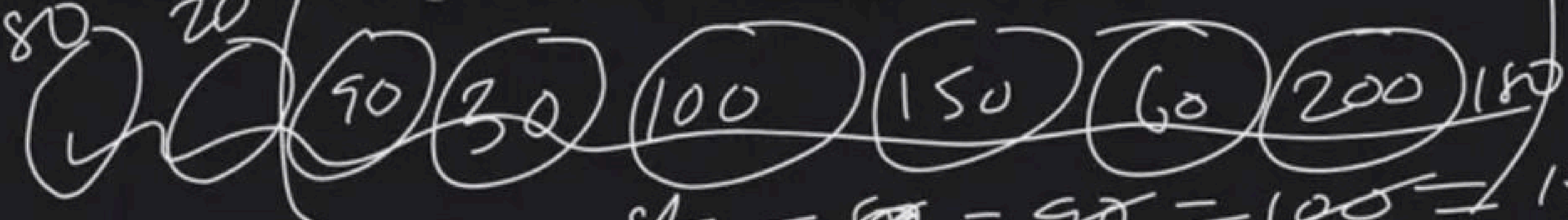
DP - Part VI

Complete Course on Algorithm for GATE - CS & IT

Subbarao Lingamgunta • Lesson 30 • Jan 3, 2023

Doubt Clearing Session

Complete Course on Algorithm for GATE - CS & IT



$max_1 = 1^{st} = \cancel{80} = \cancel{90} = \cancel{100} = 150 = 200.$

$max_2 = 2^{nd} = \cancel{20}\ \cancel{80} = \cancel{90} = \cancel{100} = 150 = 180$

Subbarao Lingamgunta • Lesson 29 • Jan 2, 2023

Sorted

	2-ele	3 ele	4 ele
① LS	n^2	n^3	n^4
② BS	$n\log n$	$n^2\log n$	$n^3\log n$
③	$O(1)$	$O(1)$	$O(1)$

Unsorted

① LS ⟹ n^2

② BS ⟹ 1. sort — $n\log n$
2. BS — $n\log n$ } $2n\log n$ ⟹ $n\log n$

③ 1. sort — $n\log n$
2. return last 2-ele — $O(1)$ } $n\log n$

④ 1. find 2-max ⟹ $O(n)$
2. return (1 time)

$O(n)$

i/p: sorted array of n- distinct ele

o/p: find any 2-ele (a, b), such that $a+b < 1000$

TC ?

① LS — n^2 ② BS — $n\log n$ sorted

③ return (first 2-ele) — $O(1)$

unsorted

① LS — n^2 ② BS — 1. sort 2. BS $\Longrightarrow n\log n$

③ 1. sort 2. return (1st 2 ele)

④ find 2-mins return (them) $\Longrightarrow O(n)$

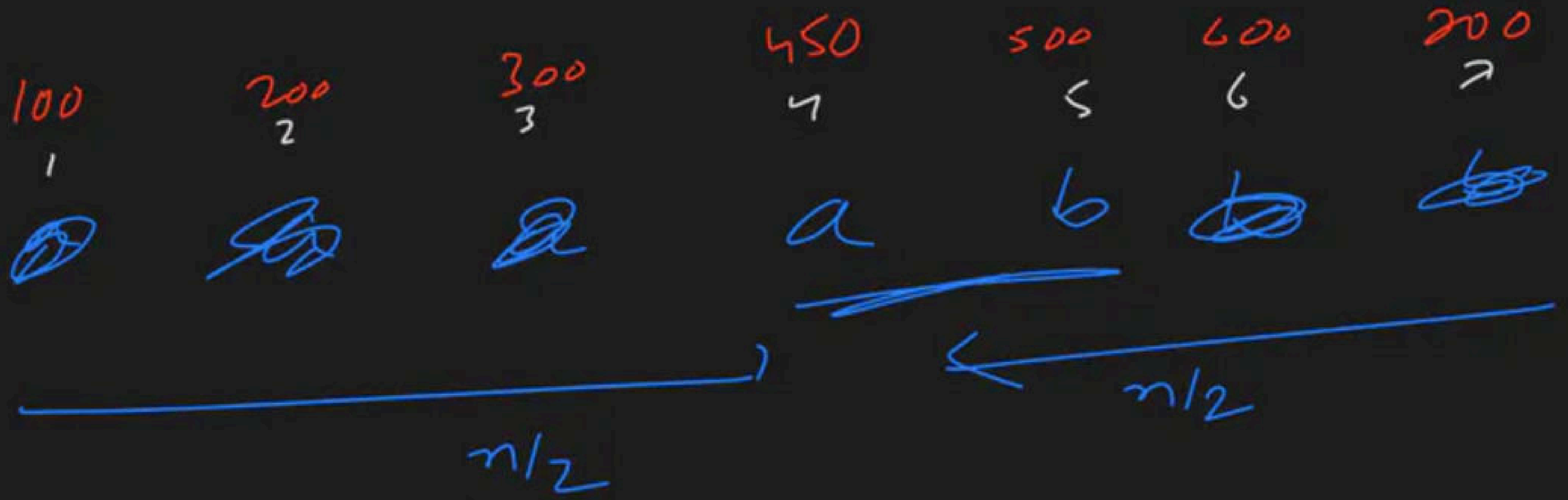
③

eg

i/p: sorted array of n-distinct ele

o/p: find any 2-ele (a,b) such that a+b == 1000

TC = ?



