


Mark scheme

Question			Answer/Indicative content	Marks	Guidance
1	a	i	<ul style="list-style-type: none"> String Integer Real / Float 	3 (AO3)	<p>Accept alternative equivalent correct data types (e.g. single/double/decimal for BP3)</p> <p>Do not accept char for BP1</p> <p><u>Examiner's Comments</u></p> <p>This question was completed very well by the vast majority of candidates, showing that the use of data types are now well understood by centres.</p>
		ii	<ul style="list-style-type: none"> <code>theTeam.length() - 1 / 5</code> <code>count</code> <code>studentName</code> <code>True</code> 	4 (AO3)	<p>Accept <code>6 / theTeam.length()</code> for BP1 (Python).</p> <p>Accept alternative length functions e.g. <code>len()</code></p> <p>Accept <code>count = 5</code> (and equivalents) for BP1. Accept "True" for BP4.</p> <p>Do not allow obvious spaces in variable names.</p> <p>Ignore capitalisation.</p> <p><u>Examiner's Comments</u></p> <p>This was a challenging question revolving around the use of an array that is iterated through to implement a linear search. Many candidates achieved two marks for the first and last points, understanding that the loop would repeat from indexes 0 to 5 and that the value <code>True</code> would be returned if the item was found. As usual, allowance was given for off-by-one errors with this loop because of the prevalence of Python in centres and how loops are count controlled loops are handled in this language.</p> <p>It was far less common for candidates to achieve the middle two marks, perhaps because the level of</p>

				<p>technical knowledge needed was greater. A number of candidates attempted here to fashion a 2D array and refer to multiple indexes, but this was not appropriate given the data structure given. The most challenging mark was certainly the use of <code>count</code> as the index of the <code>theTeam</code> array, with very few candidates correctly identifying this as the missing element.</p> <p> Assessment for learning</p> <p>Centres are encouraged to link the topics of arrays (both single and two-dimensional) to count controlled loops to be confident in answering questions like this one.</p> <p>A very common programming exercise is to iterate through every item in an array, either to search for an item, count or add items or as the pre-cursor for a search. Candidates are expected to have significant practical programming experience over the duration of their studies.</p>																														
b		<ul style="list-style-type: none"><code>javelinThrow</code> set to 14.3 on line 01 and <code>yearGroup</code> set to 10 on line 02<code>score</code> set to 2 on line 06<code>score</code> set to 4 on line 11"The score is 4" output on line 13 with no additional outputs (allow input statements) <p><u>Example</u></p> <table><tr><th>Line number</th><th>javelinThrow</th><th>yearGroup</th><th>score</th><th>Output</th></tr><tr><td>01</td><td>14.3</td><td></td><td></td><td></td></tr><tr><td>02</td><td></td><td>10</td><td></td><td></td></tr><tr><td>06</td><td></td><td></td><td>2</td><td></td></tr><tr><td>11</td><td></td><td></td><td>4</td><td></td></tr><tr><td>13</td><td></td><td></td><td></td><td>The score is 4</td></tr></table> <p>Answer may include lines where no changes or output happens (i.e. lines 3, 4, 5, 7, 8, 9, 10, 12).</p> <p>Where variable doesn't change, current</p>	Line number	javelinThrow	yearGroup	score	Output	01	14.3				02		10			06			2		11			4		13				The score is 4	4 (AO3)	<p>Max 3 if in wrong order or additional (incorrect) changes. Penalise line numbers once then FT.</p> <p>Allow FT for BP4 for current value of <code>score</code>.</p> <p>BP4 must <u>not</u> include comma. Ignore superfluous spaces. Ignore quotation marks.</p> <p>Treat any entry in output column as an output, even if "x", "-" or "0".</p> <p><u>Examiner's Comments</u></p> <p>Trace tables have appeared multiple times in J277 examination papers now and candidates are hopefully familiar with the expectations, which are consistent across series.</p>
Line number	javelinThrow	yearGroup	score	Output																														
01	14.3																																	
02		10																																
06			2																															
11			4																															
13				The score is 4																														

			value may be repeated on subsequent lines.		<p>Most candidates correctly traced through the given algorithm, which was more accessible than usual due to not containing any loops. Where mistakes were made, this was typically to do with either incorrect line numbers being given for each change (this was penalised once only and then subsequent mistakes with line numbers followed through) or additional incorrect output being given.</p> <p>Candidates should be encouraged to simply leave boxes blank if no output is given on a particular line. If (for example) 'x' is written, examiners are unsure whether the candidate meant no output or the letter 'x' should be output. This ambiguity would mean that no mark could be given.</p>
	c	i	<ul style="list-style-type: none"> inputs a value from the user <u>and stores/uses</u> checks min value ($\geq 40.0 / < 40$) checks max value ($\leq 180.0 / > 180$) ...outputs both valid / not valid correctly <u>based on checks</u> <p><u>Example 1 (checking for valid input)</u></p> <pre>h = input("Enter height jumped") if h >= 40 and h <= 180 then print("valid") else print("not valid") endif</pre> <p><u>Example 2 (checking for invalid input)</u></p> <pre>h = input("Enter height jumped") if h < 40 or h > 180 then print("not valid") else print("valid") endif</pre>	4 (AO3)	<p>Answers using AND/OR for BP2 and BP3 must be logically correct e.g. <code>if height ≥ 40 and <u>height</u> ≤ 180. Do not accept <code>if height ≥ 40 and ≤ 180</code></code></p> <p>Answers using OR will reverse output for BP4 (see examples).</p> <p>BP4 needs reasonable attempt at either BP2 or BP3. Need to be sure what is being checked to be able to decide which way around valid/invalid should be.</p> <p>Allow FT for BP4 if reasonable attempt at validating (must include at least one boundary)</p> <p>Ignore conversion to int on input. <code>input</code> cannot be used as a variable name.</p> <p>Greater than / less than symbols must be appropriate for a high-level language / ERL. Do not accept \Rightarrow (wrong way around) or \geq (not available on keyboard). No obvious spaces in variable names. Penalise once and then FT.</p> <p><u>Examiner's Comments</u></p>

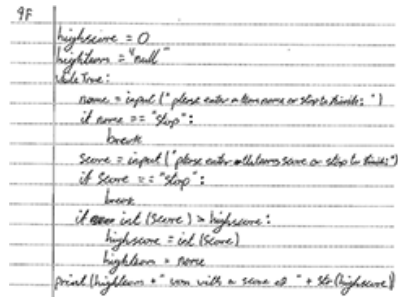
					<p>This question was done very well by the majority of candidates, with multiple ways of achieving the marks possible.</p> <p>One common problem was consistent with Question 3 (b), as again multiple conditions could be evaluated using a single <code>if</code> statement. Candidates needed to refer to their input value for each comparison if they were to achieve full marks. Another common problem was with the boundaries used; the tests must check 40 to 80 inclusive (for valid response) to be marked as correct. Obviously, if an input on these boundaries would produce the wrong output then full marks could not be given.</p> <p>One final common problem involved the use of greater than or equal to signs (and also less than or equal to). The common signs used in mathematics are not available on a typical keyboard and so would not be allowed in Section B of this paper. Instead, <code>>=</code> or <code><=</code> should be used, and these should be the correct way around.</p>
		ii	<ul style="list-style-type: none"> Any normal value (between 40 and 180 inclusive) 40.0 / 180.0 Any value less than 40 / any value greater than 180 / any non-numeric value 	3 (AO3)	<p>No need to include decimals, e.g. accept 50. Ignore cm if given.</p> <p>Answer must be actual data (e.g. 50) and not description of data (e.g. "a value between 40 and 180"). If descriptions given, do not accept this as non-numeric for BP3</p>
	d		<ul style="list-style-type: none"> <code>TeamName</code> only in first space <code>TblResult</code> in second space WHERE <code>...YearGroup = 11</code> 	4 (AO3)	<p>Max 3 if not in correct order / includes other logical errors.</p> <p>Ignore capitals. Do not accept * or additional fields for BP1</p> <p>Spelling must be accurate (e.g. not <code>TblResults</code>).</p> <p>No spaces in field names, penalise obvious spaces once and then FT. Allow quotation marks around field names, table name and 11</p>

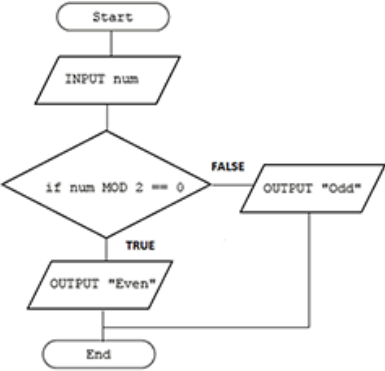
					<p>Accept == for BP4 (invalid SQL but works in some environments)</p> <p><u>Examiner's Comments</u></p> <p>A number of candidates struggled with this question. Problems included spaces in field names, misspelling of the table name (such as <code>TblResults</code> plural when <code>TblResult</code> singular was given) or misunderstanding of the WHERE clause.</p> <p>Allowance was given where == was used for comparison and examiners were instructed to allow this (as this is used for comparison in high-level languages such as Python), although this is incorrect as defined in the most recent ANSI SQL standards.</p>
	e	i	<ul style="list-style-type: none"> any example of simplification / removing data or focussing on data (in the design) <p><u>Examples :</u></p> <ul style="list-style-type: none"> - "focus on student names and events" - "ignore data such as students' favourite subjects" - "store year groups instead of ages or DOB" - "shows student IDs instead of full student details" 	1 (AO3)	<p>Must be applicable to <u>this program</u> (in the context of students and a sports day), not a generic description of what abstraction is. Give BOD where this is unclear.</p> <p><u>Examiner's Comments</u></p> <p>Both questions here asked about abstraction and decomposition of the sports day program. As explained previously in this report, where a scenario or context is given, candidates are expected to use this context. No marks were given by examiners for generic definitions of what the term abstraction or decomposition means.</p> <p>Abstraction in the sports day program could have been for focusing on anything sensible (such as event names) or removing/ignoring anything sensible (such as showing student IDs instead of names). Where the context of the sports day was used, candidates were generally successful in achieving this mark.</p> <p>Decomposition use was more tricky to correctly identify, as many</p>

