IIP First Partial - ETSInf
Date: November 26th, 2012. Time: 1.5 hours.

1. $\boxed{\text{6 points}}$ An application for the stock exchange uses a class for defining the stock values. Each stock is identified with the company name, and contains as data four real values that represent the opening, minimal, maximal, and current value for the stock. Each stock starts a session with the *opening* value, in the current moment has the *current* value, and during the session until this moment had a *minimal* and *maximal* value. Values for *current*, *minimal*, and *maximal* may change along the session.

   For that `Stock` class, **you must implement**:

   a) (0.5 points) Define private instance attributes for `company`, `opening`, `minimal`, `maximal`, and `current`.

   b) (1 point) Implement two constructors:
      - One that receives only company name and the opening value; `minimal`, `maximal`, and `current` will have the same value than the opening value (you can suppose that the opening value is correct).
      - One that receives the company name, the opening value and maximal and minimal values (you can supose they are correct); `current` will have the same value than the opening value.

   c) (1 point) Write all the consultor methods for the instance attributes.

   d) (1 point) Write a modifier method for the instance attribute `current` that must update properly `minimal` and `maximal`.

   e) (0.5 points) Implement a method `raising` that returns if the current stock is having benefits (i.e., current value is higher than opening value).

   f) (1 point) Write an `equals` method (that overrides the `equals` method of the `Object` class) to check that two stocks are equal. Two stocks are equal if and only if they are of the same company.

   g) (1 point) Write a `toString` method (that overrides the `toString` method of the `Object` class) that returns a `String` with: "`company`: `current minimal maximal`"; e.g., "Iberdrola: 12.30 10.34 13.21".

2. $\boxed{\text{4 points}}$ Considering the class of the previous question, **you must** implement a Java class `StockExchange` with the following methods:

   a) (1 point) A class method that, given two `Stock` objects, returns which one is more *volatile*, i.e., presents a larger percent difference between the minimal and maximal values of the stock. E.g., a stock with minimal value 12.3 and maximal value 15.7 presents a volatility of 27.6% ($\frac{(15.7-12.3)\times 100}{12.3} = 27.6$).

   b) (1 point) A class method that, given two real numbers `a` and `b`, returns a random real number in the interval [min(`a`,`b`), max(`a`,`b`)[.

   c) (2 points) A `main` method in which you must:
      1. Create a `Stock` object of the company "Iberdrola" and opening value 12.30. Then show its data on the screen.
      2. Create a `Stock` object with a given company (ask the user) and random values for opening, minimal and maximal; you must guarantee that the values are correct (i.e., minimal $\leq$ opening $\leq$ maximal); you can assume that all values are in the range [1,100[. Show this `Stock` on the screen.
      3. Ask the user for a new current value for the first `Stock` object and modify it properly. Show this `Stock` on the screen indicating if it is having benefits.
      4. Show on the screen the `Stock` object that is more volatile (using the method defined previously).

Solution:

Stock.java

```java
public class Stock {
  private String company;
  private double opening, minimal, maximal, current;

  public Stock(String c, double op) {
    company = new String(c);
    opening = minimal = maximal = current = op;
  }

  public Stock(String c, double op, double mi, double ma) {
    company = new String(c);
    opening = op;
    minimal = mi;
    maximal = ma;
    current = op;
  }

  public String getCompany() { return company; }
  public double getOpening() { return opening;}
  public double getMinimal() { return minimal;}
  public double getMaximal() { return maximal;}
  public double getCurrent() { return current;}

  public void setCurrent(double c) {
    if (c<minimal) minimal=c;
    else if (c>maximal) maximal=c;
    current=c;
  }

  public boolean raising() { return (current>opening); }

  public boolean equals(Object o) {
    return o instanceof Stock &&
           company.equals(((Stock) o).company);
  }

  public String toString() {
    return company + ": " + current + " " + minimal + " " + maximal;
  }
}
```

Stock.java

StockExchange.java

```java
import java.util.*;
public class StockExchange {

  public static Stock moreVolatile(Stock s1, Stock s2) {
    double v1 = (s1.getMaximal()-s1.getMinimal())/s1.getMinimal();
    double v2 = (s2.getMaximal()-s2.getMinimal())/s2.getMinimal();
    if (v1>v2) return s1; else return s2;
  }

  public static double randomInterval(double a, double b) {
    double m1 = Math.min(a,b), m2 = Math.max(a,b);
    return Math.random()*(m2-m1)+m1;
  }

  public static void main(String [] args) {
    Scanner kbd = new Scanner(System.in).useLocale(Locale.US);
    String cn;
    double cu1, op, min, max;
    Stock s1, s2;

    /** 1 **/
    s1 = new Stock("Iberdrola",12.30);
    System.out.println("Stock 1: "+s1);

    /** 2 **/
    System.out.print("Company name for stock 2: ");
    cn = kbd.nextLine();
    min = randomInterval(1,100);
    max = randomInterval(min,100);
    op = randomInterval(min,max);

    s2 = new Stock(cn,op,min,max);
    System.out.println("Stock 2: "+s2);

    /** 3 **/
    System.out.print("Current value for stock 1: ");
    cu1 = kbd.nextDouble();
    s1.setCurrent(cu1);
    if (s1.raising()) System.out.println("Stock "+s1+" is having benefits");
    else System.out.println("Stock "+s1+" is not having benefits");

    /** 4 **/
    System.out.println("The more volatile stock is "+moreVolatile(s1,s2));
  }

}
```
StockExchange.java