**Roll No:-**

**Assignment No:-**

**Assignment Name:- Write a program to find shortest path using single source shortest path.**

#include<iostream.h>

#include<conio.h>

class GRAPH

{

private:

int COST[10][10],n,DIST[10];

public:

GRAPH(int);

void READ\_GRAPH();

void SHOW\_GRAPH();

void SHORTEST\_PATH(int v);

void SHOW\_DIST();

};

GRAPH::GRAPH(int par)

{

n=par;

}

void GRAPH::READ\_GRAPH()

{

cout<<"Enter COST ADJ matrix og GRAPH: ";

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

{

cout<<"\n";

for(int j=1;j<=n;j++)

cin>>COST[i][j];

}

}

void GRAPH::SHOW\_GRAPH()

{

cout<<"GRAPH is: ";

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

{

cout<<endl;

for(int j=1;j<=n;j++)

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

}

}

void GRAPH::SHORTEST\_PATH(int v)

{

//step1

int DIST[10],S[10]; //initialize S[] & DIST[]

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

{

S[i]=0; DIST[i]=COST[v][i];

}

//step2

S[v]=1; DIST[v]=0;

//step3

for(int num=2;num<=n;num++)

{

//choose u such that min of all DIST[w]

//update DIST[]

int min=9999,u;

for(int w=1;w<=n;w++)

{

if(S[w]==0 && DIST[w]<min)

{

min=DIST[w];

u=w;

}

}

S[u]=1;

//update DIST[]

for(w=1;w<=n;w++)

{

if(S[w]==0)

{

if(DIST[w]<DIST[u]+COST[u][w])

DIST[w]=DIST[w];

else

DIST[w]=DIST[u]+COST[u][w];

}

}

}

}

void GRAPH::SHOW\_DIST()

{

int v;

cout<<"source \t DIST";

for(int w=1;w<=n;w++)

{

cout<<v<<"\t"<<w<<"\t"<<COST[w]<<"\n";

}

}

void main()

{

int n;

clrscr();

cout<<"Enter the node";

cin>>n;

GRAPH obj(n);

obj.READ\_GRAPH();

obj.SHOW\_GRAPH();

obj.SHORTEST\_PATH(1);

obj.SHOW\_DIST();

getch();

}