目录

Codeforces

Graph Game

CF 235 D

【题目大意】

在计算机科学领域里，有一种解决复杂树链问题的算法叫做”点分治”。让我们来定义一下这个算法的流程：

Solve(T) (T 是一棵树)

1.选择一个树 T 上的结点 x(通常选用树的重心)，我们将这个操作叫做”第一步”。

2.处理所有经过点 x 的路径

3.从树 T 中删除结点 x

4.然后树 T 分裂成若干子树

5.分治处理所有的 Solve(S) (S 是分裂出来的子树)

这个算法会结束，因为当 T 只有一个结点时，删除这个点是就什么都不剩了。

现在 WJMZBMR 错误的认为在”第一步”中选择任何一个点都是可以的，所以他在第一步中随机选点，而不是选树的重心。为了使情况更糟，他认为一棵”树”应该满足边数和点数相等。所以他做点分治的过程变成这样：

让我们定义变量 totalCost，初始 totalCost=0。然后，solve(T) (现在 T 是一个图)

1.totalCost=totalCost+(size of T).运算符’=’表示赋值。(Size of T)表示图 T 中的结点个数。

2.在图 T 中随机选择一个结点 x(图 T 中每个点被选中的概率相等)

3.从图 T 中删除结点 x

4.然后 T 变成了一些联通快

6.分治处理所有的 Solve(S) (S 是剩下的连通块)

他将新的 solve 函数用于一个 n 个结点 n 条边的连通图，并且认为这种算法会运行的很快，但是结果恰恰相反，它运行的非常慢。所以他想知道对于某个连通图这个算法中变量 totalCost 的数学期望。你能帮帮他吗？

数据范围：$n \leq 3000$

时间限制：1s

【问题分析与程序设计】

计算点 x 对于点 y 的贡献就是 x 到 y 的路径上没有任何一个点被选到的概率。环套树只要暴力拆环再做一次 DP 就可以了。

时间复杂度 $O(n^2 * \log_2 n)$

空间复杂度 $O(n^2)$

Graph Game

CF 235 E

【题目大意】

定义 d(n)为 n 的约数个数。现在，你有三个数 a,b,c。你的任务是计算下面式子 modulo 1073741824 (2^30)的值。$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c} d(i*j*k)$

数据范围：$1 \le a, b, c \le 2000$

时间限制：1s

【问题分析与程序设计】

$$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c} d(i*j*k) = \sum_{\gcd(i,j)=\gcd(j,k)=\gcd(i,k)=1} \left[\frac{a}{i}\right]\left[\frac{b}{j}\right]\left[\frac{c}{k}\right]$$

暴力枚举两维，第三维直接莫比乌斯反演

时间复杂度 $O(n^2 * \log_2 n)$

空间复杂度 $O(n^2)$

Tape Programming

CF 238 D

【题目大意】

对一个由数字和"<",">"构成的非空串进行模拟。最开始有一个指针指向最左字符，移动方向为向右。

重复以下操作直到指针指向串外：

1、如果指针指的位置是一个数字，输出这个数字，然后将指针沿着原来移动方向移动，同时将原来的数字减一。如果原来的数字为 0 则删除这个数字，串的长度减一；

2、如果指针指的位置是"<"或">"，那么指针的移动方向改为与符号的尖角方向相同，接着指针沿着新的移动方向移动。如果新的位置也是"<"或">"，则删除原来的"<"或">"字符。

任何时刻如果指针指向了串外，停止模拟。

给出一个字符串 S，q 个询问，每次询问 S 的一个子串，问输出的每种数字的个数。

数据范围：$|S| \le 10^5$，$Q \le 10^5$，$1 \le l_i \le r_i \le n$

时间限制：1s

【问题分析与程序设计】

算法 1：

首先对于每个询问可以对子串直接模拟，用链表维护每个位置在当前时刻与之相邻的两个位置。

时间复杂度 O(10 * Q * |S|)

空间复杂度 O(10 * |S|)

期望得分：30

这个方法的效率是比较低下的，对整个问题做更深入的研究可以发现，其实每次询问的很多模拟都是重复的。

算法 2：

其实对于每次询问的子串的模拟过程都是对于整个串模拟过程的连续的一段，所以只要先对整个串做一遍模拟。

对于一次询问，先找到第一个出现在询问区间里的位置，再利用二分找到第一个超出询问区间的位置。利用对答案[0..9]做的前缀和就得中间这段过程的输出值即可。

时间复杂度 O(10*|S|+$\log_2$|S| * Q)

空间复杂度 O(10*|S|)

期望得分：100

【实现细节】

在对与整个串做模拟的时候不能只从第一个位置开始做，从第一个位置模拟完了之后要判断后面是否存在没有被访问过的位置，如果有，那就继续从第一个没有被访问过的位置做模拟，终止条件要改成超过当前区间就跳出。

Numbers

CF 241 D

【题目大意】

有一个 1,2..n 的排列 $a_1,a_2,\ldots\ldots,a_n$，要删除一些整数,使得结果序列满足以下三个条件:

1.由此产生的序列不是空的

2.序列中所有数异或和等于 0

3.把所有数按十进制从前往后依次无间隔地写在一行形成一个大的十进制数，这个数将会被 p 整除

给出序列和 p，找到一种方法来满足上述条件。

数据范围：$1 \le n$，$p \le 50000$

时间限制：1s

【问题分析与程序设计】

算法 1：

设计 DP 状态$f[i][j][k]$表示做到第 i 个数为止，异或和等于 j，模 p 的余数为 k 的方案是否存在。

$$f[i-1][j][k]=1 \rightarrow \begin{cases} f[i][j][k]=1 \\ f[i][j\ xor\ a[i]][(k*10+\ a[i])\ mod\ p]=1, a[i]<10 \\ f[i][j\ xor\ a[i]][(k*100+\ a[i])\ mod\ p]=1, a[i]\ge 10 \end{cases}$$

时间复杂度 O($n^2 * p$)

空间复杂度 O($n^2 * p$)

期望得分：50

算法 2：

其实我们只需要取前 31 个数做即可。因为前 31 个数中能使异或和为零的方案数有很多，我们可以认为这些方案所构成的大数模 p 的余数是随机分布的。

时间复杂度 O(32 * 32 * p)

空间复杂度 O(32 * 32 * p)

期望得分：100

【实现细节】

这道题空间限制为 256M，因为最后要输出方案，所以如果不是写的记忆化搜索要注意空间问题，可以把上一个状态的三个信息压成一个整形。

Flights

CF 241 E

【题目大意】

给出一个 n 个点 m 条边的没有重边的有向无环图，需要给每条边一个权值，只能为 1 或 2，使得从点 1 开始所有能到达点 n 的路径的长度相等。求方案，如果不存在则输出 No

数据范围：$1 \le n \le 1000$，$1 \le m \le 5000$

时间限制：1s

【问题分析与程序设计】

算法：

本题要求从点 1 开始所有能到达点 n 的路径的长度都要相等，这个条件看起很弱但是实际上非常强。

首先对于这张图中有一些边是没有意义的，对于一条边(u, v)来说，如果 1 无法到达 u 或者是 v 无法到达 n，那么这条边一定不会出现在从 1 到 n 的路径中，可以随便安排边权。

排除掉没有意义的边之后，再来考虑对于一个节点 x 来说，如果 1 到 x 的路径的长度存在不相等的，那么再加上从 x 到 n 的某一条路径后，这个长度还是不相等，所以题目所给出的条件是要求从点 1 开始到所有点的路径的长度都要相等。

