IOI2016中国国家集训队第一次作业
Codechef试题泛做

宁波市镇海蛟川书院 施舟行

Contents

1 非Challenge型试题

1.1 JUN11 CLONES

试题名称

Attack of the Clones.

题目大意

定义一种形如$f : A \to B$的函数，满足A是所有长度为n，且仅由$0, 1$组成的序列的集合，$B = \{0, 1\}$。称这样的函数所组成的集合为$clone$。 现有四种特殊的$clone$如下：

1. Z：满足$f(0, 0, \cdots, 0) = 0$。
2. P：满足$f(1, 1, \cdots, 1) = 1$。
3. D：满足$!f(x_1, x_2, \cdots, x_n) = f(!x_1, !x_2, \cdots, !x_n)$。
4. A：满足若$f(a_1, \cdots, a_i, \cdots, a_n) = f(a_1, \cdots, 1-a_i, \cdots, a_n)$，则$f(b_1, \cdots, b_i, \cdots, b_n) = f(b_1, \cdots, 1 - b_i, \cdots, b_n)$。

现在给出$Q(Q \leq 100)$组询问，每组询问给出一个含Z, P, D, A的布尔表达式，来表示要求满足这个布尔表达式的$clone$的大小，表达式长度L不超过100，只含差、交、并、补运算。

算法讨论

可以使用一个四位二进制数，来表示所求集合是否包含Z, P, D, A。通过经典的表达式计算处理，可以求出询问表达式所要求的集合状态x。需要预处理出G_x表示状态为x的集合大小。为了讨论方便，可以引入F_x表示是否必须包含Z, P, D, A中的一些元素的状态为x时，集合大小。分x的取值对F_x的值进行讨论，如下：

- 0000: 2^{2^n}.
- 1000: 2^{2^n-1}.
- 0100: 2^{2^n-1}.

- 1100: 2^{2^n-2}.
- 0010: $2^{2^{n-1}}$.
- 1010: $2^{2^{n-1}-1}$.
- 0110: $2^{2^{n-1}-1}$.
- 1110: $2^{2^{n-1}-1}$.
- 0001: 2^{n+1}. 假如现在已经考虑了前i位，则对于第$i+1$位，自变量只有这一位不同的函数的值要么都相等，要么都相反，共有2^n种可能，而第一位也有2种可能，故为2^{n+1}。
- 1001: 2^n. 与0001的区别仅在于第一位取值被固定。
- 0101: 2^n.
- 1101: 2^{n-1}. 与1001和0101的区别在于最后一位的选择可以直接推出。
- 0011: 2^n. 最后一位可直接推出。
- 1011: 2^{n-1}. 第一位取值被固定且最后一位可直接推出。
- 0111: 2^{n-1}.
- 1111: 2^{n-1}.

通过F_x可求出G_x，再处理询问即可。

时空复杂度

时间复杂度为$O(n+L)$。
空间复杂度为$O(L)$。

1.2 JUN11 MINESREV

试题名称

Minesweeper Reversed.

题目大意

考虑经典扫雷游戏的反游戏：需要把一个已经完全打开的$N*M$扫雷游戏盘中的方格重新盖上。当点击一个格子时，可以直接盖上这个格子，同时，在原游戏中，可能存在点击某一个格子时，同这个格子一同被打开的格子也一起被盖上。求最少需要的点击次数。

数据范围：$T \leq 50$[1],$N, M \leq 50$。

算法讨论

放置有雷的格子必须被点击一次，可以直接统计到答案中。

其余的格子可以分为两类：周边没有雷的格子形成的一些连通块；周边有雷的格子。对于无雷格子的连通块，点击其中任意一个格子都可以盖上块内的全部格子，可以一并考虑。当点击这两种格子时，若周边有另一种格子，则同样可以把它们盖上。对于一个周边有雷的格子，若其周边没有无雷格子的连通块，则这个格子必须被点击一次，否则可以通过点击周边的连通块来代替；但若其周边有两个不同的无雷格子的连通块，点击这一个格子可以同时盖上旁边的两个连通块，相比较分别点击两个块，能够节省一次点击，并且，这样的节省不具有传递性。因此，可以建立一点数为原游戏盘上连通块数的无向图，两点间有边相连，当且仅当它们所代表的，在原游戏盘中的块，有一个公共相邻的周边有雷的格子。对这个无向图使用带花树算法求最大匹配，即可求出能够节省的点击数，从原答案中减去即可。

时空复杂度

时间复杂度为$O(TN^2M^2)$。

空间复杂度为$O(NM)$。

1.3 JUL11 BB

试题名称

Billboards.

[1]在本文中，若无特殊说明，T均指测试数据组数。

题目大意

有排成一列的N个格子，要求对其中的一些格子进行染色，满足任意连续M个格子中，都至少有K个格子被染色。求在被染色格子数量最少情况下，方案数对10^9+7取模后的结果。

数据范围：$T \leq 300, N \leq 10^9, K \leq M \leq 500$。

算法讨论

首先来考虑$M|N$的情况。此种情形下，N个格子可被划分为N/M个部分，对每个部分的最后K个格子进行染色，即可满足要求，这也是染色格子数量最少的方案。我们用一个矩阵a来表示染色的方案，$a_{i,j}(1 \leq i \leq K, 1 \leq j \leq N/M)$表示第$j$个部分中，第$i$个被染色的格子在$M$个格子中的相对位置。矩阵$a$存在性质：

$$a_{i,j} \geq a_{i,j+1}$$

$$a_{i,j} < a_{i+1,j}$$

查阅相关资料可知，矩阵a的性质，符合*semi-standard Young tableau*的定义。问题也就是要求大小为$K*(N/M)$，可选数字个数为M的semi-standard Young tableau个数。参阅相关公式，可知答案为：

$$\prod_{i=1}^{K}\prod_{j=1}^{N/M}\frac{j-i+M}{i+j-1}$$

举个例子来说，当$N=20, M=4, K=2$时，要乘上的内容有：

$$\begin{matrix} 8 & 7 & 6 & 5 & 4 \\ 7 & 6 & 5 & 4 & 3 \end{matrix}$$

而要除去的内容有：

$$\begin{matrix} 6 & 5 & 4 & 3 & 2 \\ 5 & 4 & 3 & 2 & 1 \end{matrix}$$

可以发现，这两个矩阵中，有很多连续列是对应相同的，可以约去，而剩下的列数不超过$2M$，暴力计算即可。

时空复杂度

时间复杂度为$O(TM^2)$。
空间复杂度为$O(1)$。

1.4 JUL11 YALOP

试题名称

Trial of Doom.

题目大意

在一个$N \times M$个棋盘上，要从$(1,1)$走到(N,M)再离开，格子是八连通的，并且有黑白两种颜色，每当离开一个格子时，这个格子连同与其四连通的格子都会被改变颜色。现在已知其中的一些格子在初始时是白色的，问是否存在行走的方案，使得结束时，所有格子都是黑色的。

数据范围：$T \leq 50$，$N, M \leq 10^9$，$\min(N,M) \leq 40$，$0 \leq K \leq 100000$（K为初始时为白色的格子数）。

算法讨论

考虑每个格子被经过次数奇偶性的状态，可以证明，当$N, M \geq 2$时，所有状态都是可达的，这可以分解为两种操作：

- 在不改变状态的情况下，从一个格子移动到一个相邻格子。
- 改变当前所在格子被访问次数的奇偶性。

这两个操作可以在一个格子周围的2×2的子棋盘中完成。问题也就是要判断是否存在一种满足要求的状态。

若格子$(x,y)(x > 1)$在初始时是白色的，并且存在一种满足要求的状态，那么就可以通过改变$(x-1,y)$的状态，来使得当格子(x,y)是白色，$(x-1,y-1),(x-1,y),(x-1,y+1),(x-2,y)$（如果存在）是黑色时满足要求。于是，可将格子$(x,y)(x > 1)$推移到$x = 1$的情况，推移的过程中，关于行会产生循环节。最后就是经典的开关灯问题了，设最后一行每个格子是否被经过奇数次，就可以列出具有M个未知数，M个方程的方程组，使用高斯消元即可判断是否有解。

再来考虑$\min(N,M)=1$的情况，不妨设$M=1$，则经过的格子个数与所在格子的x坐奇偶性是相同的，因此，和$N,M\geq 2$的情况不同，不是所有的状态都是可达的。通过枚举第一个格子的经过情况，可以推得其它格子的情况，再判断奇偶性即可。

时空复杂度

设L为循环节长度，经实际测试，L约2×10^4。
时间复杂度为$O(ML+M^2)$。
空间复杂度为$O(ML)$。

1.5 AUG11 SHORTCIR

试题名称

Shortest Circuit Evaluation.

题目大意

给出一个含N个布尔变量的表达式，变量i有p_i概率为真，要求安排各变量的计算顺序，使得总计算次数的期望值尽量小。如，若要计算$a\ and\ b$，若已知a为假，则无需计算b。

数据范围：$T\leq 50$，$N\leq 1000$，表达式长度L不超过30000。

算法讨论

对于表达式$a_1\ and\ a_2\ and\ \cdots a_n$，考虑如何安排变量计算顺序，可使计算次数尽量少。设变量i为真的概率为p_i，期望计算次数为e_i，当前表达式为真的概率为P，期望计算次数为E。易知：

$$P=\sum_{i=1}^{n}p_i$$

若按原顺序计算各变量，则：

$$E=\sum_{i=1}^{n}e_i(\prod_{j=1}^{i-1}p_j)$$

若交换变量i与$i+1$的计算顺序，可使E变小，则：

$$e_i(\prod_{j=1}^{i-1} p_j) + e_{i+1}(\prod_{j=1}^{i} p_j) > e_{i+1}(\prod_{j=1}^{i-1} p_j) + e_i(p_{i+1}\prod_{j=1}^{i-1} p_j)$$

化简，得：

$$\frac{e_i}{1-p_i} > \frac{e_{i+1}}{1-p_{i+1}}$$

因此，只需将各变量按$\frac{e_i}{1-p_i}$从小到大排序后，按顺序计算即可。对于or运算的情况，与上类似，不再赘述。而对于not运算，只需将概率取反即可。

采用与经典表达式计算问题类似的处理方式，维护操作优先级栈与变量栈处理即可。

时空复杂度

时间复杂度为$O(L + NlogN)$。

空间复杂度为$O(L)$。

1.6 AUG11 DIVISORS

试题名称

Something About Divisors.

题目大意

对于给定的正整数B和X，求满足条件的正整数N的个数，要求对于N，至少存在一个数$D(N < D \leq B)$能整除$N \times X$。

数据范围：$T \leq 40$，$B \leq 10^{12}$，$X \leq 60$。

算法讨论

令$K = \frac{N \times X}{D}$，由于$N < D \leq B$，故$K < X$，$N \leq \frac{B \times K}{X}$，且$K | N \times X$。

由于$K | N \times X$，故$\frac{K}{gcd(K,X)} | N$，令$A_K = \frac{K}{gcd(K,X)}$，则可以设$N = A_K \times m$，故$m \leq M_K = \frac{BK}{XA_K}$。

不考虑重复，K对答案的贡献就是M_K。考虑对于所有的$K'(K < K' < X)$满足$K'|N \times X$，也就是$B_{K'} = \frac{K'}{gcd(K', A_K \times X)}|m$。这里需要去除所有$K'$产生的重复。利用容斥原理可以统计贡献，具体计算的时候可以DP，同时还需大量优化。

时空复杂度

时空复杂度为O(玄学)。

1.7 SEP11 SHORT

试题名称

Short.

题目大意

给出n, k，求数对(a, b)个数，满足$n < a < k$，$n < b < k$，$(a-n)(b-n)|(ab-n)$。

数据范围：$T \leq 5$，$0 \leq n \leq 10^5$，$k \leq 10^{18}$。

算法讨论

首先若$n = 0$，则显然答案为$(k-1)^2$。以下默认$n > 0$。

考虑将$(a-n)(b-n)|(ab-n)$变形为：$ab - n = p(a-n)(b-n)$。若$p = 1$，则需满足$a + b = n + 1$，可知在$n > 0$的条件下无法实现，故$p \geq 2$。

考虑先枚举a，则：

$$b = n + \frac{n(a-1)}{p(a-n) - a}$$

设$d = p(a-n) - a$，$d|n(a-1)$，因此可以枚举$n(a-1)$的因数作为d，从而可以得到b, p的值。此做法复杂度极高，需要优化。

考虑只计算满足$a \leq b$的数对(a, b)数。由于$p \geq 2$，且：

$$p = \frac{a+d}{a-n}$$

故可得：

$$d \geq a - 2n$$

而：

$$b = n + \frac{n(a-1)}{d}$$

因此：

$$b \leq n + \frac{n(a-1)}{a-2n}$$

而我们已经假设$a \leq b$，于是又可得：

$$a \leq b \leq n + \frac{n(a-1)}{a-2n}$$

进一步解这个不等式，可得：

$$a \leq 2n + \sqrt{2n^2 - n}$$

因此在枚举a时，不需要完整地在区间(n, k)中枚举，而只需要在$(n, \min(k-1, 2n+\sqrt{2n^2-n}))$中枚举即可。$2n+\sqrt{2n^2-n}$大约趋向于$3.42n$。

枚举$n(a-1)$的约数，可使用预处理出约400000范围内数的质因子，然后Dfs即可。但该算法仍可改进。由于$p = (a+d)/(a-n)$，故$(a-n)|(a+d)$。而当a增大时，满足条件的d是大量减少的。由于：

$$a \leq b = n + \frac{n(a-1)}{p(a-n)-a}$$

可知：

$$p \leq \frac{a^2-n}{(a-n)^2}$$

当a比较大时，p的范围很小，因此此时由枚举d改为枚举p即可。

1.8 SEP11 CNTHEX

试题名称

Counting Hexagons.

题目大意

给出N, L, X, K，要求在$[1, X]$中可重复地选五个整数a_1, a_2, a_3, a_4, a_5，在$[L, N]$中选择一个整数a_6，满足$a_1 \leq a_2 \leq a_3 \leq a_4 \leq a_5 < a_6$，$\sum_{i=1}^{5} a_i > a_6$，且$a_1, \cdots, a_6$中，不存在一个整数出现次数超过$K$。

数据范围：$2 \leq N \leq 10^9$，$2 \leq L \leq N$，$N - L \leq 100$，$1 \leq X < L$，$1 \leq K \leq 5$。

算法讨论

考虑使用十进制数位DP。令$dp_{i,j,mask}$表示当前考虑到第i位，这一位需要得到的来自低位的进位至少为j，当前6个整数与X, L, N之间的大小关系区分状态为$mask$。容易知道$j < 5$，而若当前前五个整数和已经超过了第六个整数，接下来就无需考虑进位的问题，为了方便，可用$j = 5$标记。枚举这一位上六个整数填的数字，进行转移即可。来看这一算法的复杂度：$10 \times 6 \times 2^7 \times 10^6$，略偏高。

进一步观察发现，只有第五、六个整数的选择受到i的限制，因此，可以预处理$pre_{i,j,mask,j',mask',a5,a6}$，表示当前有$dp_{i,j,mask}$，欲向$dp_{i-1,j',mask'}$转移，第五、六个整数的第$i$位分别已选择了$a5$和$a6$，转移系数的值。于是，在转移的时候，就无需再枚举前四个整数这一位上的值，直接调用预处理的结果即可。该优化能较明显地提升算法效率，复杂度分析见下。

时空复杂度

时间复杂度为$O(6 \times 2^4 \times 10^6 + 10 \times 6^2 \times 2^7 \times 10^2 \times 2^4 \approx 1.7 \times 10^8)$。

空间复杂度为$O(10 \times 6 \times 2^7 + 7^2 \times 2^8 \times 10^2)$。

1.9 OCT11 BAKE

试题名称

The Baking Business.

题目大意

需要支持$M(M \leq 10^5)$条操作，操作有以下两种类型：

- 卖出$units_sold(units_sold \leq 100)$件物品，物品有两种属性$a$和$b(0 \leq a \leq 10, 0 \leq b \leq 3)$。购买物品的人有五种属性$c(0 \leq c \leq 10), d(0 \leq d \leq$

$20), e(0 \leq e \leq 15), gender(0 \leq gender \leq 1), age(1 \leq age \leq 90)$。其中，$a, b, c, d, e$中的部分属性是可以缺失的。

- 询问满足条件的卖出的物品数目：物品两种属性为a, b，购买人前四种属性为$c, d, e, gender$，第五种属性在范围$[agel, ager]$内。其中，一些条件是可以缺失的，表示不作要求。

笔者猜测，本题出题人的本意可能是考查字符串的处理。因此，若有需要了解具体输入格式，请参阅原题描述。

算法讨论

理解题意，并按输入格式要求，将输入转化为具体的条件后，我们使用数组记录每种属性为确定值时，当前已卖出的物品数目。回答询问时，在要求的第五种属性范围内枚举这种属性的具体值，在数组中查询其他属性满足当前条件的物品数目和即可。

时空复杂度

时间复杂度为$O(Mage)$。
空间复杂度为$O(a \times b \times c \times d \times e \times gender \times age)$。

1.10 OCT11 PARSIN

试题名称

Sine Partition Function.

题目大意

给出整数$N(1 \leq N \leq 10^9), M(1 \leq M \leq 30)$，实数$X(0 \leq X \leq 6.28)$，求:

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

算法讨论

对于$f(N,M)$，分第N个元素是否自成一组，可以从$f(N-1,M-1)$和$f(N-1,M)$递推得到。若第N个元素自成一组，则对$f(N,M)$的贡献为$f(N-1,M-1)sinX$；否则，$f(N-1,M)$中的最后一部分$sin(k_MX)$需要改变为$sin((k_M+1)X)$，并加入到$f(N,M)$中。

$$sin((k_M+1)X)=sin(k_MX+X)=sin(k_MX)cosX+cos(k_MX)sinX$$

由此，我们可以设辅助函数$g(N,M)$：

$$g(N,M)=\sum_{k_1+k_2+\cdots+k_M=N} sin(k_1X)sin(k_2X)\cdots sin(k_{M-1}X)cos(k_MX)$$

于是：

$$f(N,M)=f(N-1,M-1)sinX+f(N-1,M)cosX+g(N-1,M)sinX$$

$g(N,M)$的递推式与$f(N,M)$类似，参考公式$cos(\alpha+\beta)=cos\alpha cos\beta-sin\alpha sin\beta$，可得：

$$g(N,M)=g(N-1,M-1)cosX+g(N-1,M)cosX-f(N-1,M)sinX$$

由于N较大，建立矩阵来加速递推即可。

时空复杂度

时间复杂度为$O(M^3logN)$。
空间复杂度为$O(M^2)$。

1.11 NOV11 LUCKYDAY

试题名称

Luckdays.

题目大意

已知数列$S_1 = A, S_2 = B, S_i = (XS_{i-1} + YS_{i-2} + Z) \bmod P (i \geq 3)$。给出$Q$组询问，需要回答整数$k$的个数，满足$L_i \leq k \leq R_i, S_k = C$。

数据范围：$2 \leq P \leq 10007$，$0 \leq A, B, X, Y, Z, C < P$，$Q \leq 20000$，$L_i, R_i \leq 10^{18}$，$P$为质数。

算法讨论

对于数列中的元素S_n，可以使用一个1*3的矩阵来表示其状态：$[S_n, S_{n-1}, 1]$。设S_2的状态为数列的初始状态$init$，通过建立转移矩阵$trans$，$init \times trans^{k-2}$即是S_k的状态。可以发现，用矩阵表示的元素的状态，存在循环节，且长度不超过P^2，这也表明原数列存在循环节。我们首先考虑如何求出循环节长度L。

$$init \times trans^L = init$$

若矩阵$trans$存在逆矩阵，则$|trans| > 0$，也就是$Y \neq 0$。当$Y = 0$时，问题可以简化，特判即可。否则，使用BSGS算法，就可以求出L。 接下来，我们要求出循环节长度范围内，状态为$[C_i, j, 1]$的元素出现的位置。类似地使用BSGS算法可以求得。最后回答询问时，通过之前求得的位置和二分，可以求得答案。

时空复杂度

时间复杂度为$O(P^{1.5} + QlogP)$。

空间复杂度为$O(P)$。

1.12 NOV11 DOMNOCUT

试题名称

Colored Domino Tilings and Cutsontest.

