Manual Testing

- A type of software testing in which test cases are executed manually by a tester without using any automated tools.
- The most primitive technique of all testing types and it helps to find critical bugs in the software application.

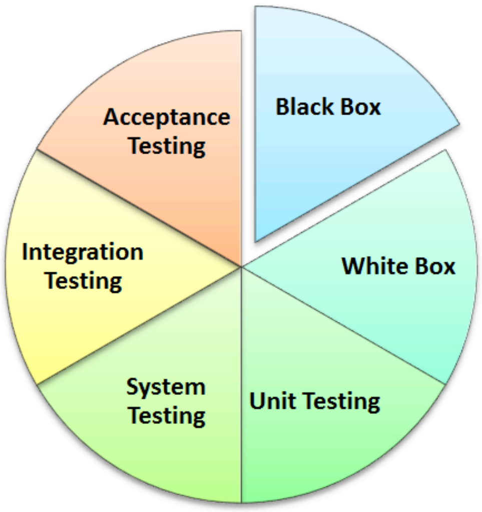
Purpose of Manual Testing

- Identify the bugs, issues, and in the software application.
- Any new application must be manually tested before its testing can be automated. Manual Software Testing requires more effort but is necessary to check automation feasibility.
- Manual Testing concepts does not require knowledge of any testing tool. One of the Software Testing Fundamental is “100% Automation is not possible“. This makes Manual Testing imperative.

Goals of Manual Testing?

- The key concept of manual testing is to ensure that the application is error free and it is working in conformance to the specified functional requirements. Test Suites or cases are designed during the testing phase and should have 100% test coverage.
- It also makes sure that reported defects are fixed by developers and re-testing has been performed by testers on the fixed defects. Basically, this testing checks the quality of the system and delivers bug-free products to the customer.

Types of Manual Testing



Manual Testing Vocabularies

- **Audit:** This is an inspection/assessment activity that verifies compliance with plans, policies and procedures and ensures that resources are conserved.
- **Baseline:** A quantitative measure of the current level of performance.
- **Benchmarking:** Comparing your company’s products, services or processes against best practices or competitive practices, to help define superior performance of a product, service or support processes.
- **Black-box Testing:** A test technique that focuses on testing the functionality of the program component or application against its specifications without

knowledge of how the system is constructed.

- **Boundary value analysis:** A data selection technique in which test data is chosen from the "boundaries" of the input or output domain classes, data structures and procedure parameters. Choices often include the actual minimum and maximum boundary values, the maximum value plus or minus one and the minimum value plus or minus one.
- **Branch Testing:** A test method that requires that each possible branch on each decision be executed at least once.
- **Brainstorming:** A group process for generating creative and diverse ideas.
- **Bug:** A catchall term for all software defects or errors.
- **Debugging:** The process of analyzing and correcting syntactic, logic and other errors identified during testing.
- **Decision Coverage:** A white-box testing technique that measures the number of – or percentage – of decision directions executed by the test case designed. 100% Decision coverage would indicate that all decision directions had been executed at least once during testing.
- **Decision Table:** A tool for documenting the unique combinations of conditions and associated results in order to derive unique test cases for validation testing.
- **Defect Tracking Tools:** Tools for documenting defects as they are found during testing and for tracking their status through to resolution.
- **Desk Check:** A verification technique conducted by the author of the artifact to verify the completeness of their own work
- **Dynamic Analysis:** Analysis performed by executing the program code. Dynamic analysis executes or simulates a development phase product and it detects errors by analyzing the response of the product to sets of input data.
- **Entrance Criteria:** Required conditions and standards for work product quality that must be present or met for entry into the next stage of the software development process.
- **Equivalence Partitioning:** A test technique that utilizes a subset of data that is representative of a larger class. This is done in place of undertaking exhaustive testing of each value of the larger class of data
- **Error or defect:** (1) A discrepancy between a computed, observed or measured value or condition and the true, specified or theoretically correct value or condition (2) Human action that results in software containing a

fault (e.g., omission or misinterpretation of user requirements in a software specification, incorrect translation or omission of a requirement in the design specification)

