UNIVERSITI UTARA MALAYSIA
PEPERIKSAAN AKHIR SEMESTER KEDUA SESI 2008/2009
FINAL EXAMINATION SECOND SEMESTER SESSION 2008/2009

KOD / NAMA KURSUS: TIW3043 – KEJURUTERAAN WEB / WEB ENGINEERING

CODE / COURSE
TARIKH : 15 APRIL 2009
DATE
MASA : 08.30 AM – 11.00 AM (2 ½ JAM / HOURS)
TIME
TEMPAT : DSB K. TM
VENUE

ARAHAN :
1. Buku soalan ini mengandungi BAHAGIAN A (12 soalan), dan BAHAGIAN B (1 soalan) di dalam SEBELAS (11) halaman bercetak tidak termasuk kuit hadapan.
2. Sila jawab SEMUA soalan di dalam ruang jawapan yang disediakan.

INSTRUCTION :
1. This book script contains SECTION A (12 questions), and SECTION B (1 question) in ELEVEN (11) printed pages excluding the cover page.
2. Answer ALL the questions in the spaces provided.

NO. MATRIK :
MATRIC NO. (dengan perkataan/in words) 
(dengan angka/in numbers)

NO. KAD PENGENALAN :
IDENTIFICATION NO.
PENSYARAH :
LECTURER

KUMPULAN GROUP : NOMBOR MEJA TABLE NO. 

JANGAN BUKA KERTAS SOALAN INI SEHINGGA DIBERI ARAHAN
PLEASE DO NOT OPEN THIS QUESTION BOOKLET UNTIL INSTRUCTION IS GIVEN.

SULIT/CONFIDENTIAL
SECTION A (80 MARKS)

1. Describe briefly THREE (3) characteristics of web applications.  

(6 marks)

2. State THREE (3) of web engineering best practices when you build industry-quality web applications.  

(6 marks)

3. Describe briefly THREE (3) umbrella activities that are crucial for a web successful web engineering project.  

(6 marks)
4. Who is stakeholder? Give **FOUR (4)** techniques/mechanisms you can use for communication in order to gather requirements from stakeholders. (6 marks)

5. As we know that risks are categorized in **THREE (3)** main categories. Describe briefly each of them. (6 marks)

6. State **TWO (2)** capabilities should exists in a modeling language. (4 marks)

7. There are four main output from analysis modeling for web applications. Describe briefly **THREE (3)** of them. (6 marks)
8. The Relationship-navigation analysis (RNA) approach is organized into five steps. Describe briefly THREE (3) of them.

(6 marks)

9. Olsina and his colleagues have prepared a quality requirement tree that identifies a set of technical attributes which lead to high quality web applications. Describe briefly THREE (3) of them.

(6 marks)

10. Describe briefly THREE (3) pragmatic interface design guidelines for web applications that has been introduced by Nielsen and Wagner.

(6 marks)
TIW3043 Web Engineering

11. Describe briefly THREE (3) characteristics of a good information architecture for web applications.  

(6 marks)

12. Describe briefly about MVC architecture.  

(6 marks)
SECTION B (30 MARKS)

1. You are required to complete the searching course operation using MVC architecture. This program was developed using Java Server Page (JSP), Java Servlet and Java Bean. The program involves 6 files which are:
   - SearchForm.jsp -- JSP file
   - SearchCourseServlet.java -- Servlet file
   - Course.java -- bean file
   - CourseDB.java
   - DBConnection.java
   - Result.jsp -- JSP file

Figure 1 and Figure 2 show the table structure of course table and records that have been stored in the course table. Figure 3 shows the location of java file stored in the specified package.

![Table structure of course](image)

Figure 1: Table structure of course

![Records in course table](image)

Figure 2: Records in course table

![Java file location](image)

Figure 3: Java file location
Figure 4 shows a form named SearchForm.jsp that will be used by the user to search course using course id. When the user clicks the 'Search Please' button (actually submit button), the program will call SearchCourseServlet.jsp file using POST method. This file will communicate with several files like Course.java and CourseDB.java. The servlet file will find the searched course id in the database. Finally, the related information about searched data will be displayed as shown at Figure 5.

