



# Cambridge IGCSE™

---

**COMPUTER SCIENCE**

**0478/22**

Paper 2 Problem-solving and Programming

**May/June 2022**

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 2022 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 **12** printed pages.

**Generic Marking Principles**

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.

**Please note the following further points:**

The words in **bold** in the mark scheme are important text that needs to be present, or some notion of it needs to be present. It does not have to be the exact word, but something close to the meaning.

If a word is underlined, this **exact** word must be present.

A single forward slash means this is an alternative word. A double forward slash means that this is an alternative mark point.

Ellipsis (...) on the end of one-mark point and the start of the next means that the candidate **cannot** get the second mark point without being awarded the first one. If a mark point has an ellipsis at the beginning, but there is no ellipsis on the mark point before it, then this is just a follow-on sentence and **can** be awarded **without** the previous mark point.

Question	Answer	Marks
<b>Section A</b>		
1(a)(i)	<p>Many correct answers, the name used must be meaningful. The name given is an example only.</p> <p><b>One mark per mark point, max three</b></p> <p>Constant           AdultCostOneDay Value               20.00 Use                  Storing the cost of an adult ticket for one day</p>	<b>3</b>
1(a)(ii)	<p>Many correct answers, the name used must be meaningful. The name given is an example only.</p> <p><b>One mark per mark point, max three</b></p> <p>Variable           NumberOfTickets // NumberOfAdultTickets Data type         Integer Use                 Inputting the number of tickets purchased // Inputting the number of adult tickets purchased</p>	<b>3</b>
1(b)	<p><b>One mark per mark point, max three</b></p> <p>MP1   using an array or variables // lookup // input <b>today's</b> date MP2   use a loop / FOR / REPEAT / WHILE statement MP3   ... to output / display <b>today's</b> date / <b>today</b> using a suitable message MP4   ... and the 6/7 following dates/days // loop 6/7 times</p> <p>or</p> <p>MP5   using an array or variables // lookup // input <b>today's</b> date MP6   array/ variables contains all the days of the week / the next seven days or dates MP7   attempt to output / display available days using a suitable message(s)</p>	<b>3</b>
1(c)	<p><b>One mark per mark point, max two</b></p> <p>MP1   method used to allocate a new number, e.g. initialise a variable or counter // select a random number MP2   how it ensured uniqueness, e.g. every booking number is one larger than the previous number       // check existing bookings to ensure it hasn't been used before</p>	<b>2</b>

Question	Answer	Marks
1(d)	<p><b>One mark per mark point, max six</b></p> <p>MP1 initialise a running total for the cost of the booking  MP2 input (with prompt) the number of tickets  MP3 ... attempt to identify type of ticket  MP4 ... add the cost of tickets to the running total  MP5 ability to book several types of ticket  MP6 attempt to check that the number of children booked is less than or equal to twice the number of adults  MP7 attempt to identify type of attraction  MP8 input <b>with prompt</b> the number of selected attractions required  MP9 ... check that this number is not greater than the total number of tickets booked  MP10 ... add the cost of the attractions to the (running) total</p> <p><b>Example answer</b></p> <pre>TotalCost ← 0 TotalPeople ← 0 OUTPUT "Do you want single tickets? Y/N" INPUT Single IF Single = "Y"   THEN     FOR Type ← 1 TO 3 // adults, children and seniors       REPEAT         Flag ← True         OUTPUT "How many ", TicketDescription[Type], " tickets"         INPUT NumberOfTickets[Type]         IF Type = 2           THEN             IF NumberOfTickets[2] &gt; 2 * NumberOfTickets[1]               THEN                 Flag ← FALSE                 OUTPUT "Too many children"               ENDIF             ENDIF           UNTIL NumberOfTickets[Type] &gt;= 0 AND Flag         TotalCost ← TotalCost + TicketPrice[Type] * NumberOfTickets[Type]</pre>	6

