S.No: 13	Exp. Name: ***Program to find All-Pairs Shortest Paths problem using Floyd's algorithm.***	Date:

Aim:

Program to Implement All-Pairs Shortest Paths problem using Floyd's algorithm

Source Code:

Floyds.c

```
#include<stdio.h>
#include<conio.h>
#include<limits.h>
int p[20][20];
int d[20][20];
int w[20][20];
void print_path(int i,int j)
{
   if(i==j)
   printf("%d",i);
   else
   {
      if(p[i][j]==-1)
      printf("No path Exists");
      else
      {
         print_path(i,p[i][j]);
         printf("-> %d",j);
      }
   }
}
void warshall(int n)
{
   for(int i=1;i<=n;i++)
   {
      for(int j=1;j<=n;j++)
      {
         d[i][j]=w[i][j];
      }
   }
   for(int k=1;k<=n;k++)
       {
         for(int i=1;i<=n;i++)
         {
            for(int j=1;j<=n;j++)
            {
               if(d[i][k]==INT_MAX || d[k][j]==INT_MAX)
               continue;
               if(d[i][k]+d[k][j]<d[i][j])
               {
                  d[i][j]=d[i][k]+d[k][j];
                  p[i][j]=p[k][j];
               }
            }
         }
       }
}
void main()
{
   int i,j,v,s,des;
   char ch;
   printf("Enter number of vertices: ");
```

```
   scanf("%d",&v);
   printf("Enter the weight matrix");
   for(i=1;i<=v;i++)
   {
      for(j=1;j<=v;j++)
      {
         if(i==j)
         {
            w[i][j]=0;
            p[i][j]=-1;
            continue;
         }
         printf("Is edge (%d,%d) present in graph (y/n): ",i,j);
         fflush(stdin);
         scanf("%c",&ch);
         if(ch == 'y' || ch == 'Y')
         {
            printf("Enter weight of edge (%d,%d): ",i,j);
            scanf("%d",&w[i][j]);
            p[i][j]=i;
         }
         else
         {
            w[i][j]=INT_MAX;
            p[i][j]=-1;
         }
      }
   }
   warshall(v);
   printf("Enter source and destination: ");
   scanf("%d %d",&s,&des);
   printf("Distance = %d",d[s][des]);
   print_path(s,des);
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter number of vertices: 3
Enter the weight matrixIs edge (1,2) present in graph (y/n): y
Enter weight of edge (1,2): 10
Is edge (1,3) present in graph (y/n): 5
Is edge (2,1) present in graph (y/n): n
Is edge (2,3) present in graph (y/n): y
Enter weight of edge (2,3): 10
Is edge (3,1) present in graph (y/n): y
Enter weight of edge (3,1): 5
Is edge (3,2) present in graph (y/n): y
Enter weight of edge (3,2): 15
Enter source and destination: 1 3
Distance = 201-> 2-> 3

Test Case - 2
User Output
Enter number of vertices: 2
Enter the weight matrixIs edge (1,2) present in graph (y/n): y

Test Case - 2
Enter weight of edge (1,2): 5
Is edge (2,1) present in graph (y/n): y
Enter weight of edge (2,1): 20
Enter source and destination: 1 2
Distance = 51-> 2