Ripon Al Wasim
riponalwasim@gmail.com
LinkedIn

Course Outline

Chapter 1: Fundamentals of Testing
Chapter 2: Testing Throughout the Software Development Lifecycle
Chapter 3: Static Testing
Chapter 4: **Test Techniques**
Chapter 5: Test Management
Chapter 6: Tool Support for Testing

Course Outline

Chapter 4: Test Techniques

No. of Session: 04
Session 06: 4 - 4.2.1
Session 07: 4.2.2 - 4.2.5
Session 08: 4.3 - 4.3.3
Session 09: 4.4 - 4.4.3

Chapter 4: Test Techniques

Session 06

4.1 Categories of Test Techniques
4.1.1 Choosing Test Techniques
4.1.2 Categories of Test Techniques and Their Characteristics
4.2 Black-box Test Techniques
4.2.1 Equivalence Partitioning

Chapter 4: Test Techniques

Categories of Test Techniques

The purpose of a test technique:

- To identify test conditions, test cases and test data.
- **Test conditions** are identified during analysis, and then used to define **test cases**.
- **Test data** are identified during test design, which are then used in test implementation.

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /1

- ➔ Type of component or system
- ➔ Component or system complexity
- ➔ Regulatory standards
- ➔ Customer or contractual requirements
- ➔ Risk levels and risk types
- ➔ Test objectives
- ➔ Available documentation
- ➔ Tester knowledge and skills
- ➔ Available tools
- ➔ Time and budget
- ➔ Software development life cycle model
- ➔ Expected use of the software
- ➔ Previous experience with using the test techniques on the component or system to be tested
- ➔ The types of defects expected in the component or system

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /2

- Type of component or system (embedded, graphical, financial, etc.): For example, a financial application involving many calculations would benefit from boundary value analysis.
- Component or system complexity: For example, a simple field with numerical input would be a good candidate for EP and BVA, but a screen with many fields, with complex dependencies, calculations and validation rules depending on aspects that change over time, would benefit from additional techniques such as decision table testing, state transition testing and white-box test techniques.

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /3

- **Regulatory standards:** For example, the aircraft industry requires the use of EP, BVA and state transition testing for high integrity systems, together with statement, decision or modified condition decision coverage depending on the level of software integrity required.
- **Customer or contractual requirements:** Sometimes contracts specify particular test techniques to use (most commonly statement or branch coverage).

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /4

- Risk levels and risk types:

safety-critical systems - need more thorough, formal testing
Commercial risk of commercial systems - more thorough testing would be appropriate
by time to market issues - exploratory testing would be a more appropriate choice

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /5

- Test objectives: test objective is simply to gain confidence that the software will cope with typical operational tasks - use cases would be a sensible approach.

the objective is for very thorough testing, then more rigorous and detailed techniques (including white-box test techniques) should be chosen.

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /6

- Available documentation: a work product such as a requirements specification) exists, and how up-to-date it is, will affect the choice of test techniques.

The content and style of the work products will also influence the choice of techniques (for example, if decision tables or state graphs have been used, then the associated test techniques should be used).

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /7

- Tester knowledge and skills: Experience-based techniques are particularly based on tester knowledge and skills.
- Available tools: tools are available for a particular technique -> that technique may be a good choice.

the specification contains a state transition diagram, state transition testing would be a good technique to use.

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /8

- ❏ Time and budget: Ultimately how much time there is available will always affect the choice of test techniques.
  - More time is available, we can afford to select more techniques.
  - Time is severely limited, we will be limited, find just the most important defects.

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /9

- Software development life cycle model: A sequential life cycle model will lend itself to the use of more formal techniques, whereas an iterative life cycle model may be better suited to using an exploratory test approach.
- Expected use of the software: If the software is to be used in safety-critical situations, for example medical monitoring devices or car-driving technology, then the testing should be more thorough, and more techniques should be chosen.

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /10

- ❏ Previous experience with using the test techniques on the component or system to be tested:

Testers tend to use techniques that they are familiar with and more skilled at, rather than less familiar techniques.

But be aware that you may get better results from using a technique that is less familiar, and when you do use it, you will increase your skill and familiarity with it.

Chapter 4: Test Techniques

Choosing Test techniques

Factors for choosing test techniques /11

- The types of defects expected in the component or system: Knowledge of the likely defects will be very helpful in choosing test techniques (since each technique is good at finding a particular type of defect).

Chapter 4: Test Techniques

Categories of test techniques and their characteristics /1

2 main categories

1. Static techniques
2. Dynamic techniques

Dynamic techniques are subdivided into three more categories:

I. Black-box (specification-based, also known as behavioral techniques)
II. White-box (structure-based or structural techniques) and
III. Experience-based

Black-box (Specification-based techniques) include both functional and nonfunctional techniques (i.e. quality characteristics).

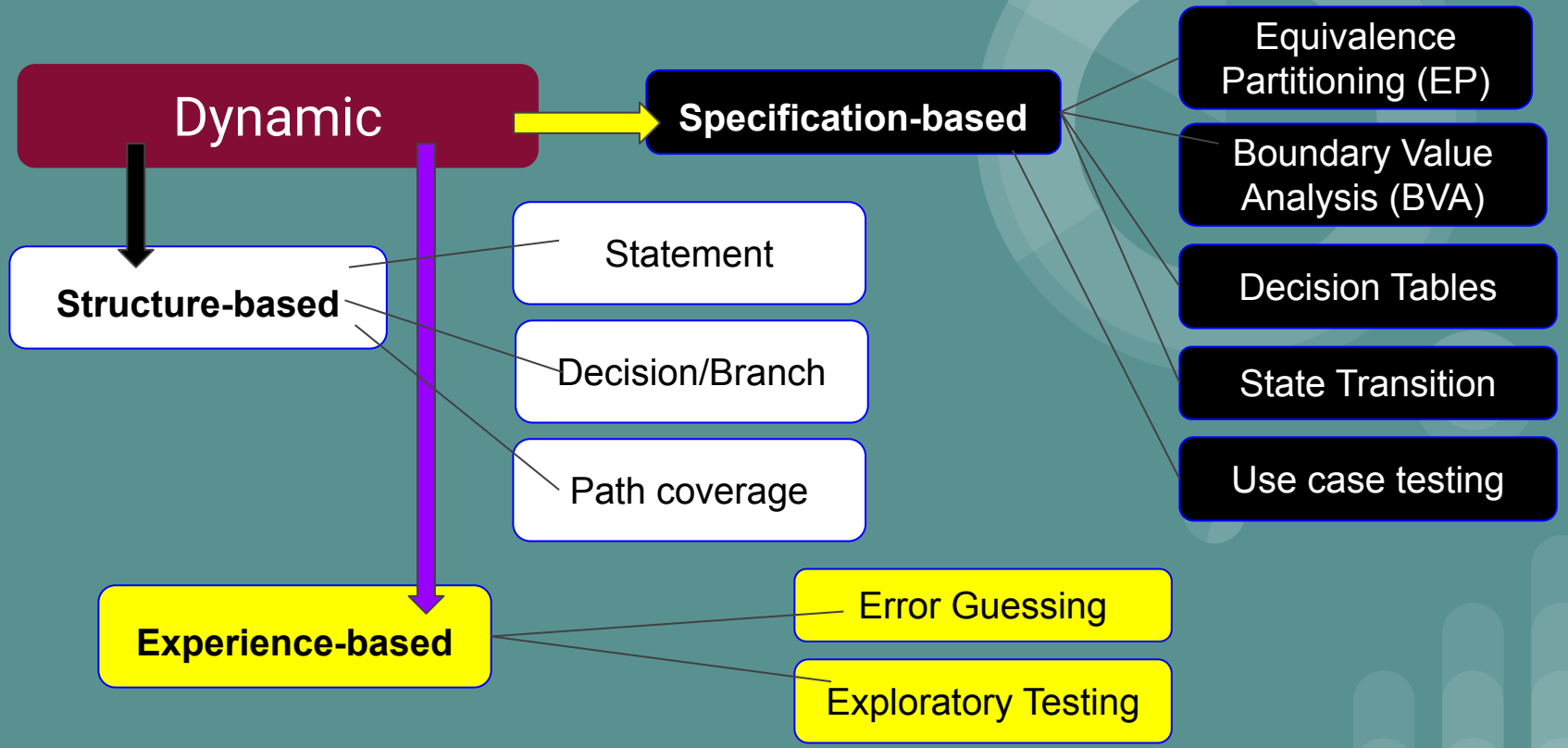
Chapter 4: Test Techniques

Categories of test techniques and their characteristics /2

Static	Dynamic		
Informal Reviews	**Specification-based**	**Structure-based**	**Experience-based**
Walkthroughs	Equivalence Partitioning (EP)	Statement	Error Guessing
Technical Reviews	Boundary Value Analysis (BVA)	Decision/Branch	Exploratory Testing
Inspection	Decision Tables		
	State Transition		
	Use case testing		

Chapter 4: Test Techniques

Categories of test techniques and their characteristics /3



Chapter 4: Test Techniques

Black-box test techniques /1

- Test conditions, test cases and test data are derived from a test basis (software requirements, specifications, use cases and user stories). The source of information for black-box tests is some description of what the system or software is supposed to do.

