AT1980 [AGC001B] Mysterious Light
AT1981 [AGC001C] Shorten Diameter
AT1982 [AGC001D] Arrays and Palindrome
AT2002 [AGC003B] Simplified mahjong
AT2044 [AGC004D] Teleporter
AT2045 [AGC004E] Salvage Robots
AT2046 [AGC004F] Namori
AT2061 [AGC005C] Tree Restoring
AT2062 [AGC005D] ~K Perm Counting
AT2064 [AGC005F] Many Easy Problems
AT2063 [AGC005E] Sugigma: The Showdown
AT2163 [AGC006B] Median Pyramid Easy
AT2164 [AGC006C] Rabbit Exercise
AT2165 [AGC006D] Median Pyramid Hard
AT2166 [AGC006E] Rotate 3x3
AT2167 [AGC006F] Blackout
AT2169 [AGC007B] Construct Sequences
AT2171 [AGC007D] Shik and Game
AT2172 [AGC007E] Shik and Travel
AT2173 [AGC007F] Shik and Copying String
AT2264 [AGC008B] Contiguous Repainting
AT2265 [AGC008C] Tetromino Tiling
AT2266 [AGC008D] K-th K
AT2267 [AGC008E] Next or Nextnext
AT2268 [AGC008F] Black Radius
AT2291 [AGC009B] Tournament
AT2292 [AGC009C] Division into Two
AT2293 [AGC009D] Uninity
AT2294 [AGC009E] Eternal Average
AT2305 [AGC010D] Decrementing
AT2306 [AGC010E] Rearranging
AT2307 [AGC010F] Tree Game
AT2339 [AGC011C] Squared Graph
AT2340 [AGC011D] Half Reflector
AT2341 [AGC011E] Increasing Numbers
AT2342 [AGC011F] Train Service Planning
AT2364 [AGC012D] Colorful Balls
AT2365 [AGC012E] Camel and Oases
AT2366 [AGC012F] Prefix Median
AT2368 [AGC013B] Hamiltonish Path
AT2369 [AGC013C] Ants on a Circle
AT2370 [AGC013D] Piling Up
AT2371 [AGC013E] Placing Squares
AT2372 [AGC013F] Two Faced Cards
AT2376 [AGC014D] Black and White Tree
AT2377 [AGC014E] Blue and Red Tree
AT2378 [AGC014F] Strange Sorting
AT2381 [AGC015C] Nuske vs Phantom Thnook
AT2382 [AGC015D] A or...or B Problem
AT2383 [AGC015E] Mr.Aoki Incubator
AT2384 [AGC015F] Kenus the Ancient Greek
AT2385 [AGC016A] Shrinking
AT2386 [AGC016B] Colorful Hats
AT2387 [AGC016C] +/- Rectangle
AT2388 [AGC016D] XOR Replace
AT2389 [AGC016E] Poor Turkeys
AT2665 [AGC017B] Moderate Differences
AT5620 [AGC039F] Min Product Sum
AT2666 [AGC017C] Snuke and Spells
AT2667 [AGC017D] Game on Tree
AT2668 [AGC017E] Jigsaw
AT2669 [AGC017F] Zigzag
AT2670 [AGC018A] Getting Difference
AT2671 [AGC018B] Sports Festival
AT2672 [AGC018C] Coins
AT2673 [AGC018D] Tree and Hamilton Path
AT2674 [AGC018E] Sightseeing Plan
AT2701 [AGC019B] Reverse and Compare
AT2702 [AGC019C] Fountain Walk
AT2703 [AGC019D] Shift and Flip
AT2704 [AGC019E] Shuffle and Swap
AT2705 [AGC019F] Yes or No
AT3856 [AGC020B] Ice Rink Game
AT3858 [AGC020D] Min Max Repetition
AT3860 [AGC020F] Arcs on a Circle
AT3868 [AGC021B] Holes
AT3869 [AGC021C] Tiling
AT3871 [AGC021E] Ball Eat Chameleons
AT3947 [AGC022B] GCD Sequence
AT3948 [AGC022C] Remainder Game
AT3950 [AGC022E] Median Replace
AT3951 [AGC022F] Checkers
AT3953 [AGC023B] Find Symmetries
AT3955 [AGC023D] Go Home

AT1980 [AGC001B] Mysterious Light

第一次反射完之后就在一个平行四边形里面了。然后记 $f(n,m)$ 表示在一个平行四边形里面反射的答案，那么可以递归到 $\lfloor \frac{m}{n} \rfloor \times 2n + f(m \bmod n, n)$ 求解，根据辗转相除法复杂度就是一只 $\log$

AT1981 [AGC001C] Shorten Diameter

显然直径一定有一个中点(k 为偶数)或一条中间边(k 为奇数)，然后枚举这个东西 dfs 一遍就做完了。

$\mathcal{O}(n^2)$ 非常好些。好像可以淀粉树做到 $\mathcal{O}(n \log n)$ 但是每什么用。

AT1982 [AGC001D] Arrays and Palindrome

首先考虑 SPJ 怎么写，对于一个回文 $[l,r]$ 就把 $l \leftrightarrow r, l+1 \leftrightarrow r-1, \ldots$ 全部连上边，最后如果图联通就是合法的。

那么 a 能连的边数是 $\frac{n-a_i\text{为奇数的个数}}{2}$，而 b 最多能连 $\frac{n}{2}$ 条边，所以如果奇数的个数大于 2 就直接不行了。

考虑 $m=1$ 的构造，直接 $b_1=1, b_2=a_1-1$ 就可以了，不难验证正确。

然后把 a_i 的奇数放在开头结尾，再让开头减去 1，结尾 $+1$，中间不变，发现这样构造恰好是中间的边错开，而前后两个也是错开的，所以是对的。

AT2002 [AGC003B] Simplified mahjong

从小到大枚举 i，先让 i 自己配对，再让 i 和 $i+1$ 配对。可以证明是最优的。

AT2044 [AGC004D] Teleporter

被翻译演了一波/jy

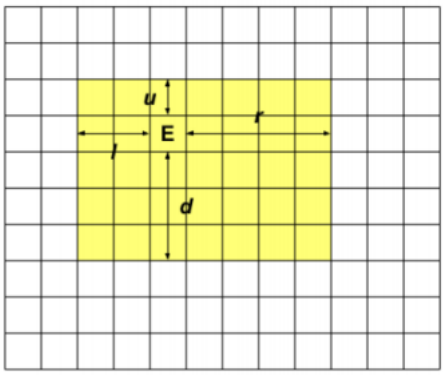
有个很显然的想法是让 1 连向自己，剩下一棵内向树。

然后其他的操作一波使得最大深度不超过 k，直接 dfs 贪心就好了。

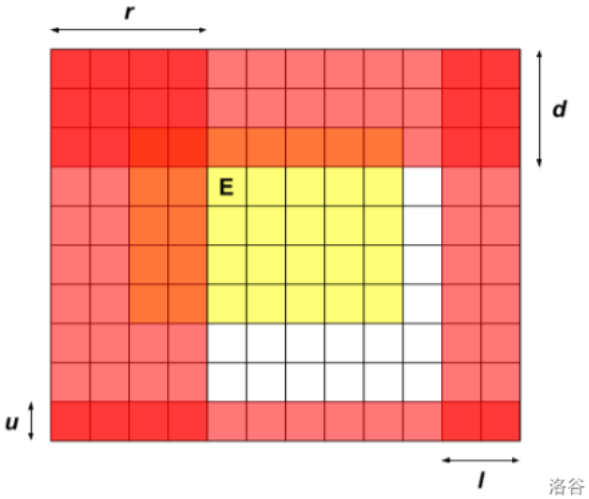
AT2045 [AGC004E] Salvage Robots

牛逼的 dp 题，考虑让起点带着一个框移动，出了框的机器人就gg了。

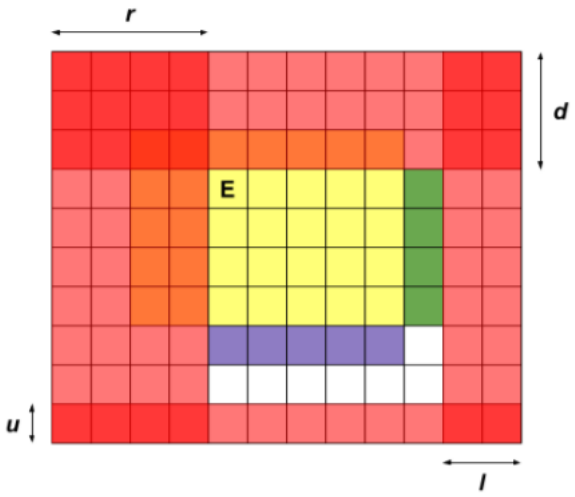
然后如果现在出口最多往上走了 u 步，往下走了 d 步，往左走了 l 步，往右走了 r 步，那么发现在黄色矩形内乱走不会出现新的伤亡，因此整个矩形是可以乱走的。



然后边上一圈红色的如果没有被就gg了。



然后考虑扩展这个矩形。显然向右扩展的贡献就是绿色部分，向下扩展的贡献就是紫色部分。上和左同理。



洛谷

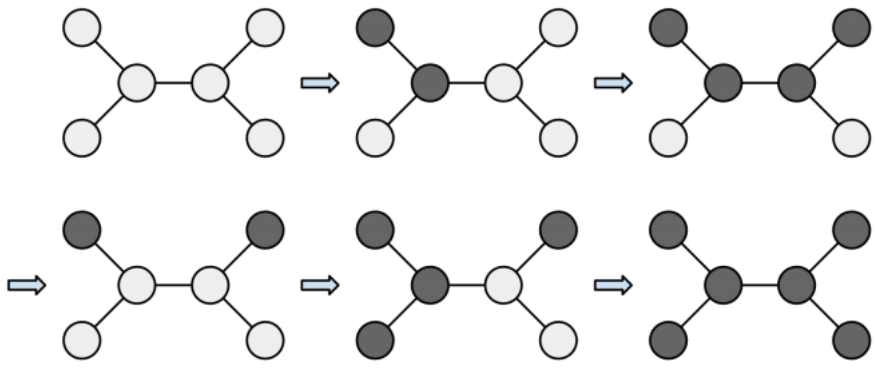
然后写个 dp 就好了。可能空间比较紧张可以用 `short`。

AT2046 [AGC004F] Namori

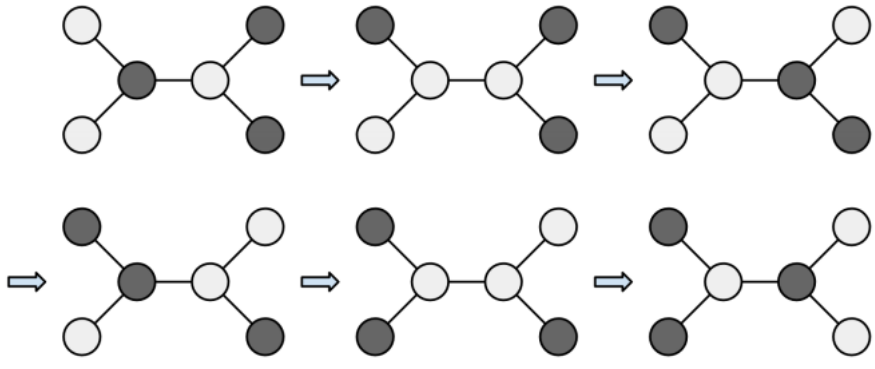
神仙思维题。

首先 n 肯定是偶数，奇数毛估估就不可能。

先来考虑树□，这个同色翻转就非常麻烦，考虑先二分图染色，然后同色一定转化成颜色不同，一次操作就是反转两个相邻节点的颜色，就像下面这张图：



Before the transformation



After the transformation

不难发现目标就是要黑白反色。那么有个条件是黑点 = 白点，然后让黑点为 -1 白点是 1，一棵子树 u 那么 $u \to fa$ 这条边交换次数至少为 u 子树内部权值之和，记为 a_i 的绝对值。可以证明这个下界是能被构造的。

然后是基环树，需要分类讨论。

- 如果是偶环，那么二分图的结论不变，设环上有一条边 $A \to B$ 且根据 dfs 知道 B 一定是 A 的祖先，然后如果 $A \to B$ 这条非树边转移了 x 个黑点那么 $B \to A$ 这条链上都会少 x，然后就变成了一个绝对值的式子要求最小值：

$$\sum |a_i - x| + |x|$$

- 如果是奇环，那么在环边上操作一次能够让黑点增加 2 或减少 2，然后这条边上的操作次数就确定了，再跑树即可。

AT2061 [AGC005C] Tree Restoring

首先找到最大的 a_i 就是直径，那么 a_i 实际上就是 i到两个端点距离的较大值，显然最小是 $\lceil \frac{mx}{2} \rceil$。然后如果直径是奇数那么最小的点有且仅有两个，否则只有一个中心点也就是只有一个。然后大于 mn 的点都至少出现两次。然后剩下的点直接挂到直径上总是有方法的。

AT2062 [AGC005D] ~K Perm Counting

首先这个东西看上去不太好做，而且 $n \leq 2000$ 看上去 $\mathcal{O}(n^2)$ 可以过，所以容斥，然后画一下图，发现就相当于 $2k$ 条链□，每条链上不能选相邻的，问钦定 i 条的方案数。然后对于一条长度为 $m-1$ 有 m 个点的链，选 i 条就是 $\binom{m-i}{i}$ ，就相当与再 m 个点选 i 个『黑白』剩下的是白点，然后所有的卷起来就是答案。

最后容斥一波就好了。

AT2064 [AGC005F] Many Easy Problems

考虑统计每一个点的贡献，记与 u 连接的连通块大小为 sz_v，那么 u 对 i 的贡献就是：

$$\binom{n}{i}-\sum_{(u,v)\in E}\binom{sz_v}{i}$$

意义显然，就是只要不要全部选在一棵子树内那么就会对答案造成 1 的贡献，然后就可以做到 $\mathcal{O}(n^2)$。然后考虑优化到能过的。复杂度主要是再后面，不妨记 cnt_i 表示为 sz_v 为 i 的个数，那么就显然有：

$$ans_k=n\binom{n}{i}-\sum cnt_i\binom{i}{k}=n\binom{n}{i}-\frac{1}{k!}\sum\frac{cnt_i i!}{(i-k)!}$$

不妨记 $a_{n-i}=cnt_i i!,b_i=\frac{1}{i!}$，然后卷积的形式就很显然了：

$$\sum a_{n-i}b_{i-k}$$

AT2063 [AGC005E] Sugigma: The Showdown

首先考虑无限循环，如果红树有一条 $u\to v$ 并且蓝树 $u\to v$ 的距离大于等于 3，那么先手到 u 或 v 游戏就All last了。

然后不考虑这样的边，那么先手每一步的距离在后手的树上就是 ≤ 2 的。记一个点在红树上的深度为 $depr_u$，在蓝树上的深度为 $depb_u$，然后每一步就是 $depr_u<depb_u$。因为在这样的情况下，一定不存在先手绕过后手的情况，否则一定会被抓。所以先手就往子树内走，后手去追他。

AT2163 [AGC006B] Median Pyramid Easy

可以证明形如 $\leq x\ x\ \geq x$ 的形式，到下一层还是满足这样的形式。

然后直接就中间 $x-1,x,x+1$ 就好了。

AT2164 [AGC006C] Rabbit Exercise

首先不妨假设 $i,i+1,i-1$ 的期望位置分别是 x,y,z，那么 $x'=\frac{2y-x+2z-x}{2}=y+z-x$

也就是维护一个数组 E，然后 a 就相当于把 $E_a\leftarrow E_{a+1}+E_{a-1}-E_a$，如果您参加过 noip2021 的话就相当于交换两个差分数组。

然后就直接维护这个置换，然后在一个环上向前走 k 步就好了，然后非常好写。

AT2165 [AGC006D] Median Pyramid Hard

B 题的 SPJ 是紫的/jy

首先让 $\geq x$ 的变成 1 否则是 0 然后跑这个中位数得到最小的，然后找到最小的 x 满足最后剩下的数是 1 就是答案。

然后就变成 01 序列这个问题就好做一些了，然后发现如果出现了连续相同的数就一直不会变了。所以连续 ≥ 2 的把序列分成了若干个部分，中间夹着一些 010101 形式，每次这样的形式会翻转，然后长度减二。然后推一推就可以得到最上面是什么了。可能有一些 border case 需要特判。

AT2166 [AGC006E] Rotate 3x3

首先把一些特别显然的情况判掉，比如一列中三个不是连续的或者原来在奇数列现在在偶数列。

然后就是大胆猜结论的时间，把奇数列上的数字排到原来的位置，发现次数的奇偶性是一定的，然后偶数列上的数字反者的个数的奇偶性是一定的，如果是奇数显然就不幸。

可以证明满足这样条件肯定可以构造方案反正我不会。

AT2167 [AGC006F] Blackout

看到 AT 的神仙题第一感觉是图论。

然后就相当于如果 $x\to y,y\to z$，那么 $z\to x$。然后每个弱连通块显然是独立的每一个考虑加起来就好了。

讨论一个弱连通块，如果可以分到两个集合且只有第一个集合到第二个集合的边，那么就不可能增加新的边。

否则考虑分成三个集合 $\{0\},\{1\},\{2\}$，那么最后有且仅有三种边 $\{0\}\to\{1\}$，$\{1\}\to\{2\}$，$\{2\}\to\{0\}$。证明可以考虑归纳地来证明，考虑一章这样的图，然后挂一条 $\{0\}\to p$，然后 $\{2\}\to\{0\}\to p$，因此 $p\to\{2\}$，然后 $p\to\{2\}\to\{0\}$，因此 $\{0\}\to p$，加点之后的图还是满的。$p\to\{0\}$ 的情况类似。并且一条链 $u\to v\to w$ 是成立所以是对的。

否则如果分不成就把使得分不成的边删掉强行染色，现在是 $\{0\}\to\{1\}\to\{2\}\to\{0\}$，然后加入一条非法的边必定可以出现 $\{x\}\to\{x\}$ 这样的情况，然后就可以得到变成了完全图。

AT2169 [AGC007B] Construct Sequences

如果要全部相等就直接 $a_i=i,b_i=n-i+1$，然后为了满足第三个性质我们给 a_{p_i} 加上 i 但是此时可能 a 就不递增。给 a,b 都乘上 $n+1$ 然后再执行后面的操作就对了。

AT2171 [AGC007D] Shik and Game

考虑Shik的最优方案一定是先搜集完 j，然后走到 $i>j$，再到 $j+1$，再到 i 搜集完所有 $j+1\sim i$ 的金币，所以有 dp：

$$f_i=\min_{j<i}\{f_j+x_i-x_j+\max(T,2(x_i-x_{j+1}))\}$$

后面的柿子的含义是走回 $j+1$ 如果已经有金币了那么以后都可以吃到金币然后时间就是 $2(x_i-x_{j+1})$，否则要再 $j+1$ 等完 T 然后再不等待地吃 $j+1\sim i$ 所有的金币。

