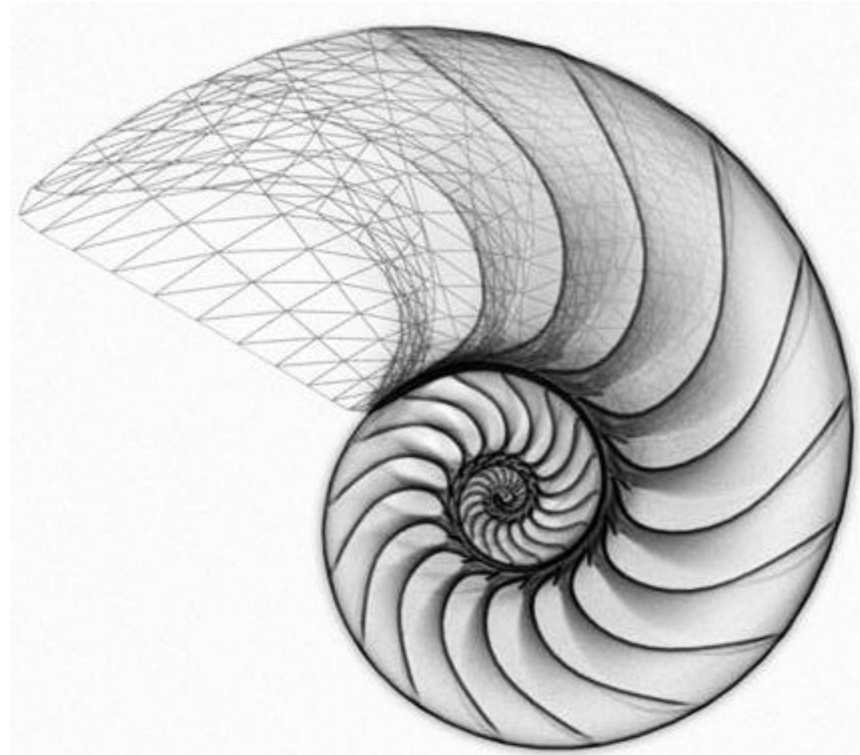


Lecture 11

Multidimensional Array

Two-Dimensional Arrays

A two-dimensional $m \times n$ array A is a collection of $m \cdot n$ data elements such that each element is specified by a pair of integers (such as J, K), called *subscripts*, with the property that

$$1 \leq J \leq m \qquad \text{and} \qquad 1 \leq K \leq n$$

The element of A with first subscript j and second subscript k will be denoted by

$$A_{J,K} \qquad \text{or} \qquad A[J, K]$$

Two-dimensional arrays are called *matrices* in mathematics and *tables* in business applications; hence two-dimensional arrays are sometimes called *matrix arrays*.

Multidimensional Array

Two-Dimensional Arrays

A two-dimensional $m \times n$ array A is a collection of $m \cdot n$ data elements such that each element is specified by a pair of integers (such as J, K), called *subscripts*, with the property that

$$1 \le J \le m \qquad \text{and} \qquad 1 \le K \le n$$

The element of A with first subscript j and second subscript k will be denoted by

$$A_{J,K} \qquad \text{or} \qquad A[J, K]$$

Two-dimensional arrays are called *matrices* in mathematics and *tables* in business applications; hence two-dimensional arrays are sometimes called *matrix arrays*.

Columns

$$\begin{array}{cc} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \text{Rows} \quad \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} A[1,1] & A[1,2] & A[1,3] & A[1,4] \\ A[2,1] & A[2,2] & A[2,3] & A[2,4] \\ A[3,1] & A[3,2] & A[3,3] & A[3,4] \end{bmatrix} \end{array}$$

