

Cambridge O Level

COMPUTER SCIENCE**2210/21**

Paper 2 Algorithms, Programming and Logic

May/June 2024**MARK SCHEME**

Maximum Mark: 75

Published

This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, which would have considered the acceptability of alternative answers.

Mark schemes should be read in conjunction with the question paper and the Principal Examiner Report for Teachers.

Cambridge International will not enter into discussions about these mark schemes.

Cambridge International is publishing the mark schemes for the May/June 2024 series for most Cambridge IGCSE, Cambridge International A and AS Level and Cambridge Pre-U components, and some Cambridge O Level components.

This document consists of **18** printed pages.

These general marking principles must be applied by all examiners when marking candidate answers. They should be applied alongside the specific content of the mark scheme or generic level descriptions for a question. Each question paper and mark scheme will also comply with these marking principles.

GENERIC MARKING PRINCIPLE 1:

Marks must be awarded in line with:

- the specific content of the mark scheme or the generic level descriptors for the question
- the specific skills defined in the mark scheme or in the generic level descriptors for the question
- the standard of response required by a candidate as exemplified by the standardisation scripts.

GENERIC MARKING PRINCIPLE 2:

Marks awarded are always **whole marks** (not half marks, or other fractions).

GENERIC MARKING PRINCIPLE 3:

Marks must be awarded **positively**:

- marks are awarded for correct/valid answers, as defined in the mark scheme. However, credit is given for valid answers which go beyond the scope of the syllabus and mark scheme, referring to your Team Leader as appropriate
- marks are awarded when candidates clearly demonstrate what they know and can do
- marks are not deducted for errors
- marks are not deducted for omissions
- answers should only be judged on the quality of spelling, punctuation and grammar when these features are specifically assessed by the question as indicated by the mark scheme. The meaning, however, should be unambiguous.

GENERIC MARKING PRINCIPLE 4:

Rules must be applied consistently, e.g. in situations where candidates have not followed instructions or in the application of generic level descriptors.

GENERIC MARKING PRINCIPLE 5:

Marks should be awarded using the full range of marks defined in the mark scheme for the question (however; the use of the full mark range may be limited according to the quality of the candidate responses seen).

GENERIC MARKING PRINCIPLE 6:

Marks awarded are based solely on the requirements as defined in the mark scheme. Marks should not be awarded with grade thresholds or grade descriptors in mind.

Mark scheme abbreviations

/ separates alternative words / phrases within a marking point

// separates alternative answers within a marking point

underline actual word given must be used by candidate (grammatical variants accepted)

max indicates the maximum number of marks that can be awarded

() the word / phrase in brackets is not required, but sets the context

Note: No marks are awarded for using brand names of software packages or hardware.

Question	Answer	Marks
1	D	1

Question	Answer	Marks												
2(a)	<p>One mark for each correct line</p> <table> <thead> <tr> <th>Description</th> <th>Programming concept</th> </tr> </thead> <tbody> <tr> <td>a subroutine that may not return a value</td> <td>function</td> </tr> <tr> <td>a value that is declared and used within a specific procedure</td> <td>procedure</td> </tr> <tr> <td>a value that a procedure expects you to supply when it is called</td> <td>parameter</td> </tr> <tr> <td>a subroutine that will always return a value</td> <td>global variable</td> </tr> <tr> <td></td> <td>local variable</td> </tr> </tbody> </table>	Description	Programming concept	a subroutine that may not return a value	function	a value that is declared and used within a specific procedure	procedure	a value that a procedure expects you to supply when it is called	parameter	a subroutine that will always return a value	global variable		local variable	4
Description	Programming concept													
a subroutine that may not return a value	function													
a value that is declared and used within a specific procedure	procedure													
a value that a procedure expects you to supply when it is called	parameter													
a subroutine that will always return a value	global variable													
	local variable													
2(b)	<p>One mark per mark point</p> <ul style="list-style-type: none"> • Correct key word - CALL • Procedure name with correct parameters - Average (25, 50) <p>CALL Average(25, 50)</p>	2												

