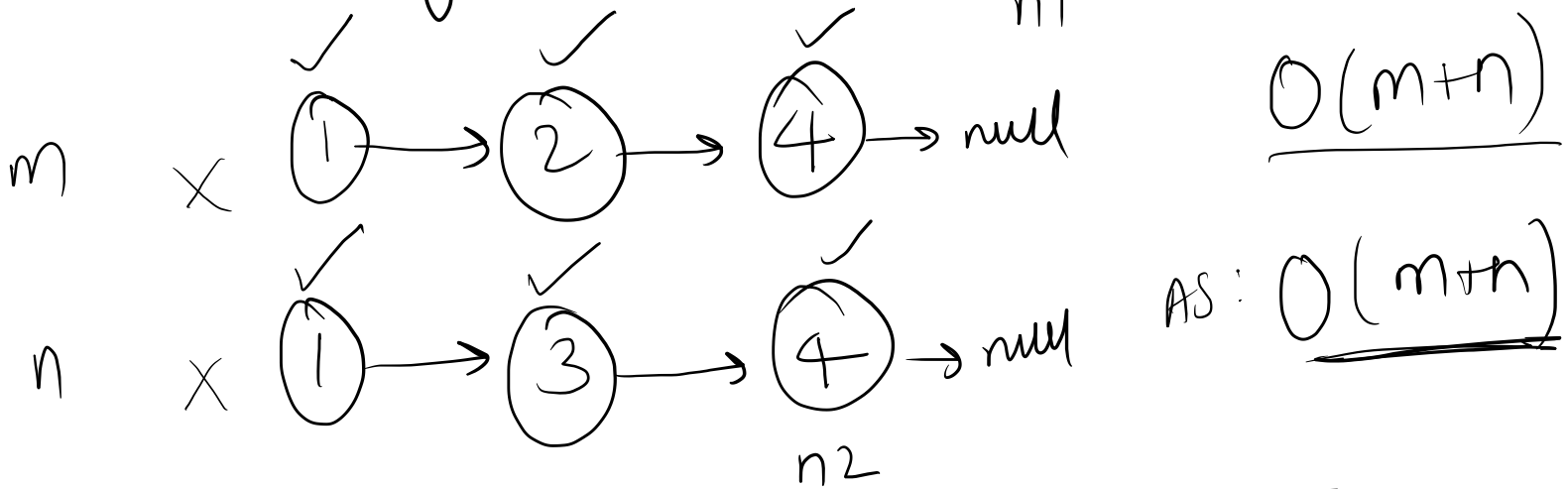
LINKED LIST

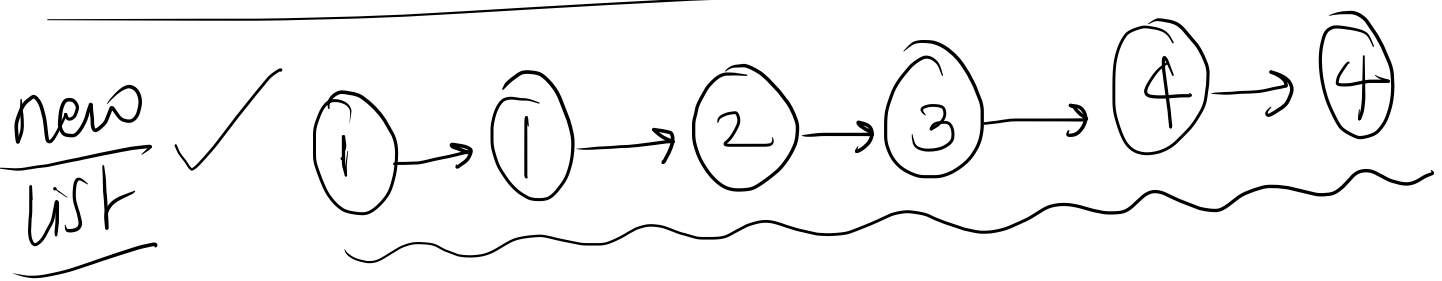
- ✓ Delete Node
- ✓ Reverse Linked List
- ✓ Remove element

Merge two Sorted Lists

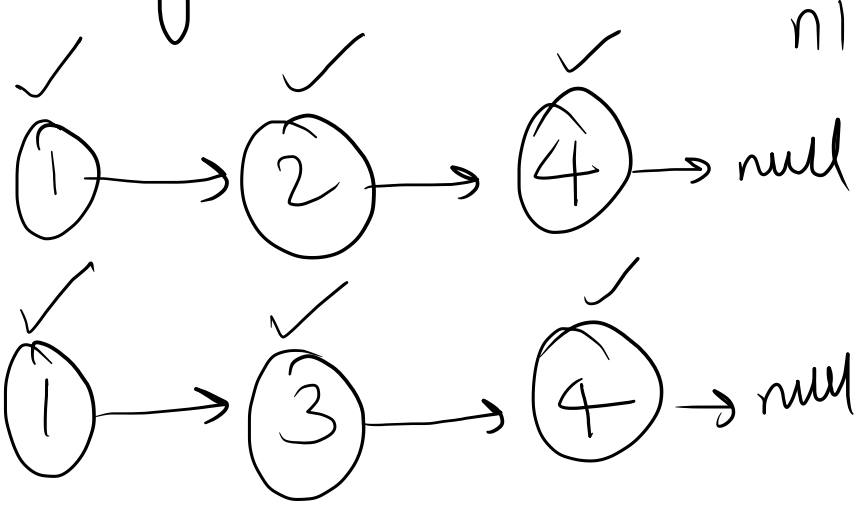


$O(m+n)$

AS: $O(m+n)$

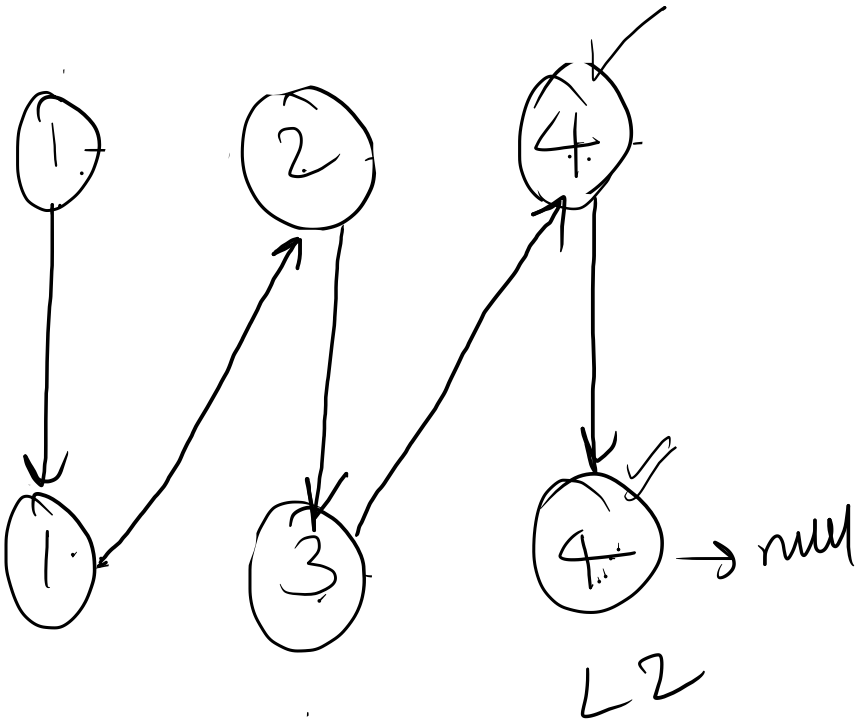


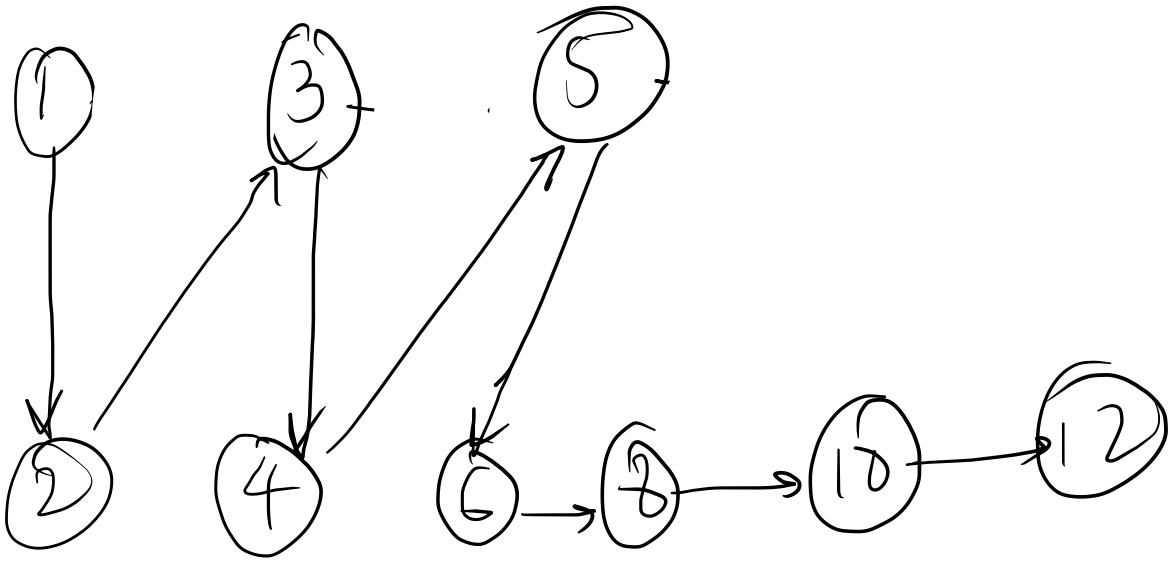
Merge two Sorted Lists

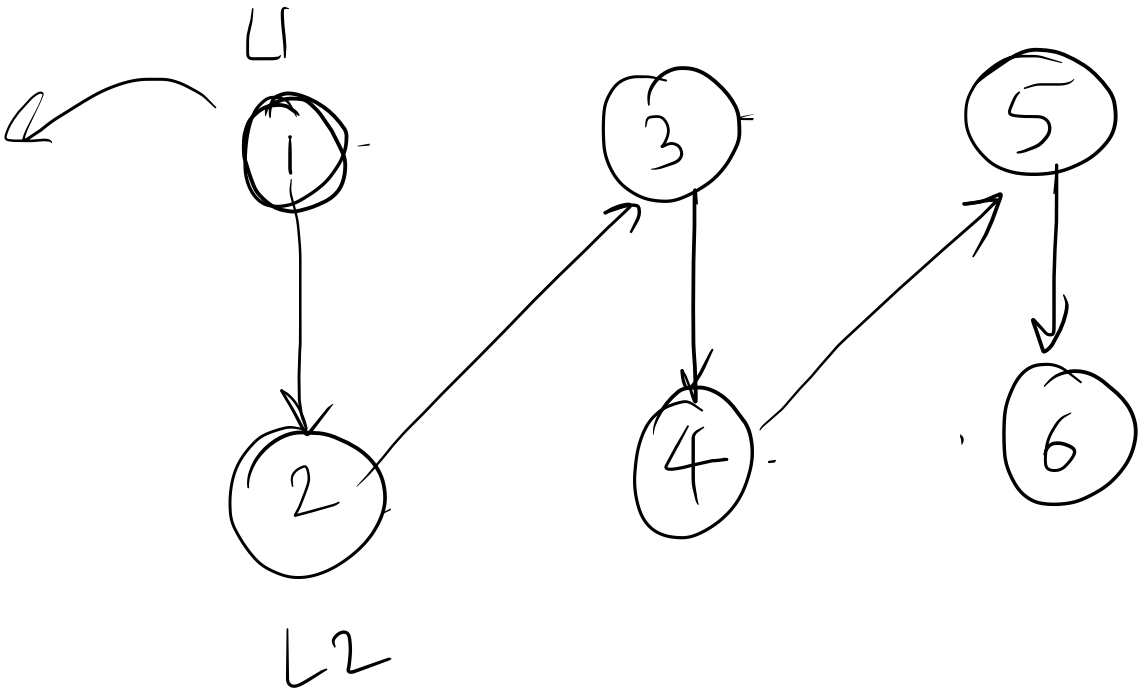


O(m+n)

