DUMPSARENA

ISTQB Certified Tester Foundation Level

iSQI CTFL_Foundation

Version Demo

Total Demo Questions: 27

Total Premium Questions: 270

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

Topic Break Down

Topic	No. of Questions
Topic 1, Fundamentals of Testing	37
Topic 2, Testing Throughout the Software Life Cycle	39
Topic 3, Static Techniques	22
Topic 4, Test Design Techniques	81
Topic 5, Test Management	66
Topic 6, Tool Support for Testing	25
Total	270

QUESTION NO: 1

Which of the following is the most correct statement about static testing techniques?

A. Static techniques find more defects than dynamic techniques.

B. Static techniques can be used before all code is ready for execution.

C. Static techniques are always cheaper than dynamic techniques.

D. Static techniques can be used by inexperienced users.

ANSWER: B

Explanation:

Static techniques can be used before all code is ready for execution. Static testing examines work products without executing the software: for example, requirements, user stories, architecture or design documents, test plans, source code, and test cases. Reviews, walkthroughs, inspections, and static analysis are common forms of static testing. Because these activities evaluate artifacts directly rather than running the application, they can begin as soon as a relevant artifact is available. A requirements review can therefore identify ambiguity, inconsistency, omissions, or non-testable requirements well before implementation is complete. Similarly, a code review or static-analysis activity can inspect code that has been written even if the full system cannot yet be built or executed. This early feedback is a central benefit of static testing: defects can be detected and addressed close to the point where they were introduced, helping prevent those defects from propagating into later development and testing work. The ISTQB Foundation Level syllabus describes static testing as evaluating work products without executing them and recognizes reviews and static analysis as key approaches. See the ISTQB Certified Tester Foundation Level information and the ISTQB glossary definition of static testing.

QUESTION NO: 2

A booking system for a city bus service prices its fares according to the time of travel:

- Peak-time tariff starts at 0600 and finishes at 1000 am
- Off-peak tariff applies during all other times of service
- The bus service does not operate between 2300 and the start of the next day's peak service

Note that all times mentioned are inclusive.

When applying the equivalence partitioning test design technique, which of the following options shows test case inputs that each fall into a different equivalence partition?

A. 0600, 1000, 1200

B. 1001, 1300, 2259

C. 0100, 0800, 2200

D. 2400, 1000, 2301

ANSWER: C

Explanation:

0100, 0800, 2200 is correct because it selects one representative time from each distinct input class for the booking system's tariff and availability rules. Equivalence partitioning divides the input domain into partitions for which the system is expected to behave equivalently; one test value from each partition is normally sufficient to represent that class. In this case, the relevant partitions are the period when the service does not operate, from 2300 through 0559; the peak-time service period, from 0600 through 1000 inclusive; and the off-peak service period, from 1001 through 2259. The value 0100 represents the unavailable-service partition, 0800 represents the peak-time tariff partition, and 2200 represents the off-peak tariff partition. Thus, these three inputs exercise each separate business outcome implied by the stated rules: no booking/service, peak fare, and off-peak fare. This is the intended use of equivalence partitioning: choose representatives

from disjoint classes whose members should be processed in the same way. The Foundation Level syllabus identifies equivalence partitioning as a black-box test technique for partitioning data into groups expected to be handled similarly. See the ISTQB Certified Tester Foundation Level certification information and the ISTQB glossary definition of equivalence partitioning.

QUESTION NO: 3

Which of the following options BEST explain the pesticide paradox principle of testing?

A. If we do not regularly review and revise our tests, we'll stop finding defects.

B. Repeatedly running a set of tests will ensure that a system is defect free.

C. Defects are, paradoxically, often contained in a small number of modules.

D. Testing, like spraying pesticide, is an effective bug / defect removal activity.

ANSWER: A

Explanation:

If we do not regularly review and revise our tests, we'll stop finding defects. correctly expresses the pesticide paradox. The principle compares testing to the repeated use of a pesticide: after the same treatment is applied over and over, the remaining pests may no longer be affected by it. Similarly, a test suite that is run repeatedly without change becomes increasingly effective only at checking the behaviours and defect types it was designed to detect. It can continue to provide useful regression confidence for those known areas, but it is less likely to reveal new defects, defects caused by changed requirements or implementation, or defects in scenarios not represented by the existing tests.

