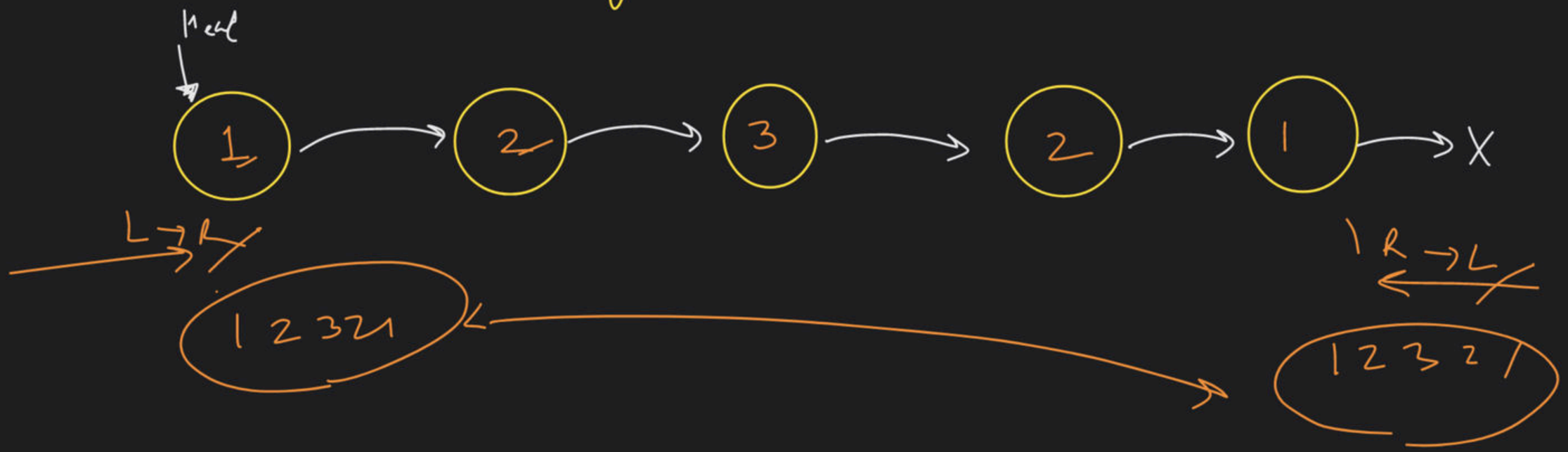
LL Class - 4

Special class

Love Babbar • Apr 22, 2023

Check whether your LL is palindrome or not



Approach #1

- normal → i/p
- reverse LL

Compare → T/F

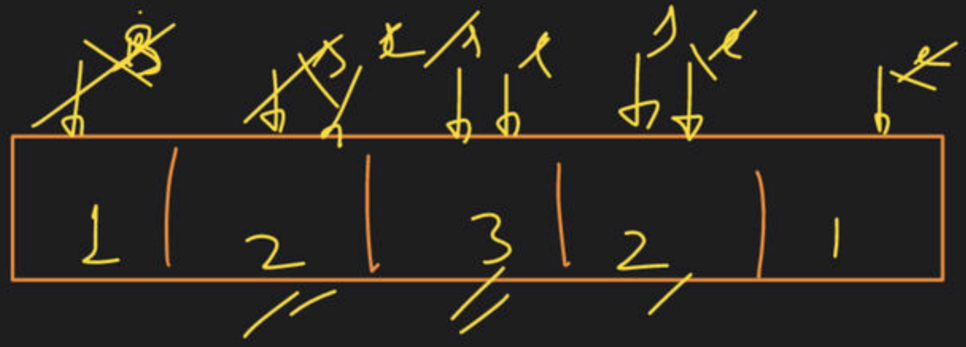
T.C → O(n)

S.C → ~~O(n)~~ O(1)

approach #2

T.C → O(n)

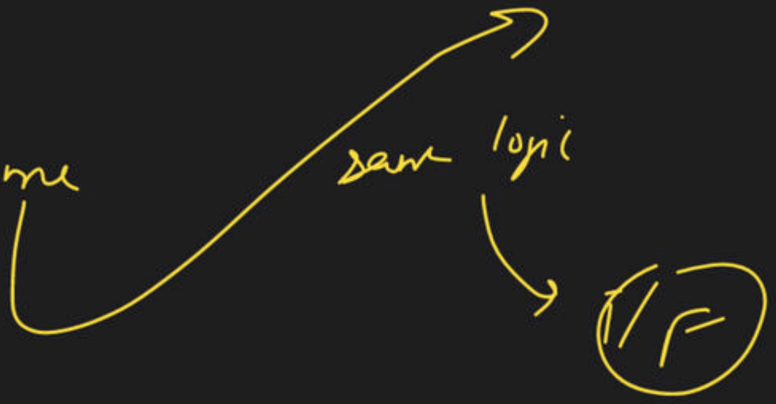
S.C → O(n)



