Linked List - IV

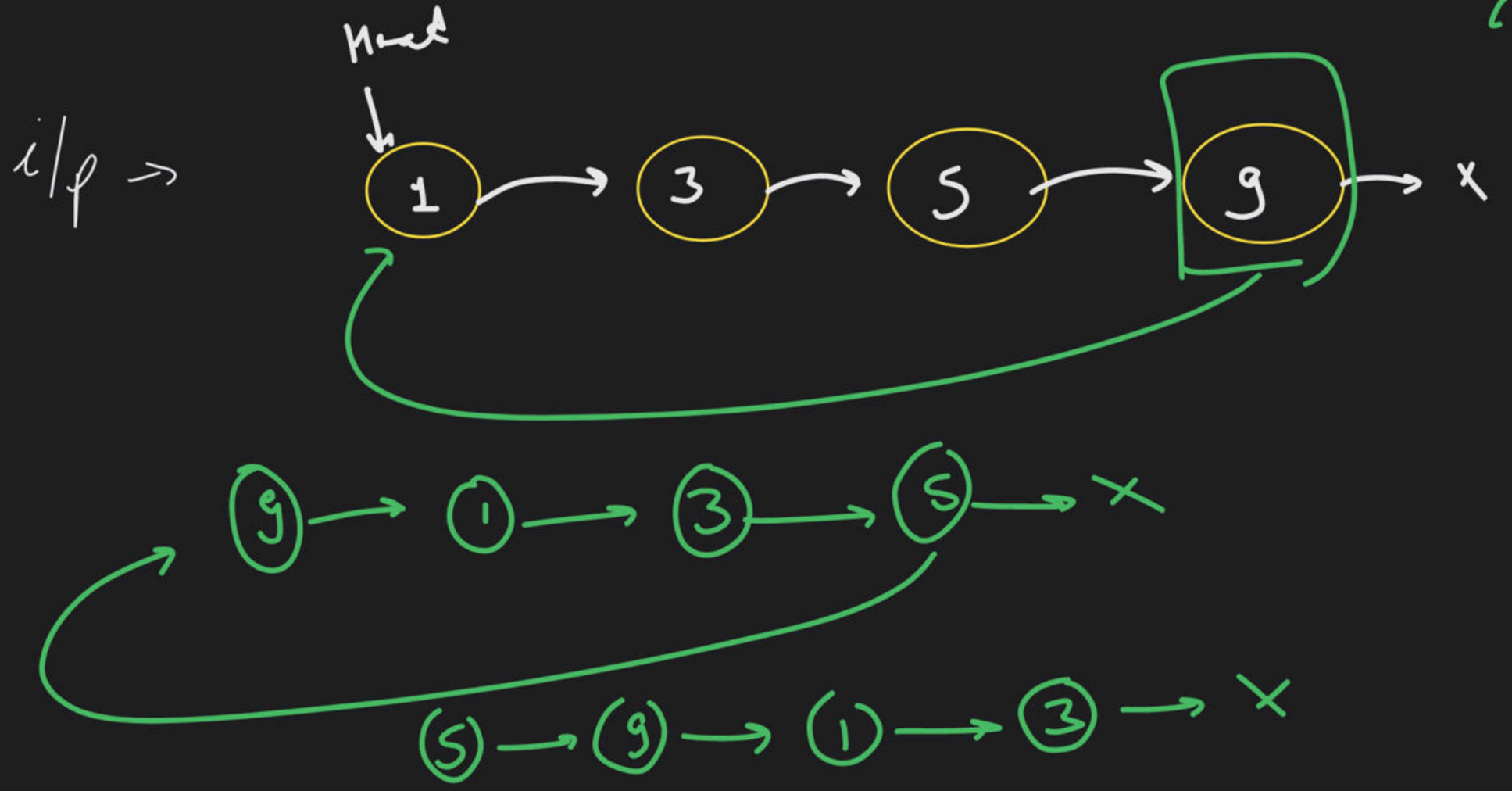
Foundation Course on Data Structures & Algorithm - Part I

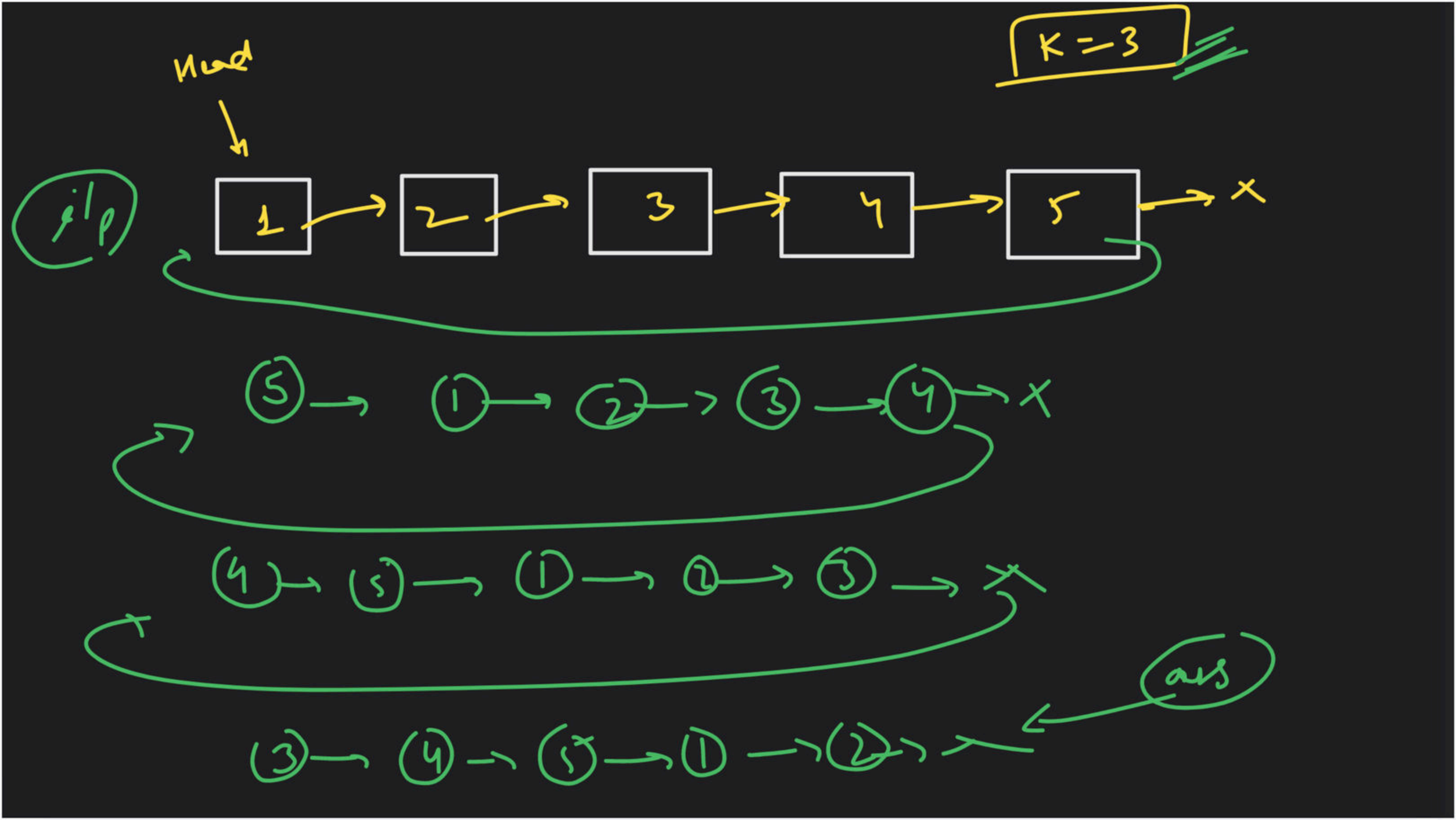
Love Babbar • Lesson 32 • June 23, 2022

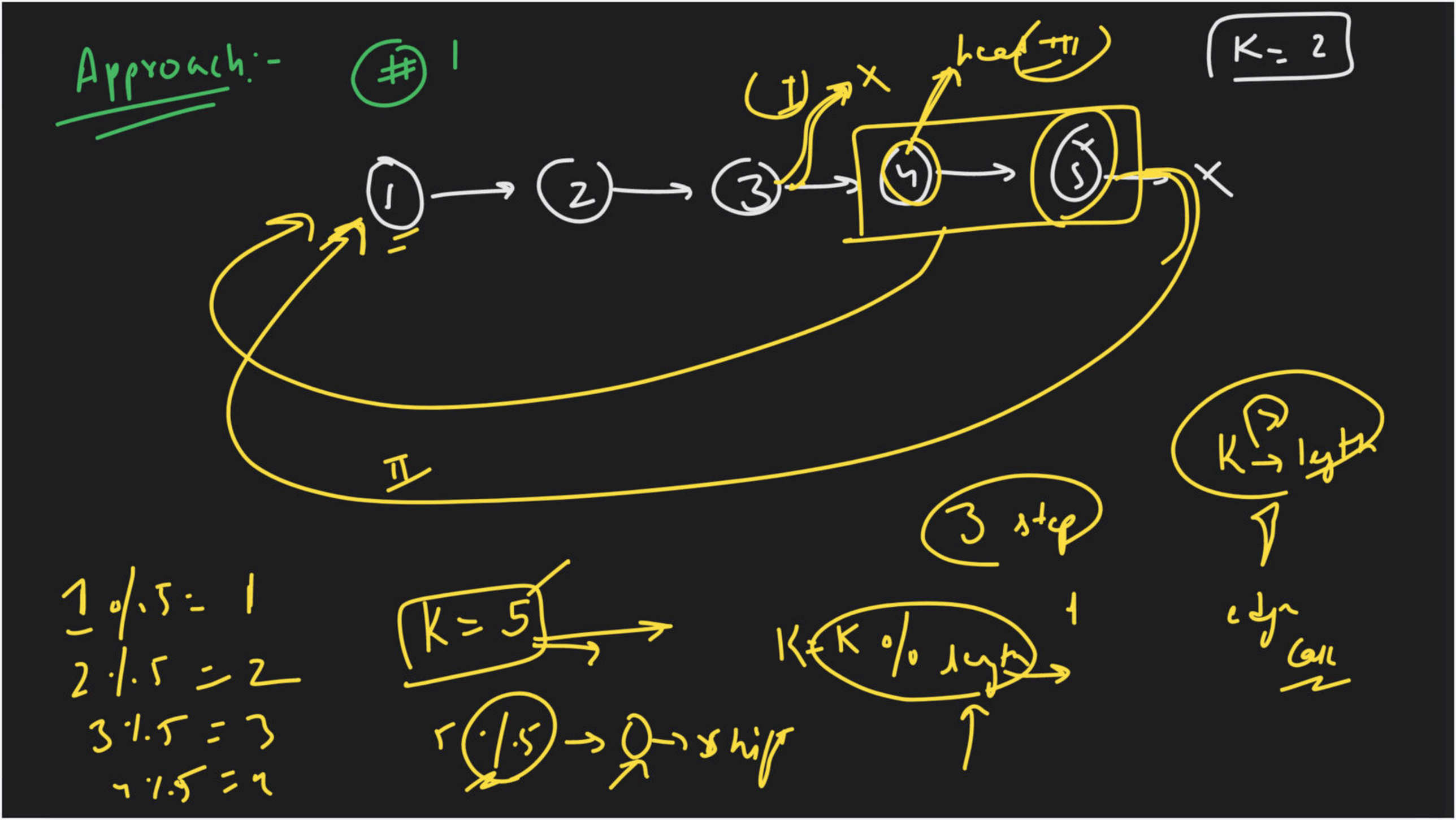
Rotate List :-

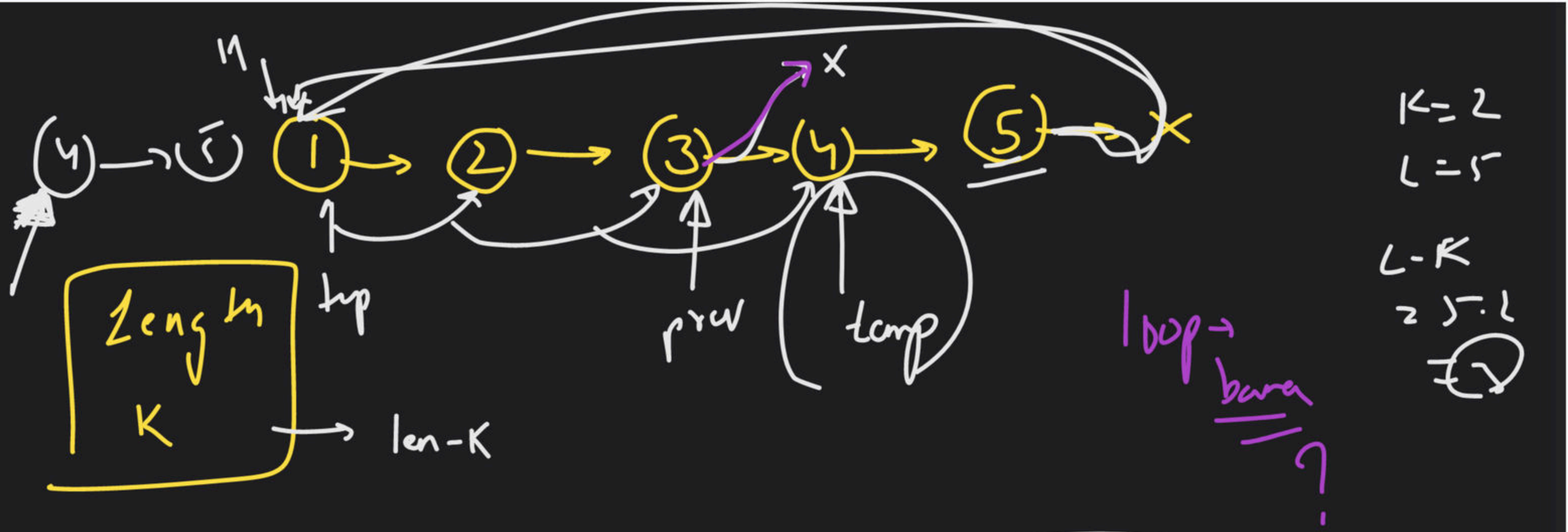
K = ~~2~~ 1

Question clear ?