					<p>candidates simply referred to how already separate data was stored. Where this extended to true decomposition (such as breaking down data into multiple tables, splitting up event data, etc.) this was credited but the average candidate fell short here. A much more successful approach was to discuss the decomposition of the program, such as having a separate algorithm for each event. Candidates attempting this angle of response did very well.</p> <p>Examiners were instructed for both questions to be generous in deciding whether candidates had indeed referred to the sports day context.</p>
		ii	<ul style="list-style-type: none"> any example of breaking down the program into sections/subroutines any example of breaking down the database into tables <p><u>Examples :</u></p> <ul style="list-style-type: none"> - “splits the program up into different events” - “separates the validation routines into subroutines” - “breaks the database down into a table per event” 	<p>1 (AO3)</p>	<p>Must be applicable to <u>this program</u>, not a generic description of what decomposition is. Give BOD where this is unclear.</p> <p>Do not give answers discussing splitting into fields (e.g. split into StudentID, YearGroup, etc).</p> <p>BOD if answer discusses one table but suggests other tables could be used.</p> <p>Do not give answers relating simply to data being split into smaller groups unless this clearly relates to how data is decomposed into tables in the DB.</p> <p>Allow reference to sports day to mean sports day program.</p> <p><u>Examiner’s Comments</u></p> <p>Both questions here asked about abstraction and decomposition of the sports day program. As explained previously in this report, where a scenario or context is given, candidates are expected to use this context. No marks were given by examiners for generic definitions of what the term abstraction or decomposition means.</p>

				<p>Abstraction in the sports day program could have been for focusing on anything sensible (such as event names) or removing/ignoring anything sensible (such as showing student IDs instead of names). Where the context of the sports day was used, candidates were generally successful in achieving this mark.</p> <p>Decomposition use was more tricky to correctly identify, as many candidates simply referred to how already separate data was stored. Where this extended to true decomposition (such as breaking down data into multiple tables, splitting up event data, etc.) this was credited but the average candidate fell short here. A much more successful approach was to discuss the decomposition of the program, such as having a separate algorithm for each event. Candidates attempting this angle of response did very well.</p> <p>Examiners were instructed for both questions to be generous in deciding whether candidates had indeed referred to the sports day context.</p>
f		<ul style="list-style-type: none"> • Input team name AND score and store / use separately • Attempt at using iteration... • ...to enter team/score until "stop" entered • Calculates highest score • Calculates winning team name... • ...Outputs highest score and team name <p><u>Example 1</u></p> <pre> highscore = 0 while team != "stop" team = input("enter team name") score = input("enter score") if score > highscore then highscore = score highteam = team endif endwhile print(highscore) </pre>	6 (AO3)	<p>For BP3, allow "stop" to be entered for either team name or score (or both). Allow third input (e.g. "do you wish to stop?")</p> <p>Allow use of <code>break</code> (or equivalent) to exit loop for BP3.</p> <p>Allow use of recursive function(s) for BP2/3.</p> <p>Initialisation of variables not needed - assume variables are 0 or empty string if not set.</p> <p>Ignore that multiple teams could get the same high score, assume only one team has the highest score.</p> <p>BP4/5 could be done in many ways – see examples. Allow any logically valid method. Allow use of max/sum</p>

		<pre>print(highteam)</pre> <p><u>Example 2 (alternative)</u></p> <pre>scores = [] teams = [] while team != "stop" team = input("enter team name") score = input("enter score") scores.append(score) teams.append(team) endwhile highscore = max[scores] highteam = teams[scores.index(highscore)] print(highscore) print(highteam)</pre>	<p>functions and use of arrays/lists.</p> <p>FT for BP6 if attempt made at calculating highest score/name</p> <p>If answer simply asks for multiple entries (not using iteration), BP2 and 3 cannot be accessed but all others available.</p> <p>For minor syntax errors (e.g. missing quotation marks or == for assignment, spaces in variable names) penalise once then FT.</p> <p>input cannot be used as a variable name.</p> <p><u>Examiner's Comments</u></p> <p>The final question in Section B is expected to be challenging and this proved to be the case, although again was perhaps more accessible than previous papers' final questions.</p> <p>Marks were available for inputs (one mark) and correctly iterating over as required (two marks), with these three marks proving to be the easiest to achieve. The next three marks required significant processing in terms of calculating the highest score and team name from multiple values entered by the user. The vast majority of candidates simply kept a running 'highest score' and updated this on each iteration. Where this was attempted, it was mostly successful. Other candidates attempted more complex solutions, including adding data to arrays and then calculating highest values; where this was done successfully, this of course achieved full marks but frequently small logic mistakes meant that not all marks were given. Centres should encourage candidates to keep their responses simple and not produce over-elaborate solutions if a simpler alternative is available.</p> <p>A common mistake was where candidates attempted to use loops in</p>
--	--	---	---

					<p>the style of <code>for x in list :.</code> In this case, the variable <code>x</code> is a reference to an item in the array and not an index. It would therefore not be appropriate to try to access <code>list[x]</code> later in the code.</p> <p>A significant number of candidates were able to create a solution that fully met the requirements of the question, doing so in an elegant and efficient manner. This is extremely pleasing and show excellent understanding, produced from excellent teaching and significant amounts of practical programming experience.</p> <p>Exemplar 3</p>  <pre> 9F highestscore = 0 highestteam = "null" while True: name = input("Please enter a team name or stop to finish: ") if name == "stop": break score = input("Please enter a team score or stop to finish: ") if score == "stop": break if int(score) > highestscore: highestscore = int(score) highestteam = name print(highestteam + " won with a score of " + str(highestscore)) </pre> <p>The candidate response shown here achieved six out of six marks. Both the <code>name</code> and <code>score</code> are input as required, with this being inside a while loop. Perhaps unconventionally (but acceptably), the candidate has used a <code>break</code> command to end the loop (which otherwise is infinite) upon stop being entered. This could have been more elegantly rewritten as <code>while name != 'stop'</code> but this would not have achieved any further marks.</p> <p>Within the loop, the candidate uses two variables to keep track of the current highest score and associated team, before printing these out in a message once the loop has ended.</p>
			Total	30	
2			<ul style="list-style-type: none"> Correct shape for all three inputs AND outputs (parallelogram) 	4 (AO2)	No need for arrows – lines are acceptable.

			<ul style="list-style-type: none"> Correct shape for decision (diamond) True and False / Yes and No labelled correctly (<i>true/Yes linking to "Even"</i>) All lines joined up correctly and link to End. 		<p>BOD for correct answers that include a loop back to the start</p> <p><u>Examiner's Comments</u></p> <p>The majority of candidates recognised the correct flowchart shapes to be used for a decision and many also were able to use the parallelogram shape for inputs and outputs. However, fewer correctly connected up the boxes to make sure that all paths started and ended at appropriate points and even fewer candidates labelled up the decision box appropriately with True/False or Yes/No. These labels are crucial to be able to follow the path of the algorithm when a decision is made.</p>
			Total	4	
3		i	<p>1 mark each</p> <ul style="list-style-type: none"> Compare to / pick out middle value (which is 6) discard only left side / retain only right side (because $6 < 10$)... ... Compare to / pick out (middle value which is) <u>10</u> 	<p>3 (AO2)</p>	<p>BP1 can be given for generic answer. BP2 and 3 must be linked to data set given</p> <p>For BP2, must remove 1, 2, 5 <u>and</u> 6 from list if discussing individual numbers. Allow FT for BP3 if this done incorrectly.</p>
		ii	<ul style="list-style-type: none"> Data must be sorted / in order 	<p>1 (AO1)</p>	
		iii	<ul style="list-style-type: none"> Merge sort 	<p>1 (AO1)</p>	
			Total	5	
4			<p>Input e.g.</p> <ul style="list-style-type: none"> Name / keyword for video (to be searched for) / search text Controls for watching video (e.g. play / pause) Rating given to video <p>Output e.g.</p> <ul style="list-style-type: none"> Video to be watched / audio Results of search (total / overall / average) rating of video Number of views (of video) 	<p>2 (AO1)</p>	<p>1 mark for a suitable input, 1 mark for a suitable output</p> <p>Allow input / print pseudocode statements if meets mark point(s). Does not have to be valid pseudocode.</p> <p>Do not allow examples of inputs (e.g. "music videos")</p> <p><u>Examiner's Comments</u></p> <p>Identification of inputs and outputs for a problem is covered in specification</p>

