ISTQB®

Certified Tester
Foundation Level

CTFL 4.0

Chapter 4

– Summary –

– Questions & Answers –

– Exam Questions Distribution –

Swipe for more

Karim Kalboussi

Examinable Learning Objectives :

Level 1 : Remember (K1)

- The candidate will remember, recognize and recall a term or concept.
- Action verbs : Identify, recall, remember, recognize.
- Exmaple : Identify typical test objectives.

Level 2 : Understand (K2)

- The candidate can select the reasons or explanations for statements related to the topic, and can summerize, compare, classify and give examples for the testing concept.
- Action verbs : Classify, compare, contrast, differentiate, ditinguish...
- Exmaple : Explain the activities of the review process.

Level 3 : Apply (K3)

- The candidate can carry out a procedure when confronted with a familiar task, or select the correct procedure and apply it to a given context.
- Action verbs : Apply, implement, prepare, use.
- Exmaple : Apply test case prioritization.

Chapter 4 Question Distribution in the Exam :

- There is a total of **11 questions** required for Chapter 4 :
  K1 = 0 questions
  K2 = 6 questions
  K3 = 5 questions
- Number of points for this chapter = 11

Question Distribution	K-Level	Number of Questions per LO (group)*	Suggested Points per Question	Probability of Appearance in the exam
Chapter 4				
FL-4.1.1	K2	1	1	1
FL-4.3.1	K2	2	1	0.6
FL-4.3.2				0.6
FL-4.3.3				0.6
FL-4.4.1	K2	2	1	0.6
FL-4.4.2				0.6
FL-4.4.3				0.6
FL-4.5.1	K2	1	1	0.5
FL-4.5.2				0.5
FL-4.2.1	K3	5	1	1
FL-4.2.2				1
FL-4.2.3				1
FL-4.2.4				1
FL-4.5.3				1

Summary of Chapter 4

4.1 Test Techniques Overview

4.1.1 (K2) Distinguish black-box test techniques, white-box test techniques and experience-based test techniques

- **Black-Box Test Techniques** (specification-based techniques) **:** Black-box test techniques are based on the analysis of the specified behavior of the test object without reference to its internal structure. Test cases are independent of how the software is implemented, so if the implementation changes but the required behavior remains the same, the test cases are still useful.
- **White-Box Test Techniques** (structure-based techniques) **:** White-box test techniques are based on the analysis of the test object's internal structure (implementation) and processing. Test cases depend on the software design, so they are created only after the design or implementation of the test object.
- **Experience-Based Test Techniques:** These techniques depend heavily on the tester's skills, knowledge, and experience for the design and implementation of test cases. These techniques can detect defects missed by black-box and white-box techniques, hence, experience-based test techniques are complementary to white-box and black-box test techniques.

4.2 Black-box Test Techniques

4.2.1 (K3) Use equivalence partitioning to derive test cases

- One test from each partition is sufficient. Equivalence partitions can be identified for any data elements (outputs, inputs, configuration items, internal values, time-related, etc.)
- Coverage items are the equivalence partitions.
- Valid partitions contain valid values, while invalid partitions contain invalid values.
- To have 100% coverage, test cases must exercise all identified partitions at least once each (valid and invalid).

→ This part (4.2.1) is better understood with exam questions practice!

4.2.2 (K3) Use boundary value analysis to derive test cases

- **2-value BVA :** For each boundary value, there are two coverage items: the boundary value and its closest neighbor belonging to the adjacent partition. To achieve 100% coverage, test cases must exercise all coverage items.
- **3-value BVA :** For each boundary value there are three coverage items : the boundary value and both its neighbors. To acheive 100% coverage, test cases must exercice all coverage items.

→ This part (4.2.2) is better understood with exam questions practice!

4.2.3 (K3) Use decision table testing to derive test cases

- The conditions and the resulting actions of the system form the rows of the decision table, while each column corresponds to a decision rule that defines a unique combination of conditions.
- To achieve 100% coverage with this technique, test cases must exercise all these columns.

→ This part (4.2.3) is better understood with exam questions practice!

4.2.4 (K3) Use state transition testing to derive test cases

- A state transition diagram models the behavior of a system by illustrating its possible states and valid state transitions.
- A state table is a model equivalent to a state transition diagram, its rows represent states, and its columns represent events.
- Unlike a state transition diagram, a state table shows invalid transitions.
- We have three coverage criteria for state transition testing :

- **All states coverage :** The coverage items are the states. 100% state coverage means that test cases must ensure that all states are visited. All states coverage is weaker that valid transitions coverage.
- **Valid transitions coverage (0-switch coverage):** The coverage items are valid transitions only. 100% valid-transitions coverage means all valid transitions are exercised by test cases. Valid-transitions coverage is the most widely used, and full valid-transitions coverage means full all states coverage.
- **All transitions coverage :** The coverage items are all the transitions in a state table. 100% all-transitions coverage means that test cases must exercise all the valid transitions and attempt to execute invalid transitions. Full all transitions coverage guarantees both full all states coverage and full valid transitions coverage.

→ This part (4.2.4) is better understood with exam questions practice!

4.3 White-box Test Techniques

4.3.1 (K2) Explain statement testing

- Coverage items are executable statements. Coverage percentage is the number of statements exercised by test cases divided by the total number of executable statements.
- 100% statement coverage ensures that all executable statements have been exercised, which means that each statement with a defect will be executed, which may cause a failure. However, exercising a statement with a test case will not detect defects in all cases, such as defects that are data-dependent (example : division by 0).
- 100% statement coverage doesn't mean the decision logic has been tested, as it may not exercise all the branches in the code !

4.3.2 (K2) Explain branch testing

- A branch is a transfer of control between two nodes in the control graph. Each transfer of control can be either unconditional (straight-line code) or conditional.
- In branch testing, the coverage items are branches, and we aim to design test cases to exercise branches in the code.
- Coverage is the number of branches exercised by test cases divided by the total number of branches.
- 100% branch coverage means all branches in the code (unconditional or conditional) are exercised by test cases.
- Exercising a branch with a test case will not detect defects in all cases; for example, it may not detect defects requiring the execution of a specific path in the code.

→ 100% branch coverage means 100% statement coverage, but 100% statement coverage does not mean 100% branch coverage !

4.3.3 (K2) Explain the value of white-box testing

- In all white-box testing techniques, the entire software implementation is taken into account during testing, which facilitates defect detection even when the software specification is vague, outdated, or incomplete (since test cases are based on the structure and not on the specification).
- If the software doesn't implement one or more requirements, white-box testing may not detect the resulting defects of omission !
- Black-box testing only, does not provide a measure of actual code coverage.
- White-box measures provide objective measurements of coverage and the necessary information to allow additional tests to increase the coverage and boost confidence in the code (support of black-box testing).

4.4 Experience-based Test Techniques

4.4.1 (K2) Explain error guessing

- Error guessing is a technique to anticipate (predict) the occurrence of errors, defects, and failures based on the tester's knowledge (how the app worked in the past, errors that developers tend to make, types of failures that occur in similar apps...)
- Errors, defects, and failures may be related to: input, output, logic, computation, interfaces...
- Fault attacks are a methodical approach to implementing error guessing. This technique requires the tester to create a list of possible errors, defects, and failures, and to design tests that will identify defects associated with those errors.

4.4.2 (K2) Explain exploratory testing

- Exploratory testing is useful when there are few or inadequate specifications, or there is significant time pressure on the testing.
- Tests are simultaneously designed, executed, and evaluated while the tester learns about the test object.

- Exploratory testing is sometimes conducted using session-based testing, where testing is conducted within a defined time-box, and the tester uses a test charter containing test objectives. The test session is usually followed by a debriefing (involving discussion between the tester and stakeholders).
- Exploratory testing is more effective if the tester is experienced, has domain knowledge, and has a high degree of analytical skills, curiosity, and creativity.

4.4.3 (K2) Explain checklist-based testing

- Testers design, implement, and execute tests to cover conditions from a checklist.
- Checklists are built based on experience, what's important to the user, and how the software fails.
- Checklist items are often phrased in the form of questions.
- Checklists can support both functional and non-functional testing.
- Checklists should be updated regularly, and we should avoid letting the checklist become too long.
- In the absence of detailed test cases, checklist-based testing provides guidance and consistency.

4.5. Collaboration-based Test Approaches

4.5.1 (K2) Explain how to write user stories in collaboration with developers and business representatives

- User stories have three critical aspects, called the "3C's": Card, Conversation, and Confirmation (acceptance criteria), and are created by business representatives, developers, and testers together.
- We can use techniques such as brainstorming and mind mapping, this will allow the team to obtain a shared vision of what should be delivered.
- Good user stories are INVEST: Independent, Negotiable, Valuable, Estimable, Small, and Testable.
- The user story format is: As a [role], I want [goal], so that I can [resulting business value].

4.5.2 (K2) Classify the different options for writing acceptance criteria

- Acceptance criteria (the result of conversations) are the conditions that an implementation of a user story must meet to be accepted by stakeholders.
- There are several ways to write/document acceptance criteria for a user story :
  - Scenario-based: Example, Given/When/Then format of BDD...
  - Rule-oriented: Bullet point verification list, tabulated form of input-output mapping...

