IOI2015中国国家集训队作业

Part2:

泛做题目解题报告

黄涛岸

厦门双十中学

目 录

§1 Cow Tennis Tournament

试题来源 Codeforces 283E

题目大意

有n头奶牛进行比赛，能力值强的总是战胜能力值弱的。FJ会对比赛结果进行m次修改，每次将能力值在一个区间的奶牛相互的比赛结果翻转。询问所有修改后，有多少三元组(p,q,r)，满足p战胜q，q战胜r，r战胜p。

$n \leq 10^5$，能力值互不相同，$0 \leq m \leq 10^5$。

关键词

组合计数　线段树　扫描线

算法讨论

首先先将问题转化为统计不合法的三元组数目S_0，那么合法三元组数目S_1就等于$\binom{n}{3} - S_0$。

对于不合法的三元组，必定有且仅有一头奶牛，战胜了另外两个对手。设第i头奶牛赢得的比赛数目为w_i，那么$S_0 = \sum_{i=1}^{n} \binom{w_i}{2}$。

剩下的任务就是算出w_i。将n头奶牛按找能力值从小到大排序，并将其离散化。我们假设有一张$n \times n$的表格，第i行第j列表示第i头牛和第j头牛的比赛结果是否翻转了，0为否，1为是。那么FJ的每次修改就对应了将一个正方形的01翻转的操作。所有操作执行完后，w_i就等于第i列中，第1行到第$i-1$行的0的数目，加上第$i+1$行到第n行的1的数目。

这个问题可以利用扫描线加线段树的方法来处理。对于所有要翻转的正方形，将左右边界当做事件点。从左到右扫描，遇到事件就将其对应的区间翻转，再进行询问。这个需求可以由线段树提供区间翻转以及区间求和的操作来解决。

复杂度分析

时间复杂度：$O((n+m)\log n)$。

空间复杂度：$O(n+m)$。

§2 Cubes

试题来源 Codeforces 243D

题目大意

有一个$n \times n$的网格，一组对角在$(i, j-1)$和(i, j)的单位网格上有一个由$a_{i,j}$个单位立方体垒成的积木塔。问题要求的是，从无穷远处，以向量$\vec{v} = (v_x, v_y, 0)$看过去，能看到多少个不同的单位立方体。

$1 \le n \le 1000$，$0 \le a_{i,j} \le 10^9$，$|v_x|, |v_y| \le 10^4$且$|v_x| + |v_y| > 0$。

关键词

计算几何　线段树

算法讨论

过每个格点作$\vec{v}$的平行线，那么一个单位正方形包含在若干连续的平行线之间。一个立方体可见，当且仅当存在某两条相邻平行线，从它们之间可以看到该立方体。利用这个性质，容易想到建立一颗线段树，线段树节点$[l, r]$维护的是第l条平行线到第r条平行线之间可见的最低高度。将每个积木塔按距离观察者的距离从近到远排序，再依次枚举进行处理：

- 询问这个积木塔对应的平行线的区间中的最低高度$h_{\min}$
- 将这个积木塔对答案的贡献$\max(a_{i,j} - h_{\min}, 0)$计入答案
- 将对应的每个平行线区间的可见高度修改为$\max(a_{i,j}, h)$

另外一提的是预处理中的两个细节（$\vec{v_0} = (v_x, v_y)$）：

1. 一个格点$A(i, j)$对应的平行线可以用$\overrightarrow{OA} \times \vec{v_0}$表示（叉积可以表示一个点到平行线距离的远近及方向）
2. 一个格点$A(i, j)$离观察者的远近可以用$\overrightarrow{OA} \cdot \vec{v_0}$表示（点积表示一个向量再另一个向量上的投影长度）

那么一个单位正方形对应的平行线区间以及距离观察者的距离就可以由四个顶点对应的叉积和点积值来决定。

复杂度分析

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

§3 Airplane Boarding

试题来源　USACO Feb 14

题目大意

有n头奶牛要登机，分别站在$x = -(n-1), \ldots, -1, 0$处，$n$个座位在$x = 1, 2, \ldots, n$处。他们有自己指定的座位$S_i$，并且到达座位要花$T_i$的时间放置行李。如果一头奶牛前面没有阻挡那么每一秒钟可以往前一步，否则只能站在原地。求从登机开始到所有牛入座的时间。

$1 \le n \le 200000$，$\sum T_i \le 10^9$。

关键词

模拟　平衡树

算法讨论

对于每头牛，它只会被编号比它大的牛限制，因此我们依次计算第n头牛到第1头牛的入座时刻。我们用一个二元组的集合来表示当前第i头牛所受的限制，集合中的二元组(a,b)表示第i头牛在第b个时刻不能逾越a位置（因此一开始这个集合只有$(0,0)$）。

有了这个集合，对于第i头牛的处理，只需要找到满足$a < S_i$的二元组中$b-a$最大的那个，那么这头牛坐下的时刻就是$V_i = b - a + S_i + T_i$，同时将二元组(S_i, V_i)加入集合。

接着将这个集合从第i头牛转化到适应第$i-1$头牛，需要找到集合中所有$a \le S_i$的二元组，将它们替换成$(a-1,b)$。

以上所述区间修改、查询，以及插入新元素等操作，都可以用平衡树优化。

复杂度分析

时间复杂度：$O(n \log n)$。

空间复杂度：$O(n)$。

§4 White, Black and White Again

试题来源 Codeforces 306C

题目大意

有w个好元素和b个坏元素，元素两两不同。现要将这些元素分配到编号为$1,2,\ldots,n$的n个集合，一个集合只能全是好元素或者坏元素，且不能为空。所有坏元素集合的编号必须连续，且不能包含第一个和最后一个集合。求方案数模10^9+9。

$3 \le n \le 4000$，$2 \le w \le 4000$，$1 \le b \le 4000$。

关键词

组合数学

算法讨论

本题中n，w，b都不大，为枚举提供可能。我们考虑枚举坏元素连续集合的个数i。那么w个元素就要被分配在$n-i$个集合中方案数为$\binom{w-1}{n-i-1}$，b个元素在i个集合中$\binom{b-1}{i-1}$（利用隔板原理）。同时w个元素和b个元素顺序可以任意，且坏元素集合有$n-i-1$个连续段可以选择，因此总方案数为

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

具体实现需要预处理阶乘及其逆元。

复杂度分析

令$m = \max(w, b)$。
时间复杂度：$O(m)$。
空间复杂度：$O(m)$。

§5 Tournament-graph

试题来源 Codeforces 323B

题目大意

要求给n个点的无向完全图每条边定向，使得满足任意两点最短路不超过2。输出任意一种方案。

$3 \leq n \leq 1000$。

关键词

构造

算法讨论

对于$n = 3$，边集$\{(1,2),(2,3),(3,1)\}$符合题意（(a,b)表示一条从a到b的有向边）。

对于$n = 4$，无解。

对于$n = 6$，边集$\{(1,2),(1,3),(4,1),(1,5),(6,1),(2,3),(2,4),(5,2),(2,6),(3,4),(3,5),(6,3),(4,5),(4,6),(5,6)\}$符合题意。

假设$n = k$存在符合题意的方案，那么在$n = k$的方案基础上，可以构造出$n = k + 2$的方案：

1. 加入点$k + 1$和$k + 2$
2. 点$k + 1$向点$1, 2, \ldots, k$都连一条有向边
3. 点$1, 2, \ldots, k$都向点$k + 2$连一条有向边
4. 点$k + 2$向点$k + 1$ 连边。

容易证明这个方案符合题意。

因此对于所有n，只要是大于4的偶数或者不小于3的奇数都能构造出方案。

复杂度分析

时间复杂度：$O(n^2)$
空间复杂度：$O(n^2)$

§6 Threatening Letter

试题来源 USACO Dec 10

题目大意

给出长度为n的模式串以及长度为m的文本串。求最少需要用多少个模式串的子串来不重叠完全覆盖文本串，子串可重复使用。

$n, m \leq 50000$。

关键词

贪心 后缀数组

算法讨论

首先有个显然的贪心算法，就是从头开始每次取最长的模式串的子串去覆盖文本串，这样是可以达到最优解的。有了这个算法，问题转化为：求文本串中每个位置开始，单次匹配最长可以覆盖多长。

这个问题可以利用后缀数组解决。将文本串和模式串用一个特殊字符连接，建立后缀数组，再求出height数组。这个数组有个经典的性质，就是排完序的第i个后缀和第j个后缀（$i < j$）的LCP是$\min_{i \leq k < j} h[k]$。因此对于每个属于文本串的后缀，它能被匹配的最大长度一定由它在后缀数组中前后最靠近它的属于模式串的后缀决定。这一步可以在线性时间内完成。

复杂度分析

时间复杂度：$O((n+m)\log(n+m))$（若使用线性算法求后缀数组复杂度为$O(n+m)$）

空间复杂度：$O(n+m)$。

§7 Tower of Hay

试题来源 USACO Open 09

题目大意

给定长度为n的正整数序列，你要分成若干段，使得对于相邻的两段，前一段的和不小于后一段的和，并且段数最多。输出最多的段数。

$1 \leq n \leq 10^5$，序列中的数不超过10^4。

关键词

动态规划 单调队列

算法讨论

首先有个结论：第一段的和越小，只要能划分得出来，能划分的段数不会减少，即，使得第一段的和最小段数一定最多。这个结论的证明思路大致是取一个段数最多的方案和第一段和最小的方案进行调整。

有了这个结论，我们可以通过动态规划来解决这个问题。设$f[i]$表示第i到第n个数进行划分，第一段的和最小是多少。$sum[i]$表示第1到第i个数的和。状态转移方程为：

$$f[i] = \min_{\forall i<j\leq n, sum[j-1]-sum[i-1]\geq f[j]} sum[j-1] - sum[i-1]$$

这样暴力求出$f[i]$复杂度是$O(n^2)$的，我们仍需要优化。

首先注意到，因为$sum[i]$是递增的，所以我们在求$f[i]$时，只要找到第一个符合条件的j。对于$f[i]$成立的决策j，有$sum[i-1] \leq sum[j-1] - f[j]$，那么对于$f[1], f[2], \ldots, f[i-1]$决策$j$都成立。这样我们可以用单调队列维护一个$i$递减，$sum[i-1] - f[i]$递减的序列。那么每次转移只需要从队头开始找到一个最靠后的合法决策就可以了。

复杂度分析

时间复杂度：$O(n)$。

空间复杂度：$O(n)$。

§8 Close Vertices

试题来源 Codeforces 293E

题目大意

给一颗n个点的树，边上带权。统计有多少个点对(u, v)满足$u < v$，u到v的路径边数不超过L，且边权和不超过W。

$1 \leq n \leq 10^5$，$1 \leq L \leq n$，$1 \leq W \leq 10^9$，边权小于10^4。

关键词

树分治

算法讨论

这是一个统计树上路径数量的问题，可以将问题转化为统计过某点的合法路径数，再使用树分治求解。

对于一个大小为m的树，我们需要统计过根的合法路径数。用l_i表示每个点到根的边数，w_i表示每个点到根的边权和。接下来我们先统计满足$l_i + l_j \leq L$且$w_i + w_j \leq W$的(i, j)对数：按w_i排序，从大到小依次枚举每个w_i，指针j从

小到大单调地扫描满足$w_i + w_j \leq W$的位置并将扫过的l_j加入树状数组，再用树状数组统计$l_j \leq L - l_i$的数量即可。最后我们需要扣除i, j同在根的某个儿子的子树中的情况，方法与上诉说的基本相同。

该子问题得以解决，那么原问题就可以利用树分治顺利解决。

复杂度分析

时间复杂度：对于大小为m的树，统计过根的合法数目的时间复杂度为$O(m \log m)$。因此利用树分治的总复杂度为$O(n \log^2 n)$。

空间复杂度：$O(n)$。

§9 Wi-fi Towers

试题来源 GCJ 2009 Final D

题目大意

平面内有n座信号塔，每座塔坐标给定为(x_i, y_i)，控制范围为以它为圆心，半径为r_i的圆。每座塔可以选择升级与否，升级会带来收益s_i（可正可负）。若要升级，要求这座塔控制范围内的塔都要升级。求最佳升级方案收益是多少。

$n \leq 500$，一个输入文件包含T组数据，$T \leq 55$。

关键词

最大权闭合子图　计算几何

算法讨论

信号塔是否升级存在依赖关系，并要求取出的信号塔权值最大，这是一个明显的最大权闭合子图问题。构图如下：

- 对于$s_i \geq 0$的信号塔，从源点向i连一条容量为s_i的边
- 对于$s_i < 0$的信号塔，从i向汇点连一条容量为$-s_i$的边
- 对于信号塔i和j，若j在i的控制范围，从i向j连一条容量无穷大的边

利用这个网络求最大权闭合子图即可通过本题。

另外值得一提的是，可以将边数优化成$O(n)$级别的。首先要明确的是只要网络的传递闭包不变，那么求出的结果是一样的。将有向边看成平面上的向量，注意到对于边$\overrightarrow{AC}$及$\overrightarrow{BC}$，若$\angle ACB \leq 60°$，那么我们可以将较长的边删除。不妨设$|\overrightarrow{AC}| > |\overrightarrow{BC}|$，那么由于A的控制范围是一个圆，必然存在A到B的连边。这样一来，对于每个点的入边最多只有6条，总边数是$O(n)$的。

复杂度分析

时间复杂度：$O(Tn^4)$，若加了边数优化则为$O(Tn^3)$（网络流使用的是$O(V^2E)$的算法）。

空间复杂度：$O(n^2)$。

§10 GCD Table

试题来源 Codeforces 338D

题目大意

给定一个n行m列的表格G，$G(i,j) = \gcd(i,j)$。再给一个长度为k的数组a。问是否存在$1 \le i \le n$且$1 \le j \le m-k+1$，满足$\forall 1 \le l \le k, G(i,j+l-1) = a_l$。

$1 \le n,m,a_i \le 10^{12}$且$1 \le k \le 10000$。

关键词

线性同余方程

算法讨论

令$L = lcm(a_1,a_2,\ldots,a_k)$，则若此问题有解，数组$a$必定在第$L$行出现过。因此若$L > n$则问题无解。设数组$a$在矩阵$G$第$L$行的第$j$到$j+k-1$列匹配，那么原问题有解的必要条件是以下方程组有解：

$$\begin{cases} j \equiv 0 \pmod{a_1} \\ j \equiv -1 \pmod{a_2} \\ \cdots \\ j \equiv -(k-1) \pmod{a_k} \end{cases}$$

根据中国剩余定理可知，方程组在$[1,L]$上有唯一解。通过将方程逐个合并的方法，可以求出原问题的解。最后只需要将解带入原问题验证是否符合题意即可。

复杂度分析

时间复杂度：$O(k \log n)$。
空间复杂度：$O(k)$。

§11 Roadside Trees

试题来源 Codeforces 264E

题目大意

从左到右排列有n个位置，一开始为空。有m个时刻，每个时刻可以在某个空位置放入一个数h，或者将编号第x小的非空位置清空。每个时刻过后，你都要求一次非空位置的最长上升子序列，再将每个非空位置的数加一。任何时刻位置中的数字两两不同，曾经放过数的位置清空之后不会再被添加新数字。

$1 \le n \le 10^5$，$1 \le m \le 2 \cdot 10^5$，$1 \le h, x \le 10$。

关键词

最长上升子序列　线段树

算法讨论

首先将时刻t加入的数值h改为$h - t$，就可以忽略每个时刻数值的增加。将每个数值用坐标(p, h)表示（p表示其位置编号，h为数值），那么我们要求的是一个满足p, h都单调递增的点的序列。每个点记录一个f_i，表示以它为起始，最长的序列是多长，那么$f_i = \max_{p_i<p_j, h_i<h_j} f_j$。

注意到一个p最多对应一个h，一个h最多对应一个p。我们可以把两个维度拆开，按x轴和y轴建立两颗线段树，而不是建立二维线段树。线段树上的结点维护对应区间的f_i的最大值。对应的操作可以进行如下维护：

- 对于插入操作，由于$h \le 10$，所以f_i会被修改的最多只有10个点。在x轴的线段树，暴力将小于h的数字删除，再按照h的值从大到小加入。加入的时候用x轴的线段树询问每个点的f_i，并在两颗线段树上更新。
- 对于删除操作，由于$x \le 10$，所以会被修改的也最多只有10个。在y轴的线段树，暴力将会被修改的点删除，再按照p的值从大到小加入。加入的时候用y轴的线段树询问每个点的f_i，并在两颗线段树上更新。

至此此问题得以解决。

复杂度分析

时间复杂度：$O(m \log n)$。
空间复杂度：$O(m + n)$。

§12 Cow Neighborhoods

试题来源　USACO Open 08

题目大意

给定平面上n个不同的点(x_i, y_i)，若两点之间的曼哈顿距离不超过C，则这两点之间有一条连边。问连通块的个数及最大的连通块大小。

$1 \le n \le 10^5$，$1 \le x_i, y_i, C \le 10^9$。

关键词

并查集　平衡树　双指针

算法讨论

先将坐标(x, y)替换为$(x+y, x-y)$，那么两点间的曼哈顿距离变为两点间

的切比雪夫距离[1]。接下来我们将每个点按照x排序，枚举每个点i，用另外一个指针扫描维护一个坐标集合。这个集合内的点(x_j, y_j)满足$x_j \le x_i$且$x_i - x_j \le C$，且按照y排序。将第i个点加入集合后，找到这个点的前驱和后继。分别检验前驱及后继这两个点是否满足题意，若满足则利用并查集将第i个点并入其所属的连通块。

这个算法的正确性是可以保证的，对集合的维护可以采用平衡树。最后统计答案的时候只需要利用并查集的信息即可。

复杂度分析

时间复杂度：$O(n \log n)$。

空间复杂度：$O(n)$。

§13 Maxim and Calculator

试题来源

Codeforces 261E

题目大意

你有两个变量a和b，初始$a = 1, b = 0$。每步可以进行以下两种操作之一：

- $a = a \cdot b$
- $b = b + 1$

问有多少个数字x（$l \le x \le r$）可以在p步之内由变量a得到。

$2 \le l \le r \le 10^9$，$1 \le p \le 100$。

关键词

暴力　动态规划

算法讨论

设cnt_p表示在$[1, 10^9]$质因数不大于p的数的个数。经过暴力枚举，可以发现cnt_p最大大约是$3 \cdot 10^6$。那么我们就可以在$O(cnt_p)$的时间内将这些数处理出来。

接着需要求出这些数最少要多少步可以得到。先将这些数从小到大排序设为$a_1, a_2, \ldots, a_{cnt_p}$。考虑使用动态规划。用$f[i][j]$表示得到$a_j$的过程中，变量$b$不超过$i$，最少的操作1的次数。那么转移方程如下：

$$f[i][j] = \begin{cases} f[i-1][j] & i \nmid a_j \\ \min(f[i][k] + 1, f[i-1][j]) & a_k \cdot i = a_j \end{cases}$$

对于a_j，满足$l \le a_j \le r$且存在$f[i][j] + i \le p$，则a_j应该被统计入答案。

[1]两个点$(x_1, y_1), (x_2, y_2)$的契比雪夫距离为$\max(|x_1 - x_2|, |y_1 - y_2|)$

复杂度分析

时间复杂度：$O(cnt_p p)$。
空间复杂度：$O(cnt_p)$，DP过程使用滚动数组。

§14 First!

