MODULE O · OCR J277 2.3.1, 2.3.2 · CAMBRIDGE 0478 7.1, 7.5, 7.6, 7.8

Robust programs

Anticipating misuse, **validation**, **verification**, **authentication**, **testing**, **syntax and logic errors**, **test data** and **test plans**. **Cambridge 0478: you need cards 02, 03, 05, 06, 07 and 08**, without the GCSE-only rows. **OCR J277: skip card 03.** Spec versions as checked in November 2026.

01 Anticipating misuse (O1) GCSE only

Defensive design Cater for all likely inputs: anticipating misuse, authentication, input validation, maintainability.

Accidental Wrong type, out of range, blank, too long, menu choice that does not exist.

Deliberate Guessing passwords again and again; malicious input such as SQL injection.

Deal with it Reject, explain what is wrong, ask again. Never crash or store it.

Guard Check a divisor is not 0 and an array index is in range first.

02 Input validation (O2) · both boards

Validation Automatic check that input is sensible and follows rules. Not that it is correct.

Checks Range (1 to 21), type (whole number), presence (not blank), length (max 30), format (B07).

Loop Input, check, error message, input again, until valid. An IF checks once only.

IGCSE only Six checks: add the check digit, calculated from the other digits (ISBN, barcode); recalculated to detect entry errors.

GCSE only No fixed list of check names; practical experience of designing validation.

03 Validation vs verification (O3) IGCSE only

Verification Checks data was entered or copied accurately: it matches the original.

Why Valid data can be wrong: 36 typed for 63 passes a range check of 1 to 100.

Visual check Data shown on screen, compared by eye with the original; confirm Y or re-enter.

Double entry Entered twice and compared; if different, enter again. `UNTIL Phone1 = Phone2`

04 Authentication (O4) GCSE only

Authentication Confirms the identity of a user before access is given.

Simple Compare username AND password with stored values; both must match.

Attempts `while attempts < 3 AND loggedIn == false`; lock after 3 failures to stop guessing.

Stronger Do not say which item was wrong; strong passwords; a second factor.

05 Why we test (O5) · both boards

Purpose Find and remove errors; show it works as intended and meets the requirements; copes with invalid input.

Iterative Each module tested as it is written; fix, re-test, repeat. Finds errors early.

GCSE only Final/terminal testing: the whole program at the end of production; modules work together.

IGCSE only PDLC: coding stage = code + iterative testing; testing stage = testing with test data.

06 Errors in programs (O6) · both boards

Find and fix Wrong operator (> for >=), AND/OR swapped, loop count off by one, total reset in a loop, wrong variable.

Tools Trace tables; test data with known expected results.

GCSE only Syntax error: breaks the grammatical rules, so it cannot run or be translated. Logic error: runs, but unexpected output.

IGCSE only Identify the error and suggest a correction in pseudocode; logic matters more than syntax.

07 Choosing test data (O7) · both boards

Normal Both boards: data that should be accepted. Range 1 to 24: 10.

GCSE only Boundary: correct type, very edge of valid (1, 24). Invalid: correct type, rejected (0, 25). Erroneous: wrong type, rejected ("six").

IGCSE only Extreme: largest/smallest accepted (1, 24). Boundary: that value and the rejected one next to it (0 and 1; 24 and 25). Abnormal: rejected (0, 25, "six").

Values Always an actual value, never "a number in range".

08 Writing a test plan (O8) · both boards

Columns Test number, what is tested, test data, type, expected result; then actual result, pass/fail.

Expected Specific: "Accepted, cost £5" or "Rejected, error message".

Cover Every rule, both edges of every range, every route through each IF.

Fail Fix the code, then run the tests again. 10 miles gave £10, not £5: `< 10` becomes `<= 10`.

GCSE only Create/complete a test plan; refining algorithms after failed tests.

IGCSE only Suggest and apply suitable test data: value, type, expected outcome.

09 Same test data, two sets of names: a stay of 1 to 24 hours

10	GCSE: normal · IGCSE: normal · accepted
1 and 24	GCSE: boundary · IGCSE: extreme (each also half of a boundary pair) · accepted
0 and 25	GCSE: invalid · IGCSE: abnormal (each also half of a boundary pair) · rejected
"six"	GCSE: erroneous · IGCSE: abnormal · rejected

01
Validation checks data is **sensible**, not **correct**. Verification (IGCSE) checks it **matches the original**.

02
Validate with a **loop**, not a single IF, so a second bad input is caught.

03
Boundary means different things: OCR, the edge values; Cambridge, a **pair** (accepted and rejected).

04
Program runs but gives the wrong answer: a **logic error**. Will not run at all: a **syntax error**.