Cambridge IGCSE

Cambridge Assessment International Education
Cambridge International General Certificate of Secondary Education

CANDIDATE NAME

CENTRE NUMBER | | | | | |

CANDIDATE NUMBER | | | | |

2509236395

COMPUTER SCIENCE **0478/21**

Paper 2 Problem-solving and Programming **May/June 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 **13** printed pages and **3** blank pages.

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

The local high school uses buses to transport students to school. There are six bus routes labelled A to F. You have conducted a survey to analyse the punctuality statistics of these buses over a four-week period. The data from the survey are shown in the table:

	Punctuality table					
Day	**Bus A**	**Bus B**	**Bus C**	**Bus D**	**Bus E**	**Bus F**
Mon1	0	0	2	1	–1	0
Tue1	0	1	0	0	–1	–5
Wed1	0	0	–1	0	–1	–5
Thu1	2	0	–1	0	–2	–5
Fri1	2	1	–2	0	–4	–4
Mon2	4	2	–2	0	–10	–3
Tue2	0	0	–3	0	–2	–5
Wed2	3	0	–1	0	0	0
Thu2	4	0	0	0	0	0
Fri2	–2	0	0	0	0	0
Mon3	–5	1	–2	2	0	0
Tue3	0	0	0	0	1	–2
Wed3	0	0	1	0	2	–3
Thu3	3	0	1	0	–3	1
Fri3	4	2	1	0	1	1
Mon4	–1	0	1	0	1	1
Tue4	8	0	–1	0	3	0
Wed4	1	1	–1	0	–1	0
Thu4	1	0	2	0	0	–2
Fri4	–2	0	–2	0	0	–5

Positive numbers represent minutes early, negative numbers represent minutes late and 0 represents the bus having been on time.

Write and test a program or programs for the local high school.

- 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 data storage.
Using arrays set up a system to enable data for each bus route to be entered covering each day of a four-week period. It must be possible to enter the data supplied or your own set of data, using suitable prompts as necessary.

Task 2 – Working out the statistics.
Extend your program so that the following statistics for the four-week period may be calculated and output:
- the number of late arrivals for each bus route
- the average number of minutes late for each bus route
- the bus route with the highest number of days on which it was late
- the average number of minutes late for each bus route, using only data from days on which it was late

All the results should be displayed with appropriate annotation.

Task 3 – Checking specific days.
Extend the program as follows:
- Allow the user to input a specific day, for example Fri3, to be used for analysis of data.
- Find and display how many buses were late on this particular day.
- For each late bus, display the route label and how late the bus was on this particular day.

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

(i) State **one** array you used for **Task 1**. State the data type and purpose of this array.

Array .....

Data type .....

Purpose .....

.....

.....

[3]

(ii) State **one** variable you used for **Task 2** and **one** variable you used for **Task 3**. In each case, state the data type and purpose of the variable.

Task 2 variable name .....

Data type .....

Purpose .....

.....

.....

Task 3 variable name .....

Data type .....

Purpose .....

.....

.....

[6]

(b) Write an algorithm to show how you calculated and output the number of late arrivals and the average number of minutes late for each bus route (part of **Task 2**), using **either** pseudocode, programming statements **or** a flowchart.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

[6]

(c) Explain how your program uses the input in **Task 3** to only find the data for that specific day (part of Task 3). Any programming statements used in your answer must be fully explained.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [3]

(d) Explain how you would alter your program in **Task 1** to allow you to choose the number of weeks to enter data on bus arrival times.

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [2]

Section B

2 Describe each of the following data types used in programming. In each case, give an example of a piece of data to illustrate your answer. Each example must be different.

Char

.....

.....

.....

String

.....

.....

.....

Boolean

.....

.....

.....

[6]

3 **(a)** Give an example of a conditional statement using pseudocode.

.....

.....

.....

.....

..... [2]

(b) Describe the purpose of a conditional statement.

.....

.....

.....

..... [2]

Question 4 starts on page 10.

