CS & IT ENGINEERING

C Programming

Arrays and Pointers

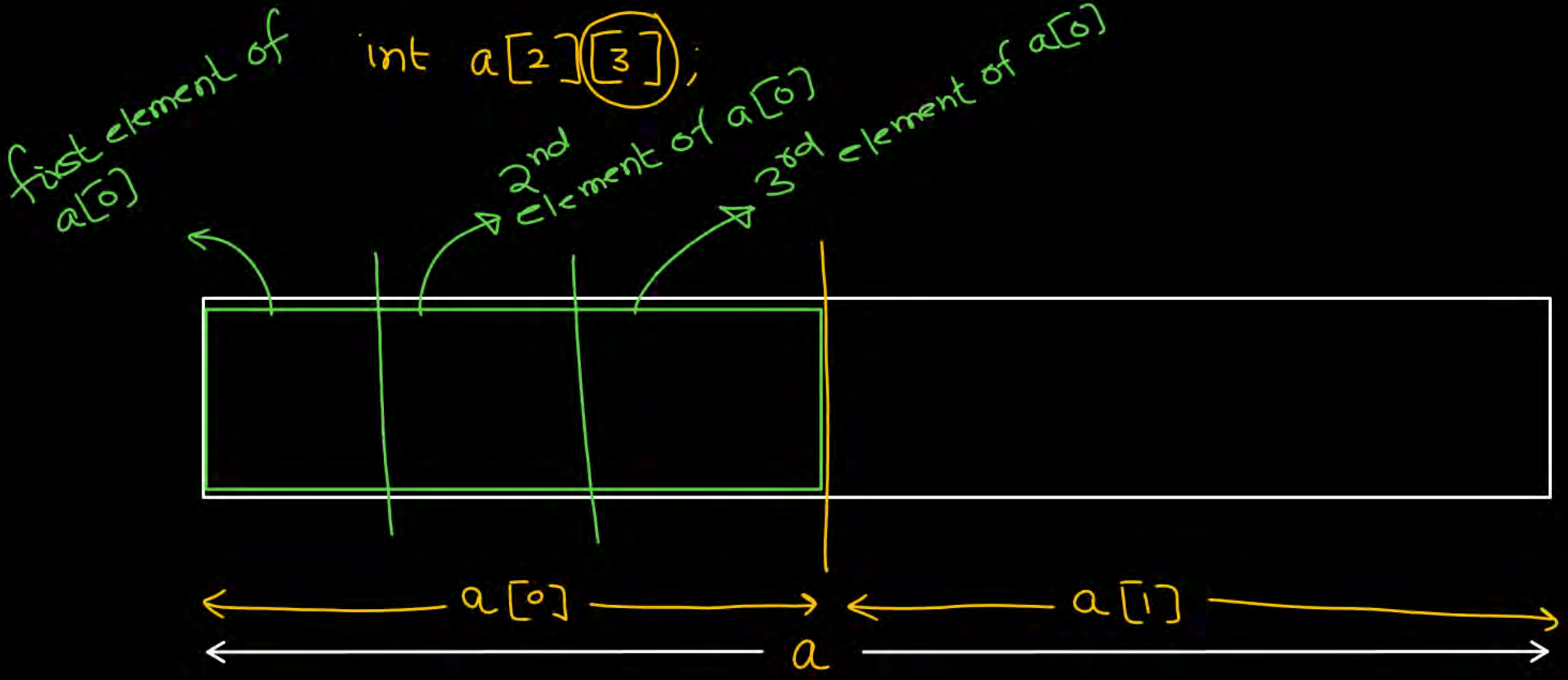
Lec - 03

By- Pankaj Sharma Sir

Arrays and Pointers (Part- 03)

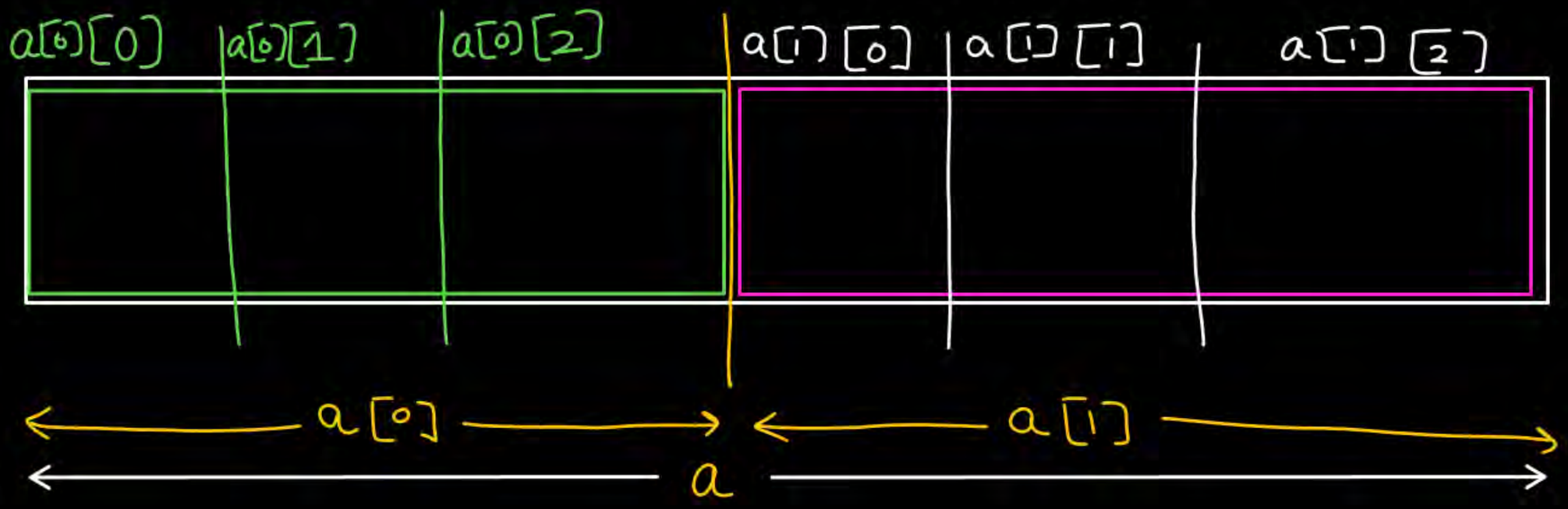
2-D array

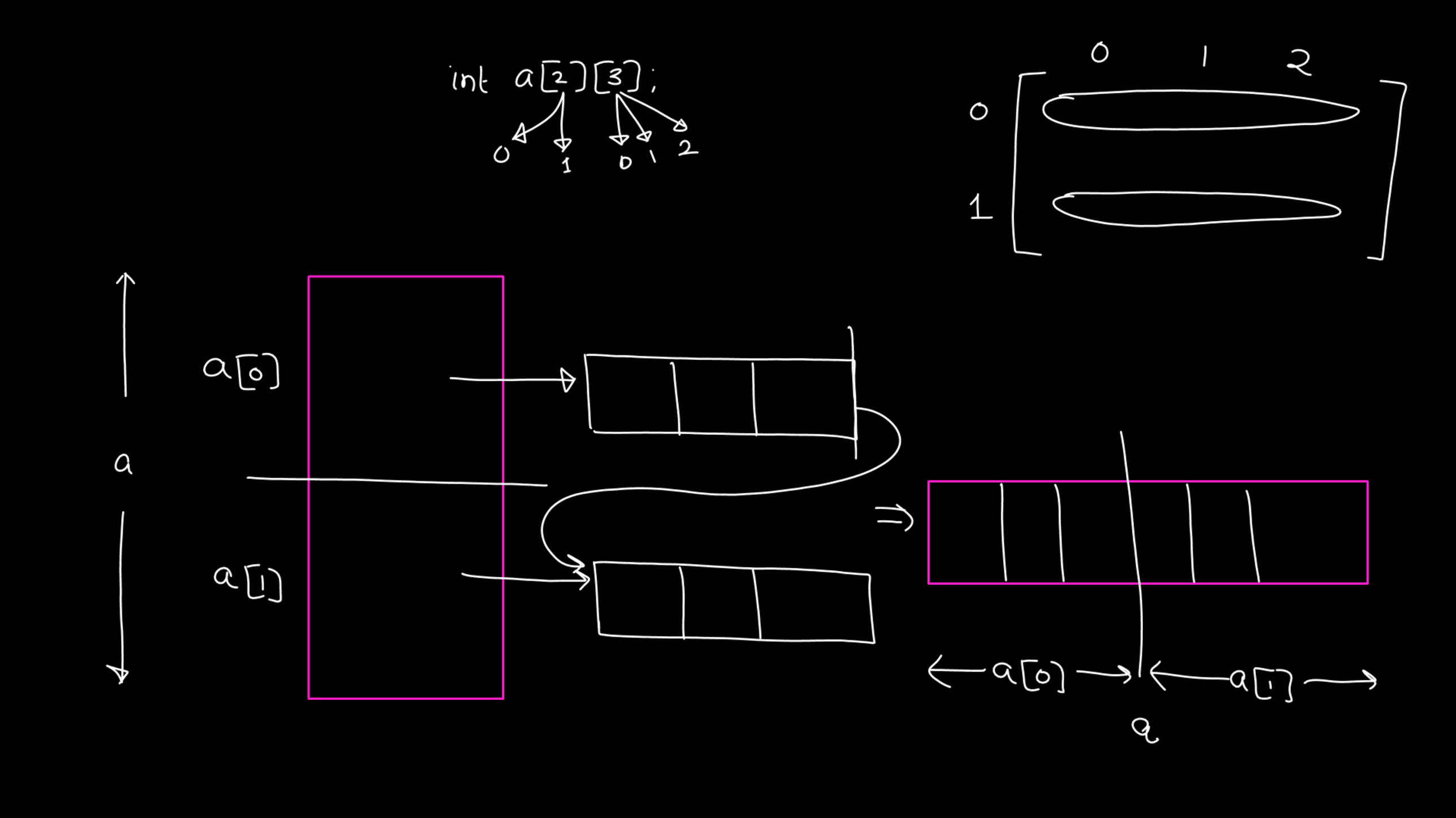
int a[2][3];



2-D array

```
int a[2][3];
```




int a[2][3];
0
1
0
1
2
0
1
2
0
1
a
a[0]
a[1]
a[0]
a[1]
a

```
int a[2][3] = {1,2,3,4,5,6};
```

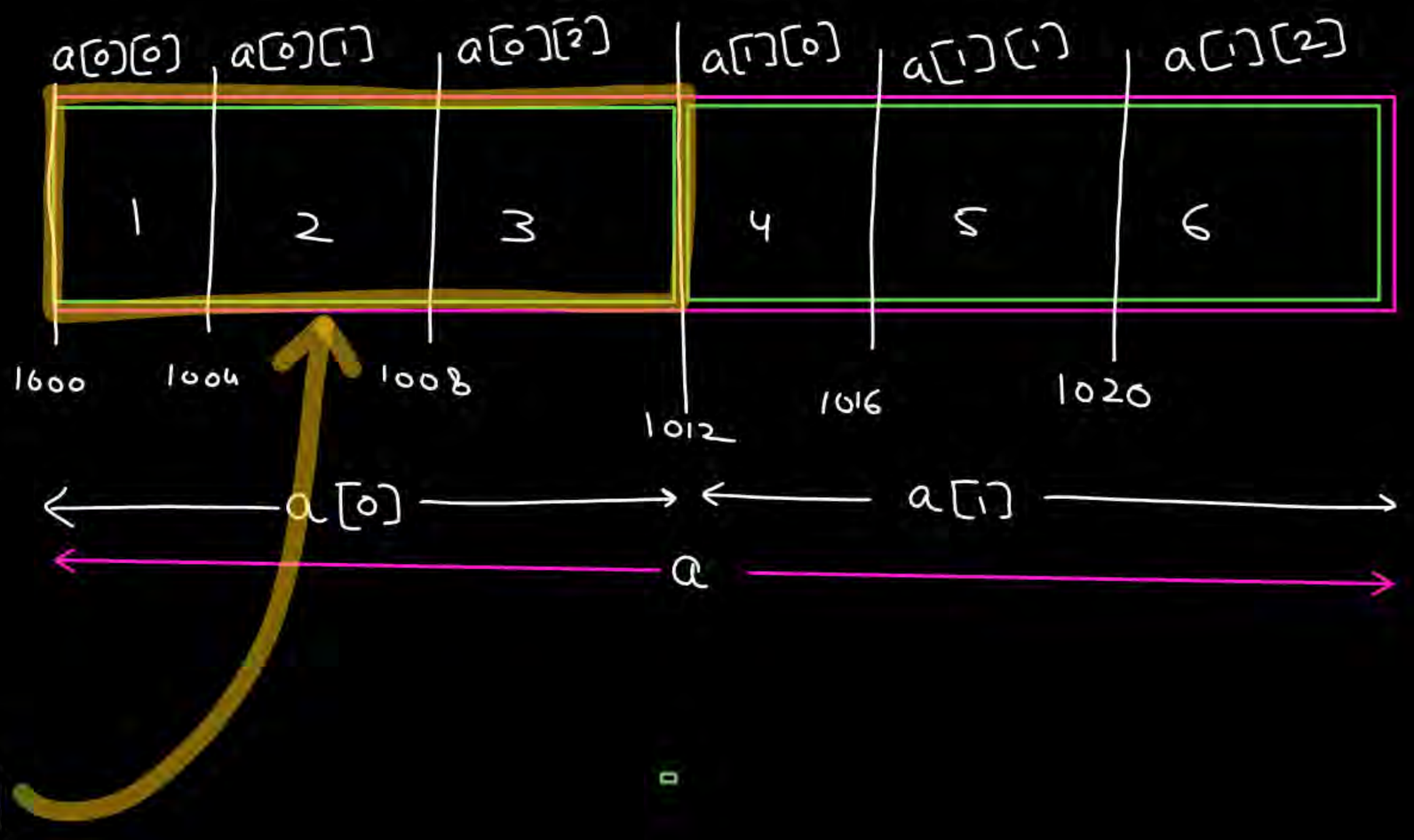
(1)

a is an array of 2 elements : a[0], a[1]

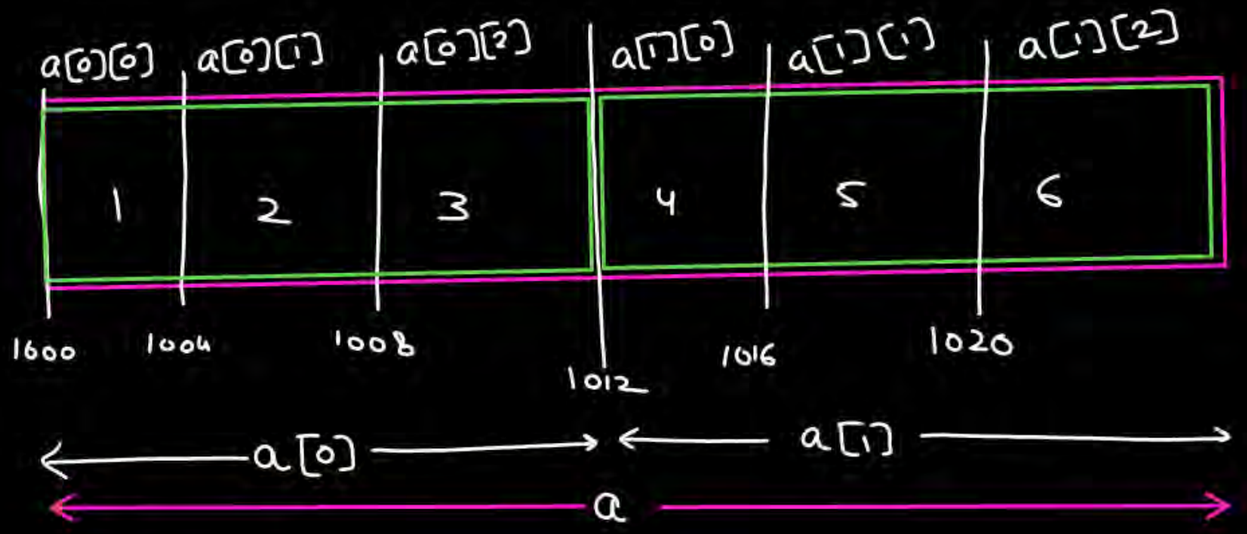
a : array-name

⇒ address of its first element.

&a[0]



```
int a[2][3] = {1,2,3,4,5,6};
```



① a is an array of 2 elements : a[0], a[1]

a : array-name

⇒ address of its first element.

&a[0]

② a[0] : an array of 3 elements a[0][0], a[0][1], a[0][2]

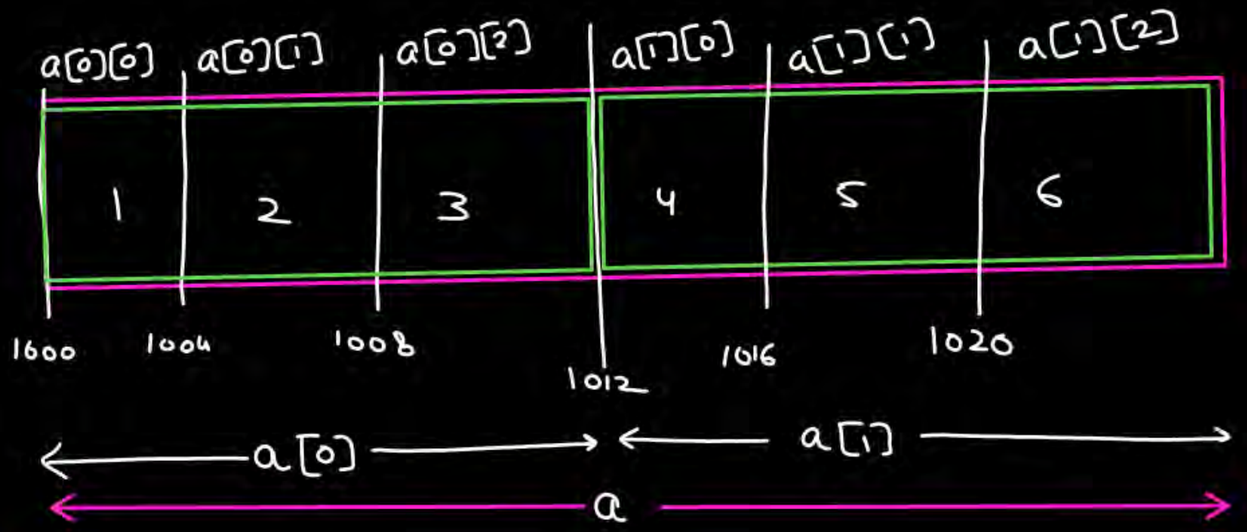
a[0] : array-name

→ address of its first element. i.e &a[0][0]

① 2 dim

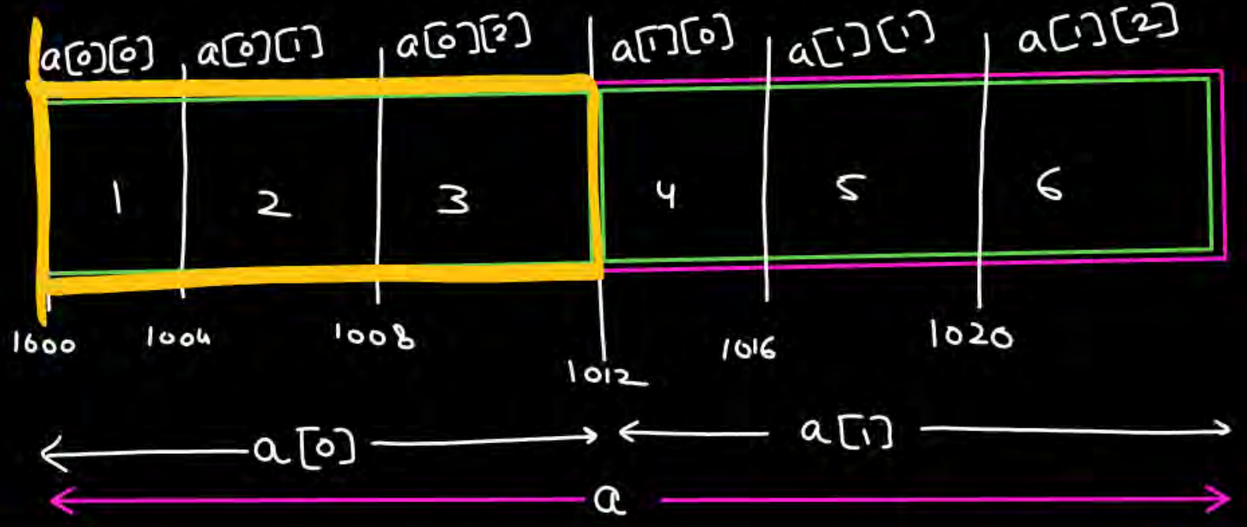
```
Void main(){

int a[2][3] = {1,2,3,4,5,6};

pf("%d", a);
pf("%d", a[0]);
pf("%d", &a);

}
```

Address



① 2 dim

```
Void main(){
int a[2][3] = {1,2,3,4,5,6};
1000 pf("%d", a);
   pf("%d", a[0]);
   pf("%d", &a);
}
```

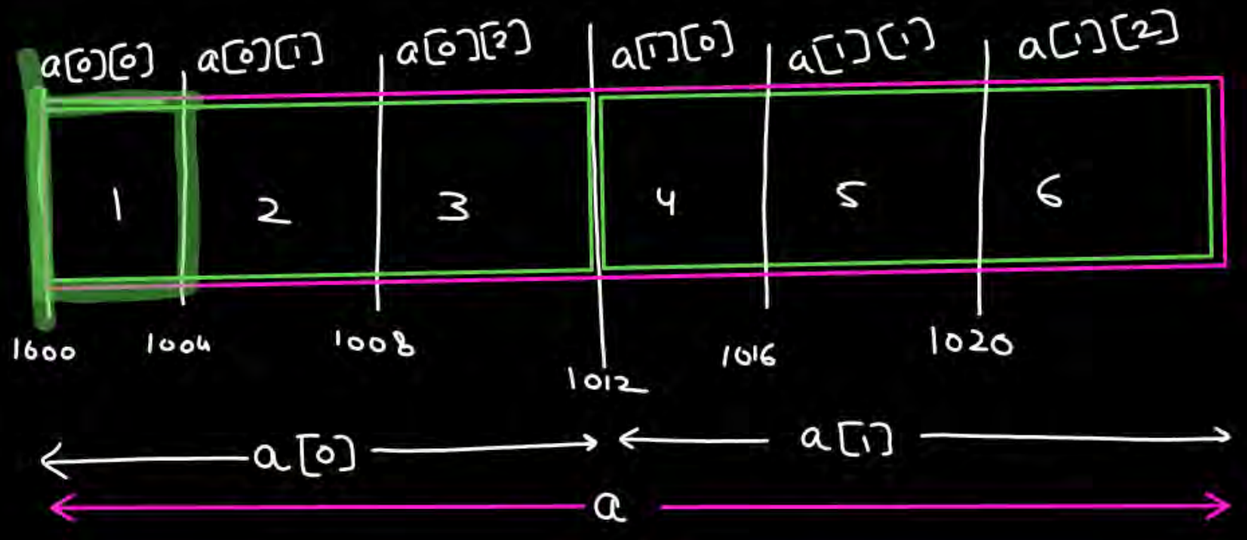


array name : add. of its 1st element ⇒ &a[0]

(12 byte)

① 2 dim

```
Void main(){
  int a[2][3] = {1,2,3,4,5,6};
1000 pf("%d", a);
1000 pf("%d", a[0]);
     pf("%d", &a);
}
```



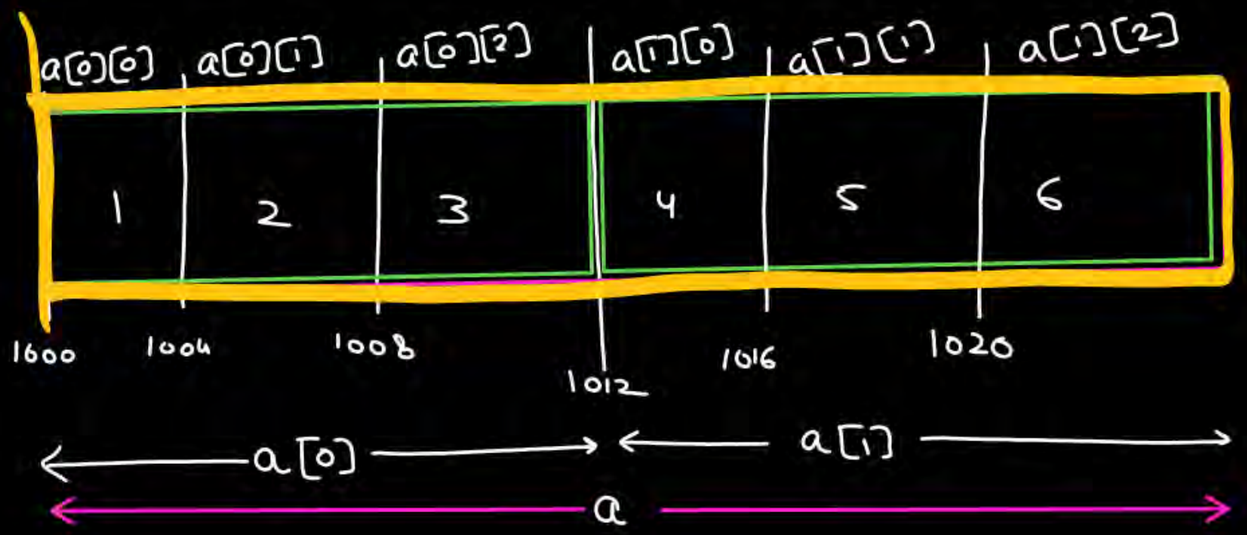
a[0] : array-name : address of its 1st element

⇒ &a[0][0] (4 byte)

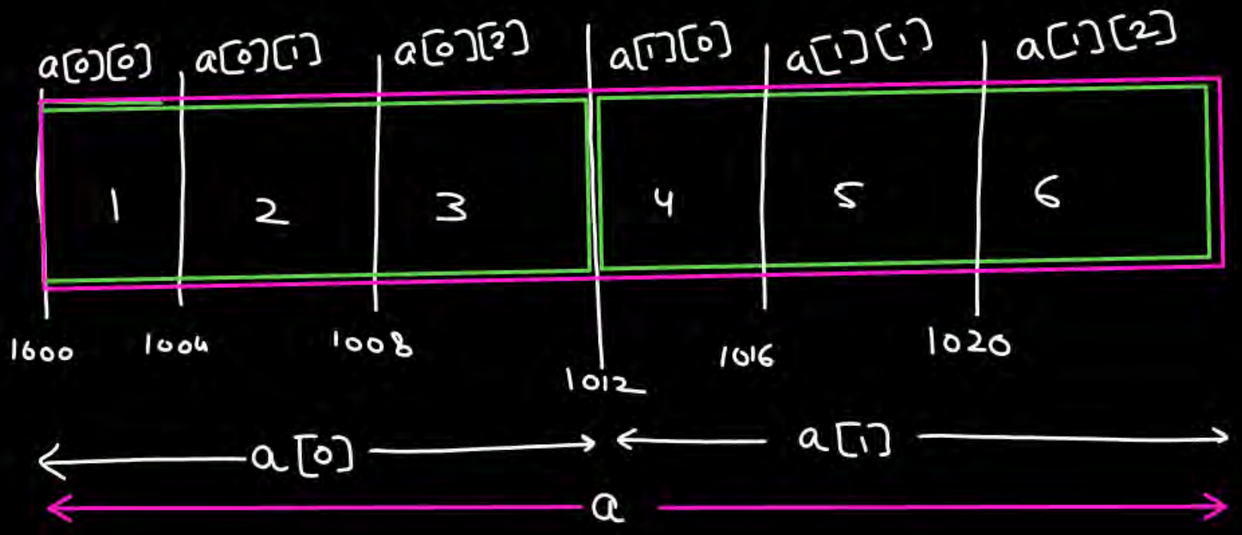
① 2 dim

```
Void main(){
  int a[2][3] = {1,2,3,4,5,6};
1000 pf("%d", a);
1000 pf("%d", a[0]);
1000 pf("%d", &a);
}
```

→ address of whole array (24 byte)



```
void main(){
 int a[2][3] = {1,2,3,4,5,6};

 pf("/d", a+1);
 pf("/d", a[0]+1);
 pf("/d", &a+1);

 }
```



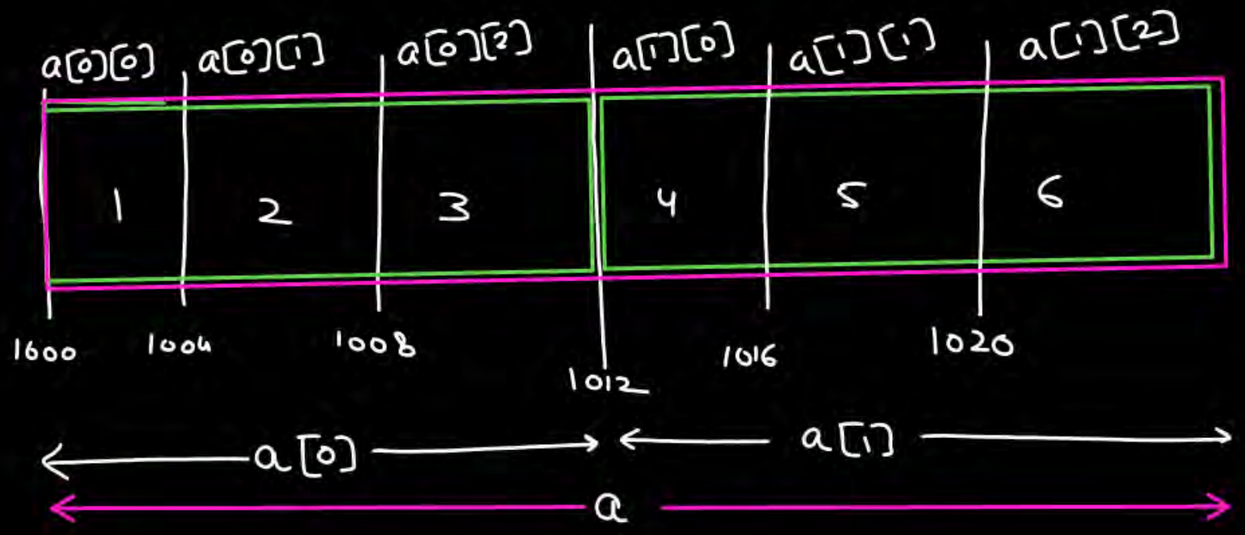
$a+1 \Rightarrow \&a[0]+1$

$\&a[0]+1 \times 12$

$= 1000 + 12$

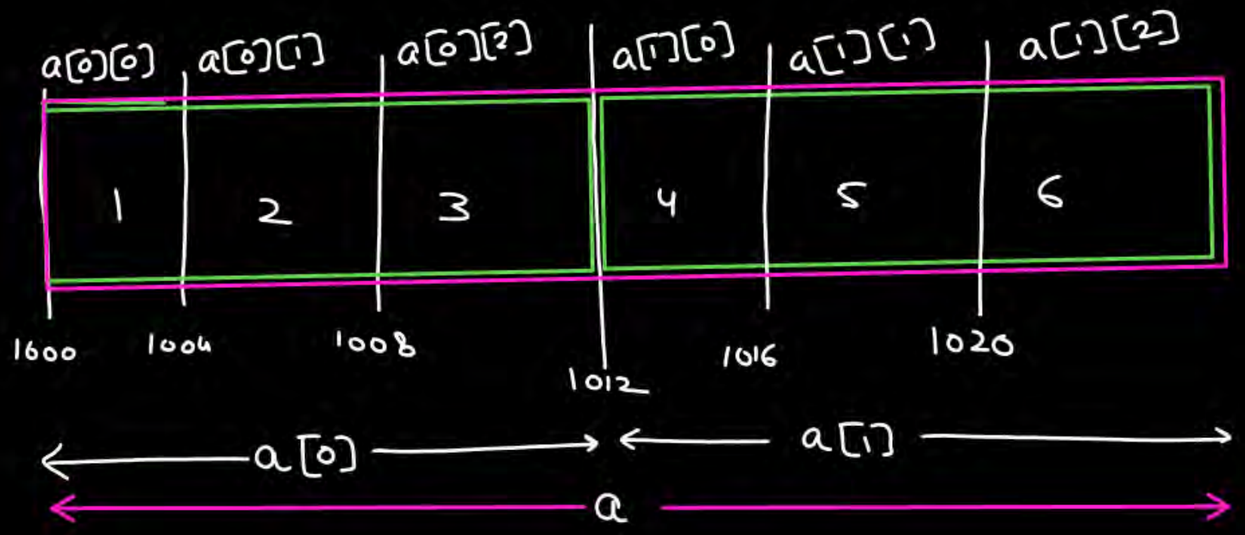
$= 1012$

```
void main(){
int a[2][3] = {1,2,3,4,5,6};
pf("%d", a+1);
pf("%d", a[0]+1);
pf("%d", &a+1);
}
```



$$a[0]+1 = \&a[0][0]+1$$
$$= \&a[0][0]+1\times 4$$
$$= 1000+4$$
$$= 1004$$

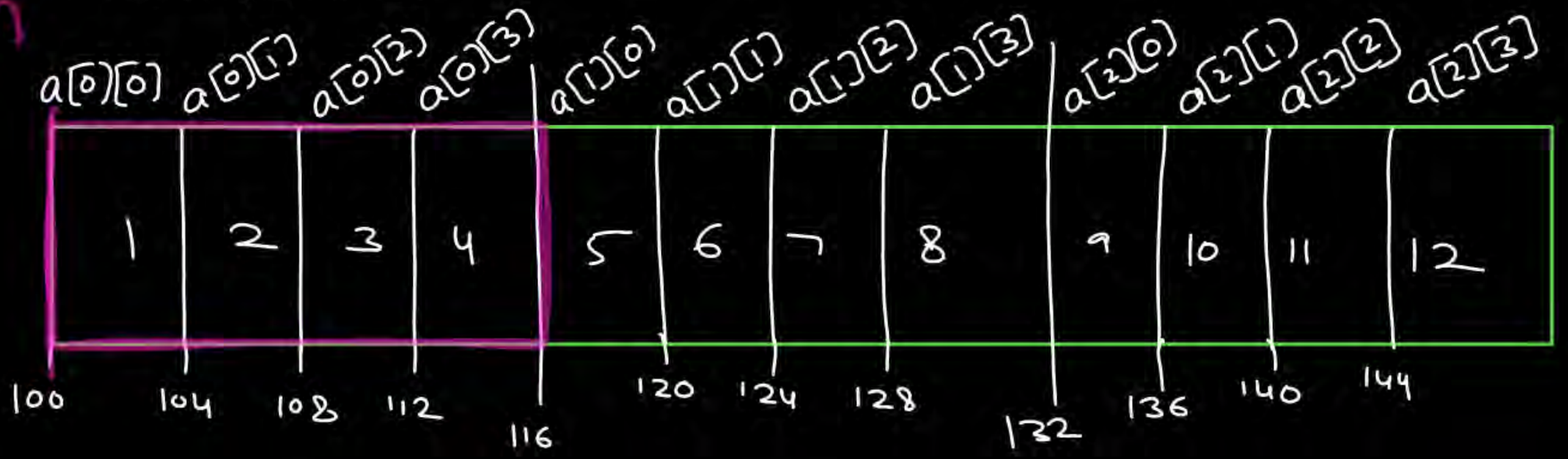
```
void main(){
 int a[2][3] = {1,2,3,4,5,6};
 pf("%d", a+1);
 pf("%d", a[0]+1);
 pf("%d", &a+1);
}
```



$$\&a + 1 = \&a + 1 \times 24$$
$$= 1000 + 24$$
$$= 1024$$

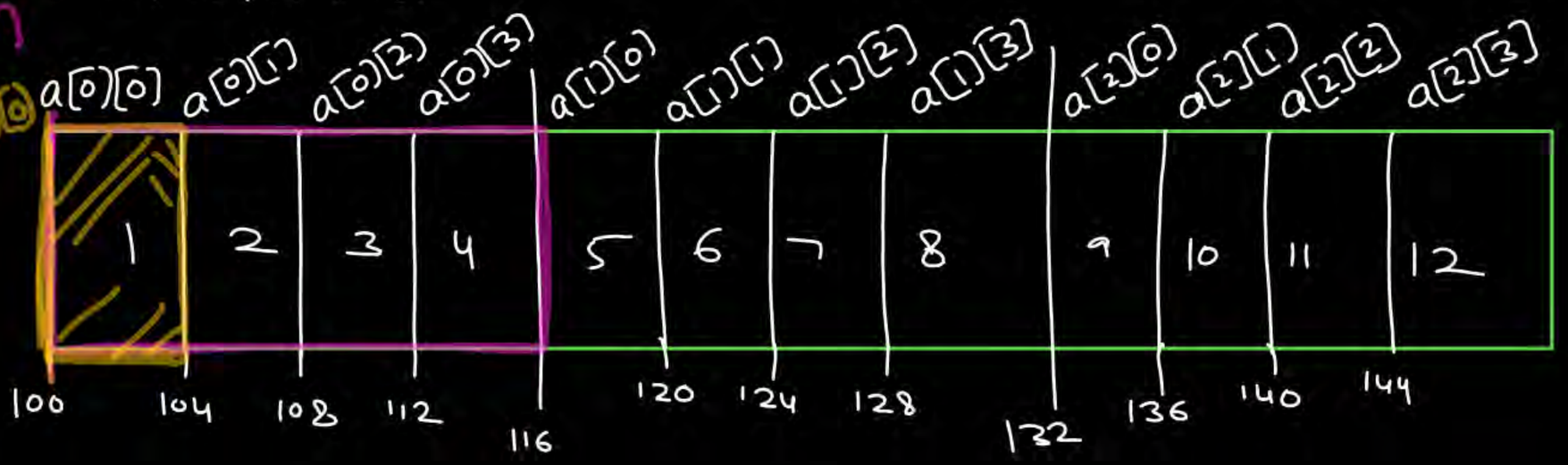
2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

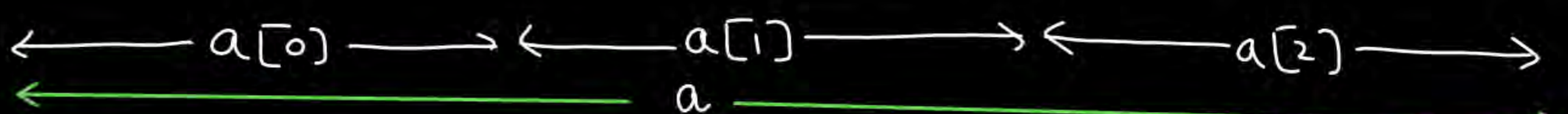
```
100 pf("%d", a);      → &a[0]
    pf("%d", a[0]);
    pf("%d", &a);
    pf("%d", a+1);
    pf("%d", a[0]+1);
    pf("%d", &a+1);
```



2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

```
100  pf("%d", a);        → &a[0]
100  pf("%d", a[0]);      → &a[0][0]
     pf("%d", &a);
     pf("%d", a+1);
     pf("%d", a[0]+1);
     pf("%d", &a+1);
```





2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

100 `pf("%d", a);` → `&a[0]`

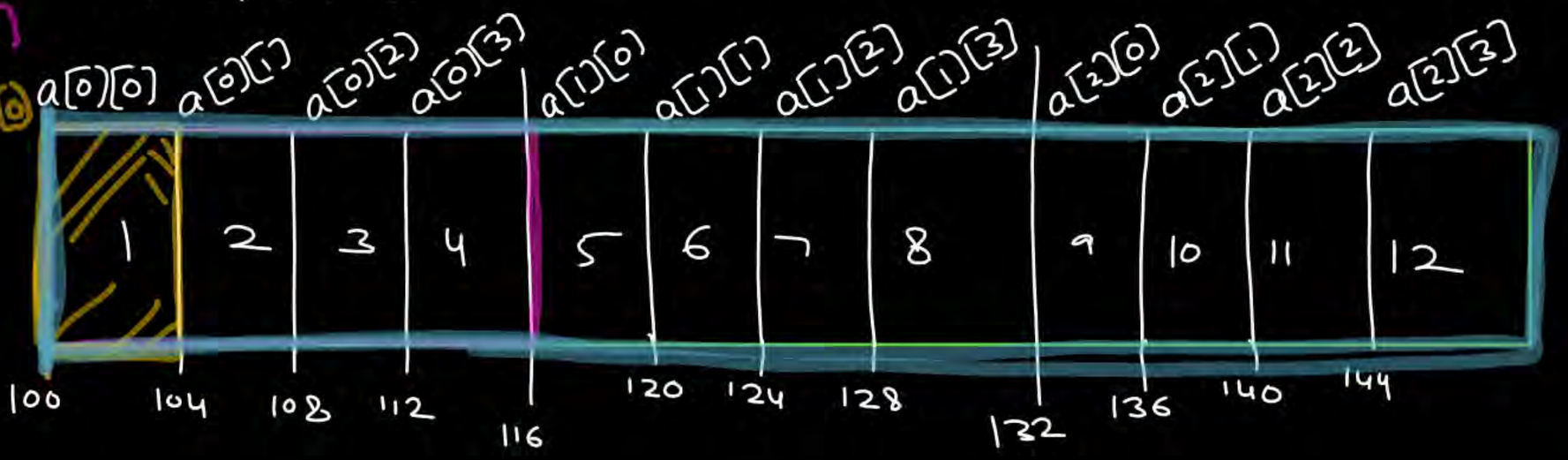
100 `pf("%d", a[0]);` → `&a[0][0]`

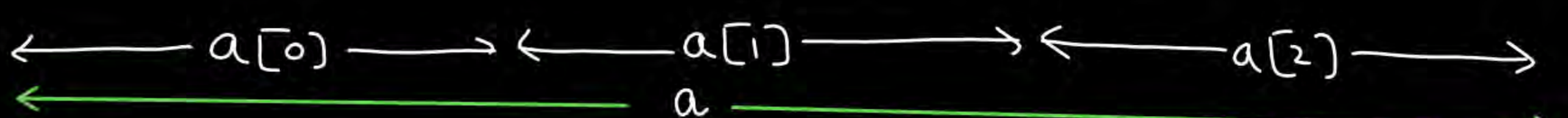
100 `pf("%d", &a);`

`pf("%d", a+1);`

`pf("%d", a[0]+1);`

`pf("%d", &a+1);`





2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

100 `pf("%d", a);` → `&a[0]`

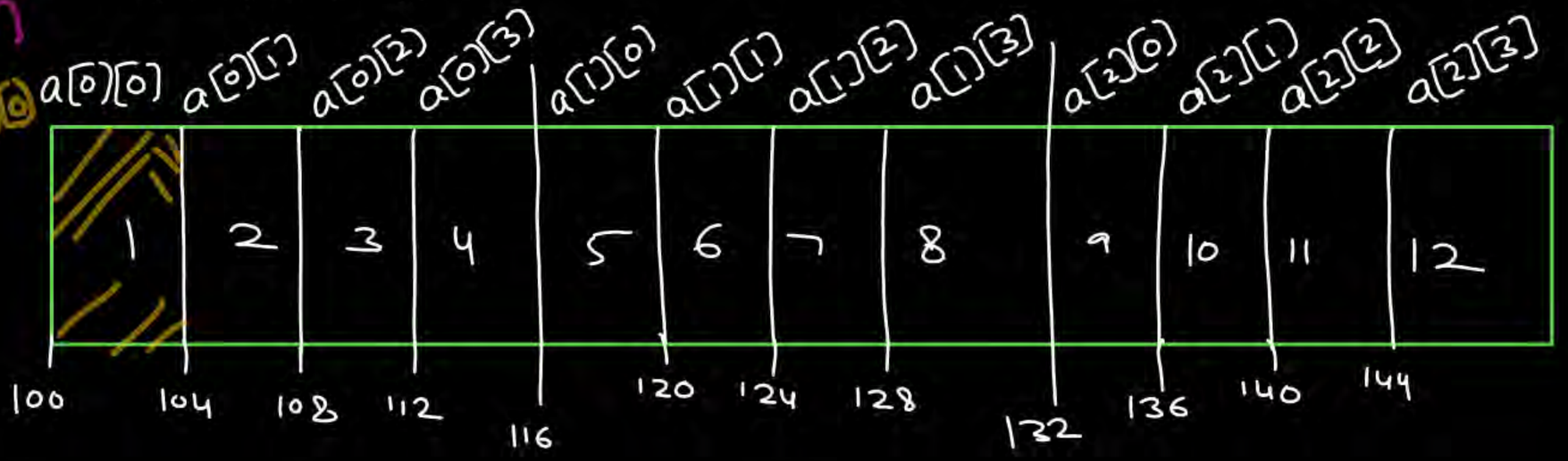
100 `pf("%d", a[0]);` → `&a[0][0]`

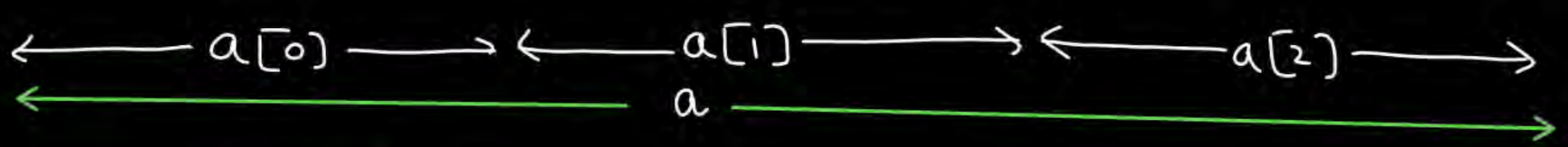
100 `pf("%d", &a);`

116 `pf("%d", a+1);`

`pf("%d", a[0]+1);`

`pf("%d", &a+1);`





$a+1 = \&a[0] + 1$

$= \&a[0] + 1 \times 16$

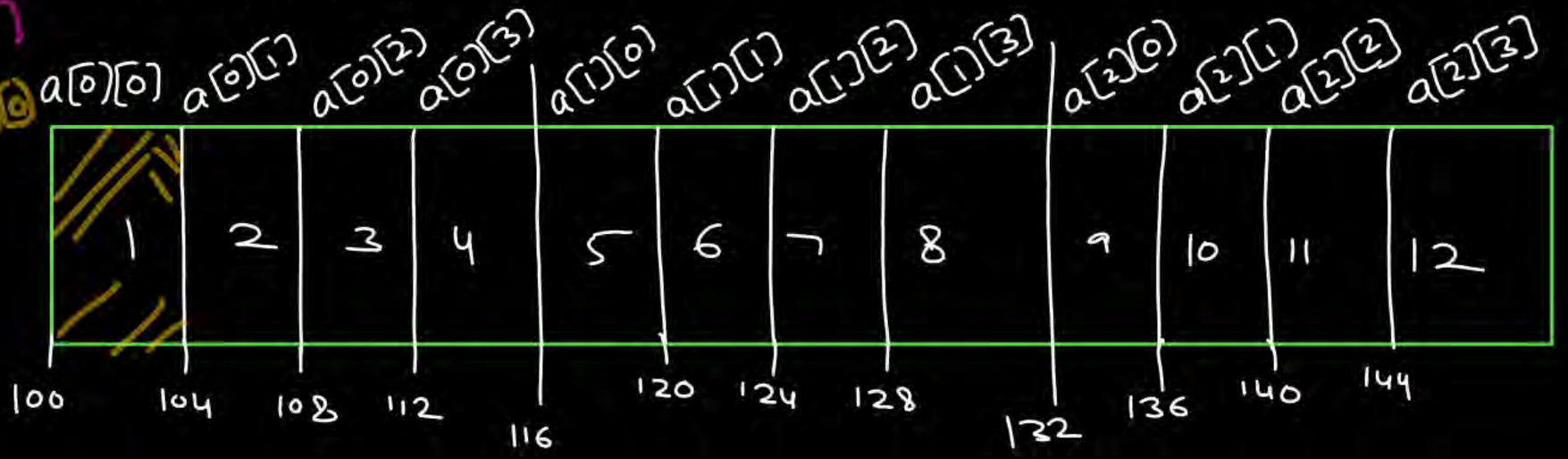
$= 100 + 16$

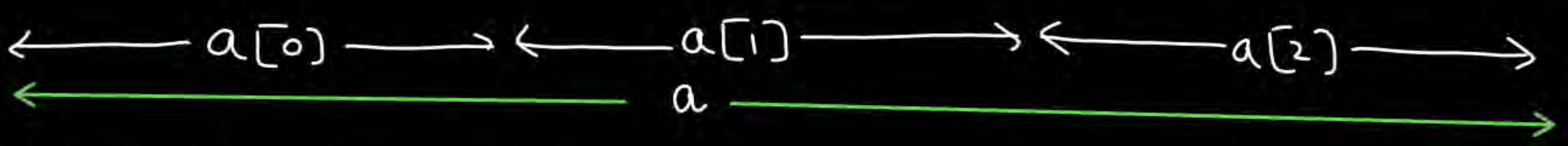
$= 116$

sizeof a[0] = 16 byte

2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

```
100  pf("%d", a);      → &a[0]
100  pf("%d", a[0]);   → &a[0][0]
100  pf("%d", &a);
116  pf("%d", a+1);
104  pf("%d", a[0]+1);
     pf("%d", &a+1);
```





