Question:

Given a row wise and column wise sorted matrix, find out whether element **k** is present or not.

A :

	0	1	2	3
0	-5	-2	1	13
1	-4	0	3	14
2	-3	2	6	19

K = 13 True

K = 2 True

K = 15 False

Brute Force

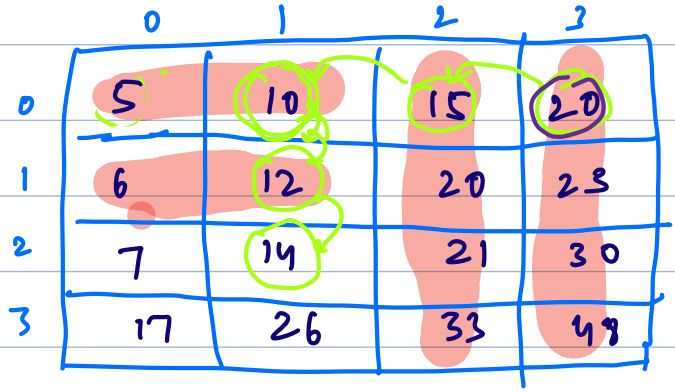
Iterate over entire matrix and search for K

T.C: O(N·M)

S.C: O(1)

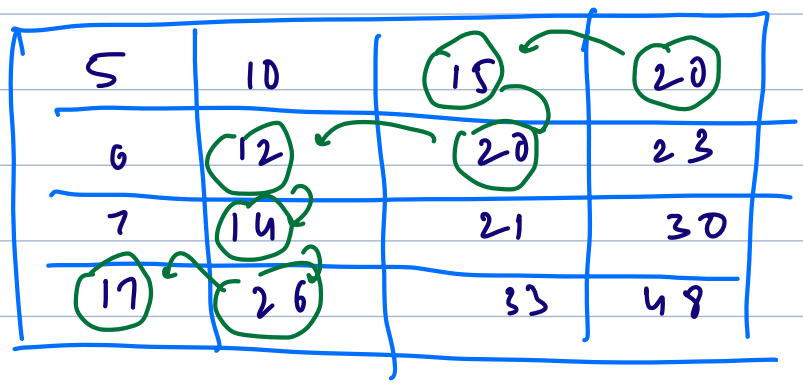
Efficient Approach:

```
if( A[row][col] == k)
    return True
else if( A[row][col] > k){
    col--
}
else
    row++
```

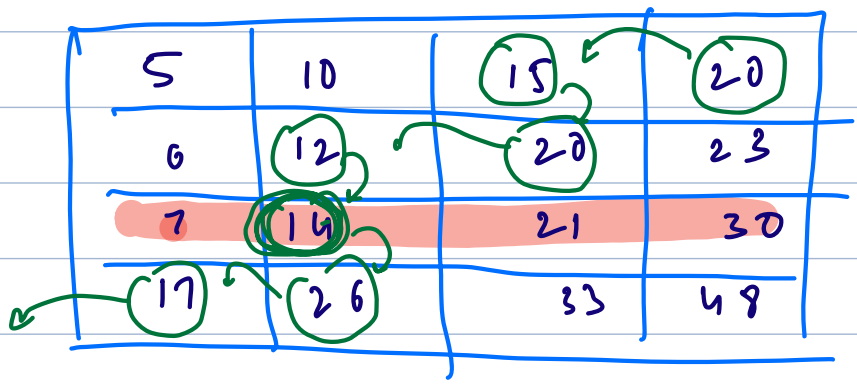


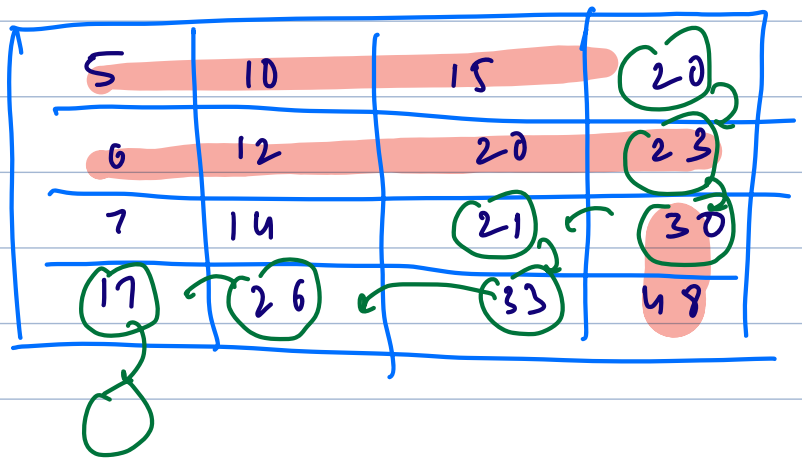
k = 14 ⇒ True

k = 17



k = 16 ✓



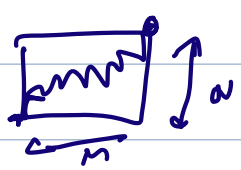


Search for target in Sorted Matrix

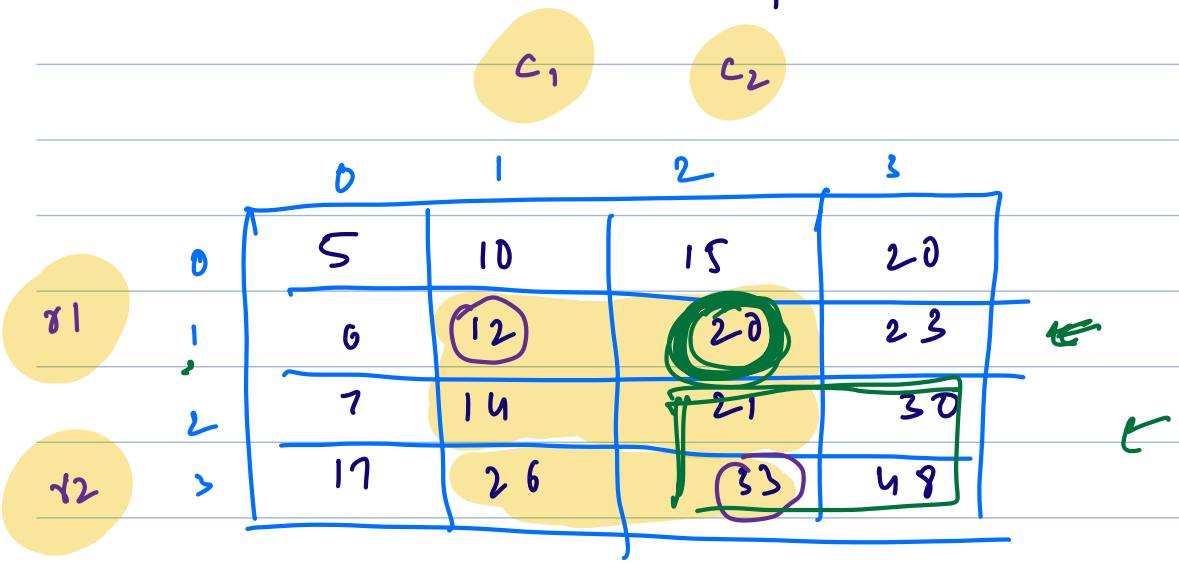
```
bool searchInMatrix(int arr[n][m], int target){
      row = 0, col = m-1;

      while(row < n && col >= 0){
            if(a[row][col] == target)
                  return true;
            else if(a[row][col] < T){
                  row++;
            }
            else
                  col--;
      }
      // If element is not found, return false
      return false;
}
```

T-C: O(N+M)

