Project 2: Quarterback (passer) rating formula

For this project, you will create a class to compute the rating of a NFL quarterback. The class will have some specific methods that will need to be implemented. There are two specific goals for this project:

1. Learn how to create a class with methods in it.
2. Learn how to perform arithmetic computations in Java.

Your class should be named "QuarterbackRating.java". There should be two constructor methods and other methods within this class. There will be another class with a main method that the instructor will provide to create QuarterbackRating objects and invoke their methods. This will be called "**testQBrating.java**". In this class, statistics about various quarterbacks will be read in from an input file, their ratings will be calculated based on your QuarterbackRating class and will then be printed to an output file. The input will be from a file called "**qbstats.txt**" and the output will be written to a file called "**qbratings.txt**". These three files will be provided on the course site in a few days.

I will provide the "**testQBrating.java**" file. There are a few points in this program where you must substitute your personal information for what is written in the code. Those places in the code are highlighted. For example, you should replace occurrences of my name (Mir Farooq Ali) with your own name. Similarly, replace the dates highlighted in the code with the dates when you work on your project, etc. ***Failure to do so will result in a point penalty.*** Test files will be posted on the website so that you can test your program's correctness. I also suggest that you create your own test input files and check your program with them. The posted test files are not meant to be a guarantee of a completely correct program.

Your Java class will be used to compute the rating of a NFL quarterback. The NFL quarterback rating formula (http://www.nfl.com/news/981202qbrate.html) is a complex formula that is comprised of four categories:

1. Percentage of completions per attempt
2. Average yards gained per attempt
3. Percentage of touchdown passes per attempt
4. Percentage of interceptions per attempt

If A, C, Y, T and I represent the number of attempts, number of completions, total yards, number of touchdowns and number of interceptions, respectively, then the point values of the four categories can be summarized as

1. Percentage of completions per attempt (*percentComp*) = $\left(\frac{C}{A}100 - 30\right)0.05$, 2.375 if computed value is greater than 2.375 or 0 if computed value is less than 0.
2. Average yards gained per attempt (*avgYards*) = $\left(\frac{Y}{A}100 - 3\right)0.25$, 2.375 if computed value is greater than 2.375 or 0 if computed value is less than 0.
3. Percentage of touchdown passes per attempt (*percentTD*) = $\left(\frac{T}{A}100\right)0.2$, or 2.375 if computed value is greater than 2.375.
4. Percentage of interceptions per attempt (*percentInt*) = $2.375 - \left(\frac{I}{A}100\right)0.25$, or 0 if computed value is less than 0.

The final computed quarterback (*QBRating*) is calculated as follows:

$$QBRating = \frac{(percentComp + avgYards + percentTD + percentInt)100}{6}$$

Methods and Field variables

The field variables and methods in the QuarterbackRating class are shown below. ***All these methods have to be completed with the specified names and parameters.***

```
private float Att;
private float Comp;
private float Yards;
private float TDs;
private float ints;

public QuarterbackRating()
public QuarterbackRating(int A, int C, int Y, int T, int I)

public float calculateCompletionPercentage()
public float calculateAverageYards()
public float calculateTDPercentage()
public float calculateIntPercentage()
public float getQBRating()
public void setAtt(int A)
public void setComp(int C)
public void setYards(int Y)
public void setTDs(int T)
public void setInts(int I)
```

Programming Standards

You'll be expected to observe good programming and documentation standards. All the discussions in class about formatting, structure, and commenting your code will be enforced. The given code satisfies the documentation standards below. You will be expected to follow these standards and others in all of your later programs. Some, but not necessarily all, specifics include:

- You must include header comments specifying your name, the compiler and operating system used and the date your source code and documentation were completed.
- The header comment must also include a brief description of the purpose of the program (sort of a user guide) — this should be in your own words, not copied from this specification.
- You must include a comment explaining the purpose of every variable you use in your program.
- You must use meaningful, suggestive (of function or purpose!) variable names.
- Precede every major block of your code with a comment explaining its purpose. You don't have to describe how it works unless you do something so sneaky it deserves special recognition.

Testing

You should test your program independently within the BlueJ environment to make sure that you implemented all the specified methods correctly. Later, you will be provided with test files to test your program with. You should be certain that your program produces the output given above when you use the

given input file and the sample test files provided on the course web site. However, verifying that your program produces correct results on a few test cases does not constitute a satisfactory testing regimen. You should make up and try additional input files as well; of course, you'll have to determine what the correct output would be.

Evaluation:

Your program will be submitted to the Curator system for evaluation of correctness (details are given below). You will have five submissions to allow you an opportunity to correct mistakes that your own testing has not found. **You will not be given more submissions for any reason!** Your submitted program will be assigned a grade based on the correctness of your program and evaluation of your program by a TA. The grade you receive in the email from the Curator is your correctness grade. The grade for the project for future assignments is computed by deducting any points for style and design from the correctness grade. Therefore, the correctness grade is the *maximum* that your program will receive. **The TA will always grade the *first* submission that has the highest points.** No exceptions will be made, so be sure that you have done everything you need to before submitting. Note: Check your grade a few minutes after submitting. **If you get "N/A" as a grade, do not submit again until you get a numeric grade.**

Submitting your Program

You will submit this assignment to the Curator System (read the *Student Guid*e), and it will be graded automatically. Instructions for submitting, and a description of how the grading is done, are contained in the *Student Guid*e. You will be allowed up to ***five*** submissions for this assignment. Use them wisely. Test your program thoroughly before submitting it. Make sure that your program produces correct results for every sample input file posted on the course website. If you do not get a perfect score, analyze the problem carefully and test your fix with the input file shown as part of the Curator output, before submitting again. The highest score you achieve will be counted. The *Student Guide* can be found at:
`http://www.cs.vt.edu/curator/`

Your class will be tested on the curator by invoking its methods from another class. This class will be made available in a few days. To emphasize the point again, you have to make sure that your class works well on its own and that it has all the methods implemented.

Program Compilation

Your program must compile and run under JDK 1.4.

Pledge

Every program submission for this class must include an Honor Code pledge. Specifically, you **must always** include the following pledge statement in the header comment for your program:

```
/**************************************************************************
 *
 * On my honor:
 *
 * - I have not discussed the Java language code in my program with
 * anyone other than my instructor or the teaching assistants
 * assigned to this course.
 *
 * - I have not used Java language code obtained from another student,
 * or any other unauthorized source, either modified or unmodified.
 *
 * - If any Java language code or documentation used in my program
 * was obtained from another source, such as a text book or course
 * notes, that has been clearly noted with a proper citation in
 * the comments of my program.
 *
```

```
 * - I have not designed this program in such a way as to defeat or
 * interfere with the normal operation of the Curator System.
 *
 * Your Name
 * Your PID
**************************************************************************/
```

Failure to include this pledge in a submission is a violation of the Honor Code.