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的、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(nlogn)，最坏O(n^2)。 | Accepted |
| 8D | 贾志鹏 | Two Friends | 两人都需要从A点前往B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。 | 计算几何。可以发现同行的线段一定在ABC构成的三角形内，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。 | Accepted |
| 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\*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\*3^4\*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，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。 | 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’>=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)。 | Accepted |
| 15E | 陈立杰 | Triangles | http://codeforces.ru/renderer/5945210be972aa5fe947f3b8a3a0378a4cade844.png  定义了一种n层三角形如上图，最顶端为H点，只能沿三角形的边走动，求从H点出发并回到H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。 | 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H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，这就需要我们在统计    这种形状从绿色点走到紫色点的方案数。可以推导出，若有m个梯形，那么方案数就是1+sigma{4\*2^k|0<=k<m}。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O(n)的算法。 | 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sigma{left[i]\*sigma{right[j]|1<=j<i}|2<=i<=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。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cnt[i]加1，cnt[j+1]减1，这样就有left[k]=sigma{cnt[i]|1<=i<=k}。right的求法类似。时空复杂度都是O(n)。 | Accepted |
| 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e1和B集合的边e2，t在P(e1)和P(e2)上，那么去掉t，使得e1两端点异色的同时会导致e2两端点同色，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(nlogn)。 | Accepted |
| 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>=2有xy>=x+y（不妨设x>=y，则可得到xy>=2x=x+x>=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。不考虑高精度计算的复杂度的话，时空复杂度均为O(n^2)。 | Accepted |
| 26E | 向鹏达 | Multithreading | 给出n个这样的线程，其中y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y的值为m。 | 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(nlogn)。 | 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，然后到P，然后再走回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)。 | Accepted |
| 30E | 罗雨屏 | Tricky and Clever Password | 对一个奇长度回文串s，我们可以把它分为prefix+middle+suffix，其中prefix与suffix长度相同。再有三个可为空的串A、B、C，t=A+prefix+B+middle+C+suffix，现在给出t，求最长的s。 | 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middle是局部极长的（即对于最优解的中心，middle不能再长），于是可以先做一遍Manacher，然后枚举中心点，算出middle的区间[l,r]。注意到suffix是t串的一个后缀，而且在middle确定的情况下，suffix的长度是可以二分的，所以最后的问题是对于t的一个后缀o，找出rev(o)，即o串倒置，最早出现的位置。设t长度为n，令f(i)为长度为i的后缀，其倒置过后在t中最早出现的位置，即t[f(i)..f(i)+i-1]=t[n-i+1..i]。注意到，f(i)随i单调不减，因为若f(i+1)<f(i)，而长度为i的后缀是长度为i+1的后缀的后缀，所以f(i)应该取到更小的f(i+1)，矛盾。于是可以利用字符串哈希在O(n)时间把f预处理出来，这样整体复杂度做到了O(nlogn)。 | Accepted |
| 32E | 乔明达 | Hide-and-Seek | 平面上给出两点P、Q，和一堵线段墙W0W1，一面线段双面镜M0M1。问P  能否看到Q。 | 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O(1)。 | Accepted |
| 35E | 乔明达 | Parade | 给出n个底边在X轴上且在X轴以上的矩形，求出它们并起来的轮廓。 | 计算几何。在每个矩形的左右边界设立事件点，排序后进行扫描。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(nlogn)。 | Accepted |
| 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^2logn)。 | Accepted |
| 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+nlogn)。 | 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>=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>=n时，先手胜；当(a+1)^b>=n且a=1时，平局；当a\*a>=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<=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，使得每一行每一列的元素乘积都是-1。注意k<max(n,m)。 | 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>=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，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)。 | Accepted |
| 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\*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\*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\*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。 | Accepted |
| 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。 | 计算几何。有斜抛运动的规律知道，不超过45度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度时，角度越大，抛得越远。于是对m堵墙算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(mlogm+nlogm)。 | Accepted |
| 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|0<=a<i,0<=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，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。 | 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)|v是u的儿子}。然后再考虑在树上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)。 | Accepted |
| 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\*n^2)。 | Accepted |
| 57D | 王启圣 | Journey | 在一个n\*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。这样最后算出sum，答案就是sum/(cnt\*cnt)。时间复杂度O(nm)。 | Accepted |
| 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root[i]上的答案减去root[y]上的答案了。时间复杂度O(m(logn)^2)。 | Accepted |
| 217D | 汤沛雯 | Bitonix' Patrol | 给出t(t<=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<=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，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。 | 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。 | Accepted |
| 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wi；删除字符，费用wd；替换字符，费用wr；交换相邻两字符，费用we。求把第一个串变成第二个串的最小费用。保证2\*we>=wi+wd。 | 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\*we>=wi+wd，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(nlogn)。 | 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。 | 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。f1(t)表示以t为根的子树的最小花费，f2(t,g,u)表示在以t为根的子树内，点g作为点t连接的关键点，目前考虑到点u的最小花费。对f1(t)，枚举子树t内的一点g作为关键点，从f2(t,g,t)转移过来；对f2(t,g,u),考虑u的每一个儿子v，若v与u在同一子树内，则从f1(t,g,v)转移过来；若v与u不在同一子树内，则v到u的边断开，则从f2(v)转移过来。不妨以0号点做根，则答案是f2(0)。时空复杂度均为O(n^3)。 | Accepted |
| 156E | 高胜寒 | Mrs. Hudson's Pancakes | n个数a[i]，下标从0到n-1，n<=10000。有m个询问，每个询问给出一个d进制带通配符“？”的串，和一个数c，问所有下标i满足与串匹配的a[i]之积与c之和的最小质因子（大于100则输出-1）。m<=30000。 | 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int。记dp(b,m,s)为b进制下，以m为模，带通配符的串为s，满足的下标的乘积。这个dp不用全部递推出来，记忆化搜索即可。其中串s可以用哈希实现，因为有“？”所以可以做一个b+1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。 | Accepted |
| 105D | 高胜寒 | Entertaining Geodetics | 一个n\*m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(x,y)这个格子的物块开始操作。维护一个队列，第一步是把(x,y)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。 | 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O(nm)。 | Accepted |
| 193D | 雷凯翔 | Two Segments | 给出1到n的一个排列，问有多少1<=l1<=r1<l2<=r2<=n满足[l1,r1]及[l2,r2]上的所有数排序后构成公差为1的等差数列。n<=3\*10^5。 | 数据结构。对数列进行变换，设原数列是p[i]，令q[p[i]]=i，则问题在q上变成了：一段连续的区间，它们的数字构成几条线段。设f([l,r])表示q的[l,r]这个区间上的数构成了几条线段。f([n,n])显然是1，我们可以考虑如何从f([l+1,i])推出f([l,i])，记g(i)=f([l,i])-f([l+1,i])，g(l)显然也是1。设与q[l]相邻的两个数在q中的位置为a和b且a<b，那么对l<=i<a，在[l+1,i]中插入l是独立的，所以g(i)=1；对a<=i<b，在[l+1,i]中插入l是与某线段相邻，所以g(i)=0；对b<=i<=n，在[l+1,i]中插入l是连接了两条线段，所以g(i)=-1。这样可以用线段树很快的从f([l+1,i])推出f([l,i])，然后统计一段区间上1和2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(nlogn)。 | Accepted |
| 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]<=t[j]，则i可以到j的条件，可以推导出就是x[i]<=x[j]且-x[i]+v\*t[i]<=-x[j]+v\*t[j]，或者是x[i]>x[j]且x[i]+v\*t[i]<=x[j]+v\*t[j]。定义两个数组a和b，设a[i]=-x[i]+v\*t[i]，b[i]=x[i]+v\*t[i]，注意到x[i]<=x[j]显然满足b[i]<=b[j]，x[i]>x[j]也满足前面的a[i]<=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(nlogn)。 | Accepted |
| 76A | 郭志芃 | Gift | 给一个点数为n边数为m的无向图，每条边都有两个属性g和s。对一棵生成树，若边集中g的最大值为gm，s的最大值为sm，则花费为G\*gm+S\*sm注意G和S是常数。求最小费用。 | 生成树的应用。把所有边按g排序，然后顺次考虑每条边。然后应该每次算一个按s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O(mlogm+mn)。 | Accepted |
| 77E | 罗剑桥 | Martian Food | 一个大圆半径为R，一个内切在大圆里面的半径为r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k个加的圆的半径。 | 反演。在极坐标中考虑问题。把大圆圆心放置在(R,0)，小圆圆心放置在(r,0)，然后进行一个变换：把点按照(r,t)->(1/r,t)进行变换，则大圆和小圆都变成了一条直线，且方程分别是x=1/2R和x=1/2r。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k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O(1)。 | Accepted |
| 79D | 罗剑桥 | Password | 一开始有n个0，你需要把k个位置变成1，k不超过10。再给出m种操作，第i个操作是把一段长度为a[i]的区间都异或1。问最少操作的次数。 | 动态规划。设原序列为s，并设s[0]=s[n+1]=0，构造新序列t[i]=a[i] xor a[i+1]，那么新得到的t序列中至多有2k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n+1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u和v，t[u]=t[v]=1时，u走到v，就可以把这两个1同时消去。所以可以用bfs找出2k个点之间的最短路，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)。 | Accepted |
| 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(空)=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,…,sn)=f(f(s1,s2,…,sn-1),sn)。现给出n个长度为m(<=20)的01串，要求把它分成两个子序列，分别变换后得到串S和T，求|S|+|T|的最小值。 | 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。于是可以记dp(len,suffix)表示以长度为len的suffix结尾的最小增量，然后我们枚举第i个串一个长度为j的前缀s1，再枚举第i-1个串一个长度为k的后缀s2，则dp(j,s1)可以转移到dp(k,s2)。时间复杂度O(n\*m^2)，空间复杂度O(2^m\*m)。 | Accepted |
| 85E | 孙伟峻 | Guard Towers | 给出平面上的n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。 | 图论。可以二分答案d，然后在距离超过d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k个联通块，则有2^k种分法。复杂度O(n^2logn)。 | Accepted |
| 86E | 王子昱 | Long sequence | 考虑k阶(k<=50)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2^k-1，请输出这个k阶递推式的各项系数，以及递推初始的k个值。 | 枚举+验证。可以证明解比较稠密，那么就随机枚举这个k阶递推式，然后判断循环节恰为2^k-1。问题变成如何判断一个线性递推的最小循环节是否为m。可以使用矩阵乘法，判断一下m是否可行；再枚举m 的每一个质因子p，判断一下m/p是否不行。 | Accepted |
| 89D | 王子昱 | Space mines | 给出空间中一个半径为R的球，在(ax,ay,az)位置以速度(vx,vy,vz)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。 | 计算几何。由R>r[i]及线段长<=1.5\*r[i]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。 | Accepted |
| 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\*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，再跑一次矩阵乘法即可。时间复杂度O(9^3\*logR)。 | Accepted |
| 97C | 杜瑜皓 | Winning Strategy | 若干次比赛，每次比赛要派n个人，若有i个人参加过以前的比赛则胜率为p(i)。每个人至多参加两次比赛。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)\*p(i)+(n-2i)\*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\*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\*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<=20），有半径相同的盘子，要把第一个柱子上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。 | 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x个，则把x-1个从a移到b，然后移动剩下的到c，在把x-1个从b移到c；而如果与cur同半径的盘子只有cur自己，那么应该直接Solve2(cur,a,b,c)；若有多个，设第一个与cur不同的是nxt，则先要Solve2(nxt,a,b,c),再把cur移到b，再Solve2(nxt,c,b,a)，再把cur移到c，最后Solve(nxt,a,b,c)。时间复杂度大概是O(2^n)。 | Accepted |
| 98C | 魏鑫鼎 | Help Greg the Dwarf | C:\Users\dell\Desktop\f81e179e45635db94c1427d73eaebd2224f438b0.png  一个L型通道，一头宽是a，一头宽是b，有一个矩形长为l，问w的最大值，使得矩形可以从一头拐到另一头。 | 二分+三分。一些奇葩的特殊情况处理略过。另外w满足二分性，所以可以二分w然后判断。判断过程如下：    其实我们想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。 | Accepted |