Question	Answer	Marks
1(d)	<pre> TotalPeople ← TotalPeople + NumberOfTickets[Type] NEXT Type ENDIF OUTPUT "Do you want a family ticket? Y/N" INPUT Family IF Family = "Y"   THEN     OUTPUT "How many?"     INPUT NumberOfTickets[4]     TotalCost ← TotalCost + TicketPrice[4] * NumberOfTickets[4]     TotalPeople ← TotalPeople + NumberOfTickets[4] * 5   ENDIF  OUTPUT "Do you want a group ticket? Y/N" INPUT Group IF Group = "Y"   THEN     OUTPUT "How many people in the group?"     INPUT NumberOfTickets[5]     TotalCost ← TotalCost + TicketPrice[5] * NumberOfTickets[5]     TotalPeople ← TotalPeople + NumberOfTickets[5]   ENDIF  REPEAT   OUTPUT "How many people would like to see the lion feeding?"   INPUT LionFeedingNumber   IF TotalPeople &gt;= LionFeedingNumber     THEN TotalCost ← TotalCost + LionFeedingNumber * LionfeedingPrice   ENDIF   OUTUT "How many people would like to see the penguin feeding?"   INPUT PenguinFeedingNumber   IF TotalPeople &gt;= PenguinFeedingNumber     THEN TotalCost ← TotalCost + PenguinFeedingNumber * PenguinfeedingPrice   ENDIF UNTIL TotalPeople &gt;= PenguinFeedingNumber AND TotalPeople &gt;= LionFeedingNumber </pre>	

Question	Answer	Marks
1(e)	<p>Explanation</p> <p><b>One</b> mark per mark point, max <b>three</b></p> <p>MP1 use of selection (or any other method) to identify bookings with 2 or more adults and/or seniors and 2 (allow 3) or more children</p> <p>MP2 calculating new price including family ticket(s) for booking / tickets sold</p> <p>MP3 ... use of condition to compare price of family ticket(s) to price of ordinary tickets / group ticket(s) ...</p> <p>MP4 ... identifying best value / lowest cost</p>	<b>3</b>



Question	Answer	Marks
<b>Section B</b>		
2(a)	<p><b>One mark per mark point, max three</b></p> <ul style="list-style-type: none"> <li>line 8 / <code>PassCheck ← TRUE</code> correction <code>PassCheck ← FALSE</code></li> <li>line 12 / <code>IF Password &lt;&gt; Password</code> correction <code>IF Password2 &lt;&gt; Password // IF Password &lt;&gt; Password2</code></li> <li>line 18 / <code>UNTIL PassCheck OR Attempt &lt;&gt; 3</code> correction <code>UNTIL PassCheck OR Attempt = 3 / UNTIL PassCheck OR Attempt &gt;= 3</code></li> </ul>	<b>3</b>
2(b)	<p><b>One mark check, one mark matching description, max four</b></p> <p>Check: <code>validation // length check</code> Description <code>length check // checks number of characters in password</code></p> <p>Check: <code>verification // double entry</code> Description <code>double entry // comparison that two inputs are the same</code></p>	<b>4</b>
2(c)	<p><b>One mark per set, one mark matching reason, max four</b></p> <p>Set 1 – any appropriate example e.g. “small” Reason must follow through from the password given e.g. abnormal data will be rejected</p> <p>Set 2 – any different appropriate example e.g. “password” and “password” Reason must be different and follow through from the password given e.g. normal data will be accepted</p>	<b>4</b>

Question	Answer	Marks
3(a)	<p><b>One mark per mark point, max two</b></p> <ul style="list-style-type: none"> <li>a list / one column of items</li> <li>... of the same data type</li> <li>... stored under a <b>single</b> identifier</li> <li>... with a single index to identify each element</li> </ul> <p><b>One mark for an example of a declaration</b></p> <ul style="list-style-type: none"> <li>example e.g. <code>DECLARE MyArray[1:10] OF INTEGER</code></li> </ul>	<b>3</b>

Question	Answer	Marks
3(b)	<p><b>One</b> mark per mark point, max <b>two</b></p> <ul style="list-style-type: none"> <li>• using a counter to index the array</li> <li>• so that the same code can be repeatedly used to check every element // every element can be checked in a loop</li> </ul>	<b>2</b>

