QUESTION 1.

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]

Question 5 begins on page 12.

[Turn over

QUESTION 2.

5 A firm employs workers who assemble amplifiers. Each member of staff works a of 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]

Write the **program code**. Do not attempt to include any validation checks.

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 ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[10]

QUESTION 3.

7 ASCII character codes are used to represent a single character.

Part of the code table is shown below.

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

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

`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("BRAZIL")` returns 6.

`CHR(ThisInteger : INTEGER) RETURNS CHAR`
returns the character with ASCII code `ThisInteger`.
For example: `CHR(65)` returns character 'A'.

`ASC(ThisCharacter : CHAR) RETURNS INTEGER`
returns the ASCII value for character `ThisCharacter`.
For example: `ASC('A')` returns 65.

(a) Show the values stored by variables `A`, `B`, `C` and `D`.

The & operator is used to concatenate two strings.

```
Num1 ← 15
A ← CHR(67) & CHR(65) & CHR(84)
B ← ASC('P') - ASC('F') + 3
C ← ASC(ONECHAR("BISCUITS", 3))
D ← CHARACTERCOUNT("New York City") + 2
```

(i) A .... [1]

(ii) B .... [1]

(iii) C .... [1]

(iv) D .... [1]

(b) A program is to be written which accepts a string and then calculates a nu this string. The input string and the calculated value are then to be sent to a rem over a communications link.

Study the following pseudocode:

```
OUTPUT "Enter string"
INPUT MyString
StringTotal ← 0

FOR i ← 1 TO CHARACTERCOUNT(MyString)
   NextNum ← ASC(ONECHAR(MyString, i))
   StringTotal ← StringTotal + NextNum
ENDFOR

OUTPUT MyString, StringTotal
```

Write the above pseudocode algorithm as **program code**.

There is no need to show the declaration of variables or comment statements.

....

....

....

....

....

....

....

....

....

....

....[6]

(c) Explain the purpose of sending the value of `StringTotal` to the remote computer, in addition to `MyString`.

....

....

....

....[2]

QUESTION 4.

7 ASCII character codes are used to represent a single character.

Part of the code table is shown below.

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

Some pseudocode statements follow which use these built-in functions:

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

`CHR(ThisInteger : INTEGER) RETURNS CHAR`
returns the character with ASCII value `ThisInteger`.
For example: `CHR(66)` returns 'B'.

`ASC(ThisCharacter : CHAR) RETURNS INTEGER`
returns the ASCII value for character `ThisCharacter`.
For example: `ASC('B')` returns 66.

(a) Give the values assigned to the variables A, B, C and D.

The & operator is used to concatenate two strings.

The expression could generate an error; if so, write ERROR.

```
Num1 ← 5
A ← ASC('F') + Num1 + ASC('Z')
B ← CHR(89) & CHR(69) & CHR(83)
C ← CHARACTERCOUNT(B & "PLEASE")
D ← ASC(ONECHAR("CURRY SAUCE", 7))
```

(i) A .... [1]

(ii) B .... [1]

(iii) C .... [1]

(iv) D .... [1]

