

# IOI2013 中国国家集训队作业 CODEFORCES 题目泛做

(截止日期：第一部分 11 月 10 日，第二部分冬令营前 10 天，具体待定)

教练：胡伟栋 唐文斌

提交方式：通过 [tsinsen.com](http://tsinsen.com) 提交，将开启提交窗口

本次作业包括试题准备和泛做两部分。每位选手原则上应完成全部作业。

## 第一部分：试题准备

这一部分的截止日期为 11 月 10 日，未能按时完成的根据完成时间酌情扣分，扣分上限为全部作业分数。

作业内容：每个准备翻译两道 CODEFORCES 的试题，每人需要翻译的试题见下表中，将翻译好的试题添加到清橙评测系统中。试题的编辑方法见

<http://tsinsen.com/help/newproblem.page>。要求：

1. 按试题的原本意思准确翻译，语言优美；
2. 试题标题使用原试题标题；
3. 试题关键字写试题所考查的算法；
4. 试题来源写：CODEFORCES ???，其中???表示题号，如 CODEFORCES 240F；
5. 题目包含完整的问题描述、输入格式、输出格式、样例输入、样例输出、数据规模和约定等部分，使用清橙的格式化后试题可正常阅读；

6. 至少 20 个数据，数据有梯度、有区分度，包含至少两个可以手算的数据；
7. 提交好正确的参与程序，此参考程序必须在 CODEFORCES 上测试通过；
8. 试题设置为共享。

## 第二部分：试题泛做

这一部分的截止日期为冬令营前 10 天。

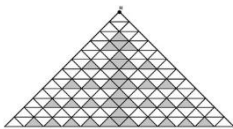

作业内容：每人做完表格中所有试题，提交到 CODEFORCES 上通过，并在清橙评测系统上提交通过，并填写表格中的题目大意（无需背景，说明模型即可）、算法讨论/说明、通过情况。在清橙作业中提交最终表格。

[泛做表格]（完成情况：108 /108）

CODEFORCES 用户 ID: zcwwzdjn

编号	负责人	题目名称	题目大意	算法讨论	Status
7E	贾志鹏	Defining Macros	按 C 的语法定义了一些宏，每个宏都是一个多项式，且可以包含之前定义的宏，最后给你一个多项式，问如果把所有的宏替换进去后运算顺序是否发生改变。	表达式计算的变形。把多项式分为 4 种，替换之后不发生改变、替换之后发生改变的、替换之后最低运算符是加减的、替换之后的最低运算符是乘除的，然后用表达式计算的经典递归方法进行计算，对 4 种多项式的合并进行讨论。另外，要先预处理定义的宏，并对宏名记录该多项式是哪一种。对每个长度为 n 的表达式，期望时间复杂度 $O(n \log n)$ ，最坏 $O(n^2)$ 。	Accepted
8D	贾志鹏	Two Friends	两人都需要从 A 点前往 B	计算几何。可以发现同行的线段一定在 ABC 构成的三角形内，	Accepted

			点, 其中一人要在路上经过 C 点, 两人到达 B 点时走过的路程不能超过各自的最短路一个数值, 问他们从 A 点开始最多能同行多长。	所以可以先二分答案,再三分从 A 点出发的角度(落在 B 和 C 之间), 就可以确定同行的终点, 分开后两人显然都沿自身的最短路走。需要一些特判, 比如两个人都先到 C 再到 B, 仍没有超过他们的上限, 等情况。	
8E	王康宁	Beads	考虑长度为 n 的 01 串(除去全 0 和全 1)。操作 A 是把串逆序, 操作 B 是把串中 0 变为 1、1 变为 0, 若两个串可以通过一些 A、B 操作互相转化, 就划作一类。对每一类只保留字典序最小的串, 问字典序第 k 小的串是什么。	动态规划。首先可以把 01 串视作二进制数, 然后二分答案, 把问题转化成统计字典序不超过某个串的合法串个数。DP 过程从填充 01 串的角度考虑。用 $f(i, eq, re, lu, ru)$ 表示已填充串的前 i 位和后 i 位, eq 表示前 i 位与 A 操作后的后 i 位的大小关系, re 表示前 i 位与 A、B 操作后的后 i 位的大小关系, lu 表示前 i 位与二分的答案的前 i 位的大小关系, ru 表示后 i 位与二分的答案的后 i 位的大小关系。通过这些条件就可以进行决策了, 枚举第 i 位和倒数第 i 位填的值, 此时要保证决策出的串合法。状态数是 $O(n \cdot 3^4)$ , 转移复杂度是 $O(2^2)$ , 再加上二分答案, 最后的时间复杂度是 $O(n^2 \cdot 3^4 \cdot 2^2)$ 。	Accepted
10E	王康宁	Greedy Change	给出 n 个正整数, 每个数可以用无限次。考虑选一些数来组成一个数 m, 一种贪心方案是每次选一个不超过 m 的最大的数 x,	枚举。把数列 a 从大到小排序, 枚举一个数 $a[i]$ 作为贪心选择的最大值。然后我们按照贪心的方法对 $m = a[i] - 1$ 做一次。可以发现, 若要比最优解差, 那么贪心方案中包括 $a[i]$ 和后面一些较小的数, 而最优解包括中间的一段数。所以再枚举中间的一段, 设成 $[i+1, j]$ 这段区间, 那么我们把之前	Accepted

			然后把 $m$ 减去 $x$ , 迭代下去。但这样选的数个数可能比最优解差, 请找出出现这种情况的最小 $m$ 。	$a[i]-1$ 的分解方案中在这个区间里的加起来, 设为 $x$ , 由于要求尽量小, 所以令 $m'=x+a[j]$ , 这样可以保证 $m'\geq a[i]$ , 对 $m'$ 按贪心方案算出一个答案, 再按照选 $[i+1, j]$ 中的数算一个答案, 进行比较。时间复杂度 $O(n^3)$ 。	
15E	陈立杰	Triangles	 <p>定义了一种 <math>n</math> 层三角形如上图, 最顶端为 <math>H</math> 点, 只能沿三角形的边走动, 求从 <math>H</math> 点出发并回到 <math>H</math> 点的路径数, 要求路径不能自交, 路径也不能包含任一灰色区域。</p>	<p>组合计数。由于图形是对称的, 又不能从灰色区域左右横穿, 所以我们只用考虑在左半部分走并回到最靠上的灰色区域的顶点。那么路径必定先是从 <math>H</math> 开始沿大三角形的边走, 然后再某个时刻往里走一步, 可以枚举这个转折点。然后必定往回走, 这就需要在统计</p>  <p>这种形状从绿色点走到紫色点的方案数。可以推导出, 若有 <math>m</math> 个梯形, 那么方案数就是 <math>1+\sum_{0\leq k&lt;m} 4*2^k</math>。注意到转折点以上的这种形状的方案数都要乘起来。所以得到了时间复杂度 <math>O(n)</math> 的算法。</p>	Accepted
17E	陈立杰	Palisection	给一个字符串, 计算有交集的回文子串对数。	字符串处理。先跑一遍 Manacher 算法 (一种线性找出所有回文子串的算法), 然后计 $left[i]$ 表示以 $i$ 位左端点的回文串个数, $right[i]$ 则表示以 $i$ 为右端点。考虑对问题进行补集转化, 统计无交集的回文串对数, 即 $\sum_{1\leq j<i} left[i]*right[j]$ , $2\leq i\leq n$ , 这个可以用前缀和做到线性。最后考虑 $left$ 的求法。这是一种类似于区间修改的操作, 我们开一个计数器 $cnt$ 。	Accepted

				考虑一个极长回文子串，左端点是 $i$ ，中心是 $j$ ，即 $i$ 到 $j$ 都是某回文子串的开头，那么在 $\text{cnt}[i]$ 加 1， $\text{cnt}[j+1]$ 减 1，这样就有 $\text{left}[k] = \sum \{\text{cnt}[i] \mid 1 \leq i \leq k\}$ 。 $\text{right}$ 的求法类似。时空复杂度都是 $O(n)$ 。	
17C	余翔	Balance	考虑只由 a、b、c 三种字母组成的串。令 $t(s, x)$ 为串 $s$ 中 $x$ 字母的个数，称 $s$ 为平衡串当且仅当 $t(s, a)$ 、 $t(s, b)$ 、 $t(s, c)$ 两两之差的绝对值不超过 1。现在给你一个串 $s$ ，你可以进行任意次操作：把一个字母替换成它前面的或后面的。问最终形成多少不同的平衡串。	动态规划。考虑原串 $X$ ，把 $X$ 中连续的相同的缩成一个得到 $X'$ ，设一些操作后得到串 $Y$ ，同样得到 $Y'$ ，那么容易知道 $Y'$ 为 $X'$ 的子序列。于是可以进行动态规划。设 $f(i, x, y, z)$ 表示已有 $x$ 个 a、 $y$ 个 b、 $z$ 个 c，且第 $x+y+z$ 个字母在 $X'$ 对应的位置是 $i$ ，的方案数。然后转移考虑 $f(i, x, y, z)$ 能转到哪些状态。用 $g(i, x)$ 表示在 $X'$ 序列的第 $i$ 位及以后第一次 $x$ 出现的位置，那么 $f(i, x, y, z)$ 可以转移到 $f(g(i, a), x+1, y, z)$ ，也可以到 $f(g(i, b), x, y+1, z)$ ，以及 $f(g(i, c), x, y, z+1)$ 。这样就是一个时间复杂度 $O(n^4)$ 的算法， $n$ 是序列长度。但注意到 $x, y, z$ 只能到 $n$ 的三分之一左右，复杂度的常数是相当小的。	Accepted
19E	余翔	Fairy	给一个无向图，问有哪些边，在去除它这一条边的情况下，图成为一个二分图。	图论+数据结构。首先一个图是二分图当且仅当该图可以黑白染色。先找一个该图的生成森林，然后染色（树一定可以染出来），然后把边分为三个集合，A 集合是树边，B 集合是非树边且两端点颜色不同，C 集合是非树边且两端点颜色相同。先考虑非树边，若 C 集合为空那么所有非树边都可以；若 C 集合大小为 1，那么只有那条属于 C 集合的边；若大于	Accepted

				1, 则去掉任一条非树边都无济于事。接着考虑树边 $t$ : 定义 $P(e)$ 为非树边 $e$ 两端点在树上的路径, 若对于一条 $C$ 集合的边 $e$ , $t$ 不在 $P(e)$ 上, 那么去掉 $t$ 无法达成条件; 若对于一条 $C$ 集合的边 $e_1$ 和 $B$ 集合的边 $e_2$ , $t$ 在 $P(e_1)$ 和 $P(e_2)$ 上, 那么去掉 $t$ , 使得 $e_1$ 两端点异色的同时会导致 $e_2$ 两端点同色, 也不能达成条件。也就是说, $t$ 必须在所有 $C$ 集合边 $e$ 的 $P(e)$ 上, 且在有 $C$ 集合边的时候不能在 $B$ 集合边 $e$ 的 $P(e)$ 上。于是我们对每条非树边, 在树上进行一些操作: 在两点间的路径上每条边加一个值。这个可以用树状数组维护 dfs 序。复杂度 $O(n \log n)$ 。	
23D	杨志灿	Tetragon	有一个有三边相等的凸四边形, 给你这三边的中点坐标, 求这个四边形。	计算几何。枚举一下三个点的顺序, 假设是 $a$ 、 $b$ 、 $c$ , $a$ 为 $AB$ 的中点, $b$ 为 $AD$ 的中点, $c$ 为 $BC$ 的中点, 且 $AB=AD=BC$ , 则有 $A$ 在 $ab$ 的中垂线上, $B$ 在 $bc$ 的中垂线上, 于是可以把 $A$ 、 $B$ 的坐标表示出来, 再利用 $a$ 为 $AB$ 中点列方程, 解出所有点, 再判断是否是凸四边形就可以了。	Accepted
23E	杨志灿	Tree	给一棵树, 划分成若干棵子树, 使得所以子树的大小乘起来的积最大。	动态规划。一个性质是某个分出来的子树里面不会有长度大于等于 3 的简单路径, 因为从中间断开后两边的点数都不小于 2, 而对 $x, y \geq 2$ 有 $xy \geq x+y$ (不妨设 $x \geq y$ , 则可得到 $xy \geq 2x = x+x \geq x+y$ )。于是可以树形 DP。令 $f(u, c)$ 表示以 $u$ 为根的子树, $u$ 所在的块大小为 $c$ 的最大乘积(不算 $c$ )。那么就有 3 种转移方式: $u$ 单独一个点; $u$ 和若干个儿子划在一起; $u$ 和某个儿子 $v$ 以及 $v$ 的若干个儿子划分在一起。	Accepted

				不考虑高精度计算的复杂度的话, 时空复杂度均为 $O(n^2)$ 。	
26E	向鹏达	Multithreading	<pre>repeat <math>n_i</math> times   <math>y_i := y</math>   <math>y := y_i + 1</math> end repeat</pre> <p>给出 <math>n</math> 个这样的线程, 其中 <math>y</math> 是全局变量, 每个线程的每一步可以分开执行。求一个执行序列, 使得所有线程结束后 <math>y</math> 的值为 <math>m</math>。</p>	构造。设每个线程 repeat 次数的序列为 $s\{n\}$ 。若 $n=1$ , 那么只有 $m=s[1]$ 时才有解; 若 $n>1$ , 若 $m$ 大于 $s$ 所有元素的和, 也没有解。注意到有一种执行序列单元可以使得 $y$ 增加 1: 就是“1 t t t t ... 1”, 中间的都不是 1。于是再分两种情况: 设 $s$ 中最小元素下标为 $a$ , 若 $m$ 小于 $s[a]$ , 则当 $m=1$ 时无解, 否则可以先用 $a$ 作为两端构造 $m-1$ 个上面说到的单元, 并使得另一个元素 $b$ 只剩 2 个, 然后再用 $b$ 作为两端构造一个单元; 若 $m$ 不小于 $s[a]$ , 而 $s$ 中所有的和是 $sum$ , 则构造一个除去两端中间长度有 $2*(sum-m)$ 的单元, 再把剩下的一个一个接上去就行了。	Accepted
28D	向鹏达	Do not fear, DravDe is kind	长度为 $n$ 的序列, 每个元素是 $(v, c, l, r)$ 四元组, 要选一个子序列出来, 使得每个元素的 $l$ 等于之前元素的 $c$ 和, 每个元素的 $r$ 等于之后元素的 $c$ 和, 并且使得 $v$ 的和最大。	动态规划。设 $f(i)$ 表示在前 $i$ 个元素中选且第 $i$ 个选中的 $v$ 和最大值。那么对于 $j<i$ , $j$ 能转移到 $i$ 当且仅当 $l[j]+c[j]=l[i]$ , $r[j]=r[i]+c[i]$ , 于是我们可以维护一个二元组集合 $(l[j]+c[j], r[j])$ , 然后每次查询的二元组是 $(l[i], r[i]+c[i])$ , 这个可以用 C++ 的 map 完成映射。边界情况: 对 $l[i]=0$ 的有 $f(i)=v[i]$ , 然后对 $r[i]=0$ 的用 $f(i)$ 更新答案。时间复杂度 $O(n\log n)$ 。	Accepted
30D	罗雨屏	Kings Problem?	平面坐标系, $X$ 轴上有一些点, 在平面内有另一个点 $P$ , 给出起始点, 求经过所有点的最小路程。	枚举。这题的背景下从一点走到另一点显然是走线段。设 $X$ 轴上最左边的点为 $A$ , 最右边的点为 $B$ 。若起点在 $P$ 点, 则要么先到 $A$ 再到 $B$ , 要么先到 $B$ 再到 $A$ ; 而若起点在 $X$ 轴上某一点 $Q$ , 则必然是先在 $X$ 轴上走一段后, 到达 $A$ 或 $B$ , 然	Accepted

