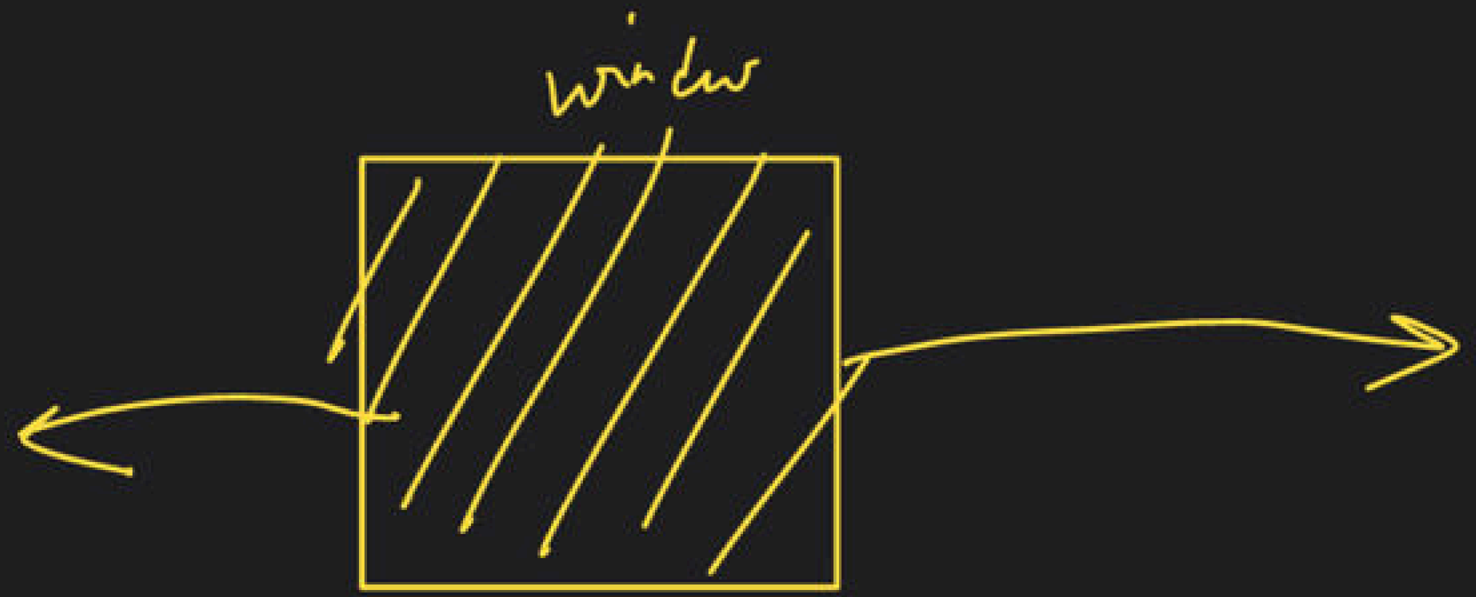
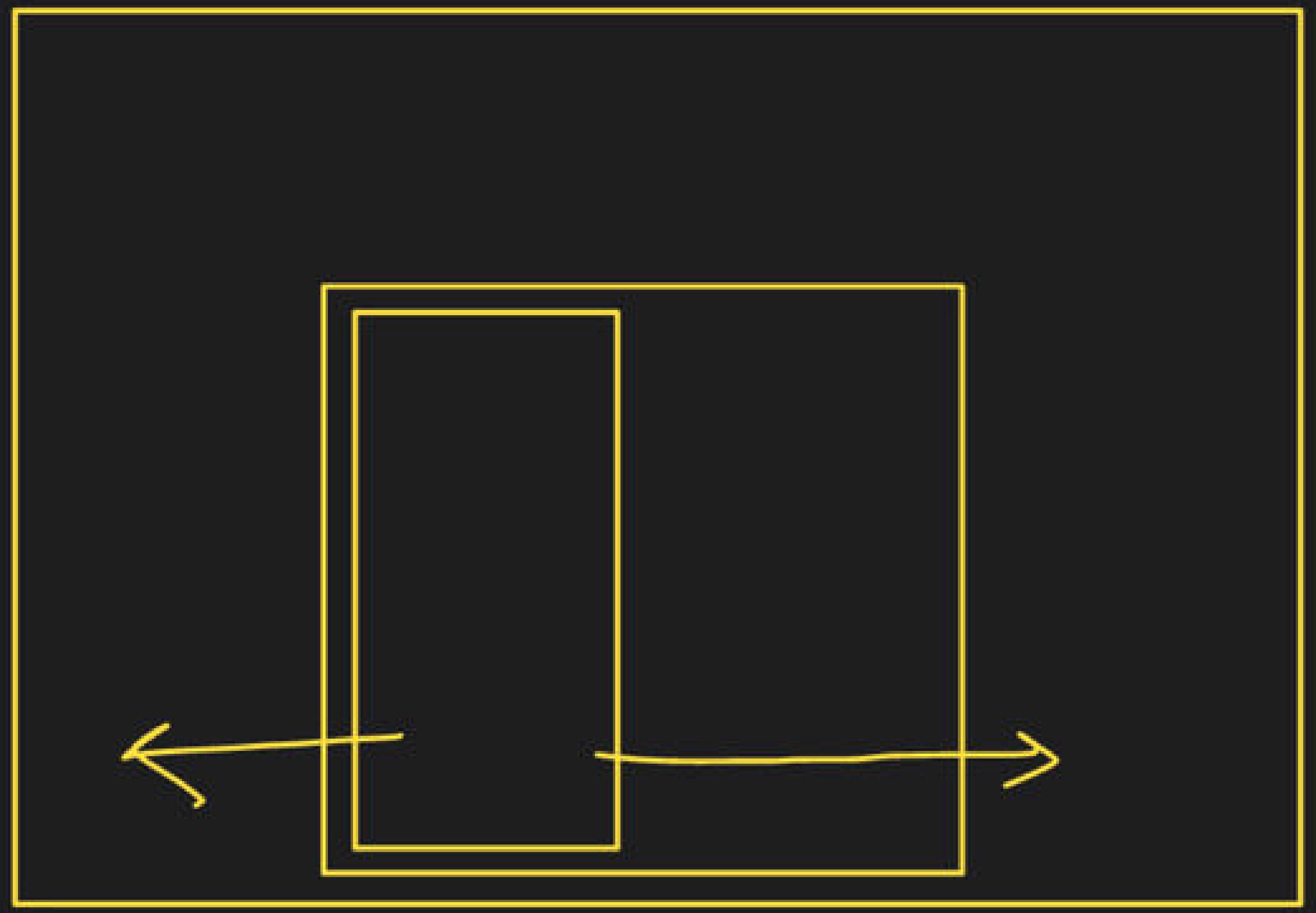
Sliding Window Technique

