1)

#include<stdio.h>

#define MAX 10

#define INFINITY 9999

#define NEXTLINE printf("\n")

#define CN count++

int GRAPH[MAX][MAX];

int V,E,count=0;

struct EDGE

{

int src,dest,w;

}edge[MAX];

void G\_create()

{

int i,j,c=0;

printf("ENTER NUMBER OF VERTICES:\n");

scanf("%d",&V);

printf("ENTER NUMBER OF EDGES:\n");

scanf("%d",&E);

printf("\n\nENTER EDGES(X,Y) AND ITS WEIGHT (W) FOR YOUR GRAPH (FORMAT X,Y,W): \n");

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

scanf("%d,%d,%d",&edge[i].src,&edge[i].dest,&edge[i].w);

}

void printCost(int dist[],int src)

{

int i;

printf("\n\SHORTEST PATHS: \n");

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

{

if(dist[i]==INFINITY)

printf("%d: oo\n",i);

else

printf("COST(%d,%d) = %d\n",src,i,dist[i]);

}

}

void path(int parent[],int i)

{

if(parent[i]==-1)

return;

path(parent,parent[i]);

printf("--> |%d| ",i);

}

void printPath(int parent[],int src)

{

int i;

printf("\n\nSHORTEST PATHS: \n");

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

{

printf("Path(%d,%d): |%d| ",src,i,src);

path(parent,i);

printf("\n");

}

}

void BellmanFord()

{

int dist[MAX],parent[MAX];

int i,j,src;

printf("ENTER THE SOURCE VERTEX: ");

scanf("%d",&src);

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

{CN;dist[i]=INFINITY;parent[i-1]=-1;CN;CN;}

dist[src]=0;CN;

for(i=1;i<=V-1;i++)

{

CN;

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

{

CN;

int u=edge[j].src,v=edge[j].dest;CN;

int weight=edge[j].w;CN;

if(dist[u]+weight < dist[v] && dist[u]!=INFINITY)

{CN;dist[v]=dist[u]+weight;CN;}

parent[v]=u;CN;

}

}

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

{

CN;

int u=edge[j].src,v=edge[j].dest;CN;CN;

int weight=edge[j].w;CN;

if(dist[u]+weight < dist[v])

{

CN;

printf("NEGATIVE CYCLE EXITS\n");

return;

}

}

printCost(dist,src);CN;

printPath(parent,src);CN;

}

int main()

{

G\_create();

BellmanFord();

printf("\n\nSTEP COUNT: %d\n",count);

}

OUTPUT

