Cambridge Assessment International Education

Cambridge IGCSE™

CANDIDATE NAME									
CENTRE NUMBER						CANDIDATE NUMBER			

1129138452

COMPUTER SCIENCE **0478/22**

Paper 2 Problem-solving and Programming **October/November 2020**

1 hour 45 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- **Do not attempt Tasks 1, 2 and 3** in the copy of the pre-release material on page 2; these are for information only.
- Use a black or dark blue pen. You may use an HB pencil for any diagrams or graphs.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 50.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages. Blank pages are indicated.

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

An online computer shop sells customised personal computers. Every computer sold includes a basic set of components costing $200 and additional items can be added from the table:

Category	Item code	Description	Price ($)
Case	A1	Compact	75.00
Case	A2	Tower	150.00
RAM	B1	8 GB	79.99
RAM	B2	16 GB	149.99
RAM	B3	32 GB	299.99
Main Hard Disk Drive	C1	1 TB HDD	49.99
Main Hard Disk Drive	C2	2 TB HDD	89.99
Main Hard Disk Drive	C3	4 TB HDD	129.99
Solid State Drive	D1	240 GB SSD	59.99
Solid State Drive	D2	480 GB SSD	119.99
Second Hard Disk Drive	E1	1 TB HDD	49.99
Second Hard Disk Drive	E2	2 TB HDD	89.99
Second Hard Disk Drive	E3	4 TB HDD	129.99
Optical Drive	F1	DVD/Blu-Ray Player	50.00
Optical Drive	F2	DVD/Blu-Ray Re-writer	100.00
Operating System	G1	Standard Version	100.00
Operating System	G2	Professional Version	175.00

As well as the basic set of components every computer must include one case, one RAM and one Main Hard Disk Drive from the table.
A computer is supplied with or without an Operating System.

Write and test a program or programs for the online computer shop.
- 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 arrays, 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 and ordering the main items.
Write a program to:
- use arrays to store the item code, description and price
- allow a customer to choose one case, one RAM and one Main Hard Disk Drive
- calculate the price of the computer using the cost of the chosen items and the basic set of components
- store and output the chosen items and the price of the computer.

Task 2 – Ordering additional items.
Extend TASK 1 to:
- allow a customer to choose whether to purchase any items from the other categories – if so, which item(s)
- update the price of the computer
- store and output the additional items and the new price of the computer.

Task 3 – Offering discounts.
Extend TASK 2 to:
- apply a 5% discount to the price of the computer if the customer has bought only one additional item
- apply a 10% discount to the price of the computer if the customer has bought two or more additional items
- output the amount of money saved and the new price of the computer after the discount.

1 All variables, constants and other identifiers must have meaningful names.

(a) (i) Identify **one** array you could have used for **Task 1** and state its purpose.

Array .....

Purpose .....

.....

.....
[2]

(ii) Identify **one** variable you could have used for **Task 2** and state its purpose.

Variable .....

Purpose .....

.....

.....
[2]

(iii) Identify **one** constant you could have used for **Task 3** and state its purpose.

Constant .....

Purpose .....

.....

.....
[2]

(b) Explain the benefits of storing Price as a real data type.

.....

.....

.....

.....

.....

.....

..... [2]

(c) Write an algorithm to show how you completed **Task 1**, using **either** pseudocode, programming statements **or** a flowchart. It is not necessary to show initialisation or setting up of arrays in your answer.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

[6]

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

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [4]

(e) Describe how you could alter your program to allow more than one computer to be bought.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [2]

Section B

2 Tick (✓) **one** box in each row to identify if the statement about subroutines is **true** or **false**.

Statement	true (✓)	false (✓)
A subroutine is called from within a program.		
A subroutine is **not** a complete program.		
A subroutine is a self-contained piece of code.		
A subroutine must return a value to the code from which it was called.		

[2]

3 This pseudocode algorithm is used as a validation check.

```
PRINT "Input a number from 1 to 5000"
REPEAT
  INPUT Number
  IF Number < 1 OR Number > 5000
    THEN
      PRINT "Invalid number, please try again"
  ENDIF
UNTIL Number >= 1 AND Number <= 5000
PRINT Number, " is within the correct range"
```

Identify **three** different types of test data. For each type, give an example of the test data you would use to test this algorithm and state a reason for your choice of test.

