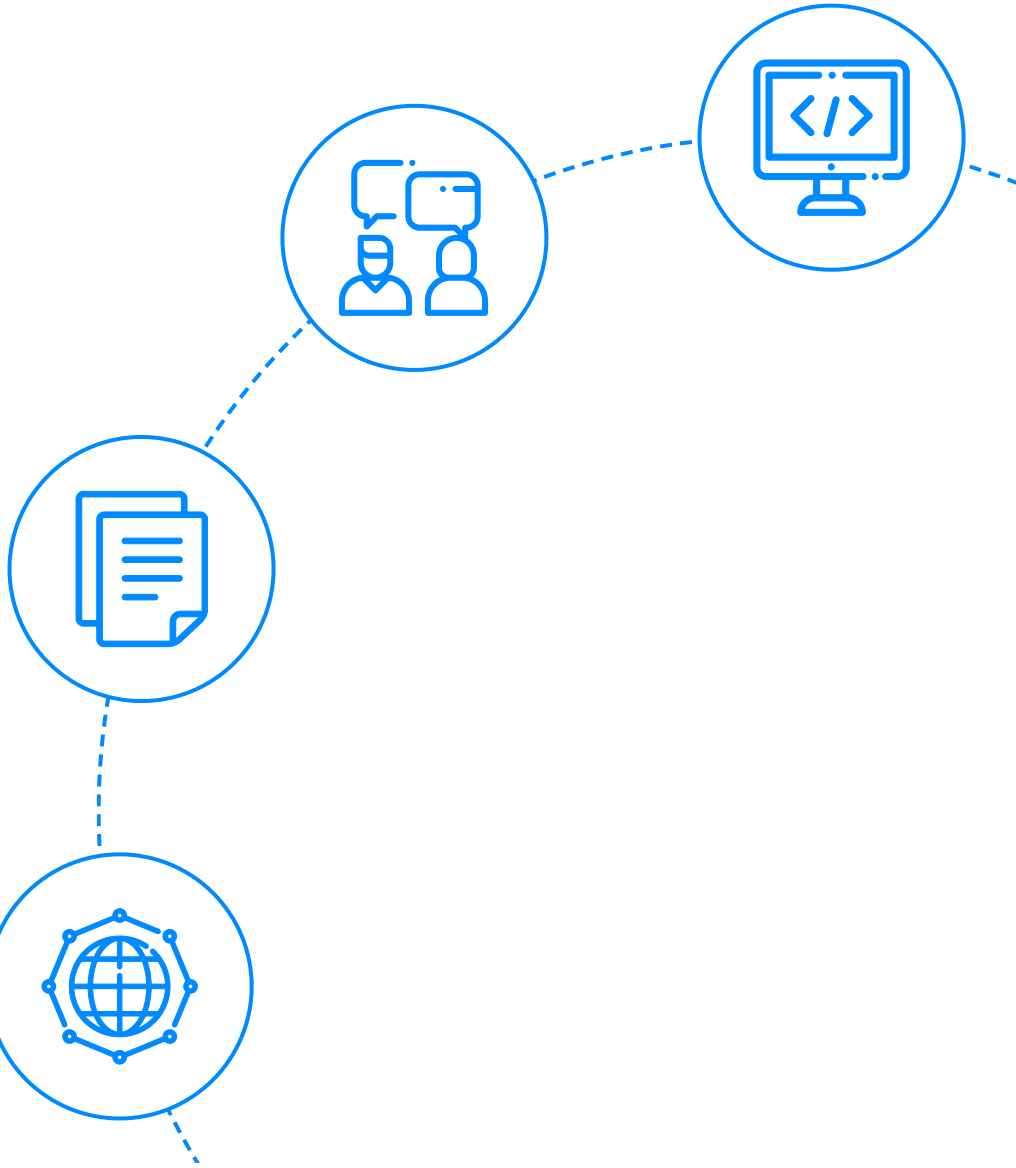
InterviewBit

Manual Testing Interview Questions



To view the live version of the page, click here.

Contents

Manual Testing Interview Questions for Freshers

Manual Testing Interview Questions for Freshers

(.....Continued)

21. Name two parameters that can be useful to check the quality of test execution.
22. What is API testing?
23. Explain use-case testing.
24. Explain Path testing.
25. Explain Endurance Testing or Soak Testing?

Manual Testing Interview Questions for Experienced

26. Explain the term testbed.
27. Explain bugs, defects, and errors.
28. What is the software testing life cycle?
29. What is black-box testing?
30. What white-box testing?
31. State the difference between bug leakage and bug release.
32. What do you mean by Critical bug?
33. What do you mean by Data flow testing?
34. What are the differences between manual and automated testing?
35. State difference between static and dynamic testing.
36. What is the term ‘quality’ mean when testing?
37. What is the difference between Quality Control(QC) and Quality Assurance(QA)?
38. State difference between alpha testing and beta testing.

Manual Testing Interview Questions for Experienced

(.....Continued)

39. Explain Monkey Testing and Performance Testing.
40. What is the role of documentation in manual testing?
41. Explain RTM (Requirement Traceability Matrix).
42. What is the importance of agile testing?
43. What is the difference between Regression and Retesting?
44. What is System testing and Unit Testing? Write the difference between them.
45. What are the types of Integration Testing?
46. Name some of the most popular integration testing tools.
47. What is Test Harness and Test Closure?
48. Explain different stages of the defect life cycle.
49. Explain Experienced-based testing techniques.
50. Write the difference between smoke testing and sanity testing.
51. What do you mean by pesticide paradox? What you can do to overcome it.

Manual Testing Scenario-Based Interview Questions

52. When to choose manual testing over automation testing and vice versa?
53. In what way will you determine when to stop testing?
54. Can 100% testing coverage be achieved? How do you ensure test coverage?
55. System testing can be done at any stage. Yes or No?
56. If proper documentation is not available for testing, what steps will you take to overcome the challenge?

Manual Testing Scenario-Based Interview Questions

(.....Continued)

57. What are some best practices that you should follow when writing test cases?
58. When the requirements are still in flux, what is the best way to test a product?
59. What makes boundary value analysis a good method for providing test cases?
60. How do you know the code has met specifications?

Let's get Started

In the area of customer service, things tend to go wrong more often than not, but once an application has been installed correctly, it should start working properly as soon as it has been installed. That's true whether or not one is using an online application or a client-server application. Bugs and glitches in a product will cause people to reject it, and the company producing it will lose credibility. Hence, the importance of quality control and testing cannot be overstated.

In the fields of software and hardware, testing is a vast field, and it is an absolute necessity before any product or software is launched. Software testing measures the quality, correctness, and completeness of developed software. **Software Testing** encompasses a wide range of disciplines, but can broadly be divided into two categories - manual testing and automated testing. Manual testing plays an integral part in the field of software development since automation testing isn't applicable for every scenario.

MANUAL Testing Interview Questions?

Recruiters are looking for evidence that you can handle the physical scrutiny of manual testing. In this article, we will introduce you to the concept of manual testing, some manual testing interview questions & answers, and some manual testing MCQs that will help you impress recruiters in your interview. A wide range of skill levels is represented in the questions, so it's suitable for both beginners and experts. The Manual Testing interview will be easier if you prepare and evaluate your answers beforehand.

Now let's take a look at some frequently asked Manual Testing interview questions.

Manual Testing Interview Questions for Freshers

1. What is Manual Testing?

In manual testing, a tester manually verifies the functionality of the software. The tester has a comprehensive list of all the manual testing test cases they should test, along with the test data. They go through each case, one by one. They launch the software as an end-user would, enter the input, and manually verify the output.

It may seem that manual testing is inefficient when compared to automated testing. It is slow, not repeatable in a consistent manner, and prone to human misjudgment.

However, manual testing allows the tester to realistically test the software, using actual user data in a natural user environment, subject to similar external conditions. Only a human, not a computer, can evaluate the usability and accessibility of the application and how it looks and feels to the end-user. It also gives a broader perspective of the system. Finally, some test scenarios just can't be automated and need to be manually tested. Thus, it is always recommended that you test the software manually before attempting to automate it.

2. What are the advantages and Disadvantages of Manual Testing?

Advantages of Manual Testing

- Preferable for products with a short life cycle.
- Saves time, money, and resources.
- Ensure the error-free product.
- Usable for exploratory testing, ad hoc testing, and usability testing.
- No need to change the entire code to make minor changes.
- Get accurate user interface feedback.
- Ability to handle difficult use case situations in a better way.
- GUI testing can be done accurately.
- Highly reliable.
- Make user-friendliness better.
- Easy to learn for new testers.

Disadvantages of Manual Testing

- Not suitable for time-bounded projects and large organizations
- More prone to human errors and mistakes
- Less efficient as the choice of recording the testing process is not available
- Less Reliable
- Regression testing is time-consuming
- Does not cover all the aspects of testing
- Load testing and performance testing can be performed manually
- More expensive in the long run process

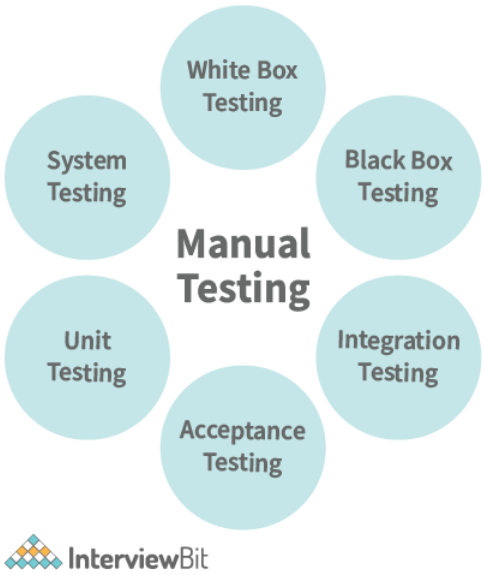
3. Name some of the manual testing tools.

