Analysis

Input: 1 → 2 → 3 → 4 → 5 → null

Output: 5 → 4 → 3 → 2 → 1 → null

Iterative Solution

Step 1:

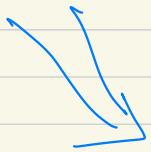
reverse
curr new_head

1 → 2 → 3 → 4 → 5 → null

curr, reverse = head

new_head = curr.next

curr.next = null



reverse
curr
2 → 1

new_head
3 → 4 → 5 → n

Step 2: while (new_head != null)

curr = new_head

new_head = curr.next

curr.next = reverse

reverse = curr

Step 3: return reverse

Recursive Solution

Input 1 → 2 → 3 → 4 → 5 → null

if head == null || head.next == null
    return head

reverse_head = this func (head.next)

head.next.next = head

head.next = null

return reverse_head.

Even in recursion, we don't change reverse_head once found it.

Test with Recursion

head

1 → 2 → 3 → 4 → 5 → null

⇓

...

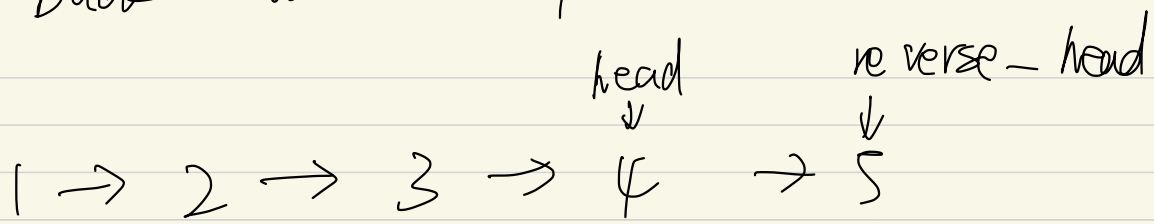
⇓

head

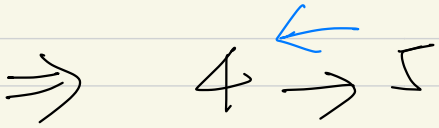
1 → 2 → 3 → 4 → 5 → null ⇒ return 5→null

reverse_head ⇒ 5→null

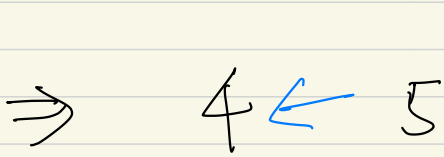
back to the previous recursion



head.next.next = head



head.next = null



reverse_head