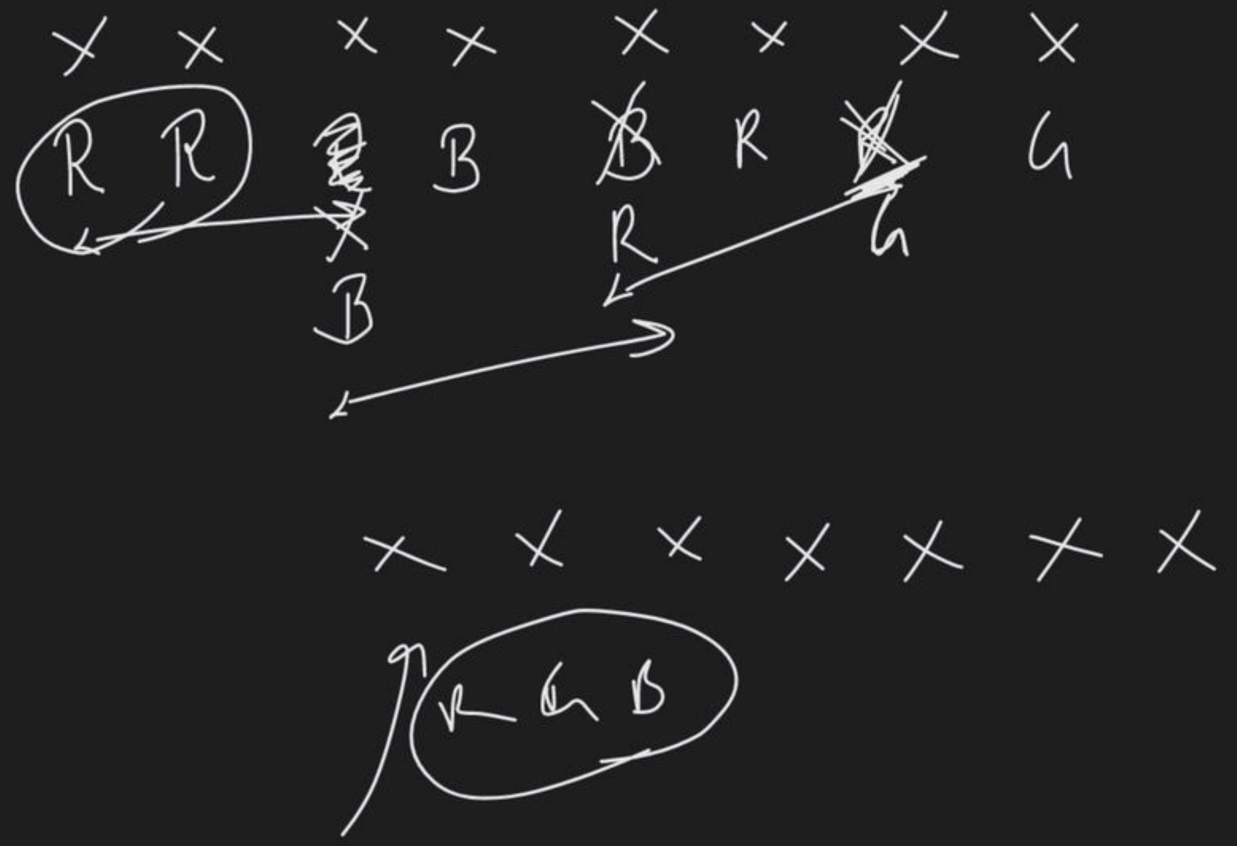
Dynamic Programming Class - 3

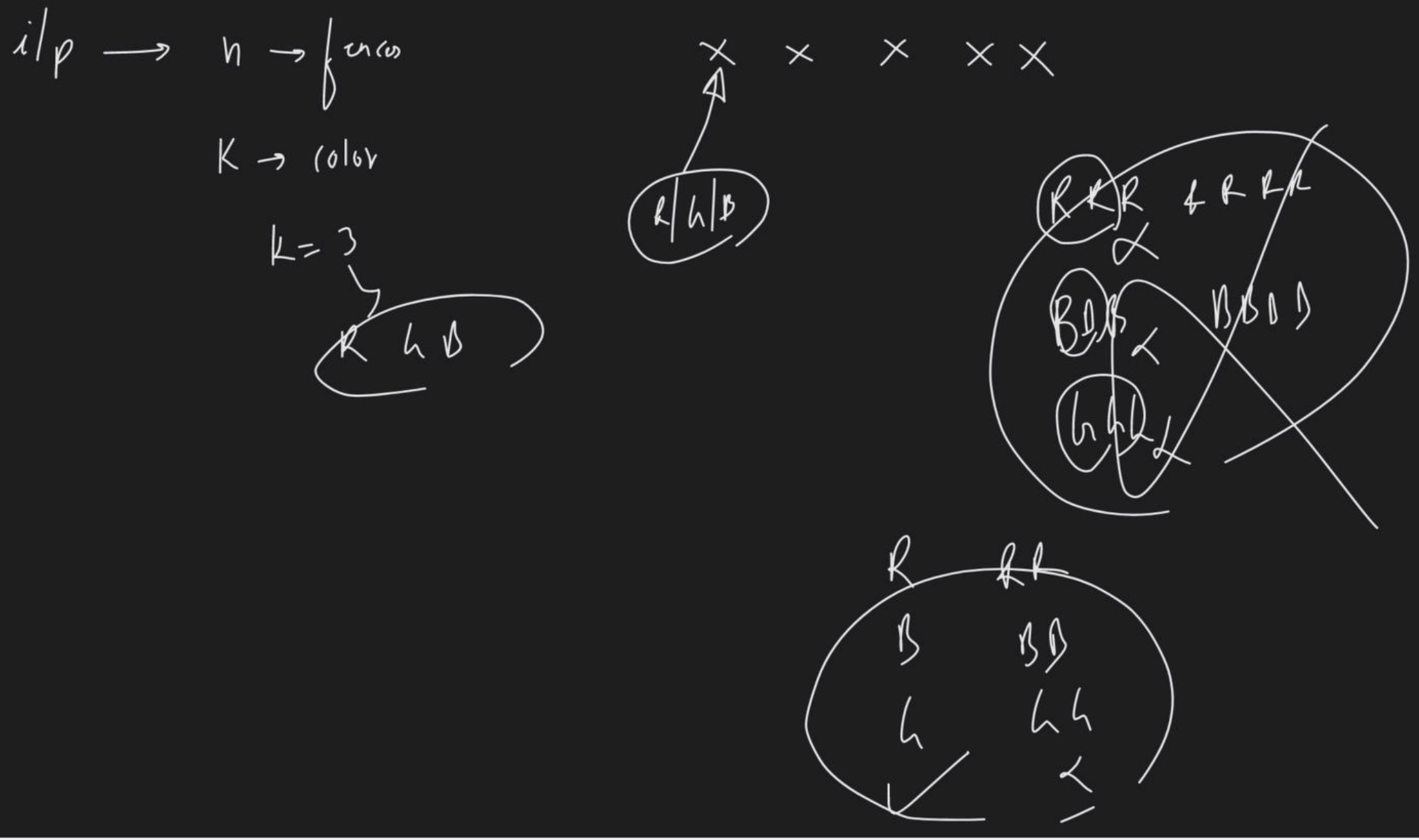
Special class

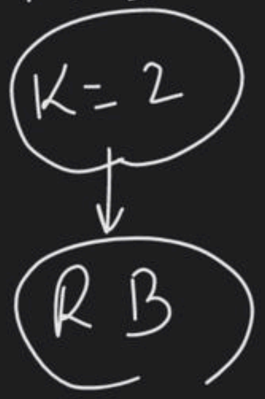
Love Babbar • June 9, 2023

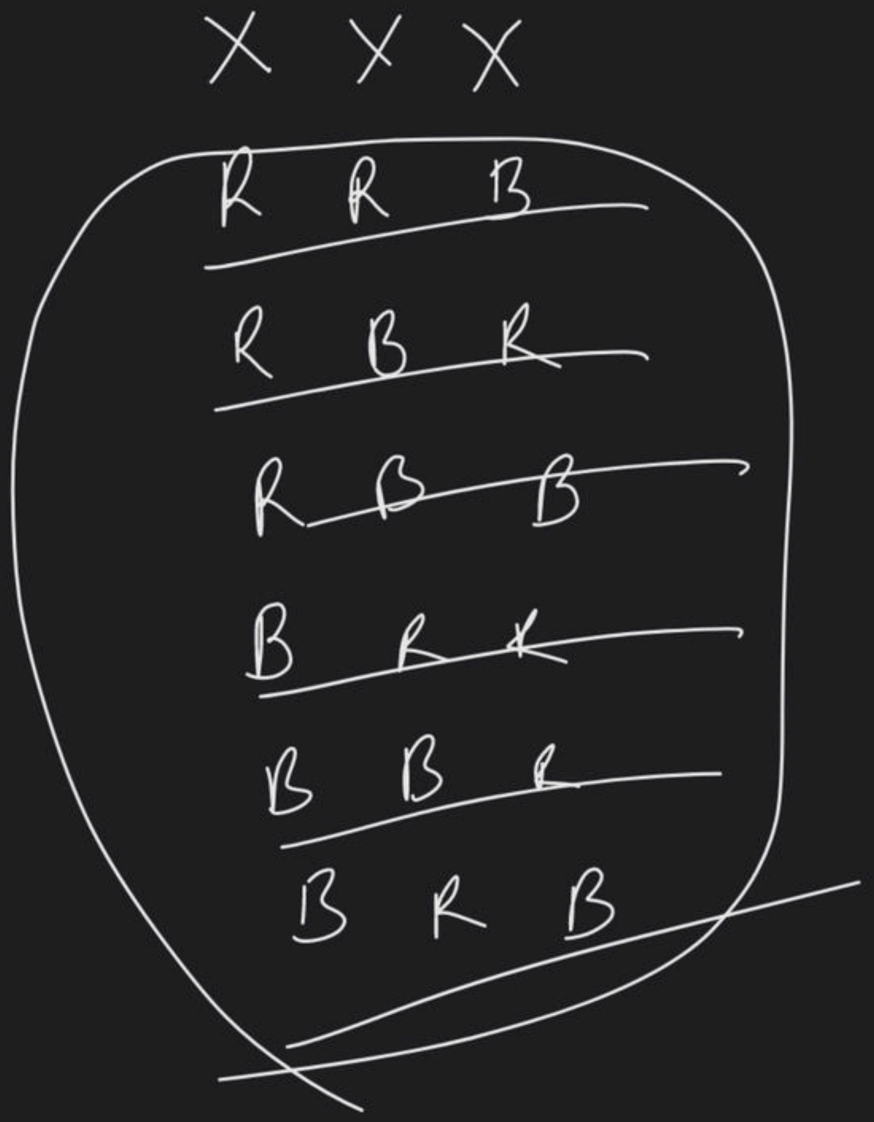
Painting Fence

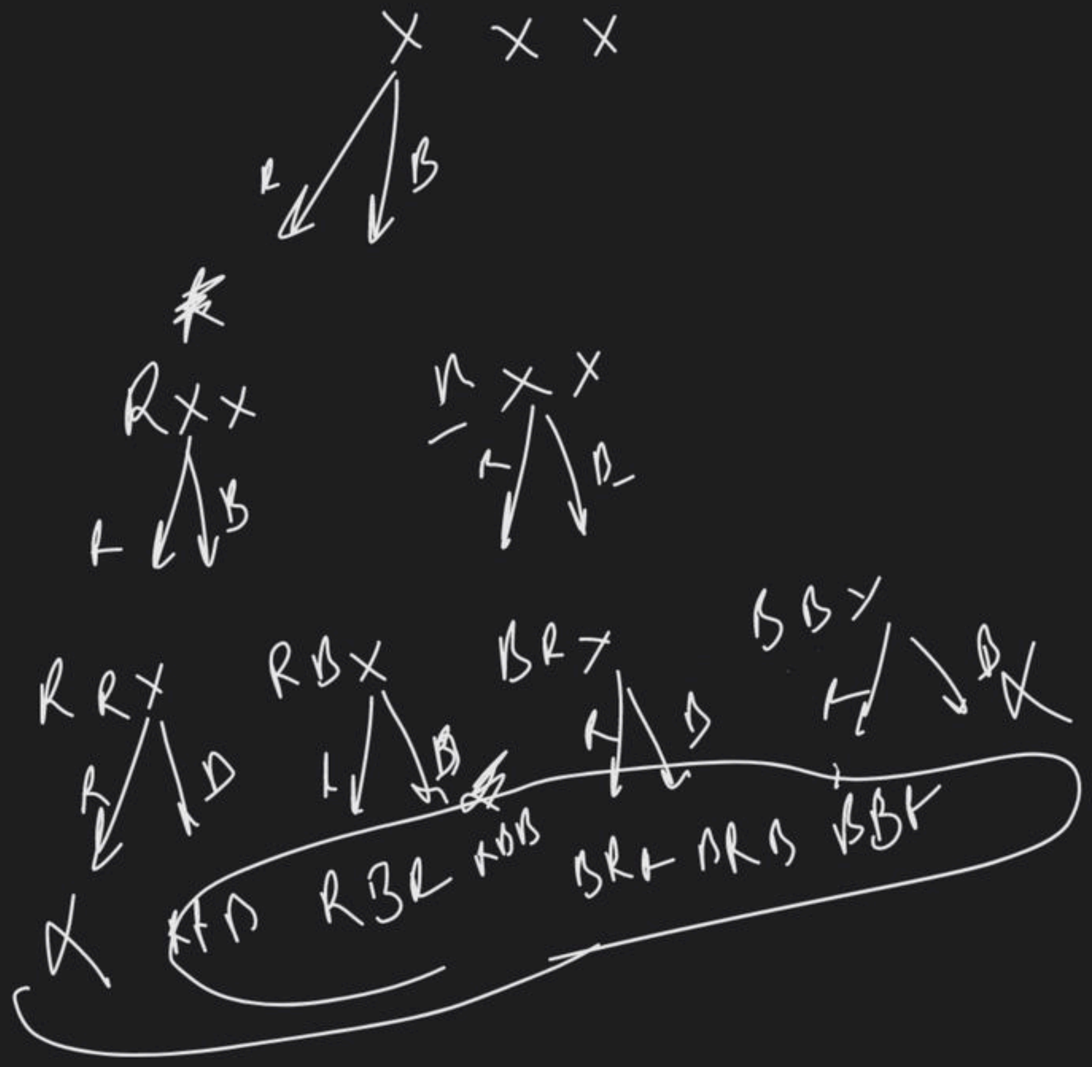
R G B

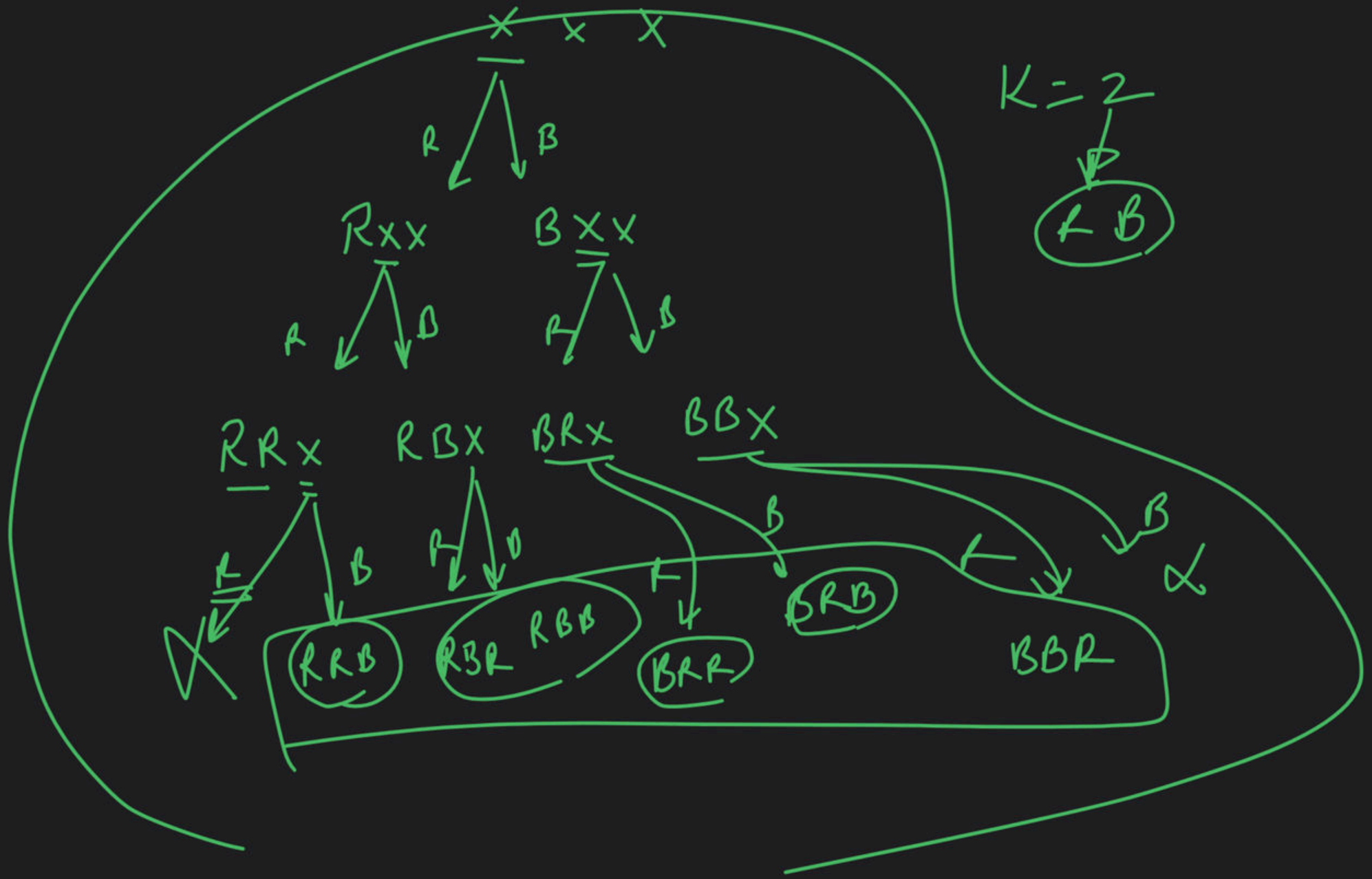


