6139. Reachable Nodes With Restrictions

My Submissions (/contest/weekly-contest-305/problems/reachable-nodes-with-restrictions/submissions/) | Back to Contest (/contest/weekly-contest-305/)

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

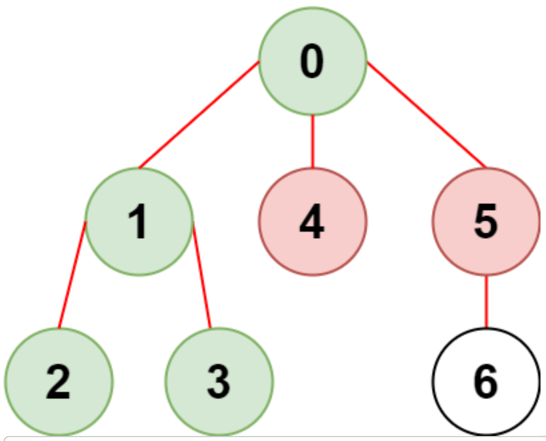
There is an undirected tree with `n` nodes labeled from `0` to `n - 1` and `n - 1` edges.

You are given 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. You are also given an integer array `restricted` which represents **restricted** nodes.

Return *the **maximum** number of nodes you can reach from node `0` without visiting a restricted node.*

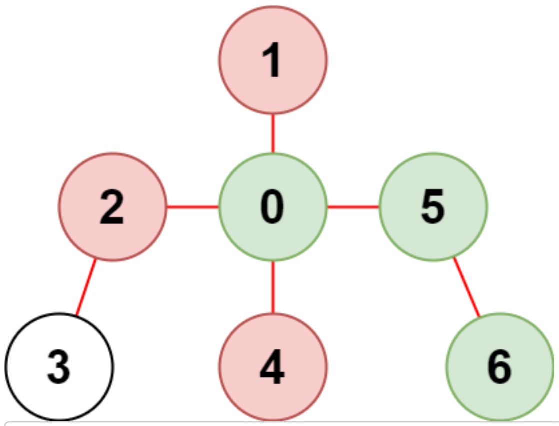
Note that node `0` will **not** be a restricted node.

Example 1:



```
Input: n = 7, edges = [[0,1],[1,2],[3,1],[4,0],[0,5],[5,6]], restricted = [4,5]
Output: 4
Explanation: The diagram above shows the tree.
We have that [0,1,2,3] are the only nodes that can be reached from node 0 without visiting a restricted node.
```

Example 2:



```
Input: n = 7, edges = [[0,1],[0,2],[0,5],[0,4],[3,2],[6,5]], restricted = [4,2,1]
Output: 3
Explanation: The diagram above shows the tree.
We have that [0,5,6] are the only nodes that can be reached from node 0 without visiting a restricted node.
```

Constraints:

- $2 <= n <= 10^5$
- `edges.length == n - 1`
- `edges[i].length == 2`
- $0 <= a_i, b_i < n$
- a_i != b_i
- `edges` represents a valid tree.

- `1 <= restricted.length < n`
- `1 <= restricted[i] < n`
- All the values of `restricted` are **unique**.

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 reachableNodes = (n, edges, restricted) => {
    let g = initializeGraph(n), notAllow = new Set(), visit = new Set(), q = [0], res = new Set([0]);
    packUG(g, edges);
    for (const x of restricted) notAllow.add(x);
    while (q.length) {
        let cur = q.shift();
        for (const child of g[cur]) {
            if (!notAllow.has(child) && !visit.has(child)) {
                res.add(child);
                visit.add(child);
                q.push(child);
            }
        }
    }
    return res.size;
};
```

Custom Testcase Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/767155859/) More Details (/submissions/detail/767155859/)

Share your acceptance!