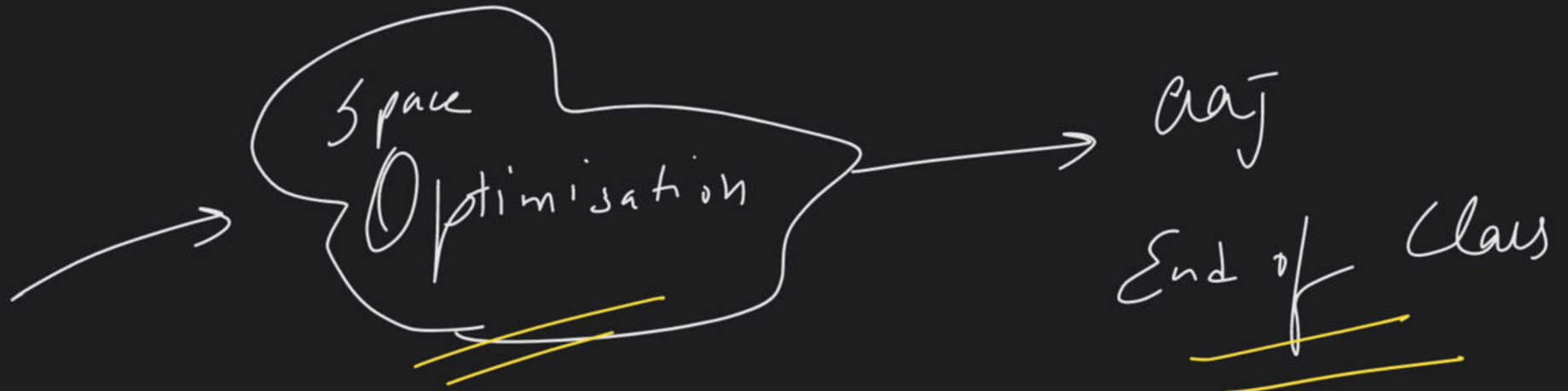
Dynamic Programming Class-5

Special class

Love Babbar • Jan 15, 2024



aaj

End of class

→ Longest Increasing Subsequence

[ ] → [10, 9, 2, 5, 3, 7, 101, 18]

i → subproblem

[10, 101]

[9, 101]

[2, 5, 7, 101]

[3, 7, 101]

[7, 101]

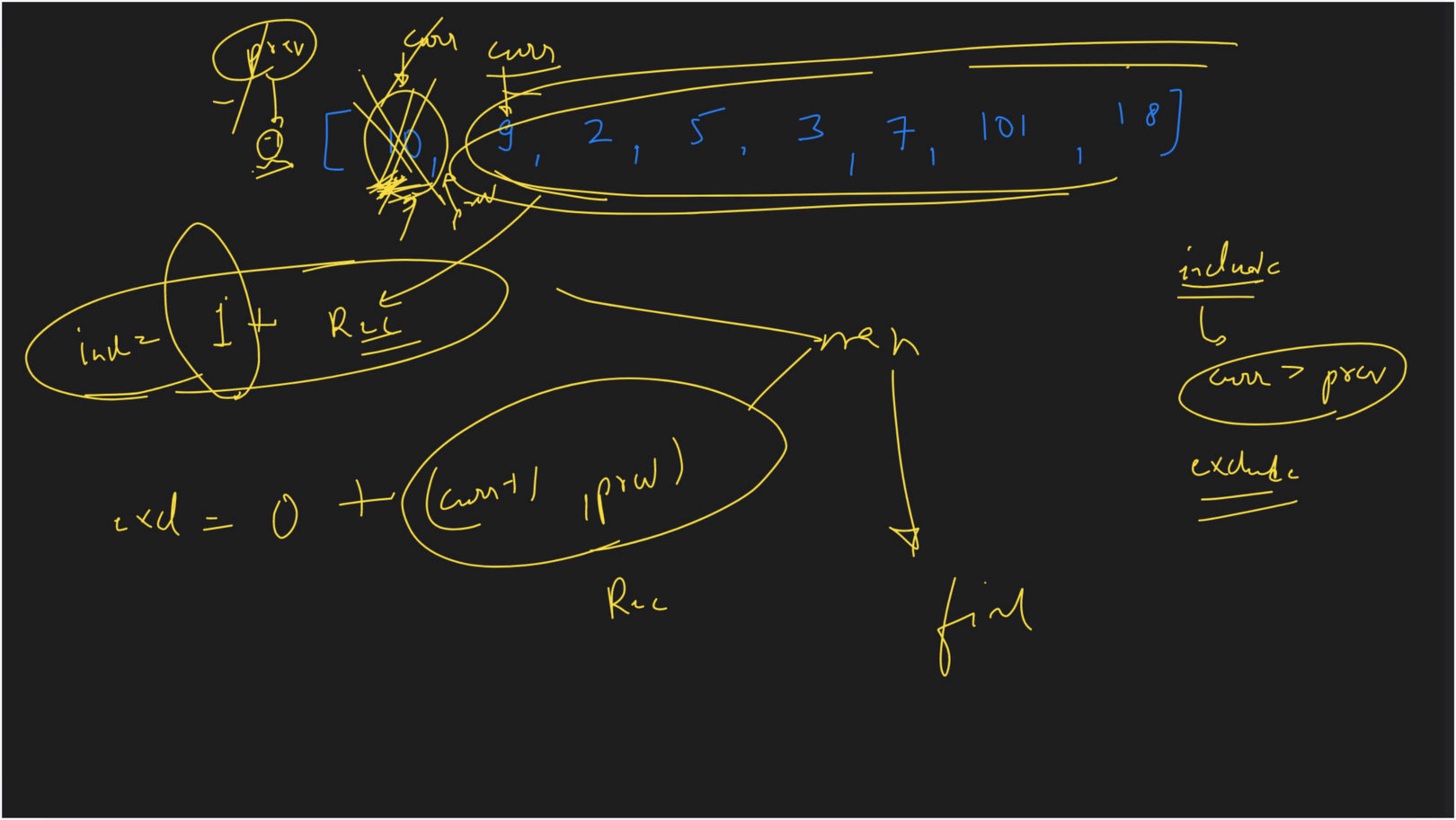
[101] [18]

