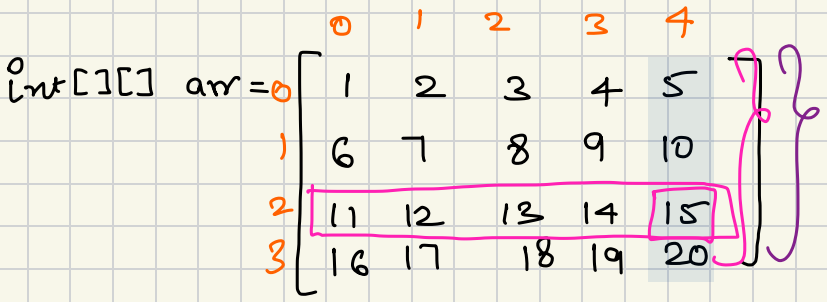
Search in 2D Matrix



to find row — to find key

TC : $O\left(\log_2 N + \log_2 M\right)$

SC : $O(1)$

- smaller or equal to last person of the row.
- greater than last person of the prev. row

just greater or equal element.

Binary Search { Searching Algo }

divide search region into 2 half and take one and eliminate another

TC: $O(\log_2 N)$ SC: $O(1)$

Agenda

- find pivot in a rotated sorted array
- find element in a rotated sorted array
- peak index in a mountain array
- peak index
- Binary Search over Soln

Rotated Sorted Array.

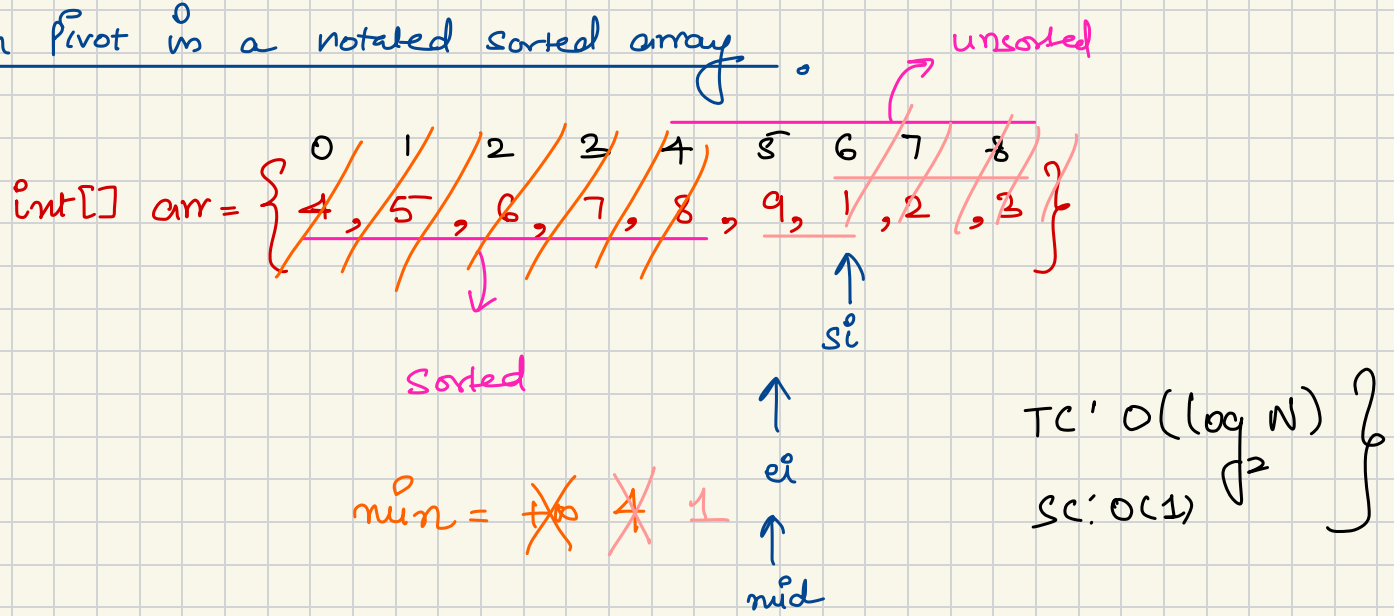
✓ int[] arr = {4, 5, 6, 7, 8, 9, 1, 2, 3}

first element to start rotation {pivot}

↓

smallest element in the array

Search Pivot in a rotated sorted array.



