IOI 2016 中国国家集训队作业
CodeChef 试题泛做

绍兴一中 孙耀峰

目　录

Chapter 1

91 道非challenge型试题

1.1 Future of draughts

【试题来源】

Codechef AUG 2015 CLOWAY

【试题大意】

给定 T 张图，每张图为 N_i 个点 M_i 条边的无向图。

定义 $G(i)$ 为第 i 张图的邻接矩阵。 $G(l,r)$ 为第 l 到 r 张图的强积图的邻接矩阵。

进行 Q 次询问，每次询问给出三个数 l_i, r_i, k_i ，需要回答矩阵 $G(l_i, r_i)$ 所对应的图中，有多少长度不超过 k_i 的闭合回路。最后答案模 10^9+7 。

关于图的强积可以参见 strong products of graphs。

数据范围：$T, N_i \leq 50$，$Q \leq 2 * 10^5$，$k_i \leq 10^4$。

【算法介绍】

这是一道比较麻烦的数学题。

先给出一些定义： $tr[G]$ 为矩阵 G 中主对角线的和。 $ans(l,r,k)$ 为矩阵 $G(l,r)$ 所对应的图中，长度**恰好**为 k 的闭合回路数量。则有：

$$ans(l,r,k) = tr[G(l,r)^k]$$

我们可以用容斥的思想来简化这个式子。把 $G(i)+I$ 看成图 i 的新矩阵，也就是允许每个回合不走。然后用二项式反演得到正确的答案，即：

$$\begin{aligned} ans(l,r,k) &= \sum_{i=0}^{k} \binom{k}{i} (-1)^{k-i} tr[(G_l+I)^i] * ... * tr[(G_r+I)^i] \\ &= \sum_{i=0}^{k} \binom{k}{i} (-1)^{k-i} P(l,r,i) \\ &= \sum_{i=0}^{k} \frac{k!}{i!(k-i)!} (-1)^{k-i} P(l,r,i) \\ &= k! \sum_{i=0}^{k} \frac{1}{i!(k-i)!} (-1)^{k-i} P(l,r,i) \\ &= k! \sum_{i=0}^{k} \frac{P(l,r,i)}{i!} \frac{(-1)^{k-i}}{(k-i)!} \end{aligned}$$

这个式子非常显然可以用 FFT 解决。但由于最后的答案模数并不是传统的 NTT 模数，然而容易知道最后答案不会超过 10^27。所以我们只需要选择三个 NTT 模数分别做 NTT，再将三个答案用中国剩余定理合并得到最后答案。

现在考虑如何计算 $P(l,r,i)$。非常常规的方法，直接用矩阵乘法算出正对角线上的和。但是这样做显然会超时，我们需要优化这个算法。

定义矩阵 M 的特征多项式 $C(M)$，记为：

$$C(M) = c_0 + c_1 M + c_2 M^2 + ... + M^N$$

根据 $Cayley\ Hamilton$ 定理,有 $C(M) = 0$。假设我们对上述等式左右两边都乘上 M^k ,则有:

$$c_0 M^k + c_1 M^{k+1} + c_2 M^{k+2} + ... + M^{k+N} = 0$$

同样满足一个恒等式，这也就意味着我们可以通过线性递推来得到 $M^{N+1}, M^{N+2}...M^{N+k}$。

如果把 M^k 换成 $tr[M^k]$，这个等式还是成立的，即：

$$c_0 tr[M^k] + c_1 tr[M^{k+1}] + c_2 tr[M^{k+2}] + ... + tr[M^{k+N}] = 0$$

有了上述等式， $P(l,r,i)$ 就可以快速算出了。回顾一下 $P(l,r,i)$，存在等式：

$$\begin{aligned} P(l,r,i) &= P(l,l,i) * ... * P(r,r,i) \\ &= tr[(G_l + I)^i] * ... * tr[(G_r + I)^i] \end{aligned}$$

所以我们只需要对每一张给定的图 l 都求出我们想要的 $P(l,l,i)$，那么所有的 P 就都可以得到了。

对于单张图 G_i，我们先求出它的特征多项式，再用矩阵乘法暴力算出前 N 项的 $tr[(G_i + I)^j]$，之后用线性递推得到所有的 $tr[(G_i + I)^j]$ 即可。

对于询问，我们可以离线对于相同 l, r 一起询问。对于相同的 l, r ,用 FFT 得到所有的 $ans(l,r,k)$。再做一遍前缀和就能得到我们想要的答案。

再整理一下本题的流程，即：

1.对每一张图求得其特征多项式，并用线性递推预处理所需要的信息。

2.然后离线做询问。每次用 FFT 计算答案，再用中国剩余定理合并之。

本题细节较多，涉及知识面比较广，还需要一定的卡时技巧才能通过本题。

时间复杂度$O(TN^4 + TNK + T^2 K \log K + Q)$，空间复杂度$O(N^3 + TK + Q)$。

【备注】

【关于特征多项式】

下面介绍本题中我求解矩阵 M 特征多项式的方法。

通过特征多项式的定义，有：

$$|M - \lambda * I| = 0$$

其中 $|M-\lambda*I|$ 表示它的行列式，而最后 λ^k 前面的系数就是特征多项式中的 c_k。

所以可以枚举 N 个不同的 λ 带入矩阵，并求的所对应的行列式。因为是多项式，所以将所得的结果用拉格朗日插值反推即可得到特征多项式的系数。

时间效率 $O(N^4)$。

【关于行列式】

只需按照一些法则，直接进行高斯消元即可。

【关于拉格朗日插值】

设当前我们已经得到 $k+1$ 个两两不同的点值，即：

$$(x_0,y_0),(x_1,y_1)....(x_k,y_k)$$

我们就可以构造出一个唯一的 k 阶的多项式 $L(x)$ 来拟合这个多项式。、

$$L(x)=\sum_{i=0}^{k}y_i*l_i(x)$$

$l_i(x)$ 的存在就是为了达到当 $x=x_i$ 时，$l_i(x)=1$，当 $x=x_j(j\neq i)$，$l_i(x)=0$。由此可以构造：

$$l_i(x)=\prod_{j=0,j\neq i}^{k}\frac{x-x_j}{x_i-x_j}$$

可以先预处理所有式子的乘积，然后对于每一项，多项式除法下来。这样我们就可以在 $O(N^2)$ 的时间内求出这些点对应的多项式系数。

【关于中国剩余定理】

我们先考虑一组两个一元同余等式的情况：

$$\begin{cases} Ans = a1 (mod\, m1) \\ Ans = a2 (mod\, m2) \end{cases}$$

考虑一种构造，令 $Ans = a1 + t * m1$ ，只需要满足：

$$a1 + t * m1 \equiv a2\ (mod\, m2)$$

即：

$$t \equiv (a2 - a1) * m1^{-1} (\ mod\, m2)$$

这样求得的肯定是最小解。对于多个方程，只需要逐个合并即可。

【关于FFT】

FFT，即快速傅里叶变化，用于 $O(N \log N)$ 解决一类多项式卷积问题。详见 Fast Fourier transform。

1.2 Simple Queries

【试题来源】

Codechef AUG 2015 DISTNUM

【试题大意】

有一个为 N 的序列 A。需要进行 Q 次操作，包含以下几种类型：

1. l, r : 令 S 为 $A_l, A_{l+1}, .., A_r$ 中所有不同元素构成的有序集合。你需要求出

$$\sum_{1 \le i < j < k \le |S|} S_i S_j S_k$$

2. x, y : 令 $A_x = y$

3. x : 删除 A_x

4. z, y : 在 A_z 后面插入元素 y ,若 $z = 0$,则在数组最前端插入

5. l, r : 输出下标在 l 到 r 范围内的不同元素个数

数据范围：$N, Q \le 10^5$

【算法介绍】

这是一道经典的数据结构题。

先考虑如何计算询问类型 1。运用一点容斥的知识，其实那个式子等价于:

$$(\sum S_i)^3 - 3 * (\sum S_i^2) * (\sum S_i) + 2 * (\sum S_i^3)$$

对于询问 5 ,其实就是询问：$\sum S_i^0$。

所以我们需要维护一个区间不同的数的零次方和,一次方和,二次方和以及三次方和。

对于维护一个区间内**不同数**的信息，这是个非常常见的问题。我们需要定义 $Pre[i]$ 为 i 号位置向前第一个权值等于 A_i 的位置。每次询问一个区间 l, r 。满足要求的数就是：

$$Pre[i] < l \text{ 且 } l \leq i \leq r$$

因为存在两个限制，根据常识，我们可以用树状数组套线段树来实现。

但目前还有一个问题就是存在插入和删除操作，这导致每个点的下标会发生变化。但因为允许离线，所以我们可以先用平衡树实现插入删除，从而得到最终序列。再将每次操作的下标进行更改即可。如果强制在线，可以用重量平衡树解决。

本题时限非常紧，还需要一定的常数优化技巧才能通过。

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

1.3 Easy exam

【试题来源】

Codechef JULY 2015 EASYEX

【试题大意】

有一个 K 面的骰子，你将掷 N 次。假设最终每种颜色正面朝上的次数为 A_i ，求：

$$E[A_1^F * A_2^F * .... * A_L^F]$$

其中 E 表示期望。

数据范围：$N, K \leq 10^9$ ， $L * F \leq 50000$ ， $F \leq 1000$。

【算法介绍】

这是一道非常巧妙的数数题。

我们设 $X_{i,j}$ 表示第 i 种颜色在第 j 轮是否正面朝上，正面朝上为 1 ，否则是 0 。显然对于每一轮，只有一种颜色是正面朝上的。由此，答案可以这样被表示：

$$E[(X_{1,1} + X_{1,2} + .. + X_{1,N})^F * .. * (X_{L,1} + X_{L,2} + .. + X_{L,N})^F]$$

现在我们对于每一个 $i(1 \leq i \leq L)$,将 $(X_{i,1} + X_{i,2} + .. + X_{i,N})^F$ 暴力展开。展开的过程可以认为是从 N 个数中顺序选择 F 个变量然后乘起来，我们设这个展开的式子为 $A(i)$。显然 $A(i)$ 有 N^F 个长度为 F 个变量组组成。

接下来我们要求的就是 $E[A(1) * A(2) * ... * A(L)]$ 。我们可以再暴力展开，也就是从每一个 $A(i)$ 中选择一个长度为 F 的变量组，然后全部乘起来，为一个长度为 $L * F$ 的变量组。

现在考虑用期望的线性性来解决本题。

我们考虑每个长度为 $L*F$ 的变量组，如果它要对答案贡献 1，那么其中所有变量都要是 1。但如果这项中存在 i,j,k，满足 $X_{i,k}$ 和 $X_{j,k}$ 都在这项中出现，那么这项肯定等于 0。因为每一轮只有一个骰子正面朝上。考虑这一项中不同的轮数有 P 种，则他对答案的贡献就是 $\frac{1}{K^P}$。

我们发现不同项对答案的贡献至于其中包含了多少不同的轮数有关。并且两两之间的颜色出现的轮都不能相同。所以我们只要处理出对于 $(X_{i,1}+X_{i,2}+..+X_{i,N})^F$ 的生成函数，其中指数表示有多少不同的轮数，系数表示方案数。设这个生成函数为 $G(x)$。

直接求解 $G(x)^L$，其中 x^P 前面的系数就是不同轮数为 P 时的方案数，而我们只需让它乘上 $\frac{N!}{(N-P)!}$ 及 $\frac{1}{K^P}$ 贡献到答案即可。

考虑如何求解 $G(x)$ 的系数。容易发现这就是第二类的 $Stirling$ 数，这是可以用 Dp 计算出来的，转移如下：

$$S[i][j]=S[i-1][j-1]+S[i-1][j]*j$$

得到 $Stirling$ 数以后， $G(x)$ 就可以表示成：

$$G(x)=S[F][0]+S[F][1]*x+S[F][2]*x^2+...+S[F][F]*x^F$$

考虑求解 $G(x)^L$，直接倍增+ FFT 即可解决。虽然模数不能使用 NTT，但是因为很小，所以直接用 FFT 即可保证精度。

时间复杂度$O(LF\log LF+F^2)$，空间复杂度$O(LF+F^2)$。

1.4 A game on a graph

【试题来源】

Codechef JULY 2015 HAMILG

【试题大意】

有一张 N 个点 M 条边的图。现有两个人在这张图上进行游戏，开始先手选择一个点，然后两个人轮流选一个还没有走过的点走过去。如果某个人不能移动了则视为这个人输了。假设两个人都采取最优策略，问先手可以选择多少点开始走使得必胜。

数据范围：$N \leq 2000$。

【算法介绍】

这是一道非常经典的图匹配问题。

我们这样考虑问题。我们先求一下这张图的最大匹配，假设先手选择一个点 u 作为起点。如果 u 是一个孤立点，即没有任何的出边，那么选择 u 显然是必胜的，因为后手没有点可以选择。否则 u 点肯定有边相连，但如果存在一个最大匹配使得点 u 不在其中，因为最大匹配的性质，所以此时 u 所有的出边对应的点肯定都在最大匹配中。此时先手肯定可以保证走出一条长度为奇数个点的增广路，因为无论后手走到哪一个点，先手总可以走到那个点对应的匹配点上去，此时同样先手必胜。

然而除去这些情况，也就是点 u 肯定出现在这张图的最大匹配中的时候。后手总有方法使得先手无路可走，此时先手必败。

现在的问题就是要求出有哪些点不一定存在于图的最大匹配中。我们可以先求出图的最大匹配。当前所有未匹配点显然不一定在最大匹配中。且那些可以从某一个未匹配点经过长度为偶数个点的路径到达的点都不一定在最大匹配中。

因为题目给定的是普通图，需要用带花树算法求解图的匹配。

时间复杂度$O(N^3)$，空间复杂度$O(M)$。

【备注】

【关于带花树算法】

普通图相比二分图，在匹配问题上的主要差别就是存在奇环。带花树算法的主要思路就是特判将奇环缩起来，然后特判一下环上的情况，较为复杂。

详情可以参见 Blossom_algorithm。

1.5 Chefbook

【试题来源】

Codechef JUNE 2015 CHEFBOOK

【试题大意】

给定 M 组数 $L_{x,y}, S_{x,y}, T_{x,y}$。要求构造 N 对数 P_x, Q_y，使得对于 M 组数都满足：

$$S_{x,y} \leq L_{x,y} + P_x - Q_y \leq T_{x,y}$$

在满足条件的情况，要求 M 组 $L_{x,y} + P_x - Q_y$ 的和最大，输出和以及对应的方案。

数据范围：$N \leq 100$，$M \leq N * N$。

【算法介绍】

这是一道巧妙的线性规划问题。首先转化一下问题：

给定一个矩阵 A 和 M 个数 $L_{i,j}$,表示 $A_{i,j} = L_{i,j}$。剩下的格子都是空的。对于每一行 i，可以整行加 P_i；同样对于每一列 j，可以整列减 Q_j。注意那些空的格子是不进行修改的。定义:

$$W_{i,j} = L_{i,j} + P_i - Q_j$$

我们的目标就是最大化 $W_{i,j}$ 的和并且满足：

$$S_{i,j} \leq W_{i,j} \leq T_{i,j}$$

我们考虑用线性规划来解决本题，先定义:

1、 $sizeRow(i)$为第 i 行非空格子的数量。

2、 $sizeCol(j)$为第 j 列非空格子的数量。

我们先把 $L_{i,j}$ 加到答案里，之后只要最大化：

$$\sum P_i * sizeRow(i) - \sum Q_j * sizeCol(j)$$

同样，每一个不等式约束可以拆成两个不等式:

1、 $P_i - Q_j \leq T_{i,j} - L_{i,j}$

2、 $Q_j - P_i \leq L_{i,j} - S_{i,j}$

运用线性规划的知识，我们可以把变量都认为是 x_i ，所有的约束可以描述成矩阵的形式，即要求 $Ax \leq B$ 。对于矩阵 A 的每一行都存在恰好一个 $+1$ 和一个 -1 。把目标函数系数也看成是矩阵，设为C。我们的目标就是最大化 Cx.

类比许多线性规划的问题，本题中我们需要将该线性规划问题转化成对偶问题。即设变量为 y ,约束为 $A^T y \geq C$ 。目标函数为最小化 By. 我们发现这个等式左右两边分别相加都是 0 ，也就是意味着我们可以把 $\geq$ 变成 $=$。

于是就变成了非常经典的费用流解决线性规划问题。对于A^T 中每一列，只有一个-1 和1。把它类比为图上一条边，那个变量就是流量。我们目标函数中的系数就是边的费用。对于等式的常数项我们认为是从图中的源或汇连接的边。直接用费用流解决就可以了。

对于输出方案，我们同样需要运用一些线性规划问题的小技巧，用 $SPFA$ 解决即可。

时间复杂度$O(costflow(N, M))$，空间复杂度$O(M)$。

1.6 Chef and Balanced Strings

【试题来源】

Codechef MAY 2015 CBAL

【试题大意】

给定一个长度为 N 字符串 S，全部都是小写字母。我们定义 S 的一个子串 $S[i,j]$ 是**平衡串**的，当且仅当 $S[i,j]$ 中所有字符都只出现偶数次。现进行 Q 次询问，每次询问给出三个整数 l,r,t，你需要输出:

$$\sum_{\substack{l \le i < j \le r \\ S[i,j]\text{是平衡串}}} (j-i+1)^t$$

数据范围：$N,Q \le 10^5$， $0 \le t \le 2$。

【算法介绍】

这道题我们用 $HASH$ +分块解决。

我们可以预处理出串 S 中每个位置 i 的 A_i，表示位置 i 前缀中所有字母出现的奇偶性。因为字符集很小，直接用有一个二进制数来表示 A_i 就可以了,其中 $A_0 = 0$。

处理好 A_i 之后，如果 $S[i,j]$ 是平衡串，当且仅当：$A_j = A_{i-1}$。

考虑一种暴力的做法，我们对于所有不同的 A_i 维护一个 $G0[j], G1[j], G2[j]$,分

别表示当前所有 $A_i = j$ 的 i^0, i^1, i^2 之和。每次加入一个新的 i ,对答案的贡献是:

$$\begin{aligned} Ans0 &= Ans0 + G0[A_i] \\ Ans1 &= Ans1 + i * G0[A_i] - G1[A_i] \\ Ans2 &= Ans2 + i^2 * G0[A_i] - 2 * i * G1[A_i] + G2[A_i] \end{aligned}$$

其中 $Ans0, Ans1, Ans2$ 分别表示询问 $t = 0, 1, 2$ 时的答案。算完对答案的贡献之后再更新 $G0, G1, G2$ 。

然而这样显然会超时。于是我们考虑用分块来优化本题。将整个序列划分成 $\sqrt{N}$ 块，对于每一块维护好从 1 到本块块头这段前缀的 $G0, G1, G2$ 。并维护好任意两块之间的答案。

每次询问我们可以用前面预处理好的块头前缀和得到这段区间的信息，之后暴力做超出两段的位置就可以了。单次效率为 $O(\sqrt{N})$ 。

时间复杂度$O(N\sqrt{N})$，空间复杂度$O(N\sqrt{N})$。

1.7 Counting on a directed graph

【试题来源】

Codechef MAY 2015 GRAPHCNT

【试题大意】

给出 N 个点 M 条边的有向图，统计有多少无序对 (x, y)，满足存在一条 1 到 x 的路径和一条 1 到 y 的路径，且这两条路径除了 1 号点以外没有其他公共点。

数据范围：$N \leq 100000$，$M \leq 500000$。

【算法介绍】

这是一道关于图的必经点算法的问题。

我们可以这样考虑本题：如果存在一个点 z $(z \neq 1)$，它既是从 1 到 x 所有路径的必经点，又是从 1 到 y 所有路径的必经点，那么 (x, y) 就不是合法的，反之就是合法的。

所以我们只要得到每个点的必经点就可以解决这道题了。关于图的必经点，存在一个非常经典的算法： *Dominator tree* 。

根据这个算法的介绍，容易知道，每个点关于一个固定起点 s 的最近必经点可以构成一个树模型。所以我们只需建立出这张图关于 1 号点的 *Dominator tree*，然后进行简单的树形 Dp 即可算出本题所要的答案。

时间复杂度$O(N\alpha N + M)$，空间复杂度$O(M)$。

【备注】

【关于 *Dominator tree*】

关于 *Dominator tree* 算法的详细介绍，可以参见： Dominator 或者《图连通性若干拓展问题探讨》（李煜东）。

1.8 Black-white Board Game

【试题来源】

Codechef April 2015 BWGAME

【试题大意】

有一个 $N*N$ 的矩阵A，给出 N 组 L_i, R_i 描述这个矩阵，表示矩阵 A 的第 i 行的第 L_i 到 R_i 列都是 1 。有两个人 Alex 和 Fedya 在矩阵 A 上进行游戏。每个回合 Alex 会选择一个之前没有选择过的长度为 N 的排列 P 。要求 P 中的逆序对数为偶数，且满足 $A_{i,P_i} = 1$; 同样 Fedya 也会类似选择一个排列 P ，但其中的逆序对数为奇数。

