Q.1 What is 7 key principles? Explain in detail?

- Software testing is a procedure of implementing software or the application to identify the defects or bugs. For testing an application or software, we need to follow some principles to make our product defects free.

The seven different testing principles

1. Testing shows the presence of defects
2. Exhaustive Testing is not possible
3. Early Testing
4. Defect Clustering
5. Pesticide Paradox
6. Testing is context-dependent
7. Absence of errors fallacy

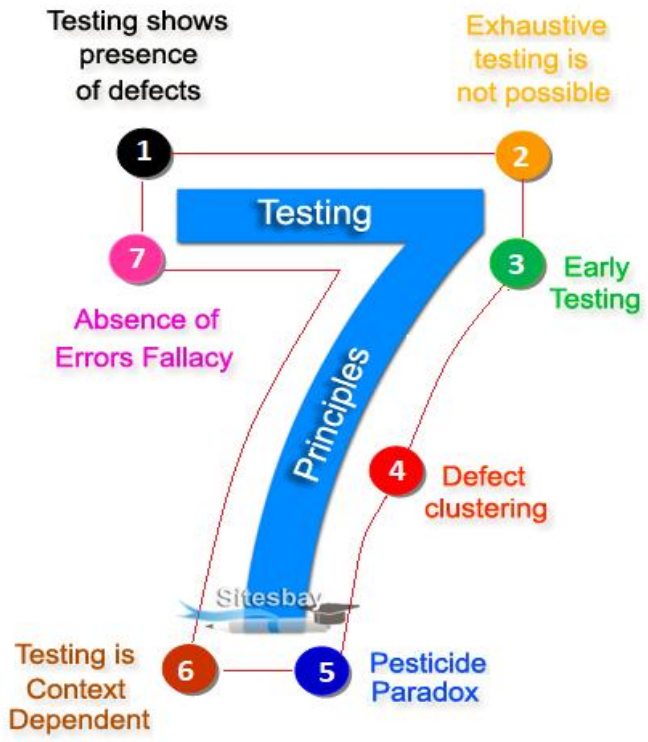


Fig: Software Testing Principles

- **Testing shows the presence of defects :-**

    - Testing talks about the presence of defects and don't talk about the absence of defects. I.e. Software testing reduces the probability of undiscovered defects remaining in the software but even if no defects are found, it is not a proof of correctness.
    - By doing testing on any application, we can decrease the number of bugs, which does not mean that the application is defect-free because sometimes the software seems to be bug-free while performing multiple types of testing on it.

    - But at the time of deployment in the production server, if the end-user encounters those bugs which are not found in the testing process.

- **Exhaustive Testing is not possible :-**

    - Exhaustive testing is not possible. Instead, we need the optimal amount of testing based on the risk assessment of the application.
    - Sometimes it seems to be very hard to test all the modules and their features with effective and non- effective combinations of the inputs data throughout the actual testing process.
    - Testing everything includes all combination of input and precondition is not possible.

- **Early Testing :-**

    - Early Testing – Testing should start as early as possible in the Software Development Life Cycle. So that any defects in the requirements or design phase are captured in early stages. It is much cheaper to fix a Defect in the early stages of testing. But how early one should start testing? It is recommended that you start finding the bug the moment the requirements are defined.
    - To perform testing, we will require the requirement specification documents; therefore, if the requirements are defined incorrectly, then it can be fixed directly rather than fixing them in another stage, which could be the development phase.

- **Defect Clustering :-**

    - Defect clustering which states that a small number of modules contain most of the defects detected. This is the application of the Pareto Principle to software testing approximately 80% of the problems are found in 20% of the modules.
    - Similar most operational failures of a system are usually confined to a smaller number of modules.
    - An important consideration in test prioritization.

- **Pesticide Paradox :-**

    - If we are executing the same set of test cases again and again over a particular time, then these kinds of the test will not be able to find the new bugs in the software or the application.
    - To overcome this, the test cases need to be regularly reviewed & revised, adding new & different test cases to help find more defects.
    - Testing identifies bugs, programmers respond to fix them. As bugs are eliminated by the programmers, the software improves.

- **Testing is context-dependent :-**

    - Testing is a context-dependent principle states that we have multiple fields such as e-commerce websites, commercial websites, and so on are available in the market.
    - There is a definite way to test the commercial site as well as the e-commerce websites because every application has its own needs, features, and functionality.
    - To check this type of application, we will take the help of various kinds of testing, different technique, approaches, and multiple methods. Therefore, the testing depends on the context of the application.

- **Absence of errors fallacy :-**

    - Once the application is completely tested and there are no bugs identified before the release, so we can say that the application is 99% bug-free.
    - But there is the chance when the application is tested beside the incorrect requirements, identified the flaws, and fixed them on a given period would not help as testing is done on the wrong specification, which does not apply to the client's requirements.
    - The absence of error fallacy means identifying and fixing the bugs would not help if the application is impractical and not able to accomplish the client's requirements and needs.

Q.2 Difference between verification and Validation.

Verification	Validation
Verification means checking the documents, design and other programming things.	Validation means testing the actual product.
Verification is also known as static testing.	Validation is also known as dynamic testing.
Requires a good understanding of the software requirements and specifications.	Requires a good understanding of end-user needs and expectations.
It can be perform automated and manual.	It can perform only manually.
Quality assurance comes under the verification testing.	Quality control comes under validation testing.
Verification does not involve the execution of the code.	In validation testing, the execution of code happens.
In verification testing, we can find the bugs early in the development phase of the product.	In the validation testing, we can find those bugs, which are not caught in the verification process.
It includes checking documents delivered by humans.	It includes the execution of a program executed by a computer.
Verification is done before the validation testing.	After verification testing, validation testing takes place.

Q.3 what is Error, Defect, Bug and failure?

Error:-

- The Problem in code leads to errors, which means that a mistake can occur due to the developer's coding error as the developer misunderstood the requirement or the requirement was not defined correctly. The **developers** use the term "**Error"**.

Defect:-

- When the application is not working as per the requirement is knows as **defects**. It is specified as the aberration from the **actual and expected result** of the application or software.
- In other words, we can say that the bug announced by the **programmer** and inside the code is called a "**Defect."**

Bug:-

- In software testing, a **bug** is the informal name of **defects**, which means that software or application is not working as per the requirement. When we have some coding error, it leads a program to its breakdown, which is known a**s** a bug**.** The test engineers use the terminology Bug**.**
- If a QA (Quality Analyst**)** detect a bug, they can reproduce the bug and record it with the help of the bug report template.

Failure:-

- Many defects lead to the software's failure, which means that a loss specifies a fatal issue in software/ application or in its module, which makes the system unresponsive or broken.
- If an end-user detects an issue in the product, then that particular issue is called a "**failure**.
- Possibilities are there one defect that might lead to one failure or several failures.
- **For example**, in a bank application if the **Amount Transfer** module is not working for end-users when the end-user tries to **transfer money**, submit button is not working. Hence, this is a **failure**.

Q.4 Difference between QA v/s QC v/s Tester

Quality Assurance (QA)	Quality Control (QC)	Testing
Process-oriented focuses on making the process of creating software better.	A product-oriented approach is a way to make sure the software meets all its requirements.	Testing the software system is about finding mistake or issues.
It works with the development process to help stop mistakes and ensure the software is of good quality.	It's done after the development process and involves running test cases and seeing how the software reacts.	This usually happens after software has been created and it`s ensuring about that`s the software`s quality is up to standard.
Process oriented activities.	Product oriented activities.	Product oriented activities.
Preventive activities.	It is a corrective process.	.It is a preventive process.
It is a subset of Software Test Life Cycle (STLC).	QC can be considered as the subset of Quality Assurance.	Testing is the subset of Quality Control.

Q.5 What is Boundary value testing ?

- Boundary value analysis is one of the widely used case design technique for black box testing. It is used to test boundary values because the input values near the boundary have higher chances of error.
- Whenever we do the testing by boundary value analysis, the tester focuses on, while entering boundary value whether the software is producing correct output or not.

Understand via given example.

- **Example:** Consider a system that accepts ages from 18 to 56.

