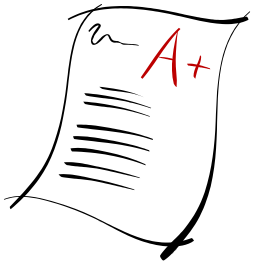

Lab 6 A GPA Calculator Part 2 ½: The Smell of OOP[1]

Due: Tuesday, October 18, 2011 at 23:00

"I never did anything worth doing by accident, nor did any of my inventions come by accident. They came by work." - Thomas Edison

1. Outcomes

1. Translate a UML class Diagram into Java Source Code
2. Document developed source code using JavaDoc
3. Understand the practice of Object Oriented Software Development

2. Introduction

Your GPA isn't everything. It is what you learn that is important. But then again, learning may lead to a high GPA.

In the second lab, you created a Java program which would calculate a persons' GPA by multiplying together course credits and quality points and averaging their value. In this lab, you will be repeating this process, just with a different design pattern.

In the first implementation, all of your code was in a single main method. It was long, very long. And if you decided that the number of courses was going to double, the amount of code you would have in your main function would double. On top of that, the program was not very re-usable. If the code required two calculations for GPA, one for courses in your major and one for courses not in your major, the amount of source code would increase again.

This week's design will attempt to resolve some of these difficulties. The code for the program will be split into three classes, only one of which you will need to write.

The two classes which are provided to your are the GPACalculatorMainClass and the CollegeCourseUI. The GPACalculatorMainClass is a driver program. It simply contains a main method which manages the College-CourseUI and CollegeCourse classes. This includes instantiating instances of CollegeCourses and the UI. While you are not responsible for writing this code, it attempts to provide a good example of how to write an application program, and it may be beneficial to read through the code and the comments. The CollegeCourseUI class is responsible for managing the user interface. This includes prompting the user to enter necessary information on a college course as well as providing rudimentary outputs to the console.

The class which you must develop is the CollegeCourse class. This class contains information which is necessary to calculate your GPA and your Major's GPA.[2] Each instance of a CollegeCourse contains details such as the course name, the course credits, the course letter grade, and a boolean value to indicate whether it is in your major of study. Appropriate accessor and mutator methods are provided as well.

[1] In homage to the Naked Gun...

[2] Your major GPA is a calculation of your GPA for courses which are directly in your major or program of study. For CE and SE students, this includes all courses with a CE, SE, or CS prefix.