4.5.3 (K3) Use acceptance test-driven development (ATDD) to derive test cases

- In ATDD, we test valid scenarios first, then invalid ones, and then non-functional quality characteristics.

→ This part (4.5.3) is better understood with exam questions practice!

Questions from Chapter 4 in the ISTQB exam

4.1.1 (K2) Distinguish black-box test techniques, white-box test techniques and experience-based test techniques

Which of the following is a characteristic of experience-based test techniques?

a) Test cases are created based on detailed design information
b) Items tested within the interface code section are used to measure coverage
c) The techniques heavily rely on the tester's knowledge of the software and the business domain
d) The test cases are used to identify deviations from the requirements

Select ONE option.

a) Is not correct. This is a common characteristic of white-box test techniques. Test conditions, test cases, and test data are derived from a test basis that may include code, software architecture, detailed design, or any other source of information regarding the structure of the software. b) Is not correct. This is a common characteristic of white-box test techniques. Coverage is measured based on the items tested within a selected structure and the technique applied to the test basis c) Is correct. This is a common characteristic of experience-based test techniques. This knowledge and experience include expected use of the software, its environment, likely defects, and the distribution of those defects is used to define tests d) Is not correct. This is a common characteristic of black-box test techniques. Test cases may be used to detect gaps within requirements and the implementation of the requirements, as well as deviations from the requirements	FL-4.1.1

You perform system testing of an e-commerce web application and are provided with the following requirement:

REQ 05-017. If the total cost of purchases exceeds $100, the customer gets a 5% discount on subsequent purchases. Otherwise, the customer does not receive a discount.

Which test techniques will be MOST helpful in designing test cases based on this requirement?

a) White-box test techniques
b) Black-box test techniques
c) Experience-based test techniques
d) Risk-based test techniques

Select ONE option.

a) Is not correct. The document does not refer to the test object's internal structure but specifies the desired behavior of the test object. Therefore, white-box test techniques will not be helpful in designing test cases b) Is correct. The document is a requirement that specifies the desired behavior of the test object. Therefore, the most suitable test techniques in this case are the black-box test techniques (e.g., Boundary Value Analysis or Decision Table Testing) c) Is not correct. Although experience-based test techniques can be used to design test cases based on this document, black-box test techniques will be more suitable. The document describes a precise business rule and, in addition, wording like "exceeds $100" suggests the existence of important equivalence partition boundaries that should be tested using black-box test techniques like Boundary Value Analysis d) Is not correct. Risk-based test techniques are not a recognized type of test technique	FL-4.1.1

What is the MAIN difference between black-box test techniques and experience-based test techniques?

a) The test object
b) The test level at which the test technique is used
c) The test basis
d) The software development lifecycle (SDLC) in which the test technique can be used

Select ONE option.

a) Is not correct. In most cases both black-box test techniques and experience-based test techniques can be used for the same test objects b) Is not correct. Both black-box test techniques and experience-based test techniques can be used at all test levels c) Is correct. Black-box test techniques (also known as specification-based techniques) are based on an analysis of the specified behavior of the test object without reference to its internal structure. So, the test basis is usually a specification. Experience-based test techniques effectively use the knowledge and experience of testers for the design and implementation of test cases. This means that the tester, when designing tests, may not use the specification at all d) Is not correct. Experience-based test techniques can detect defects that may be missed using black-box (and white-box) test techniques. Hence, experience-based test techniques are complementary to black-box test techniques and white-box test techniques and both black-box test techniques and experience-based test techniques can be used in all SDLCs	FL-4.1.1

Which of the following statements BEST describes the difference between decision table testing and branch testing?

a) In decision table testing, the test cases are derived from the decision statements in the code. In branch testing, the test cases are derived from knowledge of the control flow of the test object.
b) In decision table testing, the test cases are derived from the specification that describes the business logic. In branch testing the test cases are based on anticipation of potential defects in the source code.
c) In decision table testing, the test cases are derived from knowledge of the control flow of the test object. In branch testing, test cases are derived from the specification that describes the business logic.
d) In decision table testing, the test cases are independent of how the software is implemented. In branch testing, test cases can be created only after the design or implementation of the code.

Select ONE option.

a) Is not correct. Decision table testing is a black-box test technique, so it is specification-based, not structure-based – the test cases are not based on the decisions in the source code. In branch testing, the test cases are derived from knowledge of the control flow of the test object b) Is not correct. Anticipation of potential defects is used in error guessing (an experience-based test technique), not in branch testing (a structure-based technique). In decision table testing, the test cases are derived from the specification that describes the business logic c) Is not correct. If a test case is based on the knowledge of the control flow of the test object, it is a white-box test technique. Decision table testing is typically based on an analysis of business logic, so it is a black-box test technique. In branch testing, test cases are not derived from the specification – this would make it a black-box test technique. Branch testing is a white-box test technique, where test cases are derived based on the source code structure d) Is correct. Decision table testing is a black-box test technique, so it is based on an analysis of the specified behavior of the test object without reference to its internal structure. Therefore, the test cases are independent of how the software is implemented. Branch testing is a white-box test technique, so test cases are based on an analysis of the test object's internal structure and processing. As the test cases are dependent on how the software is designed and coded, they can only be created after the design or implementation of the test object	FL-4.1.1

If test cases are derived from looking at the code, what type of test design technique is being used?

a. Black-box
b. White-box
c. Specification-based
d. Behavior-based

B is correct.

A, C and D are all black-box and use the specifications or requirements for the test design.

Which of the following test techniques uses the requirements specifications as a test basis?

a. Structure-based
b. Black-box
c. White-box
d. Exploratory

B is correct, per syllabus. Black-box testing is based off the requirements documents.

A and C are incorrect because these use the structure of the software as the test basis.

D is incorrect because exploratory testing is often done when there is no specification, thus giving the tester the opportunity to learn about the software while testing.

4.2.1 (K3) Use equivalence partitioning to derive test cases

You are testing a simplified apartment search form which has only two search criteria:

- floor (with three possible options: ground floor; first floor; second or higher floor)
- garden type (with three possible options: no garden; small garden; large garden)

Only apartments on the ground floor have gardens. The form has a built-in validation mechanism that will not allow you to use the search criteria which violate this rule.

Each test has two input values: floor and garden type. You want to apply equivalence partitioning (EP) to cover each floor and each garden type in your tests.

What is the **minimal** number of test cases to achieve 100% EP coverage?

a) 3
b) 4
c) 5
d) 6

Select ONE option

"Small garden" and "large garden" can go only with "ground floor", so we need two test cases with "ground floor" which cover these two "garden type" partitions. We need two more test cases to cover the two other "floor" partitions and a remaining "garden type" partition of "no garden". We need a total of four test cases: TC1 (ground floor, small garden) TC2 (ground floor, large garden) TC3 (first floor, no garden) TC4 (second or higher floor, no garden) a) Is not correct. b) Is correct. c) Is not correct. d) Is not correct.	FL-4.2.1

The system for selling cinema tickets calculates the discount type based on the client's birth year (BY) and on the current year (CY) as follows:

Let D be the difference between CY and BY, that is, D = CY – BY

- If D < 0 then print the error message "birth year cannot be greater than current year"
- If 0 ≤ D < 18 then apply the student discount
- If 18 ≤ D < 65 then apply no discount
- If D ≥ 65 then apply the pensioner discount

Your test suite already contains two test cases:

- BY = 1990, CY = 2020, expected result: no discount
- BY = 2030, CY = 2029, expected result: print the error message

Which of the following test data sets should be added to achieve full valid equivalence partitioning coverage for the discount type?

a) BY = 2001, CY = 2065
b) BY = 1900, CY = 1965
c) BY = 1965, CY = 1900
d) BY = 2011, CY = 2029
e) BY = 2000, CY = 2000

Select TWO options.

There are two equivalence partitions that are not yet covered, which correspond to "student discount" and "pensioner discount". a) Is not correct. CY – BY = 64, so these inputs correspond to the already covered "no discount" partition b) Is correct. CY – BY = 65, so these inputs correspond to a partition that is not yet covered ("pensioner discount") c) Is not correct. CY – BY = –65, so these inputs correspond to the already covered "error message" partition d) Is not correct. CY – BY = 18, so these inputs correspond to the already covered "no discount" partition e) Is correct. CY – BY = 0, so these inputs correspond to a partition that is not yet covered ("student discount")	FL-4.2.1

Customers of the TestWash car wash chain have cards with a record of the number of washes they have bought so far. The initial value is 0. After entering the car wash, the system increases the number on the card by one. This value represents the number of the current wash. Based on this number the system decides what discount the customer is entitled to.

For every tenth wash the system gives a 10% discount, and for every twentieth wash, the system gives a further 40% discount (i.e., a 50% discount in total).