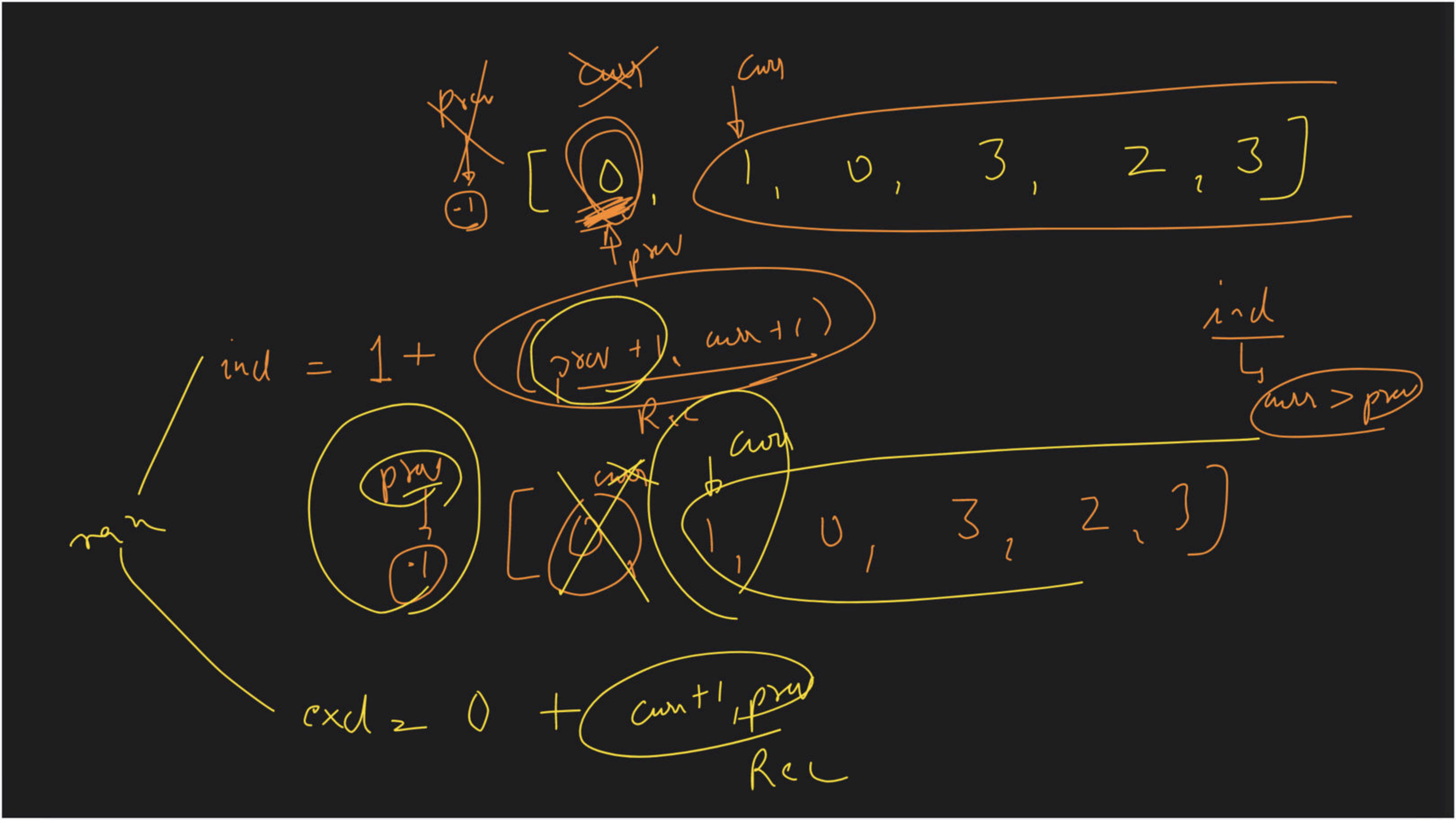
4 → subproblem

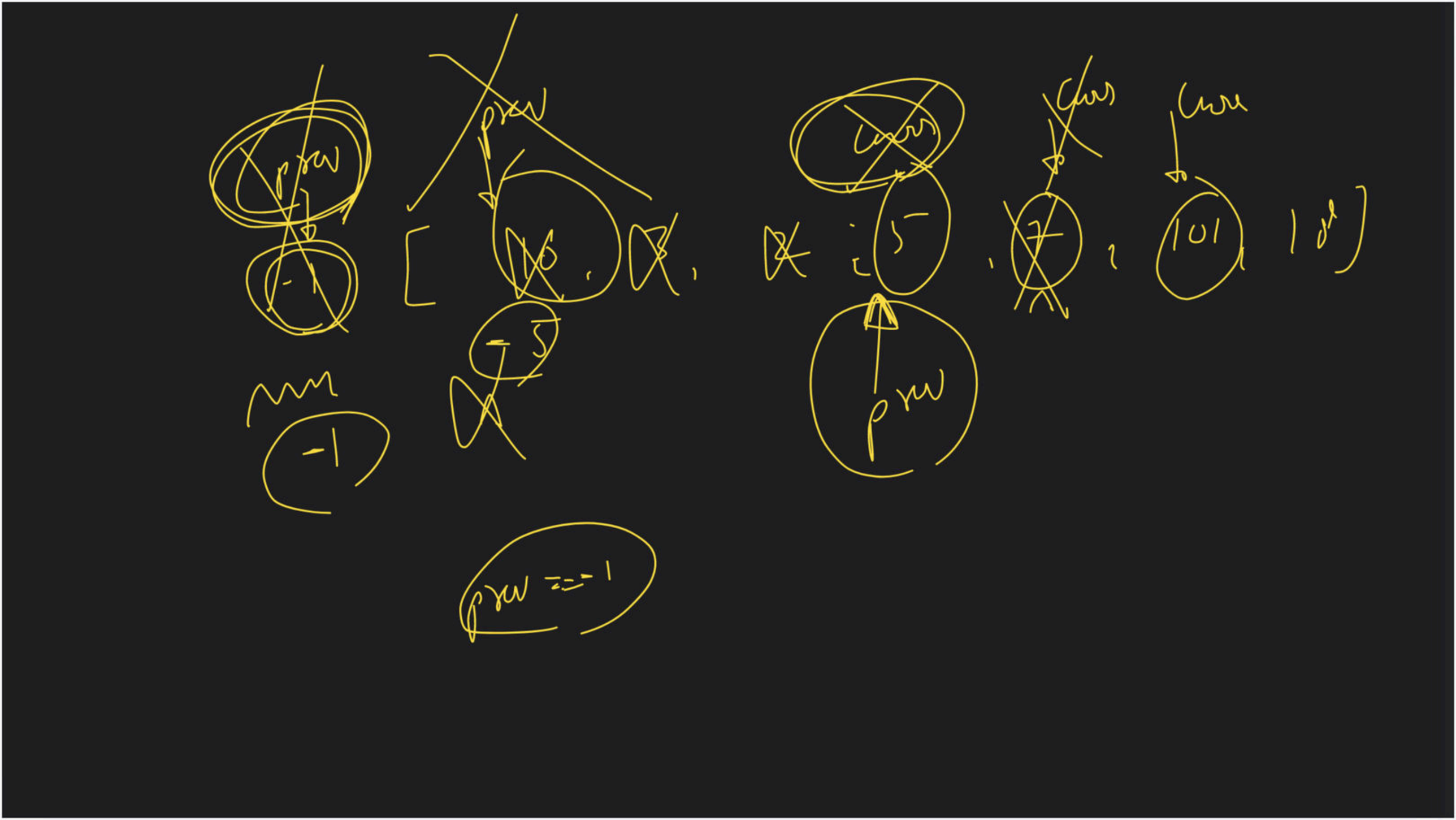
include exclude

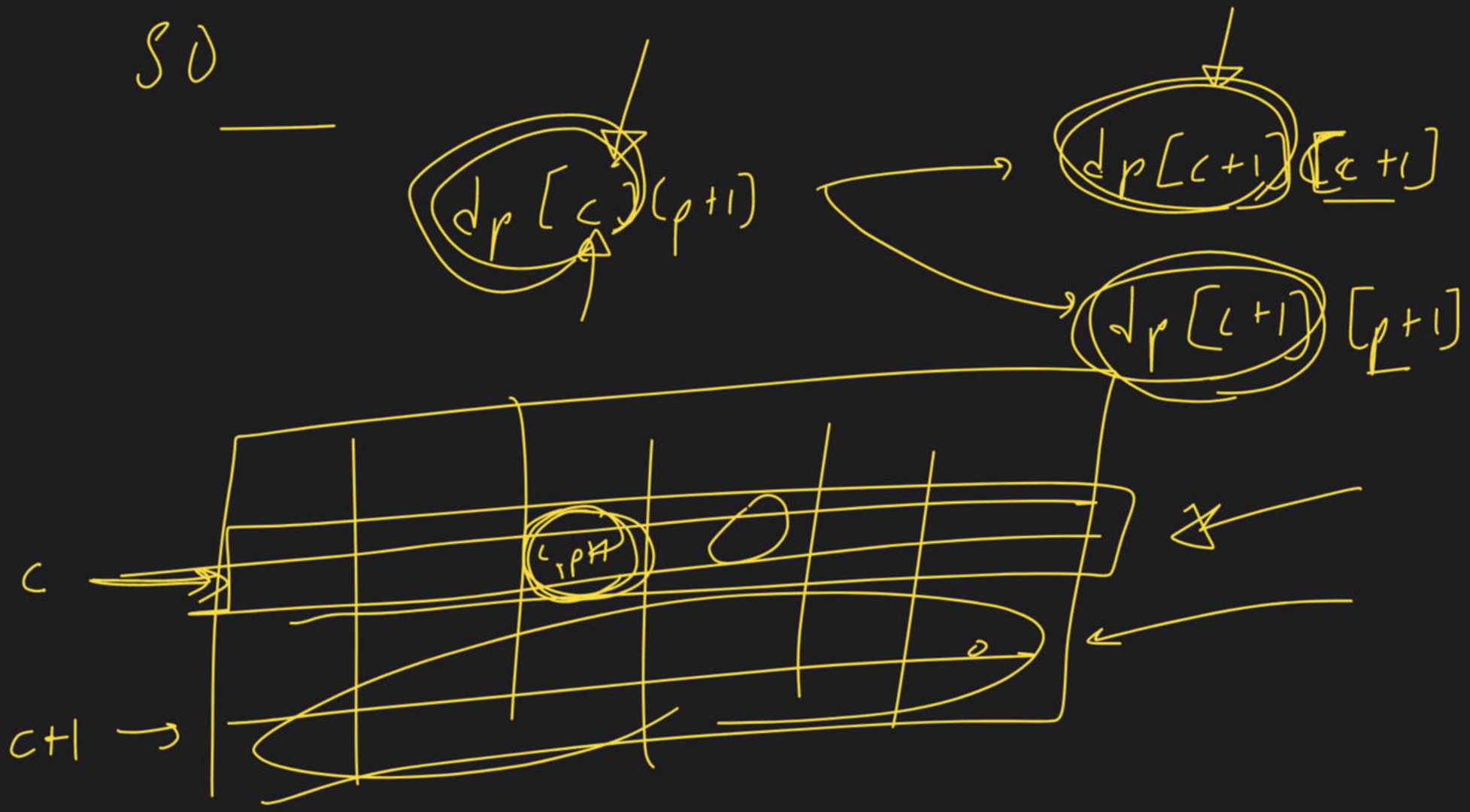
max

final


prev
curr
curr
[10, 9, 2, 5, 3, 7, 101, 18]
incl = 1 + Rec
excl = 0 + (curr, prev)
Rec
max
final
include
curr > prev
exclude

