#include <iostream> using namespace std; #include<algorithm> #include<string>

#define INFINITY 99999;

class Graph

{

int q[20][20],w[20][20],n;

string city[20]; public:

void initalize()

{

cout<<"\n enter number of cities: "; cin>>n;

cout<<"\n enter names of cities: "; for(int i=0;i<n;i++)

{

cin>>city[i];

}

cout<<"\n enter weighted matrix:\n"; for(int i=0;i<n;i++)

{

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

{

cout<<"\n enter distance between "<<city[i]<<" to "<<city[j]; cin>>w[i][j];

}

}

}

void displayWeightedMatrix()

{

cout<<"\n Weighted Matrix: \n "; for(int i=0;i<n;i++)

{

cout<<" "<<city[i];

}

cout<<"\n";

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

{

cout<<city[i]; for(int j=0;j<n;j++)

{

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

}

cout<<"\n";

}

}

int index(string str)

{

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

{

if(str==city[i])

{

break;

}

}

return i;

}

void dijsktra()

{

string source,dest; int ch,si,di;

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

{

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

{

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

{

q[i][j]=INFINITY;

}

else

{

q[i][j]=w[i][j];

}

}

}

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

{

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

{

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

{

q[i][j]=min(q[i][j],q[i][k]+q[k][j]);

}

}

}

cout<<"\n Shortest Path Matrix:\n "; for(int i=0;i<n;i++)

{

cout<<" "<<city[i];

}

cout<<"\n";

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

{

cout<<city[i]; for(int j=0;j<n;j++)

{

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

}

cout<<"\n";

}

do

{

cout<<"\n 1. One source to one destination Path."; cout<<"\n 2. One source to all destination."; cout<<"\n 3. Exit.";

cout<<"\n Enter your choice: ";

cin>>ch; switch(ch)

{

case 1:

cout<<"\n enter source and destination: "; cin>>source>>dest;

si=index(source); di=index(dest); for(int i=0;i<n;i++)

{

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

{

if(i==si && j==di)

{

"<<q[i][j];

cout<<"\nShortest distance from "<<city[si]<<" to "<<city[di]<<" is:

}

}

cout<<"\n";

}

break; case 2:

cout<<"\n enter source: "; cin>>source; si=index(source);

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

{

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

{

if(i==si)

{

"<<q[i][j];

cout<<"\nShortest distance from "<<city[si]<<" to "<<city[j]<<" is:

}

}

cout<<"\n";

}

break;

}

}while(ch!=3);

}

};

int main()

{

Graph g; g.initalize();

g.displayWeightedMatrix(); g.dijsktra();

return 0;

}

OUTPUT:

ENTER THE NUMBER OF CITIES:5

ENTER THE NAMES OF CITIES:pune

mumbai indore jaipur satara

ENTER WEIGHTED MATRIX:

ENTER DISTANCE BETWEENpunetopune0

ENTER DISTANCE BETWEENpunetomumbai7

ENTER DISTANCE BETWEENpunetoindore8

ENTER DISTANCE BETWEENpunetojaipur9

ENTER DISTANCE BETWEENpunetosatara3

ENTER DISTANCE BETWEENmumbaitopune7

ENTER DISTANCE BETWEENmumbaitomumbai0

ENTER DISTANCE BETWEENmumbaitoindore1

ENTER DISTANCE BETWEENmumbaitojaipur8

ENTER DISTANCE BETWEENmumbaitosatara10

ENTER DISTANCE BETWEENindoretopune8

ENTER DISTANCE BETWEENindoretomumbai1

ENTER DISTANCE BETWEENindoretoindore0

ENTER DISTANCE BETWEENindoretojaipur7

ENTER DISTANCE BETWEENindoretosatara2

ENTER DISTANCE BETWEENjaipurtopune9

ENTER DISTANCE BETWEENjaipurtomumbai8

ENTER DISTANCE BETWEENjaipurtoindore7

ENTER DISTANCE BETWEENjaipurtojaipur0

ENTER DISTANCE BETWEENjaipurtosatara6

ENTER DISTANCE BETWEENsataratopune3

ENTER DISTANCE BETWEENsataratomumbai10

ENTER DISTANCE BETWEENsataratoindore2

ENTER DISTANCE BETWEENsataratojaipur6

ENTER DISTANCE BETWEENsataratosatara0

Weighted Matrix:

punemumbaiindorejaipursatara

pune07893

mumbai701810

indore81072

jaipur98706

satara310260

Shortest Path Matrix:

punemumbaiindorejaipursatara

pune66593

mumbai62183

indore51272

jaipur987126

satara33264

1.One source to one destination path.

2.One source to all destination .

3.Exit.

Enter your choice:1

ENTER SOURCE AND DESTINATION:mumbai satara

Shortest distance frommumbaitosatarais:3

ENTER SOURCE :mumbai

Shortest distance frommumbaitopuneis:6

Shortest distance frommumbaitomumbaiis:2

Shortest distance frommumbaitoindoreis:1

Shortest distance frommumbaitojaipuris:8

Shortest distance frommumbaitosatarais:3