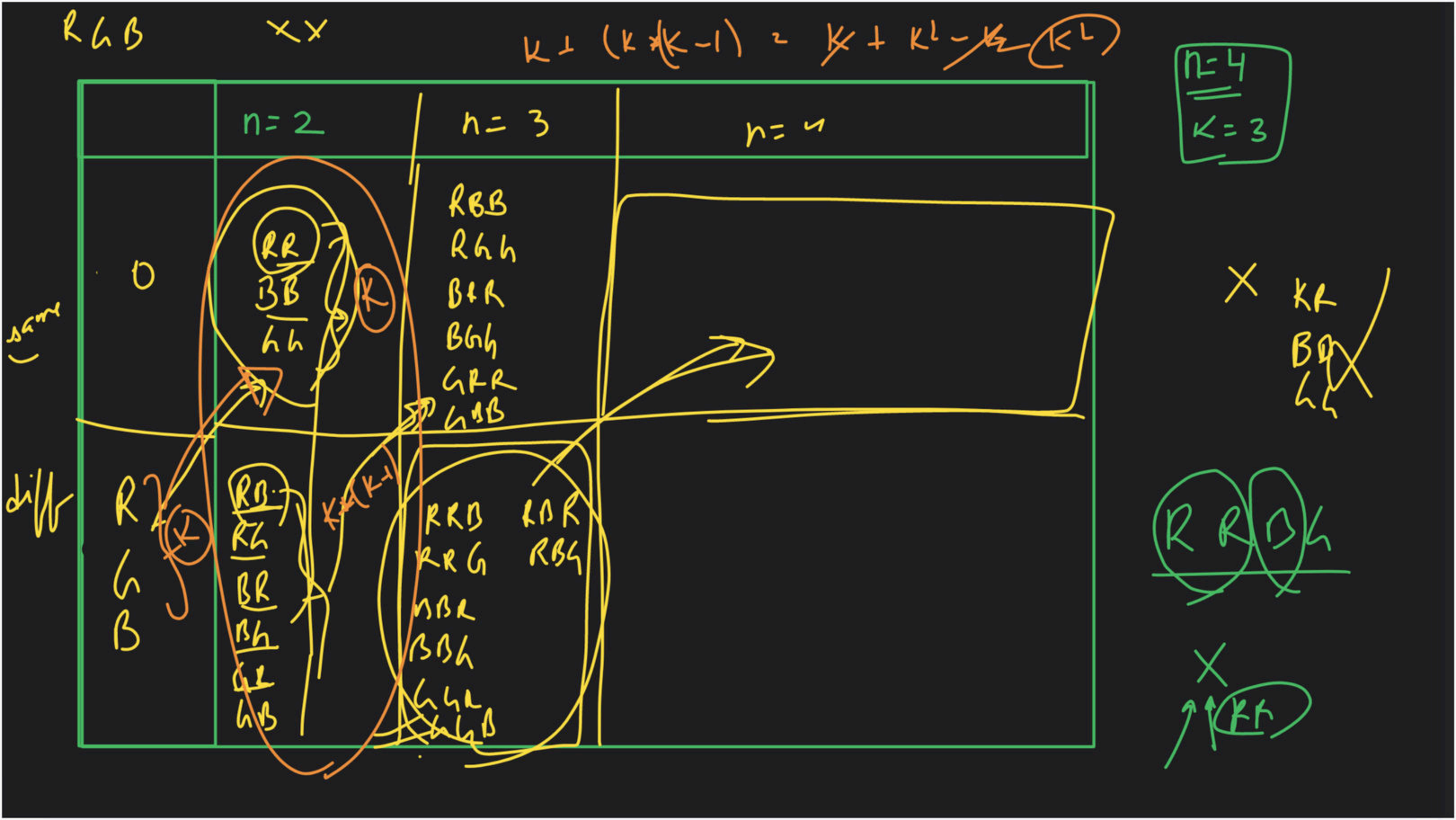

i/p → n → function
K → color
K = 3
R G B
R/G/B

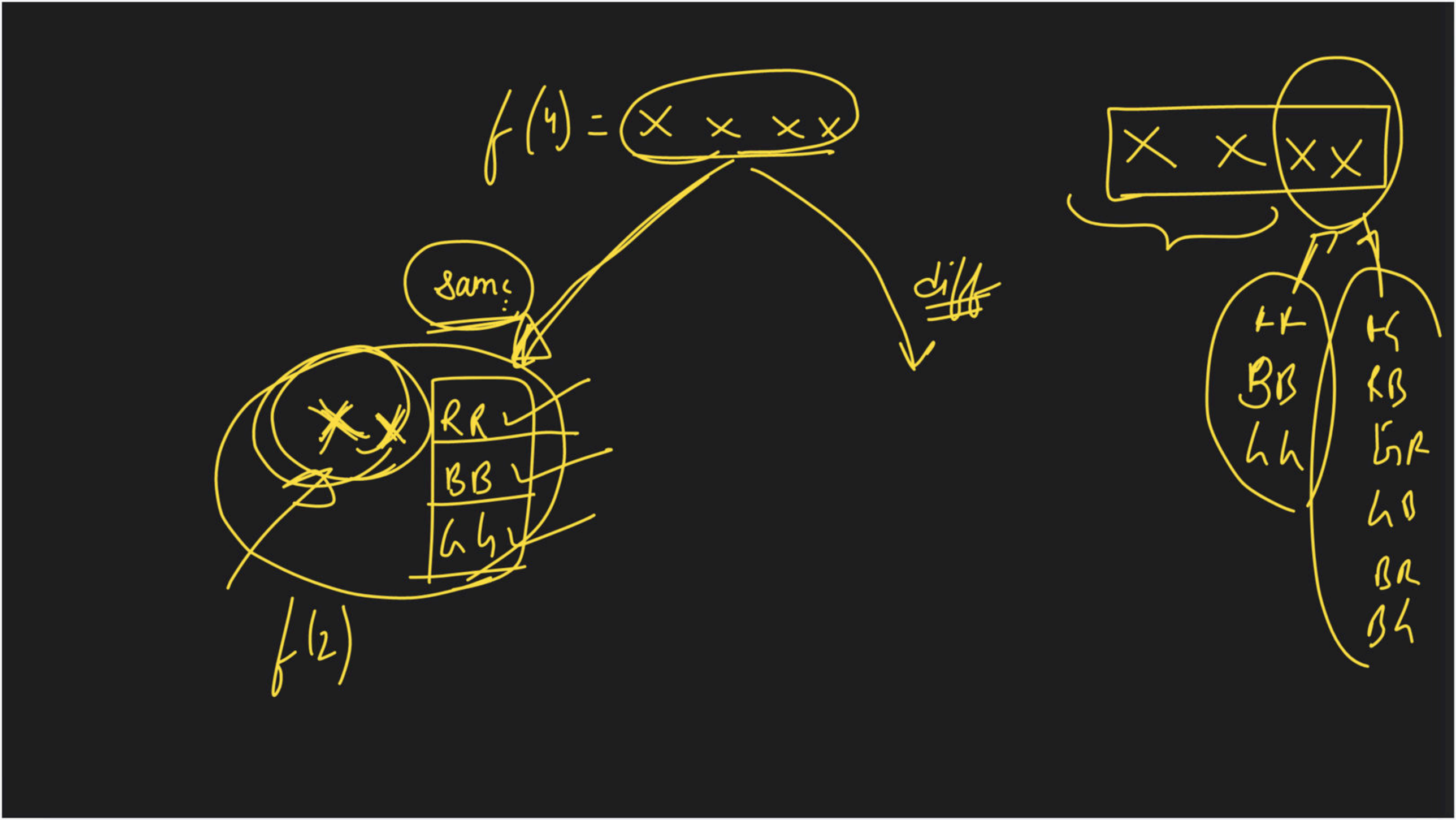

n=3
K=2
R B

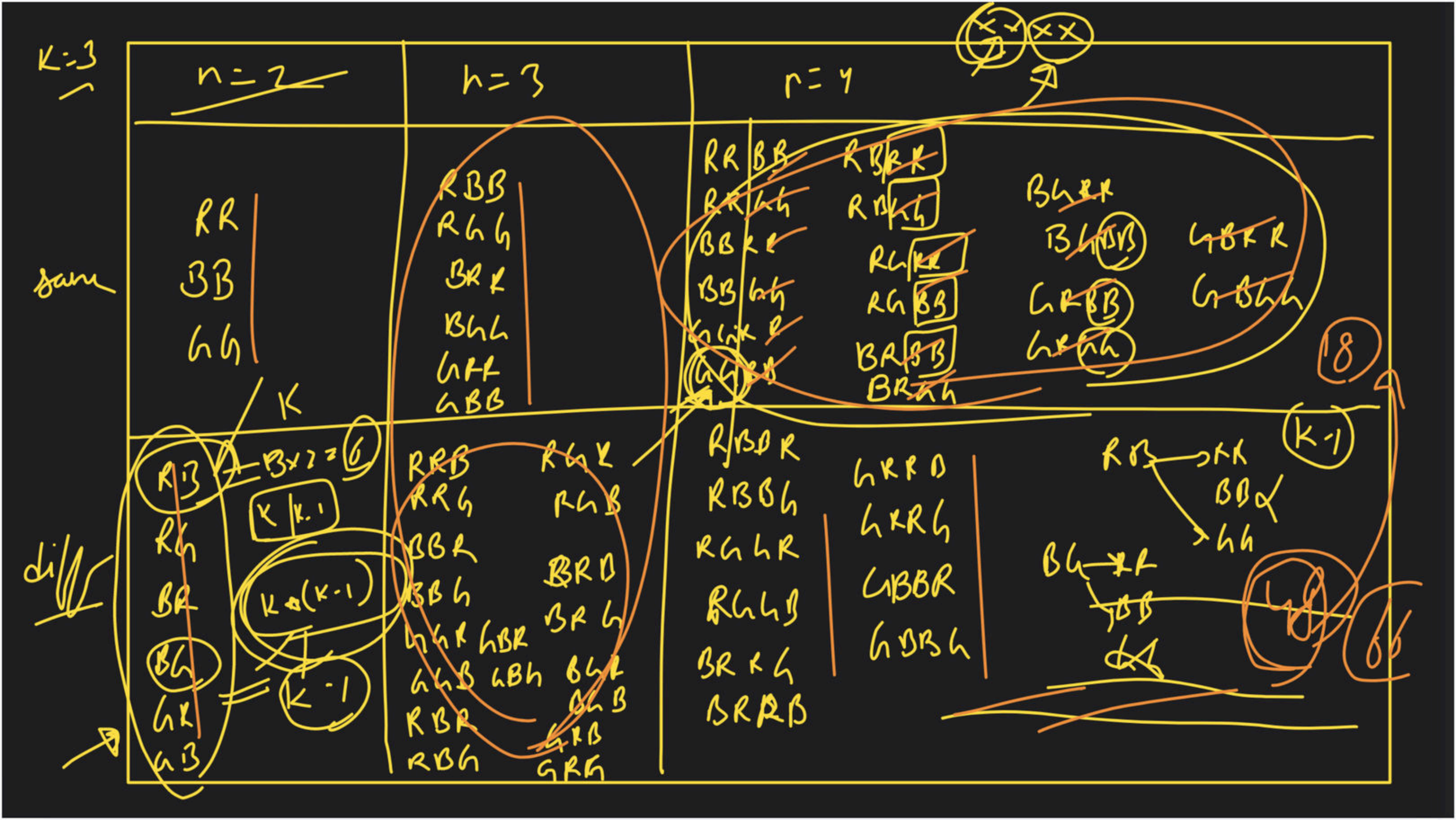

X X X
R R B
R B R
R B B
B R R
B B R
B R B

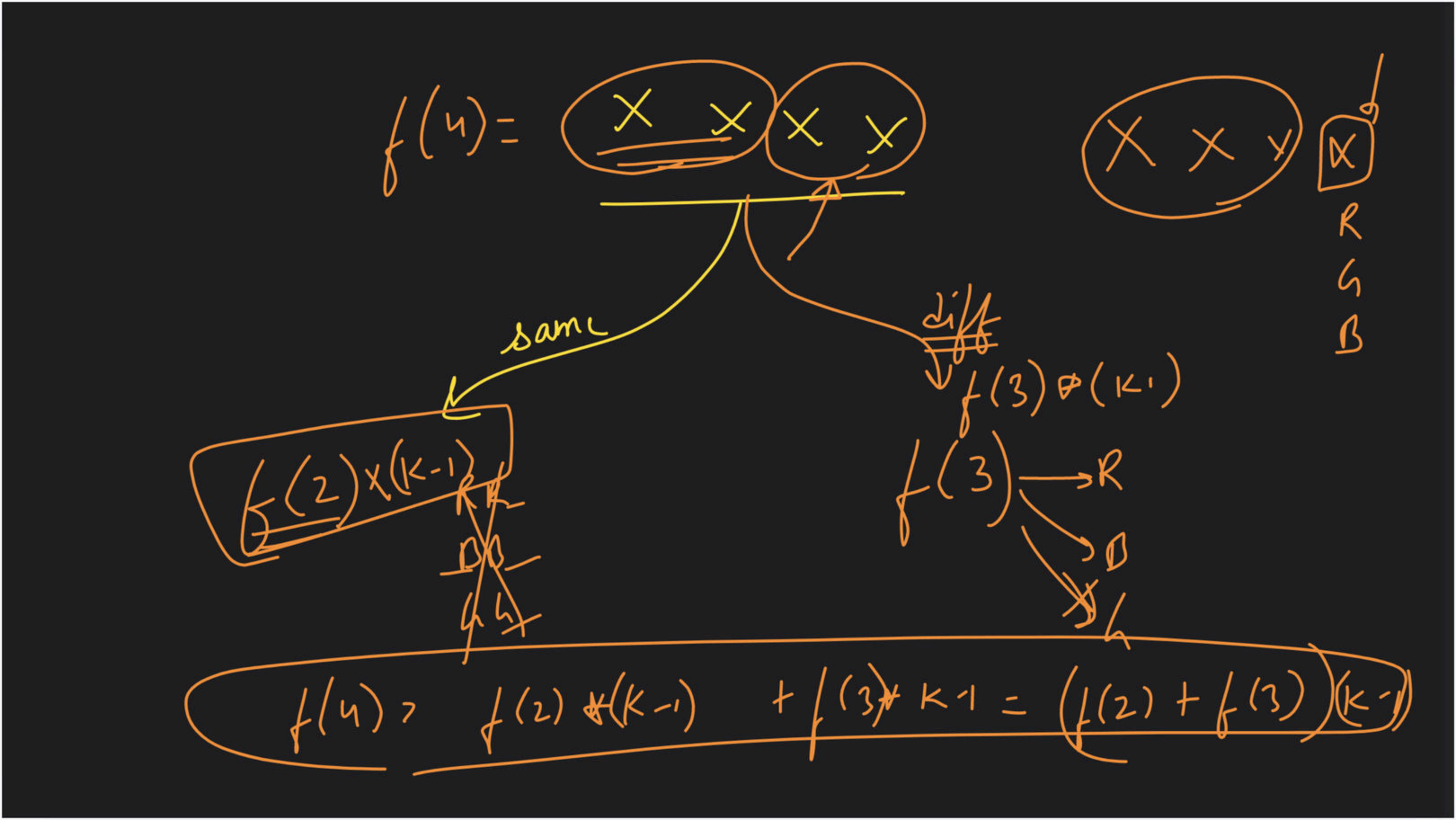



XXX
R
B
RXX
BXX
K=2
KB
RRX
RBX
BRX
BBX
RRB
RBR
RBB
BRR
BRB
BBR

