2023 Hubei Provincial Collegiate Programming Contest

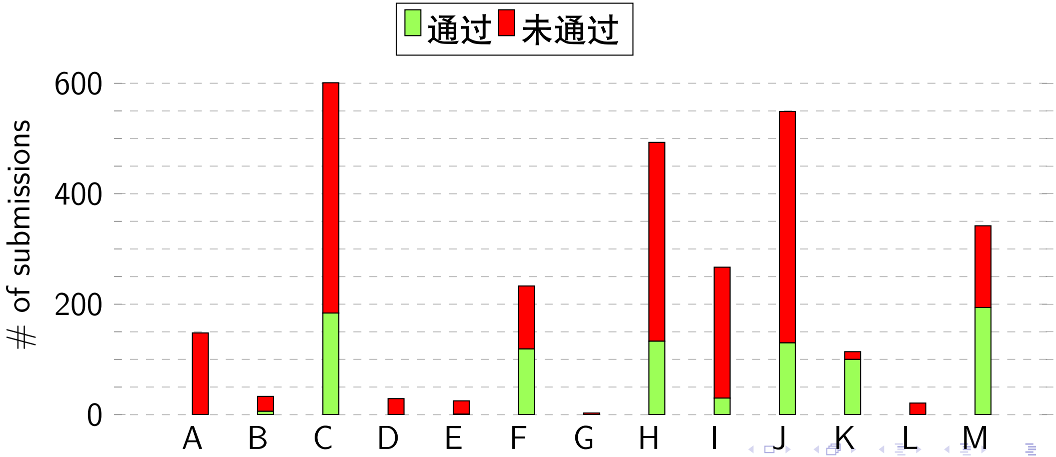
WHU/HUST ICPC 集训队

Apr. 30th, 2023

比赛小结

- 本次比赛共收到 2839 份提交代码。
- 其中 910 份代码正确。
- 196 名参赛选手有提交记录。
- 194 名参赛选手至少通过一题。
- 特别的，ChatGPT 也参与了这次比赛

各题通过情况



A. Prime Magic

题意

给定一长度为 n 的序列 $\{a_n\}$，每次可以选取一段长度为奇质数的连续子序列 $[l, r]$ 整体加一或减一，要求操作过程中序列中不得出现负数，问将整个序列转化为非严格递增序列的最少操作次数。

A. Prime Magic

题解

先将整个数组差分，记为 $\{b_i\}$，每次操作转化为对 $\{b\}$ 中两个数字分别 $+1$ 与 -1，最终使得整个数组非负，注意存在 $b_{n+1} = +\infty$。

将 b_i 按照正负分为两类，$b_i = 0$ 不影响最终结果，对于 $b_i > 0$，可以为某个 $b_j < 0$ 提供一次操作，至多 b_i 次，而每个 $b_j < 0$ 则需要 b_j 次操作使其非负。

建立一个二分图，左边为 $b_i > 0$，右边为 $b_j < 0$。注意 $n \geqslant 10$，故对于下标差为偶数的两个位置，可以通过 2 次操作使左边点向右边点进行一次传导，对于下标差为奇合数或 1 的两个位置，可以通过 3 次操作使左边点向右边点进行一次传导（哥德巴赫猜想），故可以费用流。

A. Prime Magic

题解

由于费用只有 $1,2,3$，考虑对费用流进行优化。有以下三种方法：

1. 首先只考虑费用为 1 的边进行匹配，注意由于奇偶性，如果只连费用为 1 的边，上述二分图实际上是由两个独立的二分图组成的，为左奇右偶以及左偶右奇，将费用为 1 的边匹配完成后，不难发现费用为 2 的边仅会对左右奇偶性相同的点匹配，故可以直接完成，剩下无法匹配的边则只能按照费为 3 的边匹配。

由于费用为 $1,3$ 的边只会对奇偶性不同的点连接，而费用为 2 的边只会对奇偶性相同的点连接，且 $1+3=2+2$，故上述先对费用为 1 的边优先匹配的贪心可以得到最优解。

