2AA4 - Assignment 2

Please read this document very carefully. Follow instructions exactly. If you have any questions please post them to MS Teams or ask during office hours.

This assignment is due March 25th, by 11:59pm. Late submissions will not be accepted. Please manage/plan your time well.

I have created an Assignment 2 channel in Teams. If you have questions about the assignment, please post them there. Thank you.
Unless specifically stated, assume you are not allowed to import external libraries.

Purpose

The primary objective of this assignment is to assess your ability write mathematical specifications as well as demonstrate introductory design techniques.

Note:

- Unlike the last assignment, I have made the (given) MIS intentionally intuitive. If you have any questions please ask.
- When writing your MIS I know you may have the instinct to *get back at me* by making your answer hard to understand, but this is not a winning strategy. The TA(s) will be looking at your solution, not me, and you are not doing yourself any favours by making a correct solution difficult to determine is correct.

Overview

There are no other files associated with this document. You are responsible for submitting three files:

1. `Course.java`
2. `Student.java`
3. `A2.pdf`, you can submit a .docx or anything else easily readable.

See below for details on what you are responsible for completing.

Your Tasks

At the end of this document is an MIS for for a Course module. You are responsible for implementing Course and submitting it as Course.java. Furthermore, later in this document is an informal description of a Student You are responsbile for:

- Turning this informal description into a formal specification via a full MIS in your A2.pdf document
- Implement Student and submit it as Student.java

In your A2 document also **answer the following question**: In the informal description of Student, I ask you to **return an ArrayList of courses** for the getBestSchdule() method. This is a poor design choice. Explain why, and how you might fix it.

Informal Student Description

The overall purpose of the Student class is to add a number of courses the student wishes to enrol in (desired courses), and for the class to determine an optimal schedule for a subset of those courses such that *wasted time* is minimized. See more information regarding the class's methods below:

- Students are constructed by taking in a String which represents their name
- There is a getName() method which returns a String. It does what it should do.
- In the student class there will be a set of courses the student *desires* to take.
- There is a method named: addDesiredCourse(), which takes in a single parameter of type Course and adds it to the desired course set. Note if a course with the same course code already exists in their desired set, the course will not be added.
- There is a notion of a *valid schedule*. A valid schedule is a set of exactly 5 Courses which do not conflict with each other (see the Course MIS for conflict details). Furthermore, all Courses in the valid schedule are elements of the Student's *desired* courses.
- There is a getBestSchedule() method which takes no input and returns an ArrayList of Courses. This ArrayList will represent a schedule.
- The getBestSchedule() will return a valid schedule such that the number of days required to be on campus is minimized. I know it may not hold for all of you, but let's assume if you have lecture on a certain day, you must be on campus that day. Note, there may be multiple optimal schedules. As long as the schedule returned is optimal it is correct. **Note: You may assume there is at least one valid schedule before this is called.**
- **Hint:** For implementing getBestSchedule(), you can try to do something fancy if you wish, but it may be best to just brute force it. You may assume the desired set is less than or equal to 10 in magnitude. One way to brute force it would be to generate all the subsets of the desired set, and loop over it considering only subsets of magnitude 5. Generating the power set can be tricky but you are certainly capable of it.
- **Hint:** When formally specifying the getBestSchedule() method, it will be easier to split it into small spec. For example, consider creating a local function for specifying if a schedule is valid. Furthermore, if you're afraid that your specification is hard for the TA to understand, add some comments to aid them.

Submitting and Grading

This assignment will be submitted electronically via Avenue. Part of your assignment will be auto graded, part will be done manually. **It is your responsibility to test your code. Just because you code compiles or you spent a lot of time trying to complete the assignment, does not imply you deserve grades.** A rough breakdown is given below.

- `Course.java`: 20%
- `Student.java`: 30%
- Student MIS: 40%
- Answer to the ArrayList question: 10%

Code which does not compile will be heavily penalized. Code which has incorrect method/module names will be penalized up to 20% Good luck!

Academic Dishonesty Disclaimer

All of the work you submit must be done by you, and your work must not be submitted by someone else. Plagiarism is academic fraud and is taken very seriously. The department uses software that compares programs for evidence of similar code.

Please don't copy. The TAs and I want you to succeed and are here to help. Here are a couple of general guidelines to help you avoid plagiarism:

Never look at another assignment solution, whether it is on paper or on the computer screen. Never show another student your assignment solution. This applies to all drafts of a solution and to incomplete solutions. If you find code on the web that solves part or all of an assignment, do not use or submit any part of it! A large percentage of the academic offenses in involve students who have never met, and who just happened to find the same solution online. If you find a solution, someone else will too.

Course Module Interface

Uses

None

Syntax

Exported Constants

None

Exported Types

Course = ?

Exported Access Routine

Routine Name	In	Out	Exceptions
new Course	String	Course	
getCode		String	
hasLectureOn	$\mathbb{N}$	$\mathbb{B}$	
addLecture	$\mathbb{N}$, $\mathbb{N}$		
hasLectureAt	$\mathbb{N}$, $\mathbb{N}$	$\mathbb{B}$	
conflictsWith	Course	$\mathbb{B}$	

Semantic

Local Types

- Day = [1, 5]. Comment: An interval of integers from 1 to 5 inclusive. 1 represents Monday, 5 represents Friday.
- Hour = [8, 17]. Comment: An interval of integers from 8 to 17 inclusive. 8 represents 8:00am, 17 represents 17:00 in the 24-hour clock (5:00pm in the 12-hour clock).
- LectureTime = Tuple(day : Day, hour: Hour)

State Variables

- lectures: LectureTimes{}
- courseCode : String

Assumptions

- All input arguments are of the proper type.
- If a parameter of type $\mathbb{N}$ is used as a type Day or Hour, you can assume it falls within the proper interval.
- getStartTime(day) and getEndTime(day) are only called when hasLectureOn(day).

Access Routine Semantics

new Course(courseCode):

- transition: this.courseCode = courseCode
- output: out := this

getCode():

- output: out := courseCode

hasLectureOn(day)

- output: out := ∃(lecture: LectureTime | lecture ∈ lectures: lecture.day = day)
- Comment: Returns true iff there is a lecture on the given day.

addLecture(day, hour)

- transition: lectures := lectures ∪ {(day, hour)}

hasLectureAt(day, hour)

- output: out := (day, hour) ∈ lectures
- Comment: Returns true iff there is a lecture on the given day and time.

conflictsWith(other):

- output: out := ∃(time : LectureTimes | time ∈ lectures: other.hasLectureAt(time))