IOI2016 集训队试题泛做总结

佛山石中李子豪

下面加☆的，为我认为比较好的题目。

非 Challenge 题目完成情况：100/102

10.5 时空复杂度

非 Challenge 题目

试题泛做(1)

1.1 Game of Numbers

1.1.1 题目大意

给定了两个长度为 N 的数组 A 与 B。

找出尽量多的二元组进行匹配，求最大匹配数。

对(i1,j1)与(i2,j2)，如果可以匹配，当且仅当 Bj1>Ai1,Ai2>Bj2,gcd(gcd(Ai1,Bj1),gcd(Ai2,Bj2))≠1.

1.1.2 数据范围

$N \le 400, T \le 10, 1 \le Ai, Bi \le 10^9$.

1.1.3 关键词

网络流、匹配、分解质因子

1.1.4 题解

对于此题，我们可以先考虑一个比较暴力的方法：对于任意的二元组对(i1,j1)与(i2,j2)如果可以匹配，那么连一条边，然后进行一次二分图匹配即可。

这样的话，点数为 O（N^2),边数达到 O（N^4).

那么，我们进一步考虑究竟怎样才需要连边，不互质。而不互质则意味着至少有一个公共质因子。因此，实际上我们只需要建立一些中间的节点表示含有这个质因子，然后各个点对分别连向它所含有的质因子，再进行网络流，这样优化即可通过 cc 的数据。

而如果要通过清澄上的数据的话，我们则需要再把 gcd(ai,bj)值相同的点对合并到一个点上，然后再进行网络流。

1.1.5 时空复杂度

空间复杂度 $O(N^2 logN)$, 时间复杂度 $O(maxflow(N^2, N^2 logN))$

1.2 Sereja and Equality

1.2.1 题目大意

定义对于两个排列 p1，p2，F(p1,p2)=满足 p1[l...r]同构于 p2[l..r]且 p1[l..r]的逆序对个数<=E 的数对(l,r)个数。

两个数组同构指的是数与数之间的相对大小关系一致。

求所有排列之间的 F 值的和。

1.2.2 数据范围

$\mathrm{T} \leq 10^4, N \leq 500, E \leq 10^6$

1.2.3 关键词

排列组合

1.2.4 题解

假设我们知道 G[N][i]表示 N 的排列中逆序对个数<=i 的数目，那么我们可以对于各个长度的子区间通过排列组合求出满足子区间外任意排列子区间内排列顺序固定的方案数，乘法原理即可。

而对于 G[N][i],我们可以通过考虑从 G{N-1]推出来，新加了一个数 N，那么根据 N 后面有多少个数就可以知道新增了多少逆序对，预处理出来暴力维护即可。

然后为了防止空间不够的问题，我们需要离线处理。

1.2.5 时空复杂度

空间复杂度 $\mathrm{O}(\mathrm{N}^2)$, 时间复杂度 $\mathrm{O}(\mathrm{N}^3)$.

1.3 Misinterpretation 2

1.3.1 题目大意

要找出长度在[L,R]范围内且仅有小写字母组成的满足约束条件的字符串的个数。

约束条件为:对于字符串 S，下标从 1 到 N 标号，把偶数标号位上的字母按顺序写在最前面，再把奇数位上的字母写在后面得到 S’，要求 S 与 S’相同。

1.3.2 数据范围

$\mathrm{T} \leq 5, \mathrm{R} \leq 10^{10}, R - L \leq 50000.$

1.3.3 关键词

欧拉函数、取模、数学分析

1.3.4 题解

首先，这题的答案显然就是 26^K,K 为各个要求必须相同的部分的数目。然后，对于 2N 与 2N+1，2N+1 必然比 2N 恰好大 1.而对于 2N+1，我们可以发现一点 f(i)=2i (mod 2N+1),f(i) 为后来第 I 位对应原本的第 f(i)位。

然后，我们就只需要看看对于每一个数要乘多少个 2 才能变回本身。那么，我们就可以得到一条式子$K = \mathrm{Sum}\left(\frac{\mathrm{eul[d]}}{\mathrm{ord2[d]}}\right)$ .d 为 2N+1 的因子，eul[d]为对应欧拉函数，ord2[d]为最小的 k 满足$2^{\mathrm{k}} = 1\ (\mathrm{mod}\ 2\mathrm{N}+1)$。这个也很好理解，就是对于与 2N+1 公因子恰好为（2N+1)/D 的数，总共有 eul[d]个，然后只要算出对应的 ord2[d],自然就得出不同的部分的数目了。

然后由于 R-L<=50000 且 R<=10^10，我们直接进行筛法就可以把所有的 N 进行因数分解了。然后就是算 ord2[d],这个我们可以通过一些转换去弄：

Ord2(p,q)=lcm(ord2(p),ord2(q)),对于 p，q 互质；

$\mathrm{Ord2}(\mathrm{p^i})|\mathrm{eul}(\mathrm{p^i})$,p 为质数。

因此，我们就求解 p^i 的 ord2 即可。而只需要对于 eul[p^i]进行质因数分解，然后尝试除去每一个质因子就能得到 ord2(p^i)。

综合上面的部分就能得到答案了。

1.3.5 时空复杂度

空间复杂度 O(50000log50000),

时间复杂度 $\mathrm{O}\left(50000\log 50000 + 50000\log^2 50000 + \frac{50000}{\ln(10^{10})} * sqrt(50000)\right)$.

1.4 Card Shuffle

1.4.1 题目大意

对于一个 1-N 的序列，进行 M 次操作，每次抽取中间一段反向插入到另外的位置。问 M 次操作后的序列。

1.4.2 数据范围

$\mathrm{N}, \mathrm{M} \le 10^5$.

1.4.3 关键词

平衡树

1.4.4 题解

使用一个带有翻转标记的 splay 进行维护即可。

1.4.5 时空复杂度

空间复杂度 $O(N)$，时间复杂度 $O(N\log N)$.

1.5 Sereja and Arcs

1.5.1 题目大意

有长度为 N 的数组 S，问有多少个四元组(A,B,C,D),满足 A<B,C<D,Sa=Sb,Sc=Sd,Sa!=Sc,区间[A,B]与区间[C,D]相交。

1.5.2 数据范围

$N \le 10^5, \mathrm{Si} \le 10^5.$

1.5.3 关键词

分块

1.5.4 题解

这一题我们首先可以求一下补集，首先总的很好算。

然后就是两个区间满足 a<b<c<d，这种的话维护一个前缀和和后缀和乘法原理即可。

然后就是 a<c<d<b 的情况。

接着可以对于某一对数字 X1，X2，根据出现次数 Num1 与 Num2 的大小进行分组处理。

如果 Num1，Num2<=T,直接统计左端点大于当前值且以某个为右端点的区间数，用树状数组维护即可。

如果 Num1 >T,直接扫描数组，每个数记录在它前面出现的 X1 的个数以及后面出现的 X1 的个数维护即可。

如果 Num2>T,Num1<=T,则对于所有不同的 X1，不断右移右端点，并维护右端点固定的答案数即可。

当 T 取 sqrt(N)时有最小。

1.5.5 时空复杂度

空间复杂度 $O(Nsqrt(N))$, 时间复杂度 $O(Nsqrt(N))$.

1.6 Two Companies

1.6.1 题目大意

给了一棵 N 个点的树，以及 Na 条带花费 A 路径与 Nb 条带花费 B 路径。要求找到一个 A 路径子集与 B 路径子集使得子集中 A 与 B 路径不相交，求最大花费。

1.6.2 数据范围

$N \le 10^5, Na, Nb \le 700.$

1.6.3 关键词

最小割

1.6.4 题解

这一题，比较容易看出的是典型的最小割模型，我们只需要先判断两路径相不相交来进行连边，之后再进行最小割即可。而判断是否相交的话，可以通过比较 lca 的深度关系得出，自己划一下就知道了。

1.6.5 时空复杂度

空间复杂度 $O(NlogN + Na * Nb)$,
时间复杂度 $O(Na * Nb * logN + maxflow(Na + Nb, Na * Nb))$。

1.7 PARADE 年度游行

1.7.1 题目大意

给定一个 N 个点 M 条边的无向图，每条边有花费。然后你需要选择 K 条长度大于 0 的路径（起点和终点可以相同），那么花费为总走边的花费（一条边被多次走就算多次）+没有经过的节点数*C+起末点不同的路径数*C。Q 次询问不同 C 时的最小花费。

1.7.2 数据范围

$N \le 250, M \le 30000, Q \le 10000.$

1.7.3 关键词

网络流、二分、floyd

1.7.4 题解

这道题，首先我们考虑一次询问。

我们可以尝试拆成入点和出点来构图，并且由于最终答案只和起末点有关，我们可以用 floyd 维护最短路径来优化速度。

然后，我们可以给所有入边一个流量 1 费用 0 的边表示以这个为起点，然后由于存在绕回自己的情况，我们可以选择在入点间进行连边，流量 oo，边权为最短路径（实际上也可以去维护每个点绕回自己的最短路径值从而直接连到出点），然后还有流到出点的边，边权同样是最短路径值。然后顺便再将 S 连向所有出点边权为 C 的边表示不走这个点或者意义为走回去，做费用流即可，可以证明这个是正确的。

然后，我们考虑一下一次增广，只增广一个路径，并且增广路径的费用是递增的，因此实际上我们通过记录每次增广的花费，最后与 C 来比较，就可以得到最优的方案。具体通过二分来弄即可。

1.7.5 时空复杂度

空间复杂度 $\mathrm{O}(\mathrm{N}^2)$, 时间复杂度 $\mathrm{O}(\mathrm{costflow}(\mathrm{N},\mathrm{N}^2)+\mathrm{QlogN})$。

1.8 KNGHTMOV 骑士移动

1.8.1 题目大意

要求从(0,0)走到(x,y)，存在两种移动方法从(u,v)到(u+ax,u+ay)或(u+bx,u+by)，到达(x,y)后还能再走，问有多少种到达的方案。有 K 个障碍点不能走。

1.8.2 数据范围

$\mathrm{K}\le 15$, 坐标绝对值 ≤ 500.

1.8.3 关键词

数学计算

1.8.4 题解

我们可以进行分类，首先对于(ax,ay)与(bx,by)如果线性无关的话，那么解方程可以解出使用 a 的次数和使用 b 的次数，然后再容斥原理表示至少走到哪些点的方案，然后用组合公式即可。

如果线性相关，那么我们可以只考虑一维坐标，因为另外一维是没有用的。

然后首先查看能不能走到一个环且这个环能走到终点，如果有就是无限解，否则必然存在拓扑图，我们就可以沿着拓扑图来维护答案。

而至于能不能走到一个环，我们只需要先确定 S 可达点以及可达 T 的点，然后 tarjan 一下即可。

1.8.5 时空复杂度

空间复杂度 $O(\text{绝对值}^2)$, 时间复杂度 $O(2^K + \text{绝对值}^2)$.

试题泛做(2)

2.1 Short

2.1.1 题目大意

给定 n,k,求满足以下约束的数对(a,b)个数：n<a,b<k,(a-n)(b-n)|(ab-n).

2.1.2 数据范围

$N \leq 10^5, k \leq 10^{18}$.

2.1.3 关键词

分类讨论

2.1.4 题解

首先，如果 n=0，特殊情况，特别处理，需要用到双精度；

然后，先有(a-n)(b-n)<(ab-n),因此需要(a-n)(b-n)*2<=(ab-n)才有可能整除，通过列式我们可以得出（假设 a<b)a<=4n,因此我们可以暴力枚举 a，然后由 k(a-n)(b-n)=(ab-n)得到 b 关于 a 的表达式，然后直接枚举 a，枚举因子即可；

但是，只有上面那一步的话还是会超时的，我们需要进一步的考虑。

当 a 较大时，实际上 k 的范围会下降得很快，因此我们可以选择枚举 k 而不是枚举因子，这样当分解值设于 a-n=3000 左右就能比较优秀的通过此题了。

2.1.5 时空复杂度

空间复杂度 $O(N)$, 时间复杂度 $O(3000 * N + N * 1000)$.

2.2 Counting Hexagons

2.2.1 题目大意

要求从[1,n]中选出 6 个可相同的数，要求较小的 5 个数的和大于最大的那个数，并且较小的 5 个数均小于等于 X，最大的那个大于等于 L，且不能用 K 个数相同，问方案数。

2.2.2 数据范围

$N \le 10^9, N - L \le 100, X < L.$

2.2.3 关键词

数位 DP

2.2.4 题解

这题如果能想到数位 DP 的话实际上就很好做了。

首先枚举最大的边，然后 F[i][j][k][l][ll]表示做到第 I 个数位（从低位起），总和与 L 目前的大小关系，最大的与 X 目前的大小关系，5 个数的相同关系，进位情况。维护这个 DP，并且进行一些常数优化就可以通过此题了。

2.2.5 时空复杂度

空间复杂度 $O(\log N * 2^6 * 5)$, 时间复杂度 $O(100 * \log N * 2^6 * 5 * 2^5)$.

2.3 Rectangle Query

2.3.1 题目大意

需要在平面内维护三种操作：

1.插入一个矩形；

2.删除某个之前插入的矩形；

3.询问之前插入的矩形中与当前矩形相交的个数；

2.3.2 数据范围

$Q \le 10^5.$

2.3.3 关键词

Cdq 分治、树状数组

2.3.4 题解

对于这个我们可以采取 cdq 分治来避免过大的空间开销。

然后考虑两个矩形(x1,y1,x2,y2)与(x3,y3,x4,y4)相交的情况，我们可以从反面入手：当且仅当 x1>x4 或 x2<x3 或 y1>y4 或 y2<y3 时不相交；

因此，考虑整个流程：

先进行 cdq 分治；

然后先忽略 y 求 x1>x4 或 x2<x3 的答案，可用树状数组来求；

再求 x1<=x4 且 x2>=x3 时 y1>y4 或 y2<y3 的答案，这个只需要先用快排并通过动态修改就同样可以使用树状数组来维护了。

2.3.5 时空复杂度

空间复杂度 $O(N)$，时间复杂度 $O(N\log^2 N)$.

2.4 Fibonacci Numbers on Tree

2.4.1 题目大意

要求维护一棵树，支持三个操作：

1.一条链上的点权按顺序增加 f[i],i 为链上第 I 个点，f[i]为斐波那契数列第 I 项；

2.询问子树点权和；

3.询问链上点权和。

N 个点，M 次操作。

2.4.2 数据范围

$N, M \le 10^5$.

2.4.3 关键词

剖分序

2.4.4 题解

首先考虑如何维护链和子树信息。

由于本题并没有树的形态变化，实际上我们可以先进行树链剖分，然后进行 dfs 序时优先遍历重儿子，这样就能在O（$\log^2 N$）维护链信息，O(logN)维护子树信息了。

然后，就是维护 fib，实际上如果我们维护 fib 数列的前两项的值，那么我们可以事先算出对应 N 项的和以及第 N 项的值与前两项的系数关系，这样就能支持不同的标记的合并了。

结合上面的方法，通过预处理出第 N 项值与前两项的系数关系以及构造剖分序，就可以高速通过此题了。

2.4.5 时空复杂度

空间复杂度 $O(N)$，时间复杂度 $O(N\log^2 N)$.

2.5 Room Corner

2.5.1 题目大意

给定一个封闭且边平行于某一坐标轴的简单多边形，在每个往外凸的角上有一个小朋友，然后多次询问某两个小朋友相遇最短时间。小朋友必须摸着边走，并且只有同一边上两个小朋友一起移动才能交换彼此的位置。

多次询问。

2.5.2 数据范围

不超过 2500 行以及 2500 列，询问次数 $\leq 10^4$.

2.5.3 关键词

二分

2.5.4 题解

这道题，显然可以转化为：一个圈上有 M 个节点，然后问两个节点的最短“距离”（按原题走法的距离）。因此，首先我们可以通过在原题中沿着边界走先找出 M 个节点的相对顺序以及相对距离。

紧接着对于询问某两点距离，我们发现实际上可以二分出最后是在哪一条边上相遇，从而能够计算出最短距离。

因此，实际上对于此题，我们需要先预处理维护出外围环的信息，然后二分在哪边相遇从而得到最优值。

2.5.5 时空复杂度

空间复杂度 $O(2500 * 2500)$, 时间复杂度 $O(2500 * 2500 + 10^4 * \log(2500 * 2500))$.

2.6 Observing the Tree

2.6.1 题目大意

给定一个 N 个节点的树，M 次操作：

1.将一条链的点权依次增加 a，a+b，a+2b……;

2.询问链上点权和;

3.返回到第 X 次修改之前。

2.6.2 数据范围

$N, M \leq 10^5$.

2.6.3 关键词

可持久化树链剖分

2.6.4 题解

首先只考虑前两个操作：

我们可以进行树链剖分，然后线段树每个节点记着 a,b 表示这里增加了一个以 a 为首项公差为 b 的等差数列的标记；然后直接使用带延迟标记的线段树即可解决这一道题。

而增加了第三个操作，我们可以直接进行可持久化来解决。

而为了优化空间，我们实际上可以使用永久化标记。首先给每个点记录一个 sum 表示这个点的子节点中所有标记对答案的贡献；同时记着完整覆盖这一个节点的标记的等差数列 a,b；维护答案时，只需要沿路加上所有的 sum 以及等差数列 a,b 对答案的贡献即可。空间可以优化一倍。

2.6.5 时空复杂度

空间复杂度 $O(N\log^2 N)$，时间复杂度 $O(N\log^2 N)$.

试题泛做(3)

3.1 Chef and Graph Queries

3.1.1 题目大意

给定一个 N 个点 M 条边的无向图。Q 次询问，每次询问为只保留 Li 到 Ri 的边时的连通块数量。

3.1.2 数据范围

$N, M, Q \leq 200000.$

3.1.3 关键词

可持久化线段树、动态树

3.1.4 题解

对于此题，考虑现在维护了以 L 为左端点的所有答案，考虑新插入一条编号为 L-1 的边，那么容易发现只要当右端点大于等于某个值的时候，L-1 这条边实际上就没有作用了，也就是说 L-1 号边连通的两顶点本来就联通了。因此，我们实际上可以不断维护一个最小权生成树，而权值就是编号。这样就能找到不作出贡献的那个分隔点。

接下来就是套用可持久化线段树就能维护所有可能询问的信息了。

3.1.5 时空复杂度

空间复杂度 $O(N\log N)$, 时间复杂度 $O(N\log N$）。

3.2 The Street

3.2.1 题目大意

N 个商店，M 次操作：

1.对于第 I 到第 J 的商店，I 位置新增价值 a 的商品，I+1 新增价值 a+b 的商品依次类推；

2.对于第 I 到第 J 的商店，I 位置商品增加税费 a，I+1 增加税费 a+b.....

3.询问在第 I 个商店出售物品中最贵的那个，需加上税费。

3.2.2 数据范围

$N \le 10^9, M \le 3 * 10^5$.

3.2.3 关键词

线段树

3.2.4 题解

首先因为税费维护比较简单，我们直接忽略即可。

然后，比较容易得到一个比较暴力的方法：直接在线段树上的每个顶点上维护一个凸包，这样就可以做到$O(N\log^2 N)$的复杂度了，理论上已经可以通过此题。

但其实我们还可以做一些方法上的优化，实际上我们可以在线段树的每个节点上记录一个数对(a，b)，表示整个区间内的商店可以与 a+b*i 取 maxx。

对于询问，由于询问单点，把所经过所有节点询问一次取最优即可。

而对于修改，我们可以考虑对于原本(a,b)与新增(a',b')作比较，如果一个比另一个整段优或劣，直接忽略；否则必然有其中一个节点完全更优或更劣，那么通过修改当前节点 a,b,然后只去遍历不是完全更优的那个子节点就可以了。

3.2.5 时空复杂度

空间复杂度 $O(N)$，时间复杂度 $O(N\log^2 N)$.

