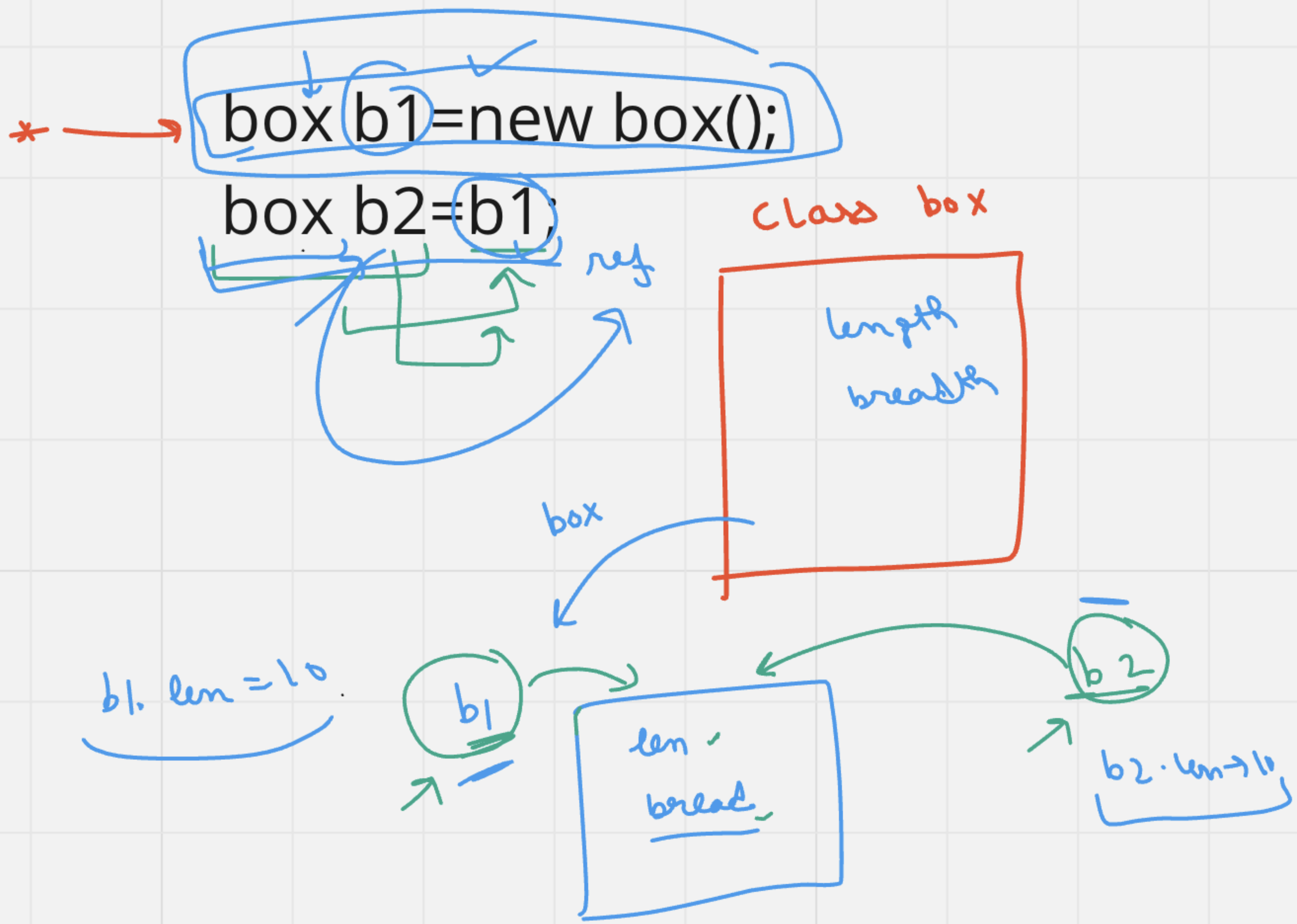
```
box b1=new box();
box b2=b1;
```



Linked List

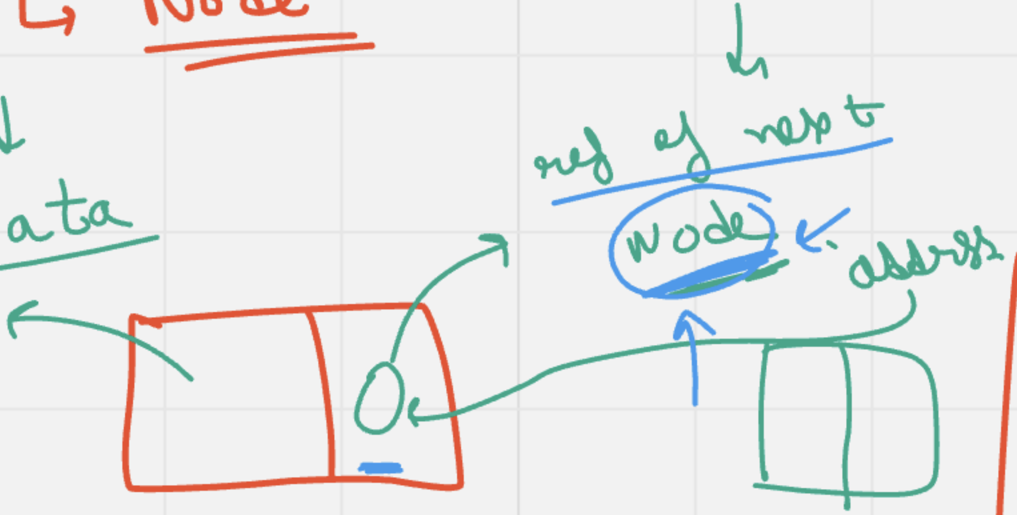
↳ Node's

↓

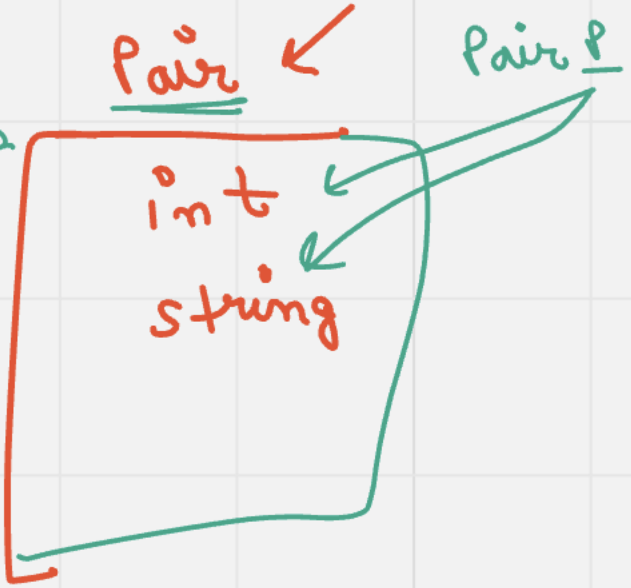
int data

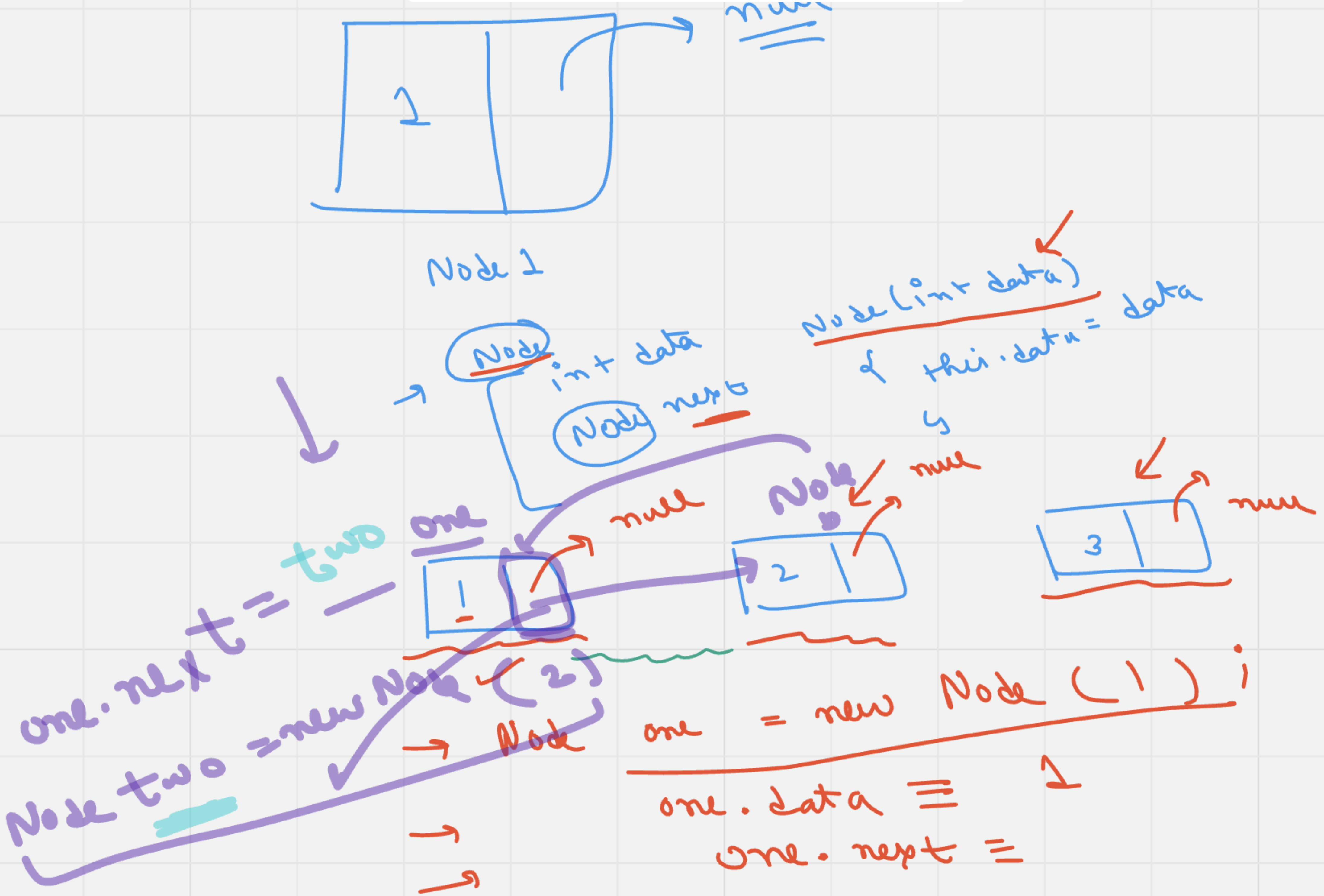
class Node

```
int data

Node next
```

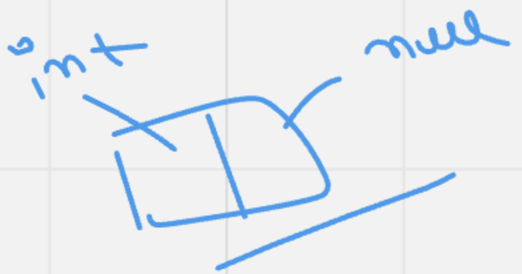


Node

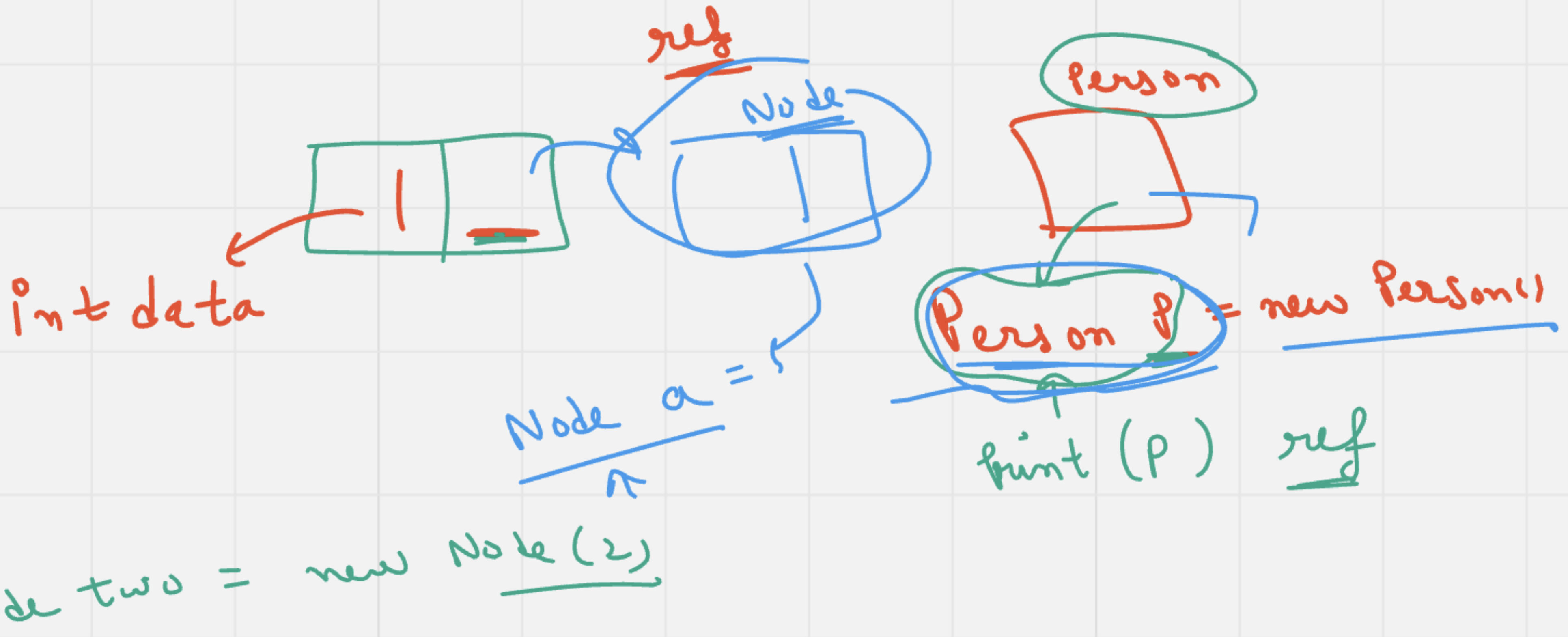


