「PS：这道题的解法是很多中等和困难难度题的基础，希望大家能熟练掌握。本文还会举一反三，给出常见的变种题目，这些题目都是大厂高频面试题。」

题目介绍[1]

给定一个头结点为 head 的非空单链表，返回链表的中间结点。 如果有两个中间结点，则返回第二个中间结点。

示例 1

> 输入：[1,2,3,4,5]
>
> 输出：此列表中的结点 3。
>
> 解释：返回的结点值为 3。

示例 2

> 输入：[1,2,3,4,5,6]
>
> 输出：此列表中的结点 4。
>
> 解释：由于该列表有两个中间结点，值分别为 3 和 4，我们返回第二个结点。

题目解答

方法一：计数法

思路和算法

最简单的解法就是先遍历整个链表一次，记录节点数为 N，节点数不管是基数还是偶数，我们只要在链表头节点遍历 N/2 次就得到了链表的中间节点。

参考代码

```
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next; }
 * }
 */
class Solution {
    public ListNode middleNode(ListNode head) {
        ListNode node = head;
        int N = 0;
        while (node != null) {
            N++;
            node = node.next;
        }
        int count = N / 2;
        while (count > 0) {
            head = head.next;
            count--;
        }
        return head;
    }
}
```

复杂度分析

n 是链表的长度

- 时间复杂度：O(n)，我们需要遍历两次链表；
- 空间复杂度：O(1)。

方法二：快慢指针

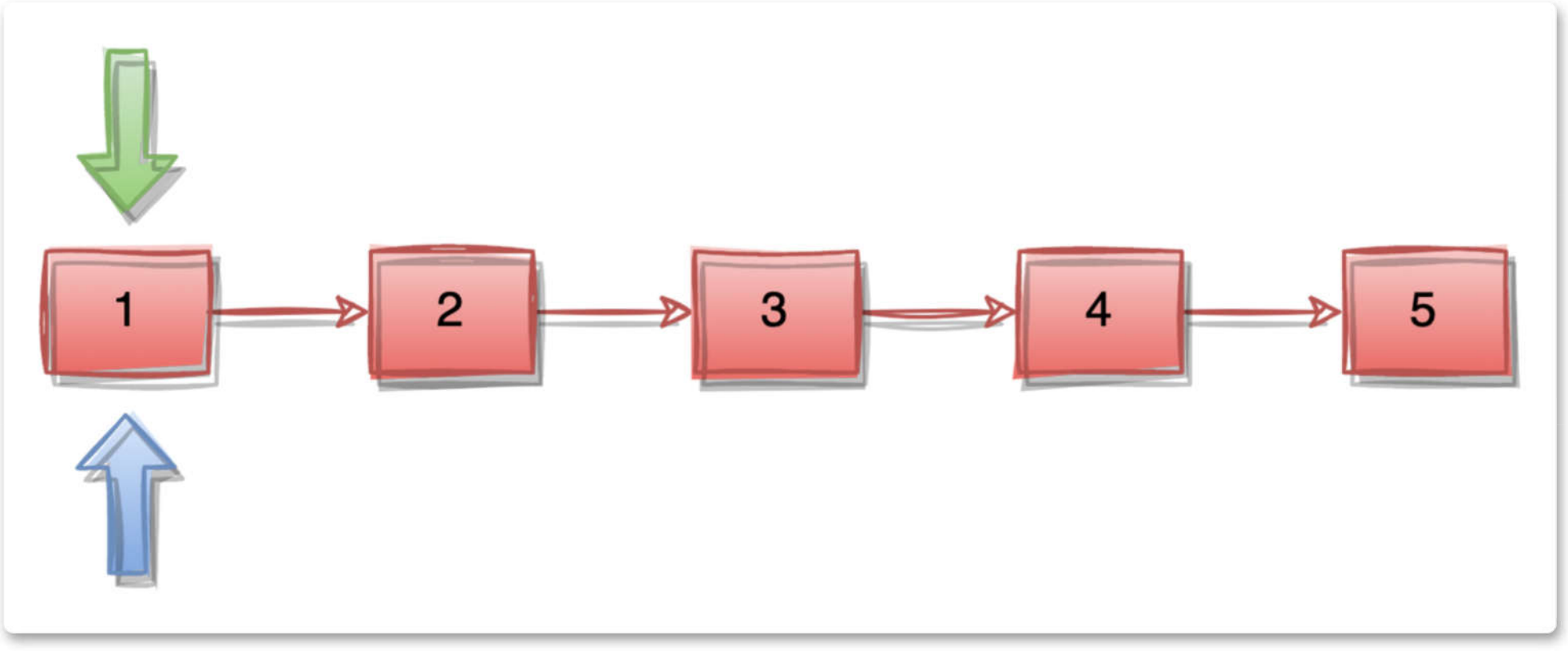
思路和算法

比较经典的做法是，使用两个指针变量，刚开始都位于链表的第 1 个结点，慢指针一次只走 1 步，快指针一次只走 2 步。快指针走过的节点数量永远是慢指针的二倍。这样当快指针走完的时候，慢指针就来到了链表的中间位置。

