import it.uniupo.algoTools.MinHeap;

import it.uniupo.graphLib.Edge;

import it.uniupo.graphLib.UndirectedGraph;

public class Dijkstra

{

private UndirectedGraph graph;

private boolean found[];

private MinHeap<Edge,Integer> heap;

public Dijkstra(UndirectedGraph graph)

{

this.graph=graph;

found=new boolean[graph.getOrder()];

heap=new MinHeap<Edge,Integer>();

}

public int[] getDistances(int source)

{

int distance[]= new int[graph.getOrder()];

distance[source]=0;

found[source]=true;

for(Edge e:graph.getIncidentEdges(source))

heap.add(e,distance[e.getTail()]+e.getWeight());

while(!heap.isEmpty())

{

Edge e=heap.extractMin();

if(!found[e.getHead()])

{

found[e.getHead()]=true;

distance[e.getHead()]=distance[e.getTail()]+e.getWeight();

for(Edge f:graph.getIncidentEdges(e.getHead()))

heap.add(f,distance[f.getTail()]+f.getWeight());

}

}

return distance;

}

}

------------------------------------------------------TEST----------------------------------TEST--------------------

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

import it.uniupo.graphLib.UndirectedGraph;

class DijkstraTest

{

Dijkstra sample;

void init()

{

sample=new Dijkstra(new UndirectedGraph("6;0 1 5;0 3 1;1 4 2;2 4 6;3 4 7;0 5 1;1 2 3"));

}

@Test

void testDijkstra()

{

init();

assertNotNull(sample);

}

@Test

void testGetDistances()

{

init();

int distance[]=sample.getDistances(0);

assertEquals(distance[0],0);

assertEquals(distance[1],5);

assertEquals(distance[2],8);

assertEquals(distance[3],1);

assertEquals(distance[4],7);

assertEquals(distance[5],1);

}

}