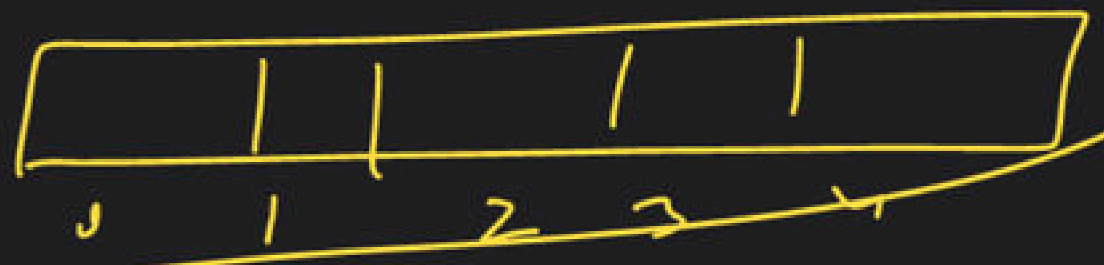
Array Level-3 [Join Here]

Special class

Love Babbar • Sept 11, 2023

→ 1D – Array

int arr[5];



int arr[] = {1, 2, 40, 50}

access
↳ , index

with function

reference

→ Arrays → Question

→ Array Level-2 → MD R.C

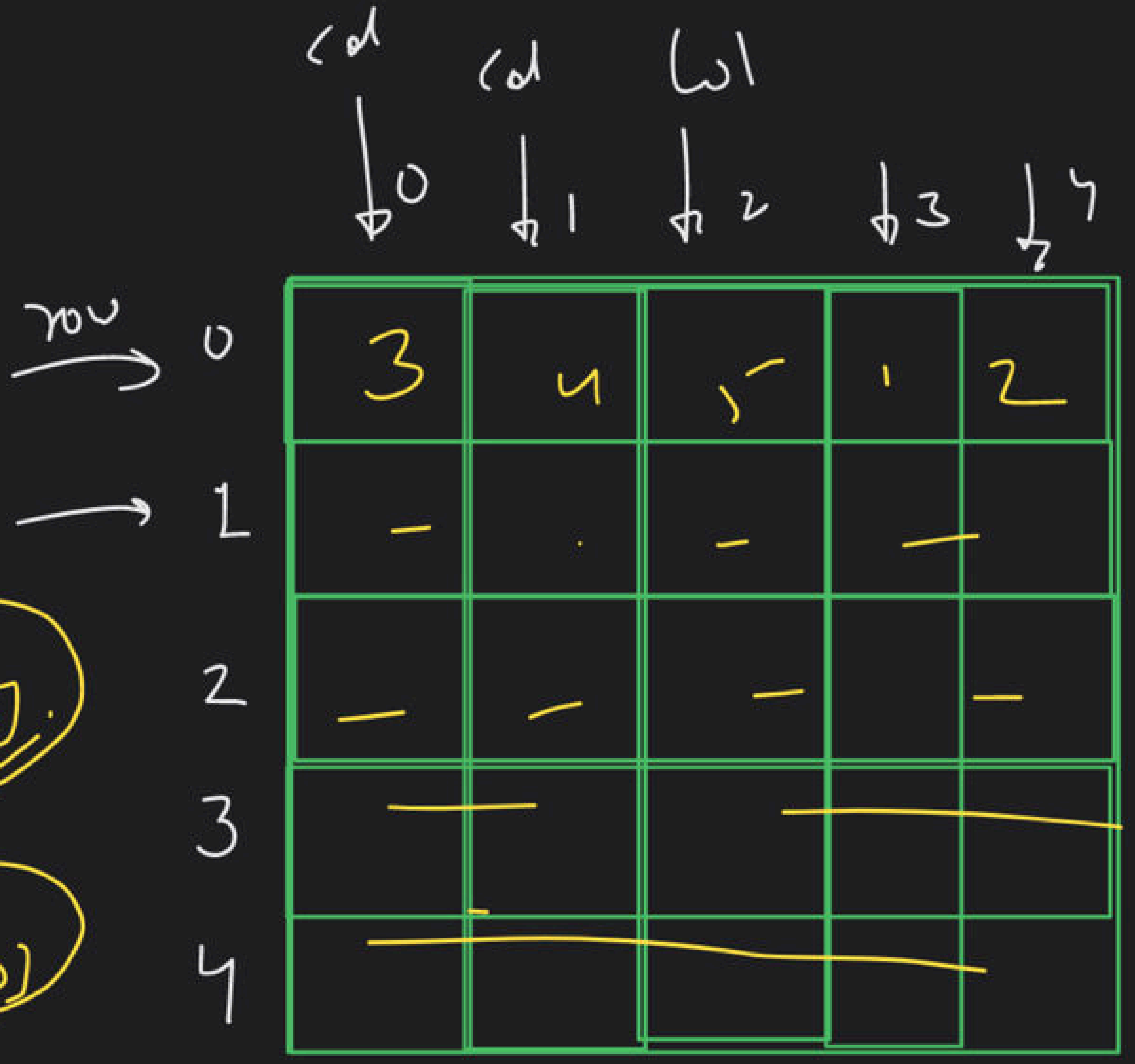
→ 2D - Array

Creation

1D → int arr[5];

2D → row → 5, col → 10 → int arr[5][10];

2D → 100 row, 1000 col → int arr[100][1000]

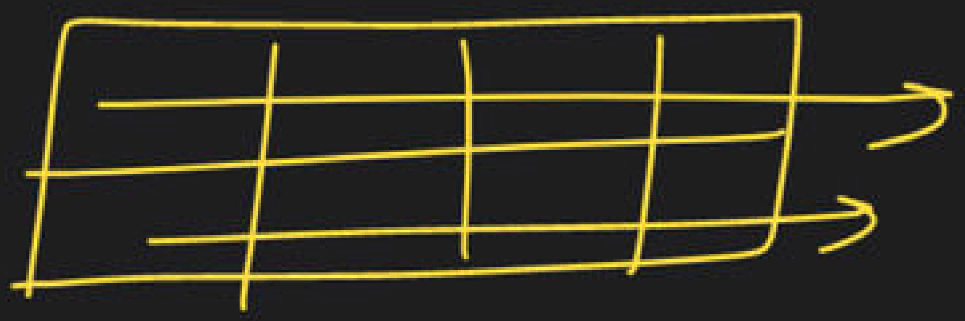


initialize

1D $\rightarrow$ int arr[] = { 10, 20, 30 }

2D $\rightarrow$ row $\rightarrow$ 2 $\longrightarrow$ int arr[2][4] = {

col $\rightarrow$ 4

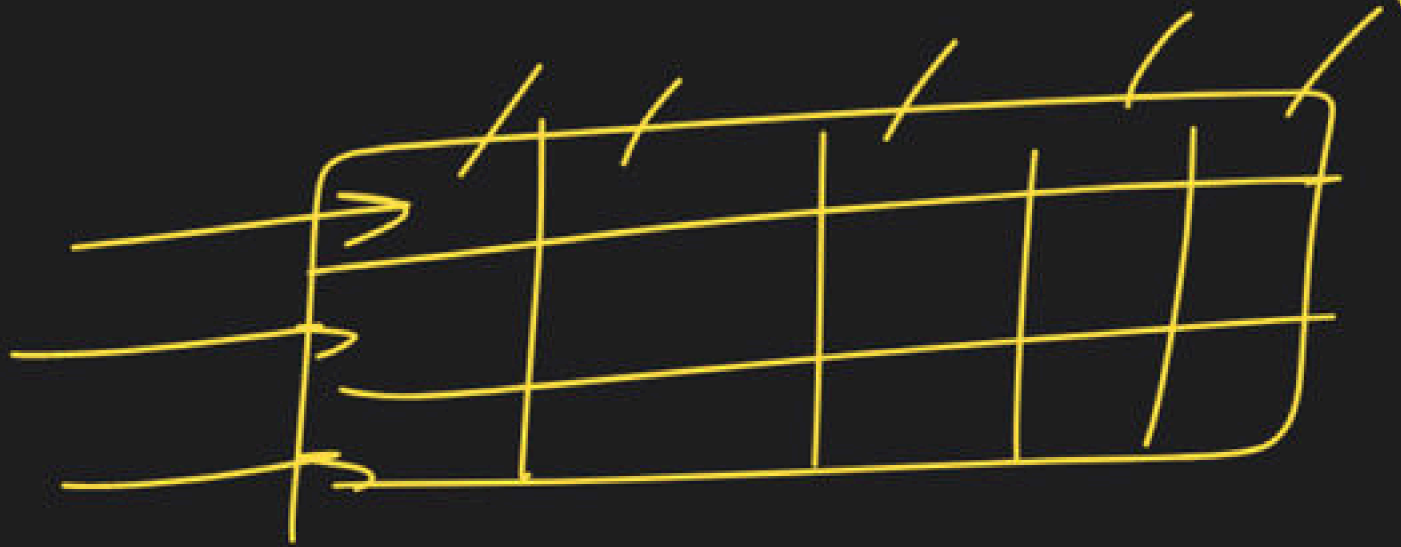


$\Rightarrow$ {10, 20, 30, 40},

$\Rightarrow$ {80, 70, 60, 50}

};

