IOI2015 中国国家集训队第一次作业

泛做表格

浙江省绍兴市第一中学　贾越凯

2015 年 1 月

目　录

Chapter 1

Codeforces (96)

Cyclical Quest

Codeforces 235C

题目大意：

有一个长度为 L 的文本串 S，有 N 个询问，每次询问给出一个模板串 x，求文本串中有多少个长度与它相同的子串 y，使得 x 与 y 循环等价。

$|S|$，$\sum |x| \leq 10^6$

关键字：

后缀自动机

算法讨论：

建立 S 的后缀自动机，对于询问串 x，设其长度为 l。先把它倍长，然后在后缀自动机里走，维护匹配串的状态(x 当前的前缀的最长后缀，出现在原串中)，如果当前前缀长度 $\geq l$，则在满足匹配长度 $\geq l$ 的情况下，当前状态不断沿父亲跳，跳到到最后把当前匹配串出现的次数(当前状态的子树大小)加入答案中。

时空复杂度：

时间：$O(|S| + \sum |x|)$

空间：$O(|S|)$

Graph Game

Codeforces 235D

题目大意：

N 个点 N 条边的无向图，每次随机选一个点，删掉这个点后图变成了若干连通块，把每个连通块的大小加入答案 ANS 中，然后递归处理每个连通块。求 ANS 的期望大小。

$N \leq 3000$

关键字：

概率　DFS

算法讨论：

如果存在一对点 u, v，满足刚要删掉 u 时 v 与 u 连通，则对答案的贡献就是 1。于是 ANS 的期望就是：每对点 x, y 发生上述事件的概率之和。

先来看树的情况。对于点对 u, v，设 u 到 v 的路径上的点数为 m，则所求概率就是 $\frac{1}{m}$。可以用归纳法证明：如果原图只有 x 到 y 的路径这 m 个点，显然成立；否则，设树的点数为 n，如果这一步选择删路径上的点，概率为 $\frac{m}{n} \cdot \frac{1}{m} = \frac{1}{n}$，如果删非路径上的点，归纳到点数为 $m-1$ 的情况，概率为 $\frac{n-m}{n} \cdot \frac{1}{m} = \frac{n-m}{nm}$，两部分相加得概率为 $\frac{1}{m}$。

对于环加外向树的情况也类似。如果 u 到 v 的路径在树上，则同上；否则，u 到 v 就有两条路径，设这两条路径的公共部分的点数为 x，剩余部分的点数分别为 y, z，则这种情况的概率为分别不删两条路径上的点的概率之和减去同时不删两条路径上的点的概率，即 $\frac{1}{x+y} + \frac{1}{x+z} - \frac{1}{x+y+z}$。

时空复杂度：

时间：$O(N^2)$

空间：$O(N^2)$

Number Challenge

Codeforces 235E

题目大意：

求

$$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c}d(ijk)$$

其中 $d(n)$ 为 n 的正约数个数。

$a,b,c\leq 2000$

关键字：

数论

算法讨论：

首先可以证明，

$$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c}d(ijk)=\sum_{\substack{i=1\\ \gcd(i,j)=1}}^{a}\sum_{\substack{j=1\\ \gcd(j,k)=1}}^{b}\sum_{\substack{k=1\\ \gcd(i,k)=1}}^{c}\lfloor\frac{a}{i}\rfloor\lfloor\frac{b}{j}\rfloor\lfloor\frac{c}{k}\rfloor$$

$$\begin{aligned}ANS&=\sum_{\substack{i=1\\ \gcd(i,j)=1}}^{a}\sum_{\substack{j=1\\ \gcd(j,k)=1}}^{b}\sum_{\substack{k=1\\ \gcd(i,k)=1}}^{c}\lfloor\frac{a}{i}\rfloor\lfloor\frac{b}{j}\rfloor\lfloor\frac{c}{k}\rfloor\\&=\sum_{i=1}^{a}\lfloor\frac{a}{i}\rfloor\left(\sum_{\substack{j=1\\ \gcd(i,j)=1}}^{b}\sum_{\substack{k=1\\ \gcd(i,k)=1\\ \gcd(j,k)=1}}^{c}\lfloor\frac{a}{i}\rfloor\lfloor\frac{b}{j}\rfloor\lfloor\frac{c}{k}\rfloor\right)\\&=\sum_{i=1}^{a}\lfloor\frac{a}{i}\rfloor\sum_{\substack{d=1\\ \gcd(i,d)=1}}^{\min(b,c)}\mu(d)\cdot F\left(i,\lfloor\frac{b}{d}\rfloor\right)\cdot F\left(i,\lfloor\frac{c}{d}\rfloor\right)\end{aligned}$$

其中

$$F(d,n)=\sum_{\substack{i=1\\ \gcd(i,d)=1}}^{n}\lfloor\frac{n}{i}\rfloor$$

记忆化 $F(d,n)$，然后直接做即可。

由于 $\sum_{i=1}^{n}\lfloor\frac{n}{i}\rfloor=O(n\log n)$，所以总的计算次数(不包括计算 gcd)为 $O(n^2\log n)$。

时空复杂度：

时间：$O(n^2\log^2 n)$，其中 $n=\max(a,b,c)$

空间：$O(n)$

Tape Programming

Codeforces 238D

题目大意：

有一个长度为 n，只由数字、“<”和“>”构成的串，还一个指针，指针初始指向串首，方向向右。

每一次重复以下操作，直到指针指向串外：

1. 如果该位置是数字，输出这个数字，并把这个数字减 1，指针沿原方向移动。如果这个数字原来是 0，则从串中删除这个数字；
2. 如果该位置是“<”或“>”，指针的移动方向改为箭头方向，并按新方向移动。如果移动后指针的位置还是箭头，则从串中删除原位置的字符；

有 m 个询问，每次给出 l, r，求只对 $[l, r]$ 的子串进行操作，每个数字会输出多少次。

$n, m \leq 10^5$

关键字：

模拟

算法讨论：

可以发现，每个时刻前指针的位置集合都是原串的一个前缀。开始对原串模拟一遍，记录每个时刻前各个数字输出的总量。对于一个询问区间，只需找到第一个进入该区间的时刻(一定是从左往右进)和第一个离开该区间的时刻，然后把两个时刻的答案相减即可。

时空复杂度：

时间：$O(10n + m \log_2 n)$

空间：$O(100n)$

Meeting her

Codeforces 238E

题目大意：

n 个点的有向图，要从点 A 到点 B。有 k 辆公交车，每辆公交车从点 s_i 驶向 t_i，路线为任意一条从点 s_i 到点 t_i 的最短路。如果当前的点上有公交车经过，就可以坐上去，乘到接下来的任意一个点。判断在最坏情况下，是否能从点 A 到点 B，如果可以，求最坏情况下至少要乘几趟公交车。

$n, k \leq 100$

关键字：

最短路　动态规划

算法讨论：

如果最坏情况下在某个点 u 可以乘上公交车 i，则点 u 一定要在点 s_i 到点 t_i 的最短路上。

设 $f[i][j]$ 表示在公交车 i 上，到达点 j 时，最坏情况下能否到达点 B，如果可以则 $f[i][j]$ 表示至少要乘几趟公交车。转移分两种：

1. 对于公交车的可能的下一个点 k，如果所有的 k 都可以到达 B，则 j 也能到达点 B，同时更新 $f[i][j]$ 为 $f[i][k]$ 的最大值；
2. 对于所有 j 能乘上的公交车 k，更新 $f[i][j]$ 为 $f[k][j]+1$ 的最小值。

由于转移有环，可以类似 Bellman-Ford 算法，不断更新，直到不能更新为止，这样最多只会更新 k 次。

时空复杂度：

时间：$O(n^3k)$
空间：$O(n^2)$

Road Repairs

Codeforces 240E

题目大意：

n 个点，m 条边的有向图，有一些边已经存在，还有一些边可以添加。求要使得能从点 1 到达所有的点，最少添加的边数，并输出方案。

$n, m \leq 10^5$

关键字：

最小树形图

算法讨论：

令已经存在的边的边权为 0，需要添加的边的边权为 1，即最小树形图问题。用朱一刘算法，时间复杂度看似是 $O(nm)$ 的，但可以证明实际是 $O(m \log_2 n)$ 的。

时空复杂度：

时间：$O(m \log_2 n)$

空间：$O(n + m)$

TorCoder

Codeforces 240F

题目大意：

长度为 n 的字符串(只有小写字母)，有 m 个操作，每次可以将区间 $[l,r]$ 中的字符重排，组成一个字典序最小的回文串(如果不能组成回文串就忽略这次操作)，求 m 次操作后的字符串。

$n,m \leq 10^5$，$a_i \leq 10^9$

关键字：

线段树

算法讨论：

对每种字符的位置维护一棵线段树。每次操作先求出每种字符在该区间中的个数，如果奇数个数的字符有两种或以上，则不能组成回文串；否则，如果有奇数个数的字符，则先放一个该种字符在区间中间。然后从小到大依次放字符即可。

时空复杂度：

时间：$O(26(n+m)\log_2 n)$

空间：$O(26n)$

Friends

Codeforces 241B

题目大意：

n 个数 a_i，选出 m 对数，使得这 m 对数的异或值的和最大。

关键字：

逐位确定　字母树

算法讨论：

显然选出的是异或值最大的前 m 对数，只要确定第 m 大的异或值是多少，然后统计大于它的异或值的和即可。

首先把所有数按位插入一棵 01 字母树中，然后逐位确定，维护每个数在字母树中的位置。需要统计的是异或值大于当前确定出来的数的个数与它们的和，只需在字母树中记录子树和与子树中第 i 位为 1 的数的个数即可。

时空复杂度：

时间：$O(n \log_2^2 a_i)$
空间：$O(n \log_2^2 a_i)$

Numbers

Codeforces 241D

题目大意：

一个 n 的排列 a_i，选出其中的若干个数，使得它们的异或和为 0，且把它们依次连起来得到的数是 p 的倍数。构造一种方案，或判断无解。

$n, p \leq 50000$

关键字：

动态规划

算法讨论：

可以证明，只需把 $a_i < 32$ 的数提出来做动态规划即可：

设取 $a_i \leq x$ 的 x 个数($x+1$ 是 2 的幂次)，则总方案共有 $2^x - 1$ 种。它们的异或和可以看做是在 $[0, x]$ 中均匀分布的，连起来得到的数模 p 也可以看做是在 $[0, p)$ 中均匀分布的，所以解大约有 $\frac{2^x}{xp}$ 种。取 $x = 31$ 时已经足够了。

时空复杂度：

时间：$O(32^2 p)$

空间：$O(32^2 p)$

Flights

Codeforces 241E

题目大意：

n 个点，m 条边的有向图，要把每条边的长度变成 1 或 2，使得从点 1 到点 n 的所有路径长度都相等。判断是否存在这样的方案，如果存在输出任意一种。保证存在至少一条从点 1 到点 n 的路径。

$n \leq 1000$，$m \leq 5000$

关键字：

图论　最短路　差分约束系统

算法讨论：

首先，原图中的一些边是没用的，即那些不在任意一条从点 1 到点 n 的路径上的边，它们的长度取任意值都不会影响答案。我们把所有的边分成有用边和无用边。对于一条从 u 到 v 的边，如果能从点 1 到点 u，且能从点 v 到点 n，则它就是有用边。求点 1 能到的所有点只要从点 1 开始 DFS 一遍即可；求能到点 n 的所有点也只要把所有边反向，然后从点 n 开始 DFS 一遍即可。

同理也可以定义有用点和无用点，其中有用点是那些在任意一条有用边上的点。如果一张图满足题目中的性质，则也有：对于所有的有用点，从点 1 到它的所有路径长度都相等。

于是我们可以对每个有用点 i，设 d_i 表示从点 1 到点 i 的路径长度。对于一条边 $u \rightsquigarrow v$，它的长度就是 $d_v - d_u$。由于边长都是 1 或 2，所以有不等式

$$1 \leq d_v - d_u \leq 2$$

整理可得两个不等式

$$d_v \leq d_u + 2$$

$$d_u \leq d_v - 1$$

上面两个不等式很像最短路算法中的三角不等式。于是可以根据上面两个不等式构建差分约束系统，即对原图的每条边 $u \rightsquigarrow v$，在新图中从 u 到 v 连边权为 2 的边，从 v 到 u 连边权为 -1 的边，然后求点 1 到每个点的最短路。如果新图存在负权环，则原问题无解，否则每条边的权值为 $d_v - d_u$。

上述算法需要判断一张图是否有负权环，所以不能用 Dijkstra 算法，要用 Bellman-Ford 算法或 SPFA 算法求最短路，当某个点更新超过 $n-1$ 次时即存在负权环。

时空复杂度：

时间：$O(nm)$(Bellman-Ford) 或 $O(km)$(SPFA)

空间：$O(n+m)$

Race

Codeforces 241F

题目大意：

$n \times m$ 的方格，格子可以是障碍、路口或道路的一部分。道路都是水平或竖直的，道路的交汇处形成了路口，没有相邻的道路或相邻的路口。从路口移到相邻格需要 1 的时间，从道路格移动到相邻格需要 $1 \sim 9$ 的一个时间。

给定一条路径的起点坐标和终点坐标，还有路径经过的各个路口(沿最短路)。沿着这条路径走，求时刻 k 时所在的坐标。

$n, m \leq 100$，$k \leq 10^5$

关键字：

模拟

算法讨论：

由于路口都是水平道路和竖直道路的交点，所以相邻两个路口之间的路径是直线，直接模拟即可。

时空复杂度：

时间：$O(nm + k)$
空间：$O(nm)$

Colorado Potato Beetle

Codeforces 243C

题目大意：

一个无限大的网格，从一个格子开始，移动 n 次，每次可以沿 4 个方向走若干个格子，走到的格子会被喷上杀虫剂。现在有虫子从无限远处开始入侵，对于一个已经被入侵的格子，它相邻的没有被喷杀虫剂的格子会被入侵。求没有被入侵的格子数量。

$n \leq 1000$，每次移动距离 $\leq 10^6$

关键字：

离散 BFS

算法讨论：

把移动过程中的顶点求出来，把它们的横纵坐标离散，于是网格就变成 $O(n) \times O(n)$ 的了，然后直接 BFS 即可。

时空复杂度：

时间：$O(n^2)$
空间：$O(n^2)$

Cubes

Codeforces 243D

题目大意：

在一个 $n \times n$ 的正方形网格中，每个格子中竖直堆放着 a_{ij} 个立方体。从无限远的地方沿着向量 (v_x, v_y) 看过去，问能看到几个立方体。

$n \leq 1000$

关键字：

线段树

算法讨论：

把 n^2 个格子沿向量 (v_x, v_y) 投影，对每个格子都可以得到一个区间。作向量 (v_x, v_y) 所在直线的垂线，然后把这 n^2 个区间按到垂线的距离排序，即被看到的顺序。依次枚举每个区间 $[l, r]$，设这个区间代表的格子中有 x 个立方体，则这一列立方体新增的能看到的立方体个数为 $x-$ 区间 $[l, r]$ 内的最小值，然后把 $[l, r]$ 内的所有值与 x 取最大值。用线段树实现这些操作即可。

时空复杂度：

时间：$O(n^2 \log_2 n)$

空间：$O(n^2)$

Piglet's Birthday

Codeforces 248E

题目大意：

有 n 个盒子，第 i 个盒子中有 a_i 罐蜂蜜，初始所有蜂蜜都是满的。

有 m 个操作 (u, v, k)，每次可以从盒子 u 中取出任意 k 罐蜂蜜，全部喝光，然后把 k 个空瓶放入盒子 v 中。输出每次操作后全是空瓶的盒子个数的期望。

$n, m \leq 10^5$，$a_i \leq 100$，$k \leq 5$

关键字：

动态规划　概率

算法讨论：

全是空瓶的盒子个数的期望即每个盒子中满瓶个数为 0 的概率之和。设 $f_i[j]$ 表示第 i 个盒子中，空瓶满瓶个数为 j 的概率。维护每个盒子中瓶子的数量，对当前操作类似背包一样转移。

时空复杂度：

时间：$O(100km)$

空间：$O(100n)$

Donkey and Stars

Codeforces 249D

题目大意：

平面里有 n 个点，开始选择原点，然后过原点作两条与x轴夹角分别为 α_1, α_2 的直线，然后在两直线中间任选一点重复操作，直到没有可选点。求最多能选出的点数。

