SNS COLLEGE OF TECHNOLOGY

Kurumbapalayam (Po), Coimbatore – 641 035

An Autonomous Institution

Accredited by NAAC – UGC with 'A++' Grade

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COURSE NAME: 23ITO201- Software Testing (OPEN ELECTIVE)

III YEAR / VI SEMESTER

1

Unit 2 - TEST CASE DESIGN STRATEGIES

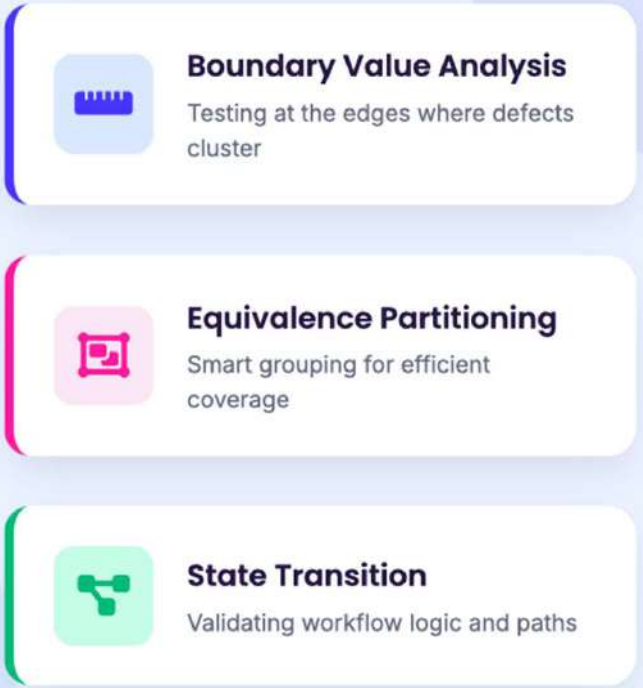
Topic : Boundary Value Analysis , Equivalence Class Partitioning and State based testing

Quality Assurance

Software Testing Techniques
Mastering the Fundamentals

A comprehensive guide to understanding and applying essential testing strategies for robust software development.

- BVA
- Equivalence Partitioning
- State Transition



Overview: Three Core Techniques

Different strategies solve different types of quality problems. Choosing the right technique maximizes testing efficiency and defect detection.

Boundary Value Analysis

- **FOCUS**
  Testing edges of ranges where logic tends to break.
- **BEST FOR**
  Numeric ranges, string lengths, dates, loops.
- **KEY WIN**
  Catches "off-by-one" errors effectively.

Equivalence Partitioning

- **FOCUS**
  Grouping inputs into valid & invalid sets.
- **BEST FOR**
  Large input spaces with identical handling.
- **KEY WIN**
  Reduces test count without losing coverage.

State Transition Testing

- **FOCUS**
  Verifying behavior as system changes states.
- **BEST FOR**
  Workflows, order processing, complex logic.
- **KEY WIN**
  Exposes invalid transitions and dead ends.

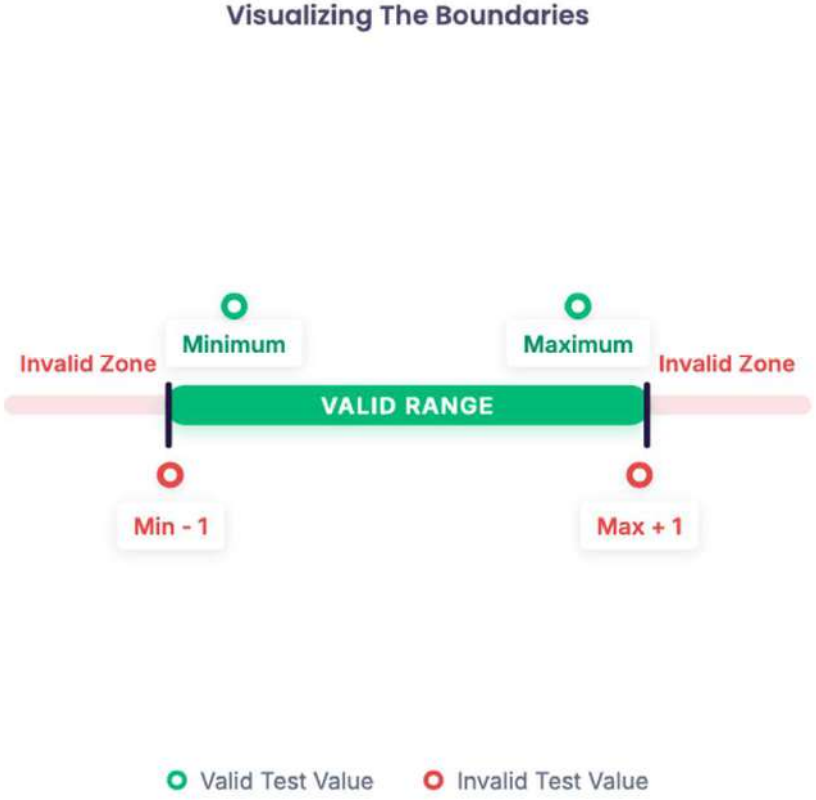
Boundary Value Analysis (BVA)

