This section presents you various set of Mock Tests related to Spring Framework. You can download these sample mock tests at your local machine and solve offline at your convenience. Every mock test is supplied with a mock test key to let you verify the final score and grade yourself.

Q 1 - What is spring?
A - Spring is an open source development framework for enterprise Java.
B - Spring is a proprietary framework.
C - Spring is a development framework for .Net applications.
D - Spring is a development framework for PHP based applications.

Q 2 - Which of the following is correct assertion about spring?
A - Spring enables developers to develop enterprise-class applications using POJOs.
B - Spring is organized in a modular fashion.
C - Testing an application written with spring is simple because environment-dependent code is moved into this framework.
D - All of above.

Q 3 - What is Dependency Injection?
A - It is a design pattern which implements Inversion of Control for software applications.
B - It is one of the spring module.
C - It is a technique to get dependencies of any project.
D - It is used to promote tight coupling in code.

Q 4 - Which of the following is correct about dependency injection?
A - It helps in decoupling application objects from each other.
B - It helps in deciding the dependencies of objects.
C - It stores objects states in database.
D - It stores object states in file system.

**Q 5 - What AOP stands for?**

A - Aspect Oriented Programming  
B - Any Object Programming  
C - Asset Oriented Programming  
D - Asset Oriented Protocol

**Q 6 - What is true about cross-cutting concerns?**

A - The functions that span multiple points of an application are called cross cutting concerns.  
B - Cross-cutting concerns are conceptually separate from the application's business logic.  
C - Logging is one of the examples of cross cutting concerns.  
D - All of the above.

**Q 7 - Which are the modules of core container?**

A - Beans, Core, Context, SpEL  
B - Core, Context, ORM, Web  
C - Core, Context, Aspects, Test  
D - Bean, Core, Context, Test

**Q 8 - Which are the modules of Data Access/ integration layer?**

A - JDBC, ORM, OXM, JMS, Transactions  
B - JDBC, ORM, OXM, JMS  
C - JDBC, ORM, Web, Beans  
D - JDBC, ORM, OXM, JMS

**Q 9 - Which are the modules of Web layer?**

A - WebSocket, Servlet, Web, Portlet  
B - WebSocket, Servlet, Web-MVC, Web  
C - HTML, JSP, WEB, Portlet  
D - HTML, Servlet, WEB, Portlet

**Q 10 - Which of the statement is not correct?**
A - Core and beans modules provide the fundamental parts of the framework, including Dependency Injection feature.
B - The SpEL module provides a powerful Expression Language for querying and manipulating an object graph at runtime.
C - Aspects module provides integration with AspectJ.
D - None of the above.

**Q 11 - Which of the statement is correct?**

A - The JDBC module provides a JDBC-abstraction layer that removes the need to do tedious JDBC related coding.
B - The ORM module provides integration layers for popular object-relational mapping APIs, including JPA, JDO, Hibernate, and iBatis.
C - The Java Messaging Service JMS module contains features for producing and consuming messages.
D - All of the above.

**Q 12 - Which of the statement is correct?**

A - The AOP module provides aspect-oriented programming implementation allowing you to define method-interceptors and pointcuts to cleanly decouple code that implements functionality that should be separated.
B - The Aspects module provides integration with AspectJ - Which is again a powerful and mature aspect oriented programming AOP framework.
C - The Instrumentation module provides class instrumentation support and class loader implementations to be used in certain application servers.
D - All of the above.

**Q 13 - What types of Dependency injection does spring supports?**

A - Constructor based, Setter based
B - Constructor based, Setter based, Getter Based
C - Setter based, Getter based, Properties based
D - Constructor based, Setter based, Properties based

**Q 14 - Which are the IoC containers in Spring?**

A - BeanFactory, ApplicationContext
B - BeanFactory, ApplicationContext, IocContextFactory
C - BeanFactory, BeanContext, IocContextFactory
D - BeanFactory, ApplicationContext, BeanContext

**Q 15 - Which is the correct implementation class of BeanFactory?**
**Q 16 - Which are the correct implementation classes of ApplicationContext?**

A - FileSystemXmlApplicationContext, ClassPathXmlApplicationContext, WebXmlApplicationContext
B - FileSystemApplicationContext, ClassPathApplicationContext, WebApplicationContext
C - AdvancedApplicationContext, FileApplicationContext
D - FileSystemApplicationContext, ClassPathApplicationContext

**Q 17 - Which of the following stands true for spring beans?**

A - Spring beans are managed by the Spring IoC container.
B - Spring beans are instantiated, assembled, and otherwise managed by a Spring IoC container.
C - Spring beans are simple POJOs.
D - All of the above.

