



Software Quality Control and Assurance Day 2

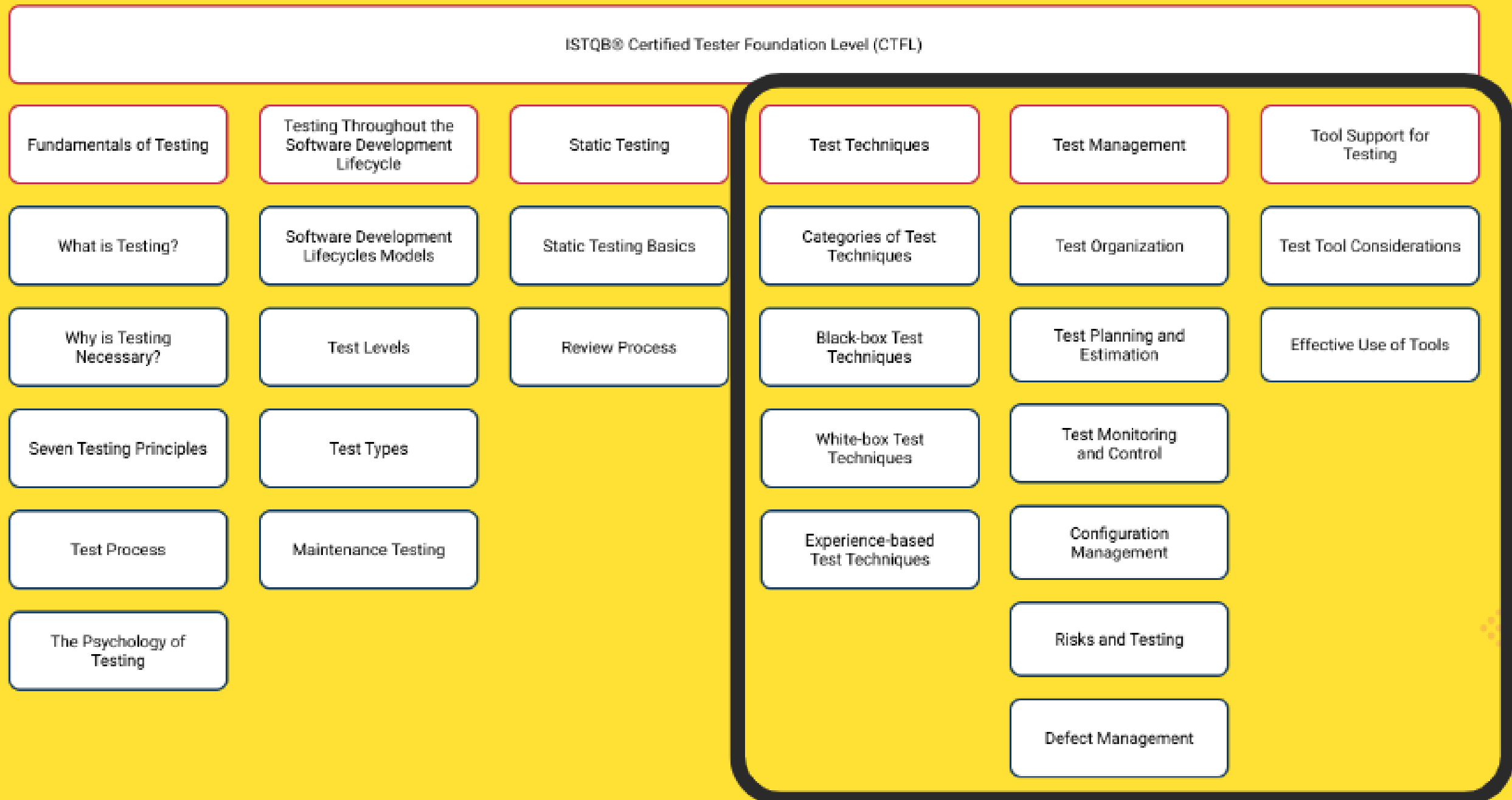
ISTQB-CTFL Syllabus 2023 V4.0

Mae Kristine Clor, ISQTB-CTFL

Day 2

ISTQB-CTFL

ISTQB® Certified Tester Foundation Level (CTFL)



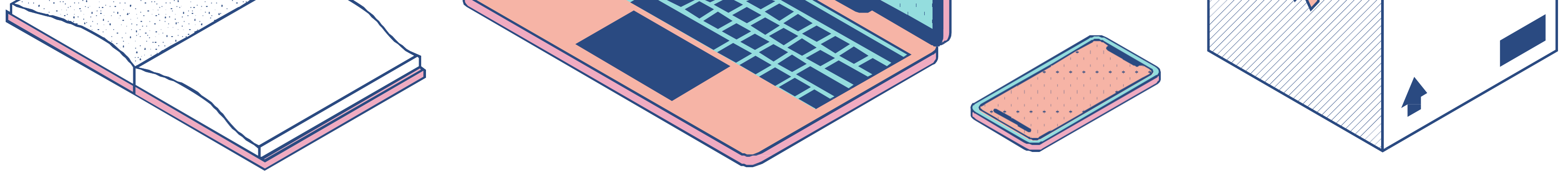
04 Test Techniques

Categories of Test Techniques

Black-box Test Techniques

White-box Test Techniques

Experience-based Test Techniques

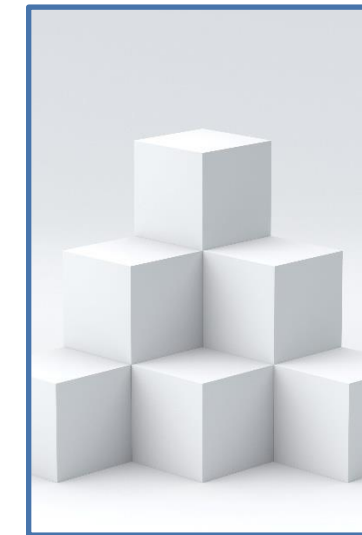


4.1 Categories of Test Techniques



Black-box test techniques

- Also called behavioral or **behavior-based** techniques
- Based on an analysis of the appropriate test basis (e.g., formal requirements documents, specifications, use cases, user stories, or business processes)
- Applicable to both functional and non-functional testing.
- concentrate on the inputs and outputs of the test object



White-box test techniques

- Also called structural or **structure-based** techniques)
- Based on an analysis of the architecture, detailed design, internal structure, or the code of the test object.

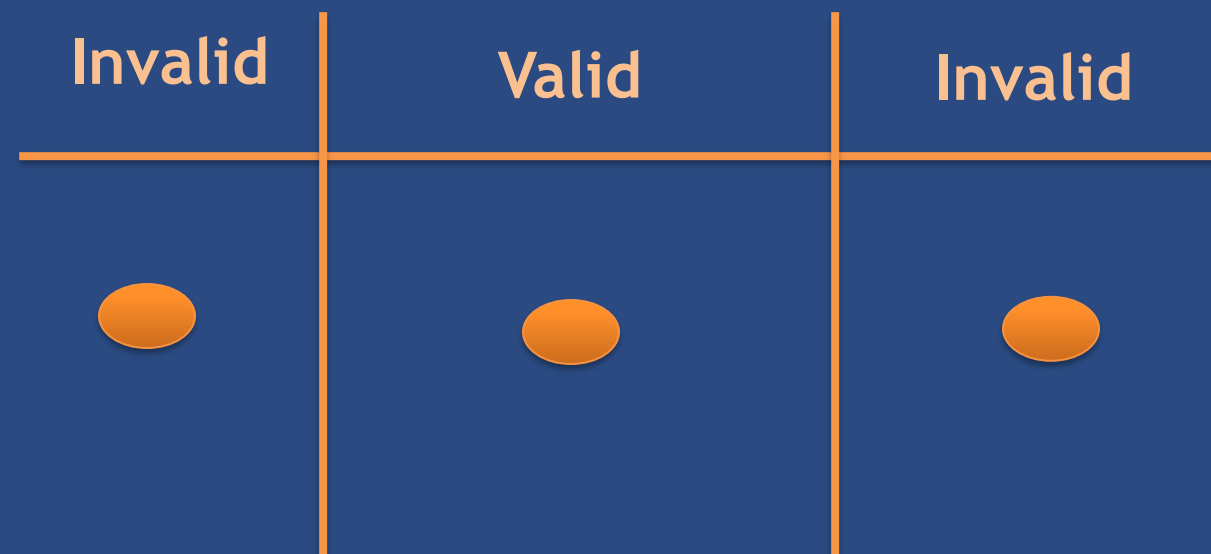


Experience-based test techniques

- Leverage the **experience** of developers, testers and users to design, implement, and execute tests.
- Often combined with black-box and white-box test techniques.



4.2 Black-box Test Techniques



Equivalence Partitioning

divides data into partitions (also known as equivalence classes) in such a way that all the members of a given partition are expected to be processed in the same way for both valid and invalid values.

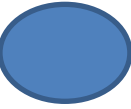


- Valid values are values that should be accepted by the component or system.
- Invalid values are values that should be rejected by the component or system.
- Partitions can be identified for any data element related to the test object, including inputs, outputs, internal values, time-related values.
- Any partition may be divided into sub partitions if required.
- Each value must belong to one and only one equivalence partition.
- When invalid equivalence partitions are used in test cases, they should be tested individually.

Exercise Questions

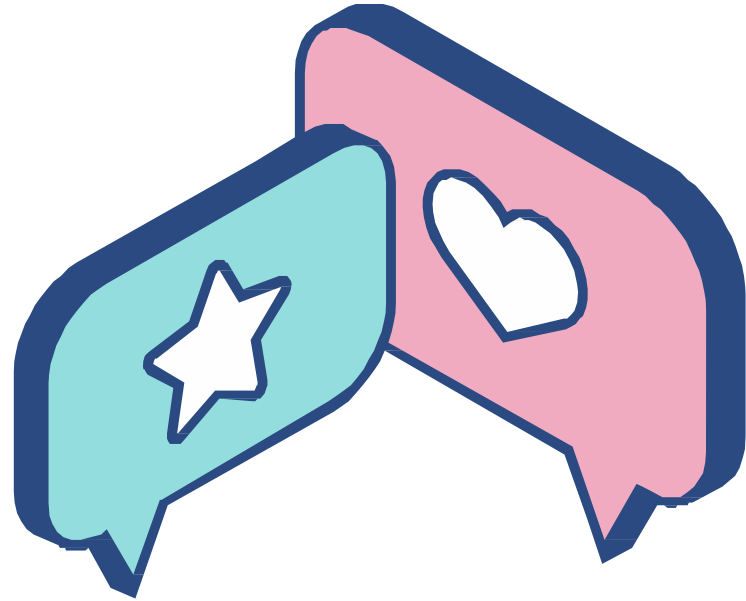


An input field contains values between 1 to 10. If we apply equivalence partition, what is the minimum number of test cases required to get the maximum coverage?

- a) 1
- b) 2
- c) 3
- d) 4

Invalid	Valid	Invalid
≤ 0	1 - 10	≥ 11
		

Exercise Questions



An input field contains values between 1 to 10. If we apply equivalence partition, which of the following is a valid collection of equivalence classes for this scenario?

- a) 0, 1-10, 11
- b) Negative numbers, 1 - 10, 20
- c) < 1 , 1 - 9, > 10



Invalid	Valid	Invalid
≤ 0	1 - 10	≥ 11

Exercise Questions



Consider this tax table:

\$1 - \$3000	0
\$3001 - \$4500	10%
\$4501 - \$32500	20%
> \$32500	30%

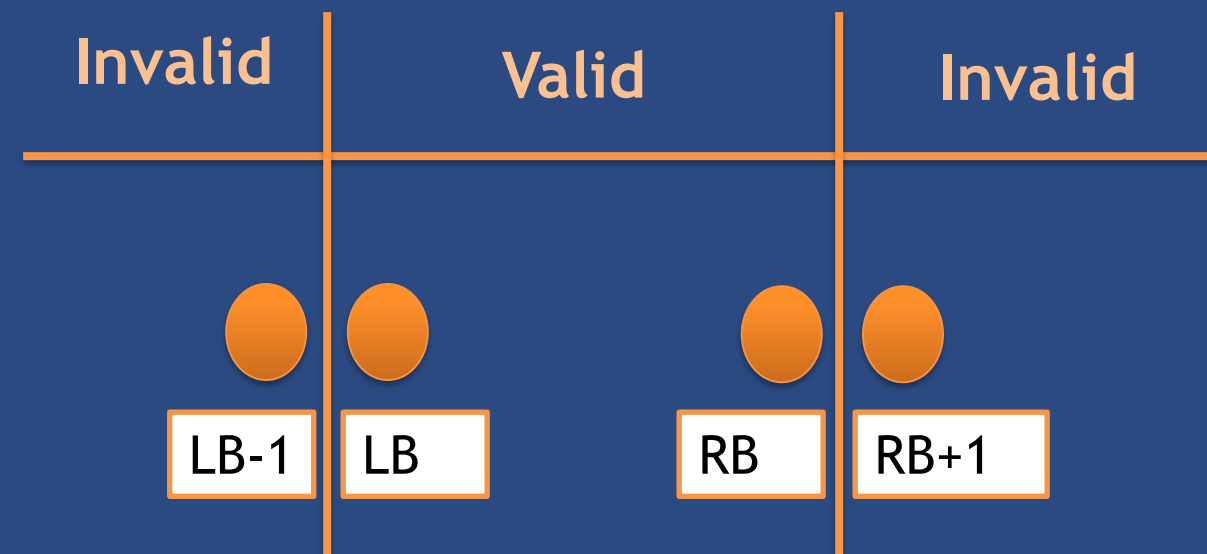
Which of these salary groups would fall into the same equivalence class?

