

# QUESTION 10.



5 A company creates two new websites, Site X and Site Y, for selling bicycles.

Various programs are to be written to process the sales data.

These programs will use data about daily sales made from Site X (using variable SalesX), Site Y (using variable SalesY).

Data for the first 28 days is shown below.

	SalesDate	SalesX	SalesY
1	03/06/2015	0	1
2	04/06/2015	1	2
3	05/06/2015	3	8
4	06/06/2015	0	0
5	07/06/2015	4	6
6	08/06/2015	4	4
7	09/06/2015	5	9
8	10/06/2015	11	9
9	11/06/2015	4	1
...			
28	01/07/2015	14	8

(a) Name the data structure to be used in a program for SalesX.

.....[2]



(b) The programmer writes a program from the following pseudocode design.

```

x ← 0
FOR DayNumber ← 1 TO 7
    IF SalesX[DayNumber] + SalesY[DayNumber] >= 10
        THEN
            x ← x + 1
            OUTPUT SalesDate[DayNumber]
        ENDIF
    ENDFOR
OUTPUT x
    
```

(i) Trace the execution of this pseudocode by completing the trace table below.

x	DayNumber	OUTPUT
0		

[4]

(ii) Describe, in detail, what this algorithm does.

.....

.....

.....

.....[3]



- (c) The company wants a program to output the total monthly sales for one websites.

The programmer codes a function with the following function header:

```
FUNCTION MonthlyWebSiteSales(ThisMonth : INTEGER, ThisSite : CHAR)
                                RETURNS INTEGER
```

The function returns the total number of bicycles sold for the given month and website.

The function will use the following:

Identifier	Data type	Description
ThisMonth	INTEGER	Represents the month number e.g. 4 represents April
ThisSite	CHAR	Coded as: <ul style="list-style-type: none"> <li>• X for website X</li> <li>• Y for Website Y</li> </ul>

- (i) Give the number of parameters of this function. ....[1]
- (ii) Some of the following function calls may be invalid.

Mark each call with:

- a tick (✓), for a valid call
- a cross (✗), for an invalid call


For any function calls which are invalid, explain why.

Function call	Tick (✓) /cross (✗)	Explanation (if invalid)
MonthlyWebSiteSales(1, "Y")		
MonthlyWebSiteSales(11, 'X', 'Y')		
MonthlyWebSiteSales(12, 'X')		

[3]

- (d) The company decides to offer a discount on selected dates. A program is written to generate a text file, DISCOUNT\_DATES, with the dates on which a discount is offered.

The program creates a text file, DISCOUNT\_DATES (with data as shown), for a number of consecutive dates.

03/06/2015 TRUE
04/06/2015 FALSE
05/06/2015 FALSE
06/06/2015 FALSE
07/06/2015 FALSE
08/06/2015 FALSE
09/06/2015 FALSE
10/06/2015 TRUE
11/06/2015 FALSE

01/07/2015 FALSE

Each date and discount indicator is separated by a single <Space> character.

The discount indicators are:

- FALSE – indicates a date on which no discount is offered
- TRUE – indicates a date on which a discount is offered

A programming language has the built-in function CONCAT defined as follows:

```
CONCAT(String1 : STRING, String2 : STRING [, String3 : STRING] )
                                                    RETURNS STRING
For example:    CONCAT("San", "Francisco") returns "SanFrancisco"
                CONCAT("New", "York", "City") returns "NewYorkCity"
```

The use of the square brackets indicates that the parameter is optional.



The following incomplete pseudocode creates the text file DISCOUNT\_DATES.

Complete the pseudocode.

```
OPENFILE "DISCOUNT_DATES" FOR .....
INPUT .....
WHILE NextDate <>"XXX"
    INPUT Discount
    ..... = CONCAT(NextDate, " ", Discount)
    WRITEFILE "DISCOUNT_DATES", NextLine
    INPUT NextDate
    .....
OUTPUT "File now created"
CLOSEFILE
```

[4]



**Question 5(e) continues on page 18.**



(e) The `DISCOUNT_DATES` text file is successfully created.

The company now wants a program to:

- key in a date entered by the user
- search the text file for this date
- if found, output one of the following messages:
  - “No discount on this date”
  - “This is a discount date”
- if not found, output “Date not found”

(i) Add to the identifier table to show the variables you need for this new program.

Identifier	Data type	Description
<code>DISCOUNT_DATES</code>	<code>FILE</code>	Text file to be used

[3]





20

**BLANK PAGE**



# QUESTION 11.



5 A company creates two new websites, Site X and Site Y, for selling bicycles.

Various programs are to be written to process the sales data.

These programs will use data about daily sales made from Site X (using variable SalesX), Site Y (using variable SalesY).