A. Prime Magic

题解

2. 先将所有点都匹配至 $n+1$，此时费用为 1 的匹配已经最优，不需要进一步考虑，而费用为 2 和费用为 3 的匹配奇偶性不同，故相互独立，考虑将费用为 2 或 3 的匹配进行优化。不难发现 3→1 与 $2\to 1, 3\to 2$ 互相独立且 $2\to 1, 3\to 2$ 减少的操作数量相同，可以先匹配完 3→1 的优化，然后对 2→1, 3→2 的优化同时匹配，最后也可以得到最优解。采用 dinic 算法的复杂度均为 $O\left(\frac{n^{2.5}}{\ln n}\right)$。

3. 优化建图且费用流效率较高可能也可以通过本题。

B. Mode

题意

求 $\sum_{i=l}^{r} f(i)$，其中 $f(x)$ 表示 x 在十进制表示下数字串众数出现次数。

B. Mode

题解

常规的状态表示为 $S=[a_0,a_1,\ldots a_9]$ 表示有 a_i 个 i 。但这样的状态数较多，考虑到我们只需要知道众数的个数，而不关心众数是什么，因此我们变更一下状态。令 $S'=[b_0,b_1,\ldots b_{18}]$ 表示出现 i 次的数有 b_i 种，那么众数次数就是最大的非零 b_i。

其中状态 S' 需要满足一些限制：

1. $\sum_{i=0}^{18} b_i = 10$ ，表示一共只有 10 种数，即 0 至 9。
2. $\sum_{i=0}^{18} i \times b_i = len \leqslant 18$ ，表示位数小于等于 18。

B. Mode

题解

我们单独处理 $r = 10^{18}$ 这个 19 位的数，其余数都是小于 18 位的。
根据以上限制我们可以搜索出符合条件的状态，发现状态数只有 $|S'| = 1477$ 。
于是我们可以设计 $f_{i,j}$ 表示剩余 i 位当前状态为 S'_j 的答案。dp 的总状态数只有 $18 \times 1477 \approx 3 \times 10^4$ 。在数位 dp 时维护当前状态 S'，利用 map 查询状态 S' 对应的编号即可完成转移。
利用记忆化，我们可以做到单组复杂度 $O(l \times 10 \times \log|S'| \times 18)$，其中 l 是数字长度，$|S'|$ 是状态数量。

C. Darkness I

题意

在一个 $n \times m$ 的白色二维网格图中，有若干个黑色格点。
每秒所有与至少二个黑色点四相邻的白点变黑。
求最少多少个黑点能最终让整个棋盘变黑。

C. Darkness I

解法

方法很多，以下给出一种做法：
答案为 $\min(n, m) + \lceil \frac{abs(n-m)}{2} \rceil$，先铺一个对角线，然后隔一列放一个，最后剩一列时要专门放。
证明：注意到任何一次变色后，黑格的总周长不会变大，那么要充满 $n \times m$ 格子，至少需要 $\lceil \frac{2(n+m)}{4} \rceil = \lceil \frac{n+m}{2} \rceil$ 个初始黑格，可以发现上面的构造可以达到这个下界。

D. Darkness II

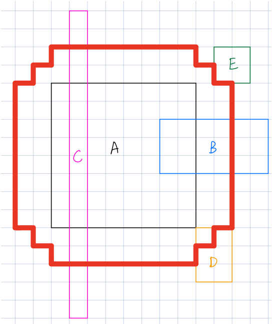
题意

在一个无限大的白色二维网格图中，给你 n 个黑色格点。
每秒所有与至少二个黑色点四相邻的白点变黑。
求最终黑点的总个数。

D. Darkness II

题解

本题允许常数不太差的 $O(n\log^2 n), O(n\sqrt{n})$ 做法通过，但存在 $O(n\log n)$ 做法。
观察可以发现，时间无所谓，本质上是做矩形的不断合并。
假设 A 需要找到其他矩形合并，那么如果存在矩形与红色框住的部分有交，则 A 可与其合并为一个新矩形，例如上图中的 B, C, D。
方便起见，后续仅考虑矩形查询。



