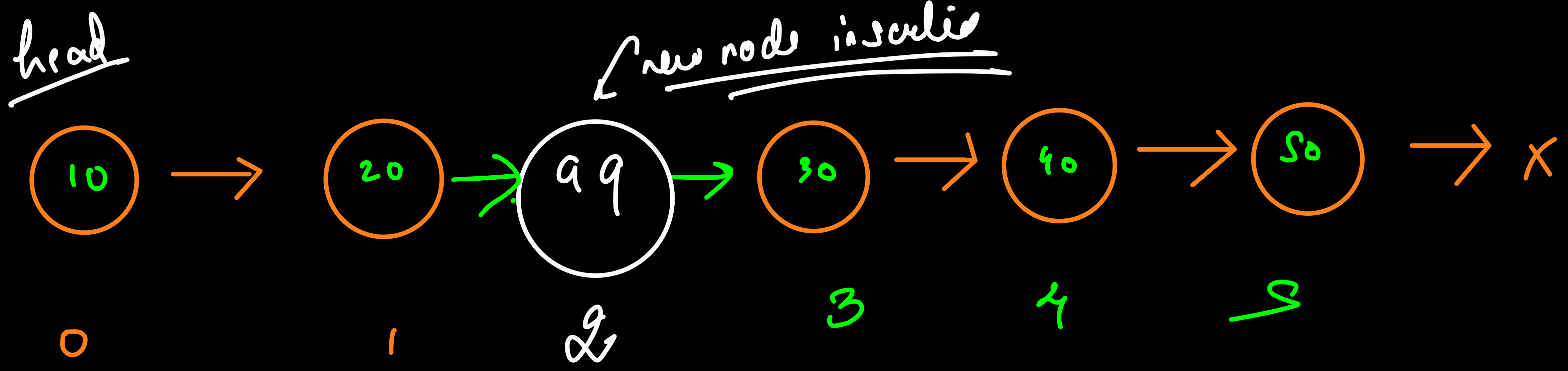
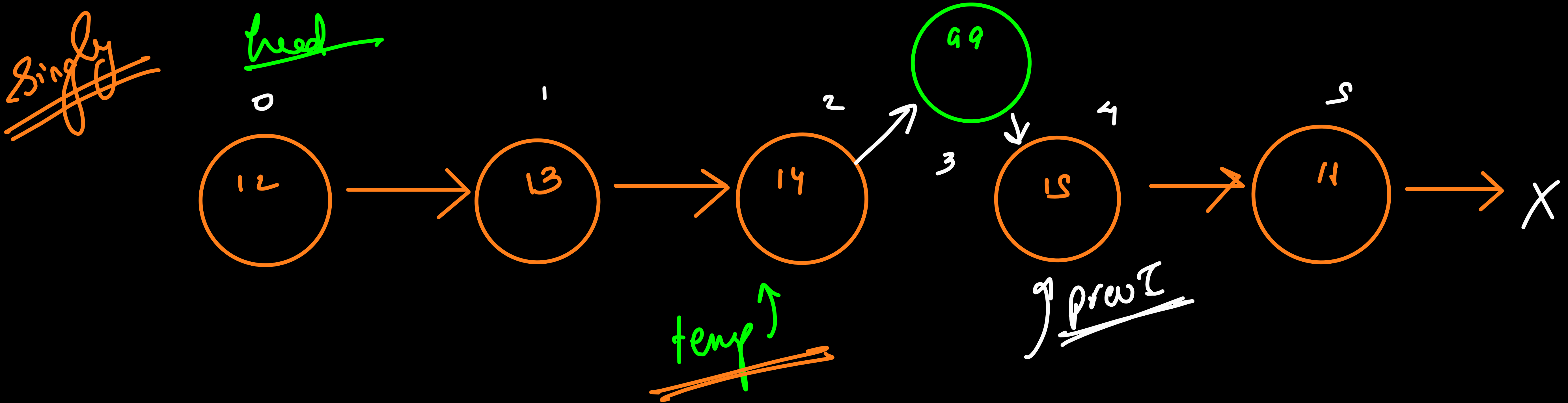


addAt (head, i, data)

addAt (head, 2, 99)



count = ~~0~~ ~~1~~ 2

loop ends here → while (count < (i-1))