Some of the top **manual testing** tools include:

- Postman
- Message queue monitors
- DB tools, etc.

4. What types of manual testing are there?

In the course of the test life cycle, there are different manual testing types or manual testing techniques that may be used. Following is a list of them:



- Black Box Testing
- White Box Testing
- Unit Testing
- System Testing
- Integration Testing
- Acceptance Testing

5. Who is a manual tester? Write its roles and responsibilities.

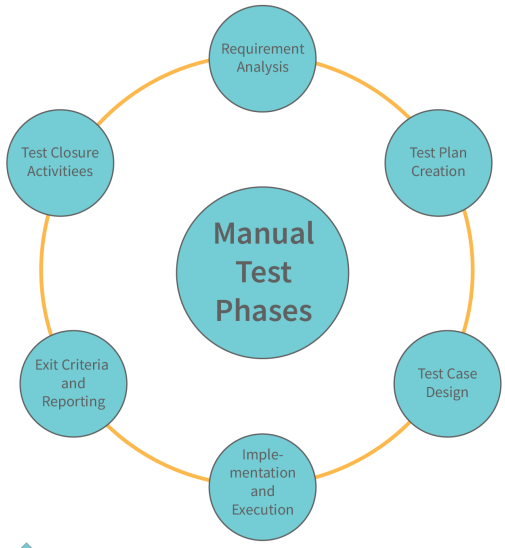
The manual tester is a professional who conducts quality checks on software applications without using automation tools or scripting. In essence, the speciality involves manually checking software for errors and fixing them. Manual testers must have the appropriate skills and be able to meet the company's requirements.

Manual Tester Roles and Responsibilities

- Analyzing client requirements.
- Reviewing written code for compliance with project specifications.
- Creating a test environment for executing test cases.
- Establishing quality assurance strategies and organizing phased testing.
- Organizing and conducting review meetings.
- Executing and analyzing test cases.
- Detecting and fixing bugs.
- Monitor system errors and discuss them with colleagues.
- Keeping in touch with the test manager, etc.

6. Describe the manual testing process.

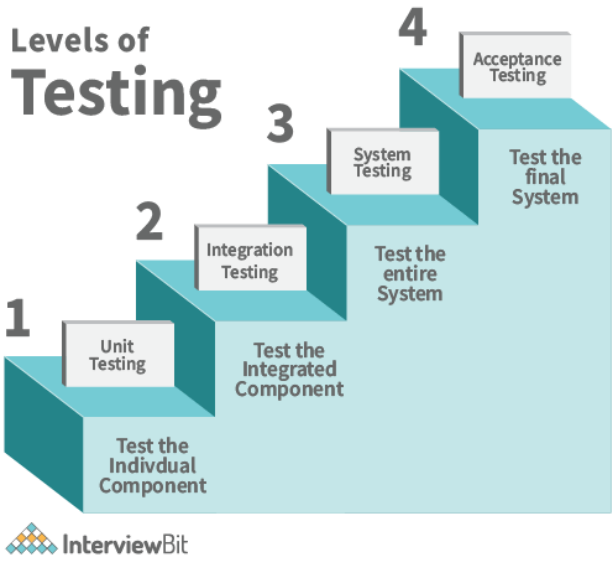
Among the steps involved in manual testing are:



- Requirement analysis
- Test plan creation
- Design test scenarios and test cases
- Test execution and defect reporting
- Evaluating exit criteria and reporting
- Test closure activities

7. Can you tell me what the different levels of manual testing are?

Different levels of testing can be carried out during the development process. Multilevel testing facilitates the identification of bugs early in the development process. The four levels of testing are as follows:



- **Unit testing:** Essentially, it is a way of testing logically isolated pieces of code within a system called units. Mainly, it focuses on the standalone module's functional accuracy.
- **Integration Testing:** Software testing at this level involves combining and testing individual units to see if they work together as they should. This test focuses on the interface between modules.
- **System Testing:** It involves testing all components of the product as a whole to ensure that overall product requirements are met. The types of system testing include regression testing, usability testing, and functional testing.
- **User Acceptance Testing:** Acceptance testing, also known as UAT (User Acceptance Testing), is the final step in the software testing process. This test determines if the software is ready for release.

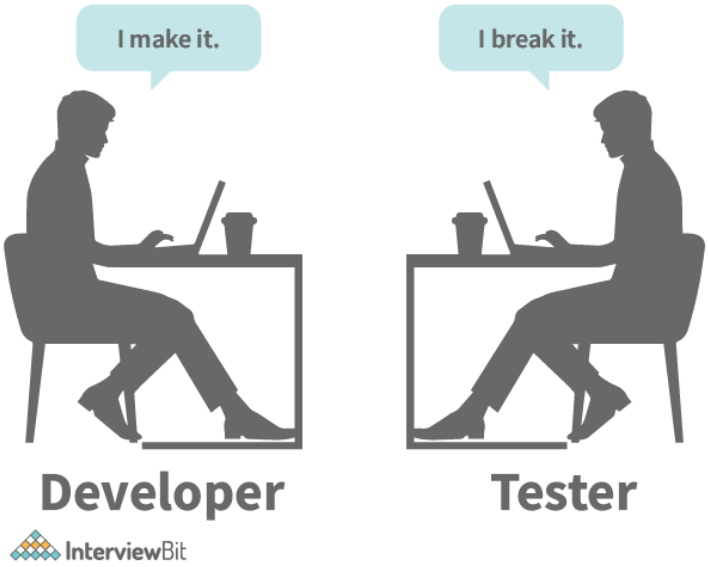
8. In order to perform manual testing, what skills are required?

The following are the important manual testing skills to acquire:

- Detail-oriented and able to report test results in a professional manner.
- A strong analytical ability.
- Ability to perform technical testing.
- Familiarity with Agile methodologies.
- Plan and track the testing process.
- Knowledge of SDLC, STLC, SQL, and manual concepts.
- An understanding of test management tools, test tracking tools, and testing techniques.

9. What is the difference between developer vs tester?

Software developers and testers differ in the following ways:



Developer	Tester
Software developers write and maintain source code for computer programming.	A software tester is responsible for identifying the quality, correctness, and completeness of software.
Its responsibility is to create individual software applications.	Its responsibility is to evaluate individual software programs.
Basically, it involves developing software through successive phases in an orderly manner.	This test evaluates how well a software application works.
They solve problems quickly, reduce costs, increase flexibility, and improve the quality of business.	By reporting problems as soon as possible, they help save money, provide security, and ensure the quality of the software.
Additionally, they provide suggestions on how to improve software applications.	Not only do they find bugs, but they also identify their root causes, so bugs can be permanently fixed.
Coding skills, time management skills, and programming skills are all essential for developers.	An ideal tester should be well versed in the system being developed, possess good communication skills, be a critical thinker, etc.

10. What is test coverage?

Test coverage is a metric that indicates how much of the source code is covered by the tests, allowing the tester to verify the quality of their testing. Testers can use it to determine whether they're testing everything they're supposed to. Depending on the way people approach testing, test coverage can mean different things to different people.

- **Product:** It means looking at test coverage to answer the question: Which features or areas of the software do your tests cover?
- **Requirements:** The software might work well, but it's not useful to the customer if it doesn't satisfy their needs. Requirements coverage indicates how many of the requirements are tested.
- **Source Code:** This is usually a developer's domain and is a white-box testing technique. The developer can check how much of their source code is covered by the unit tests.

11. Name some methods that can be used in code coverage.

Code coverage is a software testing metric that measures how many blocks, lines, or arcs of code are executed when a test suite runs. Code coverage can be determined by several methods, including:

- Statement Coverage.
- Decision Coverage.
- Branch Coverage.
- Toggle Coverage.

12. Define Latent Defect.

Latent defect, as the name suggests, is a type of defect or bug which has been in the software system for a long time but is discovered now. A latent defect is an existing defect that can be found effectively with inspections. It usually remains hidden or dormant and is a low-priority defect.

