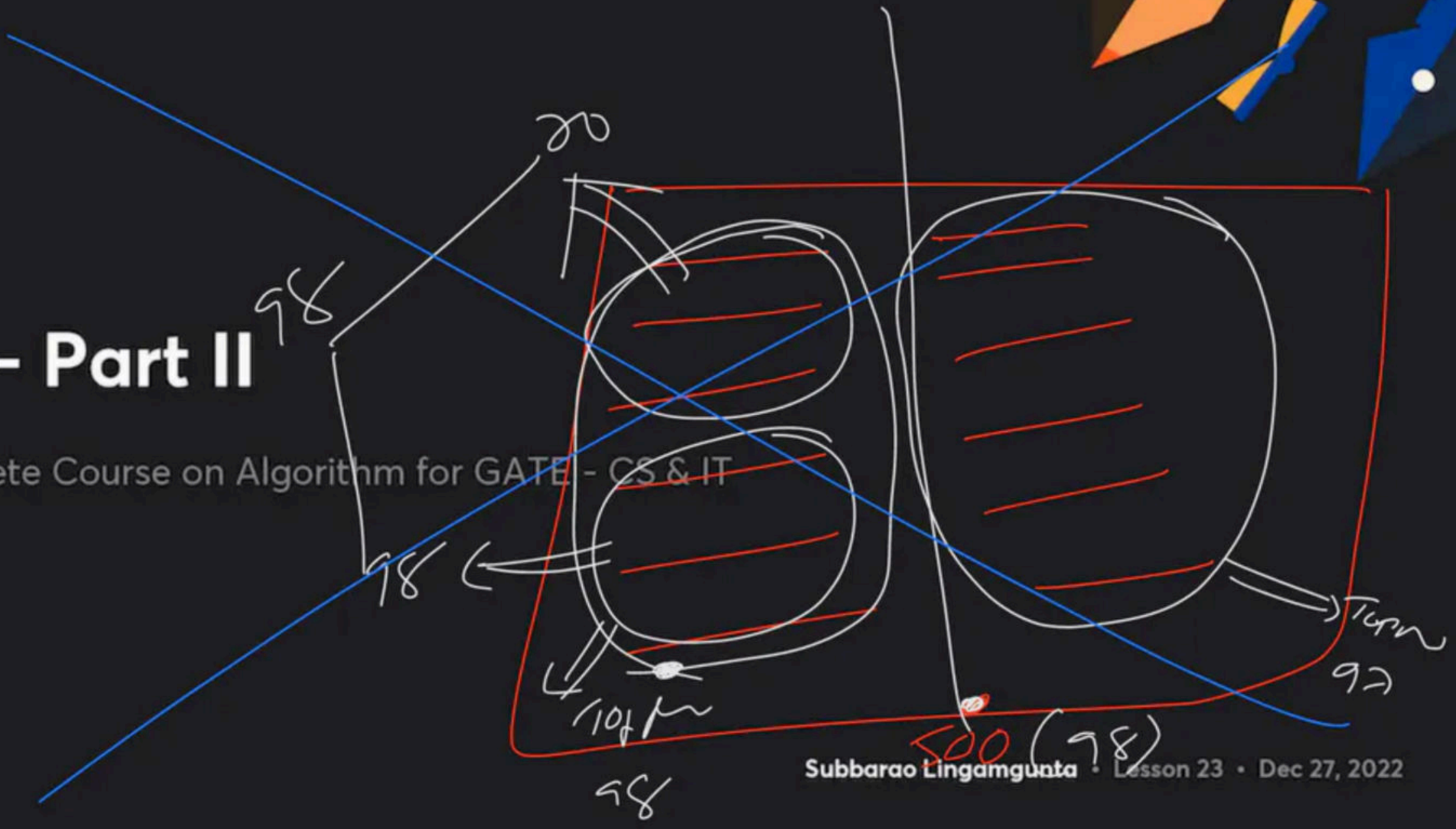
Doubt Clearing Session

Complete Course on Algorithm for GATE - CS & IT