TC: O(log N)

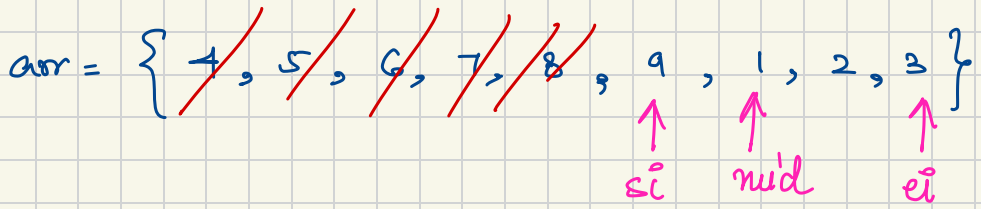
SC: O(1)

{ 5, 6, 7, 8, 9, 1, 2, 3, 4 }

ei si

mid

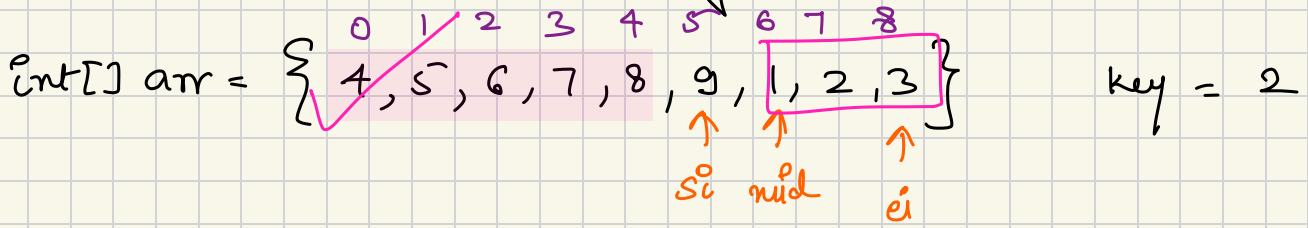
minVal = ~~i~~ 1



if arr[x-1] > arr[x] : pivot at x^{th} index

if arr[y] > arr[y+1] : pivot at $y+1^{th}$ index

find element in a rotated sorted array.



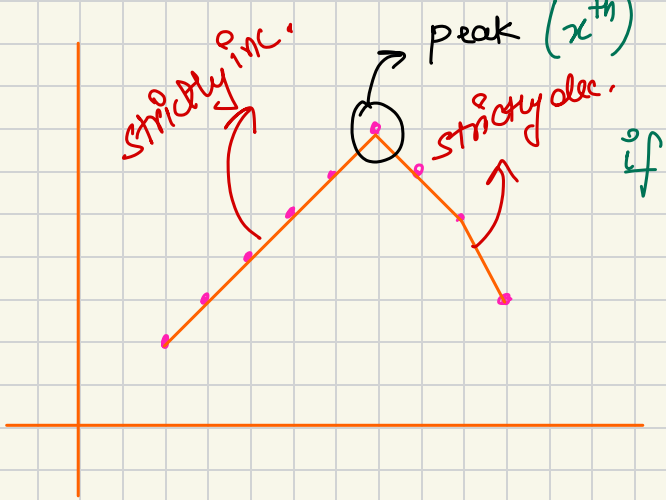
key = 2

→ [4, 8]

