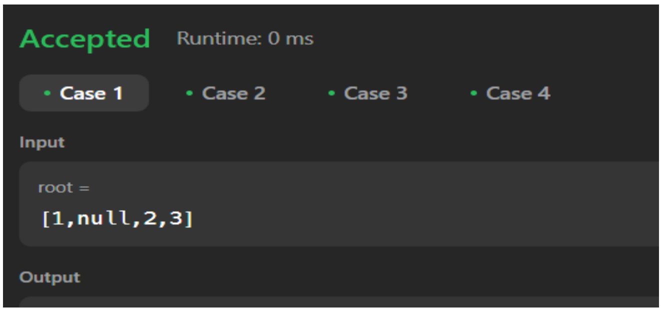
1.

```
class Solution {
  public List<Integer> inorderTraversal(TreeNode root) {
    List<Integer> res = new ArrayList<>();
    Stack<TreeNode> stack = new Stack<>();
    TreeNode curr = root;
    while (curr != null || !stack.isEmpty()) {
      while (curr != null) {
        stack.push(curr);
        curr = curr.left;
      }
      curr = stack.pop();
      res.add(curr.val);
      curr = curr.right;
    }
    return res;
  }
}
```


Accepted Runtime: 0 ms

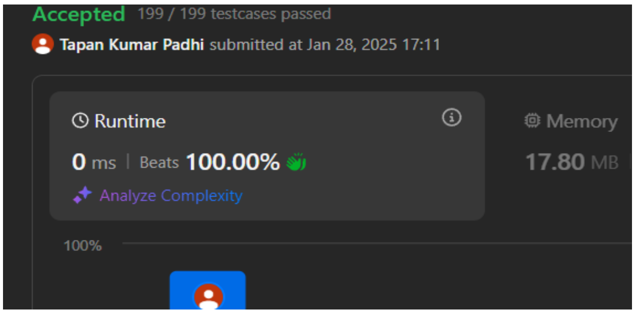
Input

root =
[1,null,2,3]

Output

2.

```
class Solution {
  public boolean isSymmetric(TreeNode root) {
    public boolean isSymmetric(TreeNode root) {
    return root == null || isMirror(root.left, root.right);
  }
  private boolean isMirror(TreeNode t1, TreeNode t2) {
    if (t1 == null || t2 == null) return t1 == t2;
    return t1.val == t2.val && isMirror(t1.left, t2.right) && isMirror(t1.right, t2.left);
  }
}
```


Accepted 199 / 199 testcases passed

Tapan Kumar Padhi submitted at Jan 28, 2025 17:11

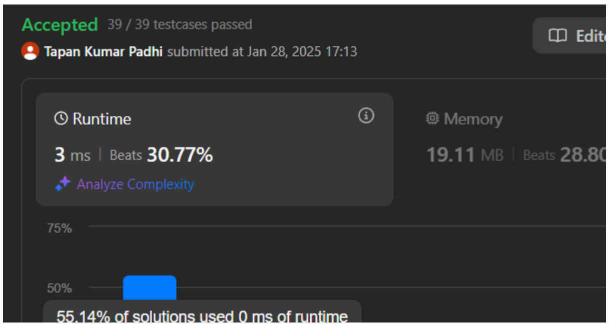
Runtime

0 ms | Beats 100.00%

17.80 MB

3.

```
class Solution {
    public int maxDepth(TreeNode root) {
        public int maxDepth(TreeNode root) {
        if (root == null) return 0;
        return 1 + Math.max(maxDepth(root.left), maxDepth(root.right));
    }
}
```



Accepted 39 / 39 testcases passed

Tapan Kumar Padhi submitted at Jan 28, 2025 17:13

Runtime

3 ms | Beats 30.77%

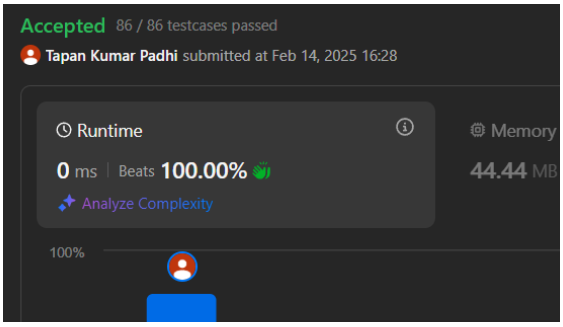
Memory

19.11 MB | Beats 28.80

55.14% of solutions used 0 ms of runtime

4.

```
class Solution {
    public boolean isValidBST(TreeNode root) {
        return validate(root, null, null);
    }
    private boolean validate(TreeNode node, Integer low, Integer high) {
        if (node == null) return true;
        if ((low != null && node.val <= low) || (high != null && node.val >= high)) return false;
        return validate(node.left, low, node.val) && validate(node.right, node.val, high);
    }
}
```



