1. **Acceptance Testing**

Acceptance testing, a testing technique performed to determine whether or not the software system has met the requirement specifications.

There are various forms of acceptance testing:

* User acceptance Testing
* Business acceptance Testing
* Alpha Testing
* Beta Testing

## Accessibility Testing

Accessibility testing is a subset of usability testing where in the users under consideration are people with all abilities and disabilities. The significance of this testing is to verify both usability and accessibility.

* Visual Impairments
* Physical Impairment
* Hearing Impairment
* Cognitive Impairment
* Learning Impairment

1. **Active testing**

Active testing, a testing technique, where the user introduces test data and analyses the result.

During Active testing, a tester builds a mental model of the software under test which continues to grow and refine as your interaction with the software continues

1. **AdHoc Testing**

When a software testing performed without proper planning and documentation, it is said to be Adhoc Testing. Such kind of tests are executed only once unless we uncover the defects.

## Forms of Adhoc Testing :

* **Buddy Testing:**Two buddies, one from development team and one from test team mutually work on identifying defects in the same module. Buddy testing helps the testers develop better test cases while development team can also make design changes early. This kind of testing happens usually after completing the unit testing.
* **Pair Testing:**Two testers are assigned the same modules and they share ideas and work on the same systems to find defects. One tester executes the tests while another tester records the notes on their findings.
* **Monkey Testing:**Testing is performed randomly without any test cases in order to break the system.

1. **Age testing**

It is a testing technique that evaluates a system's ability to perform in the future and usually carried out by test teams. As the system gets older, how significantly the performance might drop is what is being measured in Age Testing.

1. **Agile testing**

A software testing practice that follows the principles of agile software development is called Agile Testing. Agile is an iterative development methodology, where requirements evolve through collaboration between the customer and self-organizing teams and agile aligns development with customer needs

1. **All pair testing**

All-pairs also known as pairwise testing is a testing approach taken for testing the software using combinatorial method. It's a method to test all the possible discrete combinations of the parameters involved.

Assume we have a piece of software to be tested which has got 10 input fields and 10 possible settings for each input field. Then, there are 10^10 possible inputs to be tested. In this case, exhaustive testing is impossible even if we wish to test all combinations.

1. **Alpha testing**

Alpha testing takes place at the developer's site by the internal teams, before release to external customers. This testing is performed without the involvement of the development teams.

1. **API testing**

API stands for **A**pplication **P**rogramming **I**nterface, which specifies how one component should interact with the other. It consists of a set of routines, protocols and tools for building the software applications.

1. **Arc testing**

Arc Testing is nothing but branch testing. A Branch is the outcome of a decision. So branch coverage is a measure of outcomes of a branch. Determining the number of branches in a method is easy as a the output of a branch has usually two outcomes (True of False).

1. **Anomaly**

In Software testing, Anomaly refers to a result that is different from the expected one. This behaviour can result from a document or also from a testers notion and experiences.

An Anomaly can also refer to a usability problem as the testware may behave as per the specification, but it can still improve on usability. Sometimes, the anomaly can also referred as a defect / Bug.

1. **Assertion based Testing**

An assertion is a boolean expression at a specific point in a program which will be true unless there is a bug in the program. A test assertion is defined as an expression, which encapsulates some testable logic specified about a target under test.

1. **Audit**

Audit means an independent examination of a software product or processes to assess compliance with specifications, standards, contractual agreements, or other criteria.

1. **Automated Software Testing**

Software Test automation makes use of specialized tools to control the execution of tests and compares the actual results against the expected result. Usually regression tests, which are repetitive actions, are automated.

Testing Tools not only help us to perform regression tests but also helps us to automate data set up generation, product installation, GUI interaction, defect logging, etc.

1. **Backward Compatibility Testing**

An Application/Product developed using one version of a platform should still work in a newer version of a platform. The Testing that ensures new version of the product to continue to work with the older product is known as Backward compatibility testing.

1. **Baseline artefacts**

The process of managing the changes in hardware, software, firmware or documentation is known as Configuration management. Baselining is the identification of significant states within the revision history of a configuration item.

1. **Basis path testing**

Basis path testing, a structured testing or white box testing technique used for designing test cases intended to examine all possible paths of execution at least once. Creating and executing tests for all possible paths results in 100% statement coverage and 100% branch coverage.

1. **Bebugging**

Bebugging is the process of adding known defects to the application intentionally for the purpose of monitoring the rate of detection and removal. This process is also known as defect seeding or Fault injection or defect feeding.

1. **Behavioural Testing**

is a testing of the external behaviour of the program, also known as black box testing. It is usually a functional testing.

1. **Benchmark testing**

is a part of the software development life cycle that involves both developers and database administrators (DBAs) to determine current performance and make changes to improve the performances of the same.

The coding should be done very efficiently along with fine tuning the databases so that user can experience the performance improvements.

1. **Beta Testing**

Beta testing also known as user testing takes place at the end users site by the end users to validate the usability, functionality, compatibility, and reliability testing.

Beta testing adds value to the software development life cycle as it allows the "real" customer an opportunity to provide inputs into the design, functionality, and usability of a product. These inputs are not only critical to the success of the product but also an investment into future products when the gathered data is managed effectively.