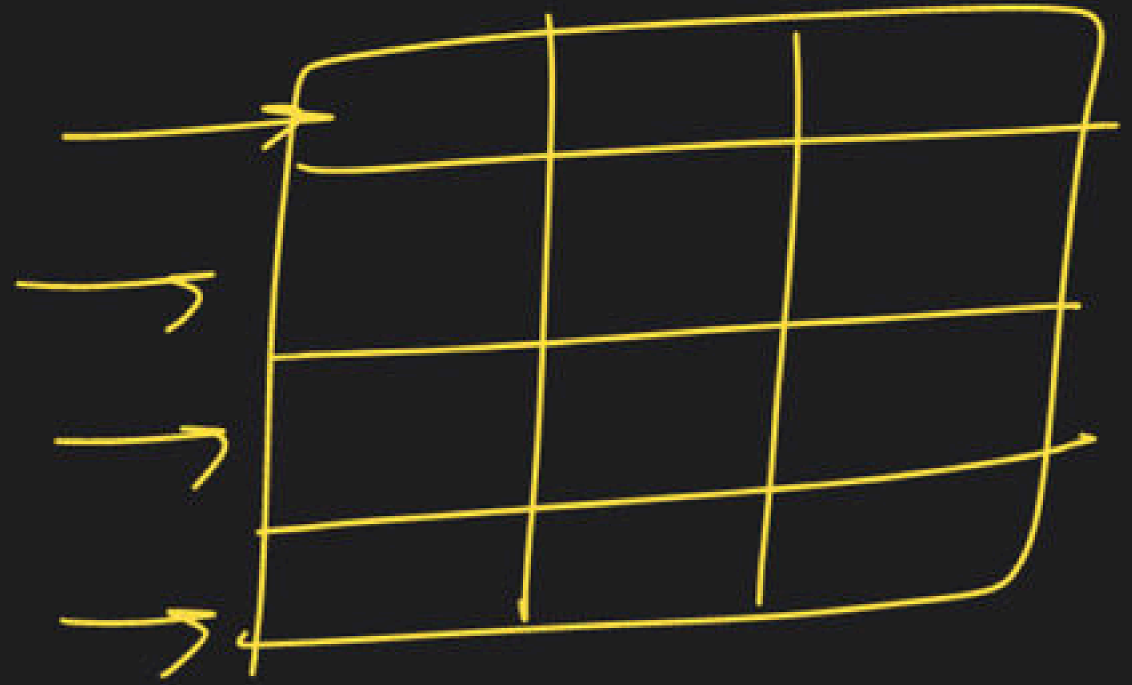
2D → row 3 row
5 cols →



```
int arr[3][5]
    = {
        {1, 2, 3, 4, 5},
        {6, 7, 8, 9, 10},
        {10, 20, 30, 40, 50}
      };
```

2D → row → 4
col → 3
=

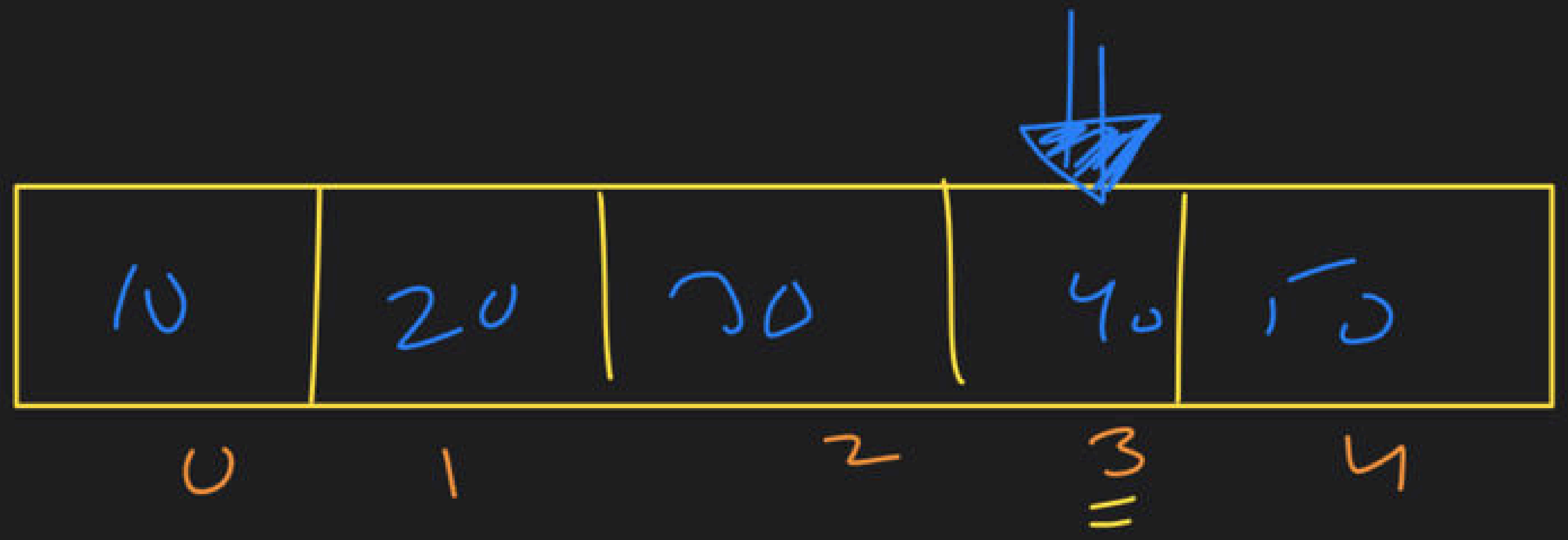
```
int arr[4][3]
  = {
      {10, 20, 30},
      {1, 2, 3},
      {4, 6, 8},
      {5, 7, 9}
    }
```