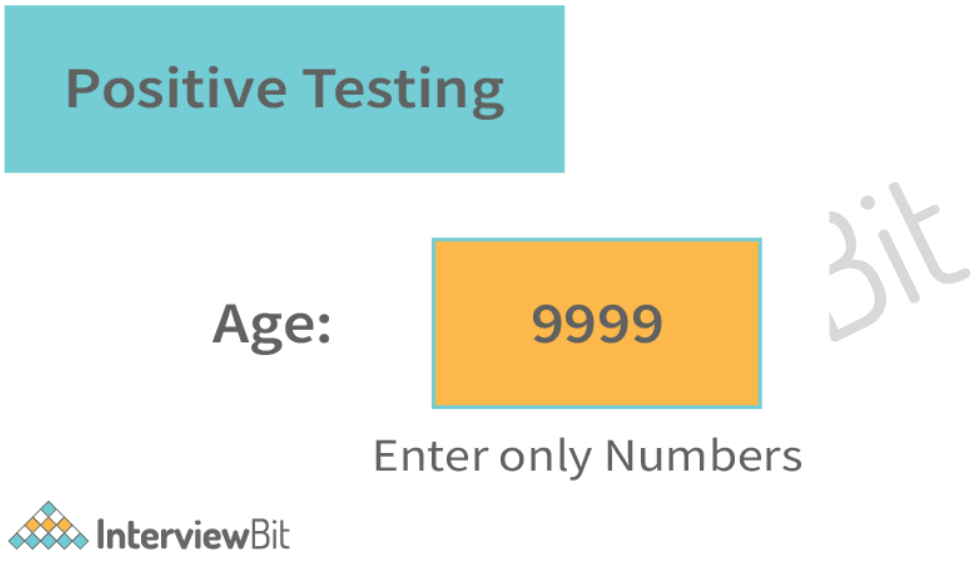
13. Name some attributes of the test case.

There are various attributes of test cases that make them more reliable, clear, and concise, avoiding any sort of redundancy. Some of them are given below:

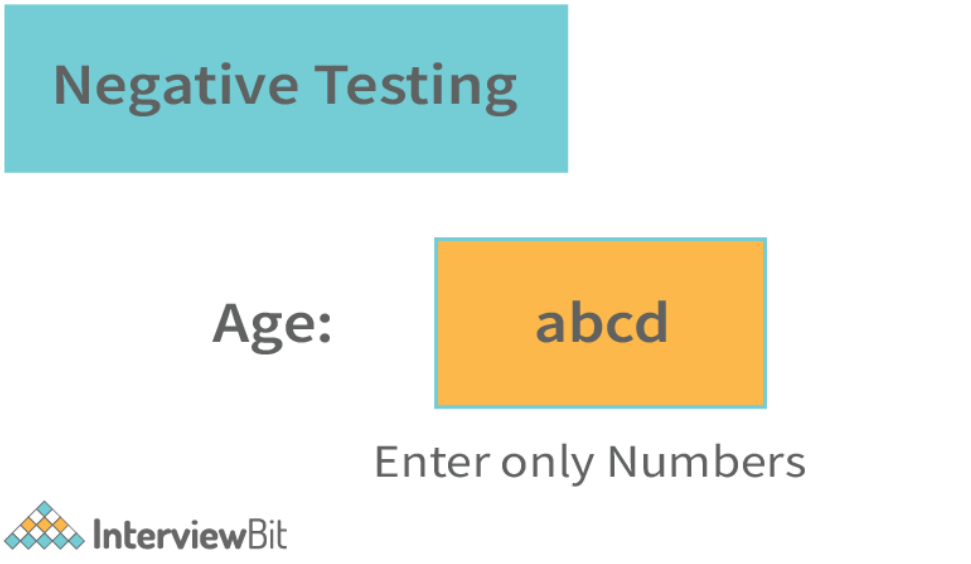
- **Test Case Id:** Unique identifier of the test case.
- **Test Summary:** One-liner summary of the test case.
- **Description:** Detailed description of the test case.
- **Prerequisite or pre-condition:** Set of conditions to be followed before implementing the test steps.
- **Test Steps:** Detailed steps for performing test cases.
- **Test Data:** Test data value used in the test case.
- **Expected Result:** Estimated result to pass the test.
- **Actual Result:** Actual result after executing the test steps.
- **Test Result:** Status of the test execution (Pass or Fail).
- **Automation Status:** Identifier for automation.
- **Date:** Test execution date.
- **Executed By:** Person name executing the test case.

14. What is Positive and Negative Testing?

- **Positive Testing:** It is a type of testing process where a software application is validated against the valid data sets as input. It is simply used to check whether the application does what it is supposed to do or not.



- **Negative Testing:** It is a type of testing process where a software application is validated against invalid data sets as input. It is simply used to check whether the system shows an error when it is supposed to do or not. In test case execution, negative testing is considered a very crucial factor.



Positive vs Negative Testing

Positive Testing	Negative Testing
It tests the application or system by giving valid data.	It tests the application or system by giving invalid data.
It accepts all the numeric and alphabetic values.	It does not accept any special character.
This type of testing is performed to identify a known set of test conditions.	This type of testing is performed to identify an unknown set of test conditions.
It is usually performed on each and every application.	It is usually performed where the chances of unexpected conditions or errors are more.
It requires less time and can be performed by people having less knowledge.	It requires more time and can only be performed by professionals.
It makes sure that the software application is normal.	It makes sure that the software applications are 100% detect-free.
It does not encompass all the possible cases.	It encompasses all the possible cases.
It is less significant or vital than negative testing.	It is more significant and vital than positive testing.

15. What is UAT (User Acceptance Testing)?

A user acceptance test, also known as an end-user test, is a testing methodology undertaken by the client or end-user to approve the production release. As the last step in the SDLC, it takes place only after testing the software thoroughly. This tool is primarily used to validate end-to-end business processes. It verifies whether or not the developed software is ready to float into the market.

16. Explain Test Driver and Test Stub.

A test driver and a test stub are both types of test harnesses that simulate an environment for testing a module or component. Both are dummy modules designed specifically for testing.

- **Test stubs:** A test stub is used in a top-down testing approach and allows testing of the upper levels of code when the lower code levels have not been developed yet. It serves as a 'called program' when subprograms are being developed.
- **Test drivers:** A test driver is used in a bottom-up testing approach and allows testing of the lower levels of the code when the upper code levels have not been developed yet. It serves as a 'called program' when main programs are being developed.

17. What is the importance of Localization Testing?

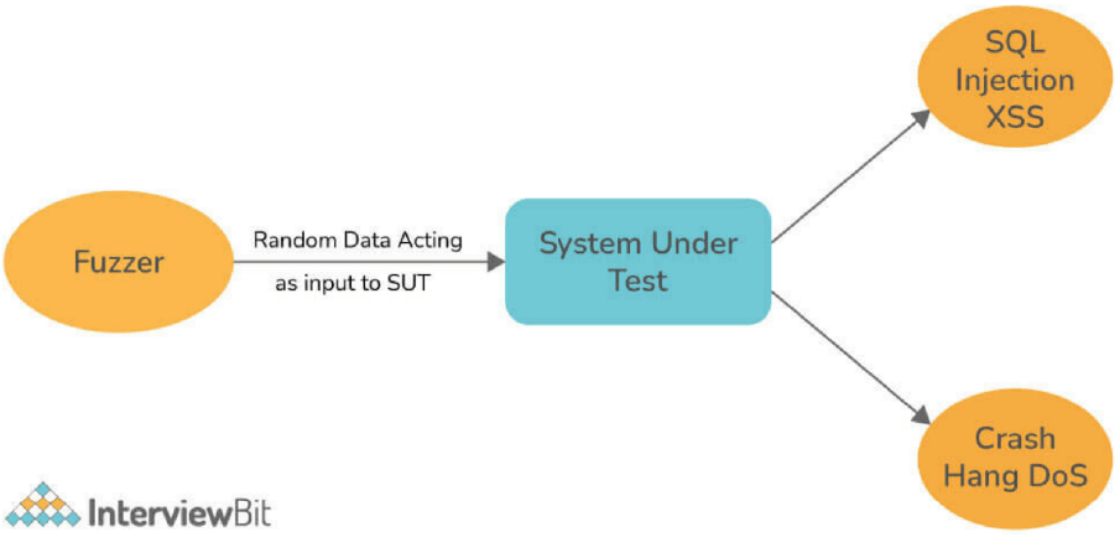
In localization testing, a software product is tested to ensure whether it offers full functionality and usability within a specific locale. Simply, it verifies the content's accuracy and suitability. In addition to linguistics, it addresses traditions, common herd behaviour, and other similar factors. Generally, it deals with the application's functionality and GUI.

18. What do you mean by Baseline Testing and Benchmark testing?

- **Baseline Testing:** It is a type of non-functional testing in which a set of tests are run to capture performance information. Using this gathered information, we can make required changes in the application and ultimately improve the performance and capabilities of the application. Generally, it refers to a benchmark that is used as a starting point for new works. This testing uncovers and resolves many errors.
- **Benchmark Testing:** It is a type of testing that involves both the developers and DBAs (Database Administrators) to determine current performance information. Using this information, one can improve the performance of the same by matching it with the benchmarks (industry standards). Its main objective is to compare the present and future software releases with their specific benchmark.

19. Describe what Fuzz Testing is and how important it is.

Fuzz testing is a software testing technique that uses a lot of random data, called fuzz, as input to find or detect security loopholes and coding errors in a software application. This is more useful for larger projects, but it only detects serious faults or defects. It is simply used to check the vulnerability of software and gives more effective results when used with beta testing, black box testing, etc.



20. Explain Configuration Testing.

Configuration testing is a software testing technique that is used to evaluate the configurational requirements of the software. It discovers the optimal configuration of the system under which the application performs at its best, therefore configuration testing is considered important. It also helps in identifying and resolving any compatibility issues.