- a) \$3800, \$15000, \$38000
- b) \$6200, \$6500, \$38000
- c) \$32500, \$42000, \$35000

Valid	Valid	Valid	Valid
1 - 3000	3001 - 4500	4501 - 32500	> 32500



4.2 Black-box Test Techniques



Boundary Value Analysis

extension of equivalence partitioning but can only be used when the partition is ordered, consisting of numeric or sequential data.

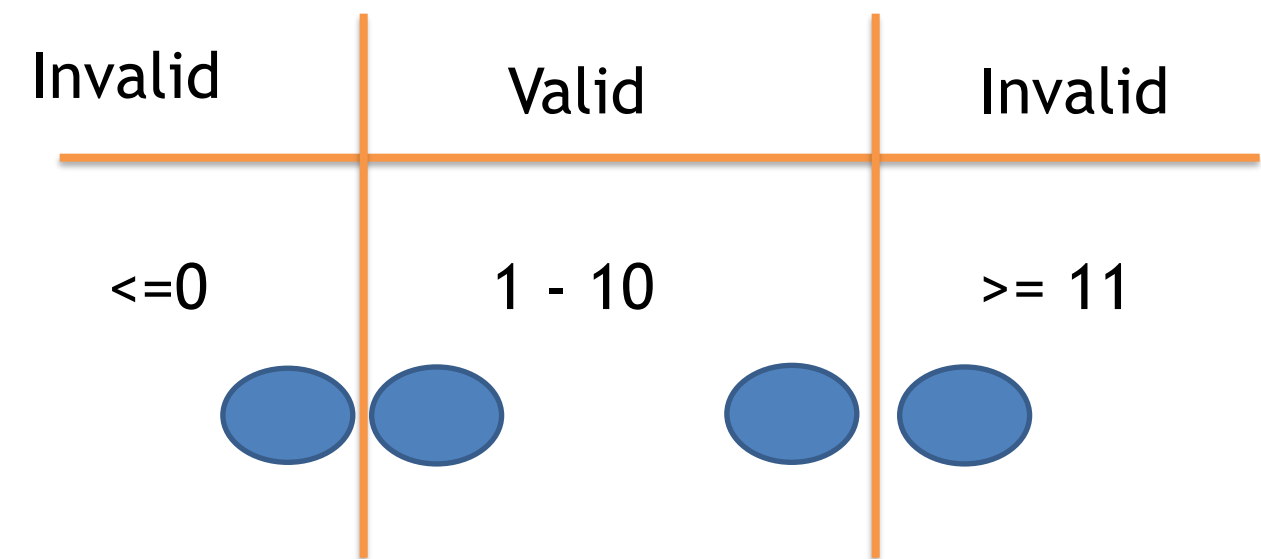
- The minimum and maximum values (or first and last values) of a partition are its boundary values.
- Behavior at the boundaries of equivalence partitions is more likely to be incorrect than behavior within the partitions.
- Can be applied at all test levels. This technique is generally used to test requirements that call for a range of numbers (including dates and times).

Exercise Questions



An input field contains values between 1 to 10. If we apply boundary value analysis, what is the minimum number of test cases required to get the maximum coverage?

- a) 1
- b) 2
- c) 3
- d) 4

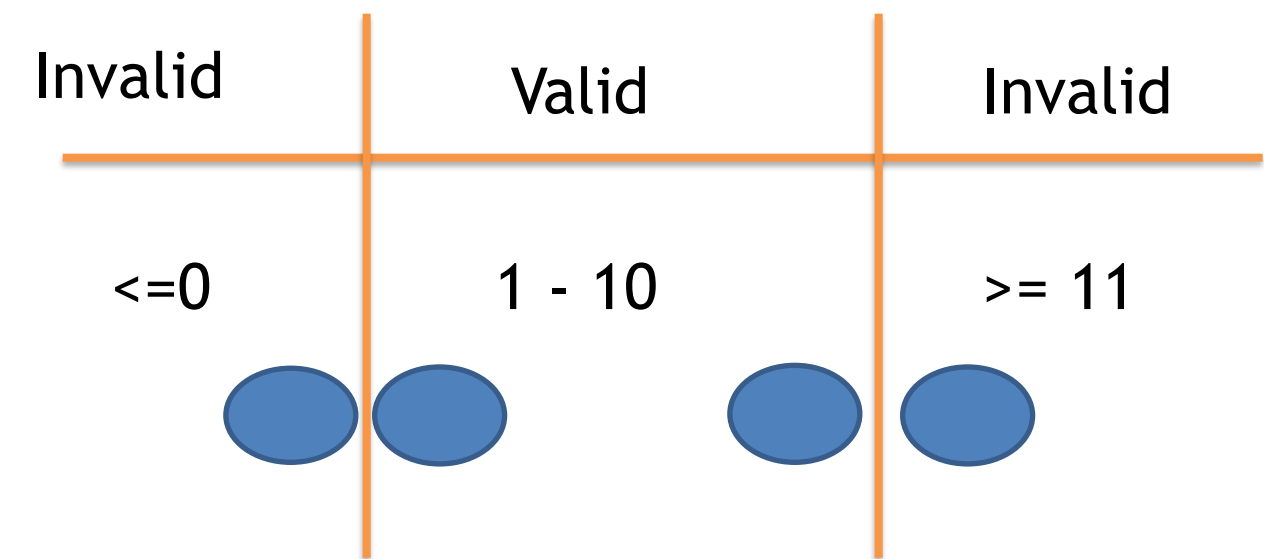


Exercise Questions

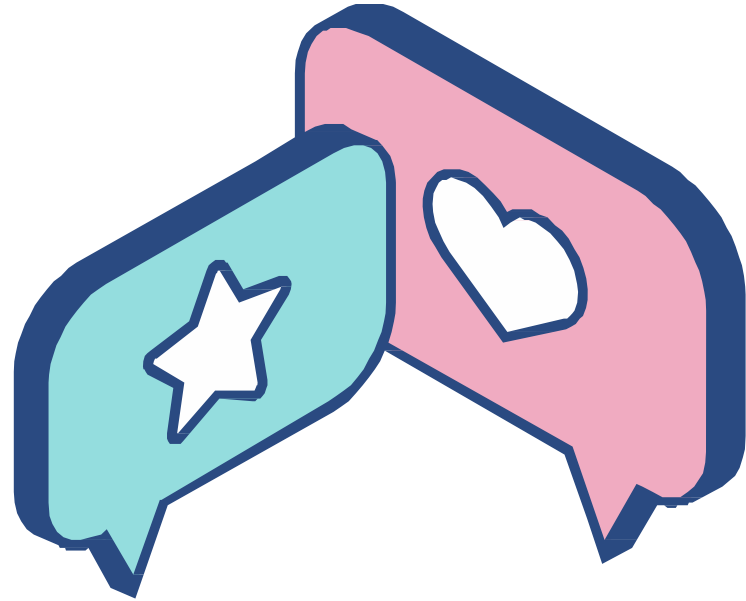


An input field contains values between 1 to 10. If we apply boundary value analysis, which of the following is a valid collection of boundary values?

- a) -1, 1, 11, 12
- b) 0, 8, 9, 10
- c) 0, 1, 10, 11

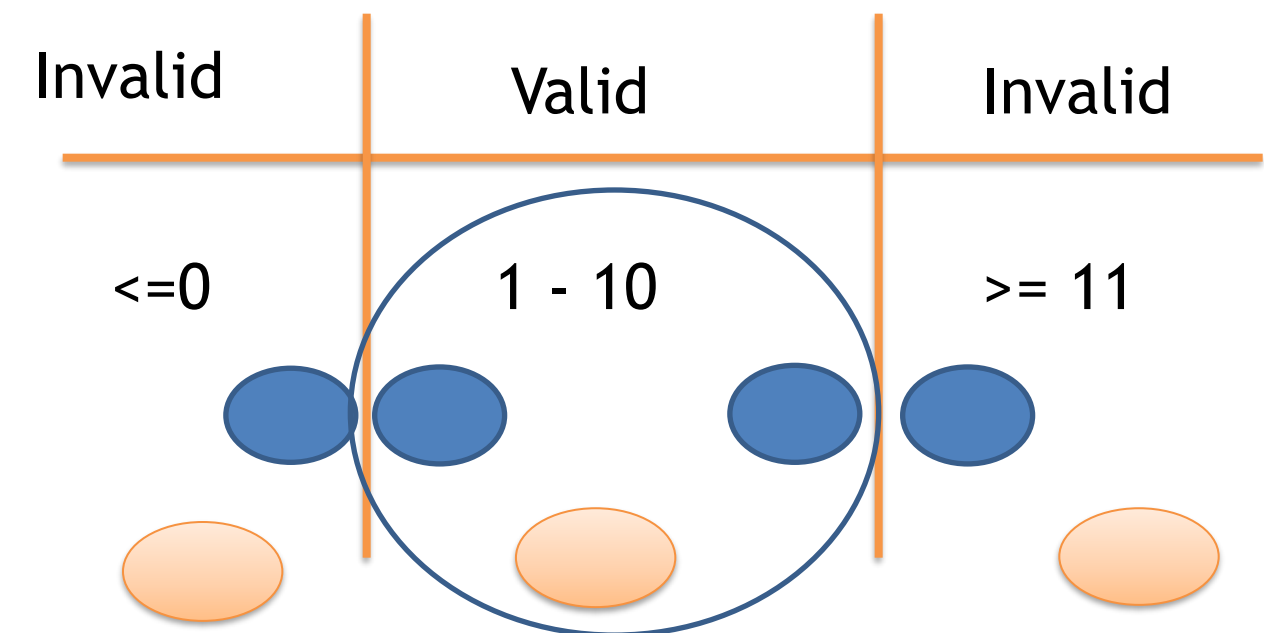


Exercise Questions



An input field contains values between 1 to 10. If we apply EP and BVA, which of the following is a valid collection of boundary values and equivalence values?

- a) 2, 5, 9
- b) 0, 5, 11
- c) -1, 3, 4





4.2 Black-box Test Techniques

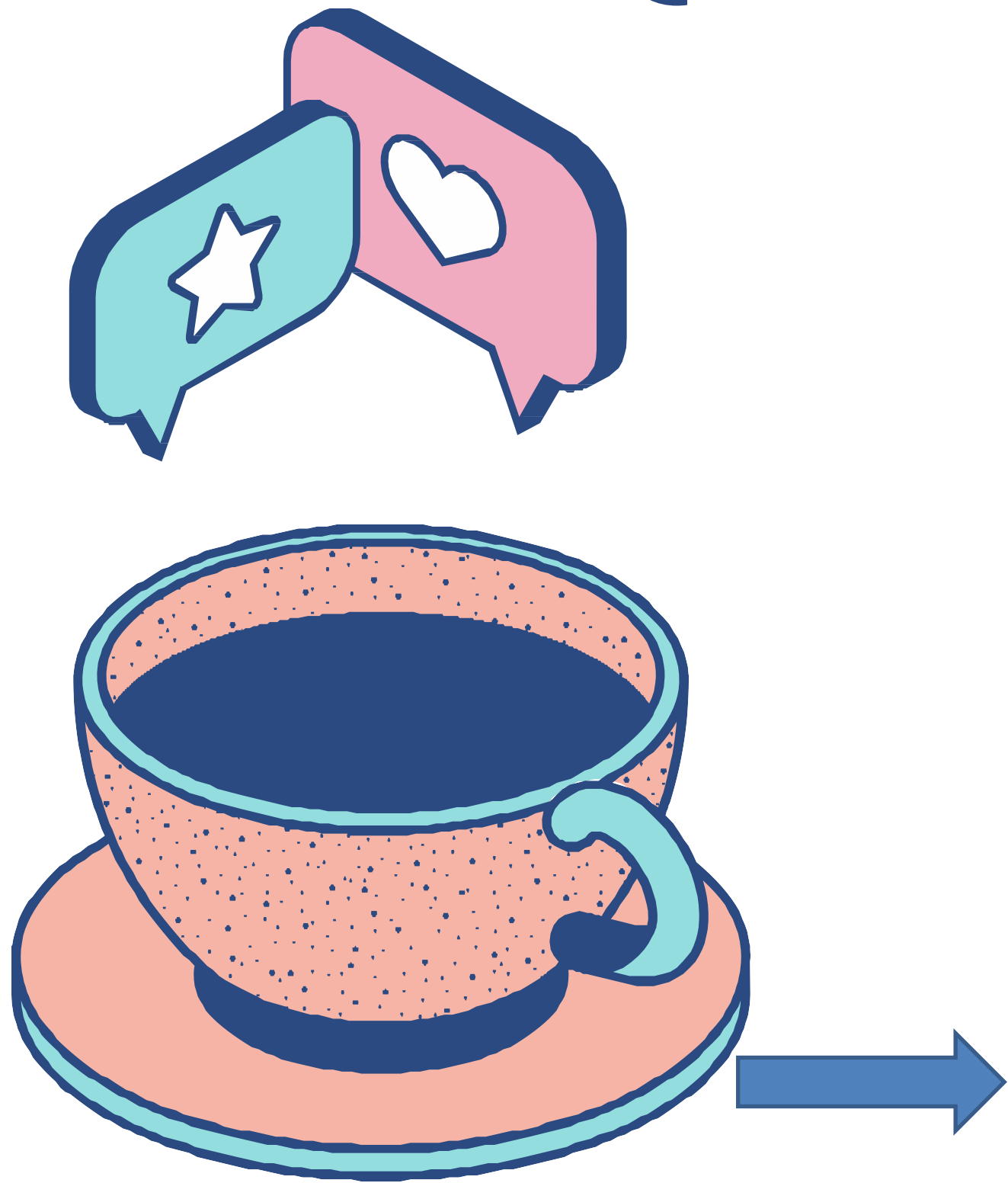
Conditions	R1	R2	R3
Withdrawal Amount <= Balance	T	F	F
Credit granted	-	T	F
Actions			
Withdrawal granted	T	T	F

Decision Table Testing

good way to record complex business rules that a system must implement.