谁先找不到合法的排列就视为另一个人赢，如果两人都找不到合法的排列则视为平局。请输出这个游戏结果。

数据范围：$N \leq 100000$。

【算法介绍】

我们可以根据矩阵的行列式来解决此问题。

观察行列式的积和式求法：对于所有长度为 N 的排列 P ，将 A_{i,P_i} 全部乘起来。如果 P 中的逆序对数为偶数则加到行列式中，否则将行列式减去该值。于是我们发现题目所要求解的其实就是矩阵 A 的行列式。

传统的行列式求法类似于高斯消元，时间效率为 $O(N^3)$ 。然而本题中 N 非常大，这种做法显然无法通过该题。但本题的矩阵有特殊性，即每一行的 1 都是连续的一堆。这启示我们是否可以用数据结构来优化高斯消元。

没错，本题中可并堆就可以实现高斯消元的优化。我们对于每一列开一个可并堆，将行 i 合并到 L_i 列的堆中，之后按照 R_i的大小建堆。

接下来直接做高斯消元。我们每次从当前列的堆中把堆头 x 拿出来作为这一行。再将堆中其他元素合并到 $R_x + 1$ 列的堆中，这可以理解为消元，即将一堆连续的 1 全部做差消掉了。注意交换两行的时候行列式要取相反数，仔细模拟就可以算出行列式并得到答案。

时间复杂度$O(N \log N)$，空间复杂度$O(N)$。

1.9 Little Party

【试题来源】

Codechef April 2015 LPARTY

【试题大意】

给出 M 个集合，每一集合中都有 N 个变量，每个变量可以是 0 或者 1 。如果一个集合包含其中 x 个变量，且对于剩下 $N-x$ 个变量的所有可能情况的集合都在这 M 个集合中出现过，则这个集合被称为是基子集，且我们称这个基子集可以覆盖所有给出的且包含这个基子集的集合。

你需要找出一些基子集，使得它们能覆盖所有给出的集合，且要求长度和最小的。

数据范围：$N \leq 5$ ，$M \leq 2^N$。

【算法介绍】

这道题我们用搜索+剪枝解决。

可以先暴力求出所有的基子集。然后将每个基子集能覆盖的集合求出来。于是问题就变成了最小点权覆盖。

这是一个非常经典的问题，我们可以用 DFS 来解决。当然优化是必不可少的，我列出了一些效果较好的优化，如下：

1、如果一个点 x 覆盖的集合是另一个点 y 覆盖集合的子集，那么点 x 我们可以直接舍去。

2、我们将所有点覆盖的集合进行二进制压位，并且从大到小排序。如果就算后面的点全部选上还是不能覆盖全集，那么直接剪枝。

3、如果当前所选长度大于等于现有答案则直接剪枝。

4、如果选上这个点对覆盖状态没有变化则直接剪枝。

5、具体实现可以用 BFS 来实现,即每一层用队列将有效的状态存下来。

加上上述优化就可以通过本题了。

时间复杂度$O(2^M)$，空间复杂度$O(M)$。

1.10 Counting on a Tree

【试题来源】

Codechef March 2015 TREECNT2

【试题大意】

给出一棵 N 个点的无根树,每条边的边权为 c_i。需要统计有多少无序点对 (x,y) $(x \neq y)$。他们路径上边权的最大公约数为 1。

并且进行 Q 次操作，每次操作给出 x,v，表示将树上第 x 条边边权改成 v。在每次操作后输出满足要求的无序点对数。

数据范围：设最大边权为 M，$N \leq 100000$，$Q \leq 100$，$M \leq 10^6$。

【算法介绍】

我们设路径边权的最大公约数是 d 的倍数的方案为 $G(d)$。则：

$$Ans = \sum_{d=1}^{M} \mu(d)G(d)$$

考虑计算 $G(d)$。只要把所有边权是 d 的倍数的边都选出来，然后做并查集。如果一个集合 i 的点数为 siz_i ,则对答案的贡献即是 $\frac{siz_i*(siz_i-1)}{2}$。

因为每一条边只会在 $\mu(d) \neq 0$ 且 $d|c_i$ 时才会被选到且对答案有贡献。所以一条边最多只会被选 2^{lim} 次。其中 lim 为将最小的 lim 个质数相乘乘积大于等于 M 的最小值，在本题中大约为 7。

这样如果没有修改，做法的时间复杂度是 $O(2^{lim}N\alpha(N))$。

对于有修改的情况，我们可以离线做。因为 Q 很小，所以对于每个 d ,我们把之前选出来的边分成两类。第一类是在 Q 次操作里没有被修改，第二类则是被修

改的。对于第一类的边，我们直接做并查集就可以了，因为这是没有任何影响的。第二类的边数量肯定小于等于 Q 。我们直接对于每次操作，暴力将那第二类边并查集就可以了，之后再还原并查集数组即可。

如果 Q 比较大，我们还可以用一些分治技巧来解决。

时间复杂度$O(2^{lim}(N\alpha(N)+Q^2)+MQ)$，空间复杂度$O(M)$。

1.11 Random Number Generator

【试题来源】

Codechef March 2015 RNG

【试题大意】

给定一个线性递推式：

$$A_i = (A_{i-1} * C_1 + A_{i-2} * C_2 + ... + A_{i-k} * C_k) \mod 104857601$$

给出 $A_0, A_2, ..., A_{k-1}$ 和 $C_1, C_2, ..., C_k$，求解 A_N。

数据范围：$N \leq 10^{18}$，$K \leq 30000$。

【算法介绍】

这是一个非常经典的问题，我们先考虑用矩阵乘法来解决这个问题。首先这个递推式的转移满足如下的矩阵转移：

$$\begin{bmatrix} C_1 & C_2 & C_3 & \cdots & C_k \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} A_{i-1} \\ A_{i-2} \\ A_{i-3} \\ A_{i-4} \\ \vdots \\ A_{i-k} \end{bmatrix} = \begin{bmatrix} A_i \\ A_{i-1} \\ A_{i-2} \\ A_{i-3} \\ \vdots \\ A_{i-k+1} \end{bmatrix}$$

同理可以得到：

$$\begin{bmatrix} C_1 & C_2 & C_3 & \cdots & C_k \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}^{N-k+1} \begin{bmatrix} A_{k-1} \\ A_{k-2} \\ A_{k-3} \\ A_{k-4} \\ \vdots \\ A_0 \end{bmatrix} = \begin{bmatrix} A_N \\ A_{N-1} \\ A_{N-2} \\ A_{N-3} \\ \vdots \\ A_{N-k+1} \end{bmatrix}$$

最后目标向量的最上面的元素便是我们需要的答案。

设转移矩阵为 M。对此，我们先求出矩阵 M 的特征多项式，记为 $F(M)$。因为本题矩阵的特殊性， M 的特征多项式为:

$$F(M) = M^K - C_1 * M^{K-1} - C_2 * M^{K-2} - .... - C_k$$

根据 $Cayley\ Hamilton$ 定理，有:

$$F(M) = 0$$

现在我们要求的是 M^{N-k+1}。我们可以将 M^{N-k+1} 对 $F(M)$ 取模，可以表示成：

$$M^{N-k+1} = D(M) * F(M) + R(M)$$

其中 $D(M)$、$R(M)$ 分别为这次除法的商和余数。因为 $F(M) = 0$，所以 $R(M) = M^{N-k+1}$。因为 $R(M)$ 的最高项系数是小于等于 $F(M)$，即小于等于 K。之后我们将转移矩阵 M 和初始向量 S 带入就可以求出最终向量 T，即：

$$T = \sum_{i=0}^{k-1} S * M^i * R(M)[M^i]$$

因为我只要求 T 向量中最上面的元素，所以我们可以直接将 $S * M^i$ 这个向量最上面的元素带入，观察发现其中 $S * M^i$ 最上面的元素其实就是 A_i，所以答案

Ans就可以这样表示:

$$Ans = \sum_{i=0}^{k-1} A_i * R(M)[M^i]$$

现在还剩下的问题是如何求解 $R(M)$。这个东西我们可以用倍增 FFT 和多项式取模来解决。

对于一般运算，除数为 0 的除法没有定义，但是在本题中，大抵可以理解为是做一个多项式降幂的工作。

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

【备注】

【关于多项式】

本题中涉及到的多项式算法主要有：多项式乘法，多项式求逆，多项式除法。详情请参照picks的博客

1.12 Devu and Locks

【试题来源】

Codechef Feb 2015 DEVLOCK

【试题大意】

给出 N, P, MM，要求对于 $M = 1...MM$ 都求出有多少 N 位的数，它能被 P 整除且各数位和小于等于 M（允许有前导零）。

数据范围：$N \leq 10^9, P \leq 50, MM \leq 15000$。

【算法介绍】

先考虑用比较常规的动态规划来解决这道题。我们设:

$$B[i] = 10^i \ mod \ P$$

设 $F[i][j][k]$ 的 i, j, k 分别表示当前做到第 i 位，该数对 P 取模的模数为 j，数位和为 k。易得下面的 Dp 方程：

$$F[i-1][j][k] \rightarrow F[i][(j + t * B[i]) \ mod \ P][k + t]$$

其中, t 为我们枚举这一位为 $0...9$。

现在考虑如何优化这个 Dp 过程。首先 $B[i]$ 有一个长度为 P 的循环节，对于 $B[i]$ 相同的那些 i，它们的转移方程是完全相同，我们是可以把它们一起处理的。

这问题也就变成了做若干次相同的 Dp，这其实我们可以用倍增得到这个 Dp 数组。即：

$$F[i][j][k] * F[i][x][y] \rightarrow F[2 * i][(j + x) \ mod \ P][k + y]$$

Dp 初始状态为 $F[1][t*B[i] \mod P][t]=1\ (t=0..9)$。

其实还有一个优化，因为所有的 $B[i]$ 都是相同的，所有在 $F[i][j][k]$ 中，其实有 $j=k*B[i] \mod P$。所以我们只要对 $F[i][k]$ 进行 DP，之后可以直接得到 $F[i][j][k]$。对于 $F[i][k]$ 的转移，非常显然是可以用倍增 NTT 来解决的。

把转移方程相同的处理好以后,我们再将各个 Dp 数组合并起来，合并的方程也与上述方程类似。但是因为现在合并的 B 不一样，所以还需要枚举模数合并。

时间复杂度$O(P^3MM+P^2MM\log MM+PMM\log MM\log N)$，空间复杂度$O(PMM)$。

1.13 Payton numbers

【试题来源】

Codechef Feb 2015 CUSTPRI

【试题大意】

定义三元组的乘法的过程 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))

若 s 为偶数，则返回(A-540,B-540,24)，否则返回(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，判断是不是质数。

数据范围：设数据组数为 T ，$T \leq 10000$，$abs|A, B, C| \leq 10^7$。

【算法介绍】

这是一道结论题。

运用一些抽象代数的知识，可以得到结论：发现只需要对读入的 A, B, C 进行一些简单的判断，并用 $Miller\_Rabin$ 即可。

时间复杂度$O(T \log A)$，空间复杂度$O(1)$。

1.14 Ranka

【试题来源】

Codechef Jan 2015 RANKA

【试题大意】

有两个人 A 和 B 一个 $9*9$ 的棋牌上在下围棋，每个回合两个人可以选择下棋或者不动。每个人下棋的原则要符合经典的围棋走法。要求每个人对于相同的局面不能重复面对。给出一个 N ，请构造一种两个人的走法，使得回合数为 N 。

数据范围：$N \leq 10000$。

【算法介绍】

这是一道有趣的构造题。我们可以通过下面这种方法来构造。

先让第一个人将除了 $(1,1)$ 这个格子的剩下格子全部填上，按照 $(9,9)$, $(9,8)$ .. $(1,2)$ 的顺序填棋子。这段过程中，第二个人不动。等第一个人完成该操作以后，第二个人把 $(1,1)$ 填上，则除了 $(1,1)$ 剩下所有棋子全部被吃光。然后第二个人再按照 $(9,9)$,$(9,8)$.. $(1,3)$ 的顺序把除了 $(1,2)$ 的格子全部填上，然后第一个人再把 $(1,2)$ 填上。

以此类推，重复这个过程，最后就可以构造出一个 81*81*2=13122 个回合的走法。可以通过此题。

时间复杂度$O(N)$，空间复杂度$O(N)$。

1.15 Xor Queries

【试题来源】

Codechef Jan 2015 XRQRS

【试题大意】

给定一个初始的空序列，需要进行 N 次操作。有下面几种类型：

0 x : 在数组最后面插入一个元素 x 。

1 l r x : 在区间 $[l,r]$ 找一个数 y ，使得 x xor y 最大。

2 k : 删除数组最后 k 个元素。

3 l r x ：在区间 $[l,r]$ 中，统计小于等于 x 的元素个数。

4 l r x ：在区间 $[l,r]$ 中，找到第 k 小的数。

数据范围：设元素最大值为 M ，$N \leq 500000, M \leq 10^6$。

【算法介绍】

这是一道简单的数据结构题。

对于 xor 的运算，非常容易想到用字典树来维护。其实字典树的本质就是一棵线段树，所以同样也支持本题中的另外两个询问。

现在考虑如何支持在最后加元素和最后删元素。这引导我们可以用可持久化算法来解决。对于每次在最后加一个元素，可以从原先最后一个数的可持久化字典树继承过来，然后插入元素。对于删除直接将末尾指针指到我们要的位置就可以了。

可以我们发现我们维护的是一个前缀的信息，所以每次询问 l,r 用 r 的信息减去 $l-1$ 的信息即可。

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

1.16 Divide or die

【试题来源】

Codechef Dec 2014 DIVIDEN

【试题大意】

给定一个 N 度角，要求仅用尺规将它 N 等分，输出方案。

数据范围：$N \leq 360$。

【算法介绍】

这是一道较麻烦的几何题。

首先有一个结论，若 N 是三的倍数，则肯定无解。否则我们按照以下方法构造：

先画出一个五边形，这样我们就可以得到72度角，再画出一个等边三角形，这样就能得到一个60度角。之后我们就可以得到一个3度角。用3度角不断去划分那个 N 度角。最后就肯定可以得到一个1度角。这样就可以将N度角N等分了。

时间复杂度$O(N)$，空间复杂度$O(1)$。

【备注】

【关于尺规作图】

关于如何用尺规画出五边形，请参见：网站资料。

1.17 Course Selection

【试题来源】

Codechef Dec 2014 RIN

【试题大意】

有 N 门课程，一共 M 个学期。每个课程都需要在其中一个学期学习，且一共只能学习一次。一个学期可以学习很多课程。

我们设 $X_{i,j}$ 表示第 i 门课在第 j 学期学习，期望的得分为 $X_{i,j}$。同样会有 K 个限制，要求第 A_i 门课在第 B_i 门课之前先学。

需要回答，如何安排课程的学习使得在满足所有限制，且所有门课都学的情况下。每门课的期望平均分最大。

数据范围：设每门课的分数最大值为 Lim，则 $N, M, K, Lim \leq 100$。

【算法介绍】

这道题我们用最小割解决。

先考虑没有 K 个限制的情况怎么做。我们先把答案设为 $N * 100$。之后把 $X_{i,j}$ 设为 $100 - X_{i,j}$,表示第 i 门课在第 j 学期选，会少得多少分。对于每门课的每个学期建立一个点，每个点向该门课的下一个学期所对应的点连边，流量为 $X_{i,j}$。对于该图做最小割的结果即是最少会少得多少分吗，然后用 $N * 100$ 减去这个值就是我们想要的答案。

考虑有限制的情况怎么做。我们先观察这样连边以后，每一个割的实际意义。如果一个点与 S 相连，则表示这门课选择的时候大于等于这个学期。

所以我们只要将 A_i 课的每一个学期 j 的点向 B_i 课的 $j+1$ 学期连流量为 INF 的边。表示这两个点一定不能割。这样若 A_i 的第 j 个学期与 S 连，那么 B_i 的 $j+1$

学期也一定与 S 相连，这样就可以保证 B_i 选的学期一定在 A_i 后面。

直接对这个图跑网络流即可。

时间复杂度$O(maxflow(NM, NM))$，空间复杂度$O(NM)$。

1.18 Chef and Churu

【试题来源】

Codechef Nov 2014 FNCS

【试题大意】

有一个长度为 N 的序列 A 和 N 个函数 L_i, R_i，表示这个函数的值为 $A_{L_i} + A_{L_i+1} + ...A_{R_i}$。进行 Q 次操作，两种类型:

1.将 A_x 修改为 y。

2.询问第 a 个函数到第 b 个函数的和。

数据范围：$N, Q \leq 100000$。

【算法介绍】

这道题我们可以用分块来解决。

我们将 N 个函数分成 S 块，对于每个块 i，预处理出 $G[i][j]$，表示块 i 中的函数 A_j 一共加了多少次。这一步我们对于每一个块中的函数 k 在 $G[i][L_k]$ 加 1，在 $G[i][R_k + 1]$ 减 1。然后线性从 1 到 N 循环一遍，将 $G[i][j]$ 加上 $G[i][j-1]$。效率为 $O(SN)$。

对于每次单点修改，我们用树状数组维护前缀和，并对于每一个块维护和，这一步可以利用我们预处理好的 G 数组快速维护。效率为 $O(S)$。

对于询问，我们可以对于在 a, b 之间的块，直接累加和。对于多出来的函数用树状数组来求和。效率为 $O(\frac{N}{S} \log N)$。

效率最优的时候，显然是当 $S = \sqrt{N \log N}$ 最优。

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

1.19 Sereja and Order

【试题来源】

Codechef Nov 2014 SEAORD

【试题大意】

有 N 个程序，每个程序需要在 A, B 两台电脑上都要运行，运行时间分别是 A_i, B_i。注意一台电脑不能同时运行两个程序，一个程序不能同时在两台电脑上运行，先在 A 上运行还是在 B 上运行的顺序不作要求。

求一种方案使得将全部程序都运行完的时间最小。

数据范围：$N \leq 100000$。

【算法介绍】

这道题我们采用贪心+随机的方法解决。

首先答案的下界肯定是:

$$\max\{\sum A_i, \sum B_i, A_i + B_i\}$$

通过观察，这个下界肯定是可以取到的。如果答案是 $A_i + B_i$，那么我们可以直接将剩下的程序填进去，直接构造。否则，这个答案肯定是有很多种的。所以我们直接随机答案，并用一个简单的贪心验证一下答案即可。

时间复杂度$O(N * 人品)$，空间复杂度$O(N)$。

1.20 Children Trips

【试题来源】

Codechef Oct 2014 TRIPS

【试题大意】

给定一棵 N 个节点的树，每条边的边权为 1 或 2 。进行 Q 次询问。每次询问给出 u, v, d ，表示每天有 d 的体力，要求从 u 走到 v 需要多少天，每天只能停在点上。

数据范围：$N, Q \leq 100000$。

【算法介绍】

这道题我们用分块+倍增 LCA 来解决。

如果单次询问的 $d \geq \sqrt{N}$ ，那么答案小于等于 $\sqrt{N}$ 。所以我们直接用倍增找这一天达到的点，以此类推暴力得到答案。

对于 $d \leq \sqrt{N}$ 的询问，不同的 d 只有 $\sqrt{N}$ 种。我们离线一起处理这些询问。对于每个 d ，预处理每个点向上 d 可以走到那个点。对于每个询问，也同样倍增一下即可。

注意在两点的 LCA 处需要特判一些细节。

设树高为 H ，实践证明，当 d 的阈值取在 $sqrtH$ ，程序运行的最快。

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

1.21 Union on Tree

【试题来源】

Codechef Oct 2014 BTREE

【试题大意】

给定一棵 N 个节点的树，进行 Q 次询问。

每次给出 T_i 组 A_i, R_i，表示在 A_i 号点有一个警察可以保护距离他小于等于 R_i 的点。对于每次询问输出有多少不同的点被保护了。

数据范围：$N, Q \leq 50000$，$\sum T_i \leq 10^5$。

【算法介绍】

这道题我们用虚树和点分树来解决。

我们先考虑 $T_i = 1$ 的情况，这是一个非常的经典的问题，我们可以用点分树解决。因为没有修改，所以可以预处理，之后每次询问的效率即 $O(\log N)$。

考虑 $T_i \geq 1$ 的情况，类比许多树上的问题，我们可以联想到虚树。

但是直接统计很难将每个点不重复的计算到。于是我们可以通过一个虚树上的树形 Dp，从其他点中得到每一个点最大的 r_i。此时即不存在三个点以上的容斥，我们直接计算每一个点所包含的点的个数，再减去虚树上相邻点公共包含的部分即可。

容易发现，两个点公共包含的部分同样也是一个警察，所以可以和其他点方法一样进行统计。注意那个警察可能处于边的中点上，所以我们需要在预处理时，把每条边中间插入一个点，对于读入的 r，乘 2 即可。

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

【备注】

【关于虚树】

这里给出一种虚树的构造方法，设当前需要对 N 个点 $A[i]$ 构造虚树。

先将这 N 个点按照 Dfn 排序，其中 $Dfn[i]$ 为 i 号点的 DFS 序。然后将相邻两个点的 LCA 放入 A 数组中，并加入 1 号点。再将这些点排序后去重。这样所有与给出的 N 个点有关的树上节点就都在 A 数组中了!

