CMSC 215

Programming Project 2 – University Honor Society

Daniel Smolsky

Section 1 – Approach

My approach to this project was to read the design guidelines and produce a plan to implement the requirements. My first step was to create the four base classes with each having the instance variable and methods required by the project outline. All of my logic to read in data from a txt file, calculate GPAs and assign a threshold GPA for honor students will be in my main project2 class. The project2 class will be implemented last once I have all the fundamental classes and instance variables created.

Section 2 – Assumptions

My code assumes that the txt file used by the project2 class is formatted correctly and that all of the data in the txt is logical. For example, my code does not have error catching in case the txt file has negative credit hour values or quality points. Additionally, if the student name isn’t in lastname,firstname format or if the students class year is spelled incorrectly or isn’t one of the default values certain functions of my program will fail. I chose not to implement any error catching in my program because the project outline said that we can assume that the data in the file will be formatted correctly, so I’m going off the assumption that there is error catching somewhere in the txt file creation process.

Section 3 – Not Implemented

Section 4 – User Guide:

Extract the .zip file and run the Project2.java file. Follow the comments in the Project2 class to change the directory of the students.txt file if the output of the program says the file is not found. If you want to add any more students and their relevant information, add it to the students.txt file in the format “lastname,firstname credithours qualitypoints grade” for example “Doer,John 19 77 Senior”.

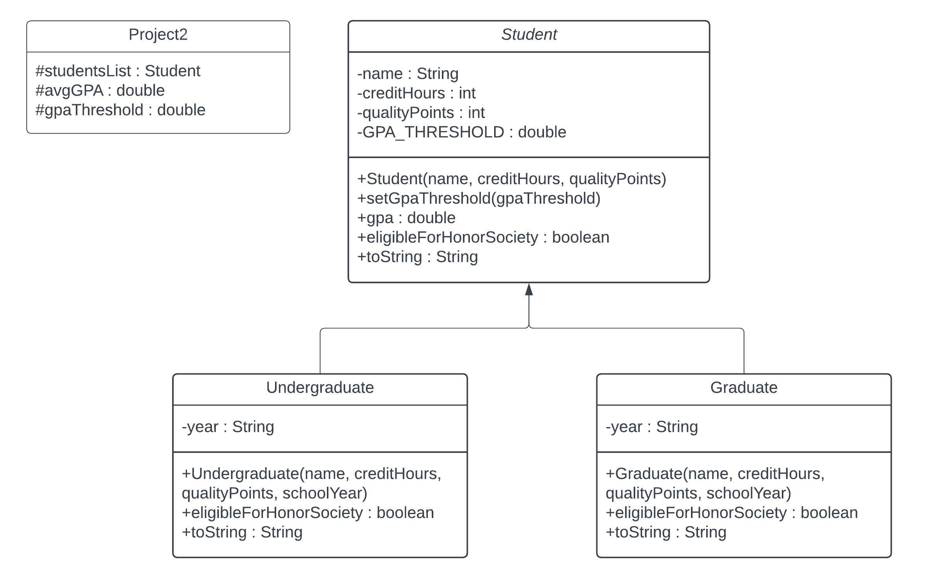
Section 5 – Lessons Learned

Created this project I learned how to implement subclasses and how to use the super method to call superclass constructors and methods. Additionally, I learned how to override superclass methods, how to annotate methods with the @ symbols and how to create file objects and read text data from them.

Section 6 – Possible Improvements

A few potential improvements for my project would have been using error handling if the students.txt file is not found instead of the .exists() method as error handling is more user friendly. Additionally, I could have implemented my rounding to two decimal places differently when I rounded GPAs. Instead of formatting strings I could have used Math.round, maybe that would also have been clearer. Finally, I’m not sure if the comments included in my code were enough to make things easy to understand for those not familiar with my code, maybe I could have been more descriptive.

Section 7 – UML



Section 8 – Source Code

My code can also be found on my [Github](https://github.com/Tarquinen/School-repo/tree/main/CMSC%20215/Week4/project2):

*/\**

*\* Daniel Smolsky*

*\* Programming Project 2: Student Honor Society*

*\* Feb 2, 2024*

*\* This class reads in the students.txt file, calculates the average student GPA, uses*

*\* average GPA to calculate a honor society GPA requirement and displays all the students*

*\* that meet the requirement.*

*\*/*

package project2;

import java.util.ArrayList;

import java.util.Scanner;

import java.io.File;

public class Project2 {

    public static void **main**(String[] *args*) throws Exception {

        ArrayList<Student> studentsList = new ArrayList<>();

*//the following 3 lines of code are only necessary because my current working directory is not*

*//the same directory as where students.txt file is saved. If they are in the same directory,*

*//replace these 3 lines with:*

*//File file = new File("students.txt");*

        System.setProperty("user.dir", "C:\\Users\\danny\\OneDrive\\Desktop\\Algo\\School-repo\\CMSC 215\\Week4\\project2");

        String currentDirectory = System.getProperty("user.dir");

        File file = new File(currentDirectory + File.separator + "students.txt");

*//exit the program if the file is not found*

        if (!file.exists()) {

            System.out.println("file does not exist, exiting...");

            System.exit(1);

        }

*//read data from the .txt one line at a time*

        Scanner input = new Scanner(file);

        while (input.hasNextLine()) {

            String[] line = input.nextLine().split(" ");

            String name = line[0].split(",")[1] + " " + line[0].split(",")[0];

            int creditHours = Integer.parseInt(line[1]);

            int qualityPoints = Integer.parseInt(line[2]);

            String schoolYear = line[3];

