GradeMax

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER | | | | | |

CANDIDATE NUMBER | | | | |

1621126030

COMPUTER SCIENCE **9608/22**

Paper 2 Fundamental Problem-solving and Programming Skills **May/June 2016**

2 hours

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.
No marks will be awarded for using brand names of software packages or hardware.

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 75.

This document consists of **18** printed pages and **2** blank pages.

DC (NF/FD) 125122
© UCLES 2016

CAMBRIDGE International Examinations

[Turn over

There is an **Appendix** on page 18. Some questions will refer you to this information.

1 The items in the table below are individual statements in a generic programming language.

For the built-in functions list, refer to the **Appendix** on page 18.

(a) (i) Show what type of programming construct each statement represents.

Complete the table by putting a tick (✓) in the appropriate column for each item.

Item	Statement	Selection	Iteration	Assignment
1	`MyScore = 65`			
2	`FOR IndexVal = 0 TO 99`			
3	`MyArray[3] = MID(MyString,3,2)`			
4	`IF MyScore >= 70 THEN`			
5	`ENDWHILE`			
6	`ELSE Message = "Error"`			

[6]

© UCLES 2016

(ii) State the purpose of each statement in the table in **part (a)(i)**.

Do **not** use mathematical symbols in your descriptions.

Item	Purpose of statement
1	....
2	....
3	....
4	....
5	....
6	....

[6]

(iii) Evaluate the following expressions when `MyString` has the value `"Adaptive Maintenance"`.

Expression	Result
`'D' & RIGHT(MyString, 4)`	
`LEFT(RIGHT(MyString, 7), 3)`	

[2]

2 A team is designing a software system to monitor temperature in a process. To do this, the system needs to sample the temperature repeatedly. If the temperature exceeds a given threshold value, an alarm will sound.

The system is to be software-based. It will include a subroutine, `SampleTemp`, which samples the temperature and sets the alarm state to either ON or OFF.

The initial design stage will produce a prototype of `SampleTemp` with a user interface.
The structured English for this is:

1. IF the temperature does not exceed threshold value, SET alarm state to OFF
2. INPUT threshold value (to two decimal places)
3. INPUT sensor value (a whole number in the range 0 to 100)
4. MULTIPLY sensor value by conversion factor 1.135 to give temperature
5. IF temperature exceeds threshold value SET alarm state to ON
6. IF temperature exceeds threshold value OUTPUT message “Temperature Alarm”
7. IF temperature does not exceed threshold value OUTPUT message “Temperature OK”

(a) The procedure needs four variables. Complete the identifier table below for these variables.

Identifier	Data type	Description
`AlarmState`		....
`SensorValue`		....
`ThresholdValue`		....
`Temperature`		....

[4]

GradeMax

(b) Write the **pseudocode** equivalent of the structured English. Use the identifiers from the table in **part (a)**.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[6]

GradeMax

Question 3 begins on page 7.

© UCLES 2016

3 A string encryption function is needed. The encryption uses a simple character-substitution method.

In this method, a new character substitutes for each character in the original string. This will create the encrypted string.

The substitution uses the 7-bit ASCII value for each character. This value is used as an index for a 1D array, `Lookup`, which contains the substitute characters.
`Lookup` contains an entry for each of the ASCII characters. It may be assumed that the original string and the substitute characters are all printable.

For example:

- 'A' has ASCII value 65
- Array element with index 65 contains the character 'Y' (the substitute character)
- Therefore, 'Y' substitutes for 'A'
- There is a different substitute character for every ASCII value

The programmer writes a function, `EncryptString`, to return the encrypted string. This function will receive two parameters, the original, `PlainText` string and the 1D array.

(a) The first attempt at writing the pseudocode for this function is shown below.

Complete the pseudocode.

For the built-in functions list, refer to the **Appendix** on page 18.

```
FUNCTION EncryptString(.............................) RETURNS STRING

   DECLARE ..................... , ...................... : CHAR

   DECLARE OldCharValue : ....................................

   DECLARE n : INTEGER

   DECLARE OutString : STRING


   ............................. //initialise the return string


   //loop through PlainText to produce OutString

   FOR n ← 1 TO ................. //from first to last character

      OldChar ← ...................//get next character

      OldCharValue ← ..............//find the ASCII value

      NewChar ← ...................//look up substitute character

      ..............................//concatenate to OutString

   ENDFOR


   ..........................................................
ENDFUNCTION
```

[10]

(b) Additional code needs to be written to allow the user to change some of the characters in the array `Lookup`.

The user will input:

- the array start position
- the number of elements to change
- each new substitute character

At the end, the program will finally output a confirmation message.

The first version of the algorithm is represented by the flowchart on the following page.

