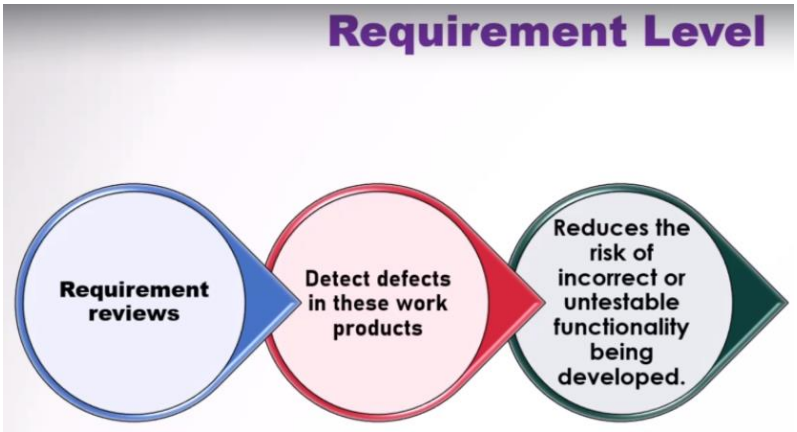
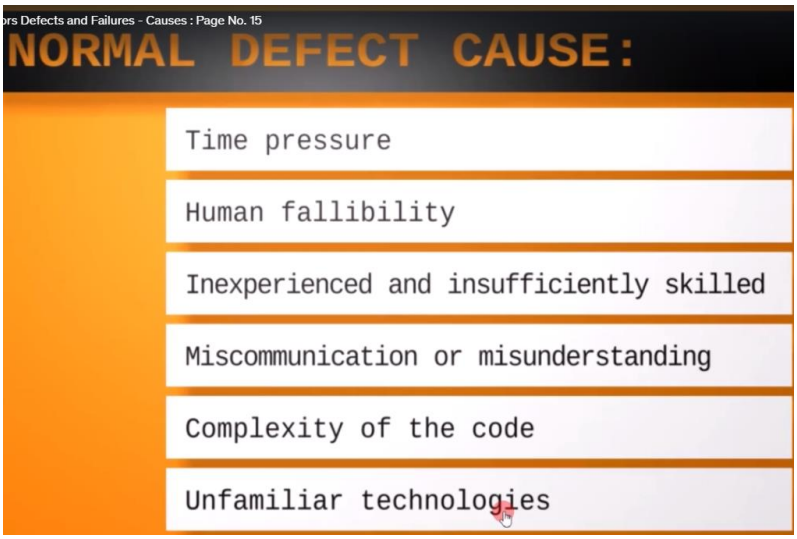
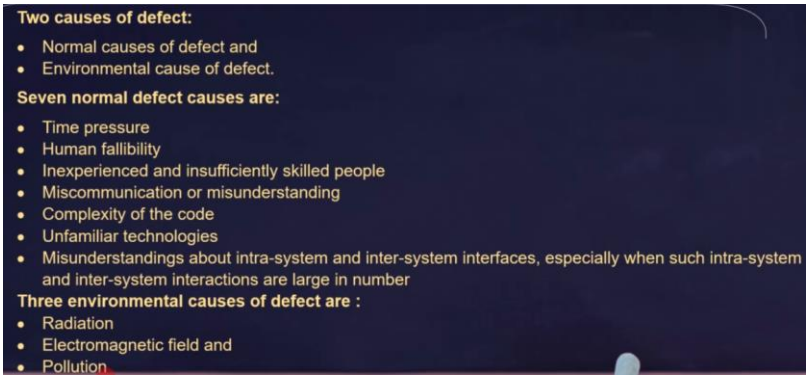
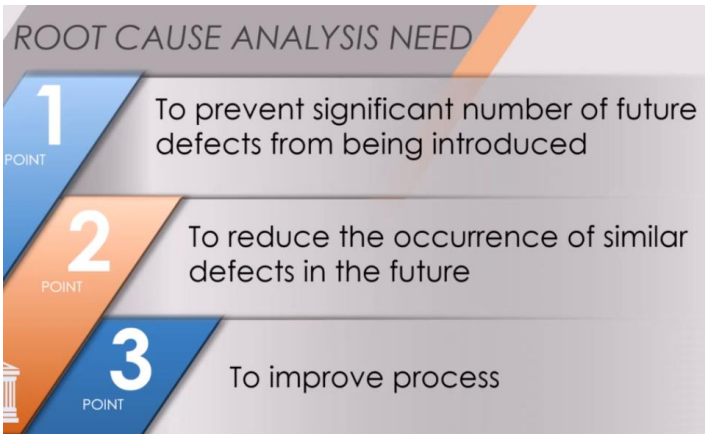

Requirement Level
Requirement reviews
Detect defects in these work products
Reduces the risk of incorrect or untestable functionality being developed.


rs Defects and Failures - Causes : Page No. 15
NORMAL DEFECT CAUSE:
Time pressure
Human fallibility
Inexperienced and insufficiently skilled
Miscommunication or misunderstanding
Complexity of the code
Unfamiliar technologies


Two causes of defect:
Normal causes of defect and
Environmental cause of defect.
Seven normal defect causes are:
Time pressure
Human fallibility
Inexperienced and insufficiently skilled people
Miscommunication or misunderstanding
Complexity of the code
Unfamiliar technologies
Misunderstandings about intra-system and inter-system interfaces, especially when such intra-system and inter-system interactions are large in number
Three environmental causes of defect are :
Radiation
Electromagnetic field and
Pollution

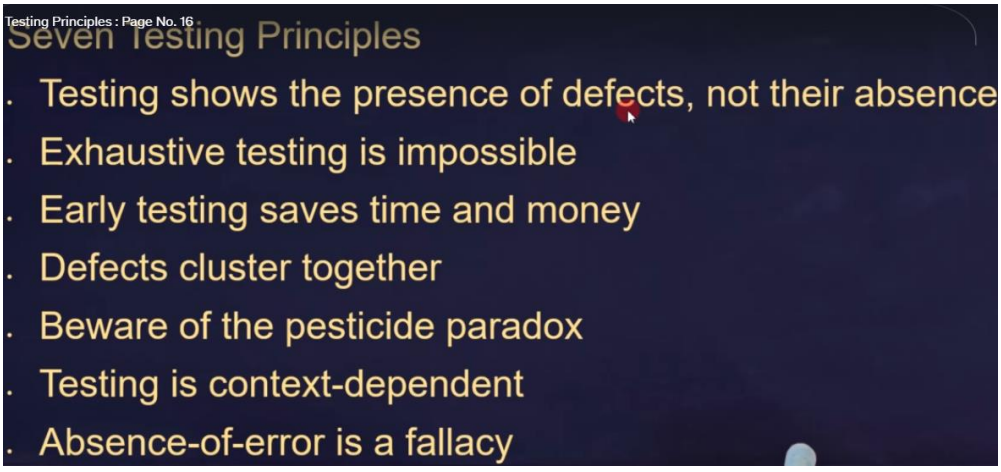


Definition of root cause,

Root cause is the earliest actions or conditions that contributed to creating the defects

Need of root cause analysis.

- To prevent a significant number of future defects from being introduced. We want to make sure that new defects don't crop up in the future
- To reduce the occurrence of similar defects in the future.
- To improve the process.



Process in Context : Page No. 17

Software development lifecycle model and project methodologies being used
Test levels and test types being considered
Product and project risks
Business domain
Operational constraints, including but not limited to:
- ❑ Budgets and resources
- ❑ Timescales
- ❑ Complexity
- ❑ Contractual and regulatory requirements

Organizational policies and practices
Required internal and external standards

TEST PROCESS

General aspects of organizational test processes

ACTIVITIES

10 20 30 40 50 60 70 80 90 100

WORK

Test activities and tasks

- Test planning
- Test monitoring and control
- Test analysis
- Test design
- Test implementation
- Test execution
- Test completion

OUTPUT

10 20 30 40 50 60 70 80 90 100

OUTPUT

Test work products

- Test plans
- Test progress reports
- Prioritized test conditions
- Test cases
- Test suites
- Defect reports
- Finalized test-ware

TRACEABILITY

10 20 30 40 50 60 70 80 90 100

RELATION

Traceability

To maintain traceability throughout the test process between each element of the test basis and the various test work products associated with that element

4_2_5 Test Analysis Activity

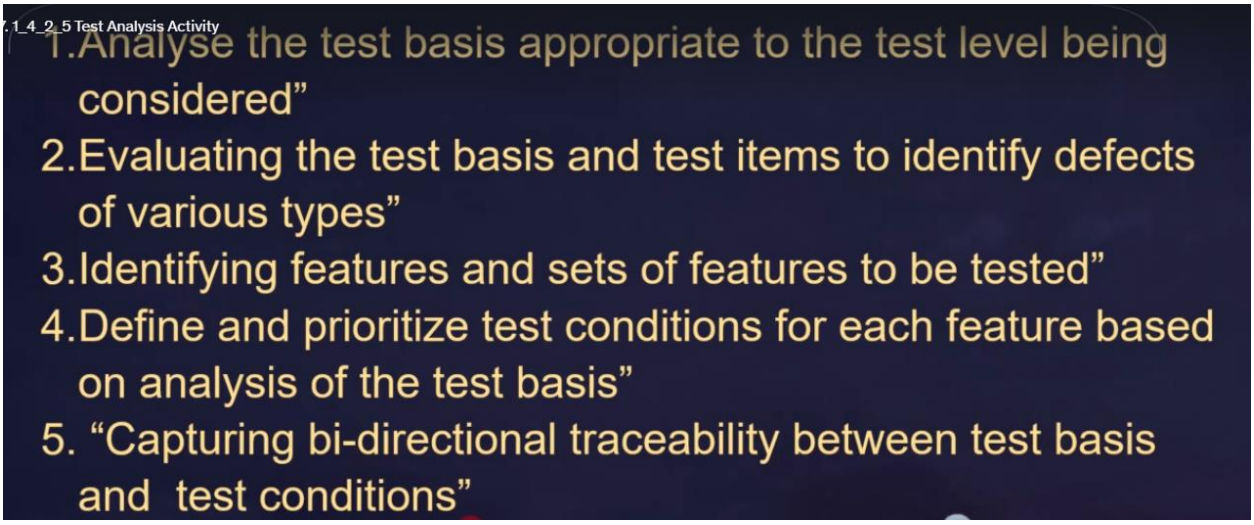
FIVE MAJOR ACTIVITIES

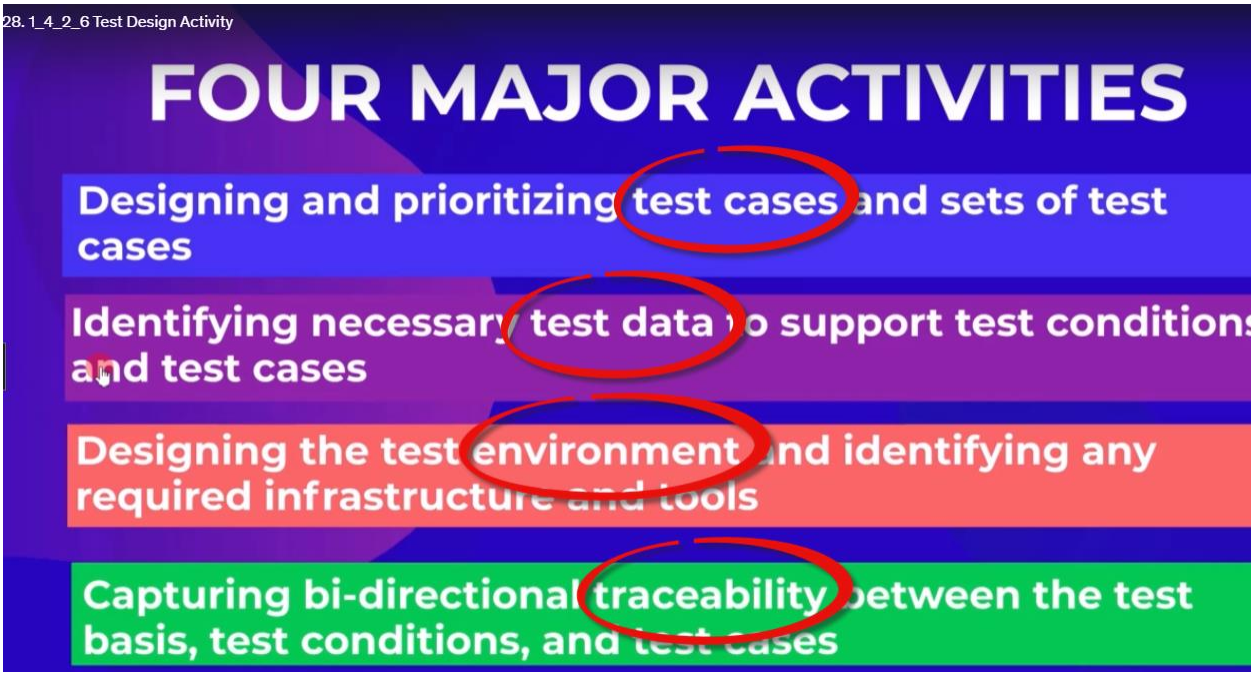
- Analyse the test basis appropriate to the test level being considered
- Evaluating the test basis and test items to identify defects of various types
- Identifying features and sets of features to be tested
- Define and prioritize test conditions for each feature based on analysis of the test basis
- Capturing bi-directional traceability between test basis and test conditions

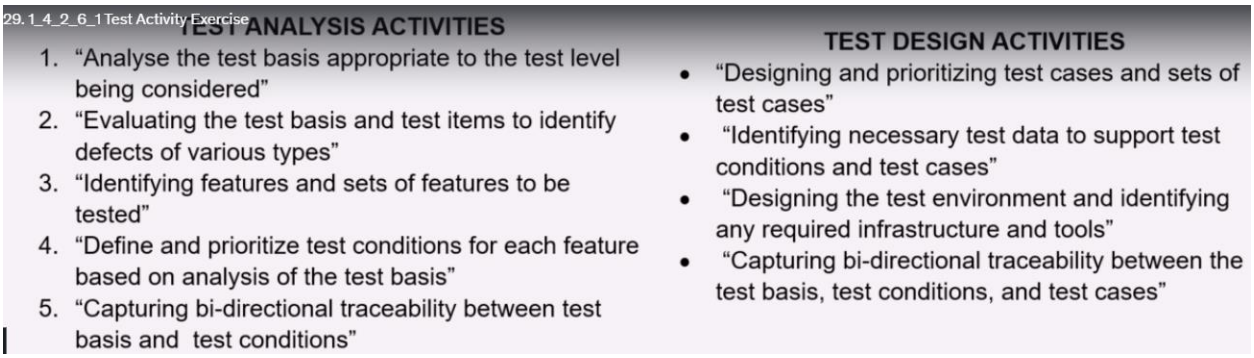
27.1_4_2_5 Test Analysis Activity

Analyse the test basis appropriate to the test level being considered

TEST LEVEL	TEST BASIS
Functional and non-functional component or system behaviour	Requirement specifications such as business requirements, functional requirements, system requirements, user stories, epics, use cases, or similar work products
Component or system structure	Design and implementation information, such as system or software architecture diagrams or documents, design specifications, call flow graphs, modelling diagrams such as UML or entity-relationship diagrams, interface specifications, or similar work products
Integration testing	The implementation of the component or system itself, including code, database metadata and queries, and interfaces
System testing	Risk analysis reports, which may consider functional, non-functional, and structural aspects of the component or system


7. 1_4_2_5 Test Analysis Activity
1.Analyse the test basis appropriate to the test level being considered"
2.Evaluating the test basis and test items to identify defects of various types"
3.Identifying features and sets of features to be tested"
4.Define and prioritize test conditions for each feature based on analysis of the test basis"
5. "Capturing bi-directional traceability between test basis and test conditions"


28. 1_4_2_6 Test Design Activity
FOUR MAJOR ACTIVITIES
Designing and prioritizing test cases and sets of test cases
Identifying necessary test data to support test conditions and test cases
Designing the test environment and identifying any required infrastructure and tools
Capturing bi-directional traceability between the test basis, test conditions, and test cases


29. 1_4_2_6_1 Test Activity Exercise
TEST ANALYSIS ACTIVITIES
1. "Analyse the test basis appropriate to the test level being considered"
2. "Evaluating the test basis and test items to identify defects of various types"
3. "Identifying features and sets of features to be tested"
4. "Define and prioritize test conditions for each feature based on analysis of the test basis"
5. "Capturing bi-directional traceability between test basis and test conditions"
TEST DESIGN ACTIVITIES
• "Designing and prioritizing test cases and sets of test cases"
• "Identifying necessary test data to support test conditions and test cases"
• "Designing the test environment and identifying any required infrastructure and tools"
• "Capturing bi-directional traceability between the test basis, test conditions, and test cases"

30. 1_4_2_7 Test implementation Activity

SIX MAJOR ACTIVITIES

1. Developing and prioritizing test procedures and, potentially, creating automated test scripts
2. Creating test suites from the test procedures and (if any) automated test scripts
3. Arranging the test suites within a test execution schedule in a way that results in efficient test execution
4. Building the test environment (including, potentially, test harnesses, service virtualization, simulators, and other infrastructure items) and verifying that everything needed has been set up correctly
5. Preparing test data and ensuring it is properly loaded in the test environment
6. Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test suites

31. 1_4_2_7_1 Test Activity Exercise

Test activities	ANALYSIS	DESIGN	IMPLEMENTATION
Developing test script			✓
Prioritized test conditions	✓		
Prioritizing test cases		✓	
Prioritizing test procedures			✓
Building the test environment			✓
Identifying necessary test data to support test conditions		✓	
Designing the test environment		✓	
Identifying features and sets of features to be tested	✓		
Preparing test data and ensuring it is properly loaded in the test environment			✓
What to test ?	✓		
Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test suites			✓

31. 1_4_2_7_1 Test Activity Exercise

TEST ANALYSIS ACTIVITIES

1. "Analyse the test basis appropriate to the test level being considered"
2. "Evaluating the test basis and test items to identify defects of various types"
3. "Identifying features and sets of features to be tested"
4. "Define and prioritize test conditions for each feature based on analysis of the test basis"
5. "Capturing bi-directional traceability between test basis and test conditions"

TEST DESIGN ACTIVITIES

1. "Designing and prioritizing test cases and sets of test cases"
2. "Identifying necessary test data to support test conditions and test cases"
3. "Designing the test environment and identifying any required infrastructure and tools"
4. "Capturing bi-directional traceability between the test basis, test conditions, and test cases"

TEST IMPLEMENTATION ACTIVITIES

1. "Developing and prioritizing test procedures, and, potentially, creating automated test scripts "
2. "Creating test suites from the test procedures and (if any) automated test scripts"
3. "Arranging the test suites within a test execution schedule in a way that results in efficient test execution"
4. "Building the test environment (including, potentially, test harnesses, service virtualization, simulators, and other infrastructure items) and verifying that everything needed has been set up correctly
5. "Preparing test data and ensuring it is properly loaded in the test environment
6. "Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test suites

32. 1_4_2_8 Test Execution Activity

1. Recording the IDs and versions of the test item(s) or test object, test tool(s), and testware
2. Executing tests either manually or by using test execution tools
3. Comparing actual results with expected results
4. Analyzing anomalies to establish their likely causes
5. Reporting defects based on the failures observed
6. Logging the outcome of test execution (e.g., pass, fail, blocked)
7. Repeating test activities either as a result of action taken for an anomaly, or as part of the planned testing
8. Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test results

Test activities	IMPLEMENTATION	EXECUTION
Developing test procedures, and, potentially, creating automated test scripts	✓	
Logging the outcome of test execution		✓
Executing tests manually		✓
Creating test suites from the test procedures	✓	
Repeating test activities either as a result of action taken for an anomaly		✓
Prioritizing test procedures	✓	
Preparing test data	✓	
Comparing actual results with expected results		✓
Ensuring it is properly loaded in the test environment	✓	
Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test suites	✓	
Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test results		✓

TEST ANALYSIS ACTIVITIES

1. "Analyse the test basis appropriate to the test level being considered"
2. "Evaluating the test basis and test items to identify defects of various types"
3. "Identifying features and sets of features to be tested"
4. "Define and prioritize test conditions for each feature based on analysis of the test basis"
5. "Capturing bi-directional traceability between test basis and test conditions"

TEST IMPLEMENTATION ACTIVITIES

1. "Developing and prioritizing test procedures, and, potentially, creating automated test scripts "
2. "Creating test suites from the test procedures and (if any) automated test scripts"
3. "Arranging the test suites within a test execution schedule in a way that results in efficient test execution"
4. "Building the test environment (including, potentially, test harnesses, service virtualization, simulators, and other infrastructure items) and verifying that everything needed has been set up correctly
5. "Preparing test data and ensuring it is properly loaded in the test environment
6. "Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test suites

TEST DESIGN ACTIVITIES

1. "Designing and prioritizing test cases and sets of test cases"
2. "Identifying necessary test data to support test conditions and test cases"
3. "Designing the test environment and identifying any required infrastructure and tools"
4. "Capturing bi-directional traceability between the test basis, test conditions, and test cases"

TEST EXECUTION ACTIVITIES

1. Recording the IDs and versions of the test item(s) or test object, test tool and testware
2. Executing tests either manually or by using test execution tools
3. Comparing actual results with expected results
4. Analyzing anomalies to establish their likely causes
5. Reporting defects based on the failures observed
6. Logging the outcome of test execution (e.g., pass, fail, blocked)
7. Repeating test activities either as a result of action taken for an anomaly, or as part of the planned testing
8. Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test results

34. 1_4_2_9 Test completion Activity

1. Checking whether all defect reports are closed,
2. Creating a test summary report to be communicated to stakeholders
3. Finalizing and archiving the test environment, the test data, the test infrastructure, and other testware for later reuse
4. Handing over the testware to the maintenance teams, other project teams, and/or other stakeholders who could benefit from its use
5. Analyzing lessons learned from the completed test activities to determine changes needed for future iterations, releases, and projects
6. Using the information gathered to improve test process maturity

35. 1_4_3 Test Work Products : Page No. 22

TEST WORKPRODUCT

ANALYSIS

- Defined and prioritized test conditions
- Creation of test charters
- Defects in the test basis.

DESIGN

- Test cases and sets of test cases.
- Design and/or identification of the test data
- Design of the test environment
- Identification of infrastructure and tools

IMPLEMENTATION

- Test procedures and the sequencing of those test procedures
- Test suites
- A test execution schedule
- Service virtualization and automated test scripts.
- Creation and verification of test data and the test environment.
- Test conditions defined in test analysis may be further refined in test implementation.

EXECUTION

- Documentation of the status of individual test cases or test procedures (e.g., ready to run, pass, fail, blocked, deliberately skipped, etc.)
- Defect reports
- Documentation about which test item(s), test object(s), test tools, and testware were involved in the testing

COMPLETION

- Test summary reports
- Action items for improvement of subsequent projects or iterations
- Change requests or product backlog items, and
- Finalized testware.

Work product	ANALYSIS	DESIGN	IMPLEMENTATION	EXECUTION
Test cases and sets of test cases		✓		
Prioritized test conditions	✓			
Defect reports				✓
Test suites			✓	
Test procedures			✓	
Test execution schedule			✓	
Test base and test case traceability		✓		
Defects in the test basis	✓			
Documentation of the status of individual test cases or test procedures				✓
Identified test data		✓		
Traceability between test procedures and the test basis			✓	

Work Product	ANALYSIS	DESIGN	IMPLEMENTATION	EXECUTION
Results like pass, fail, blocked, deliberately skipped,				✓
Work product creation by service virtualization			✓	
Creation of test charters	✓			
Turns high-level test cases into executable low-level test cases.			✓	
Traceability between the test basis and test results.				✓
Test basis and test condition traceability	✓			
Test condition further refined in which stage			✓	
Identification of infrastructure(test environment) and tools		✓		
Actual and expected result comparision				✓

"Traceability assists with: •Selecting regression tests in terms of analyzing the impact of changes. •Evaluating completeness of test execution which makes testing auditable. •Identifying which user stories have open defect reports which improves understandability of test status reports to include status of test basis items. •Evaluating whether the number of tests for each requirement is consistent with the level of product risk which provides information to assess test process quality (i.e., alignment of test effort with risk). Therefore, dis correct, per section 1.4.4."

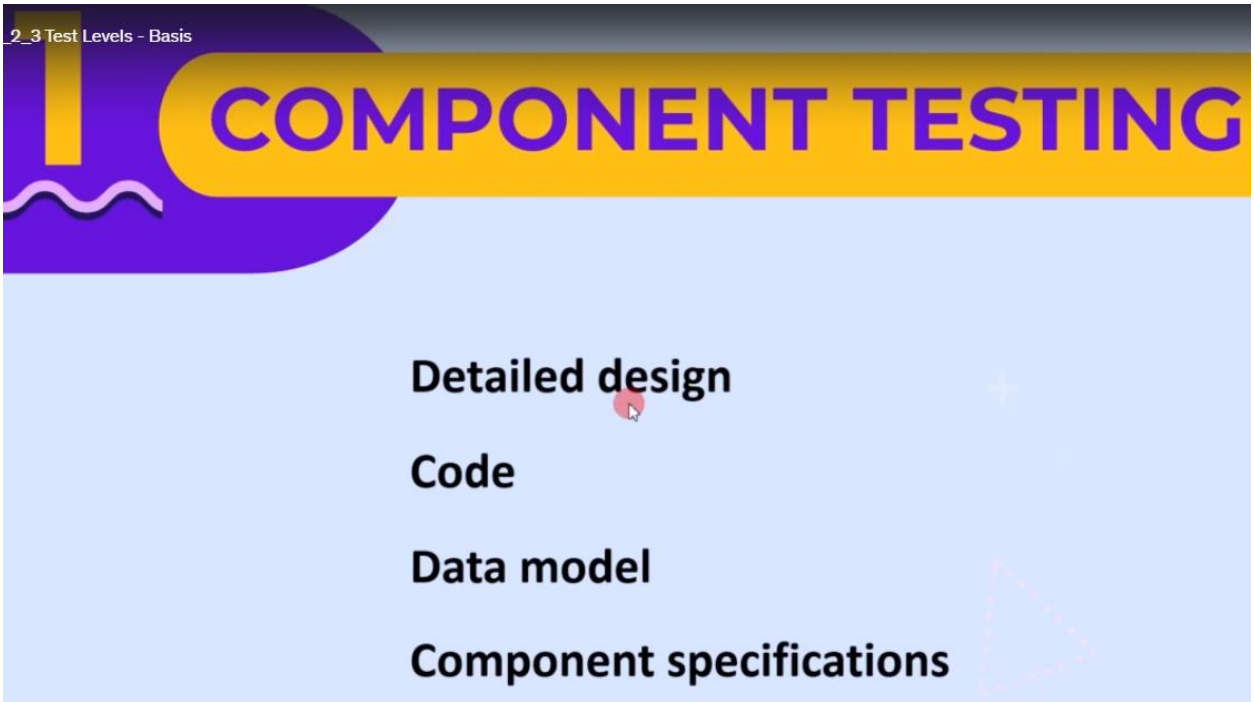
"Traceability assists with: •Selecting regression tests in terms of analyzing the impact of changes. •Evaluating completeness of test execution which makes testing auditable. •Identifying which user stories have open defect reports which improves understandability of test status reports to include status of test basis items. •Evaluating whether the number of tests for each requirement is consistent with the level of product riskwhich provides information to assess test process quality (i.e., alignment of test effort with risk).Therefore, dis correct, per section 1.4.4."

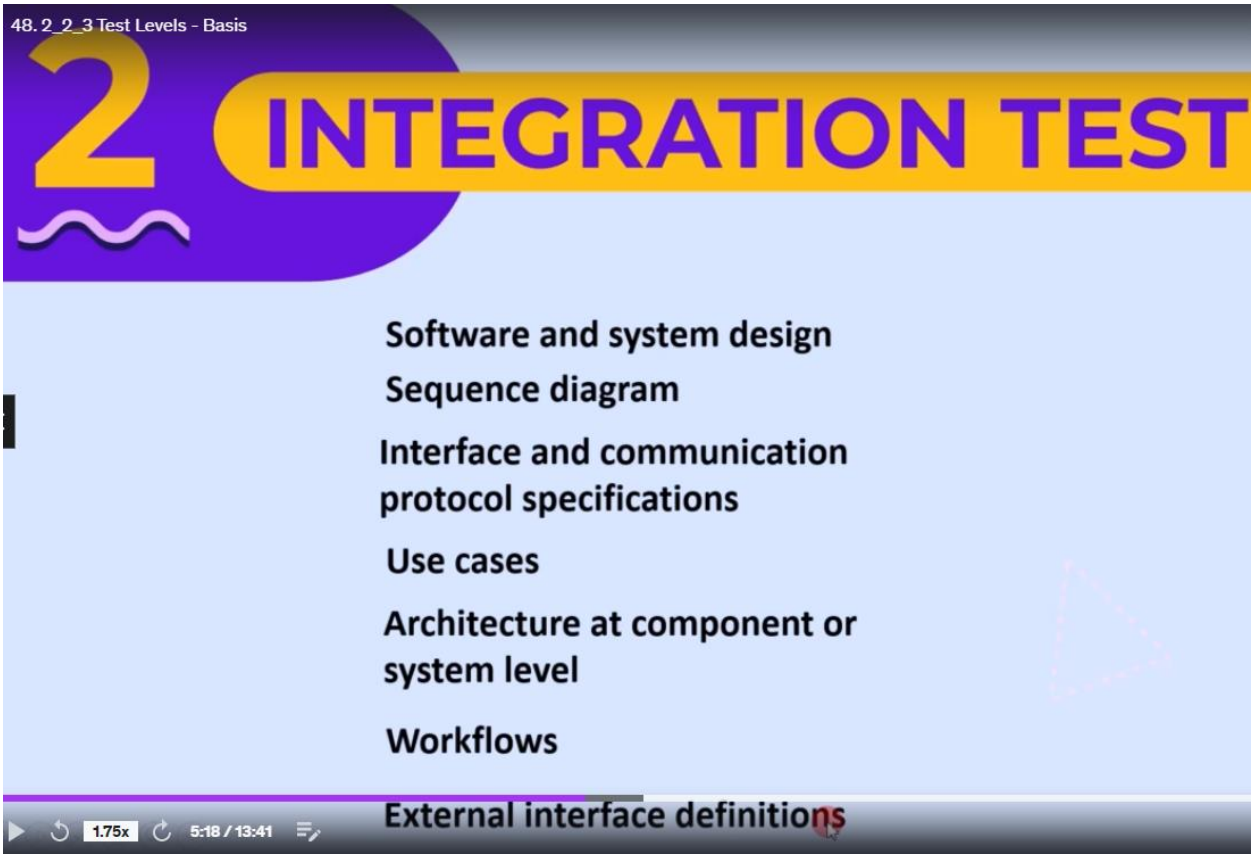
Soru 29:

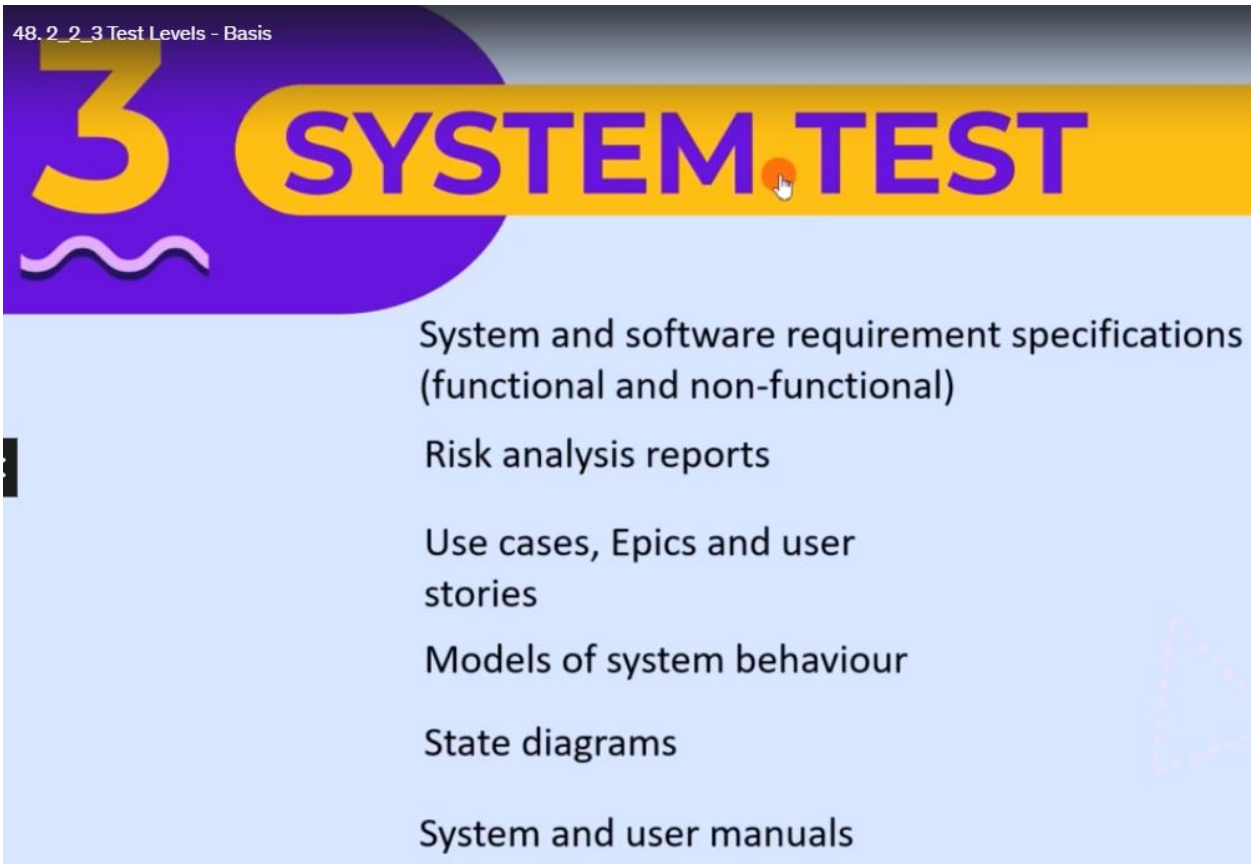
Consider the following testing activities:
1.Selecting regression tests
2.Evaluating completeness of test execution
3.Identifying which user stories have open defect reports
4.Evaluating whether the number of tests for each requirement is consistent with the level of product risk
Consider the following ways traceability can help testing:
A. Improve understandability of test status reports to include status of test basis items
B. Make testing auditable
C. Provide information to assess process quality
D. Analyze the impact of changes

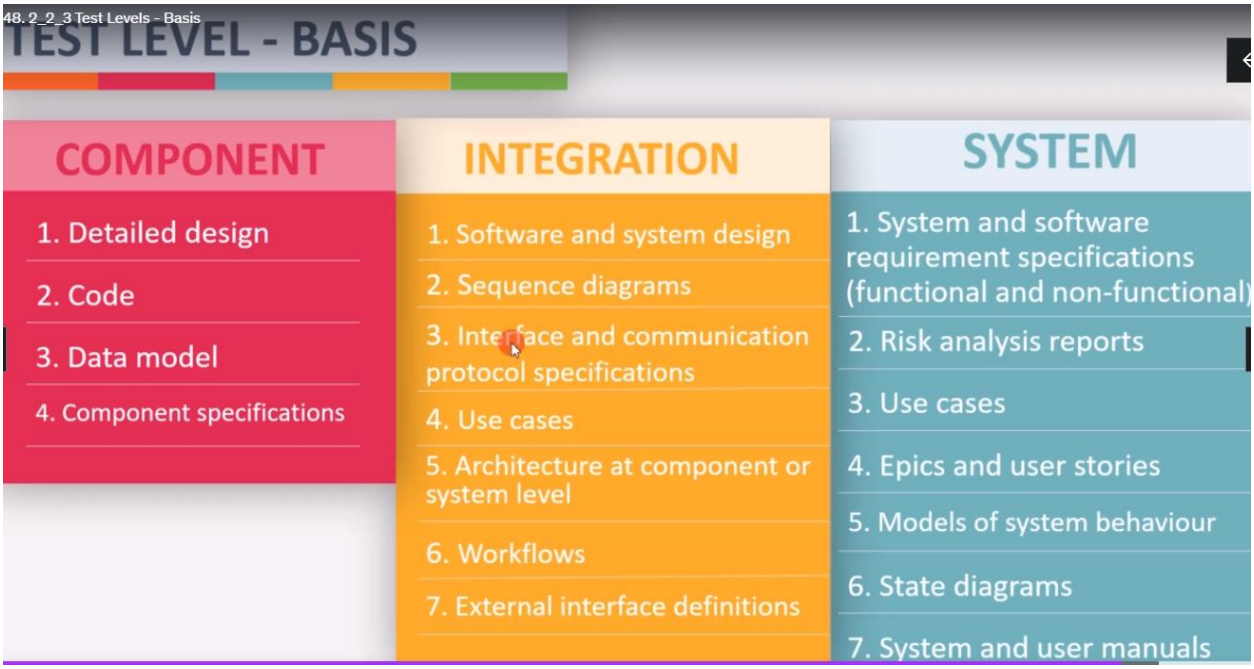
Which of the following best matches the testing activity with how traceability can assist that activity?

