Data Structures

Jasraj Meena
Assistant Professor
Department of Information Technology
Delhi Technological University

Arrays, Records and Pointers

- Data Structure are Classified as either linear or nonlinear.
   - A data structure is said to be linear if its elements form a sequence, or in other words, a linear list.
   - There are two basic ways of representing such linear structures in memory.

- One way is to have the linear relationship between the elements represented by means of sequential memory locations. These linear structures are called Arrays[1].

Arrays, Records and Pointers

- The other way is to have the linear relationship between the elements represented by means of pointers or links.
- These linear structures are called linked lists.

Linear Array

- A linear array is a list of finite number n of homogeneous data elements (i. e. data elements of the same type) such that:
  - The elements of the array are referenced respectively by an index set consisting of n consecutive numbers.
  - The elements of the array are stored respectively in successive memory locations [1].

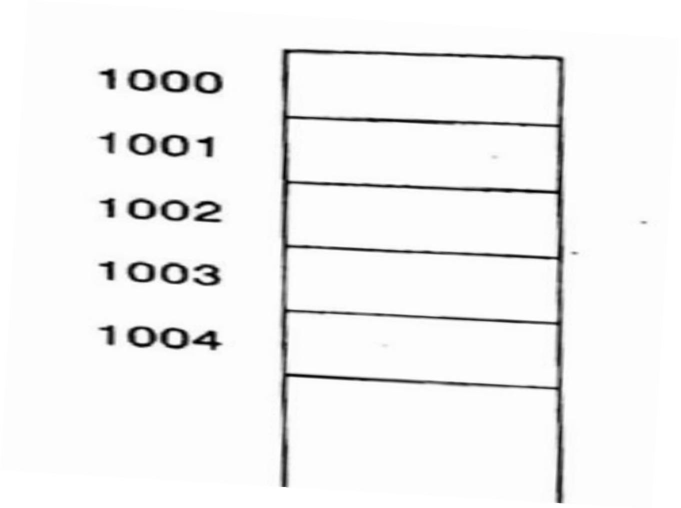
- The number of n elements is called the length or size of the array.

Linear Array

❖ Representation of Linear Array in Memory:

❖ Let LA be a linear array in the memory of computer (A memory of a computer is simply a sequence of addressed locations).

LA(LA[k])= address of the element LA[k] of the array LA.



Linear Array

- **Operations on Linear Arrays:**
  - **Traversing:**
    - Ex. Count number of elements on the array.
  - **Inserting and Deleting:**
  - **Sorting:**
  - **Searching:**

Linear Array

- **Multidimensional Arrays:**
  - The linear array discussed so far are also called one-dimensional arrays, each elements in the array is referenced by a single subscript [1].
  - Most of the programming language allow two-dimensional and three-dimensional arrays, i. e. arrays where elements are referenced, respectively, by two and three subscripts.

Linear Array

❖ Two-dimensional Arrays:

- ❖ A two dimensional M×N array A is a collection of M.N data elements such that each element is specified by pair of integers called subscript with a property that: $1 \leq j \leq N$, $1 \leq k \leq M$ [1].

- ❖ Denoted as A[j, k] or $A_{j,k}$

- ❖ Example: A[3,4]

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

Linear Array

- **Two-dimensional Arrays:**
  - Example: A[3,3]

j/k	0	1	2
0	44	55	88
1	22	12	87
2	123	76	41

2-D Array Rep. Col. Major
44
22
123
55
12
76
88
87
41

2-D Array Rep. Row Major
44
55
88
22
12
87
123
76
41

Linear Array

- **Two-dimensional Arrays:**
  - Example: A[3,3], now delete A[2,2]

