水题选讲

彭博

北京大学

2023.7

1st ucup stage 14 C LaLa and Lamp

有 $n(n+1)/2$ 个点排成一个正三角形，每个点是黑色或者白色。

你每次可以选择横着的一行，或者向左 60° 的一行，或者向右 60° 的一行，把这一行上的所有点的颜色取反。

判断能否把所有点的颜色都变成白色。

$n \leq 2000$

1st ucup stage 14 C LaLa and Lamp

可能的操作数只有 $3n$ 种，但限制有 $O(n^2)$ 个，因此只需要很小一部分的限制就足以确定唯一解，然后再判断合法。

发现最下面两行非常不错：只要枚举两行是否操作，然后就有恰好 $2n-1$ 个限制和 $2n$ 个变量，每个限制恰好连接两个变量。

然后模拟就好了。

CF1776C Library game

有一个长度为 m 的书架，以及 n 个长度 $a_1, \cdots, a_n$ 。

Alice 和 Bob 从书架上取书。每次由 Alice 选择一个之前没选过的 i ，并选择一个长度为 a_i 的区间，需要保证这个区间内的书全都没有被取过。然后 Bob 从区间内任意拿走一本书。

Bob 的目标是让 Alice 某一步没有区间可选。问谁赢，并交互构造方案。

$n \leq 100, m \leq 5000$

CF1776C Library game

结论就一句话。

可以看看 $n = 2$ 怎么做。

CF1776C Library game

把 a 从大到小排序之后，如果存在 $a_i > \lfloor m/i \rfloor$，那么 Bob 胜，否则 Alice 胜。

Alice 只需要每次贪心找空就行。Bob 只需要每次覆盖一个等分点就行。

XXI Open Cup GP of Korea – B

给定长度为 n 的序列 a 和长度为 m 的序列 b 。

有一个 $n \times m$ 的网格图，网格图上的点 (i, j) 能走当且仅当 $a_i + b_j \geq 0$ 。并且在这个网格图上只能向右或向下走。

问有多少个 (S, T) ，使得 $(S, 1) \to (T, m)$ 的路径存在。

$n, m \leq 2 \times 10^5$

XXI Open Cup GP of Korea – B

地图上能走的位置很难描述，比较诡异。

但是找最优路径并不需要描述地图。可以想到一个贪心：从 (x, y) 出发，尝试走到 (x, z) 使得 $b_z > b_y$ ，或者走到 (z, y) 使得 $a_z > a_x$ 。只要存在这样一种走法（并且路上不存在更大的位置），那么直接走过去显然是好的。

先假设 $S = 1, T = n$ ，看怎么判是否合法。可以找到一个 (X, Y) ，使得 $a_X = \max a, b_Y = \max b$ ，然后就只需要判断能否从 $(1, 1)$ 走到 (X, Y) ，以及从 (n, m) 倒着走到 (X, Y) 。这个判断就可以用前后缀最大值什么的贪心。

XXI Open Cup GP of Korea – B

那么如何对所有 S, T 判断呢？注意 Y 是不变的，而且 m 这一维的所有前后缀最大值都不会变。然后 S, T 之间会有一些变动，但是并不难用数据结构处理。

当然，我是不会在这里涉及太多数据结构的细节的（

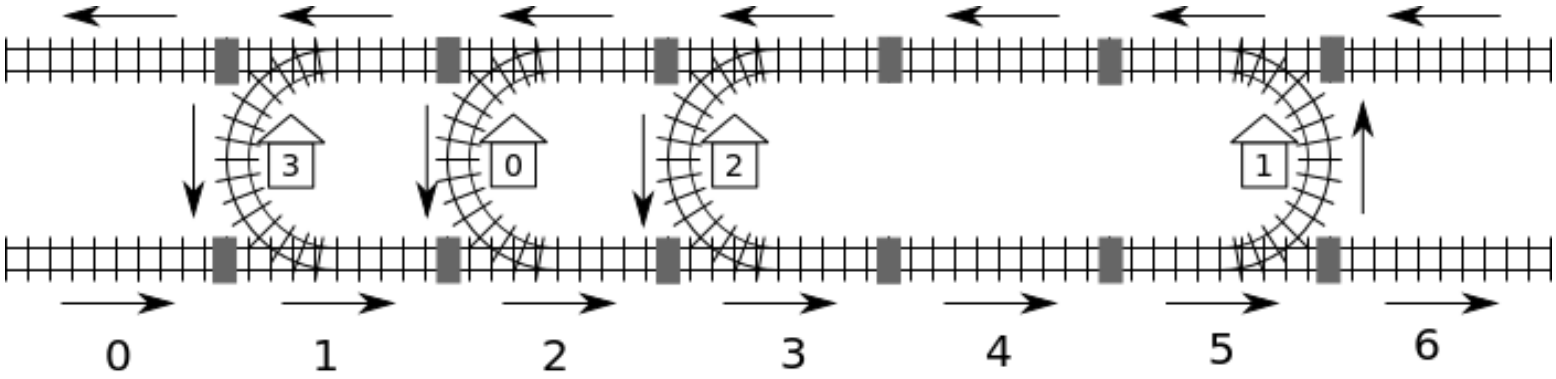
XXI Open Cup GP of Korea – B

最后可以抽象出四个条件：

- $\forall i, a_i + b_Y \geq 0$
- $\forall j, a_X + b_j \geq 0$
- $\forall (i,j), (\exists k < i, a_k + b_j \geq 0) \vee (\exists k < j, a_i + b_k \geq 0)$
- $\forall (i,j), (\exists k > i, a_k + b_j \geq 0) \vee (\exists k > j, a_i + b_k \geq 0)$

对每个条件分别可以得到若干个不合法的矩形，然后扫描线即可。

UOJ24. 【IOI2014】Rail



UOJ24. 【IOI2014】Rail

本题是交互题。

铁路被分成了若干段，每一段可能有车站。如果有车站，那么可能是 C 类或 D 类，分别表示不同的轨道弯曲方向和火车走的方向。

两个车站之间的距离定义为从一边走到另一边所经过的最少的连接器数量。连接器就是上面的灰色方块。(容易发现 $dis_{x,y} = dis_{y,x}$ ，所以就不用区分从哪到哪了。)

你现在只知道车站 0 所在的位置，以及它是 C 类。你可以询问两个车站之间的距离，求出所有车站的位置和类型。

$n \leq 5000$ ，询问次数 $\leq 3(n-1)$ 。

UOJ24. 【IOI2014】Rail

首先把两个站点之间的路线分析一下，发现有 4 种情况。对着这 4 种情况想做法会比较清晰。

先把 $dis_{0,i}$ 全部求出来，设 p 是其中的最小值，那么 p 就是 0 右边第一个 D 类站点。

再把 $dis_{p,i}$ 全部求出来。此时对于一个 x ，如果 $dis_{0,x} = dis_{0,p} + dis_{p,x}$ ，那么说明 x 在 p 左边，否则在右边。在 p 左边的站点包含在 0 左边的和在 $0, p$ 之间的，不过这两者很容易区分。

因为左右是对称的，下面就只考虑右边了。

UOJ24. 【IOI2014】Rail

因为铁路的特性，我们发现，如果一个 C 类站点 x 右边的第一个 D 类站点是 y，那么从左边看来，x 和一个在 $2loc_y - loc_x$（即关于 y 对称）的 D 类站点是没有区别的，因此从左边尝试分辨 C 和 D 没有前途，只能在右边的内部分辨。

把所有站点按 $dis_{0,i}$ 排序，暂时认为它们都是 D 类站点，给它们确定一个 loc'_i。从左往右扫，设上一个确定是 D 类站点的位置是 loc_q。对于当前考虑的 x，询问 $d = dis_{q,x}$，那么它可能是 loc'_x 的一个 D 类站点，也可能是 $loc_q - d$ 的一个 C 类站点。设这两个位置分别为 p_1, p_2。

UOJ24. 【IOI2014】Rail

我们发现，如果是后者，那么说明 $(p_1+p_2)/2$ 的位置有一个 D 类站点（这样才能从 p_1 对称到 p_2 ）；而如果是前者，那么为了使得 x 到 p 的距离是 d ，就需要 $(p_1+p_2)/2$ 有一个 C 类站点。

只要存在 D 类站点，就一定已经确定了。所以只需要看 $(p_1+p_2)/2$ 是否确定过一个 D 类站点即可。

CF1477D Nezzar and Hidden Permutations

给定一个无向图，把它定向成 DAG ，然后选取两个拓扑序使得对应位置相等的个数最少，输出方案。

$n, m \leq 5 \times 10^5$

CF1477D Nezzar and Hidden Permutations

对于一个 DAG，显然答案的下界是这样的点 x 的个数：对于任意一个其他点 y，要么存在路径 $x \to y$，要么存在 $y \to x$。

那么对于一个无向图，下界就是度数为 $n-1$ 的点的个数。然后用构造的方法证明这个下界可以取到。

取补图，然后对每棵树分别构造即可。一种简单的构造方法是划分成若干个菊花。

XXI Open Cup GP of Krakow – H

有一棵 n 个点的树，树上有 K 个老鼠和 K 个洞。保证老鼠位置两两不同且不相邻，洞也是。

每个时刻一个老鼠可以往相邻的点移动一步。移动之后仍然需要保证两两不相邻（即使处在洞中也不能相邻）。

问能否让最终的状态每个洞都有恰好一只老鼠。

$n \leq 2000$

XXI Open Cup GP of Krakow – H

注意到操作是可逆的，所以可以选取一个中间状态，只要两边都能走到这个中间状态就合法。

一个显然的直觉是要把一些老鼠尽量往叶子塞，这样它们就能对其他老鼠造成尽可能小的影响。

因此可以随便拿一个挂在边上的菊花出来，然后看两边是否都能往里面同一个点塞一只老鼠。不管是否可以，塞完之后这个菊花都不会再对外界造成任何影响了。

容易 $O(n)$ 判断是否可行，所以复杂度 $O(n^2)$ 。

CF1776M Parmigiana With Seafood

给定一棵 n 个点的无根树，两人轮流操作，每次删去一个叶子，直到把树删空。

先手希望最大化自己删去的编号最大的点，而后手希望最小化对方删去的编号最大的点。求最终结果。

$n \leq 10^5$

CF1776M Parmigiana With Seafood

首先考虑最简单的情况：先手能否保证删到 n 。

不难发现两人此时的策略都是只要能不把 n 变成叶子就随便操作，直到最后只剩三个点时没有办法。因此当 n 是偶数时答案就是 n ，下面都认为 n 是奇数。

然后考虑 $n-1$ ，不难发现如果 $n-1$ 到 n 的距离是奇数（或者说点数是偶数），那么把这条路径看做一个大点，先手用同样的策略就可以保证删到 $n-1$ 和 n 中的一个。

