Question 51: Incorrect

Consider the following activities that might relate to configuration management:

I. Identify and document the characteristics of a test item

II. Control changes to the characteristics of a test item

III. Check a test item for defects introduced by a change

IV. Record and report the status of changes to test items

V. Confirm that changes to a test item fixed a defect

Which of the following statements is true ?

- (Incorrect) **Only I is a configuration management task**
- **All are configuration management tasks**
- **I, II, and III are configuration management tasks**
- (Correct) **I, II, and IV are configuration management tasks**

Explanation

As per the syllabus page 72: *(5.4 Configuration Management)*

The purpose of configuration management is to establish and maintain the integrity of the component or system, the testware, and their relationships to one another through the project and product lifecycle.

To properly support testing, configuration management may involve ensuring the following:

· All test items are uniquely identified, version controlled, tracked for changes, and related to each other

· All items of testware are uniquely identified, version controlled, tracked for changes, related to each other and related to versions of the test item(s) so that traceability can be maintained throughout the test process

· All identified documents and software items are referenced unambiguously in test documentation

Question 52: Incorrect

A test plan included the following clauses among the exit criteria:

• System test shall continue until all significant product risks have been covered to the extent specified in the product risk analysis document.

• System test shall continue until no must-fix defects remain against any significant product risks specified in the product risk analysis document.

During test execution, the test team detects 430 must-fix defects prior to release and all must-fix defects are resolved. After release, the customers find 212 new defects, none of which were detected during testing. This means that only 67% of the important defects were found prior to release, a percentage which is well below average in your industry. You are asked to find the root cause for the high number of field failures. Consider the following list of explanations:

I Not all the tests planned for the significant product risks were executed.

II The organization has unrealistic expectations of the percentage of defects that testing can find.

III A version-control issue has resulted in the release of a version of the software that was used during early testing

IV The product risk analysis failed to identify all the important risks from a customer point of view.

V The product risk analysis was not updated during the project as new information became available

Which of the following statements indicate which explanations are possible root causes?

- (Incorrect) **II, III and IV are possible explanations, but I and V are not possible.**
- **All five are possible explanations**
- (Correct) **I, IV and V are possible explanations, but II and III are not possible**
- **III, IV and V are possible explanations, but I and II are not possible**

Explanation

Answer: I, IV and V are possible explanations, but II and III are not possible.

II is not a possible explanation because the defects percentage 67% is far below the realistic expectations.

III is not a possible explanation because the question specifies that this is a System Test, thus the tests are conducted on the final software system and version issues can rarely occur at this level.

Question 53: Correct

Which of the following is an advantage of independent testing?

- **Independent testers don't have to spend time communicating with the project team**
- **Programmers can stop worrying about the quality of their work and focus on producing more code**
- **The others on a project can pressure the independent testers to accelerate testing at the end of the schedule**
- (Correct) **Independent testers sometimes question the assumptions behind requirements, designs and implementations**

Explanation

As per the syllabus page 63: *(5.1.1 Independent Testing)*

Potential benefits of test independence include:

· Independent testers are likely to recognize different kinds of failures compared to developers because of their different backgrounds, technical perspectives, and biases

· An independent tester can verify, challenge, or disprove assumptions made by stakeholders during specification and implementation of the system

· Independent testers of a vendor can report in an upright and objective manner about the system under test without (political) pressure of the company that hired them

Question 54: Incorrect

Which of the following documents contains sample test plans ?

- (Correct) **ISO/IEC/IEEE 29119-3**
- **ISO 1992**
- **ISE / IEC 2197**
- (Incorrect) **ISO / IEEE 29113**

Explanation

As per the syllabus page 66: *(5.2.1 Purpose and Content of a Test Plan)*

The content of test plans vary, and can extend beyond the topics identified above. A sample test plan structure and a sample test plan can be found in ISO standard (ISO/IEC/IEEE 29119-3).

Question 55: Incorrect

Put the test cases that implement the following test conditions into the best order for the test execution schedule, for a test that is checking modifications of customers on a database.

1 Print modified customer record.

2 Change customer address: house number and street name.

3 Capture and print the on-screen error message.

4 Change customer address: postal code.

5 Confirm existing customer is on the database by opening that record.

6 Close the customer record and close the database.

7 Try to add a new customer with no details at all.

- (Incorrect) **5,4,2,1,3,7,6**
- **4,2,5,1,6,7,3**
- (Correct) **5,4,2,1,7,3,6**
- **1,1,2,3,4,7,6**

Explanation

Our goal is to test modifications of customers on a database, before testing any modification of a customer, we must ensure first that the customer exists in the database, that's why we should begin with [**5 Confirm existing customer is on the database by opening that record**].

The second step should be modifying the customer because this is our main scope, by looking at the choices we notice that the second option can be either [**1**] or [**4**]. [**1**] prints the modified customer record so it is not logical to choose it while [**4**] changes the customer address.