AS: O(m+n)


1
2
4
1
3
4
null
L2

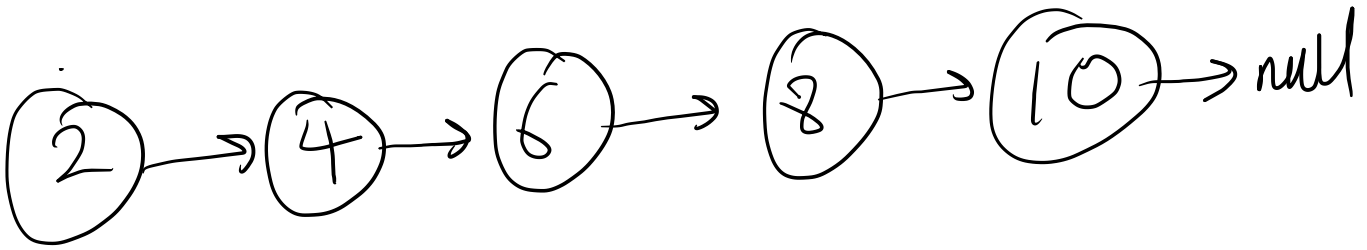

1
3
5
2
4
6
8
10
12


L1
1
3
5
2
4
6
L2

$$T.C = O(m+n)$$

$$Aux.Space = O(1)$$

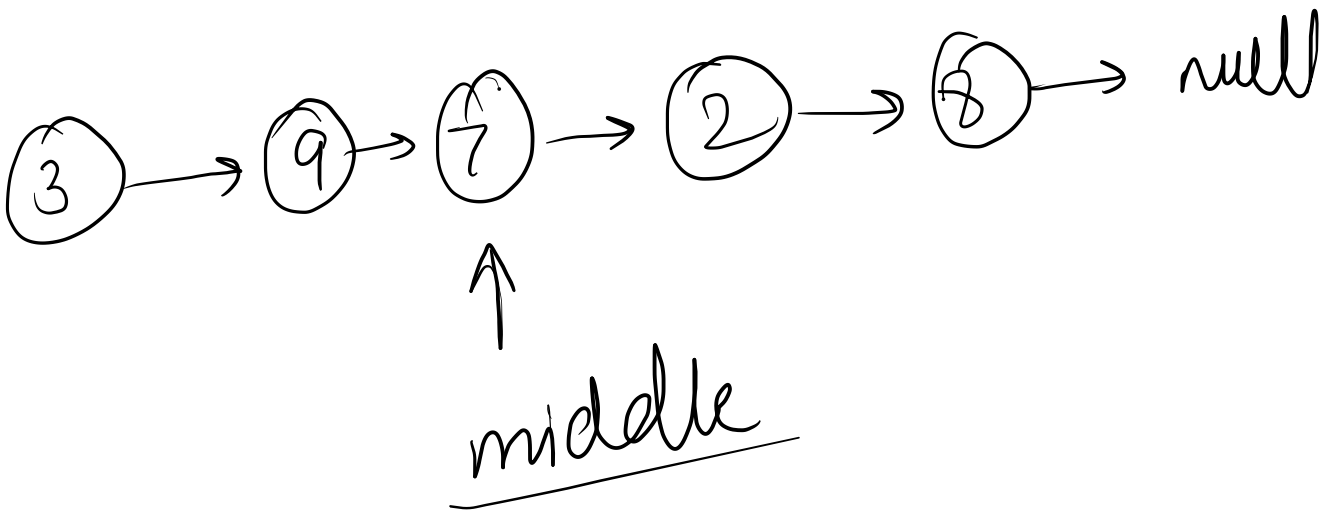
Middle of the LL

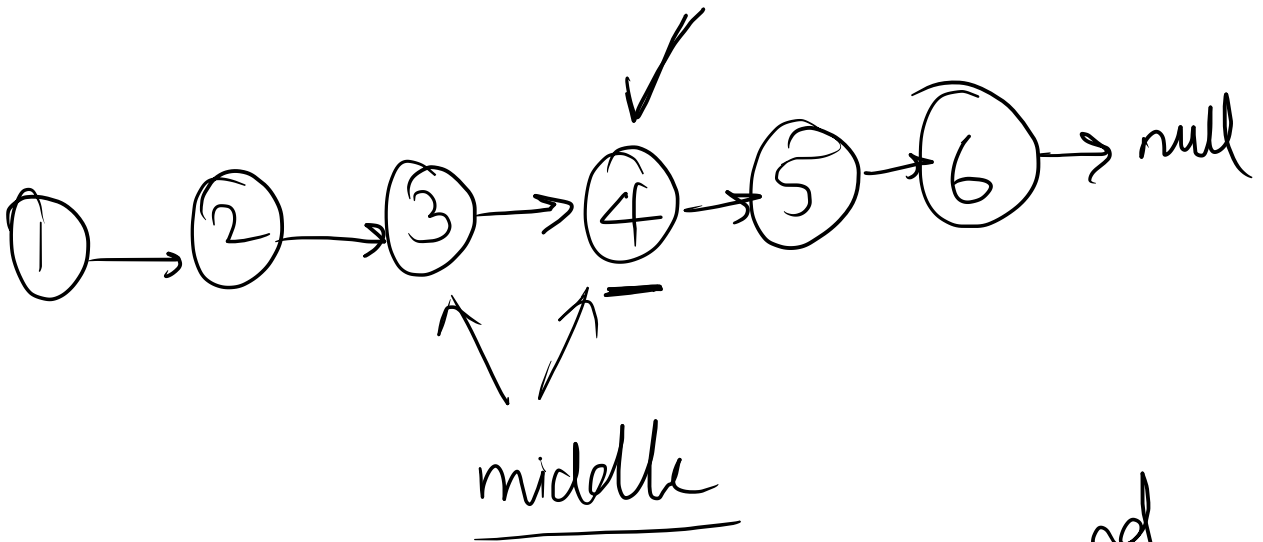


head

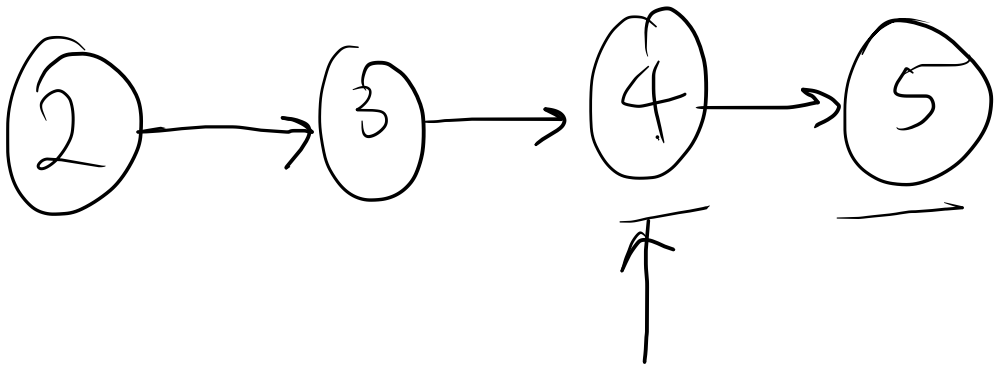
length = 5

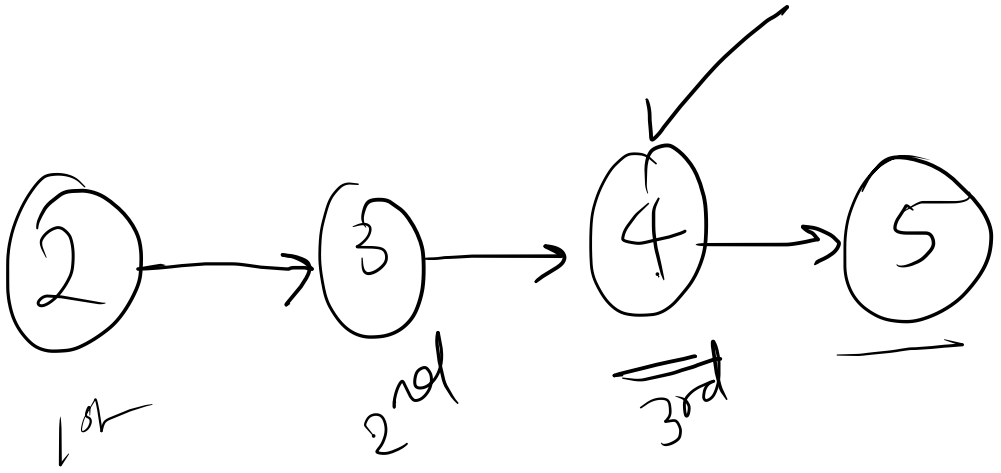
middle $\Rightarrow \overset{(ceil)}{5/2} = 3$


3
9
7
2
8
null
middle



2 middle nodes → return 2nd


2
3
4
5



length = 4

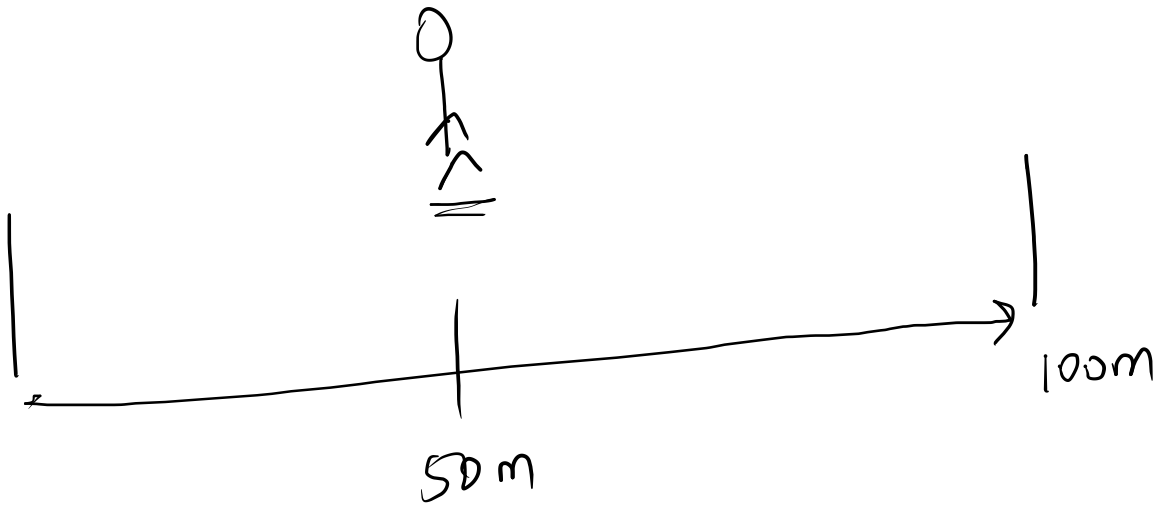
middle = 2+1 = 3

No. of iterations = 2.

