ISTQB.ISTQB-CTFL.v2023-10-24.q28

Exam Code:	ISTQB-CTFL
Exam Name:	ISTQB-Foundation Level Exam
Certification Provider:	ISTQB
Free Question Number:	28
Version:	v2023-10-24
# of views:	451
# of Questions views:	280
https://www.freeqas.com/qa/ISTQB/ISTQB-CTFL/ISTQB.ISTQB-CTFL.v2023-10-24.q28.html	

NEW QUESTION: 1
Why it is essential that defects found in a review be reported objectively?
A. In order to facilitate easy entry of detected defects in a OTS (Defect Tracking System)
B. In order to allow the author of reviewed work product(S) to take the feedback positively as an effort at improving the product (S) and not as a personal assault
C. In order to allow the review moderator to easily understand them, and assign them to the right developer for fixing
D. In order to allow augmentation of existing checklists used for reviewing the work product (S)
Answer: B (LEAVE A REPLY)
The purpose of a review is to find defects and improve the quality of the work product, not to criticize or blame the author. Reporting defects objectively means describing them factually and constructively, without using negative or emotional language that could offend the author or damage their motivation. This way, the author can take the feedback positively as an effort at improving the product and not as a personal assault. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 138.

NEW QUESTION: 2
Which of the following is NOT an experience-based technique?
A. Boundary value analysis.
B. Error guessing
C. Exploratory testing
D. Fault attack
Answer: A (LEAVE A REPLY)

Boundary value analysis is not an experience-based technique, but rather a specification-based technique (also known as black-box technique). Experience-based techniques are techniques that rely on the tester's knowledge and intuition to derive and select test cases based on their experience with similar systems, technologies, domains, risks, etc. Some examples of experience-based techniques are error guessing, exploratory testing, fault attack, checklist-based testing, etc. Specification-based techniques are techniques that rely on the tester's analysis and interpretation of the requirements or specifications of the system under test to derive and select test cases based on some criteria or rules. Some examples of specification-based techniques are equivalence partitioning, boundary value analysis, decision table testing, state transition testing, etc. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 31.

NEW QUESTION: 3
Which of the following is the most important task of a typical test leader?
A. To automate tests.
B. To prepare and acquire test data.
C. To set up the test environment.
D. To coordinate the test strategy with project managers.
Answer: D (LEAVE A REPLY)
The most important task of a typical test leader is to coordinate the test strategy with project managers. The test strategy is a high-level document that defines the general approach and objectives of testing for a project or an organization. The test leader is responsible for defining, documenting, communicating, and implementing the test strategy in alignment with the project goals and constraints. The test leader also needs to coordinate with project managers and other stakeholders to ensure that the test strategy is feasible, effective, and efficient. The other options are not the most important tasks of a typical test leader. To automate tests is a task of a test automation engineer or a test automation specialist. To prepare and acquire test data is a task of a test analyst or a test engineer. To set up the test environment is a task of a test environment manager or a test environment specialist. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 13.

NEW QUESTION: 4
4 equivalence classes are given for integer values:
0 < x <100
100<= x <= 200
200 < x < 500
x >= 500
Which of the following options represent correct set of data for valid equivalence class partitions?
A. 50; 100; 200. 1000

B. 0. 1.99, 100.200,201.499, 500;
C. 0.50; 100; 150.200.350.500;
D. 50; 100; 250; 1000
Answer: (SHOW ANSWER)
The correct set of data for valid equivalence class partitions should include one value from each equivalence class, and no value from outside the range. Option C satisfies this condition, as it has one value from each of the four equivalence classes (50, 100, 250, 500). Option A has two values from the same equivalence class (100 and 200), option B has values outside the range (0 and 0.99), and option D has two values from the same equivalence class (1000 and 500). Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 35.

NEW QUESTION: 5
A software module to be used in a mission critical application incorporates an algorithm for secure transmission of dat a.
Which review type is most appropriate to ensure high quality and technical correctness of the algorithm?
A. Walkthrough
B. Informal Review
C. Technical Review
D. Management Review
Answer: C (LEAVE A REPLY)
A technical review is a type of formal review that involves a team of technical experts who evaluate a software product against a set of predefined quality criteria. A technical review is suitable for ensuring high quality and technical correctness of complex or critical software components, such as algorithms, architectures or designs. A technical review is not a walkthrough, which is an informal review led by the author of the work product. A technical review is not an informal review, which is a review that does not follow a defined process and has no formal entry or exit criteria. A technical review is not a management review, which is a type of formal review that focuses on business aspects and project progress. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, Chapter 3, page 29-30.