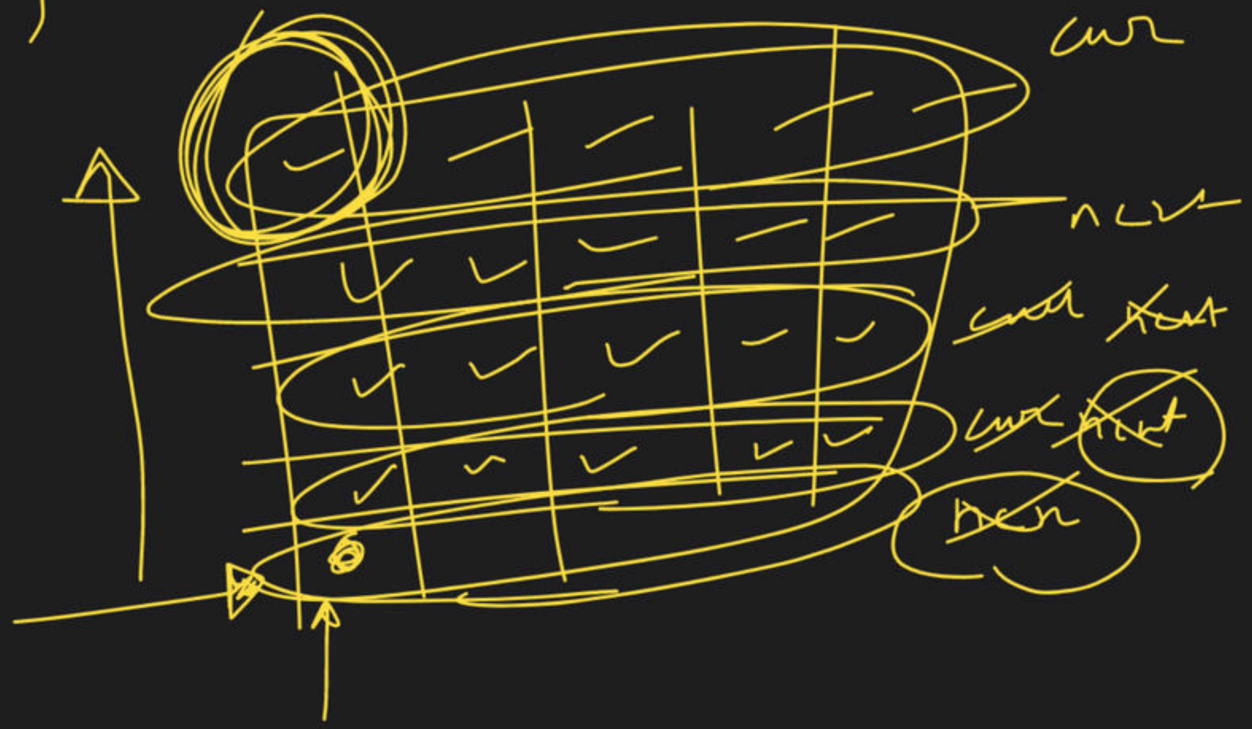

prev
curr
cur
-1
0
1, 0, 3, 2, 3]
prev
ind = 1 +
prev + 1, cur + 1
Rec
ind
cur > prev
prev
-1
curr
cur
1, 0, 3, 2, 3]
ans
excl = 0 +
cur + 1, prev
Rec

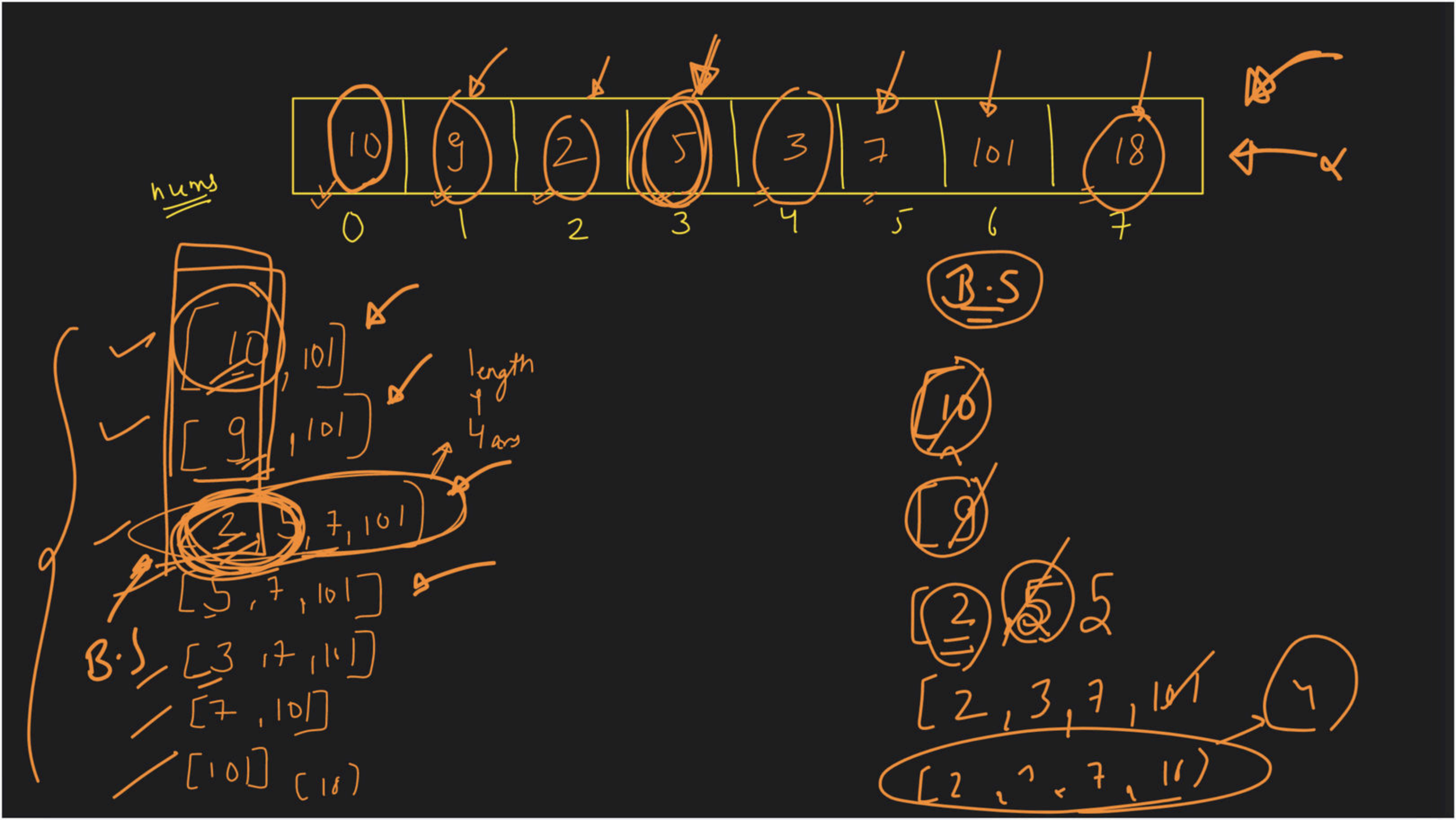

pw
-1
pw
-5
pw
pw
7
101

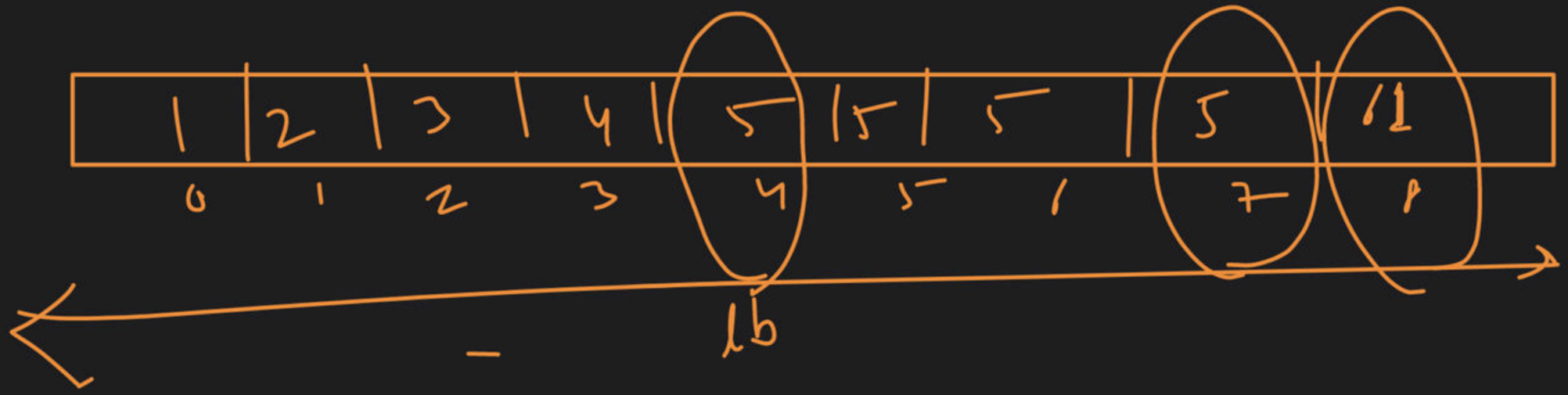

SO
dp[c][p+1]
dp[c+1][c+1]
dp[c+1][p+1]
c
c+1

for (n - 1 ⟶ 0)