| 191D | 孙猛 | Metro Scheme | 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一个简单环，且每条边恰好属于一条路。 | 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。 | Accepted |
| 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+…+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点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(logn)^2)。 | Accepted |
| 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<=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<=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k的最小费用。 | 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中个点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]。注意这是要选择一个闭合图，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)。 | Accepted |
| 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<=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\*logn)。 | Accepted |
| 113D | 刘定峰 | Museum | 一个无向连通图n个点和m条边，一开始两个人分别在a和b，已知每分钟在点i停留的概率是p[i]，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。 | 三种做法：  第一种是记dp(k,u1,v1,u2,v2)表示2^k步，一个人从u1走到u2，另一个人从v1走到v2的概率。当k=18时精度就可以满足要求了。复杂度O(18\*n^6)。  第二种是枚举每个点作为相遇的点，然后反过来算出到达(a,b)的概率。设p(x,y)表示到达(x,y)的概率，设枚举的相遇的点是i，那么初始条件p(i,i)=1，且有p(x,y)=sigma(p(u,v)\*tran((x,y),(u,v)))，这里tran((x,y),(u,v))表示从(x,y)一步走到(u,v)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O(n^7)，但是高斯消元常数写小一点也是可过的。  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Ax=B，在上一种方法中A是n^2\*n^2的，B是n^2\*1的，而每次枚举相遇的点，发生改变的只有B，所以我们可以把B扩充成n^2\*n的矩阵，这样只需解一次Ax=B就可以了，具体做法是求A矩阵的逆元，然后x=A^(-1)\*B。时间复杂度O(n^6)。 | Accepted |
| 115D | 高远 | Unambiguous Arithmetic Expression | 所有非负整数都是UAE。如果X和Y是UAE，那么(X)+(Y)， (X)-(Y)， (X)\*(Y)， (X)/(Y)都是UAE。如果X是UAE，那么-(X)和+(X)都是UAE。给你一个由数字及加减乘除组成的表达式，问有多少个UAE去括号后与之相同。表达式长度n不超过2000。 | 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dp(i)表示有i个左括号没消去的方案数，初始边界dp(0)=1。然后若是碰到了一个加号或者乘号，这后面肯定会接一个左括号，所以要把dp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dp(i)+=sigma{dp(j)}对所有的j>i。所以时间复杂度大概是O(n^2)。 | Accepted |
| 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\*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\*x\*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\*(1.0-x)\*(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，而对一个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O(mlogm\*R)，R是二分的次数。 | Accepted |
| 193E | 彭天翼 | Fibonacci Number | 定义了一个模10^13的斐波那契数列，问某个数x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long long以内。 | 数论。取10^13的两个约数A=2^13，B=5^9，那么有t%A=(t%10^13)%A，t%B=(t%10^13)%B，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(a,b)，表示模A意义下，第a个数是x%A，模B意义下，第b个数是x%B。若模A的循环节是len\_a，模B的循环节是len\_b，那么有a+len\_a\*x=b+len\_b\*y，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。 | Accepted |
| 145D | 彭天翼 | Lucky Pair | 定义只由4和7组成的数是幸运数。给一个长为n的数列（n<=10^5），保证不超过1000个幸运数，问有多少a，b，c，d满足a<=b<c<=d，且不存在一个幸运数既在[a,b]上出现，又在[c,d]上出现。 | 数据结构。设幸运数数目为m，因为m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O(m^2\*logn)。 | Accepted |
| 132E | 许昊然 | Bits of merry old England | 给出一个长度为n的序列（n<=250），以及m个变量（m<=26），需要用一些操作依次输出这n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。 | 网络流。一开始的想法是，把n的值拆点，在i与i’间连一条下界为1上界为1费用为0的边，然后建源和汇，源向每个i连一条下界为0上界为1费用为第i个数代价的边，i’向汇连一条下界为0上界为1费用为0的边。然后对于i<j，若第i个变量的值等于第j个变量的值，则从i’到j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j个数代价的边。这样跑一次流量不超过m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。 | Accepted |