NEW QUESTION: 6
A team's test strategy was to invest equal effort in testing each of a system's modules. After running one test cycle, it turned out that most of the critical bugs were detected in one of the system's modules.
Which testing principal suggests a change to the current test strategy for the next test cycle?
A. Pesticide Paradox
B. Early testing

C. Absence-of-errors fallacy
D. Defect clustering
Answer: D (LEAVE A REPLY)
Defect clustering is a testing principle that states that a small number of modules contain most of the defects detected during pre-release testing, or are responsible for most of the operational failures. Defect clustering can be explained by Pareto's principle (also known as the 80-20 rule), which states that approximately 80% of the problems are found in 20% of the modules. Defect clustering suggests a change to the current test strategy for the next test cycle, as it implies that more effort should be allocated to test the modules that have shown high defect density or criticality. Pesticide paradox is another testing principle that states that if the same tests are repeated over and over again, eventually they will no longer find any new defects. Pesticide paradox suggests a change to the current test strategy for the next test cycle, but not based on defect clustering, but rather on test diversity and coverage. Early testing is a testing principle that states that testing activities should start as early as possible in the software development life cycle and should be focused on defined objectives. Early testing does not suggest a change to the current test strategy for the next test cycle, but rather a proactive approach to prevent defects from occurring or propagating. Absence-of-errors fallacy is a testing principle that states that finding and fixing defects does not help if the system built is unusable and does not fulfill the users' needs and expectations. Absence-of-errors fallacy does not suggest a change to the current test strategy for the next test cycle, but rather a focus on quality attributes and user requirements. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, Chapter 1, page 9-10.

NEW QUESTION: 7
Which of the following statements about decision tables are TRUE?
I Generally, decision tables are generated for low risk test items.
II Test cases derived from decision tables can be used for component tests.
III Several test cases can be selected for each column of the decision table.
IV The conditions in the decision table represent negative tests generally.
A. I. III
B. I. IV
C. II. IV
D. II. III
E. Generally, decision tables are generated for low risk test items. Decision tables are not related to risk level, but rather to complexity level. Decision tables are generated for test items that have complex logic or multiple conditions and actions that need to be tested.
Answer: (SHOW ANSWER)
IV. The conditions in the decision table represent negative tests generally. The conditions in the decision table represent both positive and negative tests, depending on whether they

are valid or invalid inputs for the test item. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, Chapter 4, page 42-43.
Explanation:
A decision table is a technique that shows combinations of inputs and/or stimuli (causes) with their associated outputs and/or actions (effects). A decision table consists of four quadrants: conditions (inputs), actions (outputs), condition entries (values) and action entries (results). The following statements about decision tables are true:
II. Test cases derived from decision tables can be used for component tests. Decision tables can be used to test components that have multiple inputs and outputs that depend on logical combinations of conditions. Decision tables can help cover all possible combinations or scenarios in a systematic way.
III. Several test cases can be selected for each column of the decision table. A column of a decision table represents a unique combination of condition entries and action entries. Several test cases can be selected for each column by varying other input values or expected results that are not part of the decision table. The following statements about decision tables are false:

NEW QUESTION: 8
Manager responsibilities in formal review includes ad except one of the following:
A. Planning the review
B. Determines if the review objectives have been met
C. Allocate time for review
D. Decide on the execution of reviews
Answer: (SHOW ANSWER)
A formal review is a type of review that follows a defined process with formal entry and exit criteria and roles and responsibilities for participants. A formal review can have various roles involved, such as manager, moderator, author, reviewer and scribe. The manager responsibilities in formal review include all except one of the following:
Planning the review (correct responsibility)
Determines if the review objectives have been met (incorrect responsibility) Decide on the execution of reviews (correct responsibility) Allocate time for review (correct responsibility)
The responsibility of determining if the review objectives have been met belongs to the moderator role, not to the manager role. Verified Reference: [A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer], Chapter 3, page 28-29.