Data for the first 28 days is shown below.

	SalesDate	SalesX	SalesY
1	03/06/2015	0	1
2	04/06/2015	1	2
3	05/06/2015	3	8
4	06/06/2015	0	0
5	07/06/2015	4	6
6	08/06/2015	4	4
7	09/06/2015	5	9
8	10/06/2015	11	9
9	11/06/2015	4	1
...			
28	01/07/2015	14	8

(a) Name the data structure to be used in a program for SalesX.

.....[2]



(b) The programmer writes a program from the following pseudocode design.

```

x ← 0
FOR DayNumber ← 1 TO 7
    IF SalesX[DayNumber] + SalesY[DayNumber] >= 10
        THEN
            x ← x + 1
            OUTPUT SalesDate[DayNumber]
        ENDIF
    ENDFOR
OUTPUT x
    
```

(i) Trace the execution of this pseudocode by completing the trace table below.

x	DayNumber	OUTPUT
0		

[4]

(ii) Describe, in detail, what this algorithm does.

.....

.....

.....

.....[3]



- (c) The company wants a program to output the total monthly sales for one websites.

The programmer codes a function with the following function header:

```
FUNCTION MonthlyWebSiteSales(ThisMonth : INTEGER, ThisSite : CHAR)
                                RETURNS INTEGER
```

The function returns the total number of bicycles sold for the given month and website.

The function will use the following:

Identifier	Data type	Description
ThisMonth	INTEGER	Represents the month number e.g. 4 represents April
ThisSite	CHAR	Coded as: <ul style="list-style-type: none"> <li>• X for website X</li> <li>• Y for Website Y</li> </ul>

- (i) Give the number of parameters of this function. ....[1]
- (ii) Some of the following function calls may be invalid.

Mark each call with:

- a tick (✓), for a valid call
- a cross (✗), for an invalid call


For any function calls which are invalid, explain why.

Function call	Tick (✓) /cross (✗)	Explanation (if invalid)
MonthlyWebSiteSales(1, "Y")		
MonthlyWebSiteSales(11, 'X', 'Y')		
MonthlyWebSiteSales(12, 'X')		

[3]

- (d) The company decides to offer a discount on selected dates. A program is written to generate a text file, DISCOUNT\_DATES, containing the dates on which a discount is offered.

The program creates a text file, DISCOUNT\_DATES (with data as shown), for a number of consecutive dates.

03/06/2015 TRUE
04/06/2015 FALSE
05/06/2015 FALSE
06/06/2015 FALSE
07/06/2015 FALSE
08/06/2015 FALSE
09/06/2015 FALSE
10/06/2015 TRUE
11/06/2015 FALSE

01/07/2015 FALSE

Each date and discount indicator is separated by a single <Space> character.

The discount indicators are:

- FALSE – indicates a date on which no discount is offered
- TRUE – indicates a date on which a discount is offered

A programming language has the built-in function CONCAT defined as follows:

```
CONCAT(String1 : STRING, String2 : STRING [, String3 : STRING] )
                                                    RETURNS STRING
For example:    CONCAT("San", "Francisco") returns "SanFrancisco"
                CONCAT("New", "York", "City") returns "NewYorkCity"
```

The use of the square brackets indicates that the parameter is optional.



The following incomplete pseudocode creates the text file DISCOUNT\_DATES.

Complete the pseudocode.

```
OPENFILE "DISCOUNT_DATES" FOR .....  
INPUT .....  
WHILE NextDate <>"XXX"  
    INPUT Discount  
    ..... = CONCAT(NextDate, " ", Discount)  
    WRITEFILE "DISCOUNT_DATES", NextLine  
    INPUT NextDate  
    .....  
OUTPUT "File now created"  
CLOSEFILE
```

[4]



**Question 5(e) continues on page 18.**



(e) The `DISCOUNT_DATES` text file is successfully created.

The company now wants a program to:

- key in a date entered by the user
- search the text file for this date
- if found, output one of the following messages:
  - “No discount on this date”
  - “This is a discount date”
- if not found, output “Date not found”

(i) Add to the identifier table to show the variables you need for this new program.

Identifier	Data type	Description
<code>DISCOUNT_DATES</code>	<code>FILE</code>	Text file to be used

[3]





20

**BLANK PAGE**



# QUESTION 12.



5 A firm employs workers who assemble amplifiers. Each member of staff works an average of 10 hours each day.

The firm records the number of completed amplifiers made by each employee each day.

Management monitor the performance of all its workers.

Production data was collected for 3 workers over 4 days.

Daily hours worked	
Worker 1	5
Worker 2	10
Worker 3	10

Production data			
	Worker 1	Worker 2	Worker 3
Day 1	10	20	9
Day 2	11	16	11
Day 3	10	24	13
Day 4	14	20	17

A program is to be written to process the production data.

