OCR GCSE (J277)

Module K: Algorithm design

Workbooklet mark scheme · 8 parts · 166 marks

K4 is IGCSE (Cambridge) only and not on J277.

Contents

Part	Topic	Marks
K1	Abstraction, decomposition, algorithmic thinking, and inputs, processes and outputs: mark scheme	20
K3	Structure diagrams: mark scheme	21
K5	Reading and writing pseudocode: mark scheme	20
K6	Flowcharts: mark scheme	21
K7	Turning a problem into an algorithm: mark scheme	21
K8	Trace tables and dry runs: mark scheme	21
K9	Finding and fixing errors in an algorithm: mark scheme	21
K10	Explaining what an algorithm does: mark scheme	21
	Total	**166**

MODULE K ALGORITHM DESIGN · K1

Abstraction, decomposition, algorithmic thinking, and inputs, processes and outputs: mark scheme

20 marks · OCR J277 spec 2.1.1 (principles of computational thinking: abstraction, decomposition, algorithmic thinking, and how they are used to define and refine problems) and 2.1.2 (identify the inputs, processes and outputs for a problem)

How to use. 1 mark per bullet up to the maximum shown. **/** separates alternative answers. Words in (brackets) are not needed for the mark. Credit any other valid answer that shows the same understanding. In context questions (2 to 5), answers must apply to the scenario given.

QUESTION	ANSWER	MARKS	GUIDANCE
1(a) J277 2.1.1	Removing / hiding / ignoring unnecessary detail (from a problem) / focusing only on the details that are important (to solve the problem)	**1**	Accept "simplifying a problem by removing details that are not needed". Do not accept "making it simpler" alone, or "breaking a problem down" (decomposition).
1(b) J277 2.1.1	Breaking / splitting a (complex) problem down into smaller (more manageable) problems / parts / sub-problems	**1**	Must give the idea of smaller parts / problems. Do not accept "breaking it down" alone. Do not accept "removing detail" (abstraction).
1(c) J277 2.1.1	Identifying / working out the steps (and their order) needed to solve a problem / thinking about how a problem can be solved as a set of logical steps / an algorithm	**1**	Accept "planning the sequence of instructions to solve a problem". Do not accept "thinking about algorithms" or "writing code" alone.
2(a) J277 2.1.1	Any two from, 1 mark each: Location of each docking station Number of bikes available at each docking station / whether a station has a bike Cycle lanes Roads / paths that can be cycled on / road or street names (for planning the route) The rider's current location Direction / route to the chosen station	**2**	Max 2. Credit any other detail a rider needs to find a station with a bike or to plan a route. Do not accept "the map" / "the city" alone.

QUESTION	ANSWER	MARKS	GUIDANCE
2(b) J277 2.1.1	Any two from, 1 mark each: Shapes / pictures / details of buildings House / building numbers Names of shops / businesses Trees / parks / gardens shown in detail Bus stops / train lines / car parks Exact colours / textures of road surfaces Places outside the area the bikes can be used in	**2**	Max 2. Credit any other detail not needed to find a station or plan a cycle route. Do not credit a detail that the rider needs (e.g. docking stations, cycle lanes, bike availability).
3(a) J277 2.1.2	**Inputs** (max 2): The barcode of the parcel / parcel ID (scanned) The size of the parcel (entered by the courier) **Processes** (max 2): Search for / find an empty locker of the right size Create / generate the random (six-digit) collection code Link / store the code (and parcel) with the chosen locker **Outputs** (max 2): The locker number (shown on the screen) The (signal to) open the locker door The text message / collection code sent to the customer	**6**	1 mark per correct entry in the correct row, max 2 per row. An answer in the wrong row scores 0 (e.g. "collection code is created" as an output). The same answer written twice scores once. Accept "scan the barcode" / "enter the size" as inputs.
4(a) J277 2.1.1	Any three from, 1 mark each: Show the list of shows / dates / times Show which seats are free / show the seating plan Let the customer choose / select seats Work out / calculate the total cost Take / process the payment Get / enter the customer's details (e.g. name, email address) Mark the chosen seats as booked / update the seats left Send a confirmation / ticket to the customer	**3**	Max 3. Credit any other sensible sub-problem of an online seat booking program. Do not accept vague answers such as "the website", "the code", "input" or "output" alone. Two answers that are the same task in different words score once.
4(b) J277 2.1.1	1 mark for a way, 1 mark for a linked explanation, e.g.: Each sub-problem can be given to a different programmer … so they can work on parts at the same time / the program is finished sooner Each sub-problem is smaller / simpler … so it is easier to understand / solve / write the code for Each part can be tested on its own … so errors are easier to find / fix A part (e.g. payment) can be reused … so it does not need to be written again (in this or another program)	**2**	Max 2. The second mark must expand the first. Only one way is required; if two are given, mark the better one. Do not accept "it is faster" / "it is easier" with no reason.
5(a) J277 2.1.1	B, D in the 1st and 2nd boxes E, C, A in the 3rd, 4th and 5th boxes Correct order: B, D, E, C, A	**2**	1 mark per bullet. Each letter used once only; if a letter is repeated or missing, the bullet containing that position scores 0.

