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 MCQ and Answers Explained

Q#1. Which annotation is not used to inject dependencies into a Spring bean?

```
(a) @Inject

(b) @Bean

(c) @Autowired

(d) @Resource
```

Answer: (b) @Bean

Explanation: '@Autowired', 'Resource', and '@Inject' all are used to inject dependencies. They use different implementations of post processor classes. They all behave almost identically. The '@Autowired' and '@Inject' annotation use the ~~'AutowiredAnnotationBeanPostProcessor'~~ to inject dependencies. '@Autowired' and '@Inject' can be used interchangeably to inject dependencies. Furthermore, the '@Resource' annotation uses the 'CommonAnnotationBeanPostProcessor' to inject dependencies.

Q#2. What is Spring Core?

```
(a) A Java framework for creating user interfaces

(b) An IDE for developing Spring based applications

(c) A sub-module of Spring framework

(d) A scripting language of the Spring Framework
```

Answer: (c) A sub-module of Spring framework

Explanation: Spring Core is a sub-module of Spring framework that provides various features and functionalities that helps in building enterprise applications using the Java programming language. It is the crucial sub-module of the Spring Framework. Some of the most common features that Spring Core provides are: IoC (Inversion of control) and DI (dependency injection).

Q#3. Which of the following annotations is NOT used to specify that a class is a Spring bean?

```
(a) @Service

(b) @Qualifier

(c) @Component

(d) @Controller
```

Answer: (c) @Qualifier

Explanation: The @Qualifier annotation is used with @Autowired in the context of autowiring. All other options are the generic stereotype annotations that can be used to specify that the class is a Spring bean.

Q#4. Which of the following is INCORRECT about dependency injection in Spring?

```
(a) It helps in achieving loose coupling between objects.

(b) It helps in testing the module by injecting the dependent Mock Objects.

(c) It increases the scope of code reusability.

(d) It helps in managing database connections.
```


Answer: (d) It helps in managing database connections.

Explanation: Dependency injection allows components to be loosely coupled, making them easier to test and maintain. It helps in achieving better code reusability. You can go through the benefits of

dependency injection.

Q#5. Consider the below @Value annotation, which is written using SpEL (Spring Expression Language):

```
@Value("#{myBean.myProperty != null ? myBean.myProperty : 'default'}")
```

Which is the shorter representation of the above annotation using Elvis operator?

```
(a) @Value("#{myBean.myProperty ?: null}")

(b) @Value("#{myBean.myProperty ? myBean.myProperty : 'default'}")

(c) @Value("#{myBean.myProperty ? null}")

(d) @Value("#{myBean.myProperty ?: 'default'}")
```


Answer: (d) @Value("#{someBean.someProperty ?: 'default'}")

Explanation: Using the Elvis operator (?:) , we can shorten the ternary operator syntax for the case above. It is used in the Groovy language and also available in SpEL.

Q#6. In Spring's XML configuration file, which XML tag is used to define a bean?

```
(a) <component>

(b) <service>

(c) <bean>
```

```
(d) <dependency>
```

Answer: (c) <bean>

Explanation: In Spring XML configuration, the <bean> tag is used to define a bean. It specifies the class or interface to be instantiated as a bean and provides additional configuration options.

Q#7. What is the purpose of the @Autowired annotation in Spring?

```
(a) It marks a bean for automatic injection

(b) It configures the transaction management

(c) It specifies the automatic ordering of bean initialization

(d) It creates an object of a Spring bean
```


Answer: (a) It marks a bean for automatic injection

Explanation: The @Autowired annotation marks a field, constructor, or setter method for automatic dependency injection by Spring. It allows Spring to automatically associate the dependencies without the need for explicit configuration.

Q#8. Which annotation is used to enable Spring's aspect-oriented programming (AOP) support?

```
(a) @Aspect

(b) @AspectJ

(c) @EnableAop
```

```
(d) @EnableAspectJAutoProxy
```

Answer: (d) @EnableAspectJAutoProxy

Explanation: The @EnableAspectJAutoProxy annotation is used to enable Spring's AOP support. It activates the proxy-based AOP infrastructure, allowing the use of aspect-oriented programming in Spring applications. Here is the step by step tutorial on 'How to implement AOP in a Spring/Spring Boot based Application?'.

