Allays: location of A[10]

↓

address

location of A[9][12] in a 2d array.

In ~~all the~~ all subjects, all question we solve will be gate level of above gate level.

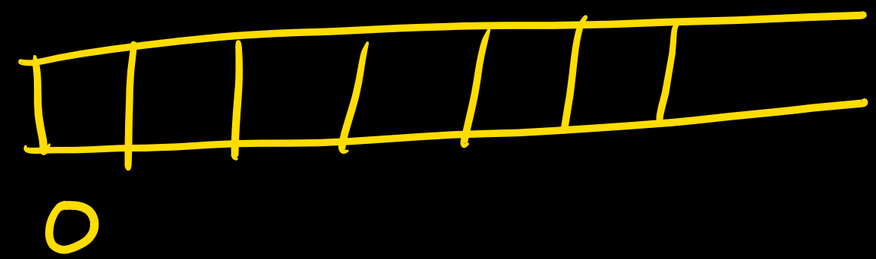
int a [110] = {10, 20, 30, 40, 50, 60 ..... 100]

This array need not be ~~C~~ any programming language.

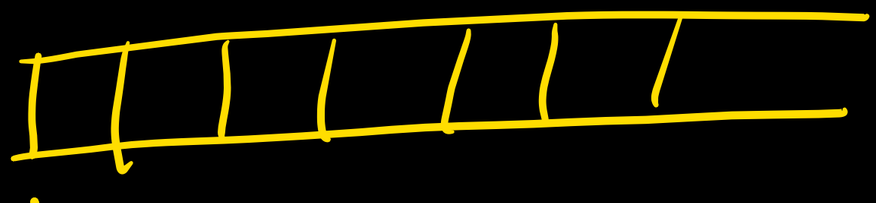
In programming languages we start with zero index.

In algo we start with 1 index in general

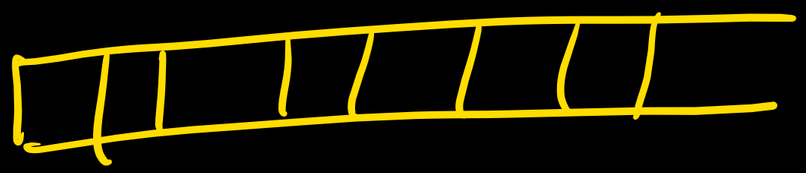
A starting index will be given in question



0



1



100

given in the question

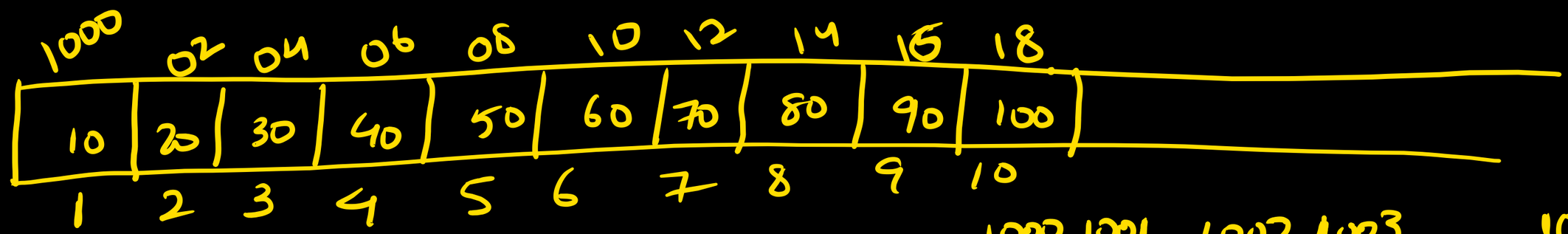
In C program:

int a[ ] = {10, 20, 30.. - .... 100}

no need to specify no size.

