PA1-1-1 Filename

本题和最长公共子序列有关。首先可以证明，答案为$n + m - 2 * LCS(A, B)$。

但n，m数据范围较大，按普通方法求解LCS无法满足时间要求。

注意在 $n + m - 2 * LCS(A, B) > k$ 时，我们不需要求解具体的LCS长度，直接汇报无解即可。

考虑LCS的动态规划状态$F[i][j]$表示串A的前i个字符和串B的前j个字符的最长公共子序列。

观察到，如果 $i - j > k$, 这样的状态必然无法转移到$n + m - 2 * LCS(A, B) > k$的状态。

可以理解为：如果我们尝试让串A的前i个字符和串B的前j的字符匹配，至少需要付出两个串长度差距的代价。

因此，我们可以只计算$O(nk)$个状态，以满足数据范围内的时间要求。

PA 1-1-2 Interview

使用链表解决本题的方法是平凡的，这里介绍一种使用数组解决本题的方法。

本题描述的过程，可以按时间逆序处理，等价为：

- 现在有n把椅子，初始时均为空。从“最后来的应聘者”开始入座；
- 所有椅子顺时针依次为第1、2、....n个面试的座位。“最后来的应聘者”坐到第1个面试的座位上；
- 入座时，从“后一个来的应聘者”的座位出发，沿顺时针方向，找到的"第m个空座位"，即为前一个应聘者的座位。

算法的核心为，在O(1)的时间复杂度内，找到“之后的m个空座位”。

为此，可以采取一种“分块数组”的方式，将大约$\sqrt{n}$ 个元素作为一组，记录每一块内的“空座位个数”。

从而，可在$O(\sqrt{n})$的时间复杂度内找到下一个“空座位”。实际常数较小，在数据范围内可以满足时间要求。

PA 1-2-1 Gift

题目大意： n 组数对 (a_i, b_i)，每组中选一个，计算总和不超过P的选法方案数。

直接的思路是进行枚举，观察数据范围发现 $O(2^n)$ 的复杂度太高。

使用中间相遇的思想来降低复杂度。枚举出前 $\frac{n}{2}$ 组数对的所有方案，存在数组C中。对于另 $\frac{n}{2}$ 组数对，同样枚举所有 $2^{\frac{n}{2}}$ 种方案，对于每一种方案，在C中查询组合后合法的方案数。结合排序和二分查找，将复杂度从 $O(2^n)$ 降低至 $O(n * 2^{\frac{n}{2}})$，可以完成本题的时间要求。

[拓展思考]
在枚举方案时，可以使用归并的思想在 $O(2^{\frac{n}{2}})$ 时间内完成有序方案列表的枚举——**对于前k组生成的有序方案列表，归并a_{k+1}, b_{k+1}生成的两组有序方案列表。** 同时，在查询时可通过遍历的方式直接计算得到合法方案数，无需二分查找。此时总的时间复杂度将降低至 $O(2^{\frac{n}{2}})$。

PA 1-2-2 Graphics

本题的思路是二分查找。

首先将 x, y 的点列排序，得到 n 条线段 $x_i \leftrightarrow y_i$。

对于查询点 P，通过向量叉积的方式可以判断 P 与线段的相对位置关系（左侧/右侧）。以此作为比较器，可在区域中实现二分查找。