3.3 Counting on a directed graph

3.3.1 题目大意

N 个点 M 条边的有向图，询问合法无序对(x,y)的个数:存在除起点为不相交的 1-x 的路径与 1-y 的路径。

3.3.2 数据范围

$N \le 10^5, M \le 5 * 10^5$.

3.3.3 关键词

必经点定理

3.3.4 题解

这个问题实际上可以比较容易的转化为：x 与 y 存不存在除 1 以外的公共必经点。

直接套用半必经点与必经点定理构造出最近必经点树，然后树上统计答案即可。

而关于半必经点与必经点的内容，可以参照李煜东老师于 wc2014 的论文。

3.3.5 时空复杂度

空间复杂度 O(N)，时间复杂度 O(NlogN 或 Nα(N)).

3.4 Chef and Balanced Strings

3.4.1 题目大意

首先定义平衡串：平衡串中所有字母的出现次数均为偶数。

给定一个长度为 N 的字符串。

然后 Q 次询问，每次询问左端点与右端点在 L,R 范围内的所有平衡串的长度的 type 次方和。强制在线。

3.4.2 数据范围

$N \leq 10^5, Q \leq 10^5.$

3.4.3 关键词

分块

3.4.4 题解

首先对于出现次数为偶数这一点，我们只需要先做一次前缀和，然后根据各个字母出现的奇偶性来分好类，那么所有平衡串都能等价于两个各字母出现次数奇偶性相同的节点。

然后我们可以进行分块，先预处理所有整块之间的答案，然后再维护以各个块结束位置的部分和，记录的是这个位置前的各个位置的分类情况。

对于询问 a,b,假设完整包含了 l 到 r 块，l-1 块右端点为 rl,r+1 块左端点为 lr,那么先得到了块 l 到 r 的答案。

然后再处理多出来的部分，一个是 a 到 rl 的答案，这个直接暴力并记录一个类别数组即可；另一个是右端点在 lr 到 b 的答案，这个我们通过事先准备好的前缀和然后同样是暴力维护即可。

3.4.5 时空复杂度

空间复杂度 O（Nsqrt(N)),时间复杂度 O(Nsqrt(N)).

3.5 FN

3.5.1 题目大意

找到最小的 n，使得 fn=C(%P),P 为质数，fn 为斐波那契数列第 N 项。

3.5.2 数据范围

$P \le 2 * 10^9. (P\%10)$是完全平方数。

3.5.3 关键词

二次剩余、二次互反律，“大步小步”

3.5.4 题解

这一题我们可以先去推导出斐波那契数列的通项公式，然后发现公式里面存在一个无理数 sqrt(5)。然后，我们回过来看一下约束条件，(P%10)为完全平方数也就是 1,4,9.然后我们根据二次互反律可以得出关于这些数 sqrt(5)都可以找到%P 意义下的对应值。因此，我们现在公式可以变为$fn = a^n + b^n$.

进一步分析，ab=1,因此$fn = a^n + \left(\frac{1}{a}\right)^n$.要求 fn=C.然后我们可以两边乘上$a^n$,并设$t = a^n$.于是，问题变成求解二次方程，可以继续用求根公式，这里需要检验存不存在二次剩余。

然后就可以得出对应的 t。

之后使用“大步小步”算法反解出 n 即可。

3.5.5 时空复杂度

空间复杂度 $O(\mathrm{sqrt}(P))$, 时间复杂度 $O(\mathrm{sqrt}(P) + \log^2 n)$.

3.6 DEG3MAXT

3.6.1 题目大意

给定一个 N 个点 M 条边的无向图，保证任意节点数大于 9 的连通块的点连通度均为 1.求 3-度限制最大生成树的权值和和方案数。

3.6.2 数据范围

$N \le 100.$

3.6.3 关键词

双连通分量、状压 DP、树形 DP

3.6.4 题解

首先根据任意节点数大于 9 的连通块点联通度均为 1 这一点，我们可以推出不存在点数大于 9 的双连通分量。

因此实际上我们可以将双连通分量看做一个点来处理，然后原图抽象为一棵树，于是就可以直接在这棵树上做一次树形 DP 即可。

那么，现在考虑对于一个双连通分量内的处理方法。

我们考虑记录各个节点的度数。

然后我们选取第一个度数不为 0 的节点 X。

如果 degre[x]=1,那么我们枚举它连接的那条边更新状态值；

否则，我们可以将原本的状态集合分成两部分，然后通过枚举 x 在各部分的度数情况进行合并即可。

然后对于上述这两种情况综合起来做一个状态压缩 DP 即可。

另外，我们可以先预处理所有可能成为树的度数状态从而优化效率。

3.6.5 时空复杂度

空间复杂度 $O\left(4^9 * \frac{N}{9}\right)$，

时间复杂度 $O(4^9 * 2^9 * 3 + 9N)$. 实际远没有达到这个复杂度。

试题泛做(4)

4.1 Cool Numbers

4.1.1 题目大意

如果一个数 n 是 cool number，当且仅当 n 数位和为 sum，存在一到三个数位总和为 s 且$(\mathrm{sum}-\mathrm{s})^{\mathrm{s}}$是 n 的倍数。求小于等于 N 且最大以及大于 N 且最小的 cool number。

4.1.2 数据范围

$\mathrm{N} \le 10^{1000}$.

4.1.3 关键词

分类讨论、打表、分析最大值

4.1.4 题解

首先，对于这一题我们可以先进行分类讨论：

1.如果这个数只有一到三个非零数位，那么显然是 cool number，而对于这一类，要找到小于等于 N 且最大的，只需要从低位开始修改非零位，而大于 N 且最小的，则是从低位开始，每找到一个非零位，就改为零，对前面的位进行加 1 操作，直到非零位不到三位，复杂度为 O(logN)。

2.如果这个数超过三个非零数位，那么我们有$(\mathrm{sum}-\mathrm{s})^{\mathrm{s}} \ge \mathrm{n}$,那么假设 n 的位数是 g，那么有$(9\mathrm{g}-27)^{27} \ge 10^{\mathrm{g}}$,解得$\mathrm{g} \le 80$.也就是说 n 位数最多只有 80 位。因此，我们可以枚举 sum-s 最多为 720，然后直接找出sum^{27}的因子里面不超过 80 位的然后再进行检验，总共只有10^4级别的数。但是直接完整的枚举完会超时，因此我们可以枚举到 sum-s 为 300，超过 300 的则打表，只有 46 个。于是就可以在 1s 内完成预处理。然后对于询问，直接二分即可。

综合上面两类情况，比较就能得到答案了。

4.1.5 时空复杂度

空间复杂度 $\mathrm{O}(10^4 * 80)$,时间复杂度 $\mathrm{O}(\mathrm{logN} + 300 * 80 * 10^2)$.

4.2 Expected Maximum Matching

4.2.1 题目大意

给定一个二分图，左边有 n 个点，右边有 m 个点，给定左边第 I 个点与右边第 j 个点有边的概率，问最大匹配的期望值。

4.2.2 数据范围

$N \le 5, m \le 100.$

4.2.3 关键词

DP、找规律

4.2.4 题解

对于这一题，我们需要先去发现一些规律。

我们来考虑比较暴力的方法：记录下右边所有点当前可能的匹配情况。也就是总共最多有2^5种可能匹配，然后2^{2^5}种状态数。

然后我们来暴力 BFS 一下，会发现真正存在的情况只有10^3级别。

因此，我们只需要记录这 10^3 的状态，然后右边一个一个点的进行 DP 即可。

4.2.5 时空复杂度

空间复杂度 $O(10^3 * m * 32)$, 时间复杂度 $O(10^3 * m * 32)$.

4.3 Two Magicians

4.3.1 题目大意

在一个 n 点 m 边的无向图里面，一开始 A 在 1 号点，B 在 2 号点，每个人有 p 次瞬移机会，轮流移动，A 先手。移动分三步（假设当前为 A）：

1.A 在其所在连通块随意移动，如果停在了 B 所在位置，则 A 胜利；

2.A 必须增加一条原本没有的边，无法增加则失败；

3.如果 A 还有瞬移机会，则可以选择使用一次，瞬移到任意房间。

问胜者是谁。

4.3.2 数据范围

$N \le 7777, M \le 10000, p \le 10000.$

4.3.3 关键词

博弈论、找规律

4.3.4 题解

对于此题，我们先考虑如果直接从必败态推必胜态，那么要记录什么状态。

首先当前移动的人所在位置 wz1 以及目标所在位置 wz2，然后我们通过分析可以知道首先 wz1 与 wz2 不可能相同（相同可立刻结束），然后 wz1 与 wz2 具体在哪并不重要，重要的是其所在的连通块的一些信息。通过分析我们可以知道是点数，因为点数直接决定着当联通不同连通块时候的边数的变化。而点数真正有用的只是奇偶，因此我们记录当前移动者所在块的奇偶 s1 以及另一玩家的 s2。然后需要记录总连通块内可加边数的奇偶 edge，这个是为了不改变位置时增加无用边的操作。还要记录奇连通块数目 odd 和偶联通块数量 even，以及当前先手的瞬移次数 p1 和后手的瞬移次数 p2。

然后，我们通过各种情况的分析，可以发现瞬移双方最多共用一次。

对于这个我们可以分类讨论：

首先，不同连通块数量只有两个的时候，瞬移没有任何意义，忽略；

因此我们可以通过考虑三个连通块的情况来分析。

而再进一步分析，通过分析自己所在块奇偶以及对手所在块奇偶以及另外那块的奇偶可发现，只有自己与对手所在都是奇数，而剩余那些块是偶数才有意义，而由此也可得出最多只会共用一次瞬移，通过这个可以进一步优化。

这里，我们会发现虽然有些情况可以到达新状态，但是我们可以发现那些移动并非很迫切，实际上可以放到之后再去移动。这些情况主要是出现在多于三个连通块时的剩余块有奇数的情况。

然后，只是这样还是通过不了的。

然后，通过奇特的找规律方法，我们可以发现 odd 和 even 大于 20 的情况都可以等价转化为 20 内的情况，这样就能通过此题了。

4.3.5 时空复杂度

空间复杂度 O(20*20*2*2*2*2),时间复杂度 O(20*20*2*2*2*2).

4.4 A Game of Thrones

4.4.1 题目大意

有 N 种不同数字，每种数字出现 ci 次，双方轮流操作。然后先手者先选择一个数，作为当前数 X，并擦去。然后后手选择另一个数 Y，要求 Y 与 X 只相差一个质因子，然后当前数改为 Y，擦去 Y，换人操作。不能操作者失败。问如果有先手必胜策略，则一开始最小选多少。

4.4.2 数据范围

$\mathrm{N} \leq 500, 数 \leq 10^{18}, \mathrm{ci} \leq 10^{9}$.

4.4.3 关键词

Miller-rabin,匹配

4.4.4 题解

首先，我们先进行预处理操作，也就是连边。

而连边这里，只需要我们枚举任意两个数，然后 miller-rabin 检验相除是否为素数即可。

然后，我们很容易发现这是一个二分图。

并且只要我们一开始选择一个不一定在最大匹配内的点，那么就必然有先手必胜。

因此，我们可以做两次最大匹配（准确点说是三次，还有一开始全部边连上找最大匹配的一次），每次先选取一边把所有边加进去，然后另一边则从大到小加边，如果到某个时候发现必须把剩下的全部加进去才能达到最大匹配，那么就找到最小值。然后综合两次取个最优值即可。

4.4.5 时空复杂度

空间复杂度 $O(N^2)$, 时间复杂度 $O(\mathrm{maxflow}(N, N^2))$.

4.5 CNTDSETS

4.5.1 题目大意

在 N 维空间，询问有多少个不同点集使得最远点的切比雪夫距离恰好为 D。不同点集指不能通过平移得到。

4.5.2 数据范围

$N \le 10^3, D \le 10^9$.

4.5.3 关键词

容斥原理

4.5.4 题解

这一题，首先我们将恰好为 D 转化为<=D.通过求解 D 与 D-1 就能得到恰好为 D 的答案了。

然后对于<=D,我们可以求至少有 K 维小于 D 的方案，这个可以用$2^{D^K*(D+1)^{N-K}}$得出，然后再进行容斥原理就能得到想要的答案了。

4.5.5 时空复杂度

空间复杂度 O(N)，时间复杂度 O(NlogN)。

4.6 TAPAIR

4.6.1 题目大意

给定一个 N 点 M 边的简单无向图。问有多少种删边无序二元组(a,b)使得删除这两条边之后原图不连通。

4.6.2 数据范围

$N \leq 10^5, M \leq 3 * 10^5$.

4.6.3 关键词

异或

4.6.4 题解

这一类题目比较经典。

我们可以给每一条非树边随机一个权值 W，然后每条树边的权值为覆盖其的非树边的权值异或和。然后如果删除的两条边，有一条权为 0 或两条异或权为 0 则不连通。

因此，我们只需要先 dfs 一次，得到非树边和树边，然后给非树边赋权后，再 dfs 一次求解树边的权即可。

4.6.5 时空复杂度

空间复杂度 O(N+M),时间复杂度 O(N+M).

试题泛做(5)

5.1 Little Elephant and Colored Coins

5.1.1 题目大意

有 N 种硬币，价值 Vi，颜色 Ci，有无限多个。
询问 Q 次，问恰好组成 S 美元中，最多有多少种颜色。

5.1.2 数据范围

$N \le 30, Vi \le 2 * 10^5, Q \le 2 * 10^5, Ci \le 10^9.$

5.1.3 关键词

求模、环、最短路径、DP

5.1.4 题解

这题与去年 CF 上一题集训队作业是比较类似的。

我们可以先找出 minn=min(Vi),用 f[i][j]表示%minn 为 I 时且当前有 j 种颜色时候最少要加上 f[i][j]*minn 才可恰好组成。因为 minn 是其中一个 Vi，所以当 f[i][j]可行时，f[i][j]+1 一定也可行（或者说会变成 f[i][j+1]+1,本身没有太多关系）。

因此，我们现在问题就是维护这个 f[i][j],要维护也比较简单。

首先是一个 DP，对于所有颜色相同的先 DP 转移一下。然后还需要对于颜色相同时影响 I 的贡献。这里我们可以使用最短路径来维护，但是基于这题特殊性，实际上这题的更新关系是若干个不相交的环，我们只需要围绕着各个环更新两圈就可以全部更新出来了。

综合以上这两步，询问时只需要从小到大枚举 j 看看存不存在 f[i][j]<S/minn 的即可。

5.1.5 时空复杂度

空间复杂度 O(N*Vi),时间复杂度 O(N*Vi+Q*N).

5.2 Making Change

5.2.1 题目大意

有 N 种不同面值的硬币，面值为 Di，保证任意两个 Di 互质，为最终组成 C 的不同硬币组成方案数。

5.2.2 数据范围

$N \le 50, C \le 10^{100}, Di \le 500.$

5.2.3 关键词

母函数

5.2.4 题解

首先，由题目，我们可以根据母函数列出式子 $f(x) = \left(x^{di*0} + x^{di*1} + \cdots.\right) * \left(x^{d(i+1)*0} + \cdots\right) * \ldots.$

X^C 项的系数即为答案。

然后利用恒等式，我们可以转化为 $\prod_{d\in D}\sum_i x^{di} = \prod_{d\in D}\frac{1}{1-x^d} = \frac{1}{1-x^n}\prod_{d\in D}\frac{1}{\sum_{i<d} x^i}$，

然后利用部分分式分解，可转化为 $\frac{A(x)}{(1-x)^n} + \sum_{d\in D}\frac{B_d(x)}{\sum_{i<d} x^i}$。

然后关于求 Bd(x),通过各种奇怪转化，$\frac{B_d(x)}{\sum_{i<d} x^i} = B_d(x)(1-x) * \sum_i x^{di}$，

而 $B_d(x) = \sum_i b_i x^i$.

而 $b_i = \sum (k+1) a_{i-jk} = b_{i-j} + \sum a_k - n\, a_{(i-j)-j(n-1)}$,

这里，$\sum a_i w_d^i * \sum i * w_d^{ji} = \sum b_i w_d^i$ 。

通过这里就能算出 B 部分的答案了。

而对于 A 部分，我们可以知道 A(x)中x^i系数为度数为 n 未知数为 i 的多项式。

这个通过知道该多项式 0-n 的取值以此计算出 x^i 的系数即可。具体求法可参照杜瑜皓老师《多项式及其求和》一文。

5.2.5 时空复杂度

空间复杂度 $O(nd)$, 时间复杂度 $O(n^2 * d)$.

5.3 Equivalent Suffix Tries

5.3.1 题目大意

给定一个字符串 S，问有多少个构造出来的后缀字母树与 S 同构的字符串。

5.3.2 数据范围

$|S| \le 10^5$.

5.3.3 关键词

后缀数组、hash

5.3.4 题解

对于这一题，我们可以发现一点：不同字母数是固定的。因此答案必然是 K*26*25*....的形式，因此，我们的问题就是找出 K。

而找出 K，实际上就等价于询问最长被包含的后缀的可能情况。因此，我们可以先进行一次后缀数组，然后找出最长被包含的后缀 A。

然后，再根据后缀 A-1 与其余的最长公共前缀 len 得知后缀 A 的某一位的禁止情况。

然后接下来就是对于所有可能的情况（与 A 的最长公共前缀>=len-1 且结束位的下一位不被禁止的后缀 B），然后统计后|S|-A+1 位的可能情况，这里可以 hash+map 来维护。

而根据 B 与 A 的公共前缀进行一段一段的复制再加上最后多出那一段分段相加得出后|S|-A+1 位的 hash 值，然后存进 map 即可。

5.3.5 时空复杂度

空间复杂度 O(N),时间复杂度 O(NlogN)。

5.4 Dynamic GCD

5.4.1 题目大意

N 个节点的树，M 次操作：

1.询问一条链上权值的 gcd 值；

2.将一条链上节点权值+d。

5.4.2 数据范围

$N, M \le 50000$.

5.4.3 关键词

Gcd 的转化，树链剖分

5.4.4 题解

我们先不考虑树的情况，考虑对于一个序列怎么做。

首先我们有 gcd(a,b)=gcd(a,a-b). 因此，我们对于一段 gcd 的维护可以等价于 gcd(a,b-a,c-b,...),而这里的话，由于是一段加，因此 delta 变的只是单点问题，因此可以直接线段树维护。而对于原值的变化则由于只是单点询问，因此也可以简单的通过打标记实现。

那么，回到树上，也是类似的，每个重儿子记着 v[a]-v[fa[a]].并且对于每个节点记着原本值。因此，套用树链剖分，就可以解决本题的修改与询问问题。

5.4.5 时空复杂度

空间复杂度 $\mathrm{O(N)}$,时间复杂度 $\mathrm{O(Nlog^2}N)$。

5.5 Evil Book

5.5.1 题目大意

需要存储 666 点魔法值。

而世界上有 N 个对手，打败第 I 个对手可以获得 Mi 的魔法值，但需要付出 Ci 的努力。每个对手只能打败一次。

而也可以选择消耗 X 点魔法，使某个对手 Ci 与 Mi 均乘上 1/3.

