Problem B. 全概率公式

全概率公式 $P(A)=\sum_{i=1}^{n}P(F_i)\cdot P(A|F_i)$

Problem C. LCT

由于读入占比时间较大，为了防止把大常数 1 log 选手卡了，本题没有卡掉所有的 2 log 做法，所以有不少常数比较小的 2 log 是可以通过的。

比如一个简单的 2 log 做法是，使用树状数组和 DFS 序维护每个点到根节点的路径上有多少边没有被断开，配合树上倍增即可做到一个简单的 2 log 算法。直接使用树链剖分算法配合 set 也可以得到一个简单的 2 log 做法。当然直接使用线段树分治， 实现不太差大概也是可以通过的。

以下是两个 1 log 做法。对于会树链剖分的选手，可以直接使用树链剖分，每条重链开一个 set 维护这条链上被断开的边，并在全局使用一个数组维护重链上深度最小的断开的边的深度。这样，对于答案不在这条重链上的重链，使用数组存的信息就可以 $O(1)$ 判断。对于最后答案在这条重链上的重链，查询一次即可。

另外一个 1 log 做法不需要使用树链剖分。可以把问题转化为查询某个点上方第一个断开的边。我们考虑把每条边断开的时间区间挂在点上，然后对这棵树进行 DFS，同时维护一棵可持久化线段树。线段树的位置 i 表示在时间 i，当前点上方第一个被修改的边的编号是多少，只需要支持区间赋值或者是区间取 max 即可。

Problem D. 股票交易

令 $b_i=a_i-a_{i-1}$，则 $\frac{a_y-a_x}{y-x}=\frac{1}{y-x}\sum_{i=x+1}^{y}b_i$，即是求连续一段 b_i 的平均值。显然，$\exists i\in[x,y]$，$b_x\geq\frac{1}{y-x}\sum_{i=x+1}^{y}b_i$（总有一个数大于等于平均值）。可见，选一个 b_i 就能选到最优结果。注意判断不交易的情况即可。

Problem E. HoMaCoMoHa!

手玩一下样例，顺便对 $k=4$，$k=5$ 的情况打个表，不难发现规律：

$$ans=\max(0,\lfloor\frac{n-\frac{k\times(k-1)}{2}+k-1}{k}\rfloor)$$

~~输出即可。~~

证明：

因为决斗的发生是独立的，所以我们只需要研究最终状态。

记 $dp_{i,j}$ 表示：使得最终状态的 $i(2\leq i\leq k)$ 级HoMaCoMoHa的数量大于等于 j 时，最少需要 1 级HoMaCoMoHa的数量。

为方便表述，将被淘汰的HoMaCoMoHa记为 0 级。

考察如下两个命题：

— **命题** $A(p)$：当 $p(2\leq p\leq k)$ 级HoMaCoMoHa的数量大于等于 1 时，设初始状态下至少需要 n_p 个 1 级HoMaCoMoHa。若初始状态HoMaCoMoHa的数量为 n_p，那么在最终状态下，我们有：

$$a_i=\begin{cases}1,i\neq p-1\\0,i=p-1\end{cases}$$

其中 $a_i(1\leq i\leq k)$ 表示在指定状态下第 i 级HoMaCoMoHa的数量。

— **命题** $B(p)$：当 $p(2\leq p\leq k)$ 级HoMaCoMoHa的数量大于等于 2 时，设初始状态下至少需要 m_p 个 1 级HoMaCoMoHa。若初始状态HoMaCoMoHa的数量为 m_p，那么在最终状态下，我们有：

$$a_i = \begin{cases} 1, i \le p-1 \\ 0, i = p-1 \\ 2, i = p \end{cases}$$

其中 $a_i (1 \le i \le k)$ 表示在指定状态下第 i 级HoMaCoMoHa的数量。

下面证明这两个命题:

当 $p = 2$ 时:

命题 $A(2)$: 易知 $n_2 = 2$，模拟得 $a_1 = 0, a_2 = 1$。所以 $A(2)$ 成立。

命题 $B(2)$: 易知 $m_2 = 4$，模拟得 $a_1 = 0, a_2 = 2$。所以 $B(2)$ 成立。

当 $p = 3$ 时:

命题 $A(3)$: 易知 $n_3 = 4$，模拟得 $a_1 = 1, a_2 = 0, a_3 = 1$。所以 $A(3)$ 成立。

命题 $B(3)$: 易知 $m_3 = 7$，模拟得 $a_1 = 1, a_2 = 0, a_3 = 2$。所以 $B(3)$ 成立。

一方面，假设 $p = k_0$ 时，$A(p)$ 成立。

设状态 $S_{t,k_0,v} (k_0 \ge 2, 0 \le t \le k_0, v \ge 1)$ 表示:

$$a_i = \begin{cases} 1, i \ne t \text{ and } i \le k_0 \\ 0, i = t \\ v, i = k_0 \end{cases}$$

这一序列。其中 $a_i (1 \le i \le k_0)$ 表示在指定状态下第 i 级HoMaCoMoHa的数量(k_0 级HoMaCoMoHa之间不会产生决斗)。特别地，因为 a_0 的实际值并不会影响到结果，所以在下面的证明中，不会特别注意判断 a_0 这一边界情况。

显然决斗发生的先后顺序对最终状态没有影响。所以初始状态的 1 级HoMaCoMoHa的数量增加 1 等价于在最终状态下的 1 级HoMaCoMoHa的数量增加 1。先考虑 $v = 1$ 的情况。

考虑在状态 $S_{t,k_0,1}$ 的最终状态下，令 $a_1 \leftarrow a_1 + 1$。

如果 $t \ge 2$，那么 $a_1 = 1$。此时有:

$$a_i = \begin{cases} 1, 2 \le i \le k_0 \text{ and } i \ne t \\ 0, i = t \\ 2, i = 1 \end{cases}$$

由于 $a_0 = 0, a_1 = 2, a_2 = 1$，当 2 个 1 级HoMaCoMoHa之间发生决斗，有:

$$\begin{aligned} a_0 &\leftarrow a_0 + 1 \\ a_1 &\leftarrow a_1 - 2 \\ a_2 &\leftarrow a_2 + 1 \end{aligned}$$

此时 $a_1 = 0, a_2 = 2, a_3 = 1$。依次类推，直到:

$$a_i = \begin{cases} 1, i \le t-3 \text{ or } i \ge t+1 \\ 0, i = t-2 \text{ or } i = t \\ 2, i = t-1 \end{cases}$$

进一步令:

$$\begin{aligned} a_{t-2} &\leftarrow a_{t-2} + 1 \\ a_{t-1} &\leftarrow a_{t-1} - 2 \\ a_t &\leftarrow a_t + 1 \end{aligned}$$

即有：

$$a_i = \begin{cases} 1, i \neq t-1 \\ 0, i = t-1 \end{cases}$$

即为 $S_{t-1,k_0,1}$。

如果 $t = 1$，那么此时有：

$$a_i = 1, 1 \leq i \leq k_0$$

此情况即为 $S_{0,k_0,1}$。

如果 $t = 0$，此时有：

$$a_i = \begin{cases} 1, i \neq 1 \\ 2, i = 1 \end{cases}$$

由于 $a_0 = 0, a_1 = 2, a_2 = 1$，当 2 个 1 级HoMaCoMoHa之间发生决斗，有：

$$\begin{aligned} a_0 &\leftarrow a_0 + 1 \\ a_1 &\leftarrow a_1 - 2 \\ a_2 &\leftarrow a_2 + 1 \end{aligned}$$

此时 $a_1 = 0, a_2 = 2, a_3 = 1$。依次类推，直到：

$$a_i = \begin{cases} 1, i \leq k_0 - 3 \text{ or } i = k_0 \\ 0, i = k_0 - 2 \\ 2, i = k_0 - 1 \end{cases}$$

