FACULTY OF COMPUTER APPLICATIONS

Advanced Java Practical List

Practical List

1.	Write a Servlet to display "Hello World" on browser.
2.	Write a Servlet to display all the headers available from request.
3.	Write a Servlet to display parameters available on request
4.	Write a Servlet to display all the attributes available from request and context
5.	Write a Servlet which displays a message and also displays how many times the message has been displayed (how many times the page has been visited).
6.	Assume that the information regarding the marks for all the subjects of a student in the last exam are available in a database, Develop a Servlet which takes the enrollment number of a student as a request parameter and displays the marksheet for the student.
7.	Develop a Servlet which looks for cookies for username and password, and forwards to a home.jsp in case the cookies are valid and forwards to login.jsp, in case the cookies are not found or the cookies are not valid.
8.	Develop a Servlet to authenticate a user, where the loginid and password are available as request parameters. In case the authentication is successful, it should setup a new session and store the user's information in the session before forwarding to home.jsp, which displays the user's information like full name, address, etc.
9.	Write a simple JSP page to display a simple message (It may be a simple html page).
10.	Write a JSP page, which uses the include directive to show its header and footer.
11.	Create a Java class called Product with the following properties: name, description, price. Create a listener that notifies (through System.out) whenever a user adds a product to a shopping cart (i.e. adds an object to the session object) or removes it again. Hint: check out the class HttpSessionAttributeListener. Make it print the name and price of the object (hint: access the session through the HttpBindingEvent object). Also, let the listener print the total price of all objects saved in the session so far (one way to accomplish this could be to keep a collection of all objects saved to the session – or just their keys – in the listener or an associated class).
12.	Create a servlet filter that logs all access to and from servlets in an application and prints the following to System.out: **a.** the time the request was received **b.** the time the response was sent **c.** how much time it took to process the request **d.** the URL of the resource requested **e.** the IP address of the visitor
13.	Develop a interest calculation application in which user will provide all

	information in HTML form and that will be processed by servlet and response will be generated back to the user.
14.	Develop an application to demonstrate how the client (browser) can remember the last time it visited a page and displays the duration of time since its last visit. (Hint: use Cookie)
15.	Develop an application to keep track of one user across several servlet invocations within the same browser session.
16.	Develop an application to write a "page-composite" JSP that includes other pages or passes control to another page. (Hint: Use <jsp:include> or <jsp:forward>).
17.	You want to reduce the amount of Java coding in your JSP using a JavaBean component. (Hint: Use <jsp:useBean> with the name of your bean).
18.	Develop a program to perform the database driven operation like insert, Delete, Update and select. To perform the above operations create one table named Employee. Field Name Field Type **a.** EmpId Integer **b.** Empname Varchar **c.** Emp_desig Varchar **d.** Emp_J_Date Varchar **e.** Emp_Salary Numeric
19.	Develop a Java application to perform the database driven operation like insert, Delete, Update and selection using PreparedStatement. To perform the above operations use the table from above exercise.
20.	Write a Java application to invoke a stored procedure using a CallableStatement. For this a stored procedure called incrementSalary may be developed to increase all the employees salary by a percentage specified in the parameter.
21.	Develop an RMI application which accepts a string or a number from client and server checks that this string or number is palindrome or not.
22.	Develop RMI application in which client sends operator and operands for computation and server sends back the result. It behaves just like a distributed calculator app.
23.	Develop a RMI distributed application where client module will send employee id to server module, server will then send employee detail to client and then client will display all these information.
24.	Develop a RMI application which accepts a sentence from client. Server module calculates number of words present within this sentence, number of vowels present within this sentence and output will be displayed within client.