D. Darkness II

题解

考虑到有些关键矩形可能后期才生成，我们首先找到一个" 若两个矩形有交，那么它们相互查询都可以成功" 的策略。
显然，暴力插入矩形是满足条件的。
具体做法可以使用线段树套线段树，在外层永久化，把经过的点全处理一遍，内层的点维护矩形标号集合和这个子树里面集合大小和。查询的时候，内层的点就看看子树里面有没有东西，有才进入子节点处理。简单分析发现，复杂度是 $\log^2 n$ 的。
也可以维护一些奇形怪状的类似 kdt 结构或四叉结构的东西，可能可以通过。

题解

接着理性分析，插入矩形的四条边是足够的，问题为动态的插入/删除线段，矩形查询。
考虑建两个树套树，控制线段在外层插入。具体的，外层的区间插入做标记永久化，内层用 set 之类的插入单点即可。
考虑到对边做操作实现起来仍然较为复杂，能否把查询点放的更小？
实际上每个矩形仅保留四个角被其余矩形查询，每个矩形每次寻找其余矩形合并时都做到此时无法再合并即可。考虑最初从格子大小 1*1 开始合成，那么不存在一种情况，使得两个所有角互不在对方查询范围内的矩形 A 和 B 被独立合成出来，也不存在 A 在 B 中 B 不在 A 中且 B 先于 A 被独立合成出的情况。
问题为动态的插入/删除单点，矩形查询。
代码可以很简单了，比如 kdt，线段树套 set...

D. Darkness II

题解

接下来介绍单 log 做法。
用 (xl_i, xr_i, yl_i, yr_i) 表示矩形。
使用线段树维护 x 维，每个节点维护两个下标集合：在 x 维与此点表示区间 $[l_x, r_x]$ 重合的矩形（即 $xl_i = l_x, xr_i = r_x$）标号集合 $id1_x$；在 x 维被此点表示区间完全包含的矩形（即 $l_x \leqslant xl_i \leqslant xr_i \leqslant r_x$）标号集合 $id2_x$。
对 (a, b, c, d) 查询时，若 $(a, b) = (l_x, r_x)$ 查 $id2_x$，若 $l_x \leqslant a \leqslant b \leqslant r_x$，查 $id1_x$，这完全消去了 x 维的影响，可以发现这样查询的点对始终有交（在 x 上）且不漏。
接着考虑另一维，矩形按在 y 上的左端点顺序合并并加入线段树。
这样保证了线段树每个节点上的 y 维形如 $yl_1 \leqslant yr_1 < yl_2 \leqslant yr_2 < yl_3 \dots$ 即若干不交区间，那么新查询的矩形会对这些区间进行可能的依次合并，分析可知是均摊 $O(1)$ 的。
因此本做法为 $O(n \log n)$ 时间复杂度。

