Hard Problems

① Median of 2 sorted arrays
- → Same size
- → Different size

② K^{th} smallest element → 2 sorted arrays

③ Median → Row wise sorted matrix

① Median of 2 sorted arrays.

Part 1: Same size

a_0 a_1 a_2 a_3 a_4

b_0 b_1 b_2 b_3 b_4

} Sorted in increasing nature & can have duplicate

Eg:

2	6	7	9	10
1	3	5	8	8

merged array:

1	2	3	5	6	7	8	8	9	10
0	1	2	3	4	5	6	7	8	9

idx

→ median = $\frac{6+7}{2}$

$= 6.5$

for 2 same sized arrays, the merged array size is always even.

Time complexity of merging = $O(2n)$

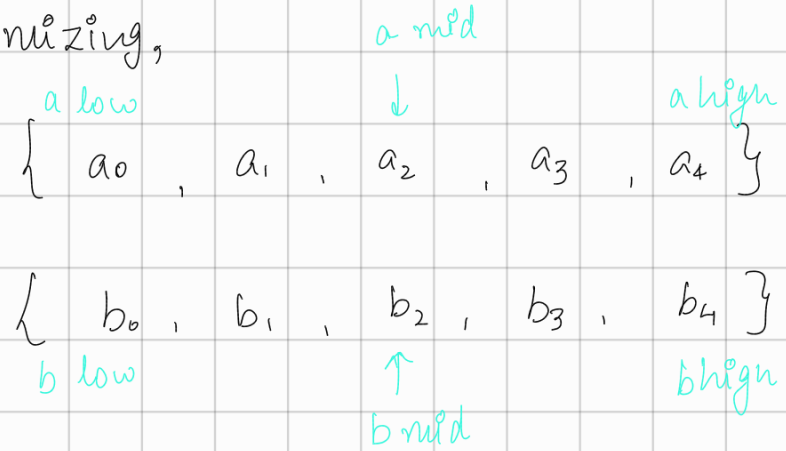
Time complexity of finding median in merged = $O(1)$

} $O(2n+1) = O(n)$

Space complexity of merging = $O(2n) = O(n)$

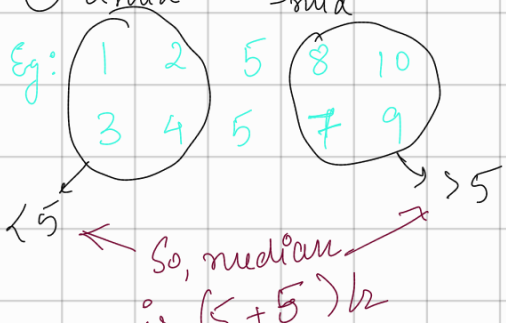
This is a brute force solution

for optimizing,



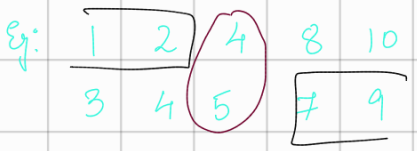
Possibilities in a_{mid} & b_{mid} relation

① $a_{mid} == b_{mid}$



because there are total 4 elements on both sides of a_{mid} & b_{mid}

② $a_{mid} < b_{mid}$



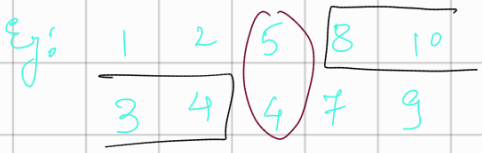
$(a_0 \cdots a_{mid-1}) \leq a_{mid}$ & $(b_{mid+1} \cdots b_{n-1}) \geq b_{mid}$

discard these 2 ranges equal number of elements removed from both sides, so median is unchanged

So search space is reduced

∵ discard left half in a array & right half in b array.