大可以写一个数据结构但是没有必要。不妨令 $g_i=f_i-x_i$ 那么：

$$g_i=\min\{g_j+\max(T,2(x_i-x_{j+1}))\}$$

可以维护一个转折点 l，再 $\le l$ 的位置上是 $2(x_i-x_{j+1})$，所以维护 g_j-2_{j+1} 的最小值，之后看看就是 g_{l+1} 最小直接取 $g_{l+1}+T$ 即可。

AT2172 [AGC007E] Shik and Travel

首先显然是满足二分的，直接枚举一个 up 满足所有的都不能超过。

有一个显然的性质是只有把一个子树所有叶子遍历完才能出去。

再一个节点上有一个 dp 数组 $f_{u,i,j}$ 表示在 u 进入的长度是 i 出去的长度是 j 是否可行。我们发现如果 $a>b,c>d$，如果 $f_{b,d}$ 可行，$f_{a,c}$ 就一定是没用的。

我们可以开一个 `vector` 来维护所有可行的状态，满足第一维递增，根据刚才的结论第二维肯定递减，然后在左子树枚举一个状态，然后在右子树找第一维尽可能长也就是第二维尽可能短的状态，然后合并。右到左同理。

考虑为什么复杂度是正确的，假设左儿子的状态数较少，那么路径只有两类，在左儿子开始的和在左儿子结束的，因此状态数最大是左儿子的两倍，经过一些高妙的分析就可以知道一次check的复杂度是 $\mathcal{O}(n\log n)$。

AT2173 [AGC007F] Shik and Copying String

考虑最优的方案一定是一段段折线：

	1	2	3	4	5	6	7	8	9	10
S0	a	b	c	x	a	x	b	x	c	x
S1	a	b	c	c	a	a	b	b	c	c
S2	a	b	b/c	c	c	a	a	b	b	c
S3	a	b	b/c	b/c	c	a/c	a	a	b	c
S4/T	a	b	b	b	c	c	c	a	b	c
	1	2	3	4	5	6	7	8	9	10

然后实际上贪心地从左往右看，每个字符和谁匹配也是确定的，因此我们只需要维护这一堆折线就好了。

发现新增一条折线，原来折线大于当前位置的拐点都没有用了，然后其它的点向左下移动一格。如果两个起点有间隙还会新增一个拐点。可以用队列维护，向左下移动打一个全局标记即可。

AT2264 [AGC008B] Contiguous Repainting

需要一点点智慧

必然有一段长度为 k 的颜色相同，除了这一段其它颜色任意。

然后枚举这一段就好了。

AT2265 [AGC008C] Tetromino Tiling

发现 T,S,Z 是没用的，合法的只有 II,LL,JJ,O 或 ILJ 。然后 IJL 可以只出现一次因为出现两次可以变成 II,LL,JJ。

然后枚举有没有 IJL 就做完了。

AT2266 [AGC008D] K-th K

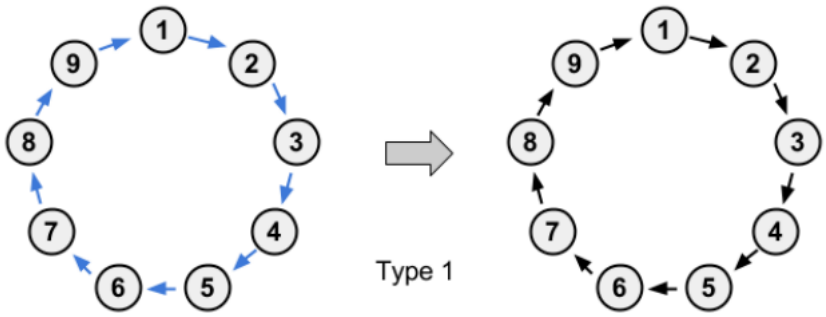
条件就等价于 $[1,p_i-1]$ 有 $i-1$ 个 i，$[p_i+1,n\times n]$ 有 $n-i$ 个 i。那么考虑第一个构造就直接从 p_i 小到大选，把第一个限制满足，然后再从大往小，把第二个限制满足。

只看前面一半有一个很显然的贪心就是尽可能往前面放，然后可以证明对后一半问题也是最优的。

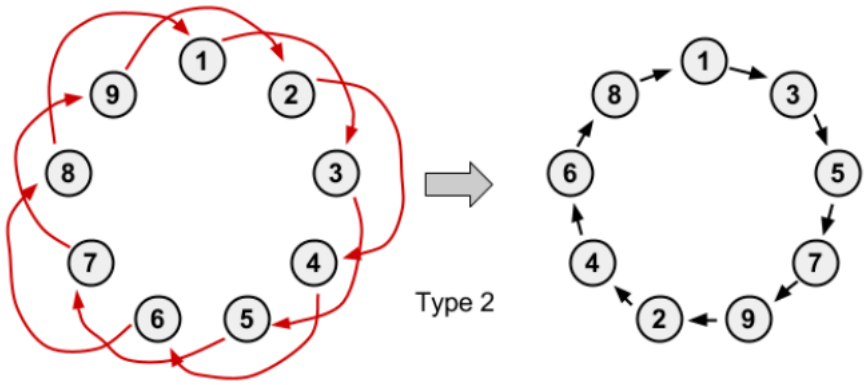
AT2267 [AGC008E] Next or Nextnext

首先如果告诉 p，那么 $i\to p_i$ 形成了若干个环。考虑每个点跳一步或者两步得到题目中的 a_i。分情况讨论：

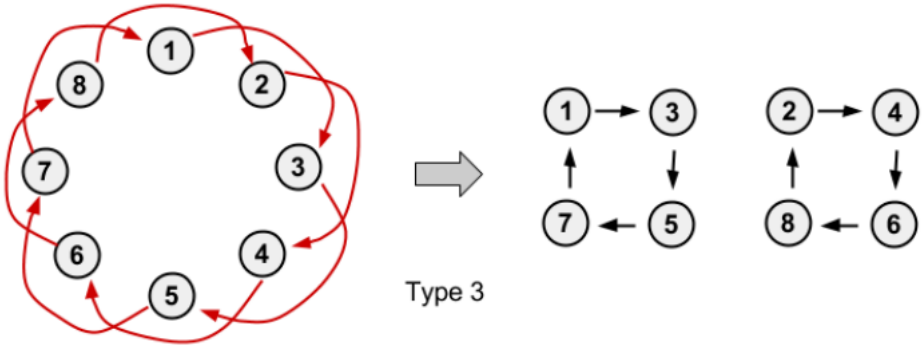
1. 全部取 p_i 那么环不变：



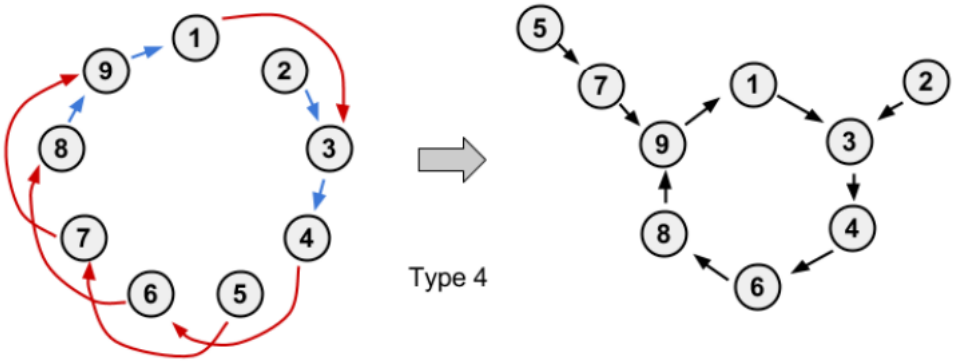
2. 全部取 p_{p_i} 并且环为奇环，那么形成一个同构的环。



3. 全部取 p_{p_i} 并且环为偶环，那么形成两个环。

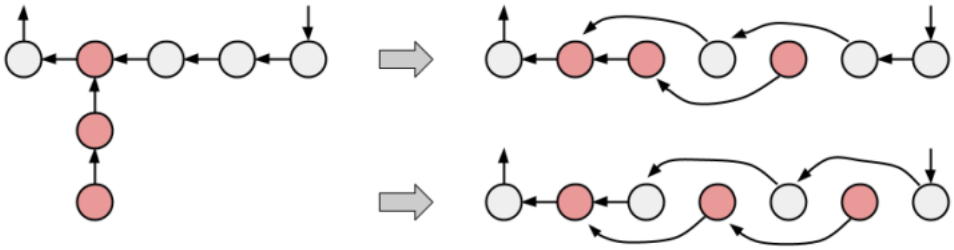


4. 有的跳一步有的跳两步形成一棵基环树，并且环上每个点最多挂一条链。



然后就是计数了，显然环和基环树是独立的可以分别计数。然后每一个长度的环也是独立的。考虑长度 i 的环有 cnt 个那么可以钦定 $2j$ 个是互相匹配得到的，那么这 $2j$ 个的方案数是 $\binom{cnt}{2j}\binom{2j}{j}j!/2^j\times i^j$。就是先选 $2j$ 个，然后算匹配的方案数，然后匹配确定后第二个环还可以转。如果 i 是奇数，剩下的 $cnt-2j$ 个环可能没变，也可能是全部两步，所以再 2^{cnt-2j}。

然后就是基环树的计数。加入比如下面伸出来一条链，像下面这样就会有两种情况。根据边数多少也有可能是 1 或 0。



然后根据乘法原理乘起来就好了。

AT2268 [AGC008F] Black Radius

神仙题。

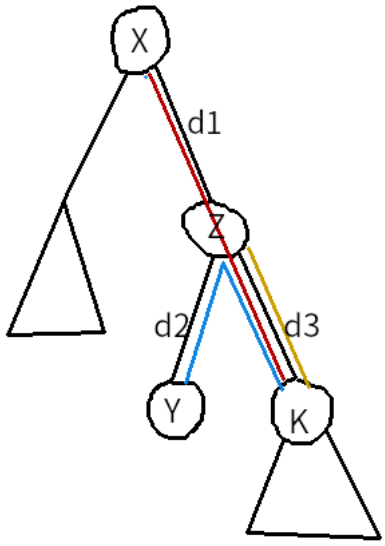
首先考虑全部喜欢的情况。记 $f(x,d)$ 表示距离 $x\le d$ 的集合，强制不考虑 $f(x,d)$ 为全集的情况，最后再加上即可。

在这样的限制下，如果 $f(u,d_1)=f(v,d_2)$，肯定有 $d_1\neq d_2$，因此我们可以在 d 最小的时候统计一次贡献。

首先不能超过树因此 x 的 d 要小于以 x 为根的最长链。然后不能有一个和它相同的集合但是比 x 小，也就是 $f(y,d-1)\neq f(x,d)$，那么 y 子树内部的肯定是一样的，但是外面为了不一样，一个可以走 d 步，一个只能走 $d-2$ 步，那么排除 y 以后最长链肯定要大于 $d-2$，也就是 $d-2<$ 以 x 为根的次长链。可以通过换根 dp 求出。

然后考虑有不喜欢的情况，对于一个不喜欢的点 x，如果 $f(x,d)$ 且 d 是最小并且可以被喜欢的点 y 表示出来，那么一定把 y 所在的子树全部覆盖，也就是给 d 再限制了一个最小值。

证明考虑下面一张图：



如果 $f(x,d_1+d_3)=f(y,d_2+d_3)$，$k$ 以下的点没有被覆盖，那么此时 $f(z,d_3)=f(x,d_1+d_3)$ 与 $f(x,d)$ 是最优矛盾。因此如果 $f(x,d)$ 能被喜欢的点表示出来一定要把一棵有喜欢的点的子树全部覆盖。

于是我们就得到了每个 d 的上界和下界，然后就做完了。

AT2291 [AGC009B] Tournament

就是要让最大深度最小

首先把 i 打败的人挂在 i 的儿子，然后就是 dp_i 表示 i 最后最小的深度，加入一个儿子 x 就是 $dp_i\leftarrow\max(dp_i+1,dp_x+1)$。毛估估就是 dp_x 从小到大转移最优。

AT2292 [AGC009C] Division into Two

不妨令 $A\ge B$，那么如果 $s_{i+2}-s_i<B$ 那么 $i,i+1,i+1$ 一定是无法分配的。于是把 0 特判掉。

然后就有一个 dp，f_i 表示 i 划分到 A 集合的方案数，考虑转移的时候只需要考虑 $s_i-s_j\ge A$ 并且 $[j+1,i-1]$ 中的 s 都符合 B 的限制。不用管端点处是否符合因为隔一个点肯定 $\ge B$。

然后这个就可以直接 `two-pointer` 加前缀和维护了。

AT2293 [AGC009D] Uninity

题目就等价于深度最小淀粉树。

等价于给每一个点一个标号，最后使得最大标号最小，并且两个相同的标号之间一定要有一个标号更大的。

根据一些高妙的分析知道题目是满足最优子结构的，也就是从叶子往上贪心是正确的。

那么考虑节点 u 的选什么，首先根据已经贪心完成的子树我们可以处理出子树内部没有匹配的节点，然后现在的值就不能在这些值里面取。然后还要找一找有没有 k 在两颗子树内都没有匹配那么现在的就至少比 k 大。

然后因为淀粉树深度是 $\log n$ 的因此可以用 `int` 存状态，然后用 `__builtin_` 优化到 $\mathcal{O}(n)$。

AT2294 [AGC009E] Eternal Average

可以看作是一棵 k 叉树，然后每个叶子上有 $0/1$，不妨设 n 个 0 的深度为 x_i，m 个 1 的深度为 y_i，根节点的树就是 $\sum_{i=1}^{m}\left(\frac{1}{k}\right)^{y_i}$。而且我们还知道如果所有都是 1 那么根节点也是 1，所以 $\sum_{i=1}^{n}\left(\frac{1}{k}\right)^{x_i}+\sum_{i=1}^{m}\left(\frac{1}{k}\right)^{y_i}=1$。而且这个是充要的。也就是转化成，有多少 Z 可以被表示成 $\sum_{i=1}^{n}\left(\frac{1}{k}\right)^{x_i}$ 并且 $1-Z$ 可以被表示成 $\sum_{i=1}^{m}\left(\frac{1}{k}\right)^{y_i}$。不妨让 $Z=(0.z_1z_2\dots z_l)_k$，那么考虑还原就是让一位减 1，后一位加 k，因此 $\sum_{i=1}^{l}z_i\equiv n\pmod{k-1}$。并且进位过程中和是减小的，因此 $\sum_{i=1}^{l}z_i\le n$。然后满足这两点就一定能构造出一组 $\{x_i\}$。然后此时 $1-Z$ 的各个数位也确定了，最后一位是 $k-z_l$ 其它是 $k-1-z_i$。然后同样要满足 $1+\sum_{i=1}^{l}k-1-z_i\equiv m\pmod{k-1}$，$1+\sum_{i=1}^{l}k-1-z_i\le m$。

然后就可以 dp 了，直接让 $f_{i,j,0/1}$ 表示前 i 位和为 j 最后一位为 0 或非 0。然后在结尾非 0 出判断这个和 j 是否合法即可。

然后把两个不等式加起来就不难得出 l 最大是 $\frac{n+m-1}{k-1}$ 了。

AT2305 [AGC010D] Decrementing

- 首先如果有 1 那么有奇数个偶数则先手必胜，否则后手必胜。
- 然后如果有奇数个偶数也是先手必胜，因为先手每次只需要取偶数变成奇数，后首不管操作什么都还是奇数个偶数，先手一直是必胜，直到出现 1 然后是先手必胜。
- 如果有偶数个偶数，然后有大于一个奇数，后手必胜，因为不管先手取什么都达到了一个后手的必胜态。
- 否则就是有偶数个偶数和一个奇数，那么先手的最优策略肯定是删去奇数，否则白白给后手送了一个必胜态然后白给。

然后就做完了。第 4 种显然次数是 $\log$ 次暴力就好了。

AT2306 [AGC010E] Rearranging

把所有不互质的数连边那么这些数在排列里相对位置不会改变。

然后对于每一个连通块我们可以贪心地给每条边定向，也就是拉出一棵字典序尽可能小的生成树，局部最优也是全局最优。

然后后手就直接拓扑排序使得最大就好了。

AT2307 [AGC010F] Tree Game

有一个结论是如果你现在在 u，那么不会去 v 如果 v 的值比 u 大，因为如果 $a_v\ge a_u$ 那么后手再回来，先手就白白浪费了一次机会。

就每次往 $a_v<a_u$ 的点走，显然后手不会走回头路，然后就可以直接在树上 处理必胜点/必败点了。

$\mathcal{O}(n^2)$ 非常好写。

AT2339 [AGC011C] Squared Graph

感受到智商被碾压/ll

首先如果 (a,b) 与 (c,d) 联通，就的等价于 $a\to c$ 的奇偶性与 $b\to d$ 的奇偶性相同。然后考虑二分图染色。分成三类连通块。

1. 单点，个数为 a。不会与任何东西连边，然后就是 $(x,*)$ 和 $(*,x)$ 的形式，于是就是 $a\times n+(n-a)\times a$。
2. 二分图，个数为 b。二分图上一种同种颜色的边是偶数，异种颜色的边是奇数，所以钦定两个二分图 A 和 B，那么 $(A黑,B黑)$，$(A白,B白)$ 全部是联通的，$(A黑,B白)$，$(A白,B黑)$ 也是全部联通的。贡献为 $2b^2$
3. 有奇环的图，个数为 c。两个点的路径可奇可偶。只有 3 的贡献显然只有 c^2，然后 3 与 2 可以 $(3,2)$ 或 $(2,3)$ 因此方案数是 $2bc$。

AT2340 [AGC011D] Half Reflector

首先手玩一下发现有两种情况：

- $s_1=A$，那么直接反弹然后结束。
- $s_1=B$，那么可以把所有遍历一遍，最终结果是全部取反然后循环左移。对于第 i 个元素，如果遍历到，上一个肯定是 A 因为你刚刚经过。
- - 如果 $s_i=A$，那么会反弹一遍，经过之后让前一个变成 B，这个还是 A。
  - 如果 $s_i=B$ 就直接取反变成 A

  然后我们就知道每次把 B 变 A，原来 A 前面的变 B。进一步手玩就可以得到上面的结论。

然后如果写得比较好看 $\mathcal{O}(k)$ 就可以通过了。然后发现如果出现了一个 $\dots BABA$ 的后缀就一定不会改变了，而且每一次第二种操作至少让长度增加 1。于是就模拟前 $2n$ 步，如果是偶数就一定是 $BABA\dots BA$，如果是奇数就是 $xBABA\dots BA$，然后每次对 x 取反。

AT2341 [AGC011E] Increasing Numbers

非常巧妙的题目。

首先一个上升数可以用 9 个 $11\dots1$ 构成，如果不足 9 个认为 0 也满足刚才的形式，假如有 k 个上升数，一定能找出 $\{r_i\}_{i=1}^{9k}$ 满足：

$$N=\sum\frac{10^{r_i}-1}{9}$$
$$9N+9k=\sum10^{r_i}$$

因此只需要 $9N+9k$ 的数位和小于等于 $9k$ 即可。可以从 1 开始枚举 k 每次加 9，然后 k 的上界是 L 级别的，然后进位次数也是 L 级别的，然后就是能过的。

