582. Kill Process

Medium 785 15 Add to List Share

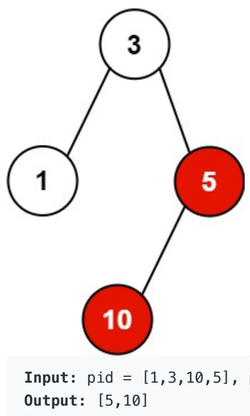
You have `n` processes forming a rooted tree structure. You are given two integer arrays `pid` and `ppid`, where `pid[i]` is the ID of the `i^th` process and `ppid[i]` is the ID of the `i^th` process's parent process.

Each process has only **one parent process** but may have multiple children processes. Only one process has `ppid[i] = 0`, which means this process has **no parent process** (the root of the tree).

When a process is **killed**, all of its children processes will also be killed.

Given an integer `kill` representing the ID of a process you want to kill, return *a list of the IDs of the processes that will be killed. You may return the answer in* ***any order***.

Example 1:




```
Input: pid = [1,3,10,5], ppid = [3,0,5,3], kill = 5
Output: [5,10]
Explanation: The processes colored in red are the processes that should be killed.
```

Example 2:

```
Input: pid = [1], ppid = [0], kill = 1
Output: [1]
```

Constraints:

- `n == pid.length`
- `n == ppid.length`
- `1 <= n <= 5 * 10^4`
- `1 <= pid[i] <= 5 * 10^4`
- `0 <= ppid[i] <= 5 * 10^4`
- Only one process has no parent.
- All the values of `pid` are **unique**.
- `kill` is **guaranteed** to be in `pid`.

Accepted 56,860 | Submissions 86,916

Seen this question in a real interview before? Yes No

Companies

0 ~ 6 months 6 months ~ 1 year 1 year ~ 2 years

Google | 3 Bloomberg | 3 Amazon | 3 Microsoft | 2

Related Topics

Array Hash Table Tree Depth-First Search Breadth-First Search

Java Autocomplete

```java
public class Solution {
    public List < Integer > killProcess(List < Integer > pid, List < Integer > ppid, int kill) {
        HashMap < Integer, List < Integer >> map = new HashMap < > ();
        for (int i = 0; i < ppid.size(); i++) {
            if (ppid.get(i) > 0) {
                List < Integer > l = map.getOrDefault(ppid.get(i), new ArrayList < Integer > ());
                l.add(pid.get(i));
                map.put(ppid.get(i), l);
            }
        }
        List < Integer > l = new ArrayList < > ();
        l.add(kill);
        getAllChildren(map, l, kill);
        return l;
    }
    public void getAllChildren(HashMap < Integer, List < Integer >> map, List < Integer > l, int kill) {
        if (map.containsKey(kill))
            for (int id: map.get(kill)) {
                l.add(id);
                getAllChildren(map, l, id);
            }
    }
}
```