(curr - 1 → . 1)



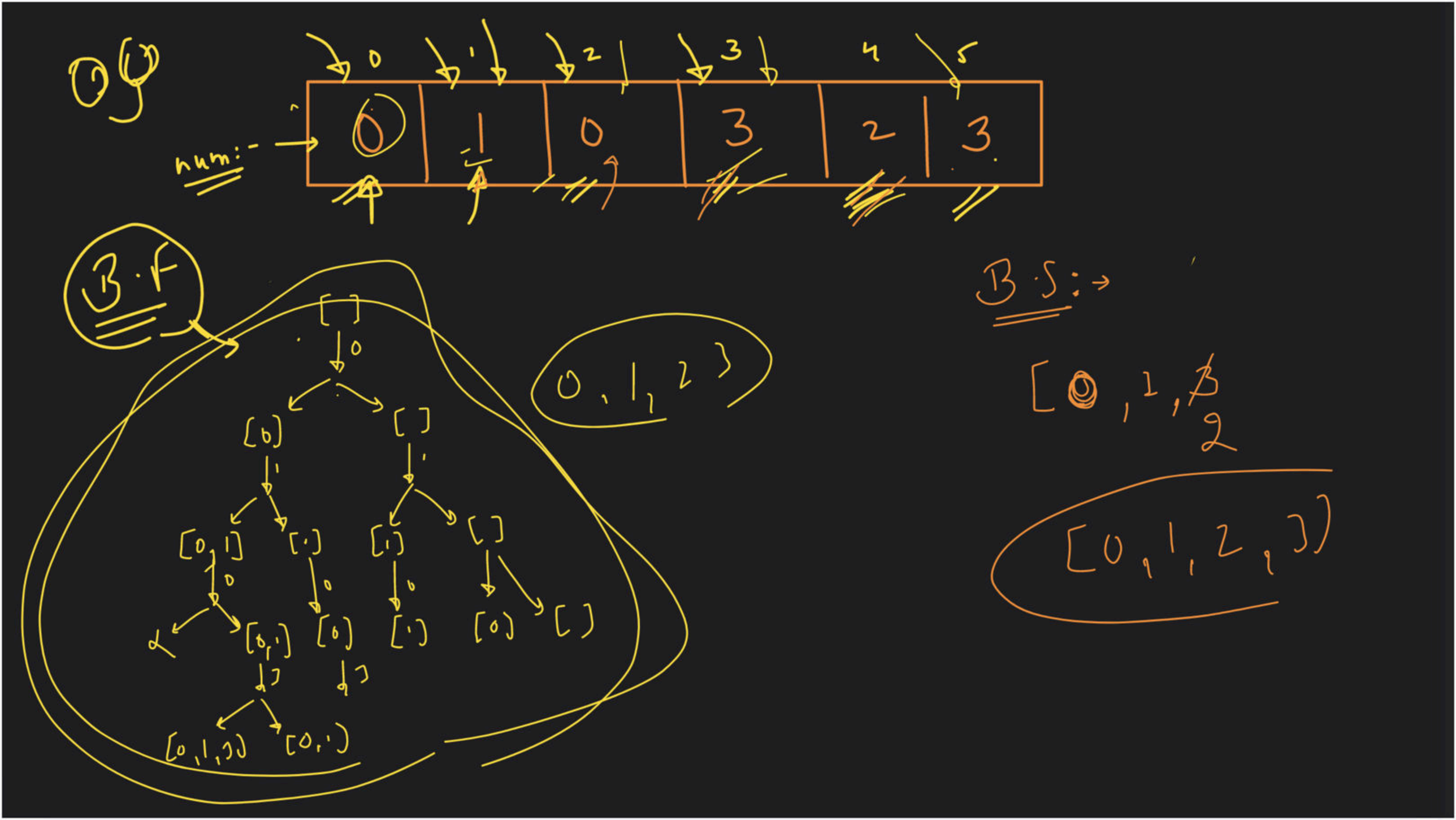

nums
10
9
2
5
3
7
101
18
0
1
2
3
4
5
6
7
length
4
B.S
B.S

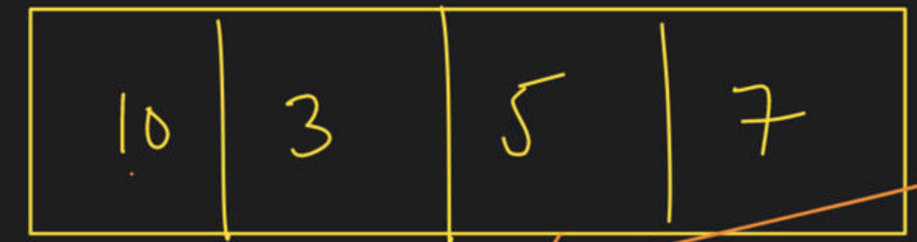


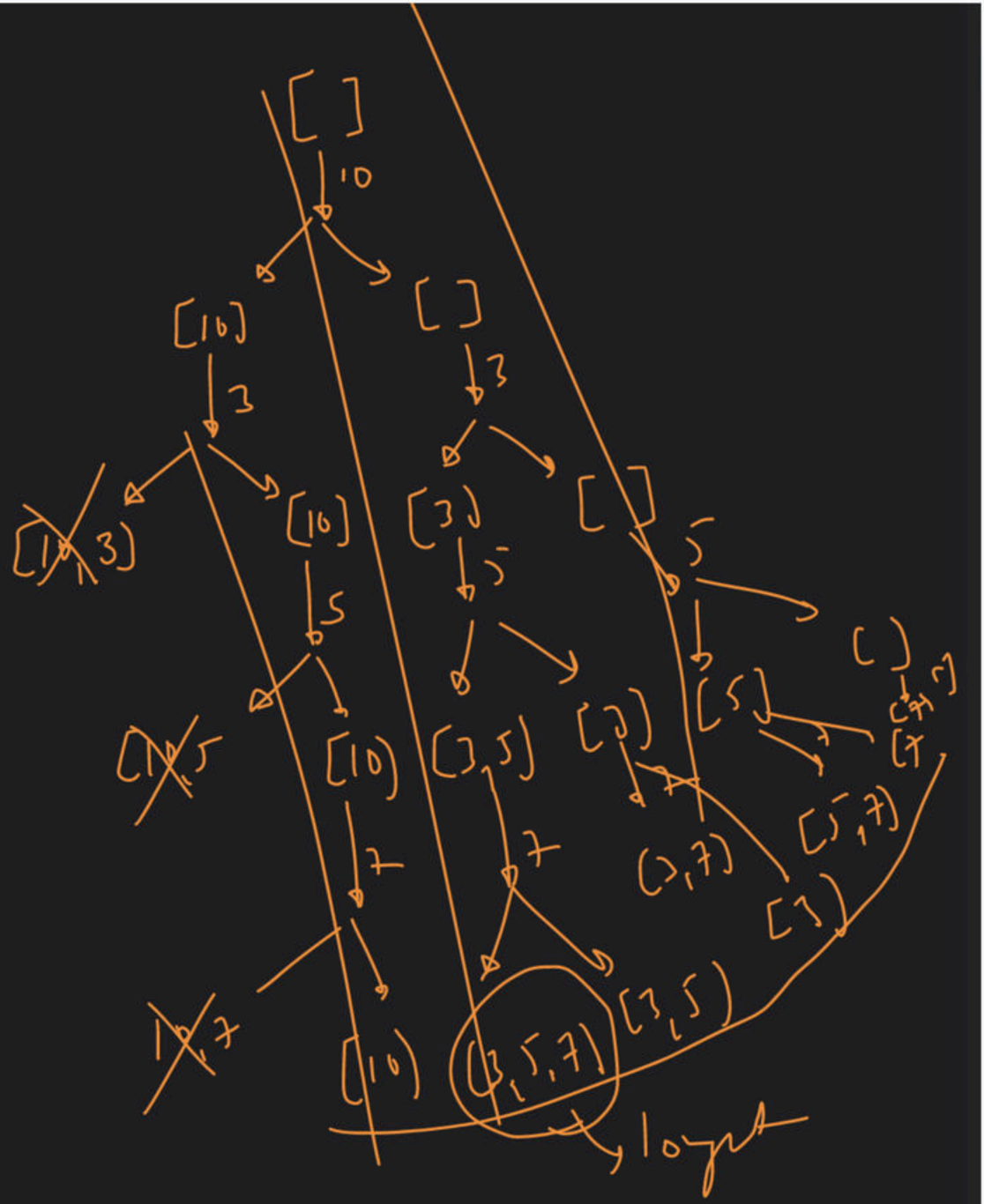
lower bound :