$n \leq 10^5$

关键字：

最长上升子序列

算法讨论：

根据夹角求出斜率 k_1, k_2，对于点 (x, y)，下一个点 (x', y') 要满足

$$k_1 x' + b_1 < y' < k_2 x' + b_2$$

其中 $b_1 = y - k_1 x, b_2 = y - k_2 x$。整理得

$$y - k_1 x < y' - k_1 x'$$

$$y' - k_2 x' < y - k_2 x$$

对每个点 (x_i, y_i)，重新令

$$x_i = y_i - k_2 x_i$$

$$y_i = y_i - k_1 x_i$$

则只要选出的点满足 x_i 递减，y_i 递增即可。把所有点按 x_i 从大到小排序，然后直接套用最长上升子序列算法。

时空复杂度：

时间：$O(n \log_2 n)$
空间：$O(n)$

Endless Matrix

Codeforces 249E

题目大意：

对于一个如图所示的无限矩阵 $a_{i,j}$，给出 x_1, y_1, x_2, y_2，求

$$\sum_{i=x_1}^{x_2}\sum_{j=y_1}^{y_2} a_{i,j}$$

1	2	5	10	17	26
4	3	6	11	18	27
9	8	7	12	19	28
16	15	14	13	20	29
25	24	23	22	21	30
36	35	34	33	32	31

$x_1, y_1, x_2, y_2 \leq 10^9$，数据组数 $T \leq 10^5$

关键字：

算法讨论：

一个矩形内数字的和可以拆成 4 个左上角在 $(1,1)$ 的矩形内数字的和。对于一个左上角在 $(1,1)$ 的矩形内数字的和，又可以拆成一个左上角在 $(1,1)$ 的正方形内数字的和与若干等差数列的和。随便推一下公式即可。

时空复杂度：

时间：$O(T)$
空间：$O(1)$

Two Sets

Codeforces 251D

题目大意：

有 n 个数 a_i，把这 n 个数分成两个集合 S_1, S_2，令 X_1 为集合 S_1 中所有数的异或和，X_2 为集合 S_2 中所有数的异或和，求一种方案，使得 X_1+X_2 最大，在满足这个条件下再使得 X_1 最小。

$n \leq 10^5$，$a_i \leq 10^{18}$

关键字：

逐位确定　高斯消元　bitset

算法讨论：

令 X 为所有数的异或和。首先不考虑让 X_1 最小。考虑 X 二进制的第 i 位，如果它是 0，则 X_1 和 X_2 二进制的第 i 位分别是 $0,1$ 或 $1,0$，对于 X_1+X_2 的贡献是相同的；如果它是 1，则 X_1 和 X_2 二进制的第 i 位分别是 $0,0$ 或 $1,1$。

从高到低依次枚举每一位，逐位确定出 X_1。对于第 i 位(X 的第 i 位为 1)，尝试 X_1 的这一位放 1 是否可以，如果可以则增加一个限制条件。把每个数选与不选看成未知数，则可以根据这个限制条件列出若干个异或方程，之要判断方程是否有解就能判断 X_1 的这一位是否可以放 1 了。

判断一个异或方程组是否有解，可以用高斯消元把方程组消成上三角矩阵，但如果直接消元的话做一遍就要 $O(nl^2)$ 了，显然不行。对于一个新加入的方程，我们可以根据已有的上三角矩阵去消这个方程，这样做一遍的复杂度就是 $O(nl)$ 了。

现在考虑让 X_1 最小。X_1 有多解只会出现在 X 的第 i 位为 0 的时候，这时尝试 X_1 的这一位放 0 是否可以，类似上述方法，用高斯消元做到 $O(nl)$。

由于方程组是异或方程组，所以可以用 bitset 把变量合并，复杂度除以 bitset 的位数 w。

时空复杂度：

时间：$O(\frac{nl^2}{w})$
空间：$O(\frac{nl}{w})$

Tree and Table

Codeforces 251E

题目大意：

$2n$ 个点的树，把它放入一张 $2 \times n$ 的表格里，表中的每个格子表示树中的一个点，要求原来在树中相邻的点在表中仍然相邻。求总方案数。

$n \leq 10^5$

关键字：

树形动态规划

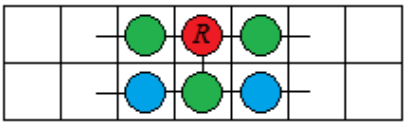
算法讨论：

如果 $n = 1$，方案数为 2。

如果树的最大度数大于 3，则方案数为 0。

如果树是一条链，先设 $f(n)$ 表示 $2 \times n$ 个点的链放入表格中，其中起点在左上角的方案数。容易得到，$f(1) = 1$，$f(n) = f(n-1) + 1$，所以 $f(n) = n$。然后枚举起点在原表格第一行的位置，再乘 2，可以得到总方案数为 $2n^2 - 2n + 4$。

否则，树中至少有一个度数为 3 的点，设其中的一个为 R，不妨令它在表格中的第一行，以它为根建树，则它的 3 个儿子在表格中应该是下图这个样子：

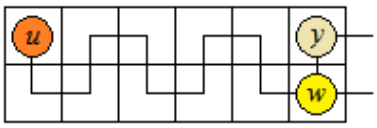


它们在表格中的位置可以根据子树大小得到。3! 枚举它们的顺序，同时枚举图中蓝色的那两个点，则可以转化成如下的子问题：

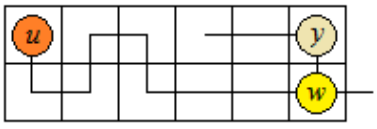
子问题 $G(u, v)$ 表示点 u 与点 v 分别在表格的左上角与左下角时的方案数(等价于在右上角与右上角)。则方案一定是先由这两个点同时延伸出两条链，直到其中一条链没有点为止，然后又转化成下面的子问题：

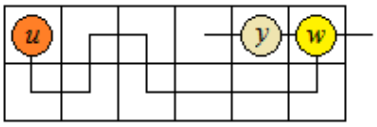
子问题 $F(u)$ 表示点 u 在表中左上角的方案数(等价于在右上角)。首先，如果 u 的子树大小 s_u 为奇数，则方案数为 0；如果 u 的子树是一条链，则根据上面的 $f(n)$，可以得到方案数为 $f(s_u/2) = s_u/2$；否则，找到 u 的子树中离 u 最近的有两个儿子的点 w，设 w 的两个儿子分别为 x, y，则转移有如下三种：

1. 用 u 到 w 之间的点去覆盖 w 所在格子的左边，w 的右边是子问题 $G(x, son(y))$：

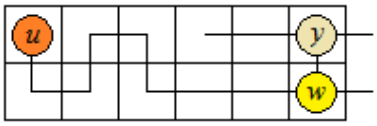


2. 如果 y 的子树是一条链，用 u 到 w 之间的点和 y 的子树去覆盖 w 所在格子的左边，w 的右边是子问题 $F(x)$。但有下面两种情况：





3. 如果 y 也有两个儿子，且其中一个儿子的子树是一条链，则用 u 到 w 之间的点和 y 的儿子的子树去覆盖 w 所在格子的左边，w 的右边是子问题 $G(x, son(y))$：



上述转移中，x 与 y 的关系可以互换。且只要转移的种类确定了，w 左边的方案也就确定了，所以每种转移的方案数只有一种。

于是，直接递归求 F 与 G 即可(需要记忆化)。

时空复杂度：

时间：$O(n)$

空间：$O(n)$

Printer

Codeforces 253E

题目大意：

有一台打印机，每秒可以打印一页纸。某些时刻会收到一些任务，每个任务用可以一个三元组 (t_i, s_i, p_i) 表示，分别代表接到任务的时间、要打印的页数和任务优先级。

当收到一个任务时，任务会进入一个队列直到任务完成。每个时刻，打印机会选择队列里优先级最高的一个任务，并打印一页。

现在有 n 个任务，但有且仅有一个任务优先级未知，却知道它完成的时间。现在要求出那个任务的优先级，和所有任务完成的时间。

关键字：

二分答案　堆

算法讨论：

显然任务优先级越高，它完成的时间越早。二分那个任务的优先级，然后用优先队列存储任务，直接模拟。

时空复杂度：

时间：$O(n \log_2^2 n)$
空间：$O(n)$

Rats

Codeforces 254D

题目大意：

一张 $n \times m$ 的地图，每个格子可能是障碍、空地或老鼠。可以在非障碍格中放置炸弹，炸弹可以炸到与这个格子的最短路 $\leq d$ 的所有格子。现在要在地图中放两个炸弹，消灭所有的老鼠，求一种方案，或判断无解。

$n, m \leq 1000$，$d \leq 8$

关键字：

枚举　BFS

算法讨论：

由于 $d \leq 8$，所以一个炸弹最多能炸到的格子数为 $2d^2 + 2d + 1 = 145$，两个炸弹能炸到的格子数 ≤ 290。显然，如果老鼠的数量大于这个数，则直接无解，所以老鼠的数量也可以看做是 ≤ 290 的。

对于任意一个包含老鼠的格子，BFS 找出所有与它的最短路 $\leq d$ 的格子，则其中的一个炸弹必须放在这些格子里。所以可以枚举第一个炸弹的位置，BFS 一遍，如果它已经能炸到所有的老鼠了，则找到了一种方案；否则，对于任意一个不能被炸到的包含老鼠的格子，再去 BFS 找出所有与它的最短路 $\leq d$ 的格子，另一个炸弹必须放在这些格子里，于是再枚举另一个炸弹的位置再判断即可。

时空复杂度：

时间：$O(4^3 d^6 + nm)$

空间：$O(nm)$

Liars and Serge

Codeforces 256D

题目大意：

有 n 个人，每一个一直说真话或假话。现在每个人说出了说真话的人数 a_i (可能是假的)，求有多少种可能的 a_i，使得可以确定出其中的 k 个人说的是假话。

$k \leq n \leq 2^8$，n 是 2 的幂次。

关键字：

动态规划　打表

算法讨论：

设说真话的人说的是 x，则 a_i 中一定出现了 x 个 x，可以确定出不符合这个要求的人说的是假话。所以问题相当于求，有多少个数列，满足 x 的出现次数不等于 x 的 x 共有 k 个。从小到大填数字，设 $f[i][j][k]$ 表示填了 i 个数，填到了数字 j，不符合要求的数共有 k 个的方案数。转移时枚举下一个要填的数字的个数即可。

这样时间复杂度是 $O(n^3k)$ 的，无法承受。但发现 n 是 2 的幂次，所以当 $n = 2^8$ 时打表即可。

时空复杂度：

时间：$O(n^3k)$
空间：$O(n^2k)$

Greedy Elevator

Codeforces 257E

题目大意：

有一幢 m 层的大楼，和一台电梯。有 n 个人要做电梯，对每个人，知道他开始等电梯的时刻、他初始位于的楼层与他想去的楼层。在某个时刻，电梯会停在某一层，电梯中所有想到这一层的人出电梯，这一层中所有等电梯的人进入电梯。

电梯初始在楼层 1，它运行方式如下：如果电梯是空的且没有人在等电梯，则下一时刻电梯仍会停在当前层；否则，设 u 为所有在当前层上面等的人与在电梯中且要去当前层上面的人之和，d 为所有在当前层下面等的人与在电梯中且要去当前层下面的人之和，如果 $u \geq d$，下一时刻电梯会上升一层，否则电梯会下降一层。

求每个人到达目的地的时刻。

$n, m \leq 10^5$

关键字：

模拟　树状数组

算法讨论：

把时间离散，维护每一层中等电梯的人与在电梯中且要到这一层的人之和，用数据结构直接模拟。需要的操作有：查询某一层前面的人数之和，在某一层插入一些人、把某一层的人删光、查询某一层前面与后面第一个有人的楼层，用树状数组即可。

时空复杂度：

时间：$O(m + n \log_2 m)$

空间：$O(n + m)$

Little Elephant and Broken Sorting

Codeforces 258D

题目大意：

一个 N 的排列，有 M 个操作，每次可以交换两个位置上的数，但每个操作有 $\frac{1}{2}$ 的概率不操作。求最后排列中逆序对个数的期望。

$n, n \leq 1000$

关键字：

动态规划　概率

算法讨论：

设 $f[i][j]$ 表示第 i 个位置上的数大于第 j 个位置上的数的概率，答案即所有的 $f[i][j]$ $(i < j)$ 之和。对于每个操作维护一下即可。

时空复杂度：

时间：$O(n^2 + nm)$

空间：$O(n^2)$

Dividing Kingdom

Codeforces 260E

题目大意：

平面上 N 个点，求两条平行于 x 轴和两条平行于 y轴的直线(不与任何一个点相交)，把点集分成 9 部分，使得每一部分中的点数 s_i 等于给定的 9 个数(不一定有序)。$N \leq 100000, x_i, y_i \leq 10^9$

关键字：

枚举　函数式线段树

算法讨论：

枚举 9 格对应的点数，根据同行中的点数和与同列中的点数之和二分找出这 4 条直线，然后就是判断每格中的点数是否符合要求，即二维数点问题。可以用函数式线段树做，把点按 y 坐标排序，每个点 i 建立一棵线段树维护前 i 个点的 x 坐标。

时空复杂度：

时间：$O((N + 9!) \log_2 N)$
空间：$O(N \log_2 N)$

Maxim and Increasing Subsequence

Codeforces 261D

题目大意：

给出长度为 n 的序列 b_i，序列 a_i 是由 t 个数列 b_i 拼起来的。求 a_i 的最长上升子序列长度。

$n \leq 10^5$，$t \leq 10^9$，$n \cdot maxb \leq 2 \times 10^7$，数据组数 $k \leq 10$

关键字：

动态规划

算法讨论：

显然 $n \times t$ 个 b_i 是不必要的，由于答案最大为 $maxb$，所以 t 最多只要 $maxb$ 即可。设 $f[i][j]$ 表示前 i 个数，末尾数 $\leq j$ 的最长上升子序列长度。这样时间复杂度看似是 $O(nmaxb^2)$ 的，但最多只会更新 $O(nmaxb)$ 次。

时空复杂度：

时间：$O(knmaxb)$

空间：$O(maxb)$

Maxim and Calculator

Codeforces 261E

题目大意：

有两个数 a,b，初始分别为 $1,0$。每次可以把 b 增加 1，或把 a 变成 $a\cdot b$。求在区间 $[l,r]$ 中有多少数字 x，使得可以通过不超过 p 次操作，把 a 变成 x。

$p\leq 100$，$l\leq r\leq 10^9$

关键字：

动态规划

算法讨论：

让 a 变成 x 的最少步数即，把 x 分成若干个数(≥ 2)的乘积，分出的数的个数加上最大的数。所以，x 中不会有大于 p 的质因子。于是可以预处理出所有 $\leq r$ 的最大质因子不超过 p 的数(最多 3000000 个)，设 $f[i][j]$ 表示分出的最大的数不超过 i，组成 j 最少要分出的数的个数。

时空复杂度：

时间：$O(3000000p)$

空间：$O(3000000)$

Rhombus

Codeforces 263E

题目大意：

一个 $n \times m$ 的矩阵 $a_{i,j}$，给出 k，在 $k \le x \le n-k+1$ 且 $k \le y \le m-k+1$ 的范围内定义函数

$$f(i,j) = \sum_{i=1}^{n} \sum_{j=1}^{m} a_{i,j} \cdot \max(0, k - |i-x| - |j-y|)$$

在定义域内找一对整数 (a,b)，使得 $f(a,b)$ 最大。

$n, m \le 1000$

关键字：

前缀和

算法讨论：

对于一对 x, y，$f(x,y)$ 相当于原矩阵的每个元素乘上一个菱形的权值矩阵中对应元素的和，菱形的权值矩阵(中心为 (x,y)，边长 $k = 4$)如图 1 所示：

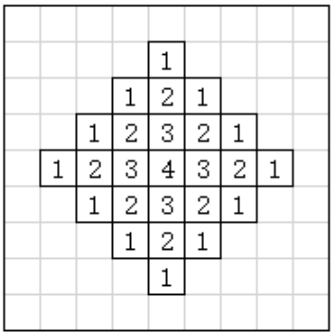


图 1

这个和，又相当于如图 2 所示的 k 个矩形内 $a_{i,j}$ 的总和：

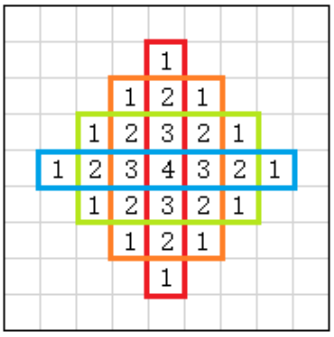


图 2

一个矩形内元素的和，可以通过预处理前缀矩形的和，然后加加减减。对于预处理好的前缀矩形和 $b_{i,j}$，它们对当前的 $f(x,y)$ 的贡献如图 3 所示：

