**Tutorial 3**

**GC 312**

**Problem 1:**

We want to create a UML [class diagram](http://www.uml-diagrams.org/class-diagrams-overview.html) for online shopping. The purpose of the diagram is to introduce some common terms for online shopping - Customer, Web User, Account, Shopping Cart, Product, Order, Payment, etc. and relationships between. It could be used as a common ground between business analysts and software developers.

Each customer has unique id, address, phone number and email and is linked to exactly one **account**. Account owns shopping cart and orders. **Customer** could register as a web user to be able to buy items online. Customer is not required to be a web user because purchases could also be made by phone or by ordering from catalogues. Web user has login name which also serves as unique id and password. **Web user** could be in several states - new, active, temporary blocked, or banned, and be linked to a **shopping cart**. Shopping cart belongs to account.

Account owns customer orders. Customer may have no orders. Customer orders are sorted and unique. Each order could refer to several **payments**, possibly none. Every payment has unique id, date of paid and details, and is related to exactly one account.

Account has an id, billing address and open/close date.

Each **order** has number, and ordered/shipped date, shipped address and order status. Both order and shopping cart have **line items (which has quantity and price)** linked to a specific product. Each line item is related to exactly one product. A product could be associated to many line items or no item at all. **Product** has an id, name and supplier.

**Problem 2:**

This problem demonstrates relationships between classes. It’s from an imaginary application that models university courses. There are two types of person: Lecturer and Student. Person will have a getName and getEmailAddress methods.

* Each Course object maintains a list of the students on that course and the lecturer who has been assigned to teach that course
* The Course object has behavior that allow the adding and removing of students from the course, assigning a teacher, getting a list of the currently assigned students, and the currently assigned teacher.
* Teachers are modeled as Lecturer objects. As a lecturer may teach more than one course there is an association between Course and Lecturer. The “taught by” relationship shows that a Course only has a single teacher, but that a lecturer may teach several Courses.
* Each Lecturer object also maintains a list of the Courses that it teaches.
* There is a similar relationship between Course and Student. A course is attended by zero or more Students, and a Student may attend multiple courses.

**Problem 3:**

Consider the air transportation system. Many flights land and depart from city’s airport. Some of the big cities may have more than one airports. Every flight belongs to specific airline. The planes may have many flights to different airports. Each plane is identified with serial number and model and hours of flight. There are specific pilots for each airline and they fly many flights. Each flight is identified by flight number and date on which flight is scheduled. The passenger reserves a seat for a flight. The seat is identified by a location. Prepare a class diagram for above description.

**Problem 4:**

Develop a class diagram for following description:
The telephone agent uses an order registry and customer catalog to obtain access to an order & a customer respectively.
The order registry uses an order number as a qualifier to select particular order instance. A customer catalog uses customer name and phone number as a qualifier to select particular customer.
The attributes of an order are the order numbers and time when it is placed. The order consists of many items.
An item has item\_number, a quantity, unit price. It also has reference to catalog item which represents listing.
When an order is cancelled or committed, it cancels or commits each of its items first. When an order’s total price method is invoked, the order calls the total price method of each of items and returns the sum.