NAME: S.Tharish reddy

REG.NO.: 192211485

CODE: CSA0734

EXPERIMENT: 27

AIM: To implement distance vector routing using JAVA programming.

PROGRAM:

```
import java.io.*; 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 initial Routing Tables are: ");
System.out.print("Please enter the Source Node for the edge whose cost has changed: "); int s =
Integer.parseInt(br.readLine());
s--;
System.out.print("Please enter the Destination Node for the edge whose cost has changed: ");
int d = Integer.parseInt(br.readLine()); d--;
        System.out.print("Please enter the new cost: "); int c = Integer.parseInt(br.readLine());
        graph[s][d] = c;
```

```
graph[d][s] = c;
dvr_calc_disp("The new Routing Tables are: ");
}
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:

```
  Please enter the number of Vertices: 4 Please enter the number of Edges: 5 Please enter data for
  Edge 1:
  Source: 1
  Destination: 2
  Cost: 1
  Please enter data for Edge 2: Source: 1
  Destination: 3
  Cost: 3
  Please enter data for Edge 3: Source: 2
  Destination: 3
  Cost: 1
  Please enter data for Edge 4: Source: 2
  Destination: 4
  Cost: 1
  Please enter data for Edge 5: Source: 3
  Destination: 4
  Cost: 4
The      Dist:   Dist:   Dist:
initial  1       2       2
Routi
ng
Tables
are:
Dist:
0
Dist:    Dist:   Dist:   Dist:
1        0       1       1
Dist:    Dist:   Dist:   Dist:
2        1       0       2
Dist:    Dist:   Dist:   Dist:
2        1       2       0
  Please enter the Source Node for the edge whose cost has changed: 2 Please enter the Destination
  Node for the edge whose cost has changed: 4 Please enter the new cost: 10
  The new Routing Tables are: Dist: 0 Dist: 1 Dist: 2 Dist: 6
  Dist: 1 Dist: 0 Dist: 1 Dist: 5
  Dist: 2 Dist: 1 Dist: 0 Dist: 4
```

```
Dist: 6 Dist: 5 Dist: 4 Dist: 0
```

RESULT: Therefore distance vector routing has been successfully excecuted using JAVA programming