Median of two Sorted array

BF

$T = O(n_1 + n_2)$

$S = O(n_1 + n_2)$

$n_1 = 6$ i

2	4	8	10	14	16
0	1	2	3	4	5

$n_2 = 4$ j

1	5	9	12
0	1	2	3

1	2	4	5	8	9	10	12	14	16
0	1	2	3	4	5	6	7	8	9

Left side

$\frac{n_1 + n_2 + 1}{2}$

median of ~~two~~

$\frac{8 + 9}{2}$

$n_1 + n_2 = 6 + 4 = 10$

$\boxed{10}$

$(n + n_2) - (\text{left side}) = \text{Right Side}$

$10 - 5 = 5$

right side

$\frac{n_1 + n_2 + 1}{2}$

$= \frac{6 + 4 + 1}{2}$

$= \frac{10 + 1}{2}$

$= \frac{11}{2} = 5$

These many elements have to be in the left array

~~Eff~~

$l_1 <= r_2$

$l_2 <= r_1$

2 4 8 10 14 16

1 5 9 12

For even length

0

X

```
      -∞ | 2 4 8 10 14 16
1 5 9 12 | +∞
```

1

X

```
    l1   r1
    2  | 4  8  10  14  16
      l2   r2
1 5 9 12 | +∞
```

2

X

```
    l1   r1
  2 4  | 8  10  14  16
    l2   r2
1 5 9  | 12
```

③ ✓

		l_1		r_1		
2	4	8	|	10	14	16
		l_2		r_2		
	1	5	|	9	12	

$l_1 <= r_2$ ✓
$l_2 <= r_1$

For even length

$$\text{median} = \frac{\max(l_1, l_2) + \min(r_1, r_2)}{2}$$

④ ✗

			l_1		r_1		
2	4	8	10	|	14	16	
			l_2		r_2		
			1	|	5	9	12

⑤ ✗

				l_1		r_1			
2	4	8	10	14	|	16			
				l_2		r_2			
				$-\infty$	|	1	5	9	12

~~6~~

For odd length :-

arr1

2	4	8	10	14
0	1	2	3	4

arr2

1	5	9	12
0	1	2	3

$n1 = 5$

$n2 = 4$

$$\frac{n1 + n2 + 1}{2} = \frac{5+4+1}{2} = \frac{10}{2} = 5$$

⇓

left side

Right side
$= 10 - 5$
$= 5$

1	2	4	5	8	9	10	12	14
0	1	2	3	4	5	6	7	8

2 4 8 10 14

1 5 9 12

$$\text{median} = \max(l_1, l_2)$$

Possible no. of elements that can be taken from arr1

0 1 2 3 4 5 6

So, we can apply binary search on this.

$T = O(\log n_1)$

So, we will always consider the smaller array i.e., n2

0 1 2 3 4

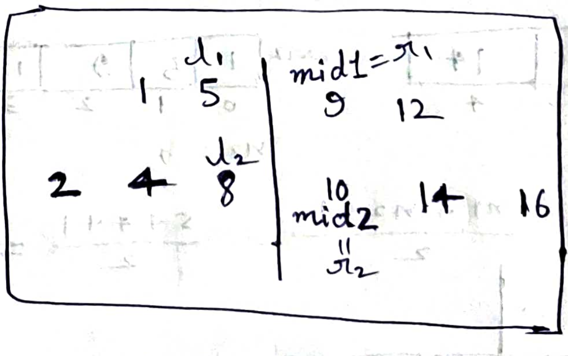
$$\boxed{T = O(\log(\min(n_1, n_2)))}$$

Implementation (using Binary search)

step 1   0 1 2 3 4

l   mid1   r

$$\boxed{mid1 = (l+r)/2}$$



$$\boxed{mid2 = leftSize - mid1}$$

$l1 <= r_2$

$l2 <= r_1$

So, median $= \dfrac{\max(5, 8) + \min(9, 10)}{2}$

$= \dfrac{8+9}{2}$

$= \boxed{\dfrac{17}{2}}$

Here, in 1st step only we got the answer

Example 2

arr1

2	3	6	15
0	1	2	3

arr2

1	3	4	7	10	12
0	1	2	3	4	5

$n1 = 4$

$n2 = 6$

$$\frac{n1+n2+1}{2} = \frac{4+6+1}{2} = 5 = \boxed{\text{left side}}$$

Right Side = $10 - 5 = 5$

Step 1

```
0     1     2     3     4
l           ↑           r
           mid1
```