Boundary Value Analysis(marks accepts 01 to 100)		
Invalid (min-1)	Valid (min, min + 1, nominal, max – 1, max)	Invalid (max + 1)
-1,0	1,2,7,10,18, 19, 37, 55, 75,88,99	101

- Valid test cases for the above can be any value entered greater than 0 OR-1 and less than 101.
- It checks for the input values near the boundary that have a higher chance of error. Every partition has its maximum and minimum values and these maximum and minimum values are the boundary values of a partition.

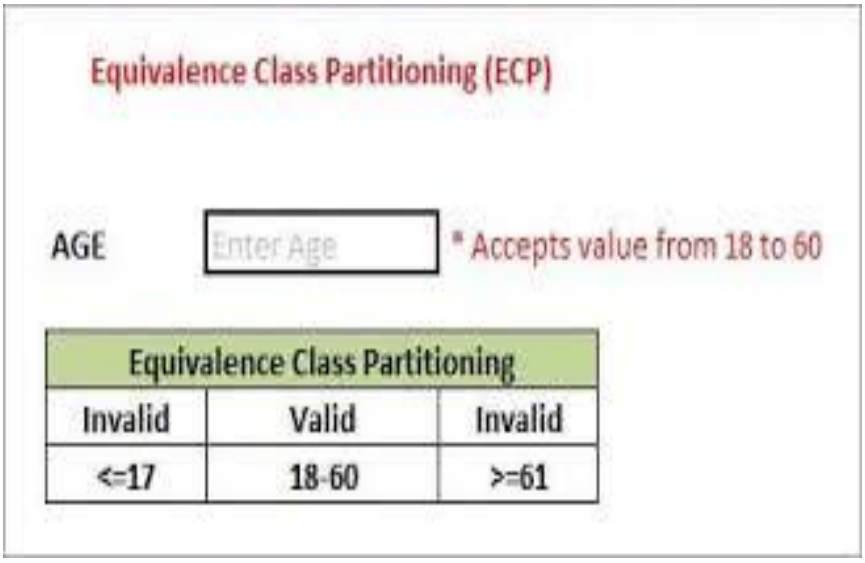
Q.6 What is Equivalence partitioning testing?

- An equivalence class describes a valise range.
- Equivalence partitioning is a technique of software testing in which input data is divided into partitions of valid and invalid values, and it is mandatory that all partitions must exhibit the same behavior.
- If a condition of one partition is true, then the condition of another equal partition must also be true, and if a condition of one partition is false, then the condition of another equal partition must also be false.
- This technique is particularly useful when dealing with a large range of input values.
- Assume that there is a function of a software application that accepts a particular number of digits, not greater and less than that particular number.
- **Example** – Testing User Registration Form

  In this example, let's consider a user registration form that requires input for the age field. The Equivalence Partitioning technique can be applied to divide the possible input values into different equivalence classes.

  Equivalence Classes:

  - **Valid Age:-** 18-60 (inclusive)
  - **Invalid Age**:- Below 18 and above 60
  - **Empty Age** :- Field



- Valid Input: 18 – 56
- Invalid Input: less than or equal to 17 (<=17), greater than or equal to 57 (>=57)
- Valid Class: 18 – 56 = Pick any one input test data from 18 – 56
- Invalid Class 1: <=17 = Pick any one input test data less than or equal to 17
- Invalid Class 2: >=57 = Pick any one input test data greater than or equal to 57

- **Advantages of Equivalence Partitioning:**

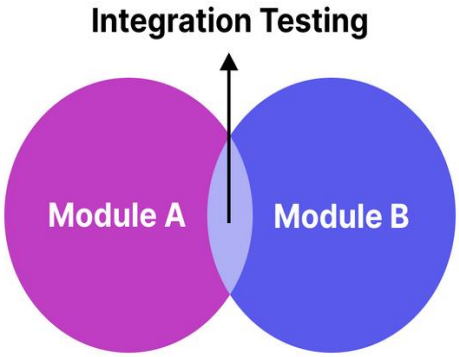
1. Enhanced Test Coverage
2. Efficiency in Test Case Design
3. Time and Effort Savings
4. Defect Detection

- **Disadvantages of Equivalence Partitioning:**

1. Limited to Input Values
2. Complex Scenarios
3. Requirement for Domain Knowledge
4. Potential for Overlooking Defects

Q.7 What is Integration testing?

- Integration testing is the second level of the software testing process comes after unit testing.
- In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.
- Once all the components or modules are working independently, then we need to check the data flow between the dependent modules is known as integration testing.

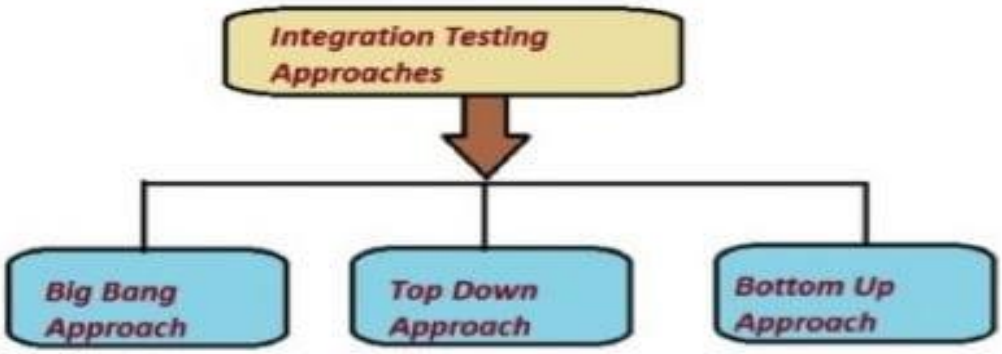


There are 2 levels of Integration Testing

1. Component Integration Testing
2. System Integration Testing

Methods of Integration Testing

- Big Bang Approach
- Incremental Integration Approach: which is further divided into the following
    a. Top Down Approach
    b. Bottom Up Approach



Q.8 What is component (Unit) testing?

- Component testing, also known as unit testing or module testing, is a level of software testing that focuses on verifying the individual components or units of a system.
- It is used to test all the components separately as well as the usability testing; interactive valuation is also done for each specific component.
- It is also known as **Module Testing or Program Testing and Unit Testing.**
- It helps in saving time by finding the bugs at a very early stage in the cycle.
- A unit component is an individual function or code of the application. White box testing approach used for unit testing and usually done by the developers.
- This testing helps tester and developers to understand the base of code that makes them able to change defect causing code quickly.
- It helps in the documentation.
- This testing fixes defects very early in the development phase that's why there is a possibility to occur a smaller number of defects in upcoming testing levels.
- It helps with code reusability by migrating code and test cases.

Below we look at some of what extreme programming brings to the world of unit testing:

- Tests are written before the code
- Rely heavily on testing frameworks
- All classes in the applications are tested
- Quick and easy integration is made possible

Advantages of component (unit) testing

- Unit testing uses module approach due to that any part can be tested without waiting for completion of another parts testing.
- The developing team focuses on the provided functionality of the unit and how functionality should look in unit test suits to understand the unit API.
- Unit testing allows the developer to refactor code after a number of days and ensure the module still working without any defect.

Disadvantages of component (unit) testing

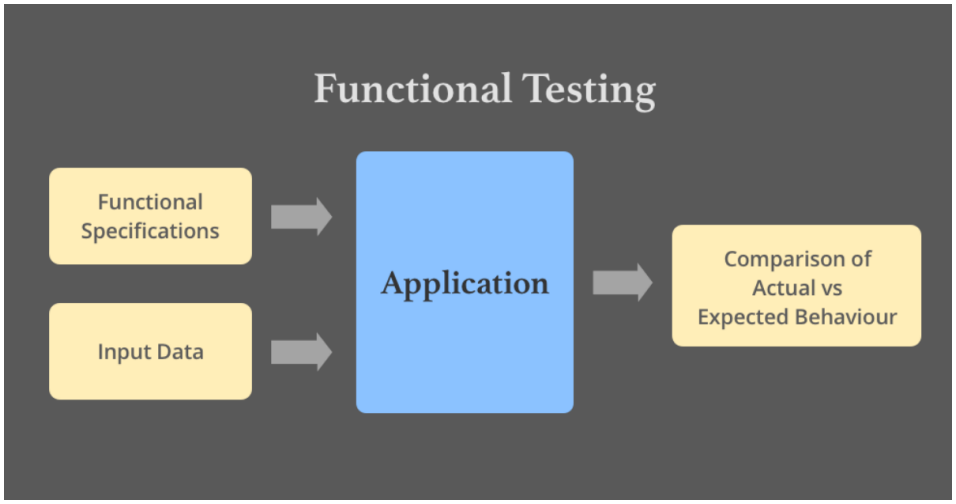
- It cannot identify integration or broad level error as it works on units of the code.
- In the unit testing, evaluation of all execution paths is not possible, so unit testing is not able to catch each and every error in a program.
- It is best suitable for conjunction with other testing activities.

