**浙江工业大学 2022/2023 学年**

**第 二 学期试卷**

**课程 班级\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**姓名\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_学号\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 教师姓名\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **题序** | **一** | **二** | **三** | **四** | **总评** |
| **计分** |  |  |  |  |  |

|  |
| --- |
| **一、选择题（共10小题，每小题2分，共20分）**  1、如果Tomcat安装后，想要修改它的端口号，可以通过修改<tomcat安装目录>/conf下的（ ）文件来实现。  A. web.xml B. server.xml  C. config.xml D. tomcat-user.xml  2、某Web应用工程部署在tomcat中，如果某个Servlet需要使用一个名为com.demo.StringUtil的工具类，则StringUtil.class类文件应该存放在哪个目录中？（ ）  A. WEB-INF/lib B. WEB-INF/lib/com/demo  C. WEB-INF/classes D. WEB-INF/classes/com/demo  3、假设在一个Web应用工程myweb的部署描述文件中有下面的定义：  <servlet-mapping>  <servlet-name>HandleDataServlet</servlet-name>  <url-pattern>\*.do </url-pattern>  </servlet-mapping>  下面哪个请求将不会由HandleDataServlet服务？（ ）  A. /myweb/\*.do B. /myweb/do  C. /myweb/handle.do D. /myweb/test.do  4、JavaBean的生命周期中，以下哪个是用来跟踪用户的会话的？（ ）  A. session B. request C. page D. application  5、以下关于JavaBean的好处，说法错误的是（ ）。  A.常用JavaBean来存放数据、封装业务逻辑等  B.JavaBean更明确地把Web页面的设计和软件业务逻辑的设计区分开来  C.JavaBean可以在多个应用程序中重用，方便程序维护  D.JavaBean可以实现安全性、事务行为、并发性和持久性  6、下列对于JSP脚本语法中的声明<%! declaration; %>的说法，错误的是（ ）。  A．JSP声明既可包含变量定义，又可包含方法定义及其内部类定义  B．一个声明仅在一个页面中有效  C．声明的变量将作为局部变量  D．声明的变量将在JSP页面初始化时被初始化  7、以下哪个不属于Servlet3.0规范的注解？（ ）  A．@WebServlet B. @WebFilter  C. @WebTag D. @WebListener  8、在Web应用中关于一个Servlet关联的过滤器数量，正确的说法是（ ）。  A．不可以为一个Servlet关联多个过滤器类  B．可以为多个Servlet关联同一个过滤器类  C．每个Servlet必须至少有一个过滤器类  D．每一个过滤器类必须关联到至少一个Servlet  9、在使用HttpSession维护当前在线人数的Web应用系统中，当一个客户离开Web应用系统时，将激发哪个方法？（ ）  A．ServletContextListener监听器的contextDestroyed( )方法  B．HttpSessionListener监听器的sessionDestroyed( )方法  C．ServletContextAttributeListener监听器的attributeRemoved( )方法  D．HttpSessionAttributeListener监听器的attributeRemoved( )方法  10、以下关于访问数据库说法错误的是（ ）。  A．加载JDBC驱动程序最常用的方法是使用Class类的forName()静态方法  B．驱动程序加载成功后可使用DriverManager类的getConnection()建立数据库连接对象，但使用这种方法的性能比数据源方式差  C．Connection、Statement和ResultSet等对象处理完成后，不用关闭，可留给下次访问数据库时使用  D．通常用PreparedStatement对象来执行动态SQL语句  **二、简答题 (共4小题，20分)**  1、请简述ServletConfig和ServletContext对象的区别。**（本题4分）**  ServletConfig对象可获取当前Servlet信息，也可获取初始化信息，每产生一个Servlet对象会产生一个ServletConfig对象，通过ServletConfig对象也可获取当前Web应用工程的ServletContext对象。  ServletContext对象可获取所有Servlet的Web应用设置，一个Web应用工程仅有一个ServletContext对象，可以获取整个Web应用工程的初始化信息。  2、当用户登录购物网站时，一般Web容器就为该用户创建一个HttpSession对象。请简述通过会话机制实现购物网站中购物车应用的原理。**（本题4分）**  **①实现购物车的Servlet使用该会话对象存储用户的购物车对象，购物车中存储着用户购买的商品列表。**  **②当客户向购物车中添加商品或删除商品时，Servlet就更新该列表。**  **③当客户要结账时，Servlet就从会话中检索购物车对象，从购物车中检索商品列表并计算总价格。**  **④一旦客户结算完成，容器就会关闭会话。**  3、请简述基于MVC设计模式实现添加客户信息的步骤，并画出各种程序之间关系的示意图。  （1）设计一个客户信息的JavaBeans对象类Customer，作为传输对象。  （2）定义一个类CustomerSave，在该类中定义一个添加客户信息的方法。  （3）编写一个添加客户信息的insertCustomer.jsp页面作为视图层，通过一个表单添加客户信息。  （4）编写一个控制器类InsertCustomerServlet，该类中使用JavaBeans对象传输对象，将insertCustomer.jsp页面提交的客户信息存入到数据库中。  程序之间的关系图如下：    4、在Web应用程序中，一个Servlet在某一时刻可能被多个用户同时访问。这时Web容器将为每个用户创建一个线程，如果Servlet涉及共享资源（比如线上买票、618或双11大促销抢购商品）的问题，需要保证Servlet是线程安全的。请简述如何编写线程安全的Servlet？  （1）用方法的局部变量保存请求中的专有数据。对方法中定义的局部变量，进入方法的每个线程都有自己的一份方法变量拷贝。如果要在不同的请求之间共享数据，应该使用会话来共享这类数据。  （2）只用Servlet的成员变量来存放那些不会改变的数据。有些数据在Servlet生命周期中不发生任何变化，通常是在初始化时确定的，这些数据可以使用成员变量保存。  （3）对可能被请求修改的成员变量同步（使用synchronized关键字）。  （4）如果Servlet访问外部资源，那么需要对这些资源进行同步处理。  **三、程序阅读题(共3小题，每小题10分，共30分)**  1、假设编译验证用户登录的Servlet（CheckLogin.java）生成的.class文件放在Tomcat主目录下的webapps/zjut/WEB-INF/classes/bank目录下，访问该Servlet的客户端请求URL为：http://localhost:8080/zjut/chkLogin.html。  若用户登录的页面login.html放在Tomcat主目录下的webapps/zjut/userLogin目录下，用户输入用户名和密码后由Servlet（CheckLogin.java）处理，请将以下划线部分补充完整。  1）以下是login.html的代码：  <html><body>  <form action=" (1) ../ chkLogin.html " method="post">  <table width="300" border="0" align="center" cellpading="2" cellspacing="1">  <tr>  <td align="right">用户名：</td>  <td align="left"><input type="text" name="username" size="15"></td>  </tr>  <tr>  <td align="right">密&nbsp;&nbsp;码：</td>  <td><input type="password" name="password" size="15"></td>  </tr>  <tr>  <td colspan="2" align="center"><input type="submit" value="登录">  &nbsp; <input type="reset" value="取消"></td>  </tr>  </form>  </body></html>  2）以下是CheckLogin.java的代码：  package bank;  import java.io.\*;  import javax.servlet.\*;  import javax.servlet.http.\*;  @webServlet("/ chkLogin.html")  public class CheckLogin extends HttpServlet {  public void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=gb2312");  PrintWriter out = (2) response.getWriter() ;  String username = (3) request.getParameter("userName") ;  String password =(4) request.getParameter("password") ;  //若用户名和密码均为admin，则登录成功，否则登录失败  if((5) username!=null&&password!=null&&username.equals("admin")&&password.equals("admin") )  out.println("登录成功");  }  else{  out.println("登录失败");  }  }  }  2、为测试<c:forEach>标签显示List对象的元素，设计了一个javabean类Book，含有isbn、title和price三个属性，编写BooksServlet创建一个List<Book>对象，然后将控制转发到books.jsp页面，在该页面中使用<c:forEach>标签显示每本书的信息。请补充以下books.jsp和BooksServlet.java划线部分的代码。  以下是books.jsp的代码：  <%@ page contentType="text/html;charset=UTF-8" %>  <%@ page import="com.model.Book" %>  <%@ taglib uri="http://java.sun.com/jsp/jstl/core" (6) prefix=*"c"* %>  <html>  <head><title>图书列表</title>  </head>  <body>  <table>  <tr>  <td>书号</td><td>书名</td><td>价格</td>  </tr>  <c:forEach var=*"*(7) book *"* items=*"*(8) ${requestScope.books} *"*>  <td>${book.isbn}</td>  <td>${book.title}</td>  <td>${book.price}</td>  </tr>  </c:forEach>  </table>  </body></html>  以下是BooksServlet.java的代码：  package com.demo;  import java.io.IOException;  import javax.servlet.\*;  import javax.servlet.http.\*;  import javax.servlet.annotation.WebServlet;  import java.util.List;  import java.util.ArrayList;  import com.model.Book;  @WebServlet("/bookShowServlet.do")  public class BooksServlet extends HttpServlet {  public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  List<Book> books =(9) new ArrayList<Book>() ;  Book book1 = new Book("978-7-302-23059-5","Java程序设计",45.00);  Book book2 = new Book("978-7-302-21540-0","Web并应用开发",39.00);  Book book3 = new Book("978-7-302-24130-1","C++程序设计",49.80);  books.add(book1);  books.add(book2);  books.add(book3);  (10) request.setAttribute("books", books) ;//将books对象在请求作用域中共享，属性名为books  RequestDispatcher rd = request.getRequestDispatcher("/books.jsp");  rd.forward(request, response);  }  }  3、在Web应用系统中经常有数据导入功能，为了支持大批量数据的导入，往往采用数据文件上传后异步处理数据入库。为模拟以上功能，编写了一个简单的模拟异步处理的Servlet类AsyncDemoServlet。