

ISTQB Certified Tester Foundation Level (CTFL) PRO 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. Which tool type is often associated with the requirement to validate tool functionality?**
 - A. Test execution tool**
 - B. Requirements management tool**
 - C. Performance testing tool**
 - D. Static analysis tool**
- 2. What is meant by 'valid and invalid advances' in the context of the ATM upgrade project?**
 - A. Various types of transactions**
 - B. Different levels of access to test features**
 - C. The range of cash amounts customers can withdraw**
 - D. The total number of supported credit cards**
- 3. Which of the following is not a type of incremental testing approach?**
 - A. Top down**
 - B. Big-bang**
 - C. Bottom up**
 - D. Functional incrementation**
- 4. Which testing technique emphasizes assessing product risks?**
 - A. Experience-based testing**
 - B. Risk-based testing**
 - C. Random testing**
 - D. Ad-hoc testing**
- 5. What can a risk-based approach to testing provide?**
 - A. The types of test techniques to be employed.**
 - B. The total tests needed to provide 100 per cent coverage.**
 - C. An estimation of the total cost of testing.**
 - D. Only that test execution is effective at reducing risk.**

- 6. What is most likely performed by developers?**
- A. Technical review of a functional specification.**
 - B. Walkthrough of a requirements document.**
 - C. Informal review of a program specification.**
 - D. Static analysis of a software model.**
- 7. Which is true about regression testing?**
- A. Regression tests are designed to ensure all defects are resolved**
 - B. They are often performed in integration testing**
 - C. Regression tests are not suitable for automation**
 - D. Regression tests are only performed before major releases**
- 8. What is the significance of independent testing?**
- A. Independent testers make fewer assumptions than developers.**
 - B. Independent testers are isolated from the development team.**
 - C. Independent testers can verify assumptions made during the system's specification and implementation.**
 - D. Independent testers have a greater sense of responsibility for quality than developers.**
- 9. Which three measures are typically part of the test approach in the medical domain that are not always applicable in other domains?**
- A. High level of documentation**
 - B. Failure Mode and Effect Analysis (FMEA) sessions**
 - C. Traceability to requirements**
 - D. Test case execution**
- 10. "How much testing is enough?"**
- A. This question is impossible to answer**
 - B. This question is easy to answer**
 - C. The answer depends on the risk for your industry, contract and special requirements**
 - D. This answer depends on the maturity of your developers**

Answers

SAMPLE

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

SAMPLE

Explanations

SAMPLE

1. Which tool type is often associated with the requirement to validate tool functionality?

- A. Test execution tool**
- B. Requirements management tool**
- C. Performance testing tool**
- D. Static analysis tool**

The requirement to validate tool functionality is primarily associated with the requirements management tool. This type of tool is designed to capture, track, and manage requirements throughout the software development lifecycle. Because requirements form the foundation upon which software is built, ensuring that these tools function correctly is paramount for the success of any project. Validating the functionality of a requirements management tool involves ensuring that it can accurately capture requirements, manage changes, facilitate traceability, and support stakeholder communication. If this tool does not work as intended, it can lead to incomplete or mismanaged requirements, which can derail the entire project. In contrast, while test execution tools, performance testing tools, and static analysis tools also require functionality validation, they do not typically focus on the requirements aspect, which is central to a requirements management tool. Each of these tools serves different purposes in the software development process, but the core role of a requirements management tool distinctly highlights the need for validation to ensure that the requirements are handled properly and effectively throughout the development cycle.

2. What is meant by 'valid and invalid advances' in the context of the ATM upgrade project?

- A. Various types of transactions**
- B. Different levels of access to test features**
- C. The range of cash amounts customers can withdraw**
- D. The total number of supported credit cards**

In the context of an ATM upgrade project, 'valid and invalid advances' typically refers to the range of cash amounts that customers can withdraw. Valid advances would represent the permissible withdrawal amounts that the ATM can dispense, based on account balance, withdrawal limits, and operational status. For instance, valid amounts might be those that are in increments defined by the bank, while invalid advances would cover amounts attempted by customers that exceed their account limits or are not dispensed by the ATM due to machine restrictions. Understanding valid and invalid cash advances is critical for ensuring the ATM functions correctly and meets customer expectations. It also plays a role in testing the ATM system to ensure that it accurately recognizes and processes valid withdrawal amounts while properly rejecting invalid ones. This is crucial for maintaining the integrity of transactions and preventing errors or fraud in cash dispensing.

