Problem 663: Equal Tree Partition

Problem Information

Difficulty: Medium
Acceptance Rate: 42.25%
Paid Only: Yes
Tags: Tree, Depth-First Search, Binary Tree

Problem Description

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:

![](https://assets.leetcode.com/uploads/2021/05/03/split1-tree.jpg)

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

Example 2:

![](https://assets.leetcode.com/uploads/2021/05/03/split2-tree.jpg)

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, 104]`. * `-105 <= Node.val <= 105`

Code Snippets

C++:

```
/**
* Definition for a binary tree node.
* struct TreeNode {
* int val;
* TreeNode *left;
* TreeNode *right;
* TreeNode() : val(0), left(nullptr), right(nullptr) {}
* TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
* TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left),
right(right) {}
* };
*/
class Solution {
public:
bool checkEqualTree(TreeNode* root) {

}
};
```

Java:

```
/**
* Definition for a binary tree node.
* public class TreeNode {
* int val;
* TreeNode left;
* TreeNode right;
* TreeNode() {}
* TreeNode(int val) { this.val = val; }
* TreeNode(int val, TreeNode left, TreeNode right) {
* this.val = val;
* this.left = left;
* this.right = right;
* }
* }
*/
class Solution {
public boolean checkEqualTree(TreeNode root) {

}
}
```

Python3:

```
# Definition for a binary tree node.
# class TreeNode:
# def __init__(self, val=0, left=None, right=None):
# self.val = val
# self.left = left
# self.right = right
class Solution:
def checkEqualTree(self, root: Optional[TreeNode]) -> bool:
```