① $n * LS \Longrightarrow n^2$ ② $n * BS \Longrightarrow n\log n$

③ ① i = 1st position   j = Last position

```
while( i != j )
{
    if(a[i] + a[j] == 1000)  ret (a[i], a[j])
    else
    if( a[i] + a[j] > 1000)
            j = j-1
    else
        i = i+1
}
```

O(n)

Greedy Algo

unsorted

1. LS $\Longrightarrow n^2$ 2. BS $\Longrightarrow$ 1. Sort, 2. BS $\}$ $2 n\log n$

3 ① Sort — $n\log n$
② Greedy Algo — n

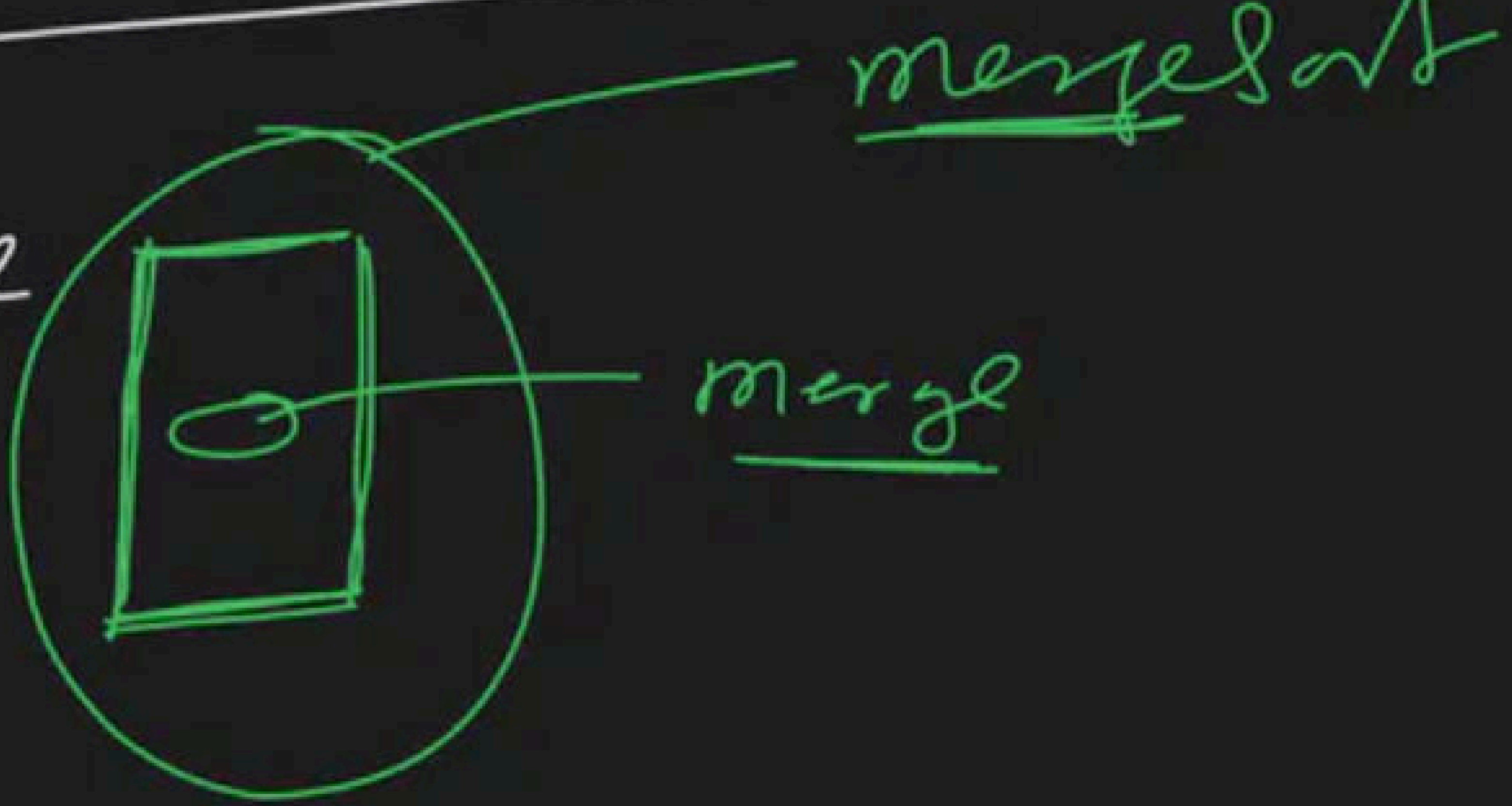
$n\log n$

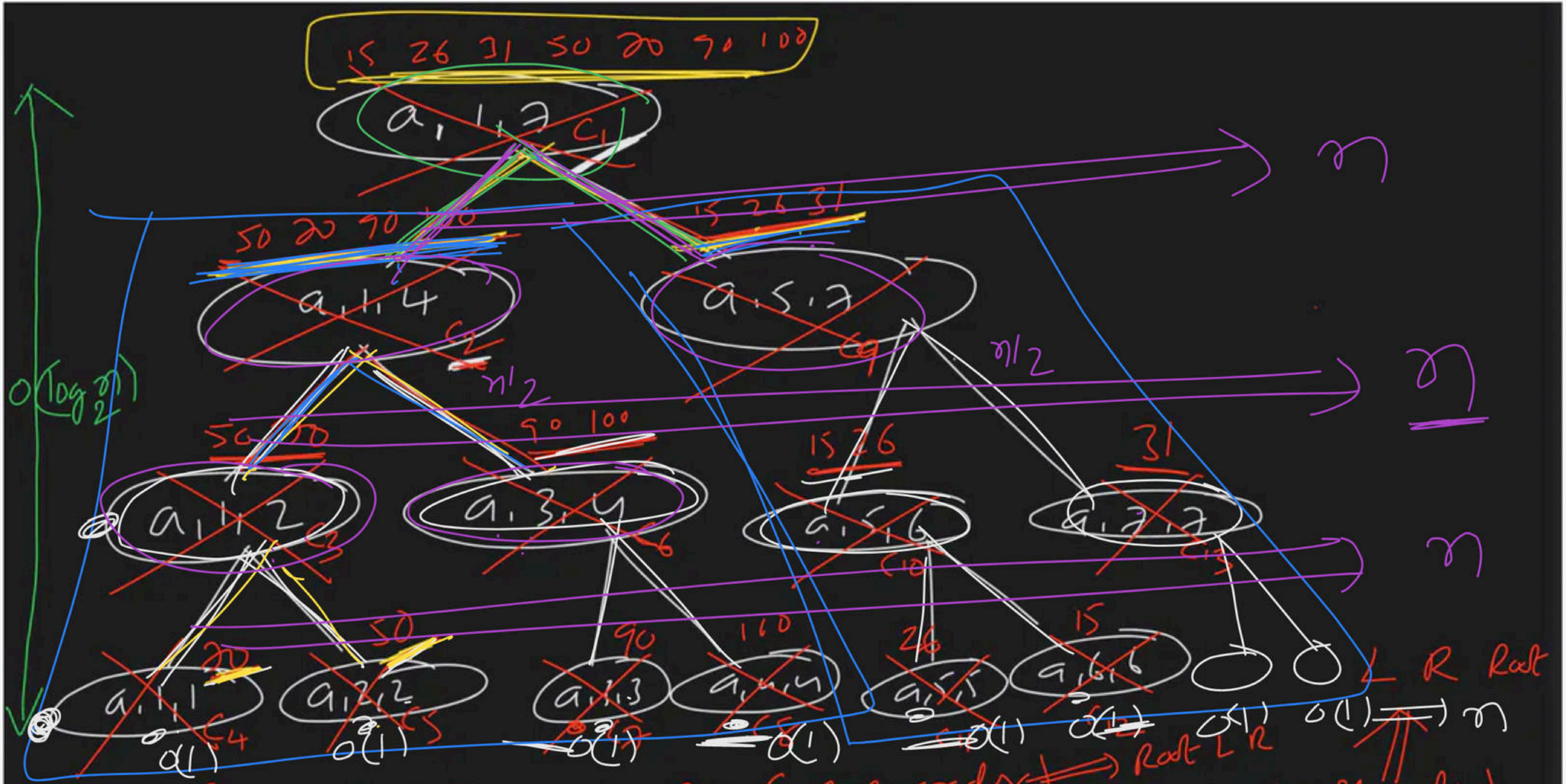
MergeSort

Note: Merging 2-sorted Subarrays is known as MergeSort.

i/p : Array of n-ele

o/p : sorted array.

