Tommorow → 8-10 am 'Sat'

2D Arrays

& 2D ArrayLists

```
for(int i=2; i<=n; i*=i){
|   sout("GFG");
}
```

$i = 2, 4, 16, 256, (256)^2, \ldots \quad n$

$2^1, 2^2, 2^4, 2^8, 2^{16}, \ldots \quad 2^x$

$$\underbrace{1 \quad 2 \quad 4 \quad 8 \quad 16 \ldots x}_{y \text{ terms}}$$

$x = 2^y \Rightarrow y = \log(x)$

$2^x = n$

$x = \log_2 n$

$T.C. = O(y)$

$= O(\log x)$

$= O(\log(\log n))$

M-1

$$i \to 2,\ 4,\ 16,\ 256,\ (256)^2 \ \ldots\ \ldots\ n$$

$$2^1,\ 2^2,\ 2^4,\ 2^8,\ 2^{16}\ \ldots\ldots\ \boxed{n}$$

$$\underbrace{2^{2^0},\ 2^{2^1},\ 2^{2^2},\ 2^{2^3},\ 2^{2^4}\ \ldots\ldots}_{\text{total 'x' terms hai}}\ \boxed{2^{2^{x-1}}}$$

$$\Bigg\} \to \quad n = 2^{2^{x-1}}$$

$$2^{x-1} = \log_2(n)$$

$$x-1 = \log_2(\log_2 n)$$

$$x = \log_2(\log_2 n)+1$$

$$T.C. = O(x)$$

$$= O(\log\log n)$$

M-2

Contents

1. Representation, Creation & Indexing
2. Traversal
3. Questions
4. ArrayList of Arraylist
5. Advance Questions

Representation of 2D Array / Matrix

length

```
int[] arr = new int[5];
```

	0	1	2	3	4
arr			6		

arr[2] = 6

rows cols

```
int[][] arr = new int[4][5];
```

	0	1	2	3	4	→ j
0						
1						
2				21		
3						
↓ i						

arr 4x5

arr[2][3] = 21

Input & Output in 2D Arrays

Array of Arrays

	0	1	2	3	4
0	7	4	6	3	1
1	5	5	1	2	6
2	8	2	7	3	4

3x5

arr

arr.length = 3 (no. of rows)

arr[0].length = 5 (no. of cols)

arr = { {7, 4, 6, 3, 1}, {5, 5, 1, 2, 6}, {8, 2, 7, 3, 4} }

⇓ arr[0] ⇓ arr[1] ⇓ arr[2]

Ques: Sum of elements in given Matrix

HW: Find the maximum element in given 2D array.

Ques: Find the row with maximum sum

	0	1	2	3	4	5
0	2	3	5	4	3	4
1	3	3	5	4	2	2
2	1	1	3	5	1	2
3	1	1	2	5	4	4

4x6

arr

HW: Find the minimum element out of all the maximum elements of each row

	0	1	2	3	4	5	
0	2	3	5	4	3	4	5
1	3	3	5	14	2	2	14
2	1	11	3	5	1	2	11
3	1	1	2	5	41	4	41

4 x 6

arr

A.S. = O(1)

Ques: Print elements of 2D Array column-wise

(printing transpose)