CF1776M Parmigiana With Seafood

黑白染色，然后二分答案 k 。只要 $k, k+1, \cdots, n$ 里面有一个点的颜色不同，答案就一定 $\geq k$ 。

而当颜色全部相同时怎么办呢？此时不能只考虑两个点了，甚至不能只考虑一条链上的多个点，必须利用度数 ≥ 3 的点。

不妨设 n 的颜色是黑色。手玩一下发现，如果存在四个黑色点 x, a, b, c ，使得 x 处于 a, b, c 中间，那么就可以保证删掉 a, b, c 中的一个。

还能再拓展吗？用奇偶性分析手玩一下，发现不行了。

因此二分答案之后乱搞搞就好了。

ARC125F Tree Degree Subset Sum

给出一棵树，设第 i 个点的度数是 d_i 。

问有多少对 (x, y) ，使得存在一种选出 x 个点的方式，它们的度数加起来恰好为 y 。

$n \leq 2 \times 10^5$

ARC125F Tree Degree Subset Sum

树的具体形态显然用处不大：只要 $\sum d_i = 2n-2$ 都可以构造出一棵合法的树。

我们的主要问题出在多了一维 x 。因为什么性质也找不出来，尝试把 x 这一维消掉。

先把所有 d_i 都减掉 1 ，那么 $\sum d_i = n-2$ 。我们尝试证明：对于总和 y ，设 $m(y), M(y)$ 分别是最小和最大的 x ，那么 $x \in [m(y), M(y)]$ 都有合法方案。

我们的证明主要和 0 的个数 z 有关。因为 $m(y)$ 必然一个 0 都没选，而 $M(y)$ 选了 z 个 0 ，所以只要 $M(y) - m(y) \le 2z$ ，就可以通过调整 0 的个数构造出每个 x 。

发现对于任意一个集合 S ，都有 $-z \le y(S) - x(S) \le z-2$ ，所以 $-z \le y - M(y) \le y - m(y) \le z-2$ ，所以 $M(y) - m(y) \le 2x-2$ ，然后就证完了。

CTS2022 D1T2

有一个 n 个点的树，每个点有一个点权 a_x 。

你每次可以选择一个点 x ，令 $a_x := -a_x + \sum_{(x,v)\in E} a_v$ ，然后把 a_v 清零。

你可以执行任意次操作。你需要使得最终只有至多一个点的点权非零，并最大化这个点权。

$n \le 10^5$

CTS2022 D1T2

考虑对 x 连续进行两次操作会发生什么。第一次操作会把周围的数吸过来，而第二次操作仅仅是把 a_x 取反。

因此，通过选择是否进行第二次操作，我们可以任意调整 a_x 的正负性。

因此，无论接下来怎么操作，我们总是可以让 a_x 对答案做出 $|a_x|$ 的贡献。

所以可以直接忽略 a_x 。

所以一次操作就是把一个点及其邻居全部清零。用树形 DP 来枚举不同的操作顺序即可。

Ptz Winter 2021 Day3A Arrange The Piranhas

有一个长度为 n 的棋盘，棋盘上共有 k 个棋子，位置分别是 $1 \le p_1 < \cdots < p_k \le n$ 。

你每次可以选定一个没有棋子的位置 x ，然后把 x 左边第一个棋子向右移动一格，右边第一个棋子向左移动一格。特别的，如果左边没有棋子就只移动右边，右边同理。你需要保证移动之后 x 仍然没有棋子。

你需要用最少的次数把棋子的位置变为 $q_1 < \cdots < q_k$ ，或判断无解。

$1 \le k \le n \le 1000$

Ptz Winter 2021 Day3A Arrange The Piranhas

设 s_i 表示前 i 个棋子的位置之和，那么一次操作只会让某个 s_i 加一，或者让所有 s_i 减一。设 s_i 加一的次数是 d_i ，所有 s_i 一起减一的次数的 d_0 。

先令 $d_0 = 0$ ，可以推出所有 d_i 。然后把 d_0 调大，使得 d_i 非负。

d_i 就是要在第 i 个棋子和第 $i+1$ 个棋子之间操作的次数。不难发现，给 i 操作一次时，$i-1$ 和 $i+1$ 都更容易操作了，因此贪心操作不劣，操作顺序不会影响结果。

严谨一点地说，就是固定一个操作序列 $m_1, \cdots, m_L$ ，表示第 i 次操作 m_i 。那么当我这次想要操作 x 时，我就把操作序列里第一个 x 拖到最前面来，这样剩下的仍然是一个合法的操作序列。

