1. Dijkstra Algorithm

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

// Number of vertices in the graph
#define V 5

int minDistance(int dist[], bool VisitedNode[])
{
   // Initialize min value
   int min_distance = INT_MAX, min_node;

   for (int v = 0; v < V; v++)
      if (VisitedNode[v] == false && dist[v] <= min_distance)
         min_distance = dist[v], min_node = v;

   return min_node;
}

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


void dijkstra(int graph[V][V], int src)
{
   int dist[V];

   bool VisitedNode[V];

   for (int i = 0; i < V; i++)
      dist[i] = INT_MAX, VisitedNode[i] = false;
```

```
    // Distance of source vertex from itself is always 0
    dist[src] = 0;


    for (int count = 0; count < V; count++) {

        int u = minDistance(dist, VisitedNode);
        VisitedNode[u] = true;


        for (int v = 0; v < V; v++)

            if (VisitedNode[v]==false
                && graph[u][v]
                && dist[u] + graph[u][v] < dist[v])
                dist[v] = dist[u] + graph[u][v];
    }


    printSolution(dist);
}

int main()
{
    int graph[V][V] = { { 0, 5, 11},
                        { 0, 0, 3 },
                        { 0, 0, 0 },

                      };
    int source=2;
    dijkstra(graph, source);

    return 0;
}
```