

### **Q1. How “around” advice works?**

Run advice before and after method execution regardless of its outcome

Run advice after a class loads

Run advice after HTTP response is returned

Run advice after HTTP request is processed

Answer: a

Explanation: Around advice wraps the target method, executing code both before and after it runs, even if an exception occurs.

---

### **Q2. The Spring Context module is used for?**

Context creation

Dependency injection

Answer: b

Explanation: The Context module provides a way to configure and wire application objects via dependency injection.

---

### **Q3. Which annotation is used for mapping a URL with a Controller in Spring Framework?**

@Mapping

@Controller

@URL

@RequestMapping

Answer: d

Explanation: @RequestMapping maps HTTP requests to handler methods in controller classes.

---

**Q4. When injecting an object, which attribute is used in setter- and constructor-injection statements?**

value

name

type

ref

Answer: d

Explanation: “ref” specifies a reference to another bean defined in the Spring context.

---

**Q5. Which of these is *not* a module in the core Spring architecture?**

Spring AOP

Spring ORM

Spring APO

Spring DAO

Answer: c

Explanation: “Spring APO” is a typo; the correct module is Spring AOP (Aspect-Oriented Programming).

---

**Q6. Which XML tag is required in a Spring configuration file when a collection of strings is injected?**

<collection>

<string>

Answer: a

Explanation: <collection> groups multiple <value> subelements for injection into a Collection-typed property.

---

**Q7. Which of the following will handle all requests and responses in a Spring MVC application?**

DispatcherServlet

ApplicationContext

WebApplicationContext

HttpServlet

Answer: a

Explanation: DispatcherServlet acts as the front controller, routing requests to handlers and view resolvers.

---

**Q8. In Spring MVC, which annotation binds a method parameter to the body of a web request?**

@RequestBody

@RequestParam

@RequestMapping

Answer: a

Explanation: @RequestBody tells Spring to deserialize the HTTP request body into a Java object.

---

**Q9. Which of the following works as a controller in the Struts Framework?**

StrutsController

StrutsFilter

StrutsPrepareFilter

StrutsPrepareAndExecuteFilter

Answer: d

Explanation: StrutsPrepareAndExecuteFilter initializes Struts and executes actions for each request.

---

**Q10. Which JSF tag converts a control's value into a number?**

`f:convertNumber`

`f:Number`

`f:convertInteger`

Answer: a

Explanation: `f:convertNumber` formats and parses numeric values according to locale.

---

**Q11. Which of the following annotations is *not* a built-in Java annotation?**

`@FunctionalInterface`

`@ThreadSafe`

`@Override`

`@Native`

Answer: b

Explanation: `@ThreadSafe` is not part of standard Java; it comes from external libraries like JSR-305.

---

**Q12. Which statement is *not* correct about Java annotations?**

A retention policy determines when an annotation is discarded

An annotation can be a marker annotation having no values

An annotation can be marked as private

The `@Repeatable` annotation allows multiple uses of an annotation

Answer: c

Explanation: Annotation types cannot have private visibility; they must be public or package-private.

---

**Q13. Which of the following is *not* true about Apache Maven?**

It is a software project management and comprehension tool

It is not a build tool like Ant

It is based on the concept of a Project Object Model

It is an automation tool used primarily for Java projects

Answer: b

Explanation: Maven is both a project management tool and a build tool, like Ant.

---

**Q14. Which of the following annotations is *not* built-in?**

@FunctionalInterface

@ThreadSafe

@Override

@Native

Answer: b

Explanation: Only @FunctionalInterface, @Override, and @Native are part of Java SE's annotation set.

---

**Q15. Which of the following is *not* true with respect to cookies?**

Cookies are one solution for session tracking

Cookies are generated at the client side

Cookies carry small pieces of information

None of the above

Answer: b

Explanation: Cookies are created by the server and sent to the client for storage.

---

**Q16. What is the full form of JNDI?**

Java Network and Directory Implementation

Java Name Directory Interface

Java Network Directory Interface

Java Naming and Directory Interface

Answer: d

Explanation: JNDI stands for Java Naming and Directory Interface.

---

**Q17. Which design pattern does *not* belong to Creational patterns?**

Builder

Observer

Singleton

Abstract Factory

Answer: b

Explanation: Observer is a behavioral pattern, not creational.

---

**Q18. When a change to one object requires changing others, which design pattern is used?**

State

Observer

Mediator

Answer: b

Explanation: Observer notifies dependent objects automatically when its state changes.

---

**Q19. Which constructor is *not* possible for the `Locale` class?**

`Locale(String language)`

`Locale(String country)`

`Locale(String language, String country)`

```
Locale(String language, String country, String variant)
```

Answer: b

Explanation: There is no single-argument constructor that takes only `country`.

---

**Q20. Which of the following is *not* an MVC framework?**

Struts

JSF

Hibernate

Spring MVC

Answer: c

Explanation: Hibernate is an ORM tool, not a web MVC framework.

