ELEMENT SEARCH IN A 2D MATRIX (ROW/COLUMN WISE)

✡ In this problem, we are given a 2D matrix of integers which has each row and column sorted individually. Our job is to find row, column where the given element exists.

Brute force solution is to obviously run an $M \times N$ loop to find the element. Time Complexity is $O(MN)$.

Better solution is to iterate through each row, if the possibility of the element existing in that row exists, then we run a Binary Search. Time Complexity is $O(M \log N)$.

Optimal solution is slightly complex. We know each row & each column is sorted, so we start at $i = 0$ and $j = N - 1$. If element is bigger than our current one, we know that anything below will also be bigger, hence we eliminate it. But, if current element is smaller, we know anything to its left is also smaller, so we remove it.

Pseudocode :

```
eleSearchRowWise (arr, M, N, K) {
    ele = (-1, -1)
    i = 0
    j = N - 1
    while(i <= M && j >= 0) {
        if (arr[i][j] == K) {
            ele = (i, j)
            return ele
        } else if (arr[i][j] > K) {
            j--
        } else {
            i++
        }
    }
    return ele
}
```