

Using base address we can access all the elements of the array

$arr[6] \rightarrow 1000 + (6 - 0) \times 4$
 $1000 + 24 = 1024$

(contiguous storage of an array elements) (Index)

(By default -0)

Lower bound is the index of base element which by default is 0



Memory address

$$arr(i) = \text{Base address} + (i - \text{Lower Bound of an index}) \times$$

Random access

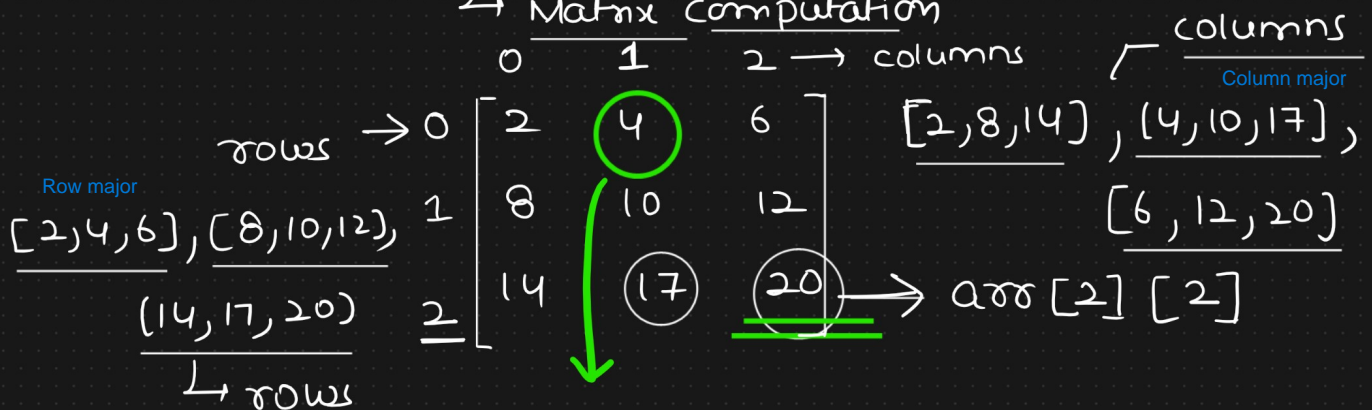
Primary feature of an array

size of an element

2D Array → Rows, columns

$\begin{cases} i \rightarrow \text{Row index} \\ j \rightarrow \text{Column index} \end{cases}$

→ Matrix computation



arr[0][1]

→ Row major form

0	1	2	3	4	5	6	7	8
2	4	6	8	10	12	14	16	18
2	8	14	4	10	17	6	12	20

(Col major form) → CMF

1) Row major form \rightarrow Row-wise

2) column major form \rightarrow column-wise

In terms of memory allocation even 2D arrays are stored in 1D using either row major approach or column major approach.

At visual level there can be multiple dimensions but at memory level we are storing element always in 1D where multiple dimensions are converted into 1D either using row major or column major approach. By default it is following row major approach.