Explore More

Subcription : Premium CDAC NOTES & MATERIAL @99

Contact to Join
Premium Group

Click to Join
Telegram Group

For More E-Notes

Join Our Community to stay Updated

TAP ON THE ICONS TO JOIN!

	codewitharrays.in freelance project available to buy contact on 8007592194	
SR.NO	Project NAME	Technology
1	Online E-Learning Platform Hub	React+Springboot+MySql
2	PG Mates / RoomSharing / Flat Mates	React+Springboot+MySql
3	Tour and Travel management System	React+Springboot+MySql
4	Election commition of India (online Voting System)	React+Springboot+MySql
5	HomeRental Booking System	React+Springboot+MySql
6	Event Management System	React+Springboot+MySql
7	Hotel Management System	React+Springboot+MySql
8	Agriculture web Project	React+Springboot+MySql
9	AirLine Reservation System / Flight booking System	React+Springboot+MySql
10	E-commerce web Project	React+Springboot+MySql
11	Hospital Management System	React+Springboot+MySql
12	E-RTO Driving licence portal	React+Springboot+MySql
13	Transpotation Services portal	React+Springboot+MySql
14	Courier Services Portal / Courier Management System	React+Springboot+MySql
15	Online Food Delivery Portal	React+Springboot+MySql
16	Muncipal Corporation Management	React+Springboot+MySql
17	Gym Management System	React+Springboot+MySql
18	Bike/Car ental System Portal	React+Springboot+MySql
19	CharityDonation web project	React+Springboot+MySql
20	Movie Booking System	React+Springboot+MySql

	freelance_Project available to buy contact on 8007592194	
21	Job Portal web project	React+Springboot+MySql
22	LIC Insurance Portal	React+Springboot+MySql
23	Employee Management System	React+Springboot+MySql
24	Payroll Management System	React+Springboot+MySql
25	RealEstate Property Project	React+Springboot+MySql
26	Marriage Hall Booking Project	React+Springboot+MySql
27	Online Student Management portal	React+Springboot+MySql
28	Resturant management System	React+Springboot+MySql
29	Solar Management Project	React+Springboot+MySql
30	OneStepService LinkLabourContractor	React+Springboot+MySql
31	Vehical Service Center Portal	React+Springboot+MySql
32	E-wallet Banking Project	React+Springboot+MySql
33	Blogg Application Project	React+Springboot+MySql
34	Car Parking booking Project	React+Springboot+MySql
35	OLA Cab Booking Portal	React+NextJs+Springboot+MySql
36	Society management Portal	React+Springboot+MySql
37	E-College Portal	React+Springboot+MySql
38	FoodWaste Management Donate System	React+Springboot+MySql
39	Sports Ground Booking	React+Springboot+MySql
40	BloodBank mangement System	React+Springboot+MySql

41	Bus Tickit Booking Project	React+Springboot+MySql
42	Fruite Delivery Project	React+Springboot+MySql
43	Woodworks Bed Shop	React+Springboot+MySql
44	Online Dairy Product sell Project	React+Springboot+MySql
45	Online E-Pharma medicine sell Project	React+Springboot+MySql
46	FarmerMarketplace Web Project	React+Springboot+MySql
47	Online Cloth Store Project	React+Springboot+MySql
48	Train Ticket Booking Project	React+Springboot+MySql
49	Quizz Application Project	JSP+Springboot+MySql
50	Hotel Room Booking Project	React+Springboot+MySql
51	Online Crime Reporting Portal Project	React+Springboot+MySql
52	Online Child Adoption Portal Project	React+Springboot+MySql
53	online Pizza Delivery System Project	React+Springboot+MySql
54	Online Social Complaint Portal Project	React+Springboot+MySql
55	Electric Vehical management system Project	React+Springboot+MySql
56	Online mess / Tiffin management System Project	React+Springboot+MySql
57		React+Springboot+MySql
58		React+Springboot+MySql
59		React+Springboot+MySql
60		React+Springboot+MySql

