Experiment 3.2

Student Name: Jitesh Kumar **UID: 20BCS2334**
Branch: CSE **Section/Group: 20BCSWM-903(A)**
Semester: 5 **Date of Performance: 31/10/22**
Subject Name: Design and Analysis of Algorithms Lab
Subject Code: 20CSP-312

1. Aim: Code and analyze to find shortest paths in a graph with positive edge weights using Dijkstra's algorithm.

2. Software used: Visual Studio IDE, GCC

3. Algorithm/pseudo code:

Dijkstra's Algorithm (G, w, s)

1. INITIALIZE - SINGLE - SOURCE (G, s)
2. S←∅
3. Q←V [G]
4. while Q ≠ ∅
5. do u ← EXTRACT - MIN (Q)
6. S ← S U {u}
7. for each vertex v ∈ Adj [u]
8. do RELAX (u, v, w)

4. Code:

```
#include <bits/stdc++.h>
using namespace std;

#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 " << 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;
}
```

4. Output:

The running time of Dijkstra's algorithm on a graph with edges E and vertices V can be expressed as a function of |E| and |V| using the Big - O notation. The simplest implementation of the Dijkstra's algorithm stores vertices of set Q in an ordinary linked list or array, and operation Extract - Min (Q) is simply a linear search through all vertices in Q. In this case, the running time is O (|V2|+|E|=O(V2).

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

Learning outcomes (What I have learnt):

1. Concept of greedy strategy.
2. Algorithm of DFS.
3. Complexity of Shortest Path in a maze.