Cambridge IGCSE

Cambridge Assessment International Education
Cambridge International General Certificate of Secondary Education

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

8649838361

COMPUTER SCIENCE **0478/21**

Paper 2 Problem-solving and Programming **October/November 2019**

1 hour 45 minutes

Candidates answer on the Question Paper.

No Additional Materials are required.

No calculators allowed.

READ THESE INSTRUCTIONS FIRST

Write your centre number, candidate number and name in the spaces at the top of this page.
Write in dark blue or black pen.
You may use an HB pencil for any diagrams, graphs or rough working.
Do not use staples, paper clips, glue or correction fluid.
DO **NOT** WRITE IN ANY BARCODES.

Answer **all** questions.

DO NOT ATTEMPT TASKS 1, 2 AND 3 in the pre-release material; these are for information only.

You are advised to spend no more than **40 minutes** on **Section A** (Question 1).

No marks will be awarded for using brand names of software packages or hardware.

Any businesses described in this paper are entirely fictitious.

At the end of the examination, fasten all your work securely together.

The number of marks is given in brackets [ ] at the end of each question or part question.

The maximum number of marks is 50.

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

This document consists of **11** printed pages and **1** blank page.

Section A

You are advised to spend no longer than 40 minutes answering this section.

Here is a copy of the pre-release material.

DO NOT attempt Tasks 1, 2 and 3 now.

Use the pre-release material and your experience from attempting the tasks before the examination to answer Question 1.

Pre-release material

You have been asked to write a program to calculate the area of a patio and the cost of the stone slabs needed to cover it. The program should work for any patio that can be represented as a rectangle, or group of rectangles that are joined together, and only one type of stone slab may be used.

Type of stone slab	Price per square metre
Dover	$30.00
Exeter	$35.00
London	$42.00
Portland	$49.50
Shaftesbury	$55.00
York	$62.75

Write and test a program or programs to calculate the cost of the stone slabs for a patio.

- Your program or programs must include appropriate prompts for the entry of data; data must be validated on entry.
- Error messages and other output need to be set out clearly and understandably.
- All variables, constants and other identifiers must have meaningful names.

You will need to complete these **three** tasks. Each task must be fully tested.

Task 1 – Setting up the system for a simple rectangular patio.
Set up your program to:
- Store the type and price per square metre of the stone slabs using a suitable programming technique.
- Prompt and allow the user to input the length and width of their patio and the type of stone slab they would like.
- Calculate and display the number of square metres of stone slabs required, rounded up to the next whole square metre, and the total cost of the stone slabs.

Task 2 – Working on more complex shapes.
Assuming that a patio can be made up of a group of rectangles, extend your program to:
- Enter the number of rectangles making up the patio and the type of stone slab to be used.
- Allow the dimensions for each rectangle to be entered.
- Calculate and display the total area of the patio rounded up to the next whole square metre.
- Calculate and display the total cost of the stone slabs.

Task 3 – Allowing for waste.
It is likely that some of the stone will not be useable, so it is sensible to allow a percentage for wastage, for example, 10%. Alter your program to allow the user to input a percentage to calculate wastage and add this to the number of square metres of stone slabs to be purchased, rounded up to the next whole square metre. Display the revised total area and cost.

1 **(a)** All variables, constants and other identifiers must have meaningful names.

Name **two** variables that you could have used for **Task 1**. State the data type and purpose of each variable.

Variable 1 name .....

Data type .....

Purpose .....

.....

Variable 2 name .....

Data type .....

Purpose .....

.....

[6]

(b) Describe how arrays could be used to store the data about the types and price per square metre of the stone slabs for **Task 1**.

.....

.....

.....

.....

.....

.....

.....[2]

(c) Write an algorithm to complete **Task 2**, using **either** pseudocode, programming statements **or** a flowchart. Assume **Task 1** has been completed.

[5]

(d) Explain how your program completes **Task 3**. Any programming statements you use in your answer must be fully explained.