Spring Boot + React JS + MySQL Project List

Sr.No	Project Name	YouTube Link
1	Online E-Learning Hub Platform Project	https://youtu.be/KMjyBaWmgzg?si=YckHuNzs7eC84-IW
2	PG Mate / Room sharing/Flat sharing	https://youtu.be/4P9clHg3wvk?si=4uEsi0962CG6Xodp
3	Tour and Travel System Project Version 1.0	https://youtu.be/-UHOBywHaP8?si=KHHfE_A0uv725f12
4	Marriage Hall Booking	https://youtu.be/VXz0kZQi5to?si=llOS-QG3TpAFP5k7
5	Ecommerce Shopping project	https://youtu.be/vJ_C6LkhrZ0?si=YhcBylSErvdn7paq
6	Bike Rental System Project	https://youtu.be/FIzsAmIBCbk?si=7ujQTJqEgkQ8ju2H
7	Multi-Restaurant management system	https://youtu.be/pvV-pM2Jf3s?si=PgvnT-yFc8ktrDxB
8	Hospital management system Project	https://youtu.be/IynIouBZvY4?si=CXzQs3BsRkjKhZCw
9	Municipal Corporation system Project	https://youtu.be/cVMx9NVyI4I?si=qX0oQt-GT-LR_5jF
10	Tour and Travel System Project version 2.0	https://youtu.be/_4u0mB9mHXE?si=gDiAhKBowi2gNUKZ

Sr.No	Project Name	YouTube Link
11	Tour and Travel System Project version 3.0	https://youtu.be/Dm7nOdpasWg?si=P_Lh2gcOFhlyudug
12	Gym Management system Project	https://youtu.be/J8_7Zrkg7ag?si=LcxV51ynfUB7OptX
13	Online Driving License system Project	https://youtu.be/3yRzsMs8TLE?si=JRI_z4FDx4Gmt7fn
14	Online Flight Booking system Project	https://youtu.be/m755rOwdk8U?si=HURvAY2VnizIyJlh
15	Employee management system project	https://youtu.be/lD1iE3W_GRw?si=Y_jv1xV_BljhrD0H
16	Online student school or college portal	https://youtu.be/4A25aEKfei0?si=RoVgZtxMk9TPdQvD
17	Online movie booking system project	https://youtu.be/Lfjv_U74SC4?si=fiDvrhhrjb4KSlSm
18	Online Pizza Delivery system project	https://youtu.be/Tp3izreZ458?si=8eWAOzA8SVdNwlyM
19	Online Crime Reporting system Project	https://youtu.be/0UlzReSk9tQ?si=6vN0e70TVY1GOwPO
20	Online Children Adoption Project	https://youtu.be/3T5HC2HKyT4?si=bntP78niYH802I7N

Spring Core MCQ and Answers Explained

Q#1. Which one is incorrect about the Spring Core module?

```
(a) Using Spring Core module, we can develop a standalone application.

(b) The Spring Core module can be used with other modules like Spring MVC, Spring

(c) The Spring Core module provides Spring Containers.

(d) To develop a Spring MVC application, the Spring Core module is not required.
```

Answer: (d) To develop a Spring MVC application, the Spring Core module is not required.

Explanation: The Spring Core module is always required to develop any type of Spring based application. It provides Spring Containers which are always required in order to work with a Spring based application.

Q#2. Which type of dependency can a Spring Container inject?

```
(a) Primitive Type

(b) Collection Type

(c) Reference Type (User defined Type)

(d) All of the above
```


Answer: (d) All of the above

Explanation: Spring container can inject dependencies if the variable's data type of dependent class is any one of the above. For more details, kindly go through the dependency injection article.

Q#3. Consider below four components A, B,C and D with @order annotations applied:

```
@Component
@Order(-1)
public class A { }

@Component
@Order(5)
public class D { }

@Component
@Order(-24)
public class C { }

@Component
@Order(5)
public class B { }
```

What will be the correct order of execution of components?

(a) A B C D

(b) C A D B

(c) C A B D

(d) B D A C

Answer: (c) C A B D

Explanation: Execution order will be as: First components with order value of negative number. Then components with order value of positive number.
Then no order value components in alphabetical order of their names.

Q#4. How can we implement dependency injection in Spring?

```
(a) By using XML configuration.

(b) By using annotations.

(c) By using XML configuration and annotations both.

(d) By implementing interfaces provided by the Spring Framework.
```

Answer: (c) By using XML configuration and annotations both.

Explanation: The @Configuration annotation is used to indicate that a class contains Spring bean definitions. It enables the creation of beans using annotations, allowing developers to define and configure beans directly in Java code rather than XML configuration.

Q#5. What is the difference between @Component and @Bean annotations in Spring?

```
(a) @Component is used for defining singleton beans, while @Bean is used for proto

(b) @Component is used for auto-detection of beans, while @Bean is used to define

(c) @Component is used for defining beans in XML configuration, while @Bean is use

(d) @Component is used for defining controller beans, while @Bean is used for serv
```


Answer: (b) @Component is used for auto-detection of beans, while @Bean is used for prototype beans

Explanation: The @Component annotation is used for auto-detection and automatic registration of beans, while the @Bean annotation is used for manual bean definition in Java-based configuration. @Component is typically used for singleton beans, while @Bean can be used for both singleton and prototype beans.

