总分：100，第一大题40分，第二大题的两个小题各30分。

第一大题：队列的三种运算各占10分，基本表述占10分

第二大题的第一小题：写出递推公式即可，公式有问题则酌情扣分。

第二大题的第二小题：必要性10分，充分性20分，证明不充分则酌情扣分，必要性和充分性弄反则扣10分

第三章栈与队列习题

3.1 请利用两个栈S1和S2来模拟一个队列。已知栈的三个运算定义如下：PUSH(ST,x):元素x入ST栈；POP(ST,x)：ST栈顶元素出栈，赋给变量x；Sempty(ST)：判ST栈是否为空。那么如何利用栈的运算来实现该队列的三个运算：enqueue:插入一个元素入队列； dequeue:删除一个元素出队列；queue\_empty：判队列为空。（请写明算法的思想及必要的注释）。

用两个栈S1, S2模拟一个队列。

可以把S1与S2背靠背排列，形如S1][S2，把这种结构看做一个队列，那么S1的栈顶代表队尾，

S2的栈顶代表队首。

将一个元素插入队列，只需将该元素压入S1；

删除队首的元素，只需删除S2栈顶的元素，如果S2为空，则需要把S1的元素倒入S2再进行操作，

具体的做法就是将S1的元素一个个退栈压入S2中，全部完成之后再删除S2栈顶的元素。

如果S1与S2均为空，才能判定队列为空，否则队列不空。

\*/

stack S1, S2;

void enqueue(x)

// 将元素x压入队列。

{

PUSH(S1, x);

}

void dequeue(x)

// 删除队首元素，并且存入x中。

{

if (!Sempty(S2))

POP(S2, x);

// 如果S2不空，只需删除S2栈顶的元素。

else

{

while (!Sempty(S1))

{

POP(S1, x);

PUSH(S2, x);

} // 将S1的元素一个个退栈压入S2中，全部完成之后再删除S2栈顶的元素。

POP(S2, x);

}

}

bool queue\_empty()

// 判定队列是否为空。

{

if (Sempty(S1) && Sempty(S2))

return true;

return false;

}

3.2

(1) 编号为1,2,…,n的n辆火车顺序开进栈式结构的站台。禁止将车厢从缓冲铁轨移动至入轨，也禁止从出轨移动车厢至缓冲铁轨。请问开出车站的顺序有多少种可能？请写出你的推导过程。

(2) 证明：从初始输入序列1，2，…，n，可以利用一个栈得到输出序列p1，p2，…，pn（p1，p2，…，pn是1， 2，…，n的一种排列）的充分必要条件是，不存在下标i，j，k，满足i<j<k 同时 Pj<Pk<Pi。

（1）卡特兰数

第一种方法：常规分析

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Catalan递归式。（f(0)=f(1)=1）

第二种方法：非常规分析

对于每一个数来说，必须进栈一次、出栈一次。我们把[进栈](https://baike.baidu.com/item/%E8%BF%9B%E6%A0%88)设为状态‘1’，出栈设为状态‘0’。n个数的所有状态对应n个1和n个0组成的2n位[二进制数](https://baike.baidu.com/item/%E4%BA%8C%E8%BF%9B%E5%88%B6%E6%95%B0)。由于等待入栈的操作数按照1‥n的顺序排列、入栈的操作数b大于等于[出栈](https://baike.baidu.com/item/%E5%87%BA%E6%A0%88)的操作数a(a≤b)，因此输出序列的总数目=由左而右扫描由n个1和n个0组成的2n位二进制数，1的累计数不小于0的累计数的方案种数。

在2n位二进制数中填入n个1的方案数为c(2n,n),不填1的其余n位自动填0。从中减去不符合要求（由左而右扫描，0的累计数大于1的累计数）的方案数即为所求。

不符合要求的数的特征是由左而右扫描时，必然在某一奇数位2m+1位上首先出现m+1个0的累计数和m个1的累计数，此后的2(n-m)-1位上有n-m个 1和n-m-1个0。如若把后面这2(n-m)-1位上的0和1互换，使之成为n-m个0和n-m-1个1，结果得1个由n+1个0和n-1个1组成的2n位数，即一个不合要求的数对应于一个由n+1个0和n-1个1组成的排列。

