**Department of Computer Science & Engineering**

Second Sessional Examination Mar 2018

Branch: **4th Semester**, **B. Tech. CSE** Course: **CSL225** Advanced Web Programming

Duration**: 1Hr** Max. Marks: **15**

Note: Attempt all questions. Credit reserved for neat comments/ to the point answers. Assume any missing data.



**Q1. Choose the appropriate option from the given choices. (2)**

1. The Maximum size of URL for Mozilla and Chrome are

|  |  |
| --- | --- |
| 1. <100 KB | 1. 100 KB |
| 1. **> 100 KB** | 1. 2 KB |

1. setContentType() method is called through \_\_\_\_\_\_\_\_\_\_ object.

|  |  |
| --- | --- |
| 1. Servlet object | 1. Request object |
| 1. **Response object** | 1. Session object |

1. setParameter() method is called through \_\_\_\_\_\_\_\_\_\_ object.

|  |  |
| --- | --- |
| 1. Session object | 1. Request object |
| 1. Response object | 1. All the above |
| 1. **None** |  |

1. Anonymous inner class definition for the object of abstract class can override only abstract methods.

|  |  |
| --- | --- |
| 1. true | 1. **false** |

**Q2.** The following database operation code uses “Statement” object “stmt”. Write an alternative code using “prepared statement” to produce the same result.

String sql="insert into person(name,age,gender) "+

"values("

+"'"+name+"'"+","+"'"+age+"'"+","+"'"+gender+"'"+")";

try{

stmt= con.createStatement();

stmt.executeUpdate(sql);

} catch (SQLException ex) {

ex.printStackTrace();

} **(2)**

**Solution:**

**String sql="insert into person(name,age,gender) "+**

**"values(?,?,?)";**

**try{**

**PreparedStatement ps=con.prepareStatement(sql);**

**ps.setString(1, name);**

**ps.setInt(2, age);**

**ps.setString(3, gender);**

**ps.executeUpdate();**

**}**

**catch (SQLException ex) {**

**}**

**Q3**. A website has following three jsp pages.

1. Home.jsp b) Products.jsp c) Purchase.jsp

The session is managed by the following code.

String username=(String) session.getAttribute(“username”);

int creditPoints= Integer.parseInt(session.getAttribute(“points”));

if(username.equals(“vnit”) && creditPoints >100)

out.println(“You are eligible”);

Write an alternative code using “url rewriting” approach to produce the same output. **(2)**

**Solution:**

**<%**

**String username=(String) request.getParameter(“username”);**

**int creditPoints= Integer.parseInt(request.getParameter(“points”));**

**%>**

**<%**

**if(username.equals(“vnit”) && creditPoints >100)**

**{**

**out.print("<h2><a href=\"Home.jsp?username="+"username"+**

**"& creditPoints="+"creditPoints"+">Home</a>|"**

**+ "<a href=\"Products.jsp?username="+"username"+**

**"& creditPoints="+"creditPoints"+">Products</a> |"+**

**"<a href=\"Purchase.jsp?username="+"username"+**

**"& creditPoints="+"creditPoints"+">Purchase</a></h2>");**

**out.print("<br><br><h3>You are eligible</h3>");**

**}**

**Q4**. Complete the following partially completed code to display “Application name” using Context parameters and to display “Servlet name” using config parameters.

Web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app version="3.1">

------

------

</web-app>

HomeServlet.java

------

------

.getInitParameter(“application\_name”);

. getInitParameter(“servlet\_name”);