Q.9 What is functional system testing?

- In functional testing, each function tested by giving the value, determining the output, and verifying the actual output with the expected value.
- Functional testing performed as black-box testing which is presented to confirm that the functionality of an application or system behaves as we are expecting. It is done to verify the functionality of the application.
- Functional testing also called as black-box testing, because it focuses on application specification rather than actual code. Tester has to test only the program rather than the system.

There are the following steps to perform functional testing:

- There is a need to understand the software requirement.
- Identify test input data
- Compute the expected outcome with the selected input values.
- Execute test cases
- Comparison between the actual and the computed result



Functional System Testing :

- A requirement that specifies a function that a system or system component must perform.
- A Requirement may exist as a text document and/or a model .

There is two types of techniques

- Requirement Based Functional Testing
- Business Process Based Testing

Requirement Based Testing

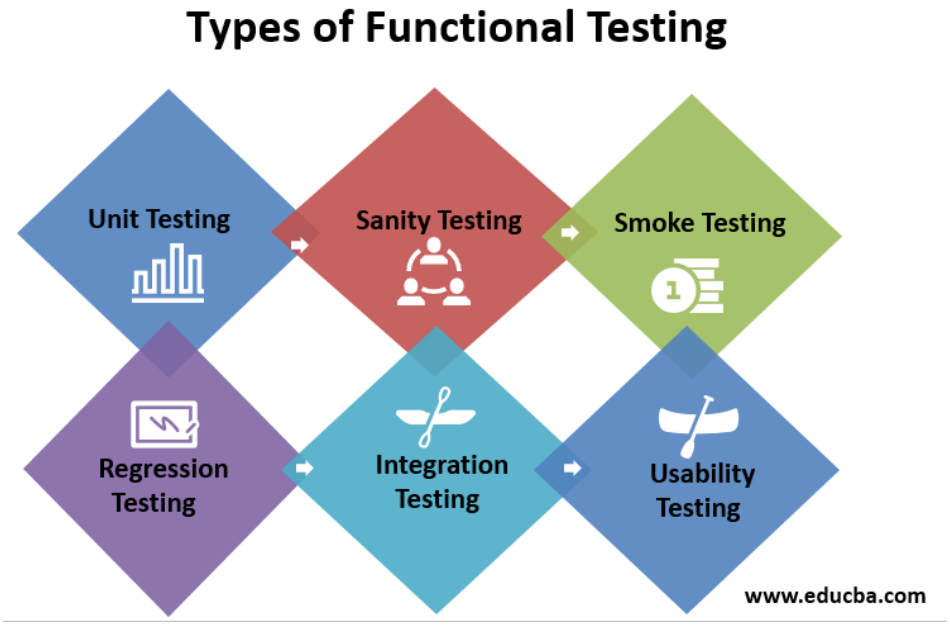
- Testing against requirements and specifications
- Test procedures and cases derived from:
    - detailed user requirements
    - system requirements functional specification
    - User documentation/instructions
    - high level System design
- Starts by using the most appropriate black-box testing techniques
- May support this with white-box techniques (e.g. menu structures, web page navigation)
- Risk based approach

Business Process Based Testing

- Test procedures and cases derived from:
    - Expected user profiles
    - Business scenarios
    - Use cases
- Testing should reflect the business environment and processes in which the system will operate.
- Therefore, test cases should be based on real business process.

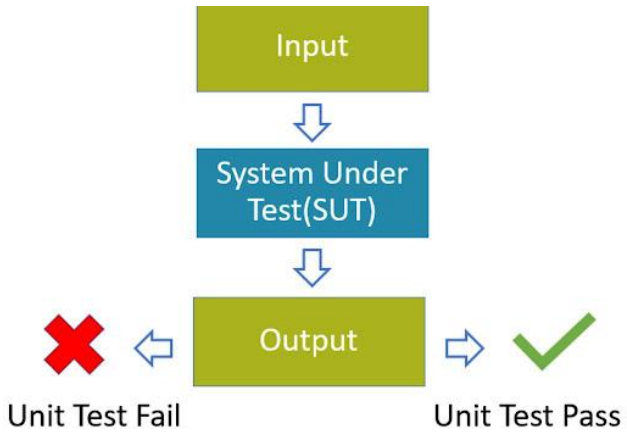
Types of Functional Testing

1. Unit Testing
2. Sanity Testing
3. Smoke Testing
4. Regression Testing
5. Integration Testing
6. Usability Testing



1. **Unit Testing :-**

- **Unit testing** is a type of software testing, where the individual unit or component of the software tested. Unit testing, examine the different part of the application, by unit testing functional testing also done, because unit testing ensures each module is working correctly.
- The developer does unit testing.
- Unit testing is done in the development phase of the application.



2. Sanity Testing :-

- **Sanity testing** involves the entire high-level business scenario is working correctly. Sanity testing is done to check the functionality/bugs fixed. Sanity testing is little advance than smoke testing.
- For example, login is working fine; all the buttons are working correctly; after clicking on the button navigation of the page is done or not.

3. Smoke Testing :-

- Smoke testing includes only the basic functionality of the system. Smoke testing is known as "**Build Verification Testing**." Smoke testing aims to ensure that the most important function work.
- For example, Smoke testing verifies that the application launches successfully will check that GUI is responsive.

4. Regression Testing :-

- This type of testing concentrate to make sure that the code changes should not side effect the existing functionality of the system.
- Regression testing specifies when bug arises in the system after fixing the bug, regression testing concentrate on that all parts are working or not. Regression testing focuses on is there any impact on the system.

5. Integration Testing :-

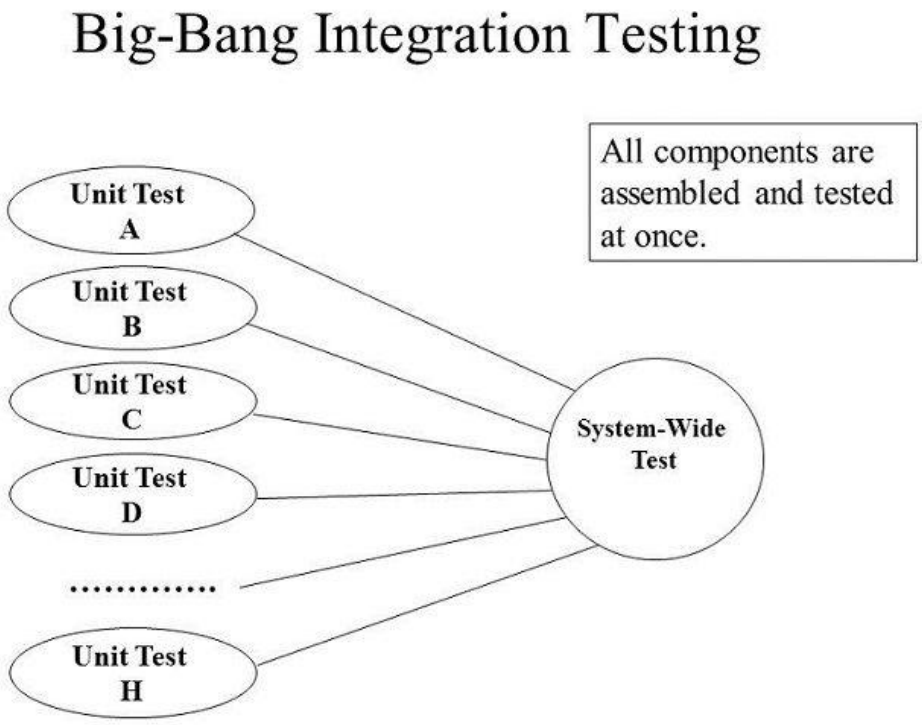
- **Integration testing** combined individual units and tested as a group. The purpose of this testing is to expose the faults in the interaction between the integrated units.
- Developers and testers perform integration testing.