Question	Answer	Marks
2(c)	<p>One mark per mark point, max three</p> <ul style="list-style-type: none"> • Procedures / functions divide the program into smaller manageable segments • ... making it more readable / easier to understand / easier to debug • Procedures / functions with meaningful names help to provide documentation for the program / enable/help/provide abstraction • Procedures and functions may be re-used (in the program / in other programs / as part of a library) / run from a single line • Procedures and functions can reduce / eliminate (repeated) code 	3

Question	Answer	Marks
3	<p>One mark for a correct statement about each data type one mark for a correct example of data for each data type.</p> <p>Example answer</p> <p>Integer A whole number [1] Example 27 [1]</p> <p>Real A number that contains a fractional part [1] ... Example 18.75 [1]</p>	4

Question	Answer	Marks
4(a)	<p>One mark per mark point</p> <ul style="list-style-type: none"> • Line 01 / DECLARE Loop : STRING should be DECLARE Loop : INTEGER • Line 07 / IF Loop \leftarrow 1 TO Limit should be FOR Loop \leftarrow 1 TO Limit • Line 09 / INPUT Loop should be INPUT Value • Line 10 / Total \leftarrow Total * Value should be Total \leftarrow Total + Value <p>Correct algorithm:</p> <pre> 01 DECLARE Loop : INTEGER 02 DECLARE Limit : INTEGER 03 DECLARE Value : REAL 04 DECLARE Total : REAL 05 Total \leftarrow 0 06 Limit \leftarrow ROUND(RANDOM() * 19,0) + 1 07 FOR Loop \leftarrow 1 TO Limit 08 OUTPUT "Enter a number" 09 INPUT Value 10 Total \leftarrow Total + Value 11 NEXT Loop 12 OUTPUT "The total of the numbers entered is ", Total 13 OUTPUT "The average number entered is ", Total / Limit </pre>	4

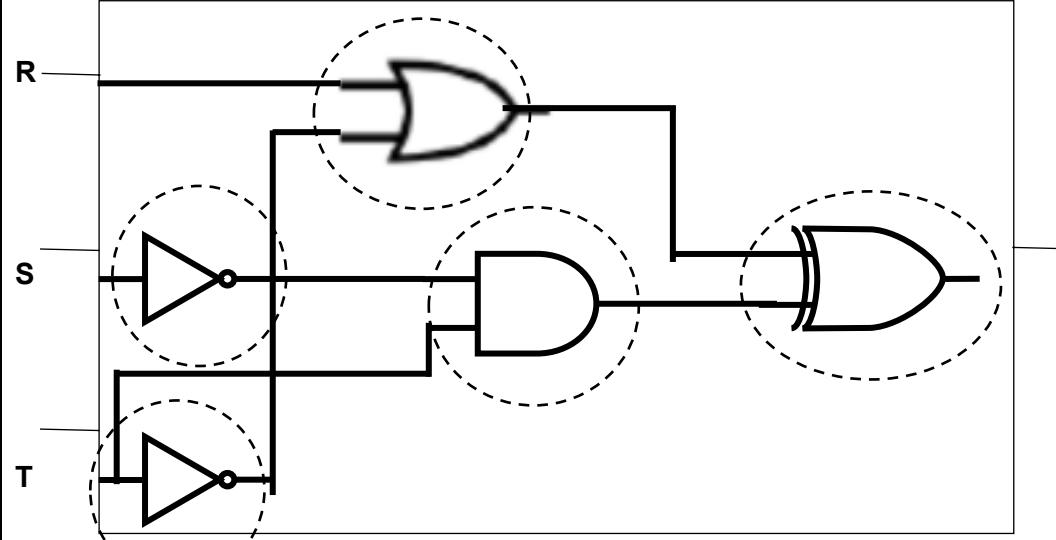
Question	Answer	Marks
4(b)	<p>One mark per mark point</p> <ul style="list-style-type: none"> Correct use of ROUND with 2 arguments separated by comma, for example ROUND(5, 2) (in statement 13) (Total / Limit,1) correct arguments <p>For example:</p> <pre>OUTPUT "The average of the numbers entered is ", ROUND(Total / Limit,1)</pre>	2
4(c)	<p>One mark per mark point, max four</p> <ul style="list-style-type: none"> After line 09 / after the input Insert a WHILE / pre-condition loop... ... to check if the value entered is between 1 and 500 inclusive If the value is not in range, output an error message ... and insert another input statement for re-input. <p>Or</p> <ul style="list-style-type: none"> Before line 08 / before the input message start a REPEAT / post-condition loop After line 09 / after the input ... close the REPEAT loop by checking if the value entered was between 1 and 500 inclusive If it wasn't, (the loop will repeat) for the number to be re-input / If it was, the program continues. 	4