(ii) An experienced programmer suggests this pseudocode would be best designed as a 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
```

[7]

QUESTION 5.

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

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				

[6]

(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"`**

BLANK PAGE

QUESTION 6.

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

[4]

Write **program code** for the procedure `OutputLocationList`.

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 ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [10]

QUESTION 7.

4 A company employs Ahmed as a programmer.

(a) At College, before joining the company, Ahmed used two items of software for pro

- a text editor
- a compiler

Describe how he could have developed programs using these software tools.

Include in the description the terms ‘object code’ and ‘source code’.

....

....

....

....

....

....

....[3]

(b) Ahmed now uses an Integrated Development Environment (IDE) for programming.

(i) State **one** feature an IDE provides to help with the identification of syntax errors.

....

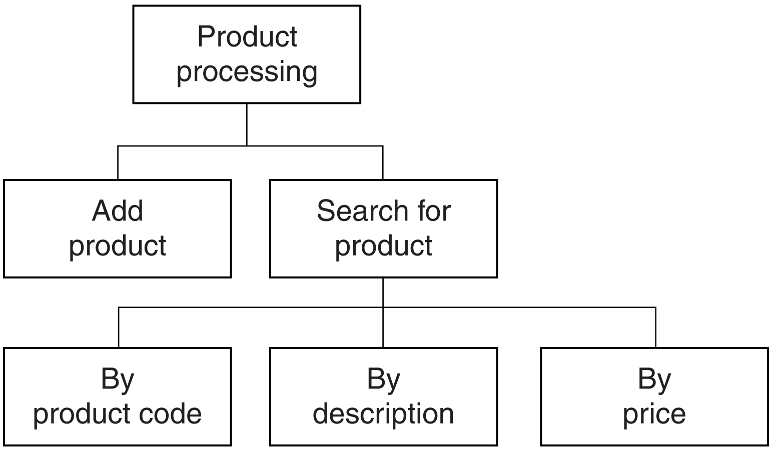
....[1]

(ii) State **one** feature an IDE provides to carry out white box testing.

....

....[1]

(c) The company maintains a file of product data. Ahmed is to write a program to add a new product and search for a product based on the structure diagram shown:



The program records the following data for each product:

- product code
- product description
- product retail price

The text file `PRODUCTS` stores each data item on a separate line, as shown below:

File **`PRODUCTS`**

`0198`
`Plums(10kg)`
`11.50`
`0202`
`Onions(20kg)`
`10.00`
`0376`
`Mango chutney(1kg)`
`02.99`
`0014`
`Mango(10kg)`
`12.75`

The program uses the variables shown in the identifier table.

Identifier	**Data type**	**Description**
`PRODUCTS`	`TEXT FILE`	Storing the code, description and retail price for all current products
`PCode`	`ARRAY[1:1000] OF STRING`	Array storing the product codes
`PDescription`	`ARRAY[1:1000] OF STRING`	Array storing the product descriptions
`PRetailPrice`	`ARRAY[1:1000] OF REAL`	Array storing the product retail prices
`i`	`INTEGER`	Array index used by all three arrays

(i) The first operation of the program is to read all the product data held i and write them into the three 1D arrays.

Complete the pseudocode below.

```
OPEN ..........................................................................

i ← 1

WHILE .........................................................................

  READFILE ("PRODUCTS", .....................................................)

  READFILE ("PRODUCTS", .....................................................)

  READFILE ("PRODUCTS", .....................................................)

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

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

ENDWHILE

CLOSE "PRODUCTS"

OUTPUT "Product file contents written to arrays"
```

[5]

When Ahmed designed the PRODUCTS file, he considered the alternative file structure shown opposite.

It stores one product per line in the text file.

File **PRODUCTS**

0198	Plums(10kg)	11.50
0202	Onions(20kg)	10.00
0376	Mango chutney(1kg)	02.99
0014	Mango(10kg)	12.75

(ii) State **one** benefit and **one** drawback of this file design.

Benefit ....

....

Drawback ....

....[2]

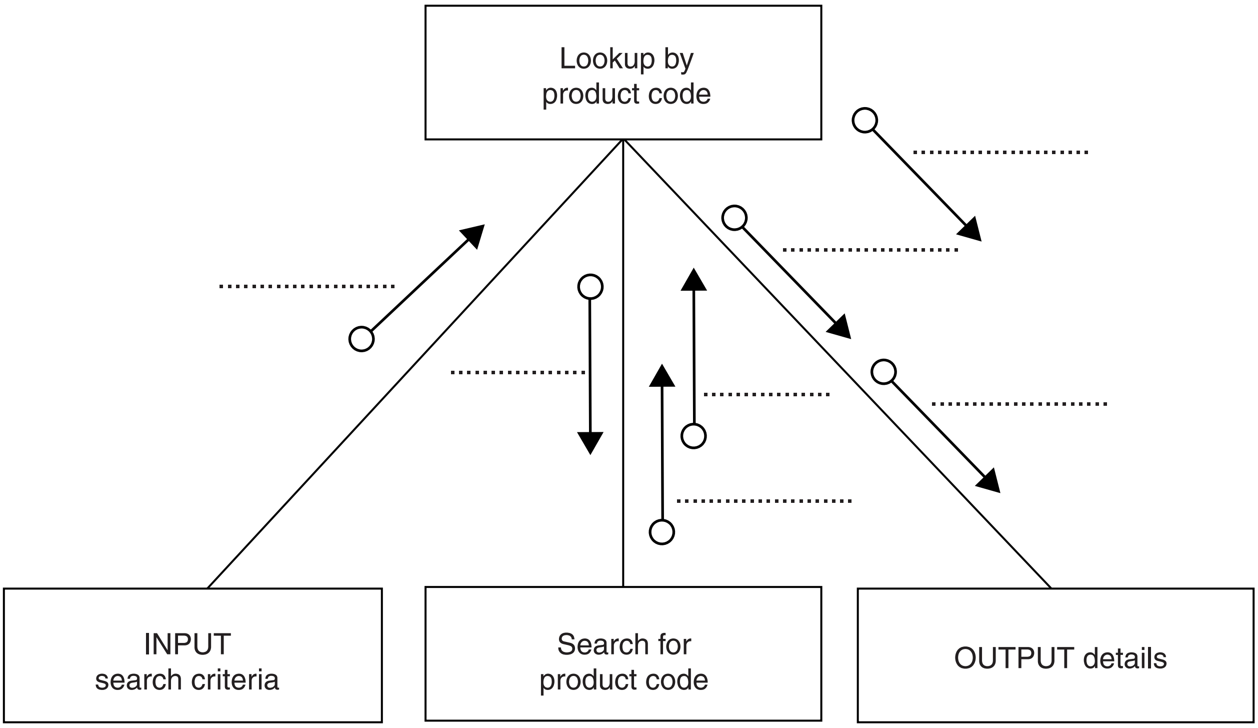
(d) To code the 'Search by product code' procedure, Ahmed draws a structure c different stages.

The procedure uses the variables shown in the identifier table.

Identifier	**Data type**	**Description**
`SearchCode`	`STRING`	Product code input by the user
`ThisIndex`	`INTEGER`	Array index position for the corresponding product
`ThisDescription`	`STRING`	Product description found
`ThisRetailPrice`	`REAL`	Product retail price found

You can assume that before the procedure is run, all the product data is read from file `PRODUCTS` and then stored in three 1D arrays as described in **part (c)(i)**.

Label the structure chart to show the input(s) and output(s).



[4]

(e) A first attempt was made at writing the 'Search for product code' module. Ahmed designs this as a function `ProductCodeSearch`.

The function returns an integer value as follows:

- if the product code is found, it returns the index position of the 1D array `PCode` be searched
- if the product code is not found, the function returns -1

Write **program code** for function `ProductCodeSearch`.

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 ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[6]

QUESTION 8.

5 A team keeps a record of the scores made by each of their eight players in a num

The data in the two tables below shows:

- the scores of the eight players after twenty games
- the eight player names.

	1	2	3		8
1	12	17	67		31
2	35	82	44		29
3	61	39	80		17
4	81	103	21		11
5	56	0	98		4
…					
19	45	6	81		77
20	12	11	3		6

1	Vorma
2	Ravi
3	Chada
4	Nigam
5	Bahri
6	Smith
7	Goyal
8	Lata

The team wants a computer program to input and record the player data.

(a) A programmer designs the following pseudocode for the input of a player's score from one game.