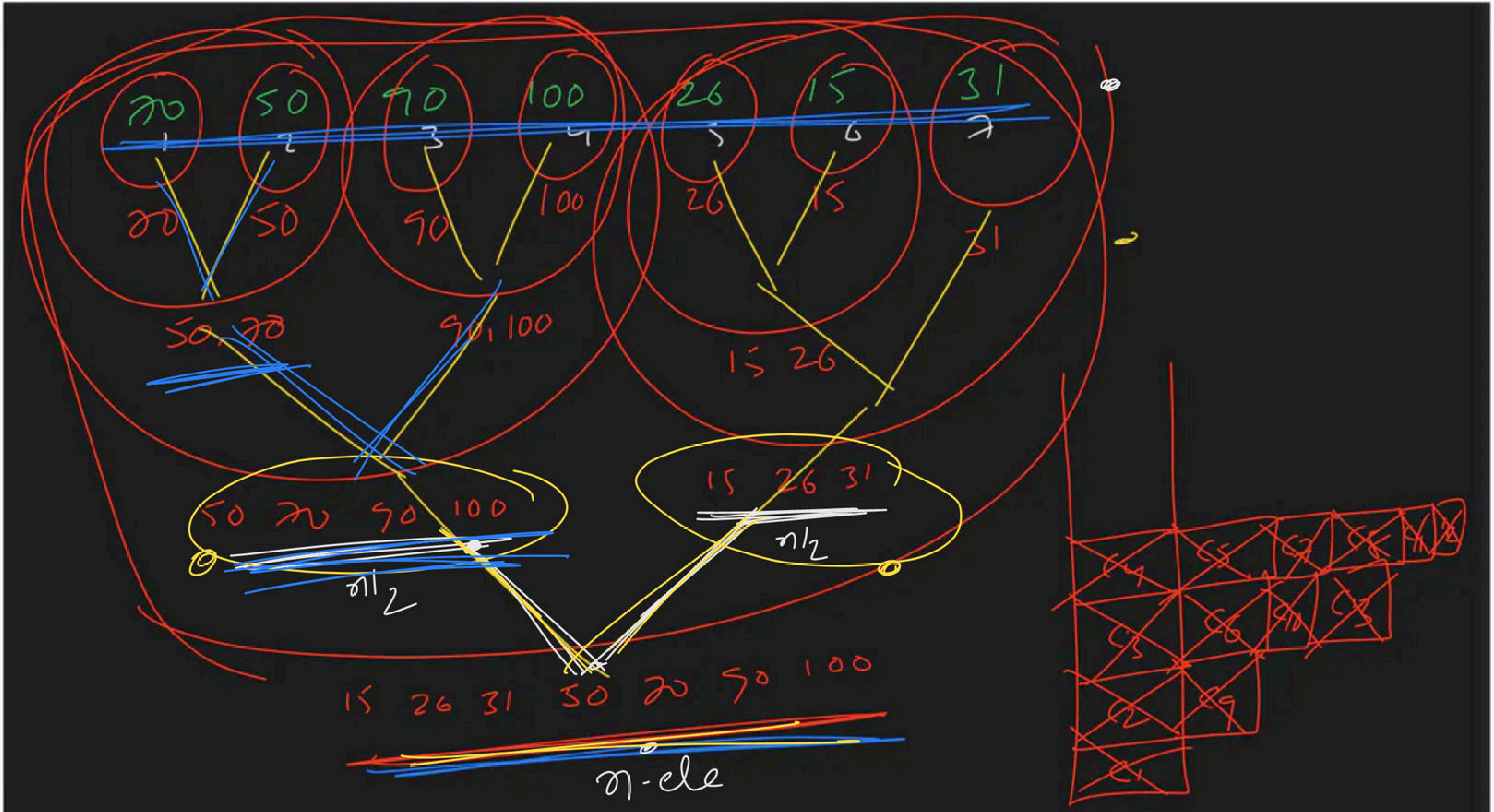




f.C = c1 c2 c3 - - - - . c13 (pre-order) ⟹ Root L R

f.E = c4 c5 c3 c7 c8 c6 c2 c11 c12 c10 c13 c9 c1 (postorder)

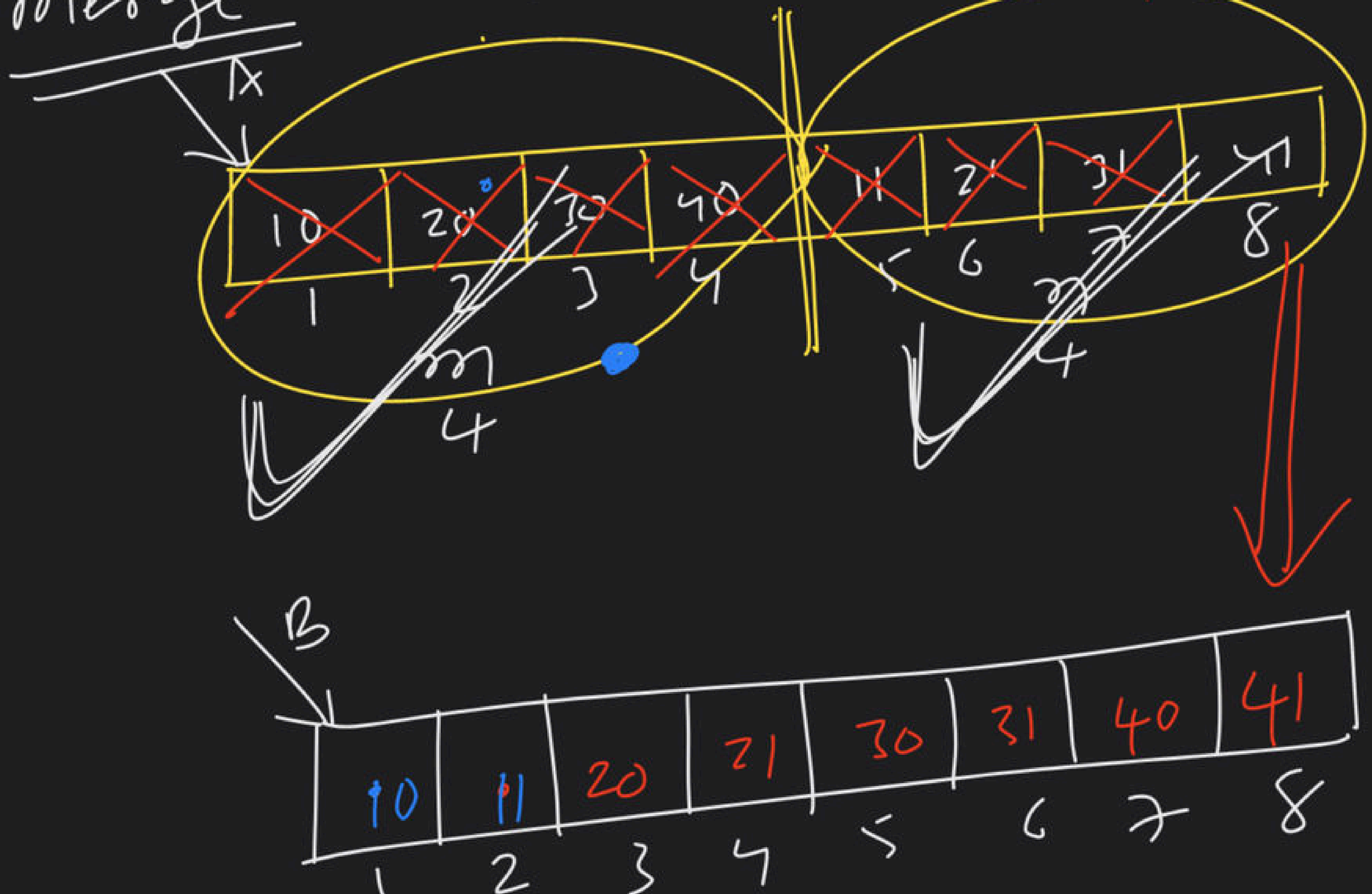
moves


70 50 90 100 26 15 31
1 2 3 4 5 6 7
70 50 90 100 26 15 31
50, 70
90, 100
15 26
50 70 90 100
15 26 31
n/2
n/2
15 26 31 50 70 90 100
n-ele
C4
C5
C2
C8
C3
C6
C10
C7
C9
C1

