IOI2015中国国家集训队第一次作业
试题泛做部分

宁波镇海蛟川书院 卢啸尘

Contents

1 第一部分(Codeforces)

1.1 263E. Rhombus

题目描述

有$N \times M$的表格，定义位置(i,j)的分数为，所有与其曼哈顿距离小于k的格子的数值乘以$k - manhattanDistance$的总和。这一定义只定义在$k \leq i \leq N-k+1, k \leq j \leq M-k+1$。求最大的分数出现在哪个位置。

数据规模

$N, M \leq 1000$

解法

我们把这个斜置正方形切分成四个等腰直角三角形。通过预处理水平、竖直、斜向的前缀和，可以$O(NM)$计算出每个等腰直角三角形放在某个位置的数值和，进而可以维护出加权以后的数值和。这其中，斜向的和是用来在移动正方形中心时计算边界上的变化，水平和竖直的和是为了维护四个部分的不加权数值和。

1.2 293B. Distinct Paths

题目描述

有$N \times M$的表格，有些位置已经被染上K种颜色中的一种。将剩下的每个位置染上那K种颜色的一种，使得从左上到右下的每一条最短路径都不包含相同颜色的格子。求染色方案数，对10^9+7取模。

数据规模

$N, M \leq 1000, K \leq 10$

解法

首先1000的数据规模是坑爹的，稍有常识的人都能看出，最短路径的长度是$N+M-2$，通过$N+M-1$个格子，从而若$K \leq N+M-1$，可直接宣告无解。

那么现在N, M的范围都变成了个位数，这启示我们进行爆搜。从上到下从左到右考虑每个格子，维护一个数组，数组中的第j个位置保存的是最后一个处理过的横坐标为j的位置，其左上所有格子的颜色的情况（二

进制表示）。每层只会在这个数组中修改一个值，并且可以直接用位运算(OR)实现。

可惜的是，这样是会超时的。

注意到搜索时有很多相似的情况。比如，如果在某个格子上填入某个尚未使用过的颜色i，那么这样的方案数将恰等于填入其它尚未使用过的颜色j的方案数：只需要将最后的方案中的这两个颜色的区域对调即可。加入这个优化以后就可以通过了。

1.3 235E. Number Challenge

题目描述

求$\sum_{i=1}^{A}\sum_{j=1}^{B}\sum_{k=1}^{C}d(ijk)$。对$2^{30}$取模。

数据规模

$A, B, C \leq 2000$

解法

首先经过观察可得以下引理：

令$S_d(r,k)=\sum_{i=1}^{r}d(ki)$

令$k=p^{\alpha}q$, 其中$(p,q)=1$，则$S_d(r,k)=(\alpha+1)S_d(r,q)-\alpha S_d([\frac{r}{p}],q)$。

关于正确性，可以试着将$S_d(r,k)$和$S_d(r,q)$的各项写出来以后进行一下分析。

这个引理启示我们，只需要计算一系列$S_d(C,ij)$即可。

又，从公式中可以看出，在计算$S_d(r_0,k_0)$时，需要计算的所有S_d函数的参数中，r都是某个$[\frac{r_0}{x}]$，这样的r 的个数在$O(\sqrt{r_0})$。而所有ij的约数，其数量级也是在$O(AB)$级别，从而，需要计算的不同的S_d数目在$O(AB\sqrt{C})$，或$O(N^{\frac{5}{2}})$。使用记忆化搜索解决。

证明引理

在$S_d(r,q)$的各项中，由于q中没有因子p，所以qi有因子p仅当i有因子p。从而，对于除了项编号(i)是p的倍数的项以外，有$d(p^{\alpha}qi)=(\alpha+1)d(qi)$。

而对于那些有因子p的项$i=p^{\beta}j$，$d(p^{\alpha}qi)=(\alpha+\beta+1)d(qj)$，$d(qi)=(\beta+1)d(qj)$，从而$(\alpha+1)d(qj)=\alpha\beta d(qj)+d(p^{\alpha}qi)$。

其中，$\beta d(qj)=d(qp^{\beta-1}j)=d(q\frac{i}{p})$。

从而多出来的部分就可以写成$\alpha\sum_{p|i}^{r}d(\frac{qi}{p})$，也即$\alpha\sum_{i'=1}^{[\frac{r}{p}]}d(qi')$，即$\alpha S_d([\frac{r}{p}],q)$。

1.4 325E. The Red Button

题目描述

有一个N个点的图，编号从0开始。每个点i有两条出边，分别指向$2i \bmod N$和$(2i+1) \bmod N$。构造一个Hamilton 回路。

数据规模

$1 \le N \le 10^5$

解法

在讨论算法之前，首先说明一个性质：若N为奇数，无解。

证明：0有一条入边来自$\frac{N-1}{2}$，另一条是自环，显然，0只能作为$\frac{N-1}{2}$的后继。同理，$N-1$也只能作为$\frac{N-1}{2}$的后继。然而$\frac{N-1}{2}$只有一个后继。故不可能。

现在我们只需要讨论N为偶数的情况。我们把i和$i+\frac{N}{2}$合成一个点，因为它们有着共同的出边。这样，问题就转变成一个Euler回路问题——所有目标点相同的边都已经被合并了，所以只要将Euler回路中所有边的初始目标点编号写下来就是原图的一个Hamilton回路。

我们首先任意为每条边（也就是原来的每个点）指定后继，从而形成一系列回路。如果回路的数量不为1，我们必然能找到一个点同时属于两个回路。交换这个点的两个出边来合并两个回路。

下面将证明如果找不到一个点属于不同的回路，则只有一个回路。方便起见，用每条边的初始目标点编号称呼这条边，特别的，边x实际上指的是初始目标点为$x \bmod N$的边。

考虑点0。它只属于一个回路，这就是说边0和边1都在这个回路中，进而推出，边$0 \sim 3$都在同一个回路中。你可以一直重复这个过程，直到推出边$0 \sim 2^k-1$属于同一个回路，其中$2^k \ge N$。

1.5 260E. Dividing Kingdom

题目描述

有一个二维平面，上面有N个点，现要用4条直线将其分成9块，第i块包含了A_i 个点。直线不能穿过点。你可以自己把编号$1 \sim 9$分配给切割后的9块。

数据规模

$N \le 10^5$

解法

首先枚举所有362880种编号分配方式。对于每种方式，我们可以立刻知道各条直线的位置，剩下的只是验证。验证相当于一个子矩形求和问题，可以使用函数式线段树解决。

1.6 325D. Reclamation

题目描述

有一个$R \times C$的矩形，其中C的一维循环联通。共N次，每次指定一个格子，如果设置为不可通行后，顶部和底部依然连通，则将其设置为不可通行，求最后不可通行的格子数目。

数据规模

$R, C \leq 3000, N \leq 300000$

解法

感性认识一下：如果顶部和底部不四联通了，那么在中间必然有一个由不可通行格子构成的八连通环。将C的一维扩大到两倍，用并查集维护，如果(i, j)和$(i, j + C)$ 八连通了，则说明顶部和底部已经被截断了。

特别的，在tsinsen上，本题可能需要加读入优化才能通过。

1.7 360D. Levko and Sets

题目描述

有M个数B_i，有N个集合，分别由一个A_i生成：最初集合里只有1，然后只要元素k存在，就加入$k \times A_i^{B_j} \bmod P$。求$N$个集合的并的大小。这里$P$是一个质数。

数据规模

$N \leq 10^4, M \leq 10^5, 1 \leq A_i \leq P \leq 10^9, 1 \leq B_i \leq 10^9$

解法

令$T = GCD(\phi(P), B_1, B_2, ..., B_M)$，则集合$i$中含有的就是在模$P$意义下$A_i^T$的所有幂。

令P的原根为g，则可以表示成$g^{k \times GCD(Index(A_i^T), \phi(P))}$。

注意到$GCD(Index(A_i^T), \phi(P))$的个数，也即$d(\phi(P))$在可接受的范围（vfleaking教导我们$d(n) = O(n^{\frac{1.066}{\ln \ln n}})$），因此之后可以暴力$d^2(\phi(P))$ 容斥处理。

1.8 339E. Three Swaps

题目描述

有一个排列，允许进行至多三次区间翻转操作，要求最后得到递增排列。构造一解即可，不要求操作数最少。

数据规模

$N \leq 1000$

解法

易知，三个操作最多产生连续的七段（升序或降序）。处理出所有的“七段”方案，与原排列进行匹配即可。

1.9 286E. Ladies' Shop

题目描述

有N个正整数组成的集合A, 其中每个数都不超过M。
构造尽量小的集合P，使得:
1. 对$A_i \in A, \exists Y : P \to N, \sum_{p \in P} pY(p) = A_i$。
2. 对$a \notin A, 0 < a \leq M, \neg\exists Y : P \to N, \sum_{p \in P} pY(p) = A_i$。

数据规模

$N, M \leq 10^6$

解法

如何说明无解？答：只要有$A_i + A_j \leq M$而$(A_i + A_j) \notin A$，则无解。证明，考虑最小的那个不符合第二个要求的a，设$a = A_{U_1} + A_{U_2} + ... + A_{U_k}$，若$k = 2$，则按照之前的方法就可以检查出来。否则，由$a$是最小的不符合第二个要求的值，$A_{U_{k-1}} + A_{U_k} \in A$，从而可以把$k$减小1，证毕。使用FFT就可以检验出无解。

在有解的情况下如何得出最小集合P？答：P包含且只包含那些不能表示成两个A 中元素和的元素。证明，如果一个数可以表示成若干个A中元素的和，则不需要放入它，反之必须放入，这一点是显然的。使用和上一

段的证明类似的方法，可以证明在有解的情况下，“若干个”和“两个”等价。现在我们得到了一个下界，显然它也是上界：所有不在这个P中的元素都可以被表示成P中若干元素的和，因此它已经是一个合法的解。同样用FFT解决。实际上，这两个问题可以在一次FFT中就能解决。

1.10 332D. Theft of Blueprints

题目描述

有N阶带权无向图，满足每K个点恰有一个公共邻居。任选K个点，求这K个点到这个邻居的边权值和的期望值。

数据规模

$1 \leq K < N \leq 2000$。

解法

我们来分析一下这个无向图的形态：

当$K = 1$：若干个大小为2的连通块。

当$K = 2$：一个三角形，或若干个共顶点的三角形。相当于在$K = 1$图的基础上加一个点，与每一个点连边。

当$K = 3$：只有四阶完全图。

当$K = 4$：只有五阶完全图。

实际上，当$K \geq 3$，只有$K + 1$阶完全图。下面将证明这一点。

考虑图中的最大度数点。设其度数为M。那么在由这M个点中的$K - 1$个构成的共C_M^{K-1}个$K - 1$元组中的任何一个，加上最大度数点本身，也构成了一个K元组。理所应当的，它有一个公共邻居——这个公共邻居只能在这M个点中，因为首先这个公共邻居是最大度数点的邻居。而我们发现，两个不同$K - 1$元组对应的这个公共邻居应当是不同的。否则，这两个不同的$K - 1$元组首先就至少有K 个点，这K 个点组成的K元组将至少存在两个公共邻居：一个是那个重复了的公共邻居，一个是那个最大度数点。从而$C_M^{K-1} \leq M$。推出$K \in \{1, 2\}$或$K \in \{M, M + 1\}$。而M至少为K，从而，若$K \geq 3$，则$M = K$。

现在考虑最大度数点和K个邻居，这$K + 1$个点的导出子图中，每K个点都与剩下一个点相连。这就是说，这是一个$K + 1$阶团。然而又因为最大度数为$M = K$，所以这个团所属的连通分量中只有这个团。由$K \geq 1$，显然这个图是连通的。结论：这个图是一个$K + 1$ 阶团，或称$K + 1$阶完全图。

由于这个图非常特殊，因此，很容易得出一个在IO复杂度下解决的算法，这里不再赘述。

1.11 335D. Rectangles and Square

题目描述

有N个互不相交的矩形，判断是否有某些矩形构成一个正方形。
若有需输出方案。

数据规模

$N \leq 10^5$。

解法

枚举两个顶点。如，左上角和右下角。

显然地，在正方形中，这两个顶点的$x-y$是相同的。我们按照$x-y$将所有的左上角和右下角分类。在每类中分别处理。

对于点i，定义参数S_i为最大的x，令从点i向右和向下的x长的线段都属于某个矩形，类似地定义T_i，但方向换成另两个。

在$x-y$相等的一类之中，按照x将所有顶点排序。进一步定义两顶点距离$d(u,v)$为顶点的x之差。

那么，左上角顶点u和处在u右下的右下角顶点v可以构成合法正方形的必要条件就是：$d(u,v) \leq min(S_u, T_v)$。也即，$x_u$ 在$[x_v - T_v, x_v]$中且x_v在$[x_u, x_u + S_u]$中。

我们观察到，如果一个右下角顶点v和多个左上角顶点满足上述条件，则只需要考虑离它最近的那个。证明：满足上述条件而不能构成合法正方形，说明这个正方形中有空白的单位正方形。更远的左上角顶点和v构成的正方形包含了原正方形，因此也包含空白单位正方形。

这就给出了一个单调栈的解决方案：这个栈包含若干个左上角顶点，其保持$x_u + S_u$单调减，特别的，在扫描中每次弹出栈顶的$x_u + S_u < x_v$的元素。每次只要检验那些满足$x_v - T_v \leq x_{top}$的(top, v)对即可。

注

以上解法是从我试题准备部分的solution中选出的解法2，原solution中还有一个暴力的$O(NX)$解法，其中X为坐标范围3000。

1.12 261D. Maxim and Increasing Subsequence

题目描述

数个询问，每次询问一个以N为周期的，长$N \times T$的数列$B \times T$的LIS长度。

数据规模

$N \times B_{max} \leq 2 \times 10^7, T \leq 10^9, N, B_{max} \leq 10^5$。

解法

首先，LIS的最大值为B_{max}。从而T的有效范围被限制在B_{max}中。也即，$T' \leq B_{max}$。从而$N \times T' \leq 2 \times 10^7$。接下来就可以暴力处理了。以$F_i$代表以值i为结尾的最长LIS，依次处理这$N \times T'$个值，在更新值时加入break，确保更新的每一步都会增大某个F_i。由于F_i包含最多N个有效值，每个有效值一开始是0，最后最多为B_{max}，因此总更新步数也在2×10^7范围内。

1.13 235C. Cyclical Quest

题目描述

有串S。多个询问：串S的多少子串与T_i循环同构？

数据规模

$|S|, \sum_i |T_i| \leq 10^6$。

解法

做出S的后缀自动机，对每个节点对应的最长串处理出在S中的出现次数。对于每个T_i，在自动机上走一遍（没有可用的儿子就像AC自动机一样找father），随后为了体现循环同构，还需要多走一个循环节，在这第二个过程中，如果走完一步后匹配长度至少为$|T_i|$，即把这个节点的次数加入答案（计数时要先把指针调整到最后的那个满足长度的祖先——否则会漏数）。

1.14 311E. Biologist

题目描述

有N个0/1值，将第i个值改掉需要V_i费用。有M个要求，每个要求要求将一个集合中的所有值改成0或1。如果满足了，得到W_i奖金，如果不满足，并且这个要求的$k_i = 1$，那么需要付出g费用。求最大收益。

数据规模

$N \leq 10^4, M \leq 2 \times 10^4$，每个集合的大小在10以内。

解法

首先如果$k_i = 1$，可以先付出g费用，然后将W_i加上g。

然后我们可以把“改掉”按照以下方式描述：给每个0/1值赋成0值或1值分别需要V_{i0}和V_{i1}费用，其中一个值是0，一个值是V_i。

现在出现了一个经典的最小割模型。为每个值建立一个点，这个点最终在S割则说明值取0——将它与T连V_{i0}的边，同理，与S连V_{i1} 的边。对每个要求建一个点，以要求改成0为例：从S向这个点连W_i，从这个点向集合中所有点连无穷大边，这样，只要有一个点到T割，这个点就分到T割，W_i的价值就被割掉了。使用任意一种最大流算法解决。