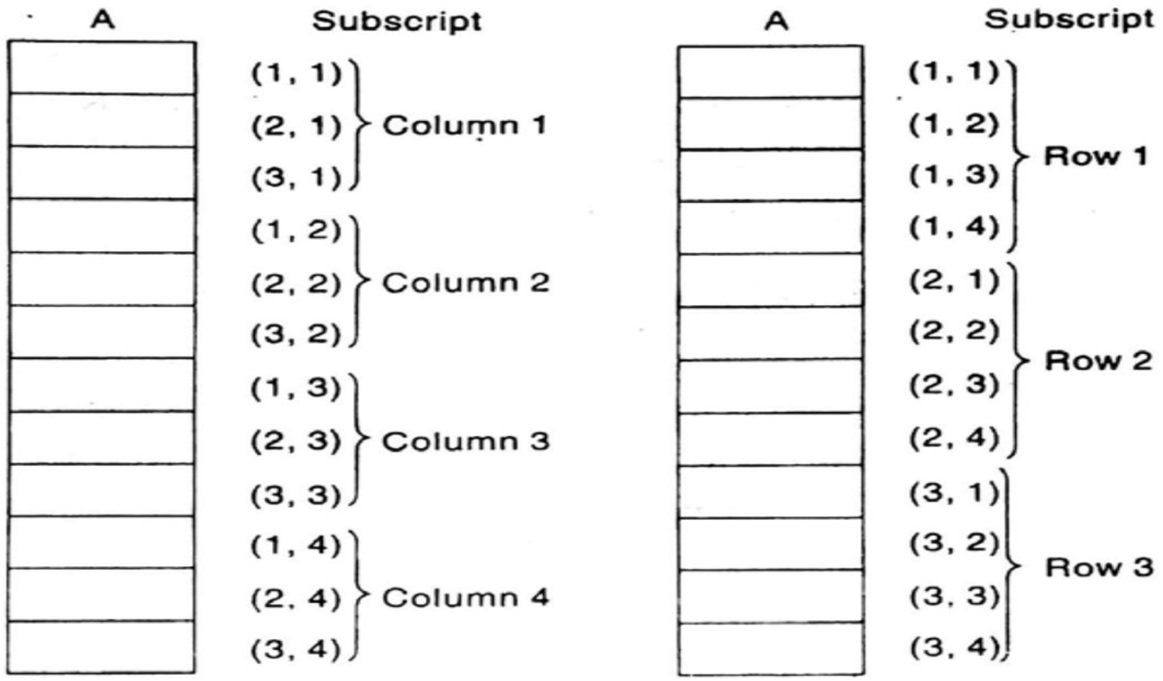
j/k	0	1	2
0	44	55	88
1	22	12	87
2	123	76	41

2-D Array Rep. Col. Major
44
22
123
55
76
88
87
41

2-D Array Rep. Row Major
44
55
88
22
87
123
76
41

Linear Array

- **Representation of two-dimensional Arrays in memory [1]:**
  - **Column major order:**
  - **Row Major order:**



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

Linear Array

- **Representation of two-dimensional Arrays in memory [1]:**
  - **Direct formula to find the memory address of an element:**

$$LOC(LA[K]) = Base(LA) + w(K - 1)$$

(Column-major order) $$LOC(A[J, K]) = Base(A) + w[M(K - 1) + (J - 1)]$$

(Row-major order) $$LOC(A[J, K]) = Base(A) + w[N(J - 1) + (K - 1)]$$

- Here, w is the number of words per memory cell for the LA .

Linear Array

❖ General Multidimensional Arrays [1]:

- An n dimensional $m_1 \times m_2 \times m_3 \ldots \times m_n$ array B is a collection of $m_1 . m_2 . m_3 . \ldots m_n$ data elements in which each element is specified by a list of n integers such as $k_1, k_2, k_3, \ldots k_n$ called subscripts. With the property that
$1 \leq k_1 \leq m_1$, $1 \leq k_2 \leq m_2$ .... $1 \leq k_n \leq m_n$