- ○ 1D,2B,3C,4A
- ○ 1B,2D,3A,4C
- ○ 1D,2C,3A, 4B
- ◉ **1D,2B,3A,4C**




48. 2_2_3 Test Levels - Basis
2
INTEGRATION TEST
Software and system design
Sequence diagram
Interface and communication protocol specifications
Use cases
Architecture at component or system level
Workflows
External interface definitions
1.75x
5:18 / 13:41


48. 2_2_3 Test Levels - Basis
3
SYSTEM TEST
System and software requirement specifications (functional and non-functional)
Risk analysis reports
Use cases, Epics and user stories
Models of system behaviour
State diagrams
System and user manuals


48. 2_2_3 Test Levels - Basis
TEST LEVEL - BASIS
COMPONENT
1. Detailed design
2. Code
3. Data model
4. Component specifications
INTEGRATION
1. Software and system design
2. Sequence diagrams
3. Interface and communication protocol specifications
4. Use cases
5. Architecture at component or system level
6. Workflows
7. External interface definitions
SYSTEM
1. System and software requirement specifications (functional and non-functional)
2. Risk analysis reports
3. Use cases
4. Epics and user stories
5. Models of system behaviour
6. State diagrams
7. System and user manuals

48. 2_2_3 Test Levels - Basis

ACCEPTANCE LEVEL – TEST BASIS

ACCEPTANCE	TESTING
1. Business processes	1. Backup and restore procedures
2. User or business requirements	2. Disaster recovery procedures
3. Regulations, legal contracts and Security standards	3. Non-functional requirements
4. Use cases and/or user stories	4. Operations documentation
5. System requirements	5. Deployment and installation instructions
6. System or user documentation	6. Performance targets
7. Risk analysis reports	7. Database packages

49. 2_2_3_1 Test Levels - Basis - Activity

TEST BASIS	COMPONENT	INTIGRATION	SYSTEM	ACCEPTANCE
System and user manuals			✓	
Sequence diagrams		✓		
Detailed design	✓			
Use cases		✓	✓	
Code	✓			
Operations documentation				✓
Architecture at component or system level		✓		
System or user documentation				✓
software requirement specifications			✓	
Component specifications	✓			
Performance targets				✓

49. 2_2_3_1 Test Levels - Basis - Activity

TEST BASIS	COMPONENT	INTIGRATION	SYSTEM	ACCEPTANCE
Regulations, legal contracts and Security standards				✓
communication protocol specifications		✓		
User or business requirements				✓
Deployment and installation instructions				✓
Risk analysis reports			✓	✓
External interface definitions		✓		
Data model	✓			
Software and system design		✓		
Database packages				✓
Models of system behaviour			✓	
Disaster recovery procedures				✓

50. 2_2_4 Test Levels - Object

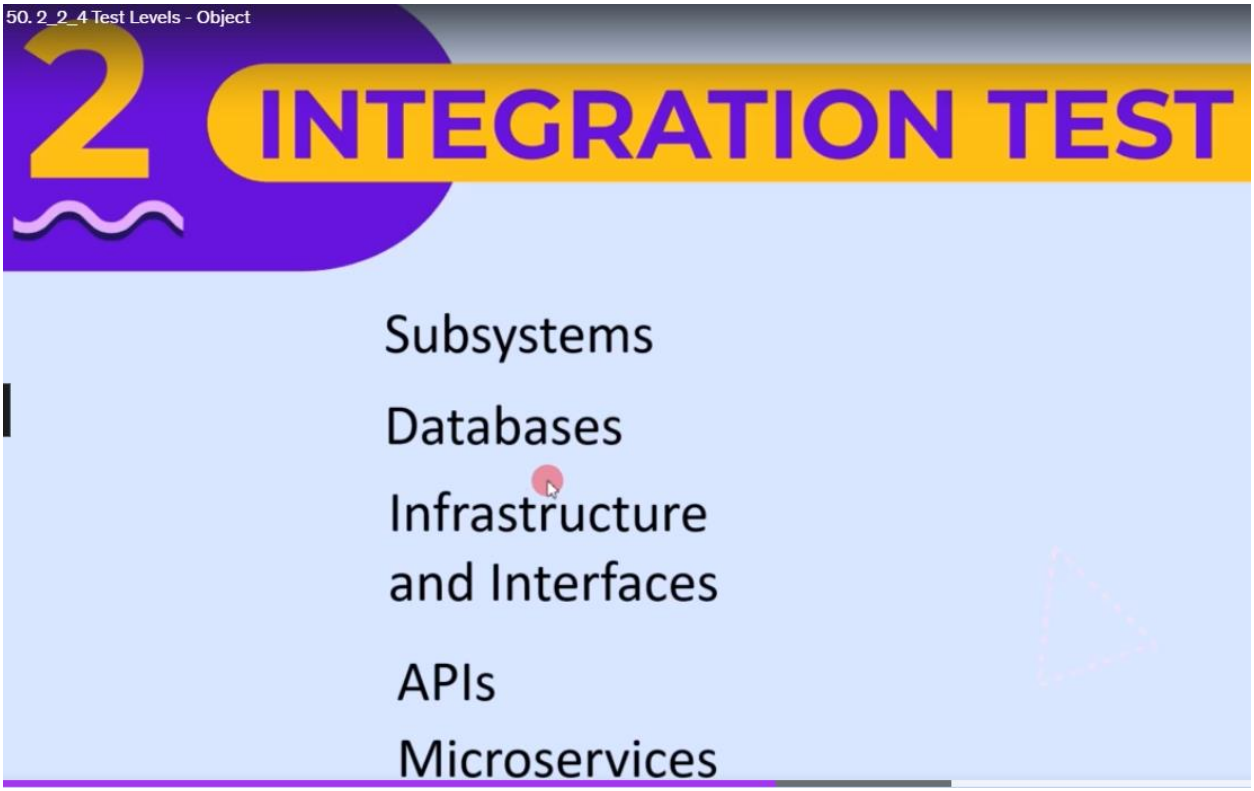
COMPONENT TESTING

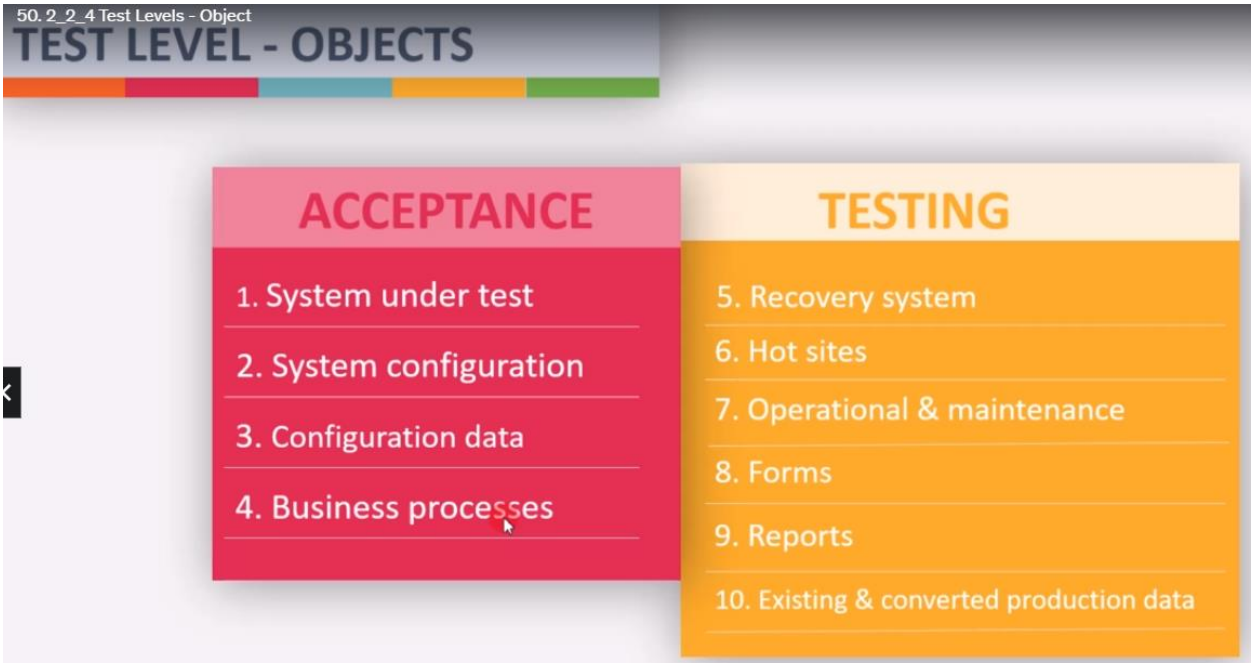
Components, units,
Modules and Code

Data structures

Classes

Database modules


50. 2_2_4 Test Levels - Object
2
INTEGRATION TEST
Subsystems
Databases
Infrastructure
and Interfaces
APIs
Microservices


50. 2_2_4 Test Levels - Object
TEST LEVEL - OBJECTS
ACCEPTANCE
1. System under test
2. System configuration
3. Configuration data
4. Business processes
TESTING
5. Recovery system
6. Hot sites
7. Operational & maintenance
8. Forms
9. Reports
10. Existing & converted production data

50. 2_2_4 Test Levels - Object

TEST LEVEL - OBJECTS

COMPONENT	INTEGRATION	SYSTEM
1. Components, units or modules	1. Subsystems	1. Applications
2. Code and data structures	2. Databases	2. Hardware/software systems
3. Classes	3. Infrastructure	3. Operating systems
4. Database modules	4. Interfaces	4. System under test (SUT)
	5. APIs	5. System configuration
	6. Microservices	6. Configuration data

51. 2_2_4_1 Test Levels - Object Activity

TEST OBJECTs	COMPONENT	INTIGRATION	SYSTEM	ACCEPTANC
Hardware/software systems			✓	
APIs		✓		
Database modules	✓			
System configuration & Configuration data			✓	✓
Code and data structures	✓			
Forms and Reports				✓
Infrastructure		✓		
Business processes				✓
Operating systems			✓	
Classes	✓			
Operational & maintenance				✓

51. 2_2_4_1 Test Levels - Object Activity

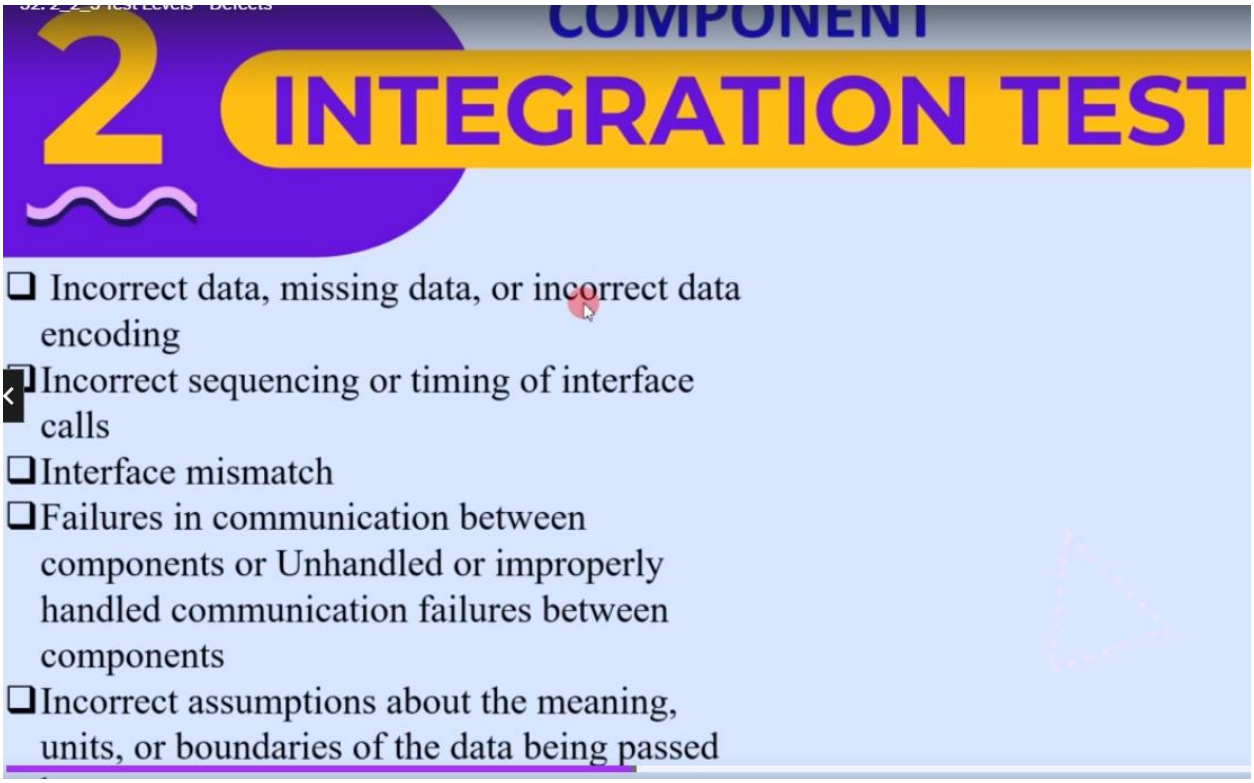
TEST OBJECTs	COMPONENT	INTIGRATION	SYSTEM	ACCEPTANCE
Recovery system				✓
Databases		✓		
Hot sites				✓
Applications			✓	
Microservices		✓		
units or modules	✓			
Existing & converted production data				✓
Subsystems		✓		
Database packages				✓

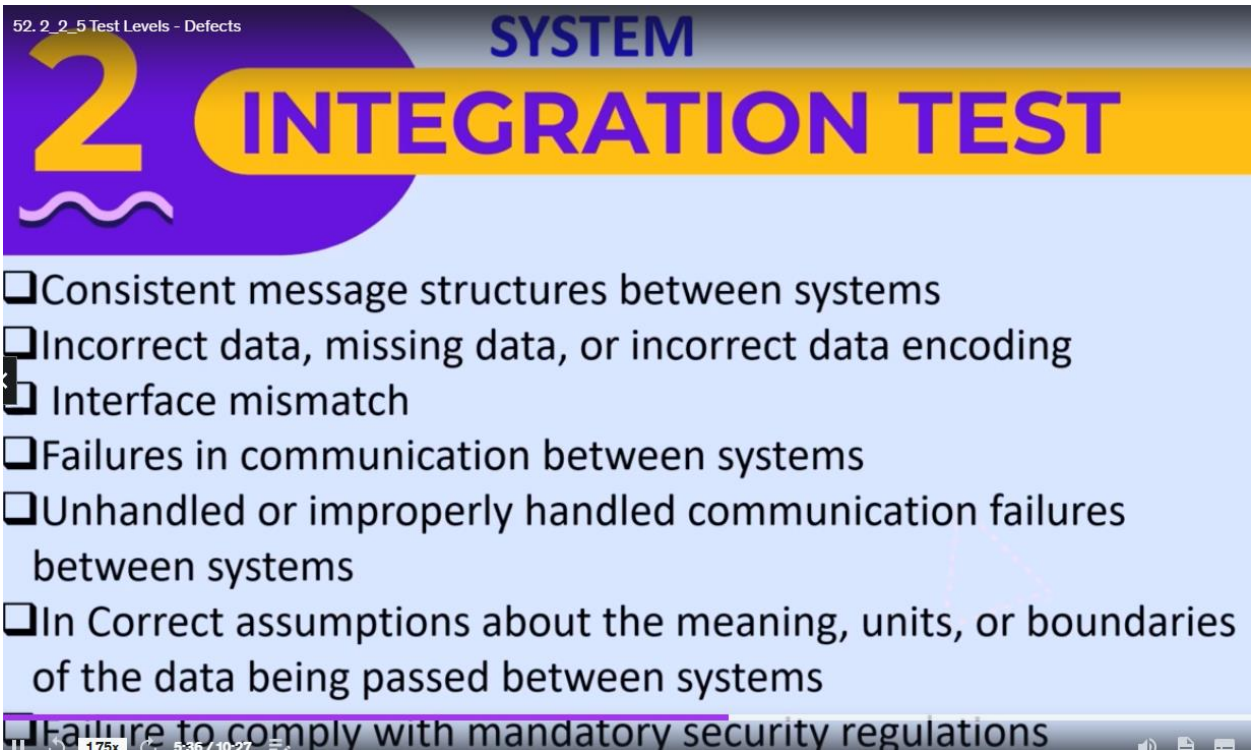
52. 2_2_5 Test Levels - Defects

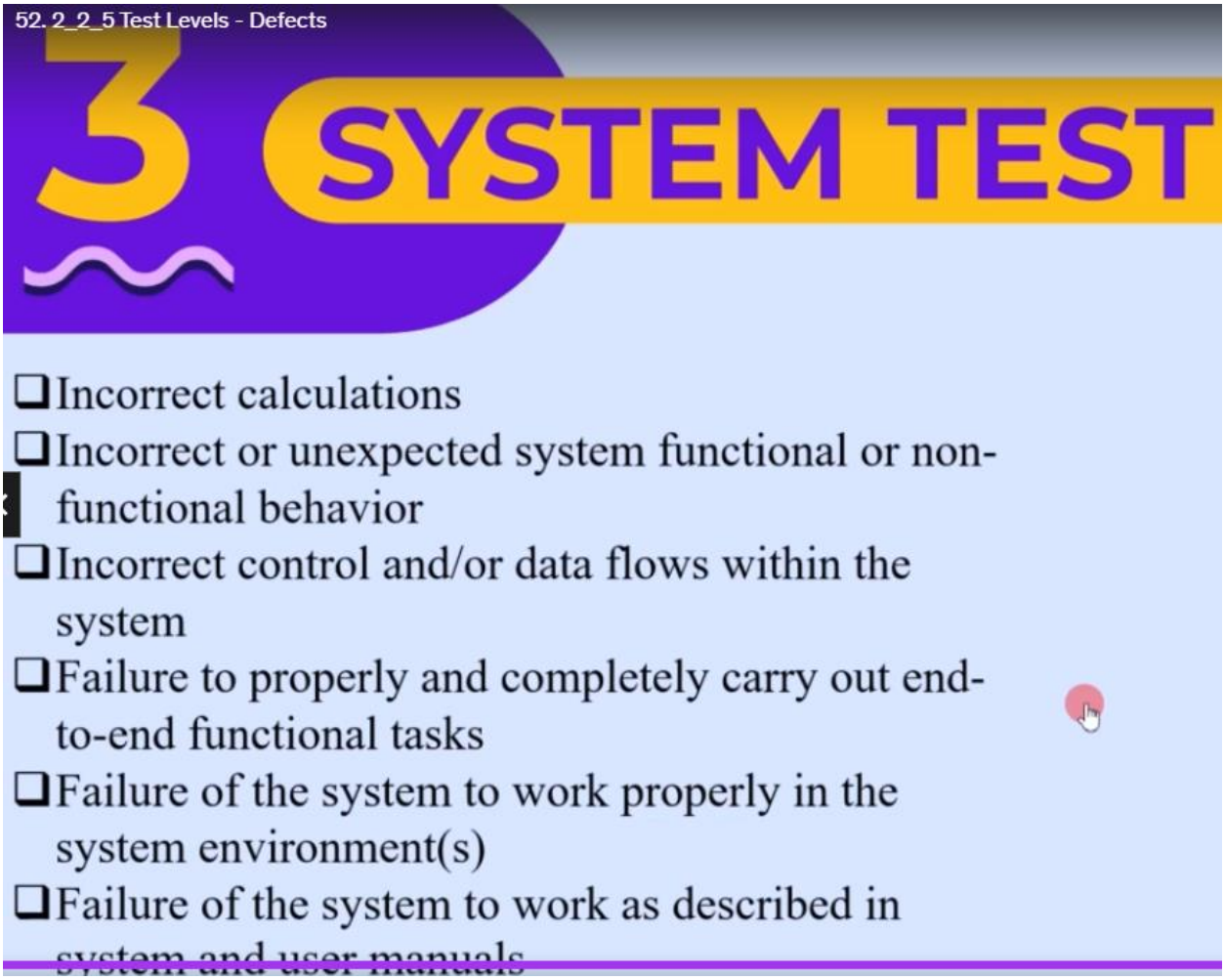
COMPONENT TESTING

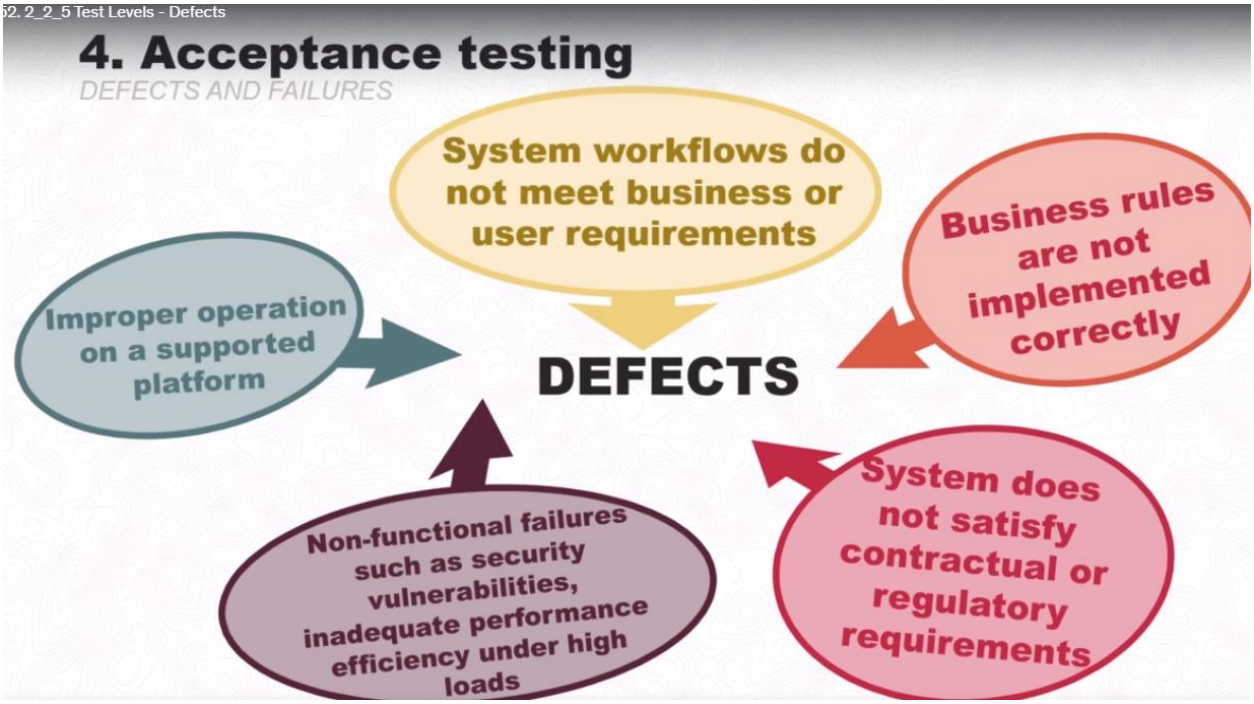
- ❑ Incorrect functionality
- ❑ Data flow problems
- ❑ Incorrect code and logic

```
if ((temperature < 0) or
(temperature > 100)) {
alert ("DANGER");
if ((speed > 100) and (load <=
speed = 50;
}
} else {
check = false;
```


COMPONENT
2
INTEGRATION TEST
Incorrect data, missing data, or incorrect data encoding
Incorrect sequencing or timing of interface calls
Interface mismatch
Failures in communication between components or Unhandled or improperly handled communication failures between components
Incorrect assumptions about the meaning, units, or boundaries of the data being passed


52. 2_2_5 Test Levels - Defects
SYSTEM
2
INTEGRATION TEST
Consistent message structures between systems
Incorrect data, missing data, or incorrect data encoding
Interface mismatch
Failures in communication between systems
Unhandled or improperly handled communication failures between systems
In Correct assumptions about the meaning, units, or boundaries of the data being passed between systems
Failure to comply with mandatory security regulations


52. 2_2_5 Test Levels - Defects
3
SYSTEM TEST
Incorrect calculations
Incorrect or unexpected system functional or non-functional behavior
Incorrect control and/or data flows within the system
Failure to properly and completely carry out end-to-end functional tasks
Failure of the system to work properly in the system environment(s)
Failure of the system to work as described in system and user manuals

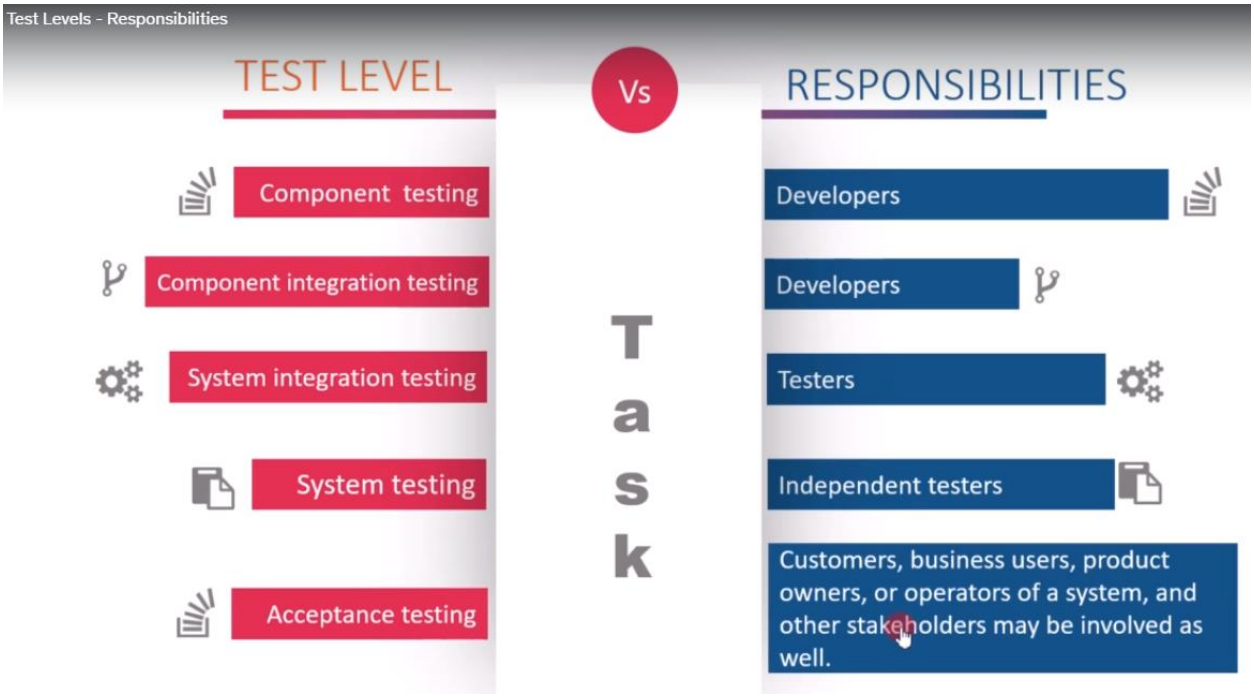

52. 2_2_5 Test Levels - Defects
4. Acceptance testing
DEFECTS AND FAILURES
System workflows do not meet business or user requirements
Business rules are not implemented correctly
Improper operation on a supported platform
DEFECTS
Non-functional failures such as security vulnerabilities, inadequate performance efficiency under high loads
System does not satisfy contractual or regulatory requirements

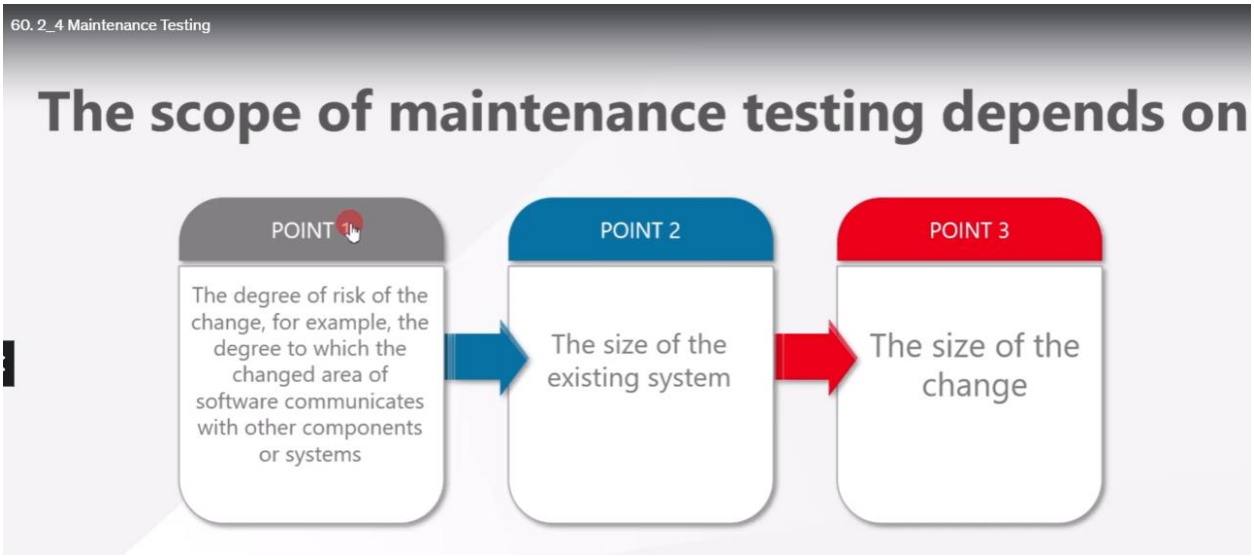
53. 2_2_5_1 Test Levels - Defects Activity

DEFECTS	COMPONENT	INTIGRATION	SYSTEM	ACCEPTANCE
Incorrect or unexpected system functional or non-functional behavior			✓	
Incorrect sequencing or timing of interface calls		✓		
Incorrect control and/or data flows within the system			✓	
System workflows do not meet business or user requirements				✓
Unhandled or improperly handled communication failures between components		✓		
Data flow problems	✓			
Failure of the system to work properly in the system environment			✓	

53. 2_2_5_1 Test Levels - Defects Activity

DEFECTS	COMPONENT	INTIGRATION	SYSTEM	ACCEPTANCE
Incorrect data, missing data, or incorrect data encoding		✓		
Business rules are not implemented correctly				✓
Incorrect functionality	✓			
Incorrect calculations			✓	
Improper operation on a supported platform				✓
Interface mismatch		✓		
Failure of the system to work as described in system and user manuals			✓	


Test Levels - Responsibilities
TEST LEVEL
Vs
RESPONSIBILITIES
Component testing
Component integration testing
System integration testing
System testing
Acceptance testing
Task
Developers
Developers
Testers
Independent testers
Customers, business users, product owners, or operators of a system, and other stakeholders may be involved as well.


60. 2_4 Maintenance Testing
The scope of maintenance testing depends on
POINT 1
The degree of risk of the change, for example, the degree to which the changed area of software communicates with other components or systems
POINT 2
The size of the existing system
POINT 3
The size of the change

43. 2_1_2 SDLM in Context

SDLM selection factors

- Product characteristic
- Project goal or type
- Business properties
- Time to market
- Project context and
- Project and product risk

To achieve project context or project goal we can combine

- Combine test level and test activities, or
- Different software development life cycle model

66. 3_1_1 Work Products Static Testing

WORK PRODUCT

- Specifications, including business requirements, functional requirements, and security requirements
- Epics, user stories, and acceptance criteria
- Architecture and design specifications
- Code
- Testware, including test plans, test cases, test procedures, and automated test scripts
- User guides
- Web pages
- Contracts, project plans, schedules, and budget planning
- Configuration set up and infrastructure set up
- Models, such as activity diagrams,

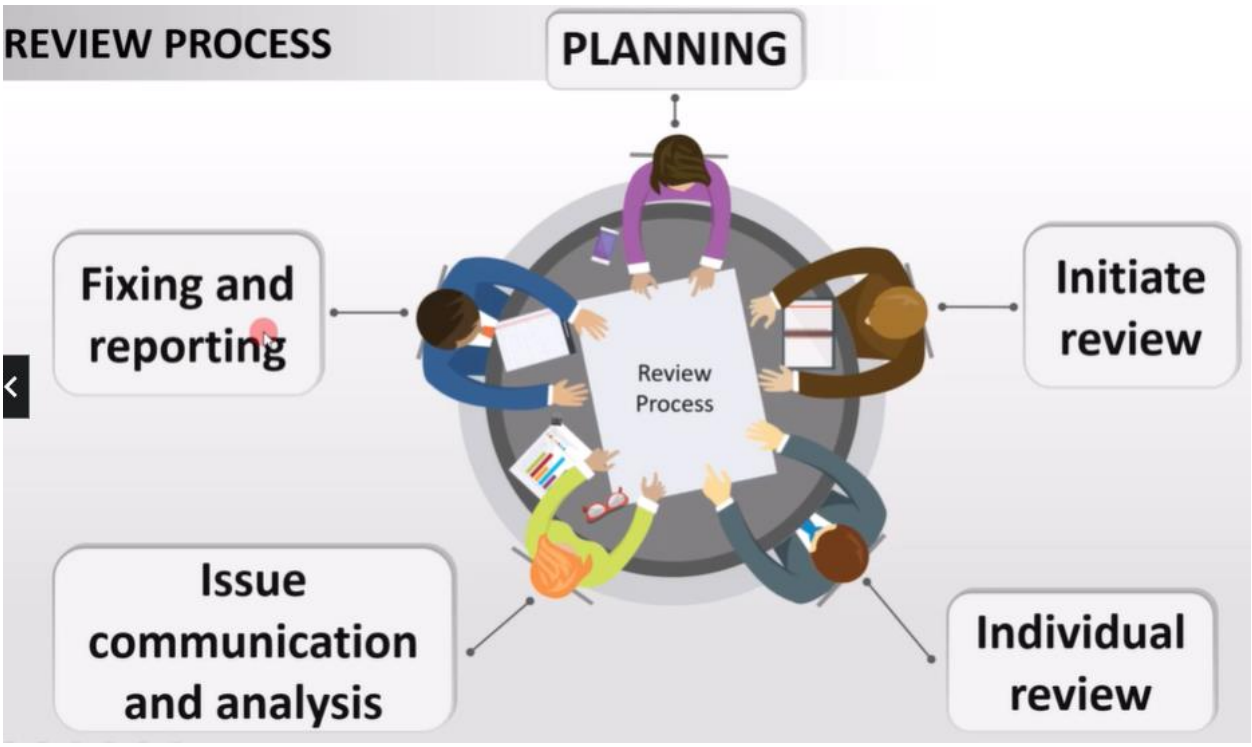
67. 3_1_2 Benefits of Static Testing

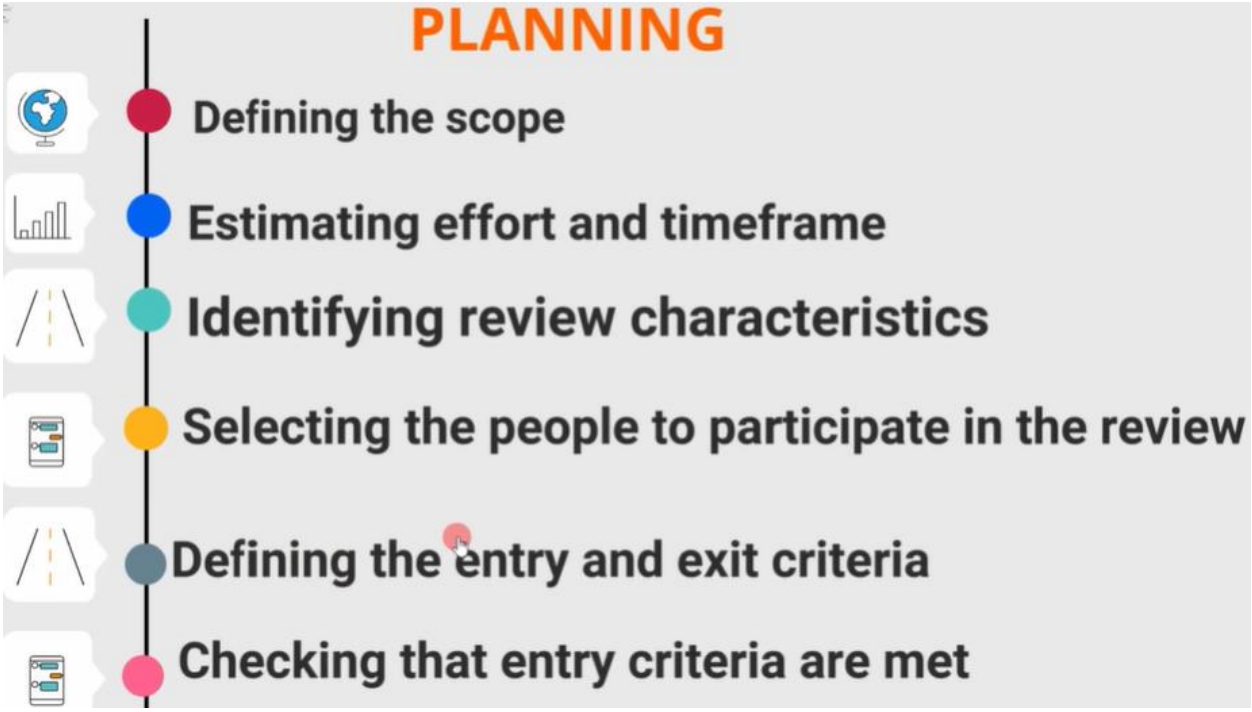
Additional benefits of static testing may include

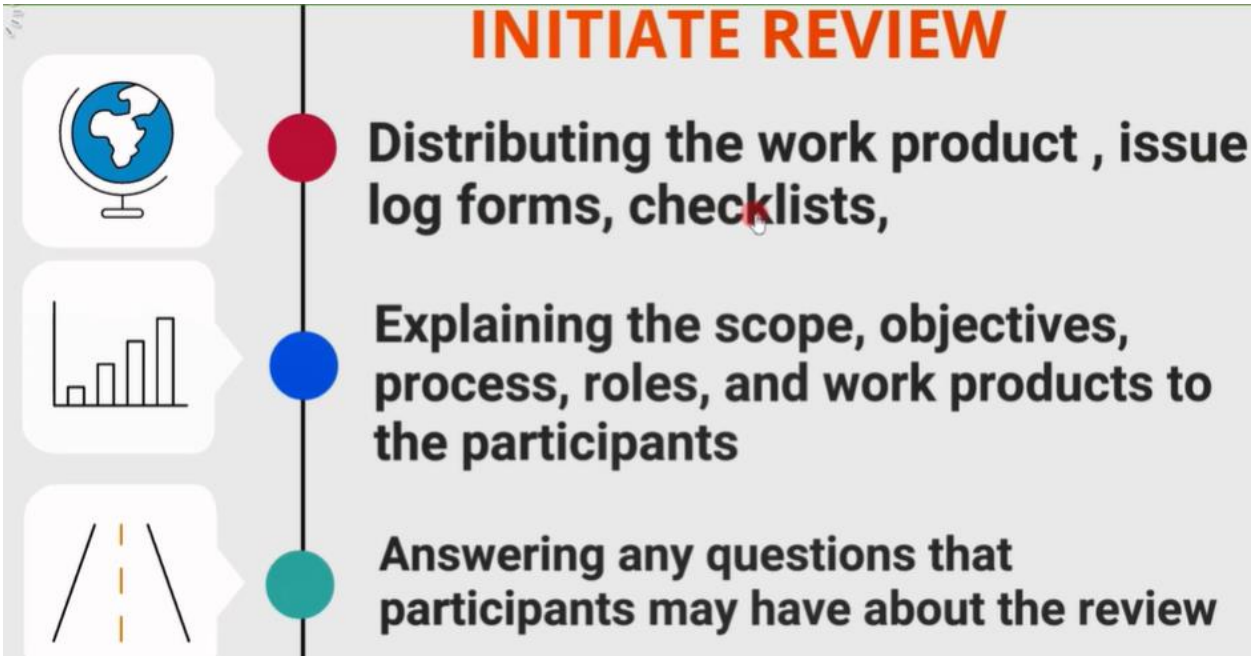
- Detecting and correcting defects more efficiently, and prior to dynamic test execution
- Identifying defects which are not easily found by dynamic testing
- Preventing defects in design or coding by uncovering inconsistencies, ambiguities, contradictions, omissions, inaccuracies, and redundancies in requirements
- Increasing development productivity (e.g., due to improved design, more maintainable code)
- Reducing development cost and time
- Reducing testing cost and time
- Reducing total cost of quality over the software's lifetime, due to fewer failures later in the lifecycle or after delivery into operation
- Improving communication between team members in the course of participating in reviews

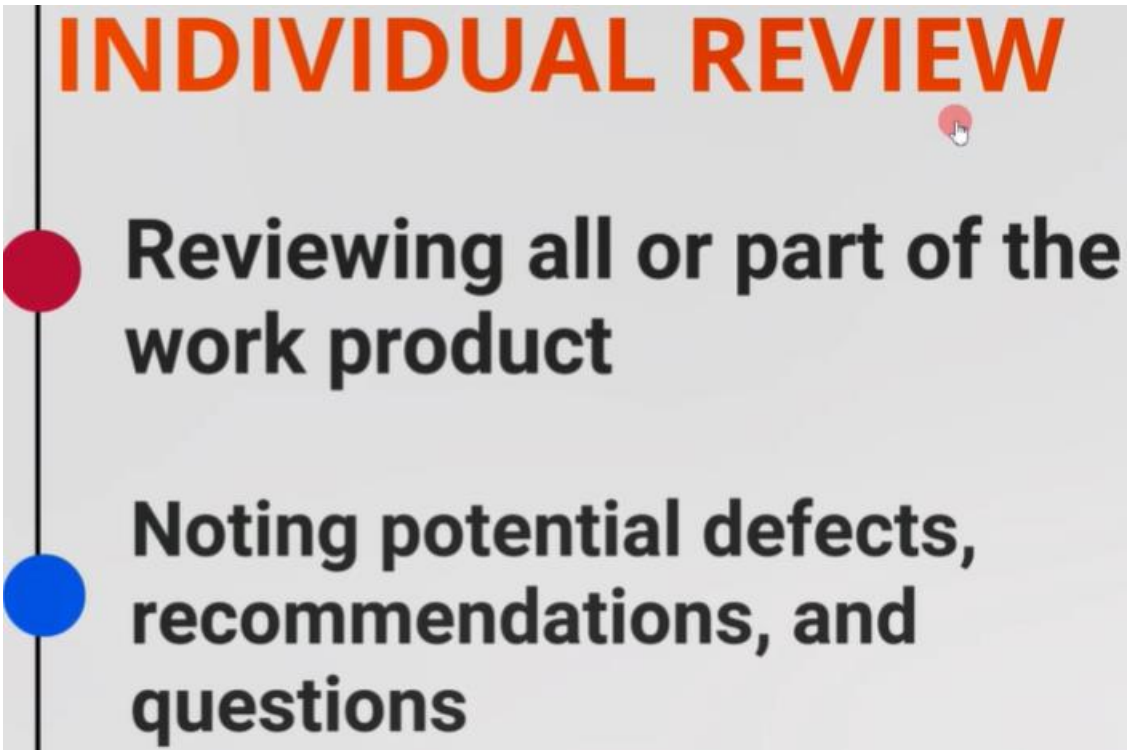
68. 3_1_3 Static VS Dynamic Testing

DEFECT TYPES	VS	EXAMPLEs
REQUIREMENT DEFECTS	DEFECTS	Consistencies, ambiguities, contradictions, omissions, inaccuracies, and redundancies
DESIGN DEFECTS		Efficient algorithms or database structures, high coupling, low cohesion
CODING DEFECTS		Variables with undefined values, variables that are declared but never used, unreachable code, duplicate code
DEVIATIONS FROM STANDARDS		Lack of adherence to coding standards
INCORRECT INTERFACE SPECIFICATIONS		Different units of measurement used by the calling system than by the called system
SECURITY VULNERABILITIES		Susceptibility to buffer overflows
INACCURACIES TRACEABILITY		Missing tests for an acceptance criterion