3. Which of the following is not a type of incremental testing approach?

A. Top down

B. Big-bang

C. Bottom up

D. Functional incrementation

The big-bang testing approach is not considered an incremental testing approach. In big-bang testing, all components or modules are integrated and tested simultaneously after the completion of the entire development phase. This means that testing starts only after all parts of the system are developed, which does not allow for any incremental testing of components as they are completed. On the other hand, incremental testing approaches such as top-down, bottom-up, and functional incrementation involve the gradual integration and testing of components or features. In top-down, testing begins with higher-level components and gradually works down to lower-level components, while bottom-up starts with lower-level components and integrates upwards. Functional incrementation focuses on adding functional features incrementally and testing each addition as it occurs. These approaches are beneficial because they allow for earlier detection of defects and more manageable testing efforts.

4. Which testing technique emphasizes assessing product risks?

A. Experience-based testing

B. Risk-based testing

C. Random testing

D. Ad-hoc testing

Risk-based testing is a technique that specifically focuses on identifying, prioritizing, and mitigating the risks associated with a product. It allows the testing process to be driven by the potential impact and likelihood of various risks, ensuring that the most critical areas of the software are thoroughly tested. By concentrating on the aspects of the product that pose a higher risk to its success or the user experience, this approach enables testers to allocate their resources effectively and address the most significant concerns first. In contrast, experience-based testing relies on the tester's past experiences and insights to inform the testing process, but it does not explicitly organize efforts around risk. Similarly, random testing involves executing tests without a specific strategy, thus lacking a focus on risk assessment. Ad-hoc testing is informal and lacks a structured approach, making it less effective in systematically addressing product risks. Therefore, risk-based testing stands out as the method that directly aligns with assessing and addressing product risks.

5. What can a risk-based approach to testing provide?

- A. The types of test techniques to be employed.**
- B. The total tests needed to provide 100 per cent coverage.**
- C. An estimation of the total cost of testing.**
- D. Only that test execution is effective at reducing risk.**

A risk-based approach to testing focuses on identifying and assessing risks associated with the software being tested. By prioritizing testing efforts based on the potential impact and likelihood of those risks, it allows teams to determine the types of test techniques that are most appropriate to address the identified risks. For instance, areas of the software that are deemed to have higher risks may leverage more detailed and rigorous testing techniques, such as exploratory testing or performance testing, to ensure that significant issues are identified early. Conversely, lower-risk areas may require less intensive techniques. This strategic alignment of testing techniques with risk levels fosters efficient resource allocation and improves overall testing effectiveness. The other options do not accurately reflect the advantages of a risk-based approach. While it may inform cost estimations and address test execution effectiveness, the primary benefit here is the tailored selection of test techniques based on risk assessment rather than a blanket statement about testing coverage or cost estimation.

6. What is most likely performed by developers?

- A. Technical review of a functional specification.**
- B. Walkthrough of a requirements document.**
- C. Informal review of a program specification.**
- D. Static analysis of a software model.**

The most likely task performed by developers in this context is static analysis of a software model. Static analysis involves examining the code or the software model without executing it. This process is crucial for identifying potential bugs or vulnerabilities early in the development cycle, helping to ensure code quality and adherence to specified standards. Developers are typically responsible for writing and maintaining the code, making static analysis an integral part of their workflow as they evaluate their own or others' code for best practices and potential issues. In contrast, while the other options involve activities that may require Technical reviews, walkthroughs, or informal reviews, these tasks usually involve multiple stakeholders, including business analysts or testers, and may focus more on functionality, requirements, or specifications. Static analysis, however, is distinctly a developer-focused activity aimed primarily at improving the quality of the code they produce.