- **Error Guessing:** Test data selection techniques for picking values that seem likely to cause defects. This technique is based upon the theory that test cases and test data can be developed based on intuition and experience of the tester.
- **Exhaustive Testing:** Executing the program through all possible combinations of values for program variables.
- **Exit criteria:** Standards for work product quality which block the promotion of incomplete or defective work products to subsequent stages of the software development process.
- **Flowchart:** Pictorial representations of data flow and computer logic. It is frequently easier to understand and assess the structure and logic of an application system by developing a flow chart than to attempt to understand narrative descriptions or verbal explanations. The flowcharts for systems are normally developed manually, while flowcharts of programs can be produced.
- **Force Field Analysis:** A group technique used to identify both driving and restraining forces that influence a current situation.
- **Formal Analysis:** Technique that uses rigorous mathematical techniques to analyze the algorithms of a solution for numerical properties, efficiency, and correctness.
- **Functional Testing:** Testing that ensures all functional requirements are met without regard to the final program structure.
- **Histogram:** A graphical description of individually measured values in a data set that is organized according to the frequency or relative frequency of occurrence. A histogram illustrates the shape of the distribution of individual values in a data set along with information regarding the average and variation.
- **Inspection:** A formal assessment of a work product conducted by one or more qualified independent reviewers to detect defects, violations of development standards, and other problems. Inspections involve authors only when specific questions concerning deliverables exist. Inspection: A formal assessment of a work product conducted by one or more qualified independent reviewers to detect defects, violations of development standards, and other problems. Inspections involve authors only

when specific questions concerning deliverables exist.

- **Integration Testing:** This test begins after two or more programs or application components have been successfully unit tested. It is conducted by the development team to validate the interaction or communication/flow of information between the individual components which will be integrated.
- **Life Cycle Testing:** The process of verifying the consistency, completeness, and correctness of software at each stage of the development life cycle
- **Pass/Fail Criteria:** Decision rules used to determine whether a software item or feature passes or fails a test.
- **Path Testing:** A test method satisfying the coverage criteria that each logical path through the program be tested. Often, paths through the program are grouped into a finite set of classes and one path from each class is tested.
- **Performance Test:** Validates that both the online response time and batch run times meet the defined performance requirements.
- **Policy:** Managerial desires and intents concerning either process (intended objectives) or products (desired attributes).
- **Population Analysis:** Analyzes production data to identify, independent from the specifications, the types and frequency of data that the system will have to process/produce. This verifies that the specs can handle types and frequency of actual data and can be used to create tests.
- **Procedure:** The step-by-step method followed to ensure that standards are met.
- **Process:** (1) The work effort that produces a product. This includes efforts of people and equipment guided by policies, standards, and procedures. (2) A statement of purpose and an essential set of practices (activities) that address that purpose
- **Proof of Correctness:** The use of mathematical logic techniques to show that a relationship between program variables assumed true at program entry implies that another relationship between program variables holds at program exit.
- **Quality:** A product is a quality product if it is defect free. To the producer, a product is a quality product if it meets or conforms to the statement of requirements that defines the product. This statement is usually shortened to: quality means meets requirements. From a customer's perspective, quality means "fit for use."
- **Quality Assurance (QA):** Deals with 'prevention' of defects in the product being developed. It is

associated with a process. The set of support activities (including facilitation, training, measurement, and analysis) needed to provide adequate confidence that processes are established and continuously improved to produce products that meet specifications and are fit for use.

- **Quality Control (QC):** Its focus is defect detection and removal. Testing is a quality control activity.
- **Quality Improvement:** To change a production process so that the rate at which defective products (defects) are produced is reduced. Some process changes may require the product to be changed

LESSON 2

- **Software:** a collection of computer programs that helps us to perform tasks
- **TYPES OF SOFTWARE**
    - 1. System Software
    - 2. Programming Software
    - 3. Application Software
    - Ex. Desktop apps, Web applications, mobile apps
- **Software Testing**: A part of the software development process. An activity to detect and identify the defects in the software Therefore, the **objective/reason** of testing is to release **quality** products to the client.
- **Software Quality**
    - Bug-free
    - Delivered on time
    - Within budget
    - Meets requirements and/or expectations
    - Maintainable

Project VS. Product

- There is software application that is developed for <u>specific customer</u> based on the requirements
- There is a software application that is developed for <u>multiple customers</u> based on market requirements.
- **Error:** human mistakes
- **Bugs/Defects**: related to application which shows deviation from expected actual behavior

Error or Defect

- A discrepancy between a computed, observed or measured value or condition and the true, specified or theoretically correct value or condition
- Human action that results in software containing a fault
- **Defect Density:** The degree of compactness of a substance
- **Defect Tracking Tools:** Tools for documenting defects as they are found during testing and for tracking their status through to resolution.