find the node at (i-1)th index

i   data

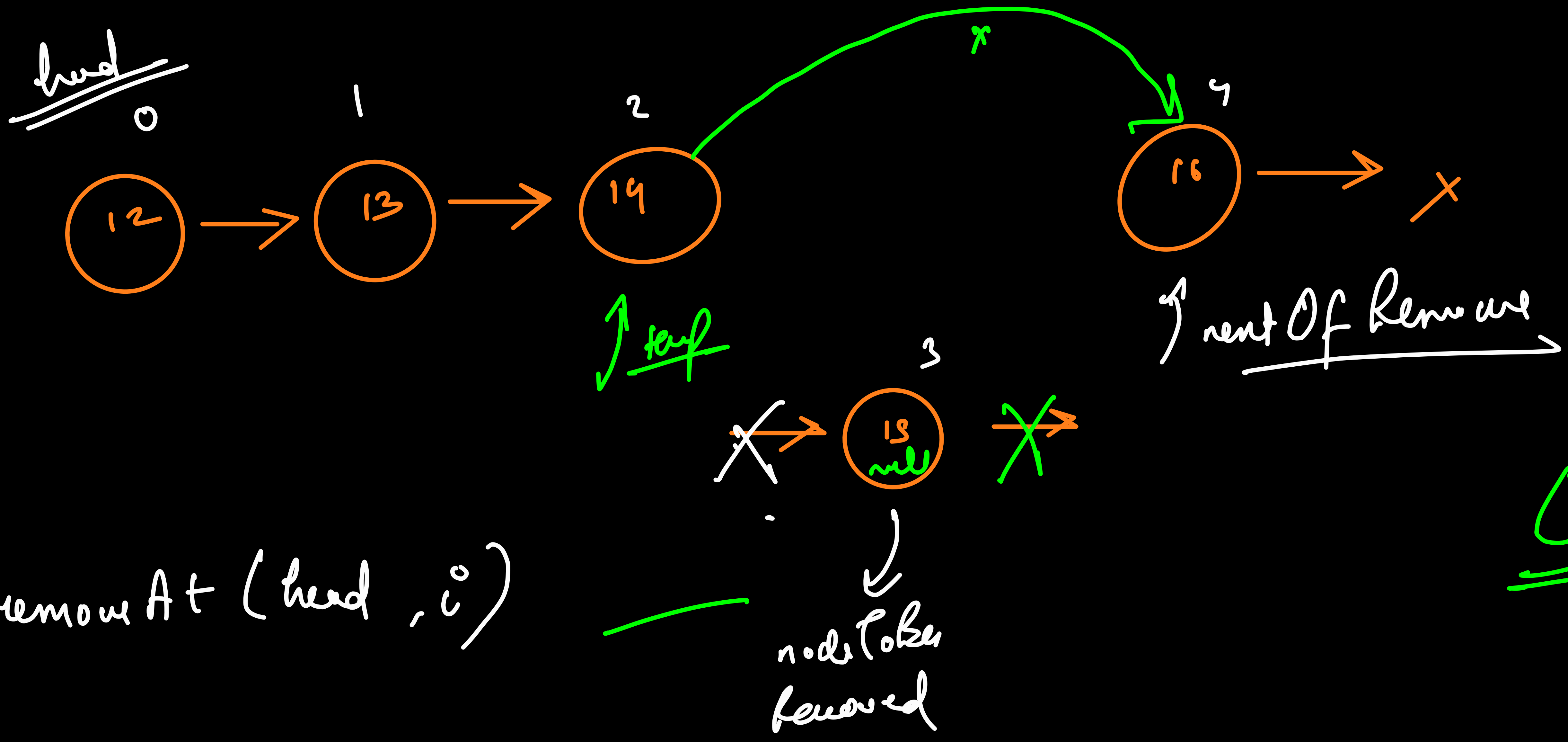
addAt (3, 45);

newNode = createNode (99)

prevI = temp.next

temp.next = newNode

newNode.next = prevI



removeAt(head, i)

removeAt(head, 3)

O(n)

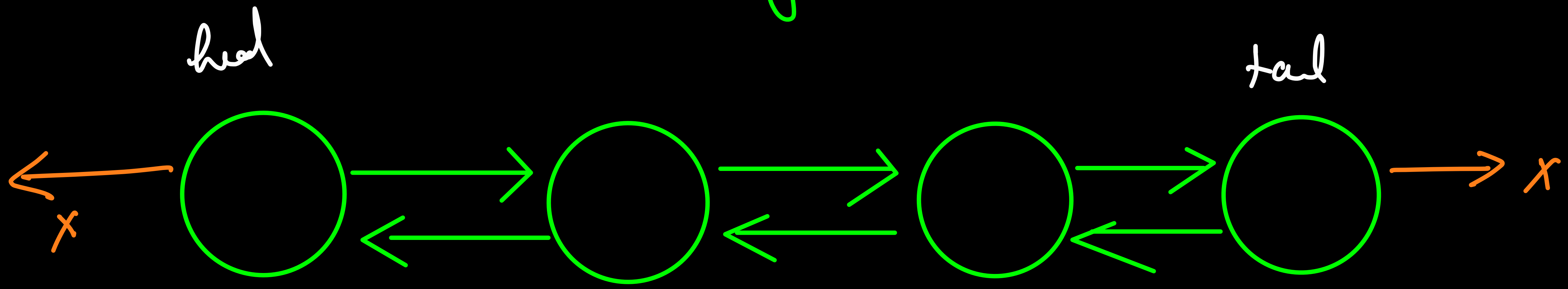
nodeToBeRemoved = temp.next.

