6378. Minimize the Total Price of the Trips

My Submissions (/contest/weekly-contest-341/problems/minimize-the-total-price-of-the-trips/submissions/) | Back to Contest (/contest/weekly-contest-341/)

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Hard

There exists an undirected and unrooted tree with `n` nodes indexed from `0` to `n - 1`. You are given the integer `n` and a 2D integer array `edges` of length `n - 1`, where $edges[i] = [a_i, b_i]$ indicates that there is an edge between nodes a_i and b_i in the tree.

Each node has an associated price. You are given an integer array `price`, where `price[i]` is the price of the i^{th} node.

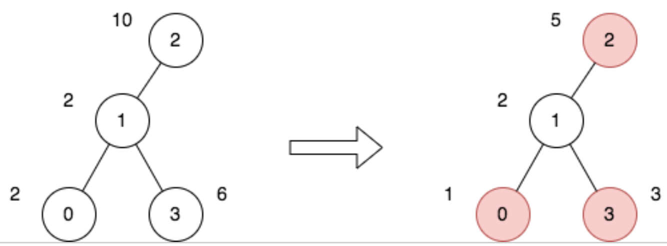
The **price sum** of a given path is the sum of the prices of all nodes lying on that path.

Additionally, you are given a 2D integer array `trips`, where $trips[i] = [start_i, end_i]$ indicates that you start the i^{th} trip from the node $start_i$ and travel to the node end_i by any path you like.

Before performing your first trip, you can choose some **non-adjacent** nodes and halve the prices.

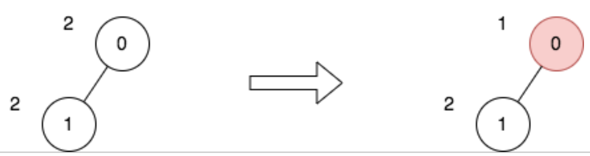
Return *the minimum total price sum to perform all the given trips*.

Example 1:




```
Input: n = 4, edges = [[0,1],[1,2],[1,3]], price = [2,2,10,6], trips = [[0,3],[2,1],[2,3]]
Output: 23
Explanation: The diagram above denotes the tree after rooting it at node 2. The first part shows the initial tree and the secon
For the 1st trip, we choose path [0,1,3]. The price sum of that path is 1 + 2 + 3 = 6.
For the 2nd trip, we choose path [2,1]. The price sum of that path is 2 + 5 = 7.
For the 3rd trip, we choose path [2,1,3]. The price sum of that path is 5 + 2 + 3 = 10.
The total price sum of all trips is 6 + 7 + 10 = 23.
It can be proven, that 23 is the minimum answer that we can achieve.
```

Example 2:




```
Input: n = 2, edges = [[0,1]], price = [2,2], trips = [[0,0]]
Output: 1
Explanation: The diagram above denotes the tree after rooting it at node 0. The first part shows the initial tree and the secon
For the 1st trip, we choose path [0]. The price sum of that path is 1.
The total price sum of all trips is 1. It can be proven, that 1 is the minimum answer that we can achieve.
```

Constraints:

- `1 <= n <= 50`
- `edges.length == n - 1`
- $0 <= a_i, b_i <= n - 1$
- `edges` represents a valid tree.
- `price.length == n`
- `price[i]` is an even integer.
- `1 <= price[i] <= 1000`
- `1 <= trips.length <= 100`
- $0 <= start_i, end_i <= n - 1$

JavaScript

```
const initializeGraph = (n) => { let g = []; for (let i = 0; i < n; i++) { g.push([]); } return g; };
const packUG = (g, edges) => { for (const [u, v] of edges) { g[u].push(v); g[v].push(u); } };
const initialize2DArray = (n, m) => [...Array(n)].map(() => Array(m).fill(0));

let g, cnt, dp, price;
const minimumTotalPrice = (n, edges, p, trips) => {
    g = initializeGraph(n), cnt = Array(n).fill(0), dp = initialize2DArray(n, 2), price = p;
    packUG(g, edges);
    for (const [u, v] of trips) tree_dp(u, -1, v);
    house_robber_III(0, -1);
    let res = 0;
    for (let i = 0; i < n; i++) res += cnt[i] * price[i];
    return res - Math.max(...dp[0]);
};

const tree_dp = (cur, par, dest) => {
    if (cur == dest) {
        cnt[cur]++;
        return true;
    }
    for (const child of g[cur]) {
        if (child != par && tree_dp(child, cur, dest)) {
            cnt[cur]++;
            return true;
        }
    }
    return false;
};

const house_robber_III = (cur, par) => {
    dp[cur] = [0, cnt[cur] * price[cur] / 2];
    for (const child of g[cur]) {
        if (child != par) {
            house_robber_III(child, cur)
            dp[cur][0] += Math.max(...dp[child]);
            dp[cur][1] += dp[child][0];
        }
    }
};
```

Custom Testcase Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/934548667/) More Details (/submissions/detail/934548667/)

Share your acceptance!