https://course.acciojob.com/idle?question=e81eafdb-0965-4651-b3a3-ac61b70ff8da

MEDIUM

Max Score: 40 Points

Symmetric Tree

Given the root node of a tree. Your task is to check whether it's symmetric about the center or not.

You have to complete `isSymmetrical` function which consists of pointer to the root of binary tree as input and prints the string output as `YES` or `NO`

Input Format

You are given the pointer to the root of the binary tree.

Output Format

For each test case print "YES" if the tree is symmetric, else print "NO"

Example 1

Input

```
2 1 1
```

Output

```
YES
```

Explanation

We have same numbers on both the subtrees.

Example 2

Input

```
1 2 3
```

Output

```
NO
```

Explanation

We have different numbers on both the subtrees.

Constraints

`1 <= n <= 10^5`

The value of any node is less than `2^32`

Topic Tags

Recursion

Trees

My code

```
// in java
import java.util.LinkedList;
import java.util.Queue;
import java.io.*;
import java.util.*;

class Main {
    static Node buildTree(String str) {
        if (str.length() == 0 || str.charAt(0) == 'N') {
            return null;
        }
        String ip[] = str.split(" ");
        Node root = new Node(Integer.parseInt(ip[0]));
        Queue<Node> queue = new LinkedList<>();
        queue.add(root);
        int i = 1;
        while (queue.size() > 0 && i < ip.length) {
            Node currNode = queue.peek();
            queue.remove();
            String currVal = ip[i];
            if (!currVal.equals("N")) {
                currNode.left = new Node(Integer.parseInt(currVal));
                queue.add(currNode.left);
            }
            i++;
            if (i >= ip.length) break;
            currVal = ip[i];
            if (!currVal.equals("N")) {
                currNode.right = new Node(Integer.parseInt(currVal));
                queue.add(currNode.right);
```

```
            }
            i++;
        }

        return root;
    }

    public static void main(String[] args) throws IOException {
        BufferedReader br =new BufferedReader(new
InputStreamReader(System.in));
        String s1 = br.readLine();
        Node root1 = buildTree(s1);
        Solution g = new Solution();
        g.isSymmetrical(root1);
    }
}


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

class Solution {
```

```
    static int fun(Node root)
    {
        if(root==null)
            return 1;

        if(root.left==null && root.right==null)
                return 1;
         if(root.left==null && root.right!=null)
                return 0;
         if(root.left!=null && root.right==null)
                return 0;
        if(root.left.data==root.right.data)
            return fun(root.left)*fun(root.right);
        return 0;
    }
  public static void isSymmetrical(Node root) {
    //Your code here
        if(fun(root)==1)
            System.out.print("YES");
       else System.out.print("NO");
  }
}
```