#include <iostream>

#include <vector>

#include <list>

using namespace std;

class Graph {

private:

int numVertices;

vector<vector<int>> adjMatrix;

vector<list<int>> adjList;

public:

Graph(int vertices) : numVertices(vertices) {

// Initialize adjacency matrix with zeros

adjMatrix.resize(numVertices, vector<int>(numVertices, 0));

// Initialize adjacency list

adjList.resize(numVertices);

}

// Add an edge between vertices u and v

void addEdge(int u, int v) {

adjMatrix[u][v] = 1;

adjMatrix[v][u] = 1; // For undirected graph

adjList[u].push\_back(v);

adjList[v].push\_back(u); // For undirected graph

}

// Print adjacency matrix

void printAdjMatrix() {

cout << "Adjacency Matrix:" << endl;

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

for (int j = 0; j < numVertices; ++j) {

cout << adjMatrix[i][j] << " ";

}

cout << endl;

}

}

// Print adjacency lists

void printAdjList() {

cout << "Adjacency Lists:" << endl;

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

cout << i << " -> ";

for (int v : adjList[i]) {

cout << v << " ";

}

cout << endl;

}

}

};

int main() {

int numVertices, numEdges;

cout << "Enter the number of vertices in the graph: ";

cin >> numVertices;

// Create a graph with user-specified number of vertices

Graph graph(numVertices);

cout << "Enter the number of edges in the graph: ";

cin >> numEdges;

cout << "Enter the edges (vertex pairs separated by space):" << endl;

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

int u, v;

cin >> u >> v;

graph.addEdge(u, v);

}

// Print adjacency matrix

graph.printAdjMatrix();

// Print adjacency lists

graph.printAdjList();

return 0;

}