Q#6. How can we enable component scanning in Spring?

(a) By adding <component-scan> in XML configuration.

(b) By using the @ComponentScan annotation on a configuration class.

(c) By enabling the 'component-scan' property in application.properties.

(d) By using the @Autowired annotation on a field.

Answer: (b) By using the @ComponentScan annotation on a configuration class.

Explanation: The @ComponentScan annotation is used to enable component scanning in Spring. It is generally applied on a configuration class and specifies the base package(s) to scan for components. Component scanning allows Spring to automatically detect and register beans based on annotations such as @Component, @Service, or @Repository etc.

Q#7. Suppose there are two classes Vehicle and Engine associated with a 'HAS-A' relationship. In the context of Spring dependency injection, which one is correct?

(a) The Engine will be a target class, while Vehicle will be a dependent class.

(b) The dependency injection concept can't be applied between the Vehicle and Engi

(c) The Vehicle will be a target class, while Engine will be a dependent class.

```
(d) None of the above.
```

Answer: (c) The Vehicle will be a target class, while Engine will be a dependent class.

Explanation: In the context of dependency injection, there are two concepts: target bean and dependent bean. A bean that takes support of other bean is called the target bean, whereas a bean that acts as a helper/supporting bean is called a dependent bean. For more details, kindly go through dependent & target beans in dependency injection.

Q#8. Which is the correct statement about Spring Container?

```
(a) It is a software program.

(b) It supports dependency injection and autowiring.

(c) It manages the bean life cycle.

(d) All of the above
```


Answer: (d) All of the above

Explanation: The Spring container is a software program that manages the whole life cycle of a bean from its creation to destruction including dependency injection and autowiring.

Q#9. Which statement is incorrect about Spring Bean?

```
(a) A Java class whose object is created and managed by Spring container is called

(b) Spring bean can be a pre-defined or user-defined or third party supplied class

(c) Spring bean can be a POJO class or component class or Java bean class.
```

```
(d) Spring bean can be any Java class or interface.
```

Answer: (d) Spring bean can be any Java class or interface. For more details on Spring Bean, kindly visit 'What is Spring Bean?'.

Explanation: We can make any Java class as a Spring Bean, except abstract class and interface.

Q#10. Suppose we have a class Employee and two of its sub-classes PermanentEmployee and ContractEmployee. We want Spring container to inject PermanentEmployee in order to avoid ambiguity issue between two classes of the same type. Which code snippet can raise the ambiguity issue?


```
(a) @Primary
    @Component
    public class PermanentEmployee extends Employee { ..... }

    @Component
    public class ContractEmployee extends Employee { ..... }
```

```
(b) <bean id="pe" class="com.dev.entity.PermanentEmployee" primary="true"/>
    <bean id="ce" class="com.dev.entity.ContractEmployee" />
```

```
(c) public class XYZCompany{
        @Autowired
        @Primary("contractEmployee")
        private Employee employee;
    }
```

```
(d) public class XYZCompany{
        @Autowired
        @Qualifier("contractEmployee")
        private Employee employee;
    }
```

Answer: (c)

Explanation: Option (c) has an incorrect use of @Primary annotation. In this situation, this can be used in sub-classes definitions.

Q#11. What is the purpose of the @PreDestroy annotation in Spring?

```
(a) It is used to indicate a method that should be called after bean initializatio

(b) It is used to specify the order of bean initialization.

(c) It is used to indicate a method that should be called before destroying the be

(d) It is used to specify a method that should be called after constructing the pr
```

Answer: (c) It is used to indicate a method that should be called before destroying the bean.

Explanation: The @PreDestroy annotation is used to mark a method that should be invoked before destroying the bean.

Q#12. Which one of the following is an incorrect use of the @Qualifier annotation in Spring?

```
(a) public class XYZCompany{
        private Employee employee;

        @Autowired
        public XYZCompany(@Qualifier("contractEmployee") Employee employee){
            this.employee=employee;
        }
    }

(b) public class XYZCompany{


        @Autowired
        @Qualifier("contractEmployee")
        private Employee employee;
    }

(c) public class XYZCompany{
```

```
        private Employee employee;

        @Autowired
        @Qualifier("contractEmployee")
        public XYZCompany(Employee employee){
            this.employee=employee;
        }
    }
```

```
(d) public class XYZCompany{

        @Autowired
        @Qualifier("permanentEmployee")
        private ContractEmployee employee;
    }
```

Answer: (c)
Explanation: If we have @Autowired annotation at the constructor level, we can use @Qualifier annotation at the parameter of the constructor, but not above the constructor. You may refer autowiring in spring to get more idea on this.