```
INPUT GameNumber
INPUT PlayerNumber
INPUT PlayerGameScore
PlayerScore[GameNumber, PlayerNumber] ← PlayerGameScore
```

Describe the data structure the programmer has used for the storage of all player scores.

.... [2]

(b) The player names are permanently stored in a text file NAMES.TXT, with on line. The player names will be read by the program and stored in a 1D array.

The design given in **part (a)** will be expanded so that the user is prompted for th name instead of the player number. Step 02 now becomes:

```
02.1    Read the player names from file NAMES.TXT into the array PlayerName
02.2    INPUT ThisPlayerName
02.3    Search the PlayerName array for ThisPlayerName to find the PlayerNumber
```

(i) State the computing term for the expansion of one or more steps in the original design.

.... [1]

(ii) Write the **program code** for step 02.1

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 ....

....

....

....

....

....

....

....

....

....

....

....

.... [4]

(iii) Program code is to be designed and written for step `02.3`

The program will use these identifiers:

Identifier	Data type	Description
`PlayerName`	`ARRAY[1 : 8] OF STRING`	Stores the player names (read from the file)
`ThisPlayerName`	`STRING`	Input by the user (step `02.2`)
`Found`	`BOOLEAN`	Flags when `ThisPlayerName` is found when searching the `PlayerName` array
`i`	`INTEGER`	Array index

Write **program code** to carry out the linear search for step `02.3`

There is no requirement to declare or comment on variables used.

Programming language ....

....

....

....

....

....

....

....

....

....

....

....

.... [4]

(c) The team wants the program to produce a report, with the following specifica

The program outputs the total number of player scores that are:

- 50 and over but less than 100
- 100 or higher.

You can assume that before the section runs, the program has assigned all eight player scores to the `PlayerScore` data structure.

A first attempt at the pseudocode is shown below:

```
01 Total50 ← 0
02 Total100 ← 0
03 FOR PlayerIndex  ← 1 TO 8
04   FOR GameIndex ← 1 TO 20
05     IF PlayerScore[GameIndex, PlayerIndex] > 100
06       THEN
07         Total100 ← Total100 + 1
08       ELSE
09         IF PlayerScore[GameIndex, PlayerIndex] > 50
10           THEN
11             Total50 ← Total50 + GameIndex
12         ENDIF
13     ENDIF
14   ENDFOR
15 ENDFOR
16 OUTPUT Total50
17 OUTPUT Total100
```

(i) Describe the control structure used in lines `03` and `04` and lines `14` and `15`.

....

....

.... [2]

(ii) Consider the following two statements.

Write either TRUE **or** FALSE next to each statement.

Statement	**TRUE or FALSE**
The pseudocode considers all the scores for a player, before progressing to the next player.	
The pseudocode considers all scores in a game, before progressing to the next game.	

[1]

(iii) The programmer has made logic errors in the design.

State a line number at which an error occurs.

Explain the error or write the corrected pseudocode statement.

Line number ....

Explanation ....

.... [1]

QUESTION 9.

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
```

[11]

QUESTION 10.

3 A string conversion function, `ExCamel`, needs to be written.

This function forms a return string, `OutString`, from a given string, `InString`, by:

1 separating the original words (a word is assumed to start with a capital letter)
2 converting all characters to lower case.

The following shows a pair of example values for the string values `InString` and `OutString`.

```
InString : "MyUserInput"
OutString : "my user input"
```

You may assume that `InString` always starts with a capital letter.

The following is a first attempt at writing the pseudocode for this function.

Complete the **pseudocode** using appropriate built-in functions.

For the built-in functions list, refer to the **Appendix** on page 13.

```
FUNCTION ExCamel (............................) RETURNS .............

    DECLARE NextChar : ............................................

    DECLARE ...................................................... : STRING

    DECLARE n: INTEGER

    ........................................... // initialise the return string

    // loop through InString to produce OutString

    FOR n ← 1 TO ............................... // from first to last

        NextChar ← ............................. // get next character

        IF ..................................... // check if upper case

            THEN

                IF n > 1                         // if not first character

                    THEN

                        ........................ // add space to OutString

                ENDIF

                ................................ // make NextChar lower case

        ENDIF

        ........................................ // add NextChar to OutString

    ENDFOR

    ............................................ // return value

ENDFUNCTION
```

[11]

QUESTION 11.

2 The following pseudocode represents a simple algorithm.

```
DECLARE NumberFound, Remainder, Number : INTEGER
DECLARE StartNumber, EndNumber, Divisor : INTEGER

INPUT StartNumber
INPUT EndNumber
INPUT Divisor
NumberFound ← 0

FOR Number ← StartNumber TO EndNumber
   Remainder ← MODULUS(Number, Divisor)
   IF Remainder = 0
      THEN
         OUTPUT Number
         NumberFound ← NumberFound + 1
   ENDIF
ENDFOR
OUTPUT "Count: " & NumberFound
```

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

(a) Complete the following trace table.

StartNumber	**EndNumber**	**Divisor**	**NumberFound**	**Number**	**Remainder**	**Output**
11	13	2	0			

[3]

(b) Describe the purpose of this algorithm.

....

....

....

....

....

....[3]

(c) Draw a program flowchart to represent this algorithm.

Variable declarations are **not** required in program flowcharts.

[10]

QUESTION 12.

3 A 1D array, `Product`, of type `STRING` is used to store information about a range shop. There are 100 elements in the array. Each element stores one data item.

The format of each data item is as follows:

`<ProductID><ProductName>`

- `ProductID` is a four-character string of numerals
- `ProductName` is a variable-length string

The following pseudocode is an initial attempt at defining a procedure, `ArraySort`, which will perform a bubble sort on `Product`. The array is to be sorted in ascending order of `ProductID`. Line numbers have been added for identification purposes only.

```
PROCEDURE SortArray
  DECLARE Temp : CHAR
  DECLARE FirstID, SecondID : INTEGER
  FOR I ← 1 TO 100
    FOR J ← 2 TO 99
       FirstID ← MODULUS(LEFT(Product[J], 6))
       SecondID ← MODULUS(LEFT(Product[J + 1], 6))
       IF FirstID > SecondID
          THEN
             Temp ← Product[I]
             Product[I] ← Product[J + 1]
             Product[J + 1] ← Temp
     ENDFOR
       ENDIF
  ENDFOR
ENDPROCEDURE
```

The pseudocode on page 8 contains a number of errors. Complete the following

- the line number of the error
- the error itself
- the correction that is required.

Note:

- If the same error occurs on more than one line, you should only refer to it ONCE.
- Lack of optimisation should not be regarded as an error.

