MODULE K · OCR J277 2.1.1, 2.1.2 · CAMBRIDGE 0478 7.1-7.3, 7.7-7.9, SECTION 4

# Algorithm design

**Decomposition**, **abstraction**, **structure diagrams**, the **life cycle**, **pseudocode**, **flowcharts**, **trace tables**, **errors** and **explaining algorithms**. **OCR J277: every card except 04**, without IGCSE-only rows. **Cambridge 0478: every card**, without GCSE-only rows. No K2. Specs as checked in November 2026.

## 01 Thinking and IPO (K1) · both boards

**Decomposition** Split a complex problem into smaller sub-problems: easier to solve, test, share and reuse.

**Abstraction** Remove unnecessary detail. A metro map leaves out the streets.

**IPO** Inputs: data entered. Processes: what is done to it. Outputs: results.

**GCSE only** Algorithmic thinking: the steps needed, in the right order.

**IGCSE only** Add storage. Every system is made of sub-systems, made of further sub-systems.

## 03 Structure diagrams (K3) · both boards

**What** A top-down (hierarchical) diagram of a problem split into sub-problems.

**Top box** The whole problem. Each level below splits a box into its sub-problems.

**Bottom** The lowest boxes are small enough to be one module or sub program.

**Not** No decisions, loops or arrows: those belong in a flowchart.

**IGCSE only** Shows a system made of sub-systems; used in the design stage of the life cycle.

## 04 Development life cycle (K4) IGCSE only

**Analysis** Problem and requirements; abstraction; decomposition.

**Design** Structure diagrams, flowcharts, pseudocode.

**Coding** Write code; test each module as it is written.

**Testing** Normal, abnormal, extreme, boundary data.

**OCR** Not on J277. No "implementation" stage.

## 05 Pseudocode (K5) · both boards

**What** Code-like steps, not tied to one language.

**Precision** Write `score > 50`, not "score is more than 50". Indent blocks; meaningful names; `//` comments.

**GCSE only** OCR ERL: `x = 3`, `print(x)`, `endif`. Section B write: ERL or a high-level language.

**IGCSE only** `Count ← 0`, `OUTPUT Count`, `ENDIF`. Pseudocode only, except the 15-mark question.

## 06 Flowcharts (K6) · both boards

**Rounded** Terminal (GCSE) / Terminator (IGCSE): Start, Stop.

**Rectangle** Process: a calculation or assignment.

**Parallelogram** Input/Output.

**Diamond** Decision: Yes/No, two labelled exits.

**Double sides** Sub program (GCSE) / Subroutine (IGCSE): a separate flowchart.

**Arrow** Line (GCSE) / Flow line (IGCSE). Arrow back = loop.

## 07 Problem to algorithm (K7) · both boards

**Start** Inputs, processes, outputs (IGCSE only: + storage).

**Steps** Sequence; selection for decisions; iteration for repeats.

**Variables** Meaningful names and starting values (a total starts at 0).

**Finish** Write it, refine it (validation, edge cases), dry-run it.

**Example** 10 points a ticket, +25 for 4 or more: 3 tickets = 30; 4 tickets = 65.

## 08 Trace tables (K8) · both boards

**Dry run** Work through the algorithm by hand, step by step, without a computer.

**Table** A column per variable plus output; a new row each time a value changes.

**Loops** Re-test the condition on every pass; record when the loop ends.

**Example** Double 7 while it is <= 50: 14, 28, 56; steps 1, 2, 3; output 3.

**GCSE only** Often a Line column; you may write your own headings.

**IGCSE only** Record variables, outputs and user prompts at each step.

## 09 Finding and fixing errors (K9) · both boards

**Answer** Line number + what is wrong + the corrected line.

**Common** < for <=; loop end one out; total not set to 0 or reset inside the loop; counter never updated; wrong variable.

**Find it** Trace table or test data: where does the output first go wrong?

**GCSE only** Syntax error: breaks the language's rules. Logic error: runs but gives unexpected output. Fix in ERL: `==`, `!=`, `endif`.

**IGCSE only** Fix in Cambridge pseudocode: ←, <>, `ENDIF`, `NEXT Index`.

## 10 Explaining algorithms (K10) · both boards

**Purpose** One sentence on the overall result: "outputs the highest of 10 numbers".

**Processes** The inputs; what is calculated, checked or repeated; what is output.

**Patterns** Totalling, counting, max/min, searching, sorting, validation loops.

**Evidence** Name the actual variables and values; trace to confirm.

**IGCSE only** 7.3: state the purpose, describe the processes. Standard methods: linear search, bubble sort, totalling, counting, max, min, average.

**GCSE only** Identify a standard search or sort from its code (Module L).

## K How each board writes code

**Assign** OCR `total = total + x` · Cambridge `Total ← Total + X`

**Output** OCR `print("Total: " + str(t))` · Cambridge `OUTPUT "Total: ", T`

**Loop** OCR `for i = 1 to 5 ... next i` · Cambridge `FOR I ← 1 TO 5 ... NEXT I`

**Not equal** OCR `!=` · Cambridge `<>`

**Never** Mix the two styles in one answer.

**01**

Conditions as code: **`x >= 1 AND x <= 100`**, never `x >= 1 AND <= 100`.

**02**

"Or more" and "at least" mean `>=`; "over" means `>`.

**03**

Structure diagram: sub-problems. Flowchart: steps, decisions and loops.

**04**

Explain the purpose with the real variables, not each line in turn.