| 138D | 许昊然 | World of Darkraft | 一个n\*m的棋盘（n,m<=20），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。 | 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(x1,y1,x2,y2)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O(n^3\*m^3)。 | 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<=10。 | 排序+枚举。若k>=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>=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<=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。 | 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)。 | Accepted |
| 183D | 成羽丰 | T-shirt | 有n个人m种衬衫，n<=3000，m<=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sigma{dp[n][j][t]|1<=t<=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，每个操作形如[l,r]，先[l,r]上偶数位置的字符取出r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k位，k<=3\*10^6。操作数目不超过5000。 | 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，那么答案就是Solve(n,k)。考虑Solve(i,j)，若r[i]>=j，那么第i次操作对Solve(i,j)没有任何影响，所以Solve(i,j)=Solve(i-1,j)；否则，对操作[l[i],r[i]]，算出由它决定的区间[a,b]（其中a=r[i]+1），然后先算出Solve(i-1,j-(b-a+1))，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k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ext库中的rope，整个时间复杂度做到了O(k\*log(k)+n)，且常数较小。 | Accepted |
| 135E | 李煜东 | Weak Subsequence | 定义一个串a是一个串b的次连续子串，当且仅当a是b的一个不连续的子序列。问由k种字符组成的字符串，且最长次连续子串长度是w，的串有多少。k<=10^6，w<=10^9。 | 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t，使得1<=t<=k，表示串的前t个字符两两不同，而t+1与之前某个相同，那么从t+1到串末尾就是一个次连续子串，所以t+1后面会有w-1个字符。同时要保证不存在超过w的前缀是次连续子串，所以串的后t个字符也要两两不同，这里要分t<=w-1及t>w-1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O(k\*log(mod))，mod是模的数。 | Accepted |
| 163D | 黄嘉泰 | Large Refrigerator | 给出长方体的体积V（按素因子分解给出，不超过10^18），求使得表面积最小的三边长。 | 枚举。设三边长为a，b，c，不妨设a<=b<=c，那么有a<=V^(1/3)，b<=(V/a)^(1/2)，所以可以暴力枚举。注意到a，b均为V的约数，所以可以只枚举V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a，然后在枚举a时，取b=c算一个表面积，这是这个时候理论的最小值；枚举b时也可以从大到小，一旦有一个解后就不用再枚举更小的b了。时间复杂度大概是O(V^(1/3)\*log(V))，那个log是我在枚举a后，找到小于等于(V/a)^(1/2)的最大的约数时用的二分查找。 | Accepted |
| 167E | 黄嘉泰 | Wizards and Bets | 一个n个点m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k个。现在需要找出k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i与源a[i]匹配，那么对所有i<j且a[i]>a[j]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p的值。n<=600。 | 行列式求值。首先需要处理一个方案中，k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p，从a[u]到u经过p，从a[v]到v经过p，即a[u]~p~u，a[v]~p~v，那么也一定存在一种方案，a[u]~p~v，a[v]~p~u，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sgn(p)表示排列p的符号，若有偶数个逆序对则是1，否则是-1，再预处理出d[i][j]表示从源j走到汇i的路径数，那么答案就是sigma{sgn(p)\*prod{d[i][p[i]]}}，sigma表示对所有排列p答案相加，prod表示对所有源p[i]走到汇i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O(n^3)。 | Accepted |
| 232D | 李凌霄 | Fence | 给一个长为n的数列h，n<=10^5，称[l1,r1]和[l2,r2]是匹配的，当且仅当两区间长度相同且不相交，对所有0<=i<=r1-l1,有h[l1+i]+h[l2+i]=h[l1]+h[l2]。有q个询问，每个询问是一个区间[l,r]，问有多少区间与它匹配。 | 后缀数组+树状数组。首先可以把条件变形，得到h[l1+i]-h[l1]=h[l2]-h[l2+i]，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h的差分数列s1，把s1每个数取相反数为s2，构造s=s1+$+s2，$是比其他数字都大的一个分隔符。问题就变成了，对于s1的某个子串，在s2中有多少个与它相同，且不相交。求出s的后缀数组，设询问串为t，长度为l，那么与以t开头的后缀LCP不小于l的都可能是解，并且这些后缀在排序后是连续的，设第i个询问的这个连续的区间是[st[i],ed[i]]，表示排名第st[i]的串到排名ed[i]的串。记pos数组，对于排名第i的后缀，若它在s2中，那么pos[i]等于它对应的原位置，否则pos[i]=0。那么问题转化成了，一个pos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O((n+q)logn)。 | Accepted |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 175E | 李凌霄 | Power Defence | 塔防：一个怪物从x轴负无穷走向x轴正无穷，速度1/s。可以再y=1和y=-1两条直线的整点上建塔，每个位置最多建一个塔。有三种塔：nf个火塔，在一个半径rf圆内造成df/s的伤害；ne个电塔，在一个半径为re圆内造成de/s的伤害；ns个冰塔，在半径rs的圆内造成减速，若怪被k个冰塔减速，那么速度变成1/(k+1)。三种塔数目之和不超过20。求最大可能伤害。 | 枚举+动态规划。可以把塔都建在y=1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n个位置，通过线段交一类的算法算出第i个位置放火塔的伤害f[i]，第i个位置放电塔的伤害e[i]。那么可以有一个dp，dp[i][j][k]表示前i个位置，用j个火塔k个电塔能造成的最大伤害。这个动态规划的复杂度是O(n^3)，对枚举出的方案都DP一次就行了。要控制一下常数。 | Accepted |