题目大意

给出一个$N \times M$的棋盘，要求用1×2的彩色骨牌去覆盖这个棋盘。求构造一种方案，使得棋盘的水平、竖直割尽量少，同时还要使颜色数尽量少。

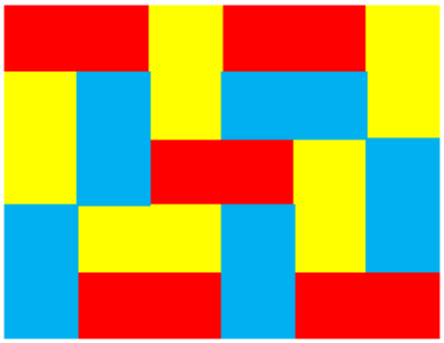
数据范围：$T \leq 3000$，$N, M \leq 500$，所有数据中NM和不超过2×10^6。

算法讨论

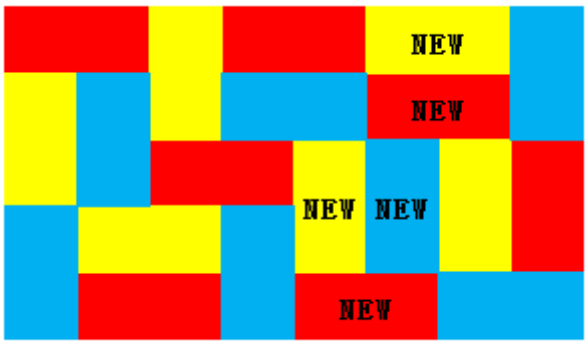
若NM为奇数，则显然无解。否则，均可以构造得解。不妨先假设$N \leq M$。

当N较小时，直接构造比较容易。

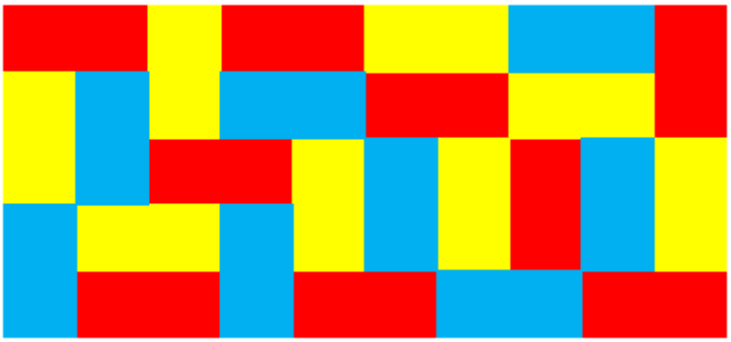
当$N \geq 5$且N, M中有一者为奇数时，不妨认为N是奇数。构造可以得到一种5×6的覆盖方案，其中割为0，颜色数为3：



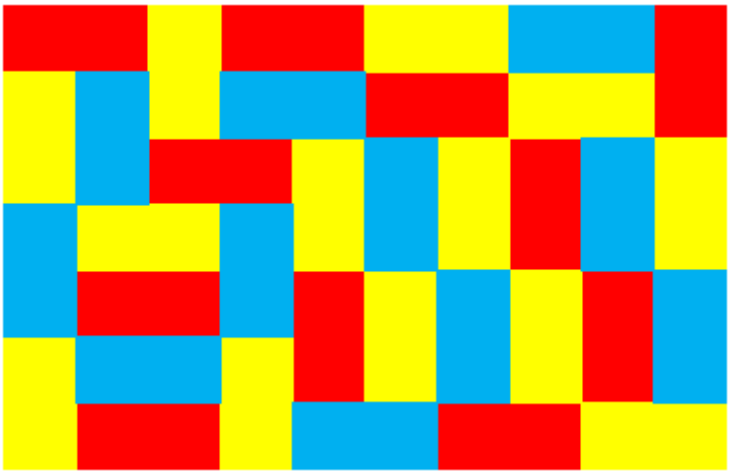
在这种方案的基础上，每次可以增加两行或两列，不改变割数和颜色数，从而扩大规模。若增加两列，则如下所示（其中New为新加入的骨牌）：



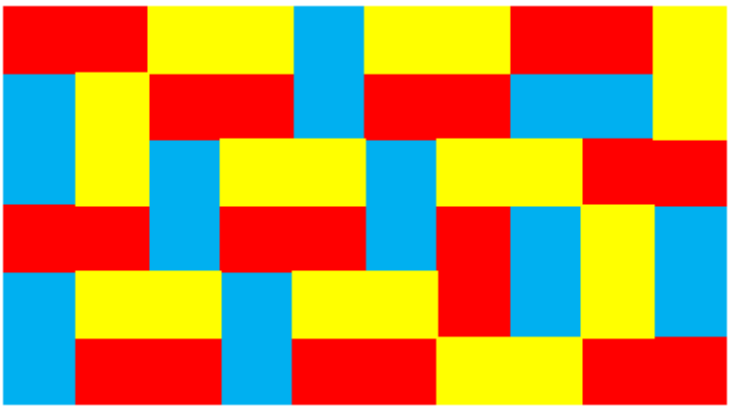
再增加两列：



增加两行：



当$N \geq 6, M \geq 8$且N, M均为偶数时，与上一种情况类似地，构造可以得到一种6×8的覆盖方案，其中割为0，颜色数为3：



在这种方案的基础上，同样可以在保证割数和颜色数不改变的情况下，扩展两行或两列。

时空复杂度

时间复杂度为$O(\sum NM)$。
空间复杂度为$O(NM)$。

1.13 DEC11 SHORT2

试题名称

Short II.

题目大意

给出质数p，求数对$(a,b)(a>p,b>p)$的数量，满足$(a-p)(b-p)|ab$。
数据范围：$T\leq 5$，$1<p<10^{12}$。

算法讨论

原问题可转变为求数对$(a,b)(a>0,b>0)$的数量，满足$ab|(a+p)(b+p)$，这个条件与$ab|p(a+b+p)$等价。考虑几种情况：

- 若$p|a$且$p|b$，则条件可转变为$\frac{ab}{p^2}|\frac{a+b+p}{p}$，也就是$xy|(x+y+1)(x>0,y>0)$，可以发现，这样的数对$(x,y)$只有5对：$(1,1),(1,2),(2,1),(2,3),(3,2)$。

- 若$p\nmid a$且$p\nmid b$，则条件可转变为$ab|(a+b+p)$。若$a=b$，则$a=b=1$。否则，可以假设$a<b$。显然$a+b+p\geq ab$，也就是$(a-1)(b-1)\leq p+1$，因此$a<w=1+\sqrt{p+1}$。设$a+b+p=kab$，则$b=\frac{a+p}{ak-1}$。设$d=ak-1$，则$bd\leq w+p$。若$b\leq d$，则$b\leq\sqrt{w+p}$，因此可以从1到$\sqrt{w+p}$枚举b，又因为$a+p\equiv 0(\mathrm{mod}\ b)$，也就是$a\equiv -p(\mathrm{mod}\ b)$，且$a<b$，所以$a$的可能取值只有一个，验证即可。若$b>d$，此时$d<\sqrt{w+p}$，从1 到$\lfloor\sqrt{w+p}\rfloor-1$枚举$d$，由于$a\equiv -p(\mathrm{mod}\ d)$，且$a=\frac{d+1}{k}$，故$a\leq d+1$，$a$的可能取值最多只有两个，验证即可。

- 若a,b中有一者是p的倍数，通过观察，可知此种情况的答案数是上一情况的两倍。这是可以证明的，因为上一种情况中的每一合法数对(a,b)，都可以转化为$(a,\frac{p(a+b)}{b})$和$(\frac{p(a+b)}{a},b)$。

时空复杂度

时间复杂度为$O(T\sqrt{p})$。
空间复杂度为$O(1)$。

1.14 DEC11 HYPER

试题名称

Hypertrees

题目大意

3-超图是其中的每条边连接三个点的特殊图。3-超树是去掉任意一条边后都不连通的3-超图。

给出$N(3 \leq N \leq 17)$，求有多少含有N个带标号的点的本质不同的3-超树.

算法讨论

用搜索可以求出点数为i的点双连通超树数量。之后可以枚举目标超树每一个点双连通分量的大小，用这些分量可以拼成一个大的超树。本地求出答案后交表即可。

时空复杂度

时间复杂度为$O(1)$。
空间复杂度为$O(N)$。

1.15 JAN12 CARDSHUF

试题名称

Card Shuffle.

题目大意

有一个含$N(N \leq 10^5)$个元素的序列，初始时为$1, 2, \cdots, N$。

要求模拟$M(M \leq 10^5)$条过程，每个过程给出三个参数A, B, C，有六个步骤，如下：

1. 从序列的头部取出A个元素。
2. 从序列的头部取出另B个元素。
3. 将第一次取出的A个元素放回序列的头部。
4. 从序列的头部再取C个元素。
5. 将第二次取出的B个元素倒序放回序列的头部。
6. 将第三次取出的C个元素放回序列的头部。

输出最终得到的序列。

算法讨论

上述所涉及的操作都属于*Splay*的基本操作，因此，只需要使用一棵*Splay*来实现上述过程即可。

时空复杂度

时间复杂度为$O(NlogN)$。

空间复杂度为$O(N)$。

1.16 JAN12 MISINT2

试题名称

Misinterpretation 2.

题目大意

给出$L, R(L, R \leq 10^{10}, R - L \leq 50000)$，求长度在$[L, R]$间的字符串个数，满足提取偶数位字符并移动到串头后，所得到的串与原串相同。字符集大小为26，答案对$10^9 + 7$取模。

算法讨论

对于一个长度为n的字符串，将其按题意调整顺序可看做是一个置换，若置换的循环数为$f(n)$，则答案为$26^{f(n)}$。当n是奇数时，调整顺序不会改变最后一位，因此$f(n)=f(n-1)+1$，故只需考虑n为偶数的情况。令$ord(i)$表示2模i的阶。对于$1\le i\le n$，i所在循环大小为$ord(\frac{n+1}{gcd(i,n+1)})$。于是可得：

$$f(n)=\sum_{d|(n+1),d>1}\frac{\phi(d)}{ord(d)}$$

接下来考虑如何求这个表达式的值。通过预处理100000的质数，可以容易地求得$n+1$的质因子，通过Dfs可以求出$n+1$的因数，同时维护ϕ和ord值。考虑如何求$ord(d)=\prod p_i^{\alpha_i}$。可以发现$ord(d)=LCM\{ord(p_i^{\alpha_i})\}$。于是就只要求$ord(p^\alpha)$。显然$ord(p^\alpha)|(p^{\alpha-1})(p-1)$，枚举$(p^{\alpha-1})(p-1)$的质因数进行试除即可。对于$p\le 100000$的$ord(p^\alpha)$预处理，那么在枚举$n(L\le n\le R)$时，每次最多只需进行一次ord的计算。

时空复杂度

时间复杂度为$O(\sqrt{R}log^3R+(R-L)(\sqrt{R}+log^2R))$。
空间复杂度为$O(\sqrt{R}(R-L))$。

1.17 FEB12 FINDSEQ

试题名称

Find a Subsequence.

题目大意

给一个长度为N的序列A，和一个长度为5的模板序列B，要求从A中找出一个长度为5的子序列，满足其中的元素大小关系与序列B相同。

数据范围：$T\le 60, N\le 1000$。

算法讨论

枚举子序列中的第二、四个元素在原序列中的位置l, r，需要进一步求得其它三个元素的位置。设第三个元素在子序列中是第k小的，则在子序列中，前$k-1$小的和后$5-k$大的元素可以递推求得，只需要预处理出原序列中，所有前缀、后缀不小于某个元素的最小值和不大于某个元素的最大值即可。然后需要求得第三个元素的位置，使用二维前缀和，可以判断$[l+1, r-1]$中是否存在满足要求的位置，若存在，暴力枚举找到这一个元素并返回方案即可。

时空复杂度

时间复杂度为$O(TN^2)$。
空间复杂度为$O(N^2)$。

1.18 FEB12 FLYDIST

试题名称

Flight Distance.

题目大意

给出一有N个点，M条带权边的无向图。要求修改其中一些边的权值，满足对于任意通过一条无向边相连的点(x, y)，不存在一条路径，从x出发，经过一些中间点后，再到y，所经过边的权值和小于这条边的权值。修改一条边的代价为权值改变量的绝对值。求最小代价和，用既约分数表示。

数据范围：$N \leq 10, M \leq 45$，边权为不超过20的正整数。

算法讨论

设$w_{i,j}$为连接点(i, j)的边的原始权值，$d_{i,j}$为这条边权值的改变量。设$Dis_{i,j}$为修改完权值后，点(i, j)之间的最短距离，需要满足：

$$Dis_{i,j} = w_{i,j} + d_{i,j} \quad \Leftrightarrow \quad Dis_{i,j} \leq w_{i,j} + d_{i,j} \leq Dis_{i,j}$$

$$Dis_{i,j} \leq Dis_{i,k} + Dis_{k,j}$$

要求最小化：

$$z = \sum |d_{i,j}|$$

为了去掉目标函数z中的绝对值，可将边(i,j)的改变量$d_{i,j}$，拆为$da_{i,j}$和$db_{i,j}$两个部分。其中，$da_{i,j}$表示权值的增加量，$db_{i,j}$表示权值的减少量，这两个量满足$da_{i,j}, db_{i,j} \geq 0$。于是可得：

$$z = \sum da_{i,j} + db_{i,j}$$

由此，我们可以建立线性规划模型。使用单纯形算法可以求解线性规划，为了方便，在使用单纯形算法时，常常会将变量的初始值调整为0，同时，满足所有的约束条件。但是，上述的模型中，变量值均为0无法满足约束条件。可以发现，当所有边的权值被调整为0 时，是满足约束条件的。因此，对于所有边(i,j)，我们可以强制其先减去$w_{i,j}$的权值，再设立一个变量$dc_{i,j}$，表示对这一操作的反悔量，满足：

$$0 \leq dc_{i,j} \leq w_{i,j}$$

总的来说，解下述线性规划即可：

$$Dis_{i,j} \leq w_{i,j} + da_{i,j} - db_{i,j} + dc_{i,j} \leq Dis_{i,j}$$

$$Dis_{i,j} \leq Dis_{i,k} + Dis_{k,j}$$

$$da_{i,j}, db_{i,j}, dc_{i,j} \geq 0$$

$$dc_{i,j} \leq w_{i,j}$$

目标函数：

$$z = \sum da_{i,j} + db_{i,j} + w_{i,j} - dc_{i,j}$$

这个解法中，需建立$2N^2$个变量。Codechef官方题解提供了一种只需建立$2M$个变量的解法，有兴趣的读者可参阅FLYDIST - Editorial。

时空复杂度

时间复杂度为$O(Simplex(N^3 + 3M, 2M^2))$（其中$Simplex(n,m)$为使用单纯形算法求解有$n$个约束条件，$m$个变量的线性规划的复杂度。尽管单纯形算法在最坏情况下，复杂度并不是多项式的，但它在实际使用时效率却非常可观）。

空间复杂度为$O((N^3+3M)M^2)$。

1.19 MAR12 EVILBOOK

试题名称

Evil Book.

题目大意

有N个物品可以使用，使用第i个物品，需要耗费C_i代价，带来M_i力量。现在可以多次使用魔法，使用一次魔法耗费X力量，并可以指定一件物品，使它的代价和力量都缩小到原来的1/3。求使得力量不低于666的最小代价。

数据范围：$N \leq 10, 10 \leq X < 666, 0 \leq C_i, M_i \leq 10^9$。

算法讨论

使用魔法次数较少的物品，应比使用次数多的物品先被使用。考虑按使用魔法次数进行按层搜索，每次选择一件物品，使用当前层魔法次数的魔法，并使用这件物品。在搜索的过程中，需要加入简单的可行性剪枝与最优性剪枝。

时空复杂度

时间复杂度为$O(Number_of_States)$。

空间复杂度为$O(N)$。

1.20 MAR12 CIELQUAK

试题名称

Ciel and Earthquake.

题目大意

给出一$R \times C$的网格图，其中每条边由P的概率不连通。求点$(1,1)$与点(R,C)连通的概率。

数据范围：$1 \leq R \leq 10^{18}$，$1 \leq C \leq 8$。

算法讨论

若R较小，则可以使用轮廓线DP，状态中记轮廓线上的点以及$(1,1)$之间的连通性即可。当R较大时（约超过40时），发现答案随R增长的倍数趋向一个值，于是就只需计算约$R \leq 40$时的答案。

时空复杂度

时间复杂度约为$O(TMB_M)$。
空间复杂度为$O(MB_M)$。

1.21 APR12 TSUBSTR

试题名称

Substrings on a Tree.

题目大意

给出一棵含$N(N \leq 250000)$个点的点上带有字符的树（类似于trie）。对于任意满足x是y祖先的两点x, y，提取从x到y路径上的字符所组成的字符串，并将这些字符串去重。

给出$M(M \leq 50000)$条询问，每组询问给出26个字母顺序表，求在以该顺序表为参照的情况下，字典序第k小的字符串。

数据保证输出文件不超过$180KB$。

算法讨论

BFS这棵树，同时建立后缀自动机（注意，DFS可能超栈空间）。对于后缀自动机上的每一个状态，求出由该状态所能转移到的状态总数，这可以采用拓扑遍历，或用链表按最大可达长度存储每个状态，并倒序枚举状态，递推求得。处理询问时，在后缀自动机上从根往下走即可。

时空复杂度

时间复杂度为$O(N+K)$（其中$O(K)$表示输出文件大小）。
空间复杂度为$O(26N)$。

1.22 APR12 CONNECT

试题名称

Find a special connected block

题目大意

给出一个$N*M$的棋盘，棋盘上的每个格子带有数字$A_{i,j}(-1 \leq A_{i,j} \leq N*M)$和代价$cost_{i,j}$。

要求选出棋盘中的一个代价和最小的四连通块，满足其中不包含数字-1，且共出现了至少K种不同的正数。

数据范围：$N, M \leq 15, K \leq 7$。

算法讨论

当棋盘上的数字较小时，可以建立斯坦纳树模型。而当数字较大时，可以考虑将这些数字随机映射到K种数字上，再求斯坦纳树，容易求得这样一次随机映射后，能够求出最优解的概率是$K!/(K^K)$（即考虑最优解中每种数字的映射情况）。由于$K \leq 7$，一次随机后能够求出最优解的概率约为0.006。于是，只需随机几百次，就可将出错率降至较低水平，足以通过本题。

时空复杂度

时间复杂度为$O(\alpha NM3^K)$（其中α为随机算法运行次数）。

空间复杂度为$O(NM2^K)$。

1.23 MAY12 LEBOXES

试题名称

Little Elephant and Boxes.

题目大意

有N个盒子，在第i个盒子中，有$p_i\%$的概率获得V_i美元，$1-p_i\%$的概率获得一颗钻石。

有M件物品，购买第i件物品需要C_i美元，D_i颗钻石。

求最多能购买的物品数量期望值。

数据范围：$2 \leq N \leq 30, 1 \leq M \leq 30, 1 \leq V_i, C_j \leq 10^7$。

算法讨论

使用类似背包的DP，可以求出$dp_{i,j}$表示拥有i颗钻石时，希望购买j件物品最少需要花费的美元。

再使用*Meet in the middle*思想，将盒子分为$\lfloor N/2 \rfloor, N - \lfloor N/2 \rfloor$的两部分。枚举第一部分的状态，求出$f_{i,j}$表示获得$i$颗钻石，不少于$j$美元的概率。再枚举第二部分状态，求出在第二部分中获得的美元和钻石数量，枚举第一部分中钻石数量和购买的物品数量，可以求出在第一部分中至少需要的美元数。在之前求得的f数组中查询即可获得概率，进而可求得期望。

时空复杂度

时间复杂度为$O(2^{N/2}NM)$。

空间复杂度为$O(2^{N/2}N)$。

1.24 MAY12 TICKETS

试题名称

Selling Tickets.

题目大意

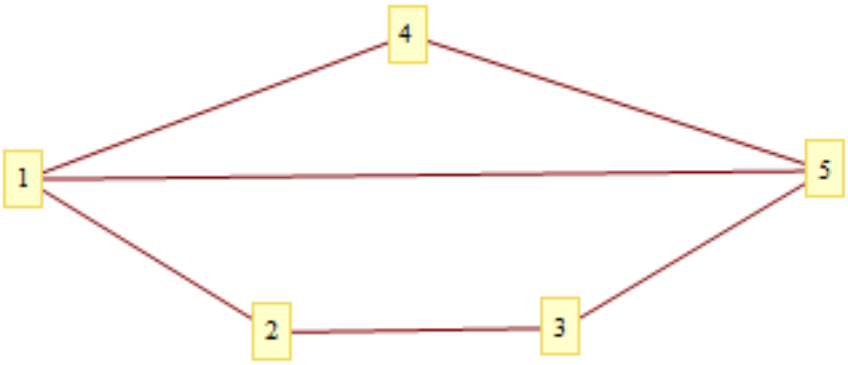
有N份菜品，M个可能会参加晚宴的人。已知第i个人要求他至少能得到菜品x_i或y_i中之一。求最小的K，满足存在一种方案，从M个人中选K个人参加晚宴，无法满足每个人的要求。

数据范围：$T \leq 15, 2 \leq N \leq 200, 0 \leq M \leq 500$。

算法讨论

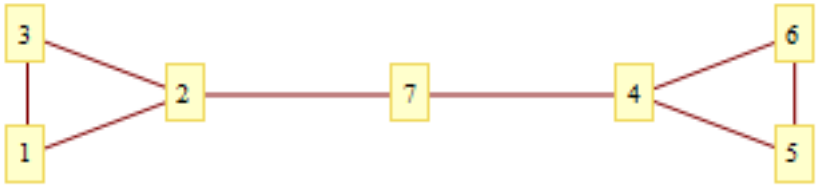
若把菜品、参加晚宴的人分别看做无向图上的点和边，问题可以转变为求这个无向图的一个边数最少的连通子图，满足边数比点数恰好多一。可以发现，这样的子图可以归为两类：

1. 点S与点T之间的三条路径。



当$S = 1, T = 5$时，一种方案如图所示。对于这一类子图，可以枚举点S, T，通过BFS，可以求出从点S到点T的三条最短路径长度和，取最小即可。

2. 两个由一条路径连接的环。



对于这一类子图，可以枚举中间路径上的一个点C，比如说，图中的点7。利用BFS，可以构出与C连通的点形成的生成树。对于一条非树边(u, v)，若点u, v均与点C连通，则这条非树边就对应着一个环，这个环可以对答案产生的贡献，为环的长度加上$LCA(u, v)$到点C的距离。将这样求出的所有环按照贡献值从小到大排序，选择贡献最小的两个环与全局答案比较即可。

时空复杂度

时间复杂度为$O(T(N^2M + NMlogN))$。
空间复杂度为$O(NlogN + M)$。

1.25 JUN12 COOLNUM

试题名称

Cool Numbers.

题目大意

对于一个长度为K的数n，其各个位上的数字为$X_1, X_2, \cdots, X_K$，若它存在一到三个数位上的数和为s，满足$(X_1 + X_2 + \cdots + X_K - s)^s$是$n$的倍数，则$n$为Cool Number。给出$T$组询问，每组询问给出一个数$N$，询问不超过$N$的最大Cool Number和超过$N$的最小Cool Number分别是多少。

数据范围：$T \leq 10^5$，$1 \leq N \leq 10^{1000}$。

算法讨论

对于一个数n，若其不为0的位数不超过3，则其显然为Cool Number，容易求得不超过x最大或超过x最小的这类Cool Numbers。

考虑另一类Cool Numbers，满足$n|((X_1+X_2+\cdots+X_K-s)^s)$且$(X_1+X_2+\cdots+X_K-s)^s \neq 0$，则有$(X_1+X_2+\cdots+X_K-s)^s \geq n$，因此$(9K-27)^{27} > 10^{K-1}$，可知$K \leq 77$。枚举$X_1+X_2+\cdots+X_K$和$s$，再用Dfs 枚举$X_1+X_2+\cdots+X_K-s$的约数，可以求出所有的这类Cool Numbers。直接枚举要在时限内完成有一些困难，因此可以部分打表。

处理询问时，对于第二类Cool Numbers，使用二分即可求得。

时空复杂度

时间复杂度为$O(Search + T(Len(N) + logC))$（其中$C$表示第二类Cool Numbers个数）。

空间复杂度为$O(Len(N) + C)$。

1.26 JUN12 MATCH

试题名称

Expected Maximum Matching.

题目大意

给一个有$N(N \leq 5)$个左边点，$M(M \leq 100)$个右边点的二分图。左边点i与右边点j之间有边的概率为$p_{i,j}$。求这个二分图的期望最大匹配数。

算法讨论

由于左边点数量很少，假若已经知道了这个二分图的连边情况，可以使用状压DP来求出最大匹配。于是，考虑在这个DP的基础上，在外层再套上一个DP，这个DP的状态为，在进行求最大匹配的DP中，所有左边点被匹配的状态是否可行。直接考虑状态数，共有2^{2^5}，无法承受。但可以发现，假若左边点被匹配的状态i可行，则i的子集也应可行，那么对于存在i的子集不可行的状态，可以删去。通过Dfs，可以求出所有有效状态，极限情况下数量不超过8000。预处理出状态之间的转移关系，再进行DP即可。

时空复杂度

时间复杂度为$O(8000M2^N + 8000N2^{2N})$。
空间复杂度为$O(8000M + M2^N)$。

1.27 JUL12 DGCD

试题名称

Dynamic GCD.

