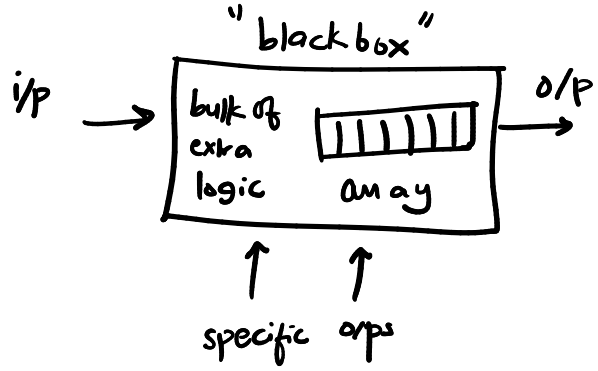
Today → Arrays ✓

1. leetcode problem 3 → sliding window / set. *
2. leetcode problem 76 → sliding window / map. **
3. leetcode problem 4 → Median of 2 sorted arrays | BS ***

4. Abstract Data Type | Containers → set, map, queue, stack, heap.

everyone is using an array.



Certain specific powers.

Problem : map.

s = "ádobecodebanc" t = "abc"

4 substrings.

shortest : "banc"
o/p

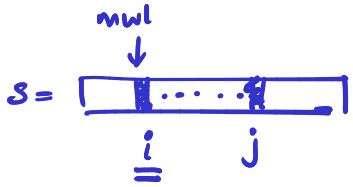
Shortest substring such that all chars of t are in substring.

s = "abab" t = "aa"

o/p : "aba"

Sol^n:

1. t = "abc" map of this t_string ; t_freq



t_freq : {a:1, b:1, c:1}

2. required_unique = 3 ↔ store the no of unique chars that are required int and are available in window.

3. window_freq = { } , window_size = 0

4. min_window_left = 0 , min_len = 10e6

5. i = j = 0 .

"abode" | window = 2 .size

s = adobecodebanc

t = abc {a:1 b:1 c:1}

Keep on going right. → window_freq window size.

j=0	c = a	{a:1}	1
j=1	c = d	{a:1 d:1}	1
j=3	c = b	{a:1 d:1 o:1 b:1}	2

j=5 c=c { a:1 b:1 d:1 e:1 o:1 c:1 } 3

window-size == required-unique

min_len = $(10^6, 6) = 6$

shed elements from i.

i=0 ~~c~~ { d:1 b:1 o:1 e:1 c:1 } 2

Continue to add elements from right.

j=6 c='o' { d:1 b:1 o:2 e:1 c:1 } 2

j=10 c="a" { ... } 3

: — j=n

min_len = (6, 10-1+1)

= (6, 10)

= 6

min_string_left = i

S.substr(i, min_len)

Q.3. Median of 2 sorted arrays. (Don't want to merge it). example.

arr1 m: ~~1~~ ~~3~~ ~~7~~ ~~9~~ 11 15 18 → 7

arr2 n: ~~2~~ ~~9~~ 13 21 25 → 5

} unequal sizes.

Assuming if they were merged:

sorted → [0 1 2 3 4 5 | 6 | 7 8 9 10 11]

~~1~~ ~~2~~ ~~3~~ ~~7~~ ~~9~~ ~~9~~ | 11 | 13 15 18 21 25]

If 25 X, median = 9.

odd → mid

even → avg(2 mids)

even.