③ $a_{mid} > b_{mid}$



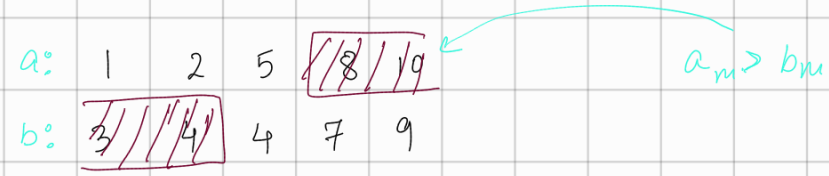
$(a_{mid+1} \cdots a_{n-1}) \geq a_{mid}$ & $(b_0 \cdots b_{mid-1}) \leq b_{mid}$

discard these 2 ranges equal number of elements removed from both sides, so median is unchanged & Search space is once again reduced.

∵ discard right half in a array & left half in b array.

In general, if both mids are equal, then that is the median, otherwise, discard left part in array with smaller mid and discard right part in array with larger mid.

Dry run for above algorithm:



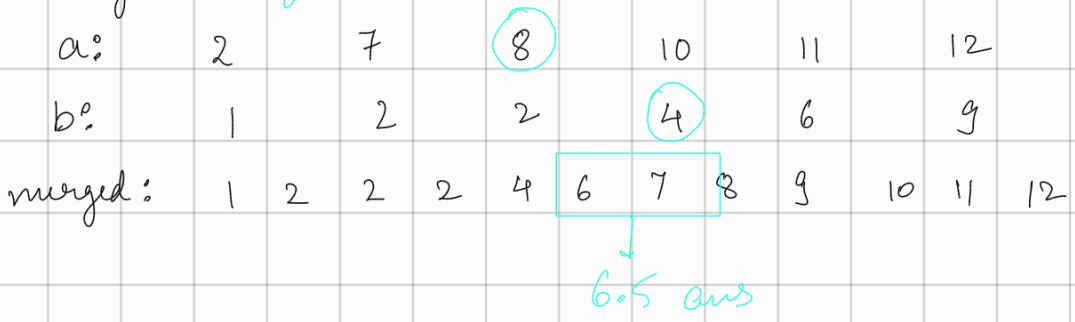
a'': 2 5
b'': 4 7

} when 4 elements are left, merge & return average of 2 middle elements.

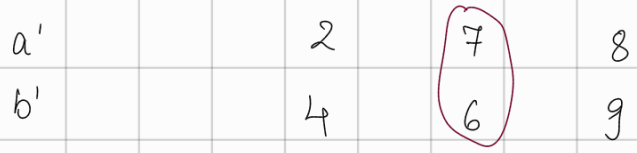
Time Complexity: $O(\log_2 n)$
Space Complexity: $O(1)$

} for arrays of same size

Dry Run for even number of elements,



Take mid in first as $(n/2) - 1$ & in second as $(n/2)$
So that number of elements removed from both is same and thus, the resulting array size is also same.



a'' 2 7
b'' 6 9

} → $\frac{6+7}{2}$ } answer = 6.5

Part -2 : Different Size → EVEN

a:	3	9	12	13	14	19		
	a_0	a_1	a_2	a_3	a_4	a_5		
b:	7	17	20	24	26	28	30	33
	b_0	b_1	b_2	b_3	b_4	b_5	b_6	b_7

Merged array :

3	7	9	12	13	14	17	19	20	24	26	28	30	33
0	1	2	3	4	5	6	7	8	9	10	11	12	13

↳ median $= \left\{\frac{17+19}{2}\right\} = 18$

Median ⇒ Left no. of elements = Right no. of elements

↓

Middle is median

→ So, we need to segregate elements into 2 parts, left & right.

→ If we have segregated one array, the other one can be easily segregated.

