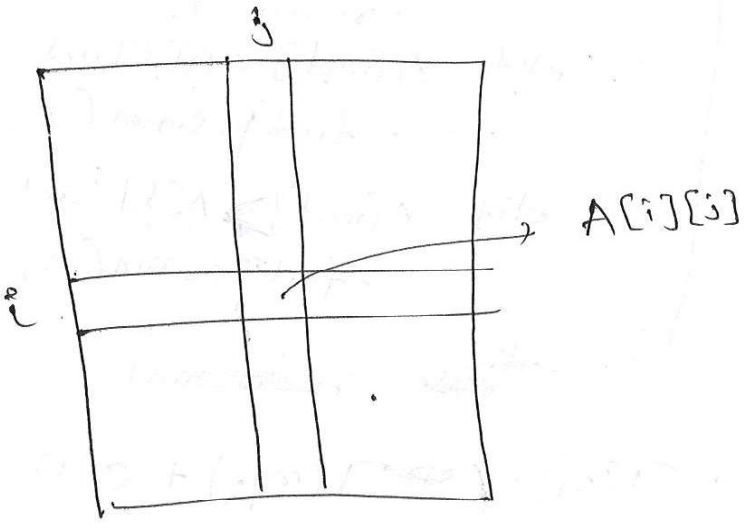
Tutorial – 4 Solutions

Q(1): Method-1: Apply binary-search method on each row (column from 1 to n. and stop when you find the key, else go till the last row n. If you couldn't find, even in the last row, claim that the key is not in the matrix. Time complexity = $O(n \log n)$

Method-2



if $A[i][j] \geq$ key then

key cannot present in $A[i+1:n]$ $A[j+1:n]$

(i.e) Solve the 3 subproblems

$A[1:i][1:j]$, $A[1:i][j+1:n]$, and

$A[i+1:n][1:j]$

else

Solve the 3 subproblems

$A[i+1:n][1:j]$, $A[1:i][j+1:n]$, and $A[i+1:n][j+1:n]$

Time complexity @ $T(N) = 3T(N/4) + O(1)$

$\Rightarrow T(N) = O(N^{\log_3 4})$

Q(2) Hint: Find the position of the largest / smallest element in

Any.

- ~~Special case : mid = 1~~

- In the following algorithm, we are looking for the position of the ~~maximum~~ smallest element

```
findPosition(A, l, r)
{ mid = ⌊(l + r)/2⌋
  if A[l] < A[mid] < A[r] then
      . min elem. is at l.        ] // array is sorted.
      . return l.
  elif A[mid] < A[l] and A[mid] < A[r] then
      . findPosition(A, l, mid).
  elif A[mid] > A[l] and A[mid] > A[r] then
      . findPosition(A, mid, r)
```

~~else return mid~~

You can improve the algorithm

$\therefore T(n) = T(n/2) + O(1)$

$= O(\log n)$

Q(3) Let A & B are two matrices of size $n \times n$. Partition A & B as follows $A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$. $B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$ each A_{ij} & B_{ij} is of size $n/2 \times n/2$

$\Rightarrow AB = \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$

$\Rightarrow$ To compute AB, we are computing 8 multiplication of $n/2 \times n/2$ size matrices and four additions of $n/2 \times n/2$ matrices

$\Rightarrow T(n) = 8T(n/2) + O(n^2)$ ← comes from matrix additions, each having $n^2/4$ elements.

$= O(n^3)$