- Test cases may be used to detect gaps between the requirements and the implementation of the requirements, as well as deviations from the requirements.

Chapter 4: Test Techniques

Black-box test techniques /2

- Coverage is measured based on the items tested in the test basis and the technique applied to the test basis.

Coverage at black-box level is based on items from the test basis. For example, does every requirement described have at least one test that exercises it?

Chapter 4: Test Techniques

Equivalence Partitioning (EP) /1

Scenario: A savings account in a bank earns a different rate of interest depending on the balance in the account.
For example, if a balance in the range $0 up to $100 has a 3% interest rate, a balance over $100 and up to $1000 has a 5% interest rate, and balances of $1000 and over have a 7% interest rate, we would initially identify three valid equivalence partitions and one invalid partition as shown below.

Chapter 4: Test Techniques

Equivalence Partitioning (EP) /2

Invalid partition	Valid (for 3% interest)		Valid (for 5%)		Valid (for 7%)
–$0.01	$0.00	$100.00	$100.01	$999.99	$1,000.00

Chapter 4: Test Techniques

Equivalence Partitioning (EP) /3

EP characteristics

- Valid values should be accepted by the component or system. An equivalence partition containing valid values is called a valid equivalence partition.
- Invalid values should be rejected by the component or system. An equivalence partition containing invalid values is called an invalid equivalence partition.

Chapter 4: Test Techniques

Equivalence Partitioning (EP) /4

EP characteristics

- Partitions can be identified for any data element related to the test object, including inputs, outputs, internal values, time-related values (for example before or after an event) and for interface parameters (for example integrated components being tested during integration testing).
- Any partition may be divided into sub-partitions if required, where smaller differences of behaviour are defined or possible.
  - ➢ For example, if a valid input range goes from -100 to 100, then we could have three sub-partitions: valid and negative, valid and zero, and valid and positive.

Chapter 4: Test Techniques

Equivalence Partitioning (EP) /5

EP characteristics

- Each value belongs to one and only one equivalence partition from a set of partitions.

➢ EP is applicable at all test levels.


THANK YOU!

Chapter 4: Test Techniques

Session 07

4.2.2 Boundary Value Analysis
4.2.3 Decision Table Testing
4.2.4 State Transition Testing
4.2.5 Use Case Testing

Chapter 4: Test Techniques

Boundary Value Analysis (BVA) /1

- BVA is essentially an enhancement or extension of EP.

Example: Consider a printer that has an input option of the number of copies to be made, from 1 to 99.

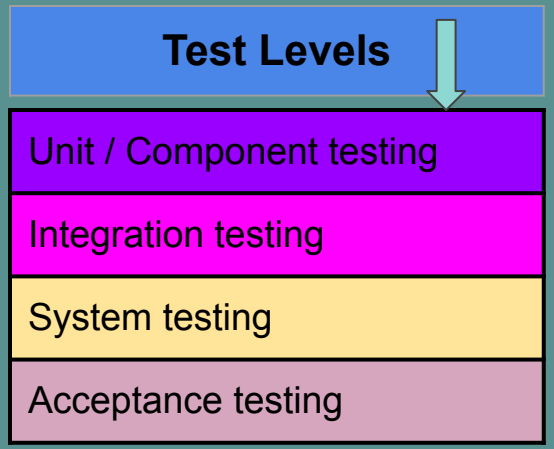
Invalid	Valid	Invalid
0	1 99	100

Specification-based
EP
BVA
Decision Tables
State Transition
Use case testing

Chapter 4: Test Techniques

Boundary Value Analysis (BVA) /2

BVA can be applied at all test levels.



Specification-based
EP
BVA
Decision Tables
State Transition
Use case testing

Chapter 4: Test Techniques

Boundary Value Analysis (BVA) /3

Equivalence Partitioning/Boundary Value Analysis exercise

Scenario: If you take the train before 9:30 am or in the afternoon after 4:00 pm until 7:30 pm ('the rush hour'), you must pay full fare. A saver ticket is available for trains between 9:30 am and 4:00 pm, and after 7:30 pm.

What are the partitions and boundary values to test the train times for ticket types? Which are valid partitions and which are invalid partitions? What are the boundary values? (A table may be helpful to organize your partitions and boundaries.) Derive test cases for the partitions and boundaries.

Are there any questions you have about this 'requirement'? Is anything unclear?

Chapter 4: Test Techniques

Decision Table Testing ('cause-effect' table) /1

Formula: 2^n, n = No. of conditions

Example: Decision table for Login

Conditions: User name, Password, No. of conditions = 2

Rules: $2^2 = 4$

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
User Name	T	T	F	F
Password	T	F	T	F
Actions/Outcomes	Go to Home page	Error message	Error message	Error message

Specification-based
EP
BVA
Decision Tables
State Transition
Use case testing

Chapter 4: Test Techniques

Decision Table Testing ('cause-effect' table) /2

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
User Name	T	T	F	F
Password	T	F	T	F
Actions/Outcomes	Go to Home page	Error message	Error message	Error message

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
User Name	Correct	Correct	Wrong/Blank	Wrong/Blank
Password	Correct	Wrong/Blank	Correct	Wrong/Blank
Actions/Outcomes	Go to Home page	Error message	Error message	Error message

Specification-based
EP
BVA
Decision Tables
State Transition
Use case testing

T = Correct
F = Wrong/Blank

Chapter 4: Test Techniques / Decision Table Testing /3

Rules	User Name	Password	Outcomes
Rule 1	Correct	Correct	Go to Home page
Rule 2	Correct	Wrong	Error message
Rule 2	Correct	Blank	Error message
Rule 3	Wrong	Correct	Error message
Rule 3	Blank	Correct	Error message
Rule 4	Wrong	Wrong	Error message
Rule 4	Wrong	Blank	Error message
Rule 4	Blank	Wrong	Error message
Rule 4	Blank	Blank	Error message

	Rule 1	Rule 2	Rule 3	Rule 4
User	Correct	Correct	Wrong/ Blank	Wrong/Blank
PW	Correct	Wrong/ Blank	Correct	Wrong/Blank
	Home	Error	Error	Error

T = Correct
F = Wrong/Blank

Chapter 4: Test Techniques

Decision Table Testing - Exercise

Exercise 1

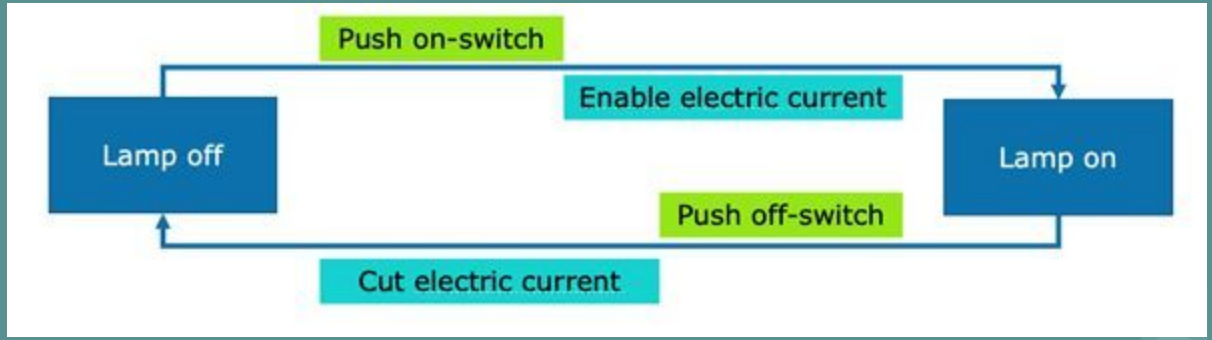
Font family: Arial, Time Roman Font style: Regular, Bold Font size: 10, 12	Produce a decision table showing all the combinations of Font family, style, size and derive test cases from the decision table.

Exercise 2: If you hold an 'over 60s' rail card, you get a 34% discount on whatever ticket you buy. If you are traveling with a child (under 16), you can get a 50% discount on any ticket if you hold a family rail card, otherwise you get a 10% discount. You can only hold one type of rail card.
Produce a decision table showing all the combinations of fare types and resulting discounts and derive test cases from the decision table.

