



# **Cambridge O Level**

---

## **COMPUTER SCIENCE**

**2210/21**

Paper 2 Algorithms, Programming and Logic

**May/June 2023**

### **MARK SCHEME**

Maximum Mark: 50

---

**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 2023 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 descriptors 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:**

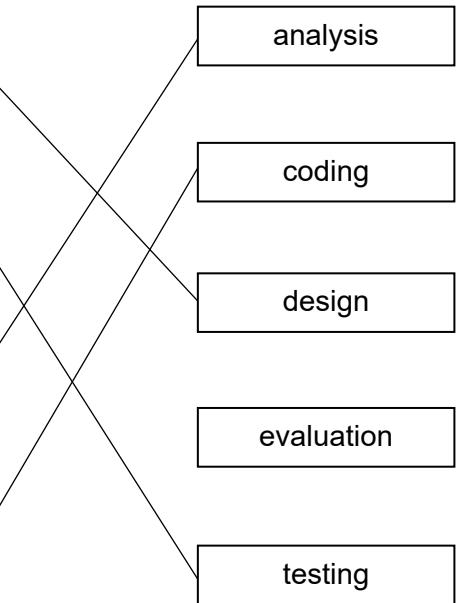
Rules must be applied consistently, e.g. in situations where candidates have not followed instructions or in the application of generic level descriptors.

**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.

Question	Answer	Marks
1(a)	<p>One mark for each correct line.</p> <p><b>Program development life cycle description</b></p> <ul style="list-style-type: none"> <li>develop an algorithm to solve the problem using structure diagrams, flowcharts or pseudocode</li> <li>detect and fix the errors in the program</li> <li>identify the problem and its requirements</li> <li>write and implement the instructions to solve the problem</li> </ul> <p><b>Program development life cycle stage</b></p> <ul style="list-style-type: none"> <li>analysis</li> <li>coding</li> <li>design</li> <li>evaluation</li> <li>testing</li> </ul> 	4
1(b)	<p>One mark for naming or describing each component part, max <b>three</b></p> <p><b>For example:</b></p> <p>inputs // what is put into the system    processes // actions taken to achieve a result    outputs // what is taken out of the system    storage // what needs to be kept for future use</p>	3

Question	Answer	Marks
2	A	1

Question	Answer	Marks
3(a)	<p><b>One</b> mark per mark point, max <b>two</b></p> <ul style="list-style-type: none"> <li>Validation is an automated check carried out by a computer</li> <li>... to make sure the data entered is sensible/acceptable/reasonable</li> </ul>	2
3(b)	<p><b>One</b> mark for each appropriate test data, max <b>three</b>  <b>One</b> mark for each correct accompanying reason, max <b>three</b></p> <p><b>For example:</b></p> <p>Normal – 75  Reason – the data lies within the required range <b>and</b> should be accepted</p> <p>Abnormal – Sixty  Reason – this is the wrong data type <b>and</b> should be rejected</p> <p>Extreme – 200  Reason – the highest value in the required range that should be accepted</p>	6

Question	Answer	Marks
4	<p><b>One</b> mark per mark point, max <b>four</b></p> <p><b>DIV</b>, max <b>two</b></p> <ul style="list-style-type: none"><li>• To perform integer division</li><li>• Meaning only the whole number part of the answer is retained</li><li>• Example of <b>DIV</b> For example <b>DIV(9, 4)</b> = 2</li></ul> <p><b>ROUND</b>, max <b>two</b></p> <ul style="list-style-type: none"><li>• To return a value rounded to a specified number of digits / decimal places</li><li>• The result will either be rounded to the next highest or the next lowest value</li><li>• ... depending on whether the value of the preceding digit is <math>\geq 5</math> or <math>&lt; 5</math></li><li>• Example of <b>ROUND</b> for example, <b>ROUND(4.56, 1)</b> = 4.6</li></ul>	<b>4</b>

Question	Answer	Marks
5(a)	<p><b>One mark per mark point, max four</b></p> <ul style="list-style-type: none"> <li>Line 04 / IF Number &lt; 0 should be IF Number &gt; 0</li> <li>Line 10 / Exit ← 1 // Line 01/ Exit ← 1 <b>and</b> Line 02 / WHILE Exit &lt;&gt; 0 should be Exit ← 0 // should be Exit ← 0 <b>and</b> WHILE Exit = 0</li> <li>Line 13 / ENDIF should be ENDWHILE</li> <li>Line 14 / OUTPUT "The total value of your numbers is ", Number should be OUTPUT "The total value of your numbers is ", Total</li> </ul> <p><b>Correct algorithm:</b></p> <pre> 01 Exit ← 1 02 WHILE Exit &lt;&gt; 0 DO 03     INPUT Number 04     IF Number &gt; 0 05         THEN 06             Total ← Total + Number 07         ELSE 08             IF Number = 0 09                 THEN 10                     Exit ← 0 11                 ENDIF 12             ENDIF 13 ENDWHILE 14 OUTPUT "The total value of your numbers is ", Total </pre>	4