初始没有魔法，问最小需要的努力值。

5.5.2 数据范围

$\mathrm{Ci,Mi} \le 10^9, \mathrm{X} < 666. \mathrm{N} \le 10.$

5.5.3 关键词

搜索

5.5.4 题解

这题首先，我们可以记录当前的击败对手的集合、当前魔法值、当前努力值。

而为了优化速度，我们可以一层一层的弄，即先考虑在某人身上只使用一次魔法的，然后结束之后，就考虑使用两次魔法，以此往上。

然后，并通过记录一些剩下的总魔法值以及当前最优之类的进行最优性剪枝与可行性剪

枝。然后就可以通过此题了。

5.5.5 时空复杂度

空间复杂度 $\mathrm{O}(\mathrm{N})$，时间复杂度 $\mathrm{O}(2^{\mathrm{N}} * \mathrm{logMi} * \mathrm{N} * \mathrm{C})$. C 是奇怪的东西。。。

5.6 Ciel and Earthquake

5.6.1 题目大意

一个 N*M 的图，四相邻节点有 p 概率连边，问最后(1,1)与(N,M)连通的概率。

5.6.2 数据范围

$\mathrm{N} \le 10^9, \mathrm{M} \le 8.$

5.6.3 关键词

找规律、连通性 DP

5.6.4 题解

首先考虑这一题的暴力。

我们可以记录当前轮廓线上的点以及(1,1)的连通情况，然后逐点移动的进行连通性 DP。通过预处理出合法状态以及转移情况从而优化速率，可以跑出 M<=40 的情况。

而接下来，一种比较正常的方法，就是根据连通性 DP 的转移状态，记录处理完一行之后的转移情况，然后套用矩阵乘法，只可惜状态数目过多，这种方法不能通过。

这时候，通过打表可以发现 Ans[i]/Ans[i-1]趋于一个常数，Ans[i]为(i,m)与(1,1)联通的概率。

因此，我们只需要暴力求出 Ans[40]与 Ans[39]，然后结合快速幂就能求出答案了。

5.6.5 时空复杂度

O(状态数*M*4),O(状态数*M*40*4+logN).

试题泛做(6)

6.1 Two k-Convex Polygons

6.1.1 题目大意

有 N 根棍子，需要选出 2K 根拼成两个凸 K 边形。

6.1.2 数据范围

$N \le 1000, k \le 10.$

6.1.3 关键词

搜索

6.1.4 题解

这一题，首先我们可以对棍子长度排序。

然后，分两种情况：

1.其中一个凸多边形最长的边比另外一个最短的还要短，那么直接枚举两个最大值判断即可，因为其余相应的 K-1 条边必然是排名紧接着的；

2.不符合上面的情况的话，我们可以直接枚举一个起点，那么从这个点往后 2K 个就是选出的棍子，问题就变成能不能凑成两个凸多边形，这个我们只需要进行搜索再配合剪枝即可，例如最大那条一定是一个多边形最大的，另一个多边形最大的绝对不在较小的 K 个里面，如果当前最长边较大那个多边形已经符合了的话，且另一个多边形最长边确定之后，那么直接给最小的给第一个多边形即可，综合这些优化就可以通过此题了。

6.1.5 时空复杂度

空间复杂度 $O(n)$,时间复杂度 $O\left(n^2 + 2^{2k} * n\right)$.

6.2 Count Special Matrices

6.2.1 题目大意

这个矩阵如果满足以下条件就称它是特殊的：

$A[x][x] = 0$ for $1 \le x \le N$.

$A[x][y] = A[y][x] > 0$ for $1 \le x < y \le N$.

$A[x][y] \le \max(A[x][z], A[z][y])$ for $1 \le x, y, z \le N$.

$A[x][y] \in \{1, 2, ..., N-2\}$ for $1 \le x < y \le N$.

任意 $k \in \{1, 2, ..., N-2\}$ 存在 $x, y \in \{1, 2, ..., N\}$, $A[x][y] = k$.
问大小为 N 的合法矩阵数。

6.2.2 数据范围

$N \le 10^7$.

6.2.3 关键词

组合

6.2.4 题解

这题我们可以去考虑边从小到大的加。然后假设加到权值为 K 的，然后如果加在其中两个连通块间，那么我们可以发现由 $A[x][y] \le \max(A[x][z], A[z][y])$ for $1 \le x, y, z \le N$ 可以推导出这两个连通块间所有边均为 K。

因此，这就简化了很多：问题变成先选择一个权值去出现两次，然后去计算这些权值的边分别连接哪些连通块的答案数，推导一下式子就可以出来了。

6.2.5 时空复杂度

空间复杂度 O(N),时间复杂度 O(N)。

6.3 Prime Distance On Tree

6.3.1 题目大意

问在树中等概率选取两个点，使得距离为质数概率。

6.3.2 数据范围

$N \leq 50000$.

6.3.3 关键词

点分治，fft

6.3.4 题解

首先忽略概率，实际就是问距离为质数的点对。

点对问题，很自然的，我们可以去考虑点分治。

然后点分治之后，考虑一个根，我们就可以直接计算出各个子树到根的各个距离的点数，发现只需要进行一次多项式乘法就可以得到答案了，因此，只需要用一次 fft 来求多项式乘法即可通过此题。

6.3.5 时空复杂度

空间复杂度 $O(N)$, 时间复杂度 $O(N\log^2 N)$.

6.4 Music & Lyrics

6.4.1 题目大意

询问 W 个单词在一段文章当中的出现次数。

6.4.2 数据范围

$W \leq 500$, 单词总长 ≤ 500000, 文章总长 $\leq 5 * 10^6$.

6.4.3 关键词

AC 自动机、fail 树

6.4.4 题解

这一题实际上只要对单词构建一个 AC 自动机，然后读入文章的时候在 AC 自动机对应节点更新，最后再遍历一些 fail 树统计子树和即可。

6.4.5 时空复杂度

空间复杂度 $O(5 * 10^6)$，时间复杂度 $O(5 * 10^6)$.

6.5 Easy Exam

6.5.1 题目大意

设想你有一个有 K 面的骰子，每个面上的数字分别是 1 到 K。同时给你两个参数 L 和 F(0<L<=K)。

你将这个骰子掷 N 次。令 ai 为掷出数字 i 的次数。求 a1^F*a2^F*...*aL^F 的期望值。

6.5.2 数据范围

$N, K \leq 10^9, L * F \leq 20000, F \leq 1000.$

6.5.3 关键词

fft

6.5.4 题解

首先对于 K 是 2003 的倍数，我们可以直接输出 0.因为分母的K^N中出现 2003 的次数不会少于分子，因此分母一定与 2003 不互质，故可直接输 0.

首先对于ai^F中的 ai 我们可以拆成很多个 1 相加，假设 F=2，而由三个一相加（设 x=y=z=1），则$(x+y+z)^2 = x^2 + y^2 + z^2 + 2xy + 2yz + 2xz$.那么对于 x^2 一项，我们只需一个单位时间，而 xy 则需要两个单位时间。那么如果我们先预处理出 f[i][j]表示 i 次方需要 j 个单位时间的方案数。那么，我们只需要把此作为多项式，然后求这个多项式的 L 次方，x^i 次方的系数则是需要 i 个单位时间的方案数，然后其余完全任意就可以算出这一项的贡献了。

而至于 f[i][j]一项，则可以通过 DP 来预处理得出。

6.5.5 时空复杂度

空间复杂度 $O(F^2)$, 时间复杂度 $O(N\log N)$.

6.6 A game on a graph

6.6.1 题目大意

在一个无向图上两个人轮流玩游戏，A 先行。先选择一个起点，然后 A 沿边走到一个点，B 再沿边走，不能到达重复点，不能走者输，问先手胜利的起点数。

6.6.2 数据范围

$N \le 2000, M \le 300000.$

6.6.3 关键词

带花树算法、一般图匹配

6.6.4 题解

这题我们可以采取匹配来做：

可以发现，只要起点可能不在一个最大匹配上，那么必然是先手必胜点，因为对方每一步总会到达一个最大匹配点（否则不是最大匹配），而自己只需再走到对应匹配点即可。

因此，我们只需要先进行一次带花树算法，找出最大匹配。

然后尝试从所有未盖点出发，走偶边走到的点全部均可以作为未盖点，标记。

最后，扫描一下统计答案即可。

6.6.5 时空复杂度

空间复杂度 O(N+M),时间复杂度 O(NM).

试题泛做(7)

7.1 Inverse Binomial Coefficient

7.1.1 题目大意

求最小的 K，使得$C(2^N-1,K)=R\ (\mathrm{mod}\ 2^N)$.

7.1.2 数据范围

$1\le N\le 120, 1\le T\le 100, 0\le R<2^N$.

7.1.3 关键词

递推、高精度、组合数学

7.1.4 题解

定义 fact2(a)=1*3*5*.....*X,X≤a 且 X 为满足条件的最大的奇数。
然后，我们有：

$C(2^n-1,K)=\frac{fact2(2^n-1)}{fact2(K)*fact2(2^n-1-K)}*C\left(2^{n-1}-1,\frac{K}{2}\right)$.①

通过这一个，我们可以知道所有的$C(2^N-1,K)$都一定为奇数。
又有：
$C(2^n-1,2K+X)=(-1)^{K+X}*C(2^{n-1}-1,K)(\%2^n)$.
那么，可以推出：
$C(2^n-1,2K)+C(2^n-1,2K+1)=2^n(\%2^{n+1})$.
那么，由这几点，我们可以得到一个求 K 的方法：

```
K=（R mod 4) div 2.
For i=3 to n
  Par=K mod 2
  If C(2^(j-1)-1,K)mod 2^j != R mod 2^j
        K=K xor 1
  K=K*2+Par
Return K
```

那么，现在的问题就是快速求$C(2^n-1，K)\mathrm{mod}\ 2^m$.
这时，可以利用①来弄。
那么，问题就是处理 fact2(a)了。
我们有$fact2(2u)=sgn*\left(fact2(2j)^{b(r,j,u)},1<=j<=r\right)(\mathrm{mod}\ 2^{2r+1})$.
Sgn 为 1 或-1，可以通过 mod 4 的值得到。

然后$b(r,j,u)=\frac{u}{j}*\left(\frac{u^2-i^2}{j^2-i^2},1<=i<=r,i!=j\right)$

然后，对逆元还有求 mod 的东西处理一下，并通过预处理，就可以快速求出 b(r,j,u).

接着，还有最后一个问题，快速求 a^b.

这个，我们先可以对 a 进行质因数分解，这样我们只需要预处理所有的质数就可以了（只有 29 个）。

然后对于p^b的求法。

我们可以使用分组的方法，将 b 表示为：$\mathrm{Sigma}\left(Ai*(2^H)^i\right)$的形式，然后通过一些预处理，预处理出 power_p[i][Ai]的值即可，就可以以$O\left(\frac{N}{H}\right)$的时间求解一次 p^b 了。取 H=15 效果比较好。

最后，再用上高精度，并通过适当的优化常数就可以通过此题了。

7.1.5 时空复杂度

空间复杂度 $O(N)$,时间复杂度 $O\left(\frac{N}{H}*N\right)$.

7.2 String Query

7.2.1 题目大意

维护一个串，要求支持在串最前面、最中间以及最后面的插入删除操作，并询问一个字符串 S 在串中出现次数。

7.2.2 数据范围

操作次数<=150000,总长度<=1500000.

7.2.3 关键词

后缀平衡树、重量平衡树

7.2.4 题解

我们先考虑一个简化后的问题：

维护一个数据结构，允许在字符串最前面插入字母，并维护任意两个后缀之间的相对关系。

这个实际上就是要维护动态后缀数组。

而在最前面插入字母，就是插入一个新的后缀而已，我们只需要能维护任意两个后缀之间的相对关系，剩下来的就是普通的平衡树插入。

而要维护任意两个后缀之间的相对关系，我们可以给每一个后缀一个 tag，使得满足后缀排名越前 tag 越小就好了。我们只需要维护一个 treap，可以保证深度以及子树大小为 O(logN）级别，那么每棵子树的 tag 都可以用一个区间表示[l,r]，然后子树的根 tag 取 mid=(l+r)//2,然后左子树就是[l,mid]，右子树就是[mid,r]，仍然可以保证精度，这样就可以解决了。

那么回到原题。

我们维护一个数据结构 SL=L+R，L,R 都为一棵后缀平衡树。这样的话，我们就可以维护最前面与最后面的动态插入删除了，当 L 或 R 为空时，把整个串均分即可，均摊仍是 O（logN）。

而至于要维护中间插入删除的话，我们再维护一个数据结构 G=SL+SR。保证 SL 最后面一个一定是中间即可，即对于每次插入删除动态维护一下即可。

对于询问，在串内就直接在平衡树中处理，在串间，就取出来做 kmp 匹配即可，由于取出的最长长度是 2*|S|-2，因此仍是 O（|S|）级别的。

最后复杂度为 O((N+|S|)logN).

适当优化常数可通过此题。

当然，也可以选择通过倍增而使得复杂度为$\mathrm{O}(\mathrm{N}\log^2 N)$.

7.2.5 时空复杂度

空间复杂度 O(N),时间复杂度 O((N+|S|)logN).

7.3 The Baking Business

7.3.1 题目大意

S 次操作：

I 产品编号[.大小编号] 省编号[.城市编号[.地区编号]] 性别 年龄 出售数量

这一行包括了出售的产品细节，位置细节以及顾客性别和年龄，以空格相隔。产品细节例如 6.2 表示第 6 号产品，2 号大小。下一个是位置例如 8.18.4 代表 8 号省的 18 号城市的 4 号区域。然后顾客的性别为 M 或者 F，年龄从 1 到 90。注意所有的编号都是从 0 开始。还需要注意方括号内的部分是可选的，因为某些出售的这些信息丢失了。出售数量不会超过 100。

询问的格式为：

Q 产品编号[.大小编号] 省编号[.城市编号[.地区编号]] 性别 起始年龄[-结束年龄]

这询问在该范围下的出售总数。如果可选部分缺失的话，那么它意味着询问在该限制下的所有出售总数。对于年龄参数来说，如果结束年龄缺失，那么这个询问只询问年龄=起始年龄的顾客的购买总数，否则就是所有在此年龄范围内的顾客的购买总数。比较特殊的是如果产品编号为-1，意思为所有的商品，类似的如果省份为-1 那么就是所有的省份。除此之外其它参数不会为-1.

7.3.2 数据范围

有 10 种产品，每种都有 3 种不同的大小。有 10 个省份，每个省份可以被划分为 20 个城市，每个城市又可以被划分成 5 个地区。S<=100000

7.3.3 关键词

暴力

7.3.4 题解

对于这一题，我们只需要记录各种编号以及各种年龄的进行暴力更新，然后对于询问则逐一年龄进行询问答案即可。

7.3.5 时空复杂度

空间复杂度 O(10*3*10*20*5*2*100),时间复杂度 O(|S|*100).

7.4 Sine Partition Function

7.4.1 题目大意

给出 N,M,X，求 f(N,M,X).

F(N,M,X)=sin(k1*x)sin(k2*x)…sin(km*x),k1+k2+…+km=N.

7.4.2 数据范围

$M \le 30, N \le 10^9$.

7.4.3 关键词

矩阵乘法、三角函数

7.4.4 题解

这一题，看到 N 的范围，我们可以尝试往矩阵乘法方向想。

然后，我们考虑如何进行暴力的 DP 转移。

$\mathrm{Sin}(\mathrm{ki} * \mathrm{x}) = \sin\big((\mathrm{ki}-1+1)\mathrm{x}\big) = \sin\big((\mathrm{ki}-1)\mathrm{x}\big)\cos\mathrm{x} + \cos\big((\mathrm{ki}-1)\mathrm{x}\big)\sin\mathrm{x} = \sin\big((\mathrm{ki}-1)\mathrm{x}\big)(2\cos\mathrm{x}) + \cos\big((\mathrm{ki}-1)\mathrm{x}\big)\sin\mathrm{x} - \sin\big((\mathrm{ki}-1)\mathrm{x}\big)\cos\mathrm{x} = \sin\big((\mathrm{ki}-1)\mathrm{x}\big)(2\cos\mathrm{x}) - \sin\big((\mathrm{ki}-2)\mathrm{x}\big)$.

再考虑 Ki=1 的情况则可以进行转移。

直接记着所以的 sin(ki*x)(N-1)与 sin(ki*x)(N)的情况，然后矩阵乘法优化即可通过此题。

7.4.5 时空复杂度

空间复杂度 $O(M^2)$, 时间复杂度 $O(M^3 logN)$.

7.5 Colored Domino Tilings and Cutsontest

7.5.1 题目大意

一个棋盘覆盖的染色是指：在棋盘上填上小写字母，使得每个格子有且仅有一个相邻格子的字母和他的一样。

棋盘的割是指一条竖直或水平的直线将棋盘分成两半，而且这两半都是合法的棋盘覆盖染色。

要求对一个 N*M 的棋盘进行覆盖，使得割最小前提下染色数最小。

7.5.2 数据范围

$N, M \le 500$.

7.5.3 关键词

分类讨论

7.5.4 题解

对这题我们可以进行分类讨论：

1.N*M 为奇数，显然无解；

然后假设 N<=M.

2.N=1，M=2K，合法方案只有一种，需要两种颜色，割数为 K；

3.N=2，M=2K，只需要将每一列两个颜色都不同，而对于某一行和 N=1，M=2K 情况相同即可；

4.N=2，M=2K+1，直接将第一列弄一个竖排的，然后就与 N=2，M=2K 情况相同；

5.N=3，M=4，我们可以构造最优方案：

1 1 2 3

2 3 2 3

2 3 1 1；

6.N=3，M=2K，割数为 1，染色数为 3：

我们考虑第一行完全横着放，然后第二行两边竖着，中间横着就能使割数为 1，并且不存在割数为 0 的方案；

7.N=4，M=2K+1，割数为 1，染色数为 3：

和第 6 种类似，前两行在最左放一竖排，然后剩余横放，后两行最右放一竖排，剩余横放，达到最优；

8.N=4，M=4，割数为 2，染色数为 3：

1 1 2 1

3 3 2 1

1 2 3 3

1 2 1 1

9.N=4，M=2K，割数为 1，染色数为 3：

类似下面摆法即可：

1 1 2 2 3 3

2 2 3 3 1 1

3 1 1 2 2 3

3 2 2 1 1 3.

先确定方法，然后染色问题可以自己找一个合适方法进行染色；

10.N=6，M=6，割数为 1，染色数为 3：

1 1 2 1 1 2

3 3 2 3 3 2

2 1 1 2 1 1

2 3 3 2 3 2

1 2 1 1 3 2

1 2 3 3 1 1

11.N=2K+1，M=2K’，割数为 0，染色数为 3：

这个我们可以选择 N=5，M=6 的棋盘然后进行插入一些放大（同样先不考虑染色，最后再选取一个合适的染色方案）；

12.N=2K,M=2K’,割数为 0，染色数为 3；

与 11 类似，只不过我们是选取 N=6，M=8 作为基本然后再往里插入放大。

而对于 11、12 两种情况，M 的增加都是选取合适的插横排机会，而 N 增加则可以选择不断拆除最后一行的横排拆成两个竖排的方式增加。

7.5.5 时空复杂度

空间复杂度 O(NM),时间复杂度 O(NM).

