Name: Mayank Singh

UID: 22BCS10205

Section: 22BCS_IOT-612-B

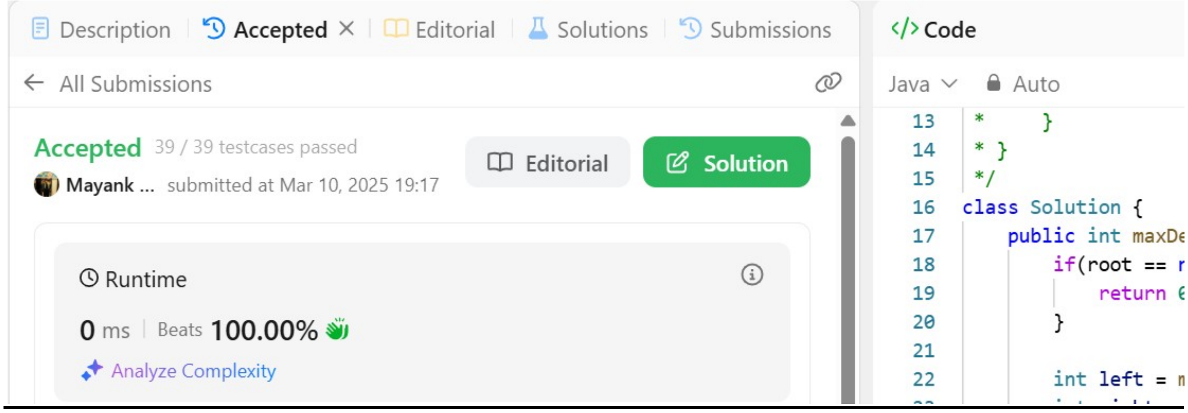
Subject: Advanced Programming Lab-2

Assignment – 5

1. Code: (Maximum Depth of Binary Tree)

```
class Solution {
    public int maxDepth(TreeNode root) {
        if(root == null) {
            return 0;
    }
        int left = maxDepth(root.left);
        int right = maxDepth(root.right);

        return Math.max(left, right) + 1;
    }
}
```

Output:



Accepted 39 / 39 testcases passed

Mayank ... submitted at Mar 10, 2025 19:17

Runtime

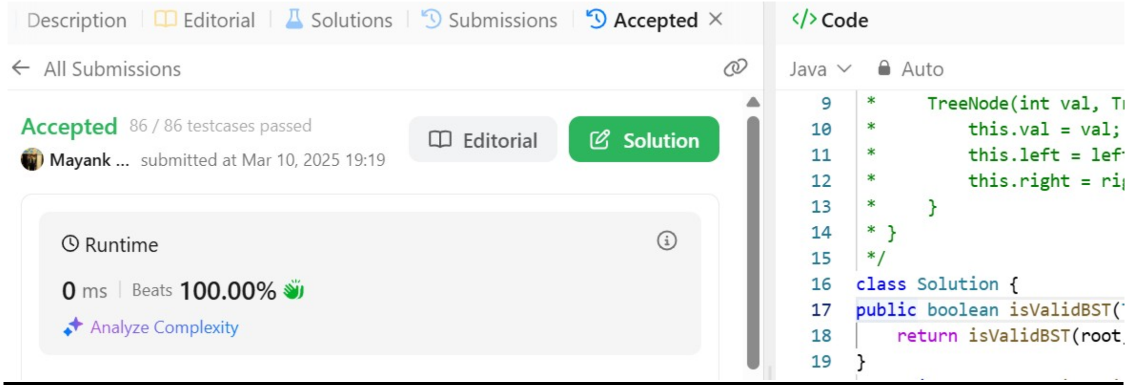
0 ms | Beats 100.00%

```
 *     }
 * }
 */
class Solution {
    public int maxDe
        if(root == r
            return 0
        }

        int left = m
```

2. Code: (Validate Binary Search Tree)

```
public boolean isValidBST(TreeNode root) {
    return isValidBST(root, Double.NEGATIVE_INFINITY, Double.POSITIVE_INFINITY);
}

private boolean isValidBST(TreeNode node, double min, double max) {
    if (node == null) {
        return true;
    }

    if (node.val <= min || node.val >= max) {
        return false;
    }

    return isValidBST(node.left, min, node.val) && isValidBST(node.right, node.val, max);
}
```

OUTPUT:



Accepted 86 / 86 testcases passed

Mayank ... submitted at Mar 10, 2025 19:19

Runtime

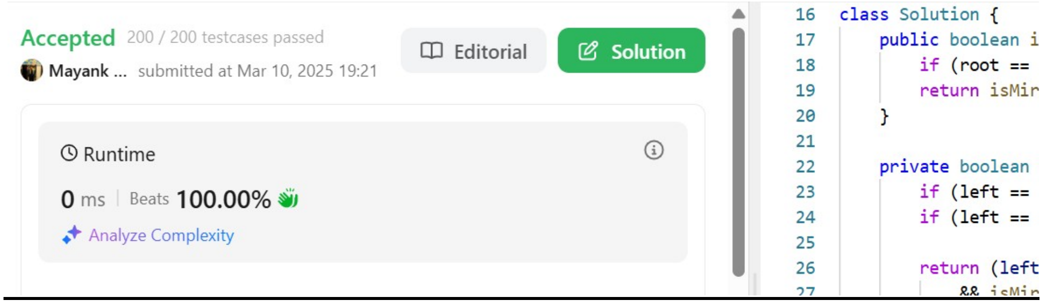
0 ms | Beats 100.00%

```
 *     TreeNode(int val, T
 *         this.val = val;
 *         this.left = lef
 *         this.right = ri
 *     }
 * }
 */
class Solution {
public boolean isValidBST(
    return isValidBST(root,
}
```

3. Code: (Symmetric Tree)

```
class Solution {
    public boolean isSymmetric(TreeNode root) {
        if (root == null) return true;
        return isMirror(root.left, root.right);
    }

    private boolean isMirror(TreeNode left, TreeNode right) {
        if (left == null && right == null) return true;
        if (left == null || right == null) return false;

        return (left.val == right.val)
            && isMirror(left.left, right.right)
            && isMirror(left.right, right.left);
    }
}
```

Output:



Accepted 200 / 200 testcases passed

Mayank ... submitted at Mar 10, 2025 19:21

Runtime

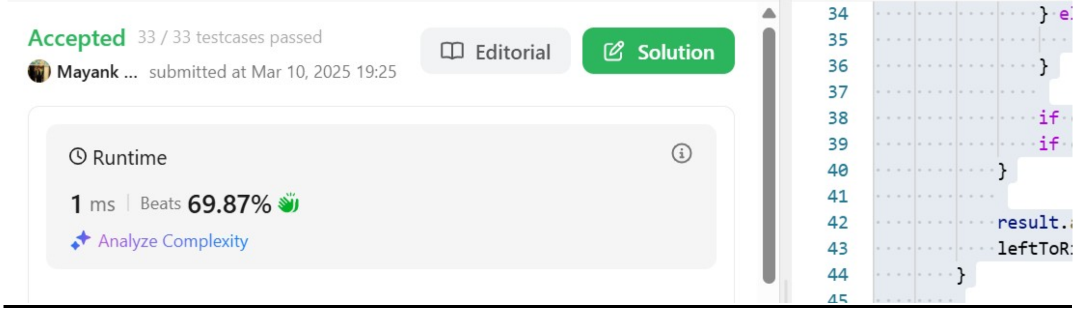
0 ms | Beats 100.00%

```
class Solution {
    public boolean i
        if (root ==
        return isMir
    }

    private boolean
        if (left ==
        if (left ==

        return (left
            && isMir
```

4. Code: (Binary Tree Zigzag Level Order Traversal)

```
class Solution {
    public List<List<Integer>> zigzagLevelOrder(TreeNode root) {
        List<List<Integer>> result = new ArrayList<>();
        if (root == null) return result;

        Deque<TreeNode> deque = new LinkedList<>();
        deque.offer(root);
        boolean leftToRight = true;

        while (!deque.isEmpty()) {
            int levelSize = deque.size();
            List<Integer> currentLevel = new ArrayList<>();

            for (int i = 0; i < levelSize; i++) {
                TreeNode node = deque.poll();

                if (leftToRight) {
                    currentLevel.add(node.val);
                } else {
                    currentLevel.add(0, node.val);
                }

                if (node.left != null) deque.offer(node.left);
                if (node.right != null) deque.offer(node.right);
            }

            result.add(currentLevel);
            leftToRight = !leftToRight;
        }

        return result;
    }
}
```

Output:



Accepted 33 / 33 testcases passed

Mayank ... submitted at Mar 10, 2025 19:25

Runtime

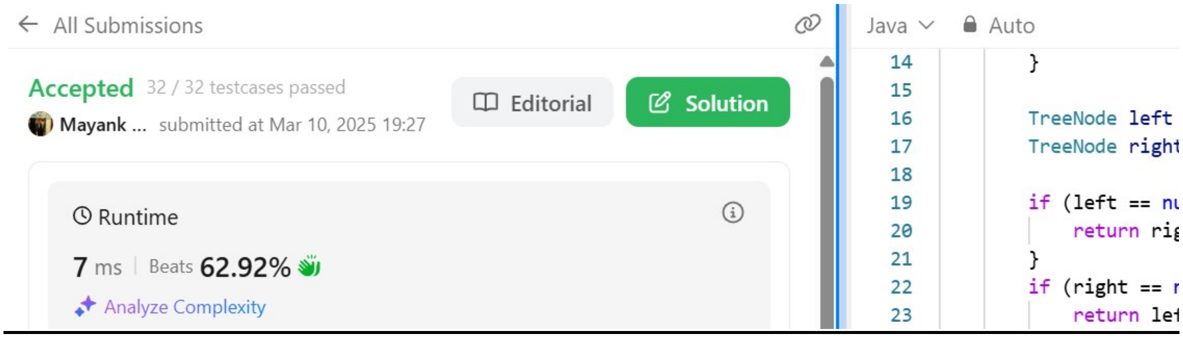
1 ms | Beats 69.87%

```
                } e
                }
                if
                if
            }

            result.
            leftToR
        }
```

5. Code: (Lowest Common Ancestor of a Binary Tree)

```
class Solution {
    public TreeNode lowestCommonAncestor(TreeNode root, TreeNode p, TreeNode q) {
        if (root == null || root == p || root == q) {
            return root;
        }

        TreeNode left = lowestCommonAncestor(root.left, p, q);
        TreeNode right = lowestCommonAncestor(root.right, p, q);

        if (left == null) {
            return right;
        }
        if (right == null) {
            return left;
        }

        return root;
    }
}
```

Output:



Accepted 32 / 32 testcases passed

Mayank ... submitted at Mar 10, 2025 19:27

Runtime

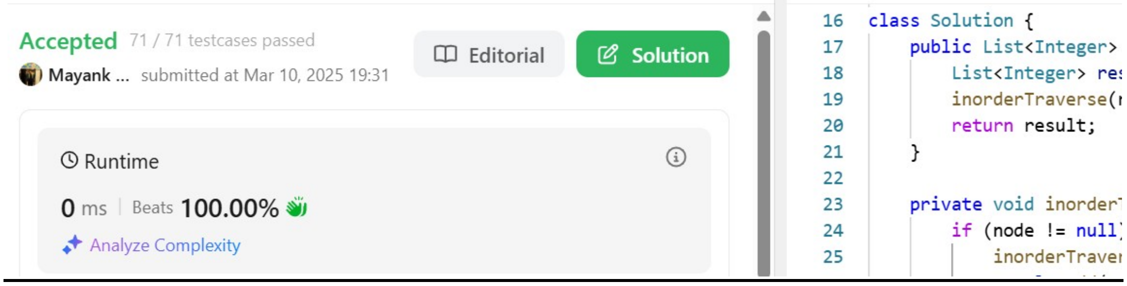
7 ms | Beats 62.92%

```
        }

        TreeNode left
        TreeNode right

        if (left == nu
            return rig
        }
        if (right == r
            return lef
```

6. Code: (Binary Tree Inorder Traversal)

```
class Solution {
    public List<Integer> inorderTraversal(TreeNode root) {
        List<Integer> result = new ArrayList<>();
        inorderTraverse(root, result);
        return result;
    }

    private void inorderTraverse(TreeNode node, List<Integer> result) {
        if (node != null) {
            inorderTraverse(node.left, result);
            result.add(node.val);
            inorderTraverse(node.right, result);
        }
    }
}
```

Output:



Accepted 71 / 71 testcases passed

Mayank ... submitted at Mar 10, 2025 19:31

Runtime

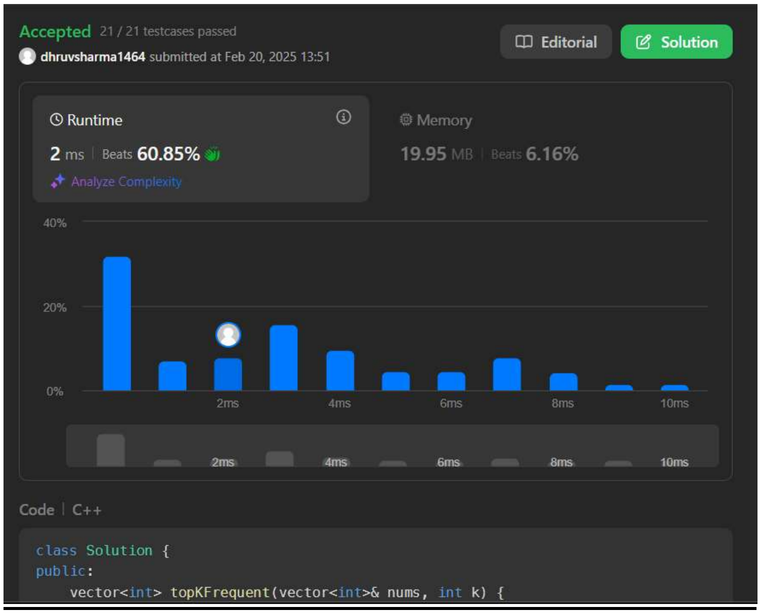
0 ms | Beats 100.00%

```
class Solution {
    public List<Integer>
        List<Integer> res
        inorderTraverse(r
        return result;
    }

    private void inorder
        if (node != null)
            inorderTraver
```

7. Code: (Binary Tree Level Order Traversal)

```
class Solution {
public:
    vector<int> topKFrequent(vector<int>& nums, int k) {
        unordered_map<int, int> freqMap;
        for (int num : nums) {
            freqMap[num]++;
        }
        vector<vector<int>> buckets(nums.size() + 1);
        for (auto& [num, freq] : freqMap) {
            buckets[freq].push_back(num);
        }
        vector<int> result;
        for (int i = nums.size(); i > 0 && result.size() < k; i--) {
            for (int num : buckets[i]) {
                result.push_back(num);
                if (result.size() == k) return result;
            }
        }
        return result;
    }
};
```

Output:



Accepted 21 / 21 testcases passed

dhruvsharma1464 submitted at Feb 20, 2025 13:51

Runtime 2 ms | Beats 60.85%

Memory 19.95 MB | Beats 6.16%

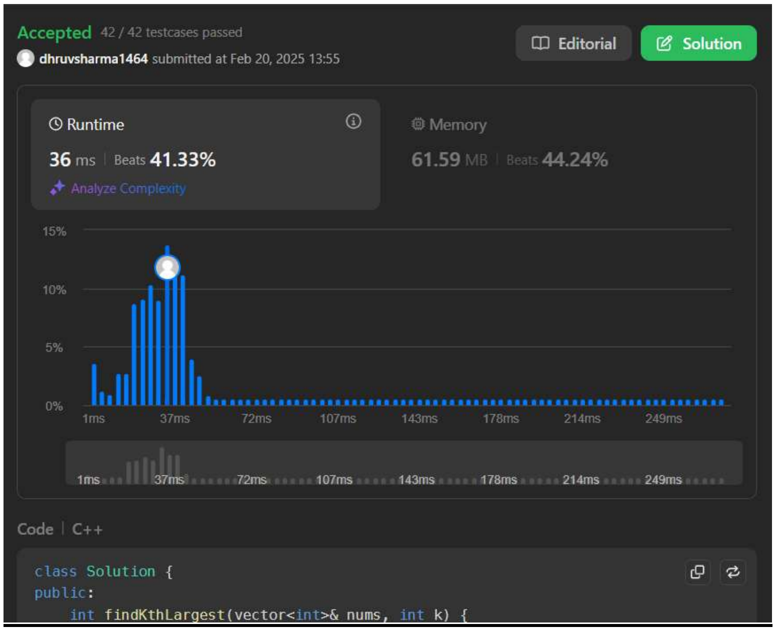
Code | C++

```
class Solution {
public:
    vector<int> topKFrequent(vector<int>& nums, int k) {
```

8. Code: (Kth Smallest Element in a BST)

```
class Solution {
public:
    int findKthLargest(vector<int>& nums, int k) {
        priority_queue<int, vector<int>, greater<int>> minHeap;
        for (int num : nums) {
            minHeap.push(num);
            if (minHeap.size() > k) {
                minHeap.pop();
            }
        }
        return minHeap.top();
    }
};
```

Output:



Accepted 42 / 42 testcases passed

dhruvsharma1464 submitted at Feb 20, 2025 13:55

Runtime 36 ms | Beats 41.33%

Memory 61.59 MB | Beats 44.24%

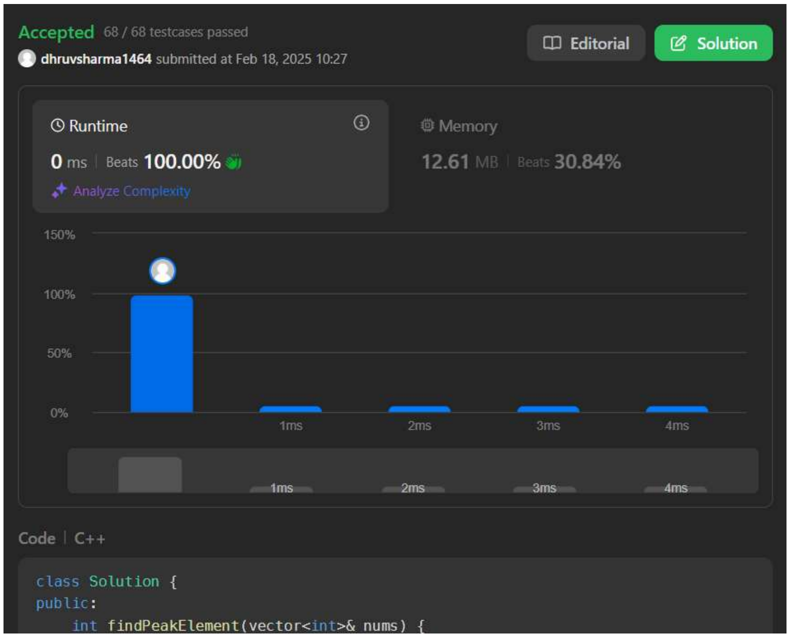
Code | C++

```
class Solution {
public:
    int findKthLargest(vector<int>& nums, int k) {
```

9. Code: (Populating Next Right Pointers in Each Node)

```
class Solution {
public:
    int findPeakElement(vector<int>& nums) {
        int left = 0, right = nums.size() - 1;
        while (left < right) {
            int mid = left + (right - left) / 2;
            if (nums[mid] > nums[mid + 1]) {
                right = mid;
            } else {
                left = mid + 1;
            }
        }
        return left;
    }
};
```

Output:



Accepted 68 / 68 testcases passed
dhruvsharma1464 submitted at Feb 18, 2025 10:27

0 ms | Beats 100.00%

12.61 MB | Beats 30.84%

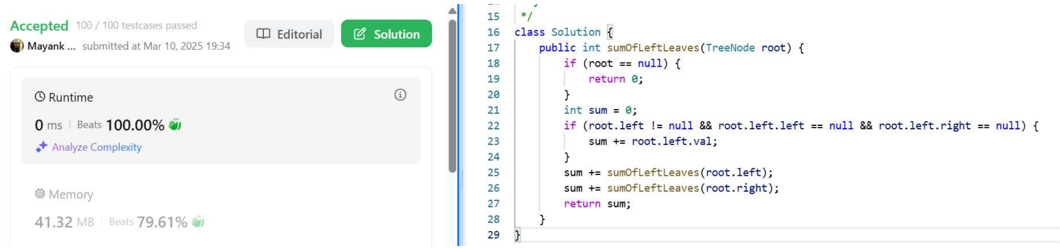
Code | C++

```
class Solution {
public:
    int findPeakElement(vector<int>& nums) {
```

10. Code: (Sum of Left Leaves)

```
class Solution {
    public int sumOfLeftLeaves(TreeNode root) {
        if (root == null) {
            return 0;
        }
        int sum = 0;
        if (root.left != null && root.left.left == null && root.left.right == null) {
            sum += root.left.val;
        }
        sum += sumOfLeftLeaves(root.left);
        sum += sumOfLeftLeaves(root.right);
        return sum;
    }
}
```

Output:



Accepted 100 / 100 testcases passed
Mayank ... submitted at Mar 10, 2025 19:34

0 ms | Beats 100.00%

41.32 MB | Beats 79.61%

```
 */
class Solution {
    public int sumOfLeftLeaves(TreeNode root) {
        if (root == null) {
            return 0;
        }
        int sum = 0;
        if (root.left != null && root.left.left == null && root.left.right == null) {
            sum += root.left.val;
        }
        sum += sumOfLeftLeaves(root.left);
        sum += sumOfLeftLeaves(root.right);
        return sum;
    }
}
```