				后到 P, 然后再走向 X 轴。于是可以枚举在到 P 前在 X 轴上经过的线段 ( 线段的一端非 A 即 B ), 计算即可。时间复杂度 $O(n)$ 。	
30E	罗雨屏	Tricky and Clever Password	<p>对一个奇长度回文串 <math>s</math>, 我们可以把它分为 <math>\text{prefix} + \text{middle} + \text{suffix}</math>, 其中 <math>\text{prefix}</math> 与 <math>\text{suffix}</math> 长度相同。再有三个可为空的串 A、B、C, <math>t = A + \text{prefix} + B + \text{middle} + C + \text{suffix}</math>, 现在给出 <math>t</math>, 求最长的 <math>s</math>。</p>	<p>字符串处理。可以注意到, 必定存在一个最优解, 使得最优解中的 <math>\text{middle}</math> 是局部极长的 ( 即对于最优解的中心, <math>\text{middle}</math> 不能再长 ), 于是可以先做一遍 Manacher, 然后枚举中心点, 算出 <math>\text{middle}</math> 的区间 <math>[l, r]</math>。注意到 <math>\text{suffix}</math> 是 <math>t</math> 串的一个后缀, 而且在 <math>\text{middle}</math> 确定的情况下, <math>\text{suffix}</math> 的长度是可以二分的, 所以最后的问题是对于 <math>t</math> 的一个后缀 <math>o</math>, 找出 <math>\text{rev}(o)</math>, 即 <math>o</math> 串倒置, 最早出现的位置。设 <math>t</math> 长度为 <math>n</math>, 令 <math>f(i)</math> 为长度为 <math>i</math> 的后缀, 其倒置过后在 <math>t</math> 中最早出现的位置, 即 <math>t[f(i) \dots f(i) + i - 1] = t[n - i + 1 \dots i]</math>。注意到, <math>f(i)</math> 随 <math>i</math> 单调不减, 因为若 <math>f(i + 1) &lt; f(i)</math>, 而长度为 <math>i</math> 的后缀是长度为 <math>i + 1</math> 的后缀的后缀, 所以 <math>f(i)</math> 应该取到更小的 <math>f(i + 1)</math>, 矛盾。于是可以利用字符串哈希在 <math>O(n)</math> 时间把 <math>f</math> 预处理出来, 这样整体复杂度做到了 <math>O(n \log n)</math>。</p>	Accepted
32E	乔明达	Hide-and-Seek	<p>平面上给出两点 P、Q, 和一堵线段墙 WOW1, 一面线段双面镜 MOM1。问 P 能否看到 Q。</p>	<p>计算几何。分两种情况, 一种是直接看到, 一种是通过镜子反射看到。中间涉及到计算两线段交点、计算点到线段的垂足等的计算几何过程。注意边界情况, 时间复杂度 <math>O(1)</math>。</p>	Accepted
35E	乔明达	Parade	<p>给出 <math>n</math> 个底边在 X 轴上且</p>	<p>计算几何。在每个矩形的左右边界设立事件点, 排序后进行</p>	Accepted



			在 X 轴以上的矩形，求出它们并起来的轮廓。	扫描。中途需要维护当前矩形高度的最大值，这个可以用一个最大堆来维护。在答案中可能出现对同一横坐标超过两个点，这要判断一下。复杂 $O(n\log n)$ 。	
36E	朱新瑞	Two paths	给出一个无向边的集合，把它分为两个集合，使得每个集合的边都可以构成一条路径。	图论。注意到构成一条路径说明这些边形成的子图存在欧拉路（或者欧拉回路）。若无向图有超过两个联通块，无解；若有两个联通块，则两个都得有欧拉路；若只有一个，再分两种情况：有 0 或 2 个奇度数点，则找一条欧拉路随便断开成 2 个即可；有 4 个奇度数点，则把其中两个连一条虚拟边，找一个欧拉路，再把虚拟边断掉即可。所以就是一个找欧拉路的复杂度 $O(E)$ 。	Accepted
37E	朱新瑞	Trial for Chief	给一个 $n*m$ 且初始全白的棋盘，每次可以选一个四联通块染成黑色或白色，给出目标状态，问至少染多少次。	模拟。考虑从目标状态出发，把它染回去。假设一开始染的块包含点 P，那么就把 P 所在的联通块反色，再把 P 所在的联通块反色，重复这个过程，在棋盘变回白色时终止。枚举 P 点，算出最小值。时间复杂度 $O(n^2m^2)$ 。	Accepted
39C	罗干	Moon Craters	给出 $n$ 条在数轴上的线段，问最多能选出多少条，使得两两之间不是包含就是不相交（在端点处相交不算）。	动态规划。按照线段长度从小到大处理。对线段 $i$ 记录 $v[i]$ 表示在它内部最多能选多少条线段出来。考虑计算 $v[i]$ ，在它里面的线段 $j$ 必定比它短，所以 $v[j]$ 都已处理好。也就是说，问题变成，在若干线段中选出若干，两两不相交，且使得选出来的 $v$ 和最大。这是个经典的 DP，把线段按右端点排序后，有 $f(i)=\max(f(i-1), f(p[i])+v[i])$ , $p[i]$ 表示第 $i$ 条线段前不与其相交的线段的最大编号。在代码中	Accepted

				我没有预处理 $p[i]$ ，而是用的二分法，所以整个时间复杂度大概是 $O(n^2 \log n)$ 。	
39A	罗干	C*++ Calculations	定义了一种 C++ 的表达式，只有 +、-、* 三种运算，变量只有一个 a，而且变量出现的方式只能是 a++ 或 ++a，* 出现时一定是一个数字乘上 a++ 或 ++a。要求确定一个计算顺序，使得最后表达式的值最大。	字符串处理+贪心。把每个 $c*a++$ 或 $c*++a$ 先抽出来，再把这些“元件”排序：第一关键字是 c 即系数，因为 a 的值一直在增大，由排序不等式知这样最优；当 c 相同时，若 c 为负数，则应先算 ++a 的，再算 a++ 的。设元件数为 n，字符串长度为 m，那么时间复杂度大概就是 $O(m+n \log n)$ 。	Accepted
39E	龙浩民	What Has Dirichlet Got to Do with That?	给三个正整数 a, b, n，两个人轮流操作，要么把 a 加 1，要么把 b 加 1，若某人操作后 $a^b \geq n$ ，则判负。若两人都采取最优策略，问是先手必胜、后手必胜还是平局。	动态规划。以 (a,b) 作为参数进行记忆化的博弈搜索。当 $a^b \geq n$ 时，先手胜；当 $(a+1)^b \geq n$ 且 $a=1$ 时，平局；当 $a^b \geq n$ 且 $b=1$ 时，胜负仅与 (n-a) 的奇偶性有关。否则就看状态 (a+1,b) 和 (a,b+1)，若都是先手胜，则 (a,b) 为后手胜，否则是先手胜。因为 $n \leq 10^9$ ， $a^b$ 当 a 和 b 都不小于 2 时，随 a 或者随 b 的增加增长都是很快的，所以状态数比较少。	Accepted
40E	龙浩民	Number Table	给一个 $n*m$ 的棋盘，棋盘上的格子只能填 1 或 -1，已经有 k 个格子的值已知，问有多少种填法，使	组合计数。首先注意到 n 和 m 必须同奇偶才可能有解（把所有行乘起来应等于把所有列乘起来），所以之后只讨论这种情况。不妨设 $n \geq m$ ，则 $k < n$ ，则必有一行所有格子都没填，设为 R。设其他 $n-1$ 行随便填但保证这 $n-1$ 行都符合条件，	Accepted

			得每一行每一列的元素乘积都是 -1。注意 $k < \max(n, m)$ 。	则 R 可以由这 $n-1$ 行推出来。一个关键点就是，行 R 必定合法。证明：若 $n$ 为偶，则 $n-1$ 为奇，则 $m$ 列的乘积为 $(-1)^{(n-1)}$ ，所以 $m$ 列中有奇数列是 -1，那么 R 行就还需填奇数（偶数 - 奇数 = 奇数）个 -1；若 $n$ 为奇则类似。时间复杂度 $O(nm)$ 。	
43E	谭震	Race	有 $n$ 辆车完成 $s$ 公里的路程，对每个车有若干 $(v, t)$ 二元组，表示该车以 $v$ 速度行驶 $t$ 时间。问整个比赛中有多少次超车。	枚举。枚举两辆车，然后就是在 $s-t$ 图像上求交点的个数。在速度改变的地方设立事件点，排序后扫描一遍统计即可。实现的时候也可以不排序，因为对每一辆车，各自的事件点都是有序的，所以就想合并两个有序集一样，维护两个指针即可。复杂度 $O(n^2 \cdot m)$ ， $m$ 是每辆车的二元组数目的大致范围。	Accepted
44J	谭震	Triminoes	给一个 $n \times m$ 的黑白染色过后的棋盘，有一些格子被去掉了，现在要用 $1 \times 3$ 的骨牌进行覆盖，要求骨牌中心覆盖到的一定是黑色，黑色也必被骨牌中心覆盖。找出任意一个解。	贪心。从左往右、从上往下遍历棋盘，若找到一个白色格子，就尝试往右或往下覆盖，迭代即可。证明：设当前最靠左上的没被覆盖的格子是 $x$ ，若 $x$ 是黑色，那么无解；若 $x$ 是白色，且它右边也是白色，那么只能向下覆盖，而若右边是黑色，那么只能向右，否则右边的格子不能被覆盖。这样甚至可以说明，若有解，那么解是唯一的。复杂度 $O(nm)$ 。	Accepted
45G	毕克	Prime Problem	把 1 到 $n$ 这 $n$ 个自然数分成若干组，使得每一组的和都是质数，并使得组数最少。	数论。令 $m = n(n+1)/2$ ，若 $m$ 是质数，则分成 1 组；若 $m$ 是偶数，根据哥德巴赫猜想，可以把 $m$ 分成两质数之和，在这题的条件下甚至可以分出一个不超过 $n$ 的质数，故分成 2 组；若 $m$ 是奇数，又若 $m-2$ 是质数，则 2 组，若 $m-2$ 不是	Accepted

				质数，则把 $m-3$ 视作偶数来做，一共分成 3 组。一开始把 $n$ 以内的质数筛出来就可以了。	
45E	毕克	Director	给出 $n$ 个姓氏和 $n$ 个名字，求一个匹配序列，使得姓氏与名字首字母相同的对数最大，在这个条件下再要求字典序最小。	贪心。设姓氏序列为 $a$ ，名字序列为 $b$ ， $aa[x]$ 表示 $a$ 中以 $x$ 开头的个数， $bb[x]$ 表示 $b$ 中以 $x$ 开头的个数，那么对于 $x$ ，贡献的匹配数就应是 $req[x]=\min(aa[x],bb[x])$ 。把 $a$ 、 $b$ 分别排序，依次处理 $a$ 中的串。设处理到串 $s$ ，首字母是 $x$ ，若当前 $req[x]=aa[x]$ ，那么必定在 $b$ 中找最小的未匹配的同首字母串；否则，就应该从 $b$ 中找最小的可选串 $t$ ，设 $t$ 首字母为 $y$ ，则 $t$ 需满足 $req[y]<bb[y]$ 。时间复杂度是 $O(n^2)$ 。	Accepted
46F	黎文杰	Hercule Poirot Problem	$n$ 个房间 $m$ 扇双向门 $k$ 个房客，每个房客在一个房间里，每扇门的钥匙在一个房客手中。一开始门都是锁着的。给出一个初始状态，问他们能否通过开门关门、通过开的门走动、把钥匙给另外的人达到一个结束状态。	图论。一个状态可达另一个状态，即两个图的连通性应该一致，也就是可被打开的门的集合一样。对初始状态和结束状态进行分析，得出各自的可以打开的门的集合，然后比较。注意到每把门的钥匙出现且仅出现一次，所以我们可以通过多次迭代，对于房间 $u$ 和 $v$ ，中间的门 $c$ ，手上有 $c$ 钥匙的人在 $p$ ，若 $p$ 和 $u$ 连通或者 $p$ 和 $v$ 连通那么 $u$ 和 $v$ 也就连通了。这个可以用并查集实现。时间复杂度大致是 $O(m^2)$ 。	Accepted
47E	黎文杰	Cannon	在 $X$ 轴正方向上有 $m$ 堵竖直的墙，从原点抛出 $n$ 个初速度相同，角度不同但	计算几何。有斜抛运动的规律知道，不超过 $45^\circ$ 的两条抛物线是只在原点处有交点的，即，初速度一定的情况下，角度不超过 $45^\circ$ 时，角度越大，抛得越远。于是对 $m$ 堵墙算	Accepted

			不超过 45 度的球,在物理规则下,问每个球的最终坐标。假设 X 轴是地面。	出越过这堵墙的最小角度,用一个类似于单调栈的结构来去除一些没用的墙,这样得到一个需要角度递增的墙序列,然后对每个球的角度二分其在序列中的位置即可。时间复杂度约为 $O(m\log m+n\log m)$ 。	
49E	王若松	Common ancestor	定义了一些变换规则,把一个字符变为两个字符。现在给你两个字符串,问它们能否从同一个串转化过来,若是,求出这个串的最短长度。串中只有小写字母。	动态规划。设两串长度分别为 $n$ 、 $m$ 。注意到一个性质,若一个串可以缩成长度为 2 的串,那么一定可以把它砍成两段,并分别缩成一个字符。所以对每个串先预处理出每一段能缩成哪些字符,用 $s1[i][j]$ 和 $s2[i][j]$ 表示对应串 $[i,j]$ 这个区间能缩成的字符集合(可以用 bitmask 实现)。考虑 dp, 计 $f(i,j)$ 表示第一个串的前 $i$ 位与第二个串的前 $j$ 位的最短的祖先串长度,那么有动态规划方程 $f(i,j)=\min\{f(a,b)+1 \mid 0 \leq a < i, 0 \leq b < j, s1[a+1][i]$ 与 $s2[b+1][j]$ 的交不为空 $\}$ 。时间复杂度 $O(n^2m^2)$ 。	Accepted
51F	王若松	Caterpillar	定义一种缩图方式:对于无向图中的两个点 $u$ 、 $v$ ,去掉 $u$ 、 $v$ 和相关的边,添加一个新的点 $w$ ,边 $(w,x)$ 存在当且仅当 $(u,x)$ 或 $(v,x)$ 存在。给一个无向图,问至少进行多少次操作可以把图变成毛毛虫。毛毛虫:无长度为正的环,	图论及动态规划。首先注意到若整个图的联通块数为 $x$ ,那么至少就需要 $x-1$ 次来合并这些联通块。其次,注意到对每一个双联通分量,因为必定有环,所以只能选择把这个环缩到一起,若分量大小为 $x$ ,则也需要 $x-1$ 次。这样问题转化成了对一棵树,要缩多少次才能变成毛毛虫。先处理一个 $f(u)$ 表示把 $u$ 这棵子树缩在一起的最小代价。若 $u$ 没有儿子,那么 $f(u)=0$ ;若 $u$ 有一个儿子 $v$ ,那么显然是把 $v$ 缩上来,于是 $f(u)=1+f(v)$ ;若有多个儿子,那么就是把 $u$ 往上缩,所以 $f(u)=1+\sigma\{f(v) \mid v \text{ 是 } u \text{ 的儿子}\}$ 。然	Accepted

			可以找到一条链，使得所有点到这条链的距离不超过 1，不能有重边，可以有自环。	后再考虑在树上 DP。设毛毛虫中的选的路径为 P。设 $g(u, 0)$ 表示 u 这棵子树，u 在 P 上且可以往上延伸的最小费用， $g(u, 1)$ 表示 u 这棵子树，u 在 P 上且不能往上延伸的最小费用。这个可以在 Dfs 过程中枚举儿子转移得到。更新答案时，重算一次除去 u 这棵子树后，剩下的子树缩到 u 的父亲的代价，再加上 $g(u, 0)$ 和 $g(u, 1)$ 中的较小值。设点数为 n，那么时间复杂度约为 $O(n^2)$ 。	
53E	王启圣	Dead Ends	给一个点数为 n 的无向图，求出它有多少棵生成树，满足叶子数为 m。n 不超过 10。	状态压缩 DP。记 $f(S1, S2)$ 表示已选择的点集为 S1，叶子集合为 S2 时的方案数，转移就是枚举一个 S1 中的点然后往外扩展。注意到这样做是会重复计算的，因为对同一个集合 S2，形成的方式却有很多种。所以要定一个序，即每次加入 S2 的点，加入后一定是编号最大的。时间复杂度约为 $O(3^n \cdot n^2)$ 。	Accepted
57D	王启圣	Journey	在一个 $n \times m$ 的棋盘上，有一些格子被挖去了，且保证每行、每列最多只有一个没挖去，没有两个挖去的格子在对角线上相邻。求随机选两个格子，它们间的期望最短路径长度。	组合计数。先不考虑挖去的格子的影响，设可选的格子有 cnt 个，可以算出这 cnt 个格子间的路径长度总和为 sum。因为题目特殊的性质，即每行每列最多一个被挖去，且没有两个挖去的格子在对角线上相邻，所以两点间的最短路径只可能被增加 2。考虑行方向的情况。若在第 i 行第 j 列被挖去，则 j 左边和右边这些点对都被增加了 2；再考虑 j 左边一些点到其他行，比如 i-1 行，那么只有在第 i-1 行的挖去格子 k 在 j 右边，(i, j) 左边的格子到 (i-1, k) 右边的格子的距离会增加 2，这其实是一种“阶梯”一样的图形。这样最后	Accepted