Type of test data 1 .....

Test data .....

Reason .....

.....

Type of test data 2 .....

Test data .....

Reason .....

.....

Type of test data 3 .....

Test data .....

Reason .....

.....

[6]

4 This pseudocode algorithm allows 5000 numbers to be entered and stored in an array called `Number`.

```
FOR Count ← 1 TO 5000
  INPUT Number[Count]
NEXT Count
```

Extend and re-write the algorithm using pseudocode to also count and output how many of the numbers stored in the array are greater than 500, using the variable `Higher`. Only output `Higher` once with an appropriate message.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [6]

5 This pseudocode represents an algorithm.

```
REPEAT
  Flag ← 0
  FOR Count ← 0 to 3
    IF Num[Count] < Num[Count + 1]
      THEN
        Store ← Num[Count]
        Num[Count] ← Num[Count + 1]
        Num[Count + 1] ← Store
        Flag ← 1
    ENDIF
  NEXT Count
UNTIL Flag = 0
```

(a) The contents of the array at the start of the algorithm are:

Num[0]	Num[1]	Num[2]	Num[3]	Num[4]
45	56	30	12	15

Complete the trace table for the algorithm using the data given in the array.

Flag	Count	Num[0]	Num[1]	Num[2]	Num[3]	Num[4]	Store
		45	56	30	12	15	

[5]

(b) Describe the purpose of the algorithm.

.....

.....

.....

..... [2]

6 Draw a flowchart symbol to represent each of the following:

Input/Output	Decision

[2]

Question 7 starts on Page 12.

7 The table AUDIOPARTS stores the part number, description, cost and quantity in stock of the items sold by a music shop.

PartNum	Description	Cost	Quantity
A01	Compact Amplifier Case	50.00	15
A02	Deluxe Amplifier Case	75.00	1
A03	Amplifier Standard	79.99	48
A04	Amplifier Midrange	149.99	50
A05	Amplifier Megablaster	299.99	48
S01	Tweeter	59.99	10
S02	Midrange Woofer	99.99	0
S03	Subwoofer	139.99	16
S04	Tower Speaker Basic	159.99	25
S05	Tower Speaker Skyscraper	219.99	9
S06	Centre Speaker	149.99	25
S07	Soundbar	89.99	2
S20	Soundbar	129.99	0
S21	Ceiling Surround Speaker	75.00	15
S22	Ceiling Full Range Speaker	100.00	1
S25	Surround Speaker	100.00	60
T19	Speaker Stands (Pair)	75.00	60

(a) State the number of records in the table AUDIOPARTS

..... [1]

(b) Identify the field that is most suitable to be a primary key and give a reason for your choice.

Fieldname .....

Reason .....

.....

.....

[2]

(c) Complete the query-by-example grid to show the items where the quantity in stock is fewer than 10. Show all the fields from the database table in descending order of cost.

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

[4]

Cambridge Assessment International Education

Cambridge IGCSE™

COMPUTER SCIENCE **0478/22**

Paper 2 **October/November 2020**

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 2020 series for most Cambridge IGCSE™, Cambridge International A and AS Level and Cambridge Pre-U components, and some Cambridge O Level components.

[Turn over

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)(i)	Any meaningful name for an array related to **Task 1** – **one** mark e.g.
	`SysStore`
	`SysPrice`
	
	Correct purpose related to **Task 1** – **one** mark e.g.
	…to store the system (components) that have been purchased
	…to store the (total) price of the system (being purchased)
		2

1(a)(ii)	Any meaningful name for a variable related to **Task 2** – **one** mark e.g.
	`Component`
	`TotalPrice`
	
	Correct purpose related to **Task 2** - **one** mark e.g.
	… to allow input of a component code
	… to store/calculate the running total price of the system
		2

1(a)(iii)	Any meaningful name for a constant related to **Task 3** – **one** mark e.g.
	`Offer5`
	`Offer10`
	
	Correct purpose related to **Task 3** - **one** mark e.g.
	… to store the one option discount rate
	… to store the two-option discount rate
		2