而第 i 个点在虚树中的父亲就是 $LCA(A[i-1], A[i])$。

效率 $O(N \log N)$。

1.22 Rectangle Query

【试题来源】

Codechef Sept 2014 QRECT

【试题大意】

在一个二维笛卡尔坐标系上，需要支持 N 项操作，类型如下：

1. 插入一个矩形。
2. 删除一个矩形。
3. 询问和一个矩形有交的矩形有几个。

数据范围：$N \leq 10^5$。

【算法介绍】

这是一道非常经典的数据结构题。可以采用 CDQ 分治+线段树来解决。

直接计算相交的矩形比较麻烦，我们可以用总数减去不相交的矩形的个数。

观察到两个矩形不相交的位置关系只有 8 种，即八个方位。对于每一种位置关系，我们都可以得到它们关于四个顶点的关系。

有了这些关系以后，可以会发现这是一个非常经典的二维数点问题。具体实现可以用在线的树套树或者离线 CDQ 分治，前人之述备矣，不再赘述。

时间复杂度$O(N \log^2 N)$，空间复杂度$O(N)$。

1.23 Fibonacci Numbers on Tree

【试题来源】

Codechef Sept 2014 FIBTREE

【试题大意】

给定一棵 N 个点的树，进行 Q 次操作，包含以下几个类型，且强制在线：

1. 链加 Fib 数。
2. 链求和，对 10^9+9 取模。
3. 以某个点为根求一个点的子树和。
4. 变成第 R 次前的状态。

数据范围：$N,Q \leq 10^5$。

【算法介绍】

这道题我们用 Fib 数的通项公式+可持久化树链剖分解决。

对于 Fib 数的求和，我们可以用通项式来求解，设 $F[N]$ 为 Fib 数的第 N 项，有：

$$\frac{1}{\sqrt{5}}((\frac{1+\sqrt{5}}{2})^N-(\frac{1-\sqrt{5}}{2})^N)$$

所以我们只要维护加等比数列和等比数列求和就可以了。

对于树链问题，我们可以用树链剖分和线段树求解。对于询问子树问题，我们特判一下根和待询问的点的关系，用 DFS 序上询问区间信息即可。

由于要返回之前的状态，所以用可持久化树链剖分即可。

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

1.24 Team Sigma and Fibonacci

【试题来源】

Codechef Aug 2014 SIGFIB

【试题大意】

定义 $Fibo[i]$ 为斐波那契数列的第 i 项。给出 N, M,求解：

$$\sum_{x+y+z=N} 6 * x * y * z * Fibo[x] * Fibo[y] * Fibo[z] \mod M$$

T 组数据，每组数据给出 N_i, M_i。

数据范围：$N \leq 10^{18}$ ， $M \leq 10^5$ ， $\sum M_i \leq 10^6$

【算法介绍】

这道题我们用生成函数+矩阵乘法解决。

我们设斐波那契数列的生成函数为 $G(x)$ ，由数列 $k * Fibo[k]$ 构成的生成函数为 $F(x)$ 。则有：

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

运用生成函数的知识，易知 $F(x)$ 所对应的就是一个 12 项的线性递推式。设 N

的答案为 $Ans(N)$，则有：

$$\sum_{i=0}^{i=12} Ans(N-i) * F^3(x)[x^i] = 0$$

所以我们可以暴力算出前 12 项的递推式。然后用倍增矩阵乘法来求出第 N 项的结果即可。

因为 T 过大，所以我们需要对于 M 较小时预处理出循环节，然后直接 $O(1)$ 输出。实践证明，循环节的总和非常小。

时间复杂度$O(T \log N 12^3)$，空间复杂度$O(12^2)$。

1.25 Push the Flow!

【试题来源】

Codechef Aug 2014 PUSHFLOW

【试题大意】

给定一张 N 个点 M 条边的无向联通图。每条边有容量。且保证每一个点只存在于一个简单环。现进行 Q 次操作，有两种类型：

1、修改一条边的容量。

2、询问 S 到 T 的最小割，保证 $S \neq T$ 。

数据范围：$N, Q \leq 10^5$。

【算法介绍】

这是一道有趣的数据结构题。

我们这样考虑问题，最小割，顾名思义就是割掉一些边使得 S 和 T 不连通，且代价和最小。因为本题中图的特殊性， S 到 T 的路径可以描述成若干环和若干链。

割法肯定只有两种可能，一是将一条链上的边割掉，二是将一个环上的两条边断掉。

考虑没有环的情况，即给定的是一棵树。非常显然我们只需要求得链上最小值即可。若是环，可以发现，若要割断这个环，则环上权值最小的边肯定要割掉。因为要保证不连通，且从两个方向上去，肯定要经过权值最小的边。

容易发现将环上权值最小的边割掉以后，整张图就变成了一棵树！而这条环上的权值最小边对答案的贡献，当且仅当最后答案要求割掉一条该点所属环上的一条边。所以我们直接将这条边属于的环上的另外所有边边权加上该边的边权即可。

这样问题就可以与树上情况类同了。

再考虑我们要维护哪些信息。首先我们需要 $Tarjan$ 得到所有的环，并用 set 维护每个环上最小边权的边。我们还需要一个支持树上的 $Link, Cut$ ，树链加，树链求 Min 的数据结构。直接 LCT 裸上即可。

时间复杂度$O(N \log N)$，空间复杂度$O(N)$。

1.26 Game of Numbers

【试题来源】

Codechef July 2014 GNUM

【试题大意】

给定两个长度为 n 的数组 A, B，每一轮你要选出两个数对 (i, j) 和 (p, q)，满足 $A_i < B_j, A_p > B_q$且 $\gcd(A_i, B_j, A_p, B_q) > 1$，$(i, j)$ 不在集合 S_1 中，(p, q) 不在 $S2$ 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：$n \leq 400$，$A_i, B_i \leq 10^9$。

【算法介绍】

这道题我们用网络流来解决。

我们把 N^2 个点对按照 A_i, B_j 的大小关系分成左右两个部分。对于可以同时选出的点对之间连容量为 1 的边，然后跑最大流就是答案。

然而这样做的网络流图点数为 $O(N^2)$ 个，边数为 $O(N^4)$，这样显然是会 TLE 的。

我们可以优化一下构图，对于每一个质因子建立一个点。对于一边里 gcd 相同的点我们缩在一起，然后对该 gcd 的因子连边。这样的点数为 $O(N^2)$，边数为 $O(N^2 \log N)$。实际测试点数和边数远远不到这个上界。

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

1.27 Sereja and Equality

【试题来源】

Codechef July 2014 SEAEQ

【试题大意】

定义两个长度为 N 的序列 A 和 B 相似，当且仅当每一个 A_i 和 B_i 在各自数组中的相对大小相同。

定义两个长度为 N 的序列 A 和 B 的相似度，为 (l,r) 满足 $A_l, A_{l+1}..A_r$ 与 $B_l, B_{l+1}..B_r$ 相似，且 A 中的逆序对数不多于 M 的点对数。

进行 T 次询问，每次给出 N, M 。表示询问对于数组 A, B 为所有长度为 N 的排列， A 和 B 的相似度总和。

数据范围：$N \leq 500, M \leq 1000000, T \leq 100000$。

【算法介绍】

这是一道简单的数数题。

我们可以先用 Dp 算出 $F[i][j]$ 表示长度为 i 的排列，逆序对为 j 的方案数。

常规的方法，枚举第 i+1 位选择大小为 k 的数，然后将 $F[i][j]$ 加到 $F[i+1][j+k]$ 中去。

可以发现一个 $F[i][j]$ 可以对 $F[i+1][j..j+i]$ 这一段区间产生贡献。所以直接前缀和优化一下，就可以在 $O(N^3)$ 的时间内算出 500 以内所有的 $F[i][j]$ 。

对于 T 次询问。每次我们 $O(N)$ 枚举相似段的长度，然后通过简单的计算得到答案，为:

$$Ans = \sum_{i=0}^{N} F[i][min(M, \frac{i*(i-1)}{2})] * \binom{N}{i}^2 * (N-i+1) * L[N-i]^2$$

其中 $L[N] = N!$。

时间复杂度$O(N^3 + TN)$，空间复杂度$O(N^3)$。

1.28 Two Companies

【试题来源】

Codechef June 2014 TWOCOMP

【试题大意】

给定一棵 N 个节点的树，有两个公司分别有 $M1, M2$ 条铁路线路，每条线路从 u 到 v，且有 c 的收益。

要求在两个公司里找出一些线路的子集。使得两个公司分别建造的线路没有交集。要求收益和最大。

数据范围：$N \leq 100000, M \leq 700$。

【算法介绍】

这是一道非常经典的网络流问题。

如果我们解决了判断两条线路是否有交，我们就可以用网络流解决了，构图如下：

1. 源向第一个公司的所有线路连容量为收益的边。
2. 第二个公司的所有线路向汇连容量为收益的边。
3. 两个公司之间不能一起选的连容量为 INF 的边。

答案为收益总和减去网络流的最小割。

考虑判断两条线路有交，我的方法是用树链剖分，然后用两个指针指过去判断 DFS 序是否有交。

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

1.29 Sereja and Arcs

【试题来源】

Codechef June 2014 SEAARC

【试题大意】

给定一个长度为 N 的序列，每个点有一个颜色 A_i。现在对于所有颜色相同的 i,j 以 i,j 为直径画一个圆。问最多有多少颜色不同的圆相交。

数据范围：$N \leq 100000$。

【算法介绍】

这是一道有趣的分块题。

我们设定一个阈值 S，分两种情况讨论问题。

第一种情况为相交的两种颜色至少有一种颜色在 A 出现次数大于等于 S。则这种颜色的个数不会超过 $\frac{N}{S}$。

我们枚举其中一个出现次数大于等于 S 的颜色 Col，记 Sum_i 为序列中下标为 i 的位置之前出现了多少 Col。

接着我们对于另外的颜色统计它与 Col 的有多少相交。对于颜色相同的一串我们用链表指过去，答案为：

$$\sum_{i>j} (Sum_i - Sum_j) * Sum_j$$

这个东西非常好维护。于是这一部分的效率就是 $O(N\frac{N}{S})$。

第二种情况即两种颜色出现次数都小于等于 S，对于这种情况，所有可能的圆弧不会超过 $\frac{N}{S}S^2$ 个。于是我们可以暴力枚举所有圆弧，用树状数组统计一下相

交的个数即可。但是这样会把相同颜色之间的相交也会统计进去，需要将不合法的减去。这一步的效率为 $O(NS\log N)$

经过计算，$S=\sqrt{\frac{N}{\log N}}$ 时，效率最好。

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

1.30 Dynamic Trees and Queries

【试题来源】

Codechef May 2014 ANUDTQ

【试题大意】

给定一棵 N 个节点的树，进行 Q 次操作。需要支持以下几个类型：

1.新建一个节点，把父亲设为 x 。

2.子树加权值。

3.删除子树。

4.询问子树和。

数据范围：$N, Q \leq 100000$ 。

【算法介绍】

这是一道非常经典的数据结构题。

对于子树信息的修改和询问，非常容易联想到 DFS 序。所以我们用 $Splay$ 动态维护这棵树的 DFS 序。这时操作就只有单点插入，区间删除，区间加和询问区间和。这些都是 $Splay$ 非常基础的操作，不再赘述。

时间复杂度$O(N \log N)$，空间复杂度$O(N)$。

1.31 Sereja and Subsegment Increasings

【试题来源】

Codechef May 2014 SEINC

【试题大意】

有两个长度为 N 的数组 A, B。你每次可以将数组 A 中一段区间 $[l, r]$ 在模 4 域下加 1。求最少进行多少次操作，使得 $A = B$。

数据范围：$N \leq 100000$。

【算法介绍】

首先我们将 $A_i = (B_i - A_i + 4) \mod 4$，再另 $A_i = A_i - A_{i-1}$。

现在问题就变成了初始有一个全 0 的序列 C，每次选择 (i, j)，其中 $i < j$，将 C_i 加 1 , C_j 减 1。使得 $C = A$。

如果没有模域的话，可以直接另 $Ans = \sum \max(0, A_i)$。因为有模域，所以我们可以选择 (i, j)，其中 $i < j$，将 A_i 加 4， A_j 减 4。使得正数和最小。

这样我们就可以贪心，注意如果一个数大于等于 2。只要前面存在 -2 或者 -3 ,我们可以将前面的数加 4，当前数减 4。这样答案肯定不会变劣。

时间复杂度$O(N)$，空间复杂度$O(N)$。

1.32 Chef and Tree Game

【试题来源】

Codechef April 2014 GERALD08

【试题大意】

有一棵 n 个节点的树，树上每一条边的颜色都为红色或者蓝色。现在两个人轮流进行操作，第一个人每次选择一条红色边删除，第二个人每次选择一条蓝色边删除，删除后和树根（ 1 号点）不连通的部分将被删除，若干轮之后不能操作的人算输。如果两个人都使用最优策略，问第一个人先手时、第二个人先手时分别是谁赢得游戏。

数据范围：$n \leq 100000$ 。

【算法介绍】

对每一个节点定义一个局面函数 f ，可以使用递推的方式来得到：叶子的函数值为 0 。每一个节点的函数值为所有孩子的贡献之和，对于一个孩子 i ，如果连接它的是红边，那么令 a 为 $f_i + a > 1$ 的最小正整数，它的贡献就是 $\frac{f_i+a}{2^{a-1}}$ 。如果连接的是蓝边，那么令 a 为 $f_i - a < -1$ 的最小正整数，它的贡献就是 $\frac{f_i-a}{2^{a-1}}$ 。

如果根节点的函数值是为 0 ，那么谁后手谁赢；如果是整数，那么无论如何都是第一个人赢；否则无论如何都是第二个赢。至于求函数值，可以用高精度来实现，可以发现一个大小为 i 的子树根节点函数值的位数不超过为 i ，所以可以对每一个子树开一棵平衡树来维护每一位，然后用启发式合并的方式来实现两个高精度数的加。

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

1.33 Chef and Graph Queries

【试题来源】

Codechef March 2014 GERALD07

【试题大意】

给定一张 N 个点 M 条边的图，进行 Q 次询问。每次给出 l_i, r_i，表示保留图中编号为 l_i 到 r_i 的边，询问有多少个连通块。

数据范围：$N, M, Q \leq 200000$。

【算法介绍】

这道题我们用 LCT 维护最小瓶颈生成树+树状数组解决。

我们枚举每一条边 u, v，如果 u, v 在 LCT 中不连通，就将他们连接起来。如果两者已经联通，我们把 LCT 中 u, v 链上编号最小的边删掉，然后连上当前边。再用一个树状数组维护一下在 LCT 中的树边的编号和。

对于每次询问，我们对于 R 相同的一起做，我们把 1 到 R 的边都连上，然后在树状数组里询问一下编号在 L 到 R 的边有多少条即可。

时间复杂度$O(N \log N)$，空间复杂度$O(N)$。

1.34 The Street

【试题来源】

Codechef March 2014 STREETTA

【试题大意】

有 N 家商店标号为 1 到 N，进行 M 次操作。需要支持三种类型：

1. 区间商店的税收加等差数列。
2. 区间商店增加一个纪念品，价格为等差数列。
3. 询问一个商店能最高能消费多少钱。

数据范围：$N \leq 10^9, M \leq 300000$。

【算法介绍】

这道题我们可以用线段树解决。

对于操作一，非常显然等差数列加一个等差数列还是一个等差数列。所以直接打上静态标记表示这一段区间整段加的等差数列，询问的时候直接单点询问即可。

对于操作二，我们可以对每一段区间，存上一条线段表示上凸壳上的一条边。每次加入一条边的时候，只要对于它所对应的线段树上 $\log(N)$ 段区间加入这条线段。这次将新加入的线段和线段树上本身存在的线段对交点进行判断。将出现在上凸壳上较多的线段保留，剩下的到线段树上的儿子里继续更新。

可以用离散或者动态开内存解决 N 过大的情况。

时间复杂度是 $O(M \log^2 M)$，空间复杂度是 $O(M)$。

1.35 Graph Challenge

【试题来源】

Codechef Feb 2014 DAGCH

【试题大意】

给出一张 N 个点 M 条边的图，每个点的编号变成对该图进行 DFS 遍历时的 Dfn。

定义一个点 u 对点 v 是好的，当且仅当存在一条 u 到 v 的路径，且中间点都要大于 v， u 小于 v。定义一个点 u 对点 v 是最好的，当且仅当 u 是所有对 v 点好的点中编号最小的。

进行 Q 次询问，每次给出 x，询问图上有多少点以 x 为最好的点。

数据范围：$N, M, Q \leq 10^5$。

【算法介绍】

这是一道与图的必经点有关的题目。

可以发现，对应于 $Dominator\ Tree$ 算法中,点 u 是点 v 的最好的点，就说明 u 是点 v 的半必经点，这可以由半必经点的定义得到。

所以我们只要对图做 $Dominator\ Tree$，统计每个点是多少个点的半必经点即可。

在 $Dominator\ Tree$ 算法中，我们需要求出原图的 DFS 树，但是题目给出的图已经不是原图了。我们可以将读入的边做如下转换。对于每个点的所有出边，按照编号从小到大排序，之后再做一遍 DFS 就可以得到 DFS 树了。

时间复杂度是$O(N\alpha N)$，空间复杂度是$O(M)$。

1.36 Count on a Treap

【试题来源】

Codechef Feb 2014 COT5

【试题大意】

先给出 $treap$ 的定义: $treap$ 根据 key 是一颗二叉搜索树，根据 $weights$ 是一个堆。

给出 N 次操作，你需要维护一颗 $max-treap$，(weights大的为根)。支持以下操作：

1. $0, k, w$：插入一个新节点， key 值为 k， $weights$ 为 w。
2. $1, k$：删除 key 值为 k 的点
3. $2, ku, kv$：输出 key 为 ku 和 kv 两点在 $treap$ 中的距离。

保证任意两点 $key, weights$ 不相同。保证每次询问都合法。

数据范围：$N \leq 200000, key, weights \leq 2^{32}$。

【算法介绍】

如何计算两点 u, v 之间的距离呢？我们可以这样表示两点的距离:

$$Dis(u,v) = Dep_u + Dep_v - 2 * Dep_{lca(u,v)}$$

我们发现 $lca(u,v)$ 其实就是键值在 $[ku, kv]$ $(ku < kv)$ 中且权重最大的点。这非常容易用线段树来维护。

接下来考虑如何计算一个点 u 的深度。考虑 $treap$ 的生成方式，每次挑一个权重最大的作为根，然后根据键值大小分成左右两子树，并作为当前点的左右儿子。

反过来看，一个点 u 的祖先必定是随着深度减小且权重越来越大。于是我们将所有点按照键值排序，一个点 w 如果是 u 的祖先必然满足键值为 $[kw, ku]$ $(kw < ku)$ 中的所有点，权重都小于 w 点的权值。对于 $kw > ku$ 的情况可以类似的表示。

所以一个点 u 的祖先们其实就是一个以键值为下标，权重递增的最长序列，而点 u 的深度也就是这个序列的长度。现在考虑如何维护区间的最长递增序列。我们可以用线段树来维护这个东西。

记 $calc(l, r, start)$ 表示区间 $[l, r]$ ，初始值为 $start$ 的最长递增序列。记 $len = calc(l, r, 0)$ 。记 $max[l, r]$ 为区间 $[l, r]$ 的最大权重，考虑两种情况的转移：

1.如果 $start \leq lch.max$ ，则：

$$calc(l, r, start) = calc(lch.l, lch.r, start) + len - lch.len$$

2.如果 $start > lch.max$ ，则：

$$calc(l, r, start) = calc(rch.l, rch.r, start)$$

于是乎， $calc(l, r, 0)$ 可以在 $O(\log^2 N)$ 时间里维护，这也就是我们想要的答案。

时间复杂度$O(N \log^2 N)$，空间复杂度$O(N)$。

1.37 Counting D-sets

【试题来源】

Codechef JAN 2014 CNTDSETS

【试题大意】

给定 N, D，询问在 N 维空间中有多少点集满足该点集中最远两个点的距离等于 D。

假设两个点坐标分别是 $(a_1, a_2, ..., a_N)$ 和 $(b_1, b_2, ..., b_N)$。定义这两个点的距离为:

$$max\{|a_1 - b_1|, |a_2 - b_2|, ..., |a_N - b_N|\}$$

显然这样的点集是有无穷个，于是我们定义两个点集是同类点集，当且仅当这两个点集点数相同，且存在一个 N 向量 $(t_1, t_2, ...t_N)$, 使得第一个点集中每一个点加上该向量 t 后等于第二个点集。

请输出有多少不同的同类点集。

数据范围：$N \leq 1000$，$D \leq 10^9$。

【算法介绍】

这是一道很传统的计数问题。

我们考虑将点集中所有点的 N 维坐标都不断减少，直到每一维的最小值都是 0。这个过程可以理解为将点集最小表示，用最小的同类项来表示点集。

这样最小表示以后，一个点集是合法的，当且仅当每一维的最大值都是 D。直接计算最大值为 D 比较麻烦，我们用最大值小于等于 D 的方案数-最大值小于等于 $D-1$的方案。

之后我们再考虑用二项式反演来计算最大值小于等于 D 且每一维最小值都是 0 的方案:

$$Ans(D) = \sum_{i=0}^{N} (-1)^i \binom{N}{i} (2^{(D+1)^{N-i} D^i} - (D+1)^{N-i} D^i - 1)$$