	0	1	2	3	4	5 → j
0	2	3	5	4	3	4
1	3	3	5	14	2	2
2	1	11	3	5	1	2
3	1	1	2	5	41	4

↓ i

arr

4x6

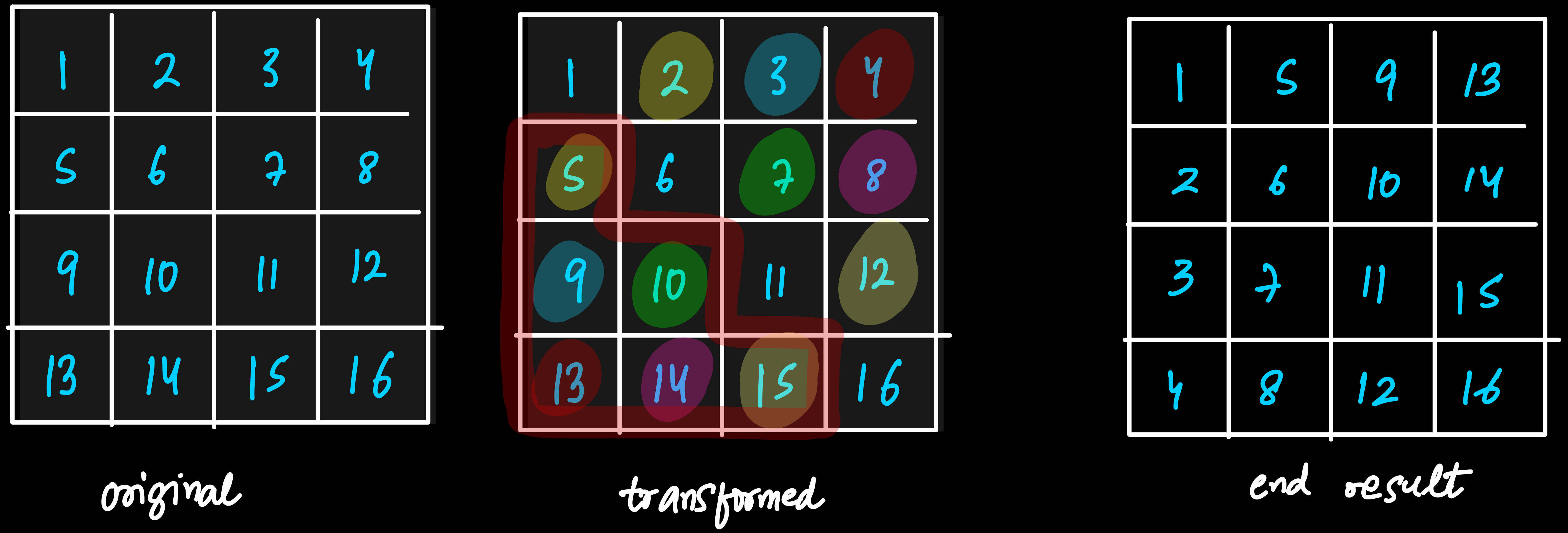
Output :

