1. What is Exploratory Testing?

* Exploratory testing is an approach to software testing that is characterized by simultaneous learning, test design, and test execution.

1. What is traceability matrix?

* A traceability matrix is a document used in software development and testing to ensure that all requirements are linked to the corresponding test cases and vice versa.

1. What is Boundary value testing?

* Boundary value testing is a software testing technique used to evaluate the behaviour of a system at the boundaries of input domains.

1. What is Equivalence partitioning testing?

* Equivalence partitioning testing is a software testing technique that divides the input data of a software system into groups (or partitions) that are expected to exhibit similar behaviour.

1. What is Integration testing?

* Integration testing is a level of software testing where individual units or components of a system are combined and tested as a group.

1. What determines the level of risk?

* The level of risk in software testing is determined by various factors related to the software being developed, the project environment, and the testing process itself.

1. What is Alpha testing?

* Alpha testing is a type of software testing performed by internal testers, usually within the organization developing the software.

1. What is beta testing?

* Beta testing is a type of software testing conducted by a selected group of external users or customers in a real-world environment.

1. What is component testing?

* Component testing, also known as unit testing or module testing, is a level of software testing where individual units or components of a software system are tested in isolation.

1. What is functional system testing?

* Functional system testing is a type of software testing that focuses on verifying the functionality of a complete software system or application.

1. What is Non-Functional Testing?

* Non-functional testing, also known as quality characteristics testing or attribute testing, is a type of software testing that focuses on evaluating the non-functional aspects of a software system.

1. What is GUI Testing?

* GUI (Graphical User Interface) testing is a type of software testing that focuses on verifying the functionality, usability, and visual appearance of the graphical user interface of a software application.

1. What is Ad hoc testing?

* Ad hoc testing is a software testing technique that is performed informally and spontaneously, without following a predefined test plan or test cases.

1. What is load testing?

* Load testing is a type of performance testing that assesses the behavior of a software system under specific load conditions.

1. What is stress Testing?

* Load testing is a type of performance testing that evaluates how a system behaves and performs under anticipated user loads and expected usage patterns.

1. What is white box testing and list the types of white box testing?

* White box testing, also known as clear box testing, glass box testing, or structural testing, is a software testing technique that examines the internal structure, logic, and code of a software application.
* Types of white box testing include:
* **Statement Coverage Testing**
* **Branch Coverage Testing**
* **Path Coverage Testing**
* **Condition Coverage Testing**
* **Loop Coverage Testing**
* **Data Flow Testing**
* **Mutation Testing**:

1. What is black box testing?

* Black box testing is a software testing technique that focuses on assessing the functionality and behaviour of a software application without examining its internal code, structure, or implementation details.

1. What are the different black box testing techniques?

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

1. Mention what are the categories of defects?

* **Functional Defects**
* **Interface Defects**
* **Usability Defects**
* **Performance Defects**
* **Security Defects**
* **Compatibility Defects**
* **Data Defects**
* **Documentation Defects**
* **Concurrency Defects**
* **Regression Defects**

1. Mention what big bang testing is?

* "Big bang testing" is a software testing approach where all or most of the components or modules of a software application are tested simultaneously.

1. What is the purpose of exit criteria?

* Exit criteria in software testing refer to the predefined conditions or benchmarks that must be met before concluding a testing phase or activity and proceeding to the next phase of the software development lifecycle.

1. When should "Regression Testing" be performed?  What is 7 key principles? Explain in detail?

* Regression testing should be performed whenever changes are made to the software, including modifications, enhancements, bug fixes, or updates.
* **Scope Management**: Define the scope of regression testing based on the nature and impact of the changes made to the software. Focus regression testing efforts on areas of the software that are directly affected by the changes, while also considering dependencies and potential ripple effects on other parts of the system.
* **Prioritization**: Prioritize regression testing based on the criticality of the affected functionality, the likelihood of regression, and the importance of maintaining system stability. Allocate testing resources and effort accordingly to ensure that high-risk and high-impact areas receive adequate attention during regression testing.
* **Automation**: Automate regression testing wherever possible to increase efficiency, repeatability, and coverage. Identify repetitive and time-consuming test cases that can be automated and implement automated test scripts or test suites to streamline regression testing efforts and reduce manual effort.
* **Comprehensive Coverage**: Aim for comprehensive coverage of regression test cases to verify that all critical and relevant functionality remains intact after changes are made to the software. Include a mix of functional, non-functional, and integration test cases to ensure thorough testing of the software across different dimensions.
* **Version Control and Baselines**: Maintain version control of the software and establish baselines for regression testing to track changes and compare different versions of the software. Use version control systems to manage code changes, track revisions, and facilitate rollback to previous versions if necessary.
* **Traceability**: Establish traceability between requirements, test cases, and defects to ensure that regression testing aligns with the intended functionality and objectives of the software. Trace test cases back to their corresponding requirements and document any deviations, defects, or changes encountered during regression testing.
* **Continuous Improvement**: Continuously evaluate and refine regression testing processes and strategies to optimize effectiveness, efficiency, and coverage. Gather feedback from regression testing activities, identify areas for improvement, and implement lessons learned to enhance regression testing practices over time.