6. User Acceptance (Usability) Testing :-

- It is a type of testing performed by the client to certify the system according to requirement. The final phase of testing is user acceptance testing before releasing the software to the market or production environment.

Q.10 Mention what big bang testing is?

- Big bang is a type of integration testing that combination of all the modules or components of a system into single unit and tests them as a whole.
- It is usually performed after unit testing and before validation testing. In integration testing, individual software components are combined and tested as a group.
- The purpose of this testing is to check if the components work together as expected.



- In Big Bang integration testing all components or modules is integrated simultaneously, after which everything is tested as a whole.
- Big Bang testing has the advantage that everything is finished before integration testing starts.
- The major disadvantage is that in general it is time consuming and difficult to trace the cause of failures because of this late integration.
- Here all component are integrated together at once, and then tested.

Advantages of big bang testing :-

- **Easy to implement:**
  - It is easy to implement as all the modules are already present and just need to be integrated.
- **Bugs can be identified at once:**
  - All the errors and bugs can be identified at once as all the modules are tested together.
- **Suitable for small projects:**
  - This approach is suitable for small projects where all modules can be integrated at once

Disadvantages of big bang testing :-

- **Difficult to identify the root cause of errors**:
  - It can be difficult to identify the cause of errors when all modules are tested together. It can be difficult to identify the root cause of errors if they are discovered during the final test.
- **Time-consuming:**
  - Big bang integration testing can be time-consuming because all the modules are integrated at once and tested together. This can lead to a lot of time being spent on debugging and fixing errors.
- **High risk:**
  - It is a high-risk approach because all the modules are integrated and tested together. This can lead to a lot of errors and failures.

Q.11 What is the purpose of exit criteria?

- Software testing teams will use exit criteria to determine if a test plan or project can exit to the next stage or be considered complete**.**
- 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.

How do we know when to stop testing? •

- Run out of time?
- Run out of budget?
- The business tells you it went live last night!
- Boss says stop?
- All defects have been fixed?
- When out exit criteria have been met?

- **Purpose of exit criteria is to define when we STOP testing either at the:**

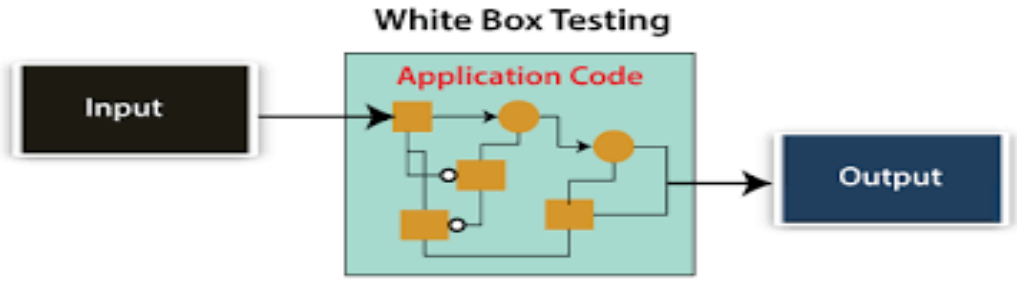
  - End of all testing – i.e. product Go Live
  - End of phase of testing (e.g. hand over from System Test to UAT)

- **Exit Criteria typically measures:**

  - Thoroughness measures, such as coverage of requirements or of code or risk coverage
  - Estimates of defect density or reliability measures. (e.g. how many defects open by category)
  - Residual Risks, such as defects not fixed or lack of test coverage in certain areas

Q.12 What is white box testing and list the types of white box testing?

- White Box Testing is testing technique in which software's internal structure, design,and coding are tested to verify input-output flow and improve design,usability,and security.
- White box testing which also know as glass box is testing, structural testing, clear box testing, open box testing and transparent box testing.
- It is based on inner workings of an application and revolves around internal structure testing. In this type of testing programming skills are required to design test case.
- Developers do white box testing. In this, the developer will test every line of the code of the program. The developers perform the White-box testing and then send the application or the software to the testing team.
- The developer fixes the bugs and does one round of white box testing.



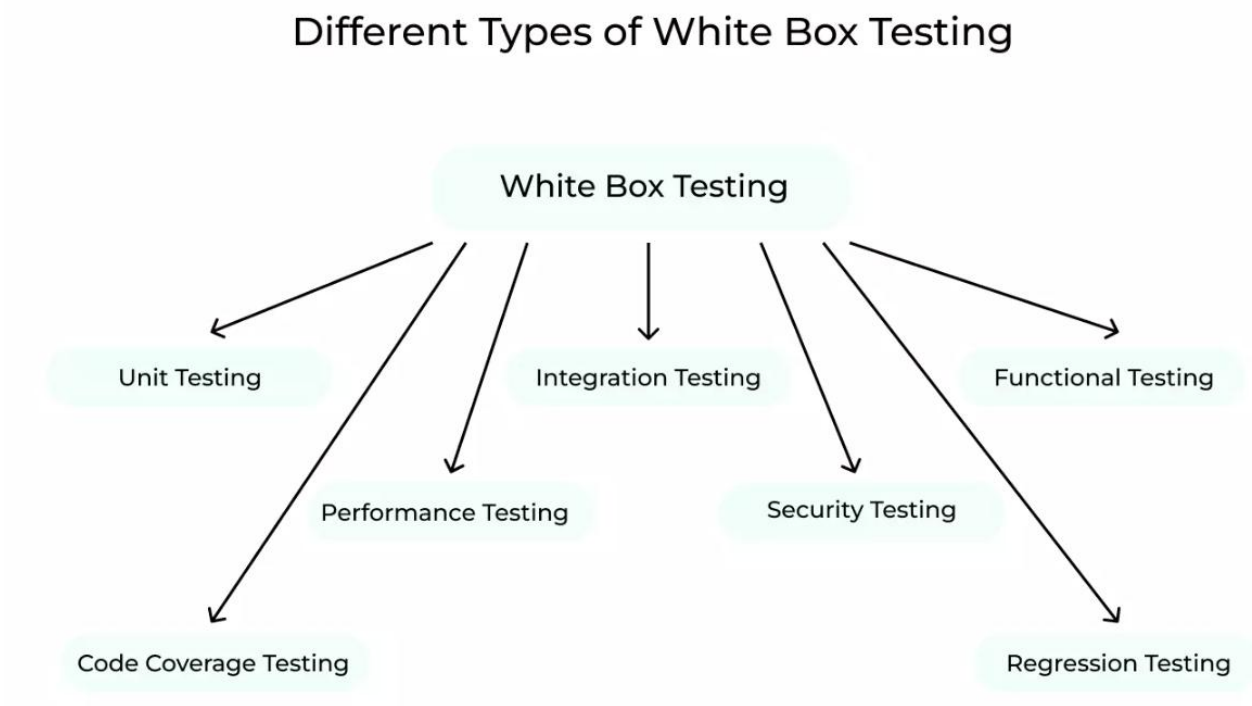
Advantages of White box testing

- White box testing optimizes code so hidden errors can be identified.
- Test cases of white box testing can be easily automated.
- This testing is more thorough than other testing approaches as it covers all code paths.
- It can be started in the SDLC phase even without GUI.

Disadvantages of White box testing

- White box testing is too much time consuming when it comes to large-scale programming applications.
- White box testing is much expensive and complex.
- It can lead to production error because it is not detailed by the developers.

- White box testing needs professional programmers who have a detailed knowledge and understanding of programming language and implementation.



Different types of white box testing

1. Unit testing
2. Integration testing
3. Functional testing
4. Performance testing
5. Security testing
6. Code coverage testing
7. Regression testing

Q.13 What is black box testing? What are the different black box testing techniques?

