MODULE K ALGORITHM DESIGN · K9

Finding and fixing errors in an algorithm: mark scheme

21 marks · Cambridge 0478 syllabus 7.8 (identify errors in given algorithms and suggest ways of correcting these errors) and 7.7 (complete a trace table to document a dry-run of an algorithm)

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. Corrections should be in pseudocode; ignore minor slips (e.g. <− or = for ←, missing `THEN`) as long as the logic is clear. 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)
0478 7.8</td><td>One mark for each error identified and corrected:
Line 06 // <code>HighCount ← 1</code> should be <code>HighCount ← 0</code>
Line 07 // <code>FOR Month ← 1 TO 11</code> should be <code>FOR Month ← 1 TO 12</code>
Line 11 // <code>IF Reading > 400.0</code> should be <code>IF Reading >= 400.0</code>
Line 17 // <code>Total / HighCount</code> should be <code>Total / 12</code></td><td>4</td><td>max 4. The line number alone, or the correction alone with no way of telling which line it replaces, scores 0.
Line 07: accept <code>0 TO 11</code> // <code>2 TO 13</code> (loop runs 12 times).
Line 11: accept <code>>= 400</code> // <code>400.0 <= Reading</code>. Do not accept <code>> 399</code> or <code>> 399.9</code> (readings are REAL).
Line 17: accept <code>Total / 12.0</code>. Do not accept <code>Total / Month</code>.</td></tr>
<tr><td>2(a)
0478 7.7, 7.8</td><td>One mark per correct column:
Runner: 1, 2, 3, 4, 5
Time: 62, 58, 71, 55, 60
Fastest: 0 (only, entered once at the start)
OUTPUT: Fastest time: 0
<table>
<tr><th>RUNNER</th><th>TIME</th><th>FASTEST</th><th>OUTPUT</th></tr>
<tr><td></td><td></td><td>0</td><td></td></tr>
<tr><td>1</td><td>62</td><td></td><td></td></tr>
<tr><td>2</td><td>58</td><td></td><td></td></tr>
<tr><td>3</td><td>71</td><td></td><td></td></tr>
<tr><td>4</td><td>55</td><td></td><td></td></tr>
<tr><td>5</td><td>60</td><td></td><td>Fastest time: 0</td></tr>
</table></td><td>4</td><td>Allow Fastest 0 repeated on every row. Do not credit the Fastest column if any other value appears in it.
OUTPUT: allow 0 alone. Ignore quotation marks. Do not penalise a final 6 in the Runner column.</td></tr>
</table>

QUESTION	ANSWER	MARKS	GUIDANCE
2(b)(i) 0478 7.8	Any one from: Fastest starts at 0 and no time is less than 0 // the condition `Time < Fastest` is never TRUE So Fastest is never changed / updated (and 0 is output)	**1**	Do not accept "the IF statement is wrong" on its own.
2(b)(ii) 0478 7.8	`Fastest ← 999` // Fastest set to any value larger than any possible time	**1**	Accept any value clearly larger than any possible time, e.g. 999, 9999. Do not accept 0, a negative value, or a value that a real 10 km time could exceed (e.g. 60, 71): the correction must work for any input data, not just this data.
2(b)(iii) 0478 7.7	Fastest time: 55	**1**	Accept 55 alone.
3(a) 0478 7.8	Any one from: The algorithm asks for the number of seats again // the loop repeats 5 is rejected even though it is a valid value	**1**	Do not accept "5 is accepted" or "the loop ends".
3(b) 0478 7.8	Both comparisons correct, including the boundaries: `Seats >= 1` and `Seats <= 8` joined with `AND` in an `UNTIL` statement **Example:** `UNTIL Seats >= 1 AND Seats <= 8`	**2**	Accept `Seats > 0` // `Seats < 9` (Seats is an INTEGER). Accept `UNTIL NOT (Seats < 1 OR Seats > 8)` for 2 marks. Do not accept `UNTIL Seats >= 1 AND <= 8` for the second mark (each side must be a full comparison).
4(a) 0478 7.8	Any two from: Count is never changed / incremented (inside the loop) Count stays at 0 so `Count < 10` is always TRUE The loop never stops // it is an infinite loop (so line 07 is never reached)	**2**	max 2. Do not accept "the loop is wrong" on its own.
4(b) 0478 7.8	`Count ← Count + 1` Placed inside the loop // between line 03 and line 06 (e.g. after line 05, before ENDWHILE)	**2**	Accept = for ←. Do not award the second mark if the statement is placed before line 03 or after line 06.

QUESTION	ANSWER	MARKS	GUIDANCE
5(a) 0478 7.8	Any two from: The No flow line goes back to the box that sets Count and Total to 0 // the initialisation is inside the loop Count is reset to 0 every time, so it never reaches 5 // the loop never ends // infinite loop Total is reset to 0 every time, so it would only hold the last price The total is never output	**2**	max 2.
5(b) 0478 7.8	The No flow line should go back to (just before) INPUT Price // join the No flow line to the flow line between the Count ← 0 / Total ← 0 box and INPUT Price	**1**	Accept "the loop should start at INPUT Price" // "move the box that sets Count and Total to 0 out of the loop".

Total: 21 marks