IOI2016 中国国家集训队作业
试题泛做

绍兴市第一中学

王文涛

目　录

1 非challenge型试题 **5**

1.1 Future of draughts 6
1.2 Simple Queries 7
1.3 Easy exam 8
1.4 A game on a graph 9
1.5 Chefbook 10
1.6 Chef and Balanced Strings 12
1.7 Counting on a directed graph 13
1.8 Black-white Board Game 13
1.9 Little Party 14
1.10 Counting on a Tree 15
1.11 Random Number Generator 17
1.12 Devu and Locks 18
1.13 Payton numbers 20
1.14 Ranka 21
1.15 Xor Queries 22
1.16 Divide or die 23
1.17 Course Selection 24
1.18 Chef and Churu 25
1.19 Sereja and Order 26
1.20 Children Trips 27

1.21 Union on Tree . . . 28
1.22 Rectangle Query . . . 29
1.23 Fibonacci Numbers on Tree . . . 30
1.24 Team Sigma and Fibonacci . . . 31
1.25 Push the Flow! . . . 33
1.26 Game of Numbers . . . 34
1.27 Sereja and Equality . . . 34
1.28 Two Companies . . . 36
1.29 Sereja and Arcs . . . 37
1.30 Dynamic Trees and Queries . . . 38
1.31 Sereja and Subsegment Increasings . . . 39
1.32 Chef and Graph Queries . . . 40
1.33 The Street . . . 41
1.34 Graph Challenge . . . 42
1.35 Count on a Treap . . . 43
1.36 Counting D-sets . . . 44
1.37 Counting The Important Pairs . . . 45
1.38 Query on a tree VI . . . 46
1.39 Petya and Sequence . . . 47
1.40 Gangsters of Treeland . . . 48
1.41 Queries With Points . . . 49
1.42 Fibonacci Number . . . 50
1.43 Two Roads . . . 50
1.44 To Queue or not to Queue . . . 51
1.45 Music & Lyrics . . . 52
1.46 Prime Distance On Tree . . . 53
1.47 Across the River . . . 54
1.48 Two k-Convex Polygons . . . 54
1.49 Count Special Matrices . . . 55

1.50 Queries on tree again! . . . 56
1.51 String Query . . . 57
1.52 Little Elephant and Colored Coins . . . 58
1.53 Room Corner . . . 59
1.54 Observing the Tree . . . 60
1.55 A New Door . . . 61
1.56 Cucumber Boy and Cucumber Girl . . . 61
1.57 Different Trips . . . 62
1.58 Quasi-Polynomial Sum . . . 63
1.59 Arithmetic Progressions . . . 65
1.60 Martial Arts . . . 66
1.61 Max Circumference . . . 67
1.62 Knight Moving . . . 69
1.63 Annual Parade . . . 70
1.64 A Game of Thrones . . . 71
1.65 Dynamic GCD . . . 72
1.66 Cool Numbers . . . 73
1.67 Expected Maximum Matching . . . 74
1.68 Little Elephant and Boxes . . . 74
1.69 Selling Tickets . . . 76
1.70 Substrings on a Tree . . . 77
1.71 Find a special connected block . . . 78
1.72 Evil Book . . . 79
1.73 Ciel and Earthquake . . . 80
1.74 Find a Subsequence . . . 81
1.75 Flight Distance . . . 82
1.76 Card Shuffle . . . 84
1.77 Misinterpretation 2 . . . 85
1.78 Short II . . . 86

1.79 Hypertrees . . . 88
1.80 Luckdays . . . 89
1.81 Colored Domino Tilings and Cutsontest . . . 90
1.82 The Baking Business . . . 91
1.83 Sine Partition Function . . . 92
1.84 Short . . . 93
1.85 Counting Hexagons . . . 94
1.86 Shortest Circuit Evaluation . . . 95
1.87 Something About Divisors . . . 96
1.88 Billboards . . . 98
1.89 Trial of Doom . . . 100
1.90 Attack of the Clones . . . 102
1.91 Minesweeper Reversed . . . 104

2 challenge型试题 106

2.1 Sereja and Permutation . . . 107
2.2 To challenge or not . . . 108
2.3 Deleting numbers . . . 108
2.4 Kali and Devtas . . . 110
2.5 Closest Points . . . 111
2.6 Similar Graphs . . . 111
2.7 Simultaneous Nim . . . 112
2.8 The Great Plain . . . 113
2.9 The Malaysian Flight Search . . . 114
2.10 Tom And Jerry . . . 116

Chapter 1

非challenge型试题

1.1 Future of draughts

【来源】

CodeChef AUG15 CLOWAY

【大意】

有T张有向图，第k张图的点数为N_k，边数为M_k。有Q次询问，每次询问给定L, R, K，表示在编号在$[L, R]$的图上进行一次游戏，初始任意选择每张图的初始点，上面放上一个棋子，进行不超过K次操作，每次操作对每张图的棋子可以不移，可以移动到相邻的某个点。但一次操作不能一个棋子都不移。到所有棋子都回到初始点时可以选择结束游戏。求有多少种不同的游戏方案。

【题解】

定义图A, B的强积$A \times B$是一张新图，点是二元组(u, v)，其中u为A的点，v为B 的点，且满足边$(u, u') \to (v, v')$ 包含于$A \times B$ 当且仅当A中存在$u \to v$ 或B中存在$u' \to v'$ 。记$G_{L:R} = G_L \times \ldots \times G_R$，那么本题的一个询问要求的就是$G_{L:R}$有多少条长度不超过$K$的环形路径。

记图G的邻接矩阵为G，那么G的长度恰为k的环形路径数即为$\mathrm{tr}(G^k)$。对于区间$[L, R]$ 的图，询问的答案（中途不能不动的方案数）就是$\mathrm{tr}(G^k_{L:R})$。可知

$$\mathrm{tr}((G_L + I)^k) \ldots \mathrm{tr}((G_R + I)^k) = \sum_{i=0}^{k} \binom{k}{i} \mathrm{tr}(G^i_{L:R})$$

即允许不动的方案数。二项式反演得

$$\mathrm{tr}(G^k_{L:R}) = \sum_{i=0}^{k} (-1)^{k-i} \binom{k}{i} \mathrm{tr}((G_L + I)^i) \ldots \mathrm{tr}((G_R + I)^i)$$

从这个式子中可以看出，只要求出了每个$\mathrm{tr}((G + I)^k)$，把$(-1)^{k-i}\binom{k}{i}$展开使得式子化为卷积形式，就可以用FFT计算出答案。

所以现在的问题就是求$\mathrm{tr}((G+I)^k)$。我们可以求出特征多项式$h(x)$（怎么求？可以把N个不同的值代入计算行列式之后插值恢复多项式），然后利用Cayley-Hamilton定理：

$$\begin{aligned} h(H) &= 0 \\ c_0 I + c_1 H + c_2 H^2 + \ldots + H^N &= 0 \\ c_0 H^{k-N} + c_1 H^{k-N+1} + c_2 H^{k-N+2} + \ldots + H^k &= 0 \\ \mathrm{tr}(c_0 H^{k-N} + c_1 H^{k-N+1} + c_2 H^{k-N+2} + \ldots + H^k) &= 0 \end{aligned}$$

由迹的线性性可知

$$c_0\mathrm{tr}(H^{k-N}) + c_1\mathrm{tr}(H^{k-N+1}) + c_2\mathrm{tr}(H^{k-N+2}) + \ldots + \mathrm{tr}(H^k) = 0$$

这样我们就得到了$\mathrm{tr}(H^{k-N})\ldots\mathrm{tr}(H^{k-1})$递推到$\mathrm{tr}(H^k)$的方法。只需要用$O(N^4)$ 暴力矩乘预处理$\mathrm{tr}(I)\ldots\mathrm{tr}(H^N)$就可以$O(KN)$递推下去了。

时间复杂度：$O(TN^4+TKN+T^2K\log K+Q)$，空间复杂度：$O(TN^2+N^3+T^2K)$（可以用离线来省空间，$O(T^2K)\to O(K)$）

1.2 Simple Queries

【来源】

CodeChef AUG15 DISTNUM

【大意】

维护一个整数序列s，支持下列操作：

- 1 l r 设S为$s[l:r]$组成的不可重集合，求$\sum_{1\le i<j<k\le|S|} S_iS_jS_k \mod 10^9+7$；

- 2 x y 把s_x赋值为y；
- 3 x 删除s_x；
- 4 z y 在s_z后插入y；
- 5 l r 求第l到r个元素中有多少不同的值。

$1 \leq N, Q \leq 10^5$

【题解】

数据结构题。先用平衡树离线处理出每个元素的静态序列。对于操作5，记$last_i$为位置i前面最近的与其值相同的位置，建立二维线段树，外层是序列，内层是$last$值。这样只需询问$[l, r]$中$last_i < l$的个数。对于操作1，可以用容斥计算：

$$6\sum_{i<j<k} S_iS_jS_k = \left(\sum_i S_i\right)^3 - 3\left(\sum_i S_i^2\right)\left(\sum_i S_i\right) + 2\left(\sum_i S_i^3\right)$$

对于其他操作，只需修改相应的$last$值，并在二维线段树中进行插入/删除操作即可。

为了卡常数，我们把$last$套到外层，这样外层就可以用树状数组。

时间复杂度：$O(N \log^2 N)$，空间复杂度$O(N \log^2 N)$（把内层改成平衡树可以做到$O(N \log N)$ （N, Q 同级）。

1.3 Easy exam

【来源】

CodeChef JULY15 EASYEX

【大意】

有一个均匀的K面骰子，上面写着1 K。你会掷骰子N次。令a_i为数i出现的次数。求$a_1^F a_2^F \dots a_L^F$的数学期望。

$N, K \leq 10^9$

$F \leq 1000$

$L \times F \leq 50000$

【题解】

设$x_{i,j}$在第j次的结果为i时为1，否则为0。则$a_i^F = (\sum_j x_{i,j})^F$。把上式展开（保留变量顺序，不合并同类项），利用期望的线性性单独处理其中每一项对答案的贡献。为了避免重复计算概率，把每一项最小表示。那么恰有i个不同变量的最小表示法有$S2(F,i)$（第二类斯特林数）种。由于不同数出现的位置不可能相同，所以只需保证不同数之间以及最小表示后不同变量之间位置不同即可。故答案为

$$\sum_{P=L}^{LF} \frac{1}{K^P} \binom{N}{P} P! \left(\sum_{i=0}^{F} S2(F,i) x^i \right)^L [P]$$

其中$[P]$表示该多项式在x^P前的系数。

斯特林数可以$O(F^2)$预处理。

多项式可以用FFT快速计算。

时间复杂度：$O(F^2 + T \times NL \log(NL) \log(L))$，空间复杂度：$O(F^2 + NL)$。

1.4 A game on a graph

【来源】

CodeChef JULY15 HAMILG

【大意】

有两个人A和B在一张无向图上玩游戏。初始时选定一个点，并在上面放一个棋子。两人轮流把棋子移动到一个相邻的未访问过的点（初始点也算访问过了）。不可操作者输。B先手。问有多少初始点使得A获胜。$T \leq 100, N \leq 2000, \sum M \leq 10^6$

【题解】

这类问题往往可以转化为匹配问题。我们有定理：如果这张图存在一个最大匹配使得点u不被匹配，则点u为获胜点。证明：由于点u不被匹配，点u出发的边必为未匹配边，A会选择其中一条。当A操作后，B 只需沿图中该点对应的匹配点移动即可。A又选择一条边，但不会是匹配边，因为那是访问过的。从而B又能沿图中匹配边移动。这样不停操作。中途B必能沿匹配边移动，因为如果不能移动，那么棋子的路径为一条长度为奇数的增广路，增广后可以使匹配数+1，与当前匹配为最大匹配矛盾。因此，B必胜。反之，A可以采用相似的策略对付B，A必胜。

故此题只需求出最大匹配后尝试增广每个未匹配点，得到的所有可能的未匹配点都是必胜点。这样就只需处理一般无向图最大匹配问题。可以用带花树算法解决。带花树算法详细内容这里不多做展开。但是我发现AC 的带花树算法大多为$O(N^3)$，根据数据范围，理论上不能通过。我也造了卡$O(N^3)$的数据。不过数据需要依赖算法处理点和边的顺序，把点的编号random_shuffle一下就不会被卡。但是从理论复杂度上讲应该写$O(NM)$的算法。

时间复杂度：$O(NM)$ ，空间复杂度：$O(N+M)$

1.5 Chefbook

【来源】

CodeChef JUNE15 CHEFBOOK

【大意】

给定一张N个点的有向图，图中边(x,y)有权值L_{xy}。可以进行若干次操作，每次操作可以选择一个点x并把其所有出边的L_{xy}增加某个值P_x 或者选择一个点y并把其所有入边的L_{xy} 减少某个值Q_y。对于x 的增加操作和对于y的减少操作都只能实施一次。最终对于每个L'_{xy} 需满足在区间$[S_{xy},T_{xy}]$内。求可能的最大L'值总和以及这时的P 和Q。

数据范围：

$N \le 100$

$M \le N^2$

$-600 \le L_{xy} \le 600$

$-1000 \le S_{xy}, T_{xy} \le 1000$

$0 \le P_x, Q_y \le 10^6$

【题解】

首先把答案中所有初始的L提出。那么只需最大化$\sum OutDegree_x \times P_x + \sum -InDegree_y \times Q_y$。对于每条边，限制就是$S_{xy} \le L_{xy} + P_x - Q_y \le T_{xy}$，可以拆成两条限制：$P_x - Q_y \le T_{xy} - L_{xy}$，$Q_y - P_x \le -S_{xy} + L_{xy}$。把$P,Q$看成变量，这个问题就是一个线性规划问题。设行向量

$$C = [InDegree_1, \ldots, InDegree_N, -OutDegree_1, \ldots, -OutDegree_N]$$

这个问题用矩阵表示为$A \times x \le B$，最大化$C \times x$。转对偶问题后就变成$A^T \times y \ge C$，最小化$B \times y$。由于A^T的每一列都恰好含一个+1一个−1，如果把对偶之后的不等式加起来的话，左边和右边都是0（右边等于0是因为每条边入度出度加起来是0），因此所有的≥都是=。这个问题可以转化为费用流。我们可以对每一个等式建一个点，对每一列连一条边，把等式当成流量平衡的等式。最后输出方案可以用SPFA来解决。

时间复杂度$O(costflow(N,M))$，空间复杂度$O(N+M)$。

1.6 Chef and Balanced Strings

【来源】

CodeChef MAY15 CBAL

【大意】

定义一个字符串是平衡的当且仅当该串的字符可以被分为两组相同的可重集合。给定一个由字母组成的字符串S，有Q 组在线询问，每组询问给定$L, R, type$，求

$$\sum_{L\le s<e\le R \text{ and } S_{s:e} \text{ is balanced}} |S_{s:e}|^{type}$$

其中$type \in \{0, 1, 2\}$

【题解】

首先，平衡的串中每种字符都出现偶数次。那么，可以对每种字符做前缀和，记录前缀$[1, i]$中该种字符出现次数的奇偶性，并用26位二进制压一下，记为P_i。可知平衡的串的s, e满足$P_{s-1} = P_e$。所以只需考虑相同的P。

可以使用分块。记块大小为S。把串分成很多块，用$O(\frac{N}{S} \times N)$的时间预处理出整块的区间内的答案和整块分界线处每种P值出现位置的0,1,2次方的前缀和。每次询问时设$l = S\lceil\frac{L}{S}\rceil, r = S\lfloor\frac{R}{S}\rfloor$，只需要在$[l, r)$的答案基础上向右向左扩展，记录每种$P$值当前扩展的0,1,2次方和（记为$Sum$数组），扩展一个元素时更新答案只需要利用原来预处理出的前缀和之差和当前记录的Sum即可。

时间复杂度：$O(\frac{N^2}{S} + QS)$，空间复杂度$O(\frac{N^2}{S})$，易知S取$\sqrt{\frac{N^2}{Q}}$时有最优复杂度$O(N\sqrt{N})$（N, Q同阶）。

1.7 Counting on a directed graph

【来源】

CodeChef MAY15 GRAPHCNT

【大意】

给定一张N个点M条边的有向图，计算有多少无序对(X,Y)使得从1号点出发有两条除了1号点没有其他交点的路径分别到达X和Y。

【题解】

如果有一组无序对不满足上述条件，那么在1号点到它们的路径上存在一个必须经过的点，也即有向图必经点。因此，此题只需构出根为1的有向图的必经点树（Dominator Tree），统计根的不同子树之间的无序对个数即可。Dominator Tree的Lengauer-Tarjan算法具体这里不多做展开。

时间复杂度：$O((N+M)\alpha(N))$，空间复杂度：$O(N+M)$

1.8 Black-white Board Game

【来源】

CodeChef APRIL15 BWGAME

【大意】

有一个$N \times N$的矩阵，第i行从L_i到R_i的格子是黑色的，其余的都是白色。现在有两个人玩游戏，每一轮先手行动一次、后手行动一次。每次行动需要给出一个

之前没有人给出过的排列P，使得所有在(i, P_i)位置上的格子都是黑色的。对于先手，还要求他给出的排列的逆序对数是偶数；同样，后手应为奇数。如果在同一轮中两方都不能行动，游戏和局。如果两方都能行动，则进入下一轮。如果只有一方能行动，则那一方胜。求两者都很聪明的情况下的游戏结局。

$N \leq 10^5$

【题解】

其实游戏就是比这样的排列里逆序对数奇偶哪个多。

由方阵和排列、逆序对这些字眼我们联想到了行列式。方阵A的行列式的一个定义就是对于所有排列，每种排列对应A_{i,P_i}的乘积再乘上(-1)的逆序对数次方，的和。而这恰好符合这题描述：把黑色看成1，白色看成0，题目就是要求矩阵的行列式的正负性。这样，这个问题就是一个求这样的特殊矩阵的行列式的问题。

我们一般求行列式的做法是高斯消元：把矩阵通过某一行减去另一行若干倍的方式消成上三角形形式，再把对角线上的元素乘起来即可。那么我们就是要模拟这个过程。从1到N枚举k，考虑所有$L_i = k$的行（如果没有这样的行，则直接返回0），用这些行里R_i最小的那行去消其他的行。这样消出来的行必然还是由连续的一段1组成的，并且它们的L 值变成了这个最小的$R_i + 1$。可以用可并堆来做这些操作：对于每个L 的位置开一个堆，按R值维护每一行。每次把某个位置的堆顶pop出来，再把这个堆合并到相应的位置去。最后只要再把剩下的凌乱的行调整成上三角形式，算一下逆序对奇偶性即可。

时间复杂度：$O(N \log N)$，空间复杂度：$O(N)$

1.9 Little Party

【来源】

CodeChef APRIL15 LPARTY

【大意】

有M次party，每次都有同样的N个人参加。每次party都不欢而散。已知每次party时每个人的心情状态（高兴/不高兴）。如果对于一部分人以及他们的心情状态的组合，其他所有的人不管是怎样的情绪，party 都会不欢而散的话，那么这个组合被认为可能是造成party不欢而散的原因。你需要选出一些这样的组合，使得可以解释所有party都不欢而散的情况，并且还要求这些组合的总人数最少。求最少总人数。

$T \leq 120, M \leq 1000, N \leq 5$

【题解】

首先那个$M \leq 1000$是唬人的，因为所有可能的不同party也就$2^N \leq 32$种。

我们可以用DFS搜索。预处理出所有可能在最优解中的组合。这些组合必须满足这个组合以外的其他人的所有可能的状态都出现在了这些party中，并且没有其他这样的组合是它的子集。DFS搜索枚举这些组合选与不选。我们可以用一个unsigned存储M个party的覆盖情况。为了避免重复搜索，可以使用hash等手段记忆化。还可以加上可行性剪枝：如果不选当前组合会导致某个party再也不能被覆盖到，那么直接放弃。如果选了当前组合与不选效果一样，那么也没有必要选。

时间复杂度：$O(2^{3^N})$，空间复杂度：$O(3^{2N} + 2^N)$

1.10 Counting on a Tree

【来源】

CodeChef MARCH15 TREECNT2

【大意】

给定一棵N个点的无根树，每条边有一个边权。有Q次操作，每次把某一条边的边权改为某个值。在所有操作之前及每次操作后输出有多少无序对(S,T)使得从点S到T的路径上的边权的gcd为1.