Access

1D → index

$\boxed{arr[i]}$



arr[3] = 40

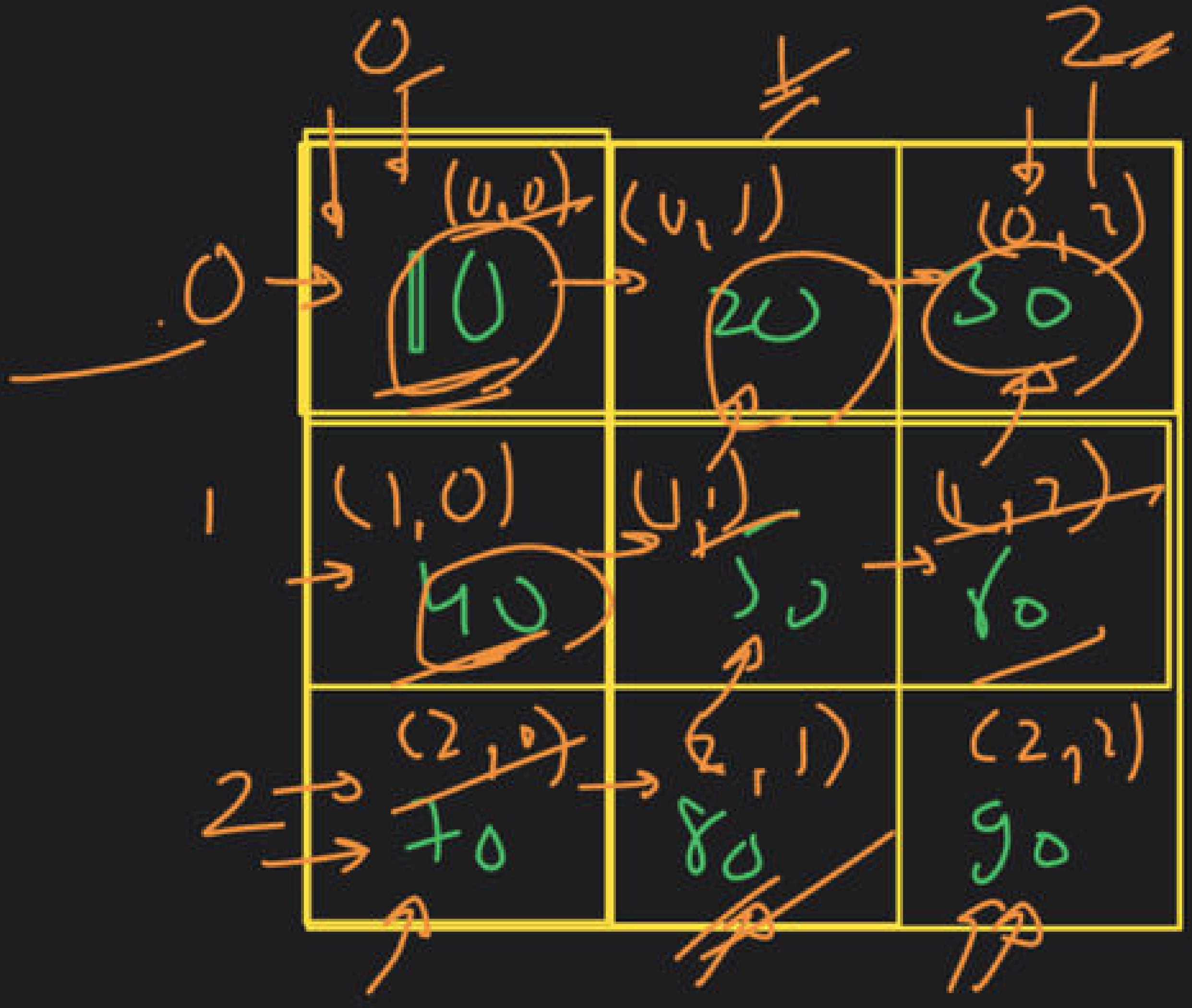
2D arr

→ index

$\boxed{arr[i][j]}$

i = row index

j = col index



arr[0][0] = 10

arr[0][1] = 20

arr[0][2] = 30

arr[1][0] = 40

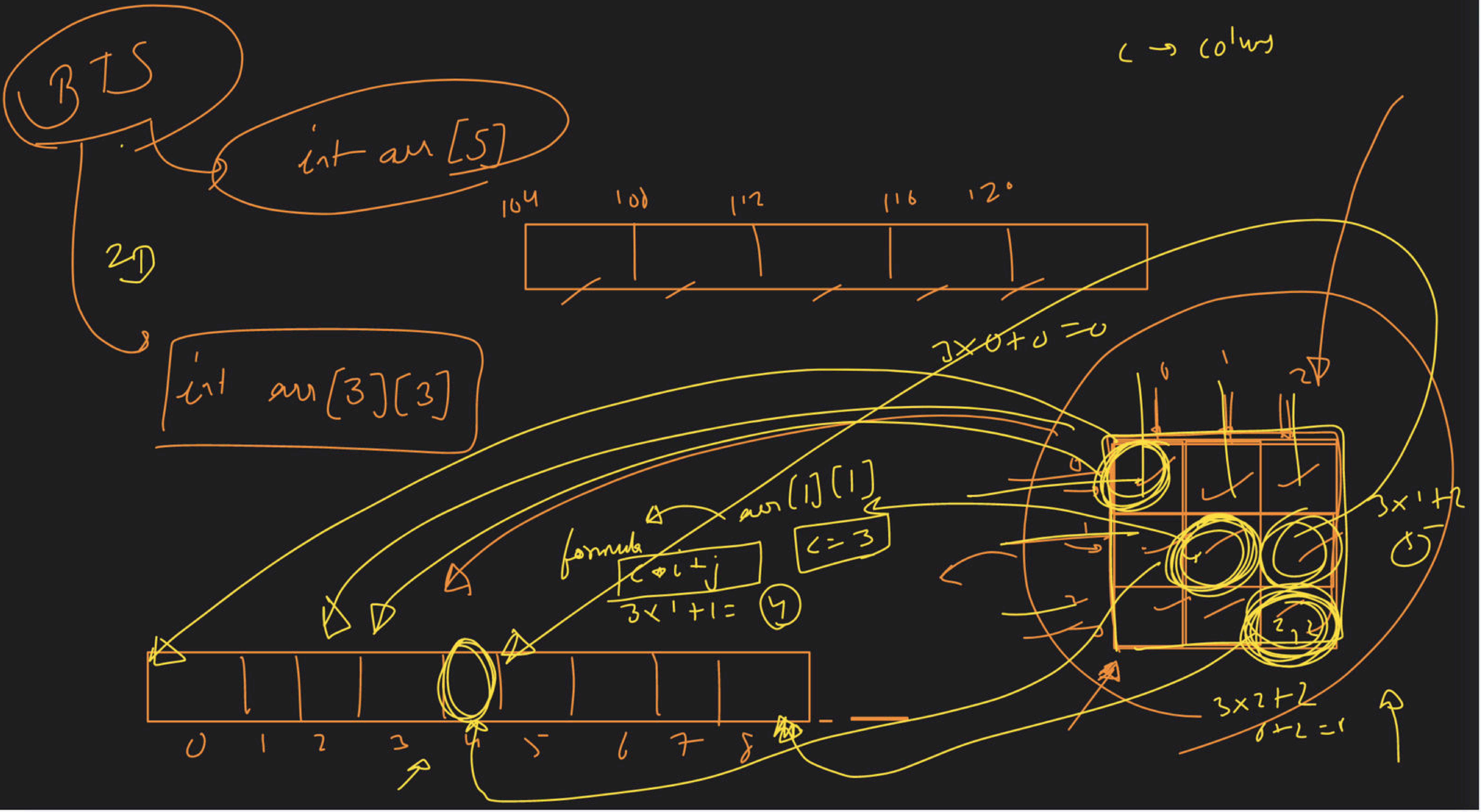
arr[1][1] = 50

arr[1][2] = 60

arr[2][0] = 7

arr[2][1] = 80

arr[2][2] = 90


BTS
int arr[5]
104
108
112
116
120
2D
int arr[3][3]
c → col'ws
3×0+0=0
0
1
2
arr[1][1]
formula
c*i+j
c=3
3×1+1=4
3×1+2
5
2,2
3×2+2
6+2=8
0 1 2 3 4 5 6 7 8

news

Xtra class

tomorrow → 9pm

$\text{rows} \rightarrow 5 \rightarrow r$

$\text{cols} \rightarrow \boxed{4} \rightarrow c$

$$\begin{array}{c|cccc}
 & 0 & 1 & 2 & 3 \\
\hline
0 & 10 & 20 & 30 & 40 \\
1 & 50 & \boxed{60} & \boxed{70} & 80 \\
2 & \boxed{90} & 100 & 110 & 120 \\
3 & 130 & 140 & 150 & 160 \\
4 & 170 & 180 & \boxed{190} & 200
\end{array}$$

$(1,2)$

$c \times i + j$

$4 \times 1 + 2$

$= \boxed{6}$

$4 \times 1 + 1$

$= \boxed{5}$

$4,2$

$c \times i + j$

$4 \times 4 + 2 = \boxed{18}$

					60	70												190	
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

target → row wise access

outer

```
for (int i=0; i<row; i++)
{
    for (j=0; j<col; j++)
    {
        cout << arr[i][j]
    }
}
```

i=3

	0	1	2
0	(0,0) 10	(0,1) 20	(0,2) 30
1	(1,0) 40	(1,1) 50	(1,2) 60
2	(2,0) 11	(2,1) 21	(2,2) 31
3	(3,0) 41	(3,1) 51	(3,2) 61

i=2
- j=0 → 2,0
- j=1 → 2,1
- j=1 → 2,2

i=1
- j=0 → 1,0
- j=1 → 1,1
- j=2 → 1,2

i=0
- j=0 → 0,0
- j=1 → 0,1
- j=2 → 0,2

→ Column wise access

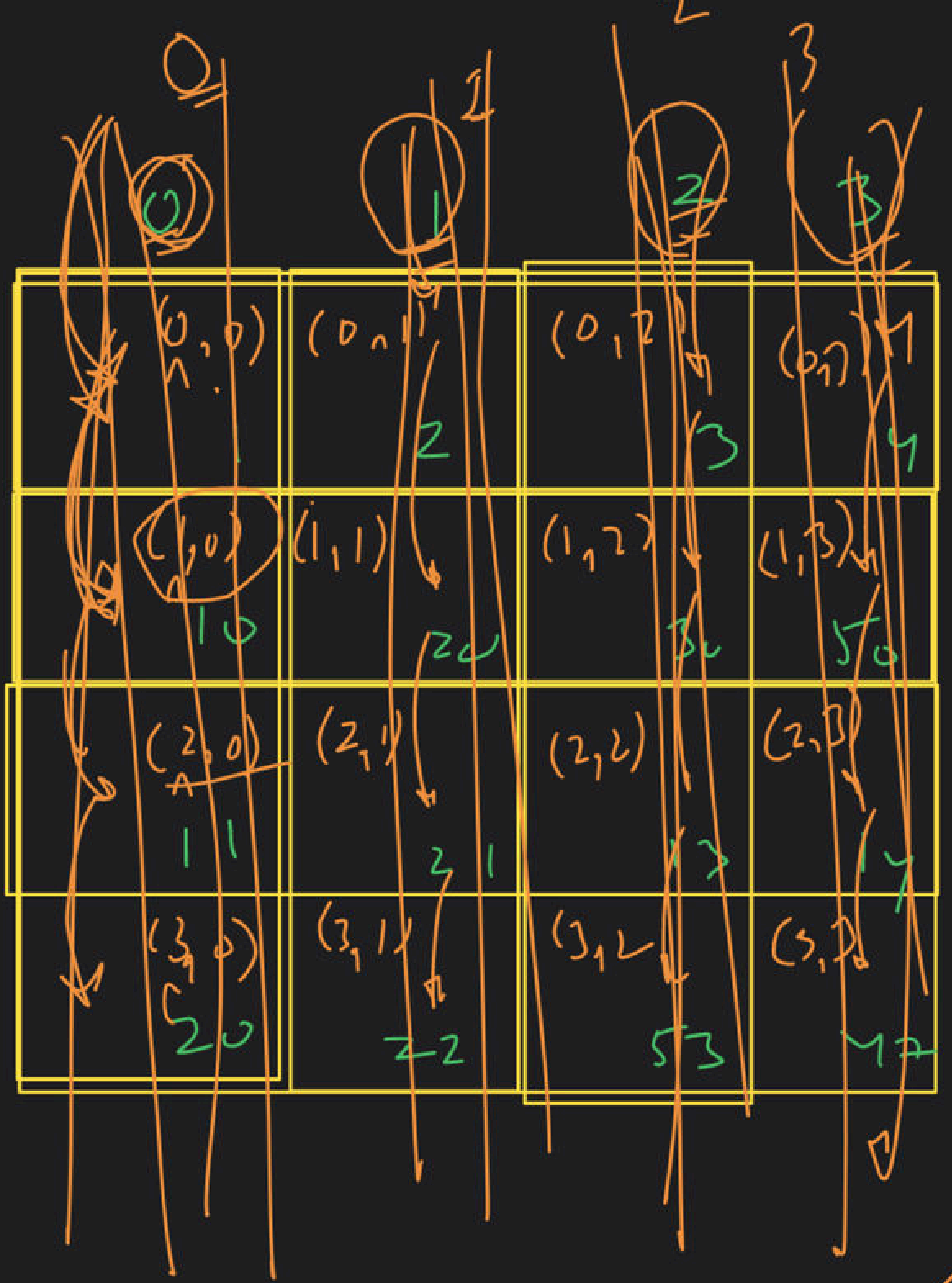
O.L

Outer loop
↳ Column

```
for (int i=0; i<col; i++)
{
    for(int j=0; j<row; j++)
    {
        cout << arr[j][i]
    }
}
```

Inner loop → row

0
1
2
3



[ ][ ]