21. Name two parameters that can be useful to check the quality of test execution.

Two parameters required to check the quality of test execution include:

- **Defect leakage ratio:** It represents the ratio of total potential rejections to the total overall production.
- **Defect reject ratio:** It represents the ratio of total rejections to the total overall production.

22. What is API testing?

An API (Application Programming Interface) test involves testing API directly and as part of an integration test to ensure they meet reliable, functional, performance, and security requirements. Due to the lack of a GUI, APIs are tested at the message layer. The tester writes code that makes an API request to the server that provides the API, provides the required inputs, collects the output from the response, and matches the actual output with the expected output.

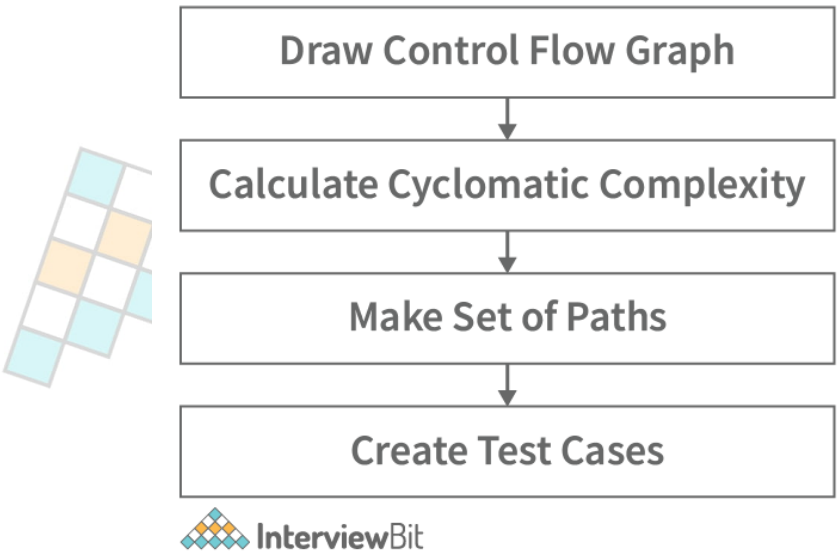
API testing primarily concerns the business logic of the software that's exposing the API. It does not involve the look and feel, accessibility, or usability of the software. API testing can be automated to make it repeatable and reproducible each time they run.

23. Explain use-case testing.

Use-case testing is basically a technique for developers and testers to identify test cases that exercise the entire system right from the very beginning to the very end of each transaction. Generally, it is part of black-box testing, which is used to develop tests or systems that achieve acceptable performance levels.

24. Explain Path testing.

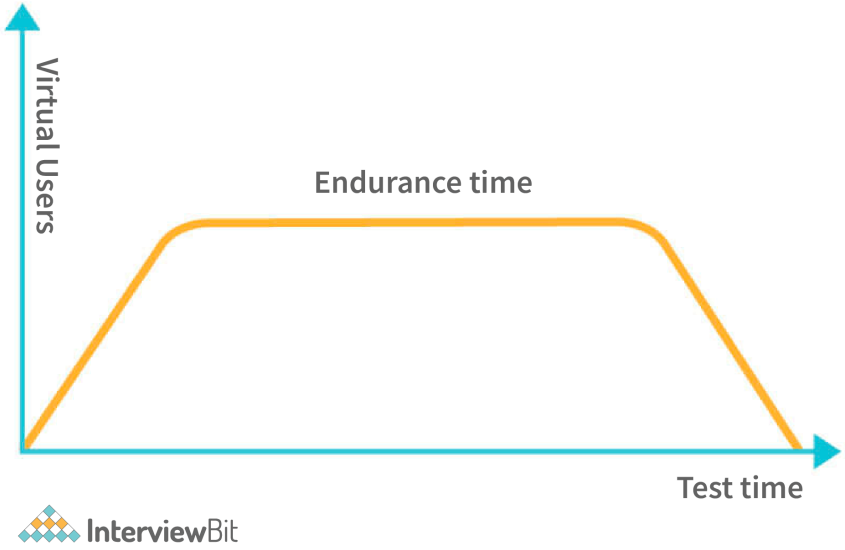
A path test is a type of test designed specifically for designing test cases. It involves the use of a control flow chart designed specifically to identify a set of linearly independent paths of execution leading to the completion of a program. The main objective of this process is to ensure that all paths are covered and executed well. In addition, it reduces or minimizes the likelihood of redundant tests occurring.



25. Explain Endurance Testing or Soak Testing?

Endurance testing, also known as Soak testing, is a type of performance testing usually performed to check the performance of the system that is under constant use. Its main purpose is to determine whether a system can sustain a continuous high load or not. Memory utilization is also monitored to identify potential leaks during this testing. Some of the endurance testing tools include:

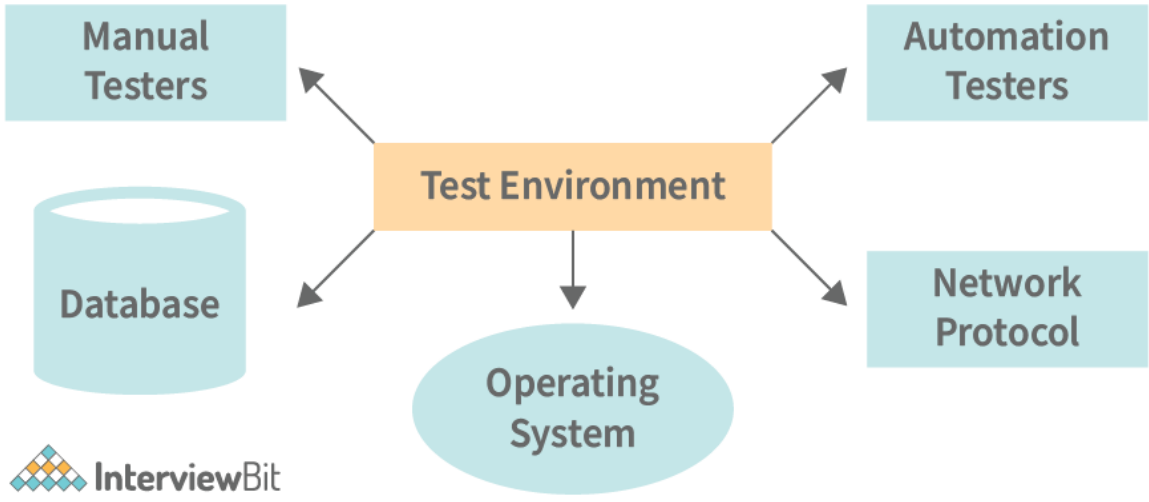
- WebLOAD
- LoadUI
- OpenSTA
- LoadComplete
- Apache JMeter, etc.



Manual Testing Interview Questions for Experienced

26. Explain the term testbed.

Testbed is generally referred to as a digital platform that is used for testing an application. It includes an operating system, database, hardware, network configuration, software application under test, and all other software-related issues.

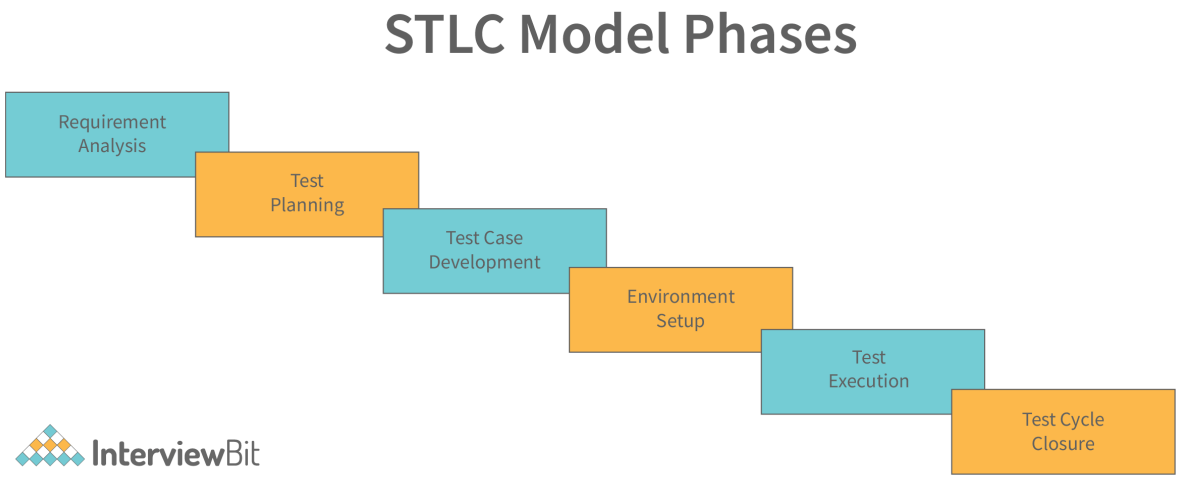


27. Explain bugs, defects, and errors.

- **Error:** The error occurs when there is a programming mistake in the code that prevents the program from executing or compiling.
- **Defect:** A defect is any variation between the actual result and the expected result determined by a tester or developer. Defects are typically detected after the product enters production and are resolved only during the development phase.
- **Bug:** A software bug is detected during the testing phase as a fault or mismatch. This affects the functionality and performance of the software.