NEW QUESTION: 9
A software system checks age in order to determine which welcome screen to display. Age groups are:
Group I: 0-12
Group II; 13-18
Group III: over 18

Which of the below represent boundary values?
A. (-1.0.12.13.18,19)
B. (-1.0,11.12.13,14,18.19)
C. (0.12.13.18.19)
D. (4.5.15.20)
Answer: A (LEAVE A REPLY)
A correct list of boundary values for the age input should include the minimum and maximum values of each age group (0, 12, 13, 18), as well as the values just below and above each boundary (-1, 19). Boundary value analysis is a test design technique that involves testing the values at or near the boundaries of an input domain or output range, as these values are more likely to cause errors than values in the middle. Option A satisfies this condition, as it has all six boundary values (-1, 0, 12, 13, 18, 19). Option B has two values from the same equivalence class (12 and 13), option C has only four boundary values (0, 12, 18, 19), and option D has no boundary values at all. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 34.

NEW QUESTION: 10
As the last stage of a test cycle of an embedded device, you are performing exploratory testing. You observed that some character. (A, X and Z) sent via a serial port to the device do not get registered on the device whereas they should be. You suspect that this could be due to a wrong configuration of the "bit parity" parameter.
Which of the following items of an incident report would you be UNABLE to write down based on this information?
A. Expected result
B. Test case identifier
C. Test setup details
D. Actual result
Answer: (SHOW ANSWER)
An incident report is a document that records the details of an incident. An incident report typically contains the following items:
Identifier: A unique identifier for the incident report
Summary: A concise summary of the incident
Description: A detailed description of the incident, including the steps to reproduce it, the expected and actual results, and any relevant screenshots or logs Severity: The degree of impact that the incident has on the system Priority: The level of urgency for resolving the incident Status: The current state of the incident, such as new, open, resolved, closed, etc. Resolution: The action taken to resolve the incident, such as fix, workaround, reject, etc. Based on the information given in the question, the tester would be able to write down all of these items except for the test case identifier. A test case identifier is a unique identifier for a test case that is used to link it to other test artifacts, such as test plans, test scripts,

test results or incident reports. However, since the tester is performing exploratory testing, there is no predefined test case that can be associated with the incident. Exploratory testing is an approach to testing that emphasizes learning, test design and test execution at the same time. Exploratory testing relies on the tester's skills, creativity and intuition to explore the software under test and discover defects. Exploratory testing does not use formal test cases or scripts, but rather uses test charters or missions that guide the tester's actions and objectives. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, Chapter 3, page 32-33; Chapter 5, page 47-48.

NEW QUESTION: 11
Which of the following statements about testware are correct?
I When closing the test activities, all the testware resources can be uninstalled and released II All the testware should be subject to Configuration Management III. The testware. at the end of the project, should be transferred to the organization responsible for maintenance IV The developers are responsible for the correct installation of the testware
A. II, III
B. I, III
C. I, IV
D. II, IV
Answer: A (LEAVE A REPLY)
Testware is a term that refers to all artifacts produced during the testing process, such as test plans, test cases, test scripts, test data, test results, defect reports, etc. The following statements about testware are correct:
II) All the testware should be subject to Configuration Management. Configuration management is a process that establishes and maintains consistency among work products throughout their life cycle. Configuration management applies to all testware, as it helps ensure their quality and consistency, track their changes and defects, control their versions and access rights, and link them to other artifacts.
III) The testware at the end of the project should be transferred to the organization responsible for maintenance. Maintenance testing is testing performed on a software product after delivery to correct defects or improve performance or other attributes. Maintenance testing requires testware from previous testing activities or phases, such as test cases, test data, test results, etc. Therefore, the testware at the end of the project should be transferred to the organization responsible for maintenance testing, such as support team or maintenance team. The following statements about testware are incorrect:
I) When closing the test activities, all the testware resources can be uninstalled and released. This statement is incorrect, as some testware resources may still be needed for future testing activities or phases, such as maintenance testing or regression testing. Therefore, when closing the test activities, some testware resources should be archived and stored for future use, while others can be uninstalled and released.

IV) The developers are responsible for the correct installation of the testware. This statement is incorrect, as the testers are responsible for the correct installation of the testware. The testers should ensure that they have access to all necessary testware resources and that they are installed and configured properly before starting the test execution. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, Chapter 6, page 58-61.

