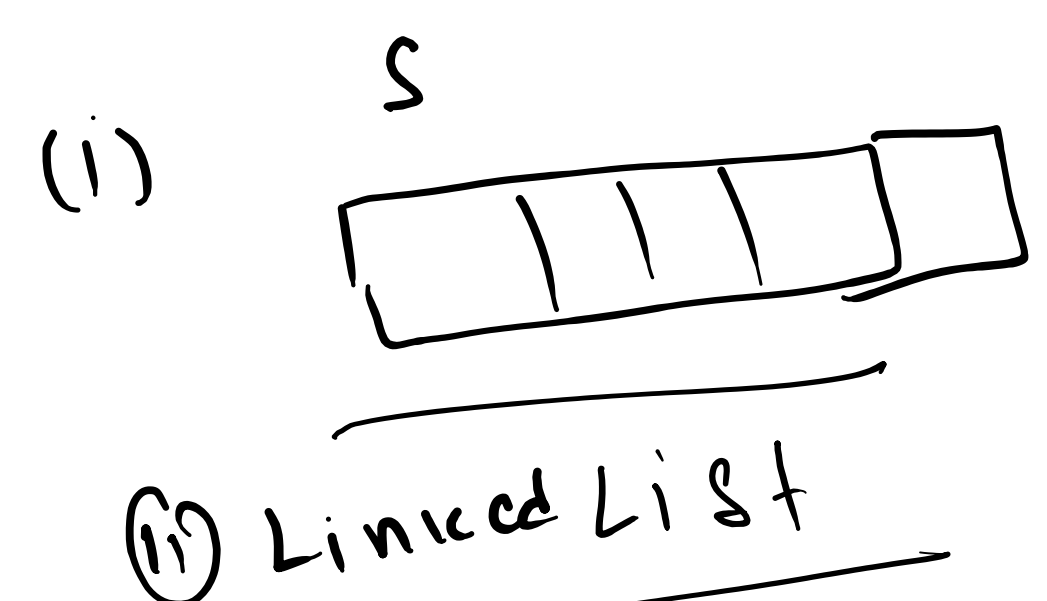
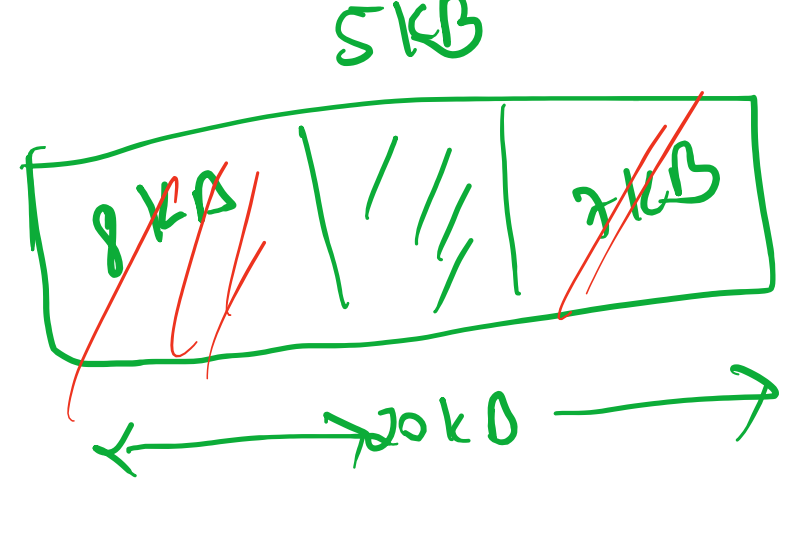
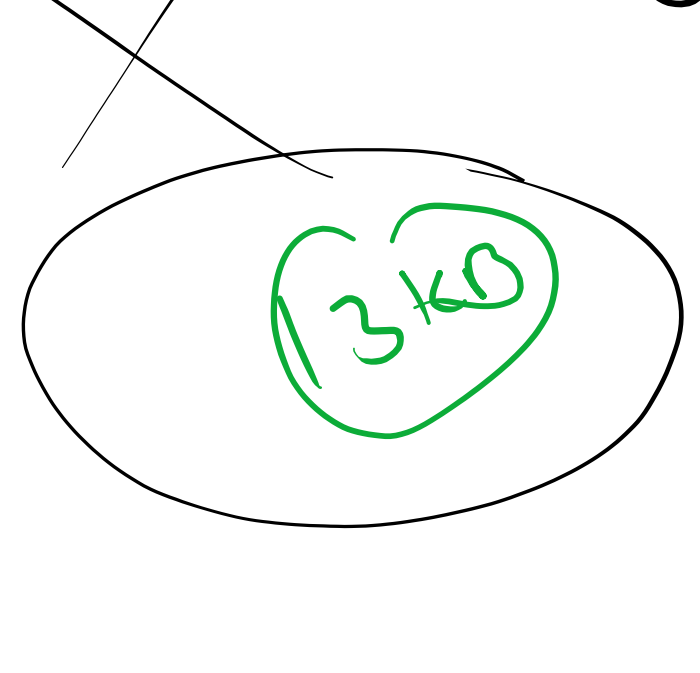


