563 Binary Tree Tilt

2018年4月15日 18:12

Given a binary tree, return the tilt of the **whole tree**.

The tilt of a **tree node** is defined as the **absolute difference** between the sum of all left subtree node values and the sum of all right subtree node values. Null node has tilt 0.

The tilt of the **whole tree** is defined as the sum of all nodes' tilt.

Example:

Input:

```
       1
     /   \
    2     3
```

Output: 1

Explanation:

Tilt of node 2 : 0

Tilt of node 3 : 0

Tilt of node 1 : |2-3| = 1

Tilt of binary tree : 0 + 0 + 1 = 1

Note:

1. The sum of node values in any subtree won't exceed the range of 32-bit integer.
2. All the tilt values won't exceed the range of 32-bit integer.

来自 <https://leetcode.com/problems/binary-tree-tilt/description/>

给定一个二叉树，计算**整个树**的坡度。

一个树的**节点的坡度**定义即为，该节点左子树的结点之和和右子树结点之和的**差的绝对值**。空结点的的坡度是0。

整个树的坡度就是其所有节点的坡度之和。

示例:

输入:

```
       1
     /   \
    2     3
```

输出: 1

解释:

结点的坡度 2 : 0

结点的坡度 3 : 0

结点的坡度 1 : |2-3| = 1

树的坡度 : 0 + 0 + 1 = 1

注意:

1. 任何子树的结点的和不会超过32位整数的范围。
2. 坡度的值不会超过32位整数的范围。

Solution for Python3：

```python
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, x):
#         self.val = x
#         self.left = None
#         self.right = None

class Solution1:
    def findTilt(self, root):
        """
        :type root: TreeNode
        :rtype: int
        """
        self.sum = 0
        self.dfs(root)
        return self.sum

    def dfs(self, root):
        if not root:
            return 0
        Lsum = self.dfs(root.left)
        Rsum = self.dfs(root.right)
        self.sum += abs(Lsum - Rsum)
        return root.val + Lsum + Rsum

class Solution2:
    def findTilt(self, root):
        """
        :type root: TreeNode
        :rtype: int
        """
        self.sum = 0
        def dfs(root):
            if not root:
                return 0
            Lsum, Rsum = dfs(root.left), dfs(root.right)
            self.sum += abs(Lsum - Rsum)
            return root.val + Lsum + Rsum
        dfs(root)
        return self.sum
```

Solution for C++:

```cpp
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution1 {
private:
    int sum = 0;
public:
    int findTilt(TreeNode* root) {
        dfs(root);
        return sum;
    }

    int dfs(TreeNode* root) {
        if (!root)
            return 0;
        int Lsum = dfs(root->left);
        int Rsum = dfs(root->right);
        sum += abs(Lsum - Rsum);
        return root->val + Lsum + Rsum;
    }
};

class Solution2 {
public:
    int findTilt(TreeNode* root) {
        int sum = 0;
        dfs(root, &sum);
        return sum;
    }

```

```
38      int dfs(TreeNode* root, int* sum) {
39          if (!root)
40              return 0;
41          int Lsum = dfs(root->left, sum);
42          int Rsum = dfs(root->right, sum);
43          *sum += abs(Lsum - Rsum);
44          return root->val + Lsum + Rsum;
45      }
    };
```