Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

8177309785

COMPUTER SCIENCE **9608/23**

Paper 2 Fundamental Problem-solving and Programming Skills **October/November 2017**

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 **14** printed pages and **2** blank pages.

DC (JP) 154213
© UCLES 2017

CAMBRIDGE International Examinations

[Turn over

1 (a) (i) Procedural high-level languages usually support different data types.

Give an appropriate data type for each data value in the following table.

Data value	Data type
27	
"27"	
"27.3"	
TRUE	
27/3/2015	
27.3	

[6]

(ii) State an appropriate data structure to store the individual test scores for a class of students.

.....[1]

(iii) Describe how characters are represented using the ASCII character set.

.....

.....

.....

.....[2]

(b) Functions and procedures are subroutines.

Explain why you should use subroutines when designing a program solution.

.....

.....

.....

.....[2]

© UCLES 2017

(c) The following pseudocode is an example of nested `IF` statements.

```
IF MyVar = 1
   THEN
      CALL Proc1()
   ELSE
      IF MyVar = 2
         THEN
            CALL Proc2()
         ELSE
            IF MyVar = 3
               THEN
                  CALL Proc3()
               ELSE
                  OUTPUT "Error"
            ENDIF
      ENDIF
ENDIF
```

Use **pseudocode** to write a `CASE` statement with the same functionality.

.....

.....

.....

.....

.....

.....

.....

.....[4]

(d) Program coding is a transferable skill.

You are given program code written in a high-level language that you have not studied.

State **two** different features of the code that you should be able to recognise.

1 .....

.....

2 .....

.....

[2]

2 The following pseudocode represents a simple algorithm.

```
DECLARE NumberFound, Remainder, Number : INTEGER
DECLARE StartNumber, EndNumber, Divisor : INTEGER

INPUT StartNumber
INPUT EndNumber
INPUT Divisor
NumberFound ← 0

FOR Number ← StartNumber TO EndNumber
   Remainder ← MODULUS(Number, Divisor)
   IF Remainder = 0
      THEN
         OUTPUT Number
         NumberFound ← NumberFound + 1
   ENDIF
ENDFOR
OUTPUT "Count: " & NumberFound
```

For the built-in functions list, refer to the **Appendix** on page 14.

(a) Complete the following trace table.

StartNumber	EndNumber	Divisor	NumberFound	Number	Remainder	Output
11	13	2	0			

[3]

(b) Describe the purpose of this algorithm.

.....

.....

.....

.....

.....

.....[3]

(c) Draw a program flowchart to represent this algorithm.

Variable declarations are **not** required in program flowcharts.

[10]

3 **(a)** A multi-user computer system stores information about users. It uses a 1D array, `UserNameArray`, of type `STRING`. There are 100 elements in the array.

The format of the string in each element of the array is as follows:

```
<UserID><UserName>
```

- `UserID` is a six-character string of numerals.
- `UserName` is a variable-length string.

Write **pseudocode** for a procedure, `BubbleSort`, to perform an efficient bubble sort on `UserNameArray`. The array is to be sorted in ascending order of `UserID`.

You should assume that `UserNameArray` has been declared as a global variable.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

© UCLES 2017

.....

.....

.....

.....

.....

.....[8]

(b) The value of `UserID` should be unique for each user but a problem has occurred and repeated `UserID` values may have been issued.

The array is sorted by `UserID`, so any repeated `UserID` values will appear in consecutive array elements.

A procedure, `FindRepeats` is required.

This will:

- compare each element with the previous element and output the `UserID` and `UserName` if the `UserID` is repeated
- output the total number of `UserID`s that are repeated.

For example, the `UserNameArray` contains the following entries.

Array element	Comment
	
`122222Jim Moriarty`	
	
`123456Fred Smith`	
`123456Eric Sykes`	Repeated User ID
`123456Kevin Turvey`	Repeated User ID
	
`222244Alice Chan`	
`222244Myra Singh`	Repeated User ID
	
`333333Yasmin Halim`	
	

For this example, the output is:

```
123456Eric Sykes
123456Kevin Turvey
222244Myra Singh
There are 3 repeated UserIDs
```

If no repeated `UserID`s are found, the output is:

```
The array contains no repeated UserIDs
```

Write **program code** for the procedure, `FindRepeats`.
You should assume that `UserNameArray` has been declared as a global variable.