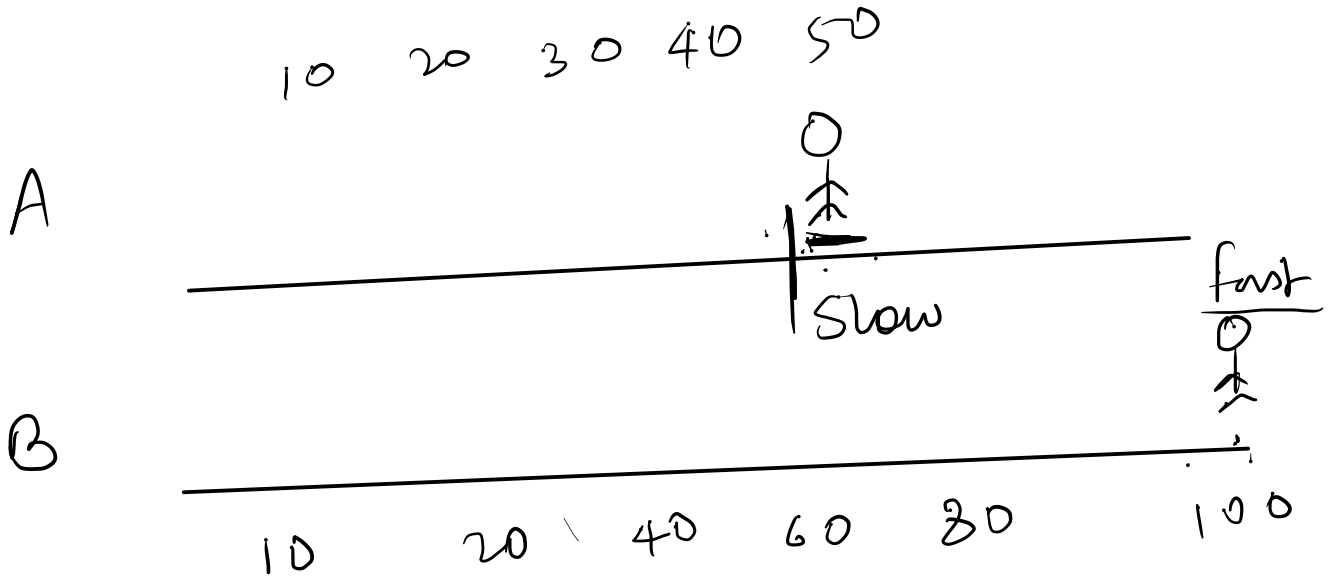
① count length → n

② reach middle → n/2

Floyd Algorithm

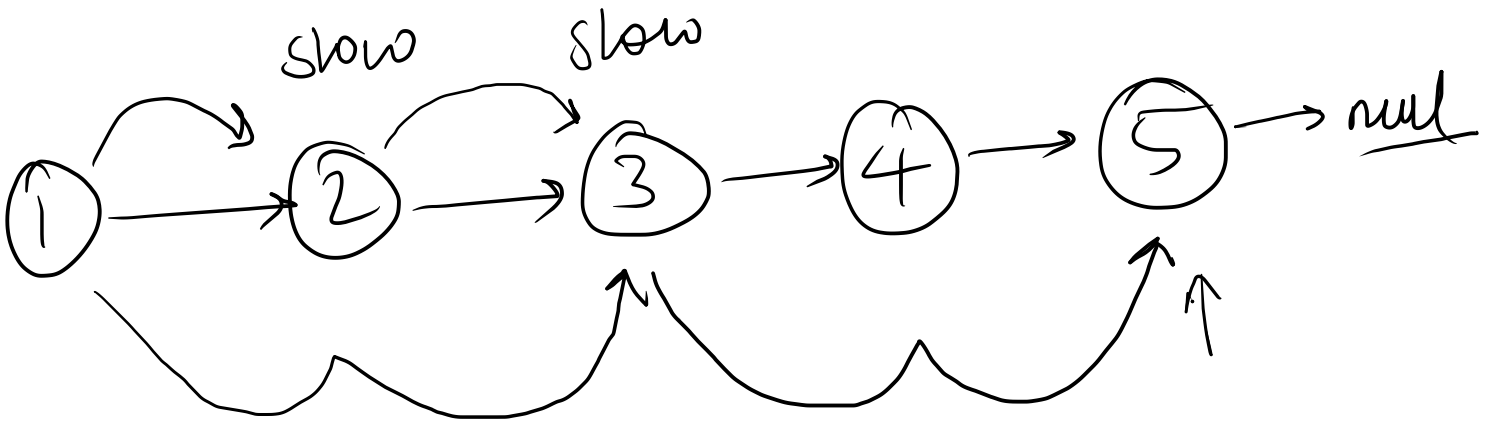


Floyd Algorithm



fast → end

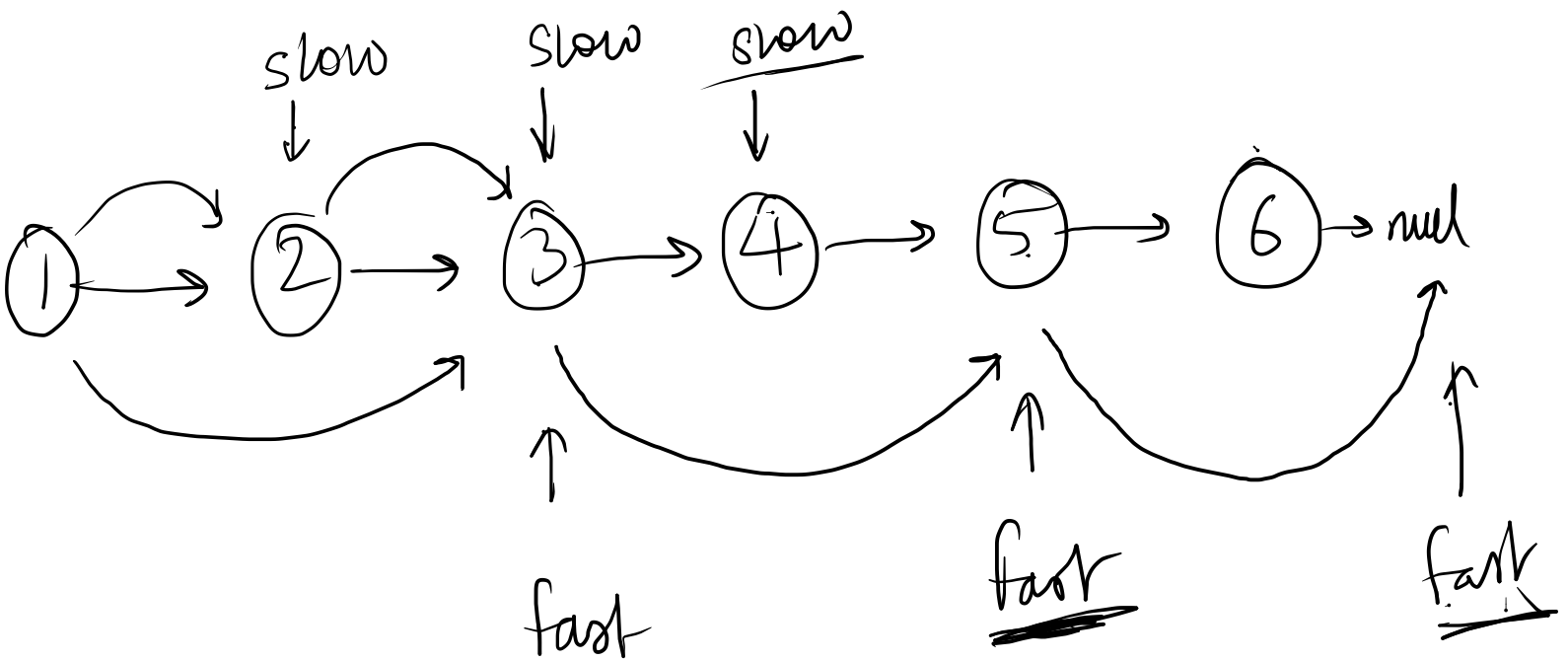
slow → middle



fast → end

slow → mid

middle = slow



fast → null

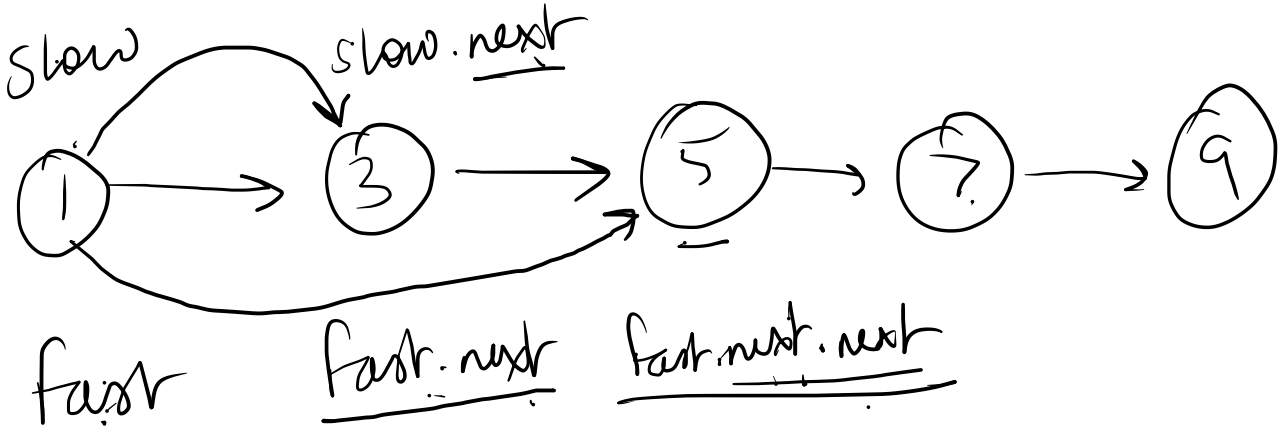
slow → mid

ans = slow

fast = null → stop

fast.next = null → stop

return slow


slow
slow.next
1
3
5
7
9
fast
fast.next
fast.next.next

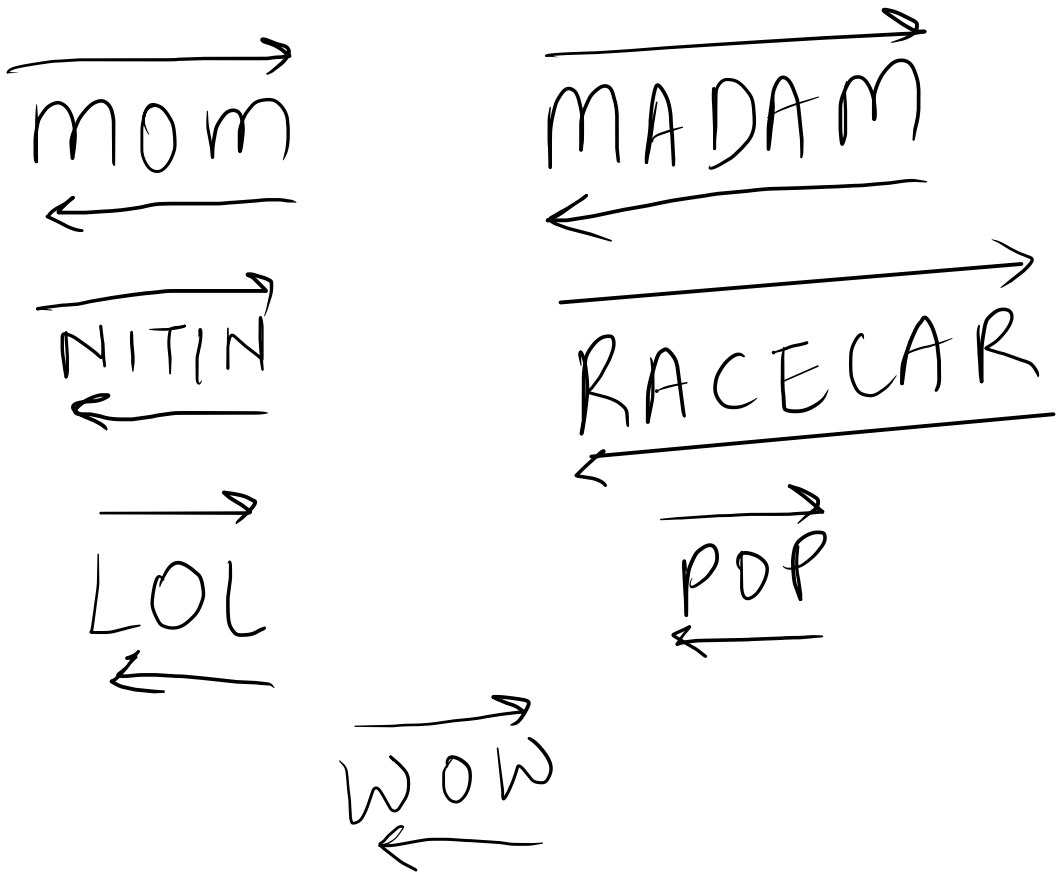
Stop when fast = null OR fast.next = null

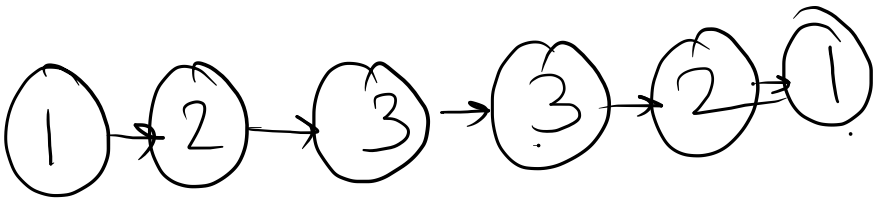
Continue when fast != null and fast.next != null

TC = O(n)

Aux. space = O(1)

Palindrome





forward

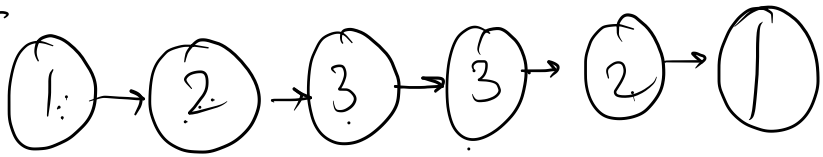
1 2 3 3 2 1

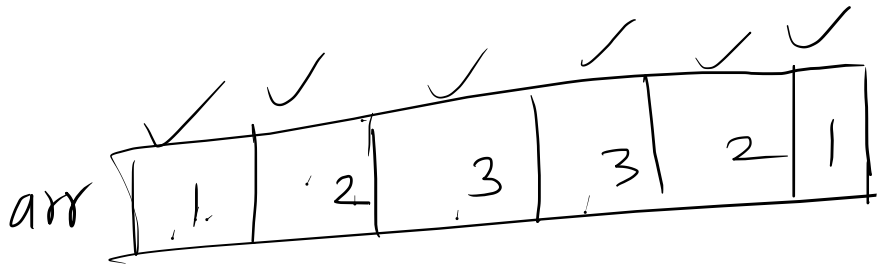
Backward

1 2 3 3 2 1

Palindrome

SC = O(n)



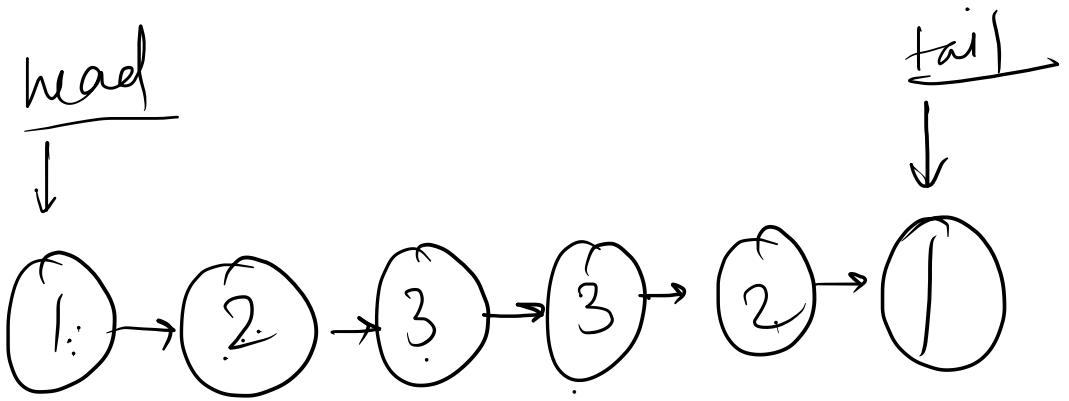


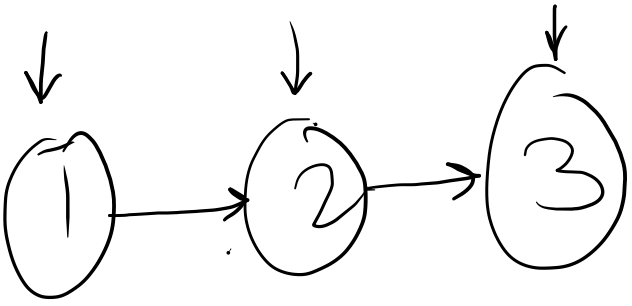
if (arr[s] != arr[e])
return false

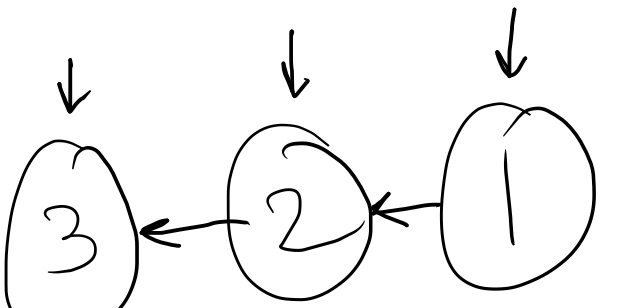
while (s < e)

