IIP

Test Unit 4 - Possible solution

Year 2016-2017

Name:

1. (7 points) Implement a datatype class `Lecture` that stores data on university lectures. You must develop:

   a) (0.75 points) Attributes for subject (`String`), teacher (`String`), start hour and minute (two integers), duration (in minutes, integer), and type of lecture (char)

   b) (0.75 points) Three public class (`static`) constant attributes that represent the different types of lectures: theoretical ('T'), seminar ('S'), and lab ('L'). **These constant attributes must be employed in any place that type is used**.

   c) (1 point) A constructor that receives as parameters subject name, teacher name, and start hour and minute. The constructor must initialise properly all attributes, using a standard duration of 90 minutes and theoretical as default type of lecture. You can suppose that all parameters will be correct.

   d) (1 point) `get` and `set` methods for subject, duration and type of lecture attributes with the following restrictions:

   - `set` method for duration must check that duration is positive
   - `set` method refered to type of lecture must avoid modifications to values not defined in the constants

   e) (1 point) An `equals` method that overrides the functionality of that `Object` class method; you must compare only subject, start hour and minutes, and type of lecture

   f) (1 point) A `toString` method that returns data on the lecture in the format `SUBJECT (TEACHER) HH:MM (DURATION min.) TYPE`, where `TYPE` must be "Theory", "Seminar", or "Lab". Hours do not need to follow the two-digit format

   g) (1 point) A method that returns the teacher effort for current lecture given the years of experience of a teacher (integer). Effort is a real number calculated as the number of hours of the lecture (real value) multiplied by a correction factor. Base correction factor is 1.1 for theoretical lectures, 1.25 for seminar lectures, and 1.5 for lab lectures. Final correction factor is obtained by decrementing in 0.01 base factor for each year of experience

   h) (0.5 points) A method that returns a `String` which represents the final hour of current lecture in "hh:mm" format (two digits format); you can suppose that no lecture finishes after midnight

```
public class Lecture {
  private String subject, teacher;
  private int h, m, dur;
  private char type;
  public static final char THEORY='T', SEMINAR='S', LAB='L';

  public Lecture(String sub, String tea, int sh, int sm) {
    subject = new String(sub);
    teacher = new String(tea);
    h = sh; m = sm;
    dur = 90; type = THEORY;
  }

  public String getSubject() { return subject; }
  public int getDur() { return dur; }
  public char getType() { return type; }

  public void setSubject(String sub) { subject = new String(sub); }
  public void setDur(int d) { if (d>0) dur = d; }
  public void setType(char t) { if (t==THEORY || t==SEMINAR || t==LAB) type = t; }

  public boolean equals(Object o) {
    return o instanceof Lecture &&
          this.subject.equals(((Lecture) o).subject) &&
          this.h==((Lecture) o).h &&
          this.m==((Lecture) o).m &&
          this.type==((Lecture) o).type;
  }

  public String toString() {
    String s=subject+" ("+teacher+") "+h+":"+m+" ("+dur+ " min ";
    if (type==THEORY) s+="Theory";
    if (type==SEMINAR) s+="Seminar";
    if (type==LAB) s+="Lab";
    return s;
  }

  public double teacherEffort(int years) {
    double hours, baseFactor=0, finalFactor;

    hours = dur / 60.0;

    if (type==THEORY) baseFactor = 1.1;
    if (type==SEMINAR) baseFactor = 1.25;
    if (type==LAB) baseFactor = 1.5;

    finalFactor = baseFactor - 0.01 * years;

    return hours * finalFactor;
  }

  public String finalHour() {
    int fh, fm, ftot;
    ftot = h*60 + m + dur;
    fh = ftot / 60; fm = ftot % 60;
    return fh/10+""+fh%10+":"+fm/10+""+fm%10;
  }
}
```

2. (3 points) Write a Java program class that, in its `main` method, asks for the data on two lectures (subject, teacher, initial hour and minute) and creates the corresponding `Lecture` objects. After that, the program must ask and set the durations of each of them. Finally, the program must call to another `static` method in the same class that receives the two lectures and returns the final hour of the longest lecture (if they are the same duration return any of them). The returned hour must be printed in the `main` method.

```
import java.util.*;

public class FinalHours {
  public static void main(String [] args) {
    Scanner kbd=new Scanner(System.in).useLocale(Locale.US);
    Lecture l1, l2;
    int h, m, d;
    String s, t;

    System.out.print("Lecture 1 subject: "); s=kbd.nextLine();
    System.out.print("Lecture 1 teacher: "); t=kbd.nextLine();
    System.out.print("Lecture 1 initial hour: "); h=kbd.nextInt();
    System.out.print("Lecture 1 initial minute: "); m=kbd.nextInt();
    l1=new Lecture(s,t,h,m);

    System.out.print("Lecture 2 subject: "); s=kbd.nextLine();
    System.out.print("Lecture 2 teacher: "); t=kbd.nextLine();
    System.out.print("Lecture 2 initial hour: "); h=kbd.nextInt();
    System.out.print("Lecture 2 initial minute: "); m=kbd.nextInt();
    l2=new Lecture(s,t,h,m);

    System.out.print("Lecture 1 duration: "); d=kbd.nextInt();
    l1.setDur(d);

    System.out.print("Lecture 2 duration: "); d=kbd.nextInt();
    l2.setDur(d);

    System.out.println("Final hour of longest lecture: "+finalHourLongest(l1,l2));
  }

  public static String finalHourLongest(Lecture l1, Lecture l2) {
    if (l1.getDur()<l2.getDur()) return l2.finalHour();
    return l1.finalHour();
  }
}
```