AP Computer Science Homework 12

Due date: Tuesday, January 3, 2017
Instructor: Mr. Alwin Tareen

Part A: Create a `SchoolRoster` Class

- Write a class called `SchoolRoster` that stores a list of student records and contains methods for calculating a student's GPA, as well as determining which students are seniors.
- The record keeping system will consist of an `ArrayList` data structure, which is declared as an instance variable called `roster`.
- Each student record consists of a **name**, the number of **credit hours** the student has attempted, the quantity of **grade points** the student has accrued, and the student's **GPA**. This information is encapsulated in the `StudentInfo` class, which means that each element in the `roster` ArrayList is an object of data type `StudentInfo`.
- The `StudentInfo` class has been provided for you. Be sure to look over it, to understand the various methods that are available.
- The `SchoolRoster` class provides a management system for all of the students in the school's roster. Generally, students can have their GPA's calculated, and they can determine if they are seniors. The `SchoolRoster` class should have the following properties:
- Instance variables:
  - `private ArrayList<StudentInfo> roster` This is the school's listing of students. It has been provided for you.
- The constructor:
  - `public SchoolRoster()` This sets up the `roster` to be an `ArrayList` of type `StudentInfo`. It has been provided for you.
- Mutator methods:
  - `public void computeGPA()` This method calculates and updates the GPA field for each student in the roster. A student's GPA is computed by dividing the grade points by the credit hours. The GPA for a student with 0 credit hours should be set to 0. *This method must be written first.*
  - `public void addStudent(StudentInfo pupil)` This method adds a `StudentInfo` object to the end of the `roster` ArrayList. It has been provided for you.
- Accessor methods:
  - `public boolean isSenior(StudentInfo student)` This method should return `true` if the given student has at least 125 credit hours and has a GPA of at least 2.0, otherwise, this method should return `false`. *This method must be written second.*
  - `public ArrayList<StudentInfo> fillSeniorList()` This method determines which students in the roster are seniors, and copies those students' records into a newly created `ArrayList`. In writing `fillSeniorList()`, you may call method `isSenior`. *This method must be written last.*
  - `public StudentInfo getStudent(int i)` This method returns the corresponding `StudentInfo` object at location `i`. It has been provided for you.

- Other methods:
  - `public String toString()` This returns a string containing each `StudentInfo` object's data. It has been provided for you.
- **Note:** You must write these methods in a particular order: `computeGPA` first, then `isSenior`, and finally `fillSeniorList`. This is because a student's GPA must be calculated before they can be designated a senior.
- You are provided with the files `StudentInfo.java`, `SchoolRoster.java`, `SchoolRosterTest.java`, and `SchoolRosterJUnitTest.java` to develop this program.
- Write your code in the file `SchoolRoster.java`, in the area indicated by `// YOUR CODE HERE`.
- When you have finished writing the `SchoolRoster` class, you may run the `SchoolRosterTest.java` test bench. Your output should look like the following:

```
King 45 171.0 0.0
Norton 128 448.0 0.0
Solo 125 350.0 0.0
Kramden 150 150.0 0.0

King 45 171.0 3.8
Norton 128 448.0 3.5
Solo 125 350.0 2.8
Kramden 150 150.0 1.0

[Norton 128 448.0 3.5, Solo 125 350.0 2.8]
```

- On your `BlueJ` project window, you should see a button labelled `Run Tests`. Press this button to run the `JUnit` tests.
- You should see a `BlueJ: Test Results` window pop up. If everything is correct, you should see a green bar that indicates that your code has passed the `JUnit` tests. If your program is incorrect, you will see a red bar. You can click on the method name to get more information about the problem. Otherwise, just click on the `Close` button, and you can go ahead and upload this program to `Web-CAT`.

Part B: Submission

- Submit your Java program `SchoolRoster.java` by uploading it to the Web-CAT automated grading platform.