GradeMax

* 0000800000001 * DFD

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

2003545536

COMPUTER SCIENCE **9618/21**

Paper 2 Fundamental Problem-solving and Programming Skills **October/November 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 (WW/CB) 343247/3
© UCLES 2025

[Turn over

GradeMax

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

1 A program is being developed to meet a particular customer requirement.

(a) The program contains these variables:

Variable	Data type
MyChar	CHAR
MyString	STRING
MyInt	INTEGER
MyDOB	DATE

Complete the table by filling in the gaps using functions and/or operators from the **insert**.

Each expression must be valid.

Expression
MyString ← 'X' ..... MyString
MyChar ← ..... ("ABCD", ... ,.....)
MyString ← ..... (..... (MyDOB))
MyInt ← ..... (..... (MyString) / 2)

[4]

(b) Different test methods will be used at different stages of the program development.

Complete the table by identifying the test method that matches the test description.

The first row has been completed for you.

Test description	Test method
carried out as soon as a program module has been coded	alpha
carried out as program modules are being combined	
completed by the developers without referring to the code	
completed by the customer	

[3]

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) During the alpha testing stage, an Integrated Development Environment (IDE) is used to help locate an error that has been identified. The IDE report window feature is used to examine the values assigned to variables.

Explain how **two** other IDE features are used together with the report window feature to help locate the error.

.....

.....

.....

.....

.....

..... [3]

GradeMax

2 `Data` is a global 1D array containing 30 elements of type `STRING`

An algorithm will output:

- all non-blank elements (elements that do **not** contain an empty string)
- the final total of the number of elements output.

Complete the program flowchart to represent the algorithm:

START

END

[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

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

3 A program is needed to manage individual rentals in a car-hire business.

The data items for each rental will be held in a record structure of type `RentalRecord`
The programmer has started to define the items that will be needed:

Item	Example value	Comment
`RentalID`	`"AB1234"`	a unique alpha-numeric value
`CarID`	`241`	a numeric value used as an array index
`DisCode`	`'S'`	a letter indicating the type of discount offered
`Start`	`13/06/2025`	when the rental starts
`Duration`	`7`	the number of days of the rental
`Completed`	`FALSE`	`TRUE` when the car is returned and the rental charge paid

(a) (i) Write pseudocode to declare the record structure for type `RentalRecord`

.....

.....

.....

.....

.....

.....

.....

..... [4]

(ii) A 1D array `Rental` containing 500 elements is used to store the data for all rental records.

Write pseudocode to declare the `Rental` array.

.....

..... [2]

(b) State **three** benefits of using an array of records to store the data for all rentals.

1 .....

.....

2 .....

.....

3 .....

..... [3]

GradeMax

4 A program contains a global 1D array `Number` consisting of 20 elements of type `REAL`

A procedure `Store()` will input a sequence of up to 20 real values, one value at a time. These values will be assigned to elements of the array using four steps:

Step 1: store the first value in the sequence in the first element of the array
Step 2: check each subsequent value input. If this value is larger than the previous value input, then assign the value to the next array element, otherwise go to **step 4**
Step 3: repeat from **step 2** unless the array is full
Step 4: output the count of the number of values stored in the array together with a suitable message.

(a) Complete the pseudocode for `Store()`

All variables used in the algorithm must be declared.

```
PROCEDURE Store()
```

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

```
ENDPROCEDURE
```

[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

(b) The requirements of the program change:

- the number of values in the sequence is unknown, but may be higher than 20
- the values will need to be accessed by another program as data.

The data will be stored in a text file instead of an array.

(i) Give **two** benefits of using a text file instead of an array.

1 .....

.....

2 .....

.....

[2]

(ii) State any change that will need to be made before each value is written to the file.

.....

..... [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

5 A program is being designed in pseudocode.

The program contains the following declaration for the global array `MyData`:

```
DECLARE MyData : ARRAY[1:10000] OF STRING
```

A function `FindFirst()` is written to search the array for a given string and to return the index of the first element where that string is found, or to return –1 if the string is **not** found.

