CS & IT ENGINEERING

Data Structure

Arrays

Chapter - 2

Lec- 01

By- Pankaj Sharma sir

Arrays-I

1 to 1000 (413)

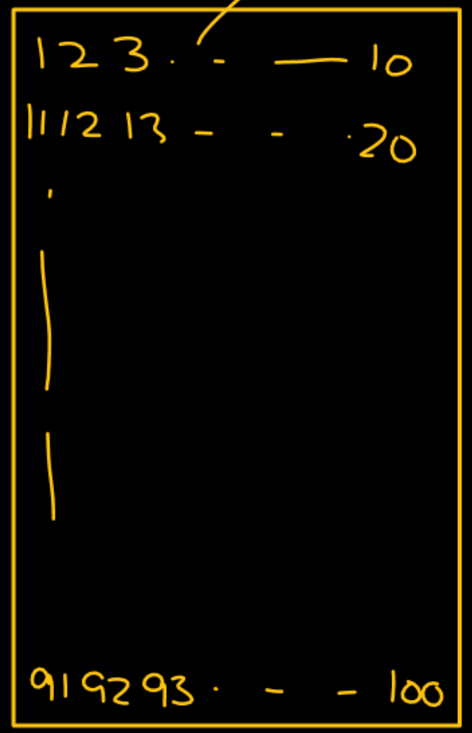
Row wise arrang. with grouping of 10

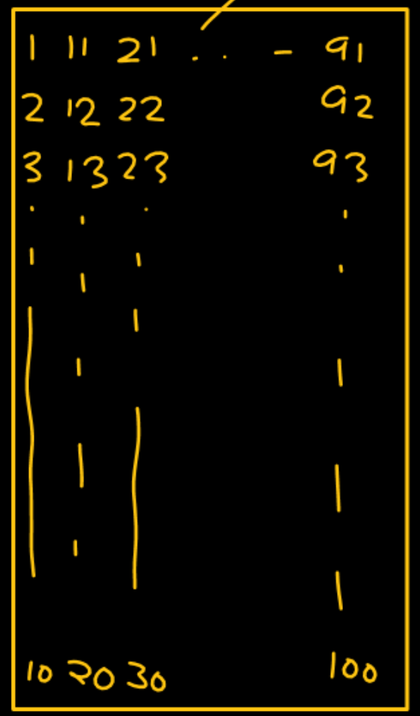
Data structure

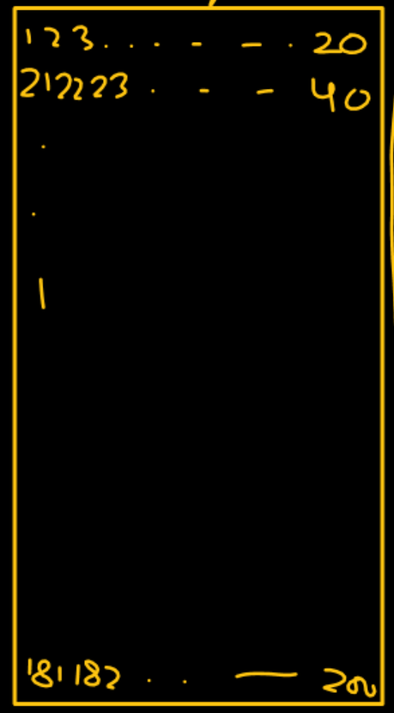
col. wise with grp of 10

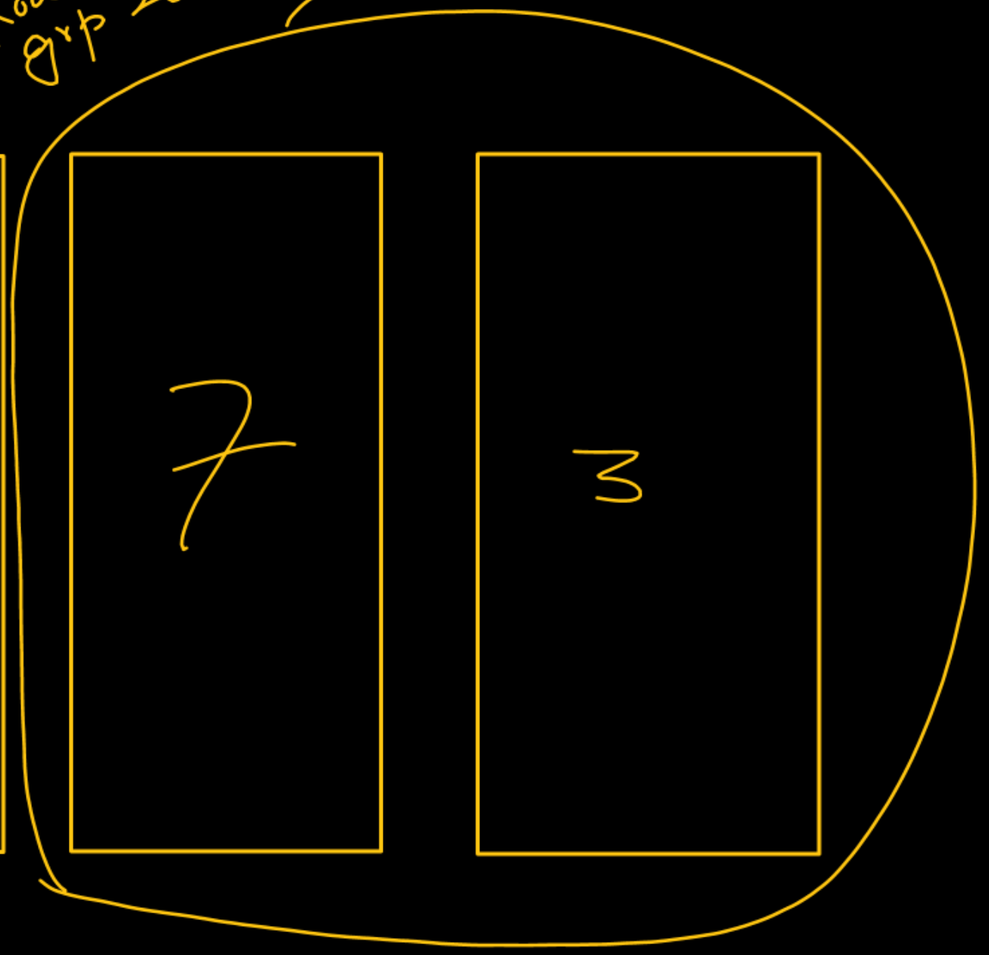
Row wise grp 20

No arrangement









5th page 2nd row 3rd col

5th Page 2nd row 3rd Col

Phone → contact list → ~~Sorted~~ X

~~search~~

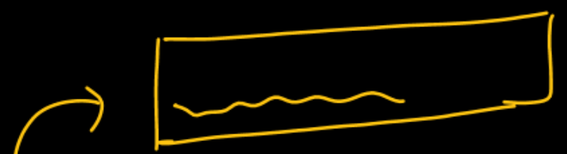
dictionary : → ~~Sorted~~

↘ ~~Search~~ ↗

Unsorted

Search ⇒

Gmail :



dictionary : → Sorted

→ unsorted

operations

① addition of word

Data structure

Linear Data structure

* at most 2 neighbour

1. Arrays
2. Linked list
3. stack and Queues

Non-Linear data structure

* Can have more than 2 neigh

4. Tree
5. Graph

Hashing

Arrays

* Similar type of elements. (size of each element is same)
* One after another in memory.
* relative addressing
* base address
* index starts from 0.

int A[10]; index → 0 to 9

int → 4 byte

base address → 1000

Add of A[0] = 1000

Add of A[2] = ?

(1) How many elements are already filled before A[2] ? = 2

(2) size of each elem = 4 byte

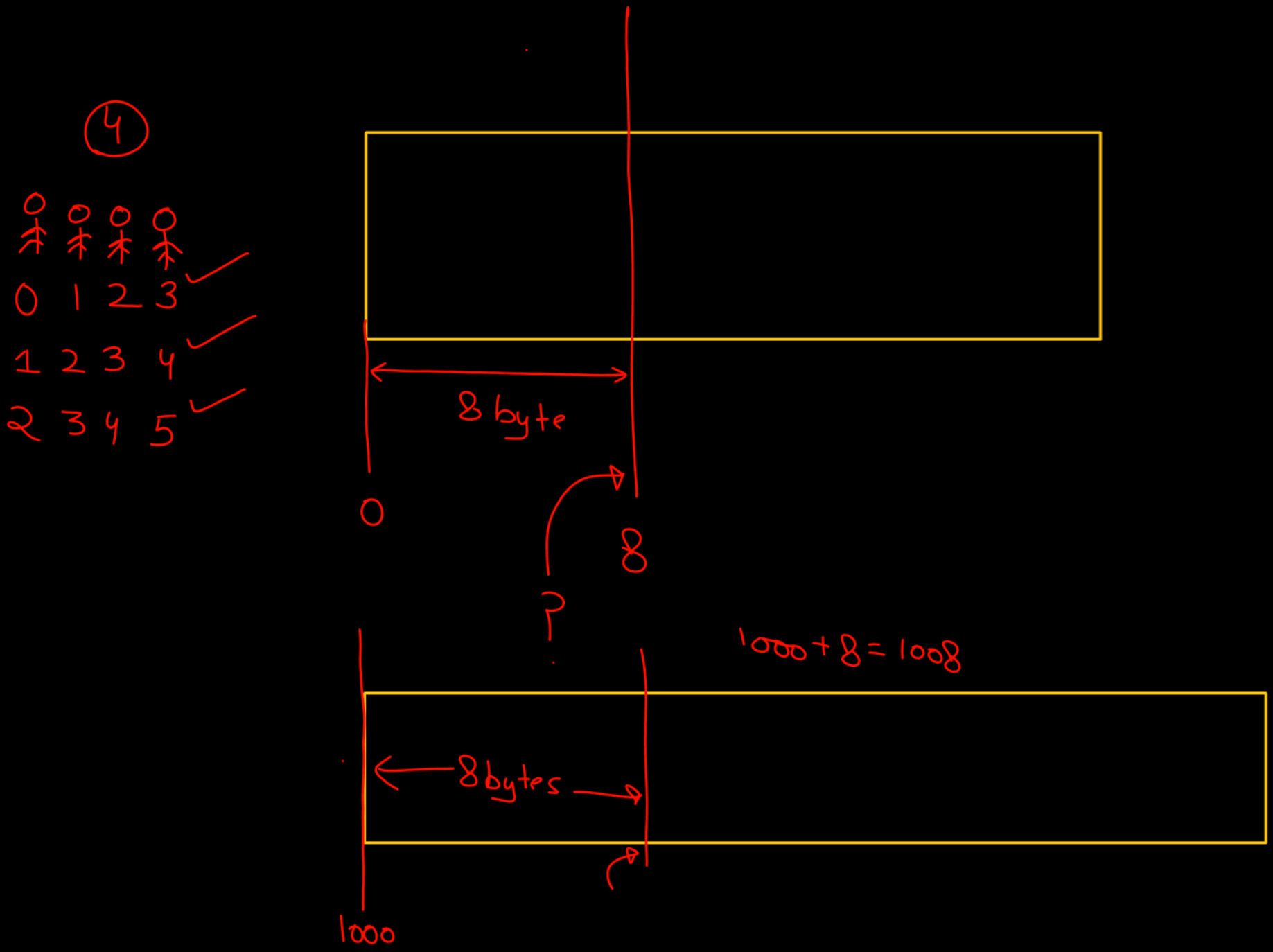
(3) How much memory already filled by these 2 elements = 2 × 4 = 8 byte

④

0 1 2 3

1 2 3 4

2 3 4 5



index - 0

int A[10]

base Add → 1000

size = 4 byte

add (A[6])

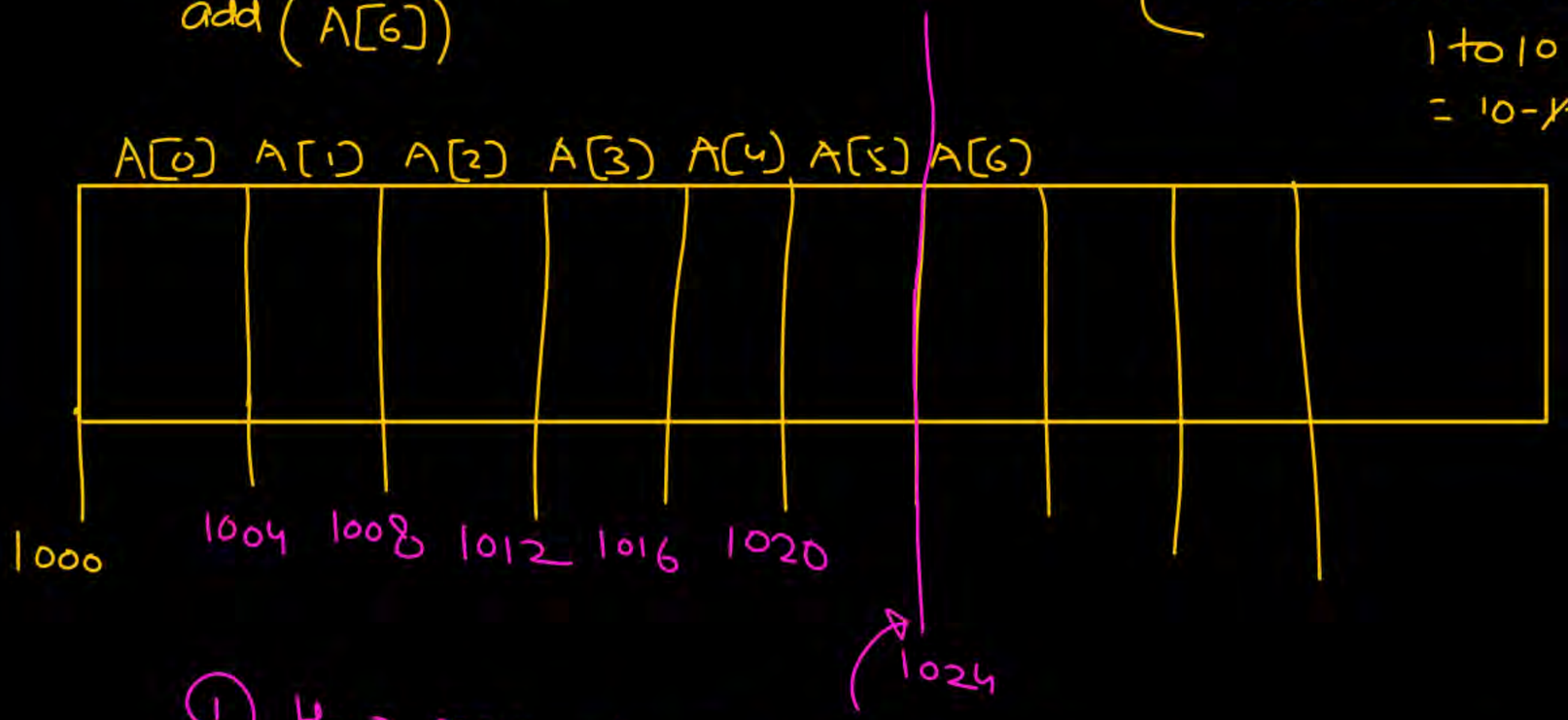
100 ↔ 200 (including both)

last - first + 1

200 - 100 + 1 = 101

1 to 10

= 10 - ~~x+1~~ = 10 ✓



② Memory already filled

= 6 x 4

= 24 byte

① How many ele. are already filled before A[6]?

