

ISTQB Agile Tester Practice Test (Sample)

Study Guide



Everything you need from our exam experts!

Copyright © 2025 by Examzify - A Kaluba Technologies Inc. product.

ALL RIGHTS RESERVED.

No part of this book may be reproduced or transferred in any form or by any means, graphic, electronic, or mechanical, including photocopying, recording, web distribution, taping, or by any information storage retrieval system, without the written permission of the author.

Notice: Examzify makes every reasonable effort to obtain from reliable sources accurate, complete, and timely information about this product.

SAMPLE

Questions

SAMPLE

- 1. How is defect density calculated?**
 - A. Defects per feature**
 - B. Defects divided by component size**
 - C. Defects per release**
 - D. Total defects documented**
- 2. What is the outcome of the test execution process?**
 - A. The selection of test cases**
 - B. The preparation of test data**
 - C. The actual results based on executed tests**
 - D. The design of test conditions**
- 3. What term describes a person's perceptions and responses resulting from the use or anticipated use of a software product?**
 - A. User experience**
 - B. User interface**
 - C. User story**
 - D. White-box testing**
- 4. Which testing approach responds dynamically to the behavior of a test object?**
 - A. Reactive testing**
 - B. Proactive testing**
 - C. Static testing**
 - D. Exploratory testing**
- 5. What is the primary focus of 'white-box testing'?**
 - A. Analyzing user satisfaction**
 - B. Assessing external customer interaction**
 - C. Examining internal code structure**
 - D. Verifying requirement specifications**

- 6. What is the name given to a scenario that describes the expected execution of test in structured sequences?**
- A. Test result**
 - B. Test scope**
 - C. Test specification**
 - D. Test session**
- 7. What are attributes of software products that prevent unauthorized access called?**
- A. Integrity**
 - B. Security**
 - C. Scalability**
 - D. Availability**
- 8. What is defined as a set of test cases or test procedures to be executed in a specific test cycle?**
- A. Test design**
 - B. Test suite**
 - C. Test specification**
 - D. Test session**
- 9. What does maintainability in software denote?**
- A. It indicates user satisfaction levels.**
 - B. It represents the adaptability to changing requirements.**
 - C. It measures the ease of performing updates and corrections.**
 - D. It assesses the speed of software execution.**
- 10. What is referred to as the 'test basis'?**
- A. Documents specifying test case execution**
 - B. All documents from which requirements can be inferred**
 - C. The final report on testing outcomes**
 - D. The schedule of testing activities**

Answers

SAMPLE

- 1. B**
- 2. C**
- 3. A**
- 4. A**
- 5. C**
- 6. C**
- 7. B**
- 8. B**
- 9. C**
- 10. B**

SAMPLE

Explanations

SAMPLE

1. How is defect density calculated?

- A. Defects per feature
- B. Defects divided by component size**
- C. Defects per release
- D. Total defects documented

Defect density is calculated by dividing the total number of defects by the size of the component being measured. This size can be expressed in various units, such as lines of code, function points, or any other measure that quantifies the size or complexity of the software component. The rationale behind using this method is that it provides a normalized metric that allows for meaningful comparisons across different components or releases. By relating the number of defects to size, teams can better understand the quality of their software relative to its complexity or functional breadth. This metric is particularly useful in identifying areas that may require further focus or improvement, as a higher defect density could indicate that a component is problematic or that more resources should be allocated to its development or testing. The other options, while relevant in their contexts, do not accurately reflect how defect density is specifically calculated. For instance, counting defects per feature or per release might provide insights into overall quality or performance but does not standardize the defect count against a consistent metric of size or complexity. Similarly, simply documenting total defects lacks the necessary context related to size, making it difficult to assess quality effectively.