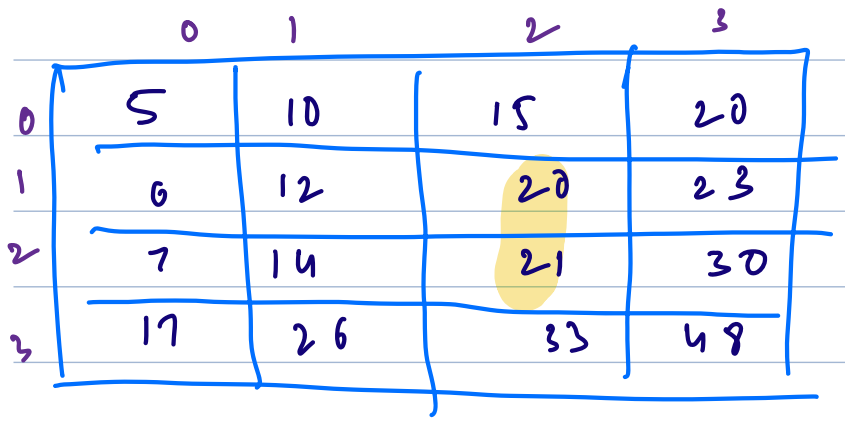
S.C: O(1)



Submatrix: Contiguous part of a matrix.
A continuous set of rows and cols



→ $r_1 = 1$ $c_1 = 1$ (Top-Left)
$r_2 = 3$ $c_2 = 2$ (Bottom-Right)



To define a submatrix: r_1, c_1, r_2, c_2

Question: Find sum of all submatrices.

$$A = \begin{bmatrix} 4 & 9 & 6 \\ 5 & -1 & 2 \end{bmatrix}$$

[4]=4 [9]=9 [6]=6 [5]=5 [-1]=-1 [2]=2

[4 9] [9 6] [5 -1] [-1 2]

13 15 4 1

[4 9 6] [5 -1 2]

19 6

$\begin{bmatrix} 4 \\ 5 \end{bmatrix}$ $\begin{bmatrix} 9 \\ -1 \end{bmatrix}$ $\begin{bmatrix} 6 \\ 2 \end{bmatrix}$

9 8 8

$\begin{bmatrix} 4 & 9 \\ 5 & -1 \end{bmatrix}$ $\begin{bmatrix} 9 & 6 \\ -1 & 2 \end{bmatrix}$

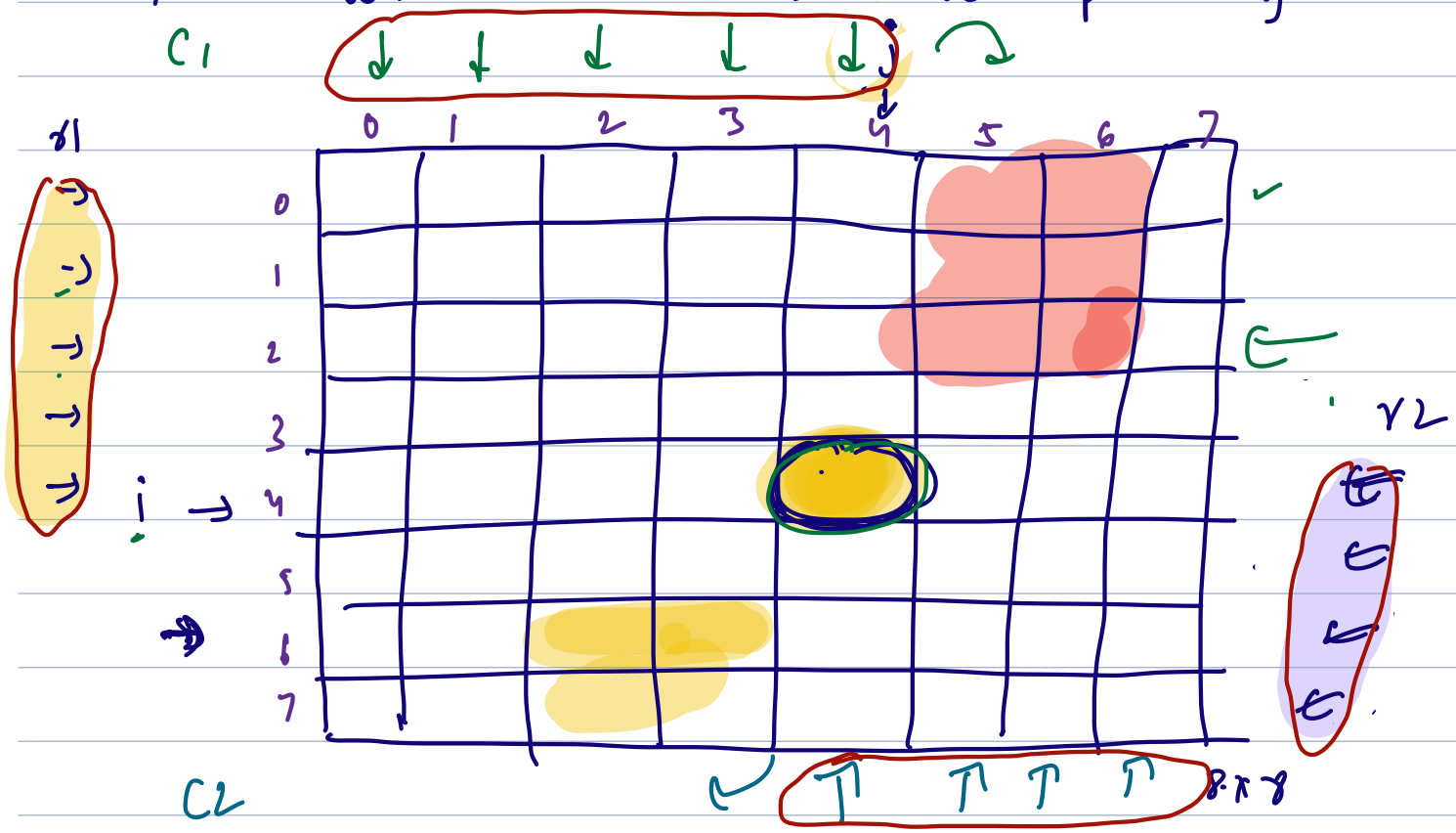
17 16

$\begin{bmatrix} 4 & 9 & 6 \\ 5 & -1 & 2 \end{bmatrix}$

25

Approach: Contribution Technique

For every A[i][j], find # of submatrices in which it is a part of.



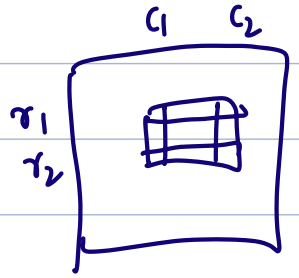
To define a submatrix:

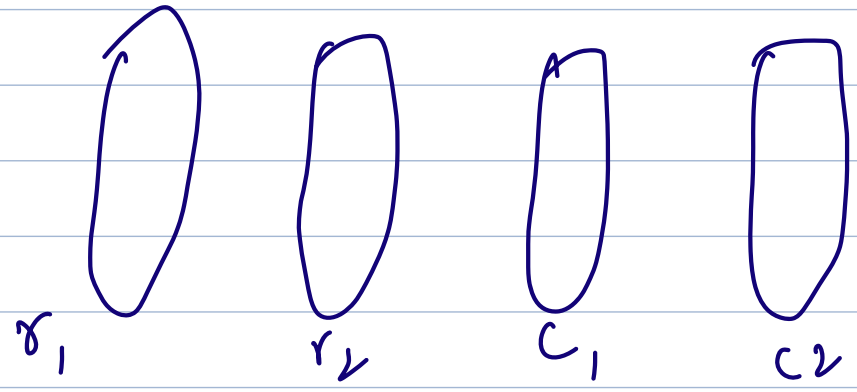
R1: [0, i] => (i+1)

R2: [i, N-1] (N-i)

C1: [0, j] (j+1)

C2: [j, M-1] (M-j)





count = $(i+1)(N-i)(j+1)(M-j)$

```
for (i = 0 → N-1) {
    for (j = 0 → M-1) {
        count = (i+1)(N-i)(j+1)(M-j)
        ans += count * A[i][j];
    }
}
return ans;
```

T.C: $O(N \cdot M)$

S.C: $O(1)$

8:24

Question: Given an unsorted array of positive integers, find the first missing natural numbers.

