Date _____ Page _____

Day-46

~~BS in 2D Array~~

2D Array Interview Questions - 3

* **BS in 2D Arrays**

	0	1	2	3	4
0	2	6	10	14	18
1	20	24	27	29	38
2	47	52	78	93	102
3	108	111	200	218	320

N = 4, M = 5, x = 52.

⇒ We have to find x in the given sorted 2D-Array.

Brute Force

⇒ Traverse all the elements of the 2D Array & check with x. If ~~ele~~ any element equal to x then return 1 otherwise 0.

Optimized Approach

⇒ First, we will check that the finding element lies in the row 1 as —

⇒ lies in the $2 \leq x \leq 18$ (X)

So, now check in the next row —

⇒ $20 \leq 52 \leq 38$ (X)

⇒ $47 \leq x \leq 102$ (✓)

⇒ Now we will do or apply BS in this row.

Date _____ Page _____

Code

```
for( i=0 ; i<N; i++){
    if(Matrix[i][0] <= x && x <= Matrix[i][n-1])
    {   start = 0 , end = n-1;
        ~~while~~
        // Apply BS here
    }
}
```

T.C → O(N + log M)

S.C → O(1)

More Optimized Approach

⇒ We can write it as in a 1-D Array because ~~it stores~~ in memory it stores like ~~an~~ a 1-D Array.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
start →	2	6	10	14	18	20	24	27	29	38	47	52	78	93	102	108
	111	200	218	320	→ end											
	16	17	18	19												

⇒ start = 0, end = 19.

⇒ mid = 9

(row_index = Index / col → 9/5 = 1
  col_index = Index % col → 9%5 = 4)

⇒ We will use the above formulas to find the value of row & col index.

Date _____ Page _____

Code

```
start=0 , end = N * M-1;
while (start <=end ){
    mid = (start+end)/2;
    row_index = mid/M;
    col_index = mid % M;
    if (Matrix[row_index][col_index]==x)
        return 1;
    else if ( Matrix[row_index][col_index]<x)
        start = mid+1;
    else
        end = mid-1;
}
return 0;
```

* Search in sorted row col wise Matrix:

	0	1	2	3	4
0	4	8	15	25	60
1	18	22	26	42	80
2	36	40	45	68	104
3	48	50	72	83	130
4	70	99	114	128	170

← decreasing
↓ Increasing

The array is sorted row or col wise but not overall sorted. just like previous question.

Date _____ Page _____

⇒ Here, we have to make a pattern like this-

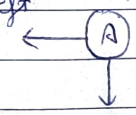
→ increasing / decreasing

↓

decreasing / increasing

⇒ To apply binary search.

⇒ If the element is larger then go the down in column and if element is smaller then go the left.




```
if (A == X)
    return 1;
else if (A < X)
    Down (i++)
else (A > X)
    left (j--)
```

⇒ We will start from the end of first row — means i=0, j=4.

Code

```
N=5, M=5, X=50
i=0, j= M-1;
while (i<N && j>=0){
    if (Matrix[i][j] == X)
        return 1;
    else if (Matrix[i][j] < X)
        i++;
    else
        j--;
}
return 0;
```

T.C → O(N+M)