题目大意

给出一棵有$N(N \leq 50000)$个点的树，树上点带有权值，要求支持$M(M \leq 50000)$条操作，操作有以下两种：

- 询问点u到点v路径上的点权值的最大公约数。
- 将点u到点v路径上的点权值加上$d(d \geq 0)$。

算法讨论

对于一个序列$a_1, a_2, \cdots, a_k$，构造序列$b_1, b_2, \cdots, b_k$，满足$b_1 = a_1, b_i = a_i - a_{i-1}(1 < i \leq k)$。可以发现，$gcd(a_l, a_{l+1}, \cdots, a_r) = gcd(a_l, b_{l+1}, \cdots, b_r)$。

于是，对树进行轻重链剖分后，对于每个点维护一个辅助权值，为其重儿子的权值与自身权值之差。使用线段树维护支持区间加、单点询问的点权值和支持单点修改、区间gcd询问的辅助权值即可。

时空复杂度

时间复杂度为$O(Nlog^3N)$。
空间复杂度为$O(N)$。

1.28 JUL12 EST

试题名称

Equivalent Suffix Tries.

题目大意

对于一个字符串，可以构出由其所有后缀组成的trie树。给出字符串s，求有多少不同的字符串的trie树与其同构。

数据范围：$T \leq 10$，$N = |s| \leq 10^5$。

算法讨论

对于串s，设长度为L的后缀是最长的后缀，满足它是另一个后缀的前缀，则长度比L小的后缀，都存在另一个后缀，是这个后缀的前缀。因此trie树的叶节点个数为$N-L$，且每个叶节点都对应前$N-L$长的后缀，对于这$N-L$个后缀，都可以在trie树上找到其位置，因此对于合法字符串，任意第$i,j(i,j \leq N-L)$个后缀的LCP都应与原串相同。所以，若在原串中前$N-L$个字符中，某两个字符是相等的，则在所有合法串中，这两位都应是相等的。故可以认为合法串中的字母与原串对应，只需最后在答案中乘上$26 \times 25 \times \cdots \times (26-D+1)$，其中$D$为$s$中不同字符个数。则合法串中前$N-L$个字符已经被确定，只需考虑最后$L$个字符。

由于第$N-L+1$个后缀是前$N-L$个后缀中某一个的前缀，因此可以枚举具体是哪一个后缀P，于是便可以得到最后L个字符，此时还需要判断前$N-L$个后缀之间的LCP是否与原串相同。令$LCP(i,j)$表示第i个后缀与第j个后缀的LCP。若对于所有的$1 \leq i < N-L$，都有$LCP(i,N-L)$与原串相等，则对于所有

的$1 \le i, j < N - L$，都有$LCP(i, j)$与原串相等。当P的选择不同时，可能产生的最后L个字符是相同的，这时可用Hash去重。于是我们就已经得到了$O(N^2)$的算法。

令$Q = \max_{1 \le i < N-L} LCP(i, N - L)$，则合法字符串中第$[N - L + 1, N - L + Q - 1]$个字符也已经确定。对于$i(1 \le i < N - L)$，若$LCP(i, N - L) < Q$，则在合法字符串中已经能满足$LCP(i, N - L)$不改变；若$LCP(i, N - L) = Q$，则合法字符串中的第$i + Q$个字符应不同于第$N - L + Q$个字符，于是我们可以得到第$N - L + Q$个字符能够选择的字母集合。对于所有待选的$P$，我们可以容易地判断其是否能被选，之后我们还需要得到最后L个字符的Hash值。不妨采用多项式Hash的方式，即对于长度为K的字符串t，其Hash值$H(t) = \prod_{i=1}^{K} t_i \times base^{K-i+1}$。若$P+L \le N-L$，则可以直接得到最后$L$个字符的Hash值；否则，后缀$P$与后缀$N - L + 1$会有重叠部分，又由于这两个后缀的前$L$位都是相等的，因此后缀$P$的前$L$位可以划分为若干循环节（其中最后一个出现的循环节未必是完整的），循环节长度为$N - L - i + 1$，计算出循环节的Hash值后可以暴力计算L位的Hash值，复杂度上界为$O(\sum_{i=1}^{N} \frac{N}{i}) = O(NlogN)$。

时空复杂度

时间复杂度为$O(TNlogN)$。
空间复杂度为$O(N)$。

1.29 AUG12 MAGIC

试题名称

Two Magicians.

题目大意

给出一有N个点M条边的无向图，无重边无自环。两个魔术师要在这张图上玩一个游戏，初始时，各有P魔法值。两人轮流操作，操作有三步：

1. 沿当前已有的边走任意多步到达一个点。如果这个点恰好是另一个魔术师的位置，则胜利。

2. 增加一条原来没有的无向边。若无法增加，则失败。

3. 如果魔法值为整数，则可以消耗1点魔法值，移动到任意一点。

判断游戏的胜利者。
数据范围：$T \leq 100$，$N \leq 7777$，$M \leq 10000$，$0 \leq P \leq 10000$。

算法讨论

简化游戏的过程，可以看做在每一个时刻，有若干个连通块，当前操作者可在图中加一条边，不能使两个魔术师处在同一个连通块，同时还可以通过移动位置来阻碍对方操作，不能操作者失败。可以发现，在操作过程中，对于边的数量和连通块大小，我们只需关注它们的奇偶性。而要通过消耗魔法值来阻碍对方，当且仅当只存在一个连通块，奇偶性与移动的目标连通块相同，并且多次进行这样的移动是没有意义的，所以可以认为$P \leq 1$。于是可以得到一个复杂度较高的DP：令$dp[c0][c1][free][x][y][px][py]$来表示当前两个魔术师所在的连通块大小奇偶性分别为$x, y$，除去这两个连通块，有$c0$个大小为偶数的连通块，$c1$个大小为奇数的连通块，有$free$条可选的不改变连通性的边，两个魔术师的魔法值分别为px, py时，获胜者是先手还是后手。枚举每种转移的情况即可。这样的复杂度为$O(N^2)$，且空间需求较大。

通过获得当$c0, c1$值较小时的DP值，可发现，当$c0$较大时（比如说超过20），DP结果与$c0 = 20$时是相同的。而当$c1$较大时（也比如说超过20），DP值随$c1$增长呈循环节长度为4的循环。于是，我们可以把$c0, c1$的规模缩小到$O(1)$，问题解决。

时空复杂度

时间复杂度为$O(TN)$。
空间复杂度为$O(N)$。

1.30 AUG12 GTHRONES

试题名称

A Game of Thrones.

题目大意

有N种数字，第i种数字为A_i，共出现C_i次。A 和B正用这些数字玩一个游戏，A先操作，并选择一个数字。接下来，两人轮流操作，所选取的数字与上一个数字必须只相差一个质因子。无法操作者败。问先手是否存在必胜策略，若存在，输出最小的数字，满足在第一轮选取这个数字后能够必胜。

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

算法讨论

若选择了数字x后可以选择y，则$\frac{\max(x,y)}{\min(x,y)}$是质数，使用Miller-Rabin算法可以判断，之后对点(x,y)连上无向边，即形成了一个二分图。在任意一个最大匹配中孤立的点所代表的数字，就是先手第一步能够选择的数字。由于同种数字会出现很多次，考虑建立网络流模型：

- 从源点向左部点i连容量为C_i的边。
- 从右部点j向汇点连容量为C_j的边。
- 若在二分图中，左部点i与右部点j之间有边相连，则在网络中从i向j连容量为∞的边。

则若数字i可在第一步被选，需要满足在任意一个最大流中，源（汇）点与i之间的边未满流。若i是左部点，从源点出发，否则，从汇点出发，沿残余网络BFS，若能达到点i，则数字i可被选。统计即可。

时空复杂度

时间复杂度为$O(Flow(N, N^2))$。
空间复杂度为$O(N^2)$。

1.31 SEP12 KNGHTMOV

试题名称

Knight Moving.

题目大意

给一个无限大的方格棋盘，有一个骑士从$(0,0)$出发，希望走到(X,Y)。当它处在位置(u,v)时，可以选择跳到$(u+ax,v+ay)$或$(u+bx,v+by)$。 在这个棋盘上还设有K个障碍物。求到达(X,Y)的不同路线数对10^9+7取模的值，或返回路线数无穷。

数据范围：$K \leq 15$，输入中所有坐标绝对值均不超过500。

算法讨论

若向量(ax,ay)与(bx,by)线性相关，则原问题可以转化为一维问题。骑士在行进过程中，有效位置的坐标绝对值不会超过500^2。 从终点(X,Y)出发逆向BFS，求出范围内能够到达终点的点。然后再从$(0,0)$出发，使用Bellman-Ford进行DP，求出到每个位置的方案数，过程中只经过能够到达终点的点。若在Bellman-Ford过程中发现环，则答案就是无穷。

若向量(ax,ay)与(bx,by)线性无关，则骑士到达平面上任意一点时，所使用的两种跳跃方式次数是确定的。设到达点i时使用两种跳跃方式的次数分别为A_i,B_i。 则若忽略平面上的障碍物，方案数就是$\binom{A_i+B_i}{A_i}$。当平面上有障碍物时，可以使用容斥，强制一些障碍物被访问，将路线分成若干段，每段用组合数求出方案数，乘起来即可。

时空复杂度

时间复杂度为$O(500^2+2^K KlogK)$。

空间复杂度为$O(500^2+K)$。

1.32 SEP12 PARADE

试题名称

Annual Parade.

题目大意

给一有N个点，M条边的有向图。图上的第i条边费用为w_i。现在可以选取一些在图上的路径，并会产生一些费用：

- 对于每条路径，若起点与终点不同，则产生费用为C。

- 如果一条边i被经过k次，则产生的费用为kw_i。
- 如果一个点没有被任何一条路径经过，则产生的费用为C。

再给出K组询问，每组询问中的C值不同，表示询问在给定的C值下，选取路径的最小费用。

数据范围：$2 \leq N \leq 250, 1 \leq M \leq 30000, K \leq 10000$。

算法讨论

对于一个点，如果没有在任何一条路径中有出度，就会产生C的代价，于是就要考虑给尽量多的点寻找出度。如下所示建立费用流模型：

- 建立点$S, T, X_i, Y_i (1 \leq i \leq N)$。
- 从点S向点X_i连容量为1，费用为0的边。
- 从点Y_i向点T连容量为1，费用为0的边。
- 从点X_i向点Y_j连容量为1，费用为点i与点j之间的最短路径长度的边。

如果我们要为某一个点寻找出度，也就是找到一条从这个点出发的路径，从源点S出发寻找一条费用最小的增广路并增广即可。

由于在K组询问中C取值不同，可以将询问按C取值从小到大排序。如果一个点在C值较小的情况下有出度，那么当C值较大时，在最优情况下，这个点依然是有出度的。按顺序考虑每个询问，并多次在网络中进行增广，直到增广的费用超过当前C取值下一个点没有出度的代价。

时空复杂度

时间复杂度为$O(NM)$。

空间复杂度为$O(N^2)$。

1.33 OCT12 MAXCIR

试题名称

Max Circumference.

题目大意

给出一个三角形ABC，有N个操作可选，第i个操作有两个参数x_i, y_i，代表可把点A的x坐标加上x_i，y坐标加上y_i。现可以在N个操作中选择不超过K个进行使用，求三角形ABC的最大周长。要求绝对误差不超过10^{-12}。

数据范围：$K \leq N \leq 500$，$|x_i|, |y_i| \leq 10^6$，$|A_x|, |A_y|, |B_x|, |B_y|, |C_x|, |C_y| \leq 10^9$。

算法讨论

$|BC|$已经固定，因此只需最大化$|AB| + |AC|$。直接求距离和，涉及到平方根运算，因此需要进行转化。

存在函数$f(x, y) = ux + vy$，满足在最大化$f(A_x, A_y)$的同时，$|AB| + |AC|$最大。假设$\max\{|AB| + |AC|\} = D$，则取到这一最大值的A在以B, C为焦点的椭圆上，因此只需过A作椭圆的切线，即可求得一组满足条件的(u, v)的值。而每一个操作对$f(A_x, A_y)$的贡献都是独立的。因此，在已经确定了一组(u, v)的值后，只需将所有操作按$f(x_i, y_i)$降序排序，取贡献为正的前若干个（不超过K)更新答案即可。

对于操作i和j，可以求出一条直线，满足将平面分成两部分，当(u, v)在其中一部分时，$f(x_i, y_i) \leq f(x_j, y_j)$；在另一个部分时，$f(x_i, y_i) \geq f(x_j, y_j)$。将分界线极角排序后，维护操作$f$值大小顺序即可。注意，可能有多个操作，在同一条分界线交换顺序，此时，不可以按随意顺序交换，而可以将一些涉及到的操作区间翻转。

最后，由于精度要求很高，需要注意几点：

- 对分界线进行极角排序时，不可直接使用atan2等三角函数，而可以分象限叉积确定顺序。
- 计算平方根时，由于要求精确的整数部分和小数部分都较长，因此可以将整数部分和小数部分分开计算和存储。
- 最终答案的整数部分超过C++中long long的上界，需要使用unsigned long long。

时空复杂度

时间复杂度为$O(N^3)$（若使用数据结构优化求答案的过程，可将复杂度降至$O(N^2logN)$）。

空间复杂度为$O(N^2)$。

1.34 NOV12 COUNTARI

试题名称

Arithmetic Progressions.

题目大意

给出$N(N \leq 10^5)$个整数$A_1, A_2, \cdots, A_N(A_i \leq M, M \leq 30000)$，求三元组$(i, j, k)(1 \leq i < j < k \leq N)$的个数，满足$A_j - A_i = A_k - A_j$。

算法讨论

对条件移项，得$A_k = 2A_j - A_i$。

考虑先枚举j，再枚举i，同时用数组维护满足$A_k = 2A_j - A_i$的k个数，可以做到$O(1)$查询，总复杂度$O(N^2)$。

若在枚举的时候不枚举i，而只枚举A_i，则复杂度可以降至$O(NM)$。但这样的复杂度还是不足以通过本题。

考虑使用指针优化，即对w个A_i的取值并行计算。当w取一个不大的值时，优化效果良好，可以使复杂度降至原复杂度的$O(\frac{1}{w})$。 当取$w = 16$左右的时候，足以通过本题。

时空复杂度

时间复杂度为$O(\frac{NM}{w})$。

空间复杂度为$O(N + M)$。

1.35 NOV12 MARTARTS

试题名称

Martial Arts.

题目大意

给出一个两部分各有$N(N \leq 100)$个点的完全二分图，图上的边带有两个边权$H_{i,j}, G_{i,j}$。要求图的一个匹配，满足在去掉了一个匹配(x, y)，先满足$H_{x,y} - G_{x,y}$最大再满足$G_{x,y}$最小的情况下，剩余匹配的H权值和与G权值和之差尽量大，再满足H尽量大。

算法讨论

首先推荐一篇与本题相关的论文：*The Dynamic Hungarian Algorithm for the Assignment Problem with Changing Costs*。

对于原图中的边(x, y)，我们可以用一个pair类型来重新定义其权值，为$(H_{x,y} - G_{x,y}, H_{x,y})$。将边按被删除的优先级排序，依次加入图中，并强制新加入的边在匹配中（通过将该边权值改为无穷来实现），使用KM 算法可以求出最大权完美匹配，与答案比较即可。一次匹配复杂度为$O(N^3)$，有N^2条边需要加入，因此复杂度为$O(N^5)$，无法满足要求。

参考之前所推荐的论文，可知，在新加入一条边后，没有必要完整地重新运行KM算法。设我们新加入的边为(x, y)，此时只需要调整点x的标号，使得完美匹配存在（这一点在KM算法的介绍中可知），并使原先与点x相关的匹配失配，再从点x出发寻找增广路即可。

时空复杂度

时间复杂度为$O(N^4)$。

空间复杂度为$O(N^2)$。

1.36 DEC12 DIFTRIP

试题名称

Different Trips.

题目大意

给出一棵有$N(N \leq 10^5)$个点的有根树。称两个点是相似的，当且仅当它们的度相同。称两条路径是相似的，当且仅当这两条路径长度相同，且经过的点对应相似。

求两两不相似的最多可选路径条数，满足路径的其中一个端点是另一个端点的祖先。由同一对点组成的路径只算作一次。

算法讨论

考虑将每个点的度看做点上的一个字符，那么，从根到每个点的路径上的字符连接，可以形成字符串。问题转变为询问这些字符串不同子串数目。从根出发Dfs这棵树，Dfs的同时，可以构建出这些字符串形成的后缀自动机。将后缀自动机上的每个状态的最大可达长度减去父亲状态的最大可达长度求和，就是要求的答案。

时空复杂度

时间复杂度为$O(NlogN)$。
空间复杂度为$O(N)$。

1.37 JAN13 ANDOOR

试题名称

A New Door.

题目大意

给出一个黑色平面，和四个顶点分别位于$(0,0),(W,0),(0,H),(W,H)$的一个白色内部的矩形，然后再给出$N(N \leq 1000)$个黑色内部的圆，求矩形内部黑色部分轮廓的长度。

算法讨论

对于每个圆分别考虑。若这个圆在矩形外部，或者被另一个圆包含，则跳过这个圆。否则，维护这个圆被其他圆覆盖或在矩形外部的弧的角度区间，将这些区间排序后，即可算出这些区间的总长度。将圆的周长减去这些区间的总长度，就是对答案的贡献，累加即可。

时空复杂度

时间复杂度为$O(N^2logN)$。
空间复杂度为$O(N)$。

1.38 JAN13 CUCUMBER

试题名称

Cucumber Boy and Cucumber Girl.

题目大意

给出B个$N \times N$的矩阵$Q_1, Q_2, \cdots, Q_B$。 对于每对$(x, y)(1 \leq x < y \leq B)$，构造一个矩阵$A_{x,y}$，满足：

$$A_{x,y}[i][j] = \sum_{k=1}^{N} Q_x[i][k] \times Q_y[j][k] \ \ (for\ 1 \leq i, j \leq N)$$

求数对(x, y)的个数，满足存在一个$1 \sim N$的排列$p_1, p_2, \cdots, p_N$，使得$A_{x,y}[i][p_i]$中至少有一个是奇数。

数据范围：$1 \leq N \leq 60$，$2 \leq B \leq 8000$.

算法讨论

考虑数对(x, y)满足要求的条件，尝试将问题转化为行列式上的问题。对于数对(x, y)，尝试构造一个矩阵$C_{x,y}$，使得可以通过$\det C$的奇偶性，来判断数对(x, y)是否满足条件。

$$C_{x,y}[i][j] = Q_x[x][i] \times Q_y[y][j] + 1$$

若数对(x, y)满足条件，则$N! - \det C_{x,y}$为奇数。当$N \geq 2$时，$N!$为偶数，因此只要判断$\det C_{x,y}$是否为奇数即可；而当$N = 1$时，原问题可以简化，直接特判即可。使用位运算优化的高斯消元，直接求出所有的$\det C_{x,y}$复杂度为$O(B^2N^2)$，无法承受。假定以下操作均在模2意义下进行。

对于$1 \le x \le B$，再构造一个$N \times N+1$的矩阵D_x，满足：

$$D_x[i][j] = \begin{cases} Q_x[i][j] & for\ 1 \le i,j \le N \\ 1 & for\ j = N+1 \end{cases}$$

则$C_{x,y} = D_x \times D_y^T$。

根据*Cauchy - Binet 公式*，设A是一个$m \times n$ 的矩阵，而B是一个$n \times m$的矩阵，令$[n]$表示集合$\{1, 2, \cdots, n\}$，$\binom{[n]}{m}$表示所有大小为m的$[n]$的子集，则：

$$\det(AB) = \sum_{S \in \binom{[n]}{m}} \det(A_{[m],S}) \det(B_{S,[m]})$$

简化地来说，若$m > n$，则$\det(AB) = 0$；否则就枚举从n列中删去$n-m$列，对A, B两个矩阵分别删去这些列后的行列式积相加，就是$\det(AB)$。

而$\det A = \det A^T$，设$E_{x,y}$为矩阵D_x删去第y列后得到的矩阵，则有：

$$\det C_{x,y} = \sum_{i=1}^{N+1} \det(E_{x,i}) \times \det(E_{y,i})$$

算出所有的$\det(E_{x,i}) \bmod 2$后，枚举(x, y)，使用位运算即可判断$\det C_{x,y}$的奇偶性。

考虑如何预处理$\det(E_{x,i})$。先对矩阵D_x使用高斯消元使之成为*行最简形(Row echelon form)*。若D_x的秩小于N，则$\det(E_{x,i}) = 0$；否则，化为行最简形后的矩阵D_x形如一个单位矩阵加上一列，设这一列为c。

$$E_{x,i} = \begin{cases} D_x[i][c] & for\ i < c \\ 1 & for\ i = c \\ 0 & for\ i > c \end{cases}$$

至此问题终于解决！

时空复杂度

时间复杂度为$O(B^2 + BN^2)$。

空间复杂度为$O(BN^2)$。

1.39 FEB13 ROC

试题名称

Room Corner.

题目大意

给出一个边界水平或竖直的多边形，这个多边形满足若一条水平线段的两个端点在多边形内，则整条线段在多边形内。在每个90 度内角处，都有一个小朋友。这些小朋友沿着多边形的边界可以形成一个环，环上相邻的两个小朋友能够直接交换位置。小朋友移动速度为每秒一单位。给出Q组询问，每组询问给出两个小朋友x, y，问交换这两者位置时，相遇的最短时间。

有关题面的更多细节，请参见原题描述。

数据范围：$Q \leq 10000$，多边形可被一个$N \times M$的矩形包含，其中$N \leq 2500, M \leq 2500$。

算法讨论

首先需要找到小朋友们在环上的顺序。由于这个多边形的特殊性质，每一行最多只有两个小朋友。也就是说，我们可以从第一行到第N行扫一遍，提取可能存在的左边的小朋友，再从第N行到第一行扫一遍，提取右边的小朋友，同时可以求出小朋友之间的距离。

处理询问时，枚举它们相遇的边，设长度为y，通过前缀和可以算出两者相遇前花费的时间x，则花费总时间为$x + y/2$，取最小值即可。

时空复杂度

时间复杂度为$O(QN + NM)$。

空间复杂度为$O(NM)$。

1.40 FEB13 QUERY

试题名称

Observing the Tree.

