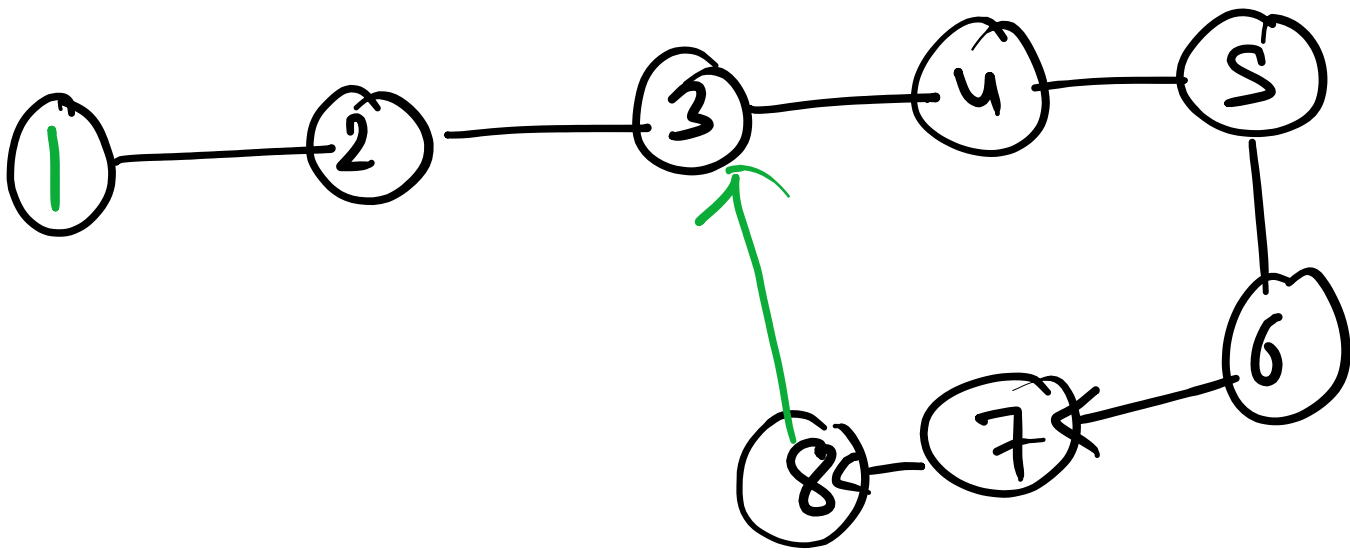


head →

tail → tail.next = null

- ① cycle Linked
- ② Circular Linked

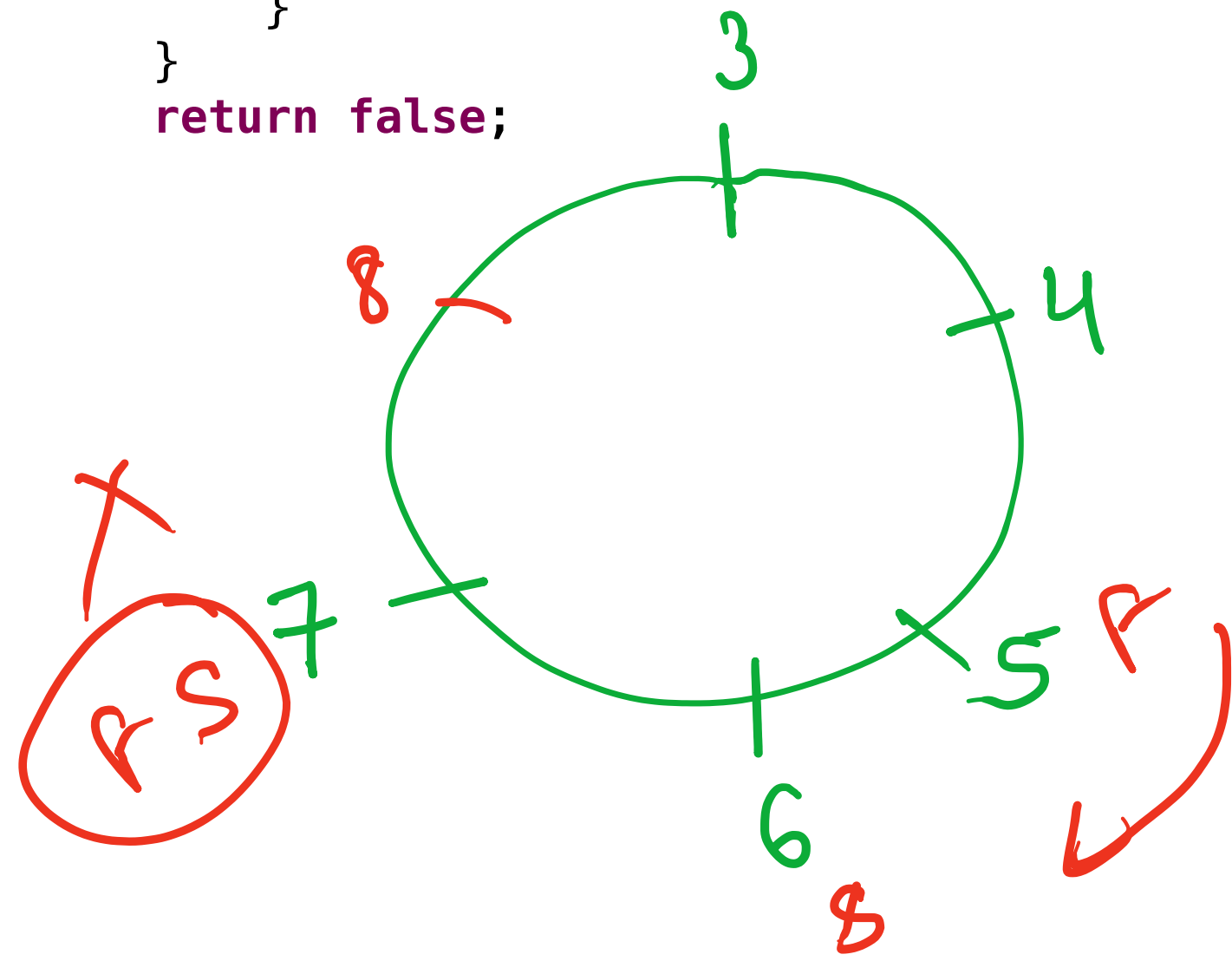
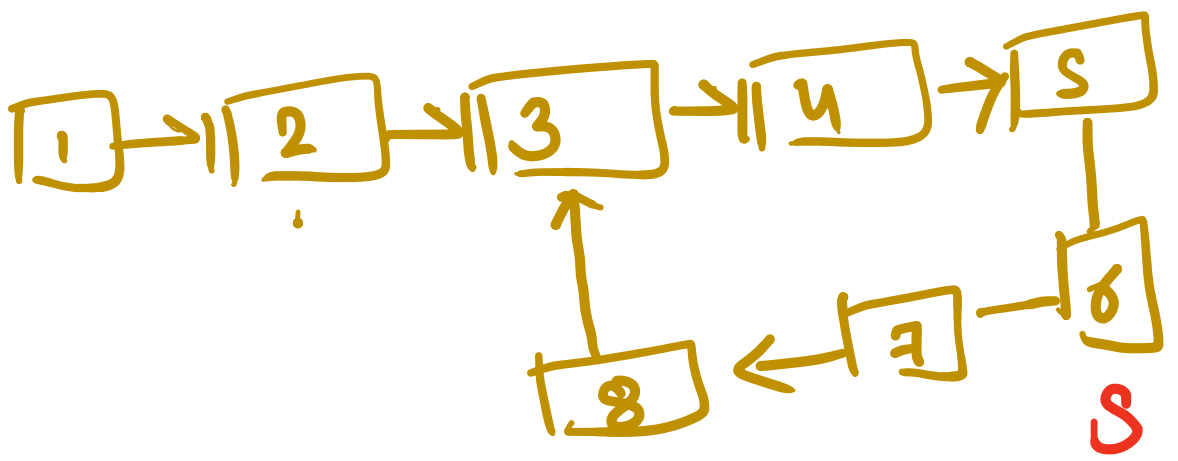
Code math



