


[Description](#) [Discussion \(18\)](#) [Solutions \(520\)](#) [Submissions](#)

834. Sum of Distances in Tree

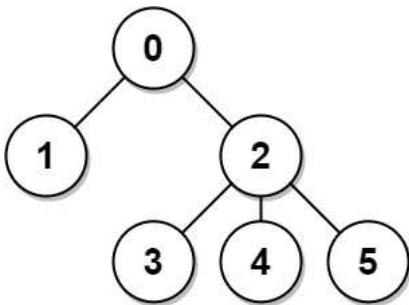
Hard Accepted 4.3K 99 Star Share
Companies

There is an undirected connected tree with n nodes labeled from 0 to $n - 1$ and $n - 1$ edges.

You are given the integer n and the array edges where $\text{edges}[i] = [a_i, b_i]$ indicates that there is an edge between nodes a_i and b_i in the tree.

Return an array answer of length n where $\text{answer}[i]$ is the sum of the distances between the i^{th} node in the tree and all other nodes.

Example 1:



Input: $n = 6$, $\text{edges} = [[0,1], [0,2], [2,3], [2,4], [2,5]]$

Output: $[8, 12, 6, 10, 10, 10]$

Explanation: The tree is shown above.

We can see that $\text{dist}(0,1) + \text{dist}(0,2) + \text{dist}(0,3) + \text{dist}(0,4) + \text{dist}(0,5)$ equals $1 + 1 + 2 + 2 + 2 = 8$.

Hence, $\text{answer}[0] = 8$, and so on.

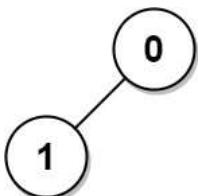
Example 2:



Input: $n = 1$, $\text{edges} = []$

Output: $[0]$

Example 3:



Input: $n = 2$, $\text{edges} = [[1,0]]$

Output: $[1, 1]$

Constraints:

- $1 \leq n \leq 3 * 10^4$
- $\text{edges.length} == n - 1$
- $\text{edges}[i].length == 2$

```

i C++ ▾ Auto
vector<unordered_set<int>>& tree,
7
8 for(const vector<int>& edge: edges){
9     const int u = edge[0];
10    const int v = edge[1];
11    tree[u].insert(v);
12    tree[v].insert(u);
13 }
14 postorder(tree, 0, -1, count, ans);
15 preorder(tree, 0, -1, count, ans);
16 return ans;
17 }
18 private:
19     void postorder(const vector<unordered_set<int>>& tree,
20                     vector<int>& count, vector<int>& ans){
21         for(const int child: tree[node]){
22             if(child == parent)
23                 continue;
24             postorder(tree, child, node, count, ans);
25             count[node] += count[child];
26             ans[node] += ans[child] + count[child];
27         }
28     }
29     void preorder(const vector<unordered_set<int>>& tree,
30                   vector<int>& count, vector<int>& ans){
31         for(const int child: tree[node]){
32             if(child == parent)
33                 continue;
34             ans[child] = ans[node] - count[child] + (tree.size() - 1 - count[child]);
35         }
36     }
  
```

[Console](#) ▾

[Run](#)
[Submit](#)

