Name: Rishanth.K
USN: 1BM19CS129
Subject: ADA LAB2

Date: 8-7-21
Signature:

4) Implement all pair shortest paths problem using floyd's algorithm.

```
#include<stdio.h>
#include<stdlib.h>
#define infinity 9999
#define MAX 100

int n;
int adj[MAX][MAX];
int D[MAX][MAX];
int Pred[MAX][MAX];

void create_graph();
void FloydWarshalls();
void display(int matrix[MAX][MAX], int n);

int main()
{
   int s, d;
   create_graph();
   FloydWarshalls();
}
```

```
void FlayWarshalls()
{
   int i,j,k;
   for(i=0; i<n; i++)
   for(j=0; j<n; j++)
 {
    if(adj[i][j]==0)
    {
      D[i][j]=infinity
      Pred[i][j]= -1;
    }
    else
    {
      D[i][j]=adj[i][j];
      Pred[i][j]=i;
    }
 }
 for(k=0; k<n; k++)
 {
    for(i=0; i<n; i++)
    for(j=0; j<n; j++)
     if(D[i][k]+D[k][j]< D[i][j])
```

```
{
    D[i][j] = D[i][k] + D[k][j];
    Pred[i][j] = Pred[k][j];
  }
}
printf("\nshortest path matrix is:\n");
display(D, n);
for(i=0; i<n; i++)
 if(D[i][i]<0)
{{
   printf("\n Error : negative cycle\n");
   exit(1);
  }
}
void findPath(int s, int d)
{
  int i, path[MAX], count;
  if(D[s][d]==infinity)
 {
   printf("\nNo path\n");
    return;
 }
 count = -1;
```

```
do
{
  path[++count] = d;
  d = Pred[s][d];
}while(d!=s);
  path[++count] = s;
  for(i=count; i>=0; i--)
  {
    printf("%d", path[i]);
    printf("\n");
  }
void display(int matrix[MAX][MAX], int n)
{
  int i, j;
  for(i=0; i<n; i++)
  {
    for(j=0; j<n; j++)
    {
      printf("%7d", matrix[i][j]);
    }
    printf("\n");
  }
}
void create_graph()
{
```

```
int i, max_edges, origin, destin, wt;
printf("\nEnter number of vertices: ");
scanf("%d", &n);
printf("Enter the adjacency matrix:\n");
for(origin = 0; origin<n; origin++)
for(destin = 0; destin<n; destin++)
scanf("%d", &adj[origin][destin]);
}
```

Modification:

```
int main()
{
int s, d;
create_graph();
Floyd_Warshalls();
while(1)
{
printf("\nEnter source vertex(-1 to exit): ");
scanf("%d", &s);
if(s == -1)
break;
printf("\nEnter destination vertex: ");
scanf("%d", &d);
if(s<0 || s>n-1 || d<0 || d>n-1)
```

```
{
printf("[redacted]\n Enter valid vertex\n");
continue;
}
{
  printf("\n Shortest path is ");
    findpath(s,d);
    printf("\n length of this path is %d\n",DE[d]);
  }
}
```