Cambridge International AS & A Level

CANDIDATE NAME	

CENTRE NUMBER	□□□□□	CANDIDATE NUMBER	□□□□

8494377800

COMPUTER SCIENCE **9608/22**

Paper 2 Fundamental Problem-solving and Programming Skills **October/November 2021**

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. Any blank pages are indicated.

1 **(a)** Design and coding are stages of the program development cycle.

State **two** activities from each of these stages.

Design stage

Activity 1 ....

Activity 2 ....

Coding stage

Activity 1 ....

Activity 2 ....

[4]

(b) The following is part of a procedure to manage the stock in a shop:

```
PROCEDURE InitVars()
   DECLARE Vs  : STRING  // program version
   DECLARE Pe  : REAL    // stock item purchase price
   DECLARE Exp : STRING  // date when item expires
   DECLARE S_lr: STRING  // supplier ID
   DECLARE Rp  : REAL    // low stock value
   DECLARE OUT : BOOLEAN // any items in stock ?

…

ENDPROCEDURE
```

(i) Give **two** reasons why the identifier names used in the procedure `InitVars()` are **not** examples of good practice.

Reason 1 ....

....

Reason 2 ....

....

[2]

(ii) Complete the following table by writing an appropriate identifier name for each of these four lines in the procedure `InitVars()`.

Line	Appropriate identifier name
102	
103	
105	
106	

[4]

(iii) Line `101` of the procedure needs to be changed because:

- the identifier should be declared as a fixed value that does not change during program execution
- the identifier name is not appropriate
- the program version should be `ver1.5.8`

Write the new correct statement for line `101`.

....

.... [2]

(c) A program can store characters using either the ASCII or the Unicode character set.

Give **two** reasons why Unicode is preferred to the ASCII character set for storing characters.

Reason 1 ....

....

....

Reason 2 ....

....

....

[2]

2 A café manager needs a program for a loyalty scheme. A customer is awarded one loyalty point for each purchase.

When a customer has ten or more points, the customer is offered a free slice of cake.

(a) During the program design, a decision is made to store the loyalty data in a text file called `LOYALTY.txt`. This file holds the membership number and total number of points for each customer on a single line.

Part of the program will check whether a particular customer has enough points for a free slice of cake.

Describe the algorithm for this part of the program using **structured English**.

....

....

....

....

....

....

....

....

....

....

.... [5]

(b) In another part of the program, the following statements assign values to variables.

```
DayOfMonth ← "15"

IsMember ← TRUE

DOB ← "22042001"

Firstname ← "Seanna"

Lastname ← "Adamson"

Points ← 12
```

Complete the table by writing the pseudocode expression that matches the description given **and** the value that each expression evaluates to.

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

Description of expression	**Pseudocode expression**	**Evaluates to**
Evaluates to `TRUE` if `DayOfMonth` is within the first seven days of the month		
Concatenates the second and third letters of `Firstname` with the last three letters of `Lastname`		
Evaluates to `TRUE` if `DOB` contains eight characters		
Evaluates to `TRUE` if the customer is a member and has enough points for a free slice of cake		

[4]

(c) Another part of the loyalty scheme program will need to access additional files. A filename contains a three-letter extension that represents its file type.

For example, the filename "thisfile.txt" has the extension "txt" and file type "Text". The filename will always be in lower case.

A function GetFileType() returns a string containing the description of the file type.

The function uses selection constructs as follows:

```
FUNCTION GetFileType(Filename : STRING) RETURNS STRING
   DECLARE FileExt  : STRING
   DECLARE FileType : STRING

   FileExt ← RIGHT(Filename, 3)

   IF FileExt = "rtf"
     THEN
       FileType ← "Rich text format"
   ENDIF

   IF FileExt = "csv"
     THEN
       FileType ← "Comma separated values"
   ENDIF

   IF FileExt = "txt"
     THEN
       FileType ← "Text"
     ELSE
       FileType ← "Unknown"
   ENDIF

   RETURN FileExt
ENDFUNCTION
```

Rewrite the function as follows:

- Replace the IF structures with a CASE structure.
- Correct the logic error in the function.

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

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [4]

3 **(a)** Describe the term **decomposition** when used to develop algorithms.

....

....

....

....

....

.... [3]

(b) Explain the term transferable skills **and** state how these skills are used in program development.

....

....

....

.... [2]

4 A global 1D array `Flower` represents the flowers in a field and is defined as follows:

- The array contains 20 elements of type integer.
- Each element of the array contains a number associated with each flower.

(a) A procedure, `InitialiseArray()`, initialises the array `Flower` by setting the value of each element to –1.

Write **pseudocode** for the procedure `InitialiseArray()`.

....

....

....

....

....

....

....

....

.... [4]

(b) A procedure called `RandomPath()` assigns values to the array `Flower` to store a path through the flowers in the fields as follows:

- the first flower to be visited is flower number 10
- randomly select the next flower to be visited between 1 and 20
- store the index of the next flower to be visited in the current element of the array `Flower` if the flower has not been visited
- visit each flower only once
- repeat until the path has been completed through all flowers.

