Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER | | | | | |

CANDIDATE NUMBER | | | | |

0832618096

COMPUTER SCIENCE **9618/23**

Paper 2 Fundamental Problem-solving and Programming Skills **May/June 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.

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

1 (a) The following table contains pseudocode examples.

Each example may include all or part of:

- selection
- iteration (repetition)
- assignment.

Complete the table by placing **one or more** ticks (✓) in each row.

Pseudocode example	Selection	Iteration	Assignment

`FOR Index ← 1 TO 3`
`  Safe[Index] ← GetResult()`
`NEXT Index`
		✓	✓

`OTHERWISE : OUTPUT "ERROR 1202"`	✓		
`REPEAT UNTIL Index = 27`		✓	
`INPUT MyName`			✓

`IF Mark > 74 THEN`
`  Grade ← 'A'`
`ENDIF`
	✓		✓

[5]

(b) (i) Program variables have values as follows:

Variable	Value
AAA	TRUE
BBB	FALSE
Count	99

Complete the table by evaluating each expression.

Expression	Evaluation
`AAA AND (Count > 99)`	FALSE
`AAA AND (NOT BBB)`	TRUE
`(Count <= 99) AND (AAA OR BBB)`	TRUE
`(BBB AND Count > 50) OR NOT AAA`	FALSE

[2]

(ii) Give an example of when a variable of type Boolean would be used.

- To terminate a (conditional) loop when a value has been found
- When the variable can take only one of two possible values
- (Accept by example): When a variable is recording when an action has been done e.g. Yes or No // light is on

..... [1]

2 A program has been written to implement a website browser and maintenance is now required.

One type of maintenance is called perfective.

Name **two other** types of maintenance that the program may require **and** give a reason for each.

Type 1 .....

Adaptive

Reason .....

To accommodate a change in the requirement / technology / legislation
e.g. a new HTML version is available

.....

.....

.....

Type 2 .....

Corrective

Reason .....

The program does not operate as expected / contains a bug
e.g. passwords are not hidden

.....

.....

.....

[4]

3 Four program modules are defined as follows:

Pseudocode module header
`PROCEDURE Sub1_A(XT : INTEGER, PB : STRING)`
`FUNCTION Sub1_B(RA : INTEGER) RETURNS BOOLEAN`
`PROCEDURE Sub1_C(SB : INTEGER, BYREF SA : STRING)`
`PROCEDURE Section_1()`

(a) A structure chart will be produced as part of the development process.

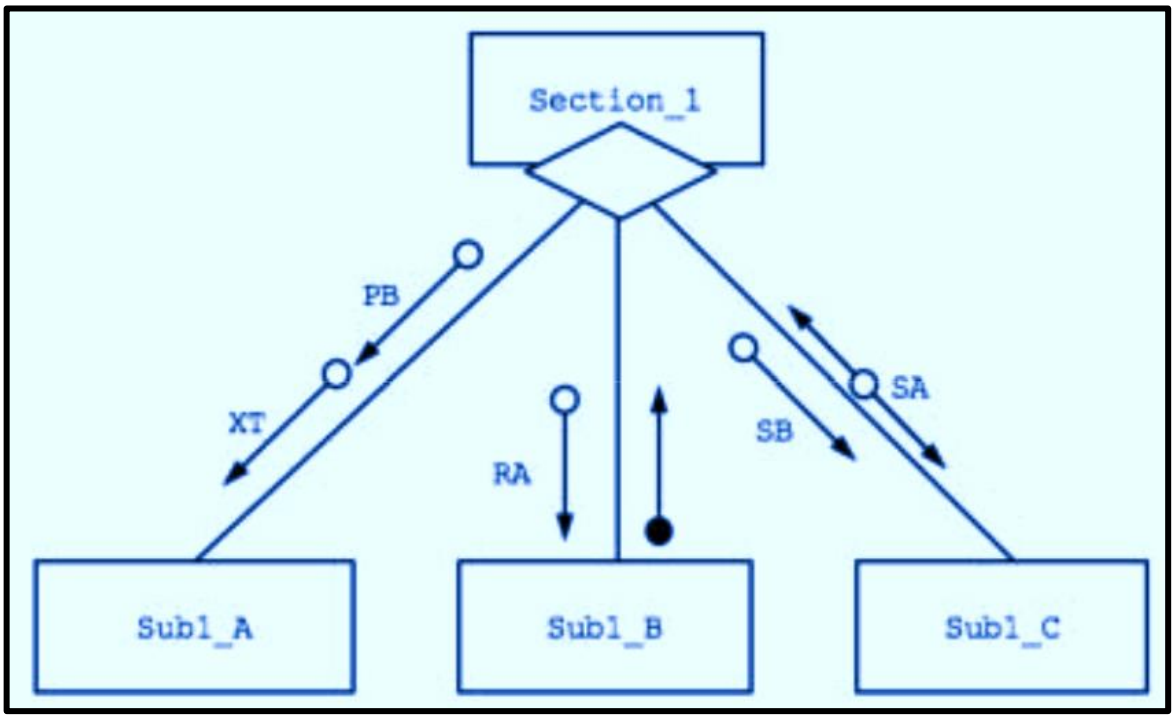
Describe the purpose of a structure chart.

