Discover. Learn. Empower. ENGINEERING

WORKSHEET-6

StudentName: Rahul Yadav **UID:**22BCS13003

Branch:CSE **Section/Group:**NTPP-603-B

Semester:6th **DateofPerformance:**20/2/25

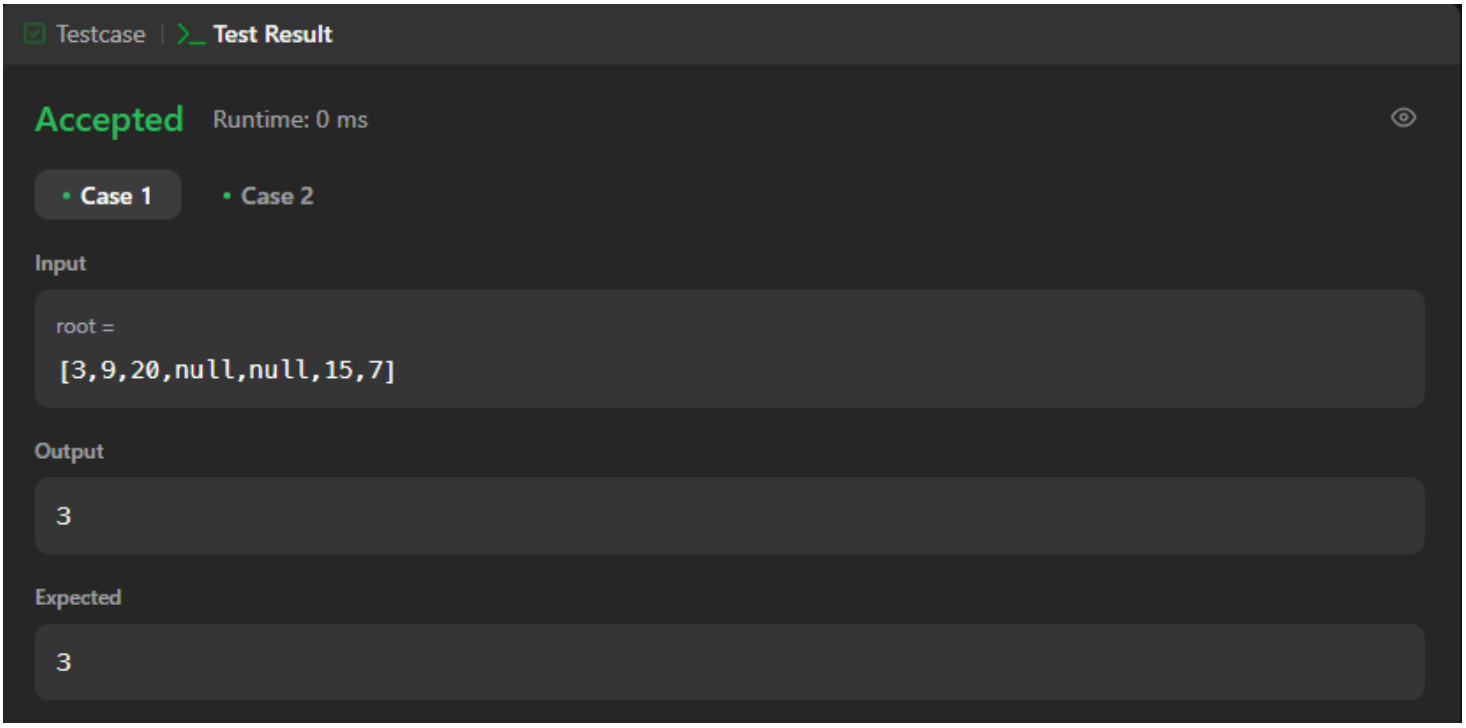
SubjectName:AP-2 **Subject Code:** 22CSP-351

