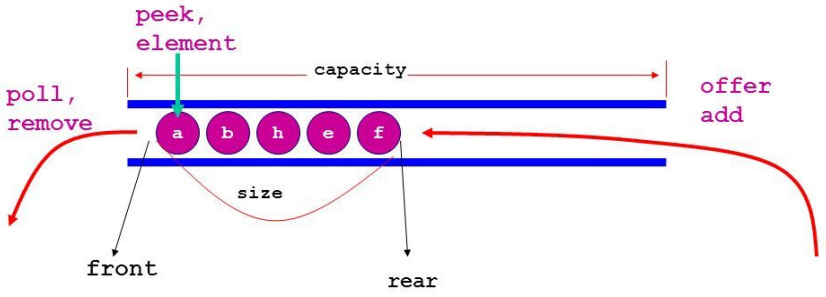
NOMBRE		ETAPA / CICLO	CURSO
		DAW	1º
APELLIDOS		ASIGNATURA/MÓDULO	CONVOCATORIA
		PROGRAMMING	2 EVALUATION
DNI	FECHA	NOTA	
	6-3-2020		

You can choose the exam model you want

MODEL A (Without Collections)

1. (3p.) Implements a generic queue that can store generic types. The queue must meet the following requirements:
   - The queue must have dynamic memory allocation.
   - The queue must have a method *remove* that returns the element placed on the front of the queue, and a method *add* to insert a element in the rear of the queue.
   - The queue **must not contain** getters or setters for the attributes, and these must be private.
   - The queue class must have a *toString()* method to display the elements of the queue.



2. (1p) Create a class Student that inherits from the Person class. The student class must store the student's NIA, and implements the necessary getters and setters to access and modify its attributes and the method *toString()* . (Pay attention to the scopes)

```
public abstract class Person {
	private String name;
	private String surname;

	public Person(String name, String surname) {
		this.name = name;
		this.surname = surname;
	}
	protected String getName() {
		return name;
	}
	protected String getSurname() {
		return surname;
	}
	protected void setName(String name) {
		this.name = name;
	}
```

NOMBRE	APELLIDOS	

```
        protected void setSurname(String surname) {
                this.surname = surname;
        }
}
```

3. (1,5p.) Write a method *removeElementAtIndex(int index)* that removes the element placed at the *index* position. This method returns true if the element has been removed successfully and false if not.

4. (1,5p.) Print a list of students sorted by surname (if two students has the same surname, sort them by name)

5. (2p.) Create a method to copy the queue into a file. And other to read the file stored into a file.

6. (1p.) Create a Main class that includes the following steps:

    1. Create a couple of students
    2. Create a queue
    3. Insert students in the queue
    4. View the queue
    5. Remove the last student inserted in the queue
    6. Visualize the new queue
    7. Copy the stack into a file

Important.- If you have not implemented any class or method, solve the rest of the exam assuming that you have implemented it.

NOMBRE	APELLIDOS	

You can choose the exam model you want

MODEL B (With Collections)

A company has asked us to develop a java application that simulates the voting process of the Eurovision contest. In this contest 22 countries participate, where each of the countries has to distribute a series of points among the rest of the countries, not being able to vote for himself or twice to the same country. The scores you must assign range from one to 8, then ten and finally 12 points (there are not nine or eleven).

At the end of the vote, it should show:

- A list by natural order of the countries together with the score obtained (alphabetical).
- A list ordered by scores from highest to lowest.

Clarifications:

- To load the names of the 22 countries, you should read these from a file called countries.txt
- The code must follow a well structured and correctly commented logic, following the good practices of object-oriented programming.

Puntuación:

1. **(1p)** Creation of the CountryFinalist class that inherits from the Country class, for the resolution of the exercise. You can see the Country class below. (Pay attention to the scopes).

```
public abstract class Country {

	private String name;

	public Country(String name) {
		this.name = name;
	}
	protected String getName() {
		return name;
	}
	protected void setName(String name) {
		this.name = name;
	}
}
```

2. **(2p)** Read from file the names of the 22 countries, and store them in a collection.
3. **(3 p)** Solution to the voting process
4. Listing Generation:
   4.1. **(1,25p)** Sorted by natural order.
   4.2. **(1,25p)** Sorted by scores.
5. **(1,5p)** Save the list sorted by scores in a text file.

Important.- If you have not implemented any class or method, solve the rest of the exam assuming that you have implemented it.