| 176D | 吕可凡 | Hyper String | 给n个基本串，定义一个字符串s，由m个基本串一次连接而成，再给出一个长度不超过2000的串t，问s和t的最长公共子序列。所有串只由字母组成，n<=2000，m<=2000，基本串长度之和不超过10^6。 | 动态规划。考虑一个简化的问题：对一个长为10^6的串a，和一个长为10^3的串b，只由小写字母组成，求LCS。这个可以以b串为基础进行DP，记dp[i][j]表示b串前i个字符与a串有一个长为j的LCS时，在a中子序列右端点的最小值。那么dp[i][j]首先可以直接从dp[i-1][j]转移过来，现在考虑如何从dp[i-1][j-1]转移过来，设pos=dp[i-1][j-1]，那么就应该找到a串pos后面第一个等于b[i]的字符，设位置是q，那么就可以用q来更新dp[i][j]，所以需要预处理出a中每个位置之后每个字母第一个出现的位置。这个可以O(26\*10^6)做到，然后DP的复杂度就是O((10^3)^2)。对于原题目，不过是多了一个m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m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n个基本串长度之和为w，那么时间复杂度为O(26\*w+26\*m+|t|^2)。 | Accepted |
| 178F2 | 吕可凡 | Representative Sampling (30 points) | 给n(n<=2000)个长度不超过500的串，要选k个串出来，使得两两串最长公共前缀长度之和最大。 | 动态规划。一般想法是设dp[l][r][t]表示从第l个串到第r个串选择t个串的最大值，然后枚举中间断点k，从dp[l][k]及dp[k+1][r]转移过来，但是有一个问题是，[l,k]中选的串与[k+1,r]中选的串配对后对答案的增量无法计算，而且k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lcp[i]表示排序后i串与i-1串的最长公共前缀长度，那么可以考虑从[l+1,r]上取lcp最小的位置p，然后断开成dp[l][p-1]和dp[p][r]，这时，因为lcp[p]是最小值，串又是排了序的，所以[l,p-1]的串与[p,r]的串的最长公共前缀都是lcp[p]。然后枚举i，j，那么dp[l][r][i+j]可以用dp[l][p-1][i]+dp[p][r][j]+i\*j\*lcp[p]来更新。时间复杂度大概就是O(nlogn\*m+n^2\*logn)，m是串的长度。 | Accepted |
| 178E3 | 何琦 | The Beaver's Problem - 2 (50 points) | 一个n\*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\*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maxd和mind，若maxd/mind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<=b,d<=100。 | 数论。2-type的特点是，总存在一个x，使得d|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 | 李凌劼 | Visit of the Great | 求LCM(k^(2^l)+1,k^(2^(l+1))+1,…,k^(2^r))+1)。1<=k<=10^6,0<=l<=r<=10^18,2<=p<=10^9且p是质数。每个数据有10^5组询问。 | 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)\*…，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[l]-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。 | Accepted |
| 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<=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>=g。分两种情况讨论：t<=g，则g-t<=sum[k]<g+r-t；t>g，则0<=sum[k]<g+r-t或2\*g+r-t<=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)logn)。 | Accepted |
| 176E | 陈睿 | Archaeology | 给出一个n的点的树(n<=10^5)，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10^5。 | 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S，按Dfs序排序后得到序列T。考虑答案由哪些部分组成。设序列T长为m，那么对2<=i<=m，T[i]到T[i-1]与T[i]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T[1]，而少算的部分正好是T[1]到T[1]与T[m]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S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set。时间复杂度O(nlogn+qlogn)。 | Accepted |
| 196D | 陈睿 | The Next Good String | 给一个仅由小写字母组成的字符串S，和一个正整数m，要求一个长度与S相同且仅由小写字母组成的字符串T，要求T的字典序大于S，且T中不包含长度大于等于m的回文子串,若有多个T求出字典序最小的。S长度<=4\*10^5。 | 暴力+哈希。先把问题转化成求的T的字典序不小于S，然后从第一位开始逐位枚举。判断不存在长度不小于m的回文子串，只用判断当前位置往前m长度以及m+1长度是不是回文子串。设最早在第i位有T[i]>S[i]，那么之后的字符可以随便填了，之后可以接aa…abb…bcc…caa…a诸如此类的，即一旦在某个位置T[i]>S[i]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O(1)。所以整个时间复杂度是O(n)，n是S串的长度。 | Accepted |
