

UNIVERSITY OF ESSEX

Undergraduate Examinations 2006

WEB APPLICATION PROGRAMMING

Time allowed: **TWO** hours

Candidates must answer **ALL** questions.

The paper consists of **three** questions.

All questions are of equal weight.

The percentages shown in brackets provide an indication of the proportion of the total marks for a **QUESTION** which will be allocated.

**Please do not leave your seat unless you are given permission by an invigilator.
Do not communicate in any way with any other candidate in the examination room.
Do not open the question paper until told to do so.
All answers must be written in the answer book(s) provided.
All rough work must be written in the answer book(s) provided. A line should be drawn through any rough work to indicate to the examiner that it is not part of the work to be marked.
At the end of the examination, remain seated until your answer book(s) have been collected and you have been told you may leave.**

Candidates must answer ALL questions.

Question 1

- (a) Answer the following with regard to Cascading Style Sheets (CSS) and JavaScript:
- (i) Explain why it is generally better to use CSS to style HTML than to use in-line HTML style attributes. [5%]
 - (ii) Describe the basic syntax of CSS with the aid of an example that would set the default colour to green for level three headings. [5%]
 - (iii) Briefly explain the advantages and disadvantages of using external versus internal style sheets, giving an example of where each one would be preferred. [5%]
 - (iv) Outline how to write client-side JavaScript code to automatically create a table of contents for a web page. [15%]
- (b)
- (i) With the aid of a comprehensive example, explain the structure of a typical web application URL. [10%]
 - (ii) Outline the actions taken by both the web browser and the web server (assuming that both are using HTTP 1.1) when a user clicks on a web page that includes 3 image (tags) that refer to 2 distinct images (i.e. one of the images appears in two places on the same page). Your answer should include the following points: [20%]
 - The expected number and duration of any socket connections.
 - The effect of the port number.
 - The type and number of HTTP requests.

Briefly explain why web servers should be multi-threaded programs. Describe what is meant by thread-pooling, and explain why it is used.
- (c) Within the context of web applications, describe what is meant by the model view controller (MVC) architecture. [10%]
- (d) Discuss the strengths and weaknesses of relational databases versus object-oriented databases (such as *Db4o*) for making Java objects persistent. [20%]
- (e) Briefly discuss how the *Once and Once Only* principle can be related to good practice in designing web applications. [10%]

Question 2

The `Bank` class below shows some prototype Java code for making a transaction between two bank accounts. The Object Database `Db4o` (version 5.0, as used in the CC213 laboratory) is being used make the transaction persistent. The `transfer` method of `Account` will throw an `Exception` if the transfer fails on that particular account (which happens if the transfer would make the account overdrawn, or if the account has been suspended). As it stands however, the `transfer` method of `Bank` does not conform to one or more of the ACID criteria for a good transaction.

- (a) Define the ACID criteria, and for each criterion explain whether the given code complies with it. [20%]
- (b) Explain how much money would be in each bank account (i.e. accounts *abc* and *def*) after running the `main` method. The initial money in each account is specified in the calls to the constructor, and both accounts are operational (i.e. not suspended). [10%]
- (c) Write a correct version of `Bank.transfer` that does conform to the ACID criteria and explain the changes you needed to make to the current version. [30%]
- (d) Briefly explain the terms *coarse-grained* locking and *fine-grained* locking, and explain why the given `Bank.transfer` method implements a coarse-grained policy. [15%]
- (e) State the conditions for potential deadlock within a multi-threaded system that uses fine-grained locking, and describe a simple but safe strategy for avoiding such deadlock. Explain how the `Bank.transfer` method could be modified to implement such a strategy. [25%]

```
public class Bank {
    public static void main(String[] args) {
        ObjectContainer db = Db4o.openFile("bank.odb");
        Account abc = new Account("abc", 1000);
        Account def = new Account("def", 2000);
        db.set(abc);
        db.set(def);
        transfer(db, abc, def, 800);
        transfer(db, abc, def, 800);
        db.close();
    }

    public static synchronized void transfer(
        ObjectContainer db, Account from, Account to, int amount)
    {
        // this code is buggy!
        try {
            // credit the balance of 'to' by amount
            to.transfer(amount);
            db.set(to);
            // debit the balance of 'from' by amount
            from.transfer(-amount);
            db.set(from);
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

Question 3

(a) Briefly describe the following XML-related concepts:

- (i) Well-formed document; [5%]
- (ii) Valid document; [5%]
- (iii) Elements and Attributes; [5%]
- (iv) The XML document shown below is intended to mark-up data relating to a CD music catalogue. The XML describes the fact that the artist Bob Dylan released an album called desire in 1976. [20%]

```
<Catalogue>  
  <BobDylan desire = "1976" />  
  ...  
</Catalogue>
```

Ignoring the ellipses (i.e., the ‘...’), state with reasons whether the document is well formed XML. Describe any flaws in the XML document design that are evident in the above sample, and rewrite the data in XML that overcomes these flaws.

(b)

- (i) State and briefly describe the four possible scopes that a JavaBean may be declared to have within a JSP page. [20%]
- (ii) Explain how JavaBeans together with the use of `<jsp:setProperty>` can be used to simplify form handling. [5%]
- (iii) With the aid of an example for each tag, describe the purpose of `<jsp:include>` and `<jsp:forward>`. [10%]

Question 3 continues...

Question 3 continued

- (c) Figure 1 shows a web-based form that enables researchers to register for some competitions, where the asterisks denote mandatory fields.

name*	<input type="text"/>
email*	<input type="text"/>
affiliation	<input type="text"/>
homepage	<input type="text"/>
intend to enter*	<p>Tick one or more</p> <p><input type="checkbox"/> Character Recognition</p> <p><input type="checkbox"/> Word Recognition</p> <p><input type="checkbox"/> Text Locating</p> <p><input type="checkbox"/> Reading</p>

Figure 1: A simple competition registration form.

- (i) Identify the types of HTML tags used as form input elements (including the buttons). [10%]
- (ii) Discuss the strengths and weaknesses of using the following JSP/Java programming techniques to generate and process forms similar to those shown in Figure 1: [20%]
- writing JSP code specifically for each form element;
 - automatic or semi-automatic form generation and processing from Java Objects (e.g. with the use of specially designed Java interfaces, or other techniques).

END OF PAPER CC213-2-SP