File Handling

1. A 1D array `Data` of type integer contains 200 elements. Each element has a unique value.

   An algorithm is required to search for the largest value and output it.

   Describe the steps that the algorithm should perform.

   Do **not** include pseudocode statements in your answer.

   .....

   .....

   .....

   .....

   .....

   .....

   .....

   .....

   .....

   .....

   ..... [5]

Answer for 1)

The programs should contain a for loop to loop it over 200 times (so that every bit gets a chance at being sorted). The algorithm should then use binary search to compare each two neighboring numbers to each other and look for the greater one, once found it should switch it to the right hand side of the algorithm. A binary search is better in this case as it's more efficient and only requires a run length equal to the number of data it's to sort. Once this process is repeated over and over again it would eventually arrange the numbers in ascending order with the largest number of them all on the far right. Once we've ensured that the program is correctly sorted, the program should take the (-1) index value of the string and output it.

2. A teacher is designing a program to perform simple syntax checks on programs written by students. Student programs are submitted as text files, which are known as project files.

A project file may contain blank lines.

The teacher has defined the first program module as follows:

Module	Description
`CheckFile()`	• takes the name of an existing project file as a parameter of type string • returns `TRUE` if the file is valid (it contains at least 10 non-blank lines), otherwise returns `FALSE`

(a) Write pseudocode for module `CheckFile()`.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [7]

Answer for a)

```
CheckFile ()
        projName = openRead("exampleProject123.txt")
        DECLARE fileName = "exampleProject123.txt"
        DECLARE isValid = FALSE
        DECLARE nbCount = 0
        FOR x = 0 to fileName.length
                IF fileName(x) is not  Nothing
                        nbCount =+ 1
                elseIF nbCount >= 10 AND fileName.length >= 10
                        isValid = TRUE
                ELSE
                        isValid = FALSE
                        Close projName
                endIF
        endFOR
        OUTPUT isValid
  END
```

Further modules are defined as follows:

Module	Description
`CheckLine()`	• takes a line from a project file as a parameter of type string • returns zero if the line is blank or contains no syntax error, otherwise returns an error number as an integer

`CountErrors()`	• takes two parameters:
	  ○ the name of a project file as a string
	  ○ the maximum number of errors as an integer
	• uses `CheckFile()` to test the project file. Outputs an error message and ends if the project file is not valid
	• calls `CheckLine()` for each line in the project file
	• counts the number of errors
	• outputs the number of errors or a warning message if the maximum number of errors is exceeded

(b) `CountErrors()` is called to check the project file `Jim01Prog.txt` and to stop if more than 20 errors are found.

Write the pseudocode statement for this call.

.....

..... [2]

Answer for b)

CountErrors(fileName,isLineChecked)

(c) Write pseudocode for module `CountErrors()`. Assume `CheckFile()` and `CheckLine()` have been written and can be used in your solution.

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [8]

Answer for c)

```
CountErrors(DECLARE fileName, DECLARE lineErrors)
    While isValid = True
        IF CheckFile(fileName) is FALSE
            isValid = FALSE
        ELSE
        DECLARE errorCount = 0
        DECLARE temp = 0
        Do                              # counting total lines of file
          temp =+ 1
         Until EOF
        FOR x =  1 to temp
            IF CheckLine(x) is Not 0
                errorCount =+1
```

```
        endIF
        IF errorCount > 5          #assuming that the "maximum error count" is fixed at 5
            OUTPUT "The program has exceeded the maximum number of
errors allowed"
endIF
OUTPUT errorCount
endFOR

endIF
endFunction
```

(d) Module `CheckLine()` includes a check for syntax errors.

Two examples of syntax error that **cannot** be detected from examining a **single** line are those involving selection and iteration.

Give **two other** examples.

1 .....

.....

.....

2 .....

.....

.....

[2]

Answer for d)

1. Variables are used but never declared
2. Or forgetting to close the structure such as endFOR could be missing and it wouldn't detect anything