#include <stdio.h>

#include <limits.h>

#define MAX 100

#define INF 99999

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

int dist[MAX], visited[MAX], parent[MAX];

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

dist[i] = INF;

visited[i] = 0;

parent[i] = -1;

}

dist[src] = 0;

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

int min = INF, u=-1;

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

if (!visited[v] && dist[v] <= min) {

min = dist[v];

u = v;

}

visited[u] = 1;

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

if (!visited[v] && graph[u][v] && dist[u] + graph[u][v] < dist[v]) {

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

parent[v] = u;

}

}

int path[MAX], path\_len = 0, temp = target;

while (temp != -1) {

path[path\_len++] = temp;

temp = parent[temp];

}

printf("Shortest Path: ");

for (int i = path\_len - 1; i >= 0; i--) {

printf("%d", path[i] + 1);

if (i != 0) printf(" to ");

}

printf("\nTotal Cost: %d\n", dist[target]);

}

int main() {

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

printf("Enter number of nodes:\n");

scanf("%d", &n);

printf("Enter weight of all the paths in adjacency matrix form:\n");

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

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

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

printf("Enter the source node (1-based index):\n");

scanf("%d", &src);

printf("Enter the target node (1-based index):\n");

scanf("%d", &target);

dijkstra(graph, n, src - 1, target - 1);

return 0;

}

A computer screen shot of a black screen

AI-generated content may be incorrect.