$N \leq 10^5$,权值$\leq 10^6$（为了方便，设为M），$Q \leq 100$

【题解】

先考虑没有修改操作的情况。对于gcd恰好为1的路径条数很难直接统计，考虑放宽条件，使用莫比乌斯反演（或者也可以理解为容斥原理）。形式化地：$f(n)$表示gcd恰好为n的路径条数，$g(n)$表示gcd为n的倍数的路径条数。则有：

$$g(d) = \sum_{d|n} f(n)$$

$$f(d) = \sum_{d|n} \mu(\frac{n}{d}) g(n)$$

我们需要的是$f(1)$，所以只要求

$$\sum_{n} \mu(n) g(n)$$

对于$g(n)$，可以使用可回退的并查集，把所有n的倍数的边并起来，在一次合并时把答案加上两个集合大小之积，做完一个$g(n)$后全部回退即可。考虑做一次并查集操作需要$O(\log N)$的时间，由于每条边会在该边权值约数的地方被做到，而约数个数是$O(\sqrt{M})$的（事实上远远不满，期望$O(\ln M)$），再加上枚举的时间复杂度，因此总时间复杂度是$O(N\sqrt{M}\log N + M\ln M)$。

考虑有修改操作的情况。我们只需把修改影响到的边放到最后再做并查集合并操作。当一次修改时，只需暴力把最后的$O(Q)$次可能被影响到的操作全部回退再重新合并即可。因此总时间复杂度是$O(N\sqrt{M}\log N + Q^2\sqrt{M}\log N + M\ln M)$。

可回退并查集不应采用路径压缩，因为那是基于均摊的。因此这样的并查集复杂度应为$O(\log N)$。

时间复杂度：$O((N+Q^2)\sqrt{M}\log N + M\ln M)$（实际$\sqrt{M}$远远不满），空间复杂度：$O(N+M+Q)$。

1.11 Random Number Generator

【来源】

CodeChef MARCH15 RNG

【大意】

已知数列A的前k项以及递推式：

$$A_i = \sum_{j=1}^{k} C_j A_{i-j}$$

求$A_N \mod 104857601$。

$N \le 10^{18}, k \le 30000$

时限：CC上15s

【题解】

这种递推被称为常系数线性递推。对于这类递推，最简单的是使用矩阵乘法，复杂度$O(k^3 \log N)$。更进一步的是对矩乘的优化，可以理解为利用矩阵的特殊性。设当前我们知道了A_i关于$A_{i-2^m k} \dots A_{i-2^m k+k-1}$的关系式。①根据这个关系式，把这个式子中每一项表示成关于这一项的前$2^m k$ 到$2^m k + k - 1$项的式子，这样我们就得到了A_i直接的关于$A_{i-2^{m+1}k} \dots A_{i-2^{m+1}k+2m-2}$ 的关系式。②再把$A_{i-2^{m+1}k+k} \dots A_{i-2^{m+1}k+2m-2}$根据初始时的递推式从右到左依次表示成关于这一项前$k$项的式子，我们不妨把这个过程称为“翻”，因为这就好像把右边这部分“翻”过去了一样。这样我们就得到了A_i关于$A_{i-2^{m+1}k} \dots A_{i-2^{m+1}k+k-1}$ 的关系式，

从而把m 倍增到了$m+1$。那么只要像矩阵乘法快速幂一样倍增上去即可。这么做复杂度降到了$O(k^2 \log N)$。

但是此题$k \leq 30000$，还不够。于是考虑继续优化上述倍增过程。我们发现，过程①其实是卷积，可以用FFT在$O(k \log k)$时间内实现，但过程②就比过程①复杂了：每“翻”一步的结果都会影响下一步，不能直接表示为卷积形式。对于这种“自我影响”的卷积，我们往往可以把式子用多项式表示出来，再使用更高级的多项式算法，如多项式求逆、多项式除法等方法。这里我采用的是多项式求逆：令B_i为过程②中A_i的关系式中$A_{i-2^{m+1}k+2k-1-i}$这一项前面的系数，D_i为每次要“翻”$A_{i-2^{m+1}k+2k-1-i}$这一项的时候之前这一项的系数，$B(x)=\sum_i B_i x^i$，$C(x)=\sum_i C_i x^i$，$D(x)=\sum_i D_i x^i$，则由“翻”的过程有

$$\begin{aligned} D(x) &= B(x)+C(x)D(x) \\ D(x) &= \frac{B(x)}{1-C(x)} \end{aligned}$$

我们只需要D的前k项，所以把这些多项式放到 $\bmod x^k$的模域下，要求D，只要求$1-C(x)$的逆，再乘上$B(x)$即可。最终我们要求D对左边的贡献，只需要求$C(x)D(x)$。

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

1.12 Devu and Locks

【来源】

CodeChef FEB15 DEVLOCK

【大意】

对于所有的$0 \leq M \leq MM$，求出各位数字之和不超过M且可以被P整除的N位十进制数的个数 $\bmod 998244353$。

$N \leq 10^9$

$P \leq 50, MM \leq 500$或$P \leq 16, MM \leq 15000$

时限：CC上5到15s

【题解】

首先，我们可以做一个DP：令$f[i][j][k]$表示各位数字之和恰为j且 $\bmod P = k$的i位十进制数的个数。则有转移：

$$f[i][j][k] \to f[i+1][j+c][(10k+c) \bmod P]$$

其中c是0 9中的某个数，表示下一位数字。由于N很大，考虑用倍增（类似矩阵乘法）来加速这个DP：把k 变成$10^i k$ 这个过程可以用线性时间实现；我们只要实现把每一维都加上某个0到9的值即可。这个可以用二维卷积实现。这样的复杂度就是$O(MM \times P \log(MM \times P) \log N)$，理论上可以通过，但实际由于常数太大，勉强可以卡过。

标算的DP是从低位到高位确定，不像我从高位到低位。这样可以利用一个性质：因为P很小，所以$10^i \bmod P$会出现循环。对于每个val，我们可以统计出有多少个$0 \leq x < N$使得$10^x = val$，记为cnt_{val}。易知不同的满足$cnt_{val} > 0$的val最多只有4种。我们把每种值的那些位置放在一起做DP，这样，我们就可以把上面DP中的最后一维省去了，因为这时状态中的$k = j \times val \bmod P$。这样，对于每种val，我们只要做一维的倍增卷积即可。最后只要把不同val的DP用二维卷积合并起来即可。

时间复杂度：$O(4 \times MM \log MM \log N + 3 \times MM \times P \log(MM \times P))$，空间复杂度：$O(P + MM)$

1.13 Payton numbers

【来源】

CodeChef FEB15 CUSTPRIM

【大意】

定义三元组(a,b,c)的乘法运算，其中$c=11$或者$c=24$。如果令$(a1,b1,c1)$和$(a2,b2,c2)$相乘：

令

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

$t=\lfloor\frac{s}{2}\rfloor+16(c1+c2)-c1c2$

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

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

如果s是偶数，那么结果就是$(A-540,B-540,24)$，不然就是$(A-533,B-533,11)$。

定义单位元A是对于任何B都满足$A\times B=B$的三元组，定义$zeroA$是对任何B都满足$A\times B=A$的三元组。定义一个三元组是素数当且仅当这个三元组不能表示成两个非零非单位元的三元组的乘积。

给定一个三元组，判断它是不是素数。

$T\leq 10^4,\ |a|,|b|\leq 10^7$

【题解】

令ω为满足方程$\omega^2=\omega-3$的解，即$\omega=\frac{1+\sqrt{-11}}{2}$，那么对于每一个三元组$(a,b,c)$，都有到域$Z[\omega]$的映射$\phi(a,b,c)=(33-2a-c)+(b-a)\omega$。于是问题就转化为判断域$Z[\omega]$下的数$a+b\omega$是否是素数。

定义共轭$(a+b\omega)'=(a+b-b\omega)$，那么令$Nx=xx'$，有下列结论：

- 如果x不是整数，那么x是质数当且仅当Nx是质数。
- 如果x是整数，那么x是质数当且仅当x是质数且要么$|x|=2$，要么$|x|\neq 11$且-11在模x域下没有二次剩余。

于是可以直接用欧拉判别法和miller-rabin 判断。

时间复杂度：$O(T\log|a|)$，空间复杂度：$O(1)$

1.14 Ranka

【来源】

CodeChef JAN15 RANKA

【大意】

现在有一种围棋，棋盘大小为9×9，两人分别执黑白棋，交替行棋。如果把同色棋子按四联通连接起来，就会有一些连通块。如果一个连通块不与任何一个空点相邻，这个连通块就呈“无气”状态。每一步，玩家必须在一个空点放置一个棋子或放弃本轮。如果玩家放置了棋子会导致对手的一些连通块“无气”，就将对手呈“无气”状态的棋子提出；否则就不能放置该棋子。为了避免死循环，“禁止全局同形再现”，如果出现“全局同形再现”，那么这一步就是非法的。

给定N，请你输出一场含N个合法行动的比赛。

$N=10000$

【题解】

这题是个构造题。有很多种构造方法。这里有一种很简洁的构造方法：让

第1个人把除了$(1,1)$的所有格子都填满，然后第2个人只要放在$(1,1)$，就可以把其他所有棋子消光；接下来再让第2个人把除了$(1,2)$的所有格子填满，然后第1个人再放在$(1,2)$；……这样的方案数就大约是$81 \times 81 \times 2$（对手放弃也算一步）。

时间复杂度：$O(N)$，空间复杂度：$O(1)$

1.15 Xor Queries

【来源】

CodeChef JAN15 XRQRS

【大意】

给定一个初始为空的整数数组以及M个询问：

- 0 在数组末尾添加一个数x
- 1 在区间$[L,R]$中找到一个y使得x xor y最大
- 2 删除末尾的k个数
- 3 在区间$[L,R]$中统计小于等于x的数的个数
- 4 在区间$[L,R]$中查找第k小的数（第k个顺序统计量，也就是相同的值算多次）

$M \leq 5 \times 10^5$

$1 \leq x \leq 5 \times 10^5$

时限：CC上1.5s

【题解】

此题是个数据结构题。可以用函数式trie来解决。在末尾添加/删除操作，只需用函数式trie在上一版本上插入/删除最后的一些版本即可。对于区间询问，由于询问都可以转化为逐位确定后求$[L,R]$中某一些确定高位下的数的个数，只需用函数式trie的相减即可。所有单步操作都可以在$O(bits)$时间内解决。

时间复杂度：$O(M\log x)$，空间复杂度：$O(M\log x)$

1.16 Divide or die

【来源】

CodeChef DEC14 DIVIDEN

【大意】

在坐标平面上给定一个N°角，问能否用尺规作图将其N等分？如能，你需要给出具体步骤。步骤数不得超过1000.

【题解】

由于三等分角是不可能的，因此，如果$N \bmod 3 = 0$，那就不能做到。否则就可以。构造一个36°角，再构造一个30°角，两角相减得到6°角，平分6°角得到3°角。通过3°角，我们可以把N°角不停减3，直到小于3 为止。如果$N \bmod 3 = 2$，就再平分这个2°角。最终我们得到了1°角。利用1°角可以很轻松地把N°角N 等分。

至于36°角的构造，可以用构造五边形的方法。

时间复杂度：$O(N)$，空间复杂度：$O(N)$

1.17 Course Selection

【来源】

CodeChef DEC14 RIN

【大意】

有M个学期，N门课。每门课都必须在某个学期上一次。一个学期可以上多门课。还有K个限制，第i个限制(A_i, B_i)表示课程A_i必须在B_i之前上。给定一张表格$X_{i,j}$表示课程i在第j学期上能获得的分数，如果第j学期没有课程i，$X_{i,j} = -1$。求最大总平均分。

所有数≤ 100

【题解】

此题与HNOI切糕一题类似。使用最小割模型。用割表示选择。由于答案要求最大化，为了适应最小割的最小化，把所有$X_{i,j} \neq -1$的$X_{i,j}$都变成$100 - X_{i,j}$或者$\sum_{j'} X_{i,j'} - X_{i,j}$，把$X_{i,j} = -1$变成$\infty$，最后再减去这些值取反即可。对第$i$门课在第$j$学期之后（包括第1学期之前）建点，记为$V_{i,j}$。连边（$(u, v, w)$ 表示从点u 到点v一条流量限制为w的边）：

对于$1 \leq i \leq N$，连边$(S, V_{i,0}, \infty)$和$(V_{i,M}, \infty)$

对于$1 \leq i \leq N, 1 \leq j \leq M$，连边$(V_{i,j-1}, V_{i,j}, X_{i,j})$，表示选择某个学期上课。

对于$1 \leq i \leq K, 1 \leq j \leq M$，连边$(V_{A_i,j-1}, V_{B_i,j}, \infty)$，表示限制。

1.18 Chef and Churu

【来源】

CodeChef NOV14 FNCS

【大意】

有一个长度为N的数组A，以及N个函数，第i个函数表示$A[L_i : R_i]$的和。现在进行Q次操作，有两种类型：

- $1\ x\ y$：把A_x改为y
- $2\ m\ n$：求第m到n个函数值的和

数据范围：$N, Q \leq 10^5$

【题解】

本题使用分块算法。方法有很多。我写的是这样的：把函数分块，块大小设为S。在整块处预处理前缀函数和（记录数组中每个位置对前缀函数和的贡献）。还要建立一棵树状数组维护A的前缀和。对于修改，在树状数组中修改（$O(\log N)$），并且修改每个整块处的前缀和的值（$O(\frac{N}{S})$）；对于询问，拆成两个前缀和，每个前缀和可以用预存好的整块处的前缀和的值再暴力扩展S个函数，每个函数只要在树状数组中询问一下即可（$O(S \log N)$）。

时间复杂度：$O(Q(\frac{N}{S} + S \log N))$，空间复杂度：$O(\frac{N^2}{S})$

易知在$S = \sqrt{\frac{N}{\log N}}$时有最优时间复杂度：$O(Q\sqrt{N \log N})$。

当然还可以对操作序列分块，复杂度可以降到$O(N\sqrt{N})$

1.19 Sereja and Order

【来源】

CodeChef NOV14 SEAORD

【大意】

有N个程序，每个程序都要在两台电脑上分别运行。第i个程序在第一台电脑上要运行a_i秒，在第二台电脑上要运行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。求一种方案使得最终完成所有程序在两台电脑上运行的任务的时间最少。

多组数据。

$N \leq 10000$, $\sum N \leq 2 \times 10^5$, $1 \leq A_i, B_i \leq 10^5$

【题解】

贪心。

首先可以确定最终总长len为$\max(\sum a_i, \sum b_i, a_i + b_i)$。然后我们在两台电脑的左端和右端共4个位置放，分别记4个位置指针为la, ra, lb, rb。

由于$a_i + b_i$大的程序容易对其他程序造成影响，为了尽可能减小影响，我们先按$a_i + b_i$从大到小依次尝试放到la和rb处，能放就放。

对于剩下的程序，按b_i从小到大依次枚举，如果$a_i \geq b_i$就尝试放到la和lb处；然后同样地，按a_i从小到大依次枚举，如果$b_i \geq a_i$就尝试放到ra和rb处。这样可以尽可能地利用$a_i \geq b_i$的程序错开两台电脑上中间可能引起冲突的la和rb指针。

最后，如果$la \geq len - rb$，说明b还有很大余地，所以我们把剩下的程序按$b_i - a_i$从小到大依次放到la和lb处；否则，相似的，说明a还有很大余地，就把剩下的程序按$a_i - b_i$从小到大依次放到ra和rb处。

时间复杂度：$O(N\log N)$，空间复杂度：$O(N)$

1.20 Children Trips

【来源】

CodeChef OCT14 TRIPS

【大意】

给定一棵N个点的树，每条边边权$\in\{1,2\}$。有M次询问，每次询问从点X到Y每次最多跳P的距离（不能跳到边上）最少要跳几步。

数据范围：$N,M\leq 100000$

时限：8s

【题解】

此题使用分块。设阈值S。对于$P>S$，那么最多只会跳$\frac{N}{S}$步。只需倍增暴力往上跳即可。对于$P\leq S$，预处理出每个这样的P往上跳一步到达的位置，然后再预处理倍增，在询问时只需$O(\log N)$的时间往上跳。让X,Y分别往上跳到不超过$LCA(X,Y)$的位置，X那部分表示最远能跳到的位置，Y那部分表示至少需要到达的位置。然后只需再特判一下，把两部分接起来即可。

时间复杂度：$O(SN\log N+M(\frac{N}{S}\log N+\log N))$

由于N,M同阶，当$S=\sqrt{N}$时有最优复杂度$O(N\sqrt{N}\log N)$

空间复杂度：$O(N\sqrt{N}\log N)$，将询问离线可以做到$O(N\log N)$

1.21 Union on Tree

【来源】

CodeChef OCT14 BTREE

【大意】

给定一棵n个点的树，树上每条边长度为1。现在有Q天，第i天给定k_i个特殊点，其中第j个特殊点为a_j，半径为r_j，可以覆盖距离它不超过r_j的所有点。问每天里有多少点被覆盖。

$n, Q \leq 5 \times 10^4$, $\sum k_i \leq 5 \times 10^5$

【题解】

对于每一天，建出这些点的虚树。然后在虚树上从下当上、从上到下做一遍DP更新r值，使得对于虚树上的一条边(u, v)（v是u的父亲）都有$r_v \geq r_u - dist(u, v)$, $r_u \geq r_v - dist(v, u)$。然后考虑容斥，对于虚树上的一条边$(u, v)$（$v$是$u$的父亲），可以找到这条边上的某个点$w$使得$r_u - dist(u, w) = r_v - dist(v, w)$。注意$w$可能在原来某条树边中点的地方，所以我们在原树上每两个点中间插入一个点。答案只要统计出每个虚树上的点以相应的半径覆盖的点数减去那些w点以$u - dist(u, w)$为半径覆盖的点数即可。

至于统计以某个点为中心的半径为某个值覆盖的点数只要构出点分树然后再点分树上容斥一下即可。

时间复杂度：$O((n + \sum k_i) \log n)$，空间复杂度：$O(n \log n)$

1.22 Rectangle Query

【来源】

CodeChef SEPT14 QRECT

【大意】

在一个二维平面上维护三种操作：

- 1. 插入一个矩形
- 2. 删除一个矩形
- 3. 给定一个矩形，询问当前有多少个矩形与这个矩形有公共点

数据范围：$Q \leq 10^5$

【题解】

首先，删除一个矩形相当于插入一个贡献为-1的矩形。那么就只有插入和询问了。对整个操作序列CDQ分治，问题就变成了先全部插入、再全部询问。考虑如何判断一个矩形与当前询问的矩形相交：x轴和y轴的投影线段分别相交。那么就可以容斥计算：总个数-左边的个数-右边的个数-下面的个数-上面的个数+左下的个数+左上的个数+右下的个数+右上的个数。每种个数都可以通过树状数组+扫描线得到。

时间复杂度：$O(Q \log^2 Q)$，空间复杂度：$O(Q)$

1.23 Fibonacci Numbers on Tree

【来源】

CodeChef SEPT14 FIBTREE

【大意】

给定一棵n个点的树，支持以下操作：

- 1. 给u到v路径加上一个斐波那契数列，其中第i个点加上fib_i，fib_i表示斐波那契数列第i项。
- 2. 询问以x为根时y的子树和。
- 3. 询问x到y路径上所有点权值和。
- 4. 让整棵树回到第i次操作后的状态。

答案可能很大，对10^9+9取模后输出。

数据范围：$n,Q\leq 10^5$

【题解】

对这颗树树链剖分。

对于操作1，由于$fib_i=\frac{1}{\sqrt{5}}((\frac{1+\sqrt{5}}{2})^i-(\frac{1-\sqrt{5}}{2})^i)$，其中$\sqrt{5}$在模$10^9+9$域下是存在的。这样就可以表示为加两种等比数列。这个可以用线段树打标记实现，标记就是把这段区间加上一个首项为多少、公比为某个固定常数的等比数列。这个标记显然是可以合并的。那么就可以用树链剖分维护这个操作。

对于操作3，同样只要树链剖分即可。但对于操作4，需要维护子树信息。这时只要利用DFS序，把树链剖分的线段树建到DFS序的线段树上。为了使得同一条链

上的节点在DFS序中连续，只需在DFS时优先往重链走。这样子树就是线段树上连续的一段区间，也很好维护。

对于操作4，只需把树链剖分的线段树可持久化。

时间复杂度：$O(Q\log^2 n)$，空间复杂度：$O(Q\log^2 n)$

1.24 Team Sigma and Fibonacci

【来源】

CodeChef AUG14 SIGFIB

【大意】

给定M, N，求

$$\sum_{\substack{x,y,z\geq 0\\ x+y+z=N}} (6\times x\times y\times z\times fib_x\times fib_y\times fib_z) \mod M$$