LinkedList
non-contig

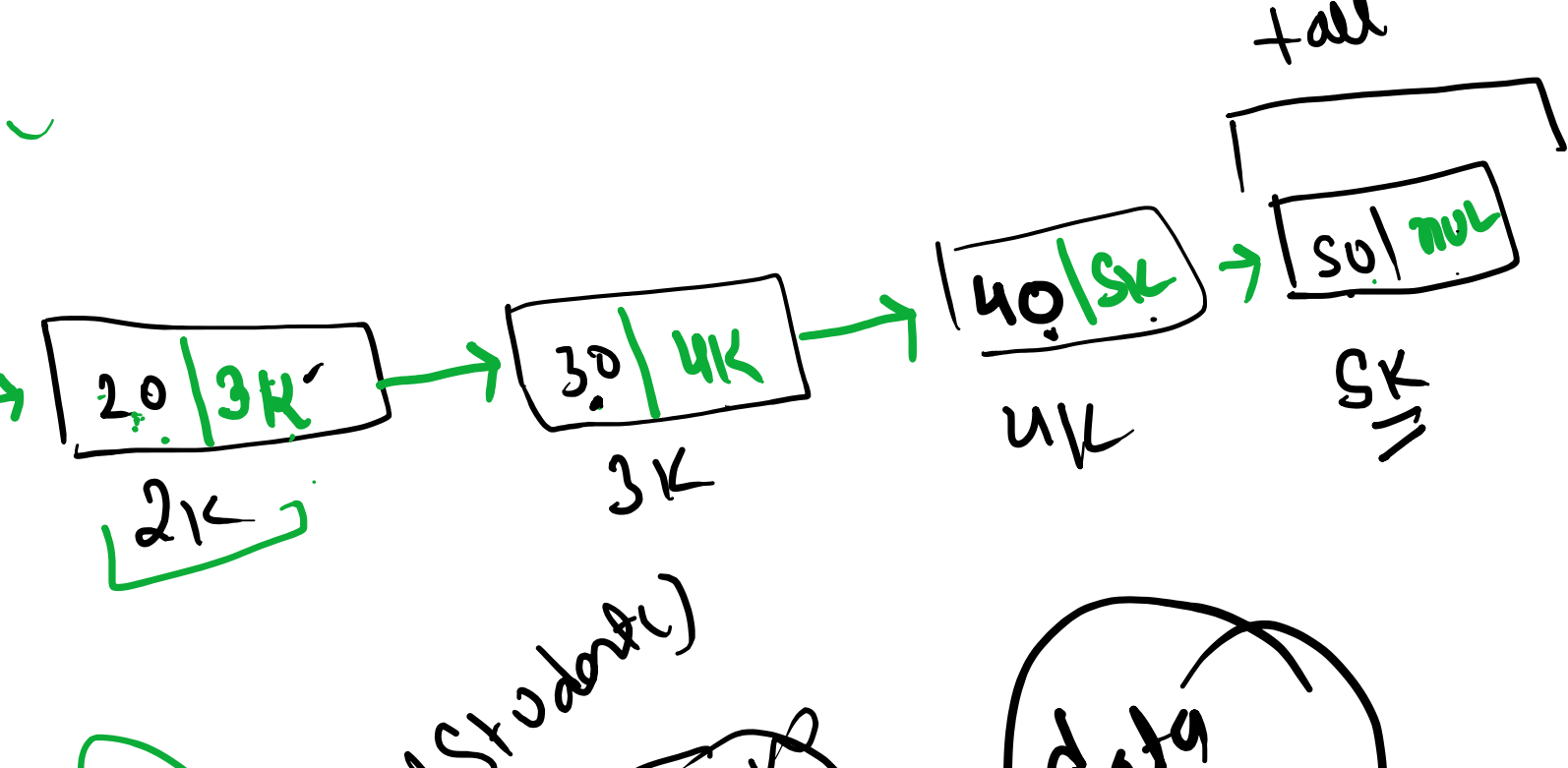


(2) LinkedList
non-contig



S →
Intense

Head



class Student {
String name;
int age;
}

class Node {
int val;
Node next;
}

Student s = new Student();
s.name = "Ravi";
s.age = 20;

data
Address

Add
AddFirst
AddLast
Add many
Remove
display

Add First