↳ LL data

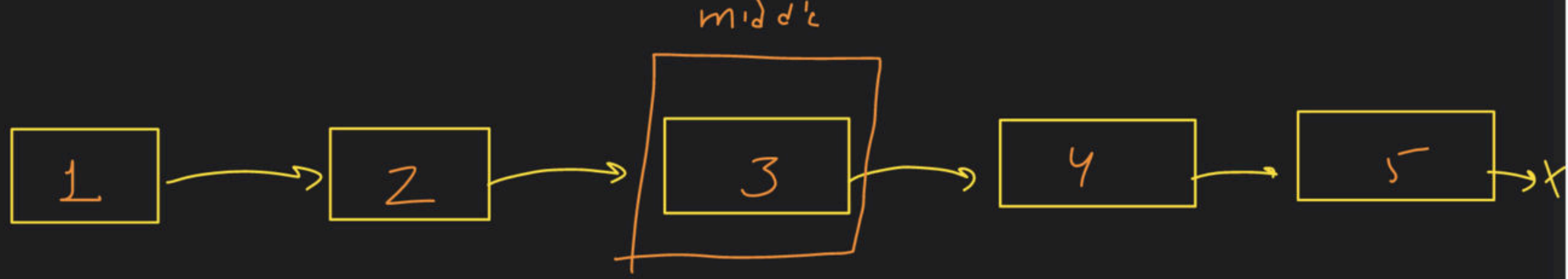
↳ copy array me

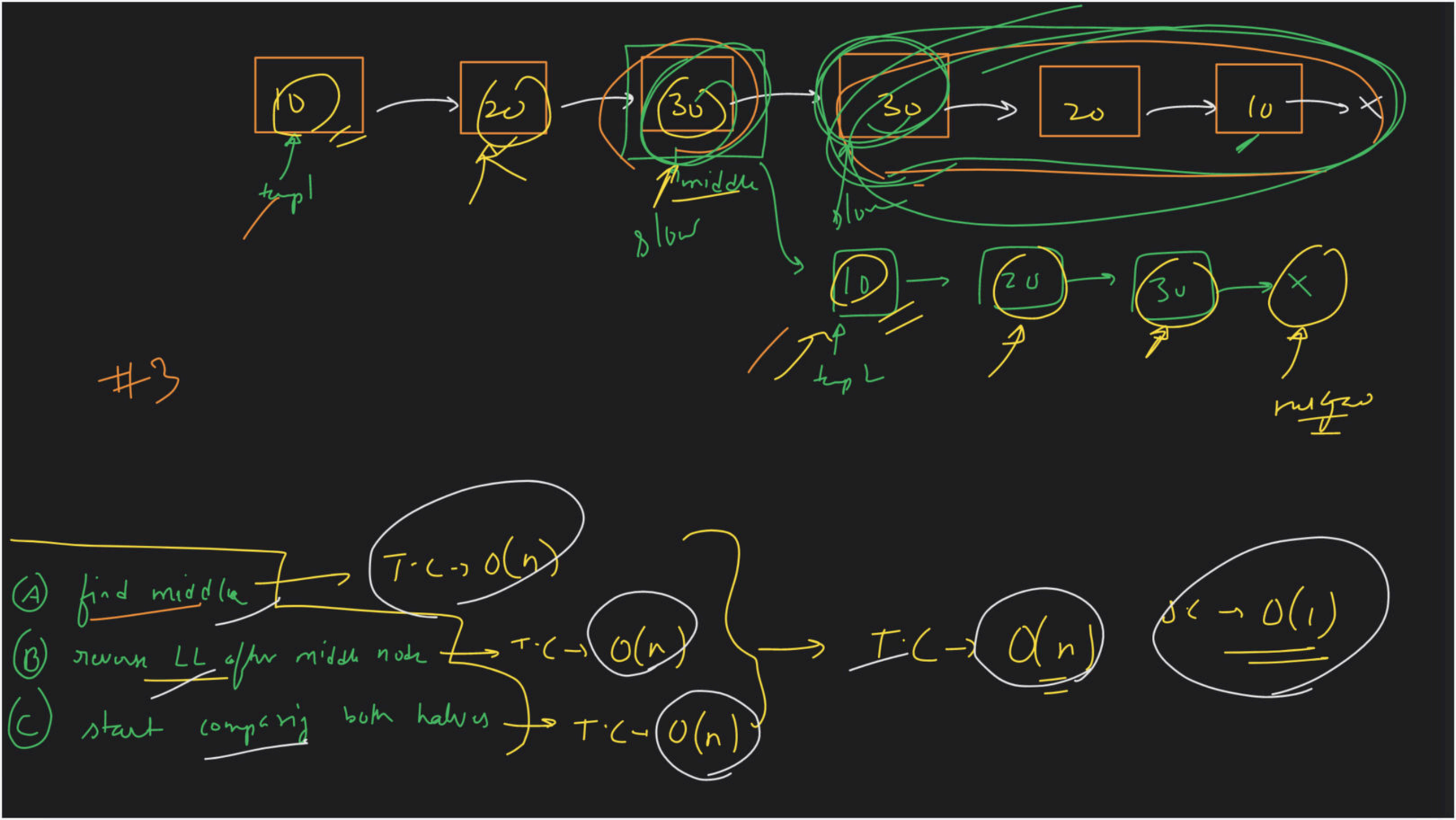
same logic

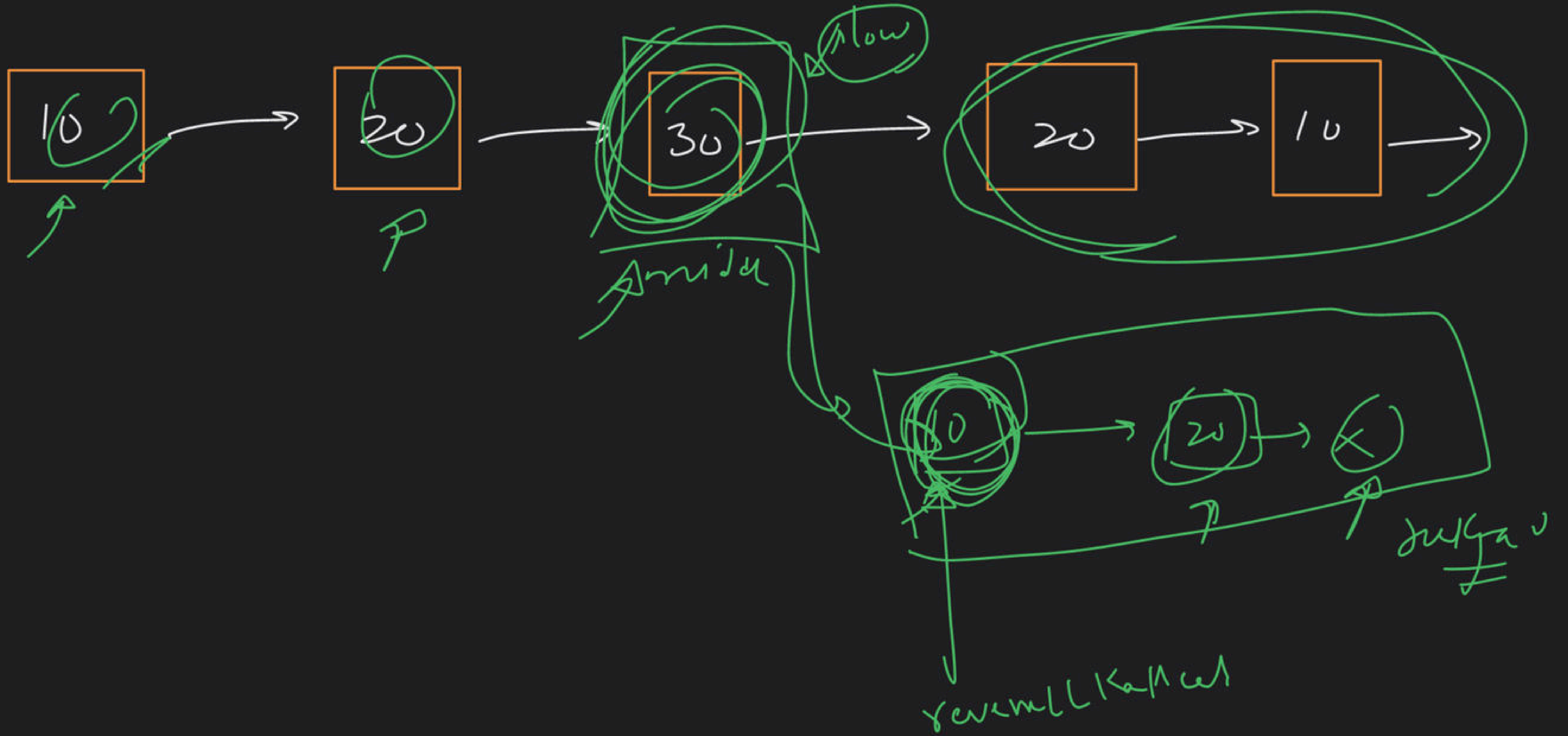


→ Rukjao

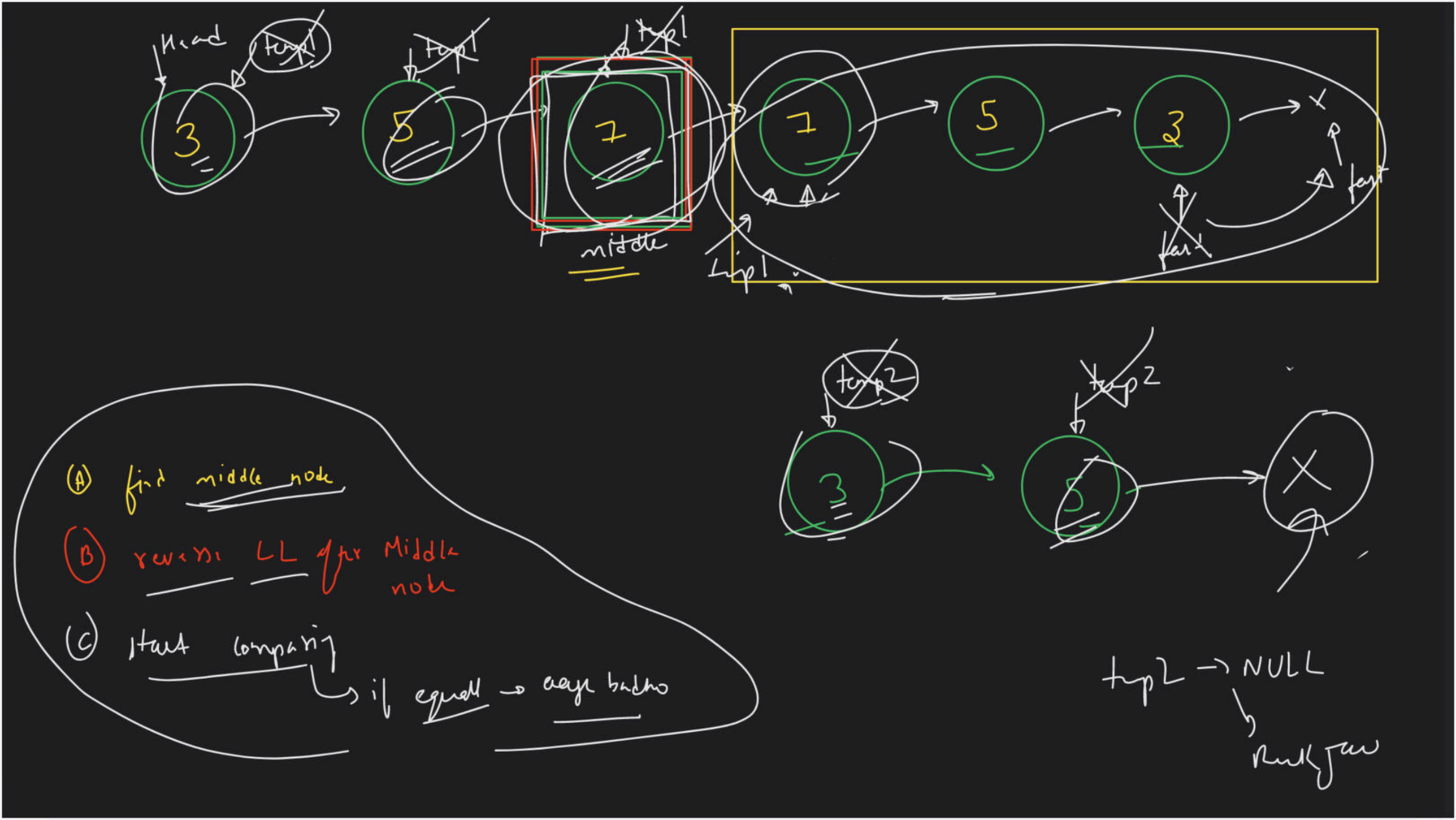
approach #3



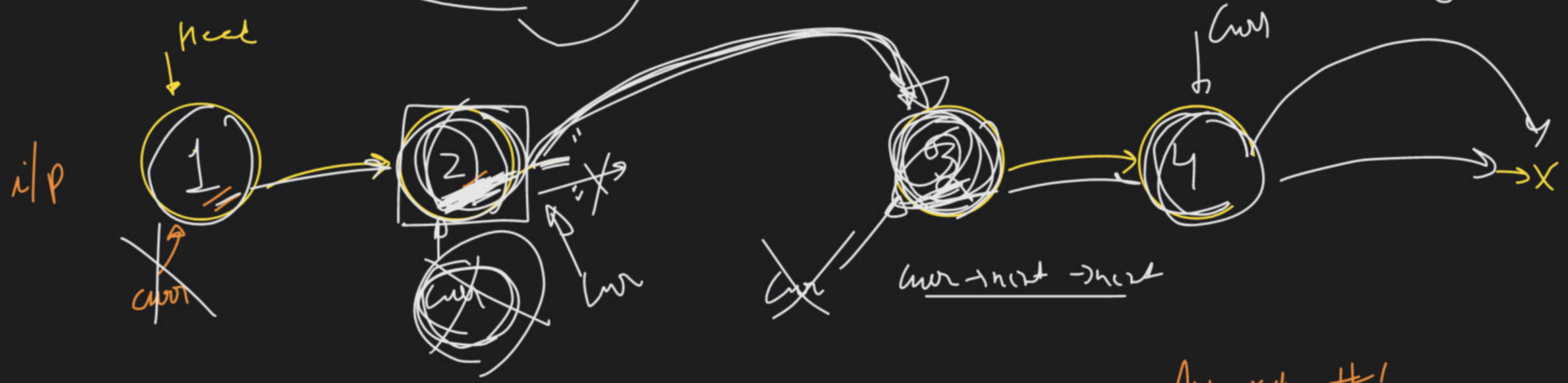

10
20
30
30
20
10
temp1
slow
middle
slow
10
20
30
X
temp2
null
#3
(A) find middle → T.C → O(n)
(B) reverse LL after middle node → T.C → O(n)
(C) start comparing both halves → T.C → O(n)
T.C → O(n)
S.C → O(1)



10
20
30
20
10
slow


Head
temp1
3
temp1
5
temp1
7
middle
temp1
7
5
3
X
fast
fast
temp2
3
temp2
5
X
(A) find middle node
(B) reverse LL after Middle node
(C) start comparing
if equal → carry on
temp2 → NULL
Run Jaw