| 198E | 钱迪晨 | Gripping Story | 考虑在一个二维平面中，在(x,y)处有一个吸引力为p吸引半径为r的磁铁，其他地方还散落着n个磁铁(n<=250000)，第i个磁铁在(x[i],y[i])位置，质量为m[i]吸引力为p[i]吸引半径为r[i]。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。 | 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，于是我们需要维护一个东西，找出第一关键字不小于R，第二关键字不小于P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set，时间复杂度是(n(logn)^2)。 | Accepted |
| 200E | 钱迪晨 | Tractor College | 一共有n(n<=300)个人，分三类，分别有c3、c4、c5个，要发完m块钱(m<=3\*10^5)，这三类每类每个人发的钱应该一样，且是w3、w4、w5，要求0<=w3<=w4<=w5。在m块钱用完的情况下，最小化abs(w3\*c3-w4\*c4)+abs(w4\*c4-w5\*c5)，请确定w3、w4和w5。 | 数论+三分。考虑在式子中w4\*c4出现了两次，不妨以它为突破点来搞。枚举w4的值，设Q=w4\*c4，P=m-Q，那么有c3\*w3+c5\*w5=P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0<=w3<=w4及w5>=w4的w3的范围。注意abs(w3\*c3-w4\*c4)+abs(w4\*c4-w5\*c5)=  abs(Q-w3\*c3)+abs(Q-w5\*c5)=  abs(Q-w3\*c3)+abs(Q-(P-w3\*c3))=  abs(w3\*c3-Q)+abs(w3\*c3-(P-Q))，  令a=Q，b=P-Q，则上式=abs(w3\*c3-a)+abs(w3\*c3-b)，这个式子关于w3是单峰的，所以可以三分找到最优解。时间复杂度O(m\*log(m))。 | Accepted |
| 200A | 李宇轩 | Cinema | 一个n\*m的矩阵，有k个询问，每个询问是一个坐标(x,y)，要找到一个坐标(a,b)，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a最小的，还有多个满足选b最小的。1<=n,m<=2000，k<=min(n\*m,10^5)。 | 暴力+并查集。注意到以(x,y)为中心，曼哈顿距离小于等于d的坐标们在一般情况下是O(d^2)的，即每个询问的答案(a,b)到(x,y)的曼哈顿距离是O(sqrt(k))的。我们从小到大枚举x与a的差值t，分别找出(x-t,y)和(x+t,y)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O(sqrt(k))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y的被选了，那么就在并查集中把y连向y-1，这样每次询问y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O(k\*sqrt(k))，但是有一个问题，在矩阵较“窄”的时候会退化，所以当n>m时交换n和m再处理即可。 | Accepted |
| 201E | 李宇轩 | Thoroughly Bureaucratic Organization | 有一个长为n的排列，每次可以做多询问m个位置的数的集合是什么，求最少询问次数确定这个排列。m<=n<=10^9。 | 二分+组合计数。显然询问次数是可以二分的，设询问次数为k，那么有至多k\*m个询问的位置。问题在于如何区分两个位置的数，即，在这k次询问中，位置i出现的询问集合不能与位置j出现的询问集合相同。而每个位置i最多寻问k次。我们枚举x从0到k，那么询问x次的位置不能超过C(k,x)个，C是组合数，而这C(k,x)的位置占据的询问位置是C(k,x)\*x个，我们可以统计出满足n个位置的询问，总共需要的询问位置数，并把它和k\*m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x的复杂度是log(n)的，这样对于每组询问的复杂度做到了O((logn)^2)。 | Accepted |
| 201D | 孟凡航 | Brand New Problem | 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<=15，m<=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<<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<=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(mlogm)，m为所有字符串的长度和。 | Accepted |
| 207B1 | 龚拓宇 | Military Trainings (20 points) | n个人，每个人有一个接收半径a[i]，若i位置的人向j位置的人传消息，那么设j位置上人的接收半径为r，那么有i<j，i>=j-r。一开始所有人按1至n站好，1要把一个消息传给n；然后n站到队首，由n把信息传到n-1，这样迭代n-1次。问最少信息需要传递的总次数。n<=250000。 | 倍增。把信息传递的方向取反，那么每次都是发射信息，i位置发射向j位置，有i>j且i-r<=j，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i记录一个它往前能发射到的最小位置p[i]。那么有一个贪心的结论，当在第i个人，且最小能发射到p[i]，那么应该选择最小的j使得p[i]<=j<i，且p[j]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i往前发射时应该到的位置q[i]。对于一次信息传送就可以O(n)了。因为有n次信息传递，所以还有优化。把序列扩充成1,2,…,n,1,2,…,n，那么信息传递就是对1<=i<=n，从n+i传递到i+1，这时对这2\*n个位置都算出q[i]，预处理一个倍增数组dp[k][i]表示从i位置走2^k步到的位置。这样对于每次信息传递可以做到O(logn)，于是整个时间复杂度就是O(nlogn)。 | Accepted |
| 207A2 | 龚拓宇 | Beaver's Calculator (30 points) | n个人，第i个人有k[i]个需求，a[i][j]表示第i个人第j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a和b，若a>b,则是一个坏对。 | 贪心。可以算出对第i个人，他的需求中坏对的数量cnt[i]，我们能够构造一个解，使得坏对数就是cnt[i]中的最大值。对每个人i，把它的需求按坏对划成cnt[i]+1段，从前往后从1到cnt[i]+1标号，设cnt[i]中的最大值是t，那么标号至多到t+1，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t个坏对；而取一个x使得cnt[x]=t,对于相邻的两块i和j，x在i中的最大值是y，在j中的最小值是z，由坏对的定义知y>z，所以i和j之间必有一个坏对，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(mlogm)，m为所有人需求之和。 | Accepted |