AT2342 [AGC011F] Train Service Planning

首先这个题意又臭又长赶紧形式化然后把题面扔进垃圾桶。

相当于有两个数列 $\{p_i\}_{i=1}^n,\{q_i\}_{i=1}^n$ 然后规定 p_0 为上行的火车出发的时间，q_0 为下行的火车到达的时间的相反数，然后 p_i 的前缀和 P_i 就是在 i 出发的等待时间，q_i 的前缀和 Q_i 就是在 i 出发的等待时间。然后就：

$$(P_{i-1}+A_{i-1}+c\times k,P_{i-1}+A_i+c\times k)\cap(-Q_{i-1}+A_n-A_i,-Q_{i-1}+A_n-A_{i-1})=\emptyset$$

然后这个 A_n 非常难受就直接加到 q_0 就好了。然后移项得到：

$$P_{i-1}+Q_{i-1}+c\times k\in[-2A_{i-1},-2A_i]$$

所以就相当于一个数列，第一个数随便取，每次可以加上一个数，有一些限制，要求加的数和最少。

然后就可以 dp 了，如果现在有 x，如果 x 没有被上一个区间包含，就肯定是从上一个区间右端点转移过来，否则就看上上个区间。

因此暂时我们只需要求出 dp_i 表示取第 i 个区间右端点的答案，然后用线段树维护最大的没有包含当前数的区间即可。

最后肯定走到一个左端点所以所有左端点取最小值就是最小等待时间和。

然后无解就是 $2a_i<k$ 肯定会撞车。

AT2364 [AGC012D] Colorful Balls

首先如果两个点可以交换就连边形成了若干个连通块。每一个连通块互相独立所以看单独一个连通块的方案数：

$$\frac{(\sum cnt_i)!}{\prod cnt_i!}$$

其中 cnt_i 表示颜色 i 的个数，而总方案数就是所有乘起来。

但是现在边数的 $\mathcal{O}(n^2)$ 的，考虑优化。

首先同色的连边就只需要连最小值的边显然不影响联通性。

然后异色的边，考虑每一种颜色有一个最小重量，不妨让颜色 1 的最小重量最小，颜色 2 的最小重量次小，假如现在有异色点 x,y 要连边，满足 $w_x+w_y<Y$：

- $c_x\neq 1,c_y\neq 1$，那么有边 $(x,mn_1),(y,mn_1)$
- $c_x=1$，那么有边 $(x,mn_2),(y,mn_1),(mn_1,mn_2)$

因此只需要向最小值和次小值连边，边数是 $\mathcal{O}(n)$ 的，总复杂度也是 $\mathcal{O}(n)$ 的。

AT2365 [AGC012E] Camel and Oases

首先只能跳 $\log_2 v$ 次。

然后考虑 $\lfloor\frac{v}{2^k}\rfloor$ 把序列分割成了若干线段，每个线段内部可以随便乱跳，然后要到其他线段必须要用跳跃。

然后第一层是钦定的，是一个区间 $[l,r]$ 然后考虑之后怎么解决，考虑 dpl_S 表示用 S 这些层向右最多扩展到哪里，dpr_S 同理。转移直接从 dpl_S+1 往右跳。然后一段区间可行就等价于：

$$[1,dpl_S]\cup[l,r]\cup[dpr_{U\setminus S},n]=[1,n]$$

然后枚举 S 是 $\mathcal{O}(v)$ 的，乍一看复杂度是 $\mathcal{O}(nv)$ 的，但是如果第一层的线段数大于 $\log_2 v$ 那么之后的容量更小线段更多是不可能走完的。

所以复杂度是 $\mathcal{O}(v\log v)$ 的。

AT2366 [AGC012F] Prefix Median

首先对 a_i 排序，然后有一个结论是 a_i 只能成为 $b_{1\dots\min(i,2\times n-i)}$。换句话说，就是 b_i 的取值只能是 $a_i\dots a_{2\times n-i}$。

但是这个条件还不够，因为你每次加两个数中位数只能移动一位，所以 $i<j$ 并且 $b_i\in(\min(b_j,b_{j+1}),\max(b_j,b_{j+1}))$ 说明 b_j 不是跳到前驱后继是肯定不行。

然后满足以上条件的 $\{b_i\}$ 就一定能被构造出来了。记 $f_{i,l,r}$ 表示到第 i 位有 l 个不同的数比 b_i 小 r 个不同的数比 b_i 大。然后转移的时候先看看左右不同的数有没有增加，然后枚举向左/向右移动到了哪个位置，然后两个位置中间的数就不能再取了。

复杂度 $\mathcal{O}(n^4)$。

AT2368 [AGC013B] Hamiltonish Path

首先我们维护一条路径 $\{a_1,\dots,a_k\}$，然后看 a_k 有没有全部连完，如果有一个点没有再路径上，就把哪个点加入路径。对 a_1 同理。

拿一个双端队列维护一下就好了。

AT2369 [AGC013C] Ants on a Circle

首先蚂蚁撞倒一起可以看作交换编号继续爬。然后就能得到所有最终的位置。

然后发现相对位置不变，因为碰撞之后反弹所以 1 的后继肯定是 2 也就是说我们只需要知道 1 最终去了那里就好了。

然后发现一只蚂蚁逆时针经过 0 然后 1 的排名就会减 1 否则顺时针排名加 1，然后就只需要算有几次顺时针穿过几次逆时针穿过就可以得到 1 最后的排名了。

AT2370 [AGC013D] Piling Up

考虑一个序列 $\{x_0,\dots,x_m\}$ 其中 x_i 表示第 i 次操作后的红色个数，一次操作有几种可能：

1. `RR`，要求 $x_{i-1}\geq 1$，操作后 $x_i\leftarrow x_{i-1}-1$
2. `RB`，要求 $x_{i-1}\geq 1$，操作后 $x_i\leftarrow x_{i-1}$

3. BB，要求 $x_{i-1}<n$，操作后 $x_i\leftarrow x_{i-1}+1$
4. BR，要求 $x_{i-1}<n$，操作后 $x_i\leftarrow x_{i-1}$

然后有一个 Naive 的想法就是 $f_{i,j}$ 表示 $x_i=j$ 的方案数，但是一个序列会被计数多次。

然后用 $g_{i,j}$ 表示 $x_i=j$ 并且到达过最低点的方案数，到达最低点指再 $x_{i-1}=1$ 时进行操作 1 或操作 2，$f_{i,j}$ 表示没有达到过最低点，发现一个合法序列在 $g_{i,j}$ 中被计数一遍。然后就好了。

AT2371 [AGC013E] Placing Squares

首先考虑组合意义，就是先插板，然后每两块板中间可以放一个黑点和一个白点。

然后就可以 dp 了，$dp_{i,0/1/2}$ 表示到了位置 i 已经有 $0/1/2$ 个点的方案数，然后如果 $i\to i+1$ 没有被 ban 掉就可以：

$$\begin{matrix}dp_{i+1,0}\leftarrow dp_{i,0}+dp_{i,2}\\dp_{i+1,1}\leftarrow 2dp_{i,0}+dp_{i,1}+2dp_{i,2}\\dp_{i+1,2}\leftarrow dp_{i,0}+dp_{1,1}+2dp_{i,2}\end{matrix}\Rightarrow\begin{bmatrix}dp_{i+1,0}\\dp_{i+1,1}\\dp_{i+1,2}\end{bmatrix}=\begin{bmatrix}1&0&1\\2&1&2\\1&1&2\end{bmatrix}\times\begin{bmatrix}dp_{i,0}\\dp_{i,1}\\dp_{i,2}\end{bmatrix}$$

然后如果不能插板就是：

$$\begin{matrix}dp_{i+1,0}\leftarrow dp_{i,0}\\dp_{i+1,1}\leftarrow 2dp_{i,0}+dp_{i,1}\\dp_{i+1,2}\leftarrow dp_{i,0}+dp_{1,1}+dp_{i,2}\end{matrix}\Rightarrow\begin{bmatrix}dp_{i+1,0}\\dp_{i+1,1}\\dp_{i+1,2}\end{bmatrix}=\begin{bmatrix}1&0&0\\2&1&0\\1&1&1\end{bmatrix}\times\begin{bmatrix}dp_{i,0}\\dp_{i,1}\\dp_{i,2}\end{bmatrix}$$

然后矩阵快速幂算一下就可以了。

AT2372 [AGC013F] Two Faced Cards

先让 n 加上 1，然后就可以把 X 中的数离散到 $[1,n+1]$。

首先考虑两个数组 A,B 如何判断是否有匹配，当然可以直接排序，但是非常菜。考虑另一种做法，对每个 A_i 就让 $[A_i,n]$ 加上 1，一个 B_i 就让 $[B_i,n]$ 减去 1。然后序列非负就说明有匹配。

然后就先让 A_i 加，然后就再让 C_i 减，如果 $B_i<A_i$ 可以花费一次代价让 $[B_i,A_i)$ 加上一，每次询问就相当于问让 $x=D_i$ or E_i，满足 x 之前 ≥ 0，x 之后 ≥ -1 最少需要多少的代价。

首先从后往前贪心使得每个数都 ≥ -1，然后再从前往后贪心计算每个前缀都 ≥ 0 最少需要多少多少代价。

AT2376 [AGC014D] Black and White Tree

如果这棵树有完美匹配显然是后手必胜。

如果没有就是先手。考虑每次选一个叶子，然后先手选叶子连接的那个点，此时后手必须选叶子。如果没有完美匹配那么最后会剩下点，先手选了就胜利了。

AT2377 [AGC014E] Blue and Red Tree

相当于再给定的 $n-1$ 条路径上每条选一条。

那么考虑一条边被路径覆盖的次数，如果没有一条边被覆盖一次，那么现在就没有边可以删除，就无解。否则把那条只被覆盖一次的边删除，再把覆盖它的路径删除就可以了。

然后用树剖维护 (u,fa_u) 的覆盖次数和覆盖边的编号的异或就好了。

AT2378 [AGC014F] Strange Sorting

就挺神仙的。

首先删去 1，然后去考虑 $[2,N]$ 需要多少次。

假设 $[2,N]$ 需要 T 次。

如果 $T=0$，那么如果 1 在开头就是 0 否则就是 1。

如果 $T\geq 1$，然后就需要考虑在 T 步之后 1 的位置，如果恰好在开有就是 T 否则是 $T+1$。

如何判断 1 是否在开头就比较麻烦。考虑 $T-1$ 步时，只考虑 $[2,N]$ 开头一定不是 2。可以假设是 2，然后再运行一次 2 就不在开头了，显然不大行。设开头元素为 $f>2$。

如果在 $T-1$ 步之后 1 在 f 与 2 之间那么再操作一次 1 就会出现在开头1否则就不会。

然后我们就需要判断 1 的位置是否在 f 与 2 之间。有一个结论是 $1,2,f$ 的『循环顺序』是 $(f,1,2)$ 那么 1 最终一定在 f 和 2 之间。『循环顺序』下 $(a,b,c),(b,c,a),(c,a,b)$ 是等价的。

然后考虑证明循环顺序不变。有一个结论是忽略 1，如果 f 不在最前面就不会是 high 因为如果是 high 那么始终无法变成开头然后就寄了。开始讨论：

- f 开头，那么只有 f 为 high『循环顺序』不变
- 2 开头，那么只有 2 为 high『循环顺序』 不变
- 1 开头
  - f 是第二个，$1,f$ 为 high 而 2 为 low 『循环顺序』不变
  - 2 是第二个，$1,2$ 为 high，f 为 low 『循环顺序』不变
  - 否则 $1,2$ 都是 low 也不变
- 否则 $1,2,f$ 都是 low 不变

然后这样就可以从 $[2,N]$ 推到 $[1,N]$ 了，可以类似地加点从 N 一直推到 1，需要维护一个时间 T_i 和 T_i-1 是第一个元素 f_i。

AT2381 [AGC015C] Nuske vs Phantom Thnook

感觉这玩意儿挺经典的，因为每个连通块都是树，所以就直接点数减去边数就可以了。

直接用三个前缀和分别是点数，横的边，竖的边，然后就可以 $\mathcal{O}(1)$ 查询了。

AT2382 [AGC015D] A or...or B Problem

首先 l 和 r 相同的情况先特判，然后如果 l 和 r 前缀相同的位就不用管了。第一位不相同的，肯定 l 是 0 r 是 1。不妨让 m 表示这一位为 1 ：

$$l = 0\ldots$$
$$r = 1\ldots$$
$$m = 1000$$

大概就是这样。然后考虑 $[l,m)$ 之间互相或最后的结果肯定还是在 $[l,m)$ ，然后 $[m,r]$ 互相找到 r 除第二个 1 然后这个 1 后面不管是什么都是可以或出来的。然后是 $[l,m)$ 与 $[m,r]$ 之间或，首先不会变小，所以至少是 $m|l$，而最大值显然是可以到 $2m-1$ 的。

然后把这些区间并起来就好了。

AT2383 [AGC015E] Mr.Aoki Incubator

挺巧妙的一个题，考虑把一个人看作一个点 (V_i,X_i)，那么 i,j 两点间的斜率 $K_{i,j}$ 的相反数就是相遇时间。

然后考虑对于一个 i，如果选了它，最终能染色的是怎样的。考虑这样如果有 $x>y>i$，那么如果 $K_{i,x}\le 0$，y 一定是能被染色的，因为如果 $K_{i,y}\le 0$ 直接被染色，否则可以间接被 x 染色。因此能被染色的一定是一段连续的区间，就可以 dp 了。

AT2384 [AGC015F] Kenus the Ancient Greek

众所周知 gcd 的复杂度是 log 的所以第一问答案应该不会很大。不妨令 $x<y$，满足 $F(i,j)=k, i<j$ 的 (i,j) 称为good pair of k。那么最小的good pair of 1就是 $(1,2)$ 最小的good pair of 2就是 $(2,3)$，最小的good pair of k就是 (f_k,f_{k+1})，其中 f_i 表示 $f_0=f_1=1$ 的斐波那契数列的第 i 项。

但是good pair of k的数量还是很大，但是发现很多good pair of k操作一次之后就会变得相同。对于每类变得相同的，我们不妨只考虑最小的 。我们把 $i<j\le 2i$ 的 (i,j) 称为excellent pair of k。然后 $(i,j+k\times i)$ 都是good pair of k。

然后处理excellent pair of k+1首先每个excellent pair of k都可以扩展，$(i,j)\to(j,i+j)$，然后还会新增一个 (f_k,f_k+f_{k+1}) 的扩展也就是 (f_{k+2},f_k+f_{k+2})。

然后询问直接考虑每个excellent pair的贡献即可。

AT2385 [AGC016A] Shrinking

首先枚举最后剩下那一个字符，然后贪心地模拟就好了。

AT2386 [AGC016B] Colorful Hats

如果颜色总数是 x 种，那么如果一只猫猫有重复就还是 x 种否则是 $x-1$ 种。所以我们求出 $\{a_i\}$ 的最大值 max 和最小值 min，那么有解必然有 $max-min\le 1$。

然后先考虑 $max=min$ 的情况，要么全部是独一无二的，也就是 $min=n-1$，要么全部都是有猫猫和它相同也就是 $2min\le n$。

然后再考虑 $max=min+1$ 的情况，令 cnt 为 $a_i=min$ 的个数，那么有 cnt 种是独一无二的，剩下 $max-cnt$ 种每种至少两遍，也就是说 $cnt<max, 2(max-cnt)\le n-cnt$。

AT2387 [AGC016C] +/- Rectangle

神仙构造。首先如果 $H \bmod h=0, W \bmod w=0$ 肯定无解。

考虑让满足 $i \bmod h=0, j \bmod w=0$ 的 (i,j) 是 $-k\times(wh-1)-1$，否则填 k，那么每个矩形的和就是 -1，然后现在的和是 $-\lfloor\frac{H}{h}\rfloor\times\lfloor\frac{W}{w}\rfloor$，因为不满足一开始的无解条件，所以肯定会至少多出一行/一列，于是只要 $\min(H,W)\times k$ 比 $\lfloor\frac{H}{h}\rfloor\times\lfloor\frac{W}{w}\rfloor$ 大就好了。

AT2388 [AGC016D] XOR Replace

考虑用异或和去替换一个数之后，异或和变成着一个数，所以就相当于交换 a_i 和 a_{n+1}，其中 a_{n+1} 是原来的异或和。然后如果 $\{b_1,\ldots,b_n\}\nsubseteq\{a_1,\ldots,a_{n+1}\}$ 就无解，否则肯定有解。然后就对于 $a_i\ne b_i$，那么就把 a_i 和 b_i 这两个数连起来，如果 a_{n+1} 是孤立点就是 边数加连通块数量否则可以再减 1。

AT2389 [AGC016E] Poor Turkeys

WYHAK很早就过了只能膜拜

考虑一只 wyh 如果有一次操作 wyh,xzm 那么我们为了保护 wyh 不被杀掉我们必须把 xzm 杀掉，然后之前我们就要保护 wyh 和 xzm，然后如果一次操作要把两个保护的人都杀掉就不行。

然后就处理出一个 wyh 对应的要保护的集合 S_{wyh}，然后如果 $S_{wyh}\cap S_{xzm}=\varnothing$ 就说明 wyh 和 xzm 可以一起留下来。

AT2665 [AGC017B] Moderate Differences

考虑枚举 $n-1$ 各种多少个是 $[C,D]$，多少个是 $[-D,-C]$，然后就可以有范围了。假设有 i 个是加的那么范围就是 $[A+i\times C-(n-i-1)\times D, A+i\times D-(n-i-1)\times C]$。判断 B 是否在即可。

AT5620 [AGC039F] Min Product Sum

这玩意当年是一个提高组模拟题/jy

首先考虑给了每行的最小值 $\{x_i\}$ 和每列的最小值 $\{y_i\}$ 怎么做，考虑把 x 和 y 扔到一个序列里面排序，然后从小往大取，假设现在取到了 t 然后之前已经有 i 行 j 列，那么如果 t 是行，那么贡献就是 t^{m-j} 否则是 t^{n-i}。

然后考虑对这个序列 dp，记 $f_{t,i,j}$ 表示处理完了 $\le t$ 的所有数，已经有 i 行 j 列。但是这样无法保证每一行和列的最小值恰好就是我们 dp 得到的东西。

再考虑一个容斥，钦定 c 行 d 列是大于我们的钦定的值，其它的是大于等于，容斥系数为 $(-1)^{c+d}$，那么这样容斥一波得到的结果就是我们希望的每行列的最小值恰好是我们钦定的结果了。然后我们就可以列出转移方程了：

$$f[t][i+a][j] \leftarrow \binom{n-i}{a} \times t^{a(m-j)} \times (k-t+1)^{aj} \times g[i][j]$$
$$f[t][i][j+a] \leftarrow \binom{m-j}{a} \times t^{a(n-i)} \times (k-t+1)^{ai} \times g[i][j]$$
$$f[t][i+a][j] \leftarrow \binom{n-i}{a} \times t^{a(m-j)} \times (k-t)^{aj} \times (-1)^a \times g[i][j]$$
$$f[t][i][j+a] \leftarrow \binom{m-j}{a} \times t^{a(n-i)} \times (k-t)^{ai} \times (-1)^a \times g[i][j]$$

