Programming Assignment 6
Introduction to Queues

1 Objectives

1. Implement queue using linked representation.
2. Implement queue using array representation.

2 Queue Interface

Organize your code under package with name

`eg.edu.alexu.csd.datastructure.queue.cs<xx>_cs<yy>_cs<zz>`
`where xx, yy and zz are your and your partners two digits class number.`

and you need to implement the following interface

```
package eg.edu.alexu.csd.datastructure.queue;

public interface IQueue {
    /**
    * Inserts an item at the queue front.
    */
    public void enqueue(Object item);

    /**
    * Removes the object at the queue rear and returns it.
    */
    public Object dequeue();

    /**
    * Tests if this queue is empty.
    */
    public boolean isEmpty();

    /**
    * Returns the number of elements in the queue
    */
    public int size();
}
```

Your class should inherit from this interface **twice**; once to implement a queue using linked-based representation, and once using an array based implementation and supply all it's method with the exact signature.
In the **array based** implementation, your queue won't have more than n elements where n is a parameter in your **class constructor** and a user defined input in your testing class.
You should provide any JUnit tests for testing **both** the implementation of the Queue.

In order to distinguish between the two queue implementations: array-based and linked-based, you need to implements the following two **empty** interfaces, respectively.

```
package eg.edu.alexu.csd.datastructure.queue;
public interface ILinkedBased { }
```

```
package eg.edu.alexu.csd.datastructure.queue;
public interface IArrayBased { }
```

This technique is a common software design technique named *Marker Design Pattern*.

3 Deliverables

- This assignment will be delivered along side the next assignment.
- You should work in teams up to 3 members.
- You should use your own data structures which were implemented in the previous assignments. Don't use any built-in data structure.
- Take into consideration that your implementation will be used later in the project, so it has to be fully functional, well documented and reusable. Try very hard to clean up your implementation. Remove all unused variables. Do not write redundant and repeated code.
- Try out your code using http://onlinetester.tk/
- Late submission is accepted for only one week.
- Delivering a copy will be severely penalized for both parties, so delivering nothing is so much better than delivering a copy.

Good Luck :)