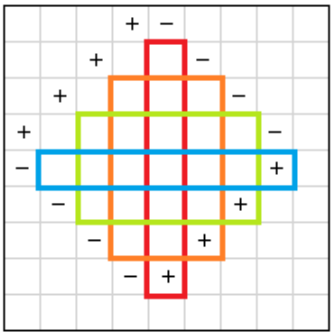


图 3

于是只需再预处理这四条线上的 $b_{i,j}$ 之和即可。

时空复杂度：

时间：$O(nm)$
空间：$O(nm)$

Colorful Stones

Codeforces 264D

题目大意：

两个字符串，长度分别为 n, m，只包含三种字母，初始两个人都分别站在两个串的串头。可以发若干条指令，每条指令是三种字母中的一个，指令发出后，站在对应字母上的人往后移一步。两个人的位置被记为一个状态，求不同的状态个数。

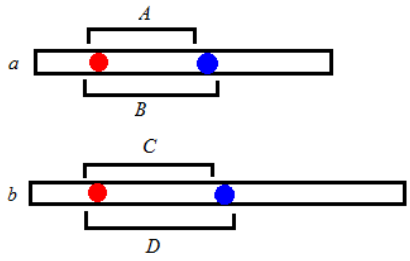
$n, m \leq 10^6$

关键字：

单调性

算法讨论：

首先判断一个状态是否可以到达。如图所示，红色为初始状态，蓝色为要判断的状态。图中有 4 个子串 A, B, C, D，显然，如果存在一个指令串 I，满足 A, C 是 I 的子序列，且 B, D 不是 I 的子序列，则该状态可以到达。



我们要判断是否有这样一串指令存在。显然，如果 D 是 A 的子序列，或 B 是 C 的子序列，则一定不存在。还有一种情况，如果 B 是形如“...xy”的，D 是形如“...yx”的，则也不存在。

所以，枚举 B 的末尾，满足 D 不是 A 的子序列且 B 不是 C 的子序列的 D 的末尾是一端区间，且随着 B 末尾往后移，区间也会往单调后移。维护合法的 D 的末尾的组成区间，即可算出合法的 D 的末尾的个数，然后再减去第二种情况中不合法的 D 即可。

时空复杂度：

时间：$O(n+m)$
空间：$O(n+m)$

Roadside Trees

Codeforces 264E

题目大意：

有 n 个位置，m 个操作，每次操作可以在位置 p 插入一个数 h，或删掉从左往右第 x 个数，操作完后所有的数都会增加 1。输出每次操作后序列的最长上升子序列长度。保证任何时刻序列中不会有两个相同的数。

$n, m \leq 10^5$，$x, h \leq 10$

关键字：

线段树

算法讨论：

首先由于最长上升子序列只需要相对高度，所以可以把“操作完后所有的数都会增加 1”这一过程去掉，把增加的量加到插入的数中。

以序列的位置为横坐标，序列中数的大小为纵坐标，序列中的所有数可以看成平面中的若干个点。维护两个数组 $f[i], g[j]$，$f[i]$ 表示把所有点按横坐标从大到小排序，以点 i 为结尾的关于纵坐标的最长下降子序列长度，$g[i]$ 表示把所有点按纵坐标从大到小排序，以点 i 为结尾的关于横坐标的最长下降子序列长度。

对于插入一个点，由于纵坐标 ≤ 10，且序列中没有相同的数，所以 $f[i], g[i]$ 都是最后面长度不超过 10 的一段受影响，直接重新计算即可；同理，对于删除一个点，由于横坐标比它小的点 ≤ 10，所以 $f[i], g[i]$ 也是最后面长度不超过 10 的一段受影响，直接重新计算即可。

由于涉及修改一个数、询问一段区间的最大值，用线段树即可。

时空复杂度：

时间：$O(10m \log_2 n)$

空间：$O(m \log_2 n)$

BerDonalds

Codeforces 266D

题目大意：

n 个点的无向图，求一个点(可以在边上的任意一点)，使得它到最远的点的距离最近，输出那个最近的最远距离。

$n \leq 200$

关键字：

枚举　最短路

算法讨论：

设那个点为 s，枚举 s 所在的边 u, v。设最优解为 w，则必有至少两个点，他们到 s 的距离都为 w。所以 w 一定是 $\frac{1}{2}$ 的整数倍。

预处理所有点两两之间的最短路 $d[i][j]$。假设 s 离所在边的一个端点 u 的距离为 x，边的长度为 l，则点 i 到 s 的最短距离是 x 的函数，$f_i(x) = \min(d[u][i] + x, d[v][i] + l - x)$，而当前这条边的答案就是 $\min_x(\max_i(f_i(x)))$。

$f_i(x)$ 的图像形如一个 $\wedge$，且 $f_i(0) = d[u][i], f_i(l) = d[v][i]$。答案一定在这些 $\wedge$ 的交点上，图像 i 与图像 j 相交(i 在 j 左边)，要满足 $f_i(0) > f_j(0)$ 且 $f_i(l) < f_j(l)$，交点的纵坐标为 $\frac{1}{2}(f_i(l) + f_j(0) + l)$。于是把 n 个图像按 $f_i(0)$ 从大到小排序，从左到右处理每个交点即可。

时空复杂度：

时间：$O(n^3 \log_2 n)$

空间：$O(n^2)$

More Queries to Array...

Codeforces 266E

题目大意：

长度为 n 的序列 a_i，有 m 个操作，每次可以把一段区间都赋成一个数，或给出一个 k，对区间 $[l,r]$ 求

$$\sum_{i=l}^{r} a_i \cdot (i-l+1)^k$$

$n, m \leq 10^5, 0 \leq k \leq 5$

关键字：

线段树　二项式定理

算法讨论：

根据二项式定理，把求和式展开，有

$$\begin{aligned}\sum_{i=l}^{r} a_i \cdot (i-l+1)^k &= \sum_{i=l}^{r} a_i \cdot \left(\sum_{j=0}^{k} \binom{k}{j} i^j (1-l)^{k-j}\right) \\ &= \sum_{j=0}^{k} \binom{k}{j} (1-l)^{k-j} \cdot \left(\sum_{i=l}^{r} a_i \cdot i^k\right)\end{aligned}$$

所以只要用线段树维护 $a_i \cdot i^k$ 的区间和即可。

时空复杂度：

时间：$O(kn \log_2 n)$
空间：$O(kn)$

Berland Traffic

Codeforces 267C

题目大意：

n 个点，m 条边的无向图，1 号点是源点，n 号点是汇点。每条边可以有任意方向流，但不能超过其流量限制，且要满足流量平衡条件。求一个最大的流，使得对于任意两个点 x, y，x 到 y 所有路径上的边的流量(可以有正负)之和相等，输出方案。

$n \leq 100, m \leq 10000$

关键字：

高斯消元

算法讨论：

题目中的那个限制，很像基尔霍夫电压定律。所以，可以对每个点设 x_i，表示它相对于源点的流量，这样一条边 i, j 的流量就是 $x_i - x_j$。然后可以根据流量平衡条件对 $2 \sim n-1$ 的点列出 $n-2$ 个方程。由于流量是成比例的，所以可以把 x_1 设为 0，x_n设为 1，用高斯消元求出所有边的流量，然后再根据流量限制求得最大的流。

时空复杂度：

时间：$O(n^3)$
空间：$O(n^2)$

Wall Bars

Codeforces 268D

题目大意：

n 个位置，每个位置可以放 $1 \sim 4$ 的数，如果两个相同的数之间的距离 $\leq h$，则可以从左边那个数到右边那个数。开始可以到位于 $1 \sim h$ 中的任意一个数。求有多少种放数的方案，使得位置 $(n-h+1) \sim n$ 中至少有一个位置能到达。

$n \leq 1000$，$h \leq 30$

关键字：

动态规划

算法讨论：

设 $f[i][a][b][c][d]$ 表示放了 i 个数，上一个数字 1 到末尾的距离为 a，上一个数字 2 到末尾的距离为 b……的方案数。如果某种数字已经不能到当前位置了，则为 h。由于末尾那个数字到末尾的距离为 0，所以 a, b, c, d 中有一个只要记录两种状态即可。

时空复杂度：

时间：$O(nh^3)$
空间：$O(nh^3)$

Maximum Waterfall

Codeforces 269D

题目大意：

有 n 条线段，高度分别为 h_i，左右端点分别为 (l_i, r_i)。同时，在高度为 0 与高度为 t 处还有两条线段，左右端点分别为 $(-10^9, 10^9)$，保证没有两条线段存在没有公共点。

如果两条线段 i, j，满足：

1. $\max(l_i, l_j) < \min(r_i, r_j)$ (两条线段存在公共部分)；
2. $h_j \leq h_i$；
3. 不存在线段 k，(i, k) 与 (k, j) 同时满足以上条件。

则线段 i 到线段 j 有 $\min(r_i, r_j) - \max(l_i, l_j)$ 的流量。

找一条从最上面的线段到最下面的线段的路径，使得相邻两条线段流量的最小值最大。

$n \leq 10^5$

关键字：

动态规划　数据结构

算法讨论：

把所有线段按高度从大到小排序，设 $f[i]$ 表示从上往下流到线段 i 的最小流量的最大值。维护线段覆盖成的若干小段，转移时，枚举所有被当前线段覆盖的小段来更新，同时把这些小段删掉，替换为当前的线段。用 set 等数据结构维护即可。

时空复杂度：

时间：$O(n \log_2 n)$

空间：$O(n)$

Dima and Figure

Codeforces 273D

题目大意：

在一个 $n \times m$ 的网格中，初始所有格子都是白色的。现在要把某些格子涂成黑色，要求：

1. 至少有一个被涂黑的格子；
2. 所有黑色格子组成一个连通块；
3. 如果只能经过黑色的格子，从一个黑色格子移动到另一个黑色格子的最少步数为它们的曼哈顿距离。

求方案数。

$n, m \leq 150$

关键字：

动态规划　前缀和

算法讨论：

显然黑色格子组成一个类似菱形的东西，即对于每一行，最左边的黑色格子所在的列先减后增，最右边的黑色格子所在的列先增后减。设 $f[i][j][k][0\ldots1][0\ldots1]$ 表示黑色连通块的高度为 i，当前最左边的黑色格子在列 j，最右边的黑色格子在列 k，左边有没有开始递增，右边有没有开始递减的方案数。从 $f[i-1][j'][k']$ 转移时，可行的 (j', k') 组成一个矩形，于是只需每次维护 $f[i][j][k]$ 的二维前缀即可。

时空复杂度：

时间：$O(nm^2)$
空间：$O(m^2)$

Dima and Game

Codeforces 273E

题目大意：

有 n 对数 $(l_i, r_i)(1 \le l_i < r_i \le p)$，两个人轮流操作，每次选出一对数 (l_i, r_i) 满足 $r_i - l_i > 2$，把它替换成 $\left(l_i + \left\lfloor \frac{r_i - l_i}{3} \right\rfloor, l_i + 2 \cdot \left\lfloor \frac{r_i - l_i}{3} \right\rfloor\right)$ 或 $\left(l_i, r_i + \left\lfloor \frac{r_i - l_i}{3} \right\rfloor\right)$，无法操作者输。

求这 n 对数有多少种方案，使得当两人都采取最优策略时，先手有必胜策略。

$n \le 1000$，$p \le 10^9$

关键字：

博弈论　找规律　动态规划

算法讨论：

首先对于一对数 (l, r)，它的 SG 值只与长度有关。设 $\mathrm{SG}(i)$ 表示 $r - l = i$ 的 SG 值，则有

$$\mathrm{SG}(i) = \mathrm{mex}\left\{\mathrm{SG}\left(\left\lfloor \frac{i}{3} \right\rfloor\right), \mathrm{SG}\left(i - \left\lfloor \frac{i}{3} \right\rfloor\right)\right\}$$

打表发现，相邻一段 $\mathrm{SG}(i)$ 的值是相同的，且都是 0,1,2。进一步发现，总的段数很少，最大时也就 100 多段。于是可以求出所有的段，设 $g[i]$ 表示 SG 值为 i 的区间个数，再设 $f[i][j]$ 表示选了 i 个区间，SG 值异或和为 j 的方案数，DP 一下即可。

时空复杂度：

时间：$O(n)$
空间：$O(n)$

The Last Hole!

Codeforces 274C

题目大意：

平面中有 n 个圆，时刻 t 时每个圆的半径都是 t。某些时候相交的圆之间会形成洞，也会有一些洞消失。求最后一个洞消失的时刻。

$n \leq 100$

关键字：

枚举　计算几何

算法讨论：

考虑什么时候会形成洞。首先是三个圆，如果它们形成了洞，洞消失的时刻即它们形成的三角形的外接圆半径，洞消失的位置即外接圆的圆心。所以，如果能够形成洞，就要满足外心在三角形内，即锐角三角形。于是只要枚举所有锐角三角形，然后判断它们的外心是否被其他圆覆盖即可。

还有一种情况：4 个圆心形成一个矩形。这是也是能形成洞的，洞消失的时刻为矩形外接圆的半径(对角线的一半)，洞消失的位置为矩形的中心。所以还要再枚举矩形判断一遍。由于确定三个点后另一个点也确定了，所以矩形的个数是 $O(n^3)$。

对于其他的洞，可以证明，在圆变大的过程中，最后都能变成上面两种情况。

时空复杂度：

时间：$O(n^4)$
空间：$O(n)$

Mirror Room

Codeforces 274E

题目大意：

一个 $n \times m$ 的网格中，有 k 个格子是障碍。从 (x, y) 的中心沿一个对角线方向射出一道光线，光线碰到网格边界或障碍会反射(遵循光的反射定理)，光线反射若干次后会进入循环。求光线经过的不同格子的数量。

$n, m, k \leq 10^5$

关键字：

模拟　数据结构

算法讨论：

首先可以发现，对于一个格子，最多只有一种方向(可以是相反方向)的光线会经过它。

然后考虑光反射的次数。光线在边界的反射次数为 $O(n+m)$，对于一个障碍格，由于最多只有两条光线经过这个格子，所以对反射次数的贡献为 2，所以总的反射次数就是 $O(n+m+k)$。

于是直接模拟光路的运行，每次找下一个反射点。用 set 等数据结构记录每种斜行中的障碍，询问时二分即可。注意光线可能会以相反方向沿同一条路径走，这时答案要除以 2。

时空复杂度：

时间：$O((n+m+k)\log_2 k)$

空间：$O(n+m+k)$

Google Code Jam

Codeforces 277D

题目大意：

有 n 道大题，每道大题有两小题，做对的得分分别为 s_{1i}, s_{2i}，所需的时间分别为 t_{1i}, t_{2i}。第一小题一定能做对，第二小题必须要完成第一小题后才能做，且第二小题有 p_i 的概率做错。做题目的总用时为正确做完的最后一小题的完成时间。求期望总分的最大值，和满足期望总分最大的前提下，最少的期望用时。

关键字：

动态规划　概率

算法讨论：

假设做的题目的集合已确定，如何安排顺序使得总用时最少。

首先显然是先把全部的第一小题安排在前面，所以只考虑第二小题的排列。假设第二小题的最优排列中，题目 i 和 j 是相邻的，它们的贡献是

$$(S + t_{2i})p_j(1 - p_i)P + (S + t_{2i} + t_{2j})(1 - p_j)P$$

其中 S, P 都是常数。现在交换 i, j，要求现在的贡献大于刚才的贡献，消去常数，即

$$\begin{aligned} t_{2i}p_j(1 - p_i) + (t_{2i} + t_{2j})(1 - p_j) < t_{2j}p_i(1 - p_j) + (t_{2i} + t_{2j})(1 - p_i) \Longrightarrow \\ -t_{2i}p_ip_j - t_{2j}p_j < -t_{2j}p_ip_j - t_{2i}p_i \Longrightarrow \\ t_{2i}p_i(1 - p_j) < t_{2j}p_j(1 - p_i) \end{aligned}$$

所以把每道大题按上面的规则排序，然后直接动态规划即可。

时空复杂度：

时间：$O(nt)$

空间：$O(t)$

Cow Tennis Tournament

Codeforces 283E

题目大意：

n 个不同的数 a_i，对于每一对 i,j，如果 $a_i > a_j$，则 i 到 j 之间有一条有向边。有 m 个操作，每次给出一个区间 $[l,r]$，对于一对 i,j 如果 $a_i \in [l,r]$ 且 $a_j \in [l,r]$，则把 i,j 之间的边反向。求最后有多少不同的点对 (i,j,k)，满足 i 到 j 有边，j 到 k 有边，k 到 i 有边。