因为最后要求点集里有至少 2 个点，最后的答案就是 $Ans(D) - Ans(D-1)$ 。中途的计算只需要用快速幂和组合数预处理就可以通过本题。

时间复杂度$O(N \log N)$，空间复杂度$O(N)$。

1.38 Counting The Important Pairs

【试题来源】

Codechef JAN 2014 TAPAIR

【试题大意】

给出一张 N 个点 M 条边的图，问有多少对边 (x, y)，满足割掉这两条边图就不连通。

数据范围：$N \leq 10^5$，$M \leq 3 * 10^5$。

【算法介绍】

这是一道有趣的结论题。

我们先用 DFS 跑出这张图的搜索树，对于搜索树上非树边，我们给它随机一个 10^{18} 以内的权值，对于树边，它的权值为横跨这条树边的非树边的权值 xor 和。如果边对 (x, y)，存在一个子集的 xor 和为 0，则说明割掉 (x, y) 后，这张图是不连通的。

证明：要让这张图不连通，非常显然我们要割掉一条树边和所有横跨这条树边的非树边。由于我们之前权值的设定，如果我们割去一对边，这对边权值的 xor 和必然为 0。但是这可能会出现割去后联通但 xor 为 0 的情况。由于我们给每条边随机一个大范围内的权值，可以证明这样的出错概率非常小。

有了上述结论，我们只要进行简单的计数就可以得到答案了。

时间复杂度$O(N \log N)$，空间复杂度$O(N)$。

1.39 Query on a tree VI

【试题来源】

Codechef Dec 2013 QTREE6

【试题大意】

给出一棵 N 个点的树，每个点有黑白两种颜色。初始全部都是白点，进行 Q 次操作，有以下两种类型。

1、将一个点改变颜色。

2、询问一个点 x 所在的同色连通块大小。

数据范围：$N, Q \leq 10^5$。

【算法介绍】

这是一道经典的数据结构题。

首先我们将黑白颜色分别建成两棵树，然后分别对两棵树维护信息。

对于每一个连通块的信息，我们都把它寄存这个联通块里高度最高的点上，称为代表点。这样对于每次询问联通块信息，我们就找到这个点所属连通块的代表点，然后询问那个点的子树信息即可。

现在只需要支持单点修改，询问子树和即可。

这是一个经典的问题，我们可以用树链剖分解决。对于重链上的点，他的权值为该点所有虚儿子的大小和。此时，重链上的一段区间就可以表示一个连通块。

至于如何找代表点，以及代表点重链上最末端同色点。这些都可以用树链剖分解决。

时间复杂度$O(N \log^2 N)$，空间复杂度$O(N)$。

1.40 Petya and Sequence

【试题来源】

Codechef Dec 2013 REALSET

【试题大意】

给出一个长度为 N 的序列 A，请问是否存在一个不是全 0 的序列 B，满足对于任意的 j：

$$\sum_{i=0}^{N-1} A_i B_{(i+j) \bmod N} = 0$$

数据范围：$N \leq 3 * 10^4$。

【算法介绍】

这是一道较难的数学题。

先考虑暴力解决这个问题。我们可以对序列 A 建立循环矩阵 M，现在序列 B 存在的条件就是矩阵 M 不满秩。

我们设:

$$f(x) = A_0 + A_1 x + A_2 x^2 + ... + A_{N-1} x^{N-1}$$

有一个定理就是:

$$rank(M) = N - deg(\gcd(f(x), x^N - 1))$$

其中 $rank(M)$ 表示矩阵 M 的秩， deg 表示多项式的最高项次数， gcd 表示两个多项式的公约式。

我们设 ω_N 为主 N 次单位根，则：

$$x^N - 1 = (x - \omega_N^0)(x - \omega_N^1) * .... * (x - \omega_N^{N-1})$$

定义分圆多项式：

$$\Phi_N(x) = \prod_{gcd(d,N)=1} (x - \omega_N^d)$$

同样有:

$$\begin{aligned} x^N - 1 &= \prod_{i=0}^{N-1} x - \omega_N^d \\ &= \prod_{d|N} \prod_{1 \le k \le N, gcd(k,N)=d} x - \omega_N^k \\ &= \prod_{d|N} \prod_{1 \le k \le N, gcd(k,N)=d} x - \omega_{\frac{N}{d}}^{\frac{k}{d}} \\ &= \prod_{d|N} \Phi_{\frac{N}{d}}(x) \\ &= \prod_{d|N} \Phi_d(x) \end{aligned}$$

因为 $\Phi_N(x)$ 满足不可约且两两不同，所以 $\gcd(f(x), x^N - 1)$ 等价于求有多少 d 满足 $d|N$ 且 $\Phi_d(x)|f(x)$。

考虑到求 $\Phi_d(x)$ 非常麻烦，我们可以做如下变化:

$$f(x) \mod \Phi_d(x) = 0 \Leftrightarrow f(x) \prod_{t=1}^{K} x^{\frac{d}{p_t}-1} \mod x^d - 1 = 0$$

其中 p_t 为 d 的所有约数。

证明如下:

$$f(x)\prod_{t=1}^{K} x^{\frac{d}{p_t}-1} \mod x^d-1 = 0$$

$$f(x)\prod_{t=1}^{K}\prod_{i|\frac{d}{p_t}} \Phi_i(x) \mod \prod_{t|d}\Phi_t(x) = 0$$

观察发现对于除数 $\prod_{d|N}\Phi_d(x)$ 中所有的 d 满足 $d<N$,都可以在分母中被消掉。而对于 $d=N$，被除数中除了 $f(x)$ 的部分是不可能与除数约简，所以这个式子与原式是等价的。

先考虑考虑如何求解 $f(x)\prod_{t=1}^{K} x^{\frac{d}{p_t}-1} \mod x^d-1$。

对于多项式乘法，我们直接暴力做就可以了。对于多项式取模，其实就类比于线性递推中的降幂过程，我们直接将将第 i 项次数的系数加到 $i \bmod d$ 上就可以了。

时间复杂度$O(NKD)$，其中 K 是不同的质约数个数， D 是 N 的约数。空间复杂度$O(N)$。

1.41 Gangsters of Treeland

【试题来源】

Codechef Nov 2013 MONOPLOY

【试题大意】

给定一棵 N 个节点的树，每个点都所属一个帮派。定义一条树边 (u, v) 代价为 1 当且仅当 u 号点和 v 号点所属帮派不一样，否则代价为 0 。定义一个点 x 走到根的代价为所经过树边的代价和。

现在需要进行 Q 次操作，操作一共有两种类型：

1. $O\ x$ ，表示点 x 到根的节点全部被同一个帮派占领。
2. $q\ x$ ，表示询问 x 子树所有点走到根的代价的平均值。

数据范围：$N, Q \leq 100000$。

【算法介绍】

这是一道简单的数据结构题。

对于操作 1 其实就是 LCT 中的 $Access$ 操作。所以很显然这道题用 LCT 去维护。

对于询问，我们可以用线段树在 DFS 序上维护子树和，即询问区间和。每次修改，只要在 LCT 上找到对应的连接点，直接对 DFS 序进行区间 -1 和区间 1 。

时间复杂度$O(N \log^2 N)$，空间复杂度$O(N)$。

1.42 Queries With Points

【试题来源】

Codechef Nov 2013 QPOINT

【试题大意】

给出 N 个多边形，每个多边形有 K_i 个点。保证多边形之间没有交。进行 Q 次询问，每次给出一个点 (x, y)，询问点 (x, y) 在哪一个多边形内。

数据范围：$N, Q \leq 10^5$，$\sum K_i \leq 3 * 10^5$

【算法介绍】

这是一道经典的点定位问题。

因为只有询问没有修改，所以可以采用线段树套 $Vector$ 解决。

我们把每一条多边形的边都按照 X 轴坐标分解到线段树上的 $Vector$ 里。之后对于线段树中每一个节点的 $Vector$，将所有线段排序。

如何求得一个点 (x, y) 在哪个多边形内呢？我们可以用射线法，找出该点向上第一条线段，然后查看这条线段在所对应的多边形内是上半部分还是下半部分。如果是下半部分，则点 (x, y) 不在任一个多边形内；否则就在该多边形内。于是对于询问，我们可以在线段树中的 $Vector$ 里二分。

这道题还有许多的细节，比如说对于刚好在点上，或是刚好在边上。对于垂直于 x 轴的线段我们也要特判掉。

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

1.43 Fibonacci Number

【试题来源】

Codechef Oct 2013 FN

【试题大意】

给出 C, P，求最小 N 满足 : $Fib[N] \mod P = C$。其中 $Fib[N]$ 为斐波那契数列的第 N 项。

数据范围：$P \leq 2 * 10^9$

【算法介绍】

这道题我们考虑用斐波那契数列的通项式求解。

设 $A = \frac{1+\sqrt{5}}{2}$， $B = \frac{1-\sqrt{5}}{2}$，则：

$$Fib[N] = \frac{1}{\sqrt{5}}(A^N - B^N)$$

因为 $A * B = -1$，所以:

$$Fib[N] = \frac{1}{\sqrt{5}}(A^N - (\frac{-1}{A})^N)$$

设 $D = \sqrt{5} * C$。转换下问题，也就是要我们求一个最小的 N，满足:

$$D \equiv A^N - (\frac{-1}{A})^N (\ mod\ P\ )$$

我们对 N 的奇偶性分开讨论,当 N 为偶数时，有:

$$\begin{aligned} D &\equiv A^N - \frac{1}{A^N} (\ mod\ P\ ) \\ A^N D &\equiv A^{2N} - 1 (\ mod\ P\ ) \\ A^N &\equiv \frac{D \pm \sqrt{D^2+4}}{2} (\ mod\ P\ ) \end{aligned}$$

奇数的情况做法类似，于是我们只考虑偶数的情况。

因为涉及在模域下的开根，所以我们求解出 P 的原根，用 $BSGS$ 求解出 $\sqrt{5}$ 以及 $\sqrt{D^2+4}$。之后再用 $BSGS$ 求解出 N。

这道题还需要特判一些被开根数为 0 的情况以及奇偶讨论，涉及到一些细节需要细心实现。

时间复杂度$O(\sqrt{N})$，空间复杂度$O(\sqrt{N})$。

1.44 Two Roads

【试题来源】

Codechef Sept 2013 TWOROADS

【试题大意】

平面上给出 N 个点，要求画两条直线，使得所有点到最近直线的最短距离平方的平均值最小。

数据范围：$N \leq 100$

【算法介绍】

这是一道较麻烦的数学题。

设点 $P(x_0, y_0)$，直线 l 解析式为 $Ax + By + C = 0$。则点 P 到直线 l 的距离平方为:

$$Dis = \frac{(Ax_0 + By_0 + C)^2}{A^2 + B^2}$$

我们先考虑一条直线的情况。设最后答案的直线为 $y = kx + b$，则：

$$\begin{aligned} S &= \sum_{i=1}^{N} \frac{(kx_i - y_i + b)^2}{k^2 + 1} \\ S &= \frac{\sum_{i=1}^{N} k^2 x_i^2 + y_i^2 + b^2 - 2kx_i y_i + 2bkx_i - 2by_i}{k^2 + 1} \end{aligned}$$

其中 x_i, y_i 都是已知的量。我们假设 k 是一个常量，观察 b 的取值情况，有:

$$S = \frac{Nb^2 + b(\sum_{i=1}^{N} 2kx_i - 2y_i) + (\sum_{i=1}^{N} k^2 x_i^2 + y_i^2 - 2kx_i y_i)}{k^2 + 1}$$

容易发现，当 S 取值最小的时候，有：

$$b = -k\overline{x} + \overline{y}$$

将 b 带入原式,通过化简得到:

$$S = \frac{k^2(\sum_{i=1}^{N}(x-\overline{x})^2) + k(2\sum_{i=1}^{N}(x-\overline{x})(y-\overline{y})) + (\sum_{i=1}^{N}(y-\overline{y})^2)}{k^2+1}$$

设：

$$\begin{aligned} A &= \sum_{i=1}^{N}(x-\overline{x})^2 \\ B &= 2\sum_{i=1}^{N}(x-\overline{x})(y-\overline{y}) \\ C &= \sum_{i=1}^{N}(y-\overline{y})^2 \end{aligned}$$

则:

$$\begin{aligned} S &= \frac{Ak^2+Bk+C}{k^2+1} \\ 0 &= (A-S)k^2 + Bk + (C-S) \end{aligned}$$

当该方程有解，要求 $\Delta \geq 0$。有：

$$\begin{aligned} \Delta &= B^2 - 4(A-S)(C-S) \\ \Delta &= -4S^2 + 4(A+C)S + B^2 - 4AC \end{aligned}$$

因为我们要使得 S 最小，容易发现当 $\Delta = 0$，求出的较小的解就是 S 的最小值，所以有：

$$S = \frac{A+C-\sqrt{(A-C)^2+B^2}}{2}$$

这样我们就解决了一条直线的情况。

对于两条直线的情况，容易发现以两条直线的夹角做角平分线，可以把这个区域分成 8 块。其中一条角平分线两边的点分别离不同的两条直线最近。因为这两条角平分线互相垂直，所以就等价于将这个平面分成了 4 个象限。而两个对角的象限里的点各自与不同的直线最近。

于是我们可以枚举这两条角平分线，然后将平面上的点分别找到他的归属直线。用我们上述的一条直线方法求得答案，相加取最小值。

如何枚举呢？容易发现这两条对角线有一条肯定是无穷逼近其中两个点；另外一条肯定卡住一条直线。所以我们枚举其中两个点确定一条直线，再将剩下的点按照垂足排序，线性的扫过去。时时维护两个对角象限的信息即可。

时间复杂度$O(N^3 \log N)$，空间复杂度$O(N)$。

1.45 Music & Lyrics

【试题来源】

Codechef Aug 2013 LYRC

【试题大意】

给出 N 个单词， M 篇文章，问每一个单词在所有文章中一共出现了多少次。

设单词长度为 $L1$ ，文章长度为 $L2$ ，字典集为 62 。

数据范围：$N \leq 500$，$M \leq 100$，$L1 \leq 5000$，$L2 \leq 50000$。

【算法介绍】

这是一道简单的字符串处理题。

我们可以对所有单词建立 AC 自动机以及 $Trie$ 图。之后只要将所有文章在 $Trie$ 上遍历即可。

设单词总长为 $S1$ ，文章总长为 $S2$ 。

时间复杂度$O(62 * S1 + S2)$，空间复杂度$O(62 * S1)$。

1.46 Prime Distance On Tree

【试题来源】

Codechef Aug 2013 PRIMEDST

【试题大意】

给出一棵 N 个点的树，求随机选一对点 u,v ，u 到 v 的路径长度为质数的概率。

数据范围：$N \leq 10^5$。

【算法介绍】

这是一道经典的点分治+ FFT 问题。

考虑点分治，设当前重心为 x ，求出当前连通块中所有点到 x 的距离，用 FFT 计算卷积。在同一棵子树内的路径会算重，可以对每棵子树单独做一次 FFT 卷积，把重复部分剪掉。 FFT 的长度与该子树深度相关，所以可以保证FFT的总长度为 $O(N \log N)$ 。

时间复杂度$O(N \log^2 N)$，空间复杂度$O(N)$。

1.47 Across the River

【试题来源】

Codechef July 2013 RIVPILE

【试题大意】

平面上有 N 个点，可以在每个点上画一个圆。一共有 M 种类型的圆，每种圆的半径为 R_i，代价为 C_i。

需要只通过圆上的点，从直线 $y = W$ 走到直线 $y = 0$，要求最小代价。

数据范围：$N, M \leq 250$。

【算法介绍】

这是一道简单的最短路问题。

非常朴素的想法，我们可以用最短路来解决这个问题。

把每个点上画某一种圆都当做一个点，如果两个之间可以通过两个圆相接，则他们之间连上相应代价的边。

这样构图的点数为 $O(NM)$，边数为 $O(N^2M^2)$。但是这样肯定会 TLE。

可以这样优化构图。将所有圆按照权值排序，相应的我们要求值保留其中半径递增的圆。贪心的想，如果点 i 画上第 a 种圆，在 j 画 b 种圆，两个点刚好卡住. 那我们就只在之间连边，不向半径比 b 大的圆连边。之后对于点 a 的第 j 种圆和第 $j+1$ 种圆连上对应的权值即可。

还有一个小优化，就是当两个点连边时，如果连边导致的效果不变，则优先选择权值小的边。构图完成后，直接用 Dij 跑最短路就可以了。

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

1.48 Two k-Convex Polygons

【试题来源】

Codechef June 2013 TKCONVEX

【试题大意】

给出 N 条线段，长度为 A_i。要求选出两组包含不同的 K 条线段，使得两组线段都能组成凸多边形。

数据范围：$N \leq 1000$，$K \leq 10$。

【算法介绍】

这是一道结论题。

结论：一组线段能够组成凸多边形的条件就是最长的线段小于剩余线段的长度总和。

如果只选出一组凸多边形，贪心的想，肯定是将线段按照长度排序后，选取连续的一段区间。

对于选出两组的情况。我们分情况讨论：如果两组线段没有交，则我们 $O(N^2)$ 去枚举两段区间，然后验证。如果有交，那肯定是选择连续的一段长度为 $2K$ 的区间。我们强制最长的线段在第一组里，对于剩下 $2K-1$ 条线段，我们进行搜索来确定在哪个组里，验证即可。

还需要一定的卡常数技巧。

时间复杂度$O(N^2 + NK\binom{2K-1}{K})$，空间复杂度$O(N)$。

1.49 Count Special Matrices

【试题来源】

Codechef June 2013 SPMATRIX

【试题大意】

给出 N，要求计算有多少满足要求的矩阵 A。其中A是一个N*N的整数矩阵。第x行第y个元素记作A[x][y]。这个矩阵如果满足以下条件就称它满足要求：

A[x][x] = 0 for 1 ⩽x ⩽N.

A[x][y] = A[y][x] > 0 for 1 ⩽x < y ⩽N.

A[x][y] ⩽max(A[x][z], A[z][y]) for 1 ⩽x, y, z ⩽N.

A[x][y] ∈1, 2, ..., N-2 for 1 ⩽x < y ⩽N.

任意k ∈1, 2, ..., N -2 存在x, y ∈1, 2, ..., N 满足A[x][y] = k.

T 组数据。

数据范围：$T \leq 10^5$，$N \leq 10^7$。

【算法介绍】

这是一道结论题。

由于读入数据只有一个整数 N，可以想到进行小范围打表，再用 $OEIS$ 查表得到序列。经查表得:

$$Ans = \frac{N!(N-1)!}{2^{N-1}}(\frac{N}{2} - \frac{2}{3} - \frac{H_{N-1}}{3})$$

其中 $H_N = \sum_{i=1}^{N} \frac{1}{i}$

时间复杂度$O(N)$，空间复杂度$O(N)$。

1.50 Queries on tree again!

【试题来源】

Codechef May 2013 QTREE

【试题大意】

给定一棵 N 个节点的基环树，定义两点间的路径为经过点数最少的路径。进行 Q 次操作，需要包含两种类型：

1. 把 u, v 路径上的边权乘上 -1 。
2. 询问 u, v 路径上边权的最大子段和。

数据范围：$N, Q \leq 10^5$。

【算法介绍】

这是一道经典的数据结构题。

我们先考虑一个简化的问题，即题目所给的是序列而不是基环树。这是非常经典的线段树问题，我们只需维护一个区间的 $sum, Min, Max, LMax, LMin, RMax, RMin$ 即可轻松得到答案。

假使题目给出的是一棵树。很容易联想到用树链剖分或者 LCT 来维护。但是这是一棵基环树，我们可以做如下的变化。找出环，然后任意选择一条边断开，这样就变成了一棵树。我们把边的信息寄存在点上。每次寻找路径可以用简单的特判知道是否要走那条断边。另外只要注意细节，大力码就可以了。

时间复杂度$O(N \log^2 N)$，空间复杂度$O(N)$。

1.51 Little Elephant and Colored Coins

【试题来源】

Codechef March 2013 LECOINS

【试题大意】

有 N 种硬币，每种硬币有价值 V_i，颜色 C_i，并且有无穷多个。

进行 Q 次询问，每次给出 S，要求选若干硬币，使得价值和等于 S。要求所有硬币的颜色种类尽量多。

数据范围：$N \leq 30$，$Q, V_i \leq 200000$，$S \leq 10^{18}$。

【算法介绍】

这道题我们可以用动态规划解决。

记 $F[i][j]$ 表示用了 i 种颜色，能拼成的价值和在模 V 域下为 j 的最小价值和为 $F[i][j]$。其中 V 为 $min(V_i)$。每次询问 S，我们只需要找出最大的 k，满足 $F[k][S \bmod V] \leq S$ 即可。

现在考虑如何得到 F。我们可以将颜色相同的硬币一起做，有如下转移:

1. $F[d][j] + V_i \rightarrow F[d+1][(j+V_i) \bmod V]$。
2. $F[d][j] + V_i \rightarrow F[d][(j+V_i) \bmod V]$