Testers should therefore periodically examine test conditions, test data, expected results, execution techniques, and coverage. They should revise existing tests where the product or its risks have changed, and add new tests that target newly identified risks, different combinations of inputs, and previously untested behaviours. This keeps the test approach capable of detecting defects rather than merely repeatedly demonstrating already-established product behaviour. The ISTQB glossary defines the pesticide paradox as the decreasing effectiveness of repeatedly executing the same tests: ISTQB Glossary: pesticide paradox. This principle is also part of the Foundation Level testing principles described by ISTQB Certified Tester Foundation Level.

QUESTION NO: 4

Which statement describes the term “Statement Coverage”?

A. A tool used to execute all the statements of the source code.

B. A method by which complex statements of the source code are completely executed.

C. A metric for the ratio of statements in a source code that are executed at least once during the test.

D. A test technique that generates a minimal set of tests cases guaranteed to exercise all the lines in source code.

ANSWER: C

Explanation:

“A metric for the ratio of statements in a source code that are executed at least once during the test.” correctly describes statement coverage. Statement coverage is a white-box, structure-based coverage measure that reports the proportion of executable statements exercised at least once by a test suite. It is commonly calculated as the number of executable statements executed divided by the total number of executable statements, expressed as a percentage. For example, if a program contains 100 executable statements and the tests execute 85 of them, statement coverage is 85%.

This measure helps testers assess how much of the code’s statement-level structure has been exercised and identify statements that have not yet run during testing. The CTFL syllabus places statement testing and statement coverage within white-box test techniques, where tests are designed from the internal structure of the component or system. “At least once”

is essential: coverage concerns whether each executable statement has been reached and executed, not how many times it ran. The ISTQB glossary defines statement coverage as the percentage of executable statements exercised by a test suite. See the ISTQB Glossary entry for statement coverage and the ISTQB Certified Tester Foundation Level materials.

QUESTION NO: 5

A company wants to enforce use of coding standards.

Which type of test tool would you recommend for this purpose?

A. Test execution tool

B. Modeling tool

C. Configuration management tool

D. Static analysis tool

ANSWER: D

Explanation:

A **Static analysis tool** is the appropriate choice because coding standards can be checked directly against source code without executing the software. Static analysis tools inspect code for defined rules and conventions, such as naming patterns, formatting, prohibited language constructs, complexity thresholds, duplicated code, and potential defects. A team can configure the tool with its required coding-standard rules and run it regularly, for example locally during development or automatically as part of a continuous-integration build. This provides fast, consistent, repeatable feedback and makes adherence to the agreed standard measurable across the codebase.

The Foundation Level syllabus identifies static analysis as an activity that can detect violations of programming standards and other maintainability concerns early in the development process. Enforcing standards at this point supports prevention: developers receive actionable findings before code proceeds to later test stages or release. It also helps maintain a uniform codebase, reducing the effort required to understand, review, modify, and maintain the software over time. See the ISTQB Certified Tester Foundation Level certification information and the Sonar static code analysis overview for further background on automated source-code rule checking.

QUESTION NO: 6

An Incident Management tool implements the following defect state: 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: MRT5561

Status: 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 description is not highlighting the source of the problem

C. The version of the application is missing

D. There is no link to the applicable requirement (traceability)

ANSWER: C

Explanation:

The version of the application is missing is the valid criticism. A defect report must contain enough information for the issue to be reproduced, investigated, fixed, and later verified. Identifying the test object merely as the “Warehouse Management” application does not establish the specific build, release, or configuration in which error message 47 occurred. Different versions of the same application can contain different code, database schemas, interfaces, configuration settings, and already-applied fixes. Without a version identifier, a developer or test team may be unable to determine whether the reported behavior applies to the build currently under investigation or whether it has already changed in another release.

The report does provide useful reproduction information, including the relevant screen, the input value, the observed result, the reporting tester, date, and associated test case. However, this information needs to be tied to a precise configuration item or application version to make the incident actionable and traceable through its lifecycle. Foundation-level testing guidance treats configuration and defect information as important test-work-product controls, because reproducibility depends on knowing the exact item under test. See the ISTQB Certified Tester Foundation Level overview and the ISTQB glossary entry for “defect report”.

QUESTION NO: 7

Under performance testing, some additional print statements are added in the code which record the time of execution of each function call as well as every loop. The observation is that the software performs better without those additional print statements.

What is the name of the effect where the measurement methods and tools change the outcome of the test?