Head
K=3
ptr
1
2
3
4
5
X
5
1
2
3
4
X
4
5
1
2
3
X
ans
3
4
5
1
2


Approach:-
#1
K = 2
1
2
3
4
5
X
X
II
1 % 5 = 1
2 % 5 = 2
3 % 5 = 3
4 % 5 = 4
K = 5
r % 5
shift
3 step
K = K % light
1
K → light
Call

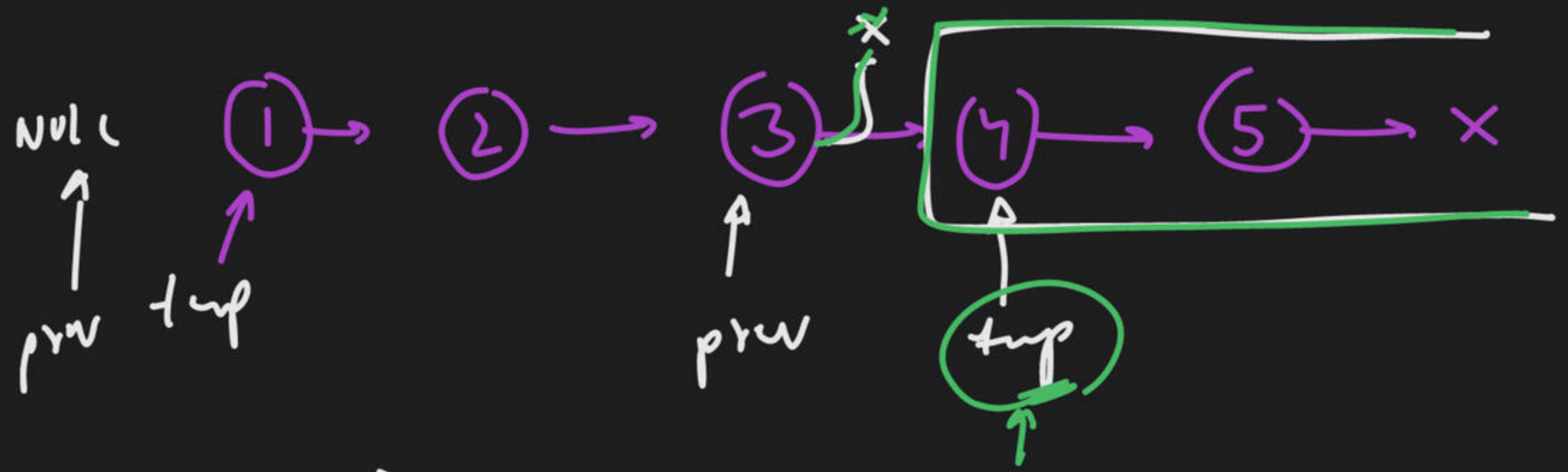


(1) (h-K) → traverse → prev, temp

(2) prev -> next = NULL;

(3) reach tail of K length LL -> tail -> next = head

(4) return temp



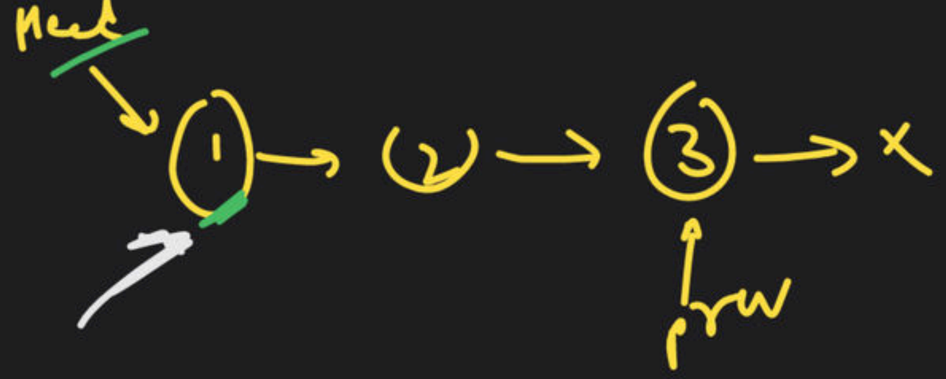
n = 5

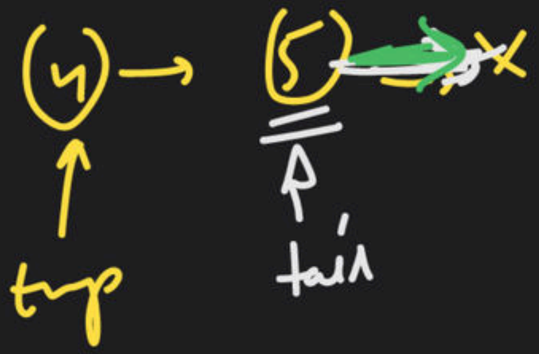
K = 2

n-K = 5-2 = 3

(1) (n-K) =

(2) prev->next = NULL





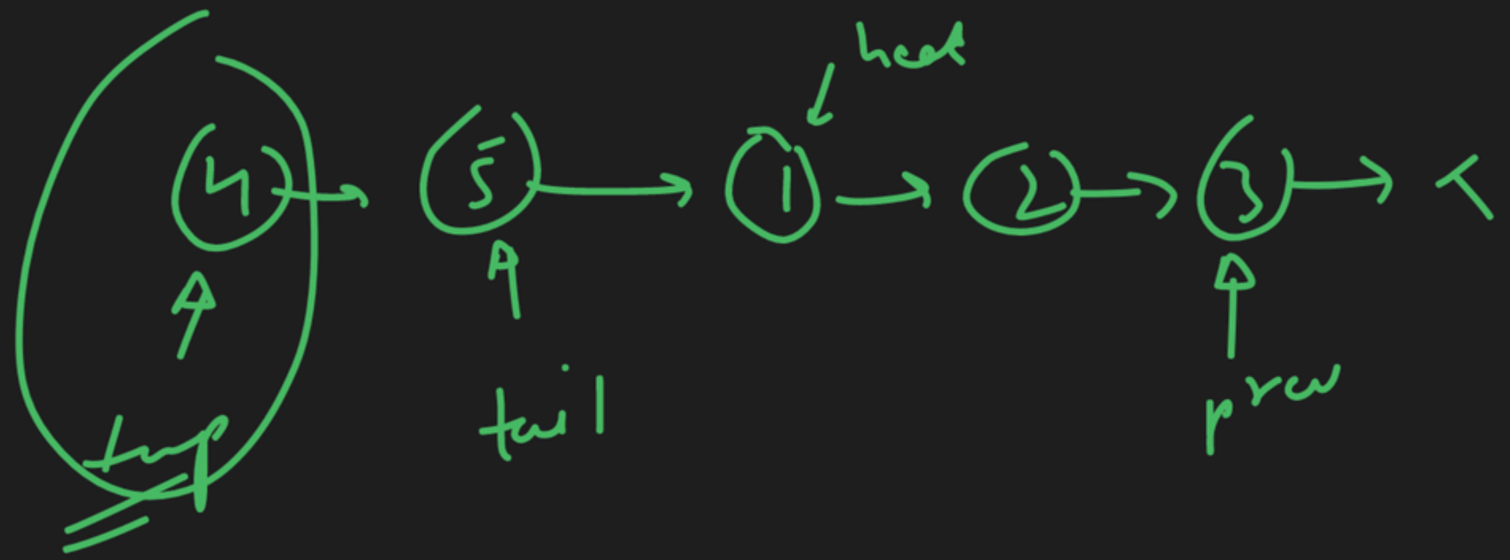
tail = temp

while (tail->next != NULL)

{ tail = tail->next }

tail->next = head

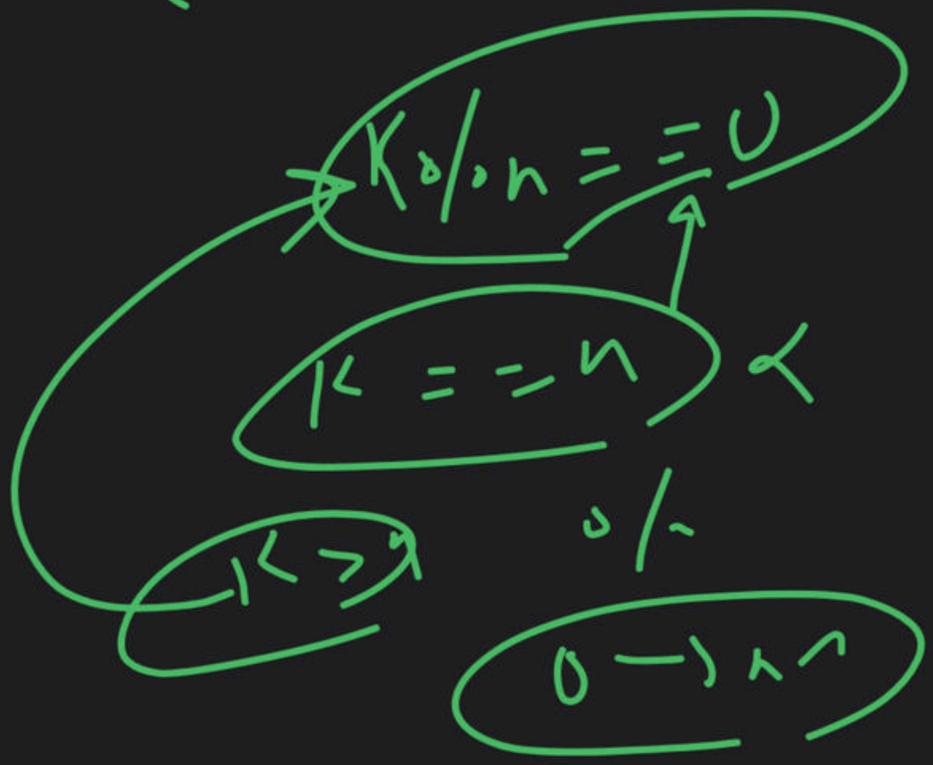
(3)

