*// 1. The original author of the SOLID principles of OOP is: Robert "Uncle Bob" Martin.  
// 2. What principle is the S in SOLID? - Single Responsibility Principle  
// 3. Is the Dependency Injection the same as Dependency Inversion Principle? - No, Dependency Inversion addresses  
// abstractions. While Dependency Injection refers to the injection of Dependencies into a class.  
// 4. What are the three types of Dependency Injection? - By Property, Setter and Constructor.  
// 5. Which type of Dependency Injection should you favor? - Constructor based injection because it requires the  
// dependency to be injected when the class is instantiated.  
// 6. Can you use Dependency Injection against a private property in Spring? - Yes, Spring will support this, and inject the  
// dependency using reflection at runtime. However, this is considered a VERY bad practice..  
// 7. Is it good practice to use concrete classes for dependency injection? - No. You should use interfaces, which  
//will allow the runtime environment to determine the implementation to inject.  
// 8. What is IoC? - Inversion of Control - the runtime environment (or framework) which injects dependencies.  
// 9. What is the ‘L’ in SOLID? - Liskov Substitution Principle, by Barbara Liskov.  
// 10. What is the annotation used in Spring to indicate you want a dependency injected?. - @Autowired.  
// 11. If you have two beans of the same type, how do you specify a preference for one over the other? - The @Primary  
// annotation can be used to designate a primary bean.  
// 12. What are the two callback interfaces you can implement to tap into the bean lifecycle? - InitializingBean and DisposableBean.  
// 13. What two annotations can be used to access the Spring Bean lifecycle? - @PostConstruct and @PreDestroy.  
// 14. How do you specify a bean name you want injected? - Use the @Qualifier annotation.*