A. Probe effect

B. Measurement error effect

C. Pesticide paradox effect

D. Performance degradation effect

ANSWER: A

Explanation:

Probe effect is the correct term because the act of observing or measuring the system changes the system’s behavior. Here, the added print statements are instrumentation: each statement performs extra work, such as formatting output, writing to a console or log destination, synchronizing access to that destination, and consuming processor time or I/O capacity. Consequently, the timings observed while the instrumentation is enabled do not necessarily represent the application’s normal performance. This is particularly relevant in performance testing, where even seemingly small additions inside frequently executed functions or loops can accumulate and alter response time, throughput, or resource utilization.

The test result therefore reflects both the software under test and the overhead imposed by the measurement mechanism. A tester should recognize this risk when choosing profiling, tracing, logging, and monitoring approaches, and should use minimally intrusive instrumentation where practical. The ISTQB glossary defines the probe effect as the effect on the performance of a component or system caused by the measurement instrument. See the ISTQB Glossary: probe effect and the ISTQB Certified Tester Foundation Level certification information.

QUESTION NO: 8

During which stage of the fundamental test process is the testability of requirements evaluated?

A. Test Implementation and Execution

B. Test Planning and Control

C. Evaluating Exit Criteria and Reporting

D. Test Analysis and Design

ANSWER: D

Explanation:

Test Analysis and Design is the stage in which the test basis—such as requirements, functional specifications, and system architecture—is examined to identify testable conditions and to determine what needs to be tested. Evaluating requirement testability is therefore an essential activity at this point. A requirement is testable when it is stated clearly enough that objective test conditions, test cases, expected results, and pass/fail criteria can be derived from it. This review also helps identify ambiguities, omissions, inconsistencies, and statements that cannot be verified effectively before substantial test implementation effort is invested.

The ISTQB Foundation syllabus describes test analysis and design as the activity that reviews the test basis for testability and identifies test conditions. Assessing testability early supports a more effective test process because testers can raise defects or questions about requirements while they are still relatively inexpensive to resolve. The resulting test conditions then provide the foundation for later work, including creating test procedures, preparing test data, and executing tests. See the ISTQB Certified Tester Foundation Level certification overview and the ISTQB CTFL syllabus resources.

QUESTION NO: 9

While reporting a defect, which of the following indicates the level of business importance assigned to the defect?

A. Urgency

B. Priority

C. Difficulty

D. Severity

ANSWER: B

Explanation:

Priority indicates the level of business importance assigned to a defect and helps determine the order in which the defect should be addressed. It reflects the stakeholder, customer, or business perspective: a defect may be given high priority when it affects an important business process, a planned release, contractual commitments, or a highly visible feature. Priority is therefore used in defect triage and planning to decide when a fix should be delivered relative to other work.

In the ISTQB Foundation Level terminology used by iSQI, defect reports support communication about anomalies and their management. Relevant defect-report information can include an indication of the defect's impact on stakeholder interests, helping the appropriate people make informed decisions about resolution and release planning. Assigning priority is not merely a technical assessment; it aligns defect handling with product risk and business needs. A defect report should consequently record priority clearly enough that the team can schedule corrective action effectively and transparently. See the ISTQB Certified Tester Foundation Level certification information and the ISTQB glossary definition of priority.

QUESTION NO: 10

Which of the following represents good testing practice for testers, irrespective of the software lifecycle model used?

A. They should start test analysis when the corresponding development level is complete.

B. They should be involved in reviewing requirements or user stories as soon as drafts are available.

C. They should ensure that the same test objectives apply to each test level.

D. They should minimize the ratio of development levels to test levels to reduce project costs.

ANSWER: B

Explanation:

"They should be involved in reviewing requirements or user stories as soon as drafts are available." represents good testing practice because testing is most effective when it begins early, rather than being deferred until executable software is available. Early tester involvement allows requirements, user stories, and acceptance criteria to be examined for testability, completeness, consistency, clarity, and ambiguity. Testers can use their quality perspective to identify potential defects or missing information while changes are still comparatively inexpensive and straightforward to make.

This practice is independent of whether a project follows a sequential, iterative, incremental, or Agile lifecycle. In every lifecycle, some form of requirements or user needs is defined, and early review provides prompt feedback that improves the shared understanding of intended system behavior. It also enables testers to prepare test conditions and test ideas earlier, supports better estimation and planning, and helps the team prevent defects rather than only detect them later. The CTFL syllabus emphasizes that early testing activities, including reviews of requirements and other work products, contribute to finding defects early and reducing the cost of quality. See the ISTQB Certified Tester Foundation Level overview and the ISTQB glossary entry for early testing.

QUESTION NO: 11

Which of the following statements about test reports are TRUE?