				算出 sum, 答案就是 $\text{sum}/(\text{cnt}*\text{cnt})$ 。时间复杂度 $O(nm)$ 。	
226E	汤沛雯	Noble Knight's Path	n 个点的树, m 次操作, 第 i 次操作有两种: 1 c 表示把 c 点的权赋成 i, 2 a b k y 表示询问从 a 到 b 权值不超过 y 的第 k 个点的编号, 保证 $y < i$ 。每个点最多被修改一次。	数据结构。注意到我们可以在路径上二分, 所以考虑解决如何统计从 a 到 b 权值不超过 y 的点数。考虑一个补集转化, 计算从 a 到 b 权值超过 y 的点数。注意到每个点至多修改一次, 且 $y < i$ , 从 a 到 b 权值超过 y 的点数 = 从 a 到 b 不超过 i 的 - 从 a 到 b 不超过 y 的 (只考虑修改过的), 而我们可以用 dfs 序 + 线段树来维护一条路径上修改过的点的个数。注意到 t 时刻的线段树与 t+1 时刻的线段树只修改了两个叶子, 树形态也是一样的, 所以可以用可持久化线段树来优化。记 $\text{root}[t]$ 表示 t 时刻线段树的根, 那么从 a 到 b 权值超过 y 的点数, 就可以用 $\text{root}[i]$ 上的答案减去 $\text{root}[y]$ 上的答案了。时间复杂度 $O(m(\log n)^2)$ 。	Accepted
217D	汤沛雯	Bitonix' Patrol	给出 t ( $t \leq 10000$ ) 个正整数, 问有多少个数集, 你不能用这些数加加减减 (每个数只能用一次), 得到一个 m 的倍数 ( $m \leq 120$ )。	搜索。首先对 m 同余的数只能选择一个; 而不同余的数至多选择 6 个, 因为若是选到 7 个, 则有 $2^7 = 128 > 120$ , 根据抽屉原理必有两个不同的集合的和对 m 同余, 这两个集合相减就得到 m 的倍数。注意到 x 和 m-x 两个数无本质区别, 所以可以认为可选的数不超过 60, 暴力的复杂度 $C(60, 6)$ 是可以的。所以直接深度优先搜索, 用一个 bitmask 记录哪些和的值已得到。这里 bitmask 要手动实现, 用两个 64 位整型即可。	Accepted
67E	钱雨杰	Save the City!	给一个简单多边形, 保证有一条边 AB 平行于 X 轴,	计算几何。先把多边形进行一些变换, 使得 AB 在 X 轴上且其他点都在 X 轴上方, 然后逆时针考虑。考虑除了 AB 和与	Accepted



			且其他所有点都在 AB 的一侧。问 AB 上有多少个整点能看到其他点。	AB 相连的两条边, 若有一条水平向右的向量, 则无解; 然后考虑不是水平的向量, 可以延长或反向延长至 X 轴计算交点, 用叉积判断应该更新左端点还是右端点。最后得到一个能看到所有点的区间, 输出区间中的整点个数即可。	
67C	钱雨杰	Sequence of Balls	给出两个长度不超过 4000 的仅由小写字母构成的字符串。4 种操作: 插入字符, 费用 $w_i$ ; 删除字符, 费用 $w_d$ ; 替换字符, 费用 $w_r$ ; 交换相邻两字符, 费用 $w_e$ 。求把第一个串变成第二个串的最小费用。保证 $2*w_e \geq w_i + w_d$ 。	动态规划, 用 $dp(i, j)$ 表示把一串的前 $i$ 位改成二串的前 $j$ 位的最小费用。那么插入删除替换的转移都很好写, 下面重点讨论交换的情况。注意到题目满足 $2*w_e \geq w_i + w_d$ , 这告诉我们, 若是交换了 $i$ 和 $i+1$ , 再交换 $i+1$ 和 $i+2$ , 还不如一次删除加一次插入, 即, 交换不能连续。设两个串为 $a, b$ , 那么转移有以下一些情况: 交换 $i$ 和 $i+1$ , 然后转移到 $dp(i+2, j+2)$ ; 若 $b[j+1]$ 后第一个与 $a[i+1]$ 相同的字符在 $p$ , 那么可以交换 $i+1$ 和 $i+2$ 后, 再中间插入一坨与 $b$ 串 $j+2$ 到 $p-1$ 相同的字符, 转移到 $dp(i+2, p)$ ; 若 $a[i+1]$ 后第一个与 $b[j+1]$ 相同的字符在 $q$ , 那么可以先删除从 $i+2$ 到 $q-1$ 的一坨字符, 再交换一次, 转移到 $dp(q, j+2)$ ; 若 $p, q$ 都存在, 则还可以通过删除、交换、插入一系列操作转移到 $dp(p, q)$ 。这样时间复杂度为 $O(n^2)$ , $n$ 为字符串长度。	Accepted
70D	王迪	Professors task	动态维护凸包, 支持插入点、询问点是否在凸包内。	数据结构。以一开始的三个点的中心建系, 对点集的极角序用平衡树维护, 注意边界, 时间复杂度 $O(n \log n)$ 。	Accepted
70E	王迪	Information Reform	在一棵树上选一些关键点, 每个点花费为 $k$ , 其	动态规划。问题等价于把一棵树划分成若干子树, 然后每棵子树选择一个关键点。考虑在树上进行 DP。 $f_1(t)$ 表示以 $t$	Accepted



			<p>他每个点计算它到最近的关键点的距离，不同的距离有不同的花费，但保证花费随距离不减，求最小花费。</p>	<p>为根的子树的最小花费，<math>f2(t, g, u)</math>表示在以 <math>t</math> 为根的子树内，点 <math>g</math> 作为点 <math>t</math> 连接的关键点，目前考虑到点 <math>u</math> 的最小花费。对 <math>f1(t)</math>，枚举子树 <math>t</math> 内的一点 <math>g</math> 作为关键点，从 <math>f2(t, g, t)</math> 转移过来；对 <math>f2(t, g, u)</math>，考虑 <math>u</math> 的每一个儿子 <math>v</math>，若 <math>v</math> 与 <math>u</math> 在同一子树内，则从 <math>f1(t, g, v)</math> 转移过来；若 <math>v</math> 与 <math>u</math> 不在同一子树内，则 <math>v</math> 到 <math>u</math> 的边断开，则从 <math>f2(v)</math> 转移过来。不妨以 0 号点做根，则答案是 <math>f2(0)</math>。时空复杂度均为 <math>O(n^3)</math>。</p>	
156E	高胜寒	Mrs. Hudson's Pancakes	<p><math>n</math> 个数 <math>a[i]</math>，下标从 0 到 <math>n-1</math>，<math>n \leq 10000</math>。有 <math>m</math> 个询问，每个询问给出一个 <math>d</math> 进制带通配符“?”的串，和一个数 <math>c</math>，问所有下标 <math>i</math> 满足与串匹配的 <math>a[i]</math> 之积与 <math>c</math> 之和的最小质因子（大于 100 则输出 -1）。 <math>m \leq 30000</math>。</p>	<p>动态规划。最小质因子可以理解为模这个质因子为 0，而不超过 100 的质数有 25 个，可以把它们分成 5 组，每组的积都不超过 <math>int</math>。记 <math>dp(b, m, s)</math> 为 <math>b</math> 进制下，以 <math>m</math> 为模，带通配符的串为 <math>s</math>，满足的下标的乘积。这个 <math>dp</math> 不用全部递推出来，记忆化搜索即可。其中串 <math>s</math> 可以用哈希实现，因为有“?”所以可以做一个 <math>b+1</math> 进制的哈希。然后对每个询问暴力即可。注意要去掉模式串高位的一些只能是 0 的问号。</p>	Accepted
105D	高胜寒	Entertaining Geodetics	<p>一个 <math>n*m</math> 的棋盘，每个格子有一个颜色（0 表示透明），一些格子上有一些物块，物块也各自有颜色。现在以 <math>(x, y)</math> 这个格子的</p>	<p>模拟。按照题目意思模拟即可，注意到每次重涂色都是把一个颜色和另一个颜色合并，这个可以用并查集实现，要在并查集的根上记录集合大小和颜色。时间复杂度约为 <math>O(nm)</math>。</p>	Accepted

			<p>物块开始操作。维护一个队列，第一步是把<math>(x,y)</math>的物块压入队列。每次取出队首的物块，若它所在的位置的颜色既不是透明也不和它相同则把所有与这个格子颜色相同的格子涂成物块的颜色，并把这个区域中的物块按某种顺序加入队列。问终止时重涂色了多少个格子。</p>		
193D	雷凯翔	Two Segments	<p>给出 1 到 <math>n</math> 的一个排列，问有多少 <math>1 \leq l_1 \leq r_1 &lt; l_2 \leq r_2 \leq n</math> 满足 <math>[l_1, r_1]</math> 及 <math>[l_2, r_2]</math> 上的所有数排序后构成公差为 1 的等差数列。<math>n \leq 3 \cdot 10^5</math>。</p>	<p>数据结构。对数列进行变换，设原数列是 <math>p[i]</math>，令 <math>q[p[i]] = i</math>，则问题在 <math>q</math> 上变成了：一段连续的区间，它们的数字构成几条线段。设 <math>f([l, r])</math> 表示 <math>q</math> 的 <math>[l, r]</math> 这个区间上的数构成了几条线段。<math>f([n, n])</math> 显然是 1，我们可以考虑如何从 <math>f([l+1, i])</math> 推出 <math>f([l, i])</math>，记 <math>g(i) = f([l, i]) - f([l+1, i])</math>，<math>g(l)</math> 显然也是 1。设与 <math>q[l]</math> 相邻的两个数在 <math>q</math> 中的位置为 <math>a</math> 和 <math>b</math> 且 <math>a &lt; b</math>，那么对 <math>l \leq i &lt; a</math>，在 <math>[l+1, i]</math> 中插入 <math>l</math> 是独立的，所以 <math>g(i) = 1</math>；对 <math>a \leq i &lt; b</math>，在 <math>[l+1, i]</math> 中插入 <math>l</math> 是与某线段相邻，所以 <math>g(i) = 0</math>；对 <math>b \leq i \leq n</math>，在 <math>[l+1, i]</math> 中插入 <math>l</math> 是连接了两条线段，所以 <math>g(i) = -1</math>。这样可以用线段树很快的从 <math>f([l+1, i])</math> 推出 <math>f([l, i])</math>，然后统计一段区间上 1 和 2</p>	Accepted

				的个数。注意到理想情况下 1 和 2 分别是最小值和次小值，所以线段树上维护最小次小以及它们的个数即可。时间复杂度 $O(n\log n)$ 。	
75E	雷凯翔	Ships Shortest Path	在平面中有一个起点和一个终点，还有一个凸多边形，在凸多边形中每单位距离花费 2，而在外面每单位距离花费 1。求从起点到终点的最小花费，注意只能走到“安全”的点上，即这个点要么在起点到终点的连线上，要么在一条凸包的边上。	计算几何。把起点到终点的连线与凸包的交点算出来，建出一个图，并计算出图上的边权，因为点数很少，所以很暴力的跑了一遍 Floyd 算法。时间复杂度 $O(n^3)$ 。	Accepted
76F	郭志芃	Tourist	有 $n$ 个事件，第 $i$ 个事件在时间 $t[i]$ 时在位置 $x[i]$ 发生。你的速度不能超过 $v$ 。两个问题：0 时刻在 0 位置，最多能经历多少个事件；0 时刻自选位置，最多能经历多少个事件。	动态规划。考虑两个事件 $i$ 和 $j$ ， $t[i] \leq t[j]$ ，则 $i$ 可以到 $j$ 的条件，可以推导出就是 $x[i] \leq x[j]$ 且 $-x[i] + v \cdot t[i] \leq -x[j] + v \cdot t[j]$ ，或者是 $x[i] > x[j]$ 且 $x[i] + v \cdot t[i] \leq x[j] + v \cdot t[j]$ 。定义两个数组 $a$ 和 $b$ ，设 $a[i] = -x[i] + v \cdot t[i]$ ， $b[i] = x[i] + v \cdot t[i]$ ，注意到 $x[i] \leq x[j]$ 显然满足 $b[i] \leq b[j]$ ， $x[i] > x[j]$ 也满足前面的 $a[i] \leq a[j]$ ，所以按 $a$ 为第一关键字， $b$ 为第二关键字排序，再求一个 $b$ 的最长不下降子序列，就是答案。时间复杂度 $O(n\log n)$ 。	Accepted

76A	郭志芑	Gift	<p>给一个点数为 <math>n</math> 边数为 <math>m</math> 的无向图，每条边都有两个属性 <math>g</math> 和 <math>s</math>。对一棵生成树，若边集中 <math>g</math> 的最大值为 <math>g_m</math>，<math>s</math> 的最大值为 <math>s_m</math>，则花费为 <math>G \cdot g_m + S \cdot s_m</math>，注意 <math>G</math> 和 <math>S</math> 是常数。求最小费用。</p>	<p>生成树的应用。把所有边按 <math>g</math> 排序，然后顺次考虑每条边。然后应该每次算一个按 <math>s</math> 值计算的最小瓶颈生成树，当然算最小生成树就可以了。注意到每次考虑的范围会多一条边，而这条边若连接两个联通块，则可以直接加进去；否则这条边必定形成了一个环，去掉这个环上的最大边即可。整体时间复杂度 <math>O(m \log m + mn)</math>。</p>	Accepted
77E	罗剑桥	Martian Food	<p>一个大圆半径为 <math>R</math>，一个内切在大圆里面的半径为 <math>r</math> 的圆，每次往大圆中加一个尽量大的圆，使得它与大圆和小圆以及上一次加的圆都相切。问第 <math>k</math> 个加的圆的半径。</p>	<p>反演。在极坐标中考虑问题。把大圆圆心放置在 <math>(R, 0)</math>，小圆圆心放置在 <math>(r, 0)</math>，然后进行一个变换：把点按照 <math>(r, t) \rightarrow (1/r, t)</math> 进行变换，则大圆和小圆都变成了一条直线，且方程分别是 <math>x = 1/2R</math> 和 <math>x = 1/2r</math>。而我们要加的圆都大小相同，在两直线中间，并与两直线相切。这样可以方便的算出第 <math>k</math> 个圆的圆心位置，再连接原点和圆心，与该圆的两个交点必然也是原本圆直径的两端。时间复杂度 <math>O(1)</math>。</p>	Accepted
79D	罗剑桥	Password	<p>一开始有 <math>n</math> 个 0，你需要把 <math>k</math> 个位置变成 1，<math>k</math> 不超过 10。再给出 <math>m</math> 种操作，第 <math>i</math> 个操作是把一段长度为 <math>a[i]</math> 的区间都异或 1。问最少操作的次数。</p>	<p>动态规划。设原序列为 <math>s</math>，并设 <math>s[0] = s[n+1] = 0</math>，构造新序列 <math>t[i] = a[i] \text{ xor } a[i+1]</math>，那么新得到的 <math>t</math> 序列中至多有 <math>2k</math> 个 1，且每次操作只会修改两个值，且这两个值位置之差就是操作的长度。所以可以建一个图，图中有 <math>n+1</math> 个点，若两点位置之差恰好是某个操作的长度，则连一条边，那么可以注意到，两个点 <math>u</math> 和 <math>v</math>，<math>t[u] = t[v] = 1</math> 时，<math>u</math> 走到 <math>v</math>，就可以把这两个 1 同时消去。所以可以用 bfs 找出 <math>2k</math> 个点</p>	Accepted