Accepted 86 / 86 testcases passed

Tapan Kumar Padhi submitted at Feb 14, 2025 16:28

Runtime

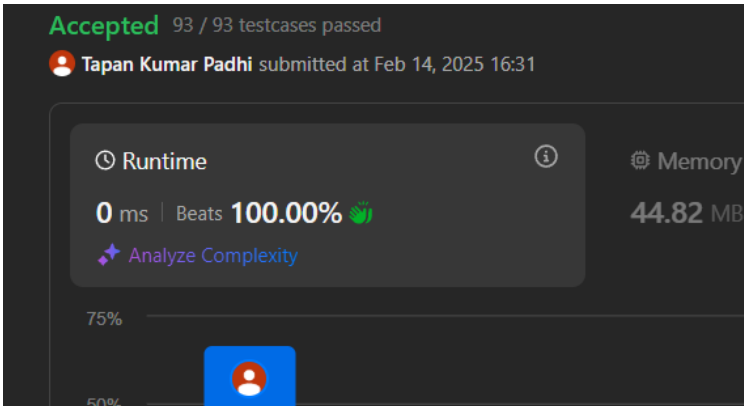
0 ms | Beats 100.00%

Memory

44.44 MB

5.

```
class Solution {
    public int kthSmallest(TreeNode root, int k) {
        Stack<TreeNode> stack = new Stack<>();
        while (true) {
            while (root != null) {
                stack.push(root);
                root = root.left;
            }
            root = stack.pop();
            if (--k == 0) return root.val;
            root = root.right;
        }
    }
}
```



Accepted 93 / 93 testcases passed

Tapan Kumar Padhi submitted at Feb 14, 2025 16:31

Runtime

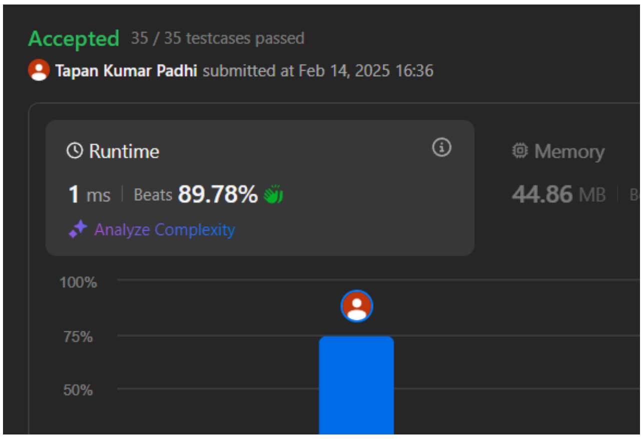
0 ms | Beats 100.00%

Memory

44.82 MB

6.

```
class Solution {
    public List<List<Integer>> levelOrder(TreeNode root) {
        List<List<Integer>> res = new ArrayList<>();
        if (root == null) return res;
        Queue<TreeNode> queue = new LinkedList<>();
        queue.offer(root);
        while (!queue.isEmpty()) {
            List<Integer> level = new ArrayList<>();
            int size = queue.size();
            for (int i = 0; i < size; i++) {
                TreeNode node = queue.poll();
                level.add(node.val);
                if (node.left != null) queue.offer(node.left);
                if (node.right != null) queue.offer(node.right);
            }
            res.add(level);
        }
        return res;
    }
}
```