I. Test reports shall be approved by the test team.

II. Test reports shall give stakeholders information as basis for decisions.

III. Test reports shall summarize what happened through a period of testing.

IV. Test reports shall be approved by the development team, the test team and the customer.V. Test reports shall include information about remaining risks.

A. I, II, IV

B. II, III, IV

C. I, III, V

D. II, III, V

E. I, II, III, IV, V

ANSWER: D

Explanation:

"II, III, V" is correct because a test report, called a test summary report in the Foundation Level syllabus terminology, communicates the outcome and status of a testing period to the relevant stakeholders. Its purpose is not merely to record execution figures; it supplies meaningful information that stakeholders can use when making release, acceptance, deployment, or further-testing decisions. A useful report therefore summarizes the testing work that occurred during the reporting period, including the activities performed and their results.

Crucially, decision-makers also need visibility of residual risk: the risks that remain after the testing performed, whether due to unresolved defects, incomplete coverage, test-environment constraints, or areas that could not be tested. Reporting those remaining risks supports an informed decision about whether the product is fit for its intended use and whether further action is necessary. This aligns with the Foundation Level view of test reporting as a communication and decision-support activity.

The ISTQB glossary defines the related test summary report as documentation that summarizes testing activities and results, while the Foundation Level certification materials describe reporting as a core test-management activity: ISTQB Glossary: test summary report and ISTQB Certified Tester Foundation Level.

QUESTION NO: 12

	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6
Conditions:						
Car Driver	No	Yes	Yes	Yes	Yes	No
Motorcycle Driver	No	No	No	No	No	Yes
Diesel	N/A	Yes	Yes	No	No	N/A
Petrol	N/A	No	No	Yes	Yes	N/A
Engine < 1600cc	N/A	Yes	No	Yes	No	N/A
Engine >1600cc	N/A	No	Yes	No	Yes	N/A
Actions:						
Can claim expenses	No	Yes	Yes	Yes	Yes	Yes
Expenses claim band A	N/A	Yes	No	No	No	Yes
Expenses claim band B	N/A	No	Yes	No	No	No
Expenses claim band C	N/A	No	No	Yes	No	No
Expenses claim band D	N/A	No	No	No	Yes	No

The decision table above shows a company's fuel expenses structure.

Which of the following Test Cases based on the decision table are Valid?

Test Case 1:

An employee who is not a car or motorcycle driver attempts to claim fuel expenses. Expected result: Expense claim not allowed.

Test Case 2:

An employee who drives a 1700cc diesel car attempts to claim fuel expenses. Expected result: Expense claim accepted at band C.

Test Case 3:

An employee who rides a motorcycle attempts to claim fuel expenses. Expected result: Expense claim accepted at band A.

A. Test Cases 1 and 3 are Valid. Test Case 2 is Invalid.

B. Test Cases 2 and 3 are Valid. Test Case 1 is Invalid.

C. Test Cases 1, 2 and 3 are all Valid.

D. Test Case 2 is Valid. Test Cases 1 and 3 are Invalid.

ANSWER: A

Explanation:

Test Cases 1 and 3 are Valid. Test Case 2 is Invalid. This result follows by matching each test case's conditions to a single rule in the fuel-expenses decision table and then checking that the stated expected result is the action associated with

that rule. A person who is neither a car driver nor a motorcycle rider meets the table's condition for an ineligible claimant, so "Expense claim not allowed" is the correct expected outcome. A motorcycle rider satisfies the motorcycle condition, and the table assigns this case to band A; therefore, the stated outcome for the motorcycle test is valid.

Decision-table testing is a black-box technique that derives tests from combinations of conditions and the corresponding actions. A sound test case must represent a feasible rule and must specify the exact action defined by that rule. This is why checking both the input combination and the expected result is essential, rather than considering only whether the employee can make a claim. The ISTQB Foundation Level syllabus describes decision table testing as a technique for testing business rules and combinations of conditions: ISTQB Certified Tester Foundation Level. The iSQI certification overview also identifies CTFL as an examination based on the ISTQB Foundation Level framework: iSQI CTFL.

QUESTION NO: 13

A software company decided to buy a commercial application for its accounting operations. As part of the evaluation process, the company decided to assemble a team to test a number of candidate applications.

Which team would be the most suitable for this goal?

