1. Implement All Pair Shortest paths problem using Floyd’s algorithm.

#include <stdio.h>

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

void floyd(int [][10],int);

int min(int,int);

int main()

{

printf("Enter the no. of vertices:");

scanf("%d",&n);

printf("Enter the cost adjacency matrix:\n");

int i,j;

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

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

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

}

}

floyd(a,n);

printf("Distance Matrix:\n");

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

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

printf("%d ",D[i][j]);

}

printf("\n");

}

return 0;

}

void floyd(int a[][10],int n){

int i,j,k;

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

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

D[i][j]=a[i][j];

}

}

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

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

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

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

}

}

}

}

int min(int a,int b){

if(a<b){

return a;

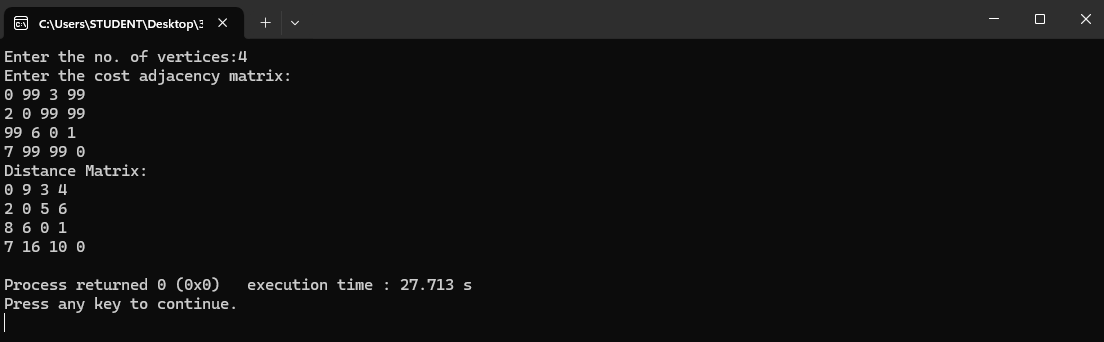
}else{

return b;

}

}

Output:



2. Compute the transitive closure of a given directed graph using Warshalls’ algorithm.

#include<stdio.h>

int n, v;

int a[10][10];

int p[10][10];

void write\_data( )

{

int i,j;

printf("The path matrix is shown below\n");

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

{

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

{

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

printf(" ");

}

printf("\n");

}

}

void read\_data( )

{

int i,j;

printf("Enter the no of vertices: \n");

scanf("%d", &v);

printf("Enter the no of nodes\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]);

}

}

}

void path\_matrix( )

{

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][k]==1 && p[k][j]==1))

p[i][j]=1;

}

}

}

}

void main( )

{

read\_data();

path\_matrix();

write\_data();

}

Output:

