-1. 叠甲

本次讲课内容包括但不限于以下内容：

① 退役云 `OIer` 口胡爆炸。

② 难度不够 `NOI`，是人看了都一眼秒。

③ `trivial` 且 `dirty` 的数学。

④ 这 `TM` 不是原题吗？

第⑤人格启动。

⑥ 提前结束，丢人下班。

主要的内容集中在计数，希望能讲的趣味一点（退役云`OIer`），可能会过于 `trivial` 请大家谅解（）。

可能有一些题没有明确的 `ref`，是出自一些【】的原因，大家能体会到那股劲就好）

由于一些【】原因，题单选的可能有点乱，最重要的是希望大家能感受到那股劲）

0. 从多项式到级数

多项式：**有限项**，可以被写成：$\sum_{i=0}^{n} f_i x^i$。显然对加减乘封闭。请注意一点，我们一般不把无穷求和的东西叫做多项式，更正规的叫法应是："级数"/"形式幂级数"等名称。

似乎在背包，卷积的视角下多项式是足够有用的，单不对四则运算封闭是~~不美好的~~。

为了使"多项式"关于除法封闭，我们引入了形式幂级数，与通常的级数不同，我们并不关系其敛散性，而只将其看作一类关于多项式在系数上的形式上的延拓。

$$\frac{1}{F(x)} = G(x) \iff F(x)G(x) = 1$$

例如：

$$\frac{1}{1-x} = 1 + x + x^2 + \cdots$$
$$(1-x)(1+x+x^2+\cdots) = (1+x+x^2+\cdots) - (x+x^2+x^3+\cdots)$$

如何求 $\frac{1}{F(x)}$，可以考虑列出级数，反解出每一项：

$$\begin{aligned}
&\frac{1}{F(x)} = G(x) \iff F(x) = G(x) \\
\iff &\sum_{i=0} f_i x^i \sum_{j=0} g_j x^j = 1 \\
\iff &f_0 g_0 = 1 \cap \forall i > 0, \sum_{j+k=i} f_j g_k = 0 \\
&g_i = -\frac{1}{f_0} \sum_{j+k=i, k\neq i} f_i g_k
\end{aligned}$$

此时可以在 $\mathcal{O}(n^2)$ 的复杂度内递推出 g_i 的每一项，可以发现我们同时得到了多项式的逆的**存在性**。即，当 $g_i \neq 0$ 时，$F(x) = \sum f_i x^i$ 存在。

Q：为什么要学习形式幂级数?

当我在学OI的时候，曾经有人把多项式比作【数据删除】，我想这的确是一种污名化。的确，过度考察FFT,多项式全家桶的确会让 OI 变成“默写竞赛”。但其实一些组合计数题目通过形式幂级数的手段，会得到更“有结构的”(structured) 的做法，会看到更“形象”的解释。以管窥豹。

---

bostan-mori †

我们经常会遇到求解形如 $[x^n]\frac{G(x)}{F(x)}$ 的问题，其中 n 可能十分巨大，而相反的 F 的度数反而可能并不是很大。

如果你有一些**多项式基础**的话，不难去理解 $\frac{F(x)}{G(x)}$ 对应着一个**线性递推**：我们假设 $H(x) = F(x)/G(x)$，有方程： $H(x)G(x) = F(x)$。展开后会发现，h_i 对应着一个长为 $m = \deg G(x)$ 的递推式。 我们先不妨假设 $[x^0]G(x) = 1$。

$$\begin{bmatrix} h_n \\ \vdots \\ h_{n-m+1} \end{bmatrix} = \begin{bmatrix} -g_1 & -g_2 & \cdots & -g_m \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} h_{n-1} \\ \vdots \\ h_{n-m} \end{bmatrix}$$

而 F 决定了 H 前 $\deg F(x)$ 项的修正系数。

经典解决常系数线性递推的方式是考虑特征多项式。

$$\vec{h}_n = M\vec{h}_{n-1} \implies \vec{h}_n = M^n \vec{h}_0$$

> 特征多项式：††
>
> 我们来补充一些线性代数的基础。我们引入特征多项式的原因就是：特征多项式蕴含了一个线性变换的性质，而这个性质就是下文要提到的 cayley-hamilton 定理。
>
> 我们定义一个矩阵的特征值 λ，与其对应的特征向量为使得 $Av = \lambda v$ 成立的 λ, v。也就是 $(A - \lambda I)v = 0$。
>
> 我们知道，如果 $v \neq 0$ 且满足上式，当且仅当 $A - \lambda I$ 可逆，即 $\det(A - \lambda I) = 0$。所以我们将求解特征值转化为求解多项式：$f(\lambda) = \det(A - \lambda I)$ 的根。 而cayley-hamilton给出了一个微妙的结果，我们有 $f(A) = 0$。0矩阵。
>
> 由于 cayley-hamilton 定理的证明涉及到大量线性代数内容，我们给出一个不尽严谨的感性理解。
>
> 我们假设有 $f(\lambda)$ 有 n 个根（很可惜，这并不一定正确；对于不正确的情况会有更为复杂的证明），我们假设他们是 $\lambda_1 \cdots \lambda_n$，每一个 λ_i 对应一个特征向量 v_i。我们考虑 $f(A)v = f_0(A)(A - \lambda_i)v = 0$，所以我们证明了 $f(A)$ 零化了每一个特征向量。在我们的假设中，$v_1 \cdots v_n$ 线性无关，所以 $f(A)$ 零化整个空间 $\mathbb{R}^n$。$f(A)$ 只可能为 0。