→ Remove duplicates from a Sorted L.L $\frac{O(n) \rightarrow T.C}{O(1) \rightarrow S.C}$



o/p → 1 → 2 → 3 → 4 → X

Approach #1

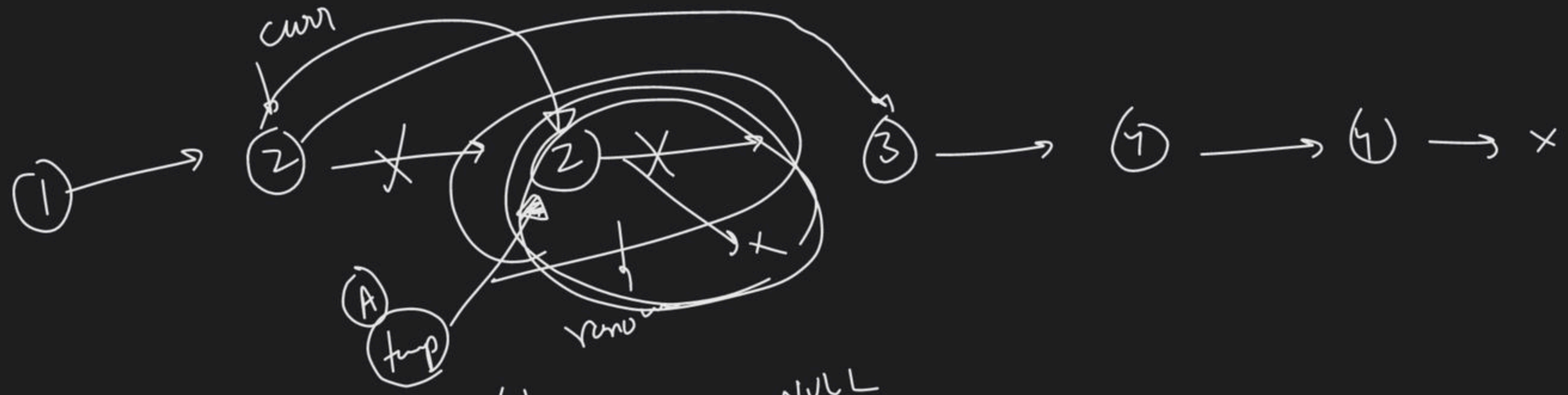
curr → data == curr → next → data

equal

curr → next = curr → next → next

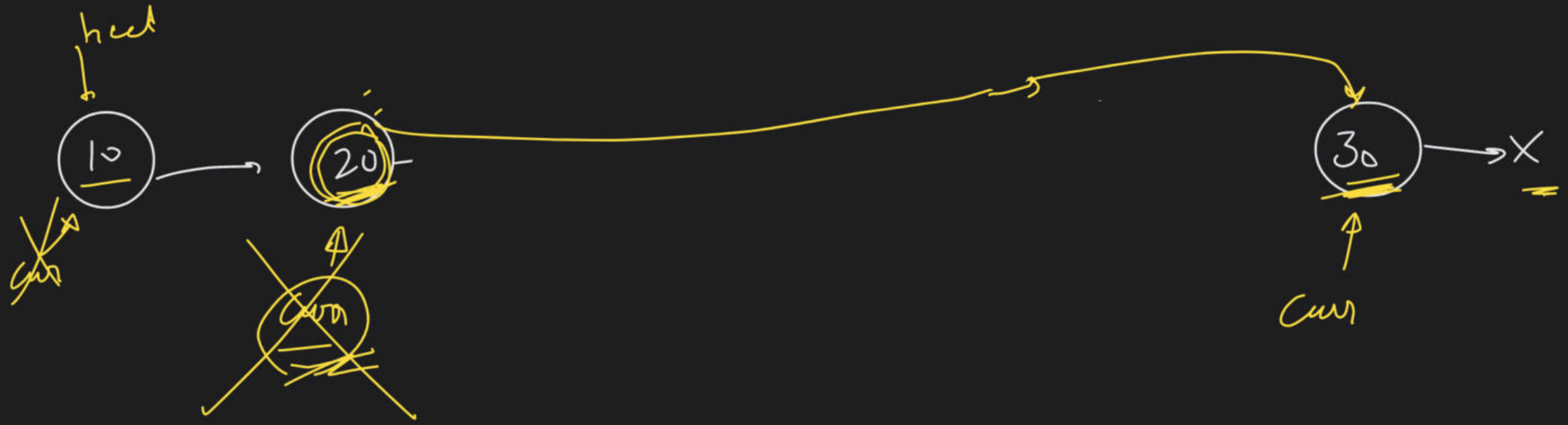
not equal

curr = curr → next



(B) temp → next = NULL

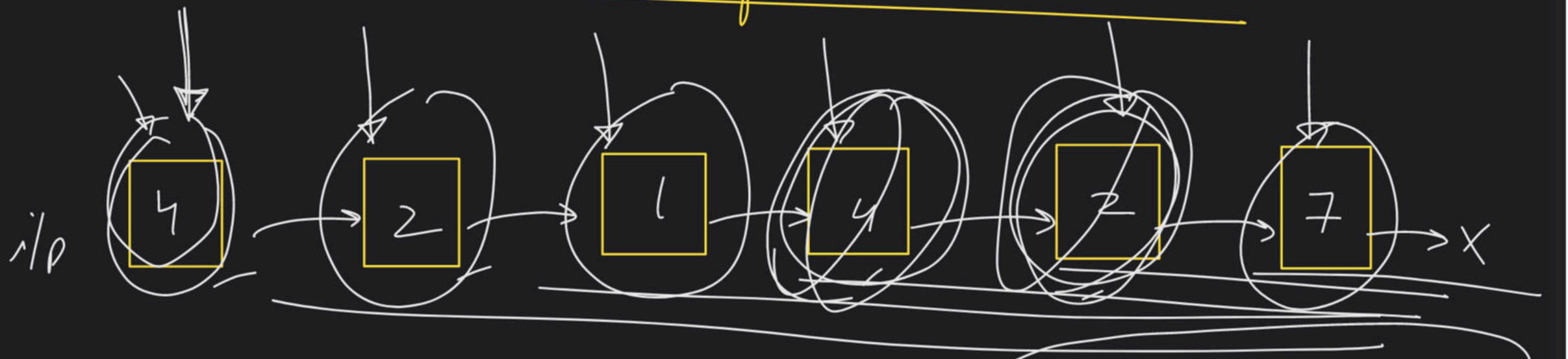
