TestpassportQ&A

Higher Quality Better Service!

We offer free update service for one year

Http://www.testpassport.com

Exam : **CTFL_Syll_4.0**

Title : ISTQB Certified Tester Foundation Level (CTFL)

Version : DEMO

1.The tests at the bottom layer of the test pyramid:
A. run faster than the tests at the top layer of the pyramid
B. cover larger pieces of functionalities than the tests at the top layer of the pyramid
C. are defined as 'UI Tests' or 'End-To-End tests' in the different models of the pyramid
D. are unscripted tests produced by experience-based test techniques
Answer: A
Explanation:
The tests at the bottom layer of the test pyramid run faster than the tests at the top layer of the pyramid because they are more focused, isolated, and atomic. They usually test individual units or components of the software system, such as classes, methods, or functions. They are also easier to maintain and execute, as they have fewer dependencies and interactions with other parts of the system. The tests at the top layer of the test pyramid, on the other hand, are slower because they cover larger pieces of functionalities, such as user interfaces, workflows, or end-to-end scenarios. They also have more dependencies and interactions with other systems, such as databases, networks, or external services. They are more complex and costly to maintain and execute, as they require more setup and teardown procedures, test data, and test environments.
Reference: ISTQB Certified Tester Foundation Level (CTFL) v4.0 sources and documents:
ISTQB® Certified Tester Foundation Level Syllabus v4.0, Chapter 3.2.1, Test Pyramid1 ISTQB® Glossary of Testing Terms v4.0, Test Pyramid2

2.Test automation allows you to:
A. demonstrate the absence of defects
B. produce tests that are less subject to human errors
C. avoid performing exploratory testing
D. increase test process efficiency by facilitating management of defects
Answer: B
Explanation:
Test automation allows you to produce tests that are less subject to human errors, as they can execute predefined test scripts or test cases with consistent inputs, outputs, and expected results. Test automation can also reduce the manual effort and time required to execute repetitive or tedious tests, such as regression tests, performance tests, or data-driven tests. Test automation does not demonstrate the absence of defects, as it can only verify the expected behavior of the system under test, not the unexpected or unknown behavior. Test automation does not avoid performing exploratory testing, as exploratory testing is a valuable technique to discover new information, risks, or defects that are not covered by automated tests. Test automation does not increase test process efficiency by facilitating management of defects, as defect management is a separate activity that involves reporting, tracking, analyzing,and resolving defects, which may or may not be related to automated tests.
Reference: ISTQB Certified Tester Foundation Level (CTFL) v4.0 sources and documents:
ISTQB® Certified Tester Foundation Level Syllabus v4.0, Chapter 3.3.1, Test Automation1 ISTQB® Glossary of Testing Terms v4.0, Test Automation2

3.Which of the following statements about how different types of test tools support testers is true?
A. The support offered by a test data preparation tool is often leveraged by testers to run automated regression test suites

B. The support offered by a performance testing tool is often leveraged by testers to run load tests
C. The support offered by a bug prediction tool is often used by testers to track the bugs they found
D. The support offered by a continuous integration tool is often leveraged by testers to automatically generate test cases from a model
Answer: B
Explanation:
The support offered by a performance testing tool is often leveraged by testers to run load tests, which are tests that simulate a large number of concurrent users or transactions on the system under test, in order to measure its performance, reliability, and scalability. Performance testing tools can help testers to generate realistic workloads, monitor system behavior, collect and analyze performance metrics, and identify performance bottlenecks.
The other statements are false, because:
A test data preparation tool is a tool that helps testers to create, manage, and manipulate test data, which are the inputs and outputs of test cases. Test data preparation tools are not directly related to running automated regression test suites, which are test suites that verify that the system still works as expected after changes or modifications. Regression test suites are usually executed by test execution tools, which are tools that can automatically run test cases and compare actual results with expected results.
A bug prediction tool is a tool that uses machine learning or statistical techniques to predict the likelihood of defects in a software system, based on various factors such as code complexity, code churn, code coverage, code smells, etc. Bug prediction tools are not used by testers to track the bugs they found, which are the actual defects that have been detected and reported during testing. Bugs are usually tracked by defect management tools, which are tools that help testers to record, monitor, analyze, and resolve defects.
A continuous integration tool is a tool that enables the integration of code changes from multiple developers into a shared repository, and the execution of automated builds and tests, in order to ensure the quality and consistency of the software system. Continuous integration tools are not used by testers to automatically generate test cases from a model, which are test cases that are derived from a representation of the system under test, such as a state diagram, a decision table, a use case, etc. Test cases can be automatically generated by test design tools, which are tools that support the implementation and maintenance of test cases, based on test design specifications or test models.
Reference: ISTQB Certified Tester Foundation Level (CTFL) v4.0 sources and documents:
ISTQB® Certified Tester Foundation Level Syllabus v4.0, Chapter 3.4.1, Types of Test Tools
ISTQB® Glossary of Testing Terms v4.0, Performance Testing Tool, Test Data Preparation Tool, Bug Prediction Tool, Continuous Integration Tool, Test Execution Tool, Defect Management Tool, Test Design Tool