根据Cayley-Hamilton 定理，我们有 $f(t) = \det(A - tI)$,那么有 $f(A) = 0$。（这里）所以：

$$g(t) = t^n \bmod f(t) \iff g(t) + f(t) \cdot h(t) = t^n$$
$$g(M) = M^n \bmod f(M)$$

所以我们只需计算：$g(M)\vec{h}_0$ 即可。注意到当前我们只需要求出该向量的一项的值看，这部分可以做到线性，所以总体可以做到复杂度 $O(M\log N\log M)$。当然如果我们暴力做多项式取模的话，可以做到 $O(M^2\log N)$，似乎也是足够好的复杂度。

正（hao）戏开始，new method

这个思想其实出现在很多算法中，比如说 `MCMF(Min Cost Maximum Flows)` 的 `scaling` 算法。（虽然似乎都是很冷门算法）

$$[x^n]\frac{P(x)}{Q(x)}=[x^n]\frac{P(x)Q(-x)}{Q(x)Q(-x)}$$

我们发现，此时 $Q(x)Q(-x)$ 在函数意义上，是一个偶函数！（$F(x)=F(-x)$），所以在级数（展开式/解析式）的意义上 $Q(x)Q(-x)$ 没有偶数次项，所以我们也能推出 $\frac{1}{Q(x)Q(-x)}$ 也之可能存在偶数项。好处是什么？我们此时没必要关注分子与 n 奇偶性不同的项了。同时，我们也可以将 $n\leftarrow\left\lfloor\frac{n}{2}\right\rfloor$。而分子，分母的项数都**不会增加**。于是我们递归 $O(\log_2 n)$ 次，就会结束。

$$\begin{aligned}&Q(x)Q(-x)=U(x^2)\\ [x^n]&\frac{P(x)Q(-x)}{Q(x)Q(-x)}=[x^n]\frac{T_0(x^2)+xT_1(x^2)}{U(x^2)}=[s^{n/2}]\frac{T_0(s)}{U(s)}\end{aligned}$$

具体推到形如上式，我们假设此时 $2\mid n$。

---

① P8XCoinChange

> `ref : topcoder srm 527` 很可惜，在笔者写这篇讲稿（？）时，`topcoder`似乎已经倒闭了。很巧合地一点是，笔者曾经将这道题搬到模拟赛里。

题目大意：你有 n 种货币，问用这些货币组成 m 的方案数，保证 $a_i<a_{i+1}$ 且 $a_i\mid a_{i+1}$。

$n\le 50,m\le 10^{18}$。

我们给出一种生成函数的做法。

$$\begin{aligned}&[x^m]\frac{1}{1-x}\cdot\frac{1}{1-x^{a_1}}\cdot\frac{1}{1-x^{a_1a_2}}\cdots\\ =&[x^m]\frac{\frac{1-x^{a_1}}{1-x}}{1-x^{a_1}}\cdot\frac{1}{1-x^{a_1}}\cdot\frac{1}{1-x^{a_1a_2}}\cdots\end{aligned}$$

这样，我们可以提取分子的 a_1 合适的项，我们将问题转化为：

$$\begin{aligned}&[x^m]\frac{p(x)}{(1-x)^c}\cdot\frac{1}{1-x^{a_1}}\cdot\frac{1}{1-x^{a_1a_2}}\cdots\\ =&[x^m]\frac{p(x)\left(\frac{1-x^{a_1}}{1-x}\right)^c}{(1-x^{a_1})^c}\cdot\frac{1}{1-x^{a_1}}\cdot\frac{1}{1-x^{a_1a_2}}\cdots\end{aligned}$$

我们不妨假设，$m=ka_1+r$。那么，我们去提取分子的 $[x^{ta_1+r}]$ 项。

注意到 $p(x)$ 的长度不会太长，$\deg p(x)\le c$，所以我们不难得到一个 $\mathcal{O}(c^3)$ 的做法。

我们暴力枚举 $P(x)=\sum_{i=1}^{c}p_ix^i$ 的 i，x^{ja_1} 的 j，然后去计算 x^{ja_1+i} 对 $[x^{ta_1+r}]$ 的贡献（枚举 t）。

详细过程 †

我们不妨记 $x=a_1$，下式是计算 $[x^{tx+r}]$ 的系数。

$$
\begin{aligned}
&\sum_{i=0}^{c} p_i \sum_{j=0}^{c}(-1)^j\binom{c}{j}\binom{tx+r-jx-i+c-1}{c-1}\\
&\sum_{j=0}^{c}(-1)^j\binom{c}{j}\left[\sum_{i=0}^{c} p_i\binom{(t-j)x+r-i+c-1}{c-1}\right]
\end{aligned}
$$

预处理出来：$f(t-j)=[\cdot]$（后面那个括号里的东西）。复杂度 $\mathcal{O}(c^2)$。

然后枚举 t,j 复杂度 $\mathcal{O}(c^2)$。总复杂度 $\mathcal{O}(c^2)$。

`Fun fact`（当时写的题解）：

首先生成函数直接做可能有些困难。

---

② Convex Sequence

> `ref : AGC049D`
>
> 本题不是 `bostan-mori` 只是想回答一下“为什么要学习形式幂级数”。

题目大意：问有多少个长为 n 的序列 a 满足：

1. $\sum a_i = M$
2. a_i 凸，即 $2a_i \le a_{i-1}+a_{i+1}$

$n,m\le 10^5$

我们考虑如果保证序列单调增的时候怎么做。

$a_1,a_2,\cdots,a_n$，我们设 $b_i=a_i-a_{i-1}$，b 需要满足单调增，我们不妨假设 $c_i=b_i-b_{i-1}$。

那么：

$$
\begin{aligned}
\sum_{i=1}^{n} a_i=\sum_{i=1}^{n}\sum_{j\le i} b_j &= \sum_{j=1}^{n}(n-j+1)b_j\\
&=\sum_{j=1}^{n}(n-j+1)\sum_{t\le j} c_t\\
&=\sum_{t=1}^{n}\frac{(n-t+1)(n-t+2)}{2}c_t
\end{aligned}
$$