4 This section of program code may be used as a validation check.

```
1   PRINT "Input a value between 0 and 100 inclusive"
2   INPUT Value
3   WHILE Value < 0 OR Value > 100
4     PRINT "Invalid value, try again"
5     INPUT Value
6   ENDWHILE
7   PRINT "Accepted: ", Value
```

(a) Give a name for this type of validation check.

..... [1]

(b) Describe what is happening in this validation check.

.....

.....

.....

.....

.....

.....

.....

..... [2]

(c) Complete the trace table for this program code using the test data: 200, 300, –1, 50, 60

Value	OUTPUT

[3]

(d) Draw a flowchart to represent this section of program code.

[5]

5 The table, BEVERAGES, shows the number of calories in 100 ml of a range of popular beverages. It also shows the availability of these drinks in a can, a small bottle and a large bottle.

BevNo	BevName	Calories	Can	Small Bottle	Large Bottle
Bev01	Cola	40	Yes	Yes	Yes
Bev02	Lime	45	Yes	No	Yes
Bev03	Energy Drink 1	52	Yes	Yes	No
Bev04	Energy Drink 2	43	Yes	No	No
Bev05	Mango	47	Yes	No	Yes
Bev06	Lemon Iced Tea	38	Yes	No	Yes
Bev07	Lemonade	58	Yes	Yes	Yes
Bev08	Orange Juice	46	Yes	Yes	No
Bev12	Apple Juice	50	Yes	Yes	No
Bev15	Chocolate Milk	83	Yes	Yes	No

(a) Give a reason for choosing BevNo as the primary key for this table.

.....

..... [1]

(b) State the number of records shown in the table BEVERAGES.

..... [1]

(c) List the output that would be given by this query-by-example.

Field:	BevNo	BevName	Can	Small Bottle	Large Bottle	
Table:	BEVERAGES	BEVERAGES	BEVERAGES	BEVERAGES	BEVERAGES	
Sort:		Descending				
Show:	☑	☑	☐	☐	☐	☐
Criteria:			= "Yes"	= "Yes"	= "Yes"	
or:						

.....

.....

.....

..... [3]

(d) Complete the query-by-example grid to output a list showing just the names and primary keys of all the beverages with a calorie count greater than 45. The list should be in alphabetical order of names.

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

[4]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education
Cambridge International General Certificate of Secondary Education

COMPUTER SCIENCE **0478/21**

Paper 1 **May/June 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 May/June 2019 series for most Cambridge IGCSE™, Cambridge International A and AS Level and Cambridge Pre-U 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)(i)	Many correct answers, the name must be meaningful. Must relate to **Task 1** 1 mark per bullet point e.g.1 • Array `BusA` • Data type integer • Purpose storing the minutes late e.g.2 • Array `Day` • Data type string • Purpose storing the code for the day of the week	**3**
1(a)(ii)	Many correct answers, the name must be meaningful. Names shown are examples only. 1 mark per bullet point • Task 2 variable name `BusAMinsLate` • Data type real • Purpose used in calculation of average minutes late • Task 3 variable name `SearchDay` • Data type string • Purpose to input the day to be searched for	**6**
1(b)	1 mark for each point: MP1 Conditional statement to identify when a bus is late (punctuality < 0) MP2 Count the number of late days for at least one bus route MP3 Total the late minutes for at least one bus route MP4 Calculation of average minutes late 1 mark for each point (max **three** points): MP5 Initialisation of counting/totalling variables MP6 Iteration through days MP7 Checking all buses… MP8 Count late days and total minutes for all bus routes MP9 Output of number of late arrivals or average minutes late for at least one bus route… MP10 Output complete with all bus routes with late arrivals and average minutes late, with appropriate messages **Example algorithm on next page**	**6**

