Problem 2: Given the **head** of a singly linked list, return *the middle node of the linked list*. If there are two middle nodes, return the second middle node.

```
class ListNode:
    def __init_(self, val=0, next=None):
        self.val = val
        self.next = next


def find_middle_node(head):
    if not head:
        return None


    slow = head
    fast = head


    while fast and fast.next:
        slow = slow.next
        fast = fast.next.next


    return slow


def list_to_linked_list(lst):
    dummy_head = ListNode()
    current = dummy_head
    for val in lst:
        current.next = ListNode(val)
        current = current.next
    return dummy_head.next


def linked_list_to_list(head):
    lst = []
    current = head
    while current:
```

```
        lst.append(current.val)
        current = current.next
    return lst


head = list_to_linked_list([1, 2, 3, 4, 5])
middle_node = find_middle_node(head)
result = linked_list_to_list(middle_node)
print(result)
```