- When creating decision tables, the tester identifies conditions (often inputs) and the resulting actions (often outputs) of the system.
- The common notation in decision tables is as follows:
 - For **conditions**:
 - Y means the condition is true (may also be shown as T or 1)
 - N means the condition is false (may also be shown as F or 0)
 - — means the value of the condition doesn't matter (may also be shown as N/A)
 - For **actions**:
 - X means the action should occur (may also be shown as Y or T or 1)
 - Blank means the action should not occur (may also be shown as - or N or F or 0)

Exercise Questions



Given the following decision table, what is the expected result for each test cases?

Conditions	R1	R2	R3
Withdrawal Amount <= Balance	T	F	F
Credit granted	-	T	F
Actions			
Withdrawal granted	T	T	F

Test Case 1 - Withdrawal Amount >= Balance, Credit granted = False
Test Case 2 - Withdrawal Amount <= Balance, Credit granted = True

- a) Test Case 1 - Withdrawal granted = T, Test Case 2 - Withdrawal granted = T
- b) Test Case 1 - Withdrawal granted = F, Test Case 2 - Withdrawal granted = T

Exercise Questions



Given the following decision table, which of the following test cases and expected results are valid?

Conditions	R1	R2	R3
Withdrawal Amount \leq Balance	T	F	F
Credit granted	-	T	F
Actions			
Withdrawal granted	T	T	F

- a) withdrawal Amount is \$100 and Balance is \$500 Withdrawal granted is T
- b) withdrawal Amount is \$100 and Balance is \$99 Withdrawal granted is T

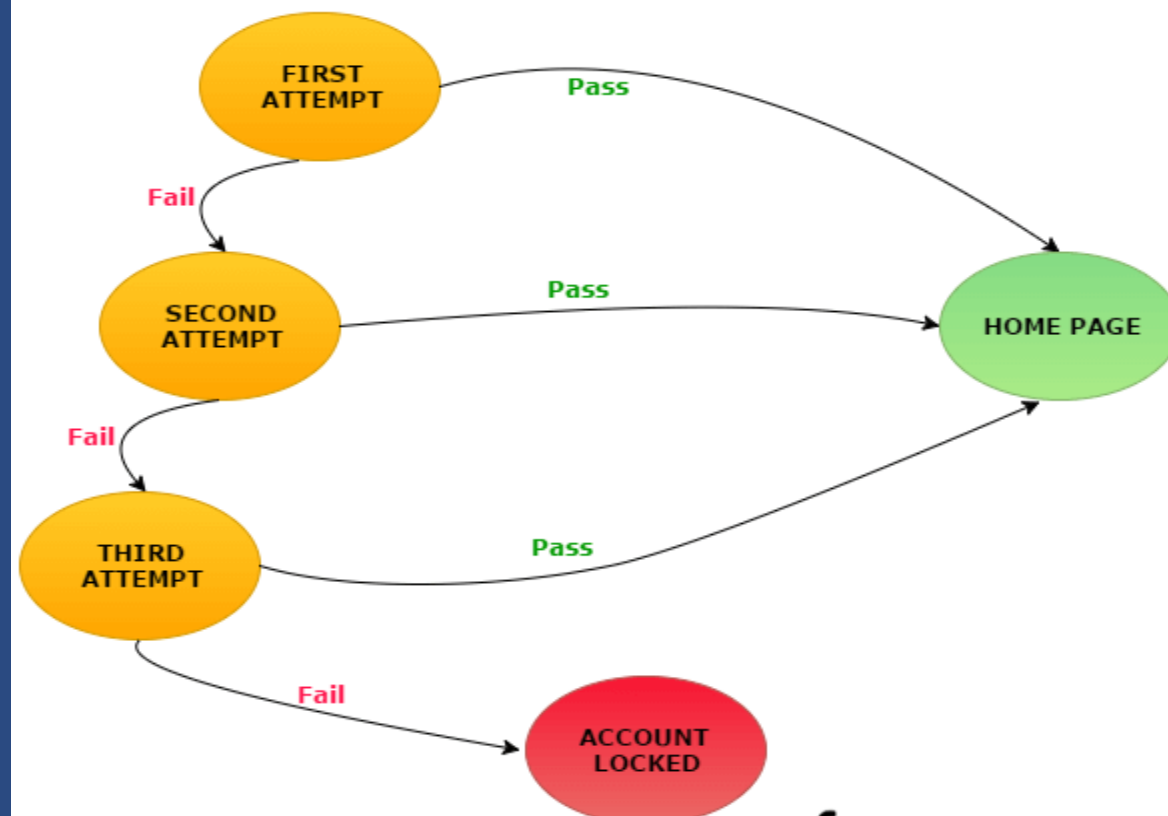
4.2 Black-box Test Techniques

State Transition Testing

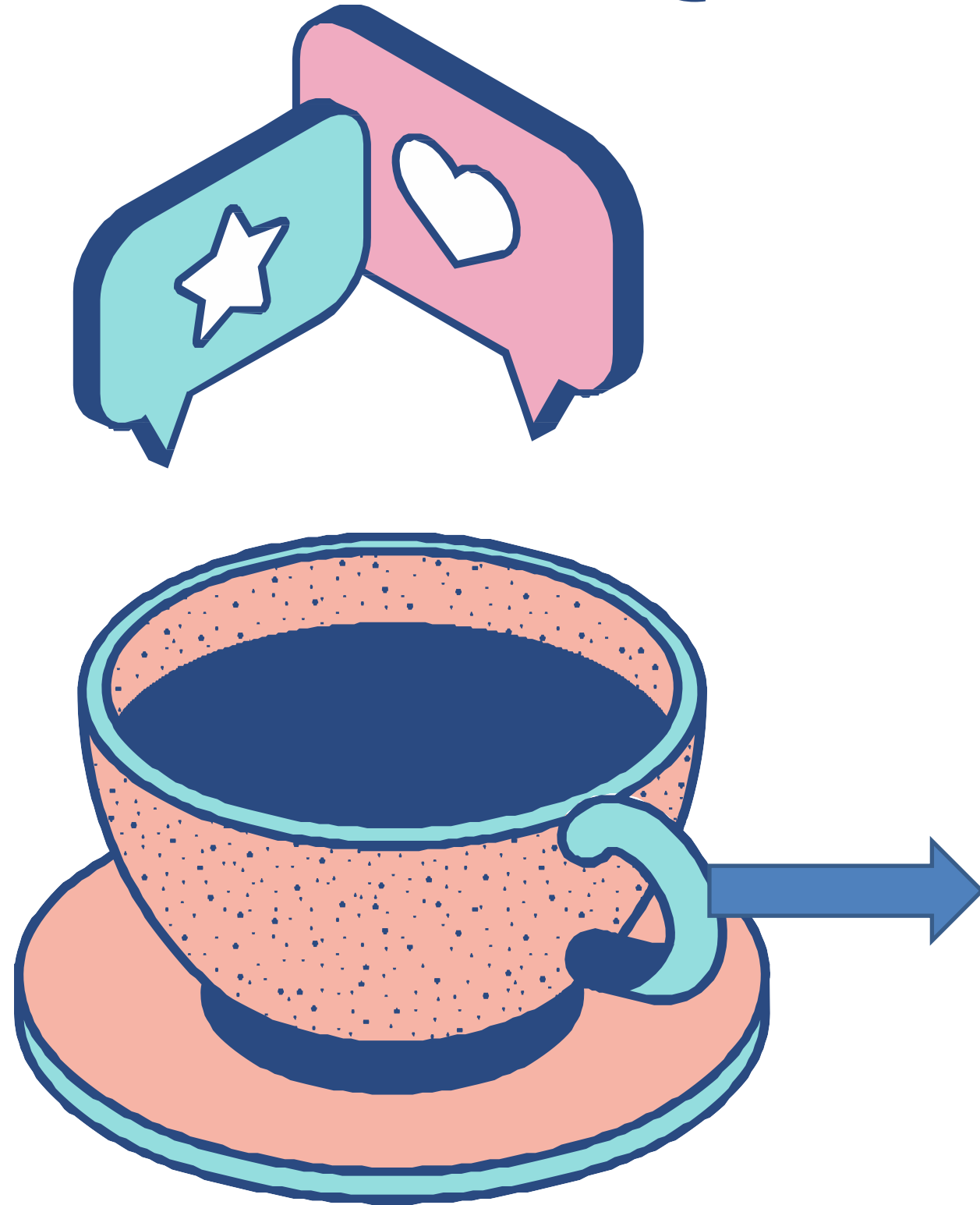
state transition diagram shows the possible software states, as well as how the software enters, exits, and transitions between states.

- A transition is initiated by an event. The event results in a transition.
- The same event can result in two or more different transitions from the same state.
- The state change may result in the software taking an action
- A state transition table shows all valid transitions and potentially invalid transitions between states
- Used for menu-based applications and is widely used within the embedded software industry.

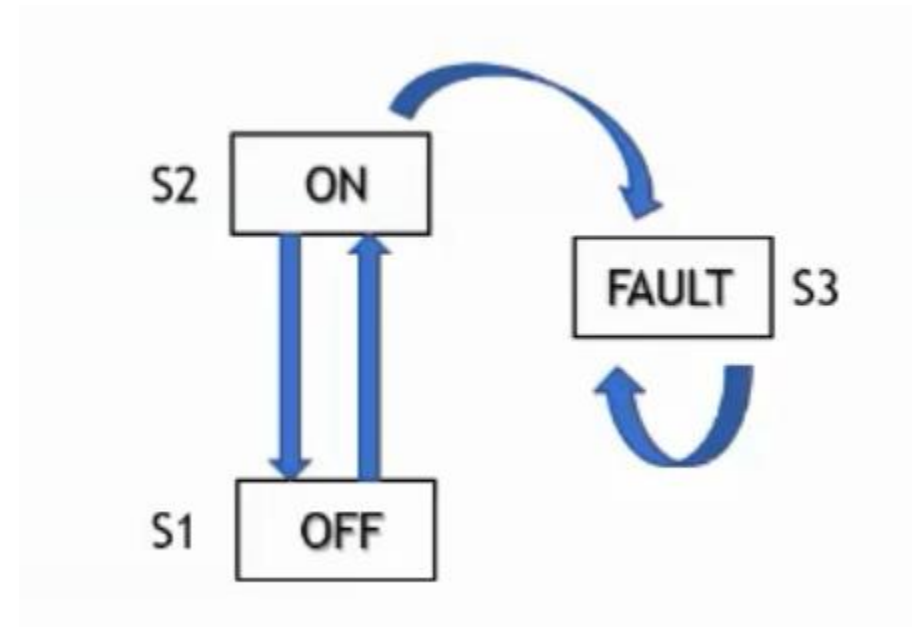
STATE TRANSITION



Exercise Questions



How many test cases can be created in this state transition diagram?

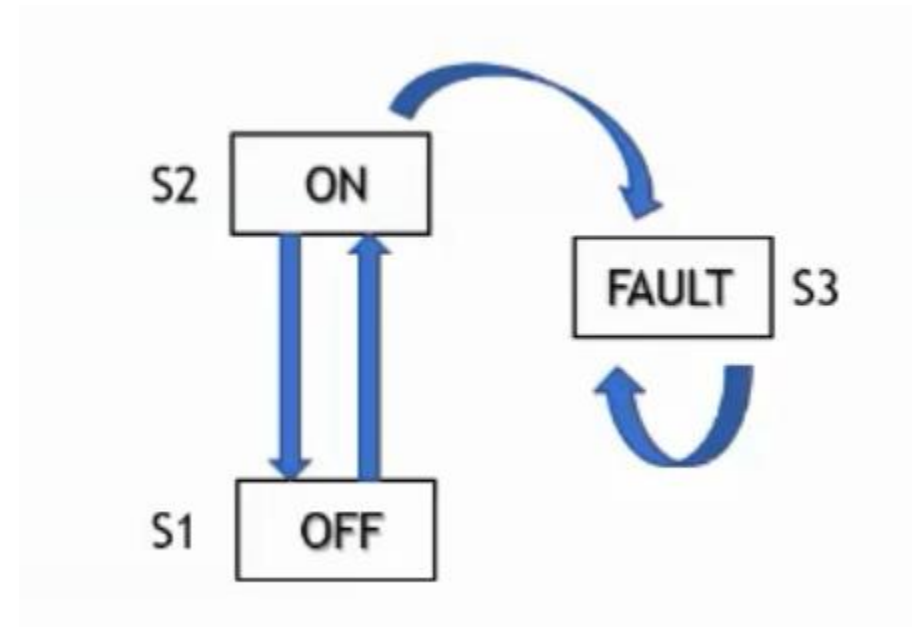


- a) 2
- b) 4
- c) 3

TC 1 - S2 > S3 (ON to FAULT)
TC2 - S2 > S1 (ON to OFF)
TC3 - S1 > S2 (OFF to ON)
TC4 - S3 > S3 (FAULT to FAULT)

Exercise Questions

Based from this state transition diagram, which test case is invalid?



- a) ON to OFF
- b) ON to FAULT
- c) OF to FAULT



4.2 Black-box Test Techniques

Main Success Scenario A: Actor S: System	Step	Description
	1	A: Inserts card
	2	S: Validates card and asks for PIN
	3	A: Enters PIN
	4	S: Validates PIN
	5	S: Allows access to account
Extensions	2a	Card not valid S: Display message and reject card
	4a	PIN not valid S: Display message and ask for re-try (twice)
	4b	PIN invalid 3 times S: Eat card and exit

Use Case Testing

specific way of designing interactions with software items. They incorporate requirements for the software functions.

