Interview Questions

1) Kth smallest element — k = 2 (2nd smallest element)

↳ Quicksort / Merge Sort → Sorted array → $\Theta(n \log n)$

Kth Largest element → return arr(k-1)

Selection Procedure → $\Theta(n)$

m → Pivot element index

20, 40, 5, 7, 9, 12, 46 (QuickSort)

Partition → return position of Pivot element

SelectionProcedure (arr, 2, 0, len(arr)-1) (k, i, J)

↳ m = 4

0	1	2	3	4	5	6
5,	7,	9,	12,	20,	40,	46

k = 2

↳ output = 7

m > k

↳ Left side

SelectionProcedure (arr, k, i, m-1)

m-i

else: Right side

J-m

SelectionProcedure (arr, k, m+1, J)

Recurrence Relation

Partition algo

$$T(n) = \begin{cases} T(n-1) + n & \rightarrow \text{Left side} \\ \text{OR} & \\ T(n-n) + n & \rightarrow \text{Right side} \end{cases}$$

Best/average

$$T(n) = T(n/2) + n$$

$$= \Theta(n)$$

Worst case

$$T(n) = T(n-1) + n$$

$$= \Theta(n^2)$$

3) Kth Largest element

Partition, Pivot

$k = 2$

0	1	2	3	4	5	6
5	7	9	12	20	40	46

→ Output = 40

$m = 4$

$m > k$:

→ Right search space

Sort colors

↳ [0, 1, 2) → Sorting algo

↳ Quicksort

↓

$\Theta(n \log n)$

Two Pointers approach

$\Theta(n)$

curr = 0

p0 → zeros (0's) p0 = 0

p2 → two (2's) p2 = len(nums) - 1.

[2, 0, 2, 1, 1, 0]

(0, 0, 1, 1, 2, 2)

↑ 0 ↑ n-1

→ nums(curr) == 1:
curr += 1

nums[curr] == 0
↳ swap(nums(curr), nums(p0))
p0 += 1
curr += 1

nums(curr) == 2:

swap(nums(curr), nums(p2))

p2 -= 1

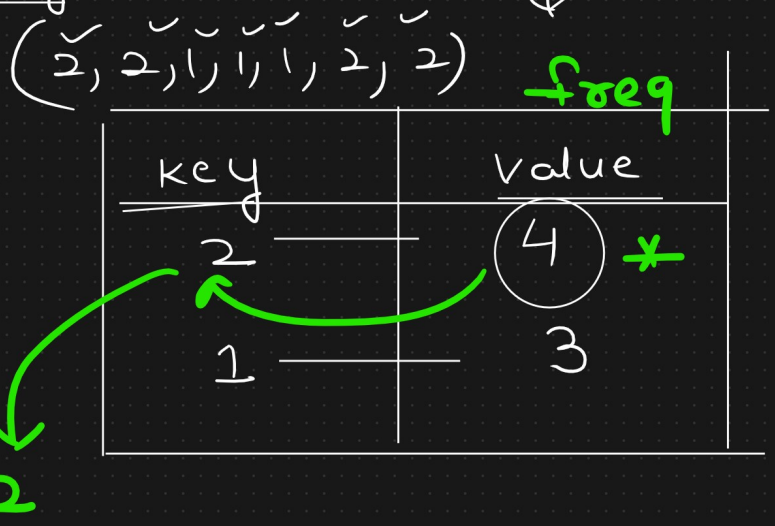
4) Majority Element

Approach 1

Hash Table

Hashing → Dictionary

(key-value) pair



Dictionary → { }

List → [ ]

(5)

[1, 2, 1, 3, 5, 6, 4]
0 1 2 3 4 5 6

nums → array name

→ Optimized approach

output → 6

n = 7

[4, 3] → 4
0 1

nums(0) > nums(1)
↳ return 0

[_ _ _ _ _ 4, 6] → 6
0 1 2 3 4 5 6

nums[-1] > nums(-2)
↳ return len(nums) - 1

start = 0
end = n - 1 (6)

(0+6)/2 = 3

(1, 2, 1, 3, 5, 6, 4)
0 1 2 3 4 5 6
↑

while start < end:

mid = start + (end - start) // 2

6 < 4

if nums(mid) < num(mid+1)

3 < 5 5 < 6

start = mid+1

end = 5

mid = 5

start = ~~5~~

else:

end = mid

mid = (5+4)/2

= 4

return end