

# CS & IT ENGINEERING



**C Programming**  
**Arrays and Pointers**  
**Lec - 03**



By- Pankaj Sharma Sir





TOPICS TO  
BE  
COVERED



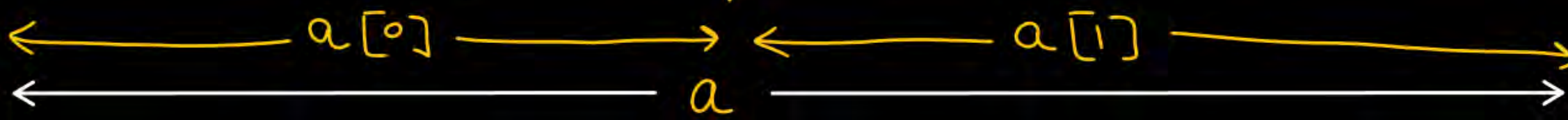
**Arrays and Pointers (Part- 03)**

2-D array

first element of `int a[2][3];`  
`a[0]`

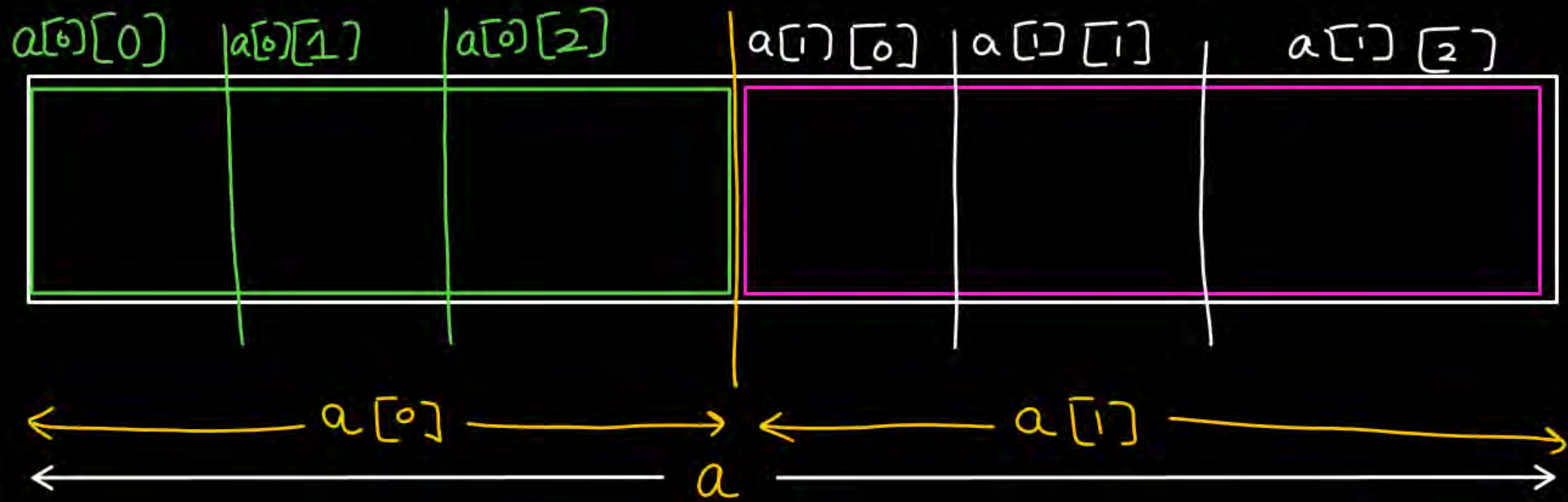
2nd  
element of `a[0]`

3rd element of `a[0]`

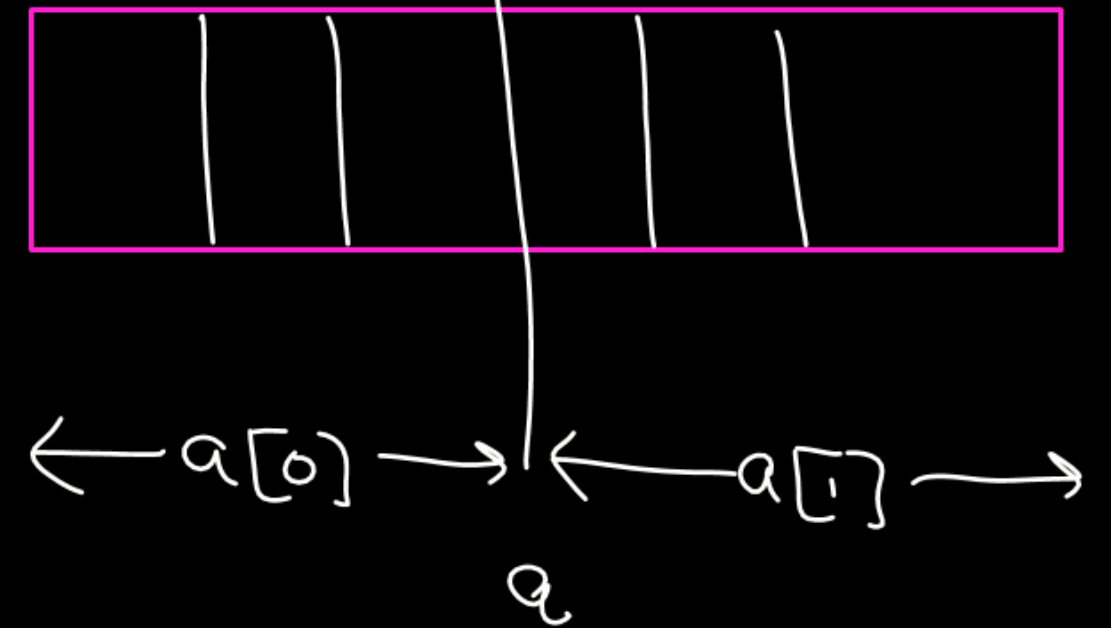
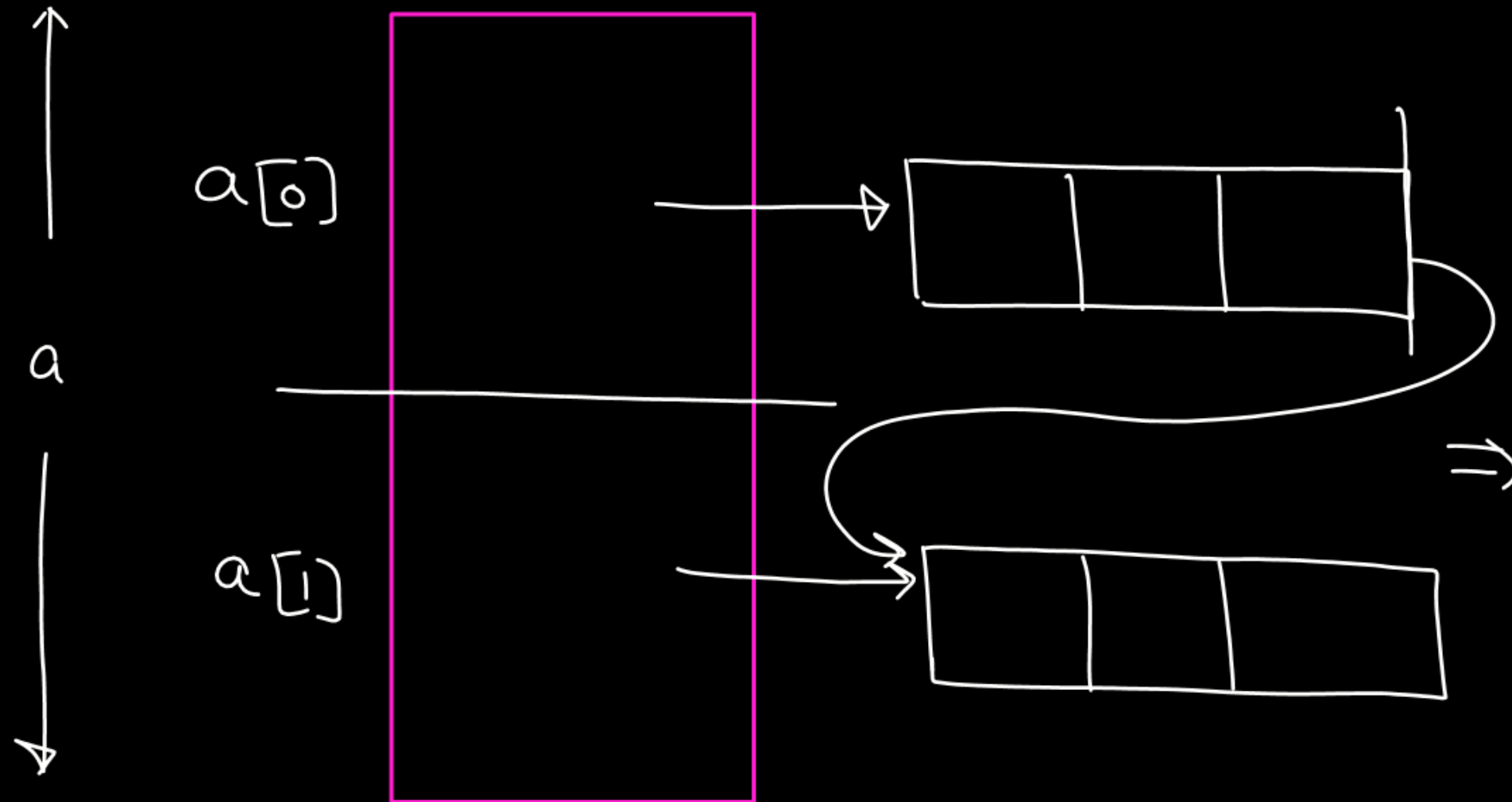
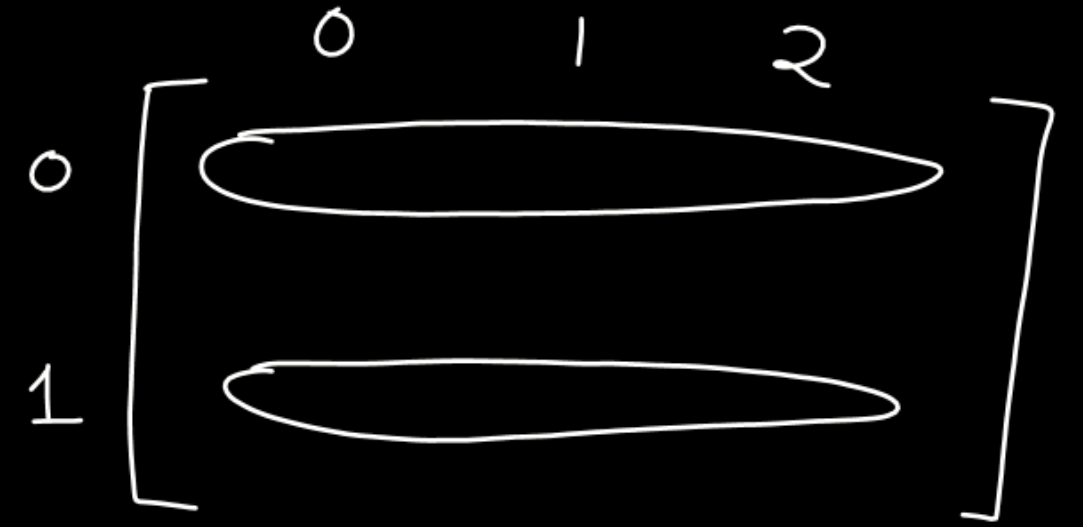


2-D array

int a[2][3];



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





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

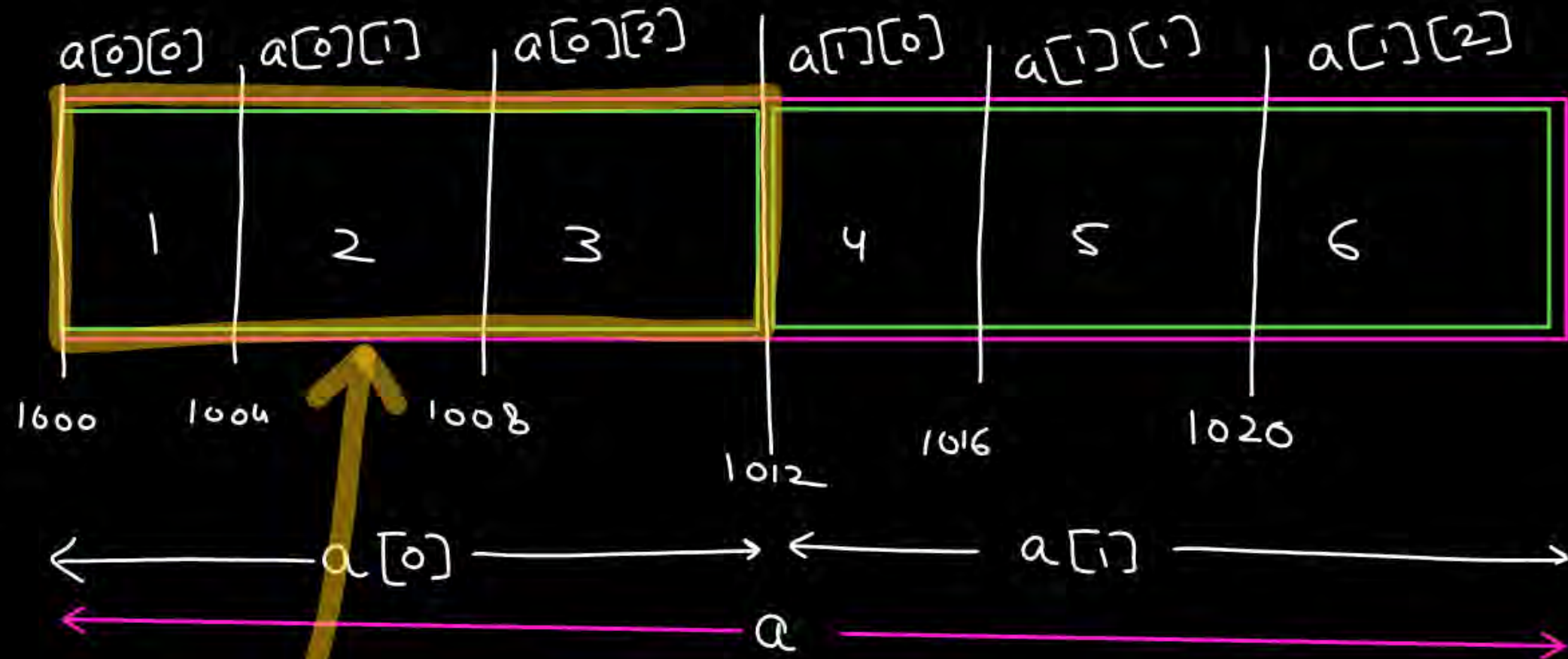
①

`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};`

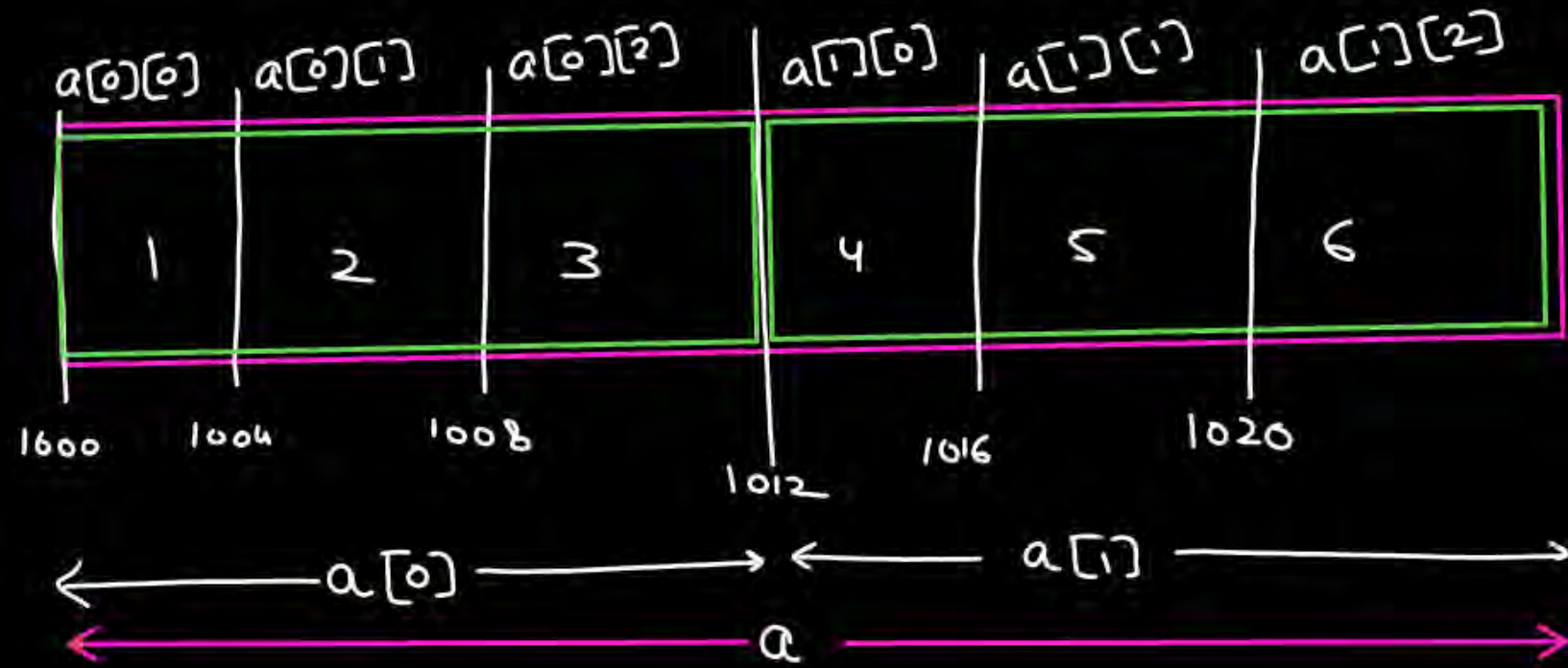
①

`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]`

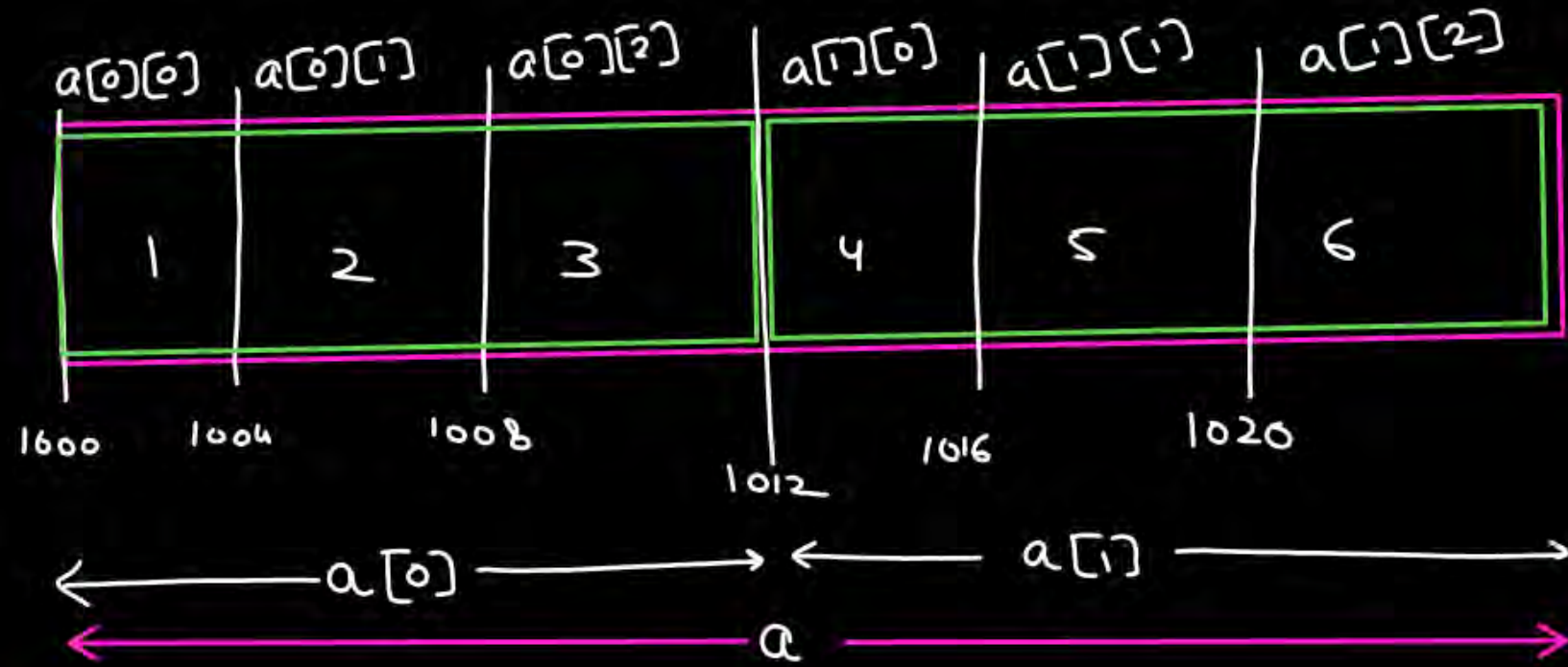
`a[0]` : array-name

→ 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





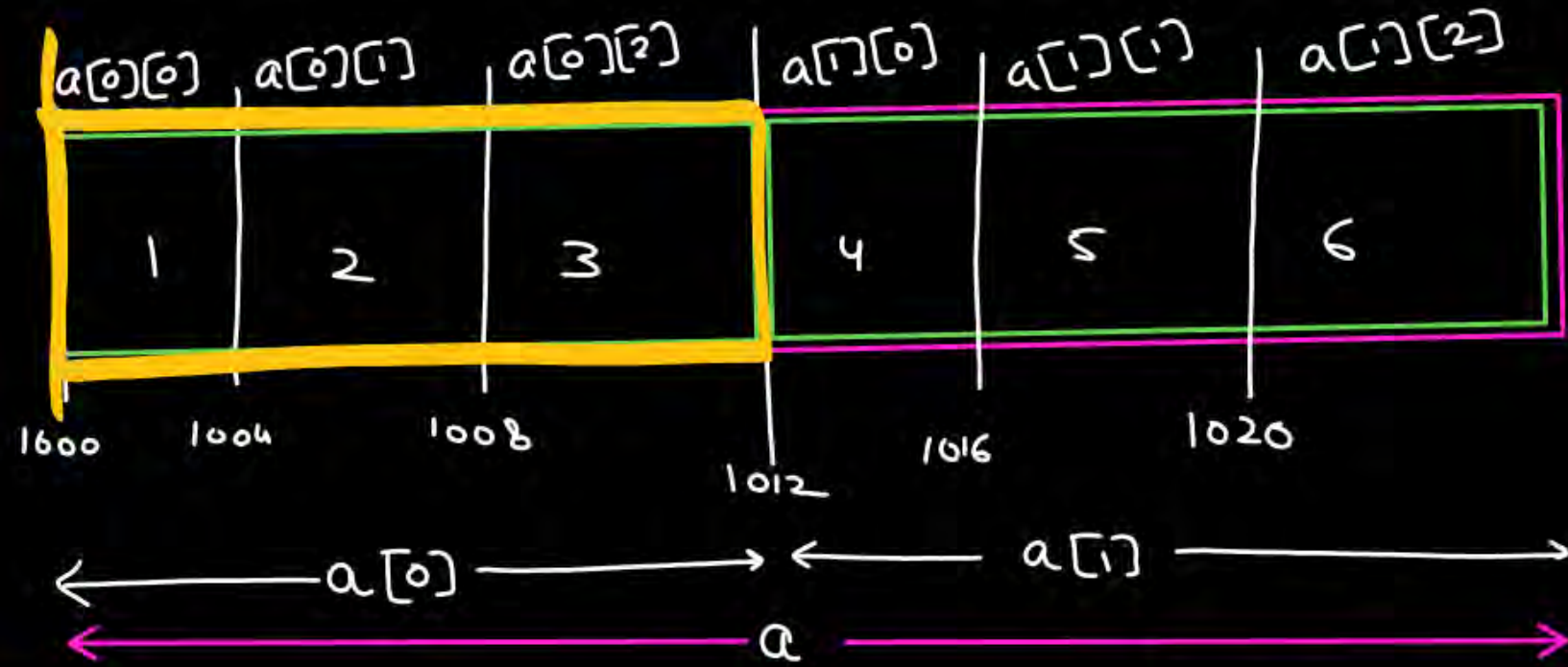
① 2 dim

```
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  $\Rightarrow$   $\&a[0]$  (12 byte)

① 2 dim

```
void main() {
```

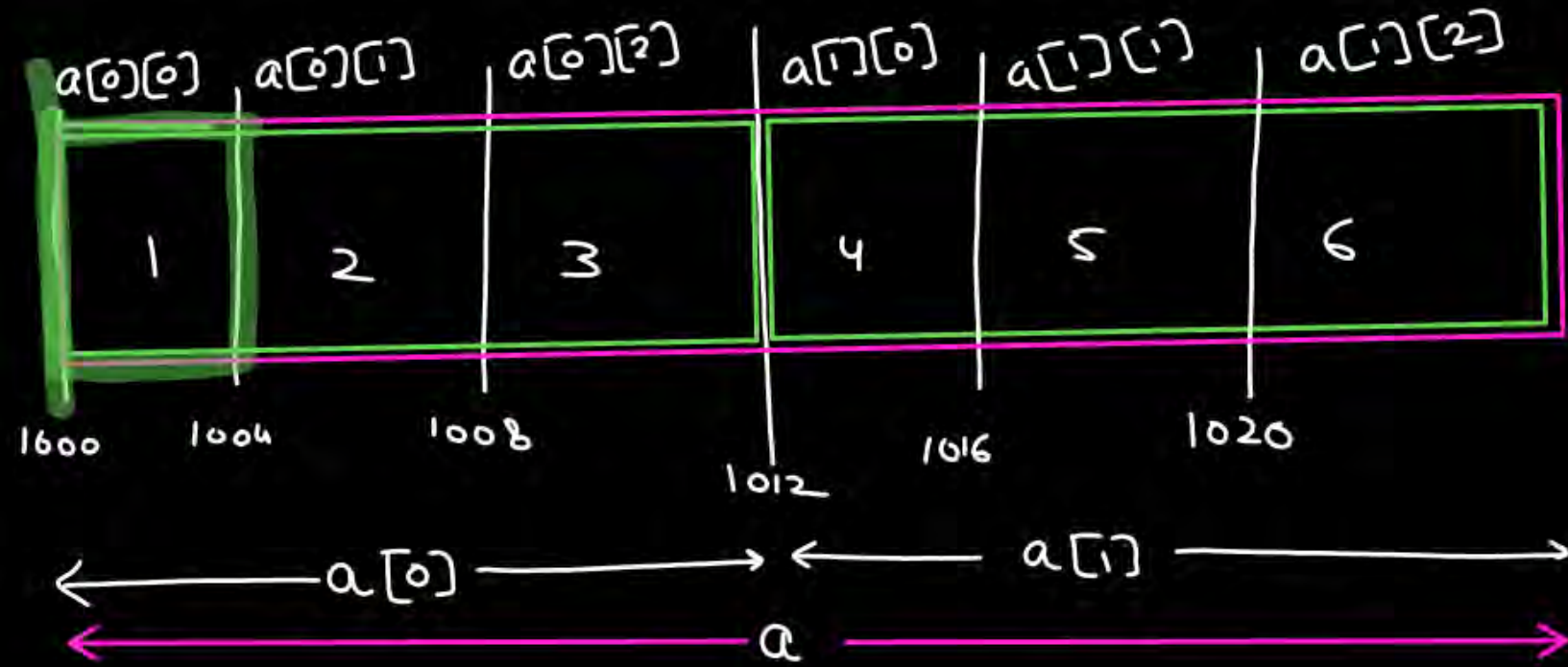
```
int a[2][3] = {1, 2, 3, 4, 5, 6};
```

```
1000 printf("/d", a);
```

```
1000 printf("/d", a[0]);
```

```
printf("/d", &a);
```

```
}
```



$\rightarrow a[0]$  : array-name : address of its 1<sup>st</sup> element  
 $\Rightarrow \&a[0][0]$  (4 byte)

① 2 dim

```
void main() {
```

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

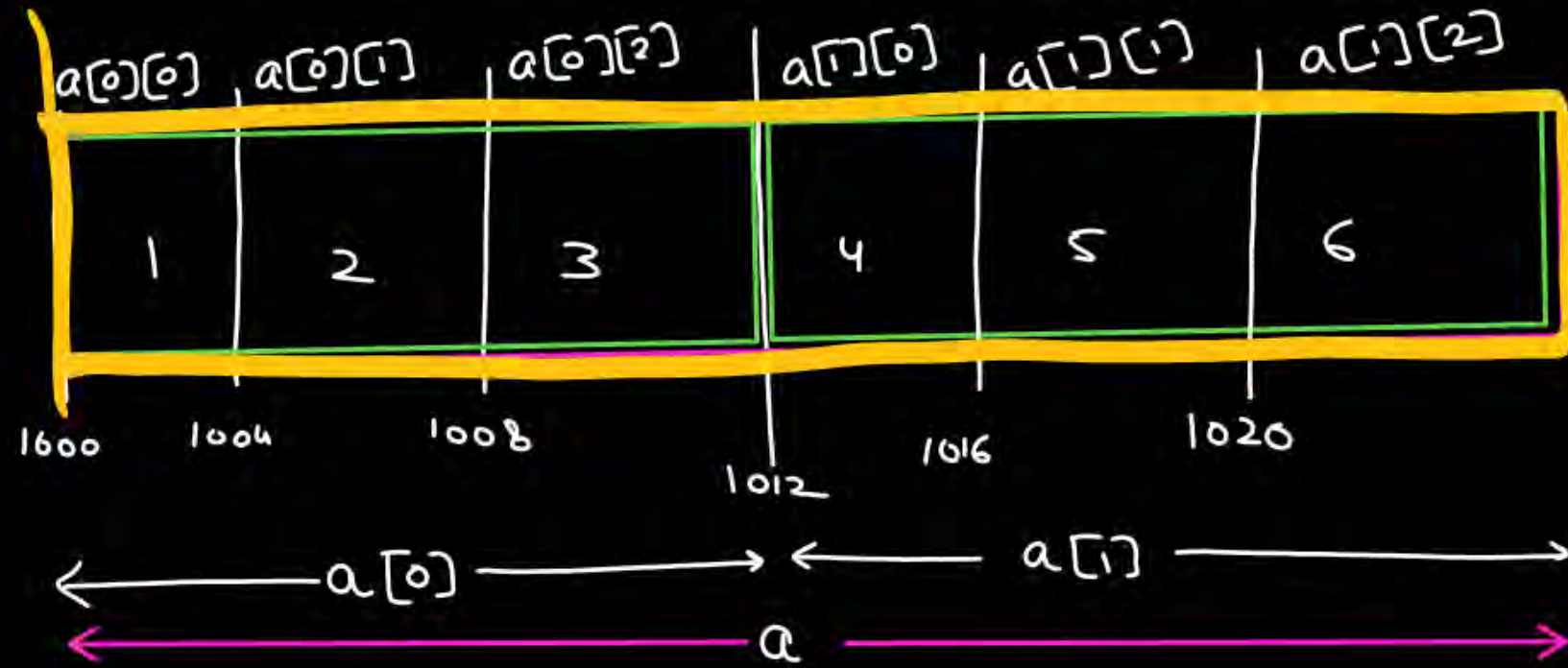
```
1000 printf("/d", a);
```

```
1000 printf("/d", a[0]);
```

```
1000 printf("/d", &a);
```

```
}
```

address of whole array (24 byte)





```
void main(){
```

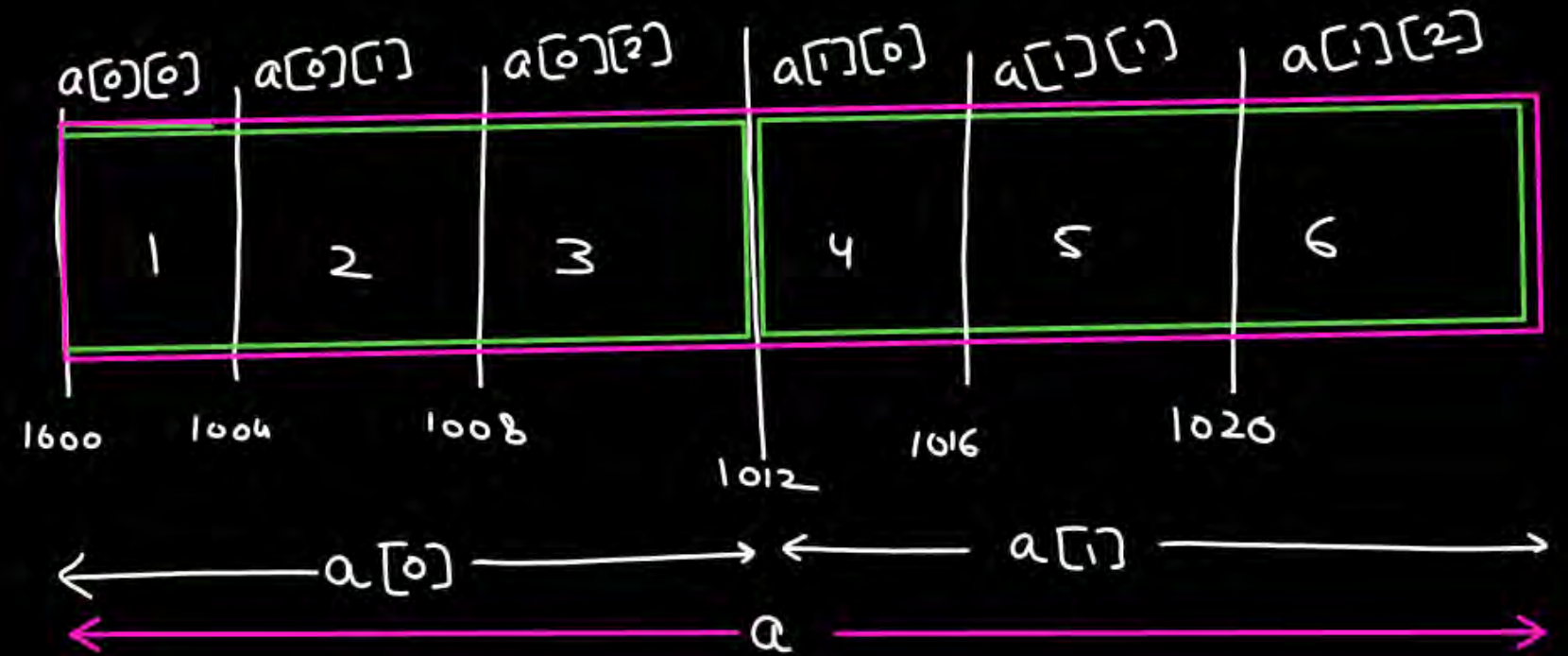
```
int a[2][3] = {1, 2, 3, 4, 5, 6};
```

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

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

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