i=0
↳ column index

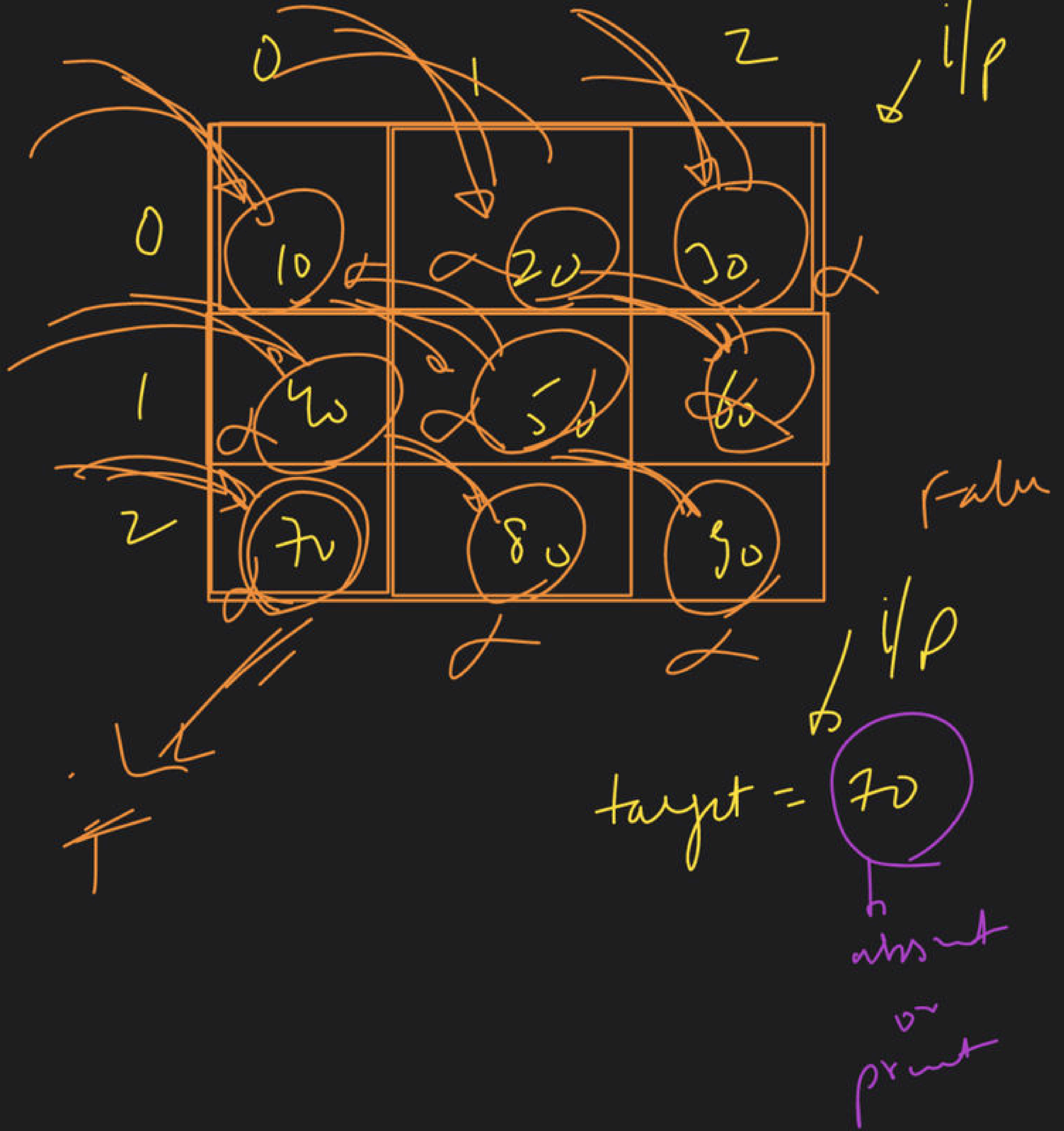
j=0
j=1
j=2
j=3
row index

i=0, j=3

arr[j][i]

→ Searching

o/p → T/F



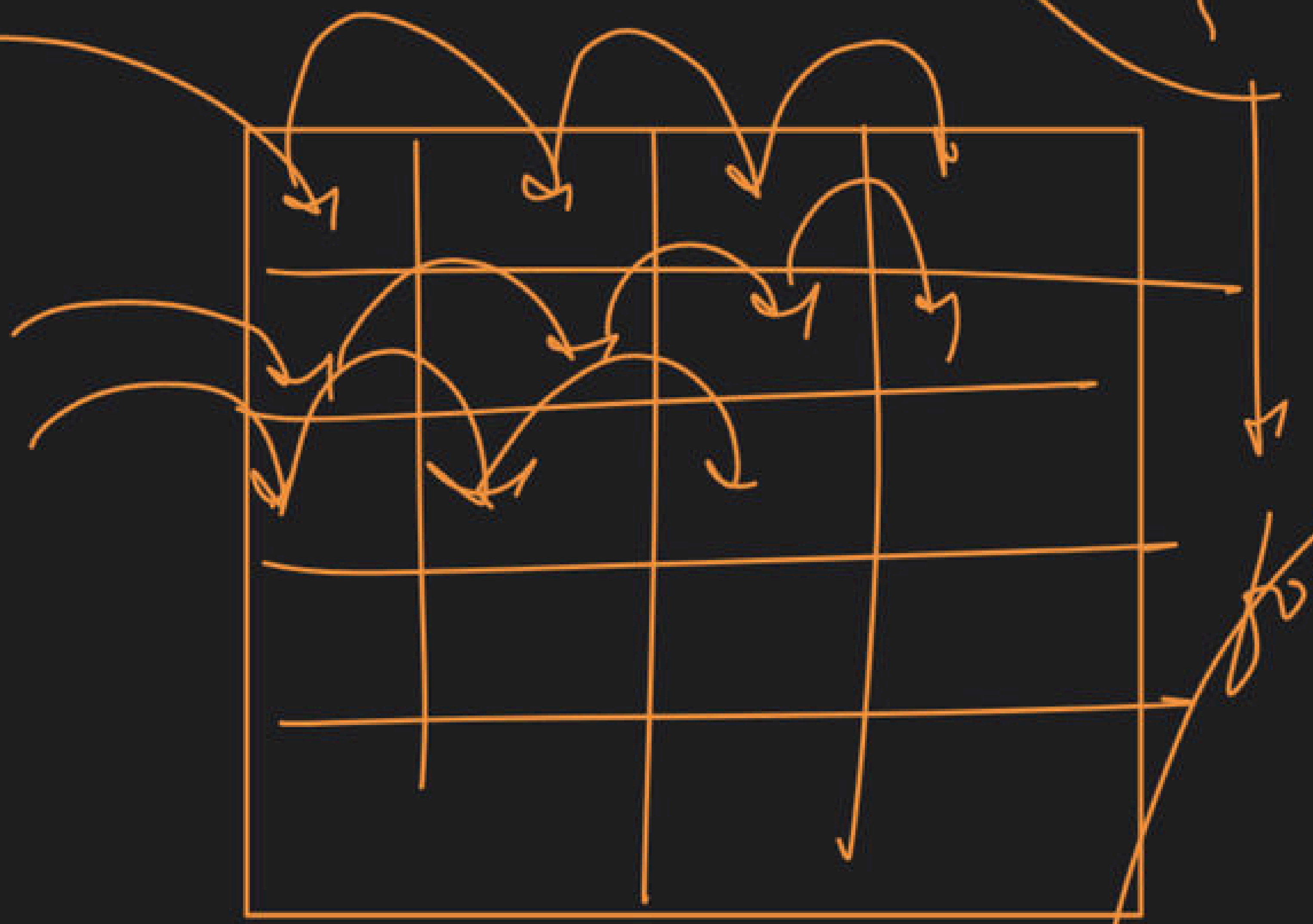
Max no in an Array

```
for( )
{
  if
}
```

int maxAns = INT_MIN;

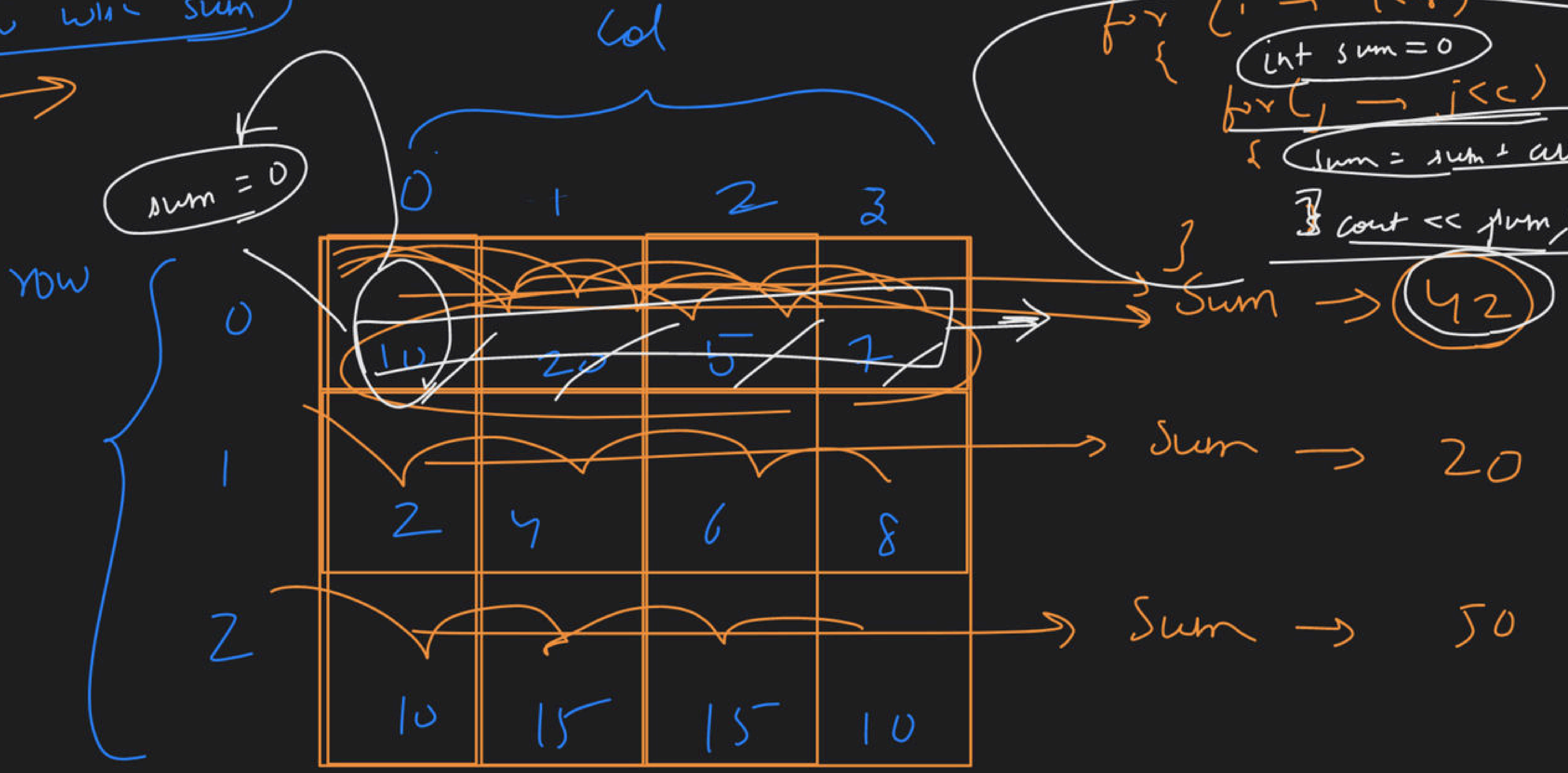
ni > maxAns

maxAns = ni



```
for( )
  for( )
  {
  }
```

Row wise sum



$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 10 & 20 & 5 & 7 \\ 1 & 2 & 4 & 6 & 8 \\ 2 & 10 & 15 & 15 & 10 \end{array}$$

```
for (i → i<r)
{
    int sum=0
    for(j → j<c)
    {
        sum = sum + arr[i][j]
    }
    cout << sum;
}
```

