2471. Minimum Number of Operations to Sort a Binary Tree by Level

My Submissions (/contest/weekly-contest-319/problems/minimum-number-of-operations-to-sort-a-binary-tree-by-level/submissions/)

Back to Contest (/contest/weekly-contest-319/)

User Accepted:	5297
User Tried:	6207
Total Accepted:	5639
Total Submissions:	9220
Difficulty:	Medium

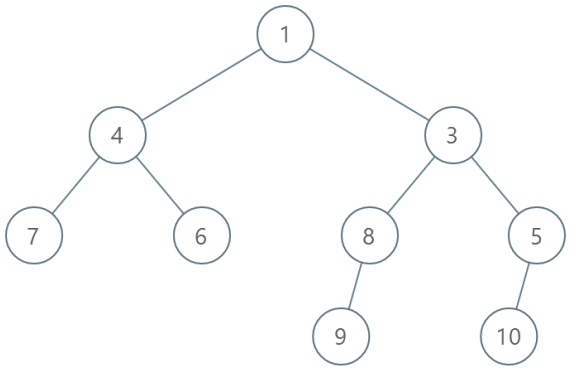
You are given the `root` of a binary tree with **unique values**.

In one operation, you can choose any two nodes **at the same level** and swap their values.

Return *the minimum number of operations needed to make the values at each level sorted in a **strictly increasing order***.

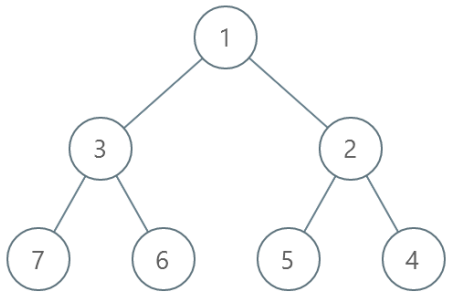
The **level** of a node is the number of edges along the path between it and the root node.

Example 1:



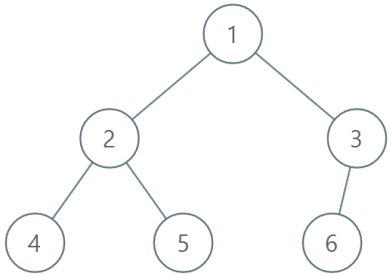

```
Input: root = [1,4,3,7,6,8,5,null,null,null,null,9,null,10]
Output: 3
Explanation:
- Swap 4 and 3. The 2nd level becomes [3,4].
- Swap 7 and 5. The 3rd level becomes [5,6,8,7].
- Swap 8 and 7. The 3rd level becomes [5,6,7,8].
We used 3 operations so return 3.
It can be proven that 3 is the minimum number of operations needed.
```

Example 2:




```
Input: root = [1,3,2,7,6,5,4]
Output: 3
Explanation:
- Swap 3 and 2. The 2nd level becomes [2,3].
- Swap 7 and 4. The 3rd level becomes [4,6,5,7].
- Swap 6 and 5. The 3rd level becomes [4,5,6,7].
We used 3 operations so return 3.
It can be proven that 3 is the minimum number of operations needed.
```

Example 3:




```
Input: root = [1,2,3,4,5,6]
Output: 0
Explanation: Each level is already sorted in increasing order so return 0.
```

Constraints:

- The number of nodes in the tree is in the range `[1, 10^5]`.
- `1 <= Node.val <= 10^5`
- All the values of the tree are **unique**.

Discuss (https://leetcode.com/problems/minimum-number-of-operations-to-sort-a-binary-tree-by-level/discuss)

JavaScript

```
const minimumOperations = (root) => {
    let d = levelOrder_BFS(root), res = 0;
    for (const a of d) res += minSwapSortArray(a);
    return res;
};

const minSwapSortArray = (a) => {
    let n = a.length, b = [...a].sort((x, y) => x - y), m = new Map(), res = 0;
    for (let i = 0; i < n; i++) m.set(a[i], i);
    for (let i = 0; i < n; i++) {
        if (a[i] != b[i]) {
            res++;
            let j = m.get(b[i]);
            m.set(a[i], j);
            m.set(b[i], i);
            [a[i], a[j]] = [a[j], a[i]];
        }
    }
    return res;
};

const levelOrder_BFS = (root) => {
    let data = [];
    getAllLevels(root, 0, data);
    return data;
};

const getAllLevels = (root, level, data) => {
    if (!root) return;
    if (level >= data.length) data.push([]);
    data[level].push(root.val);
    getAllLevels(root.left, level + 1, data);
    getAllLevels(root.right, level + 1, data);
};
```

Custom Testcase Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/842485082/) More Details > (/submissions/detail/842485082/)

Share your acceptance!