A. A team from an outsourcing company which specializes in testing accounting software

B. A team with a mix of software testers and experts from the accounting department

C. A team of users from the accounting department that will need to use the application on daily basis

D. A team from the company's testing team, due to their experience in testing software

ANSWER: B

Explanation:

A team with a mix of software testers and experts from the accounting department is the most suitable team for evaluating candidate commercial accounting applications. This evaluation must establish both that each product performs reliably and that it meets the organization's actual accounting needs. Accounting specialists contribute essential domain knowledge: they understand the company's business processes, statutory and reporting obligations, workflows, terminology, data requirements, and the practical tasks that daily users must complete. Their participation ensures that representative business scenarios and acceptance criteria are meaningful.

Testers contribute complementary expertise in planning a structured evaluation, designing effective test cases, preparing suitable test data, executing tests systematically, recording evidence, and reporting defects or risks consistently across all candidate products. Their skills help make the comparison objective and repeatable rather than based only on informal impressions. Combining these perspectives is particularly important for commercial off-the-shelf software, where the buyer must assess fitness for purpose and identify configuration, integration, usability, and process risks before committing to a product. This aligns with the Foundation Level emphasis on involving appropriate stakeholders in acceptance-oriented testing and evaluating whether a system satisfies user and business needs. See the ISTQB Certified Tester Foundation Level overview and the Foundation Level syllabus resources.

QUESTION NO: 14

Testing should provide sufficient information to stakeholders to make informed decisions about the release of the software or system being tested. At which of the following fundamental test process activity the sufficiency of the testing and the resulting information are assessed?

A. Implementation and execution

B. Requirements specification

C. Evaluating exit criteria and reporting

D. Analysis and design

ANSWER: C

Explanation:

The correct answer is **Evaluating exit criteria and reporting**. In the ISTQB Foundation Level fundamental test process, this activity determines whether testing has provided enough evidence to support a release-related decision. The test team evaluates the actual results against defined exit criteria, which may include the planned level of test coverage, completion of scheduled tests, defect status and severity, residual risk, and any other measures agreed in the test plan. This assessment is not simply a count of executed tests; it establishes whether the available test information is sufficiently complete, reliable, and relevant for stakeholders to judge the product's release readiness.

The activity also includes preparing a test summary report for stakeholders. That report communicates what was tested, the results obtained, deviations from the plan, unresolved defects, and remaining risks. Stakeholders can then use this consolidated evidence to make an informed decision to release, defer release, accept known risks, or request further testing. This directly matches the purpose stated in the question: assessing both the sufficiency of testing and the usefulness of its resulting information. See the ISTQB Certified Tester Foundation Level overview and the ISTQB glossary entry for exit criteria.

QUESTION NO: 15

In maintenance testing, what is the relationship between impact analysis and regression testing?

A. Impact analysis requires a regression testing for all program elements which were newly integrated (new functionalities).

B. Impact analysis requires a regression testing for only the tests that have detected faults in previous SW release.

C. There is no relationship between impact analysis and regression testing.

D. The impact analysis is used to evaluate the amount of regression testing to be performed.

ANSWER: D

Explanation:

"The impact analysis is used to evaluate the amount of regression testing to be performed." is correct. Maintenance testing is performed after a change to an existing system, such as a defect correction, enhancement, migration, or retirement activity. Before deciding the scope of testing, the team performs impact analysis to identify the parts of the system, interfaces, data, documentation, and test assets that may be affected directly or indirectly by that change. This analysis provides a risk-based basis for selecting and prioritizing regression tests. It helps the test team determine how much previously completed testing should be repeated to obtain sufficient confidence that unchanged functionality has not been adversely affected. The appropriate regression-test scope is therefore not automatically every existing test; it is determined by the change's potential impact, the affected areas' criticality, dependencies, and associated risks. This relationship is specifically recognized in the ISTQB Foundation Level syllabus discussion of maintenance testing, where impact analysis supports deciding the extent of regression testing needed after modifications. See the ISTQB Certified Tester Foundation Level certification materials and the ISTQB Glossary definition of impact analysis.

QUESTION NO: 16

The following condition is given:

Integer x, y;

IF x > 0 AND x < 100 y=y+x;

END-IF

Using boundary analysis for x, which test cases are required?