1. **Big-Bang testing**

Big Bang Integration Testing is an integration testing strategy wherein all units are linked at once, resulting in a complete system. When this type of testing strategy is adopted, it is difficult to isolate any errors found, because attention is not paid to verifying the interfaces across individual units.

1. **Binary portability Testing**

Binary Portability is Testing an executable for portability across platforms and environments, usually for the conformation to an Application Binary Interface(ABI) specification.

The Application Binary Interface (or ABI) defines a system interface for compiled application programs and also different for different types of hardware architecture. Since the binary specification includes information specific to the computer processor architecture for which it is intended, it is not possible to specify for a single document for all possible System. Hence, ABI is a family of specifications, rather than a single one.

1. **Black box Testing**

Black-box testing is a method of software testing that examines the functionality of an application based on the specifications. It is also known as Specifications based testing. Independent Testing Team usually performs this type of testing during the software testing life cycle.

1. **Bottom up Testing**

Each component at lower hierarchy is tested individually and then the components that rely upon these components are tested.

1. **Boundary testing**

Boundary value analysis is a type of black box or specification based testing technique in which tests are performed using the boundary values.

1. **Branch Testing**

Branch coverage is a testing method, which aims to ensure that each one of the possible branch from each decision point is executed at least once and thereby ensuring that all reachable code is executed.

1. **Breadth testing**

Breadth Testing is a test suite that validates the full functionality of a product but does not test the product features in detail.

Top-down integration testing can either take depth-first or a breadth-first approach. In breadth-first testing, all the modules are refined at the same level of control. In reality, a combination of the both breadth-first testing and depth-first testing approach is used.

1. **Bug**

In Software testing, when the expected and actual behavior is not matching, an incident needs to be raised. An incident may be a Bug. It is a programmer's fault where a programmer intended to implement a certain behavior, but the code fails to correctly conform to this behavior because of incorrect implementation in coding. It is also known as Defect.

1. **Business Process**

A business process is a collection of activities or tasks that produce a specific service or product for end users. It is usually represented as a flowchart as a sequence of activities that points to a Process Matrix.

1. **Business Requirement**

Business requirements is a phase in Software development life cycle which felicitates the requirements of the end users as the very first task in order to guide the design of the future system. Business requirements are usually captured by business analysts or product owners who analyze business activities who in turn act as subject matter expertise (SME's).

1. **Capability muturity model**

The Software Engineering Institute (SEI) Capability Maturity Model (CMM) specifies an increasing series of levels of a software development organization. The higher the level, the better the software development process, hence reaching each level is an expensive and time-consuming process.

1. **Capture/Replay Tool**

The Software Engineering Institute (SEI) Capability Maturity Model (CMM) specifies an increasing series of levels of a software development organization. The higher the level, the better the software development process, hence reaching each level is an expensive and time-consuming process.

1. **Cause Effect path**

Cause Effect Graph is a black box testing technique that graphically illustrates the relationship between a given outcome and all the factors that influence the outcome.

1. **Code Coverage**

Code Coverage testing is determining how much code is being tested. It can be calculated using the formula:

Code Coverage = (Number of lines of code exercised)/(Total Number of lines of code) \* 100%

1. **Code Freeze**

Code Freeze means the code is frozen and there will not be any further modifications from the developers. After the code freeze, the code should not be changed by the developers. Only in case of critical defects, developers change the code after the approval of the change control board and make the changes required to fix that critical defect.

1. **Code Inspection**

Code Inspection is the most formal type of review, which is a kind of static testing to avoid the defect multiplication at a later stage.

1. **Code Review**

Code Review is a systematic examination, which can find and remove the vulnerabilities in the code such as memory leaks and buffer overflows.

1. **Code Walkthrough**

Code Walkthrough is a form of peer review in which a programmer leads the review process and the other team members ask questions and spot possible errors against development standards and other issues.

1. **Code Base Testing**

Code-based testing corresponds to the testing that is carried out on code development, code inspection, unit testing in software development process.

1. **Code Driven Testing**

The Code driven testing also known as test driven development's first step is to add a test, which is enough for code to fail. Next, we execute out tests usually a complete test suite or a subset of tests to accelerate testing to ensure that the new test fails. Then, the code is updated to make it pass the new tests. Then, the fourth step is to execute the tests again. If they fail, we need to update the code and retest. Once the test passes, the next step is to go through the same process again for the next development item.

1. **Code Free Testing**

Code free testing, a testing technique performed without the execution of code. The static testing techniques such as review and static analysis are part of code free testing.

1. **Compression Testing**

Comparison testing comprises of comparing the contents of files, databases, against actual results. They are capable of highlighting the differences between expected and actual results.

1. **Compatibility Testing**

Compatibility testing is a non-functional testing conducted on the application to evaluate the application's compatibility within different environments. It can be of two types - forward compatibility testing and backward compatibility testing.

1. **Compliance Testing**

Compliance Testing is performed to maintain and validate the compliant state for the life of the software. Every industry has a regulatory and compliance board that protects the end users.

1. **Concurrency testing**

is also known as multi-user testing, performed to identify the defects in an application when multiple users login to the application.

1. **Condition Coverage Testing**

Condition coverage is also known as Predicate Coverage in which each one of the Boolean expression have been evaluated to both TRUE and FALSE.