Chapter 4: Test Techniques / State Transition Testing/1

4 main parts/components: 1. States 2. Transition 3. Events 4. Actions

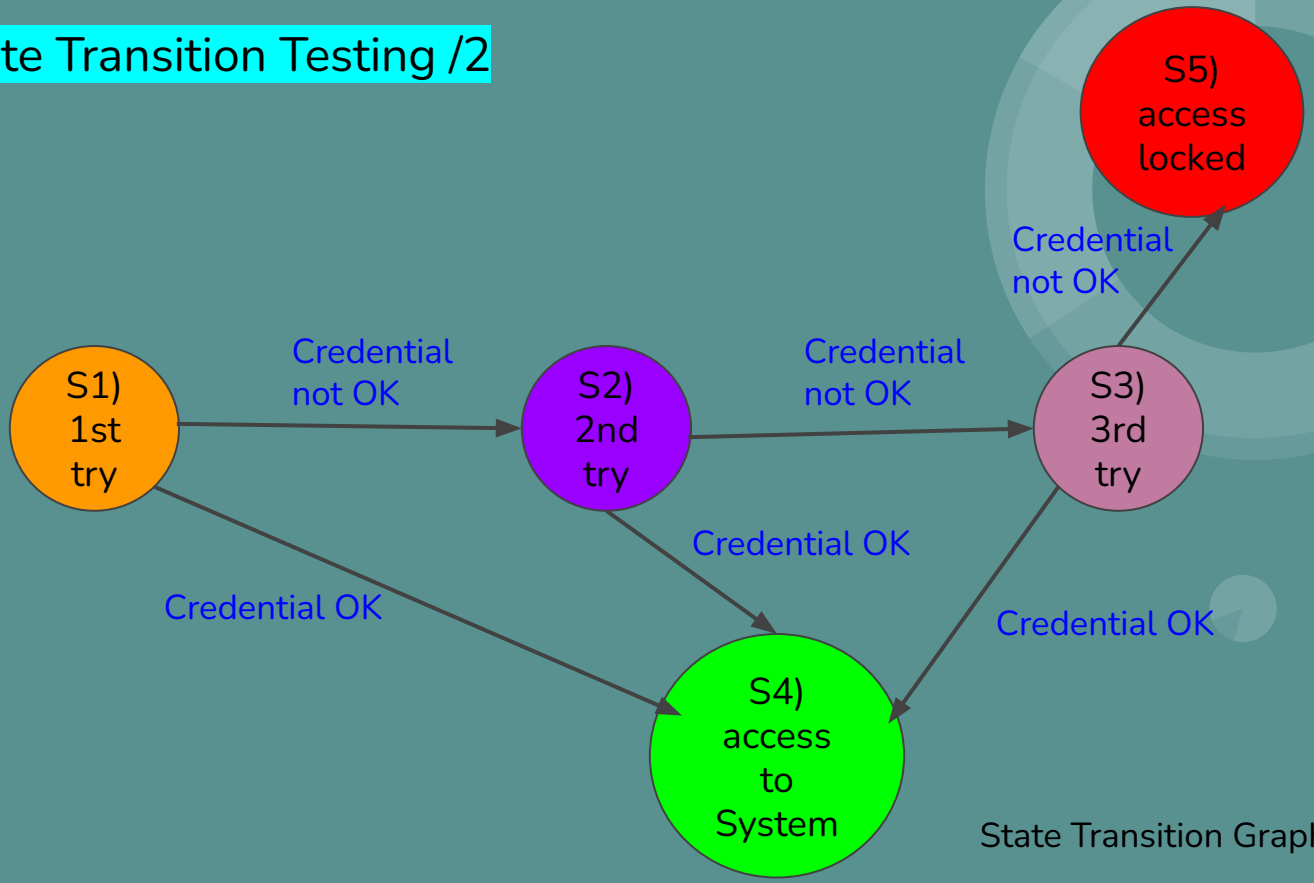
State	A state is a distinguishable situation of a system. A system can only be in one state at any point in time. A system can transition from one state to another.
Transition	A transition is a change from one state to another. A transition may also be to the same state (a "transition-to-self").
Event	An event is an occurrence inside or outside a system that can result in a transition.
Action	An action is behavior executed at a particular point in the system. It can also be a string of actions.



Specification-based
EP
BVA
Decision Tables
State Transition
Use case testing

Chapter 4: Test Techniques

State Transition Testing /2



State Transition Graph/Diagram for Login

Specification-based
EP
BVA
Decision Tables
State Transition
Use case testing

Chapter 4: Test Techniques

State Transition Testing /1

State	Login	Correct credential	Incorrect credential
S1	1st try	S4	S2
S2	2nd try	S4	S3
S3	3rd try	S4	S5
S4	Access to Home page	-	-
S5	Display a message as “Account locked, please consult Administrator”	-	-

State Transition Table for Login

Specification-based
EP
BVA
Decision Tables
State Transition
Use case testing

Chapter 4: Test Techniques

State Transition Testing /2

Consider an example of entering a Personal Identity Number (PIN) to a bank account. The states are shown as circles, the transitions as lines with arrows and the events as the text near the transitions.
(We have not shown the actions explicitly on this diagram, but they would be a message to the customer saying things such as 'Please enter your PIN'.)

Specification-based
EP
BVA
Decision Tables
State Transition
Use case testing

Chapter 4: Test Techniques

State Transition Testing /3

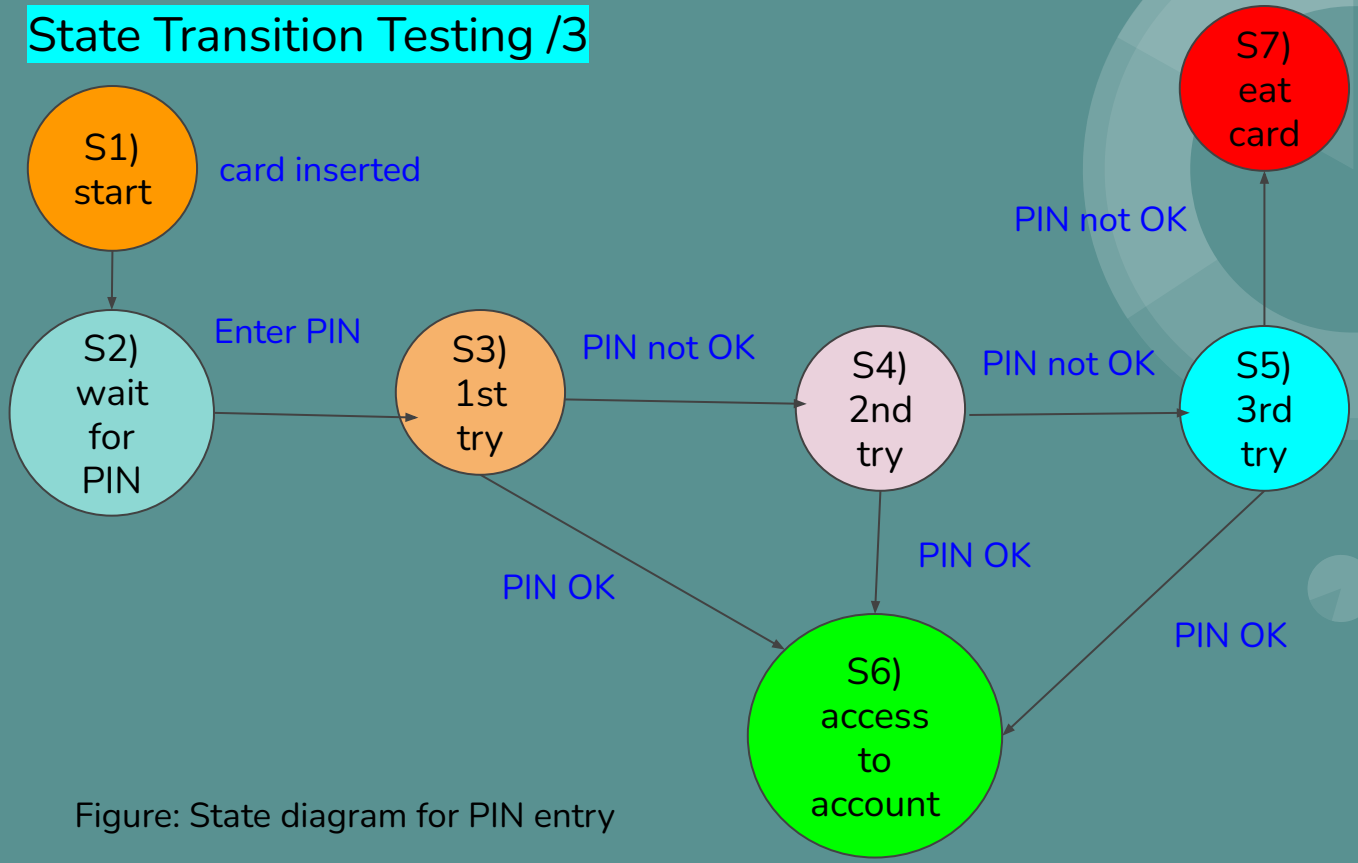


Figure: State diagram for PIN entry

Specification-based
EP
BVA
Decision Tables
State Transition
Use case testing

Chapter 4: Test Techniques

State Transition Testing - Exercise

Scenario: A website shopping basket starts out as empty. As purchases are selected, they are added to the shopping basket. Items can also be removed from the shopping basket. When the customer decides to check out, a summary of the items in the basket and the total cost are shown, for the customer to say whether this is OK or not. If the contents and price are OK, then you leave the summary display and go to the payment system. Otherwise you go back to shopping (so you can remove items if you want).

a. Produce a state diagram showing the different states and transitions. Define a test, in terms of the sequence of states, to cover all transitions.

b. Produce a state table. Give an example test for an invalid transition.