试题来源 USACO Dec 12

题目大意

给定n个只包含小写字母的字符串，在允许改变字母字典顺序的情况，问哪些字符串可以成为字典序最小的。

$1 \le n \le 30000$，字符串总长不超过300000且两两不同。

关键词

字典树 DAG判定

算法讨论

我们枚举每个字符串s进行判断。首先任意一个字符串都不能是s的前缀。在这个前提下，我们枚举其他所有字符串t，找到两个字符串第一个不同的位置i，从字母s_i向t_i连一条有向边，表示s_i在字母字典序中要先于t_i。最后我们只需要判断这张图是否为DAG即可，可以使用Tarjan算法或者进行拓扑排序。

而对于构图，可以对所有字符串建立trie树，从根沿着字符串s往下走。对于每个路径上的结点，从该点向所有兄弟结点连边即可构出这张图。

复杂度分析

设字母集的大小为$size$，字符串总长L。
时间复杂度：$O(n\,size^2)$。
空间复杂度：$O(L)$。

§15 Ciel and Flipboard

试题来源 Codeforces 321D

题目大意

有一个$n \times n$的数阵，你可以任意多次地将任意大小为$x \times x$（$x = \frac{n+1}{2}$）的子矩阵中每个数变为其相反数。问操作后所有数之和的最大值是多少。

$1 \le n \le 33$，且n为奇数。

关键词

枚举 贪心

算法讨论

我们用一个01矩阵$f_{i,j}$表示第i行第j列的数字是否被取反，是为1，否为0。注意到以下两个性质（$\oplus$表示异或运算）：

$$\forall 1 \le i \le n, 1 \le j < x: \quad f_{i,j} \oplus f_{i,x} \oplus f_{i,j+x} = 0$$
$$\forall 1 \le j \le n, 1 \le i < x: \quad f_{i,j} \oplus f_{x,j} \oplus f_{i+x,j} = 0$$

因此我们只需要确定了最左上角$x \times x$的$f_{i,j}$，就可以确定整个矩阵的情况，并且一定对应一种方案。

我们可以通过2^x枚举将$f_{1,x}, f_{2,x}, \ldots, f_{x,x}$的值固定，那么$f_{i,x}$的值都是已知的了。对于其他的位置，根据上述性质，可以发现$f_{i,j}, f_{i+x,j}, f_{i,j+x}, f_{i+x,j+x}$（$0 < i, j < x$），需要且仅需要$f_{x,j}$和$f_{i,j}$的值就可以被唯一确定了。因此，我们只需要单独枚举$f_{x,j}$并贪心地确定$f_{i,j}$即可。这样，此问题得以完美解决。

复杂度分析

时间复杂度：$O(2^{\frac{n}{2}} n^2)$。

空间复杂度：$O(n^2)$。

§16 More Queries to Array...

试题来源

Codeforces 266E

题目大意

你需要支持对一个长度为n的数组a（下标从1到n）进行m此操作。操作有两种：

1. 求$\sum_{i=l}^{r} a_i \cdot (i-l+1)^k$。
2. 将$a_l, a_{l+1}, \ldots, a_r$全部修改为$x$。

$1 \le n, m \le 10^5$，$0 \le k \le 5$，数组中的元素任何时刻满足$0 \le a_i \le 10^9$

关键词

线段树　二项式定理

算法讨论

先对询问内容进行变形：

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

其中$s_i = \sum_{j=l}^{r} a_j \cdot j^i$。

这样一来，我们只需要维护$k+1$棵线段树，编号0到k，其中第i棵维护$\sum a_j \cdot j^i$的值。询问时，分别查找第i棵线段树对应的区间和求出s_i，从而得到答案。而对于修改，使用懒标记即可。

复杂度分析

时间复杂度：$O(mk \log n)$。

空间复杂度：$O(n)$。

§17 Reclamation

试题来源 Codeforces 325D

题目大意

有一个$r \times c$的网格，把左边界和右边界粘起来使得形成一个圆柱。现在要不断地挖去其中的格子，要求挖去后存在一条从最上方到最下方的四联通路径。如果某次操作后不满足要求则不执行。问最后有多少次操作是成功的。

$r, c \leq 3000$，操作数n不超过300000。

关键词

并查集

算法讨论

我们可以将$r \times c$的圆柱侧面展开，再将网格复制一份，变成一个$r \times 2c$的网格，其中单元格(x, y)和$(x, y+c)$表示圆柱侧面的同一个格子（$1 \leq x \leq r$，$1 \leq y \leq c$）。那么存在一条最上方到最下方的四联通路径，等价于对于任意(x, y)，不存在一条从(x, y)到$(x, y+c)$，由删去格子组成的八连通路径。

有了此结论，对于题目中的每次操作，执行两个步骤：

1. 判断若删去(x,y)及$(x,y+c)$后，是否存在一条从(x,y)到$(x,y+c)$的八连通路径。
2. 若不存在，将(x,y)及$(x,y+c)$删除。

要实现这两个步骤，需要借助并查集维护已删格子的连通性。对于步骤1，只需要判断(x,y)周围八个格子及$(x,y+c)$周围八个格子是否有在同一连通块的。对于步骤2，将要删除的格子与周围八个格子合并到同一连通块即可。这样，我们按照读入顺序依次处理即可得到答案。

复杂度分析

时间复杂度：$O(n\alpha(n))$。

空间复杂度：$O(n)$（可以使用哈希表只存储被删除的位置）。

§18 Three Swaps

试题来源 Codeforces 339E

题目大意

给出将一个n的排列任选一段区间翻转不超过3次后结果，要求给出一种可行的操作方案。

$1 \le n \le 1000$。

关键词

暴力

算法讨论

设排列为$a_1,a_2,\ldots,a_n$，令$a_0=0$，$a_{n+1}=n+1$。设我们提供的方案的第i次的翻转区间为$[l_i,r_i]$。一定存在一种方案，满足在第i次操作前，a_{l_i}与a_{l_i-1}或者a_{r_i}与a_{r_i+1}为一对相邻的整数。对于原操作次数不超过2次的，这个结论显然成立。而对于原操作需要3次的，即便原操作不一定满足上面的性质，但我们总能够构造的出来。

有了这个结论，我们从最后一步开始使用回溯的搜索方法，便能很快找到解。

复杂度分析

时间复杂度：若原操作有3次，最后一步最多有11种可能的区间。因此递归的次数不超过11^3，复杂度为常数较大的$O(n)$。

空间复杂度：$O(n)$。

§19 BerDonalds

试题来源 Codeforces 266D

题目大意

给一张n个点的无向连通图，求图的一个点（可以是某条边上的任意一点），使得其他结点到这个点的最大距离最小。

$n \leq 200$。

关键词

最短路 数形结合

算法讨论

我们可以枚举答案所在的边(u, v)。设这条边的长度为L，用$f(x)$表示这条边上和u的距离为x的位置，到每个结点的最大距离（$0 \leq x \leq L$）。那么我们有:

$$f(x) = \max_{1 \leq i \leq n} (\min(dis_{u,i} + x, dis_{v,i} + L - x))$$

其中$dis_{u,v}$表示u到v的最短路。

我们要求的是$f(x)$的最小值。根据函数图像可以观察到，$f(x)$是由一些上下波动的折线段组成。又由于这些线段的斜率相等，我们可以按照$dis_{u,i}$从大到小的顺序，依次求出每个拐点，取其中的最小值即可。

复杂度分析

时间复杂度：$O(n^3)$。

空间复杂度：$O(n^2)$。

§20 Yaroslav and Algorithm

试题来源 Codeforces 301C

题目大意

你需要给出一组命令，实现对一个正整数（看做字符串）加一的操作。每个命令包含两个字符串s和w，命令的意义是：若当前字符串包含子串s，则将第一次出现的s替换为w。s和w用>>或者<>连接，若连接号为>>则执行完后从头再找第一个能被执行的命令去执行，否则退出程序。命令中的字符串可以包含问号'?'。

关键词

构造

算法讨论

命令允许使用'?'是本题的突破口。我们可以将'?'用作指针来模拟高精度加法的过程。具体来说，我们可以在串首添加"??"，再逐步移动到串末。然后把"??"替换成"?"开始从低位到高位的进位操作。

以下是具体的命令：

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

复杂度分析

时间复杂度：$O(1)$。

空间复杂度：$O(1)$。

§21 Playing with String

试题来源 Codeforces 305E

题目大意

给一个长度为n的初始序列，序列中每个位置被标记为可操作或者不可操作（头尾两个位置不可操作）。现有两人进行游戏，每次选择一个可操作位置i，将位置$i-1,i,i+1$标记为不可操作。第一个无法操作者输。问先手是否必胜，若是输出必胜策略的第一步操作。

$1 \le n \le 5000$。

关键词

组合游戏　SG函数

算法讨论

显然，我们可以将长度为i的连续可操作位置看做一个子游戏，我们称作子游戏i。全局可以看做是若干个子游戏的和，那么全局的SG函数值就是所有子游戏SG函数值的异或和。我们可以在线性时间内处理出这些子游戏。

现在讨论如何求子游戏i的$SG(i)$。我们可以枚举子游戏i所有可能的操作，那么我们有：

$$SG(i) = \begin{cases} 0 & i \leq 0 \\ \underset{1 \leq i \leq j}{mex} \{sg(j-2) \oplus sg(i-j-1)\} & i > 0 \end{cases}$$

这样我们可以求得全局的SG值。若不为零，则先手必胜，我们枚举每个可操作位置求出操作后的SG值即可判断是否为必胜策略的第一步；否则先手必败。

复杂度分析

时间复杂度：$O(n^2)$。
空间复杂度：$O(n)$。

§22 Jeff and Removing Periods

试题来源　Codeforces 351D

题目大意

对于一个数列你可以进行这样一种操作：将一段数值相同且下标成等差数列的子序列删除后重新排列。一个序列的美好度定义为将数列清空的最少操作数。

现你有一个长度为n的数组，有m次询问，问数组的第l到第r个元素组成的数列的美好度是多少。

关键词

离线　树状数组

算法讨论

根据题目性质，我们可以发现，第一次操作之后，我们可以把相同的数排在一起，之后每次操作可以将一种数字全部删除。而对于第一次删除的数字，若存在某一数字的全部下标可以构成等差数列，就优先把它删除，这种

情况数列的美好度为不同数个数；否则任删一个数字，美好度则为不同数个数+1。

这样，原问题就变为，求一段区间不同数个数，以及判断区间内是否有某个数字所有下标成等差数列。

对于区间内不同数字个数统计，是一个经典问题，解决方法如下：

1. 预处理每个位置i，前面最近一次出现a_i的位置p_i（$p_0 = 0$）。
2. 将所有询问离线，按照右端点排序。
3. 枚举每个位置i，将下标i加入树状数组，同时将下标p_i删除。
4. 对于每个右端点等于i的询问，统计区间内有多少个数在树状数组里。

而对于判断，我们预处理一个类似于p_i的值$fail_i$（$fail_0 = 0$），表示对于数值为a_i的位置，从i开始往前查找到的第一个下标不能构成等差数列的位置。那么有：

$$fail_i = \begin{cases} fail_{p_i} & i - p_i = p_i - p_{p_i} \\ p_{p_i} & \text{otherwise} \end{cases}$$

那么问题转化为，统计区间内有多少不同的数，其最后一个$fail_i$落在该区间中。统计方法与上文所说的统计区间不同数字个数类似。

复杂度分析

时间复杂度：$O(m(\log m + \log n))$。

空间复杂度：$O(n + m)$。

§23 Gangs of Istanbull

试题来源 USACO Dec 12

题目大意

你有一个特殊的数字集合，只允许包含同种数字。现要加入一个数，若这个数与集合内的数相同或者集合为空，则直接加入，否则不加入且还要从集合内取走一个元素。

现在你有范围在1到m的数字，每种数字的个数a_i，共n个。求一种字典序最小的加数方案，同时保证最后留在集合中的元素是1是最多的。

$1 \le m \le n \le 10^6$，a_i至少为1。

关键词

贪心 构造

算法讨论

首先先确定最后留在集合中的1个数的最大值。显然，我们可以贪心地让其他元素先相互抵消，不过这种贪心有可能受到个数最多的数字的限制。

设$p = \max_{1<i\leq n} a_i$，那么$cnt = \min(a_1 - (n - a_1) \bmod 2, n - 2p)$。若$cnt \leq 0$，则问题无解，否则我们就能构造出一种方案。

接下来考虑构造字典序最小的方案。先将最后留在集合中的cnt个1扣除。每轮我们将剩余数字中最小的数全部加入集合，那么接着我们要加入同样多的其他数字。我们设当前个数最多的数字为r，当前数字总数为tot。依次考虑还要加入的数字：若$a_r * 2 \geq tot$，那么我们应该优先加入r，以保证问题有解；否则我们选择剩余数字最小的加入。重复上面的过程，直到数字全部被用完。为了保证时间复杂度为线性，代码实现需要指针和标志数组的辅助。

复杂度分析

时间复杂度：$O(n)$。
空间复杂度：$O(n)$。

§24 Sereja and Squares

试题来源 Codeforces 314E

题目大意

有25种括号，可以用来组成括号序列。一个合法的括号序列要求任意前缀左括号数不少于右括号数，且存在一种配对方式满足任意两对括号只能是包含或者相离的关系。给定一个长度为n的合法括号序列的某些左括号的位置信息，问原序列有多少种。

$n \leq 10^5$。

关键词

动态规划

算法讨论

对于这类问题，容易想到使用动态规划解决。设$f[i][j]$表示前i个位置，已经匹配成功j对括号的方案数。转移方程如下：

$$f[i][j] = \begin{cases} f[i-1][j] & \text{第}i\text{个位置是一个已被确定的位置} \\ f[i-1][j] + f[i-1][j-1] & \text{第}i\text{个位置未被确定} \end{cases}$$

这个方程直接转移复杂度是$O(n^2)$的，需要一些常数优化以通过这道题：

- DP过程使用滚动数组，这样对于确定的位置i直接利用上一次的信息。
- 对于不确定的位置i，精确计算j的取值范围：$\max(0, i - n/2) \leq j \leq i/2$。

复杂度分析

时间复杂度：$O(n^2)$。

空间复杂度：$O(n)$。

§25 Doubly-sorted Grid

试题来源 GCJ 2009 Final C

题目大意

给出一个$n \times m$的方格纸，在空方格内可以任意填写字母a到z，必须保证，任意一行（或者列）从左到右（从上到下）字母字典序必须保证不降。求填字母的方案数。

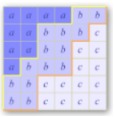
$1 \leq n, m \leq 10$。

关键词

轮廓线　状态压缩动态规划

算法讨论

观察一个合法的双调矩阵，可以发现，每个字母出现的区域可以用一对轮廓线圈出来（如右图的字母b，被黄色和橘色的轮廓线圈住）。更进一步，可以知道，这样的轮廓线满足从左到右不降的，所以总个数不超过$\binom{n+m}{n} \leq 184756$。由于总个数并不是很大，可以考虑用轮廓线表示状态。



我们关注每个轮廓线以上的字母，而不是关心每个字母的轮廓线。那么我们可以用$f_{s,c}$表示轮廓线s以上最大字母不超过c的方案数。这样我们在转移时候只考虑拐点处（即每个直角处）的字母，需要用到容斥原理，转移复杂度为$O(2^n)$，而状态数为$O(c2^{n+m})$，显然无法接受。

考虑将容斥的问题转移到状态表示上，用$f_{s,c,i}$表示轮廓线s以上最大字母不超过c，且第c小的字母全都在前i行的方案数。这样转移的时候只需要考虑第i行是否存在字母c，转移复杂度降为$O(1)$，而状态数最多只多了10倍。

复杂度分析

时间复杂度：$O(cn2^{n+m})$。

空间复杂度：$O(2^{n+m})$，DP过程使用滚动数组。

§26 Year of More Code Jam

试题来源 GCJ 2009 Final A

题目大意

一年有n天，这一年中会开始T组锦标赛。每组比赛有m_i场比赛，第j场是在该组锦标赛开始的第$d_{i,j}$天进行的（$d_{i,1} = 1$）。若一位选手在第i天（$1 \leq i \leq$

n）参加了k场比赛，那么他会收获k^2的开心度。这T场比赛会在这n天内完全随机地开始进行，问一位选手开心度的期望。

$n \le 10^9$，$2 \le m \le 50$，$2 \le d_{i,j} < d_{i,j+1} \le 10^4$，$T \le 50$。

关键词

数学期望

算法讨论

这是一道离散概率题。设第i天进行的比赛为X_i。我们要求的是：

$$Ans = E\left(\sum_{i=1}^{n} X_i^2\right) = \sum_{i=1}^{n} E\left(X_i^2\right)$$

现在考虑如何求$E(X_i^2)$。设Y_j表示第j组锦标赛在第i天的比赛场数（Y_j是一个取值为0或1的变量）。可以得到：

$$\begin{aligned} E\left(X_i^2\right) &= E\left(\left(\sum_{j=1}^{T} Y_j\right)^2\right) \\ &= E\left(\sum_{j=1}^{T} Y_j^2 + 2\sum_{1\le j<k\le T} Y_j Y_k\right) \\ &= \sum_{j=1}^{T} E\left(Y_j^2\right) + 2\sum_{1\le j<k\le T} E\left(Y_j Y_k\right) \end{aligned}$$

设$cnt_{i,j}$表示第j组锦标赛在第i天可以开始的场次，那么有:

$$E\left(Y_j^2\right) = E\left(Y_j\right) = \frac{cnt_{i,j}}{n}$$
$$E\left(Y_j Y_k\right) = \frac{cnt_{i,j}cnt_{i,k}}{n^2}$$