反过来，任何一个由n+1个0和n-1个1组成的2n位[二进制数](https://baike.baidu.com/item/%E4%BA%8C%E8%BF%9B%E5%88%B6%E6%95%B0)，由于0的个数多2个，2n为[偶数](https://baike.baidu.com/item/%E5%81%B6%E6%95%B0)，故必在某一个奇数位上出现0的累计数超过1的累计数。同样在后面部分0和1互换，使之成为由n个0和n个1组成的2n位数，即n+1个0和n-1个1组成的2n位数必对应一个不符合要求的数。

因而不合要求的2n位数与n+1个0，n－1个1组成的排列一一对应。

显然，不符合要求的方案数为c(2n,n+1)。由此得出输出序列的总数目=c(2n,n)-c(2n,n+1)=c(2n,n)/(n+1)=h(n)。

卡特兰数的应用：栈的序列，二叉搜索树，凸多边形三角划分，方格行走不跨过对角线，矩阵连乘加括号，n对括号正确匹配数目

（2）注意区分必要性和充分性的含义。

必要性：

若存在i, j, k，i < j < k, 但 pj < pk < pi.考虑实际出栈情况：

在pj和pk在入栈队列头的时候，它们必须被压入栈。因为它们后面有编号更大的pi，pi要在pj,pk之前被挪到出栈，而这个动作只有在pj,pk都在栈中才能完成。

而pj和pk在进栈的顺序是一定的，即pj在pk之前。由栈LIFO的性质，推得pk必在pj之前出栈，即pk在出栈序列中的位置在pj前，矛盾；必要性得证。

充分性：(通过逆否命题)

假设现在有一个出栈序列p1…pn，我们不知道它是否合法。但是我们还是可以通过模拟实际的出入栈来判断它是不是合法（这总是做得到的，直到某一步发生非法操作）。具体描述一下模拟的过程，如果当前出栈序列要求pj出栈，pj可能在两个位置，一个是在入栈队列里，一个是在栈内。

若pj在入栈队列里，我们就让入栈队列里pj和pj之前的元素全部入栈，然后让pj出栈就能够使pj到出栈了，所以这种情况不存在矛盾。

若pj在栈内，则若pj正处在栈顶，直接让pj出栈就可以了，也不存在矛盾；若pj不在栈顶，矛盾出现了，这时我们不能够得到要求的出栈序列。

现在假设p1…pn是不合法的序列，我们一定模拟到某一步的时候出现了上述矛盾。不妨假设现在该pj出栈，但pj不在栈顶；

栈中元素的编号一定是单调上升的（这是由于编号大的元素后入栈和栈的LIFO性质造成的），而pj不在栈顶，必有pk在栈中，且在pj之上，此时一定满足pk > pj。因为当前的假设是轮到pj出栈，而pk目前还在栈中，所以pk只可能在pj之后出栈（即j<k）；