Chapter 4: Test Techniques

Use Case Testing/1: Use case for Login functionality of a Web application

Main Success Scenario A: Actor S: System	Step	Description
	1	A: Enter Username & Password
	2	S: Validate Username and Password
	3	S: Allow access to System
Extensions	2a	Invalid Username S: System shows an error message
	2b	Invalid Password S: System shows an error message and ask for re-try 3 times
	2c	Invalid Password for 3 times S: Application closed

Specification -based
EP
BVA
Decision Tables
State Transition
Use case testing

Chapter 4: Test Techniques

Use Case Testing /2: Partial Use case for PIN entry

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

Specification -based

- EP
- BVA
- Decision Tables
- State Transition
- Use case testing


THANK
YOU

Chapter 4: Test Techniques

Session 08

4.3 White-box Test Techniques

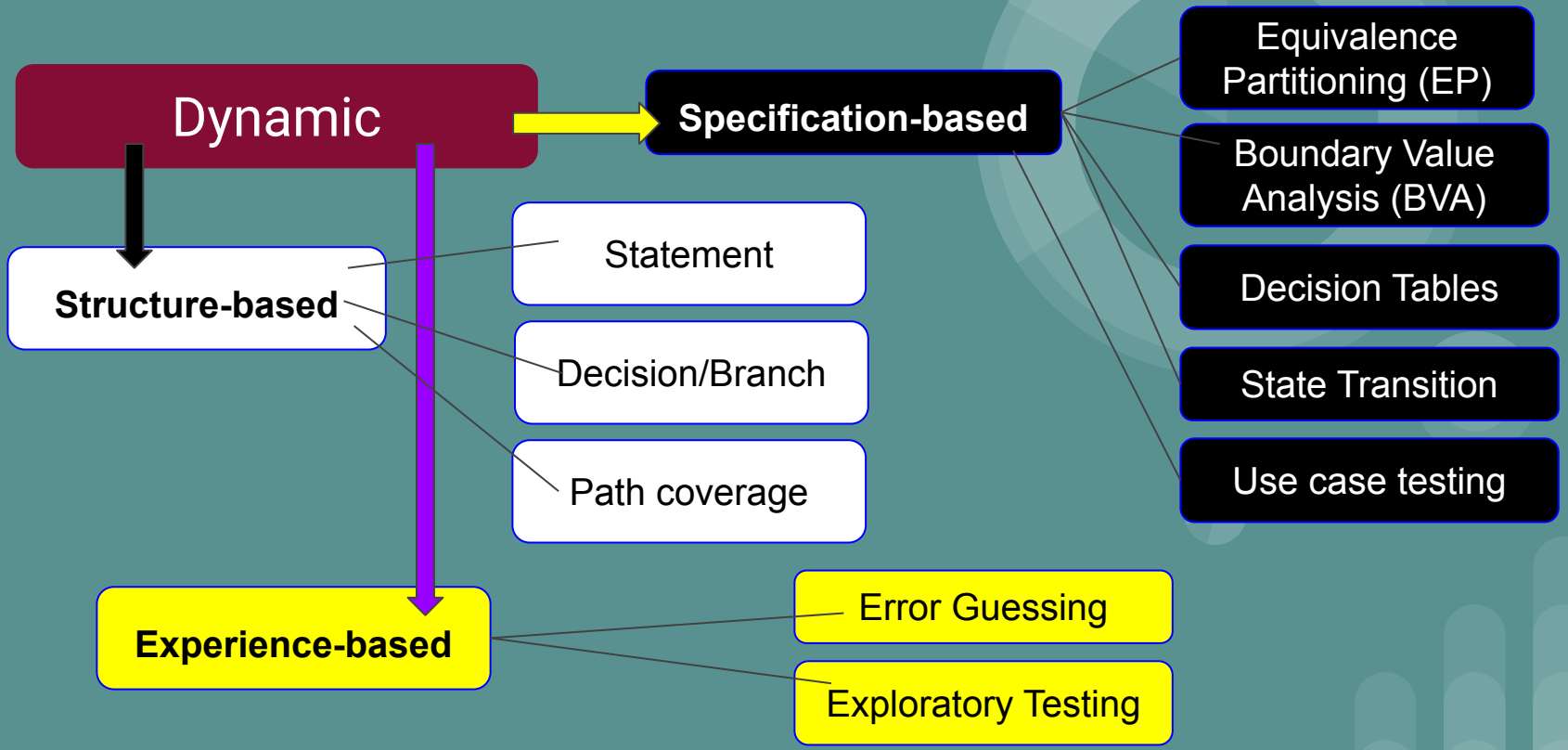
4.3.1 Statement Testing and Coverage

4.3.2 Decision Testing and Coverage

4.3.3 The Value of Statement and Decision Testing

Chapter 4: Test Techniques

Categories of test techniques and their characteristics



Chapter 4: Test Techniques

White-box (Structure-based) Test techniques

Structure-based testing techniques (dynamic rather than static) use the internal structure of the software to derive test cases. They are commonly called 'white-box' or 'glass-box' techniques since they require knowledge of how the software is implemented, that is, how it works.

For example, a structural technique may be concerned with exercising loops in the software. Different test cases may be derived to exercise the loop once, twice, and many times. This may be done regardless of the functionality of the software.

Chapter 4: Test Techniques

Statement Testing and Coverage /1

Statement coverage is a white box testing technique, which involves the execution of all the statements at least once in the source code.

It is used to calculate the total number of executed statements in the source code out of total statements present in the source code.

$$\text{Statement Coverage (SC)} = \frac{\text{Number of statements executed}}{\text{Total number of statements}} \times 100$$

Chapter 4: Test Techniques

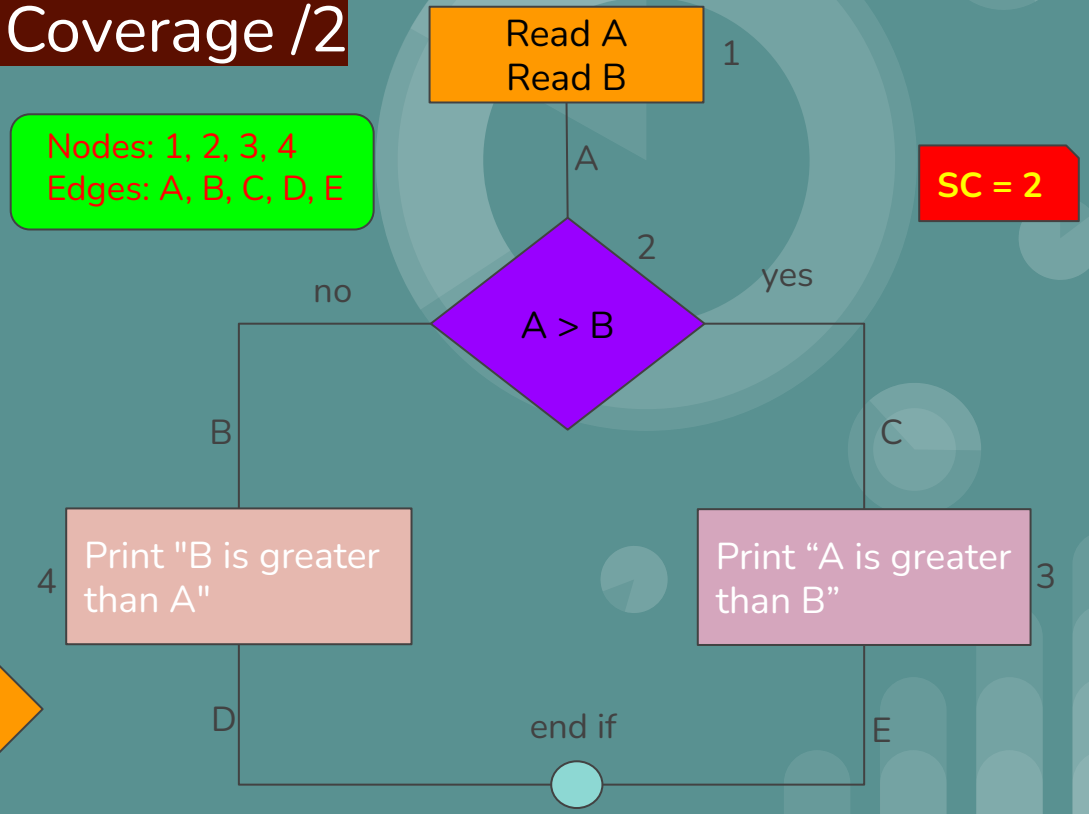
Statement Testing and Coverage /2

Example:

```
Read A
Read B
if A > B
  Print “A is greater than B”
else
  Print "B is greater than A"
Endif
```

Path:

- 1A - 2C - 3E
- 1A - 2B - 4D



Chapter 4: Test Techniques

Statement Testing and Coverage /3

Calculation:

$$SC = \frac{\text{No. of statements executed}}{\text{Total number of statements}} \times 100$$

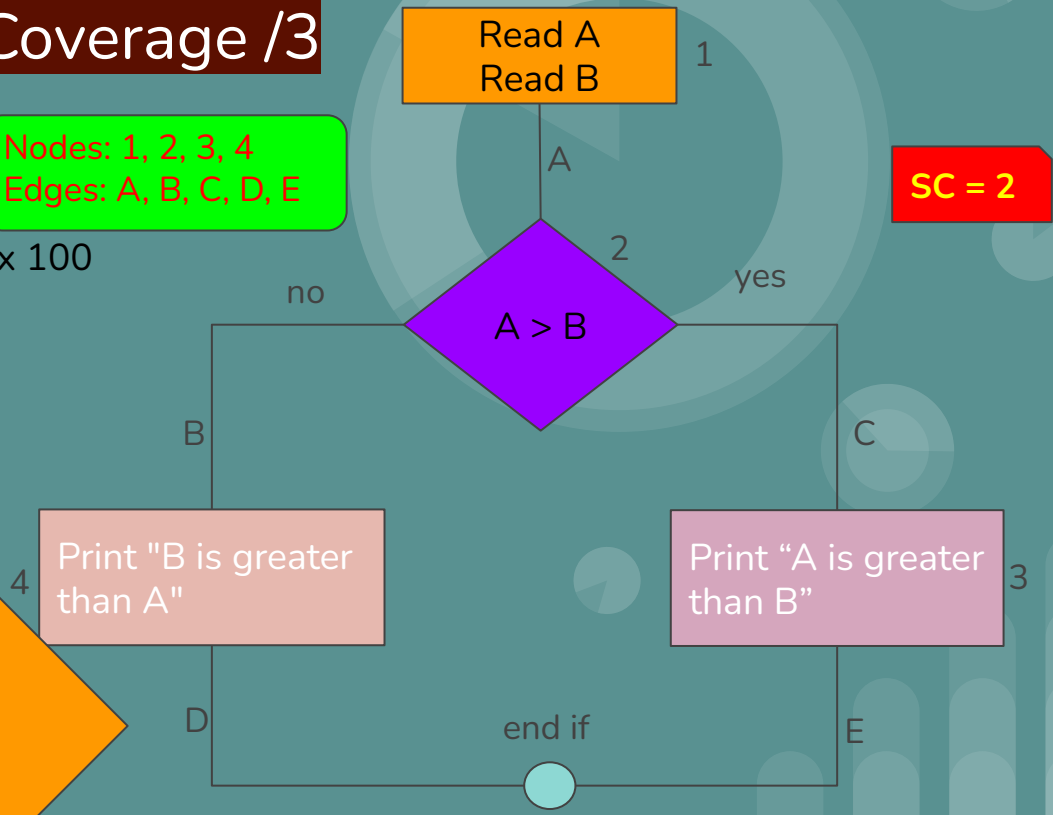
A = 11, B = 5
SC = ½ x 100 = 50%

A = 6, B = 12
SC = ½ x 100 = 50%

Path:
- 1A - 2C - 3E
- 1A - 2B - 4D

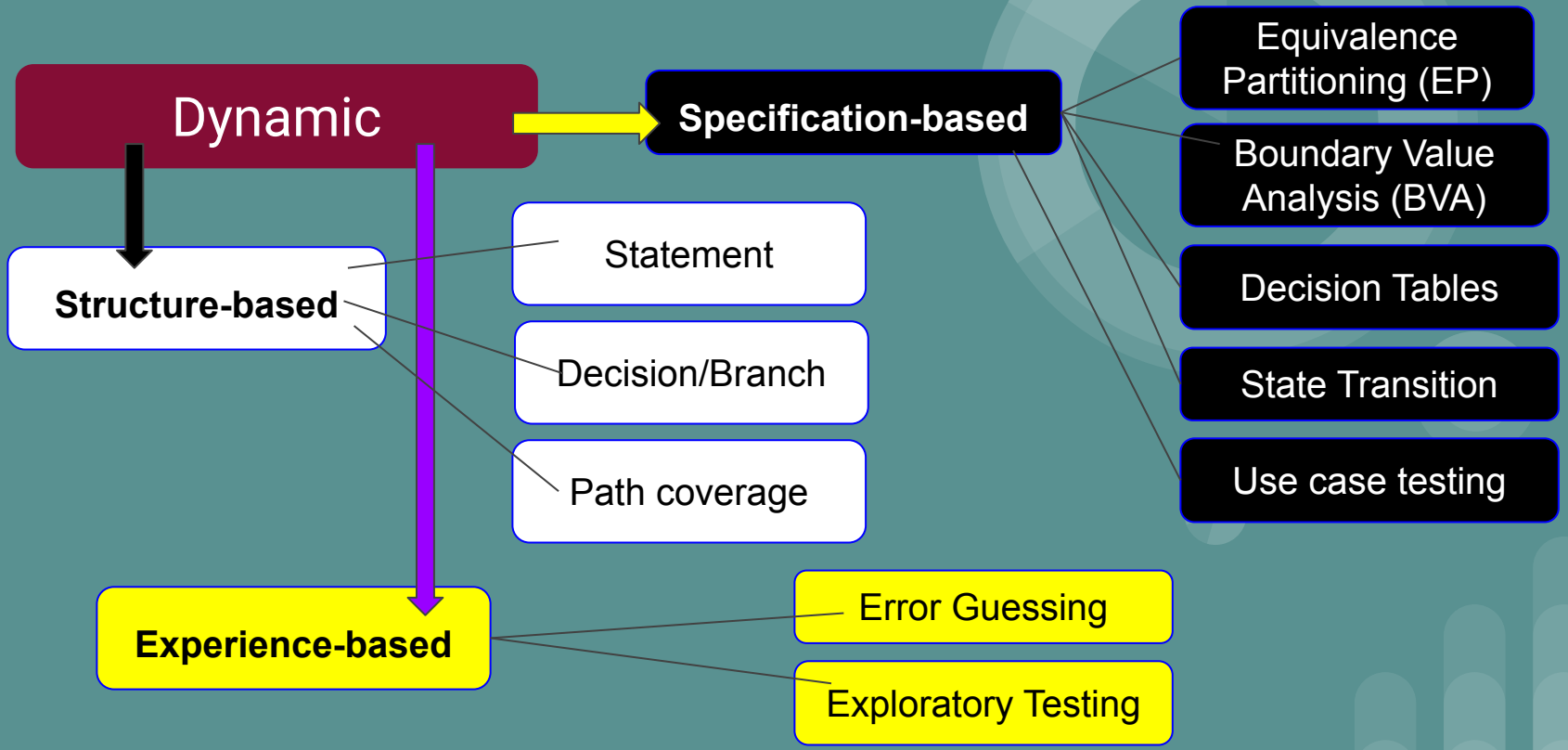
Nodes: 1, 2, 3, 4
Edges: A, B, C, D, E

SC = 2



Chapter 4: Test Techniques

Categories of test techniques and their characteristics



Chapter 4: Test Techniques

Decision Testing and Coverage /1

Decision Coverage is a white box testing technique which reports the true or false outcomes of each boolean expression of the source code. Within the scope of decision coverage testing, all possible branches from each decision point are executed at least once.

$$\text{Decision Coverage (DC)} = \frac{\text{Number of decision outcomes exercised}}{\text{Total number of decision outcomes}} \times 100$$

- Decision Coverage (DC) is also known as Branch Coverage (BC).

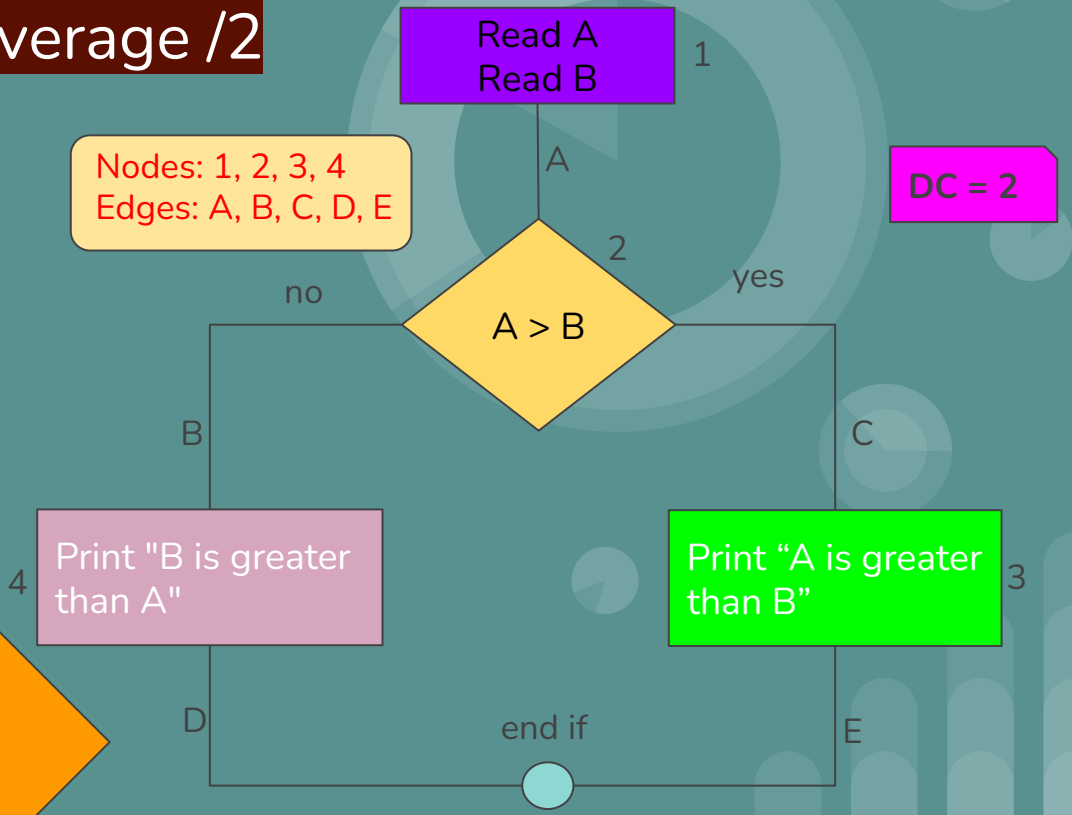
Chapter 4: Test Techniques

Decision Testing and Coverage /2

Example:

```
Read A
Read B
if A > B
  Print “A is greater than B”
else
  Print "B is greater than A"
Endif
```

Path:
- 1A - 2C - 3E
- 1A - 2B - 4D



Chapter 4: Test Techniques

Decision Testing and Coverage /3

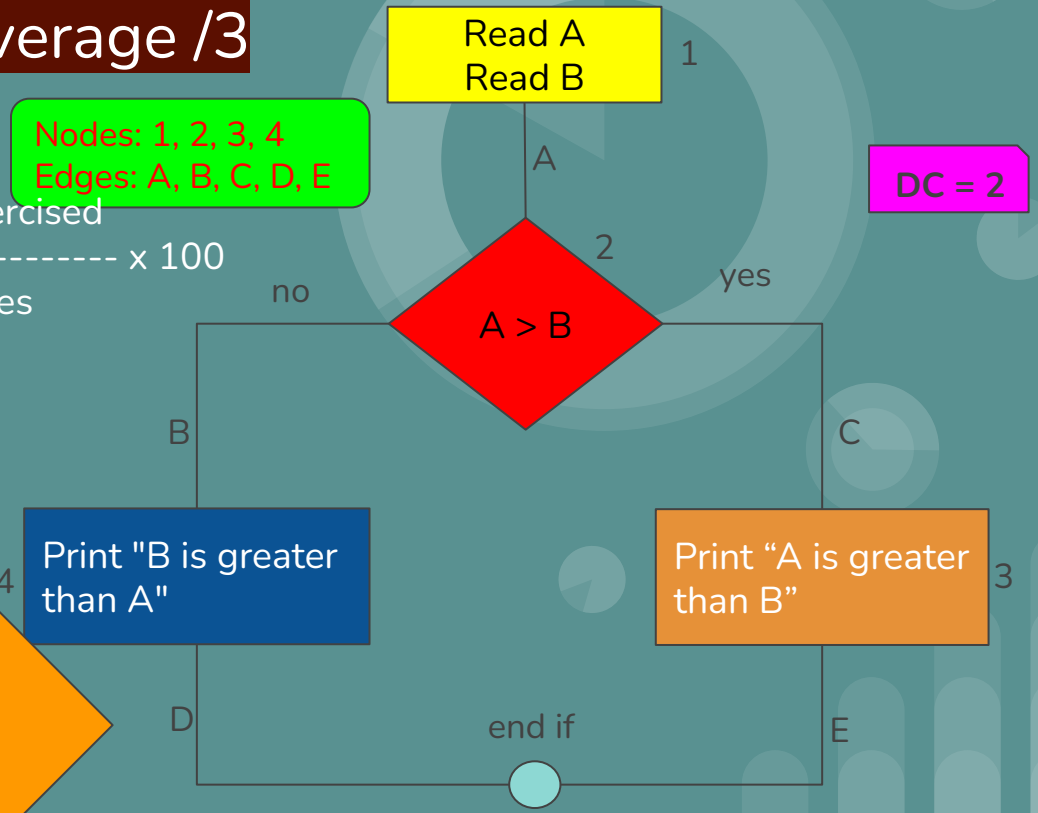
Calculation:

$$DC = \frac{\text{Number of decision outcomes exercised}}{\text{Total number of decision outcomes}} \times 100$$

A = 50, B = 30
SC = ½ x 100 = 50%

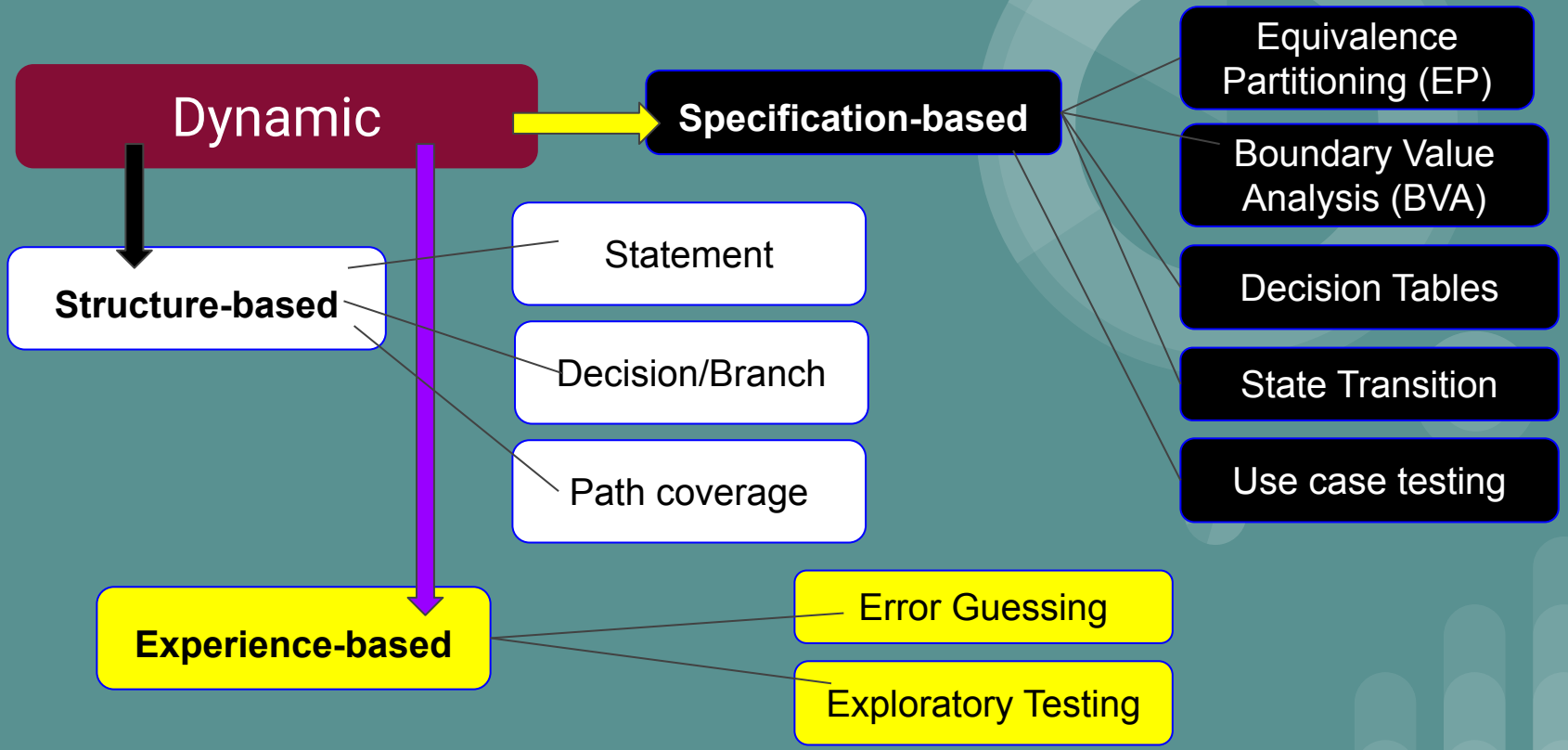
A = 10, B = 20
SC = ½ x 100 = 50%

Path:
- 1A - 2C - 3E
- 1A - 2B - 4D



Chapter 4: Test Techniques

Categories of test techniques and their characteristics



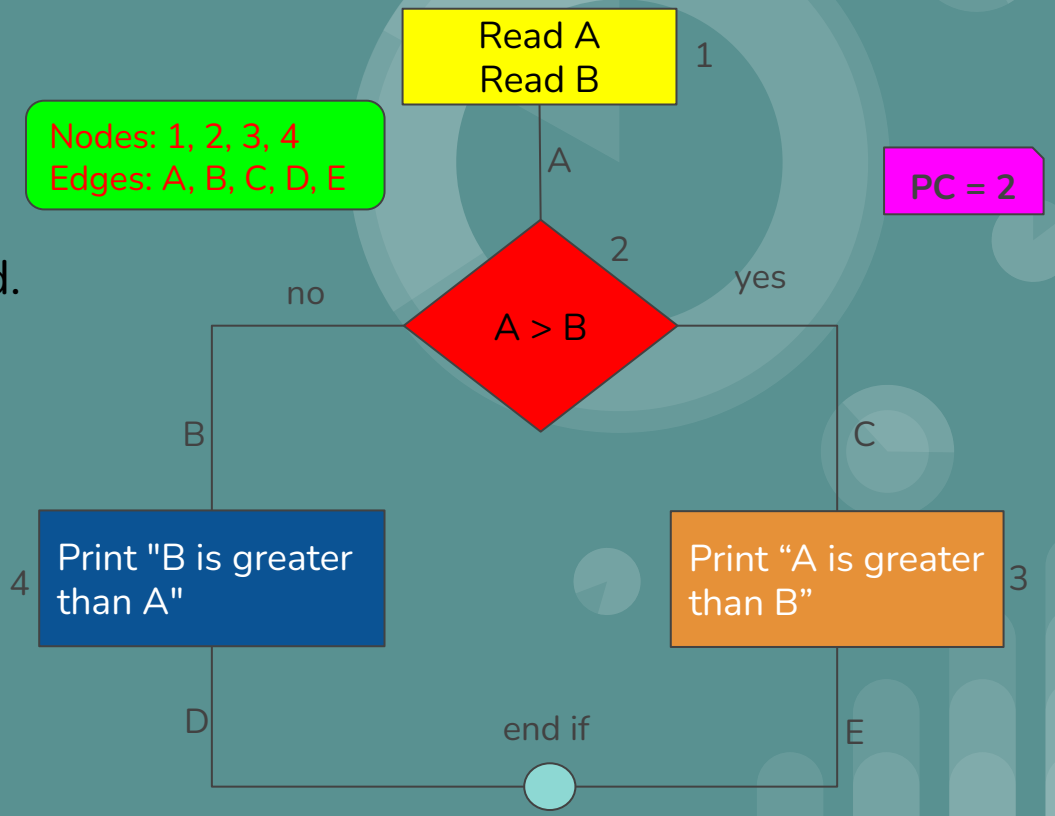
Chapter 4: Test Techniques

Path Coverage (PC)

Path Coverage ensures covering of all the paths from start to end. All possible paths are-

1. 1A - 2C - 3E
2. 1A - 2B - 4D

So path coverage is 2.



Chapter 4: Test Techniques

Other structure-based techniques

- ★ Linear code sequence and jump (LCSAJ) coverage
- ★ Condition coverage
- ★ Multiple condition coverage (also known as condition combination coverage)
- ★ Condition determination coverage (also known as multiple condition decision coverage or modified condition decision coverage, MCDC).

Chapter 4: Test Techniques

Some points of Coverage

- 100% Path coverage will imply 100% Branch/Decision coverage
- 100% Branch/Decision coverage will imply 100% Statement coverage
- 100% Path coverage will imply 100% Statement coverage
- 100% LCSAJ coverage will imply 100% Branch/Decision coverage

➢ Stronger to weaker:

Path coverage (PC) -> Branch/Decision coverage (BC/DC) -> Statement coverage (SC)

LCSAJ = Linear Code Sequence and Jump

Chapter 4: Test Techniques

The value of Statement and Decision testing

Some prime reasons for using code coverage:

- ❏ It helps you to measure the efficiency of test implementation.
- ❏ It offers a quantitative measurement.
- ❏ It defines the degree to which the source code has been tested.

100% Branch/Decision coverage ensures 100% Statement coverage

Chapter 4: Test Techniques

The Value of Statement and Decision Testing

The value of Statement coverage Testing /1

- Dead Code/Unexecuted code detection: Statement coverage identifies code sections that have not been run during testing, potentially pointing to untested functionality or dead code.
- Basic quality indicator: High statement coverage indicates that a significant percentage of the code has been used during testing, according to the basic quality indicator.
- Complete code coverage: Statement coverage aims to run every statement in the code, ensuring a higher level of code coverage than other testing methods, improving the chance of finding flaws and errors.

Chapter 4: Test Techniques

The Value of Statement and Decision Testing

The value of Statement coverage Testing /2

- Increased reliability: Statement coverage testing lowers the likelihood of unforeseen flaws, improving the software's overall dependability and quality.
- Aid in debugging: A failure during statement coverage testing reduces the range of likely reasons, making debugging easier and problem resolution more rapid.

Chapter 4: Test Techniques

The Value of Statement and Decision Testing

The value of Decision coverage Testing

- By testing every possible branch from each decision point at least once, it guarantees that no branch will lead to any abnormality in the program's operation.
- Additionally, decision coverage can address problems that may arise with statement coverage testing.

100% Branch/Decision coverage ensures 100% Statement coverage


THANK
YOU

Chapter 4: Test Techniques

Session 09

4.4 Experience-based Test Techniques

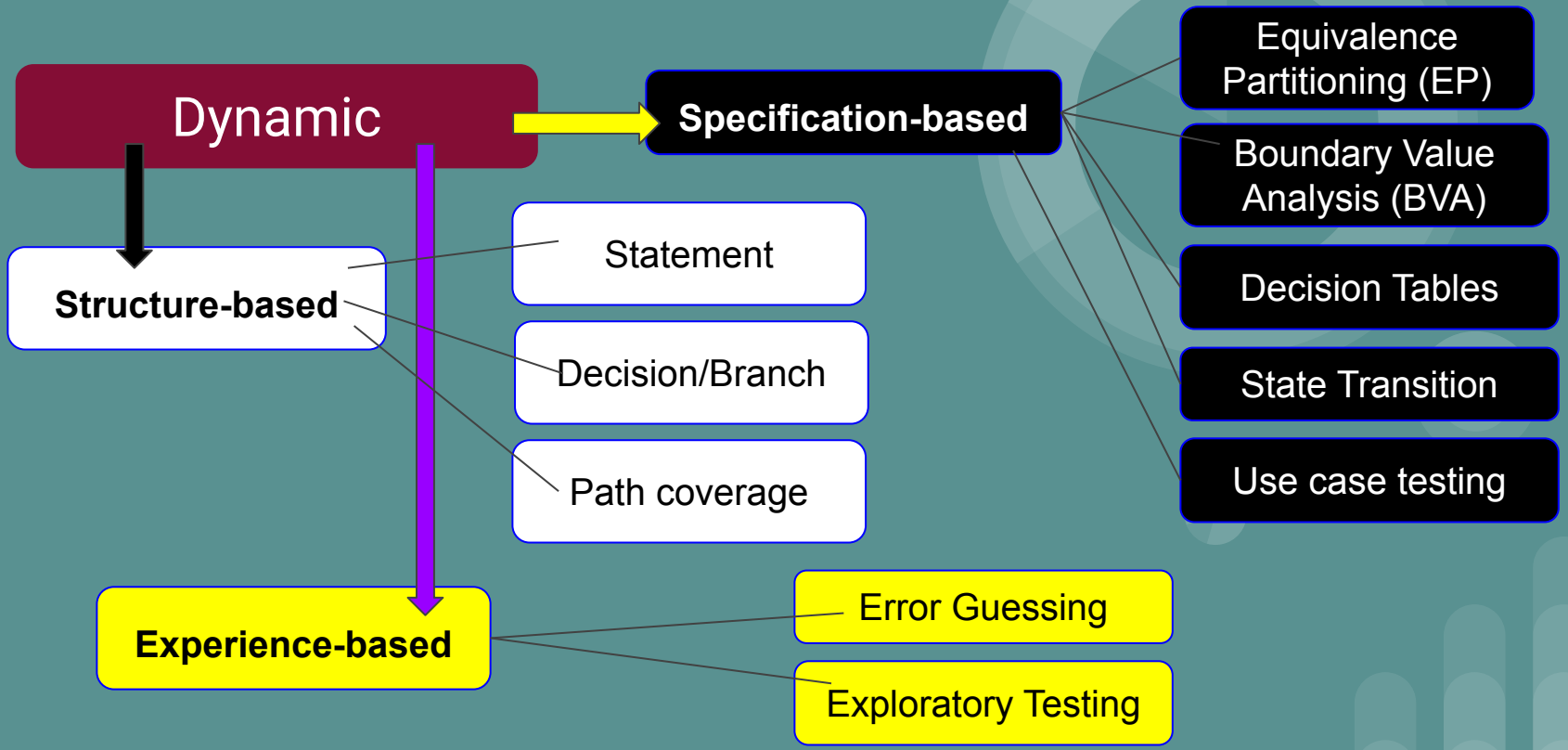
4.4.1 Error Guessing

4.4.2 Exploratory Testing

4.4.3 Checklist-based Testing

Chapter 4: Test Techniques

Categories of test techniques and their characteristics



Chapter 4: Test Techniques

Experience-based Test techniques

- It is true that testing should be rigorous, thorough and systematic.
- There is a definite role for non-systematic techniques, i.e.
  - ➔ tests based on a person's knowledge
  - ➔ Experience
  - ➔ imagination and intuition.

The reason is that some defects are hard to find using more systematic approaches, so a good 'bug hunter' can be very creative at finding those elusive defects.

Chapter 4: Test Techniques

Error Guessing /1

Error guessing may be based on:

- How the application has worked in the past.
- What types of mistakes the developers tend to make.
- Failures that have occurred in other applications.

Chapter 4: Test Techniques

Error Guessing /2

The main purpose of this technique is to identify common errors at any level of testing by exercising the following tasks:

- ➔ Enter blank space into the text fields.
- ➔ Null pointer exception.
- ➔ Enter invalid parameters.
- ➔ Divide by zero.
- ➔ Use maximum limit of files to be uploaded.
- ➔ Check buttons without entering values.

Test Levels ⇩
Unit/Component testing
Integration testing
System testing
Acceptance testing

Chapter 4: Test Techniques

Error Guessing /3

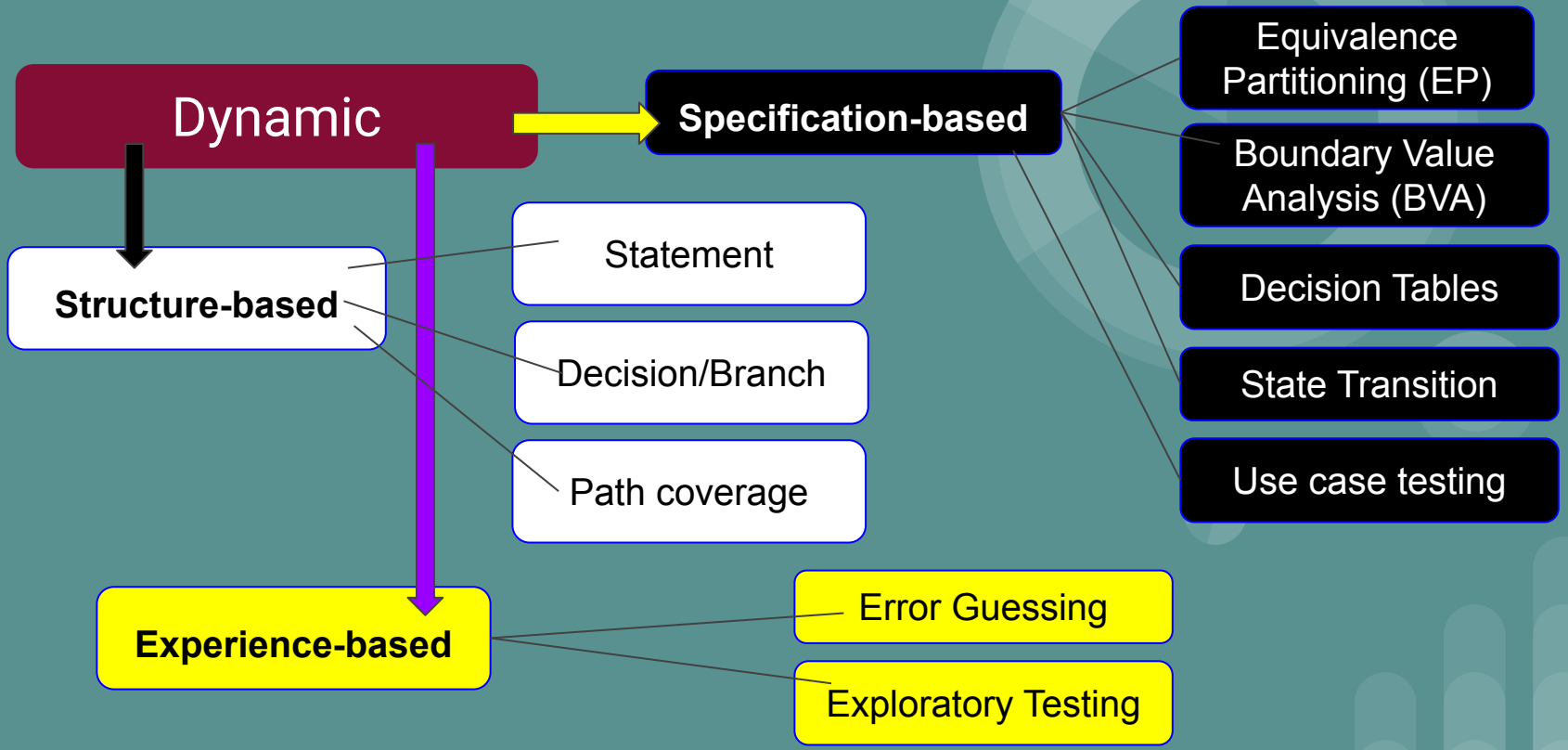
Scenario: A function of the application requires a mobile number which must be of 10 characters. Now, below are the techniques that can be applied to guess error in the mobile number field:

- ➔ What will be the result, if the entered character is other than a number?
- ➔ What will be the result, if entered characters are less than 10 digits?
- ➔ What will be the result, if the mobile field is left blank?

The increment of test cases depends upon the ability and experience of the tester.

Chapter 4: Test Techniques

Categories of test techniques and their characteristics



Chapter 4: Test Techniques

Exploratory Testing /1

- This is an approach that is most useful when there are no or poor specifications and when time is severely limited.
- Exploratory testing is a hands-on approach.
- Testers are involved in minimum planning and maximum test execution.
- The planning involves
  - the creation of a test charter.
  - a short declaration of the scope of a short (1 to 2 hour) time-boxed test effort.
  - the objectives and possible approaches to be used

Chapter 4: Test Techniques

Exploratory Testing /2

- The tester is constantly making decisions about what to test next and where to spend the (limited) time.
- The test design and test execution activities are performed in parallel typically without formally documenting the test conditions, test cases or test scripts.
- Some notes will be written during the exploratory-testing session, so that a report can be produced afterwards.

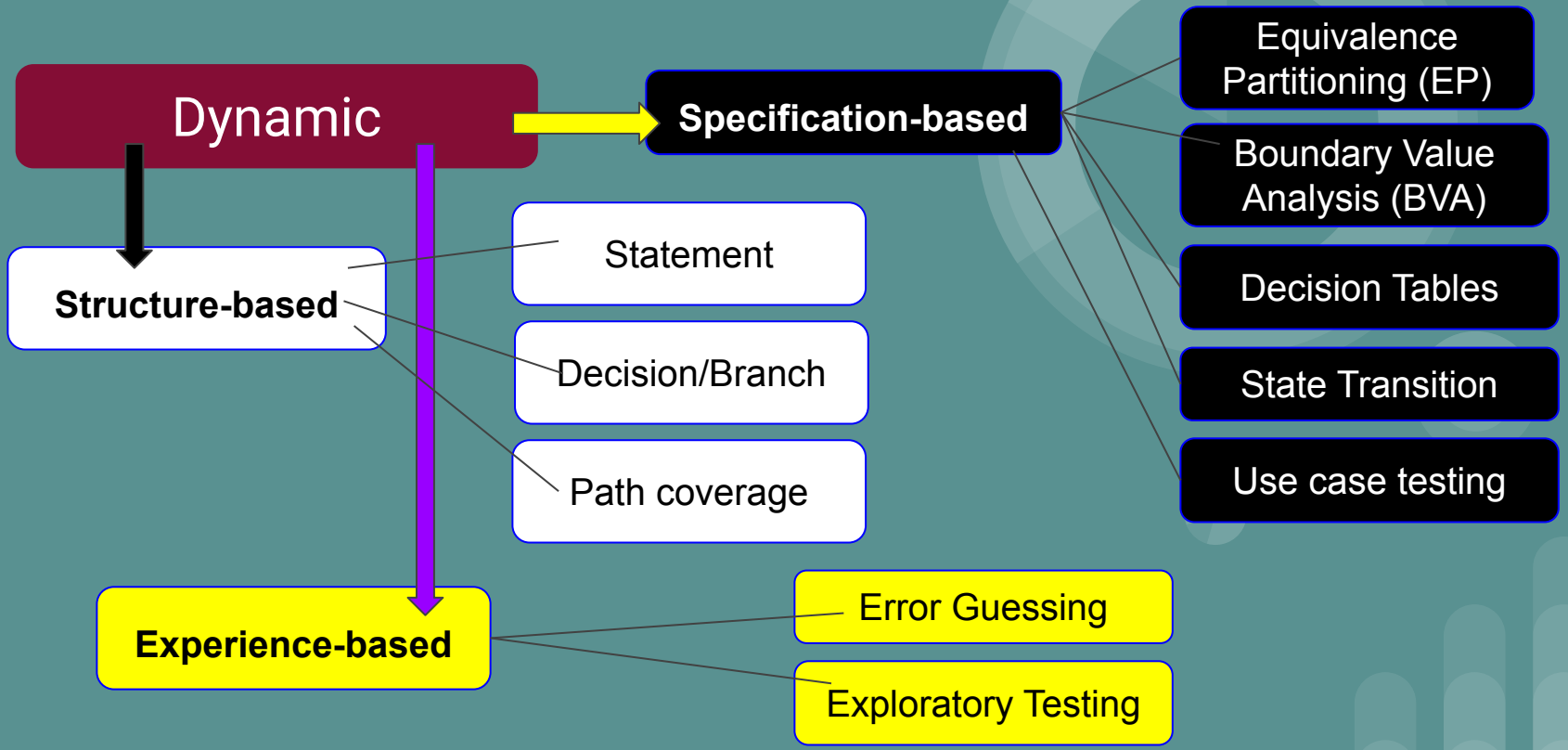
Chapter 4: Test Techniques

Exploratory Testing /3

- Exploratory testing is about exploring, finding out about the software.
  - what it does.
  - what it doesn't do.
  - what works and what doesn't work.
- It helps to establish greater confidence in the software.

Chapter 4: Test Techniques

Categories of test techniques and their characteristics



Chapter 4: Test Techniques

Checklist-based Testing /1

- Experience has been summarized and documented in a checklist.
- Testers use the checklist to design, implement and execute tests based on the items or test conditions found in the checklist.

- The checklist may be based on:
  - experience of the tester.
  - knowledge, for example what is important for the user.
  - understanding of why and how software fails.

Chapter 4: Test Techniques

Checklist-based Testing /2

- An experienced tester writes the first version of the checklist as a way to help less experienced testers do better testing.
- Checklists can be general, or more likely, aimed at particular areas such as different test types and levels. For example, a checklist for functional testing would be quite different from non-functional testing.

Test Levels	Component testing	Integration testing	System testing	Acceptance testing

Test types	functional testing	non-functional testing	structural testing	confirmation and regression testing


Thank you