至此我们有了时间复杂度为$O(nT^2)$的算法。

设所有$d_{i,j}$的最大值为D，D的取值最大到10^4。注意到$cnt_{i,j} = \max_{d_{j,k}\le i} k$，可以发现对于$i > D$，$cnt_{i,j} = m_j$，是恒定的，也就是说$E(X_i)$恒定。那么我们只需要计算$X_1, X_2, \ldots, X_D$即可，这样我们有了$O(DT^2)$的算法。

其实本题还可以继续优化。注意到$cnt_{i,j}$随着i增大只会有m_j次变化。对于每次变化，我们可以在$O(1)$的时间内算出$\sum cnt_{i,j}cnt_{i,k}$以及$\sum cnt_{i,j}$的增量，加入答案。

至此，本问题通过层层优化完美解决了。

复杂度分析

时间复杂度：$O(D + Tm + \log n)$。

空间复杂度：$O(Tm)$。

§27 Google Code Jam

试题来源

Codeforces 277D

题目大意

有一场时长为T的GCJ赛制的比赛，包含n个问题。对于一个问题，给定解决小数据及大数据的时间tS和tL，以及对应分数sS和sL。对于小数据，做出来就能保证通过，大数据则有p的概率是错误的。一场比赛的罚时是最后一次正确提交的时间。求一场比赛的最高期望得分，在这个前提下罚时尽量小。

$1 \le n \le 1000$，$1 \le T \le 1560$。

关键词

贪心　动态规划　数学期望

算法讨论

首先，我们考虑如果要解决问题i和j，应该先解决哪一个，才能使期望罚时尽量小。假设i应在j前解决，我们有：

$$(tL_i + tL_j)(1-p_j) + tL_i(1-p_i)p_j < (tL_j + tL_i)(1-p_i) + tL_j(1-p_j)p_i$$

化简得：

$$\frac{tL_i \cdot p_i}{1-p_i} < \frac{tL_j \cdot p_j}{1-p_j}$$

因此按$tL_ip_i/(1-p_i)$为关键字进行排序后，我们再考虑按顺序选取。

对于后半部分的问题，是经典的背包问题，关键在于计算“物品”的价值。用$f[i][j]$表示前i个问题，花费时间j的最大期望得分，同时也记录取到最大得分时的期望最小罚时$t[i][j]$。分两种情况转移：

1. 如果只做小数据（$j \ge tS_i$），$f[i][j] = f[i-1][j-tS_i] + sS_i$，$t[i][j] = t[i-1][j-tS_i] + tS_i$。
2. 如果整道题都完成（$j \ge tS_i + tL_i$），$f[i][j] = f[i-1][j-tS_i-tL_i] + sS_i + sL_i(1-p_i)$，$t[i][j] = (t[i-1][j-tS_i-tL_i] + tS_i)p_i + j(1-p_i)$

最后从$f[n][i]$（$0 \le i \le T$）中需找最优答案即可。

注意精度问题，对于c++代码实现时建议使用long double。

复杂度分析

时间复杂度：$O(nT)$。

空间复杂度：$O(n+T)$，DP过程使用滚动数组。

§28 Balance

试题来源 Codeforces 317C

题目大意

有n个容器，容器间有m条水管可以运输任意整数体积的水。给定每个容器的初始以及目标水量a_i, b_i。每个容器体积上限都是v。每一步可以在两个有水管连接的容器之间运水。给出一种步数不超过$2n^2$的运输方案，使得每个容器达到目标水量。

$1 \le n \le 300$，$1 \le m \le 50000$。

关键词

构造

算法讨论

首先容易知道，只要每个连通块满足$\sum a_i - b_i = 0$，那么一定有解，而且不需要关心连通块内的连边情况。因此，我们可以对原图提取一个生成森林，那么边数最多为$n-1$。

接下来，我们枚举同个连通块内每个点对(s,t)，满足$a_s < b_s$且$a_t > b_t$。设$d = \min(b_s - a_s, a_t - b_t)$，那么我们执行一个过程$flow(s,t,d)$，表示模拟从$t$到$s$运输$d$体积水的过程。显然，我们只需要在$s$到$t$这条链上运输就可以了。设$t$在这条链上的前驱结点为$w$。先从$w$运输$d' = \min(a_w, d)$体积到$t$，执行$flow(s,w,d)$，再从$w$运输$d-d'$体积到$t$。可以发现这一过程能保证每个容器体积不超过$v$，而且只会改变$s$和$t$的储水状态，使得$s$和$t$至少有一个达到目标水量。

每找到一个点对，都能使其中一个点达到目标，因此最多会有n个这样的点对。而每个点对之间的边最多$n-1$条，因此总步数最多为$2n(n-1)$，符合题意。

复杂度分析

时间复杂度：$O(n^2)$。
空间复杂度：$O(n^2)$。

§29 PE Lesson

试题来源 Codeforces 316D3

题目大意

你有一个初始排列$1, 2, \ldots, n$，你可以任意多次任选两个不同位置交换。每个位置交换次数不能超过c_i，c_i等于1或2。问最后排列可能有多少种。

$1 \leq n \leq 10^6$。

关键词

数学推导

算法讨论

对于一个排列，我们可以将其拆成若干个环。我们可以通过以下过程，生成这些环：

> 我们令$p(x)$表示x在排列中的位置。每次找到一个不在环中的x，必定存在最小的正整数m，p对x进行m次作用使得$p^m(x) = x$，那么对于所有i（$0 \leq i < m$），我们将$p^i(x)$与$p^{i+1}(x)$连边，就能得到一个环。

对于原问题，设c_i中1的个数为a，2的个数为b。显然，问题的答案只与a和b有关。

我们假设问题中只有那a个元素，那么操作后的排列，所有环的大小最多为2。设f_a表示这样操作后，排列的方案数。对于f_a，可虑第a个元素，它要么独自成环，或者与先前的$a-1$个元素的某一个成环。因此可以得到以下递推式：

$$f_a = f_{a-1} + (a-1)f_{a-2}$$

现在考虑依次加入剩余的b个元素。显然这每个元素可以任意选择插入某一条边中，或者独自成环。因此对于第k个插入的元素，有k种插入方式。那么最终的答案为：

$$ans = f_a \cdot \prod_{i=a+1}^{a+b} i$$

复杂度分析

时间复杂度：$O(n)$。
空间复杂度：$O(1)$。

§30 Colorado Potato Beetle

试题来源 Codeforces 243C

题目大意

在一个无穷大的四连通的网格平面，有一条由n个垂直或水平的网格路径首尾相接组成的轨迹（每条路径长度不超过10^6），规定不能经过这一条轨迹上的任一网格。从某个无穷远的网格出发，问平面上有多少个网格不能被到达。

$1 \leq n \leq 1000$。

关键词

离散化 DFS

算法讨论

本题坐标范围能达到10^9级别，而路径的拐点很少，这告诉我们有用的坐标其实并不多。我们把所有路径的端点处的网格留住，其他网格及网格线擦去，重新沿着留下的网格画网格线。（这本质是一个离散化处理）这样我们的网格大小就变为$O(n) \times O(n)$的。

剩下的事情就很容易了，我们在新网格上将属于路径的格子打上标记，再从最外围进行DFS。最后将DFS中未被经过的网格所代表的面积记入答案即可。

复杂度分析

时间复杂度：$O(n^2)$。

空间复杂度：$O(n^2)$。

§31 Piglet's Birthday

试题来源 Codeforces 248E

题目大意

有n个架子，一开始每个架子有a_i个装满蜂蜜的罐子。现在执行q次操作，每次操作从u_i号架子上随机取走k_i个罐子并将它们都清空，放到v_i号架子。问每次操作后全是空罐子的架子的期望个数。

$1 \le n, q \le 10^5$，$0 \le a_i \le 100$，$k_i \le 5$，$1 \le u_i, v_i \le n$。

关键词

数学期望

算法讨论

这是一道比较基础的概率统计问题。我们用$p_x(i)$表示当前第x个架子还有i个满罐子的概率。显然对于每次操作，只可能改变$p_u(i)$的值。由于k很小最多为5，我们一个一个地拿走罐子，暴力重新计算所有$p_u(i)$的值即可。设当前u号架子罐子的数目为t，考虑每次取走的是否为空罐子，那么有：

$$p'_u(i) = \frac{t-i}{t} p_u(i) + \frac{i+1}{t} p_u(i+1)$$

复杂度分析

时间复杂度：$O(qak)$。

空间复杂度：$O(na)$

§32 Tennis Rackets

试题来源 Codeforces 309D

题目大意

在边长为$n+1$的正三角形每条边上等距的排列n个点，每条边头尾m个点不能用。问在每条边上各选一个可用点，能组成钝角三角形的方案数。

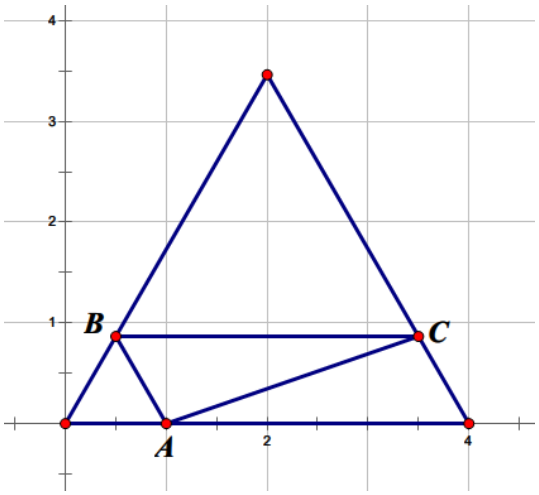
$1 \le n \le 32000$，$0 \le m \le \frac{n}{2}$。

关键词

枚举　计算几何

算法讨论

首先以三角形某个顶点及其所在的一条边为x轴建立坐标系（如下图所示）。



设$A(i,0), B(\frac{1}{2}j, \frac{\sqrt{3}}{2}j), C(L-\frac{1}{2}k, \frac{\sqrt{3}}{2}k)$，其中$L=n+1$，$m+1 \le i,j,k \le n-m$。不放设点$A$为钝角所在的顶点。那么我们有：

$$\begin{aligned} \overrightarrow{AB} \cdot \overrightarrow{AC} &< 0 \\ (\frac{j}{2}-i, \frac{\sqrt{3}}{2}j) \cdot (L-\frac{k}{2}-i, \frac{\sqrt{3}}{2}k) &< 0 \\ k &< \frac{(2i-j)(L-i)}{i+j} = (L-i)(\frac{3i}{i+j}-1) \end{aligned}$$

那么我们可以很容易得到一个枚举算法：从枚举i和j，$O(1)$计算符合题意的k的范围，最后再将答案乘3。实现过程中我用了以下几个常数优化

- i只需要从$m+1$枚举到$\lceil \frac{n}{2} \rceil$即可，因为$i$和$n+1-i$实际上是本质相同的。

- 从小到大枚举j的过程，k的范围只会越来越小。一旦k无解，就可以退出枚举j的循环。
- 解不等式时注意控制乘法以及除法次数。

通过这些常数优化，可以在CF上$1s$以内通过本题。

复杂度分析

时间复杂度：$O(n^2)$。
空间复杂度：$O(1)$。

§33 Two permutations

试题来源 Codeforces 323C

题目大意

给定两个n的排列，问第一个排列中元素位置在$[l_1, r_1]$的集合与第二个排列元素位置在$[l_2, r_2]$的集合的交集大小。询问有m组，强制在线。

$1 \le n \le 10^6$，$2 \le m \le 2 \cdot 10^5$。

关键词

函数式线段树

算法讨论

设v_i表示第二个排列位置i的数，在第一个排列中出现的位置。我们有：

$$query(l_1, r_1, l_2, r_2) = \sum_{i=l_2}^{r_2} [l_1 \le v_i \le r_1]$$

若询问允许离线，我们只需要按r_2从小到大排序，再用树状数组维护即可。由于询问强制在线，我们构建函数式线段树，第i棵线段树在第$i-1$棵线段树的基础上在位置v_i加一。对于询问，查找第l_2-1和第r_2棵线段树在区间$[l_1, r_1]$的和即可。

复杂度分析

时间复杂度：$O((m+n)\log n)$。
空间复杂度：$O(n \log n)$。

§34 The Evil Temple and the Moving Rocks

试题来源 Codeforces 329D

题目大意

你需要构造一个$n \times n$的地图，可以是空地，也可以是能往四个方向之一运动的石块。要求触发一个石块后，引发的后续碰撞不少于x次。

设$1 \le k \le 150$，$n = 2k$，$x \le \max(k^3 - k^2, k)$。

关键词

构造

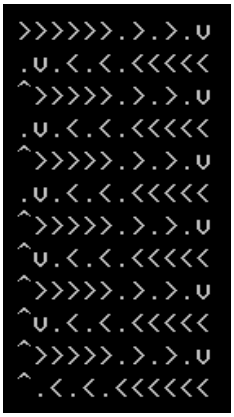
算法讨论

首先考察以下这种摆法，我们称之为碰撞结构：

$$\underbrace{>>\cdots>}_{a\text{个}>}\underbrace{>.>.\cdots>.}_{b\text{个}>.}$$

我们假设可以不断地触发最左边的石头，那么这样一串的摆法可以触发$(a+1)b$次碰撞。由于需要连续触发，每行需要留出两个空位来完成这种循环触发的结构，因此这种结构长度为$2k-2$。对于$a + 2b = 2k - 2$，当$a = 2\lfloor\frac{k-1}{2}\rfloor$，$b = \lfloor\frac{k}{2}\rfloor$时，可以最大化$(a+1)b = \frac{k(k-1)}{2}$。

而对于这种循环结构的构造，我们可以将第一列空出来，也摆放一种自下到上的碰撞结构，以沟通最后一行和第一行。沟通上下两行，我们可以蛇形地摆放这样的碰撞结构。下图是$k = 6$的情况的摆放方案：



这样一来，总共有$2k+1$个碰撞结构，每个的长度至少为$2k-2$，因此总次数不少于$\frac{1}{2}k(k-1)(2k+1) > k^3 - k^2$，符合要求。注意特判$n = 2$的情况。

复杂度分析

时间复杂度：$O(n^2)$。

空间复杂度：$O(n^2)$。

§35 Monsters and Diamonds

试题来源 Codeforces 325C

题目大意

有n种物品，m种变化规则。第i种变化规则可以将第m_i种物品，换成l_i样东西，其中一定包含一些钻石，也可能有其他物品。求出对于每个i，你有第i种物品一件，最多和最少能换到多少钻石（最终一定要只剩钻石）。

$1 \le n, m \le 10^5$，$\sum l_i \le 10^5$，每种物品至少对应一种变化规则。

关键词

贪心　记忆化搜索

算法讨论

对于第一问，求最小值。由于题目保证每个规则至少产生一颗钻石，我们可以使用一个类似Dijkstra算法的贪心。我们先将可以只变化成钻石的物品加入一个小根堆。每次取堆顶元素，累加入那些能变化出它的规则的val_i。其中val_i表示最开始执行第i个规则，最少能换到的钻石。若累加和发现val_i已经更新完毕了，那么我们将第i个规则对应的物品加入堆中，重复执行以上过程。这个过程执行完，未曾加入堆中的物品显然两问都是无解的。

对于第二问，我们采用记忆化搜索。假设我们现在要求第x个物品的最大值，我们枚举它对应的所有变化规则（当然这些规则变化出来的物品不允许包含无解的物品）。对于一种变化规则，递归搜索每种物品的最大值，加入本身。对于无穷大的判断，只需要检查这个规则中的某种物品是否已经在栈中了，如果是则说明找到了一种能从自己不断变出自己本身的方案。

复杂度分析

设$L = \sum l_i$。

时间复杂度：$O(L \log L)$。

空间复杂度：$O(L)$。

§36 Tape Programming

试题来源 Codeforces 238D

题目大意

给出一个长度为n的程序，有q次询问，每次询问将第l到r的子段截取出来执行，每种数字会被输出多少次。程序的执行流程如下：

1. 初始方向向右，初始指针指向程序第一个字符。

2. 若指针所指字符为'>'或'<'，则将方向置为此字符的方向。若上一个字符也是'>'或'<'，则将上一个删除。
3. 若为数字，将其输出。若这个数字是0则删除，否则将这个数字减一。
4. 指针向当前方向移动一步，若移动不合法则结束程序，否则跳到第二步。

$1 \leq n, q \leq 10^5$。

关键词

模拟　二分查找

算法讨论

首先根据题目性质分析可以知道，从头开始执行读入的程序，任何时刻指针移动到区间$[l, r]$，一定是最先在位置l，且方向是向右的。从这一步开始，就如同我们开始执行第l到r的子段一样；而指针某个时刻移出区间$[l, r]$，就相当于结束执行第l到r的子段。

有了以上的结论，我们只需要从头模拟程序运行的过程，记录每个时刻每个数字被输出的次数。同时每个位置维护一个指针运行到这个位置的时间集合。对于询问，只需要在$l-1$（注意实际上不一定$l-1$，而应该是第一次到达l前的那个位置）以及$r+1$两个位置对应的时间集合上二分找到最小的，且比第一次到达l的时刻大的时刻。

复杂度分析

时间复杂度：$O(n + q \log n)$。

空间复杂度：$O(n)$。

§37 Land Acquisition

试题来源　USACO Mar 08

题目大意

你有n个长方形土地，长宽分别为x_i和y_i。你可以将其分组，一组土地的价格是这组土地中最长的长和宽的乘积。求买下n个土地最小总费用。

$n \leq 50000$。

关键词

斜率优化

算法讨论

由题意可以发现，对于某个i和j，若$x_i \geq x_j$，$y_i \geq y_j$，那么j对答案无影响，因为你可以在买i的时候一起把j买下。这样，可以把这些无用的j全部去掉，那么我们可以得到一个x_i递减，y_i递增的土地序列。购买土地时，一定是

将这个序列的连续一段分在一组：比如将第l到第r个分在一组，那么这组的费用是$x_l \cdot y_r$，

这样我们用f_i表示这个序列中购买前i个土地的最小费用，显然我们有

$$f_i = \min_{1 \le j \le i} f_{j-1} + x_j \cdot y_i$$

状态数和转移都是$O(n)$。

接下来考虑优化。对转移方程移项处理：

$$f_{j-1} = -y_i \cdot x_j + f_i$$

f_i可以看成是一条过点(x_j, f_{j-1})，斜率为$-y_i$的直线在y轴上的截距。显而易见的是，能成为决策点的(x_j, f_{j-1})一定是在下凸壳上的，又由于$-y_i$是递减的，我们只需要用一个单调队列来维护这个凸壳（与常用的求凸包的Graham Scan算法类似）。