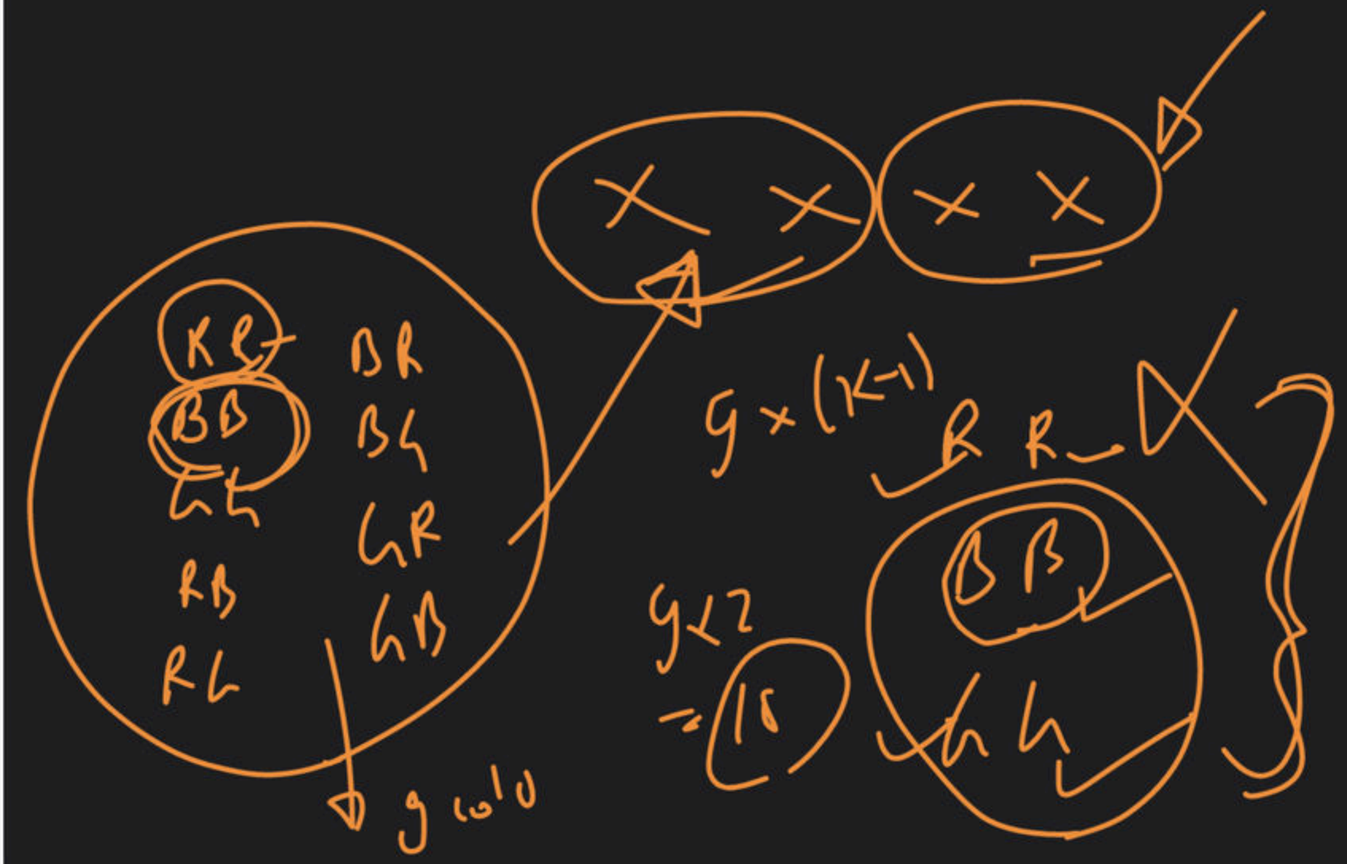

RGB
XX
n=2
n=3
n=4
RR
BB
GG
RBB
RGG
BRR
BGG
GRR
GBB
RRB
RRG
RBR
RBG
BRR
BBG
GRR
GBB
n=4
K=3
same
diff
R
G
B
KR
RRBG


f(4) =
same
diff
RR
BB
GG
f(2)


K=3
n=2
h=3
n=4
same
diff
K
K-1
K(K-1)
8
6


f(4) =
X X X X
X X Y
X
R
G
B
same
diff
f(3) # (K-1)
f(3)
R
B
f(2) x (K-1)
f(4) = f(2) # (K-1) + f(3) # K-1 = (f(2) + f(3)) (K-1)