Question	Answer	Marks
5	<p>One mark per mark point</p> <p>MP1 Correct input statement with appropriate variable MP2 Elements of selection statement present – CASE OF ENDCASE MP3 At least one correct branch in the case statement MP4 All branches from 1 to 4 correct MP5 Correct use of OTHERWISE with correct output.</p> <p>For example:</p> <pre>INPUT Number CASE OF Number 1 : OUTPUT Number 2 : OUTPUT Number 3 : OUTPUT Number 4 : OUTPUT Number OTHERWISE OUTPUT "ERROR" ENDCASE</pre> <p>Or</p> <pre>INPUT Number CASE OF Number 1 : OUTPUT 1 2 : OUTPUT 2 3 : OUTPUT 3 4 : OUTPUT 4 OTHERWISE OUTPUT "ERROR" ENDCASE</pre>	5

Question	Answer												Marks																																																																																																																																																																									
6(a)	<p>One mark per mark point</p> <p>MP1 Correct Limit column MP2 Correct Count column MP3 Array columns for input stage MP4 Array columns for sort stage MP5 Correct Flag column MP6 Correct Swap column MP7 Correct Result and OUTPUT columns</p> <table border="1" data-bbox="631 541 1275 1364"> <thead> <tr> <th colspan="12" data-bbox="631 541 1275 589">Numbers</th> </tr> <tr> <th data-bbox="339 597 473 660">Limit</th><th data-bbox="473 597 608 660">Count</th><th data-bbox="608 597 743 660">[1]</th><th data-bbox="743 597 878 660">[2]</th><th data-bbox="878 597 1012 660">[3]</th><th data-bbox="1012 597 1147 660">[4]</th><th data-bbox="1147 597 1282 660">[5]</th><th data-bbox="1282 597 1417 660">[6]</th><th data-bbox="1417 597 1551 660">[7]</th><th data-bbox="1551 597 1686 660">Flag</th><th data-bbox="1686 597 1821 660">Swap</th><th data-bbox="1821 597 1956 660">Result</th><th data-bbox="1956 597 2068 660">OUTPUT</th></tr> </thead> <tbody> <tr> <td data-bbox="339 660 473 724">7</td><td data-bbox="473 660 608 724">1</td><td data-bbox="608 660 743 724">47</td><td data-bbox="743 660 878 724"></td><td data-bbox="878 660 1012 724"></td><td data-bbox="1012 660 1147 724"></td><td data-bbox="1147 660 1282 724"></td><td data-bbox="1282 660 1417 724"></td><td data-bbox="1417 660 1551 724"></td><td data-bbox="1551 660 1686 724"></td><td data-bbox="1686 660 1821 724"></td><td data-bbox="1821 660 1956 724"></td><td data-bbox="1956 660 2068 724"></td></tr> <tr> <td data-bbox="339 724 473 787"></td><td data-bbox="473 724 608 787">2</td><td data-bbox="608 724 743 787"></td><td data-bbox="743 724 878 787">50</td><td data-bbox="878 724 1012 787"></td><td data-bbox="1012 724 1147 787"></td><td data-bbox="1147 724 1282 787"></td><td data-bbox="1282 724 1417 787"></td><td data-bbox="1417 724 1551 787"></td><td data-bbox="1551 724 1686 787"></td><td data-bbox="1686 724 1821 787"></td><td data-bbox="1821 724 1956 787"></td><td data-bbox="1956 724 2068 787"></td></tr> <tr> <td data-bbox="339 787 473 851"></td><td data-bbox="473 787 608 851">3</td><td data-bbox="608 787 743 851"></td><td data-bbox="743 787 878 851"></td><td data-bbox="878 787 1012 851">52</td><td data-bbox="1012 787 1147 851"></td><td data-bbox="1147 787 1282 851"></td><td data-bbox="1282 787 1417 851"></td><td data-bbox="1417 787 1551 851"></td><td data-bbox="1551 787 1686 851"></td><td data-bbox="1686 787 1821 851"></td><td data-bbox="1821 787 1956 851"></td><td data-bbox="1956 787 2068 851"></td></tr> <tr> <td data-bbox="339 851 473 914"></td><td data-bbox="473 851 608 914">4</td><td data-bbox="608 851 743 914"></td><td data-bbox="743 851 878 914"></td><td data-bbox="878 851 1012 914"></td><td data-bbox="1012 851 1147 914">60</td><td data-bbox="1147 851 1282 914"></td><td data-bbox="1282 851 1417 914"></td><td data-bbox="1417 851 1551 914"></td><td data-bbox="1551 