Experiment 5

Student Name: Nitil Jakhar
UID: 22BCS17300
Branch: CSE
Section/Group: NTPP_IOT-602/A
Semester: 6th
Date of Performance: 3/02/2
Subject: AP 2

1. **Aim: Trees**
2. **Objective:**
   1. Maximum Depth of Binary Tree
   2. Binary Tree Inorder Traversal
3. **Code:**

1. **Maximum Depth of Binary Tree:**

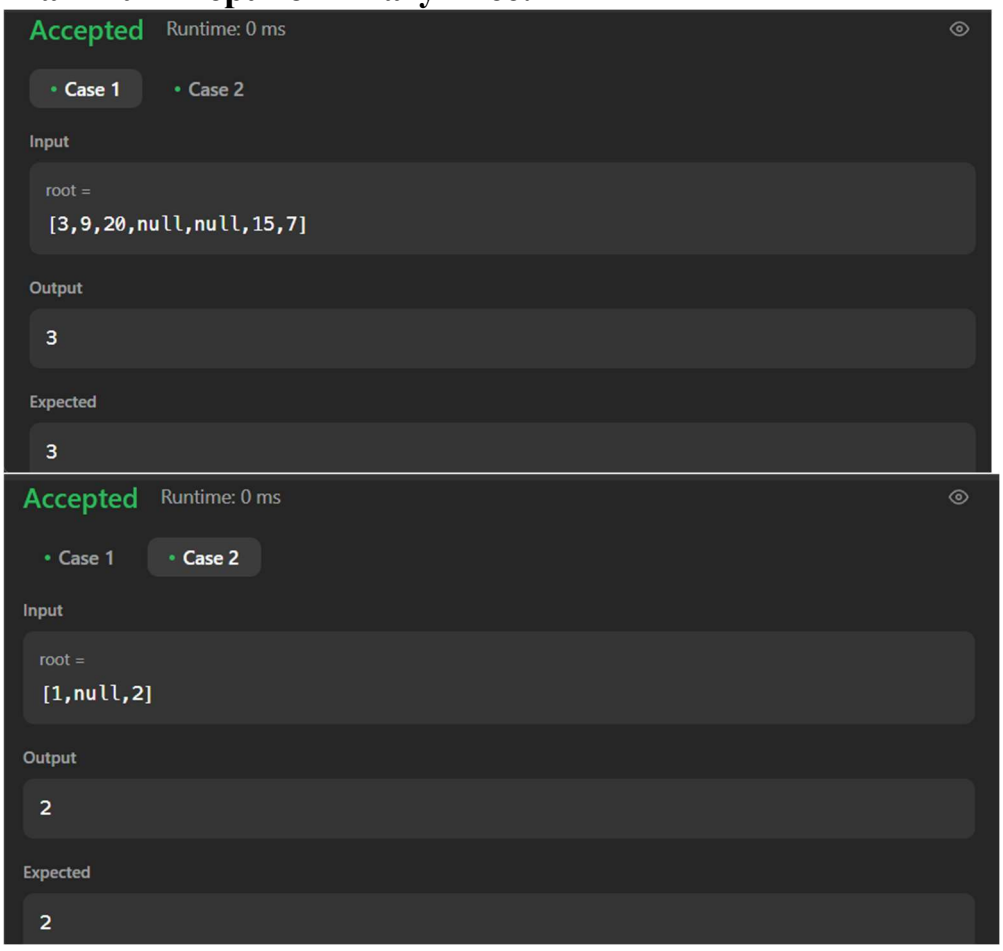
```
from typing import Optional

# Definition for a binary tree node.
class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
        self.right = right

class Solution:
    def maxDepth(self, root: Optional[TreeNode]) -> int:
        if not root:
            return 0
        left_depth = self.maxDepth(root.left)
        right_depth = self.maxDepth(root.right)
        return max(left_depth, right_depth) + 1
```