然后就可以 DP 了，根据拓扑序来做，f[i][j]表示从点 1 到点 i 所有路径长度等于 j 是否可行，转移的时候做 and 就可以了。

时间复杂度 $O(m^2)$

空间复杂度 $O(nm)$

期望得分：100

【实现细节】

最后要输出每条边的长度，所以最后要构造出一组合法解。

其实这道题可以不用这么写，可以直接设 dis[i]表示从 1 到 i 的路径长度，然后进行松弛，一开始除了 1 其他点都是 Inf，一条边(u, v)代表了两个限制条件：

1. $dis[v] \le dis[u] + 2$
2. $dis[u] \le dis[v] - 1$

做完 n 边松弛之后还有不满足的就是无解了。

Colorado Potato Beetle

CF 243 C

【题目大意】

有一个大小为$(10^{10}+1)\times(10^{10}+1)$方格图，初始有一个人在最中间，他将进行 n 次移动，给出每次移动的方向（上下左右）和距离 x，问最后的移动的轨迹所围成的图形的面积大小。

数据范围：$1\le n\le 1000$，$1\le x\le 10^6$

时间限制：1s

【问题分析与程序设计】

因为这个人最多移动 n 步，所以只有 n 个坐标是有用的，所以把所有有用的坐标进行离散化，最后的图的大小是 n^2 级别的，然后做一遍 floodfill 就可以了

时间复杂度 $O(n^2)$

空间复杂度 $O(n^2)$

【实现细节】

有用的坐标包括实际的坐标以及上下左右四个位置

Matrix

CF 243 E

【题目大意】

考虑一个大小为 n×n 的，仅包含 0、1 的矩阵，当此矩阵满足以下条件时，此矩阵称为好的：在每一行中，所有的 1 都靠在一起。即，每一行都形如 00...0011...1100...00（有可能是全部为 0 或全部为 1）

给你一个 n×n 的仅包含 0、1 的矩阵 a，你的任务是判断是否能够通过重新排列某些列使得这个矩阵变成好的矩阵 b。

数据范围：$1 \le n \le 500$

时间限制：1s

【问题分析与程序设计】

PQ-tree 的定义即为本题

Piglet's Birthday

CF 248 E

【题目大意】

Piglet 要过生日了，Winnie 想送给他一些蜜罐。

Winnie 的家里有 n 个架子，每个架子上都有一些（或没有）蜜罐。初始时所有蜜罐都是装满蜜的。Winnie 一共去了 q 次架子；第 i 次 Winnie 会先去第 u_i 个架子，拿走 k_i 个蜜罐，把这些蜜罐中的蜜吃掉，然后把这些蜜罐都放到第 v_i 个架子上。当 Winnie 拿走蜜罐时，他会在第 u_i 个架子上所有 k_i 个蜜罐的集合中等概率选择一个集合，然后把集合中的 k_i 个蜜罐拿走。（保证每次操作时 k_i 都不会超过当前第 u_i 个架子上的蜜罐的个数。）

Winnie 想知道，每次操作后，架子上所有蜜罐都被吃完的架子的期望个数是多少。

数据范围：$1 \le n \le 10^5$，$1 \le q \le 10^5$，$1 \le a_i \le 100$，$1 \le k_i \le 5$

时间限制：1s

【问题分析与程序设计】

因为和的期望等于期望的和，所以架子上所有蜜罐都被吃完的架子的期望个数等于每个架子上被吃完的蜜罐的期望个数的和。

因此，每个询问都是独立的，除了选中的两个架子，其他架子的贡献都不会变。

设计 DP 状态，f[i][j]表示第 i 个架子上还有 j 个没有吃完的蜜罐的概率，a[i]表示第 i 个架子上总共还有 a[i]个蜜罐。

转移方程 $f[i][j] = \frac{f[i][j] * (a[i]-j) + f[i][j+1] * (j+1)}{a[i]}$

时间复杂度 O(q * a_i * k_i)

空间复杂度 O(q * a_i)

Two Sets

CF 251 D

【题目大意】

Petya 非常喜欢数字。最近他的妈妈送给他了一个由 n 个非负整数组成的集合。Petya 非常喜欢和 Masha 玩。他立即决定把他 n 个数字中的一部分送给 Masha。为了让游戏更加有趣，Petya 决定使得给她的数字集合满足如下条件：

我们用 x_1 表示 Petya 的数字集合的 xor 值，用 x_2 表示 Masha 的数字集合的 xor 值。要使得 x_1+x_2 尽可能地大。假如有多种划分集合的方法使得集合满足上述条件，Petya 就要让 x_1 尽可能地小。

Xor 运算是“异或”，在 Pascal 语言中用”xor”表示，在 C/C++/Java 语言中用”^”表示。

帮助 Petya 按照上述方法划分集合。如果有多种合适的方法，就找出其中任意一种方法。请注意，Petya 将一部分数字给了 Masha 之后，他可能就没有任何剩余的数字了。反之亦然，Petya 也可以不给 Masha 任何数字。在这两种情况下，我们都假定空集的 xor 值为 0。

数据范围：$1 \le n \le 10^5$

时间限制：1s

【问题分析与程序设计】

考虑所有的值 xor 起来是 ans 的话，如果 ans 的第 i 位二进制是 1，那么无论如何对答案都没有影响，如果是 0，那么 x_1 和 x_2 的第 i 为都是 1 会使得最终结果最大化，然后做一次 DP 即可。

时间复杂度 $O(n * \log_2 n)$

空间复杂度 $O(n)$

Dividing Kingdom

CF 260 E

【题目大意】

给出平面上 n 个点，要求用两条平行于 x 的直线以及两条平行于 y 轴的直线将这个平面分成九块，每一块中的点数恰好能对应 $a_1..a_9$，直线不能经过任意一个点，求一组合法解。

数据范围：$1 \le n \le 10^5$

时间限制：1s

【问题分析与程序设计】

算法：

因为规定了只要分成 9 个平面，所以应该先用 9！枚举具体的对应情况，然后发现需要一个高效的求每一块的点的个数的方法。

对于这个问题可以使用主席树来解决，把 n 个点按横坐标从小到大排序，然后依次插入主席树即可，坐标要先离散化。

时间复杂度 $O((n+9!)*\log_2 n)$

空间复杂度 $O(nm)$

期望得分：100

【实现细节】

对于 9！的枚举之后可以先得到答案，因为平行于 x 轴的第一条直线以下一共有多少个点已经知道了，所以只要排序之后直接定位即可。

如果直接把 9 个平面内的点全部求出来之后再判合法性可能会 TLE，所以可以求一个区域就判一个区域，这样可以快一倍。

Maxim and Increasing Subsequence

CF 261 D

【题目大意】

给出一个长度为 n 的序列 $b_1..b_n$，将 b 循环 t 次得到序列 a，长度为 n*t，问序列 a 的最长上升序列的长度，有 k 组测试数据。

数据范围：$1 \le k \le 10$，$1 \le n, \max\{b_i\} \le 50000$，$1 \le t \le 10^9$，$n * \max\{b_i\} \le 2 * 10^7$

时间限制：3s

【问题分析与程序设计】

算法：

设计 DP 状态 f[i]表示最后一个数小于等于 i 的最长的上升序列的长度。

做 n * t 次更新，每次用 f[a[i] - 1] + 1 来更新 f[a[i]..max{b_i}]