- module relationships / hierarchy / selection / repetition // how a problem is broken down
- the parameters that are passed between the sub-tasks / modules // whether a module is a function or a procedure

[2]

(b) Module `Section_1()` calls one of the other three modules. The module called will be selected when the program runs.

Draw the structure chart. [5]



4 Items in a factory are weighed automatically. The weight is stored as an integer value representing the item weight to the nearest gram (g).

A function is written to validate the weight of each item. The function will return "PASS" if the weight of the item is within the acceptable range, otherwise the function will return "FAIL".

The acceptable weight range for an item is 150 g to 155 g inclusive.

The validation function is to be properly tested. Black-box testing will be used and a test plan needs to be produced.

Complete the table by writing additional tests to test this function.

Type of test data	Example test value	Expected return value	Explanation
Normal	153	"PASS"	Value within the acceptable range

[4]

Type of test data	Example test value	Expected return value	Explanation
Normal	153	"PASS"	Value within the acceptable range
Abnormal	< 149	"FAIL"	Outside acceptable range / too small
Abnormal / Boundary	149	"FAIL"	Maximum unacceptable
Boundary / Extreme / Normal	150	"PASS"	Minimum acceptable
Boundary / Extreme / Normal	155	"PASS"	Maximum acceptable
Abnormal / Boundary	156	"FAIL"	Minimum unacceptable
Abnormal	>156	"FAIL"	Outside acceptable range / too large

[Turn over

5 A program will store attendance data about each employee of a company.

The data will be held in a record structure of type `Employee`. The fields that will be needed are as shown:

Field	Typical value	Comment
`EmployeeNumber`	`123`	A numeric value starting from 1
`Name`	`"Smith,Eric"`	Format: <last name>','<first name>
`Department`	`"1B"`	May contain letters and numbers
`Born`	`13/02/2006`	Must not be before 04/02/1957
`Attendance`	`97.40`	Represents a percentage

(a) (i) Write pseudocode to declare the record structure for type `Employee`.

```
TYPE Employee
    DECLARE EmployeeNumber : INTEGER
    DECLARE Name : STRING
    DECLARE Department : STRING
    DECLARE Born : DATE
    DECLARE Attendance : REAL
ENDTYPE
```

[4]

(ii) A 1D array `Staff` containing 500 elements will be used to store the employee records.

Write pseudocode to declare the `Staff` array.

```
DECLARE Staff : ARRAY [1:500] OF Employee
```

[2]

(b) There may be more records in the array than there are employees in the company. In this case, some records of the array will be unused.

(i) State why it is good practice to have a standard way to indicate unused array elements.

- So that unused elements may be recognised when processing / searching
- Otherwise the element may contain old / unexpected data

[1]

(ii) Give **one** way of indicating an unused record in the `Staff` array.

- An `EmployeeNumber` field. e.g. < 1
- An empty string / impossible string e.g. "EMPTY" for name or department
- DOB a long time ago...
- Zero / Negative value for attendance

[1]

(c) A procedure `Absentees()` will output the `EmployeeNumber` and the `Name` of all employees who have an `Attendance` value of 90.00 or less.

Write pseudocode for the procedure `Absentees()`.