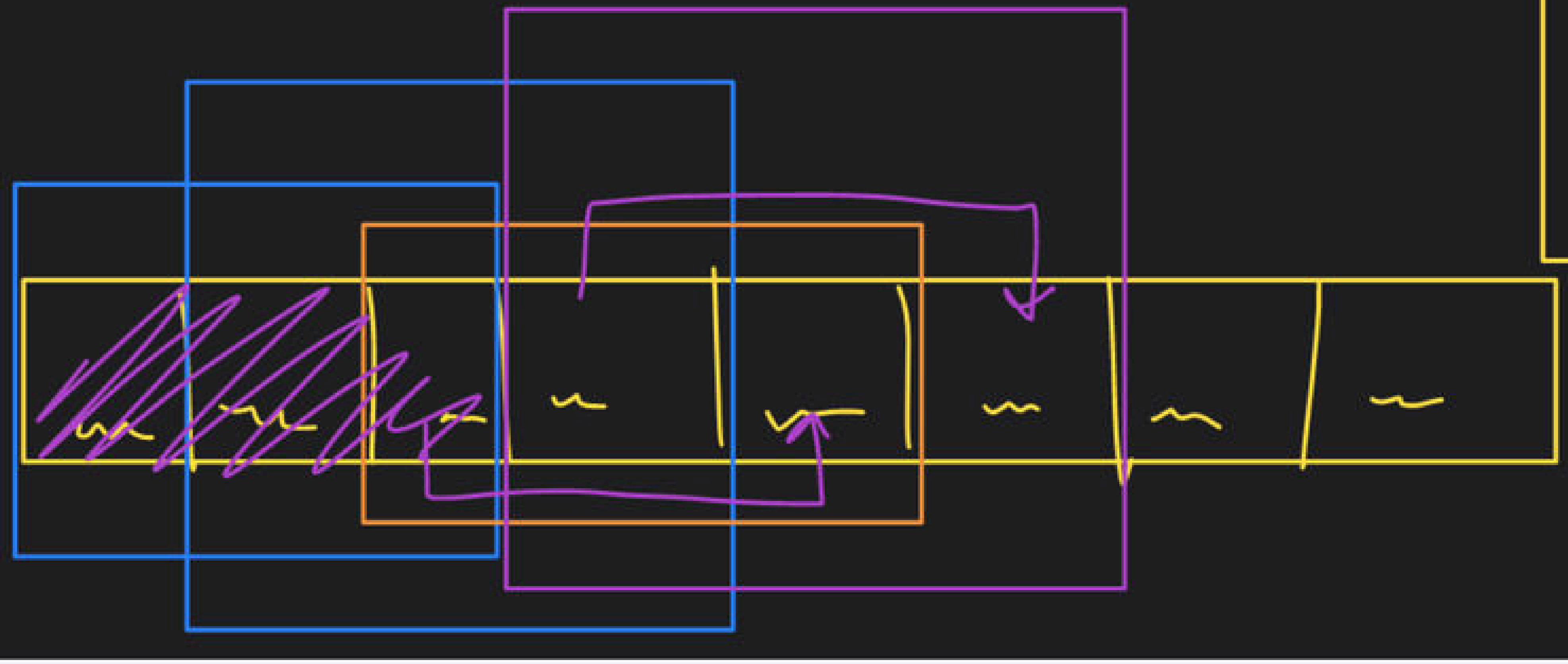
Special class

Love Babbar • Mar 14, 2024

→ Sliding Window







