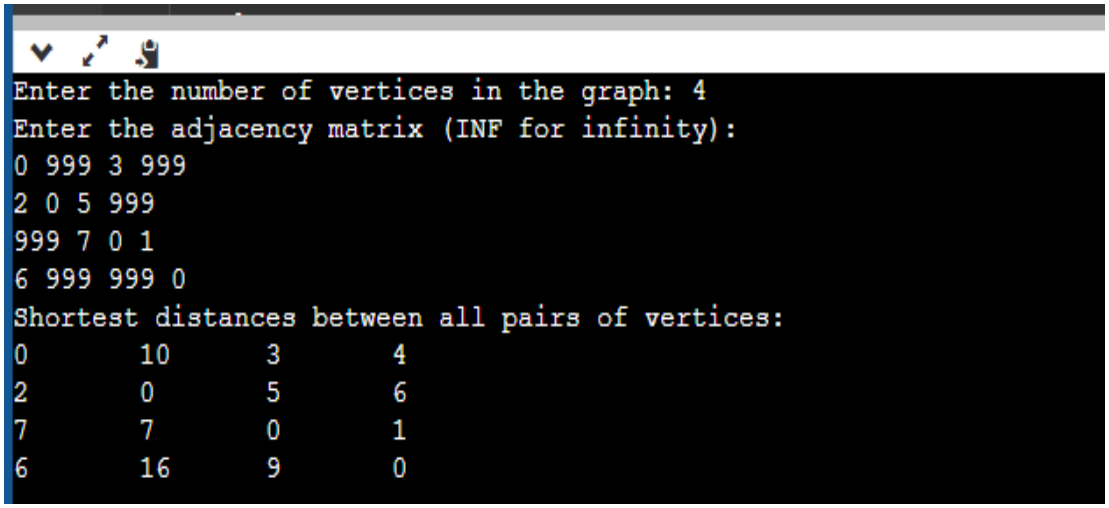
```
#include <stdio.h>
#include <limits.h>
#define INF 999
#define MAX_VERTICES 100
int min(int a, int b) {
    return (a < b) ? a : b;
}
void initialize(int graph[MAX_VERTICES][MAX_VERTICES], int
distance[MAX_VERTICES][MAX_VERTICES], int vertices) {
    for (int i = 0; i < vertices; i++) {
        for (int j = 0; j < vertices; j++) {
            distance[i][j] = graph[i][j];     }   }
}
void floydAlgorithm(int graph[MAX_VERTICES][MAX_VERTICES], int vertices) {
    int distance[MAX_VERTICES][MAX_VERTICES];
    initialize(graph, distance, vertices);
    for (int k = 0; k < vertices; k++) {
        for (int i = 0; i < vertices; i++) {
            for (int j = 0; j < vertices; j++) {
                if (distance[i][k] == INT_MAX || distance[k][j] == INT_MAX)
                    continue;
                distance[i][j] = min(distance[i][j], distance[i][k] + distance[k][j]);
          }      }    }
    printf("Shortest distances between all pairs of vertices:\n");
    for (int i = 0; i < vertices; i++) {
        for (int j = 0; j < vertices; j++) {
            if (distance[i][j] == INT_MAX)
                printf("INF\t");
          else
                printf("%d\t", distance[i][j]);
       }
    printf("\n");  } }
int main() {
    int graph[MAX_VERTICES][MAX_VERTICES], vertices;
    printf("Enter the number of vertices in the graph: ");
    scanf("%d", &vertices);
    printf("Enter the adjacency matrix (INF for infinity):\n");
    for (int i = 0; i < vertices; i++) {
        for (int j = 0; j < vertices; j++) {
            scanf("%d", &graph[i][j]);       }
    }
    floydAlgorithm(graph, vertices);
    return 0;
}
```


Enter the number of vertices in the graph: 4
Enter the adjacency matrix (INF for infinity):
0 999 3 999
2 0 5 999
999 7 0 1
6 999 999 0
Shortest distances between all pairs of vertices:
0 10 3 4
2 0 5 6
7 7 0 1
6 16 9 0