01	1	**3 marks for AO2 (apply)** 1 mark if column `z` increments by 1 and starts at 0; 1 mark if column `z` has the final value 3; 1 mark if `correct` column is correct; **z** / **correct** 0 / false 1 / true 2 / 3 / / /	3

01	2	**Mark is for AO2 (apply)** false; **I.** Case	1

01	3	**Mark is for AO2 (apply)** Second row only;	1

New Line	**Tick one box**
`IF user = us[z] OR pass = ps[z] THEN`	
`IF user = us[z] AND pass = ps[z] THEN`	Tick
`IF NOT (user = us[z] AND pass = ps[z]) THEN`	

Question	Part	Marking guidance	Total marks
01	4	**Mark is for AO2 (apply)** Maximum 2 marks from: The program will return true as soon as a match (between username and password) is found; So there is no need to (always) iterate over the complete array(s)/list of usernames; (If a match is found and is not last in the list) the algorithm will complete in fewer steps/less time; **A**. the programmer has used fewer variables	2

Qu	Part	Marking guidance	Total marks
02	1	**Mark is for AO2 (apply)** **B**: Integer; **R.** If more than one lozenge shaded.	1
02	2	**1 mark for AO2 (apply)** Boolean/bool;	1

02	3	**3 marks for AO2 (apply)**
		
		1 mark for each correct value of `valid`;;;
		
		Value of `instr` / **Final value of** `valid`:
		`ADD R0, R1` – `False`
		`ADD: R0, R1` – `True`
		`HALT` – `True`
			3

02	4	**Mark is for AO1 (understanding)** Machine code; **A.** binary; **A.** object code;	1
02	5	**2 marks for AO1 (understanding)** **Max 2 marks from:** (High-level languages) are better supported; (High-level languages) provide built-in subroutines; (High-level languages) provide programming structures such as iteration and selection; (Code written in high-level languages) is normally shorter; (High-level languages) allow creation of subroutines; (High-level languages) provide data structures; (High-level languages) are easier to understand/read; (High-level languages) are easier to debug; **A.** any other correct justification.	2

Qu	Part	Marking guidance	Total marks

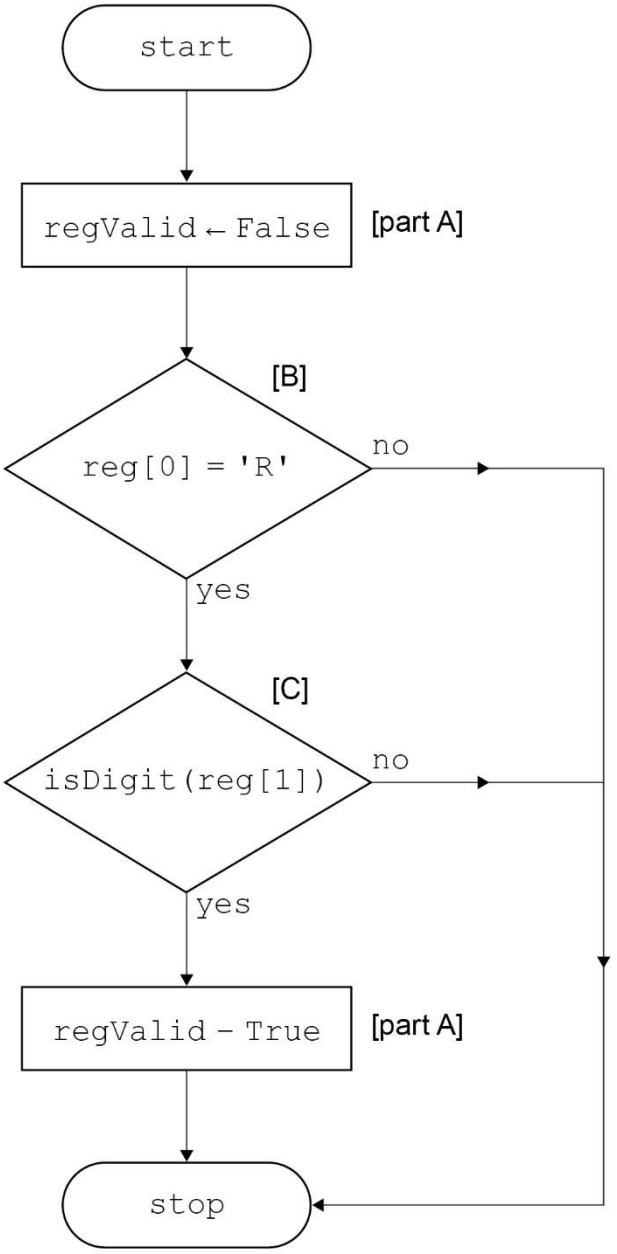
02	6	**3 marks for AO3 (program)**
		
		Mark A for setting the variable `regValid` to `True`/`False` within a selection structure;
		Mark B for using a Boolean condition that checks if the first character is an `'R'`;
		Mark C for using a Boolean condition that checks if the second character is a digit;
		
		Max 2 marks if any errors in the answer.
		