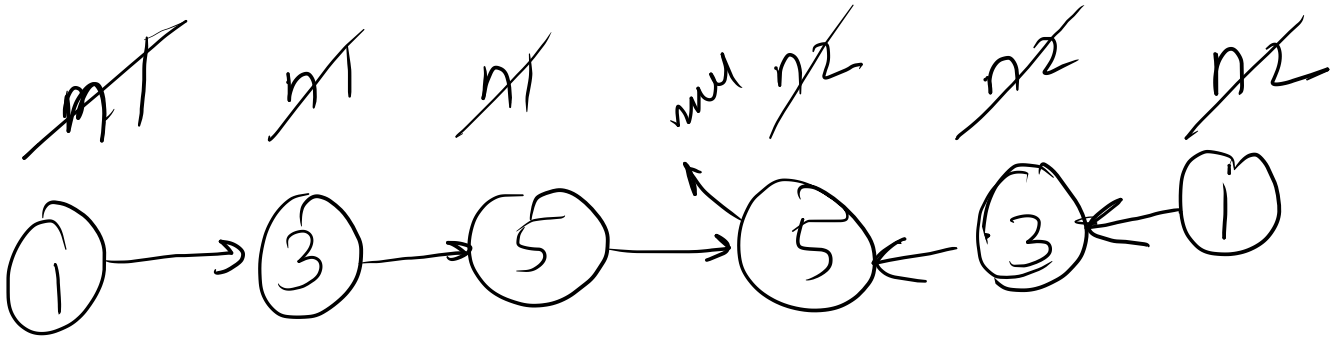
s++

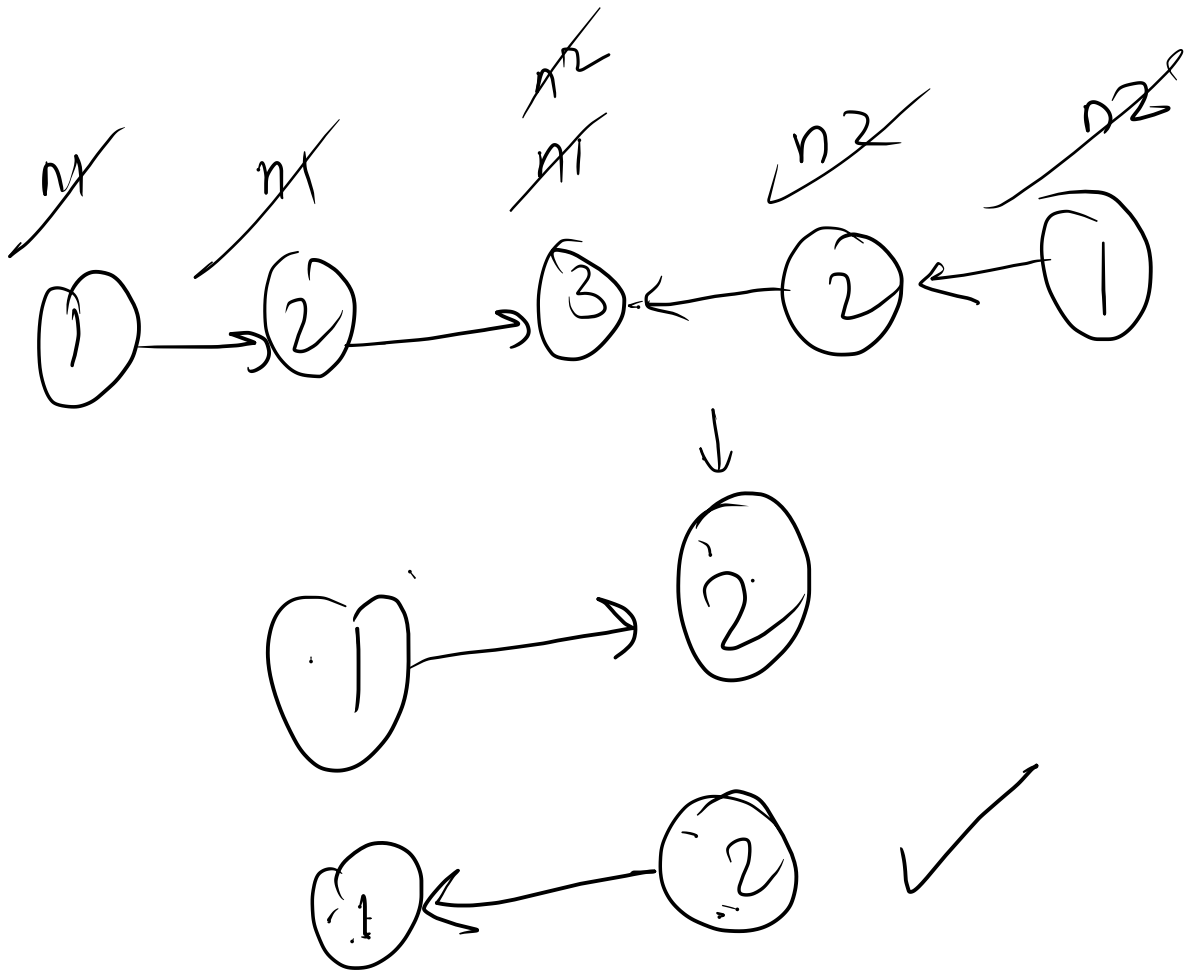
e--


head
tail
1
2
3
3
2
1


1
2
3


3
2
1

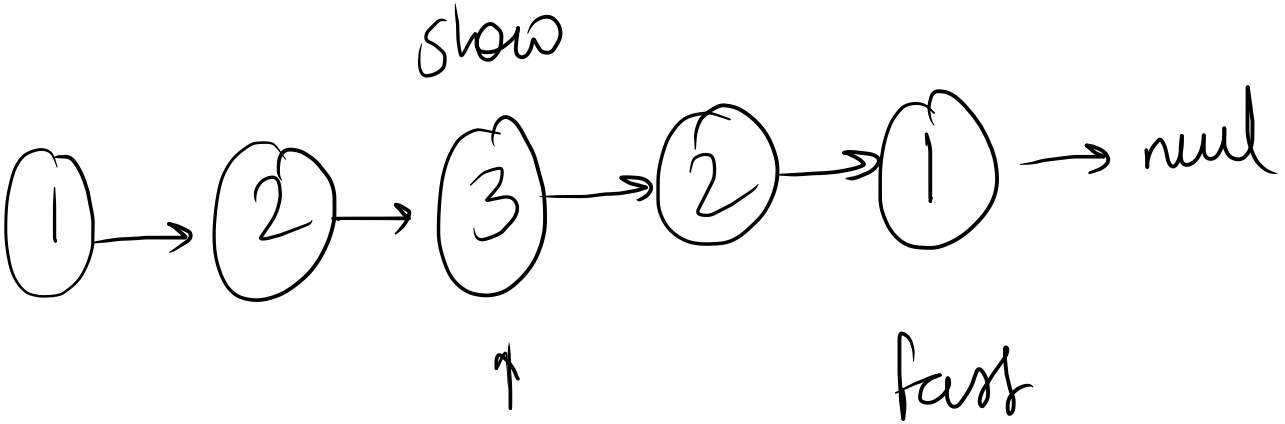

1
3
5
5
3
1

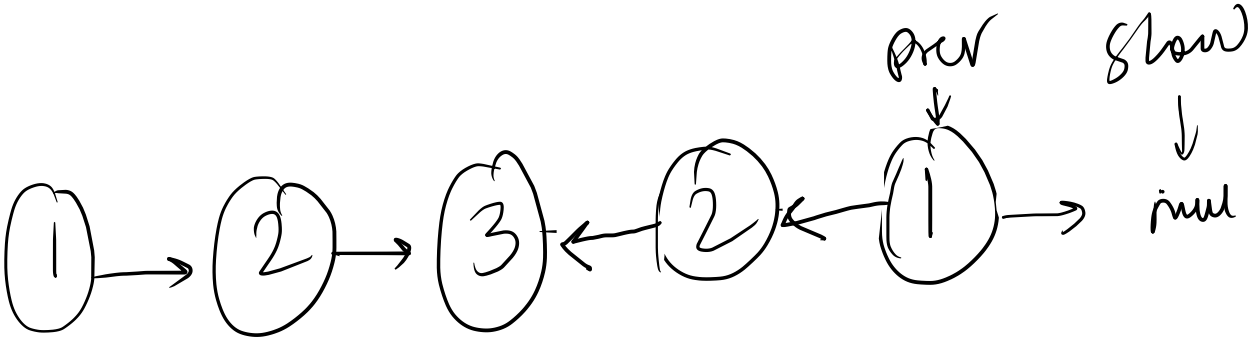

n1
n1
n2
n1
n2
n2
1
2
3
2
1
2
1
2
1
2

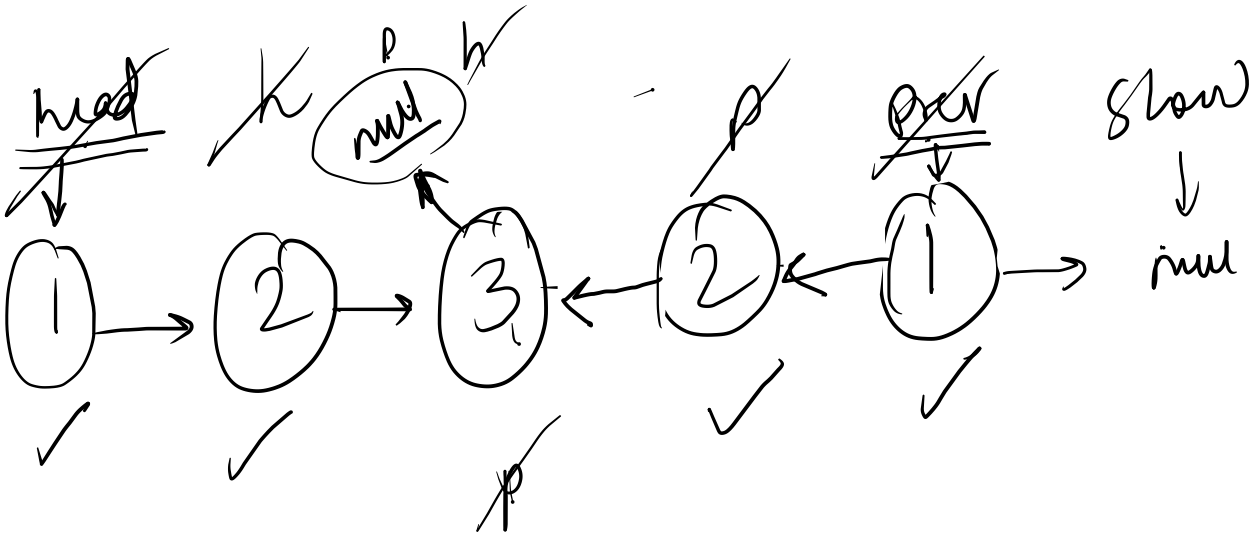
✓ Step1: Find middle

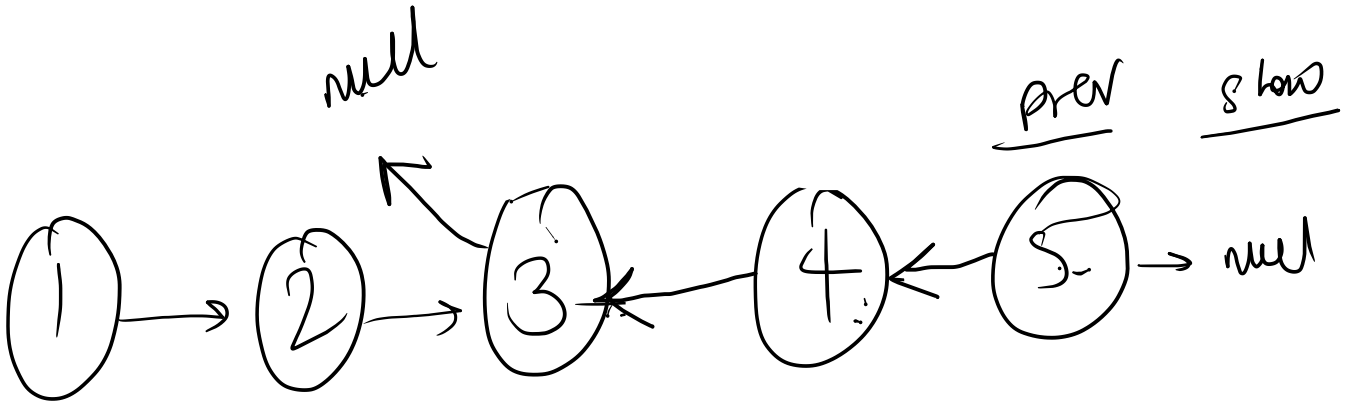
✓ Step2: Reverse second half

Step3: Traverse and check