Q#9. What is the purpose of the @Qualifier annotation in Spring?

```
(a) It specifies the order of bean initialization among multiple beans

(b) It marks a bean for automatic injection

(c) It resolves ambiguity when multiple beans of the same type are present

(d) It marks a class to become a Spring bean
```


Answer: c) It resolves ambiguity when multiple beans of the same type are present

Explanation: The @Qualifier annotation is used to resolve ambiguity when multiple beans of the same type are present.

Q#10. Which annotation is used to declare a class as a Spring bean in a Java based configuration class?

```
(a) @Service

(b) @Bean

(c) @Component
```

```
(d) @Autowired
```

Answer: (b) @Bean

Explanation: In a Java based configuration class, we use the @Bean annotation to declare a method that returns an instance of a bean. The method is invoked by Spring to create the bean and manage its lifecycle.

Q#11. Which of the following tag is required to enable component scanning in Spring XML configuration?

```
(a) <context:scan package="com.example" />

(b) <scan-components base-package="com.example" />

(c) <component-scan base-package="com.example" />

(d) <scan: component base-package="com.example" />
```


Answer: (c) <component-scan base-package=”com.example” />

Explanation: In order to enable component scanning in Spring XML configuration, we use the <component-scan> tag with the attribute “base-package” to specify the package(s) to scan for annotated components.

Q#12. Which is the incorrect purpose of the @Value annotation in Spring?

```
(a) It is generally used for injecting values into configuration variables

(b) It supports Spring Expression Language (SpEL)

(c) It is used to assign default values to variables and method arguments
```

```
(d) It is used to inject dependencies as part of autowiring
```

```
Answer: (d) It is used to inject dependencies as part of autowiring
```

Explanation: The @Value annotation is used in all purposed mentioned in options a, b, and c. It is not used to inject dependencies as part of autowiring.

Q#13. Which annotation is used to enable transaction management in Spring?

```
(a) @Transactional
```

```
(b) @EnableTransactionManagement
```

```
(c) @ConfigureTransaction
```

```
(d) @EnableTransactions
```


Answer: (b) @EnableTransactionManagement

Explanation: Spring Transaction support is not enabled by default. For that reason, we use the @EnableTransactionManagement annotation in a @Configuration annotated class to enable Transaction related support in Spring. It activates the infrastructure for declarative transaction management in Spring. For complete details with examples, you may refer Spring Transaction Annotations.

Q#14. How can you retrieve a bean from the Spring ApplicationContext using Java code?

```
(a) Using the ApplicationContext.getBean() method
```

```
(b) Using the @Bean annotation
```

```
(c) Using the @Autowired annotation
```

```
(d) Using the @Resource annotation
```

Answer: a) Using the ApplicationContext.getBean() method

Explanation: You can retrieve a bean from the Spring ApplicationContext using the getBean() method. It takes the bean name or its class as a parameter.

Q#15. Which scope in Spring ensures that object exists between every request in an HTTP session?

```
(a) Singleton

(b) Prototype

(c) Request

(d) Session
```


Answer: (d) Session

Explanation: In the Session scope, the object exists between every request in an HTTP session.

Q#16. How can you define property placeholders in Spring XML configuration?

```
(a) Using the <context:property-placeholder>  tag

(b) Using the <context:value> tag

(c) Using the <context:property> tag

(d) Using the <context:placeholder> tag
```

Answer: (a) Using the <context: property-placeholder> tag

Explanation: You can define property placeholders in Spring XML configuration using the <context: property-placeholder> tag. It allows you to specify properties from external files or system properties.

Q#17. In order to enable the Spring expression language (SpEL) in Spring beans, Which annotation is used?

```
(a) @ExpressionLanguage

(b) @EnableSpEL

(c) @SpEL

(d) @Value
```

Answer: (d) @Value

Explanation: The @Value annotation in Spring supports the Spring expression language (SpEL). It offers you to evaluate SpEL expressions and inject the result into a bean property.

Q#18. If you want to configure a bean for lazy initialization in Spring, how can you configure a bean to be initialized lazily?

```
(a) Using the @Lazy annotation

(b) Setting the "lazy-init" attribute to "true" in XML configuration

(c) Using the @PostConstruct annotation

(d) Setting the "scope" attribute to "lazy" in XML configuration
```

Answer: (a) Using the @Lazy annotation

Explanation: You can use the @Lazy annotation to configure a bean for lazy initialization in Spring. It enables that the bean is created only when it is first requested.

