Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

2970875529

COMPUTER SCIENCE **9608/21**

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.

[Turn over

1 Part of the procedure `InitVars()` is shown:

```
PROCEDURE InitVars()
   DECLARE Var1 : INTEGER  // daily rainfall
   DECLARE Var2 : REAL     // the average wind speed
   DECLARE Var3 : STRING   // the ID number of the weather station

ENDPROCEDURE
```

(a) Give a **more meaningful** identifier name for each of the following variables used by the procedure.

Variable	New identifier name
Var1	
Var2	
Var3	

[3]

(b) A function called `ProcessVars()` assigns values to variables as shown.

Variable	Value
HouseCount	12
Turnout2018	20.23
TidalRiskCategory	'C'
IsConservationArea	FALSE
StationLocationName	"Ocean Boulevard"

Some pseudocode expressions in the function are shown in the following table.

Complete the table by evaluating each expression and writing the answer in the **Evaluates to** column.

If the expression is invalid, write "ERROR" in the **Evaluates to** column.

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

Pseudocode expression	Evaluates to
`LENGTH(HouseCount) > 6`	
`MOD(INT(Turnout2018) * 3, 4)`	
`ASC(TidalRiskCategory) + Turnout2018`	
`IsConservationArea AND (HouseCount <= 50)`	
`MID(StationLocationName, 1, 5) & " Eleven"`	

[5]

(c) The function header for `ProcessVars()` is:

```
FUNCTION ProcessVars(DataItem : REAL) RETURNS REAL
```

The following is an example of a parameter value passed to the function.

```
"L-MH-245,ManorHouse,N,F,230,12.34,0.98,12,N"
```

There is an error in the function header.

State the error **and** write the correct function header.

Error .....

.....

Correct function header .....

.....

[2]

(d) A programmer writes program code using an Integrated Development Environment (IDE).

Define the following **three** features of an IDE.

Breakpoints .....

.....

.....

Report (watch) window .....

.....

.....

Single stepping .....

.....

.....

[3]

2 **(a)** The following pseudocode function counts the occurrences of a character in a string.

Line numbers are shown for reference only.

```
01   DECLARE Message : STRING
02
03   FUNCTION CharacterCount(Letter : CHAR) RETURNS INTEGER
04
05      DECLARE LetterCount, Index : INTEGER
06      DECLARE ThisChar : INTEGER
07
08      LetterCount ← 1
09
10      FOR Index ← 1 TO LENGTH(Message) - 1
11          ThisChar ← LEFT(Message, Index, 1)
12          IF ThisChar = Letter
13            THEN
14              LetterCount ← LetterCount + 1
15          ENDIF
16      ENDFOR
17      RETURN LetterCount
18   ENDFUNCTION
```

(i) State the technical name of the iterative control structure used in this function.

..... [1]

(ii) Examine the pseudocode and write the answer in the table for each item.

Item	Answer
The scope of the variable `Message` is	
The start and end line numbers of a selection structure	
The identifier name of a user-defined function is	
An arithmetic operator used in the function is	

[4]

(b) Four lines of the pseudocode given in **part (a)** contain an error.

Identify the errors **and** write the correct pseudocode.

Line number for error 1 .....

Correct pseudocode .....

.....

Line number for error 2 .....

Correct pseudocode .....

.....

Line number for error 3 .....

Correct pseudocode .....

.....

Line number for error 4 .....

Correct pseudocode .....

.....

[4]

(c) A procedure, `Frequency()`, outputs the number of times each vowel occurs in a string.

The procedure will:

- prompt and input a string
- count the occurrence of each vowel in the string using a `CASE` structure
- output each vowel with its count value.

You may assume that vowels are the upper-case or lower-case characters `'a'`, `'e'`, `'i'`, `'o'`, and `'u'`.

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

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

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [5]

3 An aeroplane carries cases.

A case is not allowed on the aeroplane if:

- the total number of cases previously checked in for a flight is 300 or more
- the weight of a case exceeds the maximum of 50 kilograms.

The following statements apply:

- Each flight is identified by a five-character string variable `FlightNo`
- Each case is identified by a three-character string variable `CaseID`
- The weight of each case is represented by a two-digit numeric string variable `CaseWeight`

A text file, `HOLD-CARGO.txt`, stores data for cases on all flights. The format of each line of the file is:

```
<FlightNo><CaseID><CaseWeight>
```

(a) A procedure, `CheckWeight()`, is required as follows:

- take the flight number as parameter
- output the list of case IDs for the flight that are not allowed on the aeroplane.