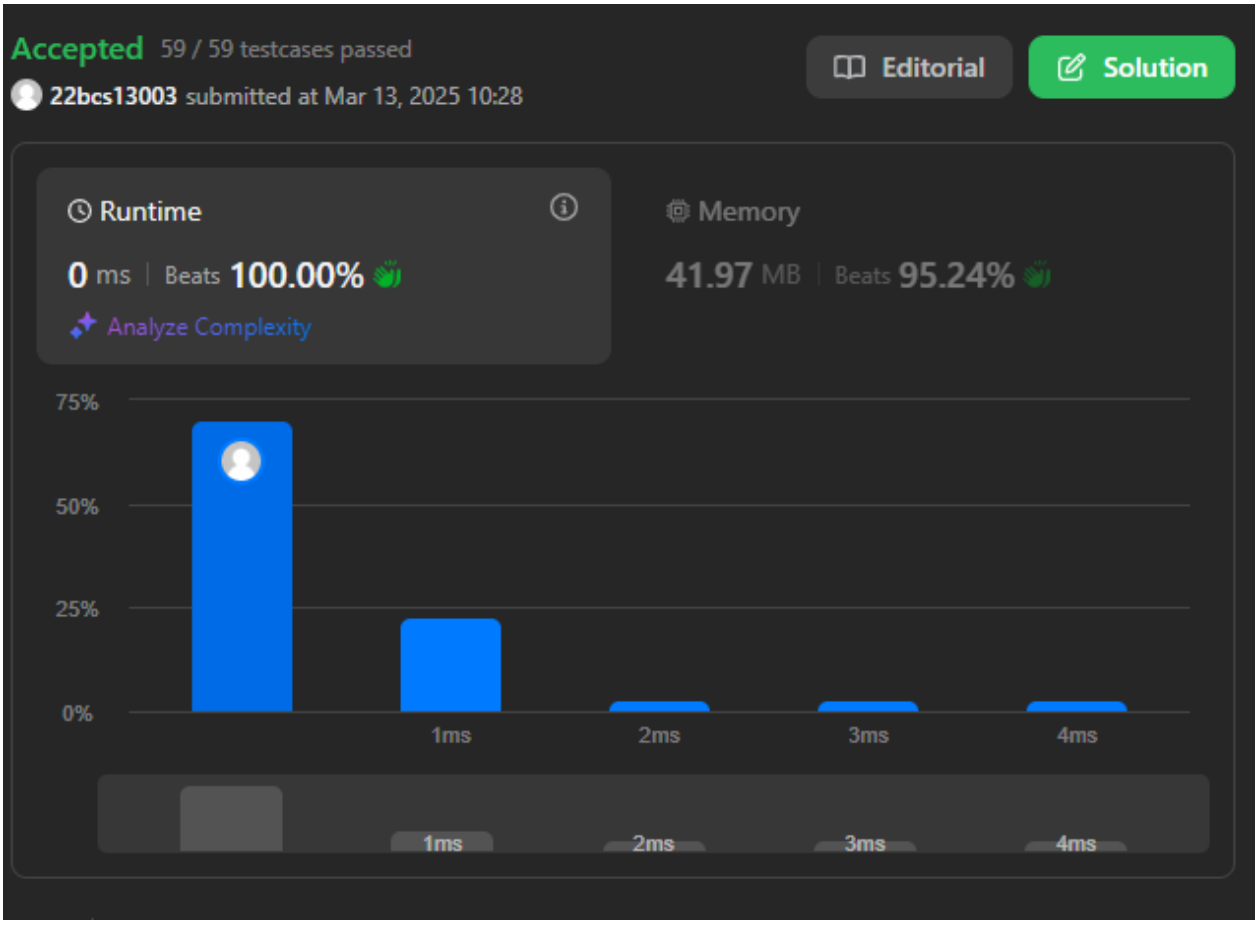
Aim(i):88.Youaregiventwointegerarraysnums1andnums2,sortedinnon-decreasing order, and two integers m and n, representingthe number of elements in nums1 and nums2 respectively. Merge nums1 and nums2 into a single array sorted in non-decreasing order.

