Assignment 2

Exercise 1 (3pt)

Describe the parts of this class:

```
public class Cube {
  private double depth = 10;
  private double width = 10;
  private double height = 10;

  private static int numOfCubes = 0;

  public static int getNoOfCubes() {
    return numOfCubes;
  }

  public Cube() {
    numOfCubes++;
  }

  public void scale(double scaling) {
    depth *= scaling;
    width *= scaling;
    height *= scaling;
  }

  public double getVolume() {
    return depth * width * height;
  }

  public double getDepth() {
    return depth;
  }

  public void setDepth(double depth) {
    this.depth = depth;
  }
}
```

Exercise 2 (2pt)

Draw an UML diagram of the previous class.

Exercise 3 (15pt)

Write a program that reads positive numbers until the user inserts a negative one (Note: the negative number does not count).

After having read the number the program should print:

- Their average
- Their minimum and maximum
- Their standard deviation (square root of the sum of the squared differences from the mean divided by total minus 1) Wikipedia
- Their median (the value separating the higher half of a data) Wikipedia

Note: for computing the last two statistics you need to store an arbitrary number of number. To do that you will need to implement the `InfinitArray` class.

This class should contain a normal array (`double[]`) as *storage* and provide an `add(double value)` method which insert values inside the storage.

When the storage is full, the add method should *create a new storage twice as big* and copy all values from the old storage to the new one.
This effects into having the same values inside the storage but its size will be larger. (Hint: store also the current length of the `InfinitArray` which can be different from `storage.length`)

You can add other fields, getters (e.g., `at(int index)`), or methods (e.g., `sort()`).

Please provide also an UML diagram of your class.

Example:

```
Insert numbers (terminate with negative number):
> 2.1
> 2.6
> 1
> -1

Their average is 1.9
Their min/max is 1.0/2.6
Their standard deviation is 0.6683312551921144
Their median is 2.1
```

Instructions

The solution of the exercises must be provides as a **java** (for the code, do not submit class files), **png** (if explicitly asked), and **pdf** (for potential text) files. The **files must be zipped** together before upload. Use the **terminal** to compile and execute the code.

Assignments not respecting these instructions will be ignored.