



1. The following pseudocode is an example of a CASE structure.

```
CASE OF MyMark
  75 to 100: MyGrade ← "Distinction"
  35 to 74: MyGrade ← "Pass"
  0 to 34: MyGrade ← "Fail"
  OTHERWISE: OUTPUT "Invalid value
entered" ENDCASE
```

(i) Describe what will happen if the pseudocode is tested when MyMark has the following values:

27 - It would fail as it is between 0 and 34

101- "Invalid value entered" as there is no option above 100

[2]

(ii) Use **pseudocode** to write an IF statement with the same functionality.

```
IF 75 <= MyMark <= 100 THEN
  OUTPUT "Distinction"
ELSE IF 35 <= MyMark <= 74 THEN
  OUTPUT "Pass"
ELSE IF 0 <= MyMark <= 34 THEN
  OUTPUT "Fail"
ELSE
  OUTPUT "Invalid value entered"
END IF
```

.....

.....

[5]

2. A 1D array, ClassName, of type STRING contains 100 elements.

The following pseudocode represents a simple algorithm to process the array.

```
DECLARE SearchValue :
STRING DECLARE FoundFlag :
BOOLEAN DECLARE Index :
INTEGER

INPUT SearchValue
FoundFlag ←
FALSE Index 1

WHILE Index < 101 AND FoundFlag =
  False IF ClassName[Index] =
  SearchValue
  THEN
    OUTPUT Index
```

```

        FoundFlag ←
        TRUE
    ENDIF
    Index ← Index +
1 ENDWHILE

IF FoundFlag =
    FALSE THEN
        OUTPUT "Not found"
ENDIF

```

(a) Describe the purpose of the algorithm.

It gets the input from the user of a certain integer index value which is stored as searchvalue. The program runs through the array called ClassName and checks if the current index of the data is equal to the number entered. If it matches the FoundFlag bool changes to TRUE and program ends, otherwise FoundFlag remains FALSE and outputs "Not found".

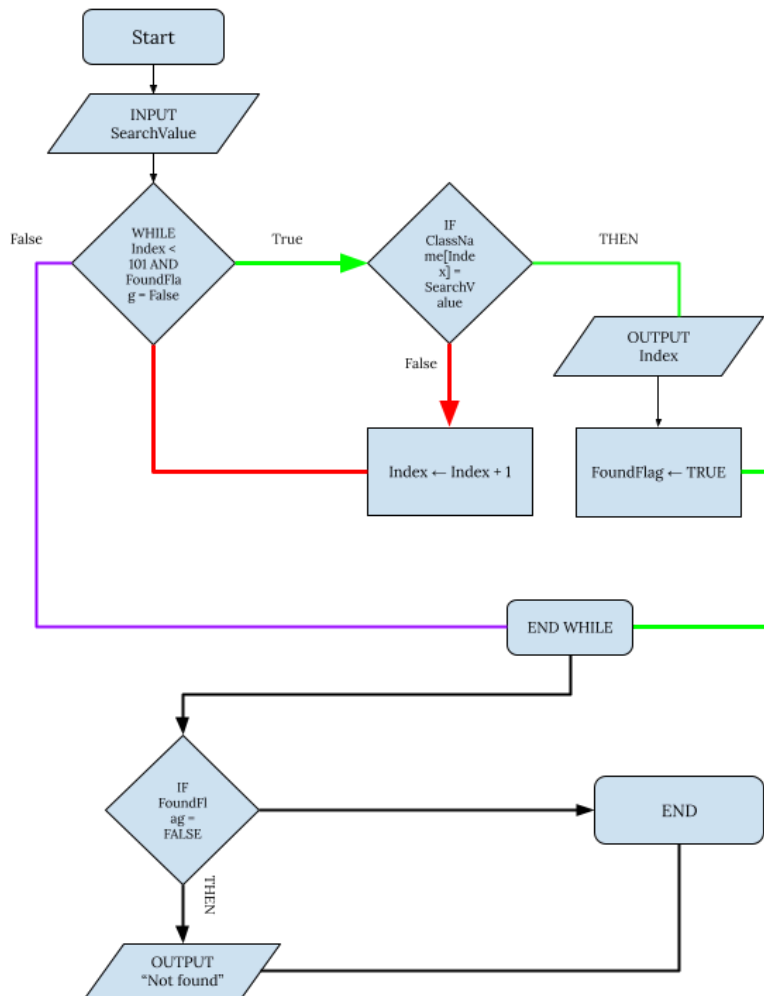
.....

.....

..... [2]

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

Note that variable declarations are not required in program flowcharts.



BLANK PAGE

Appendix

Built-in functions (pseudocode)

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

MODULUS(*x* : INTEGER, *y* : INTEGER) RETURNS INTEGER

returns the remainder when *x* is divided by *y* using integer arithmetic.
Example: MODULUS(5, 2) will return 1

INT(*x* : REAL) RETURNS INTEGER

returns the integer part of *x*.
Example: INT(27.5415) returns 27

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"

TONUM(*ThisString* : STRING) RETURNS INTEGER

returns a numeric value equivalent to *ThisString*.
Example: TONUM("1201") returns integer value 1201

Operators (pseudocode)

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