虽然 t 很大但其实没必要做那么多次，只要做 min{max{b_i}, t}次即可。

时间复杂度 O(n * max{b_i} * k)

空间复杂度 O(max{b_i})

期望得分：100

【实现细节】

因为 f 是单调的，所以更新的时候暴力更新就可以了，一旦没法更新就直接退出。

Maxim and Calculator

CF 261 E

【题目大意】

Maxim 得到了一个计算器，这个计算器有两个整数单元，一开始，第一个单元包含数字 1，第二个单元包含数字 0。 这个计算器支持一以下两种操作：

(假设第一个单元的数字为 a，第二个单元的数字为 b)

操作一：将第二个单元的数字改成 b+1。

操作二：将第一个单元的数字改成 a*b。

现在 Maxim 想知道，有多少个正整数 x($l \le x \le r$)满足， 存在一种方式从计算器初始状态开始，操作不超过 p 步之后使得第一个单元中的数字为 x。

数据范围：$2 \le l \le r \le 10^9$，$1 \le p \le 100$

时间限制：4s

【问题分析与程序设计】

算法：

在做这道题的时候很容易想到直接设计状态 f[i]表示将第一个单元中的数字变为 i 所需要的最少的步数，每次枚举第二个单元格中的数字来更新，因为第二个单元格中的数字一定是从 1 开始慢慢累加到一个数的，所以 f[i]中记的是操作二的次数，然后判断 f[i]加上每次枚举的第二个单元格中的数字是否小于等于 p 即可。但是这么做的复杂度会达到 10^{11}，显然会超时。

在做上述 DP 的时候可以发现当一个数含有大于 100 的质因子的时候必然不能在 100 步之内得到。因为步数 p 小于等于 100，所以第二个单元格中的数字 b 小于等于 100，所以最终 a 不能含有超过 100 的质因子。只要把 1..r 以内所有含有大于 100 的质因子的数全部删掉即可，但是删除的最好的复杂度是线性的而且要删的数的个数远远大于合法的数的个数，所以应该想办法生成合法的数。

生成合法的数只需要 dfs 就可以了，每次乘上一个不超过 100 的质数。而只包含 100 以内的质因子的数在 10^9 之内不超过 $3*10^6$ 个，然后直接按照原来的 DP 做就可以了。

时间复杂度 $O(3 * 10^6 * p)$

空间复杂度 $O(3 * 10^6)$

期望得分：100

【实现细节】

在生成合法数字时要注意不要生成重复的数字，然后在做 DP 之前要先将这些数字排序。

Rhombus

CF 263 E

【题目大意】

你现在得到了一个大小为 n×m 的表。在第 i 行 ($1 \le i \le n$) 与第 j 列 ($1 \le j \le m$)的交叉处，有一个非负整数 $a_{i,j}$。此外，你还得到了一个非负整数 k。

你的任务是找到一对满足下列条件的整数对(a,b)

1.$k \le x \le n - k + 1$

2.$k \le y \le m - k + 1$

3.$k \le x \le n - k + 1$ 且 $k \le y \le m - k + 1$, $f(x,y) = \sum_{i=1}^{n}\sum_{j=1}^{m} a_{i,j} * max(0, k - |i - x| - |j - y|)$的范围内定义函数的最大值是 mval。对于我们要找的整数对必须满足下列等式 f(a,b)=mval。

数据范围：$1 \le n, m \le 1000$，$1 \le a_{i,j} \le 10$

时间限制：1s

【问题分析与程序设计】

每次其实是维护一个转了 45 度的正方形，通过维护两条对角线的前缀和就可以维护这个正方形的了，需要维护的东西比较多，具体实现可以看程序。

时间复杂度 O(n * m)

空间复杂度 O(n * m)

BerDonalds

CF 266 D

【题目大意】

有一个城市的交通网络由 n 个节点和 m 条双向道路组成，每个节点有一个咖啡厅。现在要选择某个位置(可以是节点，也可以是某条道路上的任意一点)建一个餐馆，使得餐馆到最远的咖啡厅的距离最近，输出这个距离。

数据范围：$1 \le n \le 200$，没有重边或自环

时间限制：1s

【问题分析与程序设计】

算法：

预处理先做一遍 floyed，求两两之间的最短路，存在数组 d 中。

考虑一条长度为 len 边(u, v)来说，在上面建一个餐厅，餐厅跟 u 相距 x(0 ≤ x ≤ len)，那么点 i 跟餐厅的距离就是 min(d[i][u] + x, d[i][v] + len - x)，这是一个单峰函数，当 x = (d[i][v] - d[i][u]+ len) / 2 时有最大值。

而最远的咖啡厅的距离就是$\max\limits_{1 \le i \le n} min(d[i][u] + x, d[i][v] + len - x)$，同样也是一个单峰函数可以直接三分在每条边上的位置。

时间复杂度 $O(n^3 * \log_2 n)$

空间复杂度 $O(n^2)$

期望得分：100

【实现细节】

在用单调性的时候要考虑清楚，细节上需要特别注意。

More Queries to Array...

CF 266 E

【题目大意】

给出 n 个数 $A_1..A_n$，Q 个操作，操作有两种：

操作一：’0 i val’，把 A_i 赋值为 val

操作二：’1 L R k’，询问 L..R 这段区间，取不超过 k 段的最大不相交子段和

数据范围：$1 \le n$，$Q \le 10^5$，$1 \le k \le 20$，$|A_i| \le 500$

时间限制：4s

【问题分析与程序设计】

算法 1：

建线段树，每个点上记 80 个值，即取 k 段时这段区间中的最大值，左起最大值，右起最大值，左右都必取的最大值，最后做类似于最大子段和的合并即可。

时间复杂度 $O(20 * 20 * \log_2 n * Q)$

空间复杂度 $O(160 * n)$

期望得分：60

理论上这个算法是可以过的，但是由于线段树的常数太大，会卡着超时。

算法 2：

因为是要取不超过 k 段的最大不相交子段和，可以建立网络流模型，然后通过线段树来模拟网络流的过程。因为在做网络流时会取反向边，所以线段树要支持区间取反操作，因此要记对应的最小值，同样类似于之前的做法，只不过一个点上只要记 16 个值即可。