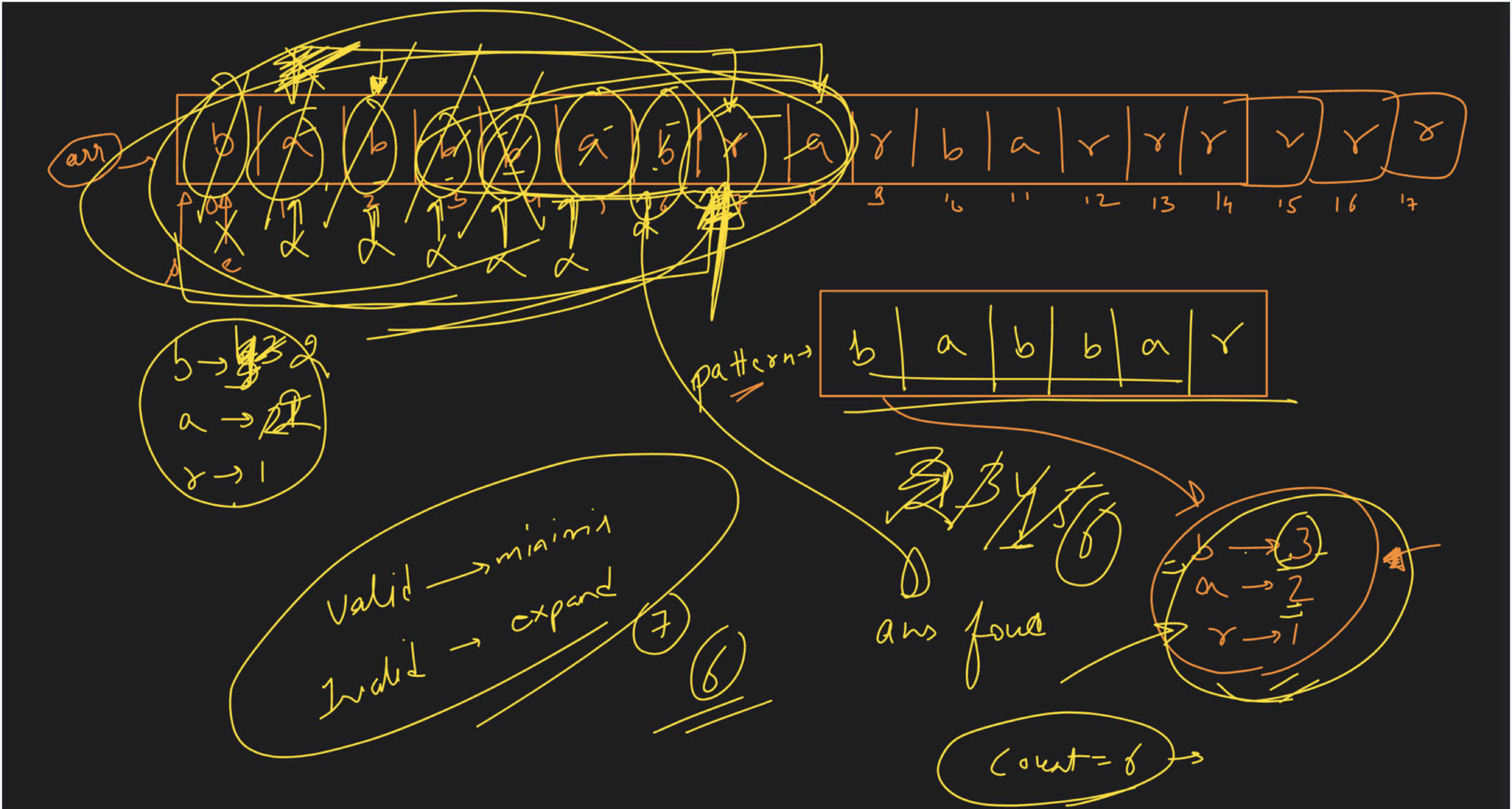
Sliding window

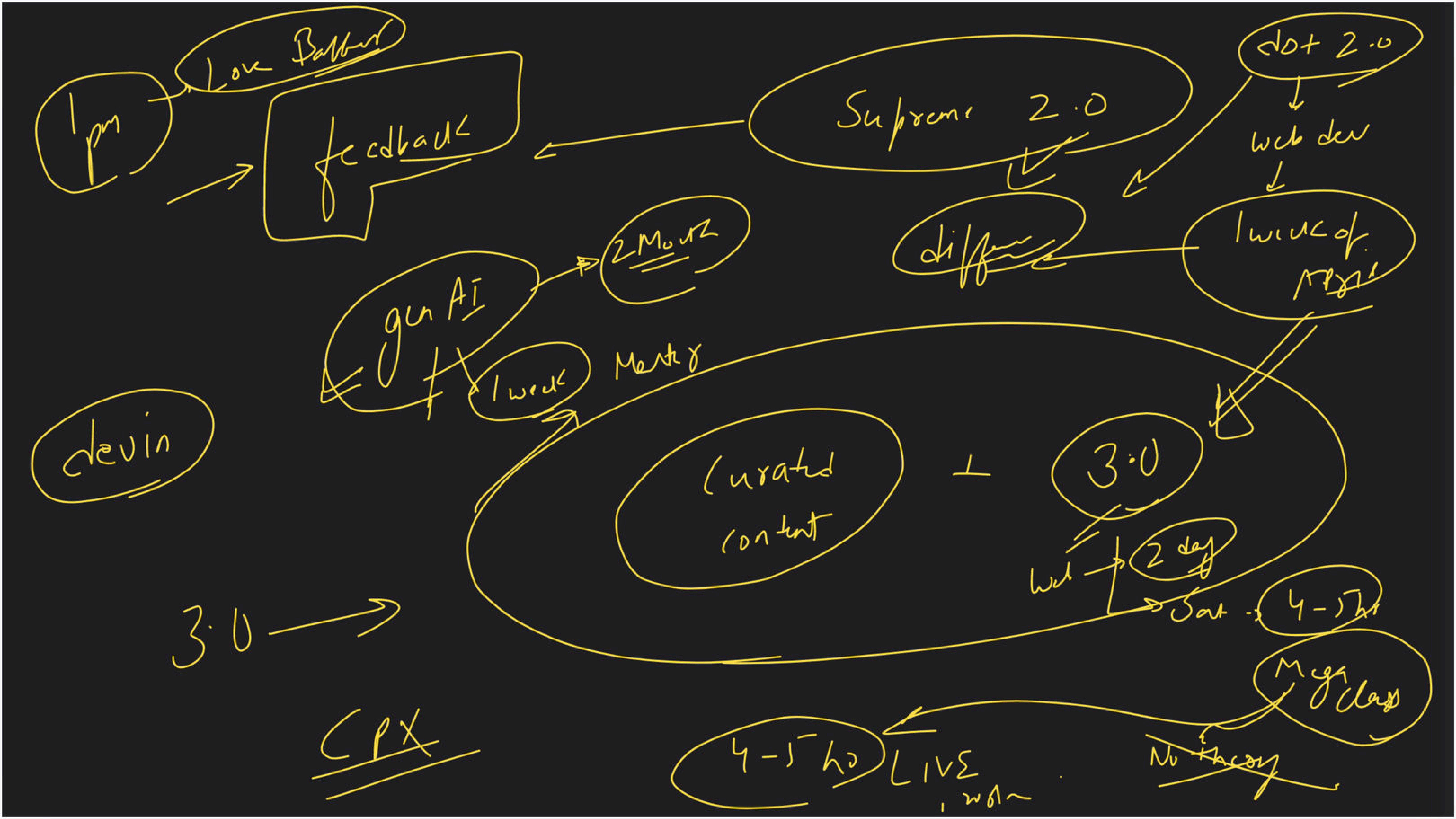
hash/ set/map/ precomputation ⟸ optimisation