(C) delete temp


head
10
20
30
X
curr
curr


10
20
30
X

H/W

Remove duplicate from Unsorted LL



i/p -> 4 -> 2 -> 1 -> 7 -> X

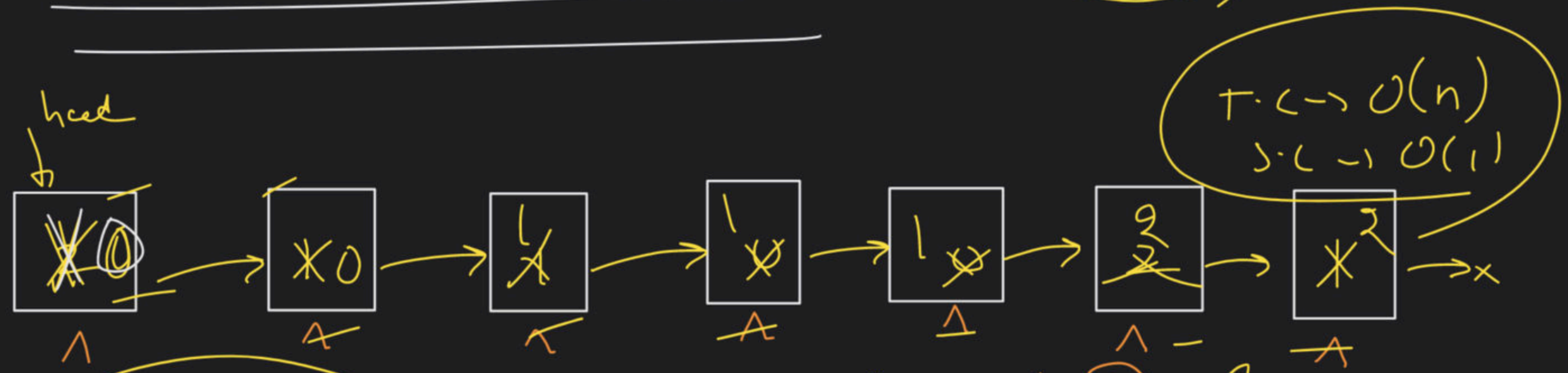
#1 → nested loop

#2 → map

#3 → 1 VI & prev logic

Sort 0s, 1s & 2s

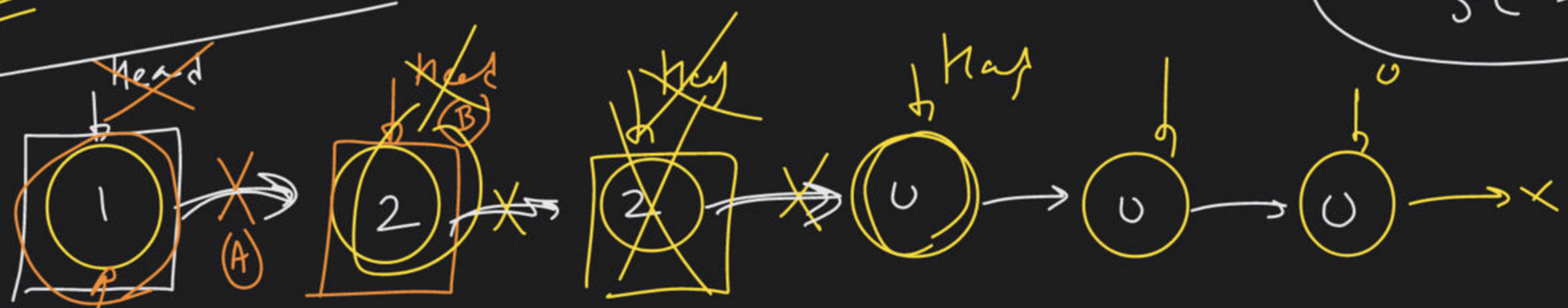
Kaise karoge → #1 → country



→ Pyaari Approach

T -(→ O(n)

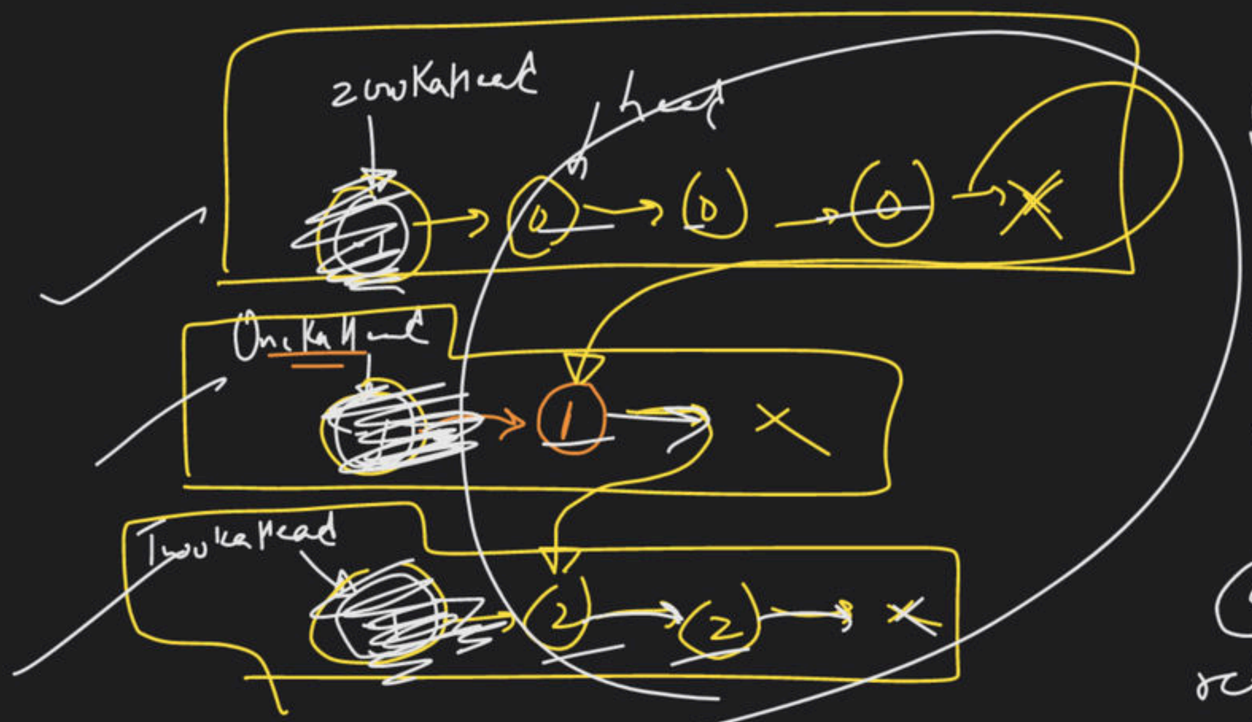
S.C → O(1)