NEW QUESTION: 12
Which of the following options cover the test types performed during typical system testing phase:
I Usability
II Requirements based scenarios
III Testing parts of the code in isolation
IV Correct order of parameters in API calls
A. I, III
B. I. II
C. II. IV
D. III. IV
Answer: (SHOW ANSWER)
System testing is a level of testing performed to evaluate the behavior and quality of a whole software product or system. System testing can include various types of testing, such as:
I) Usability testing: A type of testing that evaluates how easy, efficient and satisfying it is to use the software product or system from the user's perspective.
II) Requirements based scenarios testing: A type of testing that verifies that the software product or system meets its specified requirements or user stories by executing realistic scenarios or workflows. System testing does not include the following types of testing, as they are more suitable for lower levels of testing, such as unit testing or integration testing:
III) Testing parts of the code in isolation: A type of testing that verifies the functionality and quality of individual software components or units by isolating them from other components or units.
IV) Correct order of parameters in API calls: A type of testing that verifies the functionality and quality of software components or units that communicate with each other through application programming interfaces (APIs) by checking the correct order and format of parameters in API calls. Verified Reference: [A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer], Chapter 2, page 20-21; Chapter 4, page 34-35.

NEW QUESTION: 13
Which ONE of the following statements about acceptance testing is NOT correct?
A. Testing of disaster recovery and backup/restore is usually NOT part of acceptance testing.

B. The customers or system users are often responsible for the acceptance testing.
C. The main goal of acceptance testing is to build confidence in the system, not find defects.
D. Acceptance testing is the last level of testing performed prior to system release.
Answer: A (LEAVE A REPLY)
Acceptance testing is a level of testing performed to verify that a software product meets the agreed acceptance criteria and is acceptable for delivery. Acceptance testing is often performed by the customers or system users, who are the main stakeholders of the software product. The main goal of acceptance testing is to build confidence in the system, not find defects, as defects should have been detected and fixed in earlier levels of testing. Acceptance testing is the last level of testing performed prior to system release, unless there are any changes or fixes that require re-testing. Testing of disaster recovery and backup/restore is usually part of acceptance testing, as these are important aspects of system reliability and security that affect the customer satisfaction and trust. Therefore, statement A is not correct, while statements B, C and D are correct. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, Chapter 2, page 20-21.

NEW QUESTION: 14
Which of the following should be included in a test status report?
I Estimation details
II Total number of open and closed defects
III Actual effort spent
IV Defect reports
V Number of executed, failed, blocked tests
A. III.V
B. II, III
C. I. II. IV
D. II, III.V
Answer: D (LEAVE A REPLY)
The following should be included in a test status report: total number of open and closed defects, actual effort spent, and number of executed, failed, and blocked tests. A test status report is a document that provides information on the results and status of testing activities for a given period or phase. A test status report should include information that is relevant, accurate, and timely for the intended audience and purpose. Some of the information that should be included in a test status report are: total number of open and closed defects, which can indicate the defect trend and defect density of the software product; actual effort spent, which can indicate the productivity and efficiency of the testing process; number of executed, failed, and blocked tests, which can indicate the test progress and test coverage of the software product. The following should not be included in a test status report: estimation details, defect reports, and impact analysis. Estimation

details are not part of a test status report, but rather part of a test plan or a test estimation document. Estimation details provide information on the expected time, resources, and costs for testing activities, not on the actual results or status of testing activities. Defect reports are not part of a test status report, but rather separate documents that provide detailed information on individual defects found during testing. Defect reports include information such as defect description, defect severity, defect priority, defect status, defect resolution, etc. Defect reports can be referenced or summarized in a test status report, but not included in full. Impact analysis is not part of a test status report, but rather part of a risk assessment or prioritization process. Impact analysis provides information on the potential effects or consequences of a change or a defect on the software product or project. Impact analysis can be used to evaluate the amount or scope of testing to be performed, but not to report the results or status of testing activities. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 141.

NEW QUESTION: 15
Which of the following is NOT an objective of testing?
A. Finding defects
B. Providing information for decision-making
C. Gaining confidence about the level of quality of the software
D. Analyzing and removing the cause of failures
Answer: (SHOW ANSWER)
Analyzing and removing the cause of failures is not an objective of testing, but rather a task of development or maintenance. A failure is an event or behavior that deviates from the expected or specified result of a system under test. A failure is caused by an error (also known as a mistake or a fault) in the software code, design, or specification. Analyzing and removing the cause of failures is a process of locating and fixing errors in the software code, design, or specification, which is also known as debugging or defect resolution. Analyzing and removing the cause of failures does not aim to find or report defects, but rather to correct or prevent them. The other options are objectives of testing. Finding defects is one of the main objectives of testing, as it helps to improve the quality and reliability of the software product. Providing information for decision-making is another objective of testing, as it helps to support decision making and risk management. Gaining confidence about the level of quality of the software is another objective of testing, as it helps to assure that the software product meets its requirements and customer or user needs and expectations. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 3.

