Dijskshtra algo:-

import java.util.\*;

import java.lang.\*;

import java.io.\*;

public class Main

{

static final int V=9;

int mindist(int dis[], Boolean spt[])

{

int min=Integer.MAX\_VALUE,min1=-1;

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

if(spt[v]==false && dis[v]<=min)

{

min=dis[v];

min1=v;

}

return min1;

}

void prints(int dis[])

{

System.out.println(" vertex distance from source");

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

System.out.println(i + "\t\t\t" + dis[i]);

}

void dij(int graph[][],int src)

{

int dis[]=new int[V];

Boolean spt[]=new Boolean[V];

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

{

dis[i]=Integer.MAX\_VALUE;

spt[i]=false;

}

dis[src]=0;

for(int k=0;k<V-1;k++)

{

int u =mindist(dis,spt);

spt[u]=true;

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

if(!spt[v] && graph[u][v]!=0 && dis[u]!=Integer.MAX\_VALUE && dis[u] +graph[u][v]<=dis[v])

dis[v]=dis[u]+graph[u][v];

}

prints(dis);

}

public static void main(String[] args)

{

int graph[][] = new int[][] { { 0, 4, 0, 0, 0, 0, 0, 8, 0 },

{ 4, 0, 8, 0, 0, 0, 0, 11, 0 },

{ 0, 8, 0, 7, 0, 4, 0, 0, 2 },

{ 0, 0, 7, 0, 9, 14, 0, 0, 0 },

{ 0, 0, 0, 9, 0, 10, 0, 0, 0 },

{ 0, 0, 4, 14, 10, 0, 2, 0, 0 },

{ 0, 0, 0, 0, 0, 2, 0, 1, 6 },

{ 8, 11, 0, 0, 0, 0, 1, 0, 7 },

{ 0, 0, 2, 0, 0, 0, 6, 7, 0 } };

Main t = new Main();

t.dij(graph, 0);

}

}

Bellman ford:-

import java.util.\*;

class Main

{

dis[graph[j][0]] + graph[j][2];

}

}

static void Bellman(int graph[][], int V, int E,

int src)

{

int []dis = new int[V];

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

dis[i] = Integer.MAX\_VALUE;

dis[src] = 0;

for (int i = 0; i < V - 1; i++)

{

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

{

if (dis[graph[j][0]] + graph[j][2] <

dis[graph[j][1]])

dis[graph[j][1]] = dis[graph[j][0]+graph[j][2];

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

{

int x = graph[i][0];

int y = graph[i][1];

int weight = graph[i][2];

if (dis[x] != Integer.MAX\_VALUE &&

dis[x] + weight < dis[y])

System.out.println("Graph contains negative"

+" weight cycle");

}

System.out.println("Vertex Distance from Source");

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

System.out.println(i + "\t\t" + dis[i]);

}

public static void main(String[] args)

{

int V = 5;

int E = 8;

int graph[][] = { { 0, 1, -1 }, { 0, 2, 4 },

{ 1, 2, 3 }, { 1, 3, 2 },

{ 1, 4, 2 }, { 3, 2, 5 },

{ 3, 1, 1 }, { 4, 3, -3 } };

Bellman(graph, V, E, 0);

}

}