CS 308
SOFTWARE ENGINEERING

Midterm Exam 1

Thursday, April 30, 2009
13:40 - 15:40

PLEASE NOTE:

PROVIDE ONLY THE REQUESTED INFORMATION AND NOTHING MORE. UNREADABLE, UNINTELLIGIBLE AND IRRELEVANT ANSWERS WILL NOT BE CONSIDERED. PLEASE ANSWER EVERY QUESTION ON THE SPACES PROVIDED.

DO NOT FORGET TO FILL IN YOUR NAME AND ID-NUMBER BELOW.

NAME:
ID-NUMBER:

Question	**Maximum** Points	**Points** Received
1	15	
2	5	
3	10	
4	15	
5	5	
6	10	
7	15	
8	15	
9	10	
Total	100	

Question 1 (15 Points)

Briefly answer all questions (with two or three sentences) in the space following each question:

a) Briefly explain what *functional requirements* and *non-functional requirements* are. Provide an example requirement for each type.

b) What is traceability concerned with in a requirements specification document(RSD)? What is source traceability? What is design traceability?

c) Why is traceability an important concern in requirement engineering?

d) Suppose that you are in the phase of requirement elicitation (discovery). You have already interviewed your customers and created use case models to discover as many requirements as you can. However, you strongly suspect that there are some implicit requirements (e.g., domain knowledge that is so familiar to customers that customer find it hard to articulate or think that it is not worth articulating). Describe a requirement elicitation technique that can help you surface those implicit requirements.

Question 3) (10 points)

Write a use case description (i.e., ordered sequence of use case steps) for withdrawing money at an automatic teller machine (ATM). Suppose that to withdraw money one need to insert an ATM card into the machine, verify himself/herself with a password, choose the bank account to withdraw money from from a set of accounts associated with the card, and enter the amount to be withdrawn. Describe the normal flow only (the normal flow is the one in which there is no exceptions and errors)!

Question 4) (15 points)

Draw a class diagram corresponding to the following situation:

A grocery store (e.g., a supermarket) sells items. There are two types of items: edible (i.e., any item that can be used as food) and non-edible. Some items are sold by weight, and some are sold per unit. Some items are taxable, while others are not. Some items have special prices when sold in groups (e.g., 2 for $3). A purchase may contain many items.

In your class diagram, show only the classes/abstract classes/interfaces, associations, and multiplicities for the associations. Do not include any operations! Clearly indicate interfaces and abstract classes (if any) in your design.

Furthermore, your design should be as **flexible** as possible. In particular, you need to pay attention to the following points:

1) An item is either edible or not and this fact does not change during the lifetime of the item, whereas the pricing strategies may change during the lifetime of the item.
2) Each pricing strategy is associated with a certain set of operations. For example, for the taxing strategy, whether or not an item is taxable, we may have an operation that computes the tax (this function can then return 0 for non-taxable items)
3) Although the set of operations for a pricing strategy stays the same, the ways those operations are handled may change over time. For example, the way we compute taxes for edible and non-edible items may differ over time.
4) Although the set of operations for a pricing strategy stays the same across different items, we may want to implement those operations in different ways for different items. For example, we may want to compute the tax differently for oranges and apples.

Question 5 (5 points)

For each of the following associations, indicate whether it should be an ordinary association, a standard aggregation, or a composition and briefly explain why?

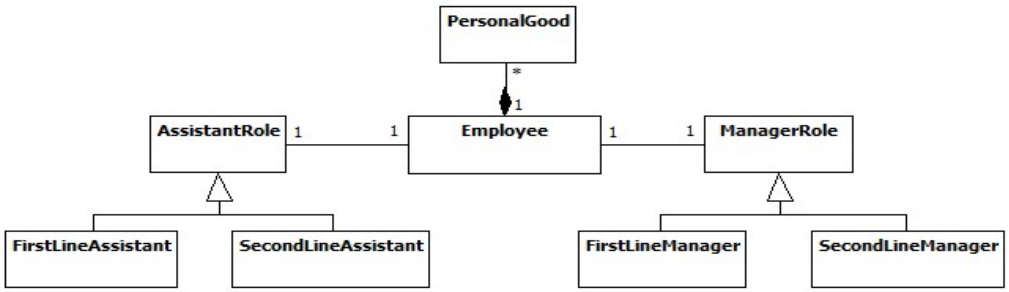
a) A telephone and its handset.

b) A school and its teachers.

c) A book and its chapters.

Question 6) (10 points)

Write the class declaration for the **Employee** class (only for the Employee class!). You must include only associations and constructors in your declaration. Furthermore, you must implement the constructors. You can use Java or C++.



Question 7) (15 points)

Suppose that you are developing a digital photo album application. A digital photo album application manages a collection of a series of digital photos. Some examples are: Google's Picasa, Adobe's Photoshop Elements, and Microsoft's Windows Photo Gallery.

The size of a digital photo (especially a high resolution one) may be a couple of megabytes and it may take a while to read that photo from disk and load it into the memory. Considering that a digital photo album may contain thousands of digital photos, when a user opens a photo album, you (as the developer) would like to reduce the need to load into memory large numbers of photos from disk, when not all of them will be needed.

Which design pattern would you use to be able to program the application *as if* all photos were located in memory? How would your solution work? What would be the benefits of your solution?

Question 8) (15 points)

Suppose that you are developing a software system that extensively deals with points in a two-dimensional space. You plan to design a **Point** class, storing only the **x** and **y** coordinates of the point (and nothing else), which are passed as inputs to the constructor of the class. Furthermore, since the system frequently creates **Point** objects and object creation is expensive, you have already decided to cache all the **Point** objects that would be created during an execution. Your cache works as follows: 1) the **x** and **y** coordinates of the point to be created are passed to the cache, 2) the cache checks whether a **Point** object with those coordinates has already been created, and 3) if so, a reference (or a pointer) to the previously created object is returned; otherwise, a new object is created, stored in the cache, and then a reference (or a pointer) to the newly created object is returned.

Note that all the **Point** objects are created by the cache. Therefore, modules in your system may have references (pointers) to the **Point** objects in the cache. Many modules may have a reference to the same **Point** object. A module may keep a reference as long as needed.

Which design pattern would you use to design the **Point** class and WHY?

Hint: If a module has a reference to a **Point** object in the cache, the module wants to make sure that the reference always points to the same two-dimensional point that the module thinks it points to.

Question 9) (10 points)

Draw the sequence diagram representing the following interaction between a client, a library object, a database object, and a loan object:

A client searches for a book using a library system. The client types in the ISBN number of the book that he/she would like to borrow, the system looks up the book in a database, if a copy of this book is available, a loan object is created.

You can use this page as a work page. You do not have to turn this page in.