Important points

1. Array is a indexed collection of homogenous elements.
2. Array element always points to first memory address.
3. Array name cannot be used with ++ -- =
4. Every Array is internally converted into pointers except when applied with sizeof( ) and &(addr of)

int id1;
int id2;
int id3;
-
,
,
:

$\Rightarrow$ int id[100];

Declaration of Arrays

1. int a[5]; ✓
2. int a[1]; ✓
3. int a[0]; $\rightarrow$ Ans1 c89 / $\boxed{c90}$ ✓
4. int a[-5]; ✗
5. int a[40000]; $\rightarrow$ ✗ / ✓

16bit $\Rightarrow$ 0 to $\boxed{65536}$

32bit $\Rightarrow$ 0 to 2^{32} ✓

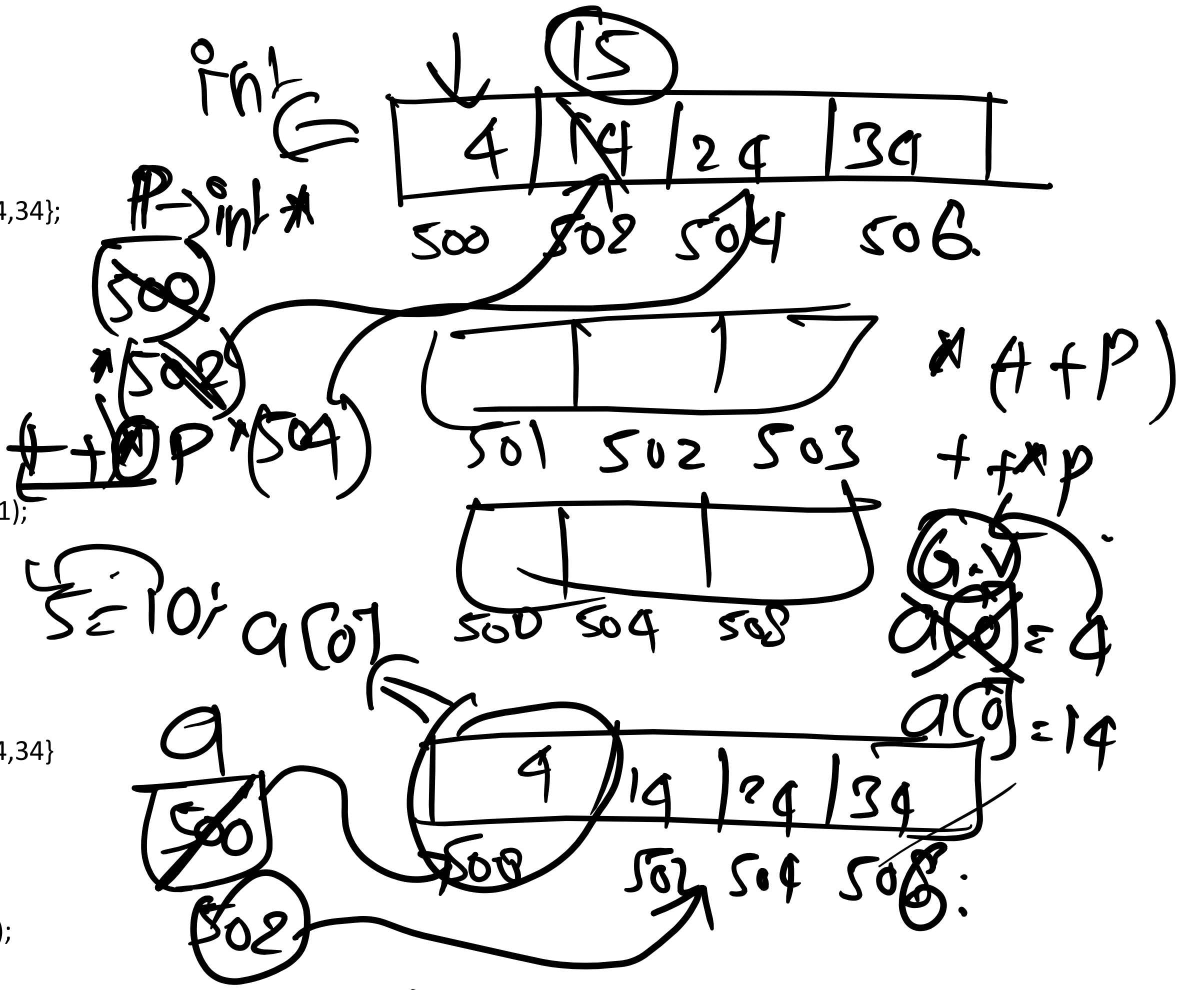
6. char a[40000]; ✓
7. int a[3.5]; ✗
8. int a['a']; ✓ ($\underset{97}{'a'}$)
9. int a[4>3]; ✓ ($\overset{1}{4>3}$)
10. int a[(int)3.5]; ✓ ($\overset{3}{(int)3.5}$)

