LeetCode
Explore
Problems
Interview
New
Contest
Discuss
Store
🎁 LeetCoding Challenge + GIVEAWAY! 🎁
Description
Solution
Discuss (805)
Submissions
Java
Autocomplete

323. Number of Connected Components in an Undirected Graph

Medium 1659 53 Add to List Share

You have a graph of `n` nodes. You are given an integer `n` and an array `edges` where `edges[i] = [a_i, b_i]` indicates that there is an edge between `a_i` and `b_i` in the graph.

Return *the number of connected components in the graph*.

Example 1:

0
1
3
2
4

```
Input: n = 5, edges = [[0,1],[1,2],[3,4]]
Output: 2
```

Example 2:

0
1
3
2
4

```
Input: n = 5, edges = [[0,1],[1,2],[2,3],[3,4]]
Output: 1
```

Constraints:

- `1 <= n <= 2000`
- `1 <= edges.length <= 5000`
- `edges[i].length == 2`
- `0 <= a_i <= b_i < n`
- `a_i != b_i`
- There are no repeated edges.

Accepted 203,464 Submissions 339,678

Seen this question in a real interview before? Yes No

Companies

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

Amazon	23
LinkedIn	3
Facebook	3
Pinterest	2

Related Topics

Depth-First Search Breadth-First Search Union Find Graph

Similar Questions

Number of Islands	Medium
Graph Valid Tree	Medium
Number of Provinces	Medium
Paths in Maze That Lead to Same Room	Medium

```java
class Solution {

     private void dfs(List<Integer>[] adjList, int[] visited, int startNode) {
        visited[startNode] = 1;

        for (int i = 0; i < adjList[startNode].size(); i++) {
            if (visited[adjList[startNode].get(i)] == 0) {
                dfs(adjList, visited, adjList[startNode].get(i));
            }
        }
    }

    public int countComponents(int n, int[][] edges) {
        int components = 0;
        int[] visited = new int[n];

        List<Integer>[] adjList = new ArrayList[n];
        for (int i = 0; i < n; i++) {
            adjList[i] = new ArrayList<Integer>();
        }

        for (int i = 0; i < edges.length; i++) {
            adjList[edges[i][0]].add(edges[i][1]);
            adjList[edges[i][1]].add(edges[i][0]);
        }

        for (int i = 0; i < n; i++) {
            if (visited[i] == 0) {
                components++;
                dfs(adjList, visited, i);
            }
        }
        return components;
    }
}
```