4.Which of the following statements about branch coverage is true?
A. The minimum number of test cases needed to achieve full branch coverage, is usually lower than that needed to achieve full statement coverage
B. If full branch coverage has been achieved, then all unconditional branches within the code have surely been exercised
C. If full branch coverage has been achieved, then all combinations of conditions in a decision table have surely been exercised

D. Exercising at least one of the decision outcomes for all decisions within the code, ensures achieving full branch coverage

Answer: D

Explanation:

Exercising at least one of the decision outcomes for all decisions within the code, ensures achieving full branch coverage, which is a test coverage criterion that requires that all branches in the control flow of the code are executed at least once by the testcases. A branch is a basic block of code that has a single entry point and a single exit point, and a decision is a point in the code where the control flow can take more than one direction, such as an if-then-else statement, a switch-case statement, a loop statement, etc. The decision outcomes are the possible paths that can be taken from a decision, such as the then branch or the else branch, the case branch or the default branch, the loop body or the loop exit, etc.

The other statements are false, because:

The minimum number of test cases needed to achieve full branch coverage, is usually higher than that needed to achieve full statement coverage, which is a test coverage criterion that requires that all executable statements in the code are executed at least once by the test cases. This is because branch coverage is a stronger criterion than statement coverage, as it implies statement coverage, but not vice versa.

For example, a single test case can achieve full statement coverage for an if-then-else statement, but two test cases are needed to achieve full branch coverage, as both the then branch and the else branch need to be exercised.

If full branch coverage has been achieved, then all unconditional branches within the code have not necessarily been exercised, as unconditional branches are branches that do not depend on any decision, and are always executed, such as a goto statement, a break statement, a return statement, etc. Unconditional branches are not part of the branch coverage criterion, as they do not represent different paths in the control flow of the code. However, they are part of the statement coverage criterion, as they are executable statements in the code.

If full branch coverage has been achieved, then all combinations of conditions in a decision table have not necessarily been exercised, as a decision table is a test design technique that represents the logical relationships between multiple conditions and their corresponding actions, in a tabular format. A decision table can have more combinations of conditions than the number of decision outcomes in the code, as each condition can have two or more possible values, such as true or false, yes or no, etc.

For example, a decision table with four conditions can have 16 combinations of conditions, but the corresponding code may have only two decision outcomes, such as pass or fail. To exercise all combinations of conditions in a decision table, a stronger test coverage criterion is needed, such as condition combination coverage, which requires that all possible combinations of condition outcomes in the code are executed at least once by the test cases.