	0	1	2	3 → i
0	2	3	1	1
1	3	3	11	1
2	5	5	3	2
3	4	14	5	5
4	3	2	1	41
5	4	2	2	4

↓ j

Ques: Print matrix in Snake Pattern

Ques: Transpose of Matrix



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

original

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

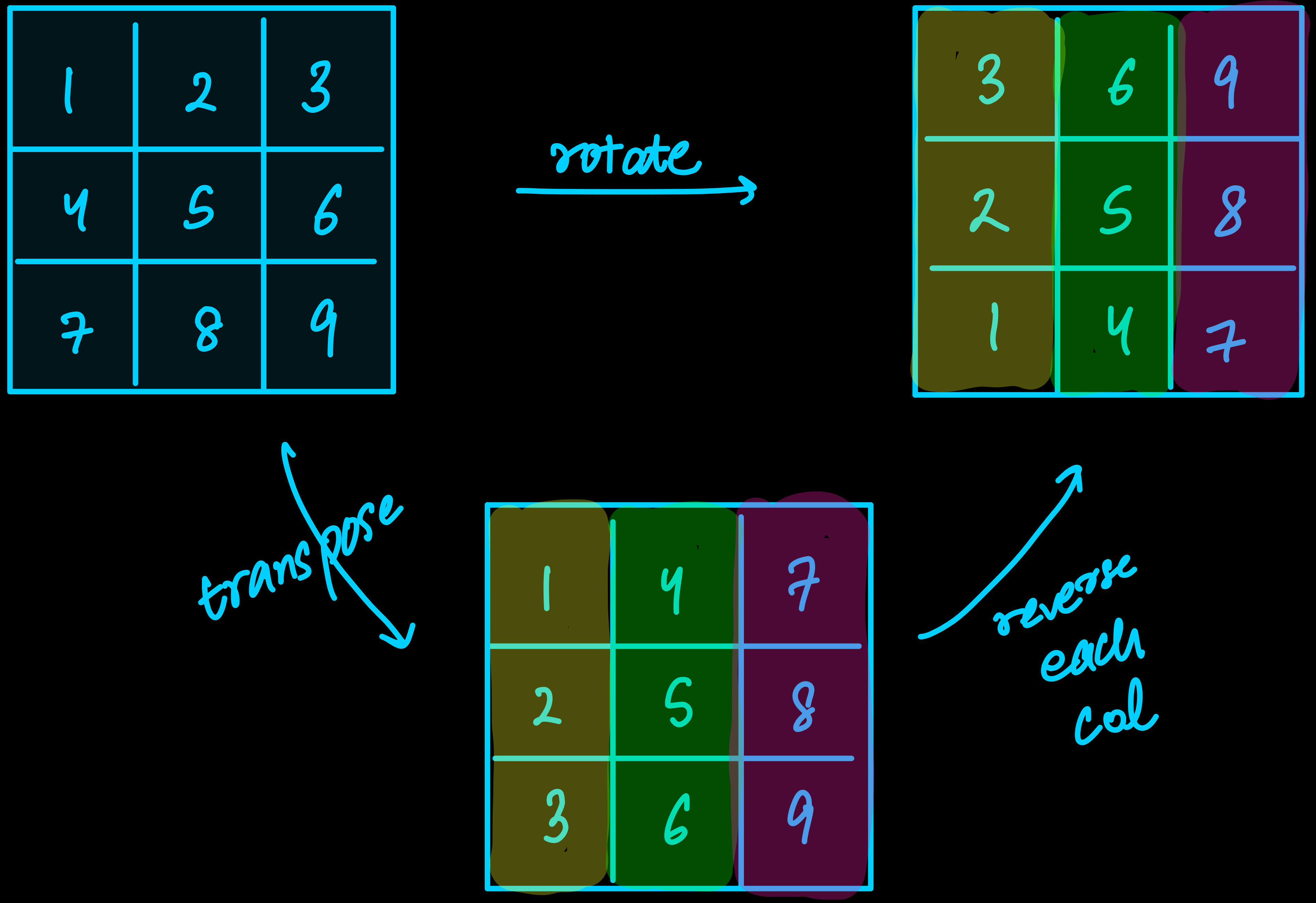
transformed

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

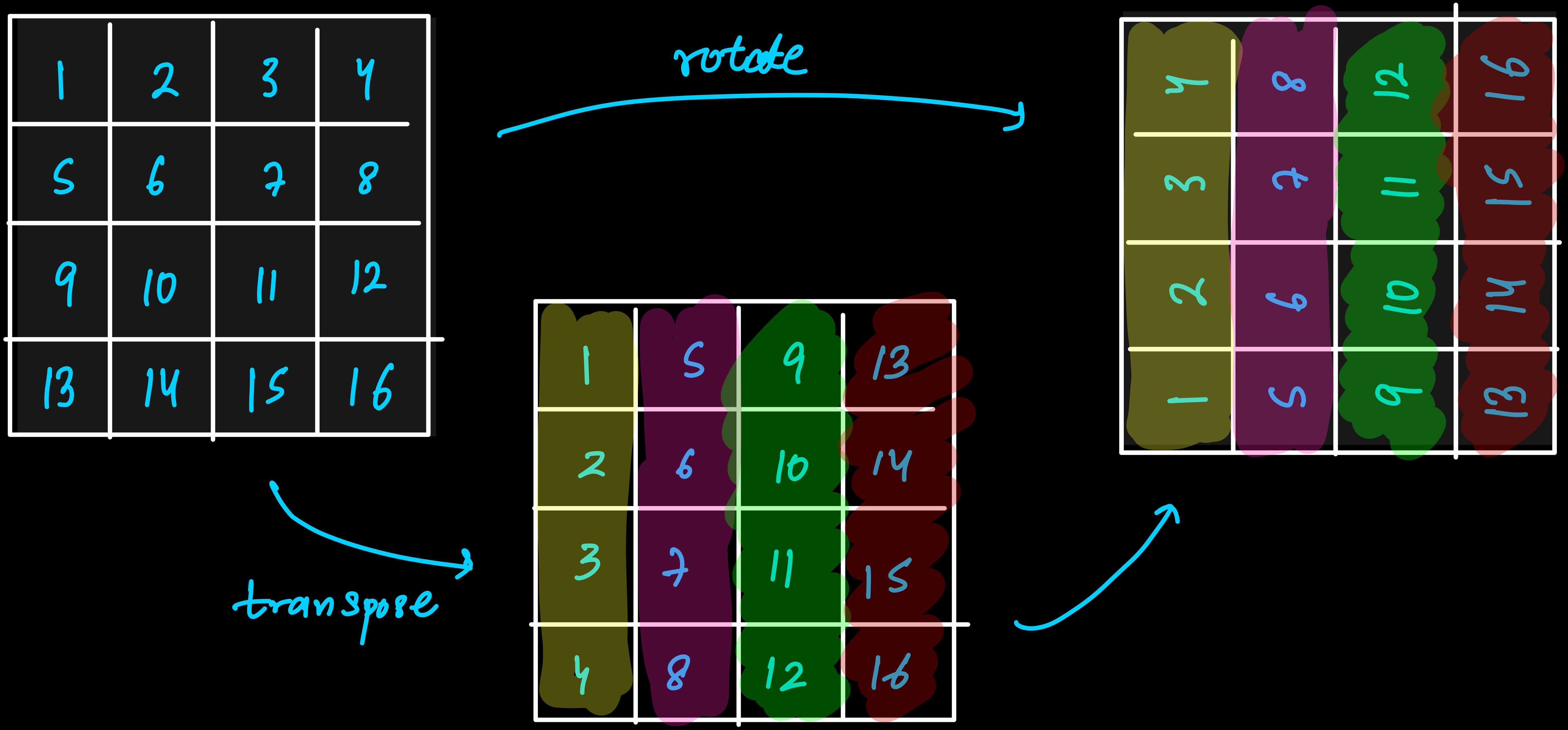
end result

$j<i$

Ques: Rotate by 90 degree



Ques: Rotate by 90 degree



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

4	8	12	16
3	7	11	15
2	6	10	14
1	5	9	13

2D ArrayLists

arr = { {1,2,3}, {1,2}, {1,2,3,4,5,6} }

1D Arraylist : ArrayList<Integer> al = new Arraylist<>()