1.15 306D. Polygon

题目描述

构造一个N边形，各边边长不等，各角相等。

数据规模

$N \leq 100$。

解法

先做一个正N边形，然后随机将各边内移/外移。由于我实现时没有写过check，而是完全纯随机，在经过了调参以后才通过此题。

1.16 249E. Endless Matrix

题目描述

对于类似下表的无穷矩阵，求子矩阵和。

1	2	5	10	17	26
4	3	6	11	18	⋮
9	8	7	12	19	
16	15	14	13	20	
25	24	23	22	21	
36	⋯				

如果值至少是10^{10}，输出后10位，前加3个点。

数据规模

坐标范围10^9，询问数10^5。

解法

首先将询问差分，之后对两个坐标的大小关系进行讨论即可。问题就在于这道题的输出既要求取模以后的结果又要判断值是否大于$10^{10}-1$，比较麻烦，可能需要用到高精度。

有一个不需要高精度的方法：用一个同10^{10}的互质的数，如$10^{10}+1$，作为模数，再求一次答案，如果和10^{10}时的答案相同，就几乎可以确定答案在10^{10}内了。

1.17 295D. Greg and Caves

题目描述

有一个$N \times M$的网格。求合法的染色方案数。称一个二染色方案为合法，当且仅当：

1. 存在黑色格子的行是连续的一段，并且，这些行中每行恰有两个黑色格子。

2. 称这些行里两个黑色格子之间的白色格子为这个行的区间，那么，存在一个行，从这个行向上，上一行的区间是下一行的区间的子集，从这个行向下，下一行的区间是上一行的区间的子集。

数据规模

$N, M \leq 2000$。

解法

令$dp_{i,j}$为使用了i行，最下行区间长为j的方案数，我们发现这个值可以从$dps_{i,j}$，dp数组在行上的两阶前缀和转移而来。因此我们可以在$O(NM)$中处理出dp和dps数组。约定一个合法方案中，“腰部”为所有有着最长区间的行中，最靠下的那个。枚举腰部的位置r和腰部的区间长度l，则这一部分的答案为$\sum_{top=1}^{r} dp_{r-top+1,l} \times \sum_{buttom=r}^{N}(dps_{buttom-r,l} - dp_{buttom-r,l}) \times (M-l+1)$。在列上对dp和dps做前缀和即可。

注

对dps做前缀和只是为了理解方便，实际上，由于$dp_{i,j} = dps_{i+1,j}$，只需要对dp做前缀和即可。同时所有的操作可以在一个数组中完成。

1.18 286D. Tourists

题目描述

在平面直角坐标系上，在某时刻T_i会出现线段$(0, L_i) - (0, R_i)$。M个询问，若在Q_i时刻，两个人同时分别从$(-1, 0)$与$(1, 0)$出发，以$(0, 1)$每秒的速度移动，那么在多少时间内，这两个人之间的视线被至少一个线段挡住。

数据规模

$N, M \leq 10^5$，$0 \leq L_i, R_i, T_i, Q_i \leq 10^9$

解法

把竖直方向上的坐标离散化。离散化后的$O(N)$段区间，每段都有一个出现时间，这个时间可以用std::set或线段树处理出来。每一段对于答案的贡献是一个关于Q的分段函数，大致的具有下面的形式。

$$F_i(q) = \begin{cases} 0, & \text{if } q \leq s_i - x_{i+1} \\ q - (s_i - x_{i+1}), & \text{if } s_i - x_{i+1} < q \leq s_i - x_i \\ (x_{i+1} - x_i) & \text{if } q > s_i - x_i \end{cases}$$

这个函数可以看成两个函数的和：

$$Y_{x_0,k}(x) = \begin{cases} 0, & \text{if } x < x_0 \\ k(x - x_0) & \text{Otherwise} \end{cases}$$

$F_i(q) = Y_{x_i,1}(q) + Y_{x_{i+1},-1}(q)$

将所有Y函数按照其x_0排列，预处理第二个case的前缀和。对每个q_i，二分出哪些Y函数取到第二个case，直接将q_i代入那个前缀和即可。

1.19 258D. Little Elephant and Broken Sorting

题目描述

对一个N排列执行M个操作(i, j)，每一个操作将有一半几率将排列的这两位互换。求最后的排列的逆序对数期望。

数据规模

$N, M \leq 1000$。

解法

用$dp_{i,j}$表示当前第i位大于第j位的概率，每次操作$O(N)$维护即可。

1.20 273D. Dima and Figure

题目描述

有一个$N \times M$的网格。求合法的染色方案数。称一个二染色方案为合法，当且仅当：

1. 至少有一个黑色格子。
2. 黑色格子是四联通的。
3. 任意一对黑色格子四联通下的距离等于曼哈顿距离。

数据规模

$N, M \leq 150$。

解法

感性认识一下，最后的方案一定是一个"凸"而"无洞"的图形。也就是说，其左右上下的边界都是单峰的。

以dp_{i,j_l,j_r,o_l,o_r}表示考虑了i行，最后一行在(j_l, j_r)，并且左边界经过/未经过(o_l)它的峰，右边界经过/未经过(o_r)它的峰。我们发现每次用来转移的是dp_{i,o_l,o_r}这个二维数组中的一个子矩形，所以只要在处理了每个i后，对它的j_l, j_r两维做二维前缀和，就可以把转移做到$O(1)$。总的复杂度是$O(NM^2)$。

1.21 283E. Cow Tennis Tournament

题目描述

有N个选手，分别有s_i的姿势水平。最初姿势水平高的选手总能打败姿势水平低的。

现在FJ认为这样太没有挑战性，会进行K个操作L_i, R_i。每个操作对于所有二元组$(x, y), (s_x, s_y \in [L_i, R_i])$, 翻转这两个选手的胜负结果。最后求有多少个无序三元组，不能决出最强选手。

数据规模

$N, M \leq 10^5$。

解法

首先取这些三元组的补，也即存在一个最强选手。这部分可以通过计算出每个选手的胜利场次数，然后将$C^2_{win_i}$求和得到。接下来的问题是如何求出每个选手的胜利场次数。

首先把所有的选手按照姿势水平的高低排序，这样最初，是矩阵的上三角为1，下三角为0，对角线不定义。每个翻转操作可以看成一个子矩形修改。每个选手的胜利场次数，就是s_i更大的里面，翻转了奇数次的个数，以及更小的里面，翻转了偶数次的个数。使用线段树维护翻转的奇偶性，使用扫描线解决，每个操作对应两个区间修改，每个选手对应两个区间查询。

1.22 354D. Transferring Pyramid

题目描述

有一个N阶三角形点阵。有K个点需要覆盖。可以用3的代价覆盖一个点。可以用$2+\sum_{i=1}^{q} i$的代价覆盖一个边长为q的子三角形点阵，其中子三角形底边与三角形点阵的底边重合。允许覆盖不需要覆盖的点。求需要的最小代价。

数据规模

$N, K \leq 10^5$。

解法

首先答案不会超过$3K$。从而我们只需要考虑那些$2+\sum_{i=1}^{q} i < 3K$的q。也即$maxq = O(\sqrt{K})$。这样一来就可以使用$O(Nmaxq)$（也即$O(N\sqrt{K})$）动态规划解决了。

1.23 305D. Olya and Graph

题目描述

有一个有向图，现要将它加边补全为一个合法图。合法图的定义如下：

1. 每一个点都可以到达(直接或间接地)所有编号更大的点。
2. 所有边从小编号点向大编号点连。
3. 如果两个点的编号差在K以内，则最短路长等于编号差。
4. 如果两个点的编号差超过了K，则要么最短路长等于编号差，要么等于编号差减K。

求补全方案数。

数据规模

$N, K \leq 10^6, M \leq 10^5$。

解法

由(1.)(2.)最后的图中，至少有$N-1$条边$(1,2),(2,3),\cdots\cdots,(N-1,N)$。除此以外只能有编号跨度为$K+1$的边，且不能有两条跨度为$K+1$的边起点编号相差超过$K$。判掉无解情况。

枚举第一条跨度为$K+1$的边的位置i，这样一来只允许了起点在i和$i+K$之间的跨度为$K+1$的边，并且必须有边$(i,i+K+1)$，判掉无解，这一部分的答案就是2的可取可不取的跨度为$K+1$的边的个数次方。

1.24 261E. Maxim and Calculator

题目描述

最初，有两个Memory：$a=1,b=0$。在一个步骤中可以执行以下两个操作之一：

1. $a'=a*b$;
2. $b'=b+1$。

称x为p步可达的，当在p个步骤内，可以让a计数器的值达到x。

统计$[l,r]$中p步可达的整数数目。

数据规模

$2\le l\le r\le 10^9$，$1\le p\le 100$。

解法

我们记一个操作序列中的操作2的个数是p_2。那么，最后a的值在素因数分解后不会有超过p_2的素因子。并且，a可以被分解成不超过$p-p_2$个p_2的数的乘积。

考虑到p的范围很小，上面的那条性质将需要考虑的数缩减到“最大素因子不超过p”的范围。在10^9的范围内，最大素因子不超过100的数的数目在3×10^6。

首先DFS做出所有的这样的数，排序。枚举p_2。显然每轮，需要用a_0的步数去更新$a_0\times p_2$。由于已经排序了，可以用类似于two pointer的技术实现均摊$O(1)$。在每次更新后，使用新的步数加上当前的p_2来更新这个数的所需步数。p轮后，就得到了每个数的步数。然后就可以在3×10^6中统计出答案。

1.25 253E. Printer

题目描述

有N个任务：每个任务有三个参数t_i（接收到任务的时间），s_i（需要执行的时间），p_i（优先级）。保证优先级各不相同。printer在每秒，将从已经接收到的，并且尚未执行完毕的任务中，选择优先级最高的那个，执行一秒钟这个任务，同时这个任务的剩余时间减一。

然而，现在在N个任务中，有一个任务的优先级是未知的。但是知道了它的完成时间。

求：这个任务的优先级（输出任意解），每个任务的完成时间。

数据规模

$N \leq 5 \times 10^4$。

解法

首先的，完成时间关于优先级的单调性很明显，所以可以二分答案。本质不同的优先级数目在$O(N)$，这一步带来一个log。

现在只需要考虑在已知所有优先级的情况下模拟这个过程。显然，只要用std::set储存所有已接受任务，然后每一步执行到接受新任务或完成当前最优先任务为止。步骤数最大为$2N$。因此每个模拟过程是$O(NlogN)$的。总的是$O(Nlog^2N)$。

1.26 316E3. Summer Homework

题目描述

维护一个数列。

1. 单点赋值。
2. 区间加。
3. 区间按斐波那契数列为权值求和。

数据规模

$N, M \leq 2 \times 10^5$。

解法

使用线段树，在每个节点上保存该区间按斐波那契数列$(1, 1, 2, 3, \cdots\cdots)$的加权和和按右移一位的斐波那契数列$(0, 1, 1, 2\cdots\cdots)$的加权和。由于$Fib_{i-1} \times Fib_j + Fib_i \times Fib_{j+1} = Fib_{i+j}$，从而就可以在维护二元组性质的前提下实

现左移右移。那么区间加就可以用简单的打标记，单点赋值就可以用推标记加赋值实现。

1.27 243C. Colorado Potato Beetle

题目描述

有一个棋盘，某人从某个格子中心出发，走了N段路。经过的格子就称作被保护了。如果一个格子不能仅四联通地通过未被保护的格子到达无穷远处，称其为除了虫的。求除了虫的格子数目。

数据规模

$N \leq 1000$。

解法

因为N很小，直接离散化+Floodfill即可。

1.28 256D. Liars and Serge

题目描述

有N个人，要么是诚实人要么是骗子。现在你问：这里有多少个诚实人？诚实人回答正确的答案，骗子在1到N之间独立地随便选择一个正确的答案以外的答案。求有多少个回答序列，使得恰好有K个人明显是骗子。

数据规模

N是2的幂。$1 \leq K \leq N \leq 256$。

解法

[处理到的数值][使用的人数][判定为骗子的人数]作为状态，$O(N)$的转移，从而$O(N^4)$的DP是显然的。然而N太大了。注意到N是2的幂，(N, K)二元组的取值范围为511种，所以可以$\sum_{i=0}^{8} 16^i$地打表。

1.29 303D. Rotatable Number

题目描述

Rotatable Number指的是一个长度为N的b进制数，将其乘1到N以后得到的数就是将这个数旋转0到$N-1$次后的结果（不必严格对应）。给

定N, x，在x以下找出一个最大的b，使得存在至少一个Rotatable Number。

数据规模

$N \leq 5 \times 10^6, x \leq 10^9$。

解法

Wikipedia教导我们如果一个数是Cyclic的（也即本题中Rotatable），则可以表示成$\frac{b^N-1}{N+1}$的形式。从而b应当是$N+1$ 的一个原根。从$x-1$开始向下，最多找$N+1$个数，分别判断是不是$N+1$的原根。预处理$N+1$的分解质因数的结果即可每次快速测试。

1.30 266E. More Queries to Array...

题目描述

维护一个序列。
1. 区间赋值。
2. 区间按据区间头距离的K次方为权做加权和。

数据规模

$N, M \leq 100000, 0 \leq K \leq 5$。

解法

由于K很小，我们只需要在线段树的每个节点储存从0到5的各个K值下的答案，然后利用二项式定理就可以实现将i^K变换成$(i+d)^K$了，总的是一个线段树的时间复杂度乘上K^2的常数。

1.31 332E. Binary Key

题目描述

有一个字符串p，构造一个长为k的0/1串。将这个串连续写若干遍直到长度达到p的长度，记为K。对p的每一位，如果K的对应位置是1，保留它，否则，去掉它。目标是使得最后的串为一个给定的串s。要求最小化这个串的字典序。

数据规模

$|p| \leq 10^6, |s| \leq 200, k \leq 2000$。

解法

枚举在每一个周期中，有多少个1。接下来就是匹配各个列（p按k为周期分，s按i为周期分）了。拿一个指针扫一下碰到匹配的就取来就可以了。

1.32 273E. Dima and Game

题目描述

纸上有一些区间(L_i, R_i)。两人轮流行动，每次选择一个大小至少为3的区间，然后把它变成$(L_i + [\frac{R_i - L_i}{3}], L_i + 2 \times [\frac{R_i - L_i}{3}])$或$(L_i, R_i - [\frac{R_i - L_i}{3}])$。不能移动者输。现在要写下一列$N$个区间，每个区间都要满足$1 \leq L_i < R_i \leq P$。求先手必胜的方案数。对$10^9 + 7$取模。

数据规模

$N \leq 10^3, P \leq 10^9$。

解法

我们可以发现，只有区间的长度$R_i - L_i$是有意义的。因此问题可以被改写成以下形式：有一列数P_i，每次选择一个至少为3的数，变成$[\frac{P_i}{3}]$或$P_i - [\frac{P_i}{3}]$。首先，由于决策的数目只有两个，所以每个状态的SG函数只可能为0, 1或2。然后，由于每次操作把数按照比例减小，所以SG函数的段数在log等级，可以打表。然后就可以处理出具有某SG函数的区间的种数。这之后只要$O(N)$的DP就可以得到答案。

1.33 333C. Lucky Tickets

题目描述

如果一个八数字序列可以通过加入加减乘和括号来得到K的结果，称其为幸运的。

要求构造M个幸运序列。

数据规模

$0 \leq K \leq 10^4, M \leq 3 \times 10^5$。

解法

用四个数字可以拼出平均60种数，用前四位拼数，按照这个数和K的大小关系，在中间加上加号或减号，在后面四位写差。

1.34 240F. TorCoder

题目描述

维护一个字符串S。只有一种操作(L_i, R_i)。如果子串$S_{(L_i,R_i)}$可以通过重排构成一个回文的话，把这个子串重排成一个字典序最小的回文。在M个操作后输出当时的串S。