Which of the following sets of input data (understood as the numbers of the current wash) achieves the highest equivalence partition coverage?

a) 19, 20, 30
b) 11, 12, 20
c) 1, 10, 50
d) 10, 29, 30, 31

Select ONE option.

a) Is correct. 19 covers the “no discount” partition, 20 covers the “50% discount” partition, and 30 covers the “10% discount” partition. These three values cover all three of the valid equivalence partitions b) Is not correct. 11 and 12 cover the “no discount” partition, while 20 covers the “50% discount” partition, so covering two of the three valid equivalence partitions c) Is not correct. 1 covers the “no discount” partition, while 10 and 50 cover the “10% discount” partition. The “50% discount” partition is not covered, so overall two of the three valid equivalence partitions are covered d) Is not correct. 29 and 31 cover the “no discount” partition, while 10 and 30 cover the “10% discount” partition. The “50% discount” partition is not covered, so overall two of the three valid equivalence partitions are covered	FL-4.2.1

You are testing a PIN validator which accepts valid PINs and rejects invalid PINs. A PIN is a sequence of digits. A PIN is valid if it consists of four digits and at least two of them are different.

Which of the following sets of input test data cover all equivalence partitions for this scenario?

a) 112, 1111, 1234, 123456
b) 1, 123, 1111, 1234
c) 12, 112, 1112, 11112
d) 1, 111, 1111, 11111

Select ONE option.

There are five equivalent partitions: (Input) Length: L1: Between 1 and 3 L2: 4 L3: above 4 (Input) Digits: D1: PIN that is all with the same digits D2: PIN with at least two different digits (Output) Two partitions: G1: Legal PIN (fulfils both L2 and D2) G2: Illegal PIN a) Is correct. 112 covers D2, L1, G2; 1111 covers D1, L2, G2; 1234 covers D2, L2, G1; 123456 covers D2, L3, G2 b) Is not correct. It doesn't cover L3 1 covers D1, L1, G2; 123 covers D2, L1, G2; 1111 covers D1, L2, G2; 12345 covers D2, L2, G1 c) Is not correct. It doesn't cover D1 12 covers D2, L1, G2; 112 covers D2, L1, G2; 1112 covers D2, L2, G1; 11112 covers D2, L3, G2 d) Is not correct. It doesn't cover D2 (and G1) 1 covers D1, L1, G2; 111 covers D1, L1, G2; 1111 covers D1, L2, G2; 11111 covers D1, L3, G2	FL-4.2.1

You are testing a banking application that allows a customer to withdraw 20, 100 or 500 dollars in a single transaction. The values are chosen from a drop-down list and no other values may be entered. How many equivalence partitions need to be tested to achieve 100% equivalence partition coverage?

a. 1
b. 2
c. 3
d. 4

D is correct. The values to be tested are 20, 100, 500 and no selection.

You are testing a machine that scores exam papers and assigns grades. Based on the score achieved the grades are as follows: 1-49 = F, 50-59 = D-, 60-69 = D, 70-79 = C, 80-89 = B, 90-100=A

If you apply equivalence partitioning, how many test cases will you need to achieve minimum test coverage?

a. 6
b. 8
c. 10
d. 12

B is correct. You need a test for the invalid too low (0 or less), one for each valid partition, and one for invalid too high (>100). It may not allow you to enter a value > 100, but it should be tested to make sure it provides a proper error.

4.2.2 (K3) Use boundary value analysis to derive test cases

You are testing a system that calculates the final course grade for a given student.

The final grade is assigned based on the final result, according to the following rules:

- 0 – 50 points: failed
- 51 – 60 points: fair
- 61 – 70 points: satisfactory
- 71 – 80 points: good
- 81 – 90 points: very good
- 91 – 100 points: excellent

You have prepared the following set of test cases:

	Final result	Final grade
TC1	91	excellent
TC2	50	failed
TC3	81	very good
TC4	60	fair
TC5	70	satisfactory
TC6	80	good

What is the 2-value Boundary Value Analysis (BVA) coverage for the final result that is achieved with the existing test cases?

a) 50%
b) 60%
c) 33.3%
d) 100%

Select ONE option.

There are 12 boundary values for the final result values: 0, 50, 51, 60, 61, 70, 71, 80, 81, 90, 91, and 100. The test cases cover six of them (TC1 – 91, TC2 – 50, TC3 – 81, TC4 – 60, TC5 – 70 and TC7 – 51). Therefore, the test cases cover 6/12 = 50%. a) Is correct. b) Is not correct. c) Is not correct. d) Is not correct.	FL-4.2.2

You are testing a temperature control system for a horticultural cold storage facility. The system receives the temperature (in full degrees Celsius) as the input. If the temperature is between 0 and 2 degrees inclusive, the system displays the message “temperature OK”. For lower temperatures, the system displays the message "temperature too low" and for higher temperatures it displays the message “temperature too high”.

Using two-value boundary value analysis, which of the following sets of test inputs provides the highest level of boundary value coverage?

a) –1, 3
b) 0, 2
c) –1, 0, 2, 3
d) –2, 0, 2, 4

Select ONE option.

There are three equivalence partitions: {…, –2, –1}, {0, 1, 2}, {3, 4, …}. For 2-value BVA all the boundary values for all the equivalence partitions must be covered. The boundary values are –1 (for the “temperature too low” partition), 0, 2 (for the “temperature OK” partition) and 3 (for the “temperature too high” partition). Thus: a) Is not correct b) Is not correct c) Is correct. The correct option is: –1, 0, 2, 3 d) Is not correct	FL-4.2.2

W A developer was asked to implement the following business rule:

INPUT: value (integer number) IF (value ≤ 100 OR value ≥ 200) THEN write “value incorrect” ELSE write “value OK”

You design the test cases using 2-value boundary value analysis.

Which of the following sets of test inputs achieves the greatest coverage?

a) 100, 150, 200, 201
b) 99, 100, 200, 201
c) 98, 99, 100, 101
d) 101, 150, 199, 200

Select ONE option.

The equivalence partitions are: {…, 99, 100}, {101, 102, …, 198, 199}, {200, 201, …}. Thus, there are 4 boundary values, which are: 100, 101, 199 and 200. In 2-value BVA, for each boundary value there are two coverage items (the boundary value and its closest neighbor belonging to the adjacent partition). As the closest neighbors are also boundary values in the adjacent partition, then there are just four coverage items. Thus: a) Is not correct. Only 100 and 200 are valid coverage items for 2-value BVA, so we achieve 50% coverage b) Is not correct. Only 100 and 200 are valid coverage items for 2-value BVA, so we achieve 50% coverage c) Is not correct. Only 100 and 101 are valid coverage items for 2-value BVA, so we achieve 50% coverage d) Is correct. 101, 199 and 200 are valid coverage items for 2-value BVA, so we achieve 75% coverage	FL-4.2.2

You are testing a form that verifies the correctness of the length of the password given as input. The form accepts a password with the correct length and rejects a password that is too short or too long. The password length is correct if it has between 6 and 12 characters inclusive. Otherwise, it is considered incorrect.

At first, the form is empty (password length = 0). You apply boundary value analysis to the "password length" variable.

Your set of test cases achieves 100% 2-value boundary value coverage. The team decided that due to the high risk of this component, test cases should be added to ensure 100% 3-value boundary value coverage.

Which additional password lengths should be tested to achieve this?

a) 4, 5, 13, 14
b) 7, 11
c) 1, 5, 13
d) 1, 4, 7, 11, 14

Select ONE option.

The domain for the password length has three equivalence partitions: • passwords too short {0, 1, …, 4, 5} • passwords OK {6, 7, …, 11, 12} • passwords too long {13, 14, …} To achieve full coverage for 3-value BVA we need to test the following values: 0, 1, 4, 5, 6, 7, 11, 12, 13, 14. Since 2-value BVA is already covered, this means that we have already tested the passwords of length: 0, 5, 6, 12 and 13. This means that the additional lengths that need to be covered to move from 2-value to 3-value are: 1, 4, 7, 11 and 14. Thus: a) Is not correct b) Is not correct c) Is not correct d) Is correct	FL-4.2.2

You are testing a scale system that determines shipping rates for a regional web-based auto parts distributor. Due to regulations, shipments cannot exceed 100 lbs. You want to include boundary value analysis as part of your black-box test design.

How many tests will you need to execute to achieve 100% two-value boundary value analysis?

Weight	0 to 10 lbs.	11 to 25 lbs.	26 to 50 lbs.	51 lbs. to 100
Shipping Cost	$5.00	$7.50	$12.00	$17.00

a. 4
b. 8
c. 10
d. 12

C is correct. 2 per valid weight range plus one for a negative weight and one for a weight exceeding 100 lbs (-1, 0, 10, 11, 25, 26, 50, 51, 100, 101).

You are testing a thermostat for a heating/air conditioning system. You have been given the following requirements:

- When the temperature is below 70 degrees, turn on the heating system
- When the temperature is above 75 degrees, turn on the air conditioning system
- When the temperature is between 70 and 75 degrees, inclusive, turn on fan only

Which of the following is the minimum set of test temperature values to achieve 100% two-value boundary value analysis coverage?

a. 70, 75
b. 65, 72, 80
c. 69, 70, 75, 76
d. 70, 71, 74, 75, 76

C is correct.

For the heating system, the values to test are 69, 70
For the air conditioning system, the values are 75, 76
For the fan only, the values are 69, 70, 75, 76

<-- 69 | 70 – 75 | 76 →

The proper test set combines all these values, 69, 70, 75, 76

4.2.3 (K3) Use decision table testing to derive test cases

Your favorite bicycle daily rental store has just introduced a new Customer Relationship Management system and asked you, one of their most loyal members, to test it.