2D Arraylist : Arraylist< Arraylist<Integer>> x = new Arraylist<>();

↓

x = { }

How ArrayLists Work Internally..

By Default → 1 capacity ka array banta hai □

arr.add(10)

10

arr.add(40)

10	40

arr.add(8)

10	40	8	49

arr.add(49)

arr.add(20)

10	40	8	49	20	4	1	7

arr.add(4)

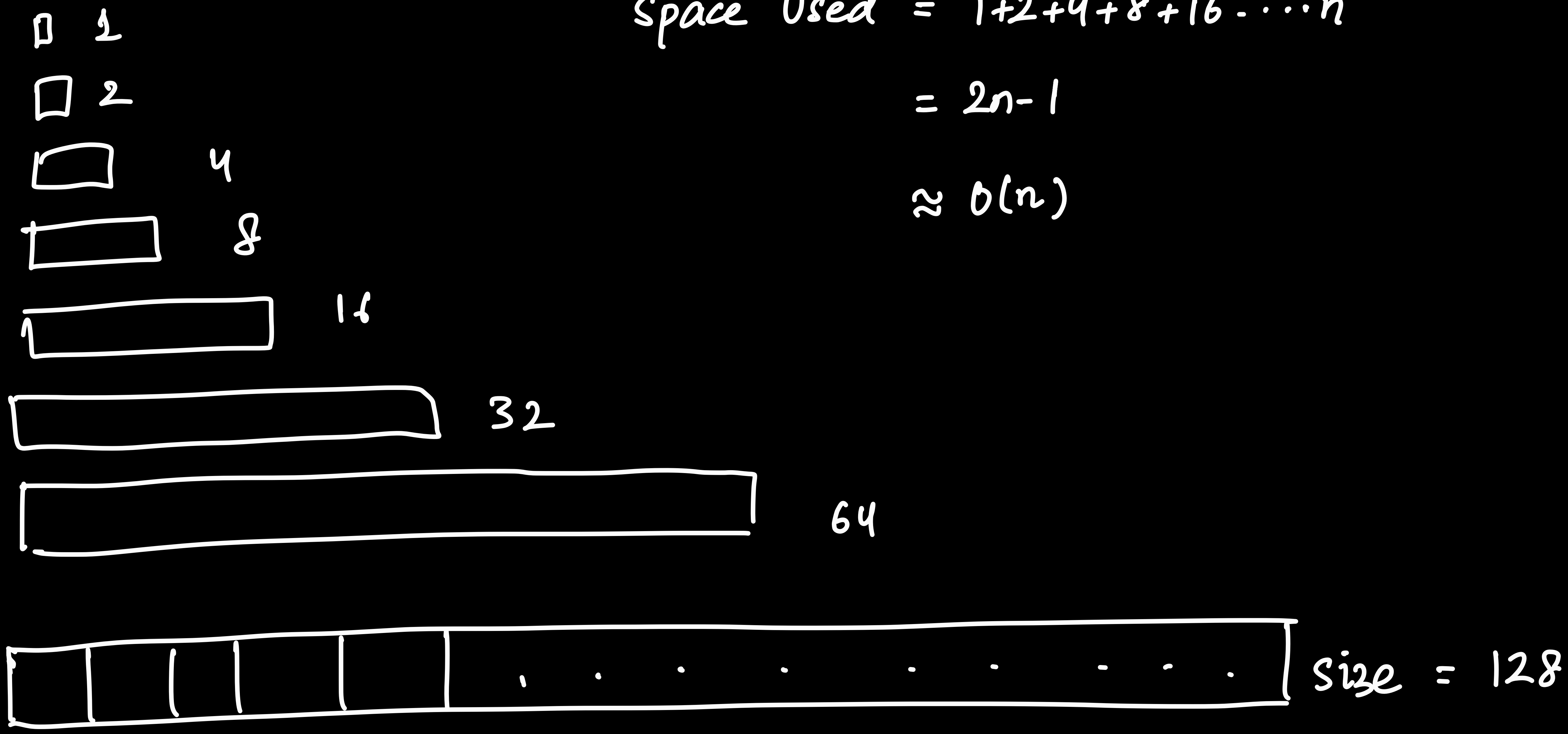
arr.add(1)   arr.add(7)

