Binary Search Playlist

Video - 22

Facebook
Instagram → codestorywithMIK

Twitter → cswithMIK

Leetcode - 4

Hard → Easy with Dry Run...

Median Of Two Sorted Arrays

(PART-2)

Watch PART-1 → (·) Brute Force O(m+n) Space
(·) O(1) space

Company Tags :- Google Microsoft

4. Median of Two Sorted Arrays

Hard 25819 2843 Add to List Share

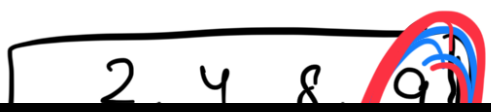
Given two sorted arrays `nums1` and `nums2` of size `m` and `n` respectively, return **the median** of the two sorted arrays

The overall run time complexity should be `O(log (m+n))`.

Best Approach

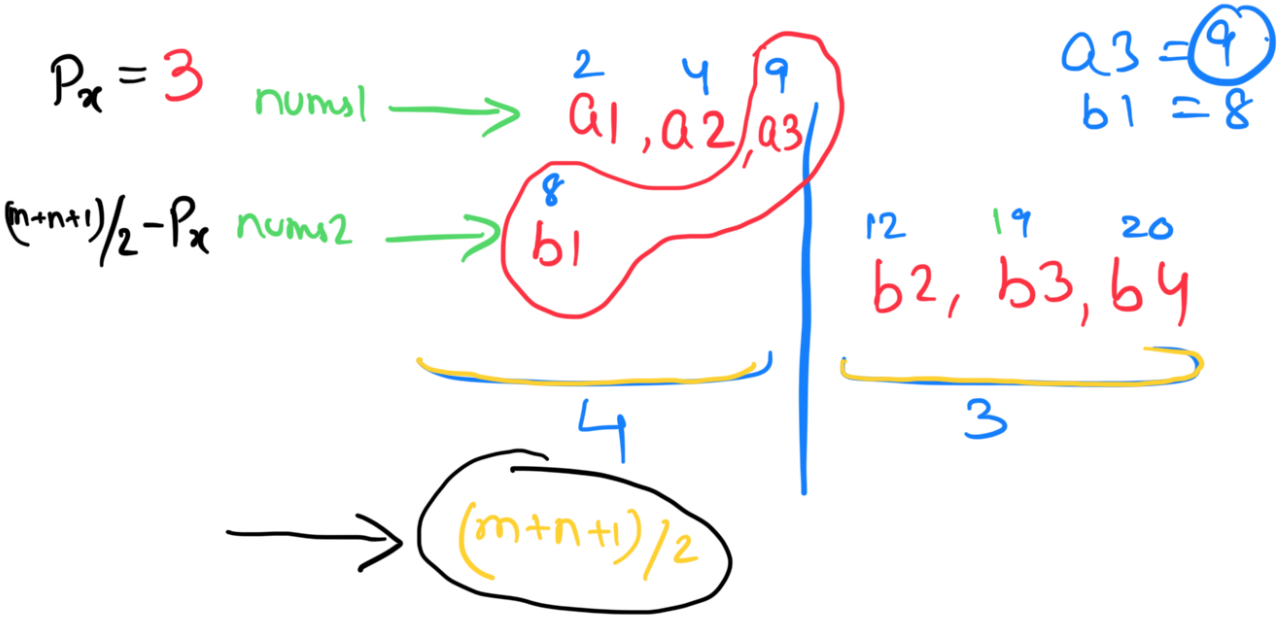
nums1 = { 2, 4, 9 } , m = 3

nums2 = { 8, 12, 19, 20 } , n = 4



nums1 = a1, a2, a3 , m = 3

nums2 = b1, b2, b3, b4 , n = 4