E. Inverse Counting Path

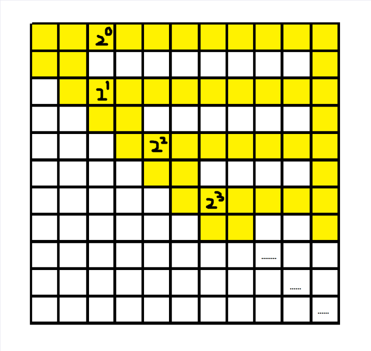
题意

构造一个 $n \times n$ $n \leqslant 30$ 列 01 矩阵，令一些地方不能通过，使得从左上到右下方案数恰好为 x。

E. Inverse Counting Path

题解

首先考虑对于 x 进行二进制分解，构造若干个 2×2 的块，左上到右下方案是 2 ，再引出若干“导线”来传送二进制幂。

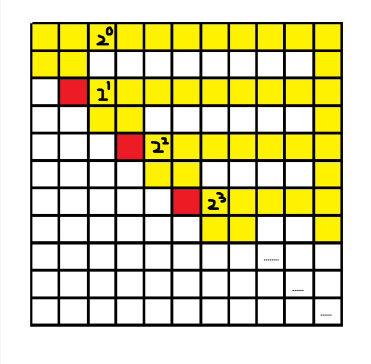


这样大概需要 $n = 60$。

E. Inverse Counting Path

题解

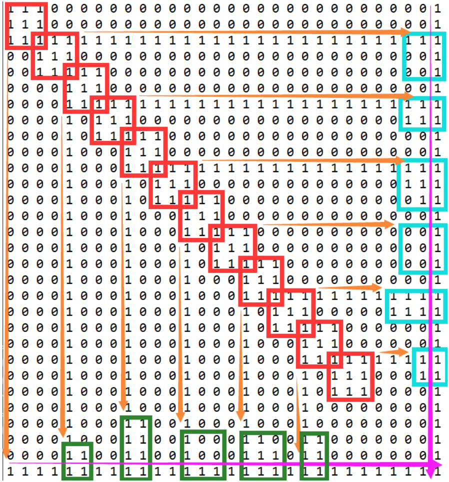
考虑这种做法浪费了许多格子，如图，红色部分是毫无意义的，它们单纯的传递了 DP 值，我们希望能去掉这部分，我们能否尝试重叠从而高效利用格子？



E. Inverse Counting Path

题解

考虑是否更优秀的进制？若以 3×3 作为基块会发生什么？这个六进制下，我们能够既共用格子，并且引出不贴贴的“导线”！此图右下侧给出了六进制下 0 到 5 的构造，该做法需要 $n = 30$。



E. Inverse Counting Path

题解

另外或许存在别的基块和进制，能构造出 n 更小的做法？
以及事实上，一些优秀的随机化做法也可以通过本题，其中一个思路是参考题目《智商锁》，分成四个乘法块和若干加法块儿。

F. Inverse Manacher

题意

给定某一个仅由 ab 字符构成的串的回文长度序列（使用 Manacher 算法），要求恢复该串。

F. Inverse Manacher

解法

这题出题人的原始做法是：
由于字符集大小只有 2，因而在回文串匹配过程中相等与不相等的关系可以转化为 2-sat 问题。
考虑倒过来执行 Manacher 算法——维护最右侧的回文子串。如果当前字符不超过最右侧回文子串范围，则证明该点已经与对称轴左侧的对应点匹配；否则暴力扩展最右侧的范围，并根据 a_i 的范围暴力向外延申（种类并查集或建图维护相等关系）。
因而总复杂度为线性。
但是由于这题保证有解，并且还给出了 '|' 两侧的回文长度。因而只需要关注 '|' 处的回文长度，如果该点回文长度仅为 1 则证明发生了 'a' 和 'b' 字符的交替，反之没有。因而总复杂度同为线性。

G. Guess the Polynomial

题意

交互题。存在一非零项不超过 n 个的多项式 $f(x) = \sum_{i=1}^{n} a_i x^{p_i}$，$1 \leqslant n \leqslant 1 \times 10^3$，$1 \leqslant a_i \leqslant 1 \times 10^5$，$0 \leqslant p_i \leqslant 8 \times 10^6$，进行不超过 2×10^4 次询问 x，返回 $f(x) \bmod 998\,244\,353$，求 $f(x)$。

G. Guess the Polynomial

题解

记 $f(x) = \sum_{i=0}^{P-2} a_i \cdot x^i$（这里 a_i 与题面含义不同）。
令

$$A_n^k = \sum_{i|i \bmod n = k} a_i = \frac{1}{n}\sum_{i=0}^{n-1} f(\omega_n^i)\,\omega_n^{-ki}$$

$$B_n^k = A_{2n}^k - A_{2n}^{k+n} = \frac{1}{n}\sum_{i=0}^{n-1} f(\omega_n^i\omega_{2n})\,\omega_n^{-ki}\omega_{2n}^{-k}$$

这里 ω_n 为 n 次单位根，在模 998 244 353 意义下 $\omega_n = g^{(P-1)/n}$ $(n \mid P-1)$，其中 g 为 P 的原根。

