

## AP Assignment-3

Name: Shivangi Gupta

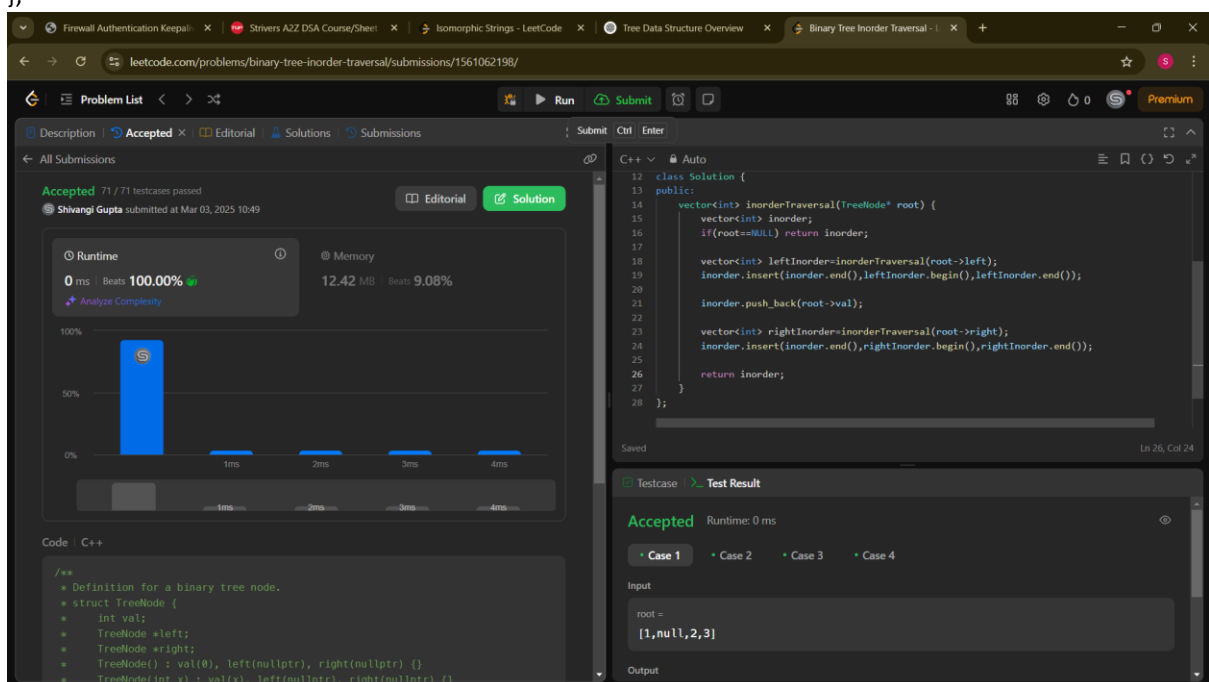
UID: 22BCS15008

Section: 601-A

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

Code:

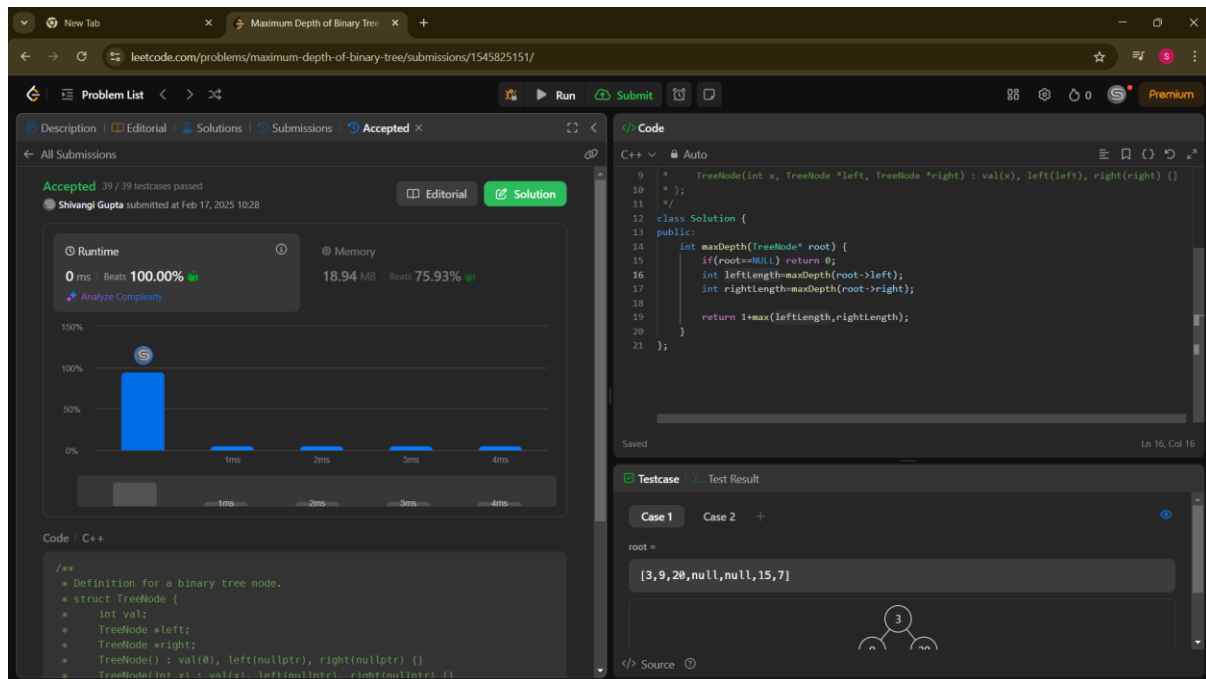
```
class Solution {  
public:  
    vector<int> inorderTraversal(TreeNode* root) {  
        vector<int> inorder;  
        if(root==NULL) return inorder;  
  
        vector<int> leftInorder=inorderTraversal(root->left);  
        inorder.insert(inorder.end(),leftInorder.begin(),leftInorder.end());  
  
        inorder.push_back(root->val);  
  
        vector<int> rightInorder=inorderTraversal(root->right);  
        inorder.insert(inorder.end(),rightInorder.begin(),rightInorder.end());  
  
        return inorder;  
    }  
};
```



104. <https://leetcode.com/problems/maximum-depth-of-binary-tree/submissions/1545825151/>

Code:

```
class Solution {  
  
public:  
  
    int maxDepth(TreeNode* root) {  
  
        if(root==NULL) return 0;  
  
        int leftLength=maxDepth(root->left);  
  
        int rightLength=maxDepth(root->right);  
  
  
        return 1+max(leftLength,rightLength);  
  
    }  
  
};
```



102. <https://leetcode.com/problems/binary-tree-level-order-traversal/description/>

Code:

```
class Solution {  
  
public:  
  
    vector<vector<int>> levelOrder(TreeNode* root) {  
  
        vector<vector<int>> levelOrderTraversal;  
  
        if(root==NULL) return levelOrderTraversal;
```

```

queue<TreeNode*> q;

q.push(root);

while(!q.empty()){

    vector<int> values;

    int level=q.size();

    for(int i=0;i<level;i++){

        TreeNode* node=q.front();

        q.pop();

        values.push_back(node->val);

        if(node->left!=NULL)

            q.push(node->left);

        if(node->right!=NULL)

            q.push(node->right);

    }

    levelOrderTraversal.push_back(values);

}

return levelOrderTraversal;

}

};

```

The screenshot shows a LeetCode submission for the problem "Binary Tree Level Order Traversal II". The submission is accepted, with a runtime of 0ms and memory usage of 17.22 MB. The code is in C++ and implements a BFS approach to traverse a binary tree level by level from bottom to top. The test case shows a root node with children 3, 9, 20, null, null, 15, 7, and a corresponding tree diagram.

107. <https://leetcode.com/problems/binary-tree-level-order-traversal-ii/description/>

Code:

```
class Solution {
```

```
public:
```

```
    vector<vector<int>> levelOrderBottom(TreeNode* root) {
```

```
        vector<vector<int>> levelOrderTraversal;
```

```
        if(root==NULL) return levelOrderTraversal;
```

```
        queue<TreeNode*> q;
```

```
        q.push(root);
```

```
        while(!q.empty()){
```

```
            vector<int> values;
```

```
            int level=q.size();
```

```
            for(int i=0;i<level;i++){
```

```
                TreeNode* node=q.front();
```

```
                q.pop();
```

```
                values.push_back(node->val);
```

```
                if(node->left!=NULL)
```

```
                    q.push(node->left);
```

```

        if(node->right!=NULL)
            q.push(node->right);
    }

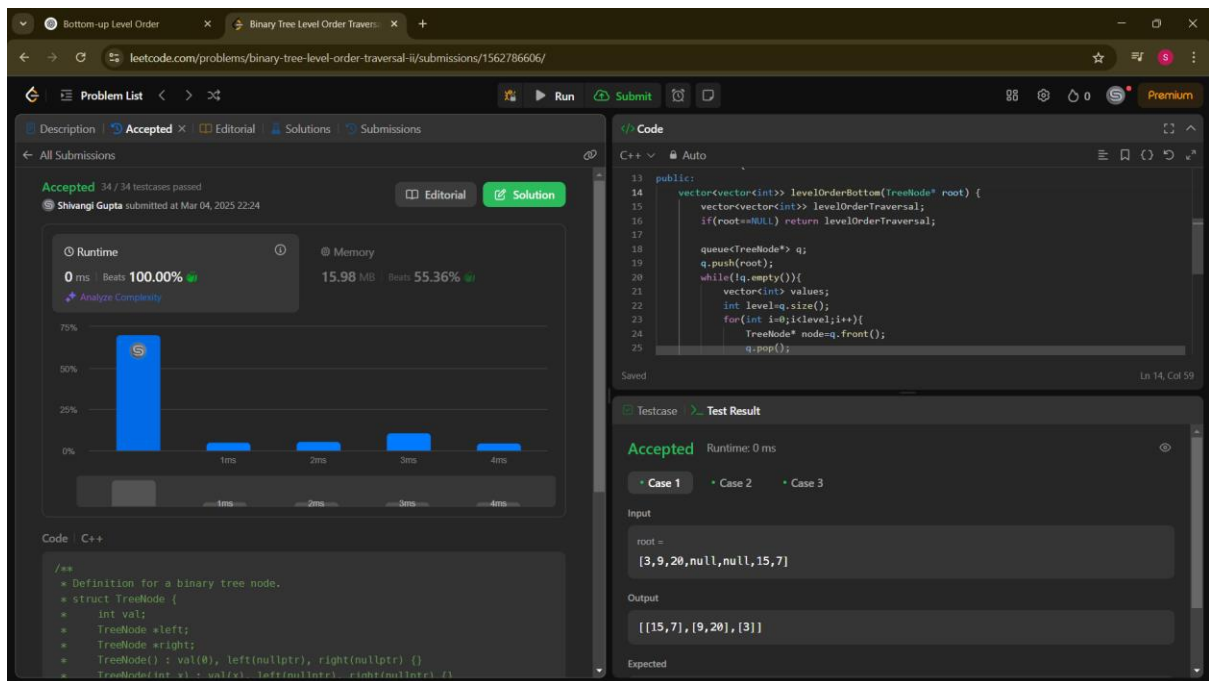
    levelOrderTraversal.push_back(values);
}

reverse(levelOrderTraversal.begin(),levelOrderTraversal.end());

return levelOrderTraversal;
}

};

```



101. <https://leetcode.com/problems/symmetric-tree/description/>

Code:

```

bool isMirror(TreeNode* t1, TreeNode* t2) {

    if (t1 == NULL && t2 == NULL) {

        return true;

    }

    if (t1 == NULL || t2 == NULL) {

        return false;
    }
}

```

Bottom-up Level Order

leetcode.com/problems/symmetric-tree/submissions/1562805302/

Problem List

Description Accepted Editorial Solutions Submissions

All Submissions

Accepted 200 / 200 testcases passed

Shivangi Gupta submitted at Mar 04, 2025 22:40

Editorial Solution

Runtime 0 ms Beats 100.00% Memory 18.60 MB Beats 9.52%

Analyze Complexity

100%

0ms 1ms 2ms 3ms 4ms

Code

```
C++ Auto
18 if (t1 == NULL || t2 == NULL) {
19     return false;
20 }
21 return (t1->val == t2->val) &&
22     isMirror(t1->left, t2->right) &&
23     isMirror(t1->right, t2->left);
24
25 bool isSymmetric(TreeNode* root) {
26     if (root == NULL) {
27         return true;
28     }
29     return isMirror(root->left, root->right);
30
31 }
```

Saved

Le 29, Col 50

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

root =

[1,2,2,3,4,4,3]

Output

true

Expected

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode() : val(0), left(nullptr), right(nullptr) {}
 *     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
 * }
```