Now we are certain that the first two steps are [**5,4**].

By thinking in a logical way, the last step of the test execution schedule should be [**6 Close the customer record and close the database**].

Now we are certain that the order of test conditions is [**5,4,a,b,c,d,6**] while the values (a,b,c,d) are unknowns.

By looking at the options, there are only two options left which are: [**5,4,2,1,3,7,6**] & [**5,4,2,1,7,3,6**].

The only difference between the two options is the order of test conditions [**3**] & [**7**].

[**3**] captures and prints an error message, so there must be a condition that leads to an error message before it which is [**7**] trying to add a customer with no details at all.

Now we are certain that the order of test conditions is [**5,4,2,1,7,3,6**] while the values (a,b,c,d) are unknowns.

==========

5.2.4 Test Execution Schedule

Once the various test cases and test procedures are produced (with some test procedures potentially automated) and assembled into test suites, the test suites can be arranged in a test execution schedule that defines the order in which they are to be run. The test execution schedule should take into account such factors as prioritization, dependencies, confirmation tests, regression tests, and the most efficient sequence for executing the tests.

Ideally, test cases would be ordered to run based on their priority levels, usually by executing the test cases with the highest priority first. However, this practice may not work if the test cases have dependencies or the features being tested have dependencies. If a test case with a higher priority is dependent on a test case with a lower priority, the lower priority test case must be executed first.

Similarly, if there are dependencies across test cases, they must be ordered appropriately regardless of their relative priorities. Confirmation and regression tests must be prioritized as well, based on the importance of rapid feedback on changes, but here again dependencies may apply.

In some cases, various sequences of tests are possible, with differing levels of efficiency associated with those sequences. In such cases, trade-offs between efficiency of test execution versus adherence to prioritization must be made.

==========

According to Glossary:

"**Test Execution Schedule**": A schedule for the execution of test suites within a test cycle.

"**Test Condition**": An aspect of the test basis that is relevant in order to achieve specific test objectives.

==========

Another explanation:

As per the syllabus page 68: *(5.2.4 Test Execution Schedule)*

Once the various test cases and test procedures are produced (with some test procedures potentially automated) and assembled into test suites, the test suites can be arranged in a test execution schedule that defines the order in which they are to be run. The test execution schedule should take into account such factors as prioritization, dependencies, confirmation tests, regression tests, and the most efficient sequence for executing the tests.

ü Applying this concept we got the most efficient sequence is:

- First, to confirm the existence of an already existing customer (5)
- Second, change the information of the existing customer record (2 & 4)
- Third, check that your modifications are done successfully by viewing the modified customer information (1)
- Fourth, try a false scenario by adding an empty customer record (7)
- Then, check the system behaviour regarding this false scenario (3)
- Finally, and the testing and close the system (6)

*Thus the best order for the test execution schedule will be:

5 => 2 or 4 => 1 => 7 => 3 => 6

Chapter 5 Questions - Results

← Return to Test Menu

Attempt 1

10 questions ✓

Question 1 (correct)

Which of the following is a benefit from the time of recording cost variances (rather than waiting for actuals to become available)?

- ✓ Quality improvement (correct)
- ✗ Facility management
- ✗ Cost accounting
- ✗ Skilled training
- ✗ None of the answers is correct

Explanation

From the syllabus page 75 (Kaizen Management)

The sooner variances are known, the sooner corrective actions can be taken to improve quality. Waiting for actuals to become available means that the variances are known only after the fact, which is a disadvantage of actual costing. Kaizen, which is a management method, emphasizes a continuous improvement in quality, efficiency, and productivity. Kaizen is a Japanese word that means "change for the better" or "improvement." It is a philosophy that focuses on continuous improvement throughout life.

Question 2 (correct)

Which of the following is a cost that is not traceable to a product?

- ✓ Write-down of obsolete inventory (correct)
- ✗ Sales commission (correct)
- ✗ Shipping and handling (correct)
- ✗ Production labor (correct)
- ✗ Automobile tires (correct)

Explanation

From the syllabus page 28 (Cost Basics, Cost Flow Diagram and Terms)

Cost Flow Diagram and Terms

Overhead is a cost which cannot be traced to the product.

Sales commission, shipping and handling, the write-down of obsolete inventory, and production labor are all examples of overhead costs.

Question 3 (correct)

From the syllabus page 75 (Kaizen Management)

Kaizen is a philosophy that focuses on continuous improvement throughout life. Kaizen is a Japanese word that means "change for the better" or "improvement." It is a philosophy that focuses on continuous improvement throughout life.

Kaizen emphasizes a continuous improvement in quality, efficiency, and productivity.

Kaizen is a management method.

Kaizen is a Japanese word that means "change for the better" or "improvement."

