

```

1  #include<stdio.h>
2  int min(int,int);
3  void floyds(int p[10][10],int n)
4  {
5      int i,j,k;
6      for(k=1;k<=n;k++)
7          for(i=1;i<=n;i++)
8              for(j=1;j<=n;j++)
9                  if(i==j)
10                     p[i][j]=0;
11                 else
12                     p[i][j]=min(p[i][j],p[i][k]+p[k][j]);
13 }
14 int min(int a,int b)
15 {
16     if(a<b)
17         return(a);
18     else
19         return(b);
20 }
21 void main()
22 {
23     int p[10][10],w,n,e,u,v,i,j;;
24     printf("\n Enter the number of vertices:");
25     scanf("%d",&n);
26     printf("\n Enter the number of edges:\n");
27     scanf("%d",&e);
28     for(i=1;i<=n;i++)

```



```

main.c
26 printf("\n Enter the number of edges:\n");
27 scanf("%d",&e);
28 for(i=1;i<=n;i++)
29 {
30     for(j=1;j<=n;j++)
31         p[i][j]=999;
32 }
33 for(i=1;i<=e;i++)
34 {
35     printf("\n Enter the end vertices of edge%d with its weight \n",i);
36     scanf("%d%d%d",&u,&v,&w);
37     p[u][v]=w;
38 }
39 printf("\n Matrix of input data:\n");
40 for(i=1;i<=n;i++)
41 {
42     for(j=1;j<=n;j++)
43         printf("%d \t",p[i][j]);
44     printf("\n");
45 }
46 floyds(p,n);
47 printf("\n Transitive closure:\n");
48 for(i=1;i<=n;i++)
49 {
50     for(j=1;j<=n;j++)
51         printf("%d \t",p[i][j]);
52     printf("\n");
53 }

```



```

35     printf("\n Enter the end vertices of edge%d with its weight \n",
36     scanf("%d%d%d",&u,&v,&w);
37     p[u][v]=w;
38 }
39 printf("\n Matrix of input data:\n");
40 for(i=1;i<=n;i++)
41 {
42     for(j=1;j<=n;j++)
43         printf("%d \t",p[i][j]);
44     printf("\n");
45 }
46 floyds(p,n);
47 printf("\n Transitive closure:\n");
48 for(i=1;i<=n;i++)
49 {
50     for(j=1;j<=n;j++)
51         printf("%d \t",p[i][j]);
52     printf("\n");
53 }
54 printf("\n The shortest paths are:\n");
55 for(i=1;i<=n;i++)
56     for(j=1;j<=n;j++)
57     {
58         if(i!=j)
59             printf("\n <%d,%d>=%d",i,j,p[i][j]);
60     }
61 }

```

are. Enter the number of vertices:4

Enter the number of edges:
5

Enter the end vertices of edge1 with its weight
1 2 10

Enter the end vertices of edge2 with its weight
3 4 20

Enter the end vertices of edge3 with its weight
2 3 15

Enter the end vertices of edge4 with its weight
< 4 5 40

Enter the end vertices of edge5 with its weight
1 3 50

Matrix of input data:

999	10	50	999
999	999	15	999
999	999	999	20
999	999	999	999

Transitive closure:

0	10	25	45
999	0	15	35
999	999	0	20
999	999	999	0

share. 1 3 50 of edges with its weight

Matrix of input data:

999	10	50	999
999	999	15	999
999	999	999	20
999	999	999	999

Transitive closure:

0	10	25	45
999	0	15	35
999	999	0	20
999	999	999	0

The shortest paths are:

<1,2>=10
<1,3>=25
<1,4>=45
<2,1>=999
<2,3>=15
<2,4>=35
<3,1>=999
<3,2>=999
<3,4>=20
<4,1>=999
<4,2>=999
<4,3>=999