BEWARE: *The higher bug density, the poorer the quality*

- **Failure**: Deviation that was identified by the end-user after sometime using the app.

5 Whys Software has Bugs

- Miscommunication or No communication
- Software Complexity
- Programming Errors
- Changing requirements
- Lack of skilled Testers
- **Brainstorming:** An individual but usually group process for generating creative and diverse ideas.

Common Functional Testing Techniques

- Smoke testing
- Sanity testing
- Black box testing
- Integration testing
- **Black box testing:** A test technique that focuses on testing the functionality of the program component or application against its specifications without knowledge of how the system is constructed.

Types of Black-box Testing

- Functional Testing
- Non-functional testing
- Regression testing

Black-box Testing Techniques

- Equivalence Partitioning
- Boundary Value Analysis
- Decision Table Testing
- State Transition Testing
- Use Case Testing
- Error Guessing

- **Equivalence Partitioning:** A test technique that utilizes a subset of data that is representative of a larger class.
- **Boundary value analysis:** A data selection technique in which test data is chosen from the "boundaries" of the input or output domain classes, data structures and procedure parameters. Choices often include the actual minimum and maximum boundary values, the maximum value plus or minus one and the minimum value plus or minus one.
- **Decision Table:** A tool for documenting the unique combinations of conditions and associated results in order to derive unique test cases for validation testing.

Types of Decision Table

- 1. Limited Entry
- 2. Extended Entry

4 Parts of Decision Tables

- 1. Condition Stubs
- 2. Action Stubs
- 3. Condition Entries
- 4. Action Entries
- **State Transition Testing:** State Transition Testing exhibits the various states of scenario/system and possible transition between them.
- **Use Case Testing:** A black box testing technique in which changes made in input conditions cause state changes or output changes in the Application Under Test(AUT).

Rules of Case Testing

- The name of an actor or a use case must be meaningful and relevant to the system.
- Interaction of an actor with the use case must be defined clearly and in an understandable way

- **Non-Functional Testing**: Non functional testing is a type of software testing that verifies non functional aspects of the product, such as performance, stability, and usability.
- **White Box Testing:** White-box testing is effectively the opposite of black-box testing. It focuses on the internal structure of a test object, rather than just analyzing the inputs and outputs.

Techniques of White Box Testing

- Statement based
    - Static
    - Dynamic
- Decision-based
    - Static
    - Dynamic
- **Statement Coverage:** The statement based technique focuses on the individual lines of code to be delivered. The idea is to analyze the actual lines of code in the software program and have tests that verify these lines.
- **Static vs Dynamic**: Whether you use statement or decision-based techniques there is another choice to make.
- **Static Analysis:** Static is a manual approach. This relies on the developer going through the code and eyeballing the code to see obvious errors. This can be done by the creator of the code

What are the scopes of testing?

- **Software Testing:** a way to assess the quality of the software and to reduce the risks of software failure in operation.

Software Testing Inclusions

- Test planning
- Analyzing
- Designing
- Implementing tests
- Reporting test progress and results
- Evaluating the quality of a test object
- **Static testing:** does not involve the execution of the component Ex.: Code reviews, Requirements review, design review
- **Dynamic Testing:** the execution of the component or system being tested Ex. Accepting invalid values, Run time errors

Testing Objectives

- 1. To prevent the defects
- 2. To verify all the specified requirements have been fulfilled
- 3. To check whether the test object is complete
- 4. To build confidence
- 5. To provide sufficient information
- 6. To comply with contractual, legal, or regulatory requirements
- **Component Testing:** To find as many failures as possible so that the underlying defects are identified and fixed early.

- **Acceptance Testing:** To confirm that the system works as expected and satisfies requirements.
- **Testing Scope:** a list of product features, product parts, or product-related integrations that must be tested in order to build a reliable assessment of a product's quality.