题目大意

给出一棵有$N(N \leq 10^5)$个点的树，点i的权值为w_i，要求支持$M(M \leq 10^5)$条操作，操作有三种：

- 提取点u到点v最短路径上的所有点，按经过的顺序排成一个序列$p_1, p_2, \cdots, p_K (p_1 = u, p_K = v)$。 则对于$1 \leq i \leq K$，将$w_{p_i}$加上$a + b(i-1)$。
- 询问点u到点v最短路径上所有点的权值和。
- 将状态恢复到第x条修改操作后。

操作强制在线。

算法讨论

我们先来考虑如何在序列上进行操作。对于一个关于区间$[l, r]$的Add操作，可以转化为对于$w_i (l \leq i \leq r)$，w_i加上$ik + b$。 可以使用一棵线段树，分别维护所有修改操作k和b的和。由于题目还要求对数据结构可持久化，而修改操作又涉及区间修改，若采用传统的打标记做法，则每次标记下传时，儿子节点都要重新新建，带来较大的空间浪费。在这里，修改操作不复杂，多个操作可以很容易地合并，因此实际上，可以采用标记不下传的“标记永久化”的方式，即在每次询问时，将在线段树上所经过的点上的标记叠加，从而免去了下传标记的步骤。

在树上进行操作时，只需要将树进行轻重链剖分，就可以将树上问题转化为在一维序列上的问题了。

时空复杂度

时间复杂度和空间复杂度均为$O(Nlog^2N)$。

1.41 MAR13 LECOINS

试题名称

Little Elephant and Colored Coins.

题目大意

给出N种硬币，第i种硬币面值为V_i，颜色为C_i，每种硬币无限多。再给出Q条询问，第i条询问要求选出一些硬币，它们的面值和恰为S_i，且颜色种数尽量多。

数据范围：$N \leq 30, V_i \leq 200000, C_i \leq 10^9, S_i \leq 10^{18}$。

算法讨论

注意到S_i很大，但V_i并不大。考虑选出一种硬币p作为基准硬币，使用其他硬币拼出的面值为x，满足$x \leq S_i$且$S_i \equiv x (\mathrm{mod}\ V_p)$，那么就可以通过加入一些$p$硬币，使得拼出面值为$S_i$。于是可以考虑用$dp_{i,j,k}$表示现在使用的硬币有$i$种不同的颜色，拼出面值在模$V_p$意义下为$j$，是否使用了和硬币$p$同色的硬币（用01数$k$来表示）时的最小总面值。回答询问时，枚举颜色数$i$，和硬币$p$的颜色使用情况$k$，将$dp_{i,S \bmod V_p,k}$与$S$进行比较，当$dp_{i,S \bmod V_p,k} \leq S$时，可更新答案。

时空复杂度

时间复杂度为$O(N^2V_i)$。

空间复杂度为$O(N^2V_i)$。

1.42 APR13 STRQUERY

试题名称

String Query.

题目大意

要求维护一个字符串，支持在头、尾、中加入或删除一个字符，并查询另一个字符串在该串中的出现次数。

数据保证任何时刻，字符串长度不短于10，操作数Q不超过150000，所有询问串长度和不超过1500000。

算法讨论

后缀平衡树是一棵可以按大小维护一个字符串所有后缀的数据结构，用treap实现的后缀平衡树是重量平衡的。后缀平衡树可支持在一个字符串

的头部加入或删除字符。在加入操作时，需要支持快速比较两个后缀大小，这可以通过在平衡树上为每个节点维护标记$[tag_l, tag_r]$，则其两个儿子的标记分别为$[tag_l, (tag_l + tag_r)/2]$和$[(tag_l + tag_r)/2, tag_r]$。当要比较两个节点的前后顺序时，只需要比较$tag_m = (tag_l + tag_r)/2$的大小即可。在删除操作时，可以合并删除点的儿子，并连接到父亲节点上，再对以其父亲为根的子树重新计算tag。若要询问一个字符串在所维护的串中的出现次数，可在后缀平衡树上行走，暴力比较后缀与询问串间的大小关系。

对于此题，可维护四个后缀平衡树LL, LR, RL, RR，将字符串划分为4个部分，其中LR和RR中的字符串与原串顺序相反，于是便可以支持在三个位置的插入和删除。若在某一时刻，发现LL为空，则可以重构LL和LR，均分原来LR中的字符，对于其他树为空的情况也类似。在询问时，若整个询问串在LL, LR, RL, RR中的某一段，则可以直接在后缀平衡树中询问。否则，询问串在原串中的出现位置跨过分界线，此时可在两边分别暴力提取小于询问串长度的串，再用KMP进行匹配。

时空复杂度

时间复杂度为$O(Q)$。
空间复杂度为$O(Q)$。

1.43 MAY13 QTREE

试题名称

Queries on tree again!

题目大意

给出一有N个点，N条边的连通无向图，其中的边带有权值。要求支持M条操作，操作有以下两种类型：

- 将点u到点v最短路径上的边权值取反。
- 按顺序提取点u到点v最短路径上的边的权值，形成一个序列，询问这个序列的权值和最大的可空连续子序列的权值和。

数据范围：$N, M \leq 10^5$，并且保证任意两点间的最短路唯一。

算法讨论

题目中给出的图为环加外向树，将树上部分进行轻重链剖分后，将点按DFS序编号。特殊的，令在环上的点DFS序连续。用线段树维护区间权值和最大、最小可空连续子序列的权值和，同时维护区间前缀和后缀的最大、最小答案。对于询问和修改操作，都可以与普通树链剖分类似地，从两个操作点向上跳，同时在线段树上做区间询问或修改。

时空复杂度

时间复杂度为$O(Nlog^2N)$。
空间复杂度为$O(N)$。

1.44 JUN13 TKCONVEX

试题名称

Two k-Convex Polygons.

题目大意

给出$N(N \leq 1000)$条边，长度分别为$A_1, A_2, \cdots, A_N$。要求从中选出$2K(3 \leq K \leq 10)$条边，构成两个分别有K条边的凸包。求一种方案。

算法讨论

对于K条边$x_1, x_2, \cdots, x_K(x_1 \leq x_2 \leq \cdots \leq x_K)$，若能构成一个凸包，当且仅当：

$$\sum_{i=1}^{K-1} x_i > x_K$$

首先可以将所有边按长度升序排序。枚举两个凸包中最长的两条边编号，假设分别为$L, R(L < R, L \geq K, R \geq 2K)$。若$R - L \geq K$，则对于第一个凸包，选择编号$[L - K + 1, L]$的边，对于第二个凸包，选择编号为$[R - K + 1, R]$的边，再判断是否可行。否则，若$R - L < K$，则第二个凸包中的一些边可能来自于$[1, L)$。在这两个凸包中，边的编号范围都处于$[R - 2K + 1, R]$。 在这$2K$条边中，枚举哪些边属于第一个凸包，剩下的边自然属于第二个凸包，判断是否可行即可。

时空复杂度

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

1.45 JUN13 SPMATRIX

试题名称

Count Special Matrices.

题目大意

给出$N(3\leq N\leq 10^7)$，求满足以下条件的$N*N$的矩阵A个数：

- $A_{i,i}=0(1\leq i\leq N)$.
- $A_{i,j}=A_{j,i}(1\leq i<j\leq N)$.
- $A_{i,j}\leq \max(A_{i,k},A_{k,j})(1\leq i,j,k\leq N)$.
- $A_{i,j}\in\{1,2,\cdots,N-2\}(1\leq i<j\leq N)$.
- $\forall k\in\{1,2,\cdots,N-2\},\exists x,y\in[1,N],A_{x,y}=k$.

共有10^5组询问，答案对10^9+7取模。

算法讨论

首先通过指数级复杂度的Dfs搜索，可以求得当N较小时的答案：$1,13,205,4245,\cdots$。将答案序列的前几项输入OEIS进行查询，可以找到数列*Triangle of coefficients from fractional iteration of e^x-1*。这个数列对应以下具有递推关系的多项式的系数：

$$a(1)=1$$

$$a(n)=\sum_{k=1}^{n-1}\left\{{n\atop k}\right\}a(k)x\quad(n>1)$$

而要求的答案序列$f(n)$为$[x^{n-2}]a(n)$。观察$a(n)$的递推关系，可得：

$$f(n)=\sum_{i=2}^{n-1}\left(\left\{\begin{matrix}i+1\\i-1\end{matrix}\right\}\prod_{j=2}^{i-1}\left\{\begin{matrix}j\\j-1\end{matrix}\right\}\prod_{j=i+2}^{n}\left\{\begin{matrix}j\\j-1\end{matrix}\right\}\right)$$

容易发现：

$$\left\{\begin{matrix}n\\n-1\end{matrix}\right\}=\binom{n}{2}$$

$$\left\{\begin{matrix}n\\n-2\end{matrix}\right\}=\binom{n}{3}+3\binom{n}{4}$$

进一步还可以得到$f(n)$的递推关系：

$$f(3)=1$$

$$f(n)=\binom{n}{2}f(n-1)+\left(\prod_{k=2}^{n-1}\binom{k}{2}\right)\left(\binom{n}{3}+3\binom{n}{4}\right)\quad(n>3)$$

预处理出$g(n)=\prod_{k=2}^{n-1}\binom{k}{2}$，即可线性递推求得$f(n)$。

时空复杂度

时间复杂度为$O(N)$。
空间复杂度为$O(N)$。

1.46 JUL13 RIVPILE

试题名称

Across the River.

题目大意

在二维平面中有N个点$(x_1,y_1),\cdots,(x_N,y_N)$。有$M$种圆盘可供使用，第$i$种圆盘半径为$R_i$，使用代价为$C_i$。 现在可以以$N$个点中的一些点为圆心，放置一些圆盘，使得能够从直线$y=0$上的某一个点经过圆盘走到直线$y=W$上的某一点。求最小代价。同一测试点中有T组数据。

数据范围：$T\leq 10,N,M\leq 250$。

算法讨论

令$f_{i,j}$为现在走到以点i为圆心，类型为j的圆盘上，需要花费的最小代价。首先对于两种圆盘p和q，若$R_p < R_q$，而$C_p \geq C_q$，则圆盘q是多余的，可以被删去。类似于最短路问题，可以使用$Dijkstra$算法。在从$f_{p,q}$转移到另一点i时，通过二分查找，可以得到以i为圆心的圆盘的最小半径，使得从点p可以走到点i。然后，找到尚未从$f_{i,k}$出发转移的最小圆盘种类编号k，满足半径至少是之前求出的最小半径，则$f_{p,q}$会对$f_{i,k}, f_{i,k+1}, \cdots, f_{i,M}$都可能产生贡献，打上标记，在下一次从点$i$出发转移时传递标记即可。其中，若$f_{i,x}$能够成为$f_{i,1}, \cdots, f_{i,M}$中代价最小的尚未被转移过的值，则容易知道$k = x$，直接检查$f_{i,x}$是否满足即可。

时空复杂度

时间复杂度为$O(TN^3logN)$。
空间复杂度为$O(TN^2)$。

1.47 AUG13 LYRC

试题名称

Music & Lyrics.

题目大意

给出$W(W \leq 500)$个长度不超过5000的模板字符串，再给出$N(N \leq 100)$个长度不超过500000 的询问串。

对于每个模板串，回答它在所有询问串中以子串形式出现的总次数。

算法讨论

首先由模板字符串构建AC自动机。

对于每个询问串，使用AC自动机去匹配这个串，记录下自动机上的每个节点被匹配的次数。当AC自动机上的一个节点被匹配一次，这个节点沿着fail指针到根的所有节点的匹配次数都应累加。具体的被匹配的次数可以采用在匹配的时候打标记，最后沿着由fail 关系构成的树dfs 一遍得到。

一个模板串在所有询问串中以子串形式出现的总次数，就是它的最后一个字符在AC 自动机上的节点被匹配的次数。

时空复杂度

时间复杂度和空间复杂度都是线性的。

1.48 AUG13 PRIMEDST

试题名称

Prime Distance On Tree.

题目大意

给出一棵有$N(N \leq 50000)$个点的树，求等概率随机选择两个不同点，这两个点之间的距离为质数的概率。

算法讨论

本题是树上点分治的经典应用。假设在分治中，当前处理的子树的根是$root$。Dfs这棵子树得到根到每个点的距离dis_i，求出子树中点对$(x,y)(x \neq y)$的数量，满足$dis_x + dis_y$为质数。由于点x和y可能属于以点$root$的某个儿子为根的子树，此时两点的LCA不为$root$，需要除去这些点对，这可以通过减去以$root$的每个儿子为根的子树中，满足$dis_x + dis_y$为质数的点对$(x,y)(x \neq y)$的数量。

如何快速地求一棵子树内满足$dis_x + dis_y$为质数的点对$(x,y)(x \neq y)$的数量呢？设$dis_x = k$的点x的数量为c_k。 建立多项式$F(x) = \sum_k c_k x^k$。 通过FFT求卷积，可得：

$$\sum_{\texttt{k is prime}} [x^k]F^2(x)$$

除去点对$(x,y)(x = y)$产生的多余答案，就是需要求的点对数。

最后将求得的点对数的和除上选择点对的总方案数$N(N-1)$，就是要求的概率。

时空复杂度

时间复杂度为$O(Nlog^2N)$。

空间复杂度为$O(N)$。

1.49 SEP13 TWOROADS

试题名称

Two Roads.

题目大意

给出平面上$N(3 \leq N \leq 100)$个点，其中无三点共线。要求构造两条直线，每一个点的权值为该点到两条直线距离的较小值，求所有点权值平均值的最小值。

算法讨论

对于两条相交的直线，形成了四个夹角，作这四个夹角的角平分线，得到两条相互垂直的直线，将N个点分成四个部分，其中相对的两个部分会选择同一条直线。枚举两个点，过这两个点作直线作为其中一条分界线，另一条分界线可以采用扫描的方式，同时维护两个分别选择同一直线的部分，求出它们中的点的x, y坐标和、二次方和，xy和以及点个数，利用二次方程根判别式可以求出最小点权值和。

时空复杂度

时间复杂度为$O(N^3 logN)$。
空间复杂度为$O(N)$。

1.50 OCT13 FN

试题名称

Fibonacci Number.

题目大意

斐波那契数列满足：$fib_0 = 0, fib_1 = 1, fib_n = fib_{n-1} + fib_{n-2}(n > 1)$。
给定一个素数p和一个非负整数C，求最小的n，满足：

$$fib_n \equiv C(\mathrm{mod} P)$$

数据范围：$11 \leq p \leq 2 * 10^9, 0 \leq C \leq p-1$，$(p \bmod 10)$是一个完全平方数，共有$T = 100$组询问。

算法讨论

定义 1.50.1. 称a为模m的二次剩余*(Quadratic residue)*，当且仅当存在整数x，满足：

$$x^2 \equiv a(\mathrm{mod}m) \quad (a,m) = 1$$

定理 1.50.1. 欧拉准则*(Euler's criterion)*：设p是奇素数，$(a,p) = 1$，则：

1. a为模p的二次剩余的充分必要条件是：

$$a^{(p-1)/2} \equiv 1(\mathrm{mod}p)$$

2. a为模p的二次非剩余的充分必要条件是：

$$a^{(p-1)/2} \equiv -1(\mathrm{mod}p)$$

当a是模p的平方剩余时，同余式$x^2 \equiv a(\mathrm{mod}p)$恰有两解。

定理 1.50.2. 设p为奇素数，则模p的简化剩余系中二次剩余与二次非剩余的个数各为$(p-1)/2$，且$(p-1)/2$个二次剩余与序列

$$1^2, 2^2, \cdots, (\frac{p-1}{2})^2$$

中的一个数同余，且仅与一个数同余。

定义 1.50.2. 设p是素数，定义勒让德符号*(Legendre symbol)*如下：

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{若}a\text{是模}p\text{的二次剩余} \\ -1 & \text{若}a\text{是模}p\text{的非二次剩余} \\ 0 & \text{若}p \mid a \end{cases}$$

定理 1.50.3. 当p是奇素数时，有：

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2}(\mathrm{mod}p)$$

定理 1.50.4. 勒让德符号还存在如下性质：

$$\left(\frac{a}{p}\right) = \left(\frac{a \bmod p}{p}\right)$$

$$\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$$

若p, q是不相同的奇素数，则：

$$\left(\frac{q}{p}\right) = (-1)^{(p-1)(q-1)/4}\left(\frac{p}{q}\right)$$

定义 1.50.3. 设$m = \prod_{i=1}^{k} p_i$，定义雅可比符号*(Jacobi symbol)*为：

$$\left(\frac{a}{m}\right) = \prod_{i=1}^{k}\left(\frac{a}{p_1}\right)$$

回到原题，我们知道，斐波那契数列的通项公式为：

$$fib_n = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right)$$

如果我们能够找到一个整数s，来表示模p意义下$\sqrt{5}$的值，问题就可以转变成求同余方程的最小整数解。通过求得勒比得符号$(\frac{5}{p})$的值，可以判断这样的s是否存在。

$$\left(\frac{5}{p}\right) = (-1)^{p-1}\left(\frac{p}{5}\right) = \left(\frac{p \bmod 5}{5}\right)$$

由于$(p \bmod 10)$是一个完全平方数，故$p \bmod 5 = -1, 1$。

$$\left(\frac{-1}{5}\right) = \left(\frac{1}{5}\right) = 1$$

因此，5总是模p的二次剩余。要求方程

$$s^2 \equiv 5 (\mathrm{mod} p)$$

的解，设求得p的原根为g，可将方程两边取指标，从而将二次同余方程转化为线性同余方程：

$$2ind_g s \equiv ind_g 5(\mathrm{mod}\phi(p))$$

解这个线性同余方程，即可求得s的值。

设$\alpha = (1+\sqrt{5})/2, \beta = (1-\sqrt{5})/2, C' = \sqrt{5}C$，现在要求同余方程

$$\alpha^n - \beta^n \equiv C'(\mathrm{mod}p)$$

的最小整数解。

$$\beta = \frac{-1}{\alpha}$$

因此，上面的方程也可以写成：

$$\alpha^n - \left(\frac{-1}{\alpha}\right)^n \equiv C'(\mathrm{mod}p)$$

设$y = \alpha^n$。对n的奇偶性分情况讨论。若n为偶数，则上面的方程可以化为：

$$y - \left(\frac{1}{y}\right)^n \equiv C'(\mathrm{mod}p)$$

这是关于y的一元二次方程。

$$y = \frac{C' \pm \sqrt{C'^2+4}}{2}$$

若C'^2+4是模p的二次剩余，则y存在整数解，此时可用*BSGS(Baby-Step-Giant-Step)*算法求出满足奇偶性的n的具体值。对于n为奇数情况，方法类似，这里不再赘述。将求得的多个n的取值取最小值，就是所求的答案。

时空复杂度

时间复杂度为$O(T\sqrt{p})$。

空间复杂度为$O(\sqrt{p})$。

1.51 OCT13 DEG3MAXT

试题名称

Three-Degree-Bounded Maximum Cost Subtree.

题目大意

给出一有n个点，m条带权边的无向图，满足其任意点双连通分量大小不超过9。求该图的3-度数限制最大生成树（不一定要包含所有点）的边权和及方案数。

数据范围：$n \leq 100$，$m \leq 450$。

算法讨论

使用Tarjan算法可以求出该图的所有点双连通分量，在此过程中，可以对每个求出的点双进行DP。令f_i表示当前点双内的点度数状态为i时，由度数不为零的点构成的生成树的最大权值和及方案数。不妨认为编号最大的点为这棵生成树的根，找到这其中编号最小的点，若该点度数为1，则该点为叶子节点，枚举其父亲后可转移到不包含该点的状态；否则，枚举当前点集的一个子集来作为生成树中该点的子树，同时枚举该点的父亲，生成树被划分为两个更小的部分，合并即可。注意在此过程中，需要跳过无效的度数状态。之后，在点双上处理得到的DP值，可以传递到该点双的根上，并在这些割点上记录度数为不同取值时的答案，再向上合并即可。

时空复杂度

时间复杂度不超过$O(12^9 n)$（实际复杂度远未达到该水平）。

空间复杂度为$O(4^9)$。

1.52 NOV13 MONOPLOY

试题名称

Gangsters of Treeland.

题目大意

给出一棵有$N(N \leq 10^5)$个点的有根树。初始时，每个点颜色不同。要求支持$Q(Q \leq 10^5)$条操作，操作有两种，如下：

- 给出点u，要求将u到根路径上的所有点的颜色，都染成点u的颜色。
- 给出点u，询问以点u为根的子树中的所有点，到根的路径上，两端点颜色不同的边的数目的平均值。

算法讨论

可以注意到，第一种操作与Link-Cut Tree中的Access过程十分相似。不妨我们就把第一种操作看做是Access操作，再来观察第二种操作。一个点到根路径上，两端点颜色不同的边，也就是Link-Cut Tree 上所经过的虚边。这样的边的数目，也就是所经过的虚边的数目。

考虑维护一棵Link-Cut Tree，在Access操作过程中，会将某两个点间的虚边变为实边，也会将某两个点间的实边变为虚边，这两者可逆。我们考虑前者。设点u和点v之间的边变为实边，其中点u在有根树中是点v的父亲，那么会产生的影响，也就是以点v为根的子树中的所有点，到根所经过的虚边数目减少一。若要把实边变为虚边，也就是虚边数目增加一。于是，我们可以在维护Link-Cut Tree 的同时，维护每个点到根路径上的虚边数目，和一棵子树内所有点到根路径上的虚边数目的和。子树的操作，可以通过Dfs序转化为序列上的操作。这时使用线段树或树状数组，来支持对序列的一个区间进行加减，并询问一个区间中元素的和即可。

时空复杂度

根据Link-Cut Tree的相关性质，一次修改操作会改变实虚的边的数目是$O(logN)$的，改变实虚还需要在线段树或树状数组上进行修改，复杂度也为$O(logN)$。故单次修改的复杂度为$O(log^2N)$。 询问时只需要在线段树或树状数组上询问，单次复杂度为$O(logN)$。 因此，总时间复杂度为$O(Qlog^2N)$。

空间复杂度为$O(N)$。

1.53 NOV13 QPOINT

试题名称

Queries With Points.

题目大意

给出平面上N个简单多边形，不同多边形之间不相交且不包含。要求在线回答Q组询问，每组询问需回答点(x,y)在哪个多边形内。

数据范围：多边形总点数K不超过300000，$Q \leq 100000$。

算法讨论

若询问不要求在线，则可以使用扫描线和平衡树。当询问要求在线时，只需要将平衡树可持久化，可以使用基于merge和split的函数式treap。

时空复杂度

时间复杂度为$O(KlogK + QlogK)$。

空间复杂度为$O(KlogK)$。

1.54 DEC13 QTREE6

试题名称

Query on a tree VI.

