Høgskolen i Agder
Avdeling for økonomi- og samfunnsfag

EKSAMEN

Emnekode:	**IS-202**
Emnenavn:	**Programming Related Topics**
Dato:	**Friday 15 December 2006**
Varighet:	**0900-1300**
Antall sider inkl. forside:	**4**
Målform:	**English**
Tillatte hjelpemidler:	**Students are allowed to use any books or other printed material and handwritten notes.**
Merknader:	**Read the entire problem set before you start to write your answer.** **You are asked to write a class in problem 1, and to make modifications to it in problems 2 and 3. You may answer 2 and 3 by writing program fragments and explaining where they should be inserted, or you may write a single class in response to all the problems and use comments to identify the answers to problems 2 and 3.** **You do not have all the information you need to write a complete program. In particular you do not have a database definition. Declare a variable or use a suitable constant, e.g. insertOrderSQL or "INSERT ORDER", to represent the missing information, and use comments as necessary to explain your assumptions.**

Introduction

This problem set concentrates on some aspects of payment in webshops. Most webshops will not ship anything unless they are sure they will get paid. Guarantee of payment can be achieved in a number of ways. For example, payment can be collected by the carrier upon delivery. In Norway this is known as "postoppkrav", which means you have to go to the post office and pay before they will give you the package. This is simple to implement, as no payment information is needed. However this kind of service is normally restricted to domestic markets, and can only be used for physical products.

Other forms of payment, like credit cards and services such as PayPal, work better for international markets and products which can be delivered electronically. With these the customer has to select a payment method and provide payment credentials before the order is accepted.

Problem 1: Handling customer checkout

The checkout process in a typical webshop requires you to select shipping method, payment method, provide payment details and confirm order. We will simplify a bit here and assume that there are no shipping options, and payment is always with credit card.

The checkout process starts when the customer wants to place an order and leave the shop. We assume that the customer has put some products in the shopping basket and that it is stored in the database (or in the Session object – it's not important). Here we are only concerned with what happens when the user clicks the checkout button.

a) Write an html page with a payment form. The form must have input fields for customer name, credit card number and security code. The submit button should be labelled "Place Order".
(Counts 10%)

b) Write a servlet to handle the payment form. The servlet should
   - Register the order in the database. Write the jdbc code, but use the string "INSERT ORDER" as the SQL code for registering the order and moving the content of the shopping basket to the order (in a real system this would be a rather complex operation).
   - Get the order id and payment amount from the database. Use "SELECT ORDER" as the SQL statement, and assume the columns are named "orderid" and "amount".
   - Get the payment by calling PaymentServer.createPayment() (see appendix A).
   - Write the order confirmation page. The message on the page should contain the customer name, order id and amount.

   (Counts 35%).

Problem 2: What if the payment fails

A potential problem that the servlet must be able to handle is that the payment can fail. There may be a problem with the credit card information, or the payment service may simply be unavailable. If the servlet just carries on as described above and the payment fails, the webshop will register an order, but will not get payment for it.

a) Assume that PaymentServer.createPayment() throws PaymentException if the payment fails. The servlet should handle the PaymentException by deleting the order by sending the SQL statement "DELETE ORDER" to the database. Make the necessary changes to the servlet.
(Counts 10%)

b) You might be tempted to handle the payment first. Will that work any better? Explain. (Hint: Consider a situation where the product is sold out while the customer is typing payment information, which could easily happen with flight or concert tickets)
(Counts 15%)

Problem 3: What if...

Payment is not the only thing that can fail. The computer can crash, the network connection may be lost, the electrical power may be lost, etc.

a) Ideally there should only be two possible outcomes of running the servlet: An order is registered and paid for, or both the order and payment are refused. Consider a situation where the customer has clicked the "Place Order" button, and the server crashes before it can send the response. What are the possible outcomes?
(Counts 15%)

b) It is impossible to prevent all of the unwanted outcomes, but we can make it easier to fix them by leaving an *audit trail*. This is a log of all the operations that have been carried out. After a crash an administrator can examine the log and correct any problems such as an order that has been registered but not paid for and vice versa. Modify the servlet to write a log that makes it possible to determine what happened when the servlet was run. You can use the log() method to write the log. (Your servlet has inherited log() from HttpServlet, see appendix B)
(Counts 15%)

Appendix A – PaymentServer

```
/**
 * This class represents a connection to a server that can
 * handle credit card payments. You do not have to create
 * PaymentServer objects, just call the static method createPayment()
 */
public class PaymentServer {
    /**
     * Create a payment transaction
     *
     * @param shopName The name of the webshop to pay to.
     * @param customerName The name of customer making the payment
     * @param creditCardNo The customers credit card number
     * @param securityCode The customers credit card security code
     * @param amount The amount to pay in NOK
     *
     * @throws PaymentException if the payment cannot be made.
     */
    public static void createPayment(String shopName,
                                     String customerName,
                                     String creditCardNo,
                                     String securityCode,
                                     int amount)
        throws PaymentException
    {
    }
}
```

Appendix B – HttpServlet.log()

Excerpt from HttpServlet javadoc

javax.servlet.http
Class HttpServlet

Method Detail

---

log
`public void` **`log`**`(java.lang.String msg)`

Writes the specified message to a servlet log file, prepended by the servlet's name.

Parameters:

`msg` - a `String` specifying the message to be written to the log file

---