MODULE O ROBUST PROGRAMS · O2

Input validation: mark scheme

21 marks · Cambridge 0478 syllabus 7.5(a) (need for validation checks; range, length, type, presence, format, check digit), 7.7 (trace tables), 7.8 (identify and correct errors), 7.9 (write algorithms), 8.1.4(c) (iteration), 8.1.4(e) (string handling: LENGTH, SUBSTRING)

How to use. One mark per creditworthy point up to the maximum shown. **/** separates alternative words; **//** separates alternative answers. Words in (brackets) are not needed for the mark. Algorithm answers must be in pseudocode; marks are for the features listed, not for one exact algorithm. Ignore minor slips in case and spacing. Accept `<-` or `=` for assignment. Do not accept English ("is more than") or maths symbols (≥, ≤, ≠) in place of operators. Credit any other valid answer that shows the same understanding.

QUESTION	ANSWER	MARKS	GUIDANCE
1(a) 0478 7.5(a)	Any two from: • to check that the data is reasonable / sensible / follows set rules (before it is accepted) • to stop invalid / unreasonable data being accepted // stored // processed • invalid data could cause incorrect results / outputs // could cause the program to crash / fail • so the user can be told the data is invalid and asked to re-enter it • the checks are carried out automatically (by the computer)	**2**	Max 2. Do not accept "to check the data is correct / accurate" or "to make sure there are no errors" (validation cannot show data is correct). Do not accept answers that describe verification (checking data matches the original / was copied accurately). Credit any other valid answer.
2(a) 0478 7.5(a), 8.1.4(e)	One mark for each correct row: • `Height` statement: range check • `LENGTH(TownName)` statement: length check • `Surname = ""` statement: presence check • `SUBSTRING(BatchCode, 3, 1)` statement: format check	**4**	Row 4: the statement tests that the third character is a /, i.e. part of a set pattern. Do not accept presence check or type check for row 4. Do not accept "limit check" for row 1 (use the syllabus name). Spelling of the check name need not be exact if the meaning is clear.

<table>
<tr><th>QUESTION</th><th>ANSWER</th><th>MARKS</th><th>GUIDANCE</th></tr>
<tr><td>3(a)
0478 7.7, 7.5(a), 8.1.4(c)</td><td>One mark for each:
• <code>Passengers</code> column: 0, 12, 9 (and nothing after 9)
• <code>Error: enter a number from 1 to 9</code> output twice only: once for 0 and once for 12
• <code>Booked for 9 passengers</code> output once, as the last output
<table><tr><th>PASSENGERS</th><th>OUTPUT</th></tr><tr><td>0</td><td>Error: enter a number from 1 to 9</td></tr><tr><td>12</td><td>Error: enter a number from 1 to 9</td></tr><tr><td>9</td><td></td></tr><tr><td></td><td>Booked for 9 passengers</td></tr></table></td><td>3</td><td>The value 4 is never input (9 is valid, so the UNTIL condition is TRUE and the loop ends): a <code>Passengers</code> column that includes 4 does not get the first mark, and any output for 4 loses the mark for that output. The final output may be on the same row as 9. An error message beside 9 loses the second mark. Ignore quotation marks round outputs. Accept <code>Booked for 9 passengers</code> with minor spacing differences.</td></tr>
<tr><td>3(b)
0478 7.5(a)</td><td>• type check</td><td>1</td><td>Do not accept range check, presence check or format check.</td></tr>
<tr><td>4(a)
0478 7.8, 7.5(a), 8.1.4(e)</td><td>One mark for each line identified and corrected:
• Line 04: <code>AND</code> should be <code>OR</code> // <code>IF (LENGTH(Surname) < 2) OR (LENGTH(Surname) > 30)</code>
• Line 08: <code>> 2</code> should be <code>>= 2</code> // <code>UNTIL (LENGTH(Surname) >= 2) AND (LENGTH(Surname) <= 30)</code></td><td>2</td><td>Line number alone, or correction alone: 0 for that error. Accept any logically equivalent correction, e.g. line 08 <code>LENGTH(Surname) > 1</code>; line 04 <code>NOT ((LENGTH(Surname) >= 2) AND (LENGTH(Surname) <= 30))</code>. Each side of a compound condition must be a full comparison. Why they are errors: with <code>AND</code> on line 04 no length can be both under 2 and over 30, so the error message is never output; with <code>> 2</code> on line 08 a valid 2-character surname is rejected.</td></tr>
<tr><td>4(b)
0478 7.5(a), 8.1.4(c)</td><td>Any one from:
• the loop makes the user re-enter the surname as many times as needed // until it is valid; an IF statement would only check (and ask again) once
• with only an IF, a second invalid surname would not be checked / would be accepted</td><td>1</td><td>“To repeat it” alone = 0 (must say what is repeated, or until when). Credit any other valid answer.</td></tr>
<tr><td>5(a)
0478 7.5(a) (check digit; see also 2.2.3)</td><td>• check digit (check)</td><td>1</td><td>Do not accept parity check, checksum or format check.</td></tr>
</table>

QUESTION	ANSWER	MARKS	GUIDANCE
5(b) 0478 7.5(a)	Any one from: • the code entered still has 7 digits / characters // the correct number of characters • a length check only checks the number of characters, not what the digits are / their order	**1**	"It is the right length" is enough. "A length check cannot detect errors" alone = 0.

6(a)
0478 7.9, 7.5(a), 8.1.4(c), 8.1.4(e)
	One mark per bullet point. Max 6:
	• prompt (OUTPUT) and INPUT of the asset tag
	• condition-controlled loop (REPEAT ... UNTIL // WHILE ... DO ... ENDWHILE) with the input inside the loop, so the tag is re-entered
	• length check: `LENGTH(Tag) = 7` (valid) // `LENGTH(Tag) <> 7` (invalid)
	• format check on the first character: `SUBSTRING(Tag, 1, 1) = "H"` (valid) // `<> "H"` (invalid)
	• the two checks combined correctly in the loop condition: AND for valid (e.g. UNTIL) // OR for invalid (e.g. WHILE)
	• error message output only when the tag is invalid, and `Tag accepted` output once, after the loop
	Example:
	`REPEAT`
	`  OUTPUT "Enter the asset tag"`
	`  INPUT Tag`
	`  IF (LENGTH(Tag) <> 7) OR (SUBSTRING(Tag, 1, 1) <> "H")`
	`  THEN`
	`  OUTPUT "Invalid tag: 7 characters, starting H"`
	`  ENDIF`
	`UNTIL (LENGTH(Tag) = 7) AND (SUBSTRING(Tag, 1, 1) = "H")`
	`OUTPUT "Tag accepted"`
		6	Accept `'H'` for `"H"`. Accept any meaningful identifier for the tag. Accept separate IF statements with separate error messages, or a Boolean flag (e.g. `Valid`) set by the checks and used in the loop condition. Each side of a compound condition must be a full comparison. AND joining the two invalid tests // OR joining the two valid tests loses the fifth mark. A single IF with no loop (only one re-entry): max 4 (prompt and input, length check, format check, messages). Credit a correct use of `UCASE` to also allow a lower-case h, but it is not required. Program code instead of pseudocode: 0.

Total: 21 marks