$a[0] \Rightarrow \&a[0][0]$

$a[0]+1 = \&a[0][0]+1$

$= \&a[0][0]+1\times 4$

$= 100+4 = 104$

sizeof a[0][0] = 4 byte

2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

100 `pf("%d", a);` → `&a[0]`

100 `pf("%d", a[0]);` → `&a[0][0]`

100 `pf("%d", &a);`

116 `pf("%d", a+1);`

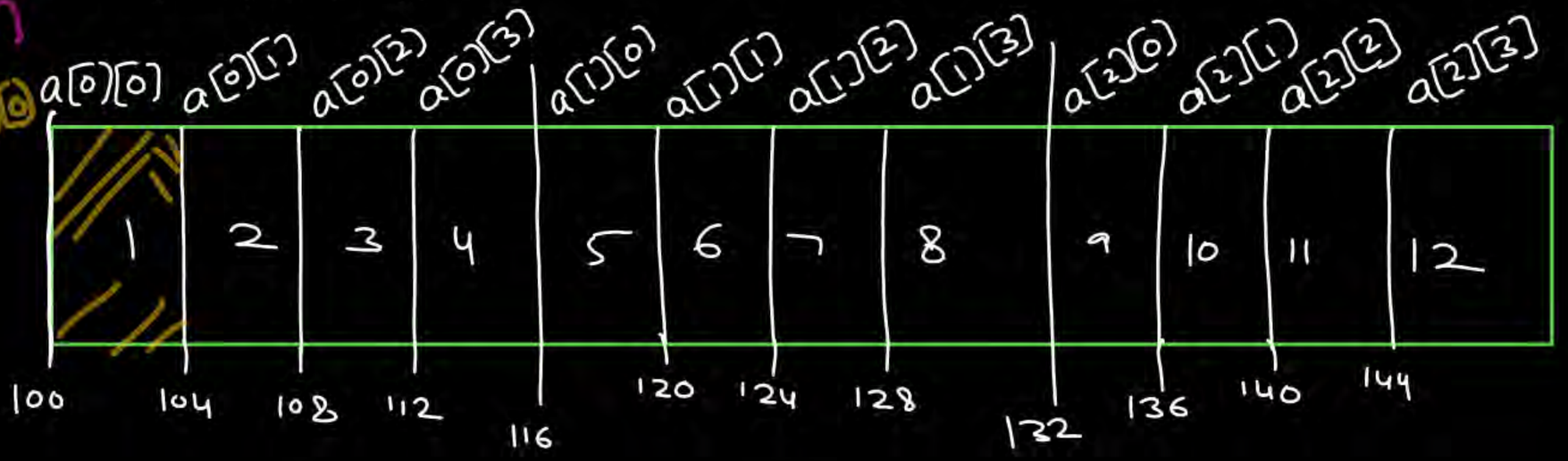
104 `pf("%d", a[0]+1);`

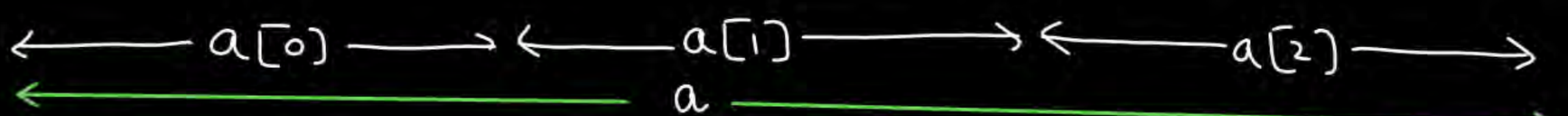
`pf("%d", &a+1);`

↳ `&a+1`

= `&a+1`

= 100 + 1×48 = 148





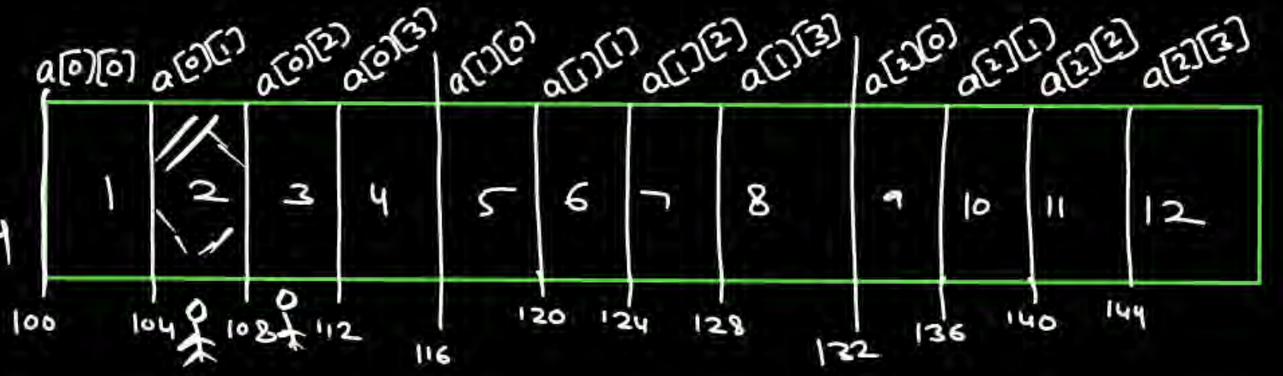
size of complete array a
= 48 byte

2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

$$a[0]+1 = \underbrace{\&a[0][0]}_{4\text{byte}} + 1$$

$$= \&a[0][0] + 1 \times 4$$

$$= 100 + 4$$



$a[0]+1 = \begin{pmatrix}\text{Memory}\\ \text{location}\\ 104\end{pmatrix} = \&a[0][1]$

$$a[0]+2 = \underbrace{\&a[0][0]}_{4\text{byte}} + 2$$

$$= \&a[0][0] + 2 \times 4$$

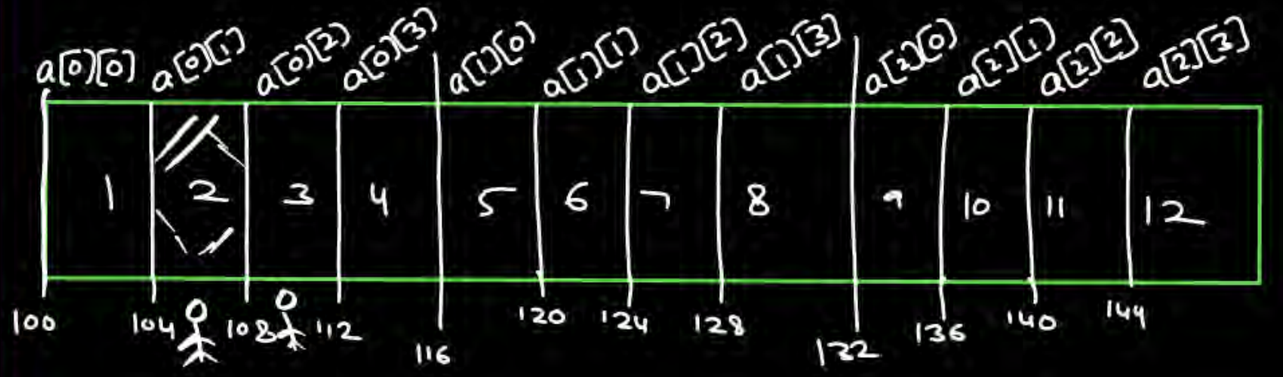
$$= 100 + 8 = 108$$

$a[0]+2 = \begin{pmatrix}\text{Memory}\\ \text{location}\\ 108\end{pmatrix} = \&a[0][2]$

2. int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};

$a[0]+1 = \text{Mem. loc} = 104 = \&a[0][1]$

$a[0]+2 = \text{Mem. loc.} = 108 = \&a[0][2]$



$a[1]+1 = \&a[1][0]+1$ (4 byte)

$= \&a[1][0] + 1 \times 4$

$= 116 + 4 = 120$

$a[1]+1 = \text{Mem. loc. } 120 = \&a[1][1]$

$a[1]+2 = \&a[1][0]+2$ (4 byte)

$= \&a[1][0] + 2 \times 4$

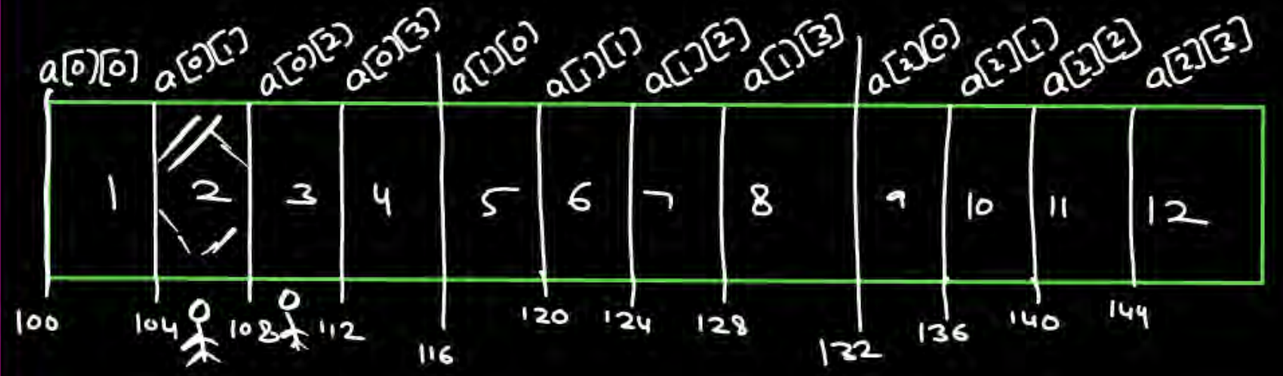
$= 124$

$a[1]+2 = \text{Mem. loc } 124 = \&a[1][2]$

2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

$a[0]+1 = \text{Mem. loc} = 104 = \&a[0][1]$

$a[0]+2 = \text{Mem. loc.} = 108 = \&a[0][2]$



$a[1]+1 = \text{Mem. loc. } 120 = \&a[1][1]$

$a[1]+2 = \text{Mem. loc } 124 = \&a[1][2]$

$*(a[0]+1) = \text{value at}(\text{Mem. loc. } 104) = *\&a[0][1]$

① — $*(a[0]+1) = a[0][1]$

$*(a[0]+2) = \text{value at}(\text{Mem. loc. } 108) = *\&a[0][2]$

② — $*(a[0]+2) = a[0][2]$

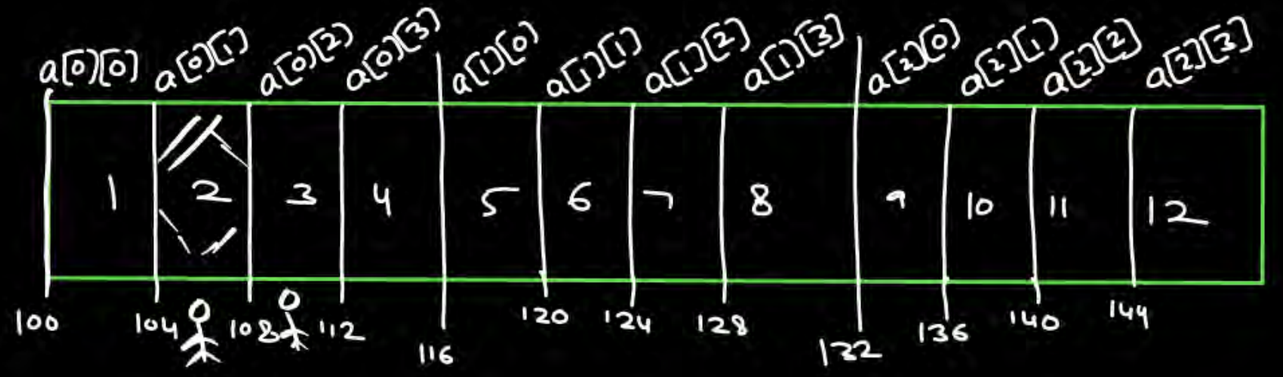
2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