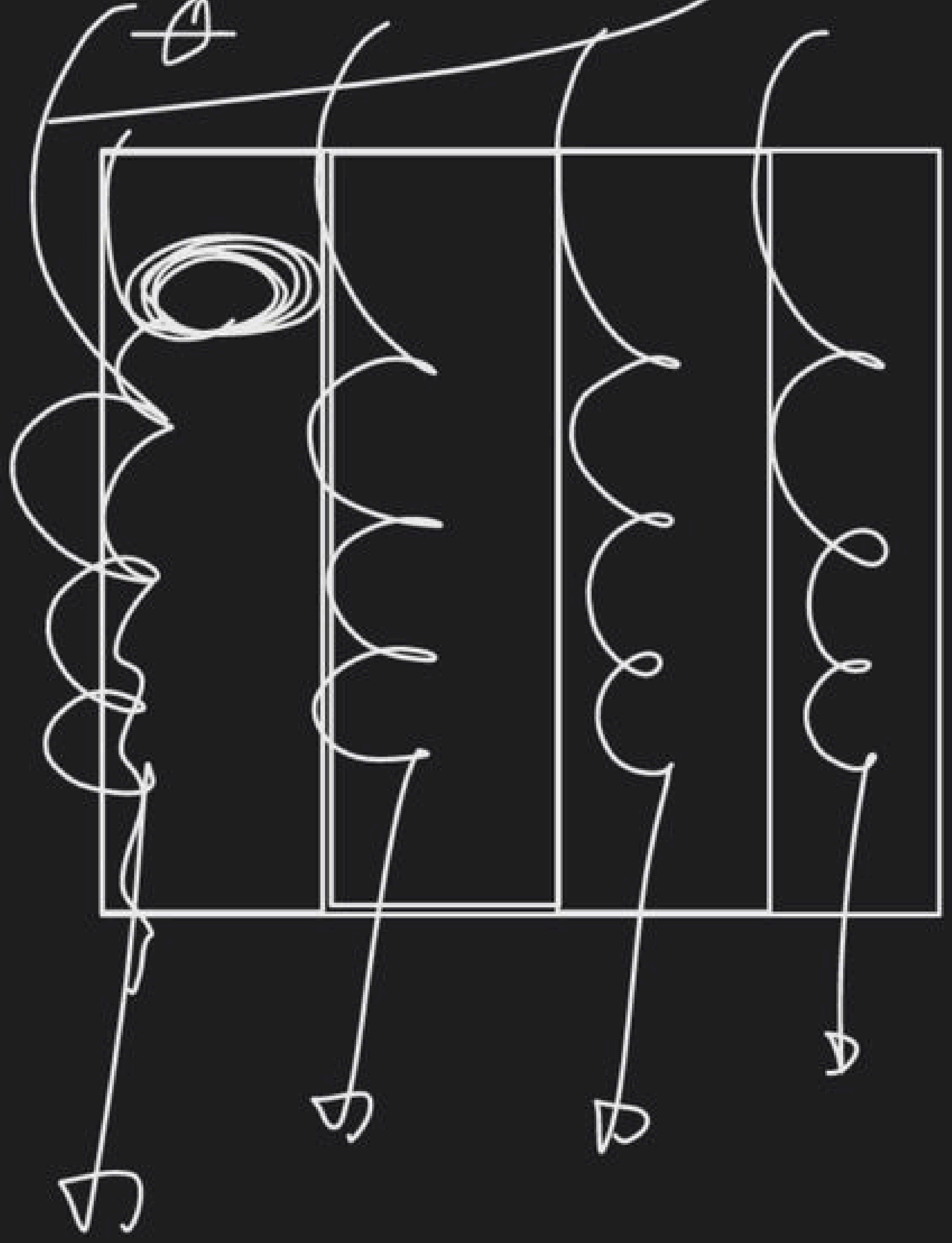
slides → setting → slid mode
↳ download

Video lag
Audio lag

$\rightarrow$ col wise sum

→ col wise

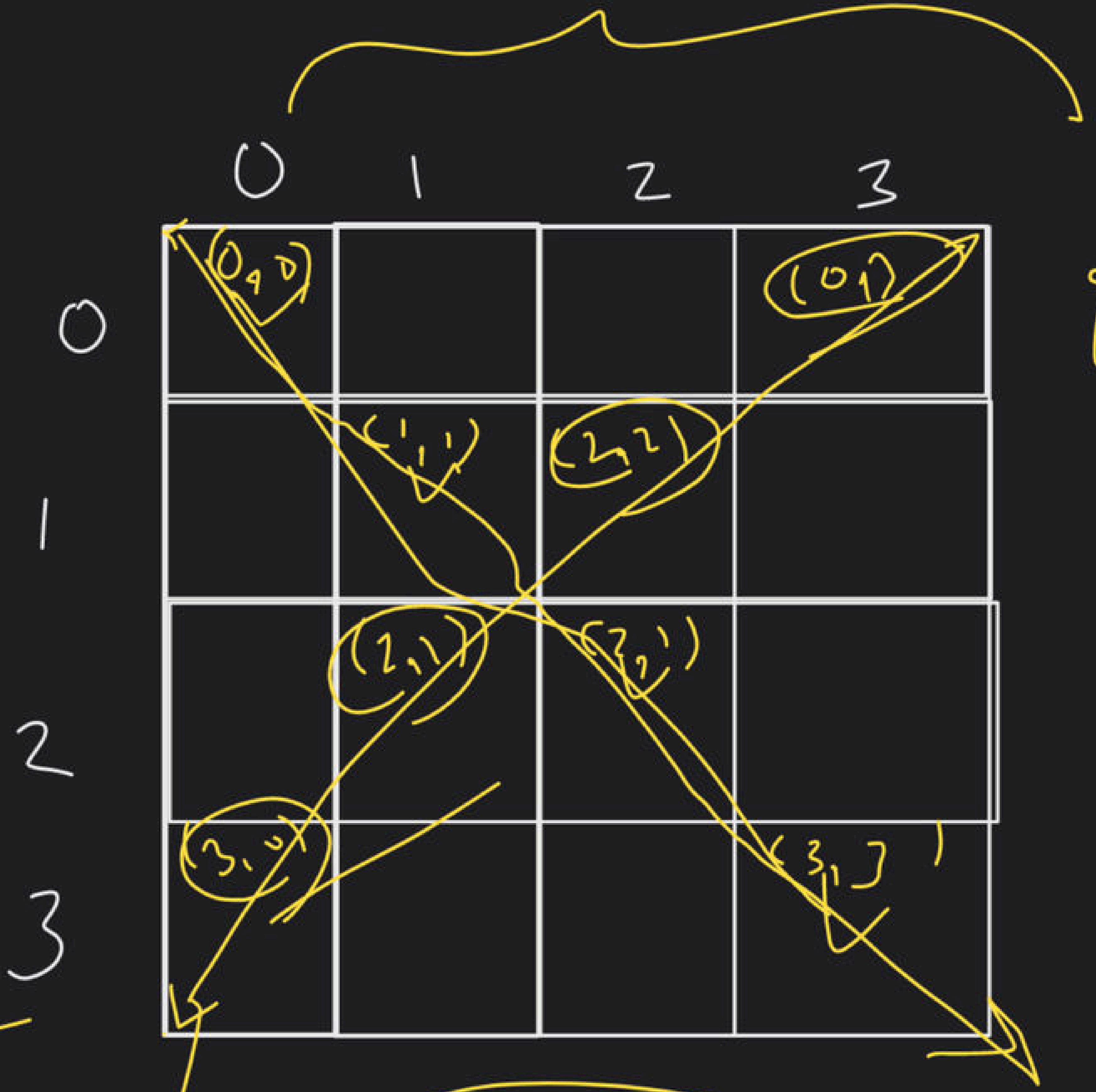
→ col wise sum



```
for (i → i<col )
{  int sum = 0
   for (j → j< row)
   {
      sum = sum +
            arr[j][i]
   }
   cout << sum;
}
```

Diagonal Print

```
for(i=0; i<row; i++)
{
cout << arr[i][i];
}
```



H/w -> second Diagonal print

```
sum=0
for( i=0; i<row; i++)
{
sum = sum +
        arr[i][i];
}
cout << sum;
```

Diagonal sum

Transpose of a Matrix

arr[0][0] ⟷ arr[0][0]
arr[1][0] ⟷ arr[0][1]
arr[2][0] ⟷ arr[0][2]
o/p
arr[2][1] ⟷ arr[1][2]

i/p

row

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 2 & 4 & 6 \\ 1 & 8 & 3 & 5 \\ 2 & 7 & 9 & 1 \end{array}$$

0,1   0   1,2   1,0   2,0   2,1

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 2 & 8 & 7 \\ 1 & 4 & 3 & 9 \\ 2 & 6 & 5 & 1 \end{array}$$

Dry Run

swapping

arr[i][j] ⟷ arr[j][i]

swap(arr[i][j],
arr[j][i])

(0,0)

arr[0][0] <-> arr[0][0]

(0,1)

