

Problem 951: Flip Equivalent Binary Trees

Problem Information

Difficulty: Medium

Acceptance Rate: 69.61%

Paid Only: No

Tags: Tree, Depth-First Search, Binary Tree

Problem Description

For a binary tree **T**, we can define a **flip operation** as follows: choose any node, and swap the left and right child subtrees.

A binary tree **X** is **_flip equivalent_** to a binary tree **Y** if and only if we can make **X** equal to **Y** after some number of flip operations.

Given the roots of two binary trees `root1` and `root2`, return `true` if the two trees are flip equivalent or `false` otherwise.

Example 1:

[Flipped Trees Diagram](https://assets.leetcode.com/uploads/2018/11/29/tree_ex.png)

Input: `root1 = [1,2,3,4,5,6,null,null,null,7,8]`, `root2 = [1,3,2,null,6,4,5,null,null,null,8,7]`

Output: `true` **Explanation:** We flipped at nodes with values 1, 3, and 5.

Example 2:

Input: `root1 = []`, `root2 = []` **Output:** `true`

Example 3:

Input: `root1 = []`, `root2 = [1]` **Output:** `false`

Constraints:

* The number of nodes in each tree is in the range `[0, 100]`. * Each tree will have **unique node values** in the range `[0, 99]`.

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 flipEquiv(TreeNode* root1, TreeNode* root2) {

    }
};
```

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 flipEquiv(TreeNode root1, TreeNode root2) {

}

}

```

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 flipEquiv(self, root1: Optional[TreeNode], root2: Optional[TreeNode]) ->
    bool:

```