复杂度分析

时间复杂度：由于预处理有个排序的过程，复杂度为$O(n \log n)$。

空间复杂度：$O(n)$。

§38 Dima and Game

试题来源

Codeforces 273E

题目大意

你写下n对数字(l_i, r_i)（$1 \le l_i < r_i \le p$）。玩家轮流操作：每次选择一对数满足$r_i - l_i > 2$，将其替换为$(l_i + k, l_i + 2k)$或者$(l_i, r_i - k)$，其中$k = \lfloor \frac{r_i - l_i}{3} \rfloor$；直到不能操作者输。问你有多少种写法，使得这是一个先手必胜的游戏（顺序不同算不同的方案）。

$1 \le n \le 1000$，$1 \le p \le 10^9$。

关键词

组合游戏　SG函数

算法讨论

首先可以知道，对于$r_i - l_i$相同的数对，实际上是本质相同的。设$r_i - l_i = x$的子游戏的SG函数$sg(x)$，那么显然有：

$$sg(x) = mex\{sg(x - \lfloor \frac{x}{3} \rfloor), sg(\lfloor \frac{x}{3} \rfloor)\}$$

$sg(x)$的取值范围为$\{0,1,2\}$。

假设我们知道了所有数对中，SG函数值为0,1,2的个数分别为v_0, v_1, v_2。那么总方案我们可以用一个简单的动态规划求解。设$f_{i,j}$表示已写下i对数，异或和为j的方案数。转移方程为：

$$f_{i,j} = \sum_{k=0}^{2} f_{i-1,j\ xor\ k} \cdot v_k$$

现在的问题是$v0, v1, v2$该怎么算？

将$sg(x)$打印出来，可以发现这是一个有很长连续相同数字的数列。具体算出来，当$p = 10^9$时，这个数列的段数只有102段。因此，我们可以暴力在程序中处理出这些段，或者预先将这102个段打成一张表。那么$v0, v1, v2$就很容易算出了。

复杂度分析

设cnt_p等于上述所说的段数。

时间复杂度：$O(cnt_p + n)$。

空间复杂度：$O(cnt_p + n)$。

§39 Number Challenge

试题来源 Codeforces 235E

题目大意

设$d(x)$表示x的因数个数。求$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c} d(ijk)$。

$1 \le a, b, c \le 2000$。

关键词

数论

算法讨论

设$f(x)$表示有多少组i,j，满足$1\le i\le a,1\le j\le b,ij=x$。

$$\begin{aligned}
&\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c}d(ijk)\\
=&\sum_{i=1}^{ab}f(i)\sum_{j=1}^{c}d(ij)\\
=&\sum_{d>0}\sum_{i=1}^{ab}f(i)\left\lfloor\frac{c\gcd(i,d)}{d}\right\rfloor\\
=&\sum_{d>0}\sum_{i=1}^{ab}\sum_{g\mid d}[\gcd(d,i)=g]f(i)\left\lfloor\frac{cg}{d}\right\rfloor\\
=&\sum_{g>0}\sum_{d>0}\sum_{i=1}^{\lfloor ab/g\rfloor}[\gcd(d,i)=1]f(ig)\left\lfloor\frac{c}{d}\right\rfloor\\
=&\sum_{g>0}\sum_{e\mid g}\sum_{d>0}\sum_{i=1}^{\lfloor ab/g\rfloor}\mu(e)f(ig)\left\lfloor\frac{c}{ed}\right\rfloor\\
=&\sum_{g>0}\left(\sum_{i=1}^{\lfloor ab/g\rfloor}f(ig)\right)\left(\sum_{e\mid g}\mu(e)\sum_{d>0}\left\lfloor\frac{c}{ed}\right\rfloor\right)
\end{aligned}$$

两个括号内的式子都可以提前预处理出来。这样，我们枚举g，对于每个g快速算出对答案的贡献。

复杂度分析

时间复杂度：$O(ab\ln ab+c\ln c)$。
空间复杂度：$O(ab+c)$。

§40 Maximum Waterfall

试题来源 Codeforces 269D

题目大意

在平面上有n块水平的板，最上方有一个无限长的板提供水源。水自上而下流，你要找一条板组成的路径，使得留到最下方时流量最大。上下两块板可以连接，当且仅当它们的竖直方向的投影有重叠，且之间没有可以承接上下的板，那么流量为投影重叠的长度。

$1\le n\le 10^5$。

关键词

动态规划　数据结构

算法讨论

考虑使用动态规划。将板从低至高排序，用f_i表示从i开始往下流，最多能流多少。假设我们维护了一个集合，这个集合是前$i-1$块板所有从正上方可看到的线段。对于f_i的转移，只需要枚举那些与第i块板投影有重叠的线段，设其所属线段j，那么$f_i = \max(\min(f_j, Overlap(i,j)))$。

那么转移完f_i，由于这些决策点都被第i块板从上方挡住，可以把它们删除，继而加入线段i。加入这个集合中的总决策点数是$O(n)$级别的，这个集合可以用STL的set维护，也可以使用线段树。

复杂度分析

时间复杂度：$O(n \log n)$。
空间复杂度：$O(n)$。

§41 Marbles

试题来源　GCJ 2009 Final E

题目大意

有n种不同颜色的珠子，每种两个，在x轴上排列成一行。同种颜色的珠子之间可以从上方或者下方连线，线都在同一平面且不能相交。可以把线的宽度当做1，问上下宽度之和最小是多少。

$1 \le n \le 500$。

关键词

树形动态规划　二分图染色

算法讨论

我们先建立一个n个点的无向图，每个颜色对应一个点，如果两个颜色对应的横坐标区间有相交，则连一条边。那么我们对这张图进行二分图染色，即可判断问题是否有解。

接下来考虑问题有解，那么如何求最小总宽度呢？

我们先来看这张图连通块之间的关系。设L_i，R_i表示第i个连通块最左及最右端点的横坐标值。可以发现所有的区间$[L_i, R_i]$要么是相离，要么是包含关系。这是一个明显的树形结构。我们定义第j个连通块的父亲为i，当且仅当$L_i < L_j < R_j < R_i$，且$L_i = \max\{L_k \mid L_k < L_j < R_j < R_k\}$。

有了这个树形结构，我们就可以使用树形动态规划来解决此问题。设$f_{x,h}$表示第x个连通块为根的子树中，上方连线不超过h，下方连线最小宽度是多少。再用$cnt_{x,0}$表示，x的父亲的其中一种染色方案中，从上方完全覆盖连通块x的连线个数，$cnt_{x,1}$则表示从下方。那么转移方程如下：

$$f_{x,h} = \min\{\max\{f_{y,h-cnt_{y,c}} + cnt_{y,1-c} \mid father_y = x\} \mid c \in \{0,1\}\}$$

最后，对于一棵树的答案就是$\min\{f_{root,h} + h \mid 0 \le h \le n\}$。原问题的答案就是森林中每棵树答案的最大值。

复杂度分析

时间复杂度：$O(n^2)$。
空间复杂度：$O(n^2)$。

§42 Colorful Stones

试题来源

Codeforces 264D

题目大意

给定两个由RGB组成的字符串，各有一个指针（一开始指向第一个字符），长度分别为n和m。你可以给出若干命令，一个命令由一个字符组成：若某个指针的下一个位置是该字符，则移动到下一个位置。求两个指针能到达的状态数目。

$1 \le n, m \le 10^6$。

关键词

字符串匹配　双指针

算法讨论

根据题意分析，我们容易得到一个算法：枚举第一个字符串的每个位置i，用两个指针单调地维护第二个字符串最早和最晚能匹配到i的位置，再将这段区间的长度计入答案。

这看似是正确的，因为对于多数情况来说，最早和最晚位置之间的答案都是可以构造出来的。然而有一种情况会有问题。设x，y分别是第一个字符串第$i-1$和i个位置的字符，且x和y不相等。如果第二个字符串是形如'$\ldots yx$'的，那么可以看出这种情况是不合法的。那么我们只需要在计算答案时将区间内连续的'yx'个数扣除即可。

复杂度分析

时间复杂度：$O(n+m)$。

空间复杂度：$O(n+m)$。

§43 Donkey and Stars

试题来源 Codeforces 249D

题目大意

给定两个角度α_1和α_2，以及在第一象限的n个点。过原点做倾斜角为α_1和α_2的两条直线，在两条直线范围内选择一个新点，再以这个点为原点重复前面过程。问最多能选几个点。

$n \leq 10^5$。给定a, b, c, d，其中$\tan\alpha_1 = \frac{a}{b}$，$\tan\alpha_2 = \frac{c}{d}$，$0° \leq \alpha_1 < \alpha_2 \leq 90°$。

关键词

计算几何　最长上升子序列

算法讨论

设$\vec{x_0} = (b, a), \vec{y_0} = (d, c)$，以$\vec{x_0}, \vec{y_0}$为基向量重新表示每个点。具体来说，我们可以将点$(x, y)$，变换为$(cx - dy, by - ax)$。

这样，我们就将此问题转化为一个二维偏序问题，即二维的最长上升序列。这是一个经典问题，先按照横坐标排序，再对纵坐标维护一个最小结尾序列即可。最后的答案即为这个序列的长度。

复杂度分析

时间复杂度：$O(n \log n)$。

空间复杂度：$O(n)$。

§44 Paradox Sort

试题来源 GCJ 2014 Final D

题目大意

给出n个元素两两之间的偏序关系（不具有传递性）。对于一个排列，从前到后依次考察每个元素，若当前元素比手中的元素大，则用它替换手中的元素。求一个字典序最小的排列，使得最后手中的元素是A。

$1 \leq n \leq 100$。

关键词

贪心

算法讨论

首先先判断问题是否有解。对于任意两个元素i, j，若元素i大于元素j，则从i向j连一条有向边。从A开始进行DFS，如果能遍历所有的元素，则该问题有解，否则无解。如果将DFS过程中的退栈顺序输出，便是一个合法的解，正确性显然。

那么对于字典序最小的解，该如何构造呢？对于此类问题，常见的想法就是贪心。假设我们已经确定了答案的前$i-1$个位置，那么对于位置i，我们从小到大枚举未加入的元素，假设是B，直到加入B后存在解能够满足题意。不妨设前$i-1$个位置操作后手中的元素是M。检验合法性分两种情况考虑：

1. 若$M = A$，A比剩余未加入的元素都大即合法。
2. 若$M \neq A$，我们将前$i-1$个已确定的元素以及B从图中删去，从A开始DFS。若未被访问到的元素都比B小，那么合法。对于解的构造，可以先加入B，接着加入未被访问到的元素，最后是此图的DFS退栈顺序。这个构造的正确性也是显然的。

我们从前到后用如上方法确定每个位置，便能得到原问题的解。

复杂度分析

时间复杂度：有n个元素要加入，每次最多检验n个元素，一次检验中DFS复杂度为$O(n^2)$。因此本题理论最差复杂度为$O(n^4)$。而实际上数据一般达不到理论极限，因此可以通过本题。

空间复杂度：$O(n^2)$。

§45 Dividing Kingdom

试题来源

Codeforces 260E

题目大意

给定平面上有n个点，以及9个正整数$a_1, a_2, \ldots, a_9$。你需要给出平行于x轴和y轴的直线各两条，将平面分成9个部分，使得每部分的点数能和这9个数一一对应。

$9 \leq n \leq 10^5$，$\sum_{i=1}^{9} a_i = n$。

关键词

枚举　数据结构

算法讨论

由于我们不知道a_i应该和哪一块对应，因此需要枚举9的全排列。那么可以假设自下到上，自左到右的每个部分的个数与$a_1, a_2, \ldots, a_9$一一对应。我们很容易通过$a_1 + a_2 + a_3$确定第一条竖直的直线，以及$a_1 + a_2 + \ldots + a_6$确定

第二条。同理水平的直线也可以确定。剩下的事情就是检验九个部分是不是和a_i一一对应了。事实上只需要检查左下角的四个部分（个数分别为a_1，a_1+a_2，a_1+a_4，$a_1+a_2+a_4+a_5$的部分）是否满足即可。这个检查我使用了在线的数据结构函数式线段树，也可以离线处理后用树状数组。

复杂度分析

设$S=9!$。

时间复杂度：$O((n+S)\log n)$。

空间复杂度：$O(n\log n)$。

§46 Tourists

试题来源 Codeforces 286D

题目大意

有n对人，第i对人在q_i时刻分别从$(-1,0)$和$(1,0)$平行于y轴正方向行走，每单位时刻前进一单位长度。有m堵墙，第i堵墙在t_i时刻出现，是$(0,l_i)$到$(0,r_i)$的线段。问每对人之间被墙挡着的总时间。

$1\le n,m\le 10^5$，$0\le t_i,q_i,l_i,r_i\le 10^9$，$q_i$升序给出。

关键词

排序　并查集　扫描线

算法讨论

首先，我们先预处理每个位置最早出现墙的时间。具体做法是，我们先将墙的端点离散化，并按照出现时刻升序排序。对于每一堵墙，我们暴力枚举它的左端点到右端点，那么这些线段最早出现墙的时刻都已经得到了，然后用并查集把这一整段合并到一起。这样枚举的时候，就能快速找到最近一个未得到答案的位置。

假设预处理完后我们有若干不相交的区间$[l_i,r_i]$，它的出现时刻为t_i。那么当$t_i-r_i\le q_j\le t_i-l_i$，对第$j$对人的贡献为$q_j-(t_i-r_i)$；当$q_j>t_i-l_i$，对第$j$对人的贡献为$r_i-l_i$。有了以上的分析，我们只需要把一堵墙分为$t_i-r_i$及$t_i-l_i$两个事件点，排完序后利用扫描线的方法即可求出每个人的答案。

复杂度分析

时间复杂度：$O(m\log m)$。

空间复杂度：$O(m)$。

§47 Wall Bars

试题来源 Codeforces 268D

题目大意

有四根高度为n的柱子。你要在每个高度任选一根柱子放踏板。在同一根柱子上，高度差不超过h的踏板可以相互到达。从地面出发，可以踩上高度不超过h的踏板。问有多少种方案，使得从地面出发可以到达至少一个高度在$n-h+1$和n之间的踏板。

$1 \le n \le 1000$，$1 \le h \le \min(n, 30)$。

关键词

动态规划

算法讨论

对于此问题，考虑使用动态规划来解决。先定义一个踏板为有效的，当且仅当从地面能到达它。设$f_{i,a,b,c,j}$表示已经放置了前i个踏板，其中j是一个布尔变量，表示第i个是否有效，a,b,c则为另外三个柱子最高的踏板与i的高度差。显然高度差超过h那么此柱子之后的踏板都失效了，因此$1 \le a,b,c \le h$。

利用这个状态表示，来考虑$f_{i,a,b,c,j}$的转移。第$i+1$个踏板有四个位置可以放置：

- 若放在第i个踏板所在的柱子上：那么有:

$$f_{i+1,a+1,b+1,c+1,j} = f_{i,a,b,c,j}$$

- 若放在其它三个柱子上：不妨设放在高度差为a的柱子，那么有（若j为真，$d=1$，否则$d=h$）：

$$f_{i+1,b+1,c+1,d,a<h} = f_{i,a,b,c,j}$$

最后答案为所有$f_{n,a,b,c,true}$之和，以及所有满足$a+b+c<3h$的$f_{n,a,b,c,false}$之和。

复杂度分析

时间复杂度：$O(nh^3)$。

空间复杂度：$O(h^3)$，DP过程使用滚动数组。

§48 Meeting Her

试题来源 Codeforces 238E

题目大意

给定一张n个点m条边的有向图，边权为1。有k条公交线路，第i条从s_i驶向t_i，司机会随机地从s_i到t_i的最短路中挑一条行驶。问从S到T，最坏情况下最少乘公交车的次数。

$1 \leq n, k \leq 100$，$0 \leq m \leq n(n-1)$。

关键词

最短路

算法讨论

首先分析每一条公交路线，对于第i条公交路线，一个结点一定被经过当且仅当它是s_i到t_i最短路网上的割点，不妨称之为关键点。我们可以利用Floyd算法方便地预处理出这些关键点。显然若问题有解，那么S和T必须是关键点。

设f_i表示从i出发前往T最坏情况下最少乘车次数。f_i可以用类似Dijkstra的方法求解。初始令$f_T = 0$。每一轮枚举每一条公交线路，对于第i条公交线路，设g_x表示在这条公交线路上从x出发，目前的最优解。枚举x在此线路上所有可能的下一个点，那么$g_x = \min(\max\{g_y\}, f_x)$。最后对于这条线路上所有关键点，用$g_x + 1$更新$f_x$。直到某轮没有$f_x$被更新，就输出$f_S$，结束算法。

复杂度分析

时间复杂度：每轮至少确定一个f_x的最优解，最多n轮。计算一次g_x需要$O(n^2)$的时间。最坏复杂度为$O(kn^3)$。

空间复杂度：$O(n^2 + kn)$。

§49 Rats

试题来源 Codeforces 254D

题目大意

在一个$n \times m$的四连通棋盘中，格子为空地或障碍，有些空地有老鼠。你需要选定两个空地放置两枚炸弹，使得所有老鼠与较近的一枚炸弹距离都不超过d。

$4 \leq n, m \leq 1000$，$1 \leq d \leq 8$。

关键词

BFS　暴力

算法讨论

由于d最大只有8，因此与一个格子距离不超过d的格子最多有$2d^2 + 2d + 1 \leq 145$。这个性质可以作为本题突破口。

对于最左上的一只老鼠，BFS出与它距离不超过d的空地集合A，A中一定有一枚炸弹。于是我们枚举A中的每个位置放置一枚炸弹，做BFS，将其控制范围内的空地标记。BFS完之后，再选择一只未被第一枚炸弹控制的老鼠，BFS出与它距离不超过d的空地集合B。第二枚炸弹一定在B中。枚举B中的每个位置放置一枚炸弹，做BFS，最后检验是否所有老鼠都在控制范围。

复杂度分析

时间复杂度：每次BFS的时间复杂度都是$O(d^2)$，因此总复杂度为$O(d^6)$。

空间复杂度：$O(nm)$。

§50 Rhombus

试题来源 Codeforces 263E

题目大意

给出一个$n\times m$的二维数组a，及正整数k。对于满足$k\le x\le n-k+1$，$k\le y\le m-k+1$的x,y，定义函数$f(x,y)=\sum a_{i,j}\cdot\max(0,k-|i-x|-|j-y|)$。求出$f(x,y)$取到最大值的$x,y$。