REVIEW PROCESS
PLANNING
Fixing and reporting
Initiate review
Review Process
Issue communication and analysis
Individual review


PLANNING
Defining the scope
Estimating effort and timeframe
Identifying review characteristics
Selecting the people to participate in the review
Defining the entry and exit criteria
Checking that entry criteria are met


INITIATE REVIEW
Distributing the work product , issue log forms, checklists,
Explaining the scope, objectives, process, roles, and work products to the participants
Answering any questions that participants may have about the review


INDIVIDUAL REVIEW
Reviewing all or part of the work product
Noting potential defects, recommendations, and questions

ISSUE COMMUNICATION AND ANALYSIS

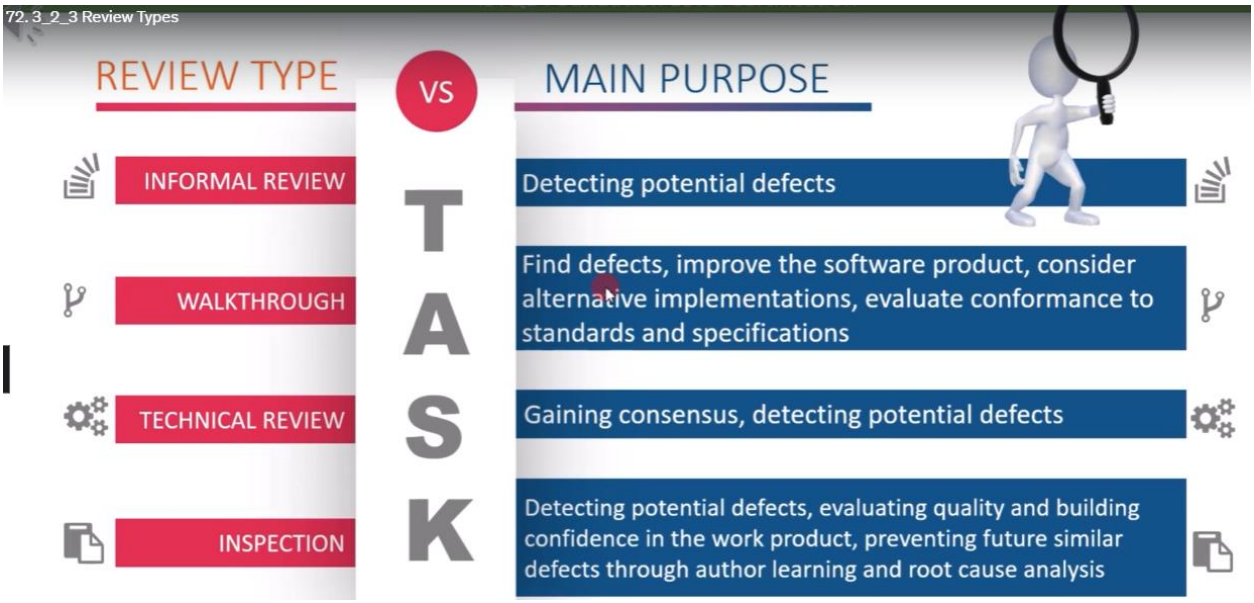
- Communicating identified potential defects
- Analyzing potential defects, assigning ownership and status to them
- Evaluating and documenting quality characteristics
- Evaluating the review findings against the exit criteria

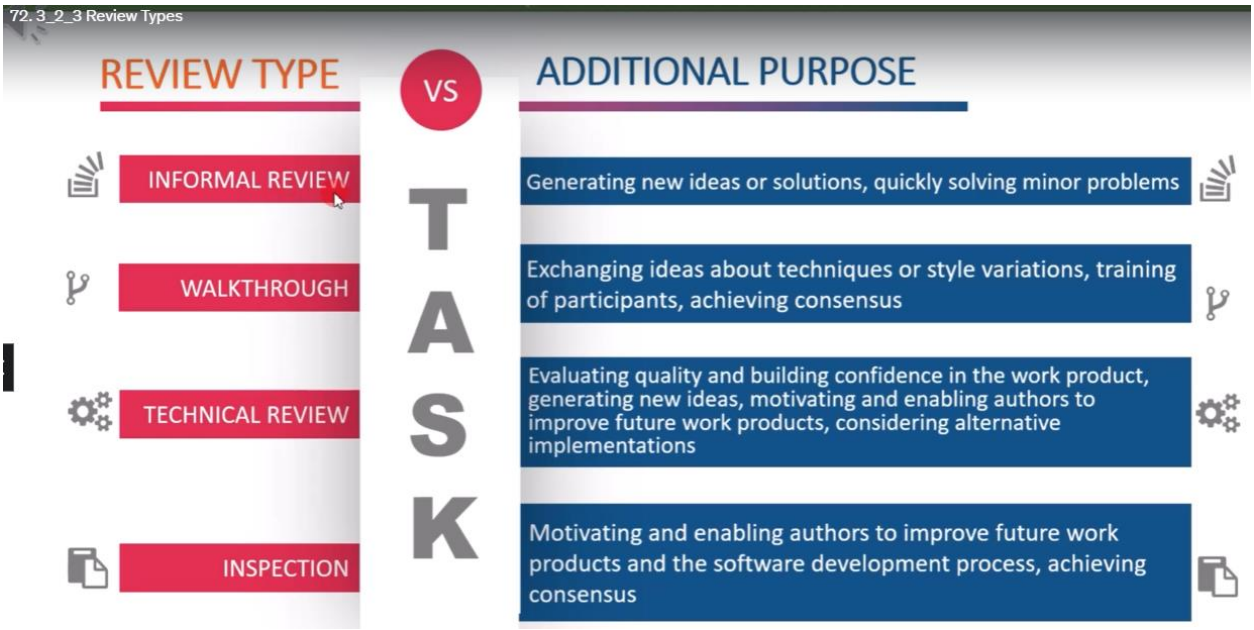
70. 3_2_1 Work Product Review Process

Fixing and reporting

- Creating defect reports for those findings that require changes to a work product
- Fixing defects found in the work product reviewed
- Communicating defects to the appropriate person or team
- Recording updated status of defect
- Gathering metrics
- Checking that exit criteria are met
- Accepting the work product when the exit criteria are reached

71. 3_2_2 Roles and responsibilities in a formal review
MAIN RESPONSIBILITIE
Creates work product
AUTHOR
Review planning
MANAGEMENT
Running review meeting
Facilitator
Responsible for review
REVIEW LEADER
Identify work product defect
Reviewers
Records defect
Scribe

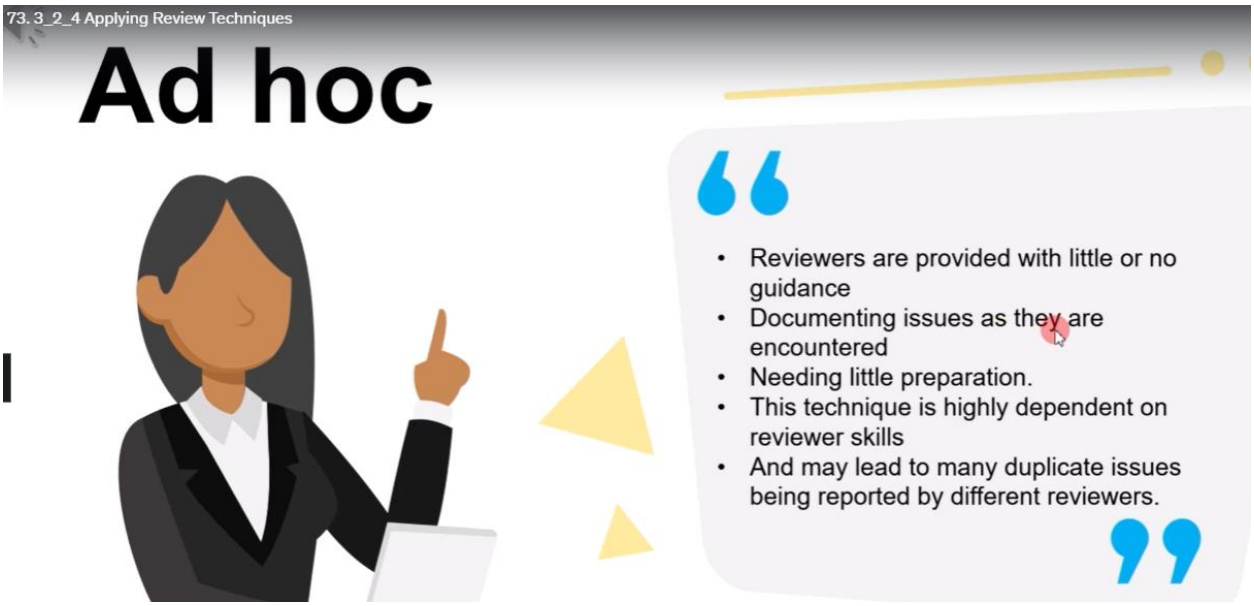

72. 3_2_3 Review Types
REVIEW TYPE
VS
MAIN PURPOSE
INFORMAL REVIEW
Detecting potential defects
WALKTHROUGH
Find defects, improve the software product, consider alternative implementations, evaluate conformance to standards and specifications
TECHNICAL REVIEW
Gaining consensus, detecting potential defects
INSPECTION
Detecting potential defects, evaluating quality and building confidence in the work product, preventing future similar defects through author learning and root cause analysis
TASK



72. 3_2_3 Review Types

	INFORMAL REVIEW	WALKTHROUGH	TECHNICAL REVIEW	INSPECTION
Individual preparation	NO	Optional	YES	YES
Review meeting	NO	YES	Optional	YES
Scribe	NO	YES	YES	YES
checklists	NO	Optional	Optional	YES
Defect logs & Review report	Optional	YES	YES	YES

73. 3_2_4 Applying Review Techniques
REVIEW TECHNIQUES
Ad hoc
Checklist-based
Scenarios and dry runs
Perspective-based
Role-based


73. 3_2_4 Applying Review Techniques
Ad hoc
• Reviewers are provided with little or no guidance
• Documenting issues as they are encountered
• Needing little preparation.
• This technique is highly dependent on reviewer skills
• And may lead to many duplicate issues being reported by different reviewers.

74. 3_2_5 Success Factors for Reviews

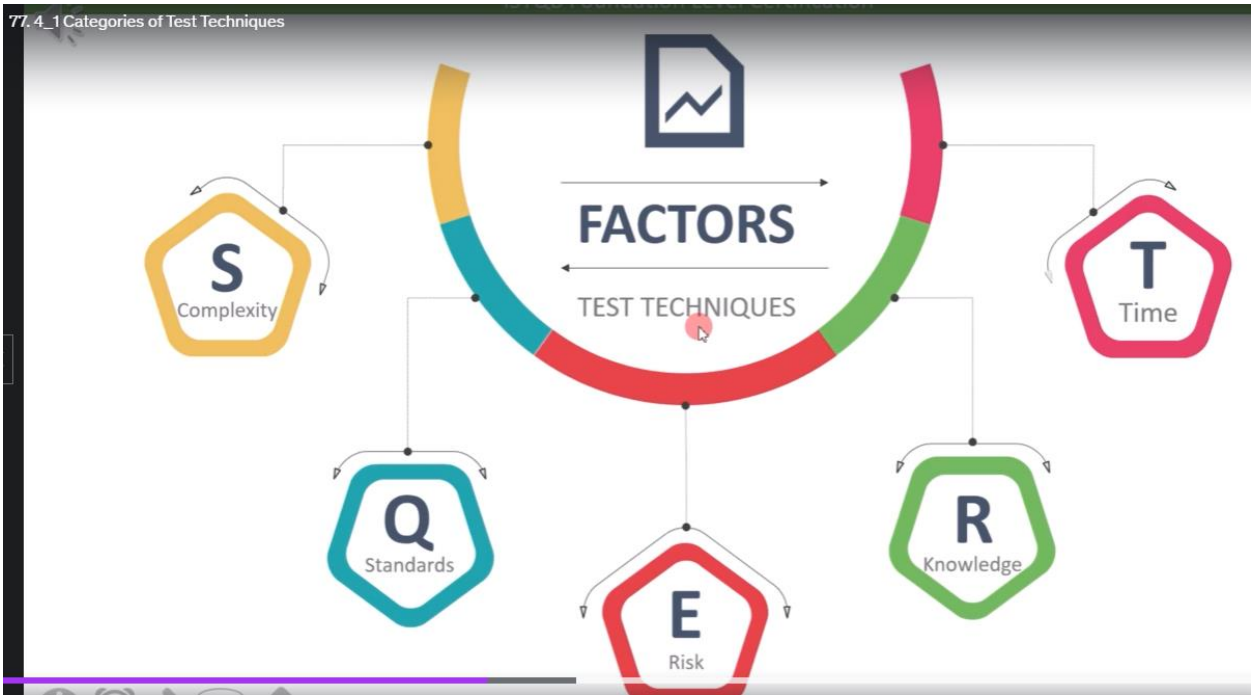
Organisational Factors

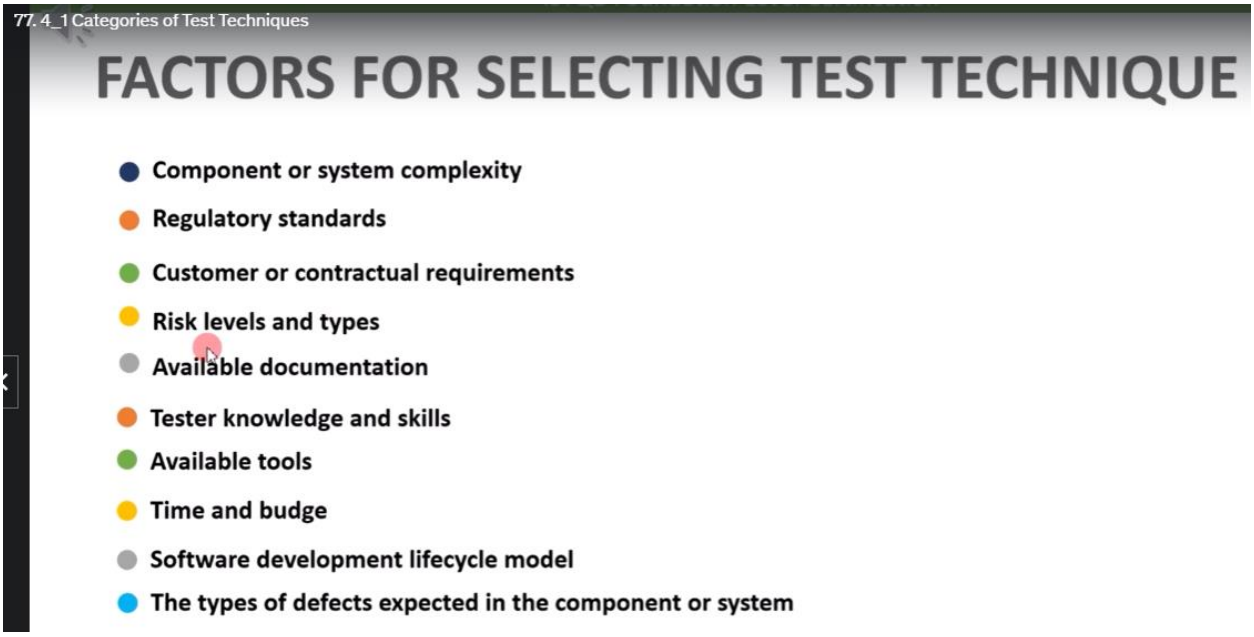
- **Each review has clear objectives, defined during review planning, and used as measurable exit criteria**
- Review types are applied which are suitable to achieve the objectives and are appropriate to the type and level of software work products and participants
- **Any review techniques used, such as checklist-based or role-based reviewing, are suitable for effective defect identification in the work product to be reviewed**
- Any checklists used address the main risks and are up to date
- **Large documents are written and reviewed in small chunks, so that quality control is exercised by providing authors early and frequent feedback on defects**
- Participants have adequate time to prepare
- **Reviews are scheduled with adequate notice**
- Management supports the review process
- **Reviews are integrated in the company's quality and/or test policies**

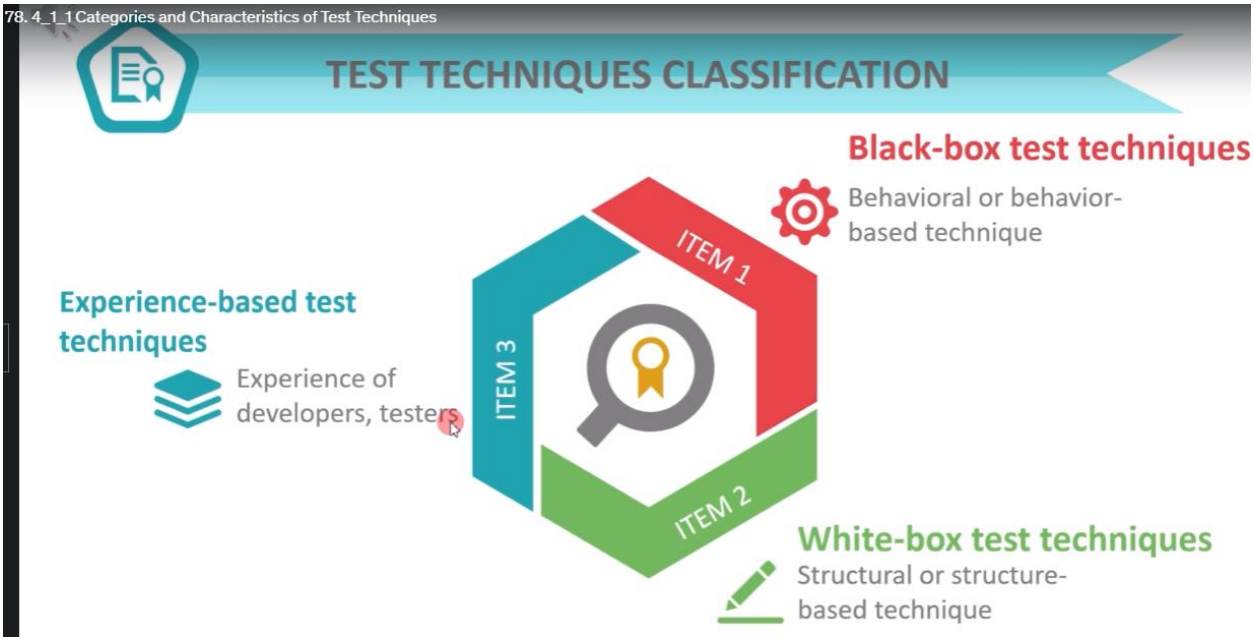
74. 3_2_5 Success Factors for Reviews

Individual Factors

- The right people are involved to meet the review objectives, for example, people with different skill sets or perspectives, who may use the document as a work input
- **Testers are seen as valued reviewers who contribute to the review and learn about the work product, which enables them to prepare more effective tests, and to prepare those tests earlier**
- Participants dedicate adequate time and attention to detail
- **Reviews are conducted on small chunks, so that reviewers do not lose concentration during individual review and/or the review meeting (when held)**
- Defects found are acknowledged, appreciated, and handled objectively
- **The meeting is well-managed, so that participants consider it a valuable use of their time**
- The review is conducted in an atmosphere of trust; the outcome will not be used for the evaluation of the participants
- **Participants avoid body language and behaviours that might indicate boredom, exasperation, or hostility to other participants**
- Adequate training is provided, especially for more formal review types such as inspections
- **A culture of learning and process improvement is promoted**

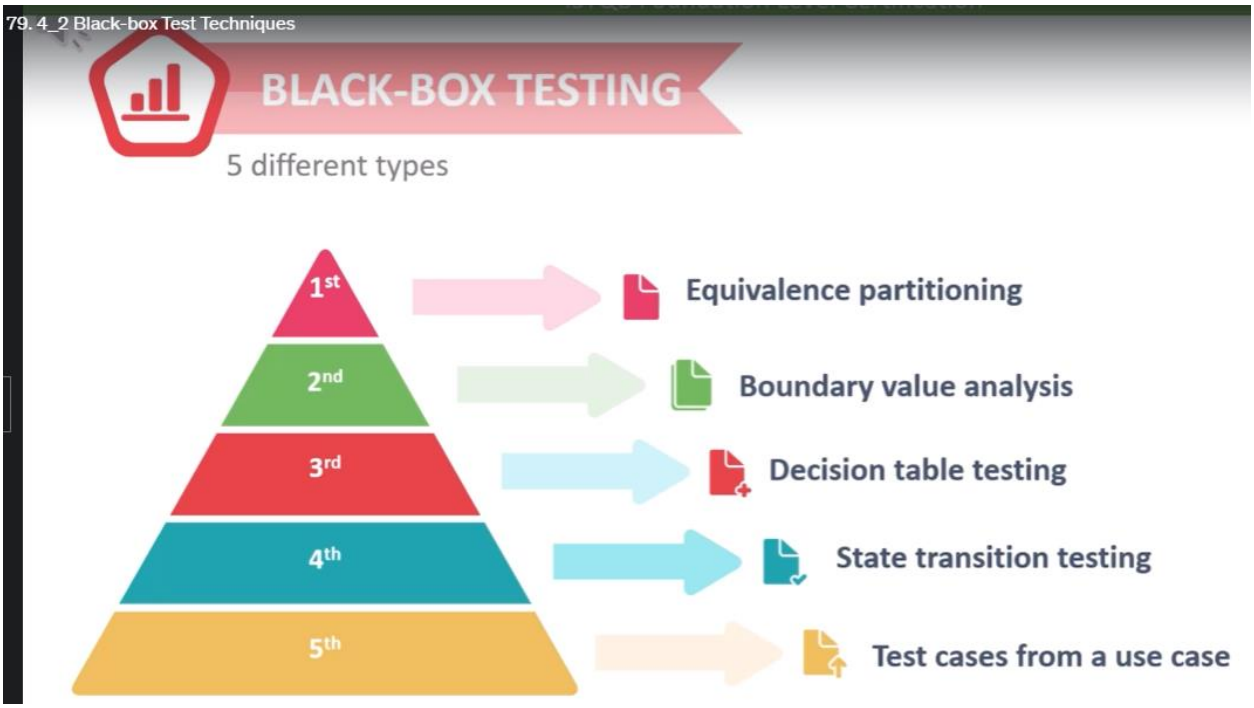

77. 4_1 Categories of Test Techniques
FACTORS
TEST TECHNIQUES
S
Complexity
T
Time
Q
Standards
R
Knowledge
E
Risk

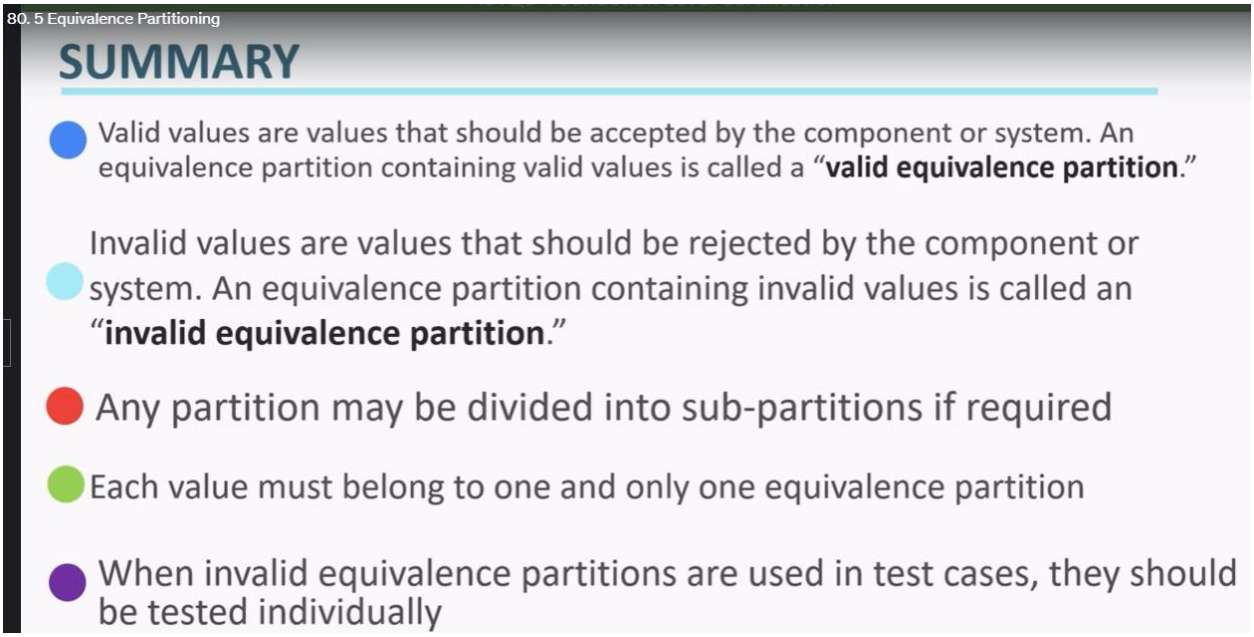

77. 4_1 Categories of Test Techniques
FACTORS FOR SELECTING TEST TECHNIQUE
Component or system complexity
Regulatory standards
Customer or contractual requirements
Risk levels and types
Available documentation
Tester knowledge and skills
Available tools
Time and budge
Software development lifecycle model
The types of defects expected in the component or system

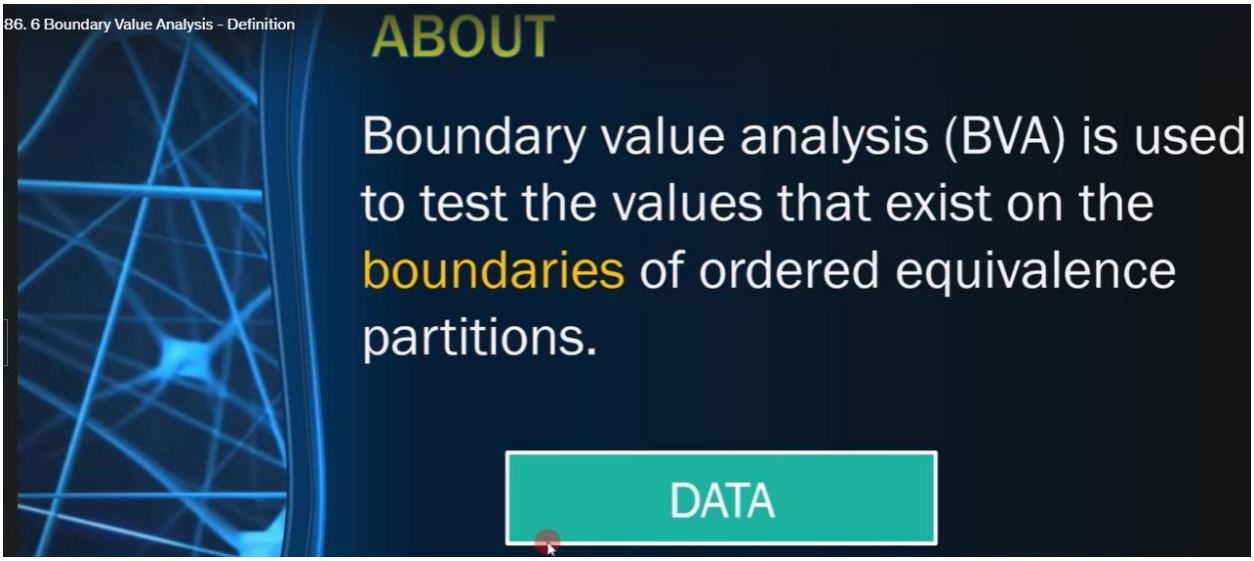


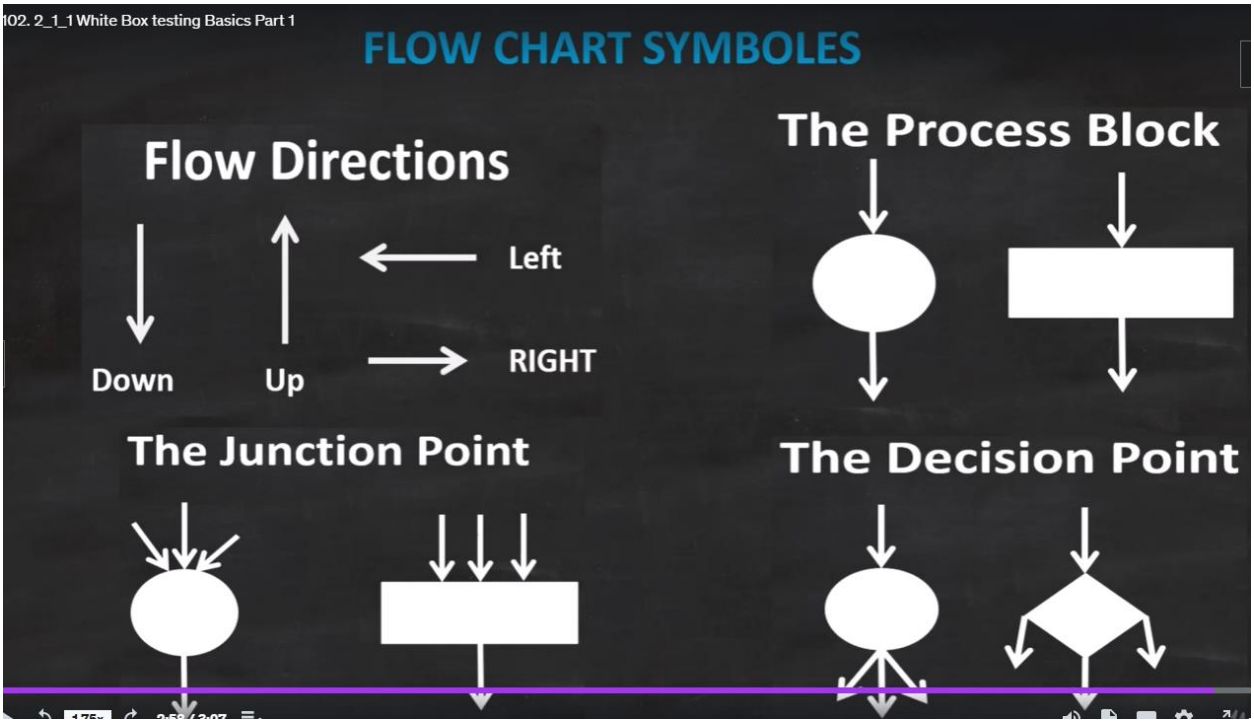
78. 4_1_1 Categories and Characteristics of Test Techniques

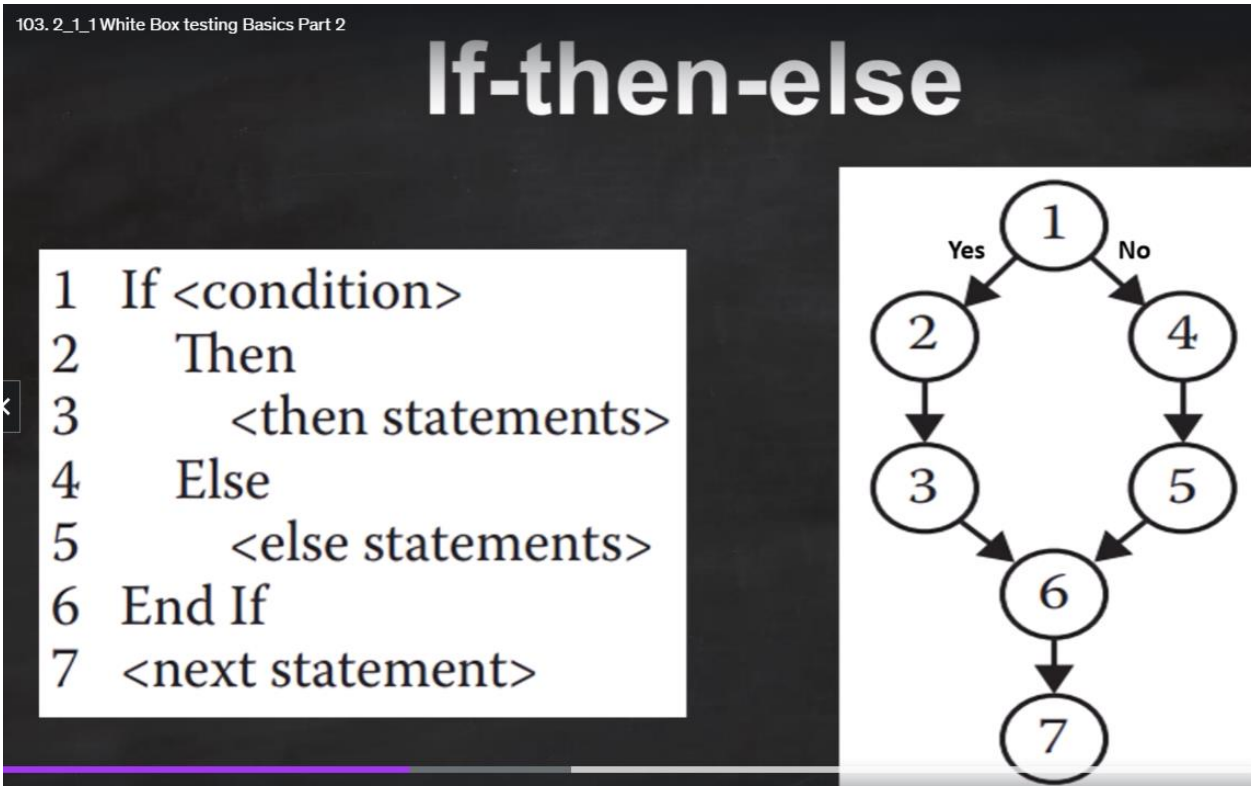
COMMON POINT	BLACK BOX TESTING	WHITE BOX TESTING	EXPERIENCED BASED TESTING
Test conditions, test cases, and test data are derived from a test basis that may include	Software requirements, specifications, use cases, and user stories	code, software architecture, detailed design, or any other source of information regarding the structure of the software	knowledge and experience of testers, developers, users and other stakeholders
Coverage is measured based on the	Items tested in the test basis and the technique applied to the test basis	Items tested within a selected structure (e.g., the code or interfaces) and the technique applied to the test basis	

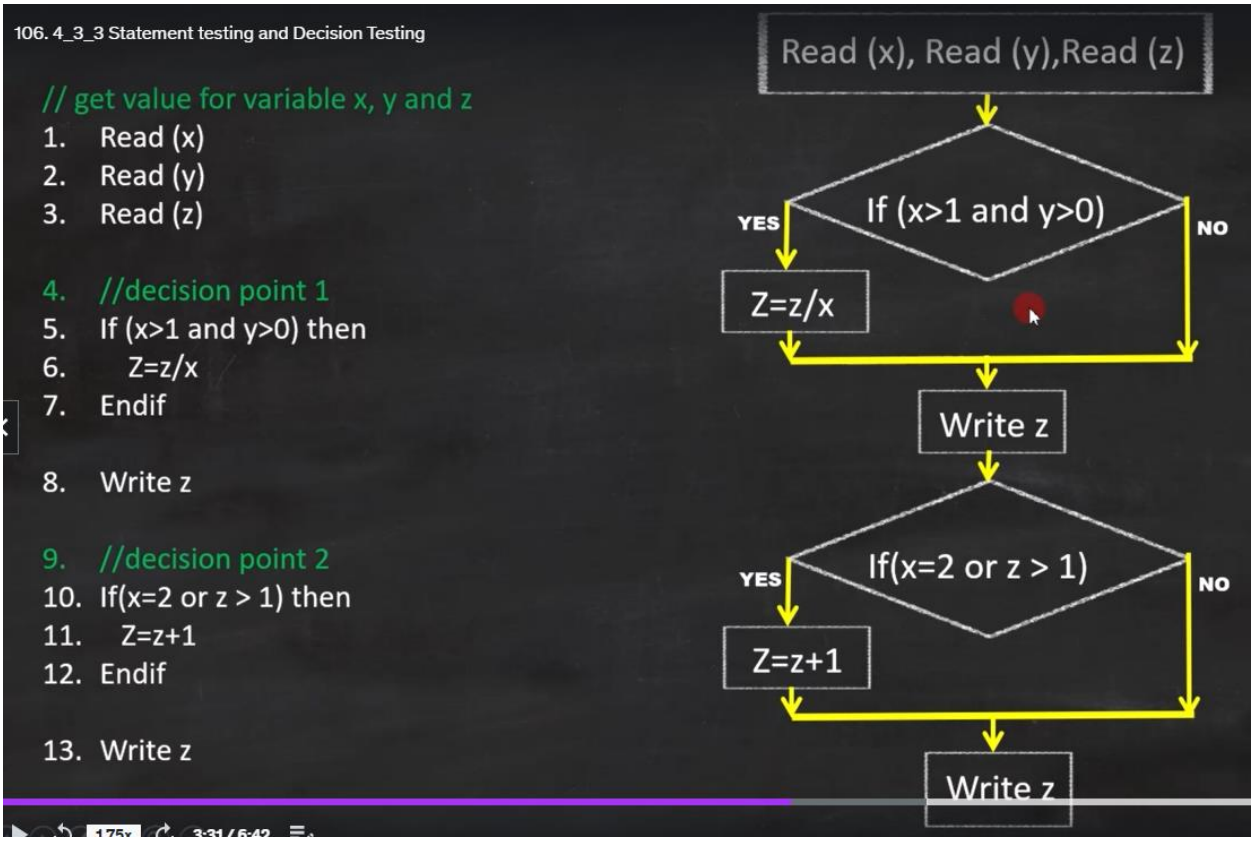

79. 4_2 Black-box Test Techniques
BLACK-BOX TESTING
5 different types
1st
2nd
3rd
4th
5th
Equivalence partitioning
Boundary value analysis
Decision table testing
State transition testing
Test cases from a use case

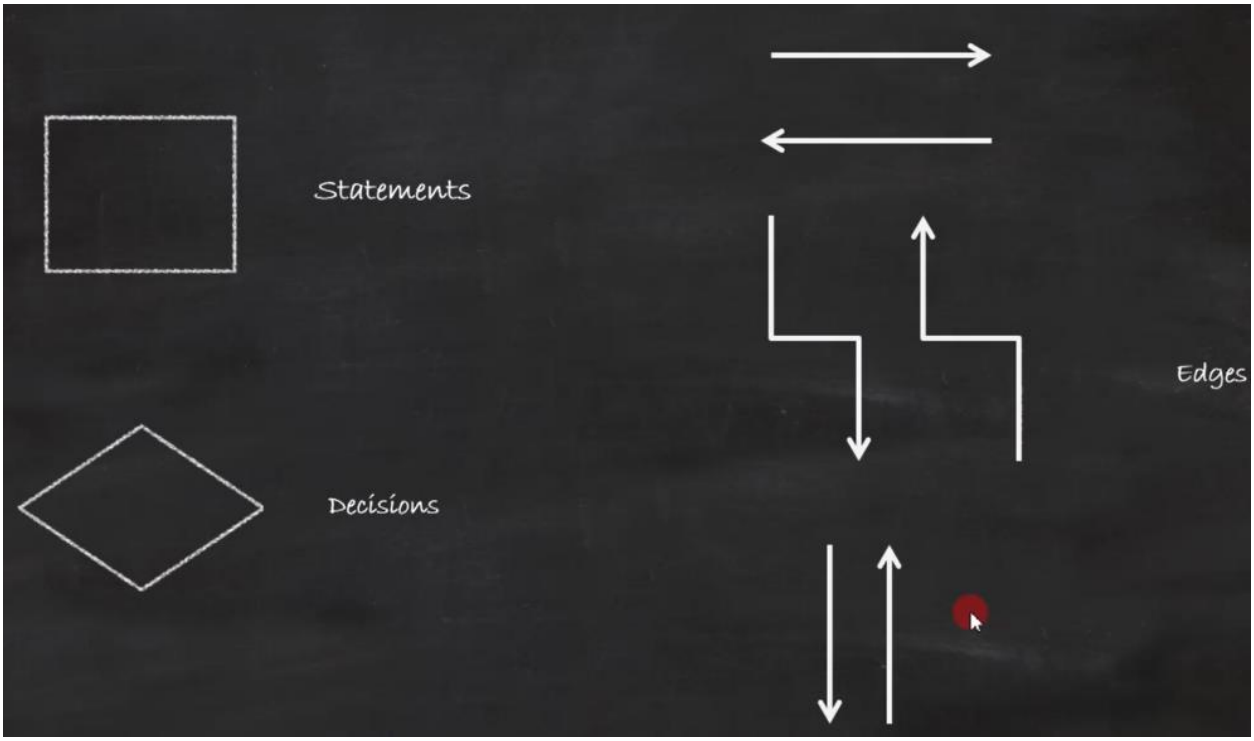

80. 5 Equivalence Partitioning
SUMMARY
Valid values are values that should be accepted by the component or system. An equivalence partition containing valid values is called a "valid equivalence partition."
Invalid values are values that should be rejected by the component or system. An equivalence partition containing invalid values is called an "invalid equivalence partition."
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