851 1686 914"></td><td data-bbox="1686 851 1821 914"></td><td data-bbox="1821 851 1956 914"></td><td data-bbox="1956 851 2068 914"></td></tr> <tr> <td data-bbox="339 914 473 978"></td><td data-bbox="473 914 608 978">5</td><td data-bbox="608 914 743 978"></td><td data-bbox="743 914 878 978"></td><td data-bbox="878 914 1012 978"></td><td data-bbox="1012 914 1147 978"></td><td data-bbox="1147 914 1282 978">80</td><td data-bbox="1282 914 1417 978"></td><td data-bbox="1417 914 1551 978"></td><td data-bbox="1551 914 1686 978"></td><td data-bbox="1686 914 1821 978"></td><td data-bbox="1821 914 1956 978"></td><td data-bbox="1956 914 2068 978"></td></tr> <tr> <td data-bbox="339 978 473 1041"></td><td data-bbox="473 978 608 1041">6</td><td data-bbox="608 978 743 1041"></td><td data-bbox="743 978 878 1041"></td><td data-bbox="878 978 1012 1041"></td><td data-bbox="1012 978 1147 1041"></td><td data-bbox="1147 978 1282 1041">63</td><td data-bbox="1282 978 1417 1041"></td><td data-bbox="1417 978 1551 1041"></td><td data-bbox="1551 978 1686 1041"></td><td data-bbox="1686 978 1821 1041"></td><td data-bbox="1821 978 1956 1041"></td><td data-bbox="1956 978 2068 1041"></td></tr> <tr> <td data-bbox="339 1041 473 1105"></td><td data-bbox="473 1041 608 1105">7</td><td data-bbox="608 1041 743 1105"></td><td data-bbox="743 1041 878 1105"></td><td data-bbox="878 1041 1012 1105"></td><td data-bbox="1012 1041 1147 1105"></td><td data-bbox="1147 1041 1282 1105"></td><td data-bbox="1282 1041 1417 1105">70</td><td data-bbox="1417 1041 1551 1105"></td><td data-bbox="1551 1041 1686 1105"></td><td data-bbox="1686 1041 1821 1105"></td><td data-bbox="1821 1041 1956 1105"></td><td data-bbox="1956 1041 2068 1105"></td></tr> <tr> <td data-bbox="339 1105 473 1168"></td><td data-bbox="473 1105 608 1168">8</td><td data-bbox="608 1105 743 1168"></td><td data-bbox="743 1105 878 1168"></td><td data-bbox="878 1105 1012 1168"></td><td data-bbox="1012 1105 1147 1168"></td><td data-bbox="1147 1105 1282 1168"></td><td data-bbox="1282 1105 1417 1168"></td><td data-bbox="1417 1105 1551 1168">TRUE</td><td data-bbox="1551 1105 1686 1168"></td><td data-bbox="1686 1105 1821 1168"></td><td data-bbox="1821 1105 1956 1168"></td><td data-bbox="1956 1105 2068 1168"></td></tr> <tr> <td data-bbox="339 1168 473 1232"></td><td data-bbox="473 1168 608 1232">1</td><td data-bbox="608 1168 743 1232"></td><td data-bbox="743 1168 878 1232"></td><td data-bbox="878 1168 1012 1232"></td><td data-bbox="1012 1168 1147 1232"></td><td data-bbox="1147 1168 1282 1232"></td><td data-bbox="1282 1168 1417 1232"></td><td data-bbox="1417 1168 1551 1232">FALSE</td><td data-bbox="1551 1168 1686 1232"></td><td data-bbox="1686 1168 1821 1232"></td><td data-bbox="1821 1168 1956 1232"></td><td data-bbox="1956 1168 2068 1232"></td></tr> <tr> <td data-bbox="339 1232 473 1295"></td><td data-bbox="473 1232 608 1295">2</td><td data-bbox="608 1232 743 1295"></td><td data-bbox="743 1232 878 1295"></td><td data-bbox="878 1232 1012 1295"></td><td data-bbox="1012 1232 1147 1295"></td><td data-bbox="1147 1232 1282 1295"></td><td data-bbox="1282 1232 1417 1295"></td><td data-bbox="1417 1232 1551 1295"></td><td data-bbox="1551 1232 1686 1295"></td><td data-bbox="1686 1232 1821 1295"></td><td data-bbox="1821 1232 1956 1295"></td><td data-bbox="1956 1232 2068 1295"></td></tr> <tr> <td data-bbox="339 1295 473 1359"></td><td data-bbox="473 1295 608 1359">3</td><td data-bbox="608 1295 743 1359"></td><td data-bbox="743 1295 878 1359"></td><td data-bbox="878 1295 1012 1359"></td><td data-bbox="1012 1295 1147 1359"></td><td data-bbox="1147 1295 1282 1359"></td><td data-bbox="1282 1295 1417 1359"></td><td data-bbox="1417 