40 Lecture - CS504

Important Subjective

1. What is Unit Testing, and why is it important in the software development process?

Answer: Unit Testing is a testing technique where individual units or components of software are tested in isolation. It ensures that each unit functions correctly. Unit Testing is essential as it helps identify defects early in the development process, promotes code reliability, and facilitates easier debugging and maintenance.

2. What are the key benefits of adopting Test-Driven Development (TDD) in Unit Testing?

Answer: Test-Driven Development (TDD) requires writing tests before implementing the code. Its benefits include improved code quality, well-documented code, faster development, better test coverage, and easier refactoring.

3. How do you create test data for Unit Testing when a unit relies on external resources, such as a database?

Answer: For Unit Testing with external dependencies, mock objects or stubs are used to simulate the behavior of external resources. This ensures that the test focuses solely on the unit being tested and is not affected by the availability or state of external resources.

4. Explain the concept of code coverage in Unit Testing and its significance.

Answer: Code coverage measures the extent to which the code is tested during Unit Testing. It helps identify areas of the code that are not exercised by tests, ensuring comprehensive test coverage and reducing the likelihood of undetected defects.

5. How can you handle exceptions and error conditions during Unit Testing?

Answer: Exception handling during Unit Testing involves using assertions to check for expected exceptions or error conditions. Properly handling exceptions helps ensure that the code behaves as intended under various scenarios.

6. What is the role of test fixtures in Unit Testing, and why are they important?

Answer: Test fixtures are the setup and cleanup procedures that prepare the environment for executing Unit Tests. They ensure that each test runs in an isolated and predictable environment, preventing interference between tests and promoting reliable results.

7. How do you decide what to include in a Unit Test and what to exclude?

Answer: Unit Tests should focus on testing individual units in isolation, mocking or stubbing external dependencies. They should cover different scenarios, including boundary cases and error conditions, to ensure comprehensive test coverage.

8. Explain the difference between stubs and mocks in the context of Unit Testing.

Answer: Stubs are fake implementations of dependent components used to simulate their behavior, while mocks are objects used to verify interactions between the unit being tested and its dependencies during the test.

9. How can you ensure that Unit Tests are maintainable and sustainable as the codebase evolves?

Answer: To ensure maintainability, Unit Tests should be well-structured, readable, and properly documented. Regularly updating tests to reflect code changes and refactoring them as needed will help maintain their relevance and effectiveness.

10. What challenges might arise while performing Unit Testing, and how can they be overcome?

Answer: Challenges in Unit Testing may include testing complex code, handling external dependencies, and managing test data. These challenges can be addressed through proper test design, using mocking and stubbing techniques, and creating appropriate test fixtures to isolate the tests.