所以我们写出此时的生成函数：

$$
\prod_{k=1}^{n}\frac{1}{1-x^{\frac{k(k-1)}{2}}}
$$

注意到，实际上 $k\ge O(\sqrt{n})$ 时，我们时不需要考虑的，因为 $\frac{1}{1-x^t}=1+x^t h(t)$。

那该如何计算 $F_1(x)=F_0(x)\cdot\frac{1}{1-x^t}$ 呢？（此时已知 $F_0(x)$，t）我们可以考虑通过方程 $F_1(x)(1-x^t)=F_0(x)$ 反解出来 F_1。

具体的，我们提取 $F_1(x)(1-x^t)$ 的 $[x^n]$ 项系数，不难发现是 f_n-f_{n-t}，这样我们就可以获得一个递推式，在 $\mathcal{O}(n)$ 时间内求出 $F_1(x)$。可喜可贺。

那如何处理不单调增的呢？我们可以考虑枚举最小值的位置（第一个 $a_i < a_{i-1}$ 且 $a_i \le a_{i+1}$ 的位置）将序列分成两半考虑。此时的生成函数形如：

$$\prod_{k=1}^{a} \frac{1}{1-x^{\frac{k(k-1)}{2}}} \prod_{k=1}^{b} \frac{1}{1-x^{\frac{k(k-1)}{2}}}$$

而其中本质不同的 a, b 对只有 $\mathcal{O}(\sqrt{n})$ 种，并且每次修改形如加入/删除一个 $\frac{1}{1-x^t}$ 我们都可以在 $\mathcal{O}(n)$ 的时间复杂度内完成。

总时间复杂度是 $\mathcal{O}(m\sqrt{m})$。

1. 容斥

众所周知的公式是：

$$\left|\bigcup_{i=1}^{n} S_i\right| = \sum_{i=1} S_i - \sum_{i_1<i_2} |S_{i_1} \cap S_{i_2}| + \sum_{i_1<i_2<i_3} |S_1 \cap S_2 \cap S_3| - \cdots$$

容斥最最基本的想法是什么？是把正确的东西的贡献系数算成 1（或某个常数），将错误的东西的贡献系数算成 0。

首先考虑一个最简单的问题：

③ 无题

题目大意：如何现在有一个序列 a_n，$0 \le a_n \le m$，请问有多少个序列满足：不存在两个连续的`0`？

$$\begin{aligned} f_n &= \sum_{k=1}^{n} \sum_{j=1}^{k} (-1)^k \binom{k-1}{j-1} \binom{n-k}{j} m^{n-k-j} \\ &= \sum_{u=1}^{n} \sum_{k=1}^{u} (-1)^k \binom{k-1}{(u-k)-1} \binom{n-k}{u-k} m^{n-u} \end{aligned}$$

怎么办？

我们写出生成函数：

$$\begin{aligned} F(x) &= m \cdot x + \sum_{i=2} (-1)^i x^i = m \cdot x - \frac{x^2}{1+x} \\ G(x) &= 1 + F(x) + F(x)^2 + \cdots = \frac{1}{1-F(x)} = \frac{1+x}{1-(m-1)x+(m-1)x^2} \end{aligned}$$

是一个二阶递推式。符合我们的直觉。

---

④ `Jiangly` 的排列数数题

> ref : [EI blogs]()

题目大意：问对于所有长为 n 的排列，有多少排列存在一个连续上升段 $\ge k$。对所有 k 回答，对大质数取模。

期望复杂度 $\tilde{O}(n^2)$。

我们首先容斥为不存在。我们来考虑一种形式化的描述“容斥系数”的方式。本题中，我们希望得到的连续段的长度的生成函数为：

$$G(x)=x+x^2+\cdots+x^{k-1}=\frac{x-x^k}{1-x}$$

那么，我们假设容斥系数的生成函数是 $F=\sum f_i x^i$ ，即，如果长度为 i 时容斥系数为 f_i。

需要满足的方程是（注意，此时我们假设 F 未知）：

$$\begin{gathered}F+F^2+\cdots=\frac{x-x^k}{1-x}\\ \frac{1}{1-F}-1=G(x)\implies F=1-\frac{1}{G(x)+1}\\ F=\frac{x-x^k}{1-x^k}\end{gathered}$$

这样我们得到了容斥系数，我们希望求的便是：

$$\left[\frac{x^n}{n}\right]\frac{1}{1-\sum\limits_{i=1}\frac{f_i x^i}{i!}}$$

我们可以发现 $F(x)$ 的项数只有 $\frac{n}{k}$ 项，我们可以暴力的求逆 $\frac{n}{k}\cdot n$ 所以，对 k 求和，复杂度即为 $\mathcal{O}(n^2\log n)$。

⑤ Yet another abc string

题目大意：请求出具有 a 个 A，b 个 B，c 个 C ，且不存在子串 ABC，BCA，CAB 的串的个数。

容斥系数是什么呢？考虑容斥的含义：将不合法的拼成0，将合法的拼成c（某个与参数无关的常数）。

$$\begin{aligned}F(a,b,c)&=(a+b+c)-3abc+abc(a+b+c)+0\times abc(ab+bc+ca)-3(abc)^2+\cdots\\&=(a+b+c)-3(abc+(abc)^2+\cdots)(3-(a+b+c))\end{aligned}$$