(a) The production data is to be stored in a 2-dimensional array `ProductionData`, declared as follows:

```
DECLARE ProductionData ARRAY[1:4, 1:3] : INTEGER
```

(i) Describe **two** features of an array.

1 .....

.....

2 .....

.....[2]

(ii) Give the value of `ProductionData[3, 2]`.

.....[1]

(iii) Describe the information produced by the expression:

```
ProductionData[2, 1] + ProductionData[2, 2] + ProductionData[2, 3]
```

.....

.....[2]



(b) Complete the trace table for the pseudocode algorithm below.

```

FOR WorkerNum ← 1 TO 3
  WorkerTotal[WorkerNum] ← 0
ENDFOR

FOR WorkerNum ← 1 TO 3
  FOR DayNum ← 1 TO 4
    WorkerTotal[WorkerNum] ← WorkerTotal[WorkerNum] +
      ProductionData[DayNum, WorkerNum]
  ENDFOR
ENDFOR

FOR WorkerNum ← 1 TO 3
  WorkerAverage ← WorkerTotal[WorkerNum]/
    (4 * DailyHoursWorked[WorkerNum])

  IF WorkerAverage < 2
    THEN
      OUTPUT "Investigate", WorkerNum
    ENDFIF
  ENDFOR

```

WorkerNum	DayNum	WorkerAverage	OUTPUT	WorkerTotal		
				1	2	3



- (c) An experienced programmer suggests that the pseudocode would be best in procedure AnalyseProductionData.

Assume that both arrays, DailyHoursWorked and ProductionData, are available to the procedure from the main program and they are of the appropriate size.

```

PROCEDURE AnalyseProductionData(NumDays : INTEGER, NumWorkers : INTEGER)

  DECLARE .....
  DECLARE .....
  DECLARE .....
  DECLARE .....

  FOR WorkerNum ← 1 TO 3
    WorkerTotal[WorkerNum] ← 0
  ENDFOR

  FOR WorkerNum ← 1 TO 3
    FOR DayNum ← 1 TO 4
      WorkerTotal[WorkerNum] ← WorkerTotal[WorkerNum] +
                               ProductionData[DayNum, WorkerNum]
    ENDFOR
  ENDFOR

  FOR WorkerNum ← 1 TO 3
    WorkerAverage ← WorkerTotal[WorkerNum] /
                    (4 * DailyHoursWorked [WorkerNum])
    IF WorkerAverage < 2
      THEN
        OUTPUT "Investigate", WorkerNum
      ENDFOR
    ENDFOR

ENDPROCEDURE
  
```

- (i) Complete the declaration statements showing the local variables. [4]
- (ii) The original pseudocode has been ‘pasted’ under the procedure header.  
 Circle all the places in the original pseudocode where changes will need to be made.  
 Write the changes which need to be made next to each circle. [3]
- (iii) Write the statement for a procedure call which processes data for 7 days for 13 workers.  
 .....[1]

15  
BLANK PAGE



# QUESTION 13.



6 A firm employs five staff who take part in a training programme. Each member must complete a set of twelve tasks which can be taken in any order. When a member successfully completes a task, this is recorded.

A program is to be produced to record the completion of tasks for the five members of staff.

To test the code, the programmer makes the program generate test data.

The program generates pairs of random numbers:

- the first, in the range, 1 to 5 to represent the member of staff
- the second, in the range, 1 to 12 to represent the task

Each pair of numbers simulates the completion of one task by one member of staff.

(a) Explain why the generation of 60 (5 staff x 12 tasks) pairs of random numbers will not simulate all tasks completed by all staff.

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

(b) Data is currently recorded manually as shown.

Staff number	Task number											
	1	2	3	4	5	6	7	8	9	10	11	12
1												
2												
3				✓								
4												
5								✓				

The table shows that two members of staff have each successfully completed one task.

The program must use a suitable data structure to store, for all staff:

- tasks successfully completed
- tasks not yet successfully completed

The program will output the staff number and task number in the order in which tasks are completed.



