UNIT – II
LINEAR DATA STRUCTURES USING SEQUENTIAL ORGANIZATION, SEARCHING AND SORTING

Concept of sequential organization, arrays as ADT

In data structures, sequential organization refers to the arrangement of data elements in a linear sequence or order. Unlike hierarchical or non-sequential structures, where elements may be interconnected in a complex network, sequential organization maintains a straightforward linear progression from one element to the next.

Array:
An array is defined as finite ordered collection of homogeneous data elements.

Some pointe about an Array: -

(i) size-

No. of elements in an array is called as size or length of array.

Length or size = Upper bound-Lower bound+1

(ii) Type-

It indicates the type of data present in an array

e.g- int a[50]

in the above example the types of elements that are present in the array are of integer type.

(iii) Base:-

It indicates the starting address of the memory where first dement of array is stored.

(iv) Index:-.

The subscript by which an array can be referred is called index.

Ex: a[5];   //5 is Index.

There are following types of arrays
(1) 1- dimensional array
(2) 2-dimensional array
(3) Multidimensional array

1 dimensional Arrays-
If only one subscript or index is required to refer all the elements in an array then it is called as one dimensional array.

e.g:-int a [50]

In the above example statement 'a' indicate name of an array, int indicates the datatype and 50 represents the max no. of elements that will be present in an array.

Storage representation of array, Matrix operations using arrays

Memory Allocation of 1-D array
Array elements are stored in a continuous manner. So the elements of arrays are present in continuous memory Location such as a[0],a[1],a[2]--- up to the maximum size of declared array.

To find out the address of a particular that present in an array we have to use the following formula: -

Loc (LA [K]) = Base (LA) + w (K-Lower bound)

where **K** indicates the element which address is to be calculated.

w indicates the word size and lower bound Indicate the lower bound value of the given array.

Example: Consider an auto company which provide information about automobile sold from year 1932 to 1984. The base address of auto record = 200 and w = 4 words per memory cell. find out the address of year (1965).

Loc (LA [K]) = Base (LA) + w (K-Lower bound)
Here k = 1965
Base (LA) = 200
w= 4
k = 1965
LB = 1932
LOC (LA[k) = 200+4[1965-1932]
LOC(LA [1965]) = 332

2-Dimensional array

A two dimensional array will be represented by two subscript a[i][j] where i indicates no of rows and j Indicates no. of columns

Representation A 2D array in memory:-

We can represent 2D array In memory by using:

(i) Row Major representation
(ii) Column Major representation

Example: - Let A be a 3X4 array is given. We have to represent it in memory by using row major order

The 3x4 array can be represented in a matrix form is defined as: -

A=

	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Row 2	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Row 3	x[2][0]	x[2][1]	x[2][2]	x[2][3]

Row Major representation

	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0] ⇨	x[0][1] ⇨	x[0][2] ⇨	x[0][3]
Row 2	x[1][0] ⇨	x[1][1] ⇨	x[1][2] ⇨	x[1][3]
Row 3	x[2][0] ⇨	x[2][1] ⇨	x[2][2] ⇨	x[2][3]

Task: Column Major representation

Formula for Row Major

Address of A[I][J] = B + W * ((I – LR) * N + (J – LC))

I = Row Subset of an element whose address to be found,
J = Column Subset of an element whose address to be found,
B = Base address,
W = Storage size of one element store in an array(in byte),
LR = Lower Limit of row/start row index of the matrix(If not given assume it as zero),
LC = Lower Limit of column/start column index of the matrix(If not given assume it as zero),
N = Number of column given in the matrix.

Example: Given an array, arr[1………10][1………15] with base value 100 and the size of each element is 1 Byte in memory. Find the address of arr[8][6] with the help of row-major order.

Solution:

Given:
Base address B = 100
Storage size of one element store in any array W = 1 Bytes
Row Subset of an element whose address to be found I = 8
Column Subset of an element whose address to be found J = 6
Lower Limit of row/start row index of matrix LR = 1
Lower Limit of column/start column index of matrix = 1
Number of column given in the matrix N = Upper Bound – Lower Bound + 1
= 15 – 1 + 1
= 15

Formula:
Address of A[I][J] = B + W * ((I – LR) * N + (J – LC))

Solution:
Address of A[8][6] = 100 + 1 * ((8 – 1) * 15 + (6 – 1))
= 100 + 1 * ((7) * 15 + (5))
= 100 + 1 * (110)
Address of A[I][J] = 210

Formula for Column Major

Address of A[I][J] = B + W * ((J – LC) * M + (I – LR))

I = Row Subset of an element whose address to be found,
J = Column Subset of an element whose address to be found,
B = Base address,
W = Storage size of one element store in any array(in byte),
LR = Lower Limit of row/start row index of matrix(If not given assume it as zero),
LC = Lower Limit of column/start column index of matrix(If not given assume it as zero),
M = Number of rows given in the matrix.

Example: Given an array arr[1………10][1………15] with a base value of 100 and the size of each element is 1 Byte in memory find the address of arr[8][6] with the help of column-major order.

Solution:

Given:
Base address B = 100
Storage size of one element store in any array W = 1 Bytes
Row Subset of an element whose address to be found I = 8
Column Subset of an element whose address to be found J = 6
Lower Limit of row/start row index of matrix LR = 1
Lower Limit of column/start column index of matrix = 1
Number of Rows given in the matrix M = Upper Bound – Lower Bound + 1
= 10 – 1 + 1
= 10

Formula: used
Address of A[I][J] = B + W * ((J – LC) * M + (I – LR))
Address of A[8][6] = 100 + 1 * ((6 – 1) * 10 + (8 – 1))
= 100 + 1 * ((5) * 10 + (7))
= 100 + 1 * (57)
Address of A[I][J] = 157