⇒ index 0 to 5 = 5 - 0 + 1

= ⑥

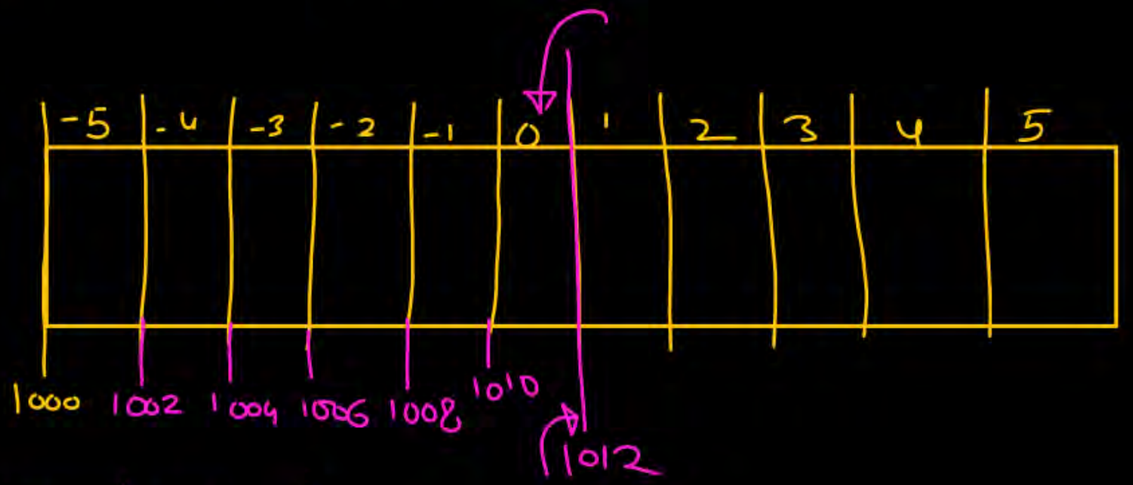
In Theory ⇒ index can start from -ve value also

A[-5..5]

w = 2 bytes

B.A = 1000

add(A[1])



① How many ele. are already filled before A[1]

= -5 to 0

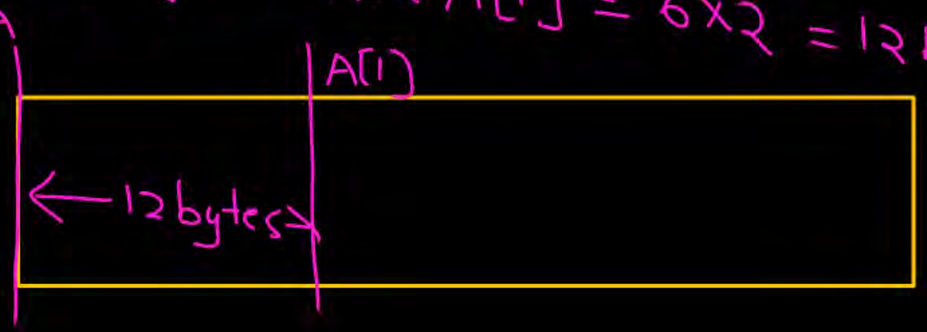
= 0 - (-5) + 1 = 6 element

② Memory already filled before A[1] = 6 x 2 = 12 bytes

add(A[1]) = BA + 12

= 1000 + 12

= 1012



Q

$A[-10..10]$

$\omega = 4$ bytes

B.A $\Rightarrow$ 104

add (A[6]) ?

elem. already filled before A[6] = -10 to 5

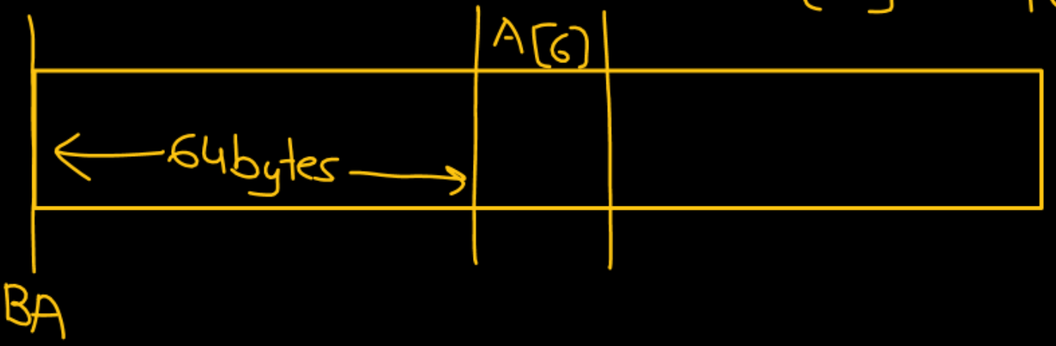
$= 5 - (-10) + 1 = 16$

Memory already filled before A[6] = $16 \times 4 = 64$ bytes.

