

给定一个二叉树，计算**整个树**的坡度。

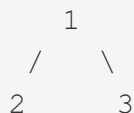
一个树的**节点的坡度**定义即为，该节点左子树的结点之和和右子树结点之和的**差的绝对值**。

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:



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。

思路，整棵树度的运算，说白了就是每个节点计算其左子树节点和与右子树节点和。那么我们返回左右子树节点和，在动态的统计度，当节点为空，返回0，使其叶子节点的度为0。

代码

```
/**
```

```
 * Definition for a binary tree node.
```

```
 * struct TreeNode {
```

```
 *     int val;
```

```
 *     TreeNode *left;
```

```
 *     TreeNode *right;
```

```
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
```

```
 * };
```

```
*/
```

```
class Solution {
```

```
public:
```

```
    int dfs(TreeNode* root, int &du) {
```

```
        if (root==NULL) {
```

```
        return 0;
    }
    int left=dfs(root->left, du);
    int right=dfs(root->right, du);
    du+=abs(left-right);
    return root->val+left+right;
}
int findTilt(TreeNode* root) {
    int du=0;
    dfs(root, du);
    return du;
}
};
```