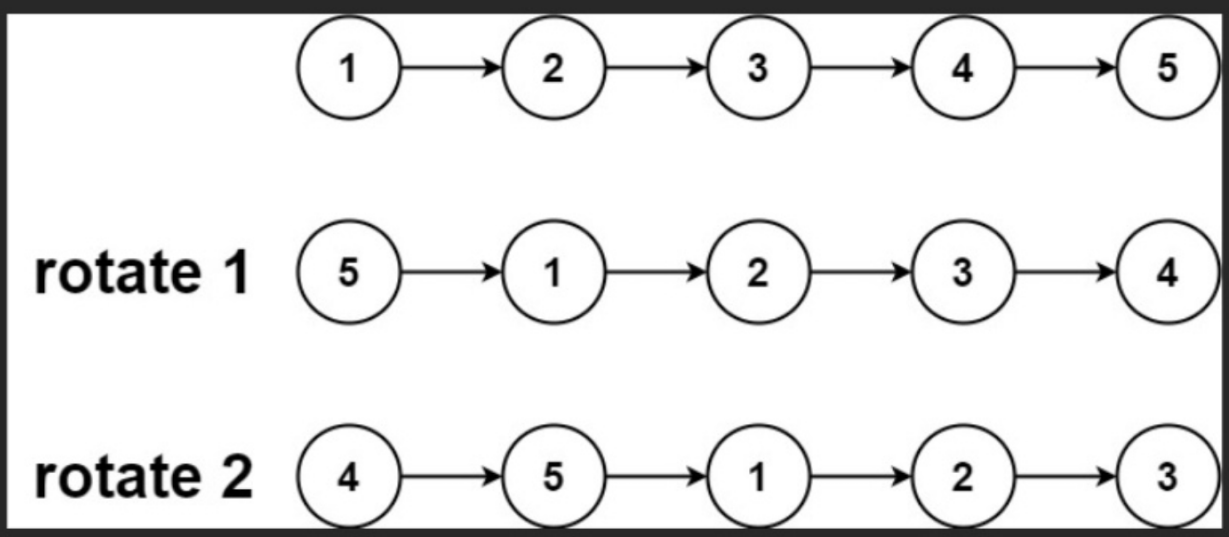
61. Rotate List

Medium 8.7K 1.4K

Companies

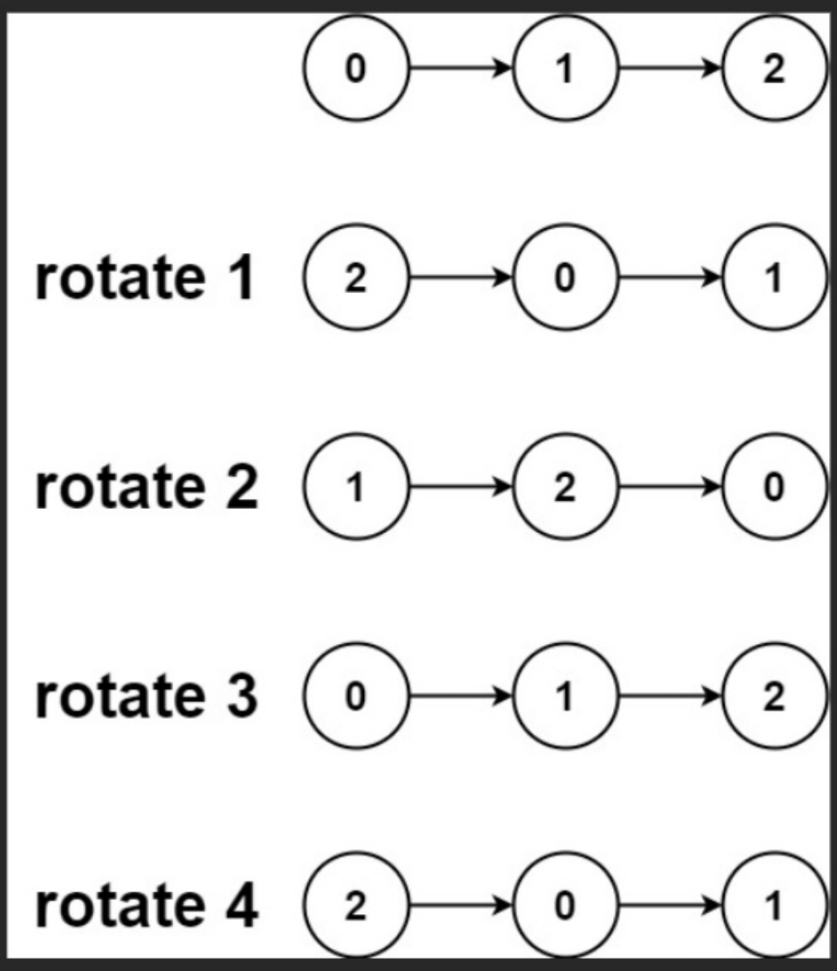
Given the `head` of a linked list, rotate the list to the right by `k` places.