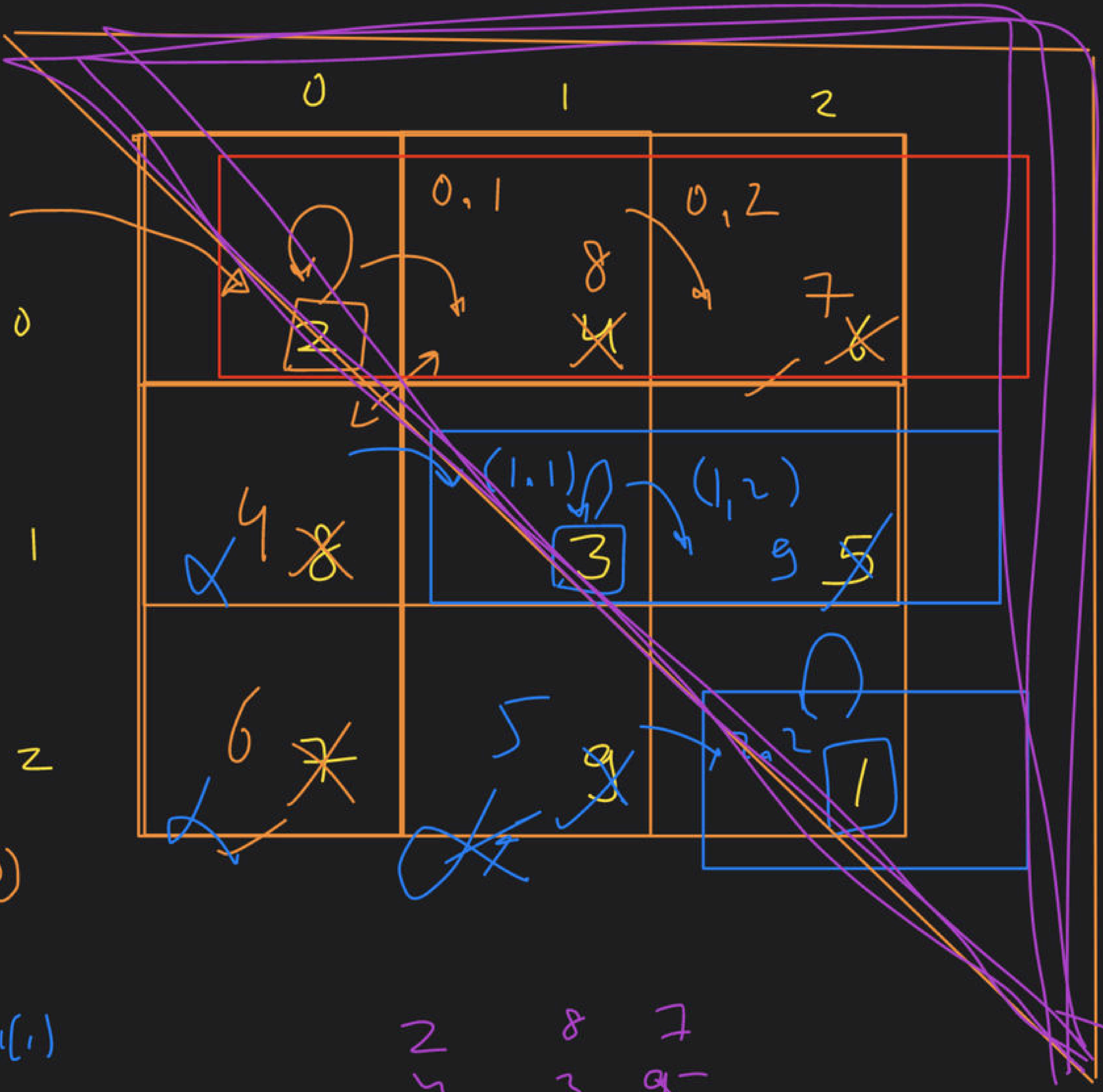
arr[0][1] <-> arr[1][0]

(0,2)

arr[0][2] <-> arr[2][0]

(1,1)

arr[1][1] <-> arr[1][1]



(1,2)

arr[1][2] <-> arr[2][1]

(2,2)

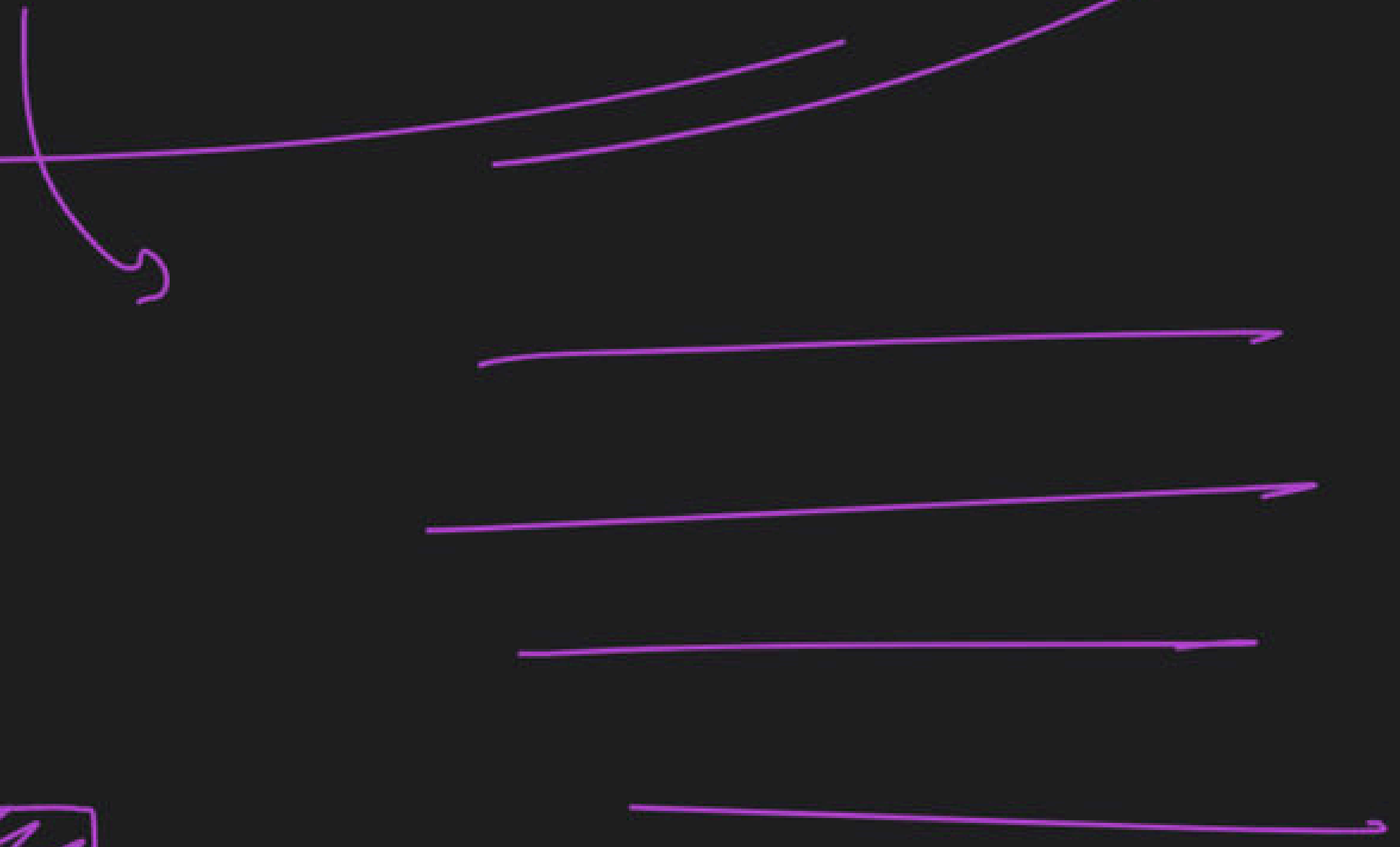
arr[2][2] <->
arr[2][2]

for loop

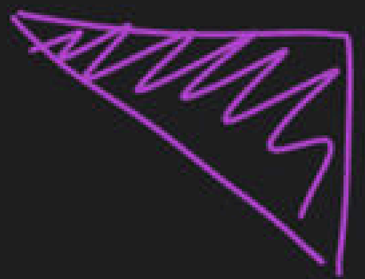
$$\begin{array}{ccc} 2 & 8 & 7 \\ 4 & 3 & 9 \\ 6 & 5 & 1 \end{array}$$

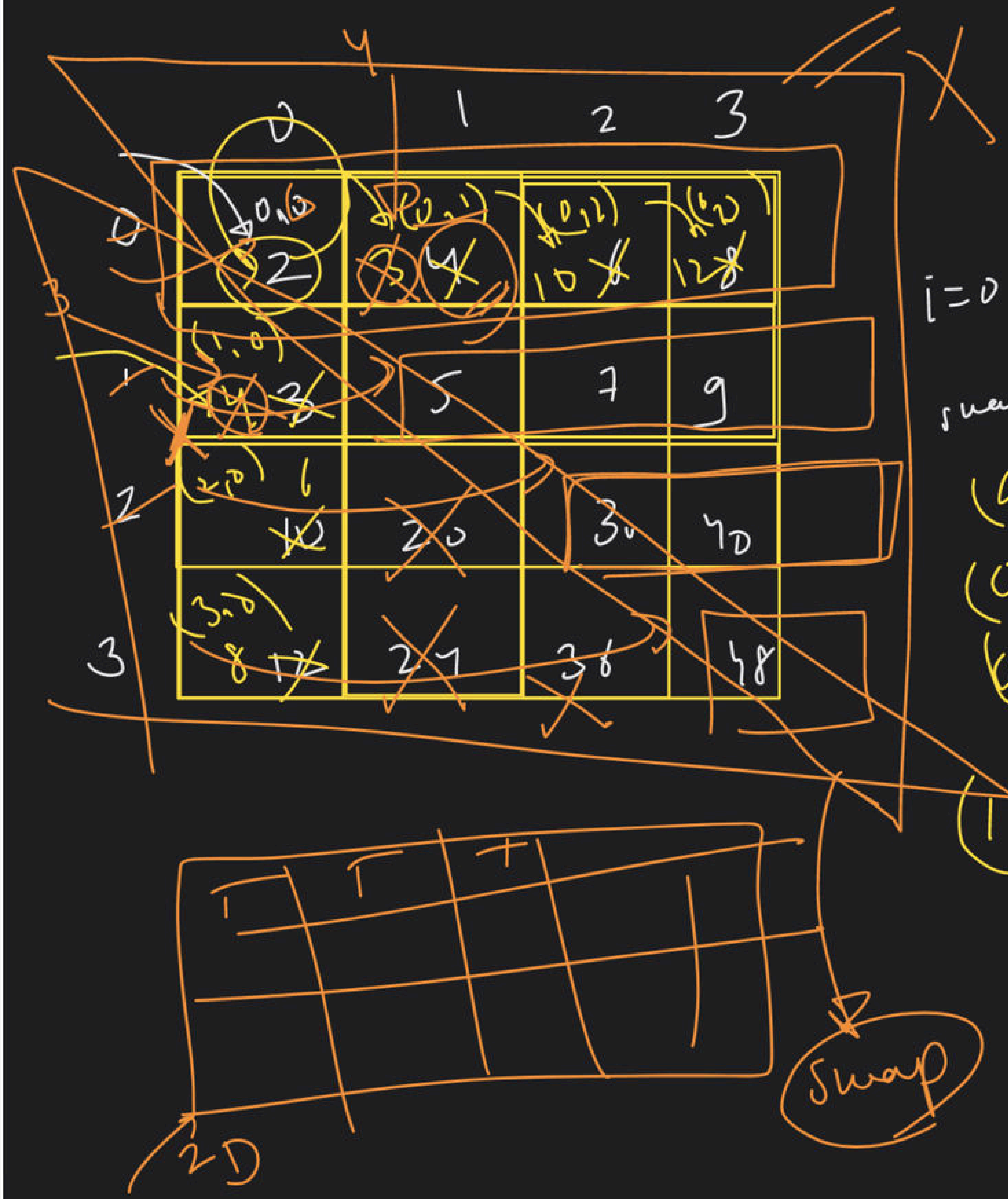
```
for (i=0) i<row; i++)

    for (j=0; j<col ;j++
```



```
for (i=0; i<row; i++)
{
    for (j=i ; j<col ,j++
        {
        }
```