NEW QUESTION: 16
A system computes prices for bus tickets. The price depends on
- the passenger type (baby, child, adult, senior citizen, student, military)
- the travelling type (as single or in a group)

- the distance (zone 1. 2. 3)
- the kind of transport (ordinary, express)

Which of the following test techniques is the most appropriate one for testing the price computation?

A. Statement coverage
B. State transition testing
C. Equivalence partitioning
D. Use case testing

Answer: (SHOW ANSWER)

Equivalence partitioning is a technique that divides the input data and output results of a software component into partitions of equivalent data. Each partition should contain data that is treated in the same way by the component. Equivalence partitioning can be used to reduce the number of test cases by selecting one representative value from each partition. Equivalence partitioning is suitable for testing the price computation, as it can identify different partitions based on the passenger type, the travelling type, the distance and the kind of transport. Equivalence partitioning is not statement coverage, which is a technique that measures how many executable statements in a source code are executed by a test suite. Statement coverage is not appropriate for testing the price computation, as it does not consider the input data or output results. Equivalence partitioning is not state transition testing, which is a technique that models how a system transitions from one state to another depending on events or conditions. State transition testing is not relevant for testing the price computation, as it does not involve any states or transitions. Equivalence partitioning is not use case testing, which is a technique that tests how users interact with a system to achieve a specific goal. Use case testing is not applicable for testing the price computation, as it does not focus on a single function or component. Verified Reference: [A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer], Chapter 4, page 37-38.

NEW QUESTION: 17

A test engineer finds a defect while testing. After the developer has fixed the defect, the test engineer decides to re-run a complete section of the tests. Which of the following is correct?

A. The lest engineer should not re-run the tests, as they have already been run, and results recorded.
B. The test engineer should not re-run the tests, they should be part of the developer tests.
C. The test engineer should re-run the tests, in order to ensure that new defects have not been introduced by the fix.
D. The test engineer should re-run the tests, because the defect shows that the test cases need to be updated.
Answer: C (LEAVE A REPLY)
The test engineer should re-run the tests, in order to ensure that new defects have not been introduced by the fix. This is also known as regression testing, which is a type of testing that verifies that previously tested software still performs correctly after a change. Regression testing helps to detect any side effects or unintended consequences of a fix or a modification. The other options are incorrect reasons for re-running the tests. The test engineer should not re-run the tests, as they have already been run, and results recorded, because this ignores the possibility of new defects caused by the fix. The test engineer should not re-run the tests, they should be part of the developer tests, because this assumes that developer tests are sufficient and reliable, which may not be true. The test engineer should not re-run the tests, because the defect shows that the test cases need to be updated, because this does not address the impact of the fix on other test cases or functionalities. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 41.

NEW QUESTION: 18
Software was found lo take much more time than the stated requirement of less than one second to save a file. Upon investigation it was found that there was an unnecessary check inside a loop which was slowing down the file-save operation. The software not being able to meet the desired response time is an example of
A. It is not a defect
B. Defect
C. Error
D. Failure
Answer: (SHOW ANSWER)
A failure is an event in which a component or system does not perform a required function within specified limits. A failure is observable by the software users or other stakeholders. A failure is caused by one or more defects in the software. In this case, the software not being able to meet the desired response time is an example of a failure, as it deviates from the stated requirement and affects the user experience. It is not a defect, which is a flaw in the software that causes the failure. It is not an error, which is a human action that produces an incorrect result. It is not a non-defect, as it clearly violates a specified requirement. Verified Reference: [A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer], Chapter 1, page 4.

NEW QUESTION: 19
An Incident Management tool implements the following defect states; Open, Assigned, Solved, Closed Consider the following defect report:
Id T000561
Test Object "Warehouse Management' application
Tester name; John Bishop
Date: 10th. April 2010
Test Case MRT558I
Status OPEN
Severity Serious
Priority
Problem- After inputting the Total Quantity item = 450 in the SV034 screen, the system shows an unexpected Error message=47 Correction:
Developer name:
Closing date:
Which of the following is a valid criticism of this report?
A. The Priority, the Correction description and the Developer name are missing
B. The version of the application is missing
C. There is no link to the applicable requirement (traceability)
D. The description is not highlighting the source of the problem
Answer: B (LEAVE A REPLY)
A valid criticism of this report is that the version of the application is missing. The version of the application is an important piece of information that should be included in a defect report, as it helps to identify which release or build of the software product contains the defect. The version of the application can also help to reproduce and debug the defect, as different versions may have different behaviors or features. The other options are not valid criticisms of this report. The priority, the correction description and the developer name are not missing, but rather not applicable for this report. The priority is a measure of how urgently a defect needs to be fixed, which can be assigned by the project manager or the defect tracking system, not by the tester who reports the defect. The correction description and the developer name are information that are added after the defect has been resolved, not when it has been reported. There is no link to the applicable requirement (traceability) is not a valid criticism of this report, because traceability is not a mandatory attribute of a defect report, but rather an optional one. Traceability is a relationship between two or more entities (such as requirements, test cases, defects, etc.) that shows how they are related or dependent on each other. Traceability can help to verify that the requirements are met by the test cases and defects, but it is not essential for reporting a defect. The description is not highlighting the source of the problem is not a valid criticism of this report, because highlighting the source of the problem is not a responsibility of the tester who reports the defect, but rather of the developer who fixes the defect. The description should provide

