CCEE Mock - I | WJP

Total points 22/40

The respondent's email (**amolgavit158121@gmail.com**) was recorded on submission of this form.

0 of 0 points

Name *

Amol Gavit

PRN (12 Digits) *

250240320013

Centre * Dropdown

Kharghar

Questions 22 of 40 points

Concept of IoC Containers in Spring
Inversion of Control (IoC) is a design principle where the framework takes control of object creation and management instead of the programmer explicitly instantiating objects. Spring provides two main IoC containers:

BeanFactory – This is the simplest container in Spring, responsible for basic dependency injection. It is lightweight and suitable for applications where memory consumption is a concern.

ApplicationContext – This is a more advanced container that extends BeanFactory. It provides additional functionalities such as event propagation, declarative bean creation, internationalization support, and automatic bean configuration.

✓ Which are the IoC containers in Spring? *

- (●) BeanFactory, ApplicationContext
- ( ) BeanFactory, ApplicationContext, IocContextFactory
- ( ) BeanFactory, BeanContext, IocContextFactory
- ( ) BeanFactory, ApplicationContext, BeanContext

BeanFactory, ApplicationContext, IocContextFactory

There is no standard IocContextFactory in Spring. The correct answer only includes BeanFactory and ApplicationContext.

BeanFactory, BeanContext, IocContextFactory

BeanContext is not an IoC container in Spring.

IocContextFactory is not a standard Spring component.

BeanFactory, ApplicationContext, BeanContext

BeanContext is not recognized as an IoC container in Spring. The correct containers are only BeanFactory and ApplicationContext.

✓ Which interface in spring is responsible for Instantiating and managing the so called Spring beans? *1/1

- (●) BeanFactory ✓
- ( ) ApplicationContext
- ( ) BeanDefinition
- ( ) BeanFactoryAware

Explanation
In Spring, the BeanFactory interface is responsible for instantiating and managing Spring beans. It is the core container that provides basic dependency injection capabilities.

ApplicationContext – While it extends BeanFactory and provides additional features like event propagation and internationalization, the primary interface responsible for bean instantiation and management is BeanFactory.

BeanDefinition – This is a metadata representation of a bean, defining its properties and dependencies, but it does not instantiate or manage beans directly.

BeanFactoryAware – This is an interface that allows a bean to access the BeanFactory that created it, but it does not manage beans itself.

✕ < %@ page language="java" % > *

The Following are the contents of test.jsp

1 < html >

2 < head >< title >A Comment Test< /title >< /head >

3 < body >

4 < h2 >A Test of Comments< /h2 >

5 < !-- This is Html Hidden Comment -- >

6 < %-- This is JSP Hidden Comment --% >

7 < /body >

8 < /html >

On executing test.jsp guess the correct output

- (●) Line 5 is inserted in response but Line 6 is not inserted in response
- ( ) Line 5 and Line 6 are inserted in response
- ( ) Both Line 5 and Line 6 are not inserted in response
- ( ) Line 5 is not inserted in response but Line 5 is inserted in response

Correct answer

- (●) Both Line 5 and Line 6 are not inserted in response

In JSP, there are two types of comments:

HTML Comments (<!-- ... -->) – These comments are visible in the browser's page source because they are part of the HTML output.

JSP Comments (<%-- ... --%>) – These comments are hidden from the client and are not included in the final response. They are processed at the server level and ignored during page compilation.

"Line 5 is inserted in response but Line 6 is not inserted in response"

Incorrect because HTML comments (<!-- ... -->) are not displayed in the rendered page but are visible in the page source.

JSP comments (<%-- ... --%>) are completely removed from the response.

"Line 5 and Line 6 are inserted in response"

Incorrect because JSP comments are never included in the response.

"Line 5 is not inserted in response but Line 5 is inserted in response"

This option is logically incorrect.

✓ Is The Session threadsafe in Hibernate? * 1/1

- True
- (●) False
- Not sure
- Bhagwaan jane

Explanation
In Hibernate, Session is not thread-safe. Each thread should obtain its own instance of a Session from the SessionFactory. Sharing a single Session instance across multiple threads can lead to concurrency issues, such as inconsistent data updates and transaction conflicts

Why Other Answers Are Incorrect
True – Incorrect because Hibernate does not guarantee thread safety for Session objects. Instead, each thread should manage its own Session instance.

Not sure – While uncertainty is understandable, the correct answer is definitively False, as Hibernate documentation and best practices emphasize that Session should not be shared across threads

✓ _____objects contains a call to the stored procedure *

- (●) Callable Statement
- Prepared Statement
- Statement
- Dynamic Statement

Explanation
In Java's JDBC API, a CallableStatement is specifically designed to call stored procedures in a database. It allows interaction with stored procedures that may have input, output, or both types of parameters.

Why Other Answers Are Incorrect
Prepared Statement – This is used for executing parameterized SQL queries efficiently, but it does not directly call stored procedures.