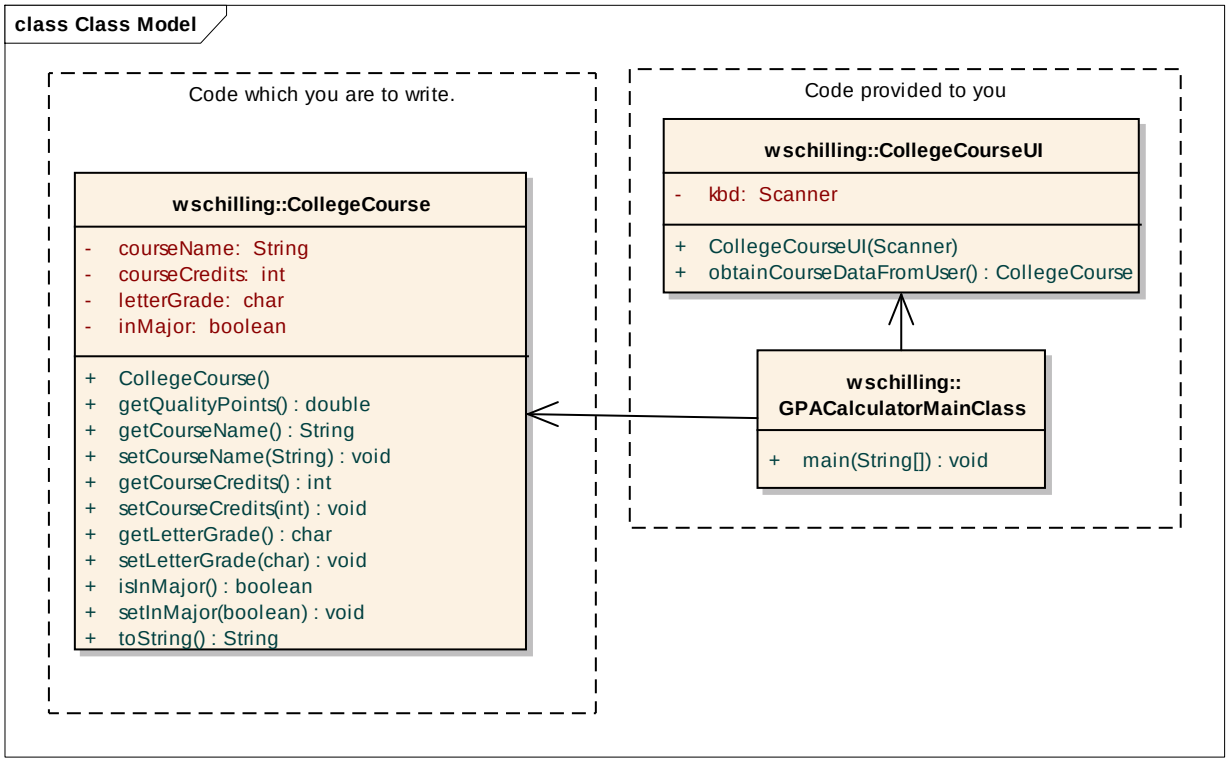

Figure 1: UML diagram showing design of baseline source code.

3. Lab Details

The first step is to create a new project in Eclipse. When you do this, you should create a package[3] of the form edu.msoe.se1011.lab6.< *yourname* > where < *yourname* > is your login id.

From the course website, download the CollegeCourseUI and GPACalculatorMainClass from the course website. You will need to change the package line near the top of the file your specific software package.[4]

Next, create a new class called CollegeCourse and implement the methods required via the CollegeCourse UML diagram. Pay specific attention to the names of methods, the names of parameters, variable scopes, and return types, as using the wrong type may cause significant problems with your implementation. Also, be certain that all methods and the class itself receives the proper JavaDoc comments. To aid in your development, JavaDoc for these classes has been included with this assignment document.

When all is finished, you should be able to run the program and match the output shown in the sample runs below.

3.1. UML Diagram

Figure 1 is a UML class diagram for this program. While there are many annotations that you have not learned about as of yet, the important part that you have been taught is how to translate a class representation into source code. In particular, your job is to write the code for the CollegeCourse class.

[3] Read Appendix 4 which discusses packaging.
[4] Essentially changing the schilling to your login id.

3.2. Specific Requirements

1. The getTotalQualityPoints method must use a switch statement to determine the correct return value.
1. The toString method shall return a string which is a mini-transcript entry for the given course. This includes the course name, followed by a carriage return, the credits, grade, and in major fields separated by tab characters.
2. An 'A' shall receive 4 quality points, a 'B' shall receive 3 quality points, a 'C' shall receive 2 quality points, a 'D' shall receive 1 quality point, and an 'F' shall result in 0 quality points.
3. The "CollegeCourse" method[5] shall assign a value of 0 to the courseCredits, an empty string "" to the courseName variable, false to the inMajorattribute, and a letter grade of 'P'.
4. All references to class scope variables shall use the "this" keyword.

4. Sample program Execution

```
Enter your name.
Leslie Nielson
How many courses have you taken?
4
Enter the name of the course.
Police Academy 1
How many credits was Police Academy 1worth?
4
What was your grade in Police Academy 1?
C
Was the course in your major(Y/N)?
Y
Enter the name of the course.
Flight Dynamics
How many credits was Flight Dynamicsworth?
5
What was your grade in Flight Dynamics?
B
Was the course in your major(Y/N)?
N
Enter the name of the course.
Shakespearian Studies
How many credits was Shakespearian Studiesworth?
3
What was your grade in Shakespearian Studies?
A
Was the course in your major(Y/N)?
N
Enter the name of the course.
Drivers Education 1
How many credits was Drivers Education 1worth?
2
What was your grade in Drivers Education 1?
D
Was the course in your major(Y/N)?
Y
*************************************
Unofficial Academic Transcript for LESLIE NIELSON
Police Academy 1
Credits: 4     Grade: C       In major: true
Flight Dynamics
Credits: 5     Grade: B       In major: false
Shakespearian Studies
Credits: 3     Grade: A       In major: false
Drivers Education 1
Credits: 2     Grade: D       In major: true
GPA:    2.64
Major GPA:     1.67
*************************************
```

[5] Which will be known as a constructor in the near future once next week's lectures begin.