① — `*(a[0]+1) = a[0][1]`

② — `*(a[0]+2) = a[0][2]`

③ — `*(a[1]+1) = a[1][1]`

④ — `*(a[1]+2) = a[1][2]`



`a[1]+1` = Mem. loc. 120 = `&a[1][1]`

`*(a[1]+1)` = value at (Mem loc. 120) = `*&a[1][1]`

`*(a[1]+1) = a[1][1]`

`a[1]+2` = Mem. loc 124 = `&a[1][2]`

`*(a[1]+2)` = value at (Mem loc 124) = `*&a[1][2]`

`*(a[1]+2) = a[1][2]`

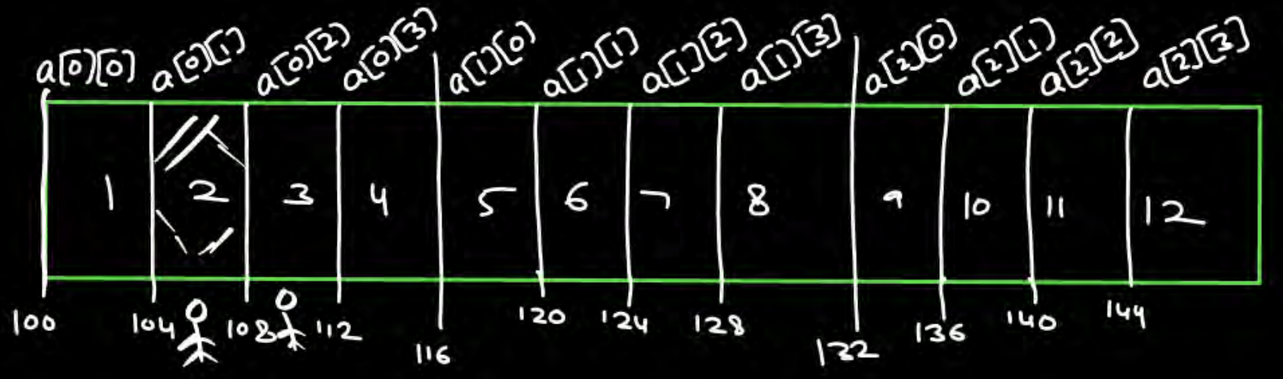
2. `int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};`

① — $*(a[0]+1) = a[0][1]$

② — $*(a[0]+2) = a[0][2]$

③ — $*(a[1]+1) = a[1][1]$

④ — $*(a[1]+2) = a[1][2]$



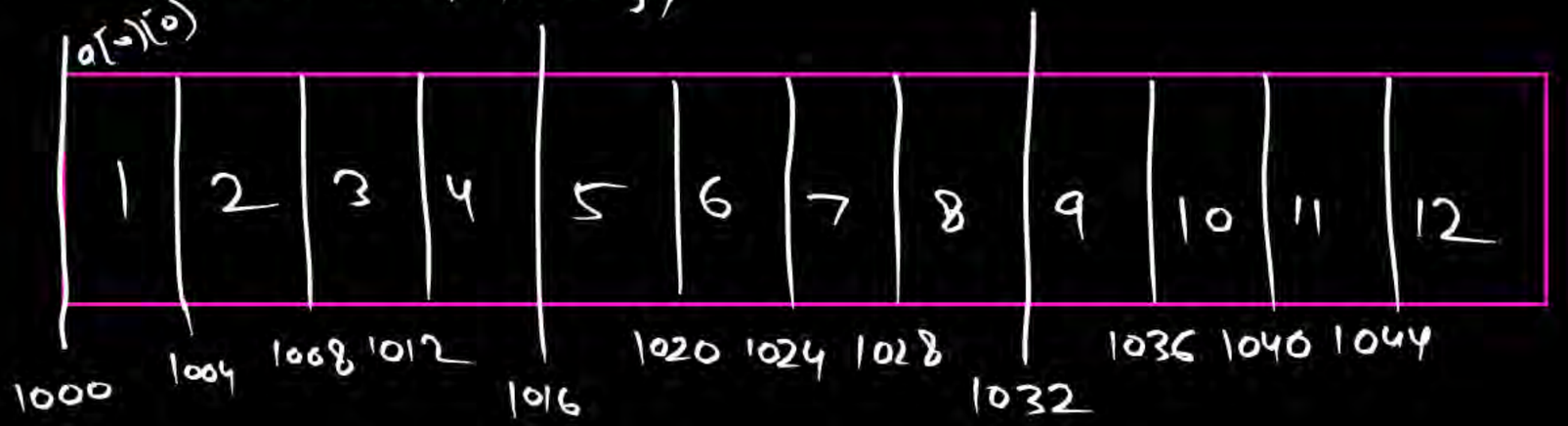
$$*(a[0]+j) = a[0][j]$$

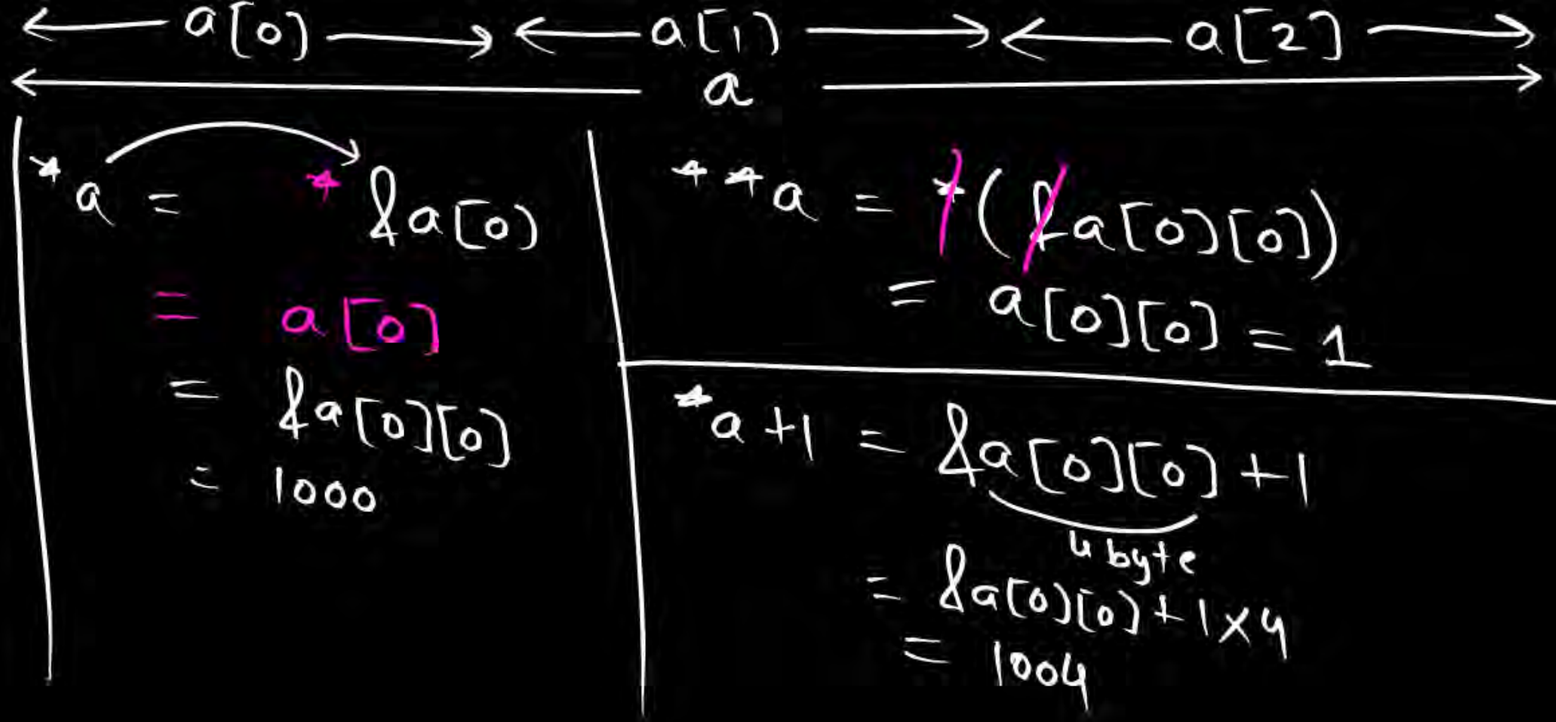
$$*(a[1]+j) = a[1][j]$$

$$*(a[i]+j) = a[i][j]$$

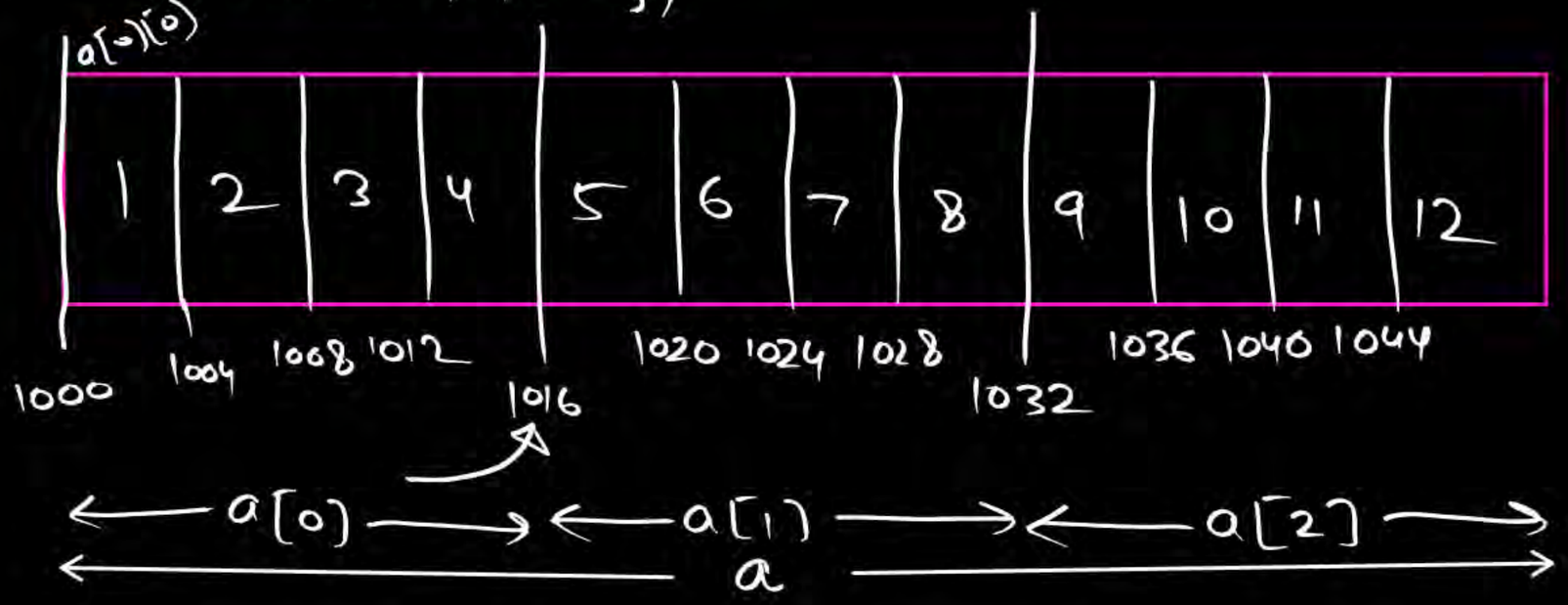
$$\Rightarrow \begin{bmatrix} *(a[i]+j) = a[i][j] \\ *(*(a+i)+j) = a[i][j] \end{bmatrix}$$

```
int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};

1000 pf("%d",a);
1000 pf("%d",a[0]);
1000 pf("%d",&a);
1000 pf("%d",*a);
   1 pf("%d",**a);
1004 pf("%d",*a+1);
     pf("%d",**a+1);
     pf("%d",*a[0]);
```





