MODULE M PROGRAMMING · M16

String handling: length, substring, case, concatenation and slicing: mark scheme

22 marks · OCR J277 spec 2.2.3 (basic string manipulation: concatenation, slicing), with 2.2.2 (casting) and 2.1.2 (trace tables, logic errors)

How to use. 1 mark per bullet up to the Max shown. **/** separates alternative answers. Words in (brackets) are not needed for the mark. Quotation marks around string values are not needed. Code answers are marked for logic, not exact syntax: accept OCR Exam Reference Language or any high-level language (e.g. `len(town)`, `town[0:3]`, `town.upper()`), and allow any logically valid method. Minor syntax slips are not penalised; a repeated slip is penalised once then follow through. Credit any other valid answer.

QUESTION	ANSWER	MARKS	GUIDANCE
1(a) J277 2.2.3	`stop.length`: 14 `stop.substring(2, 5)`: rbour `stop.right(6)`: Lights `stop.upper`: HARBOUR LIGHTS `stop.left(1) + str(stop.length)`: H14	**5**	1 mark per correct row. Case must be exact (e.g. `Rbour`, `lights` = 0). Row 1: the space is a character, so 13 = 0. Row 2: `arbou` (counting from 1) = 0. Row 4: the space between the words must be kept. Row 5: `H 14` or `H + 14` = 0; 15 (added as numbers) = 0.
2(a) J277 2.2.3 (ERL ASC)	68	**1**	Do not accept `"D"` or the binary pattern alone.
2(b) J277 2.2.3 (ERL CHR)	e (lower case)	**1**	Accept `"e"` / `'e'`. Upper case `E` = 0.

<table>
<tr><th>QUESTION</th><th>ANSWER</th><th>MARKS</th><th>GUIDANCE</th></tr>
<tr><td>3(a)
J277 2.2.3, 2.1.2</td><td>1 mark for each correct column:
<code>i</code>: 0, 1, 2, 3, 4
<code>letter</code>: K, 4, (space), 7, T
<code>clean</code>: (empty), K, K4, K47, K47T
Output: k47t (once, at the end)
<table><tr><th>I</th><th>LETTER</th><th>CLEAN</th><th>OUTPUT</th></tr><tr><td></td><td></td><td>(empty)</td><td></td></tr><tr><td>0</td><td>K</td><td>K</td><td></td></tr><tr><td>1</td><td>4</td><td>K4</td><td></td></tr><tr><td>2</td><td>(space)</td><td>K4</td><td></td></tr><tr><td>3</td><td>7</td><td>K47</td><td></td></tr><tr><td>4</td><td>T</td><td>K47T</td><td></td></tr><tr><td></td><td></td><td></td><td>k47t</td></tr></table></td><td>4</td><td>Values must be in the correct order. The initial empty value of <code>clean</code> may be left blank or shown as <code>""</code>. Repeating <code>K4</code> on the space row, or leaving that cell blank, are both fine. For <code>letter</code> at i = 2 accept a blank-looking cell, <code>" "</code>, “space” or a symbol for a space. Output <code>K47T</code> (not lower case) = 0 for the Output mark; quotation marks around the output are ignored. A space kept in <code>clean</code> (e.g. <code>K4 7T</code>) loses the <code>clean</code> mark. A value of i = 5 loses the <code>i</code> mark.</td></tr>
<tr><td>4(a)
J277 2.2.3, 2.1.2</td><td>Line 02 (identified) ...
... <code>first = code.substring(0, 1)</code> / <code>first = code.left(1)</code>
Line 03 (identified) ...
... <code>if code.length == 6 AND first == "X" then</code></td><td>4</td><td>The correction mark for each error needs the right line number. Line 02: accept <code>code[0]</code> / <code>code[0:1]</code>. Line 03: accept <code>code.length >= 6 AND code.length <= 6</code>; do not accept <code>>= 6</code> alone (accepts 7 or more characters). Accept = for == (minor slip). Do not accept changing <code>"X"</code> or the outputs.</td></tr>
<tr><td>4(b)
J277 2.2.3</td><td>1 mark each to max 2:
Convert the code / the first character to upper case (using <code>.upper</code>) ...
... before it is compared with <code>"X"</code> (on line 03) / after it is input (line 01) or when it is extracted (line 02)
... so <code>"x"</code> becomes <code>"X"</code> and the comparison is True
Example: add <code>code = code.upper</code> after line 01 / <code>if code.length == 6 AND first.upper == "X" then</code></td><td>2</td><td>Accept a description or code. “Use .upper” with no indication of what is converted or when = 0. Converting <code>"X"</code> in line 03 to upper case (it already is) = 0. An answer that compares with both <code>"X"</code> and <code>"x"</code> (no .upper) max 1.</td></tr>
</table>

QUESTION	ANSWER	MARKS	GUIDANCE

5(a)
J277 2.2.3, 2.2.1
	1 mark each to max 5:
	Inputs the town name and the route number, storing each in a variable
	Takes the first three characters of the town name: `town.left(3)` / `town.substring(0, 3)`
	Converts (the first three characters of) the town name to upper case
	Concatenates the three parts in the correct order, with `"-"` between the letters and the route number
	Outputs the route code
	Example:
	`town = input("Enter the town name")`
	`route = input("Enter the route number")`
	`start = town.left(3)`
	`routeCode = start.upper + "-" + route`
	`print(routeCode)`
		5	Accept any variable names except reserved words (e.g. `input`). Converting the whole town name to upper case before slicing is fine. If the route number is input as an integer, do not penalise a missing `str()` when it is concatenated. `substring(1, 3)` or `[1:4]` loses the slicing mark. A comma in `print` (e.g. `print(start, "-", route)`) is not concatenation: no concatenation mark, but the output mark can be given. Building the code inside `print(...)` is fine for both marks. Do not accept flowcharts, structured English or bullet points.

Total: 22 marks