**Assignment number: 6**

**Subject: ADVANCED DATA STRUCTURES LAB**

Name: ***RIA MITTAL***

Class: ***SECOND YEAR ENGINEERING***

Division: ***B***

Roll no: ***222008***

Batch: ***B1***

**PROBLEM STATEMENT:**

Tour operator organizes guided bus trips across the Maharashtra. Tourists may have different preferences. Tour operator offers a choice from many different routes. Every day the bus moves from starting city S to another city F as chosen by client. On this way, the tourists can see the sights alongside the route travelled from S to F. Client may have preference to choose route. There is a restriction on the routes that the tourists may choose from, the bus has to take a short route from S to F or a route having one distance unit longer than the minimal distance. Two routes from S to F are considered different if there is at least one road from a city A to a city B which is part of one route, but not of the other route.

**Code:**

#include<iostream>

using namespace std;

# define INF 100000

class dijk

{

int cities;

int start,end;

int \*v;

int \*parent,\*dist;

int cost[100][100];

int tour[100][100];

public:

int accept();

int locations();

int display();

int dijkstras();

};

int dijk::accept()

{

cout<<"\nenter the total number of cities : ";

cin>>cities;

v=new int[cities];

parent=new int[cities];

dist=new int[cities];

cout<<"\nenter the distances from the cities to each other\n";

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

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

cin>>tour[i][j];

for(int i=0;i<cities;i++) //to make diagonal null...

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

tour[i][i]=0;

}

int dijk::display()

{

cout<<"\nthe distances you entered are \n";

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

{

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

cout<<tour[i][j]<<" ";

cout<<endl;

}

}

int dijk::locations()

{

cout<<"\nenter the starting city : ";

cin>>start;

if(start>=cities)

cout<<"\ninvalid input city\n";

}

int dijk::dijkstras()

{

int count;

int next,min;

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

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

if(tour[i][j]==0)

cost[i][j]=INF;

else

cost[i][j]=tour[i][j];

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

{

dist[i]=cost[start][i];

parent[i]=start;

v[i]=0;

}

dist[start]=0;

v[start]=1;

count=1;

while(count<cities-1)

{

min=INF;

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

if(dist[i]<min && !v[i])

{

min=dist[i];

next=i;

}

v[next]=1;

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

if(!v[i])

if(min+cost[next][i]<dist[i])

{

dist[i]=min+cost[next][i];

parent[i]=next;

}

count++;

}

cout<<"\nenter the ending city : ";

cin>>end;

if(end>=cities)

{

cout<<"\n---------invalid input city---------\n";

cout<<"\nenter the ending city : ";

cin>>end;

}

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

if(i==end)

{

cout<<"\ndistance of the node is "<<dist[i]<<" "<<endl;

cout<<"\the path is "<<i;

int j=i;

do

{

j=parent[j];

cout<<"<--"<<j;

}while(j!=start);

}

cout<<endl;

}

int main()

{

dijk obj;

obj.accept();

obj.display();

obj.locations();

obj.dijkstras();

return 0;

}

**OUTPUT:**

enter the total number of cities : 3

enter the distances from the cities to each other

1

2

3

4

5

6

7

8

9

the distances you entered are

0 2 3

4 0 6

7 8 0

enter the starting city : 1

enter the ending city : 0

distance of the node is 4

he path is 0<--1

enter the total number of cities : 4

enter the distances from the cities to each other

12 45 12 0

21 0 122 3

7 53 3 1

12 76 23 0

the distances you entered are

0 45 12 0

21 0 122 3

7 53 0 1

12 76 23 0

enter the starting city : 2

enter the ending city : 1

distance of the node is 52

he path is 1<--0<--2