The implemented features are as follows:

- Anyone can rent a bicycle, but members receive a 20% discount
- However, if the return deadline is missed, the discount is no longer available
- After 15 rentals, members get a gift: a T-Shirt

Decision table describing the implemented features looks as follows:

Conditions	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**
Being a member	T	T	T	T	F	F	F	F
Missed deadline	T	F	T	F	T	F	F	T
15th rental	F	F	T	T	F	F	T	T
Actions								
20% discount		X		X				
Gift T-Shirt			X	X				X

Based ONLY on the feature description of the Customer Relationship Management system, which of the above rules describes an impossible situation?

a) R4
b) R2
c) R6
d) R8

Select ONE option.

a) Is not correct. A member without a missed deadline can get a discount and a gift T-Shirt after 15 bicycle rentals b) Is not correct. A member without a missed deadline can get a discount but no gift T-Shirt until they rented a bicycle 15 times c) Is not correct. Non-members cannot get a discount, even if they did not miss a deadline yet d) Is correct. No discount as a non-member that has also missed a deadline, but only members can receive a gift T-Shirt. Hence, the action is not correct	FL-4.2.3

You are working on a project to develop a system to analyze driving test results. You have been asked to design test cases based on the following decision table.

	R1	R2	R3
C1: First attempt at the exam?	-	-	F
C2: Theoretical exam passed?	T	F	-
C3: Practical exam passed?	T	-	F
Issue a driving license?	X		
Request additional driving lessons?			X
Request to take the exam again?		X	

What test data will show that there are contradictory rules in the decision table?

a) C1 = T, C2 = T, C3 = F
b) C1 = T, C2 = F, C3 = T
c) C1 = T, C2 = T, C3 = T and C1 = F, C2 = T, C3 = T
d) C1 = F, C2 = F, C3 = F

Select ONE option.

a) Is not correct. The combination (T, T, F) does not match any rule. This is an example of omission, not a contradiction b) Is not correct. The combination (T, F, T) matches only one column, R2, so there is no contradiction c) Is not correct. Both combinations (T, T, T) and (F, T, T) match only one column, R1, so there is no contradiction d) Is correct. The combination (F, F, F) matches both R2 and R3, but R2 and R3 have different actions, so this shows a contradiction between R2 and R3.	FL-4.2.3

You are designing test cases based on the following decision table.

	R1	R2	R3	R4	R5	R6	R7
C1: Age	0-18	19-65	19-65	>65	0-18	19-65	>65
C2: Experience	-	0-4	>4	-	-	-	-
C3: Registered?	NO	NO	NO	NO	YES	YES	YES
Category	A	A	B	B	B	D	C

So far you have designed the following test cases:

- TC1: 19-year-old, unregistered man with no experience;
  expected result: category A
- TC2: 65-year-old, unregistered woman with 5 years of experience;
  expected result: category B
- TC3: 66-year-old, registered man with no experience;
  expected result: category C
- TC4: 65-year-old, registered woman with 4 years of experience;
  expected result: category D

Which of the following test cases, when added to the existing set of test cases, will increase the decision table coverage?

a) 66-year-old, unregistered man with no experience; expected result: category B
b) 55-year-old, unregistered woman with 2 years of experience; expected result: category A
c) 19-year-old, registered woman with 5 years of experience; expected result: category D
d) No additional test case can increase the already achieved decision table coverage

Select ONE option.

Test cases TC1, TC2, TC3 and TC4 cover, respectively, rules R2, R3, R7 and R6 in the decision table. a) Is correct. The conditions “66-year-old”, “unregistered” and “no experience” match rule R4, which is not covered by the existing test cases, so after adding this test case, the decision table coverage will increase b) Is not correct. The conditions “55-year-old”, “unregistered” and “2 years of experience” match rule R2, already covered by TC1. So adding this test case will not increase the coverage c) Is not correct. The conditions “19-year-old”, “registered” and “5 years of experience” match rule R6, already covered by TC4. So adding this test case will not increase the coverage d) Is not correct. The existing test cases cover only 4 out of 7 columns of the decision table. The coverage can be increased by adding test cases that cover yet uncovered columns, that is, R1, R4 and R5	FL-4.2.3

You have been given the following conditions and results from those condition combinations. Given this information, using the decision table technique, what is the minimum number of test cases you would need to test these conditions?

Conditions:
Valid cash
Valid credit card
Valid debit card
Valid pin
Bank accepts
Valid Selection
Item in Stock
Results:
Reject Cash
Reject Card
Error Message
Return Cash
Refund Card
Sell Item

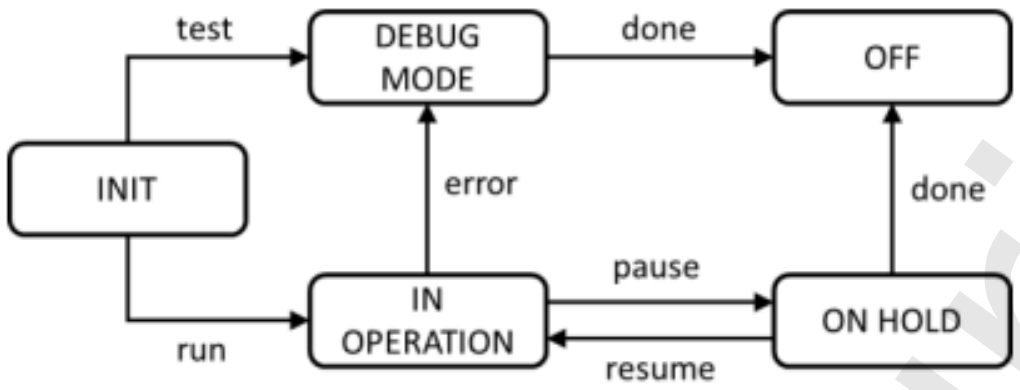
a. 7
b. 13
c. 15
d. 18

C is correct. See table below.

Conditions:	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	
Valid cash	T	T	T	F												
Valid credit card					T	T	T	T	F							
Valid debit card										T	T	T	T	T	F	
Valid pin										T	F	T	T	T		
Bank accepts					T	F	T	T		T		F	T	T		
Valid Selection	T	F	T		T		F	T		T			F	T		
Item in Stock	T		F		T			F		T				F		
Results:																
Reject Cash				X												
Reject Card						X			X		X	X			X	
Error Message		X	X	X			X	X					X	X		
Return Cash		X	X													
Refund Card							X	X					X	X		
Sell Item	X				X					X						

4.2.4 (K3) Use state transition testing to derive test cases

You test a system whose lifecycle is modeled by the state transition diagram shown below. The system starts in the INIT state and ends its operation in the OFF state.



What is the MINIMAL number of test cases to achieve valid transitions coverage?

a) 4
b) 2
c) 7
d) 3

Select ONE option.

"test" and "error" transitions cannot occur in one test case. Neither can both "done" transitions. This means we need at least three test cases to achieve transition coverage. For example: TC1: test, done TC2: run, error, done TC3: run, pause, resume, pause, done Hence a) Is not correct. b) Is not correct. c) Is not correct. d) Is correct.	FL-4.2.4

You are applying state transition testing to the hotel room reservation system modeled by the following state transition table, with 4 states and 5 different events:

	Events				
State	Available	NotAvailable	ChangeRoom	Cancel	Pay
S1: Requesting	S2	S3			
S2: Confirmed			S1	S4	S4
S3: Waiting list	S2			S4	
S4: End					

