**Software Testing Introduction**

The History of Software Testing

**Q:Which two fundamental ideas of testing were born in the 1970s?**

Regression testing and integration testing

Positive testing and negative testing

Testing for bugs and testing for performance

Usability testing and acceptance testing

**Q:Which of the following options are positive test scenarios?**

Check that the software provides appropriate feedback and error messages to users in case of incorrect input or other errors while also validating that it handles both common and unusual error conditions gracefully.

Verify that the software performs the expected operation for valid input data within the time limits specified in the requirements.

Test that the software provides correct results for all specified use cases while also ensuring that it works as intended under both standard and boundary (but within the requirements) operating conditions.

Verify that the search feature returns accurate results based on the entered query.

Test that the software handles invalid input data correctly (e.g., empty fields, incorrect types of data, etc.) while also checking that it produces the expected output for different types of valid input data.

Test that the messaging system delivers messages to the intended recipients without delay.

**Q:Which of the following options are negative test scenarios?**

Confirm that the system works as intended under normal operating conditions while also validating that input data is accepted and processed correctly.

Validate that the software rejects input data that exceeds specified limits or constraints while also verifying that it responds correctly to unexpected or erroneous data.

Check that the software provides appropriate feedback and error messages to users in case of incorrect input or other errors while also validating that it handles both common and unusual error conditions gracefully.

Test the system to ensure that it meets all non-functional requirements such as performance, security, and usability while also checking that it responds appropriately to unexpected or erroneous input data.

Test that the software handles invalid input data correctly (e.g., empty fields, incorrect types of data, etc.) while also checking that it produces the expected output for different types of valid input data.

Verify that the software can handle unusual or edge cases without crashing or producing incorrect results while also meeting all specified use cases and functional requirements.

Test that the software responds appropriately to unexpected or erroneous input data while also checking that it handles common and expected error conditions correctly.

Verify that all functional requirements have been met while also testing that the software handles unexpected error conditions gracefully*.*

**Q:Which of the following software bugs is NOT a real bug that caused a significant issue in history?**

The Therac-25 Bug

The Heartbleed Bug

The Y2K Bug

The Morris Worm

**Q:What is the purpose of testing?**

To ensure that a program will behave correctly under all circumstances and will meet defined requirements

To ensure that a program is free from bugs and performance issues

To ensure that a program performs its intended tasks whatever user does

To debug any errors in a program and requirements

**Q:What is the main difference between exhaustive testing and debugging?**

Exhaustive testing involves detecting and eliminating errors in software code, while debugging involves exploring software in individual situations and states.

Exhaustive testing involves checking all possible ways of code execution with all possible input data, while debugging involves detecting and eliminating errors in software code.

Exhaustive testing and debugging are the same thing.

Exhaustive testing involves testing software in all possible situations and states, while debugging involves figuring out what the computer is actually doing.

**Q:According to software testing theory, which of the following are reasons for errors in software development?**

Inexperienced project participants or new technologies

Complexity and misunderstandings

Lack of resources and inefficient project management

Time pressures and human fallibility

Miscommunication and unclear project goals

Inadequate testing and unreliable software tools

**Q:Which testing methodology is best suited for projects with changing requirements?**

Waterfall

V-Model

Agile

Spiral

**What was the primary reason for the emergence of software testing?**

The need for specialized testing roles

The development of systematic testing methodologies

The increasing complexity of software systems

The evolution of computers and software

**Q:What are the consequences of software errors?**

Decreased user satisfaction, lower users' productivity, and bad feedbacks in the market

Financial losses, legal liabilities, and safety threats

Increased software complexity, excessively high levels of security, and accessibility

**Q:Which of the following is NOT a stage in the SDLC?**

Design

Problem-finding

System analysis

Assurance (or testing)

**Q:Which model is suitable for a project requiring a shorter development time?**

Waterfall model

Iterative model

Incremental model

RAD model

**Q:What is the primary objective of software testing in the SDLC?**

To ensure that software meets all the functional requirements specified by the client

To speed up the software development process and reduce the overall cost

To ensure that software is visually appealing

To identify and eliminate all possible defects or errors in software

**Q:Which of the following is not a phase of the STLC model?**

Test environment setup

Test execution

Test planning

Test feedback and improvement

**Q:Which of the following statements accurately describes entry and exit criteria in the STLC?**

*Entry criteria* refers to the results that must be achieved before starting a test phase, while *exit criteria*refers to the test conditions that must be satisfied before exiting a test phase.