7.6 Luckdays

7.6.1 题目大意

定义了一个数列：

S[1]=A

S[2]=B

S[i]=(X*S[i-1]+Y*S[i-2]+Z)mod P, 对于 i>=3

Q 次询问，对每个询问 L[i],R[i],你要求出满足 L[i]<=k<=R[i]且 S[k]=C 的 k 的个数。

7.6.2 数据范围

$P \le 10^7, Q \le 20000.$

7.6.3 关键词

大步小步

7.6.4 题解

这题首先我们可以根据数对(Si,SI+1)然后去找出循环节。然后找出循环节之后，问题就变成找出所有的(C,X)的数对的位置。而这两个问题的解决方法是类似的。

我们可以采取大步小步。

假设就处理回到(A,B)这一步，另外那步也是类似的。

我们可以设$(A,B) * T^{KL-G} = (A,B)$。T 是转移矩阵。

然后就可以变成$(A,B) * T^{KL} = (A,B) * T^{G}$.

然后我们预处理出所有的 KL 与 G 次方的，然后用大步小步和 hash 处理找出答案，最后设 T=P^1.5，就能得到最优的复杂度 O(Psqrt(P)).

而询问，则先对于整个循环节部分处理，多出部分再进行二分即可。

7.6.5 时空复杂度

空间复杂度 $O(P^{1.5})$, 时间复杂度 $O(P^{1.5} + Q\log P)$.

试题泛做(8)

8.1 Children Trips

8.1.1 题目大意

在一棵边权只可能是 1 或 2 的树上，多次询问从 S 出发到达 T 步长为 G 不允许停在边中的最少步数是多少。

8.1.2 数据范围

$\mathrm{N}, \mathrm{M} \leq 10^5$.

8.1.3 关键词

分块

8.1.4 题解

这一题，首先我们可以发现，对于一条链，我们实际上直接拆成两段一直往父亲走的链是不影响的，只需要不断往上贪心，最后那步注意要不要合并即可。

因此，我么只考虑一直往父亲走的链。

这题我们可以分块，一旦存在一个不经过任何一个关键点到达此点 A 的点 B 距离 A 超过 sqrt(N)的话，则把 A 作为关键点。

可以证明这么弄之后，关键点不会超过(sqrt(N)).

然后，我们预处理出从各个节点出发往上走到上一个关键点，假设速度为 X，最终可以超过关键点多少以及步数，然后再预处理各个关键点往上多少可以到达哪个点。

把这些预处理出来后，对于询问就可以采取先走到关键点，然后再不断在关键点之间跳转，直到跳到 lca 那个点，计算超出值即可。

8.1.5 时空复杂度

空间复杂度 O(NlogN),时间复杂度 O(Nsqrt(N)).

8.2 Union on Tree

8.2.1 题目大意

在一棵 N 点树上，Q 次询问：有 K 个哨兵在位置 Xi 以及拥有保护半径 ri(距离 xi ri 距离内的点都可保护)，统计保护点数目。

8.2.2 数据范围

$\mathrm{N}, \mathrm{Q} \leq 50000, \mathrm{Sumk} \leq 500000.$

8.2.3 关键词

点分治、可持久化线段树、虚树

8.2.4 题解

先考虑一个简化版问题，问树上有多少个点与 A 距离小于 B。

这一个我们可以采用点分治来做，假设现在与 X 为根，我们先记录与 X 距离为<=D 的点数，以及各点离 X 的距离，然后就可以求出当前树中与 A 距离小于 B 的点，但需除去与 A 同一联通块的点，接着继续点分治下去即可。

还有一个问题，询问 A 子节点中距离 A 小于 B 的点数。

这个可以按 DFS 序来弄一个可持久化线段树，线段树下标为深度，这样就可以处理了。

然后，考虑询问，很自然的，我们可以先把虚树弄出来。

然后考虑虚树边上的答案统计情况。

可以找到一个点分开两段，使得这个点往下的部分从下面走比较优，上面则相反。

然后利用求整棵树的答案以及求子树的答案我们可以通过一些求补的方法得到我们想要的答案。

8.2.5 时空复杂度

空间复杂度 O(NlogN),时间复杂度 O((Sumk+N)logN).

8.3 Xor Queries

8.3.1 题目大意

给定一个初始时为空的整数序列(元素由 1 开始标号)以及一些询问：

类型 0：在数组最后加入数字 x。

类型 1：在区间 L..R 中找到数字 y，最大化(x xor y),xor 表示按位异或。

类型 2：删除数组最后 k 个元素。

类型 3：在区间 L..R 中，统计小于等于 x 的元素个数。

类型 4：在区间 L..R 中，找到第 k 小的数(第 k 个顺序统计量)。

8.3.2 数据范围

$M \le 5 * 10^5$.

8.3.3 关键词

可持久化字母树

8.3.4 题解

首先，考虑如果最大化(x xor y)，很显然，只用构造一个字母树然后在字母树上遍历即可。

然后，我们观察到每次删除的都是最后面的节点并且在最后插入，而询问区间问题。因此，只需要加上一个可持久化就可以解决这题的各种操作了。

8.3.5 时空复杂度

空间复杂度 O(MlogM),时间复杂度 O(NlogN).

8.4 Ranka

8.4.1 题目大意

两个玩家 A、B 在玩合作围棋。

设当前玩家 A，A 可以选择放一个棋子或不操作，如果放一个棋子后，使得对方某些棋子没有“气”了（具体参考围棋攻略），则吃掉，否则如果不能吃掉并且放这个棋会导致自己被吃掉则不允许放。

求一场有 N 步操作且不出现重复棋盘的方案。

8.4.2 数据范围

N=10000.

8.4.3 关键词

构造

8.4.4 题解

这里需要注意到两个特别的地方：可以不下；当全场布满别人的棋子（除一格外）,你在剩余格放棋子，则对方棋子全部消失。

因此，我们只需要不断重复：一方一直不动，等别人放剩一个的时候放棋子，然后对方全吃掉，更换两者的地位，然后重复之前操作。

这样就能构造出 N 为 10000 的情况了。

不过，需要注意放棋子顺序的影响，防止到达重复状态。

8.4.5 时空复杂度

空间复杂度 O(N),时间复杂度 O(N).

8.5 Ciel and password cracking

8.5.1 题目大意

（题目貌似简洁不了。。直接复制。。）

Ciel 打算借用其中恰好 K 个来分别破解 K 个密码。

这 C 个电脑中心和 Ciel 的餐馆在同一条街上，以 Ciel 的餐馆为坐标轴零点，第 i 个电脑中心的坐标为 X_i，坐标的正负表示所在的方向。

Ciel 将会选择 K 个不同的电脑中心去破解她的 K 个密码，她会以任意顺序去遍历它们，从她的餐馆出发，最后回到她的餐馆。

Ciel 的行走速度为 V，即从坐标 A 走到坐标 B 需要消耗$|A-B|/V$ 单位时间。

在第 j 个电脑中心，Ciel 可以使用最多 P_j 台电脑，每台电脑需要消耗 T_j 单位时间去枚举验证一种密码。这些电脑被一种奇怪的网络连接着，于是 Ciel 的计划将以下面的步骤进行：

1. 选择一部分电脑（不超过 P_j 台），这一步不消耗时间。
2. 在一台电脑中安装破解程序，这一步不消耗时间。
3. 把破解程序通过网络传输到其他被选择的电脑中，这一步的消耗时间在后文会详述。
4. 在所有被传输的电脑上同时运行破解程序，这一步的消耗时间在后文会详述。
5. 一旦其中一台电脑成功破解了密码，那么这里的工作将会全部停止，然后她将会奔赴下一个电脑中心。

1.**破解程序细节**

破解程序每次会随机验证一个没有被验证过的密码，对于第 j 个电脑中心的电脑，验证一次将消耗 T_j 单位的时间。

假如某一次验证的密码成功了，程序将通知 Ciel 并终止。

电脑和电脑之间并没有共享信息，即可能会验证在其他电脑上已经验证过的密码。

2.**网络连接细节**

通过网络连接，已经装有破解程序的电脑，可以将破解程序备份传输到没有安装的电脑上。

连接所需要的时间是随机的，这个随机值呈**指数分布**，对于第 j 个电脑中心的电脑，均值为 Sj。

只有已经装有破解程序的电脑才可以与其他电脑连接，一旦连接完成，备份传输可视为不消耗时间。

在每个时刻，一台电脑只能尝试向一台电脑连接，但一台电脑可以同时接受多台电脑的连接，一旦其中一个连接成功确立，那么其他连接将会中断停止，那些电脑将会去尝试与其他电脑连接。

3.**连接方案细节**

可以为每台电脑预先制定一个连接列表，电脑每次会选择连接列表上的第一个没有装有破解程序的电脑进行连接。一旦连接确立或中断，它将立刻选择下一个连接目标。

在所有被选择的电脑都装有破解程序后，所有程序将同时开始运行。

Ciel 想要通过分配电脑中心，选择电脑，调整连接方案，来最小化期望所需时间。

8.5.2 数据范围

$\mathrm{C} \leq 1000, \mathrm{K} \leq 5. \mathrm{Ni}, \mathrm{Pi} \leq 10^{18}$,剩余为$10^{20}$级别

8.5.3 关键词

数学

8.5.4 题解

设 Time[i][j]为第 I 个密码在第 J 个电脑中心破解的时间，如果知道这个之后只需要简单的状态压缩 DP 即可。因此，就是解决 Time[i][j]的问题。

Time[i][j]=F(N,P,S,T),表示密码范围 N，电脑数量 P，连接时间 S，运算速度 T。

设 A(k)为 S=1 连接 K 电脑的时间，B（N,K）为 T=1 用 k 台电脑破解范围 N 的时间。

那么答案等于 min(S*A(k)+T*B(n,k)).这一个函数是一个单峰函数，因此我们可以利用三分求最终答案。

因此，问题变成球 A 与 B。

考虑 A(k),因为连接时间是指数分布，所有相互之间不影响。因此，可以得到$\mathrm{A(k)} = \sum_{i=1}^{k-1} \frac{1}{i}$

当 K 较大时可以用欧拉常数+ln 求答案。

考虑 B（N,K），我们可以得出式子$\mathrm{B(N,K)} = \sum_{i=1}^{n} \left(\frac{i}{N}\right)^{k}$

然后通过利用欧拉-迈克劳林公式以及伯努利数的东西，并通过忽略较大的项，套用公式就可以求出这一部分答案。

综合上面的步骤，这题就解决了。

8.5.5 时空复杂度

空间复杂度 $O(10^6)$,时间复杂度 $O\left(C * k * 2^{k} + C * k * \log N\right)$.

8.6 Max Circumference

8.6.1 题目大意

平面上三个点 A,B,C，可以在 N 个修改向量(xi,yi)中选择 K 个去改变 A 坐标，求 dist(A,B)+dist(A,C)的最大值。

8.6.2 数据范围

$N \le 500$.

8.6.3 关键词

转化、排序

8.6.4 题解

首先，我们要注意到一点，答案必然可以转化为 Ans=Ax+By.因为，考虑达到最大值的时候，轨迹是椭圆，我们只需要做一条切线就能得到这个 Ans 了。因此，这是一个等价转化。

知道这个之后就比较简单了，我么可以比较任意两个增加向量贡献相同时的 A 与 B，然后排好序。每到达一个关键向量，就更换大小关系并不断维护最大的 K 个的值即可。

比较得到最优解。

8.6.5 时空复杂度

空间复杂度 $O(N^2)$, 时间复杂度 $O(N^2 logN)$.

试题泛做(9)

9.1 Arithmetic Progressions

9.1.1 题目大意

给定一个 N 个整数的序列，问有多少种三元组(i,j,k)满足$1 \le i < j < k \le N$并且Aj-Ai=Ak-Aj。

9.1.2 数据范围

$N \le 10^5, Ai \le 30000.$

9.1.3 关键词

分块、fft

9.1.4 题解

这一题，我们考虑暴力算法：直接枚举一个 j，然后再枚举一个 i，然后根据预处理统计合法的 k 的数目。

这里，我们尝试使用分块来处理这一道题。

假设分为 T 块。

假设 j 在第 X 块([l,r])，假设 i 和 k 中至少有一个在第 X 块，我们可以枚举第 I 块内的点然后通过事先的预处理迅速求出答案，复杂度为$O\left(\frac{N^2}{T}\right)$.

那么假设两个都不在第 X 块，我们就可以把事先求好的两个方向的预处理[1,l-1]与[r+1,n]的各数出现情况，然后进行 fft 求出和的情况，然后再询问块内点即可。

这样结合常数上的一些问题，T 取 40-50 比较优。

9.1.5 时空复杂度

空间复杂度 $O(maxc * T + n)$,时间复杂度 $O\left(\frac{n^2}{T} + T * maxc * logmaxc\right)$.

9.2 Martial Arts

9.2.1 题目大意

A，B 队均有 N 名选手。

我们事先知道 A 队选手 I 与 B 队选手 j 对上的时候双方得分为 Aij 与 Bij。

那么，A 队希望安排比赛（每名选手至多参加一场）使得 sumA-sumB 最大，如有多种，就是 sumA 最大；而 B 队则希望 sumB-sumA 最大，如有多种，就是 sumB 最大。

B 队可以选择一名选手不比赛。

9.2.2 数据范围

$\mathrm{N} \le 100$。

9.2.3 关键词

KM 算法、匹配

9.2.4 题解

首先我们先注意到一点如果 sumA-sumB 固定时，A，B 队第二条件实际上是一样的。因此，当成相同。

然后，我们更新权值为 pair(a-b,a),假设对战表确定了，B 一定删除权最大的（第一关键字最大，如有多个则第二关键字最小）。

因此我们可以按照权从小到大排序，然后通过只插入更小的边并且强制匹配当前的边从而求解答案。

然后插入边，实际就是把原本一条-oo 变成正常，而强制匹配，则视为+oo。

因此，我们现在问题就是改变一条边的边权。

这里我们尝试使用 KM 算法，利用其中的 label 来修改。设改变边连 I,j.

然后直接删除匹配 i 和 match(i)的匹配边，然后 label[i]=maxj(w[i,j]-label[j]).然后再从 i 出发找一条增广路。从而高效修改。

另外需要注意一定的常数优化。

9.2.5 时空复杂度

空间复杂度 $\mathrm{O}(\mathrm{n}^2)$, 时间复杂度 $\mathrm{O}(\mathrm{n}^4)$.

9.3 Two Roads

9.3.1 题目大意

我们需要找出两条直线，使得$\sum d_i^2$最小。$\mathrm{d_i}$为第 i 个点距离两条直线的较小值。

9.3.2 数据范围

$N \leq 100$.

9.3.3 关键词

排序

9.3.4 题解

首先我们很容易发现，一定可以利用两条相互垂直的直线分成四个象限，并使相对的两个象限内的点归一条直线管。这一点很好证明，假设已经知道两条直线，那么根据两条直线所成角作出的角平分线就是那两条相互垂直的直线了。

因此，我们可以先枚举其中一条直线，容易知道这条直线一定是过其中两个点的（关键直线），然后把另一条直线从无穷远一边开始移动，不断使得某些点改象限，并不断维护最值即可。这个只需要利用各点在枚举直线的投影位置排序即可。

而至于问作一条直线与一个点集最小答案的话，只需要用一些二次函数的东西就可以推出来了，当然也可以求导。

9.3.5 时空复杂度

空间复杂度 $O(N)$,时间复杂度 $O(N^3 * \log N)$.

9.4 To Query or not to Query

9.4.1 题目大意

M 次操作：

1. 在 S 的末尾插入一个字符，S 的长度增加 1。
2. 删除 S 的第一个字符，S 的长度减少 1。

3.在每次操作之后 Chef 想知道当前串 S 的本质不同子串的个数。

9.4.2 数据范围

$M \leq 10^6$.

9.4.3 关键词

后缀数组

9.4.4 题解

这一题，首先我们先把所有插入操作读进来得到最终的字符串并后缀数组一次。

然后，对于此题，我们可以把字符串反向从而简化题目，即更改为结尾删除、开头插入。

然后考虑假设没有插入删除操作如何利用后缀数组求本质不同子串，只需要求Sum(n-sa[i]-height[i])即可。

因此，我们考虑插入与删除操作对这个 sum 的影响。

先考虑删除操作，实际上就是改变了上式的 n，并且删掉了一个后缀。但如果至少简单了删除一个后缀并减少 n 的话，显然是错的。因为有可能导致之后的后缀排名错误。因此，我们还需要把此时 n-sa[i]=height[i]的串全部删掉，确保这点之后就不会错了；

而插入操作，则直接插入一个后缀，并查看有没有 n-sa[i]=height[i]的串，删除即可。

通过这样维护，就能实时求解答案了。

9.4.5 时空复杂度

空间复杂度 O(N),时间复杂度 O(NlogN).

9.5 Make It Zero 3

9.5.1 题目大意

有一个数字 P 和一个长度为 N 的数组，记数组的第 i 个元素为 Ai。需要除去数组中所有非零的数字。每次操作可以选择三个整数 s,t 和 k ，对所有满足 s<=i<=t 的 Ai 进行变换 Ai=(k+Ai) mod P

数组的大小可能非常巨大，我们把它分成若干个子数组依次串联起来的结果。

第二行包含一个整数 M，表示子数组的个数。

接下来的 M 行，每行包含四个整数 Fi,Ci,Di 和 Li。子数组将由以下规则产生：

设 B[i][j]表示第 i 个数组第 j 个元素，有 B[i][1]=Fi 且当 1<j<=Li 时，有 B[i][j]=(B[i][j-1]*Ci+Di) mod P

你可以认为 N=L1+L2+...+LM

9.5.2 数据范围

$\mathrm{P} \le 10, \mathrm{M} \le 100, \mathrm{Li} \le 10^{15}$.

9.5.3 关键词

数学

9.5.4 题解

首先我们先进行一次差分，Bi=Ai-Ai-1,并加入 A0 与 An+1 均为 0.

然后我们发现实际上假设我们操作的区间是一系列[li,ri],那么其实把 li 与 ri 连边，我们只需要使各个连通块为树的时候就能达到答案了，修改的值可以从叶子节点一直往上。因此，问题变成使连通块尽量多而减少边数。

因此，我们意识到我们只需要知道每个数字有多少个这一点。

然后，对于原本的读入，我们可以根据循环节比较容易就能知道各个数字有多少个了。

然后，对于这些数字，显然所有 0 自我成环。

然后，1-9,2-8,3-7,4-6,5-5,配对。

从而最终最多只剩下 5 种非 0 数字而且还有一种为 1，假设为 1,2,3,4,5.

然后找出所有子序列使得和为 0 而不存在和为 0 的非空真子集，大概只有 30 个。

然后再通过暴力尝试找出所有的最简分裂情况，这个可以使用 DP 处理，大概只有 150 种。

然后接下来我们只需要先随意构造一个合法解，然后不断选取一些子序列合成更少的子序列，不断贪心下去即可。

9.5.5 时空复杂度

空间复杂度 $O(12^4)$, 时间复杂度 $O(12^4 * 150 + C)$. C 为奇怪的东西。

9.6 Chef and Tree Game

9.6.1 题目大意

在一棵树上，边有两种颜色。

两名玩家轮流行动；A 每一次删除一条还在树上的黑边，B 每次删去一条还在树上的白边；当一条边被删去之后，所有与根节点不连通的边也会一起被删去；当一名玩家无法行动时，他就输了。

问 A 先走的胜者与 B 先走的胜者。

9.6.2 数据范围

$N \leq 10^5$.

9.6.3 关键词

博弈、hackenbush

9.6.4 题解

首先这貌似是一个经典问题，叫做 hackenbush。具体可参考百度搜索。

此题，我们可以用一个实数来表示当前状态，当>0,均 A 胜;<0，均 B 胜；=0，后手胜。

然后不同子树合并上来，直接实数相加。

然后，再与当前子树根与父亲的连边进行合并。

可由下面伪代码得到：

```
ret=0
```

```
    for each vertex v which is child of n:
      ret1=dfs(v)
      if edge connecting u and v is 0:
          find smallest p such that ret1+p > 1
          ret += (ret1+p)/(1<<(p-1))
      else:
          find smallest p such that ret1-p < -1
          ret += (ret1-p)/(1<<(p-1))
```

最后，直接用 double 或 long double 精度都不够，因此我们可以记录 a，k 表示$\frac{a}{2^k}$来存储答案。而 a 则可以用一个二进制高精度数用 map 来存储表示。

9.6.5 时空复杂度

空间复杂度 $O(N\log N)$, 时间复杂度 $O(N\log^2 N)$.

试题泛做(10)

10.1 Billboards

10.1.1 题目大意

N 个路牌，使得任意连续 M 个中至少有 K 个广告并使广告数最少，问方案数。

10.1.2 数据范围

$K, M \leq 500, N \leq 10^9$.

10.1.3 关键词

杨氏图表、数学

10.1.4 题解

首先考虑 N%M=0.

我们可以将 M 个当成一组。

然后 N/M 行 M 列，同一行的第 I 个表示这一组里面第 I 个广告的位置，那么就是横的递增，竖的非递增。

于是就转换为经典的杨氏图表的东西，然后就是求杨氏图表方案数。

Ans=1

For i=1 to row

For j=1 to col

Ans*=(r+j-i)/hook(I,j)

r 为候选数范围。Hook(I,j)表示(I,j)的钩子数。（参照杨氏图表中对钩子的介绍）

然后再把可约去的直接忽略，就能较快得出答案了。

然后考虑 N%M！=0，如果 N%M<=M-K,那么每组前 N%M 都为 0；如果 N%M>=M-K,那么后面 M-N%M 都为 0，于是可等价于 N%M 的情况。

10.1.5 时空复杂度

空间复杂度 O(1),时间复杂度 O(KMlogMod).

10.2 Trial of Doom

10.2.1 题目大意

一个 N*M 矩阵，每个格子可能是 B 或 R。从(1,1)出发到达(n,m),可以移到 8 相邻任意一格，每离开一格，格子本身及四相邻节点变色，问存不存在全部变 B 到达终点的方案。

红色格子数目为 k。

10.2.2 数据范围

$N \le 40, N \le n, m \le 10^9, k \le 10000.$

10.2.3 关键词

数学、转化、高斯消元

10.2.4 题解

如果 N>1,我们可以证明任何的移动方案都是合法的，因为我们可以弄出无影响移动以及换色不移动的情况，具体画一下就知道了。

因此，问题就变成存不存在一个所有格子的有效换色情况使得全部变成 B 了。

然后我们可以尝试把所有红色全部移动第一列，而由(I,j)是红色等价于(i-1,j-1),(i,j-1),(i+1,j-1),(i,j-2)是红色这一点。然后又由一列的红色转移到第一列的状态是带有循环的，并且循环节不大，因此我们可以进行暴力求解。

然后就能把所有红色移到第一列，最后再由最后一列的格子充当变量并移动到第一列，然后 gauss 消元就能得到答案。而用 bitset 可以对速度进行优化。

对于 N=1 的情况，我们可以特殊考虑，根据走的步数的奇偶性判断解决。

10.2.5 时空复杂度

空间复杂度 $O(N^2 + K)$,时间复杂度 $O(k + n * Len)$，Len 为循环节长度。

10.3 Sereja and Subsegment Increasings

10.3.1 题目大意

给出一个序列 Ai(0<=Ai<4),要求进行尽量少的操作,使得所有 Ai=0.操作为选择(l,r)使得 Al 到 Ar->(Ai+1)%4.

10.3.2 数据范围

$N \le 10^5.$

10.3.3 关键词

贪心

10.3.4 题解

首先，我们可以转化为每次选择一段区间减一，使得最终为 0.允许某些数+4 的变化。

这个假设没有+4 的变化，我们只需要差分一下，ans=sum max(0,ai-ai-1)，这样救能得到答案了。

但是，这题允许某些数+4，从而使得某些可以 ai-a(i-1)比较大的变成负数了，从而减小答案。

因此，我们现在可以贪心的弄，一旦遇到一个 ai-a(i-1)大于 0 的，就看看能不能前面找一个原本的负数+4 后的答案更优的，如果有就加上，然后当前-4，以此来不断贪心调整答案即可。

10.3.5 时空复杂度

空间复杂度 O（N），时间复杂度 O（N）。

10.4 Dynamic Trees and Queries

10.4.1 题目大意

M 次操作维护有根树，强制在线：

1.插入节点；

2.删除子树；

3.子树权值+a；

4.询问子树权值和。

10.4.2 数据范围

$M \leq 50000.$

10.4.3 关键词

Dfs 序

10.4.4 题解

这一题，我们只需要用出入都计数的那种 dfs 序，然后这题就变成简单的 splay 操作了。

操作 1，就是插入一个节点；

操作 2 就是删除一段；

操作 3 就是一段添加；

操作 4 就是问一段的和；

只需要用一个带标记的 splay 就能通过了。

10.4.5 时空复杂度

空间复杂度 O(n),时间复杂度 O(nlogn).

10.5 Simple Queries

10.5.1 题目大意

一开始有 N 个整数的序列，M 次操作：

1.将下标为 x 的元素赋值为 y；

2.删除下标为 x 的元素；

3.在下标 x 的元素前插入新元素 y；

4.输出下标 l 到 r 范围内的不同元素；

5.对于 l 到 r 中不同元素求 Sum ai*aj*ak,i<j<k.

10.5.2 数据范围

$\mathrm{N}, \mathrm{M} \leq 10^5$

10.5.3 关键词

离线、二维线段树、splay

10.5.4 题解

首先，这一题我们可以先进行离线把最终的下标弄出来。

然后对于找出现在下标为 x 的点最终下标，则可以用一棵 splay 来维护即可；

然后要求找不同的数，我们可以给每个数标记一个二元组(pre,now)表示前一个相同的在哪和自己在哪，那么不同的就可以用 pre<l,l<=now<=r 来表示了。

因此，我们可以对此维护一棵二维线段树，并对每个数用一个 set 来维护位置关系，这样就可以维护不同的数目了。

而至于第五种操作的询问，实际上我们只需要记着一次方和、二次方和、三次方和和零次方和，然后通过简单的操作就能解决所要的答案了。

10.5.5 时空复杂度

空间复杂度 $\mathrm{O}(\mathrm{Nlog}^2\mathrm{N})$, 时间复杂度 $\mathrm{O}(\mathrm{Nlog}^2\mathrm{N})$.

10.6 Future of draughts

10.6.1 题目大意

给了 G 幅 Ni 个点 Mi 条边的无向图，然后 Q 次询问，每次询问 l,r,k,表示在第 l 到 r 幅图中，同时随机指定一个起点，然后选定一个非空子集，然后在选中的子集中，随机将棋子移动，问 K 步之内全部回到起点的方案数。

10.6.2 数据范围

G，$\mathrm{Ni} \le 50, \mathrm{Q} \le 2*10^5, \mathrm{k} \le 10^4$.

10.6.3 关键词

fft、大步小步、容斥原理

10.6.4 题解

这一题，我们可以用容斥原理来做。

假设已经求 P(l,j,k)表示从第 l 到第 J 幅图用步数不超过 K 步（可以某一步指定空集）全部回到起点的方案数。

那么根据容斥原理，$.\mathrm{Ans} = \sum_{i=0}^{k} C(k,i) * P(l,r,i) * (-1)^{k-i}$

而这条式子，我们可以整理一下，$.\mathrm{Ans} = \mathrm{k}! * \sum \frac{P(l,r,i)}{i!*(k-i)} * (-1)^{k-i}$

因此，我们可以用 fft 优化这条式子，而这题由于模的数是个奇怪的数，因此我们可以选取模另外三个“好”质数例如{167772161, 998244353, 1004535809}，之后用中国剩余定理合并，至于中国剩余定理的写法，一开始我想用一个很逗的写法（百科上的标准写法），不过感觉要写双精度。。之后，发现其实有简单得多的方法，假设 mod m1=k1,mod m2=k2,那么可以用 k2+(k1-k2)*inv(m2,m1)%m1*m2.这样就能避免双精度了。

然后问题就是求 P(I,j,k)了，而我们这里因为可以空集，实际上图与图之间没有太大关系，因此我们就是求某个图 K 步内回到起点的方案。

这个最暴力的方法，就是直接边数+度数的矩阵的 K 次方，但会超时得很 happy。

因此，我们可以采用大步小步，那么一开始先把所有的 KT 和 G(G<T)次方的矩阵全部弄出来。然后到后面求任意 K 的时候，就可以分为 mat[a][b][KT]*mat[b][a][G],这样就能快速合并出来了。

综合上面几步，加上一些常数优化，就能通过此题。

10.6.5 时空复杂度

空间复杂度 $\mathrm{O}(\mathrm{G} * \mathrm{N}^2 * \mathrm{sqrt}(\mathrm{N}) + \mathrm{G}^2 * \mathrm{k} + \mathrm{Q})$,

时间复杂度 $O(G * N^3 * sqrt(N) + G^2 * Klogk + Q)$.

试题泛做(11)

11.1 Something About Divisors

11.1.1 题目大意

对于给定的正整数 B 和 X，求满足条件的正整数 N 的个数:要求对于 N，至少存在一个数 D（N<D<=B) | N*X。

11.1.2 数据范围

$B \le 10^{12}, X \le 60.$

11.1.3 关键词

数学、容斥原理

11.1.4 题解

首先，这一题，我们可以先进行简单的转换：等价于求合法的 K 使得 K|N*X 且 $K < X, N \le \frac{B*K}{X}$.

那么我们现在可以对 K 进行枚举，然后找合法的 N 的数目。

我们由 K|N*X -> K/gcd(K,X) | N,我们设$C[k] = \frac{K}{gcd(k,X)}$,那么令 N=C[k]*M.

那么可推出$M <= \frac{B*K}{X*C[k]}$.设$\frac{B*K}{X*C[k]}$为 lim。

通过上面的式子我们就可以算出对于一个 K 来说合法的 N 的数目。
不过，显然会有重复。
因此，我们考虑假设有一个 J(>K)重复了，那么重复部分有多少。

J|N*X-> J|c[K]*M*X -> $\frac{J}{gcd(J,c[k]*X)}$|M->设 D[j]=$\frac{J}{gcd(J,c[k]*X)}$.

因此只属于 K 不属于 J(>K)的部分的答案等于$\lim - \sum(-1)^t * \frac{lim}{lcm(D[j1],...D[jt])}$

于是再对程序进行一定常数优化，就可以通过此题了。

11.1.5 时空复杂度

空间复杂度 O(X)，时间复杂度 $O(X * 2^X)$.

11.2 Shortest Circuit Evaluation

11.2.1 题目大意

给定一个只由 and、or 与括号组成的表达式，我们需要找出一个不改变值的计算顺序，使得计算次数期望最小。

11.2.2 数据范围

变量数 ≤ 1000.

11.2.3 关键词

排序

11.2.4 题解

这题，我们考虑一个括号内的答案如果计算好了，需要返回什么值给外面。首先一个是当前计算次数,用于统计最后答案;还有一个是真的概率由于合并;只需要这两个就足够了。

然后考虑很多个二元组(pi,ti)，我们要怎么安排顺序。

这时，我们可以先考虑两个二元组(p1,t1)与(p2,t2)，假设当前是以 or 操作连接，那么分别考虑以 1 在前和以 2 在前时候的期望操作次数，然后进行化简我们得到等价于 t1/p1 和 t2/p2 的相对大小来比较确定排序顺序，这个显然是满足传递性的；而对于 and，则也是类似的方法得到 t1/(1-p1)与 t2/(1-p2),然后排好序之后，按顺序合并即可。

11.2.5 时空复杂度

空间复杂度 O（N）,时间复杂度 O（NlogN）。

11.3 Cucumber Boy and Cucumber Girl

11.3.1 题目大意

对于所有 1 到 N 的排列 p，定义：

Tr(p)[i]=Ca,b[i][pi]

Cnt(a,b)表示 Tr(p)至少有一个奇元素的排列个数，定义数对(a,b)为好的当且仅当 cnt(a,b)是奇数。

计算好数对个数。

11.3.2 数据范围

$N \le 60, B \le 8000$。

11.3.3 关键词

行列式、Cauchy-Binet 公式

11.3.4 题解

首先 N=1 为特殊情况，特殊处理即可。

下面假设 N>1.

对于一个数对(a,b),我们设 Da,b[i][j]=(Ca,b[i][j]+1)%2.

那么对于一个排列至少有一个奇元素，当且仅当.

而剩余的累乘后不为 1 的即是全部偶元素，那么 cnt(a,b)=N!-Det(a,b) %2,又因为 N>=2,N!%2=0，故 Det(a,b)=cnt(a,b) %2,因此原题转换为 Det(a,b)为奇数的对(a,b)数。

但是直接求行列式仍然很慢，因此我们要改变计算的方法。

我们先用矩阵的相乘去得到 D 数组。

设第 N+1 列全是 1.那么

然后，应用 Cauchy-Binet 公式，我们定义 Qa,c 为矩阵 Qa 删除第 C 列。

那么.

那么我们只要预处理出所有的 detQa,c，deta,b 我们可以利用压位来优化处理。

那么，最后的问题就是求 detQa,c.

直接一个一个求显然会超时。

首先用 gauss-jordan 把 Qa 变成最简式。

然后如果 Qa 的秩小于 N，显然所有 detQa,c=0.

而等于 N 的时候，必然是单位矩阵加上一列（假设是第 K 列，这一列第 I 个元素为 Gi）。

DetQa,c=Gc,1<=c<k;

DetQa,k=1;

DetQa,c=0,c>k.

通过这三条式子就能快速得到答案了。

11.3.5 时空复杂度

空间复杂度 $O(N^2 * B)$,时间复杂度 $O(N^2 * B + B^2)$.

11.4 A New Door

11.4.1 题目大意

首先一开始除了一个矩形区域为白色以外，平面其他区域均为黑色。然后，有 N 个可能相交的圆范围内也变成黑色。问最终白色（矩形的边可以不管）的周长。

11.4.2 数据范围

$N \leq 5000.$

11.4.3 关键词

计算几何

11.4.4 题解

这题，我们只需要逐个逐个圆的处理，算出圆边上的不被覆盖周长即可。

而这个问题，只需要算出圆与圆之间的覆盖范围以及圆与矩形的覆盖范围，然后按照极角序来进行覆盖问题，统计不被任何一个覆盖的弧度和即可。

11.4.5 时空复杂度

空间复杂度 $O(N)$,时间复杂度 $O(N^2 * \log N)$.

11.5 Little Elephant and Boxes

11.5.1 题目大意

有 N 个盒子，可能有钱或钻石。打开第 I 个盒子，有 Pi 概率获得 Vi 美元，有 1-Pi 概率获得一颗钻石。然后有 M 个物品，第 I 个物品恰好花费 Cj 元与 Dj 颗钻石。问期望购买物品数。

11.5.2 数据范围

$N，M \leq 30.$

11.5.3 关键词

折半搜索、DP

11.5.4 题解

首先，我们可以进行一次 DP，f[i][j]表示买了 I 个物品用了 j 颗钻石至少花费多少钱。

然后，我们进行折半搜索，算前 N/2 个物品的所有情况及概率与后 N/2 个物品所有情况及概率。

然后，枚举前一半中有 A 颗钻石的情况以及后一半中有 B 颗钻石的情况，然后根据总钻石数 A+B，我们知道这时买的物品越多花费越多，因此，我们可以对分别的情况进行排序，然后进行一次单调扫描进行统计答案以及相应的概率即可。

11.5.5 时空复杂度

空间复杂度 $O\left(2^{\frac{n}{2}}\right)$, 时间复杂度 $O\left(2^{\frac{n}{2}} * n + m^3\right)$.

11.6 Selling Tickets

11.6.1 题目大意

有 N 道菜肴和 M 位顾客，每位顾客有两道喜爱的菜，每道菜只能供应给一个客人。现在问你最多能招待多少名客人，使得无论来的是哪些顾客，顾客都至少可以吃到一道他喜欢的菜。

11.6.2 数据范围

$N \leq 200, M \leq 500.$

11.6.3 关键词

分类讨论

11.6.4 题解

首先，这题我们可以把菜当成点，顾客当成边，然后问题变成边集最小为 K 时存在大小为 K 的边集而连接的点集大小>K,那么答案为 K-1。

然后对于这个，我们很容易发现我们实际要找到是一个|E|+1=|V|的最小集合。

然后，如果有度数为 1 的点，可直接删除不管。

因此，所有度数均大于 1，因此有三种情况：

1.两个点之间有三条链；

2.两个在某点上相交的环；

3.两个不相交环之间由一条简单路径连接。

对于这三种情况，我们分开处理，并且可以保证在第一种情况计算中直接把三条链分开统计求最大不影响答案，第二、三种即使两环相交在其余地方也不影响答案，即几个部分可分开处理。

而处理最需要注意的，则是注意走的是最小环，这一天可以通过 BFS 来保证，因为每一个环必然是一条前向边和多条树边组成。

11.6.5 时空复杂度

空间复杂度 $O(M)$, 时间复杂度 $O(N^3)$.

试题泛做(12)

12.1 RIN

12.1.1 题目大意

课业计划共包含 N 项课程，每项课程都需要在 M 个学期里的某一个完成。
一些课程有前置课程：对于所有的 i(1<=i<=K),A[i]是 B[i]的前置课程。
X[i][j]表示铃在学期 j 选修课程 i 所能得到的期望分数。
求所能得到期望分数的平均值的最大值。

12.1.2 数据范围

$\mathrm{M},\mathrm{N},\mathrm{K}\le 100.$

12.1.3 关键词

最小割

12.1.4 题解

这一题，我们可以采取最小割来做。

首先，我们计算出各个课程至少在第几个学期 minn[i]才可能完成。并求出 maxx[i]表示第 I 个课程最大的完成花费。

然后，我们对于一个课程 a，我们直接(S,a1),(a1,a2),…(am-1,am),(am,T)连边，如果 i<minn[a],那么(ai,ai+1)连无穷大的边，表示不能切；否则连 maxx[a]-cost；并且给(ai),(K(i+1))连无穷大的边，如果 a 是 K 的前置。

这样构边之后，再进行最小割即可。

12.1.5 时空复杂度

空间复杂度 O(NM+MK),时间复杂度 O(maxflow(NM,NM+MK)).

12.2 DIVIDEN

12.2.1 题目大意

尺规 N 等分 N 度角。

12.2.2 数据范围

操作数 ≤ 1000.

12.2.3 关键词

尺规作正五边形

12.2.4 题解

这题，首先根据常识，3n 度角均不可以尺规作图作出 3n 等分。

而对于其他，我们可以查阅百度百科，发现可以作出正五边形。

然后，正五边形外角 72 度，而常用角有 60 度，即可作出 12 度，即可作出 3 度，那么配合原本不为 3 度的角，也就可以作出 1 度，因此就能进行 N 等分了。

具体细节需要斟酌。比较恶心。

12.2.5 时空复杂度

空间复杂度 O(操作数)，时间复杂度 O(操作数)。

12.3 Queries on tree again!

12.3.1 题目大意

定义节点 u 到 v 的最短路径为连接 u, v 的边数最少的路径。

在一个环套树上有两种操作：

f u v：对 u 到 v 的最短路径上的所有边的权值取相反数。

? u v：在 u 到 v 的最短路径上，找到一个连续的边的集合，使得集合中边的权值之和最大。

N 为点数，M 为操作数。

12.3.2 数据范围

$N, M \leq 10^5$.