剩下唯一的问题就是有没有可能把 $d_0, d_1, \cdots, d_n$ 整体调大，使得原来不存在合法操作序列，现在存在了。

Ptz Winter 2021 Day3A Arrange The Piranhas

可以证明，调大 d_0 如果存在合法的操作序列，就可以推出原来也存在合法的操作序列。证明方法就是把第一个 0 、第一个 1 、……、第一个 n 都拖到最前面来。这样尽管前 $n+1$ 次操作可能存在不合法的操作，但把前 $n+1$ 次操作截掉之后剩下的就是合法操作序列了。并且操作 $0, 1, \cdots, n$ 恰好对 p 毫无影响，因此剩下的操作序列就对应到一个调大 d_0 之前的操作序列。

所以确定了最小的 d_0 之后直接贪心即可。

ARC134F Flipping Coins

有 n 个硬币排成一排，一开始都是正面朝上。另外还等概率选择了一个长度为 n 的排列 p 。

从 1 到 n 枚举 i ，如果第 i 个硬币是正面朝上，那么就把 i 和 p_i 两枚硬币都翻转。

最后如果还有 k 个硬币朝上就获得 W^k 的收益。

求期望收益，模 998244353 。

$n \leq 2 \times 10^5$

ARC134F Flipping Coins

显然应该分开考虑 p 的每一个环。不难发现，对于一个环，把它的最小值转到第一个之后断环为链，那么最后朝上的硬币个数就是长度为奇数的极长上升子段个数。接下来的问题是怎么数数。

进行一些玄妙的转化。把所有环按照其最小值从大到小排序，然后排成一排得到排列 q 。可以发现 p, q 是一一对应的关系，而且朝上的硬币个数就是 q 的长度为奇数的极长上升子段个数。

奇数太魔怔了，再进行一个转化，把每个极长上升子段的奇数位置标 A ，偶数位置标 B ，如果长度是奇数则把最后一个位置标 C ，那么朝上的硬币个数就是 C 的个数。

ARC134F Flipping Coins

考虑这样的一个 ABC 串能对应到几个排列上。

我们发现，此时大小限制只剩两种：A 要小于后面的 B，C 要大于后面的 A 或 C。而 B 与它后面一个位置的大小关系则是任意的。

所以我们在每一个 B 的位置断开，把串分成若干段，每一段就必须要有序，而段之间没有任何限制。然后用生成函数随便搞搞即可。

LOJ#2393. 「JOISC 2017 Day 2」门票安排

有 n 个车站放在一个环上，相邻的车站有双向铁路相连。

有 m 组乘客，第 i 组有 c_i 个人，要从第 x_i 个车站到第 y_i 个车站。你需要给每个人选择两种路线中的一种，最小化每一段铁路被经过的次数的最大值。

$n \leq 2 \times 10^5, m \leq 10^5$

LOJ#2393. 「JOISC 2017 Day 2」门票安排

先二分答案 ans 。

断环为链，假装大家都不经过 $(1,n)$ 这条边，那么每个人就是给 $[x_i,y_i)$ 加一。然后可以选择翻转某些人，带来的效果就是 $[1,x_i),[y_i,n]$ 加一，$[x_i,y_i)$ 减一。

设 a_i 表示在翻转前每个位置被覆盖的次数，b_i 表示翻转后被覆盖的次数。

LOJ#2393. 「JOISC 2017 Day 2」门票安排

不难发现，只要枚举有几个区间被翻转了，就可以从左到右贪心。然而并不能枚举，所以要找性质。

首先可以发现，翻转的区间必然两两有交，否则两个都不翻转，b_i 不增。

那么可以设所有翻转区间的交是 $[l,r]$ 。这比单独拎一个位置出来要更好的地方在于我们现在可以确定有一个区间右端点是 r ，有一个区间左端点是 l 。

如果同时撤销上述两个区间的翻转，那么外面的 b 不增，但 $[l,r]$ 的 b 就会增加 2 。因此我们可以强制 $[l,r]$ 中的最大的 b_i 是 $ans-1$ 或 ans 。设最大的位置是 b_t 。

这时候发现，a_t 必然是 $[1,n]$ 中最大的。若不然，外面有一个更大的 a_i ，那么就必然有 $b_i>b_t+2$ ，与 $b_t\geq ans-1$ 矛盾。

LOJ#2393. 「JOISC 2017 Day 2」门票安排

同理可以证明，所有 a 的最大值必然都被包含在 $[l, r]$ 之内。

那我们任取一个最大值 a_t ，根据 $b_t \in \{ans - 1, ans\}$ 就可以得到翻转的区间个数，然后从左到右贪心即可。

AGC046E Permutation Cover

给定一个长度为 K 的序列 $a_1, \cdots, a_K$，并设 $n = \sum_{i=1}^{K} a_i$。你需要构造一个长度为 n 的序列 b，其值域为 $[1, K]$，且对于任意一个位置 i，都存在一个长度为 K 的子串包含位置 i，且这个子串是一个排列。

