Module 1d: Introduction to Problem Solving and Python Fundamentals

Premanand S

Assistant Professor,
School of Electronics and Engineering,
Vellore Institute of Technology, Chennai

premanand.s@vit.ac.in

July 19, 2025

What Are Test Cases?

- Test cases are inputs used to verify if a program behaves as expected.
- They ensure correctness, reliability, and stability of the code.
- Key test types: Normal, Boundary, Robust, Special/Edge

Objective

`Understand the role of test cases in validating a program's correctness and error handling.`

Boundary Test Cases

- Focus on the extreme limits of valid input.
- Used to check if boundaries are handled correctly.
- Common sources of bugs in real-world programs.

Example

```
Valid Age Range: 18 to 60

Boundary Test Cases:
 18  (lower limit)
 60  (upper limit)
 17, 61  (just outside)
```

Robust Test Cases

- Include values outside valid input range.
- Aim to test program's ability to handle invalid input.
- Check exception handling and error messages.

Example

```
Valid Age Range: 18 to 60

Robust Test Cases:
 -5 (negative)
 200 (beyond upper limit)
 "abc", None (wrong types)
```

Combined Example — Marks Input (0 to 100)

- Test case examples for a program accepting student marks.

Test Case Table

```
Normal        : 45         → Valid
Boundary      : 0, 100     → Valid
Robust        : -10, 101   → Error
Invalid Type  : "fifty"    → Error
```

Programming Perspective — Validation Logic

- Test cases help validate input and prevent runtime errors.

Python Example

```
def validate_age(age):
    if isinstance(age, int) and 18 <= age <= 60:
        return "Valid"
    return "Invalid"

test_cases = [18, 60, 17, 61, -5, "abc", None]
```

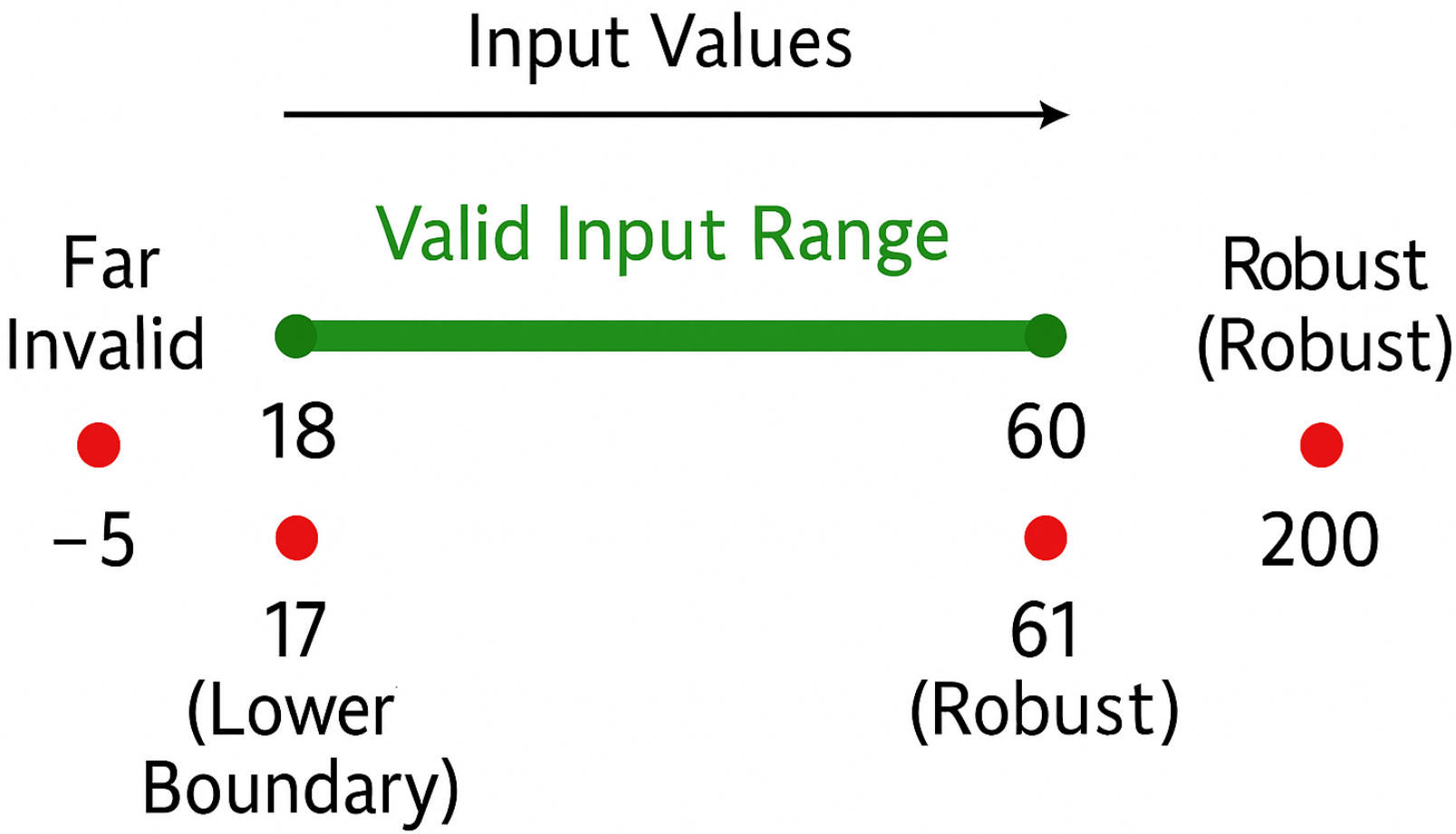
Summary: Boundary vs. Robust

- **Boundary Test Cases:** Test values at the edge of valid input.
- **Robust Test Cases:** Test values outside the valid input domain.
- Both are essential for defensive programming and reliable software.

When to Use

```
Use Boundary Testing → To confirm limits behave correctly.
Use Robust Testing   → To ensure your program handles error
```

Visualizing Boundary vs Robust Test cases



Assignment 1: Student Marks Validator

- Write a Python function that accepts a student's mark (0–100).
- Return whether it's "Valid" or "Invalid".
- Design and implement:
  - 5 Boundary test cases
  - 5 Robust test cases (invalid inputs)

Example Inputs

```
Boundary: 0, 100
Robust: -10, 150, "A", None, 101
```

Assignment 2: Age Eligibility Check

- Create a program to check voting eligibility (18–60 years).
- Write a validation function.
- Design and test:
  - Boundary inputs: 18, 60
  - Robust inputs: 17, 61, -1, "twenty", None

Objective

Ensure your program handles both valid and invalid age inputs effectively.

Assignment 3: Password Length Validator

- Password length should be between 8–16 characters.
- Write a function to check validity.
- Test using:
  - Boundary: 8, 16
  - Robust: "", 0, 7, 17, 100

Expected Output

```
"Acceptable" or "Invalid Password"
```

Assignment 4: ATM Withdrawal Validator

- ATM accepts amounts 100–10000 (multiples of 100).
- Validate withdrawal amount.
- Test Cases:
  - Boundary: 100, 10000
  - Robust: 99, 10001, 550, -500, "five hundred"

Objective

Ensure the program restricts out-of-range and non-integer/multiple values gracefully.

Assignment 5: Design Test Case Table (No Code)

- Input fields:
  - Age (0–120)
  - Temperature (35.0–42.0 °C)
  - Gender ("Male", "Female", "Other")
- Create test case table:
  - At least 3 boundary cases for each
  - At least 3 robust (invalid) cases

Deliverable

```
Test Input | Expected Output | Test Type (Boundary/Robust)
```

Automated Test Runner (Group Project)

- Build a Python utility that:
  - Accepts a function and list of test cases
  - Runs test cases and prints:
    - Pass
    - Fail
    - Error (for invalid input)
- Document with sample functions and results

Outcome

Simulate how boundary and robust tests are used in real-world software testing pipelines.