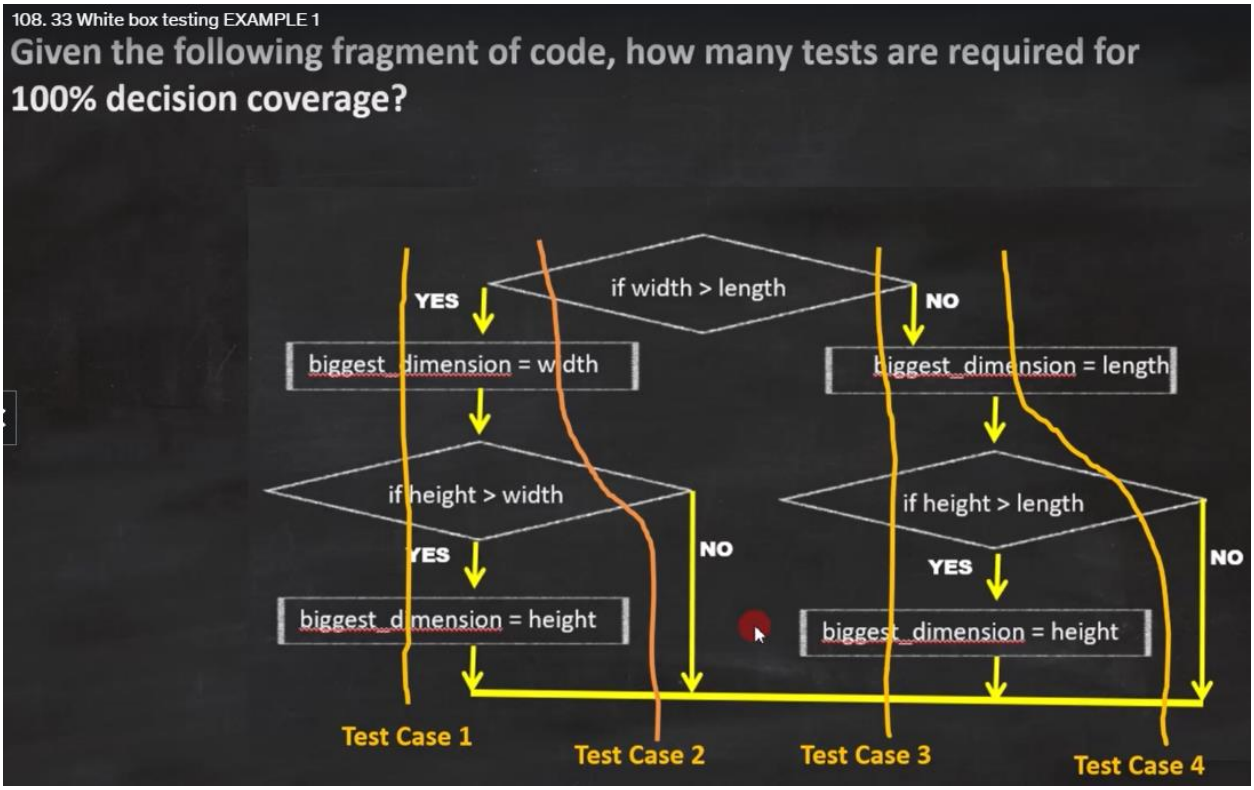

86. 6 Boundary Value Analysis - Definition
ABOUT
Boundary value analysis (BVA) is used to test the values that exist on the boundaries of ordered equivalence partitions.
DATA


102. 2_1_1 White Box testing Basics Part 1
FLOW CHART SYMBOLES
Flow Directions
Down
Up
Left
RIGHT
The Process Block
The Junction Point
The Decision Point


103. 2_1_1 White Box testing Basics Part 2
If-then-else
1 If <condition>
2 Then
3 <then statements>
4 Else
5 <else statements>
6 End If
7 <next statement>
1
Yes
No
2
4
3
5
6
7


106. 4_3_3 Statement testing and Decision Testing
// get value for variable x, y and z
1. Read (x)
2. Read (y)
3. Read (z)
4. //decision point 1
5. If (x>1 and y>0) then
6. Z=z/x
7. Endif
8. Write z
9. //decision point 2
10. If(x=2 or z > 1) then
11. Z=z+1
12. Endif
13. Write z
Read (x), Read (y),Read (z)
If (x>1 and y>0)
YES
NO
Z=z/x
Write z
If(x=2 or z > 1)
YES
NO
Z=z+1
Write z


Statements
Edges
Decisions


108. 33 White box testing EXAMPLE 1
Given the following fragment of code, how many tests are required for 100% decision coverage?
if width > length
YES
NO
biggest_dimension = width
biggest_dimension = length
if height > width
if height > length
YES
NO
YES
NO
biggest_dimension = height
biggest_dimension = height
Test Case 1
Test Case 2
Test Case 3
Test Case 4

109. 33 White box testing EXAMPLE 2

You have designed test cases to provide 100% statement and 100% decision coverage for the following fragment of code.

```
if width > length
then
biggest_dimension = width
else
biggest_dimension = length
end_if
```

The following has been added to the bottom of the code fragment above.

```
print "Biggest dimension is " &
biggest_dimension
print "Width: " & width
print "Length: " & length
```

How many more test cases are required?

Option

A. One more test case will be required for 100 % decision coverage.
B. Two more test cases will be required for 100 % statement coverage, one of which will be used to provide 100% decision coverage.
C. None, existing test cases can be used.
D. One more test case will be required for 100" statement coverage.

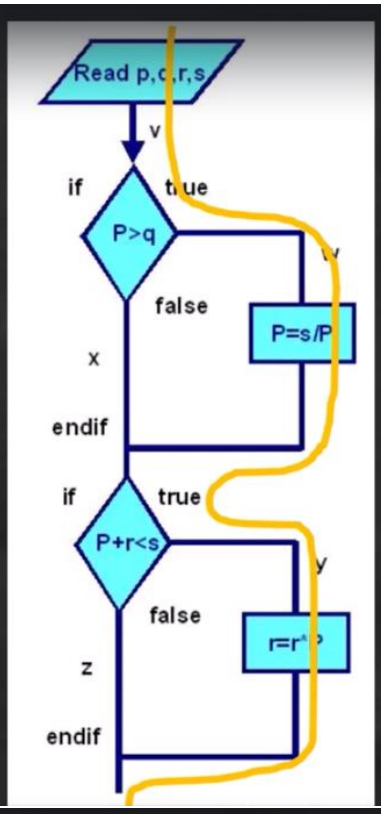
110. 33 White box testing EXAMPLE 3

The diagram represents the following paths through the code.

A. vwy
B. vwz
C. vxy
D. vxz

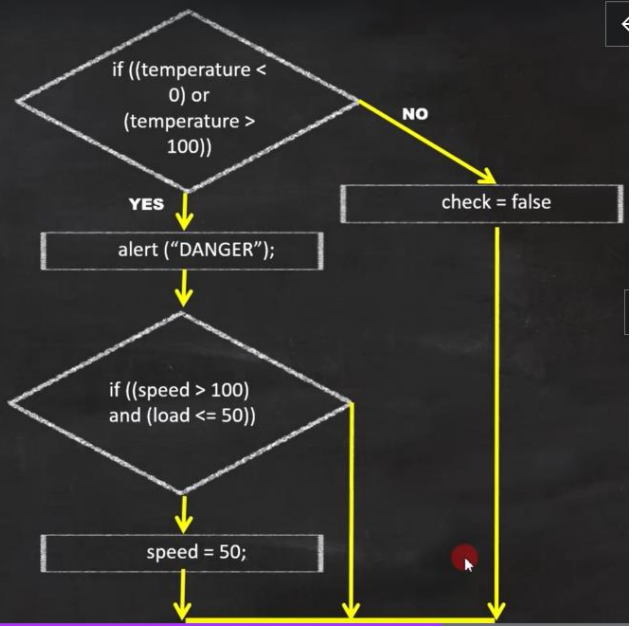
What is the MINIMUM combination of paths required to provide full statement coverage?

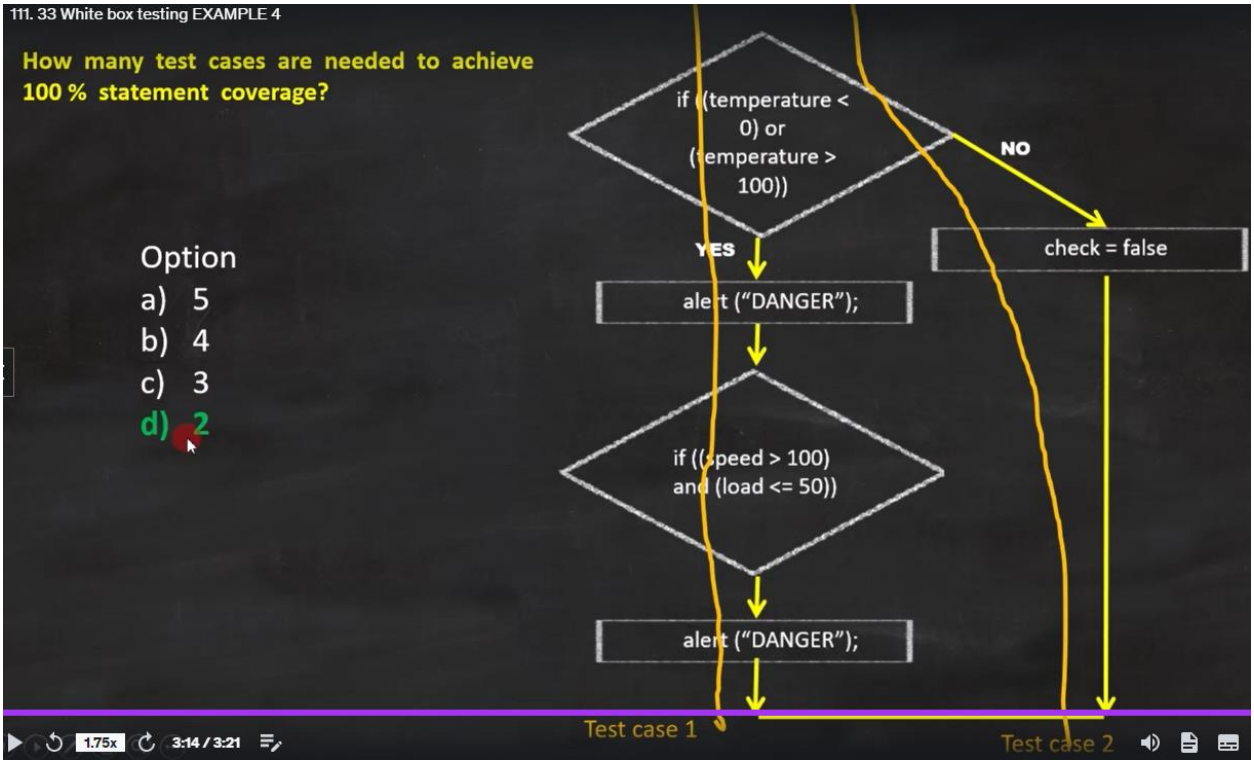
Option
A. A
B. ABD
C. ABCD
D. ACD

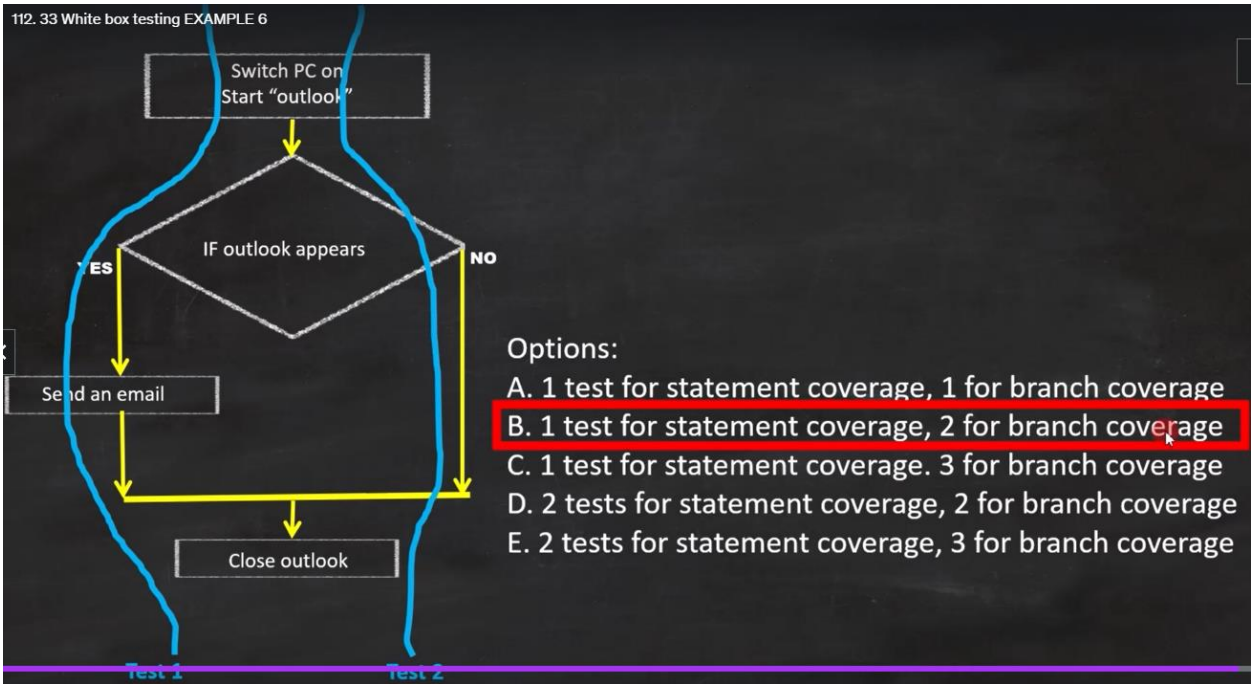


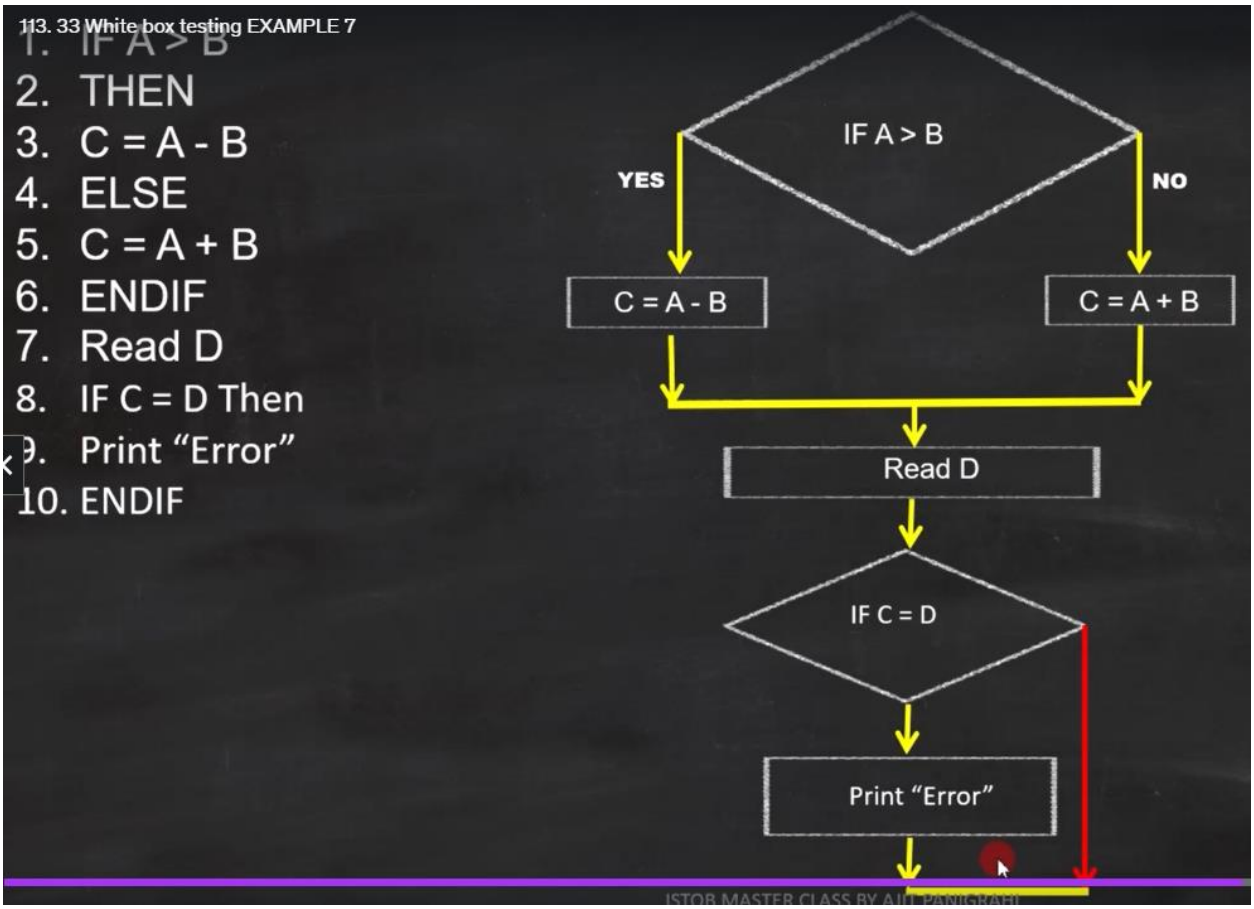
111. 33 White box testing EXAMPLE 4

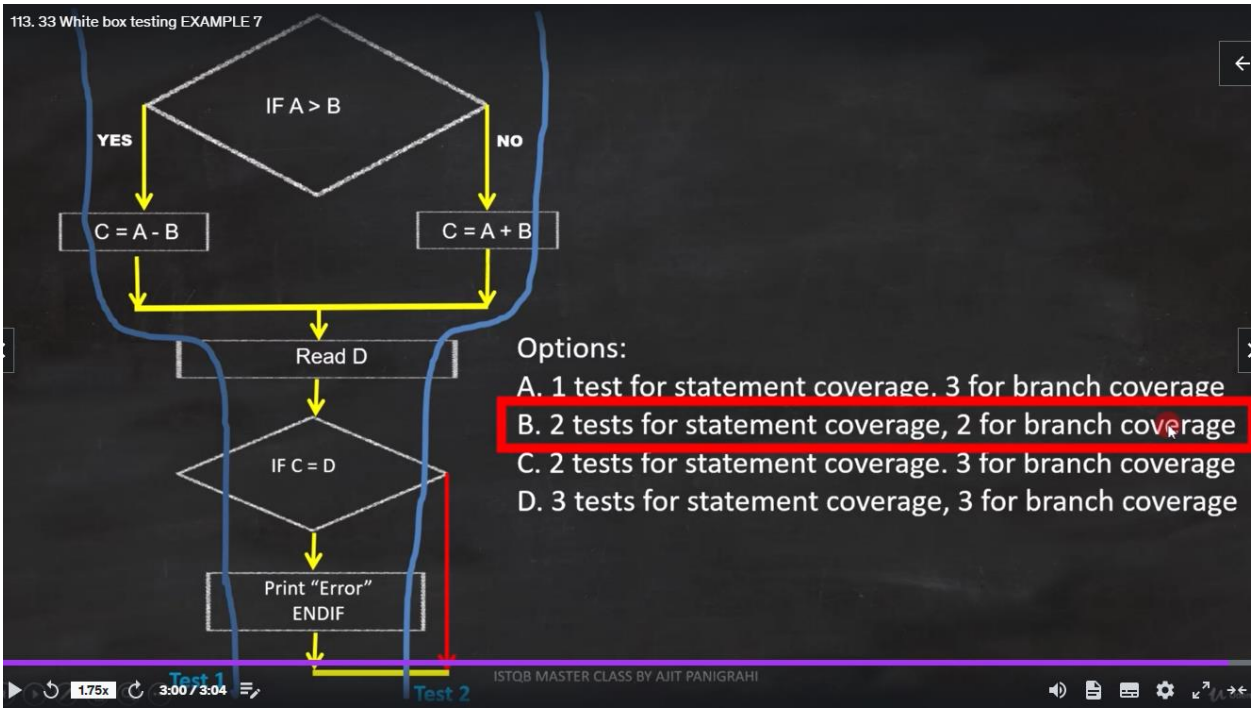
```
1. if ((temperature < 0) or
   (temperature > 100))
2. {
3. alert ("DANGER");
4. if ((speed > 100) and (load <= 50))
5. {
6. speed = 50;
7. }
8. } else
9. {
10. check = false;
11. }
```

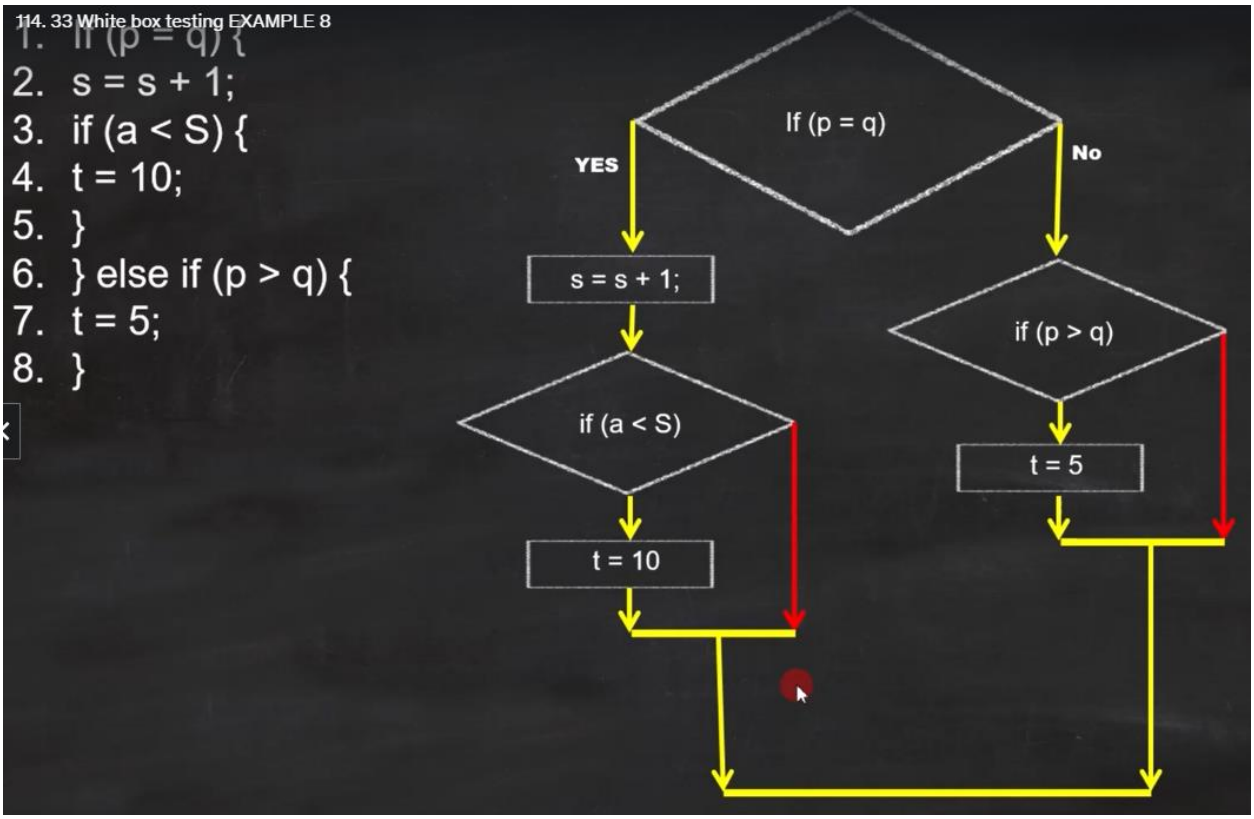


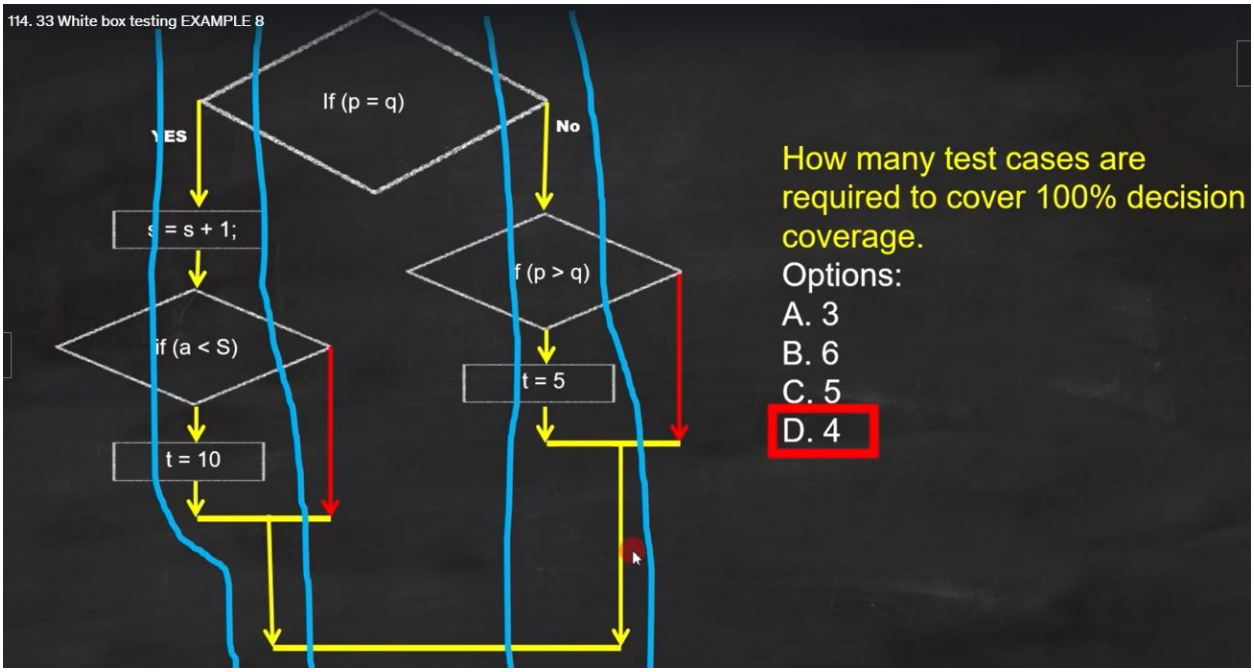

111. 33 White box testing EXAMPLE 4
How many test cases are needed to achieve 100 % statement coverage?
Option
a) 5
b) 4
c) 3
d) 2
if ((temperature < 0) or (temperature > 100))
NO
YES
check = false
alert ("DANGER");
if ((speed > 100) and (load <= 50))
alert ("DANGER");
Test case 1
Test case 2
1.75x
3:14 / 3:21

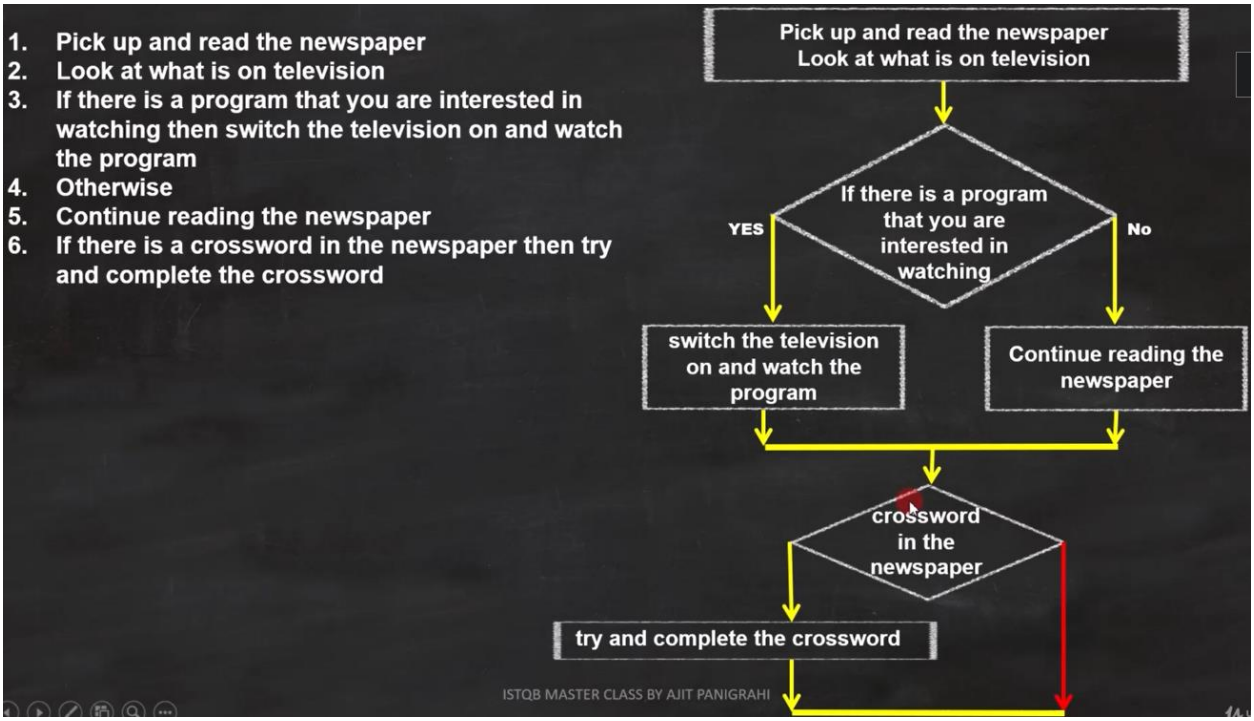

112. 33 White box testing EXAMPLE 6
Switch PC on
Start "outlook"
IF outlook appears
YES
NO
Send an email
Close outlook
Options:
A. 1 test for statement coverage, 1 for branch coverage
B. 1 test for statement coverage, 2 for branch coverage
C. 1 test for statement coverage. 3 for branch coverage
D. 2 tests for statement coverage, 2 for branch coverage
E. 2 tests for statement coverage, 3 for branch coverage

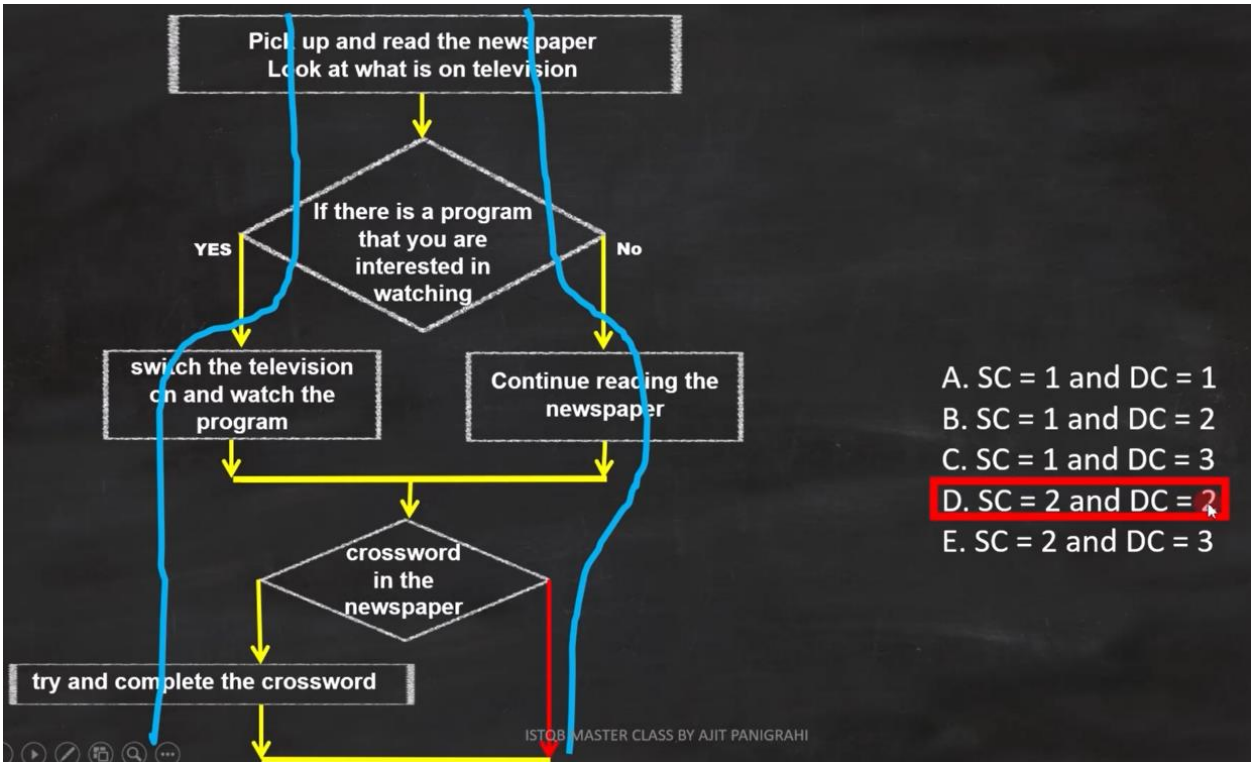

113. 33 White box testing EXAMPLE 7
1. IF A > B
2. THEN
3. C = A - B
4. ELSE
5. C = A + B
6. ENDIF
7. Read D
8. IF C = D Then
9. Print "Error"
10. ENDIF
IF A > B
YES
NO
C = A - B
C = A + B
Read D
IF C = D
Print "Error"

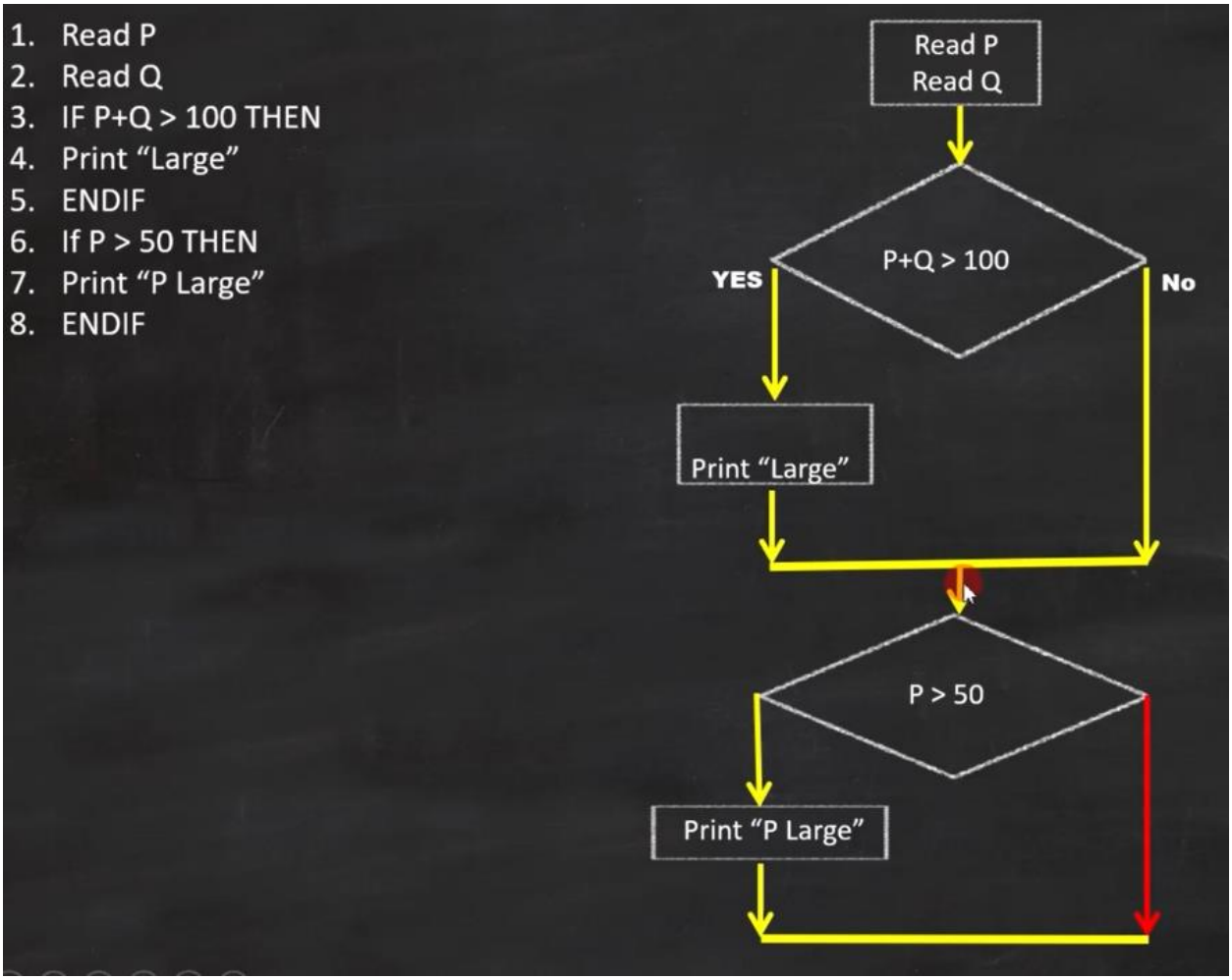

113. 33 White box testing EXAMPLE 7
IF A > B
YES
NO
C = A - B
C = A + B
Read D
IF C = D
Print "Error"
ENDIF
Test 1
Test 2
Options:
A. 1 test for statement coverage. 3 for branch coverage
B. 2 tests for statement coverage, 2 for branch coverage
C. 2 tests for statement coverage. 3 for branch coverage
D. 3 tests for statement coverage, 3 for branch coverage
ISTQB MASTER CLASS BY AJIT PANIGRAHI
1.75x
3:00 / 3:04


114. 33 White box testing EXAMPLE 8
1. if (p = q) {
2. s = s + 1;
3. if (a < S) {
4. t = 10;
5. }
6. } else if (p > q) {
7. t = 5;
8. }
If (p = q)
YES
No
s = s + 1;
if (p > q)
if (a < S)
t = 5
t = 10