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

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [8]

(b) A decision is made to use constants in the procedure `CheckWeight()` to represent the maximum weight and the maximum number of cases.

State **two** advantages of using constants instead of variables.

Advantage 1 .....

.....

Advantage 2 .....

.....

[2]

(c) `CheckWeight()` is a user-defined function.

(i) State **two** benefits of user-defined functions.

Benefit 1 .....

.....

Benefit 2 .....

.....

[2]

(ii) State **two** benefits of built-in functions.

Do **not** give the same answers as in **part (c)(i)**.

Benefit 1 .....

.....

Benefit 2 .....

.....

[2]

(d) One method of passing a parameter is by reference.

Name **and** describe another method.

Name .....

Description .....

.....

.....

.....

[3]

4 A willow tree is a type of tree that grows at an angle. A 1D array, `TreeAngle`, has 20 elements of integer type. Each element stores the angle of a willow tree measured in degrees.

A procedure called `SafetyCheck()` is required to:

- check each tree angle in the array and count the number of trees that exceed the safe limit of 36 degrees
- determine whether the count exceeds a maintenance limit stored in the global variable `MainTrigger`, in which case a maintenance visit is required
- output a suitable message if the count is less than or equal to `MainTrigger`, otherwise output a message showing the number of times the safe limit was exceeded as follows: `"Maintain 10 trees"`.

(a) The `SafetyCheck()` procedure is written in pseudocode.

Complete the following pseudocode.

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

```
PROCEDURE SafetyCheck()
   DECLARE Count : INTEGER
   DECLARE Index : INTEGER
   CONSTANT TreeCount = 20
   ..... ← 0
   FOR Index ← 1 TO .....
     IF TreeAngle[Index] > .....
       THEN
         Count ← Count + 1
     ENDIF
   ENDFOR
   IF ..... <= MainTrigger
     THEN
       OUTPUT "Maintenance not needed"
     ELSE
       OUTPUT "Maintain " & NUM_TO_STRING(Count) & " trees"
   ENDIF
ENDPROCEDURE
```

[4]

(b) The `TreeAngle` array is changed to a 2D array that contains 20 rows and 2 columns of integer numbers as follows:

- The first column stores the integer reference number of the tree.
- The second column stores the integer angle of the tree.

Study this example:

```
TreeAngle[15, 1] ← 767 // tree reference
TreeAngle[15, 2] ← 12  // the angle of the tree
```

A procedure, `CheckTree()`, is called with an integer parameter representing the tree's reference number.

The procedure will:

- search the `TreeAngle` array for the reference number
- when a match is found, prompt and input the angle of the tree, store the data in the second dimension of the `TreeAngle` array, and then output a message if its safety status has changed
- when a match is not found, output the reference number and `"No match"`.

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

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]

5 **(a)** Module iteration and module selection are features of a structure chart.

State **two other** features that may be represented in a structure chart.

Feature 1 .....

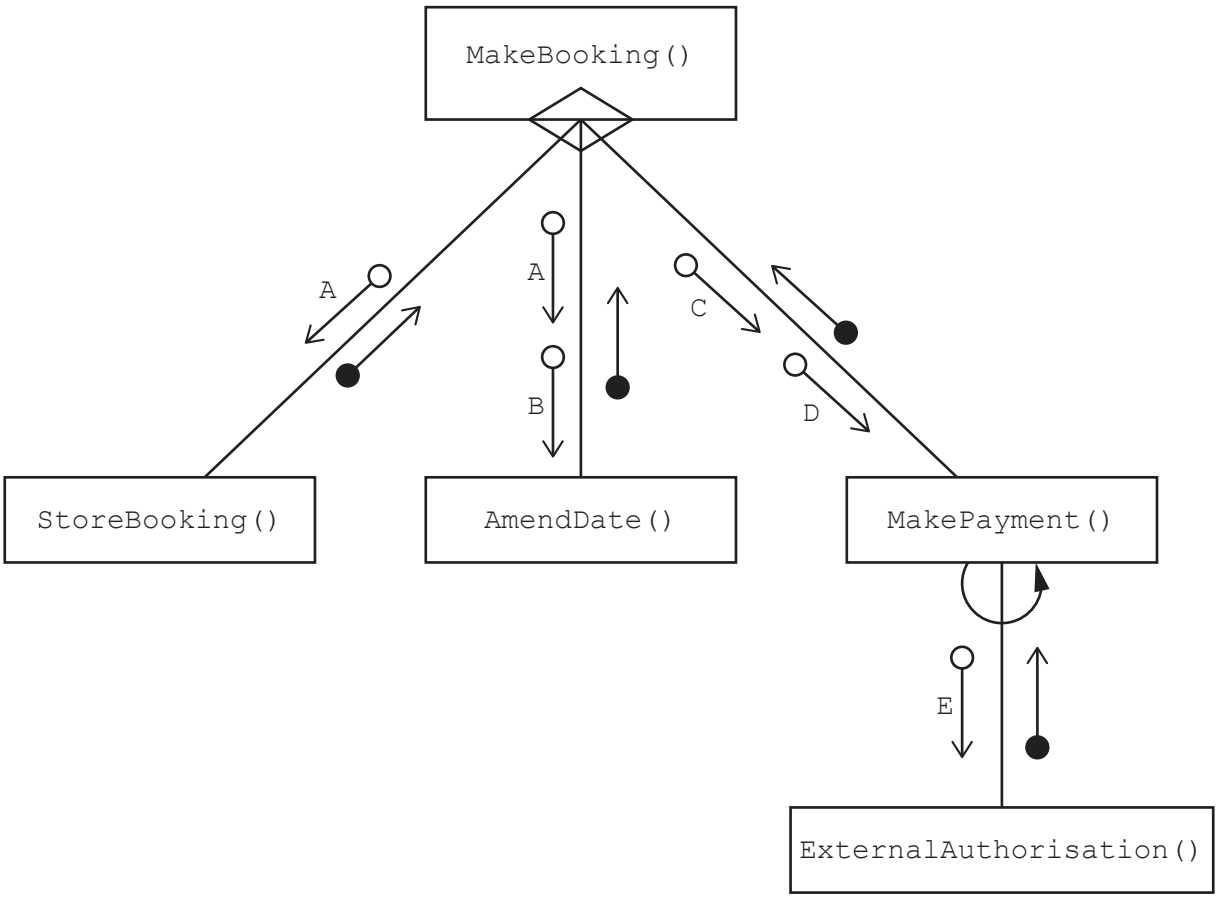
.....

Feature 2 .....

.....

[2]

(b) Examine this structure chart of a booking system.



Complete the following table by writing the correct parameter letter (A, B, C, D or E) for each parameter identifier.

Parameter identifier	Parameter letter
Quantity	
BookingID	
ItemCost	
TotalCost	
BookingDate	

[5]

6 A geocode is a string that specifies a geographical location.

The string consists of two alphanumeric characters which are followed by:

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

Here is an example of an email message that contains two geocodes. The first is for Coventry in the UK and the second is for Cambridge in the UK.

```
"This is my previous address 65+ER,CoventryUK.

I have now moved to my new location at 64+AA,CambridgeUK."
```

(a) A programmer decides to store the geocodes in a 1D array called `Location`. The array contains 10 000 elements and unused array elements are initialised with the string `"22+VV"`.

Write **pseudocode** to declare and initialise the array `Location`.

.....

.....

.....

.....

.....

.....

.....

..... [3]

(b) The following example pseudocode uses the function `FindGeoCodeIndex()` to search the array `Location` for the index position of a given geocode.

```
DECLARE CodePosition : INTEGER

CodePosition ← FindGeoCodeIndex(GeoCode)
IF CodePosition <> -1
  THEN
    OUTPUT Geocode & " found at " & NUM_TO_STRING(CodePosition)
ENDIF
```

Complete the program flowchart to represent the algorithm **only** for the function `FindGeoCodeIndex()`. Variable declarations are **not** required in program flowcharts.

[4]

(c) The function `RetrieveCode()` will extract a geocode from an email message.

Here is a summary of the design requirements for this function.

Parameters	Returns	Example
• A string containing the email message • An integer value containing the start position of the geocode in the message string	• A string containing the geocode	`Message ← "This is my previous address 65+ER,CoventryUK."` `RetrieveCode(Message, 29)` returns `"65+ER,CoventryUK"`

Assume that:

- the integer value points to the start of a valid geocode
- if a valid geocode is not at the end of the email message, it will be followed by a space or a full stop.

Write **program code** for the function `RetrieveCode()`.

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

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.... [6]

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 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 `"ABC"`

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

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

`ASC(ThisChar : CHAR) RETURNS INTEGER`
returns the ASCII value of `ThisChar`

Example: `ASC('A')` returns `65`

`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`

`UCASE(ThisChar : CHAR) RETURNS CHAR`
returns the character value representing the upper case equivalent of `ThisChar`
`ThisChar` is returned unchanged if it is not a lower-case alphabetic character

Example: `UCASE('a')` returns `'A'`

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

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

`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`

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