(i) Write **program code** to declare the array `Lookup`.

Programming language .....

.....

.....

.....[2]

(ii) Write **program code** to implement the flowchart design.

In addition to the `Lookup` array, assume that the following variables have been declared:

```
StartPos, NumToChange, n, NewChar
```

Programming language .....

.....

.....

.....

.....

.....

.....

.....

.....

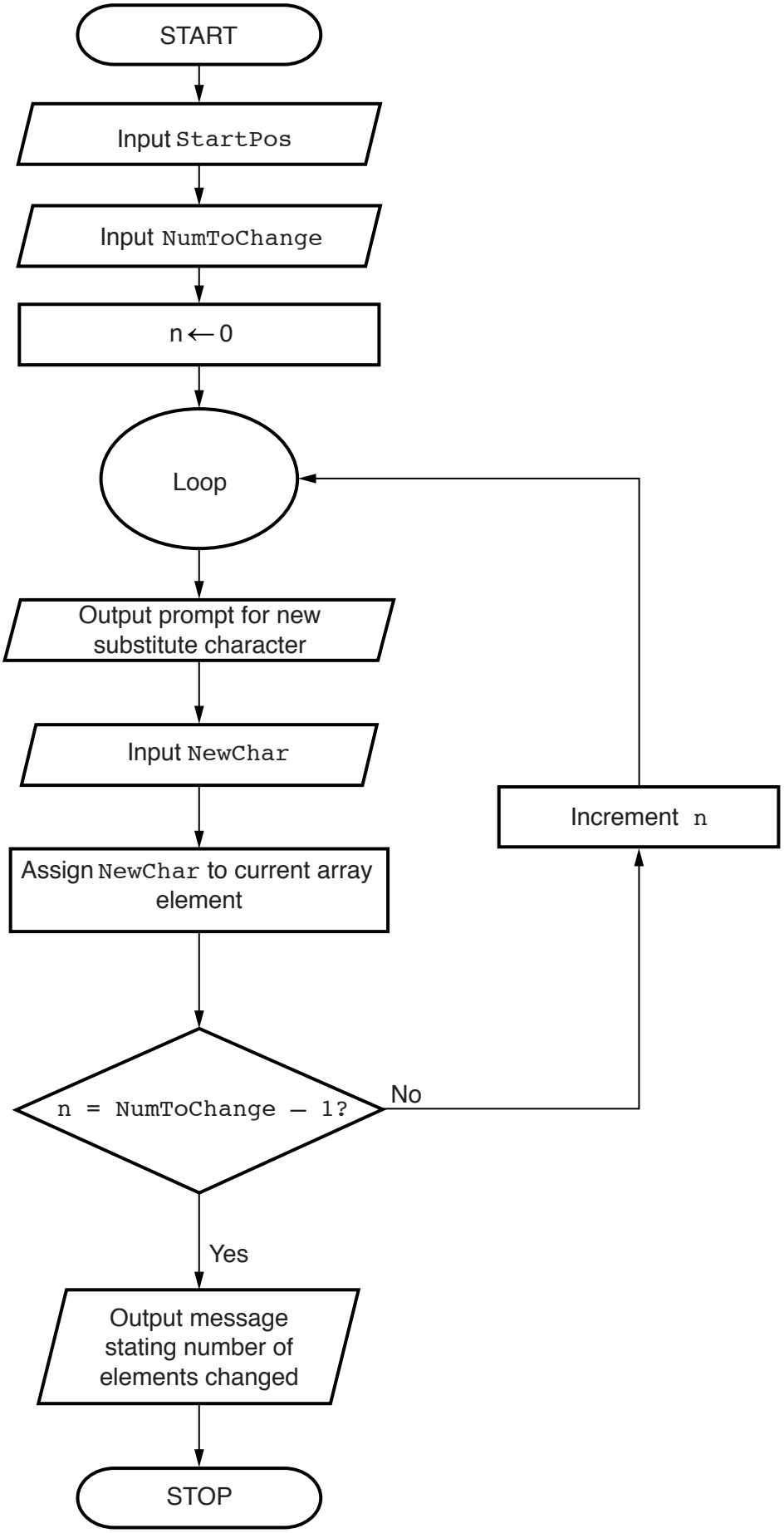
.....

.....

.....

.....[6]

9


START
Input StartPos
Input NumToChange
n ← 0
Loop
Output prompt for new substitute character
Input NewChar
Assign NewChar to current array element
n = NumToChange – 1?
No
Increment n
Yes
Output message stating number of elements changed
STOP

GradeMax

4 **(a)** Structured programming involves the breaking down of a problem into modules.

Give **two** reasons why this is done.

1 .....

.....

2 .....