题目大意

给一棵有$N(N \leq 10^5)$个点的树，每个点的颜色可以为黑色或白色。初始时，每个点都是黑色的。称两个点u,v是连通的，当且仅当两个点路径上的所有点颜色相同。现在需要支持$M(M \leq 10^5)$条操作，操作有以下两种：

- 询问和点u连通的点数。
- 修改点u的颜色。

算法讨论

考虑将树进行轻重链剖分。对于每个点，维护轻儿子中，分别只经过黑点、白点能到达的点数。同时，用树状数组维护每个点轻儿子中和该点连通的点数。

定义$count(u)$为以u为根的子树中，和点u连通的点数。这个值可以通过在树状数组上求和，求出点u所在重链上的每个点的轻儿子中，和这些点连通的点数，再加上重链的长度即可。

对于修改操作，设修改颜色的点为u。则轻儿子中和点u连通的点数，变为只经过同修改后的颜色相同的点所能到达的点数，这个值在之前已经维护。然后，沿着点u到根的路径向上跳，对于经过的每一条轻边，更新轻边靠根一端的点所维护的答案。更新答案时，应用函数$count(v)$即可。

对于询问操作，设询问的点为u，找到点u到根路径上，最浅的和点u连通的点v。这可以通过用线段树维护点的颜色，沿着树链向上跳，同时在线段树上询问来实现。于是，和点u所连通的点数就是$count(v)$。

时空复杂度

时间复杂度为$O(Nlog^2N)$。

空间复杂度为$O(N)$。

1.55 DEC13 REALSET

试题名称

Petya and Sequence.

题目大意

给出序列$A_0, A_1, \cdots, A_{n-1}$，试判断是否存在元素不全为0的序列$B_0, B_1, \cdots, B_{n-1}$，满足对于所有的$0 \leq j < n$，都有$A_0B_j + A_1B_{(j+1) \bmod n} + \cdots + A_{n-1}B_{(j+n-1) \bmod n} = 0$。

数据范围：$1 \leq T \leq 100$，$1 \leq n \leq 3 \times 10^4$，$-1000 \leq A_i \leq 1000$，各组数据中$n$的和不超过150000。

算法讨论

原问题也就是判断循环矩阵(*Circulant matrix*)$[A_0, A_1, \cdots, A_{n-1}]$是否满秩。若$gcd(f(x), x^n - 1) = d$，则该矩阵的秩为$n - d$，其中$f(x)$为

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

称$\phi_n(x) = \prod_{0 \le i < n, (i,n)=1} (x - w_n^i)$为分圆多项式，其中$w_n = e^{2\pi i/n}$。$\phi_i(x)$两两不可约，且$x^n - 1 = \prod_{d|n} \phi_d(x)$。因此，原问题可以转化为判断是否存在$d(d|n)$，满足$\phi_d(x)|f(x)$。

$$\phi_d(x)|f(x) \Leftrightarrow x^d - 1 | f(x) \prod_{p|d} (x^{d/p} - 1)$$

因为当$(a, p_1 p_2 \cdots p_k) = 1$时，有：

$$a|b \Leftrightarrow (a p_1 p_2 \cdots p_k) | (b p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k})$$

所以只需枚举d，判断$f(x) \prod_{p|d} (x^{d/p} - 1)$是否被$x^d - 1$整除即可。

时空复杂度

时间复杂度为$O(nd(n))$（其中$d(n)$为n的约数个数）。
空间复杂度为$O(n)$。

1.56 JAN14 CNTDSETS

试题名称

Counting D-sets.

题目大意

定义$N(N \le 1000)$维空间中，两个点$(a_1, a_2, \cdots, a_N), (b_1, b_2, \cdots, b_N)$的距离为$max\{|a_1 - b_1|, |a_2 - b_2|, \cdots, |a_N - b_N|\}$。

定义一个点集的直径，为点集中距离最大的两个点的距离值。

求直径恰好为$D(D \le 10^9)$的点集个数，对$10^9 + 7$取模。其中，若两个点集通过平移后是重叠的，则只算作一个点集。

算法讨论

因为点集是可以平移的，我们可认为每个点在每一维上的坐标都在范围$[0, D]$内，且对于每一维，至少存在一个点，该维的坐标为0。

问题要求的是直径恰好为D的点集个数。此处，直接计算比较困难。考虑将问题转变为计算直径不超过D的点集个数，减去直径不超过$D-1$的点集个数。当要计算直径不超过d的点集个数时，枚举强制不符合条件“至少存在一个点，该维的坐标为0” 的维度个数i，则可以被选的点数为$(d+1)^{N-i}d^i$，而选取不符合条件的维度方案数为$\binom{N}{i}$。 应用容斥原理，可知点集个数为：

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

时空复杂度

时间复杂度和空间复杂度均为$O(N^2)$。

1.57 JAN14 TAPAIR

试题名称

Counting The Important Pairs.

题目大意

给出一有$N(N \leq 100000)$个点，$M(M \leq 300000)$条边的连通无向图G，求选出两条边的方案数，满足删去这两条边后，图G不连通。

算法讨论

首先通过Dfs，可以求出原图的一棵生成树。考虑选择删去两条边后，原图依然连通的条件。

- 若选择的两条边都是非树边，显然原图连通。
- 若选择的边中有一条为非树边，另一条为树边，则删去树边后，生成树会被分裂成两个部分。这时需要一条非树边连接这两个部分，而这一条非树

边只需要满足在生成树上形成的环，包含删去的树边，且不是被删去的非树边即可。

- 若被删去的两条边皆为树边，生成树被分裂为三个部分，需要两条非树边连接这三个部分。对于两条被删去的树边，都需要被非树边包含，且包含它们的非树边集合不相同。

令S_i表示包含边i的非树边集合。对于一条树边而言，包含它的非树边，需要满足在生成树上形成的环包含这条树边。对于一条非树边而言，包含它的边就是它自身。

对于选出的两条边x, y，若删去这两条边后原图不连通，则S_x, S_y中至少有一者为空，或$S_x = S_y$。对于每一条非树边，可以在较大随机范围内随机一个整数作为其权值，于是可以用集合中非树边的权值异或和来表示一个非树边集合。当随机范围足够大时，出错概率可忽略。

因此，只需求出每条边所对应集合的权值，使用map或sort统计答案即可。

时空复杂度

时间复杂度为$O((N+M)logN)$。
空间复杂度为$O(NlogN)$。

1.58 FEB14 DAGCH

试题名称

Graph Challenge.

题目大意

给出一有$N(N \leq 100000)$个点，$M(N-1 \leq M \leq 200000)$条边的有向图，点已按照Dfs序编号，从点1 出发能够到达所有点。

称点x是点y的*supreme vertex*，当且仅当存在一条从点x到点y的路径，满足路径上除了点x和点y，其他点编号都比y大，且$x < y$。并称点y的所有*supreme vertexes*中编号最小的点为点y的*superior vertex*。求每个点是多少个其他点的*superior vertex*。

算法讨论

这里*superior vertex*的定义与Dominator Tree 中半必经点的定义完全相同。关于Dominator Tree 的介绍，可参阅GRAPHCNT 一题算法讨论。

将每个点连出的边按照编号从小到大排序，参考Dominator Tree的构建方法，可以求出每个点的半必经点$semi_i(1 \leq i \leq N)$，也就是题目中的*superior vertex*。最后用数组统计答案即可。

时空复杂度

时间复杂度为$O(N\alpha(N))$。
空间复杂度为$O(N+M)$。

1.59 FEB14 COT5

试题名称

Count on a Treap.

题目大意

要求维护一个Max-Treap，支持以下三种操作：

- 插入一个关键字为k，权值为w的点。
- 删除关键字为k的点。
- 求关键字为k_u和k_v的两个点之间的距离。

数据范围：操作数N不超过200000，所有点关键字与权值不同。

算法讨论

要求点x与$y(x < y)$的距离，可以通过分别求得这两个点的LCA，和三个点的深度来实现。两个点的LCA是关键字在区间$[x,y]$中的权值最大点。若点z是点x的祖先，则不存在关键字在区间$[x,z](or\ [z,x])$中的点，满足权值比z大。由此，可以将点的关键字离散后，映射到线段树上，并在线段树上实现这些操作。

时空复杂度

时间复杂度为$O(Nlog^2N)$。
空间复杂度为$O(N)$。

1.60 MAR14 GERALD07

试题名称

Chef and Graph Queries.

题目大意

给一个点数为N，边数为M的无向图。要求回答Q组询问。第i组询问为$L_i, R_i(1 \leq L_i \leq R_i \leq M)$，表示询问当仅保留第$L_i, L_{i+1}, \cdots, R_i$条边时，该图中有多少连通块。

$N, M, Q \leq 200000$。

算法讨论

先考虑假若满足$L_i = 1(1 \leq i \leq Q)$，那么就只需要维护一个并查集，按顺序加入每条边，若该边的两个端点不在同一个集合中，则合并这两个集合，连通块数量减少一。在加边的时候可以同时回答询问。

而当$L_i > 1$时，也就是最初加的$L_i - 1$条边是无效的。对于一条边(x_k, y_k)，找到另一条边$(x_j, y_j)(j < k)$，满足只加入第$j+1, j+2, \cdots, k-1$条边时，点x_k和点y_k是不连通的。那么对于询问$L_i, R_i(L_i > j, R_i = k)$，边$(x_k, y_k)$都会对答案产生$-1$的贡献。这里可以用树状数组维护答案。

问题只剩要如何寻找(x_j, y_j)了。按顺序加入每条边，用Link-Cut Tree维护当前已加入的边形成的生成森林。其中森林中的边带有权值，即这条边的编号。当加入一条边(x_k, y_k)时，若点x_k和y_k此时不连通，不存在边(x_j, y_j)，可以看做是$j = 0$，直接Link 点x_k与点y_k；否则，在Link-Cut Tree 上询问，找到点x_k和y_k路径上边权最小的边，这条边就是我们要找的边。将这条边在Link-Cut Tree 上删去，再Link 点x_k与点y_k。 继续处理下一条边即可。

时空复杂度

用树状数组维护答案的时间复杂度是$O(MlogN)$的，用Link-Cut Tree维护生

成森林的时间复杂度是$O(Mlog^2N)$的。综上，总时间复杂度为$O(Mlog^2N)$。

空间复杂度为$O(M)$。

1.61 MAR14 STREETTA

试题名称

The Steet.

题目大意

要求维护两个长度为$N(N \leq 10^9)$的序列$A_1, A_2, \cdots, A_N$和$B_1, B_2, \cdots, B_N$，支持$M(M \leq 3 * 10^5)$条操作，操作有三种：

- 给出l, r, k, b，对于所有的$l \leq i \leq r$，将A_i修改为A_i与$k(i-l)+b$的较大值。
- 给出l, r, k, b，对于所有的$l \leq i \leq r$，将B_i加上$k(i-l)+b$。
- 给出x，询问$A_x + B_x$的值。

算法讨论

序列$A_1, A_2, \cdots, A_N$与$B_1, B_2, \cdots, B_N$相互独立，不妨分别考虑。

序列B较于A更容易维护些。操作二中，对一个元素加上$k(i-1)+b$的操作，可以看成是加上$ki-k+b$。这样，所加上的内容就被分离成了两个部分。我们使用一棵线段树来维护序列，对于线段树上的每个节点所表示的区间$[l, r]$，记录对整个区间产生影响的$(\sum k, \sum b)$，其中，对于所有的$l \leq i \leq r$，$(\sum k, \sum b)$对B_i产生$(\sum k)i + (\sum b)$的影响。回答对位置x的询问时，只需要在线段树上从根走向节点$[x, x]$，统计所有产生的影响的和即可。

序列A的维护要稍复杂些。与序列B的维护同理，操作一中，$k(i-1)+b$可以看做$ki-k+b$。则这个操作相当于线段的覆盖。我们依然维护线段树。对于树上节点$[l, r]$，记录对整个区间产生影响的覆盖操作(k, b)。当有一条新的覆盖操作(k', b')加入，若线段$y = k'x + b'(l \leq x \leq r)$严格在线段$y = kx + b(l \leq x \leq r)$上方，则用新的线段替换原有的线段即可。若新线段严格在原线段下方，则当前操作无任何影响。否则，两条线段存在唯一的交点。取$m = (l + r)/2$。 线段树上节点$[l, r]$的两个儿子为$[l, m]$和$[m + 1, r]$。 两个儿子中，必然有一个儿子，在该区间中，两条线段中的其中一者完整覆盖另一者。于是，只需要递归处理没

有完整覆盖的儿子即可。回答对位置x的询问时，与维护序列B时类似，也只需要在线段树上从根走向节点$[x,x]$，用所经过的节点上(k_i,b_i)更新答案即可。

时空复杂度

时间复杂度和空间复杂度均为$O(MlogN)$。

1.62 APR14 GERALD08

试题名称

Chef and Tree Game.

题目大意

给出一棵有$N(N \leq 100000)$个点的树，树上的边可以为黑色或白色。两个人在树上玩游戏，轮流操作。第一个人每次可以删去一条黑边，第二个人每次可以删去一条白边。当一条边被删去后，与根不连通的边也被一并删去。不能操作者输。问分别当第一个人和第二个人先手时的胜者。

算法讨论

对于不平等博弈问题，可用Surreal Numbers[2]来表示一个状态。对于根节点u，以其为根的子树的状态，可由其若干儿子对其的贡献合并——也就是直接相加得到。对于儿子v，设其状态为x。若(u,v)之间的边为黑边，且令p为使得$x+p>1$的最小正整数，则v对u的贡献为$\frac{x+p}{2^p-1}$。若(u,v)之间的边为白边，且令p为使得$x-p<-1$的最小正整数，则v对u的贡献为$\frac{x-p}{2^p-1}$。状态的整数部分可直接存储，小数部分可用set存储，合并的时候采用启发式合并即可。设根的状态为$f(root)$，则：

- 若$f(root)>0$，则玩家一必胜。
- 若$f(root)<0$，则玩家二必胜。
- 若$f(root)=0$，则后手必胜。

[2]有关Surreal Numbers的详细介绍，可参阅方展鹏《浅谈如何解决不平等博弈问题》。

时空复杂度

时间复杂度为$O(Nlog^2N)$。
空间复杂度为$O(NlogN)$。

1.63 MAY14 ANUDTQ

试题名称

Dynamic Trees and Queries.

题目大意

给出一棵有$N(N \leq 10^5)$个节点的有根树，其中节点带有权值。要求执行$M(M \leq 10^5)$组操作。操作有以下4种类型：

- 生成一个以已存在的点x为父亲的新节点。
- 将以点x为根的子树中的所有节点的权值加上$value$。
- 删除以点x为根的子树中的所有节点。
- 询问以点x为根的子树中的所有节点的权值和。

操作是强制在线的。

算法讨论

修改和询问操作中仅涉及子树操作而没有路径上的操作，由此可以考虑维护树的带权括号序列。一个节点的左括号至右括号的部分对应着它的子树。对于操作一，可以看做插入一对左右括号；对于操作二，可以看做对括号序列的一段区间进行权值加；对于操作三，可以看做删去括号序列的一段区间；对于操作四，可以看做询问括号序列的一段区间的权值和。

因此，只需要使用数据结构维护树的括号序列，支持上述四种操作即可。比较常见且方便的，可以使用平衡树来维护。

时空复杂度

时间复杂度为$O(Mlog(N+M))$。
空间复杂度为$O(N+M)$。

1.64 MAY14 SEINC

试题名称

Sereja and Subsegment Increasings.

题目大意

给定两个长度为$N(N \leq 10^5)$的序列$A_1, A_2, \cdots, A_N$和$B_1, B_2, \cdots, B_N(0 \leq A_i, B_i \leq 3)$。现在可以对序列$A$执行若干次操作，每次操作需要选择两个参数$l, r$，表示对$A_l, A_{l+1}, \cdots, A_r$在模4意义下加上一。问要把序列$A$变为序列$B$，最少需要操作多少次。

算法讨论

首先可以求得$C_i = B_i - A_i(\mathrm{mod}4)$，再求出$D_i = C_i - C_{i-1}$。若操作不在模意义下，则答案就是$\sum_{i=1}^{N} max(D_i, 0)$。但在模意义下，问题有一些不同。观察如下的例子：

$$N = 5$$
$$C = \{3, 0, 3, 2, 3\}$$
$$D = \{3, -3, 3, -1, 1\}$$

若不在模意义下，答案为7。但事实上，我们可以使C_2多加1，并继承C_1所增加的三个单位，也就是说，使得D_2加上4，同时将D_3减去4。此时，$\sum_{i=1}^{N} max(D_i, 0) = 5$，优于非模意义下的答案。由此可以发现，在模4 意义下，可以对序列D做一些改变，每次可以选取两个参数l, r，使得D_l加上4，D_r减去4。

枚举$i(1 \leq i \leq N)$，考虑是否要将D_i减去4，若需要，则需要找到$j(j < i)$，将D_j加上4 后，使答案更优。显然，D_i需要为正数。

- 若$D_i = 1$，答案不会更优。
- 若$D_i = 2$，则需要找到一个j，满足$D_j = -3$。
- 若$D_i = 3$，则需尽量找到一个j，满足$D_j = -3$；若不存在，再看是否存在j，满足$D_j = -2$。

若能找到，作上述变换即可。在实现上，可以用队列存储$D_i = -2, -3$的所有位置i。

时空复杂度

时间复杂度为$O(N)$。
空间复杂度为$O(N)$。

1.65 JUN14 TWOCOMP

试题名称

Two Companies.

题目大意

给一棵具有$N(N \leq 10^5)$个节点的树，再给出$M_1(M_1 \leq 700)$条A型路径，$M_2(M_2 \leq 700)$条B型路径。路径带有权值。

要求选出一些路径，满足任意两条类型不同的路径没有公共点。最大化所选路径的权值和。

算法讨论

根据两种不同类型路径的相交关系，可以建立一带点权的无向图，图上的每一个点，代表一条路径。对于两条不同类型的路径，若它们有公共点，则在代表这两条路径的点间连边。于是，问题转变为，选择该无向图上的一些点，满足任意两点不相连，最大化所选点权和。这是经典的最大点权独立集问题。根据题意可知，这个无向图还是二分图。

我们知道，求二分图的最大点权独立集，可建立网络流最小割模型。具体建图方式如下：

- 从源点向代表A型路径的点连容量为该点点权的边。
- 从代表B型路径的点向汇点连容量为该点点权的边。
- 对于代表一条A型路径的点x和代表一条B型路径的点y，从x向y连容量为∞的边。这意味着，点x和点y中必然有一者不可选。

总点权和减去这个网络的最大流，就是最大的所选路径权值和。

现在问题还剩下如何判断两条路径a和b是否有公共点。为了方便，不妨假设路径a两个端点的LCA比b的深。如果路径a上存在一个点x，在路径b上，容

易发现，点x到路径a两个端点的LCA的路径上所有点都是在路径b上的。换言之，若路径a和b有公共点，则路径a两个端点的LCA一定是在路径b上的。把这个LCA点求出后，判断这个点是否在路径b上，就能容易地判断路径a和b是否有公共点了。

时空复杂度

预处理求LCA的倍增数组的时空复杂度均为$O(NlogN)$。

求最大点权独立集的时间复杂度为$O(f(M_1+M_2,M_1M_2))$，其中$f(n,m)$表示求具有n个点，m条边的带源、汇网络最大流的复杂度。

1.66 JUN14 SEAARC

试题名称

Sereja and Arcs.

题目大意

在一维数轴上有$N(N \leq 10^5)$个点，分别位于$(1,0),(2,0),\cdots,(N,0)$。 第i个点的颜色为A_i。对于所有的$i,j(i \neq j, A_i = A_j)$，点i和点j之间连有一条弧，颜色为$A_i(A_j)$。 求有多少对颜色不同的弧相交。

算法讨论

考虑到直接计算相交的异色弧对数比较困难，应用“补集转化”思想，将问题转化为计算总异色弧对数，减去相离的异色弧对数，再减去相互包含的异色弧对数。总异色弧对数和相离的异色弧对数都能容易地得到。现在来考虑如何计算相互包含的异色弧对数。设一对相互包含的异色弧，处于外围的弧的颜色为x，处于内部的弧的颜色为y。

设颜色x在点中出现的次数为c_x，同理，颜色y在点中出现的次数为c_y。

若$c_x,c_y \leq \sqrt{N}$，则两种颜色的弧总数为$O(N\sqrt{N})$。枚举所有在产生包含关系时，处在外围的弧，需要求得在这段弧中间，颜色满足在点中出现次数不超过$\sqrt{N}$的弧数量。我们可以通过枚举弧的右端点j，用树状数组维护sum_i，表示两个端点都在区间$[i,j]$中的弧数量。枚举以点j为右端点的弧，设其另一个端点为i，则对答案产生$-sum_{i+1}$的贡献。注意，在此过程中，颜色相同的弧对会被

多余计算。枚举被计算到的颜色，除去由同色弧产生的弧对数量即可（若该色点数为n，则可产生的同色弧对数为$\binom{n}{4}$）。

若$c_x, c_y > \sqrt{N}$，则(x, y)的总对数为$O(N)$。枚举所有的$(x, y)(x \neq y)$，再从小到大枚举颜色为y的点，维护当前枚举到点前颜色为x的点数的和，每次会对答案产生的负贡献，就是当前所维护的和，乘上当前所枚举点后颜色为x的点数。

若$c_x \leq \sqrt{N}, c_y > \sqrt{N}$或$c_x > \sqrt{N}, c_y \leq \sqrt{N}$，发现此时计算包含的异色弧对数非常困难。幸运的是，直接计算相交的弧数量比较方便，方法与上类似。那么，在计算“总异色弧对数时”，可以只计算两种颜色出现次数同时超过$\sqrt{N}$或同时不超过$\sqrt{N}$的异色弧对数。于是，在之后，对于本种情况，就可以正面计算了。

将几种情况对答案的贡献相加，即可得到最终答案。

时空复杂度

时间复杂度为$O(N\sqrt{N}logN)$。

空间复杂度为$O(N)$。

1.67 JUL14 GNUM

试题名称

Game of Numbers.

题目大意

给出两个长度为N的序列$A_1, A_2, \cdots, A_N$和$B_1, B_2, \cdots, B_N$。现在需要维护两个数对集合$S1, S2$。对于每次操作，可以选择两个数对(i, j)和(p, q)，满足(i, j)不在$S1$中，(p, q)不在$S2$中，且$A_i < B_j, A_p > B_q, gcd(A_i, B_j, A_p, B_q) > 1$，把$(i, j)$加入$S1$，$(p, q)$加入$S2$。问最多可以操作多少次。共有$T$组数据。

数据范围：$T \leq 10, N \leq 400, A_i, B_i \leq 10^9 (1 \leq i \leq N)$。

算法讨论

建立网络流模型。

从源点向代表数对$(i, j)(A_i < B_j)$的点连容量为1 的边，从代表数对$(p, q)(A_p > B_q)$的点向汇点连容量为1的边。对于一个至少是$A_1, A_2, \cdots, A_N, B_1, B_2, \cdots, B_N$中

其中一个数的因数的素数p_i，若$p_i|A_i, B_j, A_p, B_q$且$A_i < B_j, A_p > B_q$，从代表数对(i, j)的点向代表数对(p, q)的点连容量为1的边。对这个网络求最大流，即是所求的答案。

时空复杂度

时间复杂度为$O(f(N^2 + NlogA_i, N^2logA_i)$（$f(n, m)$表示求具有$n$个点，$m$条边的网络最大流的复杂度）。

空间复杂度为$O(N^2 + N^2logA_i)$。

1.68 JUL14 SEAEQ

试题名称

Sereja and Equality.