Line number	Error	Correction
01	Wrong procedure name – "SortArray"	`PROCEDURE ArraySort`

[8]

QUESTION 13.

4 Part of a program written in pseudocode is shown.

```
01  DECLARE NumElements : INTEGER

…

10  FUNCTION ScanArray(SearchString : STRING) RETURNS INTEGER
11
12     DECLARE ArrayIndex : INTEGER
13     DECLARE ArrayString : STRING
14     DECLARE NumberFound : INTEGER
15
16     ArrayIndex ← 0
17     NumberFound ← 0
18
19     FOR ArrayIndex ← 1 TO NumElements
20        ArrayString ← ResultArray[ArrayIndex, 1]
21        IF ArrayString = SearchString
22           THEN
23              CALL SaveToFile(ArrayString)
24              NumberFound ← NumberFound + 1
25        ENDIF
26     ENDFOR
27
28     RETURN NumberFound
29
30  ENDFUNCTION
```

(a) **(i)** Examine the pseudocode **and** complete the following table.

	Answer
The identifier name of a global integer	
The identifier name of a user-defined procedure	
The line number of an unnecessary statement	
The scope of `ArrayString`	

[4]

(ii) Describe in detail the purpose of lines `19` to `26` in the function `ScanArray()`.
Do **not** use pseudocode in your answer.

....

....

....

....

....

....

....[4]

(b) The function `ScanArray()` needs to be amended so that the comparis sensitive. For example, comparing "`Aaaa`" with "`AAAa`" should evaluate to `TRUE`

Write **program code** to implement the **amended** `ScanArray()` function.

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]

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

Name the process that involves the splitting of a problem into sub-tasks **and** 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 14.

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
            ENDIF
         ELSE
            NewString ← NewString & NextChar
            IF NextChar = Space
               THEN
                  AfterSpace ← TRUE
            ENDIF
      ENDIF
   ENDFOR

   RETURN NewString

ENDFUNCTION
```

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

```
Result ← Clean("X∇∇∇Y∇and∇∇Z")
```

The symbol '∇' represents a space character. Use this symbol to represent a spa
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 initialis

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

```
Result ← Clean("∇∇X∇∇∇Y∇and∇∇Z")
```

....

....

....

.... [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 15.

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 16.

5 Nigel is learning about string handling. He wants to write code to count the numb given string. A word is defined as a sequence of alphabetic characters that is separa 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 17.

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

   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.

....

....

....

....

....

.... [3]

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

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

Index	**NextChar**	**Upper**	**Lower**	**Digit**	**Other**

[5]

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

Value ....

Justification ....

....

....

[2]

QUESTION 18.

1 Study the following pseudocode.

```
PROCEDURE FillTank()

   DECLARE Tries : INTEGER
   DECLARE Full : BOOLEAN

   Tries ← 1

   Full ← ReadSensor("F1")

   IF NOT Full
      THEN
         WHILE NOT Full AND Tries < 4
            CALL TopUp()
            Full ← ReadSensor("F1")
            Tries ← Tries + 1
         ENDWHILE
         IF Tries > 3
            THEN
               OUTPUT "Too many attempts"
            ELSE
               OUTPUT "Tank now full"
         ENDIF
      ELSE
         OUTPUT "Already full"
   ENDIF

ENDPROCEDURE
```

(a) **(i)** The pseudocode includes features that make it easier to read and understand.

State **three** such features.

Feature 1 ....

Feature 2 ....

Feature 3 ....
[3]

(ii) Draw a program flowchart to represent the algorithm implemented in the
Variable declarations are not required in program flowcharts.

[5]

(b) (i) Programming languages support different data types.

Complete the table by giving a suitable data type for each example value.

Example value	**Data type**
`43`	
`TRUE`	
`-273.16`	
`"-273.16"`	

[4]

(ii) Evaluate each expression in the following table.

If an expression is invalid then write 'ERROR'.

Refer to the **Appendix** on page 18 for the list of built-in functions and operators.

Expression	**Evaluates to**
`RIGHT("Stop", 3) & LEFT("ich", 2)`	
`MID(NUM_TO_STRING(2019), 3, 1)`	
`INT(NUM_TO_STRING(-273.16))`	
`INT(13/2)`	

[4]

QUESTION 19.

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

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

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

   NumDots ← 0
   NumAts ← 0
   NumOthers ← 0

   FOR Index ← 1 TO LENGTH(InString)

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

   ENDFOR

   IF (NumDots >= 1 AND NumAts = 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. should test a different rule.

Justify your choices.

Value ....

Justification ....

....

....

Value ....

Justification ....

....