KASPER
ANALYTICS

kasperanalytics.com

9123820085

Top Manual Testing Interview Questions and Answers

hr@kasperanalytics.com

kasper-analytics

Preparing for an interview can be overwhelming, especially when the pressure is on to land that dream job. That's why we've taken the time to put together a complete list of questions and answers that cover a wide range of topics related to manual testing. Also, we've included valuable insights from industry experts on what hiring managers are seeking in a candidate. Whether you're getting ready for your first or fifth interview, our list of questions and answers will give you the confidence you need to succeed.

Table Of Contents

24. Is there any difference between retesting and regression testing?
25. As per your understanding, list down the key challenges of software testing.
26. What is the difference between system testing and integration testing?
27. Explain the defect life cycle.
28. How will you determine when to stop testing?
29. What if the software is so buggy that it can't be tested?
30. How do you test a product if the requirements are frozen?
31. What if an organization grows so fast that fixed testing processes are impossible? What to do in such situations?
32. How do you know the code has met specifications?
33. What is Test Harness and Test Closure?
34. What is 'configuration management'?
35. Is it true that we can do system testing at any stage?
36. What best practices should you follow when writing test cases?
37. Why does the boundary value analysis provide good test cases?
38. Can automation testing replace manual testing?
39. What are the benefits of test reports?
40. What does defect removal efficiency mean in software testing?

Conclusion

Manual Testing Questions For Freshers

Presented below are several necessary interview questions and corresponding answers designed specifically for entry-level candidates.

1. Explain what software testing is.

Software testing systematically identifies defects, errors, or differences between expected and actual results.

The primary goal of software testing is to uncover bugs or issues before the software is released to end users.

2. What is quality control, and how does it differ from quality assurance?

Quality Control	**Quality Assurance**
Focuses on identifying defects and errors in the final product or service.	Focuses on preventing defects and errors from occurring in the first place.
Performed after production to ensure that the final product meets specified quality standards.	Performed throughout the production process to ensure quality standards are met at every stage.
Inspecting and testing the final product or service to meet specified requirements.	Involves planning, designing, and implementing quality management systems to ensure that products or services consistently meet customer expectations.

3. Why is Software Testing Required?

Software testing is essential because it helps identify and rectify software application defects or errors before deployment. It ensures the software meets the desired quality standards, functionality, and performance expectations.

4. What are the advantages of manual testing?

1. Manual testing allows for human intuition and creativity, enabling testers to uncover unexpected bugs and usability issues.
2. It provides real-time feedback on the user experience, allowing testers to identify and address any issues immediately.
3. Manual testing allows for ad hoc testing, where testers can explore scenarios and test cases that may not have been considered during the initial planning phase.
4. It is cost-effective for small-scale projects or when automated testing tools are unavailable or feasible.
5. Manual testing enables testers to validate visual elements, such as UI design and layout, ensuring a visually appealing and user-friendly product.

5. What are the drawbacks to manual testing?

1. Time-consuming: Manual testing requires human effort and can be time-consuming, especially for large and complex software systems.
2. Limited coverage: Manual testing may only cover some possible scenarios and edge cases, leading to potential bugs or issues being missed.
3. Subjectivity: Test results can vary based on the tester's skills, experience, and biases, making it difficult to ensure consistent and objective evaluations.
4. Costly: Hiring and training manual testers can be expensive, especially compared to automated testing solutions offering higher efficiency and scalability.
5. Repetitive tasks: Manual testing often involves repetitive tasks, leading to human errors due to fatigue or lack of attention.

6. What kind of skills are needed for someone to become a software tester?

1. Strong analytical skills: Software testers need to be able to analyze complex systems and identify potential issues or bugs.
2. Attention to detail: They must have a keen eye for detail to spot even the smallest defects in software.

3. Problem-solving abilities: Testers should possess problem-solving skills to troubleshoot issues and find effective solutions.
4. Good communication skills: Effective communication is crucial for testers to report bugs accurately and collaborate with developers and other team members.
5. Technical knowledge: A basic understanding of programming languages, software development processes, and testing tools is essential for software testers.
6. Patience and perseverance: Testing can be repetitive and time-consuming, so testers need patience and perseverance to ensure thorough testing.
7. Adaptability: Testers should be adaptable to work with different software applications, platforms, and technologies as per project requirements.

7. Explain what SDLC is.

SDLC, or Software Development Life Cycle, is a systematic approach to developing software applications. It consists of phases that guide the entire software development process, from initial planning and requirements gathering to coding, testing, deployment, and maintenance.

8. What is a test case?

A test case refers to testers' instructions to validate a particular functionality or software application feature. It is a detailed description of a test's inputs, operating conditions, and expected outputs.

