Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

7187074046

COMPUTER SCIENCE **9608/22**

Paper 2 Fundamental Problem-solving and Programming Skills **October/November 2020**

2 hours

You must answer on the question paper.

No additional materials are needed.

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.

This document has **20** pages. Blank pages are indicated.

1 **(a)** Algorithms usually consist of three different stages.

One stage is INPUT.

Name the **other** stages.

1 .....

2 .....
[1]

(b) An algorithm may be documented using different methods. These include structured English, a program flowchart, and pseudocode.

State what a program designer represents using one or more of these methods.

.....

..... [2]

(c) Programming languages support different data types.

Complete the table by giving four **different** data types together with an example data value for each.

Data type	Example data value

[4]

(d) Draw lines to connect each of the following computing terms with the appropriate description.

Term	Description
Black-box testing	A structure for the temporary storage of data
File	A method used when the structure of the program is unknown
Assignment	A method of setting the value of a variable
Array	A structure for the permanent storage of data

[3]

(e) A pseudocode algorithm assigns values to three variables as follows:

```
FlagA ← TRUE
FlagB ← FALSE
FlagC ← TRUE
```

Evaluate the expressions given in the following table:

Expression	Evaluates to
`NOT FlagB AND FlagC`	
`NOT (FlagB OR FlagC)`	
`(FlagA AND FlagB) OR FlagC`	
`NOT (FlagA AND FlagB) OR NOT FlagC`	

[2]

2 **(a)** The following pseudocode is an attempt to define an algorithm that takes two numbers as input and outputs the larger of the two numbers.

```
DECLARE A, B : INTEGER
INPUT A
INPUT B
IF A > B
   THEN
      OUTPUT A
   ELSE
      OUTPUT B
ENDIF
```

The algorithm needs to be amended to include the following changes:

1. Input **three** values, ensuring that each value input is unique.
2. Output the average.
3. Output the largest value.

Write the **pseudocode** for the amended algorithm.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [6]

(b) Complete the pseudocode expressions in the following table.

Use **only** functions and operators described in the **Appendix** on pages 18–19.

Expression	Evaluates to
`"ALARM: " & ..... ("Time: 1202" , .....)`	`"ALARM: 1202"`
`..... ("Stepwise." , ..... , .....)`	`"wise"`
`1.5 * ..... ("OnePointFive")`	`18`
`..... (27.5)`	`"27.5"`
`..... (9, 4)`	`2`

[5]

(c) A problem may be decomposed into sub-tasks when designing an algorithm.

Give **three** benefits of using sub-tasks.

1 .....

.....

2 .....

.....

3 .....

.....

[3]

3 A car has the ability to detect a skid by monitoring the rate of rotation (the rotational speed) of each wheel. If the rate of rotation of any wheel is not within 10% of the average of all four wheels, the car skids.

A function, `CheckSkid()`, is being developed.

The function will:

- simulate real-time data acquisition, by prompting for the input of four integer values in the range 0 to 1000 inclusive, representing the rate of rotation of each wheel
- calculate the average value
- check whether any individual value is more than 10% greater than the average or more than 10% less than the average
- return `TRUE` if any individual value is more than 10% greater than the average or more than 10% less than the average and `FALSE` otherwise
- output a suitable warning message.

(a) Write **program code** for the function `CheckSkid()`.

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]

(b) Give **two** sets of test data that could be used to test the function.

Test 1 – No skid detected

Value1	Value2	Value3	Value4

Test 2 – Skid detected

Value1	Value2	Value3	Value4

[2]

4 **(a)** The following structured English describes an algorithm used to count the number of odd and even digits in an input sequence.

1. Initialise variables `OddCount` and `EvenCount` to zero.
2. Prompt and input an integer.
3. If the integer is not in the range 0 to 9 then go to step 7.
4. If the integer is an even number then add 1 to `EvenCount`.
5. Otherwise add 1 to `OddCount`.
6. Repeat from step 2.
7. Output `"Same"` if there are the same number of odd and even integers.
8. Output `"Odd"` if there are more odd than even integers.
9. Output `"Even"` if there are more even than odd integers.

Draw a flowchart on the following page to represent the algorithm.

[7]

(b) The following pseudocode is an attempt to check whether two equal-length strings consist of identical characters.