The program design in pseudocode is produced as follows:

```
01 DECLARE StaffNum          : INTEGER
02 DECLARE TaskNum           : INTEGER
03 DECLARE .....
04 DECLARE NewStaffTask      : BOOLEAN
05
06 CALL InitialiseTaskGrid
07 Completed ← 0
08 WHILE Completed <> 60
09     NewStaffTask ← FALSE
10     WHILE NewStaffTask = FALSE
11         StaffNum ← RANDOM(1,5)           //generates a random number
12         TaskNum ← RANDOM(1,12)          //in the given range
13         IF TaskGrid[StaffNum, TaskNum] = FALSE
14             THEN
15                 TaskGrid[StaffNum, TaskNum] ← TRUE
16                 NewStaffTask ← TRUE
17                 OUTPUT StaffNum, TaskNum
18             ENDIF
19     ENDWHILE
20     Completed ← Completed + 1
21 ENDWHILE
22 OUTPUT "Staff Task Count", Completed
23
24 // end of main program
25
26 PROCEDURE InitialiseTaskGrid()
27     DECLARE i : INTEGER
28     DECLARE j : INTEGER
29     FOR i ← 1 TO 5
30         FOR j ← 1 TO 12
31             TaskGrid[i, j] ← FALSE
32         ENDFOR
33     ENDFOR
34 ENDPROCEDURE
```





Study the pseudocode and answer the questions below.

Give the line number for:

- (i) The declaration of a `BOOLEAN` global variable. .... [1]
  - (ii) The declaration of a local variable. .... [1]
  - (iii) The incrementing of a variable used as a counter, but not to control a 'count controlled' loop. .... [1]
  - (iv) A statement which uses a built-in function of the programming language. .... [1]
- (c) (i) State the number of parameters of the `InitialiseTaskGrid` procedure. .... [1]
- (ii) Copy the condition which is used to control a 'pre-condition' loop.  
 ..... [1]
- (iii) Explain the purpose of lines 13 – 18.  
 .....  
 .....  
 .....  
 .....  
 .....  
 .....  
 ..... [3]
- (iv) Give the global variable that needs to be declared at line 03.  
 ..... [2]





**Question 7 begins on page 14.**

# QUESTION 14.



6 Some pseudocode statements follow which use the following built-in functions:

ONECHAR(ThisString : STRING, Position : INTEGER) RETURNS CHAR  
 returns the single character at position Position (counting from the start of the string with value 1) from the string ThisString.  
 For example: ONECHAR("Barcelona", 3) returns 'r'.

CHARACTERCOUNT(ThisString : STRING) RETURNS INTEGER  
 returns the number of characters in the string ThisString.  
 For example: CHARACTERCOUNT("South Africa") returns 12.

(a) Study the following pseudocode statements.

Give the values assigned to variables x and y.

(i)  $x \leftarrow \text{CHARACTERCOUNT}(\text{"New Delhi"}) + 3$       x ..... [1]

(ii)  $y \leftarrow \text{ONECHAR}(\text{"Sri Lanka"}, 5)$       y ..... [1]

(b) A program is to be written as follows:

- the user enters a string
- the program will form a new string with all <Space> characters removed
- the new string is output

```
NewString ← ""
INPUT InputString

j ← CHARACTERCOUNT(InputString)
FOR i ← 1 TO j
    NextChar ← ONECHAR(InputString, i)
    IF NextChar <> " "
        THEN
            // the & character joins together two strings
            NewString ← NewString & NextChar
        ENDIF
    ENDFOR

OUTPUT NewString
```

(i) Complete the identifier table below.

Identifier	Data type	Description
InputString	STRING	The string value input by the user



- (ii) An experienced programmer suggests this pseudocode would be best function.

Complete the re-design of the pseudocode as follows:

The main program:

- the user enters `MyString`
- the function is called and the changed string is assigned to variable `ChangedString`

The function:

- has identifier `RemoveSpaces`
- has a single parameter
- will include the declaration for any local variables used by the function

```
// main program
INPUT MyString
ChangedString←RemoveSpaces (.....)
OUTPUT ChangedString

// function definition
FUNCTION RemoveSpaces (.....) RETURNS .....
.....
.....
.....
.....

j ← CHARACTERCOUNT (InputString)
FOR i ← 1 TO j
    NextChar ← ONECHAR (InputString, i)
    IF NextChar <> " "
        THEN
            // the & character joins together two strings
            NewString ← NewString & NextChar
        ENDIF
    ENDFOR
.....
ENDFUNCTION
```

# QUESTION 15.



6 A string-handling function has been developed. The pseudocode for this function is given below.

For the built-in functions list, refer to the **Appendix** on page 18.

```

FUNCTION SSM(String1, String2 : STRING) RETURNS INTEGER
    DECLARE n, f, x, y : INTEGER

    n ← 0
    f ← 0

    REPEAT
        n ← n + 1
        x ← n
        y ← 1
        WHILE MID(String1, x, 1) = MID(String2, y, 1)

            IF y = LENGTH(String2)
                THEN
                    f ← n
                ELSE
                    x ← x + 1
                    y ← y + 1
            ENDIF

        ENDWHILE

    UNTIL (n = LENGTH(String1)) OR (f <> 0)

    RETURN f

ENDFUNCTION
    
```

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

SSM("RETRACE", "RAC")

n	f	x	y	MID(String1, x, 1)	MID(String2, y, 1)
0	0				



**(b) (i)** Describe the purpose of function `SSM`.

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

**(ii)** One of the possible return values from function `SSM` has a special meaning.

State the value and its meaning.