DEFINITION

A testing technique that specifically tests the **boundaries** (or edges) between valid and invalid partitions.

Why it works:

- Defects tend to cluster at the edges
- Catches "off-by-one" coding errors
- Tests inclusivity (>= vs >)



4

Why BVA is Important?

Boundary Value Analysis is a high-yield strategy because it targets the areas with the highest probability of failure.

Defects Cluster at Boundaries

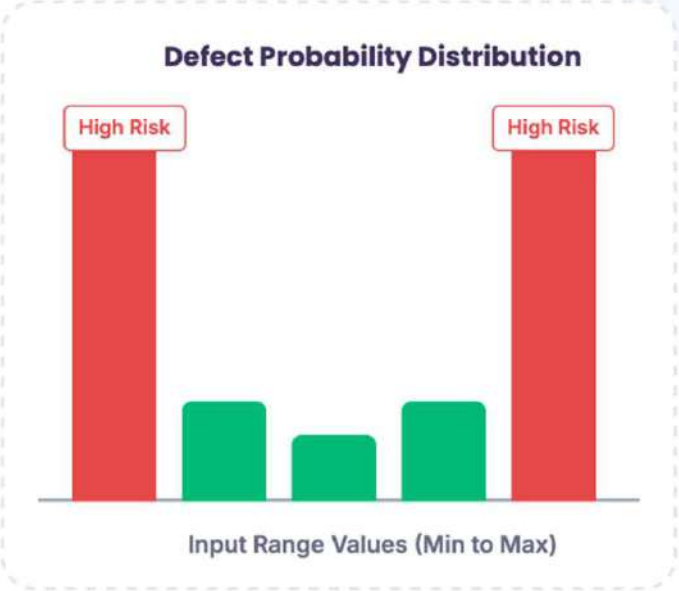
Empirical data proves that most defects occur at the edges of input ranges rather than in the center.

Prone to Developer Errors

Developers often make mistakes at limits (e.g., confusing ≤ with <), leading to off-by-one errors.

Critical for Robustness

Essential for validating critical thresholds like age limits, payment amounts, and array indices.