12

median = $\frac{9+11}{2} = 10$ ans. 6

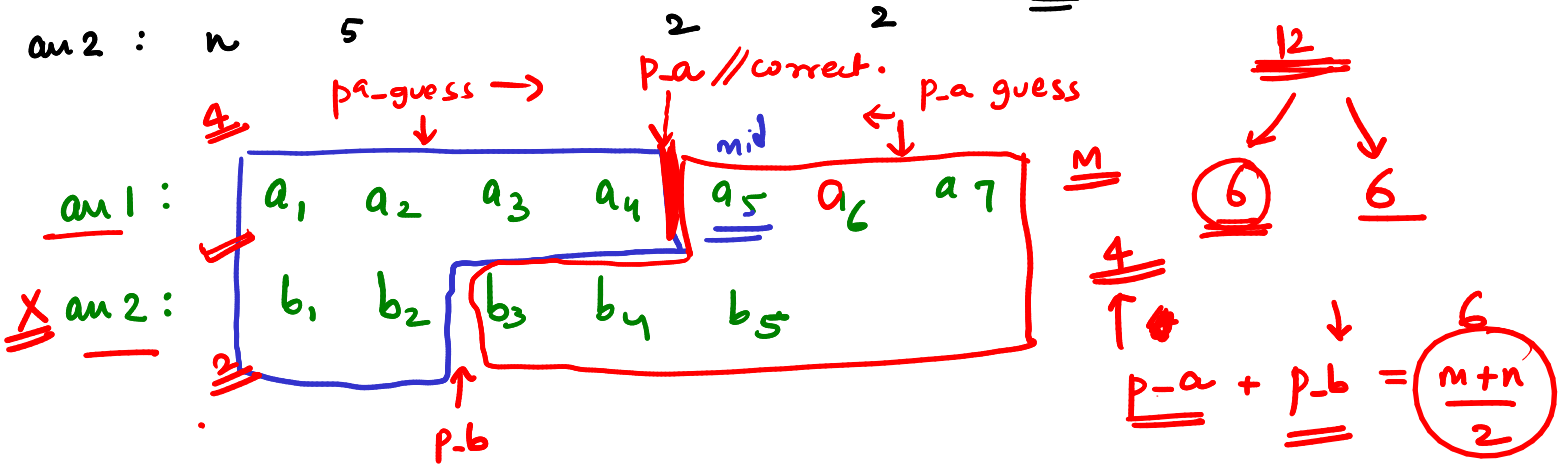
BF: $O(m+n)$ linear TC

Optimize: $O(\log(m,n))$ log TC

p_a p_b ⇒ indices

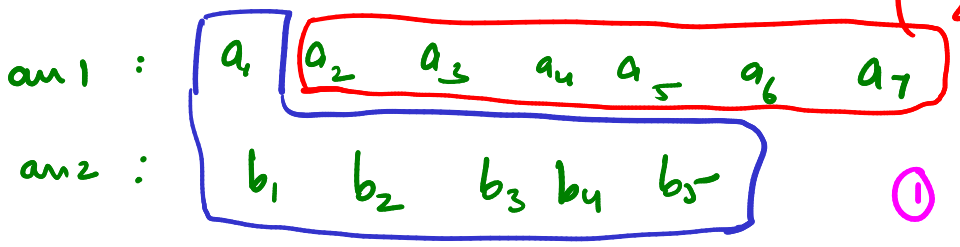
arr1 : m 7 mid = $\frac{m+n}{2}$ = $\frac{12}{2}$ = 6

arr2 : n 5



p_a → guess here. range: 0 to m-1 ← p_a

p_b = $\left(\frac{m+n}{2}\right)$ − p_a



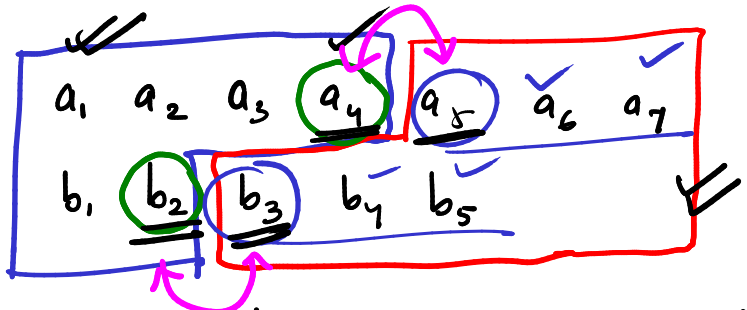
3 unknowns.

p_a → guess

① How will I know this p_a is correct ?

