**/\*11.From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra’s algorithm.\*/**

#include<stdio.h>

#include<conio.h>

#define INF 9999

void dijkstra(int a[10][10],int n,int src);

void main()

{

int a[10][10],i,j,n,u;

clrscr();

printf("enter number of vertices:\n");

scanf("%d",&n);

printf("enter the adjacency matrix:\n");

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

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

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

printf("\n enter the src node:\n");

scanf("%d",&u);

dijkstra(a,n,u);

getch();

}

void dijkstra(int a[10][10],int n,int src)

{

int cost[10][10],dis[10],pred[10];

int vis[10],cnt,mindis,nextnode,i,j;

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

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

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

cost[i][j]=INF;

else

cost[i][j]=a[i][j];

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

{

dis[i]=cost[src][i];

pred[i]=src;

vis[i]=0;

}

dis[src]=0;

vis[src]=1;

cnt=1;

while(cnt<n-1)

{

mindis=INF;

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

if(dis[i]<mindis &&! vis[i])

{

mindis=dis[i];

nextnode=i;

}

vis[nextnode]=1;

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

if(!vis[i])

if(mindis+cost[nextnode][i]<dis[i])

{

dis[i]=mindis+cost[nextnode][i];

pred[i]=nextnode;

}

cnt++;

}

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

if(i!=src)

{

printf("\n distance of node%d=%d\n",i,dis[i]);

printf("\n path=%d\n",i);

j=i;

do

{

j=pred[j];

printf("<-%d",j);

}while (j!=src);

}

}

**/\*OUTPUT**

enter number of vertices:

4

enter the adjacency matrix:

0 20 10 0

20 0 0 30

10 0 0 0

0 30 0 0

enter the src node:0 distance of node1=20

path=1<-0

distance of node2=10

path=2<-0

distance of node3=50

path=3<-1<-0

**\*/**

**/\*OUTPUT**

enter number of vertices:

4

enter the adjacency matrix:

0 20 10 0

20 0 0 30

10 0 0 0

0 30 0 0

enter the src node:0 distance of node1=20

path=1<-0

distance of node2=10

path=2<-0

distance of node3=50

path=3<-1<-0

**\*/**

**/\*OUTPUT**

enter number of vertices:

4

enter the adjacency matrix:

0 20 10 0

20 0 0 30

10 0 0 0

0 30 0 0

enter the src node:0 distance of node1=20

path=1<-0

distance of node2=10

path=2<-0

distance of node3=50

path=3<-1<-0

**\*/**

**/\*OUTPUT**

enter number of vertices:

4

enter the adjacency matrix:

0 20 10 0

20 0 0 30

10 0 0 0

0 30 0 0

enter the src node:0 distance of node1=20

path=1<-0

distance of node2=10

path=2<-0

distance of node3=50

path=3<-1<-0

**\*/**