(A) temp = head

(B) ~~temp~~ head = head -> next

(C) temp -> next = NULL



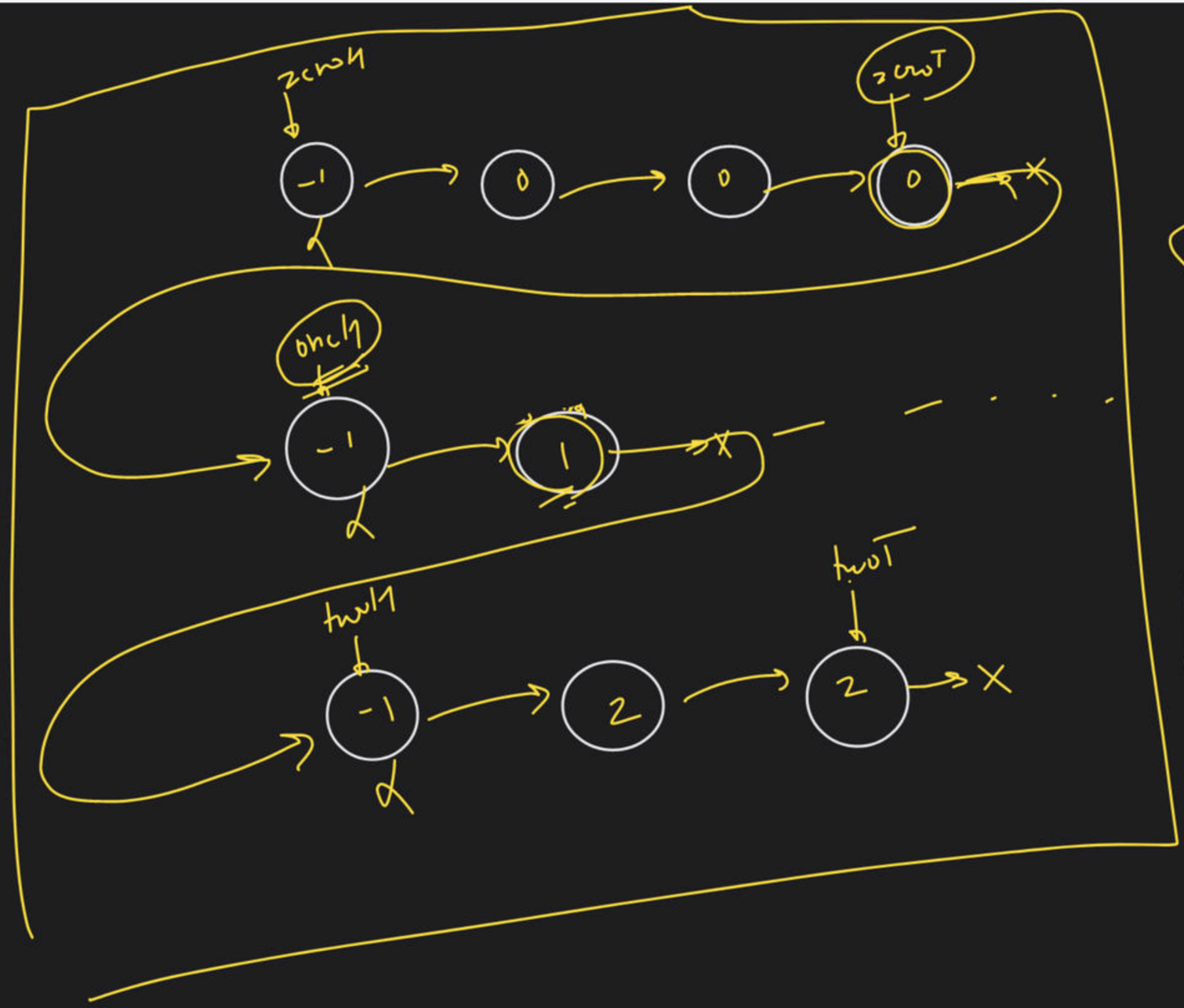
(A) Join

(B) zero ka dusra hua

(C) zero ka head

2 min

Break

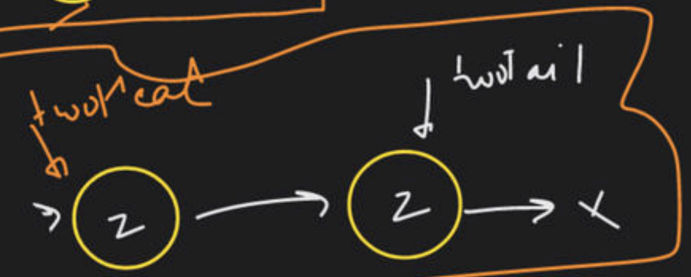


Do it

zeroTail → next = one

zero → next = one

oneTail → next = two



A

```
temp = oneHead
oneHead = oneHead->next
temp->next = NULL
delete temp
```

main

C

```
if (oneHead != NULL)
{
    zeroTail->next = oneHead;
}
else {  if (twoHead != NULL)
        zeroTail->next = twoHead;
    }
```

B

```
temp = twoHead
twoHead = twoHead->next;
temp->next = NULL
delete temp
```

D

```
temp = zeroHead
zeroHead = zeroHead
           ->next
temp->next = NULL
delete temp
```

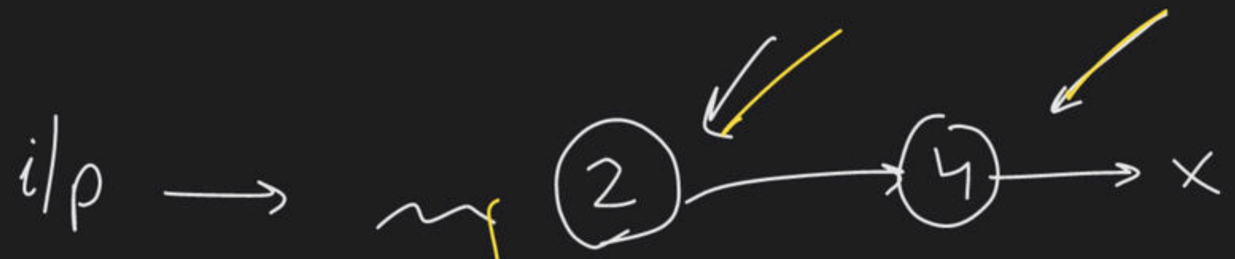
E

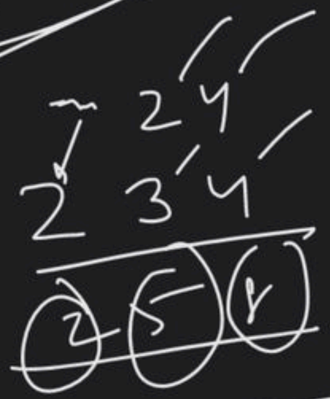
return zeroHead

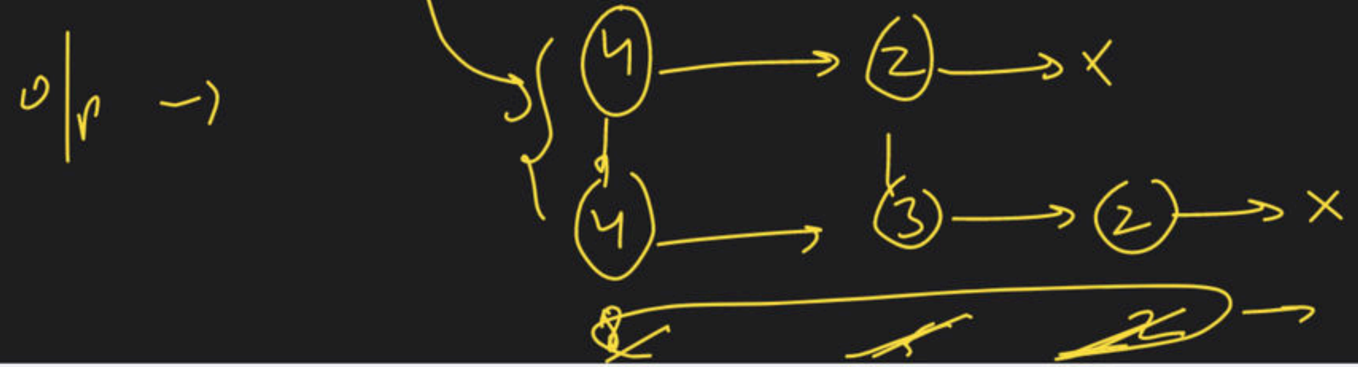
Add 2 numbers represented by Linked List

