https://course.acciojob.com/idle?question=c5f395d4-c8ed-4c3c-88e9-285a189bddca

- MEDIUM
- Max Score: 40 Points
-
-
-
-
-
-

Detect cycle in an undirected graph

Given an undirected graph with V vertices and E edges, check whether it contains any cycle or not.

Input Format

First line contains two integers V and E. Here V represent number of vertices and E represents number of edges.

Next E lines contain two integers representing an edge between them.

Output Format

For each student, standing at index i print the distance between the ith student and the student having height greater than ith student and standing towards the right of him/her.

Example 1

Input

```
5 5
1 0
0 2
```

```
2 1
0 3
3 4
```

Output

```
1
```

Example 2

Input

```
3 2
0 1
1 2
```

Output

```
0
```

Constraints

`1 ≤ V, E ≤ 200`

Topic Tags

- Graphs
- DFS

My code

```
// n java
import java.io.*;
import java.util.*;
```

```
class Solution {

        public static boolean dfs(ArrayList<ArrayList<Integer>> g, int
x, int p, boolean[] vis) {
                if(vis[x] == true) return false;

                vis[x] = true;
                boolean ans = true;
                for(int i = 0; i < g.get(x).size(); i++) {
                        int y = g.get(x).get(i);

                        if(y == p) continue;

                        boolean temp = dfs(g, y, x, vis);
                        ans = ans & temp;
                }

                return ans;
        }

    public static boolean isCycle(int V,
ArrayList<ArrayList<Integer>> adj) {
       // Your code here
                int n = V;
                boolean vis[] = new boolean[n];

                boolean ans = true;

                for(int i = 0; i < n; i++) {
                        if(vis[i] == false) {
```

```
                        boolean temp = dfs(adj, i, -1, vis);
                        ans = ans & temp;
                    }
                }

                return !ans;
        }
    }



    public class Main{
        public static void main(String args[])
        {
            Scanner sc = new Scanner(System.in);
            int N, E;
            N = sc.nextInt();
            E = sc.nextInt();
            ArrayList<ArrayList<Integer>> adj = new ArrayList<>();
            for(int i =0; i<N; i++) adj.add(i, new ArrayList<Integer>());
            for(int i =0; i<E; i++){
                int u = sc.nextInt();
                int v = sc.nextInt();
                adj.get(u).add(v);
                adj.get(v).add(u);
            }
            boolean ans = Solution.isCycle(N,adj);
            if(ans)
                System.out.println("1");
            else
            System.out.println("0");
    }
}
```