令：

$$\begin{aligned} a_{k_0-2} &\leftarrow a_{k_0-2} + 1 \\ a_{k_0-1} &\leftarrow a_{k_0-1} - 2 \\ a_{k_0} &\leftarrow a_{k_0} + 1 \end{aligned}$$

即有：

$$a_i = \begin{cases} 1, i \leq k_0 - 2 \\ 0, i = k_0 - 1 \\ 2, i = k_0 \end{cases}$$

即为命题 $B(k_0)$ 对应的状态。

对于命题 $A(k_0)$，其最终状态可以表示成 $S_{k_0-1,k_0,1}$。根据上述证明，我们将 $S_{k_0-1,k_0,1}$ 转化到 $S_{0,k_0,1}$ 的代价（即需要添加 1 级HoMaCoMoHa的数量)为 $k_0 - 1$。另外的，将 $S_{0,k_0,1}$ 转移到 $B(p)$ 中所表示的最终状态（即 $S_{k_0-1,k_0,2}$）需要的代价为 1。这说明，可以通过增加 k_0 个 1 级HoMaCoMoHa使得命题 $A(k_0)$ 表示的最终状态 $S_{k_0-1,k_0,1}$ 转移到命题 $B(k_0)$ 表示的最终状态 $S_{k_0-1,k_0,2}$。这证明了命题 $B(p)$ 在 $p = k_0$ 时成立，且 $dp_{k_0,2} = dp_{k_0,1} + k_0$。

另一方面，假设 $p = k_0$ 时，$B(p)$ 成立。

由于 $B(k_0)$ 表示的最终状态 $S_{k_0-1,k_0,2}$ 经过操作：

$$\begin{aligned} a_{k_0-1} &\leftarrow a_{k_0-1} + 1 \\ a_{k_0} &\leftarrow a_{k_0} - 2 \\ a_{k_0+1} &\leftarrow a_{k_0+1} + 1 \end{aligned}$$

立即变为：

$$a_i = \begin{cases} 1, i \le k_0 - 1 \\ 0, i = k_0 \\ 2, i = k_0 + 1 \end{cases}$$

即为命题 $A(k_0+1)$ 对应的状态。于是命题 $A(p)$ 在 $p = k_0 + 1$ 时成立，且 $dp_{k_0+1,1} = dp_{k_0,2}$。

结合 $p = 2$ 与 $p = 3$ 时 $A(p)$ 与 $B(p)$ 均成立。由螺旋数学归纳法，对所有 $p \ge 2$，$A(p)$ 与 $B(p)$ 均成立。

进而对于 $k_0 \ge 2$，均有 $dp_{k_0+1,1} = dp_{k_0,2} = dp_{k_0,1} + k_0$，且 $dp_{2,1} = 2$。

我们已知从 $S_{k_0-1,k_0,1}$ 转移到 $S_{k_0-1,k_0,2}$ 需要增加 k_0 个 1 级HoMaCoMoHa。对于最终状态 $S_{k_0-1,k_0,v}(v \ge 1)$ 与 $S_{k_0-1,k_0,1}$，它们除了 a_{k_0} 均对应位相等。所以 $S_{k_0-1,k_0,v}$ 转移到 $S_{k_0-1,k_0,v+1}$ 需要增加 k_0 个 1 级HoMaCoMoHa，即 $dp_{k_0,v} = dp_{k_0,v-1} + k_0 (v \ge 2)$。

容易推出：$dp_{i,1} = 1 + \frac{i\times(i-1)}{2}$。进而 $dp_{i,j} = dp_{i,1} + (j-1) \times i = 1 + \frac{i\times(i-1)}{2} + (j-1) \times i$。

给定 n, k，设 ans 为最终 k 级HoMaCoMoHa的数量，有：

$$n \ge dp_{k,ans} = 1 + \frac{k\times(k-1)}{2} + (ans-1) \times k$$

又因为 $ans \ge 0$，所以：

$$ans = \max(0, \lfloor \frac{n - \frac{k\times(k-1)}{2} + k - 1}{k} \rfloor)$$

Problem F.