→ because, if first array is segregated in 2 parts of 3 - 3, then we can just add 4 to each of 2 parts of first array. So that the size of segregated parts is equal at last

	a_0	a_1	a_2	a_3	a_4	a_5
b_0	b_1	b_2	b_3	b_4	b_5	b_6	b_7

If all LHS elements are smaller than all RHS elements then segregation is completely valid

→ And for that largest element in LHS should be smaller than or equal to smallest element in RHS. and then median will be

$$\left\{\frac{\text{LHS largest} + \text{RHS largest}}{2}\right\}$$

And to compare, we only need to check for

$\{(\cdots) \le a_2\}\ a_2$ ↖ | ↗ $a_3\ \{(\cdots) \ge a_3\}$

$\{(\cdots) \le b_3\}\ b_3$ ↙ | ↘ $b_4\ \{(\cdots) \ge b_4\}$

If $\begin{pmatrix} a_l \le b_r \\ a_2 \quad\&\& \quad b_4 \\ b_l \le a_r \\ b_3 \quad\quad a_3 \end{pmatrix}$ then overall segregation is valid

and if segregation is valid, then sort $\{a_2, a_3, b_3, b_4\}$ & find it's median. That will be the overall median

ODD (Always try to make LHS one larger than RHS)

$a_0\ a_1\ a_2\ a_3 \mid a_4\ a_5\ a_6$

$b_0\ b_1\ b_2\ b_3 \mid b_4\ b_5\ b_6\ b_7$

8 elements — 7 elements

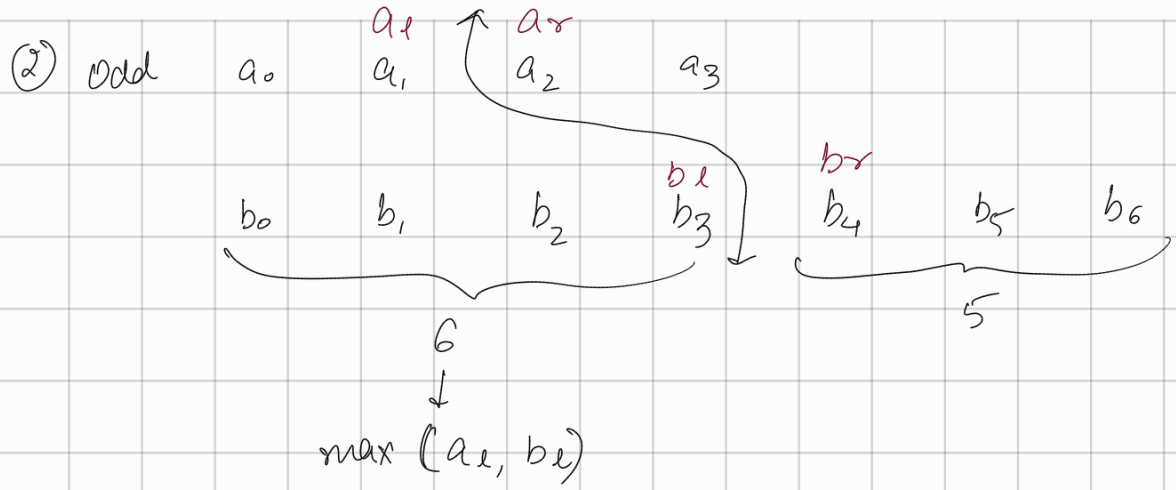
↑ largest of 8 is median if segregation is correct/valid → because it has 7 elements smaller in LHS & 7 larger on RHS.

① Even: $a_0\ a_1 \mid a_2\ a_3$ ($a_l = a_2$, $a_r = a_3$)

$b_0\ b_1 \mid b_2\ b_3\ b_4\ b_5$ ($b_l = b_1$, $b_r = b_2$)

↑ if this seg. is valid

⇒ overall median = $\{a_l, b_l, a_r, b_r\}$ → sort → median



→ So, to find segregation, use binary search on only one array. The other array portion will be automatic as sum of parts is constant

