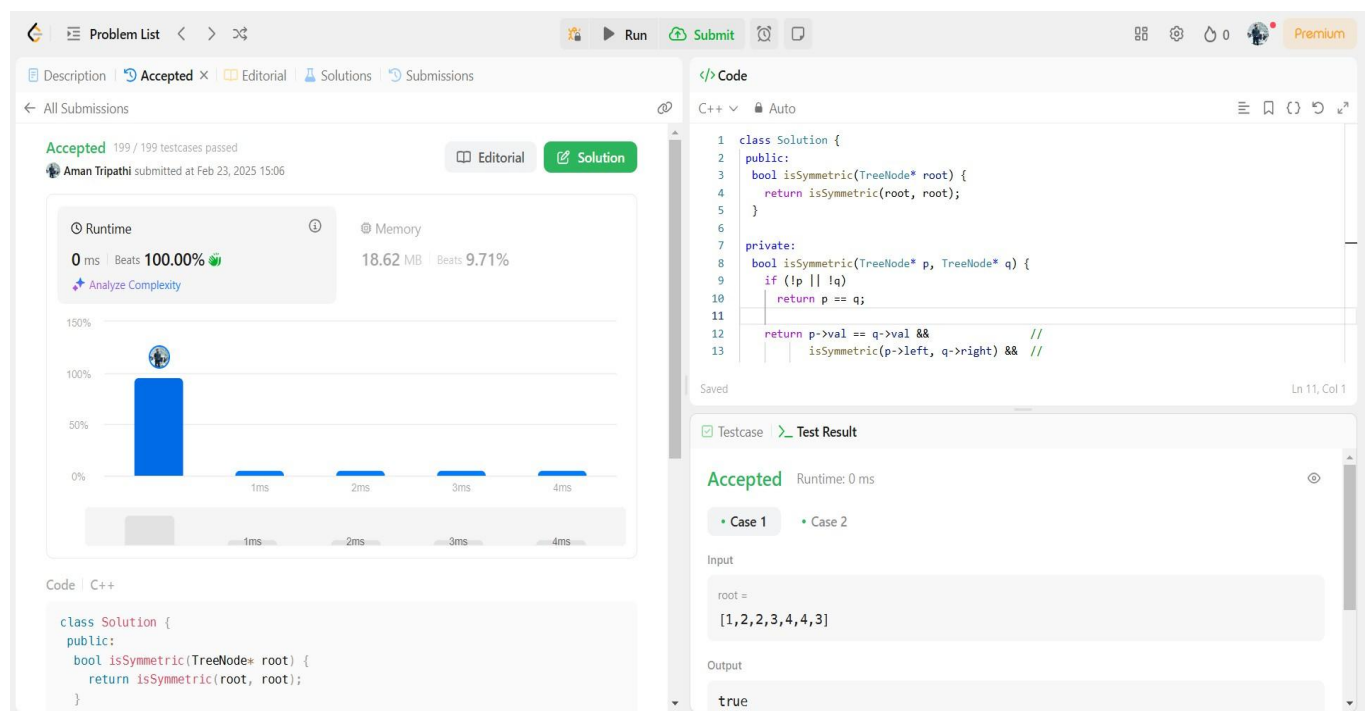
Given the root of a binary tree, check whether it is a mirror of itself (i.e., symmetric around its center).

### 2. Implementation/Code: Backend:

```
class Solution { public:
    bool isSymmetric(TreeNode* root) { return
        isSymmetric(root, root);
    }
private:
    bool isSymmetric(TreeNode* p, TreeNode* q) { if
        (!p || !q)
            return p == q;

        return p->val == q->val && //
            isSymmetric(p->left, q->right) && //
            isSymmetric(p->right, q->left);
    }
};
```

### 3. Output:



The screenshot displays a coding platform interface for a problem titled "Problem- 2". The solution is implemented in C++ and is shown in the "Code" editor. The code defines a class `Solution` with a public method `isSymmetric` and a private helper method. The public method calls the private method with the root of the tree. The private method checks for base cases (null nodes) and then recursively checks if the left child of one node is the right child of the other, and vice versa. The test result section shows that the solution is "Accepted" with a runtime of 0 ms. The input is a binary tree represented by the array [1,2,2,3,4,4,3], and the output is "true".

```
class Solution {
public:
    bool isSymmetric(TreeNode* root) {
        return isSymmetric(root, root);
    }
private:
    bool isSymmetric(TreeNode* p, TreeNode* q) {
        if (!p || !q) return p == q;
        return p->val == q->val &&
            isSymmetric(p->left, q->right) &&
            isSymmetric(p->right, q->left);
    }
};
```

Testcase: Accepted Runtime: 0 ms

Case 1 Case 2

Input: root = [1,2,2,3,4,4,3]

Output: true



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## 4. Learning Outcomes:

- Understanding binary tree structure
- Implementing recursion for tree traversal
- Checking symmetry using mirror property
- Handling edge cases like empty trees