Logic swap(arr(i)(j) ↔ arr(j)(i)).

i=0 j=0

swap

$(0,1) \longleftrightarrow (1,0)$

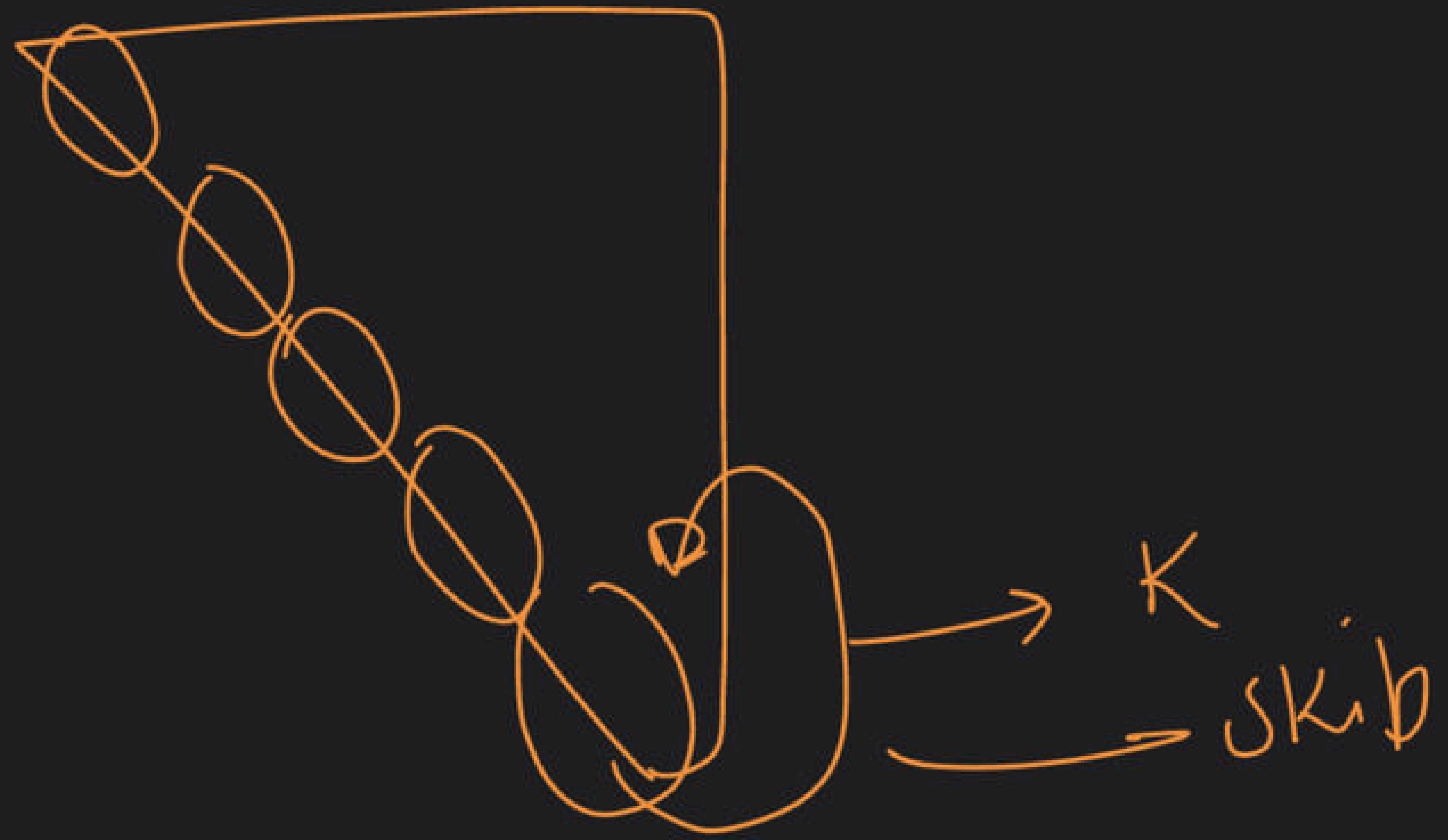
$(0,2) \longleftrightarrow (2,0)$

$(0,3) \longleftrightarrow (3,0)$

$(1,0) \longleftrightarrow (0,1)$ ~~+~~

```
for(i=0 → i<n)
{
  for(j=0; j<=i;)
  {

  }
}
```