```
}
```



$$\begin{aligned} a+1 &\Rightarrow \&a[0]+1 \\ &\&a[0]+1 \times 12 \\ &= 1000 + 12 \\ &= 1012 \end{aligned}$$

```
void main(){
```

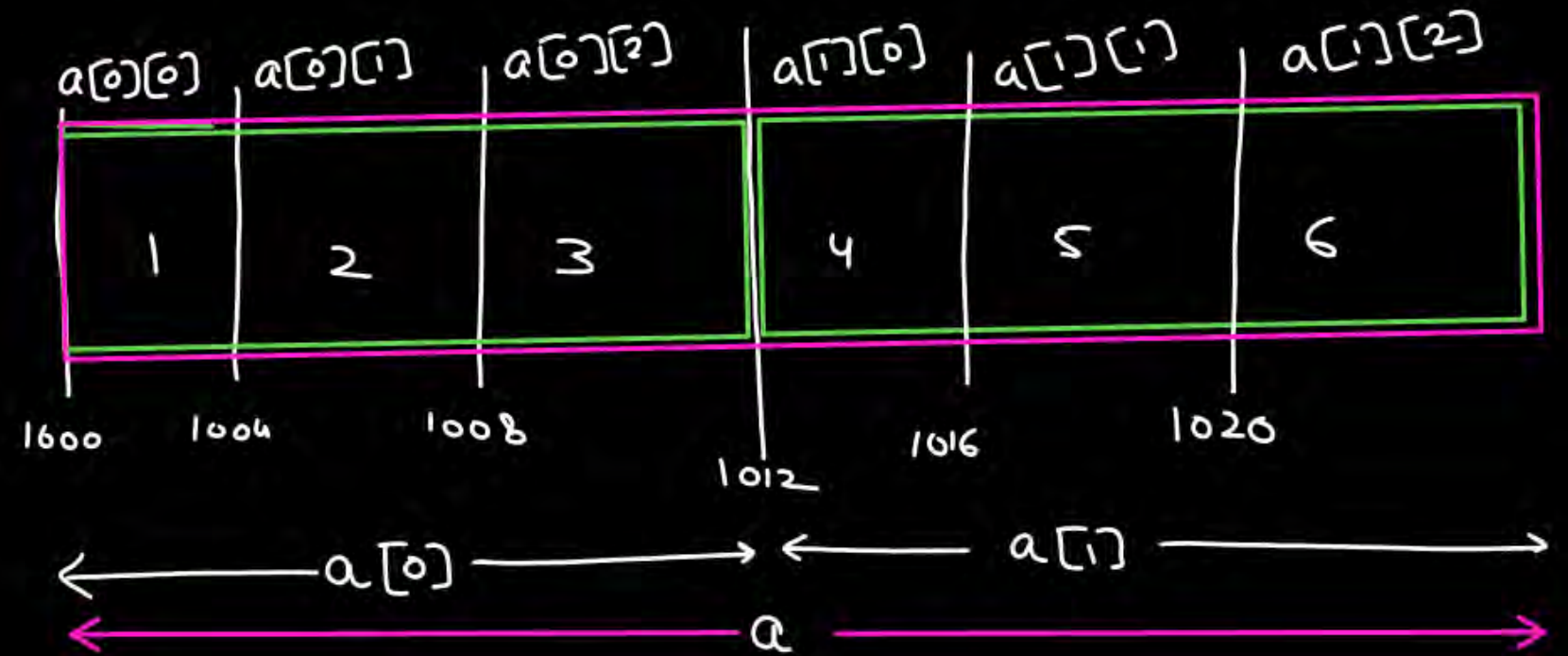
```
int a[2][3] = {1, 2, 3, 4, 5, 6};
```

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

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

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

```
}
```



$$\begin{aligned}
 a[0] + 1 &= \&a[0][0] + 1 \\
 &= \&a[0][0] + 1 \times 4 \\
 &= 1000 + 4 \\
 &= 1004
 \end{aligned}$$

```
void main(){
```

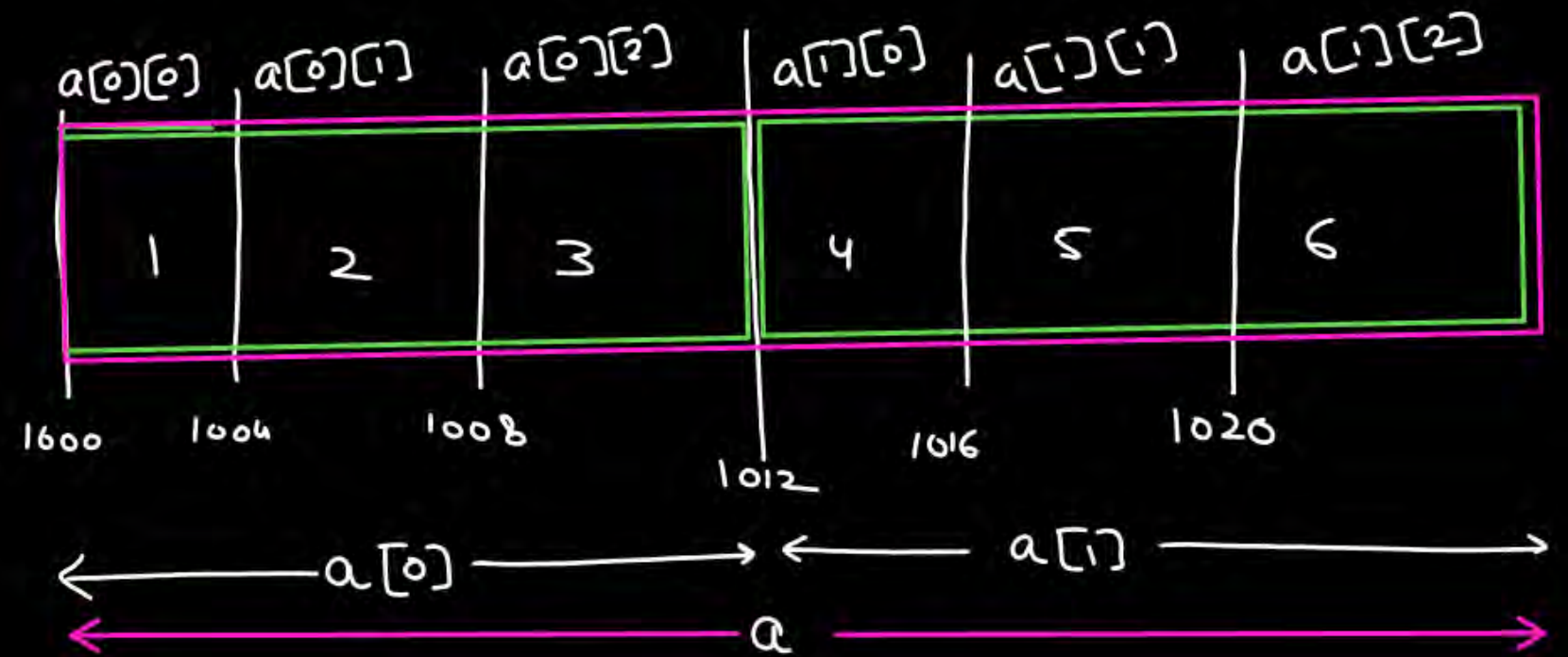
```
int a[2][3] = {1, 2, 3, 4, 5, 6};
```

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

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

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