Thus, you have to complete the code on Figure 6, Figure 7, Figure 8, and Figure 9.

(30 marks)

Figure 6: Code of SearchForm.jsp

```html
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
</head>
<body>
<form>
    Course ID: TID3163
    Course Name: Software Engineering
    Credit Hour: 3
    Pre-requisite: TIA2023
    Search Please
</form>
</body>
</html>
```
package crs.controller;

import crs.business.Course;

import java.io.*;          // import CourseDB class
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.servlet.*;
import javax.servlet.http.*;

public class SearchCourseServlet extends HttpServlet {

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        // get value from parameter/ text field
        Course course = new Course();
        try {
            // call search method in CourseDB class
            course = ..............................................

            ..............................................    // create a session

            ..............................................    // set attributes
            // for course

            // forward to Result.jsp file using RequestDispatcher

            ..............................................

        } catch (SQLException ex) {
            Logger.getLogger(SearchCourseServlet.class.getName()).log(Level.SEVERE,
            null, ex);
        }
    }  
}

Figure 7: Code of SearchCourseServlet.java
package crs.data;
import crs.business.Course;
import java.sql.*;
public class CourseDB {
    public static Course search(String courseId) throws SQLException {
        DBConnection DBCon = new DBConnection();

        Connection conn = DBCon.getConnection();

        // create statement
        String query = ____________________________ // create a query for sql statement

        ____________________________ // execute the query

        ____________________________ // create course object

        while (_______________) // read data that stored in Result Set {
            // get the data for each column from database
            ____________________________
            ____________________________
            ____________________________
            ____________________________

            // call setter methods to keep data into course object
            ____________________________
            ____________________________
            ____________________________
            ____________________________
        }
        stat.close();

        return course; // return data
    }
}
<jsp:useBean></jsp:useBean>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <title>JSP Page</title>
</head>
<body>
  Course ID : ___________________________  <br>
  Course Name : _________________________  <br>
  Credit Hour : _________________________  <br>
  Pre-requisite : ________________________  <br>
</body>
</html>

Figure 9: Code of Result.jsp
package crs.business;
public class Course {
    private String courseId;
    private String courseName;
    private int creditHour;
    private String preReq;
    public Course() {
    }
    public Course(String courseId, String courseName, int creditHour, String preReq) {
        this.courseId = courseId;
        this.courseName = courseName;
        this.creditHour = creditHour;
        this.preReq = preReq;
    }
    public void setCourseId(String courseId) {
        this.courseId = courseId;
    }
    public void setCourseName(String courseName) {
        this.courseName = courseName;
    }
    public void setCreditHour(int creditHour) {
        this.creditHour = creditHour;
    }
    public void setPreReq(String preReq) {
        this.preReq = preReq;
    }
    public String getCourseId() {
        return courseId;
    }
    public String getCourseName() {
        return courseName;
    }
    public int getCreditHour() {
        return creditHour;
    }
    public String getPreReq() {
        return preReq;
    }
} // class Course

Figure 10: Code of Course.java
package crs.data;
import java.sql.*;
public class DBCConnection {
    private Connection connection = null;
    private String url = "jdbc:mysql://localhost/crs";
    private String userName = "root";
    private String pwd = "123";
    private String driver = "com.mysql.jdbc.Driver";

    public void openConnection() {
        try {
            Class.forName(driver);
            System.out.println("Driver is loaded successfully");
            connection = DriverManager.getConnection(url, userName, pwd);

        } catch (ClassNotFoundException e) {
            System.out.println("Database driver is not found");
        } catch (SQLException e) {
            System.out.println("Error opening database connection = " + e.getMessage());
        }
    }

    public void closeConnection() {
        try {
            connection.close();
            System.out.println("Database is closed successfully");
        } catch (SQLException e) {
            System.out.println("Error closing database = " + e.getMessage());
        }
    }

    public Connection getConnection()
    {
        return connection;
    }
}

Figure 11: Code of DBCConnection.java