			<ul style="list-style-type: none"> Confirmation of data entry / data validity Messages to user / example messages (e.g "enter a rating", "your rating has been saved") in quotation marks 		point 2.1.2 and candidates are expected to be precise with their responses. Answers such as 'video' as input are problematic because the candidate would not upload or use a video for their input. Candidates who were more precise and used terms such as 'name of video' or 'keywords searched for' were positively recognised for this.
			Total	2	
5	a		Linear search	1 (AO1 1b)	Accept Sequential Search
	b		<ul style="list-style-type: none"> Declares variable to store the count (before the for loop) Initialises variable to zero Increments variable by 1 each time the word is found Outputs count value 	4 (AO3 2c)	<u>Example answer :</u> <pre>search = input("Enter a word") count = 0 for i = 0 to 7 if data[i] == search then count = count + 1 next i print(count)</pre> <p>Allow answers that completely rewrite the algorithm as long as points on left met.</p>
	c	i	<ul style="list-style-type: none"> A is sorted B is not (yet) sorted 	2 (AO1 1b) (AO2 1a)	
		ii	<p>Max 3 marks, 1 mark per point</p> <ul style="list-style-type: none"> Compare "or" with next item on left ... "or" is smaller than "when" so... ... it repeats comparison with next item inserts "or" / item in correct place, based on comparisons Repeat to insert "it" 	3 (AO2 1b)	<p>Max two marks if candidate simply states to insert "or" then "it" into the correct place without discussing how this is determined. Max one if generic answer with no reference to given array.</p> <p>Allow answers referring to swapping items down the array to get to the correct position.</p>
	d		<ul style="list-style-type: none"> Merge sort 	1 (AO1 1b)	Allow other sorting algorithms that use divide and conquer
			Total	11	

6			<ul style="list-style-type: none"> decomposition abstraction thinking sequence 	4 (AO2 1a)	Ignore minor misspellings																																																
			Total	4																																																	
7	a	i	<ul style="list-style-type: none"> DroneID, Mileage FROM AND > 50000 	4 (AO3 2b)	Accept SELECT * or selection of additional fields for BP1.																																																
		ii	<ul style="list-style-type: none"> if <u>Mileage-LastCheck > 10000</u> Output "Check" and "No Check" or equivalent correctly <u>based on logical check for BP1</u> 	2 (AO3 2a)	BP1 could be >, < or either in words. Ignore case and minor misspellings. BP2 (output) could be either way around depending on comparison for BP1.																																																
	b		1 mark per row <ul style="list-style-type: none"> c = 90 on line 05 c = 900 on line 05 pilotCode = HK900 on line 07 HK900 output on line 08 	4 (AO3 1)	Ignore additional lines that do not affect outcome. FT for missing or incorrect line numbers. FT for output based on incorrect tracing of loop. <table border="1"> <thead> <tr> <th>Line number</th><th>a</th><th>b</th><th>c</th><th>pilotCode</th><th>Output</th></tr> </thead> <tbody> <tr> <td>01</td><td>H</td><td></td><td></td><td></td><td></td></tr> <tr> <td>02</td><td></td><td>K</td><td></td><td></td><td></td></tr> <tr> <td>03</td><td></td><td></td><td>9</td><td></td><td></td></tr> <tr> <td>05</td><td></td><td></td><td>90</td><td></td><td></td></tr> <tr> <td>05</td><td></td><td></td><td>900</td><td></td><td></td></tr> <tr> <td>07</td><td></td><td></td><td></td><td>HK900</td><td></td></tr> <tr> <td>08</td><td></td><td></td><td></td><td></td><td>HK900</td></tr> </tbody> </table>	Line number	a	b	c	pilotCode	Output	01	H					02		K				03			9			05			90			05			900			07				HK900		08					HK900
Line number	a	b	c	pilotCode	Output																																																
01	H																																																				
02		K																																																			
03			9																																																		
05			90																																																		
05			900																																																		
07				HK900																																																	
08					HK900																																																
			Total	10																																																	
8	a		<u>total</u> = num1 + num2	1 (AO3 2b)	Allow other logically valid responses that result in <code>total</code> storing the correct value. Accept other suitable assignment operators (e.g. <code>←</code>) e.g. <code>total = sum(num1, num2)</code> <code>total = num2 + num1</code>																																																

					<pre>x = num1 + num2 total = x</pre> <p>Ignore any values given to the variable. Ignore capitalisation and minor misspelling. Ignore superfluous code that does not affect outcome.</p> <p><u>Examiner's Comments</u></p> <p>Candidates appear to be getting more confident at answering simple programming/pseudocode questions such as this. The majority of responses included code written to produce the required outcome. The use of multiple steps was allowed.</p>
	b	i	<code>print(12 ^ 2)</code>	1 (AO2 1a)	<p>Accept ** or other sensible operator that indicates raising to a power.</p> <p>If pseudocode operator given, must be a single word/symbol (e.g. <code>pow</code>), not containing spaces.</p>
		ii	<code>if number MOD 2 == 0 then</code>	1 (AO2 1a)	<p>Accept % or other sensible operator that indicates modulus</p> <p>If pseudocode operator given, must be a single word/symbol (e.g. <code>modulo</code>), not containing spaces.</p>
		iii	<code>difference = measurement1 - measurement2</code>	1 (AO2 1a)	<p>Accept other sensible operator that indicates subtraction.</p> <p>If a pseudocode operator given, must be a single word/symbol (e.g. <code>minus</code>), not containing spaces.</p>
	c		<p>1 mark each:</p> <ul style="list-style-type: none"> Start is set to 3 on line 01 and 3 is output on line 03. 2, 1 and 0 are output on next 3 iterations with start updated to 2, 1, 0, -1 on correct line numbers. Finished is output on line 06 	3 (AO3 2c)	<p>Ignore lines 02 and 05 in answer unless these change or output any values.</p> <p>Candidate may repeat start value when unchanged, this is acceptable.</p> <p>Penalise incorrect or missing line numbers or <u>additional</u> output once only then FT. This includes where variable change and output appear on the same line.</p> <p>-1 must not be output for BP2</p>

Line	start	Output	
01	3		BP1
03		3	
04	2		BP2
03		2	
04	1		
03		1	
04	0		BP3
03		0	
04	-1		
06		Finished	

Penalise missing or incorrect output once only for BP1 and FT for missing or incorrect output for BP2.

Finished may be with or without quotes. Ignore case or minor spelling error.

Max 2 marks if any incorrect output or changes to `start`.

Do not accept calculated values of start (e.g. $3-1$)

Examiner's Comments

This question assessed candidates' ability to trace through and understand the steps taken by an algorithm. This also tested their understanding of condition-controlled loops. Many responses were very successful with this and achieved full marks.

Mistakes tended to be with identifying the line number where each change occurred or outputting values that were not actually output (e.g. -1).

Examiners were instructed to only penalise a misunderstanding once. Where (for example) line numbers were incorrect, this would still have allowed 2 out of 3 marks to be achieved.

Total

7

1 mark each

- stores/holds **data/value/name/names** [`pos`]
- ...so (value) can be changed / swapped / moved / overwritten / inserted
- ...without being lost.
- will be assigned to `names[pos-1]`

2
(AO2
1b)


Do not allow answers that clearly refer storing the position / index (or any other out of context data) for BP1; it is the name itself that is being stored, not the position. If unclear, allow BOD.

e.g. do not allow "holds the values of the index / holds value for position of the name".

Allow FT for subsequent points.

Examiner's Comments

				<p>This question was an excellent discriminator in terms of candidate achievement. Many responses correctly described the <code>temp</code> variable as storing the name so that it could be overwritten during the process of swapping values. Partial credit was given for responses simply commenting on it storing data, as this is essentially the purpose of any variable.</p> <p>However, credit was not given where candidates discussed storing a position/index, as this was not the case; the <code>pos</code> variable is used as the index of the array. The content of this array index (a name) is stored in the <code>temp</code> variable.</p> <p>Where responses are open to interpretation, the mark scheme attempts to credit as many of these as possible. However, incorrect responses (such as reference to the position here instead of the name) are not credited.</p>
	b	<p>1 mark</p> <ul style="list-style-type: none"> • do not know how many iterations / swaps needed • do not know (at run time) how many times the value will change positions • do not know how many times a condition-controlled loop will need to run / execute <p>1 mark</p> <ul style="list-style-type: none"> • condition controlled loops run while/until a condition is true / is false / until a condition is met • repeats while value in <code>[pos-1]</code> is larger than value in <code>[pos]</code> / while (further) swap needed • will swap value until in correct position / will swap whilst in incorrect position • More efficient than / does not need to iterate as many times as count controlled / for loop 	<p>2 (AO2 1b)</p> <p>Max 1 from each section, 2 marks total.</p> <p>Do not allow “while names are in the wrong order”.</p> <p>BP4 must have reference to <u>checking</u> a condition / condition being met, not just having a condition.</p> <p><u>Examiner’s Comments</u></p> <p>Previous questions such as Question 2(a) assessed knowledge (AO1). This question focused on the application (AO2) of a technique (condition-controlled loops) to the algorithm given (an insertion sort). This assessed candidates’ understanding of the process an insertion sort follows to sort values.</p> <p>The J277 specification is clear that candidates do not have to remember the code for this algorithm. But the specification does state that</p>	


				<p>candidates must be able to understand the main steps of the algorithm and identify the algorithm if given the code for it. In this case, candidates were given all of the code for the algorithm.</p> <p>Candidates generally found this question challenging. Many responses simply repeated the question and discussed sorting values. More successful responses understood that the inner loop is the part of the algorithm that moves the <code>name</code> repeatedly in the list and that we do not know how many times this move needs to be made. Further to this, it is the condition that the <code>name</code> is in the correct position (or even better, an explanation of how this is decided on) which ends the loop.</p> <div style="text-align: center;">  Assessment for learning </div> <p>When asked to explain why a particular technique is used, it is often useful for candidates to think about why alternative options have not been used. In this question, thinking about what would happen if a count-controlled loop had been used as the inner loop may have given candidates the insight to discuss why a condition-controlled loop has been used.</p>
	c	i	<p>1 mark each for insertion and bubble sort, max 2</p> <p>Insertion sort:</p> <ul style="list-style-type: none"> • inserts/moves values into correct position • inserts value once (then in correct position) • stops when end of array reached / completes in one pass through the array • moves items down the array / left • start of array becomes sorted first 	<p style="text-align: center;">2 (AO1 1b)</p> <p>Answer must reference both bubble sort and insertion sort for 2 marks except if efficiency mark plus expansion given.</p> <p>Allow reference to big O for efficiency discussion.</p> <p>Only award efficiency once. Only award fewer iterations once</p> <p>Do not accept “completes in one iteration” for insertion sort.</p> <p>Accept list / data / values / etc for array.</p>