```
}
```



$$\begin{aligned}\&a+1 &= \&a+1 \times 24 \\ &= 1000 + 24 \\ &= 1024\end{aligned}$$



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

100 `pf("/d", a);`  $\rightarrow$  `&a[0]`

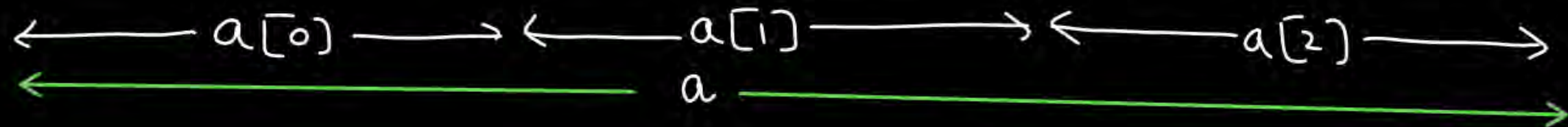
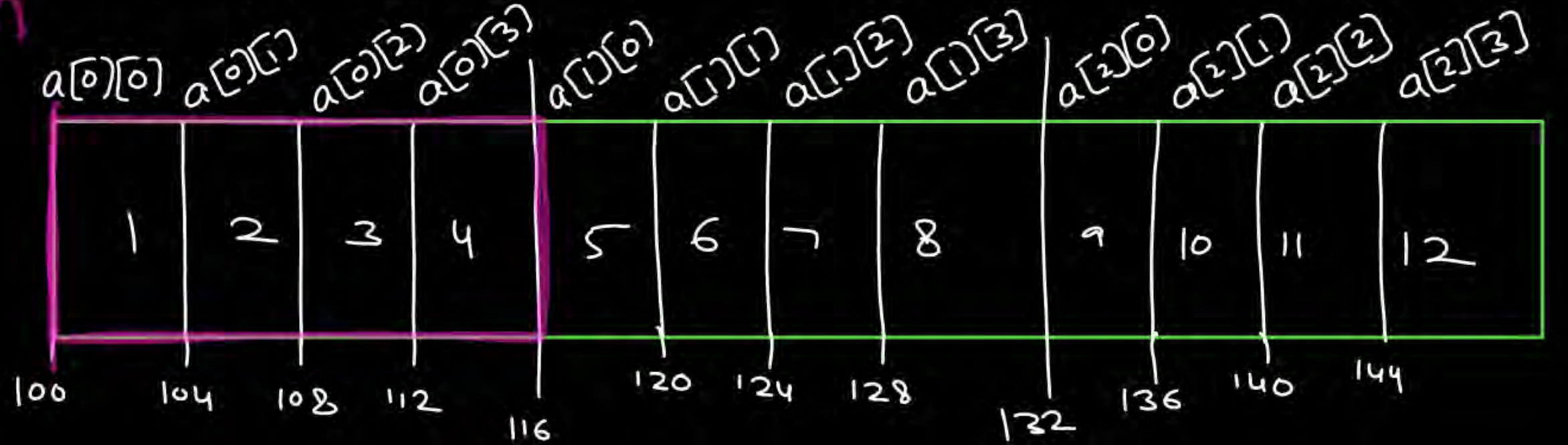
`pf("/d", a[0]);`

`pf("/d", &a);`

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

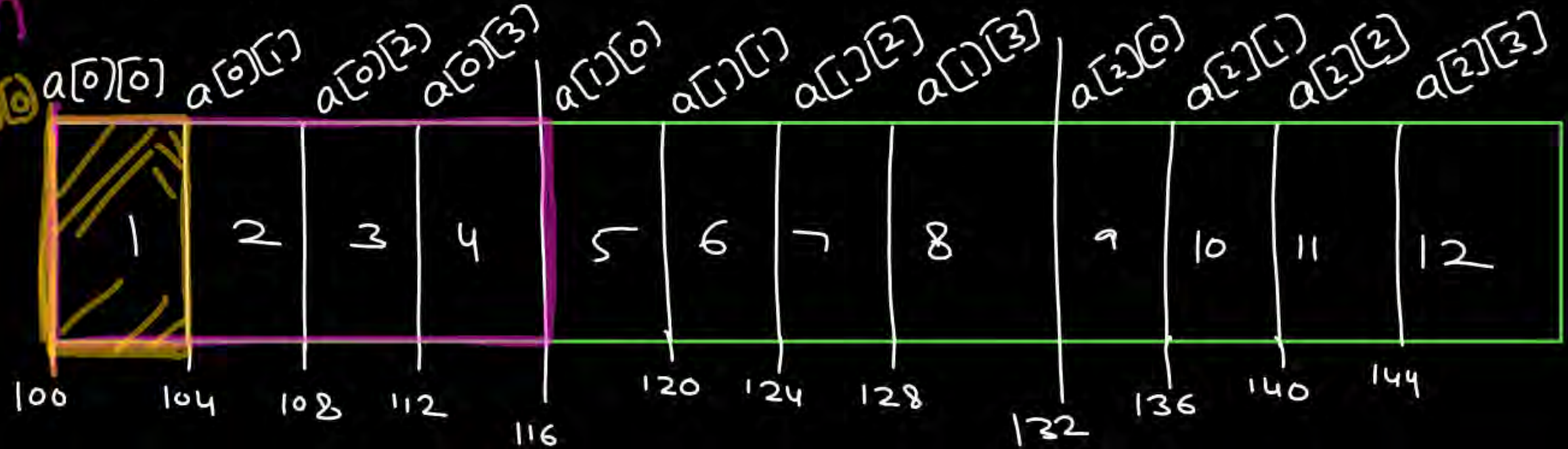
`pf("/d", a[0]+1);`

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



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

100 `printf("/d", a);`  $\rightarrow \&a[0]$   
100 `printf("/d", a[0]);`  $\rightarrow \&a[0][0]$

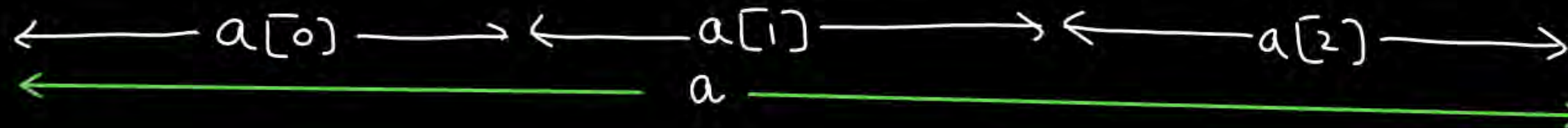


`printf("/d", &a);`

`printf("/d", a+1);`

`printf("/d", a[0]+1);`

`printf("/d", &a+1);`





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

`100 pf("/d", a);`  $\rightarrow \&a[0]$

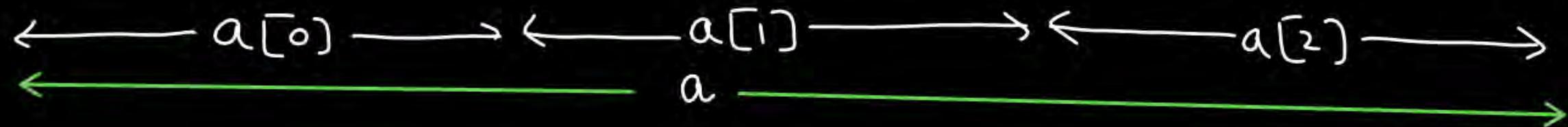
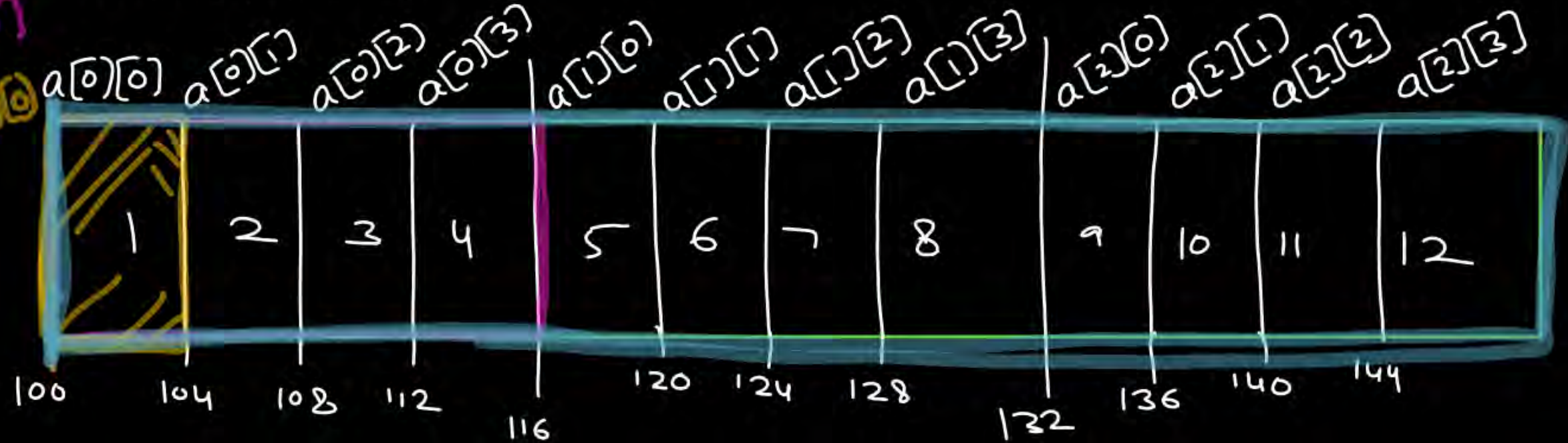
`100 pf("/d", a[0]);`  $\rightarrow \&a[0][0]$

`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 `printf("/d", a);`  $\rightarrow \&a[0]$

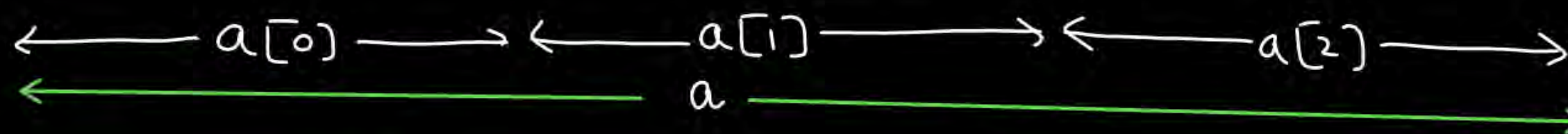
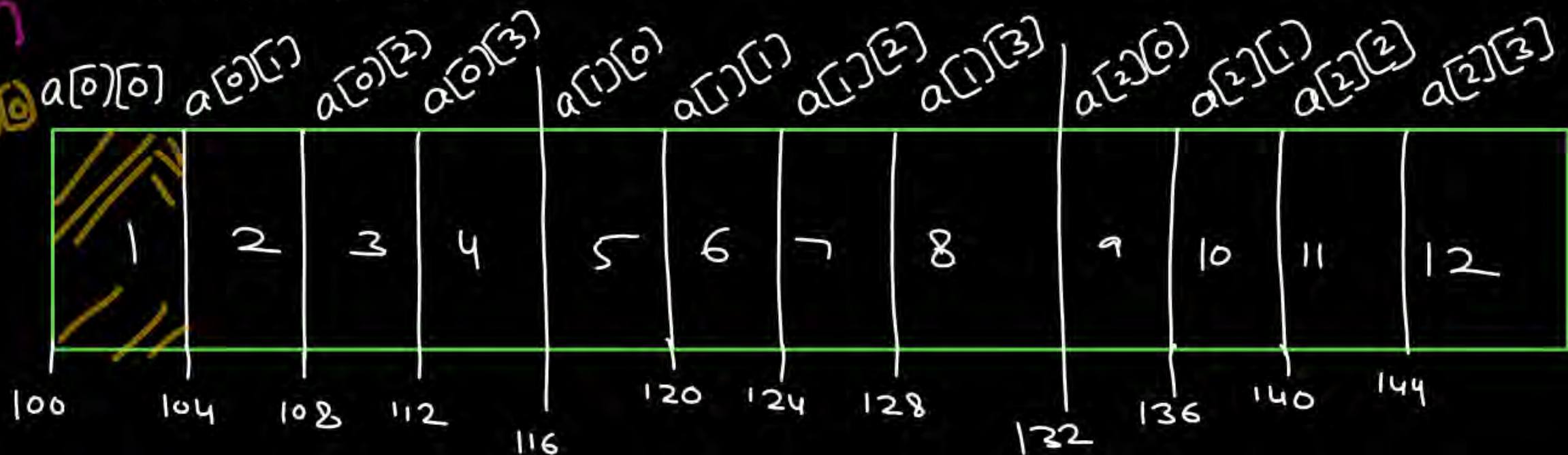
100 `printf("/d", a[0]);`  $\rightarrow \&a[0][0]$

100 `printf("/d", &a);`

116 `printf("/d", a+1);`

`printf("/d", a[0]+1);`

`printf("/d", &a+1);`



$$\begin{aligned}
 a+1 &= \&a[0] + 1 \\
 &= \&a[0] + 1 \times 16 \\
 &= 100 + 16 \\
 &= 116
 \end{aligned}$$

`sizeof a[0] = 16 byte`

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

100 `printf("/d", a);`  $\rightarrow \&a[0]$

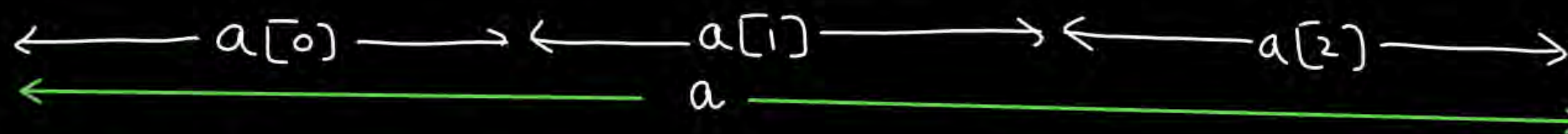
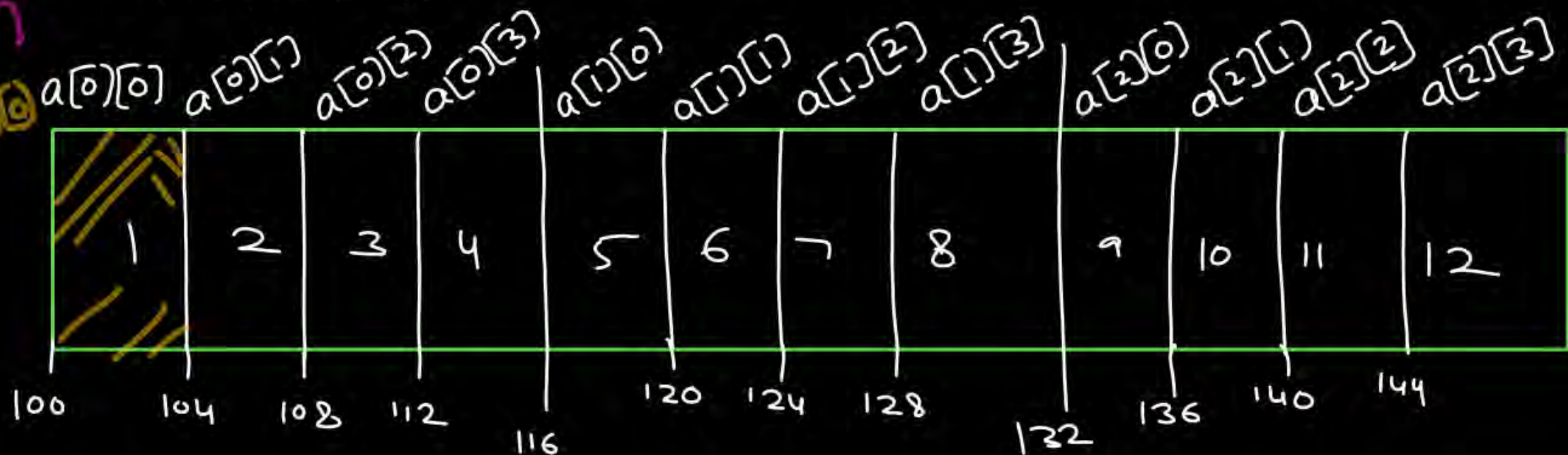
100 `printf("/d", a[0]);`  $\rightarrow \&a[0][0]$

100 `printf("/d", &a);`

116 `printf("/d", a+1);`

104 `printf("/d", a[0]+1);`

`printf("/d", &a+1);`



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