时间复杂度 $O(20 * \log_2 n * Q)$

空间复杂度 $O(32 * n)$

期望得分：100

【实现细节】

做操作二时，循环 k 次，每次取这段区间的最大子段和，取完后把取过的那一段数取反继续，如果哪一次最大子段和已经小于零就直接跳出循环。最后要把所有取反的位置还原。

Dima and Figure

CF 273 D

【题目大意】

Dima 喜欢在一块长方形纸片上作自己喜欢的画。

一块大小为 N*M 的长方形纸片，包含着 N 行与 M 列，并且初始所有格子都是白色的。

Dima 每一次作画，会把纸片上一些格子涂黑，并定义这幅“画”就是所涂黑的格子。

Dima 喜欢一幅画仅当：

*包含至少一个涂黑的格子。

*所有的涂黑的格子形成一个连通块，换句话说，你可以从任意一个涂黑的格子移动到另一个任意涂黑的格子（一个格子可以移动到四联通的格子里）。

*从一个涂黑的格子(x1,y1)到另一个涂黑的格子(x2,y2)所需的最少移动步数等于|x1-x2|+|y1-y2|。

请帮助困惑的 Dima，在 N*M 的长方形纸片上，他能画出多少幅自己喜欢的画呢？你只须回答答案对 10^9+7 取模就行了。

数据范围：$1 \le N, M \le 10^5$

时间限制：1s

【问题分析与程序设计】

算法：

考虑已经解决了 N * (M - 1)的方案数，在最后添上一列时的影响因素，首先肯定是倒数第二列的状态，因为根据题目给出的第二个条件，最后一列与倒数第二列必须是有公共边的，再根据第三个条件可以发现，每一列必然都是从 x 个到第 y 个连续的，所以可以用一个二维状态来表示倒数第二列的情况。但这是远远不够的，以为通过可以画图可以发现一些位置是不可以放的，例如第 i 行在第 M – 2 列出现了一个黑格，而而在第 M – 1 列这一行是一个白格，第 i 行第 M 列就不可以放黑格。简单来讲就是如果前 M – 1 列最上面和最下面的黑格分别在第 x_0, y_0 行，而第 M – 1 列的黑格是从第 x_1 列到 y_1 列的，那么第 M 列放黑格的时候要满足

1. 与$[x_1, y_1]$有交
2. 与$[x_0, x_1 - 1]$，$[y_1 + 1, y_0]$没有交

所以这是一个五维的状态，但是可以发现其实记最上面和最下面的黑格的具体的位置是没有意义的，真正有用价值其实只是 x_0 是等于还是小于 x_1。如果是等于，那就最后一列的开始位置就没有限制，反之就只能从第 x_1 行开始，向下也是一样的。所以状态可以精简为 f[i][j][k][0..3]，i 表示当前做到第 i 列，j, k 表示第 i 列的决策是从第 j 行一直涂到第 k 行，而 0 表示这一列涂完之后下一列上下决策都没有限制；1 表示下一列只有开始位置有限制；2 表示下一列只有结束位置有限制；最后 3 表示表示下一列的上下决策都有限制。

因为这个状态的转移比较直观但是写起来很复杂，所以这里就不再给出转移方程。

时间复杂度 $O(N^2 * M)$

空间复杂度 $O(N^2)$

期望得分：100

【实现细节】

转移的时候要对上一次的答案做一个二维部分和来加速转移。

The Last Hole

CF 274 C

【题目大意】

Luyi 在平面上放了 n 个圆，第 i 个圆的圆心在(x_i, y_i)。最开始所有圆半径都为 0，然后所有圆同时开始变大，在时刻 t $(t>0)$所有圆的半径都为 t。我们可以想象成一些黑色的实心圆放在一个无穷大的白色平面上，每个时刻都会存在一些黑色和白色的联通快。需要注意一点，随着圆的增大，越来越多的圆会相交。

我们定义一个白色的封闭区域为一个'洞 '，例如图中包含两个红色边框的'洞'，随着圆的增大，一些新的洞会出现，也会有一些旧的洞小时。Luyi 想知道最后一个洞消失的时刻。换句话说，你应该找一个最早的时刻，使得之后再也没有洞。

数据范围：$1 \le n \le 100$，$-10^4 \le x_i, y_i \le 10^4$

时间限制：1s

【问题分析与程序设计】

只要枚举任意三个点形成的三角形的外接圆的圆心。

时间复杂度 $O(n^3)$

空间复杂度 $O(n)$

k-Maximum Subsequence Sum

CF 280 D

【题目大意】

给出 n 个数 $A_1..A_n$，Q 个操作，操作有两种：

操作一：'= L R x'，把 $A_L..A_R$ 赋值为 x

操作二：'? L R k'，计算并输出$\sum_{i=L}^{R} Ai * (i - L + 1)^k$，对 10^9+7 取模

数据范围：$1 \le n$，$Q \le 10^5$，$0 \le k \le 5$，$0 \le A_i \le 10^9$

时间限制：5s

【问题分析与程序设计】

算法：

建线段树，每个点上记 10 个值：

lc, rc：左儿子，右儿子

l, r：当前区间的左端点，右端点

$S_0..S_5$：$S_k = \sum_{i=l}^{r} Ai * (i - l + 1)^k$

合并的时候用公式推一下即可。

时间复杂度 O(5 * 5 * $\log_2$n * Q)

空间复杂度 O(20* n)

【实现细节】

在合并的时候要注意 10 * (10^9 + 7) * (10^9 + 7)会爆 long long。

Cow Tennis Tournament

CF 283 E

【题目大意】

有一场比赛有 n 个人参加，每个人都有一个能力值 s_i，能力值两两不相同，任意两个人只会交手一次，能力值高的总是击败能力值低的。

现在要改变 k 次比赛结果，一次改变中有两个参数 a_i, b_i，对于任意两个能力值分别为 x, y 的人来说，如果 $a_i \le x \le b_i$ 且 $a_i \le y \le b_i$ 那么这两人交手时，原来赢的人会输，输的人会赢。

现在要统计有多少个三元组(p, q, r)满足 p 打败 q，q 打败 r，r 打败 p。

数据范围：$3 \le n \le 10^5$，$0 \le k \le 10^5$，$1 \le a_i < b_i \le 10^9$，$1 \le s_i \le 10^9$

时间限制：1s

【问题分析与程序设计】

算法：

对于这道题可以把获胜关系看成是一条边，从赢的人指向输的人，首先可以知道的是：满足条件的三元组 + 不满足条件的三元组 = $\binom{n}{3}$

那么最终答案 = $\binom{n}{3}$ - 不满足条件的三元组

可以发现一个不满足条件的三元组中一定存在一个点有边指向另外两个点，而另外两个点之间的关系没有影响。用 w_i 来表示点 i 的出度，也就是第 i 个人能够战胜的人的个数。

那么不满足条件的的三元组 = $\sum_{i=1}^{n}\binom{wi}{2}$

所以 ans = $\binom{n}{3}$ - $\sum_{i=1}^{n}\binom{wi}{2}$

再来考虑这个 w_i 如何计算，也就是第 i 个人能够战胜的人的个数。

设 f[i][j]表示能力值为 i 的人与能力值为 j 的人对战的结果，如果一次改变的参数是 a, b，那么就相当于将[a, b] * [a, b]这个矩阵里的所有交战结果取反，这个可以用线段树来解决。将这个矩阵的行看成时间，修改看成事件，把所有数离散化之后，把所有修改按时间排序，在时间开始的时候对线段树进行一次操作，结束的时候再进行一次操作。

线段树之中只要维护区间的和即可。

时间复杂度 $O(\log_2 n * m)$

空间复杂度 O(n)

期望得分：100

【实现细节】

要先对 n 个人根据能力值排序，注意在算第 i 个人能够战胜的人的时候要分两部分进行计算[1, i - 1]和[i + 1, n]，矩阵中初始都是 0。

Positions in Permutations

CF 285 E

【题目大意】

P 是 n 个互不相同且不超过 n 的正整数的一个排列。我们设排列 P 的第 i 个元素为 P[i]，n 为排列的长度。

我们称排列中的第 i 个位置是完美的，当且仅当|P[i]-i|=1。

请求出长度为 n 的而且完美的位置数刚好为 k 的排列数是多少。答案要求取模 10^9+7。

数据范围：$1 \le n \le 1000,\ 0 \le k \le n$

时间限制：1s

【问题分析与程序设计】

DP，在后面加一个数考虑这个数字放在哪里。

时间复杂度 O(n * k)

