GradeMax

* 0000800000001 *





Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER



COMPUTER SCIENCE **9618/22**

Paper 2 Fundamental Problem-solving and Programming Skills **May/June 2025**

2 hours

You must answer on the question paper.

You will need: Insert (enclosed)

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- 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.
- You may use an HB pencil for any diagrams, graphs or rough working.
- 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.
- The insert contains all the resources referred to in the questions.

This document has **20** pages. Any blank pages are indicated.

DC (CJ) 342884/3
© UCLES 2025

[Turn over

GradeMax

Refer to the **insert** for the list of pseudocode functions and operators.

1 A program is being developed to control the production line in a factory.

(a) A number of different program life cycles are available for the development of a program.

(i) Explain the need for different program development life cycles.

.....

..... [1]

(ii) Coding is a stage in a program development life cycle.

State **one** consideration that would influence the choice of programming language.

.....

..... [1]

(b) The program has been in use for a number of months and adaptive maintenance is required.

(i) Give **three** reasons why adaptive maintenance may be required.

1 .....

2 .....

3 .....

[3]

(ii) As well as adaptive maintenance, other types of program maintenance may be needed.

Identify **one** other type of program maintenance.

..... [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

(c) Pseudocode has been used to design modules for the program to control the production line.

The table shows **four** valid pseudocode expressions.

Complete the table by giving the data type of the evaluated expression.

Expression	Data type
`RIGHT(MachineCode, 4)`	
`Speed * 2.5`	
`NOT Status`	
`IS_NUM(Check)`	

[4]

(d) A global array `Product` is used as part of the pseudocode design being developed to control the production line. `Product` is used to store the number of rejected items each day.

The following pseudocode statement is used to assign a value to an element of the array:

```
Product[x, y] ← 23
```

The lower and upper bound values are shown in the table:

Variable	Lower bound	Upper bound
x	0	99
y	0	9

(i) State the number of dimensions of `Product`

..... [1]

(ii) State the total number of elements in array `Product`

..... [1]

(iii) Give the pseudocode declaration for the array `Product`

..... [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

GradeMax

2 A program to calculate the pay of employees working for a company is being designed.

(a) Stepwise refinement has been used in the design of the program.

Describe stepwise refinement.

.....

.....

.....

..... [2]

(b) One of the program modules used to calculate employees' weekly bonus pay has been completed. The amount of bonus pay they receive is based on the number of hours worked and the value of sales made as shown in the table.

Bonus pay and value of sales are both in dollars.

Hours worked	Value of sales	Bonus pay
between 1 and 40 inclusive	2000 or less	0
between 1 and 40 inclusive	above 2000	50
above 40	2000 or less	10
above 40	above 2000	100

(i) The module is tested using white-box testing.

State **two** tests, using valid data, that can be used to test different paths through the program.

Test one:

Hours worked .....

Value of sales .....

Bonus pay .....

Test two:

Hours worked .....

Value of sales .....

Bonus pay .....

[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

GradeMax

(ii) The program to calculate pay uses a number of modules which are each called from different places in the program.

The program is to be tested using stub testing before all the program modules have been completed.

Describe stub testing.

.....

.....

.....

..... [2]

(iii) The program has been completed and compiles successfully.

The program is tested using black-box testing.

Identify and describe **one** type of error that black-box testing could detect.

Type of error .....

Description .....

.....

.....

[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

GradeMax

3 An algorithm is designed to generate and output two unique random integers. Each integer value is between –10 and 10 inclusive.

If both integers output are negative, a third random integer between 30 and 35 inclusive will be generated and output.

Write pseudocode for this algorithm.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [8]

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

4 An algorithm will:

- input 100 integer values, one value at a time
- store the first value input into the first location of the array `Number`
- store the next input value in the next unused location of the array `Number`
- output the contents of `Number` array in the opposite sequence to that in which the values were input.

Complete the program flowchart to represent the algorithm.

Variable declarations are **not** required.

START

END

[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

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

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

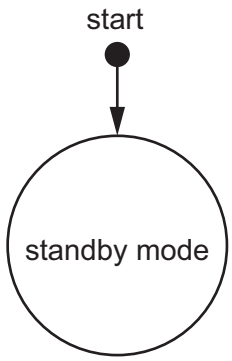
5 An automated digital camera system is used to take a sequence of pictures of animals in a garden. During the design of the system, a state-transition diagram is produced.

The table details the states in the automated digital camera system along with the events which cause the states to change.

The system starts in standby mode. The sequence of pictures is taken when in active mode.

Current state	Event	Next state
standby mode	turn on	detect mode
detect mode	turn off	standby mode
detect mode	movement detected	active mode
active mode	20 seconds elapsed	sequence complete
active mode	turn off	standby mode
sequence complete	time saved	detect mode

(a) Complete the state-transition diagram for the automated digital camera system.