Kaizen

Question 61 (Correct)

Which one of the following is a unit level type of code coverage for a particular software product your team is testing, what type of testing should you use?

- Statement Coverage
- **Structure based** (Correct)
- Function Coverage
- Experience based
- Decision Coverage

Explanation

Structure based is a unit level type of code coverage for a particular software product your team is testing, what type of testing should you use?

Structure based is also known as **Structure or Behavior based testing** and is based on an

analysis of the appropriate test basis (e.g., formal requirements documents, specifications of business rules, use cases, or business processes).

These techniques are applicable to both functional and nonfunctional testing. Black box test techniques concentrate on the inputs and outputs of the test object without reference to its internal structure.

White box test techniques (also called **structural or structure based techniques**) are based on an analysis of the architecture, detailed design, internal structure, or the code of the test object. Unlike black box

techniques,

white box test techniques concentrate on the structure and processing within the test object.

Experience based test techniques leverage the experience of developers, testers and users to design,

implement, and execute tests. These techniques are often combined with black box and white box

techniques.

ISTQB Glossary

Version 2018 (dev.1)

Defect based test techniques (**** Black box techniques in which test cases are developed from what is known about specific defects.

Based on your understanding of the mentioned section you will find out that the technique that deals with the case is known as **Structure based "White box"**

Question 62 (Incorrect)

How is statement coverage determined?

- **Number of decision points divided by the number of test cases**
- **Number of decision outcomes tested divided by the total number of decision outcomes** (Incorrect)
- **Number of possible test case outcomes tested by the total number of possible outcomes**
- **Number of statements statements tested divided by the total number of executable statements** (Correct)

Explanation

Question 62 (Statement Testing and Coverage)

"Coverage is measured as the number of **statements executed** by the tests divided by the total number of **executable statements** in the test object, **normally expressed as a percentage.**"

Question 63 (Correct)

Which of the following is the correct decision table for the following pseudocode: Select loan Number

Note: If you select or delete loans from the loans bargain, you no longer get the loans bargain.

```
Start,
Select loan bargain
If customer adds items
        while items to be added
                Ask customer which item
                Add item
        End while
End if
If customer deletes items
        while items to be deleted
                Ask customers which item
                Delete item
        End while
End if
If customer wants loans
        Add fees to order
End if
Complete order
End
```

Table 1					
Select loan bargain	T	T	T	T	T
Customer adds items	T	T	T	T	T
Add to Order	N	N	N	N	N
Customer deletes items	T	T	T	T	T
Delete item	N	N	N	N	N
Wants loans	T	T	T	T	T
Add fees	N	N	N	N	N
Complete order	N	N	N	N	N

Nezam Table (1)

Table 2					
Select loan bargain	T	T	T	T	T
Customer adds items	T	T	T	T	T
Add to Order	N	N	N	N	N
Customer deletes items	T	T	T	T	T
Delete item	N	N	N	N	N
Wants loans	T	T	T	T	T
Add fees	N	N	N	N	N
Complete order	N	N	N	N	N

Nezam Table (2)

Table 3					
Select loan bargain	T	T	T	T	T
Customer adds items	T	T	T	T	T
Add to Order	N	N	N	N	N
Customer deletes items	T	T	T	T	T
Delete item	N	N	N	N	N
Wants loans	T	T	T	T	T
Add fees	N	N	N	N	N
Complete order	N	N	N	N	N

Nezam Table (3)

Table 4					
Select loan bargain	T	T	T	T	T
Customer adds items	T	T	T	T	T
Add to Order	N	N	N	N	N
Customer deletes items	T	T	T	T	T
Delete item	N	N	N	N	N
Wants loans	T	T	T	T	T
Add fees	N	N	N	N	N
Complete order	N	N	N	N	N

Nezam Table (4)

- Table 1 (Correct)
- Table 2
- Table 3
- Table 4

Explanation

Explanation

Question 63 (Decision Testing)

Table 1 is the correct decision table for the following pseudocode: Select loan Number

Where we will find out that we have 5 Partitions (Child, Teenager, Adult)

The minimum set of test cases needed to cover all valid equivalence partitions are a case for each valid partition (Child, Teenager, Adult).

Note that: Teenagers are in the age between 13 & 19.

Question 65 (Correct)

A simple gaming application has been specified as an unit of use cases. It has been implemented in a programming language. The code has been reviewed and the code was found to obtain the attributes into the variable but no possible. Which of the following techniques would be the best choice to use in testing this application?

- **State transitions testing & use case testing**
- **Equivalence Partitioning & State transition testing**
- **Use case testing & exploratory testing** (Correct)

Explanation

Explanation

Question 65 (Categories of Test Techniques and Their Characteristics)

Understanding of the syllabus and the question)

Which one is always a best choice to testing the unit of details or documentation of your test process, which means that you will probably work with the experience based test techniques, and you will find out that the best choice is to use Use case testing and Exploratory testing.