Q#13. Spring provides multiple ways to define the scope of a bean. Which one of the following is not the correct way of defining a 'session' scope?


```
(A) @SessionScope
    public class Employee{
          // some properties & methods
    }
```

```
(B) <bean id="employee" class="com.test.Employee" Scope="session"/>
```

```
(C) @Scope("session")
    public class Employee{
          // some properties & methods
    }
```

```
(D) <bean id="employee" class="com.test.Employee" scope="session"/>
```

Answer: (b)

Explanation: Option (b) has an incorrect attribute 'Scope'. It should be 'scope'. All Other options are correct.

Q#14. How can you define a bean using @Bean annotation in Spring?

```
(a) By using the @Bean annotation on a method in a @Configuration class.

(b) By using the @Bean annotation on a method in a @Component class.

(c) By using the @Autowired and @Bean annotation on a field.

(d) By using the @Autowired and @Bean annotation on a method.
```


Answer: (a) By using the @Bean annotation on a method in a @Configuration class.

Explanation: The @Bean annotation is used to define a bean in Spring while using Java-based configuration. It is generally used on a method within a class annotated with @Configuration.

Q#15. How to mark a class as a Spring Bean?

```
(a) By using <bean> tag in the XML configuration file.

(b) By using @Component annotation in a Java class.

(c) By using @Configuration annotation in a Java class and @Bean annotation in a m
```

```
(d) All of the above
```

Answer: (d) All of the above

Explanation: All of the above methods are used to mark a Java class as a Spring Bean. For detailed explanation visit a separate section on How to mark a class as Spring Bean?.

Q#16. We can use @Autowired annotation at multiple places of a class. Consider some of the places given below.

1) Field 2) Setter Method 3) Parameter field 4) Normal Method

Which of the following is the correct combination of places where we can apply @Autowired annotation?

```
(a) 1 and  2

(b) 1 and  2 and  3

(c) 1 and 2 and 4

(d) All places
```


Answer: (d) All places

Explanation: We can apply @Autowired annotation in field, setter method, parameterized constructor, and normal method. Here, the method signature of the normal method must be same as the setter method.

Q#17. Consider the following bean definition:

```
<bean id="product" class="com.dev.test.ProductBean"/>
```

Which of the below bean scope is applied in this bean?

```
(A) Session

(B) Request

(C) Prototype

(D) Singleton
```

Answer: (d) Singleton
Explanation: When the scope is not specified, the session scope is applicable by default.

Q#18. What is the purpose of the @Configuration annotation in Spring?

```
(a) It works as a substitute of @ComponentScan.

(b) It marks a class as a source of Bean's definitions.

(c) It enables autowiring in Spring.

(d) It specifies the primary bean for a given type.
```


Answer: (b) It marks a class as a source of Bean's definitions.

Explanation: Using @Configuration annotation at any Java class represents that Spring container will consider it as a source of Bean's definitions. It also defines a configuration class for Spring.

Q#19. What is the purpose of the @Profile annotation in Spring?

```
(a) It marks a bean as eligible for autowiring.

(b) It associates a bean to a particular profile.
```

```
(c) It enables caching for a specific bean.

(d) It specifies the scope of a bean.
```

Answer: (b) It associates a bean to a particular profile.

Explanation: By using @Profile annotation, we can make a bean belong to a particular profile. The @Profile annotation simply takes the names of one or multiple profiles.

Q#20. Which of the following annotation will you use after constructing the properties of the class in the context of the life cycle methods?

```
(A) @PreDestroy

(B) @PostConstruct

(C) @BeforeDestroy

(D) @AfterConstruct
```


Answer: (b) @PostConstruct

Explanation: PostConstructs means after constructing the properties of the class. Hence @PostConstruct is the correct answer. For complete details, kindly go through the separate detailed article on Spring Bean Life Cycle methods.

Q#21. What is incorrect about the BeanFactory in Spring?

```
(a) It is an interface, also known as a Spring container.

(b) It is used to manage the life cycle of a spring bean.

(c) It is used to handle HTTP requests in a Spring MVC application.
```

```
(d) It is responsible for creating and managing Spring beans.
```

Answer: (c) It is used to handle HTTP requests in a Spring MVC application.