空间复杂度 O(n * k)

Lady's shop

CF 286 E

【题目大意】

给定 n 个背包，每个背包都只能装总重量恰好等于它能承受的重量的物体。求 k 个重量不同的物体，每个物体有无穷个。每一种取物体的方案，若总重量小于等于 m，则必须存在一个背包能够装这些物体。同样的每个背包必须存在一种取物体的方案，使得物体总重量等于它能承受的重量。

求 k 尽可能小的这样的一个物体重量的序列。

数据范围：$n \le 10^6, m \le 10^6$

时间限制：4s

【问题分析与程序设计】

首先考虑有解的情况。因为物体的选取方案和背包是一一对应的，所以单个物体的重量必须存在与 m 个背包之中。也就是说物体重量的集合是背包重量集合的一个子集。

这样，原题就转化成了另一个问题：考虑每个背包是否必要，也就是考虑这个背包的重量是否可以由重量比它小的背包相加而得。

如果可以，那么这个背包的重量不一定会在物体重量的集合中；如果不可以，必然会有一个物体的重量对应这个背包的重量。

算法 1：

暴力求每个背包是否可以由重量比它小的背包相加而得。Hash[i] = true 表示可以由其他背包的重量累加得到 i 这个重量。

Hash[i] = hash[i – a[1]] or hash[i – a[2]] or … or hash[i – a[n]].（a[i]表示背包能承受的重量）

时间复杂度 O(n*m)

空间复杂度 O(m)

这个方法的效率其实是比较低下的，没有用好物体的选取方案和背包是一一对应的这个性质。

算法 2：

因为物体的选取方案和背包是一一对应的存在一个背包的质量 $a[i] = p[i_1] + p[i_2] + \dots + p[i_n]$，那么必然存在一个背包的质量 $a[j] = p[i_1] + p[i_2] + \dots + p[i_{n-1}]$。所以如果有解，一个非必要的背包的质量必然可以由两个质量比它小的背包相加而得。

因此对第 i 个背包是否需要只需要判断是否存在 a[j] + a[k] = a[i] 即可。

时间复杂度 O(n*n)

空间复杂度 O(n)

这个方法很好的利用了题目里给出的性质，但是在时间复杂度几乎没有任何优化。

算法 3：

在算法 2 中，实质是要求任意两个数相加的和。如果将 a[i]看作是指数，这就等价于做多项式乘法。可以利用快速傅里叶变换来加速。

时间复杂度 $O(m*\log_2 m)$

空间复杂度 O(m)

【实现细节】

在算法的实现中，我们需要注意一下精度问题，最后判断时需要 + - eps。

Distinct Paths

CF 293 B

【题目大意】

小美有一个 n*m 的木板，一些块已经被涂上给出的 k 种颜色中的一种。你需要把每个没涂色的块涂色使得从左上角到右下角的每条路径都不会经过两个颜色一样的块。路径只能向右或向下走。输出答案%1000000007。

数据范围：$1 \le n, m \le 1000$，$1 \le k \le 10$

时间限制：1s

【问题分析与程序设计】

仔细阅读题目可以发现 $n + m \le 11$，所以直接暴搜加剪枝解。

时间复杂度 O(?)

空间复杂度 O(n)

Ksusha and Square

CF 293 D

【题目大意】

给出一个凸多边形，任取形内的两个点（包括边界，必须是整点），以这两个点的连线为对角线做正方形，求正方形的期望面积。

数据范围：$1 \le n \le 10^5$，坐标的绝对值 $\le 10^6$

时间限制：1s

【问题分析与程序设计】

其实对于最后的答案就是$\frac{\sum(x_i-x_j)^2+(y_i-y_j)^2}{2}$

对于 x、y 两个坐标可以分开来计算，计算对应的 x 坐标，观察上面那个式子可以发现，如果把当 x = k 是满足(x, y)在凸包内的 y 坐标的个数记为 c_k，那么最终答案就是 $c_2 * c_1 + c_3 * (c_2 + c_1 * 4) + c_4 * (c_3 + c_2 * 4 + c_1 * 9) + \ldots\ldots$

转换一下就是求 $c_1, c_2 + c_1 * 4, c_3 + c_2 * 4 + c_1 * 9, c_4 + c_3 * 4 + c_2 * 9 + c_1 * 16\ldots\ldots$

再进行一次差分就变成了 $c1, c2 + 3 * c1, c3 + 3 * c2 + 5 * c1, c_4 + 3 * c_3 + 5 * c_2 + 9 * c_1\ldots$

这个只要前缀和什么维护一下，求这个 c 只要分上下凸壳直线求交就可以了。

时间复杂度 O(n)

空间复杂度 O(n)

【实现细节】

比较坑的是本题没有保证不存在三点共线的情况，有的算法可能莫名其妙的错。

Close Vertices

CF 293 E

【题目大意】

你得到了一棵包含 n 个点的树，树上的每条边有一个非负边权，树上两点间路径的长度是该路径包含的边数，树上两点间路径的权重是指该路径包含的边的边权之和。

我们说两点是“相邻”的，当且仅当，存在一条连接该两点的路径，满足该路径的长度小于等于 L,且权重小于等于 W。

统计有多少个点对(u,v)，满足 $u < v$，且 u,v 是相邻的。

数据范围：$1 \le n \le 10^5$，$1 \le L \le n$，$1 \le W \le 10^9$

时间限制：1s

【问题分析与程序设计】

算法：

本题可以采用树分治的方法解决，具体过程如下：

1.找到这棵树的重心

2.计算所有路径经过这个重心的点对对于答案的贡献

3.删去重心，这棵树分成若干棵不相交的树，再对新树重复以上操作

这是树分治的主要流程，不难发现分治的层数最多只会有$\log_2 n$层。

关键在于如何在每一层中统计答案，对此，我们不妨把重心看成这棵树的根，一条过根的路径可以拆成由两条从根出发路径。所以我们记点 u 到根的边权和为 x_u，深度为 y_u，对于两个点 u, v 来说，只要满足 $x_u + x_v \le W$ 且 $y_u + y_v \le L$ 则 ans + 1

我们可以离线的做这个问题，把所有点的 x，y 都求出来以后按 x 排序，另一维用树状数组维护就可以了。

时间复杂度 $O(n * \log_2 n * \log_2 n)$

空间复杂度 $O(n * \log_2 n)$

【实现细节】

在统计的时候要注意必须要满足的一个条件是 u，v 属于不同的子树。对于这个限制我们可以先不管，最后把多出来的剪掉即可。

至于为什么第一维要取边权和，第二维取深度是因为深度的最大值只有 10^5，而边权的最大值达到了 10^9，虽然这没有什么太大变化，但是这道题只有 1s，要注意树分治的常数不能写的太大。

Shaass and Painter Robot

CF 294 D

【题目大意】

给你一个 N*M 的网格，一开始都是白色的。上面有一个机器人，一开始位于格子（X，Y）上（占据整个格子），面朝某个方向（左上，左下，右上，右下之一）。然后机器人会一直顺着这个方向走下去，每当遇到边界时会遵循光的反射定律改变方向，然后继续走。每当机器人走到一个格子后，它会将这个格子染黑，用掉一个单位颜料。即便这个格子已经被染黑了，也需要耗费一个单位颜料。当机器人意识到这个 N*M 的网格已经变成黑白相间的时候，它会立即停止行动。现在希望你求出，机器人停下来的时候，已经耗费了多少颜料？或者指出永远不可能停下来。