Statement – This is used for executing simple SQL queries without parameters, making it unsuitable for stored procedures.

Dynamic Statement – This is not a standard JDBC object for calling stored procedures.

✓ Select statements, which are Invalid for Cookie *

- ○ Cookies are transferred to HTTP Header
- ◉ Cookie is name and Object pair
- ○ Cookies can be disabled on client browser
- ○ Cookies should be used to give better client interaction but should not be used where business logic is dependent on them

Explanation
Cookies in HTTP are name-value pairs, not name-object pairs. A cookie consists of a name and a string value, along with optional attributes like expiration time, domain, and security settings.

Why Other Answers Are Correct
Cookies are transferred to HTTP Header – Correct, because cookies are sent via the Set-Cookie header in HTTP responses.

Cookies can be disabled on the client browser – Correct, as users can configure their browser settings to block or disable cookies.

Cookies should be used to give better client interaction but should not be used where business logic is dependent on them – Correct, because cookies are meant for session tracking and personalization, but relying on them for critical business logic can lead to security and consistency issues.

✓ What is default scope of bean in Spring framework? *

- ◉ singleton
- ○ prototype
- ○ request
- ○ session

Explanation
In the Spring framework, the default scope of a bean is singleton. This means that only one instance of the bean is created per Spring IoC container, and all requests for that bean return the same instance.

Why Other Answers Are Incorrect
Prototype – This scope creates a new instance of the bean every time it is requested. It is not the default scope.

Request – This scope creates a new bean instance for each HTTP request, but it is only valid in a web-aware Spring ApplicationContext.

Session – This scope creates a new bean instance for each HTTP session, but it is also only valid in a web-aware Spring ApplicationContext.

Explanation
In Hibernate, SessionFactory can manage contextual sessions, meaning it provides a way to retrieve the current session associated with the ongoing transaction. The method used for this is:

getCurrentSession() – This method retrieves the current session that is bound to the transaction context. It ensures that the same session is used throughout the transaction.

✓ org.hibernate.SessionFactory can manage contextual sessions and allows to retrieve them by:- *1/1

- ○ getSession() method
- ○ getCurrent() method
- ◉ getCurrentSession() method ✓
- ○ None of the above

Why Other Answers Are Incorrect
getSession() method – This method does not retrieve a contextual session. Instead, it creates a new session, which is independent of any existing transaction.

getCurrent() method – This is not a valid method in Hibernate's SessionFactory.

None of the above – Incorrect because getCurrentSession() is the correct method for retrieving contextual sessions.

In JSP, when using <jsp:useBean> with a request scope, the bean is stored as an attribute in the request object. There are multiple ways to retrieve this bean:

${requestScope.a1} – This uses Expression Language (EL) to access the bean stored in the request scope.

<%=request.getAttribute("a1")%> – This uses JSP scriptlets to retrieve the bean from the request object.

${a1} – This is a shorthand EL expression that works when the bean is available in the default scope.

Since all these methods correctly retrieve the bean, the correct answer is all the above.

✓ Given * 1/1
<jsp:useBean id=”a1” scope=”request” class=”mypack.Customer”/>
What is the syntax to read Customer object ?

- ○ ${requestScope.a1}
- ○ <%=request.getAttribute(“a1”)%>
- ○ ${a1}
- ◉ all the above ✓

Explanation
Hibernate distinguishes between transient (newly instantiated) and detached objects using multiple strategies:

Version Property – If an entity has a version property, Hibernate uses it to determine whether the object is transient or detached.

Identifier Value – If an object has no identifier value, it is considered transient. However, this method works only for Hibernate-managed surrogate keys and does not apply to natural keys or manually assigned surrogate keys.

Interceptor.isUnsaved() – Developers can implement their own strategy using Hibernate's Interceptor.isUnsaved() method to determine whether an object is transient.

✗ How does Hibernate distinguish between transient * (i.e. newly instantiated) and detached objects? Choose correct answer from following? 0/1

- ○ Hibernate uses the "version" property, if there
- ◉ If not uses the identifier value. No identifier value means a new object. This does work only for Hibernate managed surrogate keys. Does not work for natural keys and assigned(i.e. not managed by Hibernate) surrogate keys) ✗
- ○ Write your own strategy with Interceptor.isUnsaved().
- ○ All of the above

Correct answer

- ◉ All of the above

Why Other Answers Are Incorrect
Each of the individual strategies contributes to Hibernate's ability to distinguish between transient and detached objects. Since Hibernate can use all three approaches, the correct answer is All of the above.

✗ Which of the following correctly represents the following JSP statement. * 0/1

< %=x% >

- (●) < jsp:expression=x/ > ✗
- ( ) < jsp:declaration >x< /jsp:declaration >
- ( ) < jsp:statement >x< /jsp:statement >
- ( ) < jsp:expression >x< /jsp:expression >

Correct answer

- (●) < jsp:expression >x< /jsp:expression >

Explanation
In JSP, the expression tag (<%= x %>) is used to evaluate a Java expression and insert its result directly into the output. The equivalent XML-based syntax for this is <jsp:expression>x</jsp:expression>.

