链表

移除链表元素

对链表元素进行移除、调换元素顺序等操作通常需要在链表开头设置虚拟节点，然后让节点指向下下一节点即可

题目：**203. 移除链表元素（简单）**

设计链表

完成链表返回某一元素，在链表头、尾、某位置添加元素，删除链表某位置元素

注意Java中链表节点的定义，其余内容直接看代码（同样涉及虚拟节点的设计）

题目：**707. 设计链表（中等）**

反转链表

注意pre，cur，temp指针的设定，循环内链表指向改变要符合顺序

最好是画图！

题目：**206. 反转链表（简单）**

两两交换链表中的节点

一定要画图！交换步骤分三步走，要临时存储2个节点（因为第一步就会改变cur.next的指向）

题目：24. 两两交换链表中的节点（中等）

删除链表的倒数第N个节点

可以第一次扫描得到链表元素数量，第二次扫描（仅一个指针）进行删除操作

更好的方法：双指针

根据前面数组删除元素的做法，链表删除元素同样应想到双指针

删除倒数第N个节点，就先让快指针移动N步，然后快慢指针同步移动直到快指针走到链表尽头，再根据慢指针进行删除操作

题目：19. 删除链表的倒数第 N 个结点（中等）

链表相交

涉及链表的两条路径时：双指针法，需要判断相交条件（对应了两个指针何种路径）

需要考虑链表无相交时的情况，程序能将有相交和无相交的情况“通吃”是关键（还要防死循环）

题目：面试题02.07. 链表相交（简单）

环形链表II

环形链表也常用快慢指针法（涉及快指针追慢指针）

两大关键：判断是否有环，以及环的入口在哪里

判断是否有环：

快指针移动过程中出现null，则一定没环（循环条件）

有环的情况下，快指针（一次走两步）一定会追上慢指针（一次走一步），不存在跳步，这作为循环中的判断条件

快指针追上慢指针时，慢指针一定没走完一圈（走完一圈则说明快指针走了两圈，中途一定追上了慢指针），这作为判断环的入口关系式的依据

判断环的入口：

一定要画图设出未知数，探讨未知数间的关系，设置快指针合适的路径与慢指针一起到达入口

“判断是否有环”部分，也解决了环形链表I的题目

题目：

142. 环形链表 II（中等）

141. 环形链表 （简单）