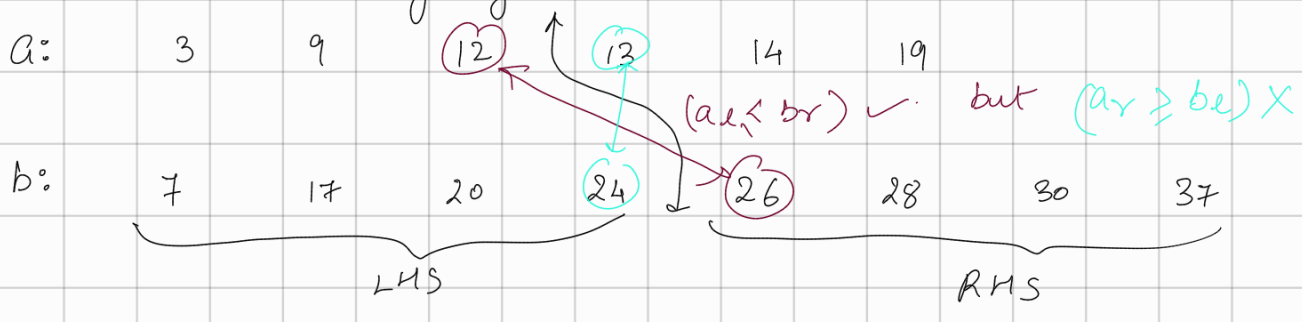
→ Also, binary search is applied from 0 to arr. length and not 0 to arr. length − 1.

	low		a_l	mid	a_r			high
a:	3	9	12	mid1	13	14	19	
idx$_a$	0	1	2	3	4	5	6	
				b_l		b_r		
b:	7	17	20	24	26	28	30	33
idx$_b$	0	1	2	3	4	5	6	7

Total size of segregation = $\left(\frac{n+m}{2}\right) = \left(\frac{6+8}{2}\right) = \underline{\underline{7}}$

at mid1 : In LHS segregation, a contributes 3 elements upto a_l, So b has to contribute (7 − 3) = 4 elements to make the total upto 7 elements.

So, now the segregation is

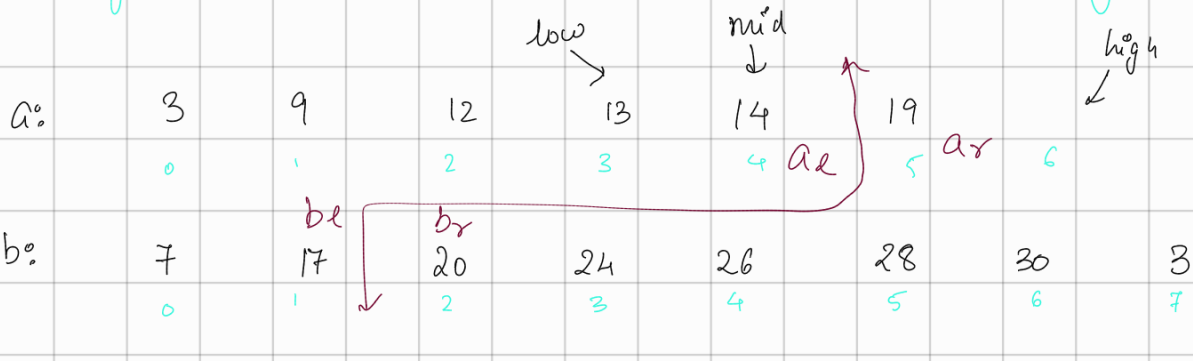


but here 13(RHS) < 24 (LHS). So the above segregation is not valid $\{a_r < b_l\}$

So, we need to take more elements from a into the segregation.

So, then we update low = mid + 1, because BS is applied on a. and we need to include more elements from a array

If $a_l > b_r$, then we need to include more elements from b into the LHS so then update high = mid - 1



So, to find b_r => 7 - a_r & $b_l = b_r - 1$ (index)

here $a_l \leq b_r$ ✓ $\{14 \leq 20\}$

& $a_r \geq b_l$ ✓ $\{19 \geq 17\}$

So, both conditions are true. Thus, our segregation is valid.