Q#19. To be eligible for Spring's event handling mechanism, which interface should a class implement?

```
(a) EventListener

(b) ApplicationListener

(c) EventHandler

(d) EventSubscriber
```

Answer: (b) ApplicationListener

Explanation: A class should implement the ApplicationListener interface in order to participate in Spring's event handling mechanism. It permits the class to receive and handle application events.

Q#20. How can you register an event listener in Spring XML configuration?

```
(a) Using the <listener> tag

(b) Using the <event-listener> tag

(c) Using the <application-listener> tag

(d) Using the <bean> tag with the ApplicationListener interface
```

Answer: (d) Using the <bean> tag with the ApplicationListener interface

Explanation: You can register an event listener in Spring XML configuration by defining a bean using the <bean> tag and implementing the ApplicationListener interface in the class.

Q#21. How can you enable asynchronous method execution in Spring?

```
(a) Using the @Async annotation

(b) Using the <async-method-execution> tag

(c) Using the @EnableAsync annotation

(d) Setting the "async" attribute to "true" in XML configuration
```

Answer: (c) Using the @EnableAsync annotation

Explanation: To enable asynchronous method execution in Spring, you can use the @EnableAsync annotation. It activates Spring's support for executing methods asynchronously.

Q#22. What is the purpose of the @Qualifier annotation in Spring?

```
(a) It specifies the order of bean initialization

(b) It marks a bean for automatic injection

(c) It resolves ambiguity when multiple beans of the same type are present

(d) It creates an object for the annotated bean
```


Answer: (c) It resolves ambiguity when multiple beans of the same type are present

Explanation: The @Qualifier annotation is used in conjunction with @Autowired or @Inject or @Resource to fix ambiguity issue when multiple beans of the same type are present for injection.

Q#23. How can you enable Spring's support for method-level caching?

```
(a) Using the @Cacheable annotation

(b) Using the <cacheable> tag
```

```
(c) Using the @EnableCaching annotation

(d) Setting the "cacheable" attribute to "true" in XML configuration
```

Answer: (c) Using the @EnableCaching annotation

Explanation: To enable Spring's support for method-level caching, you can use the @EnableCaching annotation. It activates the caching facility in Spring.

Q#24. How can you define the order of bean initialization in Spring?

```
(a) Using the <order> tag

(b) Using the @Order annotation

(c) Using the @Priority annotation

(d) Using the <priority> tag
```


Answer: (b) Using the @Order annotation

Explanation: You can use the @Order annotation to define the order of bean initialization in Spring. It allows you to assign a value to indicate the desired order of bean creation. To get complete details of @Order annotation with examples, kindly go through '@Order Annotaion In Spring'.

Q#25. Which is the correct option to enable Spring MVC in a Spring based application?

```
(a) Using the @EnableMvc annotation

(b) Using the <enable-mvc> tag

(c) Using the @EnableWebMvc annotation
```

```
(d) Using the @EnableSpringMvc annotation
```

Answer: (c) Using the @EnableWebMvc annotation

Explanation: To enable the support for Spring MVC, you can use the @EnableWebMvc annotation.

Q#26. Which annotation is used to enable scheduling tasks in Spring?

```
(a) @Scheduled

(b) @EnableScheduling

(c) @TaskScheduler

(d) @Scheduling
```


Answer: (b) @EnableScheduling

Explanation: The @EnableScheduling annotation is used to enable Spring scheduling support. It activates the scheduling facility and permits the use of the @Scheduled annotation.

Q#27. If you want to schedule a method to be executed at fixed intervals in Spring, Which annotation will you use?

```
(a) @FixedRate

(b) @FixedDelay

(c) @Cron
```

```
(d) @FixedInterval
```

Answer: (a) @FixedRate

Explanation: If you want to schedule a method to be executed at fixed intervals, the @FixedRate annotation should be your choice to use. It defines the interval between method invocations in milliseconds.

Q#28. To access a bean property, which SpEL expression is correct?

```
(a) #{beanName.propertyName}

(b) ${beanName.propertyName}

(c) @beanName.propertyName

(d) %beanName.propertyName%
```


Answer: (a) #{beanName.propertyName}

Explanation: SpEL uses the #{} syntax to access bean properties.