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

Programming language .....

Program code

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[8]

(c) (i) The `FindRepeats` procedure forms part of a program.

Name **three** stages in a program development cycle.

1 .....

2 .....

3 .....
[3]

(ii) The program containing `FindRepeats` will be created using an IDE.

State what is meant by IDE.

.....

.....[1]

(iii) Name **two** features provided by an IDE that assist in the program development cycle.

1 .....

.....

2 .....

.....
[2]

(iv) The procedure, `FindRepeats`, is written assuming there are 100 elements in `UserNameArray`.

In the main program, the global array, `UserNameArray`, has been declared with only 50 elements.

State the type of error this will cause.

.....[1]

© UCLES 2017

4 Numeric formatting converts a numeric value to a string in order to present it in a specific way.

In a generic high-level language, formatting is implemented using a mask system. In this system, each character of the mask corresponds to one character of the formatted string.

Mask characters have the following meaning:

Mask character	Meaning
`#`	Character must be a digit or a space
`0`	Character must be a digit

Any other mask characters are taken as literal values and are included in the formatted string.

(a) Using the mask `"###00.00"`, complete the following table. Use □ to represent a space. The first value has been done for you.

Value	Formatted string
1327.5	`"□1327.50"`
1234	
7.456	

[2]

(b) For each row in the following table, define the mask required to produce the formatted output from the given value. □ represents a space.

Value	Required output	Mask
1234.00	`"1,234.00"`	
3445.66	`"£3,445.66"`	
10345.56	`"$□□10,345"`	

[3]

5 A sports club maintains a record of the email address of each of its members. The details are stored in a text file, `EmailDetails.txt`. The format of each line of the text file is as follows:

```
<MembershipNumber><EmailAddress>
```

- `MembershipNumber` is a four-character string of numerals.
- `EmailAddress` is a variable-length string.

Membership of the club has increased and a four-character membership number is no longer adequate.

A procedure, `MakeNewFile`, is required to perform the following actions:

1. Create a new file, `NewEmailDetails.txt`
2. Read a line from file `EmailDetails.txt`
3. Extend `MembershipNumber` by adding two leading zero digits (for example, `"1234"` becomes `"001234"`)
4. Write the new line to file `NewEmailDetails.txt`
5. Repeat steps 2 to 4 for all lines in the original file.

(a) Write **pseudocode** for the procedure `MakeNewFile`.

For the built-in functions list, refer to the **Appendix** on page 14.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[8]

(b) A function, `IsEmailValid`, is to be written to test for a valid email address format.

An email address has a valid format if it obeys the following three rules:

1. It contains a single '@' symbol.
2. The '@' symbol must be preceded by at least one character.
3. The '@' symbol must be followed by at least three characters.

Choose **three** different invalid strings to test distinct aspects of the rules.

Explain your choice in each case.

1 .....

Explanation .....

.....

.....

2 .....

Explanation .....

.....

.....

3 .....

Explanation .....

.....

.....

[6]

Appendix

Built-in functions (pseudocode)

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

```
MODULUS(x : INTEGER, y : INTEGER) RETURNS INTEGER
```

returns the remainder when x is divided by y using integer arithmetic.
Example: `MODULUS(5, 2)` returns `1`

```
INT(x : REAL) RETURNS INTEGER
```

returns the integer part of x.
Example: `INT(27.5415)` returns `27`

```
LENGTH(ThisString : STRING) RETURNS INTEGER
```

returns the integer value representing the length of string `ThisString`.
Example: `LENGTH("Happy Days")` returns `10`

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

returns leftmost x characters from `ThisString`.
Example: `LEFT("ABCDEFGH", 3)` returns string `"ABC"`

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

returns rightmost x characters from `ThisString`.
Example: `RIGHT("ABCDEFGH", 3)` returns string `"FGH"`

Operators (pseudocode)

Operator	Description
`&`	Concatenates (joins) two strings Example: `"Summer" & " " & "Pudding"` produces `"Summer Pudding"`
`AND`	Performs a logical `AND` on two Boolean values Example: `TRUE AND FALSE` produces `FALSE`
`OR`	Performs a logical `OR` on two Boolean values Example: `TRUE OR FALSE` produces `TRUE`

© UCLES 2017

BLANK PAGE

© UCLES 2017

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.