**SHORTEST PATH BETWEEN VERTICES USING BELLMAN-FORD ALGORITHM**

**import** java.util.Scanner;

**public** **class** bell {

**private** **int** distance[];

**private** **int** noofvertices;

**public** **static** **final** **int** *max\_val*=999;

**public** bell(**int** no)

{

noofvertices=no;

distance= **new** **int**[noofvertices+1];

}

**public** **void** belleval(**int** source,**int** destination,**int** adjmat[][])

{

**for**(**int** node=1;node<=noofvertices;node++)

{

distance[node]=*max\_val*;

}

distance[source]=0;

**for**(**int** node=1;node<=noofvertices-1;node++)

{

**for**(**int** sourcenode=1;sourcenode<=noofvertices;sourcenode++)

{

**for**(**int** destinationnode=1;destinationnode<=noofvertices;destinationnode++)

{

**if**(adjmat[sourcenode][destinationnode]!=*max\_val*)

{

**if**(distance[destinationnode]>distance[sourcenode]+adjmat[sourcenode][destinationnode])

distance[destinationnode]=distance[sourcenode]+adjmat[sourcenode][destinationnode];

}

}

}

}

**for**(**int** sourcenode=1;sourcenode<=noofvertices;sourcenode++)

{

**for**(**int** destinationnode=1;destinationnode<=noofvertices;destinationnode++)

{

**if**(adjmat[sourcenode][destinationnode]!=*max\_val*)

{

**if**(distance[destinationnode]>distance[sourcenode]+adjmat[sourcenode][destinationnode])

System.*out*.println("The Graph contains negative edge cycle");

}

}

}

**for**(**int** vertex=1;vertex<=noofvertices;vertex++)

{

**if**(vertex==destination)

System.*out*.println("distance of source"+source+"to"+vertex+"is"+distance[vertex]);

}

}

**public** **static** **void** main(String args[])

{

**int** noofvertices=0;

**int** source,destination;

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

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

noofvertices=in.nextInt();

**int** adjmat[][]=**new** **int**[noofvertices+1][noofvertices+1];

System.*out*.println("enter the adjacency matrix");

**for**(**int** sourcenode=1;sourcenode<=noofvertices;sourcenode++)

{

**for**(**int** destinationnode =1;destinationnode<=noofvertices;destinationnode++)

{

adjmat[sourcenode][destinationnode]=in.nextInt();

**if**(sourcenode==destinationnode)

{

adjmat[sourcenode][destinationnode]=0;

**continue**;

}

**if**(adjmat[sourcenode][destinationnode]==0)

{

adjmat[sourcenode][destinationnode]=*max\_val*;

}

}

}

System.*out*.println("enter the source vertex");

source=in.nextInt();

System.*out*.println("enter the destination vertex :");

destination=in.nextInt();

bell bella=**new** bell(noofvertices);

bella.belleval(source,destination,adjmat);

in.close();

}

}