Example 1:




```
Input: head = [1,2,3,4,5], k = 2
Output: [4,5,1,2,3]
```

Example 2:

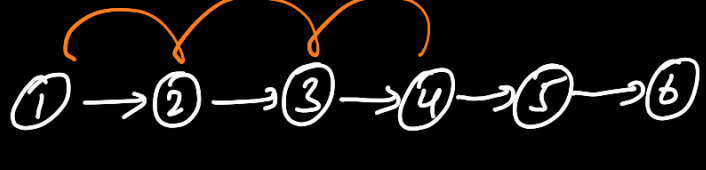



```
Input: head = [0,1,2], k = 4
Output: [2,0,1]
```

Constraints:

- The number of nodes in the list is in the range `[0, 500]`.
- `-100 <= Node.val <= 100`
- `0 <= k <= 2 * 10^9`

Approach 1:

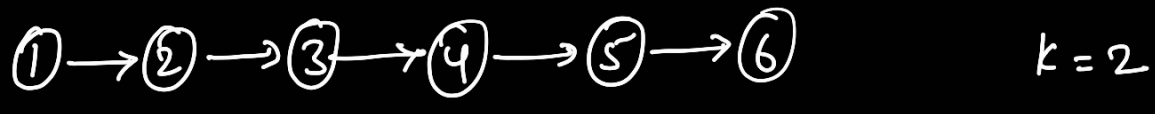


→ find length n of linked list
→ k = k % n
→ if k is 0 return head
→ x = n - k - 1
→ make x steps from head with pointer p.
→ $p_2 = p \rightarrow next$
$p \rightarrow next = NULL$
$p = p_2$
→ now move p_2 to the last node by making required no. of moves.
Then make
$p_2 \rightarrow next = head$

→ return p.

$T(n) : O(n) + O(n-k) + O(k)$
$S(n) : O(1)$

Approach 2:



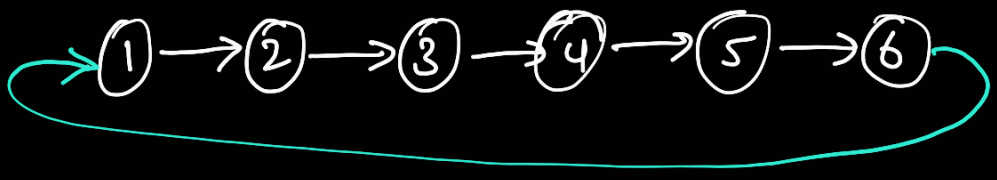
→ move pointer p to the last node and also along with it count length of list.
→ now as p is on last node

k = k % n

if (k == 0) return head

else

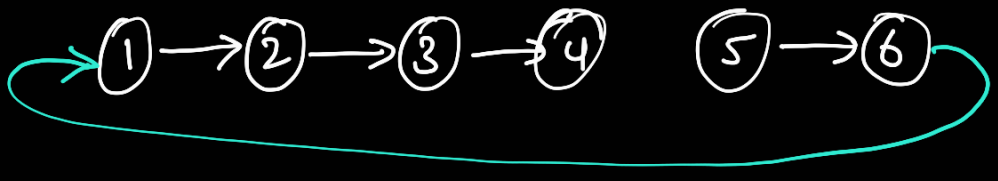
p → next = head // Creating Ring



→ now p is on last node, make n - k steps with p and then make

head = p → next

p → next = NULL // Breaking Ring



T(n) : O(n) + O(n-k)