Cambridge Assessment International Education

# Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER | | | | | |

CANDIDATE NUMBER | | | | |

*5904136477*

**COMPUTER SCIENCE** **9618/22**

Paper 2 Fundamental Problem-solving and Programming Skills **October/November 2022**

**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.


© UCLES 2022

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

**1** **(a)** A programmer is developing an algorithm to solve a problem. Part of the algorithm would be appropriate to implement as a subroutine (a procedure or a function).

**(i)** State **two** reasons why the programmer may decide to use a subroutine.

1 .....

.....

2 .....

.....

[2]

**(ii)** A procedure header is shown in pseudocode:

```
PROCEDURE MyProc(Count : INTEGER, Message : STRING)
```

Give the correct term for the identifiers `Count` and `Message` **and** explain their use.

Term .....

Use .....

.....

.....

.....

[2]

**(b)** The algorithm in **part (a)** is part of a program that will be sold to the public.
All the software errors that were identified during in-house testing have been corrected.

Identify **and** describe the additional test stage that may be carried out before the program is sold to the public.

Test stage .....

Description .....

.....

.....

.....

.....

.....

[4]

**(c)** Part of an identifier table is shown:

| Variable | Type | Example value |
|---|---|---|
| FlagDay | DATE | 23/04/2004 |
| CharList | STRING | "ABCDEF" |
| Count | INTEGER | 29 |

Complete the table by evaluating each expression using the example values.

| Expression | Evaluation |
|---|---|
| MID(CharList, MONTH(FlagDay), 1) | |
| INT(Count / LENGTH(CharList)) | |
| (Count >= 29) AND (DAY(FlagDay) > 23) | |

[3]

**2** **(a)** An algorithm will process data from a test taken by a group of students. The algorithm will prompt and input the name and test mark for each of the 35 students.

The algorithm will add the names of all the students with a test mark of less than 20 to an existing text file `Support_List.txt`, which already contains data from other group tests.

**(i)** Describe the steps that the algorithm should perform.

Do **not** include pseudocode statements in your answer.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [5]

**(ii)** Explain why it may be better to store the names of the students in a file rather than in an array.

.....

.....

..... [1]

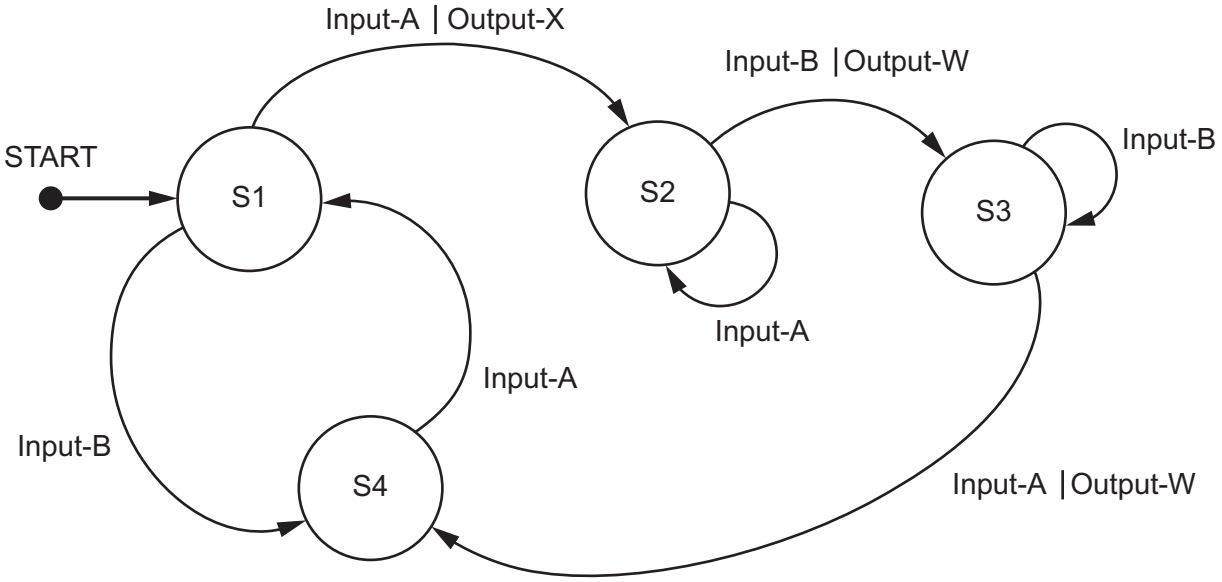
**(iii)** Explain why `WRITE` mode cannot be used in the answer to **part 2(a)(i)**.

.....

.....

..... [1]

**(b)** Examine the following state-transition diagram.




Complete the table to show the inputs, outputs and next states.

| Input | Output | Next state |
|---|---|---|
| | | S1 |
| Input-A | | |
| | | S2 |
| | Output-W | |
| | Output-W | |

[4]

**3** A stack is used in a program to store string data which needs to be accessed in several modules.

**(a)** A stack is an example of an Abstract Data Type (ADT).

Identify **one other** example of an ADT **and** describe its main features.