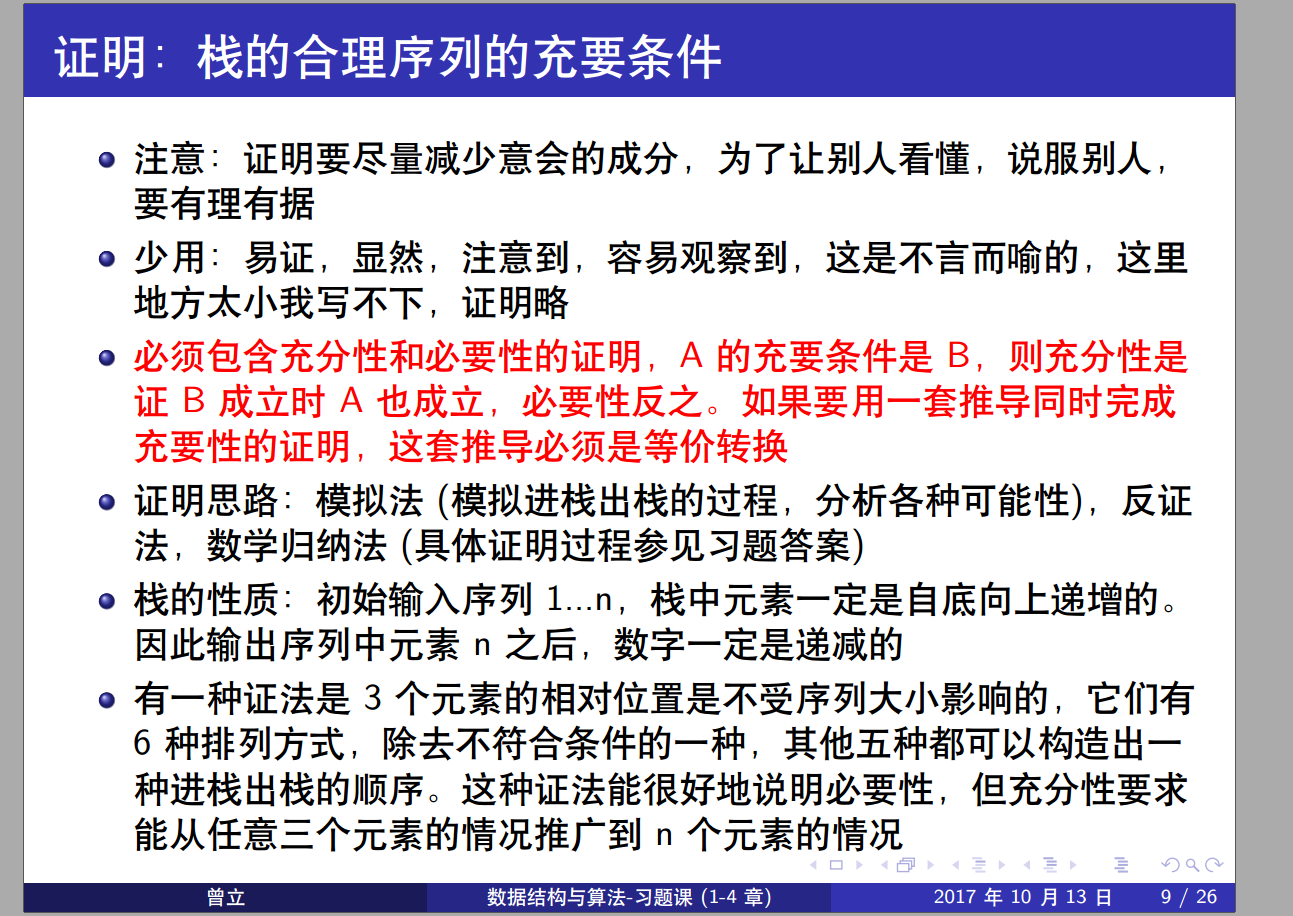
而pj，pk之所以被压在栈内，是因为必然有一个pi，pi > pk，但是pi要在pj,pk之前出栈（即i<j）；(说明：如果不存在pi，则表明栈顶前两个元素分别是pj和pk，则可以在pj入栈的时候立即出栈而不会出现pj出栈的矛盾了)