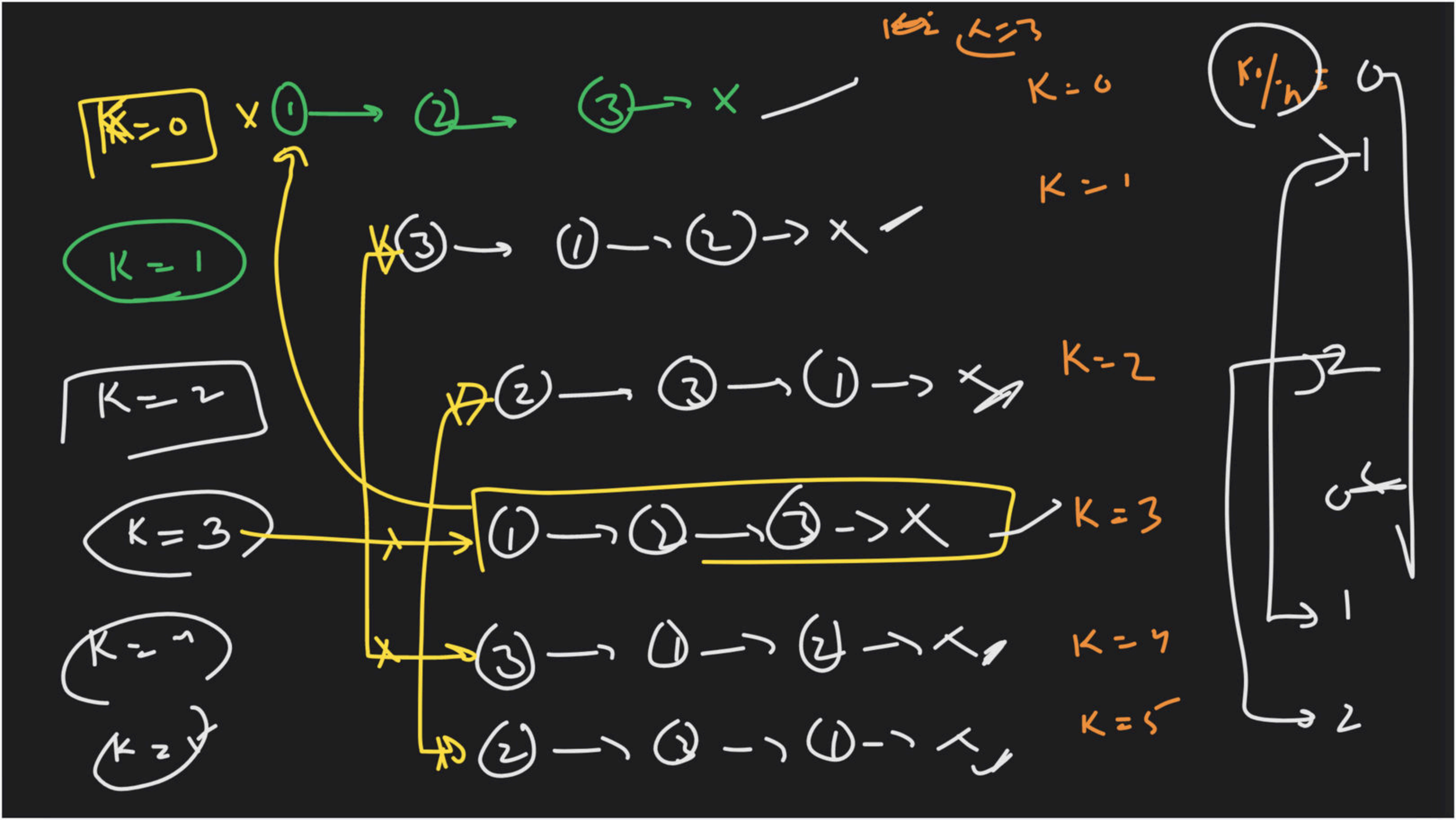


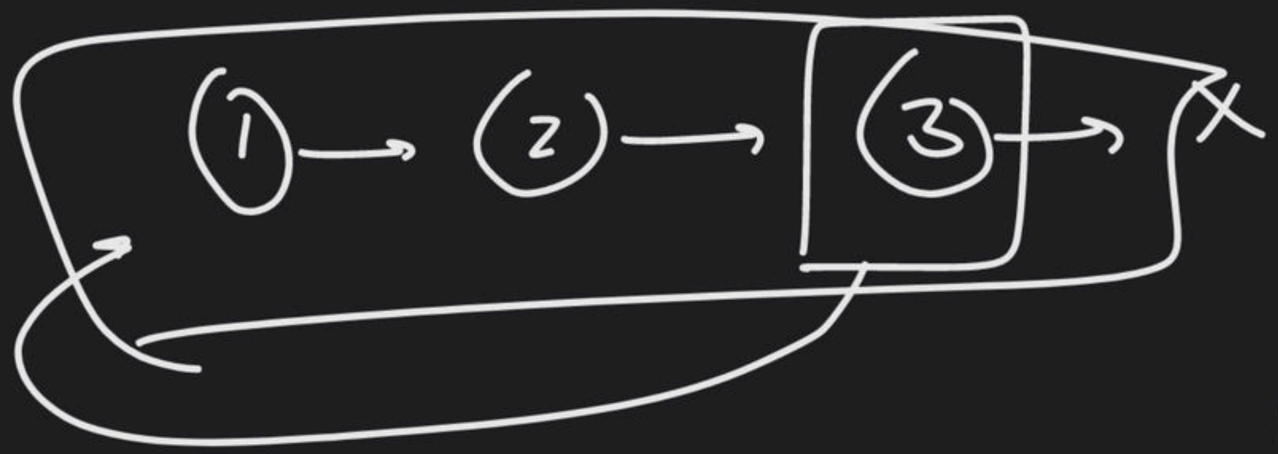
(IV) return temp

Edge Case:-

1. List Empty → Return ∝ (1) → X
2. List is 1 element → ∝
3. K = 0 → ∝



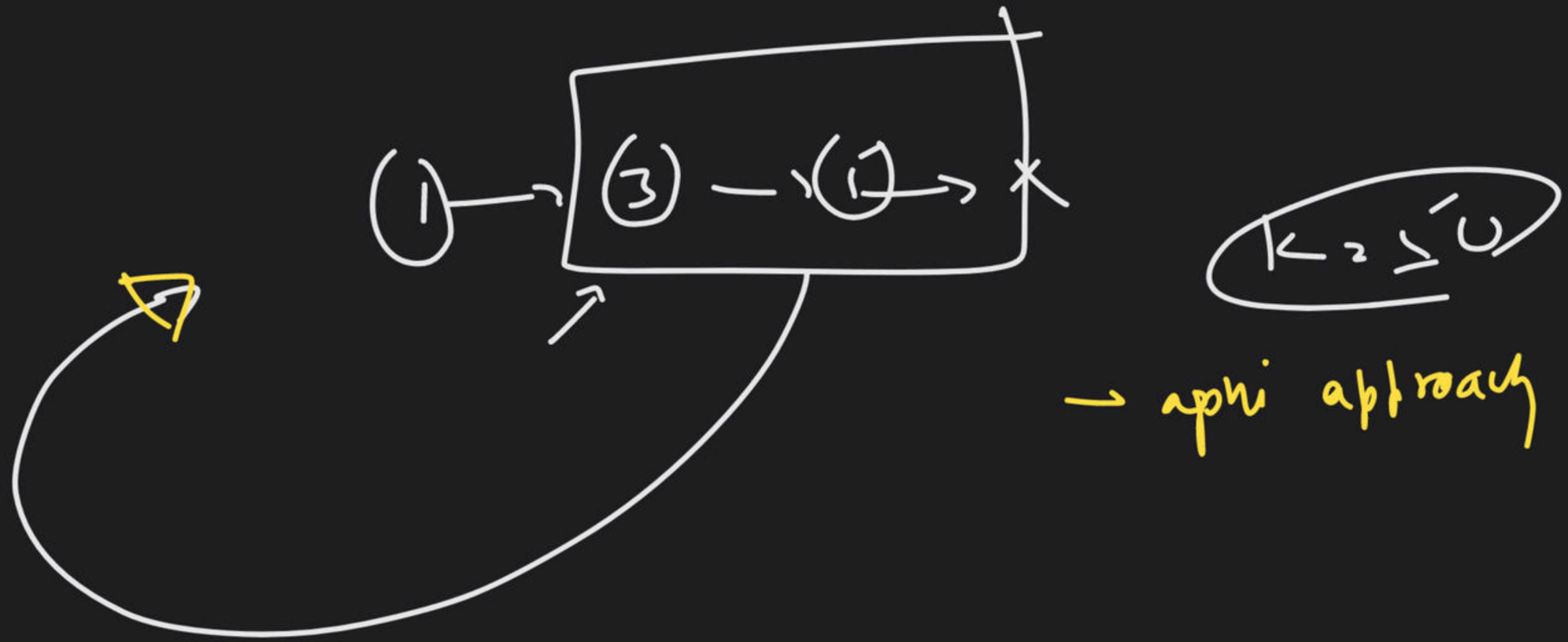

K=0
K=1
K=2
K=3
K=4
K=5
K=6
K=7
K=3
K%n = 0
1
2
1
2



K = 1

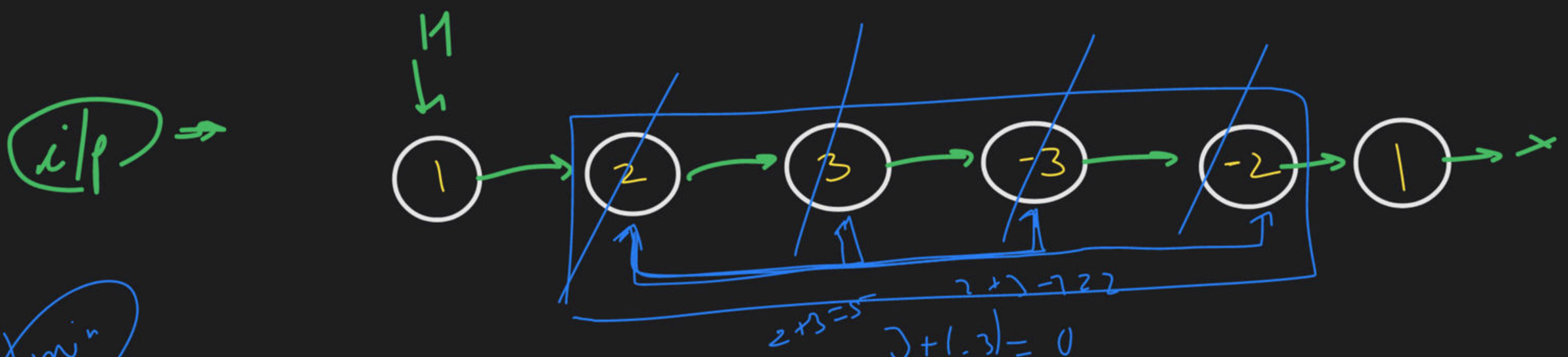
K = 50

Kuch bhi

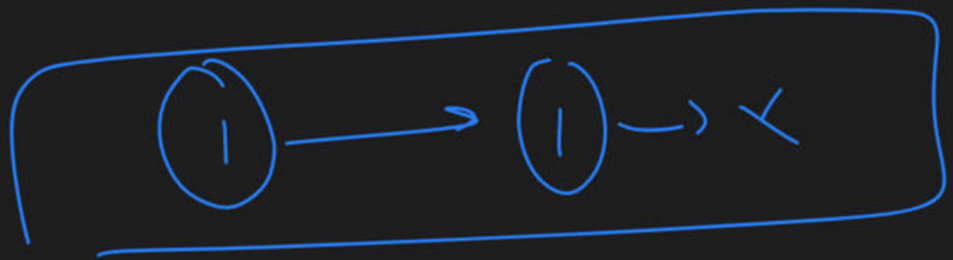


→ optimized approach

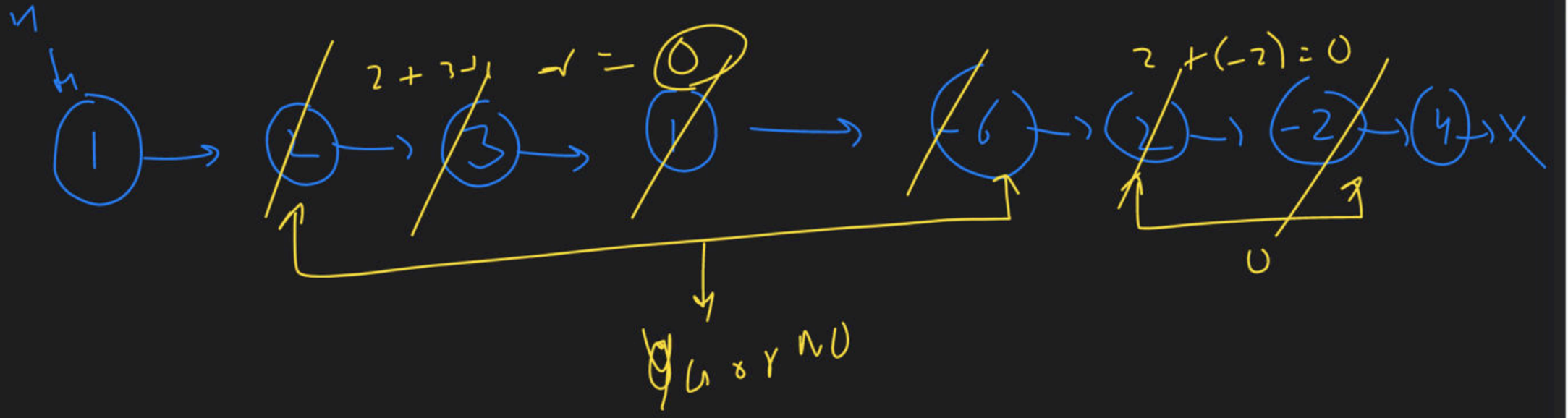
→ Remove Zero Sum Consecutive Nodes from LL