2D ArrayLists

$$\text{Space Used} = 1+2+4+8+16\cdots n$$

$$= 2n-1$$

$$\approx O(n)$$



2D ArrayLists

```
              0      1      2
arr = 0 {{100, 210},
      1  {5,   63,   90},
      2  {100, 210},
      3  {1000}}
```

change 90 to 89

2D array me → arr[1][2] = 89

2D AL me → arr.get(1).set(2, 89)

arr.get.(i).get(j) → for access

Ques: Pascal Triangle

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

→

$$
\begin{array}{c|cccccc}
 & 0 & 1 & 2 & 3 & 4 & 5 \\
\hline
0 & 1 & & & & & \\
1 & 1 & 1 & & & & \\
2 & 1 & 2 & 1 & & & \\
3 & 1 & 3 & 3 & 1 & & \\
4 & 1 & 4 & 6 & 4 & 1 & \\
5 & 1 & 5 & 10 & 10 & 5 & 1
\end{array}
$$

${}^{i}C_{j}$

arr = { {1}, {1, 1}, {1, 2, 1}, . . . }

Ques: Pascal Triangle

	0	1	2	3
0	0			
1	0	0		
2	0	0	0	
3	0	0	0	0

Step 1

n = 4

T.C. = $O(n^2)$

A.S. = $O(n^2)$

	0	1	2	3
0	1			
1	1	1		
2	1	2	1	
3	1	3	3	1

Step 2

```
if(j==0 or j==i)
    arr[i][j] = 1
else
    arr[i][j] = arr[i-1][j] + arr[i-1][j-1]
```

Ques: Pascal Triangle

n=6 1, 5, 10, 10 , 5, 1

$^5C_0$ $^5C_1$ $^5C_2$ $^5C_3$ $^5C_4$ $^5C_5$

fact

0	1	2	3	4	5
1	1	2	6	24	120

```
n--;
for(int r = 0; r<=n ; r++){
|     ans.add( ⁿCr );
}
```

