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