5870. Smallest Missing Genetic Value in Each Subtree

My Submissions (/contest/weekly-contest-258/problems/smallest-missing-genetic-value-in-each-subtree/submissions/)

Back to Contest (/contest/weekly-contest-258/)

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

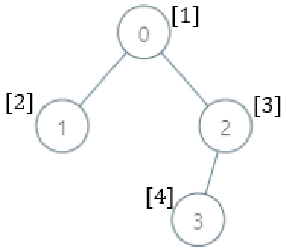
There is a **family tree** rooted at `0` consisting of `n` nodes numbered `0` to `n - 1`. You are given a **0-indexed** integer array `parents`, where `parents[i]` is the parent for node `i`. Since node `0` is the **root**, `parents[0] == -1`.

There are 10^5 genetic values, each represented by an integer in the **inclusive** range [1, 10^5]. You are given a **0-indexed** integer array `nums`, where `nums[i]` is a **distinct** genetic value for node `i`.

Return *an array* `ans` *of length* `n` *where* `ans[i]` *is the* ***smallest*** *genetic value that is* ***missing*** *from the subtree rooted at node* `i`.

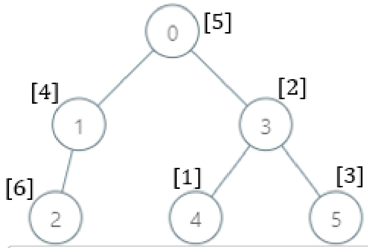
The **subtree** rooted at a node `x` contains node `x` and all of its **descendant** nodes.

Example 1:



```
Input: parents = [-1,0,0,2], nums = [1,2,3,4]
Output: [5,1,1,1]
Explanation: The answer for each subtree is calculated as follows:
- 0: The subtree contains nodes [0,1,2,3] with values [1,2,3,4]. 5 is the smallest missing value.
- 1: The subtree contains only node 1 with value 2. 1 is the smallest missing value.
- 2: The subtree contains nodes [2,3] with values [3,4]. 1 is the smallest missing value.
- 3: The subtree contains only node 3 with value 4. 1 is the smallest missing value.
```

Example 2:



```
Input: parents = [-1,0,1,0,3,3], nums = [5,4,6,2,1,3]
Output: [7,1,1,4,2,1]
Explanation: The answer for each subtree is calculated as follows:
- 0: The subtree contains nodes [0,1,2,3,4,5] with values [5,4,6,2,1,3]. 7 is the smallest missing value.
- 1: The subtree contains nodes [1,2] with values [4,6]. 1 is the smallest missing value.
- 2: The subtree contains only node 2 with value 6. 1 is the smallest missing value.
- 3: The subtree contains nodes [3,4,5] with values [2,1,3]. 4 is the smallest missing value.
- 4: The subtree contains only node 4 with value 1. 2 is the smallest missing value.
- 5: The subtree contains only node 5 with value 3. 1 is the smallest missing value.
```

Example 3:

```
Input: parents = [-1,2,3,0,2,4,1], nums = [2,3,4,5,6,7,8]
Output: [1,1,1,1,1,1,1]
Explanation: The value 1 is missing from all the subtrees.
```

Constraints:

- `n == parents.length == nums.length`
- `2 <= n <= 10`^{5}
- `0 <= parents[i] <= n - 1` for `i != 0`
- `parents[0] == -1`
- `parents` represents a valid tree.
- `1 <= nums[i] <= 10`^{5}
- Each `nums[i]` is distinct.

JavaScript

```javascript
const smallestMissingValueSubtree = (parents, nums) => {
    let n = parents.length, g = initializeGraph(n);
    for (let i = 1; i < n; i++) g[parents[i]].push(i);
    let res = Array(n).fill(0);
    const dfs = (x) => {
        let set = new Set(), miss = 1;
        for (const e of g[x]) {
            let [cset, cpos] = dfs(e);
            miss = Math.max(miss, cpos);
            set = unionSet(set, cset);
        }
        set.add(nums[x])
        while(set.has(miss)) miss++;
        res[x] = miss;
        return [set, miss];
    }
    dfs(0);
    return res;
};

const unionSet = (s1, s2) => {
    if (s1.size > s2.size) {
        for (const e of s2) s1.add(e);
        return s1;
    } else {
        for (const e of s1) s2.add(e);
        return s2;
    }
}

const initializeGraph = (n) => { let G = []; for (let i = 0; i < n; i++) { G.push([]); } return G; };
```

□ **Custom Testcase** Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/553474297/) More Details (/submissions/detail/553474297/)

Share your acceptance!

Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms)

Privacy Policy (/privacy)

🇺🇸 United States (/region)