28. What is the software testing life cycle?

STLC (Software Testing Life Cycle) is a fundamental part of SDLC which is used to test software and ensure that the quality standards are met. Verification and validation are generally involved in this process. In this, different activities are executed in a specific order throughout the software testing process. There are basically six different phases in STLC Model as shown below:



- **Requirement Analysis:** The QA team analyzes the requirements to determine what we are going to test and what the testable requirements are.
- **Test Planning:** This phase involves defining the test strategy. This step determines the project's objective and scope.
- **Test Case Development:** This stage involves defining and developing detailed test cases. Test data is also prepared by the testing team for testing.
- **Test Environment Setup:** This is the software and hardware setup that the testing teams use to execute test cases.
- **Test Execution:** This involves executing code and comparing it with the expected results.
- **Test Cycle Closure:** This involves calling out a meeting of the testing team members to evaluate and assess cycle completion criteria based on cost, time, test coverage, quality, and critical business objectives and software.

29. What is black-box testing?

In black-box testing, the tester views the software as a black box, ignoring all the internal structure and behaviour. It is solely concerned with the input provided by the user and the output generated by the system. Black-box testing verifies the program’s behaviour against the specified requirements.

As part of black-box testing, test conditions are created based on the software's functionality without knowing how it works internally. In this approach, the software is tested from the point of view of the end-user and provides a broader view of the entire system. In light of the fact that users are only concerned with whether the software meets their needs, rather than how it works, black-box testing is an excellent way to test software usability and anticipate how customers will use it.

30. What white-box testing?

White-box testing is an alternative to black-box testing that involves viewing the system as a transparent box. It is possible for the testers to observe the internal implementation of the system, which helps them conduct the test. In most cases, white-box testing is performed by the software developers during the development process.

It is also referred to as closed-box testing.

In white-box testing, we assume that the tester has some programming knowledge. The test covers all possible branches a program could follow in a running system. The more you know about the inside of a system, such as its source code and implementation details, the more thoroughly you can test it.

31. State the difference between bug leakage and bug release.

Bug leakage and bug release differ in the following ways:

Bug Leakage	Bug Release
Bug leakage refers to a defect that exists during testing yet is not discovered by the tester but is eventually discovered by the end-user.	When a piece of software is released with a known bug(s) or defect(s), it is termed a bug release.
It is usually a high priority/severity bug.	It is not a high priority/severity bug.
The bugs need to be addressed immediately and resolved as soon as possible.	Software companies do this when they can afford to release software with bugs, but cannot afford to fix them in that version.

32. What do you mean by Critical bug?

The term critical bug refers to a bug that affects the majority of an application's functionality. When a critical defect occurs, testing cannot proceed until the defect is fixed. These block the functionality of an entire system or module. An application returning a server error message after an attempt to log in is an example of a critical defect.

33. What do you mean by Data flow testing?

Data flow testing involves analyzing the flow of data within a program. It is a type of structural testing. It is possible, thus, for a programmer to perform various tests on data values and variables. This method provides a way to identify the variables that are used at every stage of the program's control flow. It helps us in the following ways:

- Eliminate or remove variables that are never used after being declared.
- Pinpoint variables that are used but never declared.
- Deallocate variable before it is used.
- Pinpoint variables that are defined multiple times before it is used.

34. What are the differences between manual and automated testing?

Manual and automated testing differ in the following ways:

Manual Testing	Automated Testing
A human tester tests the software by manually running the test cases and observing and comparing the actual and expected outputs.	A tester or a programmer uses scripts and tools that execute the software and compares the actual and expected outputs.
A manual test cannot be reproducible and repeated.	Automated testing can be consistently reproduced and repeated since it is programmed. Testers can run it as many times as they want.
It is easy for testers to test new features manually, without much configuration or setup.	Automated testing requires an initial investment in creating tests and preparing a testing environment.
A manual test can help identify bugs in the user interface or accessibility issues.	The automated testing method is more effective at catching bugs that humans would miss, such as software bugs or errors in business logic.
Tests conducted manually are prone to human error and take a long time.	Compared to manual testing, automated testing is more reliable since there is no human involvement involved (apart from writing tests). It is significantly faster than manual testing.

35. State difference between static and dynamic testing.

Static and Dynamic testing differ in the following ways:

Static Testing	Dynamic Testing
In static testing, software applications are tested without executing any code at the very beginning of the SDLC.	The test checks software functionality, memory/CPU usage, and system performance at the end of the development cycle.
It is performed during the verification stage.	It is performed during the validation stage.
This is performed prior to the deployment of the code.	Following the deployment of the code, dynamic testing is performed.
A static test prevents defects from occurring.	In dynamic testing, defects are detected and fixed.
A walkthrough, technical review, and inspection are all part of this process.	Functional and non-functional testing is involved in this process.
Errors are found early in the development process, thereby improving the quality of software applications.	The main objective is to ensure that the software product meets the business requirements.

36. What is the term ‘quality’ mean when testing?

In general, quality software is usually free of bugs, is delivered on time and on budget, meets most of the requirements and/or expectations, and is easy to maintain. However, 'quality' is a subjective concept. A lot depends on whom the "customer" is as well as the extent to which they are influential in general. For instance, a user may define quality as user-friendliness and bug-free while an accounting department might define quality as profits.

37. What is the difference between Quality Control(QC) and Quality Assurance(QA)?

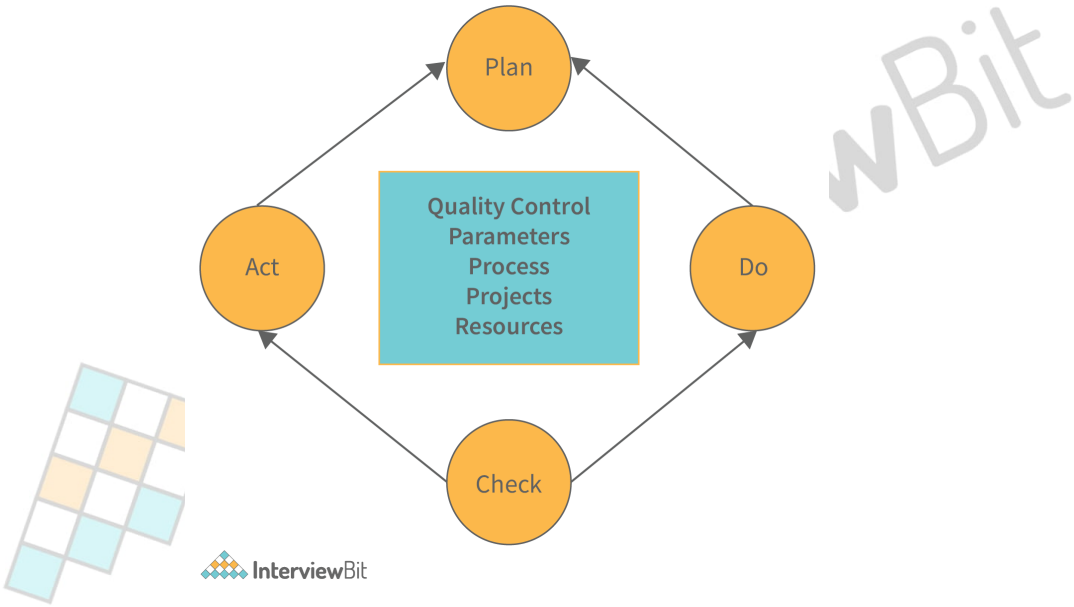
Quality Assurance:

- QA stands for Quality Assurance. As part of a software development team, a QA ensures that the software is thoroughly tested before being released to the end-user. It ensures that the shipped software is of high quality.

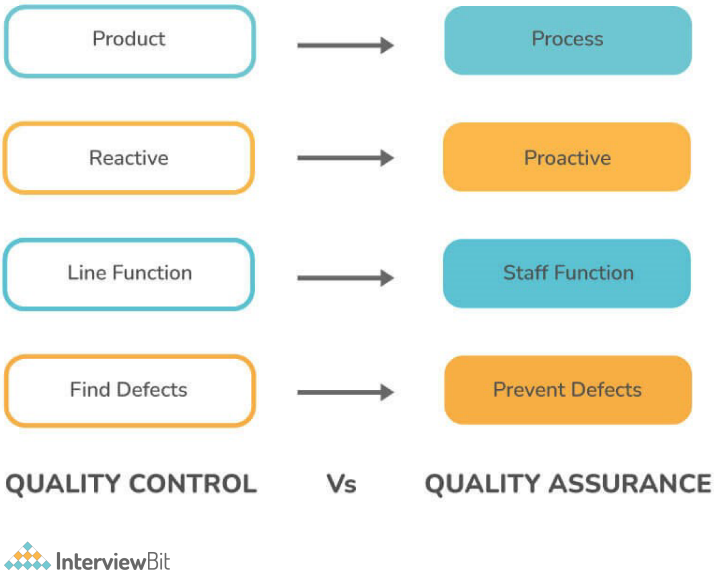
- It focuses on improving the software development process and is typically carried out during the development phase. It is possible for testers and quality assurance personnel to be the same person in many software organizations, although the roles may differ depending on the organization's size.

Quality Control:

- QC stands for Quality Control. The main goal of this process is to verify that the developed products meet the required standards.