- Black box testing is a technique of software testing which examines the functionality of software without peering into its internal structure or coding. The primary source of black box testing is a specification of requirements that is stated by the customer.
- The test team reports the result to the development team and then tests the next function. After completing testing of all functions if there are severe problems, then it is given back to the development team for correction.
- The test procedure of black box testing is a kind of process in which the tester has specific knowledge about the software's work, and it develops test cases to check the accuracy of the software's functionality.
- Specification-based testing technique is also known as 'black-box' or input/output driven testing techniques because they view the software as a black-box with inputs and outputs.
- The testers have no knowledge of how the system or component is structured inside the box. In black-box testing the tester is concentrating on what the software does, not how it does it.
- **For example**, when performing system or acceptance testing, the requirements specification or functional specification may form the basis of the tests.
- The technique of testing without having any knowledge of the interior workings of the application is Black Box testing.

Advantages of Black Box Testing

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

Disadvantages of Black Box Testing

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

Techniques of Black Box Testing

- **There are four specification-based or black-box technique**:
    - Equivalence partitioning
    - Boundary value analysis
    - Decision tables
    - State transition testing
    - Use-case Testing
    - Other Black Box Testing
        - Syntax or Pattern Testing

Q.14 What is Exploratory Testing?

- If requirement does not exist, then we do one round of exploratory testing.
- So, for this first, we will be exploring the application in all possible ways, understanding the flow of the application, preparing a test document and then testing the application, this approach is known as exploratory testing.

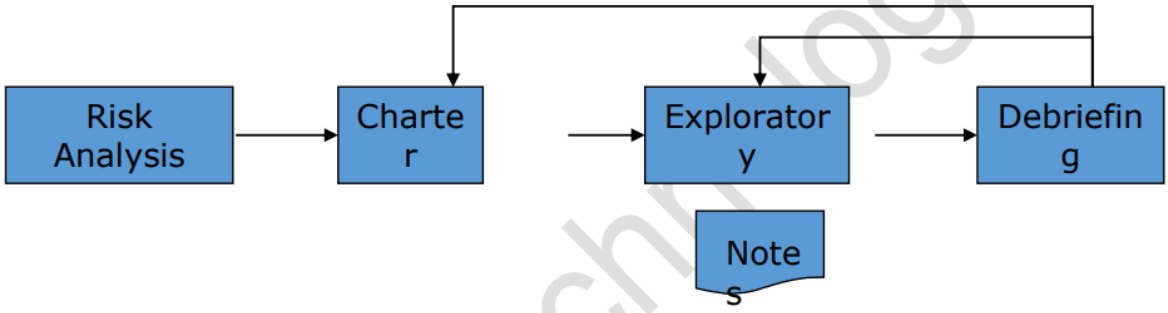
  We will use this testing for the following aspects:

  - When the requirement is missing
  - Early iteration is required
  - The testing team has the experienced testers when we have a critical application, and new testers entered into the team.

For example, to test any software or the application, first, we will perform unit, integration, and system testing.

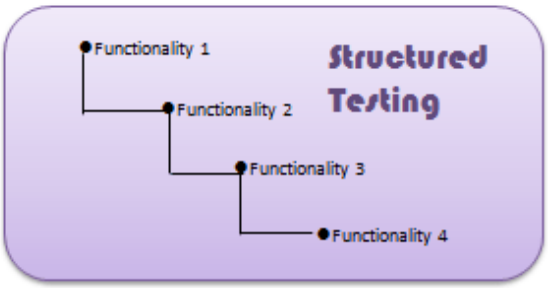
  - So if we want to understand any application first, we will perform unit or component testing, suppose the application is having a login page having many elements, and we will understand each part and doing the component testing, but actually, we are doing the exploratory testing because we are exploring the application.

- Exploratory testing is a concurrent process where
  - Test design, execution and logging happen simultaneously
  - Testing is often not recorded
  - Makes use of experience, heuristics and test patterns
  - Testing is based on a test charter that may include
    - Scope of the testing (in and out)
    - The focus of exploratory testing is more on testing as a “thinking” activity.
    - A brief description of how tests will be performed
    - Expected problems
    - Is carried out in time boxed intervals
    - More structured than Error guessing

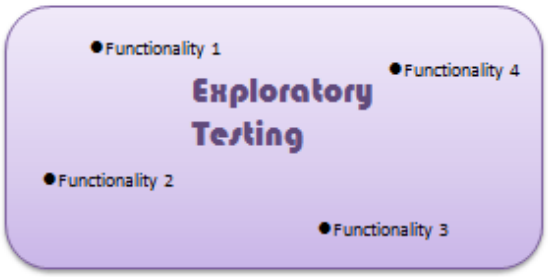


Though the current trend in testing is to push for automation, exploratory testing is a new way of thinking. Automation has its limits

- Is not random testing but it is Adhoc testing with purpose of find bugs Is structured and rigorous
- Is cognitively (thinking) structured as compared to procedural structure of scripted testing. This structure comes from Charter, time boxing etc.
- Is highly teachable and manageable
- Is not a technique but it is an approach. What actions you perform next is governed by what you are doing currently



Functionalities are checked in a structured manner



Functionalities are checked in a ad-hoc manner

Q.15 What is traceability matrix?

- Traceability matrix is a table type document that is used in the development of software application to trace requirements. It can be used for both forward (from Requirements to Design or Coding) and backward (from Coding to Requirements) tracing. It is also known as **Requirement Traceability Matrix (RTM) or Cross Reference Matrix (CRM).**
- It is prepared before the test execution process to make sure that every requirement is covered in the form of a Test case so that we don't miss out any testing. In the RTM document, we map all the requirements and corresponding test cases to ensure that we have written all the test cases for each condition.

TRACEABILITY MATRIX

Requirement Number	Test Case Name
1	. . .
2	
3	. . .
4	
5	. . .
6	. . .
7	. . .
8	. . .

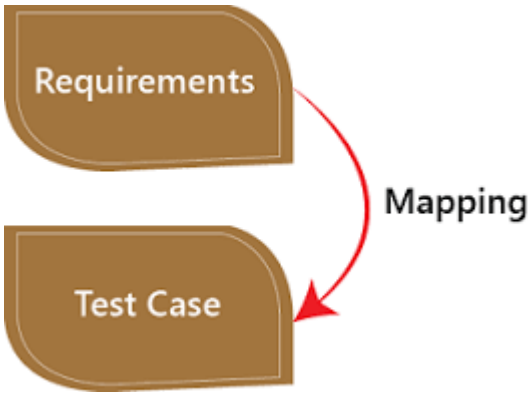
Types of Traceability Test Matrix

The traceability matrix can be classified into three different types which are as follows:

- Forward traceability
- Backward or reverse traceability
- Bi-directional traceability

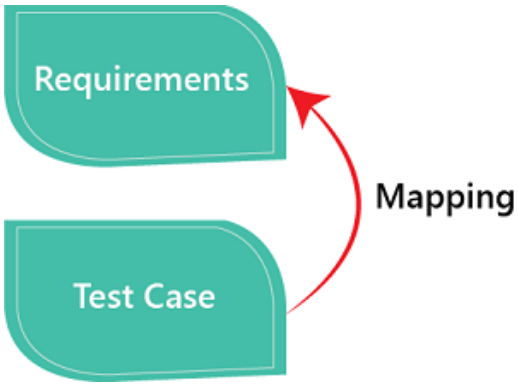
Forward traceability

- The forward traceability test matrix is used to ensure that every business's needs or requirements are executed correctly in the application and also tested rigorously. The main objective of this is to verify whether the product developments are going in the right direction. In this, the requirements are mapped into the forward direction to the test cases.



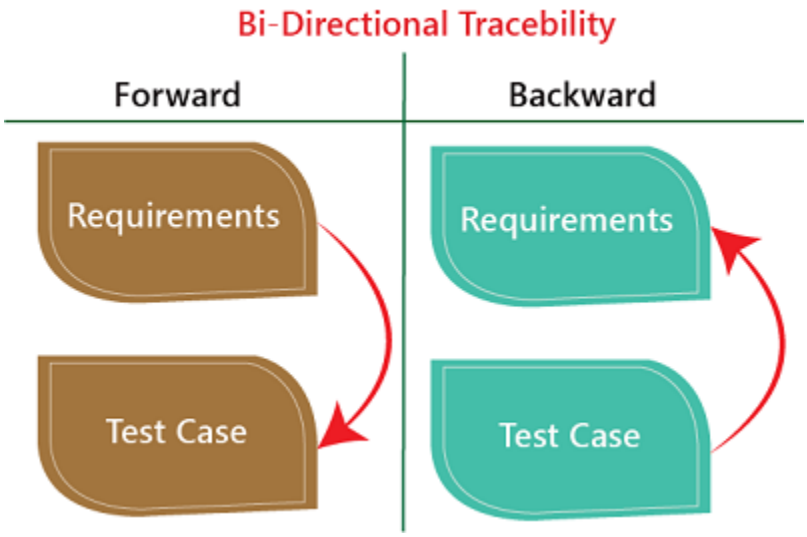
Backward or reverse traceability

- The reverse or backward traceability is used to check that we are not increasing the space of the product by enhancing the design elements, code, test other things which are not mentioned in the business needs. And the main objective of this that the existing project remains in the correct direction. In this, the requirements are mapped into the backward direction to the test cases.



