



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Experiment 06

Student Name: Rajni Gandha

Branch: BE-IT

Semester: 06th

Subject Name: Advanced Programming

UID: 22BET10080

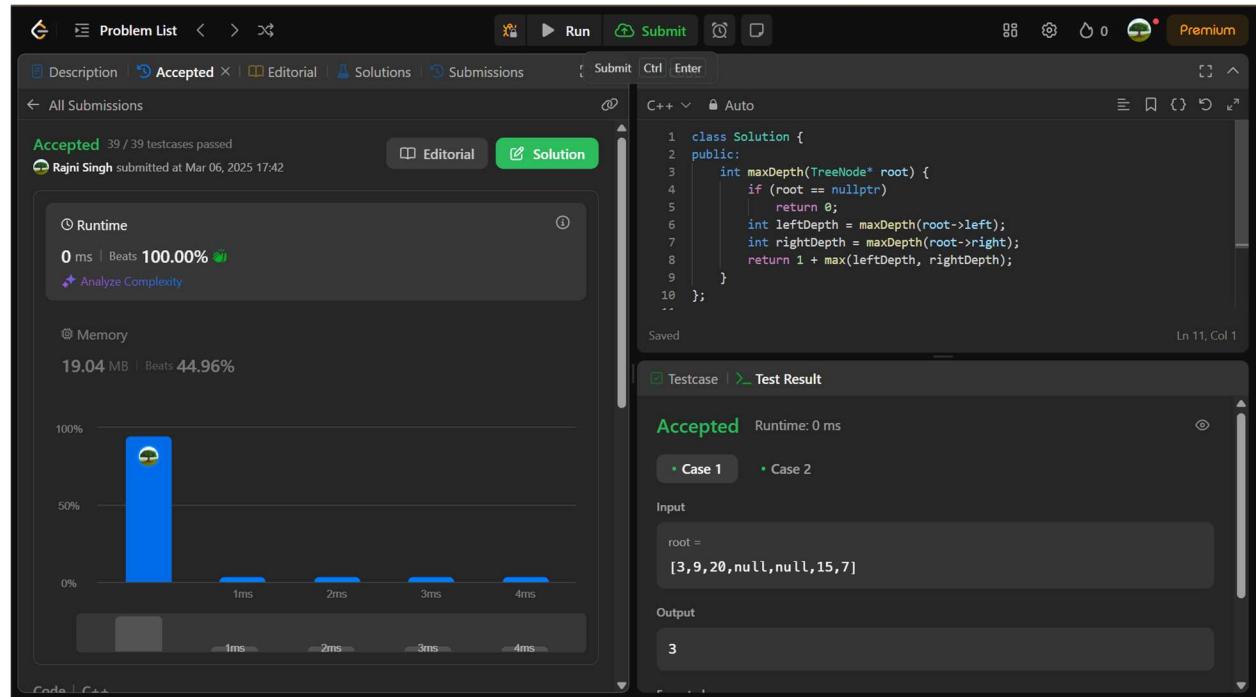
Section/Group: BET_701/A

Date of Performance: 05-03-2025

Subject Code: 22ITP-351

1. Problem: Maximum Depth of Binary Tree

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



2. Problem: Validate Binary Search Tree

```
class Solution {  
public:
```

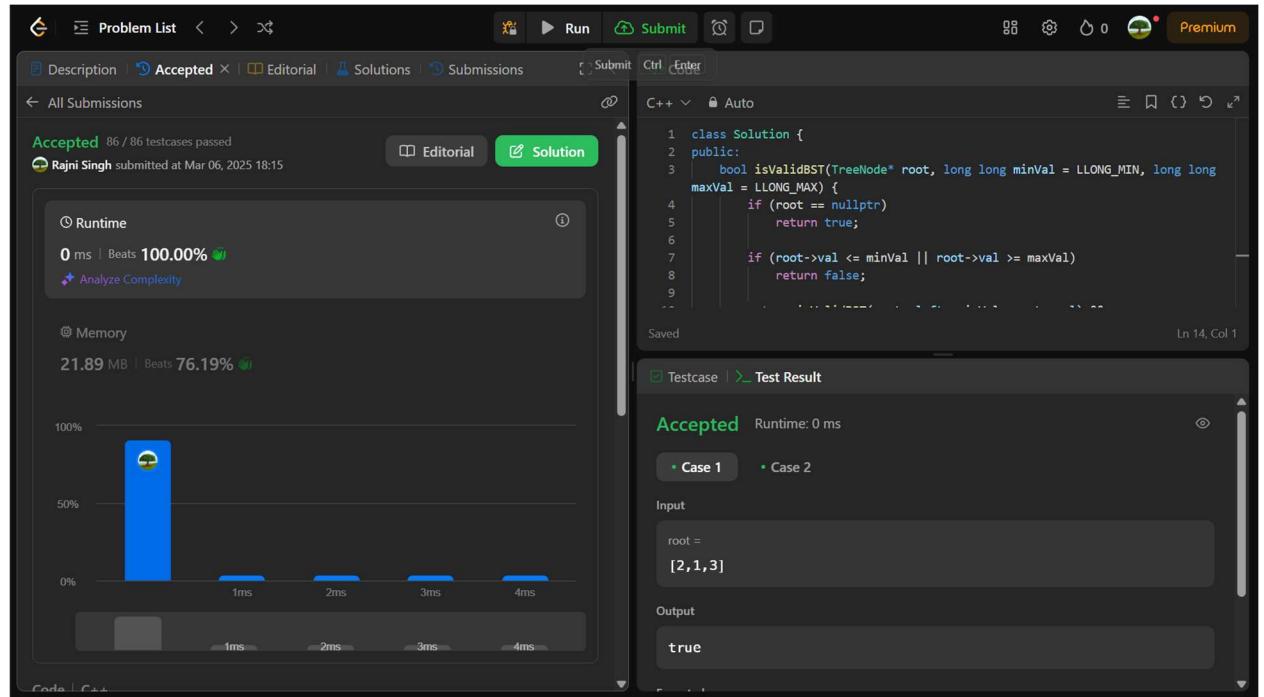


DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
bool isValidBST(TreeNode* root, long long minValue = LLONG_MIN, long long
maxValue = LLONG_MAX) {
    if (root == nullptr)
        return true;
    if (root->val <= minValue || root->val >= maxValue)
        return false;
    return isValidBST(root->left, minValue, root->val) &&
        isValidBST(root->right, root->val, maxValue);
}
```

```
};
```



3. Problem: Symmetric Tree

```
class Solution {
public:
    bool isMirror(TreeNode* t1, TreeNode* t2) {
        if (!t1 && !t2) return true;
        if (!t1 || !t2) return false;

        return (t1->val == t2->val) &&
            isMirror(t1->left, t2->right) &&
            isMirror(t1->right, t2->left);
    }
};
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
}
```

```
bool isSymmetric(TreeNode* root) {
    if (!root) return true;
    return isMirror(root->left, root->right);
}
```

```
};
```

Description | Accepted | Editorial | Solutions | Submissions | Submit | Ctrl + C Ctrl + V

All Submissions

Accepted 200 / 200 testcases passed

Rajni Singh submitted at Mar 06, 2025 17:55

Runtime 0 ms | Beats 100.00% Analyze Complexity

Memory 18.52 MB | Beats 27.15%

```
C++ v Auto
8     isMirror(t1->left, t2->right) &&
9     isMirror(t1->right, t2->left);
10 }
11
12 bool isSymmetric(TreeNode* root) {
13     if (!root) return true;
14     return isMirror(root->left, root->right);
15 }
16
17
```

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