2. What is the outcome of the test execution process?

- A. The selection of test cases
- B. The preparation of test data
- C. The actual results based on executed tests**
- D. The design of test conditions

The outcome of the test execution process is the actual results based on executed tests. During test execution, the planned test cases are carried out on the software to assess its functionality and performance. This process enables testers to observe how the application behaves in practice and to capture the actual outcomes of each test against the expected results. The primary goal of executing tests is to determine whether the software behaves as intended. By documenting the actual results, teams can identify any discrepancies from the expected outcomes. These findings are critical for evaluating the quality of the product, identifying defects, and guiding future development efforts. While the other choices involve important aspects of the testing process, they do not represent the culmination of test execution itself. The selection of test cases, preparation of test data, and design of test conditions are all preparatory activities that occur before execution. In contrast, capturing the actual results is a direct consequence of the execution and provides the data necessary for analysis, reporting, and decision-making regarding the software under test.

3. What term describes a person's perceptions and responses resulting from the use or anticipated use of a software product?

A. User experience

B. User interface

C. User story

D. White-box testing

User experience refers to the overall feelings, attitudes, and perceptions a person has while interacting with a software product, whether during actual usage or anticipating its use. This encompasses all aspects of the end-user's interaction with the product, including usability, accessibility, performance, and satisfaction. Understanding user experience is crucial in the development and testing phases, as it ensures that the software meets the needs and expectations of its users. This contrasts with user interface, which specifically refers to the visual elements and controls through which users interact with the software, rather than the feelings and perceptions derived from that interaction. A user story is a brief description of a feature from the perspective of the end user and is part of agile methodologies focusing on customer value. White-box testing relates to testing internal structures or workings of an application rather than the user's experience of it.

4. Which testing approach responds dynamically to the behavior of a test object?

A. Reactive testing

B. Proactive testing

C. Static testing

D. Exploratory testing

Reactive testing is characterized by its focus on dynamically responding to the behavior of the test object during the testing process. This approach is particularly useful in agile environments where requirements and implementations may evolve rapidly. By observing the actual performance and behavior of the application as it is being tested, testers can immediately identify any issues or unexpected behaviors. This flexibility allows for adjustments in testing strategies based on real-time feedback, ensuring that the testing effort remains aligned with the current state of the software being developed. In contrast, proactive testing involves anticipating potential issues before they arise and planning tests accordingly, while static testing centers on examining the code or documentation without executing the code. Exploratory testing, although dynamic in nature, emphasizes unscripted and session-based test execution rather than directly responding to behavior, which makes reactive testing the more appropriate choice in this context.

5. What is the primary focus of 'white-box testing'?

- A. Analyzing user satisfaction
- B. Assessing external customer interaction
- C. Examining internal code structure**
- D. Verifying requirement specifications

The primary focus of white-box testing is to examine the internal code structure of the application under test. This testing method involves looking at the internal workings of the software, including the logic, algorithms, and code paths. By doing so, testers can ensure that the internal operations behave as expected, allowing for a more thorough assessment of the application's functionality and performance. White-box testing enables testers to identify potential vulnerabilities, optimize code, and verify that the various components interact correctly at the code level. This approach often involves techniques such as code coverage analysis, control flow testing, and path testing. In contrast, the other options focus on aspects that do not align with the intent of white-box testing. For example, analyzing user satisfaction and assessing external customer interactions pertain more to user experience and usability testing rather than the internal mechanics of the software. Verifying requirement specifications is closely related to black-box testing, where the focus is on ensuring that the software meets the specified requirements without any regard to how those requirements are implemented internally.

6. What is the name given to a scenario that describes the expected execution of test in structured sequences?

- A. Test result
- B. Test scope
- C. Test specification**
- D. Test session

The correct answer is test specification. This term refers to a detailed description of the necessary conditions, inputs, execution sequences, and expected outcomes for conducting tests on a given system or application. A test specification outlines how the testing process is to be carried out and serves as a blueprint for testers to follow, ensuring a structured and methodical approach to testing. In agile testing, having a clear test specification helps the team understand the individual tests and their purpose, making it easier to execute tests in the intended order and to identify the specific objectives of each test. It provides a foundation for ensuring that all aspects of the system are tested according to the agreed criteria and helps maintain consistency across different test runs. The other options do not capture the essence of this structured approach to test execution. A test result refers to the outcome of executing a test, focusing on whether the test passed or failed, whereas test scope outlines the boundaries of what will be tested. A test session typically describes a period during which testing takes place, rather than the detailed sequences required for execution. Thus, test specification is the most appropriate term that aligns with structured sequences in testing scenarios.

