```
#include <stdio.h>
#include <limits.h>
#define SIZE 20
#define INF 9999
int minDistance(int dist[], int visited[], int n) {
    int min = INF, minIndex = -1;
    for (int v = 0; v < n; v++) {
        if (!visited[v] && dist[v] <= min) {
            min = dist[v];
            minIndex = v;
        }
    }
    return minIndex;
}
void dijkstra(int graph[SIZE][SIZE], int n, int src) {
    int dist[SIZE];
    int visited[SIZE];
    for (int i = 0; i < n; i++) {
        dist[i] = INF;
        visited[i] = 0;
    }
    dist[src] = 0;
    for (int count = 0; count < n - 1; count++) {
        int u = minDistance(dist, visited, n);
        visited[u] = 1;
        for (int v = 0; v < n; v++) {
            if (!visited[v] && graph[u][v] && dist[u] != INF &&
                dist[u] + graph[u][v] < dist[v]) {
                dist[v] = dist[u] + graph[u][v];
            }
        }
    }
    printf("Vertex\tDistance from Source %d\n", src);
    for (int i = 0; i < n; i++) {
        printf("%d\t\t%d\n", i, dist[i]);
    }
}
int main() {
    int graph[SIZE][SIZE], n, src;
    printf("Enter number of vertices: ");
    scanf("%d", &n);
    printf("Enter the adjacency matrix (0 if no edge):\n");
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
```

```
            scanf("%d", &graph[i][j]);
    printf("Enter the source vertex (0 to %d): ", n - 1);
    scanf("%d", &src);
    dijkstra(graph, n, src);
    return 0;
}
```