- Fixed-size window
- Variable-size window
- 2 pointer approach

2 min

Break

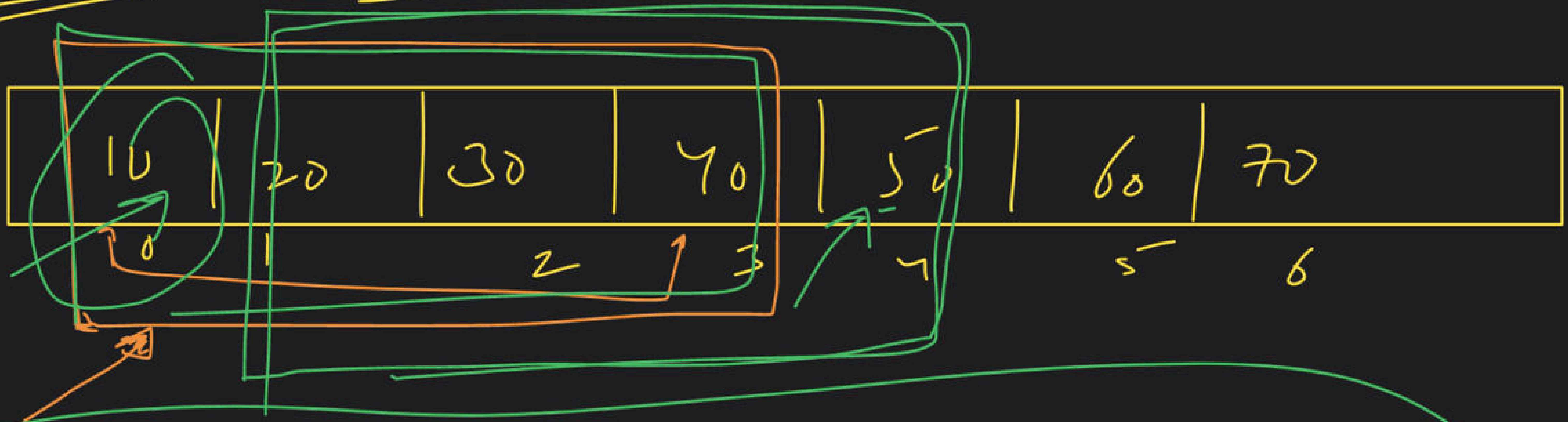

arr
b a b b b a b r a r b a r r r r r r
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
s e
b → 2
a → 2
r → 1
pattern
b a b b a r
b → 3
a → 2
r → 1
Valid → minimi
Invalid → expand
7
6
6
ans found
count = r


1 pm
Love Babbar
feedback
Supreme 2.0
dot 2.0
web dev
1 week of Apti
differ
2 Month
gen AI
1 week
Mentor
devin
Curated content
+
3.0
wk
2 day
Sat
4-5 hr
Mega class
No ~~theory~~
4-5 hr
LIVE
1 wdn
3.0
CPX

① Fixed-size window:

size = 4

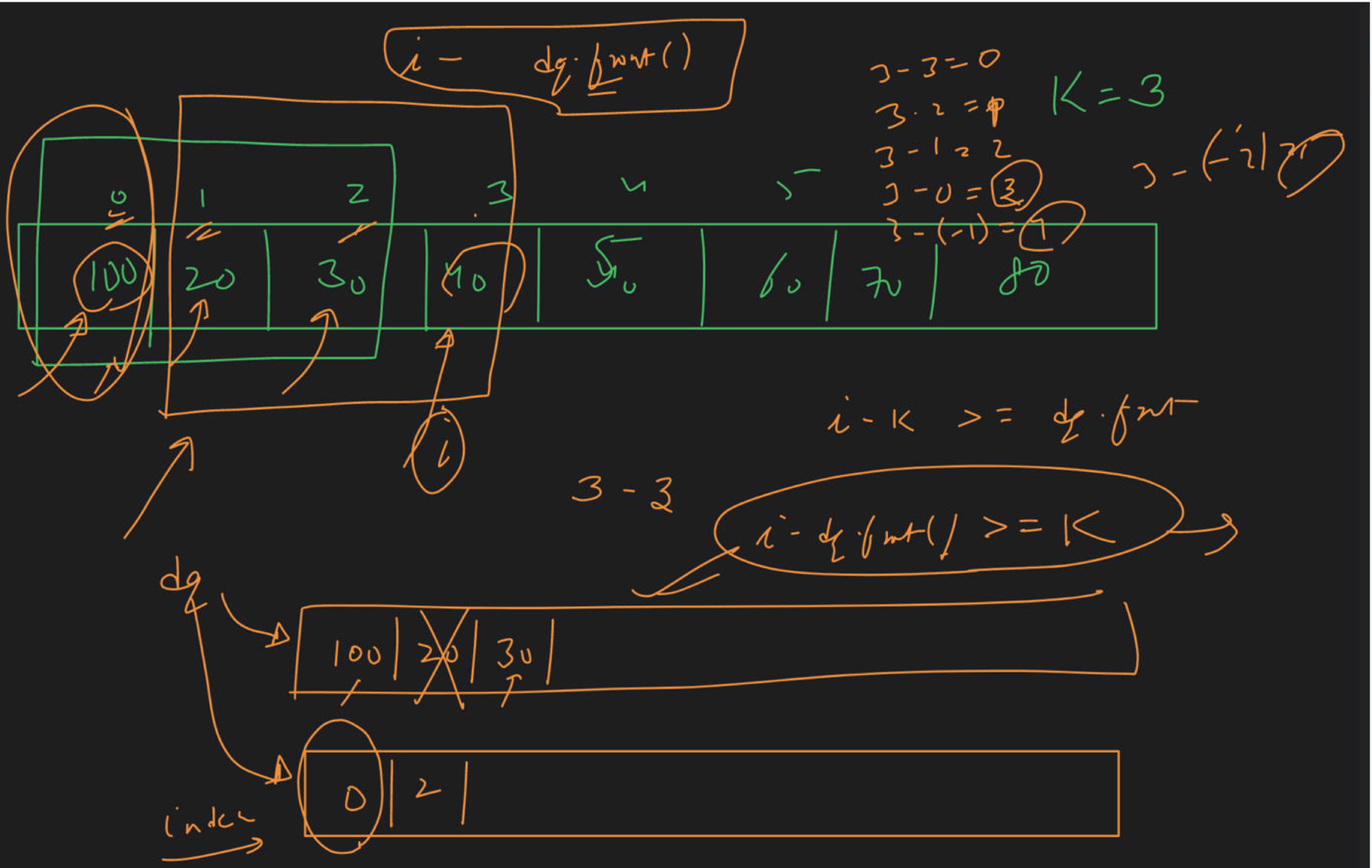
K size



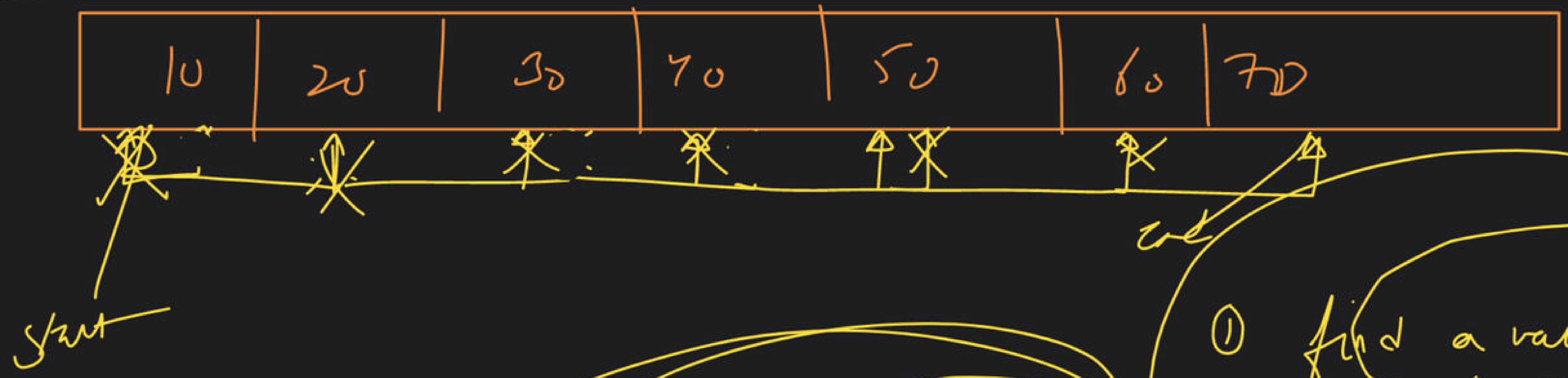
10	20	30	40	50	60	70
0	1	2	3	4	5	6

① process first "K" elements → initial state

② process remaining windows → removal → addition → ans store


i - dq.front()
3 - 3 = 0
3 - 2 = 1
3 - 1 = 2
3 - 0 = 3
3 - (-1) = 4
K = 3
3 - (-2) = 5
0
1
2
3
4
5
100
20
30
40
50
60
70
80
i
i - K >= dq.front
3 - 3
i - dq.front() >= K
dq
100 | 20 | 30
0 | 2
index