我们不妨假设 $u=abc$：

$$\begin{gathered}F(a,b,c)=(a+b+c)-\frac{u}{1-u}(3-(a+b+c))\\ \frac{1}{1-F}=\frac{1}{1-(a+b+c)+\frac{u}{1-u}(3-(a+b+c))}\end{gathered}$$

设 $s=1-a-b-c$ ，那么原式：

$$\begin{aligned}\frac{1}{1-F}&=\frac{1}{s+\frac{u}{1-u}(2+s)}=\frac{1-u}{s+2u}\\&=\frac{1-u}{s}\cdot\frac{1}{1+2(u/s)}=\frac{1-u}{1-a-b-c}\sum_{k=0}(-2)^k\left(\frac{abc}{1-a-b-c}\right)^k\end{aligned}$$

---

question 1

假设我们已经枚举好 k,该如何求：

$$[a^u b^v c^w]\frac{1}{1-a-b-c}$$

重要性质：齐次。

$$\begin{aligned}\frac{1}{1-a-b-c} &= \sum_{k=0}(a+b+c)^k \\ [a^u b^v c^w]\frac{1}{1-a-b-c} &= [a^u b^v c^w](a+b+c)^{u+v+w}\end{aligned}$$

不难发现系数是：$\frac{(u+v+w)!}{u!v!w!}$。

question 2

我们该如何求：

$$[a^u b^v c^w]\frac{1}{(1-a-b-c)^m}$$

考虑一个经典的组合意义：将一个整数 n 划分成 m 个**非负**整数的方案数。

$$\begin{aligned}\frac{1}{(1-x)^m} &= \sum_{k=0}\binom{k+m-1}{m-1}x^k \\ [a^u b^v c^w]\frac{1}{(1-a-b-c)^m} &= \sum_{k=0}\binom{k+m-1}{m-1}(a+b+c)^k \\ &= [a^u b^v c^w]\binom{u+v+w+m-1}{m-1}(a+b+c)^{u+v+w}\end{aligned}$$

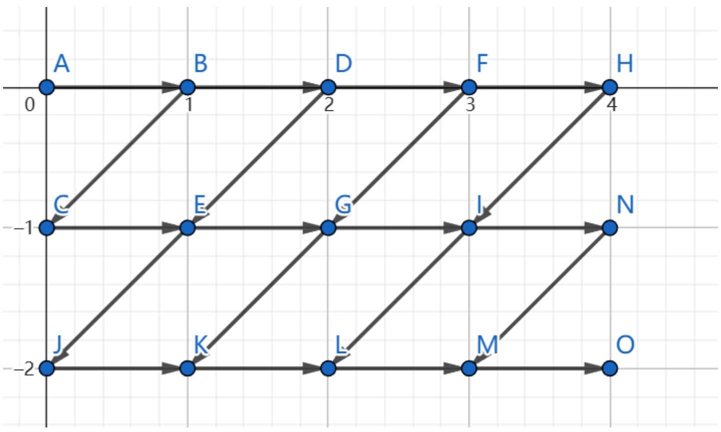
反射容斥

从一道简单的问题题引入。

⑥ [JLOI2015] 骗我呢

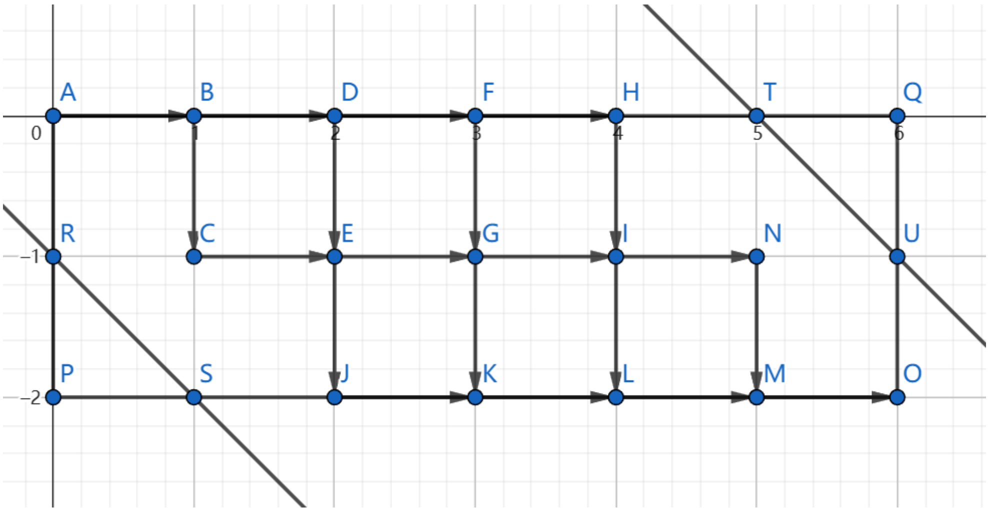
> ref : 吉林省选 2015

题目大意：给你这样一个网格，求 $(0,0)$ 到 (n,m) 的方案数。



$n,m\le 10^6$。

首先我们不难先把这个网格进行一下坐标变换



我们可以看成，在标准坐标系下（每次向 $(1,0)$ 或 $(0,1)$ 游走的网格上），从 $(0,0)$ 到 $(n,n+m)$ 不接触到两条直线的方案数。

如果是只有一条直线作为限制，我们可以类比卡特兰来容斥。对于两条直线，我们可以想到，如果分别对两条直线容斥会产生什么效果呢？我们会算重一些贡献，比如对于一条直线 l_1 翻折后，我们的确可以保证他之后不会与 l_1 交，但是可能在之前/之后与 l_2 交，这样的贡献我们便算重了。