Fig. 4.9 Two-Dimensional 3 × 4 Array A

Multidimensional Array

Storage Representation

- Row major representation
- Column major representation

Row major representation

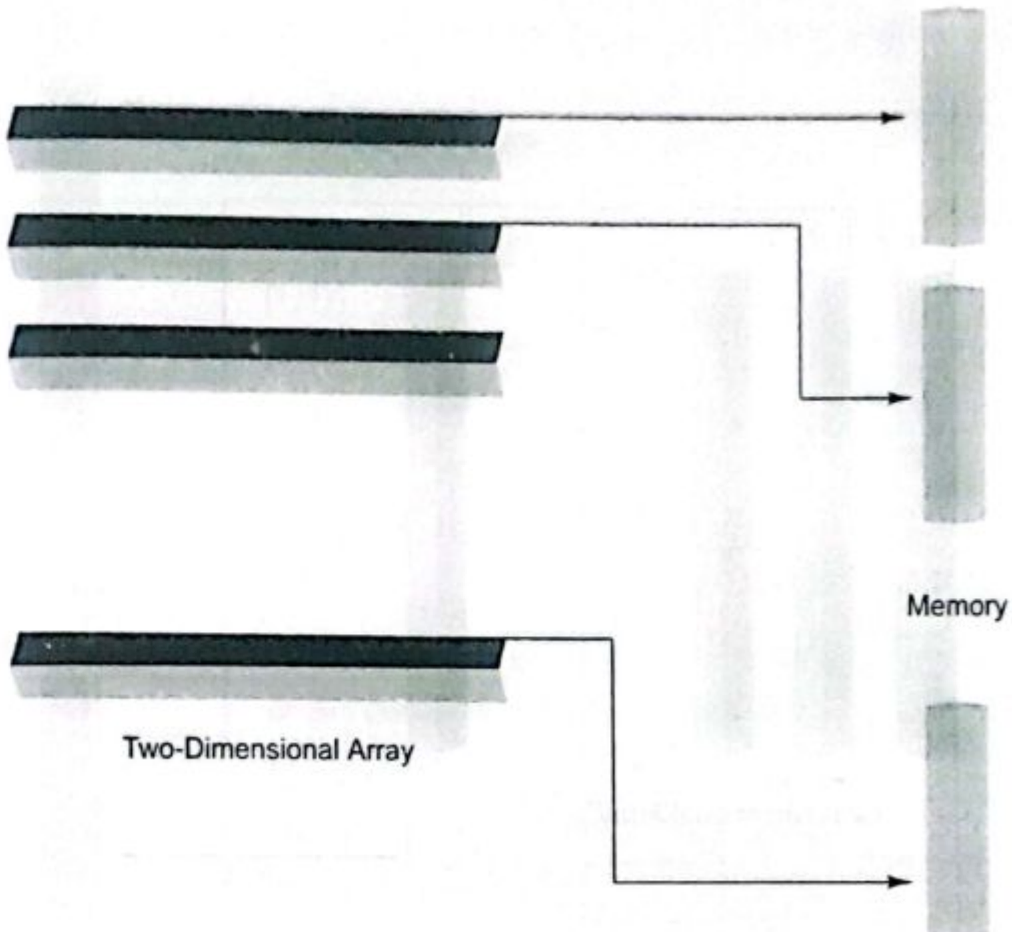


