Question based on Lec 1 – Peincple and Implementation

Single Answer

1. What is the primary objective of the SOLID principles in software engineering?
   a) To enhance code readability and maintainability
   b) To ensure faster execution of code
   c) To reduce the number of classes in an application
   d) To eliminate the need for abstraction

2. Which principle states that "a class should have one and only one reason to change"?
   a) Open-Closed Principle
   b) Single Responsibility Principle
   c) Dependency Inversion Principle
   d) Interface Segregation Principle

3. What does the Open-Closed Principle (OCP) emphasize?
   a) Classes should be open for modification but closed for extension.
   b) Classes should be open for extension but closed for modification.
   c) Classes should not depend on abstractions.
   d) Classes should implement all methods of their interfaces.

4. Which SOLID principle ensures that "clients should not be forced to implement methods they do not use"?
   a) Single Responsibility Principle
   b) Dependency Inversion Principle
   c) Interface Segregation Principle
   d) Liskov Substitution Principle

5. What is the primary goal of the Dependency Inversion Principle?
   a) To avoid tightly coupling high-level modules with low-level modules
   b) To ensure that all classes have only one responsibility
   c) To allow subclasses to override parent class methods freely
   d) To group similar methods into a single interface

6. Which of the following is NOT a guideline for problem-solving in software engineering?
   a) Divide and conquer
   b) Over-engineer solutions for future needs
   c) Keep it simple and stupid (KISS)
   d) Learn from mistakes

7. What does YAGNI ("You Ain't Gonna Need It") encourage developers to do?
   a) Write code that anticipates future requirements.
   b) Avoid writing unnecessary code that may never be used.
   c) Focus only on debugging existing code.
   d) Avoid using third-party libraries or tools.

8. Which practice emphasizes "fixing not just the bug but also improving the surrounding code"?
   a) Debugging

b) Kaizen
c) Refactoring
d) Unit Testing

9. What is the main purpose of unit testing?
a) To test the entire application as an integrated system
b) To verify individual units of code, such as functions or classes
c) To ensure that user interfaces are responsive and intuitive
d) To analyze code quality using automated tools

10. Which of the following best describes Continuous Integration (CI)?
a) A process where developers manually test their code before committing it to a shared repository
b) A practice where developers frequently integrate their code into a shared repository and verify it through automated builds
c) A method for creating multiple branches in version control systems
d) A testing framework for identifying integration bugs

Multiple Answer

11. Which are benefits of applying SOLID principles in software design? (Select two correct answers.)
a) Improved scalability and maintainability
b) Reduced need for testing frameworks
c) Easier debugging and refactoring
d) Elimination of abstraction layers

12. What are valid guidelines for approaching software solutions? (Select three correct answers.)
a) Divide and conquer complex problems into smaller ones
b) Over-engineer solutions to handle all possible future scenarios
c) Keep solutions simple and easy to understand (KISS principle).
d) Learn from mistakes and anticipate changes

13. Which practices are considered essential for improving code quality? (Select three correct answers.)
a) Code reviews
b) Frequent refactoring
c) Writing compact, unreadable code
d) Unit testing

14. Which principles are part of SOLID design principles? (Select three correct answers.)
a) Single Responsibility Principle (SRP).
b) Open-Closed Principle (OCP).
c) Dependency Injection Pattern (DIP).
d) Interface Segregation Principle (ISP).

15. Which practices are commonly followed in Continuous Integration workflows? (Select two correct answers.)
a) Developers commit code to shared repositories frequently throughout the day
b) Automated builds verify each commit to detect issues early
c) Developers avoid using version control systems during integration
d) Manual testing is preferred over automated testing

Fill in the Blanks

16. The single-responsibility principle states that "a class should have one and only one reason to change."

17. According to the Open-Closed Principle, classes should be open for extension but closed for modification.

18. The linkov substitution principle ensures that "every subclass should be able to substitute its parent class without altering functionality."

19. The KISS principle stands for "keep it simple and stupid," which encourages developers to keep their solutions simple.

20. In Continuous Integration, developers frequently commit their code to a shared repository, which is verified by an automated build process.