对于第一次选取这种颜色时，用第一种转移方程，否则用第二种转移方程。

时间复杂度$O(N^2V + NQ)$，空间复杂度$O(NV)$。

1.52 Making Change

【试题来源】

Codechef March 2013 CHANGE

【试题大意】

给定 N 个非负整数 d_i，求解不定方程：

$$\sum_{i=1}^{N} x_i d_i = C$$

该方程的非负整数解个数。保证所有 d_i 各不相同，且两两互质。

数据范围：$N \leq 50$，$D_i \leq 500$，$C \leq 10^{100}$。

【算法介绍】

这是一道较麻烦的数学题。

设集合 D 为包含所有 d_i 的集合。其实题目让我们求解的就是：

$$\prod_{d \in D} \frac{1}{1-x^d}[x^C]$$

先给出一些式子来方便后面的推导。设：

$$T_d(x) = \sum_{i=0}^{d-1} x^i$$

$$\begin{aligned} 1-x^d &= (1-x)\prod_{i=1}^{d-1}(x-\omega_d^i) \\ 1-x^d &= (1-x)T_d(x) \\ T_d(x) &= \prod_{i=1}^{d-1}(x-\omega_d^i) \end{aligned}$$

因为 d_i 两两互质，所以 $1-x^{d_i}$ 之间的公约式只有 $(1-x)$。

再给出 $\frac{1}{1-\omega_d}$ 的计算方法，令：

$$\begin{aligned} A &= \sum_{i=0}^{d-1}(i+1)x^i \\ xA &= \sum_{i=0}^{d} ix^i \\ (1-x)A &= \sum_{i=0}^{d-1}x^i - dx^d \end{aligned}$$

因为 $\sum_{i=0}^{d-1}\omega_d^i = 0$，所以：

$$\begin{aligned} (1-\omega_d)A &= -d \\ \frac{1}{1-\omega_d} &= -\frac{1}{d}\sum_{i=0}^{d-1}(i+1)(\omega_d^i) \\ \frac{1}{1-\omega_d} &= -\frac{1}{d}\sum_{i=0}^{d-1}i(\omega_d^i) \end{aligned}$$

接下来我们来解决这道题。

则：

$$\begin{aligned}\prod_{d\in D}\frac{1}{1-x^d} &= \prod_{d\in D}\frac{1}{(1-x)(\sum_{j=0}^{d-1}x^j)}\\ &= (\frac{1}{1-x})^N\frac{1}{\prod_{d\in D}T_d(x)}\\ &= \frac{A(x)}{(1-x)^N}+\sum_{d\in D}\frac{B_d(x)}{T_d(x)}\end{aligned}$$

这样我们就把若干等式用部分分式分解了。现在我们只需要对每一个部分分式计算出答案，然后相加即可。

先考虑如何计算 $B_d(x)$，将原式两边乘上 $T_d(x)$。

$$\begin{aligned}\prod_{d\in D}\frac{1}{1-x^d} &= \frac{A(x)}{(1-x)^N}+\sum_{d\in D}\frac{B_d(x)}{T_d(x)}\\ \frac{1}{1-x}\frac{1}{\prod_{d'\in D-d}(x-x^{d'})} &= B_d(x)+E(x)*T_d(x)\end{aligned}$$

因为另外式子都不会存在公约式，所以上式成立。若将 ω_d 带入 $B_d(x)$ 中，则 $T_d(x)$ 为 0，即：

$$B_d(\omega_d)=\frac{1}{(1-\omega_d)*\prod_{d'\in D-d}(1-\omega_d^{d'})}$$

因为上述关于 $\frac{1}{1-\omega_d}$ 的等式，所以有：

$$B_d(\omega_d)=(-\frac{1}{d})^N(\sum_{i=0}^{d-1}i\omega_d^i)\prod_{d'\in D-d}\sum_{i=0}^{d'-1}i*\omega_d^{i*d'}$$

所有的下标都要对 d 取模，容易发现这就是一个卷积的式子，朴素的做效率是 $O(ND^2)$。然而这样会 TLE，我们考虑这种卷积的特殊性来优化效率，令：

$$\sum_{i=0}^{d-1}a_ix^i*\sum_{i=0}^{d-1}i*\omega_d^{i*d'}=\sum_{i=0}^{d-1}b_ix^i$$

对于 b_i 有：

$$b_i = \sum_{k=0}^{d-1} k * a_{i-k*d'}$$

而对于 $b_{i+d'}$ 有：

$$b_{i+d'} = \sum_{k=0}^{d-1} k * a_{i+d'-k*d'}$$

这个过程可以理解为 $b_{i+d'}$ 由 b_i 继承过来。但是原先贡献 $k=1$ 的变成贡献 $k=2$， $k=2$ 的贡献 $k=3$,...,而原先贡献 $k=d-1$ 的贡献 $k=0$。所以我们需要加上所有 a_i，然后减去多加的即可。于是这个卷积的复杂度是 $O(Nd)$，因为对于每一个 d_i，我们都要计算 $B_d(x)$ 所以，总的复杂度是 $O(Nd^2)$。

此时我们已经得到了 $B_d(\omega_d)$，因为：

$$\sum_{i=0}^{d-1} \omega_d^i = 0$$

所以：

$$B_d(\omega_d) = \sum_{i=0}^{d-2} (b_i - b_{d-1})\omega_d^i$$

这就是一个 $d-1$ 阶的多项式。容易发现当我们将 $\omega_d^j, j \in [1, d-1]$ 带入到 $B_d(x)$ 中，所有的 $B_d(\omega_d^j)$ 的系数均为此。因为点值都符合要求，所以可以断言：

$$B_d(x) = \sum_{i=0}^{d-2} (b_i - b_{d-1})x^i$$

考虑 $\frac{B_d(x)}{T_d(x)}$ 对答案的贡献，观察发现：

$$\begin{aligned} \frac{B_d(x)}{T_d(x)} &= B_d(x) * (1-x) * \frac{1}{1-x^d} \\ \frac{B_d(x)}{T_d(x)} &= B_d(x) * (1-x) * \sum_{i=0}^{\infty} x^{id} \end{aligned}$$

因为 $B_d(x) * (1-x)$ 是一个 d 阶多项式，所以我们直接取$B_d(x) * (1-x)$中 $x^{C \bmod d}$ 处的系数就是答案。

再考虑 $\frac{A(x)}{(1-x)^N}$ 对答案的贡献。

$$\begin{aligned}\frac{1}{(1-x)^N} &= (1+x+x^2+x^3+x^4+....)^N \\ &= \sum_{i=0}^{\infty} \binom{N+i-1}{N-1} x^i\end{aligned}$$

易得 $\frac{1}{(1-x)^N}$ 的第 x^i 项的系数是一个关于变量 i 的 N 阶多项式。而 $A(x)$ 的阶数小于 N 阶，所以 $\frac{A(x)}{(1-x)^N}$ 的阶数小于等于 $2*N-1$。因为 N 范围较小，所以我们直接暴力算出前几项，然后用拉格朗日插值即可。这一步时间效率为 $O(N^2)$。

时间复杂度$O(Nd^2)$，空间复杂度$O(Nd)$。

1.53 Room Corner

【试题来源】

Codechef FEB 2013 ROC

【试题大意】

给出 N 行 N 列的网格图，描述一个房间。房间内每一个角落中都有一个小孩。

每次相邻的小孩之间可以交换位置。进行 T 次询问，每次询问 u, v 两个小孩通过交换位置最少多长时间可以相遇。

数据范围：$N \leq 2500$，$Q \leq 10000$。

【算法介绍】

这是一道有趣的模拟题。

容易发现小孩的站位和交换的法则就对于一个环，相邻两个小孩之间的距离为房间中的距离。于是可以通过暴力模拟得出这个环，再将环拉长成两倍。

对于每次询问 u, v，可以认为是找出最小的 x，使得：

$$Time = max(Dis(x, u) + Dis(x, x + 1)/2, Dis(x + 1, v) + Dis(x, x + 1)/2)$$

$Time$ 最小。这个非常显然可以通过二分得到。

由于题目给出的性质，每一行中的拐点数至多只有 2 个，所以拐点数是 $O(N)$ 级别的。

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

1.54 Observing the Tree

【试题来源】

Codechef FEB 2013 QUERY

【试题大意】

给定一棵 N 个节点的树，进行 Q 次询问，需要支持以下几种类型:

1、链加等差数列。

2、询问链和。

3、回到之前某次操作的状态。

强制在线。

数据范围：$N, Q \leq 100000$。

【算法介绍】

这是一道经典的数据结构题。

由强制在线和操作三可以看出，这道题需要我们用可持久化数据结构来解决。操作一和操作二是非常基础的树链剖分问题。所以只需可持久化树链剖分就可以解决这道题。

接着只需要考虑一些细节，大力码就可以了。

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

1.55 A New Door

【试题来源】

Codechef JAN 2013 ANDOOR

【试题大意】

给出 N 个圆和一个矩阵，求 N 个圆的外轮廓并和矩阵交的长度。

数据范围：$N \leq 1000$。

【算法介绍】

这是一道经典的计算几何题。

这个问题等价于求出 N 个圆不被其他圆覆盖且在矩阵里面的长度。

我们可以枚举每一个圆，然后用角度 $[0, 2\pi]$ 表示一个圆。对于每一个另外的圆，我们将两者的交的角度区间删掉。对于矩阵，我们把这个圆超过矩阵外的部分割掉，具体可以看成是用矩阵的四条边去割圆。两者都可以看做是将圆的一段圆弧割掉。最后只需要统计这些被删弧度区间的并，然后用总数减一减就可以得到这一个圆对答案的贡献。

剩下的就是简单的计算几何，注意公式的运用和一些细节的处理，仔细模拟即可。

时间复杂度$O(N^2 \log N)$，空间复杂度$O(N)$。

1.56 Cucumber Boy and Cucumber Girl

【试题来源】

Codechef JAN 2013 CUCUMBER

【试题大意】

给出 M 个 $N*N$ 的矩阵 Q_i。对于每对 $(a,b), 1 \le a < b \le M$，定义矩阵 $C_{a,b}$ 为：

$$C_{a,b}[i][j] = \sum_{k=1}^{N} Q_a[i][k] * Q_b[j][k], 1 \le i,j \le N$$

对于一个 $1,2,..,N$ 的排列 p，定义序列 $Tr(p)[i] = C_{a,b}[i][p_i], 1 \le i \le N$，

对于 $Cnt(a,b)$ 为所有的排列 p 中，至少存在一个 i，满足 $Tr(p)[i]$ 为奇数的排列个数。

定义 (a,b) 为好的，当且仅当 $Cnt(a,b)$ 为奇数。请计算有多少好的对。

数据范围：$N \le 60$，$M \le 8000$。

【算法介绍】

这是一道线性代数题。

如果 $N = 1$，我们可以用 $O(M)$ 的线性扫一遍得到答案。

对于 $N > 1$,我们先考虑用暴力来解决这个问题。

我们可以枚举 (a,b)，然后验证这对是不是好的。考虑如何计算 $Cnt(a,b)$。我们可以计算出不存在奇数元素的排列数，然后用 $N!$ 减去得到至少存在奇数元素的排列数。

这个可以用行列式来解决，我们设 $D_{a,b}[i][j] = (C_{a,b}[i][j] + 1)\ mod\ 2$。因为我们只关心 $Cnt(a,b)$ 的奇偶性，所以可以把整个矩阵在模 2 域下做。因为 $N > 1$，所以我们可以无视 $N!$ 这一项。容易发现 $Det(D_{a,b}) = 1$ 时，(a,b) 就是好的。

我们将所有的 a, i ,使得 $Q_a[i][N+1] = 1$,可以对 D 做如下变换：

$$\begin{aligned} D_{a,b}[i][j] &= (C_{a,b}[i][j] + 1)\ mod\ 2 \\ &= \sum_{k=1}^{N} Q_a[i][k] * Q_b[j][k] + 1\ mod\ 2 \\ &= \sum_{k=1}^{N+1} Q_a[i][k] * Q_b[j][k]\ mod\ 2 \end{aligned}$$

非常显然:

$$Det(D_{a,b}) = Det(Q_a^T * Q_b)$$

运用 $Cauchy - Binet$ 定理和 $Det(Q_a) = Det(Q_a^T)$,有：

$$Det(D_{a,b}) = \sum_{c=1}^{N+1} Det(Q_{a,c}) * Det(Q_{b,c})\ mod\ 2$$

其中 $Q_{a,c}$ 表示矩阵 Q_a 删去第 c 列的矩阵。

所以我们只要对于所有 Q_a 求出 $Q_{a,c}$，通过简单的位运算我们就可以得到 $Det(D_{a,b})$，也就可以得出答案。

关于求解 $Q_{a,c}$ 可以用非常经典的高斯消元法解决。

时间复杂度$O(\frac{N}{32}(M^2 + MN^2))$，空间复杂度$O(\frac{N}{32}MN)$。

1.57 Different Trips

【试题来源】

Codechef DEC 2012 DIFTRIP

【试题大意】

给出一棵 N 个节点的有根树，定义一个点的权值为该节点的度数。每一条从点 u 到点 v（v 点为 u 点树上的祖先），都是一条合法的路径。如果两条路径按顺序的点权值都相同，则认为两条路径相似。问整棵树有多少不同的路径。

数据范围：$N \leq 100000$。

【算法介绍】

这是一道非常经典的广义 SAM 题。直接套用经典的模板即可。

注意本题中每个点的权值为他的度数，容易发现度数的种类不会超过 $\sqrt{N}$ 个，种类最多的情况是一个类似等差数列的东西。

于是可以用 Map 记录 SAM 的 $Trans$，或者离散之后做。

时间复杂度$O(N\sqrt{N})$，空间复杂度$O(N\sqrt{N})$。

1.58 Arithmetic Progressions

【试题来源】

Codechef Nov 2012 COUNTARI

【试题大意】

给定一个长度为 N 的序列 A。求解有多少三元组 (i,j,k) $(i<j<k)$ 满足;

$$A_j - A_i = A_k - A_j$$

数据范围：$N \leq 100000, A_i \leq 30000$。

【算法介绍】

这道题我们用分块+ FFT 解决。

先转化一下限制，其实题目就是要我们求有多少 (i,j,k) ，满足：

$$2 * A_j = A_i + A_k 且 i<j<k$$

如果没有 $(i<j<k)$ 这个限制,那这是一道非常基础的 FFT 的问题。那如何解决这个限制呢？我们可以尝试用分块来优化。

我们将序列分成 S 块，对于每一块，我们将这块前面的所有数和这块后面的所有数做一个卷积，然后就可以得到一个数在块内，两个数不在块内的结果。效率为 $O(\frac{N}{S} * M \log M)$, M 为最大的权值。

对于两个数在块内和三个数都在块内的情况，我们可以直接暴力枚举其中两个在块内的数，另外一个数可以直接在数组里询问个数。效率为 $O(NS)$ 。

效率最优的时候为:

$$\frac{N}{S} * M \log M = NS$$

所以取 $S = \sqrt{M \log M}$ 。

时间复杂度$O(N\sqrt{M \log M})$，空间复杂度$O(N)$。

1.59 Martial Arts

【试题来源】

Codechef Nov 2012 MARTARTS

【试题大意】

给出两支各有 N 个球员的球队，要求安排 N 场比赛，使得每个球员都参加。

给出 $A_{i,j}$ 和 $B_{i,j}$，表示第一支球队第 i 个球员和第二支球队第 j 个球员比赛时，比分为 $A_{i,j} : B_{i,j}$。

设第一支球队得分为 H，第二只球队得分为 G。第一支需要安排这 N 场比赛使得，$H - G$ 最大，在相同情况下，使 H 尽量大。

当比赛安排好之后，第二只球队可以选择删去一组比赛，使得 $H - G$ 尽量下，相同情况下使 G 尽量大。

需要回答第一支球队应该如何安排比赛使得目标越大。

数据范围：$N \leq 100$。

【算法介绍】

这是一道带权匹配问题，采用 KM 解决。

设 $W_{i,j} = A_{i,j} - B_{i,j}$。将 (i, j) 之间的关系看成是边，这个问题就可以等价于一个最大权匹配的问题。

我们考虑如果 N 场比赛已经安排好了。第二支球队肯定选择删去 $W_{i,j}$ 最大的，相同情况下删去 $B_{i,j}$ 最小的（当然如果 W 为负数时就不用删除了）。

所以我们可以将所有边按照关键字排序，之后枚举强制删去某条边。如果第二支球队需要删除这条边，这说明第一支球队肯定只能选择比当前边关键字劣的边来

进行匹配。

如果对于每次匹配我们都重新用 KM 匹配一遍，我们就可以得到一个$O(N^5)$的算法。

然而这样太慢了，我们需要优化匹配的算法。容易发现，如果我们把所有边开始都视为 $-INF$，接着枚举每条边。我们强制要删掉这条边，要求这条边一定要匹配中，即将这条边赋值为 INF，统计总和后更新答案。接着把这条边赋值为 $W_{i,j}$，留在图中即可。

这样问题就变成了：每次修改一条边的边权，询问最大权匹配。这个问题其实有效率更加优秀的做法。每次修改 (i,j) 之间边权，可以将 KM 算法中的 i 和 $Match[i]$ 取消匹配, 将 $LabelA[i] = max_k\{W_{i,k} - LabelB[k]\}$，之后增广一次即可。

实际实现时，我们可以倒着做来进行一些最优性剪枝。

对于本题中边有两个关键字，我们可以用一个 $pair$ 类型表示一条边权，注意细节处理即可。

时间复杂度$O(N^4)$，空间复杂度$O(N^2)$。

【备注】

【关于KM算法】

KM 可以解决一类二分图带权匹配问题。我们给二分图中两排点分别一个标号，记为 $La[i], Lb[i]$。我们要满足对于任意的 i, j， $La[i] + Lb[j] \geq W[i][j]$。其中 $W_[i][j]$ 为这两点之间的权值。

对于每一个点我们都去增广，直到找到匹配点为止。一条边可行当且仅当 $La[i] + Lb[j] = W[i][j]$。如果利用当前的边不能给 i 找到匹配点，则我们就考虑修改边权来增加可行边。

可以将上一次匹配过程中被访问过的点分别标记，记为 $Va[i], Vb[i]$，当值为 1 时表示被访问过。我们取最小的 $Min = La[i] + Lb[j] - W[i][j]$，满足 $Va[i] =$

$1, Vb[j] = 0$。然后将 $Va[i] = 1$ 的 $La[i] -= Min$，$Vb[j] = 1$ 的 $Lb[j] += Min$。这个过程肯定可以使得增广路增多，最后所有点都匹配成功则找到一组最大权匹配。

总效率 $O(N^3)$。

1.60 Max Circumference

【试题来源】

Codechef Oct 2012 MAXCIR

【试题大意】

平面上给出三个点 A, B, C，和 N 个操作 X_i, Y_i，需要从中选出小于等于 K 个操作使用，使用操作 i 可以使 A 点的 x 坐标和 y 坐标分别增加 X_i, Y_i，使得 ΔABC 周长最长。

要求答案绝对误差不超过 10^{-12}。

数据范围：$N \leq 500$。

【算法介绍】

这是一道较麻烦的数学题。

非常显然，肯定存在一组实数 U, V，满足当点 A 坐标为 (x, y) 时，$f(x, y) = Ux + Vy$ 最大时，同样满足三角形周长最长。

于是我们可以令每个操作 i 的权值设为 $f(X_i, Y_i)$，这样我们只要取出前 K 大的权值中正数的操作就可以知道 A 点的坐标了。

但是 U, V 的取值范围是整个实数域，这显然不能做。但是我们发现，对于两个操作 i, j，我们把 (X_i, Y_i) 和 (X_j, Y_j) 看做两个向量，(U, V) 同样也是一个向量。那么 $f(X_i, Y_i)$ 与 $f(X_j, Y_j)$ 的大小关系，就相当于向量运算中两个向量与另一个向量点积的大小关系。这在平面图上，其实可以用一条直线来划分，当 (U, V) 在这条直线上时，$f(X_i, Y_i) = f(X_j, Y_j)$，而对于这条直线划分出来的两个半平面，当 (U, V) 在其中一个中时，就可以得到两者的大小关系。因为这道题还要涉及到权值

的正负关系，所以还需要将每个操作与 0 的划分线插入。而这条直线可以通过解方程，非常便利的解出来。

这也就引导我们，如果我们将这些操作两两之间划分的直线求出来，就会把这个平面分成若干部分。而每一个部分中,每一个操作权值的大小关系都是一样的。所以我们可以枚举每一个部分，从中任意选出一个向量作为 (U,V) ,然后将所有操作的权值从大到小排序，取出前 K 个正数的操作相加就是答案。

这样时间效率是 $O(N^3 \log N)$ ，显然会 TLE 。但是我们发现，我们将这些划分的直线按照极角排序，依次对每一个部分做。转到下一个部分时，只有两个操作的相对位置会变化。这里我们只要暴力去做就可以了。之后可以进行二分计算答案。

然而这道题对精度要求非常高，朴素的做法是通过不了本题的。我们需要很多特殊的优化才能通过此题。

时间复杂度$O(N^2 \log N)$，空间复杂度$O(N^2)$。

1.61 Knight Moving

【试题来源】

Codechef Sept 2012 KNGHTMOV

【试题大意】

在平面的 $(0,0)$ 处有一个骑士，需要走到 (Tx,Ty) 处。每次骑士可以从当前位置 (x,y) 走到 $(x+Ax,y+Ay)$ 或者 $(x+Bx,y+By)$ 点。

再给出 N 个障碍点 (X_i,Y_i) 不能走，问有多少种不同的方案数。

数据范围：设 d 为坐标范围，$N\leq 15$，$d\leq 500$。

【算法介绍】

这是一道模拟题，我们分两种情况讨论：

一.如果 (Ax,Ay) 与 (Bx,By) 线性无关，则平面上给出的每一个点都可以被唯一表示成 $u*(Ax,Ay)+v*(Bx,By)$，我们把 (u,v) 当做这个点的坐标。发现排序后只要进行简单的 $O(N^2)$ 容斥即可。

