* 0000800000001 * DFD

CAMBRIDGE
International Education

# Cambridge IGCSE™

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

* 1 5 5 0 4 5 1 7 6 1 *

**COMPUTER SCIENCE** **0478/23**

Paper 2 Algorithms, Programming and Logic **May/June 2026**

**1 hour 45 minutes**

You must answer on the question paper.

No additional materials are needed.

**INSTRUCTIONS**

- Answer **all** questions.
- 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. Do **not** use correction fluid or tape.
- Do **not** write on any bar codes.
- Calculators must **not** be used in this paper.

**INFORMATION**

- The total mark for this paper is 75.
- 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 **16** pages. Any blank pages are indicated.

DC (WW/AR) 355606/4
© Cambridge University Press & Assessment 2026

**[Turn over**

GradeMax



**1** Tick (✓) **one** box to complete this sentence.

A standard method of solution used for adding together a set of values during iteration is

**A** counting. ☐

**B** searching. ☐

**C** sorting. ☐

**D** totalling. ☐

[1]

**2** Tick (✓) **one** box to identify a programming data type used to store any combination of letters and numbers.

**A** char ☐

**B** Boolean ☐

**C** string ☐

**D** real ☐

[1]

**3** **(a)** Identify the type of test data in each description.

**(i)** Data that falls within the accepted parameters of a validation check and is accepted.

..... [1]

**(ii)** Data that is the highest or lowest value to be accepted.

..... [1]

**(b)** Explain the purpose of a check digit and give an example of where it could be used.

purpose .....

.....

.....

.....

.....

example .....

[3]

© Cambridge University Press & Assessment 2026 

DO NOT WRITE IN THIS MARGIN DO NOT WRITE IN THIS MARGIN DO NOT WRITE IN THIS MARGIN DO NOT WRITE IN THIS MARGIN DO NOT WRITE IN THIS MARGIN

GradeMax



DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

**(c)** Write the pseudocode for an algorithm to perform a validation check and a verification check on a new password to meet the following criteria:

1 Input a new password.
2 If the password is less than 10 characters in length, ask for it to be input again until it is at least 10 characters in length.
3 If the password is at least 10 characters in length, ask for it to be input a second time.
4 If both inputs of the new password match, output a password accepted message.
5 If both inputs of the new password do **not** match, go back to step 1 and ask for the new password to be input for the first time.

Use appropriate input prompts and output messages to make the requirements clear to the user.

Do **not** declare any arrays or variables for this algorithm.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [6]

© Cambridge University Press & Assessment 2026

**4** This pseudocode algorithm is intended to store the names of up to 250 runners who finish a race. The names of the runners are stored in a one-dimensional (1D) array `Runners[]` and their time in seconds in a one-dimensional (1D) array `Times[]`. After the time for each runner has been entered into the array, the name of the runner and the time of the runner in minutes and seconds are displayed for verification.

Each runner takes 60 minutes or less to finish the race. A time of 60 minutes would be entered as 3600 seconds.

```
01 DECLARE Runners : ARRAY[1:250] OF STRING
02 DECLARE Times : ARRAY[1:250] OF INTEGER
03 DECLARE Index : INTEGER
04 DECLARE RunName : STRING
05 DECLARE RunTime : STRING
06 DECLARE RunMins : INTEGER
07 DECLARE RunSecs : INTEGER
08 FOR Index ← 1 TO 250
09     OUTPUT "Enter a runner's name and then time (in seconds) "
10     INPUT RunName
11     OUTPUT RunTime
12     RunMins ← DIV(RunTime, 60)
13     RunSecs ← DIV(RunTime, 60)
14     OUTPUT "Runner's Name: ", RunName
15     OUTPUT "Time: ", RunMins, " Minutes ", RunSecs, " Seconds"
16     Runners[Index] ← RunMins
17     Times[Index] ← RunTime
18 NEXT Index
```

**(a)** Identify the line numbers of **four** errors in the pseudocode and suggest corrections.

error 1 line number .....

correction .....