$$f(n) = [f(n-2) + f(n-3)] \% (K-1)$$



$$f(4) = \left(f(2) + f(3)\right) + (k-1)$$

$$= (9 + 24) \times (3-1)$$

$$= 33 \times 2$$

$$= 11$$

$$f(n) = \left[f(n-1) + f(n-2) \right] * (K-1)$$

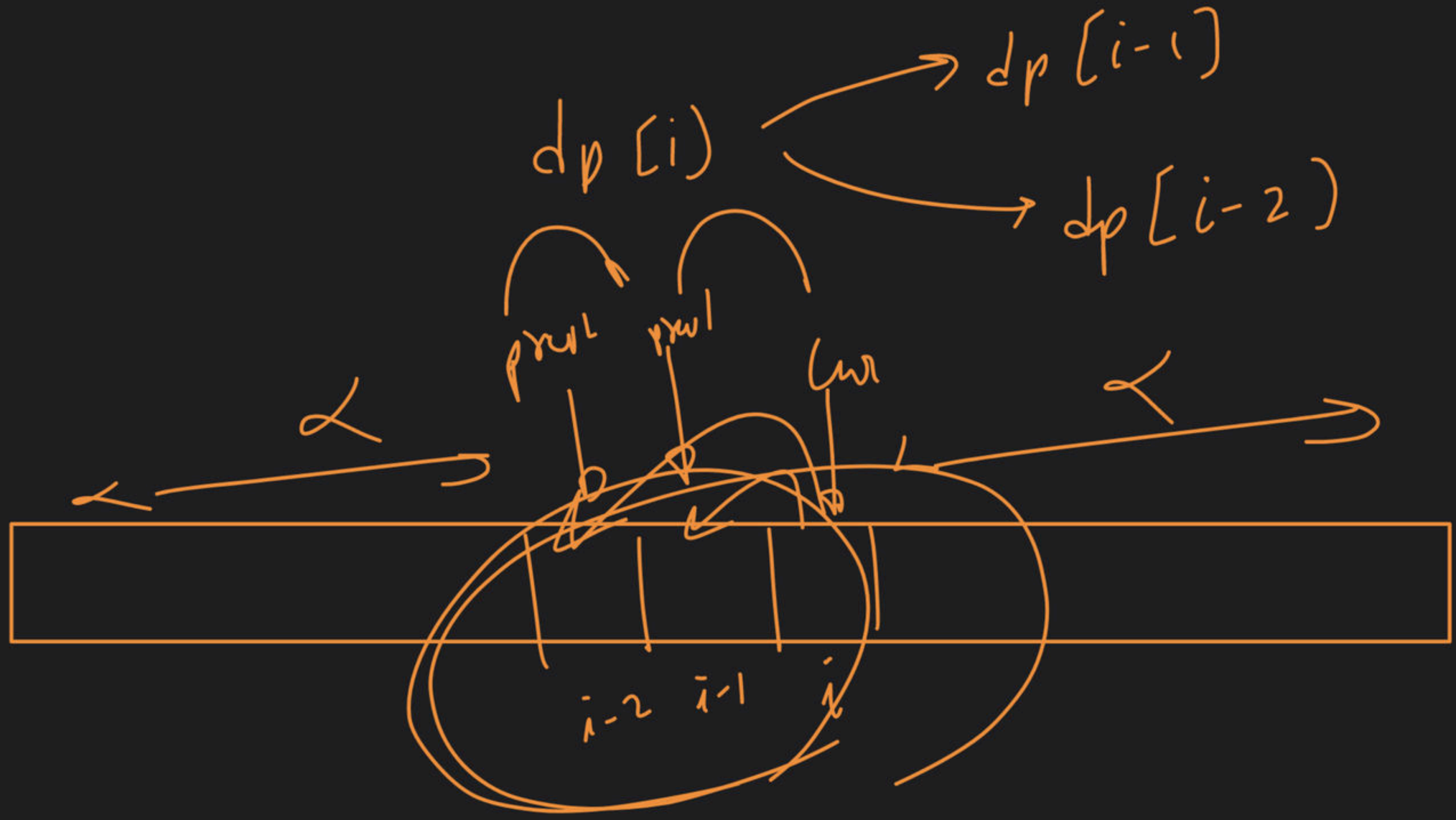
$$f(4) = (f(4-1) + f(4-2)) * (K-1)$$

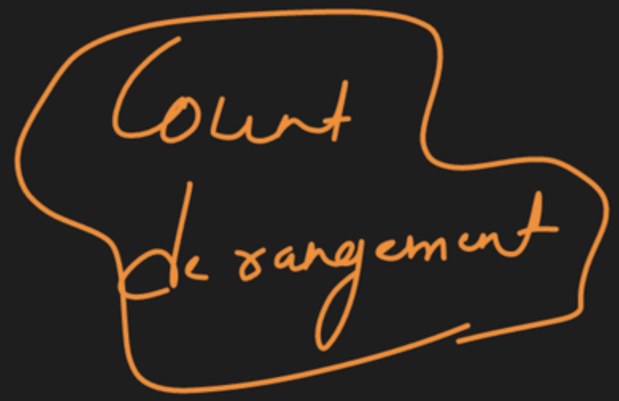