**Q 18 - Which is the way to provide configuration metadata to spring?**

A - XML Based configuration file.
B - Annotation based configuration.
C - Java based configuration.
D - All of the above.

**Q 19 - What is bean scope?**

A - Bean scope forces Spring to produce a new bean instance as per the scope defined.
B - Bean scope defines the accessibility of bean in a java class.
C - Bean scope defines the accessibility of bean in a java package.
D - Bean scope defines the accessibility of bean in a web application.

**Q 20 - What is singleton scope?**

A - This scopes the bean definition to a single instance per Spring IoC container.
B - This scopes the bean definition to a single instance per HTTP Request.
C - This scopes the bean definition to a single instance per HTTP Session.
D - This scopes the bean definition to a single instance per HTTP Application/Global session.
<table>
<thead>
<tr>
<th>Question</th>
<th>Option A</th>
<th>Option B</th>
<th>Option C</th>
<th>Option D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q 21 - What is prototype scope?</td>
<td>This scopes a single bean definition to have any number of object instances.</td>
<td>This scopes the bean definition to a single instance per HTTP Request.</td>
<td>This scopes the bean definition to a single instance per HTTP Session.</td>
<td>This scopes the bean definition to a single instance per HTTP Application/Global session.</td>
</tr>
<tr>
<td>Q 22 - What is request scope?</td>
<td>This scopes a bean definition to an HTTP request.</td>
<td>This scopes the bean definition to Spring IoC container.</td>
<td>This scopes the bean definition to HTTP Session.</td>
<td>This scopes the bean definition HTTP Application/Global session.</td>
</tr>
<tr>
<td>Q 23 - What is session scope?</td>
<td>This scopes a bean definition to an HTTP session.</td>
<td>This scopes the bean definition to Spring IoC container.</td>
<td>This scopes the bean definition to HTTP request.</td>
<td>This scopes the bean definition HTTP Application/Global session.</td>
</tr>
<tr>
<td>Q 24 - What is global-session scope?</td>
<td>This scopes a bean definition to an HTTP Application/Global session.</td>
<td>This scopes the bean definition to Spring IoC container.</td>
<td>This scopes the bean definition to HTTP request.</td>
<td>This scopes the bean definition to HTTP Session.</td>
</tr>
<tr>
<td>Q 25 - What is default scope of bean in Spring framework?</td>
<td>singleton</td>
<td>prototype</td>
<td>request</td>
<td>session</td>
</tr>
</tbody>
</table>
Q 27 - What is true about `<list>` collection configuration elements?
A - This helps in wiring a list of values, allowing duplicates.
B - This helps in wiring a list of values but without any duplicates.
C - This can be used to inject a collection of name-value pairs where name and value can be of any type.
D - This can be used to inject a collection of name-value pairs where the name and value are both Strings.

Q 28 - What is true about `<set>` collection configuration elements?
A - This helps in wiring a list of values, allowing duplicates.
B - This helps in wiring a list of values but without any duplicates.
C - This can be used to inject a collection of name-value pairs where name and value can be of any type.
D - This can be used to inject a collection of name-value pairs where the name and value are both Strings.

Q 29 - What is true about `<map>` collection configuration elements?
A - This helps in wiring a list of values, allowing duplicates.
B - This helps in wiring a list of values but without any duplicates.
C - This can be used to inject a collection of name-value pairs where name and value can be of any type.
D - This tag is not supported.

Q 30 - What is true about `<props>` collection configuration elements?
A - This helps in wiring a list of values, allowing duplicates.
B - This helps in wiring a list of values but without any duplicates.
C - This can be used to inject a collection of name-value pairs where name and value can be of any type.
D - This can be used to inject a collection of name-value pairs where the name and value are both Strings.

Q 31 - What is bean autowiring?
A - Autowiring lets Spring resolve collaborators `otherbeans` for your bean by inspecting the contents of the BeanFactory without using `<constructor-arg>` and `<property>` elements.
B - Autowiring injects values in spring beans.
C - Autowiring injects one bean into another.
D - Autowiring helps in wiring a list of values, allowing duplicates.