.....

error 2 line number .....

correction .....

.....

error 3 line number .....

correction .....

.....

error 4 line number .....

correction .....

.....

[4]

© Cambridge University Press & Assessment 2026 

DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

GradeMax



**(b)** The algorithm is intended to work for positive finish times that are 3600 seconds or less, but any number of seconds can currently be input.

**(i)** Write the pseudocode statements to show how you would change the algorithm so that only times in the correct range could be input. The time taken to finish the race needs to be input again until valid data is input.

Clearly state where in the original algorithm your new statements would be placed.

Do **not** rewrite the whole algorithm.

.....

.....

.....

.....

.....

.....

.....

.....

..... [5]

**(ii)** Identify **one** other possible problem with the algorithm.

.....

..... [1]

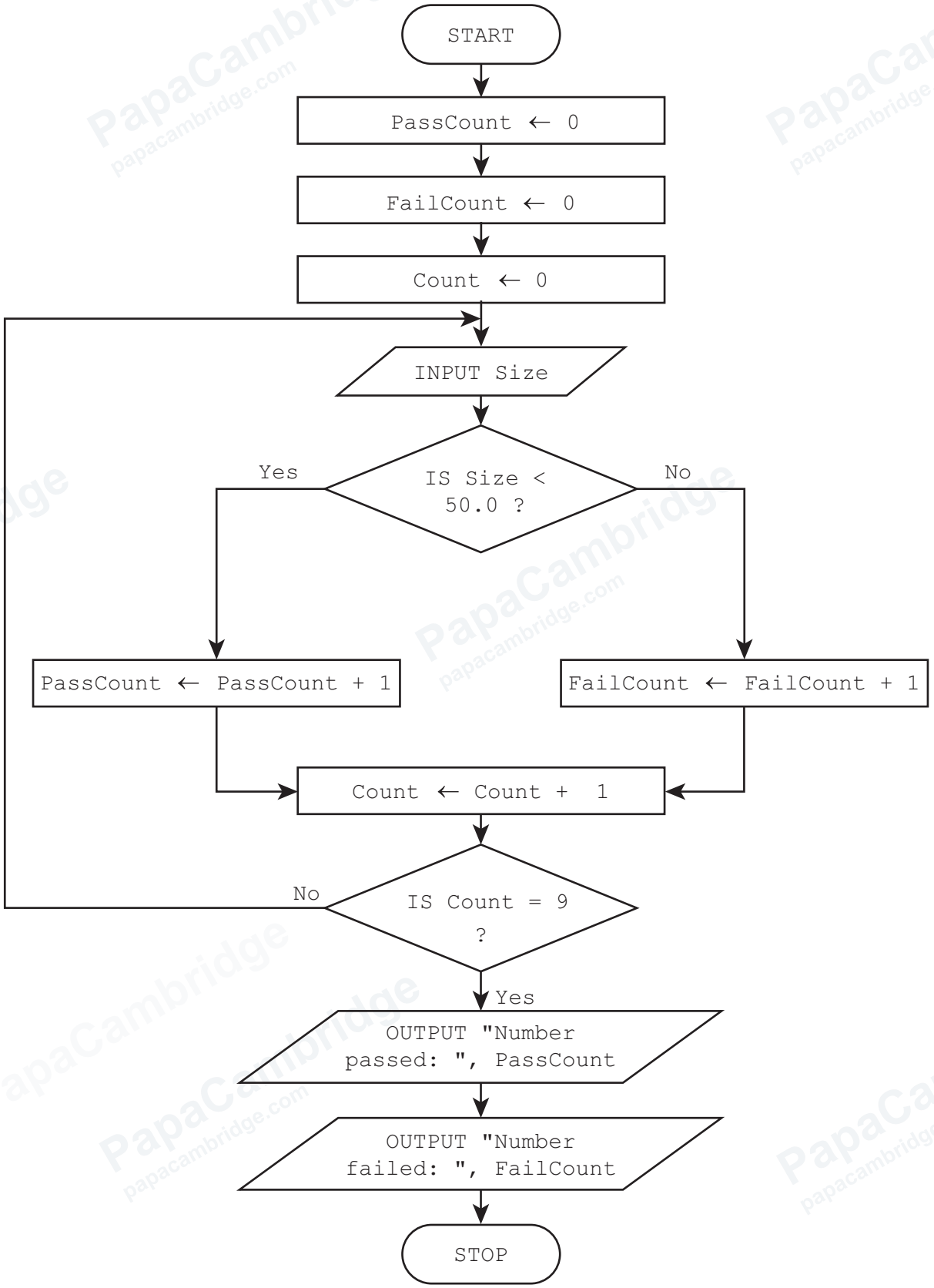
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