Now, to find median,

Sort $\{a_l, a_r, b_l, b_r\}$ & find it's median

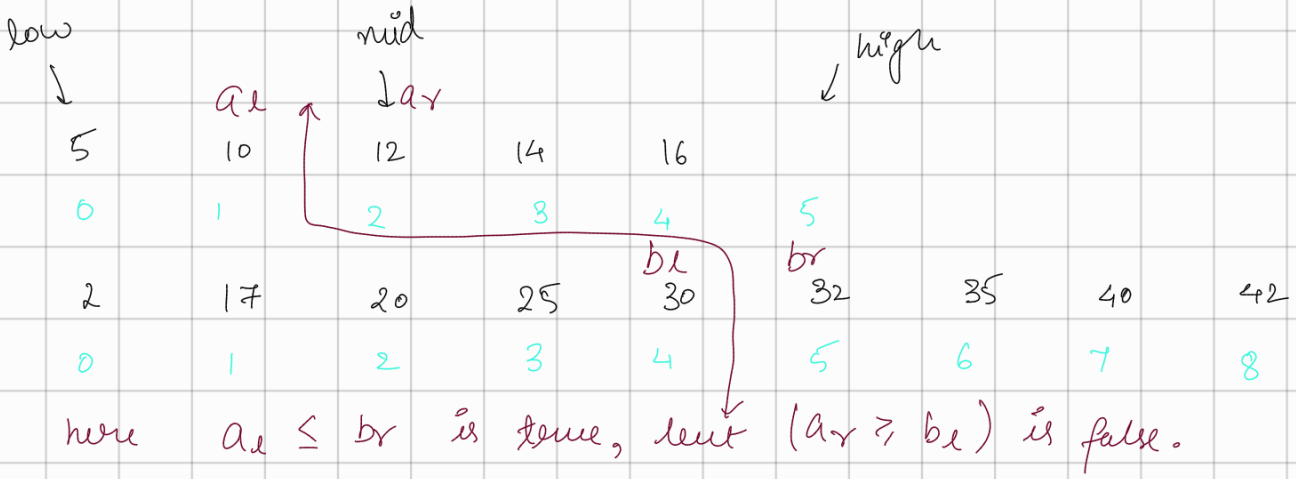
$\{14, \boxed{17, 19}, 20\}$

So, overall median = $\frac{17 + 19}{2} = \boxed{18}$

Corner Case I

☆ Take high as arr. length and not arr. length - 1 because it is possible for complete a array to be smaller & included in LHS

For Odd Elements

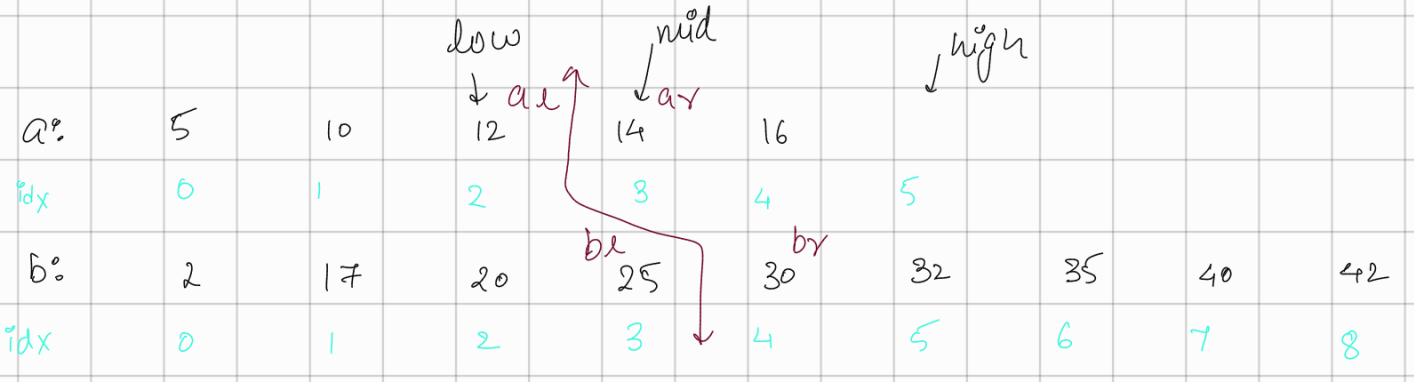


here $a_l \leq b_r$ is true, but $(a_r \geq b_l)$ is false.