然后直接做就可以了，反正时限是 $6s$。

AT2666 [AGC017C] Snuke and Spells

一根数轴上面有 $0,1,\ldots,n$，一个 a_i 就再 a_i 上多挂一根长度为 1 的绳子，然后向左拉直，然后如果能完全覆盖 $[0,n]$ 说明不需要动可以一次操作完否则就需要那些重复覆盖的绳子移到没有覆盖的地方就好了。维护有多少个 $[i,i+1]$ 被覆盖即可。

AT2667 [AGC017D] Game on Tree

首先是如果可以拆成多棵树的博弈，然后再一个点上面接一个根 sg 值加一，然后 dfs 一遍求出所有的 sg 值即可。

AT2668 [AGC017E] Jigsaw

考虑一个左接口如果 $c=0$ 就是 a 否则是 $-c$，然后一个右接口如果 $d=0$ 就是 $-b$ 否则是 d，然后两个接口值相等就可以插。

然后就相当于有 $2h$ 个点，然后 n 条边，要求分割成若干从正值开始，负值结束的路径。但是一个环是不合法的因为wyh暂时不能自己插自己。因此合法的必要条件是：

- 正值出度 $\ge$ 入度
- 负值入度 $\ge$ 出度
- 每个弱连通块至少有一个点入度不等于出度

然后毛估估也是充分的。

AT2669 [AGC017F] Zigzag

首先是一个 $\mathcal{O}(4^n m)$ 的暴力 dp

然后可以用类似插头 dp 的做法，一次转移一个状态。但是这样除了需要知道轮廓线之外，还需要知道上一条现在再哪个位置。复杂度 $\mathcal{O}(2^n n^2 m)$

但是我们发现下面两条线是完全等价的：

因此只需要保留一条轮廓线即可。

AT2670 [AGC018A] Getting Difference

首先求出 $d=\gcd(a_1,\ldots,a_n), mx=\max(a_1,\ldots,a_n)$ 然后能出现 k 就等价于 $k \bmod d=0, k \le mx$

AT2671 [AGC018B] Sports Festival

虽然是要求最大的最小但是不是二分而是直接贪心。

考虑现在最大值是 mx，那么如果不把当前活动鸽掉那么最大值一定不会比 mx 小所以直接贪心模拟就好了。

AT2672 [AGC018C] Coins

首先有三维很难受，不妨让 $e_i=a_i-c_i, f_i=b_i-c_i$，那么就相当于选 x 个 f_i 和 j 个 e_i 使得和最大。

然后然后假设 i 为 e_i j 为 f_j，如果不交换更优，那么 $e_i+f_j>e_j+f_i$ 即 $e_i-f_i>e_j-f_j$，也就是按 a_i-f_i 降序排序肯定全部选的 e 都在 f 左边。因此只需要用优先队列算出每一个前缀最大的符合条件的 e_i 的和和每一个后缀 f_i 的和，取一个最大的就好了。

好像模拟费用流也可以做但是看起来很麻烦。

AT2673 [AGC018D] Tree and Hamilton Path

首先考虑如果是哈密顿回路怎么做，就像上一题一样一条边的次数的上界是两边大小的较小值的两倍。

然后我们需要去掉一条路径。根据 ARC087D 我们知道需要取到最大值每一条路径不能在重心同一棵子树内，所以肯定是删去一条重心到一个子树的边。如果有两个重心一定是删去中间的边。

AT2674 [AGC018E] Sightseeing Plan

调了半天只是因为把 `y6` 打成 `y5` /tuu

首先记 $F(x,y)$ 表示从 $(0,0)$ 到 (x,y) 的方案数，显然就是 $\binom{x+y}{x}$

然后记 $G(x1,y1,x2,y2)$ 表示从 $(0,0)$ 到矩形 $(x1,y1)-(x2,y2)$ 内部的方案数。这个东西就是对 $F(x,y)$ 求和。然后我们发现 $F(x,y)=\sum_{j\le y}F(x-1,j)$ 也就是前缀和，所以 $G(x1,y1,x2,y2)=\sum_{x1+1\le i\le x2+1}F(i,y2)-F(i,y1-1)$。然后刚刚的柿子行和列显然是等价的因此可以进一步化简到 $F(x2+1,y2)-F(x1,y2)-F(x2+1,y1)+F(x1,y1)$。

然后枚举第二个矩形内部的点显然会超时，我们考虑枚举进入的点是那一个，复杂度是 $\mathcal{O}(n)$ 的。但是我们这样无法区分一条路径休息点是那一个。我们需要对一条路径乘上在第二个矩形中的长度，也就是曼哈顿距离，然后可以在进入时乘上 $x+y$，出去是乘上 $x+y+1$，两者做差，那么一条路径被统计的次数就是在矩形中的点数。

AT2701 [AGC019B] Reverse and Compare

本来yy了一个很菜的方法看有多少个是回文，剩下的不是回文一定翻转是不同的。

但是题解非常高妙，就直接统计 (i,j) 并且 $s_i\neq s_j$ 的数量，因为不相等肯定翻转得到不同，相等的话还不如去 $(i+1,j-1)$。

AT2702 [AGC019C] Fountain Walk

一个比较符合直觉的想法是不会走回头路，只会一直往目标的方向走。

然后发现正常的拐弯是 20，而用喷泉只需要 5π，所以尽可能走喷泉。

最多的喷泉等价于一个最长上升子序列。

如果每一行和列都有喷泉那么需要再多走 5π

AT2703 [AGC019D] Shift and Flip

思维难度小于实现难度。

想法很简单，就是枚举最终匹配的位置，然后处理出每一个右移的距离至少需要再左移多少，因为对于一个需要改动的 i，如果右移的距离小于 r_i 就至少需要左移 l_i，实际上也就是一个前缀，然后就只需要在 r_i-1 上更改再求后缀最值就可以知道要左移多少了。

AT2704 [AGC019E] Shuffle and Swap

首先如果都是 0 那么这个位置就没有意义了。然后考虑 A,B 都有 k 个 1，只有 A 有的位置个数为 $diff$。

我们先不重排 $\{a_i\}$，只重排 $\{b_i\}$，然后在 $a_i\to b_i$ 连单向边，那么现在的图有 $k-diff$ 个入度和出度均为 1 的点，$diff$ 个入度为 1 的点和 $diff$ 个出度为 1 的点，也就是若干个环加上 $diff$ 条链。现在要对 k 条边钦定操作顺序，首先每个连通块是独立的所以我们可以最后乘 $k!$，然后一条链内部的操作顺序是固定的只有 1 种所以要乘上 $\frac{1}{\text{边数}!}$，然后一个环内部的方案数是任意的所以直接乘 1。

然后考虑再计数，首先钦定 $diff$ 个入度为 1 的点和 $diff$ 个出度为 1 的点如何匹配，方案数显然是 $diff!$，然后考虑入度和初度均为 1 的点构成链和环的 $\mathbf{EGF}$：

- 链：一条有 i 个入度和出度为 1 的点的链有 $i+1$ 条边，然后有 $i!$ 种排列方式，所以 $\sum_{i=0}\frac{i!x^i}{i!(i+1)!}=\frac{e^x-1}{x}$
- 环：一个环钦定一个起始位置后有 $(i-1)!$ 种方案，所以 $\sum_{i=0}\frac{(i-1)!x^i}{i!}=-\ln(1-x)$

然后环是无差别的要 $\exp$，链是有差别的所以直接取幂，因此最后的答案就是

$$diff!k!(k-diff)![x^{k-diff}]\frac{1}{1-x}\left(\frac{e^x-1}{x}\right)^{diff}$$

$\mathcal{O}(n\log^2 n)$ 就能过了。

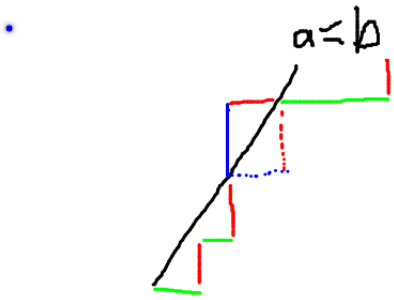
AT2705 [AGC019F] Yes or No

看了 EI 的题解认识到“组合意义天地灭，代数推导保平安”不一定是对的。

考虑你的策略是什么，一个很符合直觉的策略是如果剩下 `YES` 的数量 a 大于剩下 `NO` 的数量 b 就选 `YES`，如果 $a<b$ 就选 `NO`，否则爱选啥选啥。

不妨假设 $n>m$。

然后考虑问题的设置是一张折线图，从 (n,m) 到 $(0,0)$，往下走或者往左走。然后如果在 $a=b$ 下方就预测往左走，在 $a=b$ 上方就预测往下走，在 $a=b$ 上随便。然后考虑不在 $a=b$ 上预测正确的和



发现就是蓝线和绿线的长度和，然后把蓝线翻转，就惊奇地发现就是 n。

所以我们的出结论，所有在 $a \neq b$ 是的预测，个数一定是 n。

所以我们现在只需要统计当 $a = b$ 是预测的期望就好了。对于一条路径，到一次 $a = b$ 预测正确的期望是 $\frac{1}{2}$，所以只需要统计 $\binom{n+m}{m}$ 条路径中一共经过了多少次 $a = b$ 就好了。进一步我们可以转化到每一个 $(0,0) \to (i,i) \to (n,m)$ 的经过次数和。

然后就做完了。

AT3856 [AGC020B] Ice Rink Game

题目看错想了半天。

一个想法是对于一个后缀计算保留人数的区间 $[L,R]$，然后现在操作是 a 求操作前的人数范围，显然就是 $[\lceil \frac{L}{a} \rceil a, (\lfloor \frac{R}{a} \rfloor + 1)a)$

如果区间不存在就无解。

AT3858 [AGC020D] Min Max Repetition

首先 $l = \lceil \frac{\max(a,b)}{\min(a,b)+1} \rceil$ 显然就是最小的连续长度。

$l = 1$ 这样平凡的情况可以先判断掉。

然后手玩一下发现字符串一定是这样的：

$$\overbrace{\boxed{A\dots A}}^{l}B\overbrace{\boxed{A\dots A}}^{l}B\dots B\boxed{A\dots A}\Big|\boxed{B\dots B}A\overbrace{\boxed{B\dots B}}^{l}A\dots A\overbrace{\boxed{B\dots B}}^{l}$$

然后假设前面有 x 段 $\boxed{A\dots A}$，那么后面剩下的 `B` 的个数 $resb = B - x + 1$，然后左半部分至少需要 $(x-1)l + 1$ 个 `A` 所以右边最多还剩 $resa = A - l(x-1) - 1$ 个 `A` 来插在中间。

然后有：

$$resb \leq (resa + 1)l \iff x_{\max} = \left\lfloor \frac{l(A+l) - B - 1}{l^2 - 1} \right\rfloor$$

然后后面 `B` 的个数 $resb$ 就知道了，用掉的 `A` 也就知道了。然后就求出了分界线，直接模拟就好了。

AT3860 [AGC020F] Arcs on a Circle

挺神仙的。

首先固定最长的一条，然后变成一条链。我们发现这个题是连续的变量不方便 dp 所以我们希望变成离散的，具体的方案就是枚举小数部分的相对大小。

一共 $(n-1)!$ 种排列，每种出现的概率是相等的。然后我们就可以离散到 nc 个点，然后就可以 dp 了。$dp[i][S]$ 表示到覆盖了位置 $[1,i]$ 用了 S 中的线段。然后转移是 $\mathcal{O}(1)$ 的。

最后复杂度就是 $\mathcal{O}(n!2^n(nc)^2)$。

但是有一种更高妙的方法就是把圆分成 m 段得到的结果 $f(m)$ 是一个关于 m 的 n 次多项式，希望的值是 $\lim_{m\to+\infty} f(m)/m^n$，然后只需要插 n 个值就好了，可以把 $n!$ 扔掉。

AT3868 [AGC021B] Holes

一个方法是求出凸包，然后凸包上的一个点概率就是两条邻边的垂直平分线的夹角除以 2π

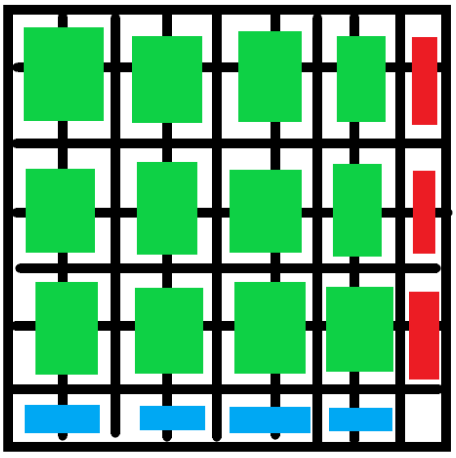
但是写起来很慢，没有用到 $n \leq 100$。比较菜的做法是直接暴力算每一个点的两条邻边，也就是用 `atan2` 计算斜率然后排序，找到使得这个角为凸的两个点，如果找不到答案显然是 0 否则就是 $\pi -$ 这个角 再除 2π。虽然跑得慢但是写得快。

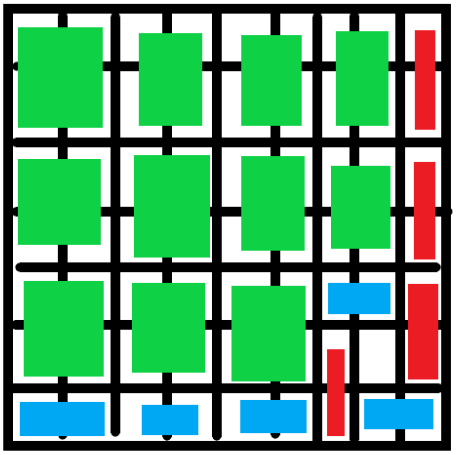
AT3869 [AGC021C] Tiling

首先 n, m 都为偶数的情况就是分成 $\frac{n}{2} \times \frac{m}{2}$ 个小正方形，然后里面放两个 1×2 或 2×1。

然后只有一个是奇数那么就是多的一行或列单独摆。

最后如果都是奇数有两种情况：





洛谷

AT3871 [AGC021E] Ball Eat Chameleons

首先如果 $n > k$ 可以直接输出 0 跑路了。

考虑一只变成红色有两种可能：

- 红色严格大于蓝色
- 相等并且最后一次操作是蓝色

然后枚举红色的个数 r 那么蓝色的个数就是 $b = k - r$

- $r < b$ 就是 0
- $r \geq b + n$ 就是 $\binom{r+b}{b}$ 因为已经严格大于了顺序无关紧要
- $b \leq r < b + n$，也就是有 $r - b$ 个是红色大于蓝色，还有 $n - r + b$ 个是相等。

  首先如果 $r = b$ 最后肯定是蓝色可以让 b 减 1。

  然后你相等感觉肯定是 `RB` 最好的，因此只需要满足有 $n - r + b$ 组 `RB` 就好了。把一个 `RB` 看作一次匹配，那么一个前缀没有匹配的 `B` 的数量不能超过 $b - (n - r + b) = r - n$，否则就没有人和这些匹配然后数量就不达标。所以我们的目标就是统计折线的数量，折线不能超过 $y = x + r - n$，起点 $(0, 0)$ 终点 (r, b)。然后可以把超过线的翻过来得到计数，最后就是：

$$\binom{r+b}{r} - \binom{r+b}{2r-n+1}$$

AT3947 [AGC022B] GCD Sequence

首先让第一个是 2 第二个是 3 就可以满足 $\gcd(a_1, \ldots, a_n) = 1$ 了。但是为了满足 $\gcd(a_i, S - a_i) \neq 1$ 即 $\gcd(a_i, S) \neq 1$ 我们必须保证和是 6 的倍数。一个比较玄学的做法是先放 $n - 1$ 个 2 或 3 的倍数进去，然后再找一个 x 使得和是 6 的倍数并且 $\gcd(x, sum) \neq 1$。可以证明一定找得到这样的 x。

AT3948 [AGC022C] Remainder Game

看到 2^k 这种奇怪的代价就知道肯定可以从高位往低位贪心了。

具体贪心就是强制不选第 i 位，然后 dp 处理一下每个 a_i 再只用 $< i$ 的和强制要选的能不能变成 b_i，如果不能那么 i 就是强制要选的。

然后裸的 dp 是 $\mathcal{O}(n^3)$ 级别的，可以用 `bitset` 优化到 $\mathcal{O}(\frac{n^3}{w})$。

AT3950 [AGC022E] Median Replace

看到这样有 `?` 的题一般是先考虑没有问号怎么判断。

我们维护一个单调栈，从栈底到栈顶为一段 1 再一段 0，考虑现在新增加了一个：

- 加入 0
  - 如果栈为空那么直接加
  - 如果栈顶为 0 能合并就合并否则直接加入
  - 如果栈顶为 1 就直接加入
- 加入 1
  - 如果栈为空直接加
  - 如果栈顶为 0 那么形成 01 能合并成什么只和下一个有关可以直接删掉
  - 否则直接加

那么最后只需要 1 的个数比 0 的个数多就一定可行。发现这个过程中 0 的个数不超过 2 所以如果 1 的个数超过 3 就没有意义了。所以单调栈的状态数是 4×3 的。

AT3951 [AGC022F] Checkers

神仙计数/kk

感觉那些转化成树的看上去都十分意识流还是官方题解的形式化比较简单。

假如我们现在手上有 N 个 N 元组其中第 i 个 N 元组 $(0, \ldots, \overbrace{\boxed{1}}^{\text{第}i\text{位}}, \ldots, 0)$，那么一次操作 A 关于 B 对称就相当于 $2B - A$。然后因为 $x \gg 2^n$ 所以我们认为最后的 N 元组不一样得到的记过就肯定不一样。

然后我们当成是 N 个可重集初始是 $\{2^0\}$，然后一次合并就相当于 $2B\cup -A$，最后得到了一个可重集，再把 $1\sim n$ 填进去。对于一个重复的数次数为 i 就要乘上 $\frac{1}{i!}$ 最后再对答案乘 $n!$。

然后有一些观察：

- $\forall x\in A$，x 可以写成 $\pm 2^k$ 的形式。$\pm 1\in A$ 并且只出现一次。
  - 证明：显然。
- 对于一个 A 设 $|A|=\sum_{x\in A}x$，那么 $|A|=1$
  - 证明：$|2B\cup -A|=2|B|-A=1$
- 若 $\pm 2^i\in A$，那么 $\pm 2^{i-1}\in A$
  - 证明：模拟一下合并过程也是显然的。
- 若 A 合法 $\forall i$，任意改变 2^i 的符号有办法满足 $\sum_{x\in A,|x|\le i}x=1$。
  - 证明：对于一个 $2B\cup -A$，我们可以操作 B 的 2^{i-1} 使得小于等于 2^{i-1} 的部分和为 1，乘 2 之后是 2，然后操作 A 的 2^i 使得变成 1 两个一减就得到了 1。这是必要性。然后这个条件也是充分的大概可以拆分来归纳地证明但是我不太会。