Fig. 4.11 Row Major Representation of a two-dimensional array

Column major representation

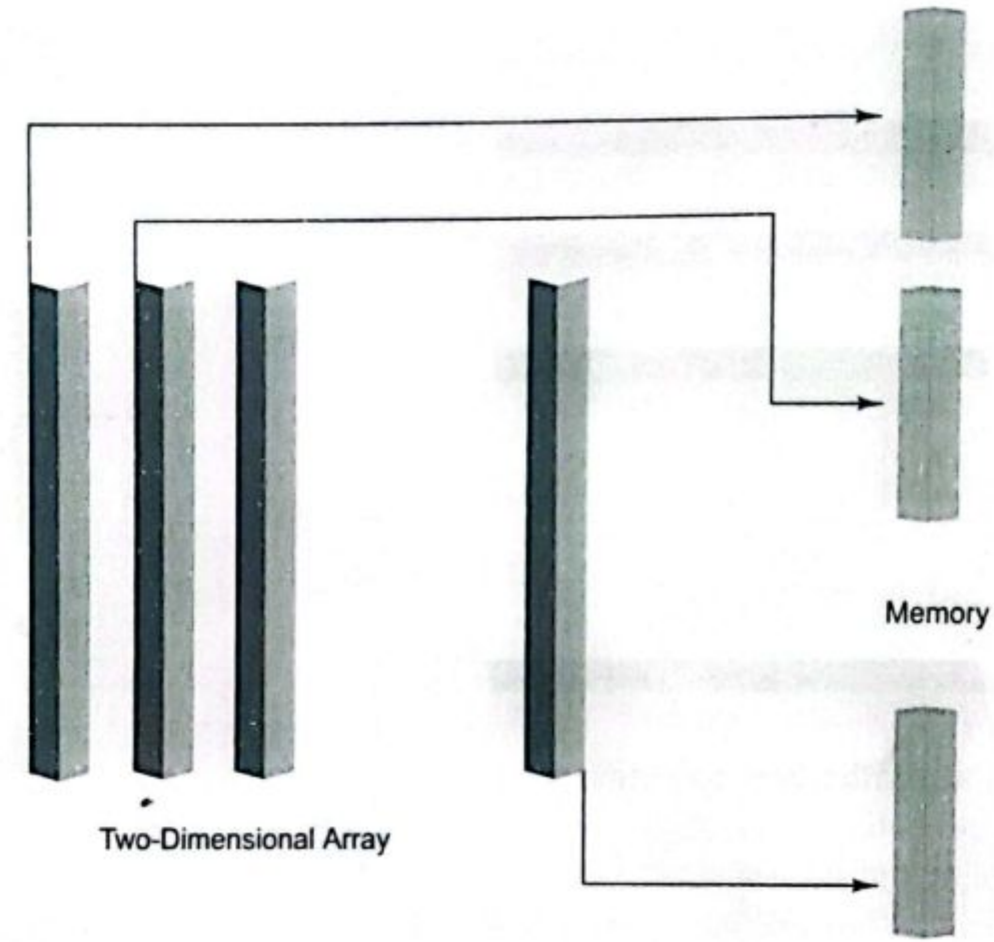
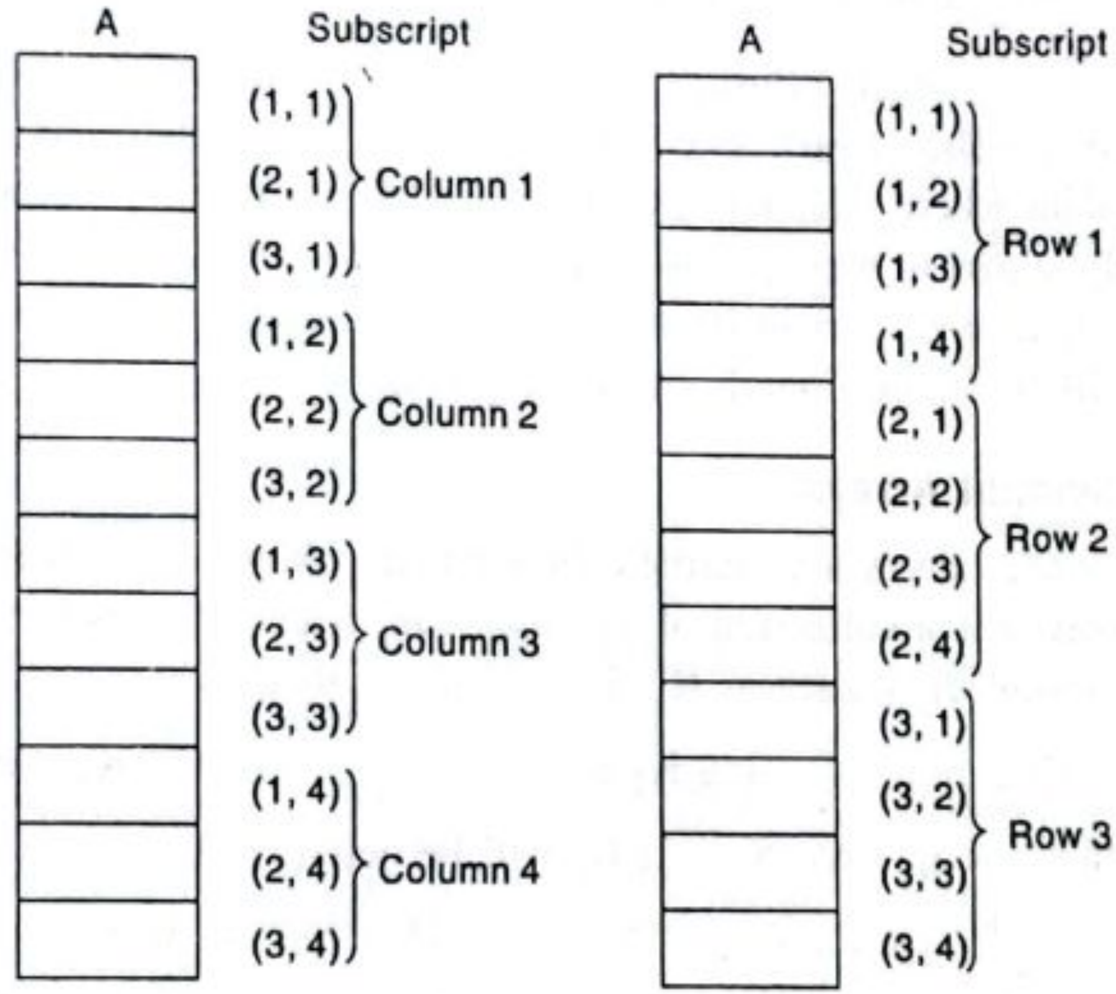