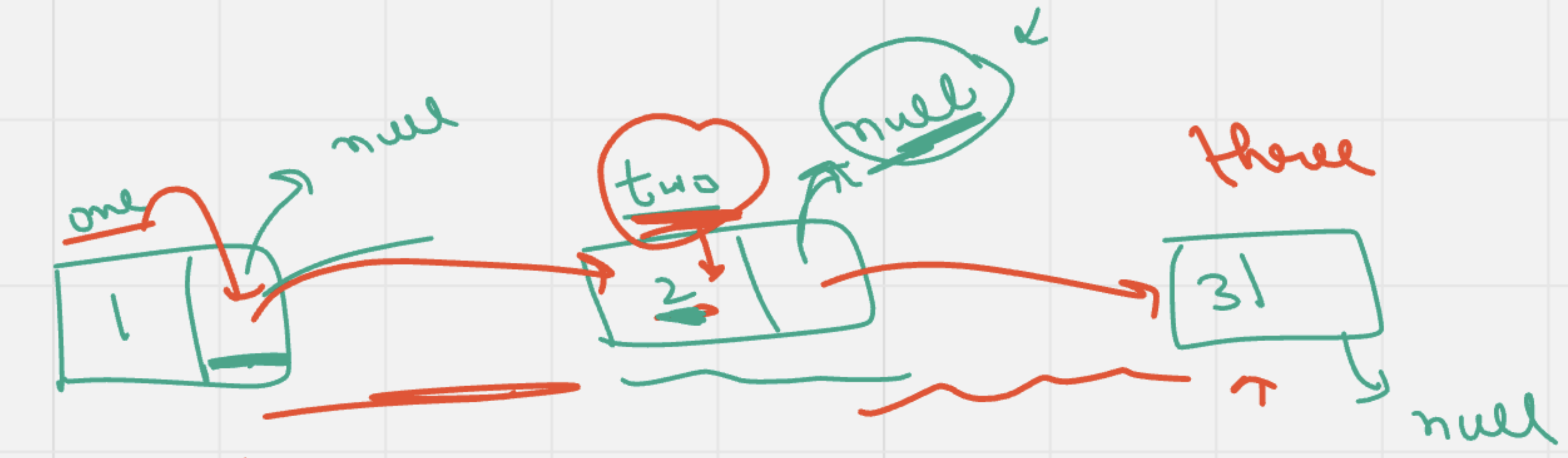


```
Node(int data)
{ this.data = data
}

Node two = new Node(2)
one.next = two one

one = new Node(1);
one.data = 1
one.next =
class Node {
  int data;
  Node next;   → null

  public Node(int data) {
     this.data = data;
  }
}
```



new Node use ref





one.next = two

one.data

1

