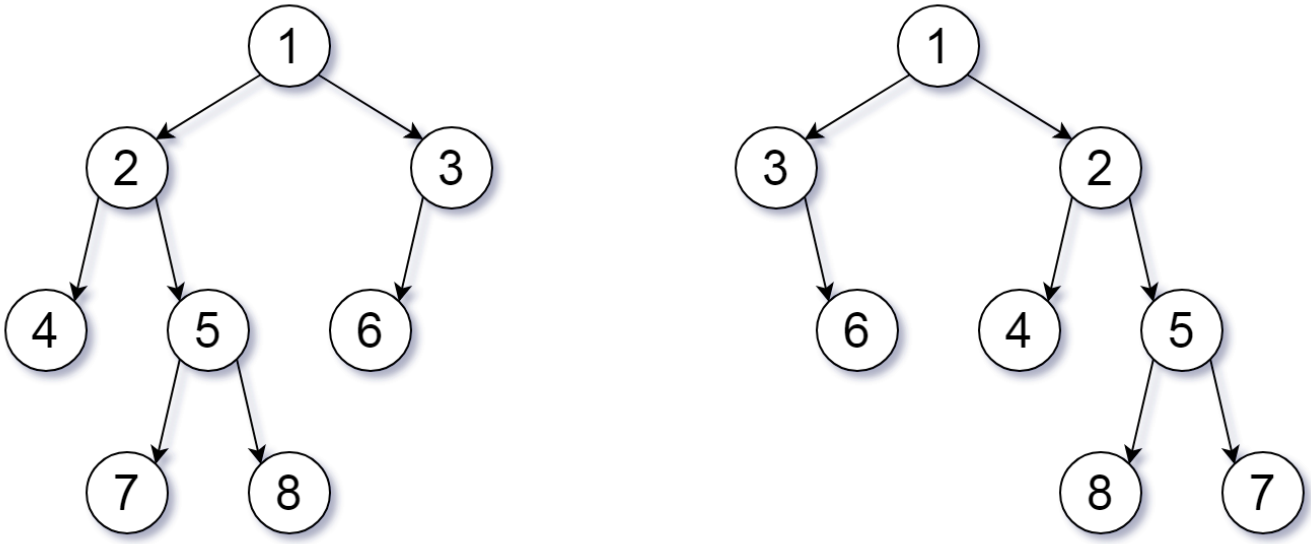
Check if Tree is Isomorphic or 951. Flip Equivalent Binary Trees

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 equivelent or `false` otherwise.



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

Example 4:

Input: root1 = [0,null,1], root2 = []
Output: false

Example 5:

Input: root1 = [0,null,1], root2 = [0,1]
Output: true

```
def flipEquiv(self, root1: TreeNode, root2: TreeNode) -> bool:
        if root1 is root2:
            return True
        if root1 is None or root2 is None or root1.val!=root2.val:
            return False
        return (self.flipEquiv(root1.left, root2.left) and
                self.flipEquiv(root1.right, root2.right) or
                self.flipEquiv(root1.left, root2.right) and
                self.flipEquiv(root1.right, root2.left))
```