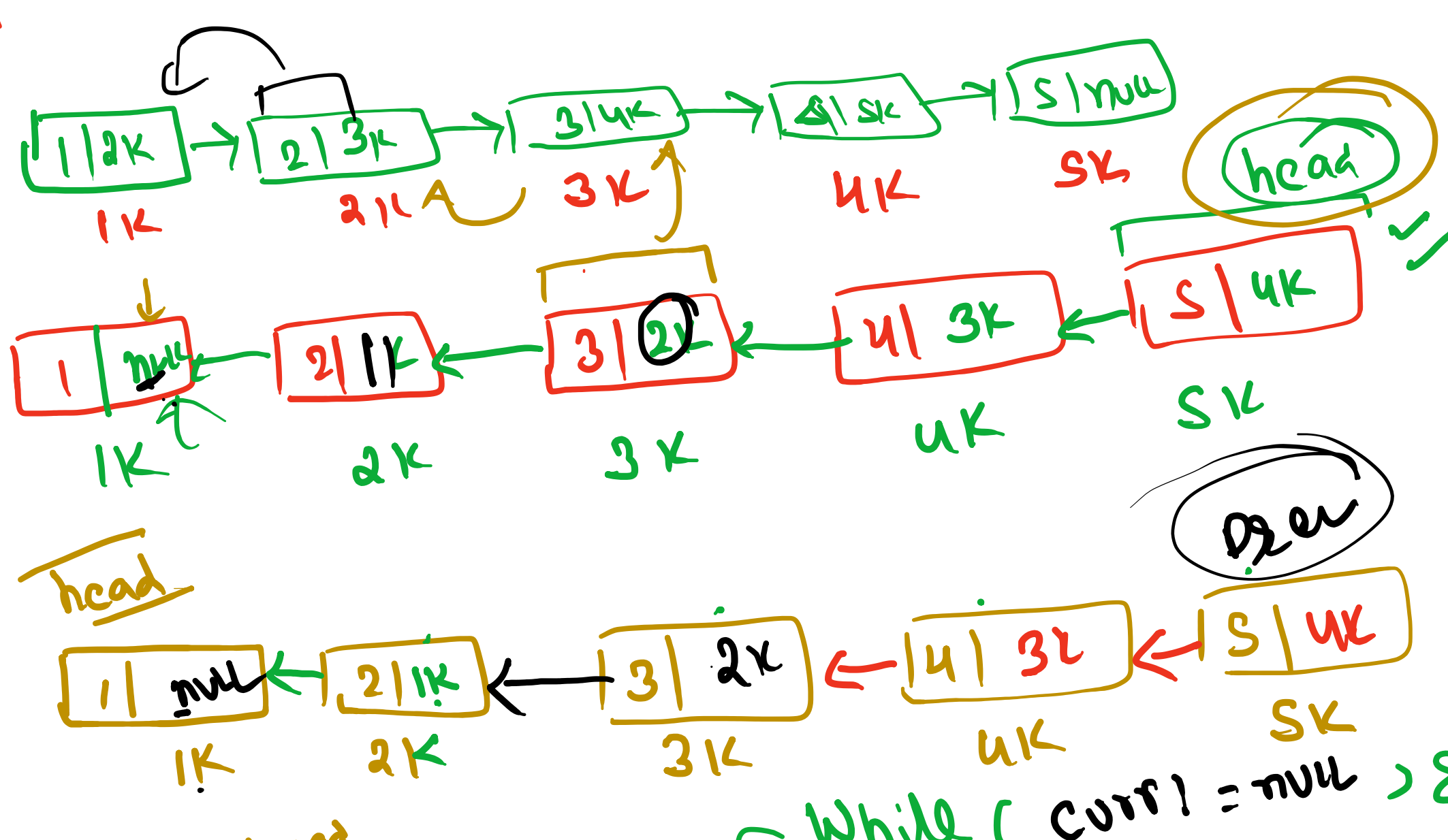
cycle



```
ListNode slow = head;
ListNode fast = head;
while (fast != null && fast.next != null) {
    slow = slow.next;
    fast = fast.next.next;
    if (slow == fast) {