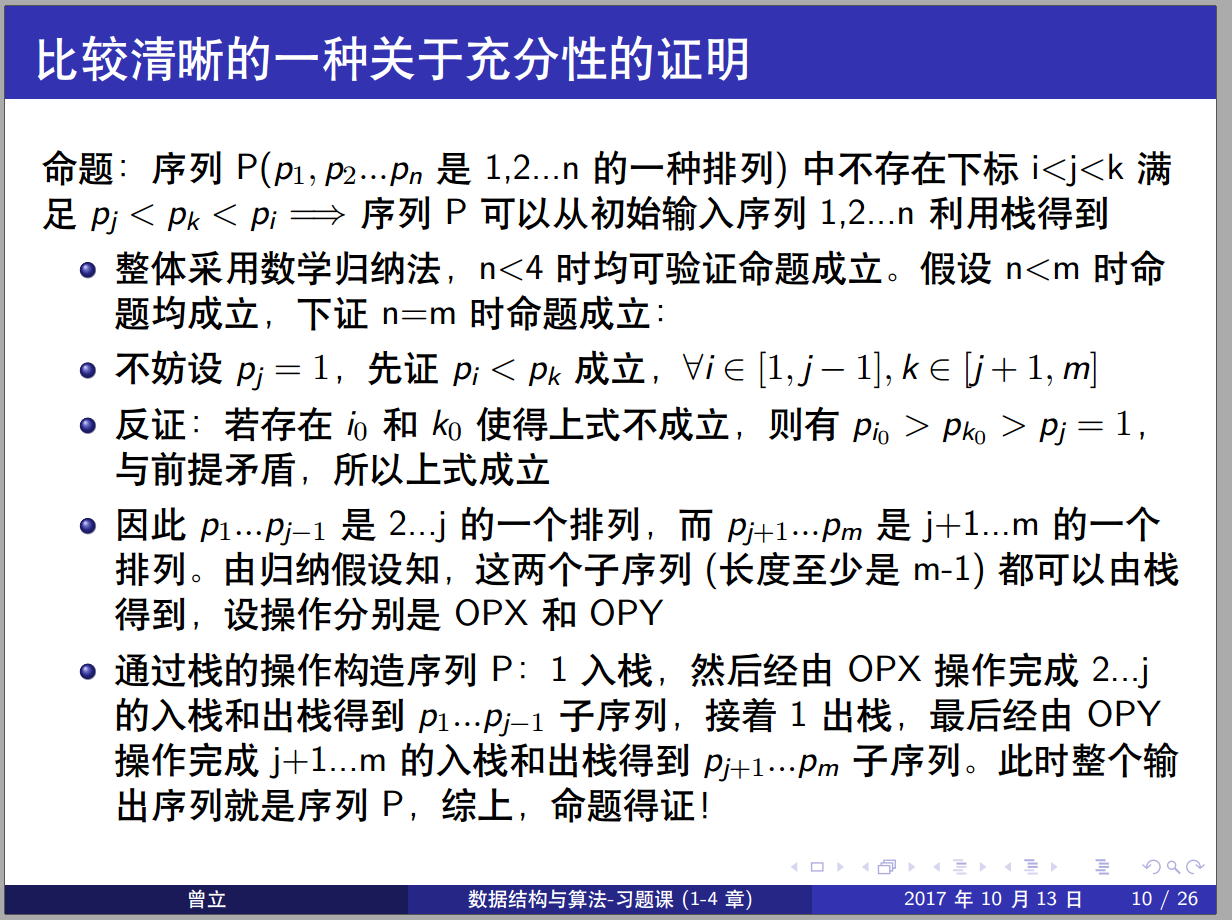
所以我们就找到了i,j,k 满足i < j < k，但 pj < pk <pi。这与条件矛盾；