Q#29. Which annotation is used to mark a method as an exception handling method in Spring MVC?

```
(a) @ExceptionHandler

(b) @ExceptionHandling

(c) @HandleException
```

```
(d) @ExceptionResolver
```

Answer: (a) @ExceptionHandler

Explanation: The @ExceptionHandler annotation is used to handle exceptions in Spring MVC. It allows you to mark a method as an exception handler method that handle specific exceptions thrown during the request processing.

Q#30. If you want to write global exception handling code that can be applied to multiple controllers in Spring MVC, which annotation will you use?

```
(a) @GlobalExceptionHandling

(b) @GlobalExceptionHandler

(c) @ControllerAdvice

(d) @EnableGlobalExceptionHandling
```


Answer: (c) @ControllerAdvice

Explanation: To configure global exception handling in Spring MVC, you can use the @ControllerAdvice annotation. It offers you to define global exception handling methods throughout multiple controllers.

Q#31. In Spring JDBC, which method will you use to execute a SQL query that can map the results to a Java object?

```
(a) executeQueryForObject()

(b) fetchObject()

(c) executeForObject()
```

```
(d) queryForObject()
```

Answer: (d) queryForObject()

Explanation: The queryForObject() method is used to execute a SQL query and map the results to a Java object in Spring JDBC. It is generally used for retrieving a single result.

Q#32. What is the role of the DispatcherServlet in Spring MVC?

```
(a) It configures the Spring MVC dependencies

(b) It manages the application's database connections

(c) It handles incoming requests and delegates them to controllers

(d) It handles security authentication and authorization.
```


Answer: (c) It handles incoming requests and delegates them to controllers

Explanation: The DispatcherServlet handles incoming requests and delegates them to controllers for processing.

Q#33. Which of the following is helpful in passing data from a controller method to a view in Spring MVC?

```
(a) HttpRequest object

(b) HttpServletResponse object

(c) HttpSession object
```

```
(d) ModelAttributes to the Model object
```

Answer: (d) ModelAttributes to the Model object

Explanation: You can pass data from a controller method to a view in Spring MVC by adding model attributes to the Model object.

Q#34. How can you enable cross-origin resource sharing (CORS) in Spring MVC?

```
(a) Using the @EnableCors annotation

(b) Using the <enable-cors> tag

(c) Using the @CrossOrigin annotation

(d) Using the @EnableCrossOrigin annotation
```


Answer: (c) Using the @CrossOrigin annotation

Explanation: You can use the @CrossOrigin annotation in order to enable cross-origin resource sharing (CORS) in Spring MVC. It allows you to specify the allowed origins, headers, and methods for cross-origin requests.

Q#35. Which of the following is used to hold the model data and view information in Spring MVC?

```
(a) ViewResolver

(b) Model

(c) ModelAttribute
```

```
(d) ModelAndView
```

Answer: (d) ModelAndView

Explanation: We use ModelAndView class to hold the model data and view information in Spring MVC.

Q#36. Which of the following is an incorrect use of Http GET request in a Spring MVC application?

```
(a) @GetMapping(value="/", produces = MediaType.TEXT_PLAIN_VALUE)

(b) @RequestMapping(value = "/emp/{id}", method = GET)

(c) @GetMapping(consumes = {"text/plain", "application/*"})

(d) @GetMapping("/")
```

Answer: (c) @GetMapping(consumes = {“text/plain”, “application/*”})

Explanation: In the option (c), path value is missing.

Q#37. In Spring Security, which annotation is not enabled by @EnableMethodSecurity by default?

```
(a) @PreAuthorize

(b) @PostAuthorize

(c) @PreFilter

(d) @Secured
```

Answer: (d) @Secured

Explanation: In Spring Security 5.6, we can enable annotation-based security using the @EnableMethodSecurity annotation in place of @EnableGlobalMethodSecurity on any @Configuration annotated class. It enables @PreAuthorize, @PostAuthorize, @PreFilter, and @PostFilter by default and also complies with JSR-250. In order to enable @Secured, we need to provide an additional attribute 'securedEnabled': @EnableMethodSecurity(securedEnabled = true)

Q#38. Which advice type is executed regardless of the method execution outcome, including exceptions in Spring AOP?