Why Other Answers Are Incorrect
<jsp:expression=x/> – Incorrect because this syntax is invalid in JSP.

<jsp:declaration>x</jsp:declaration> – Incorrect because declaration tags (<%! ... %>) are used to declare variables and methods, not to evaluate expressions.

<jsp:statement>x</jsp:statement> – Incorrect because statement tags do not exist in JSP.

✓ the HttpServlet class is extended from * 1/1

- (●) GenericServlet class ✓
- ( ) Servlet class
- ( ) ServletConfig Interface
- ( ) None of the above

Explanation
The HttpServlet class in Java is an abstract class that extends GenericServlet and implements the Servlet and Serializable interfaces. It provides methods for handling HTTP requests, such as doGet(), doPost(), doPut(), and doDelete()

Why Other Answers Are Incorrect
Servlet class – Incorrect because HttpServlet does not directly extend Servlet; instead, it extends GenericServlet, which is an abstract class that provides basic servlet functionality.

ServletConfig Interface – Incorrect because ServletConfig is an interface used to provide configuration information to a servlet, but it is not a superclass of HttpServlet.

None of the above – Incorrect because HttpServlet does extend GenericServlet.

Explanation
In JSP, the application scope allows attributes to be shared across all users and requests within the web application. The correct way to retrieve the value of an attribute stored in the application scope is:

application.getAttribute("hitcount") – This retrieves the stored value from the application scope.

pageContext.getServletContext().getAttribute("hitcount") – This is another valid way to access the attribute, using the pageContext object to get the ServletContext.

✕ Which of the following can be inserted into this JSP * page to output the current value to the response?
<% application.setAttribute("hitcount",new Integer(10)); %>
//insert here

0/1

- ○ <%=hitcount %>
- ◉ <%=application.getAttribute("hitcount"); %> ✕
- ○ <%=application.hitcount%>
- ○ <%=pageContext.getservletContext().getAttribute("hitcount") %>

Correct answer

- ◉ <%=pageContext.getservletContext().getAttribute("hitcount") %>

Why Other Answers Are Incorrect
<%=hitcount %> – Incorrect because hitcount is not a predefined variable in JSP. It must be retrieved using application.getAttribute().

<%=application.hitcount%> – Incorrect because application does not have a direct property named hitcount. Instead, attributes must be accessed using getAttribute().

✓ How many types of IOC (dependency injection) in Spring? * 1/1

- A.Setter Injection
- B.Constructor Injection
- C.Interface Injection
- D.All of the above
- E.Only A & B ✓
- F.Only A

Explanation
Spring supports two primary types of Dependency Injection (DI):

Setter Injection – Dependencies are injected via setter methods.

Constructor Injection – Dependencies are injected via constructors.

Why Other Answers Are Incorrect
Interface Injection – Spring does not support Interface Injection natively. This approach is used in some frameworks but not in Spring.

✗ REST service end point comprises an address. * 0/1

- starts with http:// and ends with ?
- starts with http:// and ends with &
- no certain URL is specified ✗
- depends upon the platform used

Correct answer

- starts with http:// and ends with ?

Explanation
A REST service endpoint is a URL that allows clients to interact with a RESTful web service. The structure of a REST API endpoint typically follows a standardized format:

It starts with http:// or https://, indicating the protocol used for communication.

It ends with ?, which signifies the beginning of a query string that contains parameters for filtering or modifying the request.

Why Other Answers Are Incorrect
"Starts with http:// and ends with &" – Incorrect because & is used within a query string to separate multiple parameters, but it does not mark the end of an endpoint.

"No certain URL is specified" – Incorrect because REST APIs follow a structured URL format to ensure consistency and accessibility.

"Depends upon the platform used" – While different platforms may have variations in implementation, the general structure of REST endpoints remains consistent across web services.

✓ Which of the following lines would initialize the out variable for sending a Microsoft word file to the browser *1/1

- ○ PrintWriter out=response.getServletOutput()
- ○ PrintWriter out=response.getPrintWriter()
- ○ PrintWriter out=response.getWriter()
- ◉ OutputStream o=response.getOutputStream() ✓

Explanation
When sending a Microsoft Word file to the browser, you need to use an OutputStream, not a PrintWriter. The response.getOutputStream() method provides a binary stream, which is required for sending files like Word documents, PDFs, and images.

Why Other Answers Are Incorrect
PrintWriter out=response.getServletOutput() – Incorrect because getServletOutput() is not a valid method in the HttpServletResponse class.

PrintWriter out=response.getPrintWriter() – Incorrect because PrintWriter is used for text-based responses, not binary file transfers.

PrintWriter out=response.getWriter() – Incorrect because getWriter() returns a character stream, which is unsuitable for sending binary files like Word documents.