			<ul style="list-style-type: none"> creates a sorted array within an array / has a sorted/unordered partition / section / list starts on 2nd value more efficient/faster than bubble sort ... because fewer iterations / comparisons (on average) ... when data more scrambled <p>Bubble sort :</p> <ul style="list-style-type: none"> compares/swaps pairs of values value is repeatedly moved/swapped (until in correct position) repeats if a swap has been made / needs multiple passes will complete a final iteration once sorted (to check for no swaps needed) moves items up the array end of array becomes sorted first moves/bubbles the highest value to the top less efficient/slower than insertion sort (on large sets of values) ... more iterations / comparisons (on average) ... when data more scrambled 		<p>“when data more scrambled” only makes sense when discussing efficiency/speed, do not give marks for saying that either can handle data that is more scrambled (they both can sort data however it is arranged).</p> <p>Do not accept “bubble/insertion sort does not” for 2nd mark.</p> <p><u>Examiner’s Comments</u></p> <p>Question 3(c)(i) and Question 3(c)(ii) asked candidates to describe one difference and two similarities between an insertion sort and a bubble sort.</p> <p>Examiners were instructed to be generous in their interpretation of the requirements of both algorithms. This included considering both their algorithmic implementation and the understanding of how these are typically described on a classroom whiteboard.</p> <p>As such, many combinations of answers could gain the marks available.</p>
		ii	<p>1 mark each to max 2 e.g.</p> <ul style="list-style-type: none"> Both produce a sorted list / array Both work in place / without duplicating data / without using divide and conquer Both need a temporary variable Both swap values Both use loops / iteration / repeats Both loops are nested / inside each other Both (may) need multiple passes Both use selection Both work with an array / list data structure Both work from left to right Both build up sorted list one item at a time (after every pass) Both compare (pairs of) values 	2 (AO1 1b)	<p>Allow reference to both sorting / putting items into order for BP1.</p> <p>“Allows sorting of numbers and strings” meets BP1</p> <p>Allow answers relating to not needing additional memory as BP2.</p> <p>Allow “breaking into smaller lists” as divide and conquer for BP2.</p> <p>If answer is a statement (e.g. “uses loops”), assume candidate is talking about both algorithms doing this.</p> <p><u>Examiner’s Comments</u></p> <p>Question 3(c)(i) and Question 3(c)(ii) asked candidates to describe one difference and two similarities</p>

			<ul style="list-style-type: none"> Both (typically) less efficient / slower than merge sort (or other sorting algorithms) Both inefficient / slow for larger / unsorted lists / efficient for small / (nearly) sorted lists Both start by comparing first two values 		<p>between an insertion sort and a bubble sort.</p> <p>Examiners were instructed to be generous in their interpretation of the requirements of both algorithms. This included considering both their algorithmic implementation and the understanding of how these are typically described on a classroom whiteboard.</p> <p>As such, many combinations of answers could gain the marks available.</p>
			Total	8	
10			<p>1 mark each to max 6</p> <ul style="list-style-type: none"> Initialise / declare <code>score</code> (to zero) before use, outside of any loop Generates 2 random numbers <u>between 1 and 10</u> Inputs answer from user displaying suitable numbers Checks if input is <u>correct answer</u>... ... if correct adds 1 to <code>score</code> Repeats BP2 to 5 three times (for bullet points attempted) Outputs <code>score</code> <u>after reasonable attempt at counting</u> 	6 (AO3 2b)	<p>No need to cast data to string/integer.</p> <p>If random numbers chosen, BP3 must use these. If no random numbers chosen, allow manually setting values</p> <p>BP6 can be awarded for either a loop repeating 3 times or the same code written out 3 times</p> <p>BP5 can be given FT if sensible attempt at BP4</p> <p>Do not award BP6 if same numbers used for every question. Must pick new values each time.</p> <p>Do not penalise potential off by 1 errors for looping (Python) or random number generation</p> <p><u>Example answer</u></p> <pre>score = 0 for count = 1 to 3 num1 = random(1, 10) num2 = random(1, 10) ans = input("What is" + num1 + " + " + num2 + "?") if ans = num1 + num2 then score = score + 1 end if next count print("You scored " + score)</pre>

					<p><u>Examiner's Comments</u></p> <p>As this question appears in Section A, candidates are free to respond in any suitable way, including using flowcharts, structured English, pseudocode or a high-level language.</p> <p>The majority of high scoring responses used a high-level language consistently.</p> <p>Where flowcharts or structured English were used, responses needed to clearly show the steps to be taken and not simply repeat the question to achieve marks.</p> <p>The given question is already decomposed for candidates and many were able to use these bullet points to build a solution that achieved the majority of marks available.</p> <p>Many responses used random number generation and iteration to create an elegant response that met all mark points. This was pleasing to see and it is extremely encouraging that candidates can use techniques such as these where appropriate without being prompted.</p> <p>Other responses manually repeated asking the required questions; on this occasion, these were also credited and could have achieved full marks.</p> <p>Where a mistake was made in one section (such as with iteration), examiners were instructed to use FT (follow through) where possible. This allowed candidates to score marks in later sections if their responses were logically constructed. This is to be fair to candidates so that mistakes are only penalised once in any given question.</p> <p>A significant number of responses did not access many marks in this question. This would suggest that</p>
--	--	--	--	--	--

					more practical programming time in lessons would be beneficial.															
			Total	6																
11			<div>1 mark per row</div> <table><thead><tr><th>Statement</th><th>Low-level</th><th>High-level</th></tr></thead><tbody><tr><td>The same language can be used on computers that use different hardware</td><td></td><td>✓</td></tr><tr><td>It allows the user to directly manipulate memory</td><td>✓</td><td></td></tr><tr><td>It allows the user to write English-like words</td><td></td><td>✓</td></tr><tr><td>It always needs to be translated into object code or machine code</td><td></td><td>✓</td></tr></tbody></table>	Statement	Low-level	High-level	The same language can be used on computers that use different hardware		✓	It allows the user to directly manipulate memory	✓		It allows the user to write English-like words		✓	It always needs to be translated into object code or machine code		✓	4 (AO1 1b)	<p>No mark if more than 1 tick for that row.</p> <p>Allow other indications of choice (e.g. cross) as long as clear.</p> <p><u>Examiner’s Comments</u></p> <p>This question was answered well by many candidates. The strongest responses showed a good understanding of the difference between low-level and high-level languages. Incorrect responses tended to be on the first row related to portability. A high-level language such as Python is portable, with translators available for many different types of processors. A low-level language is specific to one type of processor.</p>
Statement	Low-level	High-level																		
The same language can be used on computers that use different hardware		✓																		
It allows the user to directly manipulate memory	✓																			
It allows the user to write English-like words		✓																		
It always needs to be translated into object code or machine code		✓																		
			Total	4																
12	a		<ul style="list-style-type: none"><code>score = score + 1 /</code> <code>score +=1 / score++</code>	1 (AO3 2b)	<p>Allow other logically correct answers that result in score increasing by one and being overwritten. <u>Do not accept</u> <code>score + 1 / score = +1</code></p> <p>Accept valid structured English answers that refer to score increasing and overwriting the existing value by one. e.g. “score becomes/equals score plus one”</p> <p>Ignore any superfluous code that does not affect the outcome</p> <p><u>Examiner’s Comments</u></p> <p>Most candidates were able to temporarily add one to the value of score (by using code such as <code>score + 1</code>). Fewer were able to assign this</p>															

					<p>new value to the variable of score and therefore fully answer the question. Answers such as <code>score = score + 1</code>, <code>score++</code> or <code>score += 1</code> were all accepted, as were any longer responses that used intermediate variables. As long as the value of score ended up being incremented by one, examiners were instructed to give credit.</p> <div> Assessment for learning</div> <p>One notable response that was not allowed was <code>score =+1</code>. This statement assigns the value of 'positive 1' to score, overwriting the previous value held.</p> <p>This is a good example of the precision required to gain marks. Candidates with practical programming experience are more likely to recognise this.</p>
	b		<ul style="list-style-type: none">• Decomposition• Abstraction	2 (AO1 1a)	<p>Correct answer only. Ignore spelling errors.</p> <div><u>Examiner's Comments</u></div> <p>This question was answered well by the majority of candidates.</p>
			Total	3	

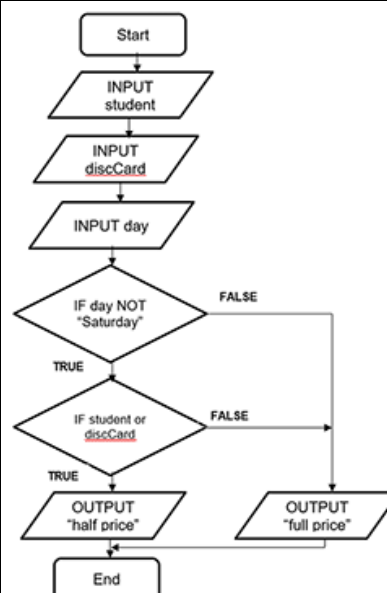
13 a

- Start and end/stop with all boxes connected, no boxes that do not lead to another box (no arrows needed)
- Input **three** variables using **parallelogram shape**
- Checks all three criteria (day, student, discount card) using **diamond shape(s)** with two lines from each
- ...**Outputs** "full price" with correct conditions using **parallelogram shape**
- ...**Outputs** "half price" with correct conditions using **parallelogram shape**

Guidance for correct outputs

Conditions	Outcome
Not Saturday and (either a student or has a discount card).	Half price
Saturday or (not a student and doesn't have a discount card).	Full price

Saturday	Student	Discount Card	Outcome
N	N	N	Full price
N	N	Y	Half price
N	Y	N	Half price
N	Y	Y	Half price
Y	N	N	Full price
Y	N	Y	Full price
Y	Y	N	Full price
Y	Y	Y	Full price



Question asks for a flowchart. Answers as pseudocode, high level language or other forms are not acceptable 9 (NAQ).

