**难度: 简单**

输入一个链表，输出该链表中倒数第k个节点。为了符合大多数人的习惯，本题从1开始计数，即链表的尾节点是倒数第1个节点。例如，一个链表有6个节点，从头节点开始，它们的值依次是1、2、3、4、5、6。这个链表的倒数第3个节点是值为4的节点。

示例：

给定一个链表: 1->2->3->4->5, 和 k = 2.

返回链表 4->5.

来源：力扣（LeetCode）

链接：https://leetcode-cn.com/problems/lian-biao-zhong-dao-shu-di-kge-jie-dian-lcof

著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。

**思路1(先计算总的节点):**

1). 先计算链表总共有多少个节点

2). 把总的节点减k,就是倒数之后的节点

3). 循环总的节点数减k的次数

4). 返回节点

|  |
| --- |
| public ListNode getKthFromEnd(ListNode head, int k) {  //临时节点  ListNode temp = head;  //次数  int count = 1;  //计算链表总的次数  while(temp.next != null) {  //每次加1  count++;  //给到下一个节点  temp = temp.next;  }  //头节点重新给到临时节点  temp = head;  //计算倒数的节点  count = count - k;  //循环count次  while(count > 0 && temp.next != null) {  temp = temp.next;  count--;  }  //返回节点  return temp;  } |

**思路2(前后指针):**

1). 创建两个节点,分别是前节点和后节点

2). 让后节点先循环k次,这样前后节点就错开了

3). 循环后节点,直到后节点的下一个节点是null,并且前节点也要跟着走下一个节点

4). 结束后,直接返回前节点

|  |
| --- |
| public ListNode getKthFromEn2(ListNode head, int k) {  ListNode afterNode = head; //后节点  ListNode beforeNode = head; //前节点  for (int i = 1; i < k; i++) {  //先让后节点先走,走k次  afterNode = afterNode.next;  }  //前后指针依次走,直到后指针停下来  while(afterNode.next != null) {  afterNode = afterNode.next;  beforeNode = beforeNode.next;  }  //返回前节点  return beforeNode;  } |