$n, m \leq 10^5$

关键字：

线段树

算法讨论：

首先补集转化，变成求不合法的方案数。现在问题相当于，一个 $n \times n$ 的 01 矩阵，初始 $(i,j)(i > j)$ 上的元素为 1，其余为 0，每次把一个矩形区域内的元素取反，设 s_i 表示最后矩阵中第 i 行 1 的个数(不包括对角线)，求 $\sum_{i=1}^{n} \binom{s_i}{2}$。

把所有矩形拆成两个底边在矩阵底部的矩形，并按矩形的顶边排序。可以发现，对于第 i 行和第 $i+1$ 行，假设已经做完了所有顶边 $\leq i$ 的矩形操作，第 i 行的状态与第 $i+1$ 行的状态仅仅相差初始时位于 $(i+1,i)$ 的那个元素。于是可以用线段树维护当前行的状态，直接通过第 i 行的状态得到第 $i+1$ 行的状态，同时统计答案。

时空复杂度：

时间：$O((n+m)\log_2 n)$

空间：$O(n+m)$

Positions in Permutations

Codeforces 285E

题目大意：

求有多少种 N 的排列 p_i，满足 $|p_i - i| = 1$ 的 i 的个数等于 K。

$k \leq n \leq 1000$

关键字：

动态规划　容斥原理

算法讨论：

首先任意一种方案可以得到一个二分图，合法方案就是要满足二分图中与相邻的点之间的匹配数恰为 k。先考虑只由相邻边的匹配构成方案数，设 $f[i][j][0\ldots1][0\ldots1]$ 表示已经做到第 i 个点，有了 j 个合法匹配，其中末尾两个点是否已经连边的方案数，简单转移。再考虑其他不由相邻边的匹配构成的方案数。设 $g[i]$ 表示至少有 i 个相邻边匹配的方案数，即剩下 $(n-i)$ 个点之间乱连，易得 $g[i] = (n-i)!\sum f[n][i][a][b]$。由于要求合法匹配恰为 k 的方案数，只要容斥一下就行了：

$$ans[i] = g[i] - \sum_{j=i+1}^{n} \binom{j}{i} ans[j]$$

时空复杂度：

时间：$O(n^2 + nk)$

空间：$O(n^2)$

Tourists

Codeforces 286D

题目大意：

在点 $(1,0)$ 和点 $(-1,0)$ 有两个人，两人同时出发以相同的速度沿y轴正方形移动。有 n 个障碍，第 i 个障碍在第 t_i 秒瞬时出现，是一条 $(0,l_i)-(0,r_i)$ 的线段。有 m 个询问，每次给出两人出发的时间 q_i，求他们不能相互看见的总时间。

$n,m\le 10^5$，$l_i,r_i,q_i\le 10^9$，$q_i<q_{i+1}$

关键字：

并查集　排序

算法讨论：

由于可能有相交的障碍，所以先要把它们拆开成一些新障碍。把所有障碍的区间端点离散，按时间排序，然后去覆盖这一段坐标。如果一段坐标是新覆盖的，它就是一个新障碍。这一过程可以用并查集做，覆盖一段坐标后，把覆盖边界合并，这样每段坐标只会被做到一次。

然后由于所有障碍不相交，所以分开考虑。对于障碍 (t_i,l_i,r_i)，在第 q_i 秒出发，到达点 $(0,l_i)$ 的时间为 l_i+q_i，到达点 $(0,r_i)$ 的时间为 r_i+q_i。两人看不见只有两种情况：如果 $l_i+q_i\le t_i\le r_i+q_i$，即 $t_i-r_i\le q_i\le t_i-l_i$，则相交时间为 $q_i+r_i-t_i=q_i-(t_i-r_i)$；如果 $t_i\le l_i+q_i$，即 $q_i\ge t_i-l_i$，则相交时间为 $r_i-l_i=(q_i-(t_i-r_i))-(q_i-(t_i-l_i))$。否则不会相交。把 $[t_i-r_i,t_i-l_i]$ 作为一个区间，则在 q_i 左边的区间左端点 l_i，对答案贡献 l_i-q_i；在 q_i 左边的区间右端点 r_i，对答案贡献 q_i-r_i。由于 q_i 递增，所以只需把区间端点排序后扫一遍即可。

时空复杂度：

时间：$O((n+m)\log_2 n)$

空间：$O(n+m)$

Ladies' Shop

Codeforces 286E

题目大意：

有 n 个数 a_i。选出一个集合 $b_i|1 \le b_i \le m$，使得：

1. 对于每个 a_i，都可以由这些 b_i 拼成(同一个 b_i 可以用多次)；
2. 对于任意一种 b_i 的拼法(同一个 b_i 可以用多次)，如果拼出的和 $\le m$，都存在一个 a_i 等于拼出的和。

求个数最少的集合 b_i，或判断无解。$n, m \le 10^6$

关键字：

FFT

算法讨论：

首先判无解。最可行的一组解为，令集合 $b_i = a_i$。这样显然满足条件 1。要判条件 2 是否成立，只要把 b_i 两两相加，判断是否存在 a_i 等于它们的和即可。

然后看是否可以优化这一解，即从当前集合中删去一些数。这一过程中条件 2 显然还是成立的。而要删去的数为，可以由若干个小的 b_i 拼成的数。由于条件 2 的存在，判断一个数是否可以删除也只要判断它是否能由两个 b_i 拼成即可。

求 b_i 两两相加得到的所有数，是个卷积，用 FFT 即可。

时空复杂度：

时间：$O(m \log_2 m)$
空间：$O(m)$

Polo the Penguin and Lucky Numbers

Codeforces 288E

题目大意：

给出两个只由 4,7 构成的 n 位数 $l, r(l < r)$，设 $[l, r]$ 中所有由 4,7 构成的数依次为 $a_1, a_2, \ldots, a_m$，求 $a_1a_2 + a_2a_3 + \cdots + a_{m-1}a_m$。

$n \leq 10^5$

关键字：

数位 DP

算法讨论：

由于长度固定为 n，先考虑 $[444\cdots4, 777\cdots7]$ 之间的数的答案，其为 $f[n]$。则 $f[n+1]$ 可由 $f[n]$ 推出，它一定是前面加个 4，或前面加个 7，或是 $4777\cdots7 \times 7444\cdots4$。求 $f[n+1]$ 时还需预处理 n 位的所有合法数字的和。

最终答案为 $[444\ldots4, r]$ 的答案减去 $[444\ldots4, l]$ 的答案，然后像数位 DP 一样做，固定一个前缀，后面可以任取。注意这时也有类似 X4777$\cdots$7 $\times$ X7444$\cdots$4 的情况。

时空复杂度：

时间：$O(n)$

空间：$O(n)$

Distinct Paths

Codeforces 293B

题目大意：

$n \times m$ 的网格，有 k 种颜色，有一些格子已经被涂上了颜色。现在要把未涂颜色的格子涂上颜色，要求从左上角到右下角的任意一条路径不会包含两个颜色相同的格子。求方案数。

$n, m \leq 1000$，$k \leq 10$

关键字：

搜索

算法讨论：

由于路径长度都是 $n+m-1$，所以如果 $n+m-1 > k$ 直接无解，所以 $n+m-1$ 最多为 10。考虑搜索，直接搜显然不行。对于那些初始没有涂过的颜色，我们可以在搜索时规定它们是无序的，最后答案乘一个排列数。再加一些剪枝，如，用二进制状态记录当前格子左上角矩形区域内出现的颜色，该格子显然不能涂这些颜色。

时空复杂度：

时间：$O(?)$
空间：$O(?)$

Ksusha and Square

Codeforces 293D

题目大意:

一个 n 个点的凸多边形，任选其内部或边界上的两个格点，以它们之间的线段为对角线作一个正方形，求正方形面积的期望。

$n \leq 10^5$，坐标绝对值 $m \leq 10^6$

关键字:

枚举　期望　几何

算法讨论:

设共有 s 个合法格点 x_i, y_i，则答案为

$$\frac{1}{s \cdot (s-1)} \sum (x_i - y_j)^2 + (y_i - y_j)^2$$

显然可以把横纵坐标分开算，两次算时只需交换所有点的横纵坐标即可。枚举横坐标 x，由于多边形是凸的，所以横坐标为 x 的合法的格点的纵坐标是一个区间，可以简单地算出它们的个数，把它们记在 $a_1, a_2, \ldots, a_m$ 中。

于是，现在就是要计算

$$\sum_{i=1}^{n} a_i a_(i-j)^2$$

$$= a_2 a_1 + a_3(a_2 + 2^2 a_1) + a_4(a_3 + 2^2 a_2 + 3^2 a_1) + \cdots$$

观察发现，

$$(a_2 + 2^2 a_1) - a1 = a_2 + 3a_1$$

$$(a_3 + 2^2 a_2 + 3^2 a_1) - (a_2 + 2^2 a_1) = a_3 + 3a_2 + 5a_1$$

$$\cdots\cdots$$

$$(a_3+3a_2+5a_1)-(a_2+3a_1)=a_3+2(a_1+a_2)$$
$$(a_4+3a_3+5a_2+7a_1)-(a_3+3a_2+5a_1)=a_4+2(a_1+a_2+a_3)$$
$$\cdots\cdots$$

于是只需维护 a_i 的前缀和与 $a_2+3a_1, a_3+3a_2+5a_1, a_4+3a_3+5a_2+7a_1, \ldots$ 即可。

时空复杂度：

时间：$O(n+m)$
空间：$O(n+m)$

Close Vertices

Codeforces 293E

题目大意：

一棵 n 个点的树，每条边有个边权。求有多少点对 (u,v)，满足 u 到 v 路径上的边数 $\leq A$，且 u 到 v 路径上的边权和 $\leq B$。

$n \leq 10^5$

关键字：

点分治　树状数组

算法讨论：

把原树进行点分治，对分治树中的一个点 p，统计 p 的子树中的合法点对数。由于可以递归计算，所以只需统计 p 的子树中，两点间路径经过点 p 的点对数。

求出 p 的子树中的每个点到该点之间的边数 d_i 与边权和 e_i，把它们按 e_i 从小到大排序。枚举一个点 i，把所有满足 $e_i + e_j \leq B$ 的点 j 的 d_j 插入树状数组(j 是单调的)，然后答案加上树状数组中 $\leq A - d_i$ 的点数。

由于上面的计算过程会把同在 p 的一个儿子的子树中的点对算进去，这是不合法的，所以还要对 p 的每个儿子做一遍，答案减去它们的和。

时空复杂度：

时间：$O(n \log_2^2 n)$

空间：$O(n)$

Shaass and Painter Robot

Codeforces 294D

题目大意：

一个 $n \times m$ 的网格，开始每个格子都是白色的。在边界上有一个机器人，初始朝着某个斜的方向。机器人会沿这个方向走下去，如果走到边界会遵循光的反射定律改变方向。机器人每走到一个格子会把这个格子染黑，直到整个网格变成黑白相间的。求机器人行走的步数，或判断机器人永远不会停下来。

$n, m \leq 10^5$

关键字：

模拟

算法讨论：

反射只会在边界上，最多只有 $(n+m-2)$ 次反射。如果要满足条件，必须走遍所有的反射点。于是直接去找下一个反射点，直到所有反射点都被走过。

时空复杂度：

时间：$O(n+m)$

空间：$O(n+m)$

Greg and Caves

Codeforces 295D

题目大意：

$n \times m$ 的网格，要在上面画一个图形。设图形顶端在第 L 行，底端在第 R 行，对于第 i 行$(L \le i \le R)$，图形的左端点在 l_i，右端点在 r_i，要求：

1. 对于所有的 $i(L \le i \le R)$，$r_i - l_i \ge 1$；
2. 存在一个 $t(L \le t \le R)$，使得对于 $L \le i < j \le t$，$l_i \ge l_j$，$r_i \le r_j$；
   对于 $t \le i < j \le R$，$l_i \le l_j$，$r_i \ge r_j$。

求方案数。

$n, m \le 2000$

关键字：

动态规划　前缀和

算法讨论：

设 $f[i][j]$ 表示在 i 行的网格中画图，图形底端在第 i 行，且底端长度为 j 的方案数。则有：

$$f[i][j] = 1 + \sum_{k=2}^{j} f[i-1][k] \cdot (j-k+1)$$

直接计算显然不行，但发现

$$f[i][j-1] = 1 + \sum_{k=2}^{j-1} f[i-1][k] \cdot (j-k)$$

$$f[i][j] - f[i][j-1] = \sum_{k=2}^{j} f[i-1][k]$$

所以再记一个 $f[i-1][j]$ 的前缀和即可。

计算答案时，枚举图形最长的那一行 i 和它的长度 j，则有

$$ANS = \sum_{i=1}^{n}\sum_{j=2}^{m}(m-j+1)\cdot S(i,j)\cdot f[n-i+1][j]$$

其中

$$\begin{aligned} S(i,j) &= 1+\sum_{k=2}^{j-1} f[i-1][k]\cdot(j-k+1) \\ &= f[i][j]-f[i-1][j] \end{aligned}$$

时空复杂度：

时间：$O(nm)$

空间：$O(nm)$

Yaroslav and Algorithm

Codeforces 301C

题目大意：

定义一种新算法：

1. 输入是一个字符串 a；
2. 算法有若干条命令，第 i 条命令形如“$s_i>>w_i$”或“$s_i<>w_i$”，其中 s_i 和 w_i 长度最多为 7 (可以为空)，只包含数字字符和‘?’字符；
3. 算法每次找一条编号最小的命令 i，使得 s_i 是 a 的子串，如果找不到，算法终止；
4. 设找到的命令编号为 k，在字符串 a 中，出现的第一个 s_k 被 w_k 替换，如果这条命令形如“$s_i>>w_i$”，算法继续执行，否则终止；
5. 算法的输出是算法终止时字符串 a 的值。

给定 n 个整数 x_i，构造一种这样的算法，使得对于所有的 a_i，把它们转成字符串后输入，能输出 a_i+1 对应的字符串。

$n \leq 100$，$a_i \leq 10^{25}$

关键字：

构造

算法讨论：

使用“?”作为指针。开始先把“?”加到串前，然后移到串尾，把“?”替换成“??”，再把形如“x??”替换成“$(x+1)$”，但要把“9??”替换成“??0”，最后把“??”替换成“1”。

具体的算法如下：

```
0??<>1
1??<>2
```

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

时空复杂度：

时间：$O(1)$
空间：$O(1)$

Yaroslav and Arrangements

Codeforces 301E

题目大意：

如果一个长度为 r 的数列 a_i 满足：

1. $|a_1 - a_2| = 1, |a_2 - a_3| = 1, \ldots, |a_{r-1} - a_r|, |a_r - a_1| = 1$；
2. a_1 为 a_i 的最小值；

则称这个数列为好数列。

统计满足以下条件的长度为 r 的数列 b_i 的数量：

1. 数列中的元素不下降；
2. $1 \le r \le n$，$1 \le b_i \le m$；
3. 重新排列数列中的元素可以得到至少一个，最多 k 个好数列；

$n, m, k \le 100$

关键字：

动态规划

算法讨论：

一个好数列可以看成由 $\frac{r}{2}$ 段上升的和 $\frac{r}{2}$ 段下降的线段组成的(r 为奇数无解)，高度最多为 $m-1$，而只要确定了上升线段，则下降线段也确定了，所以可以把 n 先除以 2。

考虑当前已经填了 i 段，高度为 j，组成好数列的方案数为 l。枚举要填入的高度为 $j+1$ 的线段数量 t，我们要求现在好数列的方案数。对于一条高度为 j 的线段，它相当于一座峰，而这座峰可以通过填入若干条高度为 $j+1$ 的线段分裂成若干个峰，或直接不填。所以再记录上一个高度填的线段数量 k，当前状态组成好数列的方案数就是 $l \cdot \binom{t+k-1}{k-1}$。

时空复杂度：

时间：$O(n^3mk)$

空间：$O(n^2mk)$

Rotatable Number

Codeforces 303D

题目大意：

如果一个 n 位的 b 进制数 k，$k, 2k, 3k, \ldots, nk$ 形成了 k 各个数位的排列(可以有前导 0)，则称 k 为 b 进制下的循环数。现在给出 n 和 x，求 $1 < b < x$ 的一个最大的 b，使得 b 进制下 n 位的循环数存在，或判断无解。

$1 \leq n \leq 5 \times 10^6, 2 \leq x \leq 10^9$

关键字：

数论　原根

算法讨论：

把 k 看成一个无限循环小数的循环节，易知那个小数的分数形式是 $\frac{k}{b^n-1}$，显然它是一个单位分数 $\frac{1}{p}$，$2k, 3k, \ldots, nk$ 表示成分数就是 $\frac{2}{p}, \frac{3}{p}, \ldots, \frac{n}{p}$。模拟除法的过程可知，循环小数的循环节是由 $b^i \bmod p$ 决定的($b^n \equiv 1 \pmod p$)，所以如果要符合题意，就要求 $b^i (0 \leq i < n)$ 互不相同，且取遍 $[1, n]$。于是得到，$p = n + 1$，且 p 为质数，b 是 p 的一个原根。

于是先判 $n + 1$ 是不是质数，然后从大到小枚举 b，判断是不是 $n + 1$ 的原根即可。由于原根是在模 $n + 1$ 域下的，所以最多只要枚举 $n + 1$ 步。注意特判 b 是 $n + 1$ 的约数和 $n = 1$ 的情况。

时空复杂度：

时间：$O(n \log_2^2 n)$
空间：$O(1)$

Random Ranking

Codeforces 303E

题目大意：

有 n 个人，每个人的得分是区间 $[l_i, r_i]$ 之间的一个随机实数。把所有人按得分从小到大排序，求每个人处于各个位置的概率。

$n \leq 80$

关键字：

概率　动态规划

算法讨论：

暴力的方法是，对于一个人，枚举他的得分 x，然后对其余每个人求出得分比 x 低的概率与得分比 x 高的概率，然后这个人得分为 x 时处于第 j 位的概率就是在其余人里选 $j-1$ 个得分比 x 小的与 $n-1-j$ 个得分比 x 大的概率之和。最后这个人排名的总概率只要积分一下即可。

现在来优化上述算法。首先把所有区间的端点提出来排序，对于一个人，得分在相邻两个端点之间的一段区间的排名概率都是相同的，所以把这个区间 $[L, R]$ 提出来做。但这样不能直接套用上述算法，因为有可能其余人的得分也位于 $[L, R]$ 中。对于这种情况，这些人对排名在一段区间内的概率的贡献都是相同的。所以，分别记录每个人得分在 $[L, R]$ 左边、在 $[L, R]$ 内与在 $[L, R]$ 右边的概率，设 $f[i][j][k]$ 表示选了了 i 个人，得分在 $[L, R]$ 左边的有 j 个人，在 $[L, R]$ 内的有 k 个人的概率。

时空复杂度：

时间：$O(n^5)$，需要常数优化。

空间：$O(n^2)$

Olya and Graph

Codeforces 305D

题目大意：

一张 n 个点 m 条边的有向图，给定 k，可以添加一些边，使得该图满足：

1. 从点 i 出发，能到达点 $i+1, i+2, \ldots, n$；
2. 对于任意一条从 u 到 v 的边，$u < v$；
3. 两点之间最多一条边；
4. 对于一对点 i, j $(i < j)$，如果 $j - i \leq k$，从 i 到 j 的最短距离为 $j - i$；
5. 对于一对点 i, j $(i < j)$，如果 $j - i > k$，从 i 到 j 的最短距离为 $j - i$ 或 $j - i - k$；

$n, k \leq 10^6$，$m \leq 10^5$

关键字：

组合数学

算法讨论：

显然最后的图应该是：对于每个 $i (i < n)$，i 到 $i+1$ 一定有边，i 到 $i+k+1$ 可以有边，也可以没有边，但要保证对于任意两个有这条边的点 i, j，满足 $j - i \leq k$，除此之外没有其他边。

枚举最左边的有第二种边的点，把这些 2 的幂次加起来即可。

时空复杂度：

时间：$O(n+m)$

空间：$O(n+m)$

Playing with String

Codeforces 305E

题目大意：

有一个字符串 S，两个人轮流操作，每次可以选出一个位置 i，满足 $S[i-1]=S[i+1]$，然后把字符串分成 $S[1\dots i-1], S[i], S[i+1\dots|S|]$ 三部分，不能操作者输。

判断当两人都采取最优策略时，是先手胜还是后手胜，如果是先手胜还要输出先手第一步选择的位置。

$|S|\le 5000$

关键字：

博弈论

算法讨论：

整段字符串的 SG 值可以由若干段连续的合法位置的 SG 值异或得到。设 $\mathrm{SG}(i)$ 表示一段长度为 i 连续合法位置的 SG 值，则有：

$$\mathrm{SG}(i)=\mathrm{mex}\{\mathrm{SG}(j-2)\oplus\mathrm{SG}(i-j-1)\}$$

枚举第一步选择的位置，然后统计该次操作后字符串的 SG 值，如果为 0 则先手胜。

时空复杂度：

时间：$O(|S|^2)$
空间：$O(|S|)$

White, Black and White Again

Codeforces 306C

题目大意：

n 个位置，有 w 个不同的 W 和 b 个不同的 B，要求每个位置至少有一个字母，且只能放相同种类的字母；先要是若干(>0)个位置放 W，然后若干个位置放 B，然后再放 W。求方案数。

$n, b, w \leq 4000$

关键字：

组合数学

算法讨论：

先求 n 个位置放同种的 m 个字母的方案数，用隔板法可知，答案为 $\binom{m-1}{n-1} \cdot m!$。

枚举放 B 的位置数 i，则位置个数相同有 $n-i-1$ 种方案。然后 i 个位置放 B，剩下 $n-i$ 个位置放 W。最后答案为

$$\sum_{i=1}^{n-2}(n-i-1) \cdot b!w! \cdot \binom{b-1}{i-1} \cdot \binom{w-1}{n-i-1}$$

时空复杂度：

时间：$O(n^2)$

空间：$O(n^2)$

Polygon

Codeforces 306D

题目大意：

构造一个 n 边形，使得它所有的角相等，所有的边不等。

$n \leq 100$

关键字：

构造　计算几何

算法讨论：

首先构造一个正 n 边形，然后把每条边向外平行一段合适的距离即可。

时空复杂度：

时间：$O(n)$

空间：$O(n)$

Context Advertising

Codeforces 309B

题目大意：

有 N 个单词，选出其中连续的一段，把它们写在不超过 R 行，每行不超过 C 个字符的表里，相邻两个单词之间必须用一个空格或换行隔开，一个单词不能拆成两行。求最多能选出的单词数，输出方案。

$N, R, C \leq 10^6$，$R \times C \leq 10^6$，单词总长 $|S| \leq 5 \times 10^6$

关键字：

单调性　DFS

算法讨论：

首先可以用单调性求出从每个单词 i 开始不超过 C 个字符能到哪个单词，令这个单词为 i 的父亲，这样构成一棵树或森林。问题转化为，求树的每个节点与它第 R 个祖先的最大编号差。然后 DFS 一遍即可。

时空复杂度：

时间：$O(N + |S|)$
空间：$O(N + |S|)$

Tennis Rackets

Codeforces 309D

题目大意：

一个正三角形的每条边被 n 个点等分成了 $n+1$ 份，边上除了最靠近顶点的 $2m$ 个点都是可选点(共 $3(n-2m)$ 个可选点)。以这些可选点为顶点，求能构成的钝角三角形个数。

$n \leq 32000$

关键字：

枚举　几何

算法讨论：

首先钝角在哪条边上的答案都是相同的，所以可以固定钝角在底边上，最后答案乘 3。然后根据左右对称性，枚举钝角顶点时只需枚举一半。

先枚举钝角顶点，再枚举其余边上的顶点，另一条边上的顶点的可取范围是一个区间，且区间端点是单调的。如何判断是否构成钝角三角形可以用余弦定理。

时空复杂度：

时间：$O(n^2)$

空间：$O(1)$

Sheep

Codeforces 309E

题目大意：

有 n 个区间，要把这些区间安排一个顺序，使得所有相交的区间之间距离的最大值最小。输出任意一种方案。

$n \leq 2000$

关键字：

二分答案　贪心

算法讨论：

对于这种使最大值最小或使最小值最大的题目，首先想到可以用二分答案。

二分最大距离 D，现在的问题是，是否存在一种方案，使得所有相交的区间之间距离不超过 D。

对于 $1 \sim n$ 的每个位置，依次放区间。假设当前在位置 i 及以前都放好了一些区间，我们要求下一个位置 $i+1$ 应该放哪个区间更优。根据之前已放的区间，可以得到其余区间能够放的位置的限制，即不能和与它相交的区间距离超过 D。这样对每个区间都可以记录一个 p_j，表示它能放的最远位置。

然后可以求出，每个位置及之前至少要放的区间个数 s_k，即小于等于它的 p_j 的个数。如果对于某个位置 k，有 $s_k > k$，则显然已经无解了。

对于下一个位置要放的区间，首先要使右端点尽量小。因为这样剩余区间的约束就少了，它们的位置限制也会变得宽松。所以，下一个放的区间，应该是所有可行区间中右端点最小的。

如何判断放一个区间是否可行？我们找到第一个满足 $s_k = k$ 的位置 k，则可行区间 j 应该要满足 $p_j \leq k$。因为如果下一个位置放了一个 $p_j > k$ 的区间，则这会使得下一轮计算中 $s_k > k$，一定不合法。

所以，最终算法为：二分最大距离 D，第一个位置放右端点最小的区间，然后依次枚举每个位置，根据与当前区间是否相交更新 p_j，再根据 p_j

算出 s_k，然后找到第一个满足 $s_k = k$ 的位置 k，在所有满足 $p_j \leq k$ 的区间中选一个右端点最小的区间作为下一个位置要放的区间。

时空复杂度：

时间：$O(n^2 \log_2 n)$
空间：$O(n^2)$

Fetch the Treasure

Codeforces 311C

题目大意：

有 h 个格子，其中有 n 个格子有宝藏，第 i 个宝藏格的位置为 a_i，有宝藏 b_i。

开始从 1 号格子走，只有一种走法 k，即每次可以走 k 格。

现在有 m 个操作，操作有三种类型：

1. 加一种走法；
2. 把某个宝藏格的宝藏减少；
3. 询问当前能走到的宝藏格中，宝藏最多的是多少，并把它取走。

$n, m \leq 10^5$，$h \leq 10^{18}$，$k \leq 10^4$，操作 1 的次数 ≤ 20。

关键字：

最短路　堆

算法讨论：

把所有格子按模 k 分类，则同一类之间可以互相到达，且只要找出一类中最小的能由 1 到达的格子编号，这一类中编号比它大的格子都能到达。于是可以设 $d[i]$ 表示模 k 为 i 的格子中能由 1 到达的最小的格子编号，转移类似最短路。

由于操作 1 最多只有 20 个，所以每次加入一种走法后重新求一遍最短路也是可以的。剩下的就只要用堆维护所有能走到的格子即可。

时空复杂度：

时间：$O(m \log_2 n + 20 \cdot \text{ShortPath}(k, 20k))$

空间：$O(n + k)$

Biologist

Codeforces 311E

题目大意：

有 n 个数 a_i，每个数都是 0 或 1。可以把若干个数取反，但需要花费 v_i 的代价。有 m 个要求，每个要求给出 w, g, x, y，还有一个集合 S，如果集合 S 中的数的 a_i 都等于 x，则可以获得 w 的价值，否则要失去 $y \cdot g$ 的价值。求最多能获得的价值。

$n \leq 10^4$，$m \leq 2000$，$|S| \leq 10$

关键字：

网络流　最小割

算法讨论：

对每个数和每个要求都建一个点。对于每个数，如果 $a_i = 0$，源向该点连流量为 v_i 的边，否则该点向汇连流量为 v_i 的边；对于每个要求，如果 $x = 0$，源向该点连流量为 $w + y \cdot g$ 的边，否则该点向汇连流量为 $w + y \cdot g$ 的边；对于一个要求 i 和该要求集合中的一个数 j，如果该要求的 $x = 0$，i 向 j 连流量为无穷的边，否则j 向 i 连流量为无穷的边。答案为所有要求的 w 之和减去该网络的最小割(最大流)。

时空复杂度：

时间：$O(\text{MaxFlow}(n + m, n + \sum |S|))$
空间：$O(n + \sum |S|)$

Sereja and Squares

Codeforces 314E

题目大意：

一个括号序列，删掉了所有的右括号和若干左括号，求可能的合法括号序列个数。

$n \leq 10^5$

关键字：

动态规划

算法讨论：

设 $f[i][j]$ 表示做到了第 i 位，未匹配的左括号个数为 j 的方案数。根据当前位置是左括号还是未知括号来转移。需要常数优化。

时空复杂度：

时间：$O(n^2)$

空间：$O(n)$

PE Lesson

Codeforces 316D3

题目大意：

一个 n 的排列 a_i，初始 $a_i = i$，每次可以交换两个数，但要求第 i 个位置参与交换的次数不超过 b_i。求可能得到的排列数。

$n \leq 10^6$，$1 \leq b_i \leq 2$

关键字：

动态规划　组合数学

算法讨论：

考虑一种排列，由初始状态得到它每个点需要的最少交换次数。根据置换的循环节可以得到若干个环，显然是环上第 i 个点与第 $i+1$ 个点交换，这样有两个数参与交换的次数为 1，其余数参与交换的次数为 2。所以一个环中最多只有两个数参与交换的次数为 1。

设有 x 个 $b_i = 1$，y 个 $b_i = 2$。设 $f[i]$ 表示有 i 个只能交换 1 次的数组成若干个环的方案数。第 i 个点可以自己成一个环，或选择前面某个点与它组成环，于是

$$f[i] = f[i-1] + f[i-2] \cdot (i-1)$$

然后考虑把可以交换两次的数插入，方案数为 $\binom{x+y}{x} \cdot y! = \frac{(x+y)!}{x!}$，所以最后的答案为

$$f[x] \cdot \frac{(x+y)!}{x!}$$

时空复杂度：

时间：$O(n)$

空间：$O(n)$

Summer Homework

Codeforces 316E3

题目大意：

有 n 个数 a_i，有 m 个操作，操作有 3 种：

1. 给出 x, v，把 a_x 改成 v；
2. 给出 l, r，求
$$\sum_{i=1}^{r-l+1} f_i \cdot a_{l+i-1}$$
其中 $f_1 = f_2 = 1, f_i = f_{i-1} + fi - 2$。
3. 给出 l, r, d，把 l 到 r 这一段的 a_i 都加上 d。

$n, m \leq 200000$

关键字：

线段树

算法讨论：

f_i 即 Fibonacci 数列，它有一个性质：

$$f_{i+k} = f_{k+1} f_i + f_k f_{i-1}$$

所以，线段树的每个节点维护

$$x = \sum_{i=1}^{r-l+1} f_i \cdot a_{l+i-1}$$

和

$$y = \sum_{i=1}^{r-l+1} f_{i-1} \cdot a_{l+i-1}$$

合并时，就能算出

$$\sum_{i=1}^{r-l+1} f_{i+k} \cdot a_{l+i-1}$$

即

$$x' = x_1 + f_{k+1}x_2 + f_k y_2$$
$$y' = y_1 + f_k x_2 + f_{k-1}y_2$$

时空复杂度：

时间：$O((n+m)\log_2 n)$
空间：$O(n)$

Suns and Rays

Codeforces 316F3

题目大意：

给出一张 $H \times W$ 的图片，图片中有若干个椭圆(可以任意旋转)，每个椭圆边上连出去若干条射线，组成一个“太阳”。要求识别出有一个几个“太阳”，和每个“太阳”边上的射线数。“太阳”之间不会相交。

$H, W \leq 1600$，射线宽度 ≤ 3，长度 $\in [10, 30]$

关键字：

图像识别

算法讨论：

首先识别出所有“太阳”，即所有连通块。

对于一个“太阳”，考虑把它周围的射线提取出来，其中的连通块数即射线数。可以设一个参数 lim_1，如果同一行中有一段宽度 $\leq lim_1$，则它很有可能是射线，然后对于列和对角线也做一遍。判断一个很有可能是射线的连通块是不是射线时，再设一个参数 lim_2，当连通块大小 $\geq lim_2$ 时就把它识别成射线。

时空复杂度：

时间：$O(HW)$

空间：$O(HW)$

Good Substrings

Codeforces 316G3

题目大意：

一个串 S，还有 N 个规则串 T_i，每个规则串还有一个区间 $[l_i, r_i]$。求 S 有多少个子串，对每个规则 i 都满足它在 T_i 中出现的次数在 $[l_i, r_i]$ 内。

$|S|, |T_i| \leq 50000$，$N \leq 10$

关键字：

后缀自动机

算法讨论：

把原串 S 和 N 个规则串拼在一起，中间用奇怪的字符隔开，然后建后缀自动机。对后缀自动机中的每个节点维护它的子树中每种类型(原来属于哪个串)的节点数。统计答案时，每个节点的子树中某种类型的节点数就是该节点对应的子串集合在该类型的规则串中出现的次数，如果对 N 个规则都满足要求就把该节点对应的子串集合大小加入答案中。