enough information to describe what happened when the defect occurred, such as input values, expected results, actual results, error messages, screenshots, etc., but it does not need to explain why or how it happened. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 140.

NEW QUESTION: 20
Which of the following tools is most likely to detect defects in functions or methods in source code?
A. configuration management tool
B. unit test framework tool
C. test design tool
D. monitoring tool
Answer: B (LEAVE A REPLY)
A unit test framework tool is a tool that supports the creation, execution, and reporting of unit tests, which are tests that verify the functionality and quality of individual software components (such as functions or methods) in source code. A unit test framework tool can help to detect defects in functions or methods in source code by providing features such as test case generation, test case execution, test result comparison, test coverage measurement, etc. Some examples of unit test framework tools are JUnit, NUnit, TestNG, etc. The other options are not tools that are likely to detect defects in functions or methods in source code. A configuration management tool is a tool that supports the management and control of different versions and variants of software products or components. A test design tool is a tool that supports the design and generation of test cases based on some criteria or rules. A monitoring tool is a tool that monitors the behavior or performance of a system or component under test. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 10.

NEW QUESTION: 21
A program got 100% decision coverage in a test. Which of the following statements is then guaranteed to be true?
A. Every executable statement Is covered.
B. Every output equivalence class has been tested.
C. Every input equivalence class has been tested.
D. The "dead" code has not been covered.
Answer: A (LEAVE A REPLY)
If a program got 100% decision coverage in a test, then it is guaranteed that every executable statement is covered. Decision coverage (also known as branch coverage) is a type of structural coverage (also known as white-box coverage) that measures how many decision outcomes have been exercised by a test suite. A decision outcome is a possible result of a decision point (such as an if-then-else statement) in a program's code. Decision coverage requires that each decision point has both true and false outcomes executed at

least once by a test suite. Decision coverage implies statement coverage, which is another type of structural coverage that measures how many executable statements have been executed by a test suite. Statement coverage requires that each executable statement is executed at least once by a test suite. Therefore, if a program got 100% decision coverage in a test, then it also got 100% statement coverage in a test, which means that every executable statement is covered. The other options are not guaranteed to be true if a program got 100% decision coverage in a test. Every output equivalence class has been tested and every input equivalence class has been tested are not guaranteed to be true if a program got 100% decision coverage in a test, because equivalence classes are based on functional requirements or specifications, not on code structure or logic. Equivalence classes are used in specification-based testing (also known as black-box testing), which is a type of testing that does not consider the internal structure or implementation of the system under test. Decision coverage is used in structure-based testing (also known as white-box testing), which is a type of testing that considers the internal structure or implementation of the system under test. Therefore, achieving 100% decision coverage does not imply achieving 100% equivalence class coverage. The "dead" code has not been covered is not guaranteed to be true if a program got 100% decision coverage in a test, because dead code (also known as unreachable code) is code that can never be executed due to logical errors or design flaws. Dead code can reduce readability and maintainability of the code, as well as increase complexity and size. Decision coverage does not account for dead code, as it only considers the decision outcomes that are possible to execute. Therefore, achieving 100% decision coverage does not imply that the dead code has not been covered. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 36.