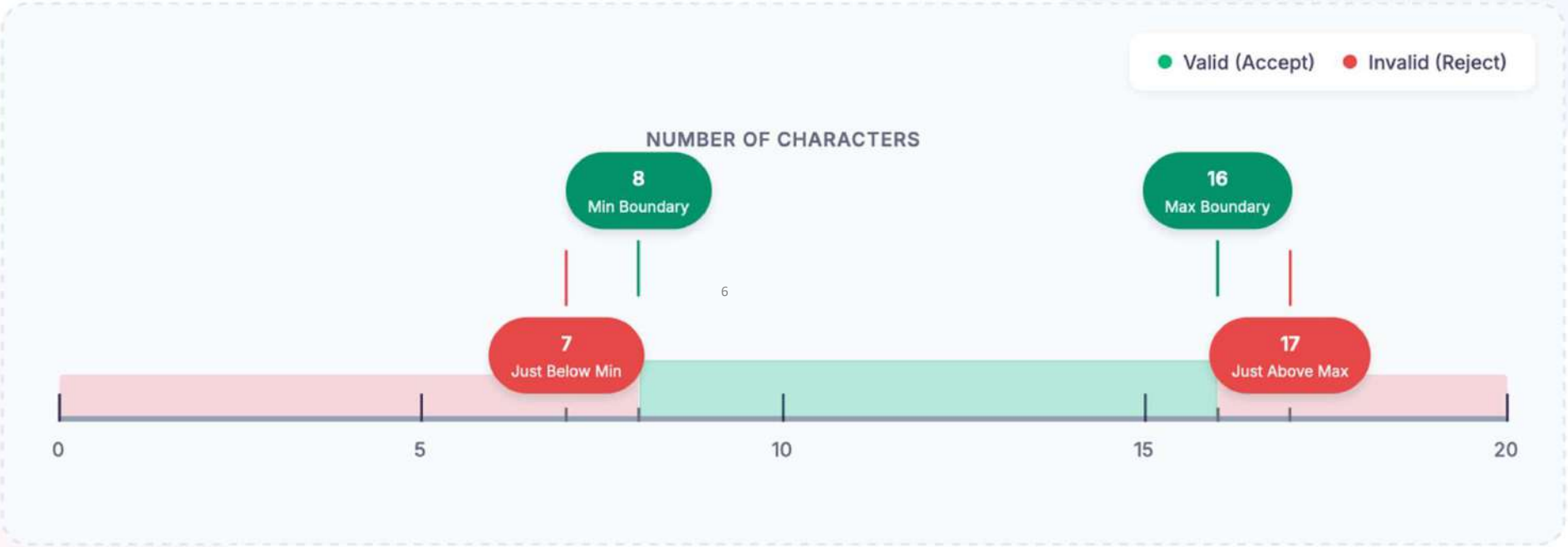


BVA Example Setup: Password Length

Defining the test scenario to identify critical boundary values.

Condition Requirement

Password length must be between **8 and 16 characters** (inclusive).



BVA Test Cases: Password Length

Verifying the boundaries for the 8-16 character requirement.

TEST CONDITION	INPUT VALUE	EXPECTED RESULT
Minimum Boundary	8	Valid
Just Below Min	7	Invalid
Maximum Boundary	16	Valid
Just Above Max	17	Invalid

7

Testing Strategy: Notice how we test the exact boundary values (8, 16) and their immediate neighbors (7, 17). This tight cluster of tests is where logical errors (like using < instead of ≤) are most likely to be exposed.

BVA: Key Points

Essential strategies to maximize defect detection with Boundary Value Analysis.

Always Test the "Core Four"

For every boundary, verify: **Minimum**, **Maximum**, **Just Below Min**, and **Just Above Max**.

Confirm Operator Logic

Verify whether boundaries are inclusive (≤, ≥) or exclusive (<, >) to prevent off-by-one errors.

Combine with Equivalence Partitioning

Use BVA values as the representatives for your Valid/Invalid partitions to optimize coverage.

Full-Stack Validation

Don't rely solely on UI limits. Ensure boundary checks exist on both **Client-side** and **Server-side**.

Why this matters

Boundaries are the most fragile part of any system logic. Focusing your testing here yields the highest return on investment for finding critical bugs.

Pro Tip: Data Types

Consider the data type! For integers, "Just above 10" is 11. For floats/currency, it might be 10.01.

Equivalence Partitioning (EP)

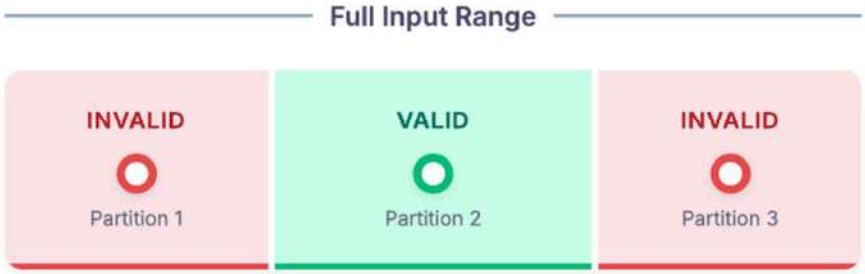
DEFINITION

A technique that divides input data into **partitions** (valid and invalid groups), assuming that all values within a group behave the same way.

Why use it?