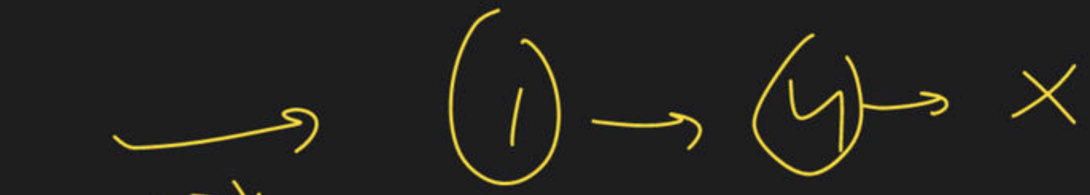


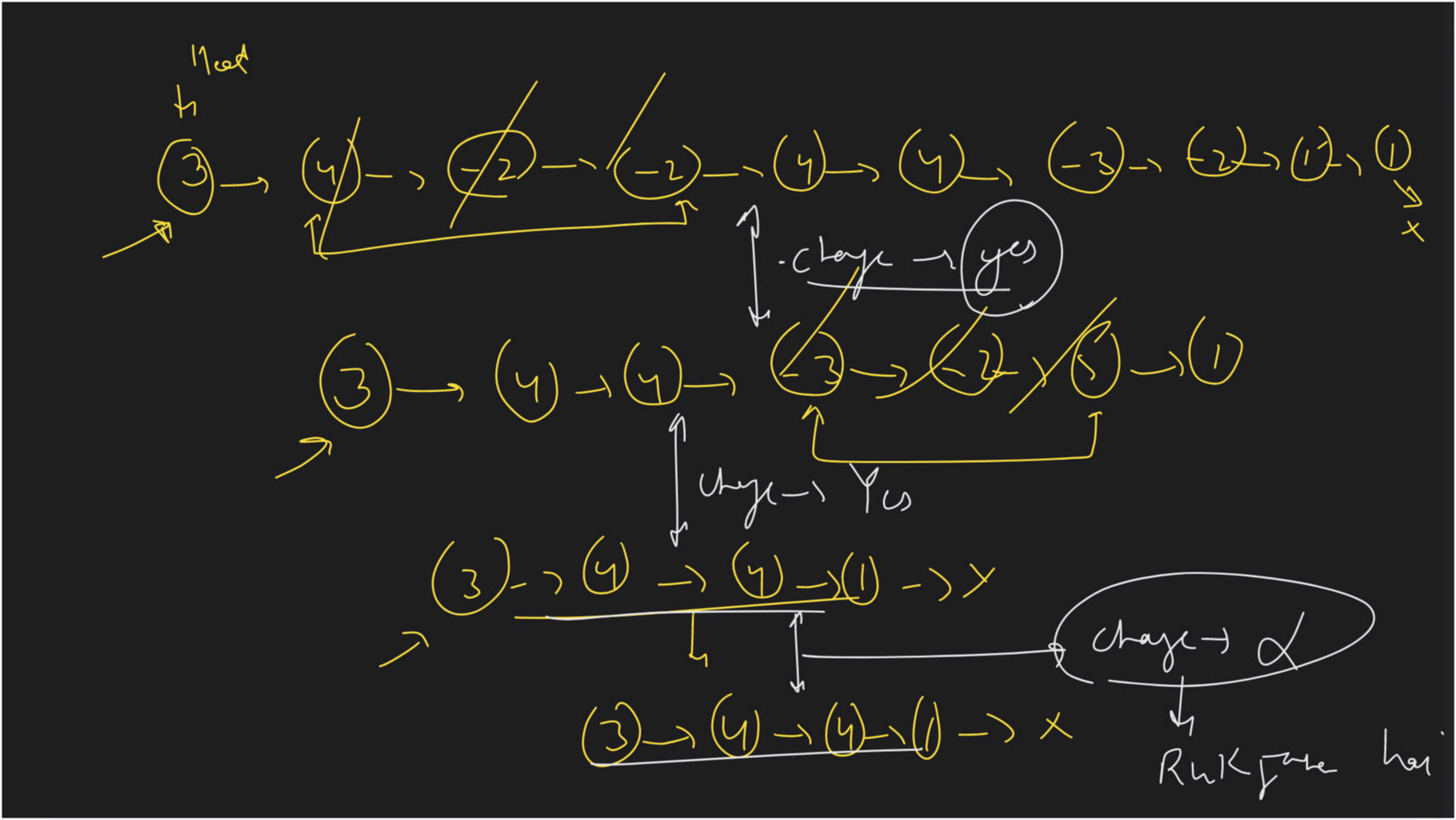
min
↓
Example
test Case

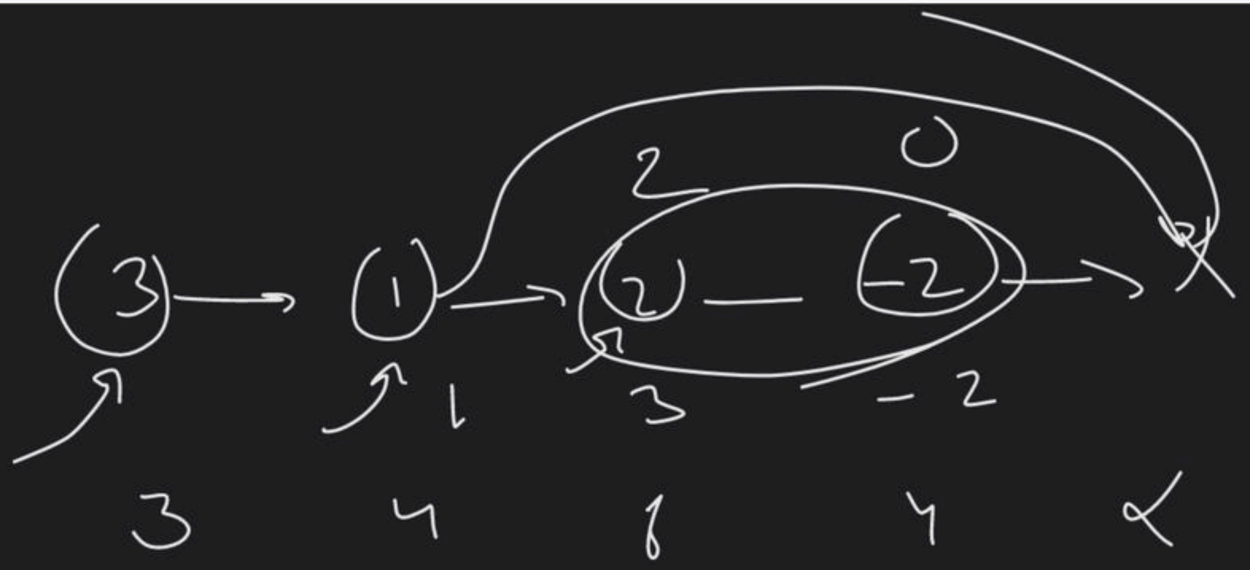


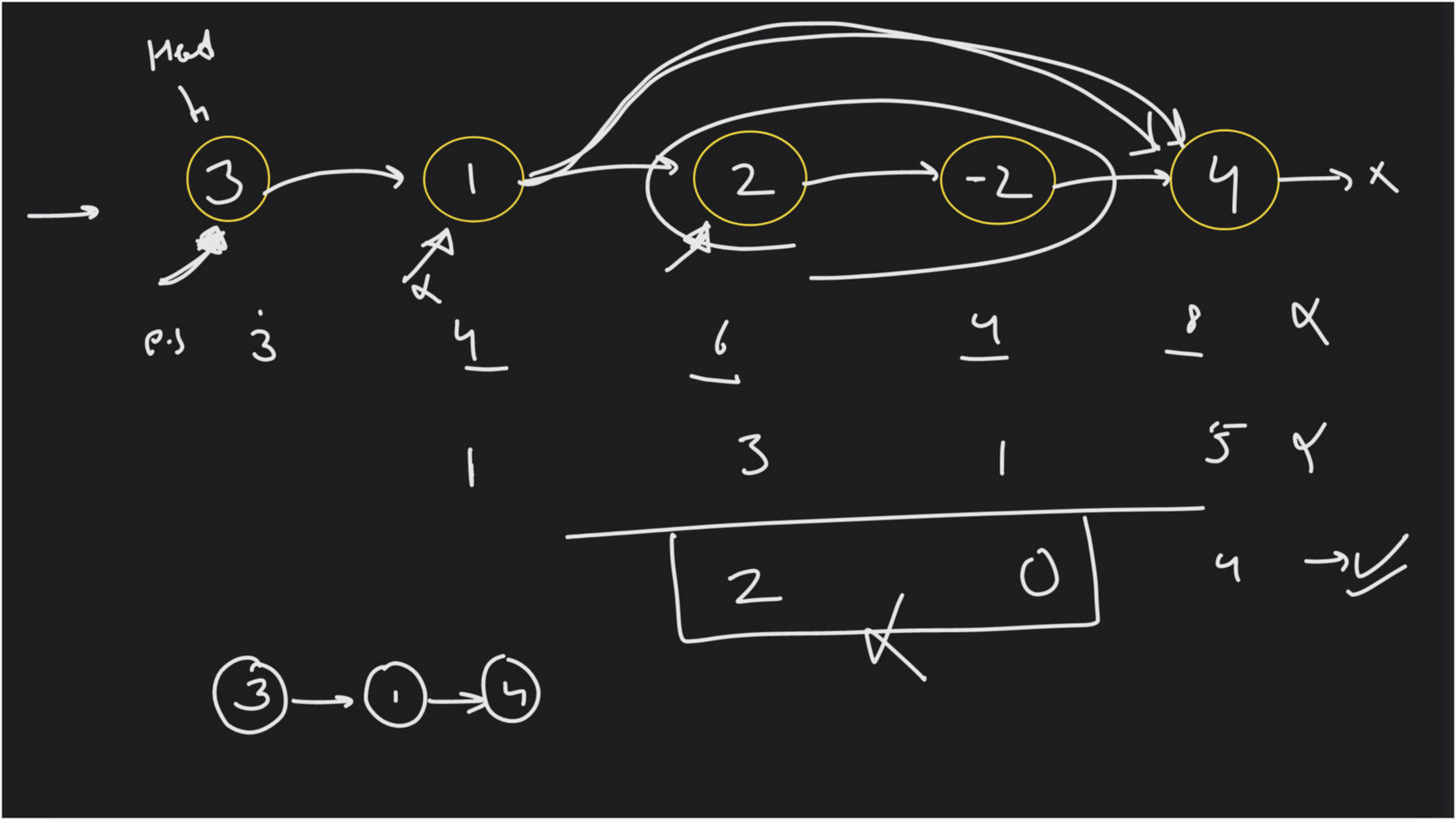
← answer

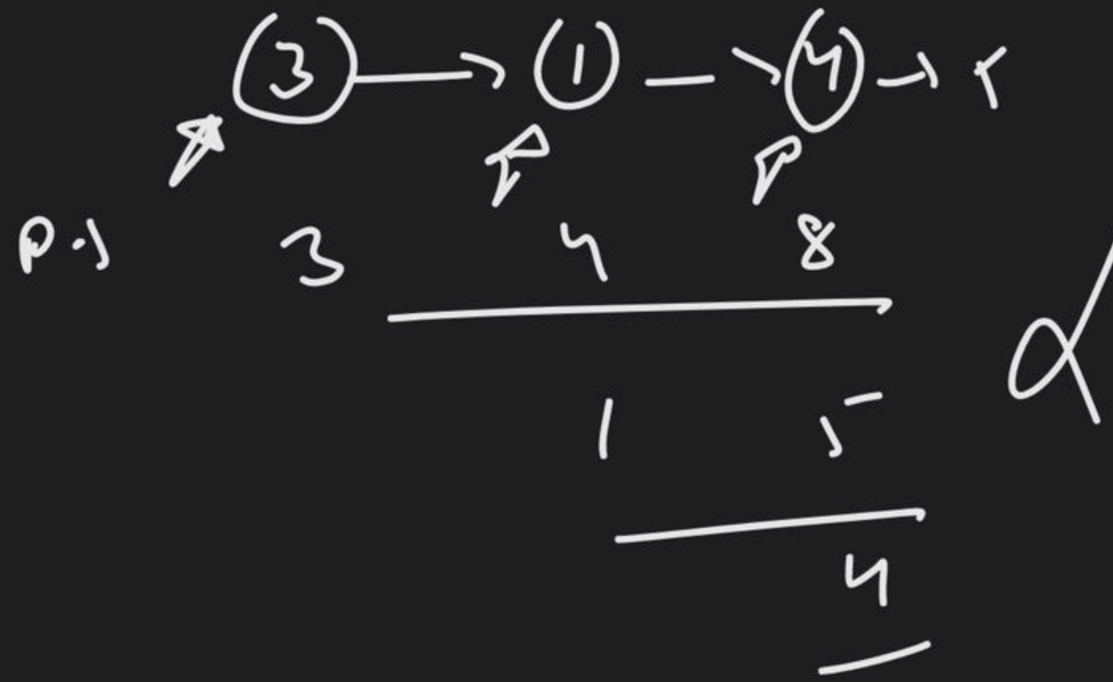

2 + 3 = 5
2 + (-2) = 0
0

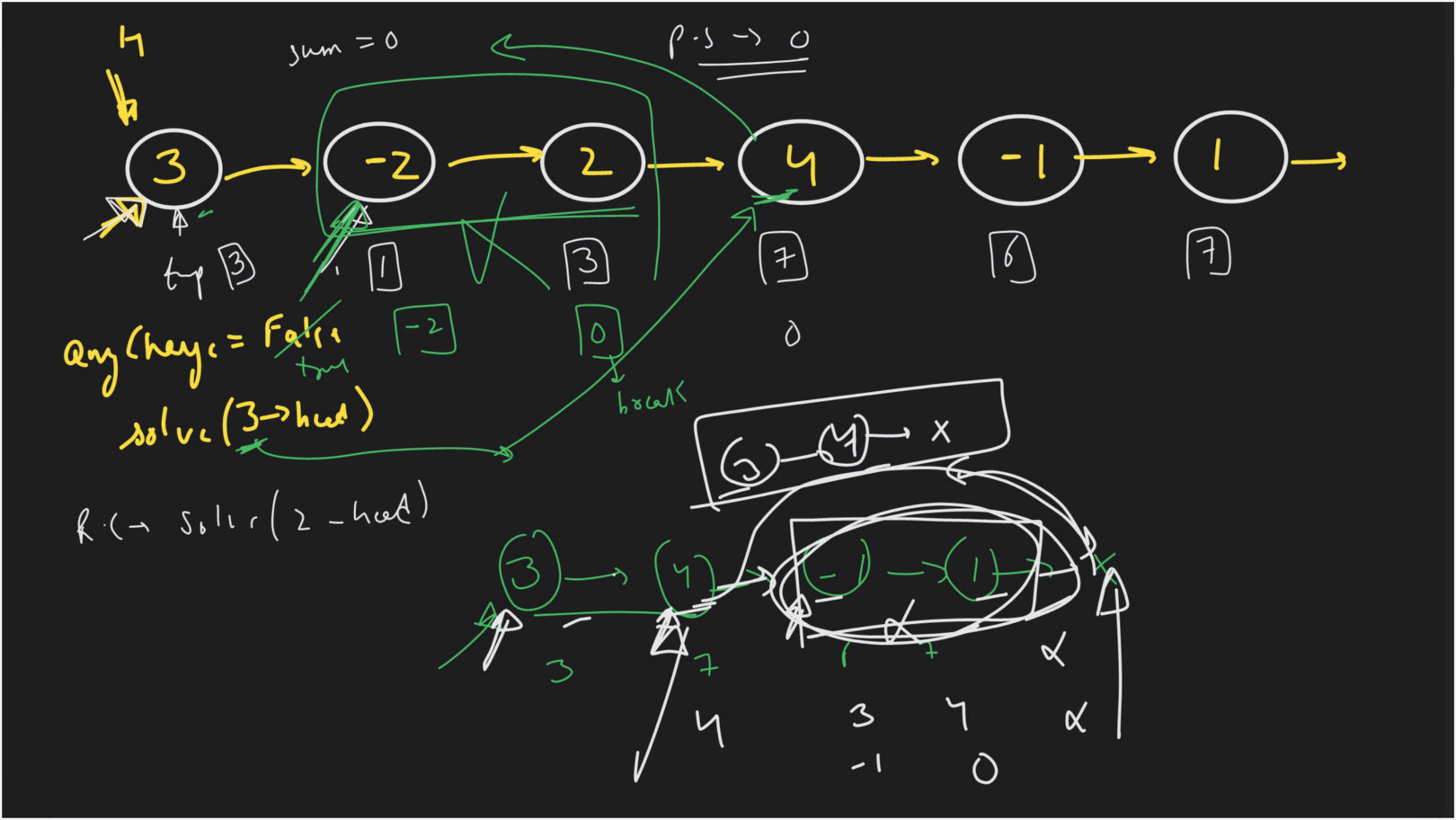


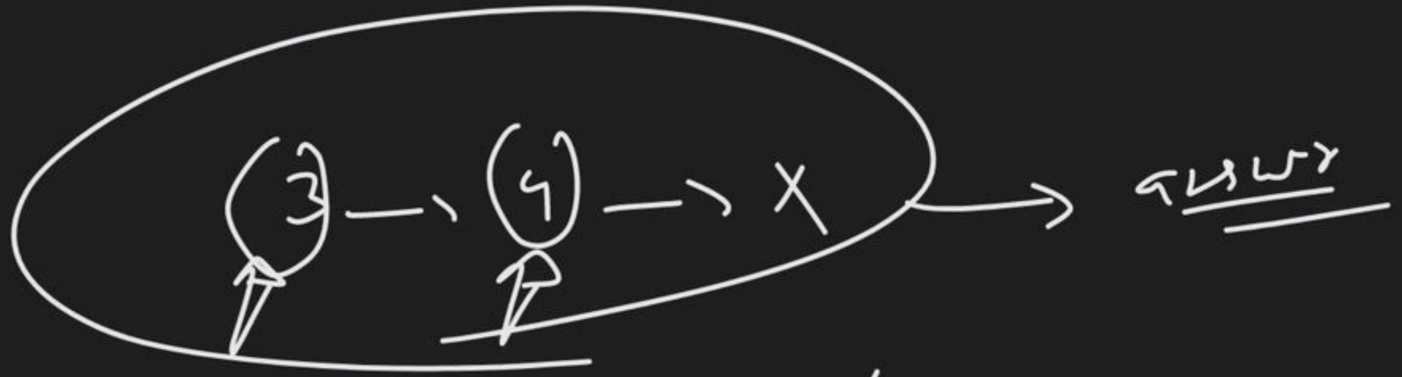

Head
3 → 4 → -2 → -2 → 4 → 4 → -3 → -2 → 1 → 1 → X
change → yes
3 → 4 → 4 → -3 → -2 → 1 → 1
change → Yes
3 → 4 → 4 → 1 → X
change → d
3 → 4 → 4 → 1 → X
Rukna hai