9. What is a test scenario?

A test scenario is defined as a real-world scenario in which software is expected to be utilized and is supposed to be tested. For instance, most applications and websites require users to authenticate their identity by logging in and out. Logging in and out is a real-world user scenario that must be tested.

10. What is a test plan?

A test plan is a document that outlines the strategy, objectives, resources, and schedule of a software testing process. The test plan will typically include details such as the type and number of tests that need to be conducted, the purpose of each test, the required tools, and how test results will be analyzed and reported. It is regularly updated throughout the testing process to reflect any discoveries or changes in strategy.

11. What is test data?

Test data is a collection of input and output values used to test a software application's functionality and performance. It is designed to simulate real-world scenarios and help identify any defects or errors in the system.

12. What is a test script?

A test script is a detailed set of instructions or steps testers follow to execute a test case. It outlines the actions to be performed, the expected results, and any preconditions or prerequisites required for the test.

In Automation, when you take the steps from a test case and automate them via some tool like Selenium , usually a test automation tool, it becomes a test script.

13. What types of manual testing are there? Break them down.

1. Functional Testing
2. Regression Testing
3. Smoke Testing
4. Integration Testing
5. System Testing
6. User Acceptance Testing (UAT)
7. Compatibility Testing
8. Performance Testing

9. Security Testing
10. Usability Testing
11. Exploratory Testing
12. Ad-hoc Testing
13. Localization/Internationalization Testing
14. Accessibility Testing
15. Installation/Configuration Testing

14. What is black box testing, and what are the various techniques?

Black-Box Testing is a software testing method that aims to determine whether software functions as intended for end-users without delving into the internal system. During this process, a tester observes the system's behavior solely through inputs and outputs. Testing techniques include equivalence partitioning, boundary value analysis, decision tables, state transition testing, and error guessing.

15. What is white box testing and its various techniques?

White-Box Testing tests the internal structure or working of an application. It is performed at a lower level of testing, for instance, unit testing by developers or testers having coding knowledge. Testing techniques include conditional testing, loop testing, control flow testing, data flow testing, branch testing, etc.

Manual Testing Interview Questions for Experienced

Hey, the following questions and answers are for those of you who have experience with manual testing.

16. Explain the Test Driver and Test Stub.

A test driver is a program or module used in software testing to simulate the environment in which a component or system will run. It provides inputs to the software under test and captures the outputs or behavior for verification. The test driver helps execute tests and ensure the software performs as expected.

On the other hand, a test stub is a small piece of code or a dummy module that mimics the behavior of an external component or module. It is used in integration testing to replace the component's functionality that is not yet available or ready for testing. The test stub provides predefined outputs to the tested component, allowing other software parts to be tested independently.

17. What will you do when a bug turns up during testing?

When a bug occurs during testing, the first step is documenting and reporting it in a bug¬tracking system. Then, try to reproduce the bug by following the steps that led to its discovery. Once reproduced, you should analyze the bug's impact on the system and prioritize it based on severity. At last, you can collaborate with developers and stakeholders to resolve the issue, ensuring proper testing and verification before closing it.

18. Why is it impossible to test a program thoroughly?

There always has to be a balance between time and cost.

This means ensuring that the best possible quality is released. But because we cannot spend infinite amount of time on testing, there will always be some bugs, that will not be discovered. Thus testing a program thoroughly is impossible.

19. What is the difference between manual testing and automation testing?

Manual Testing	Automation Testing
Testing is performed manually by human testers.	Testing uses automated tools(selenium, etc) and scripts.
Requires human intervention and effort for each test case execution.	Does not require human intervention once the scrip are set up.

20. When should you opt for manual testing over automation testing?

Manual testing should be opted over automation testing in specific scenarios:

When the application or system being tested is in its early development stages and constantly evolves, manual testing allows for flexibility and adaptability.

Manual testing ensures accurate results when the test cases are complex and require human intuition and judgment.

Manual testing is more cost-effective when developing and maintaining automated tests outweighs the benefits.

21. What are the phases involved in Software Testing Life Cycle?

The Software Testing Life Cycle (STLC) consists of several phases to ensure a software product's quality and reliability. These phases include:

1. Requirement Analysis: In this phase, the testing team analyzes the requirements provided by stakeholders to understand the software's functionalities, features, and expected behavior.
2. Test Planning: A detailed test plan is created once the requirements are understood. This plan defines test objectives, scope, strategies, resource allocation, and timelines.
3. Test Case Development: Test cases are designed based on the requirements and test objectives defined in the previous phases. These test cases outline step-by-step instructions to validate different aspects of the software.