$$= (f(3) + f(2)) * 2$$

$$= (24 + 9) * 2$$

$$= 33 \times 2$$

$$= 66$$

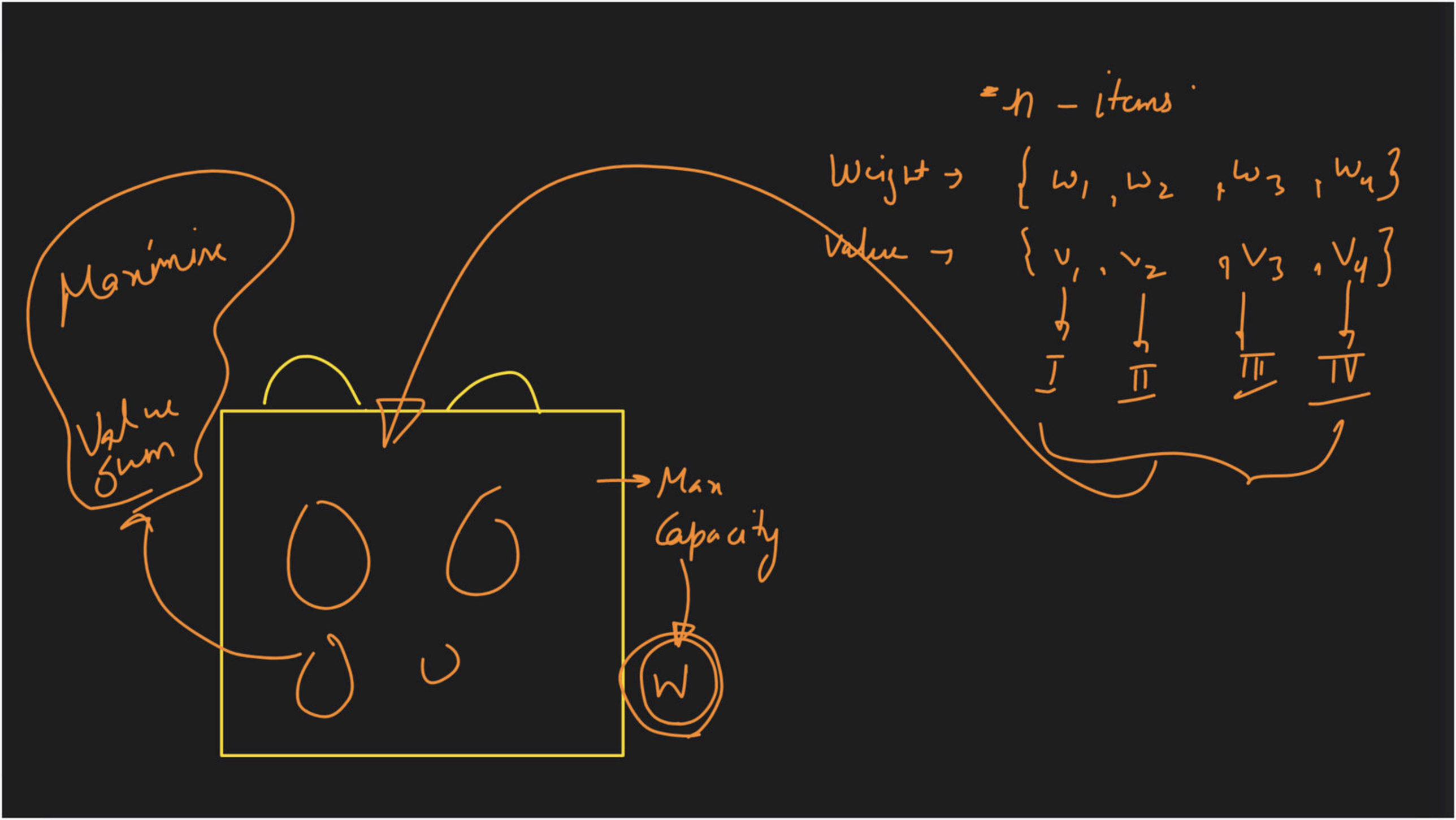

dp[i]
dp[i-1]
dp[i-2]
prv2
prv1
cur
i-2
i-1
i

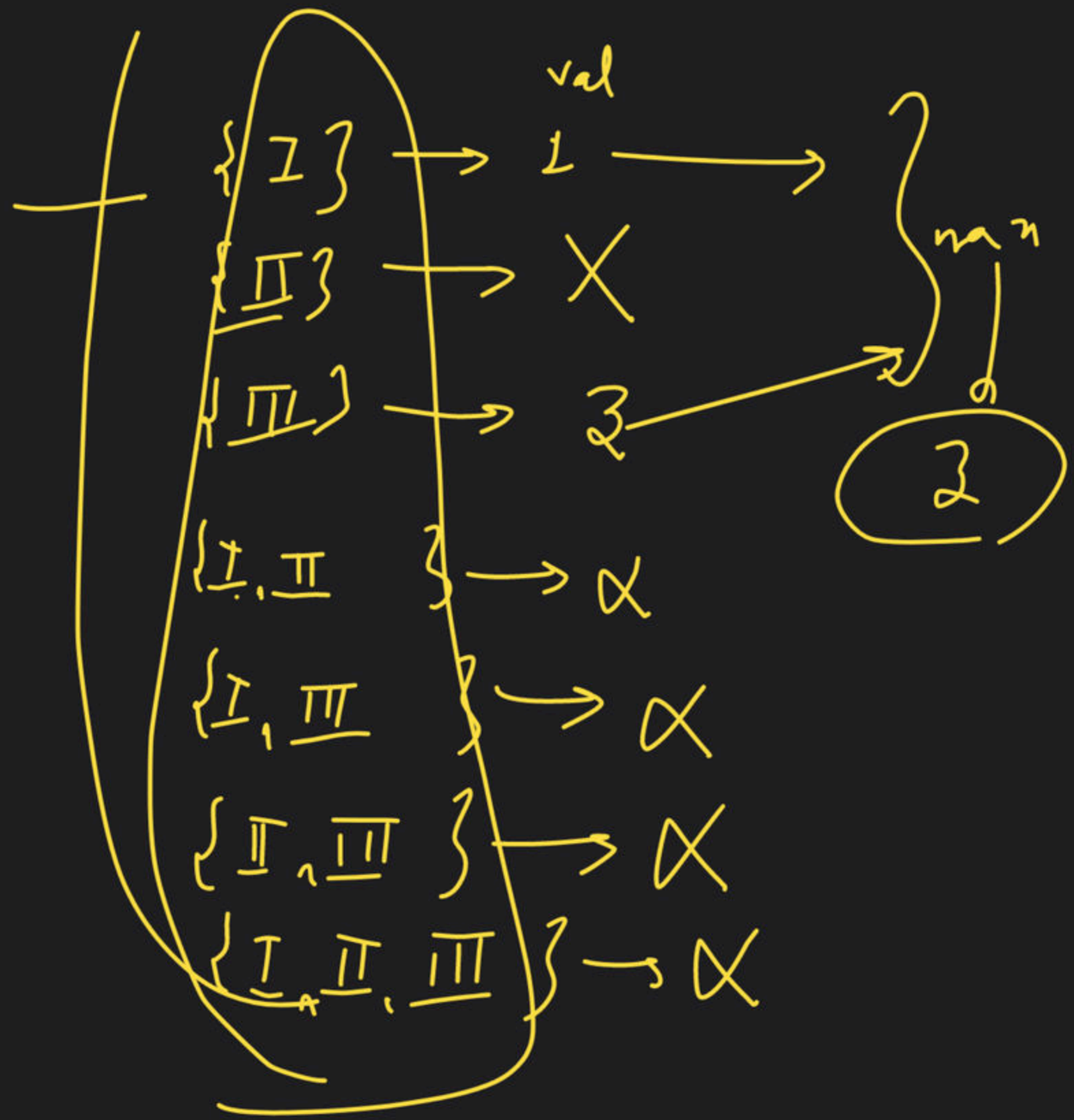

Count
derangement

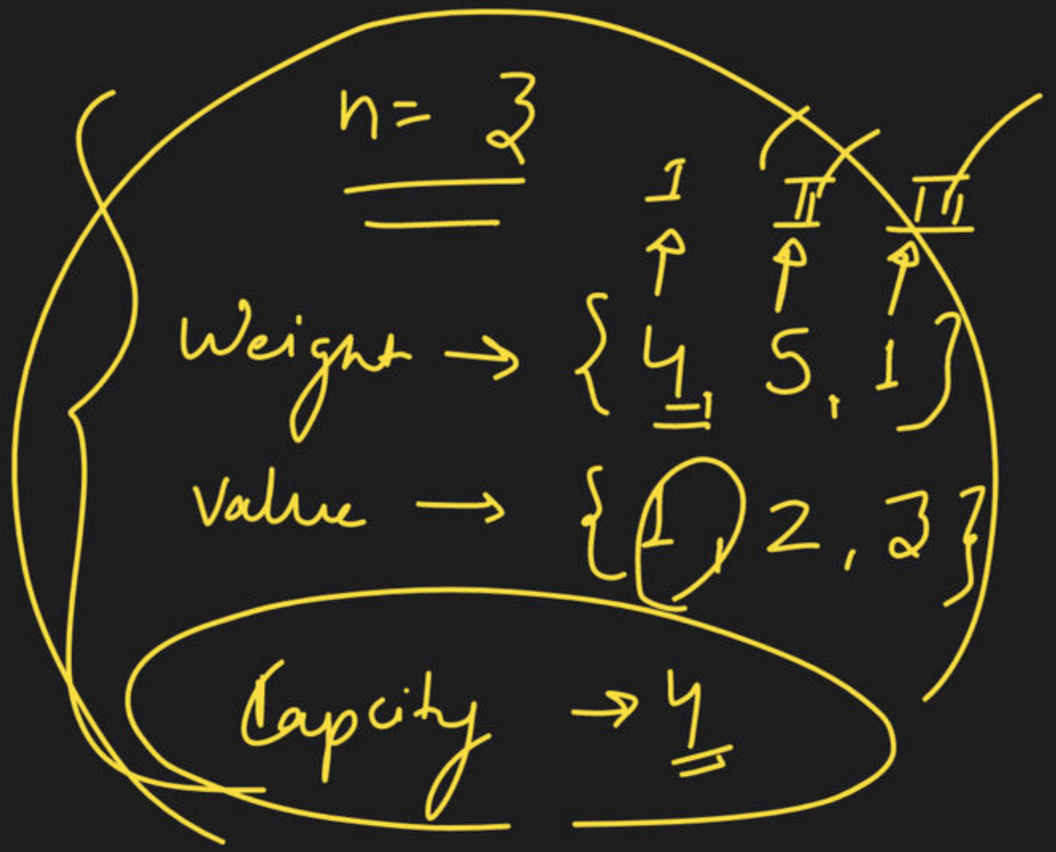
0/1 Knapsack Problem :-