Refer to the **Appendix** on pages 18–19 for the list of built-in functions and operators.

```
FUNCTION Compare(String1, String2 : STRING) RETURNS BOOLEAN
   DECLARE x, y, Len1, Len2 : INTEGER
   DECLARE RetFlag : BOOLEAN
   DECLARE NextChar : CHAR
   DECLARE New : STRING

   Len1 ← LENGTH(String1)
   RetFlag ← TRUE

   FOR x ← 1 TO Len1                          // for each char in String1
      Len2 ← LENGTH(String2)
      NextChar ← MID(String1, x, 1)           // get NextChar from String1
      New ← ""
      FOR y ← 1 TO Len2                       // for each char in String2
         IF NextChar <> MID(String2, y, 1)    // no match
            THEN
               New ← New & MID(String2, y, 1) // save this char from String2
         ENDIF
      ENDFOR
      String2 ← New                           // replace String2 with New
   ENDFOR

   IF LENGTH(String2) <> 0                    // anything left in String2 ?
      THEN
         RetFlag ← FALSE
   ENDIF

   RETURN RetFlag

ENDFUNCTION
```

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

```
Result ← Compare("SUB", "BUS")
```

The first row has been completed for you.

String1	String2	Len1	RetFlag	x	Len2	NextChar	New	y
"SUB"	"BUS"	3	TRUE	1				

[5]

(ii) State the value returned.

..... [1]

(iii) There is an error in the algorithm, which means that under certain circumstances, the function will return an incorrect value.

Describe the problem. Give **two** test strings that would demonstrate it.

Problem .....

.....

.....

.....

.....

Test String1 .....

Test String2 .....

[2]

(iv) Describe the modification that needs to be made to the algorithm to correct the error.

Do **not** use pseudocode or program code in your answer.

.....

.....

.....

.....

..... [1]

(v) State the name given to the type of testing that makes use of a trace table.

..... [1]

(vi) State **two** features found in a typical Integrated Development Environment (IDE) that may be used for debugging a program.

1 .....

2 .....

[2]

Question 5 begins on the next page.

5 A hashtag is used on a social media network. A hashtag is a string consisting of a hash character '#' followed by one or more alphanumeric characters.

A program is being developed to monitor the use of hashtags.

The program will include two global arrays each containing 10 000 elements:

- A 1D array, `TagString`, of type `STRING` stores each hashtag in a single element. All unused array elements contain an empty string (`""`).
- A 1D array, `TagCount`, of type `INTEGER` stores a count of the number of times each hashtag is used. The count value at a given index relates to the element stored at the corresponding index in the `TagString` array.

The contents of the two arrays will be stored in a text file `Backup.txt`. The format of each line of the file is:

```
<Hashtag><','><Count>
```

For example:

```
"#ComputerScienceClass,978"
```

A developer has started to define the modules as follows:

Module	Description
`InitArrays()`	• Initialise the arrays
`SaveArrays()`	• The contents of the two arrays are stored in the text file `Backup.txt` Existing file contents will be overwritten • Each hashtag and count are stored in one line of the file, as in the example above • Unused `TagString` elements are not added to the file • Returns the total number of unused `TagString` elements
`LoadArrays()`	• Values from the text file `Backup.txt` are stored in the two arrays • The number of elements stored is returned

(a) Write **pseudocode** for the module `InitArrays()`.

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [4]

(b) Write **pseudocode** for the module `SaveArrays()`.

[8]

(c) Write **program code** for the module `LoadArrays()`.

The module description is repeated here for reference.

Module	Description
`LoadArrays()`	• Values from the text file `Backup.txt` are stored in the two arrays • The number of elements stored is returned

You should assume:

- each line of the file contains a string of the correct format and no validation checks are required
- there are no more than 10 000 lines in the file.

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]

Appendix

Built-in functions (pseudocode)

Each function returns an error if the function call is not properly formed.

`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"`

`MOD(ThisNum : INTEGER, ThisDiv : INTEGER) RETURNS INTEGER`
returns the integer value representing the remainder when `ThisNum` is divided by `ThisDiv`

Example: `MOD(10,3)` returns `1`

`MID(ThisString : STRING, x : INTEGER, y : INTEGER) RETURNS STRING`
returns a string of length `y` starting at position `x` from `ThisString`

Example: `MID("ABCDEFGH", 2, 3)` returns string `"BCD"`

`DIV(ThisNum : INTEGER, ThisDiv : INTEGER) RETURNS INTEGER`
returns the integer value representing the whole number part of the result when `ThisNum` is divided by `ThisDiv`

Example: `DIV(10,3)` returns `3`

`NUM_TO_STRING(x : REAL) RETURNS STRING`
returns a string representation of a numeric value.
Note: This function will also work if `x` is of type `INTEGER`

Example: `NUM_TO_STRING(87.5)` returns `"87.5"`

`STRING_TO_NUM(x : STRING) RETURNS REAL`
returns a numeric representation of a string.
Note: This function will also work if `x` is of type `CHAR`

Example: `STRING_TO_NUM("23.45")` returns `23.45`

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`

BLANK PAGE