Assume that the `Staff` array is global.

```
PROCEDURE Absentees()
   DECLARE Index : INTEGER
   FOR Index ← 1 TO 500
      IF Staff[Index].EmployeeNumber <> -1 THEN   // not empty
         IF Staff[Index].Attendance <= 90 THEN
            OUTPUT Staff[Index].EmployeeNumber
            OUTPUT Staff[Index].Name
         ENDIF
      ENDIF
   NEXT Index
ENDPROCEDURE
```

[4]

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

For example:

factorial of 5 is given by 1 × 2 × 3 × 4 × 5 = 120
factorial of 7 is given by 1 × 2 × 3 × 4 × 5 × 6 × 7 = 5040
factorial of 1 = 1

Note: factorial of 0 = 1

A function `Factorial()` will:

- be called with an integer number as a parameter
- calculate and return the factorial of the number
- return −1 if the number is negative.

Write pseudocode for the function `Factorial()`.

```
FUNCTION Factorial(ThisNum : INTEGER) RETURNS INTEGER
   DECLARE Value : INTEGER

   IF ThisNum < 0 THEN
      Value ← -1
   ELSE
      Value ← 1
      WHILE ThisNum <> 0
         Value ← Value * ThisNum
         ThisNum ← ThisNum - 1
      ENDWHILE
    ENDIF

    RETURN Value

ENDFUNCTION
```

ALTERNATIVE RECURSIVE SOLUTION:

```
FUNCTION Factorial(ThisNum : INTEGER) RETURNS INTEGER

   IF ThisNum > 1 THEN
      RETURN ThisNum * Factorial(ThisNum - 1)
   ELSE
      IF ThisNum = 1 OR ThisNum = 0 THEN
         RETURN 1
      ELSE
         RETURN -1
      ENDIF
   ENDIF
ENDFUNCTION
```

[6]

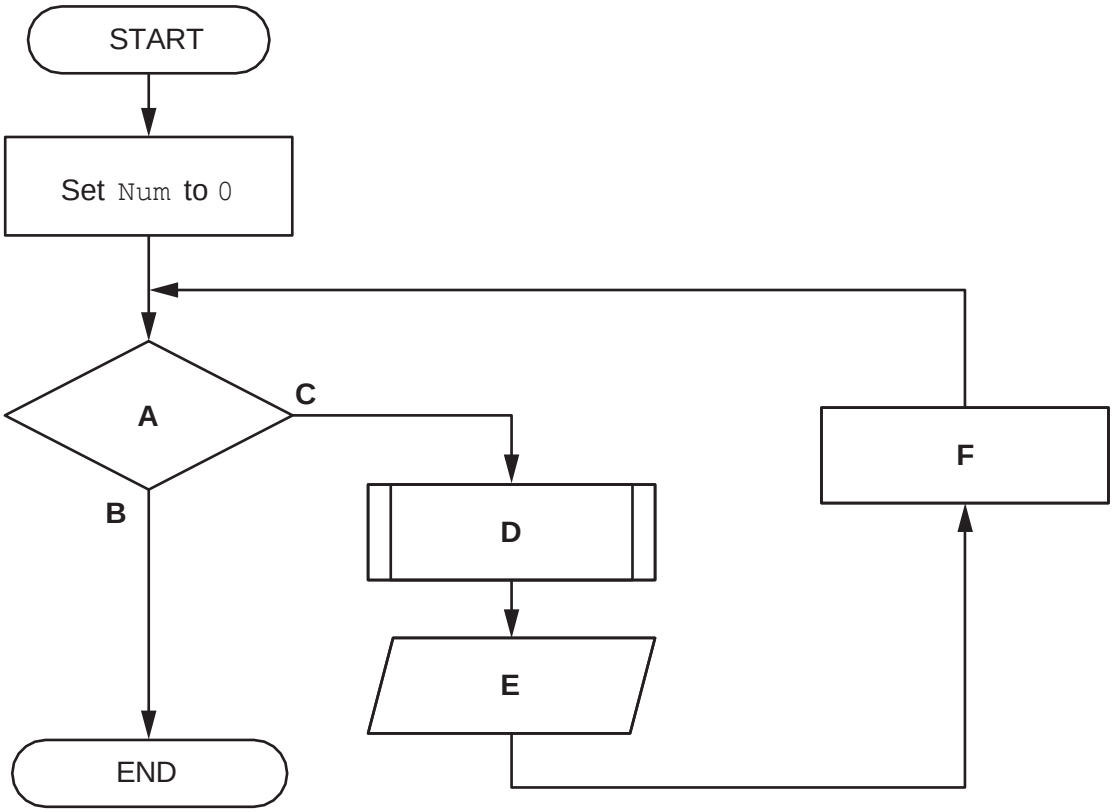
(b) A procedure `FirstTen()` will output the factorial of the numbers from 0 to 9. The procedure will use the function from **part (a)**.