5
(AO3
2a)

BP 4 and 5 only to be awarded if all decisions ensure correct output and clear what the decisions are. FT for incorrect shapes used or no inputs as long as decisions are logically correct. Must attempt all three decisions.

Allow calculation of half price / full price instead of message but this must still be output.

Inputs / decisions may be presented as individual or combined boxes but must still store as three variables.


Penalise lack of parallelogram for input/output once only then FT

BOD parallelogram shapes if not sure whether input or output as long as context is clear (e.g inputs at start, outputs at end)


Examiner's Comments

Most candidates who answered this question were comfortable using the correct flowchart symbols listed in the specification. A mark was available for including suitable start / end symbols and connecting all other symbols, so even candidates who

					<p>may struggle should feel confident of being able to access some marks.</p> <p>Marks were dropped when responses did not include suitable inputs to the algorithm. The first bullet point in the question stem was clear that these were required.</p> <p>Candidates achieving lower scores tended to group decisions so that any processing was removed (such as “can they have a discount?”). More successful responses decomposed the problem and used a succession of smaller decisions (“do they have a discount card?”, “are they a student?”, “is it Saturday?”). These decisions then point towards the correct outputs.</p>
	b		<ul style="list-style-type: none"> Number of people (at the table) / whether there are more than 5 people or not Choice between percentage and value / actual value of both percentage, value 	<p>2 (AO32a)</p> <p><u>Examiner’s Comments</u></p> <p>This question type is new to the J277 specification and asked candidates to list inputs that will be required as part of the planning stage for an algorithm. Successful candidates were able to identify the raw data needed from the user in order to be able to solve the problem.</p> <p>Unsuccessful responses tended to simply rewrite the question or miss out key information.</p>	
			Total	7	
14	a		<ul style="list-style-type: none"> Merge into correct sorted lists of size 2 (12 45 / -99 100 / -13 0 / -27 17) 	<p>3 (AO2 1b)</p>	Do not credit BP3 simply for a sorted list.

			<ul style="list-style-type: none"> • Merge into correct sorted lists of size 4 (-99 12 45 100 / -27 -13 0 17)... • ...Merge into correct sorted list of size 8 (-99 -27 -13 0 12 17 45 100) 	<p>Groups of numbers must clearly be the correct size.</p> <p>Do not all allow answers that show lists being merged and then sorting in place, this is incorrect.</p> <p><u>Examiner's Comments</u></p> <p>Most candidates were able to correctly demonstrate the required steps of the merge sort algorithm. This involves merging the set of 8 individual lists into 4 lists of size 2, then 2 lists of size 4, then 1 list of size 8. At each point, each list produced must be in order. It is this process of merging lists together that sorts the values.</p> <p>No mark was given simply for showing the final list sorted unless the previous step(s) had been followed; any candidates an alternative sorting algorithm (for example a bubble sort) would achieve 0 marks even though the final outcome is the same.</p> <p> Misconception</p> <p>Many candidates showed merged lists which were out of order to start with, and then sorted these merged lists. This is highly inefficient and not what the standard merge sort algorithm describes.</p> <p>In merge sort, the process of merging lists together produces the sorted list. There does not need be any second step to sort values as they should already be in order.</p> <p>Exemplar 2</p>
--	--	--	---	---

				<p>This exemplar shows the misconception detailed previously. Here, the candidate firstly merges together the individual lists into 4 lists of size 2, but these lists are not all in order (e.g. 17, -27 is incorrect).</p> <p>The second step then merges these lists together into 2 lists of size 4, but again, these are not in order. An unnecessary next step is then shown where the lists are then sorted in place.</p>
b		<p>Any four bullet points for 1 mark each</p> <ul style="list-style-type: none"> • Select / choose / pick middle number (or left/right of middle as even number) and .. • ...check if selected number is equal to / matches target number (<i>not just compare</i>) • ...if searched number is larger, discard left half / if searched number is smaller, discard right half • Repeat until number found • ... or remaining list is of size 1 / 0 (number not found) 	<p>4 (AO1 1b)</p>	<p>Do not allow “split the list in half” on its own as first step, this is incorrect.</p> <p>Can get BP1 and 2 in one step (e.g. “check if the middle number is the one we’re looking for”)</p> <p>For BP3, accept focussing on correct half</p> <p>Repeat (BP4) must be in the context of an attempt at a binary search. Allow correct reference to recursion.</p> <p>“until number is not in the list” is NE for final BP. Need to explain how this is known.</p> <p><u>Examiner’s Comments</u></p> <p>This question asks candidates to describe how a binary search would look for a number in a sorted list. Most candidates answered this well. The most common mark for this question was 3 out of 4.</p> <p>The majority of candidates clearly understood that the middle value is</p>

				<p>chosen and compared against the value being searched for. However, almost all candidates then went on to discuss which half of the list should be discarded or focused on based on this comparison.</p> <p>One possibility is that the middle value is the number we are searching for. Few candidates discussed this possibility. As the process repeats, this check is important if the number is ever to be found!</p> <p> Assessment for learning</p> <p>When discussing a binary search there are three possibilities when comparing the middle value to the number being searched for.:</p> <ol style="list-style-type: none"> 1. the middle value is higher than needed (in which case the left half of the list should be focused on) 2. the middle value is lower than needed (in which case the right half of the list should be focused on) 3. the middle value is equal to the value needed (in which case the value has been found and the process can stop). <p>Most responses only considered the first two of these possibilities. This is worth discussing when teaching this topic.</p>
	c		<p>1 mark each</p> <ul style="list-style-type: none"> Starting with the first value checking all values in order 	<p>2 (AO1 1b)</p> <p>2nd bullet point must cover both ideas of checking all of the values AND being done in order.</p> <p>“Checks each value” / “one by one” / “step by step” by itself is NE, does not say in order.</p> <p>Do not accept “repeat until value found” for BP2 (question says number is not in the list)</p> <p>“Checks each value from beginning</p>

					<p>to end” implies order so gets both BP1 and BP2.</p> <p><u>Examiner’s Comments</u></p> <p>Responses to this question generally lacked precision.</p> <p>The question specifically asks about searching for a number that is not in the list. However, many candidates discussed stopping when the value had been found. This is clearly not possible given the scenario.</p> <p>Many responses discussed checking each number but did not state whether this would be in a particular order. Some candidates discussed checking values randomly, or even from the right down to the left.</p> <p>For this questions, candidates may have known that a linear search starts at the left most number and then checks each value in the list in order until reaching the end of the list. However, few responses stated this or anything close to this.</p> <p>This level of precision (and conversely, not assuming that steps are obvious) should be encouraged within GCSE Computer Science.</p>
			Total	9	
15			<ul style="list-style-type: none"> input <u>and stores/uses</u> value <u>with message</u> attempt at repeating... ... <u>correctly</u> repeats number of times given as input ... <u>correctly</u> take number as input within loop <u>and</u> calculates total of these numbers ... <u>correctly</u> calculate an average (total/num) Output <u>both</u> total and average 	<p>6 (AO3 2b) (AO3 2c)</p>	<p>e.g.</p> <pre>num = input("Enter how many numbers") for x = 1 to num temp = input("Enter a number") total = total + temp next x print(total) print(total / num)</pre>

					<p>If flow chart used, correct shapes needed.</p> <p>Allow tolerance of 1 with number of loops for BP3 with for loops</p> <p>BP1 requires input with a message (can be two statements, e.g. print and then input or combined. Input must be stored or used.</p> <p>BP3, 4, 5 must be logically correct to be credited Ignore non-initialisation of total</p> <p>BP 5 can be given as FT as long as an attempt has been made at working out a total within the loop.</p> <p>BP6 can be given as FT long as attempt made at total <u>and</u> average (not necessarily in a loop)</p> <p><u>Examiner's Comments</u></p> <p>As this question appeared in Section A, candidates are free to respond in any suitable way, including using flowcharts, structured English, pseudocode or a high-level language. The majority of candidates who scored highly tended to use a high-level language consistently.</p> <p>The question is already decomposed for candidates and many were able to use these bullet points to build a solution that achieved the majority (or all) marks available.</p> <p>Where mistakes were made, these tended to be with repeating code. For example, many candidates repeated the process of adding values up without repeatedly asking for a new number (and therefore continually adding the same number).</p> <p>Other candidates used condition-controlled iteration to repeat the process but did not make sure that the condition being evaluated ever returned a False value, therefore</p>
--	--	--	--	--	--