② If p_a is wrong, figure out ⇒ overestimating OR underestimating ?

① sorted



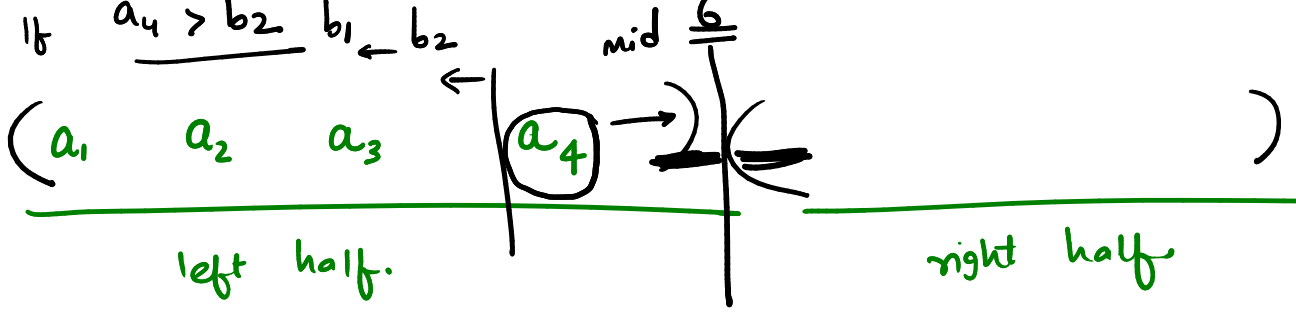
Is this correct ?