该类中创建了一个AsyncContext对象，并通过该对象以异步的方式启动了一个模拟数据入库的线程importDataExecutor。请补充以下划线部分的代码，并回答。  1）以下是AsyncDemoServlet.java的代码：  package com.demo;  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.AsyncContext;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.\*;  @WebServlet(urlPatterns = "/asyncDemo.do", asyncSupported = true)  public class AsyncDemoServlet extends HttpServlet {  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=UTF-8");  PrintWriter out = response.getWriter();  out.println("<html><head><title>异步调用示例</title></head>");  out.println("<body>");  out.println("上传文件处理中..."+ "<br>");  out.flush();  AsyncContext actx =(11) request.startAsync() ;  (12) actx.setTimeout(30 \* 1000) ; // 设置异步调用的超时时长30秒  actx.start((13) new ImportDataExecutor(actx) ); // 启动异步调用的线程  out.println("文件已上传！"+"</br>");  out.flush();  }  }  2）以下是importDataExecutor.java的代码：  package com.demo;  import javax.servlet.\*;  import java.io.\*;  public class ImportDataExecutor implements Runnable {  private AsyncContext actx = null;  public ImportDataExecutor(AsyncContext actx) {  this.actx = actx;  }  public void run() {  try {  // 等待10秒钟，以模拟导入数据业务方法的执行  Thread.sleep(10000);  PrintWriter out = actx.getResponse().getWriter();  out.println("数据导入业务处理完毕！");  out.flush();  (14) actx.complete() ; // 结束异步线程  } catch (Exception e) {  e.printStackTrace();  }  }  }  问题（15）：测试运行AsyncDemoServlet.java后，在浏览器中输出的结果是什么？  上传文件处理中...  文件已上传！  数据导入业务处理完毕！  **四、程序设计题(共2小题，每题15分，共30分)**  1、为了常态化管控新冠肺炎疫情，部分地区要求市民每隔2天或3天做一次核酸检测。某社区为了及时了解社区居民的核酸检测情况，拟开发一个核酸检测情况统计系统，其中该系统有一个登记功能，要求登记姓名、身份证号、检测日期、检测结果。请你按如下要求编写实现核酸检测结果的登记功能：  （1）编写一个名为inputTest.jsp的JSP页面，该页面包括一个表单，包含姓名、身份证号、检测日期、检测结果（阴性或阳性的单选按钮）等输入框以及保存和取消按钮。  （2）编写一个名为com.controller.InputTestSevlet的Servlet（注解地址为/testResult.do），要求当用户在inputTest.jsp中输入信息后点击“保存”按钮，请求InputTestSevlet对其处理。在InputTestSevlet中获取人员姓名、身份证号、检测日期、检测结果后将其作为检测结果的javabean类TestResult属性存储在请求对象中，然后通过请求对象的getRequestDispatcher()获得RequestDispatcher()对象，将请求转发到showTestInfo.jsp。  （3）编写一个检测结果的Javabean，类名为com.model.TestResult，变量名分别为name、id、testDate、testResult。注：编写的代码可省略set和get方法。  （4）在showTestInfo.jsp中显示核酸检测情况登记的人员姓名、检测日期、检测结果。  **(1)输入页面（**inputTest**.jsp）参考代码：**  <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <title>输入页面</title>  </head>  <body>  <form action=*"testResult.do"* method=*"post"*>  姓 名：<input type=*"text"* name=*"name"* size=*"15"* /><br><br>  身份证号：<input type=*"text"* name=*"id"* size=*"15"*/><br><br>  检测日期：<input type=*"text"* name=*"testDate"* size=*"15"*/><br><br>  检测结果：<input type=*"radio"* name=*"testResult"* value=*"阴性"*/>阴性  <input type=*"radio"* name=*"testResult"* value=*"阳性"*/>阳性  <br><br>  <input type=*"submit"* value=*"保存"* />  <input type=*"reset"* value=*"取消"* />  </form>  </body>  </html>  (2) com.controller.InputTestServlet.java参考代码：  package com.controller;  **import** java.io.IOException;  **import** javax.servlet.RequestDispatcher;  **import** javax.servlet.ServletException;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  @WebServlet("/testResult.do")  **public** **class** InputTestServlet **extends** HttpServlet {  **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  request.setCharacterEncoding("UTF-8");  TestResult t=**new** TestResult();  