Unit: IV
Testing
Lecture 1

Testing is the process of evaluating a system or its component(s) with the intent to find that whether it satisfies the specified requirements or not. This activity results in the actual, expected and difference between their results.

➔ testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements.

According to ANSI/IEEE 1059 standard, Testing can be defined as:

"A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item."

Who does testing?

It depends on the process and the associated stakeholders of the project(s). In the IT industry, large companies have a team with responsibilities to evaluate the developed software in the context of the given requirements. Moreover, developers also conduct testing which is called Unit Testing.

In most cases, following professionals are involved in testing of a system within their respective capacities:

- Software Tester
- Software Developer
- Project Lead/ Manager
- End User

Different companies have different designations for people who test the software on the basis of their experience and knowledge such as Software Tester, Software Quality Assurance Engineer and QA Analyst etc.

Testing Objectives:

- Testing is a process of executing a program with the intent of finding an error.
- A good test case is one that has a high probability of finding an as-yet discovered error.
- A successful test is one that uncovers an as-yet-undiscovered error.

Testing Principles:

- **All tests should be traceable to customer requirements:** As most severe defects (from the customer's point of view) are those that cause the program to fail to meet its requirements.
- **Tests should be planned long before testing begins:** Test planning can begin as soon as the requirements model is complete.
- **The Pareto principle applies to software testing:** Pareto principle implies that 80% of all errors uncovered during testing will likely be traceable to 20% of all program components. The problem, of course, is to isolate these suspect components and to thoroughly test them.
- **Testing should begin "in the small" and progress toward testing "in the large":** The first tests planned and executed generally focus on individual components. As testing progresses, focus shifts in an attempt to find errors in integrated clusters of components and ultimately in the entire system.
- **Exhaustive testing is not possible:** It is impossible to execute every combination of paths during testing.
- **To be most effective, testing should be conducted by an independent third party.**

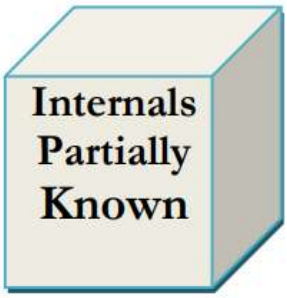
Testing Methods:

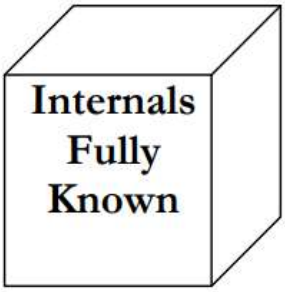
- **Black Box Testing:** The technique of testing without having any knowledge of the interior workings of the application is black box testing.
    - The tester is oblivious to the system architecture and does not have access to the source code.
    - Typically, when performing a black box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.
    - **Advantages:**
        - Well suited and efficient for large code segments.
        - Code access not required.
        - Large number of moderately skilled testers can test the application with no knowledge of implementation, programming language or operating systems.

- **Disadvantages:**
  - Inefficient testing, due to the fact that the tester only has limited knowledge about an application.
  - Blind coverage, since the tester cannot target specific code segments or error prone areas.
  - Limited coverage since only a selected number of test scenarios are actually performed.

- **White Box Testing:** White box testing is the detailed investigation of internal logic and structure of the code.
  - White box testing is also called glass testing or open box testing.
  - In order to perform white box testing on an application, the tester needs to possess knowledge of the internal working of the code. The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.
  - **Advantages:**
    - As the tester has knowledge of the source code, it becomes very easy to find out which type of data can help in testing the application effectively.
    - It helps in optimizing the code.
    - Extra lines of code can be removed which can bring in hidden defects.
    - Due to the tester's knowledge about the code, maximum coverage is attained during test scenario writing.
  - **Disadvantages:**
    - Due to the fact that a skilled tester is needed to perform white box testing, the costs are increased.
    - Sometimes it is impossible to look into every nook and corner to find out hidden errors that may create problems as many paths will go untested.
    - It is difficult to maintain white box testing as the use of specialized tools like code analyzers and debugging tools are required.

- **Grey Box Testing:** Grey box testing is a technique to test the application with limited knowledge of the internal workings of an application.
    - Unlike black box testing, where the tester only tests the application's user interface, in grey box testing, the tester has access to design documents and the database. Having this knowledge, the tester is able to better prepare test data and test scenarios when making the test plan.
    - **Advantages:**
        - Often combined benefits of black box and white box testing wherever possible.
        - Grey box testers don't rely on the source code; instead they rely on interface definition and functional specifications.
        - Based on the limited information available, a grey box tester can design excellent test scenarios especially around communication protocols and data type handling.
        - The test is done from the point of view of the user and not the designer.
    - **Disadvantages:**
        - Since the access to source code is not available, the ability to go over the code and test coverage is limited.
        - The tests can be redundant if the software designer has already run a test case.
        - Testing every possible input stream is unrealistic because it would take an unreasonable amount of time, therefore, many program paths will go untested.





Comparison between the Three Testing Types

	Black Box Testing	Grey Box Testing	White Box Testing
1.	The Internal Workings of an application are not required to be known	Somewhat knowledge of the internal workings are known	Tester has full knowledge of the Internal workings of the application
2.	Also known as closed box testing, data driven testing and functional testing	Another term for grey box testing is translucent testing as the tester has limited knowledge of the insides of the application	Also known as clear box testing, structural testing or code based testing
3.	Performed by end users and also by testers and developers	Performed by end users and also by testers and developers	Normally done by testers and developers
4.	-Testing is based on external expectations -Internal behavior of the application is unknown	Testing is done on the basis of high level database diagrams and data flow diagrams	Internal workings are fully known and the tester can design test data accordingly
5.	This is the least time consuming and exhaustive	Partly time consuming and exhaustive	The most exhaustive and time consuming type of testing
6.	Not suited to algorithm testing	Not suited to algorithm testing	Suited for algorithm testing
7.	This can only be done by trial and error method	Data domains and Internal boundaries can be tested, if known	Data domains and Internal boundaries can be better tested