The function is written in pseudocode as shown:

```
FUNCTION FindFirst(SearchString : STRING) RETURNS INTEGER
   DECLARE Index, FoundAt : INTEGER
   FoundAt ← -1
   FOR Index ← 1 TO 10000
      IF MyData[Index] = SearchString THEN // outer conditional clause
         IF FoundAt = -1 THEN              // inner conditional clause
            FoundAt ← Index
         ENDIF
      ENDIF
   NEXT Index
   RETURN FoundAt
ENDFUNCTION
```

(a) (i) Comment on why the programmer chose –1 as the initial value of `FoundAt`

.....

..... [1]

(ii) Explain the purpose of the **outer** conditional clause.

.....

.....

..... [1]

(iii) The **inner** conditional clause ensures that only the index of the **first** matching element, if any, is returned.

Explain how this clause works.

.....

.....

..... [1]

DO NOT WRITE IN THIS MARGIN

GradeMax

(b) The pseudocode does **not** use the most appropriate loop construct.

(i) Explain why this is **not** the most appropriate loop construct.

.....

.....

..... [1]

(ii) Suggest and justify a more appropriate loop construct that could be used.

Construct .....

Justification .....

.....

[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

6 Students are learning about a simple check digit method for data validation. In this method, a single check digit is appended to the end of an original number to give a new number.

The students are studying a method which:

- calculates the sum of all the digits in the original number
- uses integer division to calculate the remainder when the sum is divided by 10
- uses the remainder as the check digit
- appends the check digit to the original number, creating the new number.

For example:

original number	4162
sum of all digits	4 + 1 + 6 + 2 = 13
remainder when the sum is divided by 10 using integer division	3
new number	41623

The method described can be used to detect single-digit errors. For example, if the new number is incorrectly input as 41633, then the check digit does not match and the input will be rejected.

(a) An incorrect attempt was made to enter 41623. The **first two** digits were entered incorrectly; the **last three** digits were entered correctly.

When this number was tested, the check digit was found to be correct and the input was accepted.

Identify an example of the incorrect attempt to enter 41623 and explain why this number would **not** be rejected.

Number .....

Explanation .....

.....

[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

(b) A module `Generate()` will take an integer value representing an original number and return an integer value representing a new number which includes the check digit.

The original number is always at least three digits in length.

Write pseudocode for the module `Generate()`

Assume that the parameter is valid.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [7]