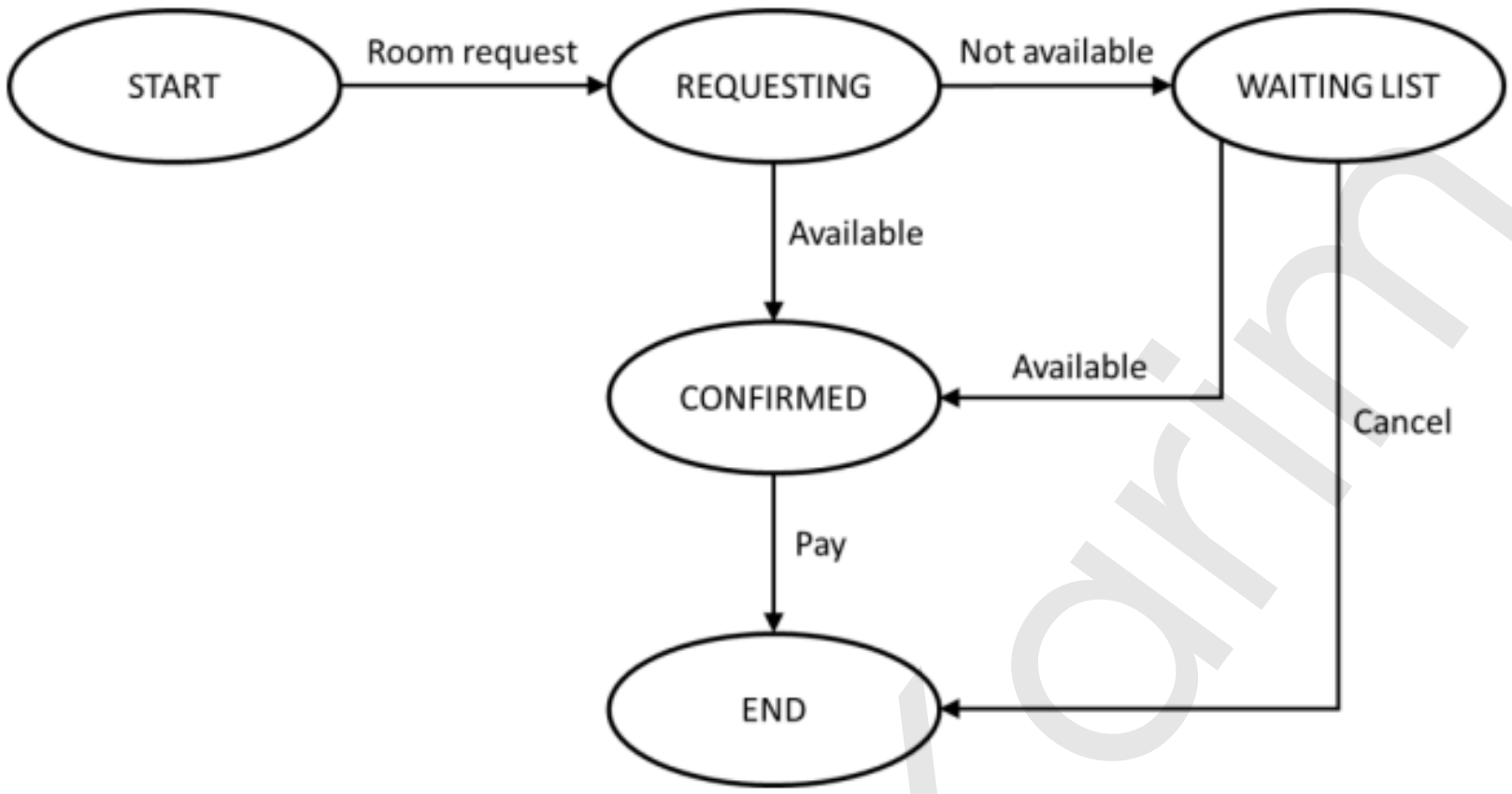
Assuming all test cases start in the 'Requesting' state, which of the following test cases, represented as sequences of events, achieves the highest valid transitions coverage?

a) NotAvailable, Available, ChangeRoom, NotAvailable, Cancel
b) Available, ChangeRoom, NotAvailable, Available, Pay
c) Available, ChangeRoom, Available, ChangeRoom, NotAvailable
d) NotAvailable, Cancel, ChangeRoom, Available, Pay

Select ONE option.

a) Is not correct. This sequence of five events covers 4 different valid transitions (both "NotAvailable" events correspond to the same transition between S1 and S3). This test case covers 4 out of 7 valid transitions b) Is correct. This sequence of five events covers 5 different transitions (the first "Available" event corresponds to a transition between S1 and S2, and the second "Available" event corresponds to a transition between S3 and S2, so two different transitions are covered). This test case covers 5 out of 7 valid transitions and achieves the highest valid transitions coverage c) Is not correct. This sequence of five events covers 3 different transitions (both "Available" events correspond to the same transition from S1 to S2; both "ChangeRoom" events correspond to the same transition from S2 to S1). This test case covers 3 out of 7 valid transitions d) Is not correct. This sequence of five events does not represent a feasible test case, because after "Cancel" the system ends up in the End state and no further valid transitions can be executed	FL-4.2.4

You are designing test cases based on the following state transition diagram:



What is the MINIMUM number of test cases required to achieve 100% valid transitions coverage?

a) 3
b) 2
c) 5
d) 6

Select ONE option.

The following three transitions: "REQUESTING -> CONFIRMED" "WAITING LIST -> CONFIRMED" "WAITING LIST -> END" cannot appear in the same test case, which suggests that at least three test cases are required. All the other transitions can appear in combination with one or more of these three transitions, so we need a minimum of three test cases. In fact, only three sequences are possible: TC1: START (Room request) → REQUESTING (Available) → CONFIRMED (Pay) → END TC2: START (Room request) → REQUESTING (Not available) → WAITING LIST (Available) → CONFIRMED (Pay) → END TC3: START (Room request) → REQUESTING (Not available) → WAITING LIST (Cancel) → END Thus: a) Is correct b) Is not correct c) Is not correct d) Is not correct	FL-4.2.4

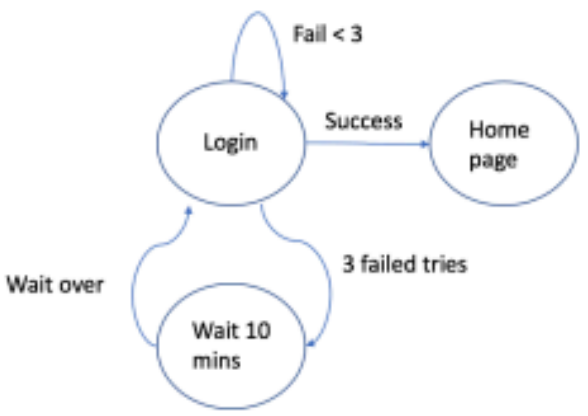
You have been given the following requirement:

A user must log in to the system with a valid username and password. If they fail to enter the correct combination three times, they will receive an error and will have to wait 10 minutes before trying again. The test terminates when the user successfully logs in.

How many test cases are needed to provide 100% state transition coverage?

a. 1
b. 2
c. 4
d. 5

a is correct. Per the diagram below, only one test is needed: Login, Fail, Fail, Fail =3, Wait, Login, Home. If you were required to exit after the Wait, a second test would be required but the question doesn't indicate that an exit is required.



4.3.1 (K2) Explain statement testing

Your test suite achieved 100% statement coverage. What is the consequence of this fact?

a) Each instruction in the code that contains a defect has been executed at least once
b) Any test suite containing more test cases than your test suite will also achieve 100% statement coverage
c) Each path in the code has been executed at least once
d) Every combination of input values has been tested at least once

Select ONE option.

a) Is correct. Since 100% statement coverage is achieved, every statement, including the ones with defects, must have been executed and evaluated at least once b) Is not correct. Coverage depends on what is tested, not on the number of test cases. For example, for code "if (x==0) y=1", one test case (x=0) achieves 100% statement coverage, but two test cases (x=1) and (x=2) together achieve only 50% statement coverage c) Is not correct. If there is a loop in the code there may be an infinite number of possible paths, so it is not possible to execute all the possible paths in the code d) Is not correct. Exhaustive testing is not possible (see the seven testing principles section in the syllabus). For example, for code "input x; print x" any single test with arbitrary x achieves 100% statement coverage, but covers one input value	FL-4.3.1

Your test suite S for a program P achieves 100% statement coverage. It consists of three test cases, each of which achieves 50% statement coverage.

Which of the following statements is CORRECT?

a) Executing S will cause all possible failures in P
b) S achieves 100% branch coverage for P
c) Every executable statement in P containing a defect has been run at least once during the execution of S
d) After removing one test case from S, the remaining two test cases will still achieve 100% statement coverage

Select ONE option.

a) Is not correct. A line with a defect, when executed, does not have to cause a failure. For example, a line x := y / z will cause a failure *only* when z equals 0 b) Is not correct. 100% statement coverage does not guarantee 100% branch coverage. For example, a test case with x=0 for the code 1. IF (x=0) THEN 2. A; 3. ENDIF achieves 100% statement coverage but does not cover the branch from 1 to 3 c) Is correct. 100% statement coverage means that each executable statement was executed at least once d) Is not correct. The removed test case may provide coverage of some statements that are not covered by either of the other two test cases, in which case the remaining two test cases together will not achieve 100% statement coverage	FL-4.3.1

You run two test cases, T1 and T2, on the same code. Test T1 achieved 40% statement coverage and test T2 achieved 65% statement coverage.