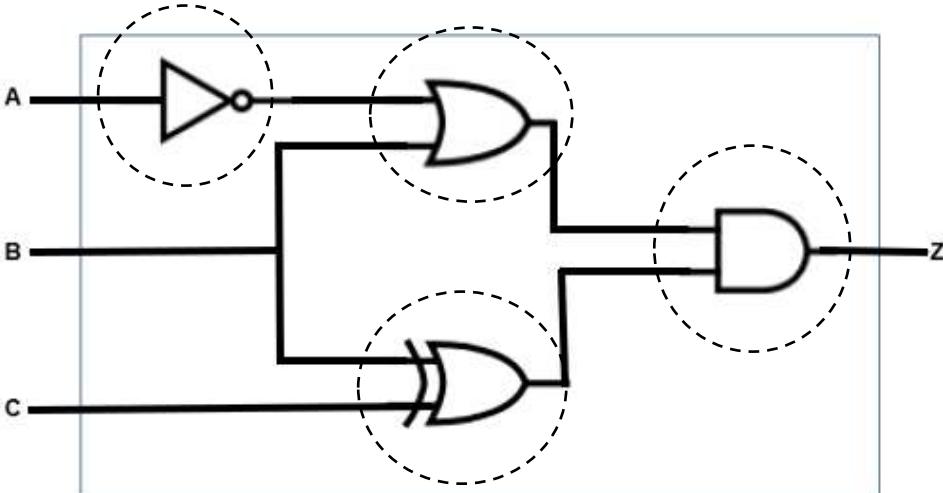
Question	Answer	Marks
5(b)	<p><b>One</b> mark per mark point, max <b>four</b></p> <ul style="list-style-type: none"> <li>• Initialise a new (counting) variable</li> <li>• ... Count <math>\leftarrow 0</math> // to count the acceptable numbers</li> <li>• Insert a counting statement between lines 05 and 07</li> <li>• ... Count <math>\leftarrow</math> Count + 1</li> <li>• Add a new output after the loop/after line 13 / at the end (of the program)</li> <li>• ... OUTPUT Count</li> </ul>	<b>4</b>

Question	Answer	Marks
6	<p><b>One</b> mark for each correct feature, max <b>two</b>  <b>One</b> mark for each correct accompanying reason, max <b>two</b></p> <p><b>For example:</b></p> <p>Meaningful identifiers – to enable the programmer (or future programmers) to easily recognize the purpose of a variable / array / constant // to enable easy tracking of a variable / constant / array through the program</p> <p>Use of comments – to annotate each section of a program so that a programmer can find specific sections / so that the programmer knows the purpose of that section of code</p> <p>Procedures and functions – to make programs modular and easier to update / add functionality</p>	<b>4</b>

Question	Answer				Marks
7(a)	<b>One mark per correct column, max four</b>				4
	<b>Pointer</b>	<b>Letter</b>	<b>Choice</b>	<b>OUTPUT</b>	
	1	F			
	2				
	3				
	4				
	5				
	6			Letter F is represented by Foxtrot	
				Another Letter? (Y or N)	
			Y		
	1	D			
	2				
	3				
	4			Letter D is represented by Delta	
				Another Letter? (Y or N)	
			N		
7(b)	(Linear) search				1

Question	Answer	Marks
7(c)	<p><b>One</b> mark per mark point, max <b>two</b></p> <ul style="list-style-type: none"> <li>• The algorithm would not stop</li> <li>• ... because it would not have found the item it was seeking</li> </ul> <p><b>Or</b></p> <ul style="list-style-type: none"> <li>• The array would run out of values after the pointer reached 13</li> <li>• the algorithm will crash</li> </ul>	<b>2</b>

Question	Answer	Marks
8(a)	<p><b>One</b> mark per mark point, max <b>three</b></p> <ul style="list-style-type: none"> <li>• Storing string in Phrase</li> <li>• Correct use of LENGTH function</li> <li>• Correct use of UCASE function</li> <li>• Correct outputs of LENGTH and UCASE</li> </ul> <p><b>For example:</b></p> <pre>Phrase ← "The beginning is the most important part" OUTPUT LENGTH(Phrase) OUTPUT UCASE(Phrase)</pre>	<b>3</b>
8(b)	<p><b>One</b> mark for each correct line, max <b>two</b></p> <p>40</p> <pre>THE BEGINNING IS THE MOST IMPORTANT PART</pre>	<b>2</b>

