Free Questions for ISTQB-CTFL by ebraindumps

Shared by Griffith on 29-01-2024

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

Can "cost" be regarded as Exit criteria?

Options:

A- Yes. Spending too much money on test ng will result in an unprofitable product, and having cost as an exit criterion helps avoid this

B- No. The financial value of product quality cannot be estimated, so it is incorrect to use cost as an exit criterion

C- Yes. Going by cost as an exit criterion constrains the testing project which will hello achieve the desired quality level defined for the project

D- No The cost of testing cannot be measured effectively, so it is incorrect to use cost as an exit criterion

Answer:

A

Explanation:

Cost can be regarded as an exit criterion for testing, because it is a factor that affects the profitability and feasibility of the software product. Testing is an investment that aims to improve the quality and reliability of the software product, but it also consumes resources, such as time, money, and human effort. Therefore, testing should be planned and executed in a way that balances the cost and benefit of testing activities. Having cost as an exit criterion helps to avoid spending too much money on testing, which may result in an unprofitable product or a loss of competitive advantage. Cost can also help to prioritize and focus the testing efforts on the most critical and valuable features and functions of the software product. However, cost should not be the only exit criterion for testing, as it may not reflect the true quality and risk level of the software product. Other exit criteria, such as defect rate, test coverage, user satisfaction, etc., should also be considered and defined in the test plan.

The other options are incorrect, because they either deny the importance of cost as an exit criterion, or they make false or unrealistic assumptions about the cost of testing. Option B is incorrect, because the financial value of product quality can be estimated, for example, by using cost-benefit analysis, return on investment, or cost of quality models. Option C is incorrect, because going by cost as an exit criterion does not necessarily constrain the testing project or help achieve the desired quality level. Cost is a relative and variable factor that depends on the scope, complexity, and context of the software product and the testing project. Option D is incorrect, because the cost of testing can be measured effectively, for example, by using metrics, such as test effort, test resources, test tools, test environment, etc.

Question 2

Question Type: MultipleChoice

Mark the correct sentences:

* Defects are a result of environmental conditions and are also referred to as "Failures"

* A human mistake may produce a defect

* A system mil totally fail to operate correctly when a failure exists in it

* When a defect exists in a system it may result in a failure

* Defects occur only as a result of technology changes

Options:

A- II, IV

B- I, II

C- IV, V

D- II, III, IV

Answer:

A

Explanation:

The question is about marking the correct sentences among the given statements related to defects, failures, and mistakes.According to the ISTQB glossary, the definitions of these terms are1:

Defect: A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g. an incorrect statement or data definition. A defect, if encountered during execution, may cause a failure of the component or system.

Failure: An event in which a component or system does not perform a required function within specified limits.

Mistake: A human action that produces an incorrect result.

Therefore, out of the five given statements, only two are correct, namely:

A human mistake may produce a defect: This is true, as a mistake is a source or cause of a defect, e.g. a programmer may make a mistake in writing a code statement, which results in a defect in the software component.

When a defect exists in a system it may result in a failure: This is true, as a defect is a potential or actual cause of a failure, e.g. a defect in the software component may cause the system to fail to perform a required function when the defect is encountered during execution.

The other three statements are incorrect, namely:

Defects are a result of environmental conditions and are also referred to as "Failures": This is false, as defects are not a result of environmental conditions, but of mistakes or other factors, and defects are not the same as failures, but rather the causes of failures.

A system will totally fail to operate correctly when a failure exists in it: This is false, as a system may not necessarily fail completely or stop operating when a failure occurs, but may continue to operate with reduced functionality or performance, or with incorrect results.

Defects occur only as a result of technology changes: This is false, as defects can occur due to various reasons, not only technology changes, such as human mistakes, design flaws, requirement changes, hardware failures, etc.

1: ISTQB Glossary of Testing Terms 4.0, 2023, available at ISTQB) and ASTQB).

Question 3

Question Type: MultipleChoice

During component testing of a program if 100% decision coverage is achieved, which of the following coverage criteria is also guaranteed to be 100%?

Options:

A- 100% Stale transition coverage

B- 100% Equivalence class coverage

C- 100% Boundary value coverage

D- 100% Statement coverage

Answer:

D

Explanation:

Statement coverage is a structural coverage metric that measures the percentage of executable statements in the source code that are executed by a test suite1.Decision coverage is another structural coverage metric that measures the percentage of decision outcomes (such as branches or conditions) in the source code that are executed by a test suite1.Decision coverage is a stronger metric than statement coverage, because it requires that every possible outcome of each decision is tested, while statement coverage only requires that every statement is executed at least once2. Therefore, if a test suite achieves 100% decision coverage, it also implies that it achieves 100% statement coverage, because every statement in every branch or condition must have been executed.However, the converse is not true: 100% statement coverage does not guarantee 100% decision coverage, because some branches or conditions may have multiple outcomes that are not tested by the test suite2. For example, consider the following pseudocode:

if (x > 0) then print("Positive") else print("Non-positive") end if

A test suite that executes this code with x = 1 and x = -1 will achieve 100% statement coverage, because both print statements are executed. However, it will not achieve 100% decision coverage, because the condition x > 0 has only been tested with two outcomes: true and false. The third possible outcome, x = 0, has not been tested by the test suite. Therefore, the test suite may miss a potential bug or error in the condition or the branch.