$$\begin{aligned} a[0]+1 &= \&a[0][0]+1 \\ &= \&a[0][0]+1 \times 4 \\ &= 100+4=104 \end{aligned}$$

`sizeof a[0][0] = 4 byte`



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

100 `printf("/d", a);`  $\rightarrow \&a[0]$

100 `printf("/d", a[0]);`  $\rightarrow \&a[0][0]$

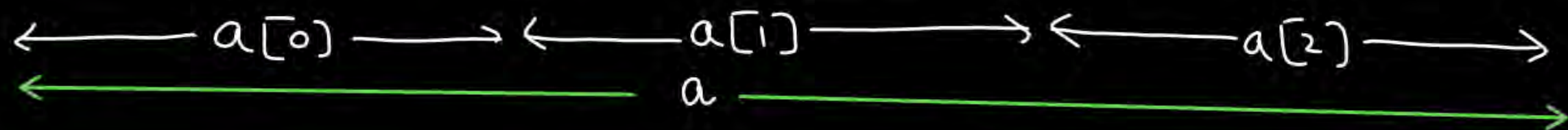
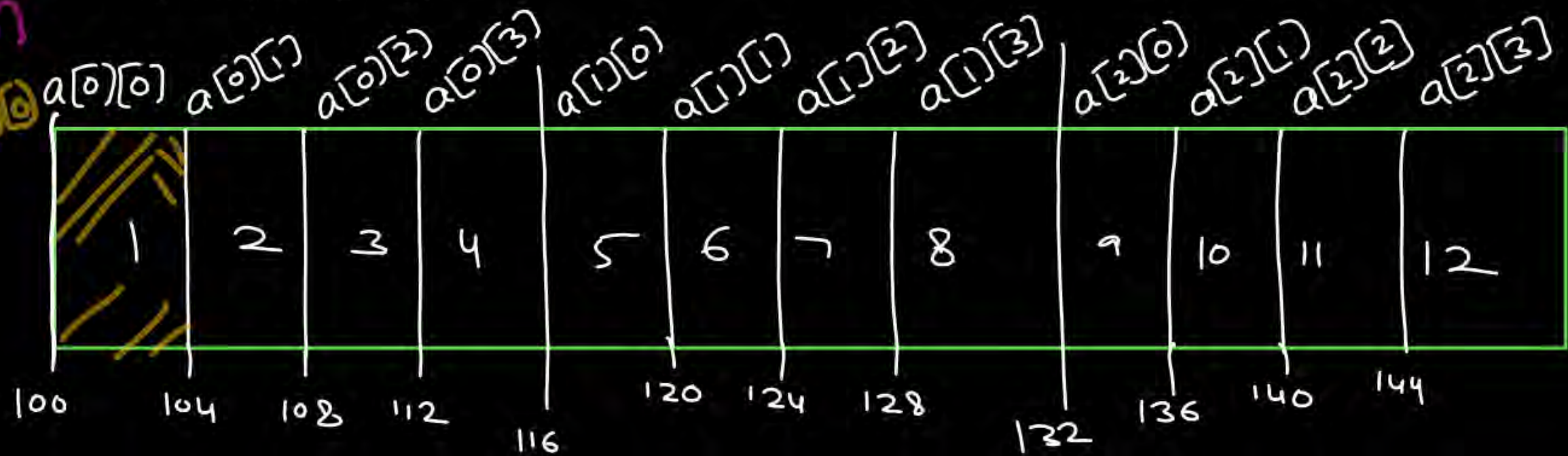
100 `printf("/d", &a);`

116 `printf("/d", a+1);`

104 `printf("/d", a[0]+1);`

`printf("/d", &a+1);`

$$\begin{aligned} &\rightarrow \&a+1 \\ &= \&a+1 \\ &= 100 + 1 \times 48 = 148 \end{aligned}$$

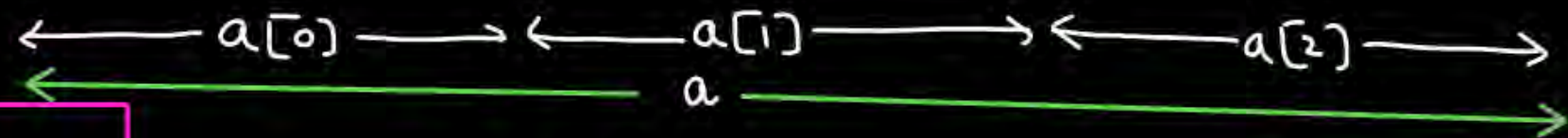
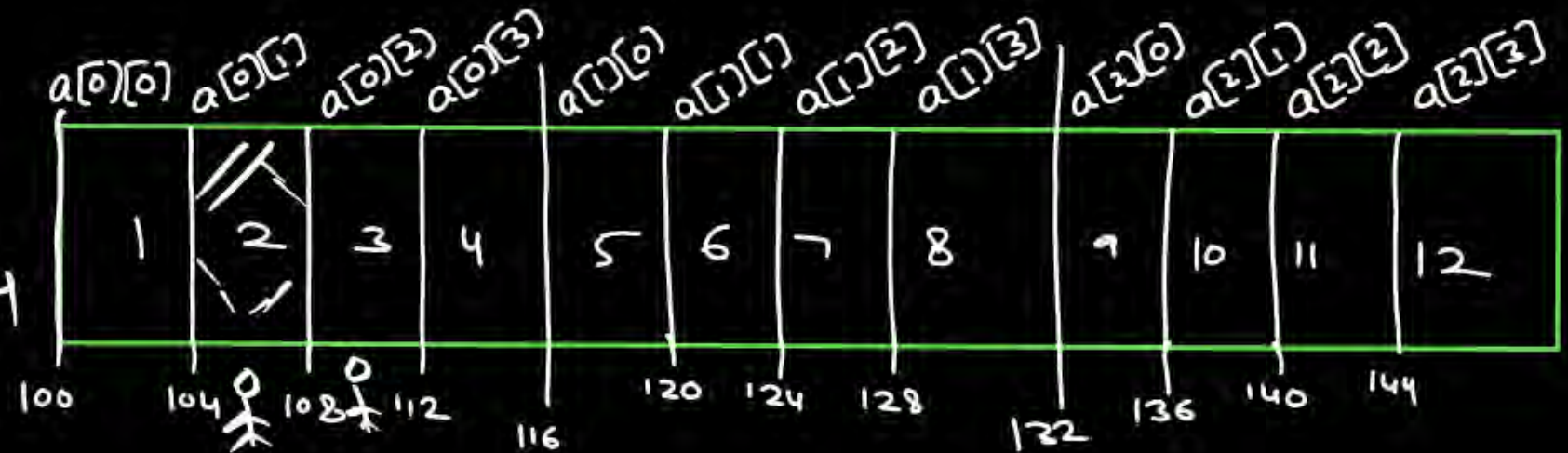


size of complete array `a`  
= 48 byte



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

$$\begin{aligned}
 a[0] + 1 &= \underbrace{4a[0][0]}_{4 \text{ byte}} + 1 \\
 &= 4a[0][0] + 1 \times 4 \\
 &= 100 + 4
 \end{aligned}$$



$$a[0] + 1 = \left( \begin{array}{c} \text{Memory} \\ \text{location} \\ 104 \end{array} \right) = 4a[0][1]$$

$$\begin{aligned}
 a[0] + 2 &= \underbrace{4a[0][0]}_{4 \text{ byte}} + 2 \\
 &= 4a[0][0] + 2 \times 4 \\
 &= 100 + 8 = 108
 \end{aligned}$$

$$a[0] + 2 = \left( \begin{array}{c} \text{Memory} \\ \text{location} \\ 108 \end{array} \right) = 4a[0][2]$$

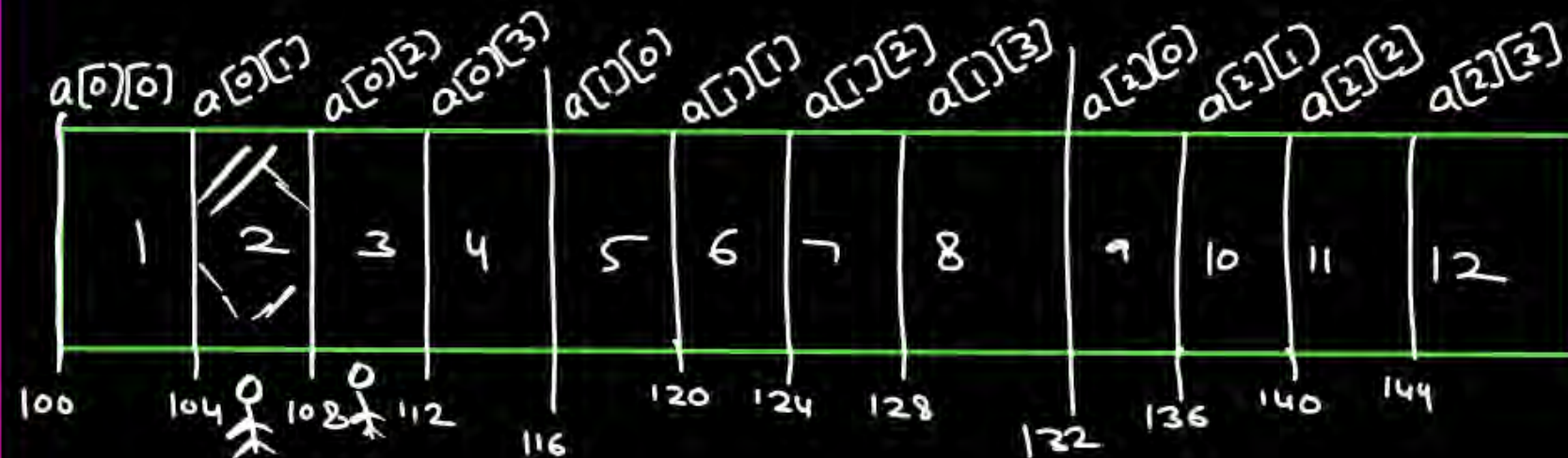
2.  $\text{int } a[3][4] = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\};$

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

104

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

108



$$\begin{aligned} a[1] + 1 &= \&a[1][0] + 1 \\ &\quad \text{4 byte} \\ &= \&a[1][0] + 1 \times 4 \\ &= 116 + 4 = 120 \end{aligned}$$

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

$$\begin{aligned} a[1] + 2 &= \&a[1][0] + 2 \\ &\quad \text{4 byte} \\ &= \&a[1][0] + 2 \times 4 \\ &= 124 \end{aligned}$$

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



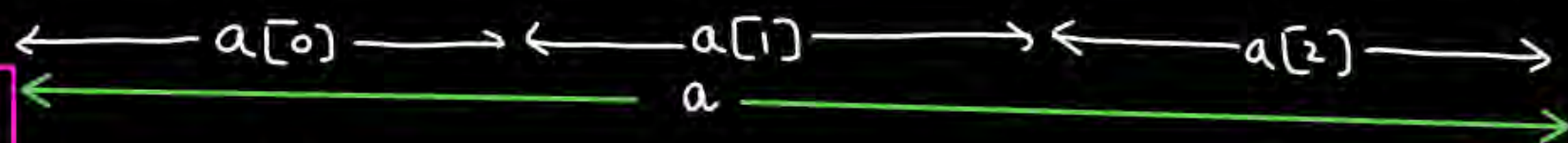
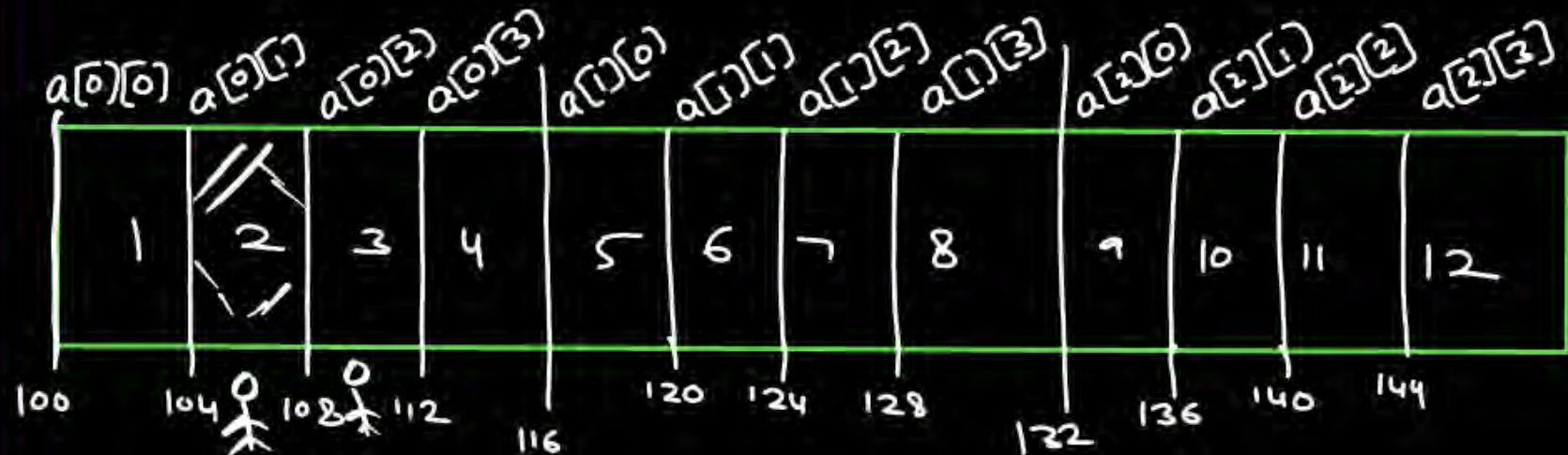
2.  $\text{int } a[3][4] = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\};$

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

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

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

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



$$\star (a[0]+1) = \text{value at } \left( \begin{smallmatrix} \text{Mem.} \\ \text{loc.} \\ 104 \end{smallmatrix} \right) = \star \&a[0][1]$$

① —  $\star (a[0]+1) = a[0][1]$

$$\star (a[0]+2) = \text{value at } \left( \begin{smallmatrix} \text{Mem.} \\ \text{loc.} \\ 108 \end{smallmatrix} \right) = \star \&a[0][2]$$

② —  $\star (a[0]+2) = a[0][2]$



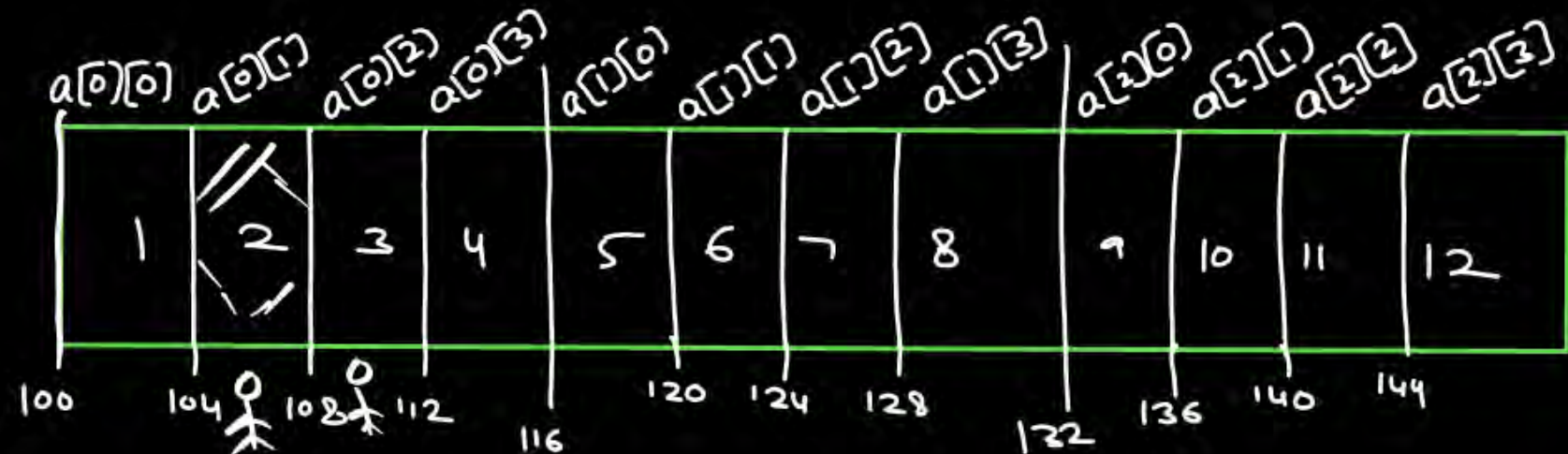
2. `int a[3][4] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};`

① -  $\star(a[0] + 1) = a[0][1]$

② -  $\star(a[0] + 2) = a[0][2]$

③ -  $\star(a[1] + 1) = a[1][1]$

④ -  $\star(a[1] + 2) = a[1][2]$



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

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

$\star(a[1] + 1) = \underset{\text{at } \text{loc. } 120}{\text{value (Mem)}} = \&a[1][1]$

$\star(a[1] + 1) = a[1][1]$

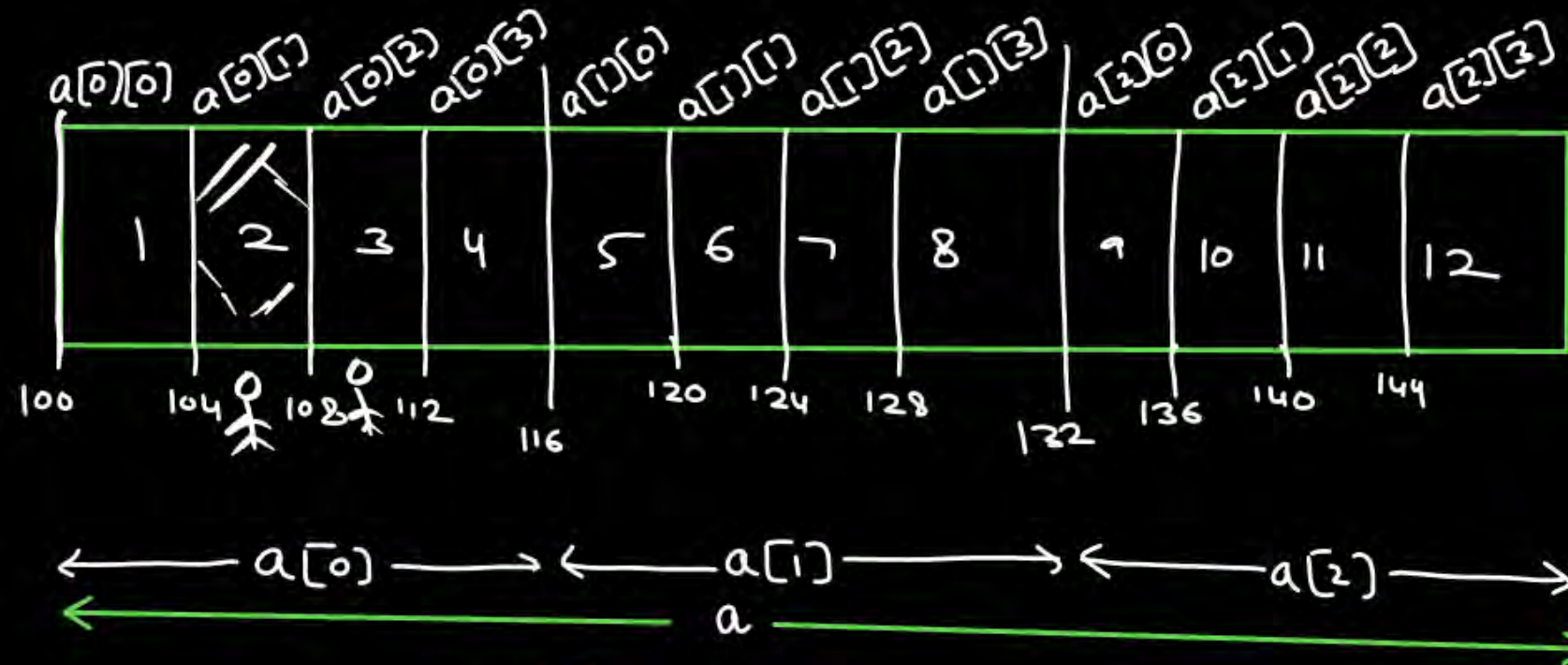
$\star(a[1] + 2) = \underset{\text{at } \text{loc. } 124}{\text{value (Mem)}} = \&a[1][2]$

$\star(a[1] + 2) = a[1][2]$



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

- ① -  $\star(a[0] + 1) = a[0][1]$
- ② -  $\star(a[0] + 2) = a[0][2]$
- ③ -  $\star(a[1] + 1) = a[1][1]$
- ④ -  $\star(a[1] + 2) = a[1][2]$



$$\star(a[0] + j) = a[0][j]$$

$$\star(a[1] + j) = a[1][j]$$

---


$$\star(a[i] + j) = a[i][j]$$

$$\Rightarrow \left[ \begin{array}{l} \star(a[i] + j) = a[i][j] \\ \star(\star(a + i) + j) = a[i][j] \end{array} \right]$$

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

`1000 printf("/d", a);`

`1000 printf("/d", a[0]);`

