```
#include <stdio.h>
#include <limits.h>

#define MAX_VERTICES 100

int minDistance(int dist[], int sptSet[], int vertices) {
   int min = INT_MAX, minIndex;

   for (int v = 0; v < vertices; v++) {
      if (!sptSet[v] && dist[v] < min) {
         min = dist[v];
         minIndex = v;
      }
   }

   return minIndex;
}

void printSolution(int dist[], int vertices) {
   printf("Vertex \tDistance from Source\n");
   for (int i = 0; i < vertices; i++) {
      printf("%d \t%d\n", i, dist[i]);
   }
}

void dijkstra(int graph[MAX_VERTICES][MAX_VERTICES], int src, int vertices) {
   int dist[MAX_VERTICES];
   int sptSet[MAX_VERTICES


   for (int i = 0; i < vertices; i++) {
      dist[i] = INT_MAX;
      sptSet[i] = 0;
   }


   dist[src] = 0;


   for (int count = 0; count < vertices - 1; count++) {

      int u = minDistance(dist, sptSet, vertices);


      sptSet[u] = 1;

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

         if (!sptSet[v] && graph[u][v] && dist[u] != INT_MAX && dist[u] + graph[u][v] <
dist[v]) {
```

```
                dist[v] = dist[u] + graph[u][v];
            }
        }
    }

    printSolution(dist, vertices);
}

int main() {
    int vertices;

    printf("Input the number of vertices: ");
    scanf("%d", &vertices);

    if (vertices <= 0 || vertices > MAX_VERTICES) {
        printf("Invalid number of vertices. Exiting...\n");
        return 1;
    }

    int graph[MAX_VERTICES][MAX_VERTICES];

    printf("Input the adjacency matrix for the graph (use INT_MAX for infinity):\n");
    for (int i = 0; i < vertices; i++) {
        for (int j = 0; j < vertices; j++) {
            scanf("%d", &graph[i][j]);
        }
    }

    int source;
printf("Input the source vertex: ");
    scanf("%d", &source);

    if (source < 0 || source >= vertices) {
        printf("Invalid source vertex. Exiting...\n");
        return 1;
    }

    dijkstra(graph, source, vertices);

    return 0;
}
```