所以若条件成立，p1…pn一定是合法序列。

注意：pj, pk, pi对应的是元素的大小（这里也对应着输入序列的位置），而i, j, k对应的是输出序列的先后位置。

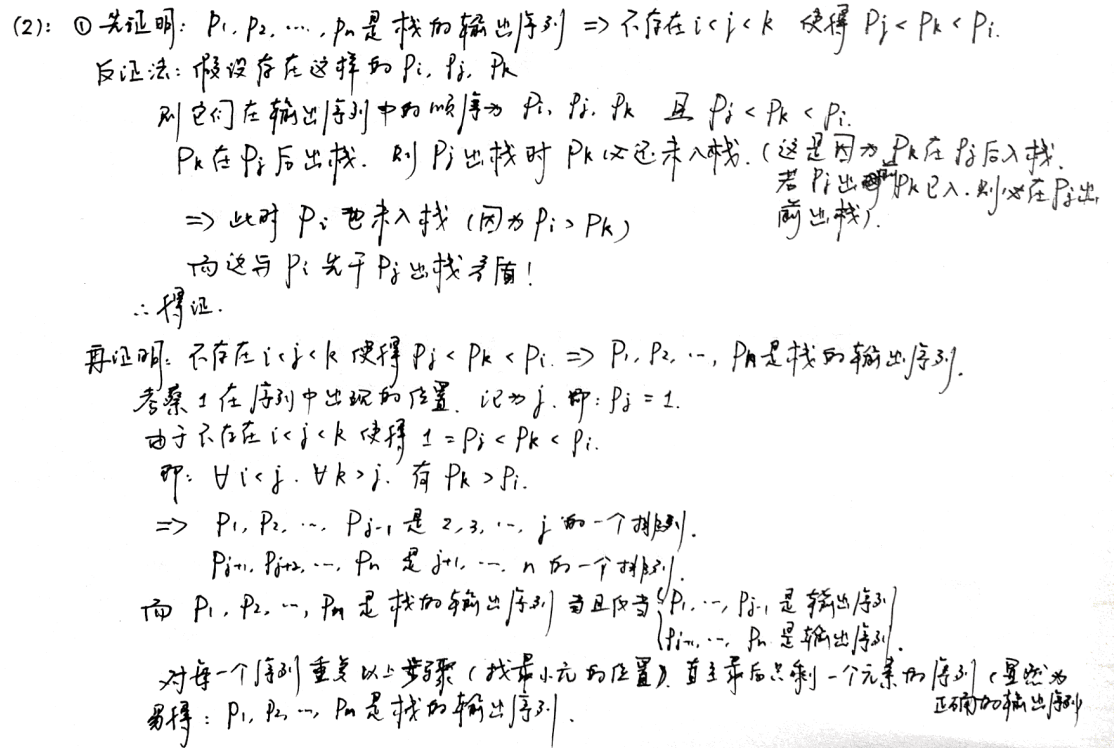
讨论：第一问中的状态标记的方式，确实可以算出对应的合理方案的总数。但同一个2n位的状态码，可能对应着多个具体的数的排列，其中有且只有一个是合理的。比如考虑依次输入3个数1,2,3，输出序列可以是321或者312，它们对应的状态码都是111000.显然，321是合理的方案，而312是不可能出现的。（状态码里考虑的限制条件只是不能从空栈中取数据，并没有考虑给定的元素输入和输出的对应关系）