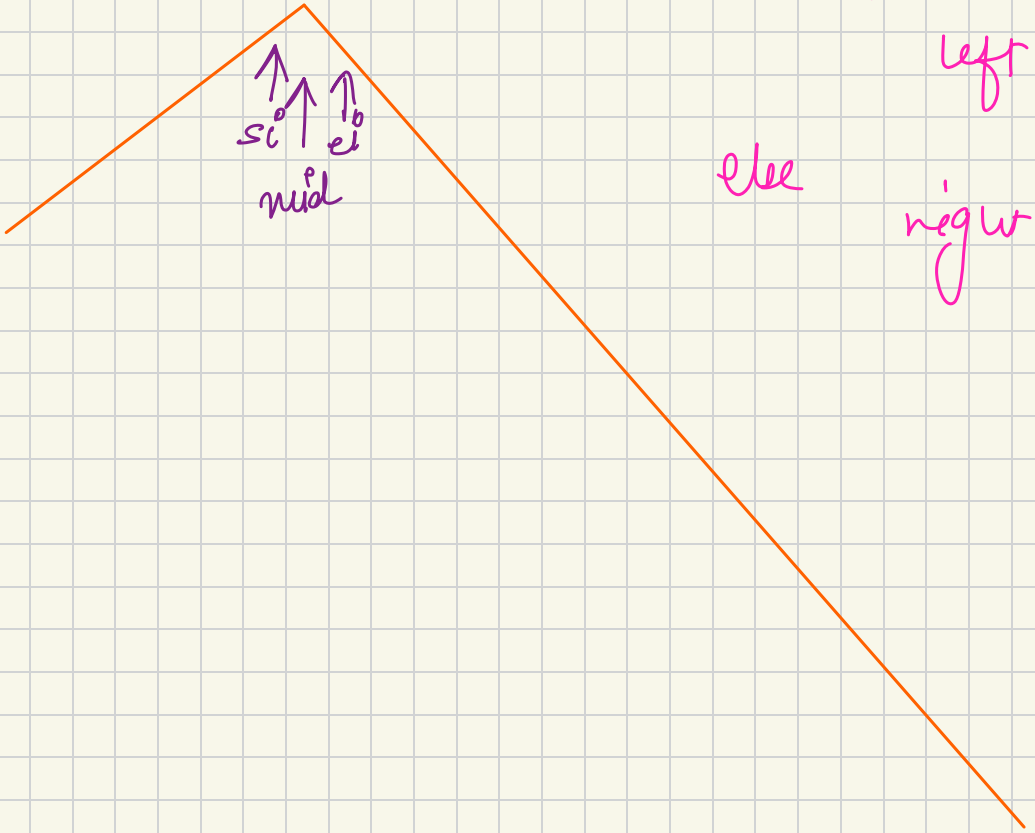
[1, 3]

Peak in a Mountain array.

int[] arr = { 1, 2, 3, 4, 5, 6, 5, 4, 2 }



if (arr[x+1] < arr[x] && arr[x-1] < arr[x])