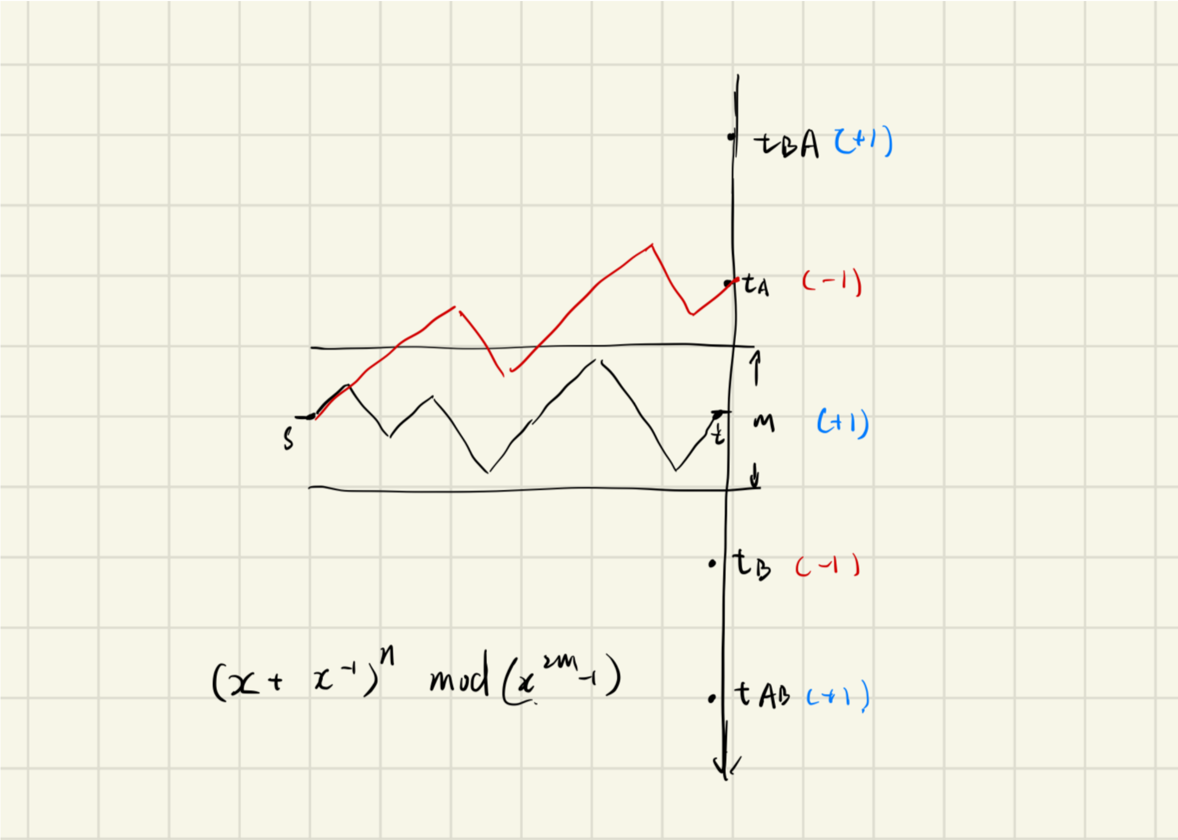
所以，我们考虑如下一个容斥，为了方便描述，我们设记号 A 表示与第一条直线交， B 表示与第二条直线交。

我们减去 A，B 的情况（钦定与一条直线交），加上 AB，BA 的情况（先钦定与一条直线交，在钦定与零一条直线交），减去 ABA,BAB 的情况....

正确性的话，我们考虑这个直线第一次碰到/从另外一边返回第一次碰到的线的次数。不难计算出容斥系数。

所以最后算出 $\frac{n}{m}$ 个组合数即可。

另一种特别的视角是，如果我们将坐标系再次变换后，可以看作 $+1/-1$ 不能碰到上下边界的问题。而基于这种标准的问题，我们可以看成形如 $(\frac{1}{x}+x)^n \bmod (x^{2m}-1)$ 的多项式问题。



⑦ Math Exam

> ref : CCPC2022 Guangzhou J

题目大意： 求满足 $4S_i = (a_i+1)^2$ ，$|a_i| \le n$ 的序列 a 的个数，其中 $S_i = \sum_{k=1}^{i} a_k$。

$n, m \le 10^7$。

$4S_n = (a_n+1)^2$，我们不难推出 $4S_n - 4S_{n-1} = (a_n+1)^2 - (a_{n-1}+1)^2 = 4a_n$ 状的方程。

也就是：$(a_n-1)^2 = (a_{n-1}+1)^2$，我们不妨只考虑 $|a_i|$ 不难发现：$|a_i| = |a_{i-1}| \pm 1$，且 $0 \le |a_i| \le M$。

经典的双线容斥。

容斥的内容当然没有结束，但我们先看一看另一个与容斥息息相关的话题。

2. 斯特林数

我们以一道例题作为引子引入斯特林数。

⑧ 无题

请问一个满足如下条件的 $[1, n]$ 的排列 p 的个数：

1. p 有 k 个前缀最大值，即 $\#\{i \mid p_i \ge p_j, \forall j < i\}$.