- Reduces total number of test cases
- Avoids redundant testing (repetition)
- Ensures valid & invalid coverage

Input Domain Partitioning



Representative Value: Instead of testing every number, we pick just **one** value from each partition to represent the whole group.

Why Use EP?

Equivalence Partitioning optimizes testing by grouping similar inputs, ensuring efficiency without sacrificing quality.

Reduces Test Cases

Instead of testing every possible input value (which is impossible), EP selects just one representative from each group.

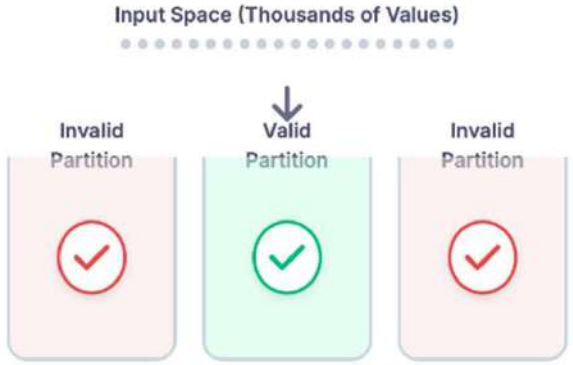
Avoids Redundant Testing

Testing multiple values from the same partition typically reveals the same bugs. EP eliminates this wasted effort.

10

Improves Coverage

By explicitly identifying all valid and invalid partitions, you ensure no logical conditions are overlooked.



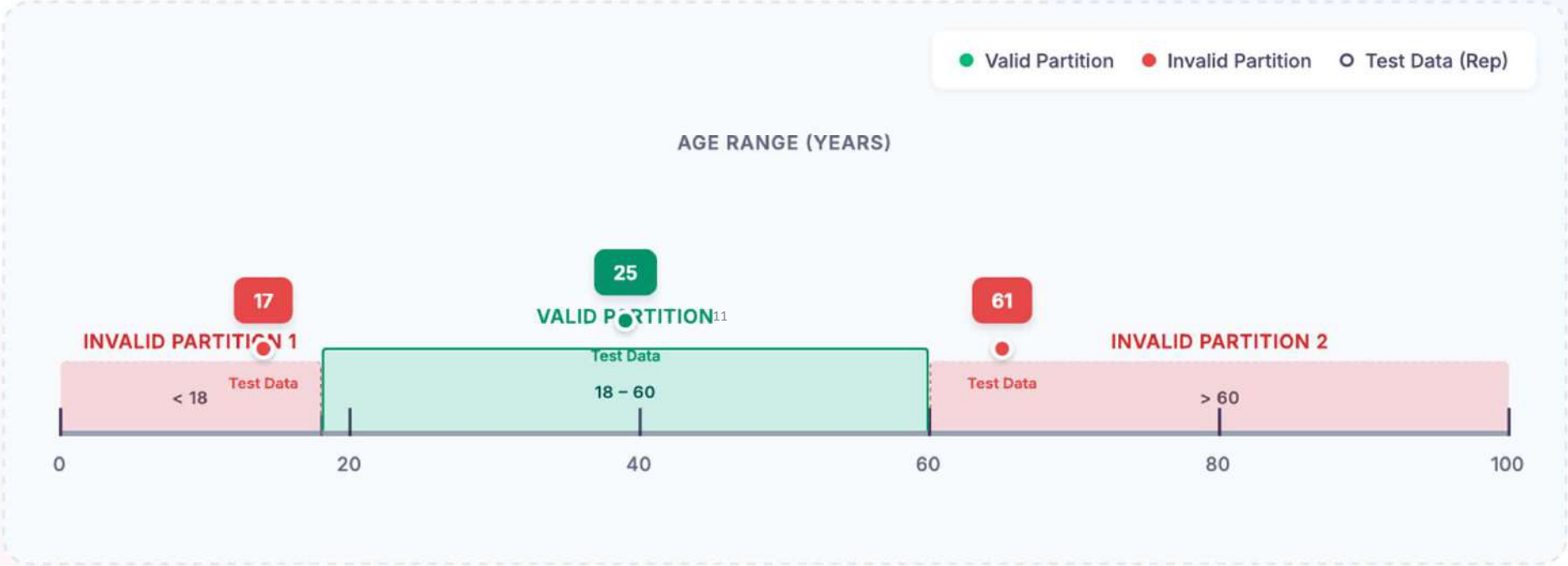
1 Representative per Partition covers the logic of thousands of inputs.