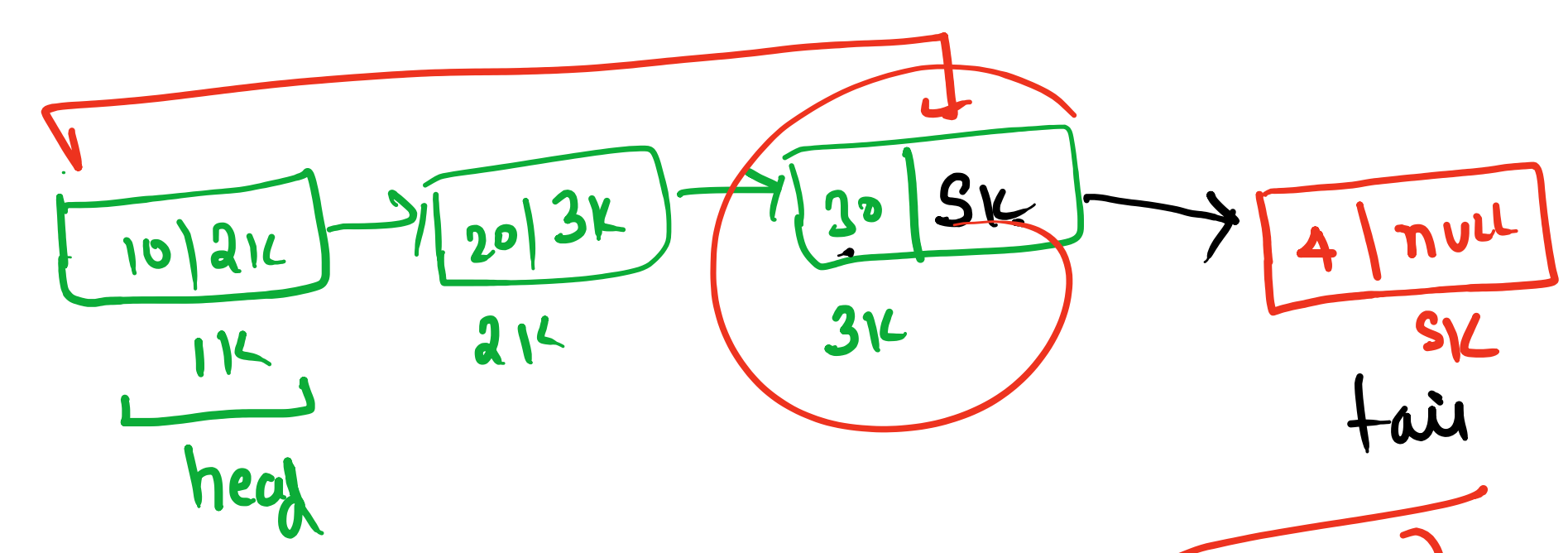
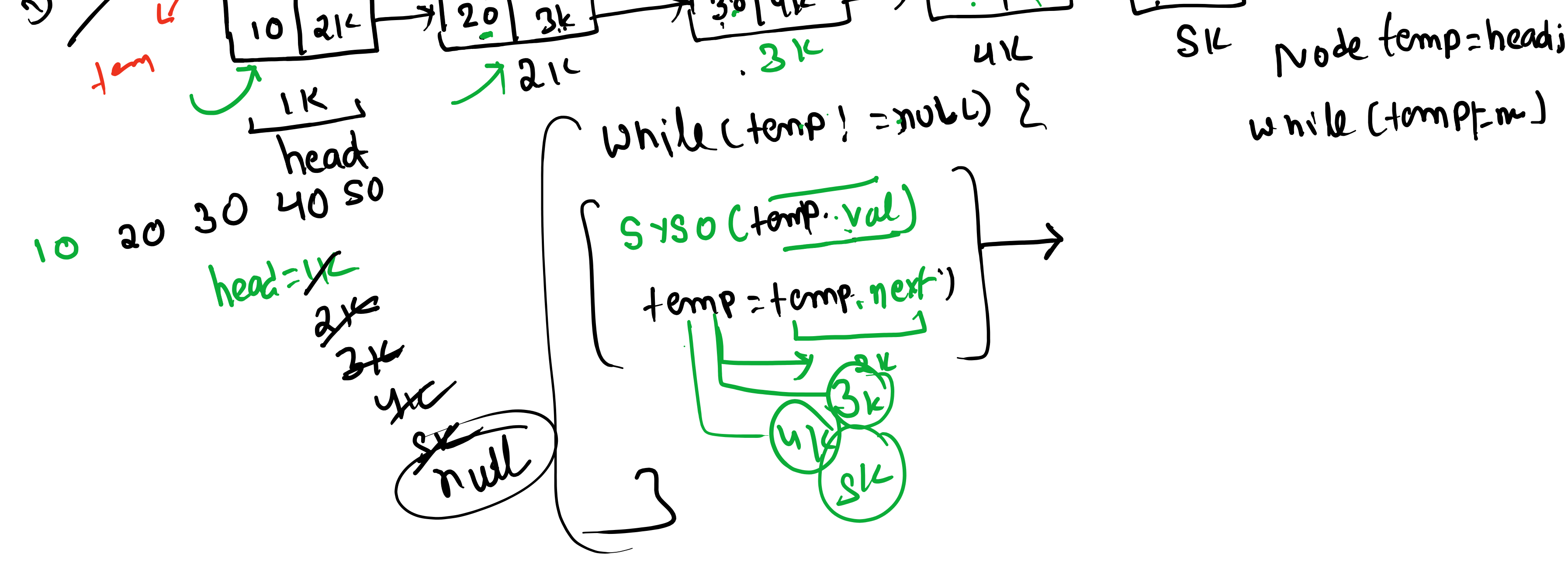
```
public void AddFirst(int item) {  
    Node nn = new Node();  
    nn.val = item;  
    if (size == 0) {  
        head = nn;  
        tail = nn;  
        size++;  
    } else {  
        nn.next = head;  
        head = nn;  
        size++;  
    }  
}
```

