**Lab Program 8**

**Q:- Implement Dijkstra’s algorithm to compute the shortest path for a given**

**topology.**

#include <bits/stdc++.h>

using namespace std;

int a[30][30], source, dist[30], path[30];

void dijkstar(int a[][30], int n)

{

set<pair<int, int>> s;

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

{

dist[i] = a[source][i];

path[i] = source;

s.insert({dist[i], i});

}

while (!s.empty())

{

pair<int, int> t = \*s.begin();

s.erase(s.begin());

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

{

if (dist[i] > t.first + a[t.second][i])

{

dist[i] = dist[t.second] + a[t.second][i];

path[i] = t.second;

s.insert({dist[i], i});

}

}

}

}

int main()

{

int n;

cout << "Enter the no. of vertices :" << endl;

cin >> n;

cout << "Enter the adjacency matrix(Enter 9999 for infinity):" << endl;

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

{

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

{

cin >> a[i][j];

}

}

cout << "Enter the source vertex :" << endl;

cin >> source;

cout << "The shortest paths from vertex ' " << source << " ' are :" << endl;

cout << "Vertex paths" << endl;

dijkstar(a, n);

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

{

int k = i;

while (k != source)

{

cout << k << " <- ";

k = path[k];

}

cout << source << " = ";

cout << "Path cost:" << dist[i] << endl;

}

return 0;

}