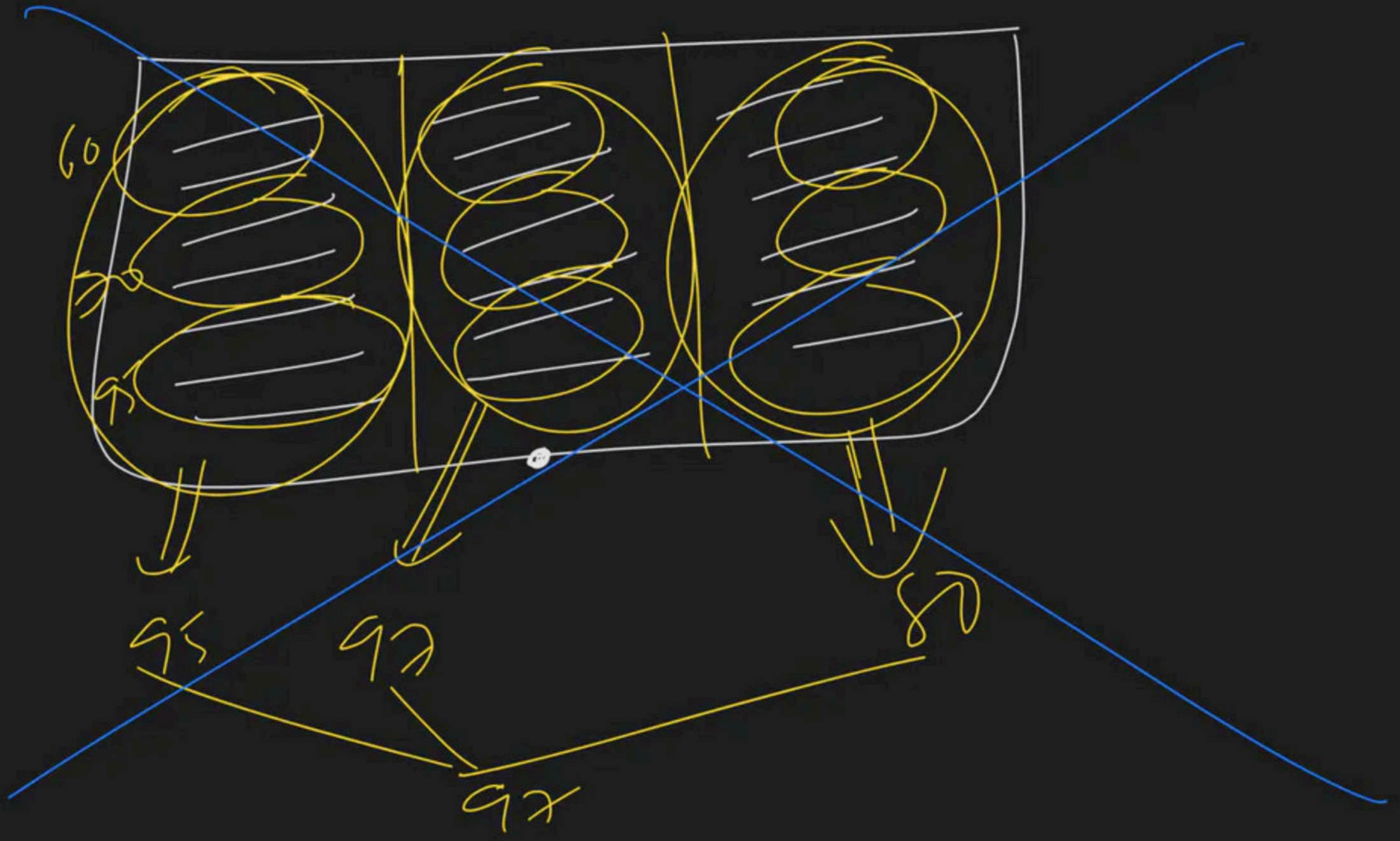
Subbarao Lingamgunta • Lesson 24 • Dec 28, 2022