Bi-directional traceability

- It is a combination of forwarding and backward traceability matrix, which is used to make sure that all the business needs are executed in the test cases. It also evaluates the modification in the requirement which is occurring due to the bugs in the application.



Advantages of traceability matrix

- Make obvious to the client that the software is bring developed as per the requirements.
- To make sure that all requirements included in the test case.
- Easy to identify the missing functionalities.
- To make sure that developers are no one has requested.

Disadvantages of traceability matrix

- No tracebility or income traceability results into :-
- Poor or unknown test coverage, more defects found in production.
- It will lead to miss some bugs in earlier test cycles which may arise in later test cycle. Then a lot of discussions arguments with other of teams and managers before release.
- Difficult project planning and tracking misunderstanding between different teams over project dependencies delays,etc…..

Q.16 Explain what Test Plan is? What is the information that should be covered.

- A test plan is a detailed document which describes software testing areas and activities. It outlines the test strategy, objectives, test schedule, required resources (human resources, software, and hardware), test estimation and test deliverables.
- The test plan is a base of every software's testing. It is the most crucial activity which ensures availability of all the lists of planned activities in an appropriate sequence.
- The test plan is a template for conducting software testing activities as a defined process that is fully monitored and controlled by the testing manager. The test plan is prepared by the Test Lead (60%), Test Manager (20%), and by the test engineer(20%).
- A document describning the scope approach, resources and schedule of intended test activities.
- Scheduling test analysis and design activities.
- Scheduling test implementation execution and evaluation.
- The organization test policy.
- Test objective.
- Project risk- e.g., business, and technical people.
- Constraints - e.g., business impulse,financial,contractual etc.

Test plans are continuously refined.

- As more information becomes available.
- As new risks or other are mitigated.
- Not set in concrete, but changes must be carefully.

Test planning activities

Approach:-

- Defining approach of testing (the test strategy),occluding the definition of the test levels and entry and exit criteria.

Making decisions about

- What to test.
- Who do testing(e.g., what roles will perform the test activities)
- When and how test activities should be done and when they should be stopped.
- How the test result should be evaluated

Q.17 Difference between Smoke and Sanity?

S.No.	Comparison Basis	Smoke Testing	Sanity Testing
1	Test coverage	It is a broad approach to testing where all parts of the application are tested.	It is a narrow approach to testing where specific parts of the application are tested.
2	Measures	It measures the stability of the system by performing rigorous testing.	It measures the rationality of the system by performing rigorous testing.
3	Technique	Smoke testing can be either manual or automated.	Sanity testing can be done without test cases or scripts.
4	Executed by	It is performed by both testers and developers.	It is performed by only testers.
5	Purpose	Testing is done without getting into deep but whenever needed tester has to go into deep.	Sanity testing does not need to go into deep of the application.
6.	Performed at	Smoke testing is the first testing performed on the initial build.	Sanity testing is performed when the build is comparatively stable.
7	Documentation	Smoke testing is documented.	Sanity testing is not documented.
8	Used to	It is used to test End to End function of the application.	It is used to test only modified or defect fixed functions.
9	Subset	It is considered as a subset of acceptance testing.	It is considered as a subset of regression testing

Q.18 Explain the difference between Functional testing and Nonfunctional testing?

Functional testing	Non-functional testing
It focuses on testing the functionality of the software or system.	It focuses on testing the non-functional aspects of the software or system.
Verifies whether the software meets the functional requirements.	Verifies whether the software meets the non-functional requirements such as performance, security, usability, reliability, and compatibility.
It involves testing the features and functionalities of the software, such as input/output, error handling, and user interface.	It involves testing the quality attributes of the software, such as response time, scalability, availability, and maintainability.
Tests are typically conducted using test cases or scenarios that validate the functional requirements.	Tests are conducted using various techniques such as load testing, stress testing, security testing, and usability testing.
It can be performed manually or using automated testing tools.	Often requires specialized testing tools and frameworks to measure and evaluate the non-functional requirements.
Done after unit testing and integration testing and before system testing.	It can be done at various stages of the development lifecycle, from design to deployment and maintenance.

Q.19 What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)?

Parameter	SDLC	STLC
Origin	Development Life Cycle	Testing Life Cycle
Objective	The main object of SDLC life cycle is to complete successful development of the software including testing and other phases.	The only objective of the STLC phase is testing.
Requirement Gathering	In SDLC the business analyst gathers the requirements and create Development Plan	In STLC, the QA team analyze requirement documents like functional and non-functional documents and create System Test Plan
High & Low-Level Design	In SDLC, the development team creates the high and low-level design plans	In STLC, the test analyst creates the Integration Test Plan
Coding	The real code is developed, and actual work takes place as per the design documents.	The testing team prepares the test environment and executes them
Maintenance	SDLC phase also includes post-deployment supports and updates.	Testers, execute regression suits, usually automation scripts to check maintenance code deployed.

Q.20 What is the difference between test scenarios, test cases, and test script?

	Test scenario	Test Cases	Test Script
1.	The test scenario is just a document that is detailed and provides details aboutthe assessment method, testing process, precondition, and anticipated output.	Test cases is a step by stepprocedure to test any functionality of the softwareapplication/prod uct.	**Test script is set of instruction or a short program to test any functionality of software application/prod uct.**
2.	The test scenarios are the ones based on the use situation and give one-lineinformation one what to check.	Test cases is a manual approachof software testing.	**Test script is an automaticapproach of software testing.**
3.	Test scenarios are one-linerstatement, however, it is linked to a few testinstances.	It is a set up that is used by thetester to test any specific function of the softwareproduct.	**It is a program developed by the tester, intended to test any specific function of the software product.**
4.	These are highlevelactions.	Point by point test case configuration encourages tester to test viably.	**Automatic testing approach is beneficial for constant execution.**
5.	Writing the test scenario'sprimary objective is an address end to get rid of functionality of a software program.	Test cases are written bymanually.	**Test scripting is done byscripting format.**
6.	It will take less time as compared to test cases.	Test case is developed in form of templates.	**Test script is developed in form of scripting.**
7.	**The test scenario will helpus in a way that is nimble of through the functionality.**	**If the tester does not have a good understanding of how theprogram is used or about the recent risks to the program, then it will be difficult to usethe test cases properly.**	**Active software projectsfrequently change.so testers have to make a continuous effort to update the scripts to match the changes of the new product.**

Q.20 What is Non-Functional Testing?

- Non function testing is a type of software testing that is performed to verify the non-function requirements of the application. It verifies whether the behavior of the system is as per the requirement or not.
- Non-Functional testing checks the Performance, reliability, scalability and other non-functional aspects of the software system.
- Non functional testing should be performed after functional testing.
- Nonfunctional testing describes how good the product works.

Types of Nonfunctional testing are

- Performance Testing
- Load Testing
- Volume Testing
- Stress Testing
- Security Testing
- GUI Testing
- Compatibility Testing

Q.21 What is Adhoc testing?

- This testing we do when the build is in the checked sequence, then we go for Adhoc testing by checking the application randomly.
- Adhoc testing is also known as **Monkey testing and Gorilla testing**.
- It is negative testing because we will test the application against the client's requirements.
- Adhoc testing is an informal testing type with an aim to break the system.
- It does not follow any test design techniques to create test cases.
- In fact is does not create test cases altogether!
- This testing is primarily performed if the knowledge of testers in the system under test is very high.
- Testers randomly test the application without any test cases or any business requirement document.
- Adhoc Testing does not follow any structured way of testing and it is randomly done on any part of application.
- Main aim of this testing is to find defects by random checking.
- Adhoc testing can be achieved with the testing technique called Error Guessing.
- Error guessing can be done by the people having enough experience on the system to "guess" the most likely source of errors.
- **The Error guessing is a technique where the experienced and good testers are encouraged to think of situations in which the software may not be able to cope.**
- Some people seem to be naturally good at testing and others are good testers because they have a lot of experience either as a tester or working with a particular system and so are able to find out its weaknesses.

