QUESTION 1.

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]

(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]

Appendix

Built-in functions (Pseudocode)

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"`**

```
CHR(x : INTEGER) RETURNS CHAR
```

returns the character whose ASCII value is `x`
Example: **`CHR(87)`** will return **`'W'`**

```
ASC(x : CHAR) RETURNS INTEGER
```

returns the ASCII value of character `x`
Example: **`ASC('W')`** will return **`87`**

```
LCASE(x : CHAR) RETURNS CHAR
```

QUESTION 2.

6 A multi-user computer system makes use of passwords.

To be valid, a password must comply with the following rules:

- at least two lower-case alphabetic characters
- at least two upper-case alphabetic characters
- at least three numeric characters
- alpha-numeric characters only

A function, `ValidatePassword`, is needed to check that a given password follows these rules. This function takes a string, `Pass`, as a parameter and returns a Boolean value:

- `TRUE` if `Pass` contains a valid password
- `FALSE` otherwise.

(a) Write **program code** to implement the new function `ValidatePassword`.

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

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[10]

(b) (i) The function will be tested.

Give a valid string to check that the function returns `TRUE` under the correct c

String1: .....

Modify the valid string given for String1 to test each rule separately.

Explain your choice in each case.

String2: .....

Explanation: .....

.....

.....

String3: .....

Explanation: .....

.....

.....

String4: .....

Explanation: .....

.....

.....

String5: .....

Explanation: .....

.....

.....

[5]

(ii) When testing a module, it is necessary to test all possible paths through the code.

State the name given to this type of testing.

.....[1]

(iii) A program consisting of several modules may be tested using a pr
stub testing.

Explain this process.

.....

.....

.....

.....[2]

Appendix

Built-in functions (pseudocode)

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

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

returns string of length `y` starting at position `x` from `ThisString`.
Example: **`MID("ABCDEFGH", 2, 3)`** returns string **`"BCD"`**

```
LENGTH(ThisString : STRING) RETURNS INTEGER
```

returns the integer value representing the length of string `ThisString`.
Example: **`LENGTH("Happy Days")`** returns **`10`**

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

returns leftmost `x` characters from `ThisString`.
Example: **`LEFT("ABCDEFGH", 3)`** returns string **`"ABC"`**

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

returns rightmost `x` characters from `ThisString`.
Example: **`RIGHT("ABCDEFGH", 3)`** returns string **`"FGH"`**

```
LCASE(ThisChar : CHAR) RETURNS CHAR
```

returns the character value representing the lower case equivalent of `ThisChar`.
If `ThisChar` is not an upper-case alphabetic character then it is returned unchanged.
Example: **`LCASE('W')`** returns **`'w'`**

```
MOD(ThisNum : INTEGER, ThisDiv : INTEGER) RETURNS INTEGER
```

returns the integer value representing the remainder when `ThisNum` is divided by `ThisDiv`.
Example: **`MOD(10,3)`** returns **`1`**

```
DIV(ThisNum : INTEGER, ThisDiv : INTEGER) RETURNS INTEGER
```

returns the integer value representing the whole number part of the result when `ThisNum` is divided by `ThisDiv`.
Example: **`DIV(10,3)`** returns **`3`**

Operators (pseudocode)

Operator	Description
`&`	Concatenates (joins) two strings. Example: **`"Summer" & " " & "Pudding"`** produces **`"Summer Pudding"`**
`AND`	Performs a logical **`AND`** of two Boolean values. Example: **`TRUE AND FALSE`** produces **`FALSE`**
`OR`	Performs a logical **`OR`** of two Boolean values. Example: **`TRUE OR FALSE`** produces **`TRUE`**

BLANK PAGE

QUESTION 3.

5 A program collects data about the performance of a car at regular time inter
`CarStatus.txt`, stores the data.

The format of each line of the text file is as follows:

```
<Time>,<Amount of fuel used>,<Distance travelled>
```

Data items are separated by a ',' (comma) character.

The program contains the following functions.

Function	Description
`GetTime()`	Returns a string representing the current time. May return `NULL` under certain circumstances.
`GetFuel()`	Returns a string representing the amount of fuel used
`GetDistance()`	Returns a string representing the distance travelled

The function `SaveStatus()` will:

- obtain the time, fuel used and distance data using the appropriate function calls
- check that the time string is not `NULL`
- return `FALSE` if the current time string remains `NULL` after three attempts
- form the text string, write it to the file and return `TRUE`

The file should not be open longer than necessary.

15

Write **pseudocode** for the `SaveStatus()` function.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[10]

Appendix

Built-in functions (pseudocode)

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

`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 string `"BCD"`

`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 string `"ABC"`

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

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

`TO_UPPER(ThisString : STRING) RETURNS STRING`
returns a string formed by converting all lower case alphabetic characters of `ThisString` to upper case. Other characters will be unchanged.

Example: `TO_UPPER("Disk Error 27")` returns `"DISK ERROR 27"`

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

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

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

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

Operators (pseudocode)

QUESTION 4.

5 The procedure `LineNumber()` will:

- read data from a text file
- modify each line by adding a line number, and the string (`": "`)
- output each modified line.

For example, when the procedure reads `MyFile.txt`, the output is:

```
10: <First line of MyFile.txt>
15: <Second line of MyFile.txt>
20: <Third line of MyFile.txt>
...
350: <Last line of MyFile.txt>
```

The procedure takes three parameters:

Identifier	Data type	Description
FileName	`STRING`	The name of the text file
StartNumber	`INTEGER`	The first line number to be added
StepNumber	`INTEGER`	The line number increment

In this example, the procedure call would be:

```
CALL LineNumber("MyFile.txt", 10, 5)
```

After every 20 lines, the program outputs a message asking whether the user wants to continue.