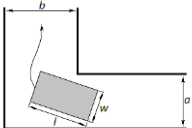
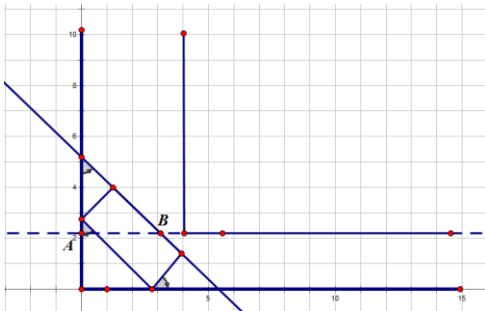
				之间的最短路，然后做一个动态规划： $dp(S)$ 表示目前还未消去的 1 的集合是 $S$ ，然后枚举两个消掉进行转移。时间复杂度 $O(knm+2^{(2k)*k})$ 。	
81E	郑舒冉	Pairs	n 个人，每个人有且仅有一个好朋友。要选尽量多的二人组，使得其中一人是另外一人的好朋友，在组数最多的同时使得异性组数尽量多。每个人只能用一次。	动态规划。每个点有且仅有一个后继，这样就形成了若干环套树。在树上用 $g(u, 0/1)$ 表示 $u$ 这棵子树中， $u$ 选还是没选，的最优答案；而在环上则用 $f(u, 0/1, 0/1)$ 表示在环上走到 $u$ ， $u$ 选还是没选，环上第一个点选还是没选，的最优答案。递推是线性的。于是时间复杂度 $O(n)$ 。	Accepted
82E	郑舒冉	Corridor	考虑夹在 $y=h$ 和 $y=-h$ 间的无限大的区域，在 $(0, f)$ 和 $(0, -f)$ 有两个点光源。有 $n$ 个窗户，对称地分布在 $y=h$ 和 $y=-h$ 上。问灯光覆盖的面积。	计算几何。首先一个光源对一条窗户形成的覆盖区域是一个梯形，一个很好的性质是任意区域至多只会被覆盖两次，所以可以枚举两两梯形，算出相交的面积再减去。相交的面积可以暴力算出所有交点后求一个凸包。时间复杂度大致是 $O(n^2)$ 。	Accepted
83E	孙伟峻	Two Subsequences	对 01 串定义了一些变换规则： $f(\text{空}) = \text{空}$ ， $f(s) = s$ ， $f(s1, s2) =$ 最短的串使得其前缀为 $s1$ 后缀为 $s2$ ， $f(s1, s2, \dots, sn) = f(f(s1, s2, \dots, sn))$	动态规划。首先可以把 01 串视作二进制数，这样可以通过位运算在 $O(m)$ 的时间内算出 $f(s1, s2)$ 。然后我们这样考虑两个子序列：把 $n$ 个串在某一些位置断开，然后计算答案的增量。比如现在枚举到第 $i$ 个串，我们考虑把 $i$ 和 $i-1$ 断开，则我们需要把 $i$ 串和之前的某个串接起来；而我们其实对这个串是什么并不在意，我们只需要知道后缀是什么。于	Accepted

			<p><math>s_1, s_2, \dots, s_{n-1}), s_n)</math>。现给出 <math>n</math> 个长度为 <math>m( \leq 20)</math> 的 01 串, 要求把它分成两个子序列, 分别变换后得到串 <math>S</math> 和 <math>T</math>, 求 <math> S  +  T </math> 的最小值。</p>	<p>是可以记 <math>dp(len, suffix)</math> 表示以长度为 <math>len</math> 的 <math>suffix</math> 结尾的最小增量, 然后我们枚举第 <math>i</math> 个串一个长度为 <math>j</math> 的前缀 <math>s_1</math>, 再枚举第 <math>i-1</math> 个串一个长度为 <math>k</math> 的后缀 <math>s_2</math>, 则 <math>dp(j, s_1)</math> 可以转移到 <math>dp(k, s_2)</math>。时间复杂度 <math>O(n * m^2)</math>, 空间复杂度 <math>O(2^{m * m})</math>。</p>	
85E	孙伟峻	Guard Towers	<p>给出平面上的 <math>n</math> 个点, 要把它分成两个集合, 定义集合的直径是这个集合中最远点对间的距离 (曼哈顿距离), 要求两集合直径的较大值最小。并求出满足这个条件的分法数。</p>	<p>图论。可以二分答案 <math>d</math>, 然后在距离超过 <math>d</math> 的点对间连边, 可以保证合法当且仅当图是二分图。而方案数, 则是在二分图中有 <math>k</math> 个联通块, 则有 <math>2^k</math> 种分法。复杂度 <math>O(n^2 \log n)</math>。</p>	Accepted
86E	王子昱	Long sequence	<p>考虑 <math>k</math> 阶 (<math>k \leq 50</math>) 的线性递推式, 每一项的系数是 0 或 1, 值要模 2。现在想使得这个递推式的循环节达到 <math>2^k - 1</math>, 请输出这个 <math>k</math> 阶递推式的各项系数, 以及递推初始的 <math>k</math> 个值。</p>	<p>枚举+验证。可以证明解比较稠密, 那么就随机枚举这个 <math>k</math> 阶递推式, 然后判断循环节恰为 <math>2^k - 1</math>。问题变成如何判断一个线性递推的最小循环节是否为 <math>m</math>。可以使用矩阵乘法, 判断一下 <math>m</math> 是否可行; 再枚举 <math>m</math> 的每一个质因子 <math>p</math>, 判断一下 <math>m/p</math> 是否不行。</p>	Accepted
89D	王子昱	Space mines	<p>给出空间中一个半径为 <math>R</math> 的球, 在 <math>(ax, ay, az)</math> 位</p>	<p>计算几何。由 <math>R &gt; r[i]</math> 及线段长 <math>\leq 1.5 * r[i]</math> 可以推出碰撞时大球要么与某个小球相切, 要么过了线段的外部端点。所</p>	Accepted

			置以速度 $(v_x, v_y, v_z)$ 匀速运动。然后有另外的 $n$ 个球，第 $i$ 个球半径为 $r[i]$ ，每个球的球心向外引出了若干条线段，线段长度不超过 1.5 倍 $r[i]$ ，且 $r < R[i]$ 。求半径为 $R$ 的球第一次碰到某个物体的时间。	以对每个球及每个点列一个一元二次方程，解之即可。时间复杂度 $O(n+m)$ ， $m$ 为线段数。	
91D	国家琦	Grocers Problem	给出 1 到 $n$ 的一个排列，每次可以选出至多 5 个数，再以任意顺序放回去，问最少多少次使得数列有序。	贪心。先把排列写成置换的形式，得到若干个环。对每个长度超过 5 的环，每次操作可以减少 4 个数，迭代使得每个环的长度不超过 5。然后长度为 1 和长度为 5 的忽略掉，长度为 4 的直接操作。最后对于长度为 2 和长度为 3 的，首先 3 和 2 可以配对，其次 3 可以拆成两个 2 再与其他 3 配对。时间复杂度 $O(n)$ 。	Accepted
93D	国家琦	Flags	对一个数列染色（白黑红黄），要求相邻不能同色，白黄不相邻，红黑不相邻，连续 3 个不能使黑白红也不能是红白黑，然后对称的染色视作一样。问数列长度在 $[L, R]$ 的染色数。	矩阵乘法。可以把问题转化成统计长度不超过 $n$ 的数列的染色数。先不考虑对称的问题，则每个元素的颜色只与前两个有关，所以可以记录下来，再加上要记录一个和，所以就是 9 个状态，可以构造一个 $9 \times 9$ 的矩阵。然后来考虑对称的问题，因为每个非回文串都计算了两次，所以可以加上回文串数再除以 2。统计回文串数，因为回文串必定是奇长度，所以对长度不超过 $n/2$ 的串都可以对应一个回文串，再跑一次	Accepted

				矩阵乘法即可。时间复杂度 $O(9^3 \cdot \log R)$ 。	
97C	杜瑜皓	Winning Strategy	若干场比赛，每场比赛要派 $n$ 个人，若有 $i$ 个人参加过以前的比赛则胜率为 $p(i)$ 。每个人至多参加两场比赛。若人无限多，比赛也无限多，求最大的平均胜率。	数学+枚举。若对于某个 $i$ 使得 $n-2i=0$ ，即可以从第二次开始都保持有 $i$ 个人参加过之前的比赛，所以 $p(i)$ 是个可行的答案；再考虑多个，根据 $n-2i$ 的正负可以分成两坨，而每一坨显然全部选一个是更优的，否则的话则有点加权平均的意味。所以枚举 $i$ 和 $j$ ，使得 $n-2i>0$ 且 $n-2j<0$ ，即一个“贡献”一个“享用”，并且循环下去，所以 $((2j-n) \cdot p(i) + (n-2i) \cdot p(j)) / (2j-2i)$ 也是个可行的答案。取最大值即可。时间复杂度 $O(n^2)$ 。	Accepted
97A	杜瑜皓	Domino	给了 28 张骨牌，每张骨牌都是 $1 \times 2$ 的，每个格子上有一个数，数字从 0 到 6，有 0-0、0-1、0-2、0-3 一直到 5-6、6-6。现在给出一个棋盘，有一些位置可以放骨牌，问有多少种放法，使得可以分成 14 个 $2 \times 2$ 的正方形，每个正方形内数字相同。	搜索。注意到每个数字恰好出现了 8 次，那么有一种方案，那么数字序列任意一个排列也满足。所以我们把左上角的正方形的数字固定为 0，然后顺次枚举其他的，注意每个数字的正方形也是恰有 2 个。对于一个正方形的数字方案，看一看需要的骨牌是否有重复的就可以判断合法了。最后答案乘上 $7!$ 即可。	Accepted
98D	魏鑫鼎	Help Monks	考虑有 $n$ 个盘子的汉诺塔 ( $n \leq 20$ )，有半径相同的盘子，要把第一个柱子上	汉诺塔的变形。设解决过程为 $Solve(cur, a, b, c)$ ，表示考虑把第 $cur$ 个盘子从 $a$ 移到 $c$ ，设经典汉诺塔的解决过程为 $Solve2(cur, a, b, c)$ 。注意在 $Solve2$ 中相同半径的盘	Accepted



			<p>的盘子移到第三个柱子上。注意，虽然有半径相同的盘子，但它们得视作不同的，于是移动后的顺序是不能变的。</p>	<p>子一起处理，顺序的话相当于就是颠倒了 2 次，正好合适。考虑 Solve，若 cur 对应的是最顶层，设最顶层有 <math>x</math> 个，则把 <math>x-1</math> 个从 <math>a</math> 移到 <math>b</math>，然后移动剩下的到 <math>c</math>，在把 <math>x-1</math> 个从 <math>b</math> 移到 <math>c</math>；而如果与 cur 同半径的盘子只有 cur 自己，那么应该直接 <math>Solve2(cur, a, b, c)</math>；若有多个，设第一个与 cur 不同的是 <math>nxt</math>，则先要 <math>Solve2(nxt, a, b, c)</math>，再把 cur 移到 <math>b</math>，再 <math>Solve2(nxt, c, b, a)</math>，再把 cur 移到 <math>c</math>，最后 <math>Solve(nxt, a, b, c)</math>。时间复杂度大概是 <math>O(2^n)</math>。</p>	
98C	魏鑫鼎	Help Greg the Dwarf	 <p>一个 L 型通道，一头宽是 <math>a</math>，一头宽是 <math>b</math>，有一个矩形长为 <math>l</math>，问 <math>w</math> 的最大值，使得矩形可以从一头拐到另一头。</p>	<p>二分+三分。一些奇葩的特殊情况处理略过。另外 <math>w</math> 满足二分性，所以可以二分 <math>w</math> 然后判断。判断过程如下：</p>  <p>其实我们要求的就是在转弯过程中 AB 的最大值，而 AB 的值关于那么被标记的角是单峰的。于是可以三分角度算出 AB 的极大值，与宽度比较即可。</p>	Accepted
191D	孙猛	Metro Scheme	<p>给一个点仙人掌，问至少划分成几条路，使得每条路不是一条简单路径就是</p>	<p>图论。先按双联通分量找到所有的简单环，注意到对于每个环，如果有不少于 2 个点的度数大于 2，即有外连的边，那么这个环可以分配到 2 个简单路径上去，否则就得是个简单</p>	Accepted

			一个简单环，且每条边恰好属于一条路。	环；然后考虑简单路径的数目，就是所有奇度数点配对，所以就是奇度数点数除以 2。这样时间复杂度 $O(n+m)$ ， $n$ 为点数， $m$ 为边数。	
164D	孙猛	Minimum Diameter	给出平面中 $n$ 个点，要恰好删去 $k$ 个，使得剩下的点中最远的两点间距离最小。	图论。删去 $k$ 个点，删去的点对最多就是 $req=n-1+n-2+\dots+n-k=O(nk)$ ，那么答案就应该出现在距离前 $req+1$ 大的点对中，不妨枚举这个成为答案的点对，设为 $AB$ 。则对于一个 $C$ ，若 $CA>AB$ 或 $CB>AB$ 则都得删掉，于是剩下的点都在以 $AB$ 为半径， $A$ 、 $B$ 分别为圆心的圆的交集中，在 $AB$ 同侧的点的距离是不超过 $AB$ 的，所以对所有 $C$ 、 $D$ 满足 $CD>AB$ 的建边，这样能得到一个二分图，而要求的就是这个二分图的最小支配集，即最大匹配，这个可以用 Dinic 算法很快的跑出来。至于方案，对一个二分图 $(X, Y)$ ，我对 $Y$ 中的每一个未匹配的边找不完全的增广路径，把经过的所有标记，那么 $Y$ 中没标记的和 $X$ 中标记了的，选这些点就是一个最小支配集。	Accepted
150E	温和	Freezing with Style	给一棵 $n$ 个点的树，要求一条长度在 $[l, r]$ 的路径，使得路径上边权的中位数最大。	树分治。按树点分治后，我们来考虑通过 $u$ 这个点的路径。显然我们可以二分答案 $x$ ，然后把小于 $x$ 的赋成 -1，不小于 $x$ 的赋成 1，则我们的目标就是找一条通过 $u$ 的路径使得新边权的和非负，所以我们可以去找一条新边权最大的。对于某个 $u$ 的儿子 $v$ ，对它之前的儿子们记录一个 $rec[d]$ 表示以 $u$ 为一端长度为 $d$ 的路径的边权和最大值。那么首先可以倒序枚举 $r$ 到 0，表示 $u$ 往 $v$ 方向走的步数，这时的最大边	Accepted