*Entry criteria* refers to the prerequisites for starting a test phase, while *exit criteria*refers to the deliverables that must be completed before exiting a test phase.

*Entry criteria* refers to the deliverables that must be completed before exiting a test phase, while *exit criteria* refer to the prerequisites for starting a test phase.

*Entry criteria* refers to the test conditions that must be satisfied before starting a test phase, while *exit criteria* refers to the results that must be achieved before exiting a test phase.

**Q:Which of the following is not a common challenge faced by the QA team during the SDLC?**

Ensuring that the software is bug-free and error-free

Managing testing schedules and timelines

Ensuring that the software meets user requirements

Providing training and support to end users

**According to the ISTQB Glossary, what is the difference between a "test case" and a "test script?"**

According to the ISTQB Glossary, the terms *test case* and *test script* are interchangeable.

A test case is a set of preconditions, inputs, actions, and expected outcomes that are designed to verify one or more test conditions, while a test script is a document that outlines the steps and data required to perform a specific test.

A test case is a specific set of conditions and inputs that are used to test a particular aspect of a system, while a test script is a program that simulates user actions and system responses to automate testing.

A test case is a formal document that outlines the steps, inputs, and expected results for a specific test scenario, while a test script is a set of instructions the tester follows to execute a test case.

**Which stage of the software testing process involves reporting defects?**

Analyzing and reporting test results

Executing test cases

Creating test cases

**Which stage of software testing involves defining the right approaches, tools, schedules, roles, responsibilities, etc., with the ultimate goal of finding the easiest way to satisfy the customer's requirements?**

Executing test cases and reporting defects

General planning and requirements analysis

Establishing a test strategy

Analyzing and reporting test results

Creating test cases

Establishing acceptance criteria

**The pesticide paradox refers to:**

The idea that the same repeated tests become ineffective in detecting new defects.

The notion that finding and fixing a large number of defects will ensure the success of a system.

The expectation that testers should find all possible defects in a system.

The idea that testing is always context-dependent and must be done differently in different contexts.

**Which of the following types of testing is NOT typically associated with software testing?**

Delta testing

Beta testing

Alpha testing

Gamma testing

**Which of the following statements about dynamic testing are NOT true?**

Dynamic testing is always less effective than static testing for identifying defects in software code.

Dynamic testing is only useful for testing software that has been developed using an Agile software development methodology.

Dynamic testing is a software testing technique that involves executing code and observing its behavior.

Dynamic testing can be performed manually or using automated testing tools.

Dynamic testing is often performed after static testing has been completed.

**Which of the following statements are true for both static and dynamic testing?**

It can be used to identify defects in software applications.

It is very effective at finding certain types of defects, such as violations of coding standards.

Testers do not need knowledge of the internal workings of software code to apply it.

The goal is to ensure that a software application works as expected and meets the specified requirements.

Software must be executed.

**Why is static testing less expensive than dynamic?**

It does not require any human intervention and can be fully automated.

It does not require any specific knowledge, and anyone can conduct it.

It does not require any specialized tools or technologies to perform.

It can be performed earlier in the software development lifecycle, which helps identify defects at an early stage.

**Why are acceptance criteria important?**

They provide a basis for mutual understanding between teams of BAs, testers, and developers.

They provide conditions with a clear pass/fail result that the software must satisfy to be accepted by a user, customer, or consuming system.

