Activity 4-2 Design Case Study

In this case we will think through a small design problem and build the main structure of a small application.

Setup

You are hired by the registrar's office of a certain college to create a record-keeping application for the college. Here is a description of its features:

- We manage the records of students as they take courses.
- We need to be able to add new students, and to enroll students to classes.
- Classes are determined by the department prefix, the course number, a section letter and the term when they were offered (we operate on three semesters, Fall, Winter and Spring).
- Each class counts for a number of units. 1 is the typical number, but we also have 0.5 and 0.25 units.
- Students may not enroll in a class that is the same as a class they have already taken (same department and number, possibly different section and term).
- Students who enroll in at least 3 units in Fall or Winter, or in 0.25 units in the Spring semester are considered "full time". Otherwise they are considered "part-time".
- Students may take up to 4 units in Fall or Winter, and up to 1 unit in the Spring.
- Students receive a grade for each class. It can be one of the normal grades, or an IP indicating a course that is "in-progress". Students may also receive "transfer credit" for a class, which means they have passed the class but there is no grade associated to it (and it's not part of a normal academic term).
- We need to be able to compute the GPA of students both overall as well as within a particular department, and also within a particular term.
- We need to be able to generate two kinds of reports:
  - A term-by-term report of the courses a student took each term and the number of units and gpa for that term, as well as overall number of units and gpa.
  - A department report listing the courses a student took in a particular department as well as computing total units and gpa.
- We need to be able to determine if a student can graduate. In order to graduate, a student must have (keeping it simple):
  - A declared major in their record, which is simply a choice of a department.

- At least 10 units of credit in that department, including transfer courses and courses that the student has passed.
- An overall GPA within the declared major of at least 2.0.
- At least 36 units of credit overall.
- An overall GPA of at least 2.0.

Our task for today will be to design some of the classes and interfaces that would support this functionality.

First steps

We start by considering a class that represents a term, we'll call it Term.

1. What data fields might this Term class contain? What are their types?
2. Do we need any other classes in order to be able to define the Term class?
3. What public methods might the Term class have?
4. Is your class handling limits on how many units a student can take in a semester and what the limit for "full-time" is, or will these happen somewhere else?

Now we consider a class to represent a particular course offering, we'll call it Section.

1. What data fields might this Section class contain? What are their types?
2. Do we need any other classes in order to be able to define the Section class?
3. What public methods might the Section class have?

Now we consider a class to represent a student, we'll call it Student.

1. What data fields might this Student class contain? What are their types?
2. Do we need any other classes in order to be able to define the Student class?
3. What public methods might the Student class have?

Now we want to somehow represent the act of a student taking a specific class. We'll represent this with an Enrollment class.

1. What data fields might this Enrollment class contain? What are their types?
2. Do we need any other classes in order to be able to define the Enrollment class?
3. What public methods might the Enrollment class have?

Imagine we also have a class that represents our connection to the database. We'll call it `Database`.

1. What data fields might this `Database` class contain? What are their types?
2. Do we need any other classes in order to be able to define the `Database` class?
3. What public methods might the `Database` class have?
4. Should this actually be an *interface* instead of a class?

Now we want classes to represent the various requests that might come to our system.

1. What are all the different requests that might come?
2. What pieces of information does each request need to hold?
3. Is there any common behavior that all these requests share?

What other classes would we need in our system?