

CMSC 215 Intermediate Programming Programming Project 2

The second programming project involves writing a program that produces a list of students who are eligible for membership in an honor society. This program consists of four classes. The first class is the `Student` class. It has three instance variables which include the student's name, credit hours earned and quality points. At a minimum, it should have following methods:

- A constructor that allows the student's name, credit hours and quality points to be initialized.
- A method named `calculateGpa` that returns the student's grade point average, which is computed as the quotient of the quality points and credit hours.
- A method `eligibleForHonorSociety` that returns whether a student's gpa exceeds the threshold for honor society membership, which applies to all students.
- A `toString` method that returns a string containing the student's name and grade point average.
- A class (static) method `setGpaThreshold` that sets the minimum gpa threshold for honor society membership.

The `Student` class has two subclasses. The first is `Undergraduate`. It has an additional instance variable that specifies the student's year - either a freshman, sophomore, junior or senior. At a minimum, it should have following methods:

- A constructor that allows the student's name, credit hours, quality points and class rank to be initialized.
- An overridden method `eligibleForHonorSociety` that applies the requirement that the student be either a junior or senior in addition to the requirement that applies to all students to be eligible for honor society membership.
- An overridden `toString` method that returns a string containing the student's name, grade point average, and year.

The second subclass is `Graduate`. It has an additional instance variable that reflects whether the student is pursuing a master's degree or doctorate. At a minimum, it should have following methods:

- A constructor that allows the student's name, credit hours, quality points and degree sought to be initialized.
- An overridden method `eligibleForHonorSociety` that applies the requirement that the student be seeking a master's degree in addition to the requirement that applies to all students to be eligible for honor society membership.
- An overridden `toString` method that returns a string containing the student's name, grade point average, and degree sought.

Finally, there should be a fourth class `Project2` that contains the main method. It should read in student information from a text file named `students.txt`. If the file does not exist, an error

message should be displayed, and the program should terminate. Each line of the text file will represent the information for one student. An example of how the text file will look is shown below:

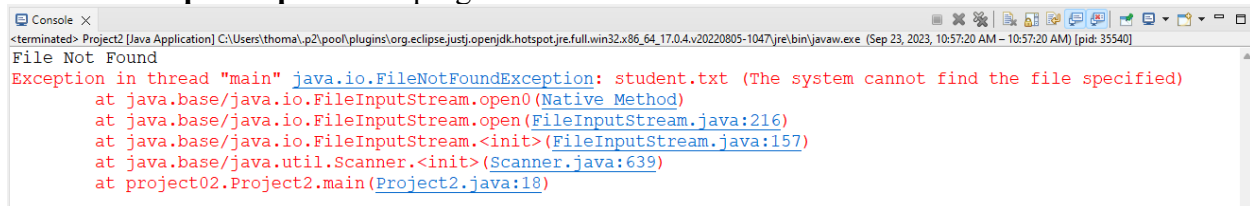
```
Brown,William 72 230 Junior
Johnson,Mary 21 77 Masters
Jones,Sally 32 95 Sophomore
```

For undergraduate students, the final value is the student's year. For graduate students, the final value is the degree the student is seeking. As the students are read in, `Student` objects of the appropriate type should be created, and they should be stored in an array list of type `Student`. As the students are read in the total of all grade point averages should be accumulated so that an average gpa can be computed. You may assume that the file that the data in the file will be formatted correctly.

Once all the student data is read in, the threshold for honor society membership should then be set to the midpoint of the average gpa and the highest possible gpa of 4.0 by calling the `setGpaThreshold` method. The threshold should also be displayed on the console. Then a report should be displayed on the console that lists all students who are eligible for membership.

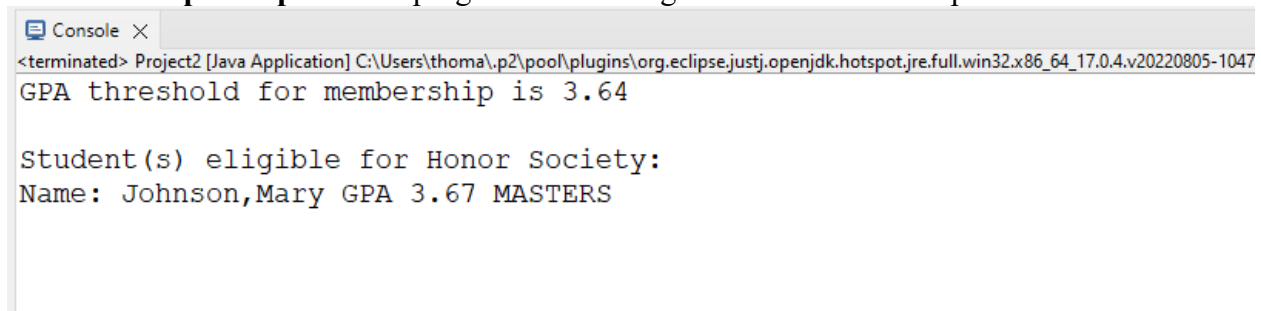
Be sure to follow good programming style, which means making all instance variables `private`, naming all constants using UPPERCASE (with appropriate access modifier and static designation) and avoiding the duplication of code. Furthermore, you must select enough test data to completely test the program.

Here is a **sample output** of the program when the File was not found.



```
Console X
<terminated> Project2 [Java Application] C:\Users\thoma\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.4.v20220805-1047\jre\bin\javaw.exe (Sep 23, 2023, 10:57:20 AM - 10:57:20 AM) [pid: 35540]
File Not Found
Exception in thread "main" java.io.FileNotFoundException: student.txt (The system cannot find the file specified)
    at java.base/java.io.FileInputStream.open0(Native Method)
    at java.base/java.io.FileInputStream.open(FileInputStream.java:216)
    at java.base/java.io.FileInputStream.<init>(FileInputStream.java:157)
    at java.base/java.util.Scanner.<init>(Scanner.java:639)
    at project02.Project2.main(Project2.java:18)
```

Here is a **sample output** of the program when using the test file with sample data shown above:



```
Console X
<terminated> Project2 [Java Application] C:\Users\thoma\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.4.v20220805-1047
GPA threshold for membership is 3.64

Student(s) eligible for Honor Society:
Name: Johnson,Mary GPA 3.67 MASTERS
```

Documentation Requirements:

Follow the naming conventions previously provided in the course announcements. Please follow these requirements:

Make sure your Java program is using the recommended style such as:

- **Javadoc comment with your name as author, date, and brief purpose of the program**
- *Comments for variables and blocks of code to describe major functionality*
- *Meaningful variable names and prompts*
- Class names are written in upper CamelCase
- Constants are written in All Capitals
- Use proper spacing and empty lines to make your source code human readable

Deliverables:

You are to submit two files.

1. The first is a .zip file that contains all the source code for the project. The .zip file should contain only source code and nothing else, which means only the .java files. If you elect to use a package the .java files should be in a folder whose name is the package name. Every outer class should be in a separate .java file with the same name as the class name. Each file should include a comment block at the top containing your name, the project name, the date, and a brief description of the class contained in that file.
2. The second is a Word document (PDF or RTF is also acceptable) that contains the documentation for the project, which should include the following:
 - a. A UML class diagram that includes all classes you wrote. Do not include predefined classes.
 - b. A test plan that includes test cases that you have created indicating what aspects of the program each one is testing as well **as the results of the testing**.
 - c. A short paragraph on lessons learned from the project.