共T组数据。$N\leq 10^{18}, M\leq 10^5, \sum M\leq 10^6$

【题解】

好题！此题有很多做法。官方题解是$O(M)$的，利用了循环节。还有很多生成函数的做法。我是这样做的：

为了方便起见，这里所有数列的下标都从0开始。

首先，对于任意一个D阶常系数线性递推的数列f，我们都可以写出他的生成函数：

$$F(x)=\frac{P(x)}{1-c_1x-c_2x^2-\cdots-c_Dx^D}$$

其中c_i表示递推式中f_{n-i}前的系数，$P(x)$为某个多项式。

具体证明可以参考斐波那契数列生成函数推导所用的移位法。

反之，对于生成函数$G(x)=\frac{P(x)}{Q(x)}$（其中$P(x),Q(x)$都是多项式，$Q(x)$常数项非零，阶数满足D的限制），就可以写出该生成函数的递推式。

那么斐波那契数列的生成函数就是$\frac{x}{1-x-x^2}$。

现在我们希望得到要求的东西关于N的函数的生成函数$F(x)$。观察式子可以发现，我们最好求出生成函数$G(x)=\sum_{i\le 0} i\times fib_i\times x^i$，那么$F(x)=6G(x)^3$。如何得到$G(x)$？斐波那契数列的生成函数是已知的，只要在此基础上把第i项乘上一个i即可。这让我们联想到求导。所以，

$$\begin{aligned} G(x) &= x\left(\frac{x}{1-x-x^2}\right)' \\ &= \frac{x+x^3}{\left(1-x-x^2\right)^2} \end{aligned}$$

$$\begin{aligned} F(x) &= 6G(x)^3 \\ &= \frac{6\left(x+x^3\right)^3}{\left(1-x-x^2\right)^6} \end{aligned}$$

注意到这个式子展开后就可以12阶常系数线性递推。

至于常系数线性递推的方法，在RNG一题中已有了详尽的各种复杂度层次的算法。这里采用$O(D^2\log N)$ 的算法即可。

然而这样还是被卡常数了。所以预处理小模数的循环节，询问时直接输出即可。

时间复杂度：$O(12^2\log N)$，空间复杂度：$O(12)$

1.25 Push the Flow!

【来源】

CodeChef AUG14 PUSHFLOW

【大意】

给定一个n个点m条边的无向仙人掌图（每个点至多属于一个简单环），每条边有个容量。维护下列两种操作：

- 0 S T：求以点S为源点、T为汇点时的最大流（$S \neq T$）
- 1 X $NEW\_CAPACITY$：把边X的容量修改为$NEW_CAPACITY$

有q次操作。

$n \leq 10^5$, $m, q \leq 2 \times 10^5$, $1 \leq$ 边的容量 $\leq 10^9$

【题解】

可以把每个环缩点然后对剩下的树树链剖分，复杂度$O(\log^2 n)$，但这样太麻烦了。

考虑对于每个环，断开容量最小的边（如果有多条随便选一条），然后把这条边的容量加到环中其他所有边上。那么我们只要维护一棵树就好了，询问S到T的最大流就是S到T树上路径的最小值。修改边权时，如果当前环上的最小边发生了变化，就要通过link/cut操作替换最小边。因此我们需要用link-cut tree维护link、cut、链加、链覆盖、链最小值、链最小值所在边。

时间复杂度：$O((n+q)\log n)$，空间复杂度：$O(n)$

1.26 Game of Numbers

【来源】

CodeChef JULY14 GNUM

【大意】

已知两个长度为N的数组A, B，每一轮你要选出两个数对(i, j)和(p, q) 满足$A_i < B_j$，$A_p > B_q$，$\gcd(A_i, B_j, A_p, B_q) \neq 1$，$(i, j)$不在集合$S1$中，$(p, q)$不在集合$S2$中。然后把$(i, j)$加入集合$S1$，$(p, q)$加入集合$S2$，问最多能进行多少轮游戏。

数据范围：$N \leq 400, 1 \leq A_i, B_i \leq 10^9$

【题解】

一个简单的思路是，把所有$\gcd(i, j) \neq 1$的数对都建一个点，从$A_i < B_j$的点向$A_i > B_j$的点连边（满足$\gcd \neq 1$），跑最大流。但这样的边数是$O(N^4)$。

考虑优化连边。$\gcd \neq 1$就是至少有一个公共质因子。那么对于所有出现过的质因子建点，然后每个数对往$\gcd(i, j)$中出现的质因子的点连入（出）∞边。这样显然与原图等价。

设$M = \max(A_i, B_j)$。时间复杂度：$O(maxflow(N^2 + N \log M, N^2 \log M))$，空间复杂度：$O(N^2 \log M)$

1.27 Sereja and Equality

【来源】

CodeChef JULY14 SEAEQ

【大意】

定义两个长度为n的数组是相似的如果对于每个$i(1 \le i \le n)$ $CA(A[i]) = CB(B[i])$，其中$CX(x)$等于满足$X[j] < x$的j的个数。

对于两个长度为n的排列$P1$和$P2$定义$F(P1, P2)$为满足$P1[l..r]$与$P2[l..r]$相似且$P1[l..r]$包含不超过E个逆序对的(l, r)的个数。

求对于所有的$(P1, P2)$ $F(P1, P2)$之和。

T组数据，每组数据给出两个数n, E

$1 \le T \le 10000$，$1 \le n \le 500$，$1 \le E \le 1000000$

【题解】

类似期望的线性性，我们单独考虑每个(l, r)的贡献。

相似的意义在于两个数组中元素的大小关系是一样的。所以我们可以用长度$l = r - l + 1$的排列来代替大小关系的表述。

那么还要满足逆序对个数不超过E。这个可以DP：定义$f[i][j]$表示有多少长度为i的排列使得逆序对个数不超过j。考虑最大元素的位置，有递推式：$f[i][j] = f[i][j-1] + (f[i-1][j] - f[i-1][j-i])$。这样就可以$O(n^3)$预处理出$f$。

考虑枚举长度l，那么有$n - l + 1$个不同的位置。有$f[l][E]$种逆序对个数不超过E的排列，每种排列在每个数组上都有$\binom{n}{l}$种填法，同时每个数组其他位置共有$(n-l)!$种填法。所以答案就是

$$\sum_{l=1}^{n} (n - l + 1) \times \left(\binom{n}{l} \times (n-l)! \right)^2 \times f[l][E]$$

时间复杂度：$O(n^3 + Tn)$，空间复杂度：$O(n^3)$

1.28 Two Companies

【来源】

CodeChef JUNE14 TWOCOMP

【大意】

给定一棵N个点的树，有两个树上链的集合A, B。每个集合内的链有一个权值。如果两条链有公共点，那么就称这两条链相交。请分别从A, B中选出一些链使得从A中选出的链与B 中选出的链不相交。最大化选出的链的权值和。

$N \leq 10^5$，$|A|, |B| \leq 700$，权值$\leq 10^6$。

时限：2s

【题解】

如果可以两两判断两条链是否相交的话，剩下的就是经典的最小割：不割表示选择，每条链建一个点，源向它/它向汇连流量限制为权值的边，在相交的两条链之间连一条边，表示两者不能同时选择。

所以只要解决判断两条链是否相交的问题。我们可以把无根树变成有根树，以1号点为根。为了方便，我们把一条链(u, v)拆成两条链：$(LCA(u, v), u)$和$(LCA(u, v), v)$，这样只要判断两条从上往下的链$(u1, v1), (u2, v2)$是否相交即可。考虑不相交的情况：要么$u1$和$u2$不是祖先关系，要么是祖先关系但不想交。为了方便，设$u1$为$u2$的祖先，那么就要满足$u2$不在链$(u1, v1)$上，即$u2$不是$v1$的祖先。对于祖先关系的判断，只需用DFS序。

时间复杂度：$O((N + |A| + |B|) \log N + |A||B| + maxflow(|A| + |B|, |A||B|))$，空间复杂度：$O(N \log N + |A||B|)$

1.29 Sereja and Arcs

【来源】

CodeChef JUNE14 SEAARC

【大意】

x轴上有N个点$(1,0),(2,0),\ldots,(N,0)$，第i个点有颜色A_i。在任意两个相同颜色的点之间连一条在x轴上方的相应颜色的弧。问有多少对不同颜色的弧相交。答案 $\bmod 10^9+7$。

$1 \le N \le 100000, 1 \le A_i \le 100000$

时限：5s

【题解】

记两种颜色分别为A,B，那么就是要求$ABAB$这样的对数。

对于每种颜色出现的次数“分块”。设阈值S。

情况1：A,B中有一种颜色出现的次数大于S。那么对于另一种颜色的每个位置统计出这个位置左边/右边有多少这种颜色。比如A出现的次数大于S，那么对于每个B出现的位置统计出sum_i，记左边的B的位置时l，右边是r，那么对答案的贡献就是$sum_l \times (sum_r - sum_l) = sum_r \times sum_l + sum_l^2$，只要统计出左边所有$l$的$sum_l$与$sum_l^2$之和即可。由于出现次数大于$S$的颜色种数不会超过$\frac{N}{S}$，而位置只有$N$ 个，所以这一步总复杂度为$O(\frac{N^2}{S})$。

情况2：A,B两种颜色出现的次数都不超过S。那么这样的弧一共不会超过NS，因为每个这种颜色的点都只会向不超过S个点连弧。我们可以用二维数点（扫描线+树状数组）来解决。这样总复杂度为$O(NS\log(NS))$。

由于$S \le N$，所以$O(\log(NS)) = O(\log N)$。所以总复杂度为$O(\frac{N^2}{S}+NS\log N)$，

当$S=\sqrt{\frac{N}{\log N}}$时有最优时间复杂度$O(N\sqrt{N\log N})$，空间复杂度$O(N)$。

1.30 Dynamic Trees and Queries

【来源】

CodeChef MAY14 ANUDTQ

【大意】

给定一棵N个点的带点权的有根树。有M次操作，每次操作可能是：

- 1. 在某个点下面新建一个叶子。
- 2. 把某个子树内所有点加上某个值。
- 3. 删除某个子树。
- 4. 询问某个子树内所有点权和。

强制在线。

$N, M \leq 10^5$

【题解】

用平衡树维护该树的括号序列。对于操作1、3，只需在相应位置插入/删除；对于操作2，区间加标记；操作4，区间和。

时间复杂度：$O((N+M)\log N)$，空间复杂度：$O(N)$

1.31 Sereja and Subsegment Increasings

【来源】

CodeChef MAY14 SEINC

【大意】

在模4域下，有两个长为n的数组A, B。每次操作可以把A数组的某一段区间在模4域下加1。问最少需要多少次操作才能把A变成B。

$1 \leq n \leq 10^5$

【题解】

首先，求出$b_i = B_i - A_i \bmod 4$。然后由于是区间问题，所以差分，求出$a_i = b_{i+1} - b_i$（注意没有模）。由于在模4域下，我们可以把b的一个区间加4，也就是选择$i < j$，把a_i加4、a_j减4。可知对于一个a最终所需的操作数是$\sum \max(0, a_i)$。我们需要最小化a中所有正数之和。

一些性质：

1. $-3 \leq a_i \leq 3$。

2. 不必对同一个a_i又加又减。因为对于如果两个区间前后相接，可以合并成一个区间。

3. 区间加操作具有可交换性。

4. 不必加一个非负的数。假设最优方案需要加一个这样的数。把该操作换到最后进行。由于不必对同一个数又加又减，所以该数只可能被加，该数始终非负。把该数加4的同时必然要把另一个数减4。把该数加4使得答案也加4，而把另一个数减4最多只能使答案减4。故不优。

5. 不必减一个非正的数。假设最优方案需要减一个这样的数。把该操作换到

最后进行。由于不必对同一个数又加又减，所以该数只可能被减，该数始终非正。把该数减4的同时必然要把另一个数加4。把该数减4答案不变，而把另一个数加4则有可能使答案增加。故不优。

6. 由4、5可知，a_i始终满足$-3 \leq a_i \leq 3$。

7. 由4、5可知，一个正数被减到非正时不会继续被减，一个负数被加到非负时不会继续被加。

8. 不必加一个≥ -1的数。因为把这个数加4至少会使答案加3，而由于相应被减的数≤ 3，所以答案至多只会减3。故不优。

9. 不必减一个≤ 1的数。因为把这个数减4至多只会使答案减1，而由于相应被加的数≥ -3，所以答案至少会加1。故不优。

10. 由8、9可知，可以忽略所有$-1 \leq a_i \leq 1$的a_i，因为它们不会被操作。

现在，假设-3和3都具有1的势能，而-2和2具有0的势能。我们发现，一次操作后答案减少的值就是两个数的势能之和，并且势能之和减少为0，这就好像“释放”了这些势能。1个势能想要被释放，就要找一个相匹配的数来帮忙。而势能为0的-2和2的功能仅仅是帮助其他数释放势能，而且-2和2的匹配不会释放势能。

因此，我们得到了一个贪心算法：从左往右扫，碰到一个3或2时，尝试在之前寻找-3，如果有-3，那么就匹配掉。否则就再找-2匹配。

时间复杂度：$O(n)$，空间复杂度：$O(n)$

1.32 Chef and Graph Queries

【来源】

CodeChef MARCH14 GERALD07

【大意】

给定一张N个点M条边的无向图。有Q次询问，每次询问如果只保留编号为$[L_i, R_i]$的边的话图中有多少个连通块。

$1 \leq N, M, Q \leq 2 \times 10^5$

【题解】

离线。我们从小到大枚举询问区间的右端点。用LCT维护最大编号生成森林：插入当前这条边时如果构成环那么删去环上编号最小的边。对于一个右端点在此处的询问，我们只需求出当前森林里有多少条编号大于等于左端点的边即可。这个只要用树状数组。

时间复杂度：$O(M \log N)$，空间复杂度：$O(N + M + Q)$

1.33 The Street

【来源】

CodeChef MARCH14 STREETTA

【大意】

有长度为N的数列A和B，初始时A中元素全是$-\infty$，B中元素全是0。有M次操作。维护3种操作：

- 1. 把A中一段区间与一个等差数列取max。
- 2. 把B中一段区间加上一个等差数列。
- 3. 询问$A_i + B_i$。

$1 \leq N \leq 10^9, 1 \leq M \leq 3 \times 10^5$

【题解】

对数列A, B分别建立线段树。由于N很大，线段树需要动态开点。

对于操作2，只需在线段树节点上打上加上首项为多少、公差为多少的标记，标记可以合并。

对于操作1，在线段树节点上记录这段区间被一个首项为多少、公差为多少的等差数列取max。修改时，在新增的等差数列与记录的原有的等差数列进行比较。如果比较结果是整段区间都是某个数列优，就保留那个数列。否则就一定是前半段是一个数列优，后半段是另一个数列优。这种情况必然有一半的区间是完全某个数列优的。我们保留那个数列，并把另一个数列传到相应的儿子中。

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

1.34 Graph Challenge

【来源】

CodeChef FEB14 DAGCH

【大意】

已知一张N个点M条边的有向图可以从某个点出发按照能走就走的DFS遍历所有点，并且编号为i的点就是第i个被遍历到的点。定义一个点y的supreme vertex为某个x，当且仅当存在一条有向路$x, v_1, v_2, \ldots, v_{k-1}, y$且$x \leq y \leq v_i$。定义一个点的superior vertex为这个点编号最小的supreme vertex。现有Q个询问，每个询问给定一个点v，求有多少点的superior vertex是v。

【题解】

如果我们把DFS树构出来的话，那么superior vertex就是有向图必经点树（Dominator Tree）的Lengauer-Tarjan算法中的半必经点。因此，只要做一遍和这个DFS顺序一样的Lengauer-Tarjan算法即可。为此，我们只需把每个点的出边按编号排序。

时间复杂度：$O((N+M)\alpha(N)+Q)$，空间复杂度：$O(N+M)$

1.35 Count on a Treap

【来源】

CodeChef FEB14 COT5

【大意】

Treap对于key值是一棵二叉搜索树，同时对于$weight$值是一个堆。

维护一个max-Treap（就是大根堆）支持下列操作：

- 0 key $weight$: 根据给定的key和$weight$插入一个新点。
- 1 key: 删除key值对应的点。
- 2 key_u key_v: 返回u和v在树上的距离。

保证所有点的key值都不同，$weight$值也都不同。

$1 \leq N \leq 2 \times 10^5, 0 < key, weight < 2^{32}$

【题解】

我们维护按key值排序的序列。首先，答案是$depth_u + depth_v - 2depth_{LCA(u,v)}$，其中$LCA(u,v)$就是$[u,v]$中$weight$最小的元素。对于某个点$u$的深度就是从$u$开始往前的上升序列长度+从$v$开始往后的上升序列长度。因此，只要维护插入、删除、区间上升序列询问。考虑用线段树。那么最关键的一步就是合并两个相邻区间。对于每个区间记录这个区间的上升序列长度以及区间最大值（也就是上升序列的最后一个元素）。对于当前要合并的两个左右区间，答案就是左区间的上升序列长度加上在右区间中传入左区间的最大值后的上升序列长度。那么我们就要解决对于某个区间传入开始的一个值得到的上升序列长度。如果传入的值大于这个区间左半边的最大值，那么就只要忽略左半边，把这个值传入右半边；如果传入的值小于左半边的最大值，那么就传入左半边，然后再加上整个区间的答案减去左半边的答案即可。这样对于一个区间就可以在$O(\log N)$ 时间内解决。由于每次操作都要合并$O(\log N)$ 次区间，所以每次时间复杂度为$O(\log^2 N)$。由于key 范围较大，最好离散一下。

时间复杂度：$O(N \log^2 N)$，空间复杂度：$O(N)$

1.36 Counting D-sets

【来源】

CodeChef JAN14 CNTDSETS

【大意】

定义N维空间中$(a_1, a_2, \ldots, a_N)$和$(b_1, b_2, \ldots, b_N)$两点的距离为$\max_{1 \le i \le N} |a_i - b_i|$。定义$N$维空间中一个点集的直径为该点集中距离最大的一对点的距离。定义两个点集属于同一类当且仅当其中一个点集可以通过平移变换变成另一个点集。统计直径为D的不同的N维整点集的类的数目 $\bmod 10^9 + 7$。

有T组数据。

$T \leq 10, N \leq 1000, D \leq 10^9$

【题解】

定义$F(N,D)$为直径不超过D的不同的N维整点集的类的数目。那么答案就是$F(N,D)-F(N,D-1)$。如何计算$F(N,D)$？考虑容斥。先算出不考虑平移变换的情况下的集合个数，然后再减去某一维可以平移的集合数，再加上某两维可以平移的集合数……这里某一维可以平移指的是没有点在该维上取到0。用式子写出来就是：

$$F(N,D)=\sum_{i=0}^{N}(-1)^{N-i}\binom{N}{i}f(N-i,D-1,i,D)$$

其中$f(N-i,D-1,i,D)$表示有$N-i$维的最大距离不能超过$D-1$、i维的最大距离不能超过D的集合数。最大距离不超过D就相当于所有点的这一维都必须在$[0,D]$范围内。由此可知

$$f(N-i,D-1,i,D)=2^{(N-i)^{D-1}i^D}$$

所有指数都可以快速幂。组合数可以预处理阶乘和阶乘逆元。

时间复杂度：$O(N+TN(\log D+\log(10^9+7)))$，空间复杂度：$O(N)$

1.37 Counting The Important Pairs

【来源】

CodeChef JAN14 TAPAIR

【大意】

给定一张N个点M条边的连通无向图，求有多少种删掉两条边的方案使得图不

连通。

$N \leq 10^5, M \leq 3 \times 10^5$

【题解】

此题类似BZOJ3563 DZY Loves Chinese。我们可以跑出一棵DFS树，然后给每条非树边随机一个边权。对于每条树边，计算出所有跨越它的非树边的边权的异或和。易知当被割的边集的边权的异或和为0时就很可能能割开整张图。当然也有一定概率误判，不过当随机边权的范围足够大（比如2^{64}）时误判概率极小，可以忽略。因此只要求出有多少对边的边权相同或其中一条边的边权为0。这个只要把所有边权排个序统计一下每种边权的个数算一下就可以了。