So from the question we have two points to remember:

It's a UIT "use a programme test" which means that the application is already in an existence and implemented. Third that we have a simple application, so it's not a complicated one.

The need to release the application fast to the market means that we need to choose techniques that are quick and cost effective, so we need to choose techniques that are experience based that includes all best techniques of all best practice for it.

And by analyzing the answers:

"State transition testing & use case testing" is not a good choice , use case testing is a good choice, but state transition testing is not a good choice, because it's a complicated one and it's not the quickest one and it's not cost effective and it's not the best one and it's not the most suitable one for the question.

"Equivalence partitioning & State transition testing" is not a good choice also, same as the above reason, Equivalence partitioning also needs a lot of time to analyze, and it's not the quickest one.

"Use case testing & exploratory testing" is the best choice , exploratory testing is an experience based test technique which does not require a lot of documentation and time to be prepared, and it's the quickest one, and it's cost effective, also testing is a good choice because it's the quickest one and it's cost effective.

"Decision table testing & exploratory testing" is not a good choice also, exploratory testing is a good choice, but decision table testing is not a good choice, because it's a complicated one and it's not the quickest one and it's not cost effective, also decision table testing, system, but again decision table testing needs a lot of time to analyze , document and execute.

Question 66 (Correct)

Consider the code below is being performed on a operating this system. Test cases are created based on the code given below. Test case 1 and 2 are shown below.

	Rules	R1	R2
Conditions	Speed > 20	T	F
	School Zone	F	F
Actions	Speed > 20	T	F
	School Zone	F	F

Given the following additional test cases:

	Rules	DT1	DT2	DT3	DT4
Input	Speed > 20	T	T	F	F
	School Zone	F	T	F	T
Expected Result	STOP Fine	P	P	P	P

Which two of the additional test cases could achieve full coverage of the complete decision table (and combinations) if the test cases had been run already, been generated for rules 1 and 2?

- **DT1 & DT2**
- **DT3 & DT4**
- **DT1 & DT4** (Correct)
- **DT2 & DT3**

Explanation

To achieve full coverage of the whole system, we will need 4 test cases with the following values:

1 Speed more than 20 & school zone

2 Speed more than 20 & not a school zone

3 Speed not more than 20 & school zone

4 Speed not more than 20 & not a school zone

Test case 1 and 2 are shown in the question, and we have one of the 4 cases which are 2 & 4, so we need to add two more test cases to have the following cases to cover the whole system:

3 Speed more than 20 & not a school zone

4 Speed not more than 20 & a school zone

The two last cases that will achieve these two inputs are **DT1 & DT4**

DT1 Speed > 20 & school zone and **DT4 Speed < 20 & not a school zone**

Decision Table Testing

Contractual test techniques are useful for testing the implementation of system requirements that have a logical relationship between conditions which affect a defined set of outcomes. This is known as decision table testing.

Decision tables are a good way to record complex business rules that a system must implement. The information in a decision table can be used to derive test cases. Decision tables are also useful for defining complex business rules to be implemented in a system, specifying complex logic in a program, or for specifying complex business rules in a database.

The common minimum coverage standard for decision table testing is to have at least one test case for each decision rule in the table. This typically means that all combinations of conditions. Coverage is measured as the number of decision rules tested by at least one test, divided by the total number of decision rules, normally expressed as a percentage.

The strength of decision table testing is that it helps to identify all the important combinations of conditions, some of which might otherwise be overlooked. It also helps in finding any gaps in requirements. It may be worthwhile to use this technique to check the logic of the software designers on a combination of conditions, as you test their software.

According to Wikipedia:

"**Decision Table**" is a table used to show sets of conditions and the actions resulting from them.

Decision Testing (Branch Testing) Black box test techniques in which test cases are designed to exercise the combinations of inputs and/or stimuli (causes) shown in a decision table.

Question 67 (Incorrect)

You have designed test cases to provide 100 % statement and 100 % decision coverage for the following pseudocode:

```
If grade1 > grade2
    print("Highest grade is grade 1")
else
    print("Highest grade is grade 2")
```

The following statements are given:

Statement 1: grade1=5, grade2=5

Statement 2: grade1=5, grade2=3

Print("The total of your grades is", grade1 + grade2)

After the test statements are run, how many more test cases are required to achieve 100 % statement coverage? 0 1 2 3 (Incorrect)

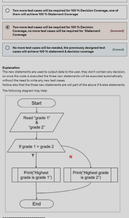
- **One test case will be required for 100 % Decision Coverage**
- **Two more test cases will be required for 100 % Decision Coverage, one of them will achieve 100 % Statement Coverage** (Correct)
- **Two more test cases will be needed, the previously designed test cases will achieve 100 % statement coverage** (Incorrect)
- **Three more test cases will be needed to achieve 100 % Statement Coverage**

Explanation

Test statements are used to make sure that the code, they don't contain any decision so once the code is executed the three new statements will be executed automatically, achieving 100 % statement coverage.