Variable size :-



① find a valid ans

true

minimise

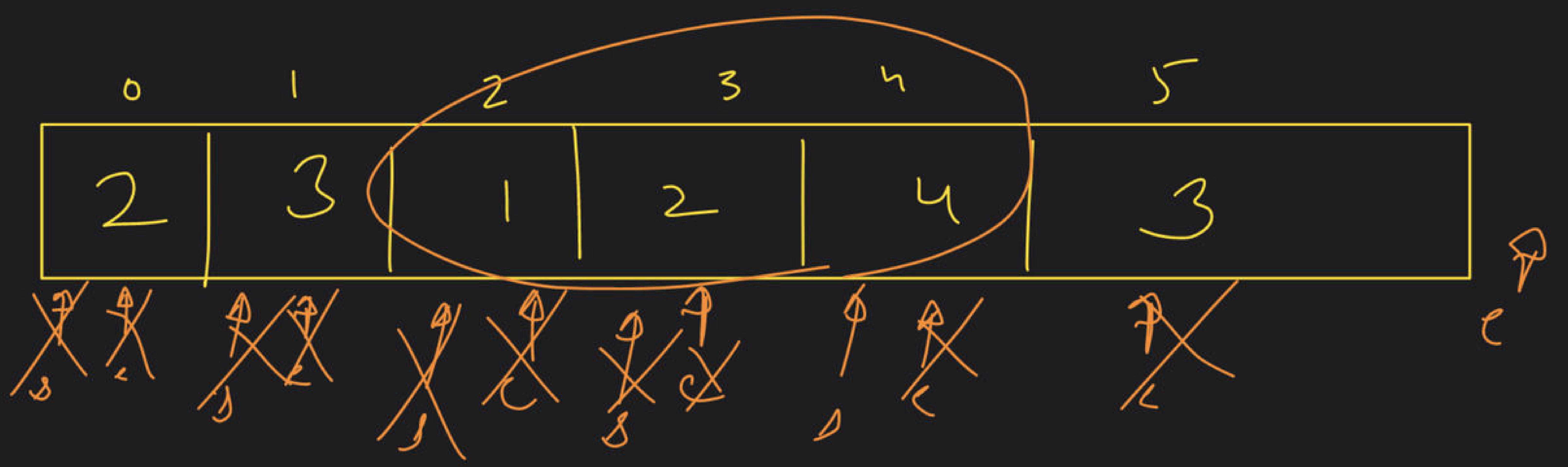
start++

When no valid ans → explore end++

Valid ans → minimise
↳ start++.