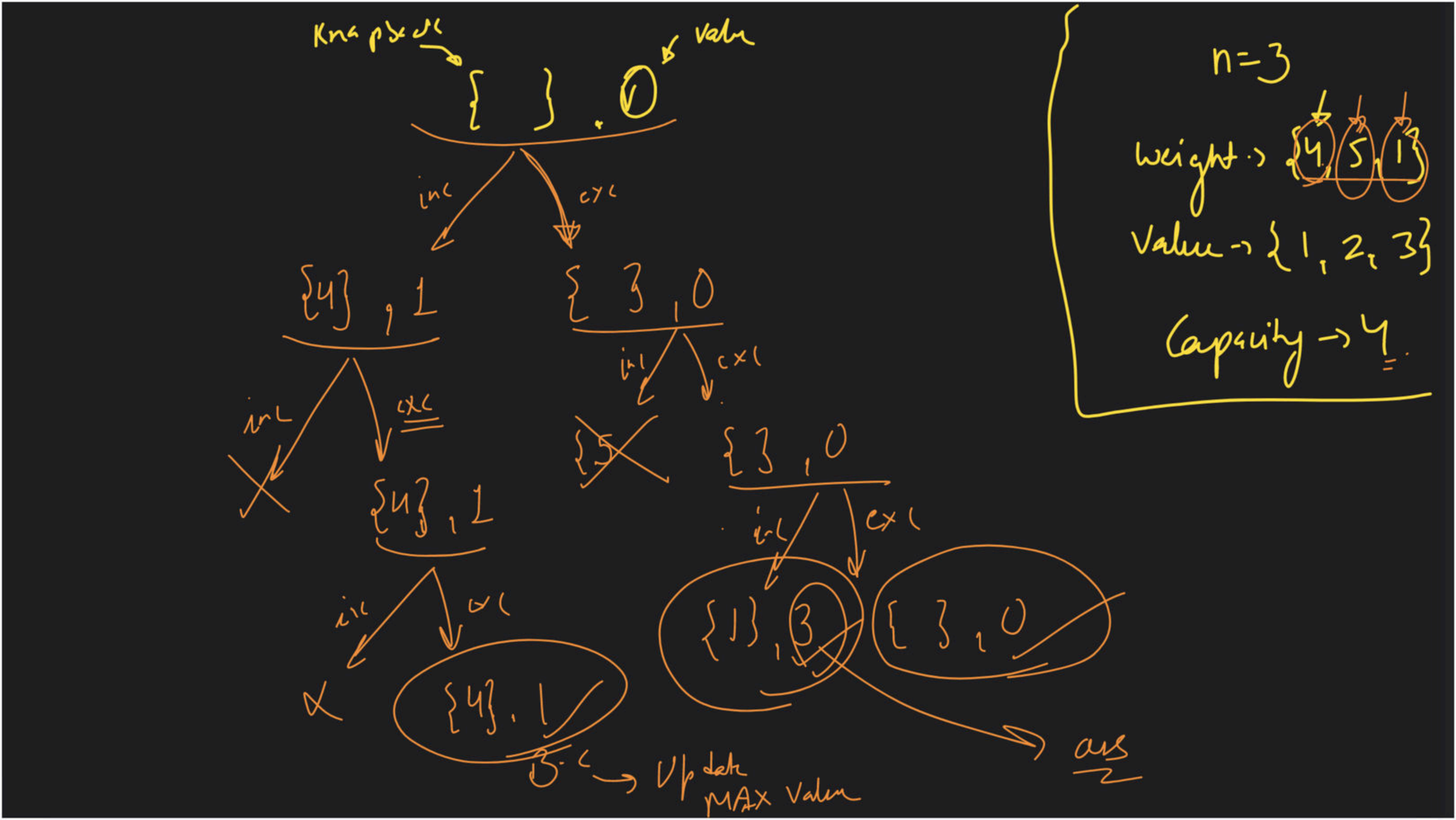
Same pattern

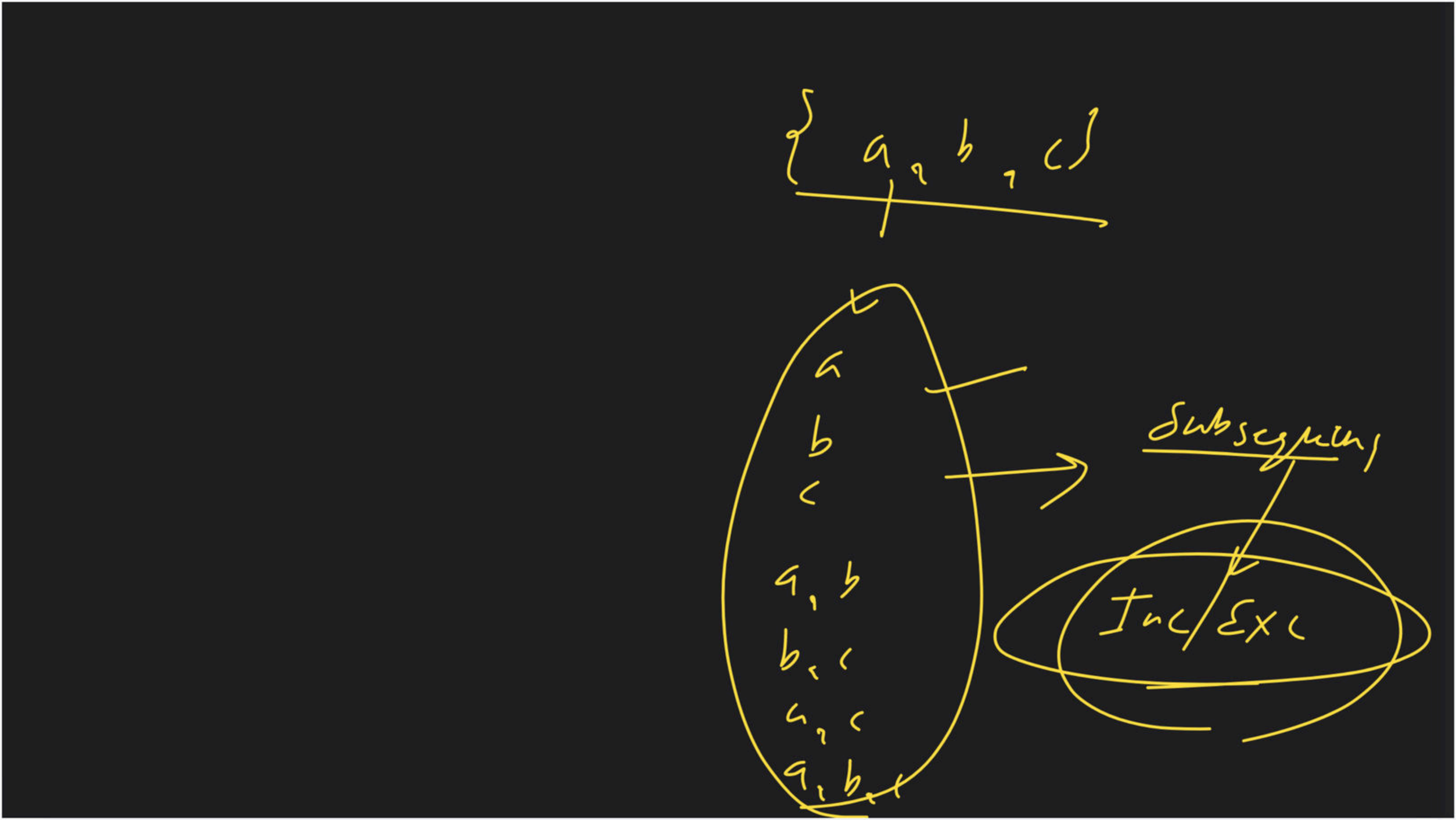
- Subset Sum
- Equal Subset sum partition
- Min Subset sum difference

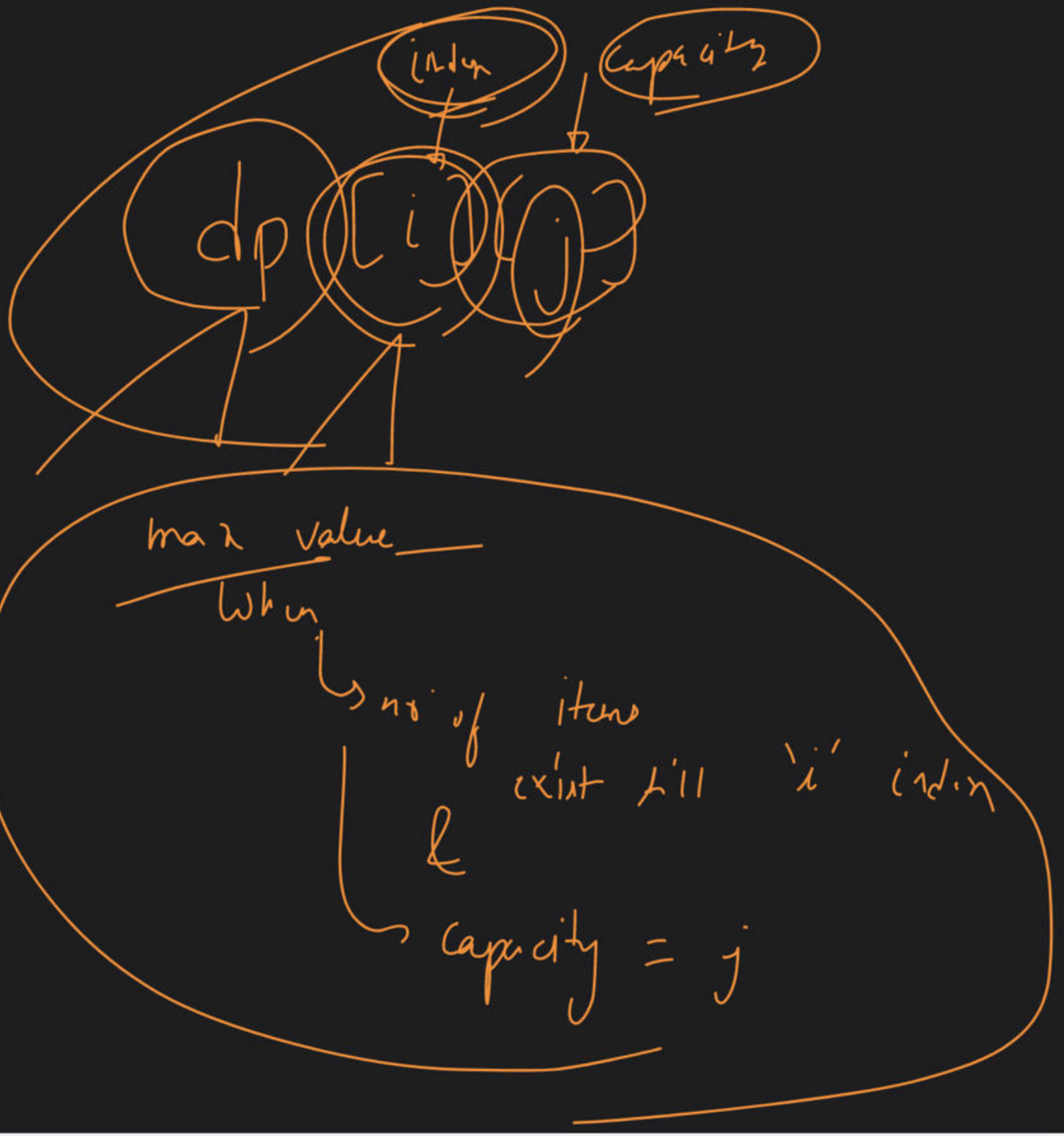

n - items
Weight → {w1, w2, w3, w4}
Value → {v1, v2, v3, v4}
I
II
III
IV
Maximise
Value Sum
Max Capacity
W