G. Guess the Polynomial

题解

假设对所有 $0 \leqslant k < n$ 已知 A_n^k 的值。由于 $A_{2n}^k = (A_n^k + B_n^k)/2, A_{2n}^{k+n} = (A_n^k - B_n^k)/2$，只需求出所有 B_n^k 即可得到 A_{2n}^k。关于 B_n^k 与 $f(\omega_n^k \omega_{2n})$ 的关系存在以下方程：

$$\begin{bmatrix} \omega_n^{0\cdot 0} & \omega_n^{0\cdot 1} & \cdots & \omega_n^{0\cdot (n-1)} \\ \omega_n^{1\cdot 0} & \omega_n^{1\cdot 1} & \cdots & \omega_n^{1\cdot (n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_n^{(n-1)\cdot 0} & \omega_n^{(n-1)\cdot 1} & \cdots & \omega_n^{(n-1)\cdot (n-1)} \end{bmatrix} \cdot \begin{bmatrix} B_n^0 \omega_{2n}^0 \\ B_n^1 \omega_{2n}^1 \\ \vdots \\ B_n^{n-1} \omega_{2n}^{n-1} \end{bmatrix} = \begin{bmatrix} f(\omega_n^0 \omega_{2n}) \\ f(\omega_n^1 \omega_{2n}) \\ \vdots \\ f(\omega_n^{n-1} \omega_{2n}) \end{bmatrix}$$

G. Guess the Polynomial

题解

由于题目限制多项式系数 $a_i \leqslant 10^5$，因此对于任意集合 $S \subseteq [0, P-2]$ 有 $\sum_{i \in S} a_i = 0 \Rightarrow \forall i \in S, a_i = 0$。若 $A_n^k = 0$，由 $A_n^k = A_{2n}^k + A_{2n}^{k+n}$ 可得 $A_{2n}^k = A_{2n}^{k+n} = 0 \Rightarrow B_n^k = 0$。设所有满足 $A_n^k \neq 0 \wedge B_n^k \neq 0$ 的 k 集合为 $\{k_0, k_1, \ldots, k_{m-1}\}$。令 $\alpha_i = \omega_n^{k_i}, b_i = B_n^{k_i}\omega_{2n}^{k_i}, f_i = f(\omega_n^i \omega_{2n})$，则上述方程可化简为：

$$
\begin{bmatrix}
\alpha_0^0 & \alpha_1^0 & \cdots & \alpha_{m-1}^0 \\
\alpha_0^1 & \alpha_1^1 & \cdots & \alpha_{m-1}^1 \\
\vdots & \vdots & \ddots & \vdots \\
\alpha_0^{m-1} & \alpha_1^{m-1} & \cdots & \alpha_{m-1}^{m-1}
\end{bmatrix}
\cdot
\begin{bmatrix}
b_0 \\ b_1 \\ \vdots \\ b_{m-1}
\end{bmatrix}
=
\begin{bmatrix}
f_0 \\ f_1 \\ \vdots \\ f_{m-1}
\end{bmatrix}
$$

G. Guess the Polynomial

题解

令

$$M = \begin{bmatrix} \alpha_0^0 & \alpha_0^1 & \cdots & \alpha_0^{m-1} \\ \alpha_1^0 & \alpha_1^1 & \cdots & \alpha_1^{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m-1}^0 & \alpha_{m-1}^1 & \cdots & \alpha_{m-1}^{m-1} \end{bmatrix}$$

则有

$$M_{i,j}^{-1} = [x^i] \prod_{k \neq j} \frac{x - \alpha_k}{\alpha_j - \alpha_k}$$

G. Guess the Polynomial

题解

因此

$$
\begin{aligned}
b_i &= \sum_j (M^{-1})_{j,i} f_j \\
&= \sum_j f_j [x^j] \prod_{k \neq i} \frac{x - \alpha_k}{\alpha_i - \alpha_k} \\
&= \prod_{j \neq i} \frac{1}{\alpha_i - \alpha_j} \cdot \sum_j f_j \cdot [x^j] \frac{F(x)}{x - \alpha_i}
\end{aligned}
$$

其中 $F(x) = \prod_i (x - \alpha_i)$。对所有 $0 \leqslant i < m$ 询问 $f(\omega_n^i \omega_{2n})$（即 f_i），则可以在 $O(m^2)$ 时间内求出所有 b_i。由于 A_n^k 非零项不超过 10^3，因此总询问次数不超过 $1 + \sum_{i=1}^{23} \min(10^3, 2^{i-1}) < 2 \cdot 10^4$。