如果有多种方案，输出字典序最小的方案。

$n \leq 1000, K \leq 100$

AGC046E Permutation Cover

首先考虑如何构造一个方案。

从空串开始，每次往后加入 $\leq K$ 个数，使得加入之后最后 K 个数恰好是一个排列。

然后可以发现，这个操作其实没必要这么灵活。只要存在合法方案，就可以每次加入一个排列以及这个排列的前缀，这就足够了。

简单玩一玩，发现 $\max a \leq 2 \min a$ 的时候就可以构造出来。而 $\max a > 2 \min a$ 的时候不难证明无解。

AGC046E Permutation Cover

那么如何字典序最小呢？我们仍然用这个操作，这样可以保证任意时刻最后 K 个都是个排列。然后考虑怎么判断后面是否合法。

用同样的方法，不难证明此时仍然需要 $\max a \le 2\min a + 1$ 。

还可以发现，如果 $\max a = 2\min a + 1$ ，那么在最后这 K 个里面，$\min a$ 必须都在 $\max a$ 的右边。然后这也很容易构造。

所以每次枚举往后加的长度，用这个方法判一下，然后把字典序最小的合法串加上去即可。

XX Open Cup GP of Nanjing – M

给定一个 n 个点 m 条边的简单带权无向图，你需要找一个长度为 k 的简单环，最大化环上所有边的边权之和。

$n, m \leq 300, k \leq 10$

XX Open Cup GP of Nanjing – M

难点在于怎么保证 k 个点两两不同。

随机!

我们随机一种黑白染色方案，就有 $k/2^k$ 的概率可以使得环恰好被分成两段长度为 $k/2$ 的同色的链。

在 $k = 10$ 的时候只需要随机 1000 次就基本可以保证能随到一个合法的染色方案。接下来我们只需要对于一种染色方案在 $O(m^2)$ 的时间内做出来即可。

XX Open Cup GP of Nanjing – M

对于黑色点和白色点分别处理，设 $f_{x,y}$ 表示黑色点 x 到 y 的长度为 $k/2$ 的简单路径的边权最大值，g 同理。

合并的时候只需要枚举两条横跨黑白的边然后合并即可。

XX Open Cup GP of Nanjing – M

对 $k/2$ 分类讨论。我们直接跳到最难的情况，$k = 10$，链上有 5 个点，4 条边。

枚举第一条和最后一条边，那么中间只需要再选出一个点就够了。

我们可以另外设 $h_{x,y}$ 表示 x 到 y 中间再选一个点能得到的最大边权和。这个也只需要枚举两条边就可以求出来。并且不仅是最大边权和，我们可以把前三大的边权和都算出来。

因为两边只 ban 掉了两个点，所以中间的前三大方案中必有一个会被选，这就做完了。

CF1693F I Might Be Wrong

给定一个长度为 n 的 01 串 s，你每次可以任意选取一个区间 $[l,r]$，付出 $|cnt_0 - cnt_1| + 1$ 的代价把这个区间排序，其中 cnt_0, cnt_1 分别表示区间中 0 和 1 的个数。

求出把整个 01 串排序所需的最小代价。

$n \leq 2 \times 10^5$

CF1693F I Might Be Wrong

手造几个 1111100 的样例玩玩，发现好像总是可以使得每次操作都 01 个数相等。

这并不难证：不妨假设 $cnt_0 > cnt_1$ 且 $s_l = 1$，那么一定存在一个前缀 $[l, r']$ 使得 01 个数相等，那么操作 $[l, r]$ 的结果和代价与先操作 $[l, r']$ 再操作 $[l+1, r]$ 的结果和代价相同。继续调整即可。

CF1693F I Might Be Wrong

假设整个 s 里 0 比 1 多。

考虑把最靠左的 1 往右推，发现每次贪心推就好了：从第一个 1 开始找一个尽量长的区间使得 01 相等，贪心操作，直到推到底为止。

可以发现贪心操作非常优秀：任何多余的操作只会使得往后推的过程更加困难，不如不做。

CF725F Family Photos

有 $2n$ 个物品，每个物品有两个属性 a_i, b_i ，表示 Alice 取走它的收益和 Bob 取走它的收益。

这 $2n$ 个物品被分成了 n 对，每一对物品必须取走了第一个之后才能取第二个。

Alice 和 Bob 轮流取物品，双方都想最大化最后自己的收益减掉对方的收益。特别的，两个人也都可以选择放弃操作，当两人分别放弃一次操作之后游戏立刻结束。

$n \leq 10^5, a_i, b_i \geq 0$

CF725F Family Photos

如果没有先取第一个再取第二个的限制，显然每个物品最后都会被取走，于是简单推一下式子发现一个物品对双方的价值都可以视作 $a_i + b_i$ 。这时候直接从大到小贪心取即可。

考虑上这个限制，如果第一个物品的价值比第二个物品要大，显然这个限制也就没什么用——没有人会想先拿第二个。

而如果反过来，说明 $(a_{i,0} - b_{i,1}) + (b_{i,0} - a_{i,1}) < 0$ 。

如果这两项都小于 0 ，那么双方都不会在这里进行操作（否则对方可以跟一步）。

如果只有一项小于 0 ，不妨假设是 $a_{i,0} - b_{i,1} < 0$ ，那么 Alice 就不会在这里操作，但 Bob 可以。并且在这里一人一步不会改变先后手顺序，所以可以直接把 $a_{i,1} - b_{i,0}$ 加到答案里，然后把这一对物品删掉。

