**MCQ Spring and Hibernate**

**Total Marks 50 (each 2 marks)**

**Question 1: REST stands for Representational State Transfer.It’s an is an architectural style which can be used to design web services, that can be consumed from a variety of clients. Which of the following is NOT rest annotation.**

1. @PathVariable
2. @RequestParam
3. @RequestMapping
4. @ResponseBody

**Question 2: While implementing Spring, Shivansh wants to provide dependent object to be used. Which of the following annotation, he should use to implement the same?**

1. @Resource
2. @Qualifier
3. @Dependent
4. @Autowired

**Question 3: Meenal wants to implement Inheritance in his Banking application. Which of the following strategy of Is-A mapping, requires discriminator column?**

1. Single Table Mapping
2. Joined Table Mapping
3. Table Per Class Mapping
4. None of the Above

**Question 4: Ram is building an Employee management system where he wants to relate Manager IS-A Employee ,In order to do the same Which of the following annotation he should use in his application?**

1. @Inheritance
2. @IsAMapping
3. @InheritanceMapping
4. @MappedBy

### Question 5: Observe following code…

### @Component ("resourceBean")

### @Scope("singleton")

### public class GlobalInvestment {

### private @Value ("GI Pvt. Ltd.")String firmName;

### Which statement is TRUE?

1. Bean will be declared as Entity Bean of Hibernate
2. Scope of the bean is 'Prototype' in JVM
3. The instance field in an object will be initialized by a string 'GI Pvt. Ltd.’
4. The @Value annotation is applicable on property and not on private fields.

**Question 6: Which of the following is not the common implementations of the ApplicationContext?**

1. The FileSystemXmlApplicationContext container loads the definitions of the beans from an XML file. The full path of the XML bean configuration file must be provided to the constructor.
2. The ClassPathXmlApplicationContext container also loads the definitions of the beans from an XML file. Here, you need to set CLASSPATH properly because this container will look bean configuration XML file in CLASSPATH.
3. The WebXmlApplicationContext: container loads the XML file with definitions of all beans from within a web application.
4. The ServletContextApplicationContext: Container converts a servlet into Spring Application context.

### Question 7: There may be a situation when you create more than one bean of the same type and want to wire only one of them with a property. In such cases, you can use the \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ annotation along with @Autowired to remove the confusion by specifying which exact bean will be wired.

### @Required

### @Qualifier

### @Autowired

### @Resource

### Question 8: The \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ represents a point in an application where we can plug-in an AOP aspect. It is the actual place in the application where an action will be taken using Spring AOP framework.

### Proxy

### JoinPoint

### pointcut

### Advice

### Target

### Question 9: Which Transaction management type is more preferable in Spring Framework?

### Declarative transaction management

### Programmatic transaction management

### None of the Above.

### Both A and B

### Question 10: Which out of following is NOT sub-annotation of @Component?

### The @Bean

### The @Service

### The @Repository

### The @Controller

### Question 11: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ provide access to the application behaviour that you typically define through a service interface. It interprets user input and transform it into a model that is represented to the user by the view. Spring implements it in a very abstract way, which enables you to create its wide variety.

1. Controllers
2. Advice
3. Request Mapping Controllers
4. WebApplicationContext
5. None of the Above

**Question 12 Which are the modules of core container?**

1. [Beans, Core, Context, expression](javascript:void(0);)
2. [Core, Context, ORM, Web](javascript:void(0);)
3. [Core, Context, Aspects, Test](javascript:void(0);)
4. [Bean, Core, Context, Test](javascript:void(0);)

### Question 13: Which are the important beans lifecycle methods?

1. afterPropertiesSet()
2. destroy()
3. Both A and B
4. None of the Above

### Question 14: If the bean implements IntializingBean, its \_\_\_\_\_\_\_\_\_\_\_\_ method is called. If the bean has init method declaration, the specified initialization method is called.

1. postProcesserBeforeInitialization()
2. afterPropertySet()
3. setBeanFactory()
4. setBeanName()

### Question 15: What is the role of ContextLoaderListener?

1. It starts a web container
2. It stats a WebApplicationContext
3. It handles all requests and responses
4. It creates and starts the ApplicationContext.

