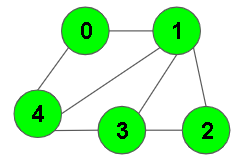
**Print adjacency list: -**

Easy Accuracy: 43.42% Submissions: 93K+ Points: 2

Given an undirected graph with **V**nodes and **E** **edges**, create and return an [adjacency list](https://www.geeksforgeeks.org/adjacency-list-meaning-definition-in-dsa/) of the graph. **0-based indexing** is followed everywhere.

**Example 1:**

**Input:**V = 5, E = 7  
edges = {(0,1),(0,4),(4,1),(4,3),(1,3),(1,2),(3,2)}



**Output:**

{{1,4},

{0,2,3,4},

{1,3},

{1,2,4},

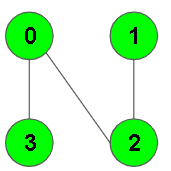
{0,1,3}}

**Explanation**:

Node 0 is connected to 1 and 4.  
Node 1 is connected to 0,2,3 and 4.  
Node 2 is connected to 1 and 3.  
Node 3 is connected to 1,2 and 4.  
Node 4 is connected to 0,1 and 3.

**Example 2:**

**Input:**V = 4, E = 3  
edges = {(0,3),(0,2),(2,1)}



**Output:**

{{2,3}

{2},

{0,1}

{0}}

**Explanation**:  
Node 0 is connected to 2 and 3.  
Node 1 is only connected to 2.  
Node 2 is connected to 0 and 1.  
Node 3 is only connected to 0.

**Your task:**  
You don't need to read input or print anything. Your task is to complete the function **printGraph()** which takes the integer V denoting the number of vertices and **edges** as input parameters and returns the list of list denoting the adjacency list.

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

**Constraints:**  
1 ≤ V, E ≤ 105

**Code: -**

//{ Driver Code Starts

#include <bits/stdc++.h>

using namespace std;

// } Driver Code Ends

class Solution {

public:

// Function to return the adjacency list for each vertex.

vector<vector<int>> printGraph(int V, vector<pair<int,int>>edges) {

vector<vector<int>> ans(V);

for(auto &p:edges){

ans[p.first].push\_back(p.second);

ans[p.second].push\_back(p.first);

}

return ans;

}

};

//{ Driver Code Starts.

int main() {

int tc;

cin >> tc;

while (tc--) {

int V, E;

cin >> V >> E;

vector<pair<int,int>>edges;

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

int u, v;

cin >> u >> v;

edges.push\_back({u,v});

}

Solution obj;

vector<vector<int>> adj = obj.printGraph(V, edges);

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

{ sort(adj[i].begin(),adj[i].end());

for(auto it:adj[i])

cout<<it<<" ";

cout<<endl;

}

}

return 0;

}

// } Driver Code Ends

**T.C: - O(V+E)**

**S.C: - O(V+E)**