数据规模

$|S|, M \leq 10^5$。

解法

如何判断是否可以变成回文？只要知道各种字母的出现次数。怎样得到最小字典序的回文？条件同上，操作是$O(|\Sigma|)$个区间赋值。因此，我们只需要在进行区间染色的时候，维护区间中各种颜色的出现次数即可。而这用线段树就可以办到。

1.35 305E. Playing with String

题目描述

有一个字符串，两人轮流行动，每次可以选择一个字符串的一个非边缘字符，要求它两侧的字符一样，然后从这个位置，将他所在的字符串切成三片。不能行动者输。判断胜负，并且如果先手胜，给出一个最小的第一步的决策。

数据规模

$|S| \leq 5 \times 10^3$。

解法

事实上有意义的只有连续的各段可选字符。因为在某一段可选字符中选一个分开字符串，不会影响其他段。所以问题变成：有多个数，每次选择一个大于零的i，变成$i-2$或j与$i-3-j$（$j \leq i-3$）。不能行动者输。然后就可以$O(N^2)$预处理SG函数，$O(N)$回答了。

1.36 235D. Graph Game

题目描述

在一个环套树上做点分治，每个子过程的代价为这个连通块的点数，每次随机选择中心点。求总代价期望值。

数据规模

$N \leq 3 \times 10^3$。

解法

首先考虑这个问题在树上的版本。

有以下公式：$ANS = \sum_i \sum_j \frac{1}{Dist(i,j)+1}$。下面是它的解释。我们定义点$i$的领域为，删除了$i$的那个子过程所处理的点集。那么，$i$会对所有处于其领域中的点带来1的删除时间贡献，也即对总代价带来1的贡献。我们考虑何时j才会处于i的领域内：在删除i的那个时刻，j仍然和i相连。从而，i是从i到j的路径上，第一个被删除的。这一事件发生的概率是1除以涉及的点数：$\frac{1}{Dist(i,j)+1}$。这样就得到了本段开头的那个公式。

接下来把它推广到环套树上。如果两个点在同一棵树上，那么上面的式子显然仍然成立。如果两个点在不同的树上，那么，就出现了两条可能的路径。从而，在删除i时j仍然在其连通块中的概率是：由路径一相连的概率，加由路径二相连的概率，减去由两条路径相连的概率。也即$\frac{1}{Dist_1(i,j)+1} + \frac{1}{Dist_2(i,j)+1} - \frac{1}{Dist_{Cycle}(i)+Dist_{Cycle}(j)+|Cycle|}$。

DFS找环，DFS计算每个点所属的树根和在这棵树上的深度，同时预处理为LCA准备的倍增表。然后枚举每个点对处理即可。

1.37 342D. Xenia and Dominoes

题目描述

一个$3 \times N$的格子，有一个是要求空出来的，有一些是障碍物。现在要在空格子中放上一些1×2的骨牌。要求：放满所有空白的位置，并且，至少有一个骨牌可以移动（滑到空格子中）。求摆放方案数。

数据规模

$N \leq 10^4$。

解法

先考虑没有“至少有一个骨牌可以移动”要求的方案数如何求：由于只有3行，所以状压DP就可以了。

接下来要考虑加上这个要求后的方案数：如果说空格子右边的骨牌是可以移动的那个骨牌，那么，我们就已经确定了一个骨牌的位置，这样的方案数目就是把这两个位置设置成障碍物以后的方案数。其他三个方向同理。然后容斥就可以了。

1.38 301C. Yaroslav and Algorithm

题目描述

有一种用于处理字符串的机器。这个机器接受一个字符串作为其输入，接受一个有序的字符串列表作为其参数，输出一个字符串。输入输出的字符串都只包含数字。列表里的字符串都满足以下规则：”[0-9?]*(< | >)>[0-9?]*”。这个机器的工作流程如下：读入字符串，保存在内存中，重复以下过程：从列表头到尾扫描，如果某字符串的前一部分”[0-9?]*”是内存中串的一个子串，在这个串的第一次出现的地方把它替换成后一部分”[0-9?]*”。随后如果中间是”<>”，结束，输出值是当前的内存。否则，回到这个循环的开头。如果没有匹配的串，结束，输出值是当前的内存。

现在给你若干组输入，内容都是正整数，要求输出一个字符串列表，使得在这些输入下，都能给出这个整数加一作为输出。

数据规模

$T \leq 100, N \leq 10^{25}$。

评分标准

1. 合法。
2. 列表元素数不超过50。
3. 对每组输入数据，答案正确。
4. 每组数据的循环轮次不超过200。

解法

以下是一个普适的输入列表。

0?? <> 1
1?? <> 2
2?? <> 3
3?? <> 4
4?? <> 5
5?? <> 6
6?? <> 7
7?? <> 8
8?? <> 9
9?? >> ??0
?? <> 1
?0 >> 0?
?1 >> 1?
?2 >> 2?
?3 >> 3?
?4 >> 4?
?5 >> 5?
?6 >> 6?
?7 >> 7?
?8 >> 8?
?9 >> 9?
? >> ??
>> ?

它的工作流程是这样的：在串头加问号，把问号移到串尾，把串尾问号变成双问号。以双问号作为前一个数字加一的标志，将其往左移动。这样，循环轮数是$O(logN)$级别的，满足要求。

1.39 309D. Tennis Rackets

题目描述

有一个边长为$N+1$的正三角形。每条边上有N个$N+1$等分点。其中，每条边两侧的各M个点是不可选择的。要求在每条边上选择一个点，使得它们构成一钝角三角形。求方案数。

数据规模

$N \leq 32000$。

解法

记三角形为ΔABC。约定：钝角三角形的钝角节点R在边AB上。且$|AR| \leq |BR|$。每个$|AR| < |BR|$的方案可以变化出6种对应的方案，每个$|AR| = |BR|$的方案可以变化出3种方案。

我们枚举$|AR|$的长度i（范围是$M+1..[\frac{N+1}{2}]$）和边AC上的节点P到A的距离$j = |AP|$（范围是$M+1..min(N-M, 2i)$）。接下来就可以计算出另一个节点Q的位置了：$|BQ| < \frac{(2i-j)(a-i)}{i+j}$。

这样一来是$O(N^2)$的。由于常数较小，而且时间限制比较长，所以可以通过。

1.40 301E. Yaroslav and Arrangements

题目描述

合法序列指的是一个循环序列，相邻两项相差1。并且，最小值等于第一个值。

好序列指的是一个单调序列，对其重排可以得到至少一个至多K个合法序列。

求长度不超过N，各项不超过M的好序列数目。

数据规模

$N, M, K \leq 100$。

解法

我们一层一层的构造合法序列。新的数可以放在那些相邻两个数相等的gap中，并且每个gap至少放一个。可以用组合数计算排列方案数。那么就可以用dp来做这个事情。三维分别是当前数，当前已用位置，当前gap数。在一个gap中放的多于一个数的部分就变成下一轮的gap。转移是直接枚举下一种数的个数。这样就可以100^4了。

2 第二部分(Codeforces)

2.1 309B. Context Advertising

题目描述

把一个词语序列里的一段连续的词语进行排版，目标是最大化使用到的词语数目。最多排R行，每行的相邻两个词语之间用一个空格隔开，每行的长度不超过C。求最多的使用词语数目。

数据规模

$N, R \times C \leq 10^6, \sum |S_i| \leq 5 \times 10^6$。

解法

先一遍two pointer做出每个词语作为行首可以在这行放几个词语，再倍增处理出以每个词语作为文首可以排版多少词语。

2.2 277D. Google Code Jam

题目描述

参加GCJ比赛，有N道题，有T时间。每个题有小数据和大数据。在过小数据之前不能交大数据。小数据和大数据分别有所需时间和得分。小数据因为是暴力，不会写错。大数据则有fst_i的几率写错（Fail System Test），但是由于是FST所以在比赛结束前是不知道有没有写错的。GCJ的排名方式是：先按照分数从高到低，同分按照最后一次正确的提交的时间从早到晚。因此你首先要最大化期望得分，在期望得分相同的时候最小化最后一次正确的提交时间（下称，罚时）的期望值。

数据规模

$N \leq 1000, T \leq 1560$。

解法

如果只需要求最大期望得分，可以按照背包来做，也即：dp[i][j]表示只用前i题，j分钟内的期望得分，转移有三种，不做，只写暴力，两个都写。

现在还需要计算期望罚时。首先容易注意到，所有的暴力应该在比赛的一开始写完。考虑有一个暴力在另一个题的标算之后写。如果在他们之后的那些提交中有一个正确，那么交换这两次提交的顺序不影响罚时。如

果这个标算写对了，交换也不影响罚时。写错了，那么把暴力放在前面写就可以减小罚时。从而，所有的暴力要放在所有的标算之前。同时又容易观察到，暴力的排列顺序不影响罚时（它们总是写对的）。

虽然暴力的排列顺序不影响罚时的期望，但是标算的排列顺序是会有影响的。假设在两个标算i,j之前的期望罚时是t_0，总时间为T，那么，先放i的期望罚时为$t_{ij}=fst_i\times fst_j\times t_0+(1-fst_i)\times fst_j\times(T+tb_i)+(1-fst_j)\times(T+tb_i+tb_j)$，先放$j$是$t_{ji}=fst_i\times fst_j\times t_0+(1-fst_j)\times fst_i\times(T+tb_j)+(1-fst_i)\times(T+tb_i+tb_j)$。

$$\begin{aligned}t_{ij}-t_{ji}&=fst_i\times fst_j\times(tb_j-tb_i)+(fst_i\times tb_i-fst_j\times tb_j)\\&=tb_i\times fst_i\times(1-fst_j)-tb_j\times fst_j\times(1-fst_i)\end{aligned}$$

从而，欲使$t_ij\le t_ji$，则应有$\frac{tb_i\times fst_i}{1-fst_i}\le\frac{tb_j\times fst_j}{1-fst_j}$。将所有题目按照$\frac{tb_i\times fst_i}{1-fst_i}$升序排序dp，那么就可以默认把新的题目的标算放在任务序列的最后了。既然已经确定了暴力和标算各自放在什么位置，就容易写出这样的dp转移：

$(i,st)\leftarrow(i-1,st)$

$(i,st+ts_i)\leftarrow[(i-1,st)_{score}+ss_i,(i-1,st)_{time}+ts_i]$

$(i,st+ts_i+ts_j)\leftarrow[(i-1,st)_{score}+ss_i+(1-fst_i)\times sb_i,ts_i+(i-1,st)_{time}\times fst_i+(st+ts_i+ts_j)\times(1-fst_i)]$

以上，(i,j)指前i题j时间的$[score,time]$二元组，"$\leftarrow$"为，用右侧的值更新左侧的值。$ss_i,sb_i,ts_i,tb_i,fst_i$分别指两个数据的分数，需要写的时间和FST概率。

2.3 341E. Candies Game

题目描述

有N个数，每次操作指定两个数x,y（$x\le y$），然后从y中去掉x，加到x中，也即，$x'=2x,y'=y-x$。最后要令所有数里恰有两个不为0。

数据规模

$N\le10^3$。所有数的和和操作允许的步数在10^6。

解法

如果本来就只有一个数不为零，宣称无解。

如果有至少三个数不为零，记为$a\ge b\ge c>0$。则以下过程可以将b变成$b\%c$。特别的，如果每次的c取成非零值中的最小值，那么就可以得到一个可以在有限步内结束的方法。这个过程是这样的。现在我们要使b变成$b\%c$，这就是说，我们要从b 中去掉c的几倍。在只有b和c的时候，这并非总是可行的，实际上只有在$[\frac{b}{c}]=2^k-1$时才可行，因为每次操作都会令c翻倍。但我们还有a可供使用。以$[\frac{b}{c}]=5$为例，$5=2^2+2^0$，从而要进

行编号从0到2的三次操作，其中编号为0和2的操作中，$x = c, y = b$，其他操作中，$x = c, y = a$。这里，藉由a的存在，在不影响b的情况下实现了对c的翻倍。这样子，实际上在一个比b更大的a存在的时候，我们可以从b中减掉任意的c的倍数。

对于这个方法的操作复杂度，我并没有进行过仔细的论证，但在实际测试中，这个方法表现得很好，所有的测试数据的方案长度都在2×10^4以内。

2.4 323B. Tournament-graph

题目描述

构造一个N阶竞赛图，使得任何一个点都能在两步之内到达所有点。

竞赛图指的是一类简单有向图，对于任意互异两点u, v，要么有边$< u, v >$，要么有边$< v, u >$。

数据规模

$N \leq 1000$。

解法

如果i阶和j阶是可构的，那么可以导出一个$i + j - 1$阶图的构造方法：将i阶图中的某一个节点换成j个节点，这j个节点内部按照构j阶图的方式构造，对外，当做被替换的那个节点，按照原来的图中边的方向连。

容易看出，一个三元环就是一个合法的3阶图。这就是说，如果x阶图可构，对于任意正整数k，$x + 2k$阶图是可构的。具体的来说，选择两个点S, T，T向S连边，S向剩下的点连边，剩下的点向T连边，从而简化到$N - 2$阶的情形。现在只要对奇数和偶数分别寻找最小阶可行解。

奇数：$N = 1$是可行的。从而所有奇数阶都是可行的。

偶数：$N = 2$只有一种且显然不可行。考虑$N = 4$。显然，这个图至少要强联通，从而，最大出度点的出度不得为3。又由抽屉原理，最大出度至少为2。从而最大出度恰为2。我们设其为u，其两个出边指向v和w，其中v向w有边，剩下一个点为z。此时我们发现w还没有出度，于是只能有$< w, z >$。为了让w在两步内到达v，它的唯一后继z应当向v连边。至此，六条边都已被定向。在这个图中，v需要三步才能到达u，从而$N = 4$时不存在合法的方案。

下面给出出一个合法的六阶图的构造方案。这个图可以被分成两个部分，每个部分内部是一个三元环，分别记为环(a_1, a_2, a_3) 和环(b_1, b_3, b_2)，注意，在两个环中，边的方向和编号的方向在一个部分是同向的，在另一个部分是异向的。对于点a_i和b_j，如果$i = j$，有边$< a_i, b_j >$，否则，有

边$< b_j, a_i >$。读者可以自己验证它的合法性。提示，每个三元环内部的点是等价的，因此只需要检查两个节点。

2.5 248E. Piglet's Birthday

题目描述

有N个架子，最初分别有A_i个满的蜜罐。然后Q个操作，从u_i架子等概率随机地拿下k_i个蜜罐（保证当时至少有足够的蜜罐），然后吃掉其中所有的蜜（可能没有蜜），然后放到v_i架子上。

每个操作后回答：没有满的蜜罐的架子的个数的期望值。

数据规模

$N, Q \leq 10^5, A_i \leq 100, k_i \leq 5$。

解法

第一个观察：每一时刻的每个架子上的蜜罐数目是确定的。

第二个观察：从不会有满的蜜罐被放到架子上，从而，每个架子上最多可能有A_i个满的蜜罐。

随着操作，维护二维实数数组$p_{i,c}$：该时刻，架子i上还有c个满的蜜罐的概率。每次操作$O(A_i k_i)$地更新即可。

2.6 316D3. PE Lesson

题目描述

有N个人进行传球，每个人最多参与$A_i \in \{1, 2\}$次传球。一次传球是一次交换。人和球都是有编号的，最后按人的编号数下来，球的编号为一个排列，求有多少种可能的最终排列。

数据规模

$N \leq 10^6$。

解法

我们考虑最后的那个排列，作为一个排列的结构。这个排列可以被拆成一系列循环的乘积。为了让传球次数尽量少，所有的传球应当都在循环的内部。因此我们只需要考虑循环是如何构成的。经过分析，一个长度至少为2的循环中，除了两个点的传球次数为1以外，其他的传球次数都是2，从而：每个循环中最多有两个限定一次传球的人。

首先，我们把所有限定一次传球的人拿出来。把他们划分成若干个集合，每个集合的大小不超过2。对于a个这样的数，我们对第a 个有两种处理方式：单独成集，这部分的方案数$p_1(a) = p(a-1)$；和前面的某个元素构成一个集合，这部分的方案数$p_2(a) = (a-1)p(a-2)$。从而有递推式$p(a) = p(a-1) + (a-1)p(a-2)$，边界条件$p(0) = p(1) = 1$。

然后我们处理所有限定两次传球的人。假设现在已经有t个人，加进去一个新的人。要么，他单独成集，新开一个循环，要么，他加入某个循环。如果他要加入某个大小为sz的循环，那么他有sz个位置可以插入，所以，加入循环的方案数恰好为t。从而加入这个人的方案数为$t+1$。

这样一来，设cnt_1和cnt_2为两类人数，最后的答案是$(cnt_1+1)^{\overline{cnt_2}}p(cnt_1)$。

2.7 323C. Two permutations

题目描述

有两个N阶排列A_i和B_i。M个询问，在第一个排列中处于$[l_1, r_1]$中，第二个排列中处于$[l_2, r_2]$中的数有多少。强制在线。

数据规模

$N \leq 10^6, M \leq 2 \times 10^5$。

解法

这是一个二维数点问题。由于强制在线，使用函数式线段树解决。第i棵线段树中新加入的位置为$A_{B_i^{-1}}$，值为1。查询时在线段树$l_1 - 1$和线段树r_1上查询区间$[l_2, r_2]$。

注

对于写指针式线段树的选手，如果每次新开节点时使用new可能导致MLE，原因不明，猜测是动态使用内存时产生了未利用的碎片空间。解决方案是预先开一个结构体数组，每次从这个结构体数组地址加一个偏移量（计数器）作为结构体指针。

2.8 343E. Pumping Stations

题目描述

在一个N个点的赋权无向完全图中求最大哈密顿路径。这个图中，u和v中间的边的边权为某指定的N阶图G中，编号同样为u 和v的最大流值。

数据规模

$N \leq 200$。

解法

这道题需要用到任意点对间的最大流值。结论：对于每一个无向图，存在一个等点数的无向树（Gomory-Hu树），使得编号相同的一对点在这两个图上的最大流相等。从而，两个点的最大流值等于树上路径最小值。

下面要构造最优解和其方案。考虑Gomory-Hu树上权值最小的边，我们切开它，递归的在两个子树中计算最优解，然后把两个解连在一起。这样最小边只对连接处有影响。考虑有至少两条路径经过这条最小边的情形，我们发现交换两个路径的一对异侧端点以后，答案不会更劣（当前连通块里的其他边的边权y都至少是这条边边权w这么大，所以$min(y,w) \leq y$）。从而答案可以改进，从而至少有一个最优解，每条边恰作为一条路径的瓶颈。从而最优解的值就是Gomory-Hu 树上的边权和。

注

关于Gomory - Hu树的构建方法，参见ZJOI2011《最小割》的题解。

2.9 267C. Berland Traffic

题目描述

构建给定无向图的一个最大流，使得从S到T的每条路径上的流量（计方向的）之和相等。

数据规模

$N \leq 100$。

解法

“每条路径”的和相等，这个概念类似于物理中看到的“保守力”，从而提示我们设置“势能”。那么这就是一个高斯消元的问题，各变量为各点势能，方程为各普通顶点上的流量平衡条件。特别规定，点S和点T之间的势能差为一常数。最后根据各边上的流量使用情况，对整个流进行等比例缩放。

2.10 240E. Road Repairs

题目描述

一个有向图，边有可用的和不可用的，可以花费1的代价将一条边变成可用的。要求花最少代价使得从点1出发可以通过可用的边到达所有点。

数据规模

$N, M \leq 10^5$。

解法

这个问题等价于以下问题：在边权只有0和1的边上求以1为根的最小树形图。

最小树形图是一个经典问题。使用朱刘算法可以解决。普通的朱刘算法在最坏情况下是$O(NM)$的，但是经过左偏树优化后可以做到$O(MlogN)$。

O(MlogN)的实现

需要的数据结构：

1. 左偏树（或其他支持标记的可并堆）；

2. 并查集；

3. (非高级数据结构)Info。Info有以下参数：类别标志(值为-1, 0或1)，费用，使用次数标志，两个操作量，类型为Info指针。类别标志为0时，说明这个Info的费用是读入的。类别标志为-1时，说明这个Info的费用是由两个Info相减而来，其中这两个Info是用指针的形式记录的，也就是操作量指针。类别标志为1时，说明这个Info的费用是由两个Info相加而来。有了类别标志，就可以实现边权值的加减同时又能输出方案。

最初，有N棵左偏树，每棵左偏树描述了一个点的所有尚未处理的入边。左偏树中的每个元素是一个int和一个Info指针的pair，描述了一条起点在int，边权信息由Info指针描述的有向边。有两个并查集，最初都是初始化成所有元素自成一集的状态。把这两个并查集称为W 和S。 接下来的处理过程中会有缩环，S并查集描述的是，原来的每个点在新图中被缩到哪个点，S并查集中的集合数目等于当前图中的点数。这个处理过程一条一条的加入边，W并查集描述的是，原来的每个点在新图的基图中处于哪个连通分量，W并查集中的集合数目等于当前图的基图的连通块数，或当前森林中的树数。之所以说是当前森林，是因为最后要构成一个有向树，所以中间产物是介于无边图和有向树间的有向森林。

算法的工作流程如下：每次，找到有向森林中的一个有尚未处理入边的树根（在本题中，由于要求以1为根，所以这一步不能找1）。在其中找到权值最小的边。我们试着加入这条边。这将产生三种结果。

1. 当这条边的起点和终点在S并查集中相连，则引入了自环。那么这条边需要被丢掉。丢掉这条边，什么都不做。

2. 否则，当这条边的起点和终点在W并查集中相连，则引入了环，需要缩环。通过记录每个点原来的入边，可以方便地找出这个新环。（不要忘了，这条边的终点是某一棵树的树根）执行缩环。把每个环中点的左偏树中的每个元素减掉其原来入边的Info（为此我们需要对左偏树实现懒标记），把所有左偏树合并作为新的点的左偏树，同时在S并查集中合并涉及到的集合。

3. 否则，这条边可以直接加入。在W并查集中合并即可。

每次我们选取一条边时就将这条边的pair中的Info指针指向的结构体的使用次数标志加一。每次我们需要对边权信息进行运算，就新建一个类别标志为-1或1的新Info，将其指针压入一个vector（这个vector的初始值是所有读入的边）。这样一来，vector中的每个Info指针的结构体的两个运算量的指针，在vector中都出现在它之前。所以为了输出方案，在树形图做完以后，将vector中的所有Info指针倒序地遍历一遍。如果其类别标志不是0（换句话说，不是vector开头的M个），就进行一次“推标记”。最后检查vector头的M个Info指针，如果其费用是1，并且使用次数标志为1，把它加入方案。

注

由于CF上的数据比较弱，这道题的数据被设计成实现为$O(NM)$的朱刘算法就可以通过。

参考文献

Tarjan R. E. "Finding Optimum Branchings" (1975) 28-31.

2.11 329E. Evil

题目描述

求平面点集的曼哈顿距离最大权哈密顿回路。

数据规模

$N \leq 10^5$。

解法

首先做这样一个约定：所有点的横坐标不同，纵坐标不同。如果不成立，可以通过加上一个或多个eps来使其成立而不影响答案。

观察曼哈顿距离回路总长的计算式，我们会发现每个点的两个坐标各出现了两次。如果单独把某类坐标拿出来，我们希望这$2N$个（N 个坐标，出现两次）坐标中，较大的N个在所在的那个绝对值括号中是较大的那个（得到+1的系数），较小的N个是绝对值括号中较小的那个（得到-1的系数）。这是答案的理论上界。下面将讨论何时可以达到这个理论上界。

做X和Y坐标中的中位数。这两条直线将平面分成四个部分，以下称这两条直线为轴X'和轴Y'，并用象限来称呼这四个部分。

当N为偶数时，轴上没有点。由于轴X'的两侧点数相同，轴Y'的两侧点数相同，可以得到第一象限和第三象限的点数相同，第二象限和第四象限的点数相同。将第一象限的点和第三象限的点间隔一个放一个构成一个环，这个回路的长度能达到上面的理论上界。另一组象限同理。那么，只要这两组象限中有一组是空的，那么就只有一个连通分量，整个问题的答案就可以取到其理论上界。否则，需要对其进行调整，将两个环并成一个。将两个环并成一个环的常见方法有：交换一对点的后继。假设我们交换了第一象限和第二象限的一对点的后继，那么，现在在Y'轴的两侧各恰存在一个点，和其左侧的点和右侧的点都有连边。那么，这两个点的X坐标对答案的贡献就没有了，从而答案从理论上界中减去了$2(X_{right} - X_{left})$。其中right是某个Y'轴右侧的点，left是某个Y'轴左侧的点。当把这两个点取成最靠近Y'轴的那两个时，这个减去的值最小，从而答案最大。这是交换的一对点在X'轴同侧的情况，在Y'轴同侧的情况同理，只是所有的X坐标换成Y坐标。

当N为奇数时，两条轴上各有一个点。这又分为以下两种情况：

1. 两条轴上的点是同一个。也即，两个坐标的中位数是同一个点。把剩下的点按照偶数时那样配成一或两个环。然后把中心点插入到任意一条边的中间。和上面同样，这时候如果发现环只有一个，那么就能取到理论上界，否则就需要调整，显然我们只要检查中心点的两个坐标和最靠近两条轴的那四个坐标的差就可以了。

2. 两条轴上的点是不同的。比如说这两个点分别存在于两轴正半轴，那么第一象限的点比第三象限的点少一个。将一三象限连接以后，在第三象限留着两个“线头”，将它们分别和两条轴上的点相连。和偶数一样将二四象限做环，然后从二四象限的环中拆掉一条边。现在：在第二象限，第四象限，两轴正半轴都存在一个“线头”，将轴X'正半轴上的点和第二象限的点相连，将轴Y'正半轴上的点和第四象限的点相连，这样就取到了理论上界。

2.12 303E. Random Ranking

题目描述

有N个人，分别有一个分数，第i个人的分数在区间$[L_i, R_i]$中均匀随机

取值。对所有的名次和所有的人求出，此人得到此名次的概率。

数据规模

$N \leq 80$。

解法

离散化。对于每一个人，对于离散化后的一个单位区间，计算他的分数落在这一区间时的各名次概率。容易在$O(N^3)$中计算出其他人中，有i个在这段区间前面，有j个在这段区间后面，有$N-1-i-j$个落在这个区间中的概率。那么这个情况就将给名次$i+1$到名次$N-j$贡献$\frac{1}{N-i-j}$的概率——既然包含本身在内，有$N-i-j$个变量是这个区间中的独立同分布随机变量，那么这些变量每个都有相等的概率去达到各个名次。这给出了一个$O(N^5)$的算法。

这个算法可以被优化到$O(N^4)$。发现每个段的情况可以被看成若干个二元一次多项式的乘积，处理第i个人的这一段时，那个$O(N^3)$的过程计算的，就是除了第i个二元一次多项式以外，其他多项式的乘积。那么只需要对每一段，都$O(N^3)$地预处理出来所有多项式的乘积，在做第i个人时，$O(N^2)$地除去对应多项式即可。

注

由于本题的精度要求严苛，所以以上的$O(N^4)$算法会由于使用了除法损失了精度而出现WA。(我的程序产生了2×10^{-6} 规模的误差)

最后提交的AC程序是$O(N^5)$的，用到了一些常数上的优化。比如，某些一次多项式中，某个未知数的系数是0，那么在之后的那些二重循环中，这一维的枚举范围就可以相应减少。

2.13 360E. Levko and Game

题目描述

有一张N阶$M+K$条边的有向图，你可以改变其中K条边的边权（在某个对每条边分别给定的范围）。目标是使s_1到f的最短路短于s_2到f的最短路。如果不能实现，目标是使这两者相等。如果还是不能实现，宣称无解。

数据规模

$N, M \leq 10^4, K \leq 100$。

解法

下述判断第一问（“短于”）的解法。

将所有可变边的边权设为其最大值。由s_1和s_2两源分别作最短路。对于任何可变边，如果s_1到其起点的距离更短，将其边权设为其最小值。再做两次最短路，判断答案。

显然，当s_1到其起点的距离更短时，将其边权变短，将只使s_1“得到实惠”——如果s_2也要经过这条边，就要先移动到这条边的起点。而从这个时刻开始，既然它们此时已经处于同一个点，那么后面部分的最短路将是相等的，因此这两条路径的长度大小关系就取决于这个前半段，而前半段，根据这个条件，将是s_1的优势。因此仅应减小这些边的边权，并且所有这样的减小都不会对我们的目标有害。

对于“不长于”一问，只需要把改变边权的判断条件中的“更短”变成“不更长”即可。

2.14 338E. Optimize!

题目描述

有两个序列。A的长度不短于B。对A的每一个len_B长的连续子序列判断：是否存在一个对B的打乱方式，使得两个序列的对应位置的数的和不小于H。

数据规模

$len_A, len_B \leq 1.5 \times 10^5$。

解法

B是死的，A是活的。所以先把每个B_i变成$H - B_i$，问题相应地变成：是否存在一个对B的打乱方式，使得A的连续子序列的每个位置的值至少为B中对应位置的值。

把所有这$2len_B$个值从小到大写成一列（一样大的B中的值在前）。容易证明，上面的问题的回答是“是”的充要条件是：对于序列的每个前缀，来自A的值的个数不超过来自B的值的个数。

那么离散化后这个问题就可以用线段树解决。操作有：单点加，单点减，查询全局最小前缀和。区间的最小前缀和可以通过额外维护区间和来维护。

2.15 285E. Positions in Permutations

题目描述

如果一个排列的位置i上的数是$i-1$或$i+1$，称其为好的位置。
求：恰有K个好位置的N阶排列的数目。

数据规模

$K \leq N \leq 1000$。

解法

一个N阶排列可以表示成这样的形式：一个$2 \times N$的表格，两行中每行都是1至N，在两行间连线，每个数的度数都是1。这么一来，好的位置的个数就等于斜率为1或-1的边的数目。N^2地dp做出在表中做i条斜率为1或-1的边的方案数，然后容斥即可。具体地，已经连了i条边的图中还需要连$N-i$条边，对应的方案数为$(N-i)!$。这$dp_{N,i}(N-i)!$种方案中可能出现重复，但是，由于这i条边的选择是完全不同的，所以一旦出现重复，那些方案中至少有$i+1$个好位置。每个$i+1$个好位置的方案都会以$i+1$种形式（也就是说：没有在dp中算进的那条边具体是哪条）出现在$dp_{N,i}(N-i)!$种方案中。这样，只要我们知道了$i+1$个好位置的方案数，就可以结合$dp_{N,i}$推出i个好位置的排列数。

而在一个N阶的排列中，好位置的数量最多为N。所以有N个好位置的方案数就是$dp_{N,N}$。所以就可以倒推到K个好位置的排列数。

2.16 238D. Tape Programming

题目描述

有一个每个字符是阿拉伯数字或左右箭头的序列。一个指针从序列头以向右1的速度移动。当它遇到序列外部，结束整个过程，当它遇到箭头，它按照箭头的方向改变自己速度的方向，移动一格。然后如果当前的符号还是箭头，就抹去之前位置的箭头。否则，它会遇到一个数字。指针按照自己的速度移动一格，然后输出原来的数字，并把这个数字减一。（如果是0，则从序列中抹去这个数字来代替减一）

多个询问：如果只考虑序列的某一段，在这一段上做以上过程时，共会输出各个数字多少次？

数据规模

$N, Q \leq 10^5$。

解法

观察这个大过程：只要序列长度不为零就一直从头进行这个过程。也即while(len)Scan(S)，S是序列，Scan是题目中描述的过程。

当指针第一次移动到某个位置的时候，这个位置以后的部分都是全新的。也就是说，我们可以从这个大过程中截取从这个时刻起的一段，它恰好就是询问需要的过程$Scan(S[i,j])$。可持久化线段树式地预处理出截止某时刻，某一位置被访问的次数，二分找出这个开始时刻。之后二分找出在这个时刻后第一次访问到目标区间以外的位置的时间。前缀和式地预处理出截止某时刻，输出各个数字的次数，相减即可。