Question	Answer	Marks

1(b)	Example algorithm
	`CountA ← 0; CountB ← 0; CountC ← 0; CountD ← 0;`
	`CountE ← 0; CountF ← 0`
	`TotalA ← 0; TotalB ← 0; TotalC ← 0; TotalD ← 0;`
	`TotalE ← 0; TotalF ← 0`
	`FOR Days ← 0 to 19`
	`  IF BusA[Days] < 0`
	`  THEN`
	`  CountA ← CountA + 1`
	`  TotalA ← TotalA + BusA[Days]`
	`  ENDIF`
	`  IF BusB[Days] < 0`
	`  THEN`
	`  CountB ← CountB + 1`
	`  TotalB ← TotalB + BusBA[Days]`
	`  ENDIF`
	`  IF BusC[Days] < 0`
	`  THEN`
	`  CountC ← CountC + 1`
	`  TotalC ← TotalC + BusC[Days]`
	`  ENDIF`
	`  IF BusD[Days] < 0`
	`  THEN`
	`  CountD ← CountD + 1`
	`  TotalD ← TotalD + BusD[Days]`
	`  ENDIF`
	`  IF BusE[Days] < 0`
	`  THEN`
	`  CountE ← CountE + 1`
	`  TotalE ← TotalE + BusE[Days]`
	`  ENDIF`
	`  IF BusF[Days] < 0`
	`  THEN`
	`  CountF ← CountF + 1`
	`  TotalF ← TotalF + BusF[Days]`
	`  ENDIF`
	`NEXT`
	`PRINT "The number of late days for each bus route are: Bus A "CountA", Bus B "CountB", Bus C "CountC", Bus D ", CountD", Bus E ", CountE", Bus F "CountF`
	`PRINT "The average number of minutes late for each route are: Bus A "TotalA/20", Bus B "TotalB/20", Bus C "TotalC/20", Bus D ", TotalD/20", Bus E ", TotalE/20", Bus F "TotalF/20`	

Question	Answer	Marks
1(c)	Explanation of how the candidate's program performed the following: **Three** from: MP1 The input stored as a variable MP2 The method used to find the position of the day in the Day array that matches the input MP3 The array index is stored as a variable MP4 The index variable used as the array index for each bus array MP5 …and the contents of each array stored/output.	**3**
1(d)	**Two** from: MP1 Add a user input and prompt to enter the number of **weeks** required to record data on arrival times MP2 Store the user input for number of weeks as a variable MP3 Calculation to change number of weeks to number of days MP4 Replace the upper limit of the loop with a variable MP5 Increase the maximum size of the arrays to accommodate a higher number of weeks	**2**

Question	Answer	Marks
2	Many possible answers, those given are examples only. 1 mark for each correct description and 1 mark for each correct example **Char** Description: A single character (from the keyboard) Example: A / # / 2 **String** Description: An (ordered) sequence of characters Example: Hello world / #123?Y / 234 78963 **Boolean** Description: A data type with two possible values Example: TRUE / FALSE	**6**