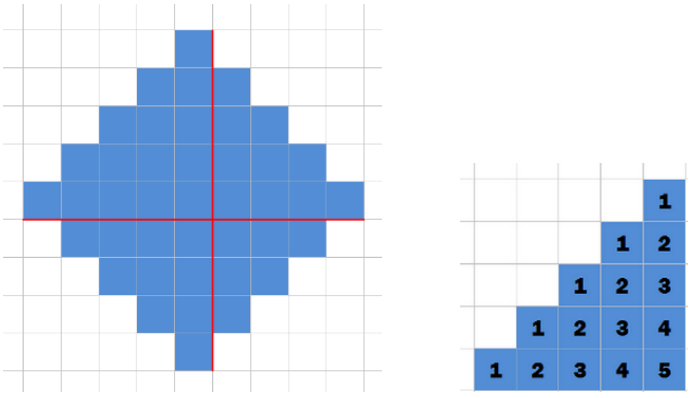
$1\le n,m\le 1000$，$1\le k\le\lfloor\frac{\min(n,m)+1}{2}\rfloor$。

关键词

部分和计算

算法讨论

由题意可知，数组中对$f(x,y)$有贡献的部分构成一个斜45度的正方形区域（如图所示），每个格子各有一个权重。



我们将其划分为四个直角三角形，划分方式如红线所示。以左上角的部分为例，单独考虑一个直角三角形，如何计算它的权值（如右上图片中，格子中标记了权重）。我们定义五个函数：

1. 对角线的和$s_1(x,y)=\sum_{i=0}^{k-1}a_{x-i,y+i}$。
2. 竖直段的和$s_2(x,y)=\sum_{i=0}^{k-1}a_{x-i,y}$。
3. 带权的竖直段的和$s_3(x,y)=\sum_{i=0}^{k-1}a_{x-i,y}(k-i)$。
4. 三角形的和$g_1(x,y)=\sum_{i=0}^{k-1}\sum_{j=0}^{k-1-i}a_{x-i,y-j}$。
5. 带权三角形的和$g_2(x,y)=\sum_{i=0}^{k-1}\sum_{j=0}^{k-1-i}a_{x-i,y-j}(k-i-j)$。

显然前三个都可以在$O(nm)$的时间预处理。而对于g_1,g_2的计算公式如下：

$$\begin{aligned} g_1(x,y+1) &= g_1(x,y)-s_1(x,y-k+1)+s_2(x,y+1) \\ g_2(x,y+1) &= g_2(x,y)-g_1(x,y)+s_3(x,y+1) \end{aligned}$$

对于其他三部分，方法与此情况本质相同。这样，原问题得以解决。

复杂度分析

时间复杂度：$O(nm)$。
空间复杂度：$O(nm)$。

§51 The Great Julya Calendar

试题来源 Codeforces 331C3

题目大意

给定整数n，每次操作可以将其减去其某一数位的数字。问最少操作几次，使它变成零。

$0\le n\le 10^{18}$。

关键词

数位DP 贪心

算法讨论

首先容易产生一个猜想，每次选择数位上最大的数去减。

如何证明呢？设f_x表示将x减为零的最小操作数。那么我们只需要证明$f_x\le f_{x+1}$恒成立。采用归纳法证明：

1. $f_0=0,f_1=1$，因此$f_0<f_1$。
2. 对于正整数k，假设$x<k$时结论成立。设h_x表示x数位上最大的数字。由于$h_x\ge h_{x+1}-1$，所以$x-h_x\le x+1-h_{x+1}$。因此$f_x=f_{x-h_x}+1\le f_{x+1-h_{x+1}}+1=f_{x+1}$。这就证明了当$x<k+1$是，$f_x\le f_{x+1}$成立。
3. 由1,2知，原命题成立。

有了这个结论，我们可以使用数位DP来求原问题的答案。用$dp_{i,j}$表示数字i，当前可以用的最大一位为j，减到零需要的操作数。转移的时候考虑将i的

最高位不断减小的过程即可。有效的状态中，i只可能是n的某段后缀，或者是$10^x - y(0 \le y \le 9)$的形式。

复杂度分析

时间复杂度：$O(\log n)$。
空间复杂度：$O(\log n)$。

§52 k-Maximum Subsequence Sum

试题来源 Codeforces 280D

题目大意

给定一个长度为n的整数序列a_i。要完成m次操作，操作有两种：将a_i修改为v；询问第l到第r个数组成的子序列中，最多k段的最大子段和。

$1 \le n, m \le 10^5$，$1 \le k \le 20$。

关键词

费用流　线段树

算法讨论

首先，我们建立一个最大费用最大流模型来解决此问题。设源点S，汇点T，对于每个询问，采取以下构图：

- 对所有i，i向i+1连一条容量为1，费用a_i的边；
- 对所有i，S向i连一条费用为0，容量为1的边；
- 对所有i，i向T连一条费用为0，容量为1的边。

使用费用流算法每次找到一条S到T费用最大的路径，然后增广，重复此过程k次便能得到问题的解。

仔细观察此图的特殊性，我们可以简化我们的算法：找到此序列的最大子段和，将这个子段的数取反，重复此过程k次。答案即为每次找到的最大子段之和。

这样一来，这些操作都可以使用线段树解决。线段树的每个节点需要维护以下四个值：区间和；包含左端点的最大子段和；包含右端点的最大子段和；此区间的最大子段和。由于还有取反操作，因此每个最大值对应要维护一个最小值。

复杂度分析

时间复杂度：每个修改操作的复杂度为$O(\log n)$，每个询问操作的复杂度为$O(k \log n)$。总复杂度为$O(n + mk \log n)$。

空间复杂度：$O(n + k)$。

§53 Figure Eight

试题来源 USACO Open 13

题目大意

在一个$n \times n$的有障碍棋盘上，寻找两个矩形构成数字8。要求矩形的边界不能有障碍，上下矩形必须紧邻且上矩形的底边必须是下矩形顶边的子集。一个数字8的分数定义为两个矩形面积之积，问最大分数是多少。

$n \leq 300$。

关键词

动态规划

算法讨论

对于本题容易想到使用动态规划解决。

首先预处理$up_{i,x,y}$，表示上矩形左右边界为x, y，在第i行，最多能往上延伸多少行。同时用$down_{i,x,y}$表示下矩形最多往下延伸多少行。

那么，我们枚举数字8中间那一行出现的位置，假设是在第i行。用$f_{x,y}$表示上矩形底边在x, y之间的最大面积，转移方程为：

$$f_{x,y} = \max(f_{x+1,y}, f_{x,y-1}, (up_{i,x,y} - 1)(y - x - 1))$$

计算f后枚举x, y，选取最大的$f_{x,y} \cdot (y - x - 1) \cdot (down_{i,x,y} - 1)$去更新答案。

复杂度分析

时间复杂度：预处理$up, down$复杂度为$O(n^3)$，枚举i后计算f以及枚举x, y复杂度都是$O(n^2)$。因此总复杂度为$O(n^3)$。

空间复杂度：$O(n^3)$。

§54 Hill Walk

试题来源 USACO Mar 13

题目大意

平面上给出n条互不相交，斜率为正的线段。一开始在第一条线段向右走，遇到端点处则垂直下落到第一条可以触及的线段，继续在线段向右走。问这个过程能到达的线段有多少条。

$1 \leq n \leq 10^5$。

关键词

扫描线　计算几何

算法讨论

考虑我们现在站在一条线段的右端点(x,y)。我们维护一个横坐标范围内包含x的线段的集合。由于线段两两无交点，因此我们可以给线段从高到低确定一个顺序，而这高低的比较则是由该线段在x处的纵坐标决定的。这样，我们从该集合中找到比当前线段低的最高线段是哪条，就确定了落下的位置。

转移到新线段上后，用扫描线的方法，先将那些已经无用的线段删除，再加入新的合法线段。同时注意到删除和加入后，集合中原有线段的偏序关系不会改变。因此这个集合可以用平衡树来方便维护。

复杂度分析

时间复杂度：$O(n\log n)$。
空间复杂度：$O(n)$。

§55 Safe Travel

试题来源 USACO Jan 09

题目大意

给定一张n个点m条边的无向图，保证以1为源点到其他点的最短路径唯一。对于2到n的每个点i，问若不经过最短路径的最后一条边，1到i的最短路是多少。

$n\le 10^5$，$m\le 2\cdot 10^5$。

关键词

最短路　路径压缩

算法讨论

先用Dijkstra算法求出1到i的最短路长度d_i。可知以1为源点所有在最短路上的边构成一棵$n-1$条边，以1为根的树。对于i的答案，我们可以枚举每条非树边(x,y)（其中x在i的子树中，y则不在），这条边对于i的解为$d_x+d_y+len_{x,y}-d_i$。

那么我们对于所有的非树边(x,y)，我们按照$d_x+d_y+len_{x,y}$从小到大依次枚举，来获得所有点的答案。边(x,y)可以更新$lca(x,y)$到x及$lca(x,y)$到y这两条链上的点。显然对于一个点，第一次得到的答案一定是最优的。因此，我们用过了边(x,y)，就可以用并查集将这两条路径压缩起来。

复杂度分析

时间复杂度：$O((n+m)\log m+m\alpha(n))$。
空间复杂度：$O(n+m)$。

§56 Min Perimeter

试题来源 GCJ 2009 Final B

题目大意

给定平面上n个点。问顶点在这n个点中的三角形，最小周长是多少。

$1 \le n \le 10^6$。

关键词

分治

算法讨论

对于本题，容易想到利用求平面最近点对的分治算法解决。我们先安装所有点横坐标排序。定义过程$solve(l,r)$表示求解第l到r个点的最小三角形周长。

对于$solve(l,r)$，我们将点集以中间那个点的横坐标分为两部分。那么对于答案在左半部分或者右半部分的情况，我们先调用$solve(l,mid)$及$solve(mid+1,r)$解决。剩下的，就是处理跨越两部分的情况。设中间竖直的分割线为$x=c$，当前最优解为d。那么显然我们只需要考虑横坐标在$[c-\frac{d}{2},c+\frac{d}{2}]$的点即可。而对于这些点，能够成为答案的点之间的纵坐标差也不会超过$\frac{d}{2}$。这样，我们只需要将这些点在分治时顺便按照纵坐标归并排序，之后用双指针维护一个$d\times\frac{d}{2}$的矩形内的点集。可以证明，这么一个点集中点的个数不会超过16个，因此可以暴力枚举。

复杂度分析

时间复杂度：根据主定理，可以计算得时间复杂度为$O(n\log n)$。

空间复杂度：$O(n)$。

§57 Flights

试题来源 Codeforces 241E

题目大意

给定n个点m条边的有向图，你需要把每条边的长度定为1或2，使得每一条从1到n的路径长度相等。

$1 \le n \le 1000$，$1 \le m \le 5000$。

关键词

差分约束系统

算法讨论

设d_i表示从起点出发到达i的路径长度。由题意可知，最终答案中d_i的值是恒定的。首先我们将边(a,b)分为两类。若边(a,b)不能沟通1和n，即1出发不能到达a，或者b出发不能到达n，那么(a,b)的长度可以任意选择。若边(a,b)能沟通1和n，那么满足$1 \le d_b - d_a \le 2$，即我们有$d_b \le d_a + 2$及$d_a \le d_b - 1$。对于第二类边，可以得出一组形如$d_x \le d_y + v$的不等式组。而这可以套用差分约束系统的模型去解决。

具体而言，对于第二类边，从a向b连一条权值为2的边，从b向a连一条权值为-1的边。从1开始用SPFA跑单源最短路，最后每条边的答案即为$dis_b - dis_a$。若图中存在负环，则原问题无解。

复杂度分析

时间复杂度：$O(nm)$。
空间复杂度：$O(n+m)$。

§58 Triangle Counting

试题来源 USACO Open 10

题目大意

给定平面上n个点，问顶点在这n个点中的所有三角形，内部包含原点的有多少个。

$1 \le n \le 10^5$。

关键词

双指针　极角排序　组合计数

算法讨论

首先容易知道，如果一些点都在某条过原点直线的一侧，那么这些点所组成的三角形都不包含原点。同时，每个不包含原点的三角形，都至少对应了这样的一条直线。

因此，我们可以统计所有不包含原点的三角形个数S，那么原问题的答案即为$\binom{n}{3} - S$。

怎么计算S？我们将每个点按照极角从小到大排序。我们依次枚举每个点i，然后统计不合法三角形中，以i为顶点，且i是极角最小的点有多少个。设i的极角为α，其余两个点的极角范围在$(a, a+\pi)$。假设j指向了这个范围内极角最大的点，那么i对答案的贡献为$\binom{j-i}{2}$。

复杂度分析

时间复杂度：由于i的极角在不断变大，因此j指针是单调向后移动的，这样计算S的时间复杂度为$O(n)$的。考虑到还有排序，总复杂度为$O(n \log n)$。

空间复杂度：$O(n)$。

§59 Berland Traffic

试题来源

Codeforces 267C

题目大意

给你一张n个点m条边的网络。你需要给每条边指定一个流量（不能超过其容量），并且满足两点之间任意路径的流量是固定的，同时满足1到n总流量最大。

$1 \le n \le 1000$，$1 \le m \le 5000$。

关键词

高斯消元

算法讨论

最终答案只要满足从起点到每个点任意路径的流量固定，就能满足题意。不妨设起点到每个点任意路径的流量为x_i，那么对于编号为2到$n-1$的点，能够用流量平衡列$n-2$个方程。对于1号点，$x_1 = 0$；对于n号点，总流入量为1，也就是假设最大流为1。可以使用高斯消元来解这个n元线性方程组。

解出了所有的x_i，此网络的最大流$f = \min_{(a,b)\in E} \frac{cap_{a,b}}{|x_a - x_b|}$。最后每条边的流量为$f \cdot (x_b - x_a)$。

复杂度分析

时间复杂度：$O(n^3 + m)$。

空间复杂度：$O(n^2 + m)$。

§60 Maxim and Increasing Subsequence

试题来源

Codeforces 261D

题目大意

给长度为n的序列a_i，问a重复t次以后的最长上升子序列。

数列中均为正整数，最大值为$maxb$。$1 \le maxb, n \le 10^5$，$1 \le t \le 10^9$，$n \cdot maxb \le 2 \cdot 10^7$。

关键词

动态规划

算法讨论

稍加分析可知，若$t \geq maxb$或者$t \geq n$，那么答案为a_i中出现的不同数的数目。剩下的情况，$t \leq \min(maxb, n)$。

我们可以考虑依次枚举这nt个数字，并用最小结尾序列的方法来处理这个问题。传统的方法中，我们需要在这个序列中二分查找第一个大于当前数字的位置。而利用此题的特殊性可知，a_i在这个序列中的位置是随着a不断重复而单调向后移动的。因此我们可以记录下a_i每一轮结束在最小结尾序列中的位置pos_i。下一轮遇到a_i，只需要从pos_i处开始往后扫描即可。

复杂度分析

时间复杂度：枚举每个数的复杂度为$O(n \cdot maxb)$。由于$pos_i \leq maxb$，因此扫描的复杂度也是$O(n \cdot maxb)$。总复杂度$O(n \cdot maxb)$。

空间复杂度：$O(n)$。

§61 Lucky Tickets

试题来源 Codeforces 333C

题目大意

对于一个八位数（允许有前导零），若在一些数字的左边或右边添加一些加减乘运算符号或者括号，使得结果等于k，那么称之为k-幸运数。你需要找出m个k-幸运数。

$m \leq 3 \cdot 10^5$，$0 \leq k \leq 10^4$。

关键词

构造

算法讨论

考虑枚举这个八位数的后四位数字，然后搜索这四位数字添加运算符和括号后能得到的结果。再将这个结果与这四位数字连接。例如$k = 3000$,枚举的四个数字为2567，它能得到的结果之一为$(2+5) \times 6 - 7 = 35$，$3000 - 35 = 2965$，那么我们就生成了两组解29652567以及25672965，显然它们符合题意。

搜索的时候并不需要考虑过于复杂的情况，只用单独搜索每个数位与前面数位的运算结果的运算关系是加减或乘即可。这种构造策略，对于每个k大约都能生成3.5×10^5组解，可以通过本题。

复杂度分析

时间复杂度：$O(10000 \cdot 3^4 + m)$=$O(m)$。
空间复杂度：$O(m)$。

§62 Best Cow Line

试题来源 USACO Dec 07

题目大意

给一个长度为n的字符串，每次可以从串首或者串末拿走一个字符，加入新串的末尾。求字典序最小的新串。

$1 \leq n \leq 30000$。

关键词

哈希

算法讨论

对于本题，容易想到贪心的做法。设当前剩余的字符串为$S[a \ldots b]$。若$S[a] < S[b]$则先选择$S[a]$；若$S[b] > S[a]$则先选择$S[b]$。这个贪心的正确性是显然的。不过要是$S[a] = S[b]$，那么我们需要更深地检查$S[a+1]$与$S[b-1]$，$S[a+2]$与$S[b-2]$... 换句话说，我们需要比较$S[a \ldots b]$与$rev(S[a \ldots b])$的字典序大小。若$S[a \ldots b] < rev(S[a \ldots b])$，则先选择$S[a]$，否则选择$S[b]$。而对于字典序的比较，可以利用二分加哈希判断的方法解决。

复杂度分析

时间复杂度：$O(n \log n)$。
空间复杂度：$O(n)$。

§63 Distinct Paths

试题来源 Codeforces 293B

题目大意

给定一个$n \times m$的棋盘，一些格子已经涂上了k种颜色中的一种。你需要给剩余格子从k种颜色选一种上色，使得左上角到右下角的所有路径不会经过两块同色的格子。求方案数。

$1 \leq n, m \leq 1000$，$1 \leq k \leq 10$。

关键词

搜索剪枝

算法讨论

对于本题，若$n+m-1>k$，则方案数为0。因此实际上我们需要面对的问题规模是很小的，$n+m$不超过11。不妨考虑使用搜索解决此问题。搜索中的几个优化有：

- 对于格子(x,y)，判断颜色c是否可行可以使用位运算快速判断之前的格子是否使用过，并且预处理(x,y)以下的部分是否出现了颜色c。
- 对于未在原棋盘中出现的颜色，事实上我们不需要关心最终棋盘上用了哪些这样的颜色，而只要关心用了几种。因此，我们在搜索中只需要枚举其出现的颜色总数x，再将x对应的排列数统计入答案。