✗ What is singleton scope? * 0/1

- ○ This scopes the bean definition to a single instance per Spring IoC container.
- ○ This scopes the bean definition to a single instance per HTTP Request.
- ○ This scopes the bean definition to a single instance per HTTP Session.
- ◉ This scopes the bean definition to a single instance per HTTP Application/ Global session. ✗

Correct answer

- ◉ This scopes the bean definition to a single instance per Spring IoC container.

Explanation
In Spring, singleton scope means that only one instance of a bean is created per Spring IoC container, and all requests for that bean return the same instance. This is the default scope in Spring.

Why Other Answers Are Incorrect
"This scopes the bean definition to a single instance per HTTP Request."

Incorrect because request scope creates a new bean instance for each HTTP request. This is only valid in a web-aware Spring ApplicationContext.

"This scopes the bean definition to a single instance per HTTP Session."

Incorrect because session scope creates a new bean instance for each HTTP session, meaning different users will have separate instances.

"This scopes the bean definition to a single instance per HTTP Application/Global session."

Incorrect because global-session scope creates a bean instance for the entire application, but it is only valid in a web-aware Spring ApplicationContext.

✓ What is the purpose of the @RestController annotation in Spring? * 1/1

- ( ) It is used to define a controller that handles requests and generates views in a Spring MVC application.
- ( ) It is used to enable the automatic creation of beans through classpath scanning in a Spring application.
- ( ) It is used to indicate that a class is a Spring bean and should be managed by the Spring container.
- (•) It is used to define a controller that handles requests and returns data directly as the response body in a RESTful Spring application. ✓

Explanation
The @RestController annotation in Spring is a specialized version of @Controller that simplifies the creation of RESTful web services. It combines @Controller and @ResponseBody, meaning that methods inside a class annotated with @RestController automatically return data as JSON or XML instead of rendering a view.

Why Other Answers Are Incorrect
"It is used to define a controller that handles requests and generates views in a Spring MVC application."

Incorrect because @Controller is used for MVC applications where responses are typically HTML pages, not RESTful data.

"It is used to enable the automatic creation of beans through classpath scanning in a Spring application."

Incorrect because @ComponentScan or @Component is responsible for classpath scanning, not @RestController.

"It is used to indicate that a class is a Spring bean and should be managed by the Spring container."

Incorrect because @Component, @Service, and @Repository are used for marking Spring-managed beans, while @RestController is specifically for RESTful controllers.

✓ The init( ) method takes _____ object as a parameter. * 1/1

- ( ) HttpServleRequest
- ( ) HttpServleResponse
- (•) ServletConfig ✓
- ( ) ServletResponse

Explanation
The init() method in a servlet is called by the servlet container when the servlet is initialized. It takes a ServletConfig object as a parameter, which provides configuration information for the servlet.

Why Other Answers Are Incorrect
HttpServletRequest – Incorrect because this object represents an HTTP request and is used in methods like doGet() and doPost(), not init().

HttpServletResponse – Incorrect because this object represents an HTTP response and is used to send data back to the client, not for servlet initialization.

ServletResponse – Incorrect because this object is used for sending responses, similar to HttpServletResponse, but is not passed to init().

✓ What does the @ExceptionHandler annotation in Spring allow you to do? * 1/1

- It is used to define a method that handles exceptions globally for all controllers in the Spring application.
- It is used to enable the automatic handling of exceptions by the Spring framework.
- (●) It is used to define a method that handles a specific type of exception thrown by a controller method. ✓
- It is used to specify the order of exception handling in the Spring application context.

Explanation
The @ExceptionHandler annotation in Spring allows you to define methods that handle specific exceptions thrown by controller methods. When an exception occurs, Spring automatically invokes the corresponding @ExceptionHandler method to process the error and return an appropriate response.

Why Other Answers Are Incorrect
"It is used to define a method that handles exceptions globally for all controllers in the Spring application."

Incorrect because global exception handling is done using @ControllerAdvice, not @ExceptionHandler.

"It is used to enable the automatic handling of exceptions by the Spring framework."

Incorrect because Spring does not automatically handle exceptions unless explicitly defined using @ExceptionHandler or @ControllerAdvice.

"It is used to specify the order of exception handling in the Spring application context."

Incorrect because @ExceptionHandler does not control the order of exception handling; it simply maps specific exceptions to handler methods.

✓ In a Maven pom.xml file, the <pluginManagement> section is used for: *

- Declaring the plugin dependencies required for the build process.
- (●) Defining configuration and execution details for plugins that should be inherited by child modules. ✓
- Specifying the repositories from which Maven should download plugins.
- Enabling the plugins for the build process.

Explanation
In Maven, the <pluginManagement> section is used to define plugin configurations that can be inherited by child modules in a multi-module project. It allows developers to centralize plugin settings in a parent POM, ensuring consistency across all modules.

Why Other Answers Are Incorrect
"Declaring the plugin dependencies required for the build process."

Incorrect because plugin dependencies are declared inside the <plugins> section, not <pluginManagement>.