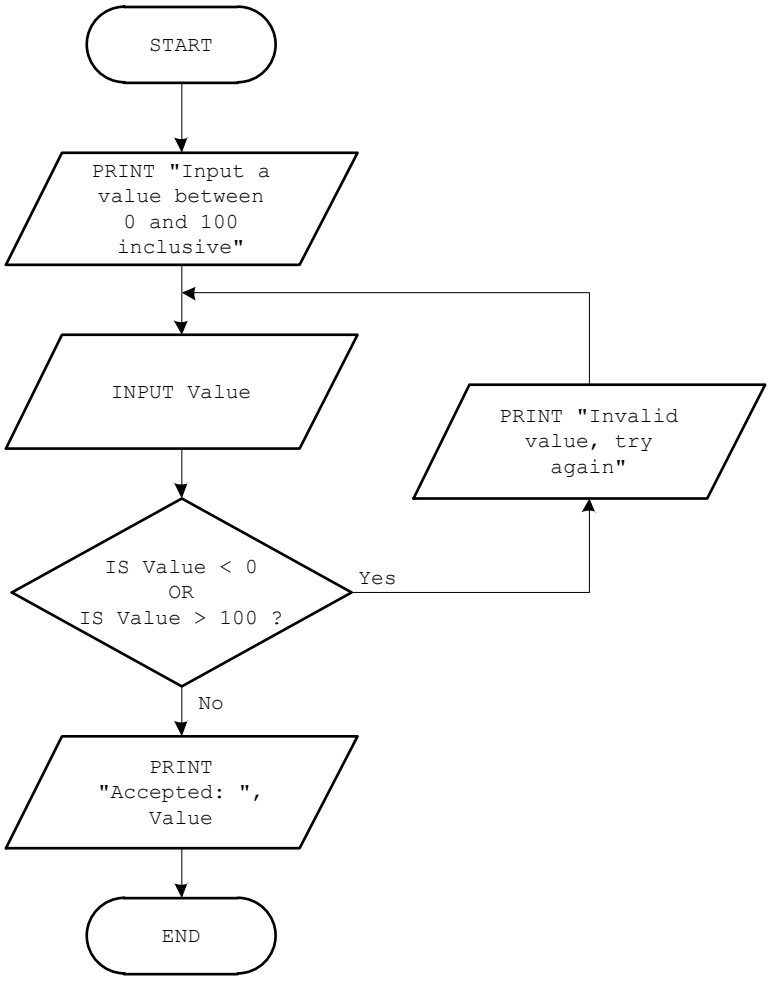
Question	Answer	Marks

3(a)	Many possible answers, those given are examples only.
	1 mark per bullet:
	• IF
	• Condition and outcome
	
	Example answer:
	`IF X < 0`
	`  THEN`
	`  PRINT "Negative"`
	`  ELSE`
	`  PRINT "Not negative"`
	`ENDIF`
	
	OR
	
	1 mark per bullet:
	• CASE
	• Condition and outcome
	
	Example answer:
	`CASE X OF`
	`  1: PRINT ("ONE")`
	`  2: PRINT ("TWO")`
	`  OTHERWISE PRINT ("Less than ONE or more than TWO")`
	`ENDCASE`
		2

3(b)	• To allow different routes through a program • dependent on meeting certain criteria	**2**

Question	Answer	Marks
4(a)	Range check	**1**
4(b)	**Two** from: • The entered number (`Value`) is being checked to see that it is not < 0 or not > 100 • If it is, it is rejected and the user has to enter another number / an error message is displayed • Otherwise the number is accepted, the word 'Accepted' is output along with the `Value`	**2**

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>4(c)</td><td>
<table>
<tr><th>Value</th><th>OUTPUT</th></tr>
<tr><td></td><td>Input a value between 0 and 100 inclusive</td></tr>
<tr><td>200</td><td>Invalid value, try again</td></tr>
<tr><td>300</td><td>Invalid value, try again</td></tr>
<tr><td>–1</td><td>Invalid value, try again</td></tr>
<tr><td>50</td><td>Accepted: 50</td></tr>
<tr><td></td><td></td></tr>
</table>
1 mark – Value column
1 mark – OUTPUT column first line
1 mark – OUTPUT column lines two to five
</td><td>3</td></tr>
</table>

Question	Answer	Marks
4(d)	 1 mark – Input prompt and input value 1 mark – Correct decision box labelled sufficiently (Yes/No) – allow 2 decision boxes 1 mark – Remaining outputs correct 1 mark – All connecting lines and arrows to be complete and correct 1 mark – Standard flowchart symbols used	**5**

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>5(a)</td><td>Each data value is unique</td><td>1</td></tr>
<tr><td>5(b)</td><td>10 records</td><td>1</td></tr>
<tr><td>5(c)</td><td>Bev07 Lemonade
Bev01 Cola

1 mark for each correct content
1 mark for each correct format
1 mark for correct order</td><td>3</td></tr>
<tr><td>5(d)</td><td>
<table>
<tr><td>Field:</td><td>BevNo</td><td>BevName</td><td>Calories</td><td></td><td></td><td></td></tr>
<tr><td>Table:</td><td>BEVERAGES</td><td>BEVERAGES</td><td>BEVERAGES</td><td></td><td></td><td></td></tr>
<tr><td>Sort:</td><td></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><td>☐</td></tr>
<tr><td>Criteria:</td><td></td><td></td><td>>45</td><td></td><td></td><td></td></tr>
<tr><td>or:</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

1 mark for correct Field row
1 mark for Table and Sort rows
1 mark for correct Show row
1 mark for correct Criteria rows</td><td>4</td></tr>
</table>