加上这些剪枝，所有数据都可以在150毫秒以内出解。

复杂度分析

时间复杂度：最坏情况可达$O((nm)^k)$。

空间复杂度：$O(k^2)$。

§64 Cows in a Skyscraper

试题来源 USACO Mar 12

题目大意

给出n个不超过W的正整数，要将这些数分成若干组使得每组的和不超过W。问最少需要分成多少组。

$1\le n\le 18$，$1\le W\le 10^8$。

关键词

状态压缩动态规划

算法讨论

对于此题n很小，可以考虑使用状态压缩动态规划解决。设$dp[S]$是一个二元组，表示取完集合S（S是一个二进制状态），最少分了多少组，在此前提下最后一组使用的空间最少是多少。对于转移，我们枚举所有不在S中的数i，方程如下：

$$dp[S\cup i]=\min\begin{cases}(dp[S].cnt,dp[S].used+a_i) & a_i+dp[S].used\le W\\ (dp[S].cnt+1,a_i) & \text{otherwise}\end{cases}$$

最后的答案即为$dp[2^n-1].cnt$。

复杂度分析

时间复杂度：$O(2^n n)$。

空间复杂度：$O(2^n)$。

§65 Polygon

试题来源 Codeforces 306D

题目大意

你需要给出一个n边形每个顶点的坐标，满足这个n边形每个内角和相等且每条边不等长。

$3 \leq n \leq 100$。

关键词

构造

算法讨论

首先容易知道，当$n \leq 4$时问题无解。对于$n > 4$的情况，下面给出一种构造方法。

设$A = \frac{2\pi}{n}$，$L = 500$，$\vec{v} = (L\cos A, L\sin A)$。以原点为第一个点，每次沿着向量$\vec{v}$从当前点找到下一个点。再把$\vec{v}$的模长减少0.01，同时向逆时针旋转$A$，继续寻找下一个点。找到了$n-1$个点后，对于最后一个点位置的确定，我们从第$n-1$个点沿着$\vec{v}$的方向，找到与x轴的交点即可。

容易证明，这样的多边形每个内角相等，并且经过实践验证，每条边的长度也不相等。

复杂度分析

时间复杂度：$O(n)$。

空间复杂度：$O(1)$。

§66 Cleaning Up

试题来源 USACO Mar 09

题目大意

给一个长度为n的数组，你可以把它分成若干连续的段。对于每一段的分数为这一段内不同数字个数的平方。求能得到的最小总分数。

$1 \leq n \leq 40000$。

关键词

动态规划

算法讨论

这个问题容易想到使用动态规划求解。设f_i表示对前i个数进行划分，能得到的最小的分数。转移方程为$f_i = \min_{1 \le j \le i} f_{j-1} + val(j,i)$。直接进行动态规划的时间复杂度至少为$O(n^2)$，无法通过本题。

注意到若我们把每个数单独分一组，那么$f_n = n$。因此对于$val(j,i) \ge n$的情况，我们不需要考虑转移。换句话说，只需要考虑不同数字个数不超过$\sqrt{n}$的段即可。这样，我们可以维护pos_x，表示满足第j个数到第i个数不同数字个数为x的最小的j，其中$1 \le x \le \sqrt{n}$。那么转移方程为$f_i = \min_{1 \le x \le \sqrt{i}} f_{pos_x - 1} + x^2$。

复杂度分析

时间复杂度：$O(n^{1.5})$。

空间复杂度：$O(n)$。

§67 Little Elephant and Broken Sorting

试题来源 Codeforces 258D

题目大意

给出一个n的排列a，之后要执行m次操作。每次操作给出x, y，有0.5的概率交换a_x和a_y。问m次操作后排列的期望逆序对数目是多少。

$1 \le n, m \le 1000$。

关键词

数学期望

算法讨论

我们用$p_{i,j}$表示$a_i > a_j$的概率。初始时，若$a_i > a_j$则$p_{i,j} = 1$，否则$p_{i,j} = 0$。

每次操作之后，设新的概率矩阵为p'。显然有$p'_{x,y} = p'_{y,x} = 0.5$。对于所有的$i$（$i \ne x$且$i \ne y$），$p'_{i,x} = p'_{i,y} = \frac{1}{2}(p_{i,x} + p_{i,y}), p'_{x,i} = p'_{y,i} = \frac{1}{2}(p_{x,i} + p_{y,i})$。对于其它位置，$p'_{i,j} = p_{i,j}$。

最后根据期望的线性性质，可以得到答案为$\sum_{i<j} p_{i,j}$。

复杂度分析

时间复杂度：$O(n(m+n))$。

空间复杂度：$O(n^2)$。

§68 Runs

试题来源 GCJ 2011 Final A

题目大意

给出一个长度为n的字符串，设其有m段连续极大相同的子串。问对这个字符串重排，有多少个不同的字符串满足连续极大相同的子串段数等于m，输出方案数模1000003。

$1 \leq n \leq 450000$，$1 \leq m \leq 100$。

关键词

动态规划　组合计数

算法讨论

首先先预处理出m的值，以及第i种字符的个数c_i。串长很大，而m和字符总数很小，因此可以利用此性质考虑动态规划。

为了方便描述，下文将连续极大相同的子串段简称为段。我们不妨考虑一类一类地加入字符。设$f_{i,j}$表示前i个字符组成的字符串，段数为j的方案数。设$T = \sum_{i=1}^{i} c_i$。对于第$i+1$类字符，考虑在字符串中加入一段的情况：

1. 如果在原有的段与段之间插入，那么新增段数为1，有$j+1$个位置可以选择。
2. 如果在两个相同字符之间插入，那么新增段数为2，有$T-j$个位置可以选择。

那么假设第一类加入了j段，第二类加入了l段，那么插入位置的方案数分别为$\binom{j+1}{k}$和$\binom{T-j}{l}$。而对于c_{i+1}个第$i+1$类字符，有$\binom{c_{i+1}-1}{k+l-1}$种划分方式，分成$k+l$组。这样就可以得到状态转移方程：

$$f_{i,j+k+2l} = \sum \binom{j+1}{k} \binom{T-j}{l} \binom{c_{i+1}-1}{k+l-1} f_{i,j}$$

设字符集大小为S，最终答案即为$f_{S,m}$。

转移方程中的组合数可以在之前用$O(n)$时间预处理阶乘及逆元。

复杂度分析

时间复杂度：$O(n + Sm^3)$。

空间复杂度：$O(Sm + n)$。

§69 Photo

试题来源　USACO Open 13

题目大意

有n个01变量x_i。给出m个区间$[a_i, b_i]$，表示$\sum_{j=a_i}^{b_i} x_j = 1$。求满足条件的$\sum_{i=1}^{n} x_i$的最大值。

$1 \le n \le 2 \cdot 10^5$，$1 \le m \le 10^5$。

关键词

动态规划　单调队列

算法讨论

考虑使用动态规划解决本题。设f_i表示$\sum_{j=1}^{i} x_j$的最大值是多少，且$x_i = 1$。用$[L_i, R_i]$表示上一个1可以出现的区间。其中$L_i = \max\{a_j | b_j < i\}$，$R_i = \min\{a_j - 1 | a_j \le i \le b_j\}$，很容易做到在$O(n+m)$的时间内预处理出$L_i, R_i$。显然，若$L_i > R_i$，则$x_i = 0$。那么转移方程为：$f_i = \max_{L_i \le j \le R_i} f_j + 1$。

这样直接转移的时间复杂度最坏为$O(n^2)$的。注意到L_i和R_i都是随着i增加而单调不降。因此可以维护一个j递增，f_j递减的单调队列，从而将复杂度降为线性。

复杂度分析

时间复杂度：$O(n+m)$。

空间复杂度：$O(n)$。

§70　Suns and Rays

试题来源　Codeforces 316F3

题目大意

给出一个$n \times m$的01矩阵，描述一幅画的每个像素。画中有若干个太阳，太阳的椭圆圆周上连接着若干光束（光束宽度为3个像素）。你需要统计有多少个太阳，以及每个太阳有多少条光束。

$1 \le n, m \le 1600$。

关键词

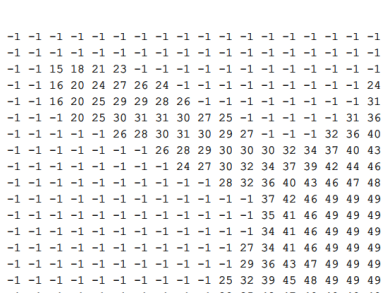
图像识别

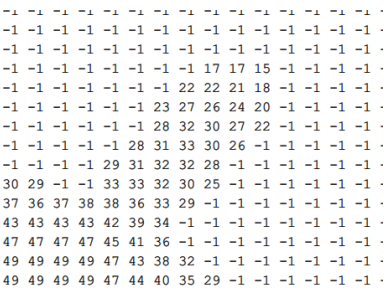
算法讨论

对于第一问，用DFS或BFS统计由1构成的四连通的连通块个数，即可知道太阳的个数。

对于第二问，有两种方法可以参考。

第一种方法的主要想法是识别光束的端点。以每个像素为中心取一个7×7的正方形，统计其内部的1的个数，我们称之为像素的特征值。通过几组样例数据可以发现，每条光束端点处都存在且仅存在一个像素的特征值不超过20的四连通块（如下图所示）。因此只需要把这些连通块都找出来，并对每个太阳统计其对应了多少个即可。





第二种方法的想法是将属于光束的1区分出来。由于光束宽度不超过3，对于所有与最近的0曼哈顿距离不超过3的1，我们先将其擦掉变为0。这样，所有的光束部分都被消除了，图中只剩下一些被缩小的椭圆。为了识别被消除的部分属于哪个太阳，我们把所有椭圆重新向外扩3个像素，再判断每个光束与哪个椭圆接壤即可。

复杂度分析

时间复杂度：两种方法都是$O(nm)$。

空间复杂度：两种方法都是$O(nm)$。

§71 Cow Schul

试题来源 USACO Jan 07

题目大意

给定n个分数$\frac{T_i}{P_i}$，分数值两两不同。定义一个分数集合的权值为$\frac{\sum T_i}{\sum P_i}$。现在你要给出所有$d$的值，满足存在去掉其中$d$个分数后的集合权值，比去掉分数值最小的$d$个分数后的集合权值大。

$1 \le n \le 50000$。

关键词

分数规划　斜率优化

算法讨论

首先先按照$\frac{T_i}{P_i}$升序排序。考虑对于某个d，怎么判断d能否成为解？记$m = \frac{\sum_{i=d+1}^{n} T_i}{\sum_{i=d+1}^{n} P_i}$，$v_i = T_i - m \cdot P_i$。显然有$\sum_{i=d+1}^{n} v_i = 0$。找一个大小为$n-d$的集合满足$\frac{\sum T_i}{\sum P_i} > m$等价于找一个大小为$n-d$的集合满足$\sum v_i > 0$。也就是说，只需要存在一对$x, y$，满足$1 \le x \le d < y \le n$且$v_x > v_y$，那么$v_x - v_y + \sum_{i=d+1}^{n} v_i > 0$，$d$就能成为答案。更进一步，我们只要求出$v_x$的最大值以及$v_y$的最小值即可。

下面，问题的重点，就成为求v_y的最小值。由于$T_i = m \cdot P_i + v_i$，v_i可以看做是平面上斜率为m，过点(P_i, T_i)的直线的截距。设$(P_{d+1}, T_{d+1}), (P_{d+2}, T_{d+2}), \ldots$

,(P_n, T_n)构成的下凸壳为H_d。显然能成为决策的点，一定是在H_d上。那么，如何从H_{d+1}得到H_d呢？这是一个只有加入点的动态凸包问题，可以使用平衡树来维护。对于求v_x的最大值，解决方法与上述类似，这里就不再赘述。

复杂度分析

时间复杂度：$O(n \log n)$。
空间复杂度：$O(n)$。

§72 The Red Button

试题来源 Codeforces 325E

题目大意

给定正整数n，你需要生成一个序列，序列的首尾必须是0，1到$n-1$必须各出现恰好一次。对于某个数i，i后面的数必须是$2i \bmod n$或者$2i+1 \bmod n$。

$1 \leq n \leq 10^5$。

关键词

构造

算法讨论

首先考虑当n为奇数的情况。对于最后一个0以及$n-1$，它们的前驱都必须是$\frac{n-1}{2}$。因此n为奇数的时候是无解的。

下面考虑n为偶数时，如何构造解？注意到对于任意$x(0 \leq x < \frac{n}{2})$，$x$和$x+\frac{n}{2}$的两个合法后继都是$2x$和$2x+1$，并且$2x$和$2x+1$的前驱只能是$x$和$x+\frac{n}{2}$。我们不妨先对$2x$和$2x+1$任意匹配一个前驱。若把序列中相邻的两个点连一条边，那么这样会构成若干个环。可以证明，若存在多于1个环，必然存在某个x，x和$x+\frac{n}{2}$在两个不同的环中。而对于x和$x+\frac{n}{2}$在不同的环，只需要交换他们的后继节点即可将两个环合并。至此，我们就找到了一种将所有环合并成一个环的方法。对于判断是否属于同一个环以及环的合并可以使用并查集。

复杂度分析

时间复杂度：$O(n\alpha(n))$。
空间复杂度：$O(n)$。

§73 Biologist

试题来源 Codeforces 311E

题目大意

有n个01变量，初始值x_i给定，改变x_i的代价为v_i。有m组限制，第i组限制要求k_i个变量同为c_i，若能满足该限制则能收益w_i。问最大的获益是多少。

$1 \le n \le 10^4$，$1 \le m \le 2000$，$1 \le k_i \le 10$。

关键词

最小割

算法讨论

本题可以看做尽量减小从总收益损失的代价，容易想到使用最小割模型解决这道题。

对于变量，如果$x_i = 0$，令$i \in S$，否则$i \in T$。那么对应的连边方法为：若$x_i = 0$，则从源点向变量i连一条容量为v_i的边，否则从变量i向汇点连一条容量为v_i的边。

对于限制，我们对每组限制建立一个点。同样，若$c_i = 0$，则从源点向限制i连一条容量为w_i的边，否则从限制i向汇点连一条容量为w_i的边。对于k_i个变量，分别向限制i连一条容量无穷大的边，表示要么改变变量的值来满足限制i，要么不改变变量的值。

这样，对这张图求出最大流，最后用总收益减去最大流即为答案。

复杂度分析

时间复杂度：$O((n+m)^2(km+n))$。

空间复杂度：$O(km+n)$。

§74 Fetch the Treasure

试题来源 Codeforces 311E

题目大意

有一个长度为h的数轴，给定n个宝藏在数轴上的坐标a_i，以及价值c_i。再有一个步长序列d_i，你可以到达任意坐标为$1 + \sum v_i \cdot d_i$（v_i为非负整数）的宝藏。初始时此集合只有一个数k。再有m个操作，操作有3种：在步长序列中添加一个数；减小某个c_i；询问能获得的最大价值的宝藏是多少。宝藏一经获得就会消失。

$1 \le n, m \le 10^5$，$1 \le a_i, d_i \le h \le 10^{18}$，$1 \le k \le 10^4$。第一类添加操作不超过20次。

关键词

最短路　堆

算法讨论

注意到h很大，而k的值很小，这可以作为本题的突破口。将坐标按照模k的值分为k类。设f_i表示能到达的第i类坐标的集合中，最小的坐标是多少。坐标f_i可以到达，那么也就意味着$f_i + p \cdot k$（p为非负整数）都可以到达。换而言之，若$a_i \geq f_{a_i \bmod k}$，则第i个宝藏是可以获得的。f_i的转移方程为：$f_i = \min\{f_{i-d_j \bmod k} + d_j\}$。这个转移可以利用最短路算法实现。

现在面对m个操作，可以用一个堆来辅助。每类操作如下处理：

- 第一类操作最多只有20个，因此可以每次暴力更新f_i的值。同时将新可达的点加入堆中。
- 第二类操作，只需要直接修改，再把它加入堆中（如果该宝藏可以到达）。
- 第三类操作，找到堆顶元素并将其删除。

复杂度分析

设第一类操作总数为S。

时间复杂度：$O(Sk \log k + m \log n)$。

空间复杂度：$O(k + n + m)$。

§75 Bottleneck

试题来源 USACO Jan 11

题目大意

有n个牛棚，形成一个以1号牛棚为根的有根树。对于每个点i，父亲为p_i，i向父亲的道路流量为m_i，初始时有c_i头牛。每个时刻，一头牛可以向根走任意距离，只要不超过流量限制，或者选择在当前牛棚不动。给出k个询问，每次询问在时间T内，最多有多少头牛可以到达根。

$1 \leq n \leq 10^5$，$1 \leq k \leq 10^4$，$1 \leq T, c_i, m_i \leq 10^9$。

关键词

贪心 堆

算法讨论

对于此题，容易得出一个贪心算法：每个时刻每个牛棚都往祖先运输尽量多的牛。直接实现这个算法复杂度是$O(nT)$的，无法通过。

可以注意到：每个点每个时刻流入流量是随着时间增加单调不增的；并且，每个牛棚的牛数量是一个关于时间的线性函数。当这个数量变为0时，这个牛棚的流入与流出将会相等，也就是说可以忽略这个牛棚了。这启示我们使用一个小根堆来维护每个牛棚牛数量变为零的时刻。对于当前堆顶元素，将其删除，并使用并查集实现路径压缩。同时修改与它最近的祖先的流入量in_i与

当前牛棚内牛的数量c_i。对于所有时刻小于下一个堆顶元素的询问，答案即为$T \cdot in_1 + c_1$。

复杂度分析

时间复杂度：$O(k \log k + n \log n)$。
空间复杂度：$O(k + n)$。

§76 Toys

试题来源 USACO Nov 08

题目大意

有n天，每天需要提供a_i个玩具。获得玩具可以以每个c元买，也可以使用之前的并经过清洗的玩具。清洗有两种，分别需要n_1和n_2天，每洗一件分别要c_1和c_2元。

$1 \leq n \leq 10^5$，$1 \leq a_i \leq 50$。

关键词

贪心　费用流

算法讨论

对于此题，经典做法是使用最小费用最大流模型解决。但是本题数据规模过大，不得不考虑使用其他算法。

首先约定，$n_1 \leq n_2$。那么如果$c1 < c2$，我们可以只使用第1种清洗，为了方便可以将c_2赋值为c_1。因此我们就有了$n_1 \leq n_2$，$c_1 \geq c_2$。