| 167D | 张闻涛 | Wizards and Roads | 给出平面上n个点，保证点之间两两x值不同，两两y值不同，且大致是随机分布的。点u和点v能连边（y[u]>y[v]），当且仅当对任何在u、v形成的拐角中的点w，都有一个点s不在拐角中，且y[s]>y[v]，x[s]在x[w]与x[u]之间。“拐角”的定义如图：  C:\Users\dell\Desktop\b66cf8cda27865d75c7e7684895957b52651ed5f.png  有m个询问，每次询问为[L,R]，只考虑横坐标在[L,R]上的点，每个点最多连一条边，最多能连多少条边。n,m,<=10^5。 | 数据结构+匹配。考虑第l个点到第r个点，哪些能连边。设中间y最大的点是第k个点，那么对于其左边的某个点u，和右边的某个点v，k必在u和v形成的拐角中，而显然是不可能找出点s满足x[s]在x[u]和x[k]之间，且s不在拐角中。所以k左边的点与右边的点不可能连边。于是点集被剖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l号点到r号点构成的二叉树中求最大匹配，而树上的匹配是可以用dp完成的。所以对每个点记录dp[i][0]表示不选i号点在i子树的最大匹配数，及dp[i][1]表示选i号点在i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O(logn)，所以时间复杂度就是O(nlogn+mlogn)，其中O(nlogn)是构造这棵二叉树的复杂度。 | Accepted |
| 209C | 张闻涛 | Trails and Glades | 给一个n个点m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)。n,m<=10^6。 | 贪心。用并查集处理出块的数目t，设第i个块有a[i]个奇度数点，b[i]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a[i]为第一关键字排序，然后在相邻两块间连边，一共连t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k个奇度数点，那么我们需要把它们变成偶度数，就需添加要k/2（往上取整）条边。时间复杂度O(m+nlogn)。 | Accepted |
| 212B | 孙伟 | Polycarpus is Looking for Good Substrings | 给一个长度为n的仅由小写字母组成的字符串s，有m个询问，每个询问时一个字符的集合t，问s中有多少子串[a,b]满足，把[a,b]上的字母取出后去重后与t相同，且没有x<=a<=b<=y，使得[x,y]比[a,b]长并同时满足条件。n<=10^6,m<=10^4。 | 状态压缩。先把所有询问离散化，方便记录答案。预处理一个next[i][j]表示i位置往后，字母j出现的最早位置。然后枚举左端点l，那么至多只有26个右端点是有用的，不妨利用next数组和当前字符集合mask求出下一个r。因为是从小到大枚举l，所以对于相同的mask，目前的r应该比之前的r大，所以要对每个mask记录一个val[mask]表示mask存在的区间的最右端点。时间复杂度O(26\*n+mlogm)。 | Accepted |
| 212D | 孙伟 | Cutting a Fence | n个木板，第i个木板高h[i]，若在[x,x+k-1]上画一个宽为k的矩形，那么高度就是x到x+k-1最小的h。给m个询问，每个询问是w，问宽是w的矩形的平均高度。n,m<=10^6。 | 计数+数据结构。先把问题转化成宽为w的所有矩形的高度和。考虑每个h[i]对答案的贡献，这样考虑：h[i]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i求出[lb,rb]，表示h[lb-1]是左边第一个不大于h[i]的，h[rb+1]是右边第一个小于h[i]的，设x=i-lb+1,y=rb-i+1，那么就有它对x\*y个区间有贡献。不妨设x<=y，对每对p，q，满足0<=p<x，0<=q<y，h[i]就会对宽为p+q+1的矩形高度和贡献一次。设宽度减一为t，那么对0<=t<x，(p,q)对数是t+1；对x<=t<=y，(p,q)对数是x+1，对y<t<=x+y，(p,q)对数是x+y-t+1，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O(nlogn)。 | Accepted |
| 212C | 周誉昇 | Cowboys | 一个长度为n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。3<=n<=100。 | 动态规划。记dp[i][a][b]表示到第i个人，前一时刻第i个人状态是a，第i-1个人的状态是b，能够变成现在状态的状态数。那么可以通过枚举第i+1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n个人的状态，然后再DP。时间复杂度O(n)。 | Accepted |
| 213E | 周誉昇 | Two Permutations | 给一个长度为n的排列a，再给一个长度为m的排列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。 | 哈希+数据结构。显然0<=d<=m-n，可以从小到大枚举d，然后维护序列a的哈希值。哈希选用乘幂取模，比如原来的哈希是a\*p^2+b\*p^1+c\*p^0，那么d加1后增加了p^2+p^1+p^0，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O((m-n)logn)。 | Accepted |
| 217C | 许瀞云 | Formurosa | 定义一种表达式，s->0|1|?|(s|s)|(s&s)|(s^s)，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。2<=n<=10^6。 | 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dp[l][r][a][b]表示[l,r]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[l,r]上不在任何括号中的运算符的位置p，从[l,p-1]和[p+1,r]转移过来。虽然[l,r]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O(mlogm)预处理后O(1)得到，m是表达式长度。这样可以先广搜找出所有有用的[l,r]，然后倒过来DP，这样可以避免递归。时间复杂度O(mlogm)。 | Accepted |
| 229E | 许瀞云 | Gifts | 要选n个数，一共有m种数，第i种数有k[i]个，第i种数的第j个价值是c[i][j]。而若在第i种数中选了u个，那么得到的是这种数中随机的u个。你选择了价值最高的n个数，而且若有多种选法就等概率的选择一种，问拿到n样价值最高的数的概率。n,m<=1000，所有k[i]的和不超过1000。 | 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cnt[i]表示第i种数中价值超过x的个数。记dp[i][j]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dp[0][0]=1.0，而dp[m][cho]/C(tot,cho)就是答案，这里的C表示组合数。再来考虑转移：有dp[i][j+o]=sigma{dp[i-1][j]\*1.0/C(k[i],cnt[i]+o)}。这个动态规划的复杂度是O(cho\*tot)的，所以整个时间复杂度大概是O(tot^2)的。 | Accepted |