"Specifying the repositories from which Maven should download plugins."

Incorrect because repositories for downloading plugins are defined in the <pluginRepositories> section, not <pluginManagement>.

"Enabling the plugins for the build process."

Incorrect because <pluginManagement> does not enable plugins directly. Instead, it provides configurations that child modules can inherit.

✕ Choose the statement that best describes the relationship between JSP and servlets *0/1

- (●) Servlets are built on JSP semantics and all servlets are compiled to JSP pages for runtime usage. ✕
- ( ) JSP and servlets are unrelated technologies.
- ( ) Servlets and JSP are competing technologies for handling web requests. Servlets are being superseded by JSP, which is preferred. The two technologies are not useful in combination.
- ( ) JSPs are built on servlet semantics and all JSPs are compiled to servlets for runtime usage.

Correct answer

- (●) JSPs are built on servlet semantics and all JSPs are compiled to servlets for runtime usage.

Explanation
JavaServer Pages (JSP) and Servlets are closely related technologies in Java web development. JSPs are essentially compiled into servlets at runtime by the servlet container. This means that JSPs rely on servlet semantics, and their execution follows the same lifecycle as servlets.

Why Other Answers Are Incorrect
"Servlets are built on JSP semantics and all servlets are compiled to JSP pages for runtime usage."

Incorrect because servlets are not compiled into JSPs. Instead, JSPs are converted into servlets.

"JSP and servlets are unrelated technologies."

Incorrect because JSPs and servlets are closely related. JSPs simplify the process of writing servlets by allowing developers to embed Java code within HTML.

"Servlets and JSP are competing technologies for handling web requests. Servlets are being superseded by JSP, which is preferred. The two technologies are not useful in combination."

Incorrect because JSPs and servlets complement each other rather than compete. Servlets handle business logic, while JSPs focus on presentation.

Explanation
The include() method of RequestDispatcher allows a servlet to include the content of another resource (such as a servlet, JSP, or HTML file) in the response. This means that the output of the included resource is merged

✓ The include() method of RequestDispatcher *

- sends a request to another resource like servlet, jsp or html
- (●) includes resource of file like servlet, jsp or html
- appends the request and response objects to the current servlet
- None of the above

Why Other Answers Are Incorrect
"Sends a request to another resource like servlet, JSP, or HTML."

Incorrect because sending a request is done using the forward() method, not include().

"Appends the request and response objects to the current servlet."

Incorrect because include() does not modify the request and response objects; it simply includes the output of another resource.

"None of the above."

Incorrect because include() does exist and is used for including content from other resources.

✗ What is object mapping? *

- The process of converting a class diagram to Java code.
- A program that emulates a mainframe terminal and passes user input to the mainframe.
- (●) The process of building object wrappers around Java interfaces. This allows the legacy system to interact with your Java application.
- The process of building object wrappers around legacy interfaces. This makes the legacy system available in an OO fashion

Correct answer

- (●) The process of building object wrappers around legacy interfaces. This makes the legacy system available in an OO fashion

Explanation
Object mapping refers to the technique of wrapping legacy system interfaces with object-oriented (OO) wrappers. This allows older systems, which may not be designed with OO principles, to be accessed and used in a modern OO environment.

Why Other Answers Are Incorrect
"The process of converting a class diagram to Java code."

Incorrect because object mapping is about integrating legacy systems, not converting UML diagrams into Java code.

"A program that emulates a mainframe terminal and passes user input to the mainframe."

Incorrect because object mapping is not about terminal emulation; it focuses on making legacy systems accessible in an OO manner.

"The process of building object wrappers around Java interfaces. This allows the legacy system to interact with your Java application."

Partially correct, but object mapping specifically deals with legacy interfaces, not just Java interfaces.

✓ < servlet > * 1/1
< servlet-name >myServlet< /servlet-name >
< servlet-class >com.servlet.myservlet< /servlet-class >
< /servlet >
< servlet-mapping >
< servlet-name > mySrvlet< /servlet-name >
< url-pattern >/foo/bar/*< /url-pattern >
< /servlet-mapping >
Works for which of the following

- (●) http://nm76:8080/foo/bar/index.html ✓
- ( ) http:// nm76:8080/foo/index.jsp
- ( ) http:// nm76:8080/foo/myservlet
- ( ) None

Explanation
In the given servlet configuration:

The servlet is named "myServlet" and mapped to the class com.servlet.myservlet.

The servlet mapping specifies <servlet-name>mySrvlet</servlet-name>, which is a mismatch with "myServlet" (case-sensitive issue).

The url-pattern is /foo/bar/*, meaning the servlet will handle requests that start with /foo/bar/ followed by any additional path.

Why Other Answers Are Incorrect
"http://nm76:8080/foo/index.jsp"

Incorrect because the mapping only applies to URLs starting with /foo/bar/, and /foo/index.jsp does not match.

"http://nm76:8080/foo/myservlet"

Incorrect because /foo/myservlet does not match the /foo/bar/* pattern.