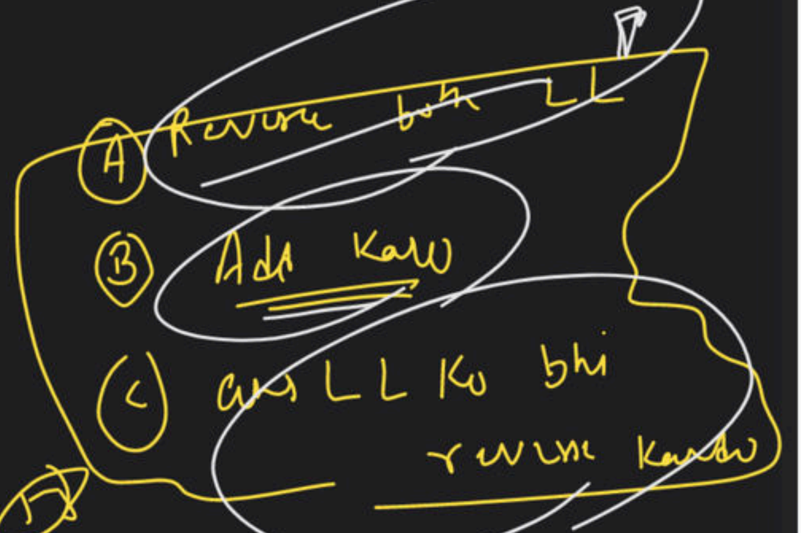
O(1)

O(max(m, n))



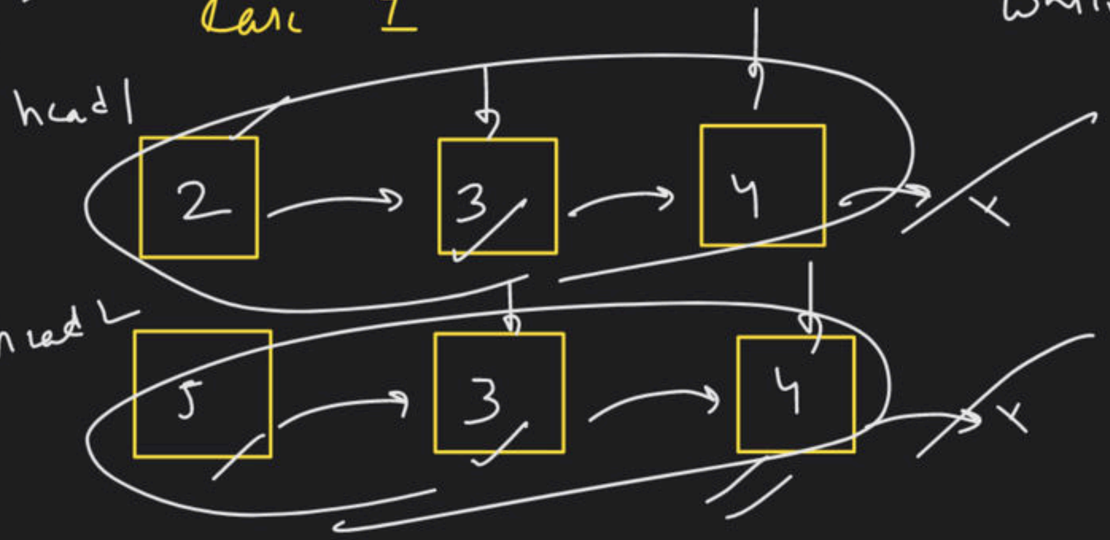




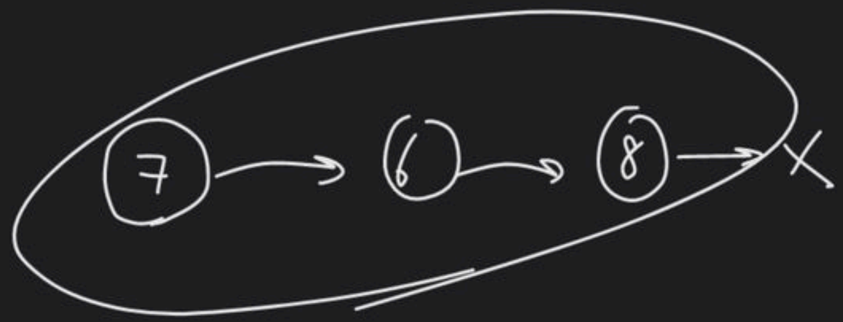


carry = 0

Case 1



while (———)



- sum = 0 + 2 + 5 = (7)
- digit = 7
- carry = (0)

sum = 0 + 3 + 3 = 6

digit = (1)

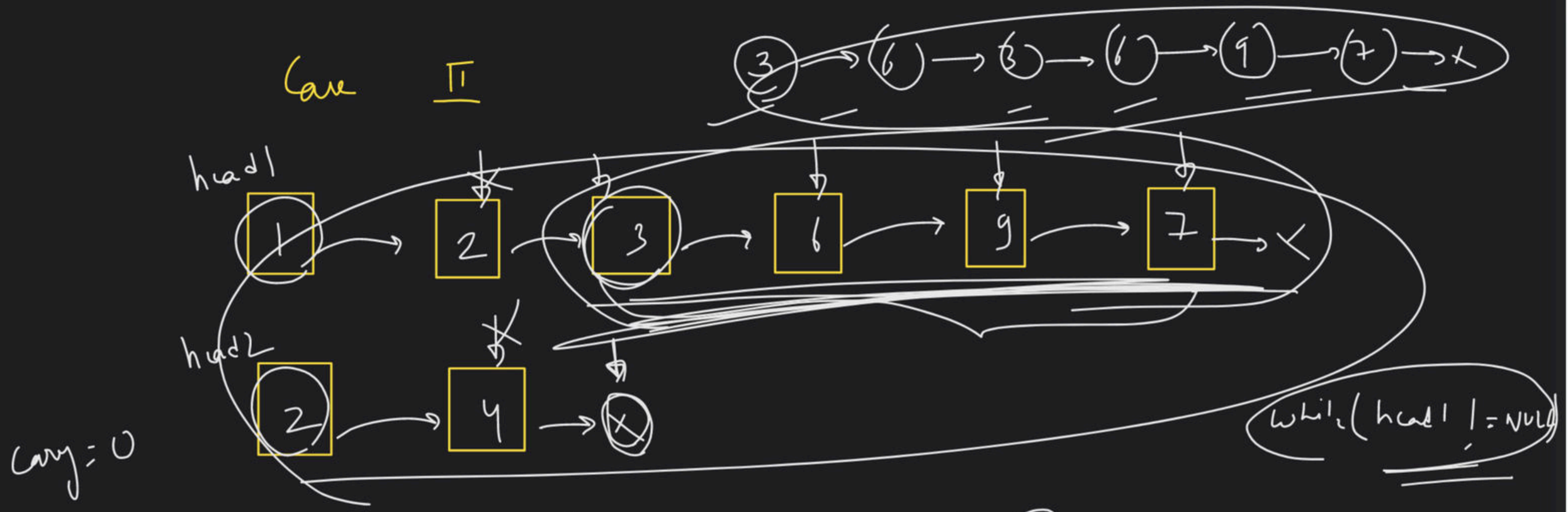
carry = 0

sum = 0 + 4 + 4 = 8

digit = (8)