Doubt

$\boxed{J == i?}$

$J = i$ ✓

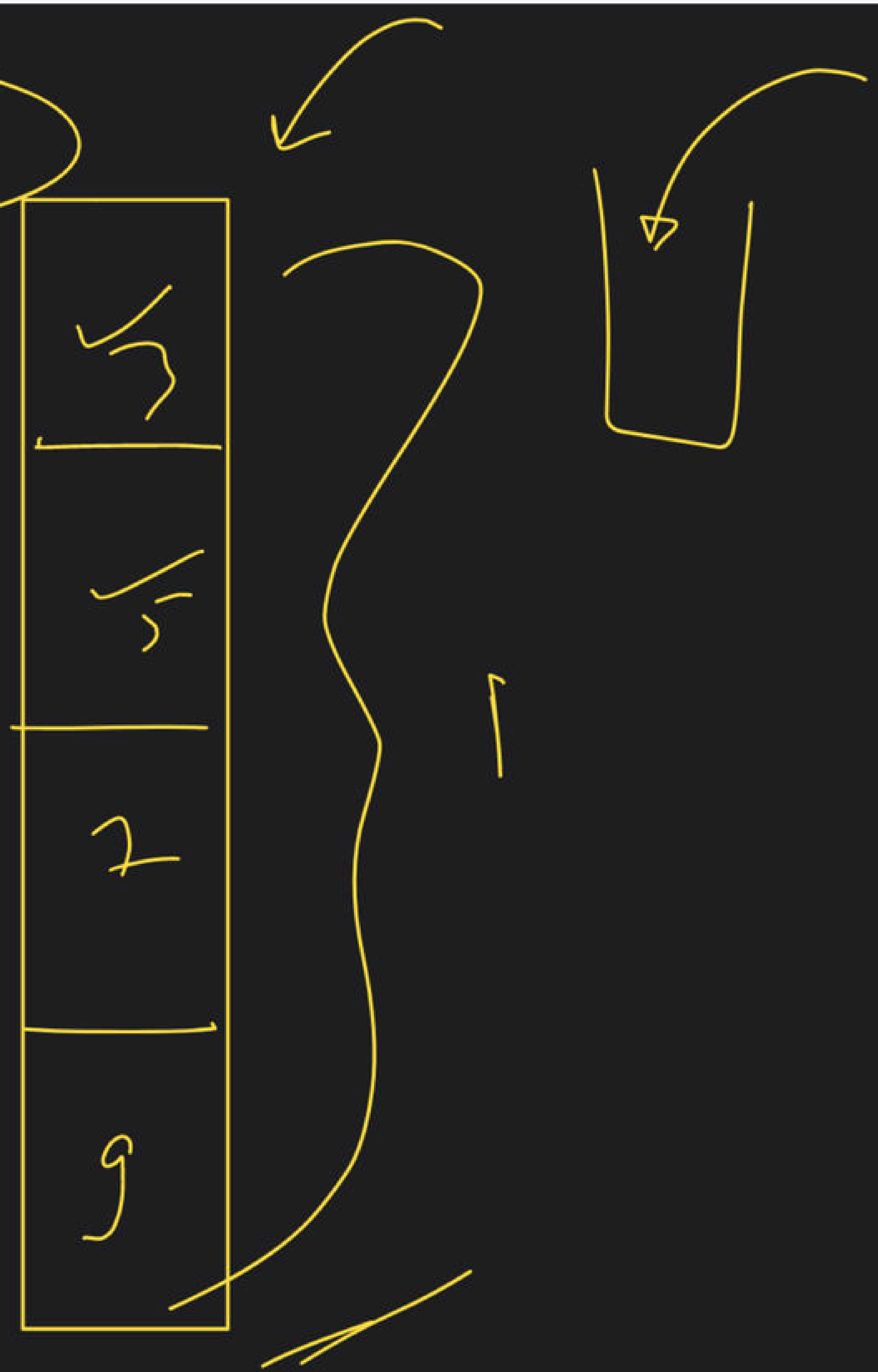
$\boxed{J = i+1}$

Vector → 2D

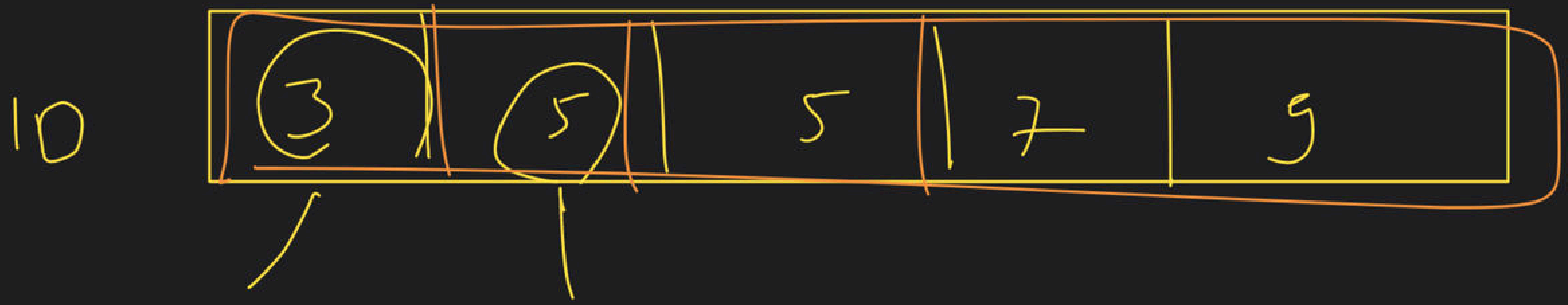
1D → vector<int> arr

2D → vector<vector<int> > arr

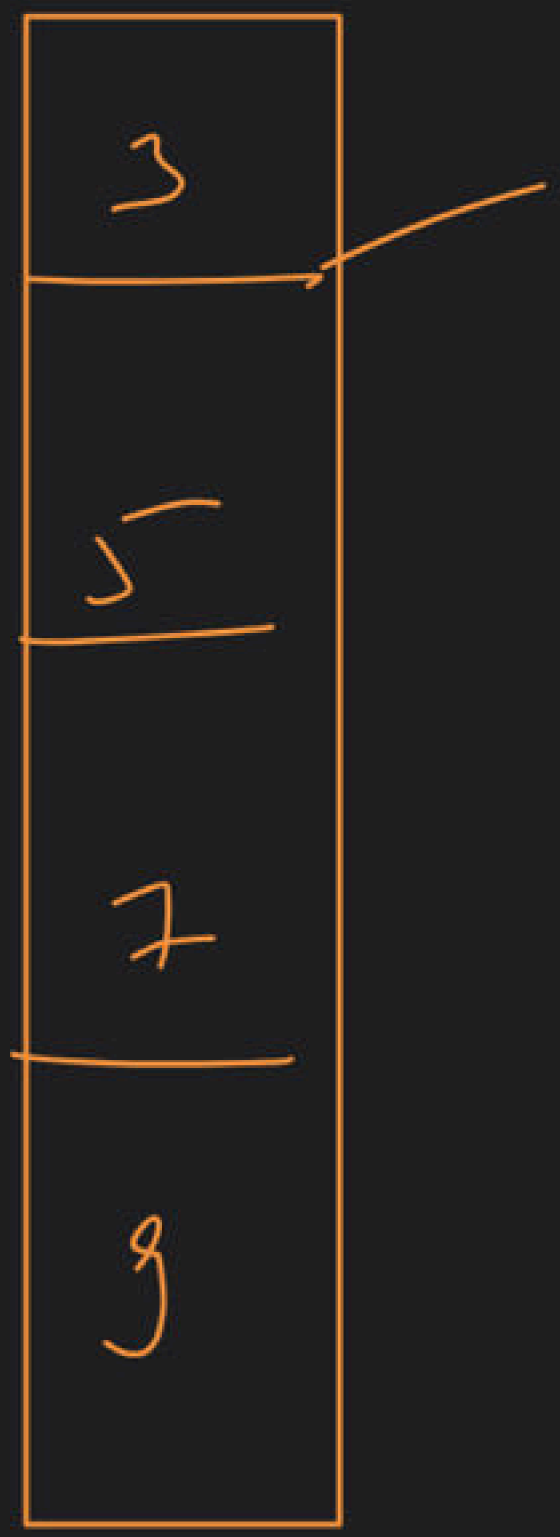
vector <int> v



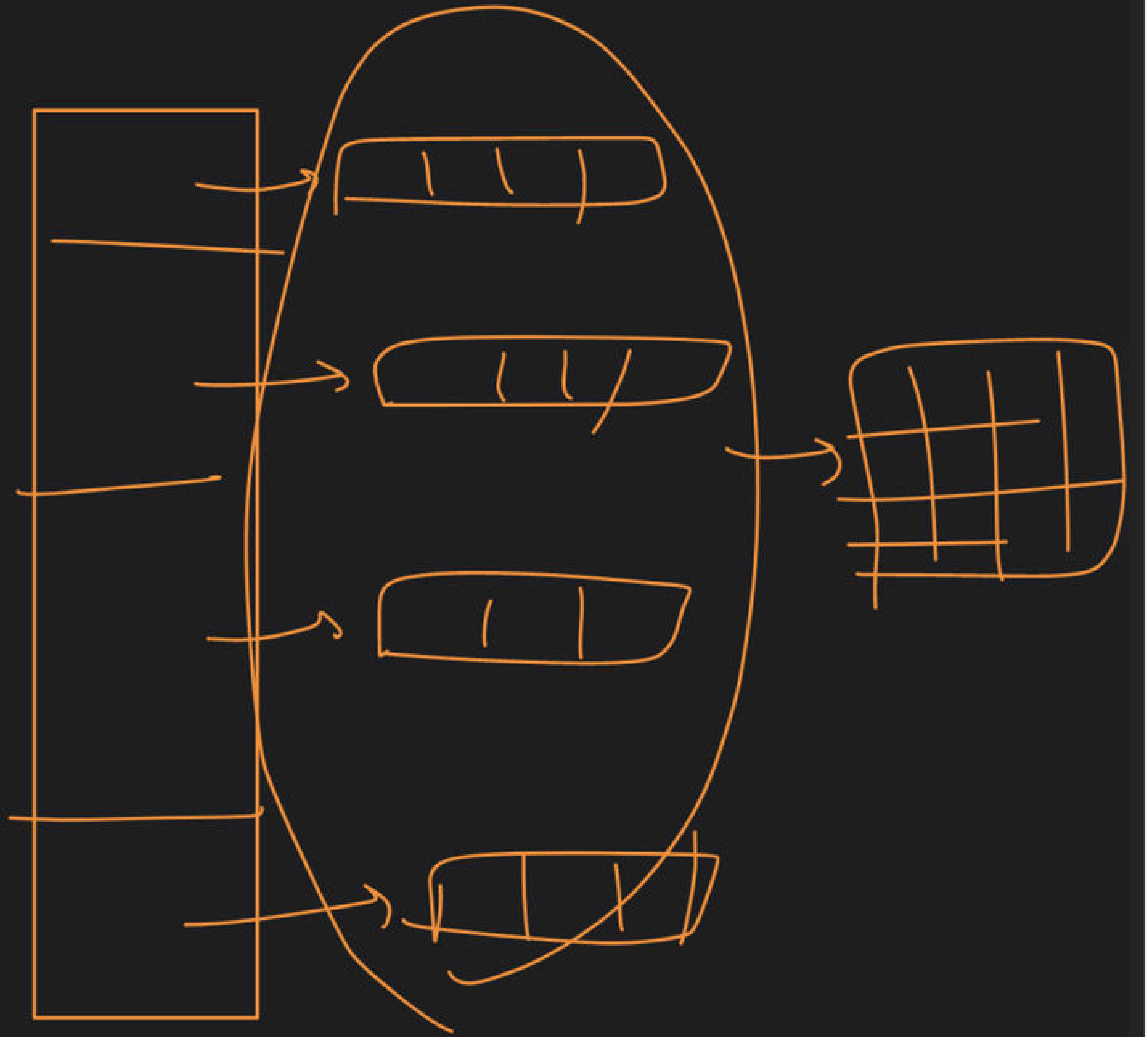
vector <int> v



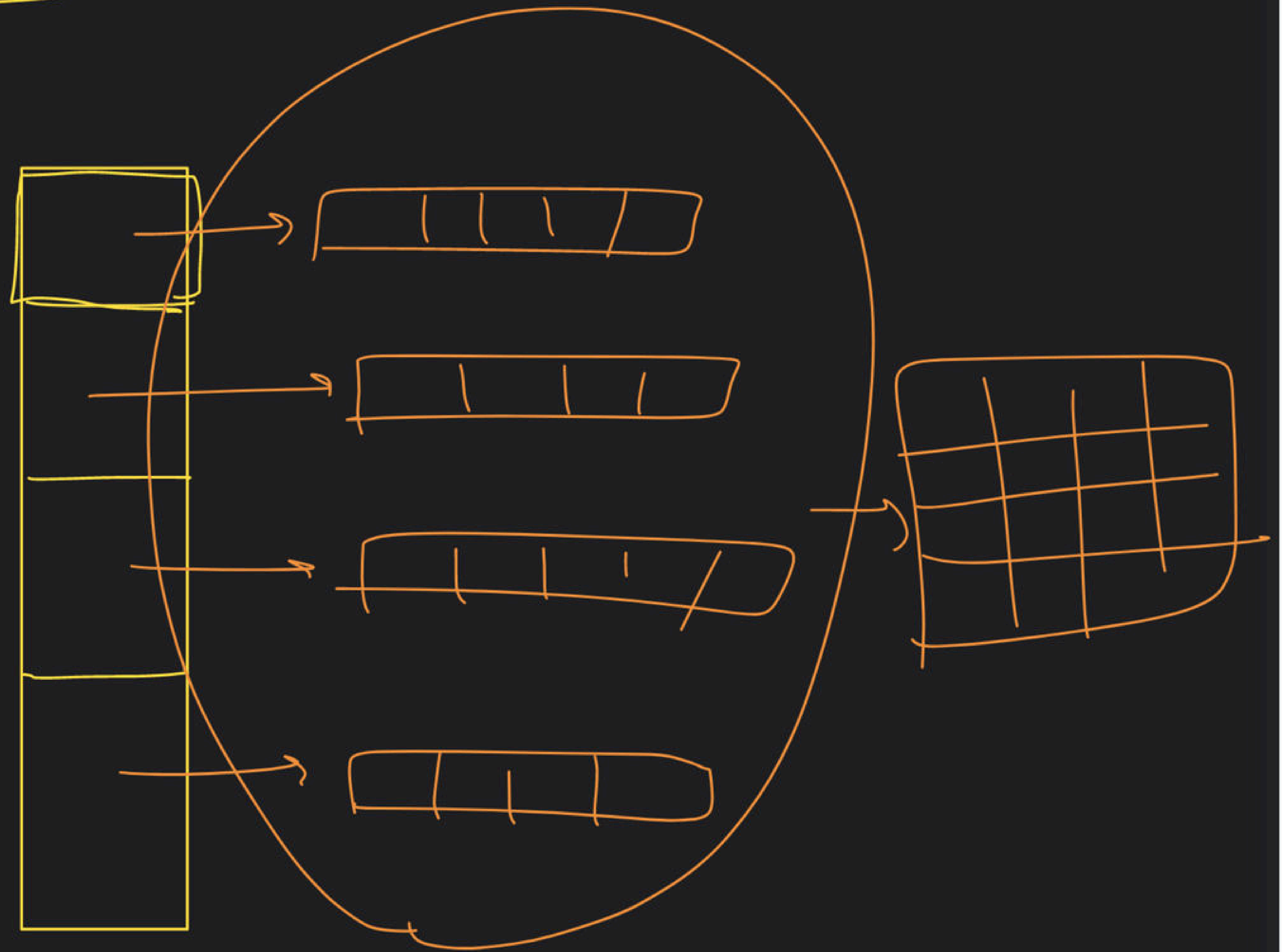
Vector <int>



Vector < vector <int> >



vector < vector < int > >



```
vector <vector <int> > arr
```

```
vector <vector <int> > arr (5, vector <int> (10, 0))
```

2D array

name

row size

row item
↳ initialize
↳ with a vector of size 10 that
↳ initialize with 0