NEW QUESTION: 22
The following sentences refer to the 'Standard for Software Test Documentation' specification (IEEE 829).
Which sentence is correct?
A. Any deviation from this standard should be approved by management, marketing & development
B. Most test documentation regimes follow this spec to some degree, with changes done to fit a specific situation or organization
C. The key to high quality test documentation regimes is strict adherence to this standard
D. This test plan outline is relevant for military projects. For consumer market projects there is a different specification with fewer items.
Answer: B (LEAVE A REPLY)
The IEEE 829 standard is a widely used specification for test documentation, but it is not mandatory or universal. Most test documentation regimes follow this spec to some degree, with changes done to fit a specific situation or organization. The standard does not require any approval from management, marketing or development for any deviation, nor does it

depend on the type of project (military or consumer market). The standard also does not guarantee high quality test documentation regimes, as it only provides a general outline and format, not the actual content or quality criteria. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 16.

NEW QUESTION: 23
Which of the following is a key characteristic of informal reviews?
A. Kick-off meeting
B. Low cost
C. Individual preparation
D. Metrics analysis
Answer: (SHOW ANSWER)
A key characteristic of informal reviews is low cost. Informal reviews are a type of review that does not follow a formal process or have any formal documentation. Informal reviews are usually performed by individuals or small groups of peers or colleagues who have some knowledge or interest in the product under review. Informal reviews can be done at any time and for any purpose, such as checking for errors, clarifying doubts, sharing ideas, etc. Informal reviews have low cost, as they do not require much time, effort, or resources to conduct. The other options are not key characteristics of informal reviews. Kick-off meeting is a characteristic of formal reviews, such as inspections or walkthroughs. Kick-off meeting is a meeting that is held before the review process starts, where the roles and responsibilities of the participants are defined, the objectives and scope of the review are agreed, and the logistics and schedule of the review are planned. Individual preparation is a characteristic of formal reviews, such as inspections or walkthroughs. Individual preparation is an activity that is performed by the reviewers before the review meeting, where they examine the product under review and identify any issues or questions that need to be discussed or resolved during the review meeting. Metrics analysis is a characteristic of formal reviews, such as inspections or walkthroughs. Metrics analysis is an activity that is performed after the review process is completed, where the data and results of the review are collected and analyzed to measure the effectiveness and efficiency of the review, as well as to identify any improvement actions or lessons learned for future reviews. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 9.

NEW QUESTION: 24
Which statement about use case testing is true?
A. The lest cases are designed to find defects in the data flew.
B. The test cases are designed to be used by real users, not by professional testers
C. The test cases are always designed by customers or end users.
D. The test cases are designed to find defects in the process flow.
Answer: (SHOW ANSWER)

Use case testing is a technique that helps identify test cases that exercise the whole system on a transaction by transaction basis from start to finish. Use cases are descriptions of how users interact with the system to achieve a specific goal. Use case testing is not focused on data flow, but rather on process flow. Use case testing can be performed by professional testers, customers or end users, depending on the context. Use case testing does not require the test cases to be designed by customers or end users, but rather by anyone who has access to the use case specifications. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, Chapter 4, page 36.

NEW QUESTION: 25
Which of the following can be considered a VALID exit criterion?
I Estimates of defect density or reliability measures.
II. The completion and publication of an exhaustive Test Report.
III. Accuracy measures, such as code, functionality or risk coverage.
IV Residual risks such as lack of code coverage in certain areas.
A. I, III, IV
B. I, II, III
C. III, IV
D. II, III, IV
Answer: A (LEAVE A REPLY)
An exit criterion is a condition that defines when a test activity has been completed or when a test phase can be concluded. An exit criterion can be based on various factors, such as:
I) Estimates of defect density or reliability measures. These are quantitative measures that indicate how many defects are present in the software product or how likely it is to fail under certain conditions. These can be used as exit criteria to ensure that the software product meets a certain level of quality or performance before moving to the next phase or releasing it to the customer.
III) Accuracy measures, such as code coverage, functionality coverage or risk coverage. These are quantitative measures that indicate how much of the software product has been tested in terms of its code, functionality or risk. These can be used as exit criteria to ensure that the test suite is adequate or complete before moving to the next phase or releasing it to the customer.
IV) Residual risks, such as lack of code coverage in certain areas, unresolved defects or unknown factors. These are qualitative measures that indicate the remaining risks or uncertainties associated with the software product after testing. These can be used as exit criteria to ensure that the residual risks are acceptable or manageable before moving to the next phase or releasing it to the customer. The following factor is not a valid exit criterion:
II) The completion and publication of an exhaustive Test Report. This is not a valid exit criterion, as it does not reflect the quality or completeness of the testing process or

product. A test report is a document that summarizes the results and outcomes of a test activity or phase. A test report can be used as an input for deciding whether to exit a test activity or phase, but it is not a condition that defines when to exit. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, Chapter 2, page 13; Chapter 6, page 58-59.