Value .....

Meaning .....

[2]

**(iii)** There is a problem with the logic of the pseudocode. This could generate a run-time error.

Describe the problem.

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



## Appendix

### Built-in functions

In each function below, if the function call is not properly formed, the function returns an error.

`MID(ThisString : STRING, x : INTEGER, y : INTEGER) RETURNS STRING`

returns the string of length `y` starting at position `x` from `ThisString`

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

`LEFT(ThisString : STRING, x : INTEGER) RETURNS STRING`

returns the leftmost `x` characters from `ThisString`

Example: `LEFT ("ABCDEFGH", 3)` will return string `"ABC"`

`RIGHT(ThisString: STRING, x : INTEGER) RETURNS STRING`

returns the rightmost `x` characters from `ThisString`

Example: `RIGHT ("ABCDEFGH", 3)` will return string `"FGH"`

`ASC(ThisChar : CHAR) RETURNS INTEGER`

returns the ASCII value of character `ThisChar`

Example: `ASC ('w')` will return `87`

`LENGTH(ThisString : STRING) RETURNS INTEGER`

returns the integer value representing the length of string `ThisString`

Example: `LENGTH ("Happy Days")` will return `10`

### String operator

`&` operator

concatenates (joins) two strings

Example: `"Summer" & " " & "Pudding"` produces `"Summer Pudding"`





## QUESTION 16.

14



6 A string-handling function has been developed.

For the built-in functions list, refer to the **Appendix** on the last page.

The pseudocode for this function is shown below.

```
FUNCTION SF(ThisString : STRING) RETURNS STRING
  DECLARE x           : CHAR
  DECLARE NewString  : STRING
  DECLARE Flag       : BOOLEAN
  DECLARE m, n       : INTEGER

  Flag ← TRUE
  NewString ← ""
  m ← LENGTH(ThisString)

  FOR n ← 1 TO m

    IF Flag = TRUE
      THEN
        x ← UCASE(MID(ThisString, n, 1))
        Flag ← FALSE
      ELSE
        x ← LCASE(MID(ThisString, n, 1))
    ENDIF

    NewString ← NewString & x

    IF x = " "
      THEN
        Flag ← TRUE
    ENDIF

  ENDFOR

  RETURN NewString
ENDFUNCTION
```

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

SF("big BEN")

n	x	Flag	m	NewString



(ii) Describe the purpose of function SF.

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

(b) Test data must be designed for the function SF.

(i) State what happens when the function is called with an empty string.

.....  
.....[1]

(ii) The function should be thoroughly tested.

Give **three** examples of non-empty strings that may be used.

In each case explain why the test string has been chosen.

String .....

Explanation .....

.....

String .....

Explanation .....

.....

String .....

Explanation .....

.....

[3]

# QUESTION 17.



3 You will need to refer to the list of pseudocode string-handling functions in the App.

ASCII code table (part)					
Character	Decimal	Character	Decimal	Character	Decimal
<Space>	32	I	73	R	82
A	65	J	74	S	83
B	66	K	75	T	84
C	67	L	76	U	85
D	68	M	77	V	86
E	69	N	78	W	87
F	70	O	79	X	88
G	71	P	80	Y	89
H	72	Q	81	Z	90

(a) For each statement, write the value assigned to the variable.

(i) `Term ← CHARACTERCOUNT ("TSUNAMI")`

Term .....[1]

(ii) `Answer1 ← ASC ('G') + ASC (<Space>)`

Answer1 .....[1]

(iii) `Answer2 ← CHR (CHARACTERCOUNT ("HELLO") + 70)`

Answer2 .....[1]

(iv) `Word ← SUBSTR ("Welcome home", 4, 7) )`

Word .....[1]



**Question 3(b) continues on page 10.**



- (b) A programmer wants to design a procedure to calculate a customer ID from a customer's surname.

The procedure will:

- input the surname
- isolate each character in the surname and find the corresponding ASCII code
- calculate the total of all these ASCII codes
- this total is the customer ID

- (i) Complete the pseudocode for this procedure.

You will need to refer to the list of pseudocode string-handling functions in the Appendix.

```

PROCEDURE CalculateCustomerID

    OUTPUT "Key in surname"

    INPUT Surname

    Length ← .....

    CustomerID ← 0

    FOR i ← 1 TO Length

        // NextChar is a single character from Surname

        NextChar ← .....

        NextCodeNumber ← ASC (NextChar)

        CustomerID ← CustomerID + .....

    ENDFOR

    OUTPUT "Customer ID is ", CustomerID
  
```

[3]





- (c) The programmer decides that it would be better to write the procedure as a function. The user will now input the surname in the main program.