5. Deliverables

1. Lab report, submitted in pdf format through the course website. This report should include:
    a. Name, date, title, and course information.
    b. A short description of what you did in this lab.
    c. Samples of the program executing showing the output written to the console. This output should show correct operation for all test cases given in this document.
    d. A short description of what went wrong and what went right during the lab.
    e. A description of what you learned from this lab.
    f. The Java source code you wrote as an appendix.
2. Your source code, namely CollegeCourse.java uploaded to the course website.

6. JavaDoc for the project

6.1. edu.msoe.se1011.lab5.wschilling Class CollegeCourse

```
java.lang.Object
  └ edu.msoe.se1011.lab5.wschilling.CollegeCourse
```

```
public class CollegeCourse
extends java.lang.Object
```

This class holds data which is relevant to a college course as it might be reported on a transcript. This includes, but is not limited to, name, grade, credits, and major.

Author:
schilling SE1011 October 5, 2009

Field Summary

private int	**courseCredits** This variable stores the course credits for the class.
private String	**courseName** This variable will hold the course name for the college course.
private boolean	**inMajor** This boolean will be true if the course is in your major of study and false if it is not.
private char	**letterGrade** This variable stores the letter grade for the given course.

Constructor Summary

CollegeCourse() This is the default constructor for the class.

Method Summary

int	**getCourseCredits**() This method will return the course credits for the class.
String	**getCourseName**() This method will obtain the course name from within this class.
char	**getLetterGrade**() This method will return the letter grade for the given class.
double	**getQualityPoints**() This method will return the quality points for this course.
boolean	**isInMajor**() This method will return true if the course is in the student's major and false otherwise.
void	**setCourseCredits**(int courseCredits) This method will set the course credits to the correct value.

void	**setCourseName**(java.lang.String courseName) This method will set the class name appropriately.
void	**setInMajor**(boolean inMajor) This method will set the inMajor attribute.
void	**setLetterGrade**(char letterGrade) This method will set the letter grade for the given class to the appropriate value.
String	**toString**() This method will return a textual representation for the given course. This shall include the name of the course, the credits, the letter grade, and whether or not the course is in the given major.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Detail

6.1.1. courseName

private java.lang.String **courseName**

This variable will hold the course name for the college course.

6.1.2. courseCredits

private int **courseCredits**

This variable stores the course credits for the class.

6.1.3. letterGrade

private char **letterGrade**

This variable stores the letter grade for the given course.

6.1.4. inMajor

private boolean **inMajor**

This boolean will be true if the course is in your major of study and false if it is not.

Constructor Detail

6.1.5. CollegeCourse

public **CollegeCourse**()

This is the default constructor for the class. It will set the credits to 0, the grade to 'P', the course name to an empty string, and the in major field to false;

Method Detail

6.1.6. getQualityPoints

`public double` **`getQualityPoints`**`()`

This method will return the quality points for this course. An A is worth 4 points, a B is worth 3 points, a C is worth 2 points, a D is worth 1 point, and an F is worth 0 points.

Returns:

The quality points based on the letter grade will be returned.

6.1.7. getCourseName

`public java.lang.String` **`getCourseName`**`()`

This method will obtain the course name from within this class.

Returns:

the courseName

6.1.8. setCourseName

`public void` **`setCourseName`**`(java.lang.String courseName)`

This method will set the class name appropriately.

Parameters:

`courseName` - the courseName to set

6.1.9. getCourseCredits

`public int` **`getCourseCredits`**`()`

This method will return the course credits for the class.

Returns:

the courseCredits

6.1.10. setCourseCredits

`public void` **`setCourseCredits`**`(int courseCredits)`

This method will set the course credits to the correct value.

Parameters:

`courseCredits` - the courseCredits to set

6.1.11. getLetterGrade

`public char` **`getLetterGrade`**`()`

This method will return the letter grade for the given class.

Returns:

the letterGrade

6.1.12. setLetterGrade

`public void` **`setLetterGrade`**`(char letterGrade)`

This method will set the letter grade for the given class to the appropriate value.

Parameters:

`letterGrade` - the letterGrade to set

6.1.13. isInMajor

`public boolean` **`isInMajor`**`()`

This method will return true if the course is in the student's major and false otherwise.

Returns:

the inMajor

6.1.14. setInMajor

`public void` **`setInMajor`**`(boolean inMajor)`

This method will set the inMajor attribute. If true, the course is considered in the student's major. If false, it is not within the students major.

Parameters:

`inMajor` - the inMajor to set

6.1.15. toString

`public java.lang.String` **`toString`**`()`

Returns

`This method will return a textual representation for the given course. This shall include the name of the course, the credits, the letter grade, and whether or not the course is in the given major.`
