Quick Work Linked List

```
class Solution(object):
  def quickSort(self, head):
    """
    input: ListNode head
    return: ListNode
    """
    # write your solution here
    if head is None:
      return None
    tail = self.get_tail(head)
    head, tail = self.quicksort(head, tail)
    tail.next = None
    return head

  def get_tail(self, node):
    while node.next:
      node = node.next
    return node

  def quicksort(self, head, tail):
    if head is not tail:
      head_left, tail_left, head_ref, tail_ref, head_right, tail_right =
self.quicksort_partition(head, tail)
      if head_left is None:
        head = head_ref
      else:
        head_left, tail_left = self.quicksort(head_left, tail_left)
        head = head_left
        tail_left.next = head_ref
      if head_right is None:
        tail = tail_ref
      else:
        head_right, tail_right = self.quicksort(head_right, tail_right)
        tail_ref.next = head_right
        tail = tail_right
      return head, tail

  def quicksort_partition(self, head, tail):
    reference = tail
    head_ref, tail_ref = reference, reference
    head_left, tail_left, head_right, tail_right = None, None, None, None
    sentinel = ListNode(None)
    sentinel.next = head
```

```
    node = sentinel
    while node.next is not tail:
      node = node.next
      if node.val > reference.val:
        if head_right is not None:
          tail_right.next = node
          tail_right = node
        else:
          head_right = node
          tail_right = node
      elif node.val < reference.val:
        if head_left is not None:
          tail_left.next = node
          tail_left = node
        else:
          head_left = node
          tail_left = node
      else:
        tail_ref.next = node
        tail_ref = node
    return head_left, tail_left, head_ref, tail_reef, head_right, tail_right
```