- Use case specifies some behavior that a subject can perform in collaboration with one or more actors.
- Can include possible variations of its basic behavior, including exceptional behavior and error handling
- A use case can be described by interactions and activities, as well as preconditions, postconditions, and natural language where appropriate.
- Interactions between the actors and the subject may result in changes to the state of the subject. Interactions may be represented graphically by workflows, activity diagrams, or business process models.

4.3 White-box Test Techniques

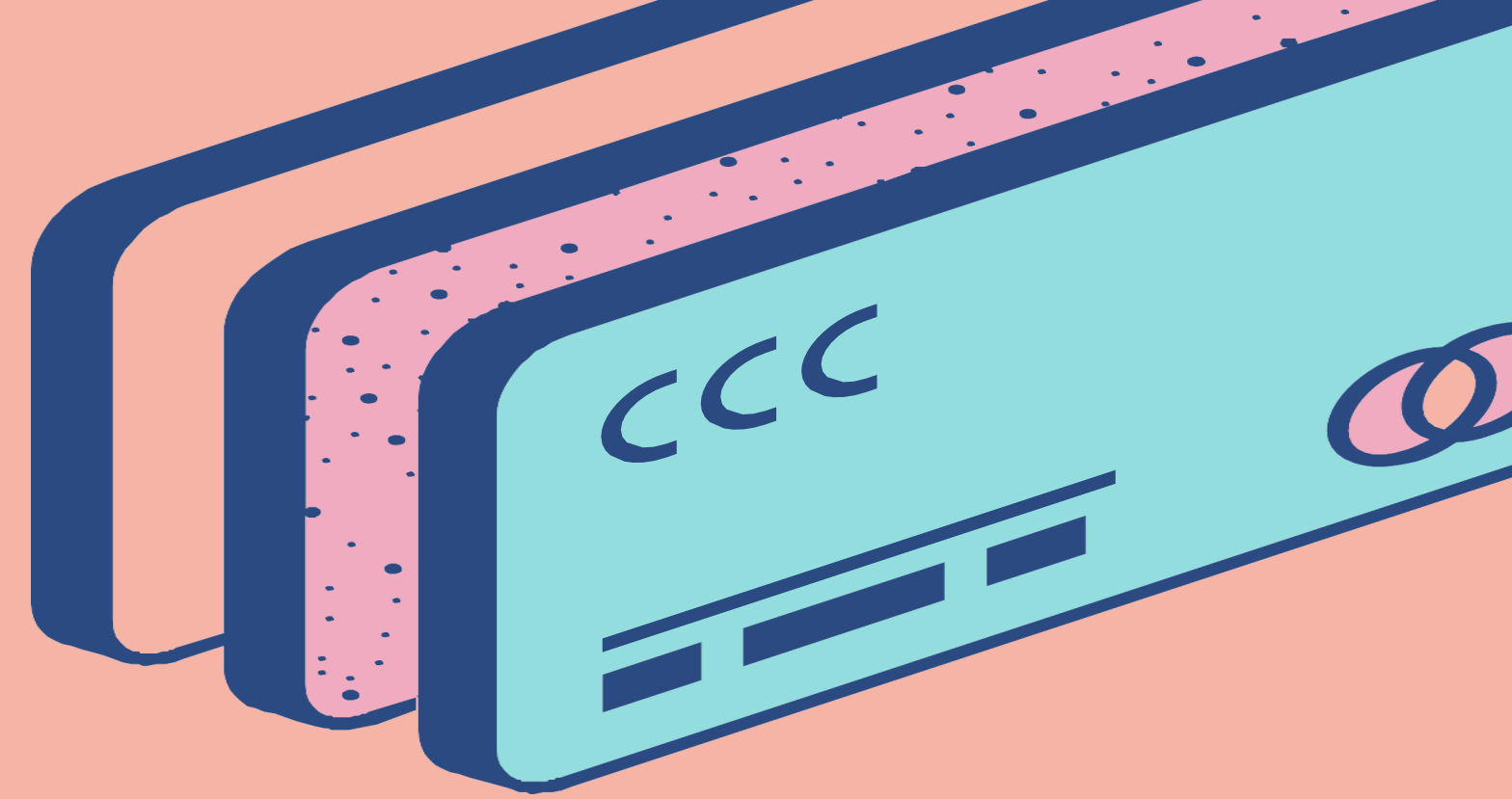
Statement Testing and Coverage

- Exercises the potential **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.

Decision Testing and Coverage

- The **decisions** in the code and tests the code that is executed based on the decision outcomes.
- 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.

$$\% \text{ Coverage} = \frac{\text{executed statements or decision outcomes}}{\text{total no. of executable statements or decision outcomes}} \times 100$$



The Value of Statement and Decision Testing

- When 100% statement coverage is achieved, it ensures 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.
- When 100% decision coverage is achieved, it executes all decision outcomes, which includes testing the true outcome and also the false outcome.

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

Exercise Questions

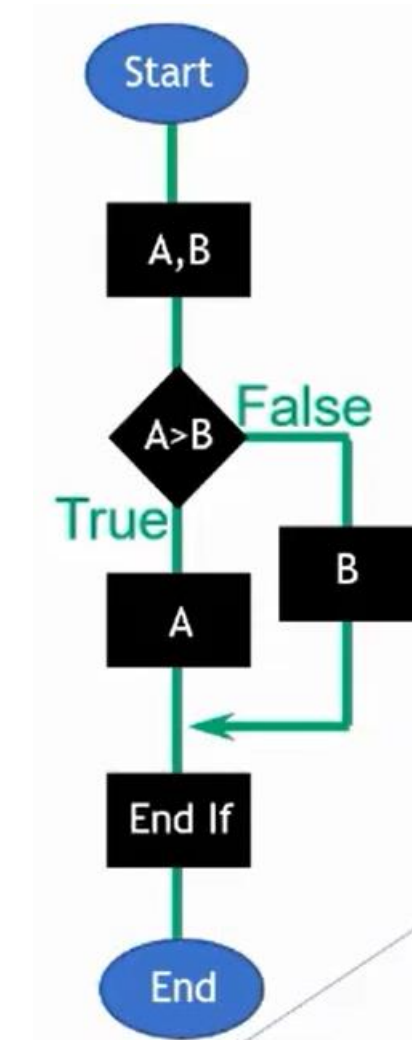


What is the minimum number is test cases required for 100% statement and decision coverage?

```
Read A
Read B
If A>B then
    Print "A is Bigger"
Else
    Print "B is Bigger"
End If
What is the minimum number test cases
required for 100% statement coverage?
```

Statement Coverage = 2

Decision Coverage = 2



Exercise Questions



What is the minimum number of test cases required for 100% statement coverage?

```
Wait for card to be inserted
IF card is a valid card THEN
    display "Enter PIN number"
    IF PIN is valid THEN
        select transaction
    ELSE (otherwise)
        display "PIN invalid"
ELSE (otherwise)
    reject card
```

Statement Coverage = No. of ELSE + 1

Decision Coverage = No. of IF + 1

Statement Coverage = 3

Decision Coverage = 3

Exercise Questions



What is the minimum number of test cases required for 100% statement coverage?

```
Read A
Read B
IF A < 0 THEN
    Print "A negative"
ELSE
    Print "A positive"
ENDIF
IF B < 0 THEN
    Print "B negative"
ELSE
    Print "B positive"
ENDIF
```

Statement Coverage = No. of ELSE
(if unnested)

Decision Coverage = No. of IF
(if unnested)

Statement Coverage = 2

Decision Coverage = 2

4.4

Experience-based Test Techniques



Error Guessing

- Used to anticipate the occurrence of errors, defects, and failures, based on the tester's knowledge, including:
 - How the application has worked in the past
 - What kind of errors tend to be made
 - Failures that have occurred in other applications

Exploratory Testing

- Informal tests are designed, executed, logged, and evaluated dynamically during test execution
- Exploratory testing is conducted within a defined time-box
- Most useful when there are few or inadequate specifications or significant time pressure on testing.

Checklist-based Testing

- Testers design, implement, and execute tests to cover test conditions found in a checklist.
- Checklists can be created to support various test types, including functional and non-functional testing
- In the absence of detailed test cases, checklist-based testing can provide guidelines and a degree of consistency.

4.5. Collaboration-based Test Approaches

Collaborative User Story Writing



- Card - the medium describing a user story (e.g., an index card, an entry in an electronic board)
- Conversation - explains how the software will be used (can be documented or verbal)
- Confirmation - the acceptance criteria
- Format: “As a [role], I want [goal to be accomplished], so that I can [resulting business value for the role]”



Acceptance Criteria

- Conditions that implementation of the user story must meet to be accepted by stakeholders.
- Acceptance criteria are used to:
 - Define the scope of the user story
 - Reach consensus among the stakeholders
 - Describe both positive and negative scenarios
 - Serve as a basis for the user story acceptance testing

4.5. Collaboration-based Test Approaches

User Story for Paying Bills	
User Story	As a regular fintech app user , I want to pay bills to registered billers using the fintech mobile app, so that I can directly settle my bills on time .
Acceptance Criteria	<ul style="list-style-type: none">• The user can log in to the app.• The user can choose the enrolled biller he/she wants to pay.• The user can input the desired amount for payment.• The user receives a receipt via the app.• The user receives a confirmation email and receipt via the enrolled email or phone number.

Acceptance Test-driven Development (ATDD)

- Test cases are created prior to implementing the user story. The test cases are created by team members with different perspectives, e.g., customers, developers, and testers)
- Specification workshop (user stories & acceptance criteria) > Create test cases (team or tester)
- Test cases are based on the acceptance criteria and how the software works
- Expressed in a way that is understandable to stakeholders

Exercise Questions





Which of the following provides the BEST description of exploratory testing?

- a) A testing practice in which an in-depth investigation of the background of the test object is used to identify potential weaknesses that are examined by test cases
- b) An approach to testing whereby the testers dynamically design and execute tests based on their knowledge, exploration of the test item and the results of previous tests
- c) An approach to test design in which test activities are planned as uninterrupted sessions of test analysis and design, often used in conjunction with checklist-based testing
- d) Testing based on the tester's experience, knowledge, and intuition



Which one of the following options is categorized as a black-box test technique?

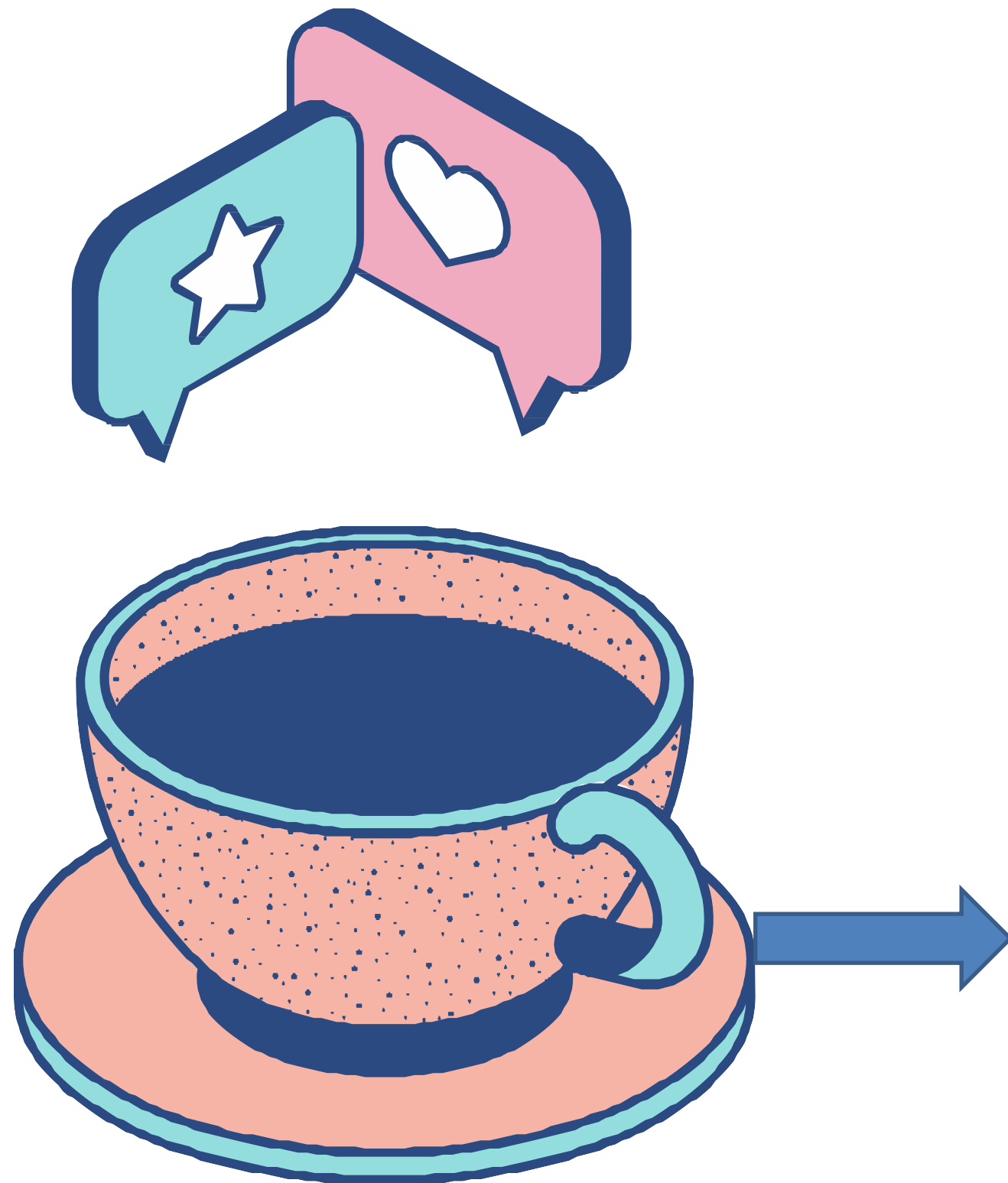
- a) A technique based on analysis of the architecture
- b) A technique checking that the test object is working according to the detailed design
- c) A technique based on the knowledge of past faults, or general knowledge of failures
- d) A technique based on formal requirements



Which statement about the relationship between statement coverage and decision coverage is true?