Initialization

1. int a[]; ✗
2. int a[5]; ✓
3. int a[5] = {1,2,3,4,5}; ✓
4. int a[] = {1,2,3,4,5}; ✓ ($\overset{5}{[\ ]}$)
5. int a[6] = {1,2,3,4,5,6,7}; ✗
6. int a[5];
   a[0] = 'a';
   a[1] = 3.5; ✓
   a[2] = a[0] + a[1]; ✓

++*p
*++p

```
int a[]={4,14,24,34};
int *p;
p=&a[0];
++p;
++*p;
printf("%d",*p1);
```

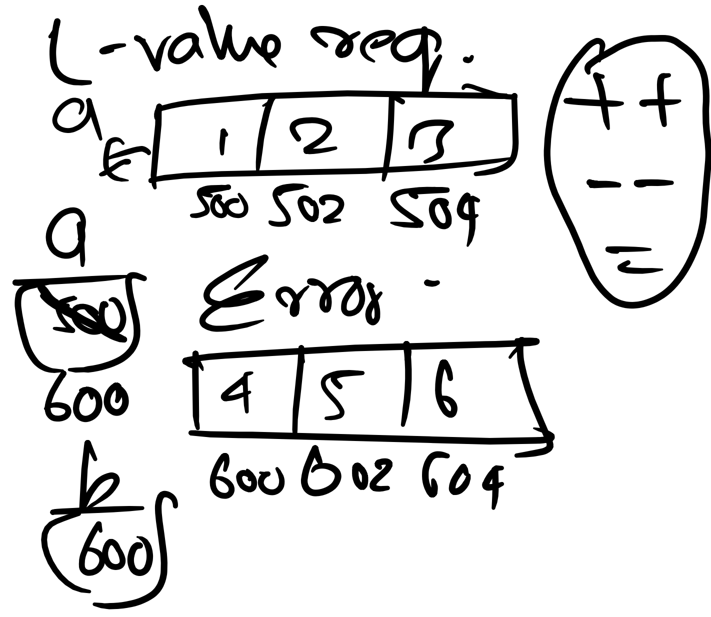


```
int a[]={4,14,24,34}
++a;
++*a;
printf("%d",*a);
```

⇓ Error
L-value req.

a[0]=>

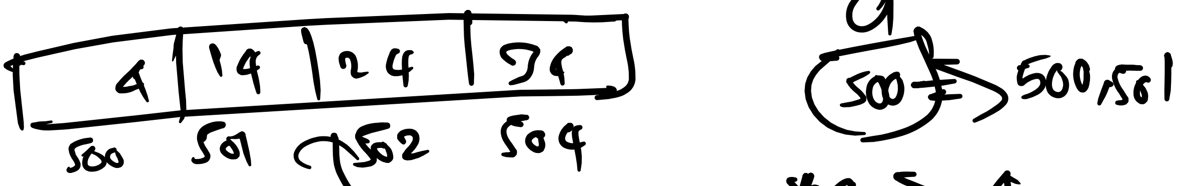
int a[] = {1,2,3};
int b[] = {4,5,6};
a = b
pf(a[0], b[0]);



int x=10, y=20, z=30;
int a[] = {x, y, z}; X
int a[10]={1,2,3}; 00.