UOJ177. 新年的腮雷

给定可重集合 A, B ，其中 $|A| = n, |B| = m$ 。每次可以在 A 中选出 m 个元素 $x_1, \cdots, x_m$ ，合并成一个元素 $\max\{x_i + b_i\}$ ，放回 A 中。

保证 $n = 1 \pmod{m-1}$ ，所以最终一定可以合并成一个元素。求最终这个元素的最小值。

$n, m \leq 5 \times 10^4$ 。

UOJ177. 新年的腮雷

先把 b 从小到大排序，把 a 从大到小排序。

正着做太难了，完全不知道策略应该长啥样。

不妨二分答案之后倒着做，维护一棵 m 叉树，每次把一个叶子拆成 m 份。最后需要满足每个 a 都可以匹配一个比自己大的叶子。

问题又出现了：现在应该拆哪个叶子呢？

UOJ177. 新年的腮雷

考虑最大的那个叶子 x，如果不拆它，那么它就要匹配掉目前最大的 a。直觉告诉我们如果把它拆掉之后最大的 a 仍然能被匹配，那么似乎拆掉会更优。

当然，策略不会这么简单。

UOJ177. 新年的腮雷

经过一番冷静分析，我们考虑 $x - b_1$ ，它是以后能拆出的最大值了。考虑所有 $a_i > x - b_1$ ，如果它们在拆掉之后无法找到能匹配的叶子，那么这个 x 肯定是不能拆的。

否则是不是一定能拆呢？假设没有拆它，而是拆了更小的 y ，发现总是可以存在一种对应方法，使得拆了 x 之后剩下的叶子对应比拆 y 之后的叶子更大。

然后拿一棵线段树维护即可。

CF1810G The Maximum Prefix

有一个长度之至多为 n 的数组 a，第 i 个位置会以 p_i 的概率取 1，以 $1-p_i$ 的概率取 -1。

另外给定一个长度为 $n+1$ 的数组 $h_0, h_1, \cdots, h_n$。定义数组 a 的价值为 h_k，其中 k 为 a 的最大前缀和。

对于每个长度 $1 \le m \le n$，求出当 a 的长度为 m 时，其期望价值。

$n \le 5000$

CF1810G The Maximum Prefix

显然只要求出最大前缀和的分布就好了。

求最大前缀和有个经典算法：从后往前扫，维护当前的最大前缀和 s 。每次令 $s := \max(0, s + a_i)$ 即可。

因此如果只需要求长度为 n 的答案，就有一个非常显然的做法：设 $dp_{i,j}$ 表示从后往前做到第 i 个位置，当前的最大前缀和是 j 的概率，然后直接转移。最后根据分布算答案。

CF1810G The Maximum Prefix

然而，因为每个长度都要做，且长度为 i 的情况是从 $dp_{i,0}=1$ 为初值开始做（而不是在 dp_{n-i} 处算答案），这就导致朴素的 dp 无法把每个长度混在一起做。

不过注意到最终算答案时是用固定的系数 h 来把 $dp_{1,j}$ 线性组合起来，因此可以使用反推贡献的技巧。设 $f_{i,j}$ 表示 $dp_{i,j}$ 对最终答案的贡献系数，那么反过来 dp 就可以得到 f，然后长度为 i 的答案就是 $f_{i,0}$ 了。

AGC040E Prefix Suffix Addition

有一个长度为 n 的序列 a，初始全部为 0。

你每次可以做两种操作之一：

- 选择一个 k 和一个长度为 k 的单调不降的序列 x，对于 $1 \le i \le k$ 令 $a_i := a_i + x_i$。
- 选择一个 k 和一个长度为 k 的单调不降的序列 y，对于 $1 \le i \le k$ 令 $a_{n-i+1} := a_{n-i+1} + y_i$。

给出长度为 n 的序列 A，用最少的操作次数把 a 变成序列 A。问最少次数是多少。

$n \le 2 \times 10^5$

AGC040E Prefix Suffix Addition

考虑只有一种操作的时候会发生什么。比如只有操作 1，那么显然可以发现答案就是 $\sum_{i=1}^{n-1}[a_i > a_{i+1}]$。

把 a_i 拆成 $b_i + c_i$，分别表示操作 1 的贡献和操作 2 的贡献。

但是此时发现好像没有什么贪心策略是有效的，所以只能考虑 DP：设 $dp_{i,j}$ 表示做了前 i 个，$b_i = j$，最小代价。

容易发现 dp_i 单调不升。

然后可以注意到，如果强制 $dp_{i,0}$ 的方案是从 dp_{i,a_i} 仅仅修改这个位置得来，会得到 $dp_{i,0} \le dp_{i,a_i} + 2$。

也就是说 dp_i 的值只有三种，那就随便做了。

ARC119F AtCoder Express 3

有 $n+1$ 个车站，编号为 0 到 n 。

对于编号在 $[1, n-1]$ 的车站，它们要么是 A 类车站，要么是 B 类车站。而 0 号和 n 号车站则两个都是。

你从 0 号车站出发，每天可以向前或向后走到下一个车站，或是向前或向后走到下一个类型与自己相同的车站。特别地，0 和 n 号车站被视为与任何一个车站类型相同。

给定一个长度为 $n-1$ 的字符串，表示 $[1, n-1]$ 的车站的类型。如果对应位置为 ? 则表示类型未定。再给一个 K ，求有多少种把 ? 替换为 a 或 b 的方式，使得从 0 走到 n 的最少天数不超过 K 。