Q 32 - Which are the different modes of autowiring?
A - no, byName, byType, constructor, autodetect
B - no, byName, byType, constructor, autocorrect
C - byName, byContent, constructor, autodetect
D - byName, byContent, setter, autodetect

Q 33 - What is no mode of autowiring?
A - Default setting which means no autowiring and you should use explicit bean reference for wiring.
B - Autowiring by property name.
C - Spring first tries to wire using autowire by constructor, if it does not work, Spring tries to autowire by byType.
D - Similar to byType, but type applies to constructor arguments.

Q 34 - What is byName mode of autowiring?
A - Default setting which means no autowiring and you should use explicit bean reference for wiring.
B - Autowiring by property name. Spring tries to match and wire its properties with the beans defined by the same names in the configuration file.
C - Spring first tries to wire using autowire by constructor, if it does not work, Spring tries to autowire by byType.
D - Similar to byType, but type applies to constructor arguments.

Q 35 - What is byType mode of autowiring?
A - Default setting which means no autowiring and you should use explicit bean reference for wiring.
B - Autowiring by property name. Spring tries to match and wire its properties with the beans defined by the same names in the configuration file.
C - Spring first tries to wire using autowire by constructor, if it does not work, Spring tries to autowire by byType.
D - Autowiring by property type. Spring tries to match and wire a property if its type matches with exactly one of the beans name in configuration file.

Q 36 - What is constructor mode of autowiring?
A - Autowiring by property name. Spring tries to match and wire its properties with the beans defined by the same names in the configuration file.
B - Spring first tries to wire using autowire by constructor, if it does not work, Spring tries to autowire by byType.
C - Autowiring by property type. Spring tries to match and wire a property if its type matches with exactly one of the beans name in configuration file.

D - Similar to byType, but type applies to constructor arguments. If there is not exactly one bean of the constructor argument type in the container, a fatal error is raised.

Q 37 - What is autodetect mode of autowiring?

A - Similar to byType, but type applies to constructor arguments. If there is not exactly one bean of the constructor argument type in the container, a fatal error is raised.

B - Autowiring by property name. Spring tries to match and wire its properties with the beans defined by the same names in the configuration file.

C - Spring first tries to wire using autowire by constructor, if it does not work, Spring tries to autowire by byType.

D - Autowiring by property type. Spring tries to match and wire a property if its type matches with exactly one of the beans name in configuration file.

Q 38
Can you inject null and empty string values in Spring?

A - Yes

B - No

Q 39 - How do you turn on annotation wiring?

A - Add <annotation-context:config/> to bean configuration.

B - Add <annotation-config/> to bean configuration.

C - Add <annotation-context-config/> to bean configuration.

D - Add <context:annotation-config/> to bean configuration.

Q 40 - What does @Required annotation mean?

A - This annotation indicates that bean property must be populated by the user.

B - This annotation indicates that bean property is required while saving the bean data to database.

C - This annotation simply indicates that the affected bean property must be populated at configuration time, through an explicit property value in a bean definition or through autowiring.

D - This annotation indicates that bean property is required while serializing the bean.

Q 41 - What is true about @Autowired annotation?

A - The @Autowired annotation can be used to autowire bean on the setter method.

B - This annotation provides more fine-grained control over where and how autowiring should be accomplished.

C - The @Autowired annotation can be used to autowire bean on the methods with arbitrary
names and/or multiple arguments.
D - All of above.

**Q 42 - What is ContextRefreshedEvent event?**
A - This event is published when the Servlet Context is either initialized or refreshed.
B - This event is published when the HTTP Request is received.
C - This event is published when the HTTP Response is returned.
D - This event is published when the ApplicationContext is either initialized or refreshed.

**Q 43 - What is ContextStartedEvent event?**
A - This event is published when the Servlet Context is either initialized or refreshed.
B - This event is published when the HTTP Request is received.
C - This event is published when the ApplicationContext is started using the start method on the ConfigurableApplicationContext interface.
D - This event is published when the HTTP Response is returned.

**Q 44 - What is ContextStoppedEvent event?**
A - This event is published when the Servlet Context is either initialized or refreshed.
B - This event is published when the ApplicationContext is stopped using the stop method on the ConfigurableApplicationContext interface.
C - This event is published when the HTTP Request is received.
D - This event is published when the HTTP Response is returned.

**Q 45 - What is ContextClosedEvent event?**
A - This event is published when the Servlet Context is either initialized or refreshed.
B - This event is published when the HTTP Request is received.
C - This event is published when the HTTP Response is returned.
D - This event is published when the ApplicationContext is closed using the close method on the ConfigurableApplicationContext interface.