---

**Q21. Which notation is correct for declaring the JSTL XML tag library with URI `http://java.sun.com/jsp/jstl/xml` and prefix `x`?**

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x" %>
```

```
<%@ page taglib-uri="http://java.sun.com/jsp/jstl/xml" namespace-  
prefix="x" %>
```

```
<%@ page taglib-uri="http://java.sun.com/jsp/jstl/xml" taglib-  
prefix="x" %>
```

```
<%@ taglib-uri="http://java.sun.com/jsp/jstl/xml" taglib-  
prefix="x" %>
```

Answer: a

Explanation: The standard directive uses `taglib uri` and `prefix`.

---

**Q22. Which attribute is *not* valid for the JSP page directive?**

language

isThread

info

extends

Answer: b

Explanation: The correct attribute is `isThreadSafe`, not `isThread`.

---

**Q23. Which directive imports the `java.util` package into the current JSP page?**

`<%@ import="java.util" %>`

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

`<%@ import="java.util.*" %>`

`<%@ page import="java.util.*" %>`

Answer: d

Explanation: The `<%@ page import="..." %>` syntax is required.

---

**Q24. Which method in `HttpSession` retrieves a session-scoped attribute?**

`Object getAttribute(String name)`

`String getAttribute(String name)`

`void getAttribute(String name)`

`String getAttribute(Object key)`

Answer: a

Explanation: `getAttribute` returns an `Object` by attribute name.

---

**Q25. Which of the following is *not* a valid Hibernate annotation?**

`@Entity`

`@Table`

`@Column`

`@AllanITAnna`

Answer: d

Explanation: There is no `@AllanITAnna` annotation in Hibernate.

---



**Q26. Which JSTL tag iterates over a collection of objects?**

`<c:iterate>`

`<c:for>`

`<c:forEach>`

`<c:loop>`

**Answer: c**

**Explanation:** `<c:forEach>` is the standard iteration tag in JSTL.

---

**Q27. Which is a valid `<jsp:getProperty>` statement?**

`<jsp:getProperty id="firstbean" name="firstproperty" />`

`<jsp:getProperty name="firstbean" property="firstproperty" />`

`<jsp:getProperty name="firstbean" value="firstproperty" />`

`<jsp:getProperty id="firstbean" param="firstproperty" />`

**Answer: b**

**Explanation:** The correct attributes are `name` (bean ID) and `property`.

---

**Q28. Where inside a WAR do external JARs belong?**

`/jars/`

`/WAR-INF/jars/`

`/WEB-INF/jars/`

`/WEB-INF/lib/`

**Answer: d**

**Explanation:** `/WEB-INF/lib/` is the standard location for JAR dependencies.

---

**Q29. What is the full form of UDDI?**

Uniform Discovery, Design and Information

Universal Description, Discovery and Integration

Unified Directory, Design and Implementation

Universal Discovery, Data and Information

Answer: b

Explanation: UDDI stands for Universal Description, Discovery, and Integration.

---

**Q30. How do you obtain a Hibernate Session object?**

`SessionFactory.open()`

`SessionFactory.getSession()`

`SessionFactory.openSession()`

`Session.getSession()`

Answer: c

Explanation: `openSession()` opens a new Session instance.

---

**Q31. Which of the following is *not* true about the JavaMail API?**

The JavaMail API is included in the Java SE runtime

`Transport.send()` is used to send messages

`InternetAddress` represents email addresses

`MessagingException` is thrown on mail errors

Answer: a

Explanation: JavaMail is not bundled with Java SE; it must be added separately.

---

**Q32. Which JSP directive includes `header.html` in a page?**

`<%@ include="header.html" %>`

`<%@ include file="header.html" %>`

`<%@ page include="header.html" %>`

```
<%@ include url="header.html" %>
```

Answer: b

Explanation: The `file` attribute specifies the resource to include.

---

**Q33. What is the root element of the Hibernate configuration file?**

```
<hibernate-config>
```

```
<hibernate-configuration>
```

```
<hibernate-mapping>
```

Answer: b

Explanation: `<hibernate-configuration>` wraps settings and mapping resources.

---

**Q34. Which of the following is *not* an implicit JSP object?**

response

config

pageContext

system

Answer: d

Explanation: There is no implicit “system” object; common ones include `session`, `application`, etc.

---

**Q35. Which JDBC driver type converts JDBC calls into ODBC calls?**

Type 1 JDBC Driver

Type 2 JDBC Driver

Type 3 JDBC Driver

Type 4 JDBC Driver

Answer: a

Explanation: Type 1 is the JDBC-ODBC bridge driver.

---

**Q36. What is the root element for a web-app deployment descriptor?**

`<web-application>`

`<web>`

`<web-app>`

`<deploy-descriptor>`

**Answer: c**

**Explanation:** The standard root tag in `web.xml` is `<web-app>`.

---

**Q37. Which JSP tag syntax denotes directives?**

`<%! ... %>`

`<%@ ... %>`

`<% ... %>`

`<%~ ... %>`

**Answer: b**

**Explanation:** `<%@ ... %>` is used for page, include, and taglib directives.

---

**Q38. Which snippet correctly calls the stored procedure `findResult()` ?**

```
Statement stmt = conn.createStatement();  
stmt.executeQuery("findResult()");
```

```
CallableStatement cs = conn.prepareCall("{call findResult()}");  
cs.executeQuery();
```

```
PreparedStatement pstmt = conn.prepareStatement("findResult()");  
pstmt.execute();
```

```
Statement stmt = conn.createStatement();  
stmt.executeStoredProcedure("findResult()");
```

**Answer: b**

**Explanation:** `CallableStatement` with the `{call ...}` syntax is required to invoke stored procedures.