* 0000800000001 *

Cambridge Assessment International Education

Cambridge IGCSE™

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

* 7 2 4 1 1 0 2 3 8 2 *

COMPUTER SCIENCE **0478/23**

Paper 2 Algorithms, Programming and Logic **May/June 2025**

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 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 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.

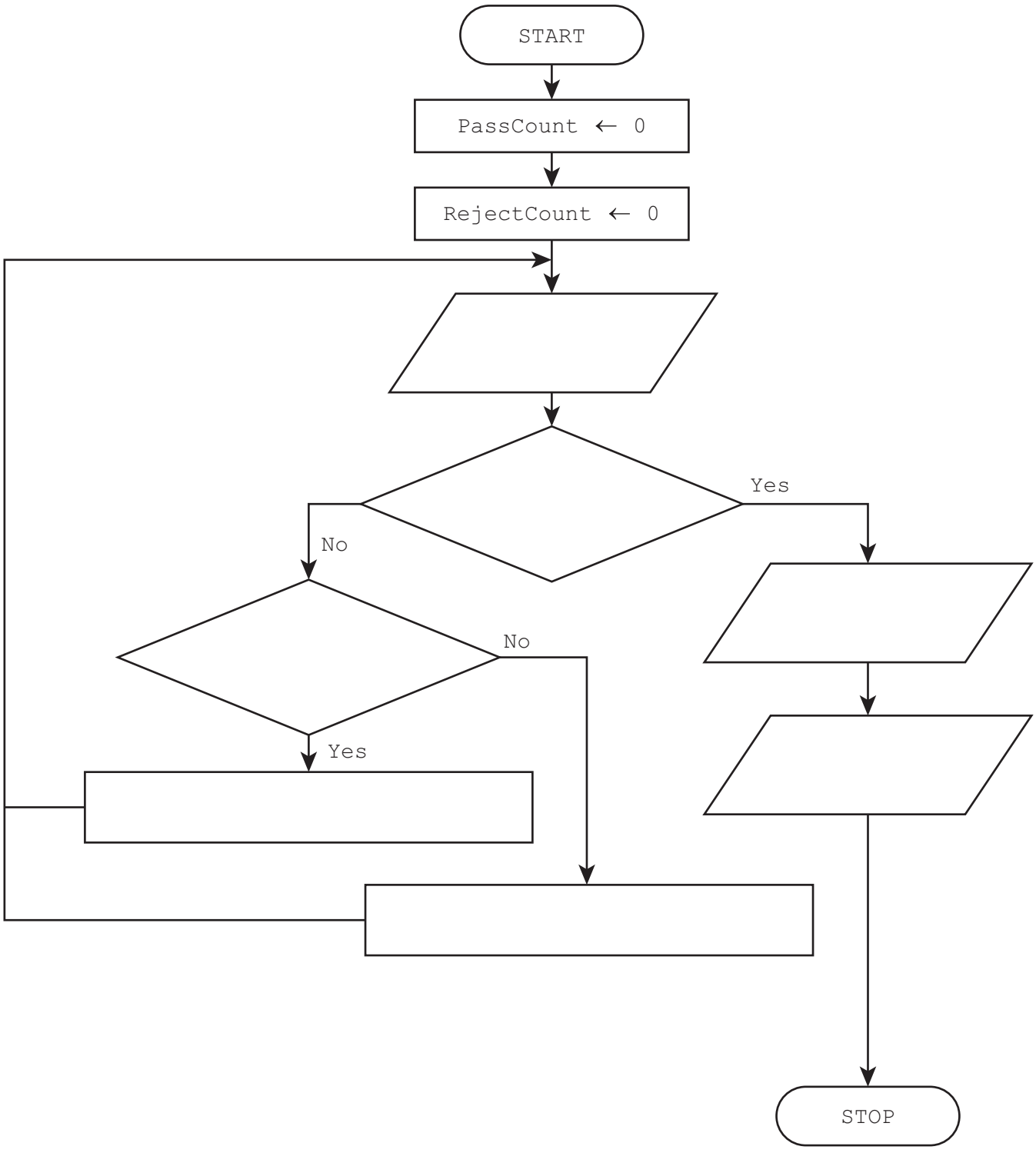
DC (DE/CB) 342017/5
© UCLES 2025

[Turn over

1 The flowchart shows an algorithm used for quality control on a biscuit production line.

The weight of each biscuit is entered into the variable `Weight`. If the weight of the biscuit is between 16.2 and 17.2 grammes, inclusive, the pass counter is incremented by 1. Otherwise, the reject counter is incremented by 1. The process continues until –1 is entered, after which, the totals for pass and reject counters are output.