Fig. 4.12 Column-major Representation of a two-dimensional array

Representation of 2D Arrays in Memory



(a) Column-major order (b) Row-major order

Problem I (1D Array)

Consider the linear arrays AAA(5 : 50), BBB(–5 : 10) and CCC(18).

(a) Find the number of elements in each array.

(b) Suppose *Base*(AAA) = 300 and w = 4 words per memory cell for AAA. Find the address of AAA[15], AAA[35] and AAA[55].

Problem I (1D Array)

Consider the linear arrays AAA(5 : 50), BBB(–5 : 10) and CCC(18).

(a) Find the number of elements in each array.

(b) Suppose *Base*(AAA) = 300 and *w* = 4 words per memory cell for AAA. Find the address of AAA[15], AAA[35] and AAA[55].

(a) The number of elements is equal to the length; hence use the formula

$$\text{Length} = \text{UB} - \text{LB} + 1$$

Accordingly,

$$\text{Length(AAA)} = 50 - 5 + 1 = 46$$
$$\text{Length(BBB)} = 10 - (-5) + 1 = 16$$
$$\text{Length(CCC)} = 18 - 1 + 1 = 18$$

Note that Length(CCC) = UB, since LB = 1.

Problem I (1D Array)

Consider the linear arrays AAA(5 : 50), BBB(–5 : 10) and CCC(18).

(a) Find the number of elements in each array.

(b) Suppose *Base*(AAA) = 300 and w = 4 words per memory cell for AAA. Find the address of AAA[15], AAA[35] and AAA[55].

(a) The number of elements is equal to the length; hence use the formula

$$\text{Length} = \text{UB} - \text{LB} + 1$$

Accordingly,

$$\text{Length(AAA)} = 50 - 5 + 1 = 46$$
$$\text{Length(BBB)} = 10 - (-5) + 1 = 16$$
$$\text{Length(CCC)} = 18 - 1 + 1 = 18$$

Note that Length(CCC) = UB, since LB = 1.

(b) Use the formula

$$\text{LOC(AAA[K])} = Base(\text{AAA}) + w(\text{K} - \text{LB})$$

Hence:

$$\text{LOC(AAA[15])} = 300 + 4(15 - 5) = 340$$
$$\text{LOC(AAA[35])} = 300 + 4(35 - 5) = 420$$

AAA[55] is not an element of AAA, since 55 exceeds UB = 50.

Problem I (1D Array) Exercise

Consider the linear arrays AAA(5 : 50), BBB(–5 : 10) and CCC(18).

(a) Find the number of elements in each array.

(b) Suppose *Base*(AAA) = 300 and w = 4 words per memory cell for AAA. Find the address of AAA[15], AAA[35] and AAA[55].

(a) The number of elements is equal to the length; hence use the formula

$$\text{Length} = \text{UB} - \text{LB} + 1$$

Accordingly,

$$\text{Length(AAA)} = 50 - 5 + 1 = 46$$
$$\text{Length(BBB)} = 10 - (-5) + 1 = 16$$
$$\text{Length(CCC)} = 18 - 1 + 1 = 18$$

Note that Length(CCC) = UB, since LB = 1.