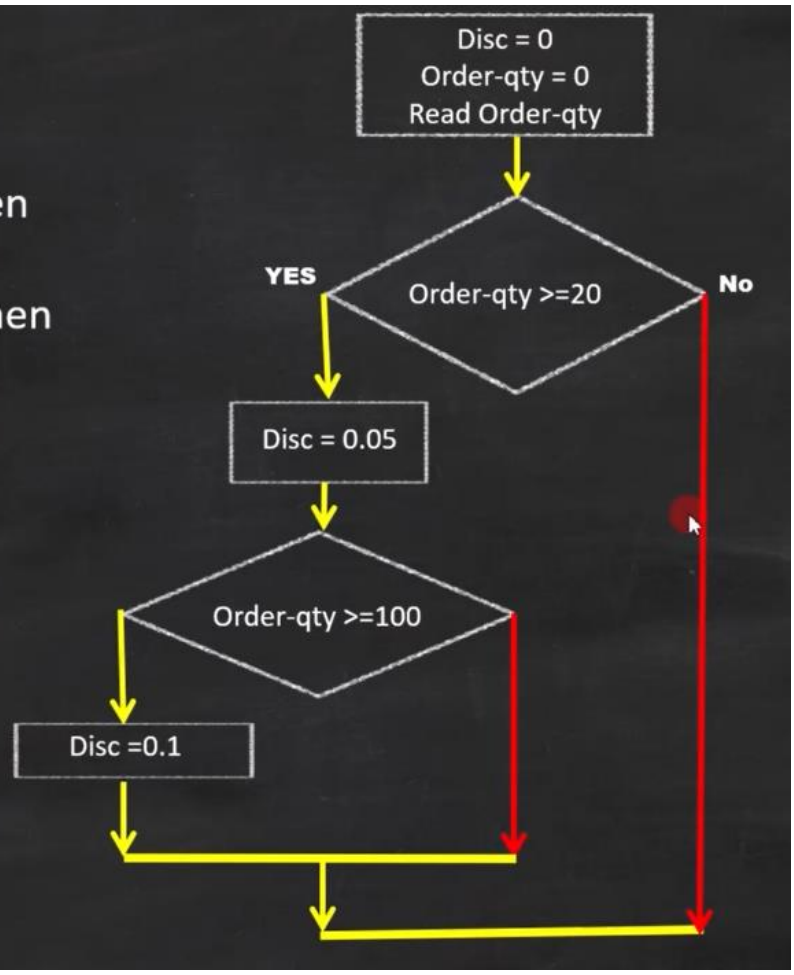

114. 33 White box testing EXAMPLE 8
If (p = q)
No
= s + 1;
f (p > q)
if (a < S)
t = 5
t = 10
How many test cases are required to cover 100% decision coverage.
Options:
A. 3
B. 6
C. 5
D. 4

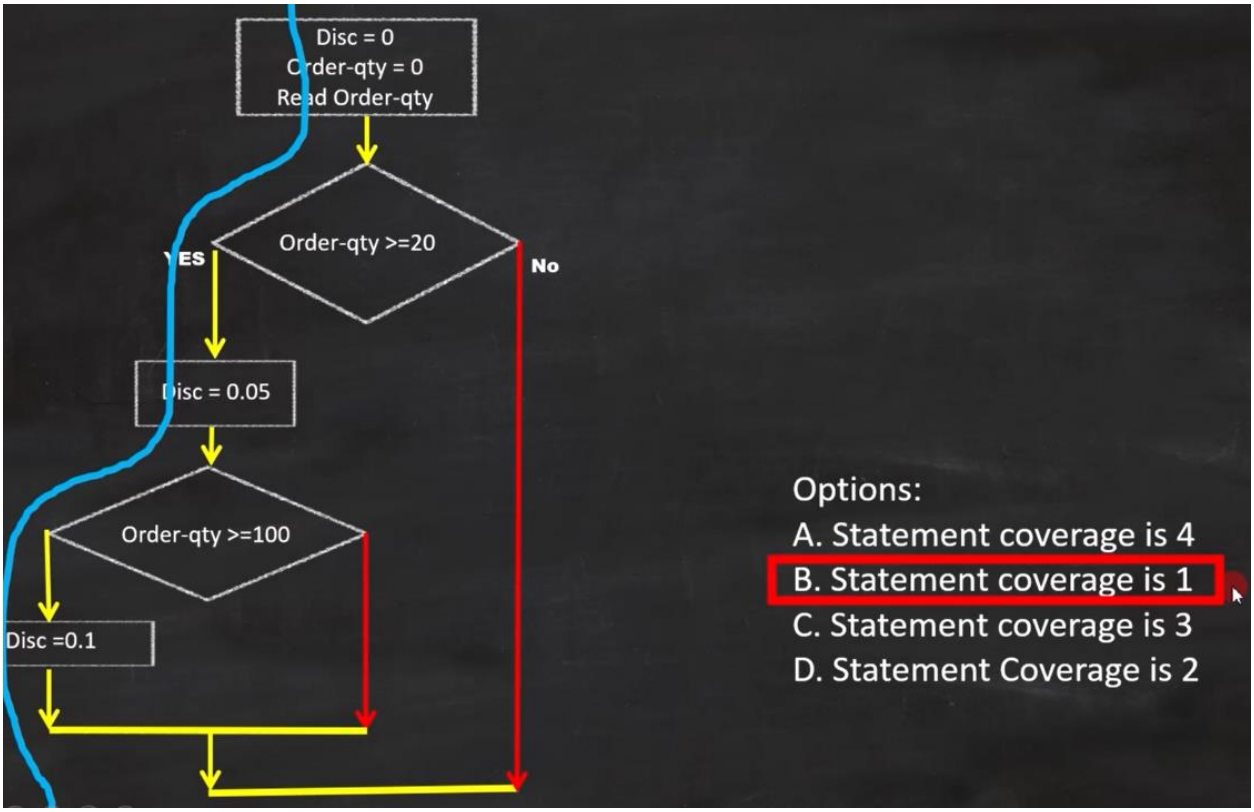

1. Pick up and read the newspaper
2. Look at what is on television
3. If there is a program that you are interested in watching then switch the television on and watch the program
4. Otherwise
5. Continue reading the newspaper
6. If there is a crossword in the newspaper then try and complete the crossword
Pick up and read the newspaper
Look at what is on television
If there is a program that you are interested in watching
YES
No
switch the television on and watch the program
Continue reading the newspaper
crossword in the newspaper
try and complete the crossword
ISTQB MASTER CLASS BY AJIT PANIGRAHI

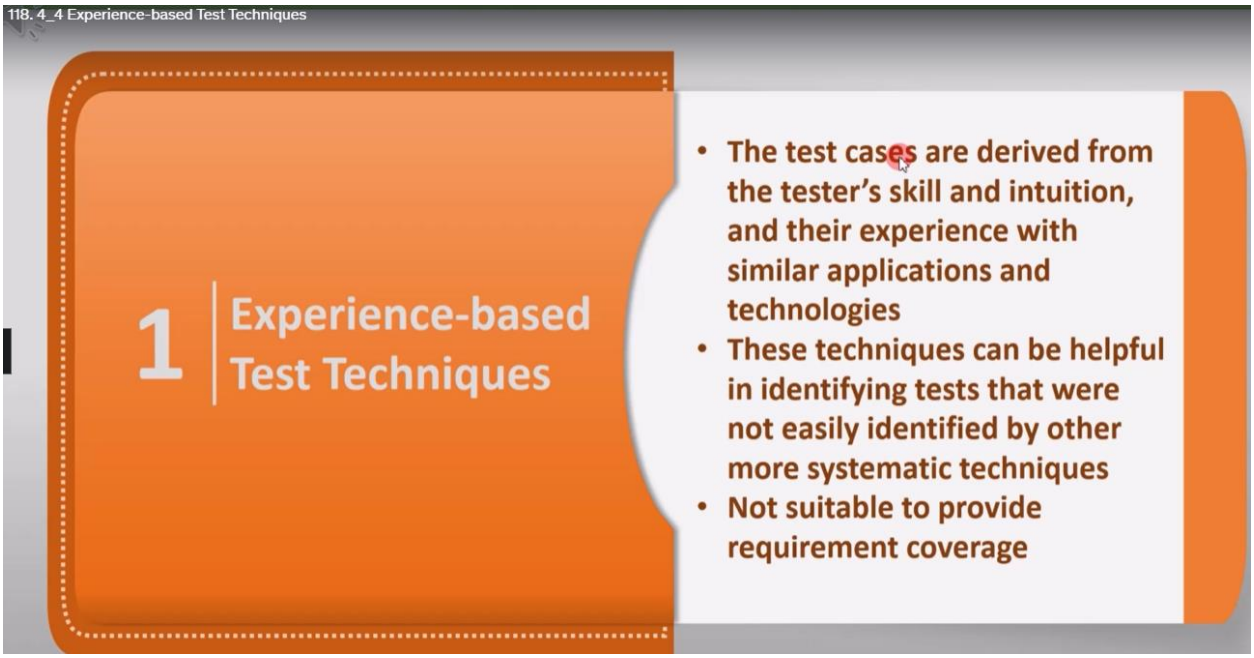

Pick up and read the newspaper
Look at what is on television
If there is a program that you are interested in watching
YES
No
switch the television on and watch the program
Continue reading the newspaper
crossword in the newspaper
try and complete the crossword
ISTQB MASTER CLASS BY AJIT PANIGRAHI
A. SC = 1 and DC = 1
B. SC = 1 and DC = 2
C. SC = 1 and DC = 3
D. SC = 2 and DC = 2
E. SC = 2 and DC = 3

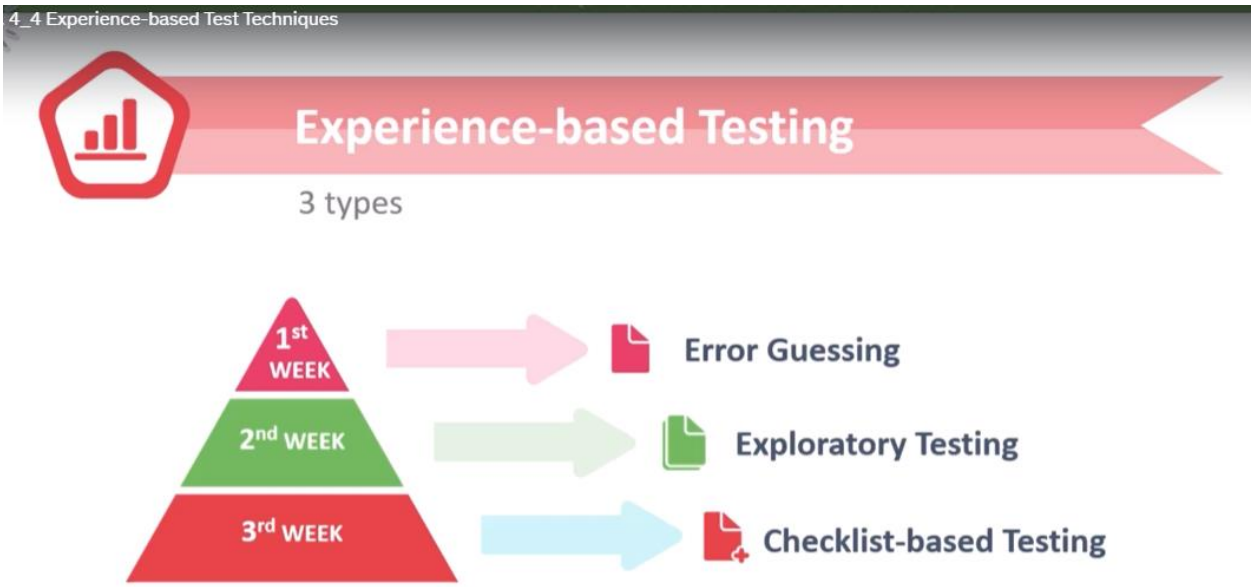

1. Read P
2. Read Q
3. IF P+Q > 100 THEN
4. Print "Large"
5. ENDIF
6. If P > 50 THEN
7. Print "P Large"
8. ENDIF
Read P
Read Q
P+Q > 100
YES
No
Print "Large"
P > 50
Print "P Large"

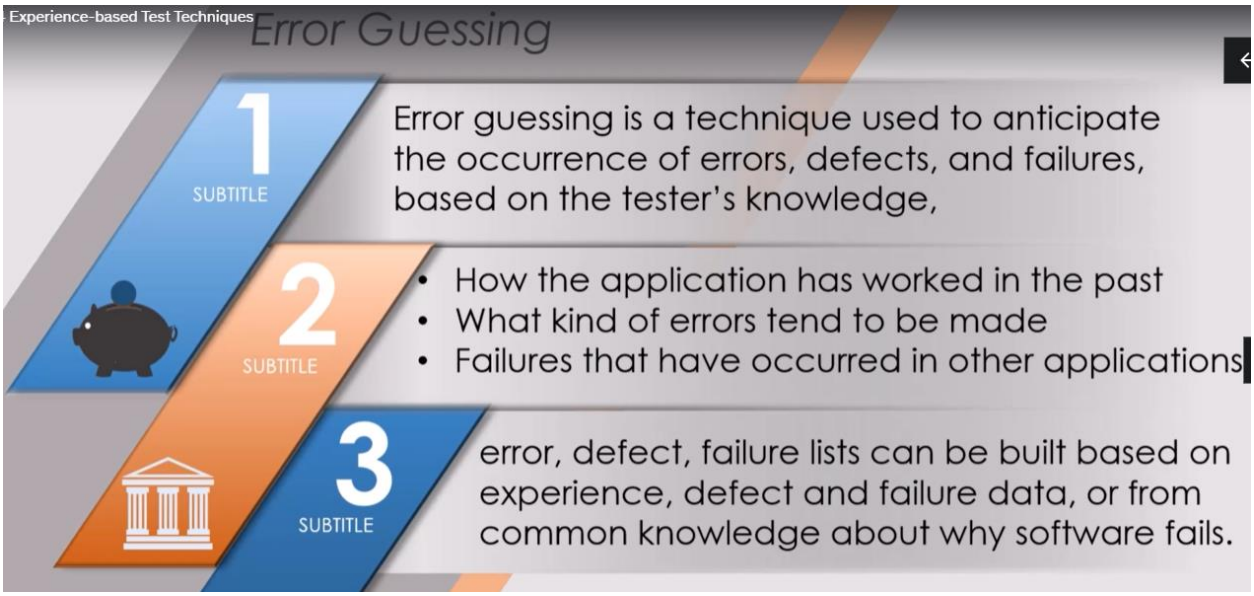
1. Disc = 0
2. Order-qty = 0
3. Read Order-qty
4. If Order-qty >=20 then
5. Disc = 0.05
6. If Order-qty >=100 then
7. Disc =0.1
8. End if
9. End if

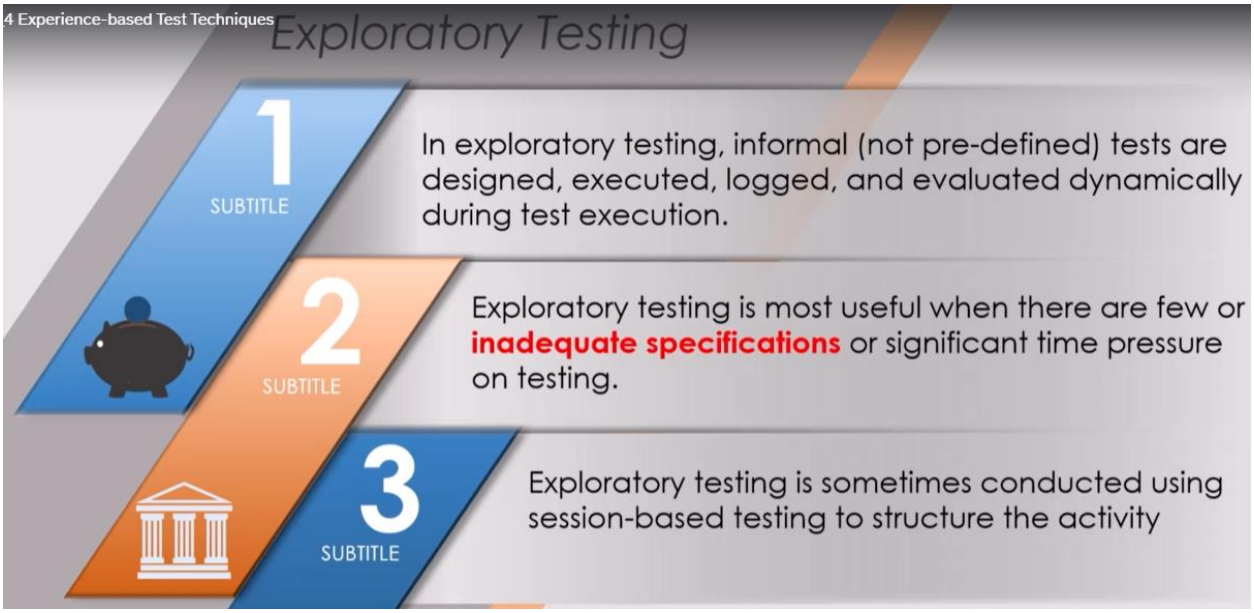


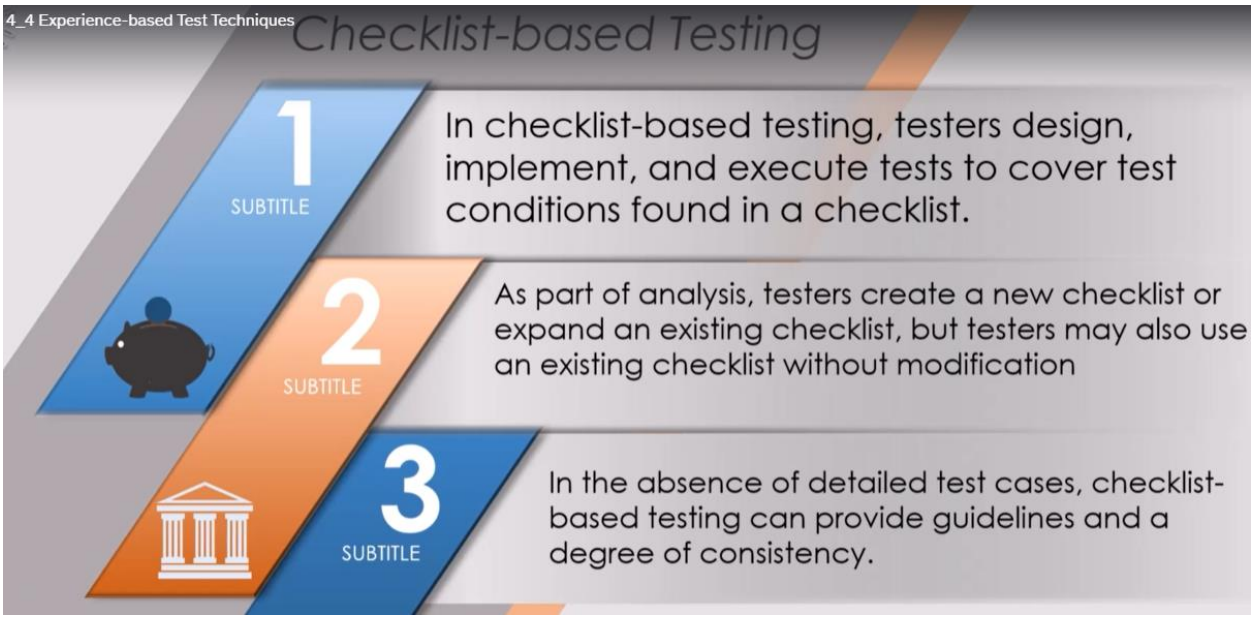

Disc = 0
Order-qty = 0
Read Order-qty
Order-qty >=20
YES
No
Disc = 0.05
Order-qty >=100
Disc =0.1
Options:
A. Statement coverage is 4
B. Statement coverage is 1
C. Statement coverage is 3
D. Statement Coverage is 2


118. 4_4 Experience-based Test Techniques
1
Experience-based Test Techniques
• The test cases are derived from the tester's skill and intuition, and their experience with similar applications and technologies
• These techniques can be helpful in identifying tests that were not easily identified by other more systematic techniques
• Not suitable to provide requirement coverage

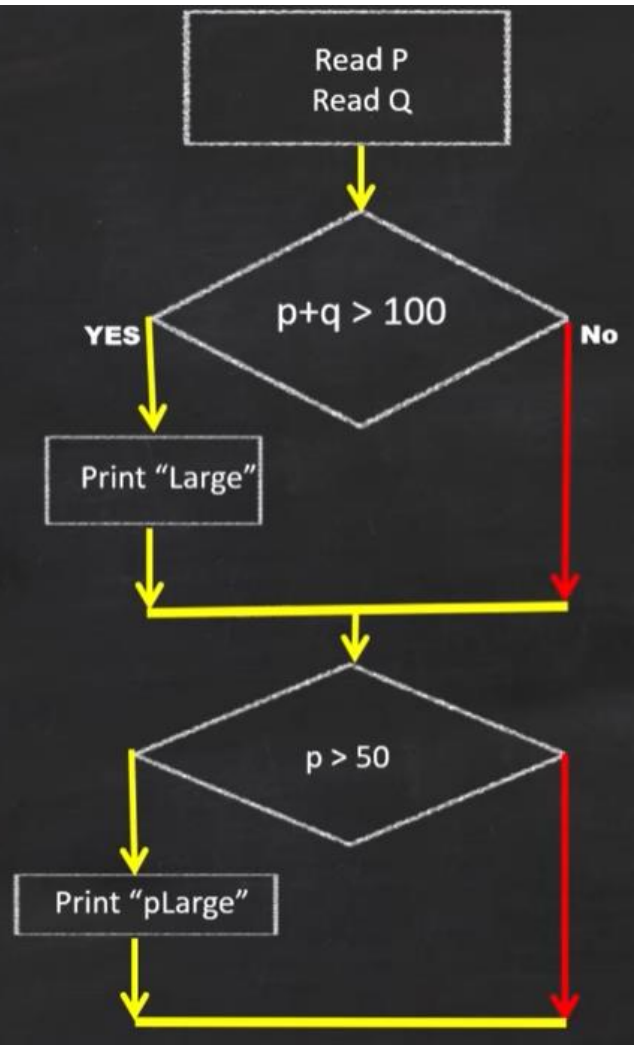

4_4 Experience-based Test Techniques
Experience-based Testing
3 types
1st WEEK
2nd WEEK
3rd WEEK
Error Guessing
Exploratory Testing
Checklist-based Testing

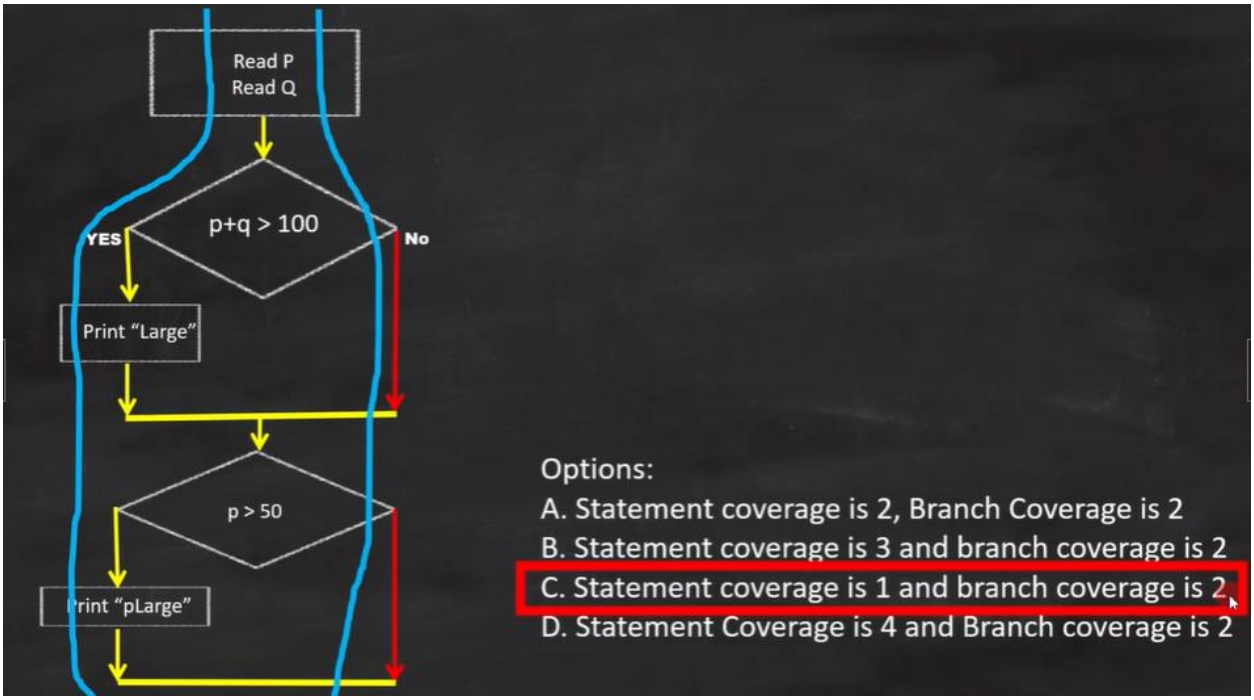

Experience-based Test Techniques
Error Guessing
1
SUBTITLE
Error guessing is a technique used to anticipate the occurrence of errors, defects, and failures, based on the tester's knowledge,
2
SUBTITLE
• How the application has worked in the past
• What kind of errors tend to be made
• Failures that have occurred in other applications
3
SUBTITLE
error, defect, failure lists can be built based on experience, defect and failure data, or from common knowledge about why software fails.

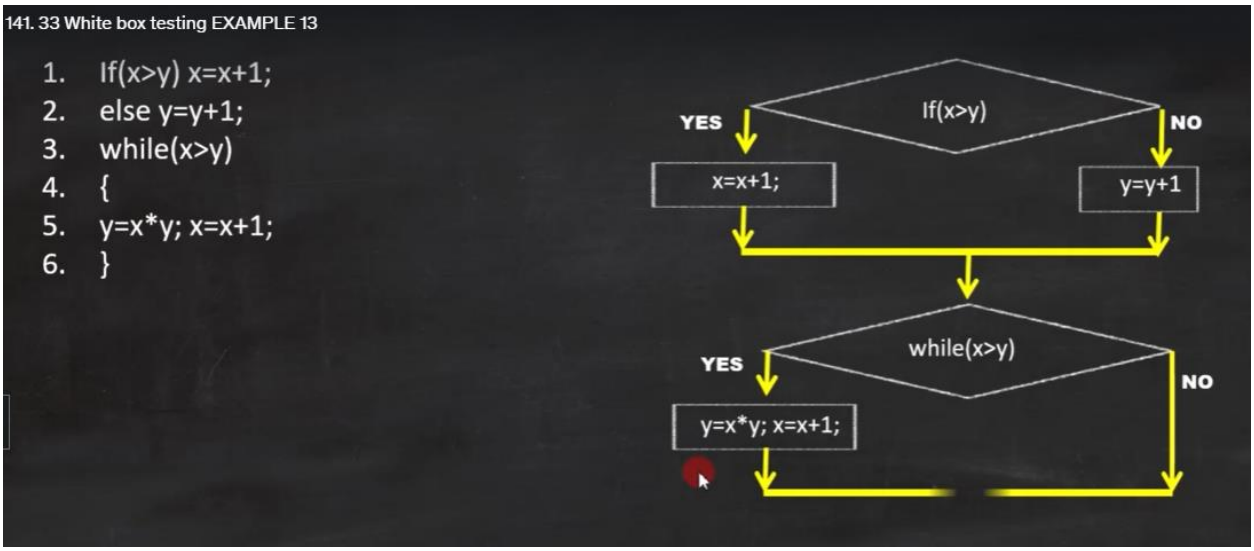

4 Experience-based Test Techniques
Exploratory Testing
1
SUBTITLE
In exploratory testing, informal (not pre-defined) tests are designed, executed, logged, and evaluated dynamically during test execution.
2
SUBTITLE
Exploratory testing is most useful when there are few or inadequate specifications or significant time pressure on testing.
3
SUBTITLE
Exploratory testing is sometimes conducted using session-based testing to structure the activity

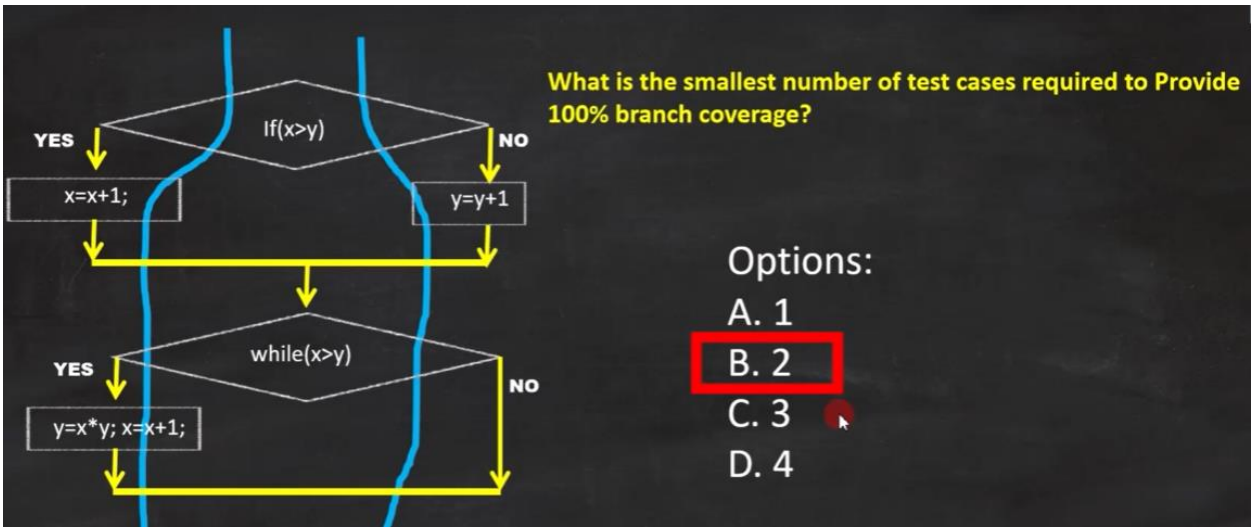

4_4 Experience-based Test Techniques
Checklist-based Testing
1
SUBTITLE
In checklist-based testing, testers design, implement, and execute tests to cover test conditions found in a checklist.
2
SUBTITLE
As part of analysis, testers create a new checklist or expand an existing checklist, but testers may also use an existing checklist without modification
3
SUBTITLE
In the absence of detailed test cases, checklist-based testing can provide guidelines and a degree of consistency.

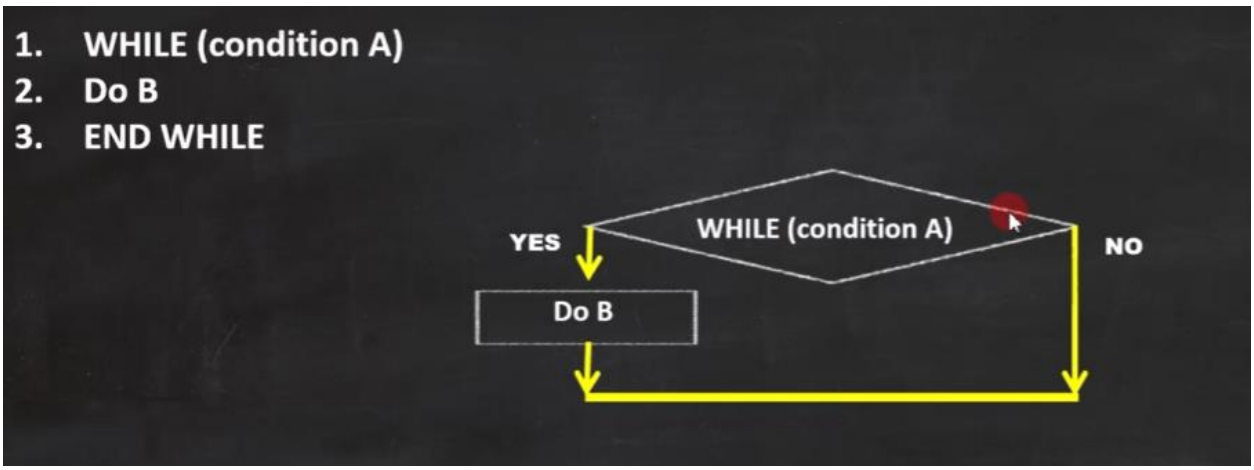
1. Read P
2. Read Q
3. If p+q > 100 then
4. Print “Large”
5. End if
6. If p > 50 then
7. Print “pLarge”
8. End if

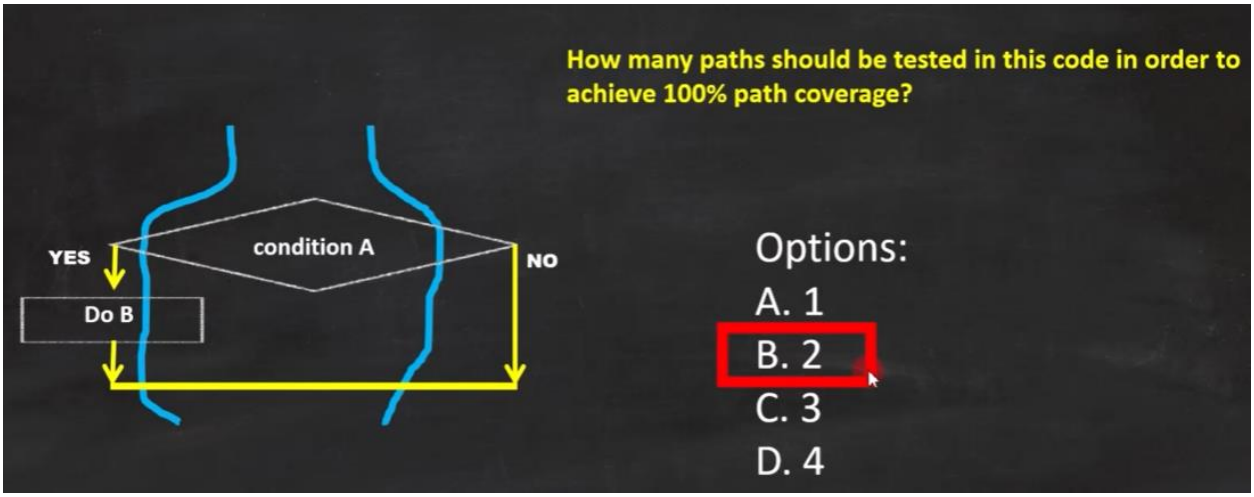


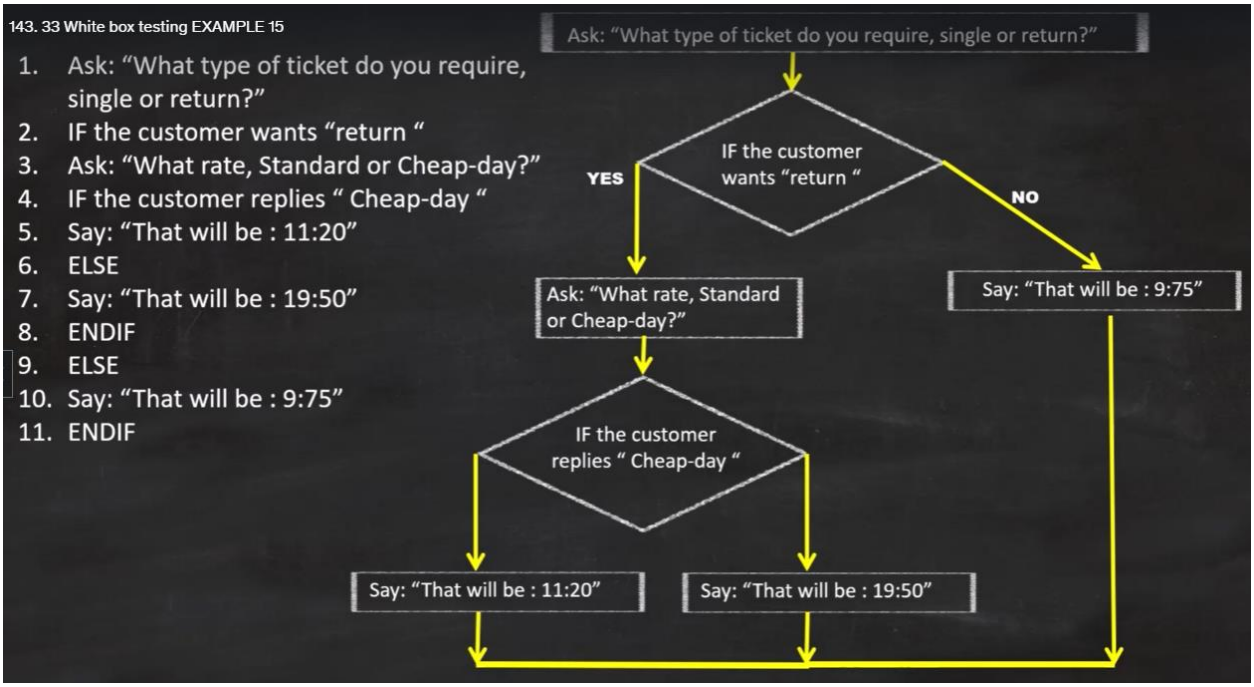

Read P
Read Q
p+q > 100
YES
No
Print "Large"
p > 50
Print "pLarge"
Options:
A. Statement coverage is 2, Branch Coverage is 2
B. Statement coverage is 3 and branch coverage is 2
C. Statement coverage is 1 and branch coverage is 2
D. Statement Coverage is 4 and Branch coverage is 2

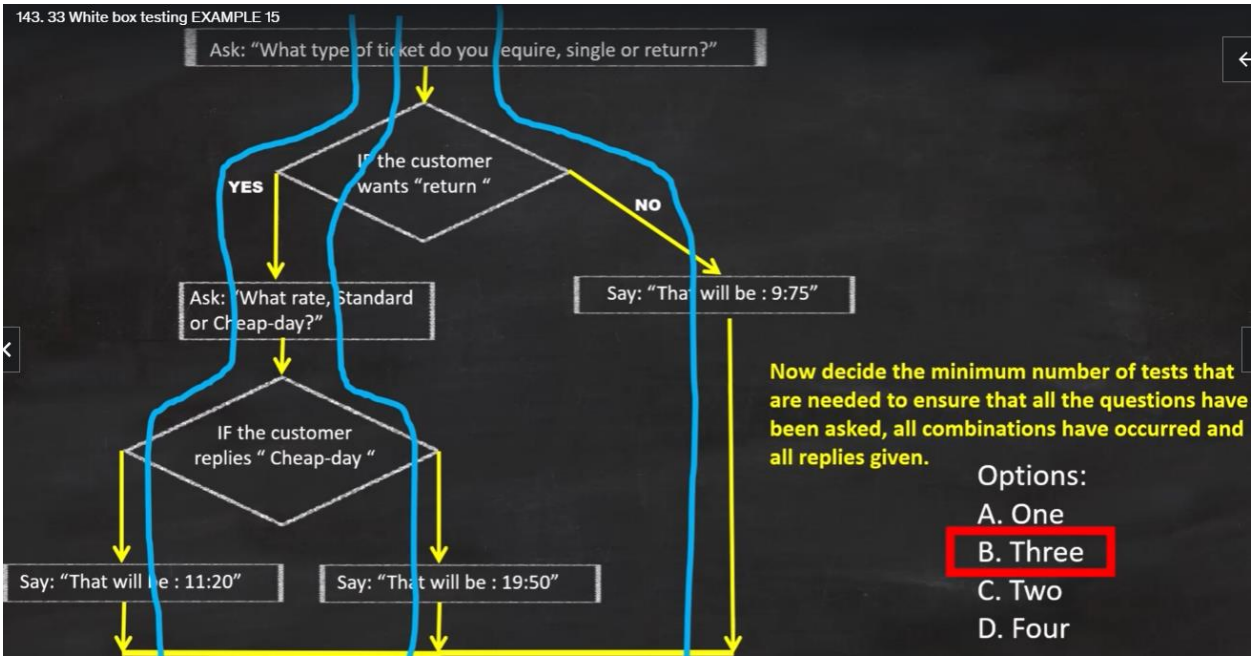

141. 33 White box testing EXAMPLE 13
1. If(x>y) x=x+1;
2. else y=y+1;
3. while(x>y)
4. {
5. y=x*y; x=x+1;
6. }
If(x>y)
YES
NO
x=x+1;
y=y+1
while(x>y)
YES
NO
y=x*y; x=x+1;