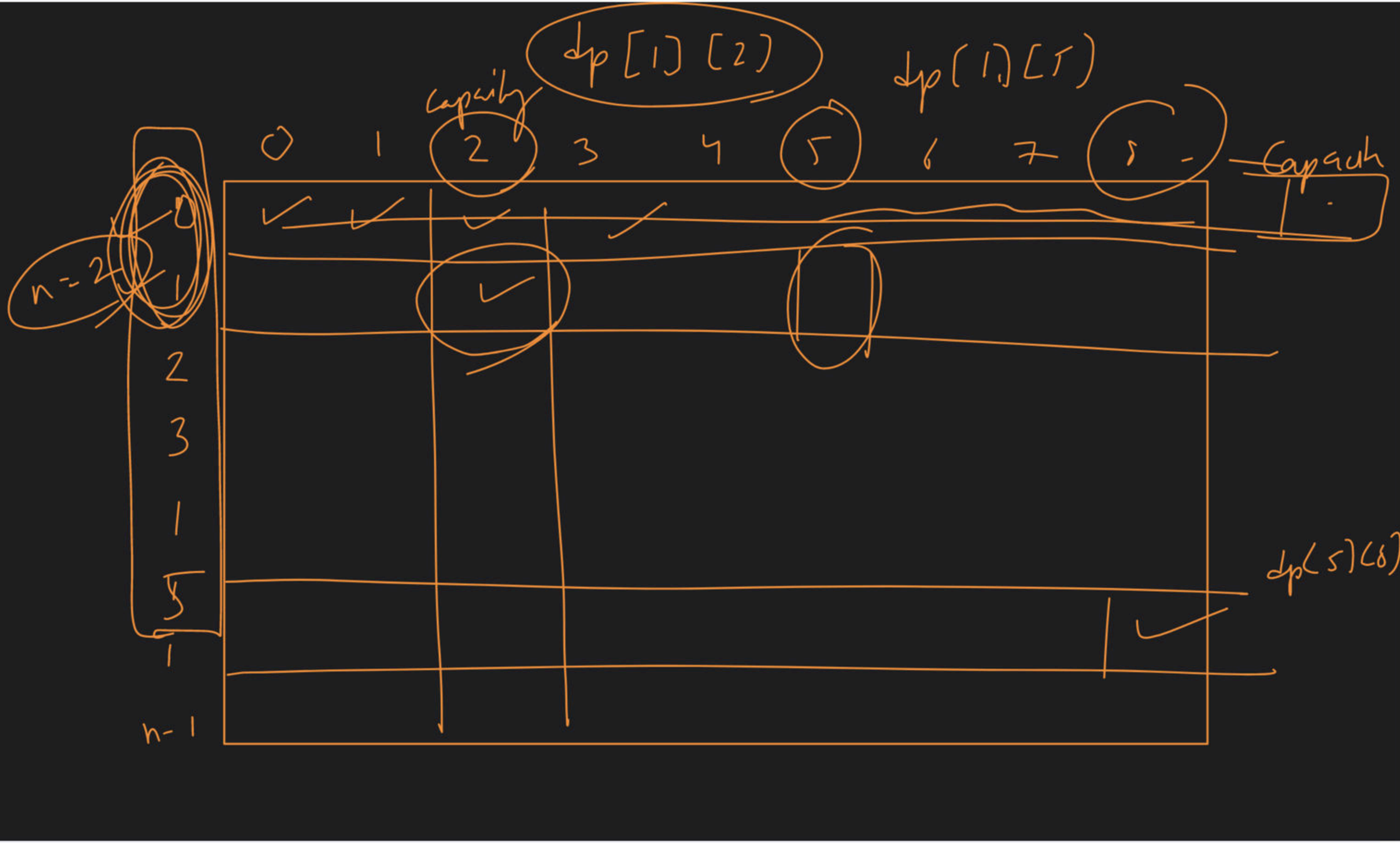

val
{I}
{II}
{III}
{I, II}
{I, III}
{II, III}
{I, II, III}
1
X
3
max
3
α
α
α
α

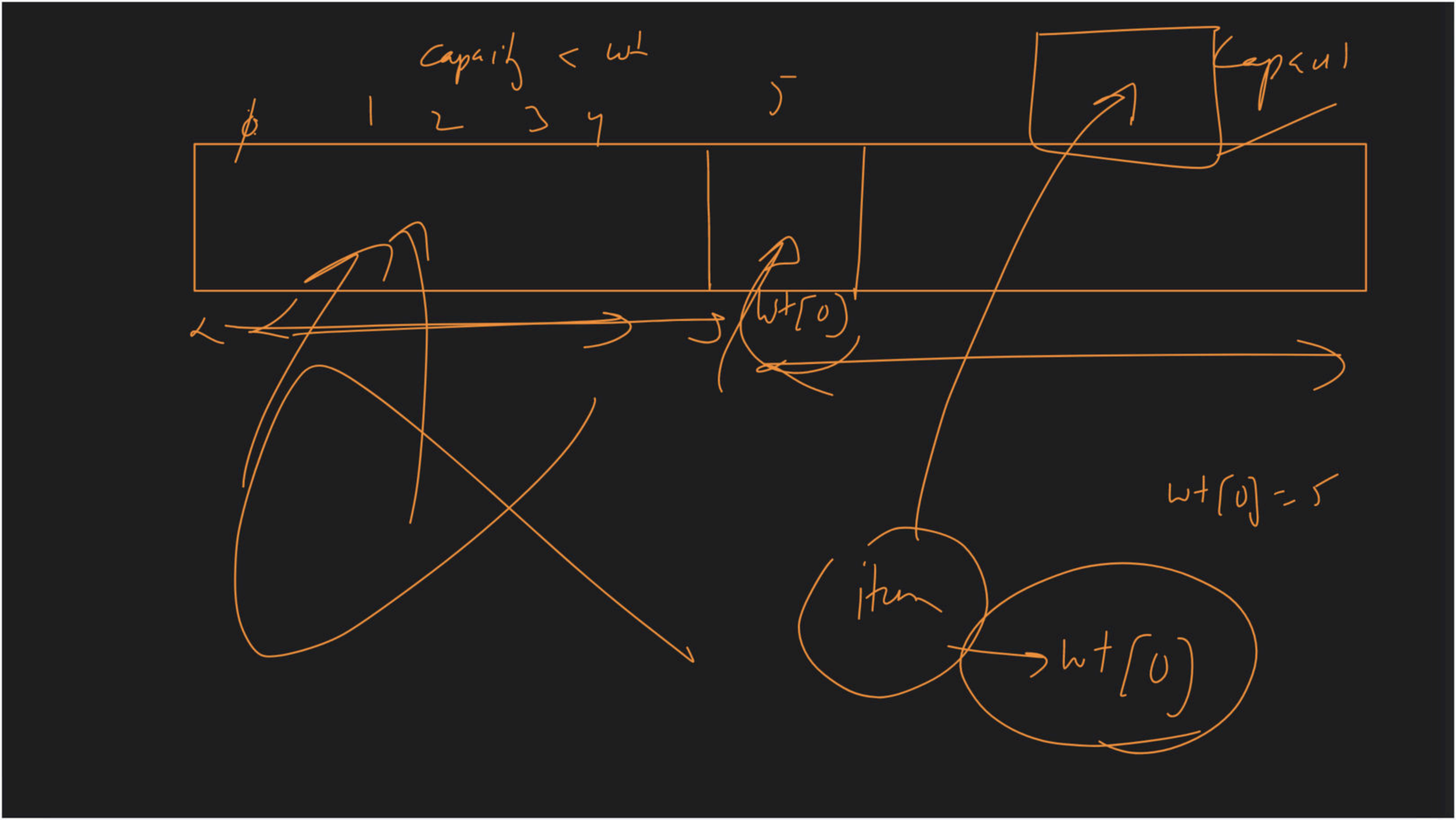

n = 3
I
II
III
Weight → {4, 5, 1}
Value → {1, 2, 3}
Capcity → 4

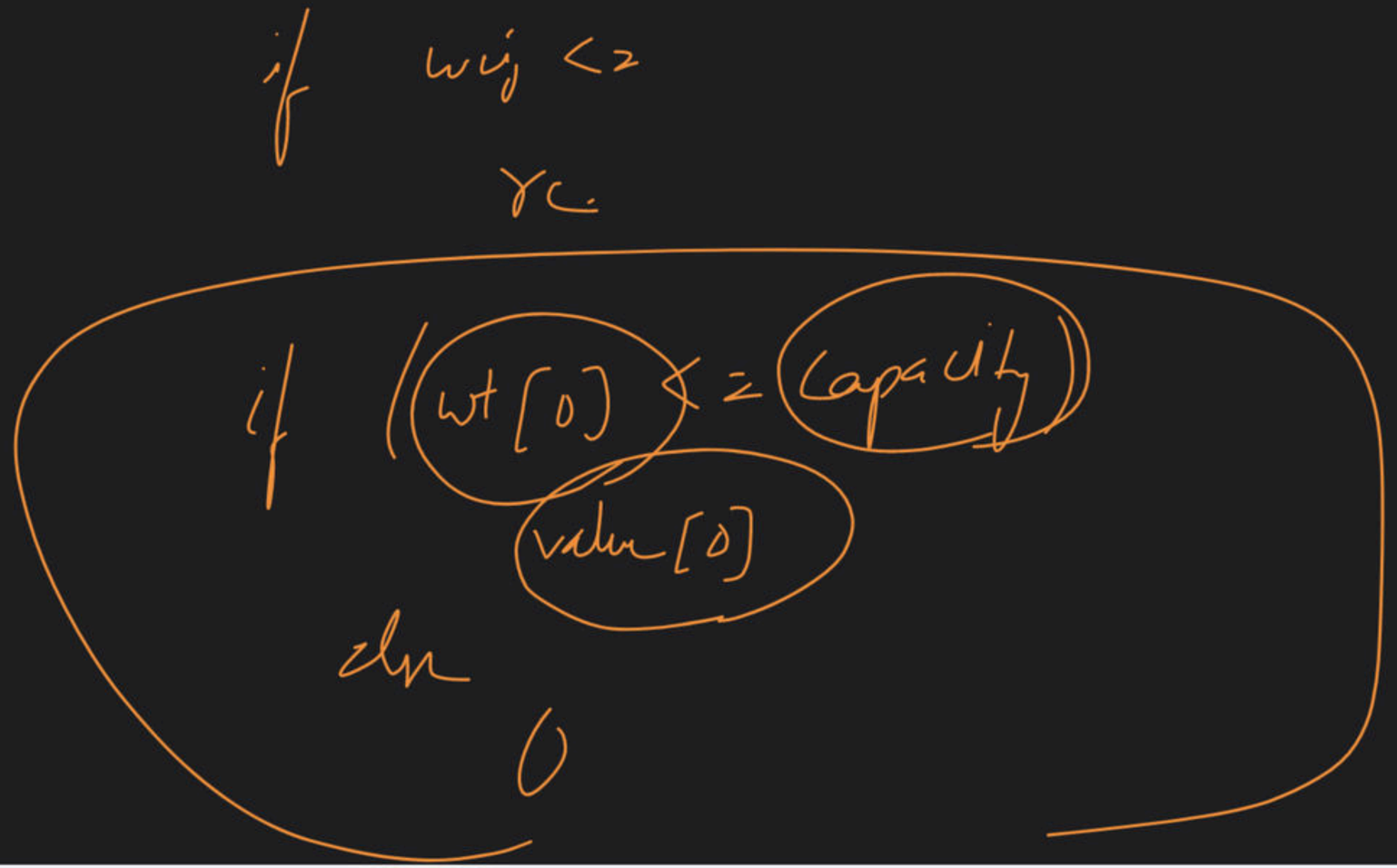

Knapsack
Value
{ } , 0
inc
exc
{4} , 1
{ } , 0
inl
exc
inl
exc
{4} , 1
{ } , 0
inc
exc
inc
exc
{4} , 1
{1} , 3
{ } , 0
3 < → Update MAX Value
ans
n=3
weight → {4, 5, 1}
Value → {1, 2, 3}
Capacity → 4.

