

# Problem 2236: Root Equals Sum of Children

## Problem Information

**Difficulty:** Easy

**Acceptance Rate:** 84.83%

**Paid Only:** No

**Tags:** Tree, Binary Tree

## Problem Description

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.

**Example 1:**



**Input:** `root = [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:** `root = [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`

## 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 checkTree(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 checkTree(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 checkTree(self, root: Optional[TreeNode]) -> bool:
```