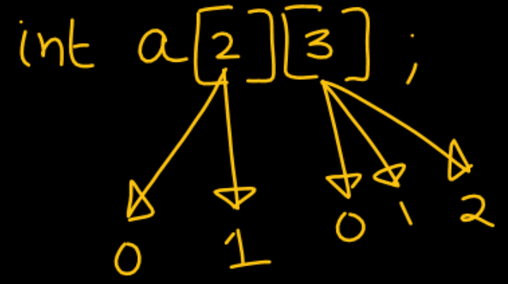
Add(A[6]) = BA + 64

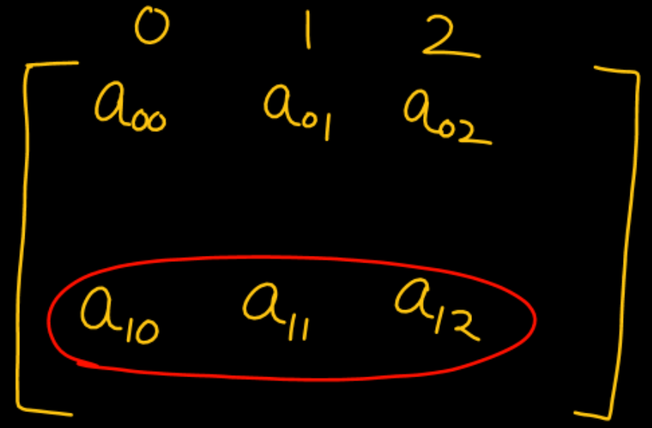
$= 104 + 64$

$= 168$



2-D array



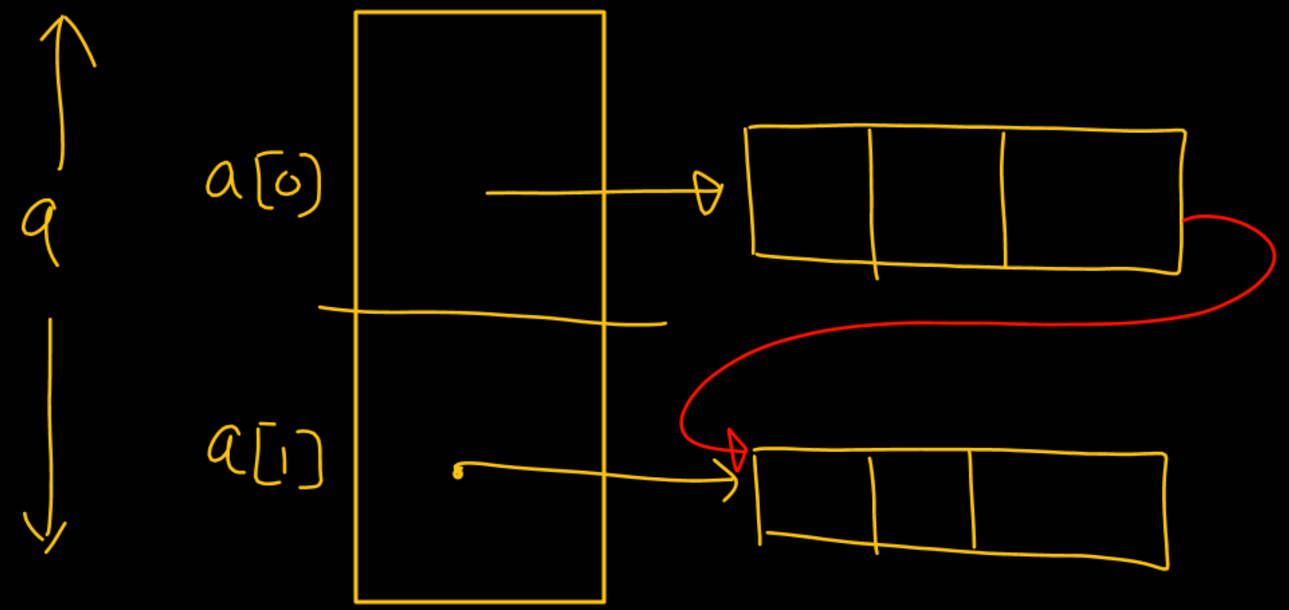


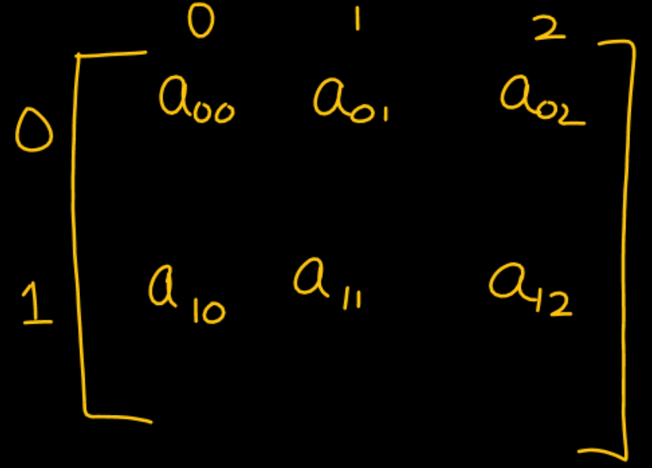
(No. of elements) row with index 0 = 3

No. of ele in row with index 1 = 3

Each row ⇒ 3 elements = No. of columns

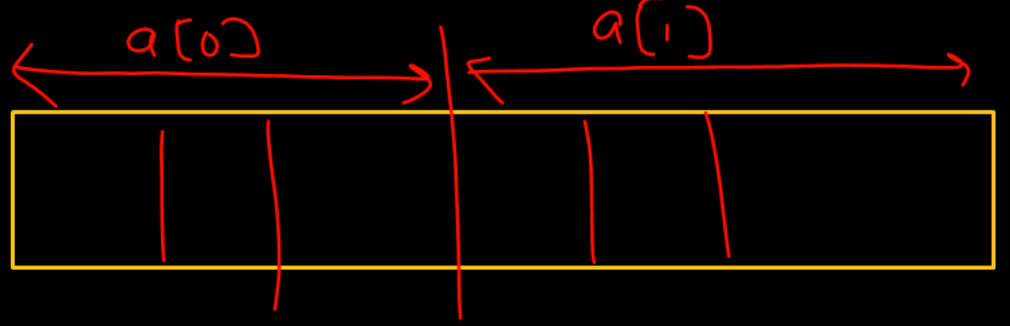
int a[2][3];