1. Difference between QA v/s QC v/s Tester ?

|  |  |  |
| --- | --- | --- |
| QA | QC | TESTER |
| QA is a proactive process focused on preventing defects and ensuring that the software development process follows established standards, processes, and best practices. | QC is a reactive process focused on identifying defects and verifying that the software meets quality standards and requirements. | A Tester is an individual responsible for executing test cases, identifying defects, and verifying that the software meets quality standards and requirements. |
| QA activities involve defining quality standards, processes, and methodologies, as well as establishing quality objectives and metrics. | QC activities involve testing, inspecting, and evaluating the software product to detect defects, deviations, or non-conformities against specified requirements and quality criteria. | Testers are involved in both QA and QC activities, depending on their role and responsibilities within the software development lifecycle. |
| QA encompasses activities such as defining quality criteria, establishing quality assurance plans, conducting reviews and audits, and promoting a culture of quality throughout the organization. | QC encompasses activities such as functional testing, non-functional testing, regression testing, and defect tracking and management. | Testers collaborate with QA professionals, developers, and other stakeholders to define test objectives, design test cases, execute test scripts, report defects, and ensure that the software product meets quality objectives. |

1. Difference between Smoke and Sanity?

|  |  |
| --- | --- |
| Smoke | Sanity |
| Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine | Sanity Testing is done to check the new functionality/bugs have been fixed |
| This testing is performed by the developers or testers | Sanity testing in software testing is usually performed by testers |
| Smoke testing is usually documented or scripted | Sanity testing is usually not documented and is unscripted |
| Smoke testing is a subset of Acceptance testing | Sanity testing is a subset of [Regression Testing](https://www.guru99.com/regression-testing.html) |

1. Difference between verification and Validation

|  |  |
| --- | --- |
| verification | Validation |
| It includes checking documents, design, codes and programs. | It includes testing and validating the actual product. |
| Verification is the static testing. | Validation is the dynamic testing. |
| It does *not* include the execution of the code | It includes the execution of the code. |
| Methods used in verification are reviews, walkthroughs, inspections and desk-checking. | Methods used in validation are Black Box Testing, White Box Testing and non-functional testing. |

1. Explain types of Performance testing.

* Performance testing is a crucial aspect of software testing that evaluates the responsiveness, scalability, reliability, and resource usage of a software application under various conditions. There are several types of performance testing, each serving different purposes and focusing on specific aspects of the application's performance. Here are the main types of performance testing:
* **Load Testing**: Load testing assesses the behavior of the system under anticipated loads or expected usage levels. The goal is to determine whether the application can handle the specified load levels without performance degradation or system failures. Load testing helps identify performance bottlenecks, scalability issues, and resource constraints by simulating realistic user loads and monitoring system response times, throughput, and resource utilization.
* **Stress Testing**: Stress testing evaluates the application's behavior under extreme or beyond-normal conditions, such as high user loads, heavy data volumes, or peak traffic levels. The objective is to identify the breaking point or saturation point of the system and determine how it responds under stress. Stress testing helps uncover stability issues, memory leaks, resource exhaustion, and other performance-related defects that may occur under extreme conditions.
* **Endurance Testing** (or Soak Testing): Endurance testing, also known as soak testing, assesses the application's performance over an extended period under sustained loads or continuous usage. The goal is to identify performance degradation, memory leaks, and resource leaks that may occur over time due to prolonged operation. Endurance testing helps assess the application's reliability and stability under prolonged usage scenarios, such as continuous operation or long-duration transactions.
* **Scalability Testing**: Scalability testing evaluates the application's ability to handle increasing loads, data volumes, or user concurrency while maintaining performance, responsiveness, and throughput. The objective is to assess how well the system scales to accommodate growing demands and to identify scalability limitations or bottlenecks. Scalability testing helps determine the maximum capacity and optimal configuration of the system to meet future scalability requirements.
* **Volume Testing**: Volume testing assesses the application's performance and behavior when subjected to large volumes of data or transactions. The objective is to verify that the system can process, store, and retrieve large datasets efficiently without performance degradation or data corruption. Volume testing helps identify scalability limitations, database constraints, and data storage issues related to handling large volumes of data.
* **Concurrency Testing**: Concurrency testing evaluates the application's behavior under simultaneous user loads or concurrent transactions. The goal is to assess how well the system manages concurrent access to shared resources, such as database connections, files, or application components. Concurrency testing helps identify race conditions, deadlocks, and synchronization issues that may occur when multiple users access the system concurrently.
* **Compatibility Testing**: Compatibility testing assesses the application's performance across different platforms, operating systems, browsers, and devices. The objective is to verify that the application functions correctly and performs consistently across various configurations and environments. Compatibility testing helps ensure a seamless user experience and broadens the application's reach to different user segments.