1295 1551 1359"></td><td data-bbox="1551 1295 1686 1359"></td><td data-bbox="1686 1295 1821 1359"></td><td data-bbox="1821 1295 1956 1359"></td><td data-bbox="1956 1295 2068 1359"></td></tr> <tr> <td data-bbox="339 1359 473 1422"></td><td data-bbox="473 1359 608 1422">4</td><td data-bbox="608 1359 743 1422"></td><td data-bbox="743 1359 878 1422"></td><td data-bbox="878 1359 1012 1422"></td><td data-bbox="1012 1359 1147 1422"></td><td data-bbox="1147 1359 1282 1422"></td><td data-bbox="1282 1359 1417 1422"></td><td data-bbox="1417 1359 1551 1422"></td><td data-bbox="1551 1359 1686 1422"></td><td data-bbox="1686 1359 1821 1422"></td><td data-bbox="1821 1359 1956 1422"></td><td data-bbox="1956 1359 2068 1422"></td></tr> </tbody> </table>	Numbers												Limit	Count	[1]	[2]	[3]	[4]	[5]	[6]	[7]	Flag	Swap	Result	OUTPUT	7	1	47												2		50											3			52										4				60									5					80								6					63								7						70							8							TRUE						1							FALSE						2													3													4											
Numbers																																																																																																																																																																																						
Limit	Count	[1]	[2]	[3]	[4]	[5]	[6]	[7]	Flag	Swap	Result	OUTPUT																																																																																																																																																																										
7	1	47																																																																																																																																																																																				
	2		50																																																																																																																																																																																			
	3			52																																																																																																																																																																																		
	4				60																																																																																																																																																																																	
	5					80																																																																																																																																																																																
	6					63																																																																																																																																																																																
	7						70																																																																																																																																																																															
	8							TRUE																																																																																																																																																																														
	1							FALSE																																																																																																																																																																														
	2																																																																																																																																																																																					
	3																																																																																																																																																																																					
	4																																																																																																																																																																																					