t.setName(request.getParameter("name"));  t.setId(request.getParameter("id"));  t.setTestDate(request.getParameter("testDate"));  t.setTestResult(request.getParameter("testResult"));  request.setAttribute("testResult", t);  request.getRequestDispatcher("/showTestInfo.jsp").forward(request, response);  }  }  **(3)** TestResult**.java参考代码：**  package com.model;  public class TestResult {  private String name;  private String id;  private String testDate;  private String testResult;  public TestResult() {  }    public TestResult(String name, String id, String testDate, String testResult) {  super();  this.name = name;  this.id = id;  this.testDate = testDate;  this.testResult = testResult;  }  //省略set和get方法  }  **(4) showTestInfo.jsp参考代码：**  <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>核酸检测结果信息</title>  </head>  <body>  姓名：${testResult.name} <br>  身份证号：${testResult.id} <br>  检测日期：${testResult.testDate} <br>  检测结果：${testResult.testResult} <br>  </body>  </html>  2、使用JSTL的<c:forEach>标签对Map对象迭代。该题目包括BigCitiesServlet  类和 bigCities.jsp 页面。在 BigCitiesServlet 类中创建一个 Map<String,String>对象 capitals，键为国家名称，值为首都名称，添加几个对象。另外创建一个 Map<String,String[]>对象 bigCities，键为国家名称，值为 String 数组包含该国家的几个大城市。在 doGet()中使用 RequestDispatcher 对象将请求转发到 bigCities.jsp 页面。    图1 bigCities.jsp页面  **(1)** BigCitiesServlet.java**参考代码：**  package com.demo;  // 此处省略若干 import 语句  @WebServlet(urlPatterns = {"/bigCities"})  public class BigCitiesServlet extends HttpServlet {  private static final int serialVersionUID = 112233;  @Override  public void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  Map<String, String> capitals = new HashMap<String, String>();  capitals.put("俄罗斯", "莫斯科");  capitals.put("日本", "东京");  capitals.put("中国", "北京");  request.setAttribute("capitals", capitals);    Map<String, String[]> bigCities = new HashMap<String, String[]>();  bigCities.put("澳大利亚", new String[] {"悉尼", "墨尔本", "布里斯班"});  bigCities.put("美国", new String[] {"纽约", "洛杉矶", "加利福尼亚"});  bigCities.put("中国", new String[] {"北京", "上海", "深圳"});    request.setAttribute("capitals", capitals);  request.setAttribute("bigCities", bigCities);  RequestDispatcher rd = request.getRequestDispatcher("/bigCities.jsp");  rd.forward(request, response);  }  }  **(2)** bigCities.jsp**参考代码：**  **<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>**  **<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>**  **<html>**  **<head><title>Big Cities</title></head>**  **<body>**  **<table>**  **<tr style="background:#448755;color:white;font-weight:bold">**  **<td>国家</td>**  **<td>首都</td>**  **</tr>**  **<c:forEach items="${requestScope.capitals}" var="mapItem">**  **<tr>**  **<td>${mapItem.key}</td>**  **<td>${mapItem.value}</td>**  **</tr>**  **</c:forEach>**  **</table>**  **<br/>**  **<table>**  **<tr style="background:#448755;color:white;font-weight:bold">**  **<td>国家</td>**  **<td>城市</td>**  **</tr>**  **<c:forEach items="${requestScope.bigCities}" var="mapItem">**  **<tr>**  **<td>${mapItem.key}</td>**  **<td>**  **<c:forEach items="${mapItem.value}" var="city" varStatus="status">**  **${city}**  **<c:if test="${!status.last}">,</c:if>**  **</c:forEach>**  **</td>**  **</tr>**  **</c:forEach>**  **</table>**  **</body>**  **</html>** |