对于题目中买玩具的行为，我们可以在第一天之前就将所需的玩具买下。设一开始买x个玩具，最小的花费为$f(x)$。有了一个x，如何求$f(x)$呢？可以使用贪心解决，策略如下：如果有未使用过的则尽量使用；如果不够，先考虑已经存放至少n_2天的玩具，最后再考虑使用存放至少n_1天的玩具。此过程中发现某天无法满足，那么令$f(x) = +\infty$。

这样，我们得出了一个新算法：枚举每个x，取$f(x)$的最小值作为答案。不过这样时间复杂度为$O(an^2)$，仍无法通过。再考虑之前所提的费用流，这个费用流是以流量表示玩具数的。假设每次只增广1单位的流量，所需要的费用是不减小的。换句话说，$f(x+1) - f(x)$的值是随着x增加而不降的。这就说明$f(x)$是个凹函数，可以三分x的值，从而获得$f(x)$的最小值。

复杂度分析

时间复杂度：$O(n \log n)$。
空间复杂度：$O(n)$。

§77 Numbers

试题来源 Codeforces 241D

题目大意

给一个n的排列以及质数p。求一个子序列，使得序列中元素异或和为0，且序列中的数按顺序首尾相接组成的数是p的倍数。

$1 \leq n, p \leq 50000$。

关键词

构造　动态规划

算法讨论

如果我们只使用排列中小于32的数构造答案，那么我们可选的子集共有2^{31}个，这其中有$\frac{2^{31}}{32} = 67108864$个子集的异或和为0。在十进制下，对于大多数质数$p$（$p \notin \{2, 5\}$），这67108864个子集对应的数模$p$的结果可以视作随机的（毕竟难以想象异或和为0与连接起来模p为0有什么相关性）。而p最多只有50000，因此找到解的概率是极高的。而对于$p = 2$或$p = 5$的情况，显然只要$n \geq p$就一定有解，否则无解。

那么数字个数的规模被我们减小到了$S = \min(n, 31)$。考虑使用动态规划构造解。设$f_{i,j,k}$表示前i个数，是否存在异或和为j，连接起来的数模p的值为k的子序列。利用这个状态进行转移即可。

复杂度分析

时间复杂度：$O(n + S^2 p)$。
空间复杂度：$O(n + S^2 p)$。

§78 Friends

试题来源 Codeforces 241B

题目大意

给出n个正整数a_i，你要选择m对不同的数，使得m对数的$a_i \oplus a_j$之和最大。

$n \leq 50000$，$m \leq \binom{n}{2}$。

关键词

Trie树　数位统计

算法讨论

设所有的$a_i \oplus a_j$第m大的值为p。不妨考虑按照二进制从高位到低位逐位确定p的值。首先，将所有a_i转化成二进制建立一颗Trie树。从高位到低位枚举每一位i，这时p中比i高的位都已经确定了。用数位统计的想法，线性枚举每个a_i，统计有多少个$a_i \oplus a_j$满足第i位为1，比i高的位与p相同。判断这个数量是否超过m即可确定p的第i位。这其中需要预处理Trie中每个子树中a_i的个数。

求这m对数的异或和，可以在确定p的过程顺便统计。对于每位i，若确定了p的第i位为0，那么需要统计异或值第i位为1，比i高的位与p相同的和。同样线性枚举每个a_i，再枚举每个低于i的数位，根据子树中第i位为0和1的数的个数，即可累计对答案的贡献。若每个节点直接维护此信息，空间复杂度为$O(ns^2)$（其中$s = \max\{\log_2 a_i\}$）。由于a_i是已经给定的，将a_i排序后建立Trie树，那么每个子树都对应着一段连续的区间。这样只需要记录Trie树上对应区间的左右端点，并对a_i每个位预处理前缀和，这样空间复杂度就降为$O(ns)$。

复杂度分析

时间复杂度：$O(ns^2)$。
空间复杂度：$O(ns)$。

§79 Have You Ever Heard About the Word?

试题来源 Codeforces 319D

题目大意

称字符串S为一个重复串，当且仅当存在字符串T，满足$S = T + T$。给定长度为n的字符串，每次删除最短重复串中最靠左的前半部分，直到不能删除为止。求最后的串。

$n \leq 50000$。

关键词

暴力　哈希

算法讨论

考虑这么一个算法：从小到大枚举重复串的循环节长度i，若存在重复串的循环节长度为i，则在$O(n)$时间内删除所有的串并构出新字符串。设不同重复串长度有m个，设为$l_1, l_2, \ldots, l_m$。又$\sum_{i=1}^{m} l_i \leq n$，所以$m$的大小是$O(\sqrt{n})$的。这样，删除和重构操作的总复杂度$O(n^{1.5})$。

那么，如何判断是否存在循环节长度为i的重复串呢？这个问题的经典做法是在字符串每隔i的长度设置一个断点。那么一个循环节长度为i的重复串比然经过两个断点。因此只需要枚举每对相邻的断点，求出这两个断点的最长公共前缀和后缀，即可判断是否存在经过这对断点的重复串了。而对于最长公共前缀和后缀，可以利用二分加哈希的方法。这样，对于每个i，复杂度为$O(\frac{n}{i}\log n)$。因此判断的总复杂度为$O(n\log^2 n)$。

复杂度分析

时间复杂度：$O(n\log^2 n + n^{1.5})$。
空间复杂度：$O(n)$。

§80 Positions in Permutations

试题来源

Codeforces 285E

题目大意

给定整数n和k。问所有n的排列，有多少个排列p，使得恰好存在k个i，满足$|p_i - i| = 1$。求这个数目对$10^9 + 7$取模的结果。

$k \le n \le 10^3$。

关键词

动态规划　组合计数　容斥原理

算法讨论

为了方便，我们称满足$|p_i - i| = 1$的位置i为好位置。首先，设f_i表示n的排列中，至少存在i个好位置的排列个数，g_i则表示恰好有i个好位置的排列个数。假设f_i已知，我们要求的是g_i。利用容斥原理，容易得到以下递推式：

$$g_i = f_i - \sum_{j=i+1}^{n} \binom{j}{i} g_j$$

这部分预处理组合数以及求g_i的复杂度都可以做到$O(n^2)$。

那么剩下的问题就是求f_i。考虑用动态规划来辅助求解。设$dp_{i,j,S}$表示排列的前i个位置中，选定了j个好位置填入数字的方案数，其中S是一个两位的二进制状态表示数i和$i+1$是否已经被填入。从状态$dp_{i,j,S}$转移到$dp_{i+1,j',S'}$时，只需要考虑$i+1$这个位置不填数，填i或者$i+1$三种情况。那么有：

$$f_i = (n-j)! \sum dp_{n,j,S}$$

至此，问题就得以解决了。

复杂度分析

时间复杂度：$O(n^2)$。
空间复杂度：$O(n^2)$。

§81 Ping-Pong

试题来源 Codeforces 319E

题目大意

定义两个区间x和y，若x的左端点或右端点严格在区间y的内部，那么就有一条x到y的有向边。给出n个操作，每次新增一个区间（新增区间的长度严格递增），或者给出之前加入的某两个区间x和y，问x是否能到达y。

$n \leq 10^5$。

关键词

并查集　线段树

算法讨论

首先根据题意可知，如果两个区间相交（不包括包含的情况），那么这两个区间相互可达。我们可以把这些相互可达的区间合并为一个大区间。这样，在同一连通块，这些大区间只有包含关系。如果x和y所在的大区间相同，那么显然x可达y。否则，只需要判断x所对应的大区间到y所对应的大区间是否符合连边条件。

那么，剩下的问题是如何将相互可达的区间合并为大区间。显然，可以利用并查集将大区间合并起来。当新增一个区间，我们需要查询它的左右端点被哪些区间包含。由于新增区间的长度严格递增，不可能出现被包含的情况，所以我们只需要简单地查询左右端点被哪些区间覆盖即可。具体实现可以利用线段树，每次得到一个大区间就将其分成$\log n$个小区间插入线段树。

复杂度分析

时间复杂度：$O(n\alpha(n)\log n)$。
空间复杂度：$O(n\log n)$。

§82 Printer

试题来源 Codeforces 253E

题目大意

有n个任务，每个任务有接收时间t_i，需要花费的时间s_i，以及优先级p_i。每个单位时刻会执行已接收任务中优先级最大的任务。其中某个任务x的优

先级丢失了，但x的结束时间T已知。求一个合法的p_x以及所有任务的结束时间。

$n \leq 50000$。

关键词

堆　模拟

算法讨论

假设我们已经知道了所有的p_i，如何求出每个任务得到结束时间？很容易想到先将任务按照t_i排序，然后从小到大扫描，用一个大根堆维护当前已接收任务的优先级，模拟这个流程。那么我们已经先把问题的第二问解决了。

剩下需要做的是找一个合法的p_x。先假设p_x的优先级是所有任务中最低的，然后我们用解决第二问的方法，求出每个任务在$[t_x, T)$这段时间内的执行时间，设为v_i。同时注意到，随着x的优先级增大，优先级小于p_x的任务的完成时间只会往后推迟。这样我们将任务按照优先级从低到高排序，找到一个k满足$\sum_{i=1}^{k} v_i \geq s_x$且$\sum_{i=1}^{k-1} v_i < s_x$。然后令$p_x = p_k$，让$p_x$不断加1直到该优先级没有出现过即可。这个做法的本质就是让任务x去占用这前k个任务在$[t_x, T)$的执行时间。容易知道，由于原问题保证有解，所有这样求出来的p_x一定合法。

复杂度分析

时间复杂度：$O(n \log n)$。
空间复杂度：$O(n)$。

§83 Shaass and Painter Robot

试题来源　Codeforces 294D

题目大意

在一个$n \times m$的棋盘上，给定处于棋盘边缘的初始位置(x, y)以及初始运动方向（一定是斜45度的方向）。运动中经过的格子被染色，碰到棋盘边缘运动方向遵循镜面反射改变。问棋盘是否能被黑白染色；若能，求出需要多少步。

$2 \leq n, m \leq 10^5$。

关键词

模拟

算法讨论

对于本题，稍加观察即可发现，该棋盘能被黑白染色，等价于运动过程中能到达恰好$n + m - 2$个处于边界上的格子。因此，我们可以将从一个边界上

的格子到下一次到达边界上的格子作为一个阶段，模拟$2(n+m-2)$个阶段，并用标记数组统计到达过的不同的边界格子数目，即可解决本题。

复杂度分析

时间复杂度：$O(n+m)$。
空间复杂度：$O(n+m)$。

§84 Binary Key

试题来源 Codeforces 332E

题目大意

规定了利用钥匙串提取密码串来获取信息串的规则，现在给定密码串p以及信息串s，求长度为k字典序最小的钥匙串。提取规则如下：钥匙串是一个01串，将其复制无数遍后从串首与密码串对齐，密码串中与1对应的位置提取出来组成信息串。

$|p| \leq 10^6$，$|s| \leq 200$，$k \leq 2000$。

关键词

枚举　贪心

算法讨论

不妨考虑枚举钥匙串中有多少个1，设为x个。现在要求有x个1且字典序最小的钥匙串，我们贪心地从后往前确定密码串的每个位置。设$T=\lceil \frac{|p|}{k} \rceil$，我们可以将信息串和文本串分成$T$段。对于每个位置判断填1是否可行，只需要枚举$T$段，检查每段对应的位置是否相等即可。

复杂度分析

时间复杂度：合法的x一定满足$xT \leq |p| \leq x(T+1)$。而对于一个x，求出对应的钥匙串的时间复杂度为$O(Tk)=O(|p|)$。因此总时间复杂度为$O(k|s|(\frac{1}{|p|}-\frac{1}{|p|+k})|p|)=O(k|s|)$。

空间复杂度：$O(|p|+|s|+k)$。

§85 Theft of Blueprints

试题来源 Codeforces 332D

题目大意

给定一张n个点带边权的无向图以及正整数k。这张图中任意k个点，都存在有且仅有一个点，满足这个点与k个点都相邻。求图中任意选择k个点，这k个点与公共点之间边权和的期望值。

$1 \leq k < n \leq 2000$。

关键词

数学期望　组合计数

算法讨论

这是一道基础的组合计数的题目。

设点i的度为d_i，点i的出边的边权和s_i。那么显然期望值为：

$$E = \frac{\sum_{i=1}^{n} s_i \binom{d_i-1}{k-1}}{\binom{n}{k}}$$

直接求解并注意精度误差是可以通过本题的。

更进一步，可以发现，当且仅当$k = 1, 2, n-1$时，才存在这么一张无向图。因此根据$\binom{n}{k} = \binom{n}{n-k}$，求解的组合数中的$k$都满足$k \leq 2$，精度误差就小了许多。

复杂度分析

时间复杂度：$O(n^2)$。

空间复杂度：$O(n)$。

§86 Ladies' Shop

试题来源　Codeforces 286E

题目大意

给出n个不超过m且两两不等的正整数$a_1, a_2, \ldots, a_n$。你需要求出最小的k，以及k个数$p_1, p_2, \ldots, p_k$。设$S = \{\sum_{i=1}^{k} x_i \cdot p_i | x_i \in \mathbb{N}\}$。这$k$个数需要满足所有$a_i \in S$，并且$S$中不超过$m$的元素，一定在$a_i$中出现过。

$1 \leq n, m \leq 10^6$。

关键词

快速傅里叶变换

算法讨论

分析题目，可知，若a_x不能表示成其他a_i的和，那么a_x一定要加入p序列；相反地，若a_x能表示成其他a_i的和，那么a_x不应该被加入p序列。更进一步，设a_x能够被如下表示：$a_x = a_{q_1} + a_{q_2} + \cdots + a_{q_k}$（其中$q_i \neq x$）。那么若问题有解，$\sum_{i=1}^{k-1} a_{q_i}$也一定能被某个$a_y$表示，因此$a_x = a_y + a_{q_k}$。也就是说，能加入$p$序列的数$a_x$，一定不能够表示成$a$中的另外两个不同数的和。

在以上分析的基础之下，容易得出一个$O(n^2)$的算法，但是无法通过本题。考虑一个m次多项式$P = \sum_{i=1}^{n} x^{a_i}$的平方。在$P^2$中，若$x^k$（$1 \leq k \leq m$）的系数为0且在$a$中出现过，则说明$k$需要被加入$p$；若$x^k$（$1 \leq k \leq m$）的系数为1且没在$a$中出现过，则问题无解。这样，剩下的事就是利用快速傅里叶变换求出P^2各项的系数。

复杂度分析

时间复杂度：$O(n + m \log m)$。

空间复杂度：$O(n + m)$。

§87 Ksusha and Square

试题来源 Codeforces 293D

题目大意

给定一个n个点的凸多边形的所有顶点(x_i, y_i)。任意选取两个在多边形内部或边界上不同的整点，求以这两个点连线为对角线的正方形的期望面积。

$3 \leq n \leq 10^5$，所有坐标的绝对值不超过10^6。

关键词

组合计数　数学期望　计算几何

算法讨论

容易得出，对于本题我们需要求的是$\sum (x_i - x_j)^2 + (y_i - y_j)^2$。从式子中可以看出，横纵坐标是分别独立的，我们可以分开来求。

现在考虑求$\sum (x_i - x_j)^2$。注意到所有坐标的绝对值不超过10^6，所以我们枚举每个i，求出直线$x = i$上有多少个符合条件的整点，设为c_i。对于c_i的求法，可以简单地绕着凸包走一圈，得到$x = i$上最高和最低的两个整点的位置，

进而得出c_i。那么x坐标对答案的贡献S为：

$$\begin{aligned} S &= \frac{\frac{1}{4}\sum c_i c_j (i-j)^2}{\binom{\sum c_i}{2}} \\ &= \frac{\frac{1}{2}\sum c_i c_j i^2 + c_i c_j j^2 - 2ic_i jc_j}{(\sum c_i)^2 - \sum c_i} \\ &= \frac{(\sum c_i)(\sum i^2 c_i) - (\sum ic_i)^2}{(\sum c_i)^2 - \sum c_i} \end{aligned}$$

其中$\sum c_i$，$\sum c_i i^2$，$\sum c_i i$均可以在线性时间内求得。

这样，原问题就得以解决。

复杂度分析

设x, y坐标的极差分别为R_x和R_y。

时间复杂度：$O(R_x + R_y + n)$。

§88 ARAM

试题来源

GCJ 2014 Final F

题目大意

游戏中你有n个角色，每个角色有p_i的概率获得比赛的胜利。如果你手中有至少1个游戏币，那么可以选择Reroll，会花费一个游戏币并且随机获得一个新角色。刚开始你有R游戏币，每玩一盘游戏你会获得$\frac{1}{G}$的游戏币，游戏币上限为R。现在进行10^{100}盘游戏，求最优策略下的期望胜率。

$1 \le n \le 1000$，$1 \le R, G \le 20$。

关键词

数学期望　动态规划

算法讨论

本题的突破口，是将问题转化为判定性问题。考虑二分答案Q，设游戏进行了y轮，获胜场次为x，那么我们需要验证$\frac{x}{y} > Q$即$x - Qy > 0$是否成立。因此，我们定义获胜一轮的收益为$1 - Q$，输掉一轮的收益为$-Q$。另外，本题中进行10^{100}盘游戏，我们可以视作无穷多轮。

接下来考虑动态规划。设f_i表示游戏币为i直到下次游戏币为$\min(i+\frac{1}{G}, R)$这个过程中的收益。对于f_i，转移的方法如下：

- 若$0 \le i < 1$，无法Reroll。显然有：

$$f_i = -Q + \frac{1}{n}\sum_{j=1}^{n} p_j$$

- 若$1 \le i \le R$，可以选择Reroll。对于Reroll的策略，显然是规定一个k，若当前角色的获胜概率是前k小的，那么Reroll，否则选择继续游戏。我们枚举一个k（$0 \le k < n$），并让p_i从小到大排序。那么有：

$$f_i = \Big(\lim_{r\to\infty}\sum_{j=0}^{r}(\frac{k}{n})^j\Big)\Big(\sum_{j=0}^{G-1} f_{i-1+\frac{j}{G}}\Big) + \frac{\sum_{j=k+1}^{n} p_j}{n-k} - Q$$

事实上，若$f_R > 0$，那么即可判断胜率Q是可以达到的。这是为什么呢？假设游戏过程中经历了S次游戏币从R回到R的过程。那么由于$f_i \ge -1$，因此进行无穷多轮游戏的收益不低于$S \cdot f_R - RG$。又S是一个趋向于正无穷的一个数字，因此$S \cdot f_R - RG$一定是一个大于零的数。