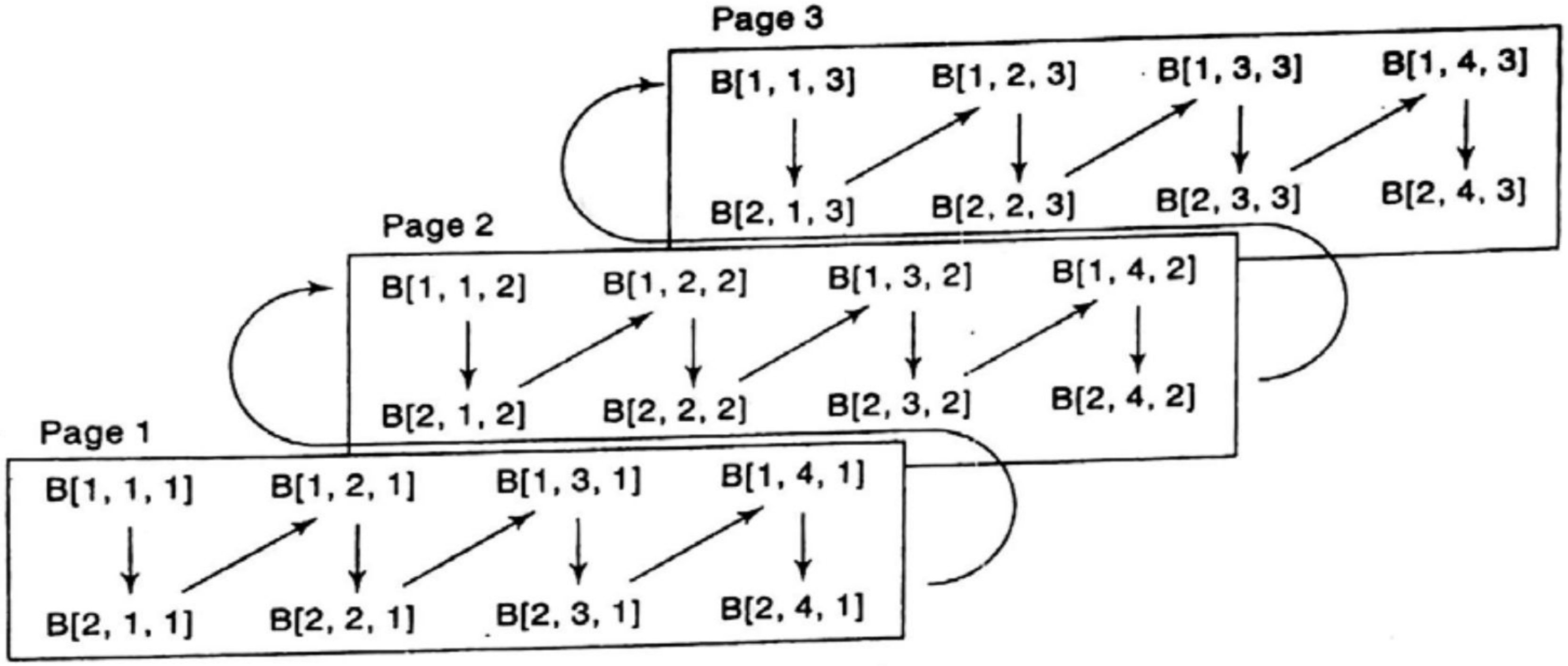
- These elements of B are denoted as: $B[k_1, k_2, k_3, \ldots k_n]$.

- Ex.: **B[2, 4, 3].**

Linear Array

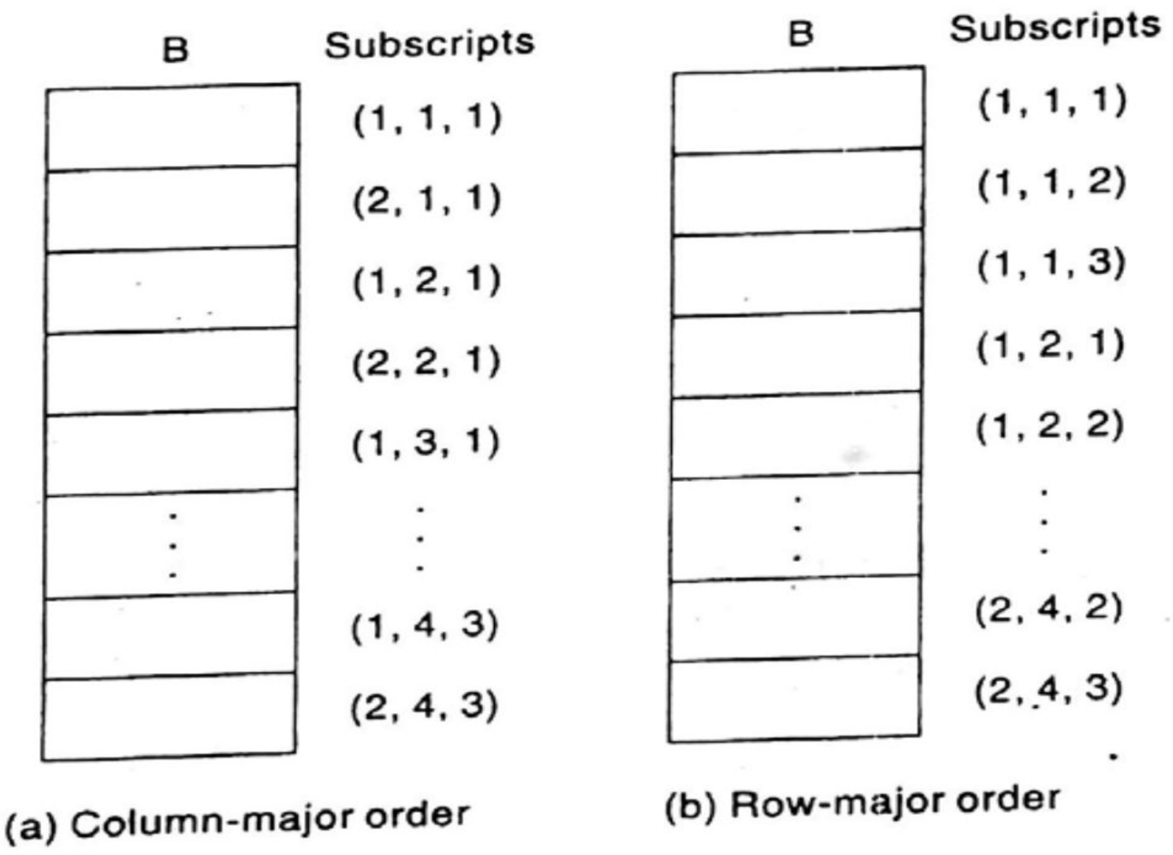
❖ General Multidimensional Arrays [1]:

- Ex.: **B[2, 4, 3].**



Linear Array [1]

- Ex.: **B[2, 4, 3].**



Linear Array [1]

➢ Formula for finding the memory address of a General Multi-dimensional Array [1]:

❖ Suppose C is the multidimensional array:

❖ The length L_i of dimension i of C is the number of elements in the index set, and L_i can be calculated as:

$$L_i = \text{upper bound} - \text{lower bound} + 1$$

Linear Array [1]

- Formula for finding the memory address of a General Multi-dimensional Array:
  - For a given subscript k_i, the effective index E_i of L_i is the number of indices preceding k_i in the index set, and E_i can be calculated from:

$$E_i = K_i - \text{lower bound}$$

Linear Array [1]

- Formula for finding the memory address of a General Multi-dimensional Array:

$$Base(C) + w[(((\ldots (E_N L_{N-1} + E_{N-1})L_{N-2}) + \ldots + E_3)L_2 + E_2)L_1 + E_1]$$

OR

$$Base(C) + w[(\ldots((E_1 L_2 + E_2)L_3 + E_3)L_4 + \ldots + E_{N-1})L_N + E_N]$$

Linear Array [1]

- Suppose a three-dimensional array MAZE is declared using

  MAZE(2:8, -4:1, 6:10)

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

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

- Accordingly, MAZE contains $L_1 . L_2 . L_3$=7.6.5= 210 elements.

- Suppose the programming language stores MAZE in the memory in row-major order, and suppose Base(MAZE)=200 and there are w=4 words per memory cell.

Linear Array [1]

- Suppose the programming language stores MAZE in the memory in row-major order, and suppose Base(MAZE)=200 and there are w=4 words per memory cell.

- The address of element of the MAZE for example, MAZE[5, -1, 8] is obtained as follows:

- The address indices of the subscripts are, respectively,

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

Linear Array [1]

❖ The address indices of the subscripts are, respectively,

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

❖ Using row major order, we have:

$$E_1 L_2 = 3 . 6 = 18$$
$$E_1 L_2 + E_2 = 18 + 3 = 21$$
$$(E_1 L_2 + E_2)\, L_3 = 21 . 5 = 105$$
$$(E_1 L_2 + E_2)\, L_3 + E_3 = 105 + 2 = 107$$

Therefore, LOC(MAZE(5, -1, 8))= 200+4(107)=200+428= 628.

$$Base(C) + w[(\ldots((E_1 L_2 + E_2) L_3 + E_3) L_4 + \ldots + E_{N-1}) L_N + E_N]$$

References

1. Seymour Lipschutz, “Data Structures”, Schaum’s Series McGraw Hill edition 2013.