A. -1, 0, 1, 99, 100, 101

B. -1, 0, 100, 101

C. -500, -10, 0, 1, 99, 100, 101, 500

D. 0, 1, 99, 100

ANSWER: A

Explanation:

The required test values are **-1, 0, 1, 99, 100, 101**. Boundary value analysis derives tests from the edges of an input domain because faults frequently occur at limits, including errors in relational operators and off-by-one implementation mistakes. Here, the condition accepts values strictly greater than 0 and strictly less than 100. Therefore, 0 is the lower boundary and 100 is the upper boundary. For each boundary, the test set includes the boundary value itself and the immediately adjacent integer on each side. Around the lower boundary, these are -1, 0, and 1. Around the upper boundary, they are 99, 100, and 101. This provides coverage of the transition into and out of the range for which the IF body is executed, while also checking the exact excluded endpoints implied by the strict greater-than and less-than comparisons. The use of adjacent integer values is appropriate because x is declared as an integer. Boundary value analysis is a black-box test-design technique described in the Foundation Level syllabus; it focuses on boundaries of equivalence partitions and values immediately on either side of them. See the ISTQB Certified Tester Foundation Level certification overview and iSQI CTFL certification information.

QUESTION NO: 17

Which of the following test types are non-functional tests?

I) Acceptance test

II) Regression test

III) Stress test

IV) Component test

V) Reliability test

A. II, III and V

B. I, II and IV

C. I, III and V

D. III and V

ANSWER: D

Explanation:

III and V is correct because stress testing and reliability testing assess quality characteristics rather than whether individual functions produce specified business results. Stress testing evaluates system behaviour at, or beyond, anticipated operational limits, such as exceptionally high workload, constrained resources, or heavy transaction volume. It is therefore a form of performance-related non-functional testing. Reliability testing evaluates the ability of a system to perform required functions consistently for a stated period under stated conditions. Typical reliability concerns include failure frequency, availability, recoverability, and stability over time.

The Foundation Level syllabus classifies non-functional testing by the quality attributes of a component or system, including performance efficiency and reliability. This classification focuses on how well the system operates, not on a particular development test level or on retesting after a change. Stress and reliability tests directly provide evidence about these non-functional characteristics and help stakeholders assess operational risk before release. The terminology is consistent with the ISTQB glossary definitions for stress testing and reliability testing.

QUESTION NO: 18

Which of the following is the BEST example of an exit criterion for system testing?

A. The test environment has been installed.

B. All planned high-priority tests have been executed and no open critical defects remain.

C. Developers have completed coding all system components.

D. The test team has been assigned to the project.

ANSWER: B

Explanation:

All planned high-priority tests have been executed and no open critical defects remain. is correct because exit criteria define conditions that must be met, or assessed, before a test level can be considered complete. Test execution status and the severity of outstanding defects provide relevant evidence for a decision about completing system testing.

An exit criterion should be measurable and related to the agreed test objectives, risks, coverage, and quality targets. The date on which testing starts or the number of developers assigned are planning considerations, not exit criteria.

QUESTION NO: 19

You are testing an e-commerce system. The system accepts four different types of Credit Cards; each card has its own rules for valid and invalid numbers.

Condition			
Valid Card Number	NO	YES	YES
Purchase Approved	NO	NO	YES
Action			
Reject message	YES	YES	NO
Accept Payment	NO	NO	YES

The combination of the conditions Valid Card Number (NO) + Purchase Approved (YES) is not feasible.

You wish to perform a test which completely covers all the combinations of equivalence partitioning for the different types of credit card, according to the rules shown in the above Decision Table.

How many test cases do you need?

A. 8

B. 4

C. 16

D. 12

ANSWER: D

Explanation:

12 is correct because complete coverage requires considering each of the four supported credit-card types independently, since each type has its own valid and invalid card-number equivalence partitions. For any one card type, there are two card-number partitions: valid and invalid. A valid number can lead either to an approved purchase or to a declined purchase, giving two feasible combinations. An invalid number can only lead to a non-approved purchase, because the question

explicitly states that an invalid card number together with an approved purchase is infeasible. Therefore, each card type has three feasible combinations to test: valid/approved, valid/not approved, and invalid/not approved.

Applying those three feasible combinations to all four card types gives 4 × 3 = 12 test cases. This is a decision-table-style derivation: test cases correspond to the feasible rules, while impossible condition combinations are excluded rather than tested. Equivalence partitioning supports selecting one representative value from each identified valid or invalid input class, and the decision table ensures that the relevant outcome combinations are covered. These techniques are part of the Foundation Level test-design knowledge area described by ISTQB Certified Tester Foundation Level and its ISTQB glossary definition of equivalence partitioning.

QUESTION NO: 20

A company that created software for managing libraries won a contract to write an application for managing inventories in a large hospital pharmacy.

The test manager decided that the test policy and all work procedures must be reviewed and updates to meet this new challenge.