Merge

Worst case

Outplace



10, 11 ⇒ 10

20, 11 ⇒ 11

20, 21 ⇒ 20

30, 21 ⇒ 21

30, 31 ⇒ 30

40, 31 ⇒ 31

40, 41 ⇒ 40

41

comparisions + moves

$m, n \Rightarrow m + n - 1 + m + n$

$4, 4 \Rightarrow 4 + 4 - 1 + 4 + 4$

$n/2, n/2 \Rightarrow n/2 + n/2 - 1 + n/2 + n/2$

$\Rightarrow$ moves TC

$m + n$

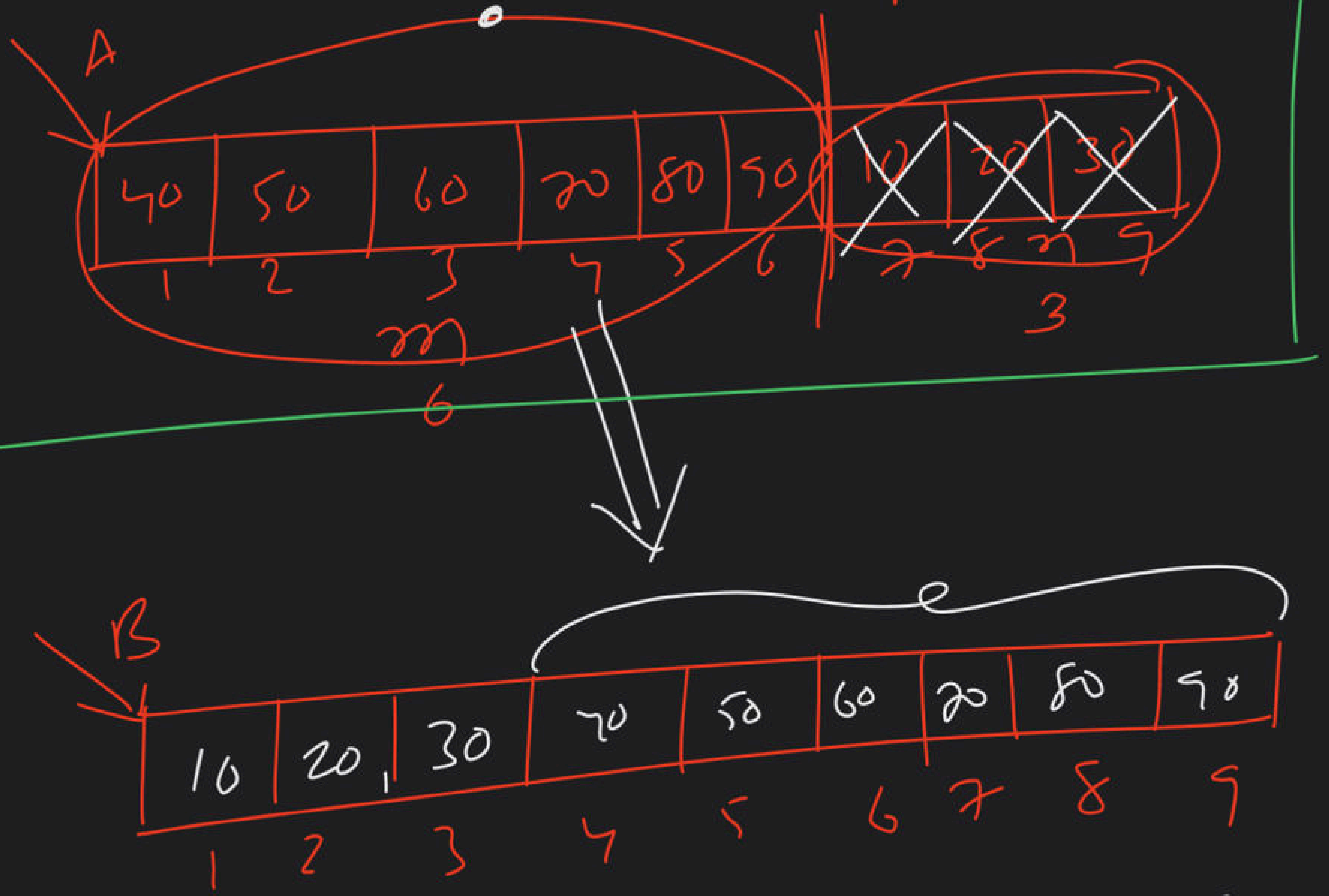
$4 + 4 \Rightarrow 8$

$n/2 + n/2 \Rightarrow n$

ex

Best case

Outplace



40, 10 ⟹ 10

40, 20 ⟹ 20

40, 30 ⟹ 30

40
50
60
70
80
90

$\frac{n/2,\ n/2}{m,\ n}$ ⟹ moves $\frac{n/2 + n/2}{m + n}$ + comparisions $\frac{n/2 \Leftarrow n/2,\ n/2}{\min(m, n)}$ = moves $n/2 + n/2 \Rightarrow n$ TC

6, 3 — 6 + 3 — 3 = 6, 3 — $m + n$

6 + 3

Note: Merging 2 sorted subarrays each of size m & n will take

$\theta\theta\theta\theta\theta\theta$

$$\overset{n/2 + n/2 \Rightarrow n}{}$$

$$\Longrightarrow \Theta(m+n) \quad \left[\text{EC becz moves are always } m+n\right]$$