Accepted 35 / 35 testcases passed

Tapan Kumar Padhi submitted at Feb 14, 2025 16:36

Runtime

1 ms | Beats 89.78%

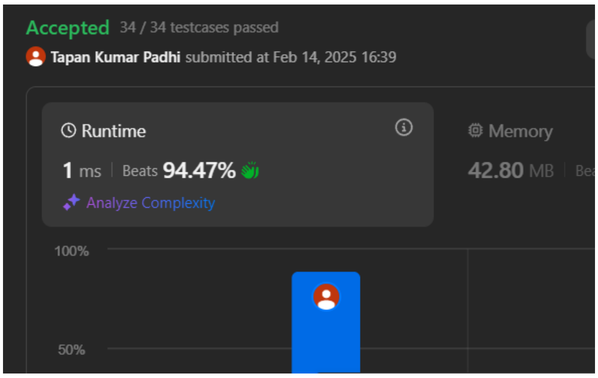
Analyze Complexity

Memory

44.86 MB

7.

```
class Solution {
  public List<List<Integer>> levelOrderBottom(TreeNode root) {
    LinkedList<List<Integer>> res = new LinkedList<>();
    if (root == null) return res;
    Queue<TreeNode> queue = new LinkedList<>();
    queue.offer(root);
    while (!queue.isEmpty()) {
      List<Integer> level = new ArrayList<>();
      int size = queue.size();
      for (int i = 0; i < size; i++) {
        TreeNode node = queue.poll();
        level.add(node.val);
        if (node.left != null) queue.offer(node.left);
        if (node.right != null) queue.offer(node.right);
      }
      res.addFirst(level);
    }
    return res;
  }
}
```



Accepted 34 / 34 testcases passed

Tapan Kumar Padhi submitted at Feb 14, 2025 16:39

Runtime

1 ms | Beats 94.47%

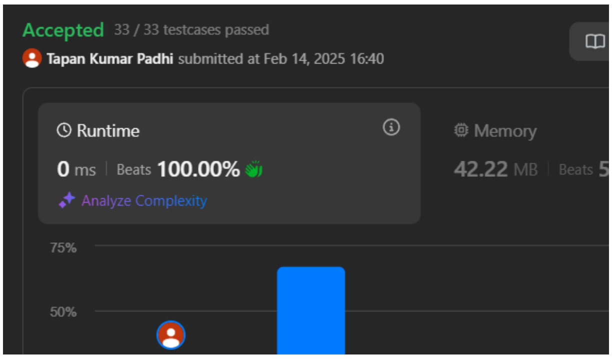
Analyze Complexity

Memory

42.80 MB | Bea

8.

```
class Solution {
    public List<List<Integer>> zigzagLevelOrder(TreeNode root) {
        List<List<Integer>> res = new ArrayList<>();
        if (root == null) return res;
        Queue<TreeNode> queue = new LinkedList<>();
        queue.offer(root);
        boolean leftToRight = true;
        while (!queue.isEmpty()) {
            LinkedList<Integer> level = new LinkedList<>();
            int size = queue.size();
            for (int i = 0; i < size; i++) {
                TreeNode node = queue.poll();
                if (leftToRight) level.addLast(node.val);
                else level.addFirst(node.val);
                if (node.left != null) queue.offer(node.left);
                if (node.right != null) queue.offer(node.right);
            }
            res.add(level);
            leftToRight = !leftToRight;
        }
        return res;
    }
}
```



Accepted 33 / 33 testcases passed

Tapan Kumar Padhi submitted at Feb 14, 2025 16:40

Runtime

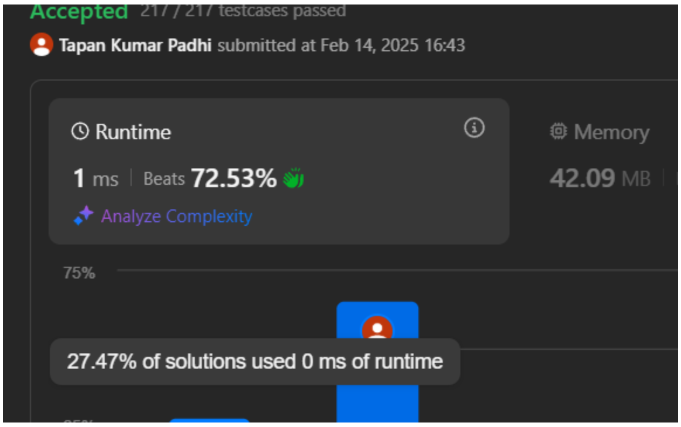
0 ms | Beats 100.00%

Memory

42.22 MB | Beats 5

9.

```
class Solution {
  public List<Integer> rightSideView(TreeNode root) {
    List<Integer> res = new ArrayList<>();
    if (root == null) return res;
    Queue<TreeNode> queue = new LinkedList<>();
    queue.offer(root);
    while (!queue.isEmpty()) {
      int size = queue.size();
      for (int i = 0; i < size; i++) {
        TreeNode node = queue.poll();
        if (i == size - 1) res.add(node.val);
        if (node.left != null) queue.offer(node.left);
        if (node.right != null) queue.offer(node.right);
      }
    }
    return res;
  }
}
```



Accepted 217 / 217 testcases passed

Tapan Kumar Padhi submitted at Feb 14, 2025 16:43

Runtime

1 ms | Beats 72.53%

Analyze Complexity

Memory

42.09 MB

27.47% of solutions used 0 ms of runtime