1. What is Error, Defect, Bug and failure?

* **Error**: An error, also known as a mistake or fault, refers to a human action that produces an incorrect result.
* **Defect**: A defect, also called a fault or flaw, is a deviation or imperfection in a software product or component that can cause the software to behave incorrectly or produce incorrect results.
* **Bug**: A bug is a specific type of defect that causes a software system to fail to perform as intended.
* **Failure**: A failure occurs when a software system or component does not meet its specified requirements or behaves in a way that disrupts its intended function or service.

1. Difference between Priority and Severity. What is Bug Life Cycle?

* The Bug Life Cycle, also known as the Defect Life Cycle, is a sequence of states that a bug goes through from its discovery to its resolution in a software development process.

|  |  |
| --- | --- |
| Priority | Severity |
| Severity is a term that denotes how severely a defect can affect the functionality of the software. | Priority is a term that defines how fast we need to fix a defect. |
| Severity is basically a parameter that denotes the total impact of a given defect on any software. | Priority is basically a parameter that decides the order in which we should fix the defects. |
| Severity relates to the standards of quality. | Priority relates to the scheduling of defects to resolve them in software. |
| The value of severity is objective. | The value of priority is subjective. |

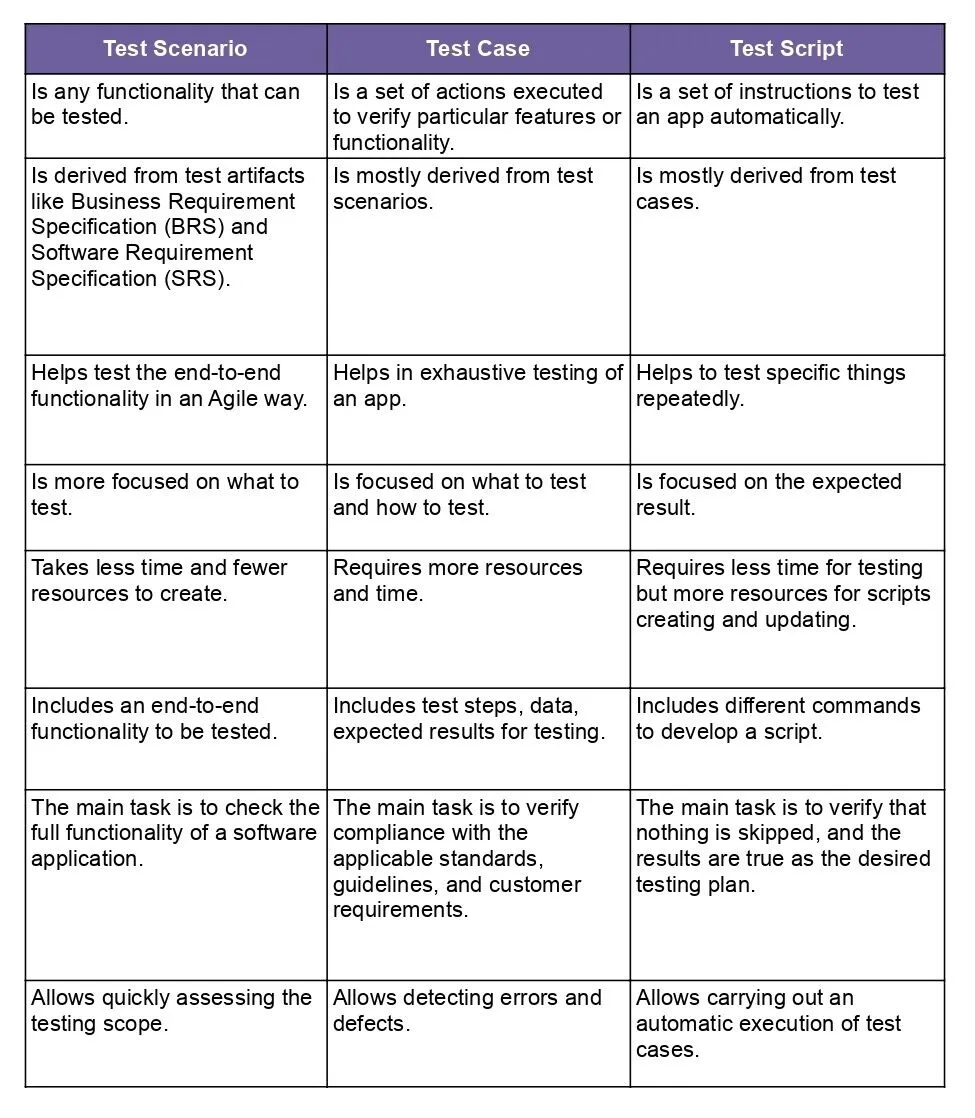
1. Explain the difference between Functional testing and Non-functional testing