int a[5];
⇓
G.V.

```
int a[5]={10,'a',3.5,3.4}; ✓
int a[]={4,14,24,34};
printf("%d", a[2]); => 24
printf("%d",2[a]); => 24.
```



a[0] => 4 → *a => 4

*(a+0)

500+1

*(502)

a[1] => 14 *(a+1) => 14

$a[i] \Rightarrow *(a+i) \Rightarrow *(i+a) \Rightarrow i[a]$

① 1+2[a] => 25 ② 2[1+a]

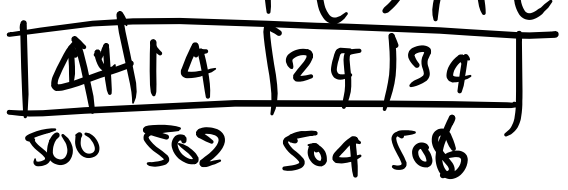
1 + *(2+a) 24 500 504

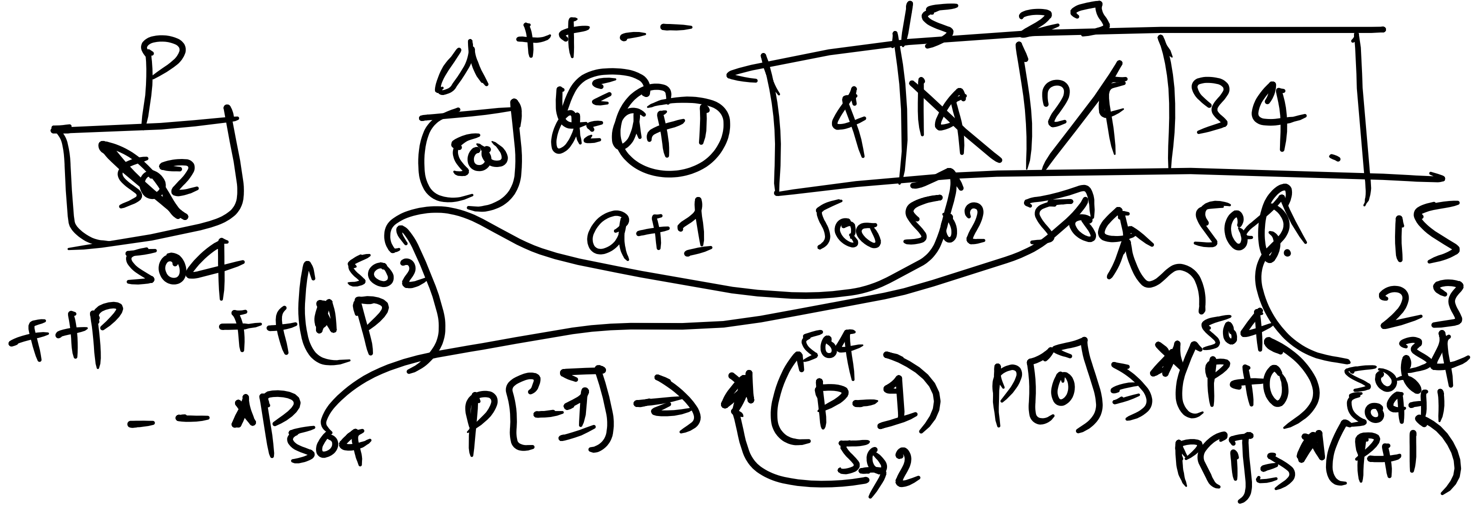
*(2+1+a) *(3+a)

*(a+3) => a[3] => 34

```
int a[]={4,14,24,34};
int *p;
p=a+1;
++*p;
++p;
--*p;
```

```
printf("%d %d %d", p[-1],
                   p[0], p[1]);
```





float f = 3.5;

float* fp = &f;

sizeof (fp); 2

⇓

500

int i = 4;

int *ip = &i

sizeof (ip); 2

⇓

500

int a[] = {1, 4, 6}

sizeof(a);