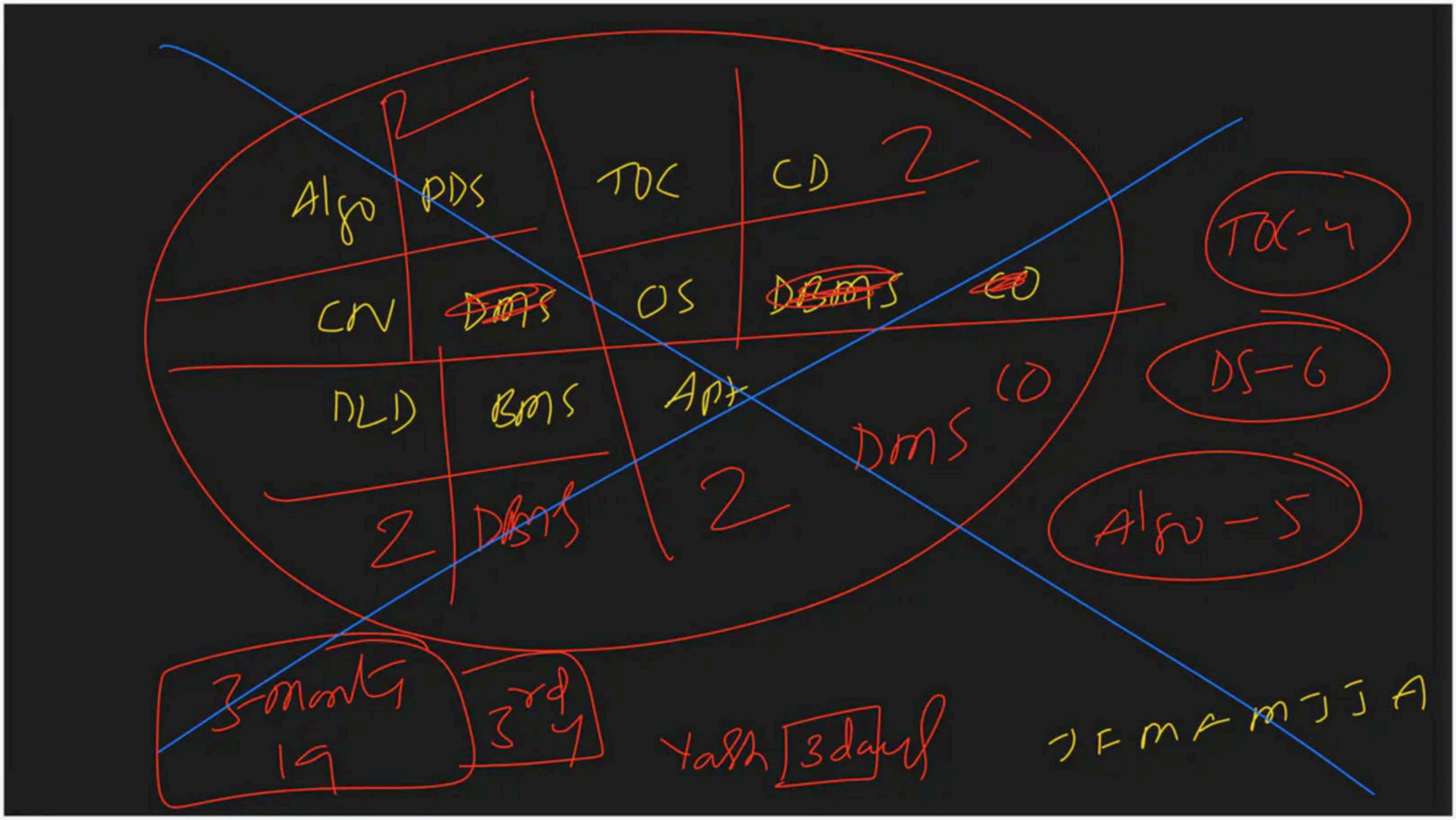
DP - Part II

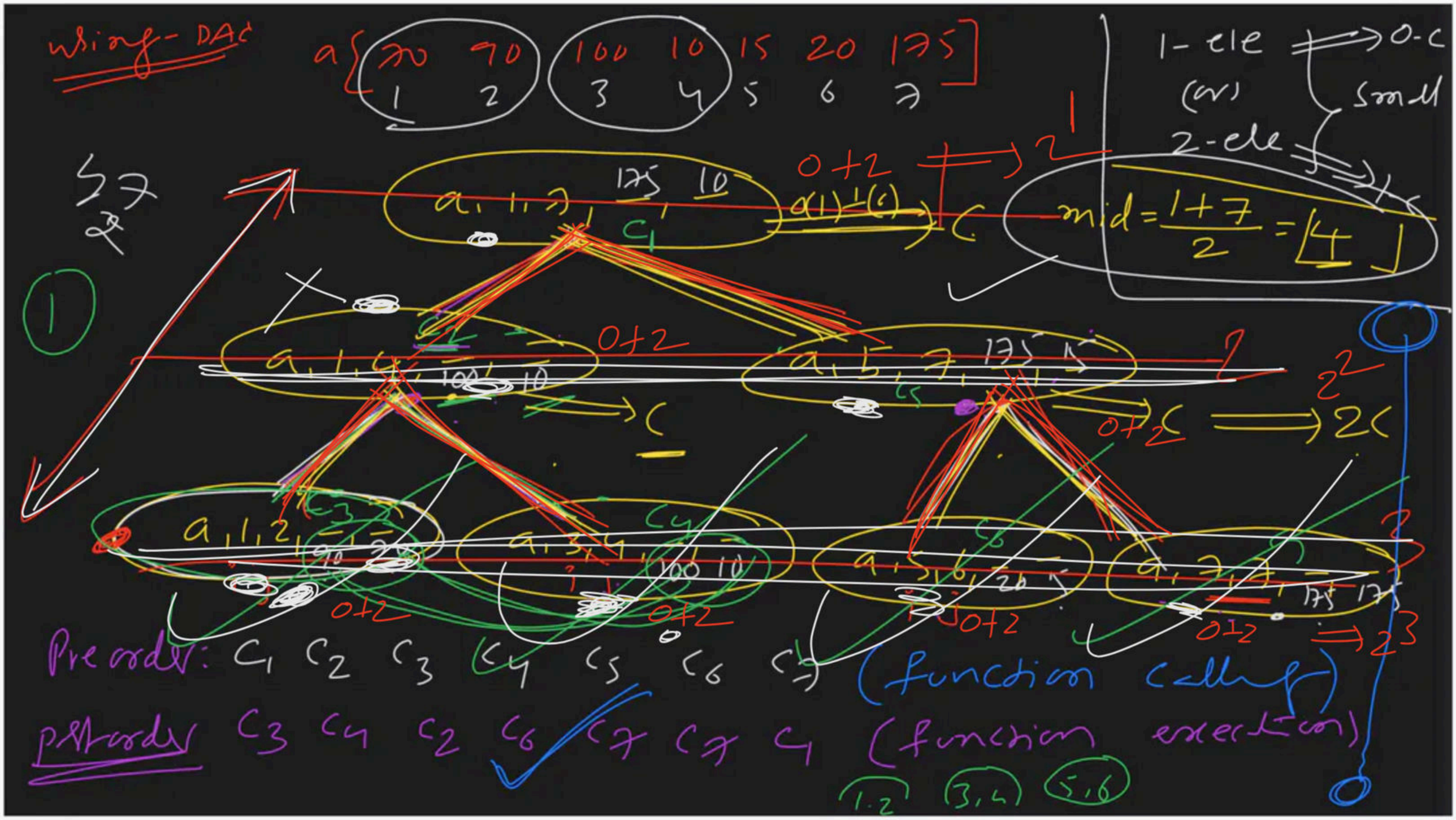
Complete Course on Algorithm for GATE - CS & IT



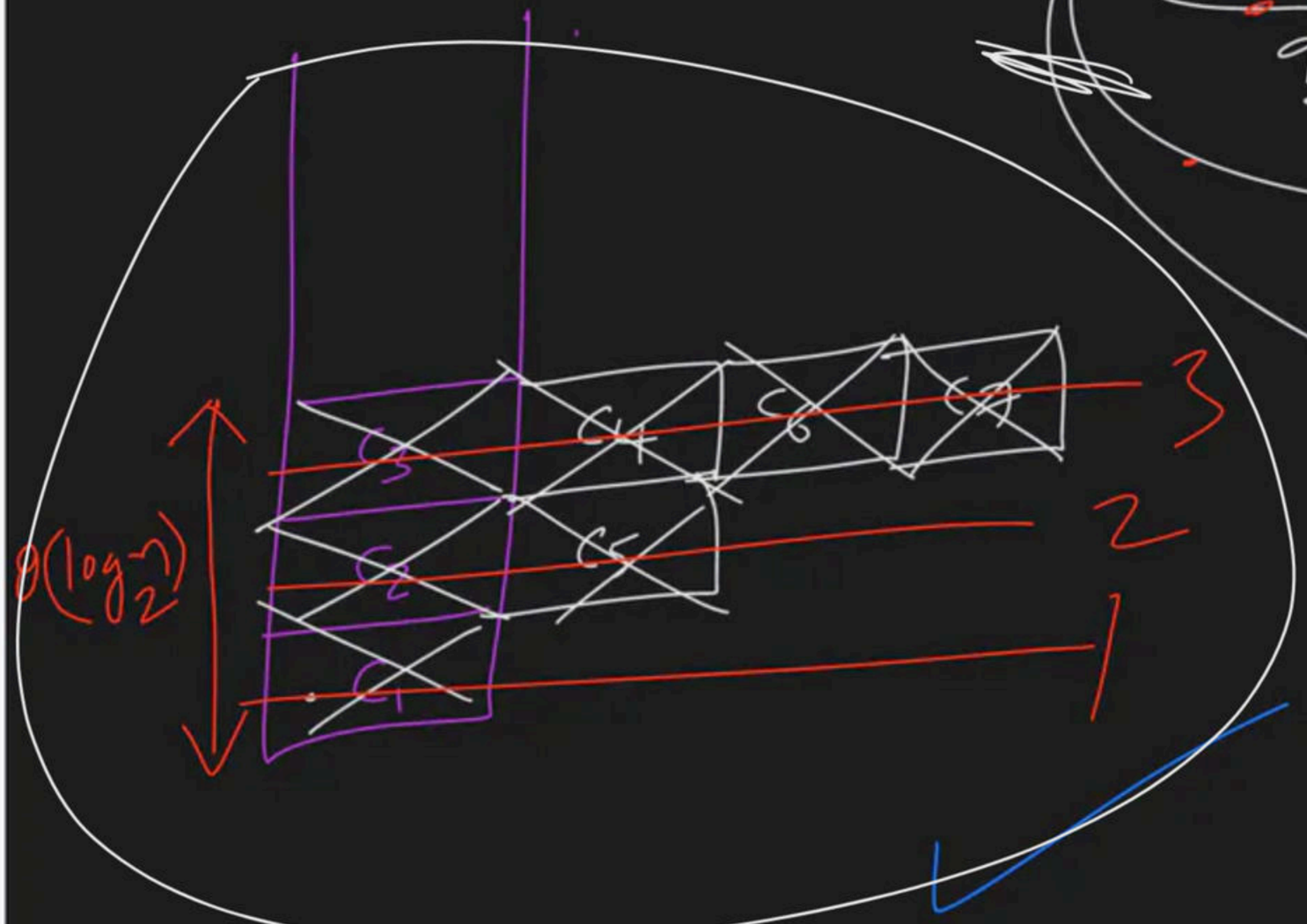
Subbarao Lingamgunta • Lesson 23 • Dec 27, 2022

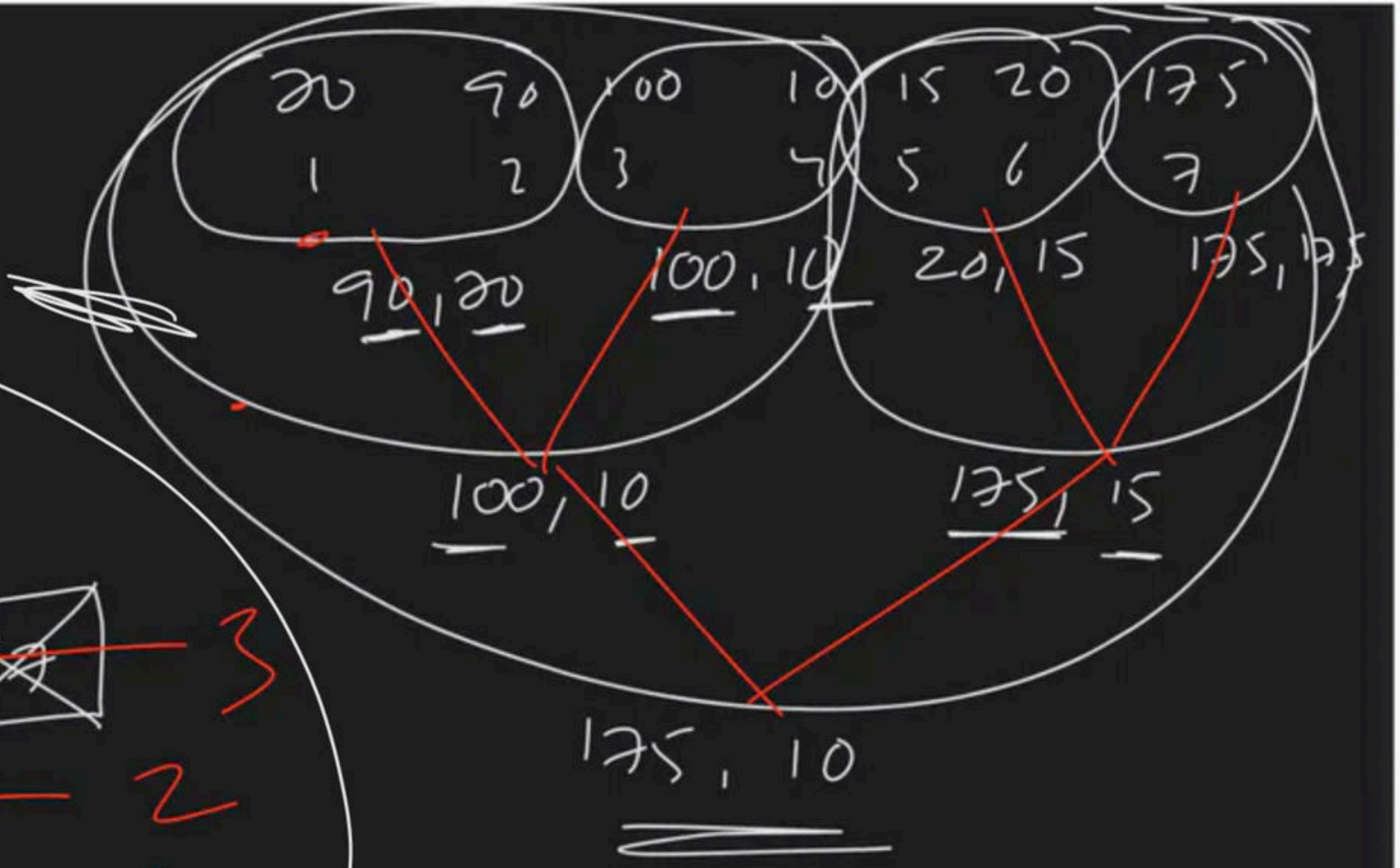



Algo
PDS
TOC
CD
CN
OS
NLD
Apt
DMS
CO
TOC-4
DS-6
Algo-5
3-month
19
3rd
3 4
Yash
3days
JFMAMJJA


using - DAC
a[20 90 100 10 15 20 125]
1 2 3 4 5 6 7
1- ele (or) 2-ele
0-c
small
mid = 1+7/2 = 4
0+2
1
2
1
2
2²
2c
Preorder: c1 c2 c3 c4 c5 c6 c7
(function calls)
preorder c3 c4 c2 c6 c7 c7 c1 (function execution)
1,2
3,4
5,6
23

2


C_5
C_4
C_6
C_7
C_3
C_2
C_1
3
2
1
$\theta(\log_2 7)$


20 90 100 10 15 20 175
1 2 3 4 5 6 7
90, 20
100, 10
20, 15
175, 15
100, 10
175, 15
175, 10