Explanation: The BeanFactory interface, being a basic Spring Container in Spring is responsible for creating and managing Spring beans, including managing the life cycle of a spring bean.

Q#22. Which is the incorrect statement about the ApplicationContext and BeanFactory in Spring?

```
(a) The ApplicationContext is a sub-interface of the BeanFactory.

(b) The ApplicationContext provides more advanced features than the BeanFactory.

(c) The BeanFactory supports XML-based configuration, while ApplicationContext sup

(d) The ApplicationContext is used for testing purposes, while BeanFactory is used
```


Answer: (d) The ApplicationContext is used for testing purposes, while BeanFactory is used in production environments.

Explanation: Since ApplicationConext provides more advanced features than the BeanFactory, it is most often used in production environments.

Q#23. What is the purpose of the BeanPostProcessor interface in Spring?

```
(a) It is used to define a custom scope for a bean.

(b) It is responsible for initializing beans after instantiation.

(c) It is used to customize the bean creation process.
```

```
(d) It is responsible for managing life cycle of beans.
```

Answer: (c) It is used to customize the bean creation process.

Explanation: The BeanPostProcessor interface in Spring allows customization of the bean creation process. It provides hooks for performing operations before and after bean initialization, enabling developers to modify or enhance bean instances.

Q#24. You are using XML driven approach to specify bean definitions. Your @Autowired or @Required annotations are not working. What entry will you add in your XML configuration file?

```
(a) <bean:annotation-config/>

(b) <context:annotation-config/>

(c) <context:config-annotation/>

(d) <context:component-scan/>
```


Answer: (b) <context:annotation-config/>

Explanation: <context: annotation-config> enables functioning of some annotations such as @Required, @Autowired, @PostConstruct, @PreDestroy, @Resource.

Q#25. How can you define a bean's scope in XML configuration?

```
(a) By using the "scope" attribute of the <bean> element.

(b) By using the @Scope annotation on a bean class.
```

```
(c) By using the <scope> element in XML configuration.

(d) By using all of the approaches mentioned above
```

Answer: a) By using the “scope” attribute of the <bean> element.

Explanation: In XML configuration, the scope of a bean can be defined by using the “scope” attribute of the <bean> element. The attribute value can be set to “request” in order to specify request scope.

Q#26. How can you define a bean’s initialization and destruction methods of bean life cycle in XML configuration?

```
(a) By using the "init-method" and "destroy-method" attributes of the <bean> eleme

(b) By using the @PostConstruct and @PreDestroy annotations.

(c) By using the <init-method> and <destroy-method> elements in XML configuration.

(d) By using the @Bean annotation with initMethod and destroyMethod attributes.
```


Answer: (a) By using the “init-method” and “destroy-method” attributes of the <bean> element.

Explanation: In XML configuration, a bean’s initialization and destruction methods can be defined using the “init-method” and “destroy-method” attributes of the <bean> element. These methods will be called after bean instantiation and before bean destruction respectively.

Q#27. Which of the following is not a Spring module?

```
(a) Spring Core

(b) Spring MVC
```

```
(c) Spring Boot

(d) Spring Framework
```

Answer: (d) Spring Framework

Explanation: The Spring Framework is the broad framework that includes various modules like Spring Core, Spring MVC, and Spring Boot. It is not a module itself but rather the complete ecosystem.

Q#28. Maxton is working on his Spring-based college project. He wants to use annotations in his project. He wants an annotation that can be used to replace the XML <bean> tag. And one more annotation that can make a method to produce a bean that can be managed by Spring Container. Help him in identifying the suitable annotations to complete his task.


```
(a) @Bean, @Bean

(b) @Bean, @Component

(c) @Bean, @Configuration

(d) @Configuration, @Bean
```

Answer: (d) @Configuration, @Bean

Explanation: The class annotated with @Configuration serves as a source of bean definitions. This annotation is used to indicate that a class declares one or more @Bean methods. @Bean annotation is applied to a method within a @Configuration class. The method annotated with @Bean will return an object that Spring will register as a bean in the application context. Hence, to replace the XML <bean> tag and manage the beans using annotations, **@Configuration** and **@Bean** are the appropriate annotations.

https://www.youtube.com/@codewitharrays

https://www.instagram.com/codewitharrays/

https://t.me/codewitharrays Group Link: https://t.me/ccee2025notes

+91 8007592194 +91 9284926333

codewitharrays@gmail.com

https://codewitharrays.in/project