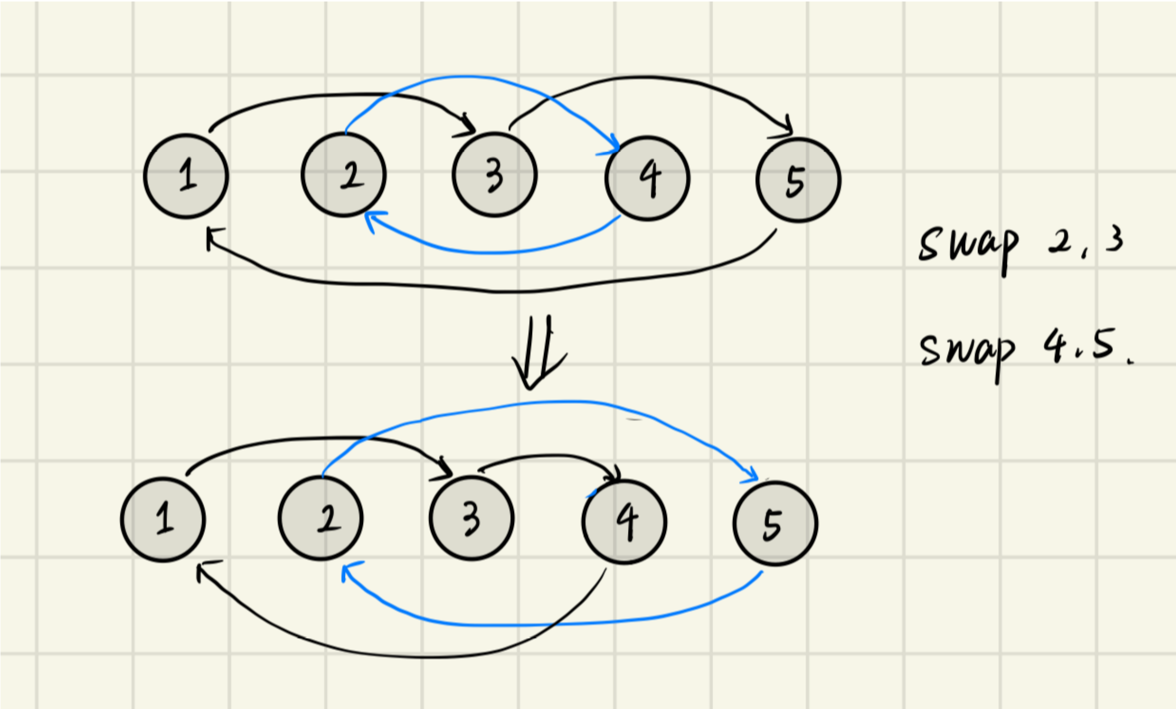
曾几何时，当笔者还是一个在役选手时，有一位远古选手告诉过笔者，序列的性质往往是比排列要好的，接下来要叙述的是一种刻画排列的方式。

排列

一般来讲，我们可以对于每一个 x 连出一条有向边 (x, p_x)，这样就可以把一个排列看成若干个有向环，在下文我们称其为“环排列”。

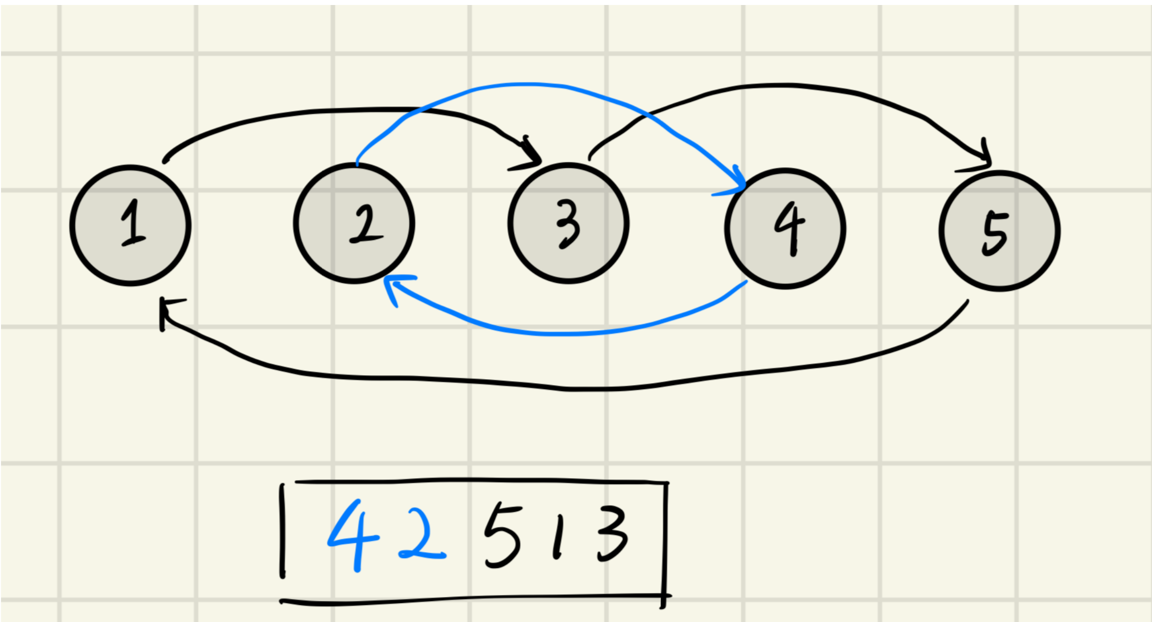
这个结构具有一些特殊的性质，比如说我们可以根据环的个数直接得到 **逆序对** 的奇偶性。