数据范围：$1 \le N,M \le 10^5$

时间限制：1s

【问题分析与程序设计】

暴力模拟所有边界上的点。

时间复杂度 O(n + m)

空间复杂度 O(n + m)

Yaroslav and Algorithm

CF 301 C

【题目大意】

你需要设计一种程序，这个程序读入一个字符串 st，你能使用的命令只有两种：

1. “s>>w”将 s 这个串替换成 w 后，重新开始整个程序。
2. “s<>w”将 s 这个串替换成 w 后，直接结束这个程序。

命令按照先后依次执行。

每次替换是找到 s 在 st 中第一次出现的位置，只替换一次，如果没有找到就不执行。

s，w 只能包含?与数字，长度小于等于 7。

现在给出 n 个数 $a_1..a_n$，你的这个程序要能使得输入任意一个 a_i 都能变成 a_i+1。

PS：

·命令的条数不能超过 50。

·算法需要对每个给出的数+1。

·为了得到结果，算法必须对于每个输入都执行不超过 200 步。

数据范围：$5 \le n \le 100$，$1 \le a_i \le 10^5$

时间限制：1s

【问题分析与程序设计】

算法：

可以直接构造，一开始在整个数之前加一个?（如 3241 变成?3241），然后通过若干例如?1>>1?的命令将问号移到最后一位（如?3241 变成 3241?），紧接着将这个?变成??（如 3241? 变成 3241??），这一步是为了与之前的命令区分开来，最后通过??来处理进位问题，对最后一位进行讨论。

时间复杂度 O(1)

空间复杂度 O(1)

【实现细节】

主要是命令的种类的选择和顺序的安排问题。

Yaroslav and Algorithm

CF 301 E

【题目大意】

Yaroslav 称数列 a_1，a_2，…，a_r 为良好的，当它满足：

1.$|a_1 - a_2| = 1$，$|a_2 - a_3| = 1$，…，$|a_{r-1} - a_r| = 1$，$|a_r - a_1| = 1$

2. $a_1 = \min_{1 \le i \le r} a_i$

Yaroslav 称数列 b_1，b_2，…，b_r 为优秀的，当它满足：

1. 数列中的元素不下降
2. 满足 $1 \le r \le n$ 、 $1 \le b_i \le m$
3. 通过重排数列中的元素可以得到至少一个至多 k 个不同的良好数列。

Yaroslav 有三个整数 n，m，k 。他需要知道有多少个不同的优秀数列。帮助 Yaroslav！考虑到答案可能相当大，将答案模 1000000007 (10^9+7)

两个数列$\{x_n\}$、$\{y_n\}$被认为不同，仅当存在一个位置 i，使 $x_i \neq y_i$

数据范围：n，m，k≤100

时间限制：1s

【问题分析与程序设计】

DP，每次考虑在最后加上一个数的方案数。

Rotatable Number

CF 303 D

【题目大意】

可旋转数：它通过旋转得到的数都是它自己乘以 1,2,3...,n（从 1 到数的长度）。旋转一个数就是将它的最后一位数字放到最前面。

例如：

142857 * 1 = 142857

142857 * 2 = 285714

142857 * 3 = 428571

142857 * 4 = 571428

142857 * 5 = 714285

142857 * 6 = 857142

二进制下的：

0011 * 1 = 0011

0011 * 10 = 0110

0011 * 11 = 1001

0011 * 100 = 1100

求最大的 b(1 < b < x)，满足在 b 进制下存在长度为 n 的正“可旋转数”（允许有前导零）。

数据范围：$1 \le n \le 5*10^6$，$2 \le n \le 10^9$

时间限制：1s

【问题分析与程序设计】

算法：

这道题有一个结论“可旋转数”= $\frac{b^{p-1}-1}{p}$，然后直接暴力枚举找到第一个可行的。

时间复杂度 O(?)

空间复杂度 O(?)

Playing with String

CF 305 E

【题目大意】

两个人进行游戏，场上有若干个字符串，每次可以选择一个字符串 S，对于这个字符串 S，选择其中第 i（$1 < i < |S|$）个字符满足 $S_{i-1} = S_{i+1}$，然后将 S 这个字符串分成三份 $S_1..S_{i-1}$，S_i，$S_{i+1}..S_{|S|}$

初始只有有一个长度为 n 的字符串，问先手是否有必胜策略，如果有，输出第一步操作。

数据范围：$1 \le n \le 5000$

时间限制：1s

【问题分析与程序设计】

这道题可以用 SG 函数来求解。

时间复杂度 $O(n^2)$

空间复杂度 O(n)

期望得分：100

White, Black and White Again

CF 306 C

【题目大意】

Polycarpus 的生活总是满足“一些好事，然后一些坏事，然后一些好事”这样的规律。所以 Polycarpus 认为接下来的 n 天也是满足这样的规律的。

Polycarpus 知道，接下来会发生 w 件两两不同的好事和 b 件两两不同坏事，每天至少发生一件事，每天要么全部发生好事要么全部发生坏事。

由于 Polycarpus 的规律，这 n 天会先有若干天发生好事，再有若干天发生坏事，再有若干天发生好事。(若干代指>0)

要求统计事件发生的方案数（每天发生的事的顺序也不一样），答案取模 10^9+9 输出

数据范围：$3 \le n \le 4000$，$2 \le w \le 4000$，$1 \le b \le 4000$，$n \le w + b$

时间限制：1s

【问题分析与程序设计】

算法：

因为题目要求一定是一些好事，一些坏事，一些好事，而且事情是不同的。先忽略 n 天的限制，考虑将所有好事排列好，再把一个坏事的排列插到这个好事的排列中，这样就满足了一些好事+一些坏事+一些好事的要求，这部分的方案数是$w! * b! * (w-1)$。在来考虑把这些事分配给 n 天，如果没有“一天只能全是好事或者全是坏事”，那么这部分的方案数就是$\binom{w+b-1}{n-1}$，如果加了这么一个限制条件就会发生问题，就是好事与坏事连接的两个部分会有可能将好事与坏事分在了同一天，这种情况只会在两天里会发生。而且如果不合法，那么这一天中也一定是一些好事+一些坏事，所以预留两天和两件事来使得这可能不合法两天变得一定合法，所以最终这部分的方案数就是$\binom{w+b-3}{n-3}$。最后的答案只要将两部分乘起来即可。

$$ans = w! * b! * (w-1) * \binom{w+b-3}{n-3}$$

时间复杂度 O(n)

空间复杂度 O(n)

Polygon

CF 306 D

【题目大意】

构造一个 n 个顶点的凸多边形，要求这个凸多边形的每个角的角度相同，每条边的长度不同。无解输出-1

数据范围：$3 \le n \le 100$

时间限制：1s

【问题分析与程序设计】

算法：

可以直接构造，每次把边长加一个较小的值即可。

时间复杂度 O(N)

空间复杂度 O(N)

期望得分：100

【实现细节】

边长的增量不能取得太大，太大会有一系列问题。起点可以从原点出发，最后一条边的边长不要考虑。只有四边形无解。

Sereja and Squares

CF 314 E

【题目大意】

给出一个序列，序列中元素为除了‘x’和‘X’的大写字母和小写字母，现在将小写字母与在它之后某个与之对应的大写字母连线（例如 a 与 A，b 与 B），如果存在一种连线方案，使得线段之间不存在部分相交（$l_1 < l_2 < r_1 < r_2$），这个序列即为合法序列。