Invalid ans → explore → end++

T = 7



2 < 7

5 < 7

6 < 7

8 > 7

6 < 7

10 > 7

7 == 7

6 < 7

9 > 7

7 == 7

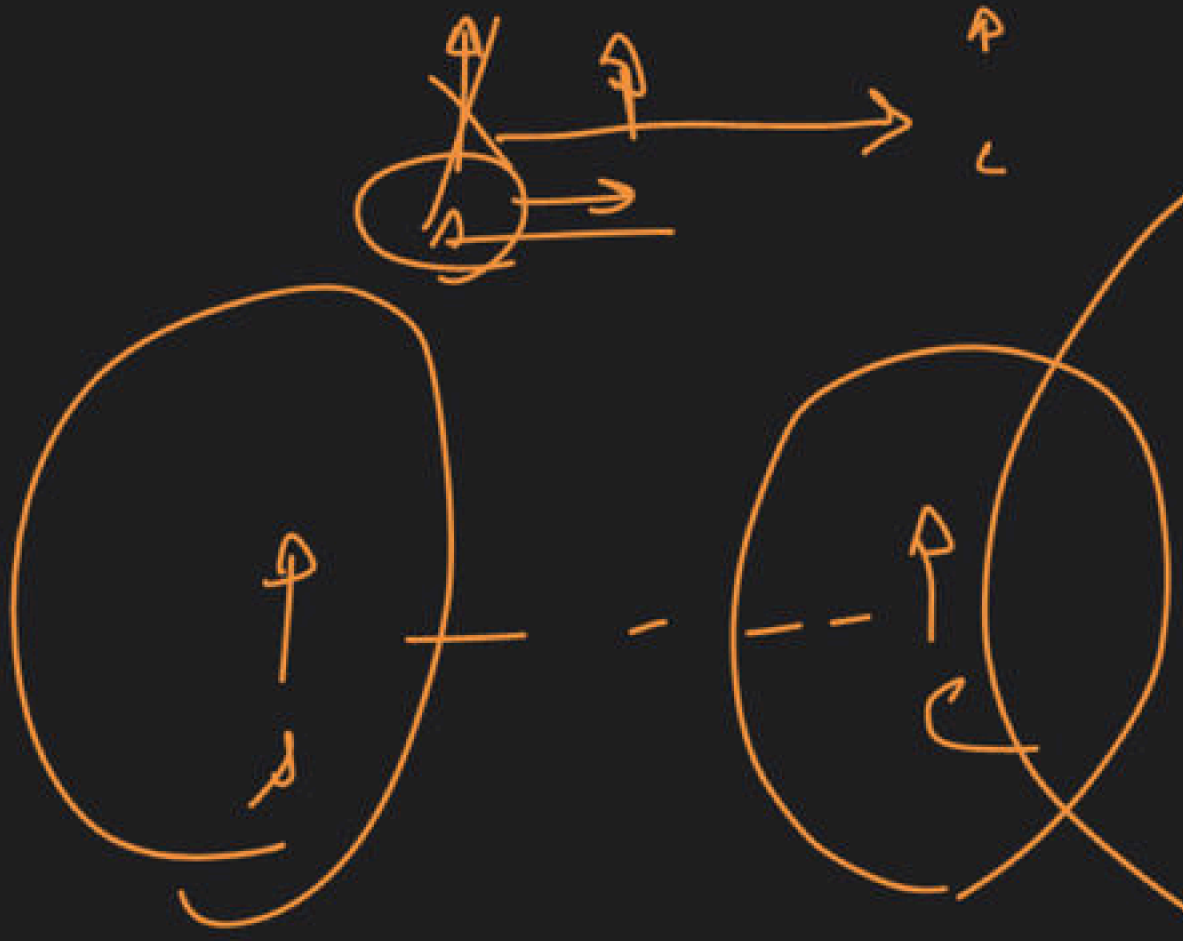
1, 2, 4 → 3

4, 3 → 2

min

2

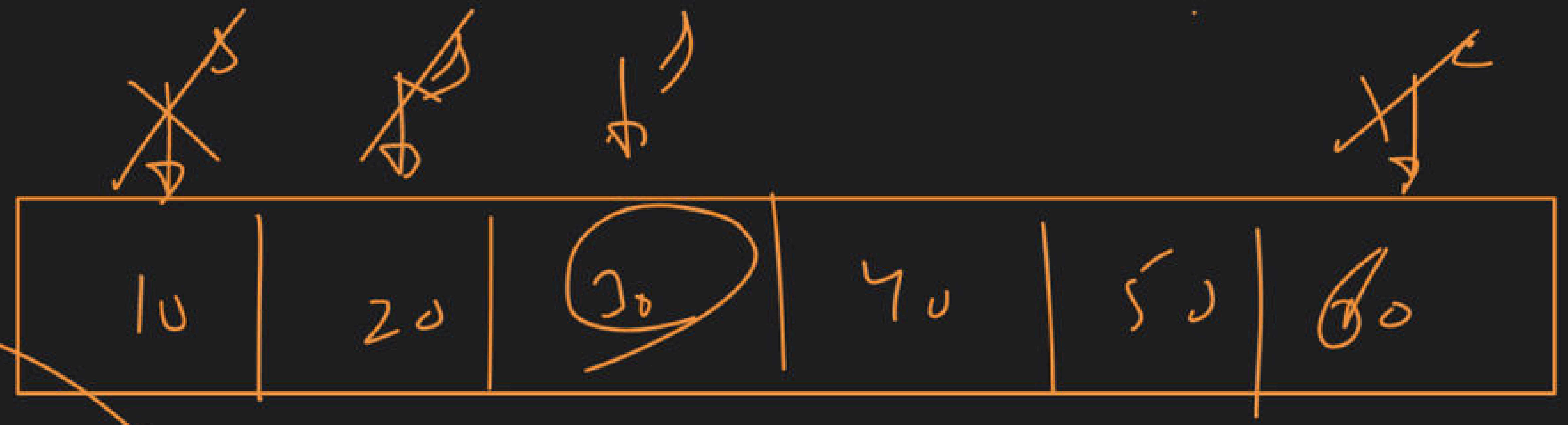
Variable size :-



Valid sol^n
↳ minimise → s++;

Invalid sol^n
↳ expand → e++

→ 2 ptr



sum < Target → s++

sum > Target → e--

s>e

↳ Ruk

x > 90 → e--

Sort

10 + 60 → 70 → != 90

70 < 90 → s++

20 + 60 → != 90

80 < 90 → s++

30 + 60 → 90

2 ptr

cond^n → ruk jao

cond^n → s ?

cond^n → e ?