Total: 20 marks

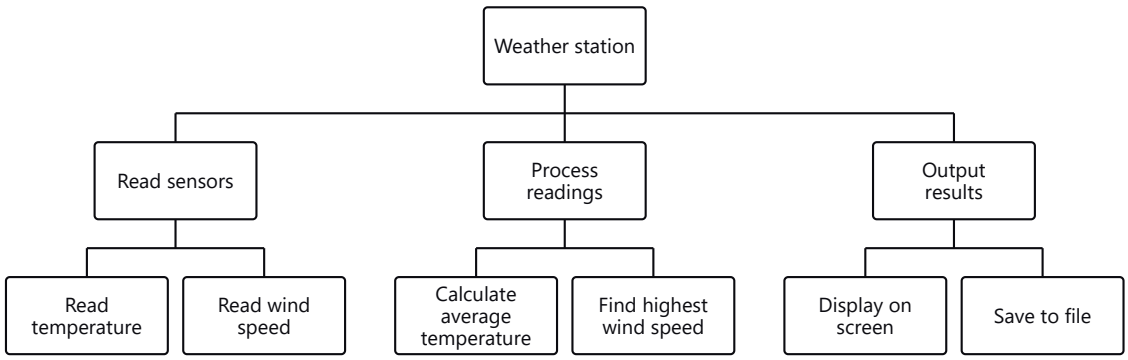
MODULE K ALGORITHM DESIGN · K3

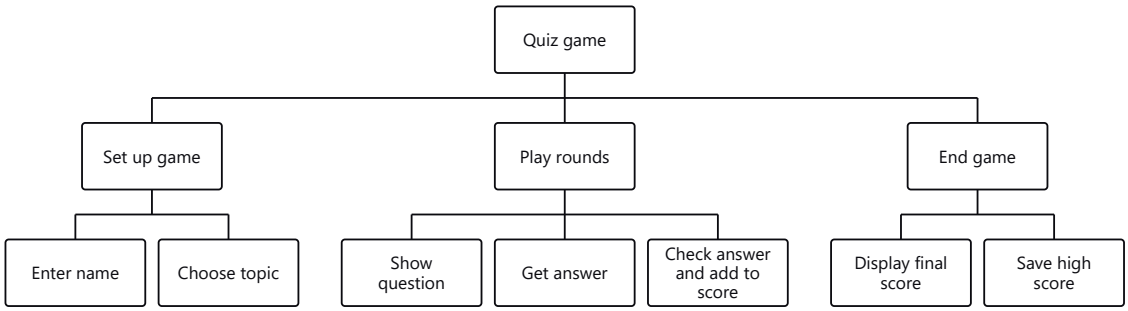
Structure diagrams: mark scheme

21 marks · OCR J277 spec 2.1.2: structure diagrams (produce simple diagrams to show the structure of a problem, subsections and their links to other subsections); identify the inputs, processes and outputs for a problem · 2.1.1: decomposition

How to use. 1 mark per bullet up to the maximum shown. **/** separates alternative answers. Words in (brackets) are not needed for the mark. Credit any other valid answer that shows the same understanding.

QUESTION	ANSWER	MARKS	GUIDANCE
1(a) J277 2.1.2	Top box represents the whole problem: True Each box broken down into sub-problems on the level below: True Diamond-shaped box for each decision: False Records the value of each variable: False	**4**	1 mark per correct row. A row with both boxes ticked scores 0. (Decisions in diamonds belong to flowcharts; variable values are recorded in a trace table.)
2(a) J277 2.1.2	How the problem is broken down / decomposed into smaller sub-problems / subsections / parts The whole problem at the top, with each level below splitting a part into smaller parts // the parts arranged in levels / a hierarchy The links between subsections // which smaller parts belong to which subsection	**2**	1 mark per bullet, max 2. Credit any other valid answer. Do not accept "the order the steps are carried out" / "the decisions" / "the flow of the program" (these describe a flowchart).
3(a) J277 2.1.2	Collect parcel	**1**	Do not accept "Parcel locker".
3(b) J277 2.1.2	The (collection) code (entered by the customer) The parcel label / barcode on the label (scanned by the courier)	**1**	Any one. Do not accept "the parcel" alone, "the locker", or a box name that is not an input (e.g. Open locker door, Send code to customer).
3(c) J277 2.1.2, 2.2.3	Write the code (to open a locker door) as one sub program / procedure / function ... and call / use it from both subsections // from both Deposit parcel and Collect parcel	**2**	1 mark per bullet. The second mark depends on the first. Accept "subroutine" / "module". Do not accept "copy and paste the code".

QUESTION	ANSWER	MARKS	GUIDANCE
3(d) J277 2.1.1	1 mark for a way, 1 mark for the matching explanation: Each programmer can be given a different subsection / box to write … so they can work at the same time / the program is finished sooner Each part is a smaller / simpler problem … so it is easier to write / solve Each part can be tested separately / on its own … so errors are easier to find / fix The diagram shows how the parts link … so the programmers know which parts must work together A part such as Open locker door can be written once … and reused	**2**	Max 2. Only one way is credited: the second mark must explain the way given. Credit any other valid way with an explanation. Do not accept "faster" / "easier" alone with no reason.
4(a) J277 2.1.2	Second-level boxes: Process readings **and** Output results Read wind speed (below Read sensors) Calculate average temperature **and** Find highest wind speed (below Process readings) Display (results) on screen **and** Save (results) to file (below Output results)	**4**	1 mark per bullet. Accept any wording with the same meaning (e.g. "Work out average temperature", "Find maximum wind speed"). The two boxes in a pair may be in either order. Follow through: if the two second-level boxes are swapped, credit each pair placed below the matching box.
4(a) Completed diagram			
5(a) J277 2.1.2	Hierarchical structure: subsections joined by lines below Quiz game, with a further level below them Three subsections for setting up, playing and ending the game (any suitable names) Enter name **and** Choose topic, below the set-up subsection At least two of: show question / get (input) answer / check answer / add 1 to score, below the playing subsection Display (final) score **and** Save high score, below the end-of-game subsection	**5**	1 mark per bullet. Accept Input / Process / Output as the subsection names if the boxes below match the description. Accept a further level (e.g. Check answer split into Compare answer and Add 1 to score). Boxes may be in any order within a level. A flowchart (steps in sequence, decision diamonds) is not a structure diagram: 0 marks.

QUESTION	ANSWER	MARKS	GUIDANCE
5(a) **Example answer**			

Total: 21 marks

MODULE K ALGORITHM DESIGN · K5

Reading and writing pseudocode: mark scheme

20 marks · OCR J277 spec 2.1.2 (create, interpret, correct, complete and refine algorithms using pseudocode and Exam Reference Language / a high-level language; identify common errors) and 2.3.1 (maintainability: naming conventions, indentation, commenting)

How to use. One mark per bullet up to the maximum shown. **/** separates alternatives; **//** separates alternative statements for the same mark. Words in (brackets) are not needed for the mark. **...** means the point depends on the one before. NE = not enough; FT = follow through. Code answers in OCR Exam Reference Language or any high-level language are marked for logic: minor syntax errors are not penalised, and a repeated slip is penalised once only. Do not accept mathematical symbols such as ≥, ≤ or ≠ in code. Credit any other valid answer that shows the same understanding.

QUESTION	ANSWER	MARKS	GUIDANCE
1(a) J277 2.1.2	Row 3 ticked only (a structured, code-like way of writing an algorithm that is not tied to one programming language)	**1**	2+ ticks = 0 marks. Row 1 is a flowchart, row 2 a trace table, row 4 a high-level language.

2(a) J277 2.1.2	1 mark for each correct condition:
	`speed > 70`
	`colour != "red"`
	`age >= 18 AND age <= 65` // `age > 17 AND age < 66`
		3	Allow `if` / `then` around the condition. Allow lower-case and; allow `NOT(colour == "red")`; allow <> for not equal; allow `18 <= age <= 65` (valid in a high-level language). Row 2: "red" must be in quotes. Row 3: do not accept `age >= 18 AND <= 65` (the second comparison has no variable), `OR` instead of AND, or =>.

3(a)(i) J277 2.1.2	02	**1**	Correct answer only. Allow 2.
3(a)(ii) J277 2.1.2	06	**1**	Correct answer only. Allow 6. Allow 06–11 given as a range.

QUESTION	ANSWER	MARKS	GUIDANCE
3(b) J277 2.1.2	1 mark for each output line: Inputs 5, 2: `1 more needed` ... `Stamps now: 7` Inputs 3, 5: `Free drink earned` ... `Stamps now: 0`	**4**	Ignore quotation marks round outputs. Ignore minor differences in spacing or capitals. Do not award an output line that includes code (e.g. `str(TARGET - stamps)`) instead of its value. Each output line is marked on its own. Outputs written in the wrong row score 0 for those lines.

3(c)
J277 2.3.1
	1 mark each, max 2:
	Comment(s) // line 01 explains what the code is for
	Indentation (inside the if and else) // lines 07, 08 and 10 are indented
	Meaningful / sensible identifiers / variable names // e.g. `stamps`, `cups` // naming conventions
	Use of a constant (`TARGET`) so the number of stamps only needs changing in one place
		2	Mark the first two answers only. Do not accept "sub programs" (there are none in this code). Do not accept "it is short / simple" (NE).

4(a) J277 2.1.2	1 mark for each line number, 1 mark for each matching correction: Line 04 ... ... `while unlocked == false AND attempts < 3` // `attempts <= 2` Line 09 ... ... `attempts = attempts + 1`	**4**	Line 04 as given allows four attempts; line 09 as given makes `attempts` go down, so the loop never ends until the code is correct. A correction only scores if its line number is correct. Allow line 02 changed to `attempts = 1` as the correction instead of line 04 (also gives exactly three attempts). Allow `NOT unlocked` or `unlocked == False`.

QUESTION	ANSWER	MARKS	GUIDANCE

5(a)
J277 2.1.2
	1 mark each, max 4:
	Count (variable) set to 0 (before the loop)
	Loop that runs exactly 10 times
	Input of a reading inside the loop, stored in a variable
	Check whether the reading is > 30 ... and if so add 1 to the count
	Output the count after the loop ends // outside the loop
	`count = 0`
	`for i = 1 to 10`
	`  reading = float(input("Enter reading"))`
	`  if reading > 30 then`
	`  count = count + 1`
	`  endif`
	`next i`
	`print(count)`
		4	Allow any logically valid method, e.g. a while or do until loop with its own counter, or `for i in range(10)`. Allow `0 to 9`. Do not penalise a missing cast. Do not award the selection mark for `>= 30` (30 is not more than 30). Do not award the output mark if the count is output every time round the loop. Flowcharts, bullet points or English score 0.

Total: 20 marks

MODULE K ALGORITHM DESIGN · K6

Flowcharts: mark scheme

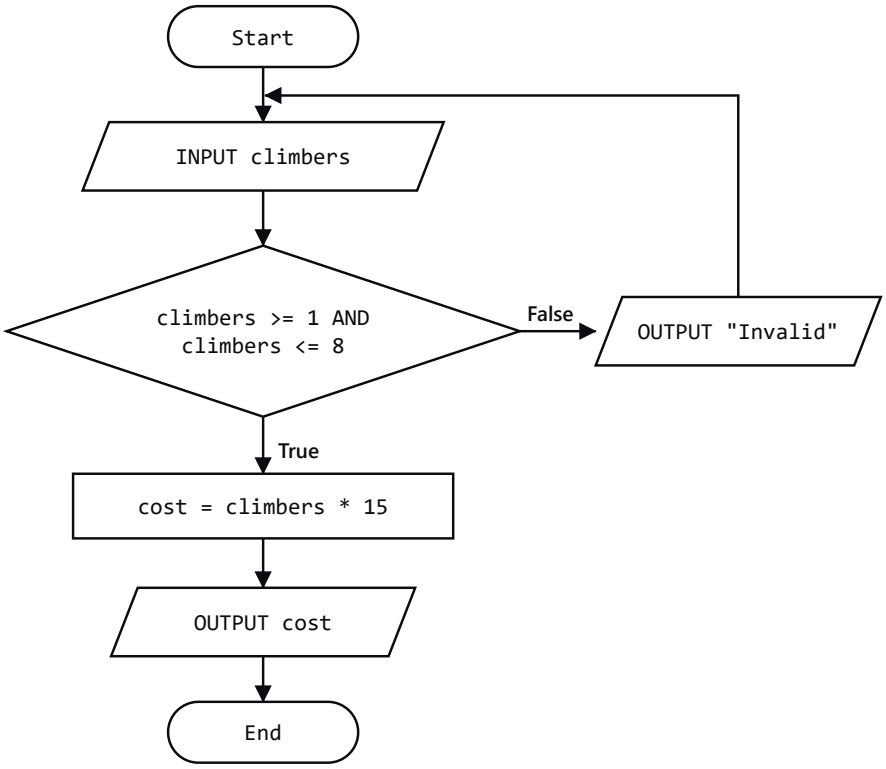
21 marks · OCR J277 spec 2.1.2: create, interpret, correct, complete and refine algorithms using flowcharts; flowchart symbols (Line, Input/Output, Process, Decision, Sub program, Terminal); trace tables

How to use. 1 mark per bullet up to the maximum shown. **/** separates alternative answers. Words in (brackets) are not needed for the mark. Credit any other valid answer that shows the same understanding. Flowchart answers are marked for logic, not exact syntax: lines without arrowheads are accepted, and minor syntax slips (missing quotes, = for == in a decision) are not penalised.

<table>
<tr><th>QUESTION</th><th>ANSWER</th><th>MARKS</th><th>GUIDANCE</th></tr>
<tr><td>1(a)
J277 2.1.2</td><td>A: Decision
B: Sub program
C: Terminal
D: Input/Output</td><td>4</td><td>1 mark per correct row. Accept subroutine / procedure / function / predefined process for B. Accept terminator / Start/End for C. Accept input or output alone for D. Do not accept process for B or D.</td></tr>
<tr><td>2(a)
J277 2.1.2</td><td>height: 100, 50, 25, 12.5, 6.25
bounces: 0, 1, 2, 3, 4
Output: 4 (once, at the end)
<table><tr><th>HEIGHT</th><th>BOUNCES</th><th>OUTPUT</th></tr><tr><td>100</td><td>0</td><td></td></tr><tr><td>50</td><td>1</td><td></td></tr><tr><td>25</td><td>2</td><td></td></tr><tr><td>12.5</td><td>3</td><td></td></tr><tr><td>6.25</td><td>4</td><td></td></tr><tr><td></td><td></td><td>4</td></tr></table></td><td>3</td><td>1 mark per correct column (all values in order). Values may be on different rows or repeated on later rows. Accept 12½ and 6¼. Do not award the output mark if any other value appears in the Output column.</td></tr>
<tr><td>2(b)
J277 2.1.2</td><td>0</td><td>1</td><td>The decision is False on the first check, so the loop is never entered.</td></tr>
</table>

QUESTION	ANSWER	MARKS	GUIDANCE
3(a) J277 2.1.2	A: `INPUT temperature` / `temperature = input()` B: `temperature < 27` C: `heater = "On"` D: `heater = "Off"`	**4**	1 mark per box. Any sensible variable name, used consistently. Do not accept `temperature <= 26` (the temperature may not be a whole number). Allow a reversed condition (e.g. `temperature >= 27`) only if C and D are swapped to match. B: do not accept a condition in English with no comparison (e.g. "is it cold?"). Do not accept OUTPUT in C or D.
4(a) J277 2.1.2	**Error 1:** `INPUT battery` is in a process symbol / rectangle (1) **Correction:** use the input/output symbol / parallelogram (1) **Error 2:** the condition `battery < 20` is wrong / a battery level of 20 outputs "Ready" (1) **Correction:** change the condition to `battery <= 20` / `battery < 21` (1)	**4**	Errors may be given in either order. For the second correction also accept `battery > 20` with the True and False branches (or the two outputs) swapped. A correction mark may be given without the matching error mark only if the correction makes the error clear. Do not accept "wrong shape" alone for error 1.
5(a) J277 2.1.2 (input validation 2.3.1)	1 mark each to max 5: Start and End terminal symbols used input of the number of climbers in an input/output symbol decision (diamond) that checks **both** limits: `climbers >= 1 AND climbers <= 8` / two decisions / equivalent ...if invalid, `"Invalid"` is output and a line goes back to the input (so the number is input again) cost calculated as `climbers * 15` in a process symbol cost output in an input/output symbol, on the valid path, before End decision branches labelled True/False or Yes/No the right way round *An example answer is drawn below the table.*	**5**	Allow any logically valid flowchart. Allow the checks in English, e.g. "is climbers between 1 and 8?", as this is a design task. Do not award the decision mark if only one limit is checked or the limits are wrong (e.g. > 1). A process box used for input or output loses that mark only. Lines without arrowheads are accepted. Ignore the £ sign in the output.