12.3.3 关键词

树链剖分、线段树

12.3.4 题解

这一题，我们只需要分别维护森林部分和环部分即可。

对于修改操作，则在森林部分通过树链剖分维护，而在环部分，则通过破环为链用线段树来维护即可。

对于询问操作，也是同样处理。

而具体处理，则分开树内以及跨树来分开处理。

对于树内，直接树链剖分；

对于跨树，则分别在各自的树内树剖，并在环上找出较短路并维护线段树的值即可。

而对于本题的操作，只需要记录 sum，maxl，maxr，ans 表示总权值，左起最大权值，右起最大权值，以及最大子段和。

并配合上延迟标记即可通过此题。

12.3.5 时空复杂度

空间复杂度 $O(N)$,时间复杂度 $O(N\log^2 N)$.

12.4 Gangsters of Treeland

12.4.1 题目大意

给定一棵树，然后有 M 次操作：

1.给一条到根路径染上一种新颜色；

2.询问一棵子树上的点到根的平均颜色段数。

12.4.2 数据范围

$N，M \le 10^5$.

12.4.3 关键词

树链剖分

12.4.4 题解

这一题，我们可以计算每一条边对其子树的贡献。

对于询问，则直接询问子树内每一条边的贡献，以及从子树的根到树的根上的颜色段数的贡献即可。

而对于修改，我们可以暴力修改每一条贡献可能会改变的边，因为对于每一个新增的颜色路径，自己新增的贡献值最多只有 O(logN).而其余都是把原本可能有的贡献变为没有。这个由重链来分析就可以得知了。因此，只需要快速找出需要改变的，暴力修改即可。

12.4.5 时空复杂度

空间复杂度 $O(N)$, 时间复杂度 $O(N\log^2 N)$.

12.5 Queries With Points

12.5.1 题目大意

平面上有 N 个不相交的简单多边形。
M 次询问，问一点在哪个多边形内。
强制在线。

12.5.2 数据范围

$N, M \le 10^5$, 总点数 $\le 3 * 10^5$.

12.5.3 关键词

射线法、可持久化 treap

12.5.4 题解

这题，我们可以先处理特殊情况：即在边上和在点上的情况，解决了这些情况之后，我们的射线法就可以采取竖直向上并忽略掉竖直的边。

然后，就是判断相交是不是奇数次，如果是，则输出最近相交的边所属的多边形，显然这就是答案，如果不是则不被任何一个包含。

那么，问题就是判断相交次数而已。

如果是离线的话，我们可以按照 x 坐标从小到大处理，然后把包含当前 x 坐标的边从低到高排序，然后询问时直接在平衡树上询问即可，修改的话，由于没有相交情况，因此只有简单的插入删除，也比较简单，这样就能解决这一问题了。

而本题强制在线，因此我们只需要维护一个可持久化 treap 就可以支持强制在线了。

12.5.5 时空复杂度

空间复杂度 O(NlogN),时间复杂度 O(NlogN).

试题泛做(13)

13.1 Push the Flow!

13.1.1 题目大意

给定一个 N 个点 M 条边的无向图，保证每一个点最多只属于一个环。
Q 次询问：
1.以 S 为源点、T 为汇点的最大流量；
2.改变一条边的边权。

13.1.2 数据范围

$\mathrm{N} \le 10^5, \mathrm{M}, \mathrm{Q} \le 2 * 10^5.$

13.1.3 关键词

树链剖分

13.1.4 题解

这一题，我们可以用树链剖分来处理。首先进行缩点。

然后，每一个点记着从它所属环的根节点走到它的父亲的根节点的流量（要求这个点是它父亲的重儿子）。

这样的话，我们通过一棵线段树一并记着环内的情况，就可以大概解决这一问题了。

对于询问，则根据树链剖分，对于重链上的已经维护好了直接得到答案，对于轻边的，则通过环内情况的答案的记录，同样可以在一个 log 的花费解决。

对于修改操作，则分开是环内抑或是环外，然后修改缩点后的树的所在节点与其重儿子的连边的值，如果是在环内，则再更新环内线段树的值。

结合这些部分，并通过一定的常数优化（例如：zkw 线段树），就可以通过此题了。

13.1.5 时空复杂度

空间复杂度 $\mathrm{O}(\mathrm{N})$,时间复杂度 $\mathrm{O}(\mathrm{N}\log^2\mathrm{N})$.

13.2 Team Sigma and Fibnacci

13.2.1 题目大意

给定 n，m，求.

13.2.2 数据范围

$N \le 10^{18}, m \le 10^5$.

13.2.3 关键词

母函数

13.2.4 题解

这一题可以通过母函数来解决。

这一个数列的母函数 $f(x) = 6g(x) * x^3 * (x^2 + 1)^3$.

$[x^n]g(x) = \frac{1}{3125} * \left[25 * C(n+5,5) * (fib[n+6] + 2 * fib[n+5]) + 150 * C(n+4,4) * fib[n+5] + 5 * C(7,2) * C(n+3,3) * (fib[n+4] + 2 * fib[n+3]) + 5 * C(8,3) * C(n+2,2) * fib[n+3] + C(9,4) * (n+1) * (fib[n+2] + 2 * fib[n+1]) + C(10,5) * fib[n+1]\right]$.

然后，直接用矩阵乘法求解即可。

而对于分数处理问题，可以采用暴力约分的方式解决。

13.2.5 时空复杂度

空间复杂度 O(1),时间复杂度 O(logn).

13.3 Flight Distance

13.3.1 题目大意

N 个节点 M 条边的简单无向图，边权为 Di，要改变尽量少的值的边权，使得 d[i][j]<=d[i][k]+d[k][j].

13.3.2 数据范围

$N \le 10, 1 \le Di \le 20$.

13.3.3 关键词

线性规划

13.3.4 题解

这题是典型的线性规划的题目，可以直接套用 simplex 解决。

即设变量 dist[i][j],以及 d+[i],d-[i]表示 i 到 j 最短路径长度以及第 I 条边增加值与减少值，然后通过通过三角不等式以及边权的限制，来列约束不等式。

不过，由于这题的 bi 是负数，需要先进行 simplex 的第一步转化，比较复杂，因此，我们可以通过换元来防止这一问题。

我们可以用 dist'[i][j]=20*sum[i][j]-dist[i][j],sum[i][j]表示 i 到 j 最短需要经过多少条边，20 为最大权值，显然这个变换仍能保证所有非负。D+'[i]=19-d+[i],D-'[i]=D[i]-1-d-[i],这些变换也同样能保证所有非负。并且这样变换之后，就保证了所有 bi 都非负，于是可以直接套用简单的 simplex。

不过，这一题还需要注意的，需要用到 simplex 的最小字典序方法， 不然会死循环。

13.3.5 时空复杂度

空间复杂度 $\mathrm{O}(\mathrm{N}^3)$,时间复杂度 $\mathrm{O}(\mathrm{simplex}(\mathrm{N}^2, N^3))$。

13.4 Find a Subsequence

13.4.1 题目大意

给定一个长度为 N 的数组，需要找出 5 个数使得相对大小顺序满足题目所给。

13.4.2 数据范围

$\mathrm{N} \leq 1000$.

13.4.3 关键词

线段树

13.4.4 题解

这道题，我们可以通过枚举+线段树解决。

我们可以枚举第二个与第四个数，然后进行贪心地选数，具体选数则可以用线段树快速找出。

举个例子，题目给的 23145.

那么，我们由给定排名第三的与第四的，首先找到第五的那个， 优先找最小的。

然后找第二那个，优先找最大的。之后再去找第一那个，需要小于第二那个且优先找最大的。

另外，为了优化，我们还可以用一个二维部分和来优化判断某个区间内某个大小范围内存不存在数。

13.4.5 时空复杂度

空间复杂度 $O(N^2)$, 时间复杂度 $O(N^2 * \log N)$.

13.5 Chefbook

13.5.1 题目大意

给定 M 个整数对(x,y),和对应的 Lx,y、Sx,y、Tx,y.
求出 N 个非负整数 Px 与 N 个非负整数 Qx。
定义：Wx,y=Lx,y+Px-Qy.
约束条件:Sx,y<=Wx,y<=Tx,y
要求最大。

13.5.2 数据范围

N<=100.

13.5.3 关键词

费用流、线性规划、对偶问题

13.5.4 题解

首先根据约束条件：Sx,y<= Lx,y+Px-Qy <=Tx,y，并射 Qy=Pn+y,我们可以得到：
Px-Pn+y<=Tx,y-Lx,y;
Pn+y-Px<=Lx,y-Sx,y;
然后我们求的目标函数为,Ci 为 Pi 出现的次数（度数）。
而前半为恒定值，因此我们只用考虑最大化后半。

然后根据对偶问题以及最优的时候必然恰好满足某一约束（上约束或下约束），我们可以以每一个 Pi 作为点，等式作为边，Ci 作为流量，原本的不等式右侧作为费用，从而就可以进行最小费用最大流了。

13.5.5 时空复杂度

空间复杂度 $O(N^2)$, 时间复杂度 $O\left(\mathrm{costflow}(N, N^2)\right)$.

13.6 Connect Points

13.6.1 题目大意

对于一个 N 个点 M 条边的无向图，询问是不是最大平面图。

13.6.2 数据范围

$\mathrm{N} \leq 7 * 10^4, \mathrm{M} \leq 3 * 10^5$.

13.6.3 关键词

最大平面图

13.6.4 题解

对于最大平面图，首先我们知道一些引理：

一个图是最大平面图，|E|必须等于 3|V|-6.

一个图是最大平面图，那么每个面必然是三角形。

因此，这一题，大概思路就是：逐个逐个三角形的找出来并进行压缩，最终再把三角形解回出来判断是否合法。具体我也不太清楚。。感觉需要参考一些平面图的相关知识。

13.6.5 时空复杂度

空间复杂度 O（N+M），时间复杂度 O（N+M）。

试题泛做(14)

14.1 Petya and Sequence

14.1.1 题目大意

给出一个长度为 n 的序列 A[0..n-1]，问是否存在一个长度为 n 的序列 B[0..n-1]满足：
-至少存在一个 0<=i<=n-1 满足 B[i]!=0
-对于任意 0<=j<=n-1 满足∑[0<=i<=n-1]A[i]*B[(i+j)mod n] = 0

14.1.2 数据范围

$\mathrm{N} \leq 3 * 10^4, -10^3 \leq \mathrm{Ai} \leq 10^3.$

14.1.3 关键词

分圆多项式、循环矩阵

14.1.4 题解

原题等价于询问一个循环矩阵 C={A0,A1,...An-1}是否满秩。
而循环矩阵的秩满足：$\mathrm{rank(A)} = \mathrm{n} - \mathrm{degree(gcd(f(x), x^n - 1))}.$
Degree 为最高次次数。
而上述式子也就是说只要 f(x)与$\mathrm{x^n} - 1$存在度数不为 0 的多项式，矩阵就不满秩。
于是可以用分圆多项式。
Fn(x)=$\prod_{i=0,\mathrm{gcd}(i,n)=1}^{n-1}(x - w_n^i)$.
并且满足几个性质：Fi(x)两两不同，Fi(x)不可约(即多项式不可以分解)，
于是原问题转换为存不存在 d|n 且 Fd(x)|f(x).
继续转换，得到 Fd(x)|f(x) -> $\mathrm{x^d} - 1$ |f(x)*$\prod_{p\text{是质数}，\mathrm{p|d}}(\mathrm{x}^{\frac{\mathrm{d}}{\mathrm{p}}} - 1)$.
于是就是求上述的倍数关系了，只需要对 d 进行质数分解然后累乘即可。而令.
于是就能求解原本问题了。

14.1.5 时空复杂度

空间复杂度 O(n),时间复杂度 O(5*n*d(n)),d(n)为 n 的因子数。

14.2 Query on a tree VI

14.2.1 题目大意

一棵 N 个节点的树，M 次操作：

0 u：询问有多少点与点 u 连通。两个点 u 和 v 是连通的，当且仅当 u 到 v 最短路径上的所有点(包括 u 和 v)颜色都相同。

1 u：切换点 u 的颜色(黑变白，白变黑)。

14.2.2 数据范围

$N, m \leq 10^5$.

14.2.3 关键词

树链剖分

14.2.4 题解

这题，依旧可以用树链剖分来做。

我们记录四个值：当前这个节点的子树的答案，当前节点的轻边的颜色 0 的答案和当前节点的轻边的颜色 1 的答案，以及 delta 表示延迟标记，可以用永久化标记实现。

然后对于询问的话，直接先找到最高的那个一路同色点，然后直接返回该点答案即可。

然后修改的话，则沿着重链往上走，一边更新重链的状态同时改变一些节点的轻边的状态值。具体只需要自己推导一下比较容易得到。

14.2.5 时空复杂度

空间复杂度 $O(n)$, 时间复杂度 $O(n\log^2 n)$.

14.3 Devu and Locks

14.3.1 题目大意

求有多少 N 位十进制数是 P 的倍数且每位之和小于等于 M，允许前导 0，答案对 998244353 取模。

14.3.2 数据范围

$1 \leq N \leq 10^9, 1 \leq P \leq 16.1 \leq MM \leq 15000$

14.3.3 关键词

fft

14.3.4 题解

这题，我们先考虑比较暴力的方法。

我们可以采取快速幂来解决这一题。

F[i][j]表示%P 的值为 I，总和为 J 的方案数。

然后，对于已知 N，求 2N 的话，则 P^2 的枚举即前 N 位 modP=i1，后 N 位 mod P=i2，然后再用 fft 求出新的数组。

这种方法的话，会超时。

然后，其实，这题我们可以不采取原本的快速幂的合并方法可能会更快。

我们可以先进行预处理看看%P=i 的各个数位分别有多少个，然后直接求出%P=i 的数位和为 K 时的方案数的数组，这个就是用 fft 来求解即可。

然后，在合并%P 不相等的数位的时候才用回原本的 P^2 的枚举，这样就能通过这题。

14.3.5 时空复杂度

空间复杂度 $O(P*MM)$, 时间复杂度 $O(P*MM*\log MM*\log N+C*P^2*MM)$.

14.4 Payton numbers

14.4.1 题目大意

定义三元组的乘法：

def multiply((a1,b1,c1), (a2,b2,c2)):

s = (a1a2 + b1b2 + c1c2) + (a1b2 + b1a2) + (c1 + c2)

t = floor[s/2] + 16(c1 + c2) - c1c2

A = (t - 2(a1b2 + b1a2) - (a1c2 + c1a2) + 33(a1 + a2)+ (b1b2 - a1a2))

B = (t - 5(a1b2 + b1a2) - (c1b2 + b1c2) + 33(b1 + b2)+ (2b1b2 + 4a1a2))

if s is even: return (A-540,B-540,24)

else: return (A-533,B-533,11)

定义 zero：若 x*任何 y=0，则称 x 是 zero

定义单位元，若 x*任何 y=y，则称 x 是单位元

定义质数，若 x 不是 zero 且不能分解成两个非单位元的乘积，则称 x 是质数

给定一个三元组，问他是不是质数

14.4.2 数据范围

$|a|,|b|\le 10^7$

14.4.3 关键词

数学、miller-rabin

14.4.4 题解

令 w 为 w^2=w-3 的根，w=$\frac{1+\sqrt{-11}}{2}$。

然后每个三元组(a,b,c)可以映射到域 Z[w]，$\mathrm{change}(a,b,c)=(33-2*a-c)+(b-a)*w$。

于是就是查看$(33-2*a-c)+(b-a)*w$是不是质数。

然后设$N(a+bw)=a^2+ab+3b^2$.设 x=a+bw.

然后有 x|Nx;

如果 x|y,Nx|Ny;

N(xy)=Nx*Ny;

Nx=1 当且仅当 x 为单位元；

即由上述推出 x 是质数，当 Nx 为质数。

而如果当 x 是质数，Nx 为质数或质数的平方，且 x=$\sqrt{p}$或$-\sqrt{p}$。

于是可以得出结论：

1.当 x 是整数，那么 x 是质数，当且仅当 x 是质数且要么 abs(x)=2 要么 abs(x)≠11 且-11 在模 x 下没有二次剩余；

2.当 x 不是整数，那么 x 是质数当且仅当 Nx 是质数，这个可以用 miller-rabin 检验。

14.4.5 时空复杂度

空间复杂度 O(1),时间复杂度 O(logN).

14.5 Different Trips

14.5.1 题目大意

在一棵 N 点树上，每个节点的权值为其度数，然后问有多少段不同的从某一节点到其祖先的链。相同的链指的是权值按顺序一一相同。

14.5.2 数据范围

$N\le 10^5$.

14.5.3 关键词

后缀数组、hash

14.5.4 题解

这一题，我们可以直接用快排的后缀数组来解决。

首先，用一下倍增，求出每个节点往上走2^j步的串的 hash 值。

然后，直接对于所有点到根的串的排序，可以直接用快排，并通过倍增+hash 来比较大小。

排完序之后，同样只需用倍增+hash 求 lcp。

求完 lcp 之后，就可以直接统计得到答案了。

14.5.5 时空复杂度

空间复杂度 $O(N\log N)$,时间复杂度 $O(N\log^2 N)$.

14.6 Quasi-Polynomial Sum

14.6.1 题目大意

给定一个 D 次多项式 P(x)的前(D+1)项 P(0) mod M….P(D) mod M 分别为 A0….AD，非负整数 Q，正整数 M,N，求$(P(0)*Q^0+\cdots+P(N-1)*Q^{N-1}) \bmod M$.

14.6.2 数据范围

$M<10^{18}, N\le 10^{100000}, D\le 20000$, M 不能被 2 到 $D+14$ 中任何一个数整除

14.6.3 关键词

分类讨论、数学、拉格朗日插值法

14.6.4 题解

G(n)为一个 D+1 次多项式。

由 $G(n)=Q^n*F(n)-F(0), G(n+1)-G(n)=Q^{n+1}*F(n+1)-Q^n*F(n)$,

得到 $F(n+1)=\dfrac{F(n)+P(n)}{Q}$.

然后又因为 F(n)是一个次数为 D 的多项式，因此满足

然后通过这方程，并把 F(i)通过递推式表示成 F(0)的一次函数，然后解出 F(0),然后解出 F(1)到 F(D+1),之后再用拉格朗日插值法求出 F(n)即可。

然而，我们仍需要考虑没有 Q 或 Q-1 的逆元的时候的处理方法。

这时，我们先把 Q=0 与 Q=1 的特殊情况处理掉。

然后考虑把 $M=m1*m2*m3, \gcd(m2*m3,Q)=1, m1|Q^u, (m1*m3.,Q-1)=1, m2|(Q-1)^v$,u 与 v 为满足条件最小自然数。

考虑解出 mod 这三个值的答案后，用中国剩余定理合并即可。

考虑 m1，由于 $m1|Q^u$,因此当 i>=u 时，$P(i)*Q^i=0\ (\bmod Q^u)$,,因此只计算前面的项即可。

而 m2，则是因为题目保证 M 与 2，3，….，d+14 互质，所以 M 最小质因子至少 17.
从而 v<=14,由$17^{14} > 10^{18}$,所以 M 与 2,3,….d+v 互质。

$$\mathrm{P(i)} * \mathrm{Q^i} = \mathrm{P(i)} * \left((\mathrm{Q}-1)+1\right)^{\mathrm{i}} = \mathrm{P(i)} \sum_{j=0}^{i} C(i,j) * (Q-1)^j$$

又有 j>=v 时，,从而可以忽略较大项。
又由上式得知次数不超过 d+v。
因此，我们只需要暴力算出第 0 到 d+v 项，然后用拉格朗日插值法求第 N 项即可。
而 m3，由于都存在逆元，直接按照正常方法即可。

14.6.5 时空复杂度

空间复杂度 O(D),时间复杂度 O(D+logN).

