MODULE M PROGRAMMING · M11

Pre-condition loops (WHILE): mark scheme

21 marks · Cambridge 0478 syllabus (exams 2026-2028) 8.1.4(c) (understand and use the concept of iteration: pre-condition loops, `WHILE ... DO ... ENDWHILE` as in syllabus section 4), with 7.7 (trace tables), 7.8 (identify and correct errors), 7.9 (write algorithms), 8.1.4(d) (counting) and 7.5(a) (validation, range check)

How to use. One mark per bullet point up to the maximum shown. **/** separates alternative words; **//** separates alternative answers. Words in (brackets) are not needed for the mark. Credit any other valid answer that shows the same understanding. Ignore the case of keywords and minor spacing; <- or = for ← is allowed. Answers must be in pseudocode where pseudocode is asked for.

QUESTION	ANSWER	MARKS	GUIDANCE
1 0478 8.1.4(c)	One mark for each correct row: Condition tested before the statements are executed: **True** Statements always executed at least once: **False** Loop terminates when the condition evaluates to FALSE: **True**	**3**	A row with more than one tick scores 0. Row 2 is false: if the condition is FALSE on the first test, the statements are not executed at all. (Always at least once describes a post-condition / REPEAT loop.)
2 0478 8.1.4(c) 0478 7.9	One mark for each correct line: Line 03: `Height < 40` // `Height <= 39` // `NOT (Height >= 40)` Line 04: `Height + 5` Line 06: `ENDWHILE`	**3**	Line 03: do not accept `Height <= 40` (a 40 cm plant is ready) or `Height > 40` / `Height >= 40` (reversed: the loop would not run). Line 04: do not accept `5` or `Height + 1`. Line 06: do not accept `NEXT`, `ENDIF`, `UNTIL` or `END`; allow `END WHILE`. (With these lines the output is "Ready to sell after 6 weeks".)

<table>
<tr><th>QUESTION</th><th>ANSWER</th><th>MARKS</th><th>GUIDANCE</th></tr>
<tr>
<td>3(a)
0478 7.7
0478 8.1.4(c)
(d)</td>
<td>One mark for each correct column:
<code>Reading</code>: 42, 65, 51, 50, −1
<code>LoudCount</code>: 0, 1, 2
<code>ReadingCount</code>: 0, 1, 2, 3, 4
OUTPUT: 2 out of 4 readings were loud
Example completed table:
<table>
<tr><th>READING</th><th>LOUDCOUNT</th><th>READINGCOUNT</th><th>OUTPUT</th></tr>
<tr><td></td><td>0</td><td>0</td><td></td></tr>
<tr><td>42</td><td></td><td>1</td><td></td></tr>
<tr><td>65</td><td>1</td><td>2</td><td></td></tr>
<tr><td>51</td><td>2</td><td>3</td><td></td></tr>
<tr><td>50</td><td></td><td>4</td><td></td></tr>
<tr><td>−1</td><td></td><td></td><td>2 out of 4 readings were loud</td></tr>
</table>
</td>
<td>4</td>
<td>Values must be in the correct order; repeated values on later rows are allowed. Rows may also be laid out with one new row for each value change.
<code>LoudCount</code> column: 50 is not above 50, so the count stays at 2; do not award if 3 appears.
<code>ReadingCount</code>: do not award if 5 appears (−1 is not counted).
OUTPUT: ignore quotation marks; the numbers and words must match. Do not award if anything is output before the loop ends.</td>
</tr>
<tr>
<td>3(b)
0478 8.1.4(c)</td>
<td>0 out of 0 readings were loud</td>
<td>1</td>
<td>Correct answer only. The condition is FALSE on the first test, so the loop is not entered and both counts stay at 0.
Do not accept "nothing" / "no output".</td>
</tr>
<tr>
<td>3(c)
0478 8.1.4(c)</td>
<td>Any two from:
The condition (on line 04) uses <code>Reading</code> // <code>Reading</code> must have a value before the condition is tested
The condition is tested before the loop is entered / before the statements in the loop are executed (so line 10 has not happened yet)
Line 03 gets the first reading // line 10 gets each next reading, ready for the next test of the condition
Without line 03, <code>Reading</code> would have no value / be undefined when the condition is first tested (causing an error)</td>
<td>2</td>
<td>Max 2.
Do not accept "to input a reading" alone, or "to start the loop" alone.
Do not accept answers saying line 10 is not needed.</td>
</tr>
</table>

QUESTION	ANSWER	MARKS	GUIDANCE
4 0478 7.8 0478 8.1.4(c)	One mark for each line identified **and** corrected: Line 03: `Boxes ← 0` Line 06: `WHILE Eggs >= 12 DO` // `WHILE Eggs > 11 DO` Line 07: `Eggs ← Eggs - 12`	**3**	The line number alone, or the correction alone, scores 0 for that error. Line 06: the loop condition must allow exactly 12 eggs to fill a box; `DO` may be omitted. Line 07 as given adds 12, so the loop would never terminate (for 13 or more eggs). Credit any other correction that makes that line work, e.g. a different valid fix that keeps the output correct for every number of eggs.

5
0478 7.9
0478 8.1.4(c)
0478 7.5(a)
	One mark per bullet point. Max 5:
	Input of the number of passengers before the loop
	`WHILE` loop with a correct condition for an invalid number and `ENDWHILE`, e.g. `WHILE Passengers < 1 OR Passengers > 9 DO`
	... output of an error message inside the loop
	... input of the number again inside the loop
	Total fare calculated as number of passengers × 12.50 (after the loop)
	Output of the total fare with a suitable message
	Example answer:
	`OUTPUT "Enter the number of passengers (1 to 9)"`
	`INPUT Passengers`
	`WHILE (Passengers < 1) OR (Passengers > 9) DO`
	`  OUTPUT "Error: a booking must be for 1 to 9 passengers"`
	`  INPUT Passengers`
	`ENDWHILE`
	`Fare ← Passengers * 12.50`
	`OUTPUT "The total fare is $", Fare`
		5	Condition: accept `Passengers <= 0 OR Passengers >= 10` // `NOT (Passengers >= 1 AND Passengers <= 9)`. Do not award the loop mark if AND is used in place of OR, if the variable is missing from the second half, or if `ENDWHILE` is missing.
			A post-condition (`REPEAT ... UNTIL`) loop or a single `IF` does not earn the loop mark; credit the other features shown.
			Calculation: accept 12.5; do not accept × or x as the operator in pseudocode. The calculation may be inside the OUTPUT statement.
			Output marks are not awarded if the variable name is inside the quotation marks. Program code instead of pseudocode: 0.

Total: 21 marks