https://course.acciojob.com/idle?question=078031e9-9e7d-42e3-8d08-3ab2cdcc2686

- EASY
- Max Score: 30 Points

Root Equals sum of children

You are given the root of a binary tree that consists of exactly 3 nodes: the root, its left child, and its right child.

Return true if the value of the root is equal to the sum of the values of its two children, or false otherwise.

Input Format

Given the root of the tree.

Output Format

Return true or false.

EXAMPLE 1

Input

```
10 4 6 N N N N
  10
 /  \
4    6
```

Output

```
true
```

Explanation

The values of the root, its left child, and its right child are 10, 4, and 6, respectively.

10 is equal to 4 + 6, so we return true.

EXAMPLE 2

Input

```
5 3 1 N N N N
  5
 /  \
3    1
```

Output

```
false
```

Explanation

The values of the root, its left child, and its right child are 5, 3, and 1, respectively.

5 is not equal to 3 + 1, so we return false.

CONSTRAINTS

The tree consists only of the root, its left child, and its right child.

-100 <= Node.val <= 100

Topic Tags

- **Trees**

My code

```
// in java
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 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{
        Scanner sc = new Scanner(System.in);
                String s = sc.nextLine();
                Node root = buildTree(s);
                Solution tree = new Solution();
                boolean result=tree.checkTree(root);
                System.out.print(result);
    }
}



class Solution{
  public boolean checkTree(Node root)
    {
    //Write code here
```

```
            if(root.data==root.left.data+root.right.data)
                return true;
                    return false;
    }
}
```