approach

09/02/2026

Monday

Median of two sorted Arrays

arr1[] = [1 3 4 7 10 12]

arr2[] = [2 3 6 15]

```
1 3 4 7 | 10 12
      2 |    3 6 15

   5         5
```

First here we taken 4 elements from array 1 and 1 element from array2 and checking conditions

Here max is 7 and in ryt side low/min is 3 so it does not form

Now take 3 elements from arr1 and 2 elements from arr2

1 3 4 | 7 10 12

2 3 | 6 15

Now we check conditions max in left side is 4 and min in right side is 6 so it is in sorted order

Now take 2 elements from arr1 and 3 elements from arr2

1 3 | 4 7 10 12

2 3 6 | 15

Now we check conditions max in left is 6 and min in right side is 4 so it is not in sorted order.

Now to calculate median

l_1 r_1

1 3 4 | 7 10 12

2 3 l_2 | r_2 6 15

we take $\frac{\max(l_1, l_2) + \min(r_1, r_2)}{2}$

if $n_1+n_2 \% 2 == 0$ even

otherwise

only $\max(l_1, l_2)$

Now

④

1 3 4 7 l_1 | r_1 10 12

2 l_2 | r_2 3 6 15

$l_1 > r_2$

high = mid - 1

1 3 l_1 | r_1 4 7 10 12

2 3 6 l_2 | r_2 15

$l_2 > r_1$

low = mid + 1

if ($l_1 <= r_2$ && $l_2 <= r_1$)
Valid

1 3 4 | 7 10

2 3 | 6 15

we do in binary search

I will take smaller array bcuz for max I can put smaller elements at one side so when compared to bigger ~~array~~ time complexity is small

arr1 [] = [7 12 14 15]

arr2 [] = [1 2 3 4 9 11]

0
low

mid 1
2

4
high

so. left over are 5-2 = 3   mid2 = 3

7 12 l_1 | 14 r_1 15

1 2 3 l_2 | 4 r_2 9   11

$l_1 > r_2$ Condition fails I mean 12<4 false

so ~~low=~~ high = mid-1

low   mid   high

0   0   1

left over are 5

$\emptyset$ are these

$-l_1$ | 7 r_1 12 14 15

1 2 3 4 9 l_2 | 11 r_2

so we assume

Integer. min_value

so here $l_2 > r_1$

so we move right side

low = mid+1   mid1 = 1

and leftover are 5-1

mid2 = 4

low   mid   high

1   1   1

mid1

7

1 2 14 15 | 7 l_1 | r_1 12 14 15

1 2 3 4 l_2 | r_2 9 11

mid2

Here $l_1 < r_2$ and $l_2 < r_1$

so valid

So now based on total length

we find median

if length is even we find

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

else
  we take max(l1, l2)

```
public int solve (int[] a, int[] b) ~~int n1,int~~
{       int n1=a.length;
          int n2=b.length
      if (n1>n2)
         return solve(b,a) n2,n1)

      int low=0
      int high= a.length;  -> int left= (n1+n2+1)/2;
      while (low<=high)    -> int n=n1+n2;

      {
         int mid1 = low+ (high-low)/2;
         int mid2 = left-mid1;
         int l1 = Integer. MAIN_VALUE;
         int l2 =     "         "   ;
         int r1 = Integer. MAX_VALUE;
         int r2 =     "        "
         if (mid1<n1)  r1 = a[mid1];
         if (mid2<n2) r2 = b[mid2];
          if(mid1-1>=0)  l1 = a[mid1-1];
         if (mid2-1>=0) l2= ~~a~~ b[mid2-1];

        if (l1<=r2 && l2<=r1) {
              if (n%2==1)  return
                        Math.max(l1, l2);
           return (double) (Math.max (l1,l2)+
        }                       Math.min(r1,r2)/2)
        else if(l1>r2)
         {
              high=mid1-1;
          }
          else
          {
            low=mid1+1;
          }
      }
     return 0;
  }
```