Question	Answer	Marks
9(a)	<p>One mark for each correct gate, with the correct input(s) as shown.</p> 	4

Question	Answer	Marks																																				
9(b)	<p><b>Four</b> marks for eight correct outputs.  <b>Three</b> marks for six or seven correct outputs.  <b>Two</b> marks for four or five correct outputs.  <b>One</b> mark for two or three correct outputs</p> <table border="1" data-bbox="332 377 631 965"> <thead> <tr> <th data-bbox="339 382 384 425">A</th><th data-bbox="406 382 451 425">B</th><th data-bbox="473 382 518 425">C</th><th data-bbox="541 382 586 425">Z</th></tr> </thead> <tbody> <tr><td data-bbox="339 441 384 484">0</td><td data-bbox="406 441 451 484">0</td><td data-bbox="473 441 518 484">0</td><td data-bbox="541 441 586 484">0</td></tr> <tr><td data-bbox="339 500 384 543">0</td><td data-bbox="406 500 451 543">0</td><td data-bbox="473 500 518 543">1</td><td data-bbox="541 500 586 543">1</td></tr> <tr><td data-bbox="339 558 384 601">0</td><td data-bbox="406 558 451 601">1</td><td data-bbox="473 558 518 601">0</td><td data-bbox="541 558 586 601">1</td></tr> <tr><td data-bbox="339 617 384 660">0</td><td data-bbox="406 617 451 660">1</td><td data-bbox="473 617 518 660">1</td><td data-bbox="541 617 586 660">0</td></tr> <tr><td data-bbox="339 676 384 719">1</td><td data-bbox="406 676 451 719">0</td><td data-bbox="473 676 518 719">0</td><td data-bbox="541 676 586 719">0</td></tr> <tr><td data-bbox="339 735 384 778">1</td><td data-bbox="406 735 451 778">0</td><td data-bbox="473 735 518 778">1</td><td data-bbox="541 735 586 778">0</td></tr> <tr><td data-bbox="339 794 384 836">1</td><td data-bbox="406 794 451 836">1</td><td data-bbox="473 794 518 836">0</td><td data-bbox="541 794 586 836">1</td></tr> <tr><td data-bbox="339 852 384 895">1</td><td data-bbox="406 852 451 895">1</td><td data-bbox="473 852 518 895">1</td><td data-bbox="541 852 586 895">0</td></tr> </tbody> </table>	A	B	C	Z	0	0	0	0	0	0	1	1	0	1	0	1	0	1	1	0	1	0	0	0	1	0	1	0	1	1	0	1	1	1	1	0	4
A	B	C	Z																																			
0	0	0	0																																			
0	0	1	1																																			
0	1	0	1																																			
0	1	1	0																																			
1	0	0	0																																			
1	0	1	0																																			
1	1	0	1																																			
1	1	1	0																																			

Question	Answer	Marks
10(a)	<p><b>One</b> mark for the correct field name  <b>One</b> mark for the correct reason</p> <p><b>For example:</b></p> <p>TVCode  Each entry in this field is a unique identifier</p>	2

Question	Answer	Marks										
10(b)	<p><b>Two</b> marks for four correct answers.  <b>One</b> mark for two or three correct answers.</p> <table border="1" data-bbox="332 314 743 636"> <thead> <tr> <th data-bbox="332 314 586 377">Field</th><th data-bbox="586 314 743 377">Data type</th></tr> </thead> <tbody> <tr> <td data-bbox="332 377 586 441">TVCode</td><td data-bbox="586 377 743 441">Text</td></tr> <tr> <td data-bbox="332 441 586 504">ScreenSize</td><td data-bbox="586 441 743 504">Integer</td></tr> <tr> <td data-bbox="332 504 586 568">SmartTV</td><td data-bbox="586 504 743 568">Boolean</td></tr> <tr> <td data-bbox="332 568 586 636">Price\$</td><td data-bbox="586 568 743 636">Real</td></tr> </tbody> </table>	Field	Data type	TVCode	Text	ScreenSize	Integer	SmartTV	Boolean	Price\$	Real	2
Field	Data type											
TVCode	Text											
ScreenSize	Integer											
SmartTV	Boolean											
Price\$	Real											
10(c)	<p><b>One</b> mark for each correct answer</p> <p>ScreenSize  Price\$  FROM  YES</p> <p><b>Correct code:</b></p> <pre data-bbox="332 970 916 1087">SELECT TVCode, ScreenSize, Price\$  FROM TVRange  WHERE SmartTV = YES;</pre>	4										