slow
1
2
3
2
1
null
fast


prev
slow
1
2
3
2
1
null

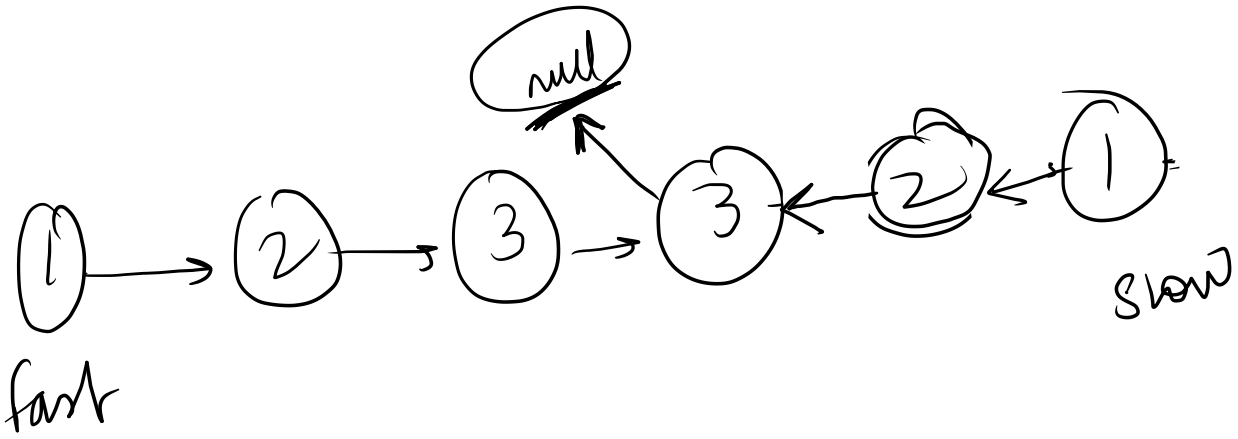

head
null
prev
slow
null
1
2
3
2
1



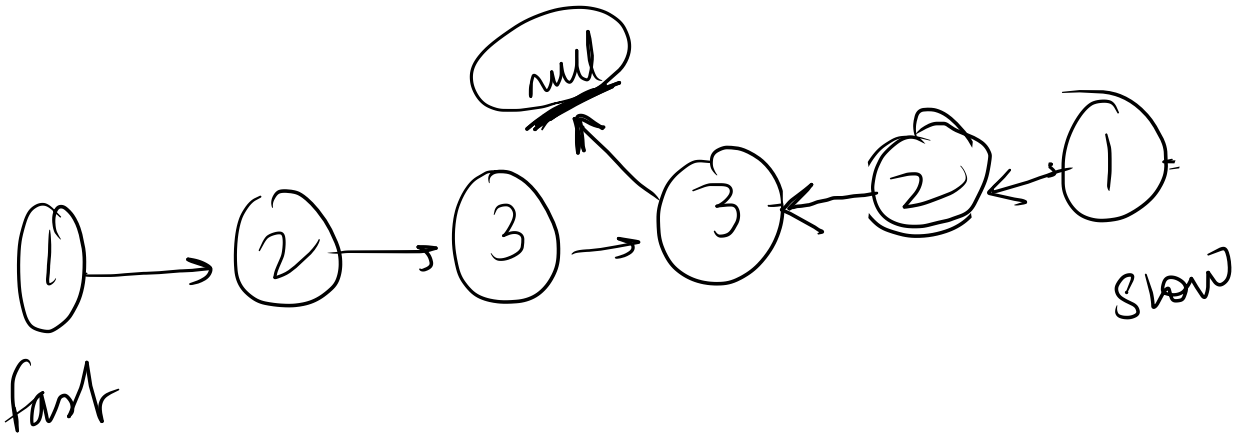
prev . next = null

temp = slow . next

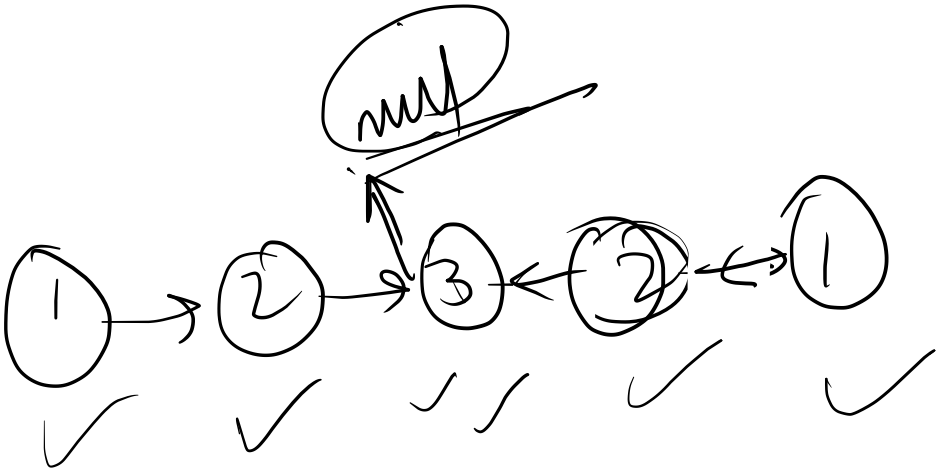
Slow . next = prev



1. temp = slow.next
2. slow.next = prev
3. prev = slow
4. slow = temp



1. temp = slow.next
2. slow.next = prev
3. prev = slow
4. slow = temp