one.next.data

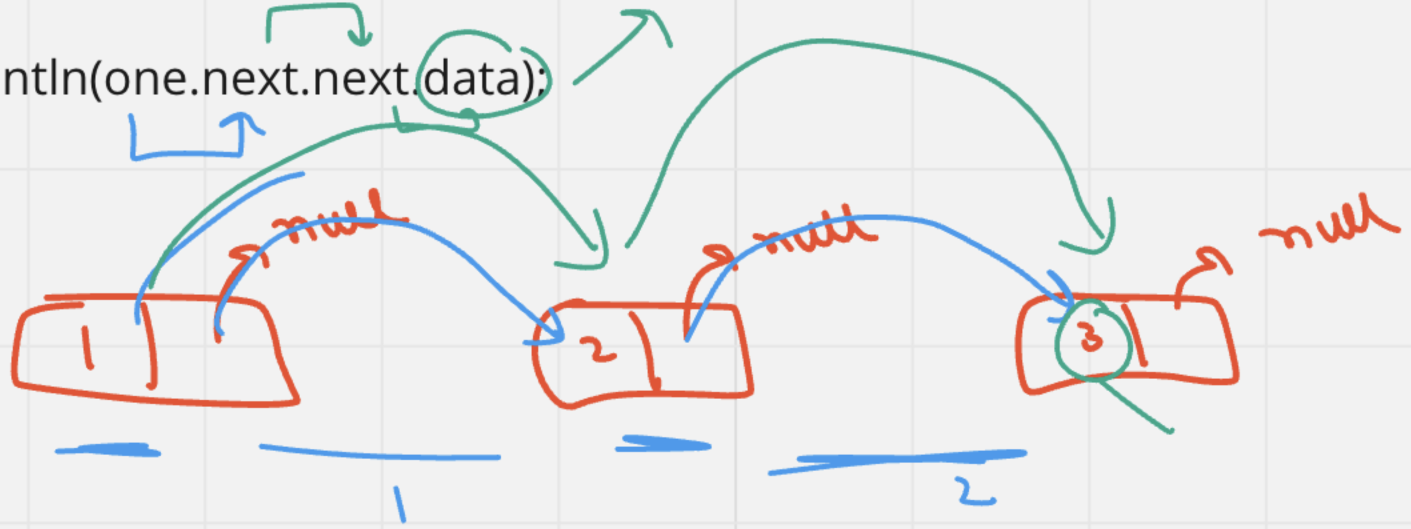
2 ← null.data

null Pointer exception

one.next.next.data

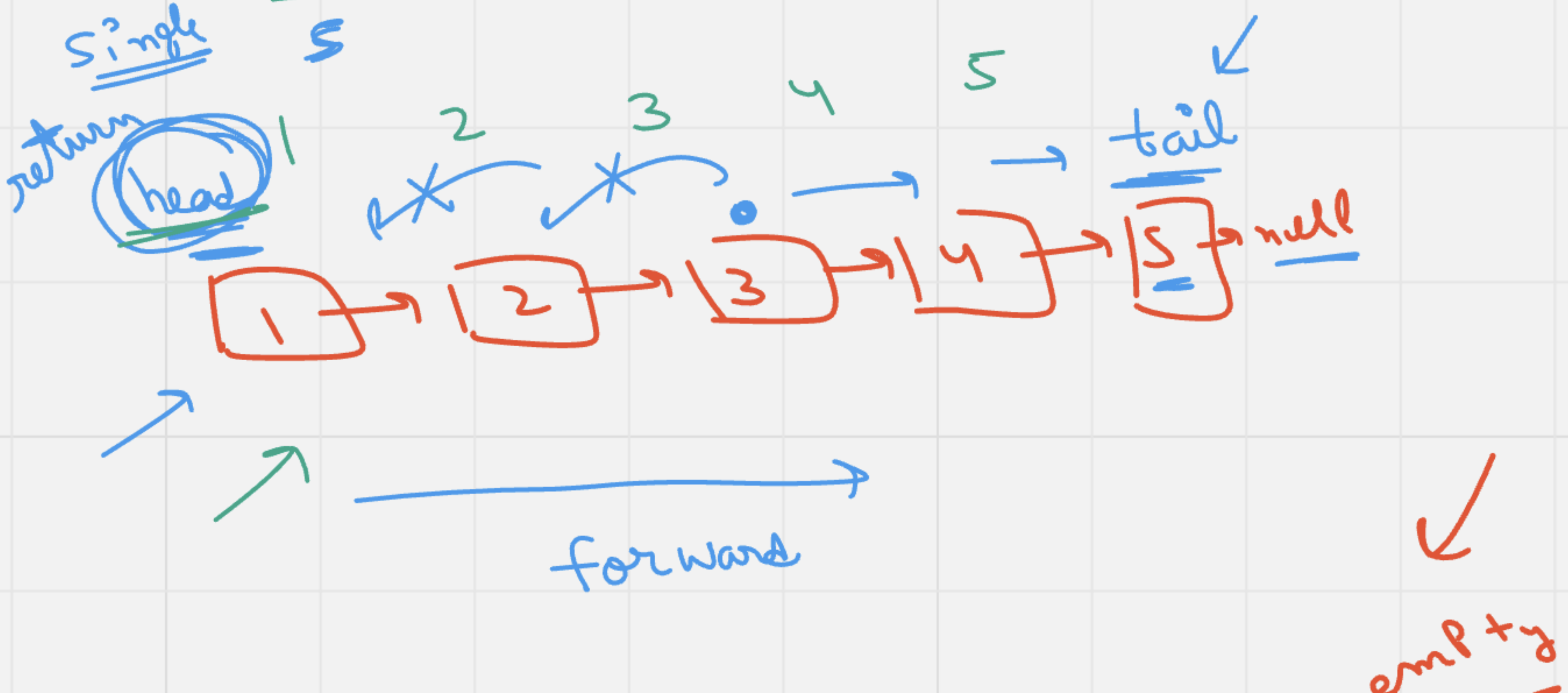
3

```java
Node one = new Node(1);
      Node two = new Node(2);
      Node three = new Node(3);

      one.next = two;
      two.next = three;

      System.out.println(one.next.next.data);
```