The other options, such as stale transition coverage, equivalence class coverage, and boundary value coverage, are not guaranteed to be 100% by achieving 100% decision coverage.Stale transition coverage is a structural coverage metric that measures the percentage of transitions between states in a state machine that are executed by a test suite3.Equivalence class coverage is a functional coverage metric that measures the percentage of equivalence classes (or partitions) of input or output values that are tested by a test suite4.Boundary value coverage is another functional coverage metric that measures the percentage of boundary values (or extreme values) of input or output ranges that are tested by a test suite4. These metrics are independent of decision coverage, because they are based on different aspects of the system under test, such as its behavior, functionality, or specification. Therefore, achieving 100% decision coverage does not imply achieving 100% of any of these metrics, and vice versa.Reference=ISTQB Certified Tester Foundation

Level Syllabus v4.0,Test Coverage in Software Testing - Guru99,Structural Coverage Metrics - MATLAB & Simulink - MathWorks India,Test Design Coverage in Software Testing - GeeksforGeeks.

Question 4

Question Type: MultipleChoice

The four test levels used in ISTQB syllabus are:

1. Component (unit) testing

2. Integration testing

3. System testing

4. Acceptance testing

An organization wants to do away with integration testing but otherwise follow V-model. Which of the following statements is correct?

Options:

A- It is allowed as organizations can decide on men test levels to do depending on the context of the system under test

B- It is allowed because integration testing is not an important test level arc! can be dispensed with.

C- It is not allowed because integration testing is a very important test level and ignoring i: means definite poor product quality

D- It is not allowed as organizations can't change the test levels as these are chosen on the basis of the SDLC (software development life cycle) model

Answer:

D

Explanation:

The V-model is a software development life cycle model that defines four test levels that correspond to four development phases: component (unit) testing with component design, integration testing with architectural design, system testing with system requirements, and acceptance testing with user requirements. The V-model emphasizes the importance of verifying and validating each phase of development with a corresponding level of testing, and ensuring that the test objectives, test basis, and test artifacts are aligned and consistent across the test levels. Therefore, an organization that wants to follow the V-model cannot do away with integration testing, as it would break the symmetry and completeness of the V-model, and compromise the quality and reliability of the software or system under test. Integration testing is a test level that aims to test the interactions and interfaces between components or subsystems, and to detect any defects or inconsistencies that may arise from the integration of different parts of the software or system. Integration testing is essential for ensuring the functionality, performance, and compatibility of the software or system as a whole, and for identifying and resolving any integration issues early in the development process. Skipping integration testing would increase the risk of finding serious defects later in the test process, or worse, in the production environment, which would be more costly and difficult to fix, and could

damage the reputation and credibility of the organization. Therefore, the correct answer is D .

The other options are incorrect because:

A) It is not allowed as organizations can decide on the test levels to do depending on the context of the system under test. While it is true that the choice and scope of test levels may vary depending on the context of the system under test, such as the size, complexity, criticality, and risk level of the system, the organization cannot simply ignore or skip a test level that is defined and required by the chosen software development life cycle model. The organization must follow the principles and guidelines of the software development life cycle model, and ensure that the test levels are consistent and coherent with the development phases. If the organization wants to have more flexibility and adaptability in choosing the test levels, it should consider using a different software development life cycle model, such as an agile or iterative model, that allows for more dynamic and incremental testing approaches.

B) It is not allowed because integration testing is not an important test level and can be dispensed with. This statement is false and misleading, as integration testing is a very important test level that cannot be dispensed with. Integration testing is vital for testing the interactions and interfaces between components or subsystems, and for ensuring the functionality, performance, and compatibility of the software or system as a whole. Integration testing can reveal defects or inconsistencies that may not be detected by component (unit) testing alone, such as interface errors, data flow errors, integration logic errors, or performance degradation. Integration testing can also help to verify and validate the architectural design and the integration strategy of the software or system, and to ensure that the software or system meets the specified and expected quality attributes, such as reliability, usability, security, and maintainability. Integration testing can also provide feedback and confidence to the developers and stakeholders about the progress and quality of the software or system development. Therefore, integration testing is a crucial and indispensable test level that should not be skipped or omitted.