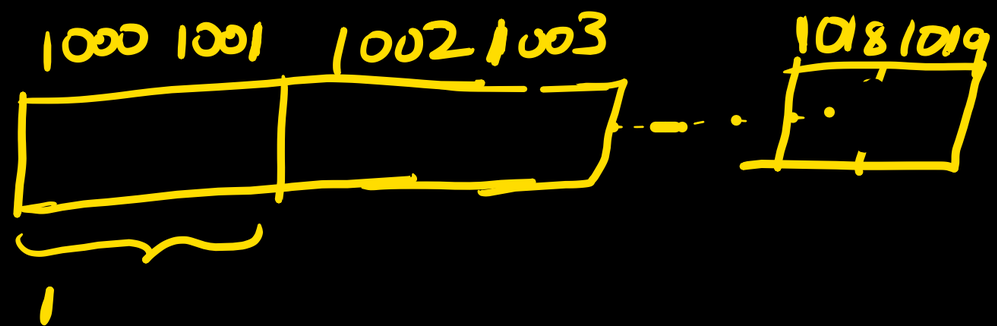
int a[ ]; → ~~not~~ wrong.

size is required.



Each cell is 2 bytes

First item add = 1000



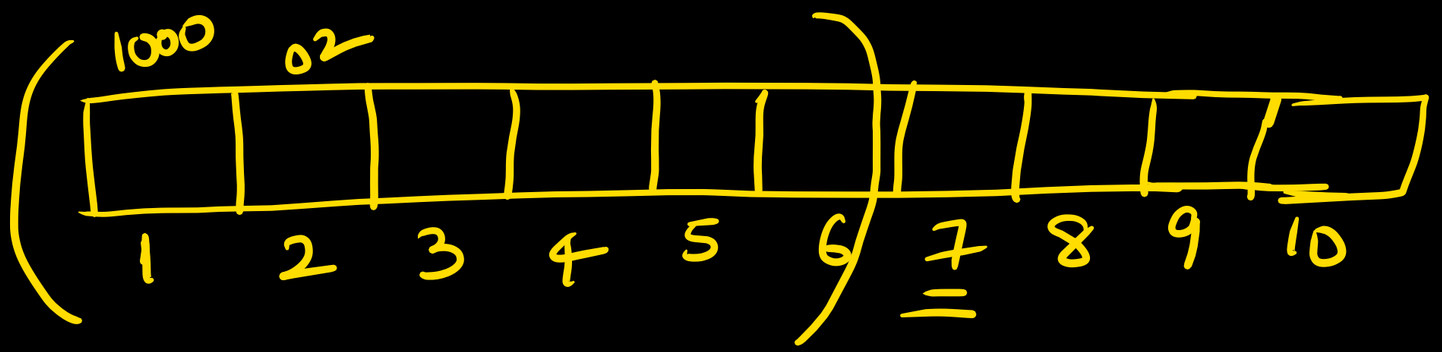
~~First~~ I address = 1000 Last address = 1019

total address = 1019 - 1000 + 1

= 20 bytes.

Total elements = 10 - 1 + 1

= 10 elements



Size of each element = 2 bytes

location(7) = BA + (7-1) * 2

(2 ↑ Size of each element)

= 1000 + 6*2

= 1012

(1 2 3 4 5)

(5 - 1) = 4 + 1 = 5

(5 - 1) + 1

1 2 3 4 5 6 7

2 3 4 5

7 - 3 + 1 = 5.

100 ... 200

(200 - 100 + 1)

Gate A [95 ..... 300]

first

last

Total elements = 300 - 95 + 1 = 206

BA = 1000 Size of each element = 5

↓

first element address

loc(A[278])

(95 96 97 ..... 277) 278

277 - 95 + 1 (278 - 95)

BA + (278 - 95) * 5

1000 + (278 - 95) * 5

= 1915.

A [-300 ..... +500]

BA = 0 Size of each element = 10 loc [A(-4)]

(-300, -299, -298 + ... - 5) (-4) -3 -2 -1 0 .....

we have to cross all these elements

$$0 + (-4 - (-300)) * 10 = 2960$$

or

$$-5 - (-300) + 1$$

$$= (-4 - (-300)) \checkmark$$

Formula : $A[lb, \ldots\ldots ub]$

BA, $Size$

location $A[i]$

$$\left(Loc(A[i]) = BA + (i - lb) * Size \right)$$

formula.

Why every programming language best starting index as 0?