题目大意

称两个长度为$n(n \leq 500)$的数组是相似的，当且仅当将它们对应位上的数字在各自数组所出现的数字中的排名都是相同的。

定义$F(P_1, P_2)$为满足$P_1[l \ldots r]$与$P_2[l \ldots r]$相似，且$P_1[l \ldots r]$包含不超过$E(E \leq 10^6)$个逆序对的数对(l, r)的数目。

求当P_1, P_2取遍所有n个元素的排列时，$F(P_1, P_2)$的总和。

一个测试点中测试数据组数T不超过10000。

算法讨论

$F(P_1, P_2)$的总和满足线性性，可以考虑把所有(l, r)对答案的贡献分开计算，求和即可。

一个长度为i的区间，对答案的贡献为：

$$(N - i + 1) * count_{i,E} \binom{n}{i}^2 * ((N - i)!)^2$$

其中，$(N - i + 1)$长度为n的数组中，长度为i的区间个数；$count_{i,E}$为满足逆序对个数不超过E的i个元素的排列数；$\binom{n}{i}^2$为P_1, P_2两个数组选用数字的方案数；$((N - i)!)^2$为两个数组区间外的部分排列方案数。

考虑如何预处理$count_{i,E}$。设$dp_{i,j}$表示逆序对个数恰为j的i个元素的排列数。$dp_{i,j}$可用以下表达式递推求得：

$$dp_{i,j} = \sum_{k=0}^{i} dp_{i-1,j-k}$$

其中，k表示枚举前$i-1$个元素有多少个比第i个元素大，也就是新产生的逆序对数。

直接处理DP复杂度为$O(n^3)$。通过前缀和优化即可将预处理复杂度降至$O(n^2)$。将$dp_{i,j}$数组求前缀和，即可得到$count_{i,j}$。

回答询问时，枚举产生贡献的区间长度，将贡献累加即可。

时空复杂度

预处理时间复杂度为$O(n^3)$，回答询问复杂度为$O(Tn)$。总时间复杂度为$O(n^3+Tn)$。

空间复杂度为$O(n^3)$。

1.69 AUG14 SIGFIB

试题名称

Team Sigma and Fibonacci.

题目大意

给出两个整数$n, m(0 \le n \le 10^{18}, 1 \le M \le 10^5)$，求:

$$\Big(\sum_{x,y,z \ge 0, x+y+z=n} 6xyzFib_xFib_yFib_z \Big) \bmod m$$

其中Fib是斐波那契数列：$0, 1, 1, 2, 3, 5, \cdots$。

一个测试点中有T组数据，T较大，但保证所有数据的m之和不超过10^6。

算法讨论

根据生成函数的基础知识可知，斐波那契数列的生成函数为：

$$F(x)=\frac{x}{1-x-x^2}$$

并且数列$\{xFib_x\}$的生成函数为：

$$G(x)=xF'(x)=\frac{x(1+x^2)}{(1-x-x^2)^2}$$

$$G^3(x)=\frac{x^3(1+x^2)^3}{(1-x-x^2)^6)}$$

问题转变为求$6G^3(x)$的幂级数展开中n次项的系数对m取模后的结果。
令$\alpha=\frac{-1-\sqrt{5}}{2},\beta=\frac{-1+\sqrt{5}}{2}$。

$$\frac{1}{(1-x-x^2)^6}=\frac{1}{(x-\alpha)^6(x-\beta)^6}$$

我们还发现$\sqrt{5}Fib_x$近似于$2Fib_{x-1}+Fib_x$。通过进一步演算可以得到$\frac{1}{(1-x-x^2)^6}$的幂级数展开中n次项的系数。该表达式较长，这里不再详细列出。由此我们便可以得到$6G^3(x)$的幂级数展开中n次项的系数对m取模后的结果了。

最后还需要快速求出$Fib(x)(n+1\le x\le n+5)$。比较经典的做法有构造矩阵，然后再使用矩阵乘法快速幂。但此做法常数较大，不足以通过本题。我们需要寻找常数小一些的求法。

斐波那契数列还存在下述性质:

- 当x是偶数时：$Fib_x=Fib_{x/2}*(Fib_{x/2+1}+Fib_{x/2+1}Fib_{x/2})$。
- 当x是奇数时，$Fib_x=Fib^2_{x/2+1}+Fib^2_{x/2}$。

利用该性质，可以递归快速求出$Fib(x)(n+1\le x\le n+5)$。复杂度为$O(logx)$，和矩阵乘法做法相同，但常数大大降低。

时空复杂度

时间复杂度主要产生在求$Fib(x)(n+1\le x\le n+5)$上。总时间复杂度为$O(Tlogn)$。

空间复杂度为$O(1)$。

1.70 AUG14 PUSHFLOW

试题名称

Push the Flow!

题目大意

给出一棵有$N(N \leq 10^5)$个点的仙人掌，仙人掌的的边有容量。要求支持两种操作：

- 修改一条边的容量。
- 询问点u和点v之间的最大流。

询问数量不超过10^5。

算法讨论

仙人掌由一些简单环组成。若把每个简单环缩成一个点，则仙人掌可以被看做是一棵树。于是，询问操作可以看做是关于树上路径的询问。考虑将缩环后得到的树进行轻重链剖分。将树上每个点所代表的原环上的点进行编号，和父亲环相连的点编为0 号点。为每个环设立权值，为重儿子环中的0号点到该环的0号点的最大流。我们用一棵线段树来维护重链上的环的区间权值最小值。在求每个环的权值时，需要询问从0 号点到任意点的最大流。而当询问操作时，若所询问的两点同属一环，则需要询问一个环上任意两点间的最大流。我们可以将环上的边在序列上展开，对每个环建立单独的线段树，维护一个区间内的边容量最小值。

我们再来考虑询问操作。当u和v同属一环时，问题较为简单，只需要求环上任意两点间的最大流，可以转化为线段树上的区间询问。当u和v分属两环时，求得两环的LCA为w，从两点所在环分别向环w方向跳，对于一段连续的重链，可以直接在线段树上询问区间权值最小值。而对于跳过轻边的情况，可以每次在所跳过的环中询问入口和出口间的最大流，可能经过的轻边的数量很少，这样询问的次数不会超过$O(logN)$。 对于w是u和v所在环其中之一的情况，作特殊处理即可。

时空复杂度

时间复杂度为$O(Nlog^2N)$。
空间复杂度为$O(NlogN)$。

1.71 SEP14 QRECT

试题名称

Rectangle Query.

题目大意

在一个二维平面上，支持三种操作：

- 加入一个矩形。
- 删除此前加入的某一个矩形。
- 给一个矩形，询问平面上有多少个矩形和该矩形至少有一个公共点。

操作共有$Q(Q \leq 10^5)$条。

算法讨论

对于一个加入操作，对答案贡献为正；对于一个删除操作，对答案贡献为负。加入操作和删除操作都可以看做加入一个带符号的矩形。

观察询问，由补集转化思想，考虑将原问题转化为求有多少个矩形和该矩形没有公共点。严格位于该矩形上方、下方、左方、右方的矩形和该矩形显然是没有公共点的。但是严格位于该矩形左上方、左下方、右上方、右下方的矩形会被统计两次，需要减去重复统计的部分。

使用树状数组，可以维护并求得严格位于一个矩形上方、下方、左方、右方的矩形个数。

采用分治算法并用树状数组维护答案，可以求得严格位于每个矩形左上角、左下角、右上角、右下角，且在询问时还未被删去的矩形个数。

时空复杂度

时间复杂度为$O(Qlog^2Q)$。
空间复杂度为$O(Q)$。

1.72 SEP14 FIBTREE

试题名称

Fibonacci Numbers on Tree.

题目大意

斐波那契数列满足$fib_1 = 1, fib_2 = 1, fib_n = fib_{n-1} + fib_{n-2}(n > 2)$.

给一棵有$N(N \leq 10^5)$个节点的树，点i的权值为w_i。初始时，所有点权值为0。要求处理$M(M \leq 10^5)$条操作，操作有以下四种：

- 将点x到点y路径上的第k个点权值加上fib_k。
- 询问当以点x为树根时，以点y为根的子树中所有点的权值和。
- 询问点x到点y路径上的所有点权值和。
- 将所有点的权值还原到第x个询问后的状态。

操作强制在线。

算法讨论

先来考虑如何处理在序列上的操作。首先，可以在$O(N)$的时间内预处理出$fib_i = a_i fib_1 + b_i fib_2, \sum_{k \leq i} = c_i fib_1 + d_i fib_2 (1 \leq i \leq N)$。我们可以使用一棵线段树来维护这个序列。对于线段树上的每个节点，维护一个标记(a, b)，表示对这个区间叠加上一个以a为首项，b为次项的斐波那契数列。利用之前已经预处理出的两个序列，这个标记可以方便地下传，并能统计出对一个区间带来的实际影响，也就是所增加的权值和。由于题目中还要求对数据结构可持久化，因此可以维护可持久化线段树。但维护带标记下传的可持久化线段树消耗内存很大。实际上，标记可以不必下传，而使用“标记永久化”的方式，这一点与之前的FEB13 QUERY一题是类似的。

再来考虑在树上如何进行操作。将树进行轻重链剖分，在修改操作的时候，对于一条重链的某一个部分，需要正向或逆向叠加一个斐波那契数列。因此，为了方便操作，可以建立两棵线段树，分别用于处理正向和逆向的修改操作。路径点权和询问较为简单，为经典的树链剖分询问操作。而关于根可变的子树权值和询问，也同样可以用归结为树链剖分上的操作。注意到树链剖分后，

以原树的根（通常取1 号点）为整棵树的根时，以某个点为根的子树中的点的DFS序是连续的，可以看做序列上的一段连续区间。设点i的Dfs序为dfn_i，子树大小为$size_i$。尝试关于询问中的假设总树根x与子树根y的位置关系进行分类讨论：

- 若$x = y$，则询问的就是整棵树，Dfs 序区间为$[1, N]$。
- 若点y在原树中不是点x的祖先，则询问的子树就是在原树中，以点y为根的子树，Dfs序区间为$[dfn_y, dfn_y + size_y - 1]$。
- 若点y在原树中是点x的祖先，且$x \neq y$，则设点z为点y到点x最短路径上第二个经过的点，则询问的子树就是原树中，以点y为根的子树除去以点z为根的子树，Dfs序区间为$[dfn_y, dfn_z - 1] \bigcup [dfn_z + size_z, dfn_y + size_y - 1]$。

至此，树上的操作就可以归结为序列上的操作了。

时空复杂度

时间复杂度和空间复杂度均为$O(Nlog^2N)$。

1.73 OCT14 TRIPS

试题名称

Children Trips.

题目大意

给一棵有N个点的树，树上边的长度只可能为1或2。有Q名学生要在树上游走。第i名学生要从点x_i走到点y_i。 他的体力值为p_i，这表示每天他最多可以走的距离为p_i。 当一天结束后，学生必须要处在一个节点上而不能是在一条边上。对于每名学生，回答完成游走的最少天数。

$N, Q \leq 100000$。

时限为8秒。

算法讨论

本题时限较大，这意味着可以接受复杂度稍高的算法。

为了尽快完成游走，学生的策略将会是每天走到所能达到的最远点。直接模拟这个过程，复杂度为$O(QN)$。

尝试优化这个算法。注意到边的长度只有两种取值。对于点i，将这个点向上w条边的权值用一个二进制数来表示，设为$mask_i$。 于是在模拟的时候每步可以不止走过一条边，设一次最多走过w条边。对于一个学生，若$p_i > 2w$，则这个学生可以一次性将这w条边走完，否则，我们事先预处理出$f_{state,p,rem}$，表示当前所在点向上w条边的权值的二进制表示为$state$，当前这个学生的体力为p，这一天还剩的体力为rem，要再走w条边，需要多少天。同时求出$g_{state,p,rem}$表示走完w条边后，所在天剩下的体力。

回答第i名学生的询问的时候，从x_i和y_i分别往两个点的最近公共祖先(LCA)方向跳，每次跳w条边，同时维护答案。在LCA附近，可能一次跳跃不足w条边，且两边需要合并，进行简单的细节处理即可。

综合算法常数因素的考虑，取$w = 9$左右时，可以通过本题。

时空复杂度

预处理的时间复杂度为$O(w^3 2^w + NlogN)$，回答询问时间复杂度为$O(\frac{QN}{w} + QlogN)$。综上，总时间复杂度为$O(w^3 2^w + NlogN + \frac{QN}{v} + QlogN)$。

空间复杂度为$O(w^2 2^w)$。

1.74 OCT14 BTREE

试题名称

Union on Tree.

题目大意

给出一棵具有N个点的树，有Q个询问，对于每组询问，给出k_i组(v_j, r_j)，表示点v_j的辐射范围为r_j，求被辐射到的点个数。

数据范围：$N, Q \leq 50000$，$K = \sum_{i=1}^{Q} k_i \leq 500000$。

算法讨论

先来考虑$k_i = 1$的情况，也就是要多次求到点v距离不超过r的点个数，可以使用离线点分治解决。

考虑$k_i \geq 1$的一般情况。对于每组询问，可以对给出的点建立虚树。使用Dfs，可以将每个点的辐射范围更新为，其他所有点辐射到这个点，还剩余的辐射量最大值。对于虚树上的点u，可求出点u的虚子树中到点u距离不超过r_u的点个数。之后考虑虚树上的边(x, y)之间点的虚子树对答案的贡献。

我们对边(x, y)找到一个中点z，满足z到y路径上的点接受产自y的辐射量更多，而x到z父亲路径上的点接受产自x的辐射量更多。对于x，要求出子树x除去子树z部分中，到点x距离不超过r_x的点个数。对于y，要求出子树z除去子树y部分中，到点y距离不超过r_y的点个数，这可以转化为，求任意点到点y距离不超过r_y的点个数，减去子树y中到y距离不超过r_y的点个数，减去子树z中到z距离不超过$r_y - dist(y, z)$的点个数，加上子树z中到z距离不超过$r_y - dist(y, z)$的点个数。而要求一个点子树中到该点距离不超过r的点个数，使用可持久化线段树即可。

时空复杂度

时间复杂度为$O((N + K)logN)$。

空间复杂度为$O(NlogN)$。

1.75 NOV14 FNCS

试题名称

Chef and Churu.

题目大意

给出一个长度为$N(N \leq 10^5)$的序列$A_1, A_2, \cdots, A_N$和N个函数。第i个函数为L_i, R_i，表示询问$\sum_{k=L_i}^{R_i} A_k$。

需要支持$Q(Q \leq 10^5)$次操作，每次操作给出两个参数x, y。操作有两种：

- 将A_x修改为y。
- 询问$\sum_{i=x}^{y} \sum_{k=L_i}^{R_i} A_k$。

算法讨论

先考虑一个复杂度较高的算法。用一个数组记录序列中每个元素的值。对于第一种操作，直接在数组中修改值即可，复杂度$O(1)$。 对于第二种操作，需要计算每个函数的值，再累加。通过使用前缀和优化，可以做到$O(N)$回答询问。总复杂度$O(QN)$，显然无法承受。

我们可以注意到，两种操作的复杂度差距悬殊。让我们试着平衡两种操作的复杂度，也就是适当提升第一种操作的复杂度，从而来降低第二种操作的复杂度。

我们引入一个参数W。对于之前所说的算法，在询问的时候，每次都要重新计算每个函数的值，十分浪费。考虑每经过W次操作，重新计算所有函数的值，而对于询问，没必要去重新计算每个函数的值，只是考虑上一次计算所有函数的值相加的结果，同目标答案还差多少。容易发现，上一次操作后，这一次操作前的修改操作产生的影响，还没有被统计到答案中。而这些操作的数量不超过W。

枚举这些修改操作，设这个操作为将A_x修改为y。我们预先保存下A_x修改后的值较修改前的值的改变量，设为d。 设当前的询问为(l,r)，那么改变量d对于我们所期望的答案，产生贡献的次数，也就是函数$l,l+1,\cdots,r$各自的区间，对修改操作作用的位置x的覆盖次数。设覆盖次数为c，则对答案的贡献就是$d\times c$。

再来考虑怎么计算函数$l,l+1,\cdots,r$各自的区间对某个位置的覆盖次数。若维护线段树套线段树，虽然的确可以计算这个覆盖次数，但每次询问的复杂度为$O(log^2N)$，显得偏大。注意到，这些函数的覆盖区间，是在初始时就已经确定的，在操作进行的过程中，它们不会发生改变。因此，我们可以预先关于函数下标，建立函数式线段树，Seg_i表示由前i个函数产生的覆盖作用所建立的线段树。在函数式线段树上单点修改，区间询问，即可做到计算一个区间内的函数对某个位置的覆盖次数。每次询问复杂度为$O(logN)$。

时空复杂度

总时间复杂度为$O(\frac{QN}{W}+QWlogN)$。使用基本不等式，即可得当$W=\sqrt{N/logN}$时，复杂度最低，为$O(Q\sqrt{NlogN})$。

空间复杂度为$O(NlogN)$。

1.76 NOV14 SEAORD

试题名称

Sereja and Order.

题目大意

有$N(N \leq 200000)$个零件需要通过两道工序加工，工厂内有两台机器，分别可以用于两道工序的加工。零件i在两道工序加工上需要花费的时间分别为A_i和B_i。同一时刻，一个零件不能在两台机器上被加工；一台机器也不能加工两个零件。求最少需要花费的时间及方案。

算法讨论

首先，我们来考虑时间t的下界。两台机器加工零件的总有效时间是一定的，分别为$\sum_i A_i$和$\sum_i B_i$。 那么，显然地，$t \geq max(\sum_i A_i, \sum_i B_i)$。对于零件$i$而言，被加工的总有效时间也是一定的，为$A_i + B_i$。 于是，$t \geq \max_i(A_i + B_i)$。因此，$t \geq max(\sum_i A_i, \sum_i B_i, \max_i(A_i + B_i))$。 这个下界似乎总是可以取到的，如下说明：

我们不妨假设$\sum_i A_i \geq \sum_i B_i$。

若$max_i(A_i + B_i) \geq \sum_i A_i \geq \sum_i B_i$，则设当$i = x$时$A_i + B_i$取到最大值。此时$\sum_{i \neq x} A_i \leq B_x$，$\sum_{i \neq y} B_i \leq A_x$。于是，对于机器A，可以先加工零件$x$，再加工剩下的所有零件。对于机器B，先加工除零件$x$外的所有零件，再等待至零件$x$的第一道工序加工完毕后，加工零件$x$的第二道工序。这种情况下需要花费的总时间为$A_x + B_x$。

若$\sum_i A_i > max_i(A_i + B_i)$，考虑一个贪心算法。我们先将零件按照加工第二道工序需要花费的时间升序排序。先假设每个零件加工第二道工序紧接着第一道工序的加工进行，求得在时间$\sum_i A_i$内，最多可以加工p个零件。然后，将加工这些零件第二道工序的时间推移到时限的末尾，并在开始时，先按顺序加工其他零件的第二道工序。这样操作，被推移的前p个零件显然是没有冲突的，关键在于除这p个零件之外的零件在加工时是否会有冲突。研究发现，是否有冲突，与零件被考虑的顺序有关。按加工第二道工序需要花费的时间升序排序，未必是合理的。当一个方案生成后，检查这个方案在时间上是否有冲突，若有冲突，则需要调换零件的顺序。具体实现上，可以设计多个排序的比较函数，或多次对所有零件随机打乱顺序(random_shuffle)。 此做法能够通过原题数据。

时空复杂度

时间复杂度为$O(N\alpha)$（其中$O(\alpha)$为调整顺序的时间复杂度）。
空间复杂度为$O(N)$。

1.77 DEC14 DIVIDEN

试题名称

Divide or die.

题目大意

尝试使用尺规作图，将$N(0 < N < 360)$度角N等分。

算法讨论

将N度角N等分后，每一部分都是1度，那么不妨先考虑能否由一个N度角作出1度角。我们尝试从一些特殊角三角形中，通过组合获得我们需要的角度。

黄金三角形，是三个角分别是$108°, 36°, 36°$的三角形。我们可以从这个三角形中获取$36°$。

等边三角形，三个角都是$60°$。我们可以从中获得$60°$，将这个角平分后，便可得到$30°$。

$$36° - 30° = 6°$$

将我们得到的$6°$角平分，便可以得到$3°$。若$N \bmod 3 \neq 0$，我们可以通过多次从N度角中减去三度角，得到一个一度角或二度角。二度角通过平分又可得到一度角。而当$N \bmod 3 = 0$时，假如N度角可被N等分，则N度角也可被三等分。我们知道，尺规作图是无法对角进行三等分的。于是，我们可以得出，N度角可以被N等分，当且仅当$N \bmod 3 \neq 0$。

最后再考虑如何用尺规作图构造黄金三角形。黄金三角形三边长比为$1 : 1 : \frac{\sqrt{5}+1}{2}$。 我们通过构造一个直角边长分别为1 和2 的直角三角形，便可以得到$\sqrt{5}$，进一步可得$\frac{\sqrt{5}+1}{2}$。 用$1, 1, \frac{\sqrt{5}+1}{2}$三边构造三角形，即可得到黄金三角形。

时空复杂度

时间复杂度为$O(N)$。

空间复杂度为$O(1)$。

1.78 DEC14 RIN

试题名称

Course Selection.

题目大意

课业计划共包含$N(N \leq 100)$项课程，每项课程都需要在$M(M \leq 100)$个学期中的某一个完成。

给出$K(K \leq 100)$对课程的前置关系。对于第i对课程(A_i, B_i)，课程A_i是课程B_i的前置课程。

$W_{i,j}(1 \leq i \leq N, 1 \leq j \leq M)$表示课程$i$在第$j$学期完成能得到的期望分数。若$W_{i,j} = -1$，则表示课程$i$不在第$j$学期开设。

安排课业计划，最大化所能得到的期望分数之和。

算法讨论

根据题目中的条件，可以建立网络流最小割模型。

在网络中建立$N(M+1)+2$个点，分别为点S（源点），点T（汇点），点$(i,j)(1 \leq i \leq N, 0 \leq j \leq M)$。

由点S向点$(i,0)(1 \leq i \leq N)$连容量为∞的边，由点$(i,M)(1 \leq i \leq N)$向点T也连容量为∞的边。由点$(i,j)(1 \leq i \leq N, 0 \leq j < M)$向点$(i,j+1)$连容量为$-W_{i,j+1} + maxW$的边。其中$maxW = \max_{i,j}\{W_{i,j}\}$。

对于一组前置关系(A_i, B_i)，由点$(A_i,k)(0 \leq k < M)$向点$(B_i,k+1)$连容量为∞的边。

对该网络求出最大流$flow$，则答案为$maxW \times N - flow$。

这样的建图方式，把求最大收益转化为求最小损失，并且确保每一项课程都恰好被学习一次，同时满足所有前置关系。

时空复杂度

时间复杂度为$O(f(NM, (N+K)M))$，其中$f(n,m)$表示求有n个点，m条边的带源、汇网络最大流的时间复杂度。

空间复杂度$O((N+K)M)$。

1.79 JAN15 RANKA

试题名称

Ranka.