Acceptance criteria are primarily used to help project teams stay on schedule and within budget.

They articulate how much value a certain job will deliver to the customer.

**Which of the following requirements are formulated well?**

The database should be backed up every hour.

As an admin, I want to add descriptions to products so that users can view these descriptions and compare the products later.

Each request should be processed in approximately 10 seconds.

As a user, I want to choose folders for backup so that they will be copied to backup.

The system cancels an order automatically if it is not paid for within 24 hours of being created.

**Which of the following statements about verification are true?**

It involves checking whether software conforms to specifications.

It always involves executing code.

It checks whether software meets customer requirements and expectations.

It uses methods like reviews, walk-throughs, inspections, and desk checking.

It finds bugs early in the development cycle.

It does not involve executing code.

It includes checking documents, design, code, and programs.

**Which of the following testing methods is most appropriate for identifying defects related to software's internal structure, such as a memory leak or a race condition?**

Grey box testing

Black box testing

White box testing

Which of the following statements about grey box testing is incorrect?

It is also known as translucent testing.

The tester has access to both the source code and the functional specifications of software.

Grey box testing is only used for testing web-based applications.

Test cases are designed based on knowledge of the inner workings of software.

Correct

**Which of the following statements about user acceptance testing (UAT) are true?**

UAT is an obligatory formal part of the software testing process.

UAT is always performed by the end users or clients.

UAT is performed after system testing and before release.

UAT is always performed by the development team.

**Which of the following statements about validation are true?**

It always involves executing code.

It uses methods like black box, white box, and non-functional testing.

It finds bugs early in the development cycle.

It is a dynamic mechanism of testing and validating the actual product.

It checks whether software meets customer requirements and expectations.

It checks whether software conforms to a specification.

**Which of the following statements are true?**

A testing pyramid is a rigid framework that must be followed exactly, with no room for customization or flexibility.

The goal of a testing pyramid is to focus testing efforts on the top of the pyramid by writing more end-to-end tests and fewer unit tests in order to increase the coverage of the testing process.

A testing pyramid can help teams balance the trade-offs between the speed, cost, and coverage of testing efforts by focusing on the most critical and high-risk parts of a system first.

A testing pyramid is comprised of three layers: unit tests, integration tests, and end-to-end tests.

**In which of the following cases would the ice cream cone testing pyramid apply?  
\* MVP is an acronym for minimum viable product, a concept from the lean startup methodology, which focuses on validating a product idea with customers early in the development lifecycle.**

Legacy codebases with no unit tests

Any application if there is a shortage of developers but a large team of QAs

MVP\* projects with tight deadlines

The ideal testing pyramid is only relevant for Agile projects, so any Waterfall project can use the ice cream cone pyramid.

Applications with a heavy focus on user interface (UI) interactions, such as gaming or multimedia applications

It depends on the customer's requirements.

**In which of the following cases would the testing trophy apply?**

Any project whose back end is implemented on an API because it's easy to automate API testing

A project with a high degree of user interaction because end-to-end tests can help ensure the user experience is smooth and error-free

A project with complex integration points between multiple systems

**Which of the following statements about alpha testing are correct?**

It can use any testing method.

It is run on a "prod-like" environment but with obfuscated data.

It reduces the risk of product failure via customer validation.

It is performed by end users.

It is performed by the application team.

It creates goodwill with customers and increases customer satisfaction.

A long execution cycle may be required.

**Which of the following statements about beta testing are correct?**

It mainly uses black box testing.

It is performed by end users.

It is run on a "prod-like" environment but with obfuscated data.

It creates goodwill with customers and increases customer satisfaction.

It reduces the risk of product failure via customer validation.

It can use any testing method.

A long execution cycle may be required.