1. **Configuration Testing**

Configuration testing is the process of testing the system with each one of the supported software and hardware configurations. The Execution area supports configuration testing by allowing reuse of the created tests.

1. **Context Driven Testing**

The context-driven software testing advocates testing based on the context of the project as opposed to go by books methodology testing or some fixed notion of best practices.

1. **Control Flow Path**

A control flow path is a graphical representation of all paths that might be traversed through a program during its execution. Most representations are of two types of blocks. Viz - An entry block through which control enters into the flow graph and the exit block through which all control flow leaves.

1. **Correctness**

Correctness from software engineering perspective can be defined as the adherence to the specifications that determine how users can interact with the software and how the software should behave when it is used correctly.

1. **Data Integrity Testing**

Data integrity corresponds to the quality of data in the databases and to the level by which users examine data quality, integrity and reliability. Data integrity testing verifies that the data in the database is accurate and functions as expected within a given application.

1. **Data Driven Testing**

Data-driven testing is creation of test scripts where test data and/or output values are read from data files instead of using the same hard-coded values each time the test runs. This way, testers can test how the application handles various inputs effectively. It can be any of the below data files.

1. **Data Flow Testing**

Data flow testing is a family of test strategies based on selecting paths through the program's control flow in order to explore sequences of events related to the status of variables or data objects. Dataflow Testing focuses on the points at which variables receive values and the points at which these values are used

1. **Debugging**

It is a systematic process of spotting and fixing the number of bugs, or defects, in a piece of software so that the software is behaving as expected. Debugging is harder for complex systems in particular when various subsystems are tightly coupled as changes in one system or interface may cause bugs to emerge in another.

1. **Decision Coverage Testing**

Decision coverage or Branch coverage is a testing method, which aims to ensure that each one of the possible branch from each decision point is executed at least once and thereby ensuring that all reachable code is executed.

1. **Defect**

A software bug arises when the expected result don't match with the actual results. It can also be error, flaw, failure, or fault in a computer program. Most bugs arise from mistakes and errors made by developers, architects.

1. **Defect Logging and Tracking**

Defect logging, a process of finding defects in the application under test or product by testing or recording feedback from customers and making new versions of the product that fix the defects or the clients feedback.

1. **Defect Life cycle**

Defect life cycle, also known as Bug Life cycle is the journey of a defect cycle, which a defect goes through during its lifetime. It varies from organization to organization and also from project to project as it is governed by the software testing process and also depends upon the tools used.

1. **Delta Release**

A delta release also known as partial release is one that includes only those areas within the release unit that have changed or are new since the last full release or delta release.

1. **Depth Testing**

Depth Testing is a testing technique in which feature of a product is tested in full detail. Each of the feature is tested exhaustively during the integration phase and the defects are logged, are captured across all parameters, functional and non functional.

1. **Destructive Testing**

Destructive testing is a testing technique in which the application is made to fail in an uncontrolled manner to test the robustness of the application and also to find the point of failure.

1. **Development Environment**

The development environment helps the developers to develop the application or product using a set of processes and programming tools.

1. **Domain Testing**

Domain testing is a software testing technique in which selecting a small number of test cases from a nearly infinite group of test cases. For testing few applications, Domain specific knowledge plays a very crucial role.

1. **Durability Testing**

Durability Testing is a Performance testing technique used to determine the characteristics of a system under various load conditions over time. This testing helps us to identify the stability of transaction response times over the duration of the test.

1. **Dynamic Testing**

Dynamic Testing is a kind of software testing technique using which the dynamic behaviour of the code is analysed.

1. **Emulator**

An emulator is a hardware or software that emulates the functions of one computer system in another computer system.

1. **End to end Testing**

End-to-end testing is a technique used to test whether the flow of an application right from start to finish is behaving as expected. The purpose of performing end-to-end testing is to identify system dependencies and to ensure that the data integrity is maintained between various system components and systems.

1. **Endurance Testing**

Endurance Testing also known as Soak Testing is performed to determine if the application under test can sustain the continuous loads.

1. **Entry Criteria**

Entry criterion is used to determine when a given test activity should start. It also includes the beginning of a level of testing, when test design or when test execution is ready to start.

1. **Equivalence Partitioning Testing**

Equivalence Partitioning also called as equivalence class partitioning. It is abbreviated as ECP. It is a software testing technique that divides the input test data of the application under test into each partition at least once of equivalent data from which test cases can be derived.

1. **Error**

When the system produces an outcome, which is not the expected one or a consequence of a particular action, operation, or course, is known as error.

1. **Error Guessing**

Error guessing is a testing technique that makes use of a tester's skill, intuition and experience in testing similar applications to identify defects that may not be easy to capture by the more formal techniques. It is usually done after more formal techniques are completed.

1. **Error Seeding**

It is a process of consciously adding of errors to the source code, which can be used to evaluate the amount of residual errors after the system software test phase. After adding the errors to the source code, one can try to estimate the number of "real" errors in the code based on the number of seeded errors found.

1. **Exhaustive testing**

is a test approach in which all possible data combinations are used for testing. Exploratory testing includes implicit data combinations present in the state of the software/data at the start of testing

1. **Exit Criteria**

Exit criterion is used to determine whether a given test activity has been completed or NOT. Exit criteria can be defined for all of the test activities right from planning, specification and execution.