EP Example Setup: Age Validation

Defining partitions to select representative test data.

Condition Requirement

Age must be between **18 and 60 years** (inclusive).



EP Test Data: Age Validation

Selecting representative values from valid and invalid partitions (18-60 years).

PARTITION TYPE	TEST DATA	EXPECTED RESULT
Valid Partition (18-60)	25	Valid
Invalid (Below Range)	17	Invalid
Invalid (Above Range)	61	Invalid

12

Efficiency Principle: In Equivalence Partitioning, we assume that if one value (e.g., 25) passes, other values in the same group (e.g., 30, 45, 59) will also pass. This drastically reduces the number of test cases needed.

EP: Key Points

Critical concepts for effective Equivalence Partitioning.

The Golden Rule of EP

If one value in a partition fails, assume all fail. Conversely, if one passes, assume all pass. This assumption allows you to drastically reduce test volume.

Select Boundary Representatives

When picking a value from a partition, prefer boundary values (e.g., 18 or 60) over middle values (e.g., 40) to catch edge-case defects simultaneously.

13

Include Negative Partitions

Create separate partitions for invalid formats such as **null values**, **non-numeric characters**, or **special symbols**.

Efficiency is Key

Equivalence Partitioning is about trust. By scientifically grouping inputs, we trust that one test tells the story of many, eliminating redundancy without sacrificing confidence.

Pro Tip: Invalid Inputs

Always test invalid partitions individually. If you combine multiple invalid inputs in one test, you won't know which one caused the failure (masking effect).