$a_4 \leftrightarrow b_3$

$b_2 \leftrightarrow a_5$

If $a_4 > b_2$ $b_1 \leftarrow b_2$

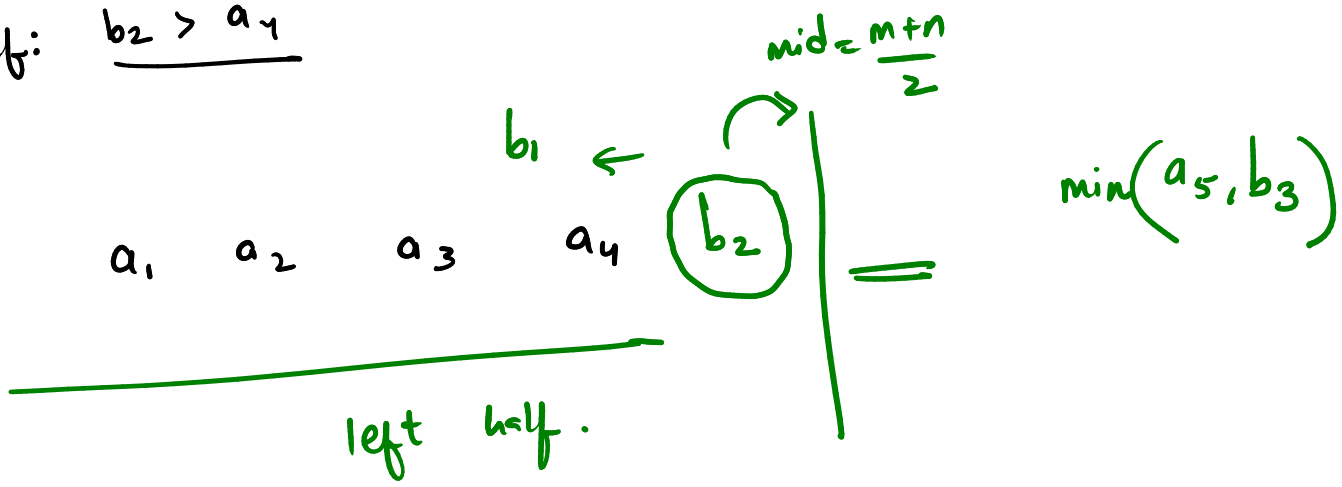
Assume it: I did



b1 a1 a2 b4 a3 a4

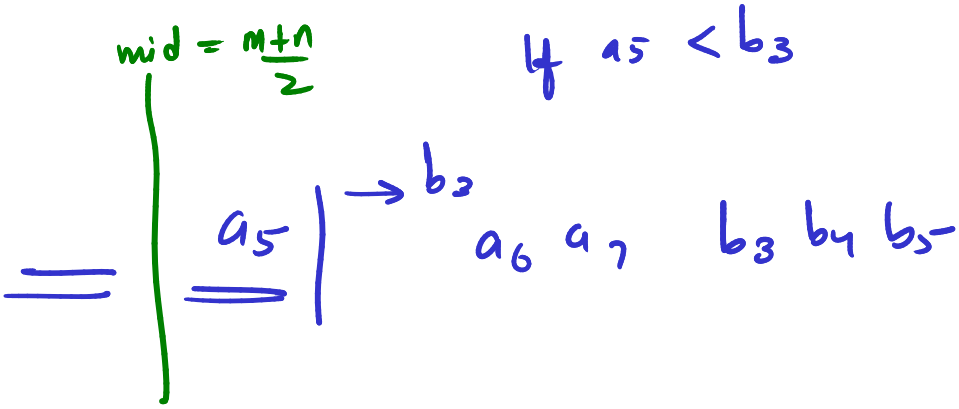
a1 a2 a3 b1 b2 a4

If: $b_2 > a_4$



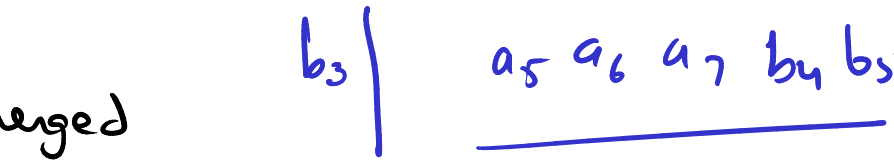
mid_left_half $\max(a_4, b_2)$

Same logic on RHS.



If $a_5 < b_3$

If $b_3 < a_5$



Logic:

✓ guess_work: p_a = 4

p_b = 6 − 4 = 2

mid_elements of the merged array are:

$$\max(a[p\_a], b[p\_b]), \quad \min(a[p\_a+1], b[p\_b+1])$$

Comparisons

$a_4 < a_5$ → Trivial — a is sorted ✗

$b_2 < b_3$ → Trivial — b is sorted

non-trivial ✓

$\left\{ a_4 < b_3 \right.$

$\left. b_2 < a_5 \right\}$ → This must be true because blue → left half, red → right half.

p_a is correct split !!

What happens → underestimate or an overestimate ?

Typical example :

a: 1 3 7 9 | 11 15 18

b: 2 9 | 13 21 25

ⓐ: a[p_a] < b[p_b+1]

ⓑ: b[p_b] < a[p_a+1]

9 < 13 ✓

9 < 11 ✓

11 < 9 ✗ ⓐ

2 < 15 ✓

5 elements from a :

1 3 7 9 11 | 15 18

2 9 13 21 25

Overestimating ← ⓐ ✗ a[p_a] < b[p_b+1]

3 elements from a :

1 3 7 | 9 11 15 18

2 9 13 | 21 25

✓ 7 < 21 ⓐ

✗ 13 < 9 ✗ ⓑ

Underestimating ← ⓑ ✗ b[p_b] < a[p_a+1]

p_a : low 0 to high m-1 ← p_b = (m+n)/2 − p_a

cond? a

✓

✗

✓

cond? b

✓ → p_a is right

✓ → p_a ↓

✗ → p_a ↑

p_b = automatically fixed.

p_a is right ✓ median → odd

median = max(a[p_a], b[p_b]) + min(a[p_a+1], b[p_b+1]) / 2

median = max(a[p_a], b[p_b])