G. Guess the Polynomial

题解

关于怎么求 M 的逆矩阵。对于任意 y_i 有
$M \cdot [y_0\ y_1\ \cdots\ y_{m-1}]^T = F([\alpha_0\ \alpha_1\ \cdots\ \alpha_{m-1}]^T)$，其中 $F(x) = \sum_{i=0}^{m-1} y_i x^i$。
用 $(\alpha_i, F(\alpha_i))$ 对 F 拉格朗日插值有

$$F(x) = \sum_j F(\alpha_j) \prod_{k \neq j} \frac{x - \alpha_k}{\alpha_j - \alpha_k}$$

因此 $y_i = [x^i] \sum_j F(\alpha_j) \prod_{k \neq j} \frac{x-\alpha_k}{\alpha_j - \alpha_k}$
又因为 $y_i = \sum_j M^{-1}_{i,j} F(\alpha_j)$，因此

$$M^{-1}_{i,j} = [x^i] \prod_{k \neq j} \frac{x - \alpha_k}{\alpha_j - \alpha_k}$$

H. Binary Craziness

题意

给定一个 $G(n, m)$（存在重边和自环），求 $\left(\sum_{i=1}^{n}\sum_{j=i}^{n} f(i, j)\right) \bmod 998\,244\,353$，其中 $f(u, v) = (deg_u \oplus deg_v)(deg_u \mid deg_v)(deg_u \mathbin{\&} deg_v)$

H. Binary Craziness

题意

给定一个 $G(n,m)$（存在重边和自环），求 $\left(\sum_{i=1}^{n}\sum_{j=i}^{n} f(i,j)\right) \bmod 998\,244\,353$，其中 $f(u,v) = (\deg_u \oplus \deg_v)(\deg_u \mid \deg_v)(\deg_u \,\&\, \deg_v)$

题解

由握手定理可知，$\sum_{i=1}^{n} \deg_i = 2m$。注意到本质不同的 $\deg_u$ 数量仅为 $\mathcal{O}\left(\sqrt{2m}\right)$，因而直接暴力枚举本质不同的 $\deg_u$ 与 $\deg_v$ 即可。总复杂度 $\mathcal{O}(2m)$。

I. Step

题意

求最小的 t ，使得对于 $k \in [1, n]$ ，$\sum_{i=1}^{t} i$ 为 a_k 的倍数。

题解

令 $M = \text{lcm}\{a_1, a_2, \ldots, a_n\}$ ，则题意等价为求最小的 t ，使得 $t(t+1)$ 为 $2M$ 的倍数。
对 M 质因数分解，记 $M = \prod_{i=1}^{m} p_i^{k_i}$ ，可知 $p_i \leqslant 10^7$ ，由于 t 与 $t+1$ 互质，每种质因子一定只能是 t 或者 $t+1$ 其中一个的质因子。
由于 $M \leqslant 10^{18}$ ，经过计算可以得到 $m \leqslant 15$ ，因此可以以 $O(2^m)$ 的复杂度枚举每个质因子属于 t 还是 $t+1$ 。

题解

将质因子分为 A, B 两个集合，记

$$a = \prod_{(p_i,k_i)\in A} p_i^{k_i}, b = \prod_{(p_i,k_i)\in B} p_i^{k_i}$$

且存在正整数 x, y ，$t = ax, t + 1 = by$ 。

则 $by - ax = 1$ ，问题转化为求出满足上式的最小的正整数 x 。

I. Step

题解

由于 $\gcd(a, b) = 1$ ，问题一定有解，可以通过 exgcd 得到 x ，即得到 $t = ax$ ，答案即为枚举过程中最小的 t。
时间复杂度 $O(2^m \log M)$ 。

J. Expansion

题意

给定长度为 n 的序列 $\{a_n\}$。初始时没有资源，且已经占据第一个点，从第一个点开始依次向右占领。假设已经占领到第 x 个点，则每秒资源增加 $\sum_{i=1}^{x} a_i$。问保证在无穷时间内资源数量非负的情况下，最快何时能占领完这 n 个点。