C) It is not allowed because integration testing is a very important test level and ignoring it means definite poor product quality. This statement is partially true, as integration testing is a very important test level that should not be ignored, and skipping it could result in poor product quality. However, this statement is too strong and absolute, as it implies that integration testing is the only factor that determines the product quality, and that ignoring it would guarantee a poor product quality. This is not necessarily the case, as there

may be other factors that affect the product quality, such as the quality of the requirements, design, code, and other test levels, the effectiveness and efficiency of the test techniques and tools, the competence and experience of the developers and testers, the availability and adequacy of the resources and environment, the management and communication of the project, and the expectations and satisfaction of the customers and users. Therefore, while integration testing is a very important test level that should not be skipped, it is not the only test level that matters, and skipping it does not necessarily mean definite poor product quality, but rather a higher risk and likelihood of poor product quality.

Reference= ISTQB Certified Tester Foundation Level Syllabus, Version 4.0, 2018, Section 2.3, pages 16-18; ISTQB Glossary of Testing Terms, Version 4.0, 2018, pages 38-39; ISTQB CTFL 4.0 - Sample Exam - Answers, Version 1.1, 2023, Question 104, page 36.

Question 5

Question Type: **MultipleChoice**

You are testing a room upgrade system for a hotel. The system accepts three differed types of room (increasing order of luxury): Platinum. Silver and Gold Luxury. ONLY a Preferred Guest Card holder s eligible for an upgrade.

Below you can find the decision table defining the upgrade eligibility:

What is the expected result for each of the following test cases?

Customer A: Preference Guest Card holder, holding a Silver room

Customer B: Non Preferred Guest Card holder, holding a Platinum room

Options:

A- Customer A; doesn't offer any upgrade; Customer B: offers upgrade to Gold luxury room

B- Customer A: doesn't offer any upgrade; Customer B: doesn't offer any upgrade.

C- Customer A: offers upgrade to Gold Luxury room; Customer B: doesn't offer any upgrade

D- Customer A: offers upgrade to Silver room; Customer B: offers upgrade to Silver room.

According to the decision table in the image, a Preferred Guest Card holder with a Silver room is eligible for an upgrade to Gold Luxury (YES), while a non-Preferred Guest Card holder, regardless of room type, is not eligible for any upgrade (NO). Therefore, Customer A (a

Preferred Guest Card holder with a Silver room) would be offered an upgrade to Gold Luxury, and Customer B (a non-Preferred Guest Card holder with a Platinum room) would not be offered any upgrade.Reference= The answer is derived directly from the decision table provided in the image; specific ISTQB Certified Tester Foundation Level (CTFL) v4.0 documents are not referenced.

Answer:

C

Question 6

Question Type: MultipleChoice

Which of the following statements about estimation of the test effort is WRONG?

Options:

A- Once the test effort is estimated, resources can be identified and a schedule can be drawn up.

B- Effort estimate can be inaccurate because the quality of the product under tests is not known.

C- Effort estimate depends on the budget of the project.

D- Experience based estimation is one of the estimation techniques.

Answer:

C

Explanation:

Effort estimate does not depend on the budget of the project, but rather on the scope, complexity, and quality of the software product and the testing activities1. Budget is a constraint that may affect the feasibility and accuracy of the effort estimate, but it is not a factor that determines the effort estimate.Effort estimate is the amount of work required to complete the testing activities, measured in terms of person-hours, person-days, or person-months2.

The other options are correct because:

A)Once the test effort is estimated, resources can be identified and a schedule can be drawn up, as they are interrelated aspects of the test planning process3.Resources are the people, tools, equipment, and facilities needed to perform the testing activities4.Schedule is the time frame and sequence of the testing activities, aligned with the project milestones and deadlines5.

B)Effort estimate can be inaccurate because the quality of the product under tests is not known, as it affects the number and severity of the defects that may be found and the rework that may be needed to fix them6.Quality is the degree to which the software product satisfies the specified requirements and meets the needs and expectations of the users and clients7.

D) Experience based estimation is one of the estimation techniques, which relies on the judgment and expertise of the testers and other project stakeholders to estimate the test effort based on similar projects or tasks done in the past. Experience based estimation can be useful when there is a lack of historical data, formal methods, or detailed information about the software product and the testing activities.

Reference=

1ISTQB Certified Tester Foundation Level Syllabus v4.0, 2023, p. 154

2ISTQB Certified Tester Foundation Level Syllabus v4.0, 2023, p. 155

3ISTQB Certified Tester Foundation Level Syllabus v4.0, 2023, p. 156

4ISTQB Certified Tester Foundation Level Syllabus v4.0, 2023, p. 157

5ISTQB Certified Tester Foundation Level Syllabus v4.0, 2023, p. 158

6ISTQB Certified Tester Foundation Level Syllabus v4.0, 2023, p. 159

7ISTQB Certified Tester Foundation Level Syllabus v4.0, 2023, p. 16

[8] ISTQB Certified Tester Foundation Level Syllabus v4.0, 2023, p. 160

[9] ISTQB Certified Tester Foundation Level Syllabus v4.0, 2023, p. 161