Example .....

Features .....

.....

.....

.....

.....

.....

[3]

**(b)** Explain how the stack can be implemented using an array.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [5]

**(c)** A second stack is used in the program. The diagram below shows the initial state of this stack. Value X is at the top of the stack and was the last item added.

Upper-case letters are used to represent different data values.

Stack operations are performed in three groups as follows:

Group 1:

```
PUSH D
PUSH E
```

Group 2:

```
POP
POP
POP
```

Group 3:

```
PUSH A
PUSH B
POP
PUSH C
```

Complete the diagram to show the state of the stack **after** each group of operations has been performed.

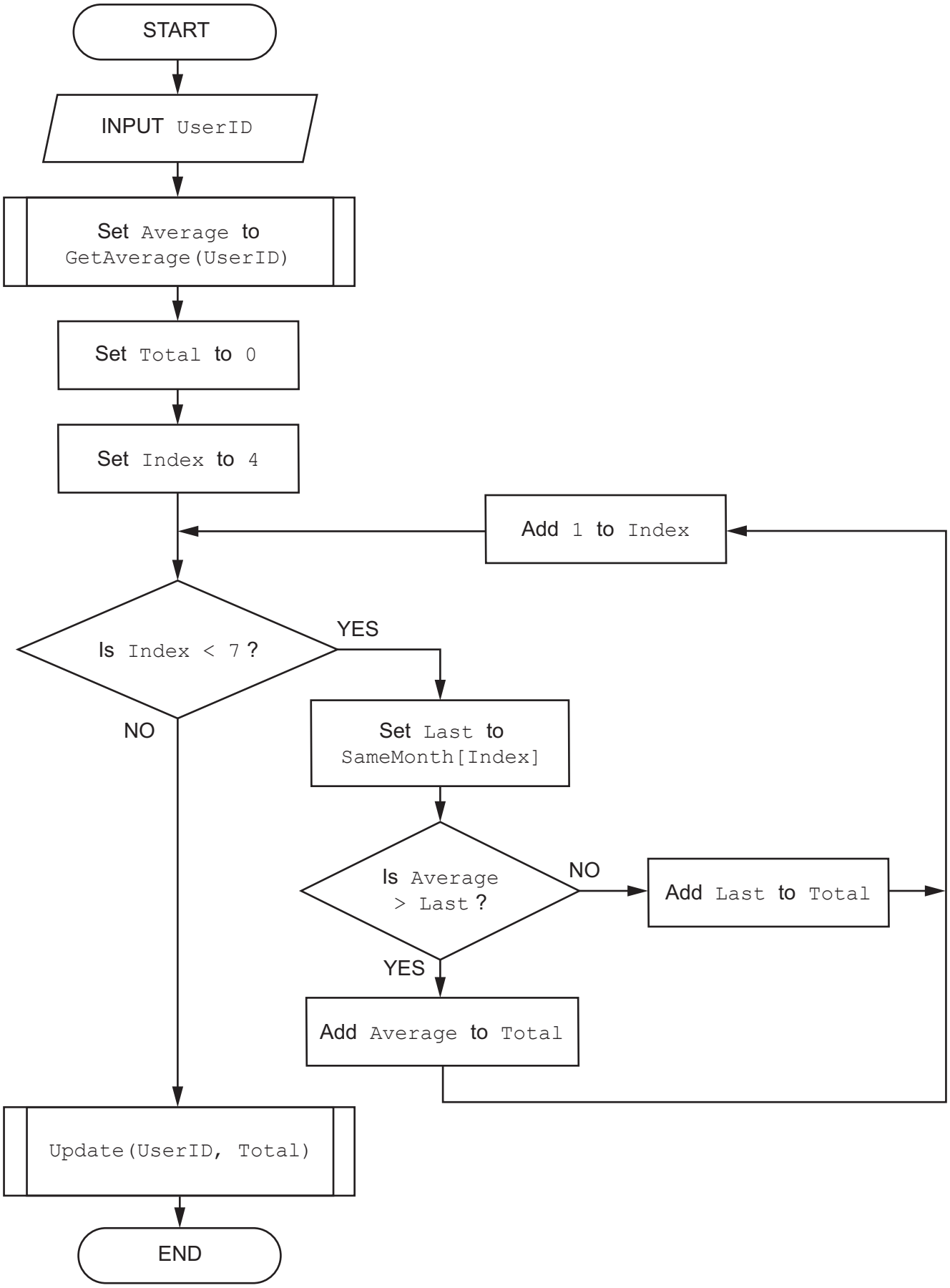
Include the current stack pointer (SP) **after** each group.

| Memory location | Initial state | | After Group 1 | After Group 2 | After Group 3 |
|---|---|---|---|---|---|
| **957** | | | | | |
| **956** | | | | | |
| **955** | | | | | |
| **954** | | | | | |
| **953** | X | ←SP | | | |
| **952** | Y | | | | |
| **951** | Z | | | | |
| **950** | P | | | | |

[5]