Which of the following is the best explanation for this decision?

A. Large organizations have bigger budgets and hence strict testing is required

B. The size of the software for the new project is different than previous ones. This requires more testing.

C. Revising the test policy and work procedures need not be done at the beginning of each project

D. Different industries have different risks and quality requirements. This impacts the test process.

ANSWER: D

Explanation:

Different industries have different risks and quality requirements. This impacts the test process. A hospital-pharmacy inventory application operates in a domain where incorrect stock data, unavailable medicines, incorrect product identification, expiry-date failures, or inadequate access control can affect patient safety and regulatory compliance. These risks differ materially from those associated with library-management software. Consequently, the organization should review its test policy and procedures to ensure that they remain appropriate for the new business domain, its risk level, applicable regulations, required test techniques, test environments, test data controls, traceability, and reporting needs.

In the Foundation Level syllabus, a test policy defines the organization’s testing principles and objectives, while test processes and procedures must be tailored to the project context. Test planning is risk-based: the level and type of testing are influenced by product risks, business-criticality, and quality characteristics. Moving into a healthcare-related context is therefore a sound reason to reassess how testing will be governed and performed. See the ISTQB Certified Tester Foundation Level overview and the European Medicines Agency guidance on good manufacturing practice.

QUESTION NO: 21

A test manager needs to estimate the effort for testing a new release of an existing application. The previous release had similar functionality and its test effort, defect history, and rework effort were recorded.

Which estimation approach is MOST appropriate to use as a basis for the estimate?

A. Using historical data from the previous similar release.

B. Estimating only the time required to execute the planned test cases.

C. Assigning the same effort to every requirement regardless of risk.

D. Using the planned development duration as the test effort estimate.

ANSWER: A

Explanation:

Using historical data from the previous similar release. is correct because estimation can be improved by using information from comparable completed projects or releases. Historical effort, defect, and rework data provide evidence that can be adjusted for known differences in scope, risk, technology, and resources.

Although the estimate should also consider the current release's risks and constraints, ignoring relevant historical information would reduce the reliability of the estimate. Test estimation is part of test planning and supports decisions about resources, schedule, and test scope.

QUESTION NO: 22

A system is being enhanced to simplify screen navigation for users.

Which of the following does NOT reflect structural testing?

A. To test all paths that users could take through the screen menu system

B. To ensure that 100% decision testing is achieved for each system component

C. To test all branches of component calls within the application call graph

D. To ensure that users can navigate to all fields on the screen

ANSWER: D

Explanation:

“To ensure that users can navigate to all fields on the screen” is the correct answer because it checks an externally observable user-oriented behaviour: whether the interface permits access to every field needed to perform work. This is a functional, and potentially usability-related, test objective. Its basis is the specified behaviour and user interaction with the screen, rather than the internal implementation of the software.

Structural testing derives test cases from the internal structure of a component or system. Typical structural coverage measures concern elements such as statements, decisions, branches, conditions, paths, or control flow. For screen navigation, a test concerned with users reaching fields is focused on the outcome experienced by the user, regardless of the particular menu logic, call graph, or decision structure used to implement it. The test may reveal a defect in that implementation, but it does not select tests by measuring or exercising internal structural coverage.

This distinction between specification-based and structure-based techniques is part of the Foundation Level testing-techniques material. See the ISTQB Certified Tester Foundation Level overview and the CTFL syllabus resources.

QUESTION NO: 23

A tester executes a test suite containing 80 test cases. Sixty test cases have passed, 10 have failed, and 10 have not yet been executed.

Which of the following is the test execution progress expressed as a percentage?

A. 75%

B. 80%

C. 87.5%

D. 100%

ANSWER: C

Explanation:

87.5% is correct. Test execution progress is calculated from the number of test cases executed divided by the total number planned. In this case, 60 passed plus 10 failed means that 70 tests have been executed. Therefore, 70 divided by 80 gives 87.5%.

The percentage of passed test cases is 75% of the planned total, but this is not the execution-progress measure asked for. Failed tests are still executed tests and must be included when measuring execution progress. Such measures support test monitoring and control. See the ISTQB Glossary for test monitoring terminology.

QUESTION NO: 24

What is the difference between system integration testing and acceptance testing?

A. System integration testing is testing non-functional requirements.
Acceptance testing concentrates on the functionality of the system.

B. System integration testing is executed by the developers. Acceptance testing is done by the customer.

C. System integration testing verifies that a system interfaces correctly with other systems. Acceptance testing verifies compliance to requirements.

D. System integration testing verifies compliance to requirements.
Acceptance testing verifies correct interaction with other systems existing in the user's environment.

ANSWER: C

Explanation:

"System integration testing verifies that a system interfaces correctly with other systems. Acceptance testing verifies compliance to requirements." is correct because the two test levels have distinct objectives. System integration testing addresses the interfaces and interactions between the system under test and external systems or services. Its purpose is to establish confidence that exchanged data, protocols, interface behavior, and end-to-end interactions work correctly across system boundaries. This testing is especially important where the system relies on connected applications, third-party services, hardware, or operational infrastructure.

Acceptance testing, in contrast, is focused on whether the system is fit for use and can be accepted by its intended users, customer, or other authorized stakeholders. It evaluates the system against acceptance criteria, user needs, business processes, contractual obligations, and other specified requirements. The emphasis is therefore on demonstrating that the delivered system satisfies the agreed basis for acceptance, rather than specifically investigating technical interface integration. The ISTQB glossary defines system integration testing in terms of testing interfaces between systems and acceptance testing as formal testing in relation to user needs, requirements, and business processes. See the ISTQB system integration testing glossary entry and the ISTQB acceptance testing glossary entry.

QUESTION NO: 25

You are working on a project that is doing a maintenance release of a large, high-risk banking system. Regression testing utilizes a set of 23525 test cases rerun against each test release. Which of the following is a benefit of test automation for this project?

A. Ease of filling bug reports

B. Reduction of repetitive work

C. Low cost of automating all the tests

D. Opportunity to learn a new skill

ANSWER: B

Explanation:

Reduction of repetitive work is the key benefit in this situation. The project repeatedly executes an exceptionally large regression suite for every test release. Once automated test scripts have been developed, reviewed, and maintained, they can run the same defined checks consistently on each build without requiring a tester to manually repeat tens of thousands of steps. This makes regression execution more efficient and repeatable, supports frequent maintenance releases, and frees testers to concentrate their time on activities that benefit from human judgement, such as exploratory testing, investigating failures, assessing risk, and designing new tests for changed functionality.

For a high-risk banking system, repeatability is particularly valuable: automated regression tests execute the same instructions and comparisons each time, helping the team detect unintended effects of maintenance changes promptly. Automation can also run unattended or outside normal working hours, which can shorten feedback cycles when the test environment is available. These benefits align with the Foundation Level treatment of test-tool support, which identifies automated test execution as especially useful for repetitive regression testing. See the ISTQB Certified Tester Foundation Level certification overview and the ISTQB glossary entry for automation.

QUESTION NO: 26

Which type of testing would be MOST likely to include code coverage?

A. functional testing

B. structural testing

C. non-functional testing

D. exploratory testing

ANSWER: B

Explanation:

structural testing is the correct answer because it derives, selects, or evaluates tests using the internal structure of the software. Code coverage is a structural coverage measure: it identifies which elements of the program's implementation have been exercised by a test suite. Depending on the coverage criterion, those elements may include statements, branches or decisions, conditions, paths, or other control-flow structures. Test execution is monitored or analyzed to calculate the proportion of the relevant code elements that was covered.

In the Foundation Level syllabus, structural testing is described as testing based on analysis of the internal structure of a component or system. Structural coverage is therefore an important way to assess the thoroughness of tests designed from, or evaluated against, that internal structure. Coverage figures can help reveal areas of code that have not yet been exercised and can guide the creation of additional tests. However, a coverage result is a measurement of exercised structure, rather than proof that all required behaviors have been correctly tested. See the ISTQB Certified Tester Foundation Level certification materials and the ISTQB glossary entry for code coverage.

QUESTION NO: 27

Which of the following statements about regression testing is TRUE?

A. Regression testing is performed only after a defect has been found in production.

B. Regression testing is performed to detect unintended side effects of changes.

C. Regression testing replaces confirmation testing after a defect has been fixed.

D. Regression testing is limited to testing newly implemented functionality.

ANSWER: B

Explanation:

Regression testing is performed to detect unintended side effects of changes. is correct because a change made to correct a defect, add functionality, or adapt the system can affect areas that were previously working correctly. Regression testing checks that existing functionality has not been adversely affected by such changes.

Confirmation testing is focused on checking that a specific reported defect has been fixed. Regression testing has a broader purpose and may include re-executing previously successful tests on changed or related areas of the system. See the ISTQB Glossary entry for regression testing.