				权可以直接 bfs 算出来，而在之前的儿子中能走的距离是段连续的区间，且在不断后移，所以可以用单调队列来维护。整个时间复杂度 $O(n(\log n)^2)$ 。	
101E	温和	Candies and Stones	有 $n$ 个糖果 $m$ 个石头，每次可以吃一个糖果或一个石头，不能把石头和糖果吃完，每次吃后统计一下已经吃的糖果 $a$ 个，已经吃的石头 $b$ 个，得分 $(x[a]+y[b])\%p$ ， $x$ 、 $y$ 、 $p$ 已知。 $n, m \leq 20000$ ，求最大得分及策略。	动态规划。因为时间给得很充裕，所以 $O(nm)$ 的动态规划不会超时，但是会超空间。记 $dp(i, j)$ 表示剩 $i$ 个糖果 $j$ 个石头时的最大得分，则 $dp(i, j)$ 可以从 $dp(i-1, j)+(x[n-(i-1)]+y[m-j])\%p$ 转移过来，也可以从 $dp(i, j-1)+(x[n-i]+y[m-(j-1)])\%p$ 转移。这个 $dp$ 可以滚动数组，于是问题变成了如何记录方案。可以用 $rec(i, j)$ 表示 $dp(i, j)$ 是转移到哪里，这是个布尔变量。但这样仍然会超空间，因为一个布尔就要占一字节。所以可以压缩一下，把相邻的 64 个压成一个，这样就可以用 64 位长整型来记录方案。可惜全部记录仍然要超空间，我们可以只记录 $n/2$ 到 $n$ 的，然后在小范围中再跑一次 $dp$ 。至多两次 $dp$ ，所以时间复杂度是 $O(nm)$ 。	Accepted
103E	胡渊鸣	Buying Sets	$n$ 个集合 ( $n \leq 300$ ) 每个集合包含的数都是 1 到 $n$ 的自然数。每个集合有一个费用，可能为负。保证对任意 $k$ 个集合，它们并集的大小不小于 $k$ 。求选 $k$ 个集合使得并集大小恰为	网络流。先把负的变成正的，正的变成负的，然后问题变成求最大值。因为任意 $k$ 个集合的并的大小不小于 $k$ ，所以把 $n$ 个集合视作 $X$ ， $n$ 个点视作 $Y$ ，那么由 Hall 定理，知二分图 $(X, Y)$ 有完备匹配。假设我们求出了一个完备匹配，对每个 $Y$ 中的点记录 $match[i]$ 表示与它匹配的 $X$ 中的点。对每个 $X$ 中点 $u$ ，枚举从它出发的非匹配边，设连向了 $v$ ，则选 $u$ 必然导致选择 $match[v]$ 。注意这是要选择的一个闭合图，	Accepted

			k 的最小费用。	所以我们可以用最大权闭合图的经典做法来搞。时间复杂度 $O(n^3)$ 。	
105E	胡渊鸣	Lift and Throw	三个人在数轴的正半轴上，每个人可以走一步，可以把与自己差 1 距离的举起来，可以把举起来的人扔出去。每个人的初始位置、走的范围、扔的范围都不超过 10，每人每种操作至多一次。问最远能到的距离。	搜索。当 3 个人分别在 8、9、10，且走的范围和扔的范围都是 10 时，答案达到最大，为 42。考虑状态数，对每个人有 42 个可选的位置，然后走了还是没走，没举起过人、举了一个、举了两个、已经扔出去了，这就一共 $2^4=8$ 种状态，所以状态总数是 $(42*8)^3$ ，且很多状态不可达到，所以直接暴力深搜，用哈希来判重就可以了。	Accepted
107D	刘定峰	Crime Management	n 个格子要染色，给出 m 个限制条件，第 i 个条件是要求颜色 c[i] 的数目必须是 k[i] 的倍数。若对同种颜色有多个限制满足任意一个即可。求染色方案数， $n \leq 10^{18}$ ，保证所有 k[i] 的积不超过 123。	矩阵乘法。先对每个颜色求出所有限制的最小公倍数，替代为新的限制。记 $dp(i, S)$ 表示染完前 i 个格子，第 j 种颜色出现数模 k[j] 等于 S[j] 的方案数。因为所有限制积不超过 123，所以 S 中所有限制的积也不超过 123，所以可以构造一个 $123*123$ 的矩阵，注意限制为 1 的数，它们是可以不用考虑在 S 中的。这样再用快速幂加速，时间复杂度大概就是 $O(123^3 * \log n)$ 。	Accepted
113D	刘定峰	Museum	一个无向连通图 n 个点和 m 条边，一开始两个人分	三种做法： 第一种是记 $dp(k, u1, v1, u2, v2)$ 表示 $2^k$ 步，一个人从	Accepted

			<p>别在 <math>a</math> 和 <math>b</math>，已知每分钟在点 <math>i</math> 停留的概率是 <math>p[i]</math>，否则以等概率向相连的其他点走。问两人相遇在每个点的概率。</p>	<p><math>u_1</math> 走到 <math>u_2</math>，另一个人从 <math>v_1</math> 走到 <math>v_2</math> 的概率。当 <math>k=18</math> 时精度就可以满足要求了。复杂度 <math>O(18 \cdot n^6)</math>。</p> <p>第二种是枚举每个点作为相遇的点，然后反过来算出到达 <math>(a,b)</math> 的概率。设 <math>p(x,y)</math> 表示到达 <math>(x,y)</math> 的概率，设枚举的相遇的点是 <math>i</math>，那么初始条件 <math>p(i,i)=1</math>，且有 <math>p(x,y)=\sum p(u,v) \cdot \text{tran}((x,y),(u,v))</math>，这里 <math>\text{tran}((x,y),(u,v))</math> 表示从 <math>(x,y)</math> 一步走到 <math>(u,v)</math> 的概率。这样可以搞一个线性方程组，高斯消元即可。虽然复杂度高达 <math>O(n^7)</math>，但是高斯消元常数写小一点也是可过的。</p> <p>第三种是对第二种的改进。注意到高斯消元是指是解矩阵方程 <math>Ax=B</math>，在上一种方法中 <math>A</math> 是 <math>n^2 \cdot n^2</math> 的，<math>B</math> 是 <math>n^2 \cdot 1</math> 的，而每次枚举相遇的点，发生改变的只有 <math>B</math>，所以我们可以把 <math>B</math> 扩充成 <math>n^2 \cdot n</math> 的矩阵，这样只需解一次 <math>Ax=B</math> 就可以了，具体做法是求 <math>A</math> 矩阵的逆元，然后 <math>x=A^{-1} \cdot B</math>。时间复杂度 <math>O(n^6)</math>。</p>	
115D	高远	Unambiguous Arithmetic Expression	<p>所有非负整数都是 UAE。如果 <math>X</math> 和 <math>Y</math> 是 UAE，那么 <math>(X)+(Y)</math>，<math>(X)-(Y)</math>，<math>(X) \cdot (Y)</math>，<math>(X)/(Y)</math> 都是 UAE。如果 <math>X</math> 是 UAE，那么 <math>-(X)</math> 和 <math>+(X)</math> 都是 UAE。给你一个由数字及加减乘除组成的表达式，</p>	<p>动态规划。首先所有数字可以视为相同的，不妨都缩为 0，然后加减号视作加号，乘除号视作乘号，此时若一个乘号前不是数字，那么就不会有 UAE 去括号后与之相同，而最后一个字符也必定是数字。排除这些无解情况后，就是有解的情况了。考虑从左到右处理，记 <math>dp(i)</math> 表示有 <math>i</math> 个左括号没消去的方案数，初始边界 <math>dp(0)=1</math>。然后若是碰到了加号或者乘号，这后面肯定会接一个左括号，所以要把 <math>dp</math> 数组右移一位；否则，这个数字后面肯定跟若干个右括号，所以</p>	Accepted

			问有多少个 UAE 去括号后与之相同。表达式长度 $n$ 不超过 2000。	$dp(i) += \sum\{dp(j)\}$ 对所有的 $j > i$ 。所以时间复杂度大概是 $O(n^2)$ 。	
120I	高远	Luck is in Numbers	给一个长为 $2n$ 的数字 $x$ ，每个数字都是以电子形式写出的，一个数字的权值是把它的前 $n$ 位和后 $n$ 位重在一起后，重复的笔画总数。求一个最小的比 $x$ 大的长为 $2n$ 的数字，使得权值增大。	贪心。从低位到高位枚举变大的最高位，然后把后面全部填成 8，因为 8 拥有所有可能的笔画，所以这时的权值是理论最大的。若这个最大值比原来的权值大，那么则逐位确定较低的位，方法还是枚举数码后在后面全部填 8 算出最大值。过程中会用到各种前缀后缀和。时间复杂度大致是 $O(10 \cdot 2n)$ 。	Accepted
123E	陈文潇	Maze	考虑一棵 $n$ 的点的树，每个点有一定概率成为起点，又有一定概率成为终点，问从起点通过 Dfs(中间一个点扩展时会等概率扩展)到终点的期望步数。	图论。从两个点考虑：合并一棵子树，计算每条边的贡献。对一棵子树 $u$ ，起点出现在里面的概率是子树每个点概率之和，设为 $x$ ，然后 $u$ 的父亲 $p$ 成为终点的概率是 $y$ ， $u$ 子树大小为 $s$ ，那么这里就给答案贡献了 $y \cdot x \cdot s$ 。对 $(u, p)$ 这条边还有一种走法，从 $u$ 子树外通过 $p$ 走到 $u$ ，这时设 $u$ 成为终点的概率是 $y$ ，那么对答案的贡献就是 $y \cdot (1.0 - x) \cdot (n - s)$ 。计算完成后就可以把 $u$ 子树合并到 $p$ 上了。时间复杂度 $O(n)$ 。	Accepted
125E	陈文潇	MST Company	给一个 $n$ 个点 $m$ 条边的无向图，求点 1 的度数恰为 $k$ 的最小生成树。	生成树。把与 1 相连的边视作白色边，其他边视作黑色边，那么就是要求一棵恰有 $k$ 条白色边的最小生成树。这时我们可以对每条白色边的权值加上一个偏移 $\delta$ ，易知随	Accepted

				<p>delta 的增加, 最小生成树中的白色边是不增的, 所以我们可以二分 delta, 然后用普通的最小生成树算法来判断。因为要输出方案, 而一个 delta 最小生成树也不是唯一的, 我们可以每次把边先乱序再排序。时间复杂度大概是 <math>O(m \log m * R)</math>, R 是二分的次数。</p>	
193E	彭天翼	Fibonacci Number	<p>定义了一个模 <math>10^{13}</math> 的斐波那契数列, 问某个数 <math>x</math> 是否出现在序列中, 若出现的话求出最早出现的位置。保证答案在 <math>\text{long long}</math> 以内。</p>	<p>数论。取 <math>10^{13}</math> 的两个约数 <math>A=2^{13}</math>, <math>B=5^9</math>, 那么有 <math>t\%A=(t\%10^{13})\%A</math>, <math>t\%B=(t\%10^{13})\%B</math>, 于是我们对模 A 和模 B 分别求出数列的循环节, 并找到二元组 <math>(a,b)</math>, 表示模 A 意义下, 第 a 个数是 <math>x\%A</math>, 模 B 意义下, 第 b 个数是 <math>x\%B</math>。若模 A 的循环节是 <math>\text{len}_a</math>, 模 B 的循环节是 <math>\text{len}_b</math>, 那么有 <math>a+\text{len}_a*x=b+\text{len}_b*y</math>, 这个方程可以转化成线性同余方程, 用扩展欧几里德算法做就可以了。</p>	Accepted
145D	彭天翼	Lucky Pair	<p>定义只由 4 和 7 组成的数是幸运数。给一个长为 <math>n</math> 的数列 (<math>n \leq 10^5</math>), 保证不超过 1000 个幸运数, 问有多少 <math>a, b, c, d</math> 满足 <math>a \leq b &lt; c \leq d</math>, 且不存在一个幸运数既在 <math>[a,b]</math> 上出现, 又在 <math>[c,d]</math> 上出现。</p>	<p>数据结构。设幸运数数目为 <math>m</math>, 因为 <math>m</math> 不超过 1000, 所以我们可以枚举左区间包含哪些幸运数。那么对于左区间右边的区域, 因为出现过的幸运数不能再选, 所以右边被分割成了若干块, 并且可以在块内自由选择。这时就需要一个东西来维护右边的分割情况。考虑先枚举左区间包含幸运数的右端点, 再从大到小枚举左端点, 这时包含的幸运数是在增加的, 就相当于在右边某些位置插入了一些数, 所以可以用平衡树来维护。计数问题的细节比较多。最终的时间复杂度大概是 <math>O(m^2 * \log n)</math>。</p>	Accepted
132E	许昊然	Bits of merry old	<p>给出一个长度为 <math>n</math> 的序列</p>	<p>网络流。一开始的想法是, 把 <math>n</math> 的值拆点, 在 <math>i</math> 与 <math>i'</math> 间连</p>	Accepted

		England	<p>(<math>n \leq 250</math>), 以及 <math>m</math> 个变量 (<math>m \leq 26</math>), 需要用一些操作依次输出这 <math>n</math> 个值。操作, 要么是对一个变量赋值, 要么是输出一个变量。注意, 赋值操作是有代价的, 其代价等于赋的值的二进制表示中 1 的数目。求最小代价。</p>	<p>一条下界为 1 上界为 1 费用为 0 的边, 然后建源和汇, 源向每个 <math>i</math> 连一条下界为 0 上界为 1 费用为第 <math>i</math> 个数代价的边, <math>i'</math> 向汇连一条下界为 0 上界为 1 费用为 0 的边。然后对于 <math>i &lt; j</math>, 若第 <math>i</math> 个变量的值等于第 <math>j</math> 个变量的值, 则从 <math>i'</math> 到 <math>j</math> 连一条下界 0 上界 1 费用 0 的边, 否则连一条下界 0 上界 1 费用为第 <math>j</math> 个数代价的边。这样跑一次流量不超过 <math>m</math> 的最小费用流即可。带上下界的费用流不太会, 由于这题的特殊性于是就直接去掉流量下界, 把必须经过的边的边权设成一个很小的值, 然后跑一般的费用流就可以了。</p>	
138D	许昊然	World of Darkraft	<p>一个 <math>n*m</math> 的棋盘 (<math>n, m \leq 20</math>), 每个格子是 L、R 或者 X。两人一次进行操作, 每次选择一个未被标记的格子, 若是 L, 则向左下和右上发出射线, 碰到标记过的格子或者边界就停止; 若是 R, 则射线方向是右下和左上, 若是 X, 则是左上、左下、右上、右下。每次操作后把射线上的格子设为已标记。不能操作的输, 问是否先手必胜。</p>	<p>组合游戏。把棋盘旋转 45 度后且填充成一个更大的矩形后, 发现每个操作都是把当前棋盘分割成若干个小矩形, 于是就可以暴力的进行动态规划了, 即状态为 <math>(x1, y1, x2, y2)</math> 表示一块矩形, 转移则是枚举操作, 然后算出 SG 值。可以记忆化实现, 时间复杂度 <math>O(n^3 * m^3)</math>。</p>	Accepted



140F	钟泽轩	New Year Snowflake	一个点集被称作中心对称的，当且仅当存在一个点 $X$ ，使得对集合中的任一点 $A$ ，可以在集合中找到一个点 $B$ 使得 $A$ 、 $B$ 关于 $X$ 对称。给 $n$ 个点，最多可添加 $k$ 个点，问有多少个不同的点可能成为对称中心。 $k \leq 10$ 。	排序+枚举。若 $k \geq n$ 那么答案为无穷。于是只考虑 $k < n$ 的情况。把所有点按 $x$ 为第一关键字 $y$ 为第二关键字排序。分两种情况：对称中心在 $n$ 个点中，对称中心不在 $n$ 个点中。对第一种情况，可能的选择只有不超过 $k*2$ 种（排序后中间的一些点），不妨枚举一个中心，然后计算需要添加多少个点。对第二种情况，枚举比对称中心 $X$ 小的点的个数 $i$ ，不妨设 $i \geq n-i$ ， $i$ 的选择也大概只有不超过 $k*2$ 种，然后分别枚举在比 $X$ 大的点中加的点的数目 $j$ ，比 $X$ 小的点中加的数目 $k$ ，然后去在比 $X$ 小的点集中去掉最小的 $j$ 个，比 $X$ 大的点集中去掉最大的 $k$ 个，然后再线性地判断一次。时间复杂度大概是 $O(k^3*n)$ 。	Accepted
147B	钟泽轩	Smile House	给一个 $n$ 个点 ( $n \leq 300$ ) 的有向图，问是否有负环，若有输出最小的负环长度。	矩阵乘法。若我们可以求出任两点间步数不超过某个值的最短路，那么就可以二分答案了。于是可以改动一下矩阵乘法，把加号改成取 $\min$ ，乘号改成加号，这样进行矩阵乘法，相当于就是求两点之间步数为某个值得最短路，所以可以二分答案 $x$ ，然后用快速幂跑矩阵乘法，算出从每个 $i$ 走不超过 $x$ 步回到 $i$ 的最短路。时间复杂度 $O(n^3*(\log(n))^2)$ ，有点卡常数。	Accepted
152D	成羽丰	Frames	定义矩形框是在棋盘上一个 $a*b$ 的矩形， $a$ 和 $b$ 都不小于 3，且边框上的格子被标成 #，边框内是空心	枚举。先算出有哪些行，满足这一行上最长连续的 # 不小于 3，设有 $a$ 个，算出有哪些列，满足这一列上最长连续的 # 不小于 3，设有 $b$ 个，若 $a$ 和 $b$ 都不超过 4，就可以暴力从这 $a*b$ 个交点找 4 个作为两个边框的对角；若 $a$ 和 $b$ 中有一个超过	Accepted

