DSA - 3 Data structures ✕ Not easy → M3

SDE

*

M1, M2 → Crisp clear

↳ easy / building blocks

How to approach any problem ?

→ Pseudo Code

→ Dry Run

→ Code

1) Interactive

↳ Raise

rbt ✓

dbt ✓

clr ✓

Questions

30 mins 1 hr 1.5 hrs

15-20 mins → Thinking ✓

15 ✓

20 mins → Hints

↳ Solution Read ✓

Medium Hard

1) Interviews

↳ fixed time

1 hr

Notes

↳ Notes

↳ Personalised ⇒

→ search in 2D Matrix ✱

→

→

✱ Contests → Weekend → 1 Module Test

↳ clearing concepts

↳ Story types

↳ Free

↳ 1-2 hrs

Attempt all of these

→ OA

↳ Code →

↳ MCQ → ⇒ 1-2 hrs

Focus

⇓

INTERVIEW

✱ Assignments

↳ CW → HW

Minimum in Rotated Sorted Array

1, 2, 3, 4, 5, 6, 7, 8 ✓

8, 1, 2, 3, 4, 5, 6, 7

7, 8, 1, 2, 3, 4, 5, 6

6, 7, 8, 1, 2, 3, 4, 5

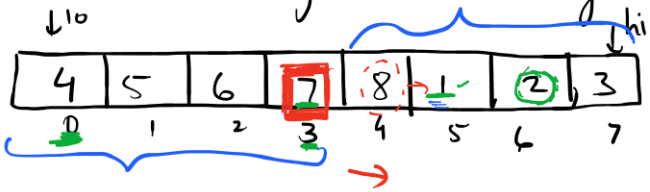
5, 6, 7, 8, 1, 2, 3, 4 ⇒ Rotated Sorted Array

↳ Find the smallest element.

1) Iterate over the entire array and find the minimum.

↳ O(n)

2) sorted + O(log n) ⇒ Binary search



lo = 0

$mid = \frac{0+6}{2} = 3$

Steps

→ mid is the greatest/smallest

Ans mid+1 mid

→ search in the non sorted part

hi = 7

1) Is 7 the greatest?

↳ compare the element on right side

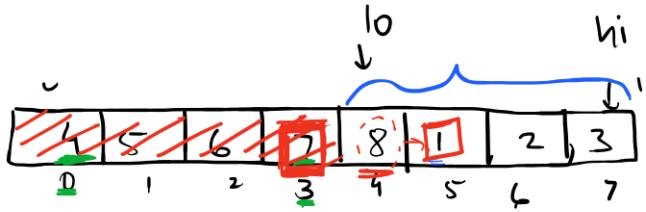
7 > 8 × 7 is not the greatest?

↳ compare the element on the left side

⇒ Is mid-1 the greatest

=

mid is the smallest



lo = 4

hi = 7

mid = 5

↳ smallest element

(mid-1) 8 > 1

↳ Yes

↳ 8 is the greatest element

1 > 2 ×

1 is not the greatest element

Edge case ✓ (1), 2, 3, 4, 5, 6, 7, (8)

① < ⑧ ✓ Array is sorted
↳ arr[0]

Search in 2D Matrix

mxn

1	3	5	7 → sorted array	10
10	11	16	20	↳ Binary search ??
23	30	34	60	

1) Brute force
↳ traverse over the entire matrix
↳ return true if target is found

O(m×n)

2) Every row is individually sorted.
↳ We can apply binary search on it

Total number of rows :

... number of rows.

m

↳ Each row is an array of size n

↳ Binary search on size n.

O(m × log n)

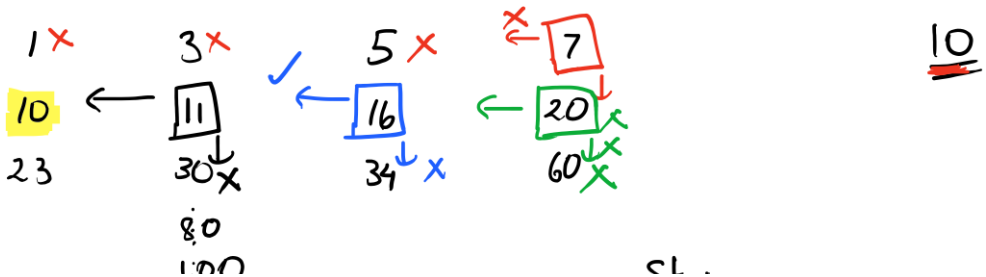
3) Similar to 2

↳ Apply Binary Search on each column

O(n × log m)

4) Observation

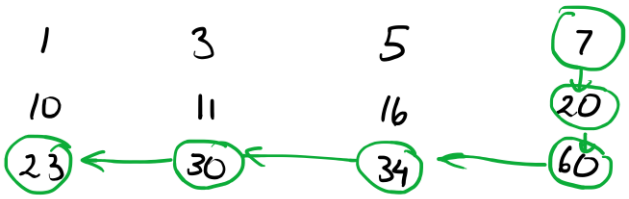
↳



10

1000

Staircase Method

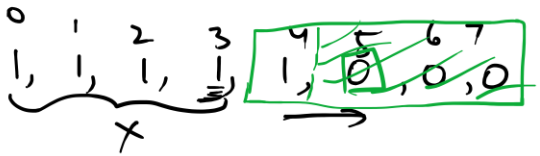


23

O(m+n) ☺

Count 1 in sorted Binary Array

↳ Yes



↳ Conditions of Binary Search will change

Count of 1

↳ (last idx of 1 +1)

↳ Binary Search ??

lo=0
hi=7

mid = 3

↳ ==1 ✓

+ update lo

last idx = ✗ 3

↳ search for better ans on right

lo = 4 mid = 5 = ans
hi = 7

↳ better ans on left side

2, 2, 2, 2, 1, 1, 1, 1, 1, 0, 0, 0

7 5
1 2 8 10 11 12 19

Floor of sorted Array
↳ element just smaller or equal to 5

1, 2, 8, 10, 11, 12, 19

Search Rotated Sorted Array

4, 5, 6, 7, 8, 1, 2, 3

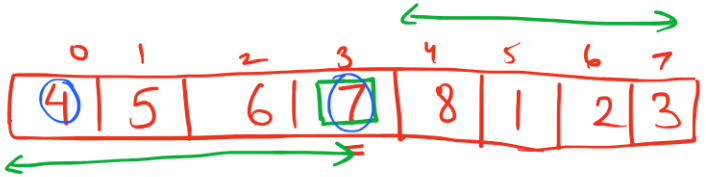
6

1) Linear Search
   ↳ O(n)

2) Binary Search ??

No
↳ sort ?? ✗ → n·logn ✗

↳ Directly ?? ✗
↳ Modify the B.S



= 6

lo = 0
hi = 7

mid = 3

7
↳ check == target
↳ if yes ☺

else
is left part sorted

yes

no/right part sorted