Exercise 1
Discovery of the JAVA API

This exercise sheet has two major goals. First, you will learn to code your first lines of java. The second goal is to get you used to navigating in the Java API documentation.

The Java API documentation is the reference of the Java language, and contains all the informations needed on everything that Java has to offer. As a matter of fact, most commonly used objects (such as element containers or helper functions) are already defined in the Java API, and you do not need to re-create them each time you need one of them.

This documentation is available online at http://java.sun.com/javase/8/docs/api/.

1 Random number generator

In java, creating a pseudo-random number is done through the java.lang.Math.random() function, which is the method random() of the Math class, located inside the java.lang package. You will find more details on this function in the API documentation.

To find documentation on this method, use your internet browser to go to the main page of the API documentation. There, in the top-left, select the java.lang package. Then, in the bottom-left, select the Math class. In the right-hand frame, you will have the documentation for all the methods of the Math class. Search for the random() method.

1. Write a function generateRandom(int N) that generates N pseudo-random numbers through the java.Math.random() method.

2. Modify that function so it also computes the mean (µ) and the standard deviation (σ) of the generated sequence using the following formulae:
   
   $\mu = \frac{\sum_{i=0}^{n} x_i}{N}$
   
   $\sigma = \sqrt{\frac{\sum_{i=0}^{N} x_i^2}{N} - \mu^2}$

3. Test your function by calling it from the main function with multiple values for N (for example, N = 100 000, N = 1 000 000, etc.)

4. Does the generated sequence follow a uniform probability law? Remainder: if it follows a uniform law, then when N gets bigger:
   
   • µ approaches 0.5
   • σ approaches $\sqrt{1/12} \approx 0.2886751$
2 Simple keyboard reading: the Scanner class

In this exercise, we will see how to handle basic text input from the console. This is done through the `java.util.Scanner` class, which was introduced in Java version 5.

1. Open the Java API documentation and read the documentation for the `java.util.Scanner` class. The following methods in particular:
   - `nextInt()`
   - `nextFloat()`
   - `nextLine()`

2. At the beginning of your program, create a new `Scanner` instance, connected on the standard console input (called `System.in` in Java):
   ```java
   Scanner entry = new Scanner(System.in);
   ```

**Warning 1:** The string return as result to `nextLine` will be formed of all characters found in `System.in` from now until the next end-of-line character (an end-of-line character is made by pression the key `entry`). The end-of-line character will be read, but will not appear on the result string.

**Warning 2:** When reading a number, the eventual white characters (spaces, tabulations and end-of-line) preceding the number will always be 'consumed' and does not have any effect. However, the blanks following the number are not consumed; as for the end-of-line often type to end the number, and stay available of the next reading. This is the default way in many languages and cause many troubles when a number is followed by the reading a a string.

**Exercise:** To try the `Scanner` class, write in Java a program that calculate and display the mean of a sequence of positifs or null numbers, where the end is indicated by a negative number.

Example of execution:
Give the number (end by -1) : 14 8
10
12 6 -1
Mean: 10.0
3 String games

The goal of this exercise is to explore the java.lang.String class by testing several methods on strings and other values read on the keyboard. The goal is to write an executable class name StringTest which the main method runs the following operation:

**Game 1:** Read a number and convert it into a string (Ex: number 12345 will become the string ’12345’).

**Game 2:** Read a string entirely made of number and convert it into the integer it represents (Ex: the string ’12345’ will become the number 12345). The solution is in the statistical method of the java.lang.Integer class.

If you program this test following the first game, you will notice that when the reading of the string follows the reading of the number you need to ’clean’ the entry by using a lost fund string between those two operations.

**Game 3:** Same question as before, but with a float number (Ex: the string ’12345e4’ will become the number 12345e4). The solution is no longer in the statistical method of java.lang.Integer...

**Game 4:** Read a string representing the name of a town or city, remove eventual blank spaces at the beginning and end and display it in cap-locks.

**Game 5:** Read on the keyboard two strings s1 and s2 and display the answer to ’Does those two strings start with the same character?’”. The solution use the instance method charAt().

**Game 6:** Read two strings s1 and s2 and display the results of the expressions:
1==s2
s1.equals(S2)
s1.compareTo(s2)
s1.compareToIgnoreCase(s2)
Among other try the couple ’abcd’ and ’abcd’, then ’abcd’ and ’AbcD’.

**Game 7:** Read two strings s1 and s2 and display the answer to ’is s1 starting with s2?’, ’is s1 ending with s2?’ and ’Does s1 include s2?’.
4 Calculation of the factorial

4.1 Writing of the function

Write an executable class with the static long factorial1(int n) that calculate recurrency the n factorial, which mean the number:

\[ n! = n \times (n-1) \times (n-2) \times \ldots \times 3 \times 2 \times 1 \]

4.2 Flooding

Find the n value for which the flooding of integer value make this method unusable. One way to check if the value of n! is exact, knowing the value of (n-1)!, is by displaying the result of \( \frac{n!}{(n-1)!} \).

4.3 New function

To overcome this flooding issue, write a new function using the java.math.BigInteger objects, that you will call static BigInteger factorial2(int n), and can be use for as big integer as wanted.

4.4 In your opinion...

Could we have use the same name to factorial1 and factorial2 functions?