Write **program code** for the following:

State your programming language .....

- (i) The function header for this new function `CalculateCustomerID`  
.....[3]

- (ii) The additional statement required within the function body to complete the change from a procedure to a function.  
.....  
.....[1]

- (iii) The statement in the main program which:
  - calls the function for surname `Wilkes`
  - assigns the result to variable `ThisID`
 .....[3]

- (d) (i) The new function `CalculateUserID` is an example of a 'user-defined function'.

State **two** differences between a built-in function and a user-defined function.

- 1 .....
- .....
- 2 .....
- .....[2]

- (ii) State **two** things that built-in and user-defined functions have in common.

- 1 .....
- .....
- 2 .....
- .....[2]



## QUESTION 18.



3 A string conversion function, `StringClean`, is to be written.

This function will form a new string, `OutString`, from a given string, `InString`, by:

- removing all non-alphabetic characters
- converting all alphabetic characters to lower case.

For example:

```
InString = "Good Morning, Dave"  
OutString = "goodmorningdave"
```

The first attempt at writing the pseudocode for this function is shown below.

Complete the pseudocode using relevant built-in functions.

For the built-in functions list, refer to the **Appendix** on page 14.

```
FUNCTION StringClean(.....) RETURNS .....  
  
    DECLARE NextChar : .....  
  
    DECLARE ..... : STRING  
  
    ..... //initialise the return string  
  
    //loop through InString to produce OutString  
  
    FOR n ← 1 TO ..... //from first to last  
        NextChar ← ..... //get next character and  
        NextChar ← ..... //convert to lower case  
        IF ..... //check if alphabetic  
            THEN  
                ..... //add to OutString  
            ENDIF  
    ENDFOR  
  
    .....//return value  
  
ENDFUNCTION
```

# QUESTION 19.



- 4 (a) High-level programming languages have many features that support the modularity of programming. One such feature is the use of parameters.

State **two** other features.

1 .....

.....

2 .....

.....

[2]

- (b) Consider the following pseudocode.

```
PROCEDURE MyProc (x)
    x ← x + 1
ENDPROCEDURE
```

Intermediate lines of pseudocode not shown

```
x ← 4
CALL MyProc (x)
OUTPUT (x)
```

Parameter  $x$  is used to pass data to procedure `MyProc`.  
There are two parameter passing methods that could be used.

Complete the following table for each of the two methods.

Name of parameter passing method	Value output	Explanation
..... .....	.....	..... ..... .....
..... .....	.....	..... ..... .....

[6]

## QUESTION 20.



- 4 Programming languages provide built-in functions to generate random numbers. To be truly random, the frequency of each number generated should be the same.

You are required to write program code to test the random number generator of your language.

The test should:

- generate a given number of random numbers between 1 and 10 inclusive
- keep a count of the number of times each number is generated
- calculate the expected frequency of each number 1 to 10
- output the actual frequency of each number 1 to 10
- output the difference between the actual frequency and the expected frequency.

The program code should be written as a procedure. In pseudocode, the procedure heading will be:

```
PROCEDURE TestRandom(Repetitions AS INTEGER)
```

The parameter, *Repetitions*, contains a value representing the total number of random numbers that should be generated.

The following example shows the expected output for the procedure call, `TestRandom(200)`.

The expected frequency is 20.

Number	Frequency	Difference
1	17	-3
2	21	1
3	12	-8
4	28	8
5	20	0
6	19	-1
7	21	1
8	16	-4
9	24	4
10	22	2





(b) Name **three** features of a typical IDE that would help a programmer to debug

Explain how each of these could be used in the debugging of the `TestRandom` from **part (a)**.

Feature 1 .....

Explanation .....

.....

.....

.....

Feature 2 .....

Explanation .....

.....

.....

.....

Feature 3 .....

Explanation .....

.....

.....

.....

[6]

(c) The procedure is developed and run using the call `TestRandom(200)`. No system errors are produced.

To ensure that the procedure works correctly, you need to check the output.

Describe **two** checks you should make to suggest the program works correctly.

1 .....

.....

.....

2 .....

.....

.....

[2]







(c) The function `ScanArray()` is one of a number of sub-tasks within a program.

Name the process that involves the splitting of a problem into sub-tasks **and** list two advantages of this approach.

Name .....

Advantage 1 .....

.....

Advantage 2 .....

.....

[3]

(d) `ResultArray` is a 2D array of type `STRING`. It represents a table containing 100 rows and 2 columns.