A = [3 7 2 1 2 6] => 4
0 1 2 3 4 5

A = [9 2 6 4 8 1 3] => 5
[1 2 3 4 6 7 9]

A = [3 4 5 8 3 7 8 4 7] => 1

A = [4 2 1 3] => 5
4

A = [1 2 5 6 4 3] => 7
6

=> A = [2 1000, 4000] Ans: [1, N+1]

Brute Force

```
for( i=1; i<=N+1; i++){
    // check if i exists        => O(N)
}
```

T.C: $O(N^2)$

S.C: $O(1)$

Approach 2 : Sort the array

→ Iterate and find the 1st missing natural number

T.C: O(N log N)

S.C: O(1)

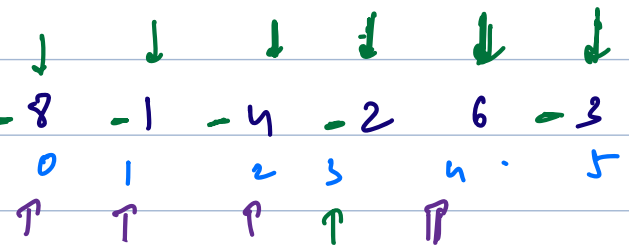
Approach 3 :

Ans : [1, N+1]

→ If any number in [1, N] is not present, then that missing number will be the answer

→ If all numbers in [1, N] are present, then N+1 will be the answer

1, 2, 3 ..... N, N+1



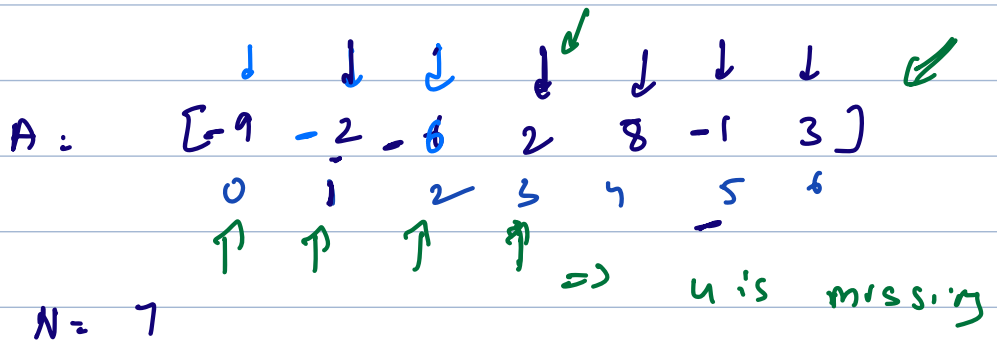
=> 5 is missing

N = 6

Ans : [1, 7] ✓

0 : Mark the presence of 1

1 : " " " " 2



N = 7

Ans: [1, 8]

Index 0 : 1

1 : 2

2 : 3

```
for(i -> 0 to N - 1) {
    ele = absolute(A[i]);

    if(ele >= 1 and ele <= N) {
        int idx = ele - 1;
        A[idx] = -1 * absolute(A[i]);
    }
}
```

```
for(i -> 0 to N - 1) {
    if(A[i] > 0) return i + 1;
}
return N + 1;
```

Question: We can have negative numbers also

A: [-3 4 2 9] ✓ Ans: [1, 5]

Will negative ever be my answer? NO!

Should we make them +ve? NO!