					<p>repeating infinitely.</p> <p>Where a mistake was made in one section (such as with iteration), examiners were instructed to use FT (follow through) marks where possible if later sections were logically constructed. This made sure that marks could be given where appropriate.</p> <p>A few candidates were not able to access many of the marks available in this question, suggesting that they would benefit from more practical programming time in lessons.</p>
			Total	6	
16	a	i	<ul style="list-style-type: none"> Checks that both <code>firstname</code> and <code>surname</code> are not empty... Checks that <code>room</code> is either "basic" or "premium"... Checks that <code>nights</code> is between 1 and 5 (inclusive)... ...Outputs "NOT ALLOWED" (or equivalent) if <u>any</u> of the 3 checks are invalid (must check all three) ...Outputs "ALLOWED" (or equivalent) <u>only</u> if all three checks are valid (must check all three) <p><i>Note : output marks are given for if entire system produces the correct output. For example, If a user enters a valid name and room but an invalid number of nights, the system should say "NOT ALLOWED" (or equivalent). If this works and produces the correct response no matter which input is invalid, BP4 should be given.</i></p> <p><i>The same process holds for the valid output - if (and only if) three valid inputs results in an output saying "ALLOWED" (or equivalent), BP5 should be given. Do not give this if ALLOWED is printed when (for example) two inputs are valid and one is invalid.</i></p> <p><i>For any output marks to be given, a sensible attempt must have been made at all three checks. These may not be</i></p>	<p>5 (AO3 2a)</p>	<p>Must have some attempt at <u>all three checks</u> to give output mark(s). Check for <code>nights</code> must check both upper and lower limits.</p> <p>Iteration can be used as validation if input repeatedly asked for until valid answer given.</p> <p><u>Do not accept</u> logically incorrect Boolean conditions such as <code>if firstname or surname == ""</code></p> <p>Do not accept \geq or \leq for \geq, \leq. Ignore capitalisation</p> <p>e.g.</p> <pre>valid = True if firstname == "" or surname == "" then valid = False end if if room != "basic" and room != "premium" then valid = False endif if nights < 1 or nights > 5 then valid = False endif if valid then print("ALLOWED") else print("NOT ALLOWED") endif</pre>

			<p><i>completely correct (and may have been penalised in BPs 1 to 3) but should be enough to allow the FT marks for output.</i></p>	<p>BP1 to 3 can check for valid or invalid inputs. . Pay particular attention to use of AND / OR. Only give marks for output if these work together correctly.</p> <p>Example above shows checking for invalid data. Checks for valid data equally acceptable Examples shown below:</p> <ul style="list-style-type: none"> • <code>if firstname != "" <u>and</u> surname != ""</code> • <code>if room == "basic" <u>or</u> room == "premium"</code> • <code>if nights >= 1 <u>and</u> nights <= 5</code> <p><u>Examiner's Comments</u></p> <p>This question stretched the understanding of even highly-achieving candidates and it was not uncommon to see low scoring responses.</p> <p>Misunderstanding of Boolean operators (AND and OR) within selection (IF) statements was something that affected a lot of candidate responses.</p> <p>As this question was in Section B, candidates needed to respond in OCR Exam Reference Language or a high-level language. Responses must be logically correct to gain the marks. As each check is two individual checks that both need to pass, responses can quickly get relatively complicated.</p> <p>As can be seen from the mark scheme, advice and examples were given to examiners to make sure that candidates who were able to successfully navigate this logic chain were credited.</p>
--	--	--	---	--

					<div data-bbox="954 107 1024 183" data-label="Image"></div> <div data-bbox="1082 125 1292 161" data-label="Section-Header">Misconception</div> <p data-bbox="954 219 1449 542">Checking whether a room is either basic or premium can be done in multiple ways. Candidates can either check for the positive (i.e. check that it is either basic or premium) or check that for the negative (i.e. check whether it is something else). However, there are many common errors that were seen :</p> <ul data-bbox="1002 582 1449 1841" style="list-style-type: none"> • <code>IF room == "basic" or "premium"</code> is incorrect as the second part of the statement is not evaluated against anything. This was perhaps the most common mistake. • <code>IF room == "basic" or room == "premium"</code> is correct and checks for validity. • <code>IF room == basic or room == premium</code> is incorrect as the lack of string delimiters means that basic and room would be treated as variables rather than strings. • <code>IF room != "basic" or room != "premium"</code> is also incorrect. This checks for invalid input but because or is used, only one condition needs to be True for the whole statement to be True. This means that if basic is entered, it would be classed as invalid (as it isn't premium) and vice-versa. There is no way for any entry in this example to be classes as valid. • <code>IF room != "basic" and room != "premium"</code> is correct. This checks for invalid inputs but needs both conditions to be True. <p data-bbox="954 1912 1449 1984">The same explanation follows for the other two necessary checks.</p>
--	--	--	--	--	--


				<div>Exemplar 3</div> <div><pre>if first Name == "" OR surname == "" then print("NOT ALLOWED") else if room != "basic" AND room != "premium" then print("NOT ALLOWED") else if night < 1 OR night > 5 then print("NOT ALLOWED") else print("ALLOWED") end if end if end if end if</pre></div> <div><p>This exemplar shows a fully correct response. The candidate checks for invalid responses and correctly uses Boolean operators to check multiple criteria at each step. If any check returns True, “Not allowed” is printed and the program ends. Efficient use of if ... else ... means that the next check only proceeds if the previous check returns False.</p><p>If all three checks return False, the final else is triggered to print “Allowed”.</p><p>It must be noted that this is only one way of achieving full marks. An equivalent program that checks for valid responses at each turn would also be possible. Candidates should be encouraged to use whatever structure they feel is sensible. If a response can logically be followed then it will achieve high marks.</p></div>												
	ii	<div><ul style="list-style-type: none">• Normal• 1 or 5 (not 0 or 6 as says allowed)• Any numeric value except 1 to 5 / any non-numeric input (e.g. "bananas")</div>	<div>3 (AO3 2c)</div>	<div>Allow other descriptions that mean normal (e.g. valid / typical / acceptable)</div> <table><tr><th>Test data (number of nights)</th><th>Type of test</th><th>Expected output</th></tr><tr><td>2</td><td>Normal</td><td>ALLOWED</td></tr><tr><td>1 / 5</td><td>Boundary</td><td>ALLOWED</td></tr><tr><td>e.g. 7</td><td>Erroneous/Invalid</td><td>NOT ALLOWED</td></tr></table> <div>Examiner’s Comments</div>	Test data (number of nights)	Type of test	Expected output	2	Normal	ALLOWED	1 / 5	Boundary	ALLOWED	e.g. 7	Erroneous/Invalid	NOT ALLOWED
Test data (number of nights)	Type of test	Expected output														
2	Normal	ALLOWED														
1 / 5	Boundary	ALLOWED														
e.g. 7	Erroneous/Invalid	NOT ALLOWED														

					This question was answered well by the majority of candidates.
	b	i	<ul style="list-style-type: none"> Function header for newPrice... ...taking (at least) two parameters ...correctly calculates price based on parameters (if present) <u>within function</u> .../ ... returns this calculated price 	4 (AO3 2b)	<p>BP1 must be clear that a new function is being defined. E.g. <code>function / def keyword</code>. Allow FT for subsequent marks if not present.</p> <p>Ignore any code outside attempt at function definition.</p> <p>Ignore additional parameters. Ignore inputs or additional code as long as these do not overwrite parameters or affect operation of function.</p> <p>If inputs used instead of parameters, FT for BP3. Allow use of <code>else</code> for second room type in BP3.</p> <p>Attempt at calculation needed to award BP4. Must return (not output) value. Return can be done e.g. in VB by assigning to function name (e.g. <code>newPrice = price</code>)</p> <p>e.g.</p> <pre>function newPrice(nights, room) if room == "basic" then if room== 60 * nights then elseif room == "premium" then price = 80 * nights endif return price endfunction</pre> <p><u>Examiner's Comments</u></p> <p>Defining functions appeared to be a concept that candidates did not fully understand.</p>

					<p>Where a candidate did not attempt to define a function and instead simply calculated the price needed, very few (if any) marks were available.</p> <p>Successful responses could have been constructed from any suitable function definition keyword such as <code>function</code> (OCR ERL, VB, JavaScript, etc), <code>def</code> (Python) or others. Answers in C#, Java or other languages referring to methods were also accepted.</p>
		ii	<ul style="list-style-type: none"> • Call function <code>newPrice...</code> • ...with <u><code>("premium", 5)</code></u> as parameters • ... Output returned value 	<p>3 (AO3 2b)</p>	<p>Order of parameters not important</p> <p>"premium" must use string delimiters (e.g. "quotes")</p> <p>e.g.</p> <pre>print(newPrice("premium", 5))</pre> <pre>x = newPrice(5, "premium")</pre> <pre>print(x)</pre> <p>Do not allow function definitions for BP1</p> <p>Ignore capitalisation of <code>newPrice</code></p> <p>Candidate could store returned value in a variable and then print this, or store parameters in variables before passing in - these are all acceptable</p> <p>Ignore any superfluous code given</p> <p>Do not credit answers where <code>newPrice</code> is overwritten prior to use.</p> <p>Ignore spaces. Allow function call if brackets missing (e.g. <code>newprice</code> instead of <code>newprice()</code>)</p> <p><u>Examiner's Comments</u></p> <p>Even if candidates were not able to create a function, this question was independent to (i) and so marks were available for simply using the function to output a value.</p>