Example answer for 5(a)



Total: 21 marks

MODULE K ALGORITHM DESIGN · K7

Turning a problem into an algorithm: mark scheme

21 marks · OCR J277 spec 2.1.2: identify the inputs, processes and outputs for a problem; create, complete and refine algorithms using pseudocode, flowcharts, Exam Reference Language or a high-level language (sequence, selection, iteration)

How to use. 1 mark per bullet up to the maximum shown. **/** separates alternative answers. Words in (brackets) are not needed for the mark. Allow any logically valid method in the algorithm questions. Minor syntax slips are not penalised; a repeated slip (e.g. spaces in an identifier, == used for assignment) is penalised once, then follow through. Do not accept maths symbols such as ×, ≥ or ≤ in code answers. Example answers are one possible response, not the only one.

QUESTION	ANSWER	MARKS	GUIDANCE
1(a) J277 2.1.2	**Input:** the number of pages printed (today) **Process:** any one from: check whether more than 30 pages were printed / subtract 30 from the number of pages / multiply the extra pages by 15 / work out the charge **Output:** the charge (in pence)	**3**	1 mark per row. Credit any other valid process. Do not accept "the charge" as an input, or "the number of pages" as an output. Do not accept "printing" / "the printer prints the pages" as the process.
1(b) J277 2.1.2	Multiply the extra pages by 15: **Sequence** Charge nothing if 30 pages or fewer: **Selection** Keep asking until 1 to 500 is entered: **Iteration**	**3**	1 mark per correct row. A row with more than one tick scores 0.

2 J277 2.1.2	1 mark each:
	Input (and store) the number of pages
	Attempt at selection that checks the number of pages
	... with the correct boundary: more than 30 (e.g. `pages > 30` / `pages >= 31` / `pages >= 30` / `pages <= 30` with the branches the other way round)
	Charge worked out as (pages − 30) multiplied by 15 when more than 30 pages are printed
	Charge is 0 when 30 or fewer pages are printed, **and** the charge is output in both cases
	Example answer (ERL):
	`pages = int(input("Pages printed today"))`
	`charge = 0`
	`if pages > 30 then`
	`  charge = (pages - 30) * 15`
	`endif`
	`print(charge)`
		5	Max 5. Accept a flowchart, pseudocode, ERL or any high-level language. In a flowchart, the decision must be a diamond with both exits labelled (Yes/No or True/False). Accept `pages * 15 - 450` for the calculation. Allow charge set to 0 at the start instead of in an else branch. Do not give the boundary mark for `pages > 31` / `pages >= 32`. Allow `pages >= 30` / `pages > 29`, because the calculation gives a charge of 0 at 30 pages. Do not accept a calculation written in English ("times the extra pages by the price") without the values.

QUESTION	ANSWER	MARKS	GUIDANCE

3(a)
J277 2.1.2
	1 mark each to max 3:
	Input of the number of pages
	A loop that repeats while the value is invalid / until the value is valid
	… with a correct condition that checks both limits (e.g. `pages < 1 OR pages > 500` to repeat / `pages >= 1 AND pages <= 500` to stop)
	The number of pages is input again inside the loop
	Example answer (ERL):
	`pages = int(input("Pages printed today"))`
	`while pages < 1 OR pages > 500`
	`  pages = int(input("Enter a number from 1 to 500"))`
	`endwhile`
	or
	`do`
	`  pages = int(input("Pages printed today"))`
	`until pages >= 1 AND pages <= 500`
		3	Max 3. A single `if` with no loop (asks again only once): max 1. Do not credit a condition with a missing variable (e.g. `pages < 1 OR > 500`) or AND/OR the wrong way round (e.g. `while pages < 1 AND pages > 500`). Accept `pages <= 0 OR pages >= 501`. In a flowchart, accept a decision with an arrow back to the input.

4(a) J277 2.1.2	Total: a meaningful identifier, e.g. `totalPassengers` / `total`, **and** starting value 0 Count: a meaningful identifier, e.g. `busyCrossings` / `count`, **and** starting value 0	**2**	1 mark per row; both the identifier and the starting value are needed. Do not accept identifiers with spaces, single letters (e.g. x), or reserved words such as `input` / `print`. Accept underscores (`total_passengers`).

