Project02 (GPA Calculation, ~26 pts) CS201

Description

You are given the task of reading a student's name, semester letter grades, and semester hours attempted from one file; calculating the semester GPA; and writing the student's name and semester GPA to another file.

GPA Calculation

GPA = Total Quality Points / Hours Attempted

Quality Points for each class = Grade Conversion Points * Hours Attempted

The letter grades with corresponding point conversions is based upon the following table:

Letter Grade	Conversion Points
A	4 points
B	3 points
C	2 points
D	1 points
F	0 points

Examples:

A student earns an 'A' in a three hour course, a 'C' in a four hour course, and an 'F' in a two hour course. The overall GPA would be:

(1) Total Quality Points = (4 * 3) + (2 * 4) + (0 * 2) = 20

(2) Total Hours = (3 + 4 + 2) = 9

(3) GPA = 20/9 = 2.22

Letter Grade	Conversion Points	Hours Attempted	Quality Points
A	4 points	3	4 * 3 = 12
C	2 points	4	2 * 4 = 8
F	0 points	2	0 * 2 = 0
		Total = 9	Total = 20
		GPA= TotalQualityPoints / Hours → 20 / 9 = 2.22	

A student earns an 'A' in a four hour course, a 'D' in a four hour course, and a 'C' in a two hour course. The overall GPA would be:

(1) Quality Points = (4* 4) + (1 * 4) + (2 * 2) = 24

(2) Total Points = (4 + 4 + 2) = 10

(3) GPA = 24/10 = 2.40

Letter Grade	Conversion Points	Hours Attempted	Quality Points
A	4 points	4	4 * 4 = 16
D	1 points	4	1 * 4 = 4
C	2 points	2	2 * 2 = 4
		Total = 10	Total = 24
		GPA = 24 / 10 = 2.40	

Project02 (GPA Calculation, ~26 pts) CS201

Requirements

- You may work on this with up to one other person from the class – only if you both provide equal effort in coding. If working as a team of two, only turn in and submit one solution.
- (~ 2 pts) Include class header documentation that contains a professional class description and your name. The description should provide a good summary overview of the class, but it need not be long.
- The code for main below must be used. Re-align as needed after copying/pasting into your code

```
public static void main(String[] args) throws IOException
{
  Scanner inFile = openFile();
  String[] gradeSummary = processGrades(inFile);
  inFile.close();
  storeGpa(gradeSummary);
  System.out.println("Data processing complete.");
} // end main
```

- (~8 pts) Create a static method named **openFile** that meets these requirements:
  - Contains proper header documentation that includes a professional description, @throws, and @return. Note that the @ throws can simply be "@throws IOException".
  - Include internal comments to describe paragraphs of code.
  - Throw an IOException.
  - Prompt for and retrieves a file name from the user. Use the exact prompt shown in the sample run.
  - Check if the file exists and exits with a value of **1** if it does not.
  - Create and return a new Scanner object that refers to the opened input file.

- (~10 pts)Create a static method named **processGrades** that meets these requirements:
  - Contains proper header documentation that includes a professional description, @param, @precondition, @param, @throws, and @return.
    - Check the call in main to see what kind of formal parameter it takes.
    - The assumption (precondition) is that the input file parameter has been successfully opened.
    - The @throws can simply be "@throws IOException".
    - It will return an array containing two strings:

"student's full name"	"calculated gpa"

  - Include internal comments to document paragraphs of code.
  - Throw an IOException.
  - Read the student's full name from the file and stores it into element one of the string array.
  - Continually read each grade (String) and hours attempted (int) until there are no more tokens left.
    - The String class split method as discussed for the Kansas Census stats lab program will be very helpful here!
    - When tallying the quality points, you may hard code in the numbers 1, 2, 3, and 4 here. They correspond to an "A", "B", "C", or "D" read. There is no a need to add a 0 to the quality points if an "F" is read.
  - Calculate the GPA and store as a string into element two of the array. A helpful method is **Double.toString(gpa).**

- (~6 pts) Create a static method named **storeGpa** that meets these requirements:
  - Contains proper header documentation that includes a professional description, @param, and @throws. Note that the @ throws can simply be "@throws IOException". There is no return value from this method (i.e. void). Determine what the formal parameter is by looking at the method call in main.
  - Internal comments are optional as the method can be very short.
  - Throw an IOException.
  - Open a file for writing. The file name will consist of the student's full name (element 0) with ".csv" appended to it. There is no need to check if the file previously existed.

- Note: On some operating systems, it is a good practice to remove spaces from file names because spaces can make file management somewhat more difficult. Your program does not need to remove spaces from the file name.

- Write the student's full name, followed by a comma, followed by the GPA to the first line of the file. Two digits of precision must be used for the GPA. See the sample output files for examples.

Additional Note

- Be sure to do a great job with coding standards including comments. Please don't lose several points here.

Submission

- Before class: Submit the source code to D2L and print the source code.
- Beginning of class: Turn in the source code and staple as needed.

Sample Runs

Run 1

Input file: **grades.csv**

```
Jack Johnson
A,2
C,3
F,2
D,2
B,4
A,4
```

Resulting output file: **Jack Johnson.csv**

```
Jack Johnson,2.59
```

Screen output (user input in dark red)

```
Enter grade input file in the form filename.ext: grades.csv
Data processing complete.
```

Run 2

Input file: **grades2.csv**

```
Miguel Ortiz
C,4
A,3
F,2
```

Resulting output file: **Miguel Ortiz.csv**

```
Miguel Ortiz,2.22
```

Screen output (user input in dark red)

```
Enter grade input file in the form filename.ext: grades2.csv
Data processing complete.
```

Run 3 – showing output if the input file is not found

Screen output (user input in dark red)

```
Enter grade input file in the form filename.ext: grades
File open error: grades
```