Black Box Testing

Black-box testing is a type of software testing in which the tester is not concerned with the internal knowledge or implementation details of the software but rather focuses on validating the functionality based on the provided specifications or requirements.

Black box testing can be done in the following ways:

1. **Syntax-Driven Testing** – This type of testing is applied to systems that can be syntactically represented by some language. For example, language can be represented by context-free grammar. In this, the test cases are generated so that each grammar rule is used at least once.

2. **Equivalence partitioning** – It is often seen that many types of inputs work similarly so instead of giving all of them separately we can group them and test only one input of each group. The idea is to partition the input domain of the system into several equivalence classes such that each member of the class works similarly, i.e., if a test case in one class results in some error, other members of the class would also result in the same error.

3. **Boundary value analysis** – Boundaries are very good places for errors to occur. Hence, if test cases are designed for boundary values of the input domain then the efficiency of testing improves and the probability of finding errors also increases. For example – If the valid range is 10 to 100 then test for 10,100 also apart from valid and invalid inputs.

4. **Cause effect graphing** – This technique establishes a relationship between logical input called causes with corresponding actions called the effect. The causes and effects are represented using Boolean graphs. The following steps are followed:

- Identify inputs (causes) and outputs (effects).
- Develop a cause-effect graph.
- Transform the graph into a decision table.
- Convert decision table rules to test cases.

5. **Requirement-based testing** – It includes validating the requirements given in the SRS of a software system.

6. **Compatibility testing** – The test case results not only depend on the product but also on the infrastructure for delivering functionality. When the infrastructure parameters are changed it is still expected to work properly. Some parameters that generally affect the compatibility of software are:

- Processor (Pentium 3, Pentium 4) and several processors.
- Architecture and characteristics of a machine (32-bit or 64-bit).
- Back-end components such as database servers.
- Operating System (Windows, Linux, etc).

Black Box Testing Type

The following are the several categories of black box testing:

1. Functional Testing
2. Regression Testing
3. Nonfunctional Testing (NFT)

Functional Testing: It determines the system's software functional requirements.
Regression Testing: It ensures that the newly added code is compatible with the existing code. In other words, a new software update has no impact on the functionality of the software. This is carried out after a system maintenance operation and upgrades.
Nonfunctional Testing: Nonfunctional testing is also known as NFT. This testing is not functional testing of software. It focuses on the software's performance, usability, and scalability.

What can be identified by Black Box Testing:

- Discovers missing functions, incorrect function & interface errors
- Discover the errors faced in accessing the database
- Discovers the errors that occur while initiating & terminating any functions.
- Discovers the errors in performance or behaviour of software.

Features of black box testing:

- **Independent testing:** Black box testing is performed by testers who are not involved in the development of the application, which helps to ensure that testing is unbiased and impartial.
- **Testing from a user's perspective:** Black box testing is conducted from the perspective of an end user, which helps to ensure that the application meets user requirements and is easy to use.
- **No knowledge of internal code:** Testers performing black box testing do not have access to the application's internal code, which allows them to focus on testing the application's external behaviour and functionality.
- **Requirements-based testing:** Black box testing is typically based on the application's requirements, which helps to ensure that the application meets the required specifications.
- **Different testing techniques:** Black box testing can be performed using various testing techniques, such as functional testing, usability testing, acceptance testing, and regression testing.
- **Easy to automate:** Black box testing is easy to automate using various automation tools, which helps to reduce the overall testing time and effort.
- **Scalability:** Black box testing can be scaled up or down depending on the size and complexity of the application being tested.
- **Limited knowledge of application:** Testers performing black box testing have limited knowledge of the application being tested, which helps to ensure that testing is more representative of how the end users will interact with the application.

Advantages of Black Box Testing:

- The tester does not need to have more functional knowledge or programming skills to implement the Black Box Testing.
- It is efficient for implementing the tests in the larger system.
- Tests are executed from the user's or client's point of view.
- Test cases are easily reproducible.
- It is used in finding the ambiguity and contradictions in the functional specifications.

Disadvantages of Black Box Testing:

- There is a possibility of repeating the same tests while implementing the testing process.
- Without clear functional specifications, test cases are difficult to implement.
- It is difficult to execute the test cases because of complex inputs at different stages of testing.
- Sometimes, the reason for the test failure cannot be detected.
- Some programs in the application are not tested.
- It does not reveal the errors in the control structure.
- Working with a large sample space of inputs can be exhaustive and consumes a lot of time.