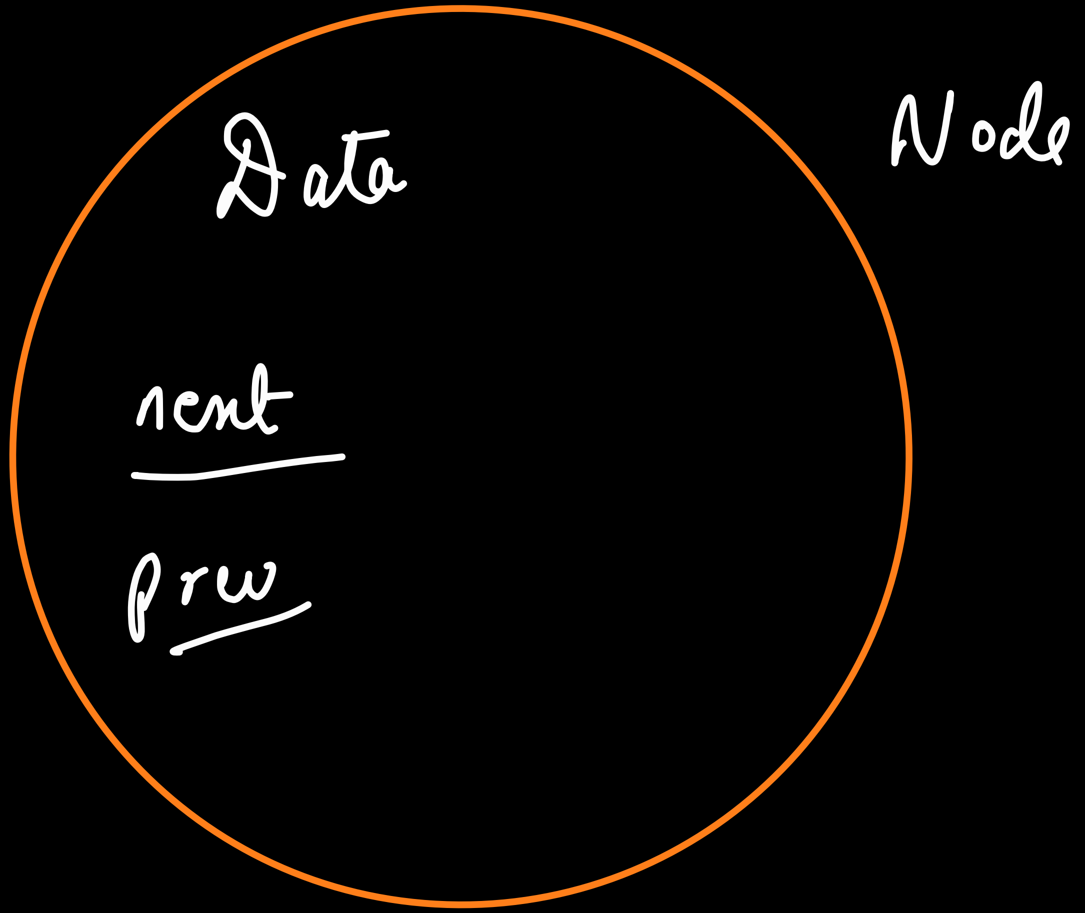
nextOfRemoved = temp.next.next;