Types of Adhoc Testing There are different types of Adhoc testing

Buddy Testing :-

- Two buddies mutually work on identifying defects in the same module. Mostly one buddy will be from development team and another person will be from testing team. Buddy testing helps the testers develop better test cases and development team can also make design changes early. This testing usually happens after unit testing completion.

Pair testing :-

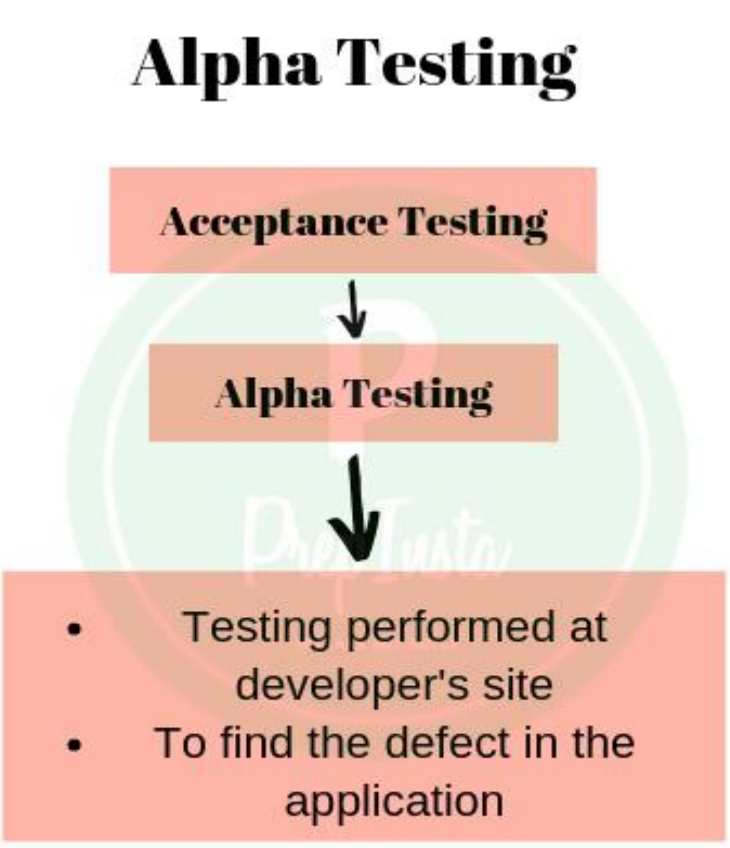
- Two testers are assigned modules, share ideas and work on the same machines to find defects. One person can execute the tests and another person can take notes on the findings. Roles of the persons can be a tester and scriber during testing.

Monkey Testing :-

- Randomly test the product or application without test cases with a goal to break the system.

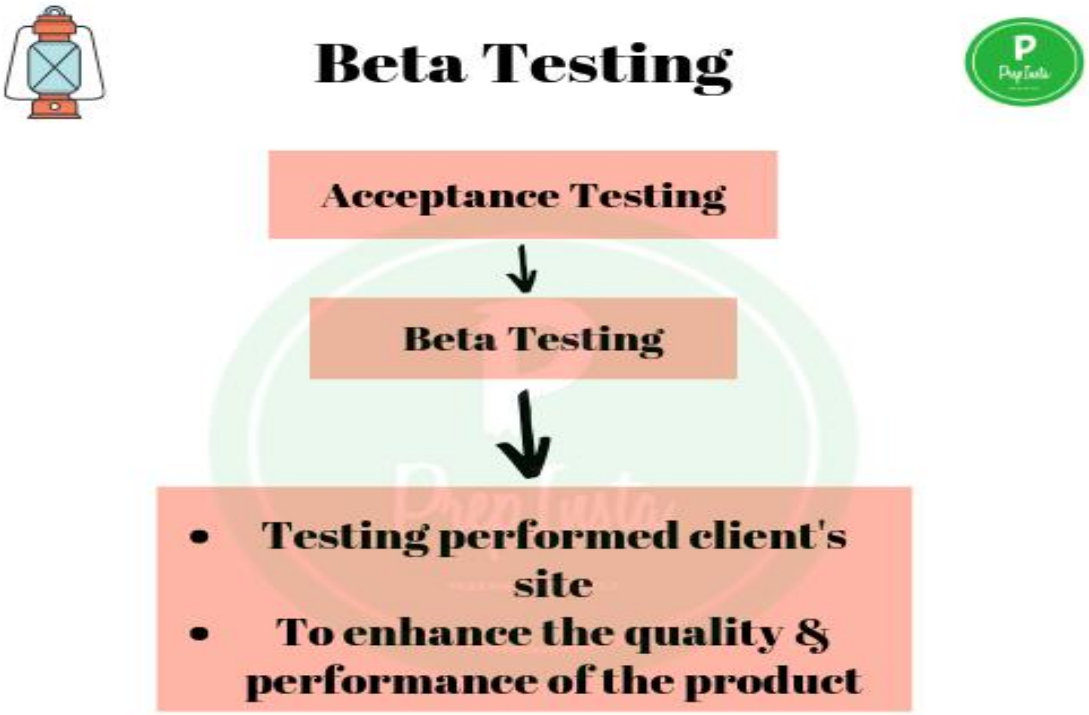
Q.22 what is Alpha testing?

- Alpha testing is always performed by the developers at the software development site.
- Sometimes it is also performed by Independent Testing Team.
- Alpha Testing is not open to the market and public
- It is conducted for the software application and project.
- It is always performed in Virtual Environment.
- It is always performed within the organization.
- It is the form of Acceptance Testing.
- It comes under the category of both White Box Testing and Black Box Testing.
- Alpha testing is happening at the stage of the completion of the software product.
- Alpha testing is doing after the unit testing, integration testing, system testing but before the beta testing.
- Alpha testing is for testing the software application, products, and projects.



Q.23 what is Beta testing (field testing)?

- Beta testing always performed by the customers at their own site.
- It is not performed by Independent Testing Team.
- Beta Testing is always open to the market and public.
- It is usually conducted for software product.
- It is performed in Real Time Environment.
- It is always performed outside the organization.
- It is also the form of Acceptance Testing.
- Beta Testing (field testing) is performed and carried out by users or you can say people at their own locations and site using customer data.
- It is only a kind of Black Box Testing.
- Beta Testing is always performed at the time when software product and project are marketed.
- Beta testing can be considered “pre-release” testing.
- Pilot Testing is testing to product on real world as well as collect data on the use of product in the classroom.



Q.24 What is GUI testing ?

- Graphical User Interface (GUI) testing is the process of testing the system's GUI of the System under Test. GUI testing involves checking the screens with the controls like menus, buttons, icons, and all types of bars – tool bar, menu bar, dialog boxes and windows etc.

WHAT DO YOU CHECK IN GUI TESTING?

  - Check all the GUI elements for size, position, width, length and acceptance of characters or numbers. For instance, you must be able to provide inputs to the input fields.
  - Check you can execute the intended functionality of the application using the GUI
  - Check Error Messages are displayed correctly
  - Check for Clear demarcation of different sections on screen
  - Check Font used in application is readable
  - Check the alignment of the text is proper
  - Check the Color of the font and warning messages is aesthetically pleasing
  - Check that the images have good clarity
  - Check that the images are properly aligned
  - Check the positioning of GUI elements for different screen resolution

- **Manual based testing**
  - Under this approach,graphical screens are checked manually by testers in conformance wirh the requirments stated in businessrequirements document.

- **Record and replay**
  - GUI testing can be done using automation tools. This is done in 2 parts.during record,test steps are captured into the automation tool. EXAMPLE:-QTP

- **Model based testing**
  - A model is a graphical description of system behavior.it helps us to understand and predict the system behavior.model help in ageneration of efficient test cases using thesystem requirements

Q.25 What is load testing ?

