在链表问题中，由于链表长度需要一次遍历才能知道，对于一些应用例如求中点等问题来说会多一次遍历。所以有了双指针的使用场景。

双指针，顾名思义就是使用两个指针，又称为快慢指针。通常在链表问题中，快指针一次走两个结点，慢指针一次一个。这样二者走的距离差就是链表长度的一半。

对于求中点问题：当快指针走到末尾即为空时，慢指针刚好走到中点位置。

while fast and fast.next:

fast = fast.next.next

slow = slow.next

双指针还可以求链表中的环以及环的入口。首先追击问题来说，快慢指针一定会在出发后的某个点重合，假设走了x步两个指针都走到重合的这个点，那么快指针和慢指针分别走了2x和x，且二者走的距离差是环的长度（快指针多走一个环），即环长c=x。然后求环的入口，因为环的入口距离环的出口刚好为c，所以快指针先走x步，然后和慢指针一起出发都一步，总会走到重合的结点，这个节点就是换的入口。注意到，第一层找重合点时慢指针已经走了x步，所以不需要从head开始重新让快指针走x步，而是令之前的慢指针为快指针，而慢指针从head开始，重合点就是环的入口。

fast = head

slow = head

step = 0

while fast and fast.next:

step += 1

fast = fast.next.next

slow = slow.next

if fast == slow:

break

# 这里是判断指针出发过没有以及是否存在环

if fast != slow or step == 0:

return None

fast = head

while fast != slow:

fast = fast.next

slow = slow.next

return fast