The required output is:

```
Factorial of 0 is 1
Factorial of 1 is 1
Factorial of 2 is 2
```

```
Factorial of 9 is 362880
```

The program flowchart represents an algorithm for `FirstTen()`.



Complete the table by writing the text that should replace each label **A** to **F**.

Label	Text
A	Is Num > 9?
B	YES
C	NO
D	Set <Identifier> to Factorial(Num)
E	OUTPUT "Factorial of ", Num, " is ", <Identifier>
F	Set Num to Num + 1 // Increment Num

[4]

7 The following pseudocode represents an algorithm intended to output the last three lines as they appear in a text file. Line numbers are provided for reference only.

```
10  PROCEDURE LastLines(ThisFile : STRING)
11     DECLARE ThisLine : STRING
12     DECLARE Buffer : ARRAY[1:3] OF STRING
13     DECLARE LineNum : INTEGER
14     LineNum ← 1
15     OPENFILE ThisFile FOR READ
16
17     WHILE NOT EOF(ThisFile)
18        READFILE Thisfile, ThisLine // read a line
19        Buffer[LineNum] ← ThisLine
20        LineNum ← LineNum + 1
21        IF LineNum = 4 THEN
22           LineNum ← 1
23        ENDIF
24     ENDWHILE
25
26     CLOSEFILE ThisFile
27     FOR LineNum ← 1 TO 3
28        OUTPUT Buffer[LineNum]
29     NEXT LineNum
30  ENDPROCEDURE
```

(a) There is an error in the algorithm. In certain cases, a text file will have at least three lines but the output will be incorrect.

(i) State how the output may be incorrect.

The lines are output in an incorrect sequence / in the wrong order / not as they appear in the file [1]

(ii) Describe the error in the algorithm **and** explain how it may be corrected.

Description

- If the final line of the file is not written into array element 3
- then outputting the elements in the sequence 1 to 3 will give the error.

Explanation

- Attempt at description of 'shuffle'
- Copy Buffer[2] to Buffer[1] **AND** copy Buffer[3] to Buffer[2]
- Read a line from the file and write it to Buffer[3]

OR

- Store the index of the last element written to buffer (the last line of the file)
- Replace the FOR loop with something that starts at index
- and then wraps around (MOD 3)

OR (two-loop solution, not using an array)

- Loop to read file to end to get number of lines
- close and re-open file
- read (and discard) lines to number of lines - 3, then loop to read and output last 3 lines

[4]

(b) The original algorithm is implemented and sometimes the last three lines of the text file are output correctly.

State the condition that results in the correct output.

If the number of lines in the text file is a multiple of three [1]

