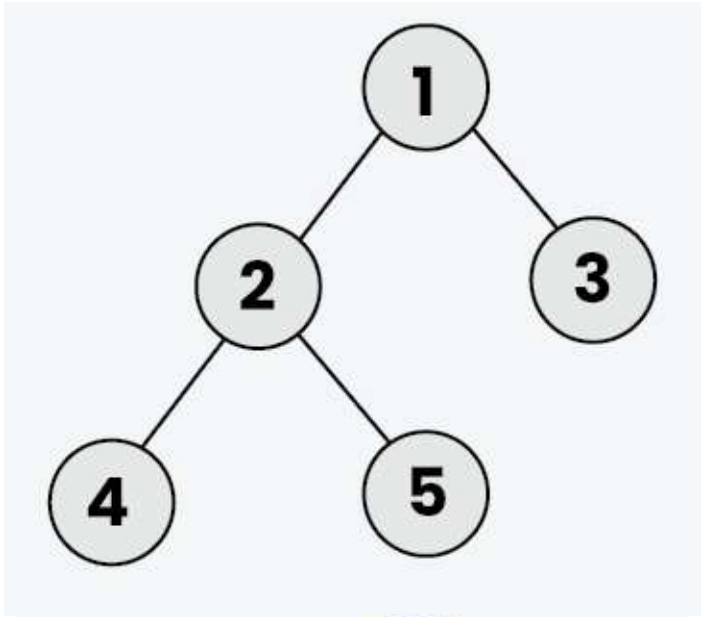
Forest Ranger's Left View

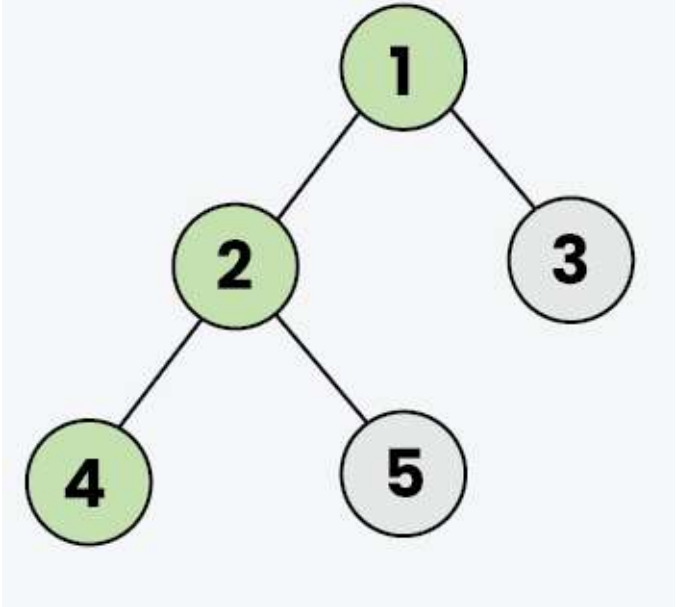
locked

Problem Submissions Leaderboard Discussions

Forest rangers are monitoring trees in a dense forest. The trees are connected in a hierarchical structure where each tree node has branches going to other trees (children). They want to take a snapshot from the leftmost side to capture the visible part of the forest from that angle.

Given the root of a binary tree representing the forest, print the values of the nodes visible from the left side when viewed from the leftmost direction.





Input Format

The first line contains space-separated integers representing the level order traversal of the binary tree.

A -1 indicates that there is no node at that position (null node).

Constraints

0 <= number of nodes <= 106 0 <= node -> data <= 105

Output Format

Print a single line with the node values visible from the left side, separated by spaces.

For the example above, the output should be:

Sample Input 0

```
1 2 3 4 5 Null Null
```

Sample Output 0

```
1 2 4
```

Explanation 0

From the left side of the tree, only the nodes 1, 2, and 4 are visible.

Sample Input 1

```
1 2 3 Null Null 4 Null Null 5 Null Null
```

Sample Output 1

```
1 2 4 5
```

Explanation 1

From the left side of the tree, the nodes 1, 2, 4, and 5 are visible.

Submissions: 20
Max Score: 10
Difficulty: Medium

Rate This Challenge:

More

Java 7

```java
import java.io.*;
import java.util.*;

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

public class Solution {
    static void disp(node root) {
        if (root == null)
            return;
        Queue<node> q = new LinkedList<>();
        q.add(root);
        while (!q.isEmpty()) {
            int s = q.size();
```

```
            for (int i = 0; i < s; i++) {
                node e = q.poll();
                if (i == 0) {
                    System.out.print(e.data + " ");
                }
                if (e.left != null) {
                    q.add(e.left);
                }
                if (e.right != null) {
                    q.add(e.right);
                }
            }
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int val = sc.nextInt();
        node root = new node(val), nn;
        Queue<node> q = new LinkedList<>();
        q.add(root);

        while (!q.isEmpty()) {
            node e = q.poll();

            if (sc.hasNext()) {
                String lstr = sc.next();
                if (!lstr.equals("Null")) {
                    val = Integer.parseInt(lstr);
                    nn = new node(val);
                    e.left = nn;
                    q.add(nn);
                }
            }

            if (sc.hasNext()) {
                String rstr = sc.next();
```

```
60                 if (!rstr.equals("Null")) {
61                     val = Integer.parseInt(rstr);
62                     nn = new node(val);
63                     e.right = nn;
64                     q.add(nn);
65                 }
66             }
67         }
68 
69         disp(root);
70     }
71 }
72 
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code