Node nn = new Node();
nn.val = item;
if (size == 0) {
 head = nn;
 tail = nn;
 size++;
}

```
public void AddFirst(int item) {  
    Node nn = new Node();  
    nn.val = item;  
    if (size == 0) {  
        head = nn;  
        tail = nn;  
        size++;  
    } else {  
        nn.next = head;  
        head = nn;  
        size++;  
    }  
}
```

10 20 30 40 50

Display

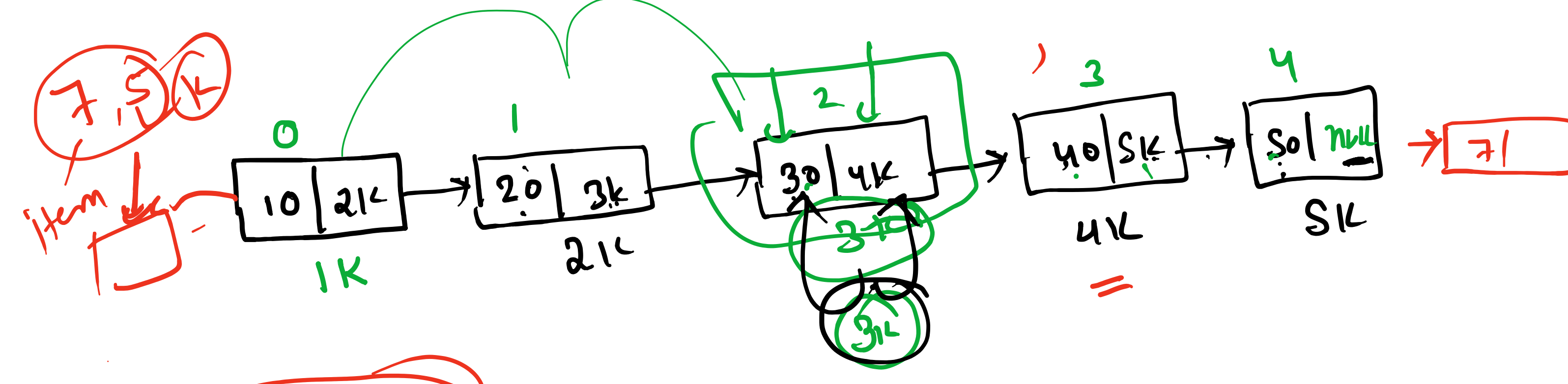


```
public void AddLast(int item) {  
    if (size == 0) {  
        AddFirst(item);  
    } else {  
        Node nn = new Node();  
        nn.val = item;  
        tail.next = nn;  
        tail = nn;  
        size++;  
    }  
}
```