2. **Binary Tree Inorder Traversal:**

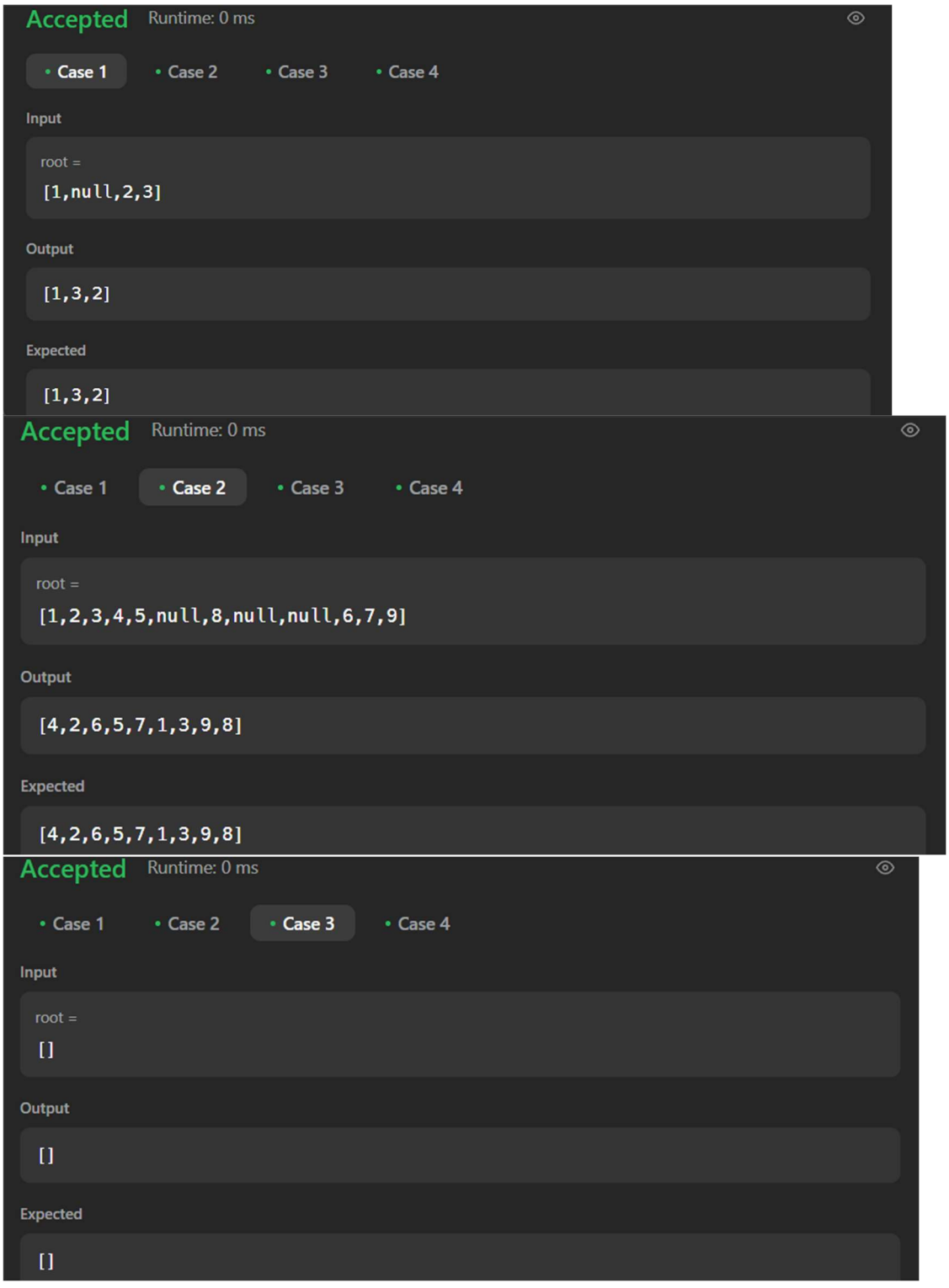
```
from typing import Optional, List

# Definition for a binary tree node.
class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
```

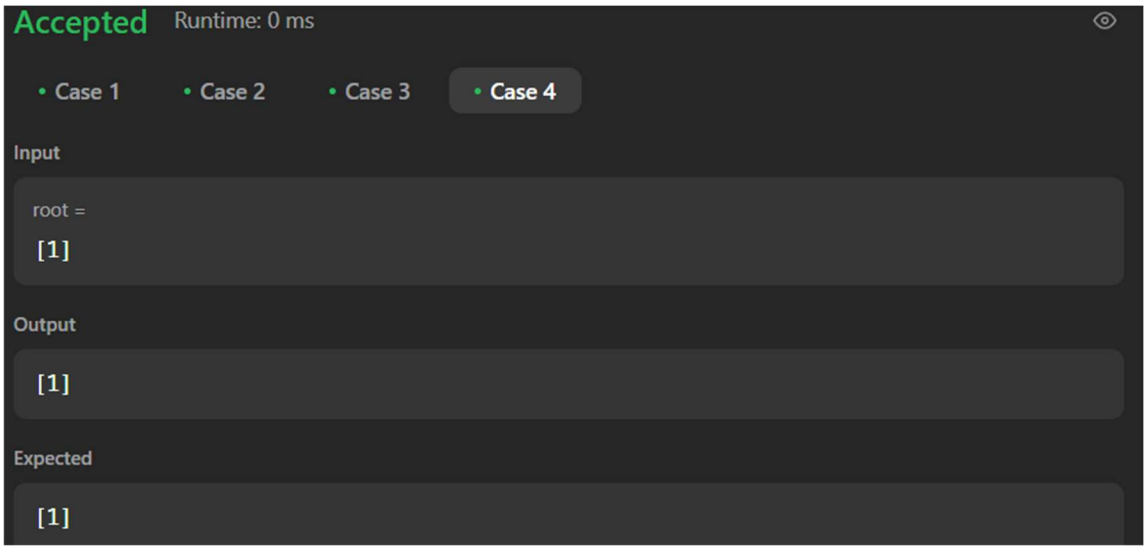
```
        self.right = right

class Solution:
    def inorderTraversal(self, root: Optional[TreeNode]) -> List[int]:
        def inorder(node):
            if not node:
                return []
            return inorder(node.left) + [node.val] + inorder(node.right)

        return inorder(root)
```

4. **Output:**
   1. **Maximum Depth of Binary Tree:**



   2. **Binary Tree Inorder Traversal:**


Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Case 4
Input
root =
[1,null,2,3]
Output
[1,3,2]
Expected
[1,3,2]
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Case 4
Input
root =
[1,2,3,4,5,null,8,null,null,6,7,9]
Output
[4,2,6,5,7,1,3,9,8]
Expected
[4,2,6,5,7,1,3,9,8]
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Case 4
Input
root =
[]
Output
[]
Expected
[]



5. Learning Outcome

1) Understand and implement **inorder traversal** (Left → Root → Right) in binary trees.
2) Learn the difference between **recursive** and **iterative** traversal methods.
3) Gain hands-on experience using **stacks** to simulate recursion in tree traversal.
4) Analyze the **time and space complexity** of different traversal approaches.
5) Develop problem-solving skills by converting recursive solutions into **iterative** ones.