4. Test Environment Setup: A suitable testing environment is set up with all necessary hardware, software, and network configurations to execute tests effectively.
5. Test Execution: The actual testing takes place in this phase. The test cases developed earlier are executed systematically to identify defects or deviations from expected behavior.
6. Test Cycle Closure: After the test execution phase, the test cycle closure involves analyzing the test results, documenting lessons learned, and preparing a final report to evaluate the overall testing process.

22. How will you overcome the challenges faced due to the unavailability of proper documentation for testing?

To overcome the challenges posed by the unavailability of proper documentation for testing, you should adopt a proactive approach. Firstly, you can communicate with the stakeholders to gather as much information as possible about the system or application under test. This could include talking to developers, business analysts, or product owners to understand the requirements and functionalities.

Also, you should leverage exploratory testing techniques to uncover hidden features or potential issues. You can compensate for the lack of documentation and ensure complete test coverage by combining effective communication and exploratory testing.

23. Is there any difference between retesting and regression testing?

Features	**Retesting Testing**	**Regression Testing**
Purpose	To verify that a specific defect or issue has been fixed and no longer exists in the software.	To ensure that changes or modifications made to the software have yet to introduce new defects in existing functionalities.

Scope	It focuses on testing the specific functionality or area where the defect was found.	Covers a broader scope, including testing the related functionalities that the changes could impact.
Execution Time	Usually performed after fixing a defect during the same development cycle.	Typically performed during subsequent development cycles, after implementing new features or making significant changes to the software.
Automation	Retesting can not be automated.	Regression testing can be done either via automated or manual testing.
Test Cases	Focuses on retesting only those test cases previously executed that identified the defect.	Involves executing existing test cases to ensure overall system stability and functionality.

24. As per your understanding, list down the key challenges of software testing.

1. Time and resource constraints: Testing must often be completed within tight deadlines and limited resources. This can result in insufficient time for thorough testing or a lack of access to necessary tools, environments, or skilled testers. Limited resources may also lead to difficulty in setting realistic test environments that mimic real-world conditions.
2. Changing requirements and frequent updates: Software development is an iterative process, with requirements evolving. This challenges testers to adapt their test plans and strategies accordingly. Regular updates or changes in requirements can lead to rework, impacting the overall testing timeline and effort.

3. Complex integration and compatibility issues: Testers must carefully navigate complex integration and compatibility issues to ensure the software functions seamlessly across different platforms and systems.

25. What is the difference between system testing and integration testing?

System Testing	Integration Testing
System testing focuses on testing the whole system, including all components and interactions between them.	Integration Testing focuses only on testing the interaction between different modules or component of the system.
Tests are performed after integration testing to ensure the system functions correctly.	Tests are performed after unit testing and before system testing to ensure that individual components work together properly.
This testing involves end-to-end system testing to validate its compliance with functional and non- functional requirements.	This testing involves testing the interfaces and data flow between different modules or subsystems of th system.
System testing is conducted by independent testers not involved in development activities.	Integration Testing is often conducted by developers and testers working to identify and resolve integratio issues.

26. Explain the defect life cycle.

The defect life cycle in software testing refers to the stages a defect goes through, from discovery to resolution.

The cycle typically begins with identifying the defect, followed by its documentation and reporting. The defect is then assigned to the appropriate team member for analysis and reproduction.

Once the defect is confirmed, it is prioritized and scheduled for resolution. After the defect is fixed, it undergoes testing to ensure it has been resolved completely.

At last, the defect is closed, and its resolution is documented for future reference.

27. How will you determine when to stop testing?

Determining when to stop testing is an important decision that depends on various factors.

Firstly, the testing objectives and criteria defined at the beginning of the process should be considered. If all the objectives have been met and the predefined criteria have been satisfied, it may be an indication to stop testing.

Resource constraints such as time and budget can influence the decision. If these constraints are exceeded without significant defects being found, it might be reasonable to conclude that further testing is unnecessary.

Lastly, feedback from stakeholders and end-users can provide valuable insights on whether the system meets their expectations, helping them decide when to stop testing.

28. What if the software is so buggy that it can’t be tested?

When software is so buggy that it can’t be tested, it can create significant challenges for developers and end-users. The consequences of releasing buggy software can be detrimental to a company’s reputation, leading to lost revenue and a loss of customer trust. In such instances, it is crucial to identify the root cause of the problem and implement a structured approach to testing and debugging the code. Please do so to avoid further delays, missed deadlines, and a subpar product. Organizations must invest in quality assurance measures to ensure their software is thoroughly tested and meets the required standards.