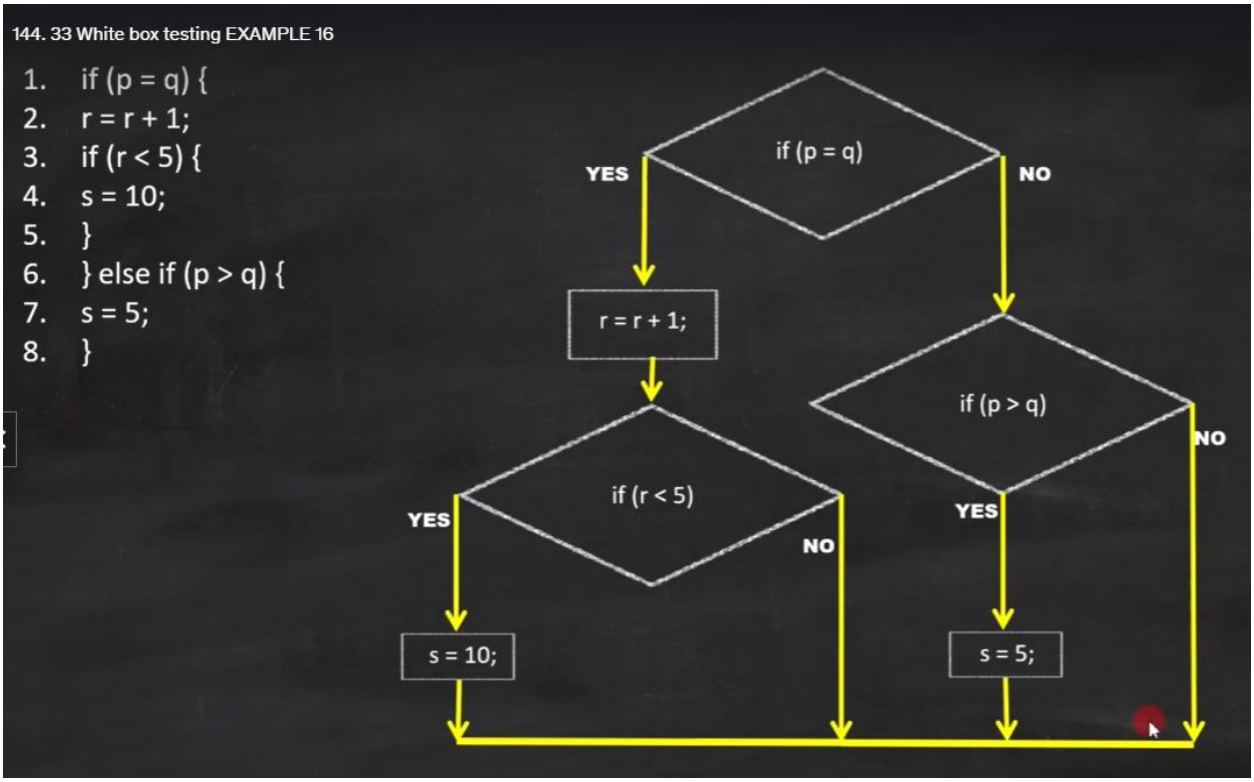

What is the smallest number of test cases required to Provide 100% branch coverage?
If(x>y)
YES
NO
x=x+1;
y=y+1
while(x>y)
YES
NO
y=x*y; x=x+1;
Options:
A. 1
B. 2
C. 3
D. 4

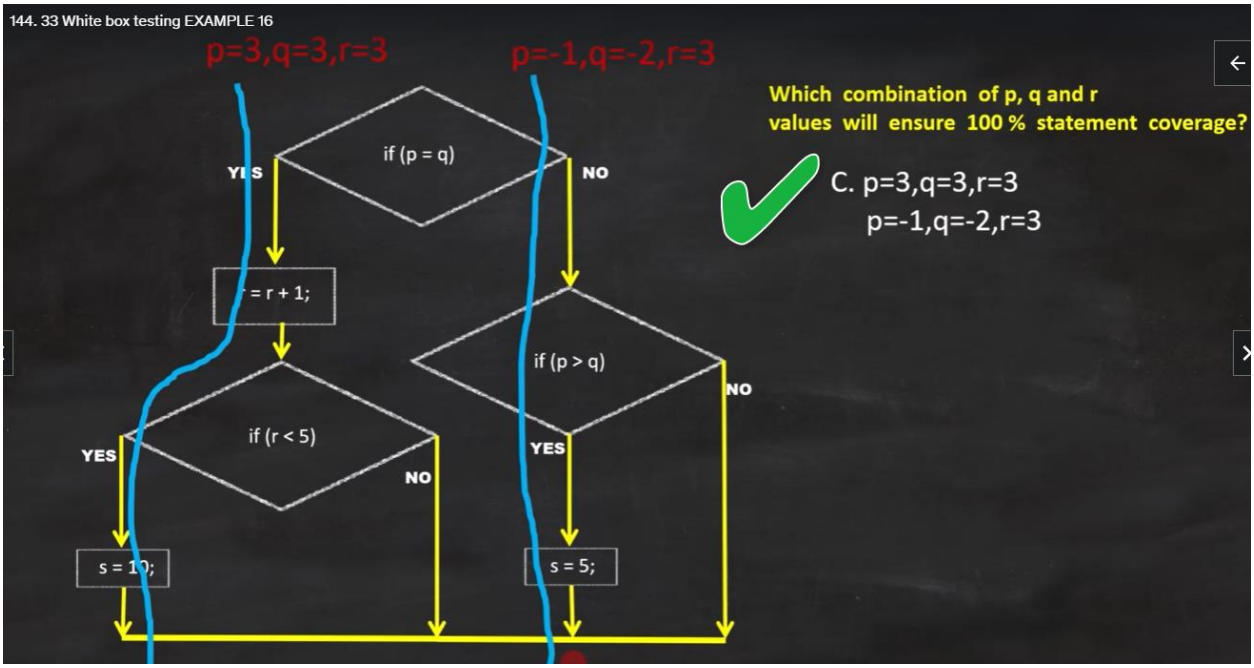

1. WHILE (condition A)
2. Do B
3. END WHILE
WHILE (condition A)
YES
NO
Do B

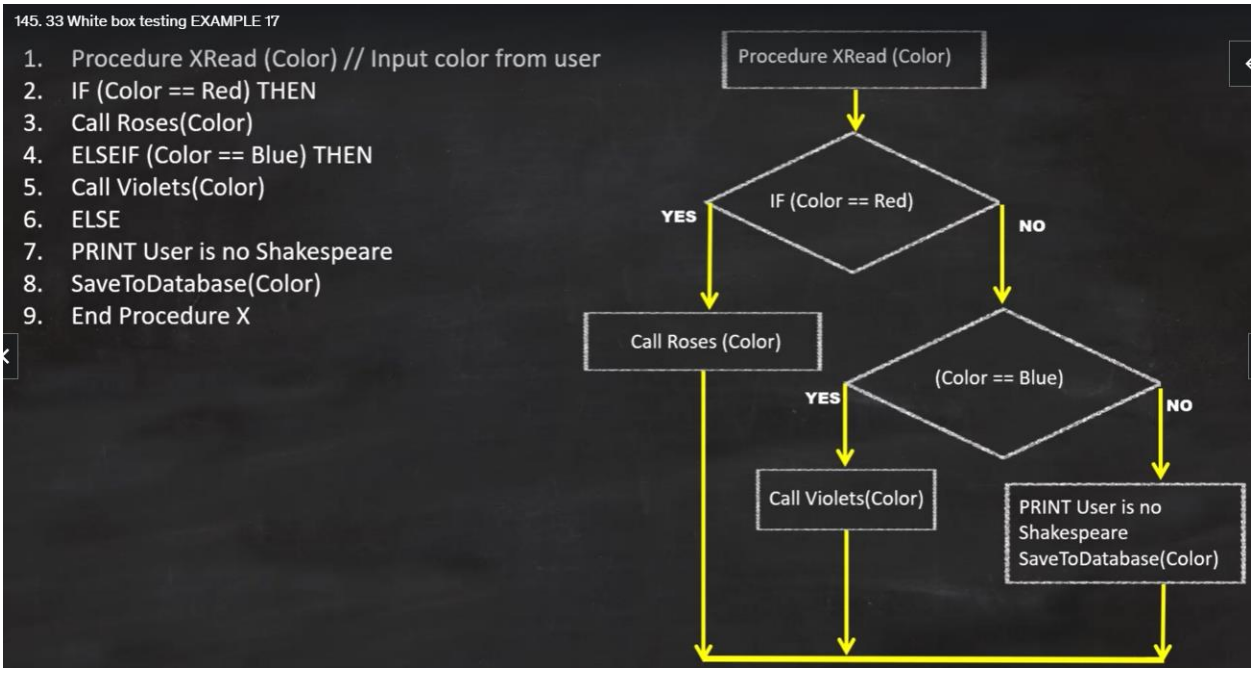

How many paths should be tested in this code in order to achieve 100% path coverage?
condition A
YES
NO
Do B
Options:
A. 1
B. 2
C. 3
D. 4

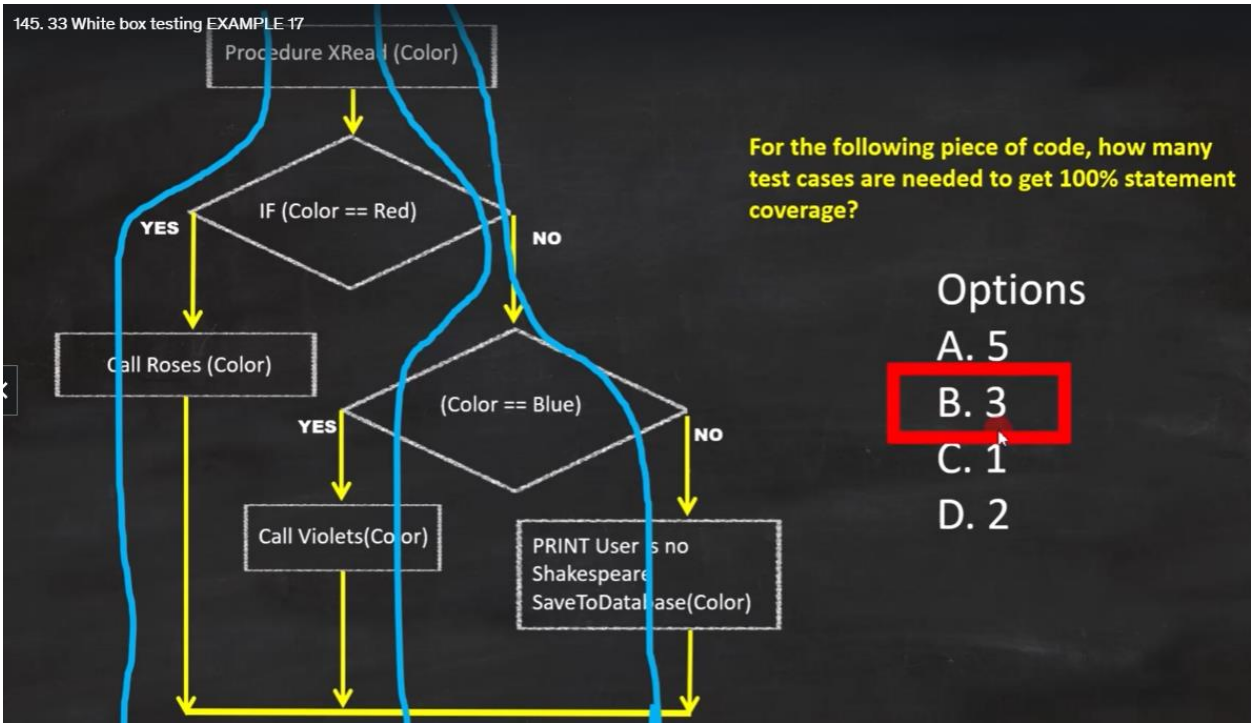

143. 33 White box testing EXAMPLE 15
1. Ask: "What type of ticket do you require, single or return?"
2. IF the customer wants "return "
3. Ask: "What rate, Standard or Cheap-day?"
4. IF the customer replies " Cheap-day "
5. Say: "That will be : 11:20"
6. ELSE
7. Say: "That will be : 19:50"
8. ENDIF
9. ELSE
10. Say: "That will be : 9:75"
11. ENDIF
Ask: "What type of ticket do you require, single or return?"
IF the customer wants "return "
YES
NO
Ask: "What rate, Standard or Cheap-day?"
Say: "That will be : 9:75"
IF the customer replies " Cheap-day "
Say: "That will be : 11:20"
Say: "That will be : 19:50"


143. 33 White box testing EXAMPLE 15
Ask: "What type of ticket do you require, single or return?"
IF the customer wants "return "
YES
NO
Ask: "What rate, Standard or Cheap-day?"
Say: "That will be : 9:75"
IF the customer replies " Cheap-day "
Say: "That will be : 11:20"
Say: "That will be : 19:50"
Now decide the minimum number of tests that are needed to ensure that all the questions have been asked, all combinations have occurred and all replies given.
Options:
A. One
B. Three
C. Two
D. Four


144. 33 White box testing EXAMPLE 16
1. if (p = q) {
2. r = r + 1;
3. if (r < 5) {
4. s = 10;
5. }
6. } else if (p > q) {
7. s = 5;
8. }
if (p = q)
YES
NO
r = r + 1;
if (p > q)
NO
if (r < 5)
YES
YES
NO
s = 10;
s = 5;


144. 33 White box testing EXAMPLE 16
p=3,q=3,r=3
p=-1,q=-2,r=3
Which combination of p, q and r values will ensure 100 % statement coverage?
C. p=3,q=3,r=3
p=-1,q=-2,r=3
if (p = q)
YES
NO
= r + 1;
if (p > q)
NO
if (r < 5)
YES
YES
NO
s = 10;
s = 5;


145. 33 White box testing EXAMPLE 17
1. Procedure XRead (Color) // Input color from user
2. IF (Color == Red) THEN
3. Call Roses(Color)
4. ELSEIF (Color == Blue) THEN
5. Call Violets(Color)
6. ELSE
7. PRINT User is no Shakespeare
8. SaveToDatabase(Color)
9. End Procedure X
Procedure XRead (Color)
IF (Color == Red)
YES
NO
Call Roses (Color)
(Color == Blue)
YES
NO
Call Violets(Color)
PRINT User is no Shakespeare SaveToDatabase(Color)


145. 33 White box testing EXAMPLE 17
Procedure XRead (Color)
IF (Color == Red)
YES
NO
Call Roses (Color)
(Color == Blue)
YES
NO
Call Violets(Color)
PRINT User is no Shakespeare SaveToDatabase(Color)
For the following piece of code, how many test cases are needed to get 100% statement coverage?
Options
A. 5
B. 3
C. 1
D. 2

If you are flying with an economy ticket, there is a possibility that you may get upgraded to business class, especially if you hold a gold card in the airline's frequent flier program. If you don't hold a gold card, there is a possibility that you will get 'bumped' off the flight if it is full and you check in late. This is shown in following Figure. Note that each box (i.e. statement) has been numbered.

Three tests have been run:

Test 1: Gold card holder who gets upgraded to business class

Test 2: Non-gold card holder who stays in economy

Test 3: A person who is bumped from the flight

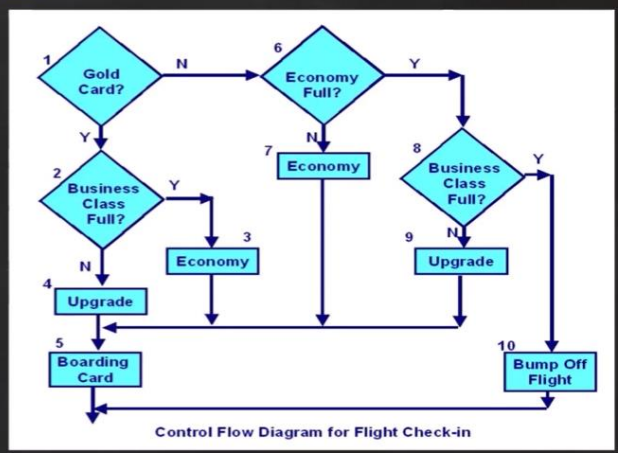
What is the statement coverage of these three tests?

A. 60%

B. 70%

C. 80%

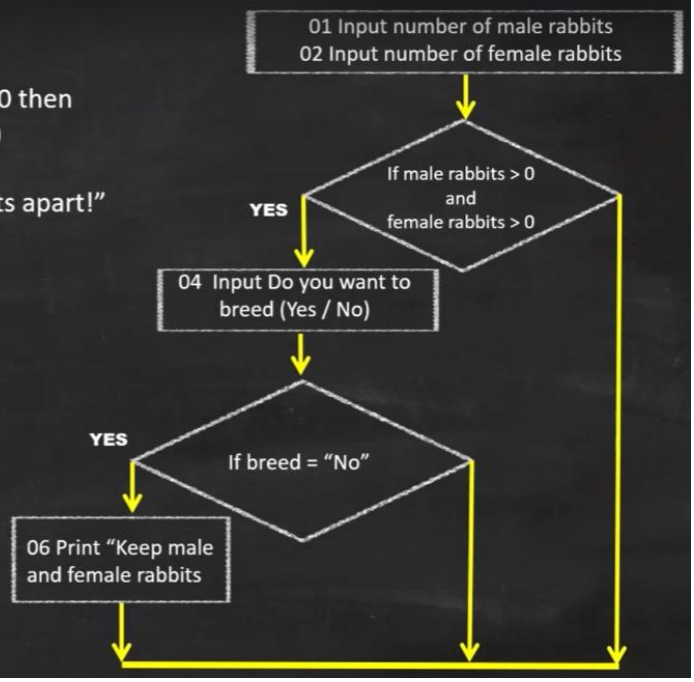
D. 90%



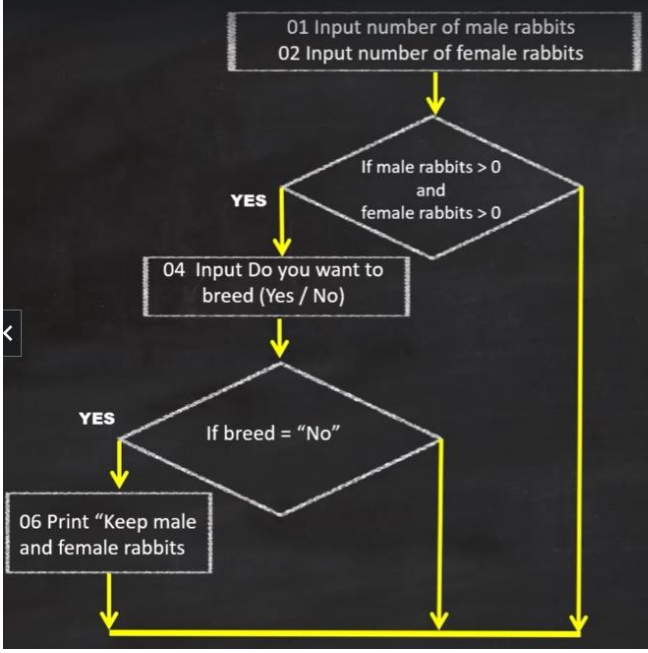
Control Flow Diagram for Flight Check-in

147. 33 White box testing EXAMPLE 19

- 01 Input number of male rabbits
- 02 Input number of female rabbits
- 03 If male rabbits > 0 and female rabbits > 0 then
- 04 Input Do you want to breed (Yes / No)
- 05 If breed = "No"
- 06 Print "Keep male and female rabbits apart!"
- 07 End if
- 08 End If



147. 33 White box testing EXAMPLE 19

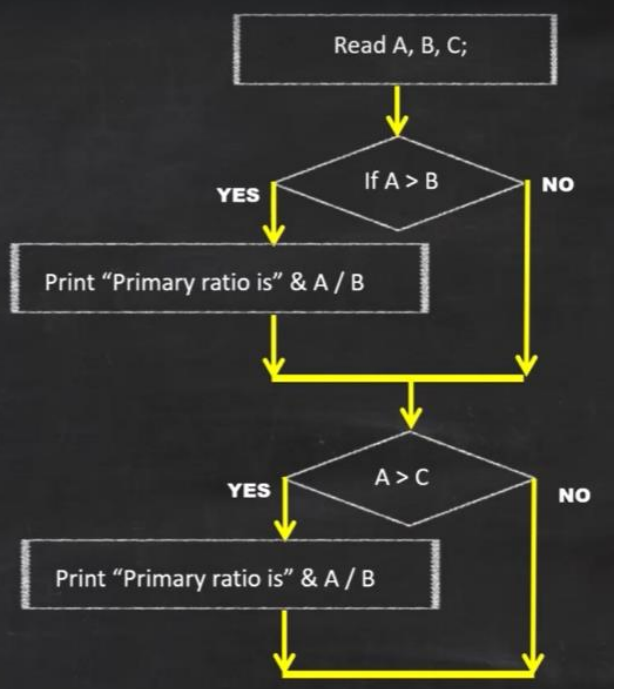


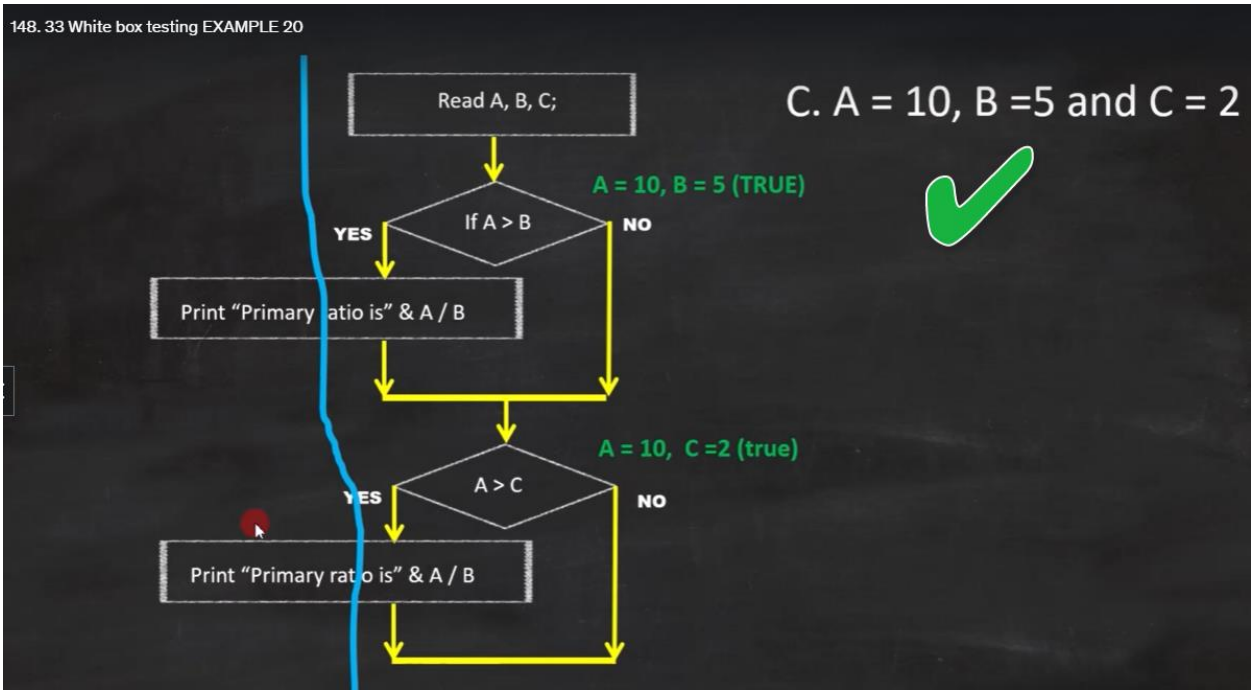
Which of the following test cases will ensure that statement "06" is executed?

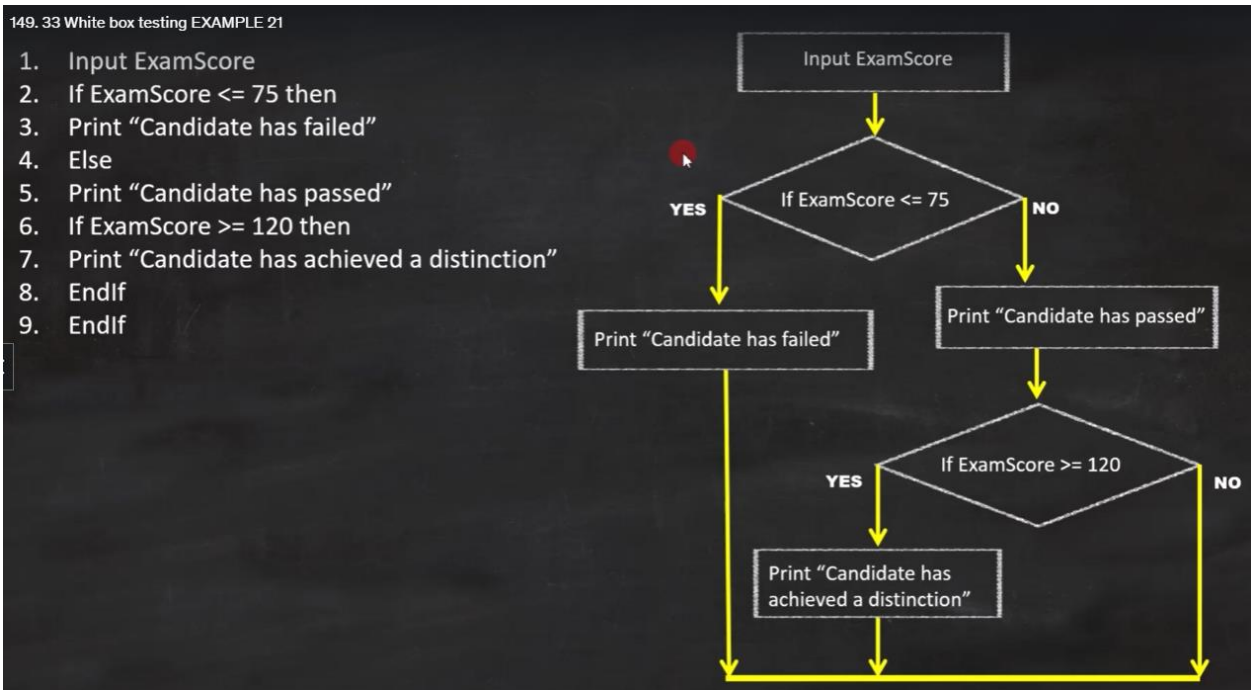
A. male rabbits = 1, female rabbits = 1, breed = "yes".
B. male rabbits = 1, female rabbits = 1, breed = "no".
C. male rabbits = 1, female rabbits = 2, breed = "yes".
D. male rabbits = 1, female rabbits = 0, breed = "no".

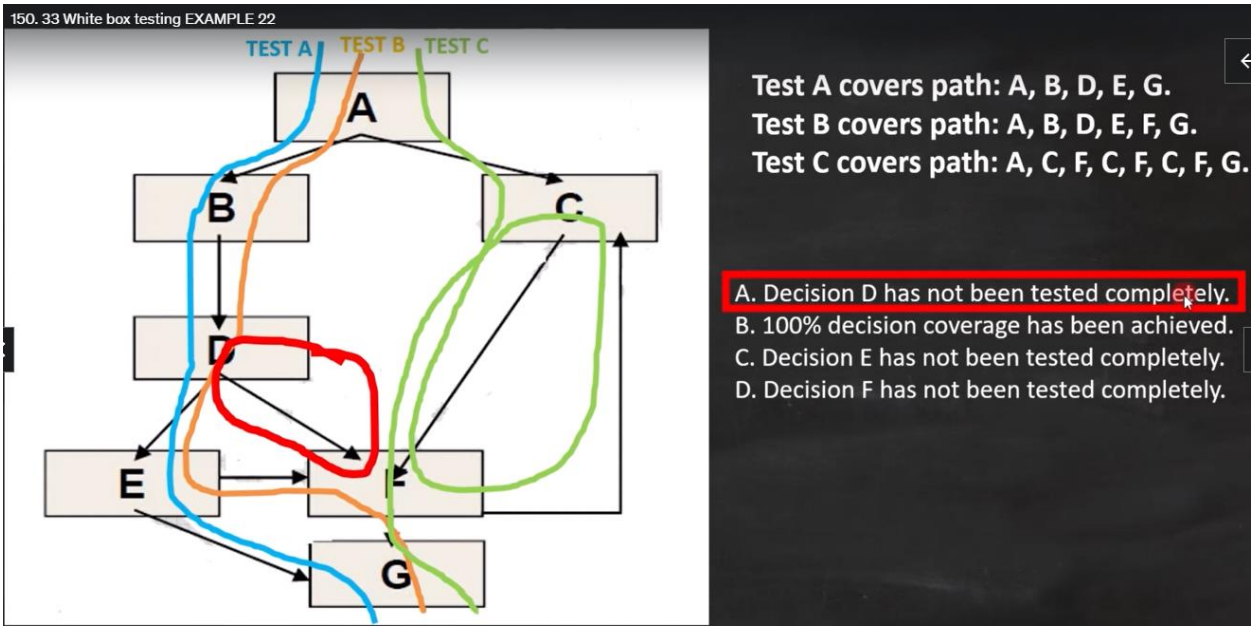
148. 33 White box testing EXAMPLE 20

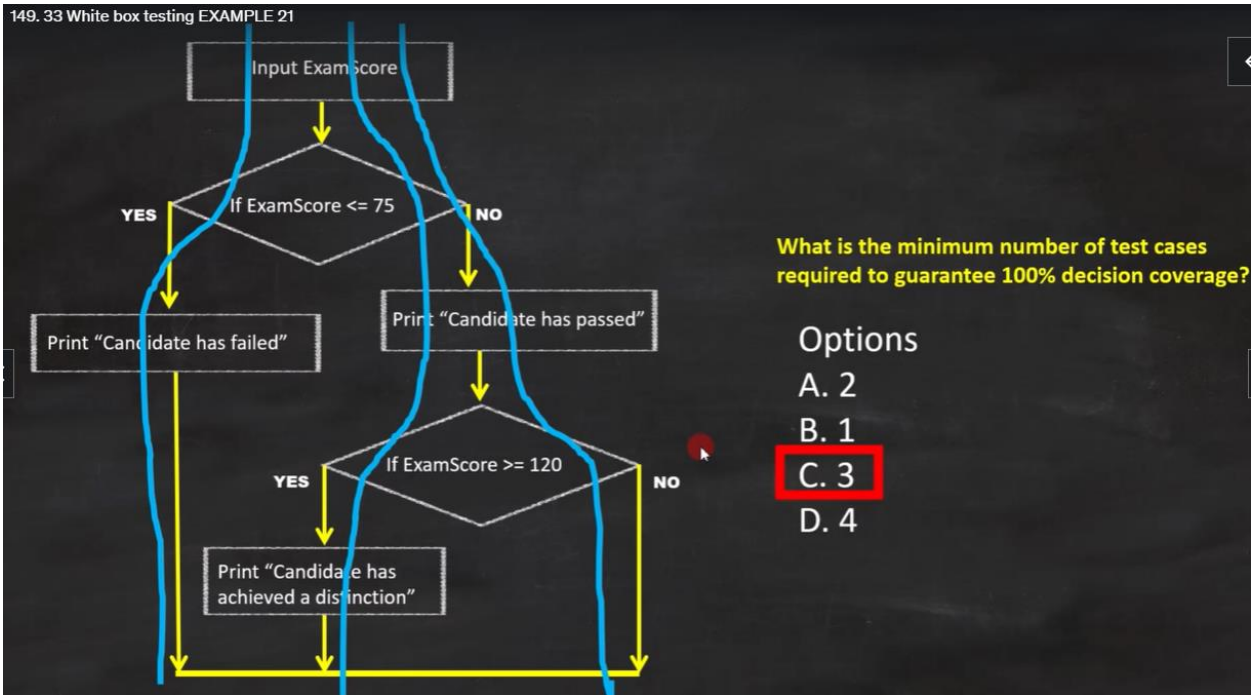
1. Read A, B, C;
2. If A > B then
3. Print "Primary ratio is" & A / B;
4. End If
5. If A > C then
6. Print "Secondary ration is" & A / C;
7. End If.

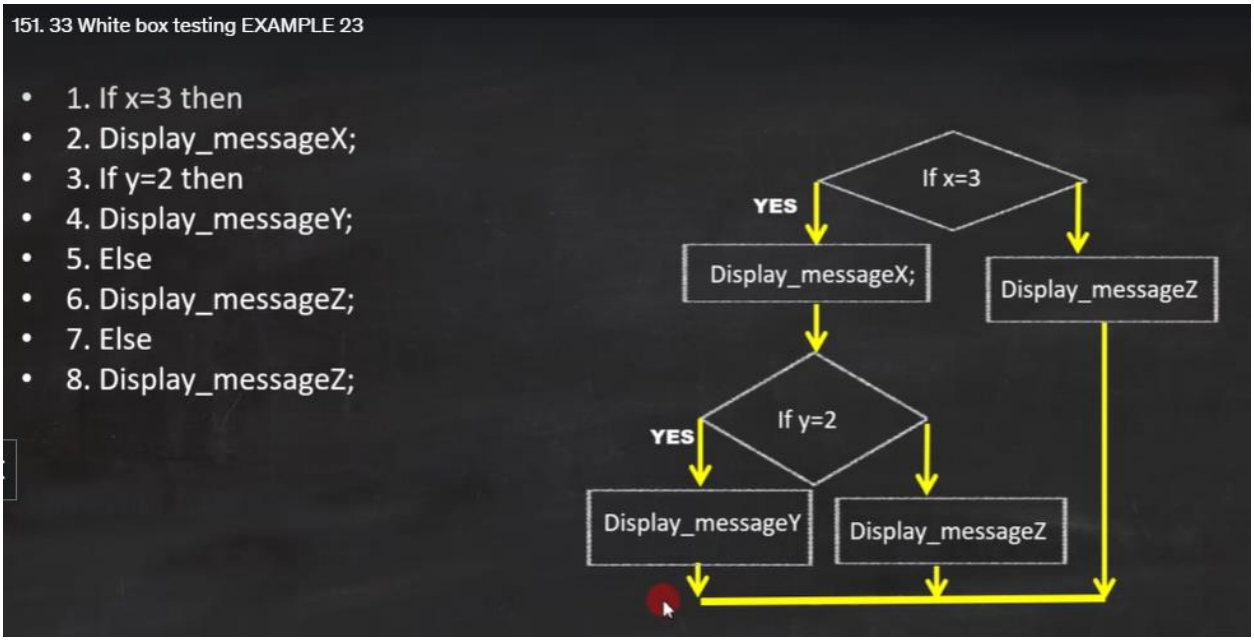


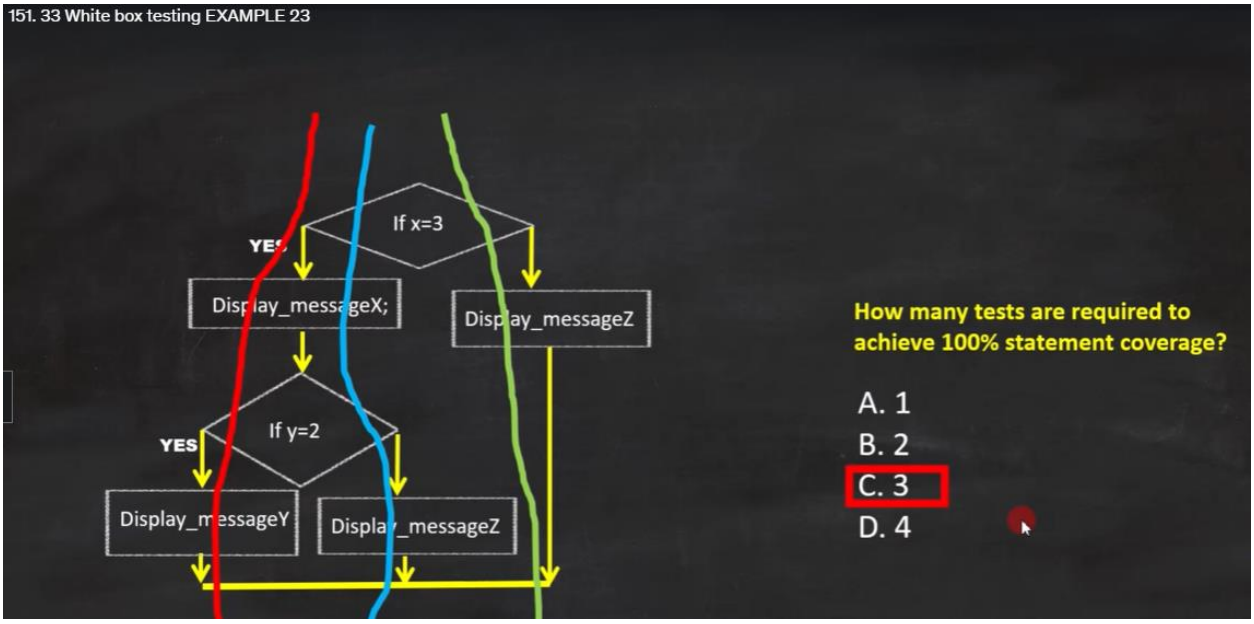

148. 33 White box testing EXAMPLE 20
Read A, B, C;
C. A = 10, B =5 and C = 2
A = 10, B = 5 (TRUE)
If A > B
YES
NO
Print "Primary atio is" & A / B
A = 10, C =2 (true)
A > C
YES
NO
Print "Primary rat o is" & A / B

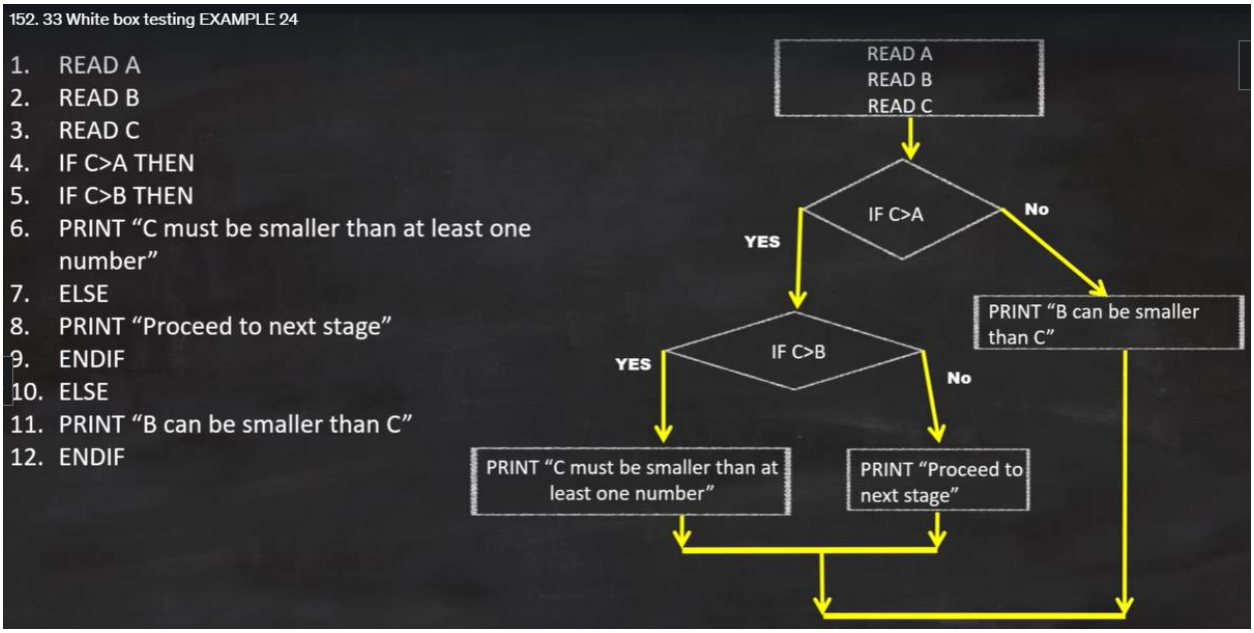

149. 33 White box testing EXAMPLE 21
1. Input ExamScore
2. If ExamScore <= 75 then
3. Print "Candidate has failed"
4. Else
5. Print "Candidate has passed"
6. If ExamScore >= 120 then
7. Print "Candidate has achieved a distinction"
8. EndIf
9. EndIf
Input ExamScore
If ExamScore <= 75
YES
NO
Print "Candidate has failed"
Print "Candidate has passed"
If ExamScore >= 120
YES
NO
Print "Candidate has achieved a distinction"

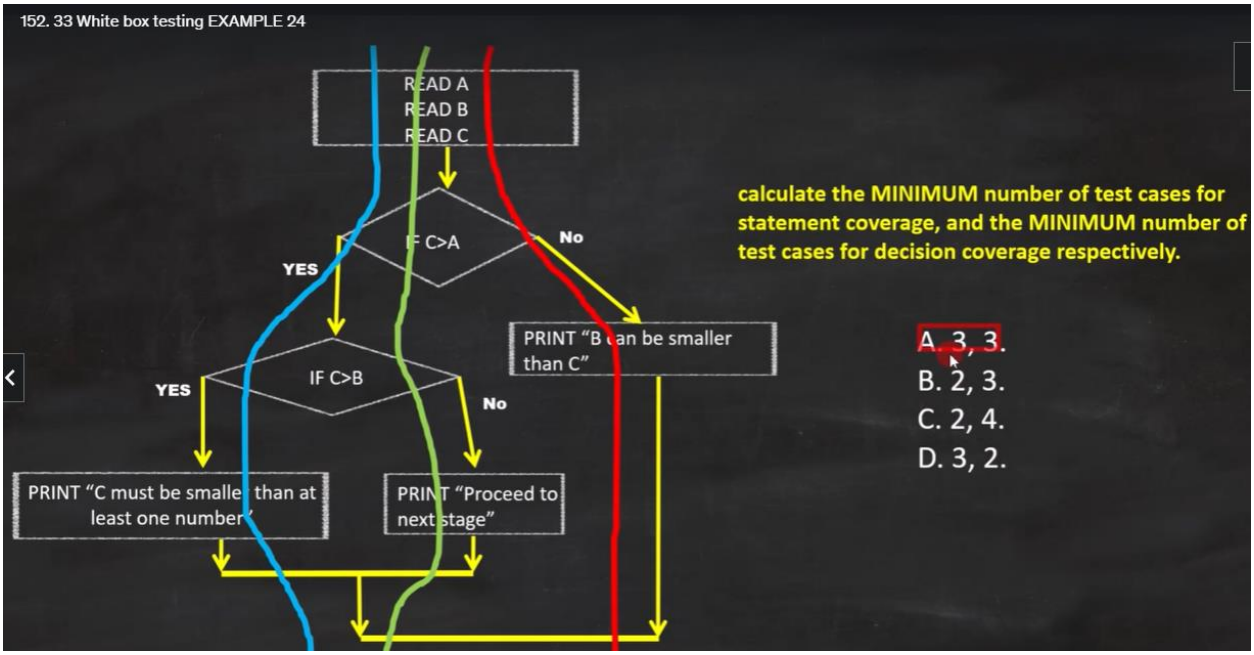

150. 33 White box testing EXAMPLE 22
TEST A
TEST B
TEST C
A
B
C
D
E
F
G
Test A covers path: A, B, D, E, G.
Test B covers path: A, B, D, E, F, G.
Test C covers path: A, C, F, C, F, C, F, G.
A. Decision D has not been tested completely.
B. 100% decision coverage has been achieved.
C. Decision E has not been tested completely.
D. Decision F has not been tested completely.