时间复杂度：$O(M \log M)$，空间复杂度：$O(M)$

1.38 Query on a tree VI

【来源】

CodeChef DEC13 QTREE6

【大意】

给定一棵n个点的树，每个点有黑白两种颜色，初始时都是黑色。有m次操作，每种操作可能是：

- 0 u: 询问点u所在同色连通块中的点数。
- 1 u: 将点u反色。

$n, m \leq 10^5$

【题解】

我们可以维护两棵树：黑树和白树。对于每棵树只要维护添加一个点和删除一个点。对树进行树链剖分。在每个点上维护这个点所有黑（白）儿子所在同色连通块以及它本身的点数。询问时只需找到它的祖先中离它最远的同色点询问一下维护的值。修改时只需在它到它最近的异色点的路径上加或减它的值即可。由于只有区间加、单点询问，用树状数组即可维护。找最近同色点也可以用树状数组维护。

时间复杂度：$O(m \log^2 n)$，空间复杂度：$O(n)$

1.39 Petya and Sequence

【来源】

CodeChef DEC13 REALSET

【大意】

给定一个长度为n的序列$A[0 \ldots n-1]$，问是否存在一个长度为n的序列$B[0 \ldots n-1]$满足：

- 至少存在一个$0 \le i < n$满足$B[i] \neq 0$
- 对于任意$0 \le j < n$满足$\sum_{0 \le i < n} A[i] \times B[(i+j) \bmod n] = 0$

T组数据。

$T \le 100$, $n \le 3 \times 10^4$, $|A[i]| \le 1000$, $\sum n \le 1.5 \times 10^5$

【题解】

问题就是问A构成的循环矩阵X的行列式是否是0。如果$\gcd(f(x), x^n - 1)$的次数为d，那么这个矩阵的秩就是$n - d$，其中$f(x) = \sum_{i=0}^{n-1} A_i x^i$。定义分圆多项

式$\Phi_n(x)$满足$\prod_{d|n}\Phi_d(x)=x^n-1$，可以发现$\Phi_n(x)$无法表示为两个次数均不为0的多项式之积。所以只要判断是否存在一个n的约数d使得$\Phi_d(x)|f(x)$。这个问题等价于$f(x)\times\prod_{i|d且i是质数}(x^{\frac{d}{i}}-1)$被$x^d-1$整除。所以就可以用简单的模拟来判定了。

时间复杂度：$O(n\log n+n\times n$的因子个数$)$，空间复杂度：$O(n)$

1.40 Gangsters of Treeland

【来源】

CodeChef NOV13 MONOPLOY

【大意】

给定一棵N个点的有根树。初始时每个点都有一个不同的颜色。有Q次操作，每次操作可以是：

- 1. 把某个点到根路径上所有点都涂上一种新的颜色。
- 2. 询问某个子树中所有点到根路径上经过的不同颜色种数的平均值。

T组数据。

$T\leq 15,\sum N,\sum Q\leq 10^5$

【题解】

注意到操作1就好像是LCT的access操作，而相同颜色之间的边就是重边，不同颜色之间的边就是轻边。某个点到根经过的不同颜色种数就是轻边的条数。由于LCT中access操作splay均摊次数是$O(N\log N)$，所以这里轻边均摊修改$O(N\log N)$。我们只要用线段树/树状数组维护DFS序上区间加、单点询问即可。

时间复杂度：$O(N \log^2 N)$，空间复杂度：$O(N)$

1.41 Queries With Points

【来源】

CodeChef NOV13 QPOINT

【大意】

二维平面上有一些没有公共部分的简单多边形。在线询问：给定一个点，求这个点在哪个多边形内。如果不在任何多边形内，输出-1。

多边形边数之和$M \leq 3 \times 10^5$, 询问数$Q \leq 10^5$

【题解】

方法1：对x坐标离散后建立线段树。对于多边形的每条边（分为上边和下边，不考虑竖边），在相应的x坐标区间上覆盖一下。然后，对于线段树中每个点上记录的线段排个序。询问时只要在线段树上相应的点上的线段里二分一下，找到询问点向上的射线碰到的第一条边，根据这条边的信息判断一下位置关系就好了。

时间复杂度：$O((M+Q)\log^2 M)$，空间复杂度：$O(M \log M)$

方法2：如果离线，就是点定位。用扫描线从左往右扫过去，用平衡树维护边的加、删。而在线只要把平衡树可持久化即可。

时间复杂度：$O((M+Q)\log M)$，空间复杂度：$O(M \log M)$

1.42 Fibonacci Number

【来源】

CodeChef OCT13 FN

【大意】

给定P, C，求满足$fib_n \bmod P = C$的最小的n，如果不存在就输出-1。

T组数据。

$T \leq 100, 0 \leq C < P \leq 2 \times 10^9$。$P$为质数且$P \bmod 10$为完全平方数。

【题解】

写出斐波那契数列通项公式，那么由P的条件可知2存在逆元、$\sqrt{5}$存在。那么就只要解方程：$x^n - x^{-n} = a$，其中$x = \frac{1+\sqrt{5}}{2}$。解得$x^n = \frac{a \pm \sqrt{a^2+4}}{2}$。如果$\sqrt{a^2+4}$在模意义下存在，就可以用BSGS解出可能的$n$，否则无解。注意$n$要分奇偶讨论。求一个数在模意义下的平方根可以用Cipolla's algorithm。

时间复杂度：$O(\sqrt{P})$，空间复杂度：$O(\sqrt{P})$

1.43 Two Roads

【来源】

CodeChef SEPT13 TWOROADS

【大意】

平面上给定N个点，要划两条直线，使得每个点到两条直线中较近的一条的距

离的平方之和最小。求最小值。

$3 \leq N \leq 100$

【题解】

如果只划一条直线，就是直线拟合，只要线性回归算一下即可。

现在有两条直线，而且还有距离取最小值，十分不便。为了去掉最小值，我们考虑每条直线能管辖的区域，发现按角平分线分隔是4块，一条直线管辖斜对角的两块，而且两条角平分线垂直。把这两条角平分线中的一条逆时针旋转（另一条也跟着转），必然要么是这条线被两个点刚住，要么是另一条线被两个点刚住。然后没被刚住的线再往某个方向平移，必然能被某个点刚住。所以只要枚举两个刚住的点，再枚举另一条直线在哪个点的区间，就可以枚举到所有可能的管辖区域内的点集划分。然后只要分别对两个点集用一条直线拟合即可。由于是最小化，因此枚举到一个不是最优解的点集划分从而产生非法的拟合直线必然不会优于答案，无需担心。至于第三个枚举，可以把所有点按照与前两个点前后形成的向量的点积排序，然后依次扫过去，这样就可以快速维护所需的信息（点数、横纵坐标之和、横纵坐标平方之和）。

时间复杂度：$O(N^3 \log N)$，空间复杂度：$O(N)$

1.44 To Queue or not to Queue

【来源】

CodeChef SEP13 TMP01

【大意】

你需要维护一个小写字母组成的字符串S。有两种操作：

- 在S的末尾添加一个字符。
- 删除S的第一个字符。

在每次操作之后求当前串S的本质不同的子串个数模10^9+7。

操作数为Q。

$Q \leq 10^6$

【题解】

Q太大，只好$O(Q)$了。

如果没有删除操作，那么可以用Ukkonen算法构造后缀树。为了方便说话，我们把连续的没有分叉的点称为一条链。对于删除操作，就是要删除当前最长的那个后缀。只要在添加时记录每个后缀对应的叶子，就能找到当前串最长的那个后缀对应的叶子节点。然后从这个点往上一路删去这条链（因为可能有一条链上有多个结点）。但是链上有可能存在一些隐藏的后缀结点。如果真是这样的话，深度最深的那个隐藏的后缀结点必然是激活点，因为删完之后这个点就变成了叶子，而在最后一个字符添加进来之前这个点最后那个字符必然是不存在的，就好像这个字符是新添加的一样，由每个后缀对应的结点序列（不妨称为后缀结点序列）的性质可知，后缀结点序列中在这个点之前的结点必然都是叶子，而这个点在删之前的后缀树上不是叶子。所以这个点就是激活点。因此对于这种情况只要截取到激活点处为止，然后把激活点指针移动到往下一个后缀就好了。

时间复杂度：$O(Q)$，空间复杂度：$O(Q)$

1.45 Music & Lyrics

【来源】

CodeChef AUG13 LYRC

【大意】

有M个串T和N个串S，要求每个T串在所有S串中出现的次数。字符集为大小写字母、数字及“-”。

【题解】

对所有T串建AC自动机。把所有S串在AC自动机上走一遍，走到的所有点都把cnt加1。最后只需对于每个T串询问在$fail$树子树中所有cnt之和。

时间复杂度：$O(63\sum|T|+\sum|S|)$，空间复杂度：$O(63\sum|T|)$

1.46 Prime Distance On Tree

【来源】

CodeChef AUG13 PRIMEDST

【大意】

给定一棵树。等概率随机选两个点，求两点间距离为质数的概率。

$N\leq 50000$

时限：5s

【题解】

由于是路径问题，对树点分治。对于每个点分结构，统计出到根距离为i的点数cnt_i。求出$\sum_{i\ is\ prime}\sum_j cnt_j cnt_{i-j}$，这个卷积用FFT做。当然这样重复计算了同一棵子树中的点对，所以只需再减去每棵子树中的答案。

时间复杂度：$O(N\log^2 N)$，空间复杂度：$O(N)$

1.47 Across the River

【来源】

CodeChef JULY13 RIVPILE

【大意】

平面上有N个点。有M种圆盘，每种圆盘有一个价格。你需要给每个点买至多一个圆盘（这个点作为圆心），使得存在一条只经过圆盘上的点的路径从直线$y = 0$到$y = W$。

$1 \leq T \leq 10, 1 \leq N, M \leq 250$

【题解】

如果建图，对于每个点建M个点表示这个点上选择了哪种圆盘，在有公共点的圆盘之间连边，那么就是一个最短路问题。然而这样边数是$O(N^2M^2)$的。考虑优化连边。对于每个点往比它半径小一点的点连0的边。然后只要在两个点之间刚刚好能相交的点之间连边即可。边数减少到$O(N^2M)$。

时间复杂度：$O(N^2M\log(NM))$，空间复杂度：$O(NM)$

1.48 Two k-Convex Polygons

【来源】

CodeChef JUNE13 TKCONVEX

【大意】

给定n根木棒的长度。问是否存在一种从中选出$2k$根木棒拼成两个凸k边形。凸多边形不能有两条邻边平行。如果存在，给出一种方案。

$n \leq 1000, 3 \leq k \leq 10$，$1 \leq$每根木棒长度$\leq 10^9$

【题解】

k根木棒可以拼成凸k边形当且仅当最长的一根木棒长度小于其他所有木棒长度之和。因此，我们把木棒按长度排成升序，那么最优方案一定是取连续的一段木棒。对于取两段不相交的长度为k的区间的情况，可以预处理后扫一遍。对于取连续的$2k$根木棒的情况，我们需要把这$2k$根木棒分成两个k根木棒的集合，可以假定最长的那根木棒是属于第二个集合。

时间复杂度：$O(n(\binom{2k-1}{k}+k))$，空间复杂度：$O(n)$

1.49 Count Special Matrices

【来源】

CodeChef JUNE13 SPMATRIX

【大意】

一个$N \times N$的矩阵A是特殊的，当且仅当满足下列条件：

- $A_{x,x} = 0$ for $1 \leq x \leq N$
- $A_{x,y} = A_{y,x} > 0$ for $1 \leq x < y \leq N$
- $A_{x,y} \leq \max(A_{x,z}, A_{z,y})$ for $1 \leq x, y, z \leq N$

- $A_{x,y} \in \{1, 2, \ldots, N-2\}$ for $1 \leq x < y \leq N$
- 任意$k \in \{1, 2, \ldots, N-2\}$ 存在$x, y \in \{1, 2, \ldots, N\}$ 满足$A_{x,y} = k$

现有T组询问，每组给出一个N，求有多少特殊矩阵。答案 $\bmod 10^9 + 7$。

$T \leq 10^5$, $3 \leq N \leq 10^7$

【题解】

通过推导，可以得到答案为

$$\frac{N!(N-1)!}{2^{N-1}} \left(\frac{N}{2} - \frac{2}{3} - \frac{H_{N-1}}{3} \right)$$

其中$H_N = \sum_{k=1}^{N} \frac{1}{k}$

所以只要预处理阶乘和逆元等，就可以做到每次询问$O(1)$了。

时间复杂度：$O(N+T)$，空间复杂度：$O(N)$

1.50 Queries on tree again!

【来源】

CodeChef MAY13 QTREE

【大意】

给定一个N个点的环套外向树（保证环长为奇数），边有边权。维护Q次操作，有两种：

- f u v: 把点u到v的最短路径上的边权取相反数。

- ? u v: 询问点u到v的最短路径上的边权组成的序列的最大子段和。

$1 \leq N, Q \leq 10^5$

【题解】

如果维护的是一条链，那么只要用线段树记录每段区间中所有数之和、左/右边开始最大/小的连续一段、最大/小子段和，维护即可。如果维护的是树，那么用树链剖分。对于环套外向树，只要把环的一处断开，这样整个结构就变成了树，然后对于修改，在相应的链上修改即可。

时间复杂度：$O(Q \log^2 N)$，空间复杂度：$O(N)$

1.51 String Query

【来源】

CodeChef APRIL13 STRQUERY

【大意】

给定一个串，维护在串开头、结尾、中间插入删除字符，询问某个字符串在这个串中出现的次数（允许重叠）。

$Q \leq 1.5 \times 10^5$，询问串总长$\leq 1.5 \times 10^6$，字符集为小写字母。

【题解】

先考虑只有开头操作的情况。这个可以用后缀平衡树来维护。询问时只要在后缀平衡树上二分即可。由于询问串总长的限制，时间复杂度为$O((Q +$ 询问串总长$) \log Q)$。

考虑没有中间操作的情况。我们可以把这个串从正中间分开，维护左右两棵后缀平衡树。一旦有一边被删空了，就全部暴力重构，重新从正中间分开。设势能为左右两边长度之差的绝对值。通过势能分析易知每次操作均摊复杂度依旧为$O(\log Q)$。

考虑有中间操作的情况。我们只要维护从中点分开的左右两部分，每部分由两棵平衡树像上面说的那样维护头尾操作。当中点移动时，只要把中间的那个字符从一边移到另一边即可。

时间复杂度：$O((Q+\text{询问串总长})\log Q)$，空间复杂度：$O(Q+\text{询问串长度})$

1.52 Little Elephant and Colored Coins

【来源】

CodeChef MARCH13 LECOINS

【大意】

给定N种硬币，第i种硬币面值为V_i、颜色为C_i。每种硬币都有无穷多个。有Q组询问，每组询问给定一个数S，要你选出一些硬币使得它们的和为S，求最多能用的不同颜色种数。如果无解输出-1。

$1 \le N \le 30, 1 \le V_i, Q \le 2 \times 10^5, 1 \le S \le 10^{18}$

【题解】

如果只需要判断是否存在一种选硬币的方案，那么就设面值最小的硬币的面值是m，就在$\bmod m$意义下DP：f_i表示能拼成的最小的$\bmod m = i$的面值。那么只要$S \ge f_i$就存在。

现在要最大化颜色种数。那就再加一维，$f_{i,j}$表示用了i种颜色可以拼成的最小

的$\bmod m = j$的面值。

时间复杂度：$O(N^2V)$，空间复杂度：$O(NV)$

1.53 Room Corner

【来源】

CodeChef FEB13 ROC

【大意】

给定一张$N \times M$的方格地图，地图中是一个由墙围成的房间。保证房间内的格子是四联通的并且如果一条横的线段的两个端点都在房间内的话整条线段都在房间内。房间内每个90°的角落的空位上站着一个用大写字母表示的小孩，并且每个小孩只会站在角落。现在他们要玩一个游戏。游戏中每个小孩必须贴着墙走，并且每步只能往上下左右中一个方向走一格。每次在墙上相邻的两个小孩都站在相应的角落上时可以开始一次交换。不同小孩之间的交换可以同时进行。两个小孩可以在格子上相遇，也可以在格子的边界上相遇。

现在有T次询问，每次给出两个小孩，问从给定的初始状态开始以最优策略进行游戏需要多少时间才能使这两个小孩相遇。

$N, M \leq 2500,\ T \leq 10000$

【题解】

首先我们需要把地图转化为环。这个可以模拟贴着墙走一圈。注意判断一个小孩在当前这个角落。

然后对于一组询问，这两个小孩可以在环上选择两条路径之一走。最终他们会在某段区间的中点处相遇，并且时间是中点到他们出发点距离的最大值。因此只要

把环拉成两倍然后分别二分找最接近整段路径中点的点即可。

时间复杂度：$O(NM + T\log(NM))$，空间复杂度：$O(NM)$

1.54 Observing the Tree

【来源】

CodeChef FEB13 QUERY

【大意】

给定一棵N个点的树，支持以下操作：

- 1. 给u到v路径加上一个等差数列。
- 2. 询问x到y路径上所有点权值和。
- 3. 让整棵树回到第i次修改后的状态。

数据范围：$N, M \leq 10^5$

【题解】

此题与FIBTREE一题类似。

对这颗树树链剖分。

对于操作1，可以用线段树打标记实现，标记就是把这段区间加上一个首项为多少、公差为多少的等差数列。这个标记显然是可以合并的。那么就可以用树链剖分维护这个操作。

对于操作2，同样只要树链剖分即可。

对于操作4，只需把树链剖分的线段树可持久化。

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

1.55 A New Door

【来源】

CodeChef JAN13 ANDOOR

【大意】

平面上给定N个圆，求这些圆并的周长在一个矩形区域内的长度。

T组数据。

$1 \leq N, T \leq 1000, \sum N \leq 5000$

【题解】

可以对于每个圆，求出不被其他圆所覆盖且在矩形范围内的圆周部分。这个只要求出对应区间然后离散一下扫一遍。注意精度问题。

时间复杂度：$O(\sum N \times N \log N)$，空间复杂度：$O(N)$

1.56 Cucumber Boy and Cucumber Girl

【来源】

CodeChef JAN13 CUCUMBER

【大意】

给定B个$N \times N$的矩阵A_i，对于数对(a,b) $(a < b)$，定义$N \times N$的矩阵B满足$B_{i,j} = \sum_{k=1}^{N} A_{a,i,k} \times A_{b,j,k}$，同时定义一个1 N的排列P是好的当且仅当至少存在一个i使得B_{i,P_i}是奇数，定义数对(a,b)是好的当且仅当好的排列有奇数个。

问有多少个好数对。

$1 \le N \le 60, 2 \le B \le 8000$

【题解】

由B的定义我们想到$B = A_a \times A_b^T$。由矩阵上的排列我们想到了行列式。由奇偶性我们想到了mod2域。如果我们定义C满足$C_{i,j} = B_{i,j} + 1 \mod 2$，那么一个好的排列对$\det(C)$的贡献是0，一个不好的排列的贡献是+1或−1。当$N \ge 2$时，总的排列数$N!$为偶数，那么数对$(a,b)$是好的等价于$\det(C) \neq 0$。由于$C_{i,j} = (\sum_{k=1}^{N} (A_{a,i,k} \times A_{b,j,k} + 1)) \bmod 2$，那么可以在每个$A_i$后面加上全1的一列，于是$C = A_a \times A_b^T$（模2意义下）。由Binet-Cauchy定理，令$A_{i,j}$为A_i删去第j列后的矩阵，那么$\det(C) = \sum_{i=1}^{N+1} \det(A_{a,i}) \det(A_{b,i})$。因此只需求出所有的$\det(A_{i,j})$即可。我们可以对$A_i$消元，由于初等变换对行列式没有影响，所以消元对答案没有影响。如果A_i不满秩，那么所有$\det(A_{i,j})$全为0。否则一定可以把A_i消成$N \times N$的单位矩阵再插入一列数的形式。设消元之后的矩阵为D，插入的那列数是第k列，那么$\det(A_{i,j}) = D_{j,k} (j \ge k)$，$\det(A_{i,k}) = 1$。所有的操作都可以用bitset优化。

时间复杂度：$O(B \times \frac{N^3}{\omega} + B^2 \times \frac{N}{\omega})$，空间复杂度：$O(\frac{N^2}{\omega} + B \times \frac{N}{\omega})$