$$10+20$$

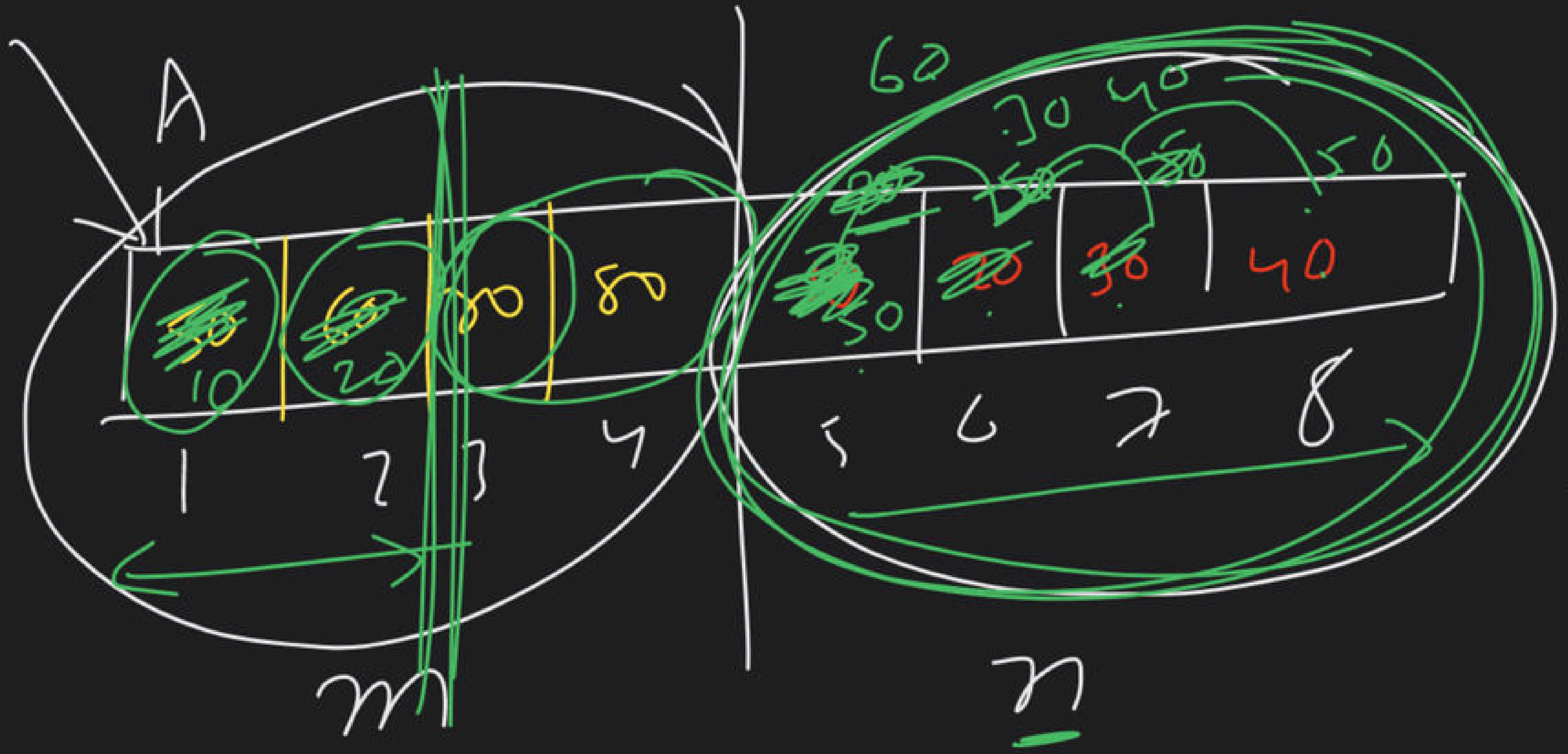
outplace

$$10\cdot 20$$

$$\Longrightarrow O(m\cdot n) \quad \left\{\begin{matrix}\text{WC} \\ \text{Inplace}\end{matrix}\right]$$

$$\frac{n}{2}\cdot\frac{n}{2} \Longrightarrow \left(\frac{n}{2}\right)^2 \Longrightarrow n^2$$