7. What are attributes of software products that prevent unauthorized access called?

- A. Integrity**
- B. Security**
- C. Scalability**
- D. Availability**

The attributes of software products that prevent unauthorized access are referred to as security. Security in software development encompasses various measures and practices designed to protect systems from threats and vulnerabilities. This includes the mechanisms that ensure confidentiality, integrity, and availability of information and services, effectively preventing unauthorized access or breaches. In a software context, security measures can involve encryption, authentication processes, and access controls that restrict who can view or manipulate data. This aspect is critical in maintaining the trust of users and ensuring the data is protected from unauthorized use. While the other options reference important qualities of software products, they do not specifically address the prevention of unauthorized access. Integrity focuses on ensuring that data remains accurate and unaltered, scalability deals with the capacity of a system to handle growth, and availability relates to the system's uptime and readiness for use. These aspects are essential for overall software reliability and performance, but they do not directly pertain to the protective measures against unauthorized access like security does.

8. What is defined as a set of test cases or test procedures to be executed in a specific test cycle?

- A. Test design**
- B. Test suite**
- C. Test specification**
- D. Test session**

A test suite is a collection of test cases or test procedures that are grouped together to be executed as part of a specific test cycle. This organization helps to streamline the testing process by allowing multiple tests to be run in a coordinated manner, ensuring that related tests are executed consistently and efficiently. In an Agile context, where rapid testing and iterative development are crucial, having a well-defined test suite allows teams to focus on verifying specific functionalities or features during each iteration. It promotes clarity in the testing process by organizing tests that may share common objectives, such as testing a particular feature or module. While the other terms have their respective meanings, they do not specifically refer to the organized collection of test cases meant for execution in a test cycle. Test design refers to the process of creating the test cases, test specification involves detailing the requirements and outcomes for testing, and a test session typically refers to a specific period during which testing is conducted, rather than the grouping of tests themselves.

9. What does maintainability in software denote?

- A. It indicates user satisfaction levels.
- B. It represents the adaptability to changing requirements.
- C. It measures the ease of performing updates and corrections.**
- D. It assesses the speed of software execution.

Maintainability in software refers specifically to the ease with which software can be updated and corrected. This encompasses the ability to make changes to the software to fix defects or enhance functionality without excessive cost, time, or complexity. High maintainability means that developers can quickly understand the system's structure and logic, making it simpler to implement changes. When consider the other options, user satisfaction addresses the measurements of how well the software meets user needs, which does not pertain directly to the software's maintenance aspects. Adaptability to changing requirements speaks to how well the software can adjust to new demands, yet this is more about flexibility rather than ease of maintenance. Lastly, assessing the speed of software execution relates to performance metrics, not to how easily the software can be maintained or altered. Thus, the focus on ease of performing updates and corrections clearly aligns with the definition of maintainability in software.

10. What is referred to as the 'test basis'?

- A. Documents specifying test case execution
- B. All documents from which requirements can be inferred**
- C. The final report on testing outcomes
- D. The schedule of testing activities

The term 'test basis' refers to the totality of documents from which requirements are derived for the purpose of creating test cases. This includes any documentation that provides insight into what should be tested, such as requirements specifications, user stories, and other relevant project documents. By identifying all potential sources of requirements, testers can ensure comprehensive coverage in their testing strategy. This understanding is crucial in the context of Agile testing, where requirements often evolve through iterative development. Testers need to rely on various documents created during these iterations to effectively validate that the product meets stakeholder needs. While the other options reference important aspects of the testing process, they do not capture the essence of what constitutes the test basis. For instance, documents specifying test case execution focus on the practical aspects of testing rather than the foundational requirements. The final report on testing outcomes reflects the results and findings after the tests have been executed, and the schedule of testing activities outlines the timeline rather than the source of requirements. Thus, the concept of the test basis is distinct and primarily centered on documents that inform what needs to be tested.