$$\boxed{\begin{array}{l} mid1 = (l+r)/2 \\ mid2 = left - mid1 \end{array}}$$

```
        l1   mid1
             ↓
    2   3  |  6    15
        l2 |
1   3   4  |  7    10    12
              ↑
             mid2
```

$mid1 = 2$

$mid2 = 3$

$$\text{meidiam} = \frac{max(3,4) + min(6,7)}{2}$$

$$= \frac{4+6}{2} = \boxed{5}$$

Here, also bychance in first step only we got the answer ;-(

Example 3

arr1

2	6	6	15
0	1	2	3

arr2

1	3	4	5	10	12
0	1	2	3	4	5

$n1 = 4$ $n2 = 6$

$$leftSize = \frac{4+6+1}{2} = \boxed{5}$$

rightSize =

Step 1

```
0     1     2     3     4
l           ↑           r
           mid1
```

$mid1 = 2$

$mid2 = 3$

```
        l1     mid1=r1
  2     6   |  6    15

  1  3  4   |  5    10   12
        l2  |  mid2=r2
```

$\boxed{l_1 > r_2}$

So, $\boxed{r = mid1 - 1}$

Step2

```
mid1
 ↓
 0      1     2     3     4
 l      r
```

```
         mid1=r1
         ↙
    l1
    -∞  |  2   6   6   15
    l2
1 3 4 5 10  |  12
               ↑
            mid2=r2
```

$mid1 = \frac{0+1}{2}$

$= 0$

$mid2 = 5 - 0$

$= 5$

$\boxed{l_2 > r_1}$

So, $\boxed{l = mid1 + 1}$

Step3

✓

```
        mid1
         ↓
   0     1     2     3     4
         l
         r
```

$mid1 = 1$

$mid2 = 5 - 1 = 4$

```
       l1      mid1=r1
       2   |   6    6    15
       l2
1  3  4  5 |   10   12
               mid2
                 ||
                 r2
```

$$\boxed{\begin{matrix} l_1 <= r_2 \\ l_2 <= r_1 \end{matrix}}$$

$$median = \frac{max(2,5) + min(6,10)}{2}$$

$$median = \frac{5+6}{2} = \frac{11}{2} = \boxed{5}$$

```
       double
public int medianOfTwoSortedArr(int[] arr1, int[] arr2){
    int n1= arr1.length;
    int n2 = arr2.length;
    if(n2 < n1) {
        return medianOfTwoSortedArr(arr2, arr1);
    }

    int leftSize = (n1+ n2 +1)/2;
    int l = 0;
    int r = n1;
    while (l<=r) {
        int mid1 =(l+r)/2;
        int mid2 = leftSize - mid1;
        int l1 =Integer.MIN_VALUE; //-∞
        int l2 = Integer. MIN_VALUE; //-∞
        int r1 =Integer. MAX_VALUE; // +∞
        int r2 = Integer. MAX_VALUE; // +∞
        if(mid1 > 0) {
            l1 = arr[mid1-1];
        }
        if(mid2 > 0) {
            l2 = arr[mid2 -1];
        }
        if(mid1 < n1) {
            r1 = arr[mid1];
        }
        if(mid2 < n2) {
            r2 = arr[mid2];
        }
        if(l1 <=r2 && l2 <= r1) {
            if((n1+n2)%2 = =0) {
                return (max(l1,l2) + min(r1,r2))/2;
            } else {
                return max(l1, l2);
            }
        }
```

```
            else if( l1 > r2) {
                r = mid1 - 1;
            } else if( l2 > r1) {
                l = mid1 + 1;
            }
      }
   return 0;
}
```