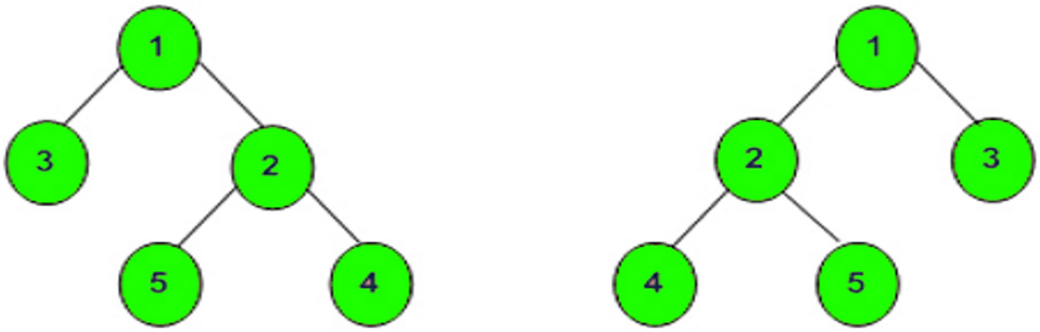
Monday, January 24, 2022 7:22 AM

Mirror Tree

Easy Accuracy: 63.43% Submissions: 100k+ Points: 2

Given a Binary Tree, convert it into its mirror.



Mirror Trees

Example 1:

Input:
```
    1
  /  \
 2    3
```
Output: 3 1 2

Explanation: The tree is
```
  1    (mirror)  1
 /  \    =>     /  \
2    3         3    2
```
The inorder of mirror is 3 1 2

Example 2:

Input:
```
    10
   /  \
  20   30
 /  \
 40  60
```

Output: 30 10 60 20 40
Explanation: The tree is

```
    10            10
  /   \ (mirror) /   \
 20   30   =>  30   20
/ \               /  \
40  60              60   40
```

The inroder traversal of mirror is
30 10 60 20 40.

Your Task:

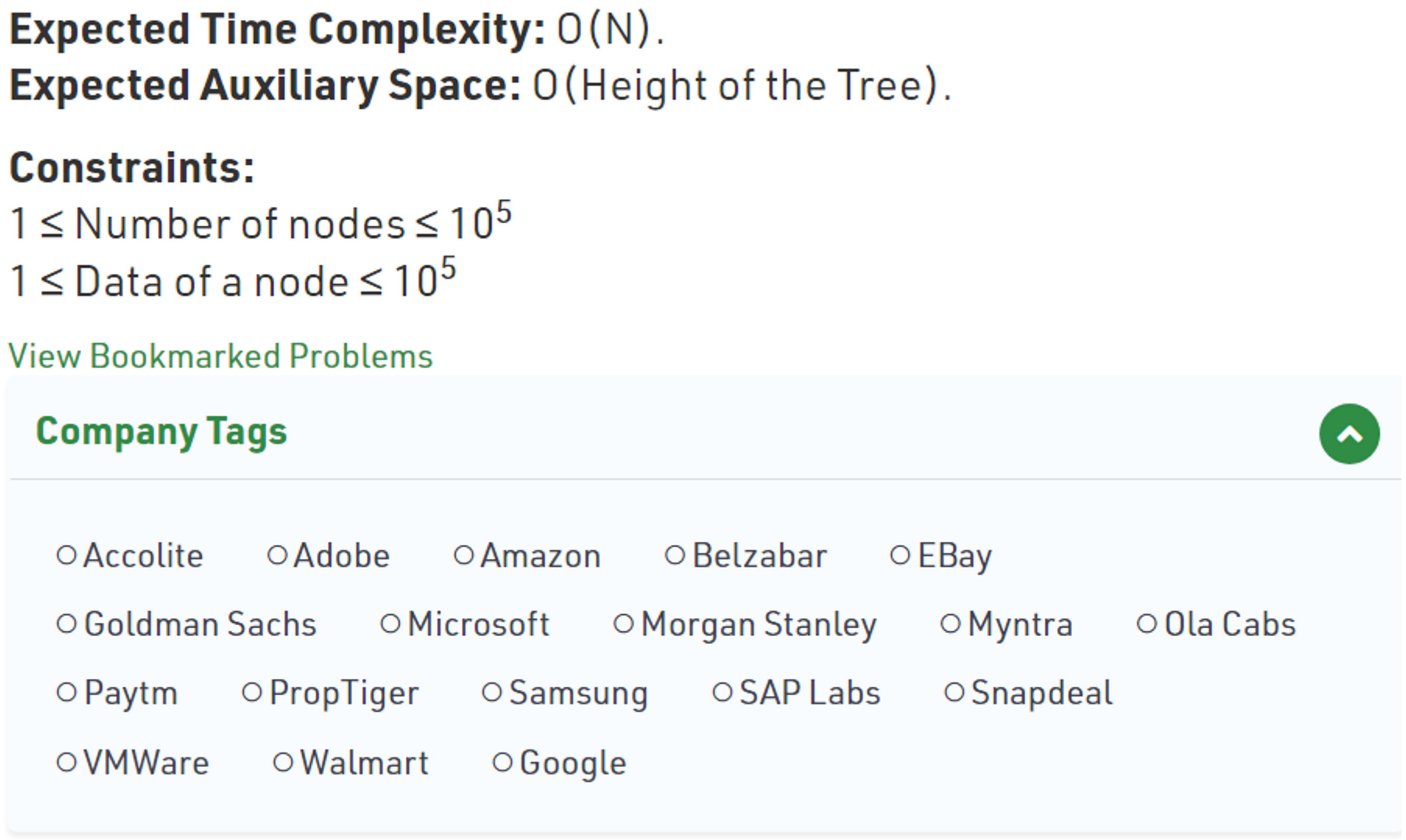
Just complete the **function mirror()** that takes **node** as **paramter** and convert it into its mirror. The printing is done by the driver code only.

Expected Time Complexity: O(N).

Expected Auxiliary Space: O(Height of the Tree).

Constraints:

From <https://practice.geeksforgeeks.org/problems/mirror-tree/1>



Expected Time Complexity: O(N).
Expected Auxiliary Space: O(Height of the Tree).

Constraints:
$1 \le \text{Number of nodes} \le 10^5$
$1 \le \text{Data of a node} \le 10^5$

View Bookmarked Problems

Company Tags

○ Accolite ○ Adobe ○ Amazon ○ Belzabar ○ EBay
○ Goldman Sachs ○ Microsoft ○ Morgan Stanley ○ Myntra ○ Ola Cabs
○ Paytm ○ PropTiger ○ Samsung ○ SAP Labs ○ Snapdeal
○ VMWare ○ Walmart ○ Google

Related Interview Experiences

○ Vmware interview experience set 4 campus

○ Paytm interview experience set 8 hiring drive for backend engineer

○ Adobe interview experience shecodes software engineer

```
package tree;
import java.util.*;
public class mirrorTree {
    void mirror(Node root) {
        // Your code here



        // mirror(root.left);
        // mirror(root.right);

        // Node temp=root.left;

        // root.left=root.right;

        // root.right=temp;

        Queue<Node> q=new  LinkedList<>();

        q.add(root);

        while(!q.isEmpty()){

            Node temp=q.peek();
            q.poll();

            Node node=temp.left;

            temp.left=temp.right;

            temp.right=node ;

            if(temp.left!=null)
            {
                q.add(temp.left);
            }

            if(temp.right!=null)
            {
                q.add(temp.right);

            }
        }
    }
}
```