Combined OO concepts

Programming Projects:

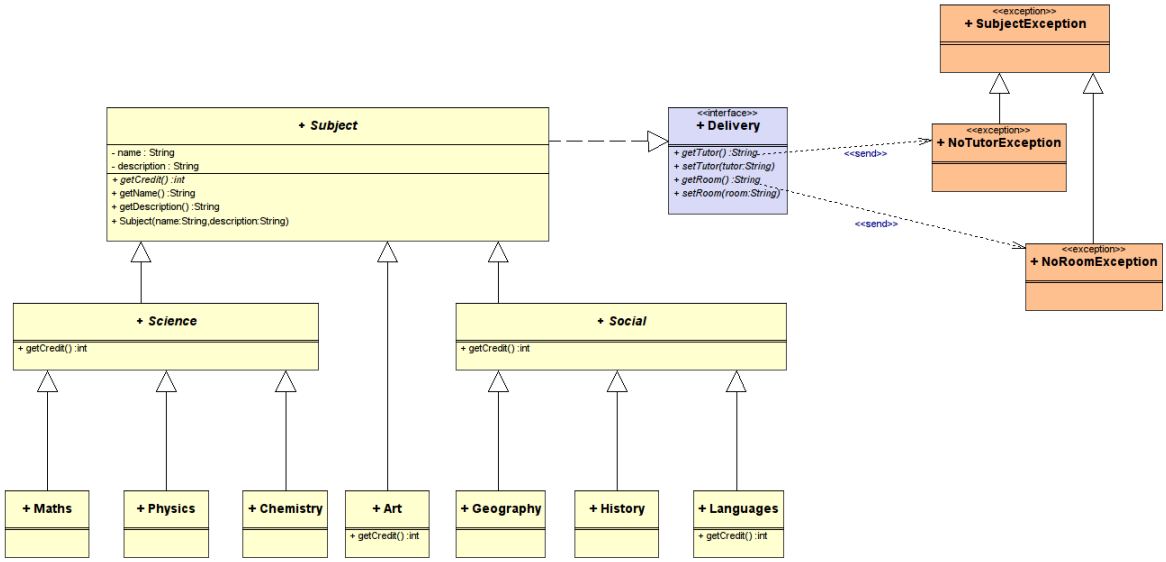
1) Programming from a Specification

For this exercise you will bring together several key object oriented programming concepts into one Java based solution.

The work involves building a mini-system (to support teaching subject delivery) that includes the use of inheritance, interfaces, polymorphism, exceptions and the correct use of program layout and Javadoc comments.

The system you are to build is shown within the UML class model below. Using this, along with the subsequent notes build a Java based solution of this system. Create this solution within an Eclipse project called '`lab4`'.

UML Class Model



Notes:

- `Subject`, `Science` and `Social` are all abstract classes (shown as Italic names on the UML class model)
- `Science`, `Art` and `Social` are sub-types of `Subject`.
- `Maths`, `Physics` and `Chemistry` are sub-types of `Science`.
- `Geography`, `History` and `Languages` are sub-types of `Social`.
- The 'name' and 'description' attribute values should be setup by the constructors (since no set methods exist).

- The `Delivery` interface is implemented by the `Subject` class (which will need new attributes adding also).
- The `getCredit()` method is declared as abstract within the Subject class.
- There are multiple implementations of `getCredit()`that return a constant value, rather than use an attribute. i.e. `getCredit()` is polymorphic.
- All `Science` based subjects have a credit amount of 100
- `History` and `Geography` have a credit amount of 50.
- `Languages` have a credit amount of 80.
- `NoTutorException` and `NoRoomException` are both sub-types of `SubjectException`.
- The `getTutor()` method throws a `NoTutorException` if the tutor name has not been set.
- The `getRoom()` method throws a `NoRoomException` if the room number has not been set.

2) Building a Driver

Create a '`Driver`' class that contains an array of `Subject` instances. Within the array create and add at least 5 different subject deliveries. e.g.

```
Subject [] subjectArray = new Subject[5];

Maths m1 = new Maths();
m1.setTutor("D.Trump");
m1.setRoom("The white house");

subjectArray[0] = m1;

Art a1 = new Art();
a1.setTutor("Salvador Dali");
a1.setRoom("Figueres");

subjectArray[1] = a1;
```

etc...

Add a `for` loop that iterates through the array, displaying the subject name, description, credit, tutor name and room number of each subject instance.

Ensure a `try…..catch` block is included in the driver class to catch and display any `SubjectException` that can be thrown.

Now, to improve the system make sure each constructor allows the “tutor name”, and the “room name” to be set when the object is created.

Check list.

Is the visibility scope of all attributes and methods set correctly?
Are all your classes named correctly?
Are all your attributes named correctly?
Is all the Javadoc complete?
Is all tabbing and general formatting correct?