**Detect cycle in an undirected graph:-**

Given a Undirected Graph. Check whether it contains a cycle or not.

**Input:**  
The first line of the input contains an integer **'T'** denoting the number of test cases. Then **'T'** testcases follow. Each testcase consists of two lines. Description of testcases are as follows: The First line of each testcase contains two integers **'N' and 'M'**which denotes the no of vertices and no of edges respectively. The Second line of each test case contains **'M'** space separated pairs **u** and **v** denoting that there is a bidirectional  edge from **u** to **v** .

**Output:**  
The method should return **1** if there is a cycle else it should return **0**.

**User task:**  
You don't need to read input or print anything. Your task is to complete the function **isCyclic**which takes the Graph and the number of vertices as inputs and returns true if the given undirected graph contains any cycle. Else, it returns false.

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

**Constraints:**  
1 <= T <= 100  
2 <= N <= 104  
1 <= M <= (N\*(N-1))/2  
0 <= u, v <=  N-1  
Graph doesn't contain multiple edges and self loops.

**Example:  
Input:**  
3  
2 1  
0 1  
4 3  
0 1 1 2 2 3  
5 4  
0 1 2 3 3 4 4 2

**Output:**  
0  
0  
1

**Explanation:  
Testcase 1:** There is a graph with 2 vertices and 1 edge from 0 to 1. So there is no cycle.  
**Testcase 2:** There is a graph with 3 vertices and 3 edges from 0 to 1, 1 to 2 and 2 to 3.  
**Testcase 3:** There is a cycle in the given graph formed by vertices 2, 3 and 4.