1.57 Different Trips

【来源】

CodeChef DEC12 DIFTRIP

【大意】

给定一棵N个点的树，1号点为根。求有多少不同的从某个点A到它的祖先B的路径。两条路径相同当且仅当两条路径长度相同且对于每个i都有路径上第i个点的度数相同。

$N \leq 10^5$

【题解】

此题类似ZJOI2015 day1 诸神眷顾的幻想乡一题。在给定的树上建广义后缀自动机（或后缀数组）统计不同字串个数即可。

时间复杂度：$O(N \log N)$，空间复杂度：$O(N)$

1.58 Quasi-Polynomial Sum

【来源】

CodeChef DEC12 QPOLYSUM

【大意】

给定模数M。已知有一个次数为d的多项式$P(x)$，给定$P(0), P(1), \ldots, P(d) \pmod M$以及一个数$Q$，求$\sum_{i=0}^{n-1} P(i)Q^i \mod M$。

$n \leq 10^{100000}$, $d \leq 20000$, $0 \leq Q < M \leq 10^{18}$

M与$2, 3, \ldots, d+14$互质。

【题解】

前置技能：模域下利用多项式在0到d处的值$O(d)$求任意一点处的值（拉格朗日

插值）。

特殊处理$Q = 0,\ Q = 1$的情况。$Q = 0$时答案就是$P(0)$。$Q = 1$时令$S(n) = \sum_{i=0}^{n-1} P(i)$，易知$S(n)$为$d+1$次多项式。我们只要算出$S(0) \dots S(d+1)$，就可以算出答案$S(n)$。

当Q不是0或1时，令

$$G(n) = \sum_{i=0}^{n-1} P(i)Q^i = Q^n F(n) - F(0)$$

其中$F(n)$是次数为d的多项式。这个可以用数学归纳法得到。

$$G(n+1) - G(n) = P(n)Q^n = Q^{n+1}F(n+1) - Q^n F(n)$$

即

$$F(n) = QF(n+1) - P(n)$$

这样，我们可以把$F(0), F(2), \dots, F(d)$表示成关于$F(d+1)$的一次函数。

由于$F(x)$是一个次数为d的多项式，就有

$$\sum_{i=0}^{d+1} (-1)^i \binom{d+1}{i} F(i) = 0$$

这是个关于$F(d+1)$的一次方程，可以解出$F(d+1)$的值。

然而我们在解方程时用到了除法，但M不一定是质数，不一定有逆元。考虑方程中$F(d+1)$的系数：

$$\sum_{i=0}^{d+1} (-1)^i \binom{d+1}{i} Q^{d+1-i} = (Q-1)^{d+1}$$

因此解方程时涉及$(Q-1)$的逆元。

设$M = m_1 m_2$满足$\gcd(m_2, Q-1) = 1$且m_2最大。只要分别算出模m_1, m_2的答案然后用中国剩余定理合并就好了。

对于模m_2的答案，只要用上述方法就好了。

考虑求模m_1的答案。必然存在一个最小的v使得$m_1 \mid (Q-1)^v$。设$m_1 = p_1^{a_1}p_2^{a_2}\dots p_k^{a_k}$，那么有$v \le \max\{a_1, a_2, \dots, a_k\}$。又因为$M$与$2, 3, \dots, d+14$互质，$M$可能的最小的质因子是17，而$17^{15} > 10^{18}$，所以$a_i \le 14$，所以$v \le 14$。所以$M$与$2, 3, \dots, d+v$互质。

然后，对于答案中的

$$P(i)Q^i = P(i)((Q-1)+1)^i = P(i)\sum_{j=0}^{i}\binom{i}{j}(Q-1)^j$$

当$j \ge v$时，$(Q-1)^j \equiv 0 \pmod{m_1}$，可以忽略。答案就是

$$\sum_{i=0}^{n-1}P(i)Q^i = \sum_{j=0}^{v-1}(Q-1)^j\sum_{i=j}^{d+1}P(j)\binom{i}{j}$$

$\binom{i}{j}$是关于i的j次多项式，那么$P(i)\binom{i}{j}$是不超过$d+v-1$次的多项式，所以$\sum_{i=0}^{n-1}P(i)Q^i$是关于n的不超过$d+v$次的多项式，设为$s(n)$。我们可以算出$s(0), s(1), \dots, s(d+v)$。由于$m_2$与$2, 3, \dots, d+v$互质，我们就可以算出$s(n)$了。

时间复杂度：$O(d+\log n)$，空间复杂度：$O(d)$

1.59 Arithmetic Progressions

【来源】

CodeChef NOV12 COUNTARI

【大意】

给定一个长为N的数组A，求满足$1 \le i < j < k \le N$且$A_j - A_i = A_k - A_j$的三元组(i, j, k)的个数。

$3 \le N \le 100000, 1 \le A_i \le 30000$

时限：CC上3s

【题解】

由$A_j - A_i = A_k - A_j$可知，$A_i + A_k = 2A_j$。这让我们想到了卷积：如果对于每个j，我们分别统计出j左右两边每种值出现的次数，再求这两个数组的卷积，取$2A_j$位置的值即可。但我们不可能对每个j都做一遍FFT。因此，我们想到折中的办法：分块。把数组分为很多大小为S的块。对于i, j, k均不在同一块的情况，在每块时做一遍FFT；对于i, j, k至少有2个在同一块的情况，比如j, k在同一块，只需统计出这一块之前各种值出现的次数，然后暴力枚举j, k即可。

设$M = \max A_i$，时间复杂度：$O(\frac{N}{S} M \log M + NS)$，当$S = \sqrt{M \log M}$时有最优时间复杂度$O(M\sqrt{M \log M})$，空间复杂度：$O(\frac{NM}{S})$，可以优化到$O(N + M)$

1.60 Martial Arts

【来源】

CodeChef NOV12 MARTARTS

【大意】

有两支各N个人的队伍比赛。每个人都必须只和对方队伍中的某一个人打一次。如果让左边第i个人和右边第j个人打，左边可以获得$A_{i,j}$的分数，右边可以获得$B_{i,j}$的分数。现在左边队长可以安排比赛。但右边队长在左边队长宣布比赛方案之后可以选择取消某一场比赛。两边队长都想要优先最大化(我方得分-对方得分)，其次最大化我方得分。求左边队长最优能得到怎样结果，即(左边得分，右边得分)。

$N \leq 100$

【题解】

为了方便，我们定义$w_{i,j} = A_{i,j} - B_{i,j}$。先只考虑最大化分差（即$w$之和），不考虑最大化得分。

先假设右边不能取消比赛。这是个二分图最大权匹配问题。可以用KM算法解决。

考虑对于某个匹配方案，右边会删去哪场比赛。右边会删去权值最大且为正的边。因此，我们可以把边按权值从小到大排序，然后依次加入每条边。初始时所有边在图上的边权为$-\infty$。在加入某条边时，如果权值为正，则假设这条边是被删去的边，要强制匹配这条边，先把这条边在图上的边权改为∞，求一下最大匹配，更新答案，然后再改为真实的值。

所以我们要维护：修改一条边的边权，询问最大权匹配。这个也可以用KM算法来做。考虑修改$w_{i,j}$。我们把i所在的匹配去掉。然后修改i的顶标$llab_i$为$\max_j (w_{i,j} - rlab_j)$，再用KM的方法不停地为$i$找增广路即可。

时间复杂度：$O(N^4)$，空间复杂度：$O(N^2)$

1.61 Max Circumference

【来源】

CodeChef OCT12 MAXCIR

【大意】

给定一个三角形ABC，以及N个操作。第i个操作可以把点A平移一个向量。你最多可以用K个操作，这些操作影响叠加，同一个操作不能重复使用。ABC三点可以共线或重合。

最大化三角形ABC的周长，答案的绝对误差不能超过10^{-12}。

$K \leq N \leq 500$, A, B, C的坐标绝对值不超过10^9, 平移向量每一维绝对值不超过10^6

【题解】

目标函数不是线性的，因此很难有有效的方法。所以我们想办法把目标函数转化成线性的。有一个性质：存在一个向量$\vec{d}$使得目标函数的最优处就是$f(\vec{v})$（称为$\vec{v}$的权）的最优处，其中$f(\vec{v}) = \vec{v} \cdot \vec{d}$（$\cdot$表示点积）。证明：目标函数小于等于最优答案的区域形成了一个椭圆，而最优答案在椭圆边界的某个点上。那么所有可行解都在那一点处的切线的一侧，切线的法向量即为一个合法的$\vec{d}$。

考虑对于某个$\vec{d}$如何计算答案。我们只要把所有向量按权排个序，取最大的K个正权即可。

考虑如何枚举可行的d。对答案有影响的只有向量两两之间的权的大小关系以及权的正负性。所以只要对于每两个向量$\vec{a}, \vec{b}$算出两个相反的作为边界的向量$\vec{c}, -\vec{c}$，对于$\vec{c}$所在直线同侧的d，$\vec{a}, \vec{b}$的权大小关系相同。再对于每个向量求出两个相反的与其垂直的向量$\vec{c}, -\vec{c}$。把所有的作为分界线的$\vec{c}$按极角排序，只要枚举分界线中每一块区域中随便一个向量即可。这样我们就得到了$O(N^3 \log N)$的算法。

考虑优化这个算法。我们发现每次排序是不必要的。只要调整在分界线上大小关系发生变化的两个向量即可。然后再在最大的K个里二分正负性。这样就做到了$O(N^2 \log N)$。由于atan2函数的精度问题，极角排序应用叉积判。一块区域中的向量只要把分界线的两个向量相加就好了。由于答案的精度要求很高，所以开根要用这样的做法：

设$\sqrt{S} = I + D$，其中$I = \lfloor \sqrt{S} \rfloor$。则可以得到$D = \frac{S - I^2}{I + \sqrt{S}}$，其中分母中的$\sqrt{S}$可以直接用math库中的sqrt函数。把$\sqrt{S}$放到分母中后，由于$I > 1$时$(\frac{1}{x})' < 1$，这样就提高了精度。

时间复杂度：$O(N^2 \log N)$，空间复杂度：$O(N^2)$

1.62 Knight Moving

【来源】

CodeChef SEPT12 KNGHTMOV

【大意】

在一个上下左右无线延伸的棋盘上，$(0,0)$处有一个骑士。如果现在骑士的坐标为(x,y)，那么下一步你可以把他移动到$(x+A_X,y+A_Y)$或$(x+B_X,y+B_Y)$。同时棋盘上有K个位置是障碍，不能走进去。问有多少种方案把骑士移动到目标(X,Y)，对10^9+7取模。如果有无穷多种方案输出-1。

$K\leq 15$，其他数的绝对值不超过$d=500$。

【题解】

如果向量$A=(A_X,A_Y)$和$B=(B_X,B_Y)$线性无关，那么平面上每个位置都可以被唯一表示为$aA+bB$。如果没有障碍物，答案就是$\binom{a+b}{a}$。有障碍物的话容斥一下就好了。组合数用阶乘和阶乘逆元预处理。

如果A,B线性相关，也就是说它们在一条直线上，那么就可以把问题缩减到一维，在数轴上跳来跳去。如果可以在起点到终点的一条路径上找到环的话就有无数种方案了，如果没有环，那么同时由起点可达和可达终点的那些点构成的图就是一张拓扑图。所以关键就是判是否是拓扑图。这个只要DFS或BFS判一下就好了。由于要构成环最坏需要走到LCM处，所以有影响的数轴范围只有$[-d^2,d^2]$。

时间复杂度：$O(2^K K+d^2)$，空间复杂度：$O(d^2)$

1.63 Annual Parade

【来源】

CodeChef SEPT12 PARADE

【大意】

给订一张N个点M条边的有向图，每条边有一个费用。有K次询问，每次给定一个数C，要你安排一些路径，使得总费用最少。总费用有三部分：

- 沿边移动的总费用。如果有多条路径经过某条边，那么这条边的费用相应地计算多次。
- 如果某条路径的起点与终点不同，那么产生C的费用。
- 如果某个点没有被任何路径经过，那么产生C的费用。

$N \leq 250,\ M \leq 30000,\ K \leq 10000,\ 1 \leq$ 边权$, C \leq 10000$，无自环，但可能有重边。

【题解】

首先简化问题。我们在图中任意两点之间（包括自己和自己）连费用为C的边。然后问题等价于求一些回路覆盖所有点使得所有回路经过的边的总费用最小。为什么可以这样转化呢？考虑一个回路可能的情况：

- 一个点的自环。这种情况代表这个点没有被任何路径经过。
- 不经过任何C边。这种情况代表原来的回路。
- 经过一些C边。我们可以把那些C边断掉，每一段代表一条首尾不同的路径。由于C边和断成的段数一样，所以转化后费用是一样的。

考虑如何处理一个C。由于不同回路可能重复经过同一段路径，我们用Folyd求出任意两点之间的最短路，这样回路就可以“跨过”一段重复的路径，问题就变成了用一些环来覆盖每个点。这是个经典问题，由于每个点入度和出度都是1，可以建一张二分图，左边N个点表示出度，右边N个点表示入度。对于每条边把相应的出度和入度连起来。只要跑个二分图最小费用匹配即可。这个可以费用流。

考虑如何处理多个C。我们先不连C边，并认为没有匹配的点之间配满了C边。费用流增广时，每增加1对匹配，就减少C的代价。由于费用流每次增广的费用是单调不减的，所以对于某个C只要做到最后一次费用不超过C的增广即可。这个只要对每个询问二分。

时间复杂度：$O(N^3 + costflow(N, N^2) + K \log N)$，空间复杂度：$O(N^2)$

1.64 A Game of Thrones

【来源】

CodeChef AUG12 GTHRONES

【大意】

给定一个由正整数组成的多重集合，其中有N种不同的数，第i种数是u_i，共c_i个。现有两人博弈。首先第一个人在第一个回合中选一个数作为初始值x，然后两人轮流操作，每一回合需要删除一个x元素，然后再选择一个y，满足$x|y$且$\frac{y}{x}$为质数或者$y|x$且$\frac{x}{y}$为质数，然后把x赋为y。不能操作者输。求谁有必胜策略。如果第一个人必胜，那么还要求最小的可以使他必胜的初始值。

数据范围：$N \leq 500$, $u_i \leq 10^{18}$, $c_i \leq 10^9$

【题解】

判断x是否可以到y只要用Miller-Rabin判一下质数即可。我们在所有可以互相到达的点之间连边。那么第二个人必胜当且仅当这张图有完备匹配。由于这张图是二分图（可以根据质因子个数奇偶性分成两部分），所以只要跑一下最大流即可。至于初始值要最小，因为某一个值可以作为必胜的初始值当且仅当存在一个最大匹配使其不在最大匹配中，体现在网络流上就是可以退流。所以只要尝试退流即可。可以不用真的退流，只要在残量网络上判一下与源或汇的连通性。

时间复杂度：$O(N^2 \log u + maxflow(N, N \log u))$，空间复杂度：$O(N \log u)$

1.65 Dynamic GCD

【来源】

CodeChef JULY12 DGCD

【大意】

给定一棵N个点的带点权的树（第i个点的点权初始为v_i），有Q次操作：

- F u v：询问u, v两点路径上所有节点权值的最大公约数。
- C u v d：给u, v两点路径上所有点的点权加上d。

$N, Q \leq 5 \times 10^4,\ 1 \leq v_i \leq 10^4,\ 0 \leq d \leq 10^4$

【题解】

首先树链剖分，这样就变成序列问题。由gcd的线性性，即$\gcd(a, b, c) = \gcd(a, b - a, c - b)$，可以对序列作差，维护作差后的值的单点修改、区间gcd，同时维护原来的点权区间加、单点询问。这些都可以用线段树解决。

时间复杂度：$O(Q\log^2 N\log(Qd))$，空间复杂度：$O(N)$

1.66 Cool Numbers

【来源】

CodeChef JUNE12 COOLNUM

【大意】

一个K位十进制数N，数位上数字依次为$X_1, X_2, \ldots, X_K$。如果N这个数存在1到3 个数位上的数之和为S，且$(X_1 + X_2 + \cdots + X_K - S)^S$是$N$的倍数，则称$N$为cool number。定义$LC(N)$和$RC(N)$分别为小于等于$N$的最大的cool number和大于$N$的最小的cool number。有$T$次询问，每次给出$N$，求$LC(N)$和$RC(N)$。

$T \le 10^5$, $N \le 10^{1000}$, 所有N的长度之和 $\le 4 \times 10^6$

【题解】

如果$(X_1 + X_2 + \cdots + X_K - S) = 0$，那么$N$中只有不超过3个非零数位。这个可以$O(K)$求出前后最近的答案。

否则$(X_1 + X_2 + \ldots X_K - S)^S \ge N$，这种情况下$N$的位数不会太多：$(9K)^{27} \ge 10^{K-1}$，经计算$K$最多是80左右。因此所有数位之和可能的值最多只有720。我们可以枚举$(X_1 + X_2 + \cdots + X_K - S)$的值，再枚举其约数。可以发现这种情况的数的个数只有32740个，因此我们只要预处理出这些数、排个序，然后在上面二分即可。

时间复杂度：$O($所有N的长度之和 $\times \log 32740)$，空间复杂度：$O($$N$的长度 $+$ $32740)$

1.67 Expected Maximum Matching

【来源】

CodeChef JUNE12 MATCH

【大意】

有一张二分图，左边点数为N，右边点数为M。在左边第i个点和右边第j个点之间有边的概率为$p_{i,j}$。求最大匹配的期望值。

$N \leq 5,\ M \leq 100$

【题解】

由于N很小，考虑DP：让右边点一个一个加进来，记录当前左边点的每个子集是否存在一个匹配使得这个子集中的所有点都被匹配，用0或1表示。总共有$2^N \leq 32$个子集，所以状态数就是$2^{2^N} \leq 2^{32}$，可以用一个unsigned存下。但实际状态远远没有这么多：考虑Hall定理，如果某个集合能完全匹配，那么这个集合的子集也都应该能完全匹配。通过搜索得到所有可达的状态只有406个。转移只要$O(2^N$枚举当前加入的这个点与左边的点的连边情况再考虑当前状态中每个1添加了这些边之后能不能扩展。转移可以预处理。

时间复杂度：$O(M \times 2^N \times$状态数$)$，空间复杂度：$O(2^N \times$状态数$)$

1.68 Little Elephant and Boxes

【来源】

CodeChef MAY12 LEBOXES

【大意】

有n个盒子，第i个盒子里有$P_i\%$的概率是V_i块钱，有$(100-P_i)\%$的概率是一颗钻石。

有m个物品可以买，第j个物品需要花费C_j块钱和D_j颗钻石。

现在你先打开所有的盒子，得到盒子里的东西，然后去买物品。你会买尽可能多的物品。求期望能买到的物品数。

$n, m, D_j \leq 30,\ V_i, C_j \leq 10^7$

【题解】

好题!

购买物品让我们想到背包。但此题钱太多，所以我们换一种DP：$f_{i,j}$表示用了i颗钻石、得到j个物品所需最少的钱数。我们预处理这个DP。

接下来我们对盒子的情况用meet in the middle。设一个M。枚举前M个盒子所有可能的情况，得到这种情况下获得的钻石数和钱数，并把状态存下来：对于每种钻石数开一个数组，记录所有可能的钱数以及这种情况的概率。排个序处理出概率的前缀和。再对于后面$n-M$个盒子枚举所有可能的情况，对于每种情况枚举最终总共得到的钻石数以及可以买到的物品数，由于当前情况的钱数、钻石数以及最终至少需要的钱数皆已知，减一减可以推得前M个盒子需要的钻石数、钱数，这样只要在之前排序过的数组上二分找到相应位置统计出概率和即可。

时间复杂度：$O(n^3 + 2^M M + 2^{n-M} nmM)$，当$M$取$\log_2 \sqrt{2^n nm}$约为20左右时即可通过全部数据，空间复杂度：$O(n^2 + 2^M)$

1.69 Selling Tickets

【来源】

CodeChef MAY12 TICKETS

【大意】

有N道菜和M个人，这些人中有一些人要来参加晚宴。晚宴时每道菜会被分配给一个人。已知第i个人只要吃到第a_i或b_i道菜就会开心，否则不开心。现在求最大的x使得任意x人来参加晚宴都存在一种分配方案使每个人都开心。

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

【题解】

好题!

