Experiment No. 2.2

Student Name: Rishav Kumar **UID: 22MCC20039**
Branch: MCA **Section/Group: MCD -1/ Grp B**
Semester: I **Date of Performance: 18th Nov 22**
Subject Name: DAA LAB **Subject Code: 22CAP-646**

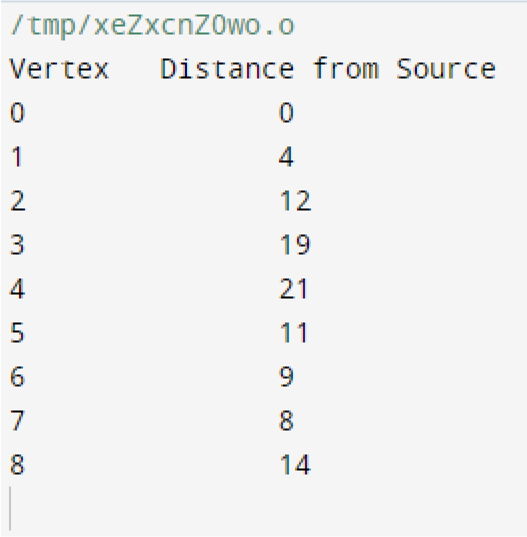
1. Aim/Overview of the practical:

From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm.

2. Code for experiment/practical:

```
#include <iostream>
using namespace std;
#include <limits.h>
#define V 9
int minDistance(int dist[], bool sptSet[])
{
   int min = INT_MAX, min_index;

   for (int v = 0; v < V; v++)
      if (sptSet[v] == false && dist[v] <= min)
         min = dist[v], min_index = v;

   return min_index;
}
void printSolution(int dist[])
{
   cout << "Vertex \t Distance from Source" << endl;
   for (int i = 0; i < V; i++)
      cout << i << " \t\t\t\t" << dist[i] << endl;
}
void dijkstra(int graph[V][V], int src)
{
   int dist[V];
   bool sptSet[V];
   for (int i = 0; i < V; i++)
      dist[i] = INT_MAX, sptSet[i] = false;
   dist[src] = 0;
   for (int count = 0; count < V - 1; count++) {
      int u = minDistance(dist, sptSet);
      sptSet[u] = true;

      for (int v = 0; v < V; 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);
}
int main()
{
   int graph[V][V] = { { 0, 4, 0, 0, 0, 0, 0, 8, 0 },
                  { 4, 0, 8, 0, 0, 0, 0, 11, 0 },
                  { 0, 8, 0, 7, 0, 4, 0, 0, 2 },
                  { 0, 0, 7, 0, 9, 14, 0, 0, 0 },
                  { 0, 0, 0, 9, 0, 10, 0, 0, 0 },
                  { 0, 0, 4, 14, 10, 0, 2, 0, 0 },
                  { 0, 0, 0, 0, 0, 2, 0, 1, 6 },
                  { 8, 11, 0, 0, 0, 0, 1, 0, 7 },
                  { 0, 0, 2, 0, 0, 0, 6, 7, 0 } };
   dijkstra(graph, 0);

   return 0;
}
```

3. Output:



```
/tmp/xeZxcnZ0wo.o
Vertex   Distance from Source
0               0
1               4
2               12
3               19
4               21
5               11
6               9
7               8
8               14
```

************************************** **THE END** ***