|  |  |
| --- | --- |
| Functional testing | Non-functional testing |
| It verifies the operations and actions of an application. | It verifies the behaviour of an application. |
| It is based on requirements of customer. | It is based on expectations of customer. |
| It helps to enhance the behaviour of the application. | It helps to improve the performance of the application. |
| Functional testing is easy to execute manually. | It is hard to execute non-functional testing manually. |

30 What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)?

|  |  |
| --- | --- |
| SDLC | STLC |
| SDLC is mainly related to software development. | STLC is mainly related to software testing. |
| Besides development other phases like testing is also included. | It focuses only on testing the software. |
| SDLC involves total six phases or steps. | STLC involves only five phases or steps. |
| In SDLC, more number of members (developers) are required for the whole process. | In STLC, less number of members (testers) are needed. |

31 What is the difference between test scenarios, test cases, and test script?



32. Explain what Test Plan is? What isthe information that should be covered.

* A test plan is a document that outlines the approach, objectives, resources, and schedule for a specific testing effort. It serves as a blueprint for the testing process, guiding testers on how to validate that a software application or system meets its requirements and functions correctly.
* Here's the information that a typical test plan should cover:
* **Introduction**: Provides an overview of the document, including the purpose of the test plan, scope, objectives, and the software or system being tested.
* **Scope**: Defines what will be tested and what will not be tested. It sets boundaries for the testing effort, including specific functionalities, features, modules, or components.
* **Objectives**: States the goals and aims of the testing effort, such as verifying functionality, ensuring performance, assessing reliability, and validating usability.
* **Roles and Responsibilities**: Identifies the individuals or teams involved in the testing process, along with their roles, responsibilities, and contact information.
* **Test Approach**: Describes the overall strategy and methodology for testing, including the types of testing to be performed (e.g., functional testing, performance testing, security testing), the testing techniques to be used, and any tools or frameworks employed.
* **Test Environment**: Specifies the hardware, software, and network configurations required for testing, including details such as operating systems, browsers, databases, and network setups.
* **Test Deliverables**: Lists the documents, reports, and artifacts that will be produced during the testing process, such as test cases, test scripts, test results, defect reports, and test summary reports.
* **Test Schedule**: Outlines the timeline and milestones for the testing effort, including start and end dates, planned testing activities, and key deadlines.
* **Resource Requirements**: Identifies the human, financial, and infrastructural resources needed for testing, including personnel, equipment, tools, and budget allocations.
* **Risks and Assumptions**: Identifies potential risks and assumptions that may impact the testing process, such as resource constraints, schedule delays, technical dependencies, and external factors.
* **Approvals**: Specifies the stakeholders who need to review and approve the test plan before testing begins, along with any sign-off procedures.
* **Appendices**: Includes any additional reference materials, templates, or supporting documentation relevant to the testing effort.

33 What is priority?

* Priority in testing refers to the level of importance or urgency assigned to a particular test case, defect, or requirement

34 What is severity?

* Severity in testing refers to the degree of impact that a defect, bug, or issue has on the functionality or usability of the software.