Which of the following sentences must be necessarily true?

a) The test suite composed with tests T1 and T2 achieves 105% statement coverage
b) There exists at least one statement that must have been executed by both T1 and T2
c) At least 5% of the statements in the code under test are non-executable
d) The test suite composed of tests T1 and T2 achieves full branch coverage

Select ONE option.

a) Is not correct. Coverage is always defined as the percentage of the covered elements. Therefore, it cannot exceed 100% b) Is correct. If the statements executed by T1 and T2 were disjoint, the coverage of the test suite {T1, T2} would be 105%, which is impossible (see answer a). Therefore, at least 5% of executable statements must have been executed by both T1 and T2 c) Is not correct. Statement coverage does not tell us anything about the number of non-executable statements in the code d) Is not correct. Even if a test suite achieves full statement coverage, this does not imply achieving full branch coverage	FL-4.3.1

4.3.2 (K2) Explain branch testing

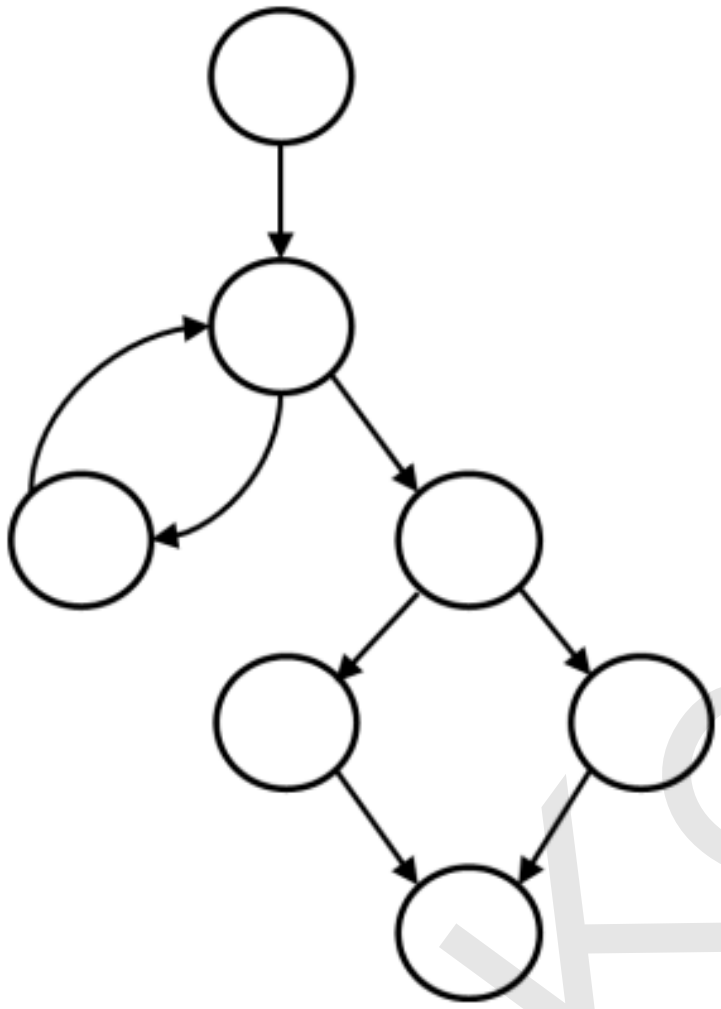
Which of the following statements about branch testing is CORRECT?

a) If a program includes only unconditional branches, then 100% branch coverage can be achieved without executing any test cases
b) If the test cases exercise all unconditional branches in the code, then 100% branch coverage is achieved
c) If 100% statement coverage is achieved, then 100% branch coverage is also achieved
d) If 100% branch coverage is achieved, then all decision outcomes in each decision statement in the code are exercised

Select ONE option.

a) Is not correct. In this case one test case is still needed since there is at least one (unconditional) branch to be covered b) Is not correct. Covering only unconditional branches does not imply covering all conditional branches c) is not correct. 100% branch coverage implies 100% statement coverage, not otherwise. For example, for an IF decision without the ELSE, one test is enough to achieve 100% statement coverage, but it only achieves 50% branch coverage d) Is correct. Each decision outcome corresponds to a conditional branch, so 100% branch coverage implies 100% decision coverage	FL-4.3.2

You want to apply branch testing to the code represented by the following control flow graph.



How many coverage items do you need to test?

a) 2
b) 4
c) 8
d) 7

Select ONE option.

In branch testing the coverage items are branches, which are represented by the edges of a control flow graph. There are 8 edges in the control flow graph. Thus: a) Is not correct b) Is not correct c) Is correct d) Is not correct	FL-4.3.2

. If you are testing a module of code, how do you determine the level of branch coverage you have achieved?

a. By taking the number of branches you have tested and dividing that by the total number of executable statements in the module
b. By taking the number of branches you have tested and dividing that by the total number of branches in the module
c. By taking the number of branches you have tested and dividing that by the total lines of code in the module
d. By taking the number of branches you have tested and dividing that by the total number of test cases you have executed for the module

B is correct. Branch coverage looks at the number of branches tested versus the number of branches in the code under test.

. If you have a section of code that has one simple IF statement, how many tests will be needed to achieve 100% branch coverage?

a. 1
b. 2
c. 5
d. Unknown with this information

B is correct. A simple IF statement will be composed of If ... then ... else.... end if. There are two branch outcomes, one for the result of the If being true and one for it being false. Since 100% branch coverage requires at least one test case for each branch outcome, two tests are needed.

A and C are incorrect because these are the wrong numbers of tests.

D would be correct if this weren't defined as a simple if statement because a complex if statement could include more than two outcomes.

4.3.3 (K2) Explain the value of white-box testing

Which of the following is NOT true for white-box testing?

a) During white-box testing the entire software implementation is considered
b) White-box coverage metrics can help identify additional tests to increase code coverage
c) White-box test techniques can be used in static testing
d) White-box testing can help identify gaps in requirements implementation

Select ONE option.

a) Is not correct. The fundamental strength of white-box test techniques is that the entire software implementation is taken into account during testing b) Is not correct. White-box coverage measures provide an objective measure of coverage and provide the necessary information to allow additional tests to be generated to increase this coverage c) Is not correct. White-box test techniques can be used to perform reviews (static testing) d) Is correct. This is the weakness of the white-box test techniques. They are not able to identify the missing implementation, because they are based solely on the test object structure, not on the requirements specification	FL-4.3.3

Why does white-box testing facilitate defect detection even when the software specification is vague, outdated or incomplete?

a) Test cases are designed based on the structure of the test object rather than the specification
b) For each white-box test technique the coverage can be well-defined and easily measured
c) White-box test techniques are very well designed to detect omissions in the requirements
d) White-box test techniques can be used in both static testing and dynamic testing

Select ONE option.

a) Is correct. A fundamental strength that all white-box test techniques share is that the entire software implementation is taken into account during testing, which facilitates defect detection even when the software specification is vague, outdated or incomplete. This means white-box testing can find defects such as an extra feature added to the code (either accidentally or deliberately) that is not supposed to be there, which black-box testing cannot detect b) Is not correct. The fact that the coverage can be precisely defined is not the right reason. The achieved level of coverage would have much more impact than the possibility to measure the coverage c) Is not correct. If the software does not implement one or more requirements, white-box testing is unlikely to detect the resulting defects of omission d) Is not correct. While this is true, this is not the right answer, because there is no connection between the capability to be used in both static testing and dynamic testing and the claim that white-box testing facilitates defect detection with poor specifications	FL-4.3.3

How can white-box testing be useful in support of black-box testing?

a) White-box coverage measures can help testers evaluate black-box tests in terms of the code coverage achieved by these black-box tests
b) White-box coverage analysis can help testers identify unreachable fragments of the source code
c) Branch testing subsumes black-box test techniques, so achieving full branch coverage guarantees achieving full coverage of any black-box technique
d) White-box test techniques can provide coverage items for black-box techniques

Select ONE option.

a) Is correct. Performing only black-box testing does not provide a measure of actual code coverage. White-box coverage measures provide an objective measurement of coverage and provide the necessary information to allow additional tests to be generated to increase this coverage, and subsequently increase confidence in the code b) Is not correct. This statement is correct, but it has nothing to do with black-box testing c) Is not correct. In general there are no subsumes relationships between white-box and black-box techniques d) Is not correct. White-box techniques are used to design tests based on the test object itself, while black-box techniques are used to design tests based on the specification. Therefore, there is no relation between coverage items derived from these two types of techniques	FL-4.3.3

You are working on a project with very tight deadlines. The code is being developed but is not yet executable. What type of testing could you apply that would help find defects now?

a. Black-box
b. White-box
c. Experience-based
d. Factor-based

B is correct. You have code that can be structurally reviewed for items such as the proper handling of branching.

A and C are not correct because nothing can be executed yet.

D is not correct because it's not a type of testing.

4.4.1 (K2) Explain error guessing

Which of the following BEST describes the concept behind error guessing?

a) Error guessing involves using your knowledge and experience of defects found in the past and typical errors made by developers
b) Error guessing involves using your personal experience of development and the errors you made as a developer
c) Error guessing requires you to imagine that you are the user of the test object and to guess errors the user could make interacting with it
d) Error guessing requires you to rapidly duplicate the development task to identify the sort of errors a developer might make

Select ONE option.

a) Is correct. The basic concept behind error guessing is that the tester tries to guess what errors may have been made by the developer and what defects may be in the test object based on past experience (and sometimes checklists) b) Is not correct. Although a testers who used to be a developer may use their personal experience to help them when performing error guessing, the test technique is not based on prior knowledge of development c) Is not correct. Error guessing is not a usability technique for guessing how users may fail to interact with the test object d) Is not correct. Duplicating the development task has several flaws that make it impractical, such as the tester having equivalent skills to the developer and the time involved to perform the development. It is not error guessing	FL-4.4.1

Which of the following is NOT anticipated by the tester while applying error guessing?

a) The developer misunderstood the formula in the user story for calculating the interest
b) The developer wrote "FA = A*(1+IR^N)" instead of "FA = A*(1+IR)^N" in the source code
c) The developer missed the seminar on new compound interest rate legislation
d) The accuracy of the interest calculated by the system is not precise enough

Select ONE option.

