Problem 3553: Minimum Weighted Subgraph With the Required Paths II

Problem Information

Difficulty: Hard
Acceptance Rate: 49.08%
Paid Only: No
Tags: Array, Tree, Depth-First Search

Problem Description

You are given an **undirected weighted** tree with \`n\` nodes, numbered from \`0\` to \`n - 1\`. It is represented by a 2D integer array \`edges\` of length \`n - 1\`, where \`edges[i] = [ui, vi, wi]\` indicates that there is an edge between nodes \`ui\` and \`vi\` with weight \`wi\`.

Additionally, you are given a 2D integer array \`queries\`, where \`queries[j] = [src1j, src2j, destj]\`.

Return an array \`answer\` of length equal to \`queries.length\`, where \`answer[j]\` is the **minimum total weight** of a subtree such that it is possible to reach \`destj\` from both \`src1j\` and \`src2j\` using edges in this subtree.

A **subtree** here is any connected subset of nodes and edges of the original tree forming a valid tree.

Example 1:

Input: edges = [[0,1,2],[1,2,3],[1,3,5],[1,4,4],[2,5,6]], queries = [[2,3,4],[0,2,5]]

Output: [12,11]

Explanation:

The blue edges represent one of the subtrees that yield the optimal answer.

![](https://assets.leetcode.com/uploads/2025/04/02/tree1-4.jpg)

* `answer[0]`: The total weight of the selected subtree that ensures a path from `src1 = 2` and `src2 = 3` to `dest = 4` is `3 + 5 + 4 = 12`.

* `answer[1]`: The total weight of the selected subtree that ensures a path from `src1 = 0` and `src2 = 2` to `dest = 5` is `2 + 3 + 6 = 11`.

Example 2:

Input: edges = [[1,0,8],[0,2,7]], queries = [[0,1,2]]

Output: [15]

Explanation:

![](https://assets.leetcode.com/uploads/2025/04/02/tree1-5.jpg)

* `answer[0]`: The total weight of the selected subtree that ensures a path from `src1 = 0` and `src2 = 1` to `dest = 2` is `8 + 7 = 15`.

Constraints:

* `3 <= n <= 105` * `edges.length == n - 1` * `edges[i].length == 3` * `0 <= ui, vi < n` * `1 <= wi <= 104` * `1 <= queries.length <= 105` * `queries[j].length == 3` * `0 <= src1j, src2j, destj < n` * `src1j`, `src2j`, and `destj` are pairwise distinct. * The input is generated such that `edges` represents a valid tree.

Code Snippets

C++:

```
class Solution {
public:
vector<int> minimumWeight(vector<vector<int>>& edges, vector<vector<int>>&
queries) {

}
};
```

Java:

```
class Solution {
public int[] minimumWeight(int[][] edges, int[][] queries) {

}
}
```

Python3:

```
class Solution:
def minimumWeight(self, edges: List[List[int]], queries: List[List[int]]) ->
List[int]:
```