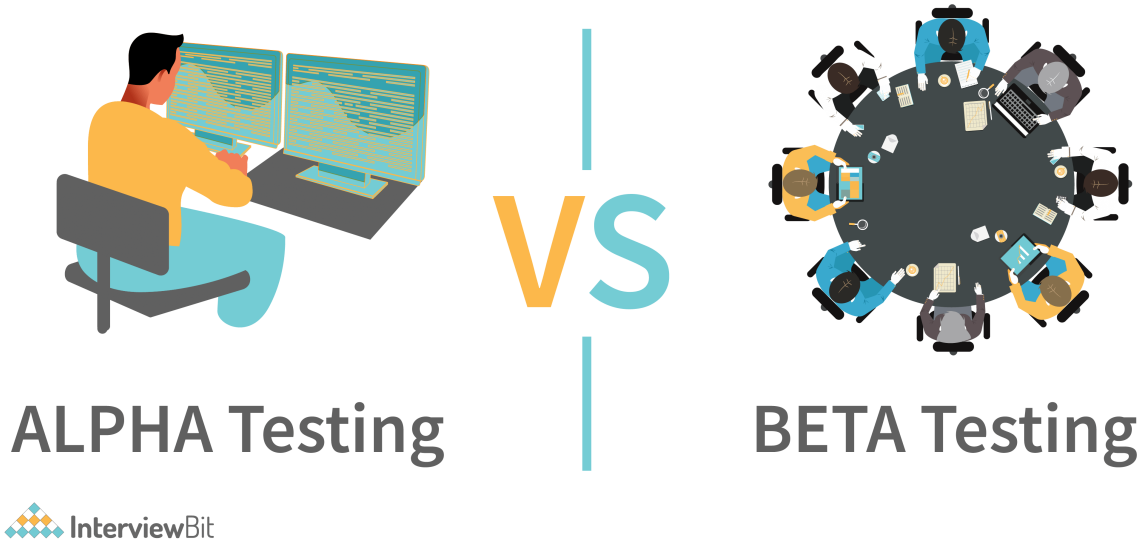


- Software quality-control tests and reviews the functional and non-functional requirements of a software product to ensure its quality. QC activities are typically performed after a product is developed to assess the quality of end products.



38. State difference between alpha testing and beta testing.

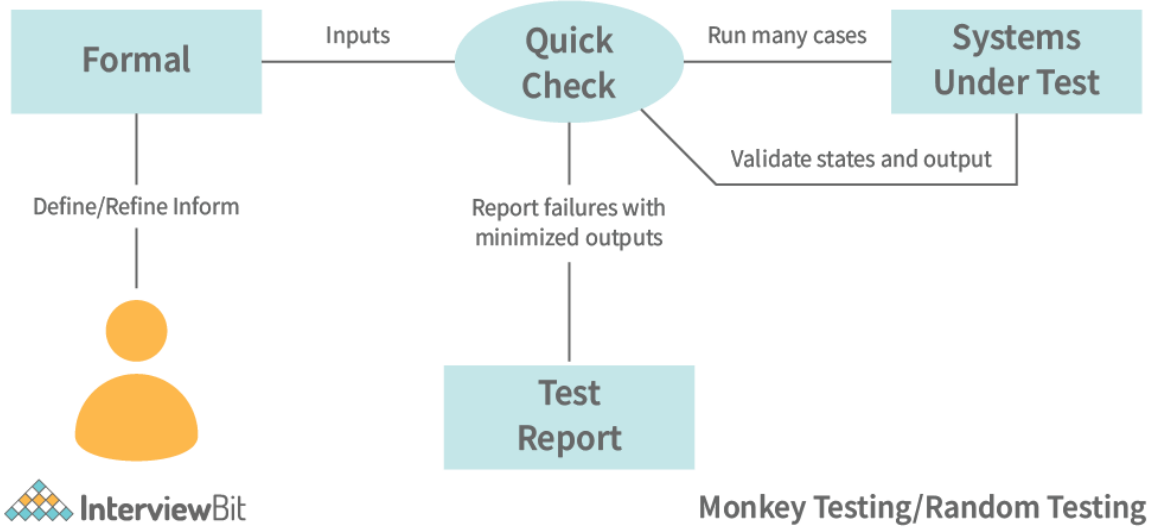
Alpha and beta testing differ in the following ways:



Alpha Testing	Beta Testing
During this type of testing, bugs are identified before the product is released to users. An example of a user acceptance test is Alpha Testing.	The test is conducted by real users of the software application in a real environment. An example of a User Acceptance Test is beta testing.
Both white box and black box testing are involved in Alpha testing.	A black-box testing is commonly used in beta testing.
In most cases, alpha testing is done by internal testers within an organization.	Clients who are not affiliated with the organization perform beta testing.
As the activities are performed on the developer's site, they can be controlled.	Because activities are performed in the real world, in the end, the user's environment, cannot be controlled.
Alpha testing does not include robustness and security tests.	Beta testing includes robustness and security tests.
The quality evaluation is the main objective.	Customer satisfaction is the main objective.
Alpha testing allows developers to quickly address critical issues.	Most of the issues or feedback gathered during beta testing will be incorporated into future versions.

39. Explain Monkey Testing and Performance Testing.

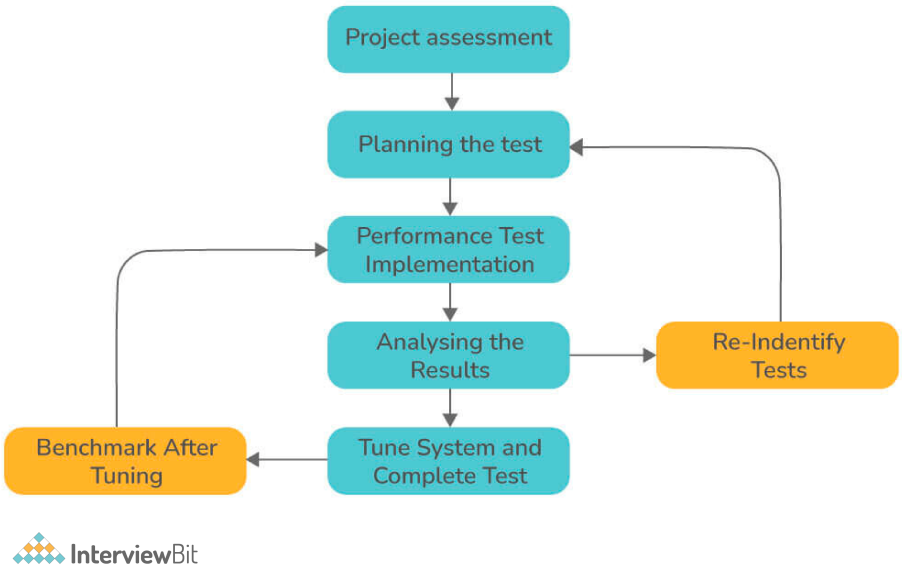
Monkey Testing: Monkey testing, also known as Random Testing, is a type of software testing technique in which data is generated randomly using a tool or some automated mechanism. This randomly generated input is used to test the system, and the results are analyzed accordingly. Testing of this type does not follow any rules.



Monkey Testing/Random Testing

Performance Testing: It is a type of non-functional software testing technique that is used to determine the system parameters like speed, scalability, and stability under different workload conditions. Its main purpose is to eliminate performance bottlenecks, not to find bugs. Some of the key parameters of performance testing include:

- CPU Utilization
- Memory Utilization
- QPS/TPS (Transaction per second)
- Average load time
- System throughput, etc.



40. What is the role of documentation in manual testing?

Effective software testing relies heavily on documentation. Documentation should include details such as requirements specifications, business rules, configurations, designs, test plans, code changes, test cases, bug reports, inspection reports, user manuals, etc. As part of software testing, the following documentation artifacts are commonly applied:

- **Test Plan:** It is essentially a dynamic document controlled and monitored by the testing manager. A well-written test plan that describes the scope and activities of software testing is crucial for the success of a testing project. In essence, it serves as a blueprint for all aspects of the testing process, such as what, when, how, and more.
- **Test Scenario:** A test scenario is a detailed description of a set of test cases or use cases. It involves testing a software application from the end user's perspective. It is usually the basis for constructing lower-level test cases or use cases. The test scenario can also be referred to as the test condition or the testing possibility. Taking a look at it will help you understand what we need to test.
- **Test Case:** As the name implies, a test case is a document that contains test data, expected results, preconditions, and postconditions. The purpose of this document is to ensure the software product meets the specific requirements for a specific test scenario. Manual testing involves executing test cases manually by a tester without relying on automated tools. In the process of developing test cases, it is possible to identify loopholes in the specifications.
- **Traceability Matrix:** This is a document, usually contained in a form table, that illustrates the relationship between requirements and other project artifacts from start to finish. To put it simply, it maps customer requirements to test cases.

41. Explain RTM (Requirement Traceability Matrix).

The RTM (Requirements Traceability Matrix) is defined as a tool used to identify and track the requirements and deliverables of a project. This is accomplished by establishing a thread for each component. In addition, it manages the overall requirements of the project. There is nothing complicated about this method, and anyone can do it.

RTMs come in many forms. A test matrix, for example, proves that tests were conducted. Additionally, it can be used during the software development process to identify issues and requirements.