**Which of the following test automation tools is best suited for mobile application testing?**

HP UFT

Selenium

TestComplete

Appium

**Which of the following test tools provides record and playback capabilities for Windows applications and uses a custom scripting language?**

TestComplete

WinAutomation

Eggplant Functional

AutoIt

**Which of the following tools is used to simulate real-life load conditions on a web application?**

TestComplete

SoapUI

LoadRunner

Apache JMeter

**Which of the following is NOT a benefit of smoke testing?**

It helps improve the quality of a software product.

It helps identify defects early in the testing process.

It saves time and effort by focusing on critical functionalities.

It ensures that all functionalities of a system are thoroughly tested.

**Which of the following is a benefit of extended testing?**

It is primarily used to test a system's usability.

It ensures that only critical functions are tested.

It reduces the time and effort required for testing.

It checks nonstandard uses of a software product.

**What is the difference between negative testing and positive testing?**

There is no difference between negative testing and positive testing.

Negative testing only tests the happy path of a system, while positive testing tests a system with invalid inputs.

Negative testing is more time-consuming than positive testing.

Negative testing tests a system with invalid or unexpected inputs, while positive testing tests it with valid inputs.

**Imagine you are a software tester for a travel website. Your team has just completed a round of non-functional testing for the website's new mobile application, which includes compatibility testing, scalability testing, and usability testing. The development team has addressed all the issues found in testing, and the application has been deployed to the app store. A week later, users report that the application frequently crashes, especially when booking flights. What type of non-functional testing might the team have missed during the initial round of testing?**

Reliability testing

Compatibility testing

Usability testing

Performance testing

**When developing a test plan, which of the following factors should be taken into consideration?**

The availability of testing tools

The regulatory and compliance requirements that must be met

The cost of testing

The size of the development team

The geographic location of the development team

The experience level of the testing team

**When creating a test plan template, which of the following sections should be included?**

A list of team members' favorite movies

Marketing strategy

A detailed list of project requirements

Company history

Risk assessment

Test objective

**Which of the following sections should be included in a sample test plan document?**

Project timeline

Test environment configuration

Project budget

Test approach

Employee performance evaluations

Company marketing strategy

**Which of the following is a key difference between the testing schedule section and the test documentation and deliverables section of a test plan document?**

The testing schedule section is typically shorter than the test documentation and deliverables section.

The testing schedule section outlines the timeline for testing activities, while the test documentation and deliverables section focuses on the specific documents that will be produced during testing.

The testing schedule section identifies the resources needed for testing, while the test documentation and deliverables section outlines the types of tests that will be performed.

The testing schedule section includes metrics for measuring test progress, while the test documentation and deliverables section does not.

**Which of the following product risks can be covered by usability testing?**

Compatibility issues

Performance issues

Security vulnerabilities

Inconsistent user experience

**You need to create a checklist for an electronic shopping cart. What approach will you use?**

Neither for a feature nor for a requirement

Both for a feature and for a requirement

For a requirement

For a feature

**Which of the following attributes are NOT included in a checklist?**

Expected result

A link to the requirement

Preconditions

Pass/Fail status

Summary

Steps

**Which of the following are good checklist items (i.e., atomic and testable)?**

Make sure free delivery and a unique postcard are included in the order when the order total exceeds $100

Verify that the user is redirected to the home page after successful login

Verify that the password field is obscured on the "Login" page

Test that the search bar is blue

Verify login button functionality

Verify that the search bar is visible on the webpage

**What is checklist-based testing?**

A method of software testing that is not based on the testing engineer's experience and doesn't require a checklist

A method of software testing where checklists are used to create test cases

A method of software testing that is based on the testing engineer's experience and requires test cases to be created based on checklists

A software testing method based on the testing engineer's experience that requires a checklist to be created as the tests progress

**In what projects do testers create checklists, not test cases, to validate the product?**

Projects with stable and experienced test teams

Long-term projects

Small, uncomplicated projects

Short-term projects

Any project if the team agreed on the approach

Projects with detailed requirements

**You need to create a checklist for calculating the shipping price based on the total sum of an order. What approach will you use?**

Neither for a feature nor for a requirement

Both for a feature and for a requirement

For a feature

For a requirement

**Which of the following are white-box test techniques?**

State transition testing

Statement testing and coverage

Error guessing

Decision testing and coverage

Use case testing

Technical review

**What are the purposes of white-box test techniques?**

To identify all the defects in an application

They provide more ideas for testing than black-box and experienced-based techniques

To verify all potential scenarios and use cases for a specific feature

To find defects earlier

They provide much more test coverage than black box testing alone

**Which of the following is a description of statement coverage?**

A metric that gives a true/false confirmation of whether all statements are covered or not.