(a) Before advice

(b) After advice

(c) Around advice

(d) After advice

Answer: (d) After advice

Explanation: The After advice is executed irrespective of the execution flow of the matched method. Whether the method is executed normally or any exception occurs after advice will be executed either way.

Q#39. Which pointcut expression is correct for the below statement?

"Any method with zero or more parameters inside EmpDao class"

```
(a) public * com.dev.dao.EmpDao.*( )
```

```
(b) public * com.dev.dao.*.*(..)
```

```
(c) public * com.dev.dao.EmpDao.*(..)
```

```
(d) public Integer com.dev.dao.EmpDao.*( )
```

Answer: (c) public * com.dev.dao.EmpDao.*(..)

Explanation: In pointcut expressions, ( ) => indicates zero parameter or without parameter, whereas (..) => indicates any number or type of parameters. To know more about pointcut, kindly go through 'examples of pointcut expressions'.

Q#40. In the context of Spring AOP, select the method which is the candidate for below Pointcut Expression?

public * *(Invoice)

```
(a) public Integer saveInvoice(Integer id){….}

(b) public void deleteInvoice(Integer id){….}

(c) public void updateInvoice(Invoice inv){….}

(d) public Invoice getInvoice(Integer id){….}
```


Answer: (c) public void updateInvoice(Invoice inv){….}

Explanation: Option (c) has Invoice datatype in it's parameter.

Q#41. Which one is the another way of writing below annotation in Spring Security?

@Secured("ROLE_MANAGER","ROLE_ADMIN")

```
(a) @PreAuthorize("ROLE_MANAGER","ROLE_ADMIN")

(b) @PreAuthorize("hasRole('ROLE_MANAGER') or hasRole('ROLE_ADMIN')")

(c) @PreAuthorize("hasRole('ROLE_MANAGER') | hasRole('ROLE_ADMIN')")

(d) @PreAuthorize("role('ROLE_MANAGER') or role('ROLE_ADMIN')")
```

Answer: (b) @PreAuthorize("hasRole('ROLE_MANAGER') or hasRole('ROLE_ADMIN')")

Explanation: The @PreAuthorize("hasRole('ROLE_MANAGER') or hasRole('ROLE_ADMIN')") is another way of writing @Secured("ROLE_MANAGER","ROLE_ADMIN"). Furthermore, The @PreAuthorize("hasRole('ROLE_ADMIN')") is equivalent to @Secured("ROLE_ADMIN") and both have the same meaning.

Q#42. Which is a correct cron expression in Spring Scheduling that executes a task every year on Feb 14th 9:00:00 PM, if given day(14th) is Sunday or Tuesday only?

```
(a) 0   0   9   2   14   1,3

(b) 0   0   9   2   14   SUN,TUE

(c) 0   0   21   14   2   SUN,TUE

(d) 0   0   9   14   2   1,3
```

Answer: (c) 0 0 21 14 2 SUN,TUE

Explanation: Except option (c), all other options are incorrect in either month, or date, or days of week place.

Q#43. As part of the cron expressions improvement in Spring 5.3, the below cron expression can also be written as:

```
0 0 0 * * 0
```

```
(a) @monthly

(b) @weekly

(c) @yearly
```

```
(d) @midnight
```

Answer: (b) @weekly

Explanation: Spring now supports some macros, which represent commonly used sequences in order to improve readability. Even we can use these macros rather than writing the six-digit values. For more details, kindly go through 'macros support in Spring Scheduling'.

Q#44. Which is a correct cron expression in Spring Scheduling that executes a task every year to wish happy birthday on 24th March.

```
(a) 0   0   0   3   24   *

(b) 0   0   24   3   0   *

(c) 0   0   0   24   3   *

(d) 0   0   0   24   3   0
```

Answer: (c) 0 0 0 24 3 *

Explanation: Except option (c), all other options are incorrect in either month, or date, or year place.

Q#45. In Spring JDBC, what does the NamedParameterJdbcTemplate class offer?

```
(a) Support for executing SQL statements with positional  parameters

(b) Support for executing SQL statements with named parameters

(c) Support for executing stored procedures
```

```
(d) Support for executing batch updates
```

Answer: (b) Support for executing SQL statements with named parameters

Explanation: The NamedParameterJdbcTemplate class in Spring JDBC offers support for executing SQL statements with named parameters. It allows you to specify parameters by name instead of their positional index.

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