[4]

GradeMax

(b) At the end of each sequence of pictures, the time is saved as a string in the format `<HH><MM><SS>` where:

- `HH` represents the hours using two digits
- `MM` represents the minutes using two digits
- `SS` represents the seconds using two digits.

For example:

- `"081230"` is stored to represent the time 8:12:30, in the morning
- `"152235"` is stored to represent the time 15:22:35, in the afternoon.

Each string is stored on a new line in the text file `TimeTaken.txt`

An algorithm is required to process the content of the text file `TimeTaken.txt` once it has been transferred to the computer.

For each hour when pictures are taken, output a suitable message showing the hour and the total number of sequences of pictures taken within that hour.

Example outputs:

```
Hour : 15  Total : 32
Hour : 18  Total : 1
```

Write pseudocode for this algorithm.

Assume the text file `TimeTaken.txt` contains at least **one** line.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

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]

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 Two arrays `Data` and `Pointer` are accessed by the procedure `Place()`

`Data` and `Pointer` are both global arrays of type `INTEGER`

The contents of these two arrays are shown:

	Data
1	1018
2	1007
3	1010
4	1056
5	1092
6	1062
7	1034
8	0
9	0
10	0

	Pointer
1	7
2	3
3	1
4	6
5	-1
6	5
7	4
8	9
9	10
10	-1

Study the pseudocode:

```
PROCEDURE Place(Value : INTEGER, Start : INTEGER, Unused : INTEGER)
   DECLARE New, Current, Last : INTEGER
   CONSTANT NullPointer = -1
   New ← Unused
   Last ← NullPointer
   Current ← Start
   WHILE Current <> NullPointer AND Data[Current] < Value
      Last ← Current
      Current ← Pointer[Current]
   ENDWHILE
   Pointer[New] ← Pointer[Last]
   Pointer[Last] ← New
   Data[New] ← Value
ENDPROCEDURE
```

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) (i) Complete the trace table below by dry running the procedure `Place()` when it is called by the statement:

```
CALL Place(1043, 2, 8)
```

Value	Start	Unused	New	Last	Current

[4]

(ii) Complete the diagram showing the contents of the global arrays `Data` and `Pointer` after the procedure `Place()` has run to completion when called as shown in part **(a)(i)**.

	Data
1	1018
2	1007
3	1010
4	1056
5	1092
6	1062
7	1034
8	
9	
10	

	Pointer
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

[3]

(b) The operation carried out by procedure `Place()` together with the arrays form part of the implementation of an Abstract Data Type (ADT).

Identify the ADT and state the operation carried out by procedure `Place()`

.....

.....

.....

..... [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

GradeMax

7 A program is being developed to implement a customer loyalty scheme for a coffee shop.

Each customer has a unique customer ID starting at `10001` with this value increasing by one each time a new customer joins the loyalty scheme.

For example, the third customer who joins the loyalty scheme is given the customer ID `10003`

The loyalty scheme is limited to 1000 customers.

A customer is awarded a loyalty point every time they buy a coffee.

The programmer has decided to use a global 2D array `Loyalty` of type `INTEGER`. The array `Loyalty` is made up of 1000 rows and 2 columns. Each row relates to one customer; column 1 contains the unique customer ID and column 2 contains the number of customer loyalty points.

Rows in the array `Loyalty` that are not currently being used have the value of Column 1 set to `99999`

The array is sorted in ascending order by customer ID.

The programmer has defined a program module:

Module	Description
`FindCustomer()`	• called with parameter of type `INTEGER` representing a customer ID • searches the `Loyalty` array for this customer ID • the search will stop as soon as the customer ID is found • the search should efficiently deal with the situation when the customer ID is **not** stored in the `Loyalty` array • if the customer ID is found, return an integer value representing the loyalty points, otherwise return `-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

GradeMax

(a) Write efficient pseudocode for module `FindCustomer()`

[8]

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) A customer can claim a free coffee for every 11 loyalty points.

The programmer has defined a second program module:

Module	Description
`PointsReport()`	• output the customer ID for each customer who has 11 or more loyalty points • output the average loyalty points for all customers in the `Loyalty` array along with an appropriate message

Write efficient pseudocode for module `PointsReport()`

Assume the array contains the data for at least **one** customer.

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

.....

.....

.....

.....

.....

.....

.....

..... [7]

(c) The programmer decides to amend the customer ID; it will be stored as a `STRING` instead of an `INTEGER`. This means that the 2D array `Loyalty` can no longer be used.

(i) Explain why the 2D array `Loyalty` can no longer be used.

.....

..... [1]

(ii) Explain how a 1D array could be used to store both the loyalty points and the amended customer ID.

.....

.....

.....

..... [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

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

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

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 (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.

© UCLES 2025