Problem 851: Loud and Rich

Problem Information

Difficulty: Medium
Acceptance Rate: 62.73%
Paid Only: No
Tags: Array, Depth-First Search, Graph, Topological Sort

Problem Description

There is a group of `n` people labeled from `0` to `n - 1` where each person has a different amount of money and a different level of quietness.

You are given an array `richer` where `richer[i] = [ai, bi]` indicates that `ai` has more money than `bi` and an integer array `quiet` where `quiet[i]` is the quietness of the `ith` person. All the given data in richer are **logically correct** (i.e., the data will not lead you to a situation where `x` is richer than `y` and `y` is richer than `x` at the same time).

Return _an integer array_`answer` _where_`answer[x] = y` _if_`y` _is the least quiet person (that is, the person_`y` _with the smallest value of_`quiet[y]`_) among all people who definitely have equal to or more money than the person_`x`.

Example 1:

Input: richer = [[1,0],[2,1],[3,1],[3,7],[4,3],[5,3],[6,3]], quiet = [3,2,5,4,6,1,7,0] **Output:** [5,5,2,5,4,5,6,7] **Explanation:** answer[0] = 5. Person 5 has more money than 3, which has more money than 1, which has more money than 0. The only person who is quieter (has lower quiet[x]) is person 7, but it is not clear if they have more money than person 0. answer[7] = 7. Among all people that definitely have equal to or more money than person 7 (which could be persons 3, 4, 5, 6, or 7), the person who is the quietest (has lower quiet[x]) is person 7. The other answers can be filled out with similar reasoning.

Example 2:

Input: richer = [], quiet = [0] **Output:** [0]

Constraints:

* `n == quiet.length` * `1 <= n <= 500` * `0 <= quiet[i] < n` * All the values of `quiet` are **unique**. * `0 <= richer.length <= n * (n - 1) / 2` * `0 <= ai, bi < n` * `ai != bi` * All the pairs of `richer` are **unique**. * The observations in `richer` are all logically consistent.

Code Snippets

C++:

```
class Solution {
public:
vector<int> loudAndRich(vector<vector<int>>& richer, vector<int>& quiet) {

}
};
```

Java:

```
class Solution {
public int[] loudAndRich(int[][] richer, int[] quiet) {

}
}
```

Python3:

```
class Solution:
def loudAndRich(self, richer: List[List[int]], quiet: List[int]) ->
List[int]:
```