© Cambridge University Press & Assessment 2026 

GradeMax



**5** This flowchart represents an algorithm.




DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

© Cambridge University Press & Assessment 2026 

GradeMax



**(a)** Complete the trace table for the algorithm, using the input data:

49.8, 48.7, 62.5, 49.9, 30.0, 58.2, –12.5, 50.0, 26.7, –49.2, 58.9

| PassCount | FailCount | Count | Size | OUTPUT |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

[5]

**(b)** Describe how you would change the flowchart to prevent negative numbers from being input.

.....

.....

.....

..... [2]

© Cambridge University Press & Assessment 2026 

DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

GradeMax



**6** The string "`I learnt to program in computer science`" will be stored in the variable `Fact`

**(a)** Complete the pseudocode algorithm to:

- calculate and output the length of the string stored in the variable `Fact`
- output, in upper case, the string stored in the variable `Fact`

```
Fact ← "I learnt to program in computer science"
```

.....

.....

.....

..... [3]

**(b)** Write the pseudocode statements to use a string function to output the word `program` taken from the string stored in the variable `Fact`

Do **not** declare any variables.

.....

.....

.....

..... [2]

DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

© Cambridge University Press & Assessment 2026 

GradeMax



7 Consider the logic expression:

**X** is 1 if (**P** = NOT 1 AND **Q** = 1) XOR (**Q** = 1 NOR **R** = 1)

**(a)** Draw a logic circuit for the given logic expression.

Each logic gate must have a maximum of **two** inputs.

Do **not** simplify this logic expression.




[4]

**(b)** Complete the truth table from the given logic expression.

| P | Q | R | working space | X |
|---|---|---|---|---|
| 0 | 0 | 0 | | |
| 0 | 0 | 1 | | |
| 0 | 1 | 0 | | |
| 0 | 1 | 1 | | |
| 1 | 0 | 0 | | |
| 1 | 0 | 1 | | |
| 1 | 1 | 0 | | |
| 1 | 1 | 1 | | |

[4]

© Cambridge University Press & Assessment 2026


DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

GradeMax

**8** A database table `OPTION_SUBJECTS` stores some data about a number of subjects taken by students in a college.

| field | description |
|---|---|
| `SubjectCode` | a code such as CSIG01 to represent the subject and level |
| `SubjectName` | the name of the subject, for example computer science |
| `Level` | the level of the course, for example IGCSE or A Level |
| `CourseWork` | whether or not coursework is included as part of the assessment |
| `PracticalExams` | the number of practical exams taken during the course |
| `TheoryExams` | the number of theory exams taken during the course |
| `CourseLength` | length of the course, for example 1 or 2, representing number of years |
| `MinimumClassSize` | the minimum number of students required for the course to run, for example 10 |
| `MaximumClassSize` | the maximum number of students admitted to each class, for example 30 |

**(a)** Identify the most appropriate field in `OPTION_SUBJECTS` for a primary key.
Give **one** reason for your choice.

primary key field .....

reason .....