29. How do you test a product if the requirements are frozen?

If a product's requirements are frozen, they cannot be changed or modified. In such a scenario, testing becomes essential to ensure the product meets all the specified requirements. The testing process involves verifying if the product functions as intended and meets the predetermined bars. This can be done through various testing techniques such as functional testing, regression testing, performance testing, and user acceptance testing. By thoroughly evaluating the product against the frozen requirements, any defects or deviations can be identified and addressed before its release.

30. What if an organization grows so fast that fixed testing processes are impossible? What to do in such situations?

When a company starts booming, its standard testing procedures might not work anymore because things get bigger and more complicated. That's when it's important to switch over to agile testing methods that are flexible and adaptable. Automate repetitive tasks to save time and be more efficient. Focus on risky areas with risk-based testing and allocate resources smartly. And last but not least, get some skilled testers on your team who can keep up with new technology and changing requirements!

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

High-quality code is depicted by its functionality, lack of bugs, readability, and maintainability. Many companies have "coding standards" that developers must adhere to ensure consistency among the entire development team. To further ensure code quality, you can use a traceability matrix (a document that maps and traces the client requirement with test cases). If every test case execution is successful, then it means that the code meets the requirements.

32. What is Test Harness and Test Closure?

A test harness is a collection of software and test data to automate the testing process. It provides a framework for executing tests, capturing results, and generating reports. A test harness helps in reducing manual effort and ensures consistency in testing.

Test closure refers to the activities performed at the end of the testing phase. It involves documenting the test results, analyzing them, and preparing a final report. Test closure also includes gathering feedback from stakeholders, identifying lessons learned, and updating the test assets for future reference.

33. What is 'configuration management'?

Configuration management tracks changes to a system's software, hardware, or network settings. Its purpose is to maintain the system's integrity and ensure it complies with organizational policies while remaining secure and stable.

34. Is it true that we can do system testing at any stage?

That is not true. We can not do system testing at any stage. System testing is typically performed after the completion of unit and integration testing. It involves testing the entire system to ensure all components work together correctly. Conducting system testing too early may result in complete or inaccurate results, as individual components may need to be fully developed or integrated.

35. What best practices should you follow when writing test cases?

1. Clearly define the objective: Before writing test cases, it is essential to have a clear understanding of the objective or purpose of the test.
2. Keep it simple and concise: Test cases should be written in a simple and easy-to- understand manner.
3. Cover all possible scenarios: Test cases should cover many positive, negative, boundary, and edge cases.

4. Use standardized templates: Using standardized templates for writing test cases improves consistency across different testers and projects.
5. Review and validate: This ensures that test cases are accurate, reliable, and aligned with the project requirements, ultimately enhancing the overall quality of the testing process.

36. Why does the boundary value analysis provide good test cases?

Boundary value analysis is a widely used technique in software testing that aims to identify defects in the system by testing the boundaries of input values. The method is based on the assumption that errors are more likely to occur at the extremes of input ranges. Boundary value analysis provides a systematic approach to testing, which is essential for large-scale software development projects.

By identifying the maximum and minimum input values, it is possible to generate test cases that adequately cover the boundaries and are a reasonable number. This approach provides good test cases because it maximizes the chances of finding defects and ensures the software is tested under realistic conditions.

37. Can automation testing replace manual testing?

Automation testing can complement manual testing but cannot completely replace it. While automation testing offers speed, efficiency, and repeatability, it needs more human intuition and creativity to identify complex issues. Manual testing allows testers to explore different scenarios, uncover edge cases, and assess the user experience. Therefore, a combination of both approaches is ideal for comprehensive software testing.

39. What are the benefits of test reports?

1. Identification of defects: Test reports provide a detailed overview of the defects or issues found during testing, allowing for their identification and subsequent resolution before the product is released to the market.

2. Quality assurance: Test reports help ensure the product meets the required quality standards by providing evidence of thorough testing and validation, instilling confidence in stakeholders and customers.
3. Documentation and traceability: Test reports are complete documentation of the testing process, including test cases, test results, and any deviations encountered. This facilitates traceability and helps in future references or audits.
4. Decision-making support: Test reports provide valuable insights into the performance and stability of the product, enabling informed decision-making regarding its readiness for release or further improvements needed.
5. Continuous improvement: By analyzing test reports over time, patterns can be identified, allowing for continuous improvement in testing processes, methodologies, and overall product quality.

40. What does defect removal efficiency mean in software testing?

Defect removal efficiency (DRE) is a metric used in software testing to measure the effectiveness of defect identification and removal processes. It represents the percentage of defects identified and fixed during testing compared to the total number of flaws reported in the software. A higher DRE indicates a more efficient testing process, implying that a more significant proportion of defects have been successfully eliminated. This metric helps organizations assess the quality and reliability of their software by providing insights into the thoroughness and effectiveness of their testing efforts.