由于要检验一个答案使得所有方案都能使所有人开心很困难，我们转化为求最小的一个不能使所有人开心的方案。我们把问题抽象为有N个点M条边的无向图，每条边要对应其中一个端点，要求一个边数最少的子图使得不能满足每条边的需求。有一些观察：

1. 这样的子图一定是连通的，因为一个不连通的不能满足每条边需求的方案必然有一个更小的子连通块不能满足每条边的需求。

2. 这样的连通块的边数一定恰好为点数+1。如果连通块的边数小于等于点数的话，要么是树，要么是环套树，这样的结构一定可以满足每条边的需求。而边数大于点数显然一定不能满足每条边的需求。

3. 考虑边数为点数+1的连通块可能的情况。我们可以先构出一棵树作为骨架，再在上面添加两条边形成两个环。这两个环要么重叠要么不重叠。如果重叠那么这个结构就是一个8字形的结构再外加一些树。如果不重叠那么就是两个环中间用一条链相连再外加一些树。由于我们要最小的方案，可以把外面的那些树都去掉。

对于8字形结构，我们可以看成是连接两个点的三条不重叠的路径。其实重叠了也没关系，因为重叠之后必然可以找到更优的结构。因此只要枚举这两个点然后BFS一下即可。

对于两个环中间连一条链的结构，我们可以枚举一个中间链上的点，以这个点为根搞出一棵BFS树，我们只要用最小的两个由一个环加上到根的路径组成的结构大小之和更新答案。每条非树边都对应一个环。由于BFS，每个点到根的路径就是到根的最短路径之一。因此一个由多条非树边对应的环必然可以不成为最优的结构。但是每个点到根的最短路径不是唯一的，那么一个由多条非树边对应的环有没有可能成为次优的结构呢？不可能。因为一条最短路径图上的非树边加上其他非树边必然不优。

时间复杂度：$O(N^2M)$，空间复杂度：$O(N+M)$

1.70 Substrings on a Tree

【来源】

CodeChef APRIL12 TSUBSTR

【大意】

给定一棵N个点的有根树，每个点有一个小写字母。一个字符串被这棵树表示了当且仅当它可以被表示为一个点往后代移动路径上经过的点的字母连起来得到的字符串。求有多少字符串被这棵树表示了。

之后有Q次询问，每次给定一个26个字母的字典序，求这棵树表示的串中字典序第K小的串。

$N \leq 2.5 \times 10^5$, $Q \leq 5 \times 10^4$

【题解】

对这棵树建后缀自动机。再在后缀自动机上DP出每个状态有多少后继状态。第一问只要用根上的信息即可。第二问只要按照字典序在自动机上逐位确定即可。

时间复杂度：$O(N+Q)$，空间复杂度：$O(N)$

1.71 Find a special connected block

【来源】

CodeChef APRIL12 CONNECT

【大意】

给定一个$n \times m$的网格图，每个格子有一个$[-1, n \times m]$中的颜色和一个代价。求一个不包含颜色为-1的格子且至少包含K种不同的正数颜色的四连通块使得其中所有格子的权值和最小。

$n, m \leq 15,\ K \leq 7,\ 1 \leq$ 权值 $\leq 10^5$

【题解】

如果颜色种数$= K$，就是经典的斯坦纳树问题。令$f_{s,u}$表示在点u处且当前连通块状态为s，只要枚举子集以及最短路转移就行了。

考虑原问题，我们可以随机生成一个$n \times m$种颜色到K种颜色的映射，并对映射之后的网格图做斯坦纳树。那么这样得到的答案一定不比最优解优，且得到的正好是最优解的概率为$\frac{k!}{k^k}$。于是只要随机T次即有很大概率得到最优解。实际$T = 400$就可以通过了。

时间复杂度：$O(Tnm(3^K + \log(nm)2^K))$，空间复杂度：$O(nm2^K)$

1.72 Evil Book

【来源】

CodeChef MARCH12 EVILBOOK

【大意】

有N个厨师，对于第i个厨师可以付出C_i点努力打败他并得到M_i点魔力，同时M_i变为0。在拥有魔力不少于X时可以使用帮助，即通过消耗X点魔力使某个厨师的C_i和M_i都乘以$\frac{1}{3}$。初始时拥有魔力为0。求收集至少666点魔力所需最少的总努力。如果不可能，输出"impossible"。

数据范围：

$1 \leq T \leq 5$

$1 \leq N \leq 10$

$10 \leq X < 666$

$0 \leq C_i \leq 10^9$

$0 \leq M_i \leq 10^9$

时间限制：CodeChef上的3s

【题解】

首先应该观察到一些基本的结论。

打败某个厨师获得的魔力必然大于对他使用帮助所消耗的魔力。不然魔力值反而减小了，没有必要。因此打败一个厨师后魔力必然增加。所以我们对某个厨师使用k次帮助应满足$\frac{M_i}{3^k} > kX$。

我们可以只在打败某个厨师之前对他使用帮助。由于使用帮助需要消耗魔力，因此在之前对他使用帮助只会使得更长时间范围内的魔力值降低，而这显然是不优

的。

同时，令k_i为最小的非负整数使得$\frac{M_i}{3^{k_i}} < 666$。设当前最多能用$can$次帮助，那么当我们有足够的魔力时，$k \geq \min(can, k_i - 1)$。因为如果$k \leq \min(can, k_i - 1)$，在打败这个厨师之后必然能完成任务。为了使得付出的努力C_i尽可能小，k应该尽可能大。

这样我们就可以使用搜索。每次选择一个未被挑战的厨师，枚举其需要使用的帮助次数k，进入下一层搜索。可以记录每个厨师使用的帮助次数的状态，如果之前已经搜索过这种状态了那么就不再搜索（因为这样的答案是相同的）。

设r为某个厨师的k最多可能的取值种数。经计算可知$r \leq 4$。这样状态数就是$r^N = 1048576$。但实际上的状态数可以更少。比如，当$M_i \leq \frac{666}{3}$ 时最多只有3种状态，因为最多只能使用2次帮助；而当$M_i > \frac{666}{3}$时，最多只会有2个恰好使用了k_i次帮助的厨师，因为如果有3个的话就一定能达到666点魔力了。因此状态数不会多于$3^N + N \times 3^{N-1} + \frac{N(N-1)}{2} \times 3^{N-2} = 551124$。但实际状态数远远小于这个值。

我们还可以使用其他的剪枝，比如加上剩下的所有魔力都不能达到666点，比如按最优的投入产出比值计算都不比当前答案优。

时间复杂度：$O(T \times r \times N \times the\ number\ of\ states)$，空间复杂度：$O(r^N)$

1.73 Ciel and Earthquake

【来源】

CodeChef MARCH12 CIELQUAK

【大意】

给定一张$R \times C$的网格图，第r行第c列的点记为(r, c)，(r_1, c_1)和(r_2, c_2)间有边相连当且仅当$|r_1 - r_2| + |c_1 - c_2| = 1$。每条边有$p$的概率毁坏。求$(1, 1)$与$(R, C)$连通的概率。你的答案绝对误差不能超过$10^{-6}$。

数据范围：

$1 \leq T \leq 50$

$1 \leq R \leq 8$

$1 \leq C \leq 10^{18}$

$0.1 \leq p \leq 1$

p小数点后最多四位。

时间限制：CodeChef上的8s

【题解】

首先我们考虑C较小的情况，比如我们假设$C \leq 50$. 然后就可以DP了。

我们可以用类似插头DP的方法。处理到某一点(r,c)时，DP状态应记录点$(1,c),(2,c),\ldots,(r-2,c),(r-1,c),(r,c-1),(r+1,c-1),...,(R-1,c-1),(R,c-1)$以及点$(1,1)$之间的连通情况。记录连通性通常使用最小表示法。我们可以预处理从初始状态向外搜索出的所有可达的合法状态以及这些状态之间的转移。$(1,1)$与其他点都不连通的状态是非法的。经验证，当$R=8$时状态数（设为S）为3432。

设$X(R,C,p)$为长为R、宽为C且概率为p时的答案。通过打表等手段可以发现，当R固定时，随着C趋向于无穷大，$\frac{X(R,C+1,p)}{X(R,C,p)}$趋向于一个常数。由此，我们可以采用这样的方法估计答案：取一个足够大的C'，DP计算到C'处，把$X(R,C',p) \times (\frac{X(R,C'+1,p)}{X(R,C',p)})^{C-C'}$作为估计值。经过实验验证，当$C'$取40到50时精度就够了。

时间复杂度：$O(T \times SRC')$，空间复杂度：$O(SN)$

1.74 Find a Subsequence

【来源】

CodeChef FEB12 FINDSEQ

【大意】

给定一个长度为N的数组A和一个12345的排列B，求一个长度为5的A的子序列，使得其中元素两两不同，且的相对顺序和B一样。即求一组$i_0 < i_1 < i_2 < i_3 < i_4$，使得对于每个$k$，$A_{i_k}$是这5个数中第$B_k$小的。

T组数据。

$T \leq 20,\ N \leq 1000,\ |A_i| \leq 10^9$。

【题解】

由于我们只关心数的相对大小，所以先把A离散化。

令前缀最小值$pre_min[i][j] = \min_{1 \leq k \leq i,\ a[i] \geq j} A_k$，这个只要$O(N^2)$预处理一下就好了。同理可以预处理出前缀最大值、后缀最小值、后缀最大值。还要预处理出前缀和。

枚举i_1, i_3。这样，其他元素的位置范围都确定了，但数值范围还可能有交集。

设$S, M, L \in \{0, 2, 4\},\ b_S < b_M < b_L$，接着我们贪心。$S$和$L$中必有一个是0或4。比如$S = 0$，那么我们自然要$A_{i_S}$尽可能小。这个只要利用前面预处理的前缀最小值就好了。接下来，剩下的两个元素M和L是“对称”的：一方面要让A_{i_M}尽可能小，一方面要让A_{i_L}尽可能大。那么就可以确定M和L中是0或4的那个元素。至于剩下一个元素，只要利用前面预处理的前缀和求一下是否存在就好了。

时间复杂度：$O(T \times N^2)$，空间复杂度：$O(N^2)$。

1.75 Flight Distance

【来源】

CodeChef FEB12 FLYDIST

【大意】

给定一张N个点、M条边的无向图，每条边初始时有正整数长度。现在可以修改每条边，第i条边的边权加上d_i（d_i可正可负，为有理数，修改了之后边权仍为正数），同时付出$|d_i|$的代价。最重要使得在改变之后，不存在一条边的长度比同样从这条边的起点到终点的其他路径要长。求最小代价，以最简分数形式输出。

$N \leq 10,\ M \leq \frac{N(N-1)}{2},\ 1 \leq$ 初始边权 ≤ 20

【题解】

这是一个线性规划问题。对于点u, v之间的边（如果存在的话），设变量$d^+_{u,v}, d^-_{u,v}$表示边权$W_{u,v}$要加上$d^+_{u,v}$、减去$d^-_{u,v}$。设$g_{i,j}$表示从点i到j的最短路。由最短路的性质有限制：

$$g_{i,j} \leq W_{i,k} + d^+_{i,k} - d^-_{i,k} + g_{k,j}$$

还要强制$g_{i,i} = 0,\ g_{u,v} = w_{u,v}$(如果存在边$(u, v)$)

由于要求边权为正数，应该有相应的限制。不过我们注意到，没有必要把一条边的边权降到1一下。证明：假设有些边降到了1一下，那么把这些边都变成1对其他大于1的边显然不会有影响，对于这些边本身也不会有问题。因此有限制：

$$W_{u,v} + d^+_{u,v} - d^-_{u,v} \geq 1$$

目标是要最小化$\sum d^+_{u,v} + d^-_{u,v}$

然而我们在做单纯形之前发现，变量全为0不一定是合法的初始解。如何构造合法的初始解？我们发现所有边权全相同（比如都是1）是一种合法的初始解。考虑对一些变量进行替换。令

$$g'_{i,j} = g_{i,j} - \text{点}i\text{到}j\text{最少需要经过的边数}$$

令

$$d'^-_{u,v} = W_{u,v} - 1 - d^-_{u,v}$$

这样$W_{u,v}+d^+_{u,v}-d^-_{u,v}\geq 1$的限制就不需要了，但是要加上$d^-_{u,v}\leq W_{u,v}-1$的限制。

最后由于分数运算较慢，计算时使用浮点数，最后枚举答案的分母。

时间复杂度：$O($单纯形$($变量数：N^2, 限制数：$N^3))$，空间复杂度：$O(N^5)$

1.76 Card Shuffle

【来源】

CodeChef JAN12 CARDSHUF

【大意】

桌上有一叠牌，从顶到底编号为1到N。现在重复M次这样的操作：

- 从顶上拿A张牌。
- 再从顶上拿另外B张牌。
- 把那A张牌放回到顶上。
- 从顶上拿C张牌。
- 把那B张牌一张一张地放回顶上。
- 最后，把那C张牌放回顶上。

输出最终牌的顺序。

$N, M\leq 10^5$

【题解】

用平衡树打翻转标记模拟即可。

时间复杂度：$O(M \log N)$，空间复杂度：$O(N)$

1.77 Misinterpretation 2

【来源】

CodeChef JAN12 MISINT2

【大意】

有多少个长度在$[L, R]$的由小写字母构成的字符串使得该字符串经过一次变换之后与原串相同。一次变换指的是把字符串所有偶数位置的字符（下标从1开始）按原来的顺序提到最前面，剩下的字符还是按原来的顺序排列。输出答案 $\bmod 10^9 + 7$。

多组数据。

$T \leq 5,\ 1 \leq L \leq R \leq 10^{10},\ R - L \leq 50000$

【题解】

考虑如何统计长度为X的这样的字符串个数。设该置换的环数为$G(X)$，那么答案就是$26^{G(X)}$。如果X是奇数，易发现第X个字符自己单独构成一个环，所以此时$G(X) = G(X-1) + 1$。因此只要考虑X是偶数的情况。设$F(t)$为变换后第t个字符原来的位置。可以发现$F(t) \equiv 2t \pmod{X+1}$。那么对于一个$i$，$i, 2i, 4i, \ldots, i \pmod{X+1}$这些数都是在一个环里的。因为$X+1$为奇数，所以$\gcd(2i, X+1) = \gcd(i, X+1)$，因此环长就是2在模$\frac{X+1}{\gcd(i, X+1)}$（设为$p$）域下的阶，记为$\mathrm{ord}(p)$。也就是说，$G(X) = \sum_{p \mid X+1,\ p \neq 1} \frac{\varphi(p)}{\mathrm{ord}(p)}$。因此，我们需要把$X+1$分解质因数。这个

只要预处理出1到$\sqrt{R+1}$的所有质数，然后枚举质数再枚举其在区间内的倍数就好了。这一步复杂度是$O(\sqrt{R}+D\log D)$。枚举p的同时可以求出$\varphi(p)$。考虑怎么求$\mathrm{ord}(p)$。有一个定理：如果$\gcd(x,y)=1$，那么$\mathrm{ord}(xy)=\mathrm{lcm}(\mathrm{ord}(x),\mathrm{ord}(y))$。因此只要求质因子幂的阶，也就是形如$\mathrm{ord}(x^y)$（$x$是质数）的值，就好了。为此，我们要把$\varphi(x^y)$分解质因数，然后枚举质因子一个一个试除（复杂度$O(\log^2 x^y)$）。对于$\leq \sqrt{R+1}$的$x^y$的阶可以预处理，而对于$X+1$的大质因子$x$（最多只有一个）的$phi(x)=x-1$我们用枚举小质数试除的方法或者Pollard-rho 算法来分解。前者的复杂度是$O(\frac{\sqrt{R}}{\log R})$（质数密度）。

时间复杂度：$O(\sqrt{R}\log^2 R+TD(\log D+\frac{\sqrt{R}}{\log R}+\log^2 R+\sqrt{R}))$，空间复杂度：$O(\sqrt{R})$

1.78 Short II

【来源】

CodeChef DEC11 SHORT2

【大意】

给定质数p，问有多少对$a,b(a,b>p)$满足$(a-p)(b-p)\mid ab$。

t组数据。

$1<p<10^{12},\ 1\leq t\leq 5$

【题解】

问题等价于求有多少对$a,b(a,b\geq 1)$满足$ab\mid(a+p)(b+p)$，等价于$ab\mid p(a+b+p)$。分3种情况讨论：

1) $p\mid a,\ p\mid b$

2) $p \nmid a,\ p \mid b$ ($p \mid a,\ p \nmid b$的情况一样，答案只要×2就好了)

3) $p \nmid a,\ p \nmid b$

对于1)：$\frac{a}{p}\frac{b}{p} \mid \frac{a}{p}+\frac{b}{p}+1$。易知满足这个条件的$(\frac{a}{p}, \frac{b}{p})$只有5对：$(1,1),(1,2),(2,1),(2,3),(3,2)$。

对于2)：令$b = pb'$。那么$ab' \mid a + pb' + p$，也就是说$a + pb' + p = kab'$。可得$b' = \frac{a+p}{ak-p}$。令$d = ak - p$。那么应该满足$d \mid a + p$且$a \mid d + p$。由于$\gcd(d, d + p) = \gcd(a, a + p) = 1$，上述条件等价于$ad \mid (a + p)(d + p)$，而这正是3)的条件。因此2)和3)的解一一对应，只要求出3)的答案就好了。

对于3)：由于a, b均不含p，$ab \mid a+b+p$。$ab \le a+b+p \Leftrightarrow (a-1)(b-1) \le p+1$。如果$a = b$，易知只有一个解$(1, 1)$。否则，假设$a < b$。可知$a < w = 1 + \sqrt{p+1}$。然后，从$a + b + p = kab$中可得$b = \frac{a+p}{ak-1}$，令$d = ak - 1$。现在我们要求满足下列条件的$(a, d)$：

0. $p \nmid a$

1. $d \mid a + p$

2. $a \mid d + 1$

3. $b = \frac{a+p}{d} > a$

我们分两种情况讨论：$d \le \sqrt{p + w}$和$b \le d$ ($b \le \sqrt{p + w}$)。

枚举所有$d \le \sqrt{p + w}$。由条件1可知$a \equiv -p \pmod d$。由条件2可知$1 \le a \le d + 1$。可见a最多只有两个可能的值。

枚举所有$b \le \sqrt{p + w}$。同样由条件1可知$a \equiv -p \pmod d$。由条件3可知$a < b$。又由于$b < d$，可见a最多只有一个可能的值。

时间复杂度：$O(\sqrt{p})$，空间复杂度：$O(1)$

1.79 Hypertrees

【来源】

CodeChef DEC11 HYPER

【大意】

一个3-超图类似于一个普通的图，只不过其中的边都连接3个点。

一个3-超树是一个去掉任意一条边以后都不连通的连通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。

【题解】

由于数据范围很小，可以本地算好答案后打表上交。

考虑一个点双连通的3-超树（没有割点）中的一条边连接的三个点。这三个点中有且仅有一个点是只属于这条边的（否则会违反上述性质）。将所有这种点移除，对相应的边进行处理后这样图就会变成一个点双连通的普通图。可以暴力或DP求出n个点m条边的普通点双连通的三无图（无重边无自环的无向图）的个数。然后就要对于n计算出这个大小的点双连通3-超树的个数，只要枚举作为骨架的普通点双连通图，然后把剩下的点配到骨架的边上，只要暴力枚举或者DP处理一下重边就好了。最后，我们再用暴力或DP把这些点双连通3-超树拼接起来就好了。

打表时间复杂度：$O(2^{\left(\lfloor\frac{N}{2}\rfloor\right)}N^2$或$N^3)$，打表空间复杂度：$O(N^2)$

1.80 Luckdays

【来源】

CodeChef NOV11 LUCKYDAY

【大意】

定义数列S：$S_1 = A,\ S_2 = B,\ S_i = (X \times S_{i-1} + Y \times S_{i-2} + Z) \mod P$。

给定A, B, X, Y, Z, P, C，有Q组询问，每次给定L_i, R_i，询问有多少k满足$L_i \leq k \leq R_i$且$S_k = C$。

