DSA Lab 8

Selvakumar – 22MAI1004

**Perform Dijkstra’s algorithm**

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#define MAX 100

int minDistance(int dist[], bool sptSet[], int n)

{

int min = 9999, min\_index;

for (int v = 0; v < n; 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 \t\t Distance from Source\n");

for (int i = 0; i < n; i++)

{

printf("%d \t\t %d\n", i, dist[i]);

}

}

void dijkstra(int graph[][MAX], int n, int src)

{

int dist[MAX];

bool sptSet[MAX];

for (int i = 0; i < n; i++)

{

dist[i] = 9999, sptSet[i] = false;

}

dist[src] = 0;

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

{

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

sptSet[u] = true;

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

{

if (!sptSet[v] && graph[u][v] && dist[u] != 9999 && dist[u] + graph[u][v] < dist[v])

{

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

}

}

}

printSolution(dist, n);

}

int main()

{

int graph[MAX][MAX], n, src;

printf("Enter the number of vertices: ");

scanf("%d", &n);

printf("Enter the adjacency matrix: ");

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

{

scanf("%d", &graph[i][j]);

}

}

printf("Enter the starting vertex: ");

scanf("%d", &src);

dijkstra(graph, n, src);

return 0;

}

**Output**

**Text

Description automatically generated**