- a) 100% decision coverage also guarantees 100% statement coverage
- b) 100% statement coverage also guarantees 100% decision coverage
- c) 50% decision coverage also guarantees 50% statement coverage
- d) Decision coverage can never reach 100%



Which of the following BEST matches the descriptions with the different categories of test techniques?

1. Coverage is measured based on a selected structure of the test object
2. The processing within the test object is checked
3. Tests are based on defects' likelihood and their distribution
4. Deviations from the requirements are checked
5. User stories are used as the test basis

Using notation for the following 4 options:

Black - Black-box test techniques

White - White-box test techniques

Experience - Experience-based test techniques

- a) Black - 4, 5 White - 1, 2; Experience - 3
- b) Black - 3 White - 1, 2; Experience - 4, 5
- c) Black - 4 White - 1, 2; Experience - 3, 5
- d) Black - 1, 3, 5 White - 2; Experience - 4



An employee's bonus is to be calculated. It cannot be negative, but it can be calculated down to zero. The bonus is based on the length of employment:

- Less than or equal to 2 years
- More than 2 years but less than 5 years
- 5 to 10 years inclusively
- Longer than 10 years

What is the minimum number of test cases required to cover all valid equivalence partitions for calculating the bonus?

- a) 3
- b) 5
- c) 2
- d) 4



A smart home app measures the average temperature in the house over the previous week and provides feedback to the occupants on their environmental friendliness based on this temperature. The feedback for different average temperature ranges (to the nearest °C) should be:

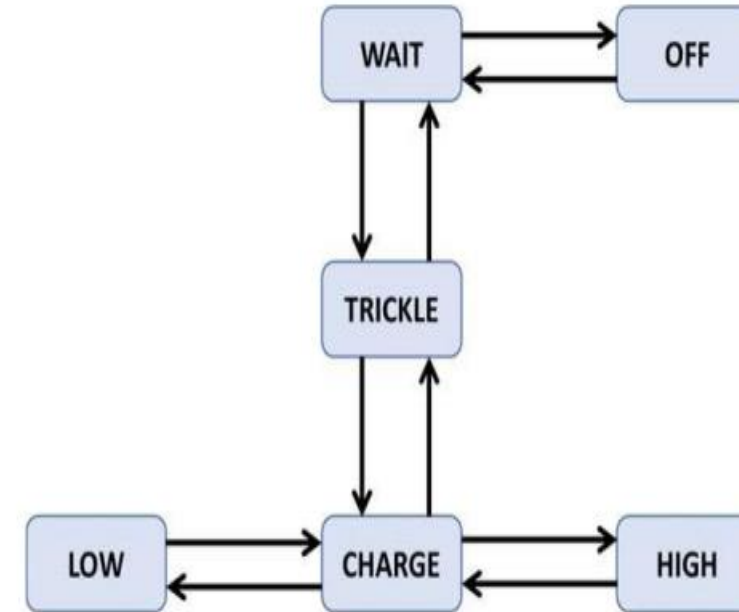
Up to 10°C - Icy Cool!
11°C to 15°C - Chilled Out!
16°C to 19°C - Cool Man!
20°C to 22°C - Too Warm!
Above 22°C - Hot & Sweaty!

Using BVA (only Min- and Max values), which of the following sets of test inputs provides the highest level of boundary coverage?

- a) 0°C, 11°C, 20°C, 22°C, 23°C
- b) 9°C, 15°C, 19°C, 23°C, 100°C
- c) 10°C, 16°C, 19°C, 22°C, 23°C
- d) 14°C, 15°C, 18°C, 19°C, 21°C, 22°C



Given the following state model of a battery charger software:



Which of the following sequences of transitions provides the highest level of transition coverage for the model?

- a) OFF → WAIT → OFF → WAIT → TRICKLE → CHARGE → HIGH → CHARGE → LOW
- b) WAIT → TRICKLE → WAIT → OFF → WAIT → TRICKLE → CHARGE → LOW → CHARGE
- c) HIGH → CHARGE → LOW → CHARGE → TRICKLE → WAIT → TRICKLE → WAIT → TRICKLE
- d) WAIT → TRICKLE → CHARGE → HIGH → CHARGE → TRICKLE → WAIT → OFF → WAIT

05 Managing Test Activities

Test Planning

Risk Management

Test Monitoring, Test
Control and Test Completion

Configuration Management

Defect Management

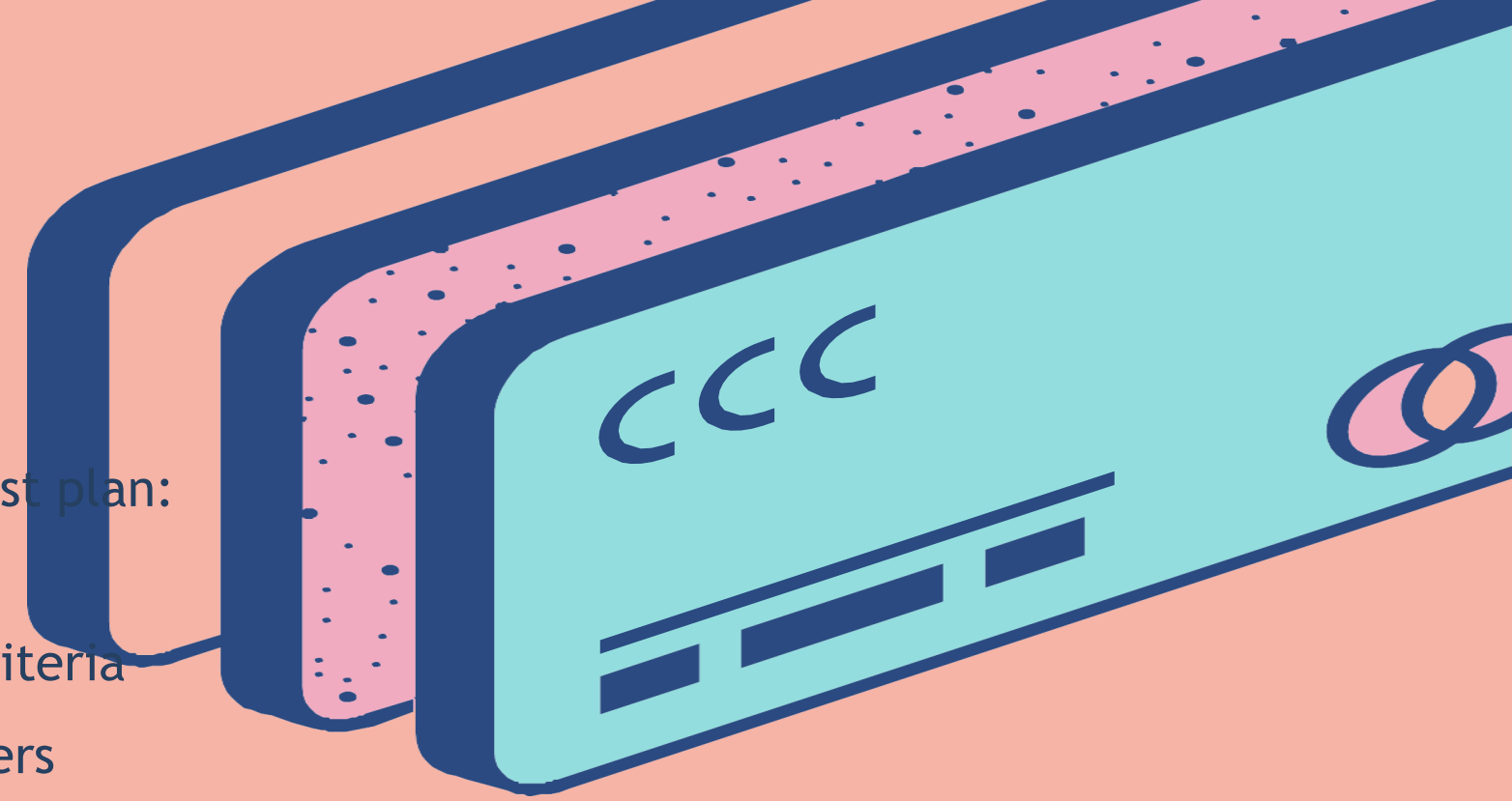
5.1 Test Planning

A test plan describes the objectives, resources and processes for a test project. A test plan:

- Documents the means and schedule for achieving test objectives
- Helps to ensure that the performed test activities will meet the established criteria
- Serves as a means of communication with team members and other stakeholders
- Demonstrates that testing will adhere to the existing test policy and test strategy (or explains why the testing will deviate from them)

The typical content of a test plan includes:

- Context of testing (e.g., scope, test objectives, constraints, test basis)
- Assumptions and constraints of the test project
- Stakeholders (e.g., roles, responsibilities, relevance to testing, hiring and training needs)
- Communication (e.g., forms and frequency of communication, documentation templates)
- Risk register (e.g., product risks, project risks)
- Test approach (e.g., test levels, test types, test techniques, test deliverables, entry criteria and exit criteria, independence of testing, metrics to be collected, test data requirements, test environment requirements, deviations from the organizational test policy and test strategy)
- Budget and schedule



Entry Criteria and Exit Criteria (Definition of Ready and Definition of Done)

Entry and exit criteria should be defined for each test level and test type, and will differ based on the test objectives.

Typical entry criteria

- Availability of testable requirements, user stories, and/or models
- Availability of test items that have met the exit criteria for any prior test levels
- Availability of test environment
- Availability of necessary test tools
- Availability of test data and other necessary resources

Typical exit criteria

- Planned tests have been executed
- A defined level of coverage has been achieved
- The number of unresolved defects is within an agreed limit
- The number of estimated remaining defects is sufficiently low
- The evaluated levels of reliability, performance efficiency, usability, security, and other relevant quality characteristics are sufficient

Test Estimation Techniques



Wideband Delphi - iterative, expert-based technique, experts make experience-based estimations. Each expert, in isolation, estimates the effort.



Three-point estimation. Three estimations are made by the experts: the most optimistic estimation (a), the most likely estimation (m) and the most pessimistic estimation (b). The final estimate (E) is their weighted arithmetic mean.

$$E = (a + 4*m + b) / 6$$
$$\text{measurement error SD} = (b - a) / 6$$



Estimation based on ratios - metrics-based technique, figures are collected from previous projects within the organization, which makes it possible to derive “standard” ratios for similar projects



Extrapolation measurements are taken early in the current project to gather data. the effort required for the remaining work can be approximated by extrapolating this data (usually by applying a mathematical model).

For example, the team may extrapolate the test effort in the forthcoming iteration as the averaged effort from the last three iterations.

If a=6, m=9, and b=18 what is the final estimation?

Solution:

$$E = (6 + 4*9 + 18) / 6 = 10$$

$$SD = (18 - 6) / 6 = 2$$

Test Case Prioritization

Risk-based prioritization

- The order of test execution is based on the results of risk analysis
- Test cases covering the most important risks are executed first

Coverage-based prioritization

- The order of test execution is based on coverage (e.g., statement coverage)
- Test cases achieving the highest coverage are executed first

Requirements-based prioritization

- The order of test execution is based on the priorities of the requirements traced back to the corresponding test cases
- Requirement priorities are defined by stakeholders.
- Test cases related to the most important requirements are executed first.

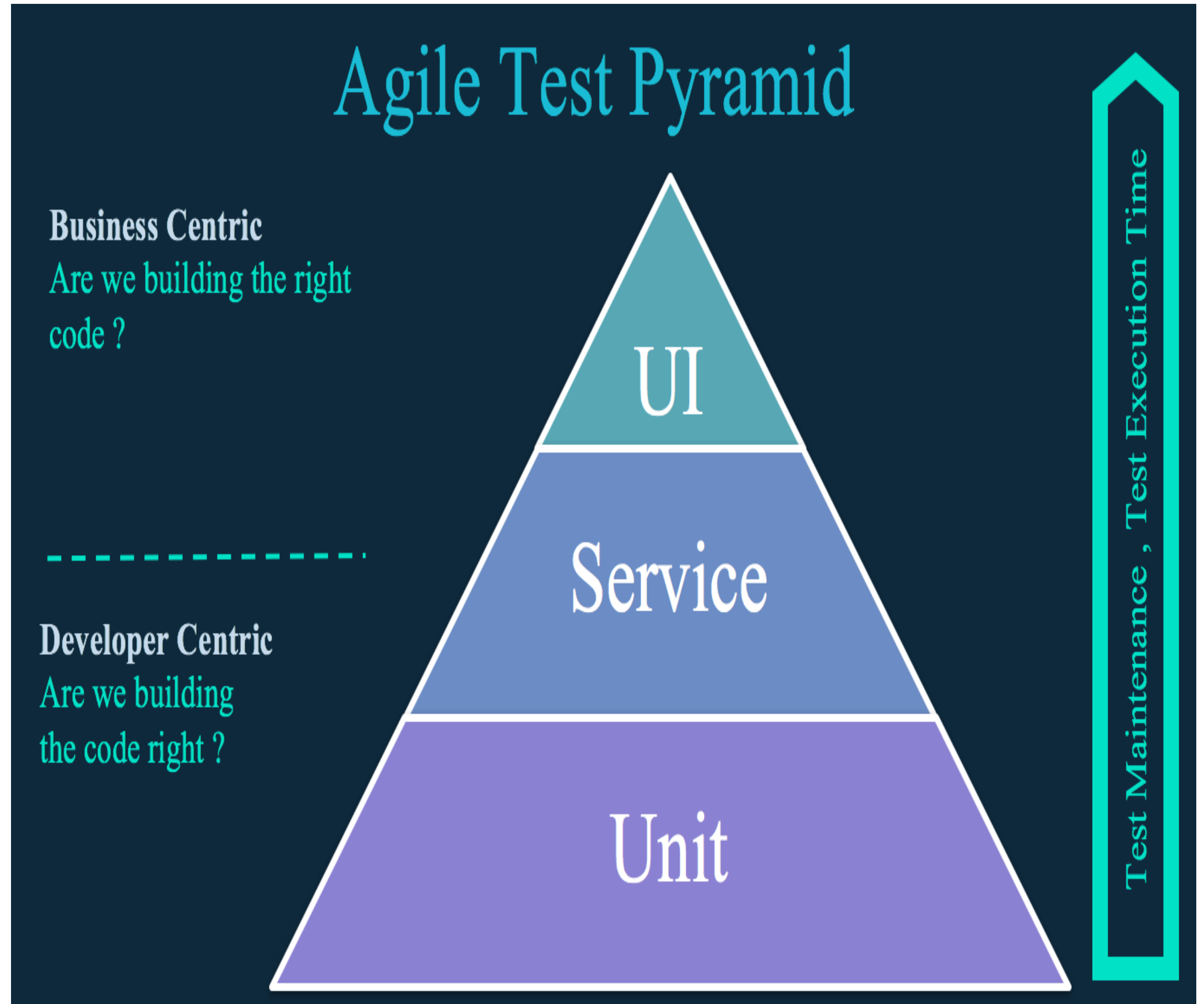


Test Pyramid

UI Tests (Top of the Pyramid): At the top of the pyramid are the UI tests, also known as end-to-end tests. These tests simulate user interactions with the entire application, including the user interface. UI tests ensure that the complete application behaves correctly from the user's perspective.