$n \leq 4000$

ARC119F AtCoder Express 3

观察一下，发现如果自己身前有一大段的相同字符，那么用另一种字符跳过去一定是最优选择。

那么说明任意时刻要考虑的字符数量应该不会太多，考虑建一个自动机来求出任意字符串对应的最短步数。

ARC119F AtCoder Express 3

最后发现有用的状态只有 13 个：

- A,AA,AAA,BA...A,AB ，人站在第一个位置。其中 A,AA,AAA 前面一格还会有个 B 。
- 上面 5 种状态取反得到另外 5 种。
- OA,OB,O ，其中 O 表示现在既可以站在 A 上也可以站在 B 上。

手玩一下即可建出自动机，然后暴力做。复杂度 $O(13n^2)$ 。

CF1667D Edge Elimination

给定一棵 n 个点的树，你每次可以删去一条边，但是要保证删掉这条边的时候两个端点的度数的奇偶性相同。

构造一个把所有边都删完的方案，或判断无解。

$n \leq 2 \times 10^5$

CF1667D Edge Elimination

直接构造，发现做不出。

注意到删边只会影响旁边两个点，有着非常好的局部性。

并且由于树的特殊性质，只要每个点给它周围的边一个删除顺序，就一定可以构造出一个整体的删除顺序，满足每个点的性质。

因此可以树形 dp 确定每条边它删除是时候两边的奇偶性是奇数还是偶数，同时分配一个旁边的点的删除编号。最后跑一个拓扑排序获得整体的顺序即可。

CF852H Bob and stages

给定平面上的 n 个点（保证没有三点共线），你需要选出 K 个点，使得它们组成一个凸包，且凸包内没有别的点。最大化凸包面积。

$n \leq 200, K \leq 50$

CF852H Bob and stages

数据范围这么小，看着就像是随便写个做法都能过。

所以我们不妨暴力一点，直接枚举凸包最左端的点是哪个。然后分别用 dp 枚举上下凸包。

以上凸包为例，设 $dp_{i,j}$ 表示走到点 i，且一共选了 j 个点，的最大面积。转移就直接枚举下一个点，就可以把新的面积加上，也保证凸包内部没有点了……？

CF852H Bob and stages

我们突然想起来还有“凸包”这个限制，所以肯定不能直接无脑枚举下一个点。

这个限制可以通过限制转移顺序来解决。还是有 n^2 种转移，但是把所有这些转移按照极角排序，就可以保证得出的方案是一个凸包了。

把所有转移都搞完之后再枚举最靠右的点，把上下凸包合并即可。

复杂度 $O(n^3K)$ 。

THU camp qualifier L Fence Decoration

给定长度为 n 的序列 x, c 和长度为 m 的序列 d，你需要选出一个长度任意的子序列 $i_1, \cdots, i_k$，满足 $i_1 = 1, i_k = n$，最大化

$$\left(\sum_{r=1}^{m}\sum_{t=2}^{k}|x_{i_t} - x_{i_{t-1}} - d_r|\right) - \left(\sum_{t=1}^{k} c_{i_t}\right)$$

$n, m \leq 10^5$

THU camp qualifier L Fence Decoration

直接设 dp_i 表示最后选的一个位置是 i 时前面的最大贡献。

注意到 m 个绝对值函数相加会得到一个凸函数，并且这个凸函数对任意 i 都是相同的。从 i 往后转移就是新增一个经过平移的凸函数，与之前的结果取 max 。

因为是同一个凸函数，所以可以得到（反的）决策单调性，用单调栈维护即可。

CF1608F MEX counting

给定 n, K 和一个长度为 n 的数列 b 。求有多少个值域在 $[0, n]$ 的长度为 n 的数列 a ，使得

$$\forall 1 \leq i \leq n, |\mathsf{MEX}(a_1, \cdots, a_i) - b_i| \leq K$$

模 998244353 。

$n \leq 2000, K \leq 50$

CF1608F MEX counting

考虑 $K = 0$ 怎么做。

$\mathrm{MEX}(a_1, \cdots, a_i) = b_i$ 蕴含两个限制，一个是 b_i 没有出现，另一个是 $[0, b_i)$ 都出现过。前者容易处理，而后者只需要容斥一下。

所以可以容斥哪些数在该出现的时候还没出现，然后就可以 DP 计数了。

$K \neq 0$ 几乎完全一样，只是每次要额外枚举前缀的真实 MEX 是多少。

转移细节留给读者思考。

CF1750F Majority

对于一个 01 串 s，你每次可以选择 l, r，满足 $s_l = s_r = 1$，且 $[l, r]$ 中 1 的个数大于等于 0 的个数，然后把 $s[l, r]$ 全都赋为 1。