Question	Answer												Marks		
6(a)			Numbers												
	Limit	Count	[1]	[2]	[3]	[4]	[5]	[6]	[7]	Flag	Swap	Result	OUTPUT		
		5									80				
							63	80		TRUE					
		6									80				
								70	80	TRUE					
		7													
		1								FALSE					
		2													
		3													
		4													
		5													
		6													
		7													
												4	60		
6(b)	<p>One mark per mark point</p> <ul style="list-style-type: none"> A set of numbers is input / stored in an array The numbers are sorted / Bubble sort into (ascending) order The middle / median value is found / output 	3													

Question	Answer	Marks
7(a)	<p>One mark per mark point, max two</p> <ul style="list-style-type: none"> • Data stored in a file gives a permanent copy / prevents the data from being lost • ... so it can be recalled / used again in a program / in another program • can be stored elsewhere / transferred to another computer 	2
7(b)	<p>One mark per mark point, max four</p> <p>MP1 Declaration of a string variable MP2 Correct use of <code>OPENFILE</code> keywords for reading MP3 Correct use <code>READFILE</code> with string variable MP4 Correct use of <code>UCASE</code> and <code>LENGTH</code> functions and output of each MP5 Correct use of <code>CLOSEFILE</code></p> <p>Example:</p> <pre>DECLARE Words : STRING OPENFILE Quotation.txt FOR READ READFILE Quotation.txt, Words OUTPUT UCASE(Words), LENGTH(Words) CLOSEFILE Quotation.txt</pre>	4

Question	Answer	Marks
8(a)	<p>One mark for each correct gate, with the correct input(s) as shown.</p>  <p>Z</p>	5

Question	Answer	Marks																																				
8(b)	<p>Four marks for eight correct outputs. Three marks for six or seven correct outputs. Two marks for four or five correct outputs. One mark for two or three correct outputs</p> <table border="1"><thead><tr><th>R</th><th>S</th><th>T</th><th>Z</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr></tbody></table>	R	S	T	Z	0	0	0	1	0	0	1	1	0	1	0	1	0	1	1	0	1	0	0	1	1	0	1	0	1	1	0	1	1	1	1	1	4
R	S	T	Z																																			
0	0	0	1																																			
0	0	1	1																																			
0	1	0	1																																			
0	1	1	0																																			
1	0	0	1																																			
1	0	1	0																																			
1	1	0	1																																			
1	1	1	1																																			

Question	Answer	Marks
9(a)	<p>One mark for an appropriate reason if no has been stated, for example: None of the fields are likely to contain unique data.</p>	1
9(b)	<p>One mark for each correct answer</p> <p>SUM NumberInStock SoftDrinks Container 'Can'</p> <p>Correct code:</p> <pre>SELECT SUM <u>NumberInStock</u> FROM SoftDrinks WHERE Container = 'Can';</pre>	5
10	<p>Data structures required:</p> <p>The names underlined must match those given in the scenario:</p> <p>Arrays or lists <u>Clubs[]</u>, <u>Statistics[]</u>, <u>Points[]</u>, <u>TieIndex[]</u></p> <p>Variables <u>Matches</u>, Won, Drawn, Lost, Counter, Maximum, TieCheck, OutLoop, Max, Count</p> <p>Requirements (techniques):</p> <p>R1 Input and store number of matches, cricket club names, number of matches won, drawn and lost. Validation of number of matches played and matches won, lost and drawn against number of matches played (with prompts, iteration, storing data in variables, validation, 1D and 2D arrays).</p> <p>R2 Calculate and store the number of points for each cricket club. Finding the index of the array with the highest score / highest score (calculation, finding the maximum, iteration).</p> <p>R3 Identifying cricket clubs with highest number of points and whether there is a tie for the top position. Output with appropriate messages (iteration, selection and output).</p>	15