1. **Expected Outcome**

During Test design stage, test cases are written such that each case has an expected outcome against which the actual outcomes are compared. The deviation, if any, is known as **defect.**

1. **Exploratory testing**

During testing phase where there is severe time pressure, Exploratory testing technique is adopted that combines the experience of testers along with a structured approach to testing.

1. **Failover Testing**

Failover testing is a testing technique that validates a system's ability to be able to allocate extra resource and to move operations to back-up systems during the server failure due to one or the other reasons. This determines if a system is capable of handling extra resource such as additional CPU or servers during critical failures or at the point the system reaches a performance threshold.

1. **Failure**

Under certain circumstances, the product may produce wrong results. It is defined as the deviation of the delivered service from compliance with the specification.

1. **Fault**

Software fault is also known as defect, arises when the expected result don't match with the actual results. It can also be error, flaw, failure, or fault in a computer program. Most bugs arise from mistakes and errors made by developers, architects.

1. **Fault injection Testing**

Fault injection is a software testing technique by introducing faults into the code for improving the coverage and usually used with stress testing for robustness of the developed software.

1. **Feasible path**

A control flow path through a function or a procedure is possible if there is an assignment to input values which drives execution down the path. We can also make use of a regular expression to describe a set of paths.

1. **Feature Testing**

A Software feature can be defined as the changes made in the system to add new functionality or modify the existing functionality. Each feature is said to have a characteristics that is designed to be useful, intuitive and effective.

1. **Functional Decomposition**

Functional decomposition corresponds to the various functional relationships as how the original complex business function was developed. It mainly focusses on how the overall functionality is developed and its interaction between various components.

1. **Functional Requirement**

A functional requirement document defines the functionality of a system or one of its subsystems. It also depends upon the type of software, expected users and the type of system where the software is used.

Functional user requirements may be high-level statements of what the system should do but functional system requirements should also describe clearly about the system services in detail.

1. **Functional Testing**

Functional Testing is a testing technique that is used to test the features/functionality of the system or Software, should cover all the scenarios including failure paths and boundary cases.

1. **Fuzz Testing**

Fuzz testing is a software testing technique using which a random data is given as the inputs to the system. If the application fails, then those issues/defects are to be addressed by the system. In short, unexpected or random inputs might lead to unexpected results.

1. **Glass Box Testing**

Glass box testing is a testing technique that examines the program structure and derives test data from the program logic/code. The other names of glass box testing are clear box testing, open box testing, logic driven testing or path driven testing or structural testing.

1. **Gorilla Testing**

Gorilla Testing is a testing technique in which testers, sometimes developers also join hands with testers to test a particular module thoroughly in all aspects.

Gorilla Testing, a technique in which repetitive Manual Testing process, which a tester would have done several times before, is done again to test the robustness of the system.

1. **Grey Box Testing**

Grey Box testing is testing technique performed with limited information about the internal functionality of the system. Grey Box testers have access to the detailed design documents along with information about requirements.

1. **GUI Software Testing**

GUI testing is a testing technique in which the application's user interface is tested whether the application performs as expected with respect to user interface behaviour.

GUI Testing includes the application behaviour towards keyboard and mouse movements and how different GUI objects such as toolbars, buttons, menubars, dialog boxes, edit fields, lists, behavior to the user input.

1. **Harness**

Test Harness, also known as automated test framework mostly used by developers. A test harness provides stubs and drivers, which will be used to replicate the missing items, which are small programs that interact with the software under test.

1. **Hybrid Integration Testing**

We know that Integration Testing is a phase in software testing in which standalone modules are combined and tested as a single entity. During that phase, the interface and the communication between each one of those modules are tested. There are two popular approaches for Integration testing which is Top down Integration Testing and Bottom up Integration Testing.

1. **Implementation Testing**

Let us first understand what implementation means. Implementation is the process of putting an action for the formulated plan. Before we implement, the plan should have been completed and our objectives should be clear.

Testing each one of those actions formulated in the plan is said to be implementation testing.

1. **Incremental Testing**

After unit testing is completed, developer performs integration testing. It is the process of verifying the interfaces and interaction between modules. While integrating, there are lots of techniques used by developers and one of them is the incremental approach.

1. **Independent Testing**

Independent testing corresponds to an independent team, who involve in testing activities other than developer to avoid author bias and is often more effective at finding defects and failures.

1. **Infeasible Testing**

Infeasible Path in software testing context can be defined as the path that cannot be verified by any set of possible input values and most expensive activities of software testing.

1. **Inspection**

Inspection is the most formal form of reviews, a strategy adopted during static testing phase.

1. Install/Unistall Testing

**Installation Testing:** It is performed to verify if the software has been installed with all the necessary components and the application is working as expected. This is very important as installation would be the first user interaction with the end users.

**Uninstallation Testing:** Uninstallation testing is performed to verify if all the components of the application is removed during the process or NOT. All the files related to the application along with its folder structure have to be removed upon successful uninstallation. Post Uninstallation System should be able to go back to the stable state.

1. **Integration Testing**

Upon completion of unit testing, the units or modules are to be integrated which gives raise to integration testing. The purpose of integration testing is to verify the functional, performance, and reliability between the modules that are integrated.