....[3]

(e) Name **two** validation checks that could be used when entering patio dimensions in **Task 1** or **Task 2** and describe their purpose.

Validation check 1 .....

Purpose .....

.....

.....

.....

Validation check 2 .....

Purpose .....

.....

.....

.....

[4]

Section B

2 For each of the **four** descriptions in the table, place a tick in the correct column to show whether it describes a **Structure diagram**, a **Flowchart** or **Library routines**.

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.			

[4]

3 Examine the following pseudocode:

```
INPUT A
INPUT B
INPUT C
INPUT D
INPUT E
INPUT F
INPUT G
INPUT H
INPUT I
INPUT J
INPUT K
INPUT L
T ← A + B + C + D + E + F + G + H + I + J + K + L
OUTPUT "The average equals ", T/12
```

(a) Describe what happens in this pseudocode.

.....

.....

.....

.....

.....

.....[3]

(b) Describe how this pseudocode could be altered to allow any number of values to be input.

.....

.....

.....

.....

.....

.....

.....

.....[3]

(c) Re-write the given pseudocode to allow any number of values to be input.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[5]

4 **(a)** Complete the trace table for this algorithm using the given input data.

```
Index ← 0
FOR Count ← 0 TO 7
  INPUT Value
  IF Value > 50
    THEN
      PassMarks[Index] ← Value
      Index ← Index + 1
  ENDIF
NEXT Count
PRINT "Number passed ", Index
```

Input data: 58, 40, 67, 85, 12, 13, 75, 82

Index	Count	Value	PassMarks								OUTPUT
			[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	

[6]

(b) Give the purpose of the algorithm shown in **part (a)**.

.....

.....

.....[1]

5 A car manufacturer makes a range of car models named Pegasus, Apollo and Cupid. It keeps a database to store the records of its range and the different options for each car model. Within the table CAR_RANGE, the following data needs to be stored:

1. Car model
2. Body style – saloon, hatchback or estate
3. How many doors it has
4. Whether it uses petrol, diesel or batteries as fuel
5. An identifier for a specific car.

(a) Complete the table to show suitable field names and an example of appropriate data for each field in the database table CAR_RANGE.

Field name	Example of data

[3]

(b) State which of your fields would be most appropriate for a primary key and give a reason for your choice.

.....

.....

.....[2]

(c) Complete the query-by-example grid to provide a list of car models using petrol and the number of doors these cars have, in alphabetical order of car model. Display only the car models and the number of doors.

Field:					
Table:					
Sort:					
Show:	☐	☐	☐	☐	☐
Criteria:					
or:					

[3]

BLANK PAGE

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

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>4(a)</td><td>
<table>
<tr><th rowspan="2">Index</th><th rowspan="2">Count</th><th rowspan="2">Value</th><th colspan="8">PassMarks</th><th rowspan="2">OUTPUT</th></tr>
<tr><th>[0]</th><th>[1]</th><th>[2]</th><th>[3]</th><th>[4]</th><th>[5]</th><th>[6]</th><th>[7]</th></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>0</td><td>0</td><td>58</td><td>58</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1</td><td>1</td><td>40</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1</td><td>2</td><td>67</td><td></td><td>67</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>3</td><td>85</td><td></td><td></td><td>85</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>3</td><td>4</td><td>12</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>3</td><td>5</td><td>13</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>3</td><td>6</td><td>75</td><td></td><td></td><td></td><td>75</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>4</td><td>7</td><td>82</td><td></td><td></td><td></td><td></td><td>82</td><td></td><td></td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>Number passed 5</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1 mark</td><td>1 mark</td><td>1 mark</td><td>1 mark</td><td colspan="7">1 mark</td><td>1 mark</td></tr>
</table>
</td><td>6</td></tr>
<tr><td>4(b)</td><td>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</td><td>1</td></tr>
</table>

<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>