			的。现在在 $n*m$ 的棋盘上放了两个框，框可以重合，请找出它们，或输出不可能。	4，不妨设 $a>4$ ，而 $b=4$ ，那么必定存在一个一边长为 3 的矩形框，这时从满足条件的行中找到靠上的两行及最靠下的两行共 4 行，按上一种情况做就好了。判断的复杂度可以做到 $O(n+m)$ 。	
183D	成羽丰	T-shirt	有 $n$ 个人 $m$ 种衬衫， $n \leq 3000$ ， $m \leq 300$ ，第 $i$ 个人适合第 $j$ 件衬衫的概率为 $p[i][j]$ 。现在要带一些衬衫，使得能找到适合衣服的人数的期望值最大。	动态规划。能找到适合衣服的人数的期望，等于每种衬衫，适应它的期望人数之和。设 $dp[i][j][k]$ 为前 $i$ 个人，至少 $k$ 个适合第 $j$ 种衬衫的概率，那么 $dp[i][j][k] = p[i][j]*dp[i-1][j][k-1] + (1-p[i][j])*dp[i-1][j][k]$ ，那么对第 $j$ 种衬衫，带 $k$ 件的期望适合的人就是 $\sum dp[n][j][t] \mid 1 \leq t \leq k$ 。又，对于同一个 $j$ ， $dp[n][j][t]$ 随 $t$ 增大时不增的，所以要使答案最大，就是要在 $dp[n][j][k]$ 中选择值最大的 $n$ 个 $(j,k)$ 二元组。这样做下来复杂度达到了 $O(m*n*m)$ 。但是通过观察发现对同一个 $j$ ，若所有人适合它的概率不是 1， $dp[i][j][t]$ 随 $t$ 的增大下降得很快（指数级），而反过来，若有概率是 1，那么直接带一件这种衬衫最好。所以可以记录一个 $sum$ ，当 $sum+dp[i][j][t]$ 小于 $EPS$ 时就可以停止枚举 $t$ ，并把 $dp[i][j][t]$ 累加到 $sum$ 里面。这样效率很高，不过时间复杂度不知道怎么算。	Accepted
217E	李煜东	Alien DNA	给一个长度不超过 $3*10^6$ 的仅由 AGCT 组成的字符串，有 $n$ 个操作，	数据结构。正向维护的话，一开始的字符串里面可能很多信息都是无用的，直接搞比较复杂。倒过来考虑这个问题。设 $Solve(i,j)$ 表示 $i$ 次操作后的前 $j$ 个字符组成的字符串，	Accepted

			<p>每个操作形如<math>[l, r]</math>，先<math>[l, r]</math>上偶数位置的字符取出<math>r</math>后面，再把奇数位置的字符取出接在后面。问最后字符串的前<math>k</math>位，<math>k \leq 3 \times 10^6</math>。操作数目不超过 5000。</p>	<p>那么答案就是 <math>\text{Solve}(n, k)</math>。考虑 <math>\text{Solve}(i, j)</math>，若 <math>r[i] \geq j</math>，那么第 <math>i</math> 次操作对 <math>\text{Solve}(i, j)</math> 没有任何影响，所以 <math>\text{Solve}(i, j) = \text{Solve}(i-1, j)</math>；否则，对操作 <math>[l[i], r[i]]</math>，算出由它决定的区间 <math>[a, b]</math>（其中 <math>a = r[i] + 1</math>），然后先算出 <math>\text{Solve}(i-1, j - (b - a + 1))</math>，再暴力地完成这次操作。因为操作中插入的字符之后不会被删除，而又只会插入 <math>k</math> 个字符，所以复杂度是可以保证的。于是需要用些东西来维护字符串的插入。可以用平衡树来实现，但是常数较大。我使用的 C++ 的 <code>ext</code> 库中的 <code>rope</code>，整个时间复杂度做到了 <math>O(k \log(k) + n)</math>，且常数较小。</p>	
135E	李煜东	Weak Subsequence	<p>定义一个串 <math>a</math> 是一个串 <math>b</math> 的次连续子串，当且仅当 <math>a</math> 是 <math>b</math> 的一个不连续的子序列。问由 <math>k</math> 种字符组成的字符串，且最长次连续子串长度是 <math>w</math>，的串有多少。<math>k \leq 10^6</math>，<math>w \leq 10^9</math>。</p>	<p>组合计数。一个重要的结论是一个串的最长次连续子串要么是这个串的一个前缀，要么是一个后缀。考虑它作为一个后缀。枚举 <math>t</math>，使得 <math>1 \leq t \leq k</math>，表示串的前 <math>t</math> 个字符两两不同，而 <math>t+1</math> 与之前某个相同，那么从 <math>t+1</math> 到串末尾就是一个次连续子串，所以 <math>t+1</math> 后面会有 <math>w-1</math> 个字符。同时要保证不存在超过 <math>w</math> 的前缀是次连续子串，所以串的后 <math>t</math> 个字符也要两两不同，这里要分 <math>t \leq w-1</math> 及 <math>t &gt; w-1</math> 分别推一下公式。对于前缀则同理，但要注意不要算重了。预处理一下阶乘和逆元，时间复杂度可以做到 <math>O(k \log(\text{mod}))</math>，<math>\text{mod}</math> 是模的数。</p>	Accepted
163D	黄嘉泰	Large Refrigerator	<p>给出长方体的体积 <math>V</math>（按素因子分解给出，不超过</p>	<p>枚举。设三边长为 <math>a, b, c</math>，不妨设 <math>a \leq b \leq c</math>，那么有 <math>a \leq V^{1/3}</math>，<math>b \leq (V/a)^{1/2}</math>，所以可以暴力枚举。注</p>	Accepted

			<p><math>10^{18}</math>), 求使得表面积最小的三边长。</p>	<p>意到 <math>a, b</math> 均为 <math>V</math> 的约数, 所以可以只枚举 <math>V</math> 的约数。有几个比较强的剪枝: 答案最小时三边长比较接近, 所以可以从大到小枚举 <math>a</math>, 然后在枚举 <math>a</math> 时, 取 <math>b=c</math> 算一个表面积, 这是这个时候理论的最小值; 枚举 <math>b</math> 时也可以从大到小, 一旦有一个解后就不用再枚举更小的 <math>b</math> 了。时间复杂度大概是 <math>O(V^{1/3} \cdot \log(V))</math>, 那个 <math>\log</math> 是我在枚举 <math>a</math> 后, 找到小于等于 <math>(V/a)^{1/2}</math> 的最大的约数时用的二分查找。</p>	
167E	黄嘉泰	Wizards and Bets	<p>一个 <math>n</math> 个点 <math>m</math> 条边的有向无环图, 把入度为 0 的点称作源, 出度为 0 的点称作汇, 保证源和汇的数目相同, 设为 <math>k</math> 个。现在需要找出 <math>k</math> 条路径, 从某个源走到某个汇, 每个源汇要恰被覆盖一次, 路径不能在顶点处相交。设某种方案中, 汇 <math>i</math> 与源 <math>a[i]</math> 匹配, 那么对所有 <math>i &lt; j</math> 且 <math>a[i] &gt; a[j]</math> 进行计数, 若是偶数, 答案加 1, 否则答案减 1。问所有方案进行后, 答案模一个质数 <math>p</math> 的值。 <math>n \leq 600</math>。</p>	<p>行列式求值。首先需要处理一个方案中, <math>k</math> 条路径并不独立的问题: 因为不能再顶点处相交, 但是考虑一个点 <math>p</math>, 从 <math>a[u]</math> 到 <math>u</math> 经过 <math>p</math>, 从 <math>a[v]</math> 到 <math>v</math> 经过 <math>p</math>, 即 <math>a[u] \sim p \sim u, a[v] \sim p \sim v</math>, 那么也一定存在一种方案, <math>a[u] \sim p \sim v, a[v] \sim p \sim u</math>, 正好在一种方案中是一个逆序对, 在另一种方案中不是。所以不考虑顶点处不能相交这个条件, 答案是一样的! 记 <math>\text{sgn}(p)</math> 表示排列 <math>p</math> 的符号, 若有偶数个逆序对则是 1, 否则是 -1, 再预处理出 <math>d[i][j]</math> 表示从源 <math>j</math> 走到汇 <math>i</math> 的路径数, 那么答案就是 <math>\sum \{\text{sgn}(p) \cdot \prod \{d[i][p[i]]\}\}</math>, <math>\sum</math> 表示对所有排列 <math>p</math> 答案相加, <math>\prod</math> 表示对所有源 <math>p[i]</math> 走到汇 <math>i</math> 的方案数乘起来。其实这个式子就是行列式的定义! 因为模是质数, 所以直接高斯消元消成三角矩阵, 把对角线上的值乘起来就是答案了。时间复杂度大概是 <math>O(n^3)</math>。</p>	Accepted

232D	李凌霄	Fence	<p>给一个长为 <math>n</math> 的数列 <math>h</math>, <math>n \leq 10^5</math>, 称 <math>[l1, r1]</math> 和 <math>[l2, r2]</math> 是匹配的, 当且仅当两区间长度相同且不相交, 对所有 <math>0 \leq i \leq r1 - l1</math>, 有 <math>h[l1+i] + h[l2+i] = h[l1] + h[l2]</math>。有 <math>q</math> 个询问, 每个询问是一个区间 <math>[l, r]</math>, 问有多少区间与它匹配。</p>	<p>后缀数组 + 树状数组。首先可以把条件变形, 得到 <math>h[l1+i] - h[l1] = h[l2] - h[l2+i]</math>, 即我们把两区间分别差分后, 对应位置互为相反数。构造数列 <math>h</math> 的差分数列 <math>s1</math>, 把 <math>s1</math> 每个数取相反数为 <math>s2</math>, 构造 <math>s = s1 + \\$ + s2</math>, <math>\\$</math> 是比其他数字都大的一个分隔符。问题就变成了, 对于 <math>s1</math> 的某个子串, 在 <math>s2</math> 中有多少个与它相同, 且不相交。求出 <math>s</math> 的后缀数组, 设询问串为 <math>t</math>, 长度为 <math>l</math>, 那么与以 <math>t</math> 开头的后缀 LCP 不小于 <math>l</math> 的都可能是解, 并且这些后缀在排序后是连续的, 设第 <math>i</math> 个询问的这个连续的区间是 <math>[st[i], ed[i]]</math>, 表示排名第 <math>st[i]</math> 的串到排名 <math>ed[i]</math> 的串。记 <math>pos</math> 数组, 对于排名第 <math>i</math> 的后缀, 若它在 <math>s2</math> 中, 那么 <math>pos[i]</math> 等于它对应的原位置, 否则 <math>pos[i] = 0</math>。那么问题转化成了, 一个 <math>pos</math> 数组, 若干个询问, 每次询问一段区间中, 大于等于某个数的值有多少, 小于等于某个数的值有多少。因为没有修改操作, 可以通过排序 + 树状数组来解决。时间复杂度 <math>O((n+q)\log n)</math>。</p>	Accepted
------	-----	-------	--	---	----------

175E	李凌霄	Power Defence	<p>塔防：一个怪物从 <math>x</math> 轴负无穷走向 <math>x</math> 轴正无穷，速度 <math>1/s</math>。可以再 <math>y=1</math> 和 <math>y=-1</math> 两条直线的整点上建塔，每个位置最多建一个塔。有三种塔：<math>nf</math> 个火塔，在一个半径 <math>rf</math> 圆内造成 <math>df/s</math> 的伤害；<math>ne</math> 个电塔，在一个半径为 <math>re</math> 圆内造成 <math>de/s</math> 的伤害；<math>ns</math> 个冰塔，在半径 <math>rs</math> 的圆内造成减速，若怪被 <math>k</math> 个冰塔减速，那么速度变成 <math>1/(k+1)</math>。三种塔数目之和不超过 20。求最大可能伤害。</p>	<p>枚举+动态规划。可以把塔都建在 <math>y=1</math> 这条线上，然后每个位置最多建 2 个。可以发现建塔有些贪心的规则：塔必定连续放置；不存在相邻的两个位置，都只放了一个塔。这样一来大概只有 13 种放塔的方式。由题目知，冰塔与众不同还有减速效果，所以枚举哪些位置放冰塔，然后设剩余了 <math>n</math> 个位置，通过线段交一类的算法算出第 <math>i</math> 个位置放火塔的伤害 <math>f[i]</math>，第 <math>i</math> 个位置放电塔的伤害 <math>e[i]</math>。那么可以有一个 <math>dp</math>，<math>dp[i][j][k]</math> 表示前 <math>i</math> 个位置，用 <math>j</math> 个火塔 <math>k</math> 个电塔能造成的最大伤害。这个动态规划的复杂度是 <math>O(n^3)</math>，对枚举出的方案都 DP 一次就行了。要控制一下常数。</p>	Accepted
176D	吕可凡	Hyper String	<p>给 <math>n</math> 个基本串，定义一个字符串 <math>s</math>，由 <math>m</math> 个基本串一次连接而成，再给出一个长度不超过 2000 的串 <math>t</math>，问 <math>s</math> 和 <math>t</math> 的最长公共子序列。所有串只由字母组成，<math>n \leq 2000</math>，</p>	<p>动态规划。考虑一个简化的问题：对一个长为 <math>10^6</math> 的串 <math>a</math>，和一个长为 <math>10^3</math> 的串 <math>b</math>，只由小写字母组成，求 LCS。这个可以以 <math>b</math> 串为基础进行 DP，记 <math>dp[i][j]</math> 表示 <math>b</math> 串前 <math>i</math> 个字符与 <math>a</math> 串有一个长为 <math>j</math> 的 LCS 时，在 <math>a</math> 中子序列右端点的最小值。那么 <math>dp[i][j]</math> 首先可以直接从 <math>dp[i-1][j]</math> 转移过来，现在考虑如何从 <math>dp[i-1][j-1]</math> 转移过来，设 <math>pos=dp[i-1][j-1]</math>，那么就找到 <math>a</math> 串 <math>pos</math> 后面第一</p>	Accepted

			<p><math>m \leq 2000</math>, 基本串长度之和不超过 <math>10^6</math>。</p>	<p>个等于 <math>b[i]</math> 的字符, 设位置是 <math>q</math>, 那么就可以用 <math>q</math> 来更新 <math>dp[i][j]</math>, 所以需要预处理出 <math>a</math> 中每个位置之后每个字母第一个出现的位置。这个可以 <math>O(26 \cdot 10^6)</math> 做到, 然后 DP 的复杂度就是 <math>O((10^3)^2)</math>。对于原题目, 不过是多了一个 <math>m</math> 个基本串的条件, 用一样的方法, 只是转移的时候可能会从一个基本串跳到另外一个基本串, 所以得需处理这 <math>m</math> 个基本串的序列, 每个位置往后每个字符第一次出现的块。设 <math>n</math> 个基本串长度之和为 <math>w</math>, 那么时间复杂度为 <math>O(26 \cdot w + 26 \cdot m +  t ^2)</math>。</p>	
178F2	吕可凡	Representative Sampling (30 points)	<p>给 <math>n(n \leq 2000)</math> 个长度不超过 500 的串, 要选 <math>k</math> 个串出来, 使得两两串最长公共前缀长度之和最大。</p>	<p>动态规划。一般想法是设 <math>dp[l][r][t]</math> 表示从第 <math>l</math> 个串到第 <math>r</math> 个串选择 <math>t</math> 个串的最大值, 然后枚举中间断点 <math>k</math>, 从 <math>dp[l][k]</math> 及 <math>dp[k+1][r]</math> 转移过来, 但是有一个问题是, <math>[l, k]</math> 中选的串与 <math>[k+1, r]</math> 中选的串配对后对答案的增量无法计算, 而且 <math>k</math> 的取值不那么重要, 甚至随便取一个, 只要能算答案都是正确的。所以考虑对所有串排序, 记 <math>lcp[i]</math> 表示排序后 <math>i</math> 串与 <math>i-1</math> 串的最长公共前缀长度, 那么可以考虑从 <math>[l+1, r]</math> 上取 <math>lcp</math> 最小的位置 <math>p</math>, 然后断开成 <math>dp[l][p-1]</math> 和 <math>dp[p][r]</math>, 这时, 因为 <math>lcp[p]</math> 是最小值, 串又是排了序的, 所以 <math>[l, p-1]</math> 的串与 <math>[p, r]</math> 的串的最长公共前缀都是 <math>lcp[p]</math>。然后枚举 <math>i, j</math>, 那么 <math>dp[l][r][i+j]</math> 可以用 <math>dp[l][p-1][i] + dp[p][r][j] + i \cdot j \cdot lcp[p]</math> 来更新。时间复杂度大概就是 <math>O(n \log n \cdot m + n^2 \cdot \log n)</math>, <math>m</math> 是串的长度。</p>	Accepted

