```java
package hw4;

import java.util.Iterator;
import java.util.HashMap;
import java.util.PriorityQueue;
import java.util.Map.Entry;
import java.util.ArrayList;
import java.util.Arrays;

/*
 * graphDLM represents a directed labeled multi-graph of strings.
 * GraphDLMs are mutable.
 * A GraphDLM consists of a set of nodes and a set of edges,
 * where each node is a unique String and each edge connects two
 * nodes that are present in the graph. An edge can start and end
 * at the same node. Additionally, each edge is labeled with a
 * String.
 */

public class GraphDLM {

    private HashMap<String, HashMap<String, PriorityQueue<String> > > data;

    /*
    * Representation Invariant:
    *
    * data != null &&
    *
    * For all keys k in data.keySet():
    *
    *       * k != null
    *
    *       * data[k] != null
    *
    *       * data[k].keySet() is a subset of data.KeySet()
    *
    *       * For all keys k2 in data[k].keySet():
    *
    *               * data[k][k2] != null
    *
    *               * data[k][k2] is not empty
    */
```

```java
/*
* Abstraction Function:
*
* Each key u in data.keySet() represents a node in the graph.
*
* Each key v in data[u].keySet() represents a set of edges from node u to node v in the
  graph.
*
* The elements in data[u][v] represent the labels associated with the set of edges
* going from parent u to child v in descending weight order.
*/

/**
 * @requires none
 * @param none
 * @modifies this.data
 * @effects initializes this.data as an empty
 *          HashMap<String,HashMap<String,PriorityQueue<String> > >
 * @returns a new GraphDLM object with data initialized as an empty HashMap.
 * @throws none
 */
public GraphDLM() {
  throw new RuntimeException("GraphDLM is not yet implemented");
}

/**
 * @requires none
 * @param nodes an ArrayList of nodes to be added to the graph
 * @modifies this.data
 * @effects initializes this.data as a
 *          HashMap<String,HashMap<String,PriorityQueue<String>>>
 *          with the distinct elements of nodes as keys and empty initialized values.
 * @returns a new GraphDLM object with data initialized as an empty HashMap.
 * @throws IllegalArgumentException if nodes == null or any element of nodes == null
 */
public GraphDLM(ArrayList<String> nodes) {
    throw new RuntimeException("GraphDLM is not yet implemented");

}

/**
 * @requires none
 * @param node the node to be added
 * @modifies self.data
 * @effects if data[node] does not exist, adds it as a new empty HashMap
 * @returns true if node is not present in data already, false otherwise
 * @throws IllegalArgumentException if (node == null)
 */
public boolean addNode(String node) {
    throw new RuntimeException("GraphDLM is not yet implemented"); }
```

```java
    /**
     * @requires none
     * @param node the node to be removed
     * @modifies self.data
     * @effects if data[node] exists, removes data[node] and for all other nodes i, removes
     *          data[i][node]
     * @returns none
     * @throws IllegalArgumentException if (node == null)
     * @throws IllegalArgumentException if node is not in graph
     */
    public void removeNode(String node) {
        throw new RuntimeException("GraphDLM is not yet implemented");
    }

    /**
     * @requires none
     * @param node the node to be searched for in the graph
     * @modifies none
     * @effects none
     * @returns true if data[node] exists, else false
     * @throws IllegalArgumentException if (node==null)
     */
    public boolean containsNode(String node) {
        throw new RuntimeException("GraphDLM is not yet implemented");
    }

    /**
     * @requires none
     * @param source the source of the edge to be added
     * @param dest the destination of the edge to be added
     * @param label the label of the edge to be added
     * @modifies self.data
     * @effects If data[source][dest] exists, adds value label to its PriorityQueue.
     *          Else, creates new queue with value label at data[source][dest]
     * @returns none
     * @throws IllegalArgumentException if (source == null) || (dest == null) ||
                                           (label == null)
     * @throws IllegalArgumentException if source or dest are not present in the graph
     */
    public void addEdge(String source, String dest, String label) {
        throw new RuntimeException("GraphDLM is not yet implemented");
    }
```

```java
    /**
     * @requires none
     * @param source the source of the edge to be removed
     * @param dest the destination of the edge to be removed
     * @param label the label of the edge to be removed
     * @modifies self.data
     * @effects if data[source][dest] exists and label is present in data[source][dest],
     *          removes one instance of label from data[source][dest]
     * @returns none
     * @throws IllegalArgumentException if (source == null) || (dest == null) ||
                                        (label == null)
     * @throws IllegalArgumentException if data[source][dest] does not exist or label is not
                                        present in data[source][dest]
     */
    public void removeEdge(String source, String dest, String label) {
        throw new RuntimeException("GraphDLM is not yet implemented");
    }

    /**
     * @requires none
     * @param source the source of the edge to be found
     * @param dest the destination of the edge to be found
     * @param label the label of the edge to be found
     * @modifies none
     * @effects none
     * @returns true if data[source][dest] exists and label is present in data[source][dest],
                false otherwise
     * @throws IllegalArgumentException if (source == null) || (dest == null) ||
                                        (label == null)
     */
    public boolean containsEdge(String source, String dest, String label) {
        throw new RuntimeException("GraphDLM is not yet implemented");
    }

    /**
     * @requires none
     * @param none
     * @modifies none
     * @effects none
     * @returns An iterator to the keySet() of self.data
     * @throws none
     */
    public Iterator<String> iterator() {
        throw new RuntimeException("GraphDLM is not yet implemented");
    }
```

```java
    /**
     * @requires none
     * @param none
     * @modifies none
     * @effects none
     * @returns An array list containing all elements in the keySet() of data.
     * @throws none
     */
    public ArrayList<String> getNodes() {
        throw new RuntimeException("GraphDLM is not yet implemented");
    }

        /**
     * @requires none
     * @param parent The node whose children will be returned
     * @modifies none
     * @effects none
     * @returns a copy of data[parent] if parent is present in data
     * @throws IllegalArgumentException if (parent == null)
     * @throws IllegalArgumentException if data[parent] does not exist
     */
    public HashMap<String, PriorityQueue<String> > getChildren(String parent) {
        throw new RuntimeException("GraphDLM is not yet implemented");
}
```