时空复杂度：

时间：$O(N \cdot (|S| + \sum |T_i|))$

空间：$O((26 + N) \cdot (|S| + \sum |T_i|))$

Balance

Codeforces 317C

题目大意：

有 N 个容器，每个容器有初始水量 a_i 和期望水量 b_i。有 M 条双向管道分别连接着两个容器，可以通过管道来减少一个容器的水量并增加另一个容器的水量。管道中的水量必须是整数，同时任意时刻任意容器的水量不能超过 V。构造一种方案，使得满足要求，步骤数不能超过 $2N^2$；或判断无解。

$N \leq 300$

关键字：

构造　DFS

算法讨论：

首先判无解，即存在一个连通块，初始水量之和不等于期望水量之和。

然后不断寻找点对 x, y，使得 $a_x > b_x$ 且 $a_y < b_y$，这样只要从 x 向 y 流 $\min(a_x - b_x, b_y - a_y)$ 的水，并且维护其他容器的水量不变，就会至少增加一个满足要求的容器。

对于上述操作，首先求一条 x 到 y 的路径 p_i，然后定义过程 $\text{flow}(i, d)$ 表示从 x 向 y 流 d 的水，当前做到路径上的 i 点。首先如果 $a_{i+1} \geq d$，则直接 $\text{flow}(i+1, d)$，然后 i 向 $i+1$ 流 d；否则先从 i 向 $i+1$ 流 $d - a_{i+1}$，使得 $a_{i+1} = d$，然后 $\text{flow}(i+1, d)$，最后 i 向 $i+1$ 流完剩余的流量。

由于最多做 N 次流操作，每次操作的步骤数最多为路径长度 $\times 2$，所以总步骤数不会超过 $2N^2$。

时空复杂度：

时间：$O(N^3)$

空间：$O(N^2)$

Princess and Her Shadow

Codeforces 317E

题目大意：

在一个无限大的网格上有两个格子 $V(v_x, v_y)$ 和 $S(s_x, s_y)$，还有 m 个障碍。V 要去追 S，每个时刻，V 可以沿四个方向中的一个移动一格，但 S 也会沿相同方向移动一格，除非那个格子是障碍。要求构造一种 V 移动的方案，使得最后 V 与 S 在同一格子中。

$m \leq 400$，所有坐标绝对值 ≤ 100

关键字：

构造

算法讨论：

显然，如果一个障碍都没有，或 V 与 S 不连通，则无解。

考虑先把两个点移到很远的地方。求出 V 到 S 的最短路径，然后 V 沿着最短路径走，如果 S 也跟着移动了，则把 S 当前的位置加入到最短路径序列中，直到 V 追上 S 或 V 到了很远的地方(所有障碍的最小外界矩形的外部)。

现在的目标是让 V 与 S 的横坐标和纵坐标相同，如果不相同，就要利用障碍了。比如，现在 $v_y > s_y$，首先把它们都移到很上面(让纵坐标很大)，然后左右移动使得 s_x 与纵坐标最大的那个障碍的横坐标相同，然后一直向下移动，S 就会被那个障碍卡住，直到 V 与 S 的纵坐标相同。用同样的方法，处理 $v_y < s_y$，$v_x > s_x$，$v_x < s_x$ 的情况即可。

时空复杂度：

时间：$O(200^2)$

空间：$O(200^2)$

Have You Ever Heard About the Word?

Codeforces 319D

题目大意：

如果有一个字符串是由两个或以上的相同串连接而成，则称这个字符串为重复串。

给出一个长度为 n 的字符串，每次找到它长度最短的重复子串(如果有多个找最左边的那个)，把这个形如 XX 的重复子串替换成 X，直到没有重复子串为止。求最后形成的串。

$n \leq 50000$

关键字：

字符串　Hash　二分答案

算法讨论：

从小到大枚举长度 l，首先判断是否存在由长度为 l 的串重复若干次的子串，如果没有就不做了；否则，最多只会存在 $O(\sqrt{n})$ 种可删的长度(因为 $1+2+\cdots+\sqrt{n}=O(n)$)，然后就可以暴力删除了。

对于如何判断两个子串是否相等，可以用 Hash，删完一种长度后把 Hash 数组重构一遍。

对于如何判断原串是否存在由长度为 l 的串重复若干次后的子串，把原串每 l 个位置分割一下，则重复子串一定恰好跨越了两个分割点，对于相邻两个分割点求一下最长公共前缀和最长公共后缀即可。求最长公共前(后)缀，可以用二分答案 + Hash。

时空复杂度：

时间：$O(n\sqrt{n}+n\log^2 n)$

空间：$O(n)$

Ping-Pong

Codeforces 319E

题目大意：

如果区间从 $[a,b]$ 到 $[c,d]$ 之间有一条有向边，当且仅当 $c<a<d$ 或 $c<b<d$。有 n 个操作，每次可以加入一个区间 $[l,r]$，或询问是否存在一条从第 x 个加入的区间到第 y 个加入的区间的路径。数据保证加入的区间长度严格递增。

$n \leq 10^5$

关键字：

线段树　并查集

算法讨论：

如果两个区间相交，则它们可以互相到达，它们之间的边可以看成无向边。这样一些相交的区间就构成若干个连通块，且如果区间 a 包含区间 b，则区间 a 所在连通块中的所有区间也包含区间 b 所在连通块中的所有区间。而这些相交的区间本身也可以并成一个区间，所以对于询问 x,y 只要判断它们是否同在一个连通块或 y 的连通块的并集是否包含 x 的连通块中的某个区间即可。

对于插入一个区间 x，由于区间长度是严格递增的，所以与 x 相交的区间就是所有包含 x 的左端点或包含 x 的右端点的区间。在线段树的每个节点上挂一张表，表示该节点代表的区间(线段)都被表中的区间连通块覆盖。然后沿着线段树走到 x 的左端点和右端点，沿途把表中的连通块合并，并更新当前区间 x 所在的连通块的并集。最后把 x 所在的连通块的并集插入线段树。

由于题目中区间端点的范围是 $[-10^9,10^9]$，需要离散。但也可以不离散，直接用原区间在线段树中操作，但这样空间复杂度是 $O(n\log_2 n)$ 的。

时空复杂度：

时间：$O(n\alpha(n)\log_2 n)$

空间：$O(n\log_2 n)$

Ciel and Flipboard

Codeforces 321D

题目大意：

一个 $n \times n$ 的矩阵 $a_{i,j}$，令 $x = \frac{n+1}{2}(n$ 为奇数$)$，每次可以选一个 $x \times x$ 的子矩阵，把子矩阵中的所有数取相反数，可以操作任意次。求可能得到的最大的矩阵和。

$n \leq 33$

关键字：

枚举　贪心

算法讨论：

设 $b_{i,j}$ 表示每个元素前的系数$(1$ 或 $-1)$。由于第 x 列正好位于矩阵的中间，所以每次操作时，对于每一行 i 和所有小于 x 的列 j，$b_{i,j}, b_{i,j+x}, b_{i,x}$ 这三个元素中要么没有数被取反，要么有且仅有两个数被取反，所以就有 $b_{i,j} \cdot b_{i,j+x} \cdot b_{i,x} = 1$。同理，对于每一列 j 和所有小于 x 的行 i，有 $b_{i,j} \cdot b_{i+x,j} \cdot b_{x,j} = 1$。

于是，只要确定了 $i \leq x$ 且 $j \leq x$ 的 $b_{i,j}$，剩下的也就确定了。枚举 $b_{x,1}, b_{x,2}, \ldots, b_{x,x}$，然后依次确定第 $1, 2, \ldots, x-1$ 行的 $b_{i,j}$。对于第 i 行，先枚举 $b_{i,x}$ 是什么，然后对于 $j < x$，$b_{i,j}, b_{i,j+M}, b_{i+M,j}, b_{i+M,j+M}$ 这四个元素的值只与 $b_{i,j}$ 有关，再枚举 $b_{i,j}$ 取个最大值即可。

时空复杂度：

时间：$O(n^2 \cdot 2^n)$

空间：$O(n^2)$

Tournament-graph

Codeforces 323B

题目大意：

构造一个点数为 n 的竞赛图(有向完全图)，使得任意两点的最短距离不超过 2，或判断无解。

$3 \leq n \leq 1000$

关键字：

构造

算法讨论：

使用增量法，假设已经构造好了 $n = k - 2$ 的图，要构造 $n = k$ 的图。设新加入的点为 $k-1, k$，则 $k-1$ 向 k 连边，然后对每个 $i(1 \leq i < k-1)$，k 向 i 连边，i 向 $k-1$ 连边，这样对新加入的两个点都能满足，所以所有的点都能满足。注意 $n = 4$ 是无解的，但对于其他偶数依然有解，所以要手动构造 $n = 6$ 的情况。

时空复杂度：

时间：$O(n^2)$

空间：$O(n^2)$

Two permutations

Codeforces 323C

题目大意：

有两个 N 的排列 a_i, b_i，有 M 个询问，每次给出两个区间 $[l_1, r_1], [l_2, r_2]$，求在 a_i 中的位置在 $[l_1, r_1]$ 内，在 b_i 中的位置在 $[l_2, r_2]$ 内的数的数量。强制在线。

$n \leq 10^6$，$m \leq 2 \times 10^5$

关键字：

函数式线段树

算法讨论：

对每个数 k 定义坐标 (x_k, y_k)，其中 x_k 为 k 在 a_i 中的位置，其中 y_k 为 k 在 b_i 中的位置。于是原问题就相当于一个二维数点问题，使用函数式线段树解决。

时空复杂度：

时间：$O((n+m)\log_2 n)$

空间：$O(n \log_2 n)$

Monsters and Diamonds

Codeforces 325C

题目大意：

有 n 个物品，m 种分裂规则。每种分裂规则给出 p, x 和一个物品集合 S，表示物品 p 能分裂成集合 S 中的物品和 x 个钻石。现在从每个物品开始，经过若干次分裂(如果一种物品有多种分裂规则就任选一种)，直到不能再分裂为止，求出最少和最多能得到多少钻石(可能是无穷大)，或者判断根本无法分裂完。

$n, m \leq 10^5$，$x \geq 1$

关键字：

最短路　DFS

算法讨论：

先来求第一问。设 f_i 表示物品 i 最少能分裂出的钻石，对每种规则 i，维护 c_i 表示该规则中未被确定最小值的物品个数，s_i 表示该规则中所有物品的 f 之和。类似 Dijkstrs 算法，用优先队列维护已确定最小值的物品。首先对于所有只能分裂出钻石的规则，令它们对应的物品的 f_i 为该规则的钻石数(如果一种物品有多种只能分裂数钻石的规则就取钻石数最少的)，并把该物品加入队列。然后每次取出队列中 f 值最小的一个物品，记为 x，由于此时 x 已经被确定最小值了，就枚举所有能分裂出该物品的规则 i，把 c_i 减去 1，把 s_i 加上 f_x，如果 $c_i = 0$ 了，就把该规则对应的物品加入队列中。最后，如果某种物品仍未被确定最小值，它就无法分裂完，否则 f_i 就是第一问的答案。

然后求第二问。从每个可以分裂完的物品开始 DFS，枚举它的一个分裂规则(要求该规则中的所有物品都可以分裂完)，递归计算该规则中所有物品的最大值。如果中途出现环了(递归到一个还在栈中的物品)，最大值就是无穷大。

时空复杂度：

时间：$O(n \log_2 m + m)$

空间：$O(n + m)$

Reclamation

Codeforces 325D

题目大意：

一个 $r \times c$ 的方格，其中第 1 和第 c 列是连着的。有 n 个障碍，每次依次放一个障碍，然后判断是否从第 1 行到第 r 行有一条四连通的路径，如果没有则不放这个障碍。求最后总共成功放了多少个障碍。

$r, c \leq 3000, n \leq 3 \times 10^5$

关键字：

并查集

算法讨论：

首先把环拆开，即把方格列数扩到两倍，格子 (x, y) 对应格子 $(x, y + c)$。从第 1 行到第 r 行没有四连通的路径等价于存在一个八连通的由障碍组成的左右向的环路径，把方格分成了上下两部分，也等价于存在一个障碍格 (x, y)，有从 (x, y) 到 $(x, y + c)$ 的八连通路径。

然后用并查集做。判断该障碍是否合法时，枚举该障碍附近的 8 个格子和该障碍对应障碍附近的 8 个格子，如果它们中有一对在同一个连通块中，即存在这样的八连通路径，该障碍不合法。否则该障碍合法，把它和它周围 8 个格子并到同一连通块中。

时空复杂度：

时间：$O(8^2 n\alpha(n))$

空间：$O(rc)$

The Red Button

Codeforces 325E

题目大意：

有 n 个点，编号分别是 $0, 1, 2, \ldots, n-1$。点 i 点与点 $2i \bmod n$ 和点 $(2i+1) \bmod n$ 之间有一条有向边。求这张图的一个哈密顿回路。

$n \leq 10^5$

关键字：

并查集

算法讨论：

首先可以发现，如果 n 是奇数则无解。

对于 n 是偶数，点 i 和点 $i + \frac{n}{2}$ $(0 \leq i < \frac{n}{2})$ 都与点 $2i \bmod n$ 和点 $(2i+1) \bmod n$ 有边，且如果最后的方案是从点 i 走到点 $2i \bmod n$，点 $i + \frac{n}{2}$ 就只能走到点 $(2i+1) \bmod n$ (反之亦然)。于是，先任意规定点 i 走到哪里，点 $i + \frac{n}{2}$ 走到哪里也确定了，且每个点都有一个出度和入度。

这样最后的方案会形成若干个环，现在要做的是把这些环拼成一个。可以发现，如果形成了环，就必有一对 i 和 $i + \frac{n}{2}$ 不在同一连通块里，这时只要交换它们两个点的出边就行了。用并查集实现。

时空复杂度：

时间：$O(n\alpha(n))$

空间：$O(n)$

The Evil Temple and the Moving Rocks

Codeforces 329D

题目大意：

在一个 $n \times n$ 的房间里，可以放置有 4 种石块，分别只能向四个方向中的某一个移动。开始可以激活其中的一个石块，激活的石块会沿着它的方向移动，直到碰到下一个石块或边界，如果碰到的是石块，就会激活它，否则直接结束。如果石块激活次数达到 10^7 也会直接结束。如果石块至少被移动了一个位置，就会发生一次碰撞。构造一种石块的放置方法，使得碰撞次数 $\geq x$。

$n = 100, x = 10^5$

关键字：

构造

算法讨论：

主要思想是，制造若干轮循环，每轮循环都能把整个房间遍历一遍。

对于奇数行，前一半是形如 >>>>>>>> 的，后一半是形如 >.>.>.>. 的；对于偶数行，前一半是形如 .<.<.<.< 的，后一半是形如 <<<<<<<< 的。然后把相邻两行连起来，最后把第一行与最后一行连起来即可。

时空复杂度：

时间：$O(n^2)$

空间：$O(n^2)$

The Great Julya Calendar

Codeforces 331C3

题目大意：

一个数 n，每次可以减去它数位中的一个数字，求让它变成 0 需要的最少步数。

$n \leq 10^{18}$

关键字：

贪心　动态规划

算法讨论：

首先显然每次减去的是它数位中最大的一个数字。

设 $f[i][j]$ 表示数字 i，除了 i 的数位外最大的数位为 j，把 i 减到 ≤ 0 的最少步数。由于 j 的关系，可能会把 i 减到 0 以下，就再记 $g[i][j]$ 表示多减了多少。转移时，找到最大的一个 10 的幂次数 p 满足 $p \leq i$，则 $f[i][j]$ 可以由 $f[i'][j']$ 转移而来，其中 $i' = i \bmod p$，$j' = \max(j, \lfloor i/p \rfloor)$。然后把 i 减去 $(i \bmod p) + g[i'][j']$，但如果 $g[i'][j'] = 0$，就把 i 减去 $\max(j, \lfloor i/p \rfloor)$。

现在来看状态数。显然 j 的范围为 $[0, 9]$；对于 i，可以发现，到后来 i 都是形如 $999\cdots9\mathrm{X}$ 的，所以 i 的的数量为 $O(10 \log_{10} n)$，所以总状态数为 $O(10^2 \log_{10} n)$。再加上转移的复杂度，所以总时间复杂度为 $O(10^3 \log_{10} n)$。

时空复杂度：

时间：$O(10^3 \log_{10} n)$

空间：$O(10^2 \log_{10} n)$

Theft of Blueprints

Codeforces 332D

题目大意：

n 个点的无向图，每条边有个边权。任选其中的 k 个点，设与它们都有边相连的点为 S(保证对于任意的点集只有一个 S)。求出 S 到这 k 个点的之间的边的边权和，求边权和的期望。