2.17 241F. Race

题目描述

有一个网格式地图，有一些障碍物，有小写字母代表路口，有阿拉伯数字代表路，数字代表从这个格子移动到相邻非障碍格的时间。所有阿拉伯数字的连通块要么是横条要么是竖条。给出起点、终点、访问路口的顺序，求某时刻的所处位置。

数据规模

访问路口数≤ 1000。 地图边长≤ 100。 模拟时刻数$\leq 10^5$。

解法

路径是唯一确定的。暴力模拟即可。

2.18 243D. Cubes

题目描述

有由$1\times 1\times 1$的立方体搭成的结构。现在由无穷远处，按一个和地面平行的方向望向这个结构。可以看到多少个立方体？

数据规模

俯视图边长≤ 1000。 高$\leq 10^9$。

解法

把所有立方体投影到一个和视线垂直的面上。每个底面正方形会占用一段宽度。同一个底面正方形的各个立方体占用的是同一段宽度。将宽度

这维离散化。按前后顺序加入各个底面正方形。首先在这个范围里查询最小高度。如果小于这个底面正方形上的高度，统计答案。之后把这段区间的低于这个高度的拔高到这个高度。

使用线段树就可以实现。

2.19 251D. Two Sets

题目描述

将N个数分到两个集合S_1和S_2。设这两个集合的元素异或和为X_1和X_2（空集的异或和是零），目标是在最大化$X_1 + X_2$的前提下，最小化X_1。输出方案。

数据规模

$N \leq 10^5, Val \leq 10^{18}$。

解法

设N个数的异或和是X。那么如果X的某位是1，X_1和X_2的该位必然只有一个为1，对$X_1 + X_2$的贡献是一定的。而如果该位是0，则我们应尽量地令X_1的该位为1（或X_2的该位为1）。又因为又有X_1的最小化这个目标。所以就可以将整个题的目标记为：将X_2的各位尽量地设为1。

然而各位的优先级又是不同的。比如说X的那些1位的优先级就没有0位的优先级高。0位可以影响$X_1 + X_2$的值，而1位可以影响X_1的值。所以各位的优先级就是：X中较高的值为0的二进制位，X中较低的值为0的二进制位，X中较高的值为1的二进制位，X中较低的值为1的二进制位。按照这个顺序做高斯消元即可。

如何输出方案？储存一个出现过的数的列表，在消元数组中保存数在列表中的编号。将高斯消元过程中的每个异或操作变成，在列表中压入一个新数，将编号改成那个新数，对每个新数记下是由哪两个数异或得来。最后推一边标记即可。

2.20 351D. Jeff and Removing Periods

题目描述

对于一个数列有以下的一种操作：

先选择一个下标间距相等，并且对应元素值也相等的子序列，从数列中删去它，然后任意重排剩下的数列。

对于一个给出的数列，有若干组询问，每组询问形如，对序列的某个连续子序列，需要做多少次操作才能将数列变空。

数据规模

$N, M \leq 10^5$。

解法

首先，第一次操作后肯定是把剩下的序列排序最好。这样剩下的步骤就是序列中不同的数的种数。

现在问题转变成：

1. 区间中有多少种不同的数。

2. 区间中是否有一种数间距相等。如果回答是是，则答案是1.的答案，否则，答案是1.的答案加一。

预处理：每个点前面的数值相同的点。

将询问按右端点排序。用树状数组实现对于数组的区间加和单点询问。一共要实现两个树状数组：cnt和arith。前者的第l位是，左端点在l，右端点在目前处理到的位置的区间的1.问答案。后者是这个区间中满足间距相等的数的数目。这样，右端点每移动一次，就对应了这两个数组中的数个区间加操作。每个询问就对应了处理到的位置等于处理到的位置时，相应数组的下标为左端点编号的那位的值。总复杂度为$O(MlogN)$。

2.21 293E. Close Vertices

题目描述

有一棵赋权树。有多少点对满足：不考虑权值，距离不超过L，考虑权值，距离不超过W？

数据规模

$N \leq 10^5$。

解法

对于这类树上路径统计问题显然是要点分治。合并两个按某一维值排序的列表时，two pointer加一个Treap储存某一个列表内的另一维值即可完成信息合并。

2.22 348E. Pilgrims

题目描述

有一棵赋权树，其中有一些黑点。现在可以移除一个非黑点。如果某个黑点可到达的最远黑点距其的距离发生了变化，答案加一。求答案的最

大值。

数据规模

$N \leq 10^5$。

解法

白色叶子没有意义，移除它们。在残余的树上求出直径。把所有黑点投影到直径上。这时会有两种情况：一种情况是投影到树中心。这时，最远黑点可能出现在直径的两侧，因此只有黑点到树中心的路径上的点的被删去会导致这个点对答案有贡献。另一种情况是不投影到树中心，这时最远黑点可能出现在直径的另一侧。对每一侧处理出最靠近树心的投影点是哪个，第二类点的贡献只发生在被删除的点落在第二类点和直径另侧最内投影点的路径上时。那么现在问题成为这个形式：若干个路径加一操作，最后询问各点点权。使用LCT即可。

注

注意到所有路径的端点中要么有树中心，要么有两侧的最内投影点。可以进一步优化到线性级。

2.23 241E. Flights

题目描述

有一个赋权有向图。最初边权都是1。将一些边权改成2，令源到汇的所有路径长相等。

数据规模

$N \leq 1000, M \leq 5000$。

解法

类似267C，引入势能（距离标号）概念。用一条有向边相连的两个点的势能满足两个不等关系。使用差分约束系统求解。

2.24 257E. Greedy Elevator

题目描述

模拟一个过程。一架电梯，由一个简单算法控制。某一时刻的选择移动方向是由“以上面为目的地的乘客+上面的电梯外等待者”和下面的类

似参数决定，如果前一个大于等于后一个则往上，否则往下。给出每个人的开始等待时间，初始楼层，目标楼层，回答每个人的到达时间。

数据规模

$N, M \leq 10^5$。

解法

类似253E，模拟类问题只需要将动作相似的若干时间段合在一起处理就可以达到“事件数(通常是线性级)”乘“对数级或常数级”的复杂度。事件有：新等待者抵达并开始等待和乘客到达目的地两种。用两个set保存上方的和下方的等待者，用两个set保存目的地在上方的和下方的乘客即可。前两个set用到达时间做关键字，后两个set用目标楼层做关键字。

2.25 268D. Wall Bars

题目描述

统计满足以下性质的N元序列的个数：

1. 每个元素在1到4之间。

2. 存在至少一个子序列，每项的值相等，第一项选自原序列前H项，最后一项选自原序列后H项，相邻两项在原序列中的位置的下标差不超过H。

对$10^9 + 9$取模。

数据规模

$N \leq 1000, H \leq 30$。

解法

使用DP解决：

$dp[i][j1, j2, j3][k]$代表当前处理了i项，除了第i项以外，剩下的有可能成为可行子序列的元素上一次出现的位置距离i的位置的距离为$j1, j2, j3$（未出现过按照最后一次出现在0位置），然后若k为$true$，则说明第i项的元素已经不可能成为可行子序列的元素（也即，在之前的某一段中，最近的两个该元素相邻超过H项）。

作为优化，约定j的取值范围为$1..H-1$和$+\inf$（储存作H），则$j1 \leq j2 \leq j3$。

边界：$dp[1][1, 1, 1][false] = 4$。

目标值：所有$j1 \in (1..H-1)$或$!k$的$dp[N][j1, j2, j3][k]$。

转移：
(符号$++x$代表$min(x+1,H)$)
$dp[i][j1,j2,j3][k] \to dp[i+1][++j1,++j2,++j3][k]$
$dp[i][j1,j2,j3][false] \to dp[i+1][1,++j2,++j3][j1==H]$
$dp[i][j1,j2,j3][false] \to dp[i+1][1,++j1,++j3][j2==H]$
$dp[i][j1,j2,j3][false] \to dp[i+1][1,++j1,++j2][j3==H]$
$dp[i][j1,j2,j3][true] \to dp[i+1][++j2,++j3,H(+\inf)][j1==H]$
$dp[i][j1,j2,j3][true] \to dp[i+1][++j1,++j3,H(+\inf)][j2==H]$
$dp[i][j1,j2,j3][true] \to dp[i+1][++j1,++j2,H(+\inf)][j3==H]$

2.26 254D. Rats

题目描述

有一个网格，每个格子要么是障碍，要么是空地。空地可能有老鼠。现在要在两个空地上放炸弹。一个炸弹可以炸掉与其距离不超过D 的老鼠。现要炸掉所有老鼠。构造一个方案或宣称无解。

数据规模

$N, M \leq 1000, K \leq 8$。

解法

随便挑一只老鼠。它肯定要被炸掉。从这只老鼠的格子开始做D深度BFS，在被访问到的格子中至少要有一个炸弹。枚举是哪个。

假设已经放了一个炸弹，检查是否还有存活的老鼠。若无，随便找个第一个炸弹所在格子以外的空格子放第二个格子。若有，随便挑一只存活的老鼠，做BFS，再枚举炸死它的炸弹在哪里。再BFS一遍检查是否合法。

一次BFS最多访问$2D^2$个格子。所以最多检查$4D^4$种组合，时间开销是$8D^6$。

注

关于如何随便找一个存活的老鼠：保存每个老鼠所在格子被炸了几遍，搞一个存活老鼠的set。

2.27 269D. Maximum Waterfall

题目描述

平面上有一系列和X轴平行的，且相互之间没有公共部分的线段。称两个线段可以完全互相看到，当且仅当：两个线段在X轴上的投影有长度非零的公共部分，并且不存在一个Y坐标介于两者之间的线段与这两个线段都完全互相看到。

两个可以完全相互看到的线段的流量定义为X轴上投影公共部分的长度。

在无穷低处有无穷长线段T，在无穷高处也有无穷长线段S。定义一条路径为线段序列$S, s_1, s_2, ..., s_k, T$，序列中的任意相邻两线段完全互相看到，这条路径的流量定义为所有相邻两线段流量的最小值。

求，所有路径的最大流量。

数据规模

$N \leq 10^5$。

解法

容易得到一个$O(N^2)$的动态规划解法。下面将其优化到$O(NlogN)$。

首先将线段按照Y坐标排序。

引入扫描线。扫描线从下到上扫描。每一时刻的扫描线上储存的信息是从当前位置向下看，各个X坐标看到的第一个线段。

我们把这个信息用一个set_{Range}保存。其中Range是一个三元组，元素分别是左界，右界，原始线段ID。

那么，在set中，每刻只有连续的一段可能成为转移。我们发现，除了这一段的头尾，中间的各段在扫描到下一个高度的时候就会从set_{Range}中删去。因此这种优化下，进行的转移次数是线性的。又由于使用了set，带有一个log。总的是$O(NlogN)$。

注

下面是一种具体的实现：定义操作$split(x)$。这个操作首先二分x所在的区间。如果x在一个Range内部，把那个Range换成两个更小的Range。每处理一条线段，先进行两个split，然后erase整段区间，加入新区间。

这种实现下可以分析出，进入set的Range个数不会超过$3N + 1$。这也是转移次数的上界。

2.28 264D. Colorful Stones

题目描述

有两个序列，每个位置可能是RGB三种字母之一。你可以发布一个指令，指令是一个只含有RGB的字符串。最初，两个指针放在各自序列的头。每个指针按顺序检查指令的每一个字符，如果指针所指的字母等于指令的这个字符，指针后移一位。一个合法的指令应当满足：指针不会移出序列尾部。

称这两个指针所指位置的二元组是一个局面。求所有合法的指令具有多少种不同的局面。

数据规模

序列长度$\leq 10^6$。

解法

把一个局面用另一个方法来描述：四个串A, A_p, B, B_p。其中A是第一个序列中，指针前的串，也就是已经识别的部分。A_p是A加上指针所指的字符。B, B_p 同理。

一个合法指令如果导致这个局面，应当满足：存在A, B作为其子序列，而A_p, B_p 不是其子序列。

得到一个显然的性质：A_p不是B的子序列，B_p不是A的子序列。

这个性质是否充分？

用S^1表示S从下标一开始的部分，也即去掉第一个字符以后的部分。

局面A, A_p, B, B_p是可行的，以下三个至少要有一个成立：

1. A和B具有相同的首字母并且局面A^1, A_p^1, B^1, B_p^1是成立的。

2. A和B具有不同的首字母并且局面A^1, A_p^1, B, B_p是成立的。为了满足前述条件须有：A_p^1不是B的子序列。

3. A和B具有不同的首字母并且局面A, A_p, B^1, B_p^1是成立的。为了满足前述条件须有：B_p^1不是A的子序列。

如果A和B具有不同的首字母，A_p^1是B的子序列，B_p^1是A的子序列，就会导致满足了条件而不能到达的局面。这意味着：$|A_p^1| = |A_p| - 1 = |A| \leq |B|$且$|B_p^1| = |B_p| - 1 = |B| \leq |A|$。也即$|A| = |B|$。这说明$A[i] = B[i+1]$ 且$B[i] = A[i+1]$，而$A[i] \neq B[i]$，则A和B应当满足以下形式：$A = xyx...yxyx, B = yxy...xyxy$。

上述特例的最小实例是x, xy, y, yx。因此加入另一个条件就可以规避这种情况：A_p的最后两个字符若不相同，则不等于B_p的最后两个字符的倒排。可以用另一种方法说明这一条件的正确性。比如A_p 的最后两个字符是RG，B_p的最后两个字符是GR。那么，我们考虑指令中最后一个非B

的字符。如果是R：既然不是G，在扫描到这个字符之前，二指针应当指向GR中的R——这将导致矛盾。最后一个字符是G 同理。

现在问题转变成，有多少个非空前缀A_p和B_p满足上面的两个条件。预处理出每个前缀包含的另一个串的最长的作为其子序列的前缀，以及每个前缀，在另一个串中至少需要多长的前缀来作为一个子序列地包含它。枚举一个指针的位置，就变成询问另一个序列的前缀的一段区间中，最后两个字符不是某两个字符的有多少。这可以用前缀和解决。

2.29 280D. k-Maximum Subsequence Sum

题目描述

给出一个N长度的序列，在其中选择至多K段不相交的子段。最大化权值和。

M个询问，每个询问有参数l, r, k。表明对序列的$[l, r]$这段以$K = k$解上述问题。

数据规模

$N, M \leq 10^5, k \leq 20$。

解法

有一个显然的费用流解法，当然，费用流是很慢的。

考虑费用流的过程中究竟发生了什么：重复K步，每步找到最大子段，计入答案，反转这个子段。

我们用线段树优化这个过程。需要支持：区间乘-1，找区间最大子段。在线段树的每个节点记：和，反转标记，最大最小子段和，包含左边界的最大最小子段和，包含右边界的最大最小子段和即可。

2.30 243E. Matrix

题目描述

有一个01矩阵。重排列各列，使得在每一行，1都是连续的一段。

数据规模

$N \leq 500$。

解法

这是一个经典问题。用PQTREE就可以解决。

实现细节

PQTREE是一棵树，它被用来表示满足一族形式为"S_i中的所有元素在排列中是连续一段"的要求的所有排列。

PQTREE中有三类节点：乱序节点（P节点），有序节点（Q节点），叶子节点。

叶子节点共有N个。一个叶节点描述的序列只有这个叶节点一种。

P节点有至少两个孩子（如果只有一个孩子，把这个孩子拿来代替它就可以了）。一个有s个后代S_i的P节点描述的序列应当满足：

这个序列可以被分成s段，第i段被S_{A_i}描述。其中A_i是一个1到s的排列。

Q节点有至少三个孩子（如果只有一个孩子，把这个孩子拿来代替它；如果有两个孩子，把这个Q节点换成P节点即可）。一个有s个后代S_i的Q节点描述的序列应当满足：

1. 这个序列可以被分成s段，第i段被S_i描述；或，

2. 这个序列可以被分成s段，第i段被S_{s+1-i}描述。

PQTREE描述的序列就是根节点描述的序列。

要求是一条一条被加入PQTREE的。最初的PQTREE的根是一个乱序节点，它有N个孩子，都是叶子节点。这时的PQTREE描述了所有的N 阶排列。

现在，我们要在一棵PQTREE中加入一个要求了。对这个要求的集合中的所有数对应的叶子节点求LCA，对以这个LCA为根的子树做DFS。处理的顺序是先孩子再父亲。这意味着处理某个节点的时候已经处理了它所有的孩子。