Write **program code** to declare `ResultArray` **and** set all elements to the value `'*'`.

Programming language .....

Program code

.....

.....

.....

.....

.....

.....

.....

.....[3]





**Question 5 begins on the next page.**

## QUESTION 22.



4 The following pseudocode is a string handling function.

For the built-in functions list, refer to the **Appendix** on page 16.

```
FUNCTION Clean(InString : STRING) RETURNS STRING

    DECLARE NewString : STRING
    DECLARE Index : INTEGER
    DECLARE AfterSpace : BOOLEAN
    DECLARE NextChar : CHAR
    CONSTANT Space = ' '

    AfterSpace ← FALSE
    NewString ← ""

    FOR Index ← 1 TO LENGTH(InString)
        NextChar ← MID(InString, Index, 1)
        IF AfterSpace = TRUE
            THEN
                IF NextChar <> Space
                    THEN
                        NewString ← NewString & NextChar
                        AfterSpace ← FALSE
                    ENDFIF
            ELSE
                NewString ← NewString & NextChar
                IF NextChar = Space
                    THEN
                        AfterSpace ← TRUE
                    ENDFIF
            ENDFIF
        ENDFOR

    RETURN NewString

ENDFUNCTION
```



- (a) (i) Complete the trace table by performing a dry run of the function which follows:

Result  $\leftarrow$  Clean("xYVandZ")

The symbol 'V' represents a space character. Use this symbol to represent a space character in the trace table.

Index	AfterSpace	NextChar	NewString

[6]

- (ii) State the effect of the function Clean().

.....

.....

[1]



(iii) The pseudocode is changed so that the variable `AfterSpace` is initialised

Explain what will happen if the function is called as follows:

```
Result ← Clean ("XandZ")
```

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

(b) The following pseudocode declares and initialises an array.

```
DECLARE Code : ARRAY[1:100] OF STRING
DECLARE Index : INTEGER

FOR Index ← 1 TO 100
  Code[Index] ← ""
ENDFOR
```

The design of the program is changed as follows:

- the array needs to be two dimensional, with 500 rows and 4 columns
- the elements of the array need to be initialised to the string "Empty"

Re-write the **pseudocode** to implement the new design.

.....  
.....  
.....  
.....  
.....  
.....  
.....  
.....  
.....  
..... [4]

(c) State the term used for changes that are made to a program in response to a specification change.

..... [1]



**Question 5 begins on the next page.**

## QUESTION 23.



4 The following is pseudocode for a string handling function.

For the built-in functions list, refer to the **Appendix** on page 16.

```
FUNCTION Search(InString : STRING) RETURNS INTEGER
```

```
    DECLARE NewString : STRING
    DECLARE Index : INTEGER
    DECLARE NextChar : CHAR
    DECLARE Selected : INTEGER
    DECLARE NewValue : INTEGER
```

```
    NewString ← '0'
    Selected ← 0
```

```
    FOR Index ← 1 TO LENGTH(InString)
```

```
        NextChar ← MID(InString, Index, 1)
        IF NextChar < '0' OR NextChar > '9'
            THEN
                NewValue ← STRING_TO_NUM(NewString)
                IF NewValue > Selected
                    THEN
                        Selected ← NewValue
                    ENDIF
                NewString ← '0'
            ELSE
                NewString ← NewString & NextChar
            ENDIF
```

```
    ENDFOR
```

```
    RETURN Selected
```

```
ENDFUNCTION
```



(a) (i) The following assignment calls the `Search()` function:

```
Result ← Search("12∇34∇5∇∇39")
```

Complete the following trace table by performing a dry run of this function call.

The symbol '∇' represents a space character. Use this symbol to represent a space character in the trace table.

Index	NextChar	Selected	NewValue	NewString

[5]

(ii) State the value returned by the function when it is called as shown in **part (a)(i)**.

.....

[1]



(b) There is an error in the algorithm. When called as shown in **part (a)(i)**, the return the largest value as expected.

(i) Explain why this error occurred when the program called the function.

.....

.....

.....

..... [2]

(ii) Describe how the algorithm could be amended to correct the error.

.....

.....

.....



## QUESTION 24.



- 5 Nigel is learning about string handling. He wants to write code to count the number of words in a given string. A word is defined as a sequence of alphabetic characters that is separated by one or more space characters.

His first attempt at writing an algorithm in pseudocode is as follows:

```
PROCEDURE CountWords (Message : STRING)

    DECLARE NumWords : INTEGER
    DECLARE Index : INTEGER
    CONSTANT Space = ' '

    NumWords ← 0

    FOR Index ← 1 TO LENGTH(Message)
        IF MID(Message, Index, 1) = Space
            THEN
                NumWords ← NumWords + 1
            ENDIF
        ENDFOR

    OUTPUT "Number of words : " , NumWords

ENDPROCEDURE
```

For the built-in functions list, refer to the **Appendix** on page 18.