1. **Interface Testing**

Interface Testing is performed to evaluate whether systems or components pass data and control correctly to one another. It is to verify if all the interactions between these modules are working properly and errors are handled properly.

1. **Internalization Testing**

Internationalization testing is the process of verifying the application under test to work uniformly across multiple regions and cultures.

1. **Inter System Testing**

Many a times, an application is hosted across locations; however, all data needs to be deployed over a central location. The process of testing the integration points for single application hosted at different locations and then ensuring correct data flow across each location is known as inter system testing.

1. **Isolation Testing**

Isolation testing is the process of breaking down the system into various modules so that defects can be spotted easily in isolation. It happens especially when the bug is difficult to locate and resolve by development team.

1. **Issue**

In the field of software testing, the terminologies such as issue, defect and bug are used interchangeably. However, Issue can be defined as the unit of work to accomplish an improvement in a system. It could be a bug, a change request, task, missing documentation, etc. It is usually raised by specifying the severity (high, medium, low or cosmetic).

1. **Keyboard Driven Testing**

Keyword-driven testing is a type of functional automation testing framework which is also known as table-driven testing or action word based testing.

1. **Key Performance Indicator**

A Key Performance Indicator (or KPI) is usually used to evaluate the software process efficiency evaluation. The important parameters and their usage are analysed and the outcome of the measurement is used to trigger any process improvements.

1. **Known Issues**

Sometimes after code freeze, there might be medium or low priority defects, which cannot be fixed due to time constraints or budgets. Hence, the software would be released into the market with the known defects/issues.

1. **LCSAJ Testing**

LCSAJ stands for Linear Code Sequence and Jump, a white box testing technique to identify the code coverage, which begins at the start of the program or branch and ends at the end of the program or the branch.

1. **Load Generator**

Load Generator is a system, which is used to simulate load for performing performance testing. It can be for concurrency testing or SQL performance Testing. It is a system, which sends a request remotely called as host system or load driving system.

1. **Load Testing**

Load testing is performance testing technique using which the response of the system is measured under various load conditions. The load testing is performed for normal and peak load conditions.

1. **Localization Testing**

Localization testing is performed to verify the quality of a product's localization for a particular target culture/locale and is executed only on the localized version of the product.

1. **Logic Coverage Testing**

Logic corresponds to the internal structure of the code and this testing is adopted for safety-critical applications such as software’s used in aviation industry. This Test verifies the subset of the total number of truth assignments to the expressions.

1. **Maintainability Cycle**

The term maintainability corresponds to the ability to update or modify the system under test. This is a very important parameter as the system is subjected to changes throughout the software life cycle.

1. **Manual Testing**

Manual testing is a testing process that is carried out manually in order to find defects without the usage of tools or automation scripting.

1. **Model Based Testing**

Model-based testing is a software testing technique in which the test cases are derived from a model that describes the functional aspects of the system under test.

1. **Modified Condition Coverage**

The Modified Condition/Decision Coverage enhances the condition/decision coverage criteria by requiring that each condition be shown to independently affect the outcome of the decision. This kind of testing is performed on mission critical application which might lead to death, injury or monetary loss.

1. **Modularity Driven Testing**

Modularity driven testing is an automation testing framework in which small, independent modules of automation scripts are developed for the application under test. These individual scripts are constructed together to form a test realizing a particular test case.

1. **Monkey Testing**

Monkey testing is a software testing technique in which the testing is performed on the system under test randomly. The Input data that is used to test also generated randomly and keyed into the system.

1. **Mutation Testing**

Mutation testing is a structural testing technique, which uses the structure of the code to guide the testing process. On a very high level, it is the process of rewriting the source code in small ways in order to remove the redundancies in the source code.

1. **Negative Testing**

Negative testing is performed to ensure that the product or application under test does NOT fail when an unexpected input is given. The purpose of Negative testing is to break the system and to verify the application response during unintentional inputs.

1. **Non-Functional Testing**

Non-Functional testing is a software testing technique that verifies the attributes of the system such as memory leaks, performance or robustness of the system. Non-Functional testing is performed at all test levels.

1. **Operational Testing**

Operational acceptance testing (OAT), a testing technique performed to verify the operational readiness (pre-release) of a product or application under test as part of Software test life cycle. This testing technique mainly focusses on operational readiness of the system, which is supposed to mimic the production environment.

1. **Orthogonal Array Testing**

Orthogonal array testing is a systematic and statistical way of a black box testing technique used when number of inputs to the application under test is small but too complex for an exhaustive testing.

1. **Pair Testing**

Pair Testing is a software testing technique in which two people test the same feature at the same place at same time by continuously exchanging ideas. It generates more ideas which result in better testing of the application under test.

1. **Pairwise Testing**

Pairwise Testing also known as All-pairs testing is a testing approach taken for testing the software using combinatorial method. It's a method to test all the possible discrete combinations of the parameters involved.

1. **Parallel Testing**

Parallel testing is a testing technique in which the same inputs are entered in two different versions of the application and reporting the anomalies.

1. **Partial test Automation**

In software testing, there are many processes that do NOT fit the need of complete automation. Partial automation allows the organizations to observe the benefits of automation without the involvement of too much cost and time.