*//add students to the ArrayList*

            if (schoolYear.equals("Masters") || schoolYear.equals("Doctorate")) {

                studentsList.add(new Graduate(name, creditHours, qualityPoints, schoolYear));

            }

            else {

                studentsList.add(new Undergraduate(name, creditHours, qualityPoints, schoolYear));

            }

        }

        input.close();

*//calculate average GPA*

        double totalGPA = 0;

        for (Student student : studentsList) {

            totalGPA += student.gpa();

        }

        double avgGPA = totalGPA / studentsList.size();

*//set the honors society threshold*

        double gpaThreshold = Double.parseDouble(String.format("%.2f", avgGPA + ((4 - avgGPA) / 2)));

        Student.setGpaThreshold(gpaThreshold);

        System.out.println("The threshold GPA to be a honor society member is " + gpaThreshold);

*//print the students in the honor society*

        System.out.println("The following students are members of the honor society:");

        for (Student student : studentsList) {

            if (student.eligibleForHonorSociety()) {

                System.out.println(student);

            }

        }

    }

}

*/\**

*\* Daniel Smolsky*

*\* Programming Project 2: Student Honor Society*

*\* Feb 2, 2024*

*\* This class creates abstract Student objects. This class has 4 base variables and 4 methods*

*\* that the Student subclasses will inherit.*

*\*/*

package project2;

public abstract class Student {

    private String name;

    private int creditHours;

    private int qualityPoints;

    private static double GPA\_THRESHOLD;

    public Student(String *name*, int *creditHours*, int *qualityPoints*) {

        this.name = *name*;

        this.creditHours = *creditHours*;

        this.qualityPoints = *qualityPoints*;

    }

    public static void **setGpaThreshold**(double *gpaThreshold*) {

        GPA\_THRESHOLD = *gpaThreshold*;

    }

    public double **gpa**() {

        return (double) this.qualityPoints / this.creditHours;

    }

    public boolean **eligibleForHonorSociety**() {

        return this.gpa() >= GPA\_THRESHOLD;

    }

    @Override

    public String **toString**() {

        return "Students name: " + this.name + " GPA: " + String.format("%.2f", this.gpa());

    }

}

*/\**

*\* Daniel Smolsky*

*\* Programming Project 2: Student Honor Society*

*\* Feb 2, 2024*

*\* This class creates Undergraduate objects extending from the Student variables and methods.*

*\* This class has 1 instance variable and 2 methods that override methods within the Student class.*

*\*/*

package project2;

public class Undergraduate extends Student {

    private String year;

    public Undergraduate(String *name*, int *creditHours*, int *qualityPoints*, String *schoolYear*) {

        super(*name*, *creditHours*, *qualityPoints*);

        this.year = *schoolYear*;

    }

    @Override

    public boolean **eligibleForHonorSociety**() {

        if (this.year.equals("Freshman") || this.year.equals("Sophomore")) {

            return false;

        }

        else {

            return super.eligibleForHonorSociety();

        }

    }

    @Override

    public String **toString**() {

        return super.toString() + " " + this.year;

    }

}

*/\**

*\* Daniel Smolsky*

*\* Programming Project 2: Student Honor Society*

*\* Feb 2, 2024*

*\* This class creates Graduate objects extending from the Student variables and methods.*

*\* This class has 1 instance variable and 2 methods that override methods within the Student class.*

*\*/*

package project2;

public class Graduate extends Student {

    private String year;

    public Graduate(String *name*, int *creditHours*, int *qualityPoints*, String *schoolYear*) {

        super(*name*, *creditHours*, *qualityPoints*);

        this.year = *schoolYear*;

    }

    @Override

    public boolean **eligibleForHonorSociety**() {

        if (this.year.equals("Doctorate")) {

            return false;

        }

        else {

            return super.eligibleForHonorSociety();

        }

    }

    @Override

    public String **toString**() {

        return super.toString() + " " + this.year;

    }

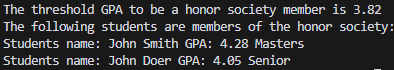
}

Section 9 – Test Plans

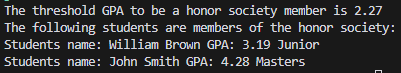
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test # | Purpose | Pos/Neg Test | Input Values | Expected Result | Pass/Fail |
| 1 | Confirm valid inputs produce correct output | Positive | Brown,William 72 230 Junior  Johnson,Mary 21 77 Masters  Jones,Sally 32 95 Sophomore  Smith,John 18 77 Masters  Doer,John 19 77 Senior | The threshold GPA to be a honor society member is 3.82  The following students are members of the honor society:  Students name: John Smith GPA: 4.28 Masters  Students name: John Doer GPA: 4.05 Senior | Pass |
| 2 | Confirm invalid inputs (negative ints) display erroneous results | negative | Brown,William 72 230 Junior  Johnson,Mary -21 77 Masters  Jones,Sally 32 95 Sophomore  Smith,John 18 77 Masters  Doer,John 19 -77 Senior | The threshold GPA to be a honor society member is 2.27  The following students are members of the honor society:  Students name: William Brown GPA: 3.19 Junior  Students name: John Smith GPA: 4.28 Masters | pass |
| 3 | Confirm program exits if Students.txt is not found | negative | No file input | file does not exist, exiting... | pass |
| 4 | Confirm program crashes if students.txt is not formatted correctly | negative | Brown,William 72 230 Junior  Johnson,Mary 21 77 Masters  Jones,Sally 32 95 Sophomore  Smith,John 18 77 Masters  Doer,John 19 77 Senior  DoerJohn1977Senior | Exception in thread "main" | pass |

Section 10 – Screen Shots

Test 1 –



Test 2 –



Test 3 –



Test 4 –