然后就是满足以上条件的 A 的个数，我们有有一个 Naive 的 dp 是用 $dp_{i,j,k}$ 表示用 i 个数，每个数是 $\pm 2^0,\dots,\pm 2^{k-1}$，然后和是 $1+j\times 2^k$。然后考虑现在选了 x 个 2^k 和 y 个 -2^k，能够转移的充分必要条件是 $x-y\equiv j\pmod 2$ 并且 $x+y\ge |j|$，转移十分显然：

$$dp_{i+x+y,\frac{j+x-y}{2},k+1}\leftarrow dp_{i,j,k}$$

发现这个 k 没有 p 用直接扔掉就好了。最终复杂度 $\mathcal{O}(n^4)$。

AT3953 [AGC023B] Find Symmetries

首先有一个 $\mathcal{O}(n^4)$ 的暴力。

发现如果 $(A,0)$ 可行，那么 这一条对角线 都可行。

所以只要枚举有 $(A,0)$ 然后判断即可。

AT3955 [AGC023D] Go Home

需要一些智慧。作为一个正常人类肯定是没有『员工』聪明的，所以正着去考虑他们的决策十分困难。考虑我们倒着来模拟。

首先有一个结论是如果 $X_1<S<X_n$，如果 $P_1\ge P_n$ 那么车会先到 1 然后一路送过去，否则就先到 n。证明大概一归纳证明是这样：

- 如果 $N=2$，那么显然成立
- 如果 $N\ge 3$ 并且 $X_{n-1}<S<X_n$ 那么肯定往左开消掉 $n-1$
- 如果 $S<X_{n-1}$ 那么考虑再到达一之前有没有到第 $n+1$ 幢楼，如果到了就变成情况 2 否则显然没有到 n

对于 $P_1<P_n$ 的情况同理。

因此对于 $P_1\ge P_n$ 的情况，n 里面的人就和 1 里面的人站在了同一条战线上， 我们完全可以强制让 n 里面的人搬家到 1 然后变成一个 $n-1$ 的子问题。

然后模拟一下就可以知道最后的时间了。

AT3956 [AGC023E] Inversions

令 $cnt[k]$ 表示 $\sum[A_i\ge k]-(n-k)$ ，也就是数字 k 还剩下的合法位置数，最后的合法序列数就是 $S=\prod_{1\le i\le n}cnt[i]$。如果有一个 $cnt[i]$ 为 0 就可以直接输出 0 跑路了。

否则我们可以统计每一对 $i<j$ 的贡献。

首先考虑 $A_i\le A_j$ 的的答案，我们希望 i 位置上的数大于 j 位置上的数，因此我们可以先强制让 $A_j\leftarrow A_i$，然后发现 i 比 j 大和 i 比 j 小的方案数是一样多的，并且会发现 $k\in[A_i+1,A_j]$ 的 $cnt[k]$ 都会减 1。因此我们不妨设 $D[k]=\sum_{i=1}^{n}(cnt[i]-1)/cnt[i]$，然后贡献就是 $\frac{1}{2}S\times D[A_j]/D[A_i]$。

我们枚举 j，然后维护一个前缀的 $1/D[A_i]$ 之和，然后查询的时候就直接询问 $[1,A_j]$ 就可以得到 $\sum D[A_j]/D[A_i]$ 了。但是还有一个问题是可能 $D[A_i],D[A_j]$ 的值都是 0 但是其实 $[A_i+1,A_j]$ 的积是不为 0 的。为了解决这个问题，我们可以把 $D[k]$ 写成 $D[k]\times 0^{x[k]}$，然后 $[l+1,r]$ 的积不为 0 就等价于 $x[l]=x[r]$。而 $x[k]$ 显然又是单调不降的，所以我们可以只查询与 $x[A_j]$ 相等的这一部分，$x[A_i]<x[A_j]$ 也就意味着 $[A_i+1,A_j]$ 会出现 0，然后就肯定没有合法方案。

现在来考虑 $A_i>A_j$ 的情况，我们发现我们刚刚的分析有些就没法用了。考虑正难则反，让 $A_i\leftarrow A_j$ 然后跑上面的算法就是 i,j 是顺序对的情况，用 S 减去这个就是逆序对的贡献数了。

AT3962 [AGC024E] Sequence Growing Hard

比较巧妙的计数。

考虑给你一个 S，你现在想要删去一个位置使得字典序变小，你会怎么做？首先，如果有一段是连续的，肯定是操作最后一个来避免重复计数。然后你发现你删去 i 要字典序变小的充要条件就是 $s_i>s_{i+1}$ 或 i 是最后一个位置。

然后这个结论其实挺强的了。现在有一个长度为 i 的串，包含了范围是 $[1,j]$，我们考虑枚举第一个 1 出现的位置 k，再枚举第一个 1 删去的时间 p，然后我们发现前后两段就独立了，并且后面 $i-k$ 个必须再 $p-1$ 步以内删完，而前面没有任何限制。于是列出转移方程：

$$f_{i,j}=\sum_{k=1}^{i}\sum_{p=i-k+1}^{i}f_{k-1,j-1}f_{i-k,j}\binom{p-1}{i-k}=\sum_{k=1}^{i}f_{k-1,j-1}f_{i-k,j}\sum_{p=i-k+1}^{i}\binom{p-1}{i-k}$$

发现后面的一坨只与 i,k 有关，可以直接 $\mathcal{O}(n^3)$ 预处理每一个 i,k 对应的值，然后再暴力转移就好了。

如果没有 1 就是从 $f_{i,j-1}$ 转移。

AT3963 [AGC024F] Simple Subsequence Problem

因为长度很小我们完全可以枚举这个答案然后判断。

考虑一个『子序列自动机』：对于一个状态 $(A|B)$，我们可以转移 B 的第一个 1 到 A 的最后面，然后把这些位置之前删除，也可以把第一个 0 填到最后面，或者直接结束匹配到 $(A|\emptyset)$。

然后这个过程有一个好处是一个串只会对 A 进行 $0/1$ 次的贡献。然后我们就可以 dp 了，大概就是 $dp_{A|B}$，转移的时候就直接暴力模拟刚刚的『子序列自动机』就好了。因为 $A+B\leq n$ 所以状态数是 $\mathcal{O}(n2^n)$ 的。

但是我们现在转移还是需要 $\mathcal{O}(n)$ 的。我们可以预处理出每一个 B 的匹配 $0/1$ 时的后继状态然后就可以做到 $\mathcal{O}(1)$ 转移了。如果用 `__builtin_` 好像也能做只不过可能有一些细节。

AT3966 [AGC025C] Interval Game

首先如果你是 A，那么你的策略很简单，就是走到两个端点较近的一个，显然是不亏的。

那么后手的想法就是让 A 反复横跳，交替跳到 R 最小的和 L 最大的。然后只需要枚举第一步是向左还是向右就可以了。

AT3969 [AGC025F] Addition and Andition

考虑答案有一个上界，考虑每一条边经过了 c_i 次，答案就是 $\sum \min(2,c_i)$。

考虑如何构造上界。首先找到一片叶子，然后找到它的父亲节点 v 和到 v 的边 id，考虑：

- $c_{id}=0$，直接删除没有影响
- $c_{id}=1$，可以直接把这条覆盖的边 (u,x) 变成 (v,x)，肯定可以覆盖 id
- $c_{id}=2$，可以找到两条路径 (u,a) 和 (b,u)，然后公共部分就直接满足要求了，然后剩下可以等价于 (b,a) 或 (a,b) 把这条两条路径合并扔给，然后把剩下的路径扔到 v。

这样操作显然时可以到上界的。

然后因为 $N\leq 2000$ 可以直接暴力模拟。但是模拟可能不是很好写，我们可以用 res_i 表示这条路径时正的还是反的，然后合并路径 (u,a) 和 (u,b) 就等价于说 (u,b) 的正反和 (b,a) 相反，(u,a) 的正反和 (b,a) 相同，可以直接把 (u,a) 改成 (b,a) 然后把 (u,b) 挂到 (b,a) 上。

AT3969 [AGC025F] Addition and Andition

首先考虑模拟这个过程，首先设 x_i 表示 x 从低到高第 i 位的值，y_i 同理，那么一次操作实际上是这样的：

- 从大到小枚举 i
- 如果 $x_i=y_i=1$，那么 $x_i,y_i\leftarrow 0$，$x_{i+1}\leftarrow x_{i+1}+1,y_{i+1}\leftarrow y_{i+1}+1$
- 然后从第 $i+1$ 位开始进位

然后利用一些人类智慧可以发现，操作第 $j<i$ 个位值产生的进位不会波及到 i 位的进位，也就是说，如果我们操作 $i,j,i,j,i,j,\dots$ 和操作 $i,i,i,\dots,j,j,j,\dots$ 是等价的，所以我们可以把上面的算法变成下面的算法：

- 从大到小枚举 i
- 设计数器变量 $z=k$，然后枚举 $j=i,i+1,\dots$
  - 如果 $x_j=y_j=1$ 且 $z>0$ 就让 $x_j,y_j\leftarrow 0$，$x_{j+1}\leftarrow x_{j+1}+1,y_{j+1}\leftarrow y_{j+1}+1,z\leftarrow z-1$
  - 如果 $x_j=2\vee y_j=2$ 就进位
  - 否则就退出。

然后除了 $x_i=y_i=1,x_{i+1}=y_{i+1}=0$ 以外，其它操作都会使 1 的数量减少，所以只要维护下一个有 1 的位值，然后跳过 11 向前移动的过程，复杂度就是 $\mathcal{O}(n)$ 的了。

AT3971 [AGC026B] rng_10s

首先特判几个情况，如果 $A<B$ 或 $D<B$ 就直接有限了。

然后考虑如果不无限，也就是意味着会出现 (C,B) 也就是这次不补下次又不够，也就是 $C<A-Bx+Dy<B\Rightarrow A-B<Bx-Dy<A-C$，然后就看看 $(A-B,A-C)$ 有没有 $\gcd(B,D)$ 的倍数就好了，如果有就是不无限，否则必定无限。

AT3974 [AGC026E] Synchronized Subsequence

考虑把 b 当作 1 a 当作 -1 然后可以分成若干个和为 0 的段，每一个段显然都是互相独立的。考虑一个段 s 使得字典序最大。

- a 开头，那么肯定是每一个 a 匹配的 b 都在后面，所以最后的字符串一定是 $abab\dots$ 的形式，我们希望最长就直接贪心就好了
- b 开头，那么记最大的前缀和为 mx，那么肯定是先 mx 个 b 然后再 mx 个 a，并且所有的 b 都在 a 前面，那么可以发现再 mx 个 a 中间的b 肯定是要选的，进一步地，我们可以推导出对于前 x 个字符使得前缀和为 mx，那么后面 $[x+1,n]$ 是全部要选的。然后模拟

这样可以求出每一段的最大字典序。然后从后往前贪心就可以得到最大的答案了。

AT4376 [AGC027B] Garbage Collector

首先一次捡 $a<b<c<d$，肯定顺序是 d,c,b,a，那么你就是 $d+4(d-c)+9(c-b)+16(b-a)+25a$，化简一下就是 $5d+5c+7b+9a$。然后我们不妨枚举次数 k，那么最远的 $2k$ 个系数是 5，然后分别是 $7,9,11,\dots$。然后这个调和级数复杂度就是 $\mathcal{O}(n\log n)$ 的。

AT4377 [AGC027C] ABland Yard

首先你不会走到一个点没有 A 出或没有 B 出。

然后就很 Naive 了，直接维护出边有没有没有 A 或者没有 B 的，然后删除，然后看一看有没有新的点要删除，如果有就入队。

AT4379 [AGC027E] ABBreviate

首先如果不能操作就直接输出 1 跑路就好了。

否则考虑一个 t 能否被构造。t 中的一个字符对应 s 中的一个区间。然后我们发现把 a 当作 1 把 b 当作 2 那么操作后一个字符串 $\bmod 3$ 下的和是不变的。然后这是一个必要条件，还有一个必要条件就是 s 可以操作或不用操作。因此 s 能变成 a 的必要条件是 $p(s)=p(\mathtt{a})$ 并且『$|s|=1$ 或 s 中有相邻的两个相等』。然后分析一波也是充分的。

然后考虑贪心匹配 t，对于 t 的开头，找到 s 的最短前缀使得和相等并且有相邻两个相等。但是这样可能会剩下一个后缀，首先要满足 $p(s)\equiv 0\pmod 3$，然后可以通过一些证明得到肯定有一种分隔方式使得分成 t。

然后就可以 dp 了，设一个 f_i 表示这个后缀的答案，用 $nx_{i,0/1}$ 表示从当前位置开始找到满足 a/b 要求的最短前缀，然后模拟就好了。因为是贪心所以不会计算重复。

AT4437 [AGC028C] Min Cost Cycle

首先边权是取 $\min$，答案也是取 $\min$，两个是同向的，所以第一个转换是每条边两个随便选，最后的答案显然就是原题的答案。

然后有一个集合 T 是你选的数，然后把一个点分成 4 类：

- `00`，$a_i,b_i\notin T$
- `11`，$a_i,b_i\in T$
- `01`，$b_i\notin T,a_i\in T$
- `10`，$b_i\in T,a_i\notin T$

然后一个结论是 $|T|=n$，并且如果第 i 个是 `*0` 那么下一个就是 `1*`。然后就分类：

- 全是 `01/10`
- 有 `11` 和 `00`

可以发现就这两种情况。第一种是平凡的，考虑第二种。首先把全部 a_i,b_i 丢进去排序，然后把前 n 个当作 T，如果全部是 `01/10` 或者出现过 `11`，那么就是合法的。

否则考虑把 n 换成 $n+1$，如果出现了 `11` 也就是 n 和 $n+1$ 来自不同的 i 或者全部 `01/10` 了就合法了。

然后如果 n 和 $n+1$ 来自同一个 i 那么现在有两种抉择：

- 用 $n+1$ 替换 n
- 用 n 替换 $n-1$

然后就没有然后了。

AT4438 [AGC028D] Chords

首先让 n 乘二。

这个环看上去就很烦，考虑断环为连直接变成链，然后 $[L_1,R_1],[L_2,R_2]$ 有交叉但不包含就说明再环上有交点。

然后考虑一个很经典的计数方法，就是枚举一个联通块统计了多少次。可以发现，钦定 $[i,j]$ 在同一个联通块内部，然后里面的点都不能往外面连边所以内外方案数独立。不妨记 $c(i,j)$ 表示 $[i,j]$ 可以自由支配的点的数量，$g(x)$ 表示 x 个点两两匹配的方案数，也就是 $1\times 3\times\cdots\times(x-1)$。

那么对于 $[i,j]$ 我们希望强制 i,j 在同一个联通块内部的方案数，但是并不好求。考虑容斥，如果 i,j 不在同一个联通块内部，那么枚举联通块 $[i,k](k<j)$，然后剩下的点就可以放任自流，然后就有了 dp：

$$f_{i,j}=g(c(i,j))-\sum_{i\le k<j}f_{i,k}\times g(c(k+1,j))$$

最后一个 $f_{i,j}$ 对答案的贡献是 $f_{i,j}\times g(n-2k-c(i,j))$

AT4439 [AGC028E] High Elements

首先肯定是按位贪心，然后检查后面可行性，能放 1 就放 1。

把原来的前缀最大值叫做旧的，否则就是新的。首先旧的最大值不会消失。然后既然只需要判断可行性，我们有一个结论是可以通过调整一个使得全部都是旧的。证明就考虑如果两个都有新的，那么可以把一个新的放到另一个序列，那么这两个新的就都没了。

然后就稍微好做一点了，考虑现在 x 的上升数量是 cx，y 的上升位数量是 cy，在 $[i+1,n]$ 有 c 个旧的最大值，假设 x 全部是旧的，y 有 k 个旧的和 m 个新的，那么：

$$cx+c-k=cy+k+m\iff 2k+m=c+cx-cy$$

然后就相当于能不能在 $[i+1,n]$ 找到一个递增序列，旧的权值为 2，新的权值为 1，权值和为 $c+cx-cy$。

维护这个东西线段树就可以做了。然后发现如果能凑出权值 w 那么一定能凑出权值 $w-2$ 所以对奇偶分别维护一颗线段树即可。

AT4440 [AGC028F] Reachable Cells

首先有一个非常 Naive 的 $\mathcal{O}(n^4)$，因为太 Naive 了所以对答案一点贡献也没有。

我们首先从下往上，从右往左便利每一个点，然后可以处理出一个点 i,j 到第 k 行最左端的点 $L_{i,j,k}$ 和最右端的点 $R_{i,j,k}$。这个 dp 显然是可以做到 $\mathcal{O}(n^3)$，但是这个不能直接拿来用，因为这一段区间里面不是所有点都能走到的。

然后考虑对于一个点开始走，如果这个点左和上都是障碍，那么这个点对只会的答案就没用过了，可以直接当做障碍。然后我们还能发现，如果一个点还没有被标记成障碍，并且 $y\in[L_{i,j,x},R_{i,j,x}]$，那么 (x,y) 就一定能被访问到。

这样就是对的，直接用 dfs 暴力模拟这个过程就可以了。

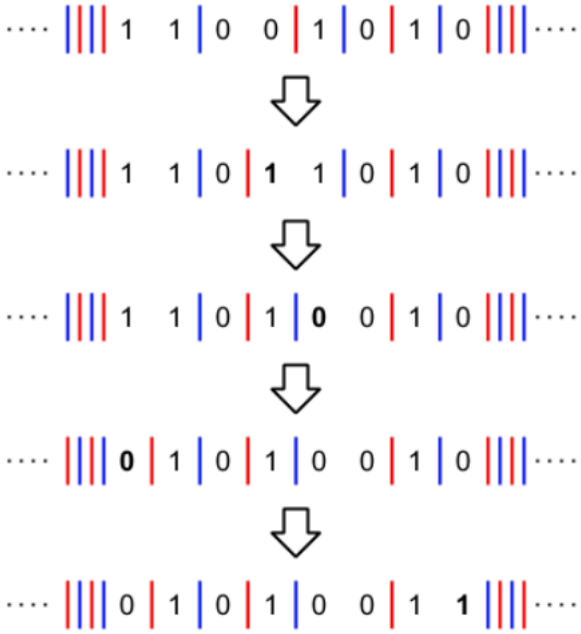
AT4501 [AGC029B] Powers of two

首先，对于一个 x，满足 $x+y=2^i,y\le x$ 的 y 是唯一的。

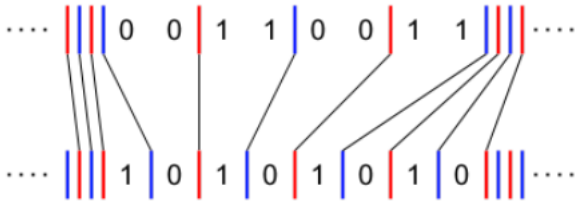
然后从大到小枚举 x，尽可能匹配那一个唯一的 y，随便用一个 `map` 维护一下就好了。