**BLANK PAGE**

**4** The program flowchart represents a simple algorithm.



**(a)** Write the equivalent pseudocode for the algorithm represented by the flowchart.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [6]

**(b)** Give the name of the iterative construct in the flowchart.

..... [1]

**5** Examine the following pseudocode.

```
IF A = TRUE THEN
   IF B = TRUE THEN
      IF C = TRUE THEN
         CALL Sub1()
      ELSE
         CALL Sub2()
      ENDIF
   ENDIF
ELSE
   IF B = TRUE THEN
      IF C = TRUE THEN
        CALL Sub4()
      ELSE
        CALL Sub3()
      ENDIF
   ELSE
      IF C = FALSE THEN
        CALL Sub3()
      ELSE
        CALL Sub4()
      ENDIF
   ENDIF
ENDIF
```

A programmer wants to re-write the pseudocode as **four** separate `IF...THEN...ENDIF` statements, each containing a single `CALL` statement. This involves writing a single, simplified logic expression as the condition in each statement.

Write the amended pseudocode.

1 .....

.....

.....

2 .....

.....

.....

3 .....

.....

.....

4 .....

.....

.....

[4]

**BLANK PAGE**

**6** **(a)** The factorial of an integer number is the product of all the integers from that number down to 1.

In general, the factorial of n is n × (n−1) × ... × 2 × 1

For example, the factorial of 5 is 5 × 4 × 3 × 2 × 1 = 120

In this question, n will be referred to as the `BaseNumber`.

A function `FindBaseNumber()` will:

- be called with a positive, non-zero integer value as a parameter
- return `BaseNumber` if the parameter value is the factorial of the `BaseNumber`
- return −1 if the parameter value **is not** a factorial.

For example:

| Parameter value | Value returned |
|---|---|
| 120 | 5 |
| 12 | −1 |
| 6 | 3 |
| 1 | 1 |

`FindBaseNumber(12)` will return −1 because 12 is not a factorial.

You may use the rest of this page for rough working.

© UCLES 2022

Write pseudocode for the function `FindBaseNumber()`.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [7]

**(b)** A program is written to allow a user to input a sequence of values to be checked using the function `FindBaseNumber()`.

The user will input one value at a time. The variable used to store the user input has to be of type string because the user will input 'End' to end the program.

Valid input will be converted to an integer and passed to `FindBaseNumber()` and the return value will be output.

Complete the table by giving **four** invalid strings that may be used to test distinct aspects of the required validation. Give the reason for your choice in each case.

| Input | Reason for choice |
|---|---|
| | ..... ..... ..... |
| | ..... ..... ..... |
| | ..... ..... ..... |
| | ..... ..... ..... |

[4]

**BLANK PAGE**

**7** A teacher is designing a program to perform simple syntax checks on programs written by students.

Two global 1D arrays are used to store the syntax error data. Both arrays contain 500 elements.

- Array `ErrCode` contains integer values that represent an error number in the range 1 to 800.
- Array `ErrText` contains string values that represent an error description.

The following diagram shows an example of the arrays.

| Index | ErrCode | ErrText |
|---|---|---|
| 1 | 10 | "Invalid identifier name" |
| 2 | 20 | "Bracket mismatch" |
| 3 | 50 | "Undeclared variable" |
| 4 | 60 | "Type mismatch in assignment" |
| … | | |
| 500 | 999 | <Undefined> |

**Note:**

- There may be less than 500 error numbers so corresponding elements in both arrays may be unused. Unused elements in `ErrCode` have the value `999`. The value of unused elements in `ErrText` is undefined.
- Values in the `ErrCode` array are stored in ascending order but not all values may be present, for example, there may be no error code `31`.

The teacher has defined two program modules as follows:

| Module | Description |
|---|---|
| `OutputError()` | • takes two parameters as integers:<br>○ a line number in the student's program<br>○ an error number<br>• searches for the error number in the `ErrCode` array:<br>○ if found, outputs the corresponding error description and the line number, for example:<br>`"Bracket mismatch on line 34"`<br>○ if not found, outputs the line number and a warning, for example:<br>`"Unknown error on line 34"` |
| `SortArrays()` | sorts the arrays into ascending order of `ErrCode` |

**(a)** Write **efficient** pseudocode for module `OutputError()`.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [6]

**(b)** Write an **efficient** bubble sort algorithm in pseudocode for module `SortArrays()`.

[8]

**(c)** Two 1D arrays were described at the beginning of the question. Both arrays contain 500 elements.

- Array `ErrCode` contains integer values that represent an error number in the range 1 to 800.
- Array `ErrText` contains string values that represent an error description.

The two arrays will be replaced by a single array. A user-defined data type (record structure) has been declared as follows:

```
TYPE ErrorRec
   DECLARE ErrCode : STRING
   DECLARE ErrText : STRING
ENDTYPE
```

**(i)** State the error in the record declaration.

.....

..... [1]

**(ii)** State **two** benefits of using the single array of the user-defined data type.

1 .....

.....

2 .....

.....

[2]

**(iii)** Write the declaration for the single array in pseudocode.

..... [1]

**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 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 2022