$BA + (i - lb) * size.$

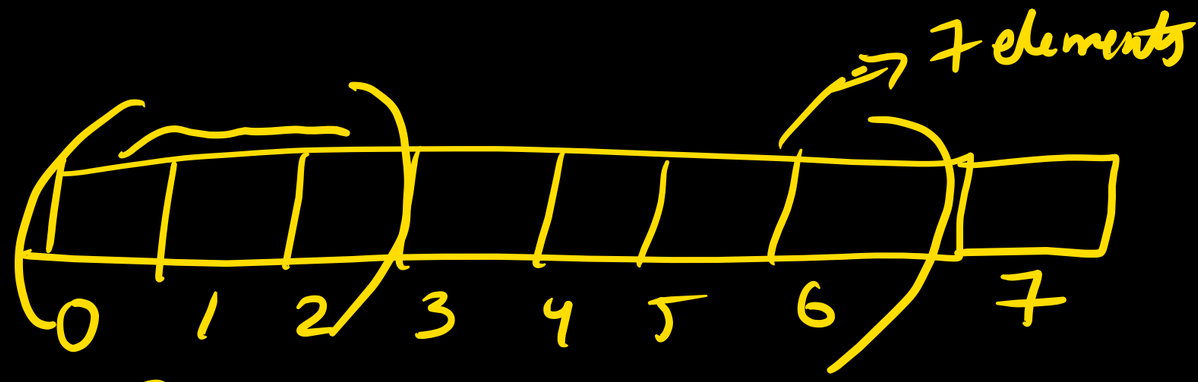
i → 0

lb is always zero if starting add is '0'.

$BA + i * size$

no need of substraction

∴ less time is taken..



BA = 1000

Loc = 7

Loc Loc(7) = 1000 + (7 − 0) * size.

Loc (3) = 1000 + (3 − 0) * size.

t

2D arrays:

C prog in not required for DS.

```
int a[][] = {10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120}
```

12 element

not filled

not allowed.

12 X 1
2 X 6
3 X 4
4 X 3
6 X 2
1 X 12

possibilities.

int a [ ] [3] = { 10, 20, 30 ..... 120 }

3 Columns

4 row 3 Columns

$\frac{12 \text{ element}}{3 \text{ col}} = 4 \text{ rows}$

a

	1	2	3
1	10	20	30
2	40	50	60
3	70	80	90
4	100	110	120

Row major order

10	20	30	40	50	60	70	80	90	100	110	120
0	1	2	3	4	5	6	7	8	9	10	11

I

How do you read it.

2D array has to be stored in 1 dimensional array

Rmo Cmo

Loc a[3][3]

Cmo

10	40	70	100	20	50	80	110	30	60	90	120

Row major order :

BA = 1000

Size = 2

	1	2	3	4
1	10	20	30	40
2	50	60	70	80
3	90	100	110	120
4	130	140	150	160

1000	02	04	06	08	10	12	14	16	18	20	22	24	26	28	30
10	20	30	40	50	60	70	80	90	100	110	120	130	140	150	160

I row II row III row IV row

(4-1)*4

Location of A[4][3]

BA
1000 +

Cross 3 rows

((4-1)*4 + (3-1))*2

Current row

(3-1)

Row A[1....4, 1....4]

4-1+1=4 4-1+1=4

A[27, ... 71] [43 ..... 120]

Gate

// nr

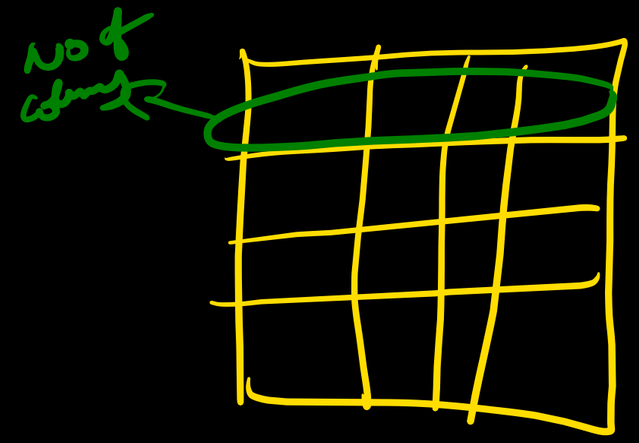
(71 - 27 + 1)

45

// nc

(120 - 43 + 1)

78



Size of each row = 78 elements

BA = 1000 Size = 10, Rmo, Loc (A[50][110])

27
28
29
:
49
50

$$BA + [(50-27) * 78 + (110-43)] * 10$$

1000

$$= 19610$$

Q: A[-93, ... +100] [-25+..... 25]