`1000 printf("/d", &a);`

`1000 printf("/d", *a);`

`1 printf("/d", **a);`

`1004 printf("/d", *a+1);`

`printf("/d", **a+1);`

`printf("/d", *a[0]);`

$a[0][0]$											
1	2	3	4	5	6	7	8	9	10	11	12
1000	1004	1008	1012	1016	1020	1024	1028	1032	1036	1040	1044

←  $a[0]$  → ←  $a[1]$  → ←  $a[2]$  →  
←  $a$  →

$*a = \&a[0]$   
 $= a[0]$   
 $= \&a[0][0]$   
 $= 1000$

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

$*a+1 = \&a[0][0] + 1$   
 $= \underbrace{\&a[0][0]}_{\text{4 byte}} + 1 \times 4$   
 $= 1004$



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

`1000 printf("/d", a);`

`1000 printf("/d", a[0]);`

`1000 printf("/d", &a);`

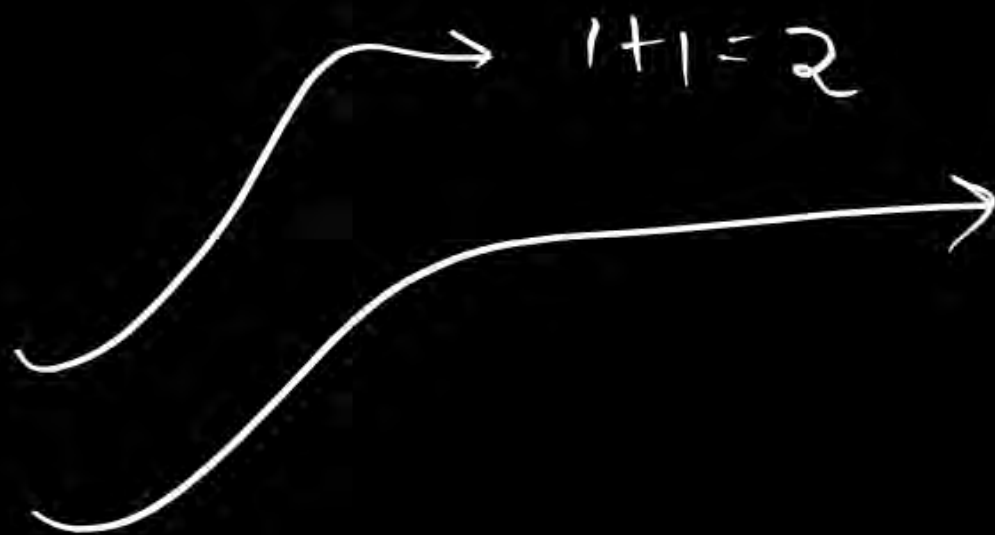
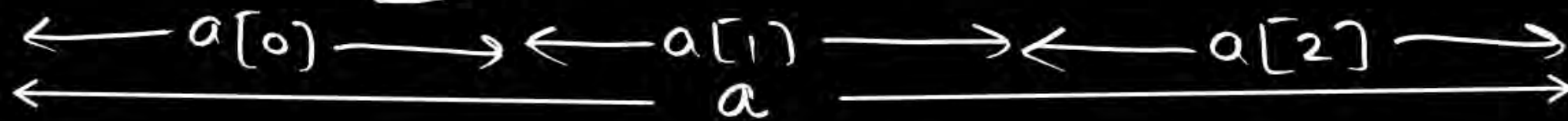
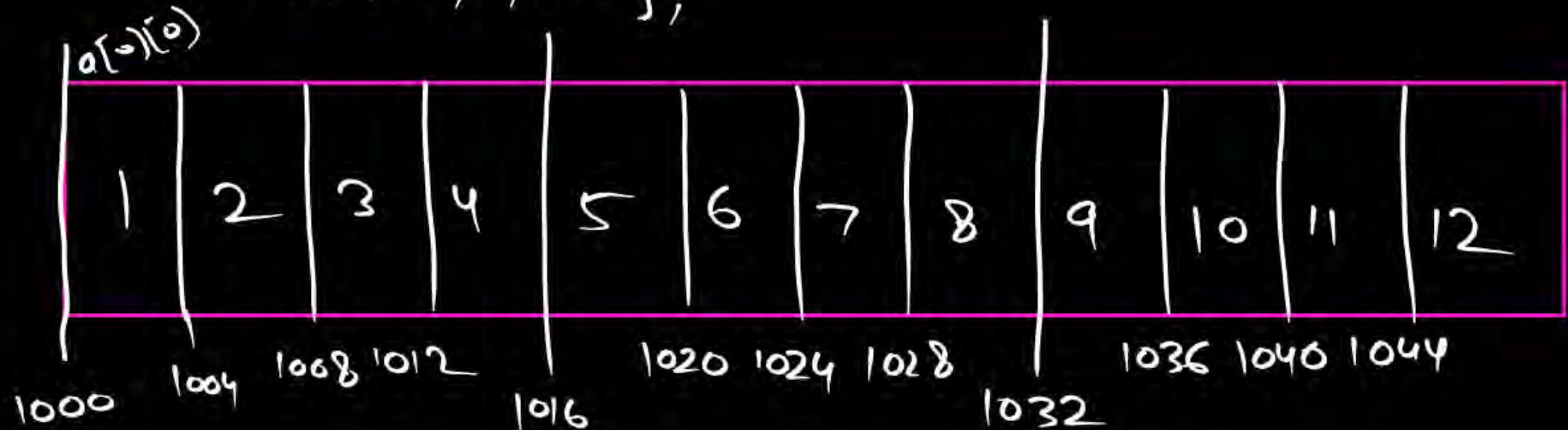
`1000 printf("/d", *a);`

`1 printf("/d", **a);`

`1004 printf("/d", *a+1);`

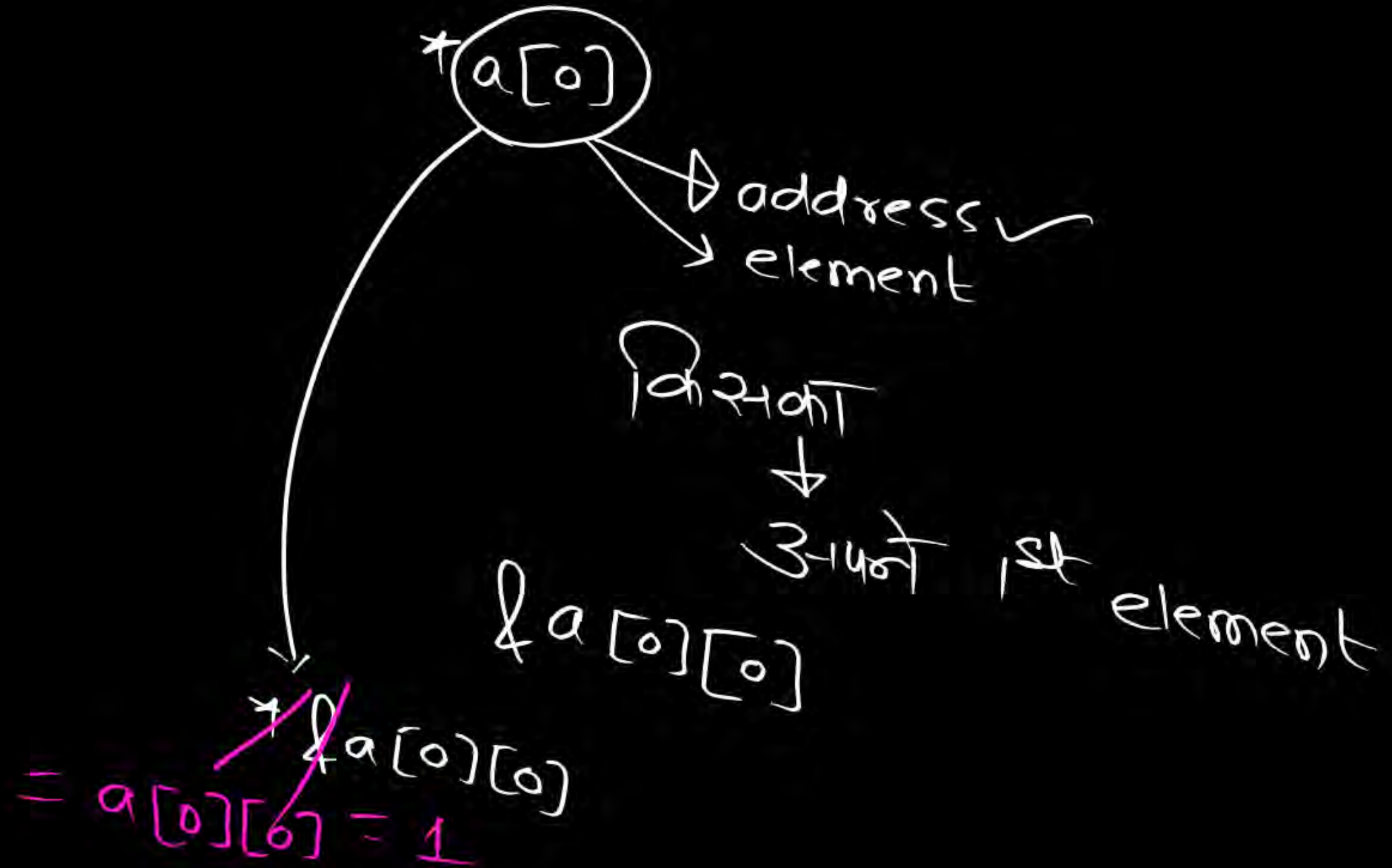
`2 printf("/d", **a+1);`

`1 printf("/d", *a[0]);`



`*a[0]`  
~~`*a[0][0]`~~  
`= a[0][0]`  
`= 1`

$a[2][3]$



$a[2][3]$

$$**a + 1$$

$$a = \overset{\checkmark}{\text{Add}} / \overset{\times}{\text{element}}$$

↓  
1st element

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

$$*a = \cancel{*} \cancel{\&a[0]}$$

$$\boxed{*a = a[0]}$$

$$*a = a[0] \begin{matrix} \nearrow \text{Add} \checkmark \\ \searrow \text{elem.} \times \end{matrix}$$

↓

$$*a = \&a[0][0]$$

$$\begin{aligned} *( *a) &= *( \&a[0][0]) \\ &= \cancel{*} \cancel{\&a[0][0]} \end{aligned}$$

$$\boxed{**a = a[0][0]}$$

$$\begin{aligned} ***a + 1 &= a[0][0] + 1 \\ &= 1 + 1 \\ &= 2 \end{aligned}$$



→ concept

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

$$a[0] + 4$$

$$\begin{aligned} &\rightarrow \rightarrow \rightarrow \underline{\underline{701}} \end{aligned}$$

$$\Rightarrow 2a[0][0] + 4$$

⇒  
4 byte

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

$$\in 1016$$