NEW QUESTION: 26
Which of the following is an example of black-box dynamic testing?
A. Functional Testing
B. Code inspection
C. Checking memory leaks for a program by executing it
D. Coverage analysis
Answer: (SHOW ANSWER)
Functional testing is an example of black-box dynamic testing. Black-box testing (also known as specification-based testing) is a type of testing that does not consider the internal structure or implementation of the system under test, but rather its external behavior or functionality. Dynamic testing is a type of testing that involves executing the system under test with various inputs and observing its outputs. Functional testing is a type of black-box dynamic testing that verifies that the system under test performs its intended functions according to its requirements or specifications. Functional testing can be performed at various levels and scopes depending on the objectives and criteria of testing. The other options are not examples of black-box dynamic testing. Code inspection is an example of white-box static testing. White-box testing (also known as structure-based testing) is a type of testing that considers the internal structure or implementation of the system under test. Static testing is a type of testing that does not involve executing the system under test, but rather analyzing it for defects, errors, or violations of standards. Code inspection is a type of white-box static testing that involves examining the source code of the system under test for quality, readability, maintainability, etc. Checking memory leaks for a program by executing it is an example of white-box dynamic testing. Memory leaks are defects that occur when a program fails to release memory that it has allocated but no longer needs. Checking memory leaks for a program by executing it requires knowledge and access to the internal structure or implementation of the program, such as memory allocation and deallocation mechanisms, pointers, references, etc. Coverage analysis is an example of white-box static testing. Coverage analysis is a technique that measures how much of the code or structure of the system under test has been exercised by a test suite. Coverage analysis requires knowledge and access to the internal structure or implementation of the system under test, such as statements, branches, paths, conditions, etc. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 7.

NEW QUESTION: 27

Which type of software development product can undergo static testing?
A. Any software development product can undergo static testing, including requirements specifications, design specifications and code.
B. Static tests should be performed on the installation and user guide documents as these documents are used by the end user.
C. Static testing is done only on the code as part of the "code review" sessions Other documents are reviewed, but not by static testing.
D. Static testing is done only on the requirements You need to execute the software in order to find defects in the code.
Answer: A (LEAVE A REPLY)
Static testing is a form of testing that does not involve executing the software, but rather analyzing it for defects, errors, or violations of standards. Static testing can be applied to any software development product, including requirements specifications, design specifications, code, test cases, test plans, user manuals, etc. Static testing can be done by using various techniques such as reviews, inspections, walkthroughs, checklists, static analysis tools, etc. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, page 7.

NEW QUESTION: 28
Which of the following statements about re-testing and regression testing are TRUE?
I Re-testing should be performed after a defect is fixed.
II Regression testing should always be performed after a defect is fixed.
III Re-testing and regression testing may be performed at any test level.
IV Regression testing may include functional, non-functional and structural testing.
V Re-testing should be included in the debugging activity.
A. I, III. Iv
B. II, v
C. I, III
D. II. IV, V
Answer: A (LEAVE A REPLY)
The following statements about re-testing and regression testing are true:
I) Re-testing should be performed after a defect is fixed. Re-testing is a type of testing that verifies that a defect has been successfully resolved by executing a test case that previously failed due to that defect. Re-testing should be performed after a defect is fixed and delivered to ensure that it does not cause any new failures or side effects.
III) Re-testing and regression testing may be performed at any test level. Re-testing and regression testing are not limited to a specific test level, but can be applied at any level depending on the context and objectives. For example, re-testing and regression testing can be performed at unit level, integration level, system level or acceptance level.
IV) Regression testing may include functional, non-functional and structural testing. Regression testing is a type of testing that verifies that previously tested software still

performs correctly after changes. Regression testing may include various types of testing depending on the scope and purpose of the changes. For example, regression testing may include functional testing to check if the software meets its requirements, non-functional testing to check if the software meets its quality attributes, or structural testing to check if the software meets its design or code standards. The following statement about re-testing and regression testing is false:
II) Regression testing should always be performed after a defect is fixed. Regression testing is not always necessary after a defect is fixed, as some defects may have a low impact or low likelihood of affecting other parts of the software. Regression testing should be performed after a defect is fixed only if there is a risk of introducing new defects or causing existing defects due to the changes made to fix the defect. Verified Reference: A Study Guide to the ISTQB Foundation Level 2018 Syllabus - Springer, Chapter 2, page 19; Chapter 4, page 45.