根据当前处理的节点的类别讨论：

1. 叶子节点。如果这个节点的编号在这个要求的集合中，标记其为满节点（F节点），否则标记其为空节点（E节点）。

2. 乱序节点（P节点）。检查它的所有后代：

2.1. 全是E节点：标记这个P节点为E节点。

2.2. 全是F节点: 标记这个P节点为F节点。

2.3. 只有E节点和F节点的后代；当前P节点是这个要求的LCA：将那些被标记为F节点的后代从后代列表中拿出来，如果其数目至少为2，新建一个P节点，后代是这些F节点，把这个新节点加入到当前P节点的后代表。如果其数目为1，把这个F后代放回去。结束这一轮处理。

（为了方便起见，下面定义函数Group：它接受一个节点列表，如果节点数目为1，返回这个节点，否则新建一个P节点，后代是这些节点，返回这个新节点。现在我们可以称2.3.的过程是：把那些F后代拿出来，把它们的Group放回去）

2.4. 只有E节点和F节点的后代；当前P节点不是这个要求的LCA：将那些被标记为E节点的后代的Group和被标记为F节点的后代的Group作为这个P节点的新后代（也就是说现在这个节点只有两个后代，相当于

在中间加了一层）。标记这个节点为S节点（单端节点），并将其类别变成Q（虽然这里出现了一个两个后代的Q节点，但是为了方便起见，不要把这个节点变成P节点，并且我们以下约定：S节点全部是Q 节点；S节点的节点列表中，下标小的是E节点，下标大的是F节点，称S节点的头尾为E端和F端）。

2.5. 有一个后代是S节点；当前P节点是这个要求的LCA；如果有至少一个F节点后代，将这些后代从后代表中拿出来，把它们的Group添加到S节点后代的后代列表的尾部。这意味着，这些F节点可以任意排列，但是必须紧贴着S节点的F端。结束这一轮处理。

2.6. 有一个后代是S节点；当前P节点不是这个要求的LCA：如果有至少一个E节点后代，将所有的E节点后代拿出来，把其Group添加到S 节点的E端；如果有至少一个F节点后代，将所有的F节点后代拿出来，把其Group添加到S节点的F端。用这个S节点代替当前P节点。（这就是说，把当前节点变成Q节点，其后代列表就是刚才的过程所产生的后代列表）

2.7. 有两个后代是S节点：首先如果当前P节点不是这个要求的LCA，宣称无解。新建一个Q节点，后代列表是：第一个S后代的后代列表的倒序+所有F后代的Group（如果有F后代的话）+第二个S后代的后代列表。用这个Q节点代替后代列表中的F节点和S节点。

2.8. 有至少三个后代是S节点：宣称无解。

3. 有序节点（Q节点）。检查它的所有后代：

3.1. 全是E节点：标记这个Q节点为E节点。

3.2. 全是F节点: 标记这个Q节点为F节点。

如果不满足3.1.或3.2.，在后代列表中找到第一个非E节点和最后一个非E节点。如果这两个节点之间的那个子序列中有非F节点，宣称无解。

3.3. 只有一个非E节点；这个Q节点是这个要求的LCA：什么都不做，结束这一轮处理。

3.4. 只有一个非E节点（不处于列表的两端）；这个Q节点不是这个要求的LCA：宣称无解。

3.5. 只有一个非E节点（处于列表的两端）；这个Q节点不是这个要求的LCA：如果这个节点是F节点，什么都不做，结束这一轮处理。否则这个节点是S节点。在这种情况下，将Q节点的后代列表中的这个S节点换成S节点的后代列表，其中S节点的E端贴着Q节点的后代列表中其他的部分（这就是说，如果非E节点在列表头部的话，要先把整个列表倒序一遍）。

在考虑了以上情况（它们是下面所述情况的特例）以后，如果还没有匹配，再考虑以下情况：

3.6. 列表的一端是F节点，最后一个非E节点是F节点：如果这一端是头，倒序整个列表。标记这个节点为S节点，并且现在列表的尾部是F端。

3.7. 列表的一端是F节点，最后一个非E节点是S节点：如果这一端是头，倒序整个列表。将这个节点后代列表中的那个S节点维持原顺序

地换成S节点的后代列表。标记这个节点为S节点，并且现在列表的胃不是F端。

考虑到这里以后，如果这个Q节点不是这个要求的LCA，就可以宣称无解了。

3.8. 所有的非E节点都在列表的内部；这个Q节点是这个要求的LCA：如果第一个非E节点是S节点，将其维持原顺序地换成这个S节点的后代列表。如果最后一个非E节点是S节点，将其倒序地换成这个S节点的后代列表。结束这一轮处理。

以上就是所有的讨论。这样就可以实现在一棵PQTREE中插入一个新的要求了。

把矩阵的每一行当成一个要求，插入到这个PQTREE中。如果整个过程中没有宣称无解那么就是有解，其中的一个解就是DFS序下的访问各个叶子的顺序。

参考文献

S. B. Kellogg & George S. Lueker, "Testing for the Consecutive Ones Property, Interval Graphs, and Graph Planarity Using PQ-Tree Algorithms" (1975)

2.31 329D. The Evil Temple and the Moving Rocks

题目描述

有一个$N \times N$的网格（$N = 2K$）。你要在里面放上一些石块，每个石块有一个方向，当这个石块被激活的时候，就会朝着这个方向移动，直到前面是边界或另一个石块。如果这个石块在被激活以后至少移动了一格，它会发出一次响声。如果使它停止的是另一个石块，另一个石块会代替这个停下的石块而被激活。你的目标是：通过合理地摆放石块，激活恰当的石块，让石块发出的响声次数至少为X。构造一个方案并说明第一步激活哪个石块。

数据规模

$N \leq 300$，也即，$K \leq 150$；$X \leq K^3 - K^2$。

解法

将以下的结构称为一个电池：先是A个向前的石块，然后B组，每组先是一个空格，再是一个向前的石块（最后一个组的石块可以不向前）。当这个电池的最后部石块被激活以后，它将发出B次响声，并激活最前端

的石块。这个结构可以用A次，共产生AB次响声。当$A \to 2B$时这个AB最大。

在这个网格的第一列摆放一个向下的电池。在右边的$N-1$列摆放N个横向的电池，奇数行向左，偶数行向右，这样就可以达到要求的次数。

2.32 293D. Ksusha and Square

题目描述

给出一个凸包。在凸包中等概率随机地选择两个互异整点，以这两个点的连线为一条对角线构造一个正方形。求这个正方形面积的期望值。精度要求10^{-6}。

数据规模

坐标范围10^6。

解法

面积等于二分之一对角线长度的平方。而对角线长度平方等于X坐标差平方加Y坐标差平方。所以答案等于X坐标差平方的期望值加Y坐标差平方的期望值再除二。

以计算X坐标差平方的期望值为例。

首先计算出各个X坐标的点数A_i。预处理A_i的前缀和S_i^0，iA_i的前缀和S_i^1，i^2A_i的前缀和S_i^2。

答案即为：$\frac{2\sum_i S_{i-1}^0 i^2 - 2S_{i-1}^1 i + S_{i-1}^2}{(S_{+\inf}^0)^2 - S_{+\inf}^0}$。

2.33 306C. White, Black and White Again

题目描述

有N天，W个好事，B个坏事。要求将这些事件分布到各天，使得：最开始的某些天全部是好事，最末尾的某些天全部是好事，中间剩下的天都是坏事，每天至少有一件事。两个方案不同，当且仅当某一天的事件的有序列表不同。也即，同类事件是不一样的，在某一天内重排同类事件也会得到不同的方案。

数据规模

$N, W, B \leq 4000$。

解法

用D_m^n代表将m个无差别的东西分配到n个有差别的篮子里，每个篮子至少有1个东西的方案数。

$$D_m^n = \begin{cases} 0 & m < n \\ C_{m-1}^{n-1} & m \geq n \end{cases}$$

则答案为$\sum_{i=1}^{N-2}(N-i-1)W!B!D_W^{N-i}D_B^i$，其中$i$枚举的是中间一段的天数。

2.34 317C. Balance

题目描述

有一个无向图。每个点都有一个储量。每一时刻的储量都不能超过V且至少为0。每个点有一个初始储量和一个目标储量。每次可以选择一条边，将任意多的储量沿着这条边的任意一条边推过去。构造一个至多$2N^2$步的方案达到目标或宣称无解。

数据规模

$N \leq 300, M \leq 50000$。

解法

首先如果有一个连通块的初始总储量不等于目标总储量，宣称无解。

做一个生成森林，下面将证明，只使用生成森林上的边总是能产生解。

只要还有一个连通块剩下至少两个点，就在生成森林中找一个叶子。这个叶子连向所在树的其他部分的边上所需流过的流量是确定的。以从叶子流出为例，流入叶子的情形可以以此类推。如果这条边的另一端有足够的空余量（V减储量），那就可以直接推过去。否则，要先把另一端留出足够的空余量，这就要先从另一端向这棵树的其它子树推送流量。

定义操作$Push(x,y,v)$为从x向y推送v流量的操作。

$Push(x,y,v)$:

$while(V - A[y] < v) \quad Push(y,z,min(s_z, A[y]+v-V))$

$A[x] -= v$

$A[y] += v$

其中z是y的除了x以外的任意一个其子树有非零总空余量的邻居，s_z为这棵子树的空余量。

类似的可以得到流入叶子的递归解法。

每一步最多推整个连通块的边。因此步数是N^2的。

2.35 297E. Mystic Carvings

题目描述

在一个圆周上有$2N$个点。有N条边。求有多少个边的三元组使得六个点的分布形如"123123"或"112233"。

数据规模

$N \leq 10^5$。

解法

边的三元组有以下形式：

"112233"囧，"123123"星，"112332"三，"123132"艹，"112323"XI。

只需求出"三"，"艹", "XI"三种情况取补集即可。

"三"这种情况只需要枚举中间边即可。

"艹"和"XI"这两种情况中，都存在两条边，在剩下的两条边中恰与其中一条相交。因此只要将所有边的相交边数乘不相交边数，这个和的一半就是"艹"和"XI"的情况数。

2.36 294D. Shaass and Painter Robot

题目描述

有一个$N \times M$的网格。机器人从边界上的某个格子出发，按照某与边界成45度角的方向移动。遇角反向，遇边反射。

机器人会把每个到达的格子染黑。目标是把网格变得黑白相间。

求：是否可能达到目标？若能，机器人需要走几步。

数据规模

$2 \leq N, M \leq 10^5$。

解法

结论：边界黑白相间等价于整个网格黑白相间。

易见前者是后者的必要条件。

我们把每个边界点与45度方向遇到的边界点之间连边。则每个点的度数最多为2。

如果有多个连通块，那么永远无法达到边界黑白相间，同时也无法得到网格黑白相间。

现在考虑唯一连通块的类型：

1. 链。既然链的两端都已经被访问过，那么在访问这两个点之间，机器人已经经过了这条链所有的边。

2. 圈。机器人经过了几乎所有的边，除了出发点的某一条边。

除了边界上的格子，中间的每个需要被访问的格子都属于至少两条边。所以在机器人少经过至多一条边的情况下，可以说明中间的每个需要被访问的格子都已经被访问过了。

那么只要一段一段地模拟直至每个边界上需要访问的格子都已经被访问过为止即可。

如何判断无解？在处理圈的时候，每个点恰好被访问一次；链的话会有一段点被访问两次，一段点被访问一次。所以只要某个格子被访问了第三次，就可以宣称无解。

2.37 321D. Ciel and Flipboard

题目描述

有一个$N \times N$（其中$N = 2X - 1$）的矩阵$A_{i,j}$。每个操作可以将一个$X \times X$的子矩阵中的所有数变为其相反数。

目标是最大化矩阵元素和。

数据规模

$N \leq 33$。

解法

新建矩阵$B_{i,j}$，如果$A_{i,j}$被取了奇数次相反数，则$B_{i,j}$为-1，否则$B_{i,j}$为1。

约定数组标号为1到$2X - 1$。

有以下结论：

1. $B_{i,j}B_{i,X}B_{i,j+X} = 1$。要么一个操作完全没有涉及第$i$行，要么恰涉及式中的两个位置。

2. $B_{i,j}B_{X,j}B_{i+X,j} = 1$。要么一个操作完全没有涉及第$j$列，要么恰涉及式中的两个位置。

$X \leq 17$，所以可以枚举中间列的前X个数。这一步是2^X的。这样就决定了整个中间列。接下来可以对$X - 1$对列分别考虑其最大值。枚举中间行的这两个位置，分别计算此时的最大值。总的来说是$O(2^X X^2)$的。

2.38 274C. The Last Hole!

题目描述

平面上有N个点。以每个点为圆心，以t半径画一个圆。t等于当前时

间。求最小的那个时刻，使得之后圆的并都是没有洞的。

数据规模

$N \leq 100$。

解法

当三个点构成的三角形是锐角三角形时，这个三角形的外心可能成为洞。

当四个点构成的四边形是矩形时，这个矩形的中心可能成为洞。

对所有这样的点计算最早被覆盖时间。取最大值即可。

复杂度是$O(N^4)$。

2.39 319D. Have You Ever Heard About The Word?