.....

[2]

(b) A team needs to write a program to implement an online shopping system. Customers will access the program via a website.

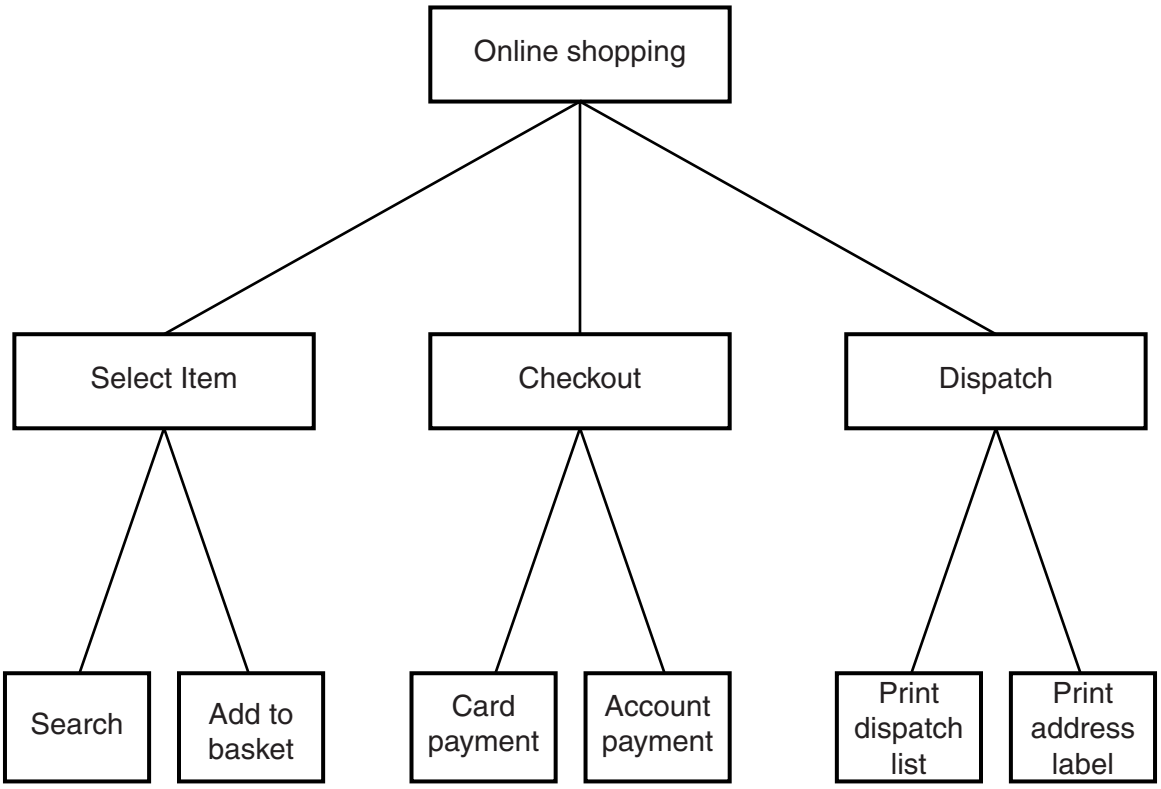
Customers can search for items before adding them to a virtual shopping basket. When they have finished shopping, they pay for the items. The program provides output for the dispatch of the items.

Some of the key features of the system are as follows:

- a customer can add many items to the shopping basket
- payment may be either by credit or debit card, or by adding to a customer account
- the shop may dispatch the items in one or more packages

The structure chart below shows the program modules only.

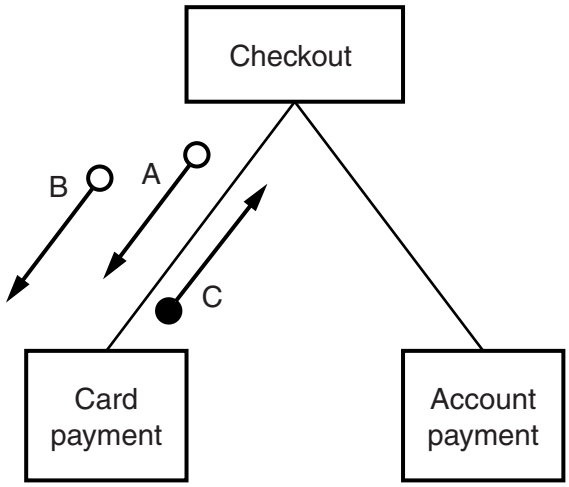
(i) Draw on the chart, the symbols that represent the key features listed in **part (b)** above.



[3]

© UCLES 2016

(ii) A section of the chart in **part (b)(i)** is shown below. It is to show the parameters passed between the Checkout and Card payment modules.