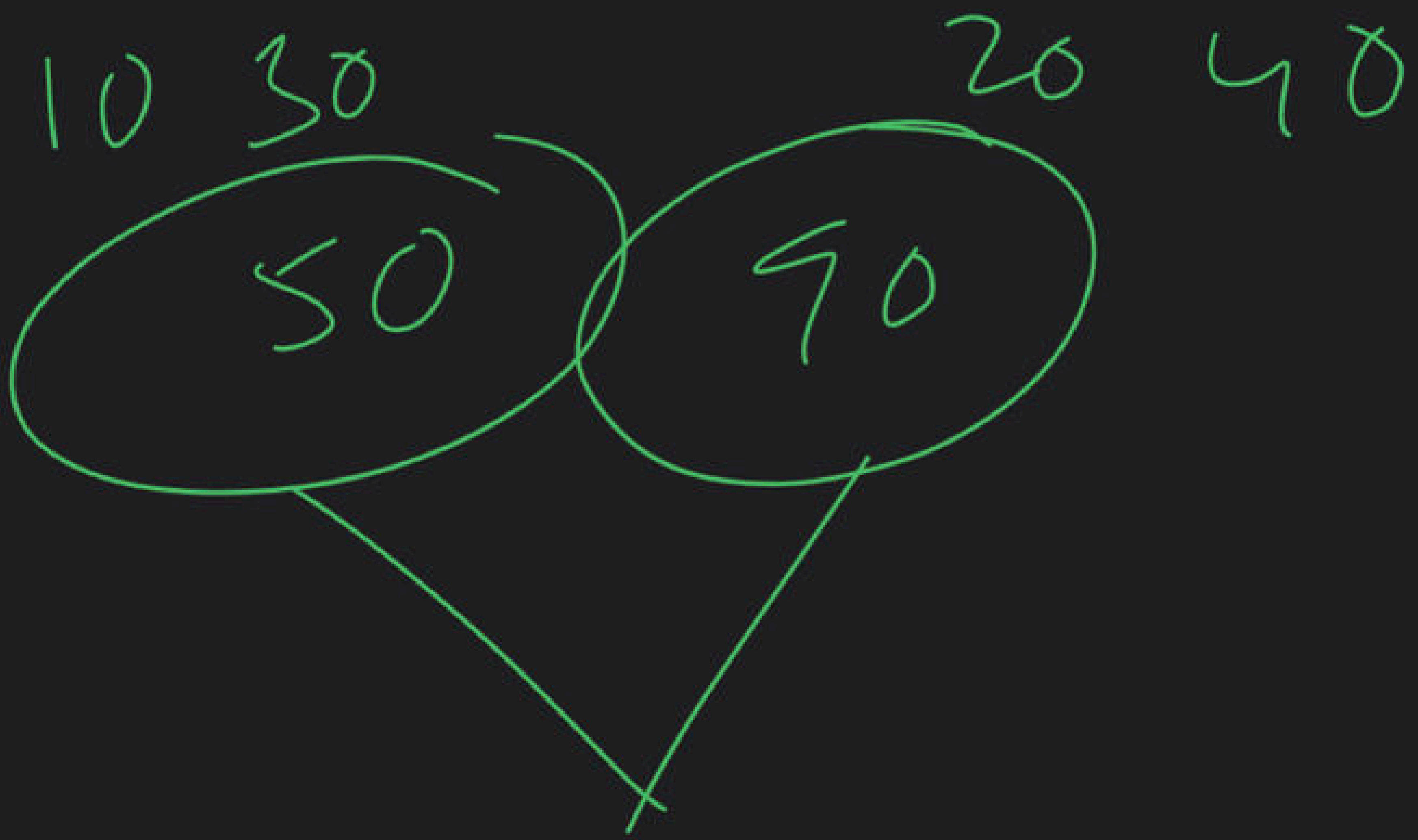
TC



$$m\left\{\begin{array}{l} 10,50 \Rightarrow 10 \Rightarrow n \\ 60,20 \Rightarrow 20 \Rightarrow n \\ \qquad\qquad\quad \Rightarrow n \\ \qquad\qquad\quad \Rightarrow n \\ \qquad\qquad\quad \Rightarrow n \end{array}\right.$$

$$m \cdot n$$

$$n/_2 \cdot n/_2 \Rightarrow \frac{n^2}{4} = O(n^2)$$



min-compr ⇒ 50

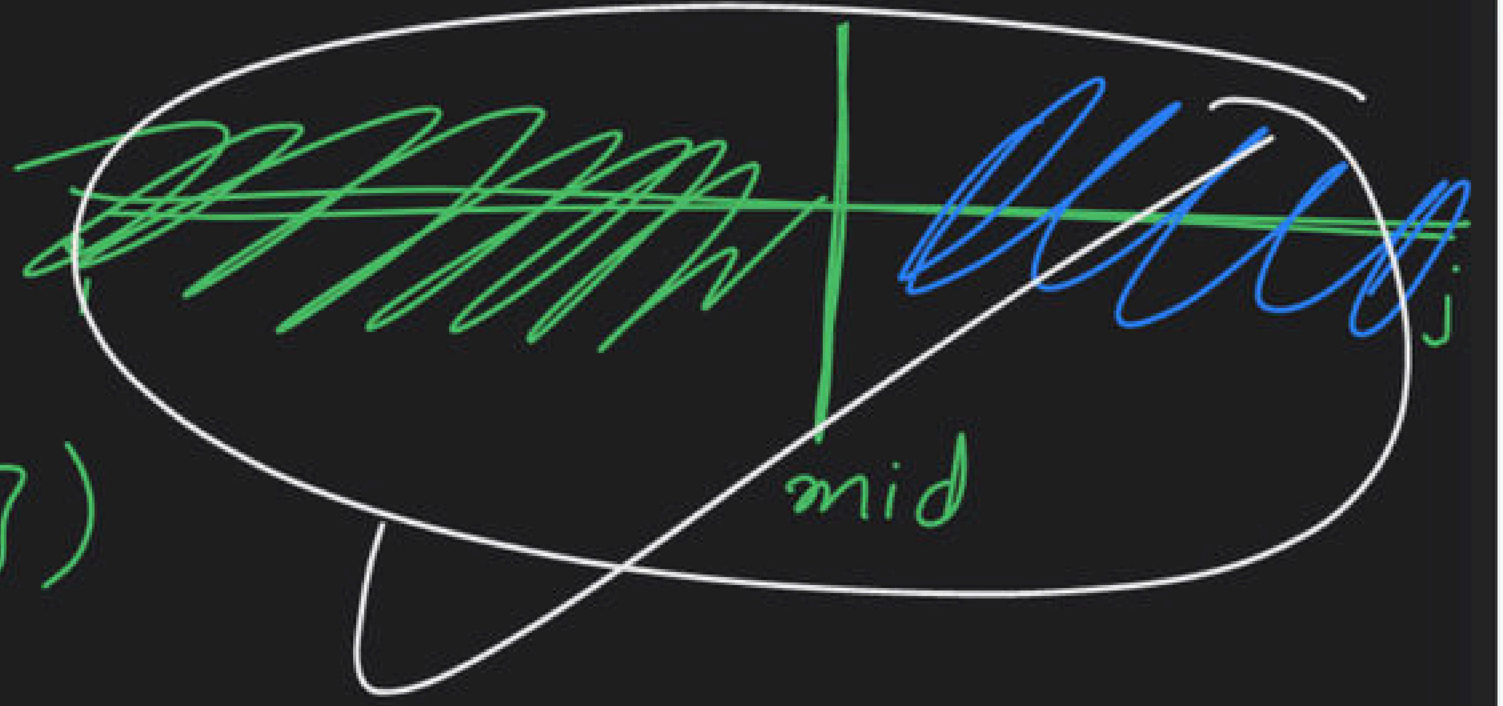
max- " ⇒ 50+90-1

min-moves ⇒ 50+90

max- " ⇒ 50+90

$\Big\}$ TC

MergeSort - Algo [outplace]



```
MergeSort(a, i, j)  ⇒ $T(n)$
{
    if(i==j) return(a[i])          $O(1)$
    else{
        mid = $\lfloor (i+j)/2 \rfloor$   ⇒ $O(1)$
        MergeSort(a, i, mid);      ⇒ $T(n/2)$ ⇒ $2T(n/2)$
        MergeSort(a, mid+1, j);    ⇒ $T(n/2)$
        Merge(a, i, mid, j)        ⇒ $n/2 + n/2$ ⇒ $\Theta(n)$
                                                    EC
                  outplace
        return(a)
    }
}
```