3

Take user input



I LL = empty

II LL = not empty

st
head == null
head = newNode
temp = head

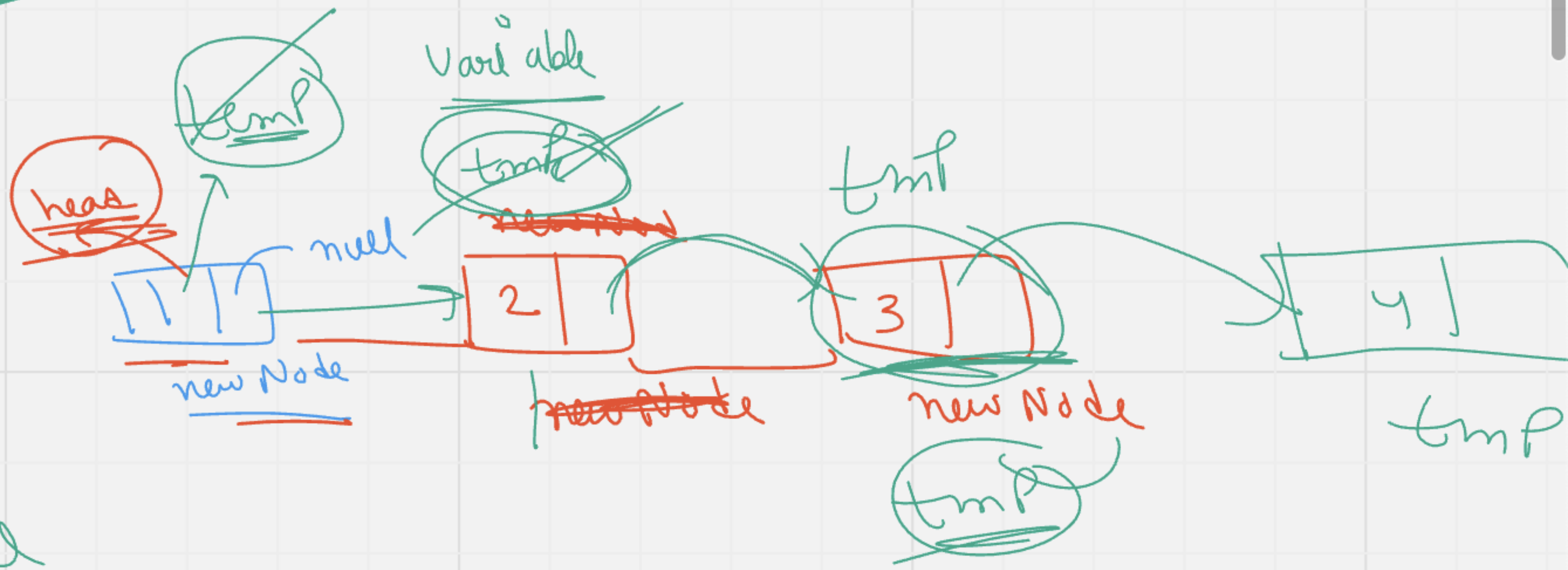
else
~~temp~~
~~head~~ .next = newNode
temp = temp.next

① Node newNode = new Node(~~2~~3);

Node head = null

1 2 3 4

Node temp = null



Node head = null

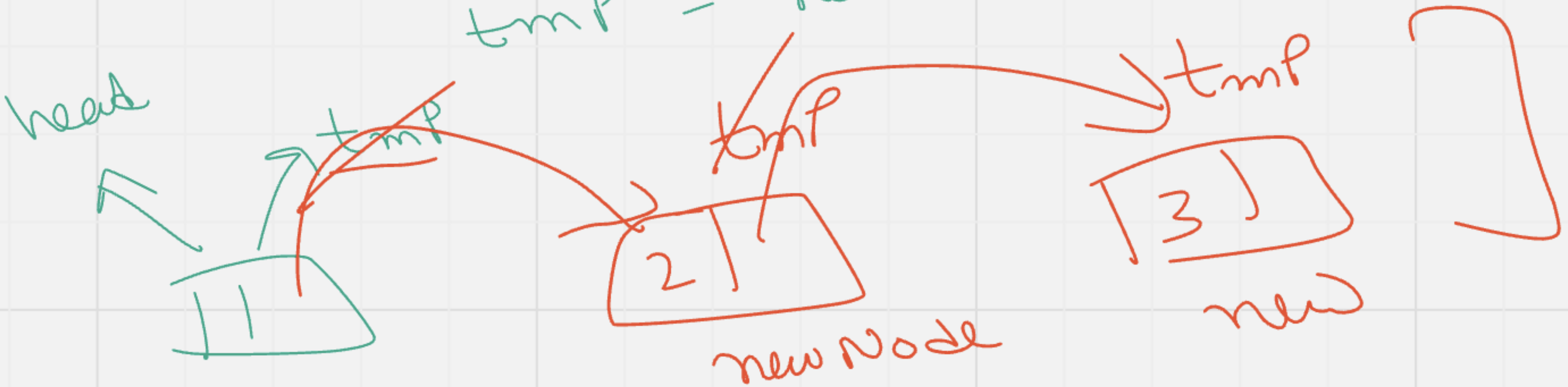
Node tmp = null

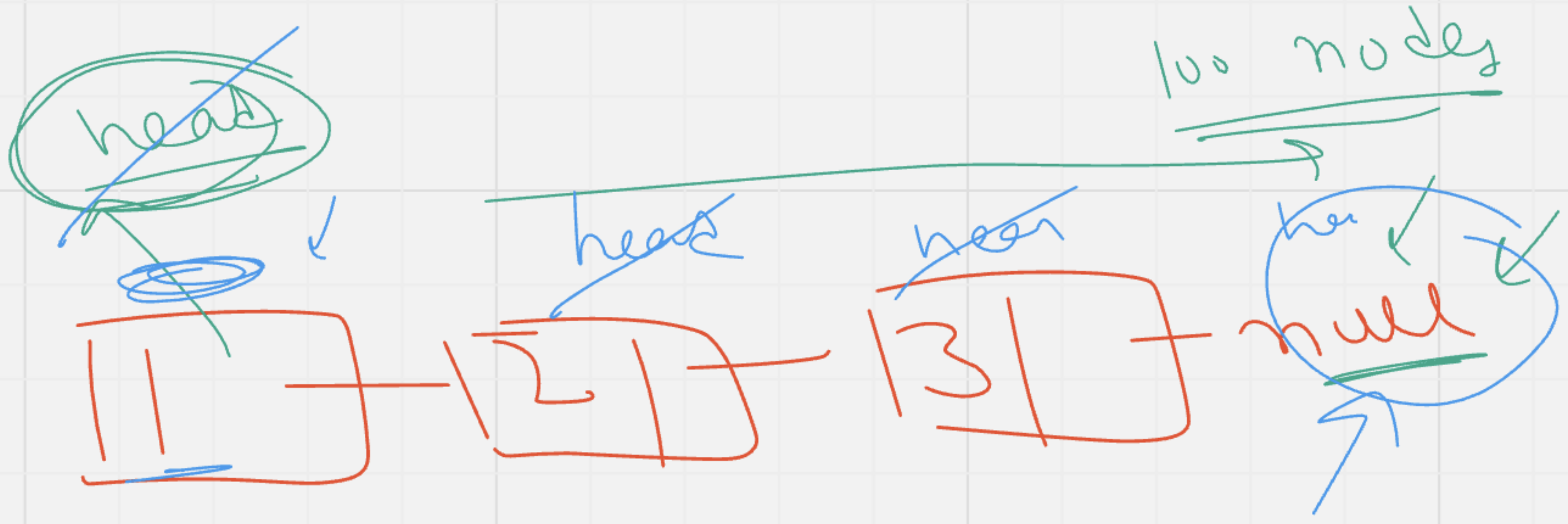
head = newNode

tmp = newNod

tmp.next = nn

temp = temp.nn





stop

Print LL

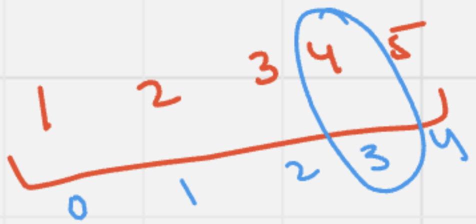
```
while ( head != null )
{
    SOP ( head . data )
    head = head . next
```