178E3	何琦	The Beaver's Problem - 2 (50 points)	一个 $n \times n$ 的像素矩阵，每个格子是黑色或者白色，本来上面有一些正方形和圆形，但是每个格子都有 20% 的概率反色。问几个圆形几个正方形。	先进行“降噪”处理：对一个格子，看以它为中心的 $5 \times 5$ 矩阵，如果超过一半的格子是黑色，就认为它是黑色。然后找到每一个大小超过某个阈值 $T$ 的联通块，判断它是圆形还是正方形。判断方法：找到块的边界，并求一个块的中心，求出边界上的点到中心的距离的最大值和最小值，设为 $\max d$ 和 $\min d$ ，若 $\max d / \min d$ 在 1 左右，就是圆，在 $\sqrt{2}$ 左右，就是正方形。时间复杂度 $O(n^2)$ 。	Accepted
180B	何琦	Divisibility Rules	考虑整除。在 10 进制下，被 2 整除只用考虑最后 1 位，这种只用考虑最后几位的称作 2-type；被 3 整除要考虑所有数码的和，这种称作 3-type；被 11 整除，要用偶数位的和减去奇数位的和，称作 11-type；要几种组合，是 6-type；否则是 7-type。给出进制 $b$ 和除数 $d$ ，判断类型。 $2 \leq b, d \leq 100$ 。	数论。2-type 的特点是，总存在一个 $x$ ，使得 $d \mid b^x$ ；3-type 的特点是，对所有 $x$ ，有 $b^x \% d = 1$ ，所以 $b \% d = 1$ ；11-type 的特点是，对所有偶数 $x$ ，有 $b^x \% d = 1$ ，对所有奇数 $x$ ，有 $b^x \% d = -1$ ，所以 $b \% d = -1$ 。这三种都很好判断，而对于 6-type，只需要对分解因数，分别判断就可以了。因为 6-type 的存在，所以时间复杂度大概是 $O(d)$ 。	Accepted
185D	李凌劫	Visit of the Great	求 $\text{LCM}(k^{(2^1)}+1, k^{(2^2)}+1, \dots, k^{(2^n)}+1)$	数论。对任意两数进行最大公约数分析，发现，若 $k$ 是偶数，那么两两最大公约数是 1，若 $k$ 是奇数，两两最大公约数就	Accepted



			$k^{(1+1)+1}, \dots, k^{(2^r)+1}$ 。 $1 \leq k \leq 10^6, 0 \leq l \leq r \leq 10^{18}, 2 \leq p \leq 10^9$ 且 $p$ 是质数。每个数据有 $10^5$ 组询问。	是 2。在这里讨论 $k$ 是偶数的情况，奇数类似。因为两两最大公约数是 1，所以最小公倍数就是把它们全部乘起来。设 $a = k^{(2^l)}$ ，那么就是求 $(a+1) * (a^2+1) * (a^4+1) * \dots$ ，注意到，往这个因式里面乘 $(a-1)$ ，最后会变成一个 $a^x - 1$ 的形式。记 $b[i] = k^{(2^i)}$ ，那么其实乘起来就是 $(b[r+1]-1)/(b[1]-1)$ 。因为 $p$ 是质数，所以除法可以实现。每组的时间复杂度都是 $O(\log(p))$ ，要注意判掉各种特殊情况。	
187D	李凌劼	BRT Contract	一条线上 $n$ 个红绿灯， $n+1$ 端路，每段路有一个通过需要的时间。所有红绿灯都是同步的，在 0 时刻恰好变绿，绿灯时长持续 $g$ ，红灯时长持续 $r$ 。有 $q$ 个询问，每个询问一个 $t$ ，表示 $t$ 时刻从路的一端出发，到达另一端的时刻。 $n, q \leq 10^5$ 。	树状数组。思路：预处理从每个路口，恰好变成绿灯时出发，到达终点的时间。然后询问就是对每个询问找到第一个遇到的红灯。考虑回答询问，出发时刻为 $t$ ，因为红绿灯周期 $m = g + r$ ，所以把 $t$ 模 $m$ 后所花的时间是不变的。记 $sum[i]$ 表示不考虑红灯，从开头走到第 $i$ 个路口花的时间模 $m$ 的值，我们就是要找最小的 $k$ ，使得 $(t + sum[k]) \% m \geq g$ 。分两种情况讨论： $t \leq g$ ，则 $g - t \leq sum[k] < g + r - t$ ； $t > g$ ，则 $0 \leq sum[k] < g + r - t$ 或 $2 * g + r - t \leq sum[k] < g + r$ 。所以问题就是求 $sum$ 中值在某个区间的下标最小值。可以对 $sum$ 离散化后记 $pos[i]$ 表示值为 $i$ 的最小下标。这时可以用线段树来维护，不过注意到，如果枚举 $i$ 从大到小插入 $sum[i]$ ，即每次修改 $pos$ 中的值都是减小，就可以用树状数组来维护了。预处理与回答询问类似。最后的时间复杂度 $O((n+q)\log n)$ 。	Accepted

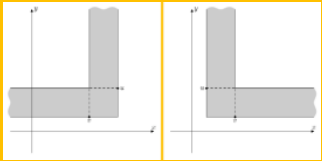
176E	陈睿	Archaeology	<p>给出一个 <math>n</math> 的点的树 (<math>n \leq 10^5</math>), 一开始所有点状态为 0。有三个操作: 把一个点的状态设成 0, 把一个点的状态设成 1, 询问把所有状态为 1 的点连在一起的最小生成树边权和。操作不超过 <math>10^5</math>。</p>	<p>数据结构。任取一个点为根建一棵有根树, 设当前状态为 1 的点集合为 <math>S</math>, 按 Dfs 序排序后得到序列 <math>T</math>。考虑答案由哪些部分组成。设序列 <math>T</math> 长为 <math>m</math>, 那么对 <math>2 \leq i \leq m</math>, <math>T[i]</math> 到 <math>T[i-1]</math> 与 <math>T[i]</math> 的最近公共祖先这段路是必须的, 且不互相重叠; 这样会少算 <math>T[1]</math>, 而少算的部分正好是 <math>T[1]</math> 到 <math>T[1]</math> 与 <math>T[m]</math> 的最近公共祖先的距离。所以我们可以对这棵树预处理一个倍增数组, 来计算最近公共祖先; 然后以 Dfs 序为关键字维护当前在 <math>S</math> 内的点, 插入、删除时用增量来维护答案, 这个可以用平衡树实现, 或者 C++ 中的 <code>set</code>。时间复杂度 <math>O(n \log n + q \log n)</math>。</p>	Accepted
196D	陈睿	The Next Good String	<p>给一个仅由小写字母组成的字符串 <math>S</math>, 和一个正整数 <math>m</math>, 要求一个长度与 <math>S</math> 相同且仅由小写字母组成的字符串 <math>T</math>, 要求 <math>T</math> 的字典序大于 <math>S</math>, 且 <math>T</math> 中不包含长度大于等于 <math>m</math> 的回文子串, 若有多个 <math>T</math> 求出字典序最小的。 <math>S</math> 长度 <math>\leq 4 \times 10^5</math>。</p>	<p>暴力+哈希。先把问题转化成求的 <math>T</math> 的字典序不小于 <math>S</math>, 然后从第一位开始逐位枚举。判断不存在长度不小于 <math>m</math> 的回文子串, 只用判断当前位置往前 <math>m</math> 长度以及 <math>m+1</math> 长度是不是回文子串。设最早在第 <math>i</math> 位有 <math>T[i] &gt; S[i]</math>, 那么之后的字符可以随便填了, 之后可以接 <code>aa...abb...bcc...caa...a</code> 诸如此类的, 即一旦在某个位置 <math>T[i] &gt; S[i]</math> 了, 后面能够在线性时间构造出一个字典序极小的后缀。用哈希来判断一个子串是否为回文串, 可以做到 <math>O(1)</math>。所以整个时间复杂度是 <math>O(n)</math>, <math>n</math> 是 <math>S</math> 串的长度。</p>	Accepted
198E	钱迪晨	Gripping Story	<p>考虑在一个二维平面中, 在 <math>(x, y)</math> 处有一个吸引力</p>	<p>数据结构+广搜。广搜是明显的, 因为有一个贪心的结论, 每次我们都会把当前磁铁能吸过来的磁铁都吸过来, 于是我</p>	Accepted

			<p>为 <math>p</math> 吸引半径为 <math>r</math> 的磁铁，其他地方还散落着 <math>n</math> 个磁铁 (<math>n \leq 250000</math>)，第 <math>i</math> 个磁铁在 <math>(x[i], y[i])</math> 位置，质量为 <math>m[i]</math> 吸引力为 <math>p[i]</math> 吸引半径为 <math>r[i]</math>。每次可以从已有的磁铁中选一个出来，把另外一个磁铁吸过来，如果被吸的磁铁在吸引半径中且重量不超过吸引力。问最后手上能有多少磁铁。</p>	<p>们需要维护一个东西，找出第一关键字不小于 <math>R</math>，第二关键字不小于 <math>P</math> 的所有磁铁，把它们加入队列，然后把它们从数据结构中删除。这个可以用线段树套平衡树来实现，我使用的是树状数组套 <math>\text{set}</math>，时间复杂度是 <math>(n(\log n)^2)</math>。</p>	
200E	钱迪晨	Tractor College	<p>一共有 <math>n</math> (<math>n \leq 300</math>) 个人，分三类，分别有 <math>c_3</math>、<math>c_4</math>、<math>c_5</math> 个，要发完 <math>m</math> 块钱 (<math>m \leq 3 \times 10^5</math>)，这三类每类每个人发的钱应该一样，且是 <math>w_3</math>、<math>w_4</math>、<math>w_5</math>，要求 <math>0 \leq w_3 \leq w_4 \leq w_5</math>。在 <math>m</math> 块钱用完的情况下，最小化 <math>\text{abs}(w_3 \cdot c_3 - w_4 \cdot c_4) + \text{abs}(w_4 \cdot c_4 - w_5 \cdot c_5)</math>，请</p>	<p>数论+三分。考虑在式子中 <math>w_4 \cdot c_4</math> 出现了两次，不妨以它为突破点来搞。枚举 <math>w_4</math> 的值，设 <math>Q = w_4 \cdot c_4</math>，<math>P = m - Q</math>，那么有 <math>c_3 \cdot w_3 + c_5 \cdot w_5 = P</math> 这个不定方程，可以用扩展欧几里得找到一组解，通过二分找到使得 <math>0 \leq w_3 \leq w_4</math> 及 <math>w_5 \geq w_4</math> 的 <math>w_3</math> 的范围。注意 <math>\text{abs}(w_3 \cdot c_3 - w_4 \cdot c_4) + \text{abs}(w_4 \cdot c_4 - w_5 \cdot c_5) = \text{abs}(Q - w_3 \cdot c_3) + \text{abs}(Q - w_5 \cdot c_5) = \text{abs}(Q - w_3 \cdot c_3) + \text{abs}(Q - (P - w_3 \cdot c_3)) = \text{abs}(w_3 \cdot c_3 - Q) + \text{abs}(w_3 \cdot c_3 - (P - Q))</math>，令 <math>a = Q</math>，<math>b = P - Q</math>，则上式 <math>= \text{abs}(w_3 \cdot c_3 - a) + \text{abs}(w_3 \cdot c_3 - b)</math>，这个式子关于 <math>w_3</math> 是单峰的，所以可以三分找到最优解。时间复杂度 <math>O(m \cdot \log(m))</math>。</p>	Accepted

			确定 w3、w4 和 w5。		
200A	李宇轩	Cinema	<p>一个 <math>n*m</math> 的矩阵，有 <math>k</math> 个询问，每个询问是一个坐标 <math>(x,y)</math>，要找到一个坐标 <math>(a,b)</math>，使得它未在之前的询问中出现过，满足这个条件后要使得两点间曼哈顿距离最小，若多个满足选 <math>a</math> 最小的，还有多个满足选 <math>b</math> 最小的。  <math>1 \leq n, m \leq 2000</math> ,  <math>k \leq \min(n*m, 10^5)</math>。</p>	<p>暴力+并查集。注意到以 <math>(x,y)</math> 为中心，曼哈顿距离小于等于 <math>d</math> 的坐标们在一般情况下是 <math>O(d^2)</math> 的，即每个询问的答案 <math>(a,b)</math> 到 <math>(x,y)</math> 的曼哈顿距离是 <math>O(\sqrt{k})</math> 的。我们从小到大枚举 <math>x</math> 与 <math>a</math> 的差值 <math>t</math>，分别找出 <math>(x-t,y)</math> 和 <math>(x+t,y)</math> 左右最近的未出现的点，更新答案，不考虑找点的操作，时间复杂度是 <math>O(\sqrt{k})</math> 的。找点的操作可以对每一行建并查集，以找某个点左边最近的点为例，若某一行上纵坐标为 <math>y</math> 的被选了，那么就在并查集中把 <math>y</math> 连向 <math>y-1</math>，这样每次询问 <math>y</math> 所在子树的根就是最近的一个没被选的点。这样时间复杂度就是 <math>O(k*\sqrt{k})</math>，但是有一个问题，在矩阵较“窄”的时候会退化，所以当 <math>n&gt;m</math> 时交换 <math>n</math> 和 <math>m</math> 再处理即可。</p>	Accepted
201E	李宇轩	Thoroughly Bureaucratic Organization	<p>有一个长为 <math>n</math> 的排列，每次可以做多询问 <math>m</math> 个位置的数的集合是什么，求最少询问次数确定这个排列。<math>m \leq n \leq 10^9</math>。</p>	<p>二分+组合计数。显然询问次数是可以二分的，设询问次数为 <math>k</math>，那么有至多 <math>k*m</math> 个询问的位置。问题在于如何区分两个位置的数，即，在这 <math>k</math> 次询问中，位置 <math>i</math> 出现的询问集合不能与位置 <math>j</math> 出现的询问集合相同。而每个位置 <math>i</math> 最多寻问 <math>k</math> 次。我们枚举 <math>x</math> 从 0 到 <math>k</math>，那么询问 <math>x</math> 次的位置不能超过 <math>C(k,x)</math> 个，<math>C</math> 是组合数，而这 <math>C(k,x)</math> 的位置占据的询问位置是 <math>C(k,x)*x</math> 个，我们可以统计出满足 <math>n</math> 个位置的询问，总共需要的询问位置数，并把它和 <math>k*m</math> 比较，来确定下一次二分的左右边界。因为组合数增长很快，所以枚举 <math>x</math> 的复杂度是 <math>\log(n)</math> 的，这样对于每组询问的复杂度做到了</p>	Accepted

				$O((\log n)^2)$ 。	
201D	孟凡航	Brand Problem New	有 $n$ 个单词组成一句话 $s[0]$ , 以及 $m$ 个由若干单词组成的长句子 $s[i]$ , 若 $s[0]$ 的某一个排列是某个句子 $s[i]$ 的子序列, 则称两句子相似, 定义差异度 $p$ 为这个排列的逆序对数, 对 $m$ 个句子, 求出最小的 $p$ 。 $n \leq 15, m \leq 10$ 。	状态压缩动态规划。对每个长句子分别 DP。记 $dp[inv][mask]$ 表示长句子出现 $mask$ 中的 $s[0]$ 中的单词, 且逆序对已有 $inv$ 个, 对长句子这个子序列的右端点的最小值。考虑转移, 对某个不在 $mask$ 中的单词, 设它在 $s[0]$ 中的序号是 $p$ , 我们找出这个单词在长句子中位置在 $dp[inv][mask]$ 后最早出现的位置 $q$ , 然后用 $q$ 更新 $dp[inv'][mask   1 \ll p]$ , $inv'$ 是新的逆序对数, 用 $inv$ 加上 $mask$ 中比 $p$ 大的数即是 $inv'$ 。对每个长句子复杂度 $O(2^n * n^2)$ , 整体时间复杂度就是 $O(m * 2^n * n^2)$ 。	Accepted
204E	孟凡航	Little Elephant and Strings	给 $n$ 个字符串, 问每个字符串有多少子串是所有 $n$ 个字符串中至少 $k$ 个字符串的子串。 $n \leq 10^5$ 。	后缀数组+线段树。先把所有字符串拼在一起 (相邻字符串加一个字典序较大的分隔符), 求一遍后缀数组。问题变成, 找出所有的 $[l, r]$ , 满足排名 $l$ 到 $r$ 的后缀, 所属的串不小于 $k$ 个, 然后在线段树中维护最大值, 用 $[l, r]$ 所有串的最长公共前缀长度在线段树中更新 $[l, r]$ 这一段。但是 $[l, r]$ 的数目依然很大, 但是我们对一个 $l$ , 只需找出最小的 $r$ , 实现方式可以维护两个指针 $l$ 和 $r$ 往前滑动, 这样就是线性的。最后用一个数据结构维护 $[l, r]$ 上最长公共前缀长度, 用后缀数组求出 $height$ 值, 那么这个长度就是 $l+1$ 到 $r$ 中 $height$ 的最小值, 因为 $[l, r]$ 是向前“滑动”的, 所以可以用单调队列来维护。时间复杂度 $O(m \log m)$ , $m$ 为所有字符串的长度和。	Accepted