.....

[2]

**(b)** Identify **one** appropriate field from `OPTION_SUBJECTS` for each of the given data types.
State **one** reason for your choice in each case.

integer data type .....

reason .....

.....

Boolean data type .....

reason .....

.....

[4]

DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

© Cambridge University Press & Assessment 2026 

GradeMax



**(c)** Complete the structured query language (SQL) statement to list only the subject code, subject name and level for the minimum class size of less than 10. The results should be listed in order of subject code, from the lowest to the highest.

```
SELECT .....

FROM .....

WHERE .....

..... ;
```

[5]

**9** Outline **two** techniques that can be used to create a maintainable program.

1 .....

.....

.....

.....

.....

.....

2 .....

.....

.....

.....

.....

.....

[6]

DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

PapaCambridge papacambridge.com

GradeMax



**10** **Five** rivers show signs of pollution and are to be monitored daily for a period of 30 days.

The one-dimensional (1D) array `RiverNames[]` stores the names of the rivers.

The two-dimensional (2D) array `RiverStats[]` stores the daily measurement of biological oxygen demand (BOD) for each river.

The two-dimensional (2D) array `DaysCount[]` stores the number of days each river is classed as moderately polluted and the number of days each river is classed as highly polluted.

The position of the data for any river is the same in all three arrays. For example, the data in index 2 of `RiverStats[]` belongs to the river in index 2 of `RiverNames[]`

Write a program for the following requirements:

- Input and store the names of the rivers in the array `RiverNames[]`
- For each river, input and store in the array `RiverStats[]`, the daily BOD in milligrams per litre (mg / L) on each of the 30 days.
- Once all the inputs for the 30 days have been completed, the following statistics are calculated, and when appropriate, stored:
  - The average daily reading for each river, rounded to two decimal places.
  - The number of days in which the BOD reading was between 2 mg / L and 8 mg / L inclusive (moderate pollution) for each river.
  - The number of days in which the BOD reading was greater than 8 mg / L (high pollution) for each river.
- Output for each river, the average daily BOD readings, the number of days with moderate pollution and the number of days with high pollution.
- Find and output the name of the river that has the most days with high pollution.

The program does **not** need to identify more than one river with an equally high pollution reading.

You must use pseudocode or program code and add comments to explain how your code works.

Do **not** declare arrays, variables or constants for this algorithm; assume this has been done. You will need to initialise the arrays and any variables used for counting or totalling before they are used.

All inputs and outputs must contain suitable messages.

.....

.....

.....

.....

.....

.....

.....

.....

© Cambridge University Press & Assessment 2026 

DO NOT WRITE IN THIS MARGIN DO NOT WRITE IN THIS MARGIN DO NOT WRITE IN THIS MARGIN DO NOT WRITE IN THIS MARGIN DO NOT WRITE IN THIS MARGIN

GradeMax



DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

PapaCambridge papacambridge.com

GradeMax



.....

.....

.....

GradeMax





**BLANK PAGE**

DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

© Cambridge University Press & Assessment 2026 

GradeMax



**BLANK PAGE**

DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN
DO NOT WRITE IN THIS MARGIN

Permission to reproduce items where third-party owned material protected by copyright is included has been sought and cleared where possible. Every reasonable effort has been made by the publisher (Cambridge University Press & Assessment) to trace copyright holders, but if any items requiring clearance have unwittingly been included, the publisher will be pleased to make amends at the earliest possible opportunity.

To avoid the issue of disclosure of answer-related information to candidates, all copyright acknowledgements are reproduced online in our Copyright Acknowledgements Booklet. This is produced for each series of examinations and is freely available to download at www.cambridgeinternational.org after the live examination series.

Cambridge International Education is the name of our awarding body and a part of Cambridge University Press & Assessment, which is a department of the University of Cambridge.

© Cambridge University Press & Assessment 2026 