**Name:** Shivani Suresh

**Register Number:** 2021503050

**Batch:** 2

**CS 6110- Computer Networks- Lab 12-9th November 2023**

**Distance Vector Routing Algorithm**

**Code:**

import java.io.\*;

import java.util.Scanner;

public class DVR

{

static int graph[][];

static int via[][];

static int rt[][];

static int v;

static int e;

public static void main(String args[]) throws IOException

{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

System.out.println("Please enter the number of Vertices: ");

v = Integer.parseInt(br.readLine());

System.out.println("Please enter the number of Edges: ");

e = Integer.parseInt(br.readLine());

graph = new int[v][v];

via = new int[v][v];

rt = new int[v][v];

for(int i = 0; i < v; i++)

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

{

if(i == j)

graph[i][j] = 0;

else

graph[i][j] = 9999;

}

for(int i = 0; i < e; i++)

{

System.out.println("Please enter data for Edge " + (i + 1) + ":");

System.out.print("Source: ");

int s = Integer.parseInt(br.readLine());

s--;

System.out.print("Destination: ");

int d = Integer.parseInt(br.readLine());

d--;

System.out.print("Cost: ");

int c = Integer.parseInt(br.readLine());

graph[s][d] = c;

graph[d][s] = c;

}

dvr\_calc\_disp("The Routing Tables are: ");

Scanner in=new Scanner(System.in);

System.out.print("Enter source: ");

int so = in.nextInt();

System.out.print("Enter Destination: ");

int de = in.nextInt();

de--; so--;

int j = so;

System.out.print("Path: ");

System.out.print(so+1);

while(j!=de){

System.out.print("->"+(via[j][de]+1));

j = via[j][de];

}

System.out.println();

}

static void dvr\_calc\_disp(String message)

{

System.out.println();

init\_tables();

update\_tables();

System.out.println(message);

print\_tables();

System.out.println();

}

static void update\_table(int source)

{

for(int i = 0; i < v; i++)

{

if(graph[source][i] != 9999)

{

int dist = graph[source][i];

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

{

int inter\_dist = rt[i][j];

if(via[i][j] == source)

inter\_dist = 9999;

if(dist + inter\_dist < rt[source][j])

{

rt[source][j] = dist + inter\_dist;

via[source][j] = i;

}

}

}

}

}

static void update\_tables()

{

int k = 0;

for(int i = 0; i < 4\*v; i++)

{

update\_table(k);

k++;

if(k == v)

k = 0;

}

}

static void init\_tables()

{

for(int i = 0; i < v; i++)

{

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

{

if(i == j)

{

rt[i][j] = 0;

via[i][j] = i;

}

else

{

rt[i][j] = 9999;

via[i][j] = 100;

}

}

}

}

static void print\_tables()

{

for(int i = 0; i < v; i++)

{

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

{

System.out.print("Dist: " + rt[i][j]+" ");

}

System.out.println();

}

}

}

**Output:**