Now that we have the three statements we will apply the above 4 data statements.

The following diagram may help:



4.3.1 Statement Testing and Coverage

Statement testing exercises the executable statements in the code. Coverage is measured as the number of statements executed by the tests divided by the total number of executable statements in the test object, normally expressed as a percentage.

4.3.2 Decision Testing and Coverage

Decision testing exercises the decisions in the code and tests the code that is executed based on the decision outcomes. To do this, the tester must follow the points flows that result from each decision in the code. Each flow may be taken as a separate test or may be part of a larger test. The decisions in the code may be in the form of an IF statement, a CASE statement, or looping statements.

Coverage is measured as the number of decision outcomes executed by the tests divided by the total number of decision outcomes in the test object, normally expressed as a percentage.

4.3.3 The Value of Statement and Decision Testing

Given a set of tests that achieve 100% statement coverage, it means that all executable statements in the code have been tested at least once, but it does not ensure that all decision logic has been tested. In the worst case, it may be possible to achieve 100% statement coverage by testing only one of the possible branches in a program.

Within 100% decision coverage is achieved, if exercised all decision outcomes, which means that all branches in the code have been tested, this will usually find defects in the program. However, it may not find all of them. In the case of an IF statement, for example, the Code Statement coverage tests to find defects in code that may not exist based by other tests, but it may not find defects in the code that are caused by the combination of two or more decisions and flows outcomes.

Achieving 100% decision coverage guarantees 100% statement coverage but not vice versa.

According to Glossary:

"**Statement**" A type of statement in which a choice between two or more possible outcomes controls which set of statements will execute.

"**Decision Statement**" The result of a decision that determines the next statement to be executed.

"**Executable Statement**" A statement which, when compiled, is translated into object code, which will be executed by the processor in response to control passing to it during program execution.

"**Branch**" A conditional transfer of control from one statement to another.

"**Statement Testing**" A white box test technique in which test cases are designed to execute statements.

"**Statement Coverage**" The percentage of executable statements that have been exercised by a test suite.

"**Statement Testing**" A white box test technique in which test cases are designed to execute statements.

Question 21 Incorrect

How many test cases are necessary to cover all the possible sequences of statements (paths) for the following program fragment?

```
Print sum (int a, int b) {
    int result = a + b;
    if (result > 0)
        print("red", result)
    else if (result < 0)
        print ("blue", result)
    }
```

- ○ 1
- ○ 4
- ⊙ 3 (Incorrect)
- ○ 2 (Correct)

Explanation
Explanation
| Section 4.3 | Statement Testing and Coverage | |

Based on the above image:

To make a 100% **Statement Coverage** you need to trigger the two print statements (Print("red",result) - Print("blue",result))

For Statement Coverage:
To trigger covers (Print("red",result)) you need a test case that made (result > 0).
To trigger covers (Print("blue",result)) you need a test case that made (result < 0).

And since the two test cases can't be done as one test case , **the minimum test cases required to cover all the possible sequences of statements are 2 test cases.**

Question 22 Incorrect

Statement Coverage will not check for the following ?

- ○ Unused branches
- ○ Dead Code
- ⊙ Unused statement (Incorrect)
- ○ Missing Statements (Correct)

Explanation
Explanation
Statement Coverage will check for any executable statement that is written in the code and this includes unused branches, dead code, & unused statements. On the other hand, missing statements are not part of the code so statement coverage is not able to check for them.

Question 23 Correct

Minimum Tests required for statement coverage and branch coverage

```
Read P
Read Q
If p+q > 100 then
Print "Large"
End if
If p > 50 then
Print "Plarge"
End If
```

- ○ Statement coverage is 2, Branch coverage is 2
- ○ Statement coverage is 1, Branch coverage is 2
- ⊙ Statement coverage is 1, Branch coverage is 3 (Correct)
- ○ Statement coverage is 4, Branch coverage is 2

Explanation
Explanation
| Section 4.3 | Statement Testing and Coverage , 4.3.2 Decision Testing and Coverage | |

Based on the above image:

To make 100% **Statement Coverage** you need to trigger the two print statements ("Print"Large" - Print "Plarge")

To make 100% **Decision Coverage** you need to pass through all the decision outcomes **(The False of the Two if Conditions)**

For Statement Coverage:
Since you can trigger both print statements with a single execution, the minimum test cases required to cover all the possible sequences of statements are 1 test case.

For Decision Coverage:
since the above test case will trigger the "Y" branches, you will need to trigger the "N" branches by another test case, so in total you need 2 test cases.

So the minimum test cases required to cover all the possible sequences of statements are 1 test case, but the minimum test cases required to cover all the possible branches are 2 test cases.

