**Q2) Floyds algorithm to find shortest path among nodes.**

#include<stdio.h>

int min(int a,int b) {

    if(a<b) return(a);

    else return(b);

}

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++)

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

        }

    }

}

int main() {

    int p[10][10],w,n,e,u,v,i,j;

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

    scanf("%d %d",&n,&e);

    for (i=1;i<=n;i++) {

        for (j=1;j<=n;j++)

        {

            if(i==j)

                p[i][j]=0;

            else

                p[i][j]=999;

        }

    }

    for (i=1;i<=e;i++) {

        printf("\nEnter 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++)

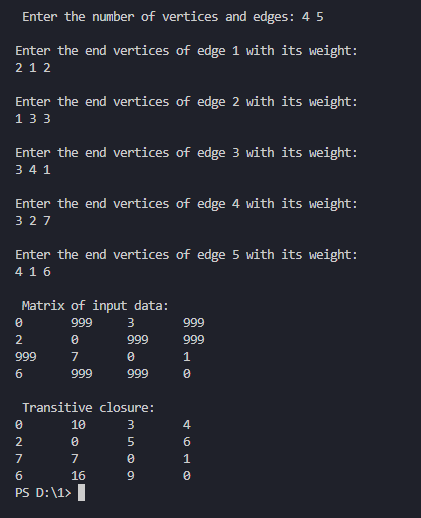
            printf("%d \t",p[i][j]);

        printf("\n");

    }

    return 0;

}

****