# 题目

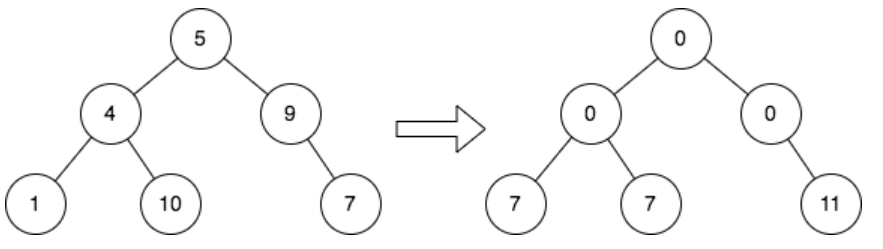
给你一棵二叉树的根root，请你将每个节点的值替换成该节点的所有堂兄弟节点值的和 。

如果两个节点在树中有相同的深度且它们的父节点不同，那么它们互为堂兄弟 。

请你返回修改值之后，树的根 root 。

注意，一个节点的深度指的是从树根节点到这个节点经过的边数。

示例 1：



输入：root = [5,4,9,1,10,null,7]

输出：[0,0,0,7,7,null,11]

解释：上图展示了初始的二叉树和修改每个节点的值之后的二叉树。

- 值为 5 的节点没有堂兄弟，所以值修改为 0 。

- 值为 4 的节点没有堂兄弟，所以值修改为 0 。

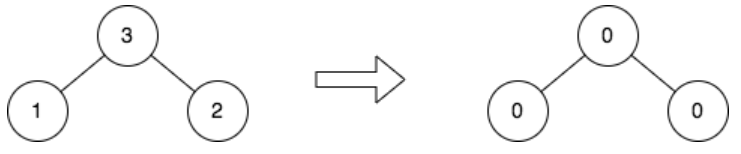
- 值为 9 的节点没有堂兄弟，所以值修改为 0 。

- 值为 1 的节点有一个堂兄弟，值为 7 ，所以值修改为 7 。

- 值为 10 的节点有一个堂兄弟，值为 7 ，所以值修改为 7 。

- 值为 7 的节点有两个堂兄弟，值分别为 1 和 10 ，所以值修改为 11 。

示例 2：



输入：root = [3,1,2]

输出：[0,0,0]

解释：上图展示了初始的二叉树和修改每个节点的值之后的二叉树。

- 值为 3 的节点没有堂兄弟，所以值修改为 0 。

- 值为 1 的节点没有堂兄弟，所以值修改为 0 。

- 值为 2 的节点没有堂兄弟，所以值修改为 0 。

提示：

树中节点数目的范围是 [1, 105] 。

1 <= Node.val <= 104

类似题目：

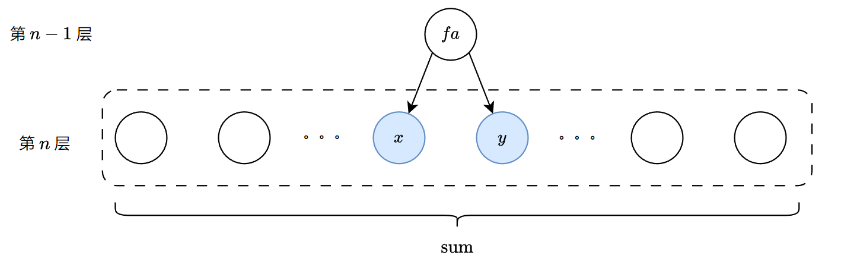
LeetCode993

# 分析

方法一：广度优先搜索

思路与算法：

题目要求将二叉树中每个节点的值替换为所有堂兄弟节点的和，而堂兄弟节点就是指那些和当前节点深度相同但父节点不同的节点。例如下图中，x的堂兄弟节点是第n层除去x和y的其他所有节点。假设第n层所有节点的和为sum，那么x的值应该被替换为sum−x−y。



在广度优先搜索的过程中，通过第n−1层去遍历第n层的节点时，可以顺便统计第n层节点的和sum。由于更新x的值时需要知道y的值（有可能不存在），因此需要通过n−1层对第n层进行第二次遍历，这时就可以使用sum−x−y更新x的值了。

在代码实现时，我们需要遍历每一层的节点两次，因此使用动态数组或链表表示的队列会更方便。

代码：

复杂度分析：

时间复杂度：O(n)，其中n为二叉树中节点的个数。

空间复杂度：O(n)。