Linked List (指针操作)

```python
"""找中点，快慢指针"""
def middle(head):
    fast = slow = head
    while fast and fast.next:  fast, slow = fast.next.next, slow.next
    return slow

"""找圆环，快慢指针(龟兔算法)"""
def detectCycle(head):
    fast = slow = head
    while fast and fast.next:
        fast, slow = fast.next.next, slow.next
        if fast == slow: # cycle detected
            slow2 = head
            while slow2 != slow: slow2, slow = slow2.next, slow.next
            return slow  # cycle point
    return None   # no cycle

"""翻转链表"""
def reverse(head):
    prev, cur = None, head
    while cur:  cur.next, prev, cur = prev, cur, cur.next
    return prev

"""归并链表"""
def mergeTwoLists(self, l1, l2):
    dummy = tail = ListNode(None)
    while l1 and l2:
        if l1.val <= l2.val:
            tail.next, tail, l1 = l1, l1, l1.next
        else:
            tail.next, tail, l2 = l2, l2, l2.next
    tail.next = (l1 or l2) if (l1 or l2) else None
    return dummy.next

"""
如果 head node 可能更改，为了避免复杂的分类讨论，
可创建 dummy node 指向 head node，程序结束后，返回 dummy.next
"""
def operation(head):
    dummy = ListNode(None)
    dummy.next = head
    # do something
    return dummy.next
```

快慢指针 (fast:两步，slow:一步)

Middle of the Linked List
Linked List Cycle I (has cycle or not?) `龟兔算法`
Linked List Cycle II (find where the cycle begins) `龟兔算法`

查增删改

Delete Linked List Elements `基本操作 O(n)`
Delete Node in a Linked List (无 head 指针) `与下一节点交换，删除下一节点`
Remove N-th Node From End of List (只能过一遍) `双指针，一个先走N步`
Insert into a Cyclic Sorted List

reverse list (见上)

Reverse Linked List I (reverse 1::N)
Reverse Linked List II (reverse a::b, one pass)
Palindrome Linked List `找到中点，翻转后半部分，逐一比较`
Swap Nodes in Pairs
Reverse Nodes in K-Group

split & merge list

Split Linked List in Parts
Odd Even Linked List
Reorder List (middle, reverse, merge)
Partition List (split, merge)
Rotate List (find)

plus list (carry 进位)

Linked List Plus One (reverse-reverse)
Add Two Numbers
Add Two Numbers II (reverse) (no-reverse: stack)

break points

Linked List Components **(hashset, count finish)**
Remove Duplicates from Sorted List
Remove Duplicates from Sorted List II

sort list

Merge 2 Sorted Lists
Merge K Sorted Lists `divide & conquer` `heap`

Smart (烧脑题)

Intersection of Two Linked Lists (8字形)
Deep Copy List with Random Pointer `old2new[A] = A'` `or A->A'->B->B'`