⇓

$\boxed{6}$

a+1

500 ⇒ 500, 501

int a[] = {50, 40, 60, 70};

500 502 504 506

a+1 ⇒ 502 (502, 503)

&a

500 ⇒ 500 → 507

&a+1 ⇒ 508

500
↓
500, 507

sizeof(a+1); $\boxed{8}$ $\boxed{6}$

sizeof(a) ⇒ 2

int a[] = {1, 2, 3, 4}; sizeof(a); ⇒ 8

1. array name ++ -- =
2. Every Array into pointer
3. Except size &

sizeof(a+1); 2

2D Array

int c[60];
int e[60];
int m[60];
int B[60];

⇒ int a[5][4]={{1,2,3},
{4,5,6}}
0000

int colg[4][60]; ✓

int a[][]; ✗

int a[][]={1,2,3,4,5,6,7,8}; ✗

$\text{int } a[\underset{3}{\;}][4]=\{\underbrace{1,2,3,4}_{r_0},\underbrace{5,6,7,8}_{r_1},\underbrace{9,0,0,0}_{r_2}\}$ ✓

int a[3][5]={10,20,30,40,50,60,70,80,90,
100,110,120,130};

r0					r1					r2				
c0	c1	c2	c3	c4	c0	c1	c2	c3	c4	c0	c1	c2	c3	c4
10	20	30	40	50	60	70	80	90	100	110	120	130	0	0
500	502	504	506	508	510	512	514	516	518	520	522	524	526	528

a = 500 ⇒ (500,509).

$a[2][3] \Rightarrow 0$ $*(2[a]+3) \Rightarrow (*(*(2+a)+3))$.

$2[a][3] \Rightarrow$

$*\boxed{a[i]}[j] \Rightarrow *(a[i]+j) \Rightarrow *(*(a+i)+j)$.

$\&a = 500(500-529)$

$a = 500\ (500 \rightarrow 509)$

$*a \Rightarrow 500(500 \rightarrow 501)$.

$**a \Rightarrow 10$

$a[0][0] \Rightarrow 10$

$*(*(a+0)+0)$

$\|$

$**a \Rightarrow 10$

① $*a+2$

② $a[1]+2$

③ $*(a+2)$

④ $*(a[2]+1)$

⑤ $*(*a+1)$.

⑥ $*(*(a+1)+2)+3)$.

*

$a \Rightarrow 500(500-509) \Rightarrow 1R$

$a+1 \Rightarrow 510 \Rightarrow (510-519)$

$(*a)+1$

$\downarrow$

$500(500/501) \Rightarrow 502$

$R+1 \Rightarrow NR$

$C+1 \Rightarrow NC$

$a \Rightarrow R$.

$*\overset{a}{\cancel{R}} \Rightarrow C$

$*C \Rightarrow V$

$35 \Rightarrow 15 \times 2$

① $*a+2$ sized(a) $\Rightarrow \underline{\underline{30}}$

$\Downarrow$

(500,509)

$*(500,509)$

$\Downarrow$

$(500,501)+2 \Rightarrow (504,505)$ ✓

② $a[1]+2$ R +I $\Rightarrow$ NR

$*(a+1)+2$ $*R \Rightarrow C$

$\Downarrow$ R

$(500,509)+1$

$\Downarrow$

$*(510,519)$

$(510,511)+2 \Rightarrow (514,515)$.

512,513,

③ $*(a+2)$.