					<p>Candidate found this question challenging. Many candidates called the function but most did not understand that the room type was a string and so required string delimiters (e.g. quotation marks) around the parameter.</p> <p>Where candidates defined local variables, assigned the values needed to the variables and then passed these into the function call were accepted.</p>
	c		<ul style="list-style-type: none"> For loop changed to include 0 total = 0 moved to before loop starts / removed 	<p>2 (AO3 2c)</p>	<p>Allow loop changed to 0 to 8 or 0 to 9 (Python)</p> <p>Do not accept moving total outside loop, NE (could be moved to after loop which would still be a logic error). Do not accept move to top of loop.</p> <p>Accept corrected code shown.</p> <p>Accept reference to count variable limits for BP1.</p> <p><u>Examiner's Comments</u></p> <p>This question acted as an excellent discriminator. The majority of candidates identified and fixed the problem with the array (starting at 0 rather than the 1 given). Fewer candidates were able to identify and fix the issue relating to the total being set to zero for each iteration.</p> <p>A number of candidates identified this second issue but were either not precise enough with their response (suggesting that the line could be moved but not stating where it should be moved to) or did not attempt to go beyond the simple identification.</p> <p>Candidates should be clear that this question required detail of how the program could be refined to fix problems, not just to identify problem(s).</p>

	d	<ul style="list-style-type: none"> Inputs hours AND electric (two separate inputs), storing or using these. Checks if car is electric (IF/Select statement)... ... correctly calculates and outputs price (hours * 2 / price / 2) for electric ... correctly calculates and outputs price (hours * 4 / electric price * 2) for non-electric Attempt at repetition of BP1 to 4... ...until 0 hours entered 	<p>6 (AO3 2c)</p>	<p>Initialisation of price and hours not necessary, but if present hours must be non-zero for BP6 to be given.</p> <p>BP5 must include all points attempted. Can still be credited if any of BP1 to 4 not attempted / incorrect.</p> <p>BP6 can be given as FT even if BP5 (loop) is in the wrong place / does not include all required code</p> <p>BP6 could be achieved as repeated function calls / recursion</p> <p>Initial input outside of loop that is then <u>also</u> included within loop is fine. For example, input of hours outside of loop but input is then repeated again at end of loop.</p> <p>Do not accept <code>while hours > 0</code> (could be -1)</p> <p>Do not penalise answers where 0 is output when loop exits</p> <p>e.g.</p> <pre>while hours != 0 hours = input("Enter hours") electric = input("enter Y for electric or N") if electric == "Y" then price = hours * 2 elseif electric == "N" then price = hours * 4 endif print(price) endwhile</pre> <p><u>Examiner's Comments</u></p> <p>This question was relatively well answered by candidates.</p> <p>Candidates were generally able to create suitable high-level program code to calculate and output the total price based on the information given.</p>
--	---	---	---------------------------	---

					<p>Many candidates ignored the requirement to repeat until 0 was entered; in this case, 4 out of the 6 marks were still available.</p> <p>To achieve marks for iteration it needed to both repeat the correct parts of the program and correctly terminate as per the requirements given.</p> <p>A typical mistake was to repeatedly calculate the price but not ask afresh for new inputs.</p>															
			Total	23																
17	a		<ul style="list-style-type: none">To convert it to binary/machine codeThe processor can only understand machine code	1 (AO1 1a)	Maximum 1 mark															
	b		<ul style="list-style-type: none">Compiler translates all the code in one go......whereas an interpreter translates one line at a timeCompiler creates an executable......whereas an interpreter does not/executes one line at a timeCompiler reports errors at the end......whereas an interpreter stops when it finds an error	4 (AO1 1b)	1 mark to be awarded for the correct identification and one for a valid description up to a maximum of 4 marks. No more than 2 marks for answers relating only to interpreters and no more than 2 marks for answers only relating to compilers.															
			Total	5																
18	a		<table border="1"><thead><tr><th>A</th><th>B</th><th>P</th></tr></thead><tbody><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td>1</td></tr><tr><td></td><td></td><td>1</td></tr><tr><td></td><td></td><td></td></tr></tbody></table>	A	B	P						1			1				2 (AO1 1b)	1 mark for each correct answer in table 'True' or 'T' are also credit worthy.
A	B	P																		
		1																		
		1																		
	b			1 (AO1 1b)	Correct Answer Only															

			Total	3	
19	a		<ul style="list-style-type: none"> • if • <u>num2</u> • print (<u>num1</u>) • print (<u>num2</u>) 	4 (AO3 2b)	Allow equivalent pseudocode expressions Variables must not have speech marks around them
	b		<ul style="list-style-type: none"> • use of condition controlled loop (while or do/until)... • ...checking condition of number larger than or equal to 0 • Input number from user within loop (FT if no loop) • multiply number input by 2... •output value in number 	5 (AO3 2b)	e.g. 1 store 10 in number while number is greater than or equal to 0 do the following: Take input from the user, store in number Multiply number by 2 Output number e.g. 2 while number >= 0 number = input() output(number * 2) Ignore non-initialisation of value used in condition for loop.
			Total	9	
20	a		<ul style="list-style-type: none"> • RebEl 	1 (AO2 1b)	Correct Answer Only (allow any case)
	b	i	<ul style="list-style-type: none"> • uitFr 	1 (AO2 1b)	Correct Answer Only (allow any case)
		ii	<ul style="list-style-type: none"> • Taking firstname, surname and teacher or student as input • Checking IF role is teacher/student (using appropriate selection) • For teacher ...Generating last 3 letters of surname using appropriate string manipulation • ...Generating first 2 of letters of firstname and adding to previous • For student.... correctly calculating as before • Correct concatenation and output <p>e.g. Ask the user to input the data, store in variables firstname, surname and role. Check whether the role entered is</p>	6 (AO3 2b)	1 mark for each correct bullet to a maximum of 6. If used, a flowchart should represent the bulleted steps in the answer column.

			teacher. If it is, join the right 3 most letters in surname with the left 2 letters in firstname. Store this in username. If it is not teacher, join the left 3 letters from firstname with the left 2 letters from surname. Store this in username. Output the value in username.		
			Total	8	
21		i	<p>1 mark per bullet, max 4</p> <ul style="list-style-type: none"> • Compare 5 (middle value) to 7 • 5 is smaller than 7 / 7 is larger than 7 so... • discard lower part of list / repeat with upper part of list • ...compare 7 to 7 (item found) 	4	Do not accept generic answers that do not refer to the data given.
		ii	<p>1 mark per bullet, max 2</p> <ul style="list-style-type: none"> • List of size 1 to compare • ...and item not matched to search term 	2	Do not accept answers relating to "end of list" – this is linear search.
		iii	<ul style="list-style-type: none"> • More efficient / Less time taken (to find item) / fewer comparisons to make (with large lists) 	1	Accept reference to big O notation as equivalent to more efficient.
			Total	7	
22			<ul style="list-style-type: none"> • Logically compares A AND / correct nested IF • ...B OR C / correct sequential IF • Output in both cases (with attempt at selection). 	3	<pre>A = input("Is the customer 15 or over?") B = input("Does the customer have a ticket?") C = input("Does the customer money to buy a ticket") if A AND (B OR C) then print ("allowed") else print ("not allowed") endif</pre> <p>Accept answers where inputs are given as strings e.g :</p> <pre>if A == "Yes" AND (B == "Yes" OR C == "Yes") then print ("allowed") else print ("not allowed") endif</pre>

			Total	3													
23	a	i	<ul style="list-style-type: none">Hiding / ignoring / removing detail / focussing on certain parts of a problem	1													
		ii	<ul style="list-style-type: none">Focus on age / number of milesIgnore other factors (such as make, model, etc)	1	Allow other examples of factors to ignore / remove for BP2												
		iii	<ul style="list-style-type: none">Ensures only certain users can access the systemUsing password / other example of authentication technique	2	Allow other examples of authentication for BP2												
	b	i	<p>1 mark per bullet, max 4</p> <ul style="list-style-type: none">Miles and age input <u>separately</u>Checks for valid mileageChecks for valid ageChecks <u>both</u> are greater than / greater than equal to zero...correctly outputs both True and False	5	<p>BP2 and 3 must check for both ends of range – must check that input data is not negative.</p> <p>Allow FT for BP4 if already penalised under BP2 and/or 3 and output is otherwise correct.</p> <p>e.g.</p> <pre>miles = input("enter miles driven") age = input("enter age of car") valid = True if miles > 10000 or miles < 0 then valid = False elseif age > 5 or age < 0 then valid = False endif print(valid)</pre>												
		ii	<p>1 mark per row, max 3</p> <ul style="list-style-type: none">Normal : miles (0 – 9,999), age (0 - 5)Erroneous: miles (less than 0, larger than 9,999), age (less than 0 / more than 5) / non-numeric dataBoundary : miles (-1/0 / 9,999 / 10,000), age (-1/0 / 5/6)	3	<p>Specific data must be given, not a description</p> <p>e.g.</p> <table><tr><th></th><th>Miles</th><th>Age</th></tr><tr><td>Normal</td><td>7,000</td><td>3</td></tr><tr><td>Erroneous</td><td>12,000</td><td>7</td></tr><tr><td>Boundary</td><td>10,000</td><td>5</td></tr></table>		Miles	Age	Normal	7,000	3	Erroneous	12,000	7	Boundary	10,000	5
	Miles	Age															
Normal	7,000	3															
Erroneous	12,000	7															
Boundary	10,000	5															