        return true;
    }
}
return false;
```



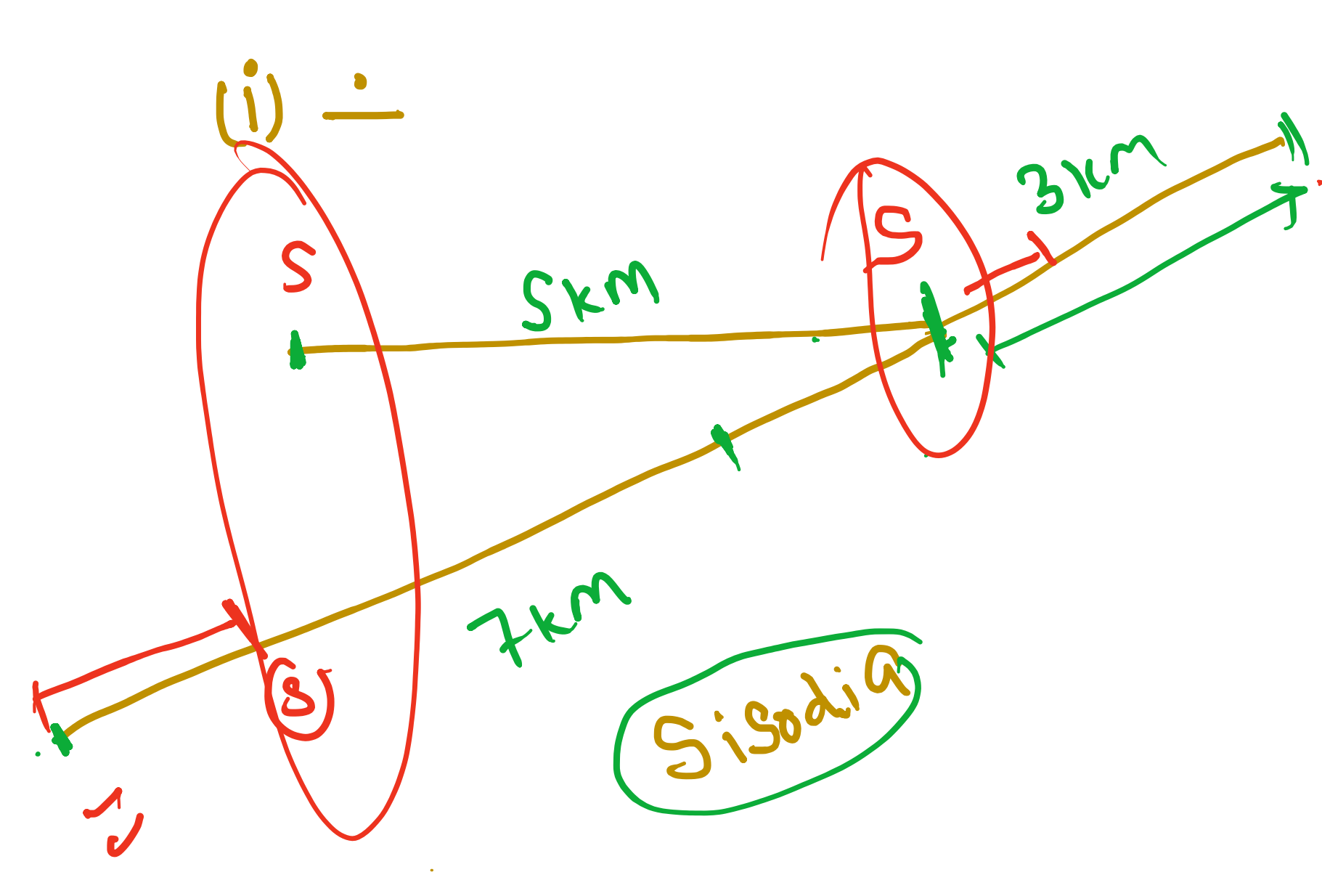
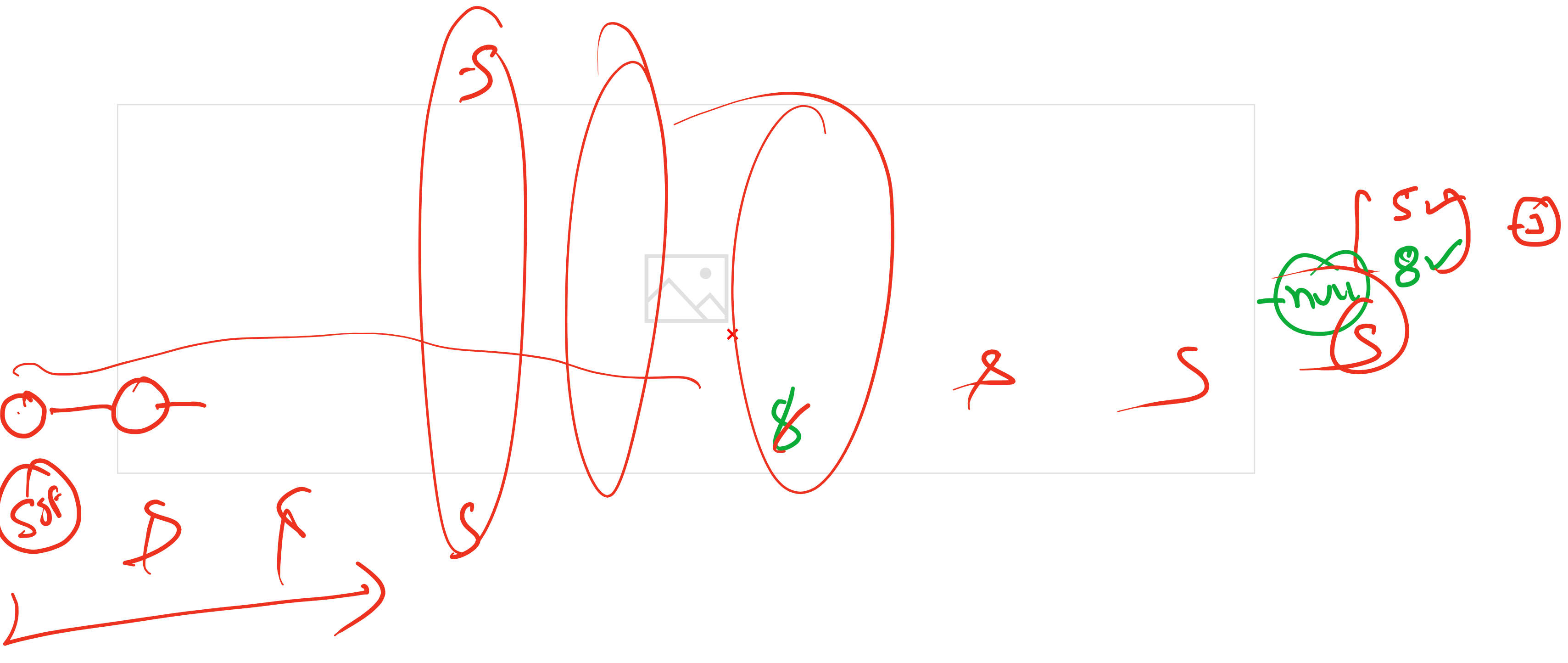
head



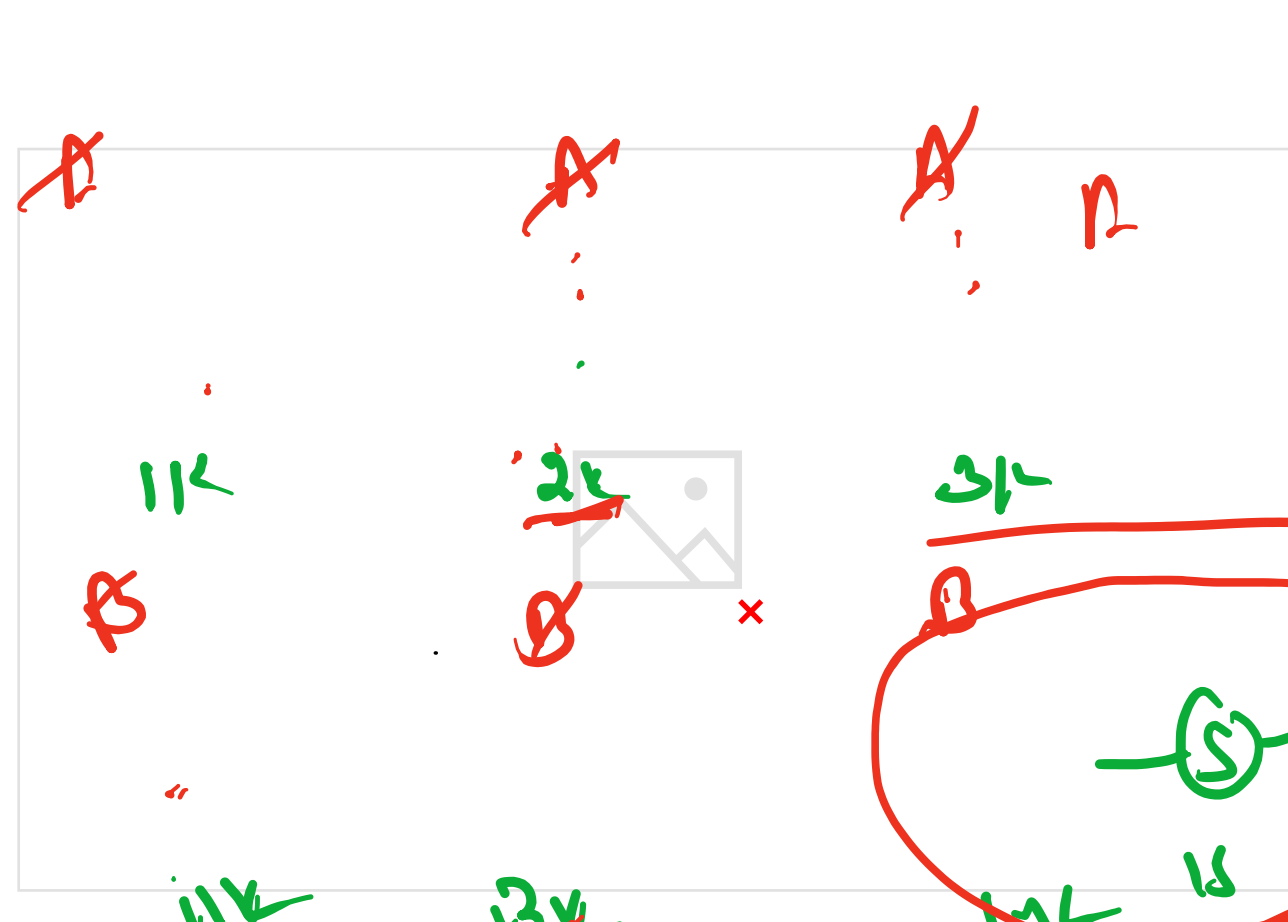
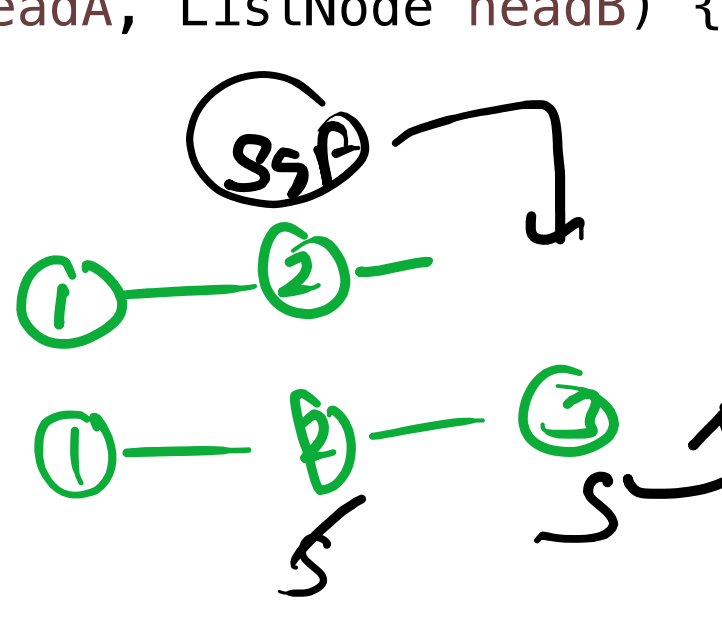