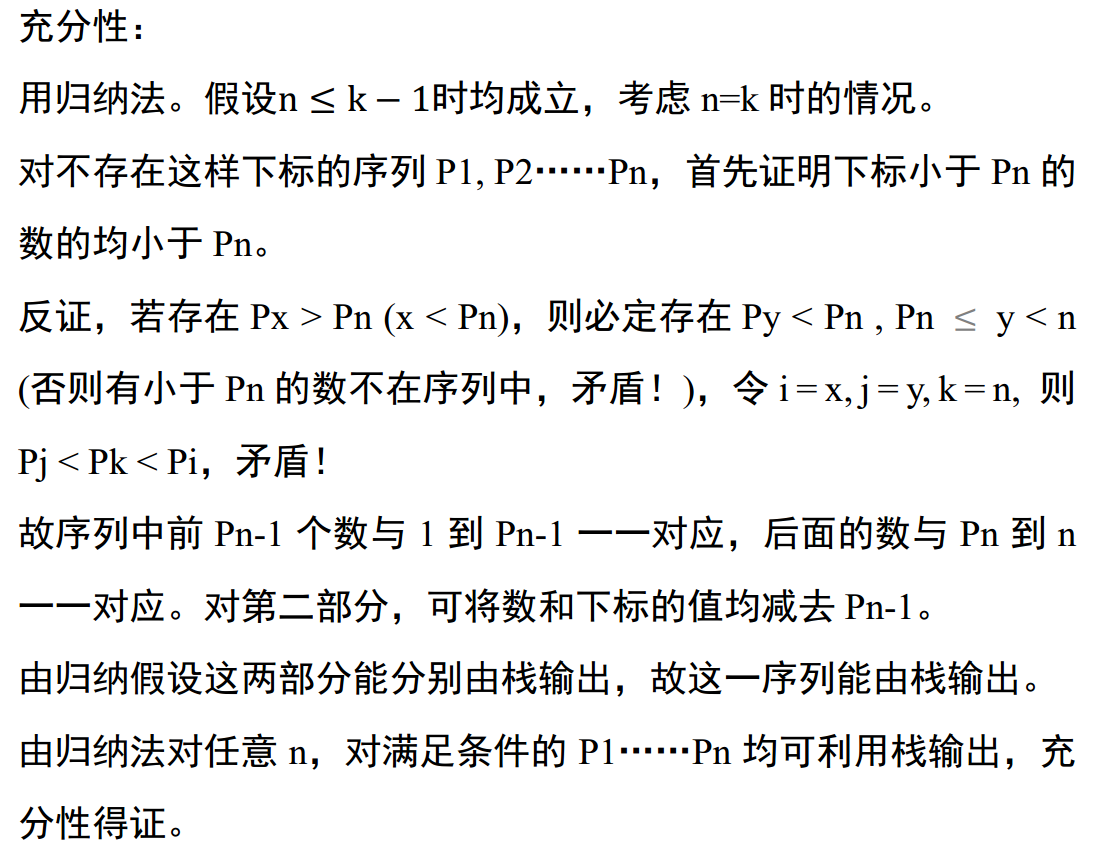


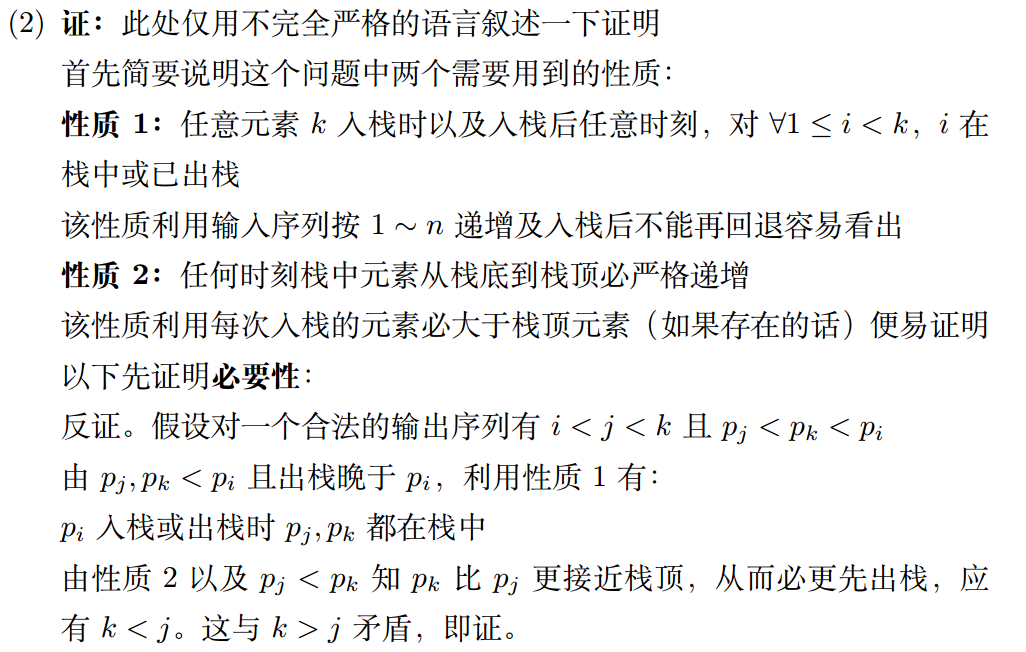
这里给出同学们的一些证明方法：

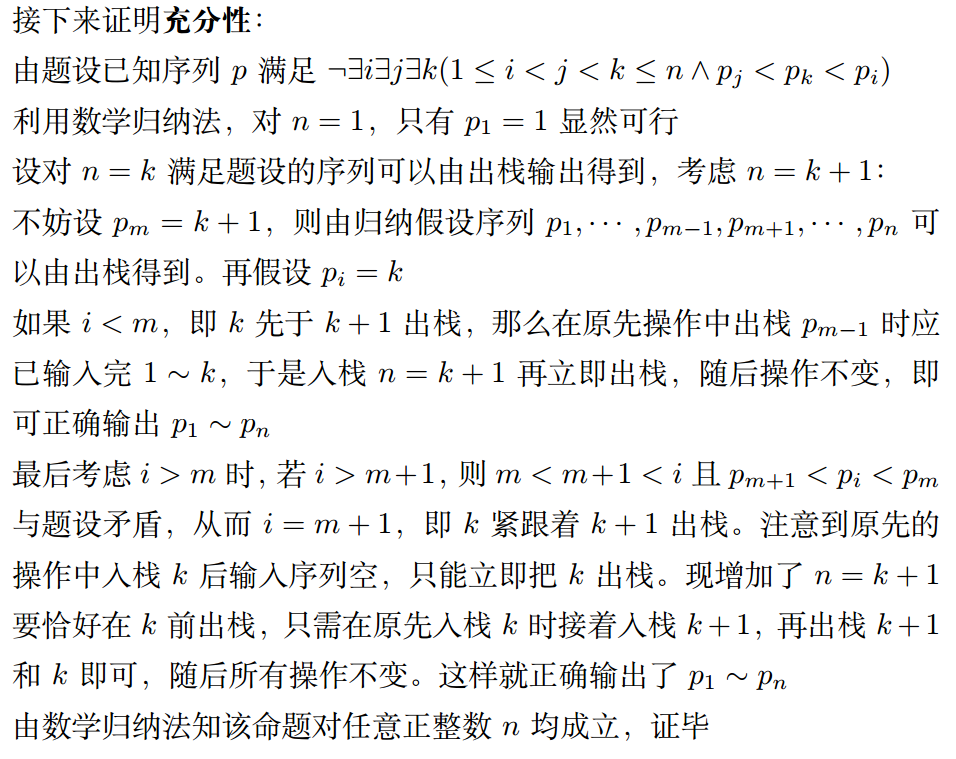
有些同学枚举了3个数的六种序列，通过模拟栈的出入顺序，证明了其中5种都与一个合法调度一一对应。这种方法可以很好地说明必要性，但对充分来说，最好还要说明下怎么从任意3个数的合法性得到整个序列（长度为n）的合法性。