2 no

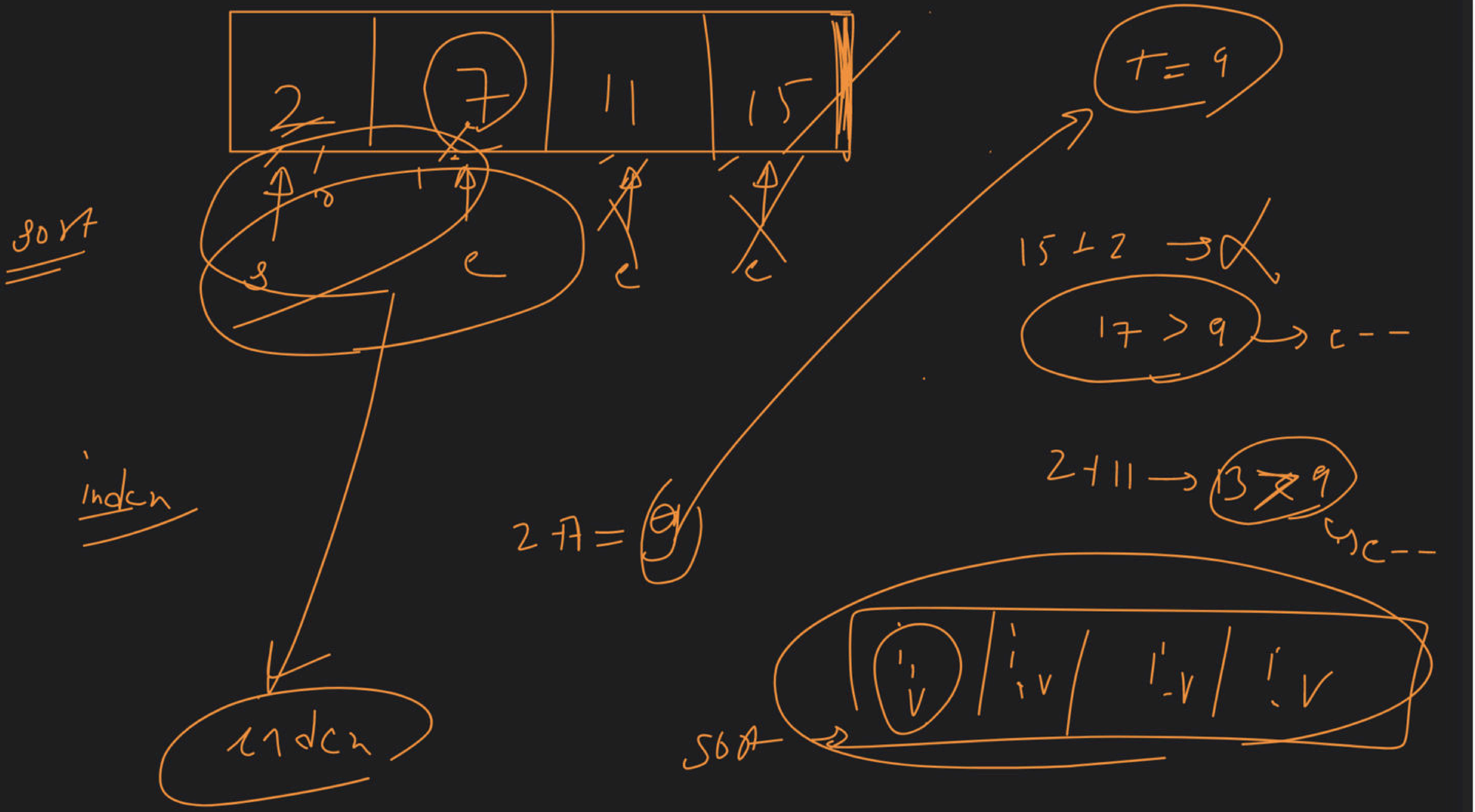
↓

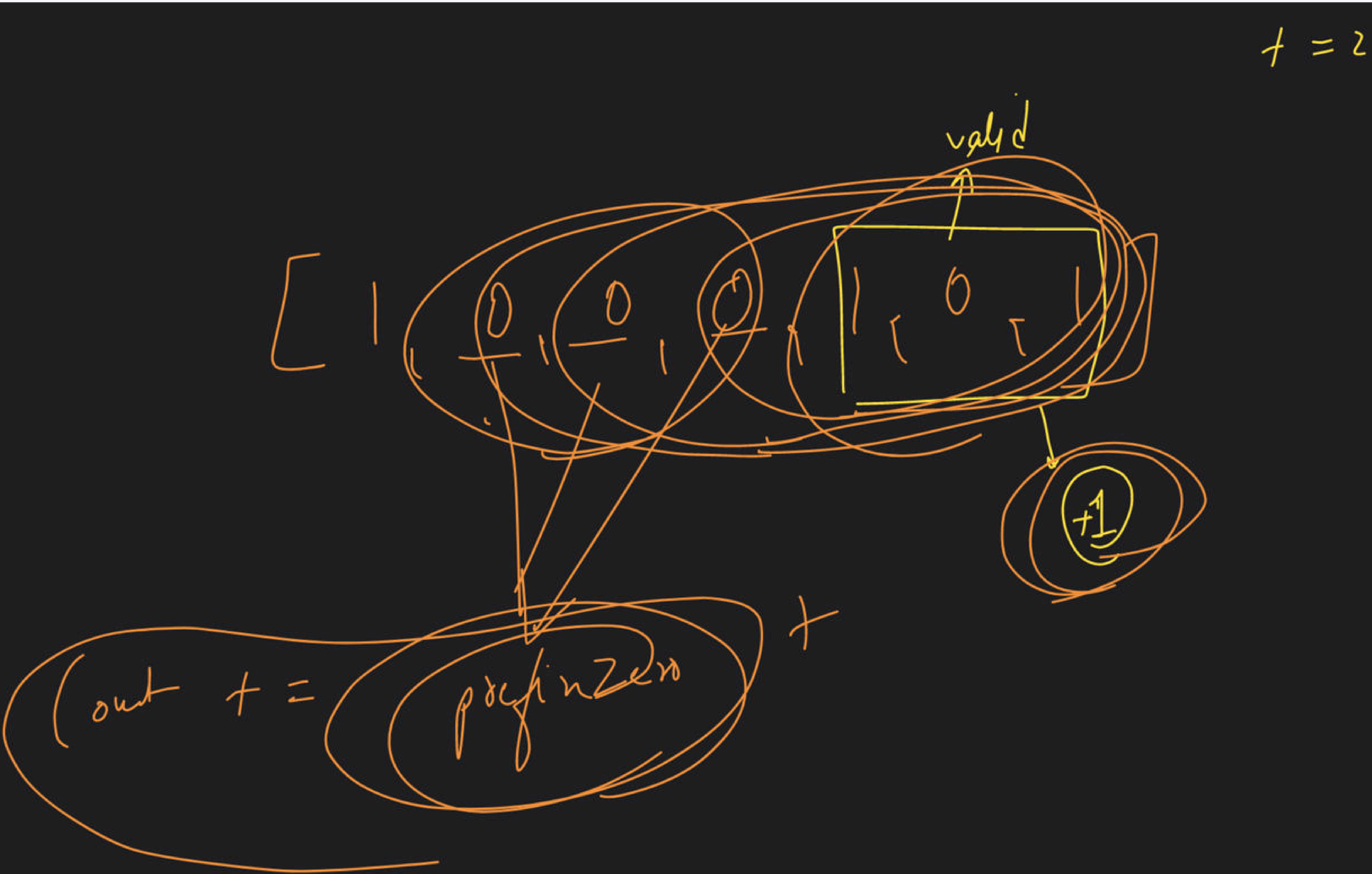
sum

↓

s

90


2
7
11
15
0
1
s
e
e
e
sort
T = 9
15 + 2 → ✗
17 > 9 → e--
2 + 11 → 13 > 9
→ e--
indcn
2 + 7 = 9
indcn
500


k = 2
valid
[1, 0, 0, 0, 1, 0, 1
+1
Count += prefixZero +