试题泛做(15)

15.1 Chef and Churu

15.1.1 题目大意

有一个含 N 个数字的数组 A，元素标号 1 到 N，同时有 N 个函数，也标号 1 到 N。第 i 个函数会返回数组中标号 L[i]和 R[i]之间的元素的和。

进行 M 次操作：

1 x y 将数组的第 x 个元素修改为 y。

2 m n 询问标号在 m 和 n 之间的函数的值的和。

15.1.2 数据范围

$N, M \le 10^5$.

15.1.3 关键词

分块

15.1.4 题解

这一题我们可以采用分块的方法。

对函数进行分块，然后记录每一个块的函数里面各个元素的出现次数。

然后对于修改，就直接根据每一块的那个元素的出现次数更新这个块的函数总和。

而询问，则是直接统计整块的答案。

而至于不是整块的部分，我们则可以用逐个函数的加进去。通过用树状数组去维护 A 数组的值即可。

因此，修改操作也需要修改树状数组的对应节点。

15.1.5 时空复杂度

空间复杂度 O(Nsqrt(N)),时间复杂度 O(Nsqrt(N)logN).

15.2 Sereja and Order

15.2.1 题目大意

有 N 个程序，每个程序都需要在两台电脑上分别运行。第 i 个程序需要在第一台电脑上运行 A[i]秒，在第二台电脑上运行 B[i]秒。一台电脑不能同时运行两个程序，一个程序也不能同时在两台电脑上运行。求最少运行时间与方案。

15.2.2 数据范围

$N \le 2 * 10^5$.

15.2.3 关键词

贪心、随机

15.2.4 题解

这题，我们需要去发现一些结论。

通过画数据以及分析，我们可以猜想一点：最少运行时间就是 sumA 与 sumB 的最大值，除了一种情况：其中一个 A[i]+B[i]>sumA,sumB 的情况。

而实际上这个也是正确的。

因此，我们可以先特判掉特殊情况，这种情况的方案比较容易构造。

之后，对于普通情况，我们已经找到答案了。那么问题就是怎么构造方案了。

假设 sumA>sumB.

那么显然当 A,B 都空时优先选 A。

然后，这时我们可以直接随机相当数量的排列顺序，然后检验是否满足即可。

因为可行方案比较多，因此不用多少次就能找到答案了。

15.2.5 时空复杂度

空间复杂度 O(N)，时间复杂度 O(C*N).

15.3 Attack of the Clones

15.3.1 题目大意

我们称一个形为 f:A→B 的函数叫做布尔函数，其中 A 是所有长度为 n 且仅由 0 和 1 组成的数列的集合，B={0,1}，我们称 n 为布尔函数的项数。

我们称满足一些条件的布尔函数构成的集合称为 clone。

现在有四个特殊的 clone 如下：

Z 是 0-保留值函数集合: 满足 f(0, ..., 0) = 0;

P 是 1-保留值函数集合: 满足 f(1, ..., 1) = 1;

D 是自对偶函数集合: 满足!f(x1, ..., xn) = f(!x1, ..., !xn);

A 是仿射函数集合: 满足 如果 f(a1, ..., c, ..., an) = f(a1, ..., d, ..., an) 则 f(b1, ..., c, ..., bn) = f(b1, ..., d, ..., bn)的函数， 在这里 c 和 d 都在某个位置 i，并且这个对于任意 i, a1, ..., an, b1, ... bn, c , d 都应成立.

现在我们有兴趣知道在上述几种集合的组合中有多少个 n 项函数。

15.3.2 数据范围

$N, q \le 100$.

15.3.3 关键词

容斥原理

15.3.4 题解

这一题，共有 16 种不同类型的函数。即属不属于 Z、P、D、A 的情况。

那么，假设我们知道这 16 种情况的答案，现在要处理询问。

显然我们可以通过记录一个2^{16}的数，表示哪些类型的数可以算进答案，然后直接根据表达式来处理，最终就可以算出可以算进答案的类型，然后直接统计答案即可。

因此，我们的问题就是求这 16 种情况的答案。

而为了方便起见，我们可以求至少属于某些集合的方案，最后只需要用一个容斥原理的方法就能求回原本所要的答案了。

而对于这 16 种可能情况，我也没有什么好的方法，只好一个一个弄了。

0000：所有函数都可以，2^{2^n}。

1000：满足 Z，也就是限定了其中一个映射关系，所以是$2^{(2^n)-1}$.

然后剩余有一系列情况也这个相似，因此直接讨论比较特别的一种情况。

0001：满足 A 的答案。这个实际上就是找出一个.即求出一个 c 数组，使得恰好对于各种 ai 取值可以对应上。因此，c 的取值不同，方案也就不同，因此不同方案是2^{n+1}。

其余情况就可相应得出了。

15.3.5 时空复杂度

空间复杂度 O(1),时间复杂度 O(logN+len).

15.4 Minesweeper Reversed

15.4.1 题目大意

你将会处理一个扫雷的逆过程。开始时所有的方块都是打开的，有的含“雷”，剩下的不含。 玩家的目标是尽快关掉所有的方块。有两种关掉方块的方式：你可以通过一次点击来关闭一个方块（可以关闭有“雷”的方块），或者点击其他的方块顺便让这个方块关闭。

点击一个方块 C1 以后，按照正常扫雷规则，所有原本可能和 C1 同时打开的方块，C2，会全部关闭。

问最少点击次数。

15.4.2 数据范围

$R, C \le 50.$

15.4.3 关键词

一般图最大匹配

15.4.4 题解

这一题，首先对于所有的雷，必须各花费一次。

然后，对于所有周围一个 0 都没有的空格，必须各花费一次。

而至于周围有 0 的，显然点 0 要优于点非 0 的。

而对于一个点 0 的情况，我们可以发现：最多只会使得原本的两个连通块一起关闭。

因此，我们可以直接把连通块抽象成一个点，然后如果两个连通块可以通过某一个 0 一起关闭就连一条边，然后对这个一般图求最大匹配，求出来之后就可以得到最少的点击次数了。

15.4.5 时空复杂度

空间复杂度 $O((RC)^2)$, 时间复杂度 $O((RC)^3)$. 实际上远远达不到这个复杂度。

15.5 Random Number Generator

15.5.1 题目大意

由 $Ai = (Ai-1 \times C1 + Ai-2 \times C2 + ... + Ai-k \times Ck) \bmod 104857601$，给定 A1 到 Ak，C1 到 Ck，求 A(n).

15.5.2 数据范围

$K \le 3 * 10^4, N \le 10^{18}.$

15.5.3 关键词

fft、特征多项式

15.5.4 题解

这一题如果直接用矩阵乘法的话，复杂度为$O(k^3 * \log N)$，显然会超时。

因此，我们可以求矩阵的特征多项式，$f(x)=x^k - \sum_{i=1}^{k} C_k * x^{k-1}$.

然后令 $g(x)=x^n \ mod\ f(x)$

那么答案就是$\sum_{i=1}^{k} [x^{i-1}] g(x) * C_i$

而至于求解 g(x),则可以参照 picks 博客的多项式求模的文章。

大概思路就是先通过倍增的方法求出多项式的逆元,然后再先通过多项式的系数反转等变化使得多项式除法中的余数没了， 然后再通过多项式逆元求得多项式除法的商，再进行一次 fft 得到多项式除法的余数即可。

15.5.5 时空复杂度

空间复杂度 O(k),时间复杂度 O(klogKlogN).

15.6 Counting on a tree

15.6.1 题目大意

给定一棵 N 个节点的带边权树，计算有多少无序数对(S,T)，满足连接 S,T 两点的路径上，所有边权的最大公约数等于 1。

但有 Q 次修改操作，表示把一条边的权值更改。

每次修改完询问答案。

15.6.2 数据范围

$N \le 10^5, Q \le 100,$ 边权 $D \le 10^6$.

15.6.3 关键词

离线处理、莫比乌斯函数

15.6.4 题解

这题，我们首先可以通过 mobius 函数将问题转化为询问只连接是 p 的倍数的边时连通的点对数。

对于这一个问题，我们可以通过离线进行整体考虑。

首先，我们先把永远不会修改的边归为一类，然后直接对含有因子 p 的边直接用并查集维护连通了。

再对会修改的边进行处理，这个我们可以逐个询问逐个询问的处理，处理一个询问的时候，就把所有曾经含有过因子 p 的会修改的边查看在当前询问的时候是不是 p 的倍数，如果是，则连通，统计答案。

然后，当统计完答案之后，再把处理当前询问时加入的边删除即可。

而为了维护这个删除，我们可以用只进行启发式合并的并查集，这个的删除操作非常的方便。

具体要注意一下写法，免得一不小心就写超时了。

15.6.5 时空复杂度

空间复杂度 $\mathrm{O}\left(\text{总因子数} + \mathrm{Q}^2 + \mathrm{N}\right)$,时间复杂度 $\mathrm{O}\left(\mathrm{Nsqrt(D)} + \text{总因子数} * \mathrm{logN}\right)$.

试题泛做(16)

16.1 Count on a treap

16.1.1 题目大意

维护一个大根堆 Treap，要求支持：
0 k w：插入一个关键字为 k，权值为 w 的点
1 k：删除一个关键字为 k 的点。
2 ku kv：返回关键字分别为 ku 和 kv 两个节点的距离
M 次操作。

16.1.2 数据范围

$M \le 2 * 10^5$.

16.1.3 关键词

线段树

16.1.4 题解

这题，我们考虑树上两个节点的距离怎么求。
可以通过 dep[a]+dep[b]-2*dep[lca]来解决。
那么我们问题就转换为求 dep 还有求 lca。
我们按照 key 值大小排序。
然后我们可以发现 lca 实际上就是两点所在的段中间权值最大的节点。因此，lca 的问题就解决了。
接下来就是求 dep。
这一个，如果考虑不断找父亲上去的话，就会陷入误区。
实际上 dep 就是往上从左儿子到父亲的次数加上往上从右儿子到父亲的次数。
而我们考虑从左儿子到父亲的次数，实际上就是往左的一个按权值且确定起点的最长上升子序列。右儿子那边也是同样的。
因此，我们问题就是维护每个点往左（及右）的最长上升子序列。
这一个，我们实际可以通过线段树来解决。而至于本题的插入删除操作，我们可以事先把所有操作读入得到一个相对顺序的数组。从而使问题变成静态的。
考虑插入删除操作，实际上都只不过是要解决知道左区间以及右区间相关信息怎么快速得到答案。
我们可以通过记录 ans，max，min 来合并。Ans 为从区间最左端开始的最长上升子序列(从最右端类似)

如果左区间 D 的 max 大于右区间 E 的 max 显然可以结束。

否则，就右区间往下走得到新的两个节点 B,C，如果 B 节点的 max>D 的 max，那么在 E 的最长序列中 B 的 max 右面的所有部分可以直接继承，问题就转换为求 B 节点中的在最长序列的部分了；否则即 B 节点的 max<D 的 max，那么 B 可以忽略，直接考虑 C 即可。

这样逐层往下，每层只访问一个节点时就能得到答案了。

16.1.5 时空复杂度

空间复杂度 $O(N)$,时间复杂度 $O(N\log^2 N)$.

16.2 Graph Challenge

16.2.1 题目大意

给定一幅 N 个节点 M 条边的图，用下面伪代码遍历：

```
int C = 1;
void DFS(int u) {
new_number[u] = C;
C++;
// initially all value of visited array is set to false
visited[u] = true;
// here v can be chosen in an arbitrary order
for each v such that there is a edge from u to v
if(visited[v] == false)
DFS(v);
}
```

然后，给一些定义。一个节点 x 被称为是另一个节点 y 的 supreme vertex，如果存在一条有向路 x = v_0, v_1, ..., v_k = y，满足 x < y < w，对于所有 0 < i < k。也就是说，一条从 x 出发到 y 有 0 个或更多个中间节点的有向路，满足所有中间节点的编号都大于 x 和 y 的编号，并且 x 的编号小于 y。如果 v 是另一个节点 w 所有的 supreme vertex 中编号最小的，v 被称为是 w 的 superior vertex。你将得到 Q 个询问。每个询问，将会给定一个节点 v，你需要找出有多少节点，将 v 视为其 superior vertex。

16.2.2 数据范围

$N, Q \le 10^5, M \le 2 * 10^5$.

16.2.3 关键词

半必经点

16.2.4 题解

这题首先询问我们可以忽略，我们只需要求出每个点有多少个点视其为 superior vertex 即可。

而询问一个点有多少个点视其为 superior vertex，实际上就是询问最近半必经点是它的点数。

而对于最近半必经点的内容，可以参照李煜东老师于 wc2014 的课件。

16.2.5 时空复杂度

空间复杂度 O(N+M),时间复杂度 O(N+M+Q).

16.3 Substrings on a tree

16.3.1 题目大意

考虑一棵有 N 个节点的树，以 1 号点为根，对于每一个节点都有一个小写字母。由此我们得到一个"树串"。

一个字符串是该"树串"的子串，当且仅当这个串可以被获得通过树上一个节点到他一个子孙路径上所构成的字符串,当然空串也输于子串。

希望求出第 K 小的不重复子串。Q 次询问。

16.3.2 数据范围

$N \le 2.5 * 10^5, Q \le 5 * 10^4$.

16.3.3 关键词

广义后缀自动机

16.3.4 题解

这一题，我们可以通过对原树构造一个广义后缀自动机。

然后，通过拓扑序上的 DP 求出每一个节点往后有多少个不同子串。

然后，就可以在后缀自动机上进行统计，逐一统计以每个字母开头的数目从而得到第一位字母是什么，然后就处理第二位，以此不断处理下去，就可以得到要输出的串了。

至于广义后缀自动机的相关内容及习题，可以参考 ZJOI2015 诸神眷顾的幻想乡一题的题解。

16.3.5 时空复杂度

空间复杂度 O(N),时间复杂度 O(26*输出总长度+N)。

16.4 Find a special connected block

16.4.1 题目大意

给出一个 N*M 的矩阵，每个格子上填着从-1 到 N*M 中的一个数。

你的任务是找到一个联通块(只能通过上下左右四个方向连接)，在这个联通块里，至少要包含 K 个不同的正数，且不能有-1。

给出选取每一个格子所需要的代价，我们要求你选出的连通块所包含的格子的代价和最小。

16.4.2 数据范围

$N, M \leq 15, K \leq 7.$

16.4.3 关键词

斯坦纳树

16.4.4 题解

这一题其实是一题典型的斯坦纳树。

考虑如果所有格子上的数的范围都在 1-K 里面。

那么，我们直接用普通的斯坦纳树即可解决。F[i][j]表示当前颜色包含的状态为 I，当前在 j 这个点的答案。然后考虑两种情况：

1.在 j 这个点把两个颜色集合合并；

2.沿着 j 这个点去走一条路径。

对于第一种，我们只需要枚举当前颜色状态集合的子集即可。

而对于第二种情况，我们则是进行一次最短路径更新即可。

然后，就能得到答案了。

而对于原题，我们实际上通过分析可以知道实际上直接给每种颜色随机赋值一种 1-K 的颜色，在足够多次的情况下，也能得到答案。

这里我只赋值了 400 次，需要用到更改随机种子。

当然，也可以通过优化常数从而增加赋值的次数。

16.4.5 时空复杂度

空间复杂度 $O(2^k * n * m)$,时间复杂度 $O(C * 3^k * n * m)$,C 为操作次数。

试题泛做(17)

17.1 Little Party

17.1.1 题目大意

N 个字母，分为大小写。
然后有 M 个长度为 N 的串。
如果一个字母的集合是基子集，当且仅当剩余的字母无论取大写还是小写都出现了。
问最少用多少个基子集才能覆盖 M 个串。

17.1.2 数据范围

$1 \le \mathrm{N} \le 5, 1 \le \mathrm{M} \le 1000.$ 数据组数 $\le 120.$

17.1.3 关键词

搜索

17.1.4 题解

这一题，首先我们可以通过暴力求出所有可能作为基子集的子集，然后再除去包含其他基子集的基子集，于是就剩下不多了。然后，我们可以通过加入一些可行性剪枝以及最优性剪枝的优化进行暴搜，然后就能获得 95 分。

对于最后一个数据点，由于基子集数目达到了 30， 暴搜只能通过比较少的数据，因此我们可以采取一旦出现重复的数据则直接范围答案的方法通过。

17.1.5 时空复杂度

空间复杂度 $\mathrm{O(NM)}$，时间复杂度 $\mathrm{O}(2^{\mathrm{N}} * \mathrm{M} + 2^{2\mathrm{N}} + 2^{\mathrm{C}}), \mathrm{C}$ 为基子集个数。

17.2 Black-white Board Game

17.2.1 题目大意

初始时，他们有一个 N*N 的矩阵，矩阵的格子都是白色的。在游戏开始之前，Alex 会以如下方式给矩阵染色：他选择 N 对整数 L[i],R[i]，然后将第 i 行的第 L[i]列到第 R[i]列(从 1 开始计数)染成黑色。

玩家一轮轮进行游戏。在游戏中的每一轮，Alex 和 Fedya 都要行动一次，Alex 先行。

当一位玩家行动时，他需要给出整数 1 到 N 的一个排列 P，Alex 给出的排列中的逆序对数应为偶数(Fedya 给出的为奇数)，另外，所有(i,Pi)对应的矩阵中的格子应为黑色。每轮游戏中玩家给出的排列不能与之前给出的重复。

如果在同一轮中 Alex 和 Fedya 都无法给出满足条件的排列，游戏和局。如果 Alex 和 Fedya 都找到了满足条件的排列，游戏继续，进入下一轮。如果其中一位玩家给出了满足条件的排列而另一位没有，则给出排列的一方获胜，游戏结束。

17.2.2 数据范围

$N \le 10^5$.

17.2.3 关键词

可并堆、数学

17.2.4 题解

这一题，可以等价转化为询问偶数个逆序对的排列多还是奇数个的排列多。

这个我们可以采取相消最后剩下一种特殊排列的方式。

假设有两行范围为[a,b]与[a,c](b<=c)，如果 b=c,显然是平手，因为这两行完全可以调换；而如果 b<c，我们可以发现当第二行取[a,b]范围内的数的时候是没有意义的，因为可以进行调换抵消掉，因此我们只用选择[b+1,c]的数，从而最短的那个我们可以直接给它选取 a。然后把除最短的 a 开头的都更新为 b+1 开头，然后继续不断操作。

因此，我们可以给每一个开头的所有行用一个可并堆维护，每次取出最小值，然后把剩下的并到 b+1 的节点那里，并在过程中查看是否已经存在到达平手的状态。

如果没有，则根据维护过程中找到的特殊排列，查看这一排列的逆序对个数从而确定胜者即可。

17.2.5 时空复杂度

空间复杂度 O(N),时间复杂度 O(NlogN).