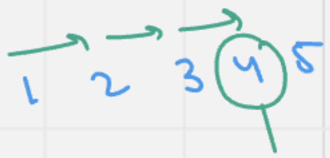
repeat

✓ Arraylist

Linked list

① indexing





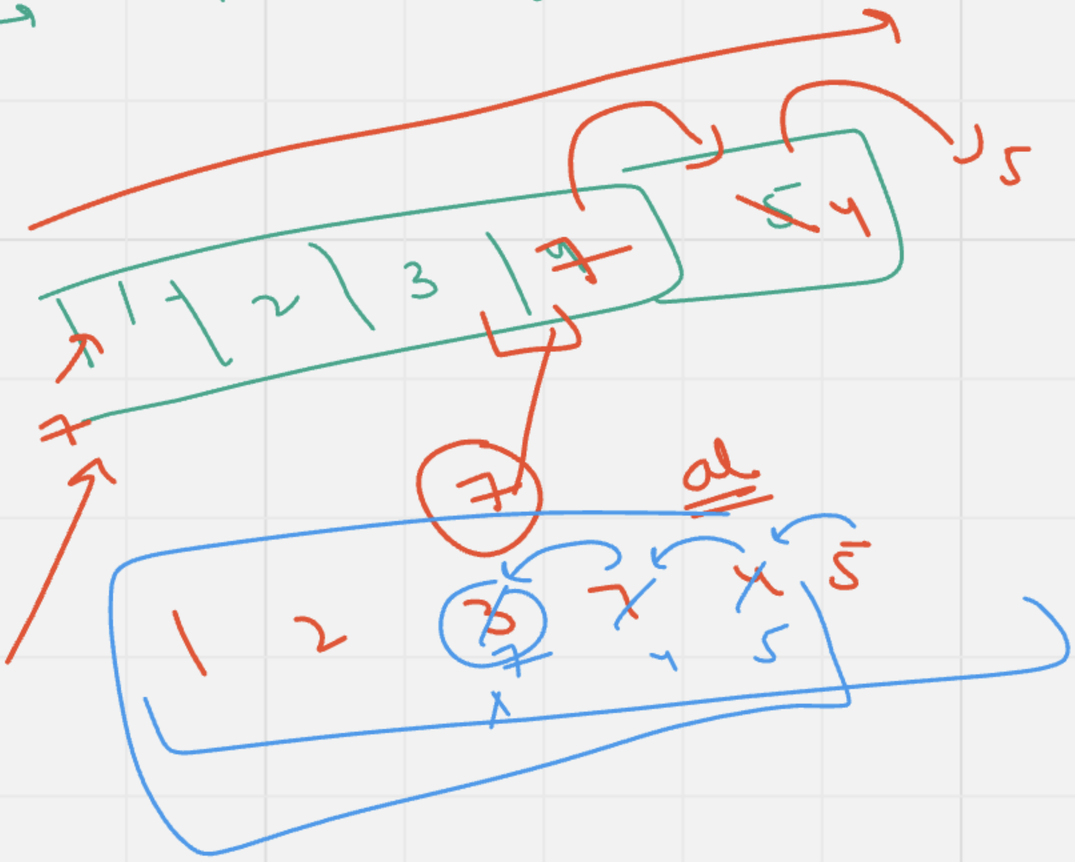
4

O(N)

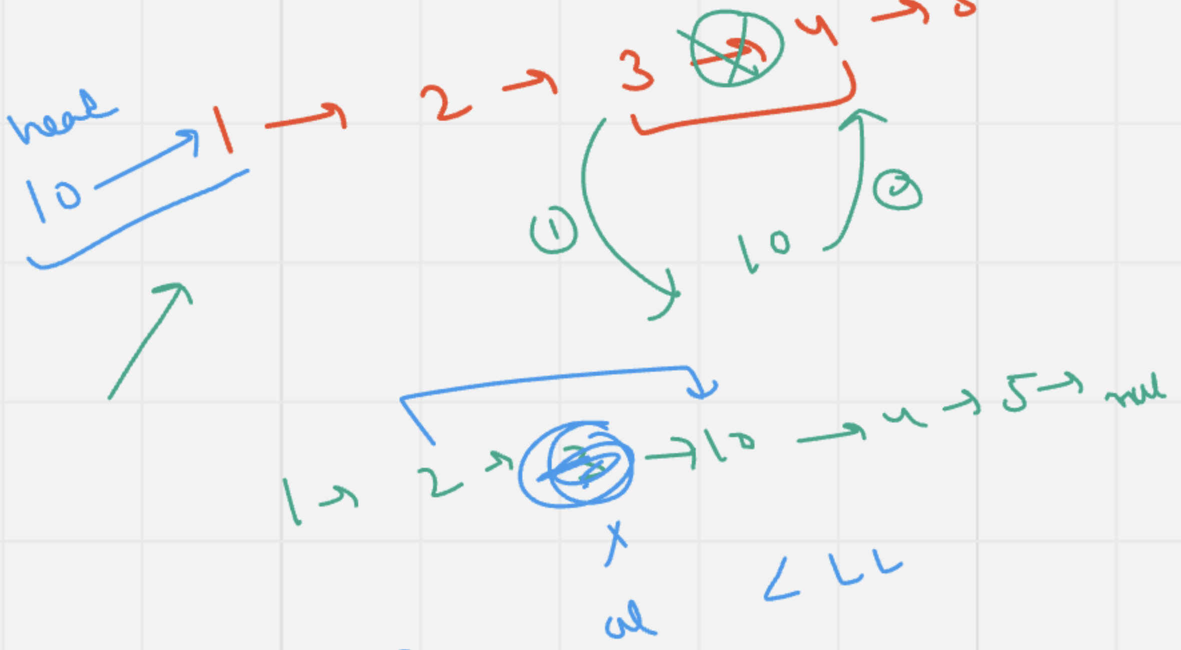
③ O(1)

