Program #4
COP 2521_706
Fall 2005

Objectives:

- Gain experience using multiple classes
- Gaining experience with Java programs of multiple classes / objects
- To gain experience in the detailed design of Java classes as well as Constructors, toString, and other functionally-required methods.
- To gain additional experience with loops and conditional expressions.
- To gain experience with method calls and parameter passing
- To gain experience with instance variables and class variables.
- To gain experience in seeing association relationships and specifically aggregation relationships and how to implement these. UML diagrams must reflect the associations / aggregation relationships.
- To gain experience using DecimalFormat class and charAt method of String.

Requirements:

This program has much to do about objects and their dependencies and relationships. You will be creating a number of objects, performing some calculations, sending messages, using toString to print out values of objects, and formatting output.

First of all, to accomplish much of this program, you will need to study/review class variables (static variables), DecimalFormat class objects, charAt method of String objects, and more.

You are to develop a class called StudentBody. This is the driver class. This means that main() will be located within this class. StudentBody will be responsible for maintaining a number of Courses and Students in these courses.

For each Student object, you are to have instance variables: name, classification, number of hours taken, and grade points. Name and classification are strings, with the classification range "freshman" … "senior" and "graduate". Hours taken and grade points are integers.

For each Course object, you are to include instance variables course number, time of offering and two students enrolled in that course. Specifically, each Course object will be created with two students in it. Course number and time of offering are strings, while students are objects of type Student. You will create three courses, each of which as two students. Input data is presented ahead.

Inputs

So that we are all dealing with the same data, create your objects using this data:
(For those bent on getting additional credit, put these entries in separate lines in a sequential file and 'read' a string in one line at a time (need Stringtokenizer, of course, and a bit more…. …. don't ask 'how many points?' and you would need to process until there are no more inputs…)

```
// create a number of Student objects
Student stu1 = new Student ("GeorgeWashington", "senior", 120, 375);
Student stu2 = new Student ("JohnAdams", "senior",  110, 320);
Student stu3 = new Student ("ThomasJefferson", "grad", 80, 240);
Student stu4 = new Student ("JamesMadison", "junior",  82, 230);
Student stu5 = new Student ("JamesMonroe", "junior",   64, 148);
Student stu6 = new Student ("JohnQAdams", "grad",  70,  275);

// creates two Course objects via declaration.
Course course1 = new Course ("cop2551", "MW6pm", stu4, stu5);
Course course2 = new Course ("cis4328", "MW3pm", stu1, stu2 );
Course course3 = new Course ("cap6100", "MW7:45", stu3, stu6);
```

This code was copied from my StudentBody.java class file.

Outputs

Your output should appear as follows: (This is what my program generates with the data above). Use the formatting indicated below.

```
================================================
Course1:  cop2551  MW6pm

Student Name: JamesMadison
Classification: junior
Hours Taken: 82
Grade Points: 230
Grade Point Average is: 2.8


Student Name: JamesMonroe
Classification: junior
Hours Taken: 64
Grade Points: 148
Grade Point Average is: 2.31


================================================
Course2:  cis4328  MW3pm

Student Name: GeorgeWashington
Classification: senior
```

```
Hours Taken: 120
Grade Points: 375
Grade Point Average is: 3.12


Student Name: JohnAdams
Classification: senior
Hours Taken: 110
Grade Points: 320
Grade Point Average is: 2.91


==================================================
Course3:  cap6100  MW7:45

Student Name: ThomasJefferson
Classification: grad
Hours Taken: 80
Grade Points: 240
Grade Point Average is: 3


Student Name: JohnQAdams
Classification: grad
Hours Taken: 70
Grade Points: 275
Grade Point Average is: 3.93


        S t a t i s t i c s

 Total Number of courses offered is: 3

      Number of graduate courses is: 1

      Number of undergraduate courses is: 2

 Total number of students is: 6
```

You will note that for each student listed, his/her grade point average is given. You will need to accommodate this in the appropriate class. Also, note the formatting of the grade point average. Use ("0.##") for your pattern.

Also, please note that the items under Statistics will all come from Courses.java and Student.java. These must be class variables in Courses and in Student and are NOT to be hardcoded; rather, they are to be calculated as follows:

- For each object that is created, you are to bump a static course counter in the Course class..
- For each student object that is created, you are to bump a static student counter in the Student class.
- The number of undergraduate courses needs to be computed. When and where you do this is up to you. However, if the fourth character is greater than a 4, such as cap6100, then the course is a graduate level course.