His first attempt is incorrect. He will use white-box testing to help him to identify the problem.

- (a) (i) State the purpose of white-box testing.

.....  
..... [1]

- (ii) Dry running the code is often used in white-box testing. In this method, the programmer records the values of variables as they change.

Identify what the programmer would normally use to record the changes.

..... [1]



(b) (i) Write a test string containing two words that gives the output:

Number of words : 2

Use the symbol '▽' to represent each space character in your test string.

Explain why the algorithm gives the output shown above.

String .....

Explanation .....

.....  
.....  
.....  
.....

[3]

(ii) Nigel tested the procedure with the strings:

String 1: "Red▽and▽Yellow"

String 2: "Green▽▽and▽▽Pink▽"

Give the output that is produced for each of the strings.

Describe the changes that would need to be made to the algorithm to give the correct output in each case.

Do **not** write pseudocode **or** program code.

String 1 .....

Description .....

.....  
.....  
.....

String 2 .....

Description .....

.....  
.....  
.....

[6]

## QUESTION 25.

5 The following pseudocode checks whether a string is a valid password.

```
FUNCTION CheckPassword(InString : STRING) RETURNS BOOLEAN

    DECLARE Index, Upper, Lower, Digit, Other : INTEGER
    DECLARE NextChar : CHAR

    Upper ← 0
    Lower ← 0
    Digit ← 0
    Other ← 0

    FOR Index ← 1 TO LENGTH(InString)

        NextChar ← MID(InString, Index, 1)
        IF NextChar >= 'A' AND NextChar <= 'Z'
            THEN
                Upper ← Upper + 1
            ELSE
                IF NextChar >= 'a' AND NextChar <= 'z'
                    THEN
                        Lower ← Lower + 1
                    ELSE
                        IF NextChar >= '0' AND NextChar <= '9'
                            THEN
                                Digit ← Digit + 1
                            ELSE
                                Other ← Other + 1
                        ENDIF
                    ENDIF
                ENDIF
            ENDIF
        ENDIF

    ENDFOR

    IF Upper > 1 AND Lower >= 5 AND (Digit - Other) > 0
        THEN
            RETURN TRUE
        ELSE
            RETURN FALSE
        ENDIF

ENDFUNCTION
```

(a) Describe the validation rules that are implemented by this pseudocode. Refer **only** to the contents of the string and **not** to features of the pseudocode.

.....

.....

.....

.....

.....

.....

.....

.....





(b) (i) Complete the trace table by dry running the function when it is called as

```
Result ← CheckPassword("Jim+Smith*99")
```

<b>Index</b>	<b>NextChar</b>	<b>Upper</b>	<b>Lower</b>	<b>Digit</b>	<b>Other</b>

[5]

(ii) State the value returned when the function is called using the expression shown. Justify your answer.

Value .....

Justification .....

.....

.....

[2]

## QUESTION 26.



4 The following pseudocode algorithm checks whether a string is a valid email address.

```
FUNCTION Check(InString : STRING) RETURNS BOOLEAN

    DECLARE Index : INTEGER
    DECLARE NumDots : INTEGER
    DECLARE NumAsts : INTEGER
    DECLARE NextChar : CHAR
    DECLARE NumOthers : INTEGER

    NumDots ← 0
    NumAsts ← 0
    NumOthers ← 0

    FOR Index ← 1 TO LENGTH(InString)

        NextChar ← MID(InString, Index, 1)
        CASE OF NextChar
            '.': NumDots ← NumDots + 1
            '@': NumAsts ← NumAsts + 1
            OTHERWISE NumOthers ← NumOthers + 1
        ENDCASE

    ENDFOR

    IF (NumDots >= 1 AND NumAsts = 1 AND NumOthers > 5)
        THEN
            RETURN TRUE
        ELSE
            RETURN FALSE
    ENDIF

ENDFUNCTION
```

(a) Describe the validation rules that are implemented by this pseudocode. Refer **only** to the contents of the string and **not** to features of the pseudocode.

.....

.....

.....

..... [3]



(b) (i) Complete the trace table by dry running the function when it is called as

Result ← Check("Jim.99@skail.com")

Index	NextChar	NumDots	NumAts	NumOthers

[5]

(ii) State the value returned when function Check is called as shown in part (b)(i).

..... [1]



(c) The function `Check()` is to be tested.

State **two** different invalid string values that could be used to test the algorithm. Each value should test a different rule.

Justify your choices.

Value .....

Justification .....

.....

.....

Value .....

Justification .....

.....