

1 (a) Algorithms are produced during program development.

State when you would produce an algorithm during program development **and** state its purpose.

When

.....

Purpose

.....

[2]

(b) Selection is one of the basic constructs used in algorithms.

Explain the term **selection**.

.....

.....

.....

..... [2]

(c) Explain the process of **problem decomposition**. State one reason it may be used.

Explanation

.....

.....

Reason

.....

.....

[2]

(d) Name **two** features provided by a typical Integrated Development Environment (IDE) that assist in the **debugging** stage of the program development cycle.

1

2

[2]

- 2 (a) A structure chart is often produced as part of a modular program design. The chart shows the relationship between modules and the parameters that are passed between them.

Give **two other** features the structure chart can show.

Feature 1

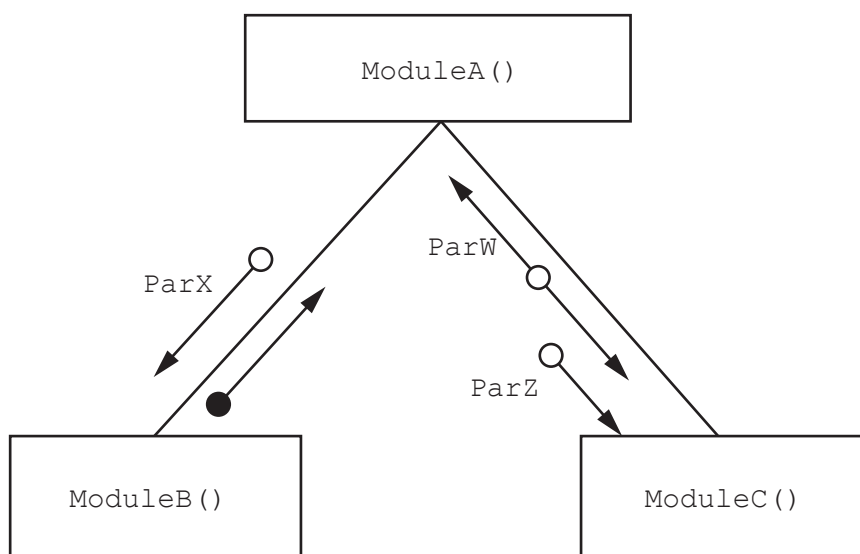
.....

Feature 2

.....

[2]

- (b) The following structure chart shows the relationship between three modules.



Parameter data types are:

ParW : REAL
ParX : INTEGER
ParZ : STRING

- (i) Write the **pseudocode** header for module `ModuleB()`.

.....

.....

..... [3]

- (ii) Write the **pseudocode** header for module `ModuleC()`.

.....

.....

..... [3]

3 (a) Study the following pseudocode.

```
Error ← Delta(Plan, Actual)
CASE OF Error
  > 10 : Steer ← Steer - 10
        0 : ZCount ← ZCount + 1
  < -10 : Steer ← Steer + 10
  OTHERWISE OUTPUT "Unexpected Error"
ENDCASE
```

Draw a program flowchart to represent the pseudocode. Variable declarations are **not** required in program flowcharts.



- (b) The following pseudocode algorithm has been developed to check whether a string contains a valid password.

To be a valid password, a string must:

- be longer than five characters
- contain at least one numeric digit
- contain at least one upper case letter
- contain at least one other character (not a numeric digit or an upper case letter).

```

10 FUNCTION Check(InString : STRING) RETURNS BOOLEAN
11
12   DECLARE Index : INTEGER
13   DECLARE StrLen : INTEGER
14   DECLARE NumUpper, NumDigit : INTEGER
15   DECLARE NextChar : CHAR
16   DECLARE NumOther : INTEGER
17
18   NumUpper ← 0
19   NumDigit ← 0
20
21   StrLen ← LENGTH(InString)
22   IF StrLen < 6
23     THEN
24       RETURN FALSE
25     ELSE
26       FOR Index ← 1 TO StrLen - 1
27         // loop for each character
28         NextChar ← MID(InString, Index, 1)
29         IF NextChar >= '0' AND NextChar <= '9'
30           THEN
31             NumDigit ← NumDigit + 1 // count digits
32           ELSE
33             IF NextChar >= 'A' AND NextChar <= 'Z'
34               THEN
35                 NumUpper ← NumUpper + 1 // count upper case
36             ENDIF
37           ENDIF
38       ENDFOR
39     ENDIF
40
41   NumOther ← StrLen - (NumDigit - NumUpper)
42   IF NumDigit >= 1 AND NumUpper >= 1 AND NumOther >= 1
43     THEN
44       RETURN TRUE
45     ELSE
46       RETURN FALSE
47   ENDIF
48
49 ENDFUNCTION

```


The pseudocode does not work under all circumstances.

The function was dry run with the string "1234AP" and the following trace table was produced. The string is an invalid password, but the pseudocode returned the value TRUE.

Trace table row	StrLen	Index	NextChar	NumUpper	NumDigit	NumOther
1	6			0	0	
2		1	'1'			
3					1	
4		2	'2'			
5					2	
6		3	'3'			
7					3	
8		4	'4'			
9					4	
10		5	'A'			
11				1		
12						3

(i) The pseudocode algorithm contains two errors.