### Question 16: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ an aspect of Inversion of Control (IoC), is a general concept, and it can be expressed in many different ways. This concept says that you do not create your objects but describe how they should be created. You don’t directly connect your components and services together in code but describe which services are needed by which components in a configuration file. A container (the IOC container) is then responsible for hooking it all up.

1. BeanPostProcessor
2. ApplicationContext
3. Aspect Oriented Programming
4. Dependency Injection
5. None of the above

### Question 17: The\_\_\_\_\_\_\_\_\_\_ is built on the application context module, providing a context that is appropriate for web-based applications. This module also contains support for several web-oriented tasks such as transparently handling multi-context environment and programmatic binding of request parameters to your business objects. It also contains integration support with different view technologies like Apache Velocity.

1. Spring WEB module
2. Spring AOP module
3. Spring ORM module
4. Spring DAO module
5. None of the above

### Question 18: The basic web services platform is XML + HTTP. All the standard web services work using the following components : SOAP,XML,UDDI.

### True or false

1. True
2. False

### Question 19: The WebApplicationContext is an extension of the plain ApplicationContext that has some extra features necessary for web applications. It differs from a normal ApplicationContext in that it is capable of resolving themes, and that it knows which servlet it is associated with.

1. True
2. False

**Question 20: @SpringBootApplication also provides aliases to customize the attributes of\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ and \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Annotations..**

1. @EnableAutoConfiguration and @ComponentScan
2. @EnableAnnotationConfig and @PackageScan
3. @ EnableAnnotationConfig and @ComponentScan
4. @EnableAutoConfiguration and @ PackageScan
5. None of the Above

### Question 21:

import java.util.concurrent.atomic.AtomicLong;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class GreetingController {

private static final String template = "Hello, %s!";//line 1

private final AtomicLong counter = new AtomicLong();//line 2

@RequestMapping("/greeting")//line 3

public Greeting greeting(@RequestParam(value="name", defaultValue="World") String name) {

return new Greeting(counter.incrementAndGet(),

String.format(template, name));

}

}

**What is NOT correct statement about the above code:**

1. [**@RestController**](https://docs.spring.io/spring/docs/current/javadoc-api/org/springframework/web/bind/annotation/RestController.html) **annotation, and the GreetingController above handles GET requests for /greeting by returning a new instance of the Greeting class**
2. **The @RequestMapping annotation ensures that HTTP requests to /greeting are mapped to the greeting() method**
3. **@RequestParam binds the value of the query string parameter name into the name parameter of the greeting() method 2**
4. **Error at Line 3**

**Question 22: What does the code at line 4 state?**

1. **@SpringBootApplication**
2. **public class SpringBootWebApplication {**
3. **public static void main(String[] args) {** 
   * 1. **SpringApplication.run(SpringBootWebApplication.class, args); //line 4**

**} }**

2. bootstrap and launch a Spring application from a Java main method**.**
3. Create an appropriate [ApplicationContext](https://docs.spring.io/spring-framework/docs/5.1.6.RELEASE/javadoc-api/org/springframework/context/ApplicationContext.html?is-external=true) instance.
4. Refresh the application context, loading all singleton beans
5. All of the Above

### Question 23: Statement A : “Authentication” is the process of establishing a principal is who they claim to be

### Statement B : “Authorization” refers to the process of deciding whether a principal is allowed to perform an action within your application.

1. Both A and B True
2. Both Statement are false
3. A is true B is false
4. B is true and A is False

### Question 24: To enable authentication and authorization support in spring boot rest apis, we can configure a utility class\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. It helps in requiring the user to be authenticated prior to accessing any configured URL (or all urls) within our application.

1. SecurityManager
2. WebSecurityConfigurerManager
3. WebSecurityManagerData
4. WebSecurityConfigurerAdapter

### Question 25: What is the role of @RequestMapping annotation?

1. The @RequestMapping annotation indicates that a particular class serves the role of a request controller.
2. The @RequestMapping annotation is used to map a URL to either an entire class or a particular handler method.
3. The  @RequestMapping  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.
4. The  @RequestMapping  Annotation is used along with @Autowired annotation to remove the confusion by specifying which exact bean will be wired.