二.如果 (Ax,Ay) 与 (Bx,By) 线性相关，则所有可以达到点都在一条直线上。我们将所有既可以从 $(0,0)$ 到达且可以到达 (Tx,Ty) 的点拿出来。将连边连上，发现这就是一张图模型。如果这张图中出现环，则肯定是无穷解，否则他就是一张拓扑图，这样我们只要进行拓扑排序，然后简单 DP 即可。容易证明，如果这张图出现环，要么肯定在坐标范围小于等于 d^2 就出现；如果没有出现环，那么走到 d^2 以外的点是没有必要的，因为肯定走不到终点。所以可以用 $O(d^2)$ 的算法解决。

对于无解和无穷的情况还需要一些特判才能通过。

时间复杂度$O(N^2+d^2)$，空间复杂度$O(d^2)$。

1.62 Annual Parade

【试题来源】

Codechef Sept 2012 PARADE

【试题大意】

给定一张 N 个点 M 条边的图，进行 Q 次询问，每次给出一个 C 。要求在图上画一些路径，这种方案的权值定义为下列三项的和：

1、所有路径的边权之和。

2、如果一个点没有被覆盖，则贡献 C 。

3、如果一条路径没有成环，则贡献 C 。

求权值最小的方案数。

数据范围：$N \leq 250$，$M \leq 30000$，$Q \leq 10000$。

【算法介绍】

这道题我们用费用流解决。

我们这样转化问题。如果我们给每一个点一条边权为 C 的自环，那么就可以去掉第 2 项了。接下来我们用最短路算法得到两两之间的最短路，记 i 点到 j 点的最短距离为 $A[i][j]$ 。

以该方式建立费用流模型：建立两排点，左边第 i 号点向右边第 j 号点连费用为 $A[i][j]$ 的边。源汇各自连完边之后，我们求出这张图的一个最小费用最大流。

观察这种方案来对应原图，可以发现，每个点的入度出度均为 1 ,且边数点数均为 N 。这就对应出原图中有许多环。且这种解一定是合法且最优的。

那么如何解决 C 呢？非常朴素的想法，我们在做费用流的过程中，会不断找

当前权值和最小的增广路，这也就意味着每次会多匹配一对点。如果当前的增广的权值大于 C，要么我们还不如不增广，直接用 C 的权值将这个点自我匹配即可。

费用流存在一个性质，即每次增广的权值都是单调递增，且是固定的。所以我们可以直接先用费用流预处理出每次增广的权值，对于每次询问，直接用上述方案询问即可。

时间复杂度$O(costflow(N, N^2) + QN)$，空间复杂度$O(N^2)$。

1.63 Two Magicians

【试题来源】

Codechef AUG 2012 MAGIC

【试题大意】

给定一张 N 个点 M 条边的无向图。开始有两个人分别在 1 号点和 2 号点，初始均为 P 点法力。

这是个回合制游戏，由第一个人先开始，每个人的回合，他都需要按顺序做以下几件事：

1. 通过现有的边任意走动，如果走到另一个人的点则游戏结束，该人获胜。

2. 必须添加一条现在没有的边，如果无法添加则视为对方胜利。

3. 如果还有法力，可以选择扣除一点法力，之后转移到任意一个点上。

假设两个都用最优策略，问最后先手还是后手胜利。一共 T 组数据。

数据范围：$N \leq 7777$，$M, P \leq 10^4$，$T \leq 100$。

【算法介绍】

这是一道分类讨论题，可以做如下分析：

容易发现，如果一开始 1 号点和 2 号点就在同一个连通块之内，则先手必胜。否则现在一定有至少两个连通块，容易发现最后一个连接两个联通块的人必输，所以两个人的最优策略均为让对方联通最后两个联通块。

设 E 为剩下还没有连接的边，也即是最多可以进行的回合数。

如果 N 为奇数，则最后两个联通块一定是一奇一偶的，所以两个联通块之间

的边一定为偶数条。虽然两个人都采取最优策略，但是每个回合所要操作的人面对的 E 的奇偶性一定相同。所以如果 E 为奇数，则先手必胜，否则后手必胜。

现在考虑 N 为偶数的情况，容易发现任何一个局面，奇数联通块总是偶数个。

我们先考虑 $P = 0$ 的时候，即不允许转移。

当 E 为奇数的时候，先手必胜必须要求最后两个联通块必须都是偶数的。容易发现如果初始状态 1 号点或者 2 号点所在联通块存在一个是偶数，先手总有策略使得必胜。因为奇数联通块为偶数个，先手一开始可以让 1、2 两个联通块都变成是偶数的。如果对方选择和并两个奇数或者两个偶数，则我可以将合并的偶数块合并到 1 号连通块；如果对方合并一奇一偶，则我可以将一个奇数合并到对方合成的奇数块。这样可以保证先手局面两个联通块总是偶数。反之，如果开始局面 1 号点或者 2 号点所在联通块都是奇数，后手同样可以用相同策略使得两个联通块都是奇数，从而使得后手必胜。

当 E 为偶数的时候，先手必胜必须要求最后两个联通块必须都是奇数的。类比上述讨论的情况，容易发现先手必胜只要求 1 号点或者 2 号点所在联通块存在一个是奇数即可必胜。反之如果 1 号点或者 2 号点所在联通块都是偶数则后手必胜。

接下来考虑 $P > 0$ 的情况。

当 E 为奇数的时候，与上述情况类似。先手必胜必须要求最后两个联通块必须都是偶数的。因为 N 为偶数，所以先手只需要使得自己所在联通为偶数，对方联通块一定也是偶数。因为可以传送，所以只需要图上存在一个偶数联通块，先手就可以通过传送使得自己的联通块是偶数，并通过与 $P = 0$ 相同的操作即可保证必胜。如果不存在偶数联通块，先手也可以通过合并两个奇数联通块，然后传送使得自己是偶数连通块。当然，如果刚开始只有两个奇数连通块，那么显然是后手必胜。总结一下先手必胜要求存在偶数联通块或者奇数连通块数大于 2 。

当 E 为偶数的时候，先手必胜必须要求最后两个联通块必须都是奇数的。可以发现无论先手如何操作，后手总有方案使得先手所在联通块是偶数的，唯独一种特殊情况，即一开始只有两个奇数连通块且至多只有一个偶数联通块，先手可以通

过连边和移动使得一次回合就让两个联通块为奇数。

统计联通，我们只需要用并查集即可。

时间复杂度 $O(M\alpha N)$，空间复杂度 $O(M)$。

1.64 A Game of Thrones

【试题来源】

Codechef AUG 2012 GTHRONES

【试题大意】

有 N 种牌，每种牌的权值为 A_i，共有 B_i 张。

两个人进行游戏，先手先选择一张牌 u，并删除之。之后每个人轮流进行操作。每次轮到操作的人，需要找到一张牌 v，使得 u,v 是相似的，如果找不到则认输。

定义两张牌 u,v 为相似的，当且仅当满足以下任意一个条件：

•$u|v$且v/u是一个质数。

•$v|u$且u/v是一个质数。

假设两个人都采取最优策略，请问先手胜利还是后手胜利。若先手胜，输出最小的可以使得先手胜利的牌。

数据范围：$N \leq 500$，$A_i \leq 10^{18}$，$B_i \leq 10^9$。

【算法介绍】

这道题我们考虑用网络流来解决。

非常显然，我们可以将 N 种数按照他分解质约数后的质因子个数的奇偶后建立二分图。然后可以用 $Miller\_Rabin$ 算法来判定两个数是否是相似的，之后进行连边。

容易发现后手必胜，当且仅当这个二分图存在完备匹配。否则是先手必胜。可以这样想，如果先手选择一个当前不在完备匹配中的点，如果不存在匹配该点的

点，则显然先手必胜。否则肯定存在一个已经在匹配中的点，那么我们用增广的思想思考，也就是说存在一条奇数个点的路径，这样显然还是先手必胜。只有当不存在不在完备匹配中的点，即这个二分图完备匹配，则后手必胜。

现在考虑如何输出最小的数满足先手必胜。非常显然，先手只需要选择一个不一定在最大匹配中的点即可。证明与上述情况类似。

这样我们枚举每一种数，如果这种数还存在多余的数没有选，则肯定是先手必胜的；如果存在另外一条增广路还替换该点所对应的路径则同样先手必胜。所以只需要在残量网络上 BFS 判断两个点联通性即可。

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

1.65 Dynamic GCD

【试题来源】

Codechef JULY 2012 DGCD

【试题大意】

给定一棵 N 个节点的树，每个点有一个权值 A_i，进行 Q 次操作，有两种类型。

1.将 u 到 v 路径上的点权值加 c。

2.询问 u 到 v 路径上的点权值的 gcd。

数据范围：$N, Q \leq 50000$。

【算法介绍】

这道题我们考虑用树链剖分解决。

先考虑一般序列上的问题，即区间加，询问区间 gcd，这个问题存在一个非常经典的算法。我们将序列差分，即令 $B_i = A_i - A_{i-1}$，存在结论：

$$gcd(a, b) = gcd(a, a - b)$$

所以区间 $[l, r]$ 的 gcd 就是 $gcd(A_l, B_{l+1}, B_{l+2}, ..., B_r)$。每次区间修改只需要修改两个端点的值就可以了。对于区间询问只需要维护差值后的区间 gcd 和单点的值就可以了。这可以用线段树非常简单的维护。

现在考虑在树上做这个问题。我们可以用树链剖分，之后对于每一条重链维护好差分值，至于 Top 是不需要修改的。对于每次询问，我们只需要将重链上答案统计好，然后在轻重链转换的时候询问出两个点的值，简单处理一下即可。

解决了这些问题，接下来只需要大力码，注意细节即可。

时间复杂度$O(N\log^3 N)$，空间复杂度$O(N)$。

1.66 Cool Numbers

【试题来源】

Codechef June 2012 COOLNUM

【试题大意】

设一个 M 位的数 N 每一个位上的数字为 $x_1, x_2, x_3, ..., x_M$。若 N 存在由某一到三个不同数位上数字相加的和 s，满足：

$$N|(\sum_{i=1}^{M} x_i - s)^s$$

则称 N 是一个 $Cool\ Number$。

T 组数据，每次给出一个数 N，求大于 N 最小的 $Cool\ Number$（$Rc(N)$），以及小于等于 N 最大的 $Cool\ Number$（$Lc(N)$）。

数据范围：$T \leq 100000$，$N \leq 10^{1000}$，$\sum M \leq 4000000$。

【算法介绍】

这道题我们用搜索解决。

首先我们发现，若数 N 是 $CoolNumber$，则必然满足：

$$10^M \leq (9 * M - 27)^{27}$$

经过计算得到，$M \leq 80$。显然，$CoolNumber$ 的个数肯定非常少。于是我们可以采用搜索+高精度来想办法找出所有的 $Cool\ Number$。

设 $k = \sum_{i=1}^{M} x_i$，我们先枚举 $k - s$，显然 $k - s \leq 720$。然后搜索得到 $(k-s)^{27}$ 的所有约数 N。这样我们就可以得到 k 以及 s。判断 N 中是否存在一到三个不同数位满足和等于 s 以及是否整除。若合法则说明找到了一个 $Cool\ Number$。

实践证明 $Cool\ Number$ 的个数小于等于 40000 个，直接搜索就可以找出所有的 $Cool\ Number$ 。

然而上述方法还漏到了一种 $Cool\ Number$ 没有找出。就是数 N 值只有小于等于 3 个数位不为 0 ，这样约束就变成 $N|0$ 。显然，这种数同样是 $Cool\ Number$ 。

对于这类数，我们可以非常轻松的找到 $Lc(N)$ 和 $Rc(N)$。而对于第一类数，我们直接二分就可以得到答案了。两者中选择合适的就是答案。

设 $Cool\ Number$的个数为 C，则时间复杂度$O($玄学的预处理 $+\sum_{i=1}^{T} M + T\log C)$，空间复杂度$O(80 * C + M)$。

1.67 Expected Maximum Matching

【试题来源】

Codechef June 2012 MATCH

【试题大意】

按以下方式随机生成一个左边 N 个点后面 M 个点的二分图：

左边第 i 个点和右边第 j 个点之间有边的概率为 $P[i][j]$。求这样生成的二分图的最大匹配的期望值。

数据范围： $N \leq 5$， $M \leq 100$。

【算法介绍】

这是一道状压 Dp 题。

可以这样设计状压：容易发现我们可以用 $2^5 = 32$ 种二进制状态来表示左边一排点是否被匹配。如果我们用 2^{32} 种二进制状态就可以表示左边一排点有哪些匹配方案是合法的。

容易发现，如果存在两种二分图的连边方式，使得他们的状态相同，在 Dp 中，这两种状态显然可以同时操作。

具体实现时可以发现，有效的状态 S 其实是很少的，我的程序只跑出 406 种可行状态。所以直接进行状压 Dp 即可。

时间复杂度 $O(2^N MS)$，空间复杂度 $O(MS)$。

1.68 Little Elephant and Boxes

【试题来源】

Codechef May 2012 LEBOXES

【试题大意】

有 N 个盒子，打开每个盒子会有 $P_i\%$ 的概率获得 V_i 元钱， $1-P_i\%$ 的概率获得一个钻石。

还有 M 个物品，每个物品需要 A_i 元钱和 B_i 个钻石购买。

对于所有盒子打开后的情况，都会选择一种最优策略去买尽量多的物品。求期望最多能买几个物品。

数据范围：$N, M \leq 30$。

【算法介绍】

这道题我们用 $Meet\ In\ Middle$ 解决。

我们可以先求出 $F[i][j]$ 表示用了不超过 i 个钻石买至少 j 个物品最少花多少钱，这可以用简单的背包实现。

之后我们折半搜索出 $\frac{N}{2}$ 个盒子打开后的所有情况，记录下有 x 个钻石， y 元钱的概率 p。之后我们枚举左右两边分别有几个钻石 i, j ，和我最后要购买至少 k 个物品。假设左右两边的钱加起来大于等于 $F[i+j][k]$ 则可以对答案贡献 $p_i * p_r$。于是我们排序好，之后两个指针扫描用前缀和计算即可。

计算一下效率为:

$$O(M\sum_{i,j}\max\{\binom{\frac{N}{2}}{i}\binom{\frac{N}{2}}{j}\})$$

通过简单的化简，可以发现时间复杂度为$O(2^{\frac{N}{2}}NM)$，空间复杂度是$O(2^{\frac{N}{2}})$。

1.69 Selling Tickets

【试题来源】

Codechef May 2012 TICKETS

【试题大意】

有 N 盘菜， M 个顾客，第 i 个顾客要吃第 A_i 或 B_i 盘菜。你需要找出最大的 x ，使得任意 x 个顾客来，都能吃到自己想吃的菜，一盘菜只能被一个人吃。

数据范围：$N \leq 200$，$M \leq 500$。

【算法介绍】

这道题需要非常巧妙的转换，然后用简单的 BFS 解决。

如果我们把每盘菜看成图上的一个点，每一个顾客 i 看做是一条连接 A_i 与 B_i 之间的边。则对于一个边集 E ，如果这些边包含的点集 V 满足 $|E| \leq |V|$ ，则这肯定合法，可以通过简单的构造实现。

也就是说，我们要求最小的 $|E|$ 使得，满足 $|E| = |V| + 1$ 。

类比 $|E| = |V|$ 的情况。我们发现答案的最优解情况只有两种：

1. 点 S 到点 T 存在三条不同的路径。可以认为是两个环并排在一起。这可以用简单的 BFS 实现，找出前三小的路径。

2. 存在两个环，且中间有一条路径连接着两个环。这可以枚举一个点做根，然后跑出一棵生成树。每一条非树边对应一个环以及到根一段路径的长度，我们只需要找出最短的两个环所代表的权值即可。

时间复杂度$O(N^2M)$，空间复杂度$O(M)$。

1.70 Substrings on a Tree

【试题来源】

Codechef April 2012 TSUBSTR

【试题大意】

给出一棵 N 个节点的树，每个节点上有一个字母。定义树的子串为树上某一对点 u, v，从 u 点到 v 点路径上点的字母连接而成的字符串。其中 u 点是 v 点树上的祖先。

题目要求回答该树有多少不同的子串，并将进行 Q 次询问。每次给出一个排列 P，表示字母 $'a' - 'z'$ 的大小关系。要求求出将该树所有不同的子串按照这个大小关系的字典序排序后的第 K 大子串。

保证最后的输出 $Output \leq 800KB$。

数据范围：$N \leq 250000, Q \leq 50000, K \leq 2^{63} - 1$。

【算法介绍】

这是一道经典的 SAM 题。我们先考虑这个问题的一个简化版本。

假设题目给出的是一条链而不是一棵树。这样问题就变成了询问一个字符串有多少不同的子串和给定大小关系，询问第 K 大的子串。

对于这两类询问，都是后缀自动机 (SAM) 算法中非常经典的问题。大致是这样的流程:

对字符串建立 SAM，根据 SAM 中的 $Parent$ 数组建立 $Parent$ 树。对于第一类询问，可以通过求解 $Right$ 数组，并在 $Parent$ 树上进行简单的树形 Dp 来统计。对于第二类操作，可以利用 SAM 中的 $Trans$ 数组对 SAM 进行 DFS 来解决。

解决的方法非常多，此次不再赘述，详情可以参见 2014 年的集训队论文中关于 SAM 的介绍。

现在考虑原问题，即一棵树的情况。

对于树的情况，非常容易联想到广义的后缀自动机。这其实就是对字典树建立后缀自动机。与序列问题不同的是，树上问题我们需要对于树上每一个节点记录一下这个节点在 SAM 中所对应的点。之后的插入并不能像序列问题那样直接插入在最末端，而应该从它树上父亲在 SAM 中所对应的点处插入。新建节点的信息可以直接覆盖之前节点创建的信息。

通过这种方法就可以得到字典树的后缀自动机。至于题目要求的两类询问，直接效仿之前的做法即可。

时间复杂度 $O((N + Output) * 26)$，空间复杂度 $O(26 * N)$。

1.71 Find a special connected block

【试题来源】

Codechef April 2012 CONNECT

【试题大意】

给出一个 $N*M$ 的矩阵。第 i 行第 j 列的格子颜色为 $A_{i,j}$，权值为 $V_{i,j}$。颜色范围为 $[-1, N*M]$。

现在需要在这个矩阵里找出一个连通块。要求颜色为 -1 的格子不能选，且这个联通块至少要包含 K 个不同的正数颜色，要求连通块内格子的权值和最小。

数据范围：$N, M \leq 15, 1 \leq K \leq 7$。

【算法介绍】

这道题我们用随机+斯坦纳树解决。我们先考虑这个问题的简化版本。

假设矩阵中只有 K 种不同的正数颜色，这就是非常经典的斯坦纳树问题。

我们记 $F[i][j][k]$ 为当前在矩阵的 (i,j) 点， k 是一个二进制状态，表示联通块中颜色的信息。如果 k 二进制上某一位是 1 则表示当前联通块中已经包含这种颜色，否则说明没有选该颜色。

存在两种转移:

1. $F[i][j][k] + V[x][y] \to F[x][y][k]$。其中 (x,y) 为与 (i,j) 相邻的格子。
2. $F[i][j][x] + F[i][j][y\ xor\ x] - V[i][j] \to F[i][j][y]$。其中 $x\ and\ y = x$。

初始状态为：如果一个非障碍的格子 (i,j) 有正数颜色 $A_{i,j}$,则 $F[i][j][2^{A[i][j]-1}] = V[i][j]$，否则 $F[i][j][0] = V[i][j]$。

单次斯坦纳树的效率是 $O(3^K NM)$。最后答案就是对于所有非障碍的格子 (i,j)， $F[i][j][2^K-1]$ 的最小值。

回到原问题，我们来考虑颜色总数大于 K 种的情况。

我们可以将所有的颜色随机给它一个 $[1,K]$ 之间的数，表示把它的颜色重标号。之后对于矩阵中所有的格子，把它的颜色变成新的标号。

因为现在整个矩阵的颜色不超过 K 种了，所有我们可以效仿之前的斯坦纳树做法，求出这种重新标号下的最优解。

容易发现，这种重标号只可能使答案变劣，且肯定都是合法的。实际随机中，因为我们要保证 $[1,K]$ 这些标号至少有矩阵中的一种颜色对应。所以我们需要先从所有颜色中挑出 K 种颜色标号为 $1..K$，之后对于没有选出的颜色随机一个 $[1,K]$ 之间的标号。

现在我们考虑这种随机方案进行单次的正确性。

我们假设最后最优答案的联通块中含有的 K 种颜色为 $A_1, A_2, .., A_K$。那么只要 $A_1, A_2, .., A_K$ 这 K 种颜色随机到的标号各不相同，也就是说是 $1-K$ 的一个排列，那么在斯坦纳树中就可以得到最优的答案。而对于另外的颜色，无论它随机到的标号是什么，都对得到最优解没有干扰。