42. What is the importance of agile testing?

The agile testing process involves software testing that adheres to agile software development principles. The software is evaluated from the customer's perspective. This software development practice involves frequent, automated testing of new code and the immediate fixing of defects as soon as they are discovered. Each feature is tested as it is developed. Among its advantages are:

- All testers and developers can work together, boosting performance.
- Each feature is tested as it is developed.
- Ensures a high-quality product that meets customer expectations.
- Reduces costs and saves time.
- Highly adaptable and flexible.
- Assists developers with releasing software as soon as possible and improves product quality.

43. What is the difference between Regression and Retesting?

- **Regression Testing:** Regression testing, also known as generic testing, revolves around re-running functional and non-functional tests. It is especially done to ensure whether previously developed and tested software still performs the same after a change or not. It can be performed either manually or using automated tests
- **Re-testing:** Re-testing, also known as planned testing, is used for specific bugs after it has been fixed by the developers. Re-testing is performed to check the scenario under the same environmental conditions after detection has been fixed.

Regression vs Retesting

Regression	Retesting
It is performed to make sure that the changes haven't affected the unchanged part.	This is done to ensure that the test cases which were filed during the last execution are passed after the detected bugs have been fixed by developers.
It is not carried out to fix specific detects.	Usually, it is based on fixing defects.
It is only the previous version functionality centric.	It is current or previous version functionality centric.
It can be performed parallel with retesting.	It is needed to perform before regression testing.
It does not include the verification of bugs.	It includes the verification of bugs.
In this type of testing, test cases can be automated and the testing style is generic.	In this type of testing, test cases cannot be automated and the testing is done in a planned manner.
It is only used for passed test cases.	It is only used for failed test cases.

44. What is System testing and Unit Testing? Write the difference between them.

- **System Testing:** It is a typical black box testing technique that is performed in a complete and fully integrated system to evaluate the system's compliance with its specific requirements. It must investigate both functional requirements and non-functional requirements. Generally, it is performed by both testers and developers.
- **Unit Testing:** In unit testing, each component of the software is individually tested. Generally, unit testing is performed by developers. Those systems that have a lot of interdependencies between their modules cannot be tested by unit testing.

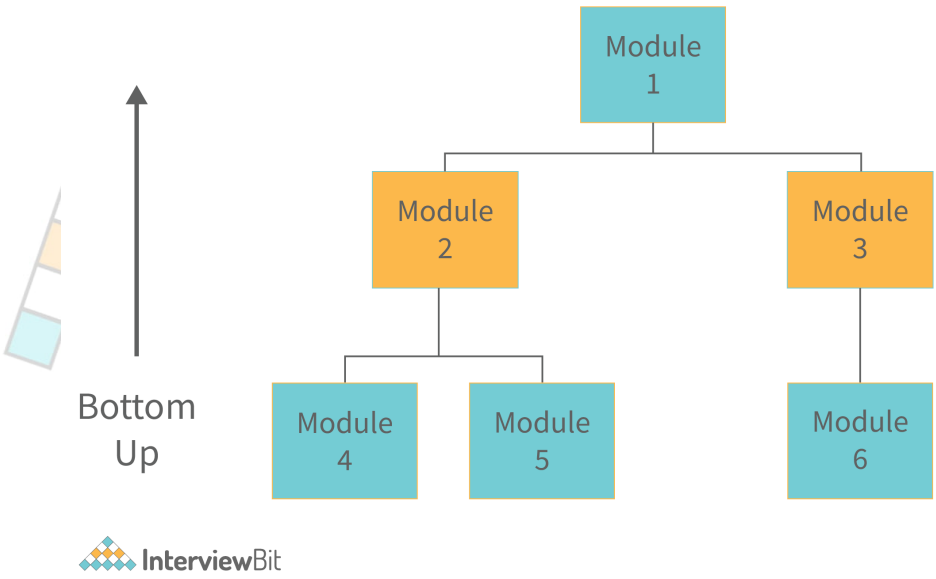
System vs Integration Testing

System Testing	Unit Testing
The system testing method involves treating each module as a separate target for testing, and integrating the modules after each has been tested.	The purpose of unit testing is to test only one module at a time, rather than the integrated version of the application.
Generally, when it comes to unit testing, a single module testing approach is taken.	For System test cases, it includes both top-down approach testing and bottom-up approach testing with all modules in integrated mode.
It focuses on system validation.	It focuses on functional verification.
It usually follows the requirements specification.	It usually follows the specification of modules.
It is also known as black-box testing.	It is also known as white-box testing.
It is a low-level test as compared to unit testing.	It is a high-level test as compared to system testing.

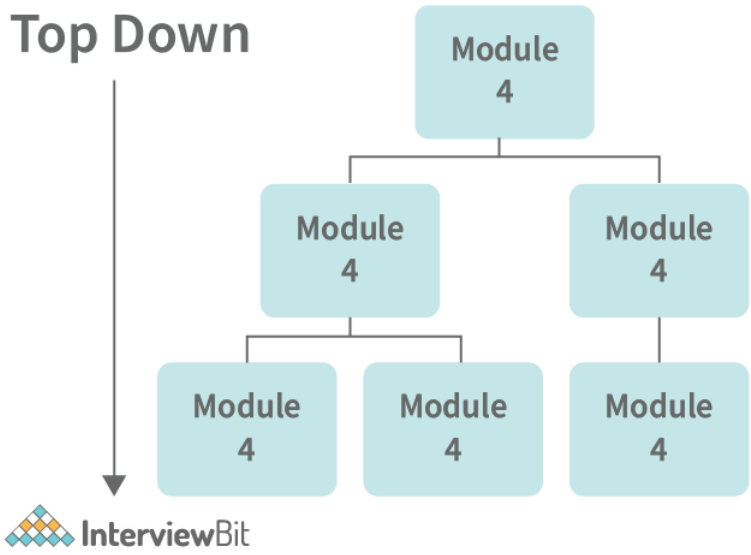
45. What are the types of Integration Testing?

Integration testing includes the following types:

- **Big bang testing:** It involves integrating all the modules and components at once and then testing them as a whole (single unit). When testing these components together, they are treated as an entity. The integration process will not proceed if all the components of the unit are not completed.
- **Bottom-Up Testing:** This strategy involves testing lower-level modules first, then moving on to higher-level modules. As long as top-level modules have been tested, the process continues. Upon integrating and testing the lower-level modules, the next level of modules will be created.



- **Top-Down Testing:** This strategy involves testing software systems from top to bottom according to the control flow. Tests are conducted first on the higher-level modules, followed by tests and integration of the lower-level modules to verify the functionality of the software. Testing is carried out using stubs when some modules are not yet ready.



46. Name some of the most popular integration testing tools.

Among the most commonly used integration testing tools are:

- DBUnit
- Mockito
- Greenmail
- REST-Assured
- JUnit 5
- H2 Database, etc.

47. What is Test Harness and Test Closure?

Test Harness: Test harness, also known as the automated test framework, is a collection of software and test data required to unit test software modules during development. It is mostly used by the developers and helps in the automation and execution of unit test cases. It generally includes two main parts as given below:

- Test execution engine
- Test script repository

Test Closure: Test closure is basically a document that provides the summary of all the tests that are performed during SDLC. It gives full detailed analysis reports of the bugs that are discovered and removed. It is usually performed prior to the end of the testing process.

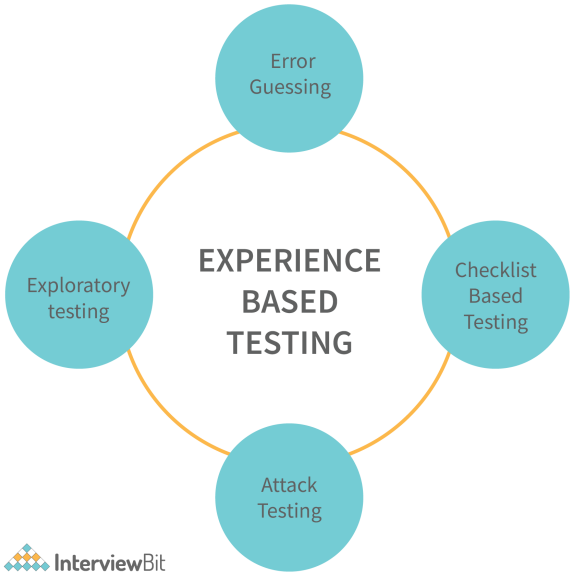
48. Explain different stages of the defect life cycle.

The defect life cycle consists of the following stages:

- **New:** Potential defect that hasn't been validated.
- **Assigned:** Assigned to a team for resolution, but not yet resolved.
- **Active:** Developers are currently investigating the defect and addressing it. Deferred or Rejected are the two possible outcomes at this stage.
- **Test:** The defect has been fixed and can be tested.
- **Verified:** The defect has been retested and the results have been verified by QA.
- **Closed:** Defect in its final state, which can be closed after retesting by QA or when considered duplicate or not a defect.
- **Reopened:** QA reopens/reactivates a defect when it has not been fixed.
- **Deferred:** A defect that can't be addressed in the current cycle is deferred to a future release.
- **Rejected:** There are three common reasons for rejecting a defect: duplicate, not a defect, and not reproducible.

