

Binary Search in 2D Array:

Searching in Matrices,

Bruteforce

```
for (r=0; r < n; r++) {  
    for (c=0; c < n; c++) {  
        if (arr[r][c] == target) {  
            }  
        }  
    }  
    // It takes  $O(N^2)$ 
```

Q: Matrix is sorted in a rowwise & column wise manner.
Ignore smaller than target.

lower bound

10	20	30	40
15	25	35	45
28	29	37	49
33	34	38	50

upper bound,

Ignore this because the column is greater than target -
target = 37

$O(2n) = O(N)$

Tips:

In interviews, if they give searching problems try to minimize the search.

Case 1: if element == target // Ans found

Case 2: if element < target
row ++

Case 3: If element > target
col --;

Code :

```
static int[] search (int[][] matrix, int target) {  
    int row = 0;  
    int col = matrix.length - 1;  
    while (row < matrix.length & col >= 0) {  
        if (matrix[row][col] == target) {  
            return new int[] {row, col};  
        }  
        if (matrix[row][col] < target) {  
            row ++;  
        }  
        else {  
            col --;  
        }  
    }  
    return new int[] {-1, -1};  
}
```


Q: Search in a sorted matrix:

mid, apply BS

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

target = 2

Ignore.

Ans: sorted

case:

① If element == target // Ans

② If element > target

Ignore rows after it.

③ If element < target

Ignore above ~~rows~~, cols.

30:00

In the end 2 rows are remaining:

1	2	3	4
5	6	7	8

① Check whether the mid row have target.

② Consider the four parts if the target does not contain in

mid.