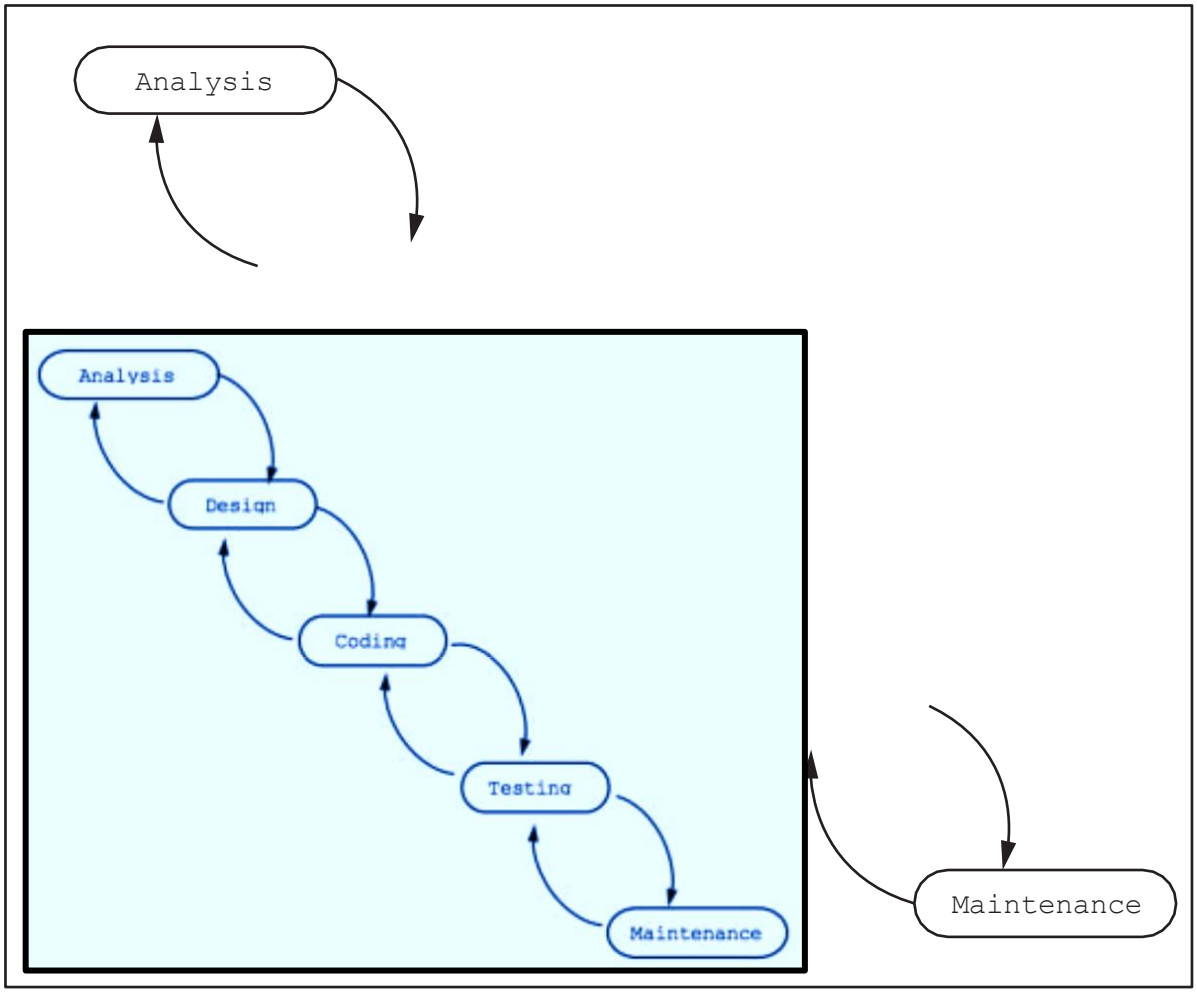
(c) Lines 20 to 23 inclusive could be replaced with a single pseudocode statement.

Write the pseudocode statement.

```
LineNum ← (LineNum MOD 3) + 1 // ((LineNum + 3) MOD 3)
          + 1
```
[2]

8 The following diagram shows the incomplete waterfall model of the program development life cycle.

(a) Complete the diagram.



[3]

(b) Explain the meaning of the downward and upward arrows.

Downward arrows .....

Downward arrows: result from one stage is input / passed to the next .....

.....

Upward arrows .....

Upward arrows: more work is required at a previous stage to complete the current stage .....

.....

[2]

(c) Identify another type of model for the program development life cycle.

. . Iterative / Rapid Application Development (RAD) ..... [1]

BLANK PAGE

9 A program allows a user to save passwords used to log in to websites. A stored password is then inserted automatically when the user logs in to the corresponding website.

A student is developing a program to generate a strong password. The password will be of a fixed format, consisting of **three groups of four** alphanumeric characters, separated by the hyphen character '-'.

An example of a password is: "FxAf-3hzV-Aq49"

A valid password:

- must be 14 characters long
- must be organised as three groups of four alphanumeric characters. The groups are separated by hyphen characters
- may include duplicated characters, **provided** these appear in different groups.