al fast random access Element

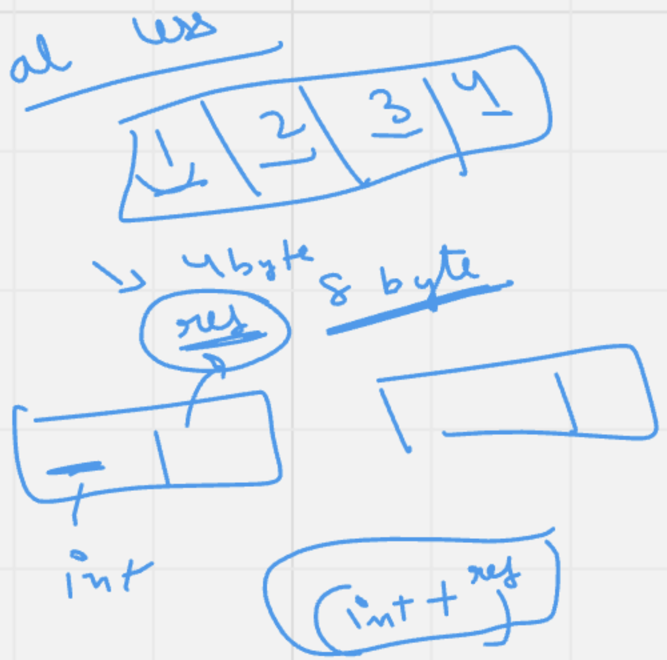
→ insertion | deletion] LL fast



LL



② ① memory



LL more

Performan

user need

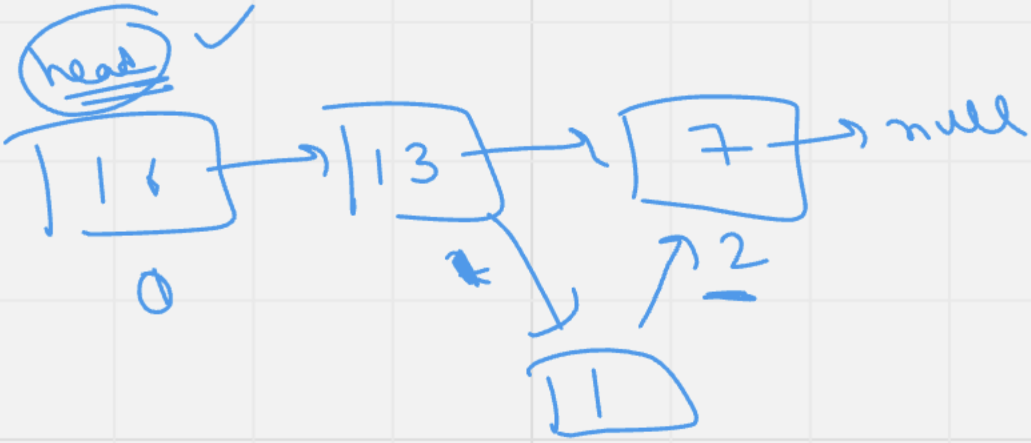
access
more
less ins/ dec

al ✓

acc less
ins/ des

LL

3
16 13 7
1 2

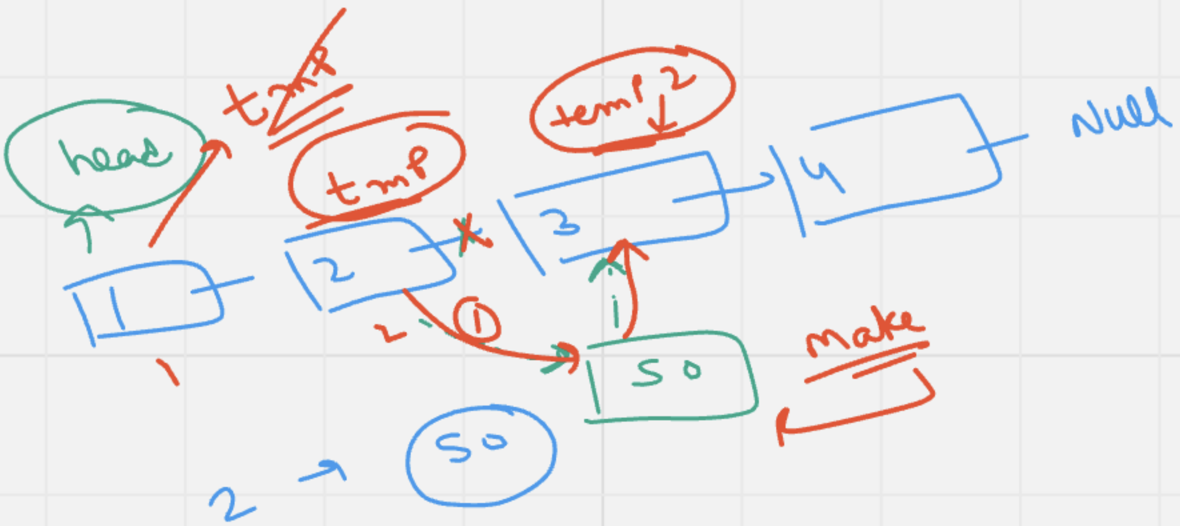


Length
n = 3

Pos = 2

Val = 1

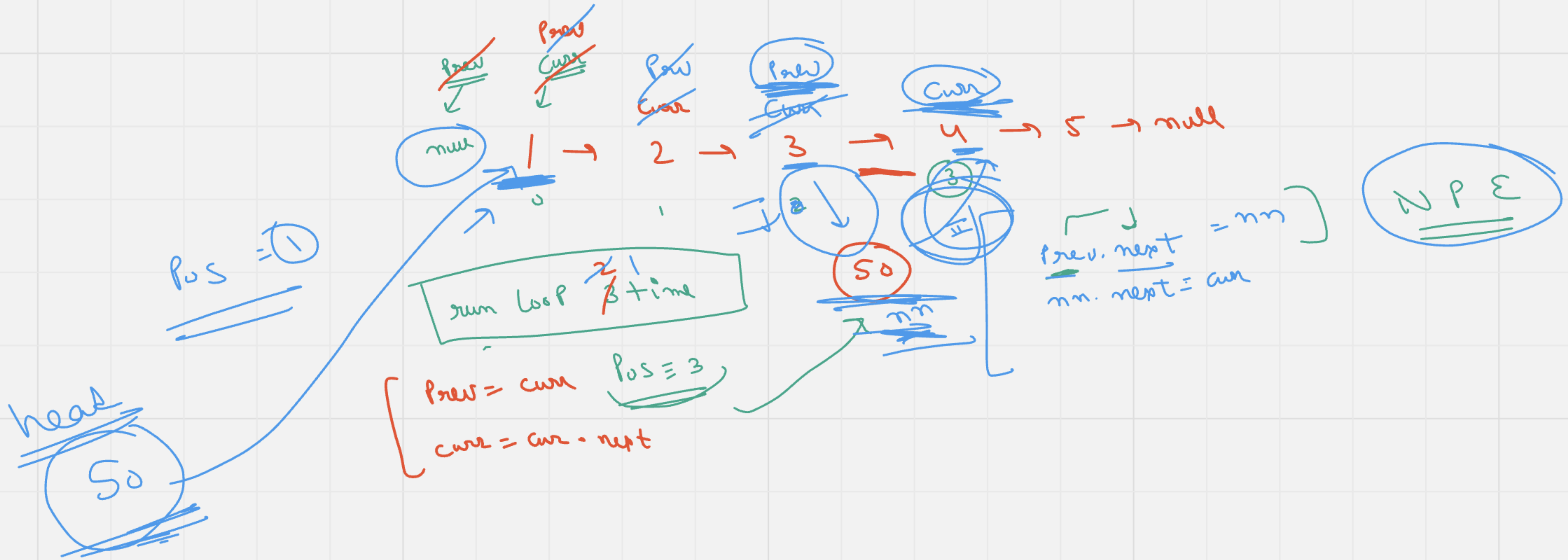