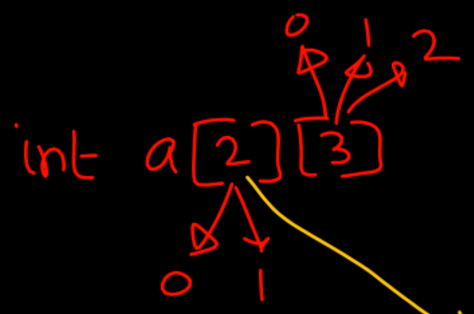




row a[0] ⇒ 3

row a[1] ⇒ 3





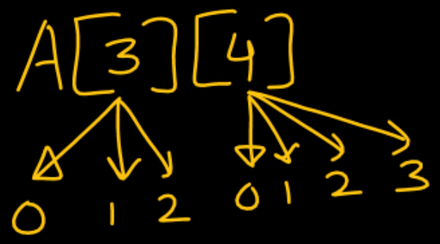
Each index/number in this dimension ⇒ 3 elements

$$\begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \end{array}$$

\# elem. in row with index 0 ⇒ 3

\# " " " " " 1 ⇒ 3

C-concept



	0	1	2	3
0	a_{00}	a_{01}	a_{02}	a_{03}
1	a_{10}	a_{11}	a_{12}	a_{13}
2	a_{20}	a_{21}	a_{22}	a_{23}

① random access

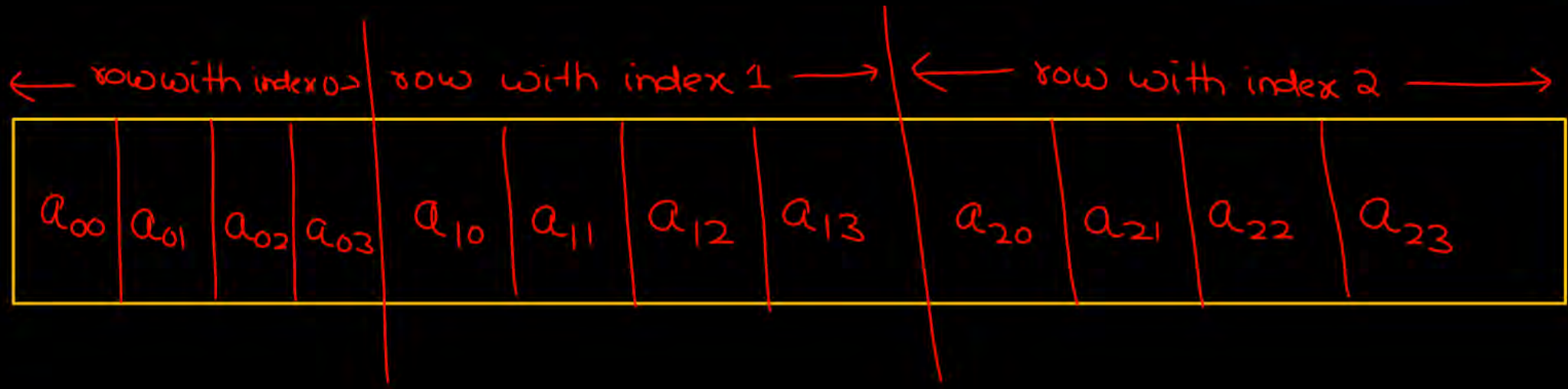
C-concept

A[3][4]

0 1 2 0 1 2 3

	0	1	2	3
0	a_{00}	a_{01}	a_{02}	a_{03}
1	a_{10}	a_{11}	a_{12}	a_{13}
2	a_{20}	a_{21}	a_{22}	a_{23}

Row-wise



C-concept

A[3][4]

0 1 2 0 1 2 3

	0	1	2	3
0	a_{00}	a_{01}	a_{02}	a_{03}
1	a_{10}	a_{11}	a_{12}	a_{13}
2	a_{20}	a_{21}	a_{22}	a_{23}

Every row ⇒ 4 elem.

Row-wise

add (a_{23})

∈2 index ⇒ 4 element

col with index 3

row index=2

row with index 0 | row with index 1 | row with index 2

1000												
	a_{00}	a_{01}	a_{02}	a_{03}	a_{10}	a_{11}	a_{12}	a_{13}	a_{20}	a_{21}	a_{22}	a_{23}

① How many rows already filled before row with index 2
= 0 to 1 = 1-0+1 = 2 rows

② elements already filled in row with index 2 before a_{23}
= 3 (0 to 2)
= 3-0+1 = 3