tail.next = nn;
tail = nn;

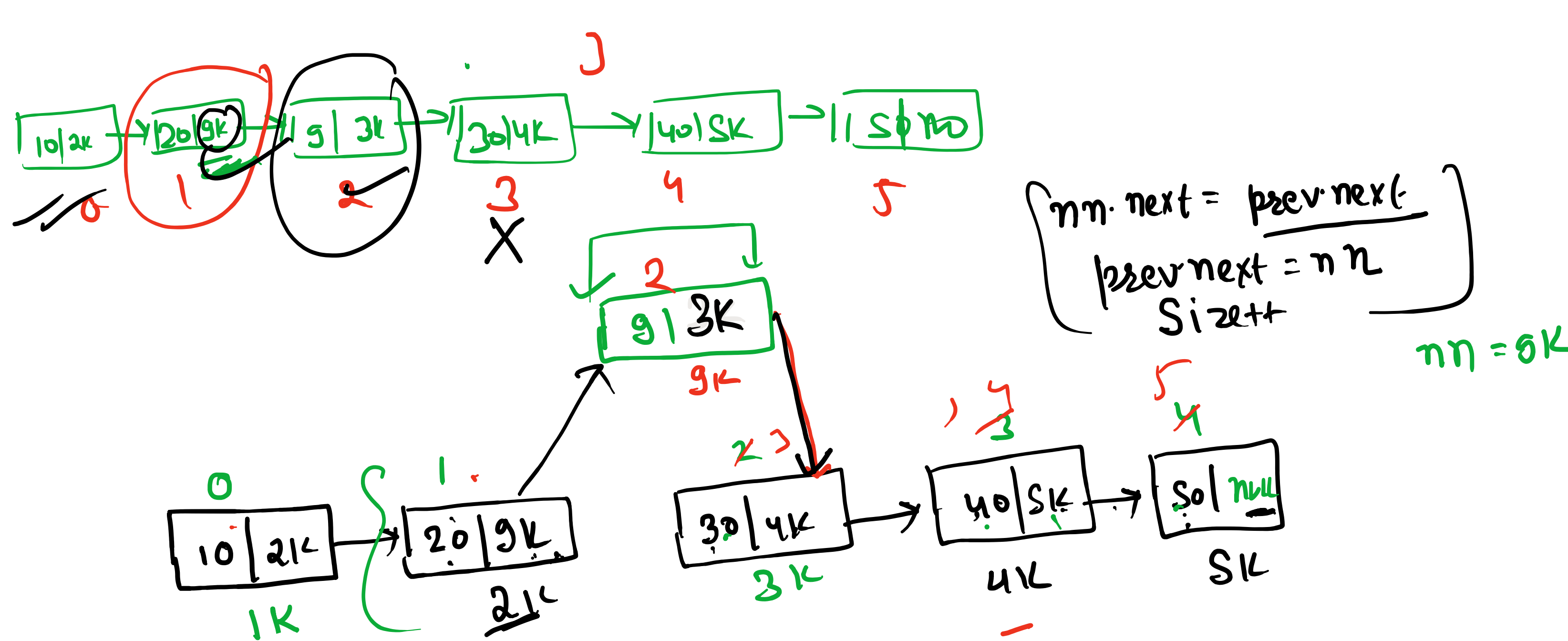
O(n)

O(1)