$n \leq 2000$

关键字：

组合数学　期望

算法讨论：

枚举每个点i，如果它的度数 $d_i \geq k$，则它可以作为 S，与它相连的任意 k 个点都可以作为一种方案。对于与 i 相连的每条边，都有 $\binom{d_i-1}{k-1}$ 种方案中有它。所以只需预处理每个点的度数和相邻边权和即可。

时空复杂度：

时间：$O(n^2)$
空间：$O(n^2)$

Binary Key

Codeforces 332E

题目大意：

给定一个长度为 N 的字符串 A 和一个 01 串T，把 01 串无限倍长，让后把所有满足 $T_i = 1$ 的 A_i 取出，组成一个新串 B。设 B 的长度为 M，现在给出串 A, B 和 T 的长度 K，求字典序最小的 T。

$N \leq 10^6, M \leq 200, K \leq 2000$

关键字：

枚举　贪心

算法讨论：

先枚举 T 中 1 的个数 l，然后把 B 每 l 个分成一段，把 A 每 K 个分成一段，判断对 B 中一段的每个字符是否在 A 的对应段中存在匹配字符即可。

由于还要字典序最小，当 1 的个数固定时，要把 1 尽量放后面，所以找匹配字符时要倒着枚举。

T 中 1 的个数为最多 M，匹配时要做 K 次比较，每次比较最多要 $\lfloor \frac{M}{i} \rfloor$ 次，所以时间复杂度最多 $O(KM \log M)$。

时空复杂度：

时间：$O(KM \log M + N)$

空间：$O(N)$

Lucky Tickets

Codeforces 333C

题目大意：

对于一个 8 位的数字串，如果可以通过在字符之间添加“+”，“−”，“×”运算符和括号使得最后的计算结果为 k，则这个数字串是合法的。输出任意 m 个合法的数字串。

$k \leq 10^4$，$m \leq 3 \times 10^5$

关键字：

构造　搜索

算法讨论：

把 8 位的数字串分成两个 4 位的数字串，枚举一个 i $(0 \leq i \leq 9999)$，然后搜索能构成 $k - i$ 的 4 位数字串，则 i 与该数字串拼成的 8 位数字串即一种合法方案。这样很快就会有 m 种方案了。

时空复杂度：

时间：$O(m)$
空间：$O(m)$

Rectangles and Square

Codeforces 335D

题目大意：

平面上有 N 个互不重叠的矩形，从中选出若干个矩形，使得它们恰好组成一个正方形，求出任意一组解。

$N \leq 10^5$，矩形坐标 ≤ 3000

关键字：

枚举　前缀和

算法讨论：

枚举矩形的坐下角作为正方形的左下角，然后再枚举正方形边长，判断是否合法。

首先判断这个正方形中被覆盖的格子数是否等于正方形面积。这个只要初始时把每个矩形覆盖一遍，求个二维前缀和就行了。复杂度为所有矩形的面积之和。

然后判断正方形的边界是否穿过一个矩形。对每条横的和竖的坐标线，预处理线上每一段是否包含矩形边界，然后就能判断正方形的一条边上是否有空隙了。

时空复杂度：

时间：$O(3000^2 + N)$

空间：$O(3000^2 + N)$

GCD Table

Codeforces 338D

题目大意：

一个 $N \times M$ 的矩阵 $A_{i,j} = \gcd(i, j)$，给出 K 个数 $a_1, a_2, \ldots, a_K$，判断这 K 个数是否在矩阵的某一行中连续出现。

$N, M, a_i \leq 10^{12}$, $K \leq 10000$

关键字：

数论　中国剩余定理　扩展欧几里得

算法讨论：

如果出现过，不妨设出现的位置为 (X, Y) (a_1 在第 Y 列)。首先可以证明，X 一定为这 K 个数的最小公倍数：设这 K 个数的最小公倍数为 i，显然有 $i|X$。假设 $X = ki$ $(k > 1)$，对于第 ki 行中的某一列 j $(Y \leq j \leq Y + K - 1)$，如果 $A_{i,j} \neq A_{ki,j}$，即存在一个 j 的约数 p 满足 $p \nmid i$ 且 $p|ki$，即 $p|\gcd(ki, j)$，即 p 是这 K 个数中某一个的约数，但这与 i 是这 K 个数的最小公倍数矛盾。所以，$k = 1$，即 $X = i$。

然后对于出现的列 Y，可以根据 $\gcd(X, i + Y - 1) = a_i$ 列出 K 个同余方程(组)

$$i + Y - 1 \equiv 0 \pmod{a_i} \quad (1 \leq i \leq K)$$

根据中国剩余定理，它的所有解是模 X 同余的，不过因为有 $\gcd(X, i) = \gcd(X, i + kX)$，所以只要求出其最小正整数解，然后判断即可。

由于 a_i 不是两两互质，不能用普通的中国剩余定理做。考虑只有两个方程(组)

$$x \equiv a_1 \pmod{m_1}$$

$$x \equiv a_2 \pmod{m_2}$$

设 $x = k_1 m_1 + a_1 = k_2 m_2 + a_2$，则有

$$k_1 m_1 - k_2 m_2 = a_2 - a_1$$

用扩展欧几里得算法解出 k_1, k_2，并得到 k_1 的最小非负整数解，由此可以得到 x 的最小非负整数解

$$x^* = k_1 m_1 + a_1$$

所有解模 $\text{lcm}(m_1, m_2)$ 同余。于是，我们得到了一个新方程

$$x \equiv x^* \pmod{\text{lcm}(m_1, m_2)}$$

此时加入下一个方程

$$x \equiv a_3 \pmod{m_3}$$

构成两个方程(组)，再用上述方法求解即可。

时空复杂度：

时间：$O(K \log_2 N)$
空间：$O(K)$

Optimize!

Codeforces 338E

题目大意：

一个长为 N 的序列 a_i 和长为 M 的序列 b_i，判断有多少 a 的长为 M 的连续子序列 c_i，使得存在一个 M 的排列 p_i，满足 $c_{p_i}+b_i \geq H$。

$M \leq N \leq 150000$

关键字：

线段树

算法讨论：

把 b_i 变成 $H-b_i$，再从小到大排序，对每个 a_i，二分找出最后一个大于等于 b 的位置 p_i，则 a_i 只能与 $k \in [1,p_i]$ 的 b_k 匹配。对于一个长为 M 的 a 的连续子序列，要判断它是否可以与 b 匹配，记 s_{p_i} 为小于等于 p_i 的 p_k 的个数，则对每个 i 要满足 $s_{p_i} \leq p_i$。对于 a 的每个连续子序列，依次处理，每次插入 a_i，删除 a_{i-M}，并维护这段连续子序列的 s_i。当插入或删除 a_i 时，只会对 $k \in [a_i,M]$ 的 s_k 有影响，即区间增减。判断是否对每个 i 有 $s_i \leq i$，即 $s_i - i \leq 0$。式中那个 $-i$ 与之后的改变无关，可以一开始就加上，于是就是判断整段的最大值是否 ≤ 0。于是就可以用线段树维护了。

时空复杂度：

时间：$O(N \log_2 M)$

空间：$O(N+M)$

Three Swaps

Codeforces 339E

题目大意：

一个 n 的排列，选择不超过 3 个区间，依次把这三个区间中的数翻转。给出操作后的排列，要求还原这些区间。

$n \leq 1000$

关键字：

搜索

算法讨论：

由于最多只会分成 7 段，所以直接搜索就行了。

时空复杂度：

时间：$O(7^6 + n)$

空间：$O(n)$

Candies Game

Codeforces 341E

题目大意：

有 N 个数 a_i，每次选两个数 $a_i, a_j (a_i \leq a_j)$，把 a_j 变成 $a_j - a_i$，把 a_i 变成 $2a_i$，要求最后只剩下两个数不为 0，求一种可行的操作序列。

$N \leq 1000, \sum a_i \leq 1000000$

关键字：

构造

算法讨论：

先考虑三个数 A, B, C $(A \leq B \leq C)$。如果我们可以通过一些操作使得最小的数变小，那么最后一定能把一个数变成 0，这样经过若干次后就会只剩两个非零数了。下面一种算法，每次可以让 B 变成 $B \bmod A$:

令 $X = \lfloor \frac{B}{A} \rfloor$，$Y = B \bmod A$，则 $B = AX + Y$。由于每次操作都是把小的数翻倍，大的数减去小的数，为了方便，把 X 表示成二进制。每次把 A 翻倍，并依此处理 X 的每一位。第 i 次时，A 变成了 $A \cdot 2^i$，如果 X 的第 i 位为 1，就把 B 减去 $A \cdot 2^i$，这样 X 的第 i 位就被消掉了；如果 X 的第 i 位为 0，就从 C 中减。这样最终 X 会变成 0，B 也变成 Y 了，且始终能保持 $A \leq B \leq C$，不会出现不够减的情况。

时空复杂度：

时间：$O(N \log^2 a_i)$

空间：$O(N)$

Xenia and Dominoes

Codeforces 342D

题目大意：

一个 $3 \times n$ 的网格，有些格子是障碍，还有一个特殊格。要在剩下的格子中不重叠地填满 1×2 的多米诺骨牌，使得至少有一块骨牌短的那条边与特殊格相邻。

$n \leq 10000$

关键字：

连通性状态压缩动态规划

算法讨论：

显然的连通性状态压缩动态规划，用二进制记录轮廓线上骨牌的状态，再记一维表示是否已经有一块符合要求的骨牌了。

时空复杂度：

时间：$O(2^4 n)$

空间：$O(n)$

Pumping Stations

Codeforces 343E

题目大意：

一张 N 个点，M 条边的无向图，每条边有个容量。求一个顶点的排列 v_i，最大化 $Flow(v_1, v_2) + Flow(v_2, v_3) + \cdots + Flow(v_{N-1}, v_N)$，其中 $Flow(s, t)$ 表示点 s 到点 t 的最大流。

$N \leq 200$, $M \leq 1000$

关键字：

分治　网络流　最小割

算法讨论：

最大流即最小割。首先用下面的的方法，用 Gomory-Hu 树求出两两点对之间的最小割：先在点集中随便找两个点 s, t，求出 s, t 之间的最小割，该最小割把原点集分成两个集合，在它们之间加一条边，边权为最小割的值，然后递归处理两个点集。这样最后会形成一条链，对于原图任意两点 u, v，u, v 之间的最小割即为树中对应两点之间所有边的边权的最小值。

然后用类似的方法，再求一棵 Gomory-Hu 树用来求答案：先在点集中随便找一个点 s，而 t 要满足 $Flow(s, t)$ 最小，然后再用上述方法递归即可。现在构出的链即最优方案。可以证明不会有比这种方案更优的方案了。

时空复杂度：

时间：$O(N \cdot \text{MaxFlow}(N, M))$

空间：$O(N + M)$

Doodle Jump

Codeforces 346E

题目大意：

给出 N, A, P, H，数列 $a_n = A \cdot n \mod P \quad (0 \le n \le N)$。把数列 a 从小到大排序，判断排序后的数列是否满足相邻两项的差都小于等于 H。

$A, H \le 10^9$，$N < P \le 10^9$，$\gcd(A, P) = 1$，数据组数 $T \le 10000$

关键字：

数论

算法讨论：

让我们来求数列 a 的最大间隔。以 $A = 5, P = 23$ 为例：

0	5	10	15	20
2	7	12	17	22
4	9	14	19	
1	6	11	16	21
3	8	13	18	

可以发现，数列 a 被分成了很多列，每一列间模 A 相等，每一行中模 A 也相等，且每一列中的数都处在连续的一段区间中，而区间的分界点就是第一列中的数。当取数列的前 N 项时，最大间隔一定出现在某一列，且可以证明，取其他列一定不会比取倒数第二列更优。于是只要求倒数第二列中的最大间隔即可，问题变成了一个规模更小的子问题 $(P' = A)$。

下面证明，每一列中的数也是一个具有通项 $a'_n = A' \cdot n \mod P'$ 的数列：在模 $P' = A$ 下，每列都相等，于是只需考虑第一列。设第一列第 i 行的数为 b_i，对于第 $i-1$ 行，易得该行的最后一个数 $x = P - (P - b_{i-1}) \mod A$，于是 $b_i = x + A - P = (A - P + b_{i-1}) \mod A$，得通项 $b_i = (A - P \mod A) \cdot n \mod A$。这样 $A' = A - P \mod A$。

最后是 N'。大概地，$N' = \lfloor \frac{A \cdot N}{P} \rfloor$。但当原数列第 N 项不在最后两列时，N' 要减 1。

于是，$\text{work}(A, N, P) \Rightarrow \text{work}(A - P \mod A, N', A)$，当只剩一行时直接返回答案退出。不过还有一个问题，如果 $P = A + 1$，$A' = A - P \mod A = A - 1$，每次只减少 1，不能保证复杂度。可以发现，$A' = P \mod A$ 与 $A' = A - P \mod A$ 是等价的，可以看做把数列 a' 都取了相反数。于是只要令 $A' = \min(P \mod A, A - P \mod A)$，就保证了 $A' \leq \frac{A}{2}$。

时空复杂度：

时间：$O(T \log_2 N)$

空间：$O(1)$

Pilgrims

Codeforces 348E

题目大意：

N 个节点的树，有 M 个点是特殊点。对于每个特殊点，所有与它距离最远的特殊点组成一个列表。现在要在树中找一点，使得去掉这个点和与它相连的边后，不能到它列表中的任何一个点的特殊点数最多，并求方案数。

$M < N \leq 100000$

关键字：

树形动态规划

算法讨论：

显然，如果去掉的点处在一个特殊点到它列表中的所有点的路径的交路径上，就能满足要求。于是现在的问题是如何求出每个特殊点的交路径。

以每特殊个点为根建树，它的交路径就是它到它列表中的所有点的最近公共祖先的路径，于是关键就是找到那个最近公共祖先点。实际做时，以任意点为根建树，然后用树形动态规划，先考虑列表中的点在当前特殊点的子树中，记录每个点向下能到的最远特殊点的距离，并维护所有最远特殊点的 LCA。然后考虑列表中的点在当前特殊点向上的路径上，再做一遍类似的树形 DP。然后就能得到每个特殊点的 LCA 点。

计算答案时，把每个特殊点到它的 LCA 点的路径上的点都覆盖一遍，统计最大覆盖数即可。这显然可以用树链剖分等算法做，但其实不用。把每条路径根据两端点的 LCA 拆成两条深度单调递增的路径，用类似前缀和的方法去覆盖，统计时求一遍子树和即可。只不过这种方法还需在树形动态规划时维护每个特殊点和它的 LCA 点的 LCA。

时空复杂度：

时间：$O(N)$

空间：$O(N)$

Jeff and Removing Periods

Codeforces 351D

题目大意：

一个长为 N 的序列 a_i，每次可以依次如下进行三个操作：

1. 任选出一个长度任意的等差数列 b_i，但要满足 a_{b_i} 都相同；
2. 把 a 中 b_i 位上的数全部删除；
3. 把剩下的数任意排列。

有 M 个询问，每次询问把一个区间中的数全部删除所需要的最少操作次数。

$N, M, a_i \leq 100000$

关键字：

离线　树状数组

算法讨论：

首先，如果把一段区间中的数重排后，答案显然就是这个区间中的数字种数。于是现在的问题是，如何判断在第一次不重排的情况下是否能消去一种数字，如果不能答案要加 1。

统计一个区间中的数字种数，可以用离线 + 树状数组，即先预处理每个位置之前第一个与它相同的数字位置 p_i，然后把询问区间按右端点排序，扫描这个序列，把 $[p_i + 1, i]$ 这段区间整体加 1，如果当前位置是一个区间的右端点，则这个区间的答案就是它左端点的覆盖次数。

判断区间中是否有一种数字对应的编号成等差数列，即一定不能成等差数列的数字种数等于区间数字种数。对于统计一定不能成等差数列的数字种数，也用与上面类似的方法，如果 $i - p_i \neq p_i - p_{p_i}$，则该种数字在左端点在 p_{p_i} 及以前的区间里不能成等差数列。

时空复杂度：

时间：$O(N \log_2 N)$

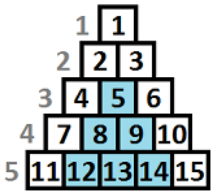
空间：$O(N)$

Transferring Pyramid

Codeforces 354D

题目大意：

如图的 N 阶数字三角形，其中有 K 个位置是特殊位置。每次可以把任意一个位置覆盖，代价为 3；或把任意一个与原三角形底边相邻的子三角形覆盖，代价为子三角形的大小 $+2$。求把 K 个特殊位置至少覆盖一遍的最小代价。