> proof：我们可以先证明“环与环”之间的贡献为0。
>
> 接下来我们尝试证明每个环贡献 $(-1)^{l-1}$ 的逆序对数。其中 l 是环的长度。



接下来我们介绍一种将环排列重新映射到序列上的方法。

我们将环按照**每个环中，最大元素**从**小**到大排序，然后将最大值放到开头，。



那么我们不难发现我们将一个排列重新映射成了另一个排列。而且我们可以根据前缀最大值来反推这个原先的置换环。

所以我们得到了一个双射。那么此时我们也能回答我们之前那个例子的答案了，他的答案就是有 k 个环的 n 阶环排列的个数，我们记作 $S(n,k)$ 因为他有一个更特殊的名字。（我们也会记作：$\begin{bmatrix} n \\ k \end{bmatrix}$）

第一类斯特林数。

我们不难得到一个 $S(n-1,\cdot)$ 向 $S(n,\cdot)$ 的递推。

$$S(n,k) = S(n-1,k-1) + nS(n-1,k)$$

第一个 $S(n-1,k-1)$ 表示的将编号是 n 结点连出一个自环。

$n \times S(n-1,k)$ 表示的是将编号是 n 的结点插入到一个新的环里。

熟悉EGF的朋友可能会立即写出 $S(n,k)$ 的生成函数：

$$\begin{aligned} S(n,k) &= \left[\frac{x^n}{n!}\right] \frac{1}{k!}(-\ln(1-x))^k \\ S(n,k) &= \left[\frac{x^n}{n!} z^k\right] \sum_{k=0} \frac{1}{k!}(-\ln(1-x)z)^k = \exp(-\ln(1-x)z) = \left(\frac{1}{1-x}\right)^z \end{aligned}$$

所以若我们固定住 n，应满足：

$$\begin{aligned} S(n,\cdot) = \left[\frac{x^n}{n!}\right]\left(\frac{1}{1-x}\right)^z &= \binom{n+z-1}{z-1} \\ \implies \sum_{k=0}^{n} S(n,k)z^k = z(z+1)\cdots(z+n-1) &= z^{\overline{n}} \end{aligned}$$

我们可以将第一类斯特林数看成，从上升幂转普通幂时，产生的常数。

我们接着来介绍以下第二类斯特林数。

组合定义：我们定义 $S_2(n,k) = \begin{Bmatrix} n \\ k \end{Bmatrix}$ 为，将 $[1,n]$ 划分到 k 个集合的方案数。（$\{1,2\},\{3\}$ 这个划分和 $\{3\},\{1,2\}$ 只被算进贡献一次）。

仿照第一类斯特林数的递推/EGF，我们不难写出第二类斯特林数的递推/EGF。

$$\begin{aligned} S_2(n,k) &= S_2(n-1,k-1) + kS_2(n-1,k) \\ S_2(n,k) &= \left[\frac{x^n}{n!}\right]\frac{1}{k!}(\exp(x)-1)^k \\ &= \left[\frac{x^n}{n!} z^k\right] \exp((\exp(x)-1)z) \end{aligned}$$

以及一个恒等式：

$$\begin{aligned} \sum_{k=0}^{n} S_2(n,k) z^{\underline{k}} &= \sum_{k=0}^{n} \left[\frac{x^n}{n!}\right] \frac{1}{k!}(\exp(x)-1)^k z^{\underline{k}} \\ &= \left[\frac{x^n}{n!}\right] \sum_{k=0}^{n} (\exp(x)-1)^k \binom{z}{k} \\ &= \left[\frac{x^n}{n!}\right] (\exp(x)-1+1)^{z} \\ &= z^n \end{aligned}$$

即，我们可以将第二类斯特林数看成，由普通幂转为下降幂时，产生的系数。

所以对于一些固定 k 求 x^k 状物，我们可以考虑第二类斯特林数.....吗？

当然是可以的，但我们可不可以有一个更好的做法呢？

exp †

$$n^k = \left[\frac{x^k}{k!}\right]\exp(nx)$$

实际上我们可以发现第二类斯特林数也蕴含着做 $\exp(x)-1$ 的变基变换。

自然数幂和：

$$\sum_{i=0}^{n} i^k = k![x^k]\sum_{i=0}^{n}\exp(ix) = k![x^k]\frac{1-\exp((n+1)x)}{1-\exp(x)}$$

可以发现 $\frac{x}{1-\exp(x)}$ 就是 Bernoulli 数的 EGF。

我们给出另一种常见的，使用斯特林数的做法：

$$\sum_{i=0}^{n} i^k = \sum_{i=0}^{n}\sum_{j=0}^{k}\left\{ {k \atop j} \right\} i^{\underline{j}} = \sum_{j=0}^{k}\left\{ {k \atop j} \right\}\sum_{i=0}^{n}\binom{i}{j}j! = \sum_{j=0}^{k}\left\{ {k \atop j} \right\}\binom{n+1}{j+1}j!$$

笔者个人认为，$\exp$ 是一种更为本质的做法，下面给出两道例题让大家体会一下 $\exp$ 的美妙之处。

⑨ 愤怒的小 N

> ref : [NOI Online 2021 提高组] T3

题目大意：给你一个无限长的串 `S=ab ba baab baababa...` 希望你算出：

$$\sum_{i=1}^{n}[s_i=b]f(i)$$

$$f(x) := \sum_{i=0}^{k} f_i x^i$$

其中 $\log n \le 5\times 10^5$，$k \le 500$。

根据上面所提到的，我们设集合 S 对应的生成函数是：

$$F_S(x) = \sum_{u\in S}\exp(ux)$$

那么： $\sum f_k k![x^k]F_S(x)$ 即为答案。、

对于字符串的长为 2^t 的前缀，我们分别维护出 $P_t(x)$，$Q_t(x)$ 表示 a,b 对应的集合的生成函数。

$$P_{t+1}(x) = P_t(x) + e^{2^t x}Q_t(x)$$
$$Q_{t+1}(x) = Q_t(x) + e^{2^t x}P_t(x)$$