There are few test cases that cannot be automated for many reasons. In such cases, the automation is done to the level that is very much beneficial and the rest of the tests are executed manually.

1. **Passive Testing**

Passive testing is a software testing technique that observes the system without interaction. On the other hand, active testing involves interaction with the system.

1. **Path Testing**

Path Testing is a structural testing method based on the source code or algorithm and NOT based on the specifications. It can be applied at different levels of granularity.

1. **Peer Review**

A peer review, a review technique, which is a static white-box testing which are conducted to spot the defects early in the life cycle that cannot be detected by black box testing techniques.

1. **Penetration Testing**

Penetration testing a black box testing technique in which an authorized attempt is made to violate specific constraints stated in the form of a security or integrity policy of the system, application, network or database. It is a testing technique for discovering and documenting all the security holes that can be found in a system.

1. **Performance Testing**

Performance testing, a non-functional testing technique performed to determine the system parameters in terms of responsiveness and stability under various workload. Performance testing measures the quality attributes of the system, such as scalability, reliability and resource usage.

1. **Portability Testing**

Portability testing is a process of testing with ease with which the software or product can be moved from one environment to another. It is measured in terms of maximum amount of effort required to transfer from one system to another system.

1. **Positive Testing**

Positive testing is a testing technique to show that a product or application under test does what it is supposed to do. Positive testing verifies how the application behaves for the positive set of data.

1. **Post Condition**

Post Condition is a statement or set of statements describing the outcome of an action if true when the operation has completed its task.

1. **Pre-Condition**

Pre-condition is a statement or set of statements that outline a condition that should be true when an action is called. The precondition statement indicates what must be true before the function is called.

1. **Predicted Outcome**

During Test design stage, test cases are written such that each case has an expected or predicted outcome against which the actual outcomes are compared. The deviation, if any, is known as **defect.**

1. **Process Cycle Test**

Process cycle Test, a black box test design technique in which test cases are designed to execute business scenarios.

1. **Progressive Testing**

Progressive testing also known as incremental testing is used to test modules one after the other. When an application with a hierarchy such as parent-child module is being tested, the related modules would need to be tested first.

1. **Prototype Testing**

Prototype Testing is conducted with the intent of finding defects before the website goes live. Online Prototype Testing allows seamlessly to collect quantitative, qualitative, and behavioural data while evaluating the user experience.

1. **Quality Assurance**

Quality Assurance is defined as the auditing and reporting procedures used to provide the stakeholders with data needed to make well-informed decisions.

1. **Quality Control**

Quality control is a set of methods used by organizations to achieve quality parameters or quality goals and continually improve the organization's ability to ensure that a software product will meet quality goals.

1. **Software Quality Management**

Software Quality Management ensures that the required level of quality is achieved by submitting improvements to the product development process. SQA aims to develop a culture within the team and it is seen as everyone's responsibility.

1. **Random Testing**

Random Testing, also known as monkey testing, is a form of functional black box testing that is performed when there is not enough time to write and execute the tests.

1. **Recovery Testing**

Recovery testing is a type of non-functional testing technique performed in order to determine how quickly the system can recover after it has gone through system crash or hardware failure. Recovery testing is the forced failure of the software to verify if the recovery is successful.

1. **Regression Testing**

Regression testing a black box testing technique that consists of re-executing those tests that are impacted by the code changes. These tests should be executed as often as possible throughout the software development life cycle.

1. **Release Condition**

Release Candidate (RC) is the build released internally to check if any critical problems have gone undetected into the code during the previous development period. Release candidates are NOT for production deployment, but they are for testing purposes only. However, in most of the cases, there are no differences between the final build and the last release candidate.

1. **Release Note**

Release notes is a document, which is released as part of the final build that contains new enhancements that went in as part of that release and also the known issues of that build.

1. **Reliability Testing**

Software reliability testing a testing technique that relates to testing a software's ability to function given environmental conditions consistently that helps uncover issues in the software design and functionality.

1. **Requirement**

The requirements are the high-level descriptions about a particular system services, constraints or to a detailed specification that are generated during the requirements gathering process.

1. **Requirement Based Testing**

Requirements-based testing is a testing approach in which test cases, conditions and data are derived from requirements. It includes functional tests and also non-functional attributes such as performance, reliability or usability.

1. **Requirement Traceability Testing**

Requirements tracing, a process of documenting the links between the requirements and the work products developed to implement and verify those requirements. The RTM captures all requirements and their traceability in a single document delivered at the conclusion of the life cycle.

1. **Result**

Result reporting is a mechanism with which the state of the product is presented to the customer from various angles

1. **Re-Testing**

Re-testing is executing a previously failed test against new software to check if the problem is resolved. After a defect has been fixed, re-testing is performed to check the scenario under the same environmental conditions.

1. **Review**

A review is a systematic examination of a document by one or more people with the main aim of finding and removing errors early in the software development life cycle. Reviews are used to verify documents such as requirements, system designs, code, test plans and test cases.

1. **Risk**

Risk can be defined as the probability of an event, hazard, accident, threat or situation occurring and its undesirable consequences. It is a factor that could result in negative consequences and usually expressed as the product of impact and likelihood.

1. **Risk Management**

Risk management is the process of identifying, assessing, and prioritizing the risks to minimize, monitor, and control the probability of unfortunate events.