So segregation is invalid

So we need to include more elements from a array into the LHS partition.

∴ low = mid + 1

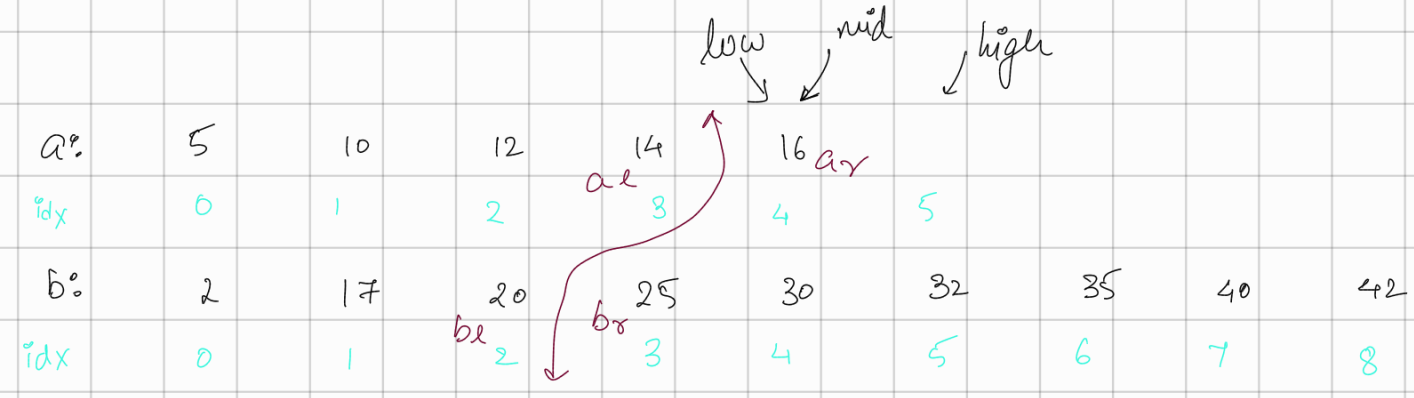


here, $(a_l \leq b_r)$ is true ✓

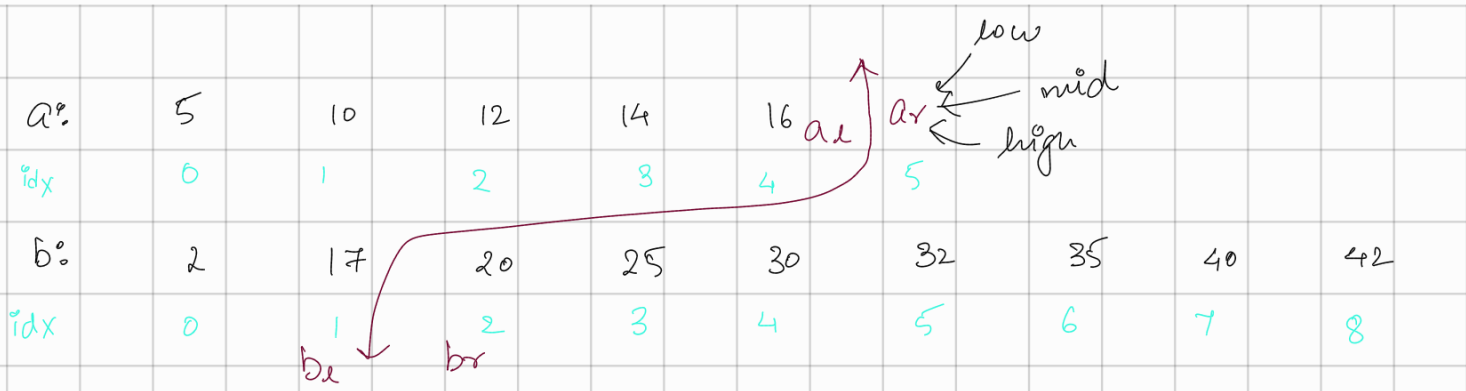
but $(b_l \leq a_r)$ is false ✗

So, we need to include more elements from a array into the LHS partition.