		iii	<ul style="list-style-type: none">During development / whilst writing the program / before development is complete.	1																			
	c		<p>1 mark per bullet, max 6</p> <ul style="list-style-type: none">Inputs the current battery charge percentageOutputs “full” if 100%Calculates the amount to chargeCalculates the time in minutes......converts to hours and minutesOutputs the time in hours and minutes	6	<p>Allow output of 0 hours 0 minutes if full.</p> <p>Allow answers referencing decimal parts (e.g. 0.8 = 80%)</p> <p>BP5 can be attempted in many ways (e.g. DIV and MOD, repeated division, etc)</p> <p>Allow FT for BP6 if reasonable attempt at conversion for BP5 has been given.</p> <p>e.g.</p> <pre>charge = input("enter battery charge") if charge == 100 then print("full") else time = (100-charge) * 10 hours = time DIV 60 mins = time MOD 60 print (hours, mins) endif</pre>																		
			Total	19																			
24	a	i	<p>One mark if two correct, two marks if four correct, three marks if all correct.</p> <table><thead><tr><th>Price input</th><th>Test type</th><th>Expected price output</th></tr></thead><tbody><tr><td>50</td><td>Normal</td><td>50</td></tr><tr><td>100</td><td>Boundary</td><td>100</td></tr><tr><td>150</td><td>Normal</td><td>130</td></tr><tr><td>200</td><td>Boundary</td><td>180</td></tr><tr><td>250</td><td>Normal</td><td>210</td></tr></tbody></table>	Price input	Test type	Expected price output	50	Normal	50	100	Boundary	100	150	Normal	130	200	Boundary	180	250	Normal	210	3	
Price input	Test type	Expected price output																					
50	Normal	50																					
100	Boundary	100																					
150	Normal	130																					
200	Boundary	180																					
250	Normal	210																					
		ii	<p>One mark per bullet point</p> <ul style="list-style-type: none">Input and store priceCheck if price is > 200......if true, reduce price by 40Check if price is >100 <u>and not >200</u>......if true, reduce price by 20	6	<p><u>High-level programming language / OCR Exam Reference Language response required</u></p> <p>Do not accept pseudocode / natural language.</p> <p>BP3 and BP5 only to be given if</p>																		

			<ul style="list-style-type: none"> Output price 		<p>sensible check for price being over the appropriate threshold. BP4 must check that price is both larger than 100 and not larger than 200; do not give mark for simply checking price is larger than 100. This may be implicit (e.g. using elseif).</p> <p>e.g.</p> <pre>price = input("enter price") if price > 200 then price = price - 40 elseif price > 100 then price = price - 20 endif print(price)</pre>
	b		<p>One mark per bullet point</p> <ul style="list-style-type: none"> checking both values (e.g. or changed to and if appropriate) if statement in correct format (e.g. checking against stocklevel for each condition) if statement uses correct comparisons (e.g. >= and <=) print statements in correct position print statements include string delimiters (e.g. speech marks) around both string outputs 	5	<p><u>High-level programming language / OCR Exam Reference Language response required</u></p> <p>Do not accept pseudocode / natural language.</p> <p>e.g.</p> <pre>stocklevel = input("Enter stock level") if stocklevel >= 5 and stocklevel <= 25 then print("In demand") else print("Not in demand") endif</pre> <p>alternative example</p> <pre>stocklevel = input("Enter stock level") if stocklevel > 5 or stocklevel > 25 then print("Not in demand") else print("In demand") endif</pre> <p>As a matter of principle, a candidate who refines the program to work fully but in a different format to that specified should gain full marks.</p>
			Total	14	
25	a		1 mark per bullet, max 4	4	D, F may be swapped around.

			<ul style="list-style-type: none"> • C • A • D/F • F/D 		e.g. <pre> graph TD A[Mobile phone app] --> B[Login] A --> C[Manage appointments] A --> D[C] C --> E[A] C --> F[View appointments] F --> G[D/F] F --> H[F/D] </pre>																														
	b	i	<ul style="list-style-type: none"> • An error that does not cause the program to crash / produces unexpected output 	1																															
		ii	1 mark per bullet, max 4 <ul style="list-style-type: none"> • Line 02 / empty = 0 • Will reset empty to 0 on each iteration of the loop • Line 07 / print ("empty") • Will print out the string "empty" instead of the value held in the variable 	4	Mark in pairs																														
			Total	9																															
26	a		<table border="1"> <tr><td>crime</td><td>bait</td><td>fright</td><td>victory</td><td>nibble</td><td>loose</td></tr> <tr><td>bait</td><td>crime</td><td>fright</td><td>victory</td><td>nibble</td><td>loose</td></tr> <tr><td>bait</td><td>crime</td><td>fright</td><td>nibble</td><td>victory</td><td>loose</td></tr> <tr><td>bait</td><td>crime</td><td>fright</td><td>nibble</td><td>loose</td><td>victory</td></tr> <tr><td>bait</td><td>crime</td><td>fright</td><td>loose</td><td>nibble</td><td>victory</td></tr> </table>	crime	bait	fright	victory	nibble	loose	bait	crime	fright	victory	nibble	loose	bait	crime	fright	nibble	victory	loose	bait	crime	fright	nibble	loose	victory	bait	crime	fright	loose	nibble	victory	4 (AO2 1b)	1 mark for each row from rows 2–5. Allow multiple swaps in one stage, where it is clear that a bubble sort has been applied.
crime	bait	fright	victory	nibble	loose																														
bait	crime	fright	victory	nibble	loose																														
bait	crime	fright	nibble	victory	loose																														
bait	crime	fright	nibble	loose	victory																														
bait	crime	fright	loose	nibble	victory																														
	b		<ul style="list-style-type: none"> • Comparing zebra to orange • Greater, so split and take right side • Further comparison (1 or 2 depending on choices made) • Correct identification of zebra using methodology above e.g. compare zebra to orange greater, split right compare to wind greater, split right	4 (AO2 1b)	1 mark per bullet (multiple ways through, marks awarded for appropriate comparison and creation of sub groups).																														

			compare to zebra																															
			Total	8																														
27	a	i	<ul style="list-style-type: none">• or• >300 / >= 301• print	3 (AO3 2b)	<p><u>High-level programming language / OCR Exam Reference Language response required</u></p> <p>Do not accept pseudocode / natural English.</p> <p>MP2 do not accept 'greater than', must use the HLL syntax > or >=</p> <p>MP3 must be a suitable output command word that could be found in a HLL e.g. print (Python), console.writeline (VB), cout (C++)</p>																													
		ii	<ul style="list-style-type: none">• Suitable invalid test data (i.e. > 300, e.g. 350)• Suitable boundary test data (e.g. 0, 300)• "Value accepted" or equivalent if boundary data 0 or 300 / "Invalid input displayed" or equivalent if boundary data -1 or 301	3 (AO3 2c)																														
	b		<table><tr><td></td><td>x</td><td>y</td><td>output</td></tr><tr><td>MP1</td><td>15</td><td>0</td><td></td></tr><tr><td rowspan="2">MP2</td><td>14</td><td>1</td><td></td></tr><tr><td>12</td><td>2</td><td></td></tr><tr><td rowspan="3">MP3</td><td>9</td><td>3</td><td></td></tr><tr><td>5</td><td>4</td><td></td></tr><tr><td>0</td><td>5</td><td></td></tr><tr><td>MP4</td><td></td><td></td><td>5</td></tr></table>		x	y	output	MP1	15	0		MP2	14	1		12	2		MP3	9	3		5	4		0	5		MP4			5	4 (AO3 2c)	<p>one mark for first row</p> <p>one mark for row 2 and 3</p> <p>one mark for rows 4, 5, and 6</p> <p>one mark for the correct output (the only value in the output column, in any position)</p>
	x	y	output																															
MP1	15	0																																
MP2	14	1																																
	12	2																																
MP3	9	3																																
	5	4																																
	0	5																																
MP4			5																															
	c		<p>1 mark per bullet</p> <ul style="list-style-type: none">• Test data either 0 or less characters, or 20 or more characters• Stating correct output• Test data between 1 and 19 characters (inc)• Stating correct output	4 (AO3 2c)	<p>Mark test data first, both must meet different criteria. Then mark output for each.</p>																													
	d	i	Input	2 (AO3 2a)	Enter text here.																													

			<ul style="list-style-type: none"> Number of hours <u>and</u> minutes <p>Output</p> <ul style="list-style-type: none"> Number of minutes 		
		ii	<ul style="list-style-type: none"> Program calls function correctly using hours and minutes variables Parameters used appropriately Calculation is computed accurately Final total is returned suitably 	<p>4 (AO3 2a)</p>	<pre>hours = input("Please enter number of hours played") minutes = input("Please enter number of minutes played") finalTotal = totalMins(hours, minutes) print (finalTotal) function totalMins(hours,minutes) total = hours + mins * 60 return total endfunction</pre> <ol style="list-style-type: none"> Parameters named in function must be used within the function itself Does not matter if function uses different names to those declared in main program Return must be included with the correct local variable for total
		iii	<ul style="list-style-type: none"> Takes input from the user Compares if input is larger than 120... ...if true, outputs "You played games for too long!" ...if false, outputs "You are under your time limit!" 	<p>4 (AO3 2b)</p>	<p><u>High-level programming language / OCR Exam Reference Language response required</u></p> <p>Do not accept pseudocode / natural English.</p> <p>Example algorithm given below</p> <pre>minutes = input("Enter minutes played") if minutes > 120</pre>

				<pre>print "You played games for too long!" else print "You are under your time limit!" endif</pre> <p>Accept alternative (but suitable) output messages.</p> <p>Accept logical comparison of input less than or equal to 120 and appropriate True/False statements.</p>
			Total	24