An element of –1 indicates that a flower has not been visited.

Write **program code** for the procedure `RandomPath()`.

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

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [7]

(c) The procedure `RandomPath()` executes but gives an unexpected result.

(i) State the type of program error that has occurred **and** identify how this error can be detected.

Type of program error ....

Method of detection ....

.... [2]

(ii) Assume that the random number has been incorrectly generated in the procedure `RandomPath()`.

State **two** possible consequences of the incorrect use of the random number function.

1 ....

....

2 ....

.... [2]

5 The pseudocode for a procedure `FormatYear()` is written as follows:

```
PROCEDURE FormatYear(Year : INTEGER, UseSuffix : BOOLEAN)

   DECLARE OutYear : STRING
   CONSTANT SUFFIX1 = "BCE"
   CONSTANT SUFFIX2 = "CE"

   OutYear ← NUM_TO_STRING(Year)

   IF UseSuffix = TRUE
     THEN
       IF Year < 0
         THEN
           OutYear ← OutYear & SUFFIX1
         ELSE
           OutYear ← OutYear & SUFFIX2
         ENDIF
   ENDIF
   OUTPUT OutYear
ENDPROCEDURE
```

(a) Draw a program flowchart to represent the algorithm for the module.

Variable and constant declarations are not required in program flowcharts.

[5]

(b) For each of the following tests, choose **three** values of the parameters `Year` and `UseSuffix` that test **three different** aspects of the procedure `FormatYear()`.

State the expected output in each case.

Test 1

Parameter	Value	Expected output
`Year`		
`UseSuffix`		

Test 2

Parameter	Value	Expected output
`Year`		
`UseSuffix`		

Test 3

Parameter	Value	Expected output
`Year`		
`UseSuffix`		

[6]

(c) **(i)** Describe **two** features of black-box testing.

....

....

....

.... [2]

(ii) Describe **two** features of white-box testing.

....

....

....

.... [2]

BLANK PAGE

6 A geocode string consists of four characters that are followed by:

- the character '+'
- two more characters
- a comma
- a description of the location.

For example:

- The geocode string for Cambridge, UK is "646A+6R,CambridgeUK".
- The geocode string for Chicago, USA is "V9PG+3P,ChicagoUSA".

A program uses two 1D arrays to store data about the locations in which photographs were taken.

- An array GeoCodeData contains a geocode string for each location. A geocode string is added to this array if it does not exist in the array.
- An array GeoCodeLog contains the first seven characters of the geocode string for each photograph, followed by a space and the date when the photograph was taken. The date format is DD/MM/YYYY.

  For example, a photograph taken in Cambridge, UK on 21/12/2020 is represented as:

  ```
  "646A+6R 21/12/2020"
  ```

Assume that:

- both arrays contain 20 000 elements
- new data is added to the next unused element
- unused array elements are indicated by the string "AAAA+0A".

(a) A programmer has started to define the program module SearchLog().

Module	**Description**
SearchLog()	• takes the first seven characters of a geocode as a parameter • counts the number of times this parameter occurs in the array GeoCodeLog • extracts the date the last photograph was taken at the geocode location • returns a string formed by concatenating the count, a single space and the date

Write **pseudocode** for the module SearchLog().

....

....

....

....

....

....

[6]

(b) A text file, `Locations.txt`, stores data from the array `GeoCodeData` combined with data extracted from the array `GeoCodeLog`. Each line of the text file contains three items of data in the following format:

- The geocode stored in each element of the array `GeoCodeData` followed by a hash symbol (#).
- The number of times each geocode is found in the array `GeoCodeLog` followed by the hash symbol.
- The date of the last photograph taken at that location.

For example, twelve photographs were taken at geocode `V9PG+3P,ChicagoUSA`.
The last photograph was taken on 09/10/2020.

The corresponding entry in the file `Locations.txt` is:

```
"V9PG+3P,ChicagoUSA#12#09/10/2020"
```

The program module `ExtractArrays()` is defined as follows:

Module	Description
`ExtractArrays()`	• creates the text file `Locations.txt` • uses `SearchLog()` to obtain the string of the count and the date of the last photograph for each geocode location • writes the entry in the text file `Locations.txt`

You may assume that unused elements of `GeoCodeData` array are not added to the file `Locations.txt`.

Write **program code** for the module `ExtractArrays()`.

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

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [7]

Appendix

Built-in functions (pseudocode)

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

`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 `"BCD"`

`LENGTH(ThisString : STRING) RETURNS INTEGER`
returns the integer value representing the length of `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 `"ABC"`

`RIGHT(ThisString: STRING, x : INTEGER) RETURNS STRING`
returns rightmost `x` characters from `ThisString`

Example: `RIGHT("ABCDEFGH", 3)` returns `"FGH"`

`INT(x : REAL) RETURNS INTEGER`
returns the integer part of `x`

Example: `INT(27.5415)` returns `27`

`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`

`RAND(x : INTEGER) RETURNS REAL`
returns a real number in the range `0` to `x` (not inclusive of `x`)

Example: `RAND(87)` could return `35.43`

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