单向链表：

Null

3

1

2

Head

循环链表：

4

3

1

2

Head

链表是一种常见的基础数据结构，结构体指针在这里得到了充分的利用。链表可以动态的进行存储分配，也就是说，链表是一个功能极为强大的数组，他可以在节点中定义多种数据类型，还可以根据需要随意增添，删除，插入节点。尽管两种结构都可以用来存储一系列的数据，但又各有各的特点。

数组的优势，在于可以方便的遍历查找需要的数据。在查询数组指定位置（如查询数组中的第4个数据）的操作中，只需要进行1次操作即可，时间复杂度为O(1)。但是，这种时间上的便利性，是因为数组在内存中占用了连续的空间，在进行类似的查找或者遍历时，本质是指针在内存中的定向偏移。然而，当需要对数组成员进行添加和删除的操作时，数组内完成这类操作的时间复杂度则变成了O(n)。

链表的特性，使其在某些操作上比数组更加高效。例如当进行插入和删除操作时，链表操作的时间复杂度仅为O(1)。另外，因为链表在内存中不是连续存储的，所以可以充分利用内存中的碎片空间。除此之外，链表还是很多算法的基础，最常见的哈希表就是基于链表来实现的。基于以上原因，我们可以看到，链表在程序设计过程中是非常重要的。

链表就像是一排挂钩，A挂住B，B挂住C，C挂住D……最后形成一个链表。

链表的插入就像是打开A的挂钩，链接上E，这个时候链子断掉了，我们需要再用E挂住A后面的B，形成了一个新的链表。

删除则反之，分别打开A和E的挂钩，把E取下来，这个时候链子断掉了，我们需要再用A挂住E后面的B，形成了一个新的链表。