Question 2

A novice programmer has attempted to create an algorithm to process student scores. The algorithm should read student name and score from a file, determine if the student pased (score >= 50) or failed, and then display a formatted message. If the student passed, their score should be added to a total_pass_score and their count to num_pass_students to calculate an average_pass_score.

The programmer's initial pseudocode is as follows:

```
 1. BEGIN process_grades()
 2.   SET total_pass_score = 0
 3.   SET num_pass_students = 0
 4.   READ student_data FROM "scores.txt"
 5.   FOR each_student IN student_data
 6.     SET student_name = GET_NAME(each_student)
 7.     SET student_score = GET_SCORE(each_student)
 8.     IF student_score >= 50 THEN
 9.       DISPLAY student_name + " scored " + student_score + ": PASS"
10.       CALCULATE total_pass_score = total_pass_score + student_score
11.     ELSE
12.       DISPLAY student_name + " scored " + student_score + ": FAIL"
13.     ENDIF
14.   END FOR
15.   CALCULATE average_pass_score = total_pass_score / num_pass_students
16.   DISPLAY "Average score for passing students: " + average_pass_score
17. END
```

a) Identify the logic errors in the provided pseudocode. For each error, provide the corrected pseudocode line(s) and explain why your correction resolves the logical flaw. (6 marks)

b) Perform a desk check for the corrected algorithm using the following input from "scores.txt" (6 marks):

- "Alice", 75
- "Bob", 40
- "Charlie", 80

c) Based on your understanding of good algorithm practice, recommend one improvement for the corrected algorithm. Justify your recommendation with reference to specific benefits.
(3 marks)

Question 2 Marking Guide

Part	Marks	Criteria	Evidence of achievement
A	1	Error 1 identified	Notes that num_pass_students is never incremented on PASS.
	1	Error 1 corrected with justification	Adds num_pass_students = num_pass_students + 1 inside the PASS branch and explains it prevents an incorrect divisor of 0 and enables average calculation.
	1	Error 2 identified	Notes potential division by zero if there are no passing students.
	1	Error 2 corrected with justification	Wraps average calculation and DISPLAY in an IF num_pass_students > 0 ELSE displays “no passing students”; explains this avoids runtime error and misleading output.
	1	Error 3 identified	Notes inconsistent use of CALCULATE vs SET for assignment or misuse of CALCULATE as a verb.
	1	Error 3 corrected with justification	Replaces CALCULATE … lines with SET … for assignments; explains alignment with pseudocode assignment convention and clearer intent.
B	1	Initialisation shown	Shows total_pass_score = 0, num_pass_students = 0.
	1	After "Alice", 75	Shows PASS, DISPLAY correct, totals total_pass_score = 75, num_pass_students = 1.
	1	After "Bob", 40	Shows FAIL, DISPLAY correct, totals unchanged.
	1	After "Charlie", 80	Shows PASS, DISPLAY correct, totals total_pass_score = 155, num_pass_students = 2.
	1	Average computed	Shows average_pass_score = 155 / 2 = 77.5.
	1	Final DISPLAYs listed	Lists all four DISPLAY lines in order, including the final average line (or the guarded alternative if no passes).
C	1	Specific improvement proposed	Recommends a concrete change, e.g., modularisation: FUNCTION FormatResult(name, score, status) or guard clause for average, or separating I/O from processing.
	1	Justification provided	Explains benefit such as improved maintainability, readability, reuse, testability, or elimination of error paths.
	1	Contextual linkage	Ties the recommendation to this algorithm with a brief example of changed lines or how the module/guard would be invoked and what risk it removes.

Sample Response

Part A

```
BEGIN process_grades()

  SET total_pass_score = 0
  SET num_pass_students = 0

  READ student_data FROM "scores.txt"

  FOR each_student IN student_data
    SET student_name = GET_NAME(each_student)
    SET student_score = GET_SCORE(each_student)

    IF student_score >= 50 THEN
      DISPLAY student_name + " scored " + student_score + ": PASS"
      SET total_pass_score = total_pass_score + student_score
      SET num_pass_students = num_pass_students + 1
    ELSE
      DISPLAY student_name + " scored " + student_score + ": FAIL"
    END IF
  END FOR

  IF num_pass_students > 0 THEN
    SET average_pass_score = total_pass_score / num_pass_students
    DISPLAY "Average score for passing students: " + average_pass_score
  ELSE
    DISPLAY "No passing students. Average not applicable."
  END IF

END
```

Part B

Step	student_name	student_score	total_pass_score	num_pass_students	average_pass_score	DISPLAY
Init	-	-	0	0	-	-
1	Alice	75	75	1	-	Alice scored 75: PASS
2	Bob	40	75	1	-	Bob scored 40: FAIL
3	Charlie	80	155	2	-	Charlie scored 80: PASS
End	-	-	155	2	77.5	Average score for passing students: 77.5

Part C

- Recommendation
  - Modularise formatting and pass-check logic.
  - Add:
    - FUNCTION IsPass(score AS INTEGER) RETURNS BOOLEAN
    - FUNCTION FormatResult(name AS STRING, score AS INTEGER, status AS STRING) RETURNS STRING
- Benefits:
  - Maintainability: clearer separation of concerns, easier to change message format in one place.
  - Reuse and testing: IsPass can be unit-tested independently; FormatResult standardises output.
  - Readability: main loop focuses on flow, not string building.
- Example:

```
FUNCTION IsPass(score AS INTEGER) RETURNS BOOLEAN
  IF score >= 50 THEN
    RETURN TRUE
  ELSE
    RETURN FALSE
  END IF
END FUNCTION

FUNCTION FormatResult(name AS STRING, score AS INTEGER, status AS STRING) RETURNS STRING
  RETURN name + " scored " + score + ": " + status
END FUNCTION

BEGIN process_grades()

  SET total_pass_score = 0
  SET num_pass_students = 0

  READ student_data FROM "scores.txt"

  FOR each_student IN student_data
    SET student_name = GET_NAME(each_student)
    SET student_score = GET_SCORE(each_student)

    IF IsPass(student_score) THEN
      DISPLAY FormatResult(student_name, student_score, "PASS")
      SET total_pass_score = total_pass_score + student_score
      SET num_pass_students = num_pass_students + 1
    ELSE
      DISPLAY FormatResult(student_name, student_score, "FAIL")
    END IF
  END FOR

  IF num_pass_students > 0 THEN
    SET average_pass_score = total_pass_score / num_pass_students
    DISPLAY "Average score for passing students: " + average_pass_score
  ELSE
    DISPLAY "No passing students. Average not applicable."
  END IF

END
```