1. IOC and DI means?

IoC (Inversion of Control) is a principle in software engineering which transfers the control of objects or portions of a program to a container or framework. We most often use it in the context of object-oriented programming.

DI (Dependency Injection) is a sub-type of IoC and is implemented by constructor injection, setter injection or Interface injection. Connecting objects with other objects, or “injecting” objects into other objects, is done by an assembler rather than by the objects themselves.

2. Spring Bean Scopes?

Spring Bean Scopes allows us to have more granular control of the bean instances creation. Sometimes we want to create bean instance as singleton but in some other cases we might want it to be created on every request or once in a session.

singleton: only one instance of the spring bean will be created for the spring container. This is the default spring bean scope. While using this scope, make sure bean doesn’t have shared instance variables otherwise it might lead to data inconsistency issues.

prototype: A new instance will be created every time the bean is requested from the spring container.

request: This is same as prototype scope; however, it’s meant to be used for web applications. A new instance of the bean will be created for each HTTP request.

session: A new bean will be created for each HTTP session by the container.

global-session: This is used to create global session beans for Portlet applications.

3. What does @SpringBootApplication do?

Spring Boot @SpringBootApplication annotation is used to mark a configuration class that declares one or more @Bean methods and triggers auto-configuration and component scanning. It’s same as declaring a class with @Configuration, @EnableAutoConfiguration and @ComponentScan annotations.

4. What is Spring AOP? Where and how to use it?

Spring AOP enables Aspect-Oriented Programming in spring applications. In AOP, aspects enable the modularization of concerns such as transaction management, logging or security that cut across multiple types and objects (often termed crosscutting concerns).

AOP provides the way to dynamically add the cross-cutting concern before, after or around the actual logic using simple pluggable configurations. It makes easy to maintain code in the present and future as well. You can add/remove concerns without recompiling complete source code simply by changing configuration files (if you are applying aspects using XML configuration).

5. What is Singleton and where to use it?

Singleton pattern is a design pattern which restricts a class to instantiate its multiple objects. It is nothing but a way of defining a class. Class is defined in such a way that only one instance of the class is created in the complete execution of a program or project. It is used where only a single instance of a class is required to control the action throughout the execution. A singleton class shouldn't have multiple instances in any case and at any cost. Singleton classes are used for logging, driver objects, caching and thread pool, database connections.

6. What is Spring Boot Actuator and Where to use it?

Spring Boot Actuator is a sub-project of the Spring Boot Framework. It includes several additional features that help us to monitor and manage the Spring Boot application. It contains the actuator endpoints (the place where the resources live). We can use HTTP and JMX endpoints to manage and monitor the Spring Boot application. If we want to get production-ready features in an application, we should use the Spring Boot actuator. Spring Boot enables security for all actuator endpoints. It uses form-based authentication that provides user Id as the user and a randomly generated password. We can also access actuator-restricted endpoints by customizing basicauth security to the endpoints.

7. What is the primary difference between Spring and Spring Boot?

Spring Framework provides comprehensive infrastructure support for developing Java applications. It contains some good functions, such as dependency injection and out-of-the-box modules, including Spring JDBC, Spring MVC, Spring Security, Spring AOP, and Spring ORM, Spring Test. Spring Boot is basically an extension of the Spring Framework. It eliminates the XML configuration required to set up Spring applications, paving the way for a faster and more efficient development ecosystem.

8. Why to use VCS?

Version control is important to keep track of changes and keep every team member working on the right version. You should use version control software for all code, files, and assets that multiple team members will collaborate on.

9. What are SOLID Principles? Give sample usages in Java.

SOLID principles are object-oriented design concepts relevant to software development. SOLID is an acronym for five other class-design principles: Single Responsibility Principle, Open-Closed Principle, Liskov Substitution Principle, Interface Segregation Principle, and Dependency Inversion Principle.

In Single Responsibility Principle every class in Java should have a single job to do. To be precise, there should only be one reason to change a class. In the given example, we have a Person class. It has single responsibility to store its specific information.

```java
public class Person
{
    private Long personId;
    private String firstName;
    private String lastName;
    private String age;
    private List<Account> accounts;
}
```

10. What is RAD model?

The Rapid Application Development (or RAD) model is based on prototyping and iterative model with no (or less) specific planning. In general, RAD approach to software development means putting lesser emphasis on planning tasks and more emphasis on development and coming up with a prototype. RAD puts clear focus on prototyping, which acts as an alternative to design specifications. This means that RAD works well wherever there's a greater focus on user interface rather than non-GUI programs. The RAD model includes agile method and spiral model.

11. What is Spring Boot starter? How is it useful?

Spring Boot Starters are dependency descriptors that can be added under the <dependencies> section in pom.xml. There are around 50+ Spring Boot Starters for different Spring and related technologies. These starters give all the dependencies under a single name. For example, if you want to use Spring Data JPA for database access, you can include spring-boot-starter-data-jpa dependency.

12. What is Caching? How can we achieve caching in Spring Boot?

In Spring, the cache abstraction is a mechanism that allows consistent use of various caching methods with minimal impact on the code. The cache abstraction mechanism applies to Java methods. The main objective of using cache abstraction is to reduce the number of executions based on the information present in the cache. It applies to expensive methods such as CPU or IO bound.

To enable caching, Spring makes good use of annotations, much like enabling any other configuration level feature in the framework. We can enable the caching feature simply by adding the @EnableCaching annotation to any of the configuration classes.

13. What & How & Where & Why to logging?

Logging is keeping a record of all data input, processes, data output, and final results in a program. This is part of a much more grand, complex process, though, so you want to program with a clear goal in mind, and not try to do several programming disciplines at once. There are several kinds of logging. You may log every operation of an application, log only when errors occur, or log critical operations done by a user, especially if you want to have control of who's doing what for audit purposes.

Applications can log at a code level for debugging or at a user level for audits and forbidden access register. Every operating system has routines or services running on the background with the task of taking notes of everything that is happening. This way a SysAdmin can detect security breaches, malfunctions etc.

14. What is Swagger? Have you implemented it using Spring Boot?

Swagger is an Interface Description Language for describing RESTful APIs expressed using JSON. Swagger is used together with a set of open-source software tools to design, build, document, and use RESTful web services. Swagger includes automated documentation, code generation (into many programming languages), and test-case generation.

In Spring Boot, we should add Swagger as a dependency.