数组

Ref Pseudocode

一维数组

数组 (array) 在一个名字下存放同一类型的多个值。每个值位于一个**下标** (index) 上。

```
DECLARE <identifier> : ARRAY[<lower>:<upper>] OF <data type>
```

```
DECLARE Names : ARRAY[1:3] OF STRING
Names[1] ← "Mei"
Names[2] ← "Sam"
Names[3] ← "Ana"
OUTPUT Names[2]
```

- 边界 (bound) 写作 `lower:upper`, 而且**两端都包含** ——`[1:3]` 有三个元素 (element)。
- 剑桥的数组通常从 **1** 开始, 不是 0。
- 每个元素的类型相同, 由 `OF` 确定。

用 FOR 循环填充与读取

数组和计数循环天生一对: 计数器就是下标。

```
DECLARE Scores : ARRAY[1:5] OF INTEGER
DECLARE Index, Total : INTEGER
FOR Index ← 1 TO 5
    Scores[Index] ← Index * 10
NEXT Index
Total ← 0
FOR Index ← 1 TO 5
    Total ← Total + Scores[Index]
NEXT Index
OUTPUT "Total: ", Total
OUTPUT "Mean: ", Total / 5
```


找出最大值

```
DECLARE Scores : ARRAY[1:5] OF INTEGER
DECLARE Index, Largest : INTEGER
Scores[1] ← 42
Scores[2] ← 17
Scores[3] ← 93
Scores[4] ← 8
Scores[5] ← 55
Largest ← Scores[1]
FOR Index ← 2 TO 5
    IF Scores[Index] > Largest
      THEN
        Largest ← Scores[Index]
    ENDIF
NEXT Index
OUTPUT "Largest: ", Largest
```

把 `Largest` 初始化为**第一个元素**, 绝不要用 0—— 遇到负数数据时,0 会胜出, 答案就错了。

常见错误

- 从声明为 `[1:5]` 的数组中读 `Scores[6]`, 这是越界 (out of bounds)。
- 写成 `Scores(3)`。数组用方括号; 圆括号是调用函数。
- 声明 `[1:5]` 却循环 `0 TO 4`。

二维数组

二维数组 (2-D array) 是一张表: 有行有列, 两个下标。

```
DECLARE <identifier> : ARRAY[<lower1>:<upper1>, <lower2>:<upper2>] OF
↪  <data type>
```

```
DECLARE Grid : ARRAY[1:2, 1:3] OF INTEGER
Grid[1,1] ← 1
Grid[1,2] ← 2
Grid[1,3] ← 3
Grid[2,1] ← 4
Grid[2,2] ← 5
Grid[2,3] ← 6
OUTPUT Grid[2,3]
```

第一个下标是行 (row), **第二个**是列 (column)。

嵌套循环遍历表格

每个维度 (dimension) 一层循环。内层循环走完一整行, 外层才前进一步。

```
DECLARE Grid : ARRAY[1:2, 1:3] OF INTEGER
DECLARE Row, Col : INTEGER
FOR Row ← 1 TO 2
    FOR Col ← 1 TO 3
        Grid[Row, Col] ← Row * Col
    NEXT Col
NEXT Row
FOR Row ← 1 TO 2
    FOR Col ← 1 TO 3
        OUTPUT "Grid[", Row, ",", Col, "] = ", Grid[Row, Col]
    NEXT Col
NEXT Row
```

△ 内层的 NEXT 闭合**内层**计数器。把 NEXT Row 写在 NEXT Col 之前就把两层循环交叉了, 毫无意义 —— 评分者一眼就看得出来。

常见错误

- 把行和列弄反。Grid[2,3] 是第 2 行、第 3 列。
- 写成 Grid[Row][Col]。剑桥写**一对**方括号, 里面用逗号。
- 如上所述把 NEXT 两行交叉。