现在将一个合法序列所有大写字母以及一些小写字母改成‘？’，问有多少种不同方案能将修改之后的序列改回一个合法序列。方案数对 4294967296 取模。

数据范围：序列长度 $N \le 10^5$

时间限制：4s

【问题分析与程序设计】

算法：

这个问题其实可以看成是括号匹配问题，小写字母对应左括号，大写字母对应右括号。题目给出部分左括号，‘？’既可以做左括号，又可以做右括号。在做左括号时有 25 种方案；在做右括号时必然与之前的某个左括号对应，所以只有唯一的一种方案。

括号匹配的一般做法是将左括号看成是 1，右括号看成-1，在从左往右扫的过程中和不能为负。

设计 DP 状态，f[i][j]表示当前做到第 i 位，当前和为 j 的方案数。

转移方程为$f[i][j] = \begin{cases} f[i-1][j-1] & \text{（若当前第 i 位是小写字母）} \\ f[i-1][j-1]*25 + f[i-1][j+1] & \text{（若当前第 i 位是‘？’）} \end{cases}$

时间复杂度 $O(N^2)$

空间复杂度 $O(N^2)$

期望得分：70~80

如果直接做 DP 会超时，而且会超空间。这个时候需要常数优化以及滚动数组。

【实现细节】

常数优化一：因为状态的第二维最大只会到 $\frac{N}{2}$，当大于这个值的时候就没有意义，以为左括号最多只会有 $\frac{N}{2}$ 个。

常数优化二：其实第一个优化可以做的更好，对于第一维 $1 \le i \le \frac{N}{2}$ 时，第二维只需要做到 i 即可；对于第一维 $\frac{N}{2} < i \le N$ 时，第二维只需要做到 N - i 即可。

对于本题，只需要以上两个常数优化即可获得满分，但是其实还是可以做的更好。

常数优化三：第一维和第二维的奇偶性必须相同。

PE lesson

CF 316 D

【题目大意】

一共有 n 个人站成一排，初始第 i 个人手中的球编号为 i，现在进行若干次传球，一次传球就是选择两个人 x, y，然后他们两个人手中的求会互换，即 x 拿到 y 手中的球，y 拿到 x 手中的球，由于有体力限制，有些人只能传一次球，而剩下的人可以传两次球。

求所有可能的进行完若干次传球后求的排列的个数，模 $10^9 + 7$。

数据范围：$n \le 10^6$

时间限制：1s

【问题分析与程序设计】

算法：

设只能传一次球的人数为 A，能传两次球的人数为 B，先来考虑只能传一次球的人，f[i] 表示只能传一次球的人数，那么有两种转移，考虑这第 i 个人如果没有传球那么方案数就是 f[i – 1]，如果这第 i 个人跟一个人进行了一次传球，那么方案数就是(i - 1) * f[i - 2]，因为有 i – 1 个人可以供我选择，而且这个人一旦跟第 i 个人进行了一次传球就与其他人无关了。所以$f[i] = f[i-1] + (i-1) * f[i-2]$

接下来考虑能传两次球的人，这 B 个人是没有限制的，所以可以把这 B 个人直接插入 A 个人的方案中，这 B 个人中有的是自成一组，有的是插到以前的 A 个人形成的组里，如果是插到原来的组里，就相当与把一条边断开变成两条边，所以最终答案就是

$$ans = (A+B)! * \frac{f[A]}{A!}$$

时间复杂度 O(n)

空间复杂度 O(n)

【实现细节】

要注意 f[0] = 1，否则如果只有能传两次球的人的时候答案就是 0 了。

PE lesson

CF 316 D

【题目大意】

在这个问题中，每个时刻您都有一个区间的集合。您每次可以从集合中的区间(a, b)移动到另一个满足 c < a < d 或者 c < b < d 的区间(c, d)。您需要判断是否有一种从区间 x 到区间 y 的移动方案。

数据范围：$n \leq 10^5$

时间限制：1s

【问题分析与程序设计】

分层统计，线段树暴力维护即可。

时间复杂度 $O(n * \log_2 n)$

空间复杂度 $O(n)$

Ciel and Flipboard

CF 321 D

【题目大意】

n*n 的矩阵（n 为奇数）上每个格子都有一个整数，现在可以执行一个操作，就是选取 x*x 的矩形区域把里面所有的整数取反，$x=(n+1)/2$，操作可以反复执行，问最后 n*n 矩阵的所有元素之和的最大值是多少。

数据范围：$n \le 33$

时间限制：1s

【问题分析与程序设计】

算法：

这是一道比较难想的枚举题，首先因为每次取的是一个大小为 x * x 的矩阵，所以根本没有办法去压位，或者是做 DP。

想要做这道题，需要发现最关键的两个等式。

记 f[i][j]为位置 a[i][j]最后翻转情况，取值为-1 或者 1

由于 x 恰好等于 n 的一半多一点，也就是如果只考虑行，每一行的第 x 列肯定会被翻转，而且第 j 列与第 j+x 列肯定不能被同时翻转，但定会有一个被翻（这个可以画图理解）

行的话也是这样，这也就可以得到两个等式

f[i][j] * f[i][x] * f[i][j + x] = 1

f[i][j] * f[x][j] * f[i + x][j] = 1

这就是说确定了任意两个都可以唯一确定第三个

可以发现第 x 行和第 x 列是非常特殊的，我们可以枚举前 x 行上第 x 列的 f，确定了这 x 数时候，通过上面的两式子中的某一个可以推出整个第 x 列的 f。之后对于第 x 行来说，前 x – 1 个数之间的取值是互不影响，可以单独枚举来判断优劣，这是因为对于不是第 x 列或是第 x 行的数来说只有 f[i][j], f[i + x][j], f[i][j + x], f[i + x][j + x]这四个值是互相影响。当确定了第 x 行和第 x 列之后，同样只有单独枚举左上角这个大小为(x - 1) * (x - 1)的矩形中每个位置的取值即可。而确定了第 x 列之后，f[x][i]的取值只会影响第 i 列的 f 的取值，所以可以单独枚举。

时间复杂度 $O(2^x * (x - 1)^2)$

空间复杂度 $O(n^2)$

期望得分：100

【实现细节】

最后的那两次单独的枚举是不需要的真的枚举的，可以通过取绝对值来取得最大值。

Tournament-graph

CF 323 B

【题目大意】

你要构造一个有 N 个结点的竞赛图，使得对任意两个结点 u 和 v（$u \neq v$），从 u 到 v 的最短距离不超过 2。

竞赛图就是基图为无向完全图的有向图(每对结点之间有一条有向边相连,且无自环)。

数据范围：$3 \leq N \leq 1000$

时间限制：1s

【问题分析与程序设计】

算法：

只是一道构造题，具体的构造方法是：

一开始先添加 N 边，使其构成一个环，接着把一种一个点拎起来，就可以看成一条链加一条返祖边，链上的节点依次标号为 1..N，然后添加所有从 i 指向 i + 2 的边，再添加从 i + 3 指向 i 的边，以此类推。事实证明最后只有 N = 4 是无解。