$\Downarrow$

$(500,509)+1 \Rightarrow (510,519)+1$

$\Downarrow$

$*(520,529) \Rightarrow (520,521)$

④ $*(a[2]+1)$ R +I $\Rightarrow$ NR +I $\Rightarrow$ NR

C +1 $\Rightarrow$ NC $\boxed{120}$

$*(*(a+2)+1)$

$\Downarrow$

$(500\text{-}509)+2 \Rightarrow *(520,529)$

$\Downarrow$

$(520,521)+1 \Rightarrow *(522,523) \Rightarrow 120$

⑤ $*(*a+1)$

$*R \Rightarrow C$

$*C \Rightarrow V$

$*(500,509) \Rightarrow (500,501)+1$

$\Downarrow$

$*(502,503) \Rightarrow 20$

⑥ $*(*(a+1)+2)+3$

$(500 \to 509)+1$

$\Downarrow$

$*(510 \to 519) \Rightarrow (510,511)+2$

$*(514,515) \Rightarrow 80+3$

$\boxed{83}$

3D Array. int a[5]; ⇒ class each id;

int a[5][60];

Row Colums.

int [3][5][60];

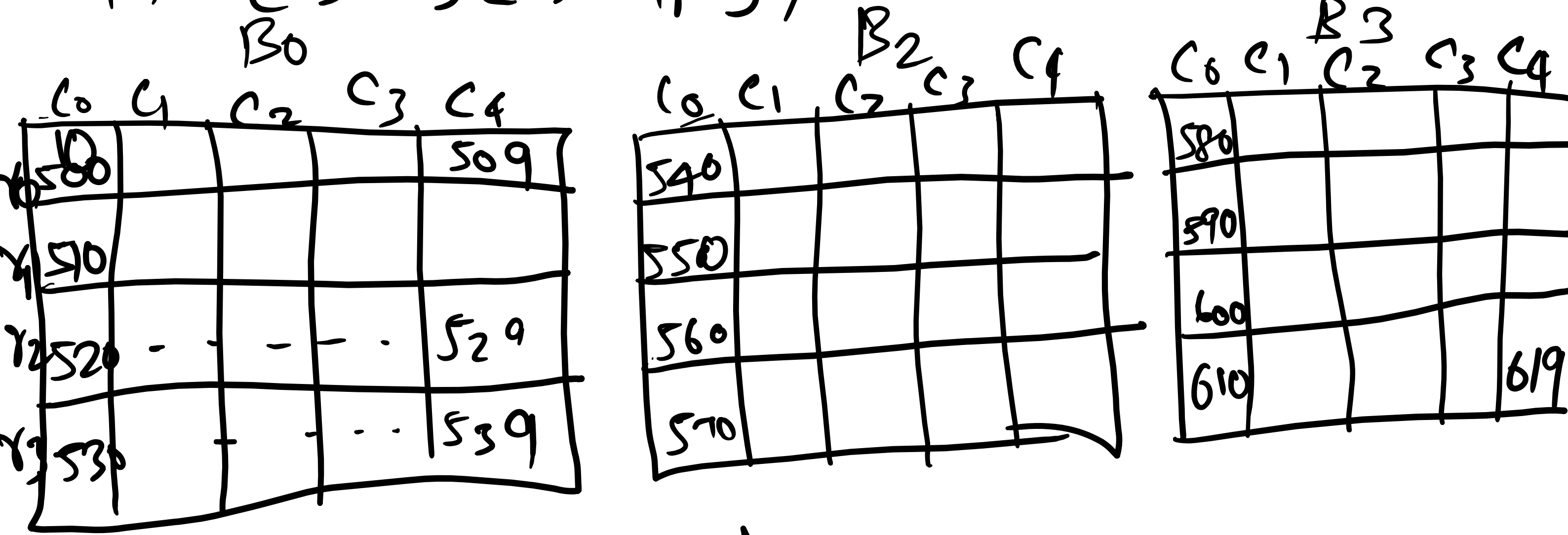
Block rows columns.

int [][][] = ~~{1,2,3}~~;

{ { → B0
{ → B1
{C0, C1},
r2
{ }

int [ ][2][3]; = {10} . . . . . .

int [3][4][5] = {10};



&a ⇒ 500 (500 - 619)

a ⇒ 500(500→539) 1Block address

*a ⇒ 500(500→509) 1Row add.

**a ⇒ 500 (500/501)

***a ⇒ 10

a[0][0][2]

1B2R

*a+1 ⇒

a+1 ⇒ 540

a[i][j][k] ⇒ *(a[i][j]+k)

((a[i]+j)+k) ⇒ *(*(*(a+i)+j)+k).