State how the given trace table indicates the existence of each error.

Error 1

.....

.....

.....

Error 2

.....

.....

.....

[2]

- (ii) Give the line number of each error in the pseudocode algorithm **and** write the modified pseudocode to correct each error.

Line number for error 1

Correct pseudocode

.....

Line number for error 2

Correct pseudocode

.....

[2]

- (c) The term **adaptive maintenance** refers to amendments that are made in response to changes to the program specification. These changes usually affect the program algorithm.

Name **one other** part of the design that can change as a result of adaptive maintenance.

..... [1]

- 4 A global 1D array, `Contact`, of type `STRING` is used to store a list of names and email addresses. There are 1000 elements in the array. Each element stores one data item. The format of each data item is as follows:

```
<Name>': '<EmailAddress>
```

Name **and** `EmailAddress` are both variable-length strings.

For example:

```
"Wan Zhu:zwan99@mymail.com"
```

A function, `Extract()`, is part of the program that processes the array. A string data item is passed to the function as a parameter. The function will return the `Name` part. Validation is **not** necessary.

- (b) The original function, `Extract()`, needs to be modified to separate the name from the email address. The calling program can then use **both** of these values.

Write, in **pseudocode**, the header for the modified subroutine. Explain the changes you have made.

Subroutine header

.....

Explanation

.....

.....

.....

.....

.....

[3]

- 5 A company hires out rowing boats on a lake. The company has 20 boats, numbered from 1 to 20.

For safety reasons, the boats have to be serviced (checked and any damage repaired) regularly.

The company is developing a program to help manage the servicing of the boats.

Every time a boat is serviced, details are added at the end of the text file, `ServiceLog.txt`, as a single line of information. Each boat is serviced before it is hired out for the first time.

The format of each line is as follows:

`<BoatNumber><Date>`

`BoatNumber` and `Date` are as follows:

- `BoatNumber` is a two-digit numeric string in the range "01" to "20"
- `Date` is an 8-digit numeric string in the format `YYYYMMDD`

The programmer has defined the first module as follows:

Module	Description
<code>GetLastService()</code>	<ul style="list-style-type: none"> • Called with a string parameter representing the <code>BoatNumber</code> • Searches through the file <code>ServiceLog.txt</code> • Returns the date of the last service in the form "YYYYMMDD"

- (b) (i) Every time a boat is hired out, details of the hire are added at the end of a text file, `Hirelog.txt`. Each line of the text file corresponds to information about the hire of one boat.

The format of each line of information is as follows:

```
<Date><BoatNumber><HireDuration>
```

- `Date` is an 8-digit numeric string in the format `YYYYMMDD`
- `BoatNumber` is a two-digit numeric string in the range "01" to "20"
- `HireDuration` is a variable-length string representing a numeric value in hours. For example, the string "1.5" would represent a hire duration of 1½ hours.

A module `GetHours()` is defined as follows:

Module	Description
<code>GetHours()</code>	<ul style="list-style-type: none">• Takes two parameters:<ul style="list-style-type: none">◦ the <code>BoatNumber</code> as a string◦ the date of the last service for that boat ("<code>YYYYMMDD</code>") as a string• Searches through file <code>Hirelog.txt</code> and calculates the sum of the hire durations for the given boat after the given date (hire durations on or before the given date are ignored)• Returns the total of the hire durations as a real

Note:

Standard comparison operators may be used with dates in this format.
For example:

```
"20200813" > "20200812" would evaluate to TRUE
```

Parameter validation is **not** required.

- (ii) An additional module, `Validate()`, has been written to check that a given string corresponds to a valid `BoatNumber`. A valid `BoatNumber` is a two-digit numeric string in the range "01" to "20".

Give **three** test strings that are **invalid** for different reasons. Explain your choice in each case.

String 1

Reason

.....

.....

String 2

Reason

.....

.....

String 3

Reason

.....

.....

[6]

- (c) A new module is described as follows:

Module	Description
<code>ServiceList()</code>	<ul style="list-style-type: none"> • Takes an integer as a parameter that represents the maximum number of hours before a boat must be serviced • Uses <code>GetLastService()</code> and <code>GetHours()</code> • Outputs: <ul style="list-style-type: none"> ◦ a suitable heading ◦ the <code>BoatNumber</code> of each boat hired for more than the maximum number of hours since its last service ◦ the total hire duration for each of these boats <p>An example output list is:</p> <pre>Boat Service List 4: 123 17: 117</pre> <p>If no boats are due to be serviced, the output is:</p> <pre>Boat Service List No boats are due to be serviced</pre>

- (d) (i) A team of programmers will work on the program. Before they begin, the team meet to discuss ways in which the risk of program faults may be reduced during the design and coding stages.

State **two** ways to minimise program faults during the design and coding stages.

1

2

[2]

- (ii) During development, the team test the program using a process known as stub testing.

Explain this process.

.....
.....
.....
..... [2]

- (iii) Explain how single stepping may be used to help find a logic error in a program.

.....
.....
.....
..... [2]

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 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"

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"

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

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

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 the Cambridge Assessment Group. Cambridge Assessment is the brand name of the University of Cambridge Local Examinations Syndicate (UCLES), which itself is a department of the University of Cambridge.