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

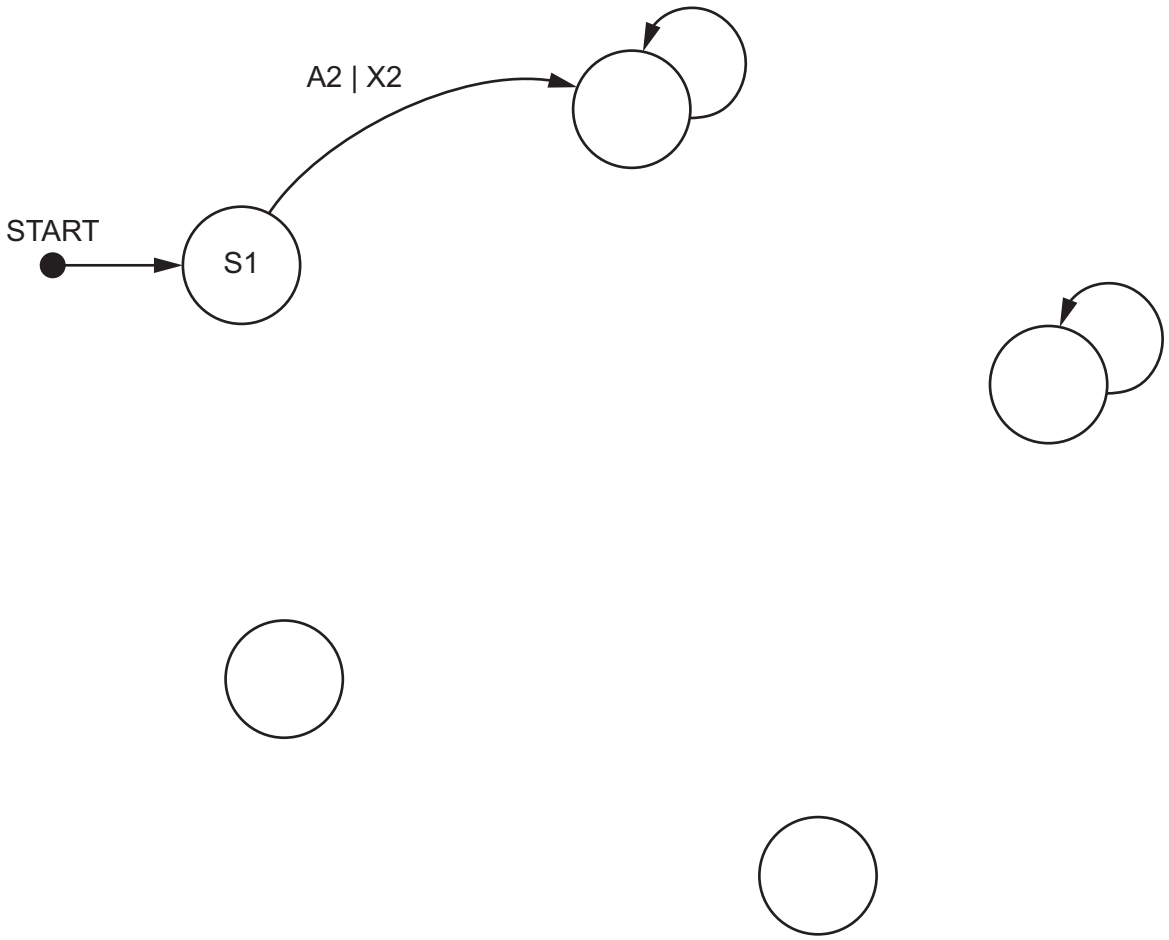
7 There are several different ways to express an algorithm during the design of a program.

(a) One part of the program contains an algorithm which is represented by a state-transition diagram.

The table shows the inputs, outputs and states for the algorithm:

Current state	Input	Output	Next state
S1	A2	X2	S3
S1	A1	X1	S2
S2	A4	X4	S5
S3	A1		S3
S3	A3	X3	S2
S3	A2	X4	S4
S4	A1	X1	S4
S4	A3		S2
S4	A4	X4	S5

Complete the state-transition diagram to represent the information given in the table.