16 → 13 → 1 → 7



2 → 50

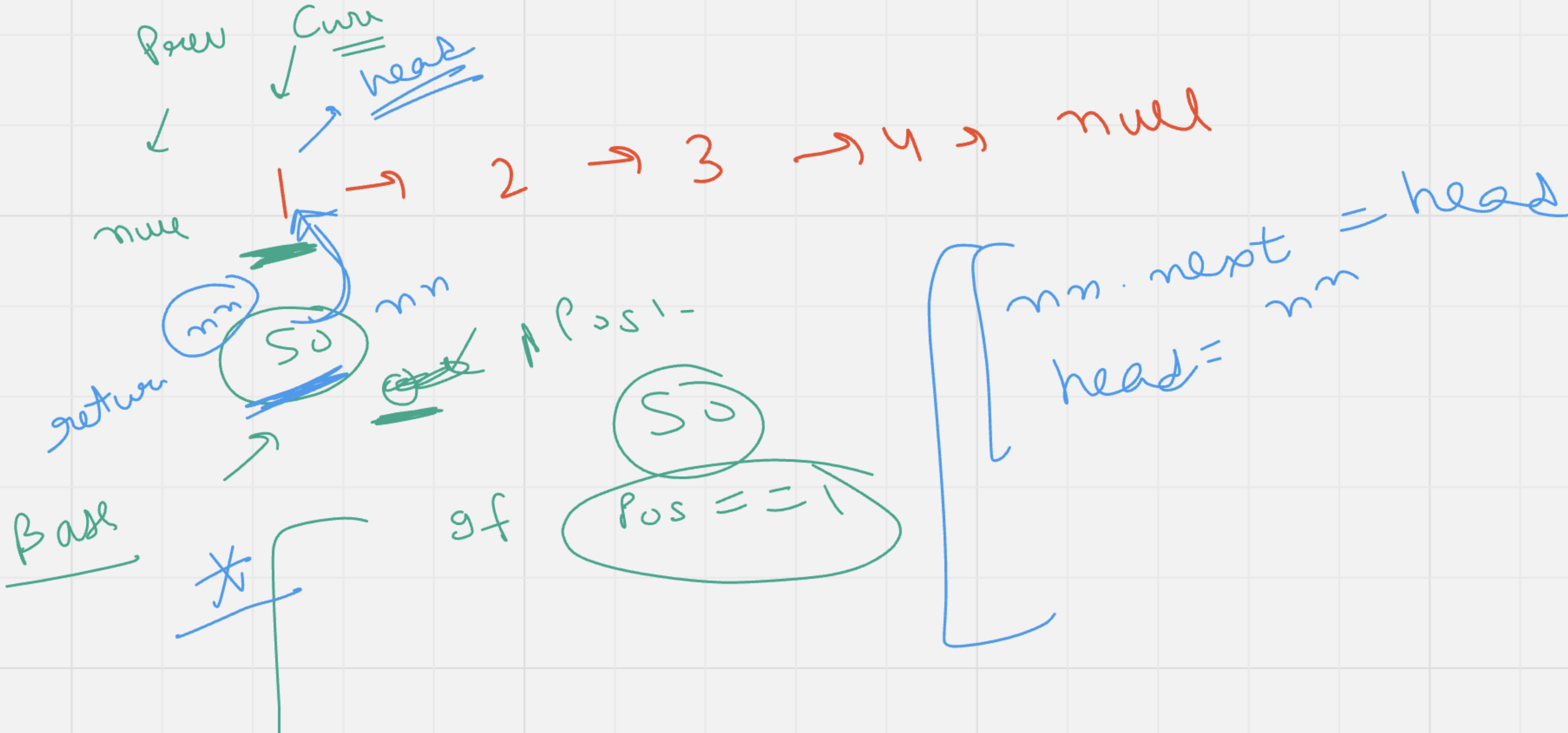
→ { tmp.next = $\boxed{50}$
$\boxed{50}$.next = temp2 }

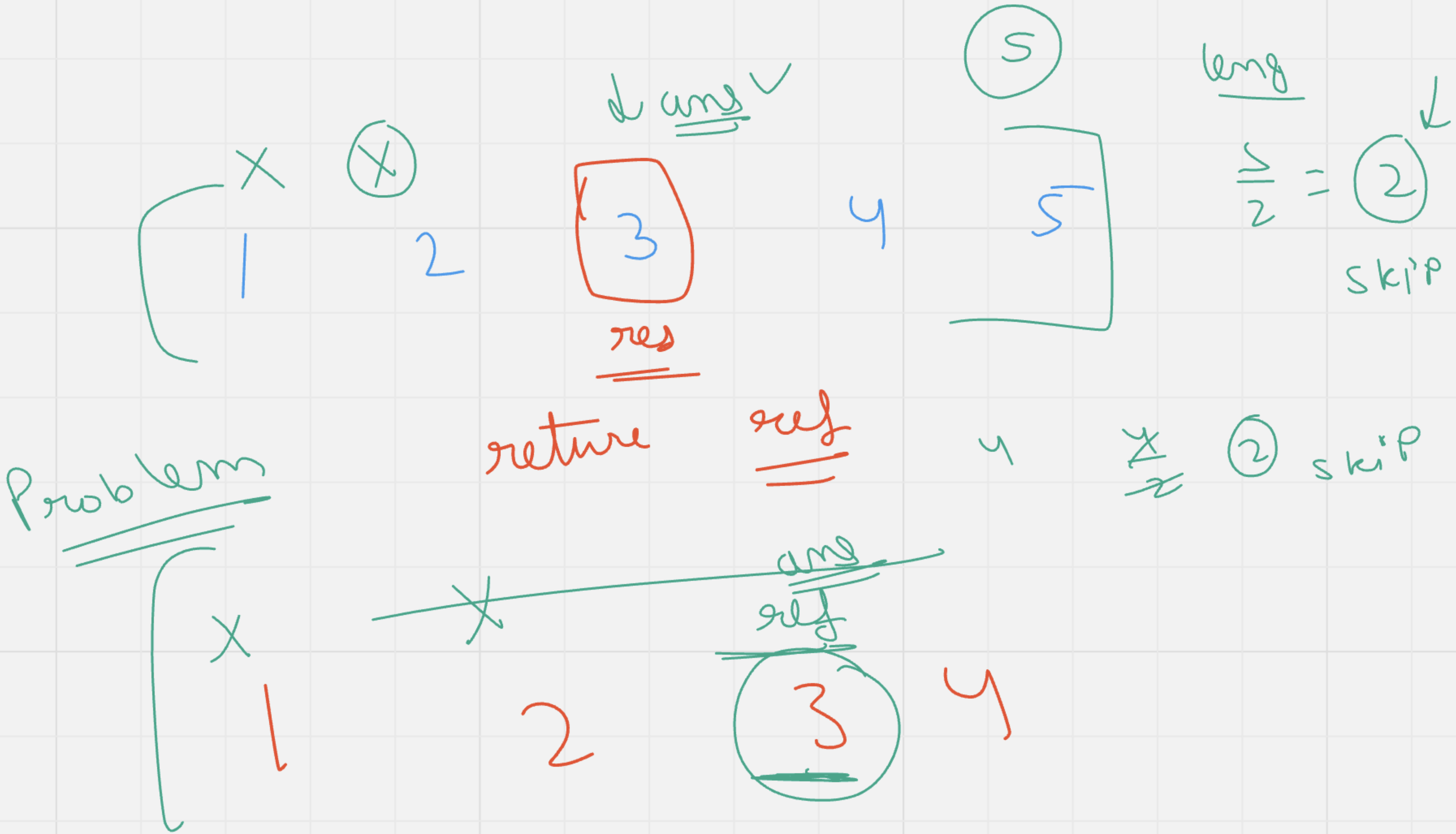
[return head



```
prev = curr
curr = curr.next
```

```
prev.next = nn
nn.next = curr
```


Prev
Curr
head
1 → 2 → 3 → 4 → null
null
nn
SO
nn
return
Pos 1
Base
SO
if pos == 1
nn.next = head
head = nn