J. Expansion

解法

找到右侧第一个大于目前停留位置的前缀和的点，若资源足够到那里，扩张过去。
考虑到扩张格子的过程总是需要那么多时间，因此逐步的在收益更多的地方停留是正确的。
可能会使用前缀和等去模拟这个扩张过程。

K. Dice Game

题意

$n+1$ 个人每个人各投一个 m 面的骰子，点数最小的人失败。如果有多个最小值，则继续投直到出现失败者。问一号的骰子一直投出 x，其余人随机使得一号失败的概率，对 $1 \leqslant x \leqslant m$ 输出答案。

K. Dice Game

解法

显然对于 $[2, n+1]$ 号人而言，他们对于游戏的贡献是相同且独立的。因而只考察一个人的答案。不妨考虑 2 号丢骰子的情况，且 1 号固定投出 x 点数。

如果 2 号第一次丢骰子投出大于 x，则 1 号直接输，$\Pr[1] = \frac{m-x}{m}$。

对于 2 号第 i 次丢骰子，前 $i-1$ 次都等于 x，第 i 次大于 x，则 1 号输。有概率 $\Pr[i] = \frac{m-x}{m^i}$。

因而 1 号输的概率为 $\sum_{i=1}^{+\infty} \Pr[i] = (m-x)\sum_{i=1}^{+\infty} \frac{1}{m^i} = \frac{m-x}{m-1}$。

因而最后 1 号固定投出 x 输的总的概率为 $\left(\frac{m-x}{m-1}\right)^n$。

L. Game

题意

每次甲可以从左侧拿走形如 $s_l = \underbrace{0\ldots0}_{x}\underbrace{1\ldots1}_{y}$ 的串，或从右侧拿走形如 $s_r = \underbrace{1\ldots1}_{y}\underbrace{0\ldots0}_{x}$ 的串，其中 $x \geqslant 1, y \leqslant 1, x + y \geqslant 2$，类似的，将甲的 01 串 flip 后得到乙的情况。

若串只剩下一个元素，那么如果是 0，甲可以操作，否则乙可以操作。

L. Game

解法

令 `solve(l,r,typ)` 为当前子问题，表示目前处于区间 $[l,r]$ 且目前该甲还是乙行动，下面以甲为例，假设：

(一). 当前串形如 1...1，则负

(二). 假设当前串形如 $\underbrace{0\ldots0}_{x}1\ldots1\underbrace{0\ldots0}_{y}$，$x \geqslant 1, y \geqslant 1$，则可直接判断胜负，方法如下：

这里存在一个观察，如果某个拿取方案能让对面元素暴露出来，一定顺手多拿一个是不劣的，因此在当前串左右两侧各补一个 0 表示被对面顺走的东西（便于后续讨论）。

接着考虑是否存在堵塞对方的方案，具体的，从左侧向右侧找到第一个 0000 或 0010 或 1111 或 1101 并记录，然后从右往左找到这些串的第一个 reverse 形式（左侧右侧各找一个），表示在此侧之前只能轮流，这是第一个甲或乙能连续拿两次的位置。

L. Game

解法

1. 若任意一侧记录的以 0 打头，则胜。
甲可以从此侧一直取到此处。

2. 若两侧记录的均以 1 打头，且两次记录不是同一个串，则败。
甲不管如何取一定先在一侧被堵塞后在另一侧再被堵塞。

3. 若两侧记录的均以 1 打头，且两次记录是同一个串。
从这个记录的 1111 （注意，若是同一个串，不可能为 1101 或 1011）分别往左右侧移动找到第一个出现的 111 或 000。

L. Game

解法

3. 若两侧记录的均以 1 打头，且两次记录是同一个串。
从这个记录的 1111 （注意，若是同一个串，不可能为 1101 或 1011）分别往左右侧移动找到第一个出现的 111 或 000。