(a) Complete this flowchart.



[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

(b) Explain how you would alter the flowchart in **1(a)** so that it calculates and outputs the average weight of only the biscuits that passed the weight check.

You may include program statements or flowchart symbols in your answer, but they must be fully explained.

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [4]

2 **Three** uses of arithmetic operators and **four** results are shown.

Draw **one** line from each use of arithmetic operator to the correct result.
Not all results will be used.

Use of arithmetic operator	Result
355 DIV 10	5
355 / 10	355
355 MOD 10	35
	35.5

[3]

DO NOT WRITE IN THIS MARGIN

3 The purpose of this pseudocode algorithm is to carry out a bubble sort to sort, in descending order, 1000 numbers stored in a one-dimensional (1D) array.

```
01 DECLARE Values : ARRAY[1:1000] OF REAL
02 DECLARE Index : CHAR
03 DECLARE Stop : BOOLEAN
04 DECLARE Hold : REAL
05 Stop ← FALSE
06 WHILE NOT Stop DO
07     Stop ← FALSE
08     FOR Index ← 1 TO 50
09         IF Values[Index + 1] > Values[Index]
10           THEN
11             Hold ← Values[Index]
12             Values[Index] ← Values[Index + 1]
13             Values[Index + 1] ← Hold
14             Stop ← FALSE
15         ENDIF
16     NEXT Index
17 NEXT Stop
```

(a) Identify the line numbers of **four** errors in the pseudocode and suggest a correction for each error.

Error 1 line number .....

Correction .....

.....

Error 2 line number .....

Correction .....

.....

Error 3 line number .....

Correction .....

.....

Error 4 line number .....

Correction .....

.....

[4]

DO NOT WRITE IN THIS MARGIN

(b) The swap section in lines 11 to 13 of the existing code are to be changed to a call statement for a procedure. This procedure will include two parameters representing the indexes of the array elements to be swapped.

(i) Complete the pseudocode for `PROCEDURE Swap`

```
PROCEDURE Swap(.....)

    DECLARE Hold : REAL

    Hold ← Values[.....]

    .....

    .....

ENDPROCEDURE
```

[4]

(ii) Write the pseudocode to transfer control to `PROCEDURE Swap`

.....

..... [2]

(c) Explain the difference between global variables and the variable declared in **3(b)(i)**.

.....

.....

.....

..... [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

4 **(a)** An input must be an integer between 10 and 95, inclusive.

(i) State the type of validation check to test if the input meets this requirement.

..... [1]

(ii) Suggest suitable abnormal and extreme test data for an integer between 10 and 95 inclusive, and give a reason for each of your choices.

Abnormal test data .....

Reason .....

.....

Extreme test data .....

Reason .....

.....

[4]

(b) A password system requires all new passwords to contain at least 12 characters.

(i) State the type of validation check to test that the password meets this requirement.

..... [1]

(ii) Write the pseudocode to allow a new password to be entered and to ensure that it contains at least 12 characters. The password is re-input until it meets this requirement. Use an appropriate error message for inputs that do **not** meet the requirement.

You do **not** need to declare any variables for this algorithm.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [5]

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
DO NOT WRITE IN THIS MARGIN

5 **(a)** A computer store uses a database table `COMPONENT` to store the stock details of a range of components needed to build computer systems.

Complete the table by giving appropriate data types for each field.

Only the data types: text, character, Boolean, integer, real, date/time are available in this database.

