2.2 Problem Solving and Programming: 30-minute Pack

OCR H446 · Calculators not allowed · about 30 minutes

Total Marks

Name: _____ Date: ____ / ____ / _____

Answer ALL questions. Show all your working.

1 A teacher stores the test scores of 3 students, each having sat 4 tests, in a 2D array scores[1:3, 1:4]:

Test1 Test2 Test3 Test4
Student1: 55 62 70 48
Student2: 80 75 90 85
Student3: 40 55 60 65

(a) Write a pseudocode DECLARE statement for the 2D array scores, capable of holding this data. **(2)**

(b) Write an algorithm to calculate and output the average score for Student2 (row 2) only. **(4)**

© Revision Library | revisionlibrary.co.uk

(c) This nested loop sums every score in the array: **(4)**

```
total <- 0
FOR row <- 1 TO 3
    FOR col <- 1 TO 4
        total <- total + scores[row, col]
    NEXT col
NEXT row
OUTPUT total
```

Complete the trace table below, showing the cumulative value of total after each row is fully processed, and state the final value output.

After row 1: total = ___
After row 2: total = ___
After row 3: total = ___

(d) State the value stored in scores[2, 3]. **(1)**

(Total for Question 1 is 11 marks)

---

2 A stock system uses product codes in the format "AB-1234": exactly 7 characters, with the first two characters being letters, the third character a hyphen, and the last four characters digits.

(a) Write a function ValidateCode(code : STRING) RETURNS BOOLEAN that checks a product code matches this format, using string handling functions such as LEN, MID and ASC. **(6)**

(b) State the value returned by MID("AB-1234", 4, 4). **(1)**

(c) State the value returned by UCASE(LEFT("ab-1234", 2)). **(1)**

(d) Explain why passing the validation in part (a) does not guarantee that a product code entered is accurate. **(2)**

(Total for Question 2 is 10 marks)

© Revision Library | revisionlibrary.co.uk

3 A programmer writes the following subroutines and main program:

```
PROCEDURE DoubleValue(BYREF x : INTEGER)
    x <- x * 2
ENDPROCEDURE

FUNCTION AddFive(y : INTEGER) RETURNS INTEGER
    y <- y + 5
    RETURN y
ENDFUNCTION

num1 <- 10
num2 <- 10
```

DoubleValue(num1)

```
num2 <- AddFive(num2)
OUTPUT num1
OUTPUT num2
```

(a) State the values output for num1 and num2. **(2)**

(b) Explain the difference between passing a parameter by reference and by value, using num1 and num2 as examples. **(4)**

(c) Rewrite DoubleValue as a function, DoubleValueFunc, that takes its parameter by value and returns the doubled value, without modifying the original argument. **(3)**

(d) State one reason a programmer might choose to write a subroutine as a procedure rather than a function. **(1)**

(Total for Question 3 is 10 marks)

© Revision Library | revisionlibrary.co.uk

Mark scheme · 2.2 Problem Solving and Programming: 30-minute Pack

Question 1

(a) correct 2D array syntax with bounds [1:3, 1:4], oe equivalent bounds **B1**

(a) correct data type OF INTEGER **B1**

(a) Answer: DECLARE scores : ARRAY[1:3, 1:4] OF INTEGER

(b) total <- 0 initialised before the loop **M1**

(b) FOR col <- 1 TO 4 loop **M1**

(b) total <- total + scores[2, col] inside the loop, correctly indexing row 2 **M1**

(b) average <- total / 4 and OUTPUT average, correct value 82.5; any correct logic, ignore minor syntax **A1**

(b) Answer:

```
total <- 0
FOR col <- 1 TO 4
    total <- total + scores[2, col]
NEXT col
average <- total / 4
OUTPUT average
```

Output = 82.5

(c) after row 1: total = 235 **B1**

(c) after row 2: total = 565 **B1**

(c) after row 3: total = 785 **B1**

(c) final OUTPUT value = 785, cao **B1**

(c) Answer: 785

(d) 90, cao **B1**

(d) Answer: 90

Question 2

(a) checks LEN(code) = 7 and sets valid to false if not **M1**

(a) checks position 3 equals "-" using MID(code, 3, 1) **M1**

(a) loop/checks over positions 1-2, confirming each is a letter (e.g. using ASC ranges for "A" to "Z") **M1**

(a) loop/checks over positions 4-7, confirming each is a digit (e.g. using ASC ranges for "0" to "9") **M1**

(a) correct use of MID to extract each character being tested **M1**

(a) returns a single BOOLEAN value combining all checks correctly; any correct logic, ignore minor syntax **A1**

(a) Answer:

```
FUNCTION ValidateCode(code : STRING) RETURNS BOOLEAN
    valid <- true
    IF LEN(code) <> 7 THEN
        valid <- false
    ELSE
        IF MID(code, 3, 1) <> "-" THEN
            valid <- false
        ENDIF
        FOR i <- 1 TO 2
            letter <- MID(code, i, 1)
            IF (ASC(letter) < ASC("A")) OR (ASC(lette
                valid <- false
            ENDIF
        NEXT i
        FOR i <- 4 TO 7
            digit <- MID(code, i, 1)
            IF (ASC(digit) < ASC("0")) OR (ASC(digit)
                valid <- false
            ENDIF
        NEXT i
    ENDIF
    RETURN valid
ENDFUNCTION
```

(b) "1234", cao **B1**

(b) Answer: "1234"

(c) "AB", cao **B1**

(c) Answer: "AB"

(d) validation only checks that data conforms to a set of format rules (length, character types, hyphen position), oe **B1**

(d) it does not check the data represents a real/correct product, e.g. "ZZ-9999" could pass validation without being an actual issued code, oe **B1**

(d) Answer: Validation checks the code's format, not whether it corresponds to a real product; a well-formatted but invented code would still pass.

© Revision Library | revisionlibrary.co.uk

Question 3

(a) num1 = 20 **B1**

(a) num2 = 15 **B1**

(a) Answer: num1 = 20, num2 = 15

(b) passing by reference passes the memory address/reference of the actual variable into the subroutine, oe **B1**

(b) so changes made to the parameter (x <- x * 2) directly change the original variable, num1, giving 20, oe **B1**

(b) passing by value copies the value of the argument into a new, separate local variable, oe **B1**

(b) so changes to y inside AddFive do not directly change num2; num2 only changes because the returned value is explicitly assigned back to it, oe **B1**

(b) Answer: By reference links the parameter directly to the caller's variable, so DoubleValue permanently changes num1. By value copies the data, so AddFive cannot change num2 directly; num2 is only updated because its new value is returned and reassigned.

(c) correct FUNCTION header with a by-value parameter (no BYREF) and RETURNS INTEGER **M1**

(c) correct calculation, e.g. RETURN x * 2 or result <- x * 2 **M1**

(c) correct RETURN statement, and the original argument is not modified since the parameter is passed by value; any correct logic, ignore minor syntax **A1**

(c) Answer:

```
FUNCTION DoubleValueFunc(x : INTEGER) RETURNS INTEGER
    RETURN x * 2
ENDFUNCTION
```

(d) a procedure is appropriate when the subroutine performs an action (e.g. output, or modifying data via reference parameters) and does not need to return a single value, oe **B1**

(d) Answer: Procedures are used when no value needs to be returned, e.g. the subroutine only needs to perform an action or modify parameters passed by reference.

© Revision Library | revisionlibrary.co.uk