题目大意

考虑一个在9×9棋盘上进行的围棋局，两名玩家交替行棋。在每一步中，玩家必须在一个空点放置一个棋子或者放弃本轮。如果玩家放置了棋子，则可能发生有一个或多个对手棋子的连通块呈“无气”状态（即不与任何空点相邻）的情况，此时可将对手呈无气状态的棋子提出盘外，称为”提子“；否则，新放置的棋子不能导致己方的棋子呈无气状态。

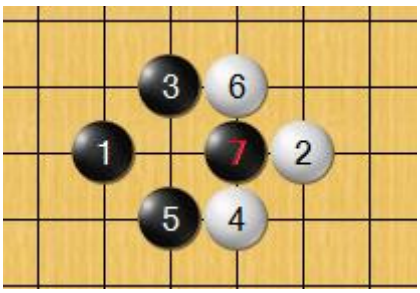
为了避免死循环，棋局需要遵循“禁止全局同形再现”原则。

给出$N(N \leq 10000)$，要求构造含N个合法行动的棋局。

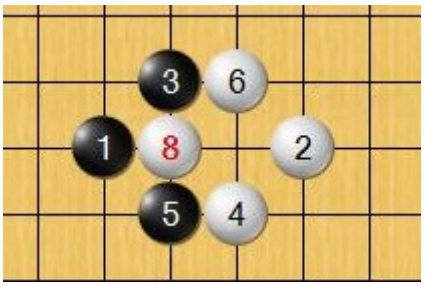
算法讨论

题目要求在一不大的棋盘上构造出最多含10000个行动的棋局。这提示我们，需要合理利用“提子”这一规则，循环使用有限的位置来构造方案。

我们尝试构造如下图所示的结构，不妨称之为“眼”：



可以认为，这个眼目前是由白方控制的，因为白方下一步只要将新棋子下在棋子1 和棋子7之间，就可以提去黑方的棋子7。 这样操作后，眼的控制者也发生了改变，变为黑方。如图：



这样的“眼”，还可以看成是一个开关，它的状态可以自由改变。但是，由于“禁止全局同形再现”原则，一个眼只能变换出两种不同的局面。

我们尝试使用多个眼，来变换出更多的局面。事实上，这些眼的组合所形成的状态，可以看成是一个二进制数。假如我们一共放置k个眼，通过变换眼的控制方，也就是变换开关的状态，就可以生成2^k种不同的局面。考虑在棋盘的四个角落和四条边各放置一个眼。算上构造形成眼的过程中产生的不同局面，再通过变换眼的控制方，一共能产生$8 \times 7 + 256$种不同的局面。但这与要求的构造最多含10000个合法行动的棋局的要求还是有差距。

注意到，我们构造眼，只使用了棋盘边缘的部分，而棋盘中央的部分，并没有被使用到。于是，我们可以在棋盘中央空闲的位置上放置棋子，每放置一个棋子，边缘的8个眼就变换256 次。如此的构造算法，就已经可以生成含13500余个合法行动的棋局了。

时空复杂度

时空复杂度均为$O(N)$。

1.80 JAN15 XRQRS

试题名称

Xor Queries.

题目大意

维护一个初始时为空的整数序列A，支持以下操作：

- 在序列末位增加一个整数$x(x \leq 5 \times 10^5)$。
- 在$A_L, A_{L+1}, \cdots, A_R$中找一个元素$y$，最大化$x\ xor\ y$的值。

- 在$A_L, A_{L+1}, \cdots, A_R$中，统计不大于$x$的元素个数。
- 在$A_L, A_{L+1}, \cdots, A_R$中，寻找第$k$小的元素。
- 删除序列末尾的k个元素。

操作总数M满足$M \leq 5 \times 10^5$。

算法讨论

先观察操作二。$x\ xor\ y$涉及到位运算。对于序列中的每个元素，尝试考虑其二进制的形式。根据它们的二进制表示，可以建立一棵边权为0或1 的trie树。从树根到每个关键点的路径，对应着序列中一个元素的二进制表示。假如这个序列不要求动态维护，而是在初始时就全部给定，那么在询问时，就只需要在trie 上从根向更深处行走。设当前走到的点为u。 若u是叶节点，则答案就是关键点u所代表的元素；若u只有一个儿子，那么直接走向这个儿子；若u有两个儿子，走向它们的边上的权值分别为0 和1，设u的深度为dep(1-based)，x的二进制表示中的第dep(1-based) 位为b，那么可以贪心地选择一条权值为a的边，使得$a\ xor\ b = 1$，沿这条边走向下一点即可。对于另外两种询问操作，都可以通过类似的在trie 上行走的方式来实现。

在原题中序列要求支持在末尾加入和删除元素的动态维护。trie树是可持久化的数据结构。于是只需在维护静态trie 树的基础上维护可持久化trie树即可。

时空复杂度

增加元素操作时间复杂度为$O(1)$，删除元素操作时间复杂度均摊为$O(1)$。询问操作复杂度为序列中元素的二进制表示长度，即$O(logx)$。 因此，总时间复杂度为$O(nlogx)$。

空间复杂度也为$O(nlogx)$。

1.81 FEB15 DEVLOCK

试题名称

Devu and Locks.

题目大意

求$f_i(0 \leq i \leq M)$表示十进制数x的个数，满足其各位数字和为i，且恰好被P整除，允许有前导零。

数据范围：$N \leq 10^9$，$M \leq 15000, P \leq 16$或$M \leq 500, P \leq 50$。

算法讨论

原问题要求的$f_i(0 \leq i \leq M)$满足：

$$f_i = [x^i] \prod_{j=0}^{P-1} (\sum_{k=0}^{9} x^k y^{jk})^{c_j}$$

其中，c_i表示整数$j(0 \leq j < N)$的个数，满足$10^j \equiv i (\mathrm{mod} P)$，并且，$y$的次数在模$P$意义下。

利用10^i存在循环节的性质，可以容易地在$O(P)$的复杂度内求出$c_i(0 \leq i < P)$。然后，枚举j，需要求得：

$$G(j) = (\sum_{k=0}^{9} x^k y^{jk})^{c_j}$$

提取y^{jk}中的y^j，得：

$$G(j) = y^{jc_j} (\sum_{k=0}^{9} x^k y^k)^{c_j}$$

此时，把xy看做同一个变量，通过FFT和快速幂，可以求得$(\sum_{k=0}^{9} x^k y^k)^{c_j}$。再枚举每一项中$y$的次数，以及在之前所求得的$\prod_{j'=0}^{j-1} (\sum_{k=0}^{9} x^k y^{j'k})^{c'_j}$中$y$的次数，用$FFT$再做卷积，即可得到$\prod_{j'=0}^{j} (\sum_{k=0}^{9} x^k y^{j'k})^{c'_j}$。

时空复杂度

时间复杂度为$O(PMlogNlogM + P^2MlogM + P^3M)$。

空间复杂度为$O(PM)$。

1.82 MAR15 TREECNT2

试题名称

Counting on a Tree.

题目大意

给出一棵有N个点的树，树上边带有权值w_i。要求询问无序点对(x,y)的数量，满足点x到点y路径上经过的边的最大公约数为1。 同时，还需要支持Q条边权修改操作，并在每次修改操作后，输出新的答案。

数据范围：$N \leq 10^5, Q \leq 100, w_i \leq 10^6$。

算法讨论

$$gcd(x,y) \Leftrightarrow \sum_{d|x,y} \mu(d) = 1$$

令f_i为仅保留权值为i的倍数的边时，连通的点对数，则答案为：

$$\sum_i \mu(i) f_i$$

在没有修改的情况下，这个答案可通过使用并查集维护连通块大小容易地求得。而在有修改的情况下，可以采用按时间分治，在每个分治区间内，将权值不会在该区间内被改变的边连接，剩余的边传递到下一层进行考虑，并用栈记录并查集被修改的信息，在每次分治结束后恢复并查集至当前分治前的状态即可。

时空复杂度

时间复杂度为：$O(Nlogw_i logQlog\alpha)$。

空间复杂度为：$O(Nlog\alpha + Nlogw_i)$。

1.83 APR15 BWGAME

试题名称

Black-white Board Game.

题目大意

给出一个$N \times N$的矩阵A，其中，第i行的$L_i, L_{i+1}, \cdots, R_i$列元素为1，其余元素均为0。 称一个排列$p_1, p_2, \cdots, p_N$是合法的，当且仅当对于$1 \leq i \leq N$，$A_{i,p_i} = 1$。求合法排列中，逆序对数为偶数的排列还是为奇数的排列比较多。

数据范围：$T \leq 15$，$N \leq 10^5$。

算法讨论

容易发现：

- 若$\det A > 0$，则逆序对数为偶数的合法排列比较多。
- 若$\det A < 0$，则逆序对数为奇数的合法排列比较多。
- 若$\det A > 0$，则两种排列同样多。

于是问题转为求$\det A$。矩阵A存在特殊的性质，也就是每一行为1的元素都是连续的一段，我们希望在消元求行列式的过程中，保留这种性质。先来考虑第一列，找到A的第i行，满足$L_i = 1$，且R_i最小。若不存在这样的i或多个i，显然$\det A = 0$。对于其他的行j，若$L_j = 1$，则通过与第i行进行消元，可将其转变为$L_j = R_i + 1$。这样的操作结束后，新矩阵的第一列，只有第i行的元素为1。 考虑对该矩阵按第i行展开，只有当删去的列为第一列时，子式的行列式可能不为0，于是我们成功将问题规模缩小到$N - 1$。而在这个过程中，可以使用C++中的set维护行信息，消元的时候进行启发式合并即可。

时空复杂度

时间复杂度为$O(TNlog^2N)$。

空间复杂度为$O(N)$。

1.84 APR15 LPARTY

试题名称

Little Party.

题目大意

给出一个由M个元素组成的集合，每个元素由N个布尔变量组成。要求找到一个总长度最小的基集合，使得以这个基集合中的元素为子集的元素的集合，恰好为给出的集合。

数据范围：$T \leq 120$，$0 \leq M \leq 1000$，$N \leq 5$。

算法讨论

基集合中的元素可在3^N个不同元素中选择，但其中有很多元素是不合法或是多余的。对于待确认的元素x，若存在一个长度为N的元素y，满足x是y的子集，而y又不属于给出的M个元素的集合，则元素x是不合法的。再来考虑哪些元素是多余的。举个例子说，当前考虑的元素是Ab，而AbC和Abc都属于初始给出的集合，那么实际上AbC和Abc都是多余的，因为可以用Ab来覆盖。消除所有不合法或多余的待选元素后，有效的元素个数不超过2^N。

接下来需要选取这些有效元素的一个集合，使得能够覆盖给出的初始集合。Dfs，再加上简单的可行性与最优性剪枝即可。

时空复杂度

时间复杂度为$O(2^{2^N}\alpha T)$。

空间复杂度为$O(4^N)$。

1.85 MAY15 CBAL

试题名称

Chef and Balanced Strings.

题目大意

称一个字符串是平衡的，当且仅当它的字符可以被分成两个相等的多重集。

现给出一个字符串$T(|T| \leq 10^5)$，有$Q(Q \leq 10^5)$组询问，第i组询问为$L_i, R_i, type_i (1 \leq L_i \leq R_i \leq |T|, 0 \leq type_i \leq 2)$，表示询问：

$$\sum_{\substack{L \leq s \leq e \leq R \\ T_{s,e} \text{ is balanced}}} |T_{s,e}|^{type}$$

询问是强制在线的。

算法讨论

令b_i为每种字符在字符串T的前i位的出现情况，用一个二进制数来表示。$T_{l,r}$是平衡的，当且仅当$b_{l-1} = b_r$。

考虑将字符串分成$\sqrt{|T|}$块，每块大小为$|T|/\sqrt{|T|}$。对于询问L_i, R_i，设L_i所在块为l，R_i所在块为r，则该询问的答案可以表示成：(s, e分别在块`l,r`的答案)+(s, e都不在块`l,r`的答案)+(s在块`l`，e不在块`r` 的答案)+(s不在块`l`，e在块`r`的答案)。

对于答案的第一部分，可以先在数组中计数pre_x(x+1属于块l且在区间$[L_i, R_i]$内)。然后对于pre_y(y属于块r且在区间$[L_i, R_i]$内)统计存在多少x满足$pre_x = pre_y$。

对于答案的第二部分，可以先预处理出$sum_{l,r}$，代表块$[l, r]$中的答案，直接回答询问即可。

对于答案的第三部分，可以预处理出$sumRight_{l,r}$，代表e在块r中，$s \geq l$产生的答案。那么这部分的答案就是：

$$\sum_{l \leq k < r} sumRight_{L_i,k}$$

答案的第四部分求法和第三部分求法相似。

将各部分答案相加，即是该询问的答案。

时空复杂度

时间复杂度为$O((|T| + Q)\sqrt{|T|})$。

空间复杂度为$O(T\sqrt{|T|})$。

1.86 MAY15 GRAPHCNT

试题名称

Counting on a directed graph.

题目大意

给出一有$N(N \leq 10^5)$个点，$M(M \leq 5 * 10^5)$条边的有向图，求无序点对(x, y)的数量，满足分别存在从点1 到点x，从点1 到点y的两条路径，这两条路径除点1外无公共点。

算法讨论

本题是*Dominator Tree*的一个应用，下面将首先介绍这一工具，若你已对此已有所了解，可跳过此部分[3]。

定义 1.86.1. 称有向图G为*Flow Graph*，当且仅当存在一点r，从r出发可以到达G中所有点。若G不为*Flow Graph*，可通过去掉从r出发不能到达的点，使之成为*Flow Graph*。

定义 1.86.2. 设有向图G的一棵搜索树为T，则边(x, y)可能是：

- 前枝边：在搜索树中。
- 前向边：搜索树中存在一条从x到y的路径，即x是y的祖先。
- 后向边（返祖边）：搜索树中存在一条从y到x的路径。
- 横叉边：搜索树中，既没有从x到y的路径，也没有从y到x的路径，且$dfn_y < dfn_x$。

定义 1.86.3. 称点x为点y的必经点，当且仅当从r出发到y的所有路径都经过点x。一个点的最近必经点$idom(x)$，为点x的所有必经点中，dfn值最大的一个。

定义 1.86.4. 点x的半必经点$semi(x)$为搜索树上，点x的祖先中，能不经过其他搜索树上在点r到x路径上的点，到达点x的深度最小的点。

[3] *Dominator Tree*相关内容参考自李煜东《图连通性若干拓展问题探讨》。

定理 1.86.1. 半必经点定理：设点集S，考虑每一条连向点y的边(x,y)：

- 若$dfn_x < dfn_y$，则将点x加入S。
- 若$dfn_x > dfn_y$，则考虑x在搜索树上的祖先z，若$dfn_z > dfn_y$，将$semi(z)$加入S。

从S中选出Dfs序最小的点，就是y的半必经点$semi(y)$。

定理 1.86.2. 必经点定理：设搜索树上$semi(x)$到x的路径上除$semi(x)$以外的点，选出其中半必经点深度最小的点y：

$$idom(x) = \begin{cases} semi(x) & semi(x) = semi(y) \\ idom(y) & semi(x) \neq semi(y) \end{cases}$$

按Dfs序从大到小考虑每个点，用并查集维护已经考虑过的点形成的生成森林，以及每个点到祖先路径上半必经点深度最小的点，同时可以求出每个点的半必经点和可能必经点。最后，按Dfs序从小到大考虑每个点，对半必经点与必经点不相等的点进行调整，即可求出每个点的必经点。由每个点到其必经点之间的边构成的树，称为图G的*Dominator Tree*。

对原题中的图构得其*Dominator Tree*，要求的也就是在*Dominator Tree*上最近公共祖先为点1 的点对数，这可以通过求出点1 每个儿子的子树大小$s_1, s_2, \cdots, s_m$，进而求得答案为：

$$\sum_{i=1}^{m}\sum_{j=1}^{i-1} s_i s_j + N - 1$$

时空复杂度

时间复杂度为$O(N\alpha(N))$。
空间复杂度为$O(N+M)$。

1.87 JUN15 CHEFBOOK

试题名称

Chefbook.

题目大意

给出N个点，M条关系，关系(x,y)涉及三个权值$L_{x,y},S_{x,y},T_{x,y}$。要求确定一组整数$P_1,P_2,\cdots,P_N$和$Q_1,Q_2,\cdots,Q_N$的值，满足$P_i,Q_i\geq 0$，且对于所有的关系(x,y)，都有$S_{x,y}\leq L_{x,y}+P_x-Q_y\leq T_{x,y}$。求使$\sum_{x,y}L_{x,y}+P_x-Q_y$最大的方案。

数据范围：$T\leq 3$，$N\leq 100$，$-600\leq L_{x,y}\leq 600$，$-1000\leq S_{x,y}\leq T_{x,y}\leq 1000$，关系无重复。

算法讨论

目标函数可看做是$\sum_{x,y}L_{x,y}+\sum_{x,y}P_x-Q_y$，不妨将$Q_1,Q_2,\cdots,Q_N$，并入$P_{N+1},P_{N+2},\cdots,P_{2N}$，于是，需要最大化$\sum_{i\leq 2N}P_iC_i$，其中$C_i$为变量$P_i$在所有限制关系中的系数和，限制关系都是$P_x-P_y\leq U$的形式。若每个变量在所有限制关系中，仅出现两次，其中一次系数为正，另一次系数为负，则可以用费用流建模求解。而目前的限制关系满足每个关系中，仅有两个变量，其中一个变量系数为正，另一个为负。于是，考虑这个线性规划问题的对偶问题，就满足费用流求解线性规划的条件[4]。如下建立费用流模型：

- 对于$1\leq i\leq 2N$，若$C_i\geq 0$，则从源点S向点i连容量为C_i，费用为0的边，否则从第i向汇点T连容量为$-C_i$，费用为0的边。
- 对于关系(x,y)，从点x向点$y+N$连容量为∞，费用为$T_{x,y}-L_{x,y}$的边；从点$y+N$向点x连容量为∞，费用为$L_{x,y}-S_{x,y}$的边。

对该网络求最小费用最大流，就是$\sum_{x,y}L_{x,y}+P_x-Q_y$的最大值。注意，网络中可能存在负环，此种情况下直接返回无解即可。

最后，考虑如何求出一种方案。在问题的开始，已经给出了若干形如$P_x-P_y\leq U$的条件，由这些条件可建立差分约束系统，使用Bellman-Ford解差分约束系统可以得到一组解，这组解满足限制条件，但不一定是最优的。对于一个限制条件，若其在费用流中的对应的边，在最小费用最大流中流量为正，则表明该限制条件是“紧”的，可把该限制中的不等关系改为相等关系。将这些相等关系也加入差分约束系统。再解差分约束系统就能获得一组最优解了。

[4]有关使用费用流解线性规划的详细内容可参阅Byvoid《NOI2008志愿者招募》一题分析。

时空复杂度

时间复杂度为$O(flow(N,M)+NM)$。
空间复杂度为$O(N+M)$。

1.88 JUL15 EASYEX

试题名称

Easy exam.

题目大意

有一个有$K(K \leq 10^9)$面的骰子，每个面上的数字分别是1到K。同时给出两个参数L和$F(0<L \leq K, 0<F \leq 1000, 0<L \times F \leq 50000)$。

现在将这个骰子掷$N(N \leq 10^9)$次。令a_i为掷出数字i的次数。求$a_1^F * a_2^F * ... * a_L^F$的期望值。

算法讨论

a_i可以看做是a_i个1的和，其中的每一个1，都代表数字i在某次掷骰子时被掷出。倘若将$a_1^F * a_2^F * \cdots * a_L^F$展开，可以发现，这个值就是若干1 的乘积的和。根据期望的线性性，我们尝试把每一部分，即每个若干1的乘积分开来计算，求和即可。

当$F=1$时，每一各个数字在投掷过程中出现的时刻形成的组合，都会对答案的分子产生贡献，其贡献值也就是其他时刻掷骰子所得到点数的方案数。若贡献不为0，则满足L个数字在过程中都出现过，方案数就是$\binom{N}{L} \times L!$，其他时间掷骰子所得点数的方案数为K^{N-L}，因此，答案的分母也就是$\binom{N}{L} \times L! \times K^{N-L}$。

当$F \neq 1$时，我们先来看一个例子。假如某数字i在3个时刻出现，且$F=2$。为了便于描述，我们将这个数字出现的三个时刻用$x,y,z(x=y=z=1)$来表示。 则$a_i^F = a_i^2 = (x+y+z)^2 = x^2+y^2+z^2+2xy+2xz+2yz$。 其中，$x^2, y^2, z^2$产生贡献时只需要占用1 个时刻，而$xy, xz, yz$产生贡献时需要占用2个时刻。这提示我们，可以尝试关于占用时刻数目进行DP。 令$dp_{i,j}(0 \leq i \leq L)$ 为当前已经考虑了前i个数字，它们对答案产生贡献时，需要占用j个时刻，枚举当前第i个数

字占用的时刻数目k，可以得到转移方程：

$$dp_{i,j} = \sum_{k=1}^{j,F} dp_{i-1,j-k} \times f_k \times \binom{j}{k}$$

其中f_k代表一个数字要占用k个时刻的方案数。比如，在上一个例子中，占用一个时刻的有x^2, y^2, z^2，方案数都为1；而占用两个时刻的有$2xy, 2xz, 2yz$，方案数为2。容易得到，可以用下式来求得$f_i (1 \le i \le F)$：

$$f_i = i^F - \sum_{j=1}^{i-1} f_j \times \binom{i}{j}$$

根据dp的结果，就可以得到答案的分母，如下：

$$\sum_{i=1}^{L \times F} dp_{L,i} \binom{N}{i} \times K^{N-i}$$

算法复杂度为$O(F^2 + L^2F^2)$。

注意到，上述的dp的转移过程与求多项式的卷积极为相似。事实上，$dp_{L,i}$的值，也就是一个生成函数L次幂的i次项的系数。我们注意到，不同的数字出现的顺序不同，是被算作多种方案的。因此，这里可以使用指数生成函数。通过构造，得到如下指数生成函数：

$$G(x) = \sum_{i \ge 0} f_i \frac{x^i}{i!}$$

则：

$$dp_{L,i} = [x^i] G^L(x)$$

通过快速幂和FFT，可以在$O(LFlog(LF)logL)$的复杂度内求得$dp_{L,i} (1 \le i \le LF)$。

时空复杂度

总时间复杂度为$O(F^2 + LFlog(LF)logL)$。

空间复杂度主要产生在存储组合数上，为$O(LF^2)$。

1.89 JUL15 HAMILG

试题名称

A game on a graph.