Error guessing is about anticipating the errors, defects and failures based on the tester's knowledge. a) Is not correct. This is an example of anticipating the developer's error b) Is not correct. This is an example of anticipating the defect c) Is correct. This is an example of a potential root cause of a defect, which is neither an error, defect nor failure, and difficult for the tester to anticipate d) Is not correct. This is an example of anticipating a failure, perhaps based on experience of previous systems in this application domain	FL-4.4.1

Consider the following list:

- Correct input not accepted
- Incorrect input accepted
- Wrong output format
- Division by zero

What test technique is MOST PROBABLY used by the tester who uses this list when performing testing?

a) Exploratory testing
b) Fault attack
c) Checklist-based testing
d) Boundary value analysis

Select ONE option.

a) Is not correct. Exploratory testing uses test charters, not a list of possible defects/failures. Although exploratory testing can incorporate the use of other test techniques, in this case fault attack is the most likely option
b) Is correct. This is a list of possible failures. Fault attacks are a methodical approach to the implementation of error guessing and require the tester to create or acquire a list of possible errors, defects and failures, and to design tests that will identify defects associated with the errors, expose the defects, or cause the failures
c) Is not correct. The tester is using a checklist of items to support their testing. Both error guessing and checklist-based testing use such lists, however, the list here is of possible failures, not test conditions, and so the MOST PROBABLE test technique is fault attack, which focuses on errors, defects and failures
d) Is not correct. BVA is based on an analysis of boundary values of equivalence partitions. The above list does not mention equivalence partitions or their boundaries

---

If you are using error guessing to target your testing, which type of testing are you doing?

a. Specification-based
b. Structure-based
c. Experience-based
d. Reference-based

C is correct. This is an experience-based technique.

What is error guessing?

a. A testing technique used to guess where a developer is likely to have made a mistake
b. A technique used for assessing defect metrics
c. A development technique to verify that all error paths have been coded
d. A planning technique used to anticipate likely schedule variances due to faults

A is correct. Error guessing is a technique used to anticipate where developers are likely to make errors and to create tests to cover those areas.

B, C and D are not correct.

4.4.2 (K2) Explain exploratory testing

Which of the following is a good reason to use experience-based testing?

a. You can find defects that might be missed by more formal techniques
b. You can test for defects that only experienced users would encounter
c. You can target the developer's efforts to the areas that users will be more likely to use
d. It is supported by strong tools and can be automated

A is correct. Experience-based testing is often used to fill in the gaps left by the more formal testing techniques.

B is not correct because it is used by experienced testers and has nothing to do with the experience level of the users.

C is not correct because it is a test technique, not a development technique.

D is not correct. There is not much tool support for these techniques and automation is not usually a goal because the effectiveness depends on the experience of the tester.

In your project there has been a delay in the release of a brand-new application and test execution started late, but you have very detailed domain knowledge and good analytical skills. The full list of requirements has not yet been shared with the team, but management is asking for some test results to be presented.

Which test technique fits BEST in this situation?

a) Checklist-based testing
b) Error guessing
c) Exploratory testing
d) Branch testing

Select ONE option.

a) Is not correct. This is a new product. You probably do not have a checklist yet and test conditions might not be known due to missing requirements b) Is not correct. This is a new product. You probably do not have enough information to make correct error guesses c) Is correct. Exploratory testing is most useful when there are few known specifications and/or there is a pressing timeline for testing d) Is not correct. Branch testing is time-consuming, and your management is asking about some test results now. Also, branch testing does not involve domain knowledge	FL-4.4.2

Which of the following is true about exploratory testing?

a) Test cases are designed before the exploratory testing session starts
b) The tester can perform test execution, but cannot perform test design
c) Exploratory testing results are good predictors of the number of remaining defects
d) During exploratory testing the tester may use black-box test techniques

Select ONE option.

a) Is not correct. In exploratory testing, test cases are usually created during the exploratory testing session, alongside test analysis, test implementation and test execution b) Is not correct. In exploratory testing, tests are simultaneously designed, executed, and evaluated while the tester learns about the test object c) Is not correct. Exploratory testing results depend heavily on the tester's experience, so even if the results of exploratory testing can be used as a predictor of risk and used to assess whether there will be fewer or more defects, for example, compared to the previous exploratory testing session, they are not a good example of reliable defect prediction models that can predict the number of remaining defects d) Is correct. During exploratory testing, the testers can use any techniques that they find useful	FL-4.4.2

Which TWO of the following statements provide the BEST rationale for using exploratory testing?

a) Testers have not been allocated enough time for test design and test execution
b) The existing test strategy requires that testers use formal, black-box test techniques
c) The specification is written in a formal language that can be processed by a tool
d) Testers are the members of an agile team and have good programming skills
e) Testers are experienced in the business domain and have good analytical skills

Select TWO options.

Exploratory testing is useful when there are few or inadequate specifications or there is significant time pressure on the testing. Exploratory testing is also useful to complement other more formal test techniques. Exploratory testing will be more effective if the tester is experienced, has domain knowledge and has a high degree of essential skills, like analytical skills, curiosity and creativeness. Thus: a) Is correct. Exploratory testing is useful when there are few or inadequate specifications or there is significant time pressure on the testing b) Is not correct. Exploratory testing is not a black-box test technique c) Is not correct. Exploratory testing is useful when the specifications are poorly written d) Is not correct. Programming skills have nothing to do with exploratory testing in principle e) Is correct. Exploratory testing will be more effective if the tester is experienced, has domain knowledge and has a high degree of essential skills, like analytical skills, curiosity and creativeness	FL-4.4.2

When exploratory testing is conducted using time-boxing and test charters, what is it called?

a. Schedule-based testing
b. Session-based testing
c. Risk-based testing
d. Formal chartering

B is correct. This is often called session-based testing and may use session sheets.

A is not correct. Exploratory doesn't usually comply to schedules but rather allows the tester to explore and learn about the software. Coverage is difficult to assess which is one of the reasons it is difficult to match the time spent to the amount accomplished.

C is not correct. This may be one of the forms of risk-based testing, but it is not entirely RBT.

D is not correct as this is not actually a testing term.

4.4.3 (K2) Explain checklist-based testing

You are testing a mobile application that allows customers to access and manage their bank accounts. You are running a test suite that involves evaluating each screen, and each field on each screen, against a general list of user interface best practices derived from a popular book on the topic that maximizes attractiveness, ease-of-use, and accessibility for such applications. Which of the following options BEST categorizes the test technique you are using?

a) Black-box
b) Exploratory
c) Checklist-based
d) Error guessing

Select ONE option.

a) Is not correct. The book provides general guidance, and is not a formal requirements document, a specification, or a set of use cases, user stories, or business processes b) Is not correct. While you could consider the list as a set of test charters, it more closely resembles the list of test conditions to be checked c) Is correct. The list of user interface best practices is the list of test conditions to be systematically checked d) Is not correct. The tests are not focused on failures that could occur, but rather on knowledge about what is important for the user, in terms of usability	FL-4.4.3

Which of the following BEST describes how using checklist-based testing can result in increased coverage?

a) Checklist items can be defined at a sufficiently low level of detail, so the tester can implement and execute detailed test cases based on these items
b) Checklists can be automated, so each time an automated test execution covers the checklist items, it results in additional coverage
c) Each checklist item should be tested separately and independently, so the elements cover different areas of the software
d) Two testers designing and executing tests based on the same high-level checklist items will typically perform the testing in slightly different ways

Select ONE option.

a) Is not correct. Although it is true that the tester can implement and execute detailed test cases based on the checklist, it does not explain how this would result in increased coverage b) Is not correct. Checklist items should not be automated. But even if they are, the automated test scripts always execute the tests in the same way, which usually does not result in increased coverage c) Is not correct. It is true that each checklist item should be tested separately and independently. But this impacts the test execution order and does not impact the achieved coverage, and so does not result in increased coverage d) Is correct. If the checklists are high-level, some variability in the actual testing is likely to occur, resulting in potentially greater coverage but less repeatability. If two testers follow a checklist of high-level items, each of them may use different test data, test steps, etc. This way, one tester will probably cover some areas not covered by the other tester and this will result in increased coverage	FL-4.4.3

4.5.1 (K2) Explain how to write user stories in collaboration with developers and business representatives

Which of the following BEST describe the collaborative approach to user story writing?

a) User stories are created by testers and developers and then accepted by business representatives
b) User stories are created by business representatives, developers, and testers together
c) User stories are created by business representatives and verified by developers and testers
d) User stories are created in a way that they are independent, negotiable, valuable, estimable, small, and testable

Select ONE option.

a) Is not correct. Collaborative user story writing means that all stakeholders create the user stories collaboratively, to obtain the shared vision b) Is correct. Collaborative user story writing means that all stakeholders create the user stories collaboratively, to obtain the shared vision c) Is not correct. Collaborative user story writing means that all stakeholders create the user stories collaboratively, to obtain the shared vision d) Is not correct. This is the list of properties that each user story should have, not the description of the collaboration-based approach	FL-4.5.1

Which collaborative user story writing practice enables the team to achieve a collective understanding of what needs to be delivered?

a) Planning poker, so that a team can achieve consensus on the effort needed to implement a user story
b) Reviews, so that a team can detect inconsistencies and contradictions in a user story
c) Iteration planning, so that user stories with the highest business value for a customer can be prioritized for implementation
d) Conversation, so that team members can understand how the software will be used

Select ONE option.

a) Is not correct. Planning poker can estimate effort for a user story that is already written. It does not help in understanding what should be delivered b) Is not correct. Reviews are not a collaborative user story writing practice c) Is not correct. Iteration planning is a project-related practice, used to plan the work, not to understand what needs to be delivered d) Is correct. Conversation explains how the software will be used and often allows the team to define meaningful acceptance criteria, thus obtaining a shared vision of what should be delivered	FL-4.5.1

When using the 3 C's technique for user story development, what is the work product that is created for the Confirmation aspect?

a. Test Approach
b. Acceptance Criteria
c. Entry Criteria
d. Exit Criteria

B is correct. The confirmation aspect of the 3 C's is the acceptance criteria.