149. 33 White box testing EXAMPLE 21
Input ExamScore
If ExamScore <= 75
YES
NO
Print "Candidate has failed"
Print "Candidate has passed"
If ExamScore >= 120
YES
NO
Print "Candidate has achieved a distinction"
What is the minimum number of test cases required to guarantee 100% decision coverage?
Options
A. 2
B. 1
C. 3
D. 4


151. 33 White box testing EXAMPLE 23
1. If x=3 then
2. Display_messageX;
3. If y=2 then
4. Display_messageY;
5. Else
6. Display_messageZ;
7. Else
8. Display_messageZ;
If x=3
YES
Display_messageX;
Display_messageZ
If y=2
YES
Display_messageY
Display_messageZ


151. 33 White box testing EXAMPLE 23
If x=3
YES
Display_messageX;
Display_messageZ
If y=2
YES
Display_messageY
Display_messageZ
How many tests are required to achieve 100% statement coverage?
A. 1
B. 2
C. 3
D. 4

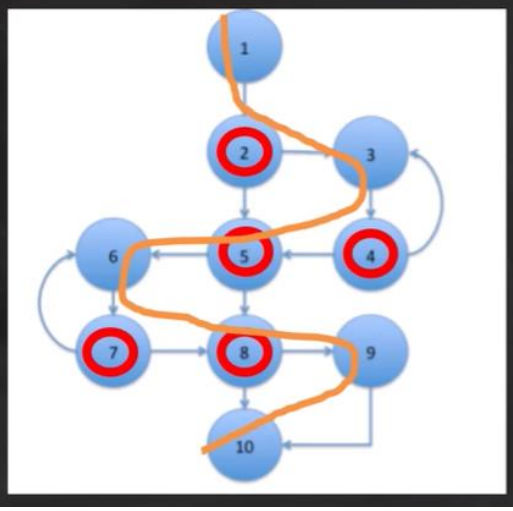

152. 33 White box testing EXAMPLE 24
1. READ A
2. READ B
3. READ C
4. IF C>A THEN
5. IF C>B THEN
6. PRINT "C must be smaller than at least one number"
7. ELSE
8. PRINT "Proceed to next stage"
9. ENDIF
10. ELSE
11. PRINT "B can be smaller than C"
12. ENDIF
READ A
READ B
READ C
IF C>A
YES
No
PRINT "B can be smaller than C"
IF C>B
YES
No
PRINT "C must be smaller than at least one number"
PRINT "Proceed to next stage"


152. 33 White box testing EXAMPLE 24
READ A
READ B
READ C
IF C>A
YES
No
PRINT "B can be smaller than C"
IF C>B
YES
No
PRINT "C must be smaller than at least one number"
PRINT "Proceed to next stage"
calculate the MINIMUM number of test cases for statement coverage, and the MINIMUM number of test cases for decision coverage respectively.
A. 3, 3.
B. 2, 3.
C. 2, 4.
D. 3, 2.

Xxxxx

156. 33 White box testing EXAMPLE 29

You have run one test case: 1-2-3-4-5-6-7-8-9-10



Options:

a. 50%

b. 80%

c. 90%

d. 100%

ISTQB MASTER CLASS BY AJIT PANIGRAHI

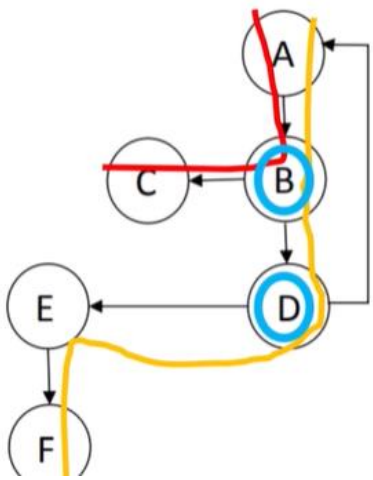
Test 1 : A-B-D-E-F

Test 2: A-B-C

Percentage of decision covered $= \frac{number\ of\ decision\ covered}{total\ number\ of\ decision} * 100$

$= \frac{3}{4} * 100$

$= 75\%$

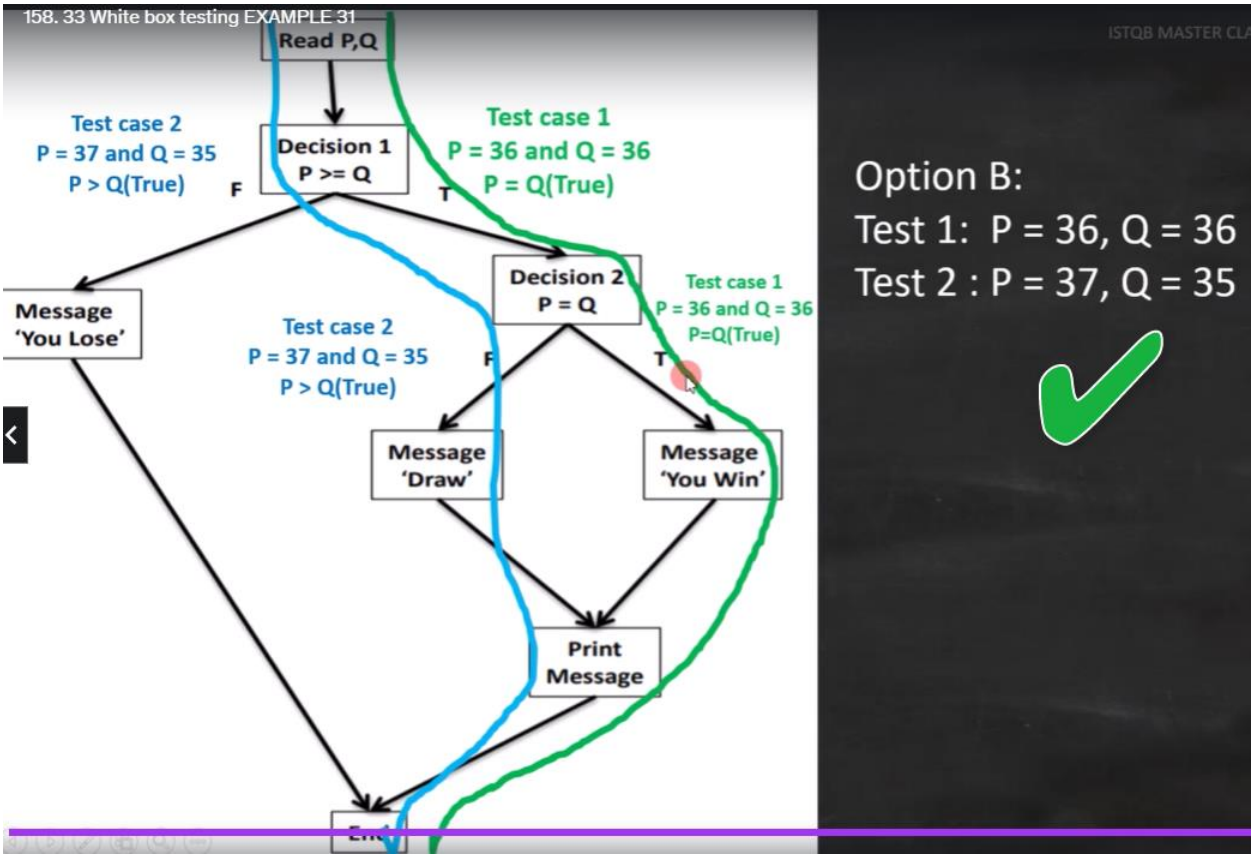


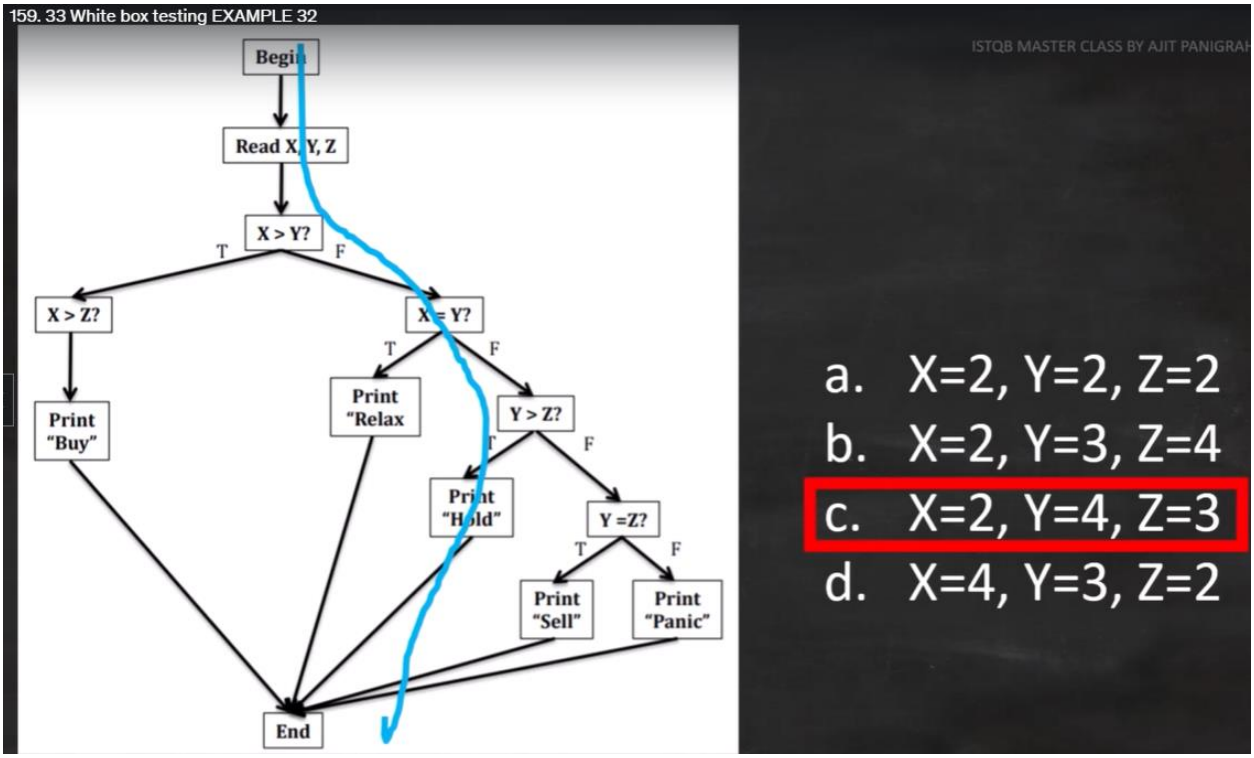
a. 33% statement, 100% decision

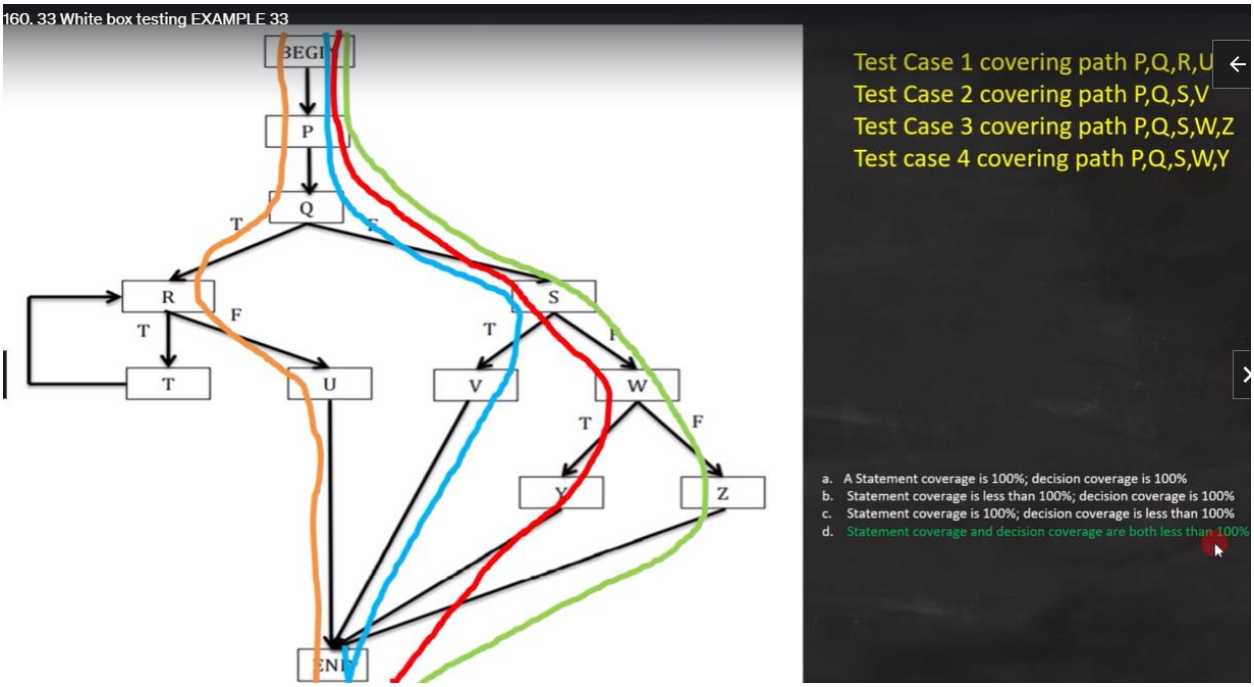
b. 50% statement, 50% decision

c. 100% statement, 75% decision

d. 100% statement, 100% decision


158. 33 White box testing EXAMPLE 31
Read P,Q
Decision 1
P >= Q
Test case 2
P = 37 and Q = 35
P > Q(True)
Test case 1
P = 36 and Q = 36
P = Q(True)
F
T
Message
'You Lose'
Decision 2
P = Q
Test case 1
P = 36 and Q = 36
P=Q(True)
Test case 2
P = 37 and Q = 35
P > Q(True)
Message
'Draw'
Message
'You Win'
Print
Message
Option B:
Test 1: P = 36, Q = 36
Test 2 : P = 37, Q = 35


159. 33 White box testing EXAMPLE 32
ISTQB MASTER CLASS BY AJIT PANIGRAH
Begin
Read X, Y, Z
X > Y?
T
F
X > Z?
X = Y?
Print
"Buy"
Print
"Relax
Y > Z?
Print
"Hold"
Y =Z?
Print
"Sell"
Print
"Panic"
End
a. X=2, Y=2, Z=2
b. X=2, Y=3, Z=4
c. X=2, Y=4, Z=3
d. X=4, Y=3, Z=2


160. 33 White box testing EXAMPLE 33
BEGIN
P
Q
R
S
T
U
V
W
Y
Z
END
T
F
Test Case 1 covering path P,Q,R,U
Test Case 2 covering path P,Q,S,V
Test Case 3 covering path P,Q,S,W,Z
Test case 4 covering path P,Q,S,W,Y
a. A Statement coverage is 100%; decision coverage is 100%
b. Statement coverage is less than 100%; decision coverage is 100%
c. Statement coverage is 100%; decision coverage is less than 100%
d. Statement coverage and decision coverage are both less than 100%

ISTQB Foundation Level Certification		
TASKS	**TEST MANAGER**	**TESTER**
Execute tests, evaluate the results, and document deviations from expected results		2 ✓
Adapt planning based on test results and progress	4 ✓	
Create the detailed test execution schedule		6 ✓
Prepare and acquire test data		8 ✓
Develop the skills and careers of testers (e.g., through training plans, performance evaluations, coaching, etc.)	10 ✓	
Design and implement test cases and test procedures		12 ✓
By AfterSchool		

ISTQB Foundation Level Certification		
TASKS	**TEST MANAGER**	**TESTER**
Write and update the test plan(s)	2 ✓	
Review tests developed by others		4 ✓
Automate tests as needed (may be supported by a developer or a test automation expert)		6 ✓
Share testing perspectives with other project activities, such as integration planning	8 ✓	
Coordinate the test plan(s) with project managers, product owners, and others	10 ✓	
Use appropriate tools to facilitate the test process		12 ✓
Support the selection and implementation of tools to support the test process, including recommending the budget for tool selection	14 ✓	
Initiate the analysis, design, implementation, and execution of tests, monitor test progress and results, and check the status of exit criteria	16 ✓	
Analyze, review, and assess requirements, user stories and acceptance criteria, specifications, and models for testability (i.e., the test basis)		18 ✓
Introduce suitable metrics for measuring test progress and evaluating the quality of the testing and the product	20 ✓	
By AfterSchool		

ISTQB Foundation Level Certification		
TASKS	**TEST MANAGER**	**TESTER**
Develop or review a test policy and test strategy for the organization	2 ✓	
Review and contribute to test plans		4 ✓
Evaluate non-functional characteristics such as performance efficiency, reliability, usability, security, compatibility, and portability		6 ✓
Plan the test activities by considering the context, and understanding the test objectives and risks.	8 ✓	
Design, set up, and verify test environment(s), often coordinating with system administration and network management		10 ✓
Promote and advocate the testers, the test team, and the test profession within the organization	12 ✓	
Prepare and deliver test progress reports and test summary reports based on the information gathered	14 ✓	
Decide about the implementation of test environment(s)	16 ✓	
Identify and document test conditions, and capture traceability between test cases, test conditions, and the test basis		18 ✓
Support setting up the defect management system and adequate configuration management of testware	20 ✓	
By AfterSchool		

- **Analytical:** This type of test strategy is based on an analysis of some factor (e.g., requirement or risk). Risk-based testing is an example of an analytical approach, where tests are designed and prioritized based on the level of risk.
- **Model-Based:** In this type of test strategy, tests are designed based on some model of some required aspect of the product, such as a function, a business process, an internal structure, or a non-functional characteristic (e.g., reliability). Examples of such models include business process models, state models, and reliability growth models.
- **Methodical:** This type of test strategy relies on making systematic use of some predefined set of tests or test conditions, such as a taxonomy of common or likely types of failures, a list of important quality characteristics, or company-wide look-and-feel standards for mobile apps or web pages.
- **Process-compliant** (or standard-compliant): This type of test strategy involves analyzing, designing, and implementing tests based on external rules and standards, such as those specified by industry-specific standards, by process documentation, by the rigorous identification and use of the test basis, or by any process or standard imposed on or by the organization.
- **Directed** (or consultative): This type of test strategy is driven primarily by the advice, guidance, or instructions of stakeholders, business domain experts, or technology experts, who may be outside the test team or outside the organization itself.
- **Regression-averse:** This type of test strategy is motivated by a desire to avoid regression of existing capabilities. This test strategy includes reuse of existing testware (especially test cases and test data), extensive automation of regression tests, and standard test suites.
- **Reactive:** In this type of test strategy, testing is reactive to the component or system being tested, and the events occurring during test execution, rather than being pre-planned (as the preceding strategies are). Tests are designed and implemented, and may immediately be executed in response to knowledge gained from prior test results. Exploratory testing is a common technique employed in reactive strategies.

162. 5_1 Test Organization MEMORY MAP

ISTQB Foundation Level Certification

Order	Team
1	Independent testers from the business organization or user community, or with specializations in specific test types such as usability, security, performance, regulatory/compliance, or portability
2	Independent developers or testers within the development teams or the project team; this could be developers testing their colleagues' products
3	No independent testers; the only form of testing available is developers testing their own code
4	Independent testers external to the organization, either working on-site (in-house) or off-site
5	Independent test team or group within the organization, reporting to project management or executive management

164. 5_1_2_1 Tasks of a Test Manager

ISTQB Foundation Level Certification

- **Develop or review a test policy and test strategy for the organization**
- Plan the test activities by considering the context, and understanding the test objectives and risks. This may include selecting test approaches, estimating test time, effort and cost, acquiring resources, defining test levels and test cycles, and planning defect management
- **Write and update the test plan**
- Coordinate the test plan with project managers, product owners, and others
- **Share testing perspectives with other project activities, such as integration planning**
- Initiate the analysis, design, implementation, and execution of tests, monitor test progress and results, and check the status of exit criteria (or definition of done) and facilitate test completion activities
- **Prepare and deliver test progress reports and test summary reports based on the information gathered**
- Adapt planning based on test results and progress (sometimes documented in test progress reports, and/or in test summary reports for other testing already completed on the project) and take any actions necessary for test control
- **Support setting up the defect management system and adequate configuration management of testware**
- Introduce suitable metrics for measuring test progress and evaluating the quality of the testing and the product
- **Support the selection and implementation of tools to support the test process, including recommending the budget for tool selection (and possibly purchase and/or support), allocating time and effort for pilot projects, and providing continuing support in the use of the tool**
- Decide about the implementation of test environment
- **Promote and advocate the testers, the test team, and the test profession within the organization**
- Develop the skills and careers of testers (e.g., through training plans, performance evaluations, coaching, etc.)

164. 5_1_2_1 Tasks of a Test Manager

ISTQB Foundation Level Certification

TASKS	TRUE	FALS
Develop or review a test policy and test strategy for the organization	✓	
Review and contribute to test plans		✓
Evaluate non-functional characteristics such as performance efficiency, reliability, usability, security, compatibility, and portability		✓
Plan the test activities by considering the context, and understanding the test objectives and risks.	✓	
Design, set up, and verify test environment(s), often coordinating with system administration and network management		✓
Promote and advocate the testers, the test team, and the test profession within the organization	✓	
Prepare and deliver test progress reports and test summary reports based on the information gathered	✓	
Decide about the implementation of test environment(s)	✓	
Identify and document test conditions, and capture traceability between test cases, test conditions, and the test basis		✓
Support setting up the defect management system and adequate configuration management of testware	✓	

164. 5_1_2_1 Tasks of a Test Manager

ISTQB Foundation Level Certification

TASKS	TRUE	FALS
Write and update the test plan(s)	✓	
Review tests developed by others		✓
Automate tests as needed (may be supported by a developer or a test automation expert)		✓
Share testing perspectives with other project activities, such as integration planning	✓	
Coordinate the test plan(s) with project managers, product owners, and others	✓	
Use appropriate tools to facilitate the test process		✓
Support the selection and implementation of tools to support the test process, including recommending the budget for tool selection	✓	
Initiate the analysis, design, implementation, and execution of tests, monitor test progress and results, and check the status of exit criteria	✓	
Analyze, review, and assess requirements, user stories and acceptance criteria, specifications, and models for testability (i.e., the test basis)		✓
Introduce suitable metrics for measuring test progress and evaluating the quality of the testing and the product	✓	

164. 5_1_2_1 Tasks of a Test Manager

ISTQB Foundation Level Certification

TASKS	TRUE	FALS
Execute tests, evaluate the results, and document deviations from expected results		✓
Adapt planning based on test results and progress	✓	
Create the detailed test execution schedule		✓
Prepare and acquire test data		✓
Develop the skills and careers of testers (e.g., through training plans, performance evaluations, coaching, etc.)	✓	
Design and implement test cases and test procedures		✓

165. 5_1_2_2 Tasks of a Tester

ISTQB Foundation Level Certification

- **Review and contribute to test plans**
- Analyze, review, and assess requirements, user stories and acceptance criteria, specifications, and models for testability (i.e., the test basis)
  Identify and document test conditions, and capture traceability between test cases, test conditions, and the test basis
- **Design, set up, and verify test environment(s), often coordinating with system administration and network management**
- Design and implement test cases and test procedures
- **Prepare and acquire test data**
- Create the detailed test execution schedule
- **Execute tests, evaluate the results, and document deviations from expected results**
- Use appropriate tools to facilitate the test process ⍰ Automate tests as needed (may be supported by a developer or a test automation expert)
- **Evaluate non-functional characteristics such as performance efficiency, reliability, usability, security, compatibility, and portability**
- Review tests developed by others

TASKS	TEST MANAGER	TESTER
Develop or review a test policy and test strategy for the organization	✓	
Review and contribute to test plans		✓
Evaluate non-functional characteristics such as performance efficiency, reliability, usability, security, compatibility, and portability		✓
Plan the test activities by considering the context, and understanding the test objectives and risks.	✓	
Design, set up, and verify test environment(s), often coordinating with system administration and network management		✓
Promote and advocate the testers, the test team, and the test profession within the organization	✓	
Prepare and deliver test progress reports and test summary reports based on the information gathered	✓	
Decide about the implementation of test environment(s)	✓	
Identify and document test conditions, and capture traceability between test cases, test conditions, and the test basis		✓
Support setting up the defect management system and adequate configuration management of testware	✓	

TASKS	TEST MANAGER	TESTER
Write and update the test plan(s)	✓	
Review tests developed by others		✓
Automate tests as needed (may be supported by a developer or a test automation expert)		✓
Share testing perspectives with other project activities, such as integration planning	✓	
Coordinate the test plan(s) with project managers, product owners, and others	✓	
Use appropriate tools to facilitate the test process		✓
Support the selection and implementation of tools to support the test process, including recommending the budget for tool selection	✓	
Initiate the analysis, design, implementation, and execution of tests, monitor test progress and results, and check the status of exit criteria	✓	
Analyze, review, and assess requirements, user stories and acceptance criteria, specifications, and models for testability (i.e., the test basis)		✓
Introduce suitable metrics for measuring test progress and evaluating the quality of the testing and the product	✓	

TASKS 1	TEST MANAGER	TESTER
Develop or review a test policy and test strategy for the organization		
Review and contribute to test plans		✓
Evaluate non-functional characteristics such as performance efficiency, reliability, usability, security, compatibility, and portability		✓
Plan the test activities by considering the context, and understanding the test objectives and risks.	✓	
Design, set up, and verify test environment(s), often coordinating with system administration and network management		✓
Promote and advocate the testers, the test team, and the test profession within the organization	✓	
Prepare and deliver test progress reports and test summary reports based on the information gathered	✓	
Decide about the implementation of test environment(s)	✓	
Identify and document test conditions, and capture traceability between test cases, test conditions, and the test basis		✓
Support setting up the defect management system and adequate configuration management of testware	✓	

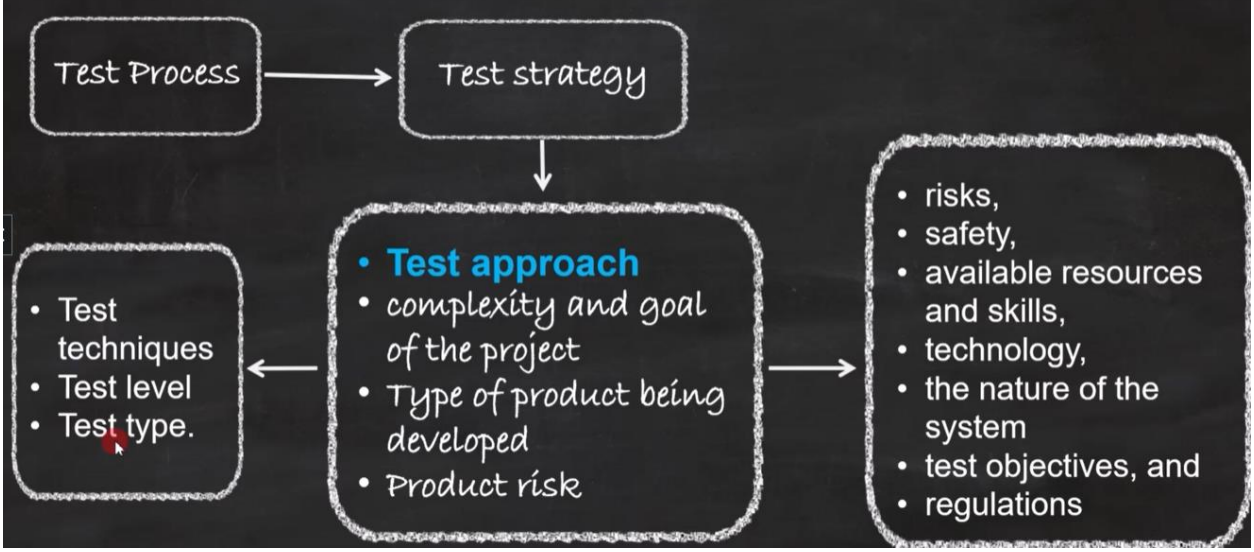
TASKS 3	TEST MANAGER	TESTER
Execute tests, evaluate the results, and document deviations from expected results		✓
Adapt planning based on test results and progress	✓	
Create the detailed test execution schedule		✓
Prepare and acquire test data		✓
Develop the skills and careers of testers (e.g., through training plans, performance evaluations, coaching, etc.)	✓	
Design and implement test cases and test procedures		✓

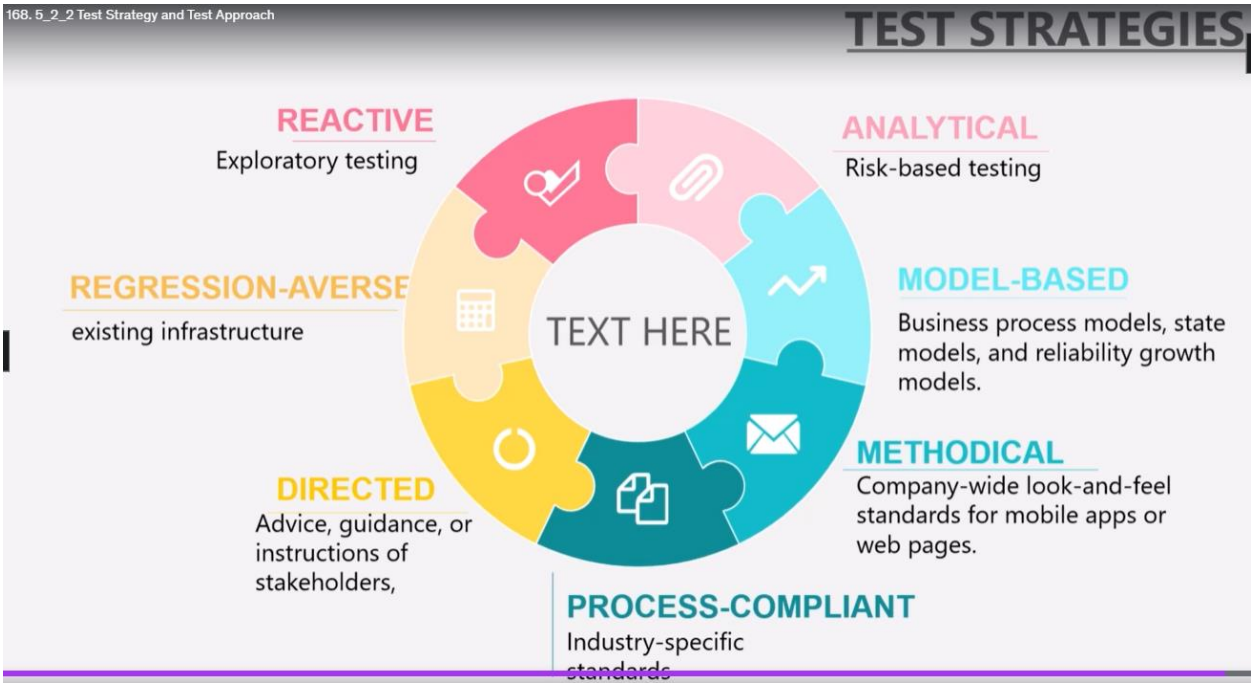
167. 5_2 Test Planning and Estimation

- Determining the scope, objectives, and risks of testing
- **Defining the overall approach of testing**
- Integrating and coordinating the test activities into the software lifecycle activities
- **Making decisions about what to test, the people and other resources required to perform the various test activities, and how test activities will be carried out**
- Scheduling of test analysis, design, implementation, execution, and evaluation activities, either on particular dates (e.g., in sequential development) or in the context of each iteration (e.g., in iterative development)
- **Selecting metrics for test monitoring and control**
- Budgeting for the test activities
- **Determining the level of detail and structure for test documentation (e.g., by providing templates or example documents)**

168. 5_2_2 Test Strategy and Test Approach

How test strategy, test process, test approach, test techniques, test levels, test type are interconnected?

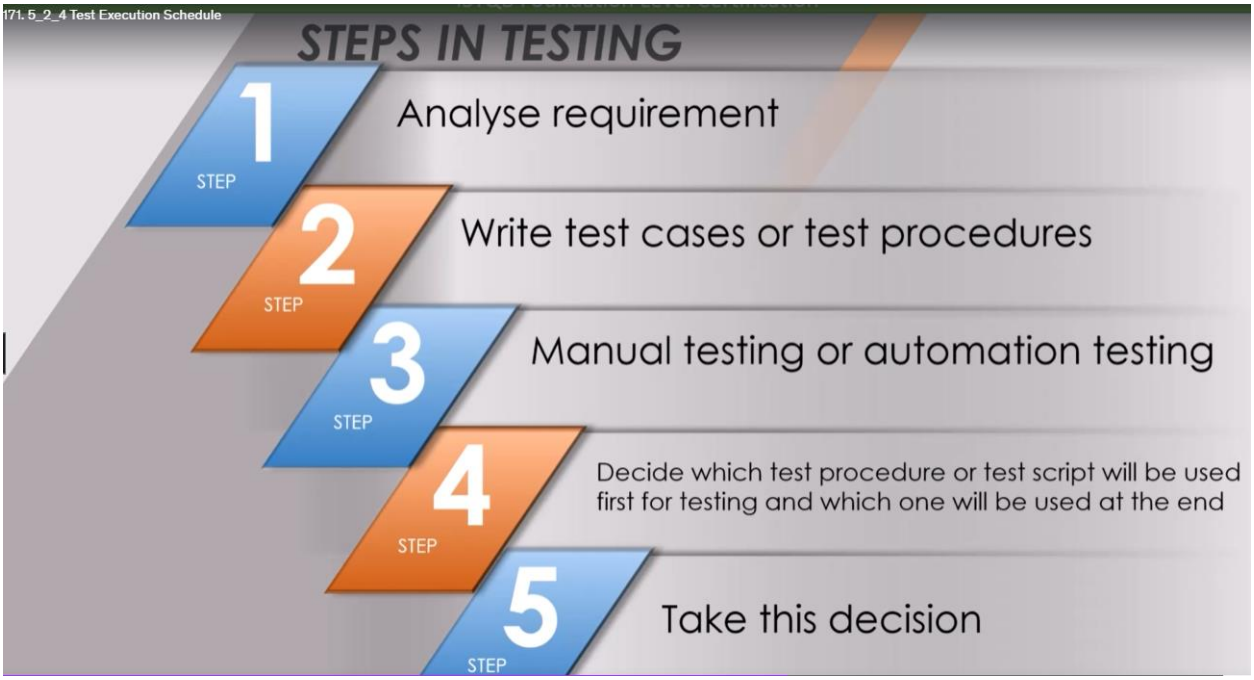




- **Analytical:** This type of test strategy is based on an analysis of some factor (e.g., requirement or risk). Risk-based testing is an example of an analytical approach, where tests are designed and prioritized based on the level of risk.
- **Model-Based:** In this type of test strategy, tests are designed based on some model of some required aspect of the product, such as a function, a business process, an internal structure, or a non-functional characteristic (e.g., reliability). Examples of such models include business process models, state models, and reliability growth models.
- **Methodical:** This type of test strategy relies on making systematic use of some predefined set of tests or test conditions, such as a taxonomy of common or likely types of failures, a list of

  important quality characteristics, or company-wide look-and-feel standards for mobile apps or web pages.
- **Process-compliant** (or standard-compliant): This type of test strategy involves analyzing, designing, and implementing tests based on external rules and standards, such as those specified by industry-specific standards, by process documentation, by the rigorous identification and use of the test basis, or by any process or standard imposed on or by the organization.
- **Directed** (or consultative): This type of test strategy is driven primarily by the advice, guidance, or instructions of stakeholders, business domain experts, or technology experts, who may be outside the test team or outside the organization itself.
- **Regression-averse:** This type of test strategy is motivated by a desire to avoid regression of existing capabilities. This test strategy includes reuse of existing testware (especially test cases and test data), extensive automation of regression tests, and standard test suites.
- **Reactive:** In this type of test strategy, testing is reactive to the component or system being tested, and the events occurring during test execution, rather than being pre-planned (as the preceding strategies are). Tests are designed and implemented, and may immediately be executed in response to knowledge gained from prior test results. Exploratory testing is a common technique employed in reactive strategies.