考察：假如第 i 块石块不是它以及之前的石块中最小的，那么当把这块石块放上塔顶的时候，高塔一定会倒塌，这个前缀会被随机打乱。之后，会从头开始再一次搭起来一个由前 $i-1$ 块石块按顺序排列而成的高塔，然后如果第 i 块石块依然不是它以及之前的石块中最小的，则会再进行一次这样的过程，直到第 i 块石块是前缀最小值为止。

可以发现，其实整个过程是经历了这样的一些子过程：把从左往右第一块不是前缀最小值的石块以及它之前的所有石块随机打乱，然后把这些石块进行一直的搭建过程，直到某一次把前 i 块石块按顺序搭成高塔。设 $f(i)$ 表示把前 i 块石块随机打乱顺序后，期望需要多少天可以把它们按顺序搭成高塔，那么答案是 n 加上所有不是前缀最小值的位置 i 的 $f(i)$ 之和。

现在计算 $f(i)$，考虑到需要先把前 $i-1$ 块石块按顺序搭好，然后有 $\frac{1}{i}$ 的概率不需要重新搭一遍，可以得到 $f(i) = f(i-1) + 1 + \frac{i-1}{i} f(i)$，用这个式子递推即可求出所有 $f(i)$。可以使用线段树维护是前缀最小值的位置的 f 之和，复杂度 $O(n + q \log^2 n)$。

Problem G. Mobiusp的希望树

复杂度： $O(\sqrt{n} log_n)$

思路： $sqrt$ 结论，最后一层结点数近似为总结点数的根号，使用尽量少的结点让倒数第二层的容量能容纳剩余的所有结点。

枚举除最后一层所需的结点数 x，计算 x 所能容纳的最后一层结点的最大编号 y。

二分查找最小的能容纳 x 的除最后一层的结点数 i ， y 等于 $\sum_{j=i}^{x} j$，当 y 大于等于 n 时结束枚举，输出 x。

Problem H. 父子局

注意到 $\frac{1}{n} = \frac{n+1}{n\times(n+1)} = \frac{n}{n\times(n+1)} + \frac{1}{n\times(n+1)} = \frac{1}{n+1} + \frac{1}{n\times(n+1)}$。

当 $n=1$ 时，假设存在 x,y 使得 $\frac{1}{n}=\frac{1}{x}+\frac{1}{y}$，那么有 $x\times y=x+y$ 即 $(x-1)\times(y-1)=1$，进而 $x=y=2$，与 $x\neq y$ 矛盾。对于 $n\geq 2$，可以令 $x=n+1, y=n\times(n+1)$，此时 $x\neq y$。

Problem I. 轮符雨

根据题意化简可得 $SoyoØ <= \sum_{i=1}^{n-1} a_i - a_{i+1} = a_1 - a_n$，于是本题目标就变为交换至多一次使得 a_1-a_n 最大，于是可以将整个数组分为 a_1、a_n与其他元素，交换的策略有五种:

1. 选择 a_1 与其他元素中的一个进行交换
2. 选择 a_n 与其他元素中的一个进行交换
3. 选择 a_1 与 a_n 进行交换
4. 选择其他元素中的两个进行交换
5. 不进行交换

策略 1 中最优策略是交换 a_1 与 $\max_{1<j<n} a_j$ ，策略 2 中最优策略是交换 a_n 与 $\min_{1<j<n} a_j$ ，策略 3 与策略 5 都只有一种情况，策略 4 对答案没有贡献，这些情况中的最大值即为答案。