AT4514 [AGC030E] Less than 3

常规做法非常不好做，题解给出了一个绝妙的转化，对于一个位置 $s_i \neq s_{i+1}$，就放一个隔板，每一次可以让隔板移动一位，大概是这样：



然后隔板的匹配确定后次数的下界也就确定了，就是每一个的距离和。



然后这个下界很明显是能取到的。然后只需要枚举 $\mathcal{O}(n)$ 种匹配就对了，复杂度 $\mathcal{O}(n^2)$。

AT4515 [AGC030F] Permutation and Minimum

首先如果 A_{2i-1} 和 A_{2i} 都不是 -1 那么实际上 B_i 就确定了，然后直接扔掉就好了。

然后实际上 $(-1,-1)$ 这样的对值是可以随便换的，所以可以先不考虑他们的顺序最后直接乘数量的阶乘即可。

所以现在的问题变成是 n 个数要匹配，匹配的值是左端点的值，然后如果 i 在一开始的序列中就出现了就 $v_i=1$ 否则就是 0，那么一个匹配如果两端的 v_i 都是 0，那么顺序是无关的，因为我们最后乘阶乘，但是如果有一端是 $v_i=1$ 那么顺序就有关了因为一定出现在特定的位置上。当然不会出现两个 v_i 都是 1。

然后就可以 dp 了。从大到小 dp，记 $f_{i,j,k}$ 表示处理了 $\geq i$ 的数，有 j 个是 $v_x=1$ 的未匹配，k 个 $v_x=0$ 的未匹配，然后可以转移：

- $v_i=1$
  - $f_{i,j+1,k} \leftarrow f_{i+1,j,k}$
  - $f_{i,j,k-1} \leftarrow f_{i+1,j,k}$
- $v_i=0$
  - $f_{i,j,k-1} \leftarrow f_{i+1,j,k}$
  - $f_{i,j-1,k} \leftarrow j \times f_{i+1,j,k}$
  - $f_{i,j,k+1} \leftarrow f_{i+1,j,k}$

然后就做完了，复杂度 $\mathcal{O}(n^3)$。

AT4693 [AGC031C] Differ by 1 Bit

比较有意思的构造。

首先这个 A,B 只需要一个就够了，你可以强制 $A' \leftarrow 0, B' \leftarrow A \oplus B$，构造完最后再异或上 A。下面只考虑 $A=0$ 的构造。

然后有一个很显然的不合法情况是如果 popcount(B) 是偶数那么一定是不合法的。我们考虑证明是奇数一定有构造方案。找到 B 有一位 j 是 1，那么把所有数的第 j 位与第 n 位交换，然后让前 2^{n-1} 个最高位是 0，后 2^{n-1} 个最高位是 1，然后变成了两个子问题，第一段以 1 结尾，第二段以 $1+2^{n-1}$ 开头操作过的 B 结尾。$n=1$ 时显然是合法的，因此构造是正确的。

代码就模拟一下就好了。

AT4694 [AGC031D] A Sequence of Permutations

首先一个排列也可以看做一个函数，$p(i)$ 表示第 i 位上的值，那么函数就可以符合，$p=a\circ b$ 也就是说 $p_i=a_{b_i}$。那么 $f(p,q)\circ p=q \Rightarrow f(p,q)=q\circ p^{-1}$。排列显然是有逆的。于是：

$$a_1 = p$$
$$a_2 = q$$
$$a_3 = q \circ p^{-1}$$
$$a_4 = q \circ p^{-1} \circ q^{-1}$$
$$a_5 = q \circ p^{-1} \circ q^{-1} \circ p \circ q^{-1}$$
$$a_6 = q \circ p^{-1} \circ q^{-1} \circ p \circ q^{-1} \circ q \circ p^{-1} \circ q^{-1}$$

然后可以发现每一步都是：

- p 换成 q，p^{-1} 换成 q^{-1}
- q 换成 $q \circ p^{-1}$，q^{-1} 换成 $p \circ q^{-1}$

然后发现 $L = q \circ p^{-1} \circ q^{-1} \circ p$ 在上面的变换下是不变的，而且经过 6 次操作 $p \to L \circ p \circ L^{-1}$，$q \to L \circ q \circ L^{-1}$，$p^{-1} \to L \circ p^{-1} \circ L^{-1}$，$q^{-1} \to L \circ q^{-1} \circ L^{-1}$。

于是 $a_n = L \circ a_{n-6} \circ L^{-1}$。

快速幂即可 $\mathcal{O}(n \log k)$，找环可以做到 $\mathcal{O}(n)$。

AT4695 [AGC031E] Snuke the Phantom Thief

先考虑一维的情况，枚举一共选的个数 k，那么对 x 排序，那么第 i 个，对限制 t, a, b

- $t = \texttt{L}$，那么如果 $i > b$，则 $x_i > a$
- $t = \texttt{R}$，那么如果 $i \le k - b$，则 $x_i < a$

然后就可以得到第 i 个的取值范围。

两维的情况就跑最大费用最大流即可。

这样不一定第 i 个横坐标一定排在第 i 个，但是跑出来符合条件就一定不会超出限制。

AT4696 [AGC031F] Walk on Graph

首先发现正着走需要记录长度很不方便，可以倒着走，一开始在 t，有一个数字 0，然后如果当前在 u 上数字为 x，经过一条边 (u, v, w) 变成 $2x + w$，问能不能走到 s 数字为 r。不妨记 (u, x) 表示在 u 手上的数字为 x 的状态。

然后一个重要结论是对于 (u, x) 可以 $(u, x) \to (v, 2x + w) \to (u, 4x + 3w) \to \dots$，因为 mod 是奇数所以经过 $\varphi(mod)$ 次之后又会回到 (u, x)，这就形成了一个环。也就是说，这种关系是双向的。因此我们可以对所有可以达到的状态连边，然后看 $(t, 0)$ 与 (s, r) 在不在同一个连通块内即可。

但是状态还是太多。考虑现在在 (u, x) 有两条边权分别为 a, b 的边，那么 $(u, x) \Leftrightarrow (u, 4x + 3a)$，$(u, x) \Leftrightarrow (u, 4x + 3b)$，也就是说我们可以在 u 加上 $3(a - b)$。不妨记这个循环节为 t_u，那么一开始 t_u 就是最大的正整数使得所有与 u 相连的边膜 t_u 同余，并且 t_u 是 mod 的因子。然后对于 (u, v, w)，(u, x) 和 $(u, x + 3t_u)$ 是等价的，$(v, 2x + w)$ 和 $(v, 2x + 6t_u + w)$ 是等价的，因此新的循环节 $t'_v = \gcd(t_v, 2t_u) = \gcd(t_u, t_v)$，不停这样下去，最后所有的 t 应当都是相等的，就是找到最大的 g 使得所有边膜 g 同余，t 就是 $\gcd(3g, mod)$。我们不妨把这个循环节当做 mod。

于是我们可以把边写成 $kg + z$ 的形式。考虑手上还有一个红色的整数是原来整数加 z，即 $(u, x) = (u, x - z)$。然后经过一条边 $kg + z$ 就变成 $(x - z) \times 2 + kg + z + z = 2x + kg$。我们如果有可以把所有的边都减去 z 得到的结果不变。

我们从 (u, X) 开始走，那么我们能走到的点一定满足 $(v, 2^p X + qg)$ 的形式。因为 mod 是 $3g$ 的因数所以 $q = 0, 1, 2$。

然后走一条边 kg 过去再回来就得到了 $(v, 2^{p+2} X + qg)$，本来还有 $3kg$ 膜 mod 就直接没了。因此我们只需要考虑 $p = 0, 1$ 的情况。

所以状态数只剩下 $6n$ 了。然后考虑 (t, z) 与 $(s, r + z)$ 在不在同一个连通块内，首先枚举 p, q 满足 (t, z) 与 $(s, 2^p z + qg)$ 在同一个连通块内。然后因为 2^p 可以任意乘 4 所以相当于说 $2^{p+2k} z = r + z - qg$，可以预处理出 $2^{p+2k} z$ 所有的值然后就可以判断了。

复杂度看上去是 $\mathcal{O}(mod + n\alpha(n))$ 的。

AT4516 [AGC032A] Limited Insertion

倒着模拟。

首先如果 $b_i > i$ 肯定无解。

否则最后一次操作肯定是在最大的 i 满足 $a_i = i$ 插入 i。撤销这次操作，然后继续模拟。

AT4517 [AGC032B] Balanced Neighbors

有实力的洛谷题面是错的，不需要边数最少。

然后可以先构造原图的补图，要求是自己和所有的相连的点加起来和相同。如果是偶数，就直接 $1 \leftrightarrow n, 2 \leftrightarrow n - 1, \dots$，取补图显然联通。如果是奇数，就 $1 \leftrightarrow n - 1, 2 \leftrightarrow n - 2, \dots$，补图满足条件。

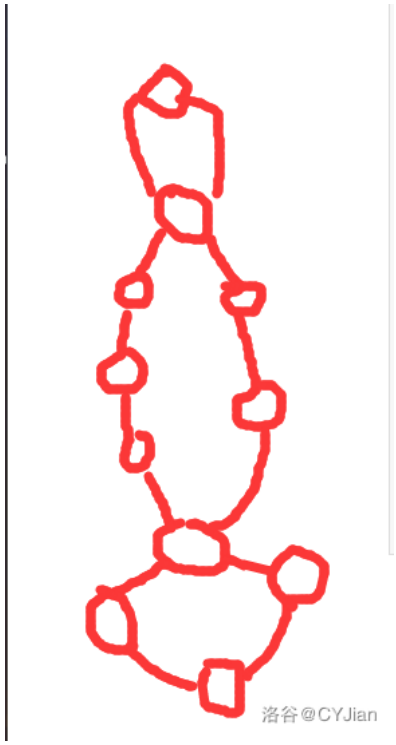
AT4518 [AGC032C] Three Circuits

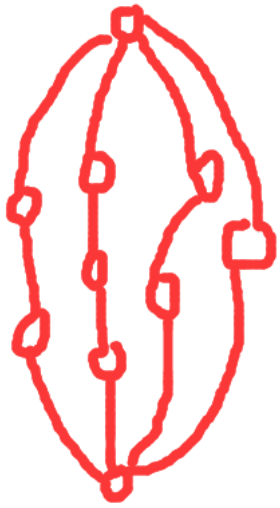
首先分成若干个环肯定所有度数都是偶数。

然后如果有一个点度数大于等于 6 可以前两个环每个使这个度数 -2 最后一个环就把剩下全部跑完。

然后如果所有点度数都小于等于 2 那么只有一个环显然不合法。

然后就是 $2, 4$ 都有，首先如果只有一个 4 是不行的，超过两个 4 必然可行。然后就是有且仅有两个 4，有两种情况：





第一种可行，第二种不可行，写一个 dfs 判断一下即可。

AT4519 [AGC032D] Rotation Sort

首先有一个结论是一个数只会移动一次，移动两次就是白给。

考虑一个钦定一个极长上升子序列（中间不能加数），那么剩下的每个数的方向也就确定了。于是就可以 dp 了，记 dp_i 表示以 i 结尾的最小代价，随便转移一下就好了。

AT4926 [AGC033C] Removing Coins

原题的操作非常麻烦，但是稍微转化一下发现操作 u 就等价于删除除了 u 以外的所有叶子。

那么如果操作的是直径的端点，那么直径点数少 1，否则少 2。

然后就变成一个非常简单的博弈了，边界是 1 个点先手必胜，2 个点后手必胜。

AT4927 [AGC033D] Complexity

比较套路的 dp 优化题。首先有一个状态数 n^4 复杂度 $\mathcal{O}(n^5)$ 的 dp 肯定是过不了的，考虑优化。

首先一个显然的性质是加一行/列凌乱度会上升。然后我们可以知道凌乱度不会超过 $\lceil \log_2 n \rceil + \lceil \log_2 m \rceil$ 的。这样状态数比值域大我们可以考虑交换状态。记 $dp_{i,j,k,x}$ 表示左上角为 (i,j)，左下角为 (k,j)，凌乱度为 x 最远能转移到的列。但是这样还有一个问题是凌乱度恰好为 x 时我们还需要枚举两边的凌乱度才能转移，我们把状态改成凌乱度**不超过** x，这样就不需要枚举两边的具体值，只要两边都不超过 $x-1$，新的就一定不会超过 x，而且还可以滚动数组优化。

转移就分两种情况转移，如果是两块竖的拼起来，就直接跳两遍就好了。如果是两条横的拼起来，就枚举分界点 $i \le w < k$，那么就是 $\min(dp_{i,j,w,x-1}, dp_{w+1,j,k,x-1})$，发现随 x 增大一个增加一个减少可以二分得到分界点然后求出最大值。

复杂度 $\mathcal{O}(n^3 \log^2 n)$

AT4928 [AGC033E] Go around a Circle

首先红和蓝是等价的，不妨设 $S_1 = \texttt{R}$。

- 如果 S 所有都是 R，那么就等价于不能有两个连续的 B，直接 dp 即可。
- 否则至少有一个 B，并且不能出现两个 B 相邻。不妨设 S 中第一段连续的 R 长度为 L，那么发现在环上两个 B 中间的 R 的个数是奇数，不然两边同时是奇数和同时是偶数必然都会出现，那么就会有的位置不能满足 L 然后就不合法。并且每一段的长度都不能超过 $L+1$。而且是奇数之后，行为就固定了，一开始首先走到一个 B，然后走掉一段 B，然后如果接下来 R 的个数是奇数就走到下一个 B 否则就反复横跳。有了这个过程之后我们知道长度也不能超过长度是奇数的极长连续段。于是我们得到了最长的长度 mx，并且一定是奇数，那么我们就相当于把不大于 $(mx+1)/2$ 的放在一个 $n/2$ 的环上的方案数，首先先求出序列，然后一个序列放到环上的方案数还需要乘上第一段的长度，设置成初始值即可。

AT4929 [AGC033F] Adding Edges

神仙题。

考虑如果限制说必须是 a, b, c 的顺序才能加边 (a, c) 怎么做。那么一条边 (u, v) 能够出现当且仅当 $u, v_1, v_2, \ldots, v_k, v$ 在树上是一条链，并且在图上联通，dfs 一遍就能得到一个点能到的位置。

然后转化成不在乎顺序，那么我们需要修一修原题给出的边。更具体地，如果顺次在链上的 (a, b, c)，满足 $(a, b), (a, c) \in G_0$，那么我们可以把 (a, c) 缩短成 (b, c)。我们希望维护这个缩短。

按顺序加入边并缩短，考虑现在的边缩到极短之后是 (u, v)，我们考虑维护 T 以 $1, 2, \ldots, n$ 为根时的有根树 $T_1, T_2, \ldots, T_n$，加边时给 T_u 的 v 子树所有点打上标记 v，表示以后加边 (u, x) 可以缩成 (v, x)。考虑标记冲突的维护，如果 T_a 的 b 上已经有了 c 标记，现在想打上 d 标记，那么这条链的顺序是 a, d, c, b，那么原来的 (a, c) 需要被缩短成 (d, c)，然后以后加入 (a, b) 会缩短到 (c, b) 所以子树内部标记不需要更改。

分析一下复杂度是 $\mathcal{O}(n(n+m))$

AT4991 [AGC034A] Kenken Race

如果出现了 `##` 那么一定不可行。

如果 $c < d$ 那么可以第二个先走肯定可行。

否则需要换位置至少要 `...`

AT4993 [AGC034C] Tests

首先这个 s 显然可二分的，s 越大式子的最大值就越大。

然后其实 $a_i < b_i$，那么 $c_i = l_i$ 否则就是 r_i。

然后就是只有一个值是 $s \bmod x$，剩下的不是 0 就是 x。然后只需要枚举哪一个是 $s \bmod x$ 剩下的贪心就好了。复杂度 $\mathcal{O}(n \log (nx))$

AT4994 [AGC034D] Manhattan Max Matching

有一个比较显然的费用流做法，可惜边数是 n^2 的。

考虑曼哈顿距离
$|x_1 - x_2| + |y_1 - y_2| = max\{x_1 - x_2 + y_1 - y_2, x_1 - x_2 - y_1 + y_2, -x_1 + x_2 + y_1 - y_2, -x_1 + x_2 - y_1 + y_2\}$

可以把两个点分离开
$|x_1 - x_2| + |y_1 - y_2| = max\{(x_1 + y_1) + (-x_2 - y_2), (x_1 - y_1) + (-x_2 + y_2), (-x_1 + y_1) + (x_2 - y_2), (-x_1 - y_1) + (x_2 + y_2)\}$

然后这个式子和我们要求的都是 max，所以我们只需要对对应情况分别连边，然后跑最大费用最大流，因为最大所以肯定选择曼哈顿距离。

然后边数就是 $\mathcal{O}(n)$ 的了。

AT4996 [AGC034F] RNG and XOR

设 $P_i = \frac{A_i}{\sum_{j=0}^{2^n-1} A_j}$，$E_i$ 表示 i 变成 0 的期望次数（和 0 变成 i 显然是相同的），那么有：

$$E_i = \begin{cases} 0 & , \quad i = 0 \\ \sum_{j=0}^{2^n-1} E_j \times P_{i \oplus j} + 1 & , \quad i \neq 0 \end{cases}$$

于是不妨记 $P(x)$ 为 $\{P_i\}$ 的集合幂级数，$E(x)$ 为 $\{E_i\}$ 的集合幂级数，$I(x)$ 为 $\{1, 1, 1, \ldots, 1\}$ 的集合幂级数，乘法运算为 $x^i \times x^j = x^{i \oplus j}$ 那么有：

$$E(x) = E(x) \times P(x) + I(x) + c$$

其中 c 为一修正 $[x^0]E(x)$ 的常数。带入 $x = 1$ 得到 $c = -2^n$。然后对两边同时 FWT：

$$\mathrm{FWT}(E(x)) = \mathrm{FWT}(E(x)) \times \mathrm{FWT}(P(x)) + \mathrm{FWT}(I(x) - 2^n)$$
$$\mathrm{FWT}(E(x))_i = \mathrm{FWT}(I(x) - 2^n)_i / (1 - \mathrm{FWT}(P(x))_i)$$

然后发现 $i \neq 0$ 时 $\mathrm{FWT}(P(x))_i \neq 1$ 所以分母有意义。但是 $i = 0$ 时 $\mathrm{FWT}(P(X))_0 = 1$ 所以我们没有办法得到 $\mathrm{FWT}(E(x))_0$ 具体的值。但是考虑更改 $\mathrm{FWT}(E(x))_0$ 之后所有 $[x^i]E(x)$ 都会改变相同的值，又知道 $[x^0]E(x) = 0$，所以整体偏移就是答案了。

AT5139 [AGC035B] Even Degrees

第一次见挺神仙的第二次见就是套路题了。

首先 m 是奇数无解，否则拉出一棵生成树，非树边随便连，在树上用 $u \leftrightarrow fa_u$ 调整奇偶即可。

AT5141 [AGC035D] Add and Remove

假设我们用天顶星科技使时光倒流，那么考虑每个数对答案的贡献，加入现在要在 a_i 和 a_i+1 中间插入一个数，那么对答案贡献的次数就是 i 和 $i+1$ 贡献的次数之和。初始条件是 a_1 和 a_n 分别贡献 1。

然后就可以 dp 了，记 $dp_{l,r,cl,cr}$ 表示区间 $[l,r]$ 其中 l 的贡献次数是 cl，r 的贡献次数是 cr 不考虑 l,r 的最小贡献，那么有转移：

$$dp_{l,r,cl,cr}=\min_{l<i<r} dp_{l,i,cl,cl+cr}+dp_{i,r,cl+cr,cr}+a_i\times(cl+cr)$$