1(b)	Mark as **either**: Two distinct different points **OR** One point and an expansion Example answers: Real data can be used in calculations directly (which is required of the Price data) (1) Data can be stored with decimal places (1) Real numbers can be used in calculations (1) which is not possible with strings (1)	**2**

Question	Answer	Marks

1(c)	Any **six** from:
	MP1 At least one input (case, RAM, HDD)
	MP2 All three inputs fully prompted
	MP3 An attempt at validation of input
	MP4 One complete validation of input with error message
	MP5 Finding the price for one chosen item
	MP6 Finding the prices of the other two chosen items correctly
	MP7 Calculation of price of the chosen items
	MP8 …add the basic components cost to the cost of the chosen items
	MP9 Storage of chosen items
	MP10 Output to show chosen items and price of the computer (with appropriate message)
	
	Example answer:
	`OUTPUT "Which type of Case would you like? Input the Item Code"`
	`ComponentFlag ← False`
	`WHILE ComponentFlag = False`
	`  INPUT CaseCode`
	`  Count ← 0`
	`  WHILE Count<2 DO`
	`  IF CaseCode = ComponentCode[Count]`
	`  THEN`
	`  CaseIndex ← Count`
	`  ComponentFlag ← True`
	`  Count ← 2`
	`  ENDIF`
	`  Count ← Count + 1`
	`  ENDWHILE`
	`  IF ComponentFlag = False`
	`  THEN`
	`  OUTPUT "Your case Item Code doesn't exist, please try again"`
	`  ENDIF`
	`ENDWHILE`
	`OUTPUT "Which type of RAM would you like? Input the Item Code"`
	`ComponentFlag ← False`
	`WHILE ComponentFlag = False`
	`  INPUT RAMCode`
	`  Count ← 2`
	`  WHILE Count<5 DO`
	`  IF RAMCode = ComponentCode[Count]`
	`  THEN`
	`  RAMIndex ← Count`
	`  ComponentFlag ← True`
	`  Count ← 5`
	`  ENDIF`
	`  Count ← Count + 1`
	`  ENDWHILE`
		6

Question	Answer	Marks

1(c)	IF ComponentFlag = False
	THEN
	OUTPUT "Your RAM Item Code doesn't exist, please try again"
	ENDIF
	ENDWHILE
	OUTPUT "Which type of Primary Hard Disk Drive would you like? Input the Item Code"
	ComponentFlag ← False
	WHILE ComponentFlag = False
	INPUT PHDDCode
	Count ← 5
	WHILE Count<8 DO
	IF PHDDCode = ComponentCode[Count]
	THEN
	HDDIndex ← Count
	ComponentFlag ← True
	Count ← 8
	ENDIF
	Count ← Count + 1
	ENDWHILE
	IF ComponentFlag = False
	THEN
	OUTPUT "Your Primary HDD Item Code doesn't exist, please try again"
	ENDIF
	ENDWHILE
	TotalPrice ← 200 + ComponentPrice[CaseIndex] + ComponentPrice[RAMIndex] + ComponentPrice[HDDIndex]
	OUTPUT "Your computer consists of ", Description[CaseIndex], " case, ", Description[RAMIndex], " RAM and ", Description[HDDIndex], " Primary Hard Disk Drive."
	OUTPUT "The total price of your computer is $", TotalPrice	

1(d)	Any **four** from: MP1 Explanation of how the number of additional parts is stored MP2 Explanation of counting of additional parts being added to the system MP3 Explanation of determination of additional parts being 1, or more than 1 MP4 Explanation of using the correct percentage discount MP5 Explanation of calculating the money saved and finding the new price MP6 Explanation of correct output of money saved and new price	**4**
1(e)	Any **two** from: MP1 Prompt and input to ask buyer how many computers they wish to purchase (at the start) // When the first computer is complete, prompt and input to ask if they would like to purchase another computer MP2 Introduce an appropriate loop structure MP3 New storage for more than one computer // Enable the ordering of multiple computers of the same specification	**2**