Explanation
JUnit is a widely used open-source testing framework for Java that provides several key features:

Open-source framework – JUnit is freely available and widely adopted for unit testing in Java applications.

Annotations for test methods – JUnit provides annotations like @Test, @Before, @After, @BeforeClass, and @AfterClass to define test cases and lifecycle methods.

Assertions for expected results – JUnit includes assertion methods such as assertEquals(), assertTrue(), assertFalse(), and assertThrows() to validate test outcomes

✗ Which of the following is correct about JUnit? * 0/1

- (●) It is an open source framework. ✗
- ( ) It provides Annotation to identify the test methods.
- ( ) It provides Assertions for testing expected results.
- ( ) All of the above.

Correct answer

- (●) All of the above.

✗ The _____ action is used to permanently transfer control from a JSP page to another location on the local server *0/1

- (●) < jsp:include > ✗
- ( ) < jsp:param >
- ( ) < jsp:plugin >
- ( ) < jsp:forward >

Correct answer

- (●) < jsp:forward >

Explanation
The <jsp:forward> action in JSP is used to permanently transfer control from one JSP page to another resource (such as another JSP, servlet, or HTML file) on the local server. When this action is executed, the current page processing stops, and the request is forwarded to the specified resource.

Why Other Answers Are Incorrect
<jsp:include> – Incorrect because this action includes the content of another resource within the current page but does not permanently transfer control.

<jsp:param> – Incorrect because this action is used to pass parameters to another JSP or servlet, but it does not handle forwarding.

<jsp:plugin> – Incorrect because this action is used to embed Java applets or other plugins in a JSP page, not for forwarding requests.

✕ Which of the following is false regarding the Type 3 JDBC driver * 0/1

- ○ A Type 3 driver is a JDBC-Net pure Java driver
- ○ This translates JDBC calls into a database -dependent net protocol.
- ◉ Vendors of database middleware products can implement this type of driver into their products to provide interoperability with the greatest number of database servers. ✕
- ○ None

Correct answer

- ◉ A Type 3 driver is a JDBC-Net pure Java driver

Explanation
The Type 3 JDBC driver, also known as the Network Protocol Driver, translates JDBC calls into a database-independent net protocol, not a database-dependent one. This protocol is then converted into a database-specific protocol by middleware, allowing connectivity to multiple databases.

Why Other Answers Are Correct
"This translates JDBC calls into a database-dependent net protocol."

Correct because the middleware server ultimately converts the JDBC calls into a database-specific protocol.

"Vendors of database middleware products can implement this type of driver into their products to provide interoperability with the greatest number of database servers."

Correct because Type 3 drivers are designed to work with multiple databases via middleware.

✓ I have the following deployed in my web.xml file. * 1/1

< session-config >

< session-timeout >30< /session-timeout >

< /session-config >

Then I perform setMaxInactiveInterval(2400) for my session object.

The First one denotes 30 minutes and second one denoted 40 minutes.

After how long would session expire.

- ○ 30 minutes
- ◉ 40 minutes ✓
- ○ gives IllegalArguementException
- ○ Cannot tell

Explanation

The <session-timeout> value in web.xml is set in minutes, meaning 30 represents 30 minutes.

The setMaxInactiveInterval(2400) method overrides the session timeout defined in web.xml. This method accepts seconds, so 2400 seconds equals 40 minutes.

Since setMaxInactiveInterval() takes precedence over the web.xml configuration, the session will expire after 40 minutes if the user remains inactive.

Why Other Answers Are Incorrect

"30 minutes" – Incorrect because the programmatic setting (setMaxInactiveInterval(2400)) overrides the web.xml configuration.

"gives IllegalArgumentException" – Incorrect because setMaxInactiveInterval() accepts integer values in seconds and does not throw an exception for valid inputs.

"Cannot tell" – Incorrect because the behavior is well-defined: programmatic settings override web.xml settings.

Explanation
Session tracking is a mechanism used in web applications to maintain state across multiple requests from the same user. Since HTTP is stateless, session tracking helps in identifying users and preserving their data across interactions.

The four primary techniques for session tracking are:

Cookies – Small pieces of data stored on the client's browser that help maintain session information.

Hidden Form Fields – Data stored in hidden fields within forms, passed between requests.

URL Rewriting – Appending session identifiers to URLs to track user sessions.

Session Object (HttpSession) – A server-side mechanism that stores session data for each user.

✗ Which of the following are the session tracking techniques? *

- ○ URL rewriting, using session object, using cookies, using hidden fields
- ○ URL rewriting, using session object, using response object, using hidden fields
- ○ URL rewriting, using servlet object, using response object, using cookies
- ◉ URL rewriting, using request object, using response object, using session object ✗

Correct answer

- ◉ URL rewriting, using session object, using cookies, using hidden fields

Why Other Answers Are Incorrect
"URL rewriting, using session object, using response object, using hidden fields."

