#include <stdio.h>

#include <limits.h>

#define MAX\_VERTICES 100

#define INF INT\_MAX

void floydWarshall(int graph[MAX\_VERTICES][MAX\_VERTICES], int num\_vertices) {

int dist[MAX\_VERTICES][MAX\_VERTICES];

for (int i = 0; i < num\_vertices; i++) {

for (int j = 0; j < num\_vertices; j++) {

if (graph[i][j] == 0 && i != j) {

dist[i][j] = INF;

} else {

dist[i][j] = graph[i][j];

}

}

}

for (int k = 0; k < num\_vertices; k++) {

for (int i = 0; i < num\_vertices; i++) {

for (int j = 0; j < num\_vertices; j++) {

if (dist[i][k] != INF && dist[k][j] != INF &&

dist[i][k] + dist[k][j] < dist[i][j]) {

dist[i][j] = dist[i][k] + dist[k][j];

}

}

}

}

printf("Shortest distances between every pair of vertices:\n");

for (int i = 0; i < num\_vertices; i++) {

for (int j = 0; j < num\_vertices; j++) {

if (dist[i][j] == INF) {

printf("INF ");

} else {

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

}

}

printf("\n");

}

}

int main() {

int graph[MAX\_VERTICES][MAX\_VERTICES];

int num\_vertices, num\_edges;

for (int i = 0; i < MAX\_VERTICES; i++) {

for (int j = 0; j < MAX\_VERTICES; j++) {

graph[i][j] = 0;

}

}

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

scanf("%d", &num\_vertices);

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

scanf("%d", &num\_edges);

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

for (int i = 0; i < num\_edges; i++) {

int src, dest, weight;

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

graph[src][dest] = weight;

}

floydWarshall(graph, num\_vertices);

return 0;

}

Enter the number of vertices: 5

Enter the number of edges: 10 Is the graph directed? (1 for Yes, 0 for No): 1

Enter the edges (format: source destination weight):

0 1 4

1 4 6

2 3 3

2 0 2

1 2 1

3 2 1

4 3 4

0 3 5

4 0 1

3 4 2

Shortest distance matrix:

0 4 5 5 7

3 0 1 4 6

2 6 0 3 5

3 7 1 0 2

1 5 5 4 0