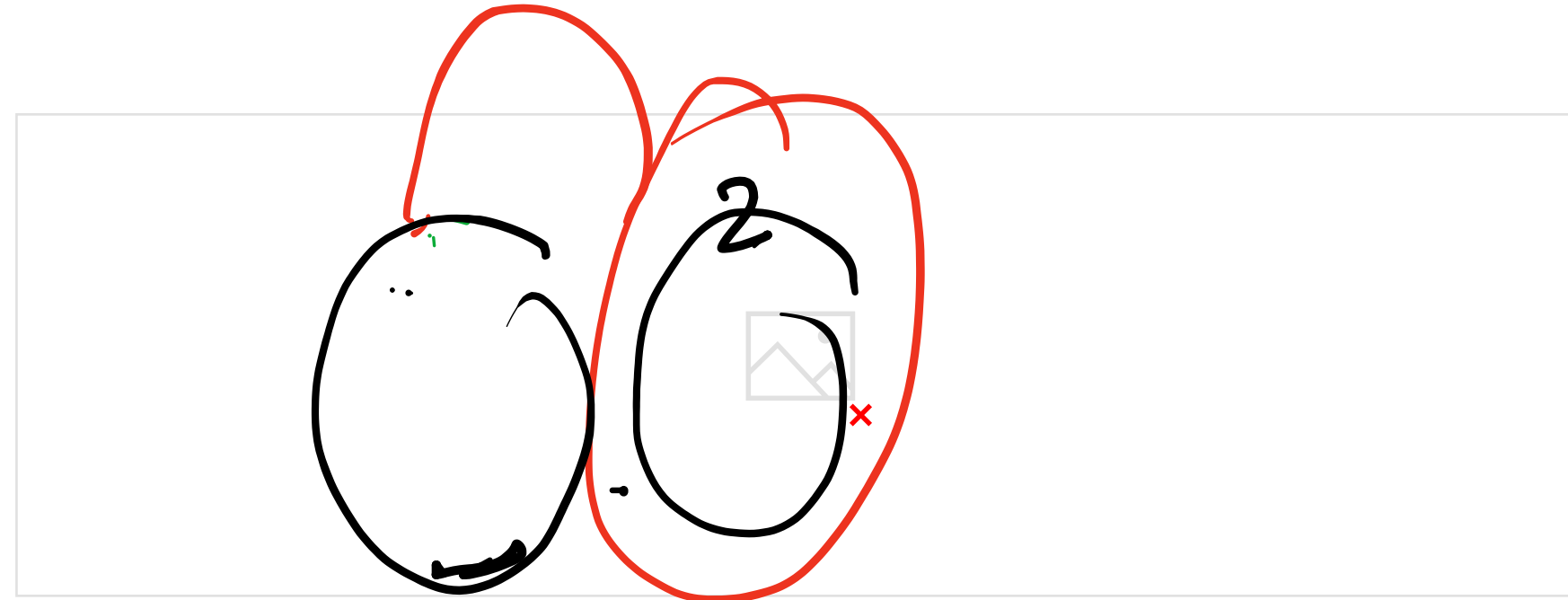


temp = head

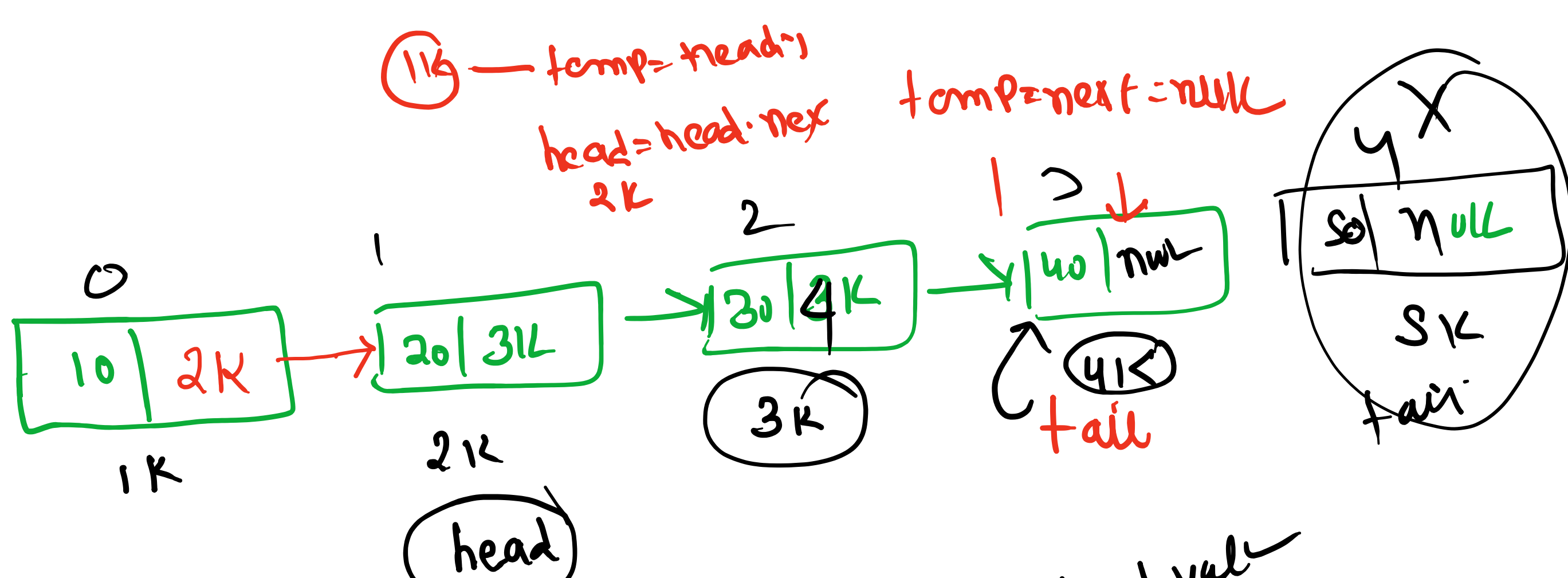
for (i=0; i < k; i++) {
 temp = temp.next;
}



```
public void AddIndex(int k, int item) {  
    if (k == 0) {  
        AddFirst(item);  
    } else if (k == size) {  
        AddLast(item);  
    } else {  
        Node nn = new Node();  
        nn.val = item;  