------ **(3)**

**Solution:**

**Web.xml**

**<?xml version="1.0" encoding="UTF-8"?>**

**<web-app version="3.1">**

**<context-param>**

**<param-name>application\_name</param-name>**

**<param-value>SampleApp</param-value>**

**</context-param>**

**<servlet>**

**<servlet-name>HomeServlet</servlet-name>**

**<servlet-class>sample.HomeServlet</servlet-class>**

**<init-param>**

**<param-name>servlet\_name</param-name>**

**<param-value>Home</param-value>**

**</init-param>**

**</servlet>**

**</web-app>**

**HomeServlet.java**

**ServletContext context= request.getServletContext();**

**ServletConfig config= request.getServletConfig();**

**String appname=(String)context.getInitParameter(“application\_name”);**

**String servletName=(String)config.getInitParameter(“servlet\_name”);**

**out.print(“<h3>Application Name is”+ appname+ “</h3><br><br>”);**

**out.print(“<h3>Servlet Name is”+ servletName+ “</h3>”);**

**Q5.** There are two main operations in the banking software which are “read” and “update”. A read operation reads a “balance” and update operation updates the “balance” by adding user entered amount into the existing “balance”. Two user threads try to access the same balance simultaneously to perform these operations. The possible operations in this scenario are RR, RU, UR, UU, where “R” means “read” and “U” means “update”. Assume suitable data and scan the necessary input from the user. Write a multithreaded java code such that data consistency should be maintained. Initial amount in the account is 1000 Rs.

**Solution:**

**class Account**

**{**

**static int bal=1000;**

**public static int read(){**

**return bal;**

**}**

**public static void update(int num){**

**bal= bal+num;**

**}**

**}**

**class MyThread extends Thread{**

**public MyThread(){**

**start();**

**}**

**boolean wflag=false;**

**boolean rflag=false;**

**boolean waitflag=false;**

**public void run(){**

**while(true){**

**Scanner sc= new Scanner(System.in);**

**System.out.println("Enter your choice");**

**System.out.println("1.Read Balance\t 2. Add Amount");**

**int choice= sc.nextInt();**

**switch(choice){**

**case 1: if(wflag==true){**

**waitflag=true;**

**try{**

**waitflag=true;**

**wait();**

**}**

**catch(Exception e){**

**e.printStackTrace();**

**}**

**}**

**rflag=true;**

**int balance= Account.read();**

**if(waitflag==true)**

**notify();**

**rflag=false;**

**System.out.println("Balance="+balance); break;**

**case 2: System.out.println("Enter deposit amount");**

**int amt=sc.nextInt();**

**if(rflag==true){**

**try{**

**waitflag=true;**

**wait();**

**}**

**catch(Exception e){**

**e.printStackTrace();**

**}**

**}**

**else if(wflag==true){**

**try{**

**waitflag=true;**

**wait();**

**}**

**catch(Exception e){**

**e.printStackTrace();**

**}**

**}**

**wflag=true;**

**Account.update(amt);**

**wflag=false;**

**if(waitflag==true)**

**notify();**

**break;**

**}**

**default: System.out.println("Wrong Input");**

**}**

**}**

**}**

**public static void main(String args[]){**

**MyThread m1= new MyThread();**

**MyThread m2= new MyThread();**

**}**

**}**

**(3)**

**Q6.**A website is facing the trouble in maintaining user log as this facility is not in existing structure. Now the website wants to add new Servlet “MyLog.java” which will store “username” and current “date” into the log file “userlog.txt”. But the website doesn’t want to modify the existing structure of calling/switching from one resource to another one and so on. Suggest a troubleshooting code which will add a new resource file “MyLog.java” into the project without changing the existing calling structure. The new resource file should add login details into the log file. Modify the already available servlet “Login.java” with necessary code to troubleshoot the problem. Also write down the complete code of new servlet “MyLog.java”. **(3)**

**Solution:**

**Login.java**

**----------**

**----------**

**String username= (String)request.getParameter("username");**

**----------**

**----------**

**RequestDispatcher rd= request.getRequestDispatcher("MyLog");**

**rd.include(request, response);**

**----------**

**----------**

**MyLog.java**

**----------**

**----------**

**String username= (String)request.getParameter("username");**

**Date date= new java.util.Date();**

**String mydate= date.toString();**

**File f= new File("log.txt");**

**FileWriter fWriter = new FileWriter(f);**

**fWriter.write(username+"\n");**

**fWriter.write(mydate+"\n");**

**fWriter.flush();**

**fWriter.close();**

**----------**

**----------**