CSC 212 Homework # 3
Queues
Due date: 14/11/2017

Guidelines:	This is an individual assignment. The homework must be **submitted electronically through LMS**. **Submissions by email and hard copy submissions are not accepted**.

Problem 1

1. Write the method **public static** `<T> T serveLast(Queue<T> q)`, user of the ADT Queue, to remove and return the last element in q. Do not use any auxiliary data structures.

   Example 1.1. *If q is $A \to B \to A \to D \to A \to F$, then after calling* `serveLast(q)`*, then q becomes $A \to B \to A \to D \to A$. The method returns F.*

2. Write the method **public static** `<T> T retrieveLast(Queue<T> q)`, user of the ADT Queue, which returns the last element of q without removing it. Do not use any auxiliary data structures.

   Example 1.2. *If q is $A \to B \to A \to D \to A \to F$, then after calling* `retrieveLast(q)`*, then q becomes $A \to B \to A \to D \to A \to F$. The method returns F.*

3. Write the method **public static** `<T> Queue<T> merge (Queue<T> q1, Queue<T> q2)` that takes as input two non-empty queues (q1 and q2) and returns a new queue that contains the elements of both q1 and q2 taken in alternate order as shown in the example below. The queues q1 and q2 should remain unchanged after calling this method. You are not allowed to use any auxiliary data structures.

   Example 1.3. *If $q1 : A, B, C, D$ and $q2 : E, F$, then the call* `merge(q1, q2)` *should return A, E, B, F, C, D.*

4. Write the method **public static** `<T>` **boolean** `isPalindrome (Queue<T> q)` (user of ADT) that takes as input a queue and returns true if the queue is palindrome false otherwise. The queue should remain unchanged after calling this method. The space complexity of the method must be $O(1)$.

Problem 2

1. Write the method **public** `T serveLast()`, member of the class `LinkedQueue` which removes and returns the last element of the queue. Assume that the queue is not empty.

2. Write the method **public void** `remove(int i)`, member of `ArrayQueue` which removes the element at position `i` from the queue. The numbering starts at 0. Assume that `i` is a valid position.

3. Write the method **private void** `insert(Queue<T> q, int i)`, member of the class `ArrayQueue`, which inserts all elements of `q` after the element at position `i`. Numbering starts at 0. Assume that `i` is a valid position and that there is enough space to insert all elements of `q`. The input queue `q` must not change.

Problem 3

1. Write a linked implementation of the ADT PQueue where the elements are kept in their order of insertion. What is the performance of the methods `enqueue` and `serve` in this case? Compare this implementation to the one seen in lecture. Which implementation is better (justify your answer)?

2. Write an array implementation of the ADT PQueue. The *serve* method must run in $O(1)$, *enqueue* in $O(n)$.

Problem 4

A store announces a sale campaign whereby any customer who buys two items gets 50% off on the cheaper one. If the customer buys more than two items, he/she must group them into pairs of two to indicate the items that the offer should apply to.

1. Suppose you want to buy n items in total. Write a method that will give you the best pairing of the items (the one with the minimum price). The method's signature is: **public static** `LinkedList<ItemPair> minPairing(LinkedList <Item> items)`.

2. If you leave it up to the store owner, he/she will try to pair the items in order to obtain the maximum price. Write a method that will help the store owner achieve this. The method's signature is: **public static** `LinkedList<ItemPair> maxPairing(LinkedList<Item> items)`.

3. How much will you gain if you use your method (instead of the shop owner's method) for the following list of item prices: 60 SAR, 100 SAR, 400 SAR, 600 SAR, 200 SAR, 80 SAR.

```
public class Item {
        private int id;
        private double price;
        public Item(int id, double price) {
                this.id = id;
                this.price = price;
        }
        int getId() {
                return id;
        }
        double getPrice() {
                return price;
        }
}
```

```
public class ItemPair {
        public Item first;
        public Item second;
        public ItemPair(Item first, Item second) {
                this.first = first;
                this.second = second;
        }
}
```