1. **Root Cause**

Root Cause is the process of identifying the contributing factors for the underlying variations in performance associated with adverse events or close calls.

1. **Safety Testing**

Safety testing in software systems aims at optimizing system safety in the design, development, use, and maintenance of software systems and their integration with safety-critical hardware systems in a production environment.

1. **Sanity Testing**

Sanity testing, a software testing technique performed by the test team for some basic tests. The aim of basic test is to be conducted whenever a new build is received for testing. The terminologies such as Smoke Test or Build Verification Test or Basic Acceptance Test or Sanity Test are interchangeably used, however, each one of them is used under a slightly different scenario

1. **Scalability Testing**

Scalability, a performance testing parameter that investigates a system's ability to grow by increasing the workload per user, or the number of concurrent users, or the size of a database.

1. **Scenario Testing**

Scenario testing is a software testing technique that makes best use of scenarios. Scenarios help a complex system to test better where in the scenarios are to be credible which are easy to evaluate.

1. **Script**

A script is a set of instructions or commands of a certain program or scripting engine used to automate the process on the system it is executed.

1. **Security Testing**

Security testing is a testing technique to determine if an information system protects data and maintains functionality as intended

1. **Simulation**

A simulation is a computer model that mimics the operation of a real or proposed system and it is time based and takes into account all the resources and constraints involved.

1. **Smoke Testing**

Smoke Testing is a testing technique that is inspired from hardware testing, which checks for the smoke from the hardware components once the hardware's power is switched on. Similarly in Software testing context, smoke testing refers to testing the basic functionality of the build.

1. **Soak testing**

Soak Testing also known as Endurance Testing is performed to determine if the application under test can sustain the continuous loads.

1. **Software Requirement Specification**

A software requirements specification (SRS) is a document that captures complete description about how the system is expected to perform. It is usually signed off at the end of requirements engineering phase.

1. **Stability**

Stability testing, a software testing technique adopted to verify if application can continuously perform well within or just above the acceptable period.

1. **State Transition Testing**

State Transition testing, a black box testing technique, in which outputs are triggered by changes to the input conditions or changes to 'state' of the system. In other words, tests are designed to execute valid and invalid state transitions.

1. **Static Testing**

Static Testing, a software testing technique in which the software is tested without executing the code

1. **Statistical Testing**

Statistical Testing makes use of statistical methods to determine the reliability of the program. Statistical testing focuses on how faulty programs can affect its operating conditions.

1. **Storage Testing**

Storage testing is a testing technique to verify the application under test, stores the relevant data in the appropriate directories and it has sufficient space to prevent unexpected termination due to insufficient disk space.

1. **Stress Testing**

Stress testing a Non-Functional testing technique that is performed as part of performance testing. During stress testing, the system is monitored after subjecting the system to overload to ensure that the system can sustain the stress.

1. **Structural Testing**

Structural testing, also known as glass box testing or white box testing is an approach where the tests are derived from the knowledge of the software's structure or internal implementation.

1. **Structural Walkthrough**

A structured walkthrough, a static testing technique performed in an organized manner between a group of peers to review and discuss the technical aspects of software development process. The main objective in a structured walkthrough is to find defects inorder to improve the quality of the product.

1. **Stub**

Stubs are used during Top-down integration testing, in order to simulate the behaviour of the lower-level modules that are not yet integrated. Stubs are the modules that act as temporary replacement for a called module and give the same output as that of the actual product.

1. **Symbolic Execution**

Symbolic execution is a software testing technique that is useful to aid the generation of test data and in proving the program quality.

1. **Syntax Testing**

Syntax Testing, a black box testing technique, involves testing the System inputs and it is usually automated because syntax testing produces a large number of tests.

1. **Syntax Integration Testing**

System Integration Testing(SIT) is a black box testing technique that evaluates the system's compliance against specified requirements. System Integration Testing is usually performed on subset of system while system testing is performed on a complete system and is preceded by the user acceptance test (UAT).

1. **System Testing**

System Testing (ST) is a black box testing technique performed to evaluate the complete system the system's compliance against specified requirements. In System testing, the functionalities of the system are tested from an end-to-end perspective.

1. **System Under Test**

System under test (SUT) refers to a system that is being validated by the testers. The terminology is also known as application under test

1. **Technical Review**

A Technical review is a static white-box testing technique which is conducted to spot the defects early in the life cycle that cannot be detected by black box testing techniques

1. **Test Approach**

A test approach is the test strategy implementation of a project, defines how testing would be carried out.

1. **Test Automation**

Software Test automation makes use of specialized tools to control the execution of tests and compares the actual results against the expected result. Usually, regression tests, which are repetitive actions, are automated.

1. **Test Basis**

Test basis is defined as the source of information or the document that is needed to write test cases and also for test analysis.

1. **Test Bed**

The test execution environment configured for testing. Test bed consists of specific hardware, software, Operating system, network configuration, the product under test, other system software and application software.

1. **Test Case**

A test case is a document, which has a set of test data, preconditions, expected results and postconditions, developed for a particular test scenario in order to verify compliance against a specific requirement.

1. **Test Suite**

Test suite is a container that has a set of tests which helps testers in executing and reporting the test execution status. It can take any of the three states namely Active, Inprogress and completed.