B+1 ⇒ NB

R+1 ⇒ NR

C+1 ⇒ NC

*B ⇒ R

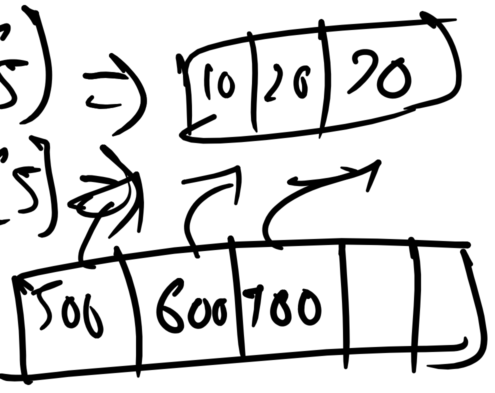
*R ⇒ C

*C ⇒ V

Array of pointer

int a[5] =>

int * p[5] =>



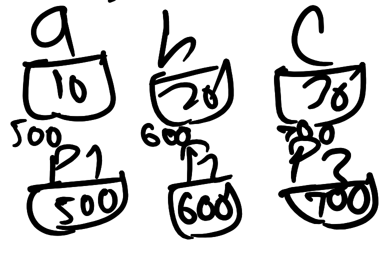
a, b, c, d, e

int *p1, *p2, *p3*, *p3 } => int *p[5]

(*p[0]   *p[1]   *p[2]   *p[3]   *p[4]) → int

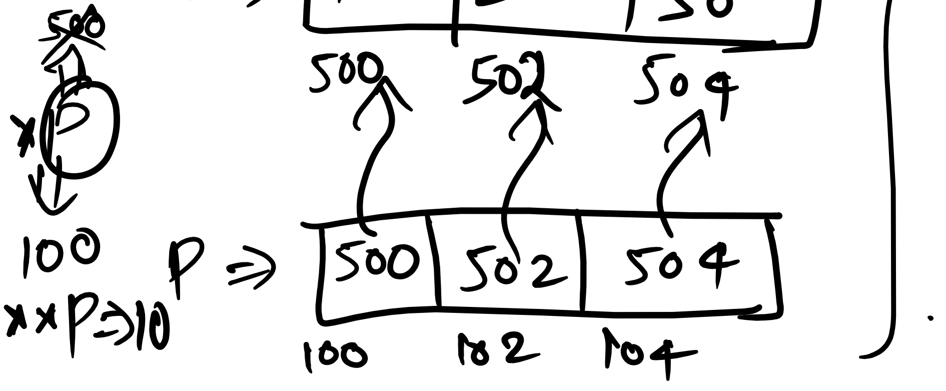
(p[0]   p[1]   p[2]) → int *

```
int a, b, c;
int *p1, *p2, *p3
  p1 = &a
  p2 = &b
  p3 = &c
```



```
int a[3]={10,20,30};
int *p1, *p2, *p3;
  p1 = a;
  p1 = a+1;
  p2 = a+2
```

```
int a[3]={10,20,30};
int *p[3];
p[0]=a;
p[1]=a+1;
p[2]=a+3;
```



① p $\overset{int**}{\Rightarrow}$ 100 (100,10)

② p[0] ⇒ 500 ⇒ int*

③ *p ⇒ ~~100~~ ⇒ 500

④ *p[0] ⇒ *(500) 10

⑤ **p ⇒ 10
  ((100)) (500)

⑥ *p[1] ⇒ *(*(p+1)) ⇒ 20

⑦ ++*p ⇒ 502

⑧ ++**p ⇒ 11

⑨ *++*p ⇒ 20

⑩ ++*++*p ⇒ 21

⑪ **++p ⇒ ~~10~~ lvalue req.

⑫ ++*p[1] ⇒ ++*(*(p+1))
  21 102 502

⑬ ++*++p[0]
  ⇓ 21
  ++*++(*(p+0))
  20 502 500 100

⑭ p[1][1] ⇒ 20.
  ++*(*(p+1))
  502 20 102 100