|  |
| --- |
| **8.Write a program to find the shortest path between vertices using bellman-ford**  **algorithm.** |

importjava.util.\*;

public class BellmanDemoFinal

{

static Scanner in = new Scanner(System.in);

public static void main(String [] args)

{

int V,E=1,chckNegative=0;

int w[][] = new int [20][20];

int edge[][] = new int [50][2];

**/\* Read the no of vertices in the graph \*/**

System.out.println("Enter the no of vertices ");

V = in.nextInt();

System.out.println("Enter the weight matrix");

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

for(int j=1;j<=V;j++)

{

w[i][j] = in.nextInt();

if(w[i][j]!=0)

{ edge[E][0]=i;

edge[E++][1]=j;

}

}

chckNegative = bellmanFord(w,V,E,edge);

if(chckNegative == 1)

System.out.println("\nNo negative weight cycle\n");

else

System.out.println("\nNegative weight cycle exists\n");

}

public static intbellmanFord(int w[][],intV,intE,int edge[][])

{

intu,v,S,flag=1;

int distance [] = new int[20];

int parent [] = new int [20];

**/\* Assign the distance of all the vertices to 999 \*/**

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

{

distance[i] = 999;

parent[i]=-1;

}

System.out.println("Enter the source vertex");

S = in.nextInt();

**/\* Assign the distance of source vertex to 999 \*/**

distance[S]=0;

**/\* Relax each edge for V-1 times \*/**

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

{

for(int k=1;k<=E;k++)

{

u = edge[k][0];

v = edge[k][1] ;

**/\* Relaxing each edge \*/**

if(distance[u]+w[u][v] < distance[v])

{

distance[v] = distance[u] + w[u][v] ;

parent[v]=u ;

}

}

}

**/\* Relax all the edges one more time to check for negative weight cycle \*/**

for(int k=1;k<=E;k++)

{

u = edge[k][0] ;

v = edge[k][1] ;

if(distance[u]+w[u][v] < distance[v])

flag = 0 ;

}

if(flag==1)

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

System.out.println("Vertex " + i + " -> cost = " + distance[i] + " parent = "+ (parent[i]));

return flag;

}

}

**Output:**

**Run 1:**

**Enter the no of vertices**

**6**

**Enter the weight matrix**

**0 3 2 5 999 999**

**3 0 999 1 4 999**

**2 999 0 2 999 1**

**5 1 2 0 3 999**

**999 4 999 3 0 2**

**999 999 1 999 2 999**

**Enter the source vertex**

**1**

**Vertex 1 -> cost = 0 parent = -1**

**Vertex 2 -> cost = 3 parent = 1**

**Vertex 3 -> cost = 2 parent = 1**

**Vertex 4 -> cost = 4 parent = 2**

**Vertex 5 -> cost = 5 parent = 6**

**Vertex 6 -> cost = 3 parent = 3**

**No negative weight cycle**

**Run2:**

**Enter the no of vertices**

**5**

**Enter the weight matrix**

**0 -1 4 999 999**

**999 0 3 2 2**

**999 999 0 999 999**

**999 1 5 0 999**

**999 999 999 -3 0**

**Enter the source vertex**

**1**

**Vertex 1 -> cost = 0 parent = -1**

**Vertex 2 -> cost = -1 parent = 1**

**Vertex 3 -> cost = 2 parent = 2**

**Vertex 4 -> cost = -2 parent = 5**

**Vertex 5 -> cost = 1 parent = 2**

**No negative weight cycle**

**Run 3:**

**Enter the no of vertices**

**5**

**Enter the weight matrix**

**0 -1 4 999 999**

**999 0 3 2 2**

**999 999 0 999 999**

**999 1 5 0 999**

**999 999 999 -5 0**

**Enter the source vertex**

**1**

**Negative weight cycle exists**