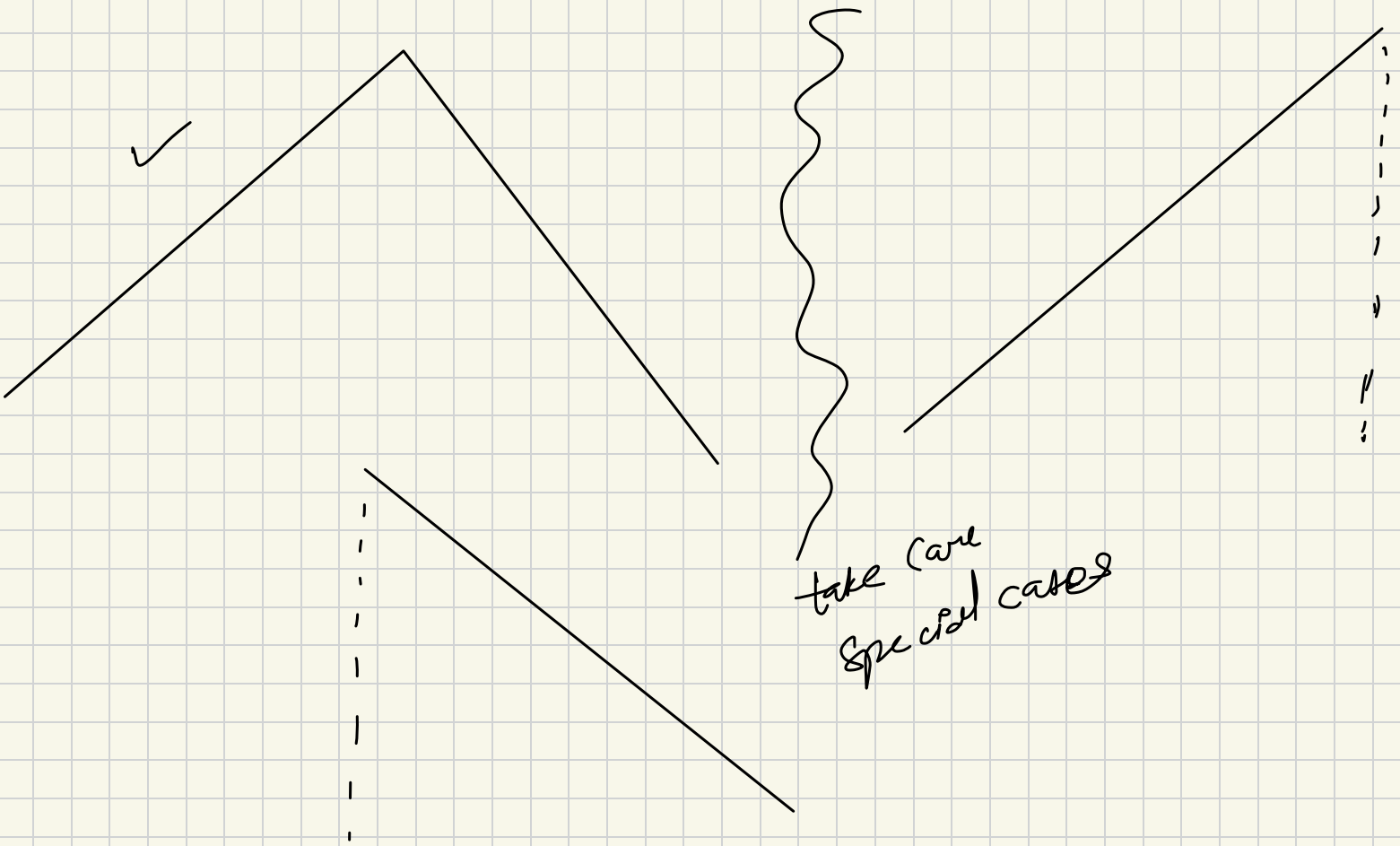


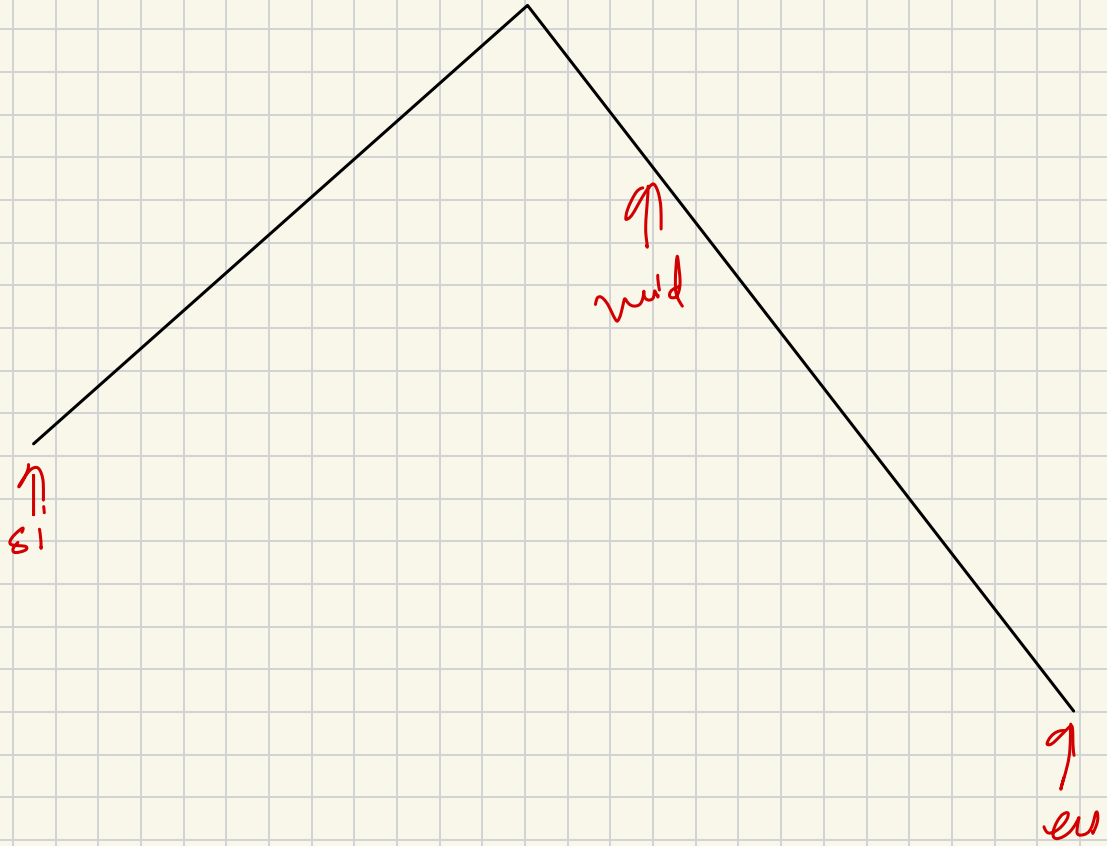
if (arr[mid-1] ≤ arr[mid])

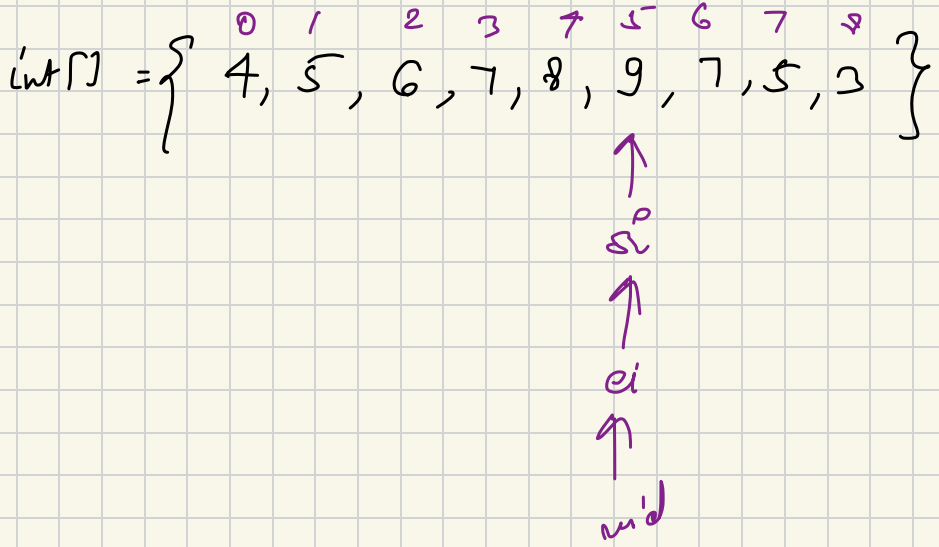
left inc.

else

right is dec


take care
special cases


q_1
q_{mid}
q_{en}


0 1 2 3 4 5 6 7 8
int[] = { 4, 5, 6, 7, 8, 9, 7, 5, 3 }
si
ei
mid

Allocate Minimum No. of Pages

int [] Books = {34, 12, 67, 90} Students = 2

- distribute these N Books among M students
- each student gets min one Book
- distribution should be contigeneous

int[] Books = {34, 12, 67, 90} Students = 2