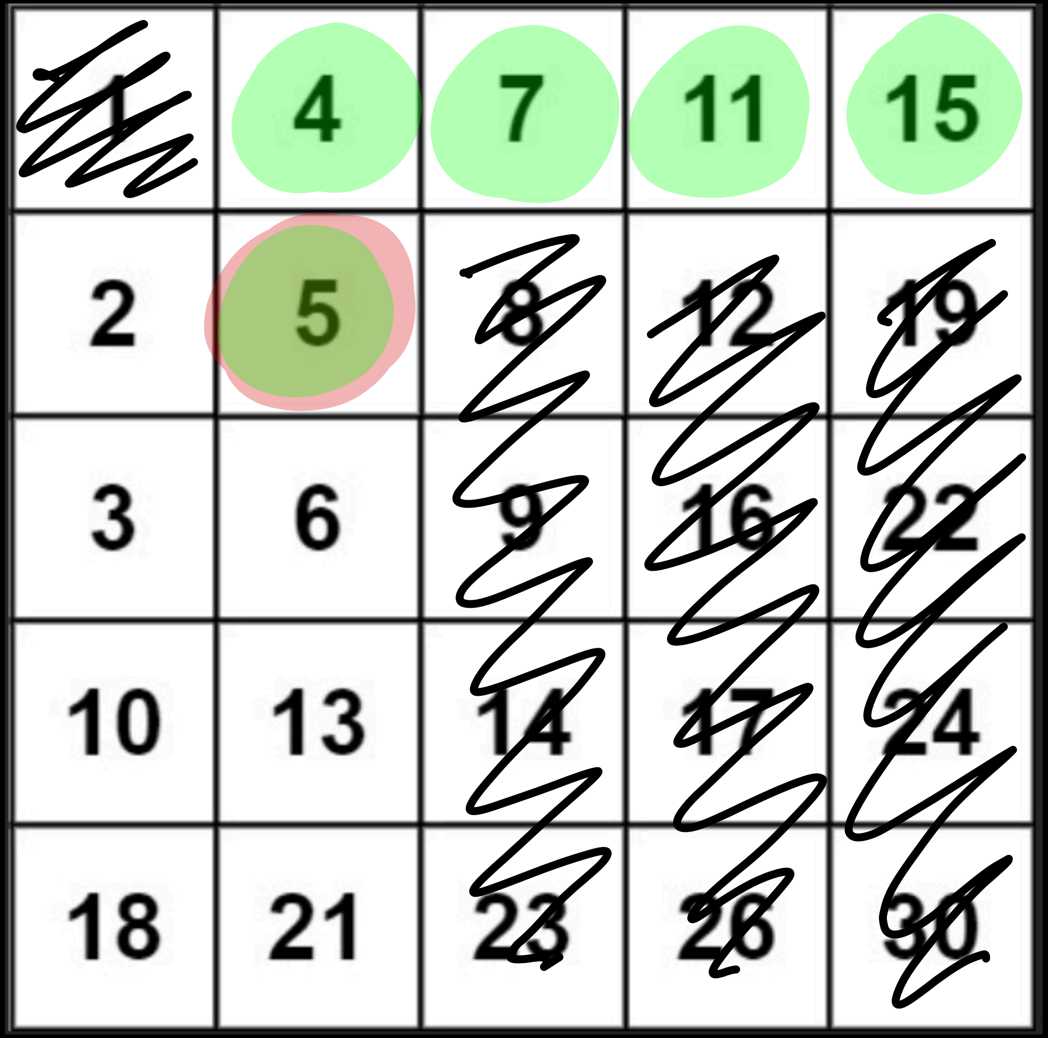
^nC_r ko fast kaise nikale

$$\boxed{^nC_{r+1} = \frac{n-r}{r-1} \cdot {}^nC_r}$$

T.C. = $O(n^2)$

A.S. = $O(n)$

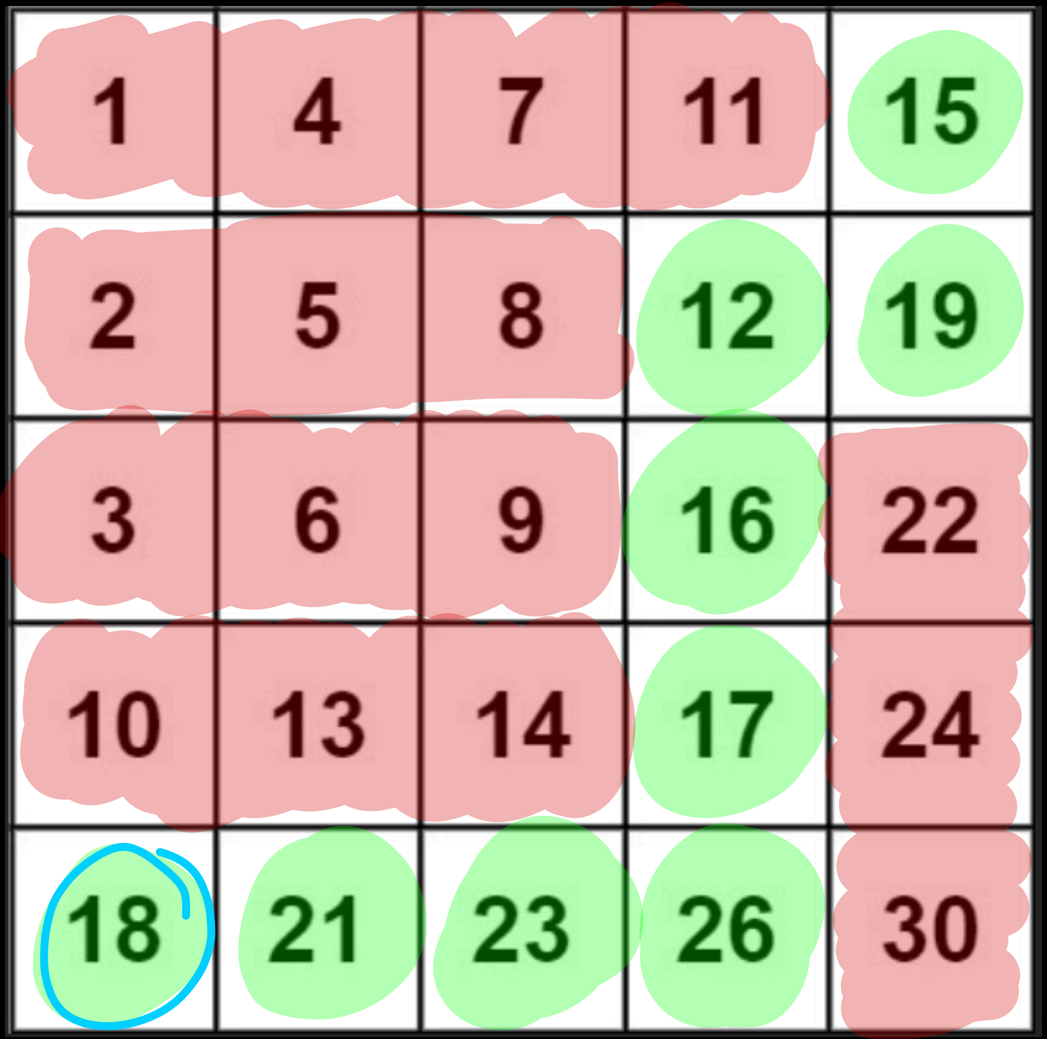
Ques: Search in a Row-Column Sorted matrix



