**15. Perform Black box testing using manual testing method – Boundary Value Analysis & Equivalence partitioning.**

**Black Box Testing**

Black Box Testing is a testing technique where the tester evaluates the functionality of an application without peering into its internal structures or workings. The tester inputs data and observes the output, ensuring the application behaves as expected.

**1. Boundary Value Analysis (BVA)**

Boundary Value Analysis involves testing the boundaries or edges of input ranges. Since errors often occur at the boundaries rather than within the range, it's essential to test these limits.

**Example: Age Validation (18-60)**

Let's consider an age validation feature that accepts ages between 18 and 60 (inclusive).

* **Valid Range**: 18 to 60

**Test Cases**:

* Lower boundary: 17 (invalid), 18 (valid), 19 (valid)
* Upper boundary: 59 (valid), 60 (valid), 61 (invalid)

**BVA Test Cases**:

* Test with age = 17 (just below the lower boundary)
* Test with age = 18 (exact lower boundary)
* Test with age = 19 (just above the lower boundary)
* Test with age = 59 (just below the upper boundary)
* Test with age = 60 (exact upper boundary)
* Test with age = 61 (just above the upper boundary)

**2. Equivalence Partitioning (EP)**

Equivalence Partitioning involves dividing input data into equivalent partitions where the system is expected to behave similarly. Each partition represents a set of valid or invalid states, reducing the total number of test cases required.

**Example: Age Validation (18-60)**

Continuing with the age validation example, we'll create partitions based on valid and invalid inputs.

**Valid Range**: 18 to 60

* **Equivalence Partitions**:
  + Valid Partition: [18 - 60]
  + Invalid Partition 1: [< 18]
  + Invalid Partition 2: [> 60]

**EP Test Cases**:

* Test with age = 25 (within the valid partition)
* Test with age = 17 (within the invalid partition 1)
* Test with age = 61 (within the invalid partition 2)

**Practical Implementation**

Let's see how we can apply these testing methods to a simple age validation function:

java

public class AgeValidation {

public boolean isValidAge(int age) {

return age >= 18 && age <= 60;

}

public static void main(String[] args) {

AgeValidation validator = new AgeValidation();

// Boundary Value Analysis Test Cases

System.out.println("BVA Tests:");

System.out.println("Age 17: " + validator.isValidAge(17)); // false

System.out.println("Age 18: " + validator.isValidAge(18)); // true

System.out.println("Age 19: " + validator.isValidAge(19)); // true

System.out.println("Age 59: " + validator.isValidAge(59)); // true

System.out.println("Age 60: " + validator.isValidAge(60)); // true

System.out.println("Age 61: " + validator.isValidAge(61)); // false

// Equivalence Partitioning Test Cases

System.out.println("\nEP Tests:");

System.out.println("Age 25: " + validator.isValidAge(25)); // true

System.out.println("Age 17: " + validator.isValidAge(17)); // false

System.out.println("Age 61: " + validator.isValidAge(61)); // false

}

}

**Summary**

* **Boundary Value Analysis**: Tests the boundaries of input ranges to catch edge case errors.
* **Equivalence Partitioning**: Divides inputs into partitions to reduce the number of test cases while maintaining effective coverage.