然后答案就是 $dp_{1,n,1,1}+a_1+a_n$。分析一下复杂度可以知道是 $\mathcal{O}(2^n)$ 的。

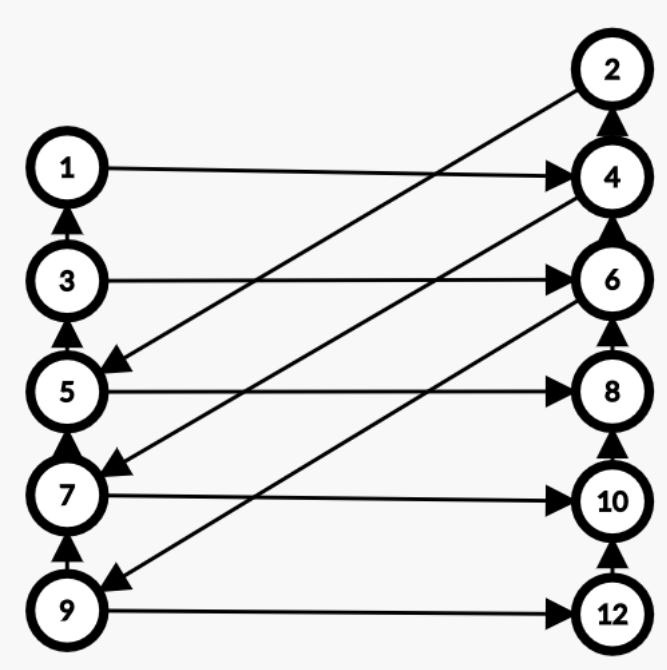
好像爬山也能过但是代码比正解长。

AT5142 [AGC035E] Develop

首先连边 $x\to x-2,x\to x+k$，然后就相当于选出一些不要的数，按照拓扑序删除，如果有环就不合法。

先考虑 k 是偶数的情况，那么奇数和偶数是独立的，可以分开计算然后乘起来。一个限制是不能超过 $\frac{k}{2}$ 个数字同时选中，不然出现了环。然后 $\mathcal{O}(n^2)$ dp 即可。

然后对于 k 是奇数的情况，把奇数放一列，偶数放一列，a 和 $a+k$ 在同一行，大概长这样：



然后发现**最小环**的形式是从一个奇数出发，往上走，往右走，往上走，再回来。那么也就意味着我们选中的点往上-往右-往上的长度不能超过 $k+1$。然后就可以在图上 dp 了，记 $f[i][j][k]$ 表示到 i 这一层，右边的选中的长度为 j，当前路径最长为 k，转移只需要枚举当前层怎么选即可。

AT5143 [AGC035F] Two Histograms

首先有一个结论如果存在 $k_i=j-1,l_j=i$ 那么换成 $k_i=j,l_j=i-1$ 是不变的，并且两个方案都不存在 $k_i=j-1,l_j=i$ 那么如果不一样那么图就一定不同构。

然后我们只需要钦定 i 个拐角存在 $k_i=j-1,l_j=i$，乘上容斥系数 $(-1)^i$ 就做完了。

$$\sum_{i=0}^{\min(n,m)}\binom{n}{i}\binom{m}{i}i!(n+1)^{m-i}(m+1)^{n-i}$$

AT5144 [AGC036A] Triangle

考虑一个点先固定在 $(0,0)$，剩下两个点分别为 $(x_1,y_1),(x_2,y_2)$，那么 $S=|x_1y_2-x_2y_1|$，可以取 $x_1=10^9,y_2=\lceil\frac{S}{x_1}\rceil,x_2=1,y_1=x_1y_2-S$

AT5145 [AGC036B] Do Not Duplicate

考虑第一个是 x，那么下次出现 x 时候会把栈弹空。因此是有循环节的，可以让 k 对循环节取模。

然后剩下的 k 不会很大，就暴力跳直到 $k\le 5$ 然后跑暴力。

AT5146 [AGC036C] GP 2

首先有三个必要条件：

- $\sum_{i=1}^n a_i=3m$
- $\max_{i=1}^n a_i\le 2m$
- $\sum_{i=1}^n a_i \bmod 2\le m$

然后大概可以证明也是充分的。然后就可以做了。首先忽略第二个条件，那么可以枚举奇数的个数 i，然后 $3m-i$ 如果是偶数那么就相当于 $\frac{3m-i}{2}$ 个数分给 n，答案就是 $\binom{n}{i}\binom{(3m-i)/2+n-1}{n-1}$

然后就是违反了第二条的要减掉。因为最多只有一个数违反第二条，所以可以假装是 a_1 然后再乘上 n。可以枚举这个值是多少，剩下的数已经 $\le m-1$ 所以奇数的个数一定满足第三条限制。

AT5147 [AGC036D] Negative Cycle

首先没有负环等价于差分约束系统有解。那么倒过来说，我们只需要确定一组解 $\{x_i\}$，那么就可以知道最多能保留那些边。因为初始的边不能删去，所以 $x_i \ge x_{i+1}$。设 $q_i = x_i - x_{i+1}$，那么一定有 $q_i \ge 0$。

假设保留一条边权为 -1 的边 $i \to j$，也就是 $i < j$，那么 $x_1 - x_j \ge 1$ 即 $q_i + q_{i+1} + \cdots + q_{j-1} \ge 1$

假设保留一条边权为 1 的边 $i \to j$，也即是 $i > j$，那么 $x_j - x_i \le 1$ 即 $q_j + q_{j+1} + \cdots + q_{i-1} \le 1$

因此一条被删去的 $i \to j, i < j$ 的条件是 $q_i + q_{i+1} + \cdots + q_{j-1} = 0$

一条被删去的 $i \to j, i > j$ 的条件是 $q_j + q_{j+1} + \cdots + q_{i-1} \ge 2$

然后就可以 dp 了，$dp(i,j)$ 表示上一个 1 在 i，上上一个 1 在 j 的已经确定的值的最小值。 $i < j < k$，那么 $dp(i,j)$ 可从 $dp(j,k)$ 转移过来，新增的值是 $[j+1, i]$ 内部的边和 $[1, j] \to [i+1, n]$ 的边。

AT5148 [AGC036E] ABC String

首先把连续的同一字母缩成一段，然后考虑 `A`, `B`, `C` 的个数分别为 c_A, c_B, c_C，不妨设 $c_A \le c_B \le c_C$。答案有一个上界是 $3c_A$，可以这道题不一定能取到。

那么我们的目标首先是 $c_C = c_B$，也就是让 c_C 减小。`A` 把字符串分成了若干段，每一段都是 `CB` 交替出现。一个方法是 `CBCB...` 可以删去开头的 `C`，`..BCBC` 可以删去结尾的 `C`，在两个 `A` 之间的不能删空，直到 $c_B = c_C$

但是可能没有办法删到 $c_B = c_C$，比如 `ACACABCBCBCBA`，那么在这种情况下我们不得不删除 `AC` 来使 c_C 的数量减少。可以证明经过这一步肯定有 $c_B = c_C$。

然后就是让 $c_A = c_B = c_C$，那么我们只需要删 `BC` 即可。可以证明一定是可以删的。

然后就输出最后的方案即可。

AT5149 [AGC036F] Square Constraints

首先 P_i 有上界 $R_i = \min(\lfloor\sqrt{4 \times n^2 - i^2}\rfloor, 2 \times n - 1)$，下界 $L_i = \max(\lceil\sqrt{n^2 - i^2}\rceil, 0)$

然后先不考虑下界，那么方案数是很好算的，就是把 R_i 从小大大排序得到 R'_i，然后就是：

$$\prod_{i=0}^{2n-1} R_i + 1 - i$$

但是现在有了下界，我们可以钦定 k 个违反下界，也就是 $\le L_i - 1$，那么乘上容斥系数 $(-1)^k$ 加起来就是答案。钦定 k 个违反下界可以 dp。首先对于所有的 (L_i, R_i)，如果没有下界就是 R_i 否则就是 $L_i - 1$ 从从小到大排序。设 $f_{i,j}$ 表示前 i 个，已经钦定了 j 个 $< L_i$ 的方案数。那么考虑转移：

- 当前的数字下界是 0

  那么这个数字无法被钦定。并且任意下界非 0 的，上界一定 $\ge$ 这个数的上界。记前面下界为 0 的个数是 c_1，那么贡献就是 $R_i + 1 - c_1 - j$

  $$f_{i+1,j} \leftarrow f_{i,j} \times (R_i + 1 - c_1 - j)$$

- 当前数字下界非 0
  - 钦定小于下界

    可以选的数字本来是 L_i，前面下界为 0 的都会占用一个位置，然后钦定小于下界的也会占用一个位置，所以贡献为 $L_i - c_1 - j$

    $$f_{i+1,j+1} \leftarrow f_{i,j} \times (L_i - c_1 - j)$$

  - 放任自流

    可以选的数字个数本来是 $R_i + 1$，下界为 0 的都会占用 n 个位置，钦定小于下界的都会占用 k 个位置，前面下界不为 0 的个数为 c_2，那么放任自流的 $c_2 - j$ 个也会占用位置，于是：

    $$f_{i+1,j} \leftarrow f_{i,j} \times (R_i + 1 - n - k - c_2 + j)$$

然后我们就可以在 $\mathcal{O}(n^2)$ 的时间内求出一个 k，在 $\mathcal{O}(n^3)$ 时间解决本题。

AT5160 [AGC037C] Numbers on a Circle

首先如果 $B_i < A_i$ 就直接输出 -1

正着做看上去很难做我们考虑倒着做，也即是 $B_i \leftarrow B_i - B_{i-1} - B_{i+1}$

那么我们现在 B_i 有一个最大值，那么这个最大值两边都不能操作，直到比最大值变小才能可能可以操作。因此我们可以直接用最大值两边的和去减最大值直到不能减为止。如果最大值不能被减并且和 a_i 不相等就无解。

一次至少能减小一半，用堆维护最大值，因此复杂度是 $\mathcal{O}(n \log n \log W)$

AT5161 [AGC037D] Sorting a Grid

题目没说要输出 -1 因此我们可以认为一定有解。

然后我们可以一列一列构造，记 x 的颜色为 $\lceil \frac{x}{m} \rceil$，然后一行向所有有的颜色连边，找出一个最大匹配就是这一列的值。把这一列扬了，后面一定仍然能够构造。跑 m 次最大流即可。

AT5162 [AGC037E] Reversing and Concatenating

首先 $k = 1$ 暴力就好了。

然后如果 $k>1$，不妨设最小的字母为 A，我们肯定希望开头连续的 A 尽可能多。那么我们的策略就是首先第一次选一段以尽可能多的 A 结尾的字符串，不妨设为 sA，然后下一次就是 sAA，再下一次是 $sAAAA$。但是最后一次是 $A\dots As_R$。由于 SS_R 是回文串，于是就相当于找一段连续的 A，首先这一段 A 数量尽可能多，如果一样多就比较后面跟着的谁小。

AT5163 [AGC037F] Counting of Subarrays

/yun

首先判断一个序列是否合法，可以找到最小值 x，把 x 的极长连续段 $[l_1,r_1],\dots,[l_k,r_k]$ 拎出来，然后如果 $r_i-l_i+1<m$ 那么肯定不合法，否则把这一段变成 $(r_i-l_i+1)/m$ 个 $x+1$。如果最后能变成一种数并且长度 1 或这 $\geq l$ 就合法。

我们希望统计合法区间个数。不妨考虑一个“加强版”：$\sum_{i,j}L_iR_j[S_{i,j}\text{ is good}]$，那么我们原来的问题就是 $L_i=R_i=1$。现在这样的好处是合并的时候可以把权值合并，比如 $1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ a\ b\ c\ d\ e$，其中 $a,b,c,d,e>1,l=3$，那么可以合并成 $2\ 2\ 2\ a\ b\ c\ d\ e$，那么取子串 $2\ 2\ a\ b\ c$，可以表示 $\overbrace{1\ 1\dots 1\ 1}^{6\sim 8}\ a\ b\ c$，也就是说 $L_2'=L_2+L_3+L_4$。然后用栈维护即可。

至于为什么是对的小编也很想知道呢。

AT5200 [AGC038C] LCMs

爆推，记 $M=10^6,c_i=\sum_{1\leq j\leq n}[A_j=i]$，然后原始和 $\sum_{i=1}^n\sum_{j=1}^n\operatorname{lcm}(A_i,A_j)$ 显然就差了 $\sum_{i=1}^nA_i$ 再除以二，于是：

$$\begin{aligned}
&\sum_{i=1}^n\sum_{j=1}^n\operatorname{lcm}(A_i,A_j)\\
=&\sum_{i=1}^M\sum_{j=1}^M\operatorname{lcm}(i,j)c_ic_j\\
=&\sum_{d=1}^M\sum_{i=1}^{\lfloor\frac{M}{d}\rfloor}\sum_{j=1}^{\lfloor\frac{M}{d}\rfloor}\frac{id\times jd}{d}c_{id}c_{jd}[\gcd(i,j)=1]\\
=&\sum_{d=1}^M\sum_{i=1}^{\lfloor\frac{M}{d}\rfloor}\sum_{j=1}^{\lfloor\frac{M}{d}\rfloor}i\times j\times d\times c_{id}\times c_{jd}\sum_{x|i,x|j}\mu(x)\\
=&\sum_{d=1}^M\sum_{x=1}^{\lfloor\frac{M}{d}\rfloor}\mu(x)\sum_{i=1}^{\lfloor\frac{M}{dx}\rfloor}\sum_{j=1}^{\lfloor\frac{M}{dx}\rfloor}ix\times jx\times d\times c_{idx}\times c_{jdx}\\
=&\sum_{T=1}^MT\left(\sum_{x|T}\mu(x)\times x\right)\left(\sum_{i=1}^{\lfloor\frac{M}{T}\rfloor}i\times c_{iT}\right)^2
\end{aligned}$$

AT5201 [AGC038D] Unique Path

其实本质就是构造题。

首先对 $c_i=0$，把 a_i,b_i 连起来，最后得到若干个连通块。只有一条路径就意味着这些连通块都是树。

然后如果 $c_i=1$，a_i,b_i 在同一个连通块内就无法构造。

然后手玩一下发现每个树上只有一个点可以向外连边，连通块个数为 k，现在还需要连 $m-n+k$ 条边。然后讨论有没有 $c_i=1$：

- 如果没有，那么 $k-1\leq m-n+k\leq\binom{k}{2}$ 就一定能构造
- 如果有，那么 $k\leq m-n+k\leq\binom{k}{2}\wedge k>2$ 就一定能构造

AT5202 [AGC038E] Gachapon

感觉如果做过类似的题目很容易想到 $\min-\max$ 反演：

$$E(\max(S))=\sum_{T\subseteq S,T\neq\varnothing}(-1)^{|T|+1}E(\min(T))$$

然后考虑钦定了 T 如何去计算 $E(\min(T))$，记 $W(T)=\sum_{i\in T}A_i$，$S=W(\{1,2,\dots,n\})$。现在不在 T 里面的我们已经不用管了，只考虑 T 的期望步数如果是 x，走一步恰好在 T 的概率是 $W(T)/S$，因此期望是 $S/W(T)\times x$ 步。于是我们现在只考虑 T 内部。然后期望有一种众所周知的转化是 $E(X)=\sum_{i=0}P(X>i)$，所以我们不妨枚举 k 步还没有走完，也就是枚举 $\sum_{i\in T}c_i=k$，$\forall i\in T,c_i<b_i$，那么这一组 c_i 对答案的贡献是：

$$\prod_{i\in T}\left(\frac{A_i}{W(T)}\right)^{c_i}\frac{k!}{\prod_{i\in T}c_i!}$$

然后比较经典地把每一个 i 的贡献提出来：

$$k!W(T)^{-k}\prod_{i\in T}\frac{A_i^{c_i}}{c_i!}$$

于是计算 T 就是：

$$S/W(T)\sum_{k=0}k!\times W(T)^{-k}\times\sum_{\sum_{i\in T}c_i=k,\forall i\in T,c_i<b_i}\prod\frac{A_i^{c_i}}{c_i!}$$

最后的答案就是：

$$\sum_T(-1)^{|T|+1}\sum_{k=0}S\times k!\times W(T)^{-k-1}\times\sum_{\sum_{i\in T}c_i=k,\forall i\in T,c_i<b_i}\prod\frac{A_i^{c_i}}{c_i!}$$

虽然这个式子看上去又臭又长，复杂度爆炸，但是确实可以做了，设 $f_{i,j,k}$ 表示考虑了前 i 个数，$W(T)=j$，$\sum c_i=k$，红色部分的贡献和，转移就考虑 i 在不在 T 里，c_i 是多少，即：

$$f_{i,j,k}=f_{i-1,j,k}-\sum_{l=0}^{b_i-1}f_{i-1,j-a_i,k-l}\times\frac{A_i^l}{l!}$$

然后答案是：

$$\sum_j\sum_k S\times k!\times j^{-k-1}\times f_{n,j,k}$$

因为 $\sum B_i$ 是 $\mathcal{O}(n)$ 级别的所以复杂度是 $\mathcal{O}(n^3)$ 的。

AT5203 [AGC038F] Two Permutations

手玩一下，发现对于一个环，要么都是 i，要么都是 P_i。然后我们求出 i 所在的环 p_i 和 q_i，每个环可以旋转或不旋转，考虑 $A_i=B_i$ 的条件：

- 如果 $P_i=Q_i=i$，那么一定
- 如果 $P_i=i,Q_i\neq i$，当且当且仅当 q_i 不旋转
- 如果 $P_i\neq i,Q_i=i$，那么当且仅当 p_i 不旋转
- 如果 $P_i\neq i,Q_i\neq i,P_i\neq Q_i$，那么当且仅当 p_i 和 q_i 都不旋转
- 如果 $P_i=Q_i\neq i$，那么当且仅当 p_i 和 q_i 状态相同

于是就可以网络流了，$S\to p_i$ 联通表示旋转，$q_i\to T$ 联通表示旋转，如果割掉表示不旋转，要付出代价，具体是：

- 没有
- $q_i\to T$ 的代价增加 1
- $S\to p_i$ 代价增加 1
- 如果都不旋转那么 $S\to q_i,p_i\to T$ 是联通的，那么就要割掉 $q_i\to p_i$ 付出 1 的代价
- 都旋转是 $p_i\to q_i$ 付出 1 的代价，都不旋转是 $q_i\to p_i$ 付出 1 的代价

