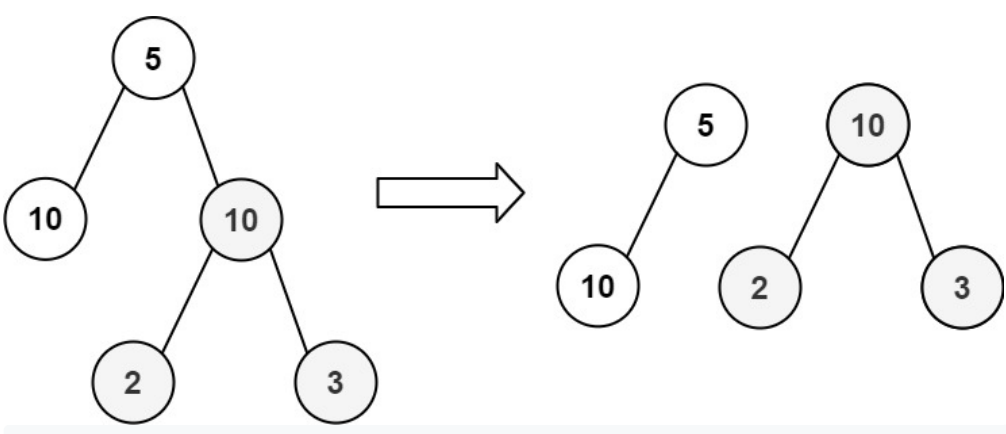


663. Equal Tree Partition

Medium👍 403🗨 30💖 Add to List🔗 Share

Given the `root` of a binary tree, return `true` if you can partition the tree into two trees with equal sums of values after removing exactly one edge on the original tree.

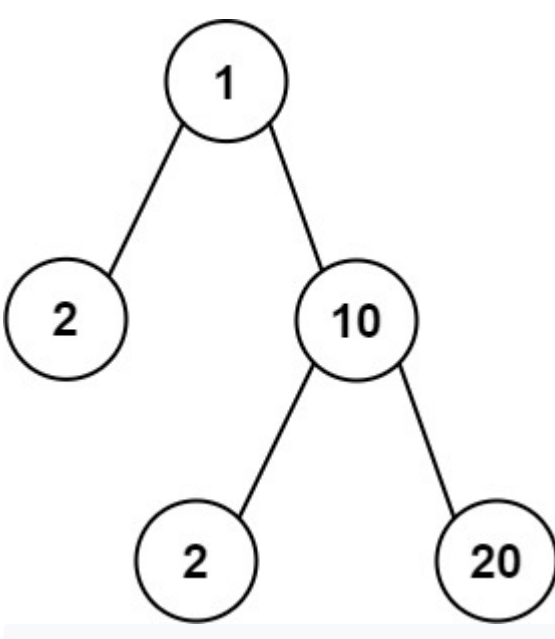
Example 1:



```
graph TD
    5((5)) --- 10L((10))
    5 --- 10R((10))
    10L --- 2((2))
    10L --- 3((3))
    5 --> 5_2[ ]
    5_2 --- 5_5((5))
    5_2 --- 10_10[ ]
    10_10 --- 10_10L((10))
    10_10 --- 10_10R((10))
    10_10L --- 10_2((2))
    10_10L --- 10_3((3))
```

Input: root = [5,10,10,null,null,2,3]  
Output: true

Example 2:



```
graph TD
    1((1)) --- 2L((2))
    1 --- 10((10))
    10 --- 2R((2))
    10 --- 20((20))
```

Input: root = [1,2,10,null,null,2,20]  
Output: false  
Explanation: You cannot split the tree into two trees with equal sums after removing exactly one edge on the tree.

- Constraints:
- The number of nodes in the tree is in the range  $[1, 10^4]$ .
  - $-10^5 \leq \text{Node.val} \leq 10^5$

Accepted 25,650 Submissions 62,735

Seen this question in a real interview before?

Companies 🍌 /

0 ~ 6 months 6 months ~ 1 year 1 year ~ 2 years

Amazon 2

Related Topics

Tree Depth-First Search Binary Tree

```
1 class Solution {
2     Stack<Integer> seen;
3     public boolean checkEqualTree(TreeNode root) {
4         seen = new Stack<>();
5         int total = sum(root);
6         seen.pop();
7         if (total % 2 == 0)
8             for (int s: seen)
9                 if (s == total / 2)
10                    return true;
11        return false;
12    }
13
14    public int sum(TreeNode node) {
15        if (node == null) return 0;
16        seen.push(sum(node.left) + sum(node.right) + node.val);
17        return seen.peek();
18    }
19 }
```