题目描述

有一个字符串S。每次找到一个长度最短的（同样长度取最左边的）平方字符串（两个相同的字符串的连接），去掉后面一半。输出最后的字符串。

数据规模

$|S| \leq 50000$。

解法

首先由于$|S|$不大，$|S|^2$的暴力可以通过。（由于单位操作都比较简单）

下面给出一个$O(N^{1.5} + Nlog^2N)$的算法。首先，对于某一个特定长度l，可以在$O(N)$复杂度中处理所有长度为$2l$的子串。我们又观察到：需要处理的长度的数量在$O(\sqrt{N})$（严格的上界是$\sqrt{2N}$）。

下面的问题就是如何找出这些需要处理的长度。我们做Hash。每次处理一个长度后重算Hash。每次计算$O(N)$，总共计算$O(\sqrt{N})$ 次。利用Hash就可以快速判断某两段子串是否相等。将串S分成多段，每段的长度为l，只需二分地计算每相邻两段的LCP和LCS即可。复杂度是$O(\frac{N}{l}logl)$。对于所有的l，其和在$O(Nlog^2N)$。所以总复杂度为$O(N^{1.5} + Nlog^2N)$。

2.40 314E. Sereja and Squares

题目描述

有N长度的序列。已知它是一个括号序列。但只知道其中的几个左括号。求原括号序列的种数，对2^{32}取模。

数据规模

$N \leq 10^5$。
由于这道题考察算法的时间常数，因此应当指出时间限制为4秒。

解法

从左向右扫描各个符号。如果是一个左括号，则向左括号栈中压入这个左括号。如果是一个未知符号，它可能是25种可选的左括号之一，或是相对应于当前左括号栈顶的元素的右括号。

用$dp_{i,d}$表示已经扫描了i个符号，左括号栈深度为d的方案数。则：
边界条件：$dp_{0,i} = [i == 0]$；
答案：$dp_{N,0}$；
转移：

$$dp_{i,j} = \begin{cases} dp_{i-1,j-1} & LEFTBRACKET \\ 25dp_{i-1,j-1} + dp_{i-1,j+1} & UNKNOWN \end{cases}$$

复杂度是$O(N^2)$，在加上足够多的常数优化以后可以通过此题。

常数优化

1. 对2^{32}取模，就是unsigned int自然溢出的意思。

2. 当前符号为左括号的时候，dp_i就是dp_{i-1}移位一次的结果。当前符号为未知符号的时候，就是移位一次再将每个位置加上两格前的位置的25倍即可。用一个数组保存各个dp_i，善用指针。

3. j与i同奇偶，并且$j \leq min(i, N-i)$时状态$dp_{i,j}$才是有意义的。

2.41 249D. Donkey and Stars

题目描述

平面上有N个点。每个点可以看到它右上方，与其连线斜率在某个开区间内的点。

找到一个最长的序列，使得第一个元素被原点看到，每个元素可以看到下一个元素。

数据规模

$N \leq 10^5$。

解法

题中的描述相当于构造了一个平面非直角坐标系。先将它扭成直角坐标系。

剩下的问题是数点类问题的经典变体。使用CDQ分治或BIT解决。

3 第三部分(USACO, GCJ)

3.1 DEC06: Cow Patterns

题目描述

有两个序列A_i和B_i。对A的每个长$|B|$的子段，判断，这个子段和B是否在离散化后相同（下称，同构）。

数据规模

$|A| \leq 10^5, |B| \leq 2.5 \times 10^4$。所有编号小于等于25。

解法

类似于KMP，每次将子段的左界向右移动一格，然后将右界移动到next位置，并尽量多的右移。

同样类似于KMP，构造next函数的方法和匹配的方法类似，因此下面只说明匹配的方法，也即“已知现在的子段与B的相同长度的前缀同构，如何判断将子段右界右移以后是否仍然保持这一性质”。

答：如果在两个序列中，新加入的数的大小排名相等，那么依然保持这一性质。判断排名的方法就是：计算在前面比这个数大的有几个，小的有几个。由于编号只有25种，只需要预处理二维前缀和即可。

3.2 OPEN07. Connect

题目描述

一个$2 \times N$的网格，操作有将一条边设为可用，将一条边设为不可用，查询两个点是否能在只通过这两个点所属的两列之间的列的点的条件下连通。强制在线。

数据规模

$N \leq 15000, Q \leq 50000$。

解法

这道题是SHOI2008: traffic的一个弱化版本。和traffic一样，使用线段树，在线段树的每一个节点保存这一段的四个边界点的连通状况即可。

3.3 DEC07. Best Cow Line, Gold

题目描述

有一个字符串S，现在要用它构造一个新字符串。每步可以把S的头或尾拿出来压到新串的末尾。目标是最小化新串的字典序。

数据规模

$|S| \leq 30000$。

解法

显然，每次比较现有S和S的反串的字典序大小，哪边小就拿哪边就可以了。这给出了一个$O(|S|^2)$的暴力，可以用来解决这个题目非Gold的版本。对于Gold的版本，只要把每次的暴力比较换成后缀数组即可。

3.4 MAR08. Land Acquisition

题目描述

第一象限有N个点。每个点的价格是其两坐标之积。

要求以尽量少的总价格选取这样的一个点集（不一定在N个点中选择），使得N个点中的每个点都至少在点集中一个点的非严格左下方。

数据规模

$N \leq 50000$。

解法

显然，只有最外侧反链的情况和原情况是等价的。

从而，问题转变为，把一个序列$\{(A_i, B_i)\}$（$A_i < A_{i+1}, B_i > B_{i+1}$）划分成多段，每段$u..v$的费用是$A_v B_u$。从而有动态规划转移方程$dp_i = min(dp_j + A_i B_{j+1})$。

我们发现B_{j+1}是单调的，从而可以使用斜率优化优化到线性。但由于构造反链的过程是$NlogN$的，总的复杂度还是带log的。

3.5 OPEN08. Cow Neighborhoods

题目描述

平面上有N个点，曼哈顿距离在C以内的一对点将被一条无向边相连。

求：有多少个连通块；最大连通块的大小。

数据规模

$N \leq 10^5$。

解法

由最小生成树的环切性质，只考虑这个点集的曼哈顿距离最小生成树上的长在C以内的边，得到的图在连通性上与原图一致。

曼哈顿距离最小生成树是一个经典问题。使用线段树或线段树类似物就可以解决。

关于实现的细节，使用任意主流搜索引擎搜索“曼哈顿距离最小生成树”都可以方便地得到相关资料。

3.6 JAN09. Safe Travel

题目描述

给出一个有向图，保证1到每个点的最短路唯一。

对除了1以外的其它点计算：不经过1到这个点的最短路上的最后一条边的最短路径。

数据规模

$N \leq 10^5, M \leq 2 \times 10^5$。

解法

做出最短路树。

考虑答案路径的最后一条边，有两种情况：

1. 来自子树外。可以直接枚举入边转移。这部分均摊是$O(M)$的。

2. 来自子树内。这时，倒数第二个点的最短路要用到这个点的最短路，自然也包括最后一条边。我们考虑在哪条边，路径从子树外进入子树内（随后沿着最短路树的一段路径逆序地向上）。我们发现，对于每个节点，这样的边就是它的所有儿子的1. 2.两部分用到的边，除去起点在子树内的那些。

得到算法：对于每个点，把1.类边作为初始左偏树保存(需要保存路径长和进入子树的那条边的起点)。DFS。对每个点，处理完所有孩子以后，将孩子的左偏树中的所有元素加上父子边权值以后合并到父亲的左偏树中，删去那些起点在这个点的子树内的。取最小元素即可。

需要实现一个支持全集合加一个值和删除的左偏树。全集合加值用打标记即可，删除只要在取最小元素时判断该点是否应被删除（懒删除）即可。

3.7 MAR09. Cleaning Up

题目描述

有一个序列。现要将其划分成数段，每段的费用是该段中不同元素的种数的平方。最小化费用。

数据规模

$N \leq 40000$。

解法

$O(N^2)$的动态规划解法是容易想到的。

注意到至少存在一个解答案为N，因此超过$\sqrt{N}$种元素的一段是不可能出现在最优解中的。因此对每个i，有用的决策最多有$\sqrt{i}$种。从而优化到$N^{1.5}$。

3.8 OPEN09. Tower of Hay

题目描述

有一个序列。现要将其划分成数段，要求每一段的总和不小于其后面一段的总和。求最大段数。

数据规模

$N \leq 10^5$。

解法

有以下引理：至少存在一个最优解，第一段取到了所有解（不一定是最优解）中的最小值。

由这个引理得到以下动态规划：由后到前处理，DP_i表示i开始的序列后缀中，第一段的总和可以取到多小。最后走一遍就可以得到最大层数。

下面对其进行优化。记S_i为序列的前缀和。DP_i可以使用决策j（形式化地，令i开始的序列后缀的第一段取$i..j$）的条件是$S_j - S_{i-1} \geq DP_{j+1}$。或$S_{i-1} \leq S_j - DP_{j+1}$。称$S_j - DP_{j+1}$为决策$j$的生效时间。每次选择编号最小的那个生效时间大于等于当前S_{i-1}的决策。

问题转化成如下的形式——处理以下操作：

1. 插入一个未生效的数；

2. 将一个未生效的数设置为生效；

3. 求最近插入的并且已经设置为生效的数的编号。

我们使用单调队列解决，队列里满足编号单调和生效时间单调。这样就能做到线性。

证明引理

假设最优解S的第一段和比最小值大，这就说，第一段和第二段的分界线R_1比某个第一段达到最小长度的非最优解S'中的相应分界线R'_1大。$R_1 > R'_1$。

同时，由于S是最优解而S'不是，有$|S| > |S'|$。

$R_{|S'|} < R_{|S|}$，所以$R_{|S'|} < N = R'_{|S'|}$。结合$R_1 > R'_1$，至少存在一个位置i 满足$R_i \geq R'_i$且$R_{i+1} \leq R'_{i+1}$。

形象化地描述，就是说组成方案S'的第$i+1$段的元素是组成方案S的第$i+1$段的元素的子集。

构造新解S^n，其中对于$j > i$，$S^n_j = S_j$，否则，$S^n_j = S'_j$。这个方案的第一段是最小长度，并且段数和最优解相等。我们将说明这个方案是合法的。对于i及以前的各段，由于和方案S'一致，是合法的；对于i段和$i+1$段，前者和方案S'中的同一段相同，后者的和还要更小，因此也是符合要求的；对于$i+1$段和$i+2$段，后者和方案S中的同一段相同，前者的和还要更大，因此也是符合要求的；对于$i+2$及以上的各段，由于和方案S一致，是合法的。综上，任意相邻段间都满足前一段的和不小于后一段。

3.9 MAR10. StarCowCraft

题目描述

有正实数常量x, y, z，已知不存在一个常量高达另一个常量的100倍以上。

已知N组关系$A_1x + B_1y + C_1z \leq A_2x + B_2y + C_2z$，对$M$对数($S_1 = A_1x + B_1y + C_1z, S_2 = A_2x + B_2y + C_2z$)判断：是否一定有$S_1 < S_2$？是否一定有$S_1 > S_2$？

保证至少存在一组合法常量，满足全部N组关系。

数据规模

$N \leq 300, M \leq 2000$。

解法

记$X=\frac{x}{z},Y=\frac{y}{z}$。则最初的$(X,Y)$域为六边形（含内部）$(0.01,0.01)-(0.01,1)-(1,100)-(100,100)-(100,1)-(1,0.01)$（此处易错当成正方形$(0.01,0.01)-(100,100)$）。每一个条件相当于$AX+BY+C\leq 0$，是一个半平面。

使用半平面交即可处理出(X,Y)的取值范围。

对于每个询问，检查半平面交的每个顶点，是否所有的顶点上的解都满足相应要求即可。

3.10 OPEN10. Triangle Counting

题目描述

平面上有N个点。统计有多少个由此定义的三角形包含原点。

没有两个点定义了通过原点的直线。

数据规模

$N\leq 10^5$。

解法

这个问题等价于统计有多少不包含原点的三角形。如果一个三角形不包含原点，三个点极角的跨度就在π以内。由此可以定义出“顺时针数过来的第一个点”（使得其它点关于原点的方向都在这个点的方向向顺时针旋转π以内的范围内）。对每个点，统计出这个π的范围内有多少点即可。而这一步可以使用极角排序+two pointer解决。

3.11 DEC10. Threatening Letter

题目描述

有两个字符串S,T，将T切成尽量少的段，令每段都是S的子串。

数据规模

$|S|,|T|\leq 5\times 10^4$

解法

我们发现从左到右贪心，能取就取是正确的。可用调整法证明。

那么问题转变成，一个空串，每次加一个字母，询问是否是S的子串。

这个问题可以用后缀数组在$O(NlogN)$时间，后缀树或后缀自动机在$O(N)$时间内完成。

3.12 MAR12. Cows in a Skyscraper

题目描述

有一个可重集。将其分成尽量少的几个子集，使得每个子集的和小于等于给定的值。

数据规模

$N \leq 18$。

解法

考虑一个最优解。找到一个可重集的排列，使得最优解中分在同一子集的元素在排列中是连续一段。在这个排列中用贪心法就可以得到一个最优解（不一定是一开始的那一个）。

这启示我们：对所有的排列做贪心法。暴力是$N! \times N$的。可以使用动态规划优化到$2^N N$。其中dp_{mask}表示，长为$|mask|$ 的前缀包含了$mask$的元素，最少需要的子集数和最后那个子集的最大剩余空间。

3.13 OPEN13. Photo

题目描述

有一个未知的零一串。给出若干个区间，已知这几个区间中各恰有一个1。求整个串中至多有几个1或宣称无解。

数据规模

$N \leq 2 \times 10^5, M \leq 10^5$。

解法

从后到前DP。DP_i表示只考虑满足那些右端点大于等于i的区间，在i位置有1的条件下最多能在从i开始的后缀中放几个1。那么转移就是寻找哪些点可以作为下一个1的位置。

如果存在某个跨越了i的区间，那么这个区间的右端点及之前都不能成为一个合法决策——那会导致某个区间中有两个1。如果存在某个完整地处

于i以后的区间，那么这个区间的右端点以后都不能成为一个合法决策——那会导致某个区间中没有1。

从而，每个点可用的决策是一个区间去掉若干个前缀和若干个后缀，也就是一个区间。用线段树维护区间最大值即可。

关于每个点对应的可行区间：假设我们建立了一个set数组，如果存在区间(a,b)就在数组的第a位的set中插入b。那么寻找区间的过程就变成一个后缀最小值和一个前缀最大值询问：预处理即可。

3.14 DEC12. First!

题目描述

有N个字符串。求：有哪些字符串可以在某种“字符大小排列”下成为所有串中字典序最小的。

数据规模

$N \leq 30000, \Sigma|S| \leq 300000$。

解法

做trie。对每个trie上的节点可以处理出：使得字典序最小的串可能出自以该节点为根的子树的条件，具体的而言是一个小于关系的传递闭包。使用位运算可以优化到每加一组条件复杂度$O(|\Sigma|)$。当且仅当一个字符串的传递闭包是偏序的并且祖先中没有其它的字符串时，这个字符串可能成为字典序最小。

3.15 DEC12. Gangs of Instanbull/Cowstantinopl

题目描述

有N个数和一个栈。当一个数来到栈，它检查栈顶：若无元素或栈顶元素等于它，它压入栈；否则将栈顶元素弹出。

构造一个这N个数来到栈的顺序，使得最后栈中的1数量最多。相同1数量的方案中输出那个字典序最小的。所有的顺序下1的个数都是0则宣称无解。

数据规模

$N \leq 10^6$。

解法

第一步是判断是否有解。（以下以C_i代表i的个数。）

如果$N - C_1$是奇数，我们至少需要损失一个1；

如果$2Max\{C_i\}(i \neq 1) > N - C_i$，我们至少需要损失$2Max\{C_i\}(i \neq 1) - (N - C_i)$个1。

从而当且仅当$max((N - C_i)\%2, 2Max\{C_i\}(i \neq 1) - (N - C_i)) \geq C_1$，无解。

第二步在有解时输出字典序最小的最优序列。

当只有两种数时，先来C_1个1再来C_2个2即可。

否则，先把要放弃的$Loss = max((N - C_i)\%2, 2Max\{C_i\}(i \neq 1) - (N - C_i))$个1压栈，最后用$C_1 - Loss$个1“打扫战场”。中间部分是这样处理的：按字典序顺序加入各个数，直到加入新的数以后会导致无法在没有更多1干涉下使栈变空（下称，冲突）。冲突的具体表现是：除了1以外未损失个数最多的那种数的个数超过了“栈中1的个数”加“除了1以外其他数的未损失个数之和”的一半。发现冲突以后，把那种最多的数全部加在最后$C_1 - Loss$个1前面（对于“中间”而言，就是加在尾部），把剩下的数按字典序放在这些数以后。

现在问题来了：如何$O(1)$判断是否发生冲突？我们把所有的数还幸存几个记下来，然后按照幸存个数把它们分类（“桶”）。每一个桶挂一个双向链表（为了支持$O(1)$删除特定元素和$O(1)$插入），然后用一个指针指向幸存最多的数的个数。每当这个指针指向了一个空桶，将指针减一。这样就能实现总复杂度$O(N)$了。

3.16 MAR13. Hill Walk

题目描述

在平面上有若干条斜率为正的互不共点的线段。每条线段的右上角都不属于这条线段。现有一人从第一条线段的左下角出发，每时每刻向右上方走。当位置不属于任何一条线段时就会向下垂直掉落。问在落到无限低处前这个人在多少条线段上走过。

数据规模

$N \leq 10^5$。

解法

由于线段不共点，所以可以使用set维护扫描线。使用扫描线做出每条线段的后继即可。

3.17 OPEN13. Figure Eight

题目描述

有一个网格，每个格子是好的或破坏了的。找出两个厚度为1的“框”，最大化两个框的内部面积的乘积，同时需要满足：

1. 框的每一个格子都是好的。（框内部可能有破坏了的格子）
2. 框的内部面积不为0。
3. 某一个框的下边被另一个框的上边完全包含。

数据规模

$N \leq 300$。

解法

1. 对于每个点，计算最远的一个右侧位置，使得这个点到那个位置的所有格子是好的。这部分是二次方的。现在我们可以$O(1)$判断一个横杠是否是完好的。

2. 对于每个横坐标二元组，计算框的左右边若在这两列，最高的上边和最低的下边的位置。这部分是三次方的。

3. 现在我们可以得到，上部框的下边在某个特定段，下部框的上边在某个特定段时，这个框的最大面积。对前者做前缀和。（这里的前缀和的第一维是反的，使得新的$i..j$包含了所有原来包含于$i..j$的段的值的最大值）这部分是三次方的。