时间复杂度 $O(N^2)$

空间复杂度 $O(N^2)$

期望得分：100

【实现细节】

具体实现的时候可以看两个点编号的差值的奇偶性来判断这条边的方向。

Two permutations

CF 323 C

【题目大意】

你有两个各包含 n 个元素的排列 p 和 q，和 m 个由 l_1，r_1，l_2，r_2 组成的询问。每次询问同时出现在 p 中位置在$[l_1,r_1]$,在 q 中位置在$[l_2,r_2]$中的数的数量。

询问强制在线。

数据范围：序列长度 $n \le 10^6$，$m \le 10^5$

时间限制：1s

【问题分析与程序设计】

算法 1：

直接哈希暴力统计。

时间复杂度 O(n * m)

空间复杂度 O(n)

期望得分：30

算法 2：

用数据结构来优化，将第二个排列作为第一个排列的下标插入权值线段树中。l_1 到 r_1 之间出现的数字就是合并两棵线段树，询问新的线段树中 l_2 到 r_2 之间的数字个数即可。

时间复杂度 O(n * $\log_2 n$ + m * $\log_2 n$)

空间复杂度 O(n * $\log_2 n$)

期望得分：100

The Red Button

CF 325 C

【题目大意】

有 N 个节点编号为 0 到 N-1，节点 i 可以到达节点$(2*i) \bmod n$ 或节点$(2*i+1) \bmod n$，求是否可以从节点 0 开始不重复的遍历所有节点最后再回到 0，如果可以，输出访问序列；否则输出-1

数据范围：$N \le 10^5$

时间限制：1s

【问题分析与程序设计】

算法：

为了方便起见，之后说到的数可能会省略 mod N 这个条件。

首先可以证明的是当 n 为奇数时，不存在解。

考虑节点 0，对于节点 0 来说满足以下两个条件之一的点才能最后回到它：

$2H = 0$

$2H + 1 = 0$

满足第一个式子的只有 $H = 0$，而满足第二个式子的只有 $H = \left\lfloor \frac{N}{2} \right\rfloor$

因为题目要求形成一个环，所有只有$\left\lfloor \frac{N}{2} \right\rfloor$才能到达 0

现在再来考虑点 $N-1$， 同样的只有满足以下两个条件之一的点才能到达它：

$2H = N - 1$

$2H + 1 = N - 1$

这次满足第二个式子的只有 $H = N-1$ 本身，而满足第一个式子的只有 $H = \left\lfloor \frac{N}{2} \right\rfloor$

因为同时存在两个只能由$\left\lfloor \frac{N}{2} \right\rfloor$才能到达的点$(0, N-1)$，而$\left\lfloor \frac{N}{2} \right\rfloor$只能有一条出边，所以当 N 为奇数的时候，不存在合法的方案。

当 N 是偶数的时候会有一些奇特的性质：

考虑节点 X 能到达点 $2*X, 2*X+1$，而点 $X + \left\lfloor \frac{N}{2} \right\rfloor$能够到达点 $2*X+N, 2*X+N+1$，可以发现两者在 mod N 的时候是等价的，而对于任意一个点 P，也有且只有两个点可以到达它：$\left\lfloor \frac{P}{2} \right\rfloor$ 和 $\left\lfloor \frac{P+N}{2} \right\rfloor$。所以点 $X, X + \left\lfloor \frac{N}{2} \right\rfloor$与点 $2*X, 2*X+1$ 之间存在对应关系，当你选择了从 X 走到 $2*X$ 的时候，你就不得不选择从 $X + \left\lfloor \frac{N}{2} \right\rfloor$走到 $2*X+1$。

问题到这里就可以直接用一个 DFS 解决构造问题了，DFS 从 0 开始。对于节点 X：

1. 如果当前节点已被访问过，直接退出；否则设置为已访问。
2. 考虑 X 从哪个点走过来，$\left\lfloor \frac{X}{2} \right\rfloor$ 或 $\left\lfloor \frac{X+N}{2} \right\rfloor$，选择没有走过的一个继续 DFS
3. 将点 X 加入队列里

时间复杂度 O(N)

空间复杂度 O(N)

【实现细节】

因为是考虑从那个点走到 X 的，所以最后的序列是反的，要反序输出。

Reclamation

CF 325 D

【题目大意】

有一个 r * c 的地图，把左边界和右边界粘起来使得形成一个圆柱，现在要不断地挖去其中的格子，要求任何时候都存在一条从最上方到最下方的路径(四联通)，如果某次操作不满足要求则不做，总共 n 次操作，问最后有多少次操作是成功的。

数据范围：$r, c \le 1000$，$n \le 10^5$

时间限制：1s

【问题分析与程序设计】

这个问题可以转化成将这个地图向右再复制一遍，如果(x, y)与(x, y + c)八连通就不可行了，这个用并查集维护一下就可以了

时间复杂度 O(r * c)

空间复杂度 O(r * c)

Theft of Blueprints

CF 332 D

【题目大意】

给出一个 n 个点的带权无向图，满足对于任意一个大小为 k 的顶点集合 S，有且仅有一个点与 S 每一个点都有边。令这个点为 v(S)，并且对 S 进行操作的代价是 S 中每个点与 v(S) 的边权之和。现在求对于一个大小为 k 的子集操作代价的期望。

数据范围：$1 \le k < n \le 2000$

时间限制：1s

【问题分析与程序设计】

因为题目里给出了一个很强的条件：满足对于任意一个大小为 k 的顶点集合 S，有且仅有一个点与 S 每一个点都有边。根据这个条件可以推断出：

1. 一共有$\binom{n}{k}$这么多个不同的集合
2. 每一条边在这么多种方案中只会出现恰好两次

若用 w_{ij} 表示从 i 到 j 这条边的权值，那么最后的答案就是$\frac{\sum w_{ij}}{\binom{n}{k}}$

时间复杂度 $O(n^2)$

空间复杂度 $O(n^2)$

【实现细节】

要注意的是当 $k=2$ 时是一个特例，用 d_i 表示表示第 i 个点的度，那么答案是$\frac{\sum(w_{ij}*(d_i-1))}{\binom{n}{k}}$

Lucky Tickets

CF 333 D

【题目大意】

Gerald 有一个朋友 Pollard。Pollard 对 lucky tickets 十分感兴趣（ticket 是一个数列）。一开始他认为：如果在这些数之间加入运算符号和括号使得最终结果等于 100，那么这个 ticket 就是 lucky 的。但是他很快就分析出了所有的 lucky ticket，所以他决定研究更一般性的问题，即 k-lucky ticket。

对于一个 ticket，如果我们在它一些数的左边或者右边加入一些运算符号(“+”, “-”, “*”)和括号使得最终结果等于 k,那么这个 ticket 就是 k-lucky ticket。

举个栗子，“224201016”就是 1000-lucky ticket ：

$(-2-(2+4))\times(2+0)+1016=1000$

Pollard 参加了一个研究 k-lucky ticket 的组织，他请求你帮他找出 m 个 k-lucky ticket

每个 k-lucky ticket 严格为 8 位数字，可以有前导 0，每个 k-lucky ticket 必须不同，数据保证存在超过 m 个 k-lucky ticket。

数据范围：$0 \le k \le 10^4$, $1 \le m \le 3 * 10^5$

时间限制：1s

【问题分析与程序设计】

对于这道题目而言，首先因为每个 k-lucky ticket 严格为 8 位数字，而且 k 最多只有 4 位数（10^4 类似），那么一开始就可以把 k 写成 a + b, b + a, a – b, -a + b, b – a, -b + a, a * b, b * a 