BA = 0 Size = 1 RMO A[95][20]

$$0 + ((95 - (-93)) * 51 + 20 - (25))) * 1$$

$$= 9633$$

Formula for RMO:

$$A[lb_1 \ldots\ldots ub_1, lb_2, \ldots\ldots ub_2]$$

BA, size, RMO

Loc A[i][j]

$nc = (ub_2 - lb_2 + 1)$

$$= BA + ((i - lb_1) * nc + (j - lb_2)) * size$$

$(i - lb_1)$ → number of rows of size of each row

Column major order:

	1	2	3	4	5
1	11	12	13	14	15
2	21	22	23	24	25
3	31	32	33	34	35
4	41	42	43	44	45

A[3][5]

2

CMO

1000	02	04	06	08	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38
11	21	31	41	12	22	32	42	13	23	33	43	14	24	34	44	15	25	35	45
I CL				II CL				III CL				IV CL				IV CL			

A [+25 ..... +97] [-16, ... +200]

nR

97-25+1 = 763

200+16+1 = 217

BA=0 S=5 Size=5 CMO

Loc (A[90][180])

-16, -15, -14 ..... +0, +1 .....) 180

$$BA + [(180 - (-16) * 73 + (90 - 25)) * 5.$$

$$= 71\,865$$

A [-42, ... +42] [-84 ..... 84]

= ⇓

nr = 85 nc = 169

BA = 1000 , Size = 10 CMO

location [A[-3][-4]]

1000 ((-4 - (-84)) * 85 + -3 - (-42)) * 10

= 69390

$A(lb_1 \ldots\ldots ub_1)(lb_2 \ldots\ldots ub_2)$

BA, size, cmo

loc $A[i][j]$

$BA + [(j - lb_2) * nr + (i - lb_1)] * sz$

$A[i][j]$

$ub_1 - lb_1 + 1$

In gate exam And interview exams only 2D are asked.

Shall we discuss 3D also.

5 min break;

why to risk.

3D arrays:

int a [3 ..... 9] [5 ..... 15] [8 ..... 24]

3 → 7 → 2D arrays.

5 → 11 rows

8 → 17 columns

RMO

Size = 2

BA = 1000

loc [a[7][12][20]]

(3 4 5 6) 7

4

$$1000 + \Big((7-3) * 11*17 + (12-5) * 17 + (20-8) \Big) * 2$$

$$= 2758$$

a[3..... 9] [5..... 15] [8 ..... 29]

↓↓ NR

BA = 1000 , Size = 2 , CMO.

location a[7][12][20]

1020

$$1000 + \left[(7-3)*11*17 + (20-8)*11 + (12-5)\right]*2$$

$$= 2774 \checkmark$$

a[-10 ..... +10, -25 ..... 25, -9 ..... +29]

51 ⇓ , 39 ⇓

BA = 0 Size = 10 , RMO location (a[8][20][15])

$$0 + \Big((8-(-10)) * 51 * 39 + (20-(-25)) * 39 + (15-(-9)) \Big) * 10$$

$$= 375810$$

CMO

$$0 + (8-(-10)) * 51 * 39 + (15-(-9)) * 51 + (20-(-25)) * 10$$

$$= 370710$$

Formula:

$$a[lb_1 \ldots\ldots ub_1, lb_2 \ldots\ldots ub_2, lb_3 \ldots\ldots ub_3]$$

BA, Size, RMO

$$loc[A[i][j][k]]$$

$$BA + [(i - lb_1)\, nr * nc + (j - lb_2) * nc + (k - lb_3)] * Size.$$

Formula

$a[lb_1 \ldots\ldots ub_1, lb_2 \ldots\ldots ub_2, lb_3 \ldots\ldots ub_3]$

BA , size ; ~~R~~ CMO loc(A[i][j][k])

CMO:

$BA + (((i - lb_1) * nr * nc + (k - lb_3) * nr + (j - lb_2)) * size$

$(ub_2 - lb_2 + 1)$ $(ub_3 - lb_3 + 1)$

LL
↓
DAC
↓
mimics
↓
arrays
↓
stacks
↓
queues

asymptotic notations
↓
Comparing the functions
↓
iterative analysis
↓
Recursion
↓
Recursive equations
↓
Sub | tree | master theorem

Arrays
- 2D < CMO / RMO
- 3D < CMO / RMO

Doing part of algo in over
Linked list → most occation and interview.