(b) Use the formula

$$\text{LOC(AAA[K])} = \textit{Base}(\text{AAA}) + w(\text{K} - \text{LB})$$

Hence:

$$\text{LOC(AAA[15])} = 300 + 4(15 - 5) = 340$$
$$\text{LOC(AAA[35])} = 300 + 4(35 - 5) = 420$$

AAA[55] is not an element of AAA, since 55 exceeds UB = 50.

Problem II (1D Array)

Consider the alphabetized linear array NAME in Fig. 4.30.

	NAME
1	Allen
2	Clark
3	Dickens
4	Edwards
5	Goodman
6	Hobbs
7	Irwin
8	Klein
9	Lewis
10	Morgan
11	Richards
12	Scott
13	Tucker
14	Walton

Fig. 4.30

(a) Fina the number of elements that must be moved if Brown, Johnson and Peters are inserted into NAME at three different times.

Problem I (Multidimensional Array)

Suppose a three-dimensional array MAZE is declared using

$$\text{MAZE}(2{:}8,\ -4{:}1,\ 6{:}10)$$

Then the lengths of the three dimensions of MAZE are, respectively,

$$L_1 = 8 - 2 + 1 = 7, \qquad L_2 = 1 - (-4) + 1 = 6, \qquad L_3 = 10 - 6 + 1 = 5$$

Accordingly, MAZE contains $L_1 \cdot L_2 \cdot L_3 = 7 \cdot 6 \cdot 5 = 210$ elements.

Suppose the programming language stores MAZE in memory in row-major order, and suppose *Base*(MAZE) = 200 and there are $w = 4$ words per memory cell. The address of an element of MAZE—for example, MAZE[5, −1, 8]—is obtained as follows.

Problem I (Multidimensional Array)

Suppose a three-dimensional array MAZE is declared using

MAZE(2:8, −4:1, 6:10)

Then the lengths of the three dimensions of MAZE are, respectively,

$$L_1 = 8 - 2 + 1 = 7, \qquad L_2 = 1 - (-4) + 1 = 6, \qquad L_3 = 10 - 6 + 1 = 5$$

Accordingly, MAZE contains $L_1 \cdot L_2 \cdot L_3 = 7 \cdot 6 \cdot 5 = 210$ elements.

Suppose the programming language stores MAZE in memory in row-major order, and suppose *Base*(MAZE) = 200 and there are $w = 4$ words per memory cell. The address of an element of MAZE—for example, MAZE[5, −1, 8]—is obtained as follows.

$$\boxed{Base(\mathrm{C}) + w[(\ldots((E_1L_2 + E_2)L_3 + E_3)L_4 + \ldots + E_{N-1})L_N + E_N]}$$

Problem I (Multidimensional Array)

Suppose a three-dimensional array MAZE is declared using

$$MAZE(2{:}8,\ -4{:}1,\ 6{:}10)$$

Then the lengths of the three dimensions of MAZE are, respectively,

$$L_1 = 8 - 2 + 1 = 7, \qquad L_2 = 1 - (-4) + 1 = 6, \qquad L_3 = 10 - 6 + 1 = 5$$

Accordingly, MAZE contains $L_1 \cdot L_2 \cdot L_3 = 7 \cdot 6 \cdot 5 = 210$ elements.

Suppose the programming language stores MAZE in memory in row-major order, and suppose *Base*(MAZE) = 200 and there are $w = 4$ words per memory cell. The address of an element of MAZE—for example, MAZE[5, −1, 8]—is obtained as follows. The effective indices of the subscripts are, respectively,

$$E_1 = 5 - 2 = 3, \qquad E_2 = -1 - (-4) = 3, \qquad E_3 = 8 - 6 = 2$$

Using Eq. (4.9) for row-major order, we have:

$$\begin{aligned}
E_1L_2 &= 3 \cdot 6 = 18 \\
E_1L_2 + E_2 &= 18 + 3 = 21 \\
(E_1L_2 + E_2)L_3 &= 21 \cdot 5 = 105 \\
(E_1L_2 + E_3)L_3 + E_3 &= 105 + 2 = 107
\end{aligned}$$

Therefore,

$$LOC(MAZE[5, -1, 8]) = 200 + 4(107) = 200 + 428 = 628$$

it will be E2

Radix Sort

122	64	233	15	18	99