Incorrect because response object is not a session tracking technique.

"URL rewriting, using servlet object, using response object, using cookies."

Incorrect because servlet object is not used for session tracking.

"URL rewriting, using request object, using response object, using session object."

Incorrect because request object and response object do not track sessions.

Explanation
The ServletContextListener interface provides two key lifecycle methods:

contextInitialized(ServletContextEvent e) – This method is called when the web application starts. It allows developers to perform initialization tasks, such as setting up database connections or loading configuration settings.

contextDestroyed(ServletContextEvent e) – This method is called when the web application is shutting down. It allows developers to clean up resources, such as closing database connections or releasing memory.

✗ What are the mechanisms available in ServletContextListener interface? *

- ◉ contextInit(), contextService(), contextDestroyed()
- ○ contextInitialized((),contextDestroyed()
- ○ contextInitialized(), contextService(),contextDestroyed()
- ○ None of the above

Correct answer

- ◉ contextInitialized((),contextDestroyed()

Why Other Answers Are Incorrect
"contextInit(), contextService(), contextDestroyed()"

Incorrect because contextInit() and contextService() are not valid methods in the ServletContextListener interface.

"contextInitialized((),contextDestroyed()"

Incorrect because the syntax contextInitialized(()) is invalid.

"contextInitialized(), contextService(), contextDestroyed()"

Incorrect because contextService() is not a valid method in ServletContextListener.

✗ Which is not REST Architectural Constraints? * 0/1

- ○ Client-Server
- ○ Stateless
- ◉ Cacheable ✗
- ○ Service orchestration

Correct answer

- ◉ Service orchestration

Explanation
REST (Representational State Transfer) defines six architectural constraints that make a web service truly RESTful2:

Client-Server – Separates concerns between the client and server, allowing independent evolution.

Stateless – Each request from the client must contain all necessary information; the server does not store session state.

Cacheable – Responses must define whether they are cacheable to improve performance.

Uniform Interface – Ensures consistent interaction between clients and servers.

Layered System – Allows intermediate layers like proxies and gateways.

Code on Demand (optional) – Allows servers to send executable code to clients.

Why "Service Orchestration" Is Incorrect
Service orchestration refers to coordinating multiple services to achieve a business process, which is not a REST constraint.

REST focuses on stateless interactions rather than orchestrating multiple services.

✕ What's true about the following @Entity association between House and Window? *0/1

```
@Entity
public class Window {
@Id
private int winNo;
@ManyToOne
private House aHouse;
}
@Entity
public class House {
@Id
private int houseNo;
@OneToMany(mappedBy="aHouse")
private List windows;
```

Why Other Answers Are Incorrect
"It's OneToMany unidirectional association"

Incorrect because both entities reference each other, making it bidirectional.

"The association owner is the Window class"

Incorrect because the owning side in a bidirectional relationship is the ManyToOne side, which is Window.

- ○ It's OneToMany unidirectional association
- ◉ It's OneToMany bidirectional association ✕
- ○ The association owner is the Window class
- ○ None

Correct answer

- ◉ It's OneToMany unidirectional association

Explanation
In JPA/Hibernate, a One-to-Many bidirectional association means that both entities reference each other, allowing navigation from both sides.

Window class:

It has a @ManyToOne annotation, meaning each Window belongs to one House.

The House reference (aHouse) establishes the many-to-one relationship.

House class:

It has a @OneToMany(mappedBy="aHouse") annotation, meaning one House can have many Window objects.

The mappedBy="aHouse" ensures that the Window entity owns the relationship.

Explanation
A ResultSet cannot be reliably returned from a method that creates a Statement and executes a query because:

ResultSet is tied to the Statement and Connection – When the method exits, the Statement and Connection are typically closed, which also closes the ResultSet.

Resource Management Issues – If the ResultSet is returned, it may become invalid when the connection is closed, leading to errors like "Operation not allowed after ResultSet closed".

Best Practice – Instead of returning a ResultSet, it is recommended to map the results to a collection of Java objects (like a List) and return that

✗ A ResultSet can be reliably returned from a method that creates a Statement and executes a query? *0/1

- (●) True ✗
- ( ) False
- ( ) Result is directly proportional to hardwork
- ( ) Result will be dependent on my actions taken in right direction.

Correct answer

- (●) False

Why Other Answers Are Incorrect
True – Incorrect because returning a ResultSet directly can lead to resource leaks and closed connections.

"Result is directly proportional to hard work" – Incorrect because this is unrelated to JDBC behavior.

"Result will be dependent on my actions taken in the right direction" – Incorrect because ResultSet behavior is determined by JDBC rules, not personal effort.

✓ Inbuilt Web Container in Tomcat is? * 1/1

- (●) Catalina ✓
- ( ) Jasmine
- ( ) Jerryton
- ( ) All of the above

Explanation
Catalina is the inbuilt web container in Apache Tomcat. It is responsible for processing servlets and JSPs, following the Java Servlet Specification. Catalina provides the runtime environment for web applications deployed on Tomcat.