**Q 46 - What is RequestHandledEvent event?**
A - This event is published when the Servlet Context is either initialized or refreshed.
B - This event is published when the HTTP Request is received.
C - This event is published when the HTTP session is initialized or refreshed.
D - This event is published when the HTTP Request is serviced.
**Q 47 - What is aspect?**
A - Aspect is a way to do the dependency injection.
B - A module which has a set of APIs providing cross-cutting requirements.
C - Aspect is used to log information of application.
D - Aspect represents properties of spring based application.

**Q 48 - What is Join point?**
A - This represents a point in your application which joins two objects.
B - This represents a point in your object where you join values.
C - This represents a point in your object where you join injected values.
D - This represents a point in your application where you can plug-in AOP aspect.

**Q 49 - What is Advice?**
A - This is the way to instruct object to behave in certain manner.
B - This is used to inject values in objects.
C - This is the actual action to be taken either before or after the method execution.
D - This is not invoked during program execution by Spring AOP framework.

**Q 50 - What is Pointcut?**
A - This represents a point in your application where you can plug-in AOP aspect.
B - This is a set of one or more joinpoints where an advice should be executed.
C - This is used to inject values in objects.
D - This is invoked during program execution by Spring AOP framework.

**ANSWER SHEET**

<table>
<thead>
<tr>
<th>Question Number</th>
<th>Answer Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>D</td>
</tr>
<tr>
<td>3</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>A</td>
</tr>
<tr>
<td>5</td>
<td>A</td>
</tr>
<tr>
<td>6</td>
<td>A</td>
</tr>
<tr>
<td>7</td>
<td>A</td>
</tr>
<tr>
<td>8</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>9</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>D</td>
</tr>
<tr>
<td>11</td>
<td>D</td>
</tr>
<tr>
<td>12</td>
<td>D</td>
</tr>
<tr>
<td>13</td>
<td>A</td>
</tr>
<tr>
<td>14</td>
<td>A</td>
</tr>
<tr>
<td>15</td>
<td>A</td>
</tr>
<tr>
<td>16</td>
<td>A</td>
</tr>
<tr>
<td>17</td>
<td>D</td>
</tr>
<tr>
<td>18</td>
<td>D</td>
</tr>
<tr>
<td>19</td>
<td>A</td>
</tr>
<tr>
<td>20</td>
<td>A</td>
</tr>
<tr>
<td>21</td>
<td>A</td>
</tr>
<tr>
<td>22</td>
<td>A</td>
</tr>
<tr>
<td>23</td>
<td>A</td>
</tr>
<tr>
<td>24</td>
<td>A</td>
</tr>
<tr>
<td>25</td>
<td>A</td>
</tr>
<tr>
<td>26</td>
<td>A</td>
</tr>
<tr>
<td>27</td>
<td>A</td>
</tr>
<tr>
<td>28</td>
<td>B</td>
</tr>
<tr>
<td>29</td>
<td>C</td>
</tr>
<tr>
<td>30</td>
<td>D</td>
</tr>
<tr>
<td>31</td>
<td>A</td>
</tr>
<tr>
<td>32</td>
<td>A</td>
</tr>
<tr>
<td>33</td>
<td>A</td>
</tr>
<tr>
<td>34</td>
<td>B</td>
</tr>
<tr>
<td>35</td>
<td>D</td>
</tr>
<tr>
<td>36</td>
<td>D</td>
</tr>
<tr>
<td>37</td>
<td>C</td>
</tr>
<tr>
<td>38</td>
<td>A</td>
</tr>
<tr>
<td>39</td>
<td>D</td>
</tr>
<tr>
<td>40</td>
<td>C</td>
</tr>
<tr>
<td>41</td>
<td>B</td>
</tr>
<tr>
<td>42</td>
<td>D</td>
</tr>
<tr>
<td>43</td>
<td>C</td>
</tr>
<tr>
<td>44</td>
<td>B</td>
</tr>
<tr>
<td>----</td>
<td>---</td>
</tr>
<tr>
<td>45</td>
<td>D</td>
</tr>
<tr>
<td>46</td>
<td>C</td>
</tr>
<tr>
<td>47</td>
<td>B</td>
</tr>
<tr>
<td>48</td>
<td>D</td>
</tr>
<tr>
<td>49</td>
<td>C</td>
</tr>
<tr>
<td>50</td>
<td>B</td>
</tr>
</tbody>
</table>