~~1~~	4	7	11	15
2	5	~~8~~	~~12~~	~~19~~
3	6	~~9~~	~~16~~	~~22~~
10	13	~~14~~	~~17~~	~~24~~
18	21	~~23~~	~~26~~	~~30~~

target = 5

Ques: Search in a Row-Column Sorted matrix



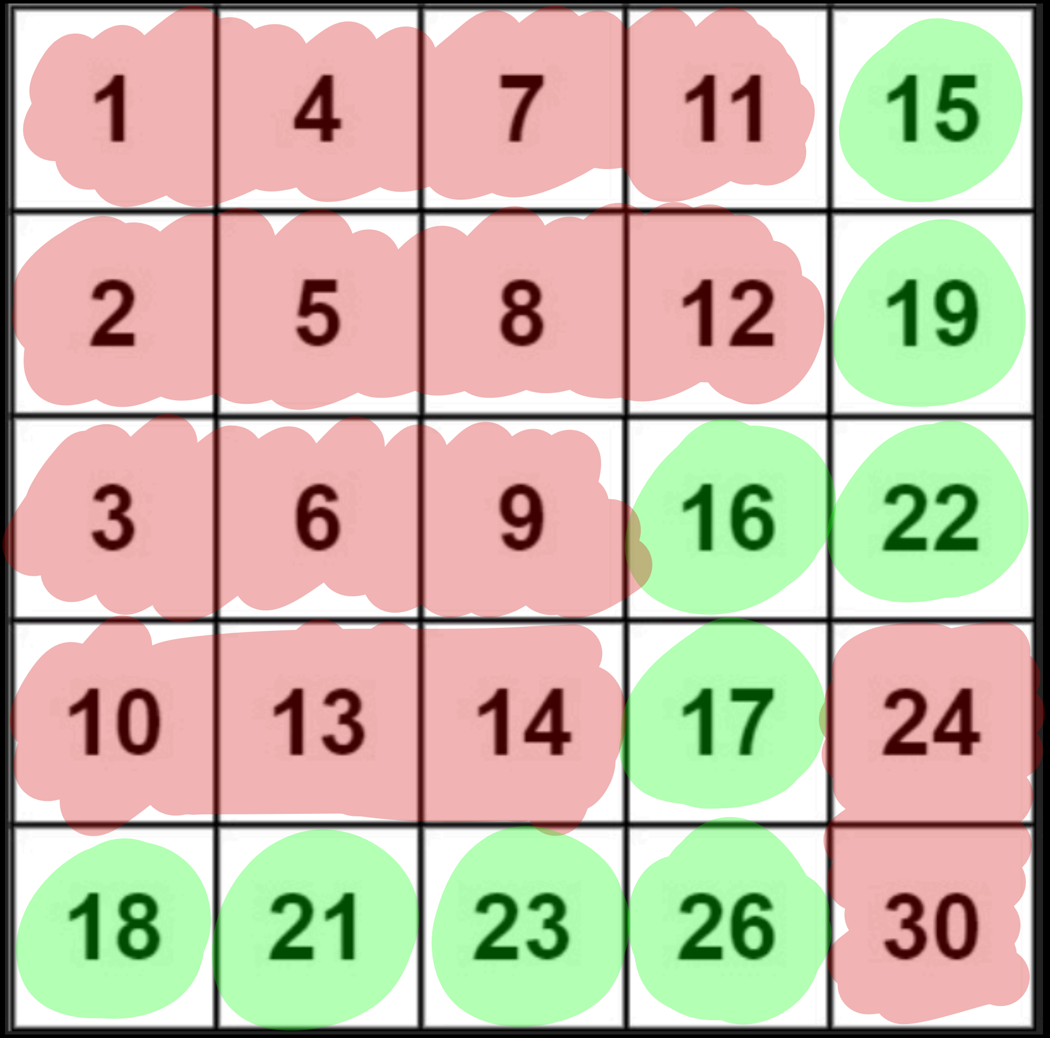
1	4	7	11	15
2	5	8	12	19
3	6	9	16	22
10	13	14	17	24
18	21	23	26	30