DACmaxmin(a, i, j) ⟹ T(n) ∈ (n)

O(1)

Small & Solution

if(i == j) { max = min = a[i] return(max, min) } 0

if(i == j-1) { if(a[i] > a[j]) max = a[i], min = a[j] else max = a[j], min = a[i] return(max, min) } 1

3

else { ① mid = ⌊(i+j)/2⌋ } Divide O(1) 0

Conquer

② (max1, min1) = DACmaxmin(a, i, mid)

(max2, min2) = DACmaxmin(a, mid+1, j)

T(n/2) ∈ (n/2)

T(n/2) ∈ (n/2)

(n-1)2 ⟹ 2n-2

③ Combine

if(max1 > max2) max = max1 else max = max2

if(min1 > min2) min = min2 else min = min1

return(max, min)

O(1)

TC of above program

RR - Time

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

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

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

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

$$= 2^3 + (n/2^3) + 2^2C + 2^1C + 2^0C$$

$$= 2^{\log n} T(1) + \underbrace{2^0C + 2^1C + \cdots + 2^{\log n - 1}C}_{\log n}$$

(4)

Stack space.

8
4
2
1

make it to

$f(n)$	n
C	n

big

$\Theta(n)$

Let $C(n)$ be the no. of comparisions between the elements in the above algo on n-ele array.

RR- comparisions

$$C(n) = \begin{cases} 0 & \text{if } n = 1 \\ 1 & \text{if } n = 2 \\ 0 + C(n/2) + C(n/2) + 2 & \text{if } n > 2 \end{cases}$$

6

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

$= 2[2C(n/2) + 2] + 2$

$= 2^2 C(n/2) + 2^2 + 2^1$

$2^3 T(n/2^3) + 2^3 + 2^2 + 2^1$

$\downarrow$ k times $= \frac{n}{2} - 1$

$2^k T(n/2^k) + 2^k + 2^{k-1} + 2^{k-2} + \cdots + 2^1$

$n = 2$

2^k

$n = 2^{k+1}$

$\log n = k+1$

$k = \log n - 1$

8

4

2

$$C(n) = 2^{\frac{n}{2}-1} C\left(n/2^{(2-1)}\right) + 2^1 + 2^2 + 2^3 + \cdots + 2^{\frac{n}{2}-1}$$

$$= \frac{n}{2} C(2) + \frac{2\left(2^{\frac{n}{2}-1} - 1\right)}{2-1}$$

$$= \frac{n}{2} \cdot 1 + 2^{\frac{n}{2}} - 2$$

(7) $$= \frac{n}{2} + n - 2$$

$$= \boxed{\frac{3n}{2} - 2} \Longrightarrow \boxed{1.5n - 2} = \Theta(n)$$

To find max & min in 1t array of 100 elements, comparisons needed

(8) $\Longrightarrow 1.5n - 2 \Longrightarrow 1.5 * 100 - 2 \Longrightarrow 150 - 2$

$\Longrightarrow 148$

To find max & min, TC ?

- $\Longrightarrow O(n)$ [Best Algo, EC]

To find max & min, where array already sorted, TC?

- $\Longrightarrow O(1)$ [Best Algo, EC] $\Longrightarrow$ return(a[1], a[n])

In the given sorted array of n-distinct elements find any element which is neither maximum nor minimum!