So, low = mid + 1



Again, $a_l \leq b_r$ ✓ but $b_l \leq a_r$ ✗ So low = mid + 1



Here, $a_l \leq b_r$ ✓ & $b_l \leq a_r$ ✓ ← ∵ out of bound ele. on right = ∞

So this is a valid segregation

But to reach this segregation, we need to consider high as arr. length and not arr. length − 1.

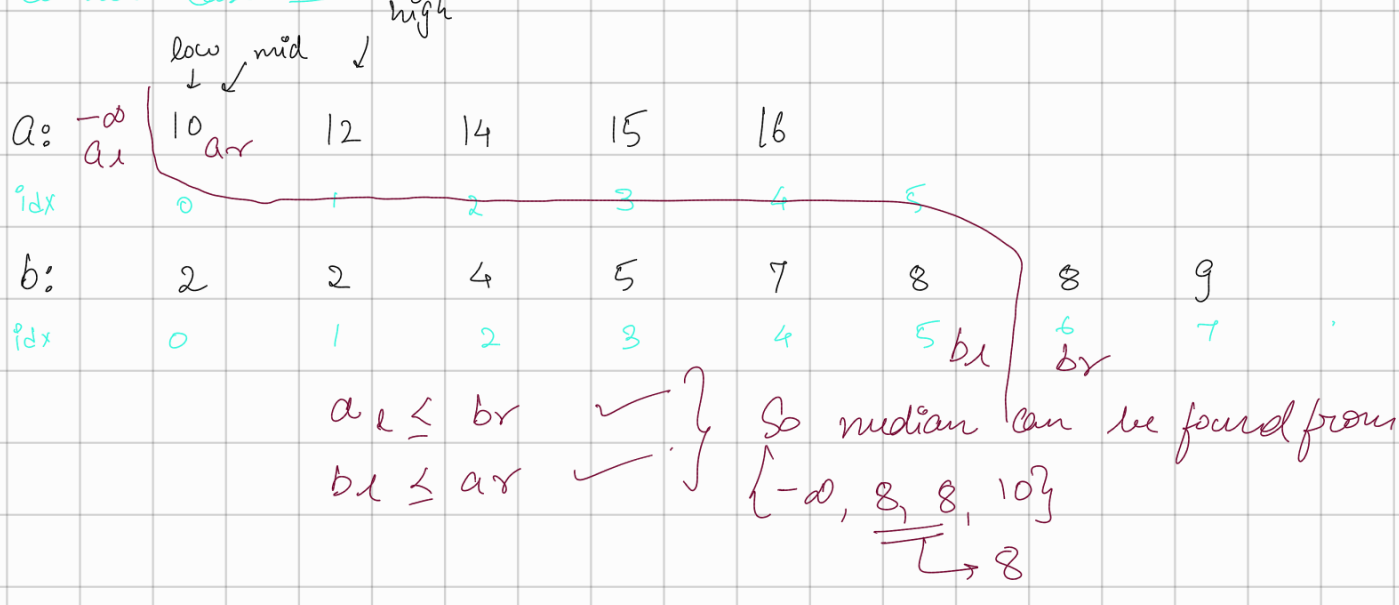
Similarly, if out of bounds on left ⇒ $-\infty$

★ Point 2: It is beneficial if first array is small, because that way the Binary Search can be applied faster

So try to apply Binary Search on the smaller array

∴ Time Complexity = $O(\log_2 N_a)$ N_a = number of elements in array a

Corner Case II:



$a_l \leq b_r$ ✓

$b_l \leq a_r$ ✓

So median can be found from $\{-\infty, 8, 8, 10\}$ → 8