$a_{(2)3}$

Row-wise

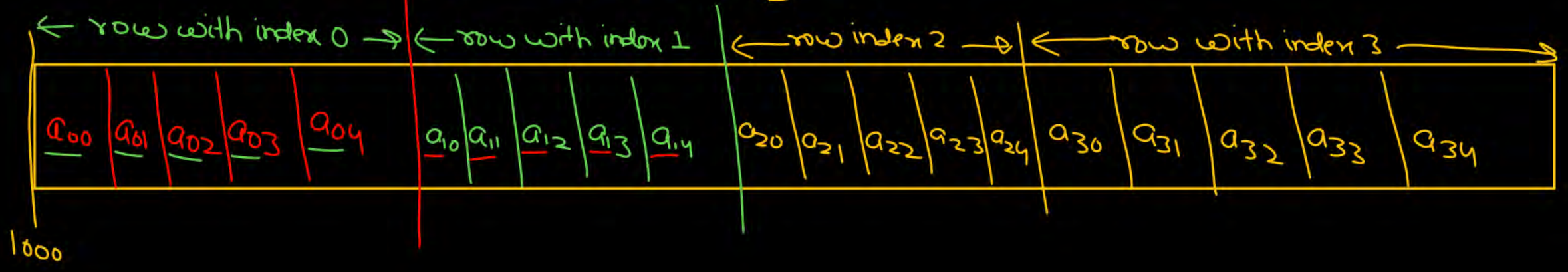
add(a_{33})

int A[4][5]

Every no/index in this dim rep = 5 elements

	0	1	2	3	4
0	a_{00}	a_{01}	a_{02}	a_{03}	a_{04}
1	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}
2	a_{20}	a_{21}	a_{22}	a_{23}	a_{24}
3	a_{30}	a_{31}	a_{32}	a_{33}	a_{34}

3 rows & 3 elements are already filled before a_{33}



Row-wise

int A[4][5]

Every no/index in this dim rep = 5 elements

add(a_{33})

- row with index 3
- col. index: 3

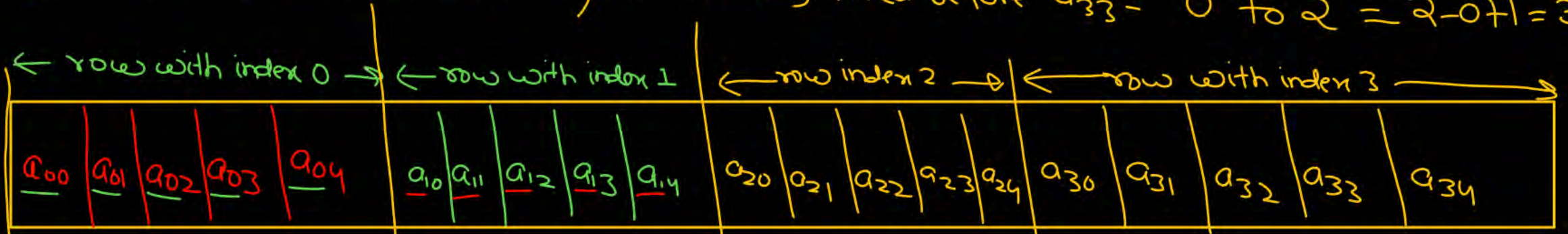
	0	1	2	3	4
0	a_{00}	a_{01}	a_{02}	a_{03}	a_{04}
1	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}
2	a_{20}	a_{21}	a_{22}	a_{23}	a_{24}
3	a_{30}	a_{31}	a_{32}	a_{33}	a_{34}

① Rows already filled before row with index-3 ⟹ 3

= index 0 to 2 = 2-0+1

= 3 rows

② Within row with index-3, ele. already filled before a_{33} = 0 to 2 = 2-0+1 = 3 ele