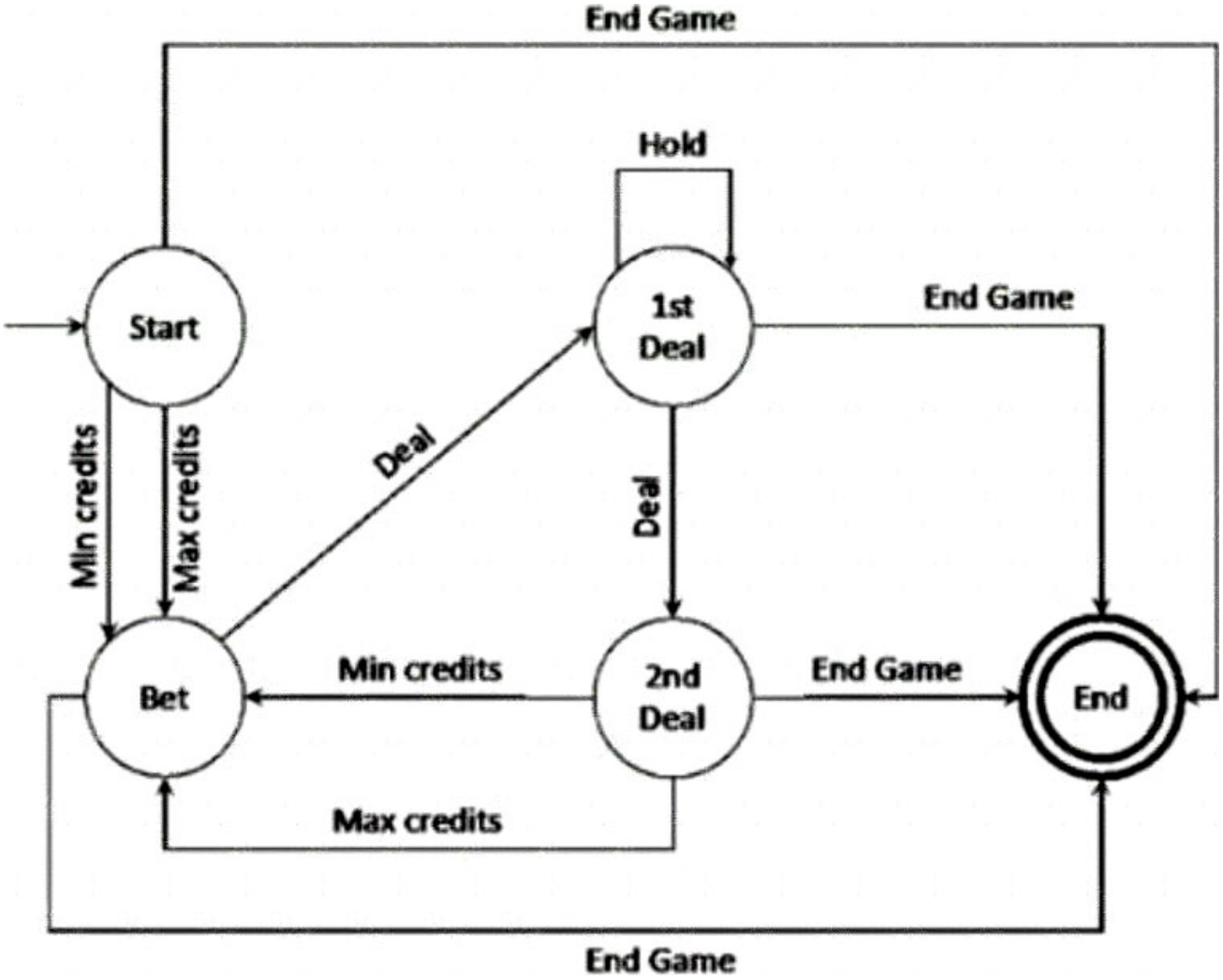
Reference: ISTQB Certified Tester Foundation Level (CTFL) v4.0 sources and documents:

ISTQB® Certified Tester Foundation Level Syllabus v4.0, Chapter 2.3.1, Test Coverage Criteria Based on the Structure of the Software

ISTQB® Glossary of Testing Terms v4.0, Branch Coverage, Statement Coverage, Branch, Decision, Decision Outcome, Unconditional Branch, Decision Table, Condition Combination Coverage

5.Consider the following simplified version of a state transition diagram that specifies the behavior of a

video poker game:



What Is the minimum number of test cases needed to cover every unique sequence of up to 3 states/2 transitions starting In the "Start" state and ending In the "End" state?

A. 1

B. 2

C. 3

D. 4

Answer: D

Explanation:

The minimum number of test cases needed to cover every unique sequence of up to 3 states/2 transitions starting in the “Start” state and ending in the “End” state is 4. This is because there are 4 unique sequences of up to 3 states/2 transitions starting in the “Start” state and ending in the “End” state:

Start -> Bet -> End

Start -> Deal -> End

Start -> 1st Deal -> End

Start -> 2nd Deal -> End

Reference: ISTQB Certified Tester Foundation Level (CTFL) v4.0 sources and documents.