Question	Answer	Marks
11	<p>Read the whole answer:  Check if each requirement listed below has been met. Requirements may be met using a suitable built-in function from the programming language used (Python, VB.NET or Java).  On place a SEEN mark if requirement met, cross if no attempt seen, omission mark and/or comment if partially met (see marked scripts).</p> <p>Use the tables for AO2 and AO3 below to award a mark in a suitable band using a best fit approach, then add up the total:</p> <ul style="list-style-type: none"> <li>• AO2 (maximum 9 marks)</li> <li>• AO3 (maximum 6 marks)</li> </ul> <p><b>Data structures required:</b>  The names underlined must match those given in the scenario:</p> <p>Arrays or lists <u>Days</u> [], <u>Readings</u> [], <u>AverageTemp</u> []</p> <p>Variables WeekLoop, DayLoop, InTemp, TotalDayTemp, TotalWeekTemp, AverageWeekTemp</p> <p><b>Requirements (techniques):</b></p> <p><b>R1</b> Input and store hourly temperatures and validation of input temperatures for each day (with prompts, range check and (nested)iteration)</p> <p><b>R2</b> Calculate, round to one decimal place and store daily average temperatures and calculate the weekly average temperature rounded to one decimal place (iteration, totalling and rounding)</p> <p><b>R3</b> Convert all average temperatures to Fahrenheit (to one decimal place) and output the average temperatures in both Celsius and Fahrenheit. Output with appropriate messages. (output and rounding)</p>	15

Question	Answer	Marks
11	<p><b>Example 15 mark answer in pseudocode</b></p> <pre> // meaningful identifiers and appropriate data structures for // all data required DECLARE Days : ARRAY[1:7] OF STRING DECLARE Readings : ARRAY[1:7, 1:24] OF REAL DECLARE AverageTemp : ARRAY[1:7] OF REAL DECLARE WeekLoop : INTEGER DECLARE DayLoop : INTEGER DECLARE InTemp : REAL DECLARE TotalDayTemp : REAL DECLARE TotalWeekTemp : REAL DECLARE AverageWeekTemp : REAL // initial population of Days[] array // input and a loop are also acceptable Days[1] ← "Sunday" Days[2] ← "Monday" Days[3] ← "Tuesday" Days[4] ← "Wednesday" Days[5] ← "Thursday" Days[6] ← "Friday" Days[7] ← "Saturday" // input temperatures inside nested loop FOR WeekLoop ← 1 TO 7     TotalDayTemp ← 0     FOR DayLoop ← 1 TO 24         OUTPUT "Enter temperature ", DayLoop, " for ", Days[WeekLoop] </pre>	

Question	Answer	Marks
11	<pre>         INPUT InTemp         // validation of input for between -20 and +50 inclusive         WHILE InTemp &lt; -20.0 OR InTemp &gt; 50.0 DO             OUTPUT "Your temperature must be between -20.0 and +50.0 inclusive. Please try                     again"             INPUT InTemp         ENDWHILE         Readings[WeekLoop, DayLoop] ← InTemp         // totalling of temperatures during the day         TotalDayTemp ← TotalDayTemp + ROUND(InTemp, 1)         NEXT DayLoop          // average temperature for the day         AverageTemp[WeekLoop] ← ROUND(TotalDayTemp / 24,1)         NEXT WeekLoop         // calculate the average temperature for the week         TotalWeekTemp ← 0         FOR WeekLoop ← 1 TO 7             TotalWeekTemp ← TotalWeekTemp + AverageTemp[WeekLoop]         NEXT WeekLoop          AverageWeekTemp ← ROUND(TotalWeekTemp / 7,1)         // outputs in Celsius and Fahrenheit         FOR WeekLoop ← 1 TO 7             OUTPUT "The average temperature on ", Days[WeekLoop], " was ", AverageTemp[WeekLoop], "                     Celsius and ",                     ROUND(AverageWeekTemp * 9 / 5 + 32), 1, " Fahrenheit"         NEXT WeekLoop          OUTPUT "The average temperature for the week was ",                 AverageWeekTemp, " Celsius and ", ROUND(AverageWeekTemp * 9 / 5 + 32, 1), "                 Fahrenheit"     </pre>	

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

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>