1. **Test Completion Criteria**

A check against the test exit criteria is very much essential before we claim that the testing is completed. Before putting an end to test process the product quality is measured against the test completion criteria.

1. **Test Completion report**

Test completion reporting is a process where by test metrics are reported in summarised format to update the stakeholders which enables them to take an informed decision.

1. **Test Data**

Test Data is data that is used to execute the tests on testware. Test data needs to be precise and exhaustive to uncover the defects.

1. **Test data Management**

Test Data management is very critical during the test life cycle. The amount of data that is generated is enormous for testing the application. Reporting the results it minimizes the time spent for processing the data and creating reports greatly contributes to the efficiency of an entire product.

1. **Test Driven Development**

Test-driven development starts with developing test for each one of the features. The test might fail as the tests are developed even before the development. Development team then develops and refactors the code to pass the test.

1. **Test Driver**

Test Drivers are used during Bottom-up integration testing in order to simulate the behaviour of the upper level modules that are not yet integrated. Test Drivers are the modules that act as temporary replacement for a calling module and give the same output as that of the actual product.

1. **Test Environment**

Test Environment consists of elements that support test execution with software, hardware and network configured. Test environment configuration must mimic the production environment in order to uncover any environment/configuration related issues.

1. **Test Execution**

Test execution is the process of executing the code and comparing the expected and actual results

1. **Test Management**

Test management, process of managing the tests. A test management is also performed using tools to manage both types of tests, automated and manual, that have been previously specified by a test procedure.

1. **Test Maturity Model**

Test maturity model is based on capability maturity model specifies an increasing series of levels of a software development organization. The higher the level, the better the software development process, hence reaching each level is an expensive and time-consuming process.

1. **Test Plan**

Test planning, the most important activity to ensure that there is initially a list of tasks and milestones in a baseline plan to track the progress of the project. It also defines the size of the test effort.

1. **Test Steps**

Test Steps describe the execution steps and expected results that are documented against each one of those steps.

1. **Test Strategy**

Test Strategy is also known as test approach defines how testing would be carried out.

1. **Testing Tools**

Tools from a software testing context can be defined as a product that supports one or more test activities right from planning, requirements, creating a build, test execution, defect logging and test analysis.

1. **Thread Testing**

A thread is the smallest unit of work that a system can execute. Thread testing, a software testing technique used during early integration testing phase to verify the key functional capabilities that carry out specific task. These kinds of techniques are very helpful if an application is of type that uses client server architecture.

Performing Thread testing on valid business transaction through the integrated client, server and network is very critical. Threads are integrated and tested incrementally as subsystems and then performed as a whole system.

1. **Top Down Integration Testing**

Top-down integration testing is an integration testing technique used in order to simulate the behaviour of the lower-level modules that are not yet integrated. Stubs are the modules that act as temporary replacement for a called module and give the same output as that of the actual product.

1. **Total Quality Management**

TQM can be defined as a management technique for improving processes, products, services and the other approaches associated with the product. It focusses on the entire business and NOT just on a particular project or process.

1. **Traceablity**

The significance of traceability within a requirement tool or a test management tool (like HP Quality Center) enables links between requirements and tests. This notifies what needs to be changed when a requirement changes.

1. **Unit testing**

Unit testing, a testing technique using which individual modules are tested to determine if there are any issues by the developer himself. It is concerned with functional correctness of the standalone modules.

1. **Unreachable Code**

Unreachable code, a part of the source code that will never be executed due to inappropriate exit points/control flow. The other kind of unreachable code is referred as dead code, although dead code might get executed but has no effect on the functionality of the system.

1. **Usability Testing**

Usability testing, a non-functional testing technique that is a measure of how easily the system can be used by end users.

1. **Use Case Testing**

Use Case Testing is a functional black box testing technique that helps testers to identify test scenarios that exercise the whole system on each transaction basis from start to finish.

1. **User Acceptance testing**

User acceptance testing, a testing methodology where the clients/end users involved in testing the product to validate the product against their requirements. It is performed at client location at developer's site.

1. **User Interface Testing**

User interface testing, a testing technique used to identify the presence of defects is a product/software under test by using Graphical user interface [GUI].

1. **Validation**

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements.

1. **Verification**

Verification is the process of evaluating work-products of a development phase to determine whether they meet the specified requirements.

1. **Virtual Users**

Virtual user is a common terminology used from a performance testing point of view. A Virtual user generator enables testers to create virtual users in order to increase the user load on the application under test.

1. **Volume Testing**

Volume testing is a Non-functional testing that is performed as part of performance testing where the software is subjected to a huge volume of data. It is also referred as flood testing.

1. **Vulnerability testing**

A software testing technique performed to evaluate the quantum of risks involved in the system in order to reduce the probability of the event

1. **Web Application Testing**

Web application testing, a software testing technique exclusively adopted to test the applications that are hosted on web in which the application interfaces and other functionalities are tested.

1. **White Box Testing**

White box testing is a testing technique, that examines the program structure and derives test data from the program logic/code. The other names of glass box testing are clear box testing, open box testing, logic driven testing or path driven testing or structural testing.

1. **Workflow Testing**

Workflow processes technique in software testing by routing a record through each possible path. These tests are performed to ensure that each workflow process accurately reflects the business process.