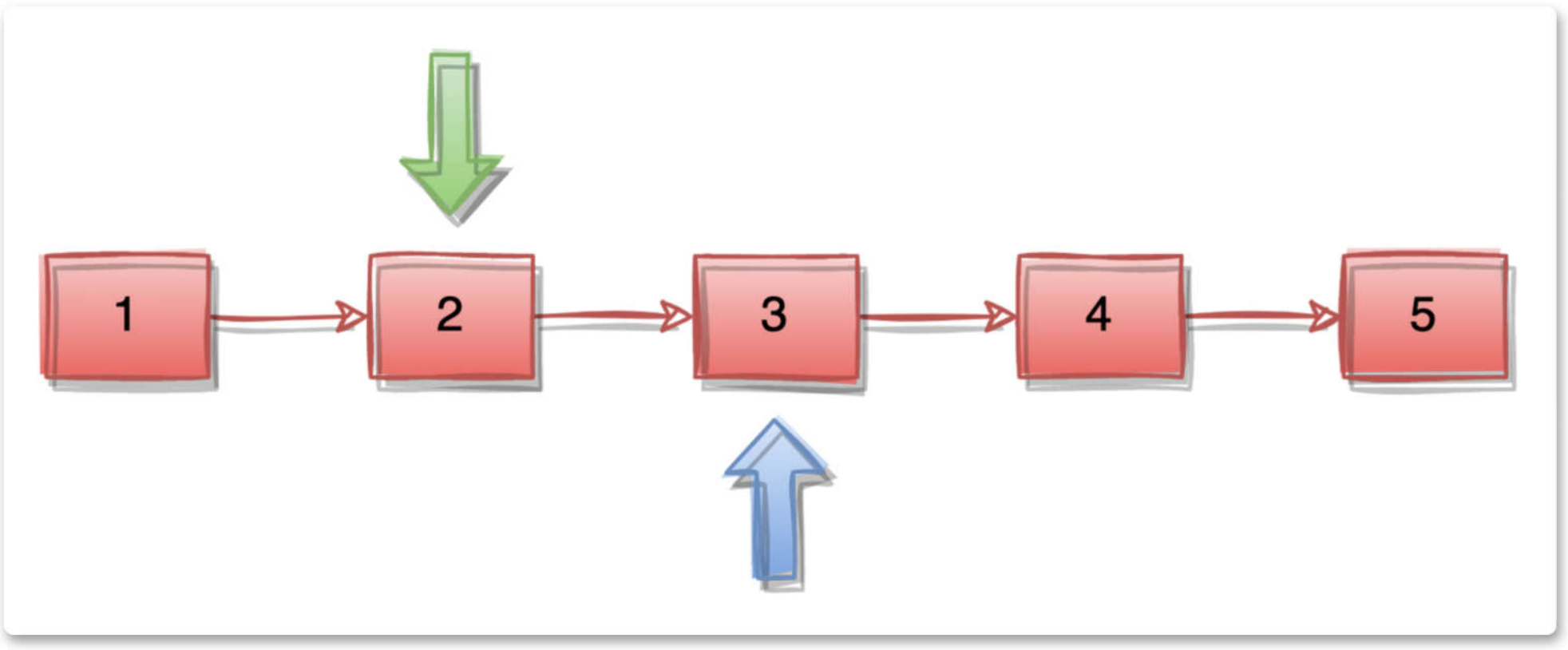
图解

其中绿色指针为慢指针，每次迭代只走一个节点，蓝色指针为快指针，每次迭代走两个节点

「初始状态」

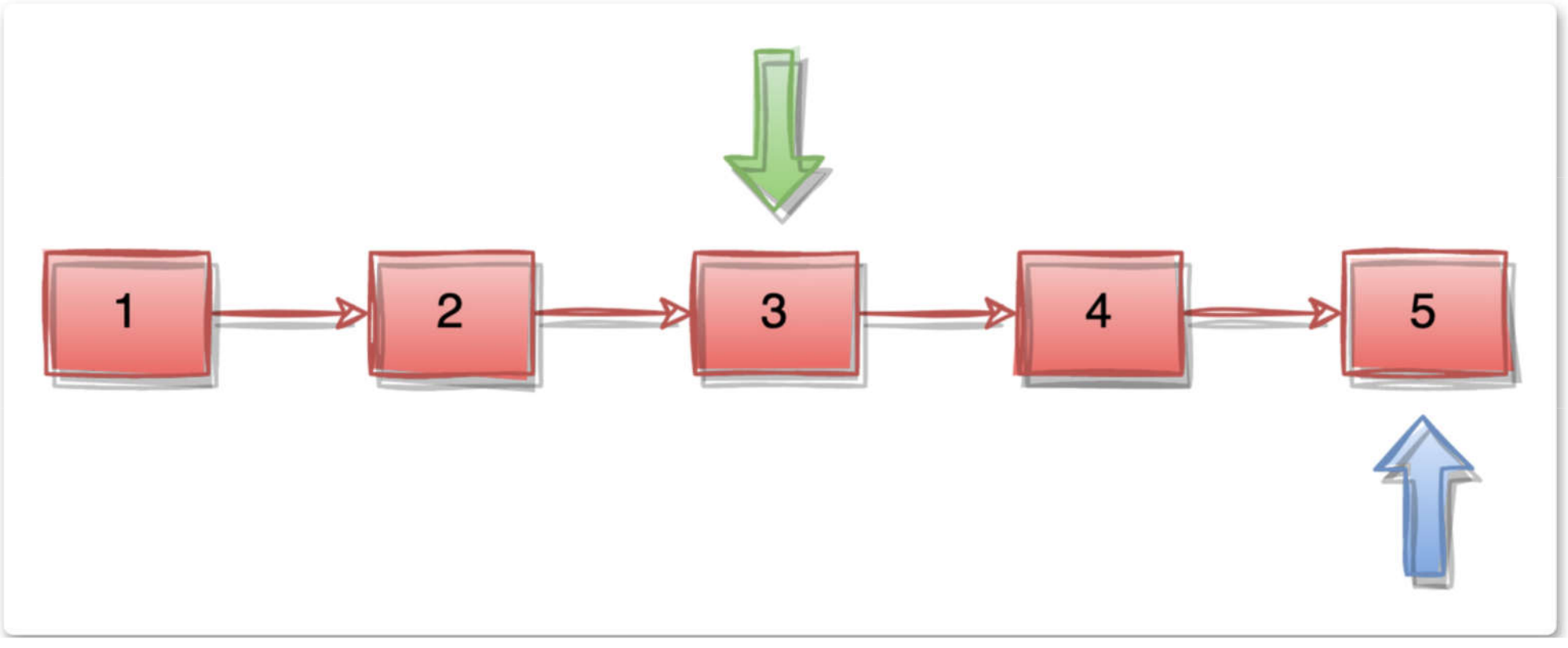


「迭代一次之后」



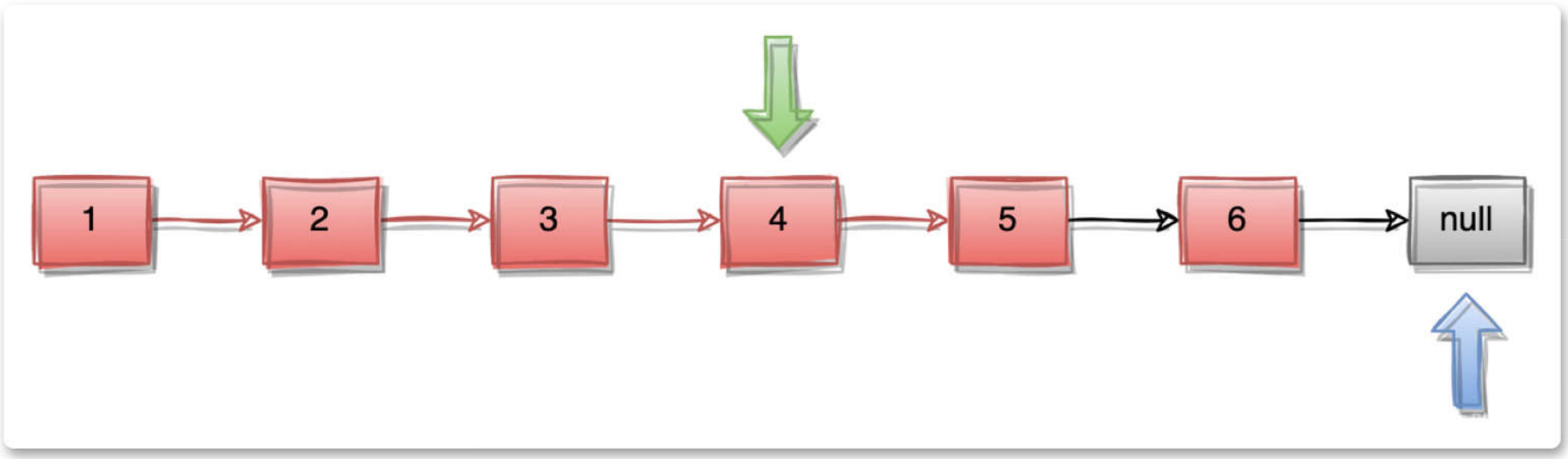
「迭代两次之后」

这时快指针到达了最后一个节点，算法停止迭代了，而慢指针正好到达了中间节点， 因为这个例子的链表有 5 个节点，为基数，所以慢指针最终恰好指向了中间节点。



「迭代三次之后」

我们看一下如果链表节点数为偶数，会是什么样。 这时快指针指向了 null，算法停止迭代了，而慢指针正好到达了靠右的中间节点，



参考代码

```
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next; }
 * }
 */
public class Solution {
    public ListNode middleNode(ListNode head) {
        if (head == null) {
            return null;
        }
        ListNode slow = head;
        ListNode fast = head;
        while (fast != null && fast.next != null) {
            slow = slow.next;
            fast = fast.next.next;
        }
        return slow;
    }
}
```

复杂度分析

n 是链表的长度

- 时间复杂度：O(n)，fast 指针需要遍历一次链表，因此最多走 n 步。
- 空间复杂度：O(1)。

面试 tips

在面试中如果碰见了这道题，只写出计数法是不会加分的，面试官其实就想考察你快慢指针的用法。

参考资料

[1] **原题链接:**
https://leetcode-cn.com/problems/middle-of-the-linked-list/