Solutions

Each answer shows the steps, then the **result**.

2.1

- validation checks the data is **sensible** (against rules) before storing
- verification checks the data was **entered/transferred correctly** (not changed)

2.2

(a) **range** check

(b) **format** check

2.3

- the data (e.g. password) is entered **twice** and the two copies are compared; if they differ, there was a typing error

2.4

- `1011001` has **four** 1s (already even), so the parity bit is `0`

3.1

- any three with examples, e.g. range (month 1–12); length (a 6-character code); type (digits only in a phone field); format (email contains `@`); presence (required field not blank)

3.2

- validation only checks the data is the right **format/range**
- a value can be sensible and correctly formatted yet **factually wrong** (e.g. "Bob" instead of "Bib"), which validation cannot detect

3.3

(a) a parity bit makes the number of 1s even (or odd) when sent; the receiver re-counts the 1s

- if the count no longer matches, a bit was flipped — an error is detected

(b) it misses an **even number** of flipped bits (e.g. two), which leave the parity unchanged

3.4

- **double-entry** verification
- it checks the two typed copies match, catching a typing error before the password is saved

3.5

- weighted sum of the digits

$$5 \times 5 + 1 \times 4 + 3 \times 3 + 2 \times 2 = 25 + 4 + 9 + 4 = 42$$

- remainder modulo 11, then 11 minus that remainder

$$42 \bmod 11 = 9 \Rightarrow \text{check digit} = 11 - 9 = \mathbf{2}$$

- check: $42 + 2 = 44 = 4 \times 11$

3.6

(a) **data entry**: double entry and visual check

- **data transfer**: parity check and checksum

(b) **validation** is an automatic check that data is **reasonable** — the right type, in range, the right format

- **verification** checks that data has been **copied or transmitted accurately**, that it matches the source

(c) a **check digit**

- transposing two digits changes the weighted sum and so changes the digit that should follow
- a range check passes both values: 14623 and 41623 are each five-digit numbers within the allowed range

(d) the check digit is calculated from the other digits by a fixed rule — often a weighted sum reduced modulo a number

- it is appended to them
- whenever the number is entered or received, the rule is applied again to the leading digits and compared with the carried digit; a mismatch means an error

(e)

```
FUNCTION Generate(Original : INTEGER) RETURNS INTEGER
    DECLARE Total, Temp, CheckDigit : INTEGER
    Total ← 0
    Temp ← Original
    WHILE Temp > 0
        Total ← Total + (Temp MOD 10)
        Temp ← Temp DIV 10
    ENDWHILE
    CheckDigit ← Total MOD 10
    RETURN Original * 10 + CheckDigit
ENDFUNCTION
```

- header with parameter and return type
- digit-extraction loop using `MOD 10` and `DIV 10`
- running total
- `CheckDigit ← Total MOD 10`
- appending it with `* 10 +`