Problem J. 炫耀快乐

先统计数对 x,y 在 S 中以 $S_i=x, S_{i+1}=y$ 形式出现的次数，记为 $cnt_{x,y}$。

观察到 m 的范围很小，考虑状压 dp。在题目给出的两个贡献式子中，T_x 和 T_y 的贡献可以分为两部分。比如当 $T_x\leq T_y$，可以在将 x 加入集合时贡献 $-k_1\times T_x$，在 y 加入集合时贡献 $k_1\times T_y$。同理，当 $T_x>T_y$，在将 x 加入集合时贡献 $k_2\times T_x$，在 y 加入集合时贡献 $k_2\times T_y$。由于 x,y 对出现了 $cnt_{x,y}$ 次，故以上贡献均需要乘上倍数 $cnt_{x,y}$。

若排列的前 i 个数已经确定为集合 S，现在需选择第 $i+1$ 个数，假设为 p。如果 $q\in S$，说明 $T_q<T_p$，否则说明 $T_q>T_p$。枚举 S,p,q 即可，时间复杂度 $O(m^2 2^m)$，空间复杂度 $O(2^m)$。

本题有复杂度 $O(m2^m)$ 的做法，可以由以上做法优化得到，由于新生赛整体难度饱受诟病，本题就到此为止了。

Problem K. agKc 与太阳甩在身后

先消除一条平行于短边的结晶，然后沿着消除出的路径消除掉其他结晶即可。

注意地形为一条线或者 1×1 的情况。

读者还可自行思考一下，当题目改为"消除桩只能消除前方结晶"的情况下该如何解决。

Problem L. 新生赛与邪恶计划

首先我们考虑如果老生不会放弃计划，最终 n 号坐在自己座位上的概率是多少。

如果有老生被分配到 n 号上，显然最终概率为0。下面我们默认没有老生被分配到 n 号。

我们将每次"吉列的豆蒸"等效为：每位新生入座时会把自己座位上的老生赶到另外一个随机空座位上。转化后，我们再考虑 n 号入座前最终局面。

首先讨论哪些人的座位是确定的：所有除了 n 号以外的新生一定坐在自己的座位上，且所有没有被新生赶走的老生一定坐在他们事先被分配好的座位上。那么相对的，所有被分配在新生座位的一定会被赶走的老生，在最终局面里会是"居无定所"状态，这些老生会在剩下的位置里随机分布。

而且我们知道，n 号入座前一定只剩一个座位。所以假设被分配在新生的座位的老生一共有 k 位，那么在最终局面时他们只会在 $k+1$ 个座位里随机分布，我们所求的剩下 n 号座位的概率就是$\frac{1}{k+1}$。

为了方便，我们将被分配到 s_i 号座位的 i 号老生看作一条连接 i 与 s_i 的边，整个计划形成一个点数为 n ，边数为 m ，且只有链和环的图。假设图中有 k 条边数大于等于 1 的最长链，那么这个图对应的概率为 $\frac{1}{k+1}$。

回到可以放弃计划的本题，答案就是原图所有生成子图的概率之和除以 2^m ，m 为原图的边数也就是初始老生的个数。我们考虑对原图的每一个环状或链状的最大连通块计算它的生成子图的贡献，然后通过背包整合。

对于点数为 n 的一条链，拆成恰好 m 条最长链的方案数为$\binom{n}{2m}$（二项式系数或者说组合数）：从 n 个点中选取 $2m$ 个点当作链的端点，那么每一种选法都唯一对应一种有 m 条最长链的生成子图。环类似，从 n 个点的环中选取 $2m$ 个端点，可以形成两种不同的 m 条最长链的情况，于是方案数为$2\binom{n}{2m}$。

接下来用分治 fft 优化背包合并即可。注意老生被分配到 n 号座位的情况。

彩蛋：本题生成函数的形式与2023华科新生赛E题(洛谷P9773)几乎完全一致，做了历年真题的有福了（？）。不会分治 fft 可以把那题代码复制下来稍微改一改。

Problem M. 战争游戏 II

首先，设树的直径长度为 d ，对于 $d \leq 2r_1 + 1$ 或 $r1 \geq r2$ 的情况，进攻方一轮就能把防守方炸死，因此无论怎么样，进攻方都能获胜；对于 $d > 2r_1 + 1$ 且 $2r_1 < r_2$ 的情况，显然，进攻方怎么做，都无法把防守方炸死（证明见 hdu 07 D 战争游戏题解）。