A metric that shows the percentage of test cases that have been executed.

A metric that shows the percentage of statements in the source code that have been executed.

A metric that shows the number of statements in the source code that have been executed by test cases that have passed.

**In which of the following ways do testers apply the equivalence partitioning test technique?**

Make a test that is equal in priority to test one feature

Divide test data into classes that must be tested in the same way

Create test data that is equivalent to the data used by end users

Divide test data into classes that are expected to produce the same output

**In which of the following ways do testers apply the boundary value analysis test technique?**

Analyze data that binds (integrates) two different features together

Create tests that include a one–two representative of each equivalence class

Analyze numeric or sequential data on the "edges" of each equivalence class

Create test data that allows different tests to be bound into one suite

**Which of the following attributes is used to describe the expected outcome of a test case?**

Test steps

Test data

Test case priority

Expected result

**According to the best practices for writing good test cases, what is the difference between a test case and a test suite?**

A test case and a test suite are interchangeable terms and can be used to describe the same thing.

A test case is a high-level description of a testing procedure, while a test suite is detailed step-by-step instructions.

A test case is a manual testing method, while a test suite is an automated testing method.

A test case is a specific condition to be tested, while a test suite is a set of test cases that checks a larger system feature.

**Which of the following features is NOT typically offered by test management tools?**

Source code management

Test planning and scheduling

Test case management

Defect tracking

**Why does a tester use test management tools?**

To automate the testing process

To manage and organize test cases

To conduct peer reviews

To spend more time on requirements analysis

To control the performance of each tester

To generate test reports and metrics

**Which THREE of the following options are NOT attributes of a test case?**

Steps

Preconditions

Postconditions

Hardware, required for the execution

Status

Cleanup procedures

Summary

Metrics

Expected results

**What does it mean that test cases should be atomic?**

Test cases should be run in a vacuum to avoid interference from other test cases.

Test cases should be written in a language that uses atoms, such as Lisp or Prolog.

Test cases should cover only one functionality or feature.

**What is the most essential information that should be included in a bug report?**

A list of potential solutions

A detailed description of the bug

The name of the person who found the bug

The date and time the bug was discovered

**What is the best way to find valid defects in an application?**

Only test the features and functions that are specified in the requirements document.

Start testing the happy path testing scenarios, and then proceed to error path testing.

Test every single feature thoroughly to uncover as many defects as possible.

Assume the developers have found and fixed all defects and only test for functionality.

**Which of the following is NOT a key attribute of a good defect report?**

Summary

Pass/fail status

Expected result

Steps to reproduce

Severity level

**Which of the following is a key aspect of a good defect report?**

Humor and opinions

Additional information (personal opinions, biases)

Clarity and grammar

Error messages and system logs

**What is the purpose of a defect report?**

To blame bad requirements

To expose problems in communication between the test and dev teams

To show the importance of testing

To guide software engineers to what needs to be fixed in the least amount of time

**Which SIX of the following best practices do testers use to find defects?**

Pay special attention to regression testing and retesting

Ensure features are understood correctly and in the same way as the project team

Keep project and test documentation in order

Hold additional meetings with the test team before test execution

Avoid early testing and devote more time to dynamic testing

Use special tools to test an application from every angle

Use as many testing types as possible

Use the benefits of different test techniques

Use appropriate testing types

**Which of the following is NOT a benefit of becoming proficient at using DTSs?**

A decreased need for communication and collaboration with coworkers and other users

The ability to generate reports and data and manage software more effectively

Increased efficiency in logging, tracking, and managing defects

The addition of a valuable skill set to your resume and the potential for career growth

**What happens to a defect that is valid when the client confirms that a fix is not needed?**

It is ignored and left unresolved.

