1. 设a, b, c, d, e, f以所给的次序进栈，若在进栈操作时，允许出栈操作，则下面的不到的序列为（）
2. fedcba B. bcafed C. dcefba D. cabdef
3. 一个栈(无穷大)的进栈序列为1，2，3，…，n，有多少个不同的出栈序列?
4. **对于某个要出栈的元素，在它之前进栈却晚出栈的元素必定是按逆序出栈的**。下述\*表示一个元素。

所以对于这类题，首先就要看例子中第一个出栈的元素。例如C，第一个出栈的元素是d，所以在栈内存储的是abc，按照规则，abc应该会逆序出栈，C中d之后出栈的元素是c\*\*ba，是逆序的，这方面没问题。在往下看比d进栈晚的元素，也就是dce，找到了e，在e之后，比它之前进栈却晚出栈的元素只有ab，之前看的时候已经知道ba逆序了，所以在往下看，也就是f，比它之前进栈却晚出栈的元素只有ab，之前看的时候已经知道ba逆序了，没有在能看的了，所以正确；例如D，第一个出栈的元素是d，所以ab在栈内，ab需要逆序出栈，D中是cab\*\*\*，不是逆序，所以D错误。

如果不能理解，也可以将各序列逐个带入，来确定是否有对应的进出栈序列。

1. 出栈问题是一种卡特兰问题。首先，我们设f（n）=序列个数为n的出栈序列种数。我们假定，最后出栈的元素为k，显然，k取不同值时的情况是相互独立的，也就是求出每种k最后出栈的情况数后可用加法原则，由于k最后出栈，因此，在k入栈之前，比k小的值均出栈，此处情况有f(k-1)种，而之后比k大的值入栈，且都在k之前出栈，因此有f(n-k)种方式，由于比k小和比k大的值入栈出栈情况是相互独立的，此处可用乘法原则，f(n-k)\*f(k-1)种。

求和便是**f（n）=f（0）\* f（n-1）+f（1）\* f（n-2）+……+f（n-1）\* f（0）。**

**递归之后 f（n）=** 

**上周上机题目的解题思路：**

顺序表：

1. 插入：需要检查顺序表是否满了。再检查插入的位置是否合法（p<0&&p>curlen 都是非法），如果都没问题，把顺序表中p位置之后的元素都向后移动一位，在给p位置赋值，curlen++。
2. Delete：需要检查顺序表是否空了。再检查删除的位置是否合法（p<0&&p>curlen 都是非法），在p位置后所有的值都向前移动一位，curlen--。
3. Getpos：从0开始查找，找到元素返回当前位置，找不到返回-1。

单链表倒置：

1. 定义两个结点，一个保存当前链表的头指针，另一个保存当前结点的next，记得把第一个的next赋值为NULL，首先保存当前结点的next，在将当前结点插入到头指针的后面，插入之后把当前结点设置为头指针。当前结点指向保存的next。最后返回开始保存的头指针。

链表合并：

首先，设置四个结点 pa，pb，pc , LC。pa指向a链表，pb指向b链表，LC保存链表的头指针，pc用来连接链表。找到两个链表第一个结点中元素值小的那个结点作为头指针，保存这个头指针，之后再进行循环，比较两个链表中当前结点的值，进行连接。当有一个链表遍历结束后，把没遍历完的链表a或b接到链表的后面

后面两道题书上有代码和思路。