CRITERIA example	ENTRY	EXIT
Availability of testable requirements, user stories, and/or models	✓	
Availability of test items that have met the exit criteria for any prior test levels	✓	
Panned tests have been executed		✓
Availability of necessary test tools	✓	
Availability of test environment	✓	
A defined level of coverage (e.g., of requirements, user stories, acceptance criteria, risks, code) has been achieved		✓
The number of estimated remaining defects is sufficiently low		✓
The number of unresolved defects is within an agreed limit		✓
Availability of test data and other necessary resources	✓	
The evaluated levels of reliability, performance efficiency, usability and security		✓



172. 5_2_4_1 Test Execution Schedule Question 1

Test Case	Test Type	Risk Priority	Dependencies
1	Browse	2	None
2	Select	3	Browse
3	Select	2	Browse
4	Shopping Cart	1	Select
5	Shopping Cart	3	Select
6	Purchase	1	Shopping Cart
7	Refund	4	Purchase

Test case	4 (ShopCa)	6 (purchase)	1 (browse)	3 (select)	2 (select)	5 (Shp Ca)p-	7 (refund)
Prio.	1	1	2	2	3	3	4
Dep.	select	Shop Ca.	None	browse	browse	select	purchase

Test case	1 (browse)	2 (select)	5 (Shp Ca)	6 (purchase)	7 (refund)	1 (browse)	3 (select)	4 (ShopCa)	6 (purchase)	7 (refund)
Prio.	2	3	3	1	4	2	2	1	1	4
Dep.	none	browse	select	Shop Ca.	purchase	none	browse	select	Shop Ca.	purchase

a) 4, 6, 1, 3, 2, 5, 7
b) 1, 2, 5, 6, 7, 1, 3, 4, 6, 7
c) 1, 3, 4, 6, 1, 2, 5, 6, 7
d) 1, 3, 2, 4, 5, 6, 7

Test case	1 (browse)	3 (select)	4 (ShopCa)	6 (purchase)	1 (browse)	2 (select)	5 (Shp Ca)	6 (purchase)	7 (refund)
Prio.	2	2	1	1	2	3	3	1	4
Dep.	none	browse	select	Shop Ca.	none	browse	select	Shop Ca.	purchase

Test case	1 (browse)	3 (select)	2 (select)	4 (ShopCa)	5 (Shp Ca)	6 (purchase)	7 (refund)
Prio.	2	2	3	1	3	1	4
Dep.	none	browse	browse	select	select	Shop Ca.	purchase

173. 5_2_4_1 Test Execution Schedule Question 2

			Priority
1	Purchase Item	none	2
2	Receive Invoice	Test 1	3
3	Receive Goods	Test 1	2
4	Send Payment	Test 2	3
5	Report Payments	Test 4	1

a) 5, 1, 3, 2, 4
b) 1, 2, 4, 3, 5
c) 1, 3, 2, 4, 5
d) 3, 4, 5, 1, 2

Test case	5	1	3	2	4
Prio.	1	2	2	3	3
Dep.	Test 4	none	Test 1	Test 1	Test 2

Test case	1	2	4	3	5
Prio.	2	3	3	2	1
Dep.	none	Test 1	Test 2	Test 1	Test 4

Test case	1	3	2	4	5
Prio.	2	2	3	3	1
Dep.	none	Test 1	Test 1	Test 2	Test 4

Test case	3	4	5	1	2
Prio.	2	3	1	2	3
Dep.	Test 1	Test 2	Test 4	none	Test 1

The following test cases need to be run, but time is limited, and it is possible that not all will be completed before the end of the test window
The first activity is to run any re-tests, followed by the regression test script. Users have supplied their priority order to tests.
Which of the following gives an appropriate test execution schedule, taking account of the prioritization and other constraints?

#	Description	Priority	Note
a	Re-test defect no 52	Low	Re-test
b	Ability to amend transaction type	High	
c	Re-test defect no 26	High	Re-test
d	Run regression test script	Medium	Regression
e	Print monthly sales figures	Medium	
f	Add special invoice to previous month	Low	
g	Reprint selected previous sales figures	High	Must be run after item e
h	Account administrator able to amend any previous month's sales figures	Low	
i	Print year-to-date figures	Medium	

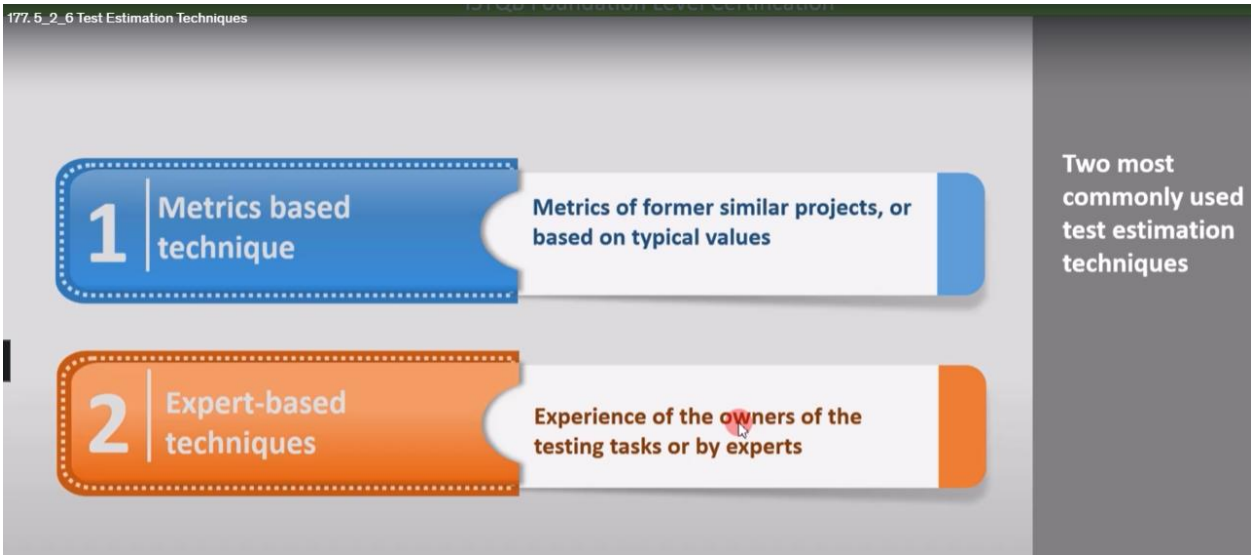
Priority 1 : re-tests in priority order
Priority 2 : regression tests
Priority 3 : other test cases.

a) b, c, g, d, e, i, a, f, h
b) a, c, d, b, g, e, i, f, h
c) c, a, d, b, e, g, i, h, f ✔
d) d, c, a, e, b, g, i, f, h

You have been given the following set of test cases to run. You have been instructed to run them in order by risk and to accomplish the testing as quickly as possible to provide feedback to the developers as soon as possible. Given this information, what is the best order in which to run these tests?

Test Case ID	Duration	Risk Priority	Dependency
1	30 mins	Low	6
2	10 mins	Medium	none
3	45 mins	High	1
4	30 mins	High	2
5	10 mins	Medium	4
6	15 mins	Low	2

a) 2, 4, 5, 6, 1, 3 ✔
~~b) 4, 3, 2, 5, 6, 1~~
~~c) 2, 5, 6, 4, 1, 3~~
~~d) 6, 1, 3, 2, 4, 5~~



177. 5_2_6 Test Estimation Techniques

ESTIMATION TECHNIQUE	AGILE DEVELOPMENT	SEQUENTIAL DEVELOPMENT
Metrics-based	BURNDOWN CHARTS	DEFECT REMOVAL MODEL
Expert-based	PLANNING POKER	WIDEBAND DELPHI

178. 5_3 Test Monitoring and Control

1 Test monitoring

The purpose of test monitoring is to gather information and provide feedback and visibility about test activities.

- To evaluate the progress of testing
- To check if exit criteria are met or not
- To verify coverage target is met or not

178. 5_3 Test Monitoring and Control

2 TEST CONTROL

Test control describes any guiding or corrective actions taken as a result of information and metrics gathered and (possibly) reported

- Re-prioritizing tests when an identified risk occurs (e.g., software delivered late)
- Changing the test schedule due to availability or unavailability of a test environment or other resources
- Re-evaluating whether a test item meets an entry or exit criterion due to rework

179. 5_3_1 Metrics Used in Testing

Defect information

- Defect density
- Defects found and fixed
- Failure rate
- Confirmation test results

DEFECT DENSITY

The number of defects per unit size of a work product

FAILURE RATE

The ratio of the number of failures of a given category to a given unit of measure

20 defects test condition

25 defects test cases

Planning : Number of test cases can fail = 20%

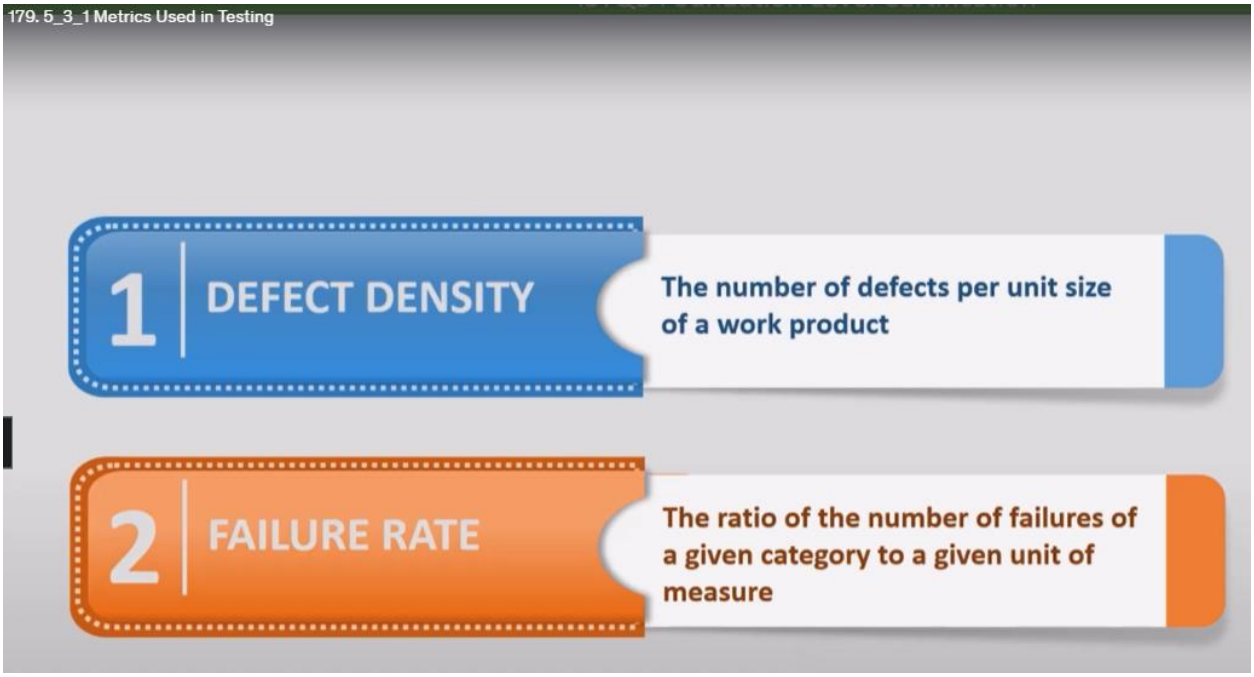
ACTUAL : Number of test cases can fail = 40%

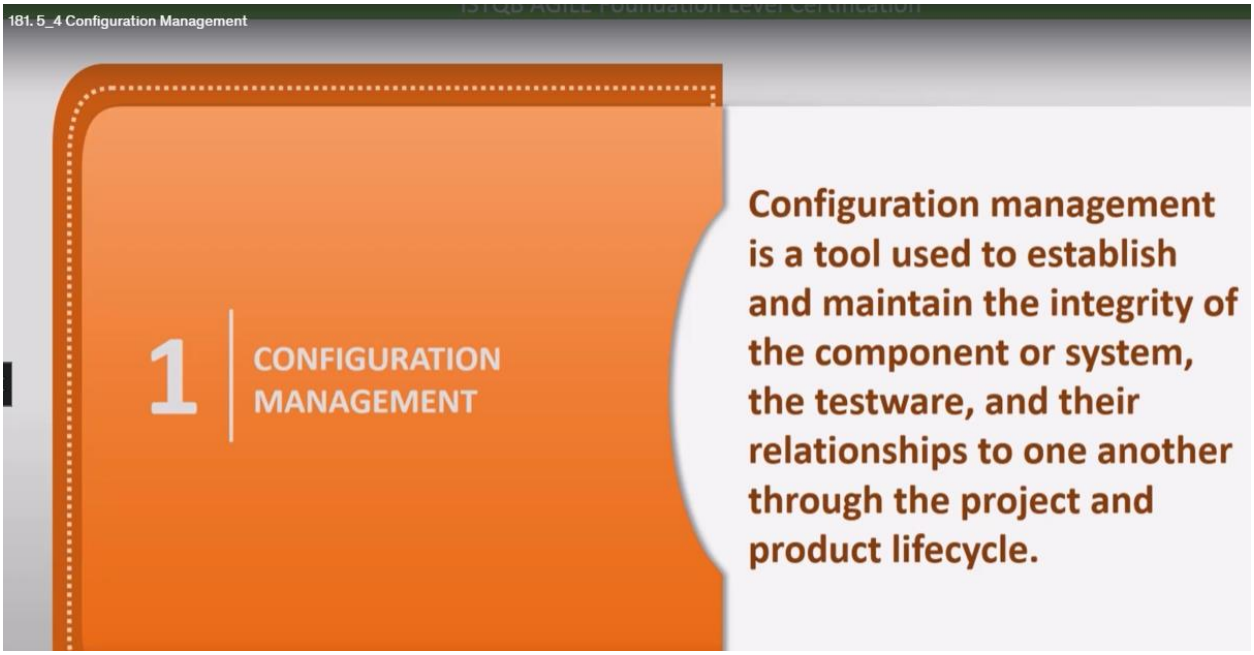
Failure rate : $\frac{40}{20} = 2$

179. 5_3_1 Metrics Used in Testing

SUMMARY

- Percentage of planned work done in test case preparation. for example, the percentage of planned test cases implemented
- Percentage of planned work done in test environment preparation
- Test case execution for example number of test cases run/not run, test cases passed/failed, and/or test conditions passed/failed
- Defect information for example 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


179. 5_3_1 Metrics Used in Testing
1
DEFECT DENSITY
The number of defects per unit size of a work product
2
FAILURE RATE
The ratio of the number of failures of a given category to a given unit of measure


181. 5_4 Configuration Management
1
CONFIGURATION MANAGEMENT
Configuration management is a tool used 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.

182. 5_5 Risks and Testing
1 RISK
the possibility of an event in the future which has negative consequences.
2 LIKELIHOOD
How frequently this can happen
3 IMPACT
What harm it will cause.

183. 5_5_2 Product and Project Risks
Product risk and project risk
PROJECT RISK
June-2017
Release 1
intermediate release
Oct-2019
Release 2
intermediate release
PRODUCT RISK
March 2020
Release 3
FINAL RELEASE
Timeline

183. 5_5_2 Product and Project Risks

Example of product risk

- Software might not perform its intended functions according to the specification
- Software might not perform its intended functions according to the user, customer, or stakeholder needs
- A system architecture may not adequately support some non-functional requirement
- A particular computation may be performed incorrectly in some circumstances
- A loop control structure may be coded incorrectly if this type of bug is there the delivered product may hang and this is a product risk
- Response-times may be inadequate for a high-performance transaction processing system
- User experience feedback might not meet product expectations if we deliver the defective product then will receive bad feedback and this is a product risk

183. 5_5_2 Product and Project Risks

Project issues

- Delays may occur in delivery, task completion, or satisfaction of exit criteria or definition of done
- Inaccurate estimates, reallocation of funds to higher priority projects, or general cost-cutting across the organization may result in inadequate funding
- Late changes may result in substantial re-work

183. 5_5_2 Product and Project Risks

Organizational issues

- Skills, training, and staff may not be sufficient
- Personnel issues may cause conflict and problems
- Users, business staff, or subject matter experts may not be available due to conflicting business priorities

183. 5_5_2 Product and Project Risks

Political issues

- Testers may not communicate their needs or the test results adequately
- Developers and/or testers may fail to follow up on information found in testing and reviews
- There may be an improper attitude toward, or expectations of, testing for example not appreciating the value of finding defects during testing

183. 5_5_2 Product and Project Risks

Technical issues

- Requirements may not be defined well enough
- The requirements may not be met, given existing constraints
- The test environment may not be ready on time
- Data conversion, migration planning, and their tool support may be late
- Weaknesses in the development process may impact the consistency or quality of project work products such as design, code, configuration, test data, and test cases
- Poor defect management and similar problems may result in accumulated defects and other technical debt

183. 5_5_2 Product and Project Risks

Supplier issues

- A third party may fail to deliver a necessary product or service or go bankrupt
- Contractual issues may cause problems to the project

184. 5_5_2 Product and Project Risks MEMORY MAP

RISKS - 2	PRODUCT RISK	PROJECT
Developers and/or testers may fail to follow up on information found in testing and reviews		
There may be an improper attitude toward, or expectations of, testing		
A particular computation may be performed incorrectly in some circumstances		
Requirements may not be defined well enough		
The requirements may not be met, given existing constraints		
The test environment may not be ready on time		
loop control structure may be coded incorrectly		
Data conversion, migration planning, and their tool support may be late		
Weaknesses in the development process may impact the consistency or quality of project work products such as design, code, configuration, test data, and test cases		
Poor defect management and similar problems may result in accumulated defects and other technical debt		

184. 5_5_2 Product and Project Risks MEMORY MAP

RISKS - 3	PRODUCT RISK	PROJECT
A third party may fail to deliver a necessary product or service, or go bankrupt		
User experience (UX) feedback might not meet product expectations		
Contractual issues may cause problems to the project		
Response-times may be inadequate for a high-performance transaction processing system		

185. 5_5_3 Risk-based Testing and Product Quality

Risk analysis is done:

- To focus the effort required during testing
- To decide where and when to start testing
- To identify areas that need more attention

Testing is used:

- To reduce the probability of an adverse event occurring
- To reduce the impact of an adverse event
- To provide information about identified risks,
- To provide information on residual (unresolved) risks

185. 5_5_3 Risk-based Testing and Product Quality

Why a risk-based approach is used

- Determine the test techniques to be employed
- Determine the particular levels and types of testing to be performed example security testing, accessibility testing
- Determine the extent of testing to be carried out
- Prioritize testing in an attempt to find the critical defects as early as possible
- Determine whether any activities in addition to testing could be employed to reduce risk, for example providing training to inexperienced designers.

- Software might not perform its intended functions according to the specification
- Software might not perform its intended functions according to user, customer, and/or stakeholder needs
- A system architecture may not adequately support some non-functional requirement(s)
- A particular computation may be performed incorrectly in some circumstances
- A loop control structure may be coded incorrectly
- Response-times may be inadequate for a high-performance transaction processing system
- User experience (UX) feedback might not meet product expectations

- Project issues:
  - Delays may occur in delivery, task completion, or satisfaction of exit criteria or definition of done
  - Inaccurate estimates, reallocation of funds to higher priority projects, or general cost-cutting across the organization may result in inadequate funding
  - Late changes may result in substantial re-work
- Organizational issues:
  - Skills, training, and staff may not be sufficient
  - Personnel issues may cause conflict and problems
  - Users, business staff, or subject matter experts may not be available due to conflicting business priorities
- Political issues:
  - Testers may not communicate their needs and/or the test results adequately
  - Developers and/or testers may fail to follow up on information found in testing and reviews (e.g., not improving development and testing practices)
  - There may be an improper attitude toward, or expectations of, testing (e.g., not appreciating the value of finding defects during testing)
- Technical issues:
  - Requirements may not be defined well enough
  - The requirements may not be met, given existing constraints
  - The test environment may not be ready on time
  - Data conversion, migration planning, and their tool support may be late
  - Weaknesses in the development process may impact the consistency or quality of project work products such as design, code, configuration, test data, and test cases
  - Poor defect management and similar problems may result in accumulated defects and other technical debt
- Supplier issues:
  - A third party may fail to deliver a necessary product or service, or go bankrupt
  - Contractual issues may cause problems to the project

Tebrikler!

"As described in section 5.5.2, product risks exist when a work product may fail to satisfy legitimate needs, while project risks are situations that could have a negative impact on the projects ability to achieve its objectives. So: A.Incorrect totals on reports = product risk B.Change to acceptance criteria during acceptance testing = project risk C.Users find the soft keyboard too hard to use with your app = product risk D.System responds too slowly to user inputduring search string entry = product risk E.Testers not allowed to report test results in daily standup meetings = project risk Th

Soru 26:

Consider the following list of undesirable outcomes that could occur on a mobile app development effort:
A. Incorrect totals on reports
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 meeting
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

Tebrikler!

"a)Is not correct.Quality should be the responsibility of everyone working on the project and not the sole responsibility of the test team b)Is not correct.First, it is not a benefit if an external test team does not meet delivery deadlines, and second, there is no reason to believe that external test teams will feel they do not have to meet strict delivery deadlines c)Is not correct.It is bad practice for the test team to work in complete isolation, and we would expect an external test team to be concerned with changing project requirements and communicate well with developers d)Is correct.Sp

Soru 27:

Which of the following BEST explains a benefit of independent testing?

- ○ The use of an independent test team allows project management to assign responsibility for the quality of the final deliverable to the test team, so ensuring everyone is aware that quality is the test teams overall responsibility
- ○ If a test team external to the organization can be afforded, then there are distinct benefits in terms of this external team not being so easily swayed by the delivery concerns of project management and the need to meet strict delivery deadlines
- ○ An independent test team can work totally separately from the developers, need not be distracted with changing project requirements, and can restrict communication with the developers to defect reporting through the defect management system
- ◉ **When specifications contain ambiguities and inconsistencies, assumptions are made on their interpretation, and an independent tester can be useful in questioning those assumptions and the interpretation made by the developer**

Tebrikler!

"The correct pairings of examples to entry and exit criteria are: -Entry criteria o(3) The trading performance test environment has been designed, set-up and verified –example of the need for a test environment to be ready before testing can begin o(5) The autopilot design specifications have been reviewed and reworked –example of the need for the test basis to be available before testing can begin o(6) The tax rate calculation component has passed unit testing –example of the need for a test object to have met the exit criteria for a prior level of testing before testing can begin -Exit cri

Soru 29:

Given the following examples of entry and exit criteria:
1.The original testing budget of doller30,000 plus contingency of doller7,000 has been spent
2.96percent 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:

- ○ Entry criteria -5, 6; Exit criteria -1, 2, 3, 4
- ○ Entry criteria -2, 3, 6; Exit criteria -1, 4, 5
- ○ Entry criteria -1, 3; Exit criteria -2, 4, 5, 6
- ◉ **Entry criteria -3, 5, 6; Exit criteria -1, 2, 4**

Tebrikler!

"a)Is not correct.Estimates may be updated as more information becomes available, but estimates are needed to assist with planning before the testing starts b)Is not correct.In the expert-based approach, the experts need to be experts in testing, not in using the test object c)Is correct.Test -Managers, who will be leading testers doing the testing, are considered experts in their respective areas and suitable for estimating the necessary resources needed d)Is not correct.While it is useful to know the testing costs from previous projects, a more sophisticated approach is needed than simply ta

Soru 30:

Which of the following statements about test estimation approaches is CORRECT?

- ○ With the metrics-based approach, the estimate is based on test measures from the project and so this estimate is only available after the testing starts
- ○ With the expert-based approach, a group of expert users identified by the client recommends the necessary testing budge
- ◉ **With the expert-based approach, the test managers responsible for the different testing activities predict the expected testing effort**
- ○ With the metrics-based approach, an average of the testing costs recorded from several past projects is used as the testing budge

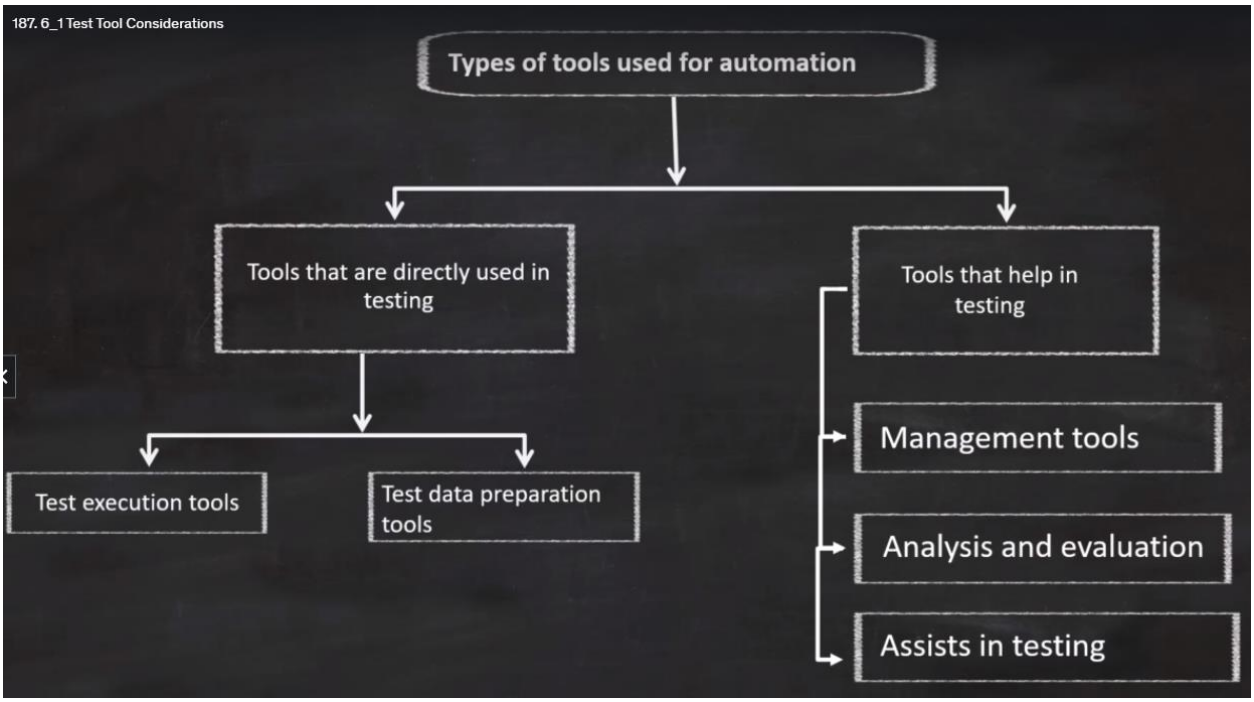
Tebrikler!

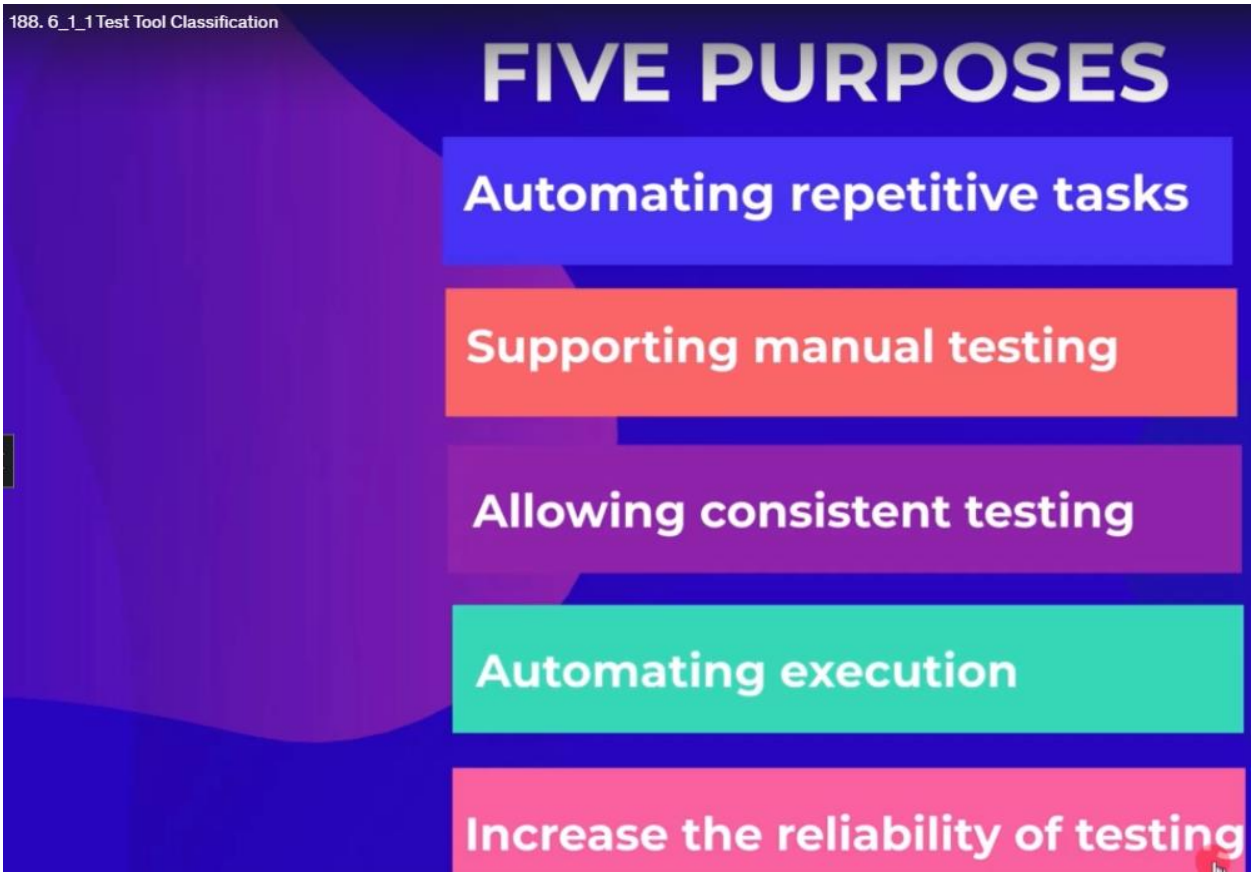
"a)Is not correct.Risk is determined by considering a combination of the likelihood of problem situations and the harm that may result from them but cannot be calculated by adding these together (the probability would be in the range 0 to 1 and the harm could be indollars) b)Is not correct.Risk is determined by considering a combination of a likelihood and an impact. This definition only considers likelihood and chance (both forms of probability) with no consideration of the impact (or harm) c)Is correct. See reasons fromincorrect answers d)Is not correct.Risk is determined by considering a co

Soru 31:

Which of the following BEST defines risk level?

- ○ Risk level is calculated by adding together the probabilities of all problem situations and the financial harm that results from them
- ○ Risk level is estimated by multiplying the likelihood of a threat to the system by the chance that the threat will occur and will result in financial damage
- ◉ **Risk level is determined by a combination of the probability of an undesirable event and the expected impact of that event**
- ○ Risk level is the sum of all potential hazards to a system multiplied by the sum of all potential losses from that system

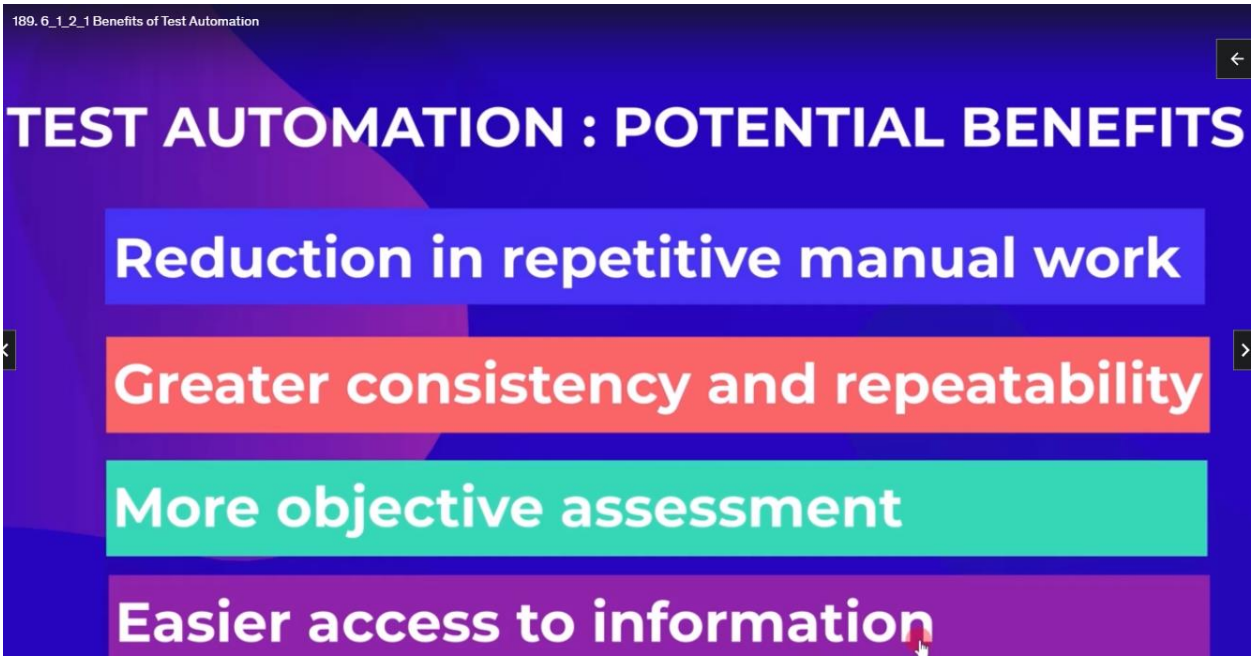




188. 6_1_1 Test Tool Classification

TEST ACTIVITY	TOOLS
TEST MANAGEMENT	• Test management tool or application management tool • Requirement management tool • Defect management tool • Configuration management tools • Continuous integration tools
STATIC TESTING	Static analysis tool
TEST DESIGN AND IMPLEMENTATION	• Model-based testing tools • Test data preparation tools
TEST EXECUTION	• Test execution tool • Coverage tools • Test harnesses
Performance measurement and Dynamic analysis tools	• Performance testing tools • Dynamic analysis tools

TOOLS USED BY DEVELOPERS	TOOLS USED BY TESTERS
. Continuous integration tool . Static analysis tools . Coverage tools . Test harnesses . Dynamic analysis tools	. Test management tool or application management tool . Requirement management tool . Defect management tool . Configuration management tools . Continuous management tools . model-based testing tools . test data preparation tools. . Test execution tool • Performance testing tools

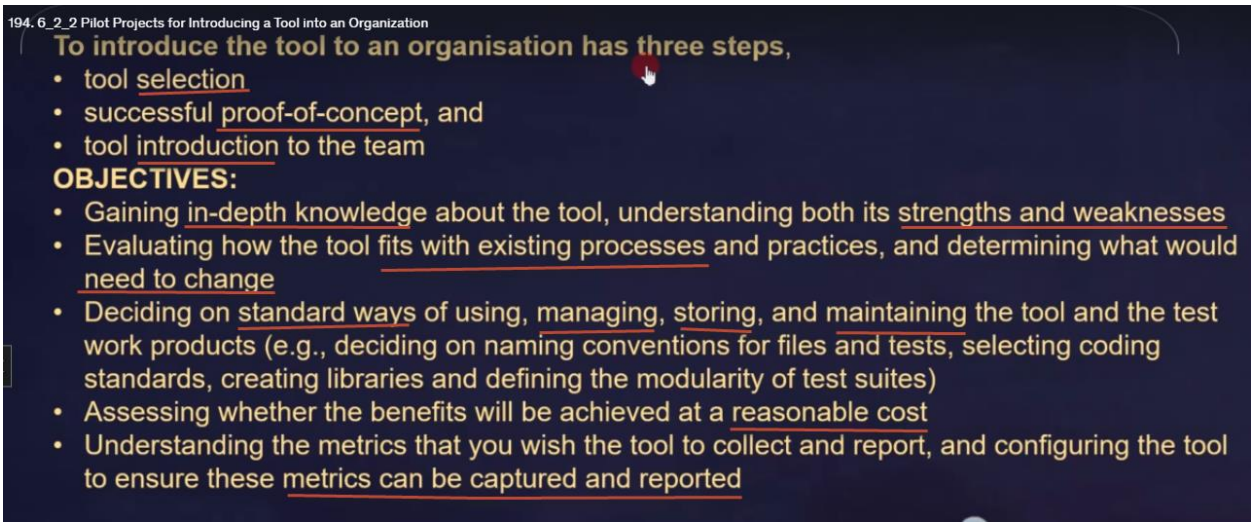


POTENTIAL RISKS OF USING TOOL

- 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 under-estimated
- 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, such as requirements management tools, configuration management tools, defect management tools and tools from multiple vendors
- 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

193. 6_2_1 Main Principles for Tool Selection

- Assessment of the maturity of its organization, its strengths and weaknesses
- Identification of opportunities for an improved test process supported by tools
- Understanding the technologies used by the test object(s), to select a tool that is compatible with that technology
- Understanding the build and continuous integration tools already in use within the organization, to ensure tool compatibility and integration
- Evaluation of the tool against clear requirements and objective criteria
- Consideration of whether or not the tool is available for a free trial period (and for how long)
- valuation of the vendor (including training, support and commercial aspects) or support for non-commercial (e.g., open-source) tools
- Identification of internal requirements for coaching and mentoring in the use of the tool
- Evaluation of training needs, considering the testing (and test automation) skills of those who will be working directly with the tool(s)
- Consideration of the pros and cons of various licensing models (e.g., commercial or open-source)
- Estimation of a cost-benefit ratio based on a concrete business case (if required)



Tebrikler!

"Item a (test data preparation tools) is, as its name implies, a tool used before test execution to prepare test data. Item b (test harness) is used at test execution to enable tests to be run on code that offers no convenient input/output interface and/or that is incomplete in some way that might prevent a test from running successfully. Item c is a tool to support the review process and is not related to test execution. Item d is used to compare test results with expected results and is therefore part of the test execution suite of tools. Item e is a tool for managing software conf

Soru 7:

Which TWO of the following test tools would be classified as test execution tools?

a. Test data preparation tools
b. Test harness
c. Review tools
d. Test comparators
e. Configuration management tools

- ○ a and b
- ○ c and d
- ○ c and e
- ◉ **b and d**

Tebrikler!

"Option A is incorrect; requirements needed to be part of the selection process. Option B is incorrect because a pilot project would be carried out only on the selected tool. Option C is correct and the pilot will assess cost effectiveness in use. Option D is incorrect because tools are not expected to fit existing processes; the pilot project is partly about identifying any process changes that may be necessary to implement the tool. "

Soru 8:

What is the main reason for using a pilot project to introduce a testing tool into an organization?

- ○ To identify the requirements for using a tool
- ○ To make a selection between alternative tools
- ◉ **To assess whether the tool will be cost- effective**
- ○ To ensure the tools fits existing processes without change

Tebrikler!

"a) TRUE – because traceability between requirements and testing is a functionality of a test management tool (section 6.1.3) b) FALSE – because this is not possible with test management tools (section 6.1.6) c) FALSE -- because this is not mainly solved by test management tools (section 6.1.3) d) FALSE – because the selection of test cases is not supported by test management tools (section 6.1.6) "

Soru 14:

A software development and test organization would like to achieve the test efficiency improvement goals listed below.

Which would best be supported by a test management tool?

- ◉ **Enable traceability between requirements, tests, and defects (bugs)**
- ○ Optimize the ability of tests to identify failures
- ○ Resolve defects faster
- ○ Automate a selection of test cases for execution

Tebrikler!

"a)Is not correct: per section 6.2.2, this is an objective for a pilot, but you have achieved it because you understand the tool much better due to the pilot. b)Is not correct: per section 6.2.2, this is an objective for a pilot, but you have achieved it because you have tailoring your testing processes. c)Is not correct: per section 6.2.2, this is an objective for a pilot, but you have achieved it because you have standardized an approach to using the tool and its associated work products. d)Is correct: per section 6.2.2, assessing the benefits and configuring the metrics collection are the t

Soru 15:

You have just completed a pilot project for a regression testing tool. You understand the tool much better, and have tailored your testing process to it. You have standardized an approach to using the tool and its associated work products.

Which of the following is a typical test automation pilot project goal that remains to be carried out?

- ○ Learn more details about the tool
- ○ See how the tool would fit with existing processes and practice
- ○ Decide on standard ways of using, managing, storing, and maintaining the tool and the test assets
- ◉ **Assess whether the benefits will be achieved at reasonable cost**

TEST ACTIVITY	TOOLS
TEST MANAGEMENT	• Test management tool or application management tool • Requirement management tool • Defect management tool • Configuration management tools • Continuous integration tools
STATIC TESTING	Static analysis tool
TEST DESIGN AND IMPLEMENTATION	• Model-based testing tools • Test data preparation tools
TEST EXECUTION	• Test execution tool • Coverage tools • Test harnesses
Performance measurement and Dynamic analysis tools	• Performance testing tools • Dynamic analysis tools

TOOLS USED BY DEVELOPERS	TOOLS USED BY TESTERS
. Continuous integration tool . Static analysis tools . Coverage tools . Test harnesses . Dynamic analysis tools	. Test management tool or application management tool . Requirement management tool . Defect management tool . Configuration management tools . Continuous management tools . model-based testing tools . test data preparation tools. . Test execution tool • Performance testing tools