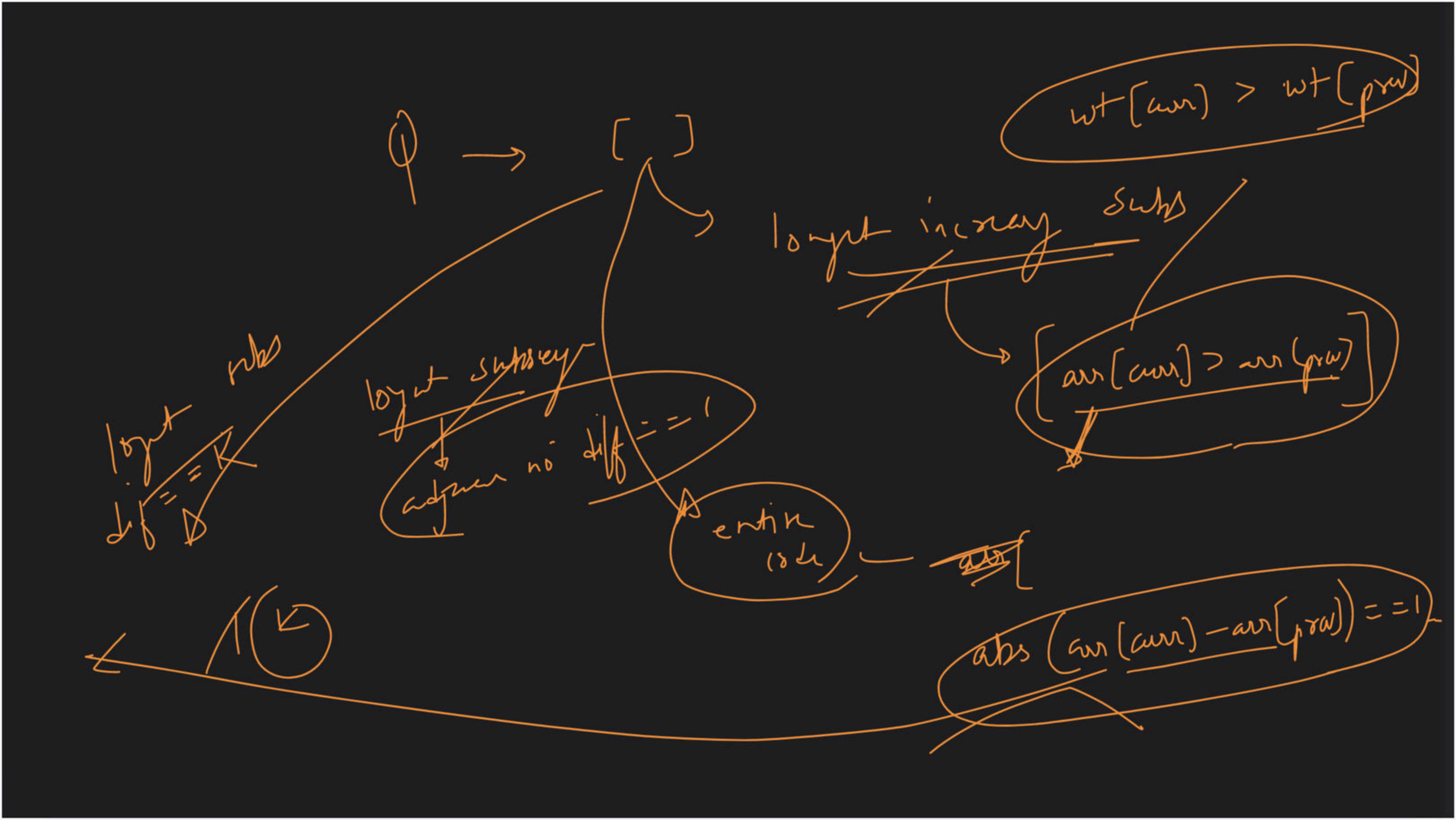
upper bound

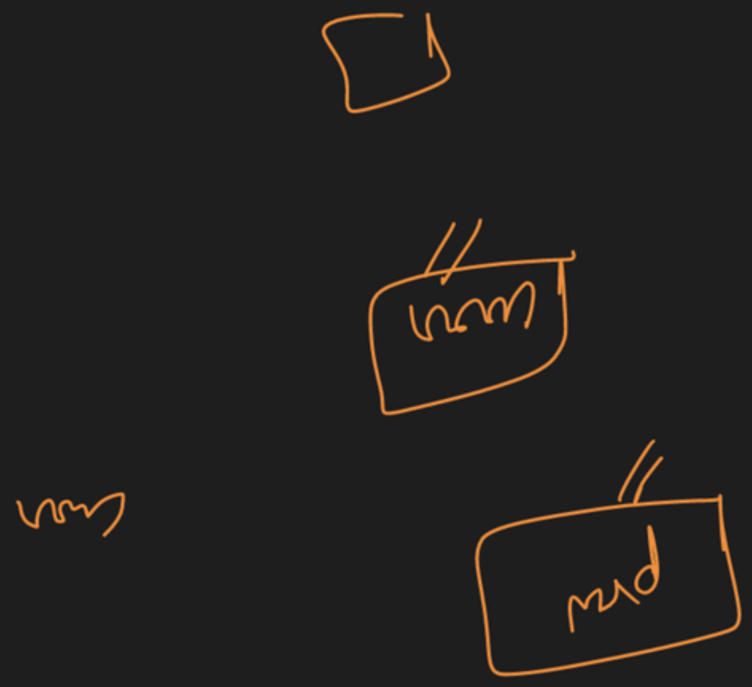
lower Bound : 10


0
1
2
3
4
5
num:-
0
1
0
3
2
3
3:f
[]
0
[0]
[]
[0,1]
[0,2]
[1]
[2]
[0,1,2]
[0,2]
[0]
[1]
[0]
[]
[0,1,2]
[0,2]
0,1,2
B.S:→
[0,1,2
2
[0,1,2,]