==========
For Statement Coverage:
We need to cover two statements
"Print"Large" - "Print "Plarge"
==========
For Decision Coverage:
We need to cover 4 decision outcomes:
1-If p + q is greater than 100
2-If p + q is not greater than 100
3-If p is greater than 50
4-If P is not greater than 50
So we need to cover decision outcomes with only two test cases:
1-If p=50,Q=51
2-If P=20,Q=10
==========
Note: **Branch coverage is the same as decision coverage.**

Question 24 Correct

Minimum Tests required for statement coverage and branch coverage

```
Switch PC on
Start "outlook"
If outlook appears Then
Send an email
Close outlook
```

- ○ Statement Coverage is 1, Branch Coverage is 1
- ○ Statement Coverage is 1, Branch Coverage is 2
- ⊙ Statement Coverage is 2, Branch Coverage is 2 (Correct)
- ○ Statement Coverage is 2, Branch Coverage is 3

Explanation
Explanation
| Section 4.3 | Statement Testing and Coverage | |

Based on the above image:

To make a 100% **Statement Coverage** you need to trigger the statement (Send an email

To make a 100% **Decision Coverage** you need to pass through all the decision outcomes **(The False of The if Condition)**

For Statement Coverage:
Since you can trigger both print statements with a single execution, the minimum test cases required to cover all the possible sequences of statements are 1 test case.

For Decision Coverage:
since the above test case will trigger the "Y" branches, you will need to trigger the "N" branches by another test case, so in total you need 2 test cases.

So the minimum test cases required to cover all the possible sequences of statements are 1 test case, but the minimum test cases required to cover all the possible branches are 2 test cases.

Question 25 Incorrect

Which of the following is correct ?

- ○ 100% statement coverage guarantees 100% branch coverage.
- ○ 100% statement coverage guarantees 100% decision coverage
- ○ 100% branch coverage guarantees 100% statement coverage. (Correct)
- ⊙ 100% branch coverage guarantees 100% path coverage. (Incorrect)

Explanation
Explanation
| Section 4.3 | The Value of Statement and Decision Testing | |

"achieving 100% decision coverage guarantees 100% statement coverage"
Note that Branch coverage is the same as decision coverage.

Question 26 Incorrect

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

- ○ You can develop tests that might be missed by more formal techniques (Correct)
- ○ You can test for defects that only experienced users would encounter
- ⊙ You can target the developer's efforts to the areas that users will be more likely to use (Incorrect)
- ○ It is supported by strong tools and can be automated

Explanation
Explanation
| Section 4.4 | Experience-based Test Techniques | |

"These techniques can be **helpful** in identifying tests that were not easily identified by other more systematic techniques. Depending on the tester's approach and experience, these techniques may achieve widely varying degrees of coverage and effectiveness."

Question 27 Correct

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

- ○ Behavior-based
- ○ Structure-based
- ⊙ Experience-based (Correct)
- ○ Reference-based

Explanation
Explanation
| Section 4.4 | Experience-based Test Techniques | |

Experience-based Test Techniques are:
-Error Guessing
-Exploratory Testing
-Checklist-based Testing

Question 28 Incorrect

If you are using a testing technique to identify test cases that were missed when you applied formal testing techniques, what type of test design are you doing?

- ⊙ Experience-based (Correct)
- ○ Informal
- ⊙ Defect-based (Incorrect)
- ○ Ad hoc

Explanation
Explanation
| Section 4.4 | Experience-based Test Techniques | |

"These techniques can be **helpful** in identifying tests that were not easily identified by other more systematic techniques. Depending on the tester's approach and experience, these techniques may achieve widely varying degrees of coverage and effectiveness."

Question 29 Incorrect

Which of the following would be the best test approach when there are poor specifications and time pressure?

- ○ Use case testing
- ⊙ Condition coverage (Incorrect)
- ○ Exploratory testing (Correct)
- ○ Path testing

Explanation
Explanation
| Section 4.4.3 | Exploratory Testing | |

"Exploratory testing is most **useful when there are few or inadequate specifications or significant time pressure** on testing. Exploratory testing is also useful to complement other more formal testing techniques."

Question 40. Correct

For the following state transition diagram

What is the number of test cases required to test all the transitions in S ?

Consider that each test case starts at the "Start" State and ends at a dead state

Start
and Internet
WAP-PC
WAP-Mobile
Wrong PIN
Wrong PIN
Wrong PIN

Question 21 (**Incorrect**)

Which of the following statements about the group '1899,1900,1901,2003,2004,2005' could be right if we know also that the **three point value which is something you could see it in the ISTQB - Advanced Test Analyst Syllabus, not in the ISTQB - Foundation Level Syllabus).**

- 1899,1900,1901,2003,2004,2005 (**Incorrect**)
- 1899,1900,1901,2003,2004,2005 (**Correct**)
- 1900,1901,2003,2004,2005
- 1900,2003,2004

Explanation
Explanation
Section 4.2.2 Boundary Value Analysis ( )

From the example we can draw the following "Number Line"

Where we will find out that we have 3 Partitions (Invalid, Valid, Invalid). And that we have 2 **Valid** Boundaries (1900 (2004) and 2 **Invalid** Boundary (1899 (2005))

As the question wants **boundaries values**:

Then we should pick one value from each side of the group ('**1899,1900,2004,2005**') is the only group that satisfies the requirement.