    }  
}
```

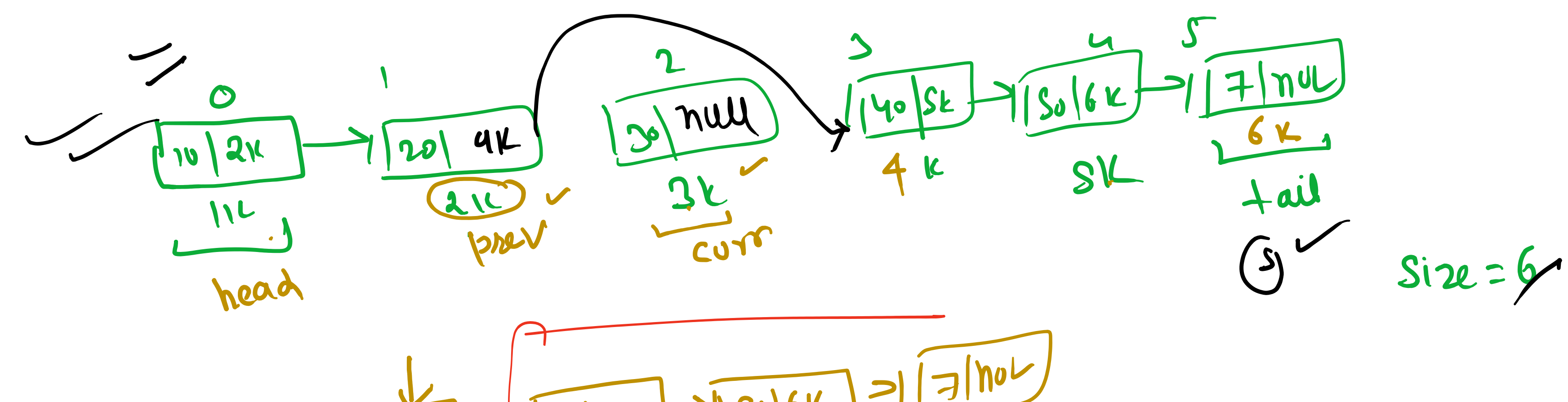


```
public int removeLast() throws Exception {  
    if (size == 1) {  
        return removeFirst();  
    } else {  
        Node nn = getNode(size - 2);  
        int val = tail.val;  
        tail = nn;  
        tail.next = null;  
        size--;  
        return val;  
    }  
}
```