		A. minor changes to variable identifiers if the meaning is still clear.
		
		Example of fully correct answer:
		
		`regValid ← False` [part A]
		`IF reg[0] = 'R' and isDigit(reg[1]) THEN` [B,C]
		`  regValid ← True` [part A]
		`ENDIF`
		
		Example of another fully correct answer:
		
		`IF reg[0] = 'R' THEN` [B]
		`  IF isDigit(reg[1]) THEN` [C]
		`  regValid ← True` [part A]
		`  ELSE`
		`  regValid ← False` [part A]
		`  ENDIF`
		`ELSE`
		`  regValid ← False` [part A]
		`ENDIF`
		
		Example of 2 mark answer:
		
		`IF reg[0] = 'R' or isDigit(reg[1]) THEN` [B,C]
		`  regValid ← True` [part A]
		`ELSE`
		`  regValid ← True` [part A]
		`ENDIF`
		
		(only 2 marks awarded as the answer contains an error in the Boolean condition)
			3

Example of another 2 mark answer:

```
IF reg[0] = 'R' and isDigit(reg[1]) THEN [B,C]
     regValid ← True                     [part A]
ENDIF
```

(only 2 marks awarded as only part of mark A is given)

Example of a fully correct flowchart solution:



<table>
<tr><th>Question</th><th>Part</th><th>Marking guidance</th><th>Total marks</th></tr>
<tr><td>03</td><td>1</td><td>2 marks for AO2 (apply)

<table>
<tr><th>Input value of
orderTotal</th><th>Input value of
deliveryDistance</th><th>Output</th></tr>
<tr><td>55.5</td><td>2</td><td>1.5;</td></tr>
<tr><td>35.0</td><td>5</td><td>7.0;
A. 7</td></tr>
</table>
</td><td>2</td></tr>
</table>

<table>
<tr><th>Question</th><th>Part</th><th>Marking guidance</th><th>Total marks</th></tr>
<tr><td>03</td><td>2</td><td>Mark is for AO2 (apply)

2 // two;</td><td>1</td></tr>
</table>

<table>
<tr><th>Question</th><th>Part</th><th>Marking guidance</th><th>Total marks</th></tr>
<tr><td>03</td><td>3</td><td>2 marks for AO2 (apply)

<table>
<tr><th>Variable identifier</th><th>Data type</th></tr>
<tr><td>deliveryCost</td><td>Float // Real // Decimal</td></tr>
<tr><td>messageOne</td><td>String // str</td></tr>
</table>

I. Case
A. Programming language specific data types eg Single in VB.NET</td><td>2</td></tr>
</table>

<table>
<tr><th>Question</th><th>Part</th><th>Marking guidance</th><th>Total marks</th></tr>
<tr><td>03</td><td>4</td><td>Mark is for AO1 (recall)

Boolean // Bool;
Int // Integer;
Date/Time;
Character;

R. Any answer that was given in 02.3

I. Case
A. Any reasonable data type</td><td>1</td></tr>
</table>

Question	Part	Marking guidance	Total marks
04	1	**Mark is for AO2 (apply)** **B** Line number 2; **R.** If more than one lozenge shaded	1

Question	Part	Marking guidance	Total marks
04	2	**Mark is for AO2 (apply)** **A** `Almost`; **R.** If more than one lozenge shaded	1

Question	Part	Marking guidance	Total marks
04	3	**Mark is for AO2 (apply)** **C** `20`; **R.** If more than one lozenge shaded	1

Question	Part	Marking guidance	Total marks

04	4	**Mark is for AO2 (apply)**
		
		1 mark for either of the following:
		
		`IF num ≤ 1 OR num > 20 THEN`
		
		//
		
		`IF num < 2 OR num > 20 THEN`
		
		I. Case
		A. answers that use an alternative style of pseudo-code
			1

Question	Part	Marking guidance	Total marks
04	5	**Mark is for AO2 (apply)** 16 / 17 / 18 / 19; **R.** If more than one value given and one of the values is not correct. **A.** If more than one value given and all are correct.	1