The program ends when the user enters an `'N'` or the end of file is reached.

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

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.... [11]

Appendix

Built-in functions (pseudocode)

Each function returns an error if the function call is not properly formed.

`MID(ThisString : STRING, x : INTEGER, y : INTEGER) RETURNS STRING`
returns a string of length `y` starting at position `x` from `ThisString`

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

`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 string `"ABC"`

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

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

`NUM_TO_STRING(x : REAL) RETURNS STRING`
returns a string representation of a numeric value.

Example: If `x` has the value `87.5` then `NUM_TO_STRING(x)` will return `"87.5"`
Note: This function will also work if `x` is of type integer.

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

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

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

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

Operators (pseudocode)

QUESTION 5.

5 The function `ReadFileLine()` returns a specific line from a text file.

The function takes two parameters:

Identifier	Data type	Description
`FileName`	`STRING`	The name of the text file
`FileLine`	`INTEGER`	The line number that is required

The following pseudocode gives an example of the use of the function.

```
FileData ← ReadFileLine(FileName, FileLine)
```

The function `ReadFileLine()` will:

- open the file, `FileName`
- read each line from the file until line `FileLine` is found or the end of the file is reached
- if the line exists, return the string from this line; otherwise return the string `"****"`

15

Write **pseudocode** for the `ReadFileLine()` function.

[10]

Appendix

Built-in functions (pseudocode)

Each function returns an error if the function call is not properly formed.

```
MID(ThisString : STRING, x : INTEGER, y : INTEGER) RETURNS STRING
```
returns a string of length `y` starting at position `x` from `ThisString`

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

```
LEFT(ThisString : STRING, x : INTEGER) RETURNS STRING
```
returns leftmost `x` characters from `ThisString`

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

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

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

```
LENGTH(ThisString : STRING) RETURNS INTEGER
```
returns the integer value representing the length of `ThisString`

Example: `LENGTH("Happy Days")` returns `10`

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

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

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

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

```
IS_NUM(ThisString : STRING) RETURNS BOOLEAN
```
returns the value `TRUE` if `ThisString` contains only numeric characters (`'0'` to `'9'`).

Example: `IS_NUM("1234X67")` returns `FALSE`

Operators (pseudocode)

QUESTION 6.

6 Nadine is developing a program to store the ID and preferred name for each stud
For example, student Pradeep uses the preferred name “Prad”.

The program will:

1. prompt and input a valid user ID and a preferred name
2. write the user ID and preferred name to one of two files
3. allow the user to end the program or repeat from step 1.

The program will consist of three separate modules. Each module will be implemented using either a procedure or a function.

Nadine has defined the modules as follows:

Module	Description
`TopLevel()`	• Call `GetInfo()` to obtain a string containing a valid user ID and a preferred name • Call `WriteInfo()` to write the string to either `File1.txt` or `File2.txt` depending on the first character of the user ID as follows: ○ ‘A’ to ‘M’: writes to `File1.txt` ○ ‘N’ to ‘Z’: writes to `File2.txt` For example, a string with a user ID of `"G1234"` writes to `File1.txt` • End the program if the file write was unsuccessful • Input (Y/N) to either repeat for the next user ID or to end the program
`GetInfo()`	• Input a user ID and repeat until the user ID is valid • Input a preferred name. This will be an empty string if no preferred name is input. • Concatenate the user ID and preferred name using a `'*'` character as a separator and return this string
`WriteInfo()`	• Open the file • Append the concatenated string to the file • Close the file • Return a Boolean value: ○ `TRUE` if the file write was successful ○ `FALSE` if the file write failed, for example, if the disk was full

A valid user ID:

- is five characters in length
- has a single upper case alphabetic character followed by four numeric characters, for example “G1234”.

Nadine has decided that global variables and nested modules must not be used.

Nadine wants all inputs to have suitable prompts.

(a) Write **program code** for the module `GetInfo()`.

Visual Basic and Pascal: You should include the declaration statements for variab
Python: You should show a comment statement for each variable used with its data t

Programming language .....

Program code

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [8]

(b) Write **program code** for the module `TopLevel()`.

Visual Basic and Pascal: You should include the declaration statements for variab
Python: You should show a comment statement for each variable used with its data t

Programming language .....

Program code

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [8]

(c) Write **pseudocode** for the module declaration of `WriteInfo()`.

.....

.....

.....

..... [3]

Appendix

Built-in functions (pseudocode)

Each function returns an error if the function call is not properly formed.

`MID(ThisString : STRING, x : INTEGER, y : INTEGER) RETURNS STRING`
returns a string of length `y` starting at position `x` from `ThisString`

Example: `MID("ABCDEFGH", 2, 3)` returns `"BCD"`

`LENGTH(ThisString : STRING) RETURNS INTEGER`
returns the integer value representing the length of `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"`

`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.

Example: `NUM_TO_STRING(87.5)` returns `"87.5"`
Note: This function will also work if `x` is of type `INTEGER`

`STRING_TO_NUM(x : STRING) RETURNS REAL`
returns a numeric representation of a string.

Example: `STRING_TO_NUM("23.45")` returns `23.45`
Note: This function will also work if `x` is of type `CHAR`

Operators (pseudocode)