Note: the group '**1899,1900,1901,2003,2004,2005**' could be right if we know also that the **three point value** which is something you could see it in the ISTQB - Advanced Test Analyst Syllabus, not in the ISTQB - Foundation Level Syllabus).

Question 22 (**Incorrect**)

What is the expected result for each of the following test cases?
A. Citibank card member, holding a silver room
B. Avis Citibank member, holding a platinum room

	Rule 1	Rule 2	Rule 3	Rule 4
Conditions				
Citibank Member	**Yes**	**Yes**	**No**	**No**
Type of Room	**Silver**	**Platinum**	**Silver**	**Platinum**
Actions				
Offer upgrade to gold	**Yes**	**No**	**No**	**No**
Offer upgrade to blue	**N/A**	**Yes**	**N/A**	**No**

- A – don't offer any upgrade, B – Don't offer any upgrade
- A – Don't offer any upgrade, B – Offer upgrade to gold (**Incorrect**)
- A – Offer upgrade to silver, B – Offer upgrade to silver
- A – Offer upgrade to gold, B – Don't offer any upgrade (**Correct**)

Explanation
Explanation
Section 4.3.2 Decision Table Testing ( )

By analyzing the two given test cases and the table you will find that
Test case B that the Rule number 2 which means that the expected result from this test case is to **Offer upgrade to gold**.

Test case B: **Rule 2** Rule number 2 which means that the expected result from this test case is to **Don't upgrade**.

Based on this, you will find out that the only relevant answer is **A-Offer upgrade to gold ,B-Don't offer any upgrade** .

Question 23 (**Incorrect**)

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

A.70 5 Fred is a 32 year old smoker resident in London
B.70,0 Jean-Michel is a 60 year old non-smoker resident in Paris

	Rule 1	Rule 2	Rule 3	Rule 4
Conditions				
UK Resident ?	**False**	**True**	**True**	**True**
Age (18-50) ?	**Don't Care**	**False**	**True**	**True**
Smoker ?	**Don't Care**	**Don't Care**	**False**	**True**
Actions				
Insure Client ?	**False**	**False**	**False**	**False**
Offer 35 % Discount ?	**False**	**False**	**False**	**True**

- A – Insure, 10% discount, B – Insure, no discount.
- A – Don't Insure, B – Don't Insure (**Incorrect**)
- A – Insure, no discount, B – Insure with 10% discount
- A – Insure, no discount, B – Don't Insure (**Correct**)

Explanation
Explanation
Section 4.3.2 Decision Table Testing ( )

By analyzing the two given test cases and the table you will find that
Test case A that the Rule number 1 which means that the expected result from this test case is **No Insure, but no discount**.

Test case B: **Rule 4** Rule number 4 which means that the expected result from this test case is **Don't Insure , and no discount also**.

Based on this, you will find out that the only relevant answer is **A-Insure , no discount ,B-Don't Insure** .

Question 24 (**Incorrect**)

-What is the expected result for each of the following test cases?
-1 frequent flyer member, travelling in Business class
-2 Non-member, travelling in Economy class

	Rule 1	Rule 2	Rule 3	Rule 4
Conditions				
Frequent Flyer Member	**Yes**	**Yes**	**No**	**No**
Class	**Business**	**Economy**	**Business**	**Economy**
Actions				
Offer upgrade to First	**No**	**No**	**No**	**No**
Offer upgrade to Business	**N/A**	**Yes**	**N/A**	**No**

- A – don't offer any upgrade, B – Offer upgrade to business (**Incorrect**)
- A – Offer upgrade to First, B – Don't offer any upgrade (**Correct**)
- A – Offer upgrade to First, B – Offer upgrade to business

Explanation
Explanation
Section 4.3.2 Decision Table Testing ( )

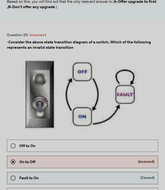
By analyzing the two given test cases and the table you will find that
Test case B that the Rule number 2 which means that the expected result from this test case is to **Offer upgrade to First**.

Test case: **B** test the Rule number 4 which means that the expected result from this test case is **Don't Offer upgrade to First**.

Based on this, you will find out that the only relevant answer is **A-Offer upgrade to First ,B-Don't offer upgrade** .