也就是说，单次的正确概率为：

$$\frac{K!K^{T-K}}{K^T} = \frac{K!}{K^K}$$

其中 T 为矩阵中不同的颜色数。

假设我们进行 $Time$ 次随机，这 $Time$ 次随机得到最优答案的概率为：

$$1-(1-\frac{K!}{K^K})^{Time}$$

由于本题中 $K \leq 7$。经过计算，只需进行 700 次的随机就可以保证正确性达到 99% 了。我提供的标准程序也正是进行 700 次斯坦纳树。

时间复杂度 $O(Time * 3^K NM)$，空间复杂度$O(2^K NM)$。

1.72 Evil Book

【试题来源】

Codechef March 2012 EVILBOOK

【试题大意】

有 N 个厨师，每个人有 C_i 点努力值和 M_i 点魔法。

你可以花费 C_i 点努力值击败第 i 个厨师，然后获得 M_i 点魔法。给定 X ，你还可以花费 X 点魔法，使得第 i 个厨师的魔法和努力值都乘上 $\frac{1}{3}$ 。

当你魔法超过 666 ，你就获胜了。你需要求出最少花费多少努力值可以获胜。

数据范围：$N \leq 10$，$C_i, M_i \leq 10^9$。

【算法介绍】

这道题我们用搜索解决。

容易发现如果当前魔法值已经超过 666 ，那么我就不需要再攻击别的厨师，直接就赢了。否则我们可以对于这个厨师进行 i 次操作，然后攻击它。如果这个厨师剩余魔法超过 666 就直接结束了；如果这个厨师剩余魔法小于 X*i ，则就没有必要攻击了。

所以需要满足等式：$X * i < \frac{M}{3^i}$。可以计算一个厨师有意义的 x 取值是一个长度为 4 的区间。于是直接搜索的效率就是 $O(4^N)$ 。

容易发现可以对所有的厨师进行的操作次数升序做，我们可以这样设计 DFS 状态， $DFS(i, m, c)$ 表示当前对于厨师都采用 i 次操作，魔法值为 m ，努力值为 c 。然后选择一些厨师均进行 i 操作，然后 DFS 下去即可。

期间可以用一些剪枝优化：比如最优性剪枝，如果当前值没有答案优就结束；

可行性剪枝，如果当前状态加上剩余厨师魔法值除以 3^i ，都不能超过 666 ，则结束。

时间复杂度 $O(4^N)$，空间复杂度 $O(N)$。

1.73 Ciel and Earthquake

【试题来源】

Codechef March 2012 CIELQUAK

【试题大意】

给定一张 $N*M$ 的网格图，每相邻之间的格子之间原来有一条边，但是会有 p 的概率断掉。问最后点 $(1,1)$ 与点 (N,M) 联通的概率是多少。

数据范围：$N \leq 8$，$M \leq 10^{18}$。

【算法介绍】

这道题我们用状压 Dp 和一些数学理论来解决这道题。

考虑轮廓线 Dp，因为 N 很小，我们可以用最小表示法来表示这个轮廓线上所有点的联通性，其中若为 1，则表示与 $(1,1)$ 联通。

我们只需枚举一下当前点向上和向左两条边是否断开，然后时时维护这个最小表示法的状态即可。实现发现，状态数 $S \approx 3500$。转移数量 $D \approx 100000$

这样我们得到一个时间效率 $O(NMD)$ 的算法，但是不能通过本题。

通过打表的手段可以发现，当 N,p 不变的时候，当 $M \geq 50$ 以后，答案随着 M 的增长，以一定比例变大。所以我们可以直接算出 $M = 50$ 和 $M = 51$ 的时候的解，然后直接计算得到 M 的答案。

时间复杂度 $O(50ND)$，空间复杂度 $O(D)$。

【备注】

【正确性证明】

为什么当 M 过大时，答案会以一定比例变化呢？这需要用到马尔科夫链的知识，详情参见Markov chain。

1.74 Find a Subsequence

【试题来源】

Codechef Feb 2012 FINDSEQ

【试题大意】

给出一个长度为 N 的序列 A，和一个长度为 5 的序列 B。要求在 A 中找出一个长度为 5 子序列 C，使得 C 中每个数的大小关系与 B 相同。

数据范围：$N \leq 1000$。

【算法介绍】

这道题我们用贪心+各种特判解决。

我们可以枚举第二个数和第四个数，然后用这两个数确定出第一个数和第五个数的可行范围。

贪心的想，如果第一个数比第三个数大，那我肯定选取第一个数可行范围内最大的数，这样可以为第三个的选择腾出更多的范围。第五个数同理。但是还要特判第一个数和第五个数之间的大小关系。比如两个数都比第三个数大则我们需要先选取较大的数，然后去更新另一个数的范围。另外情况同理可得。接着我们只要查看检查是否有可行的第三个数存在即可。

当然，我们还需要预处理出每一个前缀后缀小于或大于某个数最接近的数，以及区间某些数的和。这个可以 $O(N^2)$ 预处理出。

时间复杂度 $O(N^2)$，空间复杂度 $O(N^2)$。

1.75 Flight Distance

【试题来源】

Codechef Feb 2012 FLYDIST

【试题大意】

给出 N 个点 M 条边的图，要求对每条边的权值增加或减少若干。要求每条边的权值都要是正数。使得对于每条边 (u, v, w)，所有从 u 到 v 的路径的权值和大于等于 w。

要求每条边修改权值的绝对值的和最小。

数据范围：$N \leq 10$，$w \leq 20$。

【算法介绍】

这道题我们用线性规划解决。

定义 d_k^+ 为第 k 条边增加的权值，d_k^- 为第 k 条边减少的权值。$g_{i,j}$ 为 i 到 j 的最短路，则可以列出以下约束。

对于每一对 (i, j)，若 (i, x) 之间有边 k，则需要满足：

$$g_{i,j} \leq w_k + d_k^+ - d_k^- + g_{x,j}$$

对于第 k 条边 (i, j) 需要满足：

$$g_{i,j} \geq w_k + d_k^+ - d_k^- \quad \text{且} \quad w_k + d_k^+ - d_k^- > 0$$

我们要求最小化 $\sum d_k^+ + d_k^-$。

一般的线性规划问题都可以用单纯形解决。但是根据单纯形的定义，对于这些约束存在一个初始解，即所有非基变量均为 0。然而此题中当所有非基变量

g, d^+, d^- 都为 0 时，显然不满足上述所有约束。所以我们需要对上述非基变量进行一些转换，使得其满足约束。

可以发现，如果图上所有边的权值均相同，那么肯定满足题目要求的约束。所以最优解每条边的权值肯定在 $[1, 20]$ 之间。这启示我们可以构造一个初始解为所有边权均为 20 。设两点间经过最少边的路径边数为 $H_{i,j}$ 。然后转化变量如下：

1、$g'_{i,j} = 20 * H_{i,j} - g_{i,j}$。

2、$d_k^{+\prime} = 19 - d_k^+$。

3、$d_k^{-\prime} = w_k - 1 - d_k^-$。

容易发现当这些变量为 0 时，刚好对应所有边的边权为 20 时的情况。于是我们可以用这些变量带入最开始的若干约束和目标函数中。

容易发现当 $d_k^{+\prime}, d_k^{-\prime} \geq 0$ 时，边权肯定大于等于 1 ，所以原先对边权大于 0 的约束就不用加了。

但是还需要加入两个新的约束，即 $d_k^+, d_k^- > 0$ 。因为转换变量以后，$d_k^{+\prime}, d_k^{-\prime}$ 过大可能致使 d_k^+, d_k^- 为负数，从而导致目标函数中变量为负数。

至于输出要求为分数，有两种方法。一是在程序中用分数的结构体来代替实数。二是把实数的答案求出来，然后枚举分母，四舍五入算出分子，找出最接近实数答案的分数作为分数答案即可。

做了上述工作，只需要细心化简式子，写个单纯形算法的板子即可通过此题。

时间复杂度 $O(Simplex(N^2, N^3))$，空间复杂度 $O(N^5)$。

【备注】

【关于单纯形算法】

单纯形算法是线性规划中解决一种通用模型的算法。主要思路就是每次从目标函数中找到一个可以使得答案变优的变量。然后去约束中找到对该变量取值范围卡的最紧的约束。通过基变量和非基变量的转变，更新一下约束和目标函数。不断进

行这个过程，使得找到最优解。

但是单纯形算法只能求解实数域下的答案，且时间效率未知。

详情请参见单纯形算法。

1.76 Card Shuffle

【试题来源】

Codechef Jan 2012 CARDSHUF

【试题大意】

开始牌堆有 N 张牌，从顶至底分别标记为 $1, 2, 3, ..., N$ 。需要进行 M 次操作，每次操作给出 A, B, C 。会以如下顺序进行操作：

- 从牌堆顶端拿走 A 张牌。
- 再从牌堆顶端拿走 B 张牌。
- 将第一步拿走的 A 张牌放回到剩下的牌堆上面。
- 从牌堆顶拿走 C 张牌。
- 将第二步你拿起的 B 张牌一张一张放到牌堆顶。
- 最后，将剩下的 C 张牌放回到牌堆顶。

需要回答最后牌堆从顶至底的牌的编号。

数据范围： $N, M \leq 10^5$ 。

【算法介绍】

这是一道非常经典的 *Splay* 裸题，只需要支持区间提取、插入、删除、翻转即可。

时间复杂度 $O(M \log N)$，空间复杂度 $O(N)$ 。

1.77 Misinterpretation 2

【试题来源】

Codechef Jan 2012 MISINT2

【试题大意】

定义一个字符串 s 是好的，当且仅当将 s 中偶数位上的字符按从小到大的顺序放到最前面，然后将奇数位上的字符按顺序放到后面。如果原串和新串相同，则认为这个串是好的。

给出 L, R，要求求出所有长度范围在 $[L, R]$ 之间的好的字符串。

字典集为 26 个小写字母。一共 T 组数据。

数据范围：$L, R \leq 10^{10}$，$R - L \leq 50000$，$T \leq 5$。

【算法介绍】

这是一道较麻烦的数学题。

我们可以把整个变换看成是一个排列的置换。对于长度为 x 的字符串，设它的置换群数量为 $G(x)$，则符合要求的字符串个数为 $26^{G(x)}$。容易发现当 N 为奇数的时候，$G(x) = G(x-1) + 1$，所以我们只需要考虑 x 为偶数的情况。

设 $x = 2N$，则原串为 $s_1 s_2 s_3 s_4 .... s_{2N}$，新串就是 $s_2 s_4 s_6 ... s_{2N} s_1 s_3 ... s_{2N-1}$。设新串中第 i 个位置的编号为 $f(i)$，观察发现：

$$f(i) = 2i \ mod \ (x+1)$$

考虑如何计算长度为 x 的字符串有多少置换群。根据 gcd 的性质，有：

$$gcd(i, x+1) = gcd(f(i), x+1)$$

这说明 $gcd(i, x+1)$ 相同的 i 在该置换中存在一定的封闭性。且满足 $gcd(i, x+1) = p$ 的 i 的数目为 $\phi(\frac{x+1}{p})$。如果我们能求出每一个 i 所在置换群的大小 len，那么我们就可以计算出一共有多少置换群了。

定义 $ord(d)$ 为最小的正整数 k 满足：

$$2^k \ mod \ d = 1$$

我们发现 i 所在置换群的大小为最小的数 k 满足：

$$i * 2^k \equiv i \ (mod \ x+1)$$

等式两边同除 $p = gcd(i, x+1)$，得：

$$\frac{i}{p} * 2^k \equiv \frac{i}{p} \ (mod \ \frac{x+1}{p})$$

此时 $\frac{i}{p}$ 与 $\frac{x+1}{p}$ 互质，且 $x+1$ 为奇数。所以上式成立的要求就是：

$$2^k \equiv 1 \ (mod \ \frac{x+1}{p})$$

这就是我们之前定义的 $ord(d)$。由此，可得：

$$\begin{aligned} G(x) &= \sum_{p|(x+1), p<x+1} \frac{\phi(\frac{x+1}{p})}{ord(\frac{x+1}{p})} \\ &= \sum_{p|(x+1), x>1} \frac{\phi(p)}{ord(p)} \end{aligned}$$

现在考虑如何求解 $ord(x)$。因为 $x+1$ 为奇数，所以所有要求 $ord(d)$ 的 d 都是奇数。根据欧拉定理有：

$$2^{\phi(x)} \equiv 1 \ (mod \ x)$$

容易发现 $ord(x)$ 肯定是 $\phi(x)$ 的约数。所以令 $ord(x) = \phi(x)$，再枚举 $\phi(x)$ 的每一个质因子 a。若 $\frac{ord(x)}{a}$ 还是满足要求，则将 $ord(x)$ 除去 a。

然而这样做太慢了，我们需要新的解法。设 $a|x$ ，且 $gcd(a,\frac{x}{a})=1$ ，则有：

$$ord(x)=lcm(ord(a),ord(\frac{x}{a}))$$

现在只需要考虑如何计算 $ord(x)$ 满足 $x=p^q$，其中 p 为质数，$q>1$。

对于 $x=p$ ，可以直接暴力计算出 $ord(x)$ 。对于 $x=p^i$ ，可以发现： $ord(p^i)$ 最大为 $ord(p^{i-1})*p$ 。

于是得到了一种相对优秀的算法：枚举 $x\in[l,r]$ ，计算 $G(x)$ 。将 x 分解质因子，然后 DFS 出 x 的所有的约数。在 DFS 过程中，可以用“积性”的性质，计算得到 $\phi(d),ord(d)$ 。然后直接统计答案即可。

现在问题在于如何快速分解。且如果快速计算 $ord(p)$, p 为质数。

我们可以预处理出所有小于等于 $\sqrt{r}$ 的质数，以及我们想要的质数幂次的 ord 信息。因为每一个只有可以用一个大于 $\sqrt{r}$ 的质因子，所以对于这个质因子，我们可以 RHO 分解来暴力求 $ord(p)$ 。

对于分解 $[l,r]$ 之间的数，因为这是连续的数，所以同样可以枚举 $\sqrt{r}$ 以内的质数，然后去试除，最后剩下的数肯定是一个质数。

还需要一些卡常数的小技巧才能通过本题。

设 $M=\sqrt{r},N=10^{10}$，因为 $r-l$ 和 M 范围差不多，所以下面也简写成 M ，约数个数为 D。

时间复杂度 $O(M\log^2 N+T*(M\log^2 N+MN^{\frac{1}{4}}+MD*\log N)$，空间复杂度 $O(M\log N)$。

【备注】

【关于RHO算法】

RHO 是一种质因数分解的算法，时间效率 $O(N^{\frac{1}{4}})$ 。

详情参见：Pollard-Rho-Methode。

1.78 Short II

【试题来源】

Codechef Dec 2011 SHORT2

【试题大意】

给定一个质数 p，问有多少对 $a, b(a > p, b > p)$，满足 ab 被 $(a-p)(b-p)$ 整除。

数据范围：$p \leq 10^{12}$

【算法介绍】

这是一道较麻烦的数学题。

首先原问题等价于求满足 $ab|p(a+b+p)$ 的数对个数。我们分三种情况讨论问题：

1、$p \mid a, p \mid b$，此时满足条件的数对只有 5 对。

2、$p \nmid a, p \nmid b$，我们直接枚举即可。

3、a, b中只有一个被 p 整除，可以发现这种数对数是 2 中数对数的两倍。

时间复杂度 $O(\sqrt{p})$，空间复杂度 $O(1)$。

1.79 Hypertrees

【试题来源】

Codechef Dec 2011 HYPER

【试题大意】

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。

数据范围：$N \leq 17$。

【算法介绍】

因为读入的 N 范围非常小，可以用某种手段打个表解决。

时间复杂度 $O(1)$，空间复杂度 $O(1)$。

1.80 Luckdays

【试题来源】

Codechef Nov 2011 LUCKYDAY

【试题大意】

给出数 a, b, x, y, z, p, c, q。定义数列 S为：

1、 $S[1] = a$。

2、 $S[2] = b$。

3、 $S[i] = (S[i-1] * X + S[i-2] * Y + Z) \ mod \ p$， $i > 2$。

接下来有 q 组 l_i, r_i，询问有多少 k 满足：$l \leq k \leq r$ 且 $S[k] = c$。

数据范围：$p \leq 10007$，$q \leq 20000$，$l_i, r_i \leq 10^{18}$，p 是质数。

【算法介绍】

这道题可以用矩阵 $BSGS$ 解决。

我们先考虑用矩阵乘法的形式定义递推式。

对于每一个 i 有行向量 $(S_i, S_{i-1}, 1)$，有转移矩阵 M 为：

$$\begin{pmatrix} x & 1 & 0 \\ y & 0 & 0 \\ z & 0 & 1 \end{pmatrix}$$

对于 $y = 0$ 的情况，可以证明数列 s 的变换是一个 ρ 形的变换，且循环节小于等于 p，这个我们可以暴力处理。

对于 $y \neq 0$ 的情况，可以证明 s 的变换是一个圆形的变换。所以假设我们对于任意的 $d \in [0, p-1]$，都求出 $(c, d, 1)$ 第一次出现的下标 i；再求出这个圆的循环节长度。对于每次询问，我们就可以二分计算答案了。

考虑如何计算从 $(b, a, 1)$ 到 $(c, d, 1)$，以及 $(b, a, 1)$ 变回 $(b, a, 1)$ 的最小正步数。我们可以考虑用矩阵 $BSGS$ 解决问题。

时间复杂度 $O(P\sqrt{P} + Q \log P)$，空间复杂度 $O(P\sqrt{P})$。

1.81 Colored Domino Tilings and Cuts

【试题来源】

Codechef Nov 2011 DOMNOCUT

【试题大意】

一个 N 行 M 列的矩形棋盘。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。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。

给定 N 和 M 。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。

数据范围： T 组数据，$T \leq 3000$，$N, M \leq 500$，$\sum N * M \leq 2000000$。

【算法介绍】

这是一道构造题。

让 N, M 都比较大的时候，显然无法用两种颜色染色，而我们可以构造出6 ×8 和5 ×6 的两种棋盘，这两个棋盘的割数是0，染色数是3，而且这两种棋盘的特点是可以在不改变割数的情况下增加两行或者两列（只需要添加一排横着的或者竖着的1 ×2 单位即可）。

于是我们可以通过这两种基础的网格构造出N, M比较大的时候割数等于0 的解。至于染色数，可以发现构造出来的棋盘依然是满足三染色的，可以在拓展棋盘的过程中完成构造。

当 NM 比较小的时候我们可以通过打表来输出答案。

时间复杂度 $O(TNM)$，空间复杂度 $O(NM)$。

1.82 The Baking Business

【试题来源】

Codechef Oct 2011 BAKE

【试题大意】

进行 N 次操作，每次要么给出一种产品的信息；要么询问满足一定条件的产品的信息。

数据范围：$N \leq 10^5$。

【算法介绍】

这是一道简单的模拟题。

因为每件产品的所涉及到的信息类别和范围非常小。所以只要开个多维数组维护一下前缀和就可以了。

时间复杂度 $O(N)$，空间复杂度 $O(1)$。

1.83 Sine Partition Function

【试题来源】

Codechef Oct 2011 PARSIN

【试题大意】

给定 M, N, X ，求解下面这个式子：

$$Ans = \sum_{k_1+k_2+...+k_M=N} \sin(k_1X)\sin(k_2X)...\sin(k_MX)$$

数据范围：$M \leq 30$，$N \leq 10^9$，$0 \leq X \leq 2\pi$。

【算法介绍】

这道题用简单的三角函数公式和矩阵乘法即可解决。

根据三角函数的和角公式有：

$$\begin{aligned} \sin(\alpha+\beta) &= \sin(\alpha)\cos(\beta)+\cos(\alpha)\sin(\beta) \\ \cos(\alpha+\beta) &= \cos(\alpha)\cos(\beta)-\sin(\alpha)\sin(\beta) \end{aligned}$$

所以我们设：

$$\begin{aligned} f(N,M) &= \sum_{k_1+k_2+...+k_M=N} \sin(k_1X)\sin(k_2X)...\sin(k_MX) \\ g(N,M) &= \sum_{k_1+k_2+...+k_M=N} \cos(k_1X)\sin(k_2X)...\sin(k_MX) \end{aligned}$$

易得递推式：

$$f(N,M) = f(N-1,M-1) * \sin(X) + f(N-1,M) * \cos(X) + g(N-1,M) * \sin(x)$$
$$g(N,M) = f(N-1,M-1) * \cos(X) - f(N-1,M) * \sin(X) + g(N-1,M) * \cos(x)$$

矩阵乘法之。

时间复杂度 $O(M^3 \log N)$，空间复杂度 $O(M^2)$。

1.84 Short

【试题来源】

Codechef Sept 2011 SHORT

【试题大意】

给定两个数 N, M，需要求出有多少数对 (a, b)，满足 $N < a < M, N < b < M$，并且 $(a-N)(b-N)|ab-N$。

数据范围：$N \leq 10^5$，$M \leq 10^{18}$。

【算法介绍】

这是一道非常玄学的数学题。

当 $N = 0$ 的时候，答案显然为 $(M-N-1)^2$，需要用高精度解决。否则，$(a-N)(b-N)$ 肯定小于 $ab-N$。

设 $c = a-N, d = b-N$，若原式成立，则肯定存在 $k(k>1)$ 满足：

$$\begin{aligned} kcd &= (c+N)(d+N) - N \\ kcd &= cd + Nc + Nd + N^2 - N \\ ((k-1)c - N)d &= N(N+c-1) \\ d &= \frac{N(N+c-1)}{(k-1)c-N} \end{aligned}$$