7. Which is true about regression testing?

- A. Regression tests are designed to ensure all defects are resolved
- B. They are often performed in integration testing**
- C. Regression tests are not suitable for automation
- D. Regression tests are only performed before major releases

Regression testing is primarily focused on verifying that recent changes in code have not adversely affected the existing functionality of the software. Although option B states that regression tests are often performed in integration testing, it reflects a common practice within the testing lifecycle. Integration testing often involves combining different components or modules to see how they work together, and regression testing can be a part of this phase to ensure that integrating these components hasn't introduced any new defects or issues in already functioning parts of the system. This means running regression tests during integration testing helps to validate that the interactions between integrated components behave as expected. Regression tests can and should be executed after various changes, not just during integration testing, to keep a system stable and functional. The nature of regression testing allows it to fit into different phases of software development and different types of testing practices, making it an integral focus throughout the software lifecycle. Therefore, the assertion in option B makes it a true statement regarding regression testing.

8. What is the significance of independent testing?

- A. Independent testers make fewer assumptions than developers.
- B. Independent testers are isolated from the development team.
- C. Independent testers can verify assumptions made during the system's specification and implementation.**
- D. Independent testers have a greater sense of responsibility for quality than developers.

Independent testing plays a crucial role in the software development process, particularly because it allows testers to approach the application from an objective standpoint. The significance of independent testing is primarily rooted in the ability of independent testers to verify and validate the assumptions made during the system's specification and implementation phases. This verification process is vital because it ensures that the developed system meets the specified requirements and functions correctly according to the intended use. Independent testers bring a fresh perspective, which can uncover issues that developers, who may have been too close to the project, might overlook. Their role is to challenge and confirm the assumptions made at earlier stages, ensuring comprehensive quality assurance. By focusing on the verification of assumptions, independent testers help to enhance the overall reliability and effectiveness of the product, contributing significantly to delivering high-quality software.

9. Which three measures are typically part of the test approach in the medical domain that are not always applicable in other domains?

- A. High level of documentation**
- B. Failure Mode and Effect Analysis (FMEA) sessions**
- C. Traceability to requirements**
- D. Test case execution**

In the medical domain, the approach to testing is heavily influenced by the need for safety, compliance, and regulatory standards. The emphasis on a high level of documentation is crucial because it provides a clear and traceable record of every phase of the testing process. This documentation not only ensures compliance with stringent regulations imposed by health authorities, but it also plays a vital role in demonstrating the effectiveness and safety of medical products or software. The documentation serves multiple purposes: it aids in risk management, facilitates peer reviews, and provides necessary information for investigations or audits. Given the potential impact on patient safety, thorough documentation is particularly critical in medical testing, making it a distinctive aspect compared to other industries where the same level of documentation may not be as rigorously required. In contrast, while FMEA sessions, traceability to requirements, and test case execution are valuable in many testing contexts, they may not be emphasized as strongly or mandated in other domains outside of medical applications. This difference highlights the unique testing requirements inherent in medical devices and software, where oversight is paramount.

10. "How much testing is enough?"

- A. This question is impossible to answer**
- B. This question is easy to answer**
- C. The answer depends on the risk for your industry, contract and special requirements**
- D. This answer depends on the maturity of your developers**

The question of "How much testing is enough?" can vary significantly based on several factors, notably the context in which the software operates. The correct answer emphasizes that the needed extent of testing is influenced by the risks associated with a particular industry, contractual obligations, and any special requirements that may be applicable. For instance, in industries like healthcare or finance, where software failures can lead to significant harm or legal implications, the amount of testing is likely to be more extensive compared to less critical domains. Additionally, contractual requirements might dictate specific levels of assurance or validation that affect the testing scope. This context-driven approach acknowledges that one size does not fit all when it comes to software testing; testing must be tailored to mitigate risks effectively. Understanding the risk-based approach to testing is crucial. It places emphasis on evaluating what could go wrong, how severe the consequences could be, and the likelihood of those failures. This insight allows teams to prioritize their testing efforts, focusing on high-risk areas, thus determining a suitable level of testing that balances quality assurance with project constraints.