carry = 0

Case II



sum = 0 + 1 + 2 = 3

digit = 3

carry = 0

sum = 0 + 2 + 4 = 6

digit = 6

carry = 0

sum = 0 + 3

digit = 3

carry = 0

sum = 0 + 6

d = 6

c = 0

sum = 0 + 7

d = 7

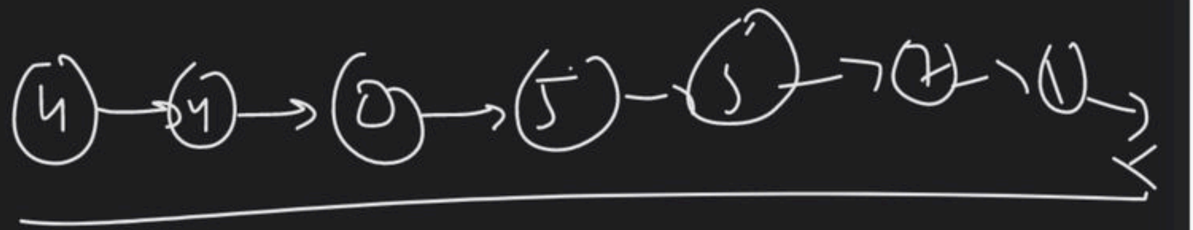
c = 0

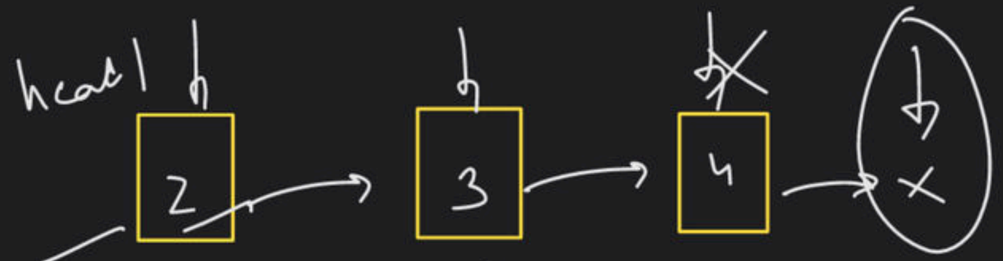
sum = 0 + 7

d = 7

c = 0

Case III





sum = 0 + 2 + 2 = 4

digit = 4

carry = 0

sum = 0 + 3 + 1 = 4

digit = 4

carry = 0

sum = 0 + 4 + 6 = 10

digit = 0

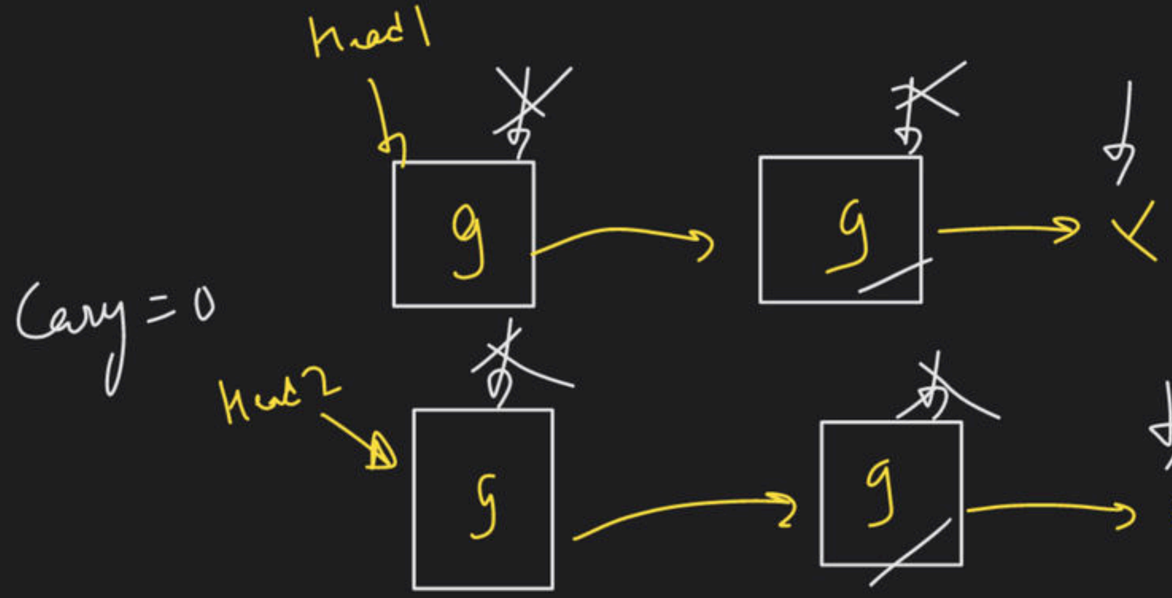
carry = 1

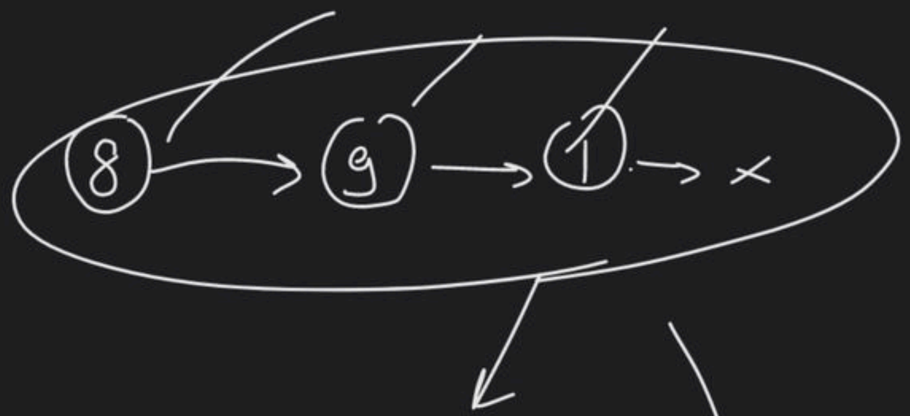
sum = 1 + 4 = 5

d = 5

c = 0

while (head2 != N)





sum = 0 + 9 + 9 = 18

digit = 8

carry = 1

sum = 1 + 9 + 9 = 19

digit = 9

carry = 1

sum = 1

d = 1

c = 0

Merge 2 sorted Linked List