Name the three data items corresponding to the arrows.

Arrow	Data item
A	
B	
C	

[3]

© UCLES 2016

GradeMax

Question 5 begins on page 13.

GradeMax

5 Toni has a large collection of jazz CDs that are stored in different places. She wants to record where the CDs are stored. She decides to write a program to do this.

The program must store the data in a file, `MyMusic`.

(a) (i) Why is a file needed?

.....

.....[1]

(ii) `MyMusic` is a text file with the data for each CD as one line of text.

Data for a typical CD are:

Title: Kind of Green
Artist: Miles Coltrane
Location: Rack1-5

The line will be formed by concatenating the three data items.

For the example above, the line stored will be:

```
Kind of GreenMiles ColtraneRack1-5
```

Describe a problem that might occur when organising the data in this way.

.....

.....

.....

.....

Describe a possible solution.

.....

.....

.....

.....
[4]

© UCLES 2016

[Turn over

GradeMax

(b) Toni must input the data into the file for all of her CDs.

A procedure, `InputData`, is needed to do this.

Toni designs the procedure and chooses the following identifiers:

Identifier	Data type
`CDTitle`	`STRING`
`CDArtist`	`STRING`
`CDLocation`	`STRING`

The procedure repeatedly performs the following steps:

- input a CD title (A rogue value of “##” is to be used to end the input)
- input the artist
- input the location
- create the text line
- write the text line to the file

When the rogue value is encountered the file is closed.

© UCLES 2016

15

Write **program code** for the procedure `InputData`.

Visual Basic and Pascal: You should include declaration statements for variables.
Python: You should show a comment statement for each variable used with its data type.

Programming language .....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[8]

GradeMax

6 A string-handling function has been developed. The pseudocode for this function is shown below.

For the built-in functions list, refer to the **Appendix** on page 18.

```
FUNCTION SSM(String1, String2 : STRING) RETURNS INTEGER
   DECLARE n, f, x, y : INTEGER

   n ← 0
   f ← 0

   REPEAT
      n ← n + 1
      x ← n
      y ← 1
      WHILE MID(String1, x, 1) = MID(String2, y, 1)

         IF y = LENGTH(String2)
            THEN
               f ← n
            ELSE
               x ← x + 1
               y ← y + 1
         ENDIF

      ENDWHILE

   UNTIL (n = LENGTH(String1)) OR (f <> 0)

   RETURN f

ENDFUNCTION
```

(a) Complete the trace table below by performing a dry run of the function when it is called as follows:

```
SSM("RETRACE", "RAC")
```

n	f	x	y	MID(String1, x, 1)	MID(String2, y, 1)
0	0				

[6]

© UCLES 2016

(b) (i) Describe the purpose of function `SSM`.

.....

.....

.....

.....[2]

(ii) One of the possible return values from function `SSM` has a special meaning.

State the value and its meaning.

Value .....

Meaning .....
[2]

(iii) There is a problem with the logic of the pseudocode. This could generate a run-time error.

Describe the problem.

.....

.....

.....

.....[2]

GradeMax

Appendix

Built-in functions

In each function below, if the function call is not properly formed, the function returns an error.

`MID(ThisString : STRING, x : INTEGER, y : INTEGER) RETURNS STRING`

returns the string of length `y` starting at position `x` from `ThisString`

Example: **`MID("ABCDEFGH", 2, 3)`** will return string **`"BCD"`**

`LEFT(ThisString : STRING, x : INTEGER) RETURNS STRING`

returns the leftmost `x` characters from `ThisString`

Example: **`LEFT("ABCDEFGH", 3)`** will return string **`"ABC"`**

`RIGHT(ThisString: STRING, x : INTEGER) RETURNS STRING`

returns the rightmost `x` characters from `ThisString`

Example: **`RIGHT("ABCDEFGH", 3)`** will return string **`"FGH"`**

`ASC(ThisChar : CHAR) RETURNS INTEGER`

returns the ASCII value of character `ThisChar`

Example: **`ASC('W')`** will return **`87`**

`LENGTH(ThisString : STRING) RETURNS INTEGER`

returns the integer value representing the length of string `ThisString`

Example: **`LENGTH("Happy Days")`** will return **`10`**

String operator

`&` operator

concatenates (joins) two strings

Example: **`"Summer" & " " & "Pudding"`** produces **`"Summer Pudding"`**

© UCLES 2016

GradeMax

BLANK PAGE

© UCLES 2016

GradeMax

BLANK PAGE

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 International Examinations Copyright Acknowledgements Booklet. This is produced for each series of examinations and is freely available to download at www.cie.org.uk after the live examination series.

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

© UCLES 2016