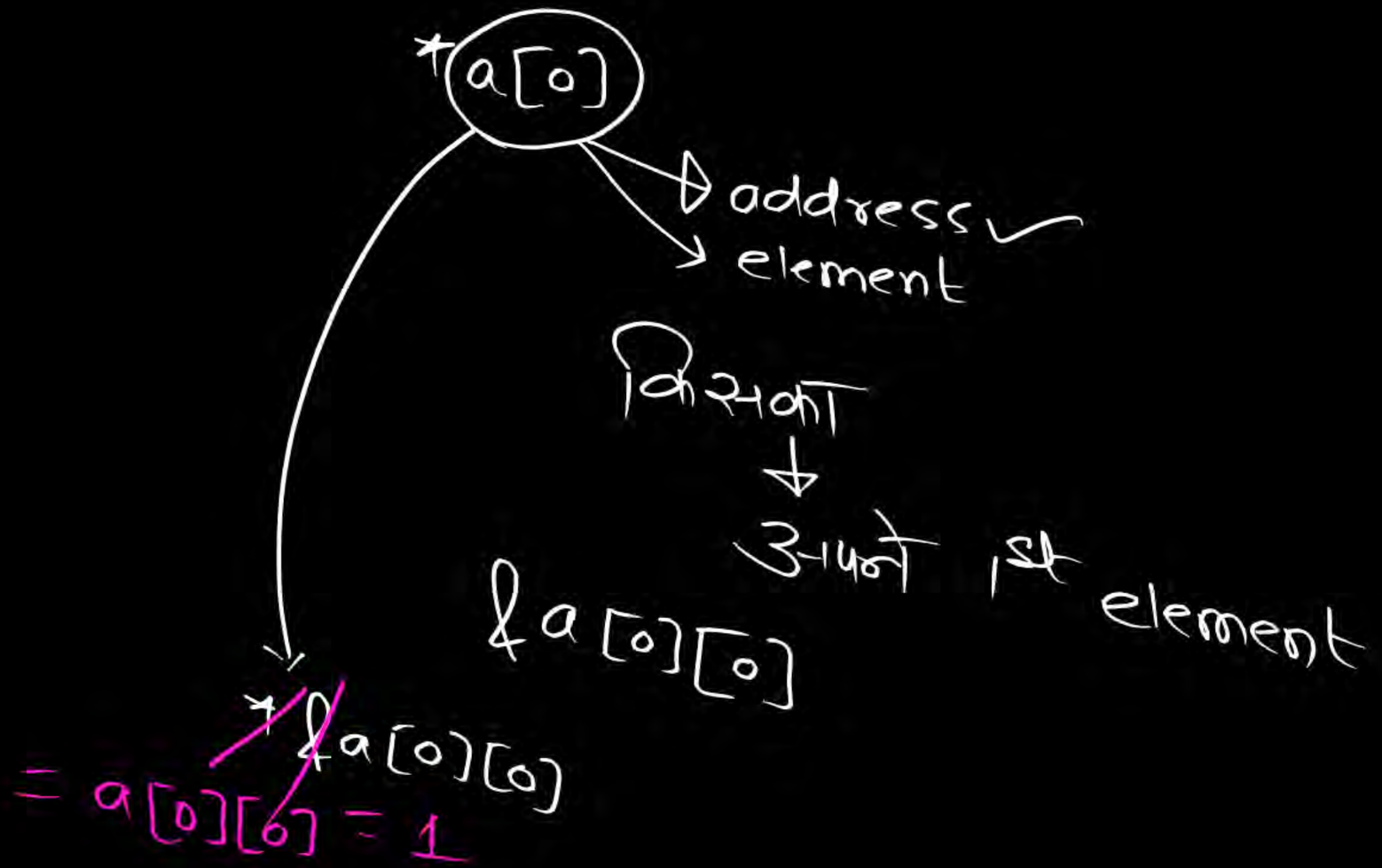
```
int a[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};

1000 pf("%d",a);
1000 pf("%d",a[0]);
1000 pf("%d",&a);
1000 pf("%d",*a);
   1 pf("%d",**a);
1004 pf("%d",*a+1);
   2 pf("%d",**a+1);
   1 pf("%d",*a[0]);
```



1+1 = 2

*a[0] = *&a[0][0]
= a[0][0]
= 1

a[2][3]



a[2][3]

**a + 1

a = Add/element (✓ / ✗)

↓

1st element

a = &a[0]

*a = *(&a[0])

*a = a[0]

*a = &a[0][0]

*(*a) = *(&a[0][0])

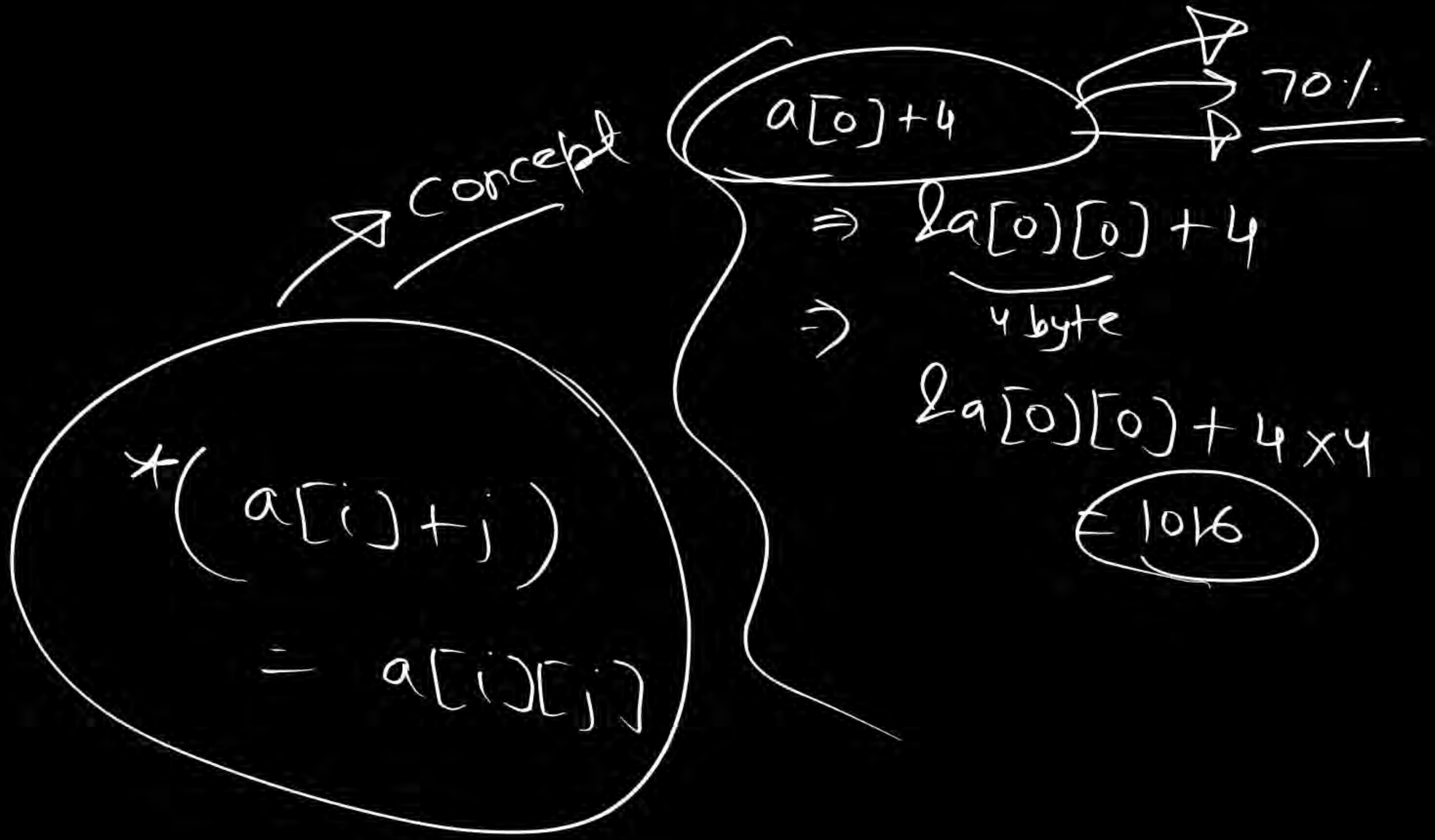
= *&a[0][0]

**a = a[0][0]

**a + 1 = a[0][0] + 1

= 1 + 1

= 2


Concept
a[0]+4
70%
⇒ &a[0][0]+4
4 byte
⇒ &a[0][0]+4×4
1016
*(a[i]+j)
= a[i][j]


THANK YOU GW
SOLDIERS !