1. The problem consists on the creation of a generic class to handle the document buffer of a printer. Documents are printed using a numeric priority system (lower number => higher priority), being FIFO when priorities are equal. The printer has a limited buffer for documents and can only receive documents as long as its total memory is not overflowed. The printer works with objects, which implement the ***Document*** interface. Consider that a class named ***BufferDocument*** exists and implements the ***Document interface***.

```
public interface Document extends Comparable <Document> {
     Integer getPriority();
     Integer getSize();
     String getName();
     String getAuthor(); }
```

a) Create the generic class **PriorityBufferPrinter** (and its attributes), which parameter is or extends Document and implements the native generic interface **Iterable**.

b) Implement one constructor of **PriorityBufferPrinter** which receives the object's maximum allowed memory.

c) Create the method **addDocument** of **PriorityBufferPrinter** which adds a Document to the buffer in the order presented above. Use ordered insertion.

d) Create the method **getDocument** of **PriorityBufferPrinter** which returns and deletes from the buffer the next Document in the order presented above.

e) Create the method **delDocument** of **PriorityBufferPrinter** which deletes a document from the buffer (if it exists), given a name and an author.

f) Create the method **delDocumentsAbove** of **PriorityBufferPrinter** which deletes all the documents which size is superior to a given size.

g) Using **JUnit**, create test(s) for the methods in c), d), e) and f).

2. Given a list of non-repeated integers and k centers integers of this list, we intend to create k-sublists (k >= 2) such that in each k-sublist are the integers closest to the center, i.e. integers of initial list with least absolute difference with respect to the center k.
As an example, the following list L = {2, 9, 7, 5, 10, 15, 6, 12, 3} and centers C1 = 3, C2 = 6 and C3 = 10. The original list L will be divided into three Sublists: L1 = {3, 2}, L2 = {6, 7, 5} and L3 = {10, 9, 15, 12} that must be returned in a container that stores the <Center, sublist> pairs.

```
Map<Integer, LinkedList<Integer>> KsubLists(LinkedList<Integer> list,
                                                    ArrayList<Integer> centers)
```

Complementary Exercises

1. Write a **multiReverse** function that creates multiple sublists from an initial list. The sublists are created by reversing N elements from the received list. For example, consider the list L = {2, 9, 7, 5, 10, 15, 6, 12, 3, 11} with 10 elements and the number of elements to invert, N = 3. The original list L will be divided into the following sublists: L1 = {7,9,2}, L2 = {15,10,5}, L3 = {11,3,12,6}. If the remaining elements of the original list division are less than N, they will be added to the last list.
The sublists are returned in a map container that holds the <Number of the list, sublist> pairs.

```
Map<Integer, LinkedList<Integer>> multiReverse (LinkedList<Integer> list, Integer n)
```

2. The goal is to simulate a warehouse in which operates a forklift whose mission is to arrange crates of any type. These crates can be overlaid up to a maximum of 5 crates. The stacker only puts one crate on the floor if it can not be placed on top of another crate. The forklift removes a crate from the ground when there are no overlapping crates.

a) Define the Warehouse class

b) Add to the class the methods:

1. Stack a crate
2. Unwrap a crate
3. Visualize the warehouse