2
0
3
1
2
-2
1
3
-2
3
4
1
4
2


Mod
h
3
1
2
-2
4
x
p.s
3
4
6
4
8
4
1
3
1
5
4
2
0
4
3
1
4


3
1
4
P1
3
4
8
1
3
4

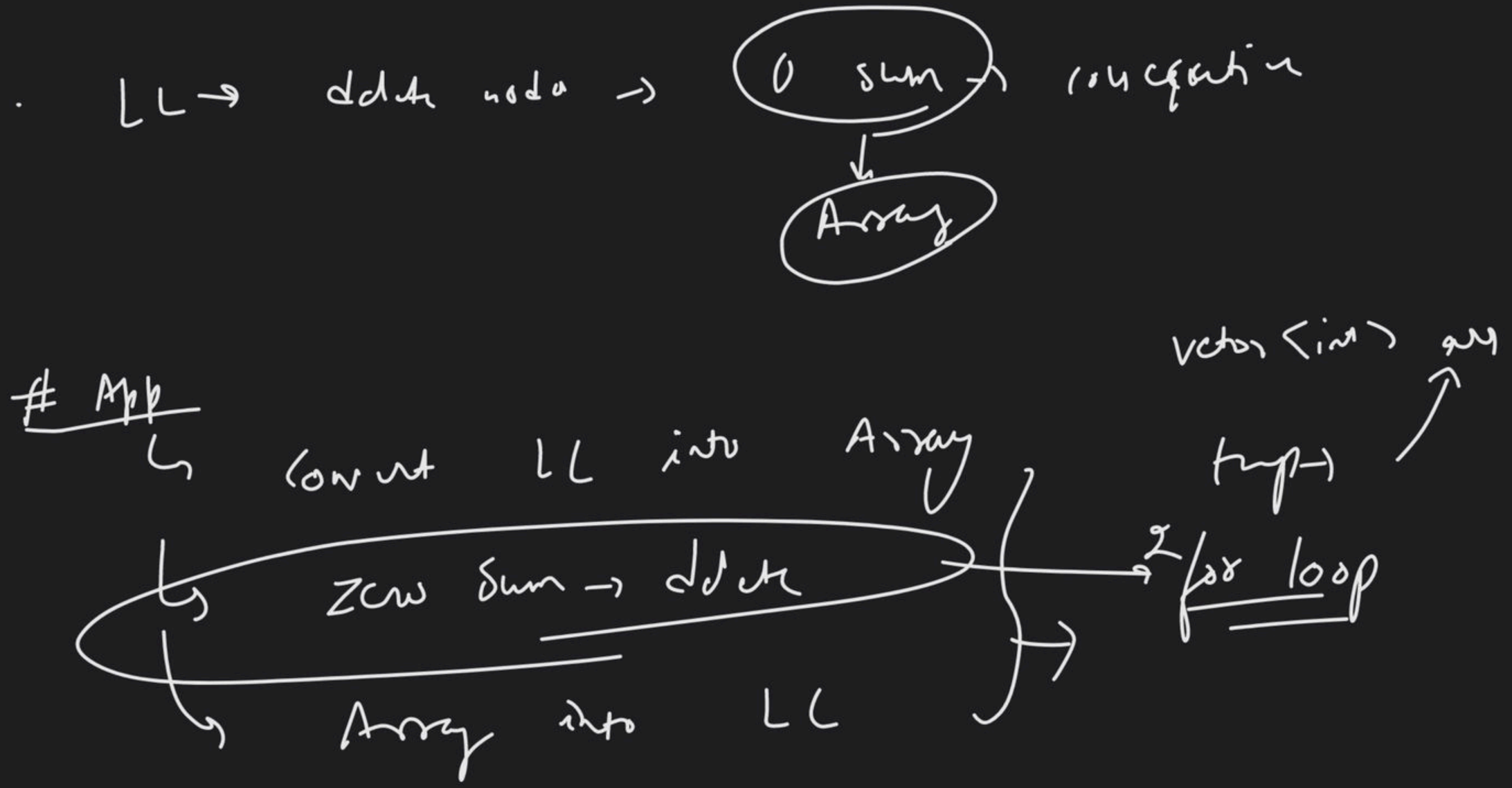

sum = 0
p->s -> 0
3
-2
2
4
-1
1
temp 3
1
3
7
6
7
-2
0
0
query(key = False
true
solve(3->next)
local
return solve(2->next)
3
4
x
3
4
-1
1
3
4
4
3
4
0

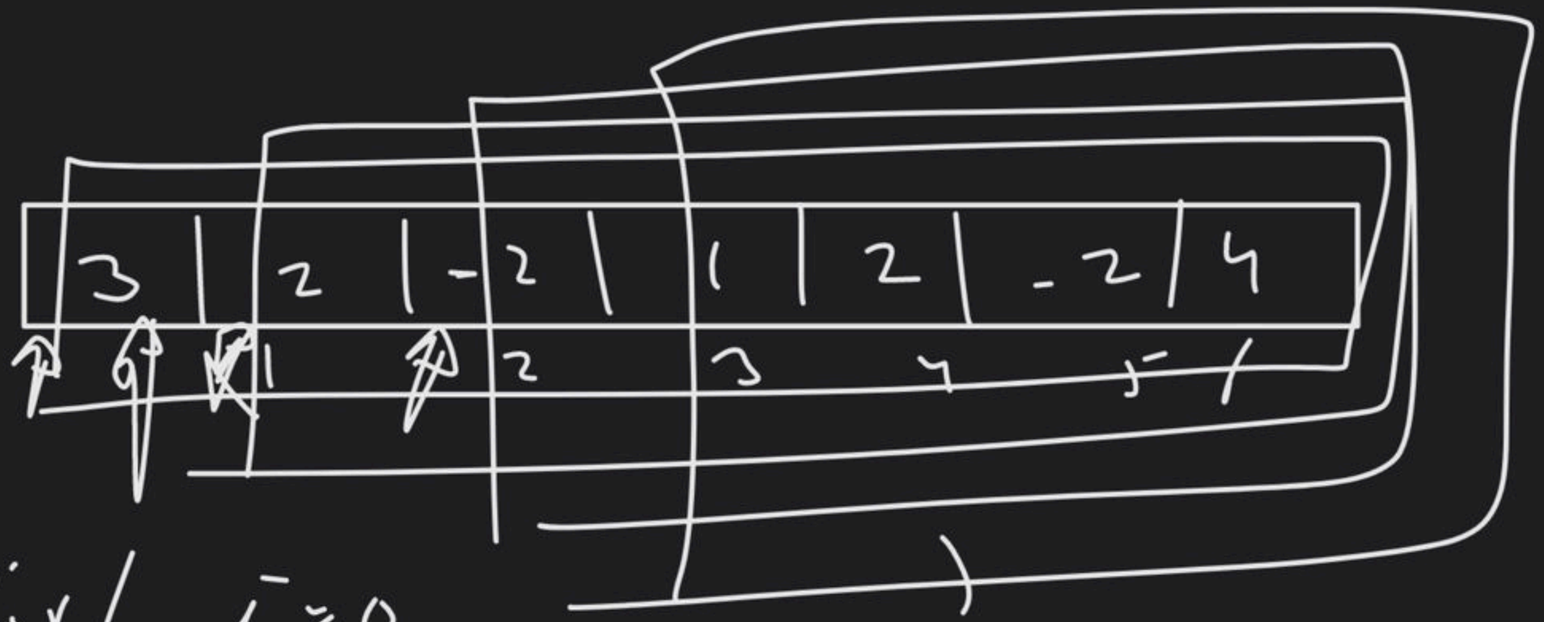


1 3 4 α

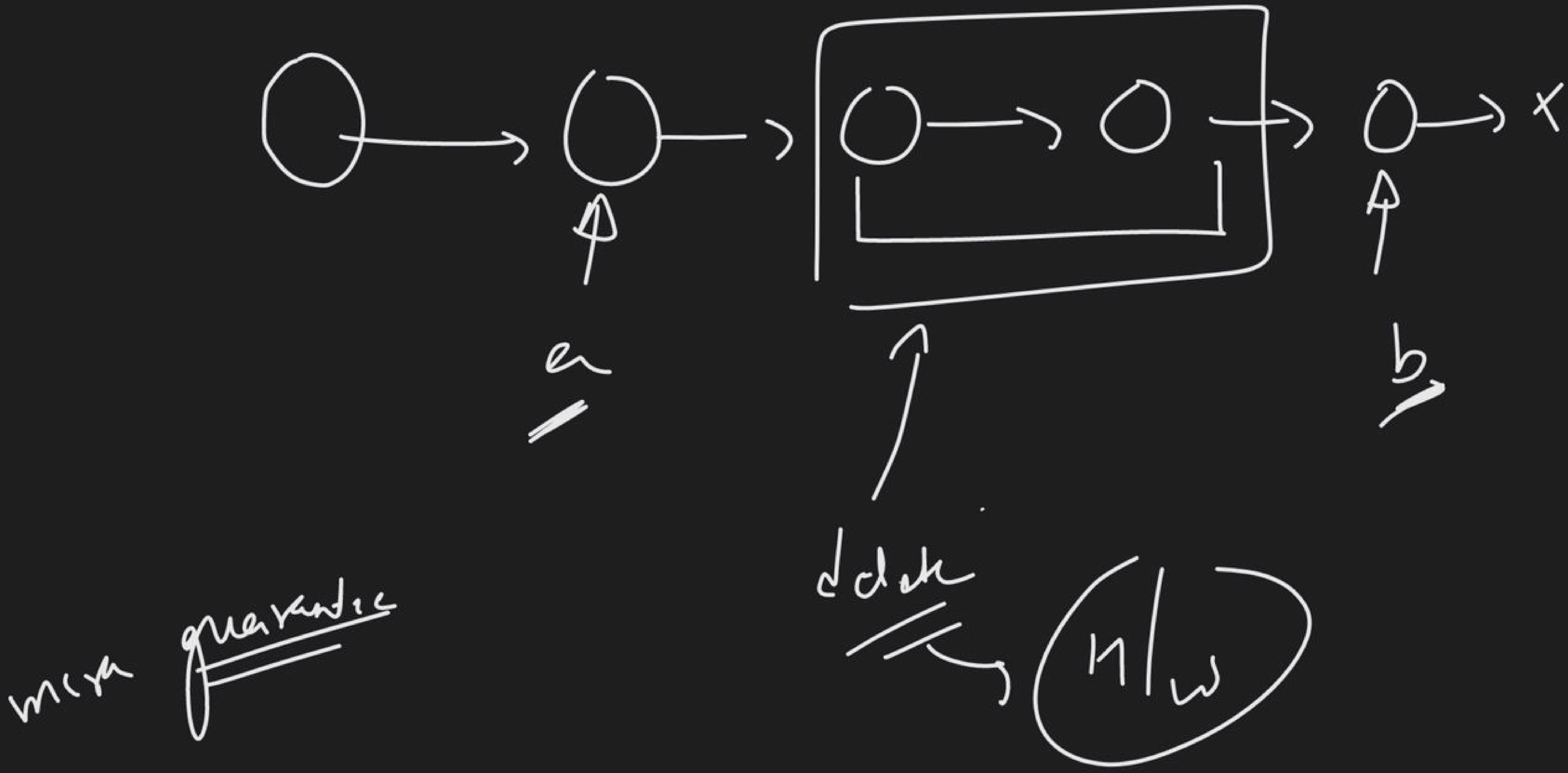
4 α

Recursion

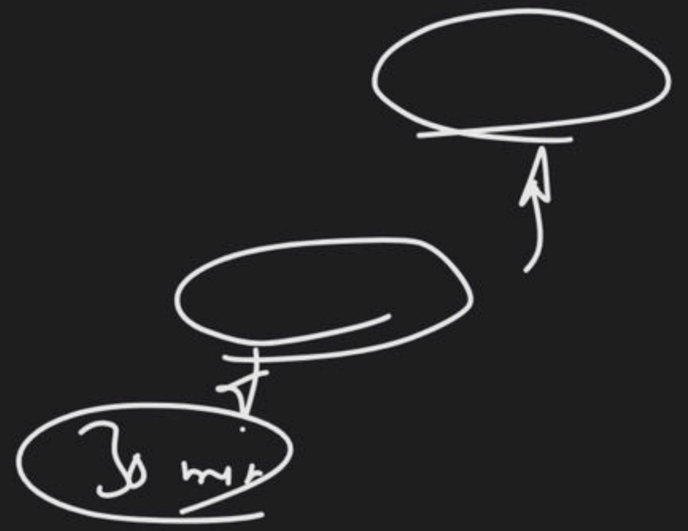

LL → delete node →
0 sum
reduction
Array
Approach
Convert LL into Array
vector<int> arr
temp
2 for loop
Zero Sum → delete
Array into LL