给定 n, mod，求有多少个长度为 n 的 01 串，经过若干次操作后能变成全 1 串，模 mod。

$n \leq 5000$

CF1750F Majority

显然我们只需要考虑 $s_1 = s_n = 1$ 的串，不然肯定不合法。

对于一个固定的 01 串显然可以贪心，能操作就操作。

贪心结束时，s 被划分为若干个连续段，满足每个 0 段长度都比相邻的两个 1 段长度加起来更长。不难发现这是无法操作的充要条件。

显然不同的 1 段是独立操作的，因此我们可以对这东西 dp 。

CF1750F Majority

设 $dp_{i,j}$ 表示长度为 i 的 01 串，满足 $s_1 = s_i = 1$，贪心结束时最左边的连续段长度为 j，的方案数。

直接转移，$dp_{i,j} = \sum_{k>j+l} dp_{i-j-k,l}$。然后发现这东西可以前缀和优化两次，就做完了。

CF1268E Happy Cactus

给定一个带边权的无向仙人掌，对于每个 x 求出有多少个 y 使得存在一条从 x 到 y 的边权单调递增的路径。保证边权两两不同。

$n, m \leq 5 \times 10^5$

CF1268E Happy Cactus

对于树的情况，可以按边权从大到小往图中加边，同时维护答案 dp_x 。因为加入一条边 (x, y) 时 x, y 并不联通，所以只需要把答案相加即可。

变成仙人掌之后加入 (x, y) 时 x, y 可能已经存在唯一一条路径了。此时直接相加可能会算重。

不过注意到算重当且仅当 $x \to y$ 的路径上的边权是先增后降的，设最大的边是 e ，那么恰好是经过 e 能到达的点会被算重。所以除了维护 dp_x 还维护 f_e 即可。转移就是 $dp'_x = dp'_y = f_{(x,y)} = dp_x + dp_y - f_e$ 。

CF1442D Sum

给定 n 个单调不降的数组 $a_{i,1}, \cdots, a_{i,t_i}$，你可以从每个数组中取走一个前缀，使得最后一共取了 K 个数。最大化取走的 K 个数之和。

$n, K \leq 3000, \sum t_i \leq 10^6$

CF1442D Sum

由于元素单调不减，可以发现除了一个数组以外其他都是整个选或整个不选。

枚举哪个数组选了一部分，然后相当于要求删掉一个元素的背包，用经典的分治背包算法即可解决。

CF1456E XOR-ranges

给定数组 c ，定义一个数的代价是 $p(x) = \sum_{i \in x} c_i$ ，其中 $i \in x$ 表示 x 二进制第 i 位是 1 。

给定 n 个限制 $[l_i, r_i]$ ，你需要构造一个数组 a ，满足 $a_i \in [l_i, r_i]$ ，且最小化 $\sum_{i=1}^{n-1} p(a_i \oplus a_{i+1})$ 。

$n \leq 50, l_i, r_i \leq 2^{50}$

CF1456E XOR-ranges

先把一个区间拆成 log 个区间，每个区间都是固定前缀，后缀任意。

如果每个数都已经确定了要选哪个区间，那么每一位就已经独立了，代价就是相邻的这一位不同的数的个数。

从低位往高位做，那么一个数一旦确定了自己选哪个区间，在考虑更高位时，它两边的数就独立了。

所以可以区间 dp：设 $dp_{i,l,r,x,y}$ 表示考虑到了第 i 位，(l,r) 中的数还没有确定区间，l,r 分别确定了区间 x,y，更高位的最小代价。转移就枚举某一个确定了区间的数，在这个位置把 $[l,r]$ 劈成两半即可。

CF1476F Lanterns

有 n 个路灯，第 i 个路灯的位置是 i，照明强度是 p_i。

每个路灯都可以选择往前照或往后照，即覆盖 $[i-p_i, i-1]$ 或 $[i+1, i+p_i]$。注意一个路灯不能覆盖自己。

判断是否存在一种方案使得每个路灯都被覆盖，并输出方案。

$n \leq 3 \times 10^5, 0 \leq p_i \leq n$

CF1476F Lanterns

暴力 dp 是直接设 $dp_{i,j}$ 表示考虑了前 i 个路灯，最早的还没被覆盖的路灯是 j，那么往后最远能覆盖到多少。

虽然暴力，但这东西可以用数据结构维护，判断是否有解还是简单的。但是输出方案就有点麻烦了。

CF1476F Lanterns

先来手玩一下。第一个路灯显然会往后照，那么只要确定了某个路灯 j 满足 $j - p_j \le 1$ 且 j 往后找，那么 $[2, \max(j-1, 1+p_1)]$ 里的路灯就都可以无脑往前照了。而被 $[2, \max(j-1, 1+p_1)]$ 的路灯覆盖到的路灯也可以往前照，等等等等，直到推不动为止。

所以我们可以把方案分成若干段，每段里只有一个路灯往前照，其他路灯都往后照。

CF1476F Lanterns

实际上不需要把段分的这么明确。直接设 dp_i 表示前 i 个路灯都被覆盖了，那么可以往后最远覆盖多少。

转移有两种。第一种是贪心往后转移，即当 $dp_i \geq i+1$ 时可以把 dp_{i+1} 与 $\max(dp_i, i+1+p_{i+1})$ 取 max 。

第二种是另起新的一段，让 i 成为这一段里唯一一个向前照的路灯。此时可以从任意一个满足 $dp_j+1 \geq i-p_i$ 的 j 转移过来。

因为我们不是很关心 dp_j 的具体值——上一段本就不应该对这一段产生任何影响——所以直接选取合法的最小的 j 即可。