Question 25 (**Incorrect**)

-Consider the above state transition diagram of a switch, Which of the following represents an invalid state transition



- Off to On
- Go to Off (**Incorrect**)
- Fault to On (**Correct**)
- Fault to Fault

Explanation
Explanation
Section 4.3.4 State Transition Testing ( )

An Invalid Transition , is any transition that is not drawn in the state transition diagram, Like

Going for the **FAULT** state to **OFF**

Going for the **FAULT** state to **OFF**

Based on this, you will find out that the only relevant answer is **Fault to On**.

Question 26 (**Correct**)

Which is not true The black box tester

- **should be able to understand a functional specification or requirements document**
- **should be able to understand the source code.** (**Correct**)
- **is highly motivated to find faults**
- **is creative to find the system's weaknesses**

Explanation
Explanation
Section 5.2.2 Categories of Test Techniques and Their Characteristics( )

By analyzing the answers & based on your understanding of the above mentioned paragraph.

-**Should be able to understand a functional specification or requirements document:** false because Common characteristics of black-box test techniques include:Test conditions, test cases, and test data are derived from a test basis that may include requirements, specifications, use cases, and user stories.

-**is highly motivated to find faults:** true, because based on your understanding of the ISTQB Foundation level syllabus, especially this section (5.5.2 Tester's and Developer's Mindsets),you understand that it is a common characteristic of a good tester.

-**is creative to find the system's weaknesses:** true, because it is also a common characteristic a good tester, same as above one.

-**Should be able to understand the source code :** false, because It's a Common characteristic of white-box tester not a black-box tester. Common characteristics of white-box test techniques include the following: 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.

Question 27 (**Incorrect**)

Use technique of Black-Box testing is Equivalence Partitioning. In a program statement that accepts only positive integers from range 10 possible choices, numbered 1through 10, the middle partition could be from ___ to ____

- 4 to 6
- 0 to 10 (**Incorrect**)
- 1 to 10 (**Correct**)
- No answer is correct

Explanation
Explanation
Section 4.2.1 Equivalence Partitioning ( )

In this question we will have three partitions
1-Invalid partition **(Less than 1)**
2-Valid partition **(1 to 10)**
3-Invalid partition **(More than 10)**

Partition (2) is the Valid partition, partition (3) is the Iad partition, **3-partition (2) is the middle partition because it has a partition before it and another partition after it.**

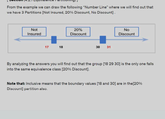
Question 28 (**Incorrect**)

Find the Equivalence class for the following test case. Enter a number to test the validity of being accepting the numbers between 1 and 99

- **All numbers < 1**
- **All numbers > 99** (**Incorrect**)
- **Number = 0**
- **All numbers between 1 and 99** (**Correct**)

Explanation
Explanation
Section 4.2.1 Equivalence Partitioning ( )

From the example we can draw the following "Number Line", where we will find out that we have 3 Partitions (Rejected, Accepted, Rejected).And that the only valid partition is the partitions within the boundaries (1 , 99), which is the (Accepted) partition.



As the question want you to choose the range of the numbers (Equivalent Class Partition) that represent the valid numbers of the (Accepted) partition in the above "Number Line".

Based on this, you will find out that the only relevant answer is **All numbers between 1 and 99**.

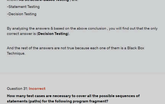
Question 29 (**Incorrect**)

Given the following specification, which of the following values for age are in the same equivalence partition? If you are less than 18, you are too young to be insured. Between 18 and 30 inclusive, you will receive a 20% discount. Anyone over 30 is not eligible for a discount

- **17,18,19** (**Incorrect**)
- **26,30,31**
- **18,26,30** (**Correct**)
- **17,26,31**

Explanation
Explanation
Section 4.2.1 Equivalence Partitioning ( )

From the example we can draw the following "Number Line" where we will find out that we have 3 Partitions (Not Insured, 20% Discount, No Discount).



By analyzing the answers you will find out that the group (18-29 30) is the only one falls in the same equivalence class (20% Discount).

Note that Inclusive means that the boundary values (18 and 30) are in the(20% Discount) partition also.

Question 30 (**Incorrect**)

Which one of the following techniques is structure-based ?

- **Decision Testing** (**Correct**)
- **Boundary Value Analysis** (**Incorrect**)
- **Equivalence Partitioning**
- **State Transition Testing**

Explanation
Explanation
Section 3 (White-box Test Techniques)

Based on the ISTQB Foundation level syllabus, the White-box Test Techniques (Also known as **Structure-Based Testing**) are:

-Statement Testing
-Decision Testing

By analyzing the answers & based on the above conclusion , you will find out that the only correct answer is **Decision Testing**.

And the rest of the answers are not true because each one of them is a Black Box Technique.

Question 31 (**Incorrect**)

How many test cases are necessary to cover all the possible sequences of statements (paths) for the following program fragment?