Question	Answer	Marks
10	<p>Example 15-mark answer in pseudocode</p> <pre> // meaningful identifiers and appropriate data structures // similar variables grouped to save space as in HL languages DECLARE Clubs : ARRAY[1:12] OF STRING DECLARE Statistics : ARRAY[1:12, 1:3] OF INTEGER DECLARE Points : ARRAY[1 : 12] OF INTEGER DECLARE TieIndex : ARRAY[1 : 12] OF INTEGER DECLARE Matches, Won, Drawn, Lost : INTEGER DECLARE Counter, Maximum, TieCheck, OutLoop : INTEGER DECLARE Max, Count : INTEGER // input number of matches REPEAT OUTPUT "How many matches have been played (Maximum 22) ? " INPUT Matches UNTIL Matches <= 22 // input of data as a single loop - multiple loops for data entry // is also acceptable. FOR Counter ← 1 TO 12 OUTPUT "Enter the name of the cricket club" INPUT Clubs[Counter] // input of match results with validation REPEAT OUTPUT "Enter the number of matches won, drawn and lost for ", Clubs[Counter] INPUT Won, Drawn, Lost IF Won + Drawn + Lost <> Matches THEN OUTPUT "Your inputs must total ", Matches, " please try again" ENDIF UNTIL Matches = Won + Drawn + Lost Statistics[Counter, 1] ← Won Statistics[Counter, 2] ← Drawn Statistics[Counter, 3] ← Lost // calculating and storing points </pre>	

Question	Answer	Marks
10	<pre> Points[Counter] ← Statistics[Counter, 1] * 12 + Statistics[Counter, 2] * 5 NEXT Counter // finding the highest points Max ← -100 FOR Maximum ← 1 TO 12 IF Points[Maximum] > Max THEN Max ← Points[Maximum] ENDIF NEXT Maximum // finding the winner(s) Count ← 0 FOR TieCheck ← 1 TO 12 IF Points[TieCheck] = Max THEN Count ← Count + 1 TieIndex[Count] ← TieCheck ENDIF Next TieCheck // outputting the results FOR OutLoop ← 1 TO Count OUTPUT "Winning Club(s): ", Clubs[TieIndex[OutLoop]] OUTPUT "Number of wins: ", Statistics[TieIndex[OutLoop], 1] Next OutLoop OUTPUT "Winning Points: ", Max </pre>	

Marking Instructions in italics			
AO2: Apply knowledge and understanding of the principles and concepts of computer science to a given context, including the analysis and design of computational or programming problems			
0	1–3	4–6	7–9
No creditable response.	At least one programming technique has been used. <i>Any use of selection, iteration, counting, totalling, input and output.</i>	Some programming techniques used are appropriate to the problem. <i>More than one technique seen applied to the scenario, check the list of techniques needed.</i>	The range of programming techniques used is appropriate to the problem. <i>All criteria stated for the scenario have been covered by the use of appropriate programming techniques, check the list of techniques needed.</i>
	Some data has been stored but not appropriately. <i>Any use of variables or arrays or other language dependent data structures e.g. Python lists.</i>	Some of the data structures chosen are appropriate and store some of the data required. <i>More than one data structure used to store data required by the scenario.</i>	The data structures chosen are appropriate and store all the data required. <i>The data structures used store all the data required by the scenario.</i>

Marking Instructions in italics			
0	1–2	3–4	5–6
No creditable response.	Program seen without relevant comments.	Program seen with some relevant comment(s).	The program has been fully commented.
	Some identifier names used are appropriate. <i>Some of the data structures used have meaningful names.</i>	The majority of identifiers used are appropriately named. <i>Most of the data structures used have meaningful names.</i>	Suitable identifiers with names meaningful to their purpose have been used throughout. <i>All of the data structures used have meaningful names.</i>
	The solution is illogical.	The solution contains parts that may be illogical.	The program is in a logical order.
	The solution is inaccurate in many places. <i>Solution contains few lines of code with errors that attempt to perform a task given in the scenario.</i>	The solution contains parts that are inaccurate. <i>Solution contains lines of code with some errors that logically perform tasks given in the scenario. Ignore minor syntax errors.</i>	The solution is accurate. <i>Solution logically performs all the tasks given in the scenario. Ignore minor syntax errors.</i>
	The solution attempts at least one of the requirements. <i>Solution contains lines of code that attempt at least one task given in the scenario.</i>	The solution meets most of the requirements. <i>Solution contains lines of code that perform most tasks given in the scenario.</i>	The solution meets all the requirements given in the question. <i>Solution performs all the tasks given in the scenario.</i>