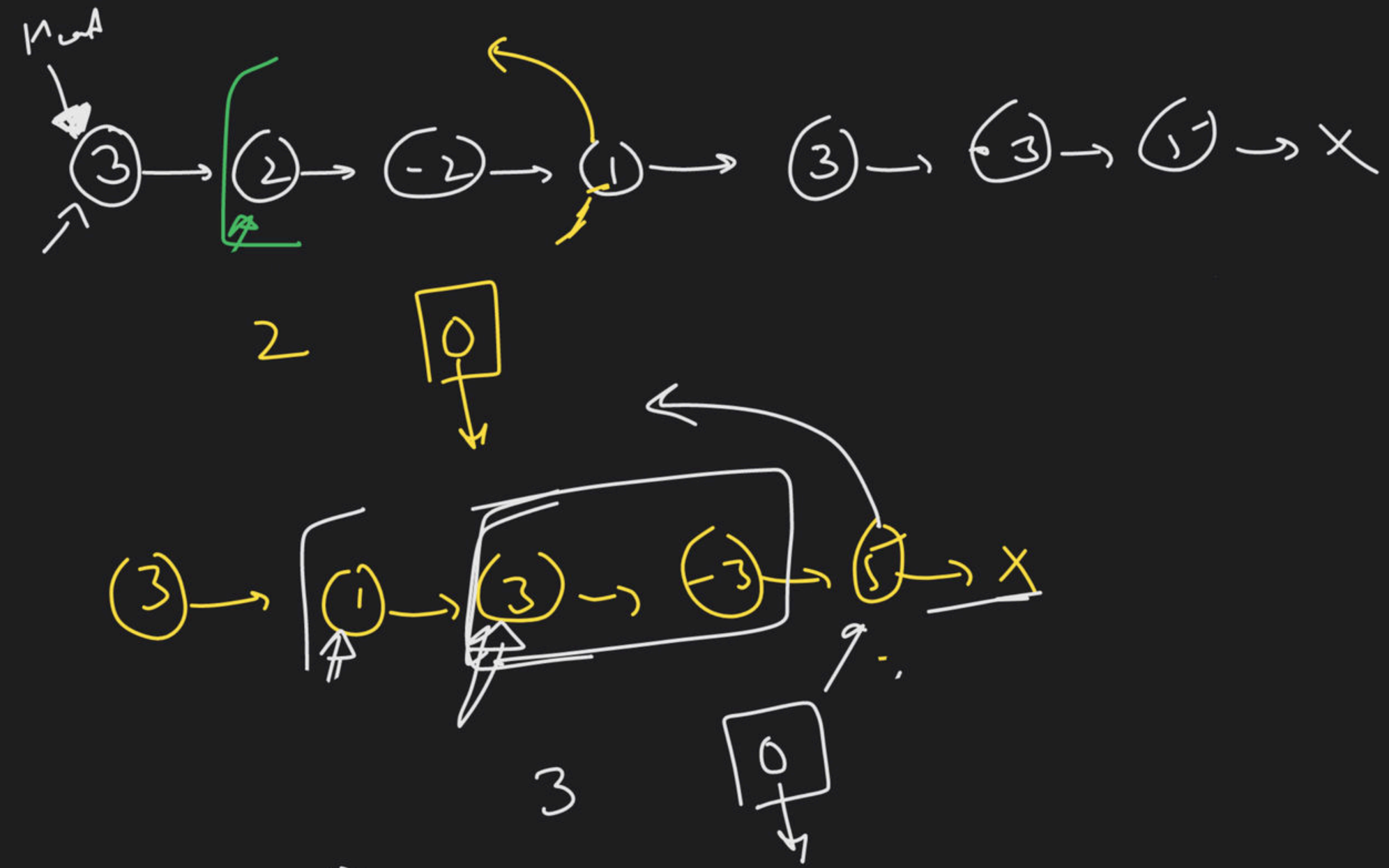


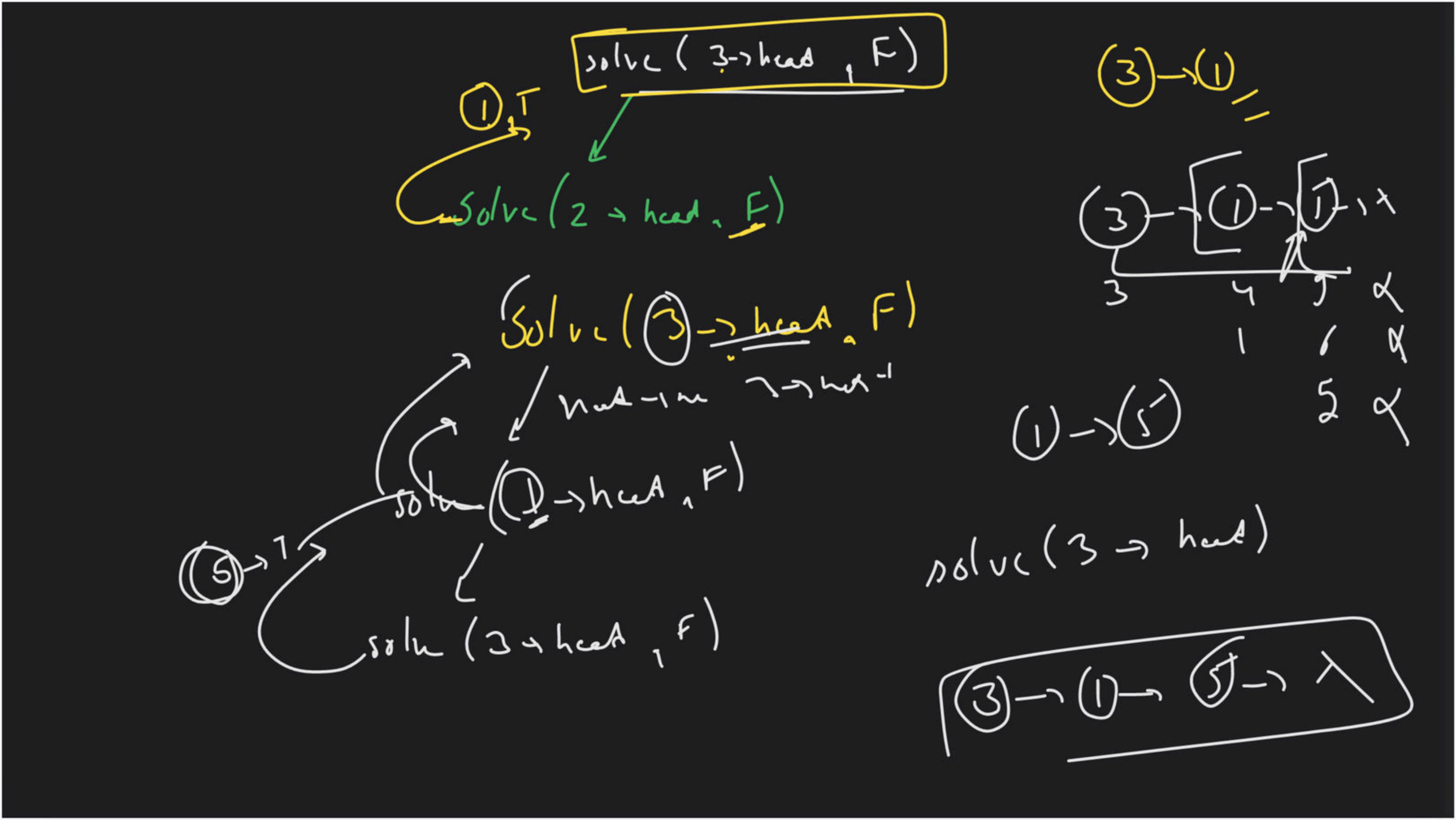

```
for( i=0
{
for (int j=i
}
}
```

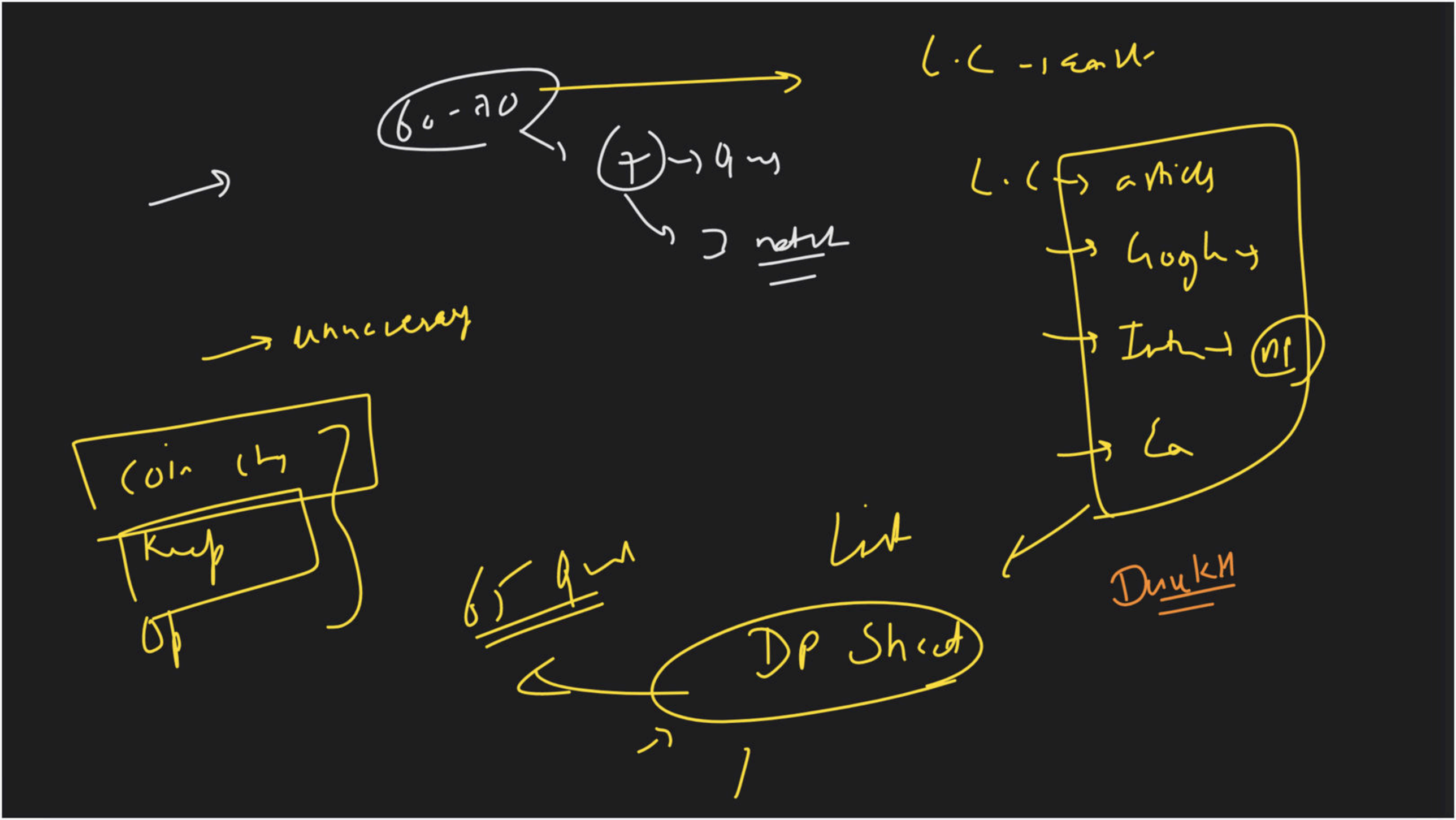

x
a
data
b
H/W
mera guarantee

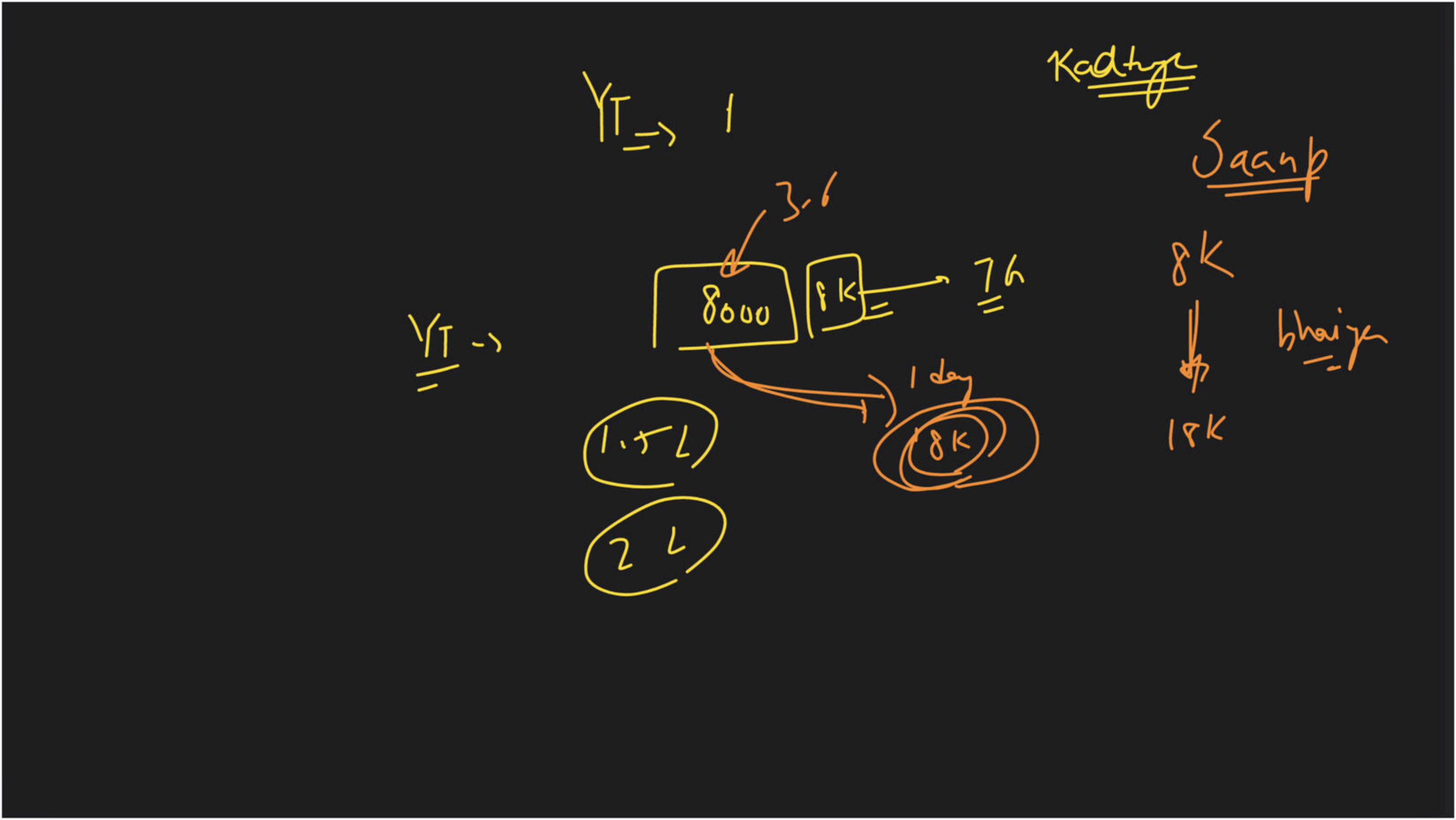
issue →?