Why Other Answers Are Incorrect
Jasmine – Incorrect because Jasmine is not a component of Tomcat.

Jerryton – Incorrect because Jerryton does not exist in Tomcat architecture.

All of the above – Incorrect because only Catalina is the correct answer.

✓ Which is an example of the syntax used to import a class in a JSP? * 1/1

- (●) <%@ page import=”java.util.Date”%> ✓
- ( ) <% page import=”java.util.Date”%>
- ( ) <%@ page import=”java.util.Date”@%>
- ( ) <% page import java.util.Date; %>

Explanation
In JSP, the import directive is used to include Java classes in a JSP file. The
correct syntax follows this format:

jsp
<%@ page import="java.util.Date" %>
This directive ensures that the specified class (java.util.Date

Why Other Answers Are Incorrect
<% page import="java.util.Date" %>

Incorrect because the page directive must be prefixed with @, making @page mandatory.

<%@ page import="java.util.Date"@%>

Incorrect because the syntax @%> is invalid in JSP.

<% page import java.util.Date; %>

Incorrect because directives require the @ symbol and must follow the correct format.

✗ If the HTTP error 500 is generated by your servlet, you do not want to show the "Internal Server Error" page to the client. Instead, you want a custom error page to be displayed. What is the best way to accomplish this? *0/1

- (●) Forward the user to the error page using HttpServletResponse.sendRedirect method. ✗
- ( ) Forward the user to the error page using RequestDispatcher.forward method.
- ( ) Specify the mapping of the error-code 500 and the error page in the deployment descriptor.
- ( ) It is not possible to accomplish this

Correct answer

- (●) Specify the mapping of the error-code 500 and the error page in the deployment descriptor.

Explanation
To display a custom error page when an HTTP 500 Internal Server Error occurs, you should configure the web.xml deployment descriptor using the <error-page> element. This ensures that whenever a 500 error is encountered, the user is redirected to a predefined error page.

Example Configuration in web.xml
xml
```
<error-page>
   <error-code>500</error-code>
   <location>/customErrorPage.jsp</location>
</error-page>
```
This setup ensures that any 500 error will automatically redirect users to customErrorPage.jsp.

Why Other Answers Are Incorrect
"Forward the user to the error page using HttpServletResponse.sendRedirect method."

Incorrect because sendRedirect() creates a new request, which does not preserve the original error details.

"Forward the user to the error page using RequestDispatcher.forward method."

Incorrect because forward() is typically used for internal request forwarding, but it does not automatically handle HTTP error codes.

"It is not possible to accomplish this."

Incorrect because custom error pages can be configured using web.xml

Explanation
In Hibernate, mapping files define how Java objects are mapped to database tables. The standard naming convention for Hibernate XML mapping files follows the format:

plaintext
<classname>.hbm.xml
For example, if you have an entity class named Employee, the corresponding mapping file would be:

plaintext
Employee.hbm.xml

✓ What is the naming convention for Hibernate XML mapping file extensions? *1/1

- .hibernate.xml
- (●) .hbm.xml ✓
- .hibernate_data.xml
- None of the above

Why Other Answers Are Incorrect
".hibernate.xml" – Incorrect because Hibernate does not use .hibernate.xml as a standard mapping file extension.

".hibernate_data.xml" – Incorrect because this is not a recognized Hibernate mapping file format.

"None of the above" – Incorrect because .hbm.xml is the correct naming convention.

✗ Maven is used to do which of the following? * 0/1

A. Build software development projects
B. Publish software development projects
C. Deploy software development projects

- (●) Only A ✗
- Both A & B
- All A,B & C
- None

Correct answer

- (●) All A,B & C

Explanation
Maven is a powerful build automation tool that helps manage software development projects efficiently. It provides a structured approach to:

Building software development projects – Maven automates the compilation, testing, and packaging of applications.

Publishing software development projects – Maven allows developers to publish artifacts to repositories like Maven Central or Nexus.

Deploying software development projects

Explanation
The HEAD request method in HTTP is used to retrieve only the headers of a response, without the actual body content. This is useful for:

Checking the document size using the Content-Length header.

Determining the last modification time using the Last-Modified header.

Verifying the availability of a resource without downloading it.

✓ _____ request indicates that client wants to see only the headers of the response, to determine the document size, modification time, or general availability *1/1

- PUT
- GET
- TRACE
- HEAD ✓

Why Other Answers Are Incorrect
PUT – Incorrect because PUT is used to update or create a resource on the server.

GET – Incorrect because GET retrieves both headers and body of the response.

TRACE – Incorrect because TRACE is used for debugging, showing the request as received by the server.

Feedback 0 of 0 points

From now onwards, I will give my best in everything in my life without any excuses. Because I know, problem is the part and parcel of life. We should always look for solutions. *

- I PROMISE

How was your experience ? Anything to share ?

.....

Difficulty level of mock *

Dropdown

Moderate