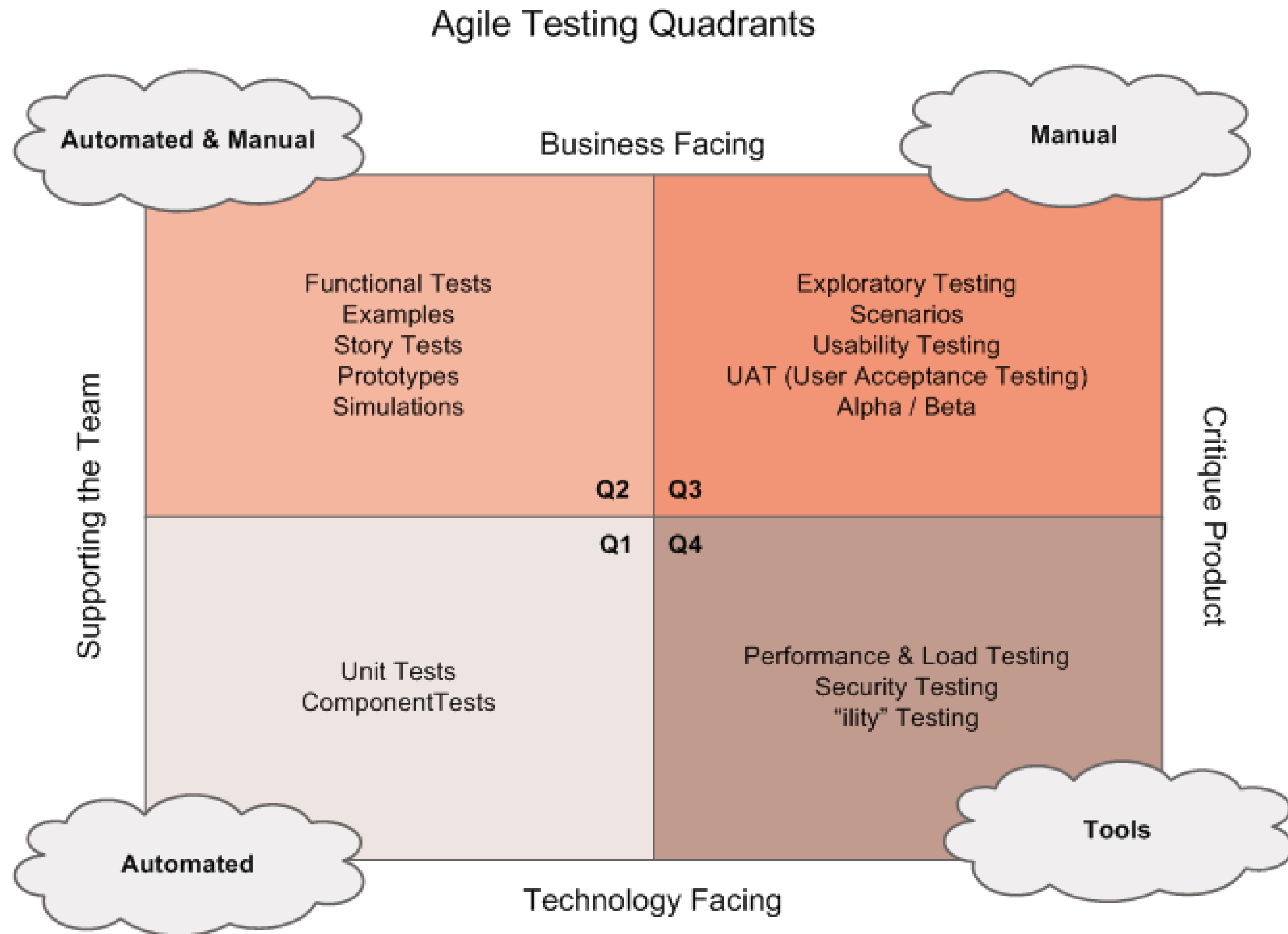
Service Tests (Middle of the Pyramid): Above the unit tests, the next layer consists of service tests, sometimes referred to as component tests or integration tests. These tests validate the interaction between different components or services of the system. Service tests ensure that the components work well together, catching integration issues.

The base of the pyramid is composed of a large number of unit tests. These tests focus on individual code units (functions, methods, classes) and ensure that they work correctly in isolation. Unit tests are typically fast, providing rapid feedback to developers. They are essential for catching basic code errors and regressions



Testing Quadrants

- Quadrant Q1 (technology facing, support the team). This quadrant contains component and component integration tests. These tests should be automated and included in the CI process.
- Quadrant Q2 (business facing, support the team). This quadrant contains functional tests, examples, user story tests, user experience prototypes, API testing, and simulations. These tests check the acceptance criteria and can be manual or automated.
- Quadrant Q3 (business facing, critique the product). This quadrant contains exploratory testing, usability testing, user acceptance testing. These tests are user-oriented and often manual.
- Quadrant Q4 (technology facing, critique the product). This quadrant contains smoke tests and non-functional tests (except usability tests). These tests are often automated.





Risk Management

Risk Definition and Risk Attributes

- Risk is a potential event, hazard, threat, or situation whose occurrence causes an adverse effect
- Risk likelihood - the probability of the risk occurrence (greater than zero and less than one)
- Risk impact (harm) - the consequences of this occurrence

Project Risks

- Project risks are related to the management and control of the project.
- Organizational issues (e.g., delays in work products deliveries, inaccurate estimates, cost-cutting)
- People issues (e.g., insufficient skills, conflicts, communication problems, shortage of staff)
- Technical issues (e.g., scope creep, poor tool support)
- Supplier issues (e.g., third-party delivery failure, bankruptcy of the supporting company)

Product Risks

- Product risks are related to the product quality characteristics
- Examples of product risks include: missing or wrong functionality, incorrect calculations, runtime errors, poor architecture, inefficient algorithms, inadequate response time, poor user experience, security vulnerabilities

Product Risk Analysis

- The goal of product risk analysis is to provide an awareness of product risk in order to focus the testing effort in a way that minimizes the residual level of product risk
- Risk identification is about generating a comprehensive list of risks
- Risk assessment involves: categorization of identified risks, determining their risk likelihood, risk impact and level, prioritizing, and proposing ways to handle them
- Risk assessment can use a quantitative or qualitative approach, or a mix of them

Product Risk Control

- Comprises all measures that are taken in response to identified and assessed product risks
- Risk mitigation involves implementing the actions proposed in risk assessment to reduce the risk level
- The aim of risk monitoring is to ensure that the mitigation actions are effective, to obtain further information to improve risk assessment, and to identify emerging risks

5.3 Test Monitoring, Test Control and Test Completion

- The purpose of test monitoring is to gather information and provide feedback and visibility about test activities.
- Test control describes any guiding or corrective actions taken as a result of information and metrics gathered and (possibly) reported.
- Test completion collects data from completed test activities to consolidate experience, testware, and any other relevant information

Metrics Used in Testing

- Percentage of planned work done in test case preparation (or percentage of planned test cases implemented)
- Percentage of planned work done in test environment preparation
- Test case execution (e.g., number of test cases run/not run, test cases passed/failed, and/or test conditions passed/failed)
- Defect information (e.g., defect density, defects found and fixed, failure rate, and confirmation test results)
- Test coverage of requirements, user stories, acceptance criteria, risks, or code
- Task completion, resource allocation and usage, and effort
- Cost of testing, including the cost compared to the benefit of finding the next defect or the cost compared to the benefit of running the next test



Purposes, Contents, and Audiences for Test Reports

The purpose of test reporting is to summarize and communicate test activity information, both during and at the end of a test

during test activity - test progress report
end of a test activity - test summary report.

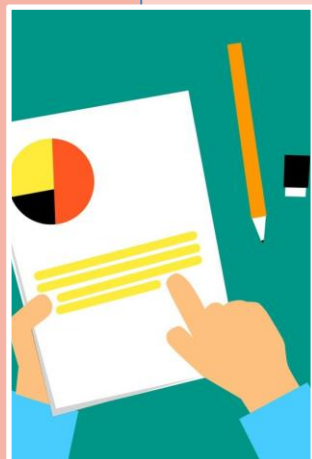
Typical test progress reports include:

- The status of the test activities and progress against the test plan
- Factors impeding progress
- Testing planned for the next reporting period
- The quality of the test object



Typical test summary reports include:

- Summary of testing performed
- Information on what occurred during a test period
- Deviations from plan, including deviations in schedule, duration, or effort of test activities
- Status of testing and product quality with respect to the exit criteria or definition of done
- Factors that have blocked or continue to block progress
- Metrics of defects, test cases, test coverage, activity progress, and resource consumption.
- Residual risks
- Reusable test work products produced



The best means of communicating test status varies, depending on test management concerns, organizational test strategies, regulatory standards, or, in the case of self-organizing teams. Options include:

- Verbal communication with team members and other stakeholders
- Dashboards (e.g., CI/CD dashboards, task boards, and burn-down charts)
- Electronic communication channels (e.g., email, chat)
- Online documentation
- Formal test reports





5.4 Configuration Management

Establish and maintain the integrity of the component or the system, the testware, and their relationships to one another through the project and product lifecycle

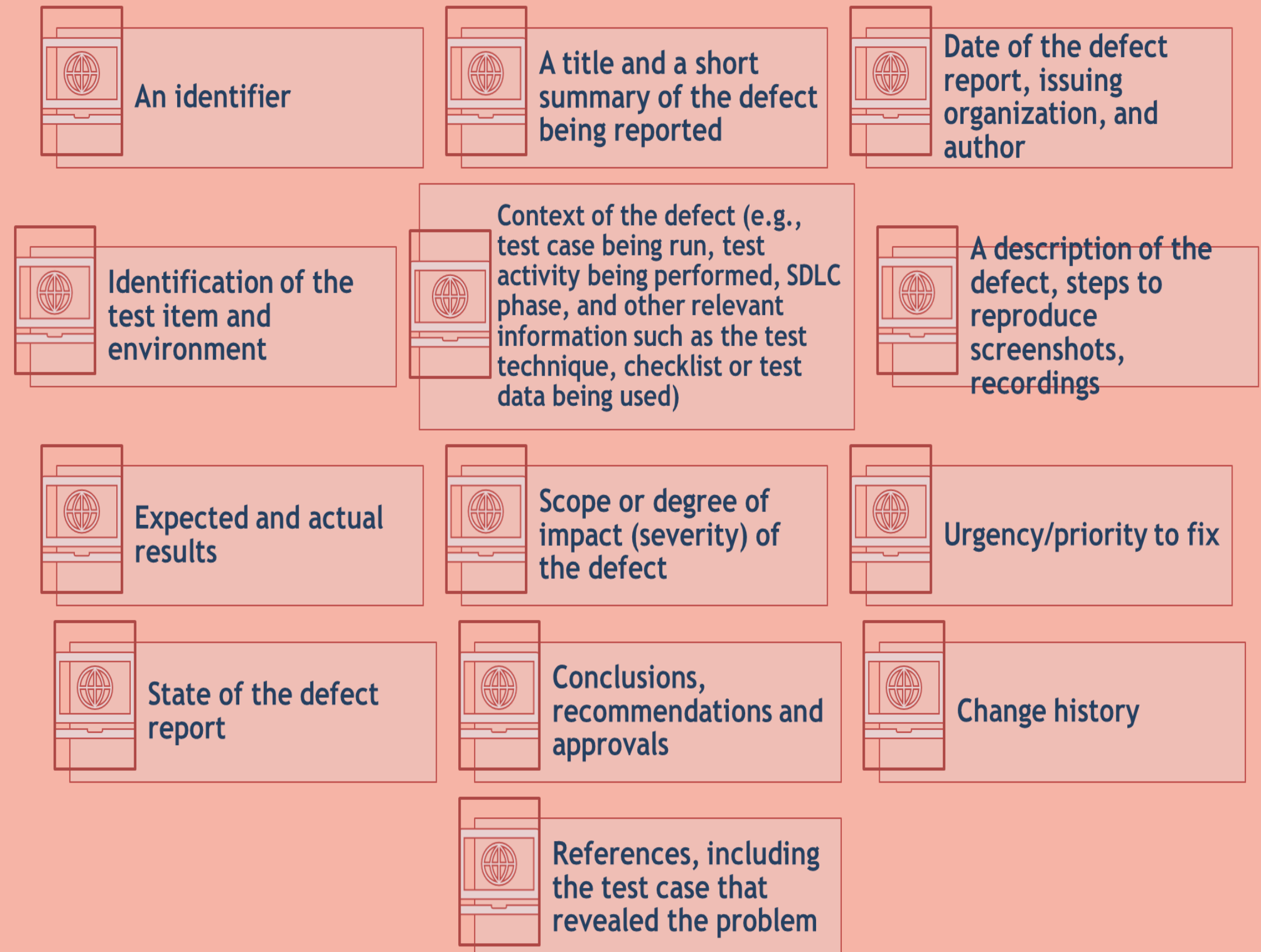
To properly support testing, configuration management may involve ensuring the following:

- 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

5.6 Defect Management

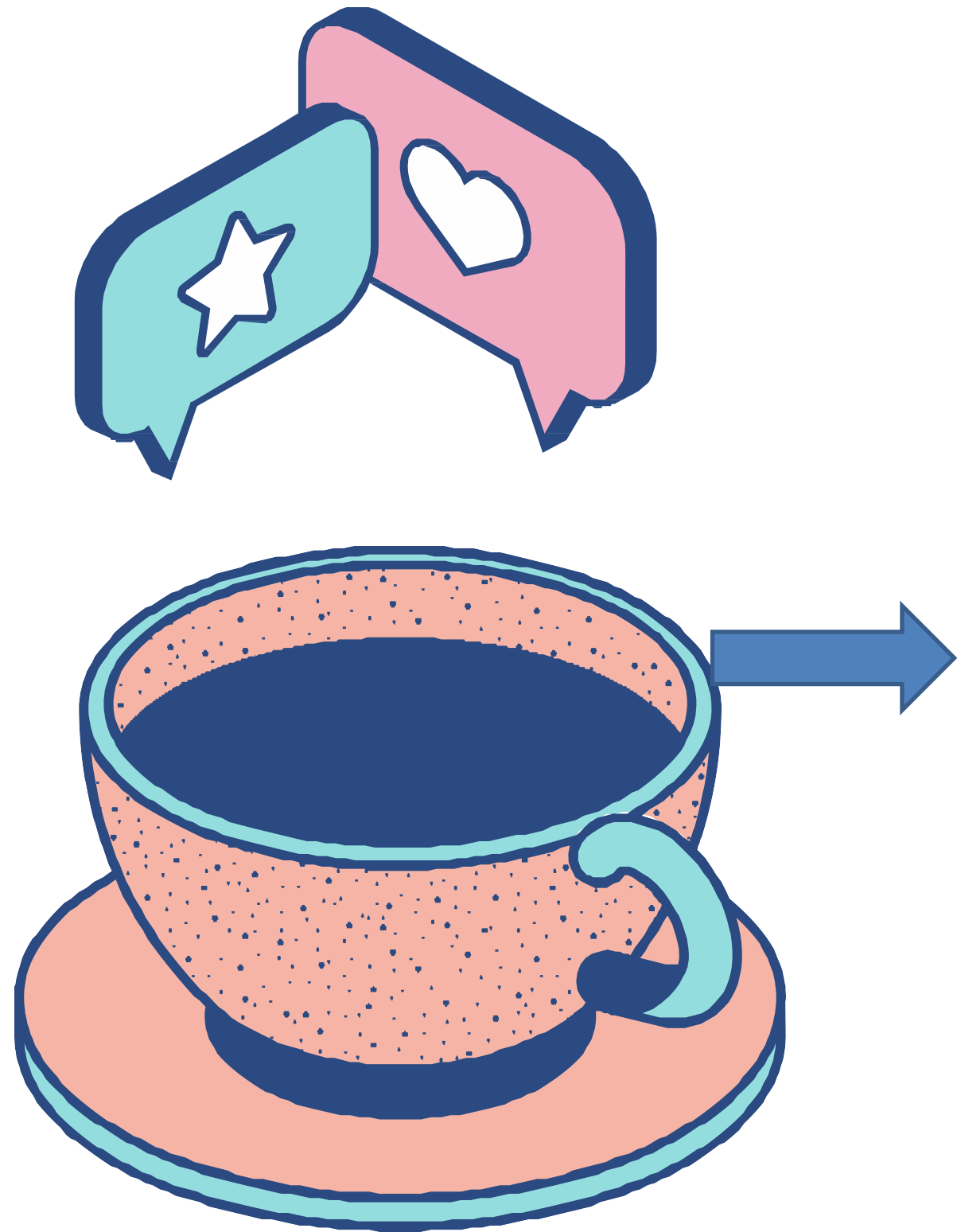
Typical defect reports have the following objectives:

- Provide those responsible for handling and resolving reported defects with sufficient information to resolve the issue
- Provide a means of tracking the quality of the work product
- Provide ideas for improvement of the development and test process



Exercise Questions





Which of the following are two factors that can be used to determine the level of risk?

- a) Testing and development
- b) Dynamic and reactive
- c) Statement and decision
- d) Likelihood and impact



Which of the following metrics would be **MOST** useful to monitor during test execution?

- a) Percentage of executed test cases
- b) Average number of testers involved in the test execution
- c) Coverage of requirements by source code
- d) Percentage of test cases already created and reviewed



Given the following examples of entry and exit criteria:

1. The original testing budget of \$30,000 plus contingency of \$7,000 has been spent
2. 96% of planned tests for the drawing package have been executed and the remaining tests are now out of scope
3. The trading performance test environment has been designed, set-up and verified
4. Current status is no outstanding critical defects and two high-priority ones
5. The autopilot design specifications have been reviewed and reworked
6. The tax rate calculation component has passed unit testing.

Which of the following BEST categorizes them as entry and exit criteria:

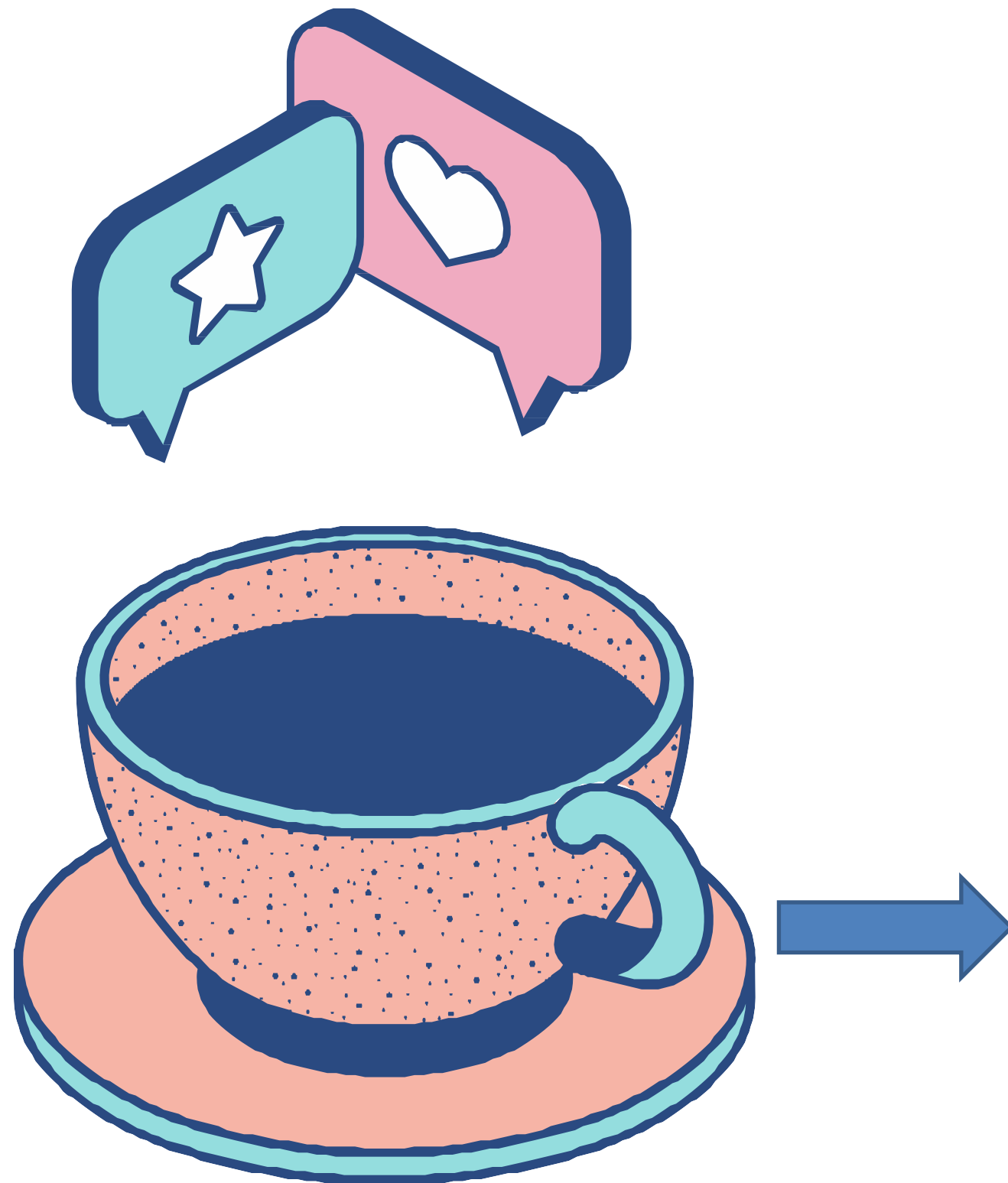
- a) Entry criteria - 5, 6; Exit criteria - 1, 2, 3, 4
- b) Entry criteria - 2, 3, 6; Exit criteria - 1, 4, 5
- c) Entry criteria - 1, 3; Exit criteria - 2, 4, 5, 6
- d) Entry criteria - 3, 5, 6; Exit criteria - 1, 2, 4





Which one of the following is NOT included in a test summary report?

- a) Defining pass/fail criteria and objectives of testing
- b) Deviations from the test approach
- c) Measurements of actual progress against exit criteria
- d) Evaluation of the quality of the test object



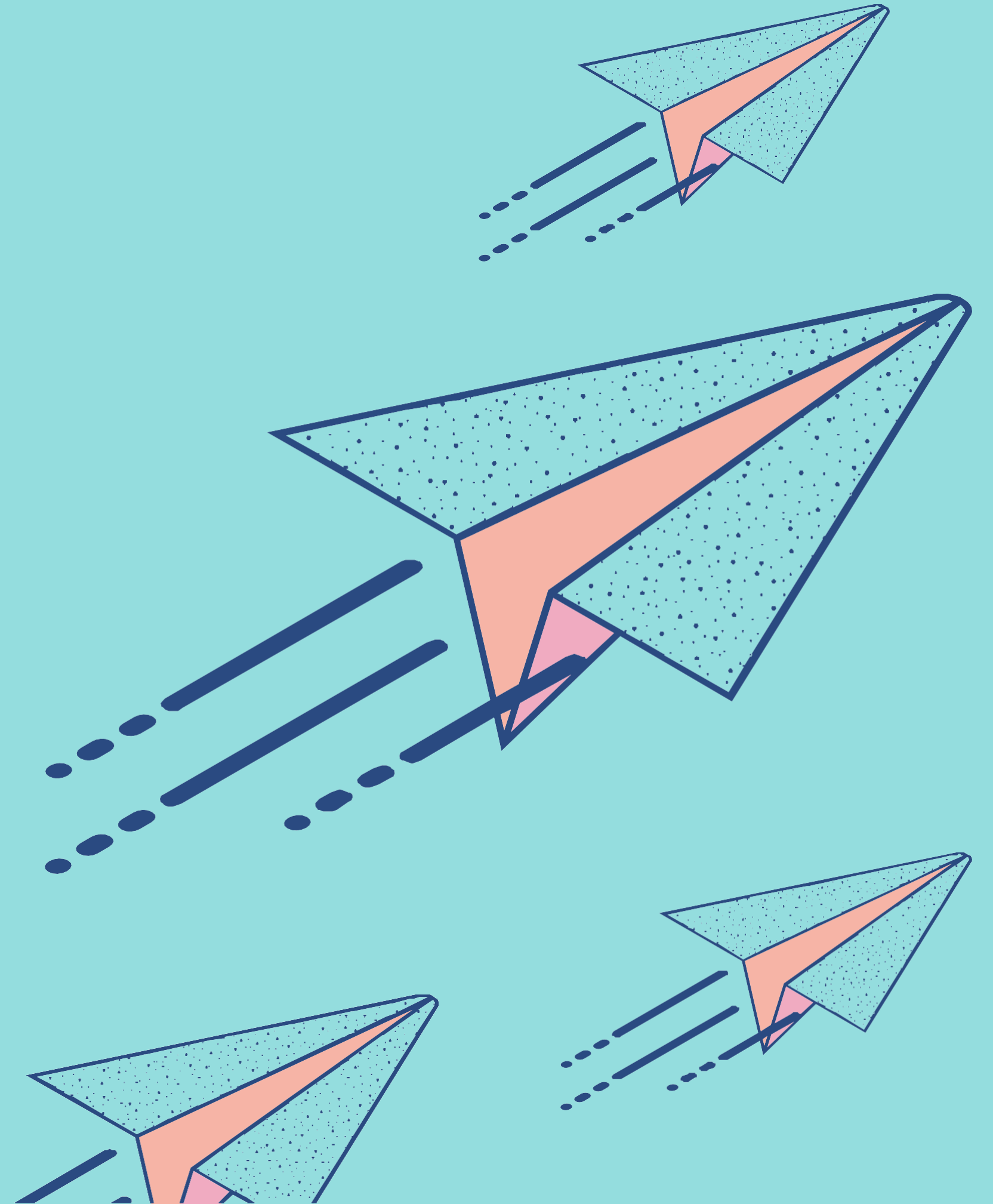
Consider the following list of undesirable outcomes that could occur on a mobile app development effort:

- A. Incorrect totals on screens
- B. Change to acceptance criteria during acceptance testing
- C. Users find the soft keyboard too hard to use with your app
- D. System responds too slowly to user input during search string entry
- E. Testers not allowed to report test results in daily standup meetings

Which of the following properly classifies these outcomes as project and product risks?