然后就可以做到 $\mathcal{O}(n\sqrt{n})$ 了。

AT5617 [AGC039C] Division by Two with Something

首先有一个结论是一次操作就相当于把这个 n 位二进制数最后一位取反移到最高位。

然后就相当于把 a 和 $\neg a$ 拼起来得到 b，每次把 b 的最低位移到最高位，问多少次才能回到原来。也就是循环节。

然后考虑循环节 T 有什么性质，首先 $T|2n$ 是肯定的，又因为我们刚才的定义所以 n 不是循环节所以 $T\not|\ n$。

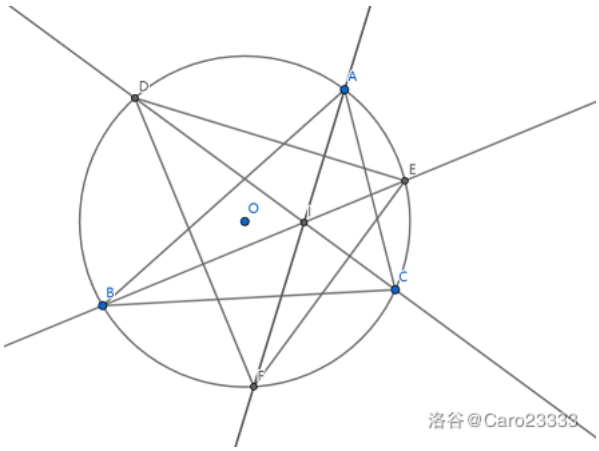
然后我们可以断言 $\frac{2n}{T}$ 一定是奇数，也就是把整个串分成了奇数段。然后设循环节前 $\frac{T}{2}$ 是 p 后 $\frac{T}{2}$ 是 q 那么前半个串就是 $\overline{pq\dots qp}$，后半个串是 $\overline{qp\dots pq}$。

现在我们还有 x 的限制，不妨先枚举 T，然后前 $\frac{T}{2}$ 位是 s，如果 $\overline{s\neg ss\neg s\dots}$ 是合法的话就有 $s+1$ 否则只有 s

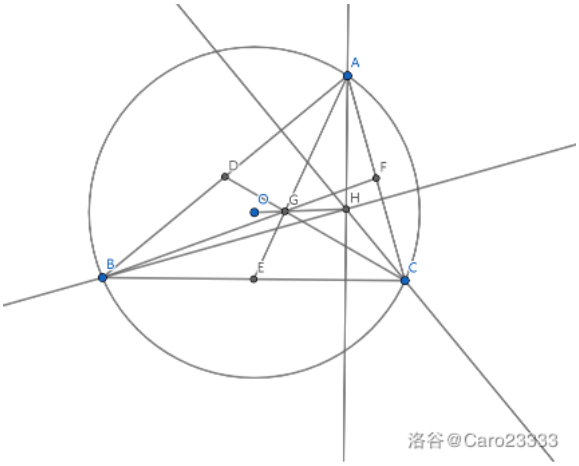
然后我们这个是钦定循环节是 T，可能循环节实际上是 T 的因数，所以还需要减去那些值，才是恰好为 T。

AT5618 [AGC039D] Incenters

首先把三条角平分线画出来：



然后 I 就是我们要求的点。可以证明 I 是垂心。然后只看 $\triangle DEF$：



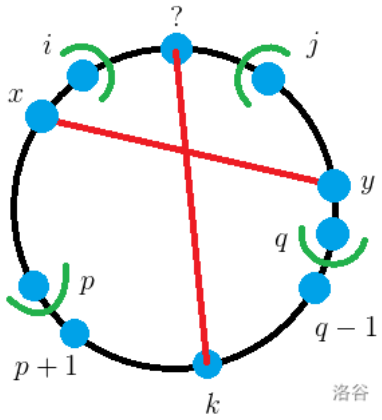
然后 G 是重心，H 是垂心，根据某典中典我们知道 $\vec{OH}=3\vec{OG}$，所以重心的坐标乘三就是垂心的坐标。

于是我们只要求出三个弧中点的和的期望就可以了，然后就只需要对每一个中点计算对答案的贡献即可。

AT5619 [AGC039E] Pairing Points

首先让 $n=2N$，我们枚举 1 和谁连接，那么分成了两个区间。

然后我们发现对于区间 $[i,j]$，连边大概是这样的：



有且仅有一条出边，那么为了连成树，必须有一条从 $[i,k)\to(k,j]$。于是枚举最上面的一条 $x\to y$。那么找到 $[i,p]$ 满足除了 x 没有向外的边 $[q,j]$ 同理。于是我们设 dp 数组 $f_{l,i,r}$ 表示 $[l,r]$ 留出 i 向外连边的合法方案数，那么有转移：

$$f_{l,i,r}=\sum f_{l,x,p}f_{p+1,i,q-1}f_{q,y,r}[A_{x,y}]$$

然后直接做这个复杂度是 $\mathcal{O}(n^7)$ 的，因为常数很小所以是可以过的。发现当 p,q 固定时，$\sum$ 内部的东西除了 $f_{p+1,i,q-1}$ 都是和 i 无关的，于是可以一起计算。$f_{p+1,i,q-1}\sum f_{l,x,p}f_{q,y,r}[A_{x,y}]$。令 $g_{l,k,p}=\sum f_{l,j,p}[A_{j,k}]$，那么后面的就是 $\sum g_{l,k,p}f_{q,k,r}$，计算 f 的时候顺便计算 g 就可以做到 $\mathcal{O}(n^5)$。

AT5660 [AGC040B] Two Contests

首先可以有每个区间最大左端点 $L=\max l_i$，和最小右端点 $R=\min r_i$，那么如果 L,R 是一条线段贡献的，那么这条线段所在集合一定是 $[L,R]$，另一个集合选最长那条即可。当然如果 $n=2$ 只有一种情况。

然后剩下的情况是两种：

- 如果 $[L,?]$ 和 $[?,R]$ 在同一个集合，那么这个集合的交一定是 $[L,R]$，另一个集合选最长
- 否则在不同集合，如果 $[l_i,r_i]$ 和 $[L,?]$ 在同一个集合，那么这个集合的长度就要 $\operatorname{chkmin}(\max(0,r_i-L+1))$，如果和 $[?,R]$ 在同一个集合那么就要 $\operatorname{chkmin}(\max(0,R-l_i+1))$。于是就变成一些点，每个点有两个参数 (a,b)，第一个集合的和是第一维的最小值，第二个集合是第二个的最小值，按第一位排序，枚举到一个 a 时 $(<a)$ 的 b 的最小值就是此时第二个集合的权值。

AT5661 [AGC040C] Neither AB nor BA

考虑一个转换是对一个 s，把 s 奇数位上的 `A` 和 `B` 反转，那么不能删的就变成了 `AA` 和 `BB`。这显然是一个双射。

不能删 `AA` 和 `BB` 那么需要 `A` 和 `B` 的数量不大于 $\lfloor\frac{n}{2}\rfloor$。首先这个是必要的。然后可以归纳地证明是充分的。

然后就不难计数了，首先随便填，然后减去非法的即可。

AT5662 [AGC040D] Balance Beam

首先如果顺序固定，我们可以把 Alice 和 Bob 的折线（位移-时间图像）画下来，然后交点就是他们相遇的地方。不妨设 $\sum A_i=suma$，那么 Alice 折线的终点就是 $(n,suma)$。

然后我们现在要算概率。其实这个概率非常假，因为合法的是连续的一段区间（显然越远越难追到）。具体的计算方式是下移 Bob 的折线直到恰好有交点，再往下就没有交点，然后这时 Bob 的折线会和坐标轴有一个交点 $(k,0)$，也就是最远距离。然后 k/n 就是我们要求的概率。

但是现在方案没有给我们。有一个求法是计算 $(k,0)$ 出自那一块砖，不妨设为 (A_i,B_i)，那么 $\lceil k\rceil$ 的纵坐标至少为 B_i，我们希望最小化后面的线段数量。然后后面的折线观察肯定是先往上走 Bob 的折线，有交点，然后走 Alice 的折线，于是一块砖最大的贡献是 $\max(a_i,b_i)$，我们把斜率尽可能大的放在 (A_i,B_i) 这块砖右边就可以保证后面的砖尽可能少了。然后我们需要证明一定可以取到。其实只需要按照 a_i-b_i 从小到大摆放即可。

于是就很清楚了，用一个分数表示答案，我们枚举砖，然后二分后面需要多少块，得到有多少块之后可以得到具体的坐标，对所有坐标取最大值即可。

p.s. 为了实现方便可以先存到 n 距离的最小值。

AT5663 [AGC040E] Prefix Suffix Addition

这道题也比较演。首先其实这个前缀加、后缀加都是唬人的，因为我们可以补 0 来做到任意一段加不降子序列、不升子序列。

然后如果只有一种是很好做的，比如只有不降那么总数就是 $\sum_{i=0}^{n}[A_i>A_{i+1}]$。但是有两种就比较麻烦，我们需要 dp。然后有一个很显然的 dp 是 $f_{i,j}$ 表示 i 有 j 分配给了不降，剩下 A_i-j 分配给了不升。那么转移是 $f_{i+1,j}\leftarrow f_{i,k}+[k>j]+[A_i-k<A_{i+1}-j]$，后面化简一下是 $[k-j>0]+[k-j>A_{i+1}-A_i]$。于是我们的 $f_{i,j}$ 相同时希望 j 越大肯定是最优的。然后我们还发现 $f_{i,*}$ 的极差不会超过 2，因为最差顶多增加 2，于是我们只需要记录取最小值时最优的 j，取次小值时最优的 j 即可。

状态数是 $\mathcal{O}(n)$ 的。

AT5664 [AGC040F] Two Pieces

神仙计数。

还是先转化，为了防止操作重复，我们当距离为 1 的时候只能使用瞬移而不是移动后一颗棋子，那么可以发现两个操作序列不同的每一步位置的序列也一定不同。现在就只需要对操作计数即可。

我们把前面的棋子叫做 b，后面的棋子叫做 a，那么 b 操作向前的次数一定是 B，但是 a 操作的次数不一定，我们假设向前次数是 k。

如果没有瞬移操作，那么这个是容易计数的，看成一个点 $(b\text{的坐标},a\text{的坐标})$，那么一次上移一步或左移一步，除了 $(0,0)$ 不能碰到 $x=y$ 这条线，最终的目标是 (B,k)。这个方案数计数非常经典如果碰到了就翻转路线，也就是 $\binom{B+k-1}{B-1}-\binom{B+k-1}{k-1}$。

然后考虑插入瞬移操作。首先最后一个瞬移操作的位置是确定的，我们要把 (B,k) 上移到 (B,A)，然后折线会和 $x=y$ 有交点，最后一个交点就一定是我们最后一次瞬移。

然后不考虑这一次瞬移，前面的瞬移能操作当且仅当目前距离为 d_i，$j>i$，都满足 $d_j>d_i$，否则就撞到线上去了。然后分析一波可以选的 $d_i\in[0,A-k]$，于是 $A-k+1$ 个变量和为 $n-B-k-1$，随便组合数一波即可。

复杂度 $\mathcal{O}(n)$。

AT5695 [AGC041D] Problem Scores

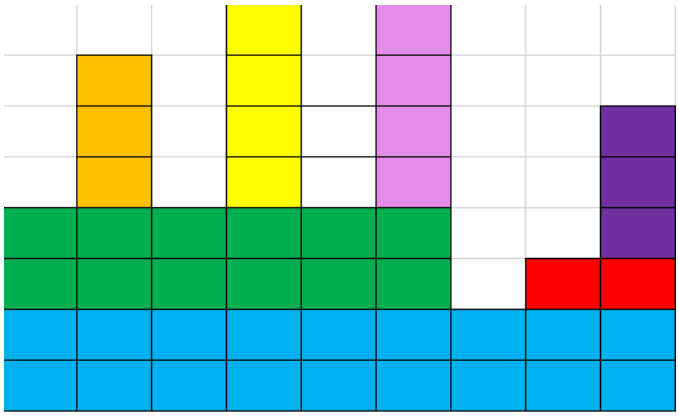
首先这个要求三是最难处理的，我们先考虑这个约束。因为要对任意满足，所以 S 取后 k 个，T 取前 $k+1$ 个。然后我们记 $\Delta=\sum_{x=1}^{k+1}A_x-\sum_{x=n-k+1}^{n}A_y$。然后考虑 $k\leftarrow k+1$，那么 Δ 的改变量是 $A_{k+2}-A_{n-k}$，所以 $k=\lfloor\frac{n}{2}\rfloor$ 时 Δ 是最小的，我们只需要让这个时候是 $\Delta>0$ 即可。

然后考虑这个不降子序列，然后 $1\le A_n\le n$ 的限制，我们可以每次选一个前缀全部 -1，然后这样得到的一定是不降的。对每一个前缀，Δ 的减小量也是固定的，于是就相当于有 n 个物品，每个可以取无限个，要求重量和不超过 $n-1$。简单 dp 即可。

AT5697 [AGC041F] Histogram Rooks

牛逼计数

图大概是这样的：



首先这个完全覆盖我们考虑容斥，钦定若干个不被覆盖，剩下的随意，然后就可以发现和这些钦定的同一行、同一列的联通的都不能放車。容斥系数显然是 $(-1)^k$，其中 k 表示钦定不被覆盖的个数。

然后考虑一个非常 Naive 的 dp 就是像上图一样分割成笛卡尔树的形式，一个点被钦定不选那么同一行是不能放車了，并且这一列也不能放車，因此我们 dp 的时候记下当前有多少列是不能放车的，然后合并就是一个背包，因为在笛卡尔树上复杂度是 $\mathcal{O}(n^2)$ 的。

然后考虑现在加入新的一行。我们之前已经有 p 列不能放車，现在的长度是 len，那么两种情况：

- 这一行没有車，那么系数就是 2^{len-p}
- 这一行有車，那么系数就是 $\sum_{i=1}^{p}\binom{p}{i}(-1)^i=-[p>0]$

但是这样还有一个问题就是我们有 p 列不能放車，但是这 p 列不一定有钦定的点，我们还需要再上一个容斥，钦定若干列是不能放車但是有钦定的点，假设被钦定的有 q 列。于是我们重新考虑两类系数：

- 没有車显然是 2^{len-p}
- 有车显然是 $-[p>q]$

于是就可以 dp 了，随便分析一下就知道是 $\mathcal{O}(n^2)$ 的。因为我比较菜写了 log。

AT5800 [AGC043C] Giant Graph

谁能想到这是博弈论/yun

首先一个结论是 10^{18} 很大，因此我们肯定是按照 $x+y+z$ 从大往小选。

于是我们可以连有向边，从小的往大的连边，于是一个点被选当且仅当没有后继或所有后继都不选。

这个看上去和博弈论很像。其实就是一个博弈论，答案是所有必败点的权值和。

然后根据某经典结论三个游戏是独立的，SG 函数就是分别在三张图的 SG 函数的异或和，于是我们设三张图的 SG 函数分别是 $f(x),g(x),h(x)$，于是就相当于：

$$\sum_{f(x)\oplus g(y)\oplus h(z)} 10^{18(x+y+z)}$$

因为 SG 函数值域是 $\mathcal{O}(\sqrt{m})$ 的所以我们可以直接枚举 $f(x),g(y)$ 的值，然后乘起来，复杂度 $\mathcal{O}(m)$。

AT5801 [AGC043D] Merge Triplets

首先一个观察是不可能出现 a_i 满足 $a_i>a_{i+1,i+2,a+3}$，不然分析一波根本不可能选 a_i

然后这个是一个必要条件，但是不够充分。考虑前缀最大值把整个序列划分成了若干段，刚刚的条件意味着每一段长度都 ≤ 3。

分析一个长度为 3 的块，要么是变成 3 的段一次选完，要么是 1 和 2，要么是 3 个长度为 1 的。于是必定有长度为 1 的段的数量 $\ge$ 长度为 2 的段。

于是就可以 dp 了，按这个段 dp，顺便记录一个长度为 $1-$ 长度为 2 的值即可。

复杂度 $\mathcal{O}(n^2)$。

AT5802 [AGC043E] Topology

绕绳子一不小心把自己绕进去了/dk

首先考虑 SPJ 怎么写。你要把绳子拉出去，那么一定有可以拉的地方。官方题解给出了一种很高妙的方法是对于一个点 $(i+0.5,0.5)$，顺着绳子有一圈，如果从上面经过那么记录下 u_i，从下面经过记录下 d_i，得到一个字符串，如果这个字符串有相邻的两位是相同的，就消去，也就意味着把绳子拉过来。如果最后能消完，也就意味着能取出，否则一定不能取出。而有了一个串我们也很容易构造出对应的绳子。

然后知道了这个之后我们考虑一种极其特殊的情况 `111...1110`，也就只有全部满的是不行的，否则一定可以拿出。这个看上去非常难构造。考虑一下方法：

假装有 n 个点我们已经构造出了一个合法的串 f_n，现在我们在最前面加入了一个 0，我们还希望是合法的，那么我们先把 f_n 里所有的标号加 1，比如 $u_i\to u_{i+1}$，变成 a，记 a 的翻转为 a'，那么 $u_0\ a\ u_0\ d_0\ a'\ d_0$ 是一个符合条件的 f_{n+1}，如果你去掉的是 0 那么 a 和 a' 中间两个一定是一样的肯定可以消空，否则不是的话可以先把 a 和 a' 消空，然后 u_0 消掉，d_0 消掉。

一个显然的观察是点越多越难拉出来，因此如果 S 可以拉出来但是 $S\subset T$ 却不能拿出来就一定是不合法的。然后没有这种情况就一定合法，考虑给出构造：

找到一个 S 满足 S 不能拿出来但是 $T\subset S$ 都能拿出来，也就意味着这个就是上面说的情况，直接按照上面的构造即可。

但是这样极小的 S 可能不唯一，把所有构造出来的绳子缝合起来即可。

上界很松随便写一写就过了。

AT5803 [AGC043F] Jewelry Box

首先因为要满足差分约束，所以重量交叉肯定是不优的，由此我们知道同一家店的珠宝是按照重量排序的。

然后我们设 $X_{i,j}$ 表示 $(i,1)\dots(i,j)$ 的购买数量，那么：

- $0=X_{i,0}\le X_{i,1}\le\cdots\le X_{i,k_i}=A$
- $X_{i,j+1}-X_{i,j}\le C_{i,j}$
- 对于限制 (u,v,w) 和珠宝 (v,j)，假设 k 是最小的满足 $S_{u,k}+w\ge S_{v,j}$ 的值，那么限制就等价于 $X_{u,k-1}\le X_{v,j-1}$

于是我们可以写长如下的形式：

$$\min\left\{\sum_{i,j}(\infty\max(X_{i,j}-X_{i,j+1},0)+\infty\max(X_{i,j+1}-X_{i,j}-C_{i,j},0)+P_{i,j}\max(X_{i,j+1}-X_{i,j},0))+\sum_{u,v,w,j}\infty\max(X_{u,j}-X_{v,k},0)+\sum_i(\infty X_{i,}\right.$$

这个东西对偶再取反是一个费用流，我们设原点是所有 $X_{i,0}$ 没有任何流量限制，汇点是 X_{i,k_i}，然后对于一条 $c\max(X_v-X_u-w)$，我们看做是从 u 到 v 连一条流量为 c 费用为 w 的边。

然后这个就是一组数据的情况，考虑这个 A 变化时费用是 $C-AF$，其中 C 是吧 A 当成 0 的费用，F 是流量，取反后是 $AF-C$，然后就相当于求 $\min\{AF-C\}$，我们可以求费用流的时候顺便求出每一对 (F,C)，因为最短路长度是变大的，因此当最短路超过 A 时费用就变成正的了显然不优，于是我们找到长度最长但是不超过 A 的对应的 (F,C) 输出 $AF-C$ 即可。