我们强制 $c \leq d$，则有：

$$\begin{aligned} c &\leq \frac{N(N+c-1)}{(k-1)c-N} \\ (k-1)c^2 - Nc &\leq N^2 + Nc - N \\ (k-1)c^2 - 2Nc - N^2 + N &\leq 0 \end{aligned}$$

根据二次函数，有：

$$c \leq \frac{2N + \sqrt{4N^2 - 4(k-1)(-N^2+N)}}{2(k-1)}$$

现在我们要求 c 的上界，可以发现当 $k = 2$ 的时候， c 上界最大，为：

$$c \leq N + \sqrt{N^2 - N}$$

这个范围非常小，可以直接枚举可行的 c ，然后枚举 $N(N + c - 1)$的约数 d ，判断一下合法性即可。

当 c 过大的时候，我们可以枚举 $(k-1)c - N$ ，因为 c 比较大，所以加了一些次数之后， $c \leq d$ 这个约束就不能满足了。然而具体要枚举多少次，我也不清楚。

可以预处理每个数的质约数，然后 DFS 出约数来加速。

时间复杂度 $O(N * \text{玄学})$，空间复杂度 $O(N \log N)$。

1.85 Counting Hexagons

【试题来源】

Codechef Sept 2011 CNTHEX

【试题大意】

有 N 种木棍，每种木棍的长度为 $[1, N]$，且每种木棍只有 K 个。

需要在其中选择出 6 根木棍，使得最长的那根木棍长度大于等于 L，其他木棍小于等于 X。且这些木棍可以组成一个凸六边形。求方案数。

数据范围：$N \leq 10^9$，$N - L \leq 100$。

【算法介绍】

这道题可以用动态规划解决。

首先有个结论，一堆木棍可以组成凸六边形，当且仅当最长木棍的长度小于剩下木棍长度之和。

因为 $N - L$ 非常小，所以我们可以枚举最长的木棍长度 L，然后相当于求解有多少 5 根木棍满足：

1、 木棍和大于 L。

2、 所有木棍长度小于等于 X。

3、 每根木棍最多用 K 根。

我们考虑用数位 Dp，即从低位到高位用二进制来解决问题。根据这些约束，我们这样设计状态，令 $F[a][b][c][d][e]$ 表示当前在第 a 位，之前填的木棍的和与 L 大小关系为 b。进位为 c，再用一个二进制数表示将这 5 根木棍从大到小排序以后，相邻两位是否相等，为了解决这些木棍和 X 的大小关系，我们将 X 也记到 d

中，再用 e 表示 X 是第几大的数即可。

每次转移，我们需要枚举当前位，这些木棍是 0 或 1。然后可以快速得到我们想要的信息。为了加速，我们需要预处理出所有可以转移的状态。

为了避免 Dp 过程中，存在木棍长度为 0 的情况。我们可以令初始木棍长度均 $+1$，然后令 $L-5$ 即可。

最后对于每一个状态，我们只需要判断约束是否都满足，再累加答案即可。

时间复杂度 $O((N-L)\log N * 2^5 * 2^5 * 2 * 5 * 5)$，空间复杂度 $O(2^5 * 2 * 5 * 5)$。

1.86 Shortest Circuit Evaluation

【试题来源】

Codechef Aug 2011 SHORTCIR

【试题大意】

给定一个长度为 N，仅包含括号、 and、 or、 not及 M 个变量的布尔表达式。并给出每一个变量为 $true$ 的概率。

假使计算这个表达式值的时候，采用短路原则，即若干的 or 运算时，只要算到一个 $true$，即直接返回该表达式为 $true$，不对剩下变量进行运算。

要求自定义顺序去计算每一个变量的值。求在最优情况下，期望能确定这个表达式值的最小步数。

数据范围：$N \leq 30000$，$M \leq 1000$。

【算法介绍】

这道题需要用一些简单的语法分析，以及一些期望、贪心的知识解决。

我们先单纯考虑如何计算表达式的值。我们可以建立一个栈，将表达式中的信息逐个加入栈中讨论。对于左括号， not 我们直接压入栈中。因为 and 优先级比 or 高，所以我们遇到 and 就直接压入栈中，遇到 or 则将前面的数和 and 全部消光，计算出他们的值然后当成一个数。遇到右括号，则现将末尾的 and 消光。现在这对括号里只可能有 or，然后在计算出值并消光。每次加入数的时候需要将前面的 not 消光。这样我们就可以计算出一个表达式的值了。

再考虑如何计算本题要求的期望。我们对于每一个变量记一个 e 表示期望确定该变量值的最优步数，p 表示这个变量是真的概率。

对于一堆 *and* 运算。假设我们计算 $1,2,3,4..,t$ 这些变量，利用期望的线性性，容易发现，这一组变量的值为：

$$E = \sum_{i=1}^{t} e_i \prod_{j=1}^{i-1} p_j$$

$$P = \prod_{i=1}^{t} p_i$$

如何计算最优的 E 呢，我们可以利用贪心的思想，假设交换其中两个元素，然后列不等式求解。最后结论是将所有变量按照 $\frac{e}{1-p}$ 升序排序，然后依次计算即可。

对于 *or* 运算同样类似，这里不再赘述。

时间复杂度 $O(N + M \log M)$，空间复杂度 $O(N)$。

1.87 Something About Divisors

【试题来源】

Codechef Aug 2011 DIVISORS

【试题大意】

给出 B 和 X，求有多少 N 满足，至少存在一个数 $D(N < D \leq B)|NX$。

数据范围：$B \leq 10^{12}$，$X \leq 60$。

【算法介绍】

令正整数$i = NX \mid D$，那么显然有$i < X$，考虑枚举每一个可能的 i，为了避免重复的统计，我们可以计算满足 $i \mid NX$ 且不存在 j 满足 $i < j < X$ 且 $j \mid NX$ 的 N。

因为 $i|N * X$，所有 $\frac{i}{gcd(i,X)}|N$，令$A_i = \frac{i}{gcd(i,X)}$。那么就有$N = A_i * p$。因为有 $N \leq \frac{B*i}{X}$，所以就有 $p \leq \frac{B*i}{X*A_i}$。把这个取值的上界记为 P。

因为 $j \mid A_i * X * p$，所以有 $\frac{j}{gcd(AiX,j)} \mid M$。令 $B_j = \frac{j}{gcd(A_i*X,j)}$。

那么满足 $i \mid NX$ 的同时满足 $j \mid NX$ 的条件是 $D_j \mid M$。所以我们可以枚举 i，然后对所有 $i < j < X$，进行容斥，即$ans_i = \sum (-1)^t \frac{P}{lcmB_j}$。

直接计算显然会超时，但是可以发现当分子大于 P 时贡献就是 0 了，再加上可能的 lcm 的值也不多，所以就可以用 DP 的方式来进行。注意优化一下常数就能通过了。

设 K 为可能的 lcm 个数，实践证明 $K \leq 10000$。

时间复杂度 $O(X^2K)$，空间复杂度 $O(XK)$。

1.88 Billboards

【试题来源】

Codechef July 2011 BB

【试题大意】

有 N 个广告牌，要求任意连续的 M 个广告牌都至少有 K 个广告。求有多少种方案满足摆放的广告数最少。

数据范围：$N \leq 10^9$，$K \leq M \leq 50$。

【算法介绍】

这是一道结论题，用杨氏图表的知识解决。

我们先考虑 $M|N$ 的情况。即整个广告位被分成了 $\frac{N}{M}$ 块。显然每个块中必须有 K 个广告。如果我们将这些广告统一放在每一块的最右边若干位。显然任意的 M 个连续的块中都有 K 个广告。

我们定义从右边数过来第 j 个广告在第 i 个块中的位置为 $A_{i,j}$。观察发现，$A_{i,j} \geq A_{i+1,j}$，$A_{i,j} \geq A_{i,j+1}$。

这个是非常经典的半标准型杨氏图表。假设该杨氏图表的长宽为 N, M，且每个格子有 r 个数可以选，则他的计数有如下定理：

$$dim = \prod_{i=1}^{N} \prod_{i=1}^{M} \frac{r+j-i}{hook(i,j)}$$

其中 $hook(i,j)$ 为格子 (i,j) 向右和向下的格子数+1。我们将图标画出来可以发现，有很多元素是可以抵消的，所以只需要计算 $K * M$ 个数就可以了。

现在考虑 M 不整除 N 情况。分两种情况讨论。若 $N\%M \leq M - K$，则要求每一块的前 $N\%M$ 都是 0。否则，则要求每一块最后的 $M - N\%M$ 都是 1。

时间复杂度 $O(KM \log N)$，空间复杂度 $O(1)$。

1.89 Trial of Doom

【试题来源】

Codechef July 2011 YALOP

【试题大意】

给定一个 $n*m$ 的网格，每一个格子为蓝色或者红色，一共有 k 个红色格子，你可以在这个网格里走，每当你离开一个格子的时候，这个格子和它的四周四个格子会改变颜色，现在让你判断是否存在一条左下角到右上角的可以经过相同格子的路径，使得从左下角开始，沿着这条路径走，走到右上角再离开这个网格后，网格里所有格子都变成了蓝色。

数据范围：$N, M \leq 10^9$，$min(N, M) \leq 40$，$k \leq 10000$。

【算法介绍】

这是一道结论题。

我们假定 $n \leq m$，当 $n > 1$ 的时候，路径的限制是不重要的，即对于任意的格子的集合 S，都存在一条路径使得这条路径经过了 S 中的格子奇数次，其它格子偶数次。这样就转化成了一个经典问题，只要能够列出方程，就可以使用高斯消元来解决。可以发现此时每一个红色的格子都是独立的，只有格子 (i, j) 是红色，等价于只有格子 $(i-1, j-1), (i-1, j), (i-2, j), (i-1, j+1)$ 是红色的。因此可以使用这种方法把第 j 列的格子给移动到第 $j-1$ 列上。

因为 n 很小，可以发现一个第 i 列的格子对第 j 列和第 $j-1$ 列的影响是存在一个周期的。即存在一个数 T 使得对第 $j, j-1$ 列的影响和第 $j-T, j-T-1$ 列的影响是一样的。那么可以先预处理出这个循环节，那么就可以在 $O(nK)$ 的时间内把所有红色格子都移动到第一列。同样利用这个循环节，我们可以把设在最后一列

的变量给移动到第一列，这样就可以得到 n 个方程，判断这个方程组是否有解即可。上述过程可以用位运算优化掉一个 n。当 $n = 1$ 的时候，路径的长度的奇偶性和 n 一定相同，于是我们可以先暴力枚举第一个格子经过次数的奇偶性，然后解出每一个格子经过了多少次，然后分别判断一下和的奇偶性是否满足要求即可。

设 Len 为循环长度。时间复杂度 $O(K + n * Len)$，空间复杂度 $O(Len)$。

1.90 Attack of the Clones

【试题来源】

Codechef June 2011 CLONES

【试题大意】

题目定义了四种集合 Z, P, D, A，集合之间两两有一定的相交关系。给出一个表达式，包含括号，并集，交集，差集，补集以及四种集合。要求出这个表达式代表的集合的大小。

数据范围：设表达式长度为 L，数据组数为 T， $L, T \leq 100$。

【算法介绍】

这是一道简单的模拟题。

首先我们可以先预处理出四种集合的大小。我们可以用一个 2^4 的数表示一个集合是否被这四个集合包含。最后的结果肯定是若干个这个的结果组成，于是可以用 2^{16} 的数表示最后的答案。

知道了这些，只需要写一个非常简单的语法分析器就可以了。

时间复杂度 $O(T * L)$，空间复杂度 $O(L)$。

1.91 Minesweeper Reversed

【试题来源】

Codechef June 2011 MINESREV

【试题大意】

给出一个 $N * M$ 的网格图，包含了空格和雷。这是一个扫雷的逆过程，每次可以点击一个点，所有在正常做扫雷时可以顺便打开的格子就会一起被关上。求最少步数，使得每个格子都被关上。

数据范围： T 为数据组数， $T, N, M \leq 50$ 。

【算法介绍】

这道题我们用匹配算法解决。

非常显然，雷肯定要通过一次点击关上。对于剩下的格子，分成两类。一类与雷相邻，一类没有。对于第二类，可以构成若干连通块，我们只需要点击联通块内的一个点，整个联通块就可以被关上。但是对于那些与雷相邻的格子，可能会被两个连通块包含。点这个格子的时候，两个联通块会被同时关闭。这可以形象的认为是一条边，而连通块就是一个点。我们只会要求出这张图的最大匹配，然后用总点数减去它就是最小步数了。

因为这张图是普通图，所以需要用带花树算法解决。

时间复杂度 $O(TN^3M^3)$，空间复杂度 $O(NM)$。

Chapter 2

10 道challenge型试题

2.1 Similar Graphs

【试题来源】

Codechef April 2012 SIMGRAPH

【试题大意】

给出两张 N 个点的图，要求将两张图的点都重新标号。使得两张图的公共边数量尽可能多。即最大化点对 (u, v)，点u和点v在两张图中都存在边。

数据范围：$N \leq 75$。

【算法介绍】

这道题是一道 *challenge* 题，我采用的方法是随机。

题目需要我们找到一组关于两张图的点标号，使得这两张图尽量同构。容易发现，我们可以强制第一张图的标号为 1,2,3,..., N，之后我们只要对第二张图进行标号即可。

非常朴素的想法，我们可以对第二张图随机一些长度为 N 的排列表示标号，取答案最大的为答案，之后我们可以进行一些爬山、模拟退火等算法调整一下解。事实证明这样已经可以得到不错的解了。

我的方案是每次挑出一对 u, v，之后将第二张图中的点 u 和点 v 的标号互换，计算一下交换后的代价，根据代价的优劣性，以不同的概率选择是否交换点 u 和点 v 的标号。重复这个过程 $Time$ 次，最后取最优解作为答案。数据的标准输出也是根据这个方案生成的。

毕竟是 $Challenge$ 类型的题目，方案非常多，在此不一一列举，但所有方案应该是基于随机的策略。

时间复杂度 $O(Time * N)$，空间复杂度$O(N^2)$。

2.2 Closest Points

【试题来源】

Codechef June 2012 CLOSEST

【试题大意】

三维空间中有 n 个点，现在给出了 q 组询问，每一组询问给出了一个点，你需要从给定的 n 个点中找出距离这个点最近的点的编号。

数据范围： $n, q \leq 50000$ 。

【算法介绍】

这是一个经典问题，我们可以直接用传统的 $KD-tree$ 来达到100% 的正确率。

注意坐标范围是 10^9 ，所以两点之间距离的平方可以达到 $1.2 * 10^{19}$ ，这已经超过 $long\ long$ 的范围了。

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

2.3 Stepping Average

【试题来源】

Codechef Nov 2011 STEPAVG

【试题大意】

考虑以下的方法求出N个数的“迭代平均数”。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”。值得注意的是，不同的合并顺序最后会导出不同的“迭代平均数”。

给定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。

数据范围： $N \leq 1000$ 。

【算法介绍】

这道题我们采用贪心的方法。

我们将整个序列排序，如果整个序列整体大于 K ，那么我们把最小的数 x 留下来，令剩下的数同样也递归这个过程，使得其最接近 $2K - x$ 。如果整体小于 K ，同理，我们把最大的数 x 留下来，令剩下的数最接近 $2K - x$ 。否则我们直接将当前最大的数和最小的数合并，重复这个过程。

时间复杂度 $O(N^2)$，空间复杂度$O(N)$。

2.4 Kali and Devtas

【试题来源】

Codechef Dec 2014 KALKI

【试题大意】

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 Ci 全部增加 1（包括自身）。

数据范围： $N \leq 400$ 。

【算法介绍】

直接对于给定的图做最小生成树即可。

时间复杂度 $O(N^2logN)$，空间复杂度$O(N^2)$。

2.5 Maximum Sub-rectangle in Matrix

【试题来源】

Codechef OCT 2012 MAXRECT

【试题大意】

给定一个 $N * M$ 的矩阵 A，你需要找出一个子矩阵使得权值和最大。

注意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。

数据范围： $N, M \leq 300$ 。

【算法介绍】

这道题我们用随机算法解决。

先随机一个初始解，然后进行若干次以下过程。每次找到一个行或列，使得改变它的选否状态可以使答案变得更优。

再进行一定的随机调整即可。

设随机次数为 $Time$，时间复杂度 $O(Time * (N + M))$，空间复杂度$O(NM)$。

2.6 Sereja and Permutation

【试题来源】

Codechef April 2014 SEAPERM

【试题大意】

给定一个长度为 N 的序列 A' 和整数 S，需要将这个序列重新排列，设重排后的序列为 A。

令函数 $f(A,i)=S-\sum_{k=i}^{j}A_k$，其中 j 为最大的数满足 $S-\sum_{k=i}^{j}A_k\geq 0$。

现在给定 K，需要最小化 $\frac{\sum_{i=1}^{K}f(A,i)}{K}$。

数据范围：$N\leq 2000$。

【算法介绍】

假设我们已经知道了第 $k+1$ 个数到第 n 个数是什么，我们通过排列剩下的数，最小化 $f(A,k)$，并在此基础上最小化 $f(A,K-1)$，依次类推。这样的贪心如果朴素的时限的话是 $O(n^2)$ 的，但是我们可以使用平衡树来优化这个过程到每次 $O(n\log n)$。

随机若干次使得解最优。

设随机次数为 T，时间复杂度 $O(Tn\log N)$，空间复杂度$O(N)$。

2.7 The Great Plain

【试题来源】

Codechef Oct 2011 LAND

【试题大意】

给定一个 $n*m$ 的网格，网格中有一些格子已经填上了1 到50 的整数，你需要在其他的格子中填上这个范围内的整数。填好之后，网格中任意一对相邻的格子 (i,j) 会产生 $2^{|A_i-A_j|}$ 的代价，其中A_i 表示格子i 上的整数，两个格子时相邻的当且仅当它们恰好存在一条公共边。

你需要尽可能地最小化网格的代价和

数据范围： $N,M \leq 100$ 。

【算法介绍】

考虑一个贪心的方法，我们扫描每一个网格，在其它格子都不变的情况下调整它的值，使得它和它周围的格子产生的代价和最低。显然这是一个迭代的过程，即扫描完网格之后我们再扫描一次，格子的值依然有可能发生变化。我们可以扫描若干次直到它稳定为止。

上述做法的效果依赖于初始解的选择，我们可以用贪心的方法来选择初始解，即把每一个空白格子划分给离它曼哈顿距离最近的初始给定的格子，并把它的权值赋为那个格子的权值。

这个做法存在一个弊端，就是在迭代的过程中可能会陷入一个比较劣的解的死锁中，即每一个格子都没有办法进行修改，但是这个解很差。因此随机的扰动是比较必要的，可以在每一次迭代之后都随机 S 个格子把它们的权值减去1。

还有一个优化是，与空白格子产生的代价和初始给定的格子产生的代价的估价应该是不同的，因为空白格子可以继续调整而初始给定的格子不行。因此在估价的时候可以给它们分别乘上一个常数。

通过这种方法就能得到一个比较优秀的解了。

设随机次数为 P，时间复杂度 $O(PNM)$，空间复杂度$O(NM)$。

2.8 Simultaneous Nim

【试题来源】

Codechef Sept 2012 SIMNIM

【试题大意】

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，使得每一组的异或和都为 0。

数据范围：$N \leq 1000$。

【算法介绍】

进行若干次随机，每次随机一个排列，然后找出最小的集合使得 xor 和为 0，并删去这个集合。

我们可以用高斯消元解异或方程来找出最小的集合。

设随机次数为 P，M 为最小的数满足 $2^M \geq A_i$。

时间复杂度 $O(PNM)$，空间复杂度$O(N)$。

2.9 Deleting numbers

【试题来源】

Codechef Aug 2013 DELNMS

【试题大意】

给定一个长度为 N 的序列 A，你需要对它进行若干次操作。每次操作需要选定两个数 v,t，另 k 为最大的数满足 $v+kt \leq N$ 且 $A_v = A_{v+t} = A_{v+2t} = ... = A_{v+kt}$。然后将这些数删去。需要找一种方法使得最小步数删完所有数。

数据范围： $N, A_i \leq 10^5$ 。

【算法介绍】

这道题我们用贪心的思想解决。可以将出现次数最多的数拎出来，然后想办法把其他数都删光，最后一次就可以消光所有数了。

接下来我们做若干次以下的过程。每次暴力找一组 v,t 然后删掉。注意要卡一卡时间。

时间复杂度 O(刚好一秒)，空间复杂度$O(N)$。

2.10 To challenge or not

【试题来源】

Codechef June 2013 CHAORNOT

【试题大意】

给定一个长度为 N 的数组 B，你需要从中选出尽可能多的数使得选出的数中不存在任意的三个数可以组成等差数列。

数据范围：$N, B_i \leq 100000$。

【算法介绍】

考虑最简单的贪心，把数组排序之后，从小到大判断，如果当前的数能加入到选取的集合中，那么就选取这个数，否则就不取。假设这样得到的答案个数为 K，因为这个做法得到的 K 不会很大，所以可以比较快地跑出来。

时间复杂度 $O(K^2 + M)$，空间复杂度$O(M)$。