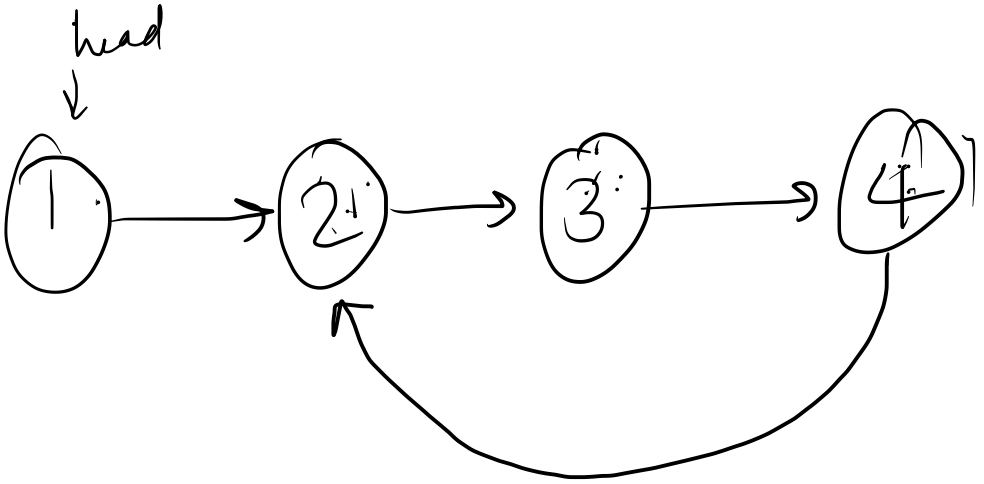

null
1
2
3
2
1

$TC = O(N)$

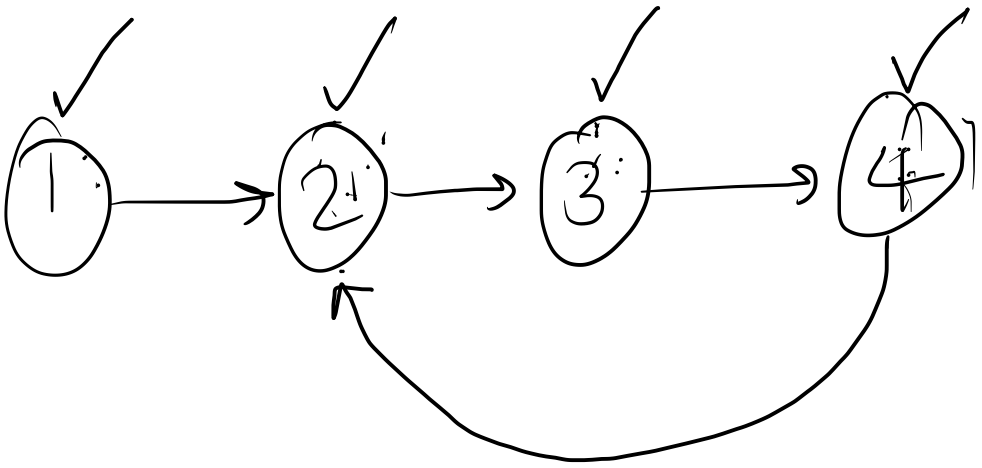
$Aux\ space = O(1)$

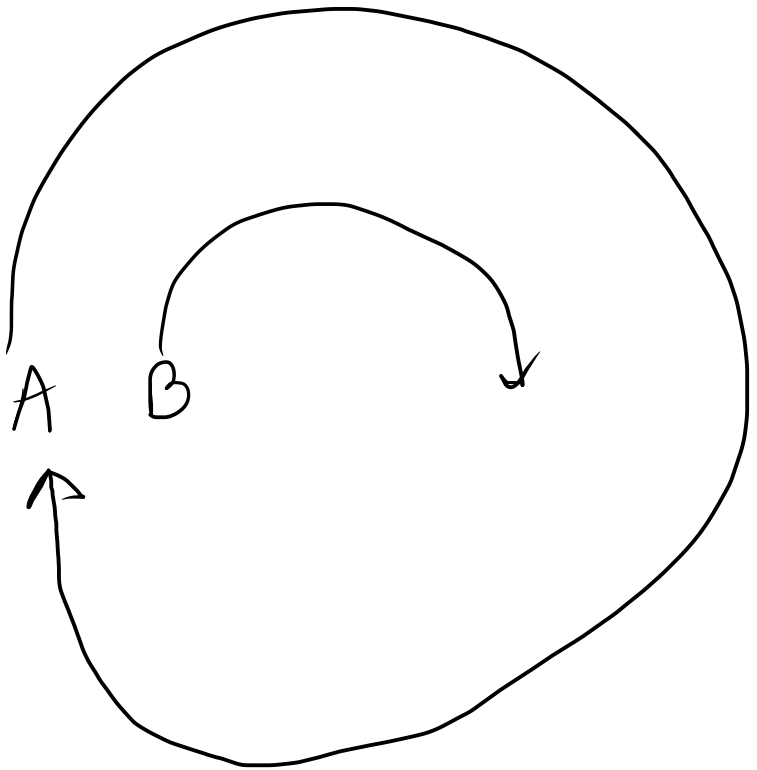
$TC = O(N)$

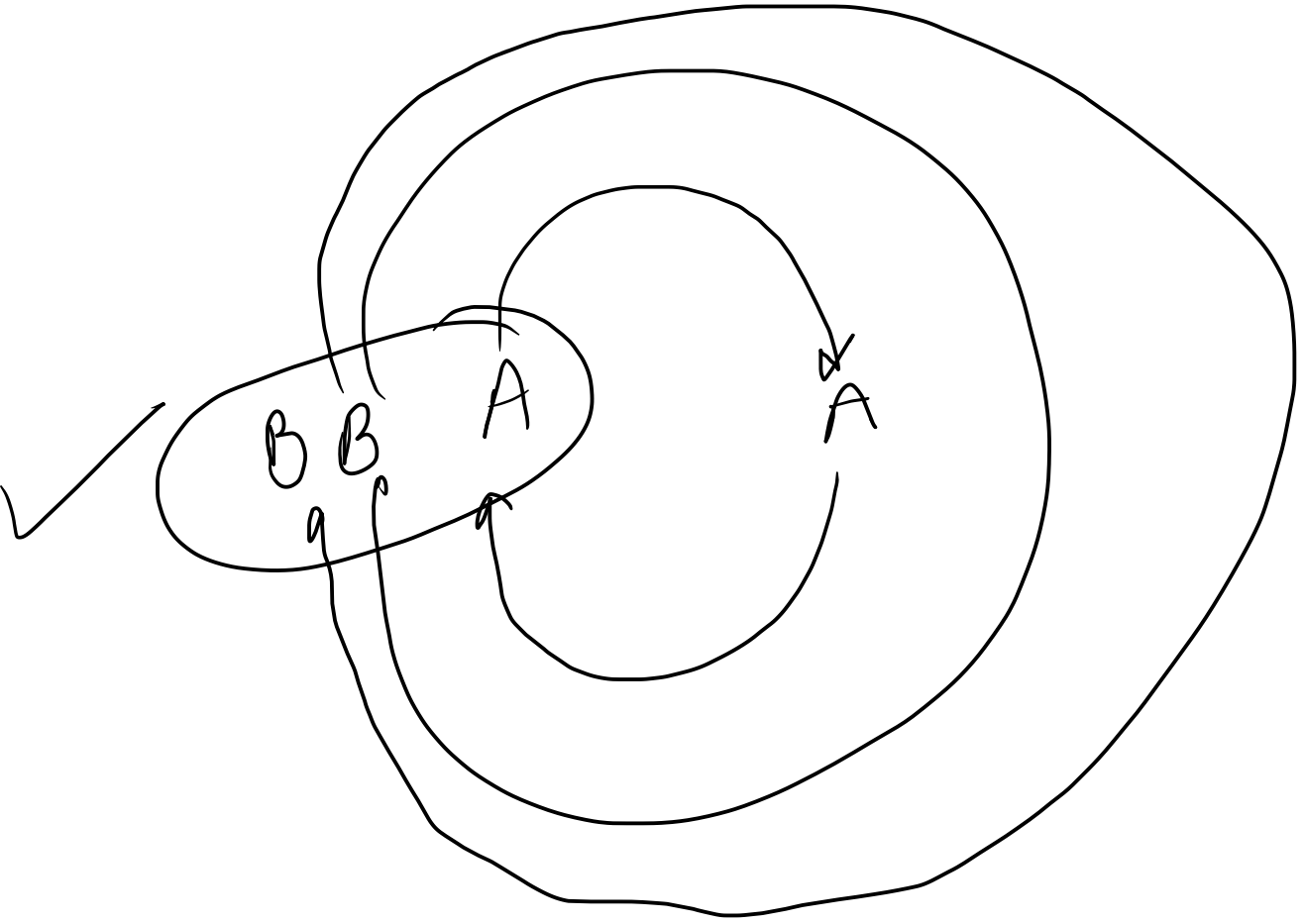
Aux space $= O(1)$



cycle in LL

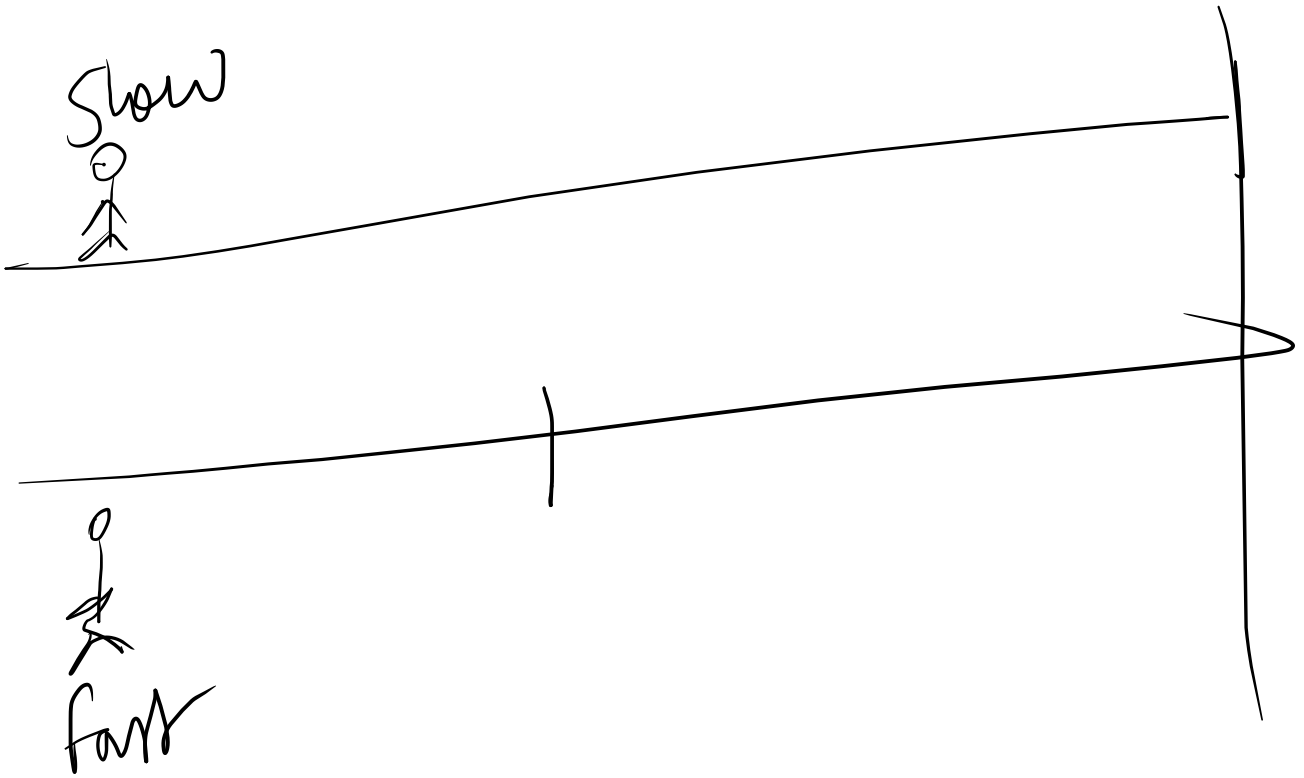

1
2
3
4


A
B


B
B
A
A

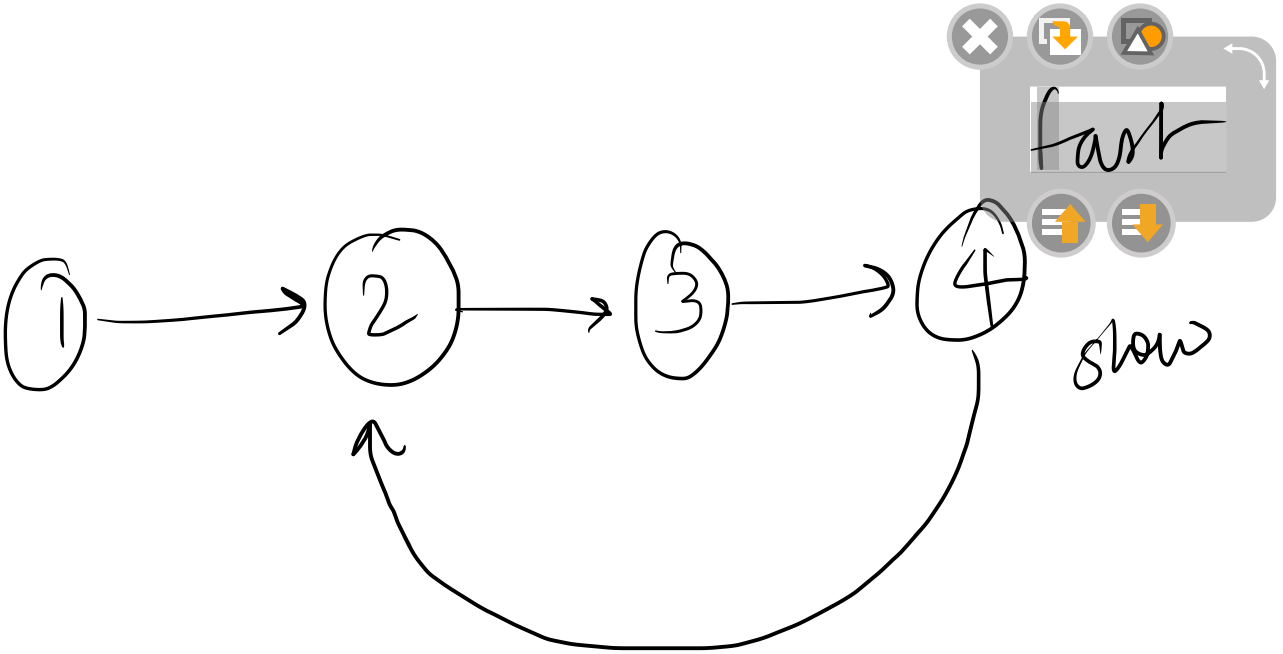
Slow

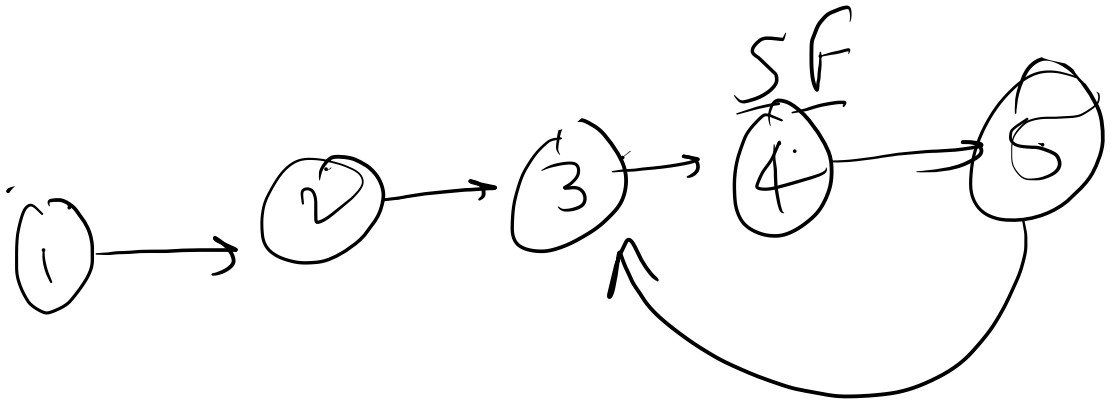