35 Bug categories are…

* **Performance Bugs:**
* **Security Bugs:**
* **Unit Level Bugs:**
* **Functional Bugs:**
* **Usability Bugs:**
* **Compatibility Errors:**
* **Logic Bugs:**

36 Advantage of Bugzila .

* It improves the quality of the product.
* It enhances the communication between the developing team and the testing team.
* It has the capability to adapt to multiple situations

37 Difference between priority and severity

|  |  |
| --- | --- |
| priority | severity |
| Priority is a term that defines how fast we need to fix a defect. | Severity is a term that denotes how severely a defect can affect the functionality of the software. |
| Priority is basically a parameter that decides the order in which we should fix the defects. | Severity is basically a parameter that denotes the total impact of a given defect on any software. |
| Priority relates to the scheduling of defects to resolve them in software. | Severity relates to the standards of quality. |
| The value of priority is subjective. | The value of severity is objective. |

38 What are the different Methodologies in Agile Development Model?

* In Agile development, various methodologies have emerged to facilitate the iterative and collaborative approach to software development. Some of the most widely used Agile methodologies include:
* **Scrum**: Scrum is one of the most popular Agile methodologies, characterized by short, time-boxed iterations called sprints. During each sprint, cross-functional teams work collaboratively to deliver a potentially shippable increment of the product. Scrum emphasizes roles (Product Owner, Scrum Master, and Development Team), artifacts (Product Backlog, Sprint Backlog, and Increment), and ceremonies (Sprint Planning, Daily Standup, Sprint Review, and Sprint Retrospective).
* **Kanban**: Kanban is a visual management method that focuses on continuous delivery and optimization of workflow. Work items are represented as cards on a Kanban board, which typically consists of columns representing different stages of the workflow (e.g., To Do, In Progress, Done). Teams use the board to visualize work, limit work in progress (WIP), and identify bottlenecks to improve flow.

39 Explain the difference between Authorization and Authentication in Web t esting.What are the common problems faced in Web testing?

|  |  |
| --- | --- |
| **Authentication** | **Authorization** |
| In the [authentication](https://www.geeksforgeeks.org/authentication-in-computer-network/) process, the identity of users are checked for providing the access to the system. | While in [authorization](https://www.geeksforgeeks.org/what-is-aaa-authentication-authorization-and-accounting/) process, a the person’s or user’s authorities are checked for accessing the resources. |
| In the authentication process, users or persons are verified. | While in this process, users or persons are validated. |
| It is done before the authorization process. | While this process is done after the authentication process. |
| It needs usually the user’s login details. | While it needs the user’s privilege or security levels. |
| Authentication determines whether the person is user or not. | While it determines **What permission does the user have?** |
| Generally, transmit information through an ID Token. | Generally, transmit information through an Access Token. |
| The OpenID Connect (OIDC) protocol is an authentication protocol that is generally in charge of user authentication process. | The OAuth 2.0 protocol governs the overall system of user authorization process. |
| Popular Authentication Techniques-   * Password-Based Authentication * Passwordless Authentication * 2FA/MFA (Two-Factor Authentication / Multi-Factor Authentication) * [Single sign-on (SSO)](https://www.geeksforgeeks.org/introduction-of-single-sign-on-sso/) * Social authentication | Popular Authorization Techniques-   * Role-Based Access Controls (RBAC) * [JSON web token (JWT) Authorization](https://www.geeksforgeeks.org/json-web-token-jwt/) * SAML Authorization * OpenID Authorization * OAuth 2.0 Authorization |
| The authentication credentials can be changed in part as and when required by the user. | The authorization permissions cannot be changed by user as these are granted by the owner of the system and only he/she has the access to change it. |
| The user authentication is visible at user end. | The user authorization is not visible at the user end. |
| The user authentication is identified with username, password, face recognition, retina scan, fingerprints, etc. | The user authorization is carried out through the access rights to resources by using roles that have been pre-defined. |
| **Example**: Employees in a company are required to authenticate through the network before accessing their company email. | **Example:** After an employee successfully authenticates, the system determines what information the employees are allowed to access. |

* **Ensuring cross browser compatibility**
* **Dealing with dynamic content**
* **Performance and scalability testing**
* **Security testing**
* **Test data management**
* **Communication and collaboration**

40 When to used Usability Testing?

* Usability testing ideas. Once you've got an idea, conduct usability testing before putting any design resources to work. ...
* Usability testing prototypes. ...
* Usability testing before launch. ...
* Usability testing after launch.

41 What is the procedure for GUI Testing?

* Create the model.
* Verify the inputs for the model.
* For the particular model analyze the expected result.
* Implement the tests.
* Balance the actual result with the expected result.