207B1	龚拓宇	Military Trainings points) (20	<p>n 个人, 每个人有一个接收半径 <math>a[i]</math>, 若 <math>i</math> 位置的人向 <math>j</math> 位置的人传消息, 那么设 <math>j</math> 位置上人的接收半径为 <math>r</math>, 那么有 <math>i &lt; j</math>, <math>i \geq j - r</math>。一开始所有人按 1 至 <math>n</math> 站好, 1 要把一个消息传给 <math>n</math>; 然后 <math>n</math> 站到队首, 由 <math>n</math> 把信息传到 <math>n-1</math>, 这样迭代 <math>n-1</math> 次。问最少信息需要传递的总次数。 <math>n \leq 250000</math>。</p>	<p>倍增。把信息传递的方向取反, 那么每次都是发射信息, <math>i</math> 位置发射向 <math>j</math> 位置, 有 <math>i &gt; j</math> 且 <math>i - r \leq j</math>, 这样可以保证每个人的可能的后继都是一段连续的区间, 这时可以对每个人 <math>i</math> 记录一个它往前能发射到的最小位置 <math>p[i]</math>。那么有一个贪心的结论, 当在第 <math>i</math> 个人, 且最小能发射到 <math>p[i]</math>, 那么应该选择最小的 <math>j</math> 使得 <math>p[i] \leq j &lt; i</math>, 且 <math>p[j]</math> 最小。这个可以转化成区间取最小值问题, 用 ST 算法可以做一个预处理, 那么能得到每个位置 <math>i</math> 往前发射时应该到的位置 <math>q[i]</math>。对于一次信息传送就可以 <math>O(n)</math> 了。因为有 <math>n</math> 次信息传递, 所以还有优化。把序列扩充成 <math>1, 2, \dots, n, 1, 2, \dots, n</math>, 那么信息传递就是对 <math>1 \leq i \leq n</math>, 从 <math>n+i</math> 传递到 <math>i+1</math>, 这时对这 <math>2*n</math> 个位置都算出 <math>q[i]</math>, 预处理一个倍增数组 <math>dp[k][i]</math> 表示从 <math>i</math> 位置走 <math>2^k</math> 步到的位置。这样对于每次信息传递可以做到 <math>O(\log n)</math>, 于是整个时间复杂度就是 <math>O(n \log n)</math>。</p>	Accepted
207A2	龚拓宇	Beaver's Calculator points) (30	<p><math>n</math> 个人, 第 <math>i</math> 个人有 <math>k[i]</math> 个需求, <math>a[i][j]</math> 表示第 <math>i</math> 个人第 <math>j</math> 个需求的花费。你要完成所有的需求, 且对于每个人他的需求的顺序不能改变, 求一种安排方案, 使得“坏对”尽量少, 所谓“坏对”; 即对于你的工作序列中两个相邻的花费</p>	<p>贪心。可以算出对第 <math>i</math> 个人, 他的需求中坏对的数量 <math>cnt[i]</math>, 我们能够构造一个解, 使得坏对数就是 <math>cnt[i]</math> 中的最大值。对每个人 <math>i</math>, 把它的需求按坏对划成 <math>cnt[i]+1</math> 段, 从前往后从 1 到 <math>cnt[i]+1</math> 标号, 设 <math>cnt[i]</math> 中的最大值是 <math>t</math>, 那么标号至多到 <math>t+1</math>, 因为每一块都是不降的, 所以我们把标号相同的块合并成一个不降的序列, 那么标号相同的块中不会有坏对, 所以至多有 <math>t</math> 个坏对; 而取一个 <math>x</math> 使得 <math>cnt[x]=t</math>, 对于相邻的两块 <math>i</math> 和 <math>j</math>, <math>x</math> 在 <math>i</math> 中的最大值是 <math>y</math>, 在 <math>j</math> 中的最小值是 <math>z</math>, 由坏对的定义知 <math>y &gt; z</math>, 所以 <math>i</math> 和 <math>j</math> 之</p>	Accepted

			a 和 b, 若 $a > b$ , 则是一个坏对。	间必有一个坏对, 所以至少也有 $t$ 个坏对。所以答案就是 $t$ , 用堆维护或直接排序就可以了。时间复杂度 $O(m \log m)$ , $m$ 为所有人需求之和。	
167D	张闻涛	Wizards Roads and	<p>给出平面上 <math>n</math> 个点, 保证点之间两两 <math>x</math> 值不同, 两两 <math>y</math> 值不同, 且大致是随机分布的。点 <math>u</math> 和点 <math>v</math> 能连边 (<math>y[u] &gt; y[v]</math>), 当且仅当对任何在 <math>u, v</math> 形成的拐角中的点 <math>w</math>, 都有一个点 <math>s</math> 不在拐角中, 且 <math>y[s] &gt; y[v]</math>, <math>x[s]</math> 在 <math>x[w]</math> 与 <math>x[u]</math> 之间。”拐角”的定义如图:</p>  <p>有 <math>m</math> 个询问, 每次询问为 <math>[L, R]</math>, 只考虑横坐标在 <math>[L, R]</math> 上的点, 每个点最多连一条边, 最多能连多少条边。 <math>n, m, \leq 10^5</math>。</p>	<p>数据结构+匹配。考虑第 1 个点到第 <math>r</math> 个点, 哪些能连边。设中间 <math>y</math> 最大的点是第 <math>k</math> 个点, 那么对于其左边的某个点 <math>u</math>, 和右边的某个点 <math>v</math>, <math>k</math> 必在 <math>u</math> 和 <math>v</math> 形成的拐角中, 而显然是不可能找出点 <math>s</math> 满足 <math>x[s]</math> 在 <math>x[u]</math> 和 <math>x[k]</math> 之间, 且 <math>s</math> 不在拐角中。所以 <math>k</math> 左边的点与右边的点不可能连边。于是点集被割成了两个独立的区间。注意到, 这样做下去可以得到一棵二叉树, 而我们要完成的操作就是对 1 号点到 <math>r</math> 号点构成的二叉树中求最大匹配, 而树上的匹配是可以用 <math>dp</math> 完成的。所以对每个点记录 <math>dp[i][0]</math> 表示不选 <math>i</math> 号点在 <math>i</math> 子树的最大匹配数, 及 <math>dp[i][1]</math> 表示选 <math>i</math> 号点在 <math>i</math> 子树的最大匹配数, 然后每次询问的话就是合并一系列信息。因为点时随机分布的, 所以二叉树的深度大概是 <math>O(\log n)</math>, 所以时间复杂度就是 <math>O(n \log n + m \log n)</math>, 其中 <math>O(n \log n)</math> 是构造这棵二叉树的复杂度。</p>	Accepted



209C	张闻涛	Trails and Glades	<p>给一个 <math>n</math> 个点 <math>m</math> 条边的无向图, 问至少添多少条边, 使这个图有欧拉回路 (必须过点 1)。 <math>n, m \leq 10^6</math>。</p>	<p>贪心。用并查集处理出块的数目 <math>t</math>, 设第 <math>i</math> 个块有 <math>a[i]</math> 个奇度数点, <math>b[i]</math> 个偶度数点。注意若有孤立的点 (若不是 1 号点) 则应该舍弃。把所有点按 <math>a[i]</math> 为第一关键字排序, 然后在相邻两块间连边, 一共连 <math>t</math> 条, 连成一个环。连边注意尽量都选择块中的奇度数点连。这样若图中还有 <math>k</math> 个奇度数点, 那么我们需要把它们变成偶度数, 就需添加要 <math>k/2</math> (往上取整) 条边。时间复杂度 <math>O(m+n\log n)</math>。</p>	Accepted
212B	孙伟	Polycarpus is Looking for Good Substrings	<p>给一个长度为 <math>n</math> 的仅由小写字母组成的字符串 <math>s</math>, 有 <math>m</math> 个询问, 每个询问时一个字符的集合 <math>t</math>, 问 <math>s</math> 中有多少子串 <math>[a, b]</math> 满足, 把 <math>[a, b]</math> 上的字母取出后去重后与 <math>t</math> 相同, 且没有 <math>x \leq a \leq b \leq y</math>, 使得 <math>[x, y]</math> 比 <math>[a, b]</math> 长并同时满足条件。 <math>n \leq 10^6, m \leq 10^4</math>。</p>	<p>状态压缩。先把所有询问离散化, 方便记录答案。预处理一个 <math>next[i][j]</math> 表示 <math>i</math> 位置往后, 字母 <math>j</math> 出现的最早位置。然后枚举左端点 <math>l</math>, 那么至多只有 26 个右端点是有用的, 不妨利用 <math>next</math> 数组和当前字符集合 <math>mask</math> 求出下一个 <math>r</math>。因为是从小到枚举 <math>l</math>, 所以对于相同的 <math>mask</math>, 目前的 <math>r</math> 应该比之前的 <math>r</math> 大, 所以要对每个 <math>mask</math> 记录一个 <math>val[mask]</math> 表示 <math>mask</math> 存在的区间的最右端点。时间复杂度 <math>O(26*n+m\log m)</math>。</p>	Accepted
212D	孙伟	Cutting a Fence	<p><math>n</math> 个木板, 第 <math>i</math> 个木板高 <math>h[i]</math>, 若在 <math>[x, x+k-1]</math> 上画一个宽为 <math>k</math> 的矩形, 那么高度就是 <math>x</math> 到 <math>x+k-1</math></p>	<p>计数+数据结构。先把问题转化成宽为 <math>w</math> 的所有矩形的高度和。考虑每个 <math>h[i]</math> 对答案的贡献, 这样考虑: <math>h[i]</math> 作为一段区间的最小值, 且是区间上最靠左的最小值, 对每个 <math>i</math> 求出 <math>[lb, rb]</math>, 表示 <math>h[lb-1]</math> 是左边第一个不大于 <math>h[i]</math> 的,</p>	Accepted



			<p>最小的 <math>h</math>。给 <math>m</math> 个询问，每个询问是 <math>w</math>，问宽是 <math>w</math> 的矩形的平均高度。  <math>n, m \leq 10^6</math>。</p>	<p><math>h[rb+1]</math> 是右边第一个小于 <math>h[i]</math> 的，设 <math>x=i-lb+1, y=rb-i+1</math>，那么就有它对 <math>x*y</math> 个区间有贡献。不妨设 <math>x \leq y</math>，对每对 <math>p, q</math>，满足 <math>0 \leq p &lt; x, 0 \leq q &lt; y</math>，<math>h[i]</math> 就会对宽为 <math>p+q+1</math> 的矩形高度和贡献一次。设宽度减一为 <math>t</math>，那么对 <math>0 \leq t &lt; x</math>，<math>(p, q)</math> 对数是 <math>t+1</math>；对 <math>x \leq t &lt; y</math>，<math>(p, q)</math> 对数是 <math>x+1</math>，对 <math>y &lt; t \leq x+y</math>，<math>(p, q)</math> 对数是 <math>x+y-t+1</math>，容易发现，这三段贡献值都是等差数列，于是我们以矩形宽度为下标建一个线段树，那么操作就是每次在一段区间上加一个等差数列，或者询问某个点的值。这个通过打标记就可以实现。时间复杂度 <math>O(n \log n)</math>。</p>	
212C	周誉昇	Cowboys	<p>一个长度为 <math>n</math> 的环上，每个人要么面向他的顺时针方向，要么面向他的逆时针方向。在某个时刻，若两个人发现他们是面对着的，他们会同时转身，而且所有这样的对会同时动。给出下一时刻的状态，问上一时刻可能的状态数。  <math>3 \leq n \leq 100</math>。</p>	<p>动态规划。记 <math>dp[i][a][b]</math> 表示到第 <math>i</math> 个人，前一时刻第 <math>i</math> 个人状态是 <math>a</math>，第 <math>i-1</math> 个人的状态是 <math>b</math>，能够变成现在状态的状态数。那么可以通过枚举第 <math>i+1</math> 个人的状态，通过一系列条件判断来转移，因为是一个环，所以一开始不妨枚举第 1 个人和第 <math>n</math> 个人的状态，然后再 DP。时间复杂度 <math>O(n)</math>。</p>	Accepted
213E	周誉昇	Two Permutations	<p>给一个长度为 <math>n</math> 的排列 <math>a</math>，再给一个长度为 <math>m</math> 的排列</p>	<p>哈希+数据结构。显然 <math>0 \leq d \leq m-n</math>，可以从小到大枚举 <math>d</math>，然后维护序列 <math>a</math> 的哈希值。哈希选用乘幂取模，比如原来的</p>	Accepted

			<p>b, 问有多少 d 满足, a 的每个数加上 d 后, 这个序列是 b 的一个子序列。</p>	<p>哈希是 <math>a \cdot p^2 + b \cdot p + c</math>, 那么 d 加 1 后增加了 <math>p^2 + p + 1</math>, 这时一个常数, 可以预处理。然后维护一个以元素在 b 中下标为关键字的平衡树, 当从 d 枚举到 d+1, 时, 删除 d+1 这个数对应的下标, 插入 n+d+1 这个数对应的下标, 同时维护整个哈希值。这样每次看看哈希值是否一致即可。时间复杂度 <math>O((m-n) \log n)</math>。</p>	
217C	许瀚云	Formurosa	<p>定义一种表达式, <math>s \rightarrow 0 1 ? (s s) (s \&amp; s) (s^s)</math>, 现在给出一个表达式, 和 n 个 0/1 变量, 保证这些变量不全相等, 每次可以把问号替换为一个变量, 你会得到算出来的值, 你可以进行任意次操作, 问能否确定这 n 个变量的值。 <math>2 \leq n \leq 10^6</math>。</p>	<p>表达式处理+动态规划。注意到, n 的值是没有意义的, 只要能解决 2 个变量的情况, 再多的变量都可以确定。记 <math>dp[l][r][a][b]</math> 表示 <math>[l, r]</math> 上的这个表达式, 把 0 和 1 填进去可以得到 a, 而在这种情况下把 0 换成 1, 1 换成 0, 可以得到 b, 是否可能。那么转移就是找到 <math>[l, r]</math> 上不在任何括号中的运算符的位置 p, 从 <math>[l, p-1]</math> 和 <math>[p+1, r]</math> 转移过来。虽然 <math>[l, r]</math> 是平方级别的, 但是用到的数目其实是线性的。找运算符 p 可以通过一些预处理, 比如利用 ST 算法在 <math>O(m \log m)</math> 预处理后 <math>O(1)</math> 得到, m 是表达式长度。这样可以先广搜找出所有有用的 <math>[l, r]</math>, 然后倒过来 DP, 这样可以避免递归。时间复杂度 <math>O(m \log m)</math>。</p>	Accepted
229E	许瀚云	Gifts	<p>要选 n 个数, 一共有 m 种数, 第 i 种数有 <math>k[i]</math> 个, 第 i 种数的第 j 个价值是 <math>c[i][j]</math>。而若在第 i 种数中选了 u 个, 那么得到</p>	<p>动态规划。设第 n 高的价值是 x, 而价值为 x 的数有 tot 个, 我么需要的有 cho 个; 设 <math>cnt[i]</math> 表示第 i 种数中价值超过 x 的个数。记 <math>dp[i][j]</math> 表示前 i 种数, 选了 j 个价值为 x 的, 得到最高的 n 个的概率和。那么边界条件就是 <math>dp[0][0] = 1.0</math>, 而 <math>dp[m][cho] / C(tot, cho)</math> 就是答案,</p>	Accepted

			<p>的是这种数中随机的 <math>u</math> 个。你选择了价值最高的 <math>n</math> 个数，而且若有多种选法就等概率的选择一种，问拿到 <math>n</math> 样价值最高的数的概率。<math>n, m \leq 1000</math>，所有 <math>k[i]</math> 的和不超过 1000。</p>	<p>这里的 <math>C</math> 表示组合数。再来考虑转移：有 <math>dp[i][j+o] = \sigma\{dp[i-1][j] * 1.0 / C(k[i], cnt[i] + o)\}</math>。这个动态规划的复杂度是 <math>O(cho * tot)</math> 的，所以整个时间复杂度大概是 <math>O(tot^2)</math> 的。</p>	
--	--	--	--	---	--