curr = head  
prev = null

```
while (curr != null) {
    Node ahead = curr.next;
    curr.next = prev;
    prev = curr;
    curr = ahead;
}
```

ahead = 2x  
3x 4x 5x

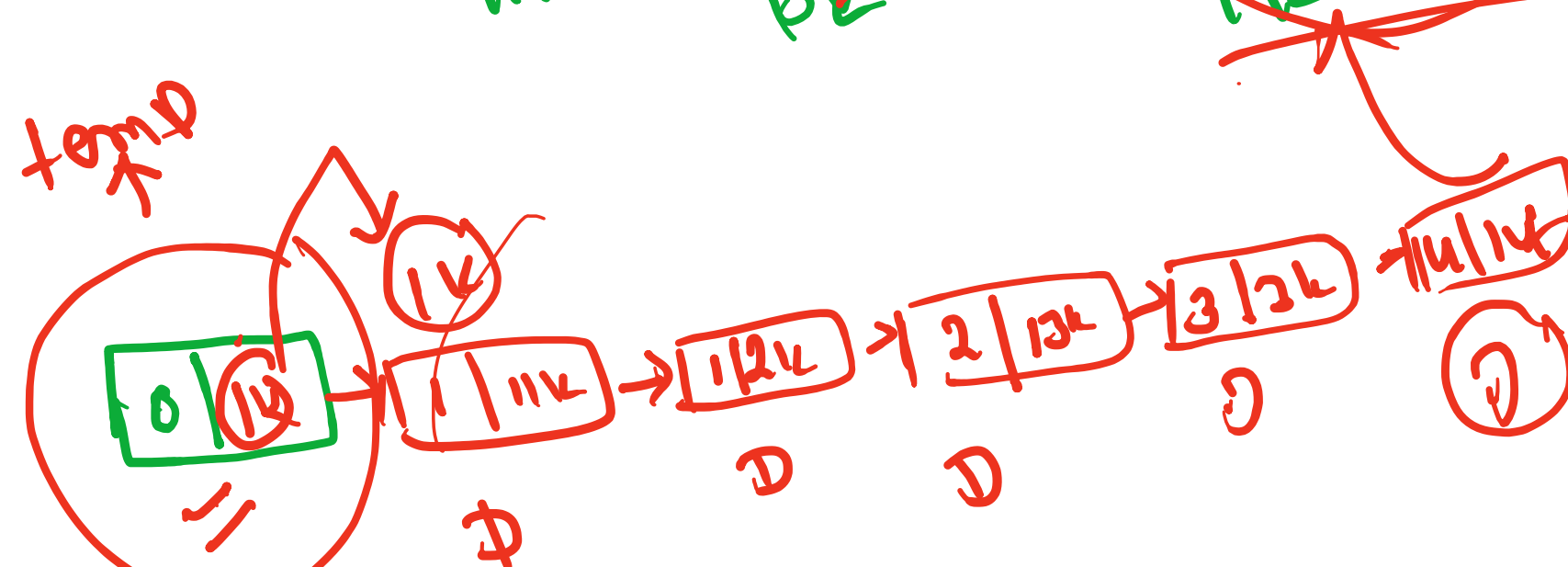


```
public ListNode getIntersectionNode(ListNode headA, ListNode headB) {
    ListNode s = headA;
    ListNode sgf = headB;
    while (s != sgf) {
        if (s == null) {
            s = headA;
        } else {
            s = s.next;
        }
        if (sgf == null) {
            sgf = headB;
        } else {
            sgf = sgf.next;
        }
    }
    return s;
}
```



[1, 2, 4]  
[1, 3, 4]

while (A != null && B != null) {  
 if (B.val < A.val) {  
 dummy.next = B;  
 B = B.next;  
 dummy = dummy.next;  
 }  
 if (A.val < B.val) {  
 dummy.next = A;  
 A = A.next;  
 dummy = dummy.next;  
 }  
}



256