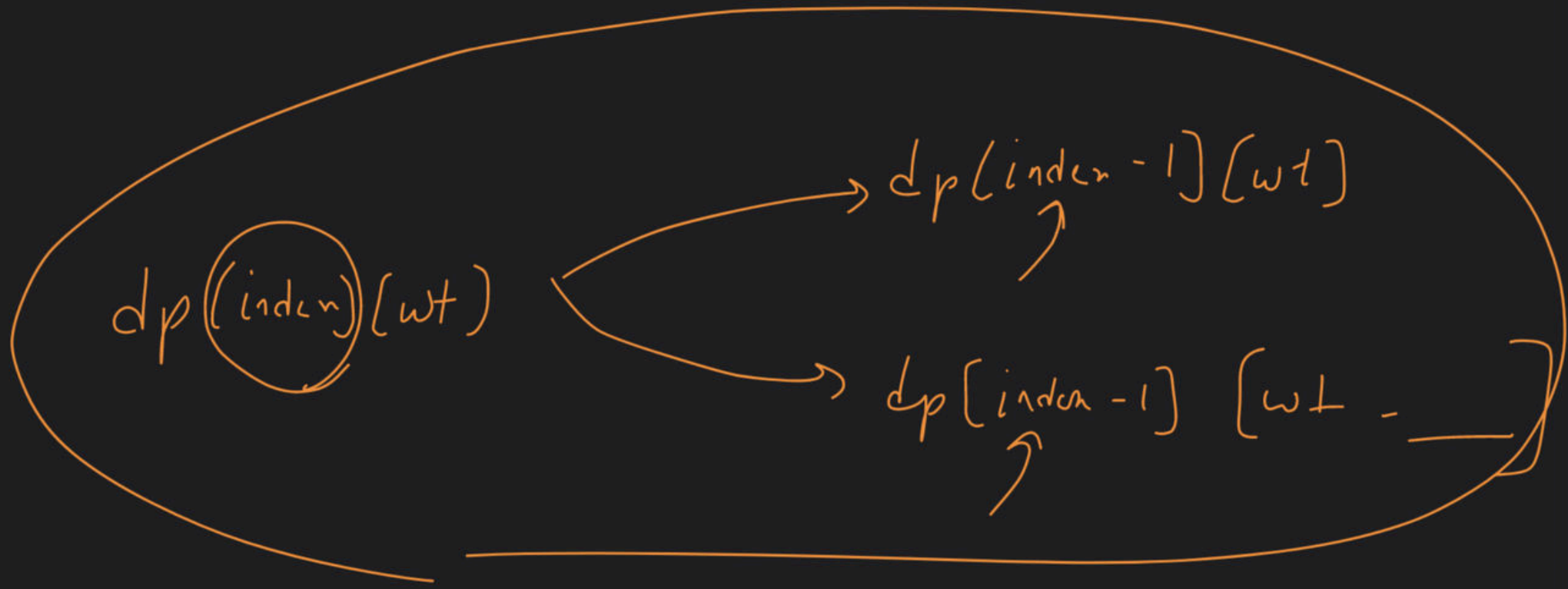

{a, b, c}
a
b
c
a, b
b, c
a, c
a, b, c
Subsequence
Inc/Exc

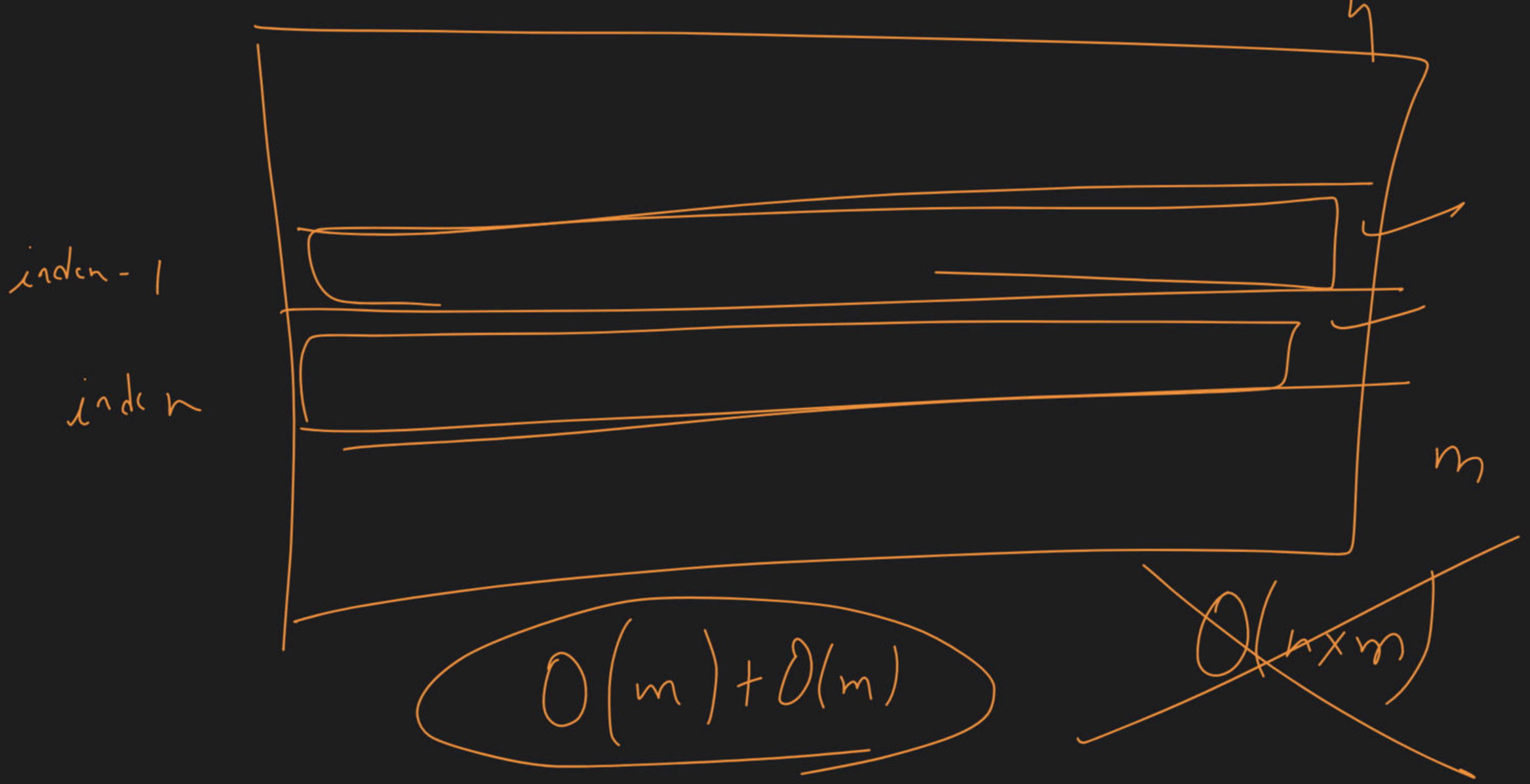

index
capacity
dp [i] [j]
max value
When
no of items
exist till 'i' index
&
capacity = j


dp[1][2]
dp[1][7]
capacity
0
1
2
3
4
5
6
7
8
capacity
n=2
0
1
2
3
1
5
1
n-1
dp[5][8]

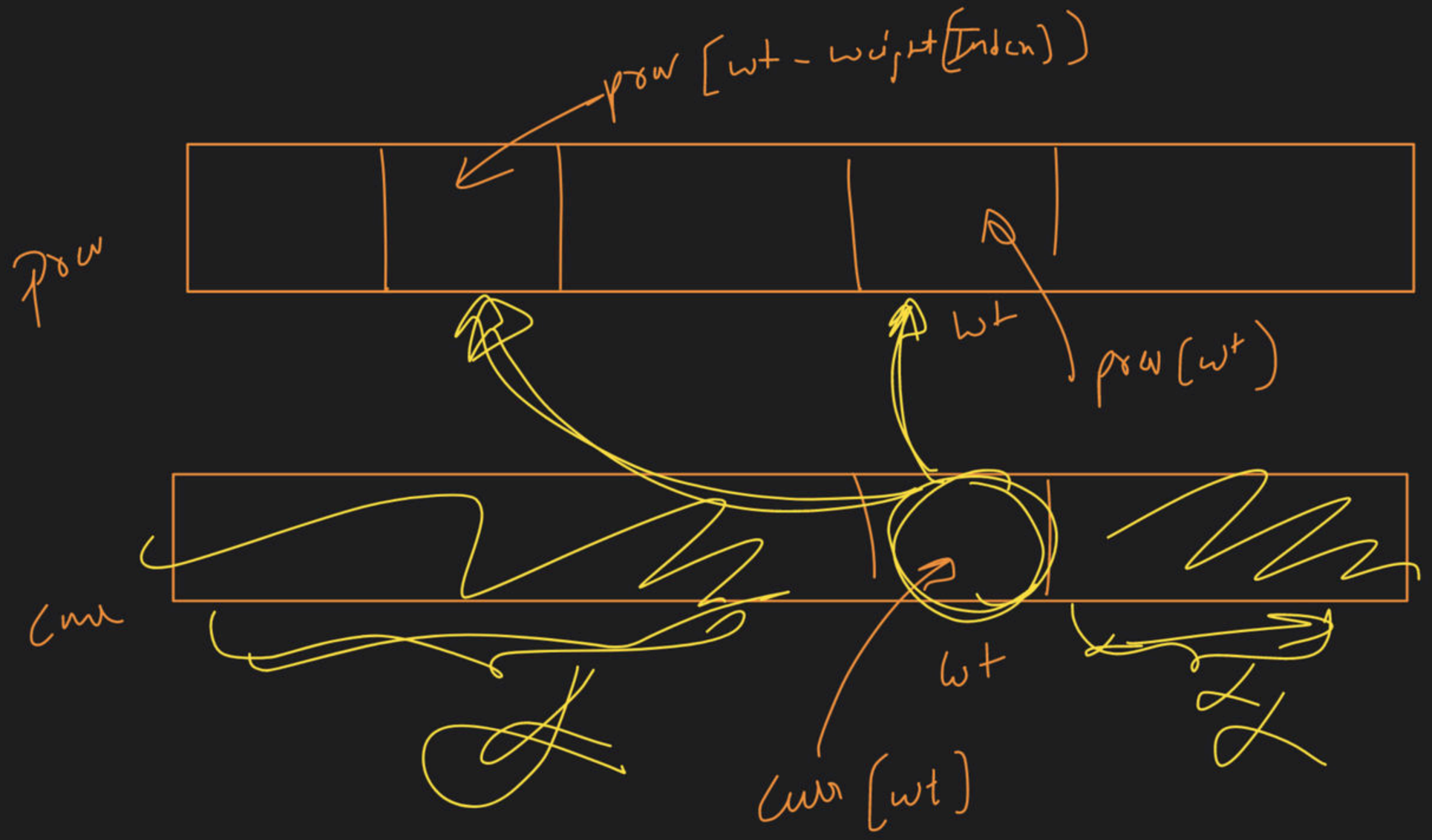

capacity < wt
0 1 2 3 4
j
cap[4]
wt[0]
wt[0] = 5
item
wt[0]

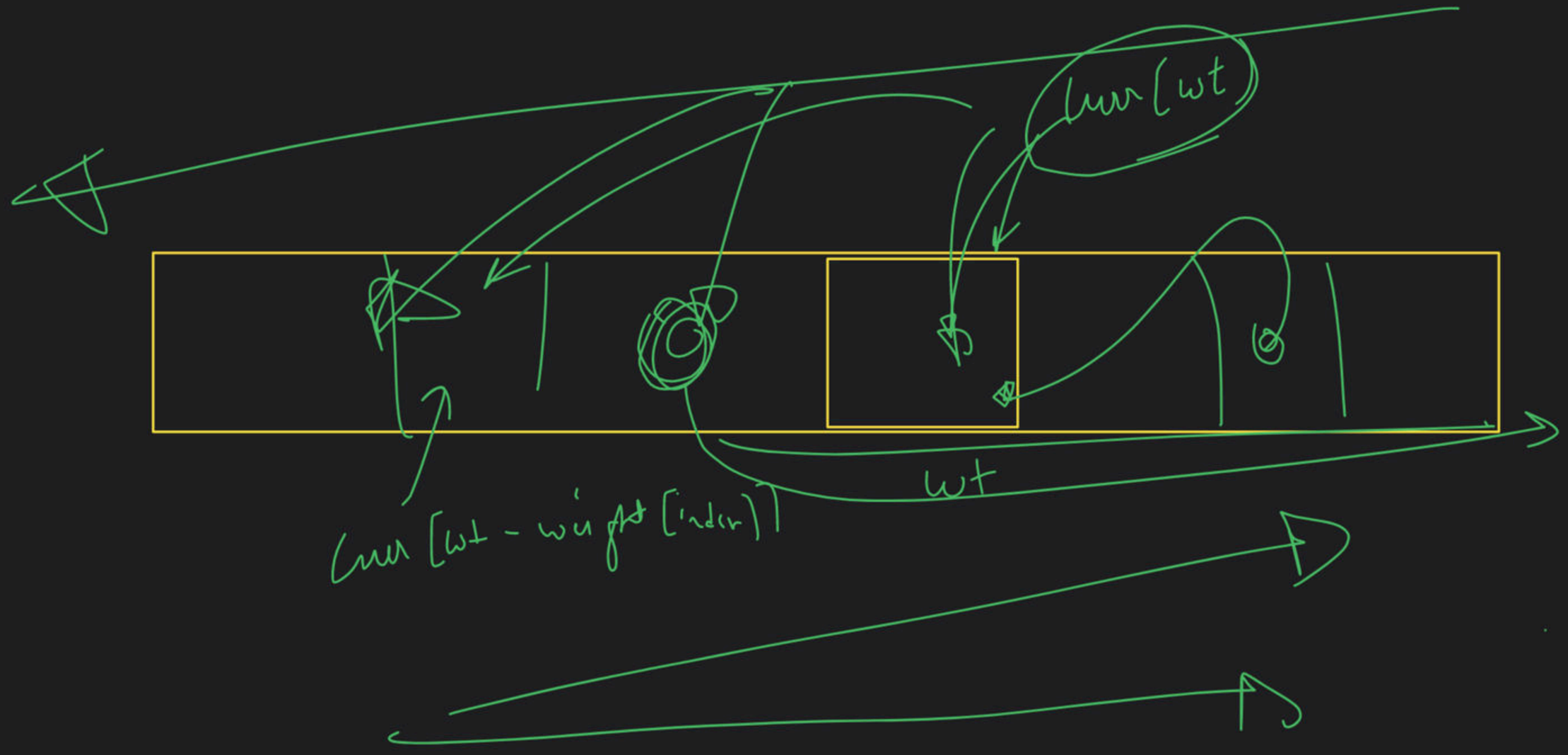

if wij <z
yc
if (wt[0] <= capacity)
value[0]
else
0


dp[index][wt]
dp[index-1][wt]
dp[index-1][wt-___]


index - 1
index
n
m
O(m) + O(m)
O(n x m)


prev
curr


prev[wt - weight[index]]
prev
wt
prev[wt]
curr
wt
curr[wt]


curr[wt
wt
curr[wt - weight['index']]