对于不在上面的情况的剩下情况，我们先考虑只有一个询问怎么做，我们考虑一个朴素的递推：

我们枚举树上的每个节点作为轰炸地点，然后对于剩下的每个节点，计算防守方若在这个节点，是否会被这次轰炸炸死。这样，我们就能计算出所有的可以被一次轰炸炸死的点。然后我们把这些可以被一次轰炸炸死的点删去，再枚举树上剩下的每个节点作为轰炸地点，然后计算树上剩下每个节点是否会被这次轰炸炸死，我们就能计算出所有的可以被两次轰炸炸死的点。然后我们再把这些可以被两次轰炸炸死的点删去，再重复以上操作……直到树被删空。这样，我们就能求出每个节点至少要几轮才能被炸死。

在以上的递推过程中，我们能得出两条结论：

1、在删点的过程中，剩下的点一定能组成一棵树；

2、在最后一轮删点时，剩下的点所能组成的树的直径的长度一定不超过 $2r_1 + 1$ ，且它的逆命题也成立；

3、一棵树被删空需要的轮数，只与其直径的长度有关。

以上三条结论的证明是容易的，在此处不再赘述。

然后，我们来考虑如何用较优的复杂度来回答多个询问：

在以下的讨论中，均默认以树的直径的中点为根。

首先，对于任何作为防守方起始位置的节点，若他的 $r_2 - r_1 - 1$ 级祖先存在，那么第一轮轰炸它的 $r_2 - r_1 - 1$ 级祖先是最优的。

考虑上面的朴素的递推过程，若一个节点他的 $r_2 - r_1 - 1$ 级祖先存在，那么若以他的 $r_2 - r_1 - 1$ 级祖先作为轰炸中心，是最有可能把这个节点炸死的（经过简单的分类讨论可以证明）。因此，我们把递推过程对应到正常的轰炸过程（把递推过程反过来），就可以证明第一轮轰炸他的 $r_2 - r_1 - 1$ 级祖先是最优的。

然后，确定了第一轮轰炸的节点后，剩下所需轰炸的轮数，只与以第一轮轰炸的节点所在的子树的深度（子树深度=子树内最深的节点的深度-子树根节点深度+1）有关，这个可以通过按子树内节点深度由深到浅递推证明。

最后，若作为防守方起始位置的节点的 $r_2 - r_1 - 1$ 级祖先不存在，那么第一轮轰炸直径中点一定是最优的（在递推过程中，这对应着这个节点在最后一轮才被删掉，这是容易证明的），特判一下就可以了。

实现的时候要实现求树上 k 级祖先，视 n , q 同阶，根据实现的方式，可以做到 $O(n)$ 或 $O(nlogn)$ 的复杂度。

Problem N. 虚拟换乘

不妨设 $f_{i,j,k,l}$ 为上一次乘坐的交通工具为 l , 终点为 k，这一次乘坐的交通工具为 j , 终点为 i 时的最小开支。则状态转移方程为

$$f_{i,j,k,l} = min(f_{i,j,k,l}, f_{k,l,k_2,l_2} + w_j + (dis_{i,k} \times cost_j))f_{i,j,k,l} = min(f_{i,j,k,l}, f_{k,l,k_2,l_2} + (dis_{i,k} \times cost_j))$$

$$j == l_2 \ and \ l \neq l_2 \ and \ (dis_{i,k} * cost_j) < T \ and \ k_2 \neq 1$$

其中, 上上一次乘坐的交通工具为 l_2， 终点为 k_2。初始状态 $f_{1,i,1,j} = 0 \quad (0 \leq i, j < K)$, 其它均置为 inf