5929. Process Restricted Friend Requests

My Submissions (/contest/weekly-contest-267/problems/process-restricted-friend-requests/submissions/)

Back to Contest (/contest/weekly-contest-267/)

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

You are given an integer `n` indicating the number of people in a network. Each person is labeled from `0` to `n - 1`.

You are also given a **0-indexed** 2D integer array `restrictions`, where `restrictions[i] = [`x_i`, `y_i`]` means that person x_i and person y_i **cannot** become **friends**, either **directly** or **indirectly** through other people.

Initially, no one is friends with each other. You are given a list of friend requests as a **0-indexed** 2D integer array `requests`, where `requests[j] = [`u_j`, `v_j`]` is a friend request between person u_j and person v_j.

A friend request is **successful** if u_j and v_j can be **friends**. Each friend request is processed in the given order (i.e., `requests[j]` occurs before `requests[j + 1]`), and upon a successful request, u_j and v_j **become direct friends** for all future friend requests.

Return *a **boolean array** `result`, where each `result[j]` is `true` if the j^{th} friend request is **successful** or `false` if it is not.*

Note: If u_j and v_j are already direct friends, the request is still **successful**.

Example 1:

```
Input: n = 3, restrictions = [[0,1]], requests = [[0,2],[2,1]]
Output: [true,false]
Explanation:
Request 0: Person 0 and person 2 can be friends, so they become direct friends.
Request 1: Person 2 and person 1 cannot be friends since person 0 and person 1 would be indirect friends (1--2--0).
```

Example 2:

```
Input: n = 3, restrictions = [[0,1]], requests = [[1,2],[0,2]]
Output: [true,false]
Explanation:

Request 0: Person 1 and person 2 can be friends, so they become direct friends.
Request 1: Person 0 and person 2 cannot be friends since person 0 and person 1 would be indirect friends (0--2--1).
```

Example 3:

```
Input: n = 5, restrictions = [[0,1],[1,2],[2,3]], requests = [[0,4],[1,2],[3,1],[3,4]]
Output: [true,false,true,false]
Explanation:
Request 0: Person 0 and person 4 can be friends, so they become direct friends.
Request 1: Person 1 and person 2 cannot be friends since they are directly restricted.
Request 2: Person 3 and person 1 can be friends, so they become direct friends.
Request 3: Person 3 and person 4 cannot be friends since person 0 and person 1 would be indirect friends (0--4--3--1).
```

Constraints:

- `2 <= n <= 1000`
- `0 <= restrictions.length <= 1000`
- `restrictions[i].length == 2`
- `0 <=` x_i`,` y_i `<= n - 1`
- x_i `!=` y_i
- `1 <= requests.length <= 1000`
- `requests[j].length == 2`
- `0 <=` u_j`,` v_j `<= n - 1`
- u_j `!=` v_j

JavaScript ▾

```
/**
 * @param {number} n
 * @param {number[][]} restrictions
 * @param {number[][]} requests
 * @return {boolean[]}
 */
var friendRequests = function(n, restrictions, requests) {

};
```

☐ **Custom Testcase** Use Example Testcases

Run Submit

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

United States (/region)