**Detect cycle in an undirected graph**

**Medium**

Given an undirected graph with V vertices and E edges, check whether it contains any cycle or not. Graph is in the form of adjacency list where adj[i] contains all the nodes ith node is having edge with.

**Example 1:**

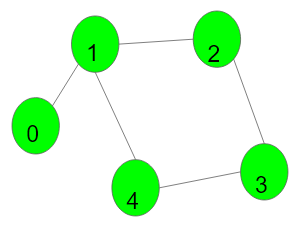
**Input:**

V = 5, E = 5

adj = {{1}, {0, 2, 4}, {1, 3}, {2, 4}, {1, 3}}

**Output:** 1

**Explanation:**



1->2->3->4->1 is a cycle.

**Example 2:**

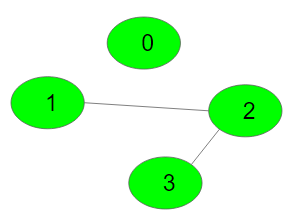
**Input:**

V = 4, E = 2

adj = {{}, {2}, {1, 3}, {2}}

**Output:** 0

**Explanation:**



No cycle in the graph.

**Expected Time Complexity:**O(V + E)  
**Expected Space Complexity:**O(V)

**Constraints:**  
1 ≤ V, E ≤ 105

**Company Tags**

[**Flipkart**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Flipkart) [**Amazon**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Amazon) [**Microsoft**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Microsoft) [**Samsung**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Samsung) [**MakeMyTrip**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=MakeMyTrip) [**Oracle**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Oracle) [**Adobe**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Adobe)

//{ Driver Code Starts

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class CodingMaxima {

public static void main(String[] args) throws IOException {

BufferedReader br =

new BufferedReader(new InputStreamReader(System.in));

int T = Integer.parseInt(br.readLine().trim());

while (T-- > 0) {

String[] s = br.readLine().trim().split(" ");

int V = Integer.parseInt(s[0]);

int E = Integer.parseInt(s[1]);

ArrayList<ArrayList<Integer>> adj = new ArrayList<>();

for (int i = 0; i < V; i++) adj.add(i, new ArrayList<Integer>());

for (int i = 0; i < E; i++) {

String[] S = br.readLine().trim().split(" ");

int u = Integer.parseInt(S[0]);

int v = Integer.parseInt(S[1]);

adj.get(u).add(v);

adj.get(v).add(u);

}

Solution obj = new Solution();

boolean ans = obj.isCycle(V, adj);

if (ans)

System.out.println("1");

else

System.out.println("0");

}

}

}

// } Driver Code Ends

class Solution {

// Function to detect cycle in an undirected graph.

public boolean isCycle(int V, ArrayList<ArrayList<Integer>> adj) {

boolean[] visited=new boolean[V];

for(int i=0;i<V;i++){

if(!visited[i] && dfs(i, visited,-1, adj))

return true;

}

return false;

}

private boolean dfs(int u, boolean[] visited, int parent, ArrayList<ArrayList<Integer>> adj){

visited[u]=true;

for(int v :adj.get(u)){

if(!visited[v]){

if(dfs(v, visited, u, adj)){

return true;

}

}

else if(v != parent){

return true;}

}

return false;

}

}