(1) 若存在一侧先出现 000，则胜。
甲的决策为先走这一侧直到碰到 000（假设在左侧碰到），然后走另一侧。
到 1111 时为乙决策点，若乙拿走 11，那么剩余串为 $001\{1001\}^n$，甲拿走左侧后两人均无决策，甲胜；若甲拿走 0111，那么剩余串为 $001\{1001\}^n10$，甲拿走右侧后，乙仍无法胜利。
eg1. $\underline{01}_1\underline{10}_2 0011\underline{110}_3$, eg2. $\underline{001}_3 10011\underline{11}_2\underline{10}_1$, $0011\underline{0}_3\underline{0111}_2\underline{10}_1$

(2) 否则，负
甲没有决策，乙取走 0111 后剩余串为 $\{0110\}^n$，甲无法赢。
eg1. $\underline{01}_1\underline{10}_2\underline{0111}_4\underline{10}_3$, eg2. $01100\underline{111}_2\underline{10}_1$

L. Game

解法

4. 若无任何记录
查询是否存在 000，若存在，则胜，否则败。
串为 $\{0110\}^n0\{0110\}^n$，甲可以调整最后到 000，然后取走最后一个单 0

(三) 假设当前串形如 $\underbrace{0\ldots0}_{a}1\ldots0\underbrace{1\ldots1}_{b}$，$a \geqslant 1, b \geqslant 1$ 设左侧甲能连续操作次数为 x，右侧乙能连续操作次数为 y，且左侧操作 x 次后可得到的最右端位置为 lp，右侧 y 次后的最左端位置为 rp，即

$\underbrace{0...1}_{共\,x\,次}\underbrace{1}_{lp}\ldots\underbrace{0}_{rp}\underbrace{0...1}_{共\,y\,次}$

若 $x > y + 1$ 则胜，若 $x < y$ 则负，分别代表甲堵住乙或乙堵住甲，其余情况需要分类讨论。

L. Game

解法

令 lp_{-1}, rp_{-1} 分别表示时 lp 与 rp 的前一个位置（上次从此侧取串的位点）

1. 若 $lp - 1 \leqslant rp$
   则左右两侧不重叠，中间部分为 $11\ldots00$ 或 10 或空。
   (1) 若 $x = y + 1$
   存在可决策点在子问题 $solve(lp_{-1}, rp, typ)$ 处，此时两侧均为 0，进入情况 2 继续。
   (2) 若 $x = y$
   可进入 $solve(lp, rp_{-1}, typ^1)$
   是否还存在其他决策空间？

L. Game

解法

1. 若 $lp - 1 \leqslant rp$
则左右两侧不重叠，中间部分为 $11\ldots00$ 或 10 或空。

是否还存在其他决策空间？

令 lmx 表示左侧最后一次遇到 000 时所处的操作次数，rmx 表示从右侧最后一次遇到 111 时所处的操作次数。

(1) 若 $lmx > rmx$，甲可以一次拿走原来两次的量，调整至 $x = y$ 的情况。

(2) 若 $lmx \leqslant rmx$，乙可以一次拿走原来两次的量，调整至 $x = y + 1$ 的情况。

由于进入子问题后不存在需要递归处理的决策，因此暴力枚举可能的调整即可。

L. Game

解法

2. 若 $\mathrm{lp}-1>\mathrm{rp}$

串本质上变成了此情况 $0\ldots\underbrace{0}_{\mathrm{rp}}0101\ldots0101\underbrace{1}_{\mathrm{lp}}\ldots1$，中间发生了重叠。

两个人的决策空间也只有一次拿走原来两次的量以调整奇偶性，同样求出 lmx 与 rmx，若 $\mathrm{lmx}>\mathrm{rmx}$，那么多枚举一个 lmx 处多拿一次的情况，否则枚举 rmx 处多拿一个的情况。枚举后不存在决策了，模拟即可。

M. Different Billing

题意

A 类队伍不收费，B 类队伍一队一千，C 类队伍一队两千五，构造一个共 x 队，收到 y 元的方案。

解法

注意到 A 类队伍不收费，B 类队伍和 C 类队伍收费标准都在一千美元以上，因而总收费虽然在 10^9 级别，但是 B 和 C 类队伍的数目只有 10^6 级别。因而可以直接枚举 B 类队伍数目即可。