The programmer has started to define program modules as follows:

Module	Description
`RandomChar()`	• Generates a single random character from within one of the following ranges: ◦ 'a' to 'z' ◦ 'A' to 'Z' ◦ '0' to '9' • Returns the character
`Exists()`	• Takes two parameters: ◦ a string ◦ a character • Performs a case-sensitive search for the character in the string • Returns `TRUE` if the character occurs in the string, otherwise returns `FALSE`
`Generate()`	• Generates a valid password • Uses `RandomChar()` and `Exists()` • Returns the password

Note: in a case-sensitive comparison, 'a' is not the same as 'A'.

(a) Write pseudocode for the module `Generate()`.

```
FUNCTION Generate() RETURNS STRING
   DECLARE Password, Group : STRING
   DECLARE NextChar : CHAR
   DECLARE ACount, BCount : INTEGER
   CONSTANT HYPHEN = '-'

   Password ← ""

   FOR ACount ← 1 TO 3
      Group ← ""
      FOR BCount ← 1 TO 4
         REPEAT
            NextChar ← RandomChar()
         UNTIL Exists(Group, NextChar) = FALSE
         Group ← Group & NextChar
      NEXT BCount
      Password ← Password & Group & HYPHEN
   NEXT ACount
   Password ← LEFT(Password, 14)  // remove final hyphen
   RETURN Password
ENDFUNCTION
```

..... [7]

(b) A global 2D array `Secret` of type `STRING` stores the passwords together with the website domain name where they are used. `Secret` contains 1000 elements organised as 500 rows by 2 columns.

Unused elements contain the empty string (`""`). These may occur anywhere in the array.

An example of part of the array is:

Array element	Value
`Secret[27, 1]`	`"www.thiswebsite.com"`
`Secret[27, 2]`	`"●●●●●●●●●●●●"`
`Secret[28, 1]`	`"www.thatwebsite.com"`
`Secret[28, 2]`	`"●●●●●●●●●●●●"`

Note:

- For security, the passwords are stored in an encrypted form, shown as `"●●●●●●●●●●●●"` in the example.
- The passwords cannot be used without being decrypted.
- You may assume that the encrypted form of a password will **not** be an empty string.

Additional modules are defined as follows:

Module	Description
`Encrypt()`	• Takes a password as a string • Returns the encrypted form of the password as a string
`Decrypt()`	• Takes an encrypted password as a string • Returns the decrypted form of the password as a string
`FindPassword()`	• Takes a website domain name as a string • Searches for the website domain name in the array `Secret` • If the website domain name is found, the decrypted password is returned • If the website domain name is not found, an empty string is returned
`AddPassword()`	• Takes two parameters as strings: a website domain name and a password • Searches for the website domain name in the array `Secret` and if not found, adds the website domain name and the encrypted password to the array • Returns `TRUE` if the website domain name and encrypted password are added to the array, otherwise returns `FALSE`

The first three modules have been written.

Write pseudocode for the module `AddPassword()`.

```
FUNCTION AddPassword(Name, Password : STRING)
         RETURNS BOOLEAN
   DECLARE Index : INTEGER
   DECLARE Added : BOOLEAN

   Added ← FALSE
   Index ← 1

   IF FindPassword(Name) = "" THEN // Domain name not in
                                   // array
      WHILE Added = FALSE AND Index <= 500
         IF Secret[Index, 1] = "" THEN
            Secret[Index, 1] ← Name
            Secret[Index, 2] ← Encrypt(Password)
            Added ← TRUE
         ELSE
            Index ← Index + 1
         ENDIF
      ENDWHILE
   ENDIF

   RETURN Added

ENDFUNCTION
```

[6]

(c) The content of the array `Secret` is to be stored in a text file for backup.

It **must** be possible to read the data back from the file and extract the website domain name and the encrypted password.

Both the website domain name and encrypted password are stored in the array as strings of characters.

The encrypted password may contain any character from the character set used and the length of both the encrypted password and the website domain name is variable.

Explain how a single line of the text file can be used to store the website domain name and the encrypted password.

Solution based on field length:

- Convert the length of the website domain name (either field) …
- … to a string of fixed length
- Form a string by concatenate this string with the other two (and write as one line of the file)

Solution based on use of separator character:

- Select a (separator) character that cannot occur in the domain name (e.g. space)
- Create a string from the domain name followed by the separator
- ...Concatenate the encrypted password (and write as one line of the file)

..... [3]

BLANK PAGE

BLANK PAGE