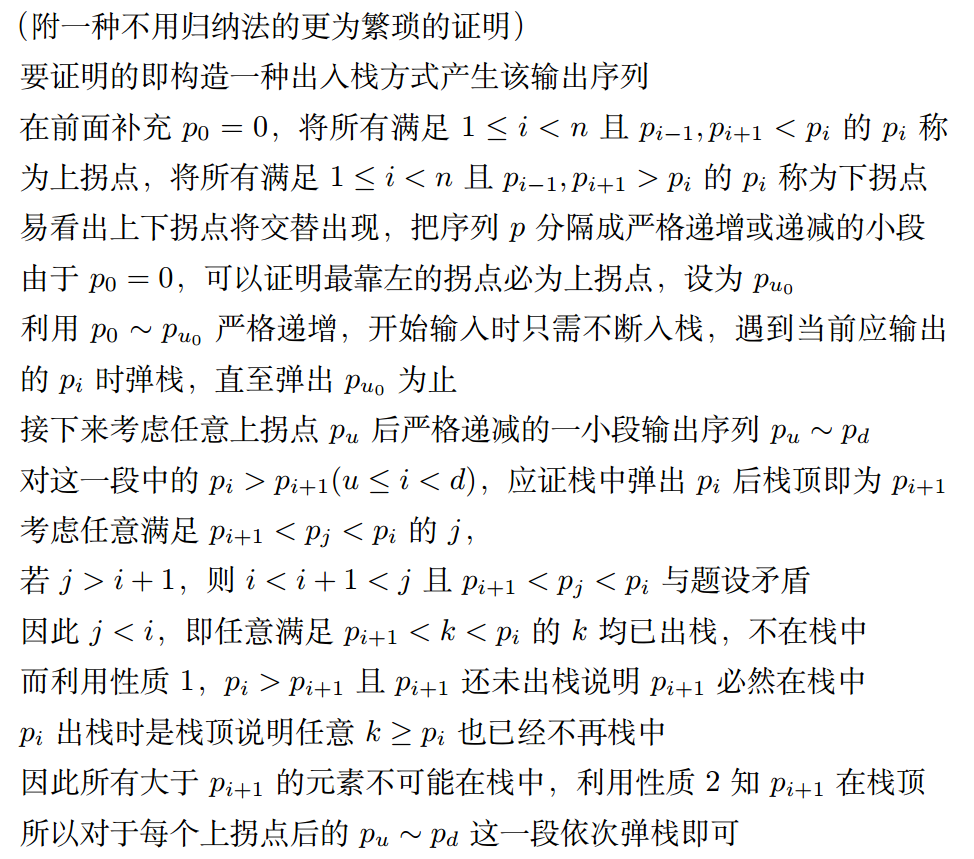


上面这个证明如果用数学归纳法来叙述，会更好。最好说明下如何构造栈的出入顺序，使得2,3…j在1之前输出，而j+1,j+2…n在1之后输出。









# 附录

相关问题：<https://baike.baidu.com/item/%E5%8D%A1%E7%89%B9%E5%85%B0%E6%95%B0/6125746?fr=aladdin>

<http://www.cnblogs.com/wuyuegb2312/p/3016878.html>

卡特兰数的推导

<http://m.blog.csdn.net/acdreamers/article/details/7628667>

问题描述:  
12个高矮不同的人，排成两排，每排必须是从矮到高排列，而且第二排比对应的第一排的人高，问排列方式有多少种？  
这个笔试题，很YD，因为把某个递推关系隐藏得很深。  
  
问题分析:  
我们先把这12个人从低到高排列,然后,选择6个人排在第一排,那么剩下的6个肯定是在第二排.  
用0表示对应的人在第一排,用1表示对应的人在第二排,那么含有6个0,6个1的序列,就对应一种方案.  
比如000000111111就对应着  
第一排：0 1 2 3 4 5  
第二排：6 7 8 9 10 11  
010101010101就对应着  
第一排：0 2 4 6 8 10  
第二排：1 3 5 7 9 11  
问题转换为，这样的满足条件的01序列有多少个。  
观察1的出现，我们考虑这一个出现能不能放在第二排，显然，在这个1之前出现的那些0,1对应的人  
要么是在这个1左边，要么是在这个1前面。而肯定要有一个0的，在这个1前面，统计在这个1之前的0和1的个数。  
也就是要求，0的个数大于1的个数。

另一个很YD的问题:  
有编号为1到n(n可以很大，不妨在这里假定可以达到10亿)的若干个格子，从左到右排列。  
在某些格子中有一个棋子，不妨设第xi格有棋子(1<=i<=k, 1<=k<=n)  
每次一个人可以把一个棋子往左移若干步，但是不能跨越其它棋子，也要保证每个格子至多只有一个棋子。  
两个人轮流移动，移动不了的为输，问先手是不是有必胜策略。