Blackbox

Boundary Analyse for Address:

Street (length 2–47; alphabetic text)

Equivalence classes

- Valid: alphabetic (incl. spaces/æøå allowed if you wish), length [2..47]
- Invalid: length <2 or >47, or empty

Boundary test set

- min−1 (invalid): "" (length 0), "A" (1)
- min (valid): "Aa" (2)
- min+1 (valid): "Aaa" (3)
- nominal (valid): "Primulavej" (~10)
- max−1 (valid): "<46-char-name>" (46)
- max (valid): "<47-char-name>" (47)
- max+1 (invalid): "<48-char-name>" (48)

Example strings

- 46 chars: "ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUV"
- 47 chars: "ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVX"
- 48 chars: "ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVXY"

Number (1–999, optionally followed by one uppercase letter)

Equivalence classes

- Valid: 1..999, or 1..999 + single A–Z
- Invalid: 0, negative, >999, letters-only, letter before digits, more than one letter, spaces inside

Boundary test set

- min−1 (invalid): "0", "0A"
- min (valid): "1", "1A"
- min+1 (valid): "2", "2Z"
- nominal (valid): "43", "43B"
- max−1 (valid): "998", "998C"
- max (valid): "999", "999D"
- max+1 (invalid): "1000", "1000A"
- format invalids: "A1", "12ab", "12 A", " 12A", "12Å" (non-ASCII letter if your rule is ASCII only)
- Containing: I, J, O, Q

Floor (either "st" or 1–99)

Equivalence classes

- Valid: "st", 1..99 (string or number)
- Invalid: 0, negative, >99, non-integer, other strings

Boundary test set

- min−1 (invalid): "0", 0
- min (valid): "1", 1
- min+1 (valid): "2", 2
- nominal (valid): "5", 5
- max−1 (valid): "98", 98
- max (valid): "99", 99
- max+1 (invalid): "100", 100
- format invalids: "st ", "ST", "kl", "k2", "1.5", "1a"

Door

Equivalence classes

- Valid (labels): "th", "tv", "mf"
- Valid (numeric): 1..50 (no leading zeros)
- Valid (pattern): ^[a-z](-)?\d{1,3}$
- Invalid: numbers 0 or >50, uppercase letter, missing digits after letter, >3 digits, multiple letters, wrong order, spaces

Boundary test set

Label branch

- Valid: "th", "tv", "mf"
- Invalid: "TH", "Tv", "mf " (case/whitespace)

Numeric branch

- min−1 (invalid): "0"
- min (valid): "1"
- min+1 (valid): "2"
- nominal (valid): "17"
- max−1 (valid): "49"
- max (valid): "50"
- max+1 (invalid): "51"
- format invalids: "01" (leading zero if disallowed), "3 " (space)

Pattern branch ([a-z][-]?\d{1,3})

- letter-only (invalid): "a"
- min digits (valid): "a1"
- min with dash (valid): "a-1"
- two digits (valid): "b12", "b-12"
- three digits (valid, max): "c123", "c-123"
- three+1 (invalid): "d1234", "d-1234"
- uppercase (invalid): "A12", "A-12"
- multiple letters (invalid): "ab12"
- dash-only (invalid): "e-"
- wrong order (invalid): "12a", "12-a"
- spaces (invalid): "f -12", "f- 12"

Postal code & town

- Postal code: always 4 digits
- Town: letters-only, length ≥1 (you said “unspecified amount”)

Equivalence classes

Postal code

- Valid: ^\d{4}$
- Invalid: length ≠ 4, non-digits, leading/trailing spaces

Town

- Valid: letters-only (define whether to allow spaces & Danish letters; examples below assume A–Z + æøå and spaces allowed)
- Invalid: empty, contains digits/symbols if not allowed, leading/trailing spaces if not trimmed

Boundary test set

Postal

- short (invalid): "999", "12"
- exact (valid): "0001", "2100", "8000"
- long (invalid): "12345"
- non-digit (invalid): "12A4", " 2100"

Town

- empty (invalid): ""
- min (valid): "A" (or "Å" if allowing Danish letters)
- nominal (valid): "Odense", "Sønderborg"
- with space (valid if allowed): "Frederiksberg C"
- invalid chars: "Aarhus2", "Nørre*”

Equivalence Partitioning for API tests

In our Postman API test I have used the techniques from Equivalence Partitioning to logically divide up how my method should be tested. The API endpoint should return - and I quote - “*Return fake person information in bulk (all information for 2 to 100 persons)*”

Valid partition:

- Integers between 2 and 100

Non-valid partitions:

- Integers below 2
- Integers above 100
- Non-integer numbers (edge case)
- Non-numeric characters (edge case)

I have therefore concluded that the following numbers could be valid for testing.

Valid values (positive testing)

- 51

Non-valid values (negative testing)

- -50
- 0
- 150
- 550.5
- “EK”