4.5.2 (K2) Classify the different options for writing acceptance criteria

Which of the following BEST describes the way acceptance criteria can be documented?

a) Performing retrospectives to determine the actual needs of the stakeholders regarding a given user story
b) Using the given/when/then format to describe an example test condition related to a given user story
c) Using verbal communication to reduce the risk of misunderstanding the acceptance criteria by others
d) Documenting risks related to a given user story in a test plan to facilitate the risk-based testing of a given user story

Select ONE option.

a) Is not correct. Retrospectives are used to capture lessons learned and to improve the development and testing process, not to document the acceptance criteria b) Is correct. This is the standard way to document acceptance criteria c) Is not correct. Verbal communication does not allow to physically document the acceptance criteria as part of a user story ("card" aspect in the 3C's model) d) Is not correct. Acceptance criteria are related to a user story, not a test plan. Also, acceptance criteria are the conditions that have to be fulfilled to decide if the user story is complete. Risks are not such conditions	FL-4.5.2

Consider the following acceptance criteria for a user story written from the perspective of an online store owner.

```
Given that the user is logged in and on the homepage,
When the user clicks on the "Add Item" button,
Then the "Create Item" form should appear,
And the user should be able to input a name and price for the new
item.
```

In what format is this acceptance criteria written?

a) Rule-oriented
b) Scenario-oriented
c) Product-oriented
d) Process-oriented

Select ONE option.

a) Is not correct. The rule-oriented format includes formats like bullet point verification lists or tabulated forms of input-output mappings, explicitly showing the rules to be followed. Given/When/Then is a scenario-oriented format because it describes a scenario to be verified b) Is correct. This is a Given/When/Then format, which is scenario-oriented c) Is not correct. There is no "product-oriented" format of acceptance criteria d) Is not correct. There is no "process-oriented" format of acceptance criteria	FL-4.5.2

You are writing some acceptance criteria for a story. You have decided to make a list of all the likely inputs to the code and the expected outputs based on those inputs. What format are you using?

a. IPO chart
b. Acceptance-based
c. Rules-oriented
d. Behavior-driven

C is correct per the syllabus. This is a rules-oriented format.

A is not correct because, although you are tracking the inputs and outputs, you are not tracking the processing.

B is not correct because this is not a recognized format.

D is not correct because the given/when/then format is referred to as scenario-oriented.

4.5.3 (K3) Use acceptance test-driven development (ATDD) to derive test cases

Consider the following user story:

As an Editor
I want to review content before it is published
so that I can assure the grammar is correct

and its acceptance criteria:

- The user can log in to the content management system with "Editor" role
- The editor can view existing content pages
- The editor can edit the page content
- The editor can add markup comments
- The editor can save changes
- The editor can reassign to the "content owner" role to make updates

Which of the following is the BEST example of an ATDD test for this user story?

a) Test if the editor can save the document after edit the page content
b) Test if the content owner can log in and make updates to the content
c) Test if the editor can schedule the edited content for publication
d) Test if the editor can reassign to another editor to make updates

Select ONE option.

a) Is correct. This test covers two acceptance criteria: one about editing the document and one about saving changes b) Is not correct. Acceptance criteria cover the editor activities, not the content owner activities c) Is not correct. Scheduling the edited content for publication may be a nice feature, but it is not covered by the acceptance criteria d) Is not correct. Acceptance criteria state about reassigning from an editor to the content owner, not to another editor	FL-4.5.3

You have just started designing test cases for the following user story.

As a customer, I want to be able to filter search results by price range, so that I can find products within my budget more easily. Acceptance criteria: 1. The filter should work for all versions of the application from version 3.0 upwards 2. The filter should allow the customer to set a price range with a minimum and a maximum price 3. The search results should update dynamically as the customer adjusts the price range filter

In all test cases the precondition is as follows: there are only two products available, products A and B. Product A costs $100 and product B costs $110.

Which of the following is the BEST example of a test case for this user story?

a) Enter webpage and set filter to show prices between $90 and $100. Expected result: results show product A only. Set maximum price to $110. Expected result: results now include both products A and B
b) Enter webpage. Expected result: the default minimum and maximum prices are $100 and $110 respectively. Add product C to stock, with price $120. Refresh the client's webpage. Expected result: the default maximum price changes to $120
c) Enter webpage and set filter to show prices between $90 and $115. Expected result: results show both products A and B. Change currency from USD to EUR. Expected result: the filter range changes correctly to EUR values, according to the current exchange rate
d) Enter webpage with three different browsers: Edge, Chrome and Opera. In each browser set filter between $90 and $110. Expected result: results include both products A and B and the results layout is the same in all three browsers

Select ONE option.

a) Is correct. This test case is related to acceptance criteria 2 and 3, because we check if we can set price range (acceptance criterion 2) and if the results update dynamically after adjusting the price range filter (acceptance criterion 3) b) Is not correct. This test case is not related to any of the acceptance criteria. It checks if the filter dynamically sets the default minimum and maximum price range, and not that a customer can do it c) Is not correct. This test case is not related to any of the acceptance criteria. It checks the currency exchange feature, which is not discussed in this user story d) Is not correct. This test case is not related to any of the acceptance criteria. It checks the application's compatibility with different browsers, which is not discussed in this user story	FL-4.5.3

You are using acceptance test-driven development and designing test cases based on the following user story:

As a Regular or Special user, I want to be able to use my electronic floor card, to access specific floors. Acceptance Criteria: AC1: Regular users have access to floors 1 to 3 AC2: Floor 4 is only accessible to Special users AC3: Special users have all the access rights of Regular users

Which test case is the MOST reasonable one to test AC3?

a) Check that a Regular user can access floors 1 and 3
b) Check that a Regular user cannot access floor 4
c) Check that a Special user can access floor 5
d) Check that a Special user can access floors 1, 2 and 3

Select ONE option.

a) Is not correct. We want to check that Special users have the rights of Regular users, so we need to test access rights for a Special user, not for a Regular user b) Is not correct. We want to check that Special users have the rights of Regular users, so we need to test access rights for a Special user, not for a Regular user c) Is not correct. There is no floor 5 described in the acceptance criteria. The test cases should not extend the scope of the user story. But even if we would like to perform negative testing, this test is not directly related to AC3 d) Is correct. This way we can check if a Special user can access floors which are accessible to a Regular user	FL-4.5.3

You are creating test cases for the following story, applying the ATDD approach.

> As a hotel owner
> I want to reserve all the rooms on a floor before moving to the next floor
> So I can maximize the efficiency of the housekeeping staff

You have decided to apply boundary value analysis to this requirement and have identified the following partitions for the occupancy of a floor:

> 0 | 1 - floor full | overbooked

You also want to be sure that the software is usable by the staff and that it performs quickly in determining which floors have availability.

You have designed the following test cases:

1. Test with 1 occupant on the floor
2. Test with the floor full and ensure the next floor is made available for bookings
3. Test with 0 occupants on the floor and ensure that floor is only available when lower floors are fully booked
4. Test the usability to ensure hotel staff will find the software usable
5. Test for response time when the system is at average load and the hotel is 80% occupied

What are you missing?

a. A test with a floor partially occupied but not fully occupied
b. A test for performance
c. A test for trying to assign someone to a floor that is full
d. A test for usability

C is correct. The test that is missing is to try to see if you can allocate someone to a floor that is already fully booked. This is a negative test and should result in an error.

A is covered by testing with one occupant.

B and D are covered.

You are creating test cases for the following story, applying the ATDD approach.

> As a hotel owner
> I want to reserve all the rooms on a floor before moving to the next floor
> So I can maximize the efficiency of the housekeeping staff

You have decided to apply equivalence partitioning to this requirement and have identified the following partitions for the occupancy of a floor:

> 0 | 1 - floor full | overbooked

You also want to be sure that the software is usable by the staff and that it performs quickly in determining which floors have availability.

Given this information, what should be the priority order for the tests you will design?

a. 0, 1-floor full, overbooked, usability, performance
b. Performance, 1-floor full, usability, overbooked, 0
c. Usability, performance, overbooked, 0, 1-floor full
d. Overbooked, 0, 1-floor full, performance, usability

A is correct. This tests the valid scenarios first, then invalid (overbooked), then the non-functional tests.

Conclusion

In this document :

- **We identified the examinable learning objectives.**
- **We presented the probability of questions for each part of chapter 4.**
- **We summarized chapter 4.**
- **We provided section-wise questions and their answers for chapter 4.**

If you need any assistance or have questions, feel free to reach out! 😊

Karim Kalboussi

kalboussikarim3@gmail.com