草莓(guiltiness)

要变成 1×1 的话，每行每列都要掰一次。所以我们相当于需要给 x_i 和 y_i 一起安排一个顺序，使费用最小。在最终的操作序列中，要使得费用最小，x_i 内部显然应该单调递减，y_i 也是。对于一个在操作序列中相邻的 x_i 和 y_j，不妨假设目前 x_i 恰好在 y_j 的前一位。考虑把 x_i 和 y_j 交换，发现交换后更优的条件是 $y_j > x_i$。所以最优的操作序列就是把 x_i 和 y_i 放在一起从大到小排序。

三色(color)

$\mathcal{O}(n^4)$

记 $f_{i,j,k,l}$ 表示填到第 i 位，上一个红色/蓝色/紫色在 $j/k/l$ 的方案数。

$\mathcal{O}(n^3)$

发现 j,k,l 中必有一个是 i，所以可以少记一维，记 $f_{i,j,k}$ 即可。

$\mathcal{O}(n^2)$

状态应该没有优化空间了，考虑优化转移。

把 j,k 两维看成一个矩阵，当 i 转移到 $i+1$ 时，如果 $i+1$ 位和第 i 位的颜色一样，对应的操作是把 i 的矩阵复制到 $i+1$ 处；如果 $i+1$ 位是第一种不一样的颜色，则是把矩阵的每列累加到 $i+1$ 行的位置；如果 $i+1$ 位是第二种不一样的颜色，则是把矩阵的每行累加到 $i+1$ 列的位置。

通过维护矩阵每一行和每一列的和，这些操作可以 $\mathcal{O}(n)$ 处理。

考虑怎么处理限制，当 $x_i = 1$，对应的操作是把除了左上角的一个子矩阵外都清空（令左上角为第一行第一列）；当 $x_i = 2$，是除了两个子矩阵外都情况；当 $x_i = 3$，是除了右下角的一个子矩阵外都清空。

发现清空之后每一行保留下来的都是一个区间，可以每一行用 deque 维护还没被清空的位置，只要每个点只会被清空一次时间就是正确的。

注意到行和列是对称的，所以只维护一个也可以。

博弈（game）

a_i 互不相同

当 a_i 互异时，保留的三个数也一定不同，我们来研究一下必胜条件：

显然剩下的三个数的绝对大小没有意义，不妨把三个数 a,b,c 排序之后，对相邻的数做差分，得到 x,y，因为 a_i 互不相同，所以 $x,y>0$。

如果我们操作 a,b，对 x,y 的影响是进行若干次的 `x-=2,y+=1`，操作 b,c 同理，限制是不能小于 0。

如果我们操作 a,c，则是若干次的 `x-=1,y-=1`，并且当其中一个数减到 0 的时候，接下来就会变成 `x-=2,y+=1`。

可以注意到，只要 $x,y>0$，必然存在必胜策略：操作 a,c 直到 x,y 出现 0，如果目前这个状态是必败状态，那么在此停止即可；如果是必胜状态，我们继续模仿必胜策略的操作即可（因为当 x,y 存在 0 时，接下来只有一种操作方法）。

所以答案为 $\binom{n}{3}$。

$\sum n \le 400, a_i \le 10^5$

接下来研究三个数是 a,a,b 的情况：要想胜利，我们只能把 a,b 操作成 $\frac{a+b}{2}$（做不到的话必败），不然出现三个不同的数对手就必胜了。可以发现我们只关注 $|a-b|$，令 $f(n)$ 为 $|a-b|=n$ 时的胜负情况，$f(n)$ 只能由 $f(n/2)$ 转移过来。

先计算出 $f(n)$，再枚举保留的三个数计数即可。

$\sum n \le 3\times 10^5$

结果分析可以知道，$f(n)$ 为必胜当且仅当 n 最低位的 1 是第偶数位。

所以枚举 a，枚举 $a-b$ 的最低位的 1，用 map 查找 b，时间两个老哥。

$\sum n \le 3\times 10^6$

可以在 trie 上 dp 来计数，时间一个老哥。

实现的好的话说不定 map 或者 unordered_map 也可以卡过去（

后缀数组（sa）

$n,m \le 1000$

先处理出最终的序列，考虑怎么计数。

首先一定有 `s[sa[i]]<=s[sa[i+1]]`，进一步可以发现 `s[sa[i]]=s[sa[i+1]]` 或者 `s[sa[i]]+1=s[sa[i+1]]`，要不然就不是合法的字符串。如果不相等的话这里的限制就已经满足了，如果相等的话还要比较下一位。但是可以发现比较下一位其实很简单，因为我们知道每个后缀的排名，只要满足 `rk[sa[i]+1]<rk[sa[i+1]+1]` 即可。

所以每一个位置都会有一种或者两种可能，那么答案就是 2^k。

$n \le 10^5$

用平衡树处理出操作过后的序列，再进行计数。

$n \le 10^9$

可以注意到满足 $|sa_i - sa_{i+1}| = 1$ 的一长段是可以快速处理的，因为如果这一位相等要比较下一位的话，下一位除了边界情况都在附近，可以快速统计。

我们仍然使用平衡树维护，平衡树的节点是上述的一个连续段，题目中的两种操作每次最多产生两个新的连续段，所以最终连续段的数量是 $\mathcal{O}(m)$ 的。

得到这些连续段后，先统计段内的答案，再把段的边界的贡献处理一下即可。

复杂度一个老哥。这题题解说的比较笼统，具体的处理细节留给选手自己考虑（