Field	Data type	Description
`ID`		unique identifier
`Name`		component name
`Description`		component description
`Price`		selling price of component to 2 decimal places
`NumberAvailable`		number in stock
`MinimumLevel`		level at which the component is reordered
`ReOrdered`		whether or not the component has been reordered
`DateOrdered`		the date the component was reordered

[4]

(b) Complete the structured query language (SQL) statement to list only the unique identifier, component name and reorder date of all the components that have been reordered.

```
SELECT .....

FROM .....

WHERE ..... ;
```

[4]

6 This pseudocode represents an algorithm.

```
DECLARE Store : ARRAY[1:100] OF CHAR
DECLARE Word : STRING
DECLARE Index : INTEGER
DECLARE Letter : CHAR
INPUT Word
FOR Index ← 1 TO LENGTH(Word)
    Letter ← SUBSTRING(Word, Index, 1)
    CASE OF Letter
      'A' : Letter ← ""
      'E' : Letter ← ""
      'I' : Letter ← ""
      'O' : Letter ← ""
      'U' : Letter ← ""
    ENDCASE
    Store[Index] ← Letter
NEXT Index
FOR Index ← 1 TO LENGTH(Word)
    IF Store[Index] <> ""
      THEN
        OUTPUT Store[Index]
    ENDIF
NEXT Index
```

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

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

			Store[]								
Word	Index	Letter	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	OUTPUT

[5]

(b) Describe the processes in the algorithm.

.....

.....

.....

.....

.....

..... [3]

DO NOT WRITE IN THIS MARGIN

7 **(a)** Draw the symbol for a NOR logic gate and complete the truth table for it.

NOR logic gate symbol:

Truth table:

A	B	Z
0	0	
0	1	
1	0	
1	1	

[2]

(b) Consider the rules given in the following table:

Input	Binary value	Description
L	0	dark
	1	light
M	0	no motion detected
	1	motion detected
S	0	no sound detected
	1	sound detected

A security system uses sensors to sound an alarm if an intruder is detected.

The alarm sounding is represented by the output **A = 1**.

The alarm sounds if it is dark and if either motion or sound or both motion and sound are detected.

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

(i) Write the logic expression for the security system.

A = ...

..... [2]

(ii) Complete the truth table for the security system.

L	M	S	Working space	A
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]

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

8 A collector needs a video library system for a collection of videos stored on 4K Blu-ray discs, standard Blu-ray discs, DVD, or as digital downloads.

The two-dimensional (2D) array `Video[]` is used to store, for each video item, the title, the format of the video (4K Blu-ray disc, standard Blu-ray disc, DVD, or digital download), the year it was released stored as a string and a storage code to represent where the video can be found in the collection, for example:

```
Video[1,1] is "Macbeth"
Video[1,2] is "digital download"
Video[1,3] is "2015"
Video[1,4] is "DG276"
```

The two-dimensional (2D) array `Results[]` is used to store the results of a search to find a specific video. The search uses the video title, and if the video is found, its data is copied from the array `Video[]` to the array `Results[]`. The search continues and if other videos with the same title are found, this data is also copied to the array `Results[]`. The search ends when the end of the data is reached.

Write a program that meets the following requirements:

- The video library array needs to be initialised with the null string (""), allowing for up to 10 000 records. The array `Results[]` must be initialised before every search with the null string and must be able to hold up to 20 search results with the same title.
- Create a menu to add a new video to the library to search for an existing video by title, or to stop, with validation of the input.
- When the user enters a new video, the following data is stored in the first available location of the relevant array:
  - video title
  - format (4K Blu-ray disc, standard Blu-ray disc, DVD, digital download)
  - year of release as string
  - storage code.
- Allow input of data for another video if required.
- When searching for an existing video, if a match is found, transfer all the data for that video into the array `Results[]`, and continue to search until all videos with the same title have been found.
- Output the results from the array `Results[]`
- If the video title is not found, output a suitable message.
- The system returns to the menu after completing the input or the output, until the user chooses to stop.

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

You do **not** need to declare any arrays, variables or constants. You may assume that this has already been done.

All inputs and outputs must contain suitable messages.

.....

.....

.....

.....

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
DO NOT WRITE IN THIS MARGIN

[15]

DO NOT WRITE IN THIS MARGIN

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

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 (UCLES) 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 the Cambridge Assessment International Education 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 Assessment International Education is part of Cambridge Assessment. Cambridge Assessment is the brand name of the University of Cambridge Local Examinations Syndicate (UCLES), which is a department of the University of Cambridge.