Cambridge Assessment International Education
Cambridge International General Certificate of Secondary Education

COMPUTER SCIENCE **0478/21**

Paper 2 **October/November 2019**

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 October/November 2019 series for most Cambridge IGCSE™, Cambridge International A and AS Level components and some Cambridge O Level components.

This syllabus is regulated for use in England, Wales and Northern Ireland as a Cambridge International Level 1/Level 2 Certificate.

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.

Question	Answer	Marks
1(a)	Two examples of: Any meaningful name for a variable related to **Task 1** – **one** mark Correct data type related to **Task 1** – **one** mark Correct purpose related to **Task 1** – **one** mark e.g. • Length // Width • … real • … to store the length // width of the patio • StoneType • … string • … to store the type of stone slab chosen by the user • PatioArea • … integer • … to store the area of stone needed for the patio Note: variable names should not contain spaces or punctuation marks	**6**
1(b)	**Two** from: • Use of two one-dimensional arrays … • … with matching indexes • … each with a specific data types, e.g. string for stone and real for price • Size of array / number of elements / length of each array is 6 • Meaningful array names, e.g. Stone and Price	**2**

Question	Answer	Marks

1(c)	**Five** from:
	MP1 Prompt and input for number of rectangles making up the patio …
	MP2 … and the type of stone to be used
	MP3 Loop to input dimensions for rectangles
	MP4 Prompt and input for dimensions for each rectangle inside loop
	MP5 Calculation of area of a rectangle
	MP6 Running total of area of patio
	MP7 Looking up the cost of the stone
	MP8 Calculation of cost of stone …
	MP9 …rounded up to the nearest square metre
	MP10 Output of cost of patio with annotation
	Example
	`TotalCost ← 0`
	`OUTPUT "Please enter type of stone needed"`
	`INPUT StoneType`
	`OUTPUT "Please enter the number of rectangles needed"`
	`INPUT NumberRectangle`
	`FOR Counter ← 1 TO NumberRectangle`
	`  OUTPUT "Please enter the length"`
	`  INPUT Length`
	`  OUTPUT "Please enter the Width"`
	`  INPUT Width`
	`  TotalCost ← TotalCost + CostFromTask1(Length, Width, StoneType)`
	`  // use Task 1 to calculate cost`
	`NEXT Counter`
	`OUTPUT ("Total Cost of Patio " TotalCost`
		5

1(d)	**Three** from: • Explanation of user input to name the percentage value to be used • Explanation of the calculation to add this percentage to the already calculated quantity of stone required • Explanation of rounding this value up • Explanation of the calculation of the new cost • Explanation of the output that will include annotation, quantity of stone to the nearest square metre and cost of stone If only program statements given with no explanation, **zero** marks.	**3**

Question	Answer	Marks
1(e)	Two examples of: **One** mark for each correct validation check related to patio dimensions in **Task 1 or Task 2** and **one** mark for an appropriate related purpose e.g. • Range check // Limit check • … to make sure the dimension is entered is greater than zero and less than the maximum size • Type check • … to make sure any dimension entered is a number • Presence check • … to make sure a length/width has been entered for the area of the patio	**4**

Question	Answer	Marks
2	**One** mark for each correct row	**4**

Description	Structure diagram	Flowchart	Library routines
A modelling tool used to show the hierarchy of a system	✓		
A collection of standard programs available for immediate use			✓
A graphical representation used to represent an algorithm		✓	
A graphical representation to show how a system is broken into sub-systems	✓		

Question	Answer	Marks
3(a)	• Inputs a series of values • Finds the total • Prints out the average	**3**
3(b)	**Three** from: • Use of loop structure • Allow input to define the limit of the loop / use sentinel value • Keeping a count of the number of values • It could use a totalling process to keep a running total	**3**

3(c)	Marks awarded as follows (maximum five marks):
	• Initialise Total
	• Enter limit
	• Suitable loop structure
	• Correct input
	• Correct totalling
	• Correct output
	e.g.
	
	`Total ← 0`
	`INPUT CounterLimit`
	`FOR LoopCounter ← 1 To CounterLimit`
	`  INPUT Number`
	`  Total ← Total + Number`
	`NEXT LoopCounter`
	`OUTPUT "The average equals ", Total / CounterLimit`
		5

Question	Answer	Marks

4(a)

Index	Count	Value	PassMarks								OUTPUT
			[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	
0											
0	0	58	58								
1	1	40									
1	2	67		67							
2	3	85			85						
3	4	12									
3	5	13									
3	6	75				75					
4	7	82					82				
5											Number passed 5
1 mark	1 mark	1 mark	1 mark	1 mark							1 mark

6

4(b)

One from:
- Stores numbers greater than 50 in an array
- Outputs number of times pass mark has been met
- Find the number of pass marks

1

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>5(a)</td><td>
<table>
<tr><th>Field name</th><th>Example of data</th></tr>
<tr><td>CarID</td><td>ID07</td></tr>
<tr><td>Model</td><td>Pegasus // Apollo // Cupid</td></tr>
<tr><td>BodyStyle</td><td>estate //saloon // hatchback</td></tr>
<tr><td>Doors</td><td>1 // 2 // 3 // 4 // 5</td></tr>
<tr><td>FuelType</td><td>batteries // petrol // diesel</td></tr>
</table>
One mark – 1 or 2 suitable names and corresponding examples of data // 5 suitable field names but all data incorrect
Two marks – 3 or 4 suitable names and corresponding examples of data
Three marks – 5 suitable names and corresponding examples of data
Notes: CarID can be anything that could be used as a unique identifier. e.g. Number Plate
Number of doors can be any number of sensible doors for a car – 1, 2, 3, 4, 5.
Other data must come from the given text. Allow codes for model and body style.
</td><td>3</td></tr>
<tr><td>5(b)</td><td>
• CarID
• Which contains unique values to identify each record
</td><td>2</td></tr>
<tr><td>5(c)</td><td>
<table>
<tr><td>Field:</td><td>Model</td><td>FuelType</td><td>Doors</td><td></td><td></td></tr>
<tr><td>Table:</td><td>CAR_RANGE</td><td>CAR_RANGE</td><td>CAR_RANGE</td><td></td><td></td></tr>
<tr><td>Sort:</td><td>Ascending</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Show:</td><td>☑</td><td>☐</td><td>☑</td><td>☐</td><td>☐</td></tr>
<tr><td>Criteria:</td><td></td><td>="Petrol"</td><td></td><td></td><td></td></tr>
<tr><td>or:</td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
1 mark for each completely correct column (maximum three marks)
</td><td>3</td></tr>
</table>