$P \leq 10007$且为质数。$Q \leq 2 \times 10^4$。$L_i \leq R_i \leq 10^{18}$。

【题解】

如果$Y = 0$，那么这个数列后面的循环节是$O(P)$的，只要暴力找一下就好了。

如果$Y \neq 0$，我们把递推关系表示成矩阵的形式：初始矩阵$Init = [A, B, 1]^T$，转移矩阵$Trans$。注意到转移矩阵是可逆的。我们可以用矩阵$BSGS$来求出循环节长度（满足$Init \times Trans^x = Init$的最小的正整数$x$）以及对于每个0到$P-1$的数求出满足$Init \times Trans^x = [C, \text{这个数}, 1]^T$的最小的非负整数$x$，这样就得到了在一个循环节内所有答案。考虑到这样的数有P个，易知最优复杂度为$O(P^{1.5})$。对于每个询问，我们只要根据循环节在预处理出来的所有在一个循环节内的答案上二分即可。

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

1.81 Colored Domino Tilings and Cutsontest

【来源】

CodeChef NOV11 DOMNOCUT

【大意】

给定一个$N \times M$的矩形棋盘。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。一个格子与另一个格子相邻当且仅当他们有公共边。每个字母对应一种颜色。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。也就是说，这条直线不能穿过一对有相同颜色的相邻格子。

你要构造一个棋盘覆盖染色使得，棋盘的割的数量最少。如果有多个解，你要使得使用的颜色最少。如果使用了K种颜色，那么只能用前K个小写字母。如果还有多个解，任意输出一组解。如果无解，输出"IMPOSSIBLE"。

T组数据。

$T \leq 3000,\ 1 \leq N, M \leq 500,\ \sum N \times M \leq 2000000$

【题解】

无解当且仅当$N \times M$为奇数。首先，最多只需要3种颜色。然后我们可以构造出5×6和6×8的棋盘使得割数为0。然后我们可以通过在这两种棋盘内插入若干列1×2的骨牌以及若干行2×1的骨牌从而扩展到任意N, M更大的棋盘。至于染色方案，可以在扩展的过程中构造。而当N, M较小时我们可以打表。

时间复杂度：$O(\sum N \times M)$，空间复杂度：$O(N \times M)$

1.82 The Baking Business

【来源】

CodeChef OCT11 BAKE

【大意】

有Q个操作，每个操作可能是送货或者询问，每个操作如下描述：（其中方括号表示可以省略）

对于送货：

I 产品编号[.大小编号] 省份编号[.城市编号[.区域编号]] 性别 年龄 送货量

表示给相应的顾客送货。送货量不超过100。如果有省略，表示信息未知。

对于询问：

Q 产品编号[.大小编号] 省份编号[.城市编号[.区域编号]] 性别 起始年龄[-结束年龄]

表示询问当前已经送给相应的条件下顾客的总货量。如果没有结束年龄，则只询问等于起始年龄的顾客，否则询问在起始年龄到结束年龄之间的顾客。如果有省略或产品编号或身份编号为-1表示相应的条件没有限制，询问所有这样的顾客。如果有的顾客该信息未知，则忽略。

上述所有的编号都从0开始。有10种产品，每种产品3种大小，10个省份，每个省份20个城市，每个城市5个区域，性别为M或F，年龄介于1到90之间。

$Q \leq 100000$

【题解】

开一个7维数组记录所有货量。对于每一维添加一项表示没有限制。对于每个

订单在相应的2^5个位置更新。询问只要暴力询问相应的年龄区间即可。

时间复杂度：$O(Q(2^5+90))$，空间复杂度：$O(11\times 4\times 11\times 21\times 6\times 2\times 90)$

1.83 Sine Partition Function

【来源】

CodeChef OCT11 PARSIN

【大意】

给定整数M,N和实数X，求

$$f(M,N,X)=\sum_{k_1+k_2+\cdots+k_M=N}\sin(k_1X)\sin(k_2X)\ldots\sin(k_MX)$$

其中$k_1,\ldots,k_M$为非负整数。

T组数据。

$1\le T\le 10, 1\le M\le 30, 1\le N\le 10^9, 0\le X\le 6.28<2\pi$，$X$小数点后最多两位。答案的绝对或相对误差不得超过$10^{-1}$。你可以放心地假定答案的数量级最大是$10^{300}$。

【题解】

考虑DP。定义

$$\begin{aligned} f_{n,m} &= \sum_{k_1+k_2+\cdots+k_m=n}\prod_{i=1}^{m}\sin(k_iX)\\ g_{n,m} &= \sum_{k_1+k_2+\cdots+k_m=n}\cos(k_mX)\prod_{i=1}^{m-1}\sin(k_iX)\end{aligned}$$

那么考虑从$n-1$推到n，可以使$m+1$，也可以沿用原来的m。根据和角公式有递推式：

$$\begin{aligned} f_{n,m} &= (f_{n-1,m-1}+g_{n-1,m})\sin(X)+f_{n-1,m}\cos(X) \\ g_{n,m} &= (f_{n-1,m-1}+g_{n-1,m})\cos(X)-f_{n-1,m}\sin(X) \end{aligned}$$

由于N很大，可以用矩阵乘法快速幂加速DP，已可以通过本题。一个复杂度更好的做法是类似常系数线性递推的$O(M^2\log N)$做法倍增。

时间复杂度：$O(T\times M^2\log N)$，空间复杂度：$O(M)$

1.84 Short

【来源】

CodeChef SEPT11 SHORT

【大意】

给定n和k，求满足$n<a,b<k$且$ab-n$可以被$(a-n)(b-n)$整除的整数对(a,b)的个数。

$0\le n\le 10^5,\ n<k\le 10^{18}$

【题解】

当$n=0$时，答案显然是$(k-1)^2$。因此下文中默认$n>0$。

$(ab-n)$可以被$(a-n)(b-n)$整除也就是存在一个整数p使得$ab-n=p(a-n)(b-n)$。由于$n>0$，可以发现$p>1$。

由于是二元组，考虑枚举a，把b表示成关于a的形式：$b=\frac{n(p(a-n)-1)}{p(a-n)-a}=n+\frac{n(a-1)}{p(a-n)-a}$。我们可以通过枚举$n(a-1)$的约数$d$来枚举$b$，当然要保证$p=\frac{d+a}{a-n}$为整数。

为了减少枚举量，我们只枚举$a \le b$的情况。由于$p \ge 2$，$d + a \ge 2(a - n)$，可以写成$d \ge a - 2n$。如果$a > 2n$，那么$b \le n + \frac{n(a-1)}{a-2n}$，由于$a \le b$，$a \le n + \frac{n(a-1)}{a-2n}$，整理得$a^2 - 4na + (n + 2n^2) \le 0$，这是一个关于$a$的二次函数，可以得到$a \le 2n + \sqrt{2n^2 - n}$。由此可见$a$可能的取值范围是$O(n)$的。这个限制也可以通过打表得到。

所以只要枚举a和d做一下就好了。

然而对于一些情况这个做法仍然较慢。不难发现当a变大时有效的约数d的个数越来越少。因此，在某些情况下我们可以用枚举p来代替枚举d。由$a \le b$，也就是$a \le n + \frac{n(a-1)}{p(a-n)-a}$，可以得到$p \le \frac{a^2-n}{(a-n)^2}$。

时间复杂度：$O(n \times$ 约数个数$(n) \times \ln n)$，空间复杂度：$O(n)$

1.85 Counting Hexagons

【来源】

CodeChef SEPT11 CNTHEX

【大意】

给定N, L, X, K，统计满足以下条件的无序六元组个数：

- ①每个数都是$[1, N]$ 中的整数。
- ②如果选取这六个数作为六根木棒的长度，这六根木棒能围成面积为正的六边形。
- ③最大的数大于等于L。
- ④其他数都小于等于X。
- ⑤某个数在这六个数中出现的次数不能超过K。

答案 $\bmod 10^9+7$。

$N \le 10^9,\ X < L \le N,\ N-L \le 100,\ K \le 5$

【题解】

②等价于其他所有数之和大于最大的数。

枚举最大的数。那么只要考虑剩下的5个数。首先补集转化，求出所有满足④⑤的个数，再减去满足④⑤而不满足②的个数。求满足④⑤的个数只要预处理所有可能的相同状态然后组合数算一下就好了。

用数位DP求满足④⑤而不满足②的个数。DP状态要记录这些信息：

- DP到了第几位（$O(\log N)$）
- 当前5个数之和是否是M的前缀（$O(2)$）
- 当前5个数中最大数是否是X的前缀（$O(2)$）
- 从低位来的进位（$O(5)$）
- 这5个数排序后的相同状态（$O(16)$）

转移时枚举这一位上5个数是什么。

易知在二进制下数位DP最优。

时间复杂度：$O((N-L)\times\log N\times 2\times 2\times 5\times 16\times 2^5)$，空间复杂度：$O(2\times 2\times 5\times 16)$

1.86 Shortest Circuit Evaluation

【来源】

CodeChef AUG11 SHORTCIR

【大意】

给定一个布尔表达式，可以调换运算顺序来减少运算量，比如A or B中如果已知A为true，则无需计算B。表达式中每个变量只出现一次。给定每个变量为true的概率，求最优计算顺序下期望要调用多少次变量的值。

$T(T \leq 50)$组数据，每个表达式长度不超过30000，变量数不超过1000，变量名不超过5个字母。

【题解】

首先是一个表达式的问题，用栈或者递归或者表达式树分解出字符串中的元素。比如用递归，递归函数返回二元组(w,p)表示期望要用w次才能确定这个表达式，并且这个表达式值为true的概率为p。那么对于串起来的and或or运算，考虑怎么算最优：设有运算顺序相邻的两个字表达式$(w_1,p_1),(w_2,p_2)$，它们不论怎么排列都对前后其他元素没有影响，应该满足交换之后不优。按原来的排列总期望是$w_1+p_1w_2$，换一换之后总期望是$w_2+p_2w_1$，对答案贡献的变化是$(w_2+p_2w_1)-(w_1+p_1w_2)=(1-p_1)w_2-(1-p_2)w_1$。所以只要用叉积排序，或者说按斜率排序。

时间复杂度：O(表达式长度+变量数$\log$变量数)，空间复杂度：O(表达式长度)

1.87 Something About Divisors

【来源】

CodeChef AUG11 DIVISORS

【大意】

给定正整数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$D(N < D \leq B)$能整除NX。

T组数据。

$T \leq 40, B \leq 10^{12}, X \leq 60$

【题解】

由于D的范围很大，而X的范围很小，所以考虑枚举$K = \frac{NX}{D}$，这样，$D = \frac{NX}{K}$。

$$\begin{cases} N < D \leq B \\ D|NX \end{cases} \Leftrightarrow \begin{cases} K < X \\ N \leq \frac{BK}{X} \\ K|NX \end{cases} \Leftrightarrow \begin{cases} K < X \\ N \leq \frac{BK}{X} \\ \frac{K}{\gcd(K,X)}|N \end{cases}$$

令$C(K) = \frac{K}{\gcd(K,X)}$, $M = \frac{N}{C(K)}$, $Limit = \frac{BK}{X \cdot C(K)}$,

$$\Leftrightarrow \begin{cases} K < X \\ M \leq Limit \end{cases}$$

考虑枚举K，对于某个K，满足上述条件的N的个数就是$Limit$。但是这样会算重。满足存在J且满足下述条件的N会算重：

$$\begin{cases} N \leq \frac{BK}{X} \\ K|NX \\ J|NX \\ K < J < X \end{cases} \Leftrightarrow \begin{cases} M \leq Limit \\ J|C(K) \cdot MX \\ K < J < X \end{cases}$$

令$D(J,K) = \frac{J}{\gcd(J,C(K),X)}$,

$$\Leftrightarrow \begin{cases} M \leq Limit \\ D(J,K)|M \\ K < J < X \end{cases}$$

根据容斥原理，枚举到这个K对答案的贡献就是

$$\sum_{\{J_1,J_2,\ldots,J_t\} \subseteq \{K+1,K+2,\ldots,X-1\}} (-1)^t \left\lfloor \frac{Limit}{\operatorname{lcm}(D(J_1,K), D(J_2,K), \ldots, D(J_t,K))} \right\rfloor$$

这样，我们就得到了一个$O(2^X)$的算法。显然不能满足时限。因此要加一些优化：

1. 如果$D(J_1,K)|D(J_2,K)$，去除$D(J_2,K)$。
2. 合并lcm相同的状态。
3. 如果当前状态lcm $> Limit$就直接放弃这个状态。
4. 把X相同的询问放到一起，按最大的B做。
5. 每次扩展状态时，

$$\text{lcm}' = \text{lcm} \cdot \frac{D(J,K)}{\gcd(\text{lcm} \bmod D(J,K), D(J,K)}$$

其中$\frac{D(J,K)}{\gcd(\text{lcm} \bmod D(J,K),D(J,K)}$的取值只有不超过10种。由于开始时的状态是有序的，只要对每种取值开一个队列，把新的状态放到对应的队列中，最后合并这些有序的队列即可。

时间复杂度：O(玄学)，空间复杂度：O(玄学)

1.88 Billboards

【来源】

CodeChef JULY11 BB

【大意】

有一个长度为N的数组，数组中的每个数都是0。现在要把其中一些0变成1，使得任意连续M个数里至少有K个1。求有多少种方案使得1的总个数最少。

T组数据。

$T \le 300,\ 1 \le K \le M \le 500,\ M \le N \le 10^9$

【题解】

先考虑1的总个数最少是多少。把数组分成一段一段长度为M的块，最后剩下一个长度为$N \bmod M$的残余的块。每个完整的块里都需要至少K个1。我们发现，这样贪心是能使1的个数最少的：在每个完整的块的末尾放上K个1，在残余的块的末尾放上$\max(0, N \bmod M - (M - K))$个1。

接下来我们根据$N \bmod M$和$M - K$的大小关系分类讨论。

如果$N \bmod M \leq M - K$，那么实际上就是每个块前$N \bmod M$个位置都是0，剩下的位置里放K个1。

如果$N \bmod M > M - K$，那么实际上就是每个完整的块末尾$M - N \bmod M$个位置都是1，前面$N \bmod M$个位置里放$K - (M - N \bmod M)$个1。

这样，我们就把问题转化为在m段区间的每段区间内要选k个1使得题中所述条件（即任意连续M个数里至少有K个1）成立。

设第j块里第i个1的位置时$pos_{i,j}$，在原数组上的位置时$ori_{i,j}$。可以发现，对于某个i，$pos_{i,1}, pos_{i,2}, \ldots, pos_{i,m}$是单调不增的。证明：如果存在一个$j$使得$pos_{i,j} < pos_{i,j+1}$，那么在原数组上从$ori_{i,j} + 1$到$ori_{i,j} + M$这段区间内1的个数不足$K$个。

于是，这个问题就变成了在一个$k \times m$的矩阵里填数，每个数的范围是$[1, r]$，对于矩阵里每个位置满足这个位置上的数大于它上面的数且小于等于它左边的数。这就是一个半标准杨氏矩阵，方案数计算公式：

$$dim = \prod_{(i,j)} \frac{r + j - i}{hook(i, j)}$$

其中$hook(i, j)$表示(i, j)的钩长，即为矩阵中同一行或同一列中横坐标或纵坐标大于等于(i, j)的位置数。

可以发现，把答案展开后分子分母形式相同，且当$m > r - k$时，分子分母中就有明显的重复部分。我们可以约去重复部分，这样分子分母都只有$k \times \min(m, r - k)$个数，只要预处理出不超过$2M$的阶乘及阶乘逆元，即可做到每组数据$O(M^2)$的时间复杂度。

时间复杂度：$O(T \times M^2)$，空间复杂度：$O(M)$。

1.89 Trial of Doom

【来源】

CodeChef JULY11 YALOP

【大意】

在一个$n \times m$的格子图构成的房间里，每个格子可以是蓝色或红色。现在你在$(1,1)$，出口在(n,m)，你最终需要到达终点并使得所有格子都是蓝色的（到了终点还可以再走）。每步你可以移动到八个相邻的格子，每当离开一个格子，这个格子和它周围的四个格子会改变颜色。现在给定房间内有那些格子是红色（设有k个红色格子），问是否能离开这个房间。

T组数据。

$T \leq 50,\ 1 \leq n,m \leq 10^9, \min(n,m) < 40, 0 \leq k \leq \min(m \times n, 10000)$

【题解】

假定$m \leq n$，$m > n$只要换一换就好了。

注意到如果访问到了某个格子2次，就相当于没有访问。

接下来，我们分两种情况讨论：$m \geq 2$和$m = 1$。

如果$m \geq 2$，那么我可以达到任意的格子访问状态，也就是说，我可以想访问哪个格子就访问哪个格子，与我走的路径无关。

证明：在2×2的4个格子里，格子之间可以两两直接到达。那么，我们可以构造下列两种移动：

1. 不改变格子的访问状态从一个格子移动到另一个格子。这个只要借助第三个格子就好了。

2. 改变一个格子的访问状态。这个只要先走到另一个格子，再利用第1种移动

走回来就好了。

这样，我们就可以随便走，从而达到任意的格子访问状态。

所以只要看是否存在一种格子的访问状态使得所有格子都是蓝的。

注意到如果确定了第一列格子的访问状态，那么只要一列一列推下去，后面所有格子的访问状态都是确定的。最后一列可能还有剩余的红色格子。

同样的，如果只有某个位置上的一个红色格子，我们也可以通过访问它右边的格子来把红色格子的影响推到后面两列去，一直可以推到最后一列。可以利用循环节来加速这个推的过程。

所以，我们可以把所有红色格子转化到最后一列格子的状态上去。然后找出第一列的每个格子的访问状态对最后一列格子的影响，只要看第一列的这些格子的影响能不能线性表示最后一列的状态。这个问题只要用高斯消元或线性基就好了。

如果$m = 1$，那么我只能从起点一路走到终点。当然中间可以往回走到某个位置再回来，这样中间经过的那些位置都抵消了，就相当于把两个位置的访问状态都改变一下。因此我可以达到所有每个位置访问状态之和的奇偶性与m-1相同的访问状态。

我们可以通过访问相邻的两个格子来使得一个红色格子移动3格。这样，我们可以把所有红色格子移到前三个格子里而不改变所需操作数的奇偶性。对于第三个格子，只要访问一下第一个和第二个格子，就可以消掉这个红色或产生一个红色。对于前两个格子，如果这两个格子都是红色，就可以通过1次访问第一个格子来消掉，当然这种情况下访问状态之和的奇偶性必须是奇数。如果都是蓝色，就只要是偶数。不然就没有办法消掉了。但是可以利用倒数第三个格子产生一个变化然后把这个变化三格三格移到前面来从而改变前两个格子的状态。

时间复杂度：$O(T\times(k+\min(n,m)\times$循环节长度$+\frac{n^2}{\omega}))$，空间复杂度：$O(\min(n,m)\times$循环节长度$+\frac{n}{\omega})$。

1.90 Attack of the Clones

【来源】

CodeChef JUNE11 CLONES

【大意】

我们称一个形为$f : A \to B$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$f(0, \ldots, 0) = 0$;
- P 是1-保留值函数集合: 满足$f(1, \ldots, 1) = 1$;
- D 是自对偶函数集合: 满足$!f(x_1, \ldots, x_n) = f(!x_1, \ldots, !x_n)$;
- A 是仿射函数集合: 满足如果$f(a_1, \ldots, c, \ldots, a_n) = f(a_1, \ldots, d, \ldots, a_n)$ 则$f(b_1, \ldots, c, \ldots, b_n) = f(b_1, \ldots, d, \ldots, b_n)$ 的函数，在这里c 和d 都在某个位置i，并且这个对于任意$i, a_1, \ldots, a_n, b_1, \ldots b_n, c, d$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。例如$n = 2$，有8个在Z集合的函数，4个在Z和P交集的函数，8个在A补集的函数等等。

给出q个集合表达式，对于每个表达式输出答案 $\bmod 1000003$。

$n, q \leq 100$, 表达式长度 ≤ 100

【题解】

如果我们能对于每种集合的组合算出这些集合的交的大小，就可以知道恰好在

其中一些集合而一定不在其他集合中的函数个数，只要用二进制压一下然后表达式求值就好了。

考虑D的性质：当所有$x_1 = 0$的函数值都确定了之后，所有$x_1 = 1$的函数值也都确定了。

考虑A的性质：对于变量x_i，如果存在$x_1, \ldots, x_n$使得x_i的改变会引起函数值的改变，那么对于任意的$x_1, \ldots, x_n$只要x_i改变，函数值就会改变。也就是说，x_i始终直接影响函数值。而另一种情况是x_i始终不影响函数值。因此，所有变量可以分为2类：对函数值有影响和没有影响。而函数值初始时可能为任意值。