题目大意

两个玩家Askar和Bob，正在用一个在无向图G上的硬币进行游戏。这个游戏如下进行：

- Ascar选择一个起始点，并把硬币放在这个点上。
- 接下来，两个玩家轮流操作。Bob先进行操作。
- 轮到每个玩家操作时，他需要把硬币沿着一条边移至另一点。
- 硬币不能重复到达同一点。
- 无法操作的玩家将输掉这个游戏。

称一个点v为胜利点，当且仅当Askar能够通过选择v为起始点来获胜。假设两个玩家都按最优策略进行操作。给出有N个点，M条边的无向图G，求出有多少胜利点。

数据范围：$N \leq 2000, M \leq 10^6$。

算法讨论

设Askar选择的起始点u为胜利点，且接下来两名玩家各操作K次，Bob每次移动硬币后的终点为$x_1, x_2, \cdots, x_K$，Askar每次移动后的终点为$y_1, y_2, \cdots, y_K$。此时，x_i与$y_i (1 \leq i \leq K)$可以相互匹配，匹配数为K。若此时Bob 还可以操作，设可将硬币移向点v，则点u可与x_1匹配，$x_i (2 \leq i \leq K)$可与y_{i-1}匹配，y_K与v匹配，此时匹配数为$K+1$，较之前获得的匹配更大。而若x_i与y_i的匹配，属于图G的某一个最大匹配，且点u是未被匹配的点，Bob 就无法找到下一步操作的点v。 于是可以发现，点u为胜利点的充要条件是：在任意一个图G的最大匹配中，点u为孤立点，也就是未被匹配。

我们可以使用带花树算法(*Edmonds' Blossom Algorithm*) 求出图G的一种最大匹配。以普通方式实现的带花树算法复杂度为$O(NM)$（实际测试时往往无法达到上界），经过优化的算法可将复杂度将至$O(N^{0.5}M)$。

使用带花树算法求出图G的一种最大匹配后，我们可以枚举点i，并判定其是否是胜利点。若点i为胜利点，则在原图中删去点i后，图的最大匹配较原匹配减少，也就是点i一定会出现在所有的最大匹配中。这个算法的复杂度为$O(N^2M)$。

使用带花树算法求图的最大匹配复杂度较高。实际上，并没有必要多次求最大匹配。我们可以在第一次匹配的基础上，枚举每一个孤立点，仿照带花树算法寻找匹配的过程，从这个孤立点出发寻找增广路。假设原先有匹配：

$$(x_1, y_1), (x_2, y_2), \cdots, (x_j, y_j)$$

现在从孤立点u出发，依次沿图G上的边走过$u, x_1, y_1, x_2, y_2, \cdots, y_i$，此时可以修改匹配为：

$$(u, x_1), (y_1, x_2), \cdots, (y_{i-1}, x_i), (x_{i+1}, y_{i+1}), \cdots, (x_j, y_j)$$

此时点y_i成为了孤立点，而匹配数没有发生改变。可以发现，每一条长度为偶数的增广路的终点，都可能在最大匹配中成为孤立点，也就是游戏中的胜利点。

除了在寻找增广路过程中所访问到的，满足其配偶先被访问的点是符合要求的胜利点，在带花树花*(blossom)*中的所有点，都可以通过调整增广路经过花上点的顺序，使它们都可能是偶数长增广路的终点。因此，在缩花的时候，把在花上的点都标记为胜利点即可。

算法复杂度与带花树算法复杂度相同，为$O(NM)$。

1.90 AUG15 CLOWAY

试题名称

Future of draughts.

题目大意

给出T个无向图，点数为N_i，边数为M_i。现在定义对一个图集合的操作：在初始时，可在每个图上选择一个起始点，接下来，每步选择一个非空子集，并随机移动一步其中的点，操作结束当且仅当所有点回到初始位置。进行Q次询问，每次询问给出三个参数L_i, R_i, K_i，表示询问存在多少种对图$[L_i, R_i]$操作的方案，使得在K_i步内结束。答案对$10^9 + 7$取模。

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

算法讨论

先来考虑只有一个图的情况。

令G为该图的邻接矩阵，则经过K步回到起始点的方案数为矩阵G^K对角线上的元素和，也就是矩阵G^K的迹$tr[G^K]$。

非零向量v可以被称作矩阵A的一个特征向量，当且仅当存在λ使得$Av = \lambda v$。这里λ被称作矩阵A的特征值。矩阵A的特征值集合称作矩阵的谱，记作$sp(A)$。存在性质，若$x \in sp(A)$，则$x^k \in sp(A^k)$。存在定理，矩阵的迹等于其特征值之和。

考虑存在多个图的情况。设A, B是两个图的邻接矩阵。于是可以使用一个点对，来记录操作时，两个图中操作点的位置。因此，两个图可以被合并成一个点数取平方的新图。

$$sp(Merge(A,B)) = \{(\alpha+1)(\beta+1)-1 | \alpha \in sp(A), \beta \in sp(B)\}$$

$$tr[Merge(A,B)^K] = \sum_{i=0}^{K} \binom{K}{i} (-1)^{K-i} \sum_{\alpha \in sp(A)} (\alpha+1)^i \sum_{\beta \in sp(B)} (\beta+1)^i$$

而

$$\sum_{\alpha \in sp(A)} (\alpha+1)^i = tr[(A+I)^K]$$

于是可以扩展到有多个图的情况：

令

$$P(l,r,k) = \prod_{i=l}^{r} tr[(G_i+I)^K]$$

于是可得

$$tr[Merge(G_l, G_{l+1}, \cdots, G_r)^K] = \sum_{i=0}^{K} \binom{K}{i} (-1)^{K-i} P(l,r,i)$$

考虑如何计算$tr[(G+I)^K]$。直接用矩阵乘法暴力预处理复杂度是$O(TN^3K)$的。对于两个矩阵A,B，若要求$tr[A\times B]$，是可以在$O(N^2)$时间内解决的，因为我们只需关注对角线上的答案。于是，采用Baby-Step-Giant-Step思想，可在$O(TN^2K+TN^3\sqrt{K})$复杂度内完成预处理。

因此，对于一组询问L_i,R_i,K_i，答案也就是$tr[Merge(G_{L_i},G_{L_i+1},\cdots,G_{R_i})^{K_i}]$。该式可化成卷积的形式：

$$tr[Merge(G_l,G_{l+1},\cdots,G_r)^K]=K!\sum_{i=0}^{K}\frac{P(l,r,i)}{i!}\times\frac{(-1)^{K-i}}{(K-i)!}$$

枚举$l,r(1\le l\le r\le T)$，再构造多项式$A=\sum_{i=0}^{K}\frac{P(l,r,i)}{i!}x^i$和$B=\sum_{i=0}^{K}\frac{(-1)^i}{i!}x^i$，通过数论变换可以求出两个多项式的卷积，也就得到了答案。注意，因为模数为10^9+7，不能直接做NTT。可以选择三个质模数P_1,P_2,P_3满足$P_1P_2P_3\ge KP^2$，分别在模三个模数的意义下做NTT，再用增量法解线性同余方程组合并。

时空复杂度

时间复杂度为$O(TN^2K+TN^3\sqrt{K}+T^2KlogK)$。
空间复杂度为$O(T^2K+N^2\sqrt{K})$。

1.91 AUG15 DISTNUM

试题名称

Simple Queries

题目大意

维护初始长度为N的序列A，支持Q条操作，操作包含插入、删除、修改和以下两种询问操作：

- 令S是$A_l,A_{l+1},\cdots,A_r$组成的集合，求：

$$(\sum_{1\le i<j<k\le|S|}S_iS_jS_k)\bmod(10^9+7)$$

- 询问$A_l,A_{l+1},\cdots,A_r$中不同元素个数。

数据范围：$N, Q \leq 10^5$。

算法讨论

问题不要求算法在线，因此可以先获取所有修改信息，将原序列进行扩展，使得在后续操作时，不再需要支持动态插入、删除操作，而只需支持修改与查询操作。维护平衡树，即可获得所涉及的每个元素在扩展序列中的下标。

若要询问$A_l, A_{l+1}, \cdots, A_r$中不同元素个数，可以先求出所有的$B_i (1 \leq i \leq N)$，表示在$A_i$之前，值与$A_i$相同的最大下标。若位置$i$要对不同元素个数的询问产生贡献，则$B_i < l$。于是便可以维护线段树套线段树或树状数组套线段树，来支持单点修改与区间询问。

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

$$= \Big(\sum_{1 \leq i \leq |S|} S_i \Big)^3 - 3 \Big(\sum_{1 \leq i \leq |S|} S_i^2 \Big) \Big(\sum_{1 \leq j \leq |S|} S_j \Big) + 2 \sum_{1 \leq i \leq |S|} S_i^3$$

因此，第一种询问操作，可以转化为维护区间整数的$1 \ldots 3$次方和的问题，同样可以用树套树解决。

时空复杂度

时间复杂度为$O((N+Q)log(N+Q))$。
空间复杂度为$O((N+Q)log(N+Q))$。

2 Challenge型试题

2.1 NOV11 STEPAVG

试题名称

Stepping Average.

题目大意

考虑用以下方法求N个数的“迭代平均数”：每次拿出其中两个数，并用它们的平均数取代，重复$N-1$次直到只剩一个数。称这个剩下的数为“迭代平均

数”。给出N个数$A_1, A_2, \cdots, A_N$，要求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：$T \leq 10$，$N = 1000$。K和A_i均在$[1, 10^9]$范围内随机生成。

算法讨论

假设要依次合并数按顺序为$a_1, a_2, \cdots, a_n$，容易知道，最后剩下的数为：

$$Avg = \frac{a_1}{2^{n-1}} + \frac{a_2}{2^{n-1}} + \frac{a_3}{2^{n-2}} + \frac{a_4}{2^{n-3}} + \cdots + \frac{a_n}{2}$$

先对所有数确定一个临时顺序，这个顺序也是它们的大小顺序，并计算出Avg。之后，重复C次，每次随机选择两个数x, y，交换这两个数的顺序，并计算新的Avg，若能使得$|Avg - K|$减小，则保留这两者交换之后的顺序。当C取10^6左右时，可在Tsinsen上获得满分，并在Codechef原比赛中，获得18264.189427的得分，这也是所有提交者中的最优得分。事实上，在比赛中，有很多选手的方案都获得了这个分数，完全相同。或许，这也是本题所难以超越的极限。因此，笔者认为，将本题作为Challenge题有些许不妥之处。

时空复杂度

时间复杂度为$O(T(C + N))$。

空间复杂度为$O(N)$。

2.2 JAN12 AMBIDEX

试题名称

Ambidextrous Chefs.

题目大意

有N种工具，M个厨师。厨师i能够使用工具L_i和R_i。现在需要由这些厨师组成若干个队伍，满足每个队伍中，每种工具至少有一位厨师能够使用。要求组成尽量多的队伍，同时选取尽量少的厨师，以使得分尽量大。设组成的队伍数为C，选取的厨师数为H，则得分为：

$$\frac{C \times N - H}{M}$$

数据范围：$T = 10, 10 \leq N \leq 100, 200 \leq M \leq 1000$。数据随机生成。

算法概述

由于本题不是交互型试题，因此可以设计并运行多个不同的算法，选取得分最高的一种方案。下面介绍三种不同算法，将以下几种算法结合，即可在本题中取得满分。

算法一

考虑每一个厨师队伍，再按顺序考虑每一个尚未被选的厨师。首先选择能够使用的两个工具都未在当前队伍中出现的厨师，再选择能够使用的其中一个工具未在当前队伍中出现的厨师。实际操作时，可以多次对厨师进行*random_shuffle*后再运行上述算法。

算法二

枚举厨师队伍数量，按能够使用该工具的厨师数量从小到大考虑每个工具的分配情况。对于每个厨师队伍，若还没有厨师能够使用当前考虑的工具，那么优先选择两种工具都还没有出现的厨师，再选择只有一种工具还没有出现的厨师。在选取的过程中，会有很多选取方案，这里可以贪心地选取两种工具出现次数和最大的厨师。实际操作时，可以多次对厨师进行*random_shuffle*后再运行上述算法。

算法三

考虑每一个厨师队伍，要使得尽量多的工具，满足与另一种工具匹配，这两种工具由同一厨师来操作。这便可以使用带花树算法来求任意图最大匹配。然后再每次选取另一种工具出现次数最大的厨师，满足当前工具尚未在厨师队伍中出现，直到当前厨师队伍完整。

2.3 MAY12 CKROACH

试题名称

Killing Gs.

题目大意

有N只虫子需要被杀死，现在有M种杀虫剂，第i种杀虫剂的费用为C_i，使用后，第j只虫被杀死的概率为$P_{j,i}$。

现在，要求使用费用和尽量少的杀虫剂（同种杀虫剂不能多次使用），使得每只虫子的生存概率不高于10%。

数据范围：$T = 10, 50 \leq N \leq 200, 50 \leq M \leq 200, C_i \leq 10^6$。

在生成数据时，会从集合$\{1, 2, 5, 10, 100\}$中随机选择一个整数x，对于任意$j(1 \leq j \leq M)$，在$[0, 1)$范围内均匀随机得到一个实数y_j，则$C_j = \lfloor (x + y_j) * \sum_{i=1}^{N} P_{i,j} \rfloor$。

算法讨论

首先可以容易地从读入推断出x和$y_j(1 \leq j \leq M)$的值，并对所有杀虫剂按照y_j值从小到大排序。暴力枚举前d种杀虫剂的使用情况，然后当还有虫子的生存概率高于10%时，选择一种尚未被使用的，估价函数值最高的杀虫剂进行使用。一种简单的估价函数设计方式为，使用这种杀虫剂会带来的生存概率高于10%的虫子的生存概率下降值，再减去费用的若干倍后的结果。取$d = 8$，估价函数中费用减去100倍，可在Tsinsen 获得113分。

时空复杂度

时间复杂度为$O(2^d NM^2)$。

空间复杂度为$O(NM)$。

2.4 JUN12 CLOSEST

试题名称

Closest Points.

题目大意

给出空间内$N(N \leq 50000)$个白点，$M(M \leq 50000)$个黑点。要求对每个黑点找一个欧几里得距离最近的白点。得分与正确求得最近白点的黑点数正相关。

算法讨论

先由N个白点建立空间内的KD-Tree，然后对于每个黑点，要在KD-Tree中查询最近点。如果直接查询，虽然能够保证每个黑点的答案都正确，但复杂度过高。因此，可以在KD-Tree查询的时候加入一些优化：

- 若查询点到当前KD-Tree上节点所表示的区域边界的距离，已经超过了当前的最近距离，就可以跳过当前节点。
- 在往KD-Tree上儿子节点递归时，优先访问查询点所在的区域。
- 对每个黑点所允许经过的KD-Tree上的节点数设置一个上限，若在查询时已经达到该上限，则直接退出。

该算法可在Tsinsen上获得130分。

2.5 SEP12 SIMNIM

试题名称

Simultaneous Nim.

题目大意

给出N个异或和为0的数$A_1, A_2, \cdots, A_N$，要求把这些数分成K组，满足每组中的数异或和为0。 目标是使K的值尽量大。

数据范围：$N \leq 1000, A_i < 2^M, M \leq 60, T \leq 10$。

算法概述

以下两个算法，在每次执行前，都将A数组使用*random_shuffle*打乱顺序。结合使用两个算法，即可通过本题。

算法一

对于任意两个相同的数，都可以先组成一组。在剩下的数中，按顺序考虑未被分组的数，若当前未被分组的数的异或和xor在剩下的数中有相同的出现，则将这些数和值为xor的数分为一组，清空xor。重复上述过程直至所有数均被分组。

算法二

每次选取一些数组成一个组时，贪心地希望选出的数尽量少。可以根据数的每一位异或和为0的条件，列出方程组，方程组中的未知数表示每一个数是否选择。使用高斯消元解方程，方程组中的自由元取值为0，其他未知数中，任意选择一个未知数强制取1，即可确定其他未知数的取值。将本次解方程求得的取值为1的未知数所代表的在原数组中的数，都归为一组。重复上述过程直至所有数均被分组。

时空复杂度

设算法一执行α次，算法二执行β次，则：
时间复杂度上界为：$O(T(\alpha NlogN + \beta N^3M/32))$。
空间复杂度为：$O(NM)$。

2.6 JUN13 CHAORNOT

试题名称

To challenge or not.

题目大意

给出一个数集$B(|B| \leq 10^5)$，要求这个数集的一个尽量大的子集，满足其中不存在三个元素，能构成等差数列。

算法讨论

按顺序考虑数集B中的每一个元素，若将这个元素加入当前所选的子集，会导致能够选出三个构成等差数列的元素，则跳过这个元素，否则，使用一个带有随机化的函数，计算是否要将这个元素加入所选子集。多次执行这个算法取最优即可。该算法可在Tsinsen 上获得365分以上。

时空复杂度

时间复杂度为$O(|B|Answer)$。
空间复杂度为$O(|B|)$。

2.7 DEC13 SMPAINT

试题名称

Art in Digital Age.

题目大意

给出一$N \times N(N \leq 1000)$的图片，其中该图片已经被若干个总颜色数不超过100的彩色矩形填充。要求构造一种填充方案来得到该图片。

算法讨论

按从上到下、从左到右的顺序遍历该图片的每个像素，若该像素未被填充，则需要填充上目标颜色。与此同时，向下、右一维地寻找最长能延伸的距离，满足所经过的像素颜色都与当前要填充的相同，选择较长的一个方向。之后，再沿另一个方向延伸，也要满足颜色相同，求出最长距离。这样便得到了一个二维区域，将这个区域填上同一种颜色即可。该算法可在Tsinsen上通过测试。

时空复杂度

时间复杂度为$O(N^2)$。
空间复杂度为$O(N^2)$。

2.8 OCT14 CHEFPNT

试题名称

Chef and Painting.

题目大意

在一个$N \times M$的棋盘上，有K个格子是黑色的，另一些是白色的。现在可以进行多次操作，每次选择一个格子(x, y)，再选择一个方向，可以是水平或竖直，然后在该方向上前进，直到遇到一个红色或黑色格子，并把经过的格子染成红色。要求构造方案，进行尽量少的操作，使得所有白色格子变为红色。

数据范围：$1 \leq N, M \leq 100$，$K \leq \min(N \times M - 1, 3000)$。

算法讨论

由于题目不要求在线，可以设计多个算法分别运行，取最优。

- **算法一**：按从上到下，从左到右的顺序遍历每个格子，每当遇到一个白色格子，则计算出从该格子向右、下延伸分别最远能走的距离，并取距离较长的一个方向，将所经过的格子进行一次染色。
- **算法二**：对于每一行中，连续的白色格子形成了一些区间，而一个区间可一次染色。特殊地，对于一些长度为1，在竖直方向连续的区间，可以一次染色。

结合上述两个算法，可在Tsinsen上获得101分。

时空复杂度

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

2.9 JAN15 SEAND2

试题名称

Sereja and Number Division 2.

题目大意

给出一个长度N位的整数A，同时给出M个整数$B_1, B_2, \cdots, B_M$，要求对A中的数字重排，来使得$\sum_{i=1}^{N} A \mod B_i$尽量小。

数据范围：$T \leq 100$，$N \leq 1000$，$M \leq 100$，$B_i \leq 10^6$。

算法讨论

对于$1 \leq i \leq N$，$1 \leq j \leq M$，都可以预处理出$10^{i-1} \mod B_i$的值。可以对A中的数字随机打乱顺序α次，每次打乱后，再进行β次的随机调整，也就是每次选择两位x, y，交换x, y位上的数字，可以在$O(M)$的复杂度内计算出新的代价，与最优答案进行比较即可。经实际测试，该算法效果较佳，可在Tsinsen上获得满分。

时空复杂度

时间复杂度为$O(\alpha NM + \alpha\beta M)$。
空间复杂度为$O(NM)$。

2.10 JUL15 MAXDIFFW

试题名称

Maximum Difference Walk.

题目大意

给一个有$N(N \leq 50)$个点的带边权完全有向图。有两名玩家A和B要在这张图上玩一个游戏，两人轮流操作，玩家A先手。操作者每次选取一条之前未被选取过的边，满足这条边的起点与上一条被选取的边的终点相同。当无法再选出符合条件的边时，游戏结束。玩家的得分为在游戏中所选取的边的权值和。

这是一道交互型*Challenge*试题。要求设计一个程序，来实现玩家A的操作过程，以与玩家B进行交互。已知玩家B只有两种策略：

1. 每次选择能选的边中权值最大的边。

2. 每次随机选择一条能选的边。

在一局游戏中，玩家B只会选取其中的一种，且不会在中途改变策略。

要求使玩家A的得分与玩家B的得分之差尽量大。

算法讨论

由于玩家B的策略在一局游戏中不会发生改变，且一共只有两种可能的策略，因此可以尝试尽快确定玩家B所采用的是那种策略，之后再对采取两种不同策略的情况下，分别设计算法。

若玩家B采取的是第一种策略，则他在每一轮操作中所作出的操作都是可以被玩家A所预测的。每一局玩家A操作完毕后，寻找接下来能选择的权值最大的边，再看这条边是否就是玩家A在下一轮中选取的边，若不是，则玩家B所采取的是第二种策略，否则，有很大可能采取的就是第一种策略。期望情形下，如果前两局玩家B作出的操作都与预测相符，就基本可以认为玩家B选择了第一种策略。

如果知道了玩家B选取的是第一种策略，那么对于玩家A来说，在其作出决策的时候，可以提前地考虑到未来的情况。因此，在每一次玩家A操作时，都可以枚举2 $\sim$ 3步内的操作，并预测出玩家B的操作，选取在这一段时间内得分最高的一种方案。

若玩家B采用的是随机策略，则无法准确地预测未来的情况。这时，可以设计估价函数，对于每一条可选的边进行估价，从而选取估价值最高的边。具体设计方案很多，这里介绍几种比较简单的设计方式：

1. 提取在选择了当前边后，下一轮玩家B可以选择边的权值，计算这些权值的中位数或平均数作为当前边的估价值。
2. 计算权值平方和的平均数再开平方后的结果作为当前边的估价值。
3. 计算权值平方和开平方后，再除上总可选边数的结果作为估价值。
4. 计算权值与下一轮操作方案数按比例合成后值，再取平均值后的结果作为估价值。

在实际调试时，也可以在估价函数上设计一些额外的参数，多次调整参数的取值来获得更佳的效果。