复杂度分析

时间复杂度：$O(TnRG)$，其中T为根据精度要求进行二分答案的次数。

空间复杂度：$O(RG + n)$。

§89 Symmetric Trees

试题来源 GCJ 2014 Final C

题目大意

给定一棵n个点的树，每个点都染有一种颜色c_i。判断是否能够在二维平面内画出这棵树，使得它是一个轴对称图形。

$2 \le n \le 10000$。

关键词

树形动态规划　树哈希

算法讨论

本道题需要快速判断两棵子树是否同构，可行的方法是对子树求哈希值。定义在一棵有根树中，以x为根的子树的哈希值$hash_x = A(c_x \prod hash_y) + B$，其中$y$为$x$的每个孩子，$A$为$B$自行选择的常量。两棵子树同构，可以等价于两棵子树的哈希值相等。

回到原问题，先考虑对称轴上没有点的情况。这时，我们只需要找到唯一的一条边，满足将其断开后两棵子树大小相等。对于这条边，判断将其断开后的两个子树的哈希值是否相等即可。这一种情况的时间复杂度为$O(n)$

若对称轴上有点，那么这棵树的重心一定在对称轴上。设重心为C，先在$O(n)$内预处理出以C为根所有子树的哈希值。接下来，考虑使用树形动态规划。设dp_x表示以x为根的子树，x在对称轴上，是否能画出一个轴对称图形。我们将x的所有子树的哈希值列出来，将相等的两两配对，表示将其分在对称

轴两侧。最后看无法配对的子树个数，设为T，这些子树设为$y_1, y_2, \ldots, y_T$。显然无法配对的子树只能画在对称轴上。转移分两种情况：

1. 若x为根，对称轴上只能再画两棵子树，$dp_x = T \le 2 \wedge dp_{y_1} \wedge dp_{y_2} \wedge \cdots \wedge dp_{y_T}$。
2. 若x不为根，对称轴只能画一棵子树，$dp_x = (T = 0) \vee (T \le 1 \wedge dp_{y_1})$。

最后，dp_C就是对称轴上有点时的解。

复杂度分析

时间复杂度：根据代码实现不同，复杂度为$O(n)$或$O(n \log n)$。

空间复杂度：$O(n)$。

§90 Candies Game

试题来源 Codeforces 341E

题目大意

给定n个非负整数$a_1, a_2, \ldots, a_n$。每次操作可以选择一对i, j（$a_i \le a_j$），令$a_j = a_j - a_i$，$a_i = 2a_i$。目标是使得a_i中有且仅有2个数为正，其余$n-2$个数为0。输出操作方案，步数不能超过10^6。

$3 \le n \le 1000$，所有a_i之和不超过10^6。

关键词

构造

算法讨论

首先考虑只有3个数的状态(A, B, C)，不妨规定$A \le B \le C$。设$B = qA + r$，其中q为非负整数，$0 \le r < A$。下面给出一种构造方案，实现从状态(A, B, C)得到状态(r, B', C')：

> 设x为初始的A值。可以发现，只要A一直是较小的那个数字，对A进行了k此操作，A就会变成$2^k x$。将q写成二进制，从低位到高位考察每一位。利用类似于二分求幂的方法，若当前这位为1，则对(A, B)进行操作；否则对(A, C)进行操作。设q共有$k+1$位，由于这个过程始终有$A \le 2^k \le B$，而初始时有$2^k \le B \le C$，因此对于每次操作，增加的都是A，减小的是B或者C。这样，就能刚刚好让B减少qx。

这也说明，我们每次都能从(A, B, C)得到一个新状态(A', B', C')满足$A > A'$；换而言之，一定能到达某个状态使得A变为0。

有了以上的推导作为基础，我们可以轻松得到一个算法：每次找三个正数，将其中某个变为零，直到数组中只剩下2个正数。

那么这个算法构造出的总步数是多少呢？设$S=\sum_{i=1}^{n}a_i$。注意到从一个状态转移到其他状态直到有一个数为0，这一过程类似辗转相除法，因此最多只会经过$\log S$个状态。而每一个状态间单次转移的步数为$\log \frac{B}{A}<\log S$。因此总步数不会超过$n\log^2 S\leq 4\cdot 10^5$，满足要求。

复杂度分析

时间复杂度：$O(n\log^2 S)$。
空间复杂度：$O(n\log^2 S)$。

§91 Summer Homework

试题来源

Codeforces 316E3

题目大意

给定一个长度为n的序列$a_1,a_2,\ldots,a_n$以及m个操作。操作有三种：将a_x修改为v；求$\sum_{x=0}^{r-l}(f_x\cdot a_{l+x})$；将$a_l$到$a_r$每个数加上$d$。其中$f_i$为斐波那契数列的第$i$项。

$1\leq n,m\leq 2\cdot 10^5$。

关键词

线段树

算法讨论

本题是一道明显的数据结构题目。先考虑使用线段树。使用线段树的关键在于解决信息的合并的问题。设$G(x)=f_{0+x}a_l+f_{1+x}a_{l+1}+\cdots+f_{r-l+x}a_r$。如果我们在线段树的结点$[l,r]$维护了$G(0)$的值，在信息合并时，我们需要能够从$G(0)$快速得到$G(r-l+1)$。注意到对于$x\geq 2$，$G(x)=G(x-1)+G(x-2)$。因此，$G(x)=G(0)f_{x-2}+G(1)f_{x-1}$。这样，我们在线段树多维护一个$G(1)$的值，即可完成信息合并。

最关键的信息合并得以解决，原题中区间询问，区间修改都能用线段树轻松实现。

复杂度分析

时间复杂度：$O(n+m\log n)$。
空间复杂度：$O(n)$。

§92 Context Advertising

试题来源

Codeforces 309B

题目大意

给定n个单词以及正整数r, c。你要制作一个r行的广告牌，每行最多有c个字符。你要从n个单词中选择一段连续的单词，从上到下从左到右的顺序填入广告牌。每行相邻的单词间有一个空格。问最多能填入多少个单词，输出一种方案。

$1 \leq n, r \times c \leq 10^6$。总字符数不超过$5 \cdot 10^6$。

关键词

倍增

算法讨论

考虑如果确定了一行的第一个单词，那么我们一定是尽量多地接着填入后面的单词，直到达到长度限制。那么，我们就容易得到以下算法。预处理出第i个单词作为某行之首，该行最后一个单词的编号p_i。那么显然下一行的最后一个单词为p_{p_i}，再下一行是$p_{p_{p_i}}$，以此类推。对于第r行的结果，我们可以用倍增得到。最后在最终的p数组中找到最大$i - p_i$即为答案，将其输出即可。

复杂度分析

设总字符数为L。

时间复杂度：$O(L + n \log r)$。

空间复杂度：$O(L)$。

§93 Race

试题来源 Codeforces 241F

题目大意

在一个$n \times m$的网格有一些道路，没有道路的地方是障碍物。所有道路的相交处，称之为路口，都被一个小写字母标识。现给出初始位置及目标位置，以及途中经过所有的路口，同时给出所有格子走到相邻格子所要的时间。问时间k时所处的位置。

$n, m \leq 100$，$k \leq 1000$。

关键词

模拟

算法讨论

这是一道很简单的题目。稍加分析即可知道，从一个路口到下一个路口所经过的格子一定是在同一条直线上的。如果不在同一直线，必定会走向别的道路经过其他路口，再到达目标路口，这与题意不符。

因此，只需要一步一步模拟行走的路线即可。

复杂度分析

时间复杂度：$O(nm+k)$。
空间复杂度：$O(nm)$。

§94 Olya and Graph

试题来源 Codeforces 305D

题目大意

给定一张n个点m条边的有向图，以及整数k。问有多少种添加边的方案，满足以下条件：

1. 所有u到v的有向边满足$u<v$，且无重边。
2. 对于所有i,j（$i<j$），若$j-i\le k$则i到j的最短路长度必须等于$j-i$；否则最短路长度可以是$j-i$或者$j-i-k$。

$2\le n\le 10^6$，$1\le k\le 10^6$，$0\le m\le 10^5$。

关键词

组合计数

算法讨论

我们可以把所有点看做是数轴上的点。那么所有的边只能是从左向右连的，并且对于所有i到j的边，一定满足$j-i=1$或者$j-i=k+1$。对于每条满足$j-i=1$的第一类边，都必须存在于图中。对于图中$j-i=k+1$的第二类边，必须满足两两相交。

有了以上的分析，我们可以考虑枚举所有i，令i到$i+k+1$这一条边是最左边的第二类边，这一条边必须和原图中所有的第二类边相交。那么其余的第二类边的起始点，必须落在$[i+1,\min(n-k-1,i+k)]$内。设这段区间内没有第二类出边的点有c个，那么i对答案的贡献为2^c。

复杂度分析

时间复杂度：$O(n+m)$。
空间复杂度：$O(n)$。

§95 Greg and Caves

试题来源 Codeforces 295D

题目大意

在一个$n \times m$的网格，问有几种染色方案满足以下条件：

1. 存在$[L, R]$，使得第L到第R行都恰好有2个黑格子。
2. 存在整数t使得：对于$L \leq i \leq j \leq t$，第i行两个黑格之间的白格集合是第j行的子集；对于$t \leq i \leq j \leq R$，第j行两个黑格之间的白格集合是第i行的子集。

$n, m \leq 2000$。

关键词

动态规划

算法讨论

对于这类组合计数题目，首先应考虑动态规划。注意到染色方案的上下两部分其实是本质相同的，因此设计状态时不妨只从上半部分入手。设$f_{i,j}$表示上半部分不超过i行，最底层是宽度是j的方案数。转移时枚举上一层的宽度k，那么有$j-k+1$个位置可以摆放，容易得到状态转移方程为：

$$f_{i,j} = 1 + \sum_{k=2}^{m} (f_{i-1,k}(j-k+1))$$

这个转移利用前缀和优化容易做到$O(1)$。

总的方案数如何统计呢？我们需要更加明确上半部分的定义：图形中最宽的每一行都必须属于上半部分。那么我们可以枚举上半部分最底层的行号i宽度j，下半部分从$i+1$行开始，宽度必须严格小于j。容易得到最后的答案为：

$$ans = \sum_{i=1}^{n} \sum_{j=2}^{m} (f_{i,j}(f_{n-i+1,j} - f_{n-i,j})(m-j+1))$$

复杂度分析

时间复杂度：$O(nm)$。

空间复杂度：$O(nm)$。

§96 Levko and Game

试题来源 Codeforces 360E

题目大意

给一张n个点$m+k$条边的带权有向图。有两个玩家A和B，分别从s_1和s_2出发，前往t。双方采用最优策略，谁先到达t就获胜，可以有平局。现在A可以

改变k条边的权值，第i条边可以改为$[L_i, R_i]$内的任意整数值。给出一种方案，使得A能获得最好的游戏结果。

$1 \leq n, m \leq 10^4$，$1 \leq k \leq 100$。

关键词

最短路　贪心

算法讨论

对于每条可以改变权值的边，若这条边只存在于A或者存在于A和B到T的最短路上，那么可以将边权定为L_i；否则，这条边只可能是只存在于B或者不存在于A和B到T的最短路上，那么可以将边权定为R_i。

现在可以规定那些可改变边权的边长度只能是L_i或R_i，那么我们可以使用贪心算法解决本题。首先将所有未确定的边暂定为R_i，求出A和B从各自结点出发到T的最短路。然后枚举每条边权暂定为R_i的边(u, v)，若$distA[u] < distB[u]$那么我们就将这条边改为L_i。接着重新求出两个人到T的最短路，继续重复以上修改过程，直到没有边可以修改为止。

复杂度分析

时间复杂度：$O(k(n+m+k)\log(m+k))$。

空间复杂度：$O(n+m+k)$。

§97 Torcoder

试题来源　Codeforces 240F

题目大意

给一个长度为n只包含小写字母的字符串以及m次操作。每次操作给定l, r，如果可以的话，将字符串第l到r个字符重排成字典序最小的回文串，否则忽略该操作。求最后的字符串。

$1 \leq n, m \leq 10^5$。

关键词

线段树　贪心

算法讨论

首先考虑如何将一个字符串重排成一个字典序最小的回文串。如果该字符串出现奇数次的字符个数超过1，那么就无法构成回文串。否则，我们可以把出现奇数次的那个字符排在中间（如果有的话），其它字符按照字符大小从小到大分布在两侧。

回到原问题，根据以上的分析，对于每个操作，我们只需要知道区间内每个字符出现的次数，就可以得到每个字符重新排列后所对应的区间。这样的区间最多有53个，逐个将那些区间全部改为所对应的字符即可。以上所述可以用线段树轻松实现。

最后输出答案只要在线段树上做DFS即可。

复杂度分析

设字符集大小为S。

时间复杂度：$O(n + mS^2 \log n)$。

空间复杂度：$O(nS)$。

§98 Xenia and Dominoes

试题来源 Codeforces 342D

题目大意

给一个$3 \times n$的棋盘，棋盘中'X'表示障碍，'.'表示空地，其中还有个位置是字符'O'。现在要用1×2的多米诺骨牌不重叠地覆盖所有空地。并且要求至少有一个骨牌，向上下左右某个方向平移一个单位后恰好能覆盖到字符'O'的位置。求总方案数。

$n \leq 10^4$。

关键词

状态压缩动态规划　容斥原理

算法讨论

假设不需要考虑字符'O'的限制，只要求完全覆盖空地。这个问题是个经典的状态压缩动态规划问题。用$dp_{i,S}$表示第i列状态为S的总方案数，逐列进行转移即可。

现在，有了字符'O'，那么不妨枚举哪些位置的骨牌可以通过平移一个单位覆盖'O'。注意到由于列数的限制（只有3列），因此能平移到'O'的骨牌最多只有3个（水平方向2个，竖直方向1个）。这样，可以先枚举2^3种可能，将其覆盖上，再进行动态规划。值得注意的是，最后的答案并不是将2^3种可能的结果累加，而是要用容斥原理进行计数。

复杂度分析

时间复杂度：$O(2^6 n) = O(n)$。

空间复杂度：$O(2^3 n) = O(n)$。

§99 Yaroslav and Arrangements

试题来源 Codeforces 301E

题目大意

如果一个数列首尾相接排列，任意相邻两个数之差绝对值为1并且首项是整个数列的最小值，那么称其为良好的。如果一个数列不降且长度在$[1,n]$之间，数列中每个数在$[1,m]$之间，重排后能得到的良好数列个数在$[1,c]$之间，那么称其为优秀的。给定n,m,c，求优秀数列的个数。

$1 \le n,m,c \le 100$。

关键词

组合计数 动态规划

算法讨论

观察一个良好的数列$\{b_n\}$，其中n必然为偶数，并且排成环后相邻两项之差为1以及-1的个数是相同的，均为$\frac{n}{2}$。如果我们只保留所有满足$b_i < b_{i+1}$的b_i，那么显然新数列只剩下$\frac{n}{2}$个元素。更进一步，这样的新数列能唯一确定一个良好数列。例如良好数列$1,2,3,4,3,4,3,2,3,2$，变换后的新数列是$1,2,3,3,2$，这个新数列也能唯一确定原来的数列。

有了以上的分析，我们可以把良好数列变换后的数列作为动态规划的切入点。现在考虑我们有个新数列，如何求有多少种重排方案使得其对应的原数列是良好数列呢？我们从小到大加入该数列中的数k（假设k有c_k个），显然k只能成段地加到$k-1$的后面。那么添加k的方案数为$\binom{c_{k-1}+c_k-1}{c_{k-1}-1}$。设$f_{n,k,c_k,x}$表示新数列长度为$n$，当前最大数为$k$，重排的方案数为$x$的总方案数。利用这个状态进行动态规划即可。

复杂度分析

时间复杂度：$O(n^3mc)$。虽然这个复杂度达到五次，但是其中的n都已经减少了一半，m也受到n的限制，并且状态中有一维表示的是方案数，因此真正有用的状态不会很多。

空间复杂度：$O(n^2mc)$。

§100 Rotatable Number

试题来源 Codeforces 303D

题目大意

一个b进制下长度为n的数x称为可旋转数当且仅当x通过旋转n次得到的n个数组成的集合为$\{x,2x,\ldots,nx\}$。给出n和b_m，求最大的b满足$1 < b < b_m$且b进

制下存在长度为n的可旋转数。

$1 \le n \le 5 \cdot 10^6$，$2 \le b_m \le 10^9$。

关键词

数论

算法讨论

当$n = 1$时，答案显然为$b_m - 1$。接下来，只考虑$n > 1$的情况。注意到对于一个数x，$bx \bmod (b^n - 1)$相当于将x的最高位换到最低位上。于是对于可旋转数x，我们有：

$$\{kx|1 \le k \le n\} = \{b^k x \bmod (b^n - 1)|0 \le k < n\}$$

设$m = \frac{b^n-1}{\gcd(x,b^n-1)}$。对集合中的数同时除以$\gcd(x, b^n - 1)$并对$m$取模，再乘上$\frac{x}{\gcd(x,b^n-1)}$模$m$意义下的乘法逆元：

$$\{k|1 \le k \le n\} \equiv \{b^k|0 \le k < n\} \pmod m$$

由于$\gcd(b^n - 1, b) = 1$，因此右式中集合不存在m的倍数，左式中也不应该有，因此$m \ge n + 1$。右式中集合在模m意义下，任意两个数相乘得到的数都存在于这个集合中。利用这点用反证法稍加讨论，可以得到$m < n + 2$。

由此可知$m = n + 1$。可以推出：

$$(n + 1)x \equiv 0 \pmod{b^n - 1}$$

又$n \ge 2$，可以确定唯一可能的x是：

$$x = \frac{b^n - 1}{n + 1}$$

接着，我们还要证明：x是可旋转数的充要条件是$n+1$是质数却b是$n+1$的原根。

先证充分性。$b^k x \bmod (b^n - 1)$将x代入，可以写成$(b^k \bmod (n + 1))x$。由于b是原根因此每个$b^k \bmod n$与$1, 2, \ldots, n$一一对应。因此x是可旋转数。

再证必要性。由于我们需要让$b^0, b^1, \ldots, b^{n-1}$取遍1到$n$，所以$n + 1$需要是质数且$b$是原根。

这样，最终的算法就很容易了。首先判断$n + 1$是否为质数，再从大到小枚举b判断是否为原根。

复杂度分析

时间复杂度：$O(\sqrt{n} + \log^2 n(b_m - answer))$。

空间复杂度：$O(\log n)$。