

8.2 Arrays

Name: _____ Class: _____ Date: _____ **Total: 33 marks**

KEY WORDS two-dimensional (2d) 二维 • one-dimensional (1d) 一维
• a one-dimensional array 一维数组 • a two-dimensional array 二维数组 • arrays 数组

Objective

Build the skills to answer exam questions on **arrays** 数组 — declaring and using **one-dimensional (1D)** 一维 and **two-dimensional (2D)** 二维 arrays, and using a loop to fill or read them.

You must be able to:

- declare a 1D and a 2D array and use an **index** 索引 to read/write a value
- use a loop (and a nested loop) to process an array
- choose between a 1D and a 2D array for a problem

1 Worked examples

■ 1D and 2D arrays

index	1	2	3	4	5
scores	90	75	60	88	50

↑
scores[2] is 75

A 1D array is a list; each value is found by one index

			column		
			1	2	3
row	1				
	2				
	3				

← grid[2,3]

A 2D array is a table; each value is found by two indexes [row, column]

- 1D: DECLARE scores : ARRAY[1:5] OF INTEGER, then scores[2] ← 75.
- 2D: DECLARE grid : ARRAY[1:3, 1:3] OF INTEGER, then grid[2, 3] ← 8.

■ Arrays, files, and the tests that find the bugs

A **one-dimensional array** 一维数组 is a list of values under one name, each reached by an index: DECLARE Scores : ARRAY[1:10] OF INTEGER, so Scores[3] is the third value. A **two-dimensional array** 二维数组 is a table with a row and a column index: DECLARE Grid : ARRAY[1:5, 1:4] OF STRING, so Grid[2,3] is row 2, column 3. Use a nested loop to work through a 2D array: the outer loop over rows, the inner over columns.

Reading and writing a file in Cambridge pseudocode:

```
OPENFILE "Marks.txt" FOR WRITE
WRITEFILE "Marks.txt", Name
CLOSEFILE "Marks.txt"

OPENFILE "Marks.txt" FOR READ
READFILE "Marks.txt", Line
CLOSEFILE "Marks.txt"
```

Always close a file: leaving it open can lose the data that has not yet been written.

Validation 验证 checks that the data is **sensible** as it is entered, and the computer can do it: a **range check** (is the mark between 0 and 100?), a **length check** (is the password at least 8 characters?), a **type check** (is it a whole number?), a **presence check** (was anything entered?), a **format check** (does the date look like DD/MM/YYYY?), and a **check digit** (does the extra digit match a calculation on the others?).

Verification 校验 checks that the data has been **copied correctly**: **double entry** (type it twice and compare) or a **visual check** (read it against the original).

Test data. For a mark out of 100: **normal** data such as 45 is accepted; **abnormal** data such as -5 or A is rejected; **extreme** (boundary) data such as 0 and 100 is

accepted, while 101 is rejected. Write a test plan as a table of the value, the reason for choosing it and the expected result.

Worked example. State one validation and one verification check for a password, with a reason. Validation: a length check that the password is at least eight characters, because a short password is easily guessed. Verification: double entry, typing it twice and comparing, because a typing mistake would lock the user out of their own account.

2 Practice

2.1 An array is declared `DECLARE temps : ARRAY[1:7] OF REAL`. State how many values it can hold. [1]

2.2 State one difference between a 1D and a 2D array. [2]

2.3 Give the pseudocode declaration for an array holding 20 real numbers. [2]

2.4 Identify the type of validation check for each: a date entered as DD/MM/YYYY; an age between 0 and 120; a name box that must not be left empty. [3]

2.5 Explain the difference between validation and verification. [2]

3 Exam-style questions

3.1 Which data structure best stores the marks of 30 students in **one** subject? [1]

- **A** a single variable
- **B** a 1D array
- **C** a 2D array
- **D** a constant

3.2 An array **names** holds 5 names in elements 1 to 5.

(a) Write pseudocode to output every name using a loop. [3]

(b) State which type of array (1D or 2D) you would use to store the marks of 30 students in **5** subjects, and why. [2]

3.3 A program stores the marks of 30 students in a one-dimensional array called **Marks**, out of a maximum of 100.

(a) Write the pseudocode declaration for the array. [2]

(b) Write pseudocode that inputs 30 marks into the array, rejecting any mark that is not between 0 and 100 and asking again. [6]

(c) Complete the test plan by giving one value of each type and the expected result. [3]

Type of test data	Value	Expected result
normal		
abnormal		
extreme		

(d) The marks are then written to a file. Write the three lines of pseudocode that open the file for writing, write `Marks[1]` and close it. [3]

(e) Explain why the file must be closed. [1]

(f) The teacher's name is typed twice when the program starts. Identify this check and explain its purpose. [2]

Solutions

2.1 7.

2.2 a 1D array is a single list using one index; a 2D array is a table using two indexes [row, column].

2.3 `DECLARE Numbers : ARRAY[1:20] OF REAL.`

2.4 Format check; range check; presence check.

2.5 Validation checks that the data entered is sensible or reasonable, and the computer performs it automatically; verification checks that the data has been copied or entered accurately compared with the original.

3.1 B. One subject = a single list = a 1D array.

3.2 (a) `FOR i ← 1 TO 5; OUTPUT names[i]; NEXT i.`

(b) a 2D array; because each student needs 5 marks, so two indexes [student, subject] are required.

3.3 (a) `DECLARE Marks : ARRAY[1:30] OF INTEGER.`

(b) `FOR Count ← 1 TO 30` with `NEXT Count`; a loop that repeats the input while the value is out of range, such as `REPEAT ... UNTIL Mark >= 0 AND Mark <= 100; INPUT Mark` inside that loop; an error message when the value is rejected; `Marks[Count] ← Mark` inside the outer loop.

(c) Normal: 45, accepted. Abnormal: -5 or A, rejected with an error message. Extreme: 0 or 100, accepted; 101, rejected.

(d) `OPENFILE "Marks.txt" FOR WRITE; WRITEFILE "Marks.txt", Marks[1]; CLOSEFILE "Marks.txt".`

(e) Data still held in memory may not have been written to the disc, so closing the file makes sure it is saved.

(f) Double entry, a verification check; it catches a typing mistake, because a slip is unlikely to be repeated identically the second time.