UNIT V – TEST-DRIVEN DEVELOPMENT

Object-Oriented Testing: Object-Oriented Unit Testing, Object-Oriented Integration Testing, Object-Oriented System Testing - Software Complexity: Unit-Level Complexity, Integration-Level Complexity, System- Level Complexity - Model-Based Testing for Systems of Systems: Characteristics, Sample Systems of Systems Software Engineering for Systems of Systems.

Object-Oriented Testing:

Object-Oriented Testing (OOT) is a software testing method used to check applications developed using object-oriented programming (OOP) principles. It mainly concentrates on testing classes, objects, and the communication between them. It ensures that important OOP features such as encapsulation, inheritance, and polymorphism are implemented and work as expected.

- Checks the functionality and behavior of classes and objects.
- Examines the interaction and communication between different objects and components.
- Verifies the correct use of OOP concepts such as encapsulation, inheritance, and polymorphism.

Dependencies in Object-Oriented Testing

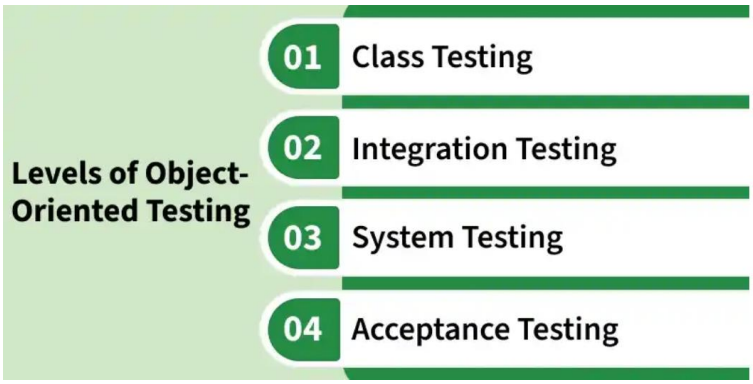
In Object-Oriented Testing, dependencies refer to the connections or relationships between classes, methods, variables, and messages. These relationships need to be properly tested to make sure that objects communicate and work together correctly and that the overall system performs as expected.

Types of Dependencies

- **Class-to-Class Dependency:** One class is related to or depends on another class through **inheritance, association, or aggregation**.
- **Class-to-Method Dependency:** A class depends on its **methods or functions** to perform the required operations.
- **Class-to-Message Dependency:** A class depends on **messages exchanged between objects** for communication and coordination.
- **Class-to-Variable Dependency:** A class depends on its **attributes or data members** to store and handle data.
- **Method-to-Variable Dependency:** A method depends on the **variables used inside the method** while performing its operation.
- **Method-to-Message Dependency:** A method depends on **messages sent to other objects** to complete a particular operation.
- **Method-to-Method Dependency:** One method depends on **another method** to complete or support its functionality.

Levels of Object-Oriented Testing

Object-Oriented Testing is carried out at different levels to check whether classes, objects, and their interactions work properly. Each testing level concentrates on a particular part of the object-oriented system and helps identify errors at different stages of testing.



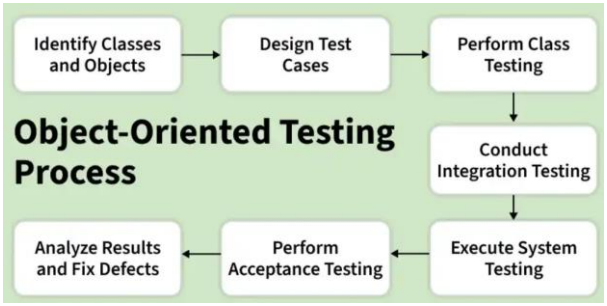
Class Testing: Class testing is the lowest level of object-oriented testing. It verifies the functionality of an individual class by testing its methods, attributes, and behavior independently.

Integration Testing: Integration testing checks the interaction and communication between multiple classes and objects. It ensures that objects work together correctly and exchange data as expected.

System Testing: System testing evaluates the complete integrated application as a whole. It verifies whether the entire object-oriented system meets the specified functional and non-functional requirements.

Acceptance Testing: Acceptance testing is performed by end users or clients to validate that the software satisfies business requirements and is ready for deployment in the real-world environment.

Object-Oriented Testing Process



Identify Classes and Objects: Determine the classes, objects, attributes, and methods that are involved in the application and need to be tested.

Prepare Test Cases: Develop test cases to check the behavior, functions, and interactions of classes and objects under various situations.

Test Individual Classes: Test each class separately to verify that its methods, data members, and operations produce the expected results.

Perform Integration Testing: Test the communication and interaction among different classes and objects to ensure that they exchange data correctly.

Perform System Testing: Test the entire integrated application to check whether all components work together according to the given requirements.

Carry Out Acceptance Testing: End users or customers test the application to verify that it satisfies their needs and is suitable for use.
Review Results and Correct Defects: Examine the test results, find and fix errors, and perform testing again to confirm that the software works correctly and reliably.

Testing Techniques in Object-Oriented Testing (OOT)

Object-Oriented Testing uses different methods to check the behavior of classes, objects, and their communication. These techniques help to make sure that the software works properly in various situations.

- **State-Based Testing:**
  State-Based Testing checks how an object behaves when it changes from one state to another during its lifecycle. Test cases are prepared to verify whether the object moves to the correct state when a particular event or input occurs.
- **Class-Based Testing:**
  Class-Based Testing checks a single class by testing its methods, attributes, constructors, and operations. It verifies whether the class performs its required functions correctly on its own.
- **Scenario-Based Testing:**
  Scenario-Based Testing checks the software using practical, real-life situations. It represents how users normally interact with different objects and components. This technique helps to find problems that may occur during real-world usage.
- **Use Case Testing:**
  Use Case Testing is performed based on use cases and user requirements. Test cases are created for different user actions and workflows to check whether the system performs the required functions and produces the expected results.