#include <stdio.h>

#include <stdlib.h>

// Function to create a graph with 'vertices' vertices

int\*\* createGraph(int vertices) {

int\*\* graph = (int\*\*)malloc(vertices \* sizeof(int\*));

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

graph[i] = (int\*)malloc(vertices \* sizeof(int));

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

graph[i][j] = 0; // Initialize all elements to 0 (no edges)

}

}

return graph;

}

// Function to add an edge to the graph

void addEdge(int\*\* graph, int src, int dest) {

// For an undirected graph, mark both src->dest and dest->src as 1

graph[src][dest] = 1;

graph[dest][src] = 1;

}

// Function to display the adjacency matrix

void displayAdjacencyMatrix(int\*\* graph, int vertices) {

printf("Adjacency Matrix:\n");

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

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

printf("%d ", graph[i][j]);

}

printf("\n");

}

}

int main() {

int vertices, edges;

printf("Enter the number of vertices: ");

scanf("%d", &vertices);

printf("Enter the number of edges: ");

scanf("%d", &edges);

int\*\* graph = createGraph(vertices);

printf("Enter the edges (format: source destination):\n");

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

int src, dest;

scanf("%d %d", &src, &dest);

addEdge(graph, src, dest);

}

displayAdjacencyMatrix(graph, vertices);

// Free memory

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

free(graph[i]);

}

free(graph);

return 0;

}

