Experiment6

StudentName: Khush Gulia **UID:**22BCS15819
Branch:BE-CSE **Section/Group:**22BCS_NTPP_IOT_603
Semester:6th **DateofPerformance:**27-02-25
SubjectName:APLab-2 **SubjectCode:**22CSP-351

1. Aim:Symmetric Tree

2. Objective:

Giventherootofabinarytree,*checkwhetheritisamirrorofitself* (i.e., symmetric around its center).

3. Implementation/Code:

```
boolhelp(TreeNode*r1,TreeNode*r2){ if(r1 ==

        NULL and r2 == NULL){
            returntrue;
        }
        if(r1==NULLandr2!=NULL){
            returnfalse;
        }
        if(r1!=NULLandr2==NULL){
            returnfalse;
        }
        if(r1->val!=r2->val){ return
            false;
        }

        boolone=help(r1->left,r2->right);
        booltwo=help(r1->right,r2->left);

        boolans=one&two;

        returnans;
    }
    bool isSymmetric(TreeNode* root) {

        returnhelp(root->left,root->right);

    }
```

4. Output

Accepted Runtime: 0 ms

• Case 1 • Case 2

Input

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

Output

true

Expected

true

Accepted Runtime: 0 ms

• Case 1 • Case 2

Input

root =
[1,2,2,null,3,null,3]

Output

false

Expected

false

5. **LearningOutcome:**

i. WeLearnAbouttheuseofReccursion.
ii. WeLearnAbouttheuseofListNode.
iii. WeLearnAbouttheuseofBaseCases.
iv. WelearnAbouttheCallingfunctionin recc..

Question 2

1. **Aim**:-**KthSmallestElementinaBST**

2. **Objective**:-

Giventherootofabinarysearchtree,andanintegerk, return*the*kth*smallestvalue(**1-indexed**)ofallthevaluesofthe nodes in the tree*.

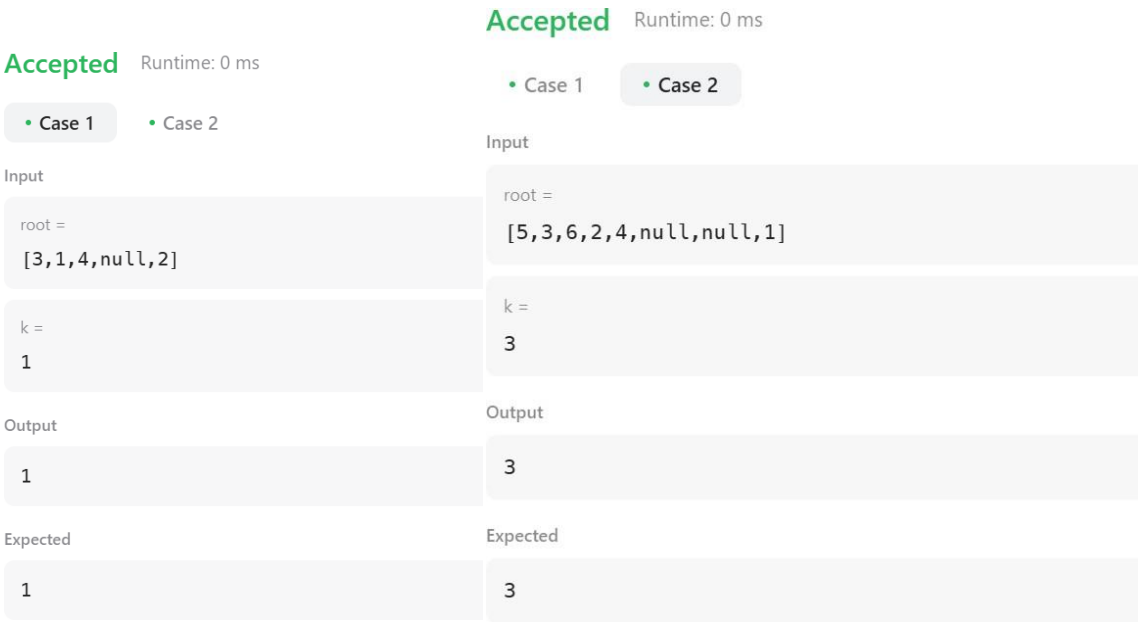
3. **Implementation/Code:-**

```
voidhelp(TreeNode*root,vector<int>&ans){

    if(root == NULL){
       return;
    }
    help(root->left,ans);
    ans.push_back(root->val);
    help(root->right,ans);
}
intkthSmallest(TreeNode*root,intk){
    vector<int>ans;
    help(root,ans);

    returnans[k-1];
}
```

4. Output:-



5. **LearningOutcome:**
   1. WeLearnabouttheinordertraversal
   2. WeLearnaboutthefunctioncalls
   3. Welearnedaboutrecursion.

Question 3

6. **Aim**:-**ConvertSortedArraytoBinarySearchTree**

7. **Objective**:-

Givenanintegerarraynumswheretheelementsaresorted in**ascendingorder**,convert*ittoa**height-balanced**binarysearch tree*.

8. Implementation/Code:-

```
TreeNode*help(ints,inte,vector<int>&nums){ if(s>

        e){
            returnNULL;
        }

        intmid=(s+e)/2;
        TreeNode*newnode=newTreeNode(nums[mid]);

        newnode->left = help(s,mid-1,nums);
        newnode->right=help(mid+1,e,nums);
        return newnode;
    }
    TreeNode*sortedArrayToBST(vector<int>&nums){ int
        s =0;
        int e = nums.size()-1;
        returnhelp(s,e,nums);
    }
```

9. Output:-

10. LearningOutcome:

- Welearnabouttocreateanewnode.
- Welearnaboutfunctioncalls.
- Welearnaboutthetopushmiddlevalue.
- Welearntomakeatreefrom recc.