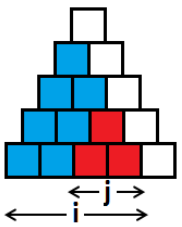


$N, K \leq 100000$

关键字：

动态规划

算法讨论：



如图所示，设 $f[i][j]$ 表示蓝色区域中的特殊位置都被覆盖的最小代价，枚举红色三角形的边长，有转移

$$f[i][0] = \min_{0 \leq j \leq i} \left(f[i-1][j-1] + \frac{j \cdot (j+1)}{2} + g[i][j] \times 3 + 2 \right)$$

$$f[i][j] = \min(f[i][j-1], f[i-1][j-1] + g[i][j] \times 3)$$

其中 $g[i][j]$ 表示第 i 列除了最底下 j 个位置外其余区域的特殊格子数。

上述方程是 $O(N^2)$ 的，但可以发现有用的 j 是很少的。当 $j > \sqrt{6K}$ 时，即第二种覆盖选择的三角形边长大于 $\sqrt{6K}$，所需的代价约为 $3K$，还不如每个特殊位置都进行第一种覆盖 $(3K)$。于是方程第二维只需 $\sqrt{6K}$ 就行了。

时空复杂度：

时间：$O(N\sqrt{K})$

空间：$O(K)$

Xenia and String Problem

Codeforces 356E

题目大意：

定义好串 T 要满足下面的条件：

1. $|T|$ 为奇数；
2. 对于中点 $m = \frac{|T|+1}{2}$，字符 T_m 只出现一次；
3. $|T| = 1$，或子串 $S[1 \dots m-1], S[m+1 \dots |T|]$ 相同且都为好串。

给出字符串 S，最多改变 S 的一个字符，使 S 的所有为好串的子串的长度的平方和最大。

$|S| \leq 100000$

关键字：

哈希　枚举

算法讨论：

显然好串的长度都是 2 的幂次减 1，所以以某个位置为中心的好串个数不会超过 $\log_2 |S|$ 个，于是可以暴力哈希统计出不改变字符时的答案。

对于改变一个字符，枚举改变的位置和改变后的字符，首先可以通过预处理得到包含该位置的好串长度的平方和，即答案的减少量；然后需要统计加上新字符后答案的增加量，即求包含该位置新字符的好串长度的平方和。先枚举所有以该点为中点的好串，然后分别向左或向右翻一遍，如果能得到新的好串，就继续递归下去。

时空复杂度：

时间：$O(26|S| \log_2 |S|)$
空间：$O(|S| \log_2 |S|)$

Levko and Sets

Codeforces 360D

题目大意：

给出质数 P 和 N 个数 a_i 和 M 个数 b_i，要生成 N 个集合。对于第 i 个集合，首先 1 属于该集合。然后每次选出集合中的一个数 c，对于任意的 b_j，把数 $c \cdot a_i^{b_j} \bmod P$ 加入集合，直到集合大小不变为止。求最后 N 个集合并的大小。

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

关键字：

数论　原根　指数

算法讨论：

先考虑只有一个集合。由于 P 是质数，所以 P 存在原根，设其中的一个为 g，并令 $a_i = g^r$。然后集合中的每个数一定能写成 $a_i^{\sum c_j b_j}$ $(c_j \geq 0)$ 的形式，即 $g^{r \cdot \sum c_j b_j}$。根据费马小定理 $a^k \equiv a^{k \bmod (p-1)} \pmod p$，且底数为 P 的原根时幂与指数一一对应，所以即求 $r \cdot \sum c_j b_j$ 在模 $P-1$ 下的数字种数。首先考虑 $\sum c_j b_j \bmod (P-1)$，它一定是 $t = \gcd(b_1, b_2, \ldots, b_M, P-1)$。加上 r 后即 $\gcd(rt, P-1)$，设其为 q_i，于是第 i 个集合中的所有数即 g^{kq_i} $(0 \leq k \leq \frac{P-1}{q_i})$。

下面考虑求 q_i。直接求出原根再离散对数显然不行。考虑这个集合的大小 $d = \frac{P-1}{q_i}$，同时，d 也是 a_i^{kt} 的数字种数，即 $\mathrm{ord}_P a_i^t$ (a_i 模 P 的指数，即使 $a_i^k \equiv 1 \pmod P$ 成立的最小正整数 k)，于是 $q_i = \frac{P-1}{d}$。于是现在只需枚举 $P-1$ 的约数求出指数即可。

然后考虑有多个集合。对每个 a_i，求出对应的 q_i，问题就是求有多少个 x，满足存在一个 i，使得 $q_i | x$。由于 q_i 是 $P-1$ 的约数，所以问题等价于求有多少个 x，满足存在一个 i，使得 $q_i | \gcd(x, P-1)$。枚举一个 $P-1$ 的约数 k 作为 $\gcd(x, P-1)$，其中 k 要满足存在一个 i，使得 $q_i | k$。但由于 k 是 $P-1$ 的约数，所以枚举不会很多。然后就是一个经典问题：求有

多少 x，满足 $\gcd(x, P-1) = k$，答案为 $\varphi(\frac{P-1}{k})$。求 $\varphi(n)$ 时，直接用公式 $\varphi(n) = n - \sum_{d<n, d|n} \varphi(d)$ 递推。

时空复杂度：

时间：$O(\sigma(P-1)^2)$，$\sigma(n)$ 表示 n 的正约数个数。

空间：$O(N+M)$

Levko and Game

Codeforces 360E

题目大意：

n 个点，$m+k$ 条边的有向图，其中的 m 条边有固定边权，k 条边的边权可以在一个区间内。给出三个点 a,b,f，要求确定那 k 条边的边权，判断是否可以使得 $dist(a,f) < dist(b,f)$，如果不能再判断是否可以使得 $dist(a,f) = dist(b,f)$。($dist(a,b)$ 表示从点 a 到点 b 的最短路长度)

$n,m \leq 10^4$，$k \leq 100$

关键字：

最短路　贪心

算法讨论：

首先判断是否能使 $dist(a,f) < dist(b,f)$。先令这 k 条边的边权都为它的区间的最大值，求一遍最短路，如果对于这 k 条边中的某条边 (u,v)，满足 $dist(a,u) < dist(b,u)$，则一旦最后方案中两个点到 f 的最短路都经过该边，就一定能满足了。于是我们要尽量让两个点到 f 的最短路能都经过这条边，于是就要让这条边的边权尽量小，于是就把这条边的边权设为它的区间的最小值。这样一直做下去，直到不能更新为止。

然后判断如果不能满足上面的要求时，是否能使 $dist(a,f) = dist(b,f)$。用类似的方法，求出最短路后，如果对于这 k 条边中的某条边 (u,v)，满足 $dist(a,u) \leq dist(b,u)$，就把这条边的边权设为它的区间的最小值。

时空复杂度：

时间：$O(k \cdot \text{ShortPath}(n,m))$

空间：$O(n+m+k)$

Chapter 2

Google Code Jam (4)

Year of More Code Jam

Google Code Jam 2009 Final A

题目大意：

有 N 天，T 场比赛。每场比赛有 m_i 轮子比赛，分别在比赛开始后的第 $d_1, d_2, \ldots, d_{m_i}$ 天开始($d_1 = 1$)，每场比赛的开始时间在 $[1, N]$ 中随机。如果某一天进行了 s 轮子比赛，则答案的贡献为 s^2。求答案的期望，表示成分数形式。

$N \leq 10^9$，$T, m_i \leq 50$

关键字：

概率　期望

算法讨论：

枚举时间 t，则对于某场比赛 i，设 b_i 为 $\leq t$ 的 d_j 的个数，则这场比赛有 b_i 种开始时间的选择，使得它其中的一轮子比赛是在第 t 天进行的。把贡献 s^2 拆开，即求任意两场比赛 i, j 都在第 t 天有子比赛的贡献的期望，即

$$\sum_{1 \leq i \leq T} \frac{b_i}{N} + \sum_{\substack{1 \leq i,j \leq T \\ i \neq j}} \frac{b_i b_j}{N^2}$$

但是由于 $N \leq 10^9$。所以不能直接枚举时间。注意只有当 t 为某个 d_j 时，b_i 才会变化。所以把所有不同的 d_j 提取出来，一段一段枚举即可。

时空复杂度：

时间：$O(T^3 m)$
空间：$O(Tm)$

Letter Stamper

Google Code Jam 2010 Final A

题目大意：

有一台打印机，支持三种操作：

1. 把一个字母加到栈顶；
2. 打印栈顶字母；
3. 删除栈顶字母。

求打印出一个长度为 N 的序列 S 的最小操作次数，并要求最后栈为空。

$N \leq 2000$，S 只包含“A”、“B”、“C”三种字母。

关键字：

动态规划　贪心

算法讨论：

解决这个问题需要用到如下的性质：

1. 如果下一个要打印的字母就在栈顶，那么一定是直接打印；
2. 栈中不会出现两个连续的相同字母，因为这两个字母的作用是一样的；
3. 一定存在一种最优解，使得栈中不出现形如“XYX”的连续序列。假设当前栈顶是“XY”，现在需要打印“X”。一种方法是加入“X”，打印，最后删掉“X”；还有一种是删掉“Y”，打印“X”，然后重新加入“Y”。这两种方法的操作次数是一样的。

所以，栈中的字母一定是形如“$XYZXYZXYZ\ldots$”分布的，所以栈中字母只有 6 种情况：“ABC”、“ACB”、“BAC”、“BCA”、“CAB”和“CBA”。所以可以用动态规划，设 $f[i][j][k]$ 表示当前打印了前 i 个字母，栈中字母的状态是 j，栈中的字母数为 j 的最小操作次数，根据当前要打印的字母转移。

时空复杂度：

时间：$O(N^2)$
空间：$O(N)$

Graduation Requirements

Google Code Jam 2013 Final A

题目大意:

在一个环岛上，有个 N 路口。在 $0 \sim X$ 秒内，有 C 辆车在环岛里逆时针行驶，速度为 1 路口每秒。第 i 辆车在时刻 t_i 进入路口 s_i，然后从路口 e_i 离开(不会再回到路口 s_i)。你要开车在环岛里顺时针行驶，可以在任意时刻进入任意路口，行驶若干圈之后从任意路口离开，但中途不能与任意一辆车相遇。求你能在环岛里行驶的最长时间。

$N, X \leq 10^{10}$，$C \leq 1000$

关键字:

枚举

算法讨论:

如图 1，以路口位置为横坐标，以时间为纵坐标，建立平面直角坐标系。所有逆时针行驶的车的运行轨迹都可以表示成一条斜率为 1 线段，线段的端点都是整点，我们不妨称其为“正线段”。同理，那辆顺时针行驶的车的运行轨迹也可以表示成一条斜率为 -1 的线段，不妨称其为“反线段”。注意，这里的横坐标是循环的，即 N 的下一格是 1。现在问题转化成了：求那条反线段的最大长度，使得它不与任何正线段有公共点。

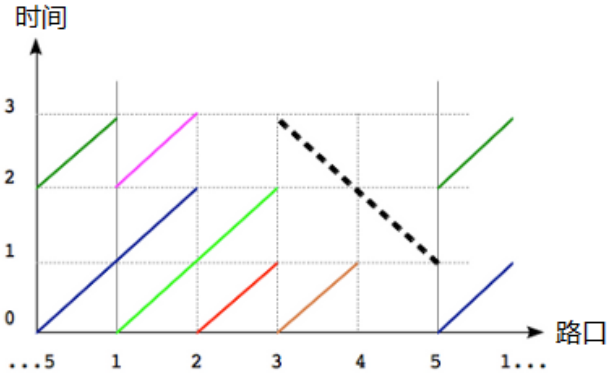


图 1

进一步分析发现，对于一种方案，我们可以把反线段不断平移，直到它将要碰到正线段为止(图 2)。

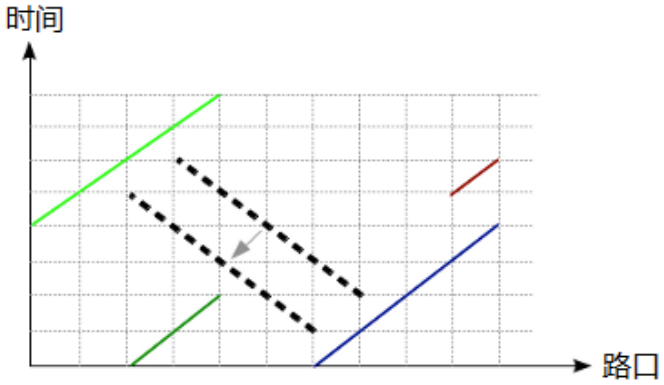


图 2

所以，一定存在一种最优方案，在这种方案里反线段离最近的正线段不会很远。更确切地，如图 3 所示，反线段一定经过某条正线段附近的这几个点(用红色标记)：

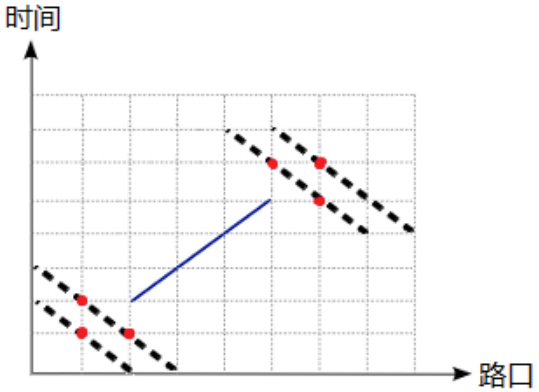


图 3

所以，算法大致成型了：枚举每条正线段，再枚举这条正线段附近的这些点，求出经过这个点的最长反线段长度，取个最大值即可。于是现在的问题是，如何求出经过某个点的最长反线段长度。

由于反线段斜率已知，它上面的某个点也已知，所以可以得到反线段所在的直线。用这条直线去与其他正线段求交，求出所有的相交时刻，并排序，相邻两个时刻的最大间隔即为答案。由于横坐标是循环的，所以还

有一些细节要处理。最重要的一点是，在循环的横坐标下，直线与线段的交点可能不止一个，但最多为两个(因为正线段长度最多为 $N-1$)，如图 4 所示：

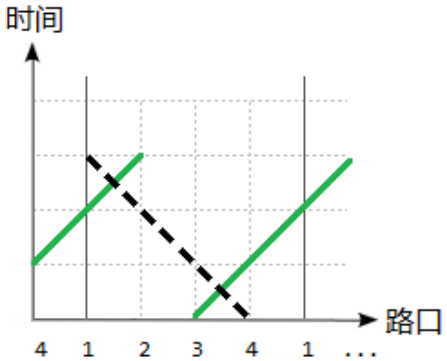


图 4

所以在求所有的相交时刻时这条线段要加入两个时刻。

时空复杂度：

时间：$O(C^2 \log_2 C)$
空间：$O(C)$

Paradox Sort

Google Code Jam 2014 Final D

题目大意：

N 个点的竞赛图，求一种字典序最小的点的排列，以排列的第一个点为起点，对于后面的点，如果当前点与它有边，则走到该点，否则不动，要求最后走到点 M。或判断无解。

$N \leq 100$

关键字：

逐位确定　图论　构造

算法讨论：

首先新加一个点，向所有点连边，于是起点就固定了。

由于要字典序最小，所以使用逐位确定，枚举排列中每一位的点，然后判断是否可行。于是现在的问题就是，给定起点 S 和终点 T，还有未被确定的点的集合 $U(S \notin U)$，判断是否存在一个 U 中点的排列，使得能从 S 到达 T。

首先如果 $S = T$，S 必须不能与 U 中的点有边，否则出去后就回不来了。

然后从 T 开始反向遍历 U 中的点，设所有能到的点的集合为 V，V 形成了一棵以 T 为根的树。则只要判断是否满足下面两个条件：

1. 如果 $V \neq U$，即 U 中有不在 V 中的点，令 $W = U \setminus V$，则 S 到 W 中的点之间不能有边。因为从 W 中的点开始一定不能到 T，所以它们只能被忽略。设 W 中的一个点为 x。如果是用 V 中的某个点 y 去忽略 x，则说明 y 到 x 之间没有边。而由于原图是竞赛图(有向完全图)，所以一定有 x 到 y 的边，但这样就有 $x \in V$ 了，矛盾。所以就只能用 S 去忽略 x，所以要满足 S 到 W 中的点之间不能有边。

2. S 至少要与一个 V 中的点有边。这是显然的，不然 S 就不能到 T 了。

这样就可以构造出一种 V 中点的排列，使得能从 S 到达 T。具体构造方法是：把 V 中的点按树中的深度排序，依次添加。则当前点的深度会不断减小，直到 T 为止。

时空复杂度：

时间：$O(N^4)$
空间：$O(N^2)$