Software Modelling, fall 2009, exercise 5

1. Categorize the following relationships into generalization, composition, aggregation, association or dependency. You could draw a small class diagram on each.

   - A dining philosopher uses a fork
   - In Unix a file is either a normal file or directory
   - A file consists of bytes
   - A polygon is composed of an ordered set of points
   - A student uses Java in programming project
   - A person has birthdate
   - Mouse and keyboard are I/O-devices
   - A path two connects villages

2. Let's extend last week's class diagram of Monopoly game.

   For the rules, see eg. wikipedia.

   Here are some things that should be found in the diagram.

   There are several types of squares:

   - start square
   - prison
   - chance and community chest (in Finnish sattuma and yhteismaa)
   - stations and institutions
   - normal streets (with a name)

   Monopoly game must know the location of both the start square and the prison.

   There is some kind of actions attached to every square.

   There are *cards* attached to chance and community chest squares. Each card has some sort of action attached to it.

   There are several kinds of actions attached to squares and to cards. There is yet no need to specify the nature of these actions.

   you can build maximum of 4 houses or one hotel to a normal street. Some of the players can own a street/streets. Players have money.

   Now it is no need to attach any methods to the classes, that's going to happen next week.

3. Let's complete the LAL system class diagram with the parts we left out in the last week's exercises.

4. In the end of the course Advanced programming a very useful class called ArrayList is introduced. The objects of the ArrayList class can be thought as automatically self-increasing, variable length arrays, to which other objects can be stored. So ArrayList objects are a sort of object storages.

   Find ArrayList class in the Java API. (http://java.sun.com/j2se/1.5.0/docs/api/)

Let's study the features of ArrayList. Let's not pay attention to the use of ArrayList, but to it's location the Java API class hierarchy. We are interested in what's the upper class of ArrayList, which "sibling classses" (that is, other subclasses of the upper class) it has, what is the upper class of the upper class, which are the "cousin classes" of ArrayList, etc.. . .

Find out also what interface classes ArrayList implements. What does each interface mean?

Draw a class diagram of the situation. If the diagram grows too big, don't put all the information in. At least there is no need to mention many of the method names.

5. ArrayList is probably one of the best ways to implement a connection between several objects.

   Many students can participate a course. A course has a method addParticipant(), that a student uses to join a course. A course also has a method printParticipants() which works as one might presume.

   Fill in the implementation of the class Course:

```
import java.util.*;

public class Course{
    private ArrayList<Student> participants;

    public Course(){ ... }

    public void addParticipant(Student o){ ... }

    public void printParticipants(){ ... }
}
```

   The class Student can be implemented as follows:

```
public class Student{
    private String name;

    public Student(String n){ name=n };

    public void print(){ System.out.print( name ); }
}
```

   Write a main program that you use to test the functionality of the program.

   You may find information on use of ArrayList in Java course book, or form internet, for example from the following link: