

Question 5.6

(FL-5.1.6, K1)

What does the test pyramid model describe?

- A. Team effort spent on testing, increasing from iteration to iteration.
- B. A list of project tasks sorted by descending number of required test activities for each task.
- C. The granularity of tests at each test level.
- D. Test effort at each test level.

Choose one answer.

Question 5.7

(FL-5.1.7, K2)

In which test quadrant is component testing located?

- A. In the technology-oriented, team-supporting quadrant that includes automated tests and being part of the continuous integration process.
- B. In the business-oriented, team-supporting quadrant that includes acceptance criteria testing.
- C. In the business-oriented, product-criticizing, that includes tests focusing on users needs.
- D. In the technology-oriented, product-criticizing quadrant that includes automated non-functional tests.

Choose one answer.

Question 5.8

(FL-5.2.1, K1)

The likelihood of a system performance risk was estimated as “very high.”

What can be said about the impact of this risk?

- A. We know nothing about impact; impact and probability are independent.
- B. The impact is also very high; high likelihood risks also have a high impact.
- C. The impact is low because impact is inversely proportional to likelihood.
- D. Until this risk occurs, we cannot assess its impact.

Choose one answer.

Question 5.9

(FL-5.2.2, K2)

Which of the following is an example of a consequence of a project risk?

- A. User death due to software failure.
- B. Failure to complete all tasks intended for completion in a given iteration.
- C. Very high software maintenance costs.
- D. Customer dissatisfaction due to an inconvenient user interface.

Choose one answer.

Question 5.10

(FL-5.2.3, K2)

The tester is working on a project preparing a new version of a mobile home-banking application. During risk analysis, the team identified the following two risks:

- Overly complicated interface for defining the transfers—especially for seniors.
- Malfunctioning of the transfer mechanism—transfers are executed late when the payment falls on Saturday or Sunday.

What are the most reasonable risk mitigation actions a tester should propose for these two risks?

- A. Technical review for transfer interface and branch coverage for transfer mechanism.
- B. Component testing for transfer interface and acceptance testing for transfer mechanism.
- C. Beta testing for transfer interface and usability testing for transfer mechanism
- D. White-box testing for transfer interface and non-functional testing for transfer mechanism.

Choose one answer.

Question 5.11

(FL-5.2.4, K2)

After performing system testing and releasing the software to the customer, the software producer took out insurance with an insurance company. The producer did this in case a malfunction in the software caused its users to lose their health.

What kind of mitigation action are we dealing with here?

- A. Contingency plans.
- B. Risk mitigation through testing.
- C. Risk transfer.
- D. Risk acceptance.

Question 5.12

(FL-5.3.1, K1)

Which of the following is NOT a metric used for testing?

- A. Residual risk level.
- B. Coverage of requirements by source code.
- C. Number of critical defects found.
- D. Test environment implementation progress.

Choose one answer.

Question 5.13

(FL-5.3.2, K2)

Which of the following will NOT normally be included in a test completion report?

- A. The remaining (unmitigated) risks are risks R-001-12 and R-002-03.
- B. Deviations from test plan: integration testing delayed by 5 days.
- C. Number of open critical defects: 0.
- D. Tests scheduled for the next reporting period: component tests of the M3 component.

Choose one answer.

Question 5.14

(FL-5.3.3, K2)

Which of the following is the BEST form of communicating the status of testing?

- A. Test progress reports and test completion reports, because they are the most formal form of communication.
- B. Which form of communication is best will depend on various factors.
- C. Emails, because they allow for quick exchange of information.
- D. Verbal, face-to-face communication, because it is the most effective form of communication between people.

Choose one answer.

Question 5.15

(FL-5.4.1, K2)

The team received information from the client about a software failure. Based on the software version number, the team was able to reconstruct all the component and testware versions that were used to generate the software release for this customer. This made it possible to locate and fix the defect more quickly, as well as to analyze which other versions of the software release should be patched related to the defect.

Which process enabled the team to execute the above scenario?

- A. Configuration management.
- B. Impact analysis.
- C. Continuous delivery.
- D. Retrospective.

Choose one answer.

Question 5.16

(FL-5.5.1, K3)

While testing a new application, a tester finds a malfunctioning of the login component. According to the documentation, the password is supposed to have at least ten characters, including a minimum of one uppercase and one lowercase letter and one digit.

The tester prepares a defect report containing the following information:

- Title: Incorrect login.
- Brief summary: Sometimes, the system allows passwords of 6, 7, and 9 characters.
- Product version: compilation 11.02.2020.
- Risk level: High.
- Priority: Normal.

What VALUABLE information is missing from the above defect report?

A. Steps to reproduce the defect.
B. Data identifying the product being tested.
C. Defect status.
D. Ideas for improving the test case.

Choose one answer.

Exercises for Chapter 5

Exercise 5.1

(FL-5.1.4, K3)

A group of three experts estimates the test effort for the task of “conducting system testing” using a method that is a combination of planning poker and three-point estimation. The procedure is as follows:

- Experts determine the pessimistic, most likely, and optimistic values for three-point estimation. Each of them is determined by conducting a planning poker. The poker session is carried out until at least two experts give the same value, in which case the poker ends and the result is the value determined by the majority of experts.
- Experts use three-point estimation with the pessimistic, most likely, and optimistic parameters determined in the previous step; this estimation is the final result. The standard deviation is also calculated to describe the estimation error.

The results of the poker session are shown in Table 5.5. All values are expressed in person-days.

Table 5.5 Results of the planning poker session

Estimated value	Iterations of planning poker
Optimistic (a)	Results of iteration 1: 3, 5, 8 Results of iteration 2: 3, 3, 5
Most likely (m)	Results of iteration 1: 5, 5, 3
Pessimistic (b)	Results of iteration 1: 13, 8, 21 Results of iteration 2: 8, 13, 40 Results of iteration 3: 13, 13, 13

Table 5.6 Historical data on a project

Phase	Number of people involved	Duration (days)
Design	4	5
Implementation	10	18
Testing	4	10

What are the final effort estimation and estimation error for the task under analysis?

Exercise 5.2

(FL-5.1.4, K3)

Note—this task is quite difficult and requires some analytical skills and a good understanding of product metrics such as effort. However, we give this exercise on purpose, to show what kind of problems managers may deal with in practice. Often, these are non-trivial problems, like the one below.

Table 5.6 shows historical data from a completed project in which the design, implementation, and testing phases were performed sequentially, one after the other:

You want to use ratio-based effort estimation technique to estimate the effort needed to run a new, similar project. It is known that the new project will have four designers, six developers, and two testers. The contract calls for the project to be completed in 66 days.

How many days should we plan for design, how many for implementation, and how many for testing in a new project?

Hint. Calculate the effort for each of the three phases in the previous project (in person-days). Then calculate the total effort required for the new project. Finally, use the ratio-based technique to distribute the effort in the new project among the phases.

Exercise 5.3

(FL-5.1.5, K3)

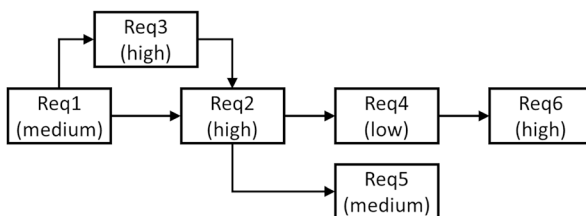
You are testing an application that supports help-line operation in an insurance company and allows you to find a customer’s policy according to their ID number. The test cases are described in Table 5.7. Priority 1 means the highest priority and 4—the lowest priority.

Define the correct order in which the test cases should be executed.

Table 5.7 Test cases for the help-line system testing with priorities and dependencies

Test case	Test condition covered	Priority	Logical dependence
TC001	Search by ID number	1	002, 003
TC002	Entering personal data	3	
TC003	ID number modification	2	002
TC004	Deletion of personal data	4	002

Fig. 5.20 Priorities and relationships between requirements



Exercise 5.4

(FL-5.1.5, K3)

The team wants to prioritize tests according to the prioritization of requirements presented to the team by the customer. The priority of each of the six requirements Req1–Req6 is specified by the customer as low, medium, or high. In addition to the priorities, there is a certain logical order between the requirements. Some of them can be implemented (and tested) only after others have been implemented.

The priorities and dependencies between requirements are shown in Fig. 5.20. An arrow leading from requirement A to requirement B means that requirement B can be implemented and tested only after the implementation and testing of requirement A are complete.

Determine the final order in which the requirements should be tested.

Exercise 5.5

(FL-5.5.1, K3)

You are testing an application for an e-commerce store. The requirements to consider the discount for a customer are given below:

- If a customer has made a purchase of less than \$50 and does not have a loyalty card, they do not receive a discount.
- If a customer has made a purchase of less than \$50 and has a loyalty card, they will receive a 5% discount.
- If a customer has made a purchase of at least \$50 and less than \$500 and does not have a loyalty card, they will receive a 5% discount.
- If a customer has made a purchase of at least \$50 and less than \$500 and has a loyalty card, they will receive a 10% discount.
- If a customer has made a purchase of at least \$500 and does not have a loyalty card, they will receive a 10% discount.
- If a customer has made a purchase of at least \$500 and has a loyalty card, they will receive a 15% discount.

Table 5.8 Table of test case results (Exercise 5.5)

Test case	Purchase amount [\$]	Has a card?	Discount value [\$]	To be paid [\$]	Test result
TC 001	25	Yes	1	24	Fail
TC 002	50	Yes	0	50	Fail
TC 003	50	No	2.50	48	Fail
TC 004	500	Yes	50	450	Pass
TC 005	600	Yes	50	550	Fail

You have executed several test cases. Their results are shown in Table 5.8. Prepare a defect report for TC 003.

Chapter 6 Test Tools



Keyword

Test automation The use of software to perform or support test activities


6.1 Tool Support for Testing

FL-6.1.1 (K2) Explain how different types of test tools support testing

A well-known saying goes that automation replaces what works with something that almost works, but is faster and cheaper [70]. This saying quite accurately captures the reality of test automation.

We already know that the basic tasks of testing are:

- Product analysis and evaluation
- Determining what measures will be used in testing
- Test analysis
- Test design
- Test implementation
- Test execution
- Analyzing the test result
- Test environment management

Not all of these activities can be fully automated. In fact, automated testing means computer-aided testing, and in practice, when talking about **test automation** , one usually means automating their execution, i.e., implementing and executing automated test scripts. However, it is important to note that automation can also include other areas of testing. For example, in a *model-based testing* (MBT) approach, automation can include *test design* and *determination of the expected result* based on the analysis of the model provided by the tester.

Today's tools that support testing support several testing activities, such as:

- Test design, implementation, and execution
- Test data preparation
- Test management, defect management, and requirements management
- Test execution monitoring and reporting

The use of test tools can have several purposes:

- Automate repetitive tasks or tasks that require a lot of resources or significant effort to perform manually (e.g., regression tests)—this allows us to increase the efficiency of the tests.
- Support manual activities and increase the efficiency of test activities—which increases the reliability of testing.
- Increase the consistency of testing and the reproducibility of defects, thus increasing the quality of testing.
- Automate activities that cannot be done or are very difficult to do manually (such as performance testing).

Probe effect

Some types of test tools can be invasive—their use can affect the actual test result. This phenomenon is called the *probe effect*. For example, when we use a performance testing tool, the performance of the software under test may be slightly worse, due to the introduction of additional code instructions to the software by performance testing tool.

Test tools support and facilitate many testing activities. Examples include, but are not limited to, the following groups of tools:

- Management tools—increase the efficiency of the test process by facilitating application lifecycle management, test basis to requirements traceability, test monitoring, defect management, and configuration management; they offer team-work tools (e.g., scrum board) and provide automated reporting.
- Static testing tools—support the tester in performing reviews (mainly in review planning, supporting traceability, facilitating communication, collaborating on reviews, and maintaining a repository for collecting and reporting metrics) and static analysis.
- Dynamic testing tools—support the tester in performing dynamic analysis, code profiling, monitoring memory usage during program execution, etc.
- Test design and test implementation tools—facilitate the generation of test cases, test data, and test procedures, support model-based testing approach, and provide frameworks for behavior-driven development (BDD) or acceptance test-driven development (ATDD).
- Test execution and coverage measurement tools—facilitate the automated execution of tests (test scripts) and the automated measurement of coverage achieved by these tests, as well as enable the measurement of expected and actual test results; these tools include tools for automated GUI/API testing and frameworks

for component testing and for executing tests in approaches such as BDD or ATDD.

- Non-functional testing tools—allow the tester to perform non-functional testing that is difficult or impossible to perform manually (e.g., generating load for performance testing, scanning for security vulnerabilities, usability testing of interfaces and web pages, etc.).
- DevOps tools—support deployment pipeline, workflow tracking, build automation process, automated software deployment, continuous integration, and continuous delivery.
- Collaboration tools—facilitate communication.
- Tools to support the scalability and standardization of deployments (e.g., virtual machines, containerization tools, etc.).

6.2 Benefits and Risks of Test Automation

FL-6.2.1 (K2) Recall the benefits and risks of test automation

Simply owning and using a tool does not yet guarantee success. A well-known Grady Booch saying goes: “a fool with a tool is still a fool.” Achieving real and lasting benefits from implementing a new tool in an organization always requires additional effort. A tool is *just* a tool and will not do all the work for the tester. It’s like expecting the hammer we bought to drive nails by itself.

Potential benefits of using the tools include:

- Saving time by reducing repetitive manual work (e.g., running regression tests, re-entering the same test data, comparing expected and actual test results, checking code against coding standards).
- Greater consistency and repeatability prevents simple human errors (e.g., tests are consistently derived from requirements, test data is created in a systematic way, and tests are executed by the tool in the same order, in the same way, and with the same frequency).
- More objective assessment (e.g., consistent coverage measurement) and the ability to calculate metrics that are too complex for humans to calculate.
- Easier access to test information (e.g., statistics, charts, and aggregated data on test progress, defects, and execution time) to support test management and reporting.
- Reduced test execution time, providing earlier defect detection, faster feedback, and faster time to market.
- More time for testers to design new, stronger, and more effective tests.

There are also certain risks associated with the use of testing tools:

- Unrealistic expectations about the benefits of the tool (including functionality and ease of use).

- Inaccurate/erroneous estimation of the time, cost, and effort required to implement the tool, maintain testing scripts, and change the existing manual testing process.
- Using a test tool when manual testing is more appropriate (e.g., interface usability testing subject to human evaluation).
- Relying on the tool when human critical thinking is needed.
- Dependence on the tool vendor, which may go out of business, retire the tool, sell the tool to another vendor, or provide poor support (e.g., responses to inquiries, updates, and error fixes).
- When using the open-source tool, this tool project may be abandoned, meaning that no further updates will be available or its internal components may need to be updated quite frequently as part of the tool's further development.
- The platform and the tool are not compatible with each other.
- The failure to comply with regulatory requirements and/or safety standards.

Sample Questions

Question 6.1

(FL-6.1.1, K2)

Which of the following activities should be supported by a test management tool?

- A. Test design.
- B. Requirements management.
- C. Test execution.
- D. Defect reporting.

Choose one answer.

Question 6.2

(FL-6.1.2, K1)

Which TWO are the benefits associated with using test tools?

- A. Tool dependence.
- B. Tool vendor dependency.
- C. Increasing the repeatability of tests.
- D. Mechanical coverage assessment.
- E. Cost of maintaining testware higher than estimated.

Choose two answers.

Part III
Answers to Questions and Exercises

Answers to Sample Questions



Answers to Questions from Chap. 1

Question 1.1

(FL-1.1.1, K1)

Correct answer: B

Answer A is incorrect. This is one of the test objectives according to the syllabus.

Answer B is correct. The objective of acceptance testing is to confirm (validation) that the system works as expected, rather than looking for failures (verification).

Answer C is incorrect. This is one of the test objectives according to the syllabus.

Answer D is incorrect. This is one of the test objectives according to the syllabus.

Question 1.2

(FL-1.1.2, K2)

Correct answer: A

Triggering failures (ii) and performing retests, or confirmation testing (iv), are the test activities. Debugging is the process of finding, analyzing, and fixing the causes of failure in a component or system, so finding defects in code (i) and analyzing the defects found (iii) are the debugging activities. Thus, answer A is correct.

Question 1.3

(FL-1.2.1, K2)

Correct answer: B

Answer A is incorrect. Such a remark can create conflict in the team, and this should not be allowed, because conflict threatens the achievement of project goals.

Answer B is correct. Reporting this deficiency or including the issue of the time of turning an object into gold as one of the story acceptance criteria illustrates well the

contribution of testing in the development process. This is because it minimizes the risk of developers not taking this requirement into account.

Answer C is incorrect. There is no justification for the *immediate* reaction of the product owner. Moreover, testing cannot *force* anyone to do anything. It can only inform about problems.

Answer D is incorrect. Performance efficiency is a nonfunctional software quality characteristic. The problem found is a functional defect, not a nonfunctional one. Also, we do not improve the game performance by removing this defect.

Question 1.4

(FL-1.2.2, K1)

Correct answer: A

Quality assurance focuses on preventing the introduction of defects by establishing, implementing, and controlling appropriate processes (i), while testing focuses on evaluating software and related products to determine whether they meet specified requirements (iii). Quality assurance does not *control* the quality of the product being developed (ii). Testing does not focus on *removing* defects from software (iv). So the correct sentences are (i) and (iii). Thus, the correct answer is A.

Question 1.5

(FL-1.2.3, K1)

Correct answer: B

Answer A is incorrect. This is the definition of *error* according to the ISTQB® glossary.

Answer B is correct. According to the ISTQB® glossary, a defect is an imperfection or deficiency in a work product where it does not meet its requirements or specifications.

Answer C is incorrect. This is the definition of *failure* according to the ISTQB® glossary.

Answer D is incorrect. A test case is a work product, not a defect in a work product.

Question 1.6

(FL-1.3.1, K2)

Correct answer: D

Answer A is incorrect. This principle states that early testing and defect detection allow us to fix defects early in the SDLC, reducing or eliminating costly changes that would have to be made if the defects were discovered later, such as after the release.

Answer B is incorrect. This principle says that testing is done differently in different business contexts.

Answer C is incorrect. This principle says that you need to modify existing tests and test data and write new tests so that the test suite is constantly ready to detect new defects.

Answer D is correct. This principle says exactly that “A small number of system components usually contain most of the defects discovered or are responsible for most of the operational failures. This phenomenon is an illustration of the Pareto principle.”

Question 1.7

(FL-1.4.1, K2)

Correct answer: C

According to the syllabus, the testability of the test basis is checked during test analysis. Hence, the correct answer is C.

Question 1.8

(FL-1.4.2, K2)

Correct answer: C

Answer A is incorrect. Budget has a significant impact on the test process.

Answer B is incorrect. Standards and norms have a significant impact on the test process, especially in audited projects or critical systems projects.

Answer C is correct. The number of certified testers employed in an organization has no *significant* impact on the testing process.

Answer D is incorrect. Testers’ knowledge of the business domain has a significant impact on the testing process, as it enables more effective communication with the customer and contributes to testing efficiency.

Question 1.9

(FL-1.4.3, K2)

Correct answer: D

Answer A is incorrect. A test progress report is a typical work product of test monitoring and control.

Answer B is incorrect. Information about the current risk level of a product is the typical information reported as part of test monitoring.

Answer C is incorrect. If decisions made within test control are documented, it is in this phase.

Answer D is correct. The test completion report is a work product produced during the test completion phase.

Question 1.10

(FL-1.4.4, K2)

Correct answer: A

Answer A is correct. If risks are quantified, test results can be tracked to test cases, and these to the risks they cover. If all test cases traced back to a given risk pass, the risk level remaining in the product can be considered to have decreased by the value of that risk.

Answer B is incorrect. Defining an acceptable level of code coverage is an example of establishing an exit criteria. The traceability mechanism has nothing to do with this process.

Answer C is incorrect. Traceability will not help determine the expected outcome of a test case, because the traceability does not have the property of a test oracle.

Answer D is incorrect. Deriving this type of test data can be possible by using a proper test technique rather than a traceability mechanism.

Question 1.11

(FL 1.4.5, K2)

Correct answer: D

The test management activities mainly include tasks performed in test planning, monitoring, test control, and test completion. In particular, this includes coordinating the implementation of the test strategy and test plan (i), creating a test completion report (iii), and deciding on test environment implementation (v). The test activities include tasks that occur mainly during the test analysis, test design, test implementation, and test execution phases. Thus, tester is responsible for defining test conditions (ii), test automation (iv), and verifying test environments (vi). Thus, the correct answer is D.

Question 1.12

(FL-1.5.1, K2)

Correct answer: C

According to the syllabus, typical generic skills of a good tester include, in particular, analytical thinking (A), domain knowledge (B), and communication skills (D). Programming skill (C) is not a critical attribute of a tester. It is not necessarily needed for good test execution (e.g., for manual or exploratory testing).

Question 1.13

(FL-1.5.2, K1)

Correct answers: B, E

Answer A is incorrect. Typically, developers, not testers, implement and execute component tests.

Answer B is correct. This is one feature of the “whole team” approach, relying on the cooperation of all stakeholders.

Answer C is incorrect. The business representative is not competent to choose the tools for the development team—it is the team that chooses the tools it wants to use, and the formal decision is made by management or, in agile methodologies, by the team itself.

Answer D is incorrect. The client is not competent in nonfunctional test design; this task falls on the testers and developers.

Answer E is correct. Shared responsibility and attention to quality are two of the basic principles of the “whole team” approach.

Question 1.14

(FL 1.5.3, K2)

Correct answer: A

Answer A is correct. Testers have a different perspective on the system under test than developers and avoid many of the cognitive errors of the work products' authors.

Answer B is incorrect. Developers can (and indeed should) test the code they create.

Answer C is incorrect. This is how testers should work, but failures can also be reported constructively by other stakeholders. This is not the reason for independent testing.

Answer D is incorrect. Finding defects should not be seen as a criticism of developers, but this has nothing to do with independence of testing, only with the desire for harmonious cooperation between developers and testers.

Answers to Questions from Chap. 2**Question 2.1**

(FL-2.1.1, K2)

Correct answer: D

Answer A is incorrect. In a sequential model, like V-model, and for life-critical software like autopilot, experience-based test techniques should be a complementary, not the primary technique.

Answer B is incorrect. The choice of SDLC model does not directly affect whether static tests will be present in the project. Moreover, performing static tests early, especially for life-critical systems, like autopilot, is considered a good practice.

Answer C is incorrect. The V-model is a sequential SDLC model, so there are no iterations. In addition, because of its sequential nature, the running software, even in prototype form, can usually only be available in later phases of the development cycle.

Answer D is correct. In the early phases of SDLC models like V-model, testers are usually involved in requirements reviews, test analysis, and test design. Executable code is typically developed in later phases, so dynamic testing usually cannot be performed in the early phases of the SDLC.

Question 2.2

(FL-2.1.2, K1)

Correct answer: D

In the V-model, each development phase (the left arm of the model) corresponds to the associated testing phase (the right arm of the model). On the model, this is represented by the horizontal arrows between the phases (e.g., from acceptance testing to the requirements phase; from system testing to the design phase; etc.). Hence, the correct answer is D.

Question 2.3

(FL-2.1.3, K1)

Correct answer: B

Answer A is incorrect. TDD (test-driven development) is about writing low-level component tests that do not use user stories.

Answer B is correct. ATDD (acceptance test-driven development) uses acceptance criteria of the user stories as the basis for test case design.

Answer C is incorrect. In the FDD (feature-driven development) approach, the basis for software development are the defined features, not tests; this approach has nothing to do with ATDD (see correct answer).

Answer D is incorrect. BDD (behavior-driven development) uses as a test basis a description of the desired behavior of the system, usually in Given/When/Then format.

Question 2.4

(FL-2.1.4, K2)

Correct answer: B

Answer A is incorrect. DevOps has nothing in common with automated test data generation.

Answer B is correct. Activities performed automatically after the developer commits code to the repository, such as static analysis, component testing, or integration testing, allow very quick feedback to the developer on the quality level of the code committed by the developer.

Answer C is incorrect. This type of activity is possible in model-based testing, for example. The DevOps approach is not about automated test case generation.

Answer D is incorrect. The DevOps approach does not affect the time of release planning and iteration planning; moreover, even if this was true, it is not a test-related benefit, but rather a project management benefit.

Question 2.5

(FL-2.1.5, K2)

Correct answer: A

Answer A is correct. One example of the shift-left approach is the use of the test-first approach exemplified by the acceptance test-driven development (ATDD).

Answer B is incorrect. The shift-left approach does not distinguish exploratory testing in any way. Rather, the emphasis on using specific test types depends on risk analysis.

Answer C is incorrect. *Creating* GUI prototypes is not an example of using the shift-left approach, because the creation itself is not related to testing.

Answer D is incorrect. This is an example of a *shift-right* approach, that is, *late* testing, after the software has been released to the customer, in order to monitor the quality level of the product in the operational environment on an ongoing basis.

Question 2.6

(FL-2.1.6, K2)

Correct answer: C

Answer A is incorrect. Testers should participate in retrospective meetings by addressing all issues raised at these meetings.

Answer B is incorrect. Testers should participate in all aspects of the retrospective meeting. The role described is more like that of a facilitator.

Answer C is correct. This is the typical activity performed by a tester in a retrospective meeting: to discuss what happened during the completed iteration.

Answer D is incorrect. This is not the purpose of a retrospective meeting. The tester should discuss what happened during the last iteration.

Question 2.7

(FL-2.2.1, K2)

Correct answer: B

Answer A is incorrect. The component integration tests we are dealing with here focus on the interaction and communication between components, not on the components themselves.

Answer B is correct. We want to perform integration testing. The architecture design is the typical test basis for this type of testing, because it usually describes how the various components of the system communicate with each other.

Answer C is incorrect. Risk analysis reports are more useful for system testing than integration testing.

Answer D is incorrect. Regulations are usually useful for high-level testing and validation, such as in acceptance testing.

Question 2.8

(FL-2.2.2, K2)

Correct answer: D

Answers A and C are incorrect. These tests check “what” the system does and are therefore examples of functional testing.

Answer B is incorrect. This is an example of a white-box test, not a nonfunctional test.

Answer D is correct. This is an example of a nonfunctional test or, more precisely, a performance test. This test type checks “how” the system works, not “what” it does.

Question 2.9

(FL-2.2.3, K2)

Correct answer: D

Confirmation testing is performed after a defect has been found and reported to be fixed. Since we don’t know when the test will trigger a failure, nor do we usually know how long the repair will take, we cannot predict the timing of the confirmation test execution. Thus, it is impossible to accurately schedule these tests in advance. All other test types can be planned in advance, and a schedule for their execution can be put into the test plan.

Question 2.10

(FL-2.3.1, K2)

Correct answer: C

Answer A is incorrect. We do not update the software, but we fix it.

Answer B is incorrect. It does not appear from the scenario that we are performing any software migration activities.

Answer C is correct. Software modification is one of the events that trigger maintenance. Repairing a defect is a software modification.

Answer D is incorrect. While this is a maintainability trigger for IoT systems, in this scenario, we want to fix a defect—not introduce new system functionality.

Answers to Questions from Chap. 3

Question 3.1

(FL-3.1.1, K1)

Correct answer: A

Answer A is correct. The document that defines the rules for reviews can also be reviewed. In doing so, it does not have to be reviewed according to the provisions of this document; the review can be conducted based on common-sense criteria.

Answer B is incorrect. The document talks about the rules for conducting reviews, but we can review it without following the rules it discusses, just using common sense.

Answer C is incorrect. The fact that a document is not the work product of some specific process does not exclude it from being included in a review.

Answer D is incorrect. Reviews can be applied to any human-understandable document.

Question 3.2

(FL-3.1.2, K2)

Correct answer: B

Answer A is incorrect because we did not run the code but only analyzed its properties.

Answer B is correct, this is a classic example of the gain from using a static technique—in this case, static analysis.

Answer C is incorrect because the measurement of cyclomatic complexity, the analysis of this measurement, and the refactoring of code are not managerial activities but technical ones.

Answer D is incorrect because static analysis is not an example of a formal test technique; such techniques include equivalence partitioning, boundary value analysis, or other black- or white-box test techniques. The result of static analysis is not a test case design.

Question 3.3

(FL-3.1.3, K2)

Correct answer: C

Answer A is incorrect. Static testing directly detects defects, not failures. It cannot detect failures, since we do not execute the work product under test. Dynamic testing directly detects failures, not defects.

Answer B is incorrect. For example, reviews may be performed at very late stages (e.g., they may check the user documentation), and dynamic tests may start early in the implementation phase.

Answer C is correct. Static analysis and dynamic analysis have the same goals (see Sect. 1.1.1), such as identifying defects (directly or indirectly, through failures)

as early as possible in the development cycle. So, regarding the *purpose*, there is no difference between these techniques.

Answer D is incorrect. First, reviews usually do not require any programming skills; second, it does not answer the question, which was about the criterion of purpose, not the required skills.

Question 3.4

(FL-3.2.1, K1)

Correct answer: C

Sentence (i) is false; developers implement those features that are required by the business and are part of the iteration. When they complete their tasks, they support other tasks related to the iteration.

Sentence (ii) is true; frequent feedback helps focus attention on the features of greatest value.

Sentence (iii) is false; early feedback may even result in the need for more testing due to frequent or significant changes.

Sentence (iv) is true; users indicate which requirements are skipped or misinterpreted, resulting in a final product that better meets their needs.

Thus, the correct answer is C.

Question 3.5

(FL-3.2.2, K2)

Correct answer: C

Answers A and D are incorrect. These activities are part of the planning phase.

Answer B is incorrect. Collecting metrics is part of the “fixing and reporting” activity.

Answer C is correct. According to the syllabus, this is an activity performed during the review initiation phase.

Question 3.6

(FL-3.2.3, K1)

Correct answer: B

Answer A is incorrect. The review leader’s role is overall responsibility for the review, as well as deciding who is to participate in the review and where and when the review is to take place. Thus, the leader has an organizational role, while the facilitator (see the correct answer) is a technical role directly related to conducting review meetings.

Answer B is correct. The moderator (also known as the facilitator) is responsible for ensuring that the review meetings run effectively.

Answer C is incorrect. It may be the author’s responsibility to make a presentation or comment on their work product, but the author never steps into the role of moderator.

Answer D is incorrect. The reviewers' job is to substantively review the work product, not to moderate meetings. The moderator is a role that precisely relieves reviewers of the need to write down observations made by them during the review meeting.

Question 3.7

(FL-3.2.4, K2)

Correct answer: D

Understanding the work product (or learning something) is one of the goals of a walk-through. Walk-throughs can take the form of so-called dry runs, so as a result of using this type of review, the team can most effectively understand how the software works and thus more easily discover the cause of a strange failure.

Question 3.8

(FL-3.2.5, K1)

Correct answer: C

Answer A is incorrect. The main purpose of inspection is to find defects. Evaluating alternatives is a more appropriate goal for a technical review.

Answer B is incorrect. Inspections are usually conducted by *peers* of the author. The presence of management at a review meeting carries the risk of misunderstanding the purpose of the review and, for example, the risk assessment by that management of the individual members of the meeting.

Answer C is correct. Training in review techniques is one of the success factors for reviews.

Answer D is incorrect. Measuring metrics helps improve the review process. Moreover, in a formal review such as inspection, collecting metrics is a mandatory activity.

Answers to Questions from Chap. 4**Question 4.1**

(FL-4.1.1, K2)

Correct answer: B

Analysis of the software input domain takes place when using black-box test techniques such as equivalence partitioning and boundary value analysis.

Question 4.2

(FL-4.1.1, K2)

Correct answer: D

Answer A is incorrect. It describes a common feature of white-box test techniques.

Answer B is incorrect. If we already want to design test data based on code analysis, we need to have access to it, so we can do it through white-box testing.

Answer C is incorrect. It describes how coverage is measured in white-box test techniques.

Answer D is correct. The techniques listed in the question are black-box test techniques. According to the syllabus, in black-box test techniques, test conditions, test data, and test cases are derived from a test basis external to the object under test (e.g., requirements, specifications, use cases, etc.). Therefore, test cases will be able to detect discrepancies between these requirements and their actual implementation.

Question 4.3

(FL-4.2.1, K3)

Correct answer: A

The domain is the amount of purchase. There are three equivalence partitions for this domain according to the specification:

- “No discount” partition {0.01, 0.02, ..., 99.98, 99.99}
- “5% discount” partition {100.00, 100.01, ..., 299.98, 299.99}
- “10% discount” partition {300.00, 300.01, 300.02, ...}

Note that the smallest possible amount is 0.01 as the smallest possible *positive* number.

Answer A is correct. The three values 0.01, 100.99, and 500, each of which belongs to a different partition, cover all three equivalence partitions.

Answer B is incorrect. We have only three partitions, so the minimum set of values taken for testing should have three elements.

Answer C is incorrect. The values 1 and 99 belong to the same partition. This set does not contain a value for which the system should allocate a 10% discount, so the last partition is not covered.

Answer D is incorrect. All the values belong to the “no discount” partition. They are values representing possible discount percentages (0, 5, 10), but we are not analyzing the discount type domain but the input domain (amount of purchases).

Question 4.4

(FL-4.2.1, K3)

Correct answer: C

We are supposed to check two situations: one in which the machine does not give change and one in which it gives change.

Answer A is incorrect. Both scenario 1 and scenario 2 are scenarios in which the machine does not return any change. So we miss a case in which a change is given.

Answer B is incorrect. Both scenarios only cover the case in which the machine gives change. We miss a test in which the machine does not give change.

Answer C is correct. In scenario 1, the machine does not give change (the amount inserted is exactly equal to 75c), and in scenario 3, the machine gives 25c change.

Answer D is incorrect, because two scenarios are enough to cover all (two) equivalence partitions—see a justification for the correct answer.

Question 4.5

(FL-4.2.1, K3)

Correct answer: A

To achieve 100% equivalence partitioning “each choice” coverage, we need to cover every card type and every discount. The existing test cases do not cover only the diamond card and the 5% discount. These two items can be covered with one additional test case:

PT05: diamond card, 5%.

Thus, the correct answer is A.

Question 4.6

(FL-4.2.2, K3)

Correct answer: D

The domain under consideration is the length of the password, and the equivalence partitions look as follows:

- Password too short: {0, 1, 2, 3, 4, 5}
- Password with correct length: {6, 7, 8, 9, 10, 11}
- Password too long: {12, 13, 14, ...}

Existing test cases achieve 100% 2-value BVA coverage, so they must cover all the boundary values, that is, 0, 5, 6, 11, and 12. To achieve the 100% 3-value BVA coverage, we need to cover the following values:

- 0, 1 (for the boundary value 0)
- 4, 5, 6 (for the boundary value 5)
- 5, 6, 7 (for the boundary value 6)
- 10, 11, 12 (for the boundary value 11)
- 11, 12, 13 (for the boundary value 12)

So, in total, the values 0, 1, 4, 5, 6, 7, 10, 11, 12, and 13 should be tested. However, since we know that the values 0, 5, 6, 11, and 12 are already in our test set (because the 100% 2-value BVA is achieved), the missing values are 1, 4, 7, 10, and 13. Hence, answer D is correct.

Question 4.7

(FL-4.2.2, K3)

Correct answer: D

Achieving full BVA coverage is infeasible. The numbers of consecutive free washes are all multiples of 10: {10, 20, 30, 40, 50, ...}. But in order to apply the BVA method, the partitions must be *consistent*, i.e., they must not have “holes.” Therefore, if we wanted to apply the BVA to this problem, we would have to derive infinitely many equivalence partitions:

{1, ..., 9}, {10}, {11, ..., 19}, {20}, {21, ..., 29}, {30}, {31, ..., 39}, {40}, etc.,

and this would mean that we have to execute infinitely many test cases (since we have infinitely many boundary values: 1, 9, 10, 11, 19, 20, 21, 29, 30, 31, etc.).

Note that if we used the equivalence partitioning method, the problem could be solved, because we would have only two partitions: numbers divisible by 10 and other numbers. Therefore, only two tests, such as 9 and 10, would be enough to achieve coverage of equivalence partitions.

Question 4.8

(FL-4.2.3, K3)

Correct answer: A

Requirements are contradictory if, for a given combination of conditions, we can indicate two different sets of corresponding actions. In our case, the two different actions are “free ride”=YES and “free ride”=NO. To force the value of NO, the condition “student”=YES must be satisfied. To force the value of YES, either “member of parliament”=YES or “disabled person”=YES must occur.

Answer A is correct. This combination matches both rules R1 and R3, which give contradictory actions.

Answer B is incorrect. This combination only matches rules R1 and R2, which result in the same action.

Answer C is incorrect. This combination only matches column R3, so there can be no contradiction within a single rule.

Answer D is incorrect. It fits neither rule R1 nor R2 nor R3. So this is an example of a missing requirement, but not contradictory requirements.

Table 1 Combinations of conditions

Combination	Age	Residence	Earnings
1	Up to 18	City	Up to 4000/month
2	Up to 18	City	From 4001/month
3	Up to 18	Village	Up to 4000/month
4	Up to 18	Village	From 4001/month
5	19–40	City	Up to 4000/month
6	19–40	City	From 4001/month
7	19–40	Village	Up to 4000/month
8	19–40	Village	From 4001/month
9	From 41	City	Up to 4000/month
10	From 41	City	From 4001/month
11	From 41	Village	Up to 4000/month
12	From 41	Village	From 4001/month

Question 4.9

(FL-4.2.3, K3)

Correct answer: C

The number of columns in the full decision table is equal to the number of all possible combinations of conditions. Since we have three conditions, with 3, 2, and 2 possible choices, respectively, the number of all possible combinations of these conditions is $3 \cdot 2 \cdot 2 = 12$. These are listed in Table 1.

Question 4.10

(FL-4.2.4, K3)

Correct answer: A

Since we have three states and four events, there are $3 \cdot 4 = 12$ possible combinations (state, event). The table from Question 4.10 contains only four of them (i.e., there are only four valid transitions in the machine). Thus, invalid transitions are $12 - 4 = 8$. Here they are (the state and event are placed in parentheses sequentially):

1. (Initial, LoginOK)
2. (Initial, LoginError)
3. (Initial, Logout)
4. (Logging, Login)
5. (Logging, Logout)
6. (Logged, Login)
7. (Logged, LoginOK)
8. (Logged, LoginError)

None of these combinations appear in the list of valid transitions given in the task.

Another way to show that there is eight invalid transitions is to count empty cells in the state table, which looks as shown in Table 2.

Table 2 State table for Question 4.10

State\Transition	Login	LoginOK	LoginError	Logout
Initial	Logging			
Logging		Logged	Initial	
Logged				Initial

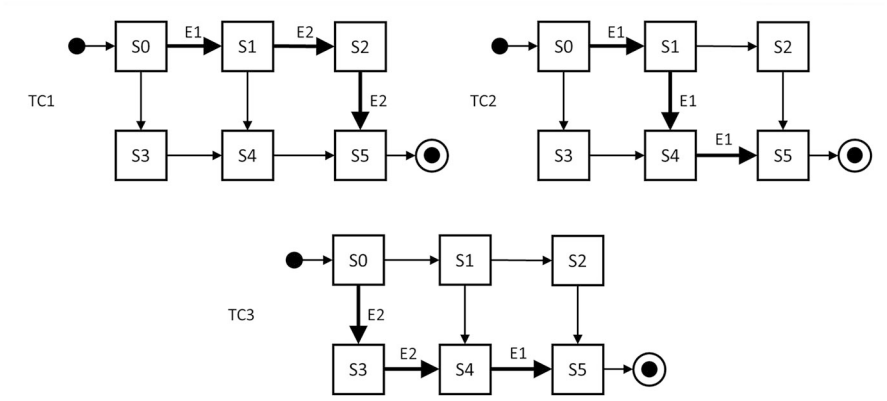


Fig. 1 Paths determined by tests covering all valid transitions (Question 4.11)

Question 4.11

(FL-4.2.4, K3)

Correct answer: B

Note that no two of the following three transitions can occur within a single test case:

- S0 > (E2) > S3.
- S1 > (E1) > S4.
- S2 > (E2) > S5.

This means that we need at least three test cases to cover all valid transitions. In fact, three test cases are enough, and these are:

TC1. S0 > (E1) > S1 > (E2) > S2 > (E2) > S5.

TC2. S0 > (E1) > S1 > (E1) > S4 > (E1) > S5.

TC3. S0 > (E2) > S3 > (E2) > S4 > (E1) > S5.

The paths defined by these test cases are shown in Fig. 1. Note that each transition (arrow) is covered by at least one test case.

Question 4.12

(FL-4.3.1, K2)

Correct answer: D

Answer A is incorrect. The branch coverage criterion subsumes the statement coverage criterion, but not vice versa. For example, for the code:

```
1. IF (x==0) THEN
2.   x := x + 1
3. RETURN x
```

one test case (for $x = 0$) will result in the passage of path 1, 2, and 3 and therefore achieve 100% statement coverage, but this test covers only two of the three branches: (1, 2) and (2, 3). The branch (1, 3), which will be executed when input x is different from zero, is uncovered.

Answer B is incorrect. The code example above shows this. The test case ($x = 0$) achieves 100% code coverage but causes only the truth outcome of the decision in statement 1. We do not have a test that forces a false outcome for this decision.

Answer C is incorrect. The program above can return any number other than zero. Test ($x = 0$) achieves 100% statement coverage but only forces the return of the value $x = 1$.

Answer D is correct. Statement coverage forces the execution of every statement in the code, so in particular, it means executing every statement that contains a defect. Of course, this *does not* mean triggering every failure caused by these defects, because the execution of an invalid statement may not have any negative effect. For example, the execution of the statement $x := a / b$ will be completely correct, as long as the denominator (b) is not equal to 0.

Question 4.13

(FL-4.3.2, K2)

Correct answer: C

Branch coverage requires that tests cover every possible flow of control between executable statements in the code, i.e., all possible branches in the code—both unconditional and conditional. The code under test has a linear structure, and each execution of this code will exercise statements in the order 1, 2, 3. This means that each time this program is run, all two unconditional branches occurring in it will be covered: (1, 2) and (2, 3). Hence, the correct answer is C: one test case with any input data x , y is enough.

Question 4.14

FL-4.3.3 (K2)

Correct answer: A

Answer A is correct. This is a fundamental property and advantage of white-box test techniques. In this approach, tests are designed directly on the basis of the structure of what is going to be tested (e.g., source code), so a full, accurate specification is not necessary for the test design itself (except from deriving the expected result).

Answer B is incorrect. White-box techniques do not always require programming skills. They can be used, for example, at the system test level, where the covered structure is, for example, the program menu.

Answer C is incorrect. There is no relationship between the coverage metrics of black-box and white-box techniques.

Answer D is incorrect. There is no direct relationship between white-box coverage and risk, because the risk level depends specifically on the impact of the risk, not just on how many lines of code has the function associated with a particular risk.

Question 4.15

(FL-4.4.1, K2)

Correct answer: D

Answer A is incorrect. The document does not mention boundary values.

Answer B is incorrect. In checklist-based testing, the checklist defines the “positive” features of the software, while the analyzed document talks about possible failures.

Answer C is incorrect. These are not use cases.

Answer D is correct. The document appearing in the question is a list of possible defects or failures. Such lists are used in the technique of fault attacks, a formalized approach to error guessing.

Question 4.16

(FL-4.4.2, K2)

Correct answer: A

Exploratory testing utilizes the knowledge, skills, intuition, and experience of the tester but gives the tester full room for maneuver when it comes to the repertoire of techniques they can use in a session-based exploratory testing. Therefore, answer A is correct.

Answers B and C are incorrect—see A.

Answer D is incorrect. Although formal test techniques are allowed (see A), the explanation in this answer is incorrect. In an exploratory approach, it is not necessary to have the test basis needed to derive test cases, since this is an experience-based test technique.

Question 4.17

(FL-4.4.3, K2)

Correct answer: D

Answer A is incorrect. Although checklists can be organized around nonfunctional testing, this is not the main advantage of using checklists.

Answer B is incorrect. In an experience-based approach such as checklist-based testing, it is impossible to precisely define meaningful measures of coverage, especially measures of code coverage.

Answer C is incorrect. Using checklists does not necessarily require expertise (especially in case of low-level, detailed checklists)—this approach fits more with exploratory testing.

Answer D is correct. In the absence of detailed test cases, testing based on checklists can provide a degree of consistency for testing.

Question 4.18

(FL-4.5.1, K2)

Correct answer: D

Answer A is incorrect. The tester cannot decide the acceptance criteria alone. User stories, including acceptance criteria, are written based on the cooperation of the product owner, developer, and tester.

Answer B is incorrect. This solution does not make sense. First, story planning is not the moment to write tests. Second, acceptance tests must be created on the basis of established and precise acceptance criteria, and at the moment, the team is in the process of negotiating these criteria, and it is not yet clear what form they will take.

Answer C is incorrect. The scenario does not say anything about performance, but even if this topic came up in the discussion, the product owner cannot be excluded from it.

Answer D is correct. This is a model example of negotiating a user story by all team members as part of a collaboration-based test approach.

Question 4.19

(FL-4.5.2, K2)

Correct answers: B, C

Answer A is incorrect. The acceptance criteria may address a nonfunctional aspect such as performance, but the term “fast enough” used in this criterion is imprecise and therefore untestable.

Answer B is correct. This is the desired mechanism for offering functionality in this software: the user can only order purchases when registered. This is a precise and testable criterion, directly related to the content of the story.

Answer C is correct. Acceptance criteria can take into account “negative” events, such as a user making an error during the registration process, which may result in a denial of further processing. This is a precise and testable criterion, directly related to the content of the story.

Answer D is incorrect. While it is a reasonable, precise and testable acceptance criterion, it does not directly address the story from the scenario. This is because it is written from the point of view of the system operator, not that of the e-shop customer.

Answer E is incorrect. This is an example of a rule for writing acceptance criteria, not a specific acceptance criterion for a user story.

Question 4.20

(FL-4.5.3, K3)

Correct answer: B

Test 1 is inconsistent with the business rule. The first condition is met (the time of the longest book held does not exceed 30 days), but after borrowing two new books, having already borrowed three others, the student will have a total of five books borrowed. This will not exceed the limit from the second condition of the business rule. So the system should allow lending, but in Test 1, the decision is “does not allow.”

Test 2 follows the business rule: the student does not keep any of the four borrowed books for more than 30 days and wants to borrow one new one, so they will have a total of five borrowed books. They will not exceed the limit, so the system allows them to borrow.

Test 3 follows the business rule: a professor has at least one held book (Days > 30), so the system cannot allow a book loan.

Test 4 follows the business rule: a professor has a clean account and wants to borrow six books, which is within the limit of ten books. The system should allow the loan.

Hence, only one is incorrect in the tests, so the correct answer is B.

Answers to Questions from Chap. 5**Question 5.1**

(FL-5.1.1, K2)

Correct answer: A

The test plan must be in line with the test strategy, not the other way around. So, test strategy is not a part of test plan. All other elements, budgetary constraints, scope of testing, and risk register, are parts of the test plan. Thus, the correct answer is A.

Question 5.2

(FL-5.1.2, K2)

Correct answer: D

Answer A is incorrect. Detailed risk analysis for user stories is performed during iteration planning, not during release planning.

Answer B is incorrect. Identification of nonfunctional aspects of the system to be tested is done during iteration planning, not during release planning.

Answer C is incorrect. Estimating the test effort for new features planned for an iteration is done during iteration planning, not during release planning.

Answer D is correct. During release planning, testers are involved in creating testable user stories and their acceptance criteria.

Question 5.3

(FL-5.1.3, K2)

Correct answer: D

Entry criteria include, in particular, availability criteria and the initial quality level of the test object. Therefore, entry criteria are the availability of testers (i) and passing all smoke tests (iv). In contrast, typical exit criteria are measures of thoroughness. Therefore, the absence of critical defects (ii) and achieving a certain threshold of test coverage (iii) are exit criteria. Thus, the correct answer is D.

Question 5.4

(FL-5.1.4, K3)

Correct answer: B

The team wants to calculate $E(5)$, for which it will need the values of $E(4)$, $E(3)$, and $E(2)$. Since the value of $E(4)$ is unknown, the team must first estimate $E(4)$ and then $E(5)$. According to the effort extrapolation model, we have:

$$E(4) = (1/3) * (E(3) + E(2) + E(1)) = (1/3) * (12 + 15 + 18) = 15.$$

So $E(4) = 15$. Now we can extrapolate the effort in the fifth iteration:

$$E(5) = (1/3) * (E(4) + E(3) + E(2)) = (1/3) * (15 + 18 + 15) = 16.$$

So $E(5) = 16$. This means that the correct answer is B.

Question 5.5

(FL-5.1.5, K3)

Correct answer: C

The first executed test case will be the one achieving the highest feature coverage, i.e., TC1, which covers four of the seven functions: A, B, C, and F. The second in order will be the one that covers the most of the previously uncovered features (i.e., D, E, G). Each of the TC2, TC3, and TC4 covers only one of these additional features, while TC5 covers two additional uncovered features: D and G. Thus, TC5 will be executed second. These first two test cases, PT1 and PT5, cover a total of six of the seven functions: A, B, C, D, F, and G. The third test case will be the one that covers the most of the features not covered so far, that is, feature E. This feature is covered only by TC4, so this test case will be executed third. Thus, the correct answer is C.

Question 5.6

(FL-5.1.6, K1)

Correct answer: C

Answer A is incorrect. Team effort has nothing to do with the concept of test pyramid.

Answer B is incorrect. The test pyramid does not refer to design tasks but directly to testing at different test levels.

Answer C is correct. The test pyramid illustrates the fact that we have more granular (detailed) tests at lower testing levels, and therefore, usually, more of these tests are needed, because each test achieves relatively low coverage. The higher the level, the lower the granularity of the tests, and usually, a decreasing number of them is needed, since usually, a single test at a higher level achieves more coverage than a single test at a lower level.

Answer D is incorrect. The test pyramid does not model test effort; it models the granularity and number of tests at each test level.

Question 5.7

(FL-5.1.7, K2)

Correct answer: A

Answer A is correct. According to the testing quadrants model, component (unit) tests are placed in the technology-oriented and team-supporting quadrant, as this quadrant contains automated tests and tests that are part of the continuous integration process. Component tests are typically this kind of tests.

Answers B, C, and D are incorrect—see the rationale for the correct answer.