It is removed from the DTS.

It will be fixed automatically in the nearest future release.

It must be moved to the final stage (rejected/canceled).

**Why can a defect be rejected?**

The defect was reported without discussing it with the project manager.

The final decision was not to fix this bug.

There is not enough time to fix this bug during the current sprint.

The defect cannot be reproduced because it was fixed by other changes in the code.

The tester misunderstood the requirements.

**Which of the following is NOT a primary feature of a DTS?**

The ability to track a defect's status and progress through the development process.

The ability to automatically fix defects without developer intervention.

The ability to collaboration with your team and manage tasks.

The ability to generate reports and statistics on defects and their resolution.

A centralized location for storing defect reports and related information.

**Which of the following statements are true?**

A DTS can be used for non-software projects if bugs or issues need to be tracked.

A DTS ensures more efficient use of testing time.

Using spreadsheets to track bugs is cheaper than using a DTS.

A DTS helps reduce the number of bugs in a project.

DTSs are only useful for large projects; they help keep track of a large number of bugs.

**Which of the following is a benefit of using defect-tracking software?**

It eliminates the need for human involvement in the defect-reporting process.

It decreases the likelihood of defects being discovered during testing.

It provides a centralized location for all defect reports and related information.

It increases team productivity by reducing the need for manual testing.

**Which of the following is NOT a benefit of maintaining an accurate and well-documented history of defects in a tracking system?**

Reducing the need for time-consuming meetings among developers and quality testers

Providing transparency and accountability for all teams involved in the project

Increasing the number of defects that get fixed promptly

Helping to identify defects that may have been overlooked during periodic reviews

**Which of the following can be the final stage in the defect lifecycle?**

Reopen

Closed

Declined

Deferred

Open

Ready for demo

**A test result report helps testers to:**

Check if any actions need to be done

See if the test team is moving in the right direction

Predict possible issues and potential risks

Get acquainted with the test strategy the test team is using

See who found most of the defects

See if there are any problems right now

Check information about the daily status of the test team

**Who is responsible for creating a TRR and providing it to all stakeholders?**

Project manager

Scrum master

Each team member

Test lead

Key tester

**Which of the following statements highlight the importance of TRRs?**

The TRR provides the entire project team with test results to understand what was done by the test team and what should be done next.

The TRR is critical in ensuring accountability and transparency.

The TRR helps stakeholders to track the progress of product development and control the resources that are needed.

The TRR serves as a record of the test, providing detailed information on the data collected while testing.

The TRR enables stakeholders to make informed decisions based on concrete evidence.

**Which of the following can be considered TRR stakeholders?**

Project team members

Scrum master

Customer

Test lead

Project manager

Beta testers

**Which three of the following statements about TRRs are true?**

A TRR should be created regularly according to a schedule: weekly, per sprint, for releases that happen every two/three/four months, etc.

To create a TRR, testers use a template approved for a specific project based on customer or other stakeholders' requirements.

A TRR should be created for each sprint to make it possible to analyze the testing process and product quality and manage process or product issues.

A TRR should be clear, detailed, specific, and standard.

A TRR should be created only by the test lead, as other test team members might mispresent the test results to please the customer or hide personal performance problems.

**Which three of the following sections are NOT included in a TRR?**

Critical success factors

Test result summary

Test team responsibilities

Key project resources

Recommendations and suggested actions

Metrics and statistics

Test objectives

**The content of a TRR varies and depends on the software development methodology being used, the type of project and project specifics, the features under test during the reporting period, and the managers' (PM, test lead) and customer's requirements.**

True

False