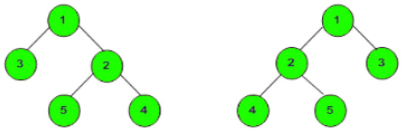
Problem | Editorial | Submissions | Comments

Mirror Tree

Difficulty: **Easy** Accuracy: **72.67%** Submissions: **199K+** Points: **2**

Given a Binary Tree, convert it into its mirror.



Mirror Trees

Examples:

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

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.

Expected Time Complexity: O(n)
Expected Auxiliary Space: O(height of the tree)

Constraints:
$1 \leq$ Number of nodes $\leq 10^5$
$1 \leq$ Data of a node $\leq 10^5$

Try more examples

Seen this question in a real interview before ? Yes No

Company Tags

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

Topic Tags

Related Interview Experiences

Related Articles

Report An Issue
If you are facing any issue on this page. Please let us know.

Java (1.8) Average Time: 20m Start Timer

```java
//{ Driver Code Starts
// Initial Template for Java

// Contributed by Sudarshan Sharma
import java.util.LinkedList;
import java.util.Queue;
import java.io.*;
import java.util.*;

class Node {
    int data;
    Node left;
    Node right;
    Node(int data) {
        this.data = data;
        left = null;
        right = null;
    }
}

class GfG {

    static Node buildTree(String str) {

        if (str.length() == 0 || str.charAt(0) == 'N') {
            return null;
        }

        String ip[] = str.split(" ");
        // Create the root of the tree
        Node root = new Node(Integer.parseInt(ip[0]));
        // Push the root to the queue

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

        queue.add(root);
        // Starting from the second element

        int i = 1;
        while (queue.size() > 0 && i < ip.length) {

            // Get and remove the front of the queue
            Node currNode = queue.peek();
            queue.remove();

            // Get the current node's value from the string
            String currVal = ip[i];

            // If the left child is not null
```

Custom Input Compile & Run Submit