University of Puerto Rico – Mayagüez
Department of Electrical and Computer Engineering

ICOM 4015: Advanced Programming

Programming Assignment 2 Exam Scoring Program

Introduction

This program is supposed to generate a numerical score and a letter grade for students based on their answers for an exam. Each question in the exam is of True/False type. There are exactly 25 questions on the exam.

Program Details

Your main program should be in a file called ScoreExams.java. Within the main method open the input and output files and then invoke a method called generateScores() that will process the input file and generate the output file containing student scores. Also write more methods, as you deem appropriate. The program should read the student answers from an input file called students.txt and store them in and ArrayList<Exam> where Exam is a class that represents the results of a single student. You should also store the answer key in a separate class AnswerKey with a isTrue(i) method that returns true if the correct answer for question i is T. You have freedmon to make any additional design decisions based on on your best engineering judgment. We encourage you to use this freedom to develop a piece of software that we all can be proud of.

Input File

The input file comprises of the following lines:

1. The first line contains the answer key in terms of 'T' and 'F' characters. An example of this line is given below.

   ```
   TFTFFFTTTFTFFTTTFFTFFTTFF
   ```

2. Note that there are exactly 25 characters in the above string indicating answers to 25 questions. An example of this is given below.

   ```
   123456789|TFTTFFTF FTFTTTTFFTFFTTFF
   ```

3. There could be an empty space between the T and F characters indicating that the student did not answer the question. In the above example, the student did not answer question 9.

4. The number of students in the input file is not fixed. Assume that there will be at least one student line.

5. Each correct answer is worth 4 points, each incorrect answer is worth –2 points and an unanswered question is worth 0 points. Your program should compare the student responses to the key provided and generate an aggregate score for the exam.

Output File

The output file should be named scores.txt and contain the following information:

1. The first line printed out should be your name. The second line should be the string "Exam scoring program". An empty line should follow these two lines.

2. After that, each line should contain the student ID followed by the exam score and the exam grade. The exam grade is a letter grade based on the following scale:

   93.00 - 100.00 A
   90.00 - 92.99 A-
   87.00 - 89.99 B+
   83.00 - 86.99 B
   80.00 - 82.99 B-
   77.00 - 79.99 C+
   73.00 - 76.99 C
   70.00 - 72.99 C
   67.00 - 69.99 D+
   63.00 - 66.99 D
   60.00 - 62.99 D-
   0 - 59.99 F

3. An example line is shown below:

```
123456789 91.00 A
```

The last line in the output file should be something like this:

```
Average score: 75.50
```

Hints:

1. Refer to the Java API at http://java.sun.com/j2se/1.4.2/docs/api/ whenever you need to get more information about particular classes.

2. Use the code from your labs for reading from and writing to files.

3. If you read in the student responses as a string, use the charAt() method of the String class to extract individual answers from the string.

4. Declare your exam score variables to be of type double.

5. Use the following code for formatting the exam scores to contain exactly two numbers after the decimal point. First you need to import the java.text.DecimalFormat class. Assume that examscore is a variable of type double.

```
DecimalFormat df = new DecimalFormat();
df.setMinimumFractionDigits(2);
df.setMaximumFractionDigits(2);
String str = df.format(examscore);
```

6. The String str will then contain the formatted number for printing out.

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 unit testing of your program using the Junit framework to make sure that you implemented all the above requirements 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. Your program for this project is a stand-alone program. No other program will invoke its methods. You should invoke the main method of the class in the Eclipse environment to run the program.

Team Work

You have a choice to work on this problem in pairs. However, each member of the team is responsible for participating in the design of the entire project. In particular, each member of the team must be familiar with all the details of the application.

Program Compilation

Your program must compile and run under JDK 1.4.

Continues...

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>
*
*********************************************************************/
```

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