$P_0(x)=1, Q_0(x)=1$。

直接卷积似乎太为暴力，我们仔细观察，做换元：$F_t(x)=P_t(x)-Q_t(x)$，$G_t(x)=P_t(x)+Q_t(x)$.

$$F_{t+1}(x) = F_t(x)\left(1-e^{2^t x}\right)$$
$$G_{t+1}(x) = F_t(x)\left(1+e^{2^t x}\right)$$

可以发现 F 每次乘上的是一个 $x+\cdots$，所以 F 只有前 k 位有用。

而 G 是 $\sum_{i=1} e^{ix}$ ，其实是自然数幂和，可以按照上述做法解决。（当然，如果你愿意插值也可以。）

这样总复杂度为 $\mathcal{O}(k^3+sk)$。

杂项

⑩ 山河重整 †

> `ref : [AHOI2022] t3`

题目大意：给定整数集合 $S=\{1,\cdots,n\}$，计算有多少个子集 $T\subseteq S$，使得 $1,2,\cdots,n$ 都可以被表示为 T 的一个子集中所有数的和。

$n\le 5\times 10^5$ 任意模。

什么时候存在一个最小的数 x 不能被表示出来？当且仅当，所有的 $<x$ 的数能全被表示。

存在一个集合 $A\subseteq T$ 使得， $\sum_{u\in A} u=x-1$。而 $T-A$ 最小元素 $>x$。

我们考虑对于任何一种选取集合的方案，都存在一个最小的不可被表达的数，所以枚举最小的不可被表达的数，记为 $k+1$。

根据上述性质，我们实际上不太关心前面那 k 个可以被表示的数的结构，假设能把 $1\sim k$ 表示出，且和为 k 的子集数为 a_s。

我们有如下**方程**：

$$\sum_k a_k x^k(1+x^{k+2})(1+x^{k+3})\cdots=(1+x)(1+x^2)\cdots(1+x^n)\cdots$$

而答案即为： $\sum_{i=0}^n a_i 2^{n-i-1}$。

注意到一个关键性质：

$$\prod_{i=1}^{\infty}(1+x^i)=\sum_{h=1}^{\infty}x^{h(h+1)/2}\prod_{i=1}^{h}\frac{1}{1-x^i}$$

有一个非常精彩的解释：我们考虑左式是拆分数对应的 `GF`，我们假设 $x_1<x_2<\cdots<x_h$

那么做一次线性变换后 $v_i=x_i-i$， $v_1\le v_2\le\cdots\le v_h$。 此时，我们考虑对差分数组计数，即为 $\prod_{i=1}^h\frac{1}{1-x^i}$。

我们考虑如何反解出 a_k？ 设 $b_k=a_{k-1}$

$$\begin{aligned}x\cdot\text{LHS}&=\sum_k b_k x^k\sum_i\frac{x^{ik}x^{i(i+1)/2}}{(1-x)\cdots(1-x^i)}\\&=\sum_i\frac{x^{i(i+1)/2}}{(1-x)\cdots(1-x^i)}B(x^{i+1})\end{aligned}$$

故：

$$B(x)=x\cdot\text{RHS}-\sum_{i\ge 1}\frac{x^{i(i+1)/2}}{(1-x)\cdots(1-x^i)}B(x^{i+1})$$

这样我们可以倍增计算 $B(x) \bmod x^n$ （$n \leftarrow 2n$） 。而和式也只需要计算 $\sqrt{n}$ 项即可。倍增时一次复杂度为：$n\sqrt{n}$

复杂度为 $\sum_{i=1}^{\log n} \frac{n}{2^i}\sqrt{\frac{n}{2^i}} = \mathcal{O}(n\sqrt{n})$。

⑪ 排列游戏

> ref : thupc 2024 决赛。

题目大意：给你个排列，每次交换 (i,j) 的代价是 $|i-j|$，你希望用 $\le m$ 的代价将排列还原成 $[1,n]$。问可行的排列数。

$n \le 500, m \le 5000$。

我们首先观察出一个关键性质，我们每次只会在一个环排列上交换。

而一个环排列的贡献是：$\frac{1}{2}\sum_{(a,p_a)}|p_a-a|$。

我们每次找到要给三元组 (a,b,c) 使得 (a,b,c) 的相对排名是 $(2,1,3)$，或者 $(1,3,2)$ 然后交换 (a,b) 即可发现上述的“势能”会随着降低代价那么多。不难归纳。

所以现在问题转化为 $\sum|i-p_i| \le 2m$。我们从小往大插入结点 i ，我们类似斯特林数那样考虑每次是“新建一个环”，还是在头/尾插入，抑或是合并两个环。我们统计目前为止还有多少个“空”的环头/尾，他们的贡献每次会+1。

$f_{i,j,k}$ 表示当前插入的是第 i 个结点，空出 j 个头/尾，和是 k。一个发现是 j 不会超过 $O(\sqrt{n})$ 因为一次最多消失两个“空”的位置，而一次贡献 j。所以其实 j 会延后贡献至少 $O(j^2)$ 。所以总复杂度为 $O(nm\sqrt{m})$。