17.3 Hypertrees

17.3.1 题目大意

一个 3-超图类似与一个普通的图,只不过其中的边都连接三个点.
一个 3-超树是一个去掉任意一条边都以后都不连通的 3-超图.
给定 N,问有几种含有 N 个带标号的点的本质不同的 3-超树.

17.3.2 数据范围

N<=17

17.3.3 关键词

搜索、打表

17.3.4 题解

这题 N 并不大，因此我们可以通过考虑一下不太优的算法最后打表输出。
那么，我们需要考虑这题的解决方法。
这题我们实际上可以考虑 f[i][j]表示用了 i 条边当前连通状态为 j 的答案数。然后枚举其中一条边（这条边连接的三个点需要在同一连通块），然后，再枚举删除这条边之后这三个点分开的连通块，有可能分开成三个也有可能分开成两个，对于这些情况分别讨论处理，并通过优化适当常数，这个方法可以在不算太长的时间内跑出解。最后还要注意选边的顺序，以免顺序不同导致的重复。

17.3.5 时空复杂度

空间复杂度 O(1),时间复杂度 O(1).

17.4 Short II

17.4.1 题目大意

给定 p(一个质数)，问有多少对 a,b(a>p,b>p)满足 ab 被(a-p)(b-p)整除。

17.4.2 数据范围

$1 < \mathrm{p} < 10^{12}$.

17.4.3 关键词

数学

17.4.4 题解

首先通过简单变换，原题等价为 ab|p*(a+b+p).a,b 需要为正整数。
然后分类讨论。
如果 p|a 且 p|b,那么 a/p 与 b/p 只有 5 组解，(1,1)(1,2)(2,1)(2,3)(3,2);
否则，p⊥a 或 p⊥b，原题继续等价为 ab|a+b+p.
此时 p+1>=(a-1)(b-1),如果 a=b，则只有(1,1);如果 a≠b，设 a<b,则有$\mathrm{a} < \mathrm{w} = 1 + \sqrt{\mathrm{p}+1}$.
并且令 d=ak-1.
则需要找到所有合法的(a,d)：
1.p⊥a；
2.d|a+p;
3.a|d+1;
4.b>a;
那么，可以分两种情况：
1.遍历所有$\mathrm{d} \le \sqrt{p+w}$,每一个合法的 d 有两个 a 满足；
2.遍历所有$\mathrm{b} \le \mathrm{d}$ 即 $\mathrm{b} \le \sqrt{p+w}$,每一个合法的 b 只有 1 个 a 满足；

最后，还有一种情况 p⊥a 且 p⊥b，这里的每一种情况都可以由上一种的得来，因此是上一种的方案数的 2 倍（因为上一种我们计算时直接约去了一个 p）。

17.4.5 时空复杂度

空间复杂度 O(1),时间复杂度 O(sqrt(P)).

试题泛做(18)

18.1 Across the River

18.1.1 题目大意

有一条宽度为 W 的河，所有在 xy-坐标系中符合 0 <= y <= W 的点都属于这条河流。

河面上有 N 个木桩，还有 M 种可以用的木头圆盘，第 k 个木桩的坐标为 (Xk,Yk)。第 k 中圆盘半径为 Rk,每一块的价格为 Ck.

Ciel 可以买任意多的圆盘，每一个圆盘的中心都必须为某一个木桩的位置。

只能在 直线 y = 0 或 直线 y = W 或圆盘上移动（相切的两者之间可以移动)，问从 直线 y = 0 到直线 y = W 修建一座可以走过去的桥最少的花费。

18.1.2 数据范围

$\mathrm{N, M} \le 250, \mathrm{Ck} \le 10^6$, 其余所有坐标 $\le 10^9$.

18.1.3 关键词

最短路径

18.1.4 题解

这一题，我们可以使用最短路径。

考虑一种比较暴力的方法。

直接每一个点拆成 M 个点，(I,J)表示在 I 点放第 J 个圆环且从 S 出发的最短路径。

那么，我们直接连接所有相交的(I1，j1)与(i2,j2)点即可。

但是，这种方法显然会超时。

然后，我们很容易分析得到，如果对于两个圆盘(r1,c1)与(r2,c2),如果 r1>=r2 且 c1<=c2,那么(r2,c2)这个圆盘是没有用的。

因此，我们可以只维护有用的圆盘，从而得到一个圆盘半径递增、费用递增的圆盘序列。

从而我们可以对边进行优化，连接两种边:

1.(I,j)->(I,j+1),表示圆盘增大，权值为费用差；

2.(I,j)->(k,l),i，j,k 任意，而 l 则是使得半径尽可能小并且也相交的圆环编号，权值为 l 号圆环的费用。

这样处理之后，边数就变成了三次方级别的，然后用 dijkstra 算法就可以通过此题了。

18.1.5 时空复杂度

空间复杂度 O(N*N*M),时间复杂度 O(N*N*M*log(N*M))。

Challenge 题目

1 Factorisation

1.1 题目大意

给定一个整数 N。

你的任务是给出 M 个整数 A1...AM,满足 A1 * A2 * ... * AM = N。

你需要最大化数列的项数 M。

1.2 数据范围

对于每个测试点，你的代码的评分是你的代码对每组数据给出的 M 的平方和，你的实际得分占总分的比值(可能高于 1)为：

你的代码的评分/September Challenge 期间最高分代码的评分

如果你的某个方案不合法，这个测试点会直接判为 0 分。

T = 100。

其中 10 组，有 2<=N<=10^18:N 随机均匀分布。

另 15 组，有 2<=N<=10^18: N 不是随机均匀分布（对于 Tsinsen 上的数据，N 为 3 个质数的积）

另 50 组，有 2<=N<=10^1000: N 随机均匀分布。

另 25 组，有 2<=N<=10^1000:所有 N 的素因子不超过 10^18（据我猜测，原数据里 N 是由若干不超过 10^18 的随机数相乘得来，因而这里也采用了这种生成方式。对于这类数据，如果生成数据采用若干 10^18 级别的素数相乘，似乎很难比直接输出原值得到更多的分>_<）

1.3 关键词

搜索

1.4 题解

这一题，我们可以先观察评分方式，M 的平方和。

这种评分方式就决定了前面的 25 组数据影响不会太大，因为后面 75 组的 M 必然会很大，平方之后影响就更大了。

因此，对于前面 25 组，我们也没有太大必要用太好的算法了，直接找一些小质数去筛选就算了。

而对于后面的 75 组，我们同样可以先用一些小质数去筛选，但不超过 10^18 的随机数相乘的话，必然还会有不少的大质数，因此我们必然还要枚举一些相对大一点的质数，而对于 75 组都弄的话效果可能还没有对其中一部分弄。

因此，我选择了弄其中 25 组，弄些大质数出来。

然后再加上一些常数优化之后，就能跑过了。

1.5 时空复杂度

空间复杂度 O（sumM），时间复杂度 O(100*100*1000+25*1000*90000).常数优化之后速度还是比较快的。

2 Efficient Painting

2.1 题目大意

一开始全是白色的 N*N 矩形纸，需要用尽量少的染色次数使得某一些格子变成黑色。
有三种染色方法：
1.某矩阵全染白；
2.某矩阵全染黑；
3.某矩阵颜色全部翻转。

2.2 数据范围

$N \leq 50$.黑色概率为 0.4-0.6.

2.3 关键词

贪心

2.4 题解

这一题，我们可以只使用翻转操作的情况下达到比较优的结果。
那么，只使用翻转操作的话，我们还可以进行进一步分析。
S[i][j]=A[i-1][j-1]^A[i-1][j]^A[i][j-1]^A[i][j].
然后我们可以发现当所有的 S[i][j]变为 0 时就是全白（把结束状态当成全白）。
而对一个矩形进行翻转，真正变化的只有边界四个点。
因此，我们可以进行贪心，优先选择四个点都是 1 的，然后进行翻转。
之后再选择三个点都是 1 的。
然后，我们可以比较容易的证明除了全为 0 的情况以外，如果不存在四个点都是 1 的情况，必然存在三个点都是 1 的情况。这个可以通过几步的反证即可。
然后，就能得到比较优的解了。

2.5 时空复杂度

空间复杂度 $O(N^2)$,时间复杂度 $O(n^5)$.

3 Closest Points

3.1 题目大意

在三维空间中，包含 n 个响应中心和 q 个银行。你需要为每个银行找到离他欧几里得距离最近的响应中心，如果有多个响应中心，你可以任意输出其中的一个。

3.2 数据范围

N=q=50000.

3.3 关键词

K-D tree

3.4 题解

这一题，是一个典型的 K-D tree 题目，但是 K-D tree 的复杂度难以保证。因此，我们需要进行一些修改。

首先，就是一些优化，确保能跑得尽量快。首先，是 K-D tree 的最优性剪枝，一定要写得比较优。

然后，就是询问的点数要有限制，并且对其中的一小部分点进行比较完整的处理，因为这道题是以你的正确答案数量/标程的正确答案数量评分，我们要是平均的话，或许还会导致各各都比较优却没多少最优的。因此，我们可以选取前面的一小部分询问进行比较完整的 K-D tree 询问，对于后面的，就用比较少次数一点的。

这样处理之后，就可以通过此题了。

3.5 时空复杂度

空间复杂度 O(N),时间复杂度 O(NlogN+TlogN),T 为总遍历次数。

4 Deleting numbers

4.1 题目大意

假设当前有 n 个数形成的数组 a[n]，每次可以选择 v, t 两个数，满足 v + t <= n + 1，且令 k 为最大的满足 v + kt <= n 的数，必须满足 a[v] = a[v + t] = a[v + 2t] = ... = a[v + kt]。然后这些数将会删除(若 v + t = n + 1 则只删除第 v 个数)，之后形成新的由 n - k 个数形成的数组 a[n - k]，满足这个数组的元素是原来数组被删除之后剩下的元素，且相对位置保持不变。

设计一种方案删除所有的数，你的得分与你的方案的总步数有关。

4.2 数据范围

$1 \le n \le 100000, 1 \le a$ 数组中的每个元素 ≤ 100000。

4.3 关键词

暴力

4.4 题解

这一题的话，其实并没有什么特别好的解决方法，并且数据似乎比较水。。。

我们就怎么方便怎么做吧。。

我们可以从后往前删，然后每次取出最后面那个，然后看看有没有和他相同的，有的话取最后那个相同的，然后往前看看等差数列的位置上还有多少个，有多少个就删多少个。而删除的时候，直接往前退就可以，也没有太大必要用平衡树。

然后，这样一弄之后，似乎分数很高。。。$2 * 10^5$的分数。。

具体更优的方法我也不太清楚了。

4.5 时空复杂度

空间复杂度 $O(N)$，时间复杂度 $O(N^2)$,最坏情况下。

5 Fault Tolerance

5.1 题目大意

有 N 个变量，以及 M 个异或方程，要求删除尽量小的方程，使得不能解出 N 个变量的唯一解。

5.2 数据范围

$50 \le \mathrm{N} \le 200,$
$2 * \mathrm{N} \le \mathrm{M} \le 1000.$

5.3 关键词

随机化、线性关系

5.4 题解

首先，我们先考虑一个基本问题：

M 个异或方程，要怎么判断是否存在 N 个变量的唯一解。

第一种经典的方法，是直接用高斯消元解出最大无关行向量组，如果这组基的大小<N,那么就没有唯一解，否则大小等于 N，即有唯一解。

第二种方法，就是查看存不存在一列可以被其他列线性表出的，如果有，那么没有唯一解，否则就有唯一解。

这里，从第二种方法入手，比较方便。

首先，最容易想到的方法是随机选择一列，然后随机选择其他列的一个集合，然后将这一列异或选出的列，再统计 1 的个数即可，因为这里只要把 1 所在的行删掉，就可以线性表出了。关于这种方法，我是直接随机选择了 N/4 列为初始，然后再随机找其他列来异或，初始分数只有 4.88，不算优秀。

然后，我们可以由上面的方法得到一个优化之后的，因为我们对于找集合的话，实际上集合个数有很多，我们可以枚举的却很少，可能不够优。

因此，我们实际上可以更新为随机选择一列，然后再随机选择该列上的一个 1，然后就把其他在该 1 所在行上有 1 的列进行异或找最优值，不断地重复操作。这样之后，每一列就不是初始的列，而是一些列的异或之后得到的结果了。

这个的话，操作 50000 次左右可以达到 3.77.

然后，上面的方法，都需要用到压位以便于快速统计一列上的 1 的个数。

5.5 时空复杂度

空间复杂度 O（NM），时间复杂度 O(50000*N*M /15).

6 Stepping Average

6.1 题目大意

每次拿出其中任意两个数，并用他们的平均数取代他们，重复 N-1 次直到最后只剩一个数。我们叫这个剩下的数叫"迭代平均数"。

给你 N 个数。找到一个合并的顺序使得"迭代平均数"尽量接近给定的数 K。

6.2 数据范围

N=1000.

6.3 关键词

贪心

6.4 题解

这一题，我们可以直接一步一步的贪心下去。

假设当前还有 N 个数，然后从小到大排序是 A1 到 An。

然后，我们可以选取 A1 和 An，然后如果两数的平均数比 K 小得多，那么我们可以把最大的数留下来，然后就要使前面的数合成的数尽可能接近 2*K-maxx；

而如果比 K 大得多，则把最小的数留下来，更新 K 值；

而如果差不多，就直接合并这两个数，K 值不变。

这样合并下去，直到只剩下一个数。

这样就能得到一个比较优的解。

6.5 时空复杂度

空间复杂度 $O(N)$, 时间复杂度 $O(N^2 \log N)$.

7 Chef and Painting

7.1 题目大意

游戏的棋盘是一个 N 行 M 列的矩阵，行从 1 到 N 标号，列从 1 到 M 标号。

初始时，棋盘上有一些黑色的格子，其它的格子都是白色的。

每一步，大厨可以选择任意一个白色的格子和一个方向（左-右或上-下）

如果大厨选择了左-右，那么他会从选定的白色格子开始，向左边依次把格子涂成红色，直到到达了棋盘边界或者到达了某个之前已经被染成了黑色或红色的格子时停止涂色；然后再从选定的格子开始，向右边进行同样的涂色操作。

如果大厨选择了上-下，也与左-右类似，大厨会在选定的格子的上下方向上颜色。

游戏的目标是用尽可能少的步数把所有原来的白色的格子都涂成红色。

7.2 数据范围

$\mathrm{N,M} \le 100.$ 黑色格子数 $\le 3000.$

7.3 关键词

贪心、随机

7.4 题解

这道题，我们可以用两种方法然后取一个较优解。

其一，就是直接从左上开始贪心，直接看看左右优还是上下优，选择更优的就直接染色。然后再去找下一个格子。这个方法在比较多的情况下已经很优了。但是，还是无法通过全部数据。

这时，我们还可以再写一个方法。就是随机，这次不一定在左上开始贪心，而是随机从一个角、从某个方向开始贪心，然后贪心准则也是一样的，直接比较左右染色好还是上下好，选取更优的即可。

随机足够多次之后，将找到的最优解再与第一种合并即可。

7.5 时空复杂度

空间复杂度 $\mathrm{O(N^2)}$, 时间复杂度 $\mathrm{O(T*N^2)}$, T 为随机次数。

8 Sereja and Number Division 2

8.1 题目大意

首先，我们定义关于数字 A 的一个函数 f：

f(A)=sigma{A(mod B[i]) , 1<=i<=N}

现在 Sereja 想要对 A 中的数字进行重排，来最小化 f(A)的值。请你帮他找到尽可能优的解。

8.2 数据范围

$A \le 10^{1000}, N = 100.$

8.3 关键词

随机

8.4 题解

这些题也没有什么其他更好的方法了，我们可以随机来弄。

就是直接随机选取两个位，然后看看能不能更优，如果更优则直接交换，如果不更优就再随机一下看看概率是否小于某个预定的 p，如果是那也交换。而这个 p，是一个关于与增加答案相关的函数。

而为了使随机次数能多一点，我们需要加快判断更优的复杂度，我们可以通过预处理各个数位模各个 B[i]的值，从而支持 O（N）的一次交换与更新判断。

这样随机足够多次之后也能得到比较优的解了。

8.5 时空复杂度

空间复杂度 O(logA+N),时间复杂度 O(T*N),T 为随机次数。

9 Maximum Sub-rectangle in Matrix

9.1 题目大意

给出一个 H×W 的整数矩阵 A，行标号 0~H−1，列标号 0~W−1，求一个子矩阵使其中元素之和尽可能大。

这个子矩阵不要求是连续的，即求出一些行和一些列，选取这些行列相交处的元素，输出这些行列。

9.2 数据范围

$H, W \le 300.$

9.3 关键词

随机、贪心、调整

9.4 题解

这题首先我们可以用随机。

先随机各种各样可能的选取子矩阵的情况然后找出比较优的初始解。

然后就可以用一些贪心调整。

直接看看某些行某些列如果反过来（选<->不选）会不会更优，如果更优直接交换，如果不是的话，则同样是根据概率 p 来确定要不要换，p 与增加答案的值有关。然后如果还与剩余操作次数相关的话，应该会更加优，即还有比较多操作次数的时候，p 可以大一点，到后面就弄小一点，更有利于增大询问到的情况，从而取得更优的解。

然后反复这个过程足够多次数即可。

9.5 时空复杂度

空间复杂度 O(H*W),时间复杂度 O(T*H*W).

10 Similar Graphs

10.1 题目大意

Chef 只考虑一对都是 N 个点图，Chef 会给每张图的每个点以 1 到 N 的一个编号，且每个编号只会出现一次。Chef 定义两张图的相似之处为 2*COMMON/TOTAL,COMMON 表示两张图中共同出现的边数，即有多少二元组(A,B)满足两张图中，点 A 和点 B 之间都有一条边。TOTAL 表示两张图的边数总和。

Chef 发现这两张图的相似程度取决于每个点的标号。Chef 想你帮助他去找到两张图一种标号使得相似程度尽量大。

10.2 数据范围

对于单个测试点的每一组数据，N 是 30-75 之间的随机数。一个实数 D 是 0.05 到 0.5 之间的随机数。对于每一对合法的点对，存在边的概率为 D。这就生成了第一张图。

一个整数 C 是 0 到 N*(N-1)/2 之间的随机数。之后会选择 C 对不同的合法点对。对于对于一对点对，如果之前它们之间已经存在边，则会有(1-D)的概率删除这条边。如果它们之间没有边，则有 D 的概率添加一条边。之后会生成一个随机的点标号的排列，这就生成了第二张图。

10.3 关键词

随机、贪心

10.4 题解

这题我们同样是照常的方法。

首先，我们确定第一幅图标号恰好是 1-N，然后随机第二幅图的标号。

直接随机足够多次，然后每次随机选取两个标号，然后算一下交换这两个标号后的代价

变化，如果相似程度更大，直接交换，否则，则按概率 p 交换，p 与相似程度减少量有关，也可以进一步和随机的次数有关。

然而只是这样的话，无法通过这一题的数据。

因为，主要是这个的最优方案（或者说较优方案）并不多（由数据的生成方式得知），我们随机的效率可能并不高，因此，我们把每一次的随机改成 N^2 个标号的交换都枚举一下，然后交换准则和上面的一样，然后减少操作轮数，这样的结果反而会更加的优。

10.5 时空复杂度

空间复杂度 $\mathrm{O}(\mathrm{N}^2)$,时间复杂度 $\mathrm{O}(\mathrm{T} * \mathrm{N}^2)$，T 为操作轮数。