2 ans
5
leng
5/2 = 2
skip
X
1
2
3
4
5
res
return ref
4
4/2 2 skip
Problem
ans
ref
X
1
2
3
4

10 20 30 40 50

len → 5

skip = $\frac{len}{2}$ = $\boxed{2}$

~~2~~ ~~1~~ 0

```
while(skip-- > 0) {
    head = head.next;
}
```

Two time

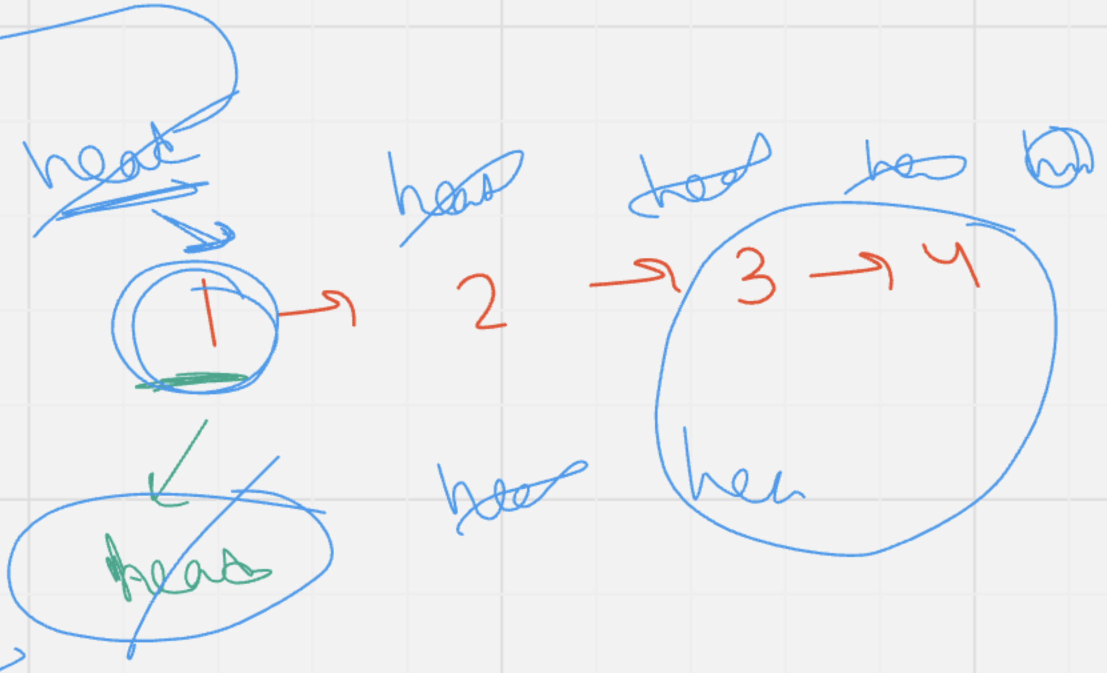
$\boxed{2 > 0}$ True

$\boxed{1 > 0}$ True

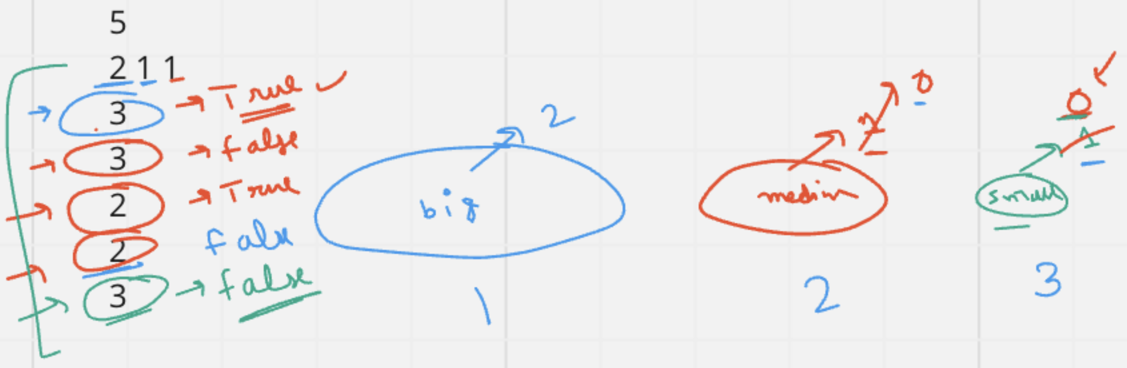
$\boxed{0 > 0}$ False

```
while ( skip>0)
{
    skip--;
}
```

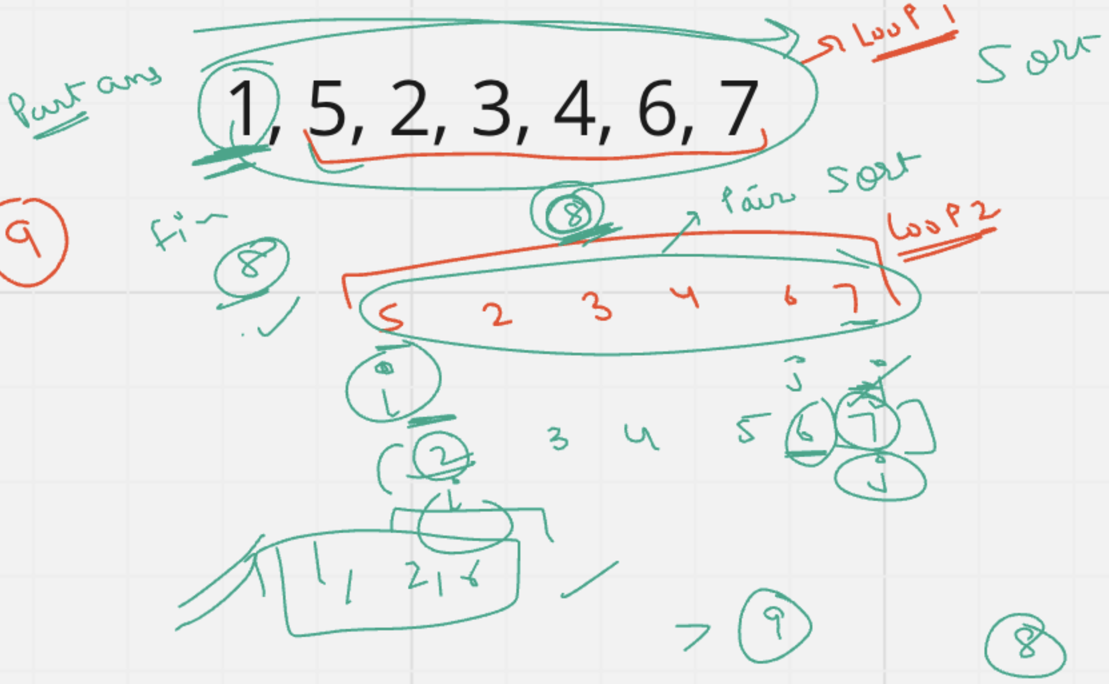
```
static int LenOfLL(Node head) {

    int cnt = 0;
    while(head != null) {
        head = head.next;
        cnt++;
    }
    return cnt;

}

static Node midpointOfLinkedList(Node head)
{
    int len = LenOfLL(head);

    int skip = len / 2;

    while(skip-- > 0) {
        head = head.next;
    }

    return head;

}
```



3 4 ✓


5
2 1 1
3 → True
3 → false
2 → True
2 false
3 → false
big
2
1
medium
2
small
3

Hasmap


Loop 1
Sort
Part ans
1, 5, 2, 3, 4, 6, 7
9
fin
8
8
pair sort
Loop 2
5 2 3 4 6 7
1 2 3 4 5 6 7
j
1, 2, 8
9
8