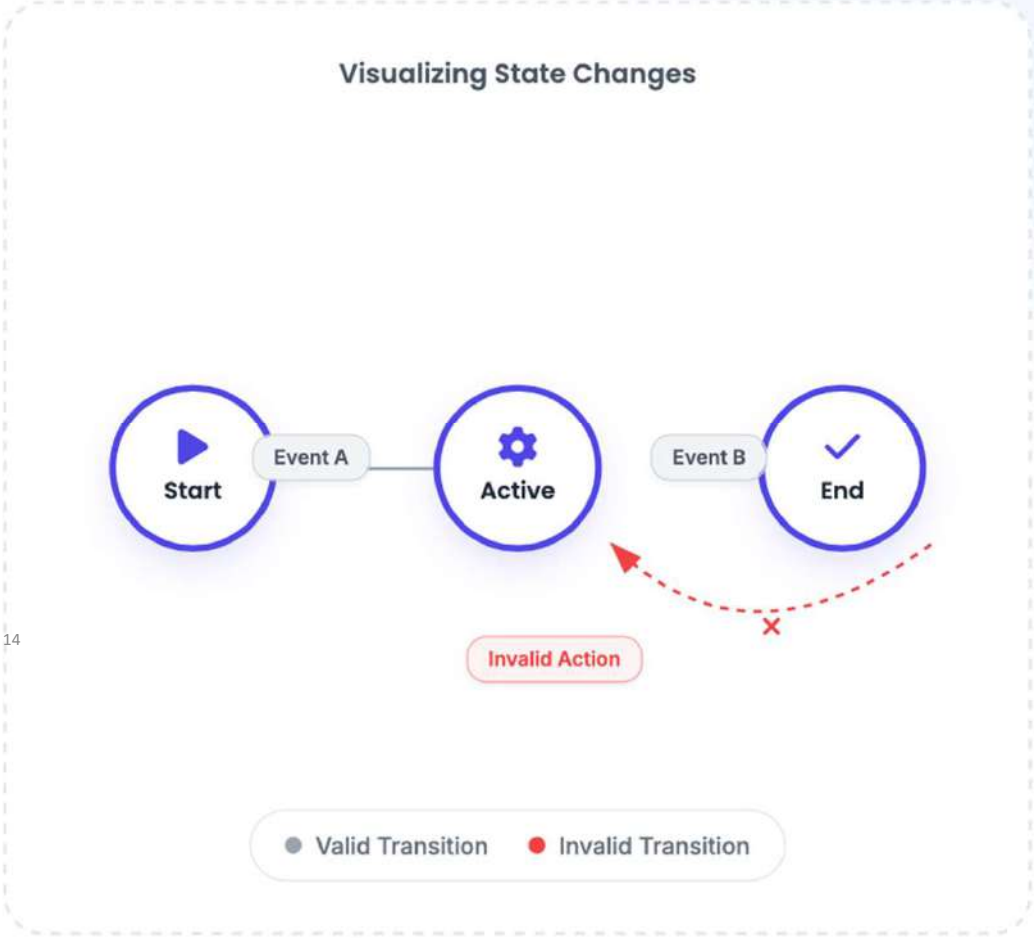
State Transition Testing

DEFINITION

A technique used to test systems with a finite number of states, verifying behavior as the system moves from one **state** to another via triggered **events**.

Why it's essential:

- Applications often change states (e.g., Login)
- Invalid transitions often cause defects
- Focuses on sequence and logic flow



Activity

Activity 2: Boundary Value Analysis (BVA)

📌 **Scenario**

Age Validation for Course Registration

Allowed Age: 18–25

✍️ **Task for Students**

Design test cases using **boundary values**

Type	Value
Just below min	17
Minimum	18
Just above min	19
Just below max	24
Maximum	25
Just above max	26

15

Test Case	Age	Expected Result
TC1	17	Rejected
TC2	18	Accepted
TC3	25	Accepted
TC4	26	Rejected

State Transition Example: Order Processing

Visualizing the lifecycle of an order to identify valid and invalid logic paths.

16

⚠ Invalid Transition Defect

Scenario: System allows changing status from "Delivered" back to "Packed".
Why it's a bug: Logically impossible workflow without a return/refund intermediate state. This breaks inventory integrity.

Valid Transition
Invalid Transition (Block This!)

MCQ

Equivalence Class Partitioning (ECP)

1. Equivalence Class Partitioning is used to:

A. Test all possible input values

B. Divide input data into valid and invalid classes

C. Focus only on boundary values

D. Test internal code logic

Answer: B

2. How many test cases are required for each equivalence class?

A. One

B. Two

C. All possible

D. None

Answer: A

Mind Map


Boundary Value Analysis (BVA)
Edge Values
Limits Testing
49 | | 50 | 100, 101
Min / Max
Equivalence Class Partitioning (ECP)
Valid Classes
Invalid Classes
10–20
30–40
Others
Decision Table Testing
Conditions & Actions
Rule Combinations
Rule
Test Case Design Strategies
Error Guessing
Tester's Intuition
Common Errors
State-Based Testing
State Changes
State A
State B
Event Flows
State-Based Testing
State A
Card Inserted
PIN Entered
State B
Idle
Enter PIN
Start
Transaction
Logout
Withdraw Cash / Logout

State Transition: Key Points

Best practices for validating system workflows, logic gates, and status integrity.

Test Valid & Invalid Paths

Don't just test the happy path. Verify that **invalid transitions** are explicitly blocked by the system.

Start & Terminal States

Ensure the process initiates correctly and always reaches a final **terminal state** (no infinite loops).

Verify Guard Conditions

Test the rules that allow a transition (e.g., "Payment Success" is required *before* "Shipping").

Robustness & Timeouts

Test behavior when unexpected events occur in each state (e.g., Back button, Network Timeout).

Workflow Integrity

Complex applications fail when users deviate from the expected flow. State transition testing is essential for catching logic holes in business processes.

Pro Tip: Visual Models

Draw a diagram first! It's much easier to spot missing arrows (transitions) or isolated states on a visual chart.

TEXT BOOKS	
1.	Srinivasan Desikan and Gopalaswamy Ramesh, ―Software Testing – Principles and Practices, Pearson Education, 2006.
REFERENCES	
1.	Ilene Burnstein, ―Practical Software Testing, Springer International Edition, 2003
2.	Edward Kit, Software Testing in the Real World – Improving the Process‖, Pearson Education, 1995
3.	Boris Beizer, Software Testing Techniques – 2nd Edition, Van Nostrand Reinhold, New York, 1990.
4.	Aditya P. Mathur, ―Foundations of Software Testing _ Fundamental Algorithms and Techniques, Dorling Kindersley (India) Pvt. Ltd., Pearson Education, 2008.

Thank You