→ Change all -ve numbers to a value which is never the answer

→ May be change it to N+2

A = [0 1 4 3]

```
for(i -> 0 to N - 1) {
    if(A[i] <= 0) {
        A[i] = N + 2;
    }
}

for(i -> 0 to N - 1) {
    ele = abs(A[i]);

    if(ele >= 1 && ele <= N) {
        idx = ele - 1;
        A[idx] = -1 * abs(A[i]);
    }
}

for(i -> 0 to N - 1) {
    if(A[i] > 0) return i + 1;
}
return N + 1;
```

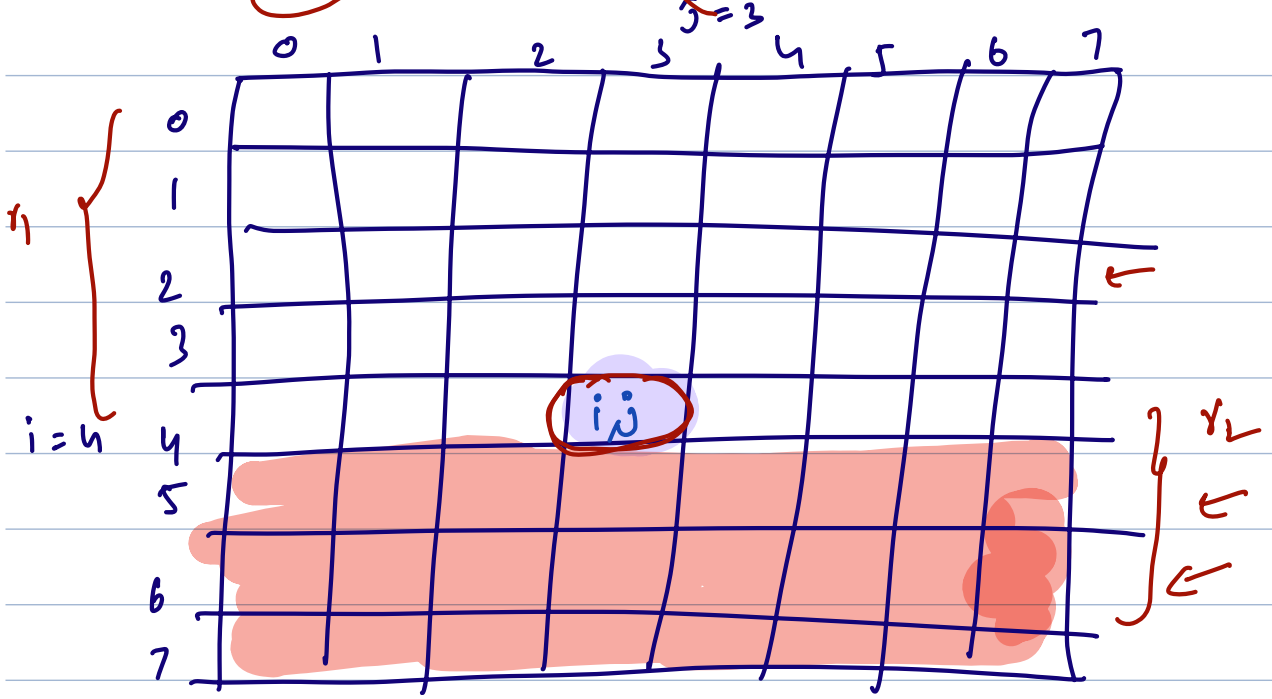
T.C : O(N)

S.C: O(1)