target = 18

if (arr[i][j] < target) go down

if (arr[i][j] > target) go left

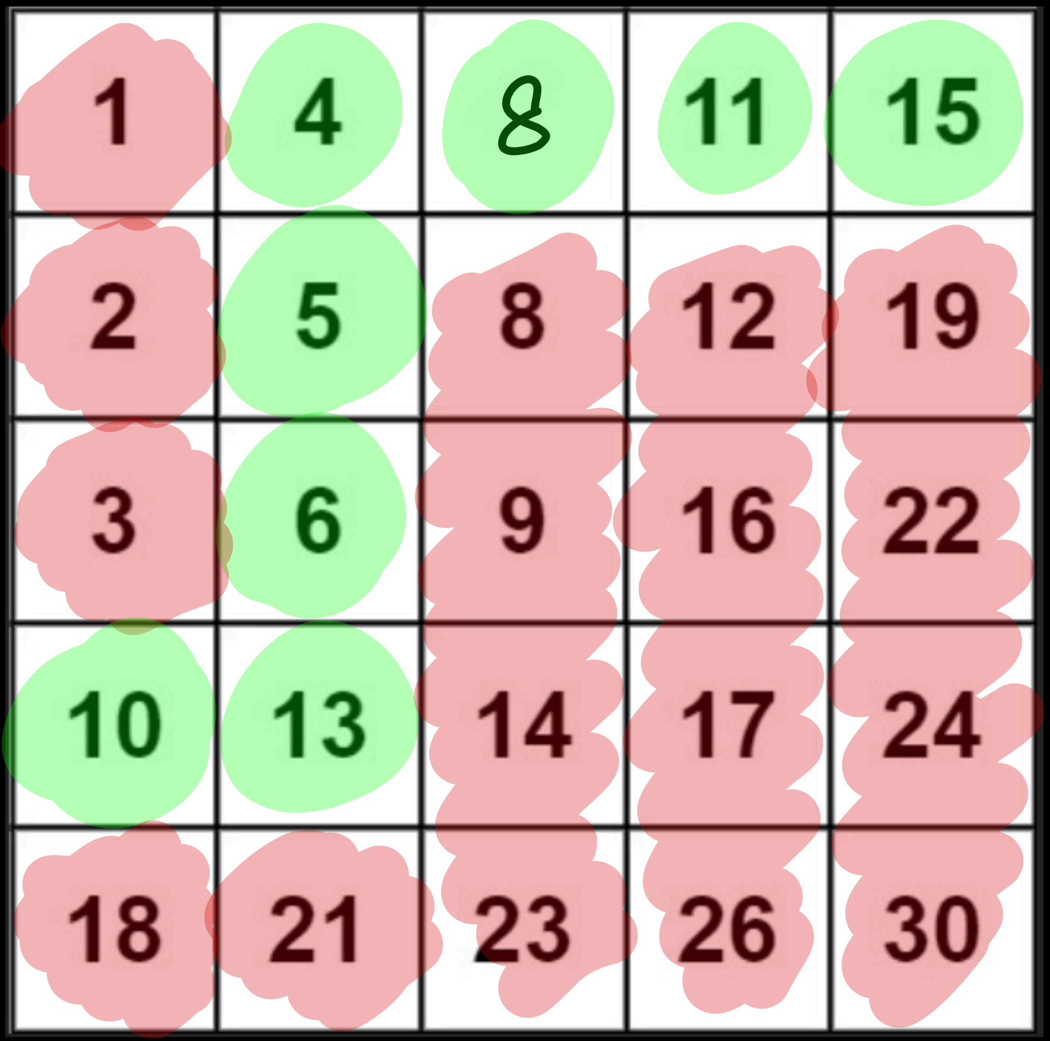
Ques: Search in a Row-Column Sorted matrix



1	4	7	11	15
2	5	8	12	19
3	6	9	16	22
10	13	14	17	24
18	21	23	26	30

target = 20

Ques: Search in a Row-Column Sorted matrix



1	4	8	11	15
2	5	8	12	19
3	6	9	16	22
10	13	14	17	24
18	21	23	26	30

target = 7

while (i<m && j>=0)

Ques: Set matrix zeroes

	0	1	2	3	4	5
0	3	6	2	1	7	3
1	8	0	-2	3	-5	-9
2	4	5	6	4	0	3
3	1	2	0	2	4	9

	0	1	2	3	4	5
0	3	0	0	1	0	3
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0

1st row
1st col

2nd row
4th col

3rd row
2nd col

0	0	0
4	0	0
0	0	0

arr

1	0	7
4	2	5
6	3	0

copy

Approach 1 → make a Same to Same copy.

traverse in original array, make changes in duplicate array

↓

T.C. = $O(m*n*(m+n))$

A.S. = $O(m*n)$

THANKYOU

Cuties