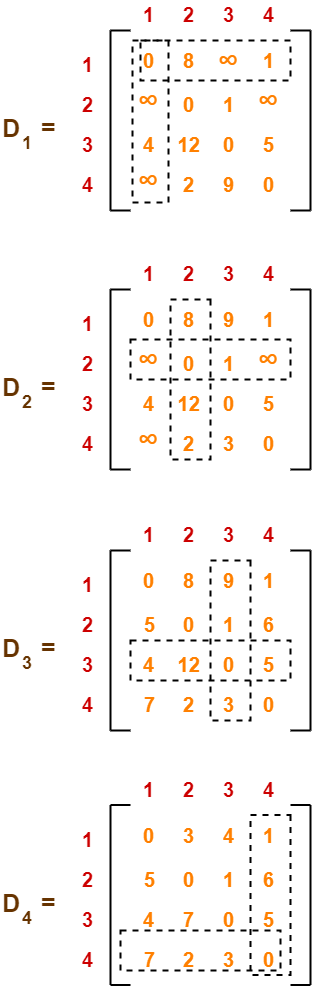
D4 is final answer



#include<stdio.h>

#include<conio.h>

int min(int,int);

void floyds(int p[10][10],int n) {

int i,j,k;

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

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

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

if(i==j)

p[i][j]=0; else

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

}

int min(int a,int b) {

if(a<b)

return(a); else

return(b);

}

void main() {

int p[10][10],w,n,e,u,v,i,j;

;

// clrscr();

printf("\n Enter the number of vertices:");

scanf("%d",&n);

printf("\n Enter the number of edges:\n");

scanf("%d",&e);

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

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

p[i][j]=999;

}

for (i=1;i<=e;i++) {

printf("\n Enter the end vertices of edge%d with its weight \n",i);

scanf("%d%d%d",&u,&v,&w);

p[u][v]=w;

}

printf("\n Matrix of input data:\n");

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

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

printf("%d \t",p[i][j]);

printf("\n");

}

floyds(p,n);

printf("\n Transitive closure:\n");

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

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

printf("%d \t",p[i][j]);

printf("\n");

}

printf("\n The shortest paths are:\n");

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

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

if(i!=j)

printf("\n <%d,%d>=%d",i,j,p[i][j]);

}

getch();

}

Enter the number of vertices:4

Enter the number of edges:

6

Enter the end vertices of edge1 with its weight

1 2 8

Enter the end vertices of edge2 with its weight

1 4 1

Enter the end vertices of edge3 with its weight

2 3 1

Enter the end vertices of edge4 with its weight

4 3 9

Enter the end vertices of edge5 with its weight

4 2 2

Enter the end vertices of edge6 with its weight

3 1 4

Matrix of input data:

999 8 999 1

999 999 1 999

4 999 999 999

999 2 9 999

Transitive closure:

0 3 4 1

5 0 1 6

4 7 0 5

7 2 3 0

The shortest paths are:

<1,2>=3

<1,3>=4

<1,4>=1

<2,1>=5

<2,3>=1

<2,4>=6

<3,1>=4

<3,2>=7

<3,4>=5

<4,1>=7

<4,2>=2

<4,3>=3