SourceCode:

```
publicclassSolution{
    publicintmaxDepth(TreeNoderoot){
        //Basecase:Ifthenode is null,thedepthis0 if
        (root == null) {
            return0;
        }

        //Recursivelygetthedepthoftheleftandrightsubtrees int
        leftDepth = maxDepth(root.left);
        intrightDepth=maxDepth(root.right);

        //Thedepthofthecurrent node isthemaxofthe leftandrightsubtrees'depths+1 return
        Math.max(leftDepth, rightDepth) + 1;
    }
}
```

OUTPUT:





LEARNINGOUTCOME:

1. WelearntMergeSort.
2. WelearnthowtosortArrays.

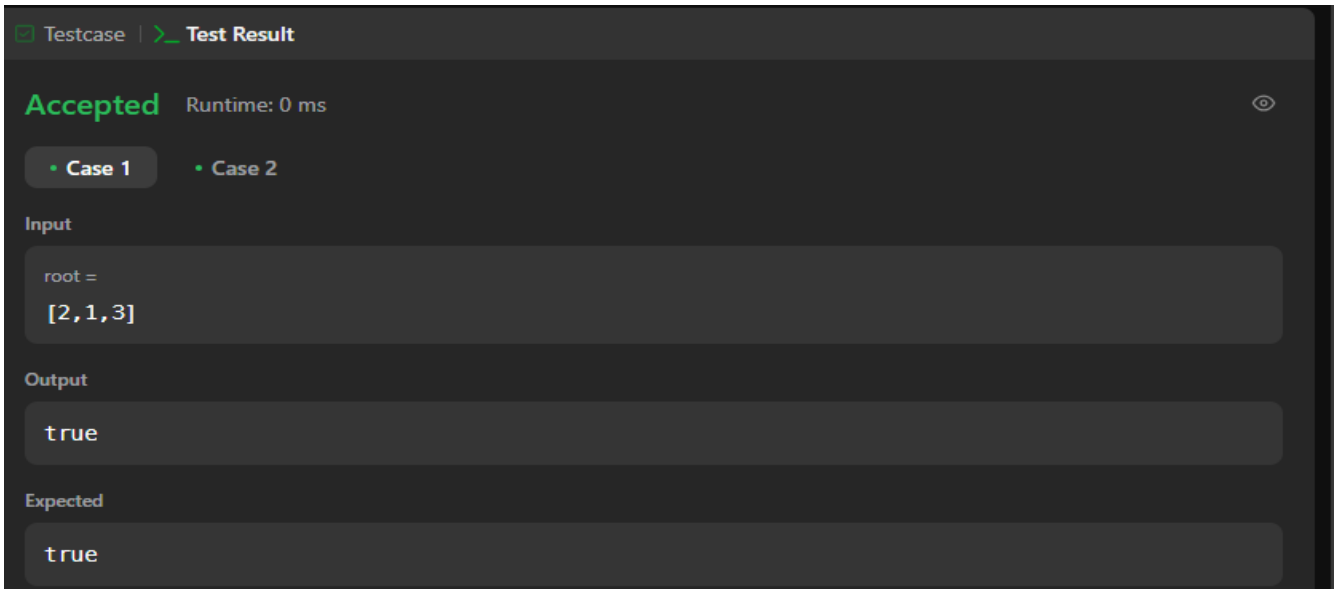
Aim(i):98Abinarytreeisavalid BSTif:

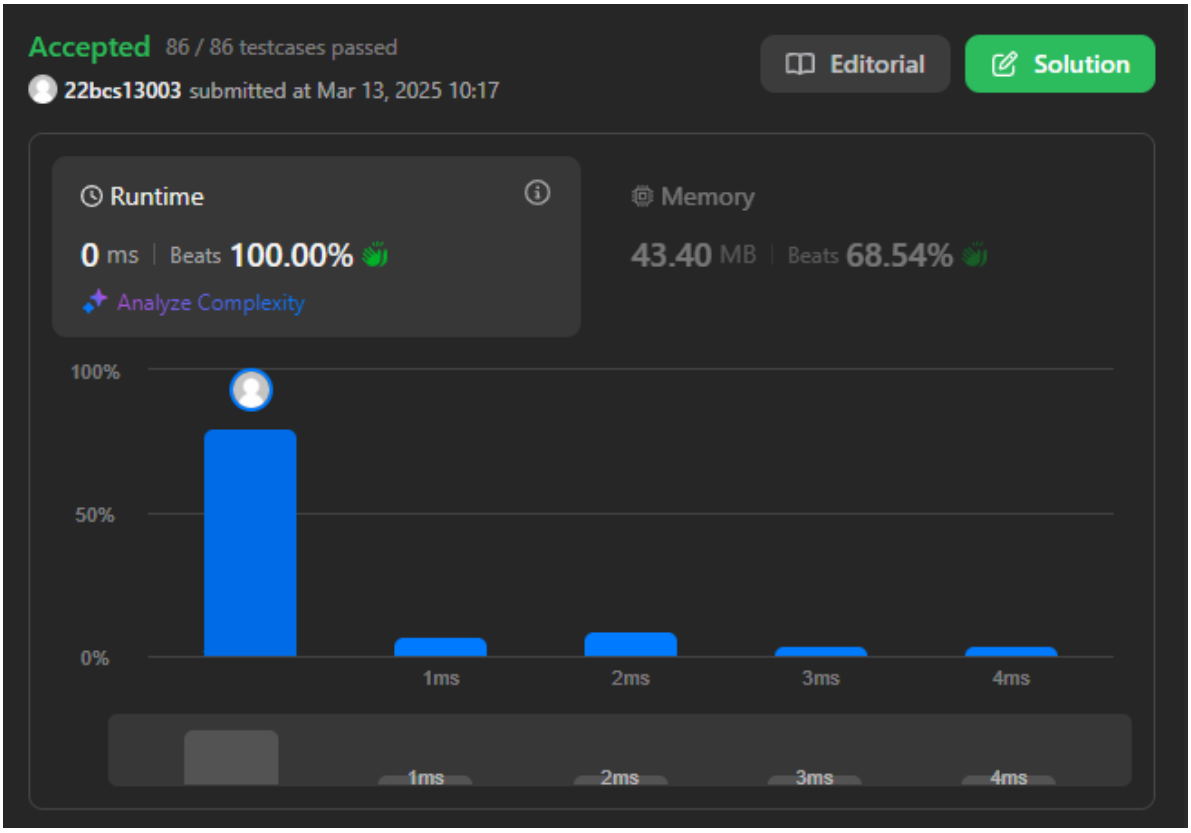
- Theleftsubtreeofanodecontainsonlynodeswithvalues**lessthan**thenode'svalue.
- Therightsubtreeofanodecontainsonlynodeswith values**greaterthan**thenode'svalue.
- Boththeleftandrightsubtreesmustalsobebinarysearchtrees.

SourceCode:

```
publicclassSolution{
    publicbooleanisValidBST(TreeNoderoot){
        returnisValidBSTHelper(root,Long.MIN_VALUE,Long.MAX_VALUE);
    }

    // Helper function to validate each node's value with its valid range
    privatebooleanisValidBSTHelper(TreeNodenode,longmin,longmax){
        //Basecase:ifthenodeisnull,itisvalid if
        (node == null) {
            returntrue;
        }

        //Checkifthecurrentnode'svalueiswithinthevalidrange if
        (node.val <= min || node.val >= max) {
            returnfalse;
        }

        //Recursivelycheckthe leftandrightsubtreeswithupdatedranges return
        isValidBSTHelper(node.left, min, node.val) &&
isValidBSTHelper(node.right,node.val,max);
    }
}
```

OUTPUT:


Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
root =
[2,1,3]
Output
true
Expected
true


Accepted
86 / 86 testcases passed
22bcs13003 submitted at Mar 13, 2025 10:17
Editorial
Solution
Runtime
0 ms | Beats 100.00%
Analyze Complexity
Memory
43.40 MB | Beats 68.54%
100%
50%
0%
1ms
2ms
3ms
4ms
1ms
2ms
3ms
4ms

Aim(iii): Givenabinarytree,determine ifit is **symmetric**around itscenter.Asymmetrictreeisatree where:

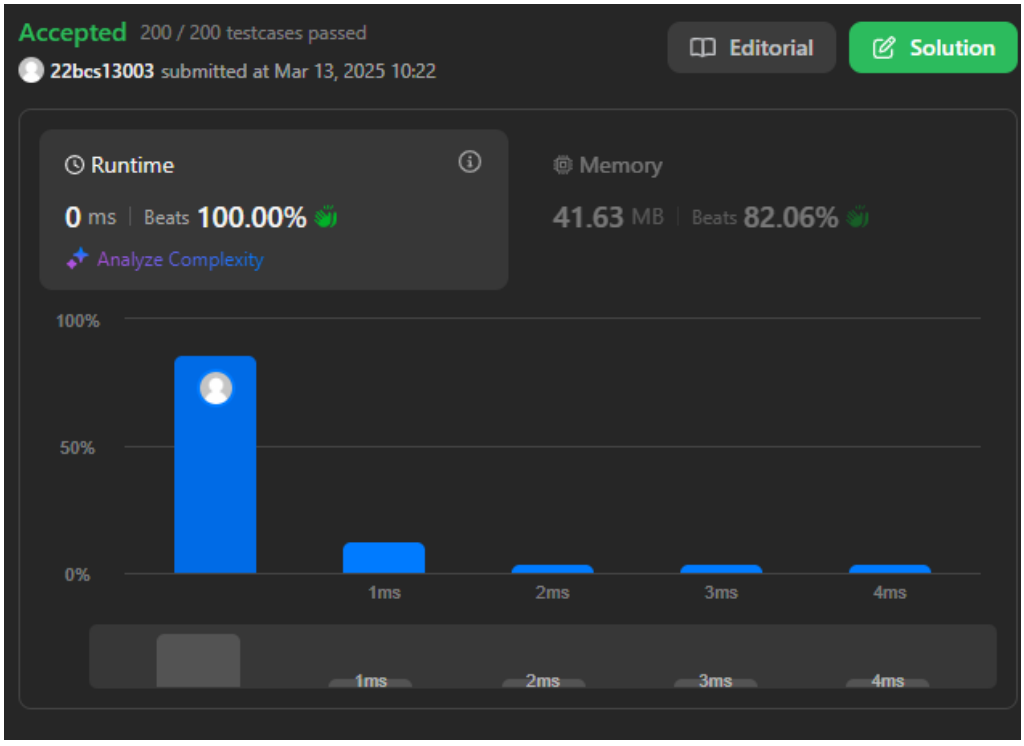
- Theleftandrightsubtreesaremirrorimagesofeachother.

SourceCode:

```
publicclassSolution{
    publicbooleanisSymmetric(TreeNoderoot){
        //Iftherootisnull,thetreeissymmetric(emptytreeissymmetric). if (root
        == null) {
            returntrue;
        }

        //Checkiftheleftandrightsubtreesaremirrorimagesofeachother. return
        isMirror(root.left, root.right);
    }

    //Helperfunctiontocheckiftwotreesaremirrorimagesofeachother. private
    boolean isMirror(TreeNode left, TreeNode right) {
        //Basecase:ifbothnodesarenull,theyaresymmetric. if (left
        == null && right == null) {
            returntrue;
        }

        //Ifonlyoneofthemisnull,theyarenotsymmetric. if (left
        == null || right == null) {
            returnfalse;
        }

        //Thevaluesatthecurrentnodesmustbethesameandtheleftsubtreeoftheleftnode
        //mustbeamirrorimageoftherightsubtreeoftherightnode,andviceversa. return
        (left.val == right.val)
            && isMirror(left.left, right.right)
            &&isMirror(left.right,right.left);
    }
}
```

OUTPUT:


Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
root =
[1,2,2,3,4,4,3]
Output
true
Expected
true


Accepted
200 / 200 testcases passed
22bcs13003 submitted at Mar 13, 2025 10:22
Editorial
Solution
Runtime
0 ms | Beats 100.00%
Analyze Complexity
Memory
41.63 MB | Beats 82.06%
100%
50%
0%
1ms
2ms
3ms
4ms
1ms
2ms
3ms
4ms

LearningOutcomes

1. WelearntCountingSort.
2. UsageofaHashMap.