现在再花一个三次方的时间就可以得到答案。总复杂度是三次方的。

3.18 GCJ14C. Symmetric Trees

题目描述

有一棵树，每个点有个颜色。判断能否将这棵树重新在平面上排列（使得没有边相交），新的树关于某一条直线对称。

数据规模

$N \leq 10000$。

解法

首先用Hash法为每个子树和子树的补标号。

讨论这条对称轴和树的边的关系：

1. 树的某一条边被对称轴等分。这种情况表现为存在一个子树和它的补同构。

2. 其他情况。具体的说，若干条边落在对称轴上。这种情况表现为存在一个子树和它的补都是可对称的（也即，可以被重排为对称的形式，并且根落在对称轴上，根沿对称轴的两个方向中至少有一个方向空出来）。如何快速判断一个子树或子树补是否可对称？在预处理时就把所有的子树和补处理出来。如果将子树或子树补的所有子树中同构的两两配对后，没有剩下子树或剩下一个子树并且那个子树是可对称的，这个子树或补就是可对称的。

3.19 GCJ14D. Paradox Sort

题目描述

有一个N阶竞赛图。构造一个N-排列，使得在以下过程以后memory为给定的值：

最初memory是排列的第一位。从第二位开始检查排列的各位：如果编号为排列的这位的点到编号为memory的值的点有一条有向边，则将memory的值改为排列的这一位。

数据规模

$T, N \leq 100$。“请大胆使用渐进复杂度高的算法”。

解法

首先判断是否有解。

结论：如果从给定的节点可以到达所有点，则有解。否则无解。

构造：做一棵以给定的节点为根的DFS树形图。在树上做后根遍历。遍历序列就是一个可行解。

下面将对以上构造的正确性进行证明：

引理：每个点都到它前面的那个点有有向边。证明，如果这个点和前一个点是父子关系，由后根，这个点是前一个点的父亲，这个点到前一个点有有向边；对于其他情况，由DFS的性质，既然这个点不在前一个点所在的子树内，前一个点到这个点没有有向边，由竞赛图的性质，这个点就到前一个点有有向边。

证明：由于每个点都到它前面的那个点有有向边，所以每时每刻的memory就是最后一个检查过的序列元素。故扫描一遍以后的memory就是根节点，即给定节点。

然后给出方案。$O(N^2)$逐位确定。下面将给出一个方案，判断在当前memory和剩余元素给定时，判断是否有解。

以给定的节点为根DFS，只允许经过剩余元素和当前memory。当对当前memory中的元素进行DFS时，不接着DFS下去。最后检查没有被访问到的点，如果都可被当前memory的元素一步可达，即有解，否则无解。

构造：先用当前memory中元素把没访问到的元素干掉，然后依然是做后根遍历（去掉memory中的元素）。由于memory中的元素到它前面的所有点都有边（事实上，在竞赛图的后序DFS序列中，每个点到它前面的所有点——除了遍历树中的子孙，都是有边的，而访问到memory 时不接着DFS下去，这就使得memory在遍历树中没有子孙），所以在后根遍历到memory为止，memory的值没有变化。在memory以后，memory的值要么不变，要么就是最后一个检查的元素——memory将保持不变直到碰到一个到它有有向边的元素，这样的元素至少有它的父亲（当memory没有父亲的时候，也即memory就是给定元素时，经过专门的讨论可以得出该做法依然正确）。

总的来说是$O(N^4)$的——共做$O(N^2)$次DFS，每次$O(M) = O(N^2)$。由于“请大胆使用渐进复杂度高的算法”，实际上是可以通过的。

3.20 GCJ14E. Allergy Testing

题目描述

有N种食物，实验者对其中一种过敏。每次可以选择一个集合的食物进行试验。在A天以后可以得到结果。如果都不过敏就可以马上开始下一个实验，否则还需要等待$B - A$天才能开始下一个实验。求找出那一种的所需时间。

数据规模

$N \leq 10^{15}, A, B \leq 10^{12}$。

解法

对于每个时间我们可以计算出在这个时间中可以检验的最大N值$f(x)$。

$f(x)$是以下定义的：$f(x) = \begin{cases} 1 & x < A \\ f(x-A) + f(x-B) & Otherwise \end{cases}$

这个函数在$x < 0$也有定义只是为了方便。当$A <= x < B$时，尽管$x - B < 0$，但是知道结果的时间是A天，所以只需要A天就可以知道过敏原在$f(x - A)$种中还是$f(x - B)$种中——如果是在$f(x - B) = 1$种中，那么就直接确定了过敏原，自然也不用再等$B - A$天做新的实验了。

我们做一棵树，根节点在高度x，所有的叶子节点在高度A以下，每个高度为h的非叶子节点有两个孩子，分别在高度$h - A$和高度$h - B$。那么$f(x)$就是这棵树中的叶子个数。

我们注意到，每个高度在$A-B..-1$的叶子都对应了一个高度在$A..B-1$的非叶子。从而叶子的个数就等于高度在$0..B-1$的节点数目。问题转化为计算这棵树的高度在$0..B-1$的点的个数。注意到如果$x > B\log N$，则$f(x) > N$。所以只要枚举$Ax+By$中的y，使用组合数就可以快速算出$f(x)$了。

有了$f(x)$就可以用二分答案算出最小天数了。

3.21 GCJ13E. Let Me Tell You a Story

题目描述

有一个序列。重复以下过程：如果这个序列并非不升，随机删掉一个元素。求：删掉元素的顺序有多少种。

数据规模

$N \leq 2000$。

解法

在这个过程结束的时候，剩下的是原序列的一个不升子序列。枚举这个不升子序列。假设其长为x。则有$(N-x)!$种方式删到这个子序列。但是并非这$(N-x)!$种方式都是合法的。如果在最后一次删除之前已经是不升的，那么这种方案是不合法的。直接计算不合法的做法无法实现，我们可以把不合法而删掉的这部分贡献归于最后一个删除之前的不升序列，这就是说：

每个长为$x>1$的不升序列的贡献为$(N-x)!-x(N-x)!$。后一部分是这样来的：将这个序列删除一个元素可以得到x种长为$x-1$的不升序列。在这x种序列中，各有$(N-x)!$种操作序列由于最后一次删除前是这个序列而变得不合法。故认为这个序列有$(1-x)(N-x)!$的贡献。

计算出各个长度的不升子序列数目即可。

3.22 GCJ11A. Runs

题目描述

对于一个字符串，Run表示字符串中极大的连续段，使得整段都是同一种字符。

给出一个字符串，有多少种重排方式，使得新老字符串的Run数相等。两个重排方式不同，当且仅当新的字符串不同。

数据规模

$N \leq 450000, countRun \leq 100$。

解法

由于Run数很小，所以可以使用动态规划解决：$dp[s][i]$表示前s种字符，有i个Run的方案。转移时枚举这种字符有几个Run，有多少是插入原有的Run的空隙的，有多少是插入原有的Run的内部的。复杂度是$|\Sigma| countRun^3$。

3.23 GCJ11C. Program within a Program

题目描述

输出一个自动机。内态范围和纸带符号范围都可以看成无限（10^6）。目标是使停机位置在起始位置以右N格。

数据规模

$0 \leq N \leq 5000$。 允许使用的指令数≤ 30。

参考解

以$N = 19$为例，下面是一个参考解（初始内态是$W0$，初始纸带上全是nil）：

$statu$	sig	$nxtStatu$	$nxtSig$	$nxtDirection$	$isStop$
$W0$	nil	$W1$	1	$\leftarrow$	
$W1$	nil	$W2$	1	$\leftarrow$	
$W2$	nil	$W3$	0	$\leftarrow$	
$W3$	nil	$W4$	0	$\leftarrow$	
$W4$	nil	$W5$	1	$\leftarrow$	
$W5$	nil	I	nil	$\rightarrow$	
I	nil				YES
I	0	I	nil	$\rightarrow$	
I	1	$C1$	nil	$\rightarrow$	
$C0$	nil	$M1$	1	$\leftarrow$	
$C0$	0	$C0$	0	$\rightarrow$	
$C0$	1	$C1$	0	$\rightarrow$	
$C1$	nil	$M0$	0	$\leftarrow$	
$C1$	0	$C0$	1	$\rightarrow$	
$C1$	1	$C1$	1	$\rightarrow$	
$M0$	nil	I	nil	$\rightarrow$	
$M0$	0	$M0$	0	$\leftarrow$	
$M0$	1	$M0$	1	$\leftarrow$	
$M1$	0	$M1$	1	$\leftarrow$	
$M1$	1	$M0$	0	$\leftarrow$	

其中各个N下W阶段都是不同的，而W以外的部分是相同的。

工作流程

在W阶段，把N按二进制写在纸带上。进入循环：

1. M阶段。这个阶段用来将二进制数减一。将二进制数减一的方法是，从后到前寻找最低的1，将它以下的各位翻转。$M0$状态代表已经经过了最低的1，$M1$状态表示还没有经过。可以将Mx状态理解成，将到指针位置为止的这个前缀减掉x。

2. I阶段。这个阶段用来去掉二进制数头部的前缀零。在I阶段遇到nil则说明整个数已经减到零，可以结束了。

3. C阶段。这个阶段用来将二进制数整体右移一位。

整个过程是：每次将二进制数减一以后把二进制数的位置右移一位。二进制数表示了还要进行几轮这样的子过程。扣减到零时指针就已经处于以右这么多个格子的位置了。

3.24 GCJ11E. Google Royale

题目描述

在赌场中赌钱。目标是使资金a至少为V。失败条件是$a \leq 0$。

每一次可以下$1..min(a,M)$范围内任意整数的注。每一次下注有一半概率赢一半概率输。输了的话，如果下注量的两倍在M以内的话，可以追加下两倍的注，直至不想加注，两倍超过M，或者赢了这一次下注为止。每一轮下注结束以后结算全局的胜负。

求一个策略最大化成功的概率。

数据规模

$V \leq 10^{16}$。

解法

一个明显的事实：每一时刻的资金期望值都是初始值A。因此，失败时的资金数越小，相对应的成功率就越大。

另一个明显的事实：当前资金越多，成功率就越大，从而，下注超过$V-a$是无意义的。

下面推出一个性质：至少存在一个最优策略，使得每一步的操作只有以下两种：下1的注，不追加（下称，微调）；下所有的注（当$a \leq M$），每次失败都追加（下称，试图翻倍）。

证明使用的是调整法。假如某一步，操作是下s（$s < a$）的注，一直翻倍直到资金数达到$a-p$（也就是，前若干次下注和追加都是失败）或$a+s$（也就是，在前若干次下注内有一次成功）。

当$a-p>0$：用一系列微调来代替这个操作。微调直至资金数达到$a-p$或$a+s$。由资金期望值一定，这两个操作是等价的。

当$a-p \leq 0$：若资金达到$a-p$时还可以追加下注，那么当然是应当追加下注，因为这样可以减小失败时的资金数，从而提高成功率。因此以下认为$a-p$时已经不可以追加下注了。首先微调直至资金数达到s或$a+s$。若达到$a+s$，结束。否则，试图翻倍。如果失败了，资金是$s-p$，小于$a-p$。如果成功了，资金是$2s$，继续微调直至资金数达到s或$a+s$。在这个新方案中，失败的资金数更小，成功时资金数一样，所以成功率更大。

以下我们就只考虑这两种操作了。

用M_i来代表下注限制M右移i位的结果。M_i是最大的那个可以追加i次注的初始投注量。性质：如果a不是某个M_i，则至少有一个最优解下，a处的操作只有微调。

证明：选择一个最大的数a，使得这个数不是某个M_i，但在这个数的决策是试图翻倍。微调直至资金数是$[\frac{a+1}{2}]$或$2a$。如果是后者，相当于试图翻倍成功，如果是前者，试图翻倍。失败的话资金数小于原来的资金数，成功地话资金数目为a或$a+1$了。跳到微调的那一步。这样一来失败的资金数小于原来的资金数，所以成功率就大了。

现在，对于这log个可以试图翻倍的点，都可以计算出试图翻倍失败时的失败资金数。如果一个点的失败资金数不小于比它值小的那些点的失败

资金数，完全可以先微调至那个点或当前的两倍，然后如果微调到了某个失败资金数比它小的点，再试图翻倍。这样就可以得出所有决策为试图翻倍的点。

在每个点的决策都已经确定的情况下就可以得出每个点的成功率了。成功率大致是个分段函数，每段都是线性的，转折点是那些试图翻倍的点。

3.25 GCJ10A. Letter Stamper

题目描述

有一个有三种字符的字符串。现在有一个带栈的打字机，可以在一单位时间内执行以下操作：

1. 打印栈顶字符；
2. 弹出栈顶；
3. 压入任意字符。

求最小的步数，以输出这个字符串。在这个过程前后栈都应当是空的。

数据规模

$N \leq 2000$。

解法

至少有一个最优解满足以下性质：

性质一：任何时候，栈内没有两个相邻元素是一样的（无XX）。证明显然。

性质二：任何时候，栈内没有两个距离为2的元素是一样的（无XYX）。证明是调整：将压入第二个X变成弹出Y，将弹出第二个X变成压入Y。

从而栈内元素只可能是"XYZXYZXYZ"形式，状态数为$6N$。从而就可以$6N^2$地DP了。

3.26 GCJ09A. Year of More Code Jam

题目描述

有随机分布在$1..N$的变量A_i（$1 \leq i \leq T$）。每个A_i对$P[A_i + O_{ij} - 1]$（$1 \leq j \leq M[i]$）有一的贡献。求$\sum_{i=1}^{N} P[i]^2$的期望值。

数据规模

$N \leq 10^9, M, T \leq 50$。

解法

对于所有$sumM^2$个无序二元组$(A_i + O_{ij} - 1, A_{i'} + O_{i'j'})$，如果两个元素相等且在$1..N$，就对$\sum_{i=1}^{N} P[i]^2$有一的贡献。枚举这$sumM^2$个二元组，把对答案有贡献的概率相加即可。

3.27 GCJ09B. Min Perimeter

题目描述

平面上有N个点，求它们所决定的三角形（包括退化的）中最小的周长。

数据规模

$N \leq 10^5$。

解法

类似平面最近点对，使用分治法解决。现在问题转化为，在两个相邻的矩形中分别选择一个和两个点，构成周长尽量小的三角形。

设既有答案为p，则我们只考虑距分界线$\frac{p}{2}$以内的点。将所有点按照在分界线上的投影位置排序，从左到右枚举各个点，然后在右边投影距离$\frac{p}{2}$范围内枚举下面的两个点。这样就可以通过本题。

3.28 GCJ09D. Wi-fi Towers

题目描述

经典问题：最小权闭合子图问题。

数据规模

$N \leq 500$。

解法

使用最大流类算法解决。

3.29 GCJ08E. The Year of Code Jam

题目描述

有一张表格。有一些已经确定了颜色（W/B），有一些没有。要为每个未确定格子确定一个颜色。对于每个B格子，最初它的价值是4，它每和一个B格子相邻，价值减一。目标是最大化总价值。

数据规模

$T \leq 100, N, M \leq 50$。

解法

首先在表格外面加上一圈W格子。这样一来价值不变，但是我们可以把价值表示成：两侧颜色不同的边数。

然后把所有两坐标奇偶性相同的格子反色，这样一来，就可以把价值表示成，两侧颜色相同的边数。

换言之，就是所有内部边的数目减掉两侧颜色不同的边数。现在的目标就是最小化两侧颜色不同的边数。这样就可以使用最小割解决了。