a1, a2, a3

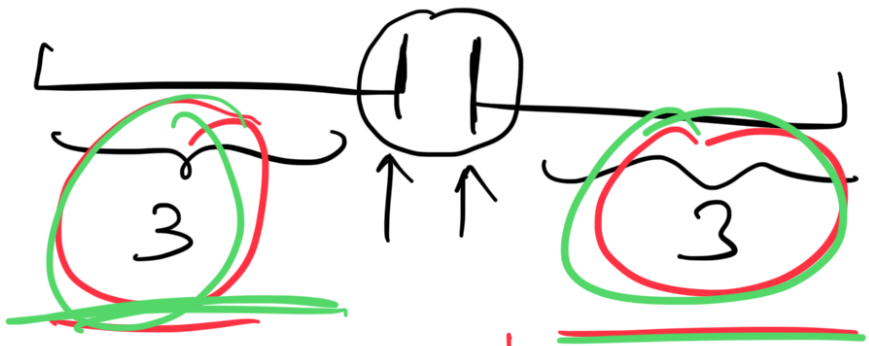
b1, b2, b3, b4

m+n=6

↓

even

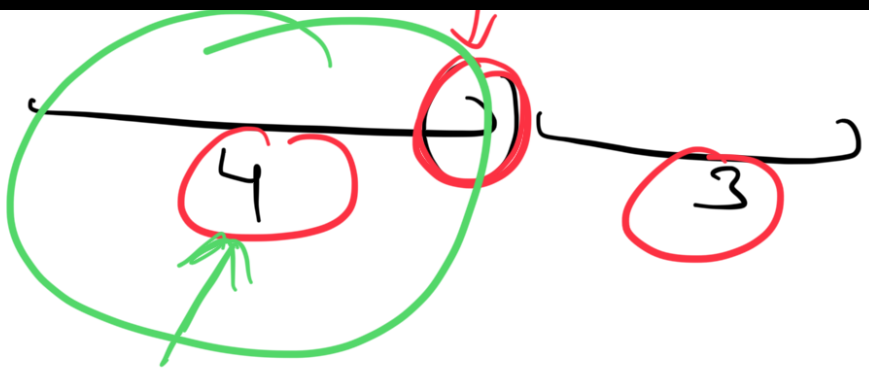
m+n=7



← (m+n)/2

(m+n+1)/2

= (6+1)/2 =



$= 3$

$(m+n)/2$

$= (m+n+1)/2$

$= 4$

$(m+n+1)/2$

nums1 = { 2, 4, 9 } m = 3

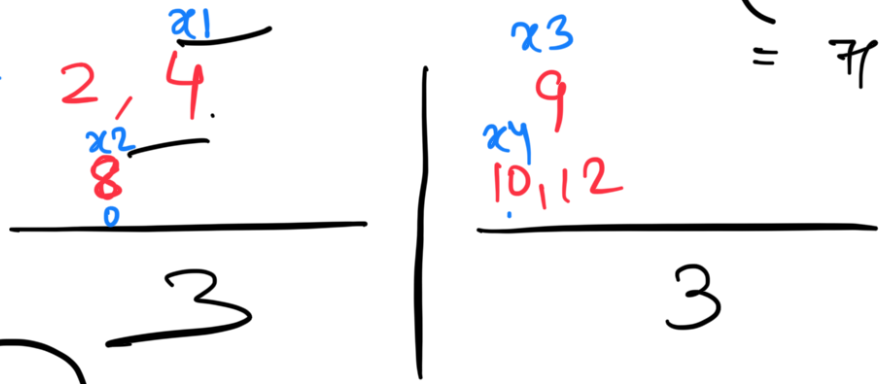
nums2 = { 8, 10, 12 } n = 3

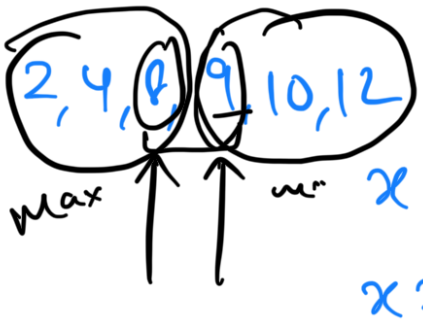
$(m+n+1)/2$

$= 7/2 = 3$

nums1 → Px = 2

nums2 → Py = 1





x1 <= x4 && x2 <= x3

x1 = nums1[Px-1]; // 4

x2 = nums2[Py-1]; // 8

```
x3 = nums1[Px] ; //9
x4 = nums2[Py] ; //10

if ((m+n) % 2 == 1)
    (max(x1, x2) + min(x3, x4)) / 2.0;

return max(x1, x2);
```

Binary search

nums1 → Px

0
1

$l = 0$ $\quad r = m$ 2 ✓

$mid = l + (r-l)/2$ 3 ✓

$=$

$Px = l + (r-l)/2$;

$Py = (m+n+1)/2 - Px$