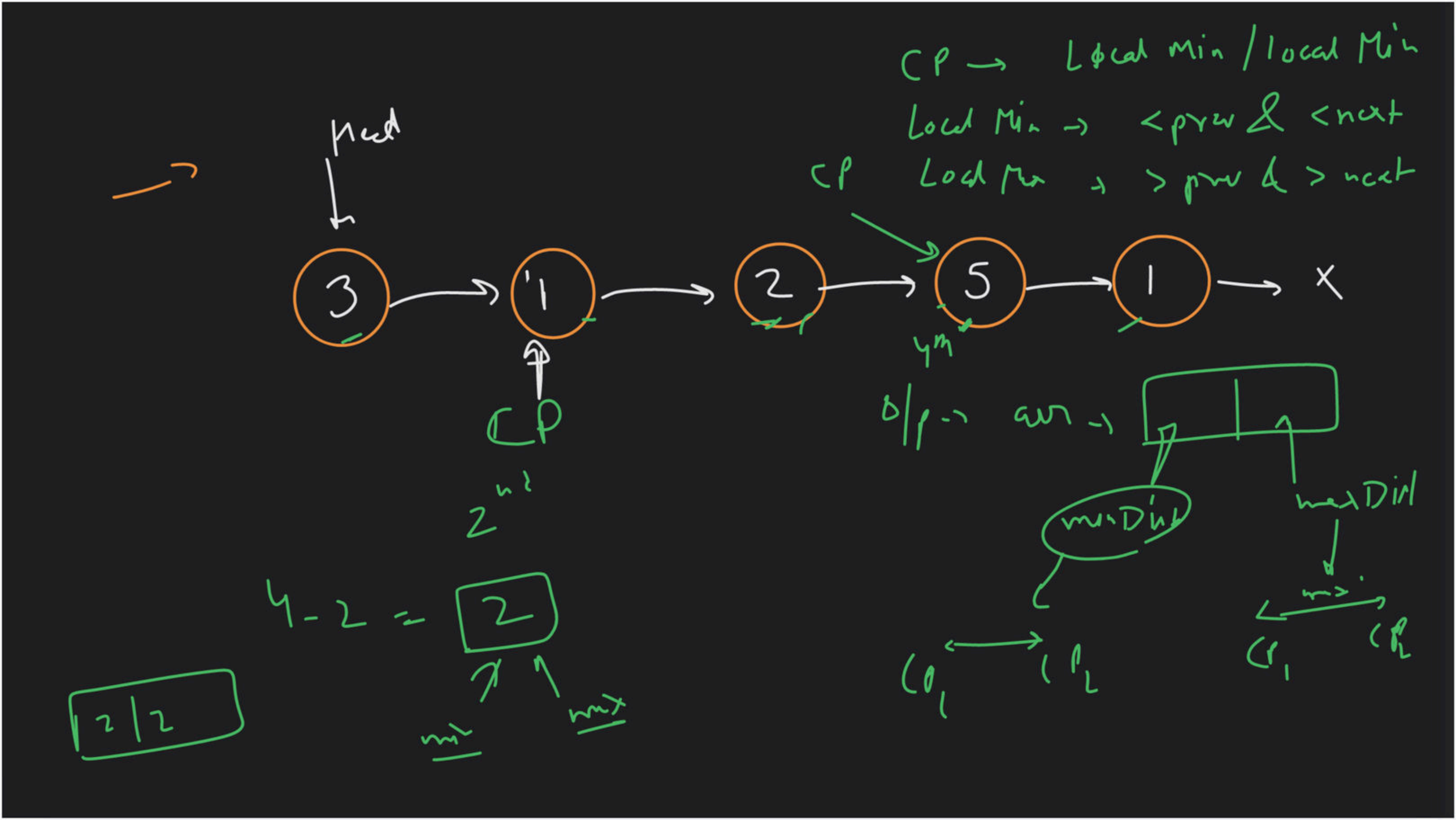


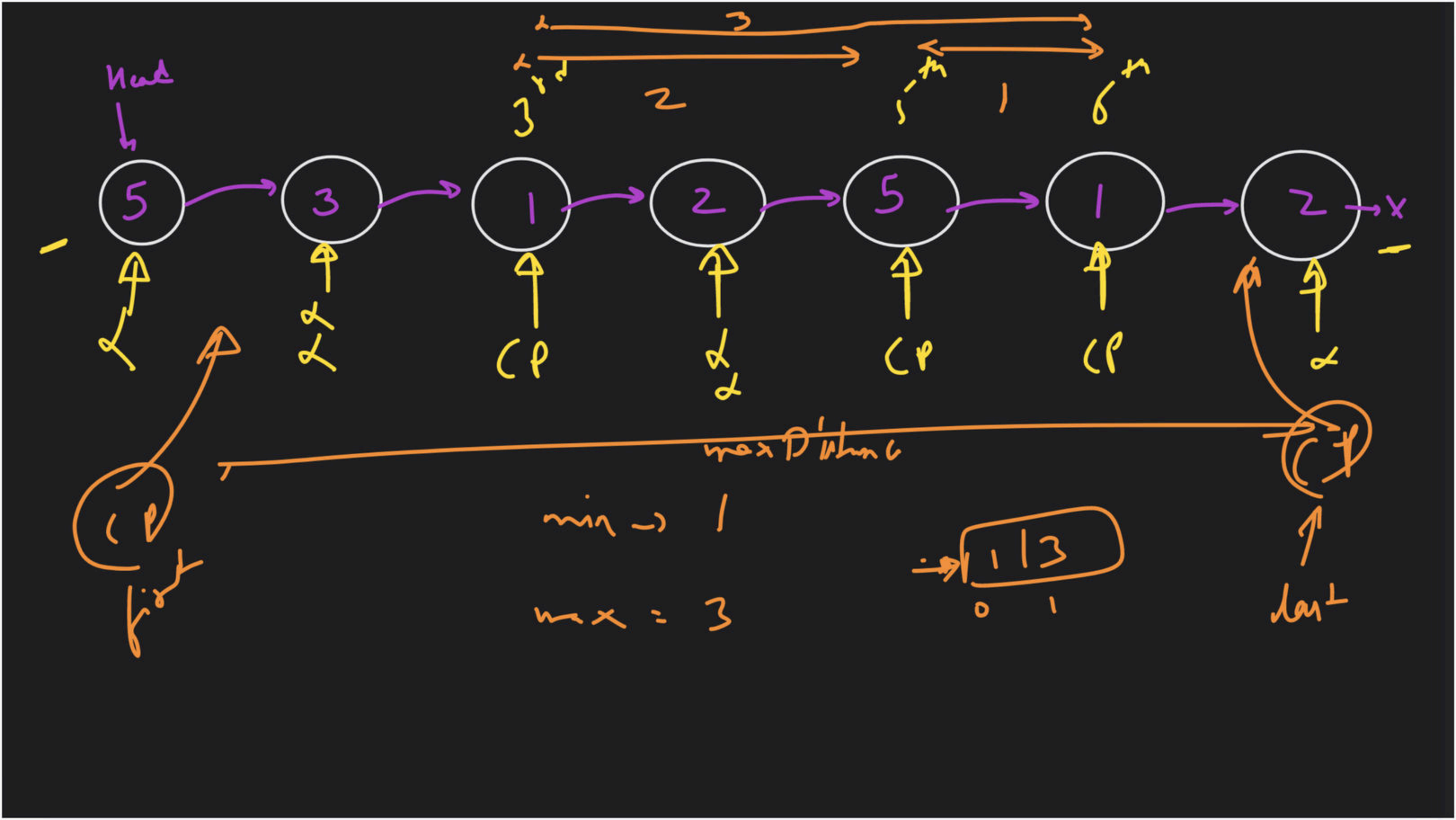

3
2
-2
1
3
-3
1
X
2
0
3
1
3
-3
5
X
3
0
3
1
5
X


solve (3->head , F)
1, T
Solve (2 -> head , F)
Solve (3 -> head , F)
head->next
3->head->next
solve (1 -> head , F)
5 -> T
solve (3 -> head , F)
3 -> 1
3
1
1
3
4
7
1
1
5
2
1 -> 5
solve (3 -> head)
3 -> 1 -> 5 -> ^


Dp Sheet


Kadhya
YT -> 1
Saanp
3.6
8000
1K
76
8K
YT ->
1 day
8K
bhaiya
1.5L
2L
18K


CP → Local Min / local Max
Local Min → < prev & < next
CP Local Max → > prev & > next
head
3
1
2
5
1
X
CP
CP
2 nd
4 - 2 = 2
mid
mx
2 | 2
O/P → arr →
maxDist
maxDist
mx
CP1
CP2
CP1
CP2


Head
5
3
1
2
5
1
2
x
3
2
1
maxDistance
min -> 1
max = 3
first
last
1
3
0
1

① head = NULL || head->next = NULL || head->next->next = NULL

return [-1, -1];

② minDistance = INT_MAX;
maxDistance = INT_MIN

total cp

③ int totalCp = 0 → < 2 → [-1, -1]

④ int cp = -1;

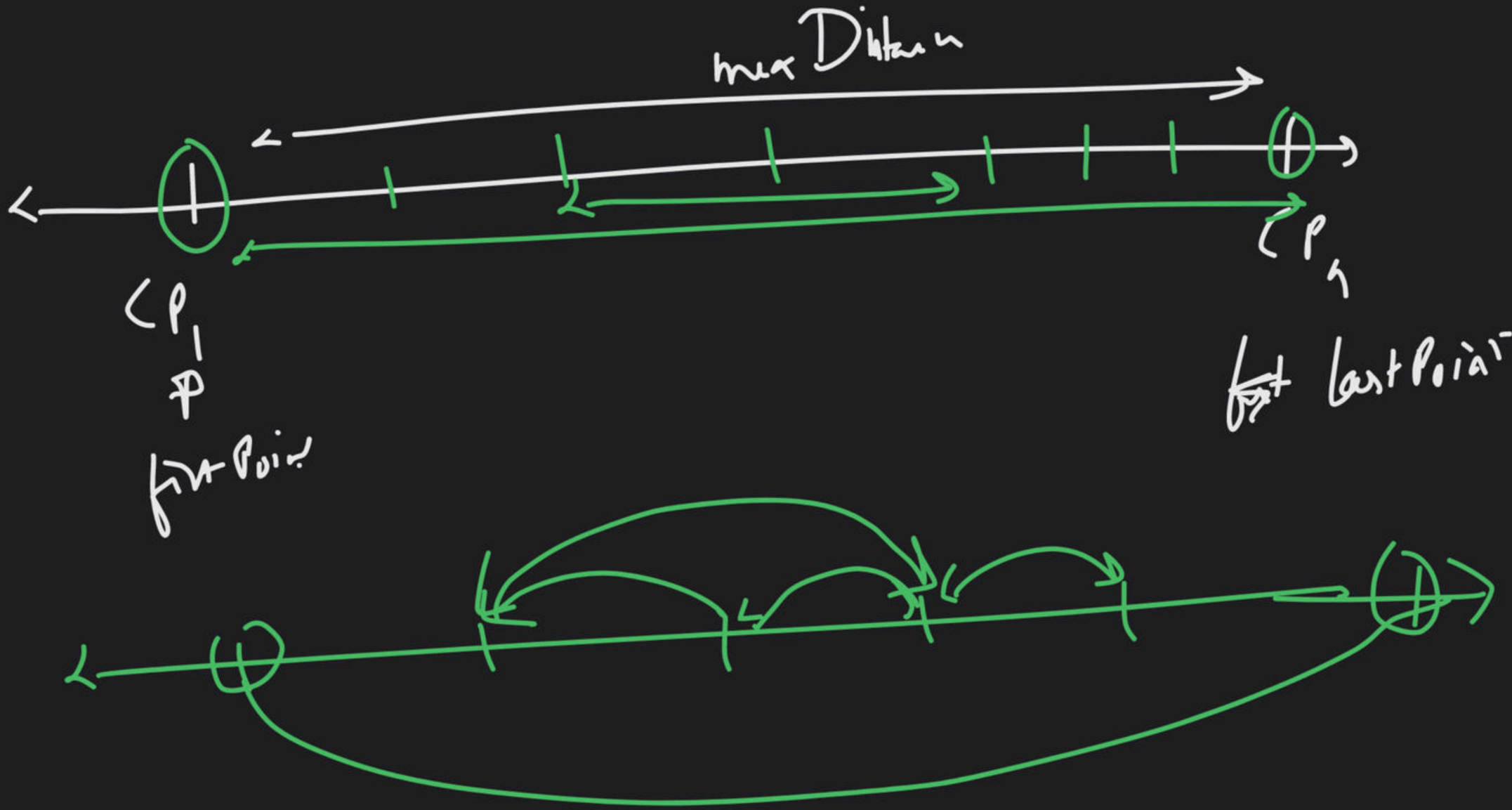
1 < p →

int firstPoint = INT_MAX;
lastPoint = INT_MIN;

