Problem Tutorial: "树"

先考虑一个经典问题：给一棵树，计算其自同构的数量。

假如是有根树意义下（即边集全等改为父亲数组全等）的同构，算法就是：对每个点的每个子树按同构划分出若干等价类，将每种等价类大小的阶乘乘入答案。

在无根树意义下，我们需要找到一个点只能对应到自己。当重心唯一时，重心满足条件，否则可以在两个重心相连的边上加一个点，答案不变。

那么构造便很简单了。有解条件是 K 可表为一堆阶乘的乘积，大体是将一堆菊花串成链，之后考虑重心可能还要进行调整。

Problem Tutorial: "谜题"

考虑所有排列的第一位：A_1 出现了 $n - A_1$ 次，$A_1 - 1$ 出现了 $A_1 - 1$ 次，另一个是 A_2 或 $A_2 - 1$. 我们找出所有可能的 A_1，显然只有 $O(1)$ 种。

之后你可以枚举哪个排列是删掉 A_1 得到的，通过预处理一些贡献去快速计算 hash 值可做到每次 $O(n)$ 判断；或者用类似的办法讨论次数，可由 A_i 推到 A_{i+1}. 其他一些奇怪的做法复杂度可能也是 $\tilde{O}(n^2)$ 的。

如果不知道 A_1 只有 $O(1)$ 种而尝试去枚举，复杂度会多一些 n 得到部分分。

Problem Tutorial: "打字机"

显然是求压栈的最小次数。观察一下栈的形态：

- 一旦压入一个字符，下一步一定是打印它；
- 如果当前字符与栈顶字符相同，显然直接打印，于是栈的相邻两个不会相等；
- 如果当前字符与栈顶下面一个相等，简单调整可证压栈不优于弹栈，从而栈中隔一个也不会相等。

这样栈一定形如 `XYZXYZXY`$\cdots$，状态数是 $O(|S|)$ 的。直接 DP，记录打印到了哪里和栈的形态，复杂度便为 $O(|S|^2)$.

用上面的部分性质去优化爆搜可以得到一些部分分；$|S| \leq 100$ 考虑第一次压入的字符在什么时候弹出，将原序列分成了若干段，可对此进行区间 DP.