序列异或

折半。

从 1 到n枚举第三个元素的位置k。对于k，拿一个数组cnt_x记录一下$i < j < k$，并且$a_i \text{ xor } a_j = x$的对数。这个时候我们只要枚举最后一个元素l，然后查询$cnt_{a_k \text{ xor } a_l}$即可。枚举完这个$k$，考虑$k+1$ 的时候，只需要将$a_i \text{ xor } a_k (i < k)$加入到$cnt$中即可。

时间复杂度$O(n^2)$。

乘法破译

容易发现，对于 0，那么对应的行列一定都是相同的。然后对于 $1 \sim p-1$ 中的每个元素i，那么它在乘法表中，对应的十位一定是 $0 \sim i-1$，出现了i个不同的数字，直接统计十位的个数即可。

时间复杂度$O(n^2)$。

幸运数字

继续考虑折半搜索，统计答案肯定是每一位分开来统计。

如果对10^{k+1}取模之后，最高位是 4，那么一定满足左右两边取模后加起来在$[4 \times 10^k, 5 \times 10^k)$和$[14 \times 10^k, 15 \times 10^k)$这两段区间内。

如果左右两半按10^{k+1}取模之后排序，就可以线性扫描了。

但是直接排序，时间复杂度会变成$O(2^{n/2} n \log W)$，不一定好通过。

于是，我们按k从小到大做，每次排序相当于在对10^k取模的基础上，加上了最高位，这个可以用类似于双关键字排序的基数排序做法在$O(2^{n/2})$解决。

所以总的时间复杂度是$O(2^{n/2} \log W)$的。

排列计数

插板法dp。从小到大插入每个元素，考虑当前序列被分成了多少段。

每次插入一个元素a_i，如果它和前面一段的元素拼起来，那么它一定比前面的元素大，那么对答案的贡献是a。否则它需要会被一个更大的元素拼起来，它的贡献是$-a$。注意要特别考虑这个元素如果就是整个序列的端点，那么不会有贡献。方案数通过简单的计数可以算出来。

对于前k大，我们只要保留一下前i个元素，被划分成了j段，当前贡献的前k大和对应的方案即可，合并的时候用归并排序归并一下。

时间复杂度$O(n^2k)$。