下面是每种集合的组合下的交的大小：

- 全集: 2^{2^n}
- Z: 2^{2^n-1}
- P: 2^{2^n-1}
- D: $2^{2^{n-1}}$
- A: 2^{n+1}
- ZP: 2^{2^n-2}
- ZD: $2^{2^{n-1}-1}$
- ZA: 2^n
- PD: $2^{2^{n-1}-1}$
- PA: 2^n
- DA: 2^n
- PDA: 2^{n-1}
- ZDA: 2^{n-1}

- ZPA: $\sum_{k \bmod 2=0} \binom{n}{k}$
- ZPD: $2^{2^{n-1}-1}$
- $ZPDA$: 2^{n-1}

时间复杂度：$O(n+$表达式长度$)$，空间复杂度：$O(n+$表达式长度$)$

1.91 Minesweeper Reversed

【来源】

CodeChef JUNE11 MINESREV

【大意】

给定一个$R \times C$的扫雷棋盘，告诉你所有雷的位置。开始时所有方块都是打开的，你可以通过一次点击关闭一个方块（可以关闭含雷的方块）。在你关闭一个方块后，所有在普通扫雷游戏中可能和这个方块同时打开的格子都会被关闭。求最少需要点击多少次才能关闭所有方块。

多组数据。

$T \leq 50,\ 1 \leq R, C \leq 50$

【题解】

这里所有的相邻都是指8连通。

首先每一个雷都需要一次点击。我们把不含雷的格子分为两类，第一类是与雷相邻的，第二类不相邻。第二类格子构成了一些连通块。如果关闭了一个第一类格子，所有与其相邻的连通块以及和这些连通块相邻的第一类格子都会被关闭。

可以发现第一类格子至多与2个连通块相邻。如果不与任何连通块相邻，它本身也要用一次点击。我们把第二类格子的连通块看成点，与2个连通块相邻的第一类格子看成无向边，那么答案就是这张图的点数减去最大匹配数，用带花树即可。

时间复杂度：$O(T \times (RC)^2)$，空间复杂度：$O(RC)$

Chapter 2

challenge型试题

2.1 Sereja and Permutation

【来源】

CodeChef APRIL14 SEAPERM

【大意】

有一个含N个整数的数组A'。现在要重新排列数组的元素，即用一个置换p来得到一个新的数组A，使得$A_i = A'_{p_i}$。　　令函数$f(A,i) = S - A_i - A_{i+1} - \cdots - A_j$，其中$j$是使得$A_i + A_{i+1} + \cdots + A_j \leq S$的最大整数，若$A_i > S$，则$f(A,i) = S$。

请找到一个置换p，使得$\frac{f(A,1)+f(A,2)+\cdots+f(A,k)}{k}$的值尽可能的小。

【题解】

考虑乱搞。我们有两个指针i, top。从1到k枚举i。根据i以及之后所有已经放好的数之和，当能够在末端top处添加新的数时，不停地添加最大的能添加的数。最后把所有剩下的数随便放到最后就好了。

这个做法对于$S \geq \sum A'_i$的数据当然是最优的。对于其他的数据，我发现对于小数据这种做法不太优，随着数据的增大越来越优，甚至远超该题数据制作者的答案。

另一种方案是，假设已经确定了第$k+1$到n个数是什么，我们通过排列剩下的数，先最小化$f(A,k)$，再最小化$f(A,k-1)$……每次只要找能放下的最大的数就好了，这个只要用树状数组就可以做到$N \log N$。

对于这两种做法，我们都可以通过多次随机初始的一些值来找到尽可能优的解。

时间复杂度：$O(P \times N^2)$或$O(P \times N \log N)$（其中P为随机次数），空间复杂度：$O(N)$

2.2 To challenge or not

【来源】

CodeChef JUNE13 CHAORNOT

【大意】

给定由m个不同的数组成的集合B，你需要选择B的一个尽可能大的子集A使得A中不存在三个数构成等差数列。

数据生成方式为随机一个整数$L \in [10^4, 10^5]$和实数$p \in [0.1, 0.9]$，对于$[0, L)$中的每个数，有p的概率在B中。

设$N = |A|$。

【题解】

最简单的贪心：把B中的数从小到大排序，然后从小到大选，能选就选。判断能不能选是$O(m + N^2)$的。由于数据均为随机，N很小，因此可以很快得到答案。这样就已经可以得到很高的分数了。为了得到更优的答案，还可以按一定概率随机决定要不要选择一个可以选的数，比如可以取$w = \frac{1}{70}$为不选的概率。但实际上对所有数据都取相同的值不是最适合的。我们可以通过打表、调参来找出各个数据范围下最优的w值。

时间复杂度：$O(m + N^2)$，空间复杂度：$O(m + N)$。

2.3 Deleting numbers

【来源】

CodeChef AUG13 DELNMS

【大意】

有一个初始长度为n的序列。每次可以选择v,t两个数，令k为最大的满足$v+kt\leq n$的数，如果$a_v=a_{v+t}=a_{v+2t}=\cdots=a_{v+kt}$，就可以从序列中删掉这些数，删掉之后空缺由后面的元素递补，序列长度减小。

求一种删掉所有数的方案，步数越少越好。

$n,a_i\leq 10^5$

【题解】

这题有很多搞法。最简单的：你可以从后往前一个一个地删。加强一下：你可以统计出出现次数最多的数，然后先一个个删去其他的数，再一口气删去这些数。然而这样还是不够优。

考虑每次找能删的最长的操作。枚举t，那么最后那个元素一定在$[n-t+1,n]$范围里，只要枚举最后那个元素的位置往前找就好了。而序列中的每个位置都对应了$[n-t+1,n]$中的一个位置。因此找一次的复杂度为$O(N^2)$。考虑到最坏情况下要找N次，总复杂度就是$O(N^3)$。

然而这个复杂度显然无法接受。我的做法是为t设一个枚举的上限，然后倒着找能删的子序列，一旦找到了一个长度有点长（比如大于10）的方案，就直接删掉，然后继续找。最后把所有剩下的数一个个删掉。

一个更优的做法是，把序列分成若干长度为S的小段，从后往前一段一段地消。对于每个小段进行上述的$O(N^3)$的贪心策略。

时间复杂度：$O(\text{玄学})$或$O(nS^2)$，空间复杂度：$O(n)$

2.4 Kali and Devtas

【来源】

CodeChef DEC14 KALKI

【大意】

给定二维欧氏平面内的N个点，你需要返回这些点的一个生成树，使得C_i的最大值最小。

C_i的定义是：对于每个点，设在生成树中与其相邻的点中最远的点的距离为R，那么以该点为圆心，半径R以内的的点的C_i全部增加1（包括自身）。

T组数据。

$T \leq 100,\ 3 \leq N \leq 400,\ |x_i|, |y_i| \leq 2000$

坐标在平面中随机均匀生成。

【题解】

首先可以想到的是，构出完全图，边权定义为两点间的欧几里得距离，求出最小生成树。该做法在清澄上能拿到100分，但在CodeChef上practice中只有0.35分。

考虑用类似Prim的做法来求生成树。其他做法都和Prim一样，就是边权的定义不一样，并且会改变。定义f_u为点u以当前已连接的点中距离最远的那个点的距离为半径能覆盖到的点数。那么(u, v)的边权实时地定义为连上这条边以后会使得f_u和f_v增加多少。这样做能更优一些。

时间复杂度：$O(T \times N^2 \log N)$，空间复杂度：$O(N^2)$

2.5 Closest Points

【来源】

CodeChef JUNE12 CLOSEST

【大意】

给定三维空间中的n个点，有q次询问，每次给定一个点，询问距离这个点最近的点的编号。

$n = q = 50000$, 坐标绝对值 $\leq 10^9$

【题解】

这题是一个经典问题，可以当传统题来做。用kd-tree完美解决：从根往下递归每个点，如果当前点管理的长方体区域与询问点的最近距离都不比当前已知最优值优的话就直接退出，否则用这个点到询问点的距离更新答案，然后递归左右子树。注意距离的平方需要用unsigned long long来保存。注意卡一下常数，比如不要用指针来建树，比如用方差最大的那一维排序而不是交错排序。

时间复杂度：$O(n \log n + qn^{\frac{2}{3}})$，空间复杂度：$O(n)$

2.6 Similar Graphs

【来源】

CodeChef APRIL12 SIMGRAPH

【大意】

给定两个N个点的无向图。求一种给两个图的点重标号的方法使得这两个图的相似度最大。相似度定义为满足边(A, B)在两张图中都存在的边数。

T组数据。

$T = 5$, N为$[30, 75]$中随机的整数。

【题解】

我们可以固定第一张图的标号，只需处理第二张图的标号。

首先，容易想到的是爬山或模拟退火。每次尝试随机交换两个点的标号。如果交换后的相似度增加了就交换一下，或者减少了一点点，那么有一定概率接受。

但是，在当前方案已经很难改变时，每次随机选两个点很难选到能更优的调整方案。我们可以用枚举代替随机。可以找到一个更优的调整方案就调整，也可以找最优的那个调整方案来调整。

为了尽可能利用时间，我们需要卡时使得爬山尽可能多次。我们还可以对邻接矩阵压位来加速计算。

时间复杂度：$O(T \times$ 爬山次数 $\times$ 调整次数 $\times \frac{N^3}{\omega})$，空间复杂度：$O(N^2)$

2.7 Simultaneous Nim

【来源】

CodeChef SEPT12 SIMNIM

【大意】

给定n个异或和为0的数A_i，你需要把它们分成尽可能多组使得每一组内所有

数的异或和都为0。

令$M = 60$。

$n \leq 1000,\ A_i < 2^M$

【题解】

首先想到一个贪心的方法：把这些数一个一个加入线性基集合中，记录每个线性基由那些数异或得到，如果发现无法某个数加入线性基集合中那就可以根据bitset找到一组异或和为0的数，把这些数作为一组提取出来，剩下的数重复这个过程。

另一种方法是，寻找包含第一个数的一组。对于剩下的数，我们用01变量表示选或不选，可以建立方程组，那么我们的要求就是解中1的个数尽可能少。可以高斯消元。由于变量很多，但方程数很少，很有可能1的个数也很少，所以考虑从这些变量中多次随机选出一小部分变量做高斯消元，取1的个数最少的就好了。高斯消元也可以用bitset优化。

时间复杂度：$O(\frac{NM^2(M+N)}{\omega})$或$O(\frac{NTM^3}{\omega})$（其中$T$表示随机次数），空间复杂度：$O(\frac{MN}{\omega})$

2.8 The Great Plain

【来源】

CodeChef OCT11 LAND

【大意】

给定一个$n \times m$的方格图$A_{i,j}$，你需要在空格子里填上1到50之间的整数，有一些格子已经填好了。填完之后，任意相邻（有公共边）的两个格子u, v会产

生$2^{|A_u-A_v|}$的代价，其中A_u表示u中填的数，A_v表示v中填的数。你需要最小化所有代价之和。

T组数据。

$T \leq 10$, n, m为$[10, 100]$中随机生成的数。

【题解】

首先考虑一个贪心：不停地扫描方格图上的每个格子，把这个格子修改为与当前周围四个格子代价最小的数值，直到不能修改为止。初始值随便选，比如全是1。但是我们发现，有可能会出现到了一个不是最优的方案但是却不能继续修改的情况。于是我们考虑如何让初始解更优。我们可以让初始时每个空格填上与这个空格曼哈顿距离最近的已经填好的格子的值。这样初始解就大大优化了。但是还是可能出现有一部分不是最优但不能修改的情况。我们可以在每次不能修改时随机扰动一下，选几个点把它们的值加1或减1。还有一个优化是，与空格子产生的代价应该比与已经确定好的格子产生的代价估价的权重不同，因为空格子可以再调整而确定好的格子不能调整了。

时间复杂度：$O(T \times P \times nm)$（P为随机次数），空间复杂度：$O(nm)$

2.9 The Malaysian Flight Search

【来源】

CodeChef MAY14 ANUMFS

【大意】

这是一道交互题。你需要搜索马航的黑匣子。我们可以把地图简化为一个二维网格图。你需要确定黑匣子在哪个格点里。现在我们确定了一个搜索范围。这个

搜索范围是一个四连通块。给定四连通块边界上的所有n个点。有m架飞机，每架飞机有两个属性R和C，R为该飞机可以发射信号的半径，C为飞行单位里程所需付出的费用。注意这里的“半径”和“里程”都是指在曼哈顿距离意义下的。如果该飞机在某点处发射了一个信号，可以检测出黑匣子是否在与该点曼哈顿距离不超过R的范围内。初始时所有飞机都在坐标原点。每次，你可以派一架飞机飞到搜索范围内的某个点(x, y)，然后发射一个信号，检测是否存在黑匣子，并飞回原点。你需要调用一个函数来进行这个操作，函数会返回1或0表示是否检测到黑匣子，同时操作付出的费用就是$2C(|x| + |y|)$。保证一定存在$R = 0$的飞机。

请用尽可能小的费用找到黑匣子的位置。注意黑匣子可能不在搜索范围内，如果这样返回$(-1, -1)$。

数据范围：$1000 \le n \le 5000, 40 \le m \le 50$, $0 \le R \le 600$, $100 \le C \le 10000$，搜索范围在一个1000×1000的正方形内，并且搜索范围内的整点个数不超过2×10^5，原点到搜索范围内任意一点的距离不超过10^5。R, C在范围内随机。

【题解】

由于搜索范围内点数不多，可以用floodfill找出所有搜索范围内的点。一个简单的做法是，让$R = 0$的飞机一个一个点试过去。为了减少代价，我们可以把所有可能的点按到原点的曼哈顿距离从小到大排序。但这样显然远远不优。

我们可以让适当大的飞机尝试访问每个没有被确定的点。如果发现这次访问没有发现黑匣子，就把覆盖到的点全部标记为不可能，以后也不必找了。如果找到了，那么除了范围内的其他点都是不可能的，只要在这个范围内继续找就好了。我们再用更小一些的飞机到这个范围内找。这样不断重复，直到确定了为止。每次我们可以估计一下每个飞机可能要用多少费用，找估价最少的飞机。

时间复杂度：$O(m \times 1000 \times 1000)$，空间复杂度：$O(1000 \times 1000)$

2.10 Tom And Jerry

【来源】

CodeChef MARCH12 TOMJERRY

【大意】

Tom和Jerry（猫和老鼠）在一个$N \times M$方格地图上博弈。初始时，告诉你Jerry的位置，有一些格子是门，有一些格子是障碍。每一轮，Tom先操作：选择某个空位放上一个障碍；接着Jerry会这样移动：找到最短的一条通往一个门的路，沿路径的第一步移动（如果有多条这样的路那么按照某个上下左右的优先级）。现在Tom想围死Jerry，也就是使得Jerry无法到达任何一个门。你需要为Tom设计这样的方案。当然，放的障碍数越少越好。题目设计的评分方式会使得失败的方案（最终Jerry走到某个门）的得分远差于成功的方案的得分，同时成功的方案步数越少分数越好。

数据范围：$N, M \leq 50$

时间限制：2s

【题解】

为了方便，定义预计路径为当前情况根据Jerry的策略预计要走的路径。

首先必须注意到最重要的是要围死Jerry，因为失败的话分数就会非常差。其次才是最小化放的障碍数。

想法1：搜索

最简单的爆搜就是枚举放的位置，然后让Jerry走一步，然后再枚举放的位置……

但这样显然是不能承受的。我们要想办法让Tom更聪明，不是盲目地放障碍。

想法2：跟着Jerry放障碍

我们可以在Jerry四周放障碍。但这样往往效果不好，因为Jerry会溜走。所以最好留有余地，比较好的是在距离不超过2、3的范围内放障碍。但实际在比较空旷的地图上效果并不好。

想法3：在Jerry预计路径上放障碍

这样的障碍会改变Jerry的预计路径，从而拖延时间，甚至堵死Jerry。我们可以枚举当前Jerry的预计路径上的格子放上障碍。

想法4：把门堵死

对于门比较少的情况，把门堵死是很好地策略。这样，一个门最多只要4个障碍，还可以避免和Jerry直接周旋。实际实现时，我们要优先堵住当前最有可能的方向。我们可以优先填上Jerry当前预计路径的最后一个空格。

但对于门较多的情况，这样做效果就不好了：把每个门堵住就需要耗费很多的障碍，需要消耗很多时间，Jerry就更有机会在很多门里选择。

想法5：给每个格子一个估价值

我们可以给每个格子一个估价值，估计堵住这个格子对Jerry未来可能的路径的影响大小，再找一个估价值最大的格子堵上。估价值的设计要考虑很多方面：距离不能太近，要优先堵上当前最紧迫的方向，要堵在要塞处，等等。估价值的设计比较困难，只能作为一个大概的判断，不能处理很精确的问题，难以应对各种各样的环境。

想法6：设计陷阱伏击Jerry

我们仔细想就会发现，Jerry的策略并不是很高明：他只知道往当前最近的路上走，却没有考虑中途被堵死的危险性。与其想着如何把Jerry到门的路堵住，不如想办法引诱Jerry，直接把Jerry抓住。没错，这就是本题一个很重要的想法：利用地形设计一个陷阱，埋伏在Jerry的路上，等到Jerry进入陷阱后趁机围住Jerry。这种做法在地形空旷的情况下效果拔群，因为Jerry往往没走几步就进了圈套。

对于陷阱的设计，如果是空旷地带，没有什么有利地形，那么就可以设计这样的陷阱：

```
.##.
->..
.##.
```

其中→表示Jerry进入的方向。当Jerry从图中(2,2)到达图中(2,3)位置后，就立即在(2,4)位置放上障碍。此时Jerry往往会往回走，想逃出陷阱，从而回到(2,2)。此时只要在(2,1)处放上障碍，就可以把Jerry抓住了。

```
.##.
#*.#
.##.
```

构造一个陷阱需要在Jerry到来之前留有一定的时间放置障碍。为此，我们可以引入空余障碍的概念：如果Tom暂时不确定这一步的障碍放在哪里，那么就把这一步的障碍称为空余障碍。在不影响Jerry的预计路径的前提下，在后来的步骤中可以在有需要的地方放置空余障碍，也就是决定之前的那一步障碍放在哪里。由于不能影响Jerry的预计路径，所以当前这一步产生的空余障碍不能放在当前到这个空余障碍被决定放置之前所有Jerry的预计路径上。

利用空余障碍，当Jerry走到某个位置时，我们就可以在条件允许的情况下构造一个陷阱。

看起来要用代码实现这样的陷阱比较困难，而且这样的模式不依赖原有的地形。

为了让我们的陷阱聪明地利用原有地形，在要塞处放置障碍，从而减少所需的障碍数，我们可以用最小割。注意到一个陷阱必然是在Jerry的路径上找2个连续的空格使得当Jerry试图退出陷阱时留有余地。当我们找到这样合适的2格后，就找一个最小割分隔开这2格和其他所有的门。可以这样构图：

对于每个格子建2个点$in_{x,y}, out_{x,y}$。S表示源，T表示汇。连边：(u, v, w)表示点u到v有一条流量限制为w的边。

对于每个空格：$(in_{x,y}, out_{x,y}, 1)$

对于两个相邻的非障碍格子：$(out_{x1,y1}, in_{x2,y2}, \infty)$

对于2个特殊格子：$(S, out_{x,y}, \infty)$

对于每个门：$(in_{x,y}, T, \infty)$

在这张图上跑最大流。最后得到一组最小割只需从源在参量网络上DFS一下，所有跨越被DFS到的点和没被DFS到的点的边构成最小割。最小割必然在流量为1的边上，相应的格子就是我们的陷阱的组成部分。

放障碍的时候要考虑某些空余障碍不能放在某个位置上。对于一前一后2个对Jerry的预计路径有影响的障碍留到最后依次放置。

最终算法

综合以上想法，我们可以得到这样的算法：

首先，用搜索让Jerry先走几步，走的方向需要找出预计路径（可以用BFS求出）。然后让Tom利用想法2、3的方式枚举格子放障碍或者暂时不定这个障碍的位置，把它作为空余障碍留到以后需要时再确定放哪里。同时也可以尝试利用最近的2步设置一个陷阱把Jerry直接抓住。（想法4其实已经包含在这个算法里：先迫使Jerry改变路径拖延时间，再把门旁边剩下的那些缺口看成一个巨大的陷阱）由于搜索的状态太多，我们可以使用卡时、迭代加深等方法在有限的时间内得到尽可能优的解。

当然其他的策略还有很多，解法不唯一。