A = [-1 1 -1 -2 -3]
    0 1 2 3 4

N = 5

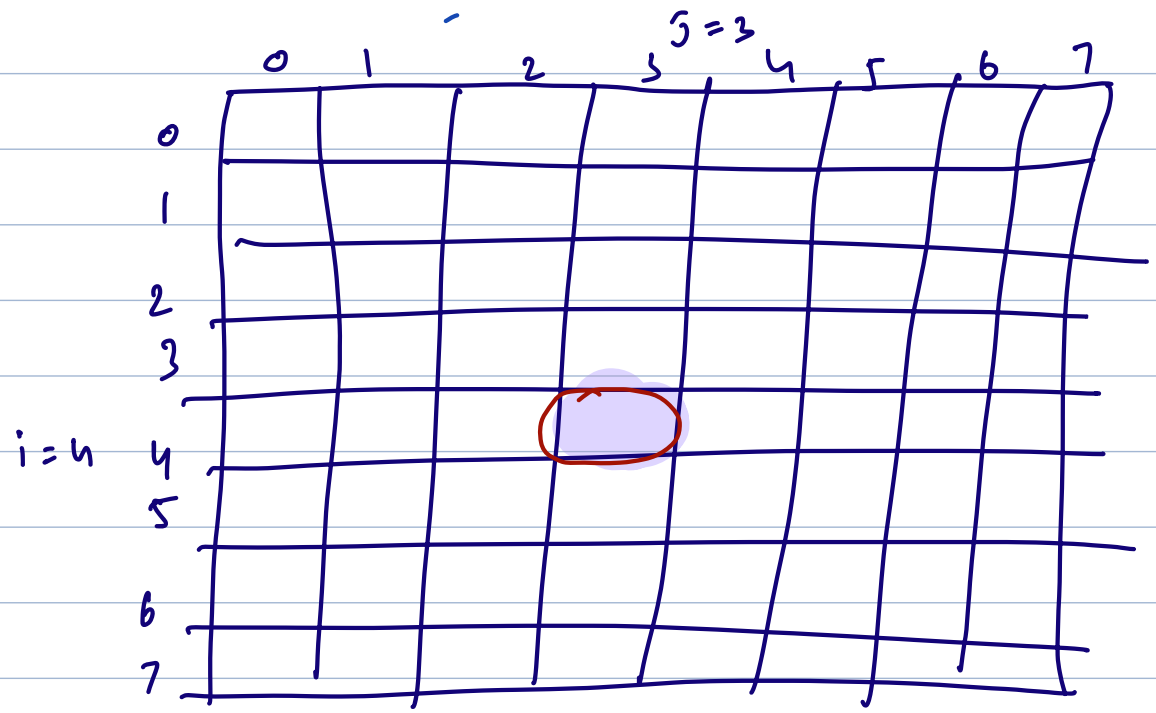
Ans: [1, 6]

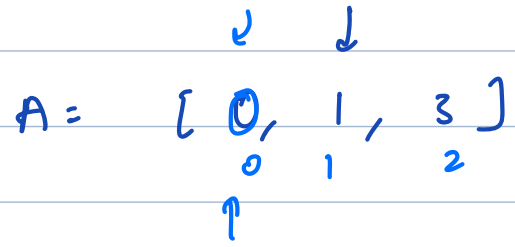


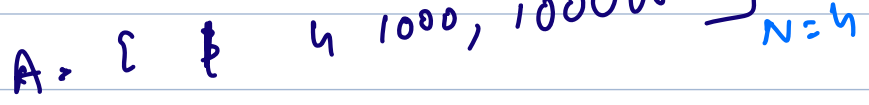
r_1, r_2, c_1, c_2

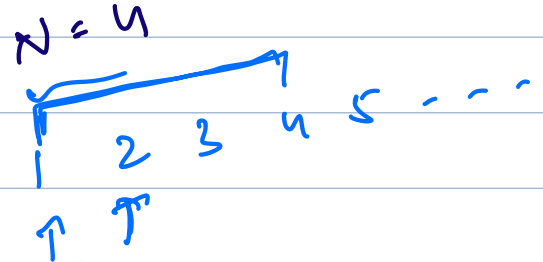
r_1: [0, 4]
r_2: [4, N-1]
c_1: [0, 3]
c_2: [3, M-1]

[0, i]
[i, N-1]




A = [0, 1, 3]
0 1 2


A = [4 1000, 10000]
N = 4


N = 4
2 3 4 5 ...


A = [4 2 3 1]
N = 4