OSCN LAB 2

MOHAMMED NIZAMUDDIN

M.TECH CSE

2503B05144

Write a C++ program to implement **Dijkstra's Single Source Shortest Path Algorithm** for a given weighted, undirected graph using an **adjacency matrix representation**.

1. **Problem Setup**
    - We have **9 vertices** (0 to 8)

// A C++ program for Dijkstra's single source shortest path

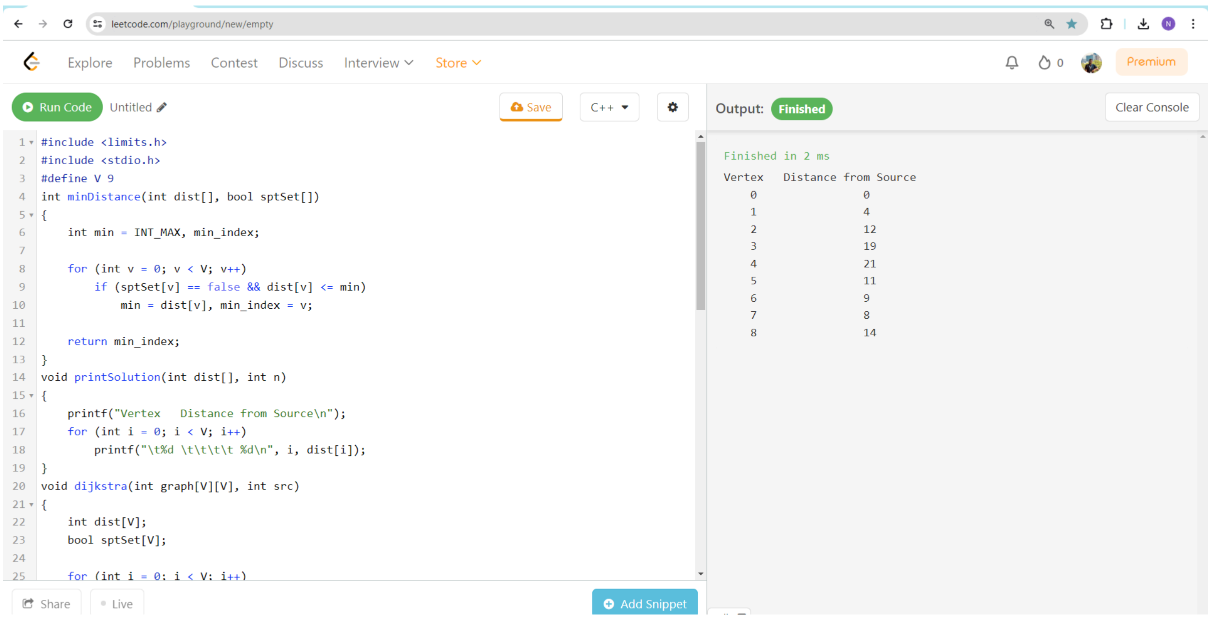
// algorithm. The program is for adjacency matrix

// representation of the graph

CODE:

```
#include <limits.h>

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

}

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;

}
```



```
#include <limits.h>
#include <stdio.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[], int n)
{
    printf("Vertex   Distance from Source\n");
    for (int i = 0; i < V; i++)
        printf("\t%d \t\t\t\t %d\n", i, dist[i]);
}
void dijkstra(int graph[V][V], int src)
{
    int dist[V];
    bool sptSet[V];

    for (int i = 0; i < V; i++)
```

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