**Floyd’s Algorithm:**

#include<stdio.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;

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++)

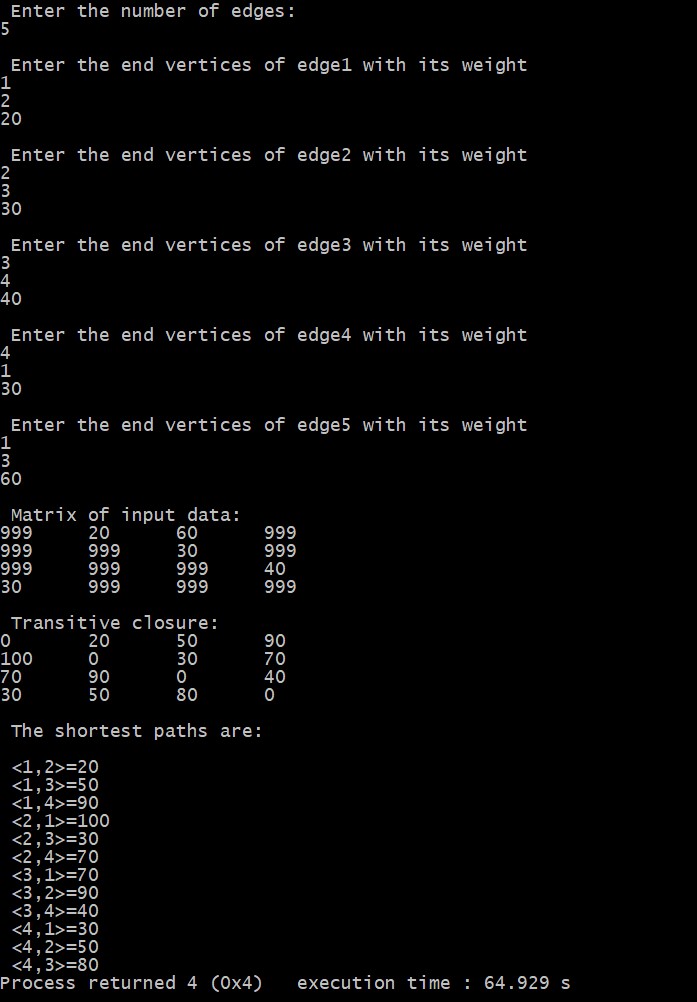
for (j=1;j<=n;j++) {

if(i!=j)

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

}

}



**Knapsack:**

#include<stdio.h>

#include<conio.h>

int w[10],p[10],v[10][10],n,i,j,cap,x[10]= {0};

int max(int i,int j) {

return ((i>j)?i:j);

}

int knap(int i,int j) {

int value;

if(v[i][j]<0) {

if(j<w[i])

value=knap(i-1,j); else

value=max(knap(i-1,j),p[i]+knap(i-1,j-w[i]));

v[i][j]=value;

}

return(v[i][j]);

}

void main() {

int profit,count=0;

printf("\nEnter the number of elements\n");

scanf("%d",&n);

printf("Enter the profit and weights of the elements\n");

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

printf("For item no %d\n",i);

scanf("%d%d",&p[i],&w[i]);

}

printf("\nEnter the capacity \n");

scanf("%d",&cap);

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

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

if((i==0)||(j==0))

v[i][j]=0; else

v[i][j]=-1;

profit=knap(n,cap);

i=n;

j=cap;

while(j!=0&&i!=0) {

if(v[i][j]!=v[i-1][j]) {

x[i]=1;

j=j-w[i];

i--;

} else

i--;

}

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

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

{

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

}

printf("\n");

}

printf("Items included are\n");

printf("Sl.no\tweight\tprofit\n");

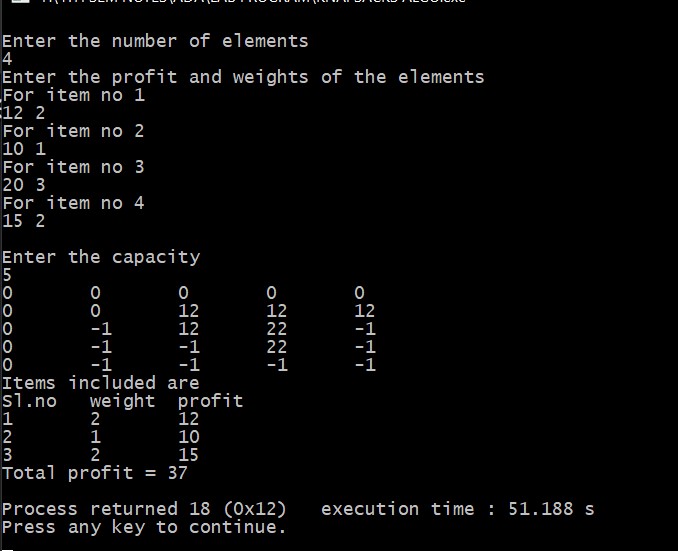
for (i=1;i<=n;i++)

if(x[i])

printf("%d\t%d\t%d\n",++count,w[i],p[i]);

printf("Total profit = %d\n",profit);

}



**Warshel’s Algorithum:**

#include<stdio.h>

#include<conio.h>

int n,a[10][10],p[10][10];

void warshall(int n,int a[10][10],int p[10][10])

{

int i,j,k;

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

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

p[i][j]=a[i][j];

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

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

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

if((p[i][j]==0) && (p[i][k]==1 && p[k][j]==1))

p[i][j]=1;

}

void main()

{

int i,j;

printf("enter the 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]);

}

}

warshall(n,a,p);

printf("trasitive closure\n");

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

{

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

{

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

}

printf("\n");

}

}