getFirst() → 10
getLast() → 50
getIndex(2) → 30

head.val
tail.val
getNode().val
30.val



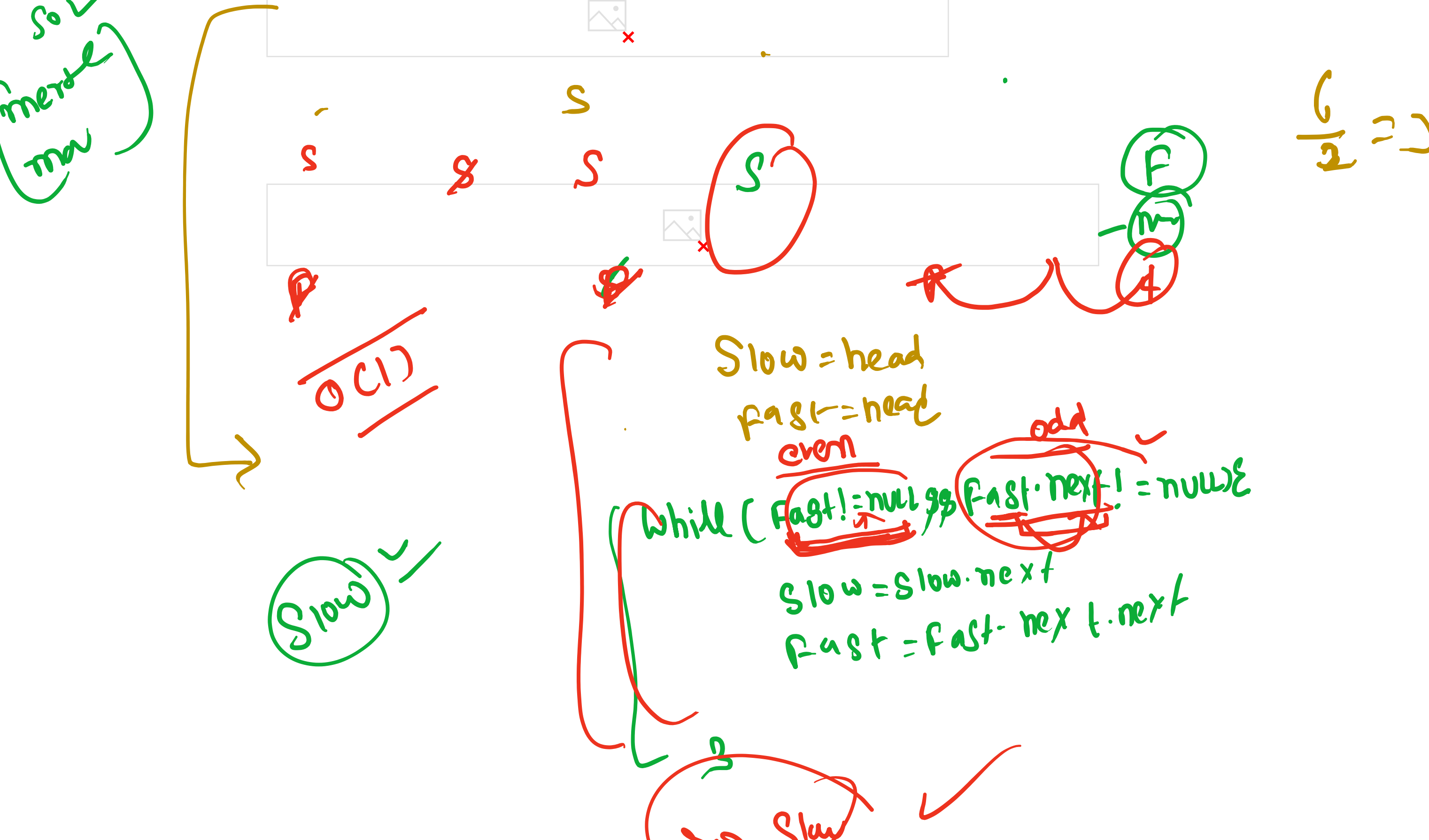
```
Node prev = getNode(k-1);  
Node curr = prev.next;
```

prev.next = curr.next;
curr.next = null

Fast

Fast = Fast.next.next

Slow = Slow.next



2

S/2 = 2

(S/2) = 2

While (fast != null && fast.next != null) {
 slow = slow.next;
 fast = fast.next.next;
}

slow