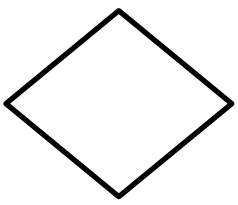
<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td colspan="3">Section B</td></tr>
<tr><td>2</td><td>
<table>
<tr><th>Statement</th><th>true (✓)</th><th>false (✓)</th></tr>
<tr><td>A subroutine is called from within a program.</td><td>✓</td><td></td></tr>
<tr><td>A subroutine is not a complete program.</td><td>✓</td><td></td></tr>
<tr><td>A subroutine is a self-contained piece of code.</td><td>✓</td><td></td></tr>
<tr><td>A subroutine must return a value to the code from which it was called.</td><td></td><td>✓</td></tr>
</table>
Two marks for four correct rows
One mark for any two correct rows
</td><td>2</td></tr>
</table>

Question	Answer	Marks
3	**One** mark for each correct type of test and **one** mark for each correct accompanying example of test data and reason (max **six**) e.g. • Extreme data • 5000 • to check it is accepted • Normal data • 300 • To check it is accepted • Abnormal data • 10000 • To check it is rejected	**6**

Question	Answer	Marks

4	Any **six** from:
	MP1 Initialisation of `Higher` to `0` before the loop
	MP2 Use of IF statement
	MP3 Correct condition in IF statement
	MP4 Correct counting statement inside loop
	MP5 `OUTPUT/PRINT` statement with correct reference to `Higher`
	MP6 Appropriate message in output
	MP7 Correct location of `OUTPUT` and `IF` statements
	
	`Higher ← 0`
	`FOR Count ← 1 TO 5000`
	`  INPUT Number[Count]`
	`  IF Number[Count] > 500`
	`  THEN`
	`  Higher ← Higher + 1`
	`  ENDIF`
	`NEXT Count`
	`OUTPUT "There are ", Higher, " values that are greater than 500"`
		6

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>5(a)</td><td>
<table>
<tr><th>Flag</th><th>Count</th><th>Num [0]</th><th>Num [1]</th><th>Num [2]</th><th>Num [3]</th><th>Num [4]</th><th>Store</th></tr>
<tr><td></td><td></td><td>45</td><td>56</td><td>30</td><td>12</td><td>15</td><td></td></tr>
<tr><td>0</td><td>0</td><td></td><td></td><td></td><td></td><td></td><td>45</td></tr>
<tr><td></td><td></td><td>56</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td><td>45</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>3</td><td></td><td></td><td></td><td></td><td></td><td>12</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td>15</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>12</td><td></td></tr>
<tr><td>0</td><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>3</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></tr>
</table>
One mark – <code>Flag</code> column
One mark – <code>Count</code> column
One mark – <code>Num[0]</code> and <code>Num[1]</code> columns
One mark – <code>Num[2], Num[3]</code> and <code>Num[4]</code> columns
One mark – <code>Store</code> column
</td><td>5</td></tr>
<tr><td>5(b)</td><td>Any two from:
• The algorithm sorts/orders numbers
• … into descending order / from largest to smallest</td><td>2</td></tr>
</table>

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>6</td><td>Input/Output
 
Decision
 
One mark for each correct symbol</td><td>2</td></tr>
</table>

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>7(a)</td><td>17</td><td>1</td></tr>
<tr><td>7(b)</td><td>One mark for correct fieldname and one mark for correct reason

PartNum

The data stored in this field is unique for each record</td><td>2</td></tr>
<tr><td>7(c)</td><td>
<table>
<tr><td>Field:</td><td>PartNum</td><td>Description</td><td>Cost</td><td>Quantity</td></tr>
<tr><td>Table:</td><td>AUDIOPARTS</td><td>AUDIOPARTS</td><td>AUDIOPARTS</td><td>AUDIOPARTS</td></tr>
<tr><td>Sort:</td><td></td><td></td><td>Descending</td><td></td></tr>
<tr><td>Show:</td><td>☑</td><td>☑</td><td>☑</td><td>☑</td></tr>
<tr><td>Criteria:</td><td></td><td></td><td></td><td><10</td></tr>
<tr><td>or:</td><td></td><td></td><td></td><td></td></tr>
</table>
One mark for correct field and table rows
One mark for sort row
One mark for show row
One mark for correct criteria</td><td>4</td></tr>
</table>