题目地址(876. 链表的中间结点)

https://leetcode-cn.com/problems/middle-of-the-linked-list/

入选理由

1. 简单难度的双指针。接下来几天的题目类型分别是：二分，快慢指针，滑动窗口

标签

- 双指针
- 链表

难度

- 简单

题目描述

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 (序列化形式：[3,4,5])
返回的结点值为 3 。 (测评系统对该结点序列化表述是 [3,4,5])。
注意，我们返回了一个 ListNode 类型的对象 ans，这样：
ans.val = 3, ans.next.val = 4, ans.next.next.val = 5, 以及 ans.next.next.next = NULL.
示例 2：

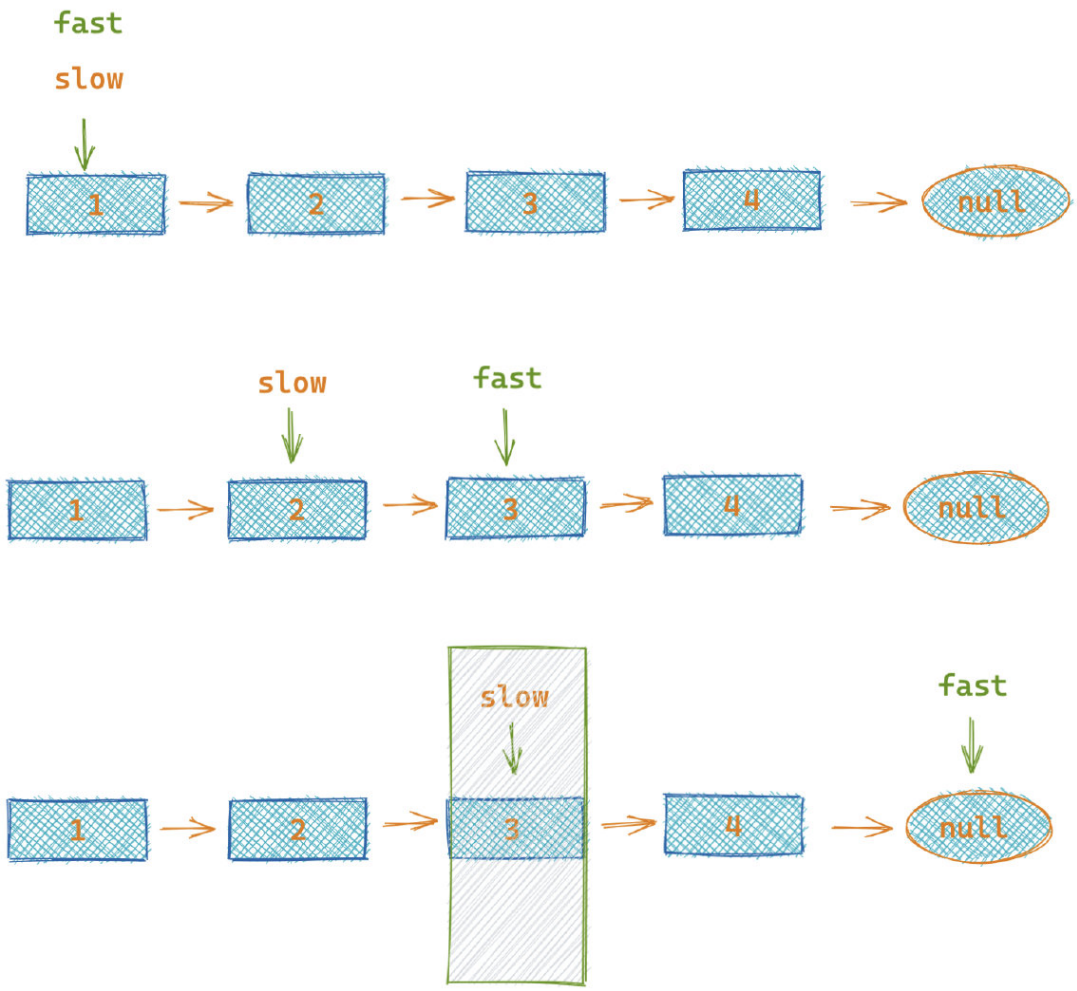
输入：[1,2,3,4,5,6]
输出：此列表中的结点 4 (序列化形式：[4,5,6])
由于该列表有两个中间结点，值分别为 3 和 4，我们返回第二个结点。

提示：

给定链表的结点数介于 1 和 100 之间。

思路

用两个指针记为快指针和慢指针，快指针每次走 2 步，慢指针每次走 1 步，当快指针走到末尾的时候，慢指针刚好到达链表中点。



证明方法见讲义。

代码

代码支持：JS，Python3

JS Code:

```
/**
 * @param {ListNode} head
 * @return {ListNode}
 */
var middleNode = function (head) {
```

```
  let slow = (fast = head);
  while (slow && fast && fast.next) {
    fast = fast.next.next;
    slow = slow.next;
  }
  return slow;
};
```

Python3 Code:

```
class Solution:
    def middleNode(self, head: ListNode) -> ListNode:
        slow = fast = head
        while fast and fast.next:
            fast = fast.next.next
            slow = slow.next
        return slow
```

复杂度分析

令 n 为链表长度

- 时间复杂度: $O(n)$
- 空间复杂度: $O(1)$

上一页 下一页