⟹ Bell_Algo

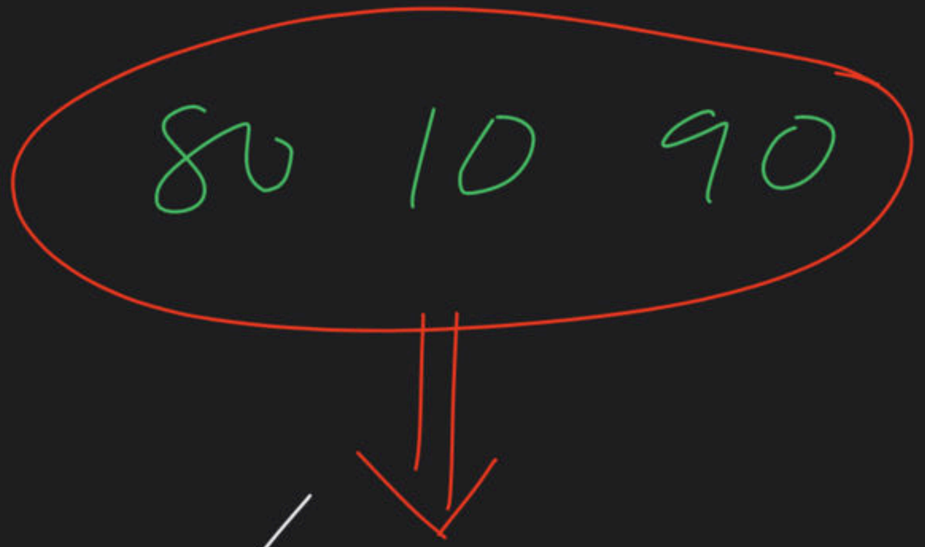
⟹ return (2nd element)

⟹ O(1) [EC]

In the given array of n-distict elements find any element which is neither max nor min?

⟹

80 10 90 ≤ 16 32 100 3 400



10 80 90

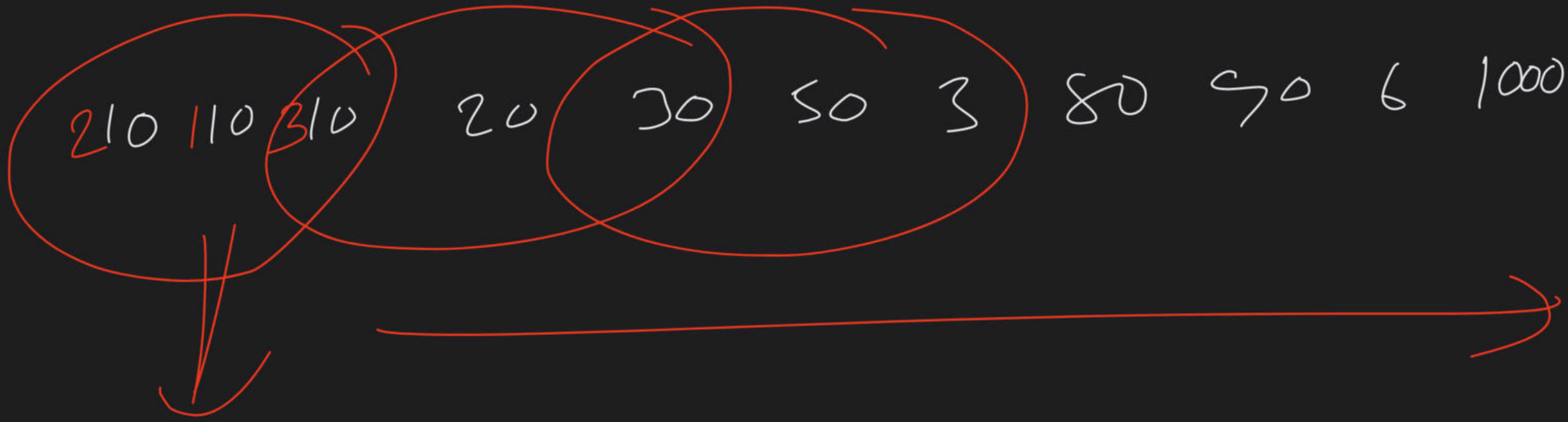
10

80

$\Theta(1)$ [Best Algo, Ec]

not sorted, no-distinct

find neither min nor max



110 210 310

O(n) [Best Algo, WC – n, BC – 1]

$= 2^{\log n - 1} T(1) + c[2^0 + 2^1 + \cdots + 2^{\log n - 1}]$

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

$= \frac{n}{2} \cdot O(1) + c \cdot n$

Leaf node

$= \frac{n}{2} \cdot c + c \cdot n \Longrightarrow \Theta(n)$ [$\because c$]

$2^0 (2^{\log n} - 1)$ / $2 - 1$ $= 2^{\log n}$

(5)

$2T(n/2) + c$

$\log n$

$\Theta(n)$

Space complexity = Stack space = $\Theta(\log n)$.