Let $T(n)$ be the TC of above algo

RR-TC

$$T(n) = \begin{cases} O(1) & \text{if } n = 1 \\ O(1) + 2T(n/2) + \Theta(n) & \text{if } n > 1 \end{cases}$$

$$T(n) = 2T(n/2) + n$$

$$= 2^2 T(n/2^2) + n + n$$

$$= 2^3 T(n/2^3) + n + n + n$$

$\log n$

$$= 2^{\log n} T(n/2^{\log n}) + n \cdot \log n$$

$$= n \cdot T(1) + n \log n$$

$$= n \cdot O(1) + n \cdot \log n$$

$$= n + n \log n = \Theta(n \log n)$$

EC

TC

Space comp

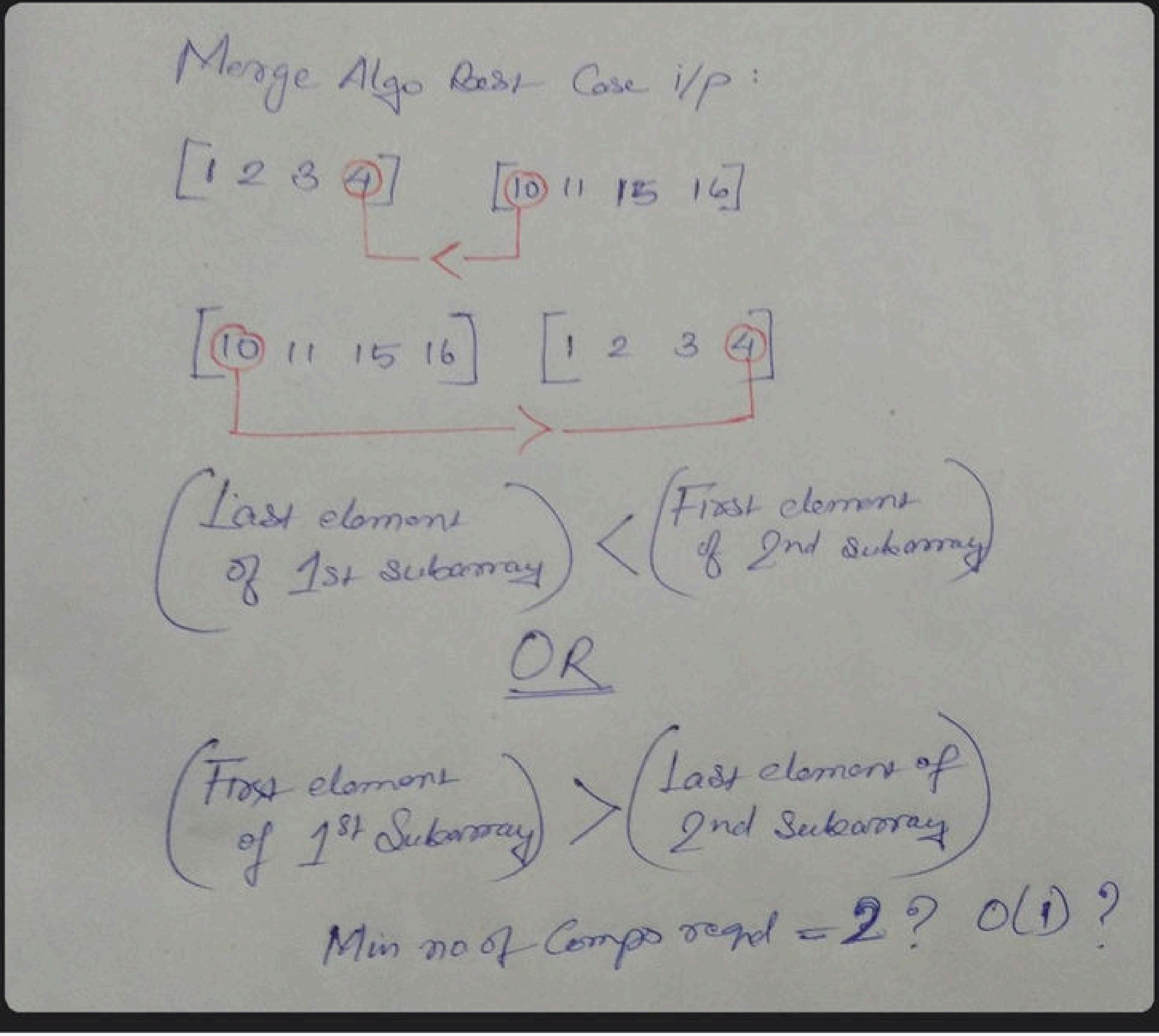
Stack merge

$$\underbrace{\log n + n}_{n}$$

$$n = O(n)$$

▲ 1 • Asked by Mayur Prat...

Sir if we detect the best case i/p by there 2 comparisons we can directly copy, min comps 2 ie 0(1)?



Merge Algo Best Case i/p :

[1 2 3 4]   [10 11 15 16]

[10 11 15 16]   [1 2 3 4]

(Last element of 1st subarray) < (First element of 2nd subarray)

OR

(First element of 1st Subarray) > (Last element of 2nd Subarray)

Min no of Comps reqd = 2 ? O(1) ?