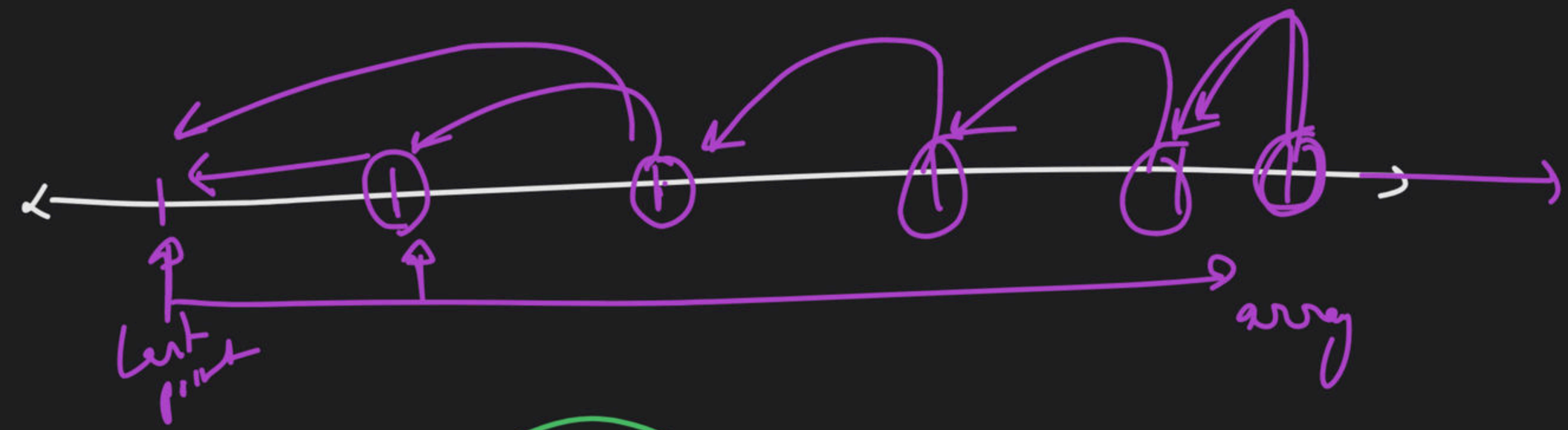
→ maximum

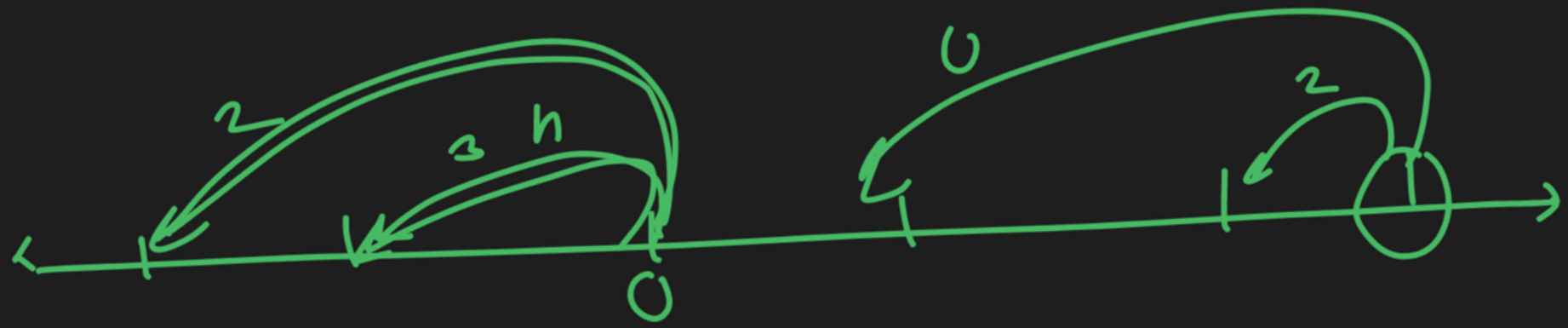
(5) int index = 1; → (store → location

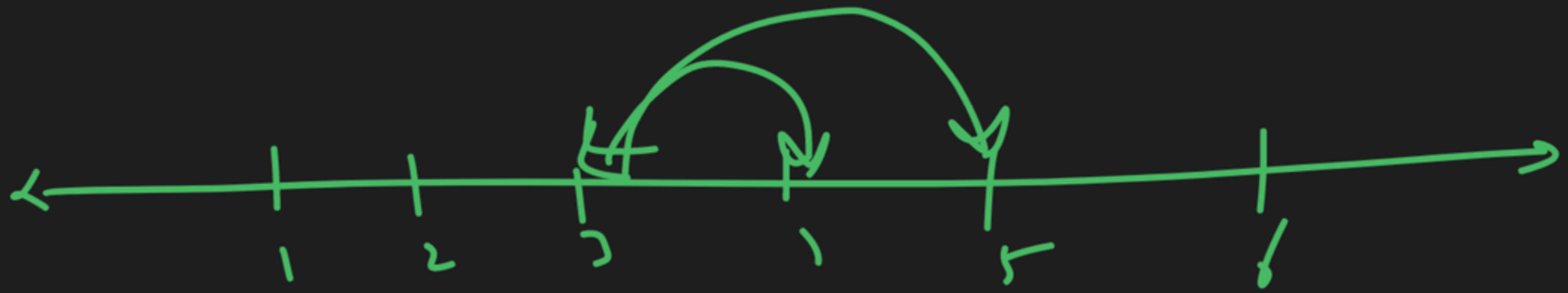
maxDistance

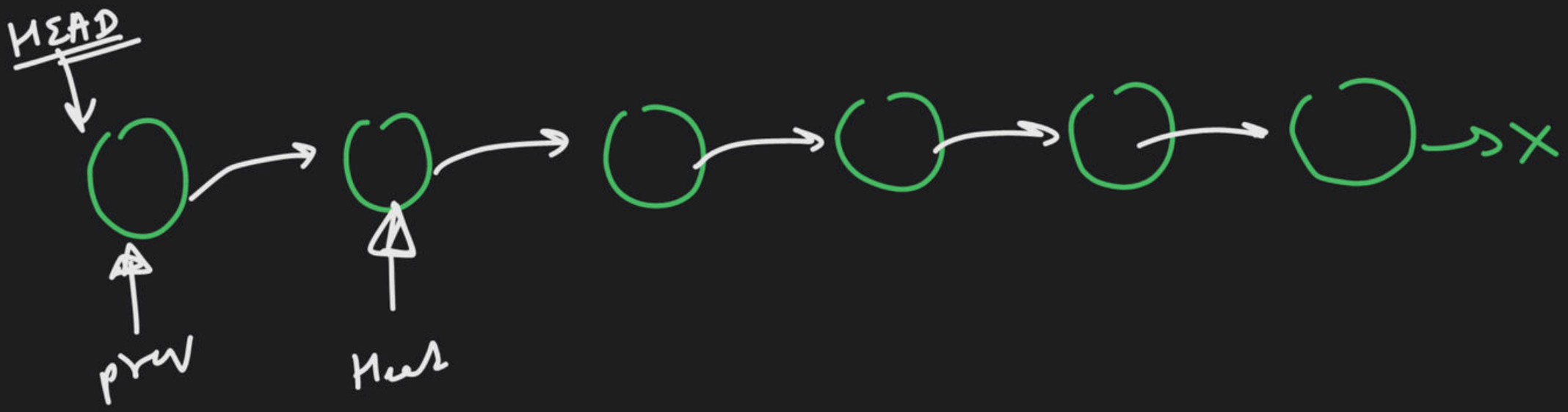


mind listen




2
3
h
C
U
2

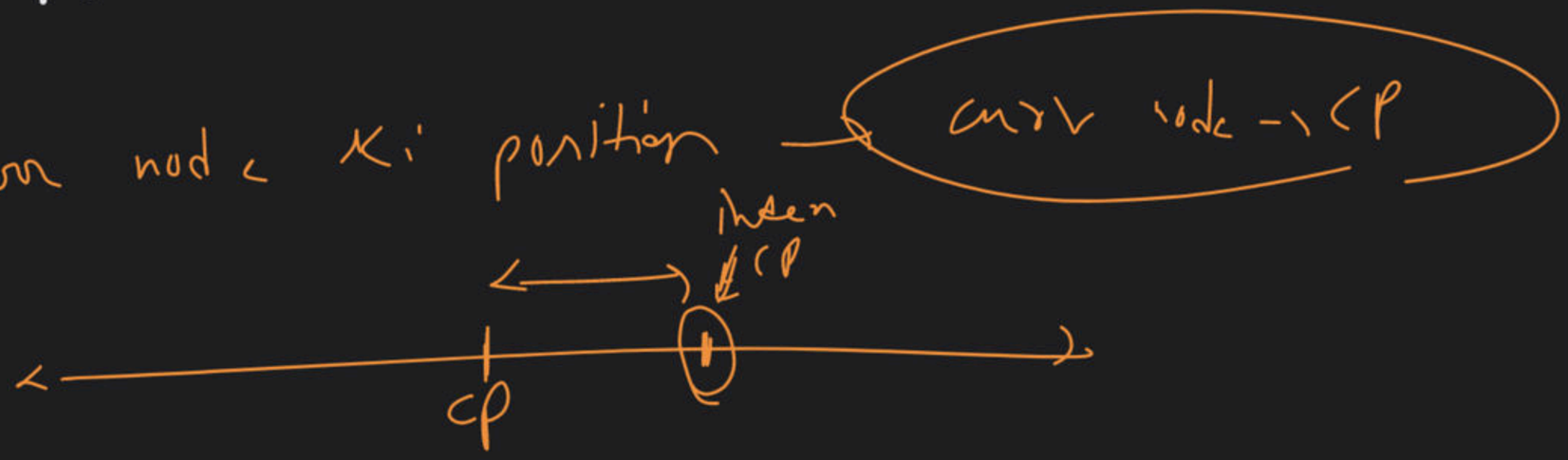

1
2
3
1
5
1



Node * prev = head

Node * head = head -> next

index -> curr node ki position



while (head -> next != NULL)

{

Local minima → if (head -> data < prev -> data

&&

head -> data < head -> next -> data)

{

firstPoint = min (firstPoint, index);

LastPoint = max (lastPoint, index)

if (cp != -1)

minDistance = min (minDistance, index - cp)

update cp → cp = index ;

totalCp++; }

Local maxima

```
if ( head -> data > prev -> data
     &&
     head -> data > head -> next -> data )
{
```

for calculating maxDistance:

- firstPoint = min (firstPoint , index)
- lastPoint = max (lastPoint , index)

for calculating minDistance:

```
if ( cp != -1 )
    minDistance = min ( minDistance , index - cp )
```

cp update to track position of recently found CP & trace total CP

```
cp = index
totalCp++ ;
```

next node →

index ++;

prev = head

head = head -> next;

}

done

seconds

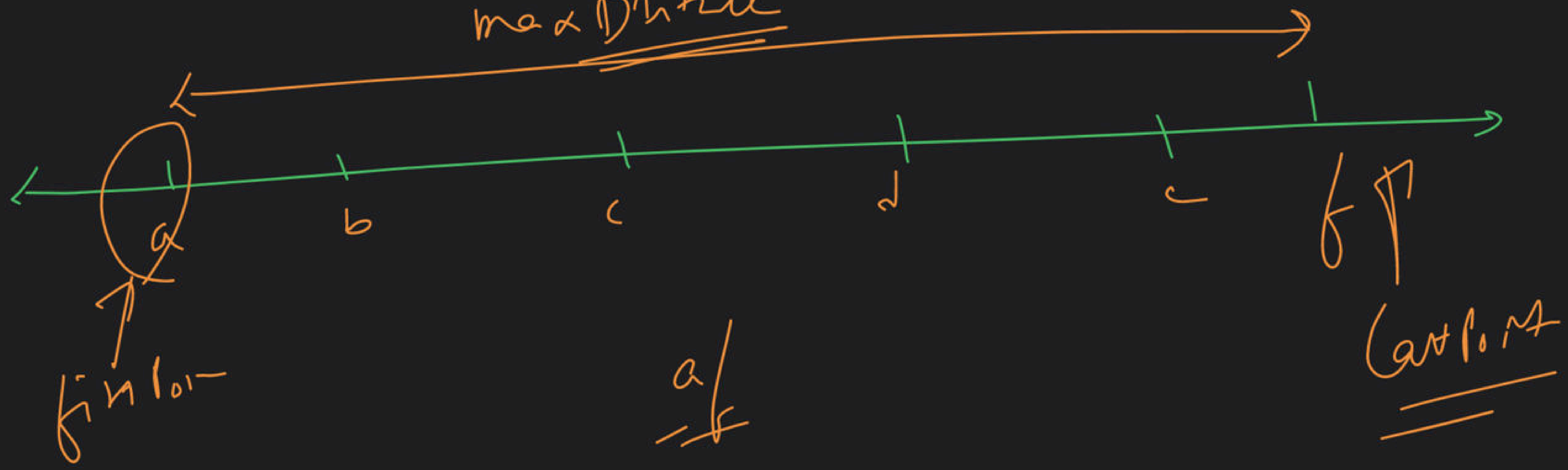
if (total(P < 2)

return [-1, -1];

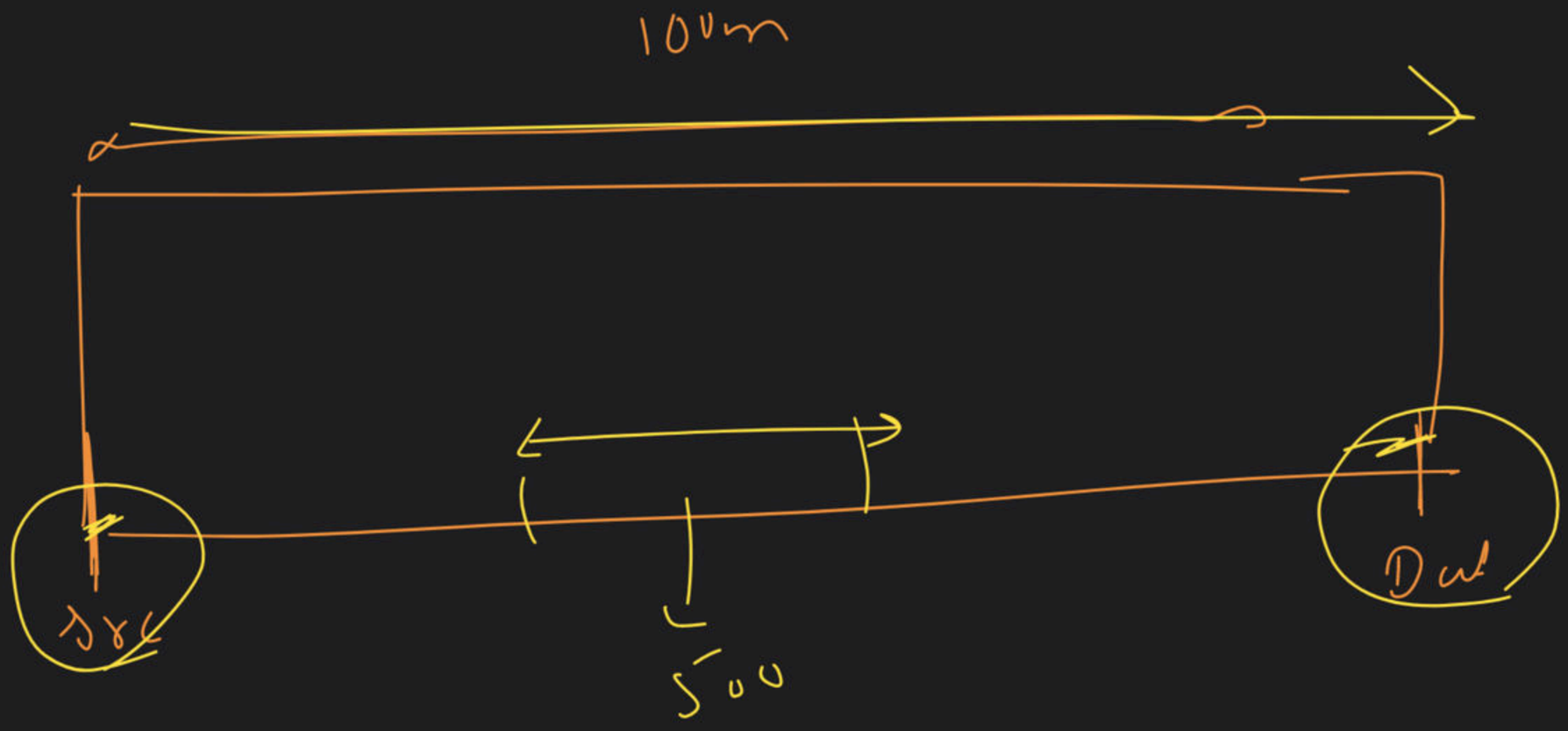
maxDistance = lastPoint - firstPoint

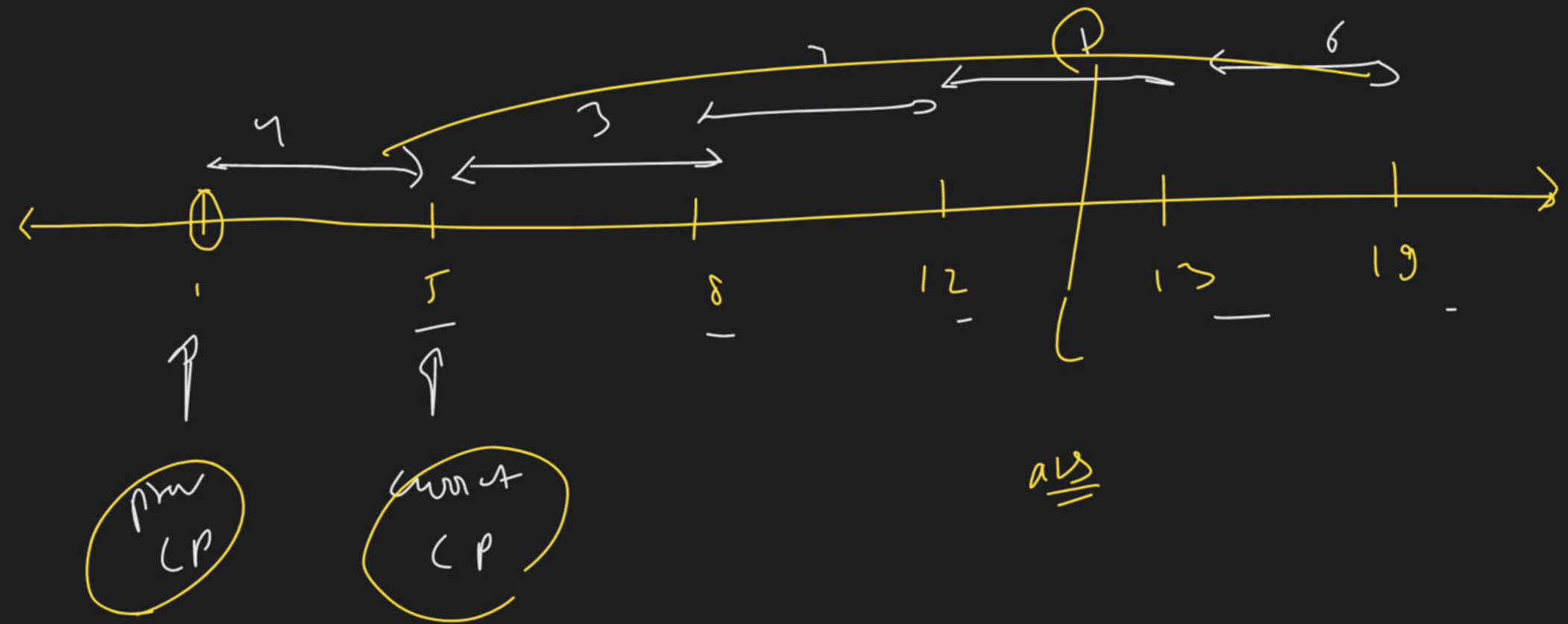
arr → minDist, maxDi;

return arr; }



max Distance → 0


100m



als

minDistance = min(minDistance, index - 4)

How it works?

L.L → assignment → Stacks