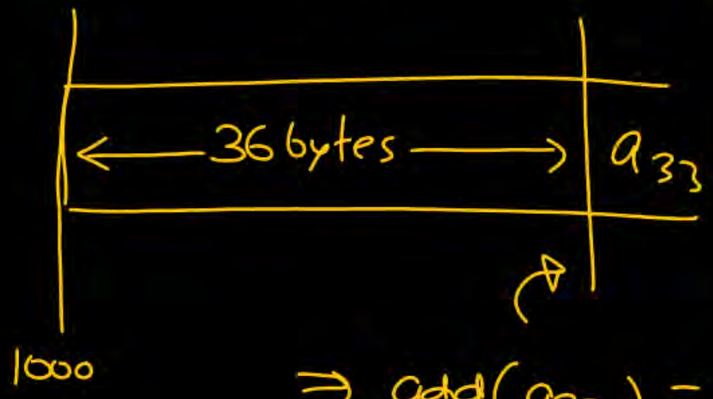
3 rows = 3 × 5 = 15

total elem already filled = 15 + 3 = 18

Elem already filled = 18

Size = 2

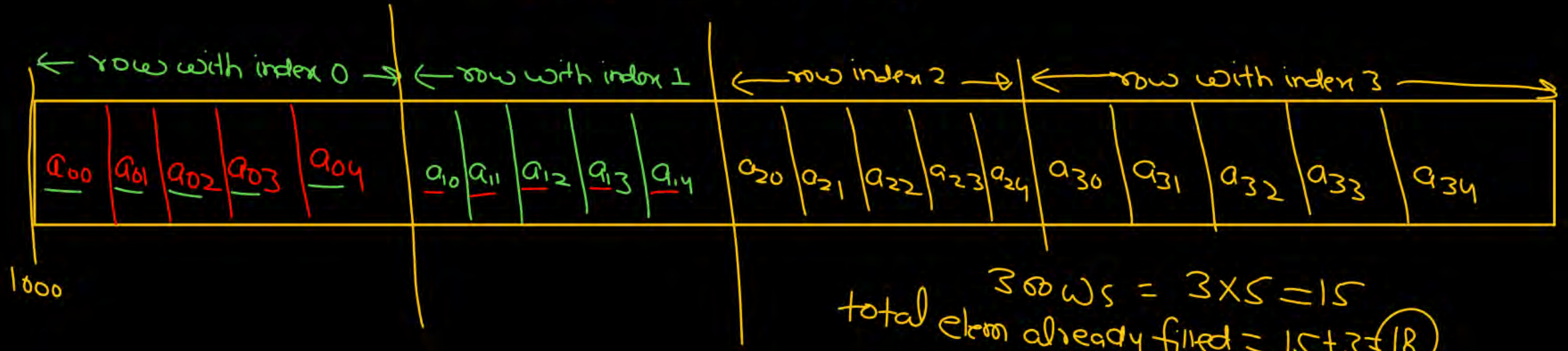
Memory already filled = 36 bytes



int A[4][5]

	0	1	2	3	4
0	a_{00}	a_{01}	a_{02}	a_{03}	a_{04}
1	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}
2	a_{20}	a_{21}	a_{22}	a_{23}	a_{24}
3	a_{30}	a_{31}	a_{32}	a_{33}	a_{34}

$\Rightarrow add(a_{33}) = 1000 + 36 = 1036$



3 rows = 3 × 5 = 15

total elem already filled = 15 + 3 = 18

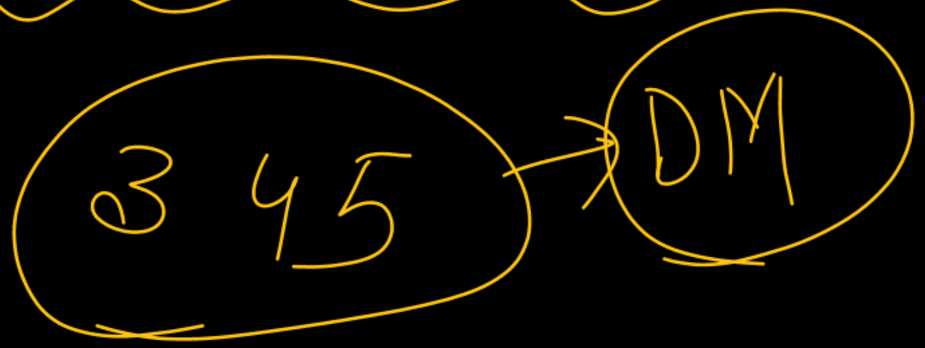
[C Recording ✓
DPPs ✓]

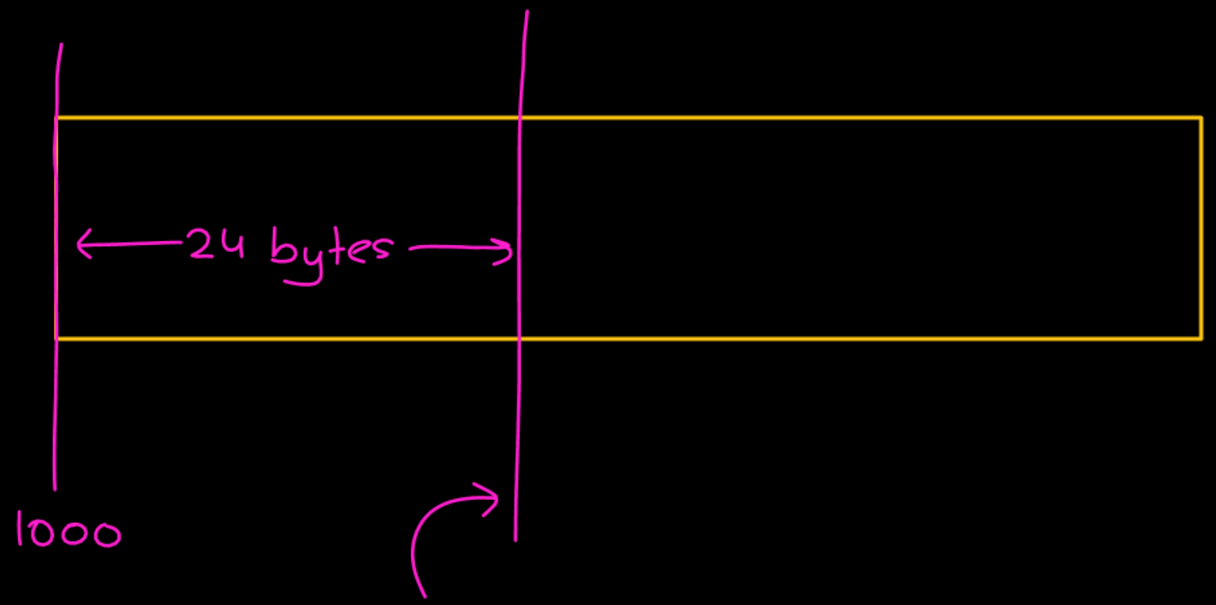
Problems ?

Row-major Order
(RMO)

Array - 1

Address calculation





B.A + 24

= 1000 + 24

= 1024


THANK YOU GW
SOLDIERS !