QUESTION	ANSWER	MARKS	GUIDANCE

4(b)
J277 2.1.2
	1 mark each to max 5:
	A loop that runs 8 times
	Input of the number of passengers inside the loop
	Number of passengers added to a running total inside the loop
	Selection checking more than 150 passengers (`> 150` / `>= 151`)
	… count increased by 1 only when the condition is true
	Total **and** count both output after (outside) the loop
	Example answer (ERL):
	`totalPassengers = 0`
	`busyCrossings = 0`
	`for crossing = 1 to 8`
	`  passengers = int(input("Passengers on this crossing"))`
	`  totalPassengers = totalPassengers + passengers`
	`  if passengers > 150 then`
	`  busyCrossings = busyCrossings + 1`
	`  endif`
	`next crossing`
	`print("Total passengers: " + str(totalPassengers))`
	`print("Crossings over 150: " + str(busyCrossings))`
		5	Max 5. ERL or a high-level language only: a flowchart, structured English or bullet points score 0. Allow `for crossing = 0 to 7` and Python `range(8)`. Uninitialised total/count can be assumed to start at 0. Do not give the condition mark for `>= 150`. Do not accept `busyCrossings =+ 1`. Total or count reset inside the loop: do not give the running-total / count mark.

Total: 21 marks

MODULE K ALGORITHM DESIGN · K8

Trace tables and dry runs: mark scheme

21 marks · OCR J277 spec 2.1.2: trace tables; create and use trace tables to follow an algorithm; identify syntax/logic errors in code and suggest fixes

How to use. 1 mark per bullet up to the maximum shown. **/** separates alternative answers. Words in (brackets) are not needed for the mark. **Trace tables:** 1 mark for each correct sequence of values in a column, in the right order. A value may be repeated on later rows (do not penalise). Rows may be laid out differently from the example, as long as the order of changes in each column is right. Outputs are written without `print` and without quotation marks; a blank cell means no output, but a cell holding a mark such as "-" or "x" in the Output column may be read as an output. Credit any other valid answer that shows the same understanding.

<table>
<tr><th>QUESTION</th><th>ANSWER</th><th>MARKS</th><th>GUIDANCE</th></tr>
<tr><td>1(a)
J277 2.1.2</td><td>To follow an algorithm (step by step) / to record the values of the variables (and outputs) as the algorithm runs
To check an algorithm works / gives the expected output / to find (logic) errors in an algorithm
To work out what an algorithm does / its purpose</td><td>1</td><td>Any 1 bullet. Accept "to test / check the algorithm (by hand)". Do not accept "to store data" / "to make the program run faster".</td></tr>
<tr><td>1(b)
J277 2.1.2</td><td>Logic (error)</td><td>1</td><td>Do not accept syntax error / runtime error.</td></tr>
<tr><td>2(a)
J277 2.1.2</td><td>height: 0, 30, 75, 95, 110
boxes: 0, 1, 2, 3, 4
boxSize: 30, 45, 20, 15 (and no 40)
Output: 4, once, in the last row (after all the other values)
<table><tr><th>height</th><th>boxes</th><th>boxSize</th><th>Output</th></tr><tr><td>0</td><td>0</td><td></td><td></td></tr><tr><td>30</td><td>1</td><td>30</td><td></td></tr><tr><td>75</td><td>2</td><td>45</td><td></td></tr><tr><td>95</td><td>3</td><td>20</td><td></td></tr><tr><td>110</td><td>4</td><td>15</td><td></td></tr><tr><td></td><td></td><td></td><td>4</td></tr></table></td><td>4</td><td>1 mark per correct column. The loop ends because 110 is not less than 100, so the fifth input (40) is never read: boxSize containing 40 = 0 for the boxSize column, and height or boxes carried on past 110 / 4 = 0 for that column. Output given as anything other than 4 (e.g. 110) = 0 for Output.</td></tr>
</table>

<table>
<tr><th>QUESTION</th><th>ANSWER</th><th>MARKS</th><th>GUIDANCE</th></tr>
<tr><td>3(a)
J277 2.1.2</td><td>i: 1, 2, 3, 4
highest: 12, 17, 21 (only)
day: 0, 1, 3 (only)
Output: Day 3: 21
<table><tr><th>i</th><th>highest</th><th>day</th><th>Output</th></tr><tr><td></td><td>12</td><td>0</td><td></td></tr><tr><td>1</td><td>17</td><td>1</td><td></td></tr><tr><td>2</td><td></td><td></td><td></td></tr><tr><td>3</td><td>21</td><td>3</td><td></td></tr><tr><td>4</td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td>Day 3: 21</td></tr></table></td><td>4</td><td>1 mark per correct column. highest and day must not change when i is 2 (9) or 4 (15); repeating the current value (17 / 1 at i = 2; 21 / 3 at i = 4) is fine. A 9 or 15 in highest = 0 for that column. Output: allow small spacing/punctuation slips (e.g. Day 3 21); do not accept 21 alone or Day 4: 21.</td></tr>
<tr><td>3(b)
J277 2.1.2</td><td>Finds the highest / largest / maximum temperature (in the array)
Finds / outputs the position / index / day number of the highest temperature
Outputs the highest temperature (with its position)</td><td>2</td><td>1 mark per bullet, max 2. A line-by-line description with no overall result (e.g. “it loops from 1 to 4 and compares”) = max 1. Do not accept “sorts the temperatures” / “finds the hottest day of the year”.</td></tr>
<tr><td>4(a)
J277 2.1.2</td><td>count: 1, 2, 3, 4 and reading: 8, 6, 10, 4
total: 0, 8, 6, 10, 4
average: 1 / 1.0 and Output: 1 / 1.0 (after the loop)
<table><tr><th>count</th><th>reading</th><th>total</th><th>average</th><th>Output</th></tr><tr><td></td><td></td><td>0</td><td></td><td></td></tr><tr><td>1</td><td>8</td><td>8</td><td></td><td></td></tr><tr><td>2</td><td>6</td><td>6</td><td></td><td></td></tr><tr><td>3</td><td>10</td><td>10</td><td></td><td></td></tr><tr><td>4</td><td>4</td><td>4</td><td></td><td></td></tr><tr><td></td><td></td><td></td><td>1.0</td><td>1.0</td></tr></table></td><td>3</td><td>1 mark per bullet. The trace must follow the code as written: a total column of 0, 8, 14, 24, 28 (the corrected algorithm) = 0 for total, and an average / Output of 7 = 0 for the third mark. FT: average and Output correct for the candidate’s final total ÷ 4.</td></tr>
<tr><td>4(b)
J277 2.1.2</td><td>Line 04
<code>total = total + reading</code></td><td>2</td><td>1 mark for the line number, 1 for the refined line. Accept <code>total += reading</code> / <code>total = reading + total</code>. Accept the refined line with no line number given. Do not accept an English description alone (“add reading to total”). Do not accept <code>total =+ reading</code>.</td></tr>
</table>

