## ADDRESS CALCULATION

Linear data structure is the logical structure created in memory to contain data linearly or sequentially. Such structures are arrays, stacks and queues. The number of bytes occupied in the memory by an element of array depends upon the type of data it contains. Hence, to manage the memory efficiently and accessing the data properly from the array, it is must to know the physical address of an element that is stored in the array. This is why, address calculation is essential in an array.

### Address Calculation in Single Dimensional Array

Let the array be Arr. The address of an element Arr[I] can be calculated as:

Address = B + W (I – L)

Where,

B = Base address

W = Word width (Number of bytes an element contains in the memory).

I = Index or cell number of the element whose address is to be calculated.

L = Lower boundary i.e. the lowest index of the array.

Number of bytes contained by an element in the memory:

char type : 2 bytes
int type : 4 bytes
float type : 4 bytes
long type : 8 bytes
double : 8 bytes

Lowest cell number or array index is taken to be 0 by default unless mentioned otherwise. If the cell rage is mentioned as Arr[-4.......10], the lowest subscript is to be taken as -4 for address calculation.

**Address Calculation in Double Dimension Array**

In double dimensional array, the orientation of the elements can be row-wise or column-wise. The address of an element can be different as per orientation of the elements in double dimensional array. Hence, there are separate systems for calculating addresses in row major form and column major form orientation of the elements in the matrix.

***Row Major Orientation***

Address calculation of an element Arr[I][J] can be:

Address of element Arr[I][J] = $B+W\ [N\ (I-R_0)+(J-C_0)]$

$B$ = Base address
$W$ = Word width
$I$ = Row subscript of the element
$J$ = Column subscript of the element
$R_0$ = Lower range of the row (Lowest row number)
$C_0$ = Lower range of the column (Lowest column number)
$N$ = Number of columns

***Column Major Orientation***

Address calculation of an element Arr[I][J] can be:

Address of element Arr[I][J] = $B+W\ [(I-R_0)+M\ (J-C_0)]$

$B$ = Base address
$W$ = Word width
$I$ = Row subscript of the element
$J$ = Column subscript of the element
$R_0$ = Lower range of the row (Lowest row number)
$C_0$ = Lower range of the column (Lowest column number)
$M$ = Number of rows

System uses number of rows and number of columns by default as 0 unless mentioned. Sometimes the dimension of a matrix may be specified in the form Arr[LR.......UR , LC......... UC]. In such situation, the number of rows and the number of columns can be calculated as follows:

Number of rows (N): $(UR-LR)+1$

Number of columns (M): $(UC-LC)+1$

Where,

LR = Lowest row number
UR = Highest row number
LC = Lowest column number
UC = Highest column number

**Example 1: A single dimensional array Arr contains 10 integer numbers. If the base address of the array is 1000, find the address of sixth and eighth elements of the array.**

**Solution:**

$B = 1000$
$I = 7$ (Subscript of the 6th element is 7)
$W = 4$ (Word width of integer number is 4)
$L = 0$ (Lowest cell number is 0 by default)

Address of 6th element $= B+W\ (I-L)$
$= 1000+4(6-0)$
$= 1000+24$
$= 1024$

Subscript of the eighth element (I) = 9

Address of the eighth element = B + W (I - L)

= 1000 + 4 (9 - 0)

= 1000 + 36

= 1036

**Example: 2: A double dimensional array Arr[4][10] contains characters. If Arr[0][0] is stored at the memory location 200, find the address of Arr[3][7]. If matrix is:**

- **Row major form**
- **Column major form**

**Solution:**

B = 200

W = 2

I = 3

J = 7

$R_0$ = 0

$C_0$ = 0

N = 10 colm

M = 4 row

*Row-wise calculation:*

Address of array element Arr[3][7] = B + W [N (I - R0) + (J - C0)]

= 200 + 2[10(3 - 0) + (7 - 0)]

= 200 + 2(30 + 7)

= 200 + 2 x 37

= 274

*Column-wise calculation:*

Address of array Arr[3][7] = B + W [(I - R0) + M (J - C0)]

= 200 + 2 [(3 - 0) + 4 (7 - 0)]

= 200 + 2 x (3 + 28)

= 200 + 2 x 31

= 200 + 62

= 262

**Example 3: A double dimensional array Arr[-10..12, -15..20] contains elements in such a way that each element occupies 4 byte memory space. If the base address of Array is 2000, find the address of the array element Arr[-6][8]. Perform the calculations assuming that the elements are:**

- **Row major oriented**
- **Column major oriented**

**Solution:**

B = 2000

I = -6

J = 8

$R_0$ = -10

$C_0$ = -15

Number of rows (M) = [12 - (-10)] + 1 = 23

Number of columns (N) = [20 - (-15)] + 1 = 36

W = 4

*Row-wise calculation:*

Address of array element Arr[-6][8] = B + W [N (I - R0) + (J - C0)]

= 2000 + 4 [36(-6 - (-10)) + (8 - (-15))]

= 2000 + 4 [36(-6 + 10) + (8 + 15)]

$= 2000 + 4\ [(36 \times 4) + 23]$
$= 2000 + 4 \times (144 + 23)$
$= 2000 + 4 \times 167$
$= 2000 + 668$
$= 2668$

*Column-wise calculation:*

Address of array Arr[-6][8] = B + W [(I - R0) + M (J - C0)]

$= 2000 + 4\ [(-6 - (-10)) + 23\ (8 - (-15))]$
$= 2000 + 4\ [(-6 + 10) + 23\ (8 + 15)]$
$= 2000 + 4\ [4 + 23 \times 23]$
$= 2000 + 4 \times (4 + 529)$
$= 2000 + 4 \times 533$
$= 2000 + 2132$
$= 4132$

**Example: 4: A double dimensional array Arr[4][4] contains floating type elements. If Arr[2][2] is stored at memory location 2000, find the base address of the array. Assume that the array is row major oriented.**

**Solution:**

B = ?
I = 2
J = 2
$R_0 = 0$
$C_0 = 0$

Number of columns (N) = 4

Address of the element Arr[I][J] = B + W [N (I - R0) + (J - C0)]

Address of element Arr[2][2] = B + 4 [4 (2-0) + (2-0)]

or, 2000 = B + 4 [4 x 2 + 2]
or, 2000 = B + 4 x 10
or, 2000 = B + 40

Hence, Base address (B) = 2000 - 40 = 1960

**Example 5: A double dimensional array Arr[4][4] contains integer numbers. Find the address of Arr[3][2], if the array is column major oriented. The base address of the array is 1000.**

**Solution:**

B = 1000
W = 4
I = 3
J = 2

Number of rows (M) = 4

Address of array element Arr[3][2] = B + W [(I - R0) + M (J - C0)]

$= 1000 + 4[(3-0) + 4(2-0)]$
$= 1000 + 4 \times (3 + 8)$
$= 1000 + 4 \times 11$
$= 1000 + 44$
$= 1044$