DSA Lab 9

Selvakumar – 22MAI1004

**Perform Minimum Spanning Tree**

#include <stdbool.h>

#include <stdio.h>

#include <stdlib.h>

#define MAX 100

int minKey(int key[], bool mstSet[], int n) {

int min = 9999, min\_index;

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

if (mstSet[v] == false && key[v] < min) {

min = key[v];

min\_index = v;

}

}

return min\_index;

}

void printMST(int parent[], int graph[][MAX], int n) {

printf("Edge \tWeight");

for (int i = 1; i < n; i++) {

printf("\n%d - %d \t%d", parent[i], i, graph[i][parent[i]]);

}

}

void primMST(int graph[][MAX], int n) {

int parent[MAX];

int key[MAX];

bool mstSet[MAX];

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

key[i] = 9999;

mstSet[i] = false;

}

key[0] = 0;

parent[0] = -1;

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

int u = minKey(key, mstSet, n);

mstSet[u] = true;

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

if (graph[u][v] && mstSet[v] == false && graph[u][v] < key[v]) {

parent[v] = u;

key[v] = graph[u][v];

}

}

}

printMST(parent, graph, n);

}

int main() {

int graph[MAX][MAX], n;

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]);

}

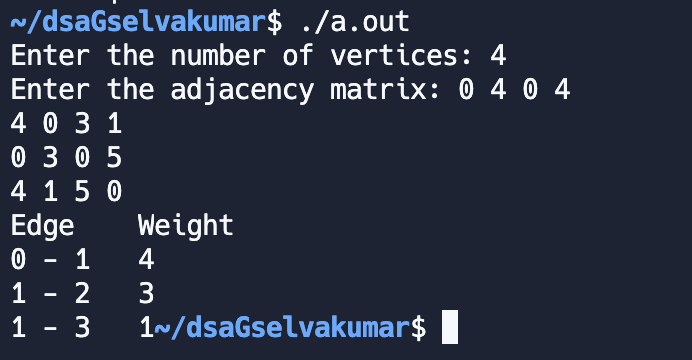
}

primMST(graph, n);

return 0;

}

**Output**

****