temp.next = nextOfRemoved

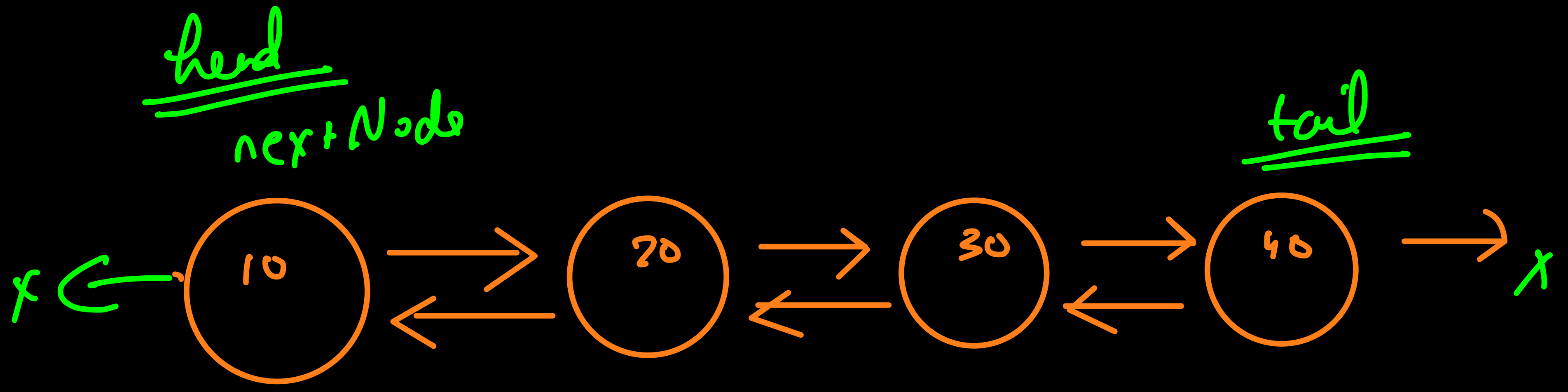
nodeToBeRemoved.next = null;

Doubly Linked List





add At head
remove At Head
add At tail
remove At Tail



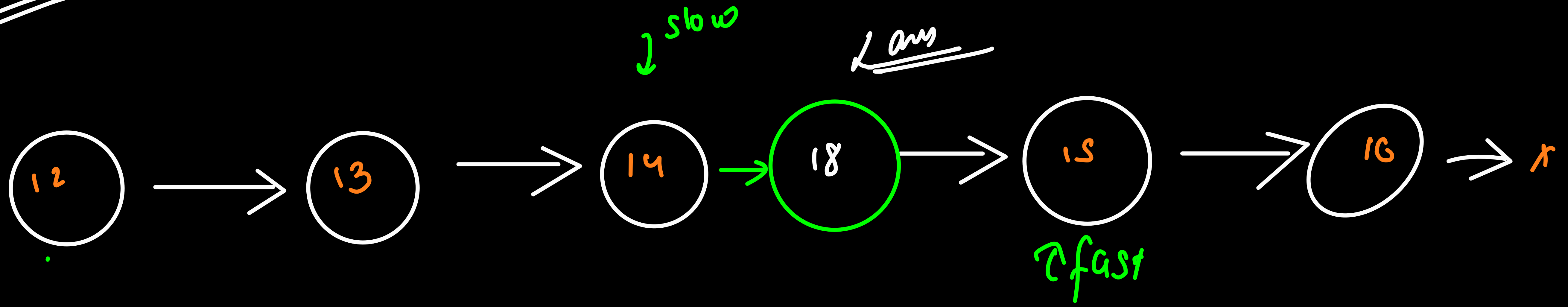
4

head == null

```
newNode.next = head
head.prev = newNode
head = newNode
```

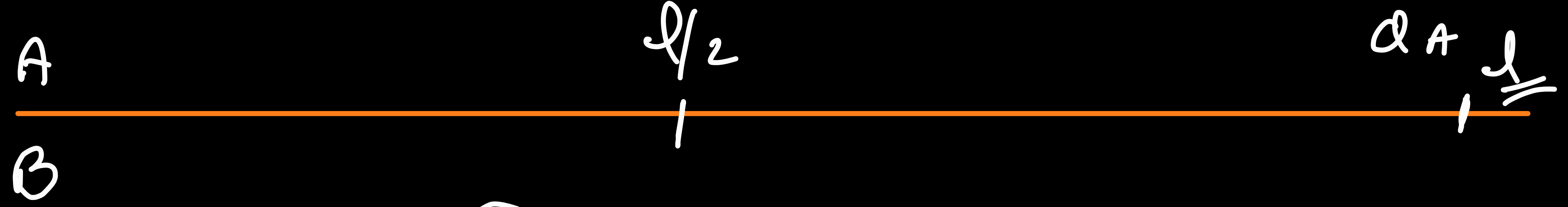
```
nextNode = head.next
head.next = null;
nextNode.prev = null
head = nextNode;
```

hare & rabbit



length → l middle → $l/2$

$$d_B = 2x \times t \quad \text{—} \ (1)$$
$$d_A = x \times t \quad \text{—} \ (2)$$
$$\frac{d_B}{d_A} = \frac{2}{1}$$



A → x m/s $\boxed{B}$ → $2x$ m/s

$$\boxed{\frac{d_B}{2} = d_A}$$

If both runs for t unit of time and B completes the track in t unit of time,

$$S = \frac{d}{t} \rightarrow d = S \times t$$

→ O(N) O(1)

```
slow= head
fast= head
while (fast.next != null  and  fast.next.next (=null){
        slow = slow.next ;
        fast= fast.next.next;
}
if (fast.next == null)  → odd length ll
      return slow
else
      return slow.next;
```