Question	Answer	Marks																																																																																											
4(a)	<p><b>One</b> mark for each correct column</p> <table border="1"> <thead> <tr> <th>Counter</th> <th>Hot</th> <th>Cold</th> <th>Serve</th> <th>Temp</th> <th>Error</th> <th>OUTPUT</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>1</td> <td></td> <td></td> <td>1</td> <td>75</td> <td></td> <td></td> </tr> <tr> <td>2</td> <td></td> <td></td> <td>2</td> <td>78</td> <td></td> <td></td> </tr> <tr> <td>3</td> <td></td> <td></td> <td>3</td> <td>84</td> <td></td> <td></td> </tr> <tr> <td>4</td> <td>1</td> <td></td> <td></td> <td>87</td> <td></td> <td>Too Hot</td> </tr> <tr> <td>5</td> <td>2</td> <td></td> <td></td> <td>91</td> <td></td> <td>Too Hot</td> </tr> <tr> <td>6</td> <td></td> <td></td> <td>4</td> <td>80</td> <td></td> <td></td> </tr> <tr> <td>7</td> <td></td> <td></td> <td>5</td> <td>75</td> <td></td> <td></td> </tr> <tr> <td>8</td> <td></td> <td></td> <td>6</td> <td>70</td> <td></td> <td></td> </tr> <tr> <td>9</td> <td></td> <td></td> <td>7</td> <td>65</td> <td></td> <td></td> </tr> <tr> <td>10</td> <td></td> <td>1</td> <td></td> <td>62</td> <td></td> <td>Too Cold</td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> <td>-1</td> <td>30</td> <td>30</td> </tr> </tbody> </table>	Counter	Hot	Cold	Serve	Temp	Error	OUTPUT	0	0	0	0				1			1	75			2			2	78			3			3	84			4	1			87		Too Hot	5	2			91		Too Hot	6			4	80			7			5	75			8			6	70			9			7	65			10		1		62		Too Cold					-1	30	30	<b>7</b>
Counter	Hot	Cold	Serve	Temp	Error	OUTPUT																																																																																							
0	0	0	0																																																																																										
1			1	75																																																																																									
2			2	78																																																																																									
3			3	84																																																																																									
4	1			87		Too Hot																																																																																							
5	2			91		Too Hot																																																																																							
6			4	80																																																																																									
7			5	75																																																																																									
8			6	70																																																																																									
9			7	65																																																																																									
10		1		62		Too Cold																																																																																							
				-1	30	30																																																																																							

Question	Answer	Marks
4(b)	<ul style="list-style-type: none"><li>include a message to explain the value output / e. g. "The percentage of meals not served" // outputting <code>Hot, Cold and Serve</code></li></ul>	<b>1</b>
4(c)	<ul style="list-style-type: none"><li>updating the <code>Serve</code> variable // <code>Serve ← Serve + 1</code></li></ul>	<b>1</b>

Question	Answer	Marks																																																												
5(a)	<p><b>One</b> mark per mark point, max <b>four</b></p> <ul style="list-style-type: none"> <li>• correct rows Field and Table</li> <li>• correct row Show</li> <li>• correct Criteria for SingleUse <b>or/and</b> Uses</li> <li>• correct Criteria for StockLevel less than ReorderLevel</li> </ul> <table border="1" data-bbox="533 427 1771 820"> <tr> <td>Field:</td> <td>ItemNumber</td> <td>Description</td> <td>SingleUse</td> <td>StockLevel</td> </tr> <tr> <td>Table:</td> <td>NURSE</td> <td>NURSE</td> <td>NURSE</td> <td>NURSE</td> </tr> <tr> <td>Sort:</td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Show:</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> <tr> <td>Criteria:</td> <td></td> <td></td> <td>True</td> <td>&lt;[ReorderLevel]</td> </tr> <tr> <td>or:</td> <td></td> <td></td> <td></td> <td></td> </tr> </table> <p>Or</p> <table border="1" data-bbox="533 887 1771 1279"> <tr> <td>Field:</td> <td>ItemNumber</td> <td>Description</td> <td>Uses</td> <td>StockLevel</td> </tr> <tr> <td>Table:</td> <td>NURSE</td> <td>NURSE</td> <td>NURSE</td> <td>NURSE</td> </tr> <tr> <td>Sort:</td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Show:</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> <tr> <td>Criteria:</td> <td></td> <td></td> <td>= 1</td> <td>&lt;[ReorderLevel]</td> </tr> <tr> <td>or:</td> <td></td> <td></td> <td></td> <td></td> </tr> </table>	Field:	ItemNumber	Description	SingleUse	StockLevel	Table:	NURSE	NURSE	NURSE	NURSE	Sort:					Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Criteria:			True	<[ReorderLevel]	or:					Field:	ItemNumber	Description	Uses	StockLevel	Table:	NURSE	NURSE	NURSE	NURSE	Sort:					Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Criteria:			= 1	<[ReorderLevel]	or:					4
Field:	ItemNumber	Description	SingleUse	StockLevel																																																										
Table:	NURSE	NURSE	NURSE	NURSE																																																										
Sort:																																																														
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>																																																										
Criteria:			True	<[ReorderLevel]																																																										
or:																																																														
Field:	ItemNumber	Description	Uses	StockLevel																																																										
Table:	NURSE	NURSE	NURSE	NURSE																																																										
Sort:																																																														
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>																																																										
Criteria:			= 1	<[ReorderLevel]																																																										
or:																																																														
5(b)	the field Uses already shows this information // duplication of data // redundant data	1																																																												