Software Testing Techniques

BOUNDARY VALUE ANALYSIS, EQUIVALENCE CLASS TESTING, UNIT TESTING

1. Boundary Value Analysis (BVA) – What is it?

- Definition: A technique in black-box testing that focuses on the boundaries of input values.
- It assumes that errors are more likely to occur at the edges of input domains rather than the center.
- Typically used when inputs are numerical, ordinal, or bounded by specific constraints.
- The idea is to test minimum, minimum+1, nominal, maximum-1, and maximum.

2. Why Boundary Values Matter

- Many real-world bugs appear at the edges of input ranges.
- Example: If input range is 1–100, common mistakes happen at 0, 1, 100, and 101.
- Tests near these values often catch off-by-one and boundary validation errors.

3. Real-World Example – Age Validation

- System requires users to be between 18 and 60 years old.
- Test Cases: 17 (invalid), 18 (valid), 59 (valid), 60 (valid), 61 (invalid).
- Validates whether system handles age limits properly for things like registration.

4. BVA Use Case – Triangle Classification

- Inputs: Three side lengths to classify a triangle.
- Valid ranges: 1 to 200.
- Boundary test cases: (100, 100, 100), (1, 1, 1), (200, 200, 200), etc.
- Edge values help detect misclassification like 'Not a triangle' or wrong triangle type.

5. Robust Boundary Value Testing

- Extends basic BVA by adding values slightly outside valid range.
- Includes min-1 and max+1 to validate error handling.
- Used for exception handling, especially in loosely typed systems or when user input is uncertain.

6. Limitations of BVA

- Assumes input variables are independent.
- Not suitable if variable combinations affect outcome.
- Doesn't test the actual logic—only the input boundaries.

7. Equivalence Class Testing (ECT) – Introduction

- Definition: A technique where input data is divided into partitions (classes) that are treated equally.
- Assumes that if one value in the class works, others will too.
- Reduces number of test cases while ensuring coverage.

8. Types of ECT

- Weak Normal ECT: One valid input from each class.
- Strong Normal ECT: All combinations of valid inputs (Cartesian product).
- Weak Robust ECT: One invalid value per test case.
- Strong Robust ECT: Combinations of invalid values across all classes.

9. Real-World Example – Date Input

- Inputs: Day (1-31), Month (1-12), Year (1900-2099).
- Valid classes: 1–31, 1–12, 1900–2099.
- Invalid: Day <1 or >31, Month <1 or >12, Year out of range.
- Use test cases like 29/02/2024 (leap year), 31/04/2023 (invalid), etc.

10. ECT Use Case – Form Validation

- Field: Phone Number – must be 10 digits, numeric.
- Valid class: 10-digit numbers (e.g., 9876543210).
- Invalid classes: fewer digits, letters, special characters.
- Efficiently catches format errors without testing every number.

11. Mind Map – Equivalence Class Testing

- Partition input domain → Valid/Invalid classes.
- Sample from each class.
- Combine for strong normal/robust tests.
- Focuses on functionality, reduces redundancy.

12. Unit Testing – What and Why?

- Definition: Testing individual units or functions in isolation.
- Goal: Ensure that each component behaves as expected.
- Often automated using frameworks (e.g., JUnit, PyTest).

13. Anatomy of a Unit Test

- Setup: Prepare input data or mocks.
- Execution: Call the function/method.
- Assertion: Compare actual vs. expected output.
- Teardown: Cleanup if necessary.

14. JUnit Basics (Java)

- @Test: Marks a method as a test case.
- Assertions: assertEquals, assertTrue, assertNotNull, etc.
- Test suites group multiple tests for bulk execution.
- Tests can be run via IDE or command line (CI/CD pipelines).

15. Real-World Example – Calculator Testing

- Function: add(a, b) → returns a + b.
- Test Cases: (1, 1) = 2, (0, 5) = 5, (-1, -1) = -2.
- Also test for edge cases: null inputs, float handling, etc.
- Ensures reliability of math library or business logic.

16. Benefits of Unit Testing

- Catches bugs early during development.
- Supports refactoring (test stays same, code can change).
- Enables Continuous Integration (CI).
- Forms documentation for intended behavior.

17. Summary Comparison of Techniques

- BVA: Best for numeric range checking.
- ECT: Best for input validation and domain coverage.
- Unit Testing: Best for internal logic validation.
- All three are complementary in software QA strategy.