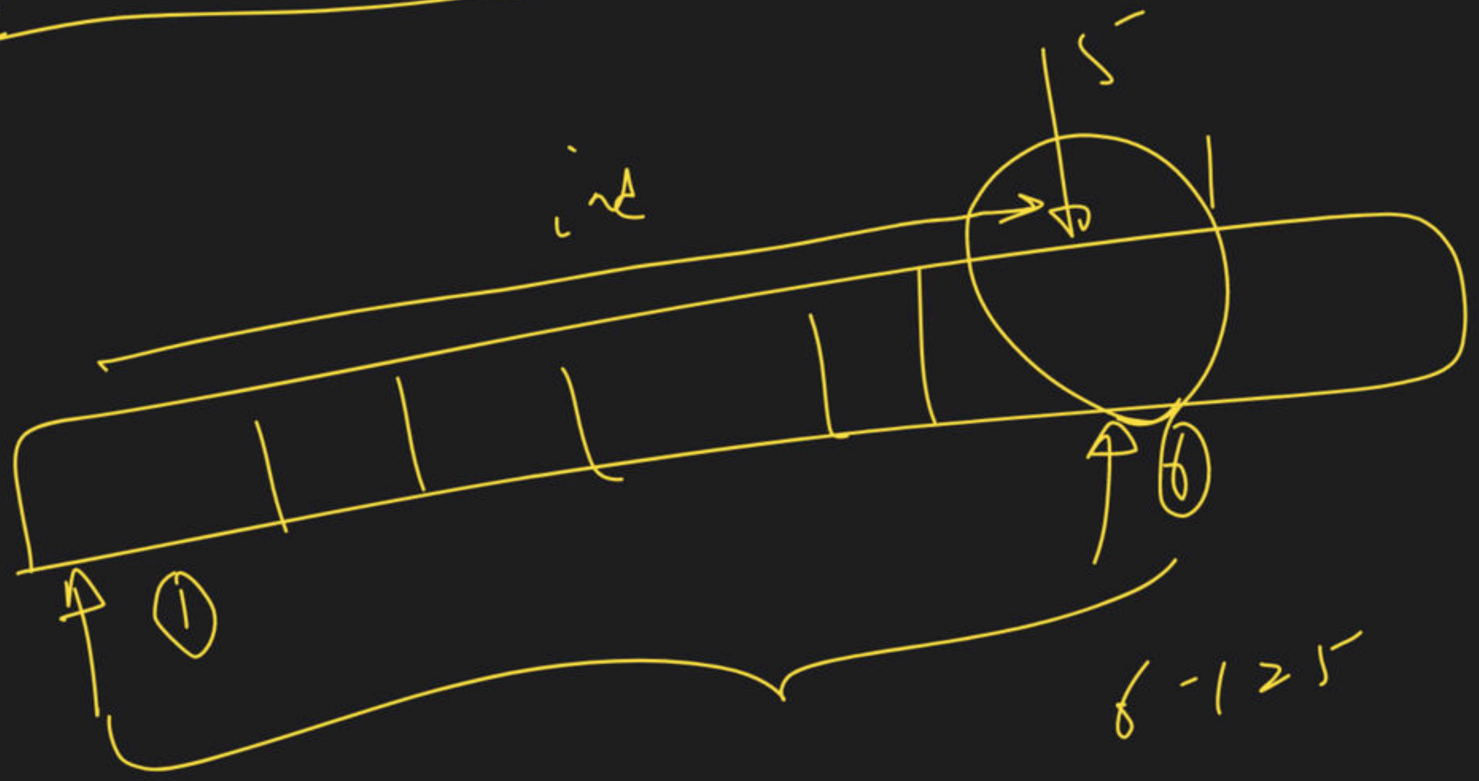

10
3
5
7

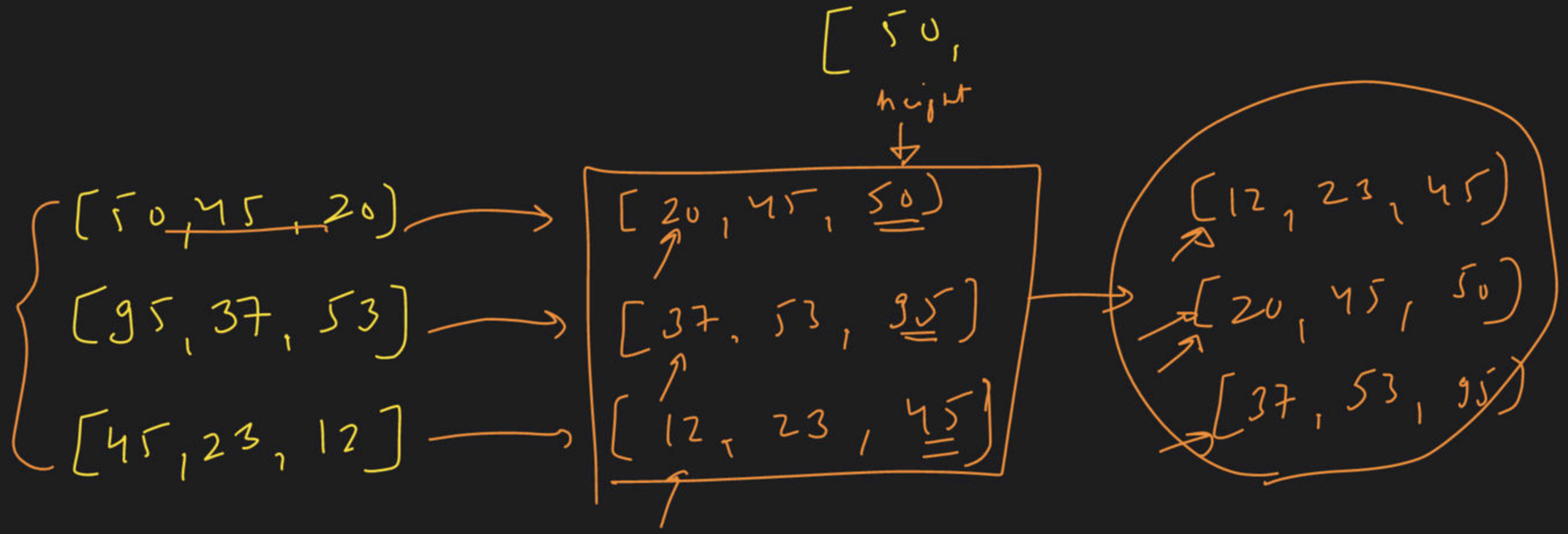

[ ]
10
[10]
3
[ ]
3
[1 3]
[10]
[3]
[ ]
5
5
[1 5]
[10]
[3 5]
[5]
[ ]
7
[10]
[3 5 7]
[3]
[7]
log n

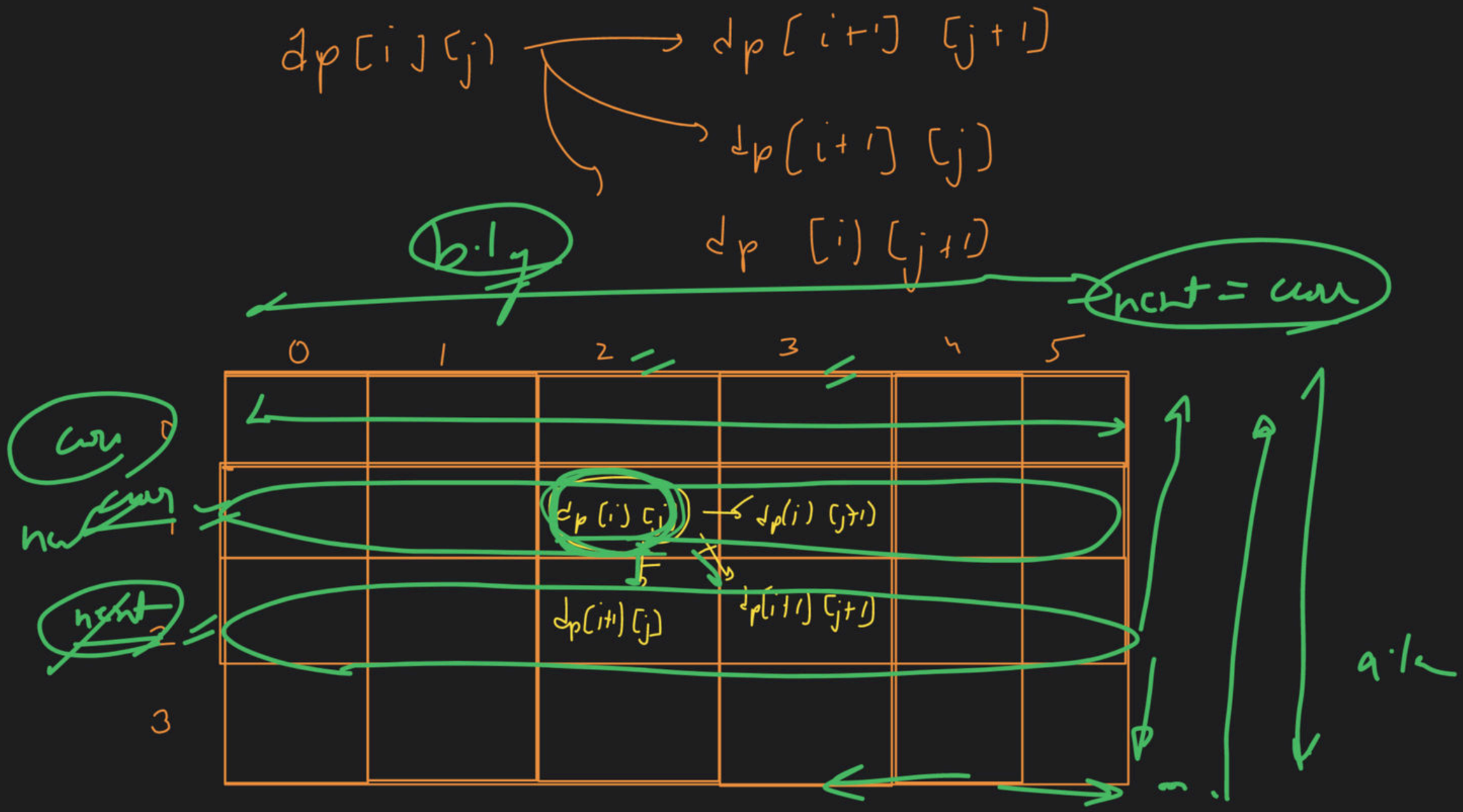

wt[curr] > wt[prev]
longest increasing subseq
arr[curr] > arr[prev]
longest subseq
adjacent no diff == 1
entire idx
abs(arr[curr] - arr[prev]) == 1
longest path
diff == K
K



lower_bound (arr.begin(), arr.end(), num[i]) - arr.begin();




[50, 45, 20]
[95, 37, 53]
[45, 23, 12]
[50,
height
[20, 45, 50]
[37, 53, 95]
[12, 23, 45]
[12, 23, 45]
[20, 45, 50]
[37, 53, 95]


dp[i][j]
dp[i+1][j+1]
dp[i+1][j]
dp[i][j+1]
0
1
2
3
4
5
3
dp[i][j]
dp(i)(j+1)
dp(i+1)[j]
dp(i+1)[j+1]