[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

GradeMax

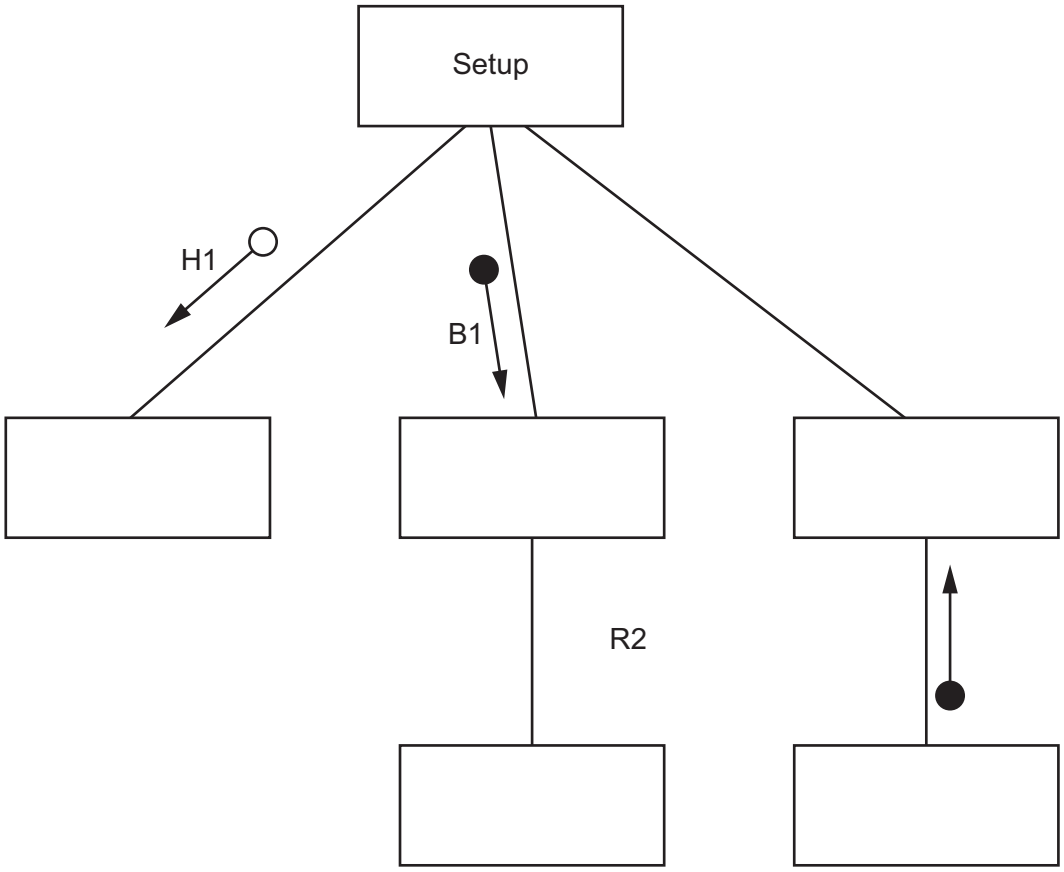
(b) A structure chart is used to document a different part of the program.

This part of the program contains six modules:

Pseudocode module header
`PROCEDURE Setup()`
`PROCEDURE Restart(H1 : STRING, C1 : INTEGER)`
`FUNCTION Modify(B1 : BOOLEAN) RETURNS INTEGER`
`PROCEDURE Final(T1 : INTEGER)`
`PROCEDURE Update(BYREF R2 : STRING)`
`FUNCTION Confirm() RETURNS BOOLEAN`

Module Setup will repeatedly call **three** of the modules.

Complete the structure chart to document the information given in the table.



[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

GradeMax

8 A program is being developed to manage student book loans from a college library. Students may borrow up to five books at a time from the library.

The programmer has defined a record type to define each loan.

The record data items are:

Data item	Data type	Comment
`StudentID`	`STRING`	the unique ID of the student who has borrowed the book
`BookID`	`STRING`	the unique ID of the book being borrowed
`OnLoan`	`BOOLEAN`	`TRUE` if the book has **not** been returned

The programmer has defined a global array `Loan` to store 5000 loan records.

There are more elements in the array than books in the library. Unused elements have the `StudentID` set to an empty string. These may occur anywhere in the array.

The programmer has defined the first program module:

Module	Description
`OKToBorrow()`	• called with a parameter of type `STRING` representing a `StudentID` • search the array for loan records for the specified student • output a suitable message to say whether the student may, or may not, borrow another book

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 the module `OKToBorrow()`

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [7]

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 second module is defined:

Module	Description
`ReturnBook()`	• called with two parameters of type `STRING` representing a `StudentID` and a `BookID` • searches the array for the relevant loan record • when found, sets `OnLoan` to `FALSE` and returns `TRUE` • if a loan record is **not** found, or the book has already been returned, then returns `FALSE`

Write efficient pseudocode for the module `ReturnBook()`

Assume that each student is only allowed to borrow each book only once. That means that there will be no more than **one** loan record for a given combination of student and book.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

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

.....

.....

.....

.....

.....

..... [8]

(c) It is decided to introduce a system of fines for books that have been borrowed for too long.

Two new requirements are defined:

1. Each loan has a maximum length, represented as a number of days.
2. Each book in the library will be assigned one of three categories. Each category has a different maximum loan length.

Record structure and program design changes are needed to meet these two requirements.

Outline the changes that are necessary to meet the **two** requirements.

.....

.....

.....

.....

.....

.....

..... [2]

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