- Load testing - Its a performance testing to check system behavior under load. Testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system's response time degrades or fails.
- Load testing is a kind of performance testing which determines a system's performance under real-life load conditions. This testing helps determine how the application behaves when multiple users access it simultaneously.

This testing usually identifies –

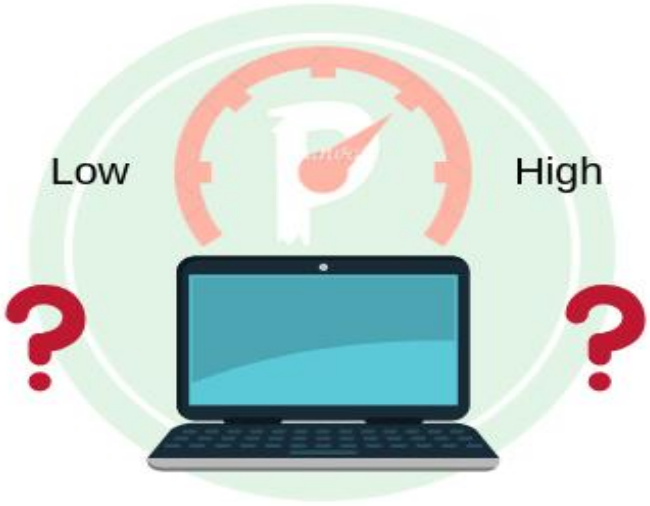
- The maximum operating capacity of an application
- Determine whether current infrastructure is sufficient to run the application
- Sustainability of application with respect to peak user load
- Number of concurrent users that an application can support, and scalability to allow more users to access it.
- It is a type of non-functional testing. Load testing is commonly used for the Client/Server, Web based applications – both Intranet and Internet.

- The load testing is used to perform the **maximum quantity** of software applications without important performance breakdown.
- It is used to govern the scalability of the application and allows various users to access it.
- It is used to identify the total count of users that can access the application simultaneously.

Q.26 What is stress testing ?

- Stress testing is to test the system behavior under extreme conditions and is carried out till the system failure.
- Stress testing determines the breaking point of the system to reveal the maximum point after which it breaks.
- Stress testing tries to break the system by testing with overwhelming data or resources.
- Stress testing - System is stressed beyond its specifications to check how and when it fails. Performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to system or database load.
- Stress testing is used to test the stability & reliability of the system. This test mainly determines the system on its robustness and error handling under extremely heavy load conditions.
- It even tests beyond the normal operating point and evaluates how the system works under those extreme conditions.
- Stress Testing is done to make sure that the system would not crash under crunch situations.
- Stress testing is also known as endurance testing.

Stress Testing



How System Works ?

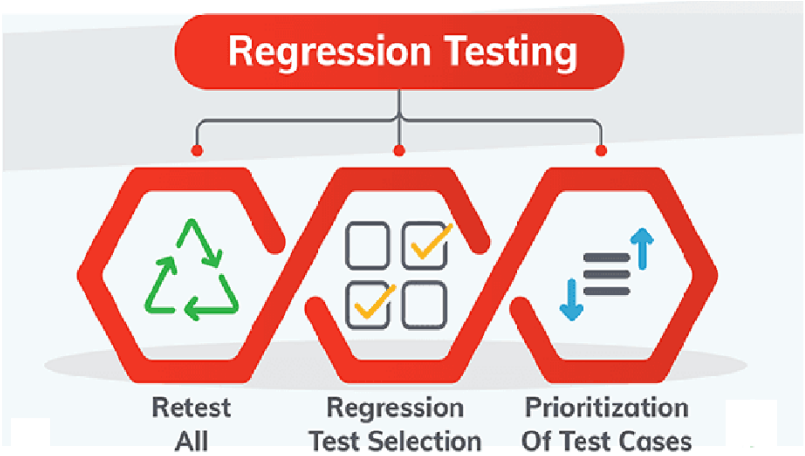
Q.27 When to used Usablity Testing?

- In usability testing, you'll be looking at aspects of your web application that affect the user's experience, such as:
    - How easy is it to navigate through your web application?
    - Is it obvious to the user which actions are available to him or her?
    - Is the look-and-feel of your web application consistent from page to page, including font sizes and colors?
    - If a user forgets to fill in a required field, you might think it is a good idea to present the user with a friendly error message and change the color of the field label to red or some other conspicuous color.
    - However, changing the color of the field label would not really help a user who has difficulty deciphering colors. The use of color may help most users, but you would want to use an additional visual clue, such as placing an asterisk beside the field in question or additionally making the text bold.
    - In Usability Testing, a small-set of target end-users, of a software system, "use" it to expose usability defects.

- Need For Usability Testing
    - Aesthetics and design are important. How well a product looks usually determines how well it works.
    - There are many software applications / websites, which miserably fail, once launched, due to following reasons –
    - Where do I click next?
    - Which page needs to be navigated?
    - Which Icon or Jargon represents what?
    - Error messages are not consistent or effectively displayed
    - Session time not sufficient.
    - Usability Testing identifies usability errors in the system early in development cycle and can save a product from failure.

Q.28 when should "Regression Testing" be performed ?

- Regression testing is a black box testing techniques. It is used to authenticate a code change in the software does not impact the existing functionality of the product. Regression testing is making sure that the product works fine with new functionality, bug fixes, or any change in the existing feature.
- Regression testing is performed before each new instance new instance of the product is deployed, guaranteeing that the program works perfectly in each setting. For instance, we need to make sure the program works perfectly in each setting. for instance, we need to make sure the product continues to properly in a production environment before we release it.
- Regression testing is a type software testing test case are re-executed to check the previous functionality of the application is working fine, and the new changes have not produced any bugs.



- **<u>Regression Testing Techniques</u>**
    - **Retest All** :- This is one of the methods for regression testing in which all the tests in the existing test bucket or suite should be re-executed. This is very expensive as it requires huge time and resources.
    - **Regression Test Selection** :- Instead of re-executing the entire test suite, it is better to select part of test suite to be run . Test cases selected can be categorized as 1) Reusable Test Cases 2) Obsolete Test Cases. . Re-usable Test cases can be used in succeeding regression cycles. . Obsolete Test Cases can't be used in succeeding cycles.
    - **Prioritization Of Test Cases** :- Prioritize the test cases depending on business impact, critical & frequently used functionalities. Selection of test cases based on priority will greatly reduce the regression test suite.

Q.29 Explain types of Performance testing?

- Software performance testing is a means of quality assurance (QA). It involves testing software applications to ensure they will perform well under their expected workload.
- Features and Functionality supported by a software system is not the only concern. A software application's performance like its response time, do matter. The goal of performance testing is not to find bugs but to eliminate performance bottlenecks
- The focus of Performance testing is checking a software programs
    - Speed – Determines whether the application responds quickly
    - Scalability – Determines maximum user load the software application can handle.
    - Stability – Determines if the application is stable under varying loads

Types of Performance Testing

- Load testing
- Stress testing
- Volume testing
- Scalability testing

Q.30 When to used Usablity Testing?

In usability testing, you'll be looking at aspects of your web application that affect the user's experience, such as:

- How easy is it to navigate through your web application?
- Is it obvious to the user which actions are available to him or her?
- Is the look-and-feel of your web application consistent from page to page, including font sizes and colours?
- If a user forgets to fill in a required field, you might think it is a good idea to present the user with a friendly error message and change the colour of the field label to red or some other conspicuous colour.
- In Usability Testing, a small set of target end-users, of a software system, "**use**" it to expose usability defects.
- This testing mainly focuses on the user's-ease to use the application, flexibility in Handling controls and ability of the system to meet its objectives.
- This testing is recommended during the initial design phase of SDLC, which gives More visibility on the expectations of the users.

Need For Usability Testing

- Aesthetics and design are important. How well a product looks usually determines how Well, it works.
  - There are many software applications / websites, which miserably fail, once launched,
- Due to following reasons –
  - Where do I click next?
  - Which page needs to be navigated?
  - Which Icon or Jargon represents what?
  - Error messages are not consistent or effectively displayed
  - Session time not sufficient.
  - Usability Testing identifies usability errors in the system early in development cycle
- And can save a product from failure.