49. Explain Experienced-based testing techniques.

The experience-based testing technique is a type of testing that is based on the tester's experience with testing to understand the essential areas of a system. This type of testing is generally used in a low-risk system. Individuals' information, abilities, and foundation knowledge are prime supporters of the test conditions and experiments in experienced-based techniques. There are four different experienced-based testing techniques as shown below:



50. Write the difference between smoke testing and sanity testing.

- **Smoke Testing:** It is a type of testing performed to ensure that the acute functionalities of the program are working well. It acts as a confirmation of whether the quality assurance team can further proceed with testing or not.
- **Sanity Testing:** It is an unscripted form of testing performed to ensure that the code changes that are made are working well. It is performed by the test team for some basic tests. This testing focuses on one or a few areas of functionality and is usually narrow and deep.

Smoke Testing	Sanity Testing
In this test, the test team measures the system's stability before moving on to more rigorous testing.	In this testing, the test team measures the rationality of the system before moving on to more rigorous testing.
Developers or testers usually perform this task.	Testers usually perform this task.
This is a subset of regression testing.	This is a subset of acceptance testing.
Both manual and automated methods can be used to execute it.	Automated tools cannot be used to execute it; they must be done manually.
In most cases, it takes place during the development of a new product.	The test is generally performed after the regression test.
There is documentation, and it is used to test the application's end-to-end functionality.	There is no documentation, and it is used to test only functions that have been modified or have been fixed.
It is also considered a subset of acceptance testing.	It is also considered a subset of regression testing.

51. What do you mean by pesticide paradox? What you can do to overcome it.

The pesticide paradox is basically a phenomenon where the more one tests the software, the more it becomes immune to its tests. To overcome this, testers should always find new strategies, approaches, and test cases, so that they can identify bugs and resolve them. The following methods can be used to prevent the pesticide paradox:

- Create a new set of test cases for different components of the software.
- Adding new test cases to the existing test cases.

The use of these methods can lead to finding more defects in the areas where defect levels have declined.

Manual Testing Scenario-Based Interview Questions

52. When to choose manual testing over automation testing and vice versa?

Choosing Manual Testing over Automation Testing

- When test cases need to be run for a short duration of time (once or twice).
- When one needs to perform exploratory testing, usability testing, or ad-hoc testing.
- When assessing an application's user-friendliness.
- Whenever flexibility is needed.
- Whenever one wants to better manage complex situations/scenarios.

Choosing Automation Testing over Manual Testing

- Whenever test cases have to be run repeatedly over a long period of time.
- When one needs to perform performance testing, load testing, or regression testing.
- Whenever one wishes to record the testing process.
- When one has a limited amount of time to complete the testing phase.
- When tests are needed to be executed in a standard runtime environment.
- When tests involve repetitive steps.
- When there are multiple and quick deployments for the product, the manual becomes very time taking and redundant.

53. In what way will you determine when to stop testing?

Testing can be quite challenging when it comes to determining when to stop. In the modern world, many software applications are so complex and run in so many interdependent environments, that complete testing is impossible. The following factors are often considered when deciding when to stop testing:

- If deadlines are met (release deadlines, testing deadlines, etc.) and there are no high-priority issues left in the system.
- Completion of test cases with a certain passing percentage.
- As soon as the test budget is depleted.
- The mean time between two inherent failures is known as the MTBF (Mean Time Between Failure). When the MTBF is quite high, the testing phase may be stopped depending on stakeholder decisions.
- As soon as the automated code coverage meets a specified threshold value and there are no critical bugs.
- If the bug rate drops below a certain level.
- After the Beta or Alpha testing period has ended.

54. Can 100% testing coverage be achieved? How do you ensure test coverage?

Testing a product 100% is considered impossible. You can, however, get closer to your goal by following the steps below.

- Developing an effective testing strategy.
- Prepare a checklist for all activities related to testing.
- Establish a priority list for the application's critical areas.
- List all application requirements.
- Identify the risks associated with the application.
- Utilize automated testing.

55. System testing can be done at any stage. Yes or No?

No, system testing cannot be conducted at any stage of the development process. In system testing, all components of the software are tested together to ensure that the overall product meets the specified specifications. Therefore, system testing cannot take place at any stage; instead, it must be done only after all modules or units are in place and are working properly, but before User Acceptance Testing (UAT).

56. If proper documentation is not available for testing, what steps will you take to overcome the challenge?

QAs should refer to the following references if they cannot find standard documents such as System Requirements Specification or Feature Description Document.

- Screenshots
- Wireframes
- A previous version of the application.

In addition, having discussions with the business analyst and the developer is another reliable method. This is helpful in resolving doubts and bringing clarity to requirements. The emails exchanged could also serve as testing references.

Another option for verifying the application's functionality is to perform smoke testing. This would expose a few very basic bugs in the application. In cases where none of the above options work, we can simply use our previous experience to test the software application.

57. What are some best practices that you should follow when writing test cases?

When writing test cases, you should follow the following guidelines:

- Assess the project's risks and deadlines before planning and prioritizing test cases and write accordingly.
- Keep the 80/20 rule in mind. For the best coverage of your application, 20% of your tests should cover 80% of its functionality.
- Make sure you don't try to test all cases at once, but rather improvise them as you go.
- Create a list of your test cases and categorize them according to business scenarios and functionality.
- Modularity and granularity are important when designing test cases.
- Provide test cases in a way that can easily be understood by others and modified if necessary.
- Remember that the software designed is ultimately for customers, so keep their requirements in mind.
- Manage a stable release cycle by using a test management tool.
- Keep track of your test cases on a regular basis. Test cases must be unique and irrelevant or duplicated must be removed.

58. When the requirements are still in flux, what is the best way to test a product?

For some products, a requirement stack is not available. It may require considerable effort to identify if an application has unexpected functionality, which indicates a deeper problem with the software development process. Removing functionality that isn't necessary for the purpose of the application is a good idea. Otherwise, create a test plan based on the assumptions you've made about the product. But, it is important that you thoroughly document all assumptions in the test plan.

59. What makes boundary value analysis a good method for providing test cases?

A boundary value analysis is defined as a software testing technique that uses the boundary values of equivalence classes as input to test cases. Black box testing uses boundary value analysis as one of the most commonly used case design techniques.

In boundary value analysis, values at the boundaries are included in test cases. It is generally true that there are more errors at the boundaries of an input domain than in its center, which makes boundary value analysis an excellent method for generating test cases. In boundary value analysis, values at the boundaries are included in test cases. When the input lies within the boundary range, it is a positive test, but if it lies outside, it is a negative test. The values may include the maximum, the minimum, the inside edge, the outside edge, the typical value or the error value.

Example: Assume you're testing an input box accepting numbers from 1 to 20. As a result of boundary value analysis, we can divide test cases into three categories:

- The test data will be the same as the input boundaries of input: 1 and 20.
- Input values above the extreme edges: 2 and 21.
- Input values below the extreme edges: 0 and 19.

Therefore, the boundary values are 0, 1, 2, and 19, 20, 21.

60. How do you know the code has met specifications?

Code that is maintainable, readable, and bug-free is considered good. Almost every organization has 'coding standards' that developers should adhere to, but everyone has different ideas about what's best and what's too many or too few. Many tools are available that ensure that test cases map to requirements, such as traceability matrixes. In the event that all test cases are successfully executed, then the code fulfils the requirement.

Conclusion

QAs should add manual testing to any test strategy as it enables them to gain deeper insights from the end user's perspective that can be extremely useful. As manual testing relies on human testers without the aid of test automation frameworks, it is a powerful tool for judging software based on the most important metric: customer/user experience. The agile software development process continuously demands a shift towards automated testing, but manual testing will never cease to exist. An experienced, well-rounded candidate that can provide both manual and automation testing skills can assist QAs in quickly and efficiently completing the necessary tests. When you prepare for the manual interview, you are more likely to impress the hiring manager and progress to the next stage of the process.

In light of this, we have put together a comprehensive list of interview questions frequently asked during manual testing interviews. Here we have given an overview of manual testing and compiled a list of the top 50+ manual testing interview questions for job seekers at all stages of their careers. The candidate should have a thorough understanding of key concepts, as well as the ability to clearly and persuasively present their ideas. So, prepare yourself accordingly. Good luck with your future endeavours.

Additional Interview Resources

- https://www.interviewbit.com/automation-testing-interview-questions/
- https://www.interviewbit.com/qa-interview-questions/
- https://www.interviewbit.com/blog/types-of-software-testing/
- https://www.interviewbit.com/technical-interview-questions/

Links to More Interview Questions

C Interview Questions	Php Interview Questions	C Sharp Interview Questions
Web Api Interview Questions	Hibernate Interview Questions	Node Js Interview Questions
Cpp Interview Questions	Oops Interview Questions	Devops Interview Questions
Machine Learning Interview Questions	Docker Interview Questions	Mysql Interview Questions
Css Interview Questions	Laravel Interview Questions	Asp Net Interview Questions
Django Interview Questions	Dot Net Interview Questions	Kubernetes Interview Questions
Operating System Interview Questions	React Native Interview Questions	Aws Interview Questions
Git Interview Questions	Java 8 Interview Questions	Mongodb Interview Questions
Dbms Interview Questions	Spring Boot Interview Questions	Power Bi Interview Questions
Pl Sql Interview Questions	Tableau Interview Questions	Linux Interview Questions
Ansible Interview Questions	Java Interview Questions	Jenkins Interview Questions