



**NANYANG
TECHNOLOGICAL
UNIVERSITY**
SINGAPORE

Tutorial #8 – Equivalence Class and Boundary Value Testing

Dr. Wang Wenhan



Equivalence Class Testing

Partition Input Values into Equivalence Classes

A set of input values form an **equivalence class (EC)** if

- They all are supposed to produce the **same output**

- If one value catches a bug, the others probably will too

- If one value does not catch a bug, the others probably will not either

Design Test Cases Using ECs

1. Test **VALID** inputs of several parameters

Test valid inputs of several parameters at the same time (i.e. per test case)

Choose one valid input for each input parameter from an equivalence class

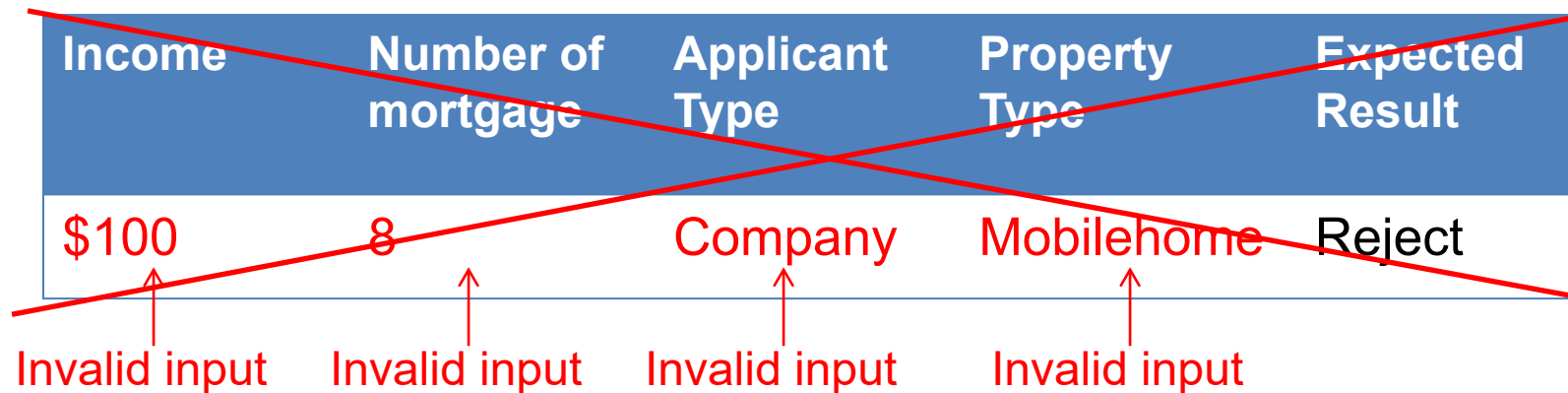
2. Test **INVALID** inputs of several parameters

Test **ONLY ONE INVALID** input from an equivalence class of one input parameter at the same time (i.e. per test case)

Choose valid inputs from an equivalence class of all other input parameters

Test Invalid Inputs Using ECs

You must test **ONLY ONE INVALID** input value from an equivalence class of one input parameter per test case, i.e., each test case should have no more than one invalid input.



Income	Number of mortgage	Applicant Type	Property Type	Expected Result
\$100	8	Company	Mobilehome	Reject

Invalid input Invalid input Invalid input Invalid input

If we test only one invalid input for each test case, then we can learn the error handling or behavior for each type of invalid input

Boundary Value Testing (BVT) – Which Values to Test?

BVT is only applicable to **range of values**

Focus on boundaries of the equivalence classes of range of values

Select a minimal set of valid and invalid inputs (values on-the-boundary, just-below, or just-above boundaries) to test

What is the boundary values of parameter Property type?

Do **discrete values**, such as property type, have boundary values?

NO

Boundary Value Testing (BVT) Process

The tester

1. Identify the **lower** and **upper** boundaries of equivalence classes of the range-of-values parameters
2. Choose three input values for each boundary
 - One value **on-the-boundary**, one value **just-below**, and one value **just-above**
 - Just-below and just-above value depends on the value's unit
 - No need to duplicate the test cases if the just-below or just-above values fall into other ECs (including current EC)
3. Determines expected outputs (oracle) for chosen inputs
4. Constructs tests with the chosen inputs (e.g., write JUnit program)

Black Box Testing – ECs and BVs

	Range of Values	Discrete Values
Equivalence Classes (ECs)	ONE Valid EC TWO Invalid ECs	ONE valid EC ONE Invalid EC

Range of values	Lower Boundary	Upper Boundary
Boundary Values (BVs) <i>(for each EC)</i>	ONE value on-the-boundary ONE value just-below ONE value just-above	ONE value on-the-boundary ONE value just-below ONE value just-above

Discrete Values	NO Boundary Values
Boundary Values (BVs) <i>(for each EC)</i>	

Question 1

Grain Elevator System

A silo cannot accept any more grain if its temperature or humidity exceeds normal levels. Humidity ranges 0-100%, the normal humidity is 35 – 50%.

Temperature ranges -10 – 100 C, the normal temperature is 40-60 C.

- a) Determine equivalence classes and boundary values for humidity and temperature
- b) Systematically design a set of test cases to test whether silo can accept grain or not based on its temperature and humidity.

Question 1 (a) - Answer

- Humidity Equivalence class: 0-34 (invalid), 35 – 50 (valid), 51-100 (invalid)
- Valid equivalence class
 - Value on boundary values: 35, 50
 - Value just below the boundary: 34;
 - Value just above the boundary: 51
- Invalid equivalence classes
 - Value on boundary values: 0, 34
 - Value just below the boundary: -1;
 - Value just above the boundary: 35
 - Value on boundary values: 51, 100
 - Value just below the boundary: 50;
 - Value just above the boundary: 101

Question 1 (b) - Answer

Valid Humidity Boundary Values (35, 50) + Valid Temperature Boundary Values (40, 60)

- 35 40 Accept
- 35 60 Accept
- 50 40 Accept
- 50 60 Accept

Invalid humidity (0, 34, 51, 100) + valid temperature (two valid boundary values: 40, 60)

- | | |
|----------------|---------------|
| • 34 40 Reject | 0 40 Reject |
| • 34 60 Reject | 0 60 Reject |
| • 51 40 Reject | 100 40 Reject |
| • 51 60 Reject | 100 60 Reject |

Question 1 (b) - Answer

Valid humidity (two boundary values: 35, 50) + invalid temperature (-10, 39, 61, 100)

- 35 39 Reject 35 -10 Reject
- 50 39 Reject 50 -10 Reject
- 35 61 Reject 35 100 Reject
- 50 61 Reject 50 100 Reject

Question 1 (b) - Answer

If assume the sensor can return invalid data (humidity \leq -1 or humidity \geq 101; temperature \leq -11 or temperature \geq 101), need to test the following cases:

- -1 40 Exception: invalid data
- 101 40 Exception: invalid data
- -1 60 Exception: invalid data
- 101 60 Exception: invalid data

- 35 -11 Exception: invalid data
- 50 -11 Exception: invalid data
- 35 101 Exception: invalid data
- 50 101 Exception: invalid data