Fast


Slow
Fast

Slow == fast

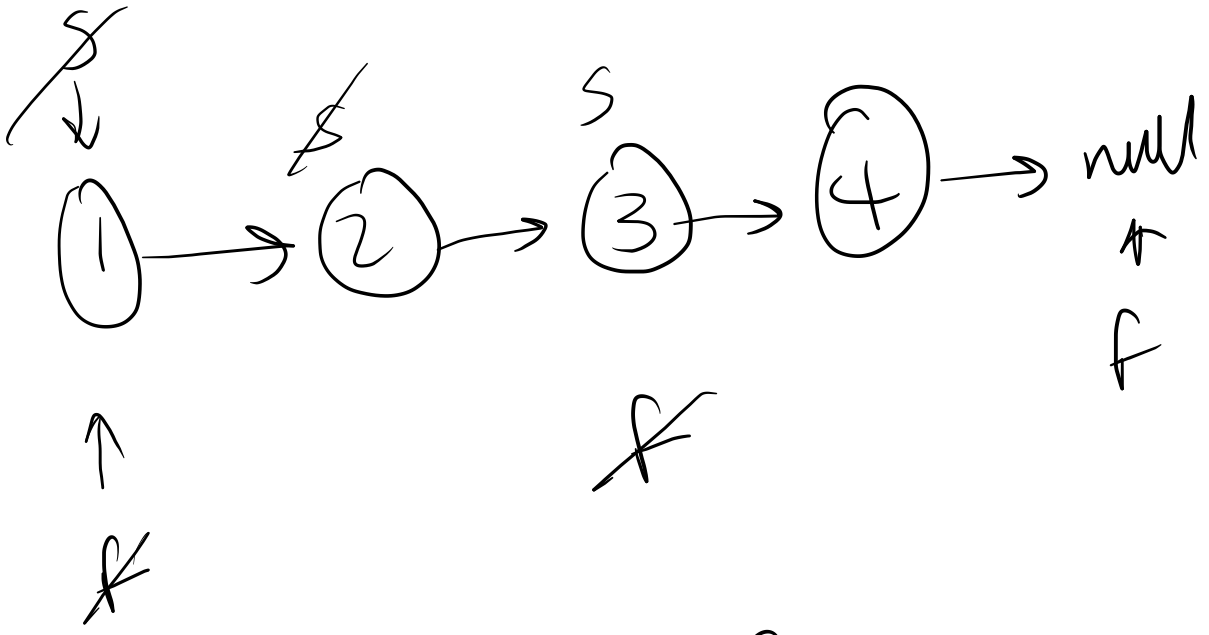
when true


fast
1
2
3
4
slow

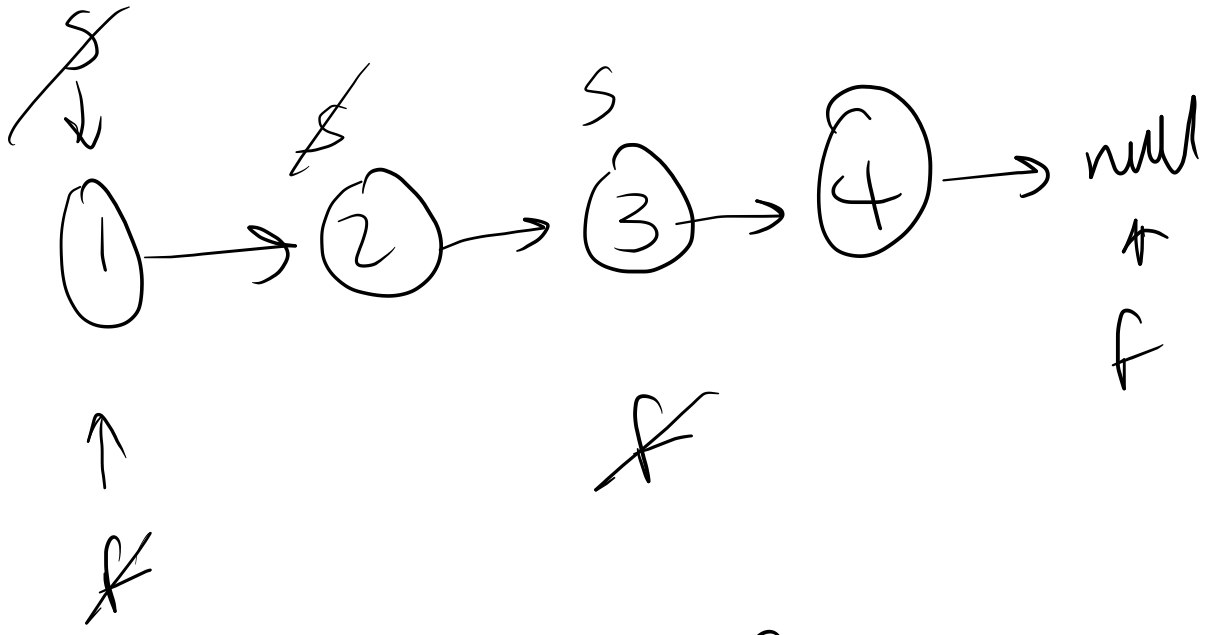


slow == fast

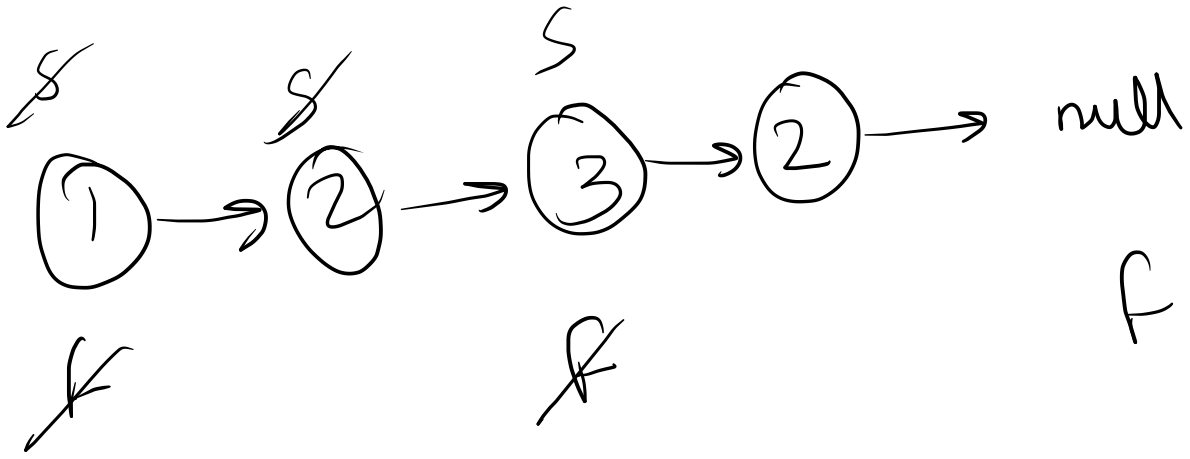
return true

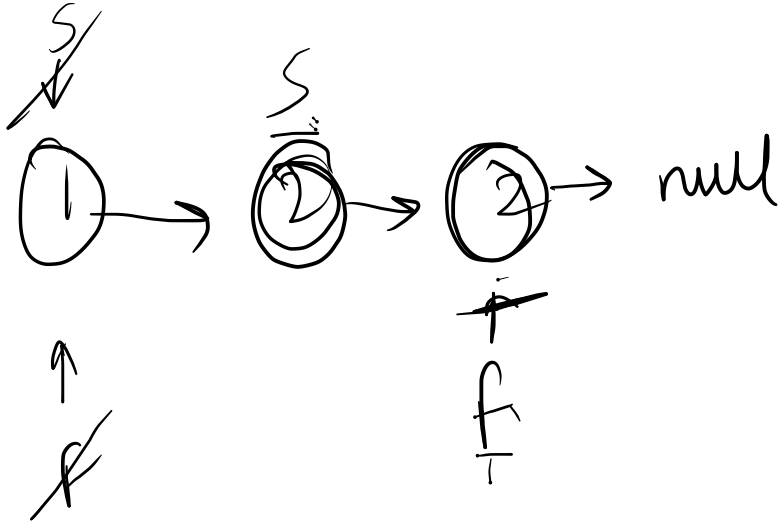


return false



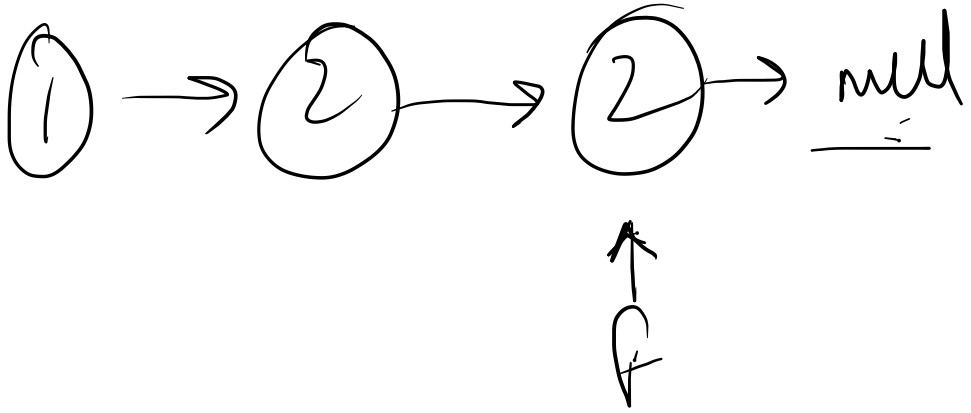
return false


s
s
s
1
2
3
2
null
f
f
f


S
S
1
2
2
null
f
f

$TC = O(n)$

Aux Space $= O(1)$


1
2
2
null
f