- a) Product risks: B, E; Project risks: A, C, D
- b) Product risks: A, C, D; Project risks: B, E
- c) Product risks: A, C, D, E Project risks: B
- d) Product risks: A, C Project risks: B, D, E

Do you have
any questions?



06 Test Tools

Tool Support for Testing

Benefits and Risks
of Test Automation



6.1 Tool Support for Testing

Test tools can be used to support one or more testing activities. Such tools include:

- Tools that are directly used in testing, such as test execution tools and test data preparation tools
- Tools that help to manage requirements, test cases, test procedures, automated test scripts, test results, test data, and defects, and for reporting and monitoring test execution
- Tools that are used for analysis and evaluation
- Any tool that assists in testing

Test tools can have one or more of the following purposes depending on the context:

- Improve the efficiency of test activities by automating repetitive tasks or tasks that require significant resources when done manually
- Improve the efficiency of test activities by supporting manual test activities throughout the test process
- Improve the quality of test activities by allowing for more consistent testing and a higher level of defect reproducibility
- Automate activities that cannot be executed manually
- Increase reliability of testing

6.1 Tool Support for Testing

Management tools

- increase the test process efficiency by facilitating management of the SDLC, requirements, tests, defects, configuration
- HP Quality Center, IBM Rational Quality Manager, and Zephyr

Static testing tools

- support the tester in performing reviews and static analysis
- Code review tools like CodeCollaborator, Crucible, and GitHub

Test design and implementation tools

- facilitate generation of test cases, test data and test procedures
- TestRail, PractiTest, and TestLink, HPQC, Azure Devops, Selenium, Appium, TestComplete

Test execution and coverage tools

- facilitate automated test execution and coverage measurement
- HP ALM, IBM Rational Quality Manager, and Zephyr

Non-functional testing tools

- allow the tester to perform non-functional testing that is difficult or impossible to perform manually
- Jmeter, LoadRunner, Jenkins

DevOps tools

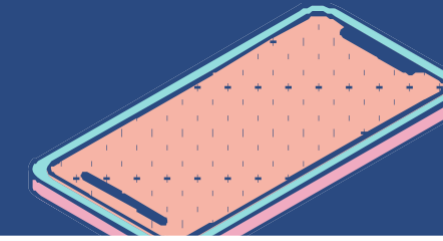
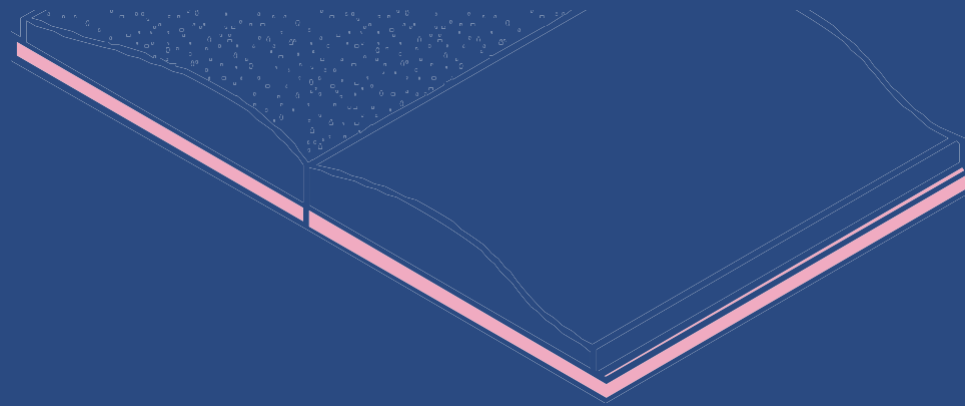
- support the DevOps delivery pipeline, workflow tracking, automated build process(es), CI/CD
- Git, Kubernetes, Azure, Git, CircleCI

Collaboration tools

- facilitate communication, Slack, MS Teams, Zoom, Webex, Project Management Tools (Trello, Asana, JIRA), MS365, Google Workspace

Tools supporting scalability and deployment standardization (e.g., virtual machines, containerization tools)

Any other tool that assists in testing (e.g., a spreadsheet is a test tool in the context of testing)



6.2 Benefits and Risks of Test Automation

Potential benefits of using tools to support test execution include:

- Reduction in repetitive manual work
- Greater consistency and repeatability
- More objective assessment
- Easier access to information about testing

Potential risks of using tools to support testing include:

- Expectations for the tool may be unrealistic
- The time, cost and effort for the initial introduction of a tool may be under-estimated
- The time and effort needed to achieve significant and continuing benefits from the tool may be under-estimated
- The effort required to maintain the test work products generated by the tool may be underestimated
- The tool may be relied on too much
- Version control of test work products may be neglected
- Relationships and interoperability issues between critical tools may be neglected
- Potential defect logs and review reports are produced
- The tool vendor may go out of business, retire the tool, or sell the tool to a different vendor
- The vendor may provide a poor response for support, upgrades, and defect fixes
- An open-source project may be suspended
- A new platform or technology may not be supported by the tool
- There may be no clear ownership of the tool

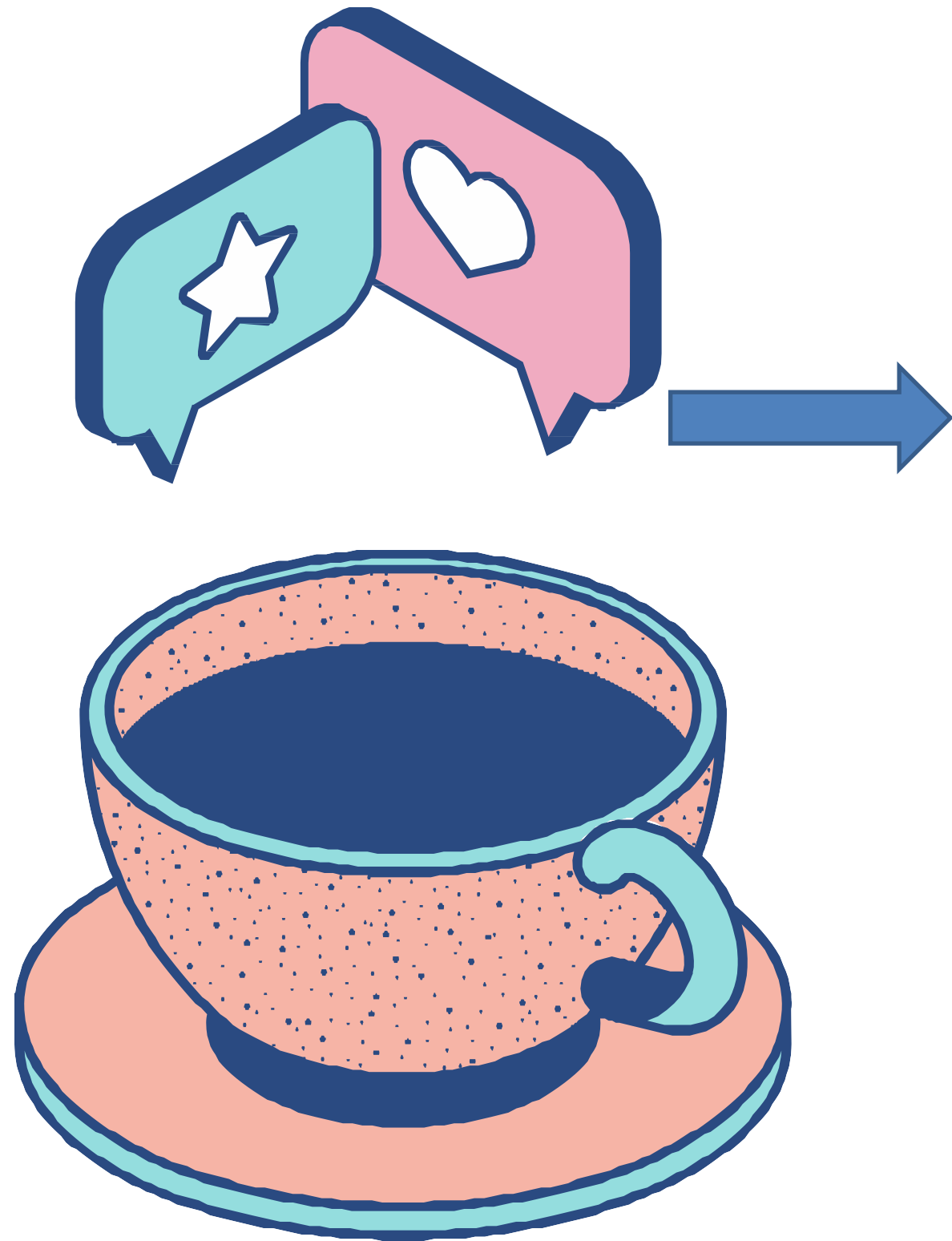
Exercise Questions





Which one of the following is *MOST* likely to be a benefit of test execution tools?

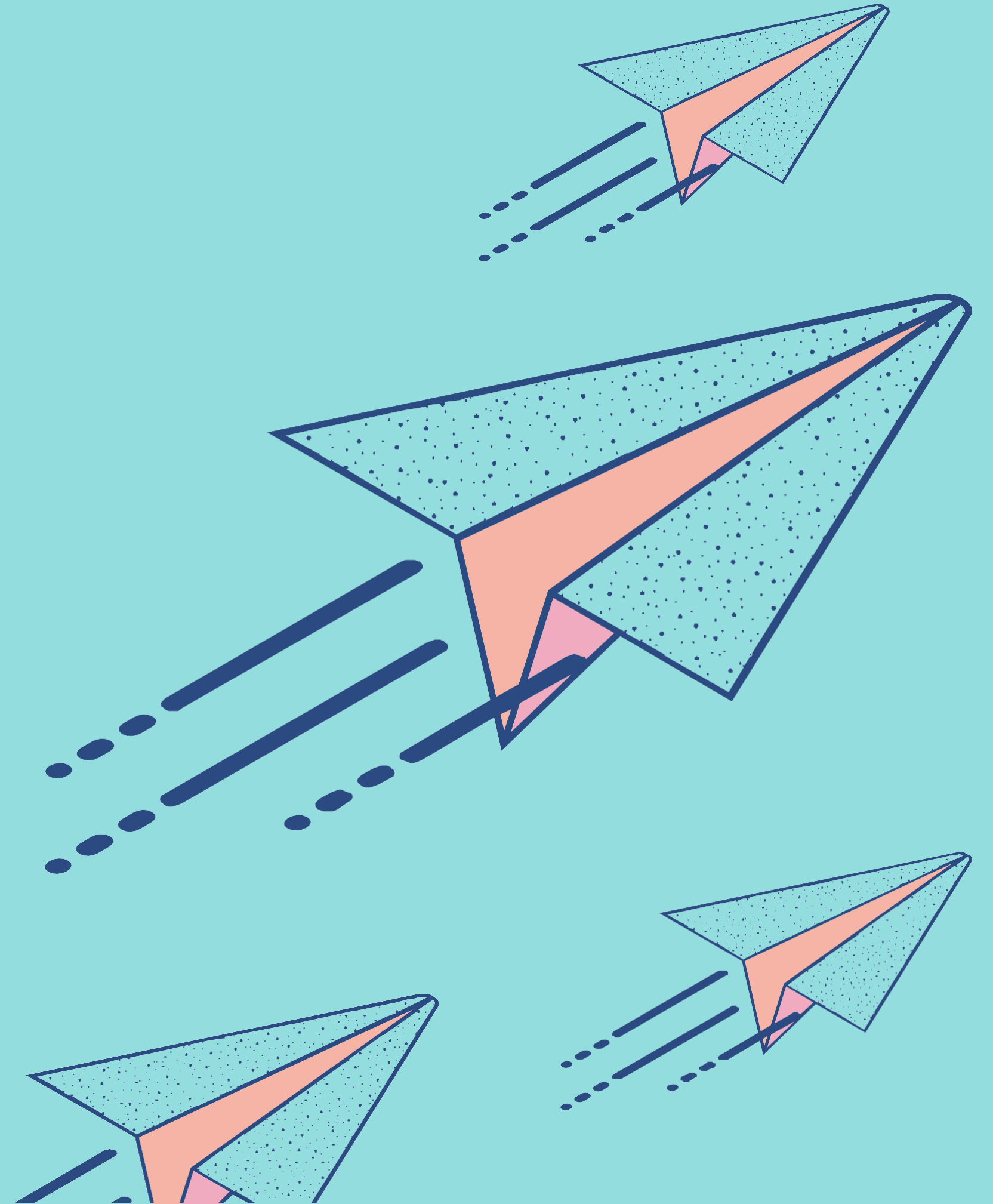
- a) It is easy to create regression tests
- b) It is easy to maintain version control of test assets
- c) It is easy to design tests for security testing
- d) It is easy to run regression tests



Which of the following tools is most useful for reporting test metrics?

- a) Test management tool
- b) Static analysis tool
- c) Coverage tool
- d) Model-Based testing tools

Do you have
any questions?



END of Day 2.
Thank you!

