This first Java programming work consists of the partial implementation of two data types that are described in the book Estruturas de Dados e Algoritmos em Java, by António Adrego da Rocha, published by FCA in 2011. And of unitary test programs to validate the correction of methods already implemented or to be implemented.

It is intended to implement the ***Time*** data type using defensive programming with exceptions thrown in case of abnormal data situations that violate the preconditions and/or postconditions of the methods. For this purpose, an exception class called *InvalidValueException* is provided to indicate when a time information exceeds, by default or by excess, the acceptable value for hours, minutes or seconds or a total time of one day in seconds. The other exception class used *NullPointerException* already exists in the Java language.

1. Start by reading the ***Time*** class description (**Time.html** document). Then implement the missing *addTimes* and *subTimes* methods. Under no circumstances should you change the remaining methods otherwise the unitary testing program may stop working.
2. Complete the **TestTime.java** unitary testing program. Add tests using the assert statement to validate the correction of missing methods: *getHours*; *getMinutes*; *getSeconds*; *setHours*; *setMinutes*; *setSeconds*; *compareTo*; *addTimes* e *subTimes*.

It is intended to implement the ***Memory*** data type that represents a RAM memory. A memory of this type is used to search, select and sort values. It is a parametric implementation that only accepts elements that implement the *Comparable* interface. It uses defensive programming with exceptions thrown in case of abnormal data situations that violate the preconditions and/or the postconditions of the methods. The exception classes are available for this purpose: *IndexOutOfMemoryException*; *MemoryEmptyException* e *MemoryFullException*. The other exception class used *NegativeArraySizeException* already exists in the Java language.

1. Start by reading the ***Memory*** class description (**Memory.html** document). Then implement the missing methods: *insertPos*; *deletePos*; *biggerElement* and *smallerElement*. Under no circumstances should you change the other methods.
2. The **SimMemoryTime.java** simulation program allows simulating a RAM memory to store time information from the ***Time*** class. Run the simulations you think are necessary to test the methods already provided and the ones you have to implement. Two files are also provided that represent two memories with size 10: *times.csv* stores 8 valid time information, while *empty.csv* does not store any time information. The latter serves to test the methods in the empty memory situation.
3. Complete the **TestMemoryTime.java** unitary testing program which only tests the *biggerElement* and *smallerElement* methods. Add tests using the assert statement to validate the correction of missing methods: *getElement*; *setElement*; *insert*; *insertPos*; *delete*; *deletePos*; *search*; *copy* and *clear*).

The work must be delivered by April 30, 2022. Only one of the students in the group must send by email to adrego@ua.pt a zip file containing only the final files of the classes and the unit test programs: **Time.java**, **TestTime.java**, **Memory.java** and **TestMemoryTime.java**.

For any questions about Java you can consult the book Estruturas de Dados e Algoritmos em Java. You can and should also contact me by email with any doubts about the work.