```
root =
[1,2,2,3,4,4,3]
```

Output

```
true
```

4. Problem: Binary Tree Level Order Traversal

```
#include <vector>
#include <queue>
using namespace std;
```

```
class Solution {
public:
    vector<vector<int>> levelOrder(TreeNode* root) {
        vector<vector<int>> result;
        if (!root) return result;
        queue<TreeNode*> q;
        q.push(root);
        while (!q.empty()) {
            int levelSize = q.size();
            vector<int> level;
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
for (int i = 0; i < levelSize; i++) {  
    TreeNode* node = q.front();  
    q.pop();  
    level.push_back(node->val);  
  
    if (node->left) q.push(node->left);  
    if (node->right) q.push(node->right);  
}  
result.push_back(level);  
}  
return result;  
}  
};
```

The screenshot shows a code editor interface with a dark theme. On the left, there's a navigation bar with 'Problem List', 'Accepted' (highlighted), 'Editorial', 'Solutions', and 'Submissions'. Below it, a message says 'Accepted 35 / 35 testcases passed' by 'Rajni Singh' at 'Mar 06, 2025 17:58'. On the right, the 'Code' tab is active, displaying the C++ code for level-order traversal. The 'Runtime' section shows '0 ms | Beats 100.00%' with a green bar chart. The 'Memory' section shows '17.16 MB | Beats 44.39%' with a blue bar chart. The 'Test Result' section shows 'Accepted' with 'Runtime: 0 ms' and three test cases: Case 1 (Accepted), Case 2 (Accepted), and Case 3 (Accepted). The input is '[3,9,20,null,null,15,7]' and the output is '[[3], [9,20], [15,7]]'.

```
#include <vector>  
#include <queue>  
using namespace std;  
  
class Solution {  
public:  
    vector<vector<int>> levelOrder(TreeNode* root) {  
        vector<vector<int>> result;  
        if (!root) return result;  
        queue<TreeNode*> q;  
        q.push(root);
```

5. Problem: Convert Sorted Array to Binary Search Tree

```
class Solution {  
public:  
    TreeNode* sortedArrayToBST(vector<int>& nums) {  
        return buildBST(nums, 0, nums.size() - 1);  
    }
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

private:

```
TreeNode* buildBST(vector<int>& nums, int left, int right) {  
    if (left > right) return nullptr; // Base case  
  
    int mid = left + (right - left) / 2;  
    TreeNode* root = new TreeNode(nums[mid]);  
  
    root->left = buildBST(nums, left, mid - 1);  
    root->right = buildBST(nums, mid + 1, right);  
    return root;  
}  
};
```

The screenshot shows a LeetCode submission interface. The code has been accepted with 31/31 testcases passed by Rajni Singh at Mar 06, 2025 17:59. The runtime is 3 ms (Beats 59.83%) and the memory usage is 23.11 MB (Beats 12.51%). The code editor shows the C++ implementation of the buildBST function. The test results show an accepted status with runtime 0 ms for both Case 1 and Case 2. The input is [-10,-3,0,5,9] and the output is [0,-10,5,null,-3,null,9].

6. Problem: Binary Tree Inorder Traversal

```
class Solution {
```

```
public:
```

```
void inorder(TreeNode* root, vector<int>& result) {  
    if (!root) return;  
    inorder(root->left, result);  
    result.push_back(root->val);  
    inorder(root->right, result);
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

}

```
vector<int> inorderTraversal(TreeNode* root) {  
    vector<int> result;  
    inorder(root, result);  
    return result;  
}  
};
```

The screenshot shows a LeetCode submission page for a C++ solution. The code is pasted into the editor. The runtime statistics show 0 ms (Beats 100.00%) and memory usage of 10.68 MB (Beats 98.01%). The test results section shows all four test cases passed (Case 1, Case 2, Case 3, Case 4). The input is [1,null,2,3] and the output is [1,3,2].

```
vector<int> inorderTraversal(TreeNode* root) {  
    vector<int> result;  
    inorder(root, result);  
    return result;  
};
```

7. Problem: Construct Binary Tree from Inorder and Postorder Traversal

```
#include <vector>  
#include <unordered_map>  
using namespace std;
```

```
class Solution {  
public:  
    unordered_map<int, int> inorderMap;  
    int postIndex;
```

```
TreeNode* build(vector<int>& inorder, vector<int>& postorder, int left, int right) {  
    if (left > right) return nullptr; // Base case
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
int rootVal = postorder[postIndex--];
TreeNode* root = new TreeNode(rootVal);

int inIndex = inorderMap[rootVal];

root->right = build(inorder, postorder, inIndex + 1, right);
root->left = build(inorder, postorder, left, inIndex - 1);

return root;
}

TreeNode* buildTree(vector<int>& inorder, vector<int>& postorder) {
    postIndex = postorder.size() - 1; // Start from last element in postorder

    for (int i = 0; i < inorder.size(); i++) {
        inorderMap[inorder[i]] = i;
    }

    return build(inorder, postorder, 0, inorder.size() - 1);
}
};
```

The screenshot shows a code editor interface with the following details:

- Accepted:** 202 / 202 testcases passed by Rajni Singh at Mar 06, 2025 18:04.
- Runtime:** 0 ms | Beats 100.00%.
- Memory:** 27.45 MB | Beats 59.01%.
- Code Snippet (C++):**

```
// Store inorder indices for quick lookup
for (int i = 0; i < inorder.size(); i++) {
    inorderMap[inorder[i]] = i;
}
return build(inorder, postorder, 0, inorder.size() - 1);
```
- Test Result:** Accepted | Testcase 1, Case 1, Case 2.
- Input:** inorder = [9,3,15,20,7], postorder = [9,15,7,20,3].
- Output:** (empty)



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

8. Problem: Kth Smallest element in a BST

```
class Solution {
public:
    int count = 0, result = -1;

    void inorder(TreeNode* root, int k) {
        if (!root) return;

        inorder(root->left, k);
        count++;
        if (count == k) {
            result = root->val;
            return;
        }
        inorder(root->right, k);
    }

    int kthSmallest(TreeNode* root, int k) {
        inorder(root, k);
        return result;
    }
};
```

The screenshot shows a code editor interface with the following details:

- Code Area:** The code is written in C++ and defines a `Solution` class with methods `inorder` and `kthSmallest`.
- Runtime Statistics:** The code has a runtime of 3 ms, which beats 8.88% of submissions.
- Memory Usage:** The code uses 24.47 MB of memory, which beats 42.84% of submissions.
- Testcase Results:** The code has been accepted for all testcases, with a runtime of 0 ms.
- Input:** The input for the `kthSmallest` function is `root = [3,1,4,null,2]` and `k = 1`.
- Output:** The output is not explicitly shown in the screenshot.



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

9. Problem: Populating Next Right Pointers in Each Node

```
class Solution {
public:
    Node* connect(Node* root) {
        if (!root) return nullptr;
        queue<Node*> q;
        q.push(root);
        while (!q.empty()) {
            int size = q.size();
            for (int i = 0; i < size; i++) {
                Node* node = q.front();
                q.pop();
                if (i < size - 1) {
                    node->next = q.front();
                }
                if (node->left) q.push(node->left);
                if (node->right) q.push(node->right);
            }
        }
        return root;
    }
};
```

The screenshot shows a LeetCode submission interface. The top bar includes navigation links like 'Problem List', 'Run', 'Submit', and 'Premium'. The main area displays the problem description, which is already solved ('Accepted'). It shows the author's name, Rajni Singh, and the submission time, Mar 06, 2025 18:09. Below this, the 'Runtime' section shows a bar chart with a single bar at 8 ms, labeled 'Beats 92.17%'. The 'Memory' section shows a bar chart with a single bar at 19.13 MB, labeled 'Beats 49.66%'. The 'Code' section contains the C++ implementation provided above. The 'Test Result' section shows the code was accepted with a runtime of 0 ms, passing both Case 1 and Case 2. The input and output fields show the expected values for the problem.