QUESTION	ANSWER	MARKS	GUIDANCE
5(a) J277 2.1.2	Column headings: one for each variable (n, steps) and one for Output n: 13, 6, 3, 1 steps: 0, 1, 2, 3 Output: 3, once, after the other values <table><tr><th>n</th><th>steps</th><th>Output</th></tr><tr><td>13</td><td>0</td><td></td></tr><tr><td>6</td><td>1</td><td></td></tr><tr><td>3</td><td>2</td><td></td></tr><tr><td>1</td><td>3</td><td></td></tr><tr><td></td><td></td><td>3</td></tr></table>	**4**	1 mark per bullet. Allow an extra Line number column (not marked) and the fourth column left blank. Headings: variable names must match the code; allow "print" for Output. n uses integer division (13 DIV 2 = 6): 6.5 / 3.25 / 1.625 = 0 for n. Do not award the n or steps mark if the column carries on after n reaches 1. Award the n, steps and Output marks for correct values even if headings are missing or wrong, provided it is clear which variable each column traces.

Total: 21 marks

MODULE K ALGORITHM DESIGN · K9

Finding and fixing errors in an algorithm: mark scheme

21 marks · OCR J277 specification 2.1.2 (identify common errors; identify syntax/logic errors in code and suggest fixes; trace tables; correct and refine algorithms in pseudocode, flowcharts and reference language) and 2.3.2 (syntax and logic errors)

How to use. 1 mark per bullet up to the maximum shown. **/** separates alternative words or answers. Words in (brackets) are not needed for the mark. Corrected code may be in OCR Exam Reference Language or any high-level language: mark the logic, and do not penalise minor syntax slips (e.g. missing `then`, `endif`, a colon or a bracket). Credit any other valid answer that shows the same understanding.

QUESTION	ANSWER	MARKS	GUIDANCE
1(a) J277 2.1.2, 2.3.2	1 mark per correct row: `print("Goodbye)`: Syntax error `for i = 1 to 9`: Logic error `iff code == "4821" then`: Syntax error	**3**	A row with both boxes ticked scores 0. Row 2 is a logic error: the line is valid code but the loop runs 9 times, not 10.
2(a) J277 2.1.2	Line 03 (identified) … `for i = 0 to 5` / `for i = 0 to temps.length - 1` Line 04 (identified) … `if temps[i] >= 30 then`	**4**	The correction mark for each line may be given if the line number is missing but the rewritten line makes clear which line is meant. Line 03: allow `for i in range(6)` / `range(0, 6)`. Do not allow `0 to 6` or `1 to 5`. Line 04: allow `> 29` (the readings are whole numbers) / `30 <= temps[i]`. Do not allow `>=` written as ≥ or `=>`.

<table>
<tr><th>QUESTION</th><th>ANSWER</th><th>MARKS</th><th>GUIDANCE</th></tr>
<tr><td>3(a)
J277 2.1.2</td><td>bid: 1, 2, 3, 4
amount: 25, 40, 15, 30 (in this order)
highest is 0 and never changes ... AND Output is 0 (once, at the end)
<table>
<tr><th>BID</th><th>AMOUNT</th><th>HIGHEST</th><th>OUTPUT</th></tr>
<tr><td></td><td></td><td>0</td><td></td></tr>
<tr><td>1</td><td>25</td><td></td><td></td></tr>
<tr><td>2</td><td>40</td><td></td><td></td></tr>
<tr><td>3</td><td>15</td><td></td><td></td></tr>
<tr><td>4</td><td>30</td><td></td><td>0</td></tr>
</table></td><td>3</td><td>Values may be repeated on later rows (e.g. 0 written in the highest column on every row). Allow the output on its own final row. Do not give the third mark if any other value (25, 40, 15 or 30) appears in the highest column.</td></tr>
<tr><td>3(b)
J277 2.1.2</td><td>40</td><td>1</td><td></td></tr>
<tr><td>3(c)
J277 2.1.2</td><td>Line 04 (identified)
... <code>if amount > highest then</code></td><td>2</td><td>Allow >= / <code>highest < amount</code>.
Do not credit changes to line 01 or line 05 (they do not fix the error).</td></tr>
<tr><td>4(a)
J277 2.1.2</td><td>Max 2 from:
count is never changed / stays at 0 ...
... so <code>count < 5</code> is always true
The loop never ends / it is an infinite loop
It keeps asking for parcel weights (more than five times)
The total weight is never output</td><td>2</td><td>Do not allow “it crashes” or “it doesn't work” on their own.</td></tr>
<tr><td>4(b)
J277 2.1.2</td><td>Error 1
count is never increased / updated inside the loop (lines 03 to 07)
... add <code>count = count + 1</code> inside the loop (e.g. after line 06 / before line 07)
Error 2
Line 04 sets / resets <code>total</code> to 0 every time the loop runs (so only the last weight is kept)
... delete / remove line 04 / move it before the loop (it is already done on line 01)</td><td>4</td><td>For the Error 1 fix, allow replacing the while loop with a count-controlled loop that runs 5 times (e.g. <code>for count = 1 to 5</code> ... <code>next count</code>).
Allow <code>count += 1</code>. Do not allow <code>count =+ 1</code>.
The fix for Error 1 must be inside the loop.</td></tr>
<tr><td>5(a)
J277 2.1.2</td><td>Heating off</td><td>1</td><td>Ignore quotation marks.</td></tr>
</table>

QUESTION	ANSWER	MARKS	GUIDANCE
5(b) J277 2.1.2	Change the decision to `temp < 18` / change > to < in the decision box	**1**	Allow `18 > temp`. Do not allow `temp <= 18`, or swapping the True and False labels / outputs on their own: the heating would then come on at exactly 18 °C, which is not below 18.

Total: 21 marks

MODULE K ALGORITHM DESIGN · K10

Explaining what an algorithm does: mark scheme

21 marks · OCR J277 spec 2.1.2 (interpret algorithms; trace tables), 2.1.3 (identify a standard algorithm from code), 2.3.1 (maintainability: naming)

How to use. 1 mark per bullet up to the maximum shown. **/** separates alternatives. Words in (brackets) are not needed for the mark. **...** means the bullet depends on the one before it. Credit any other valid answer that describes the algorithm accurately; do not credit vague answers that would fit any algorithm ("it processes the data").

QUESTION	ANSWER	MARKS	GUIDANCE
1(a) J277 2.1.2, 2.3.1	1 mark per row: A: 4 (running total) B: 1 (largest value) C: 3 (re-enter until in range) D: 2 (adds one when a condition is met)	**4**	Description 5 is not used. A row with more than one number scores 0.
2(a) J277 2.1.2	1 mark each: y: 1, 2, 3 x: -1, 5, -4 z: 3, -1, -4 Output: -4 (once, after the loop ends) **Example:** `01: z 3 · 02: y 1 · 03: x -1 · 05: z -1 · 02: y 2 · 03: x 5 · 02: y 3 · 03: x -4 · 05: z -4 · 08: output -4`	**4**	Mark each variable's changes in order; line numbers are not marked. Values may be repeated on later rows. Each variable's values must be in the correct order. Ignore an extra final value of 4 for y. Output mark: -4 only, with no other output in the column (a blank cell is not an output).
2(b) J277 2.1.2	Outputs the lowest / smallest / minimum / coldest (of the four) temperature(s) entered	**1**	Do not accept "outputs a temperature", "outputs the last temperature" or "sorts the temperatures". Do not accept "outputs -4" alone (that is the result for one set of data, not the purpose).

QUESTION	ANSWER	MARKS	GUIDANCE
3(a) J277 2.1.2	1 mark each to max 4: Repeats / loops 20 times // once for each of 20 parcels Inputs the weight of each parcel Asks for the weight again until it is more than 0 and 30 (kg) or less // validates the weight (is in range 0 to 30) Checks whether the weight is more than / over 25 (kg) ... and if so adds 1 to heavy // counts the parcels over 25 kg in heavy heavy starts at 0 Outputs the number of heavy parcels (with the message "Heavy parcels: ") (after all 20 are entered)	**4**	Max 3 if no reference to the actual values (20, 0, 30, 25) is made. Do not accept "25 or more" / "at least 25" for the heavy check (25 is not counted). Accept "between 0 and 30" for the validation bullet. Do not accept "checks the weight" alone.
4(a) J277 2.1.3	Bubble sort	**1**	Do not accept "sort" alone, insertion sort or merge sort.
4(b) J277 2.1.3	1 mark from: Records / shows whether a swap was made (during a pass) So the loop repeats until a pass makes no swaps // so the algorithm stops when the prices are in order / sorted	**1**	Do not accept "swaps the prices" (that is done by lines 05 to 07 using `temp`).
4(c) J277 2.1.3	Ascending (order) / smallest (price) first / lowest to highest	**1**	Do not accept "sorted" or "in order" alone, or "numerical order" alone.
5(a) J277 2.1.2	12	**1**	Accept 12.0. (12 + 15 + 9 = 36; 36 / 3 = 12.)
5(b) J277 2.1.2	1 mark each to max 3: Calculates / returns the mean / average water level (of the readings in the array) Ignores / leaves out readings of -1 // only uses readings that were taken (successfully) `t` holds the total of the (valid) readings `n` counts the number of (valid) readings Returns `t` divided by `n` Returns 0 if there are no valid readings // stops a division by zero	**3**	Max 2 if the average / mean is not identified. Do not accept "checks the readings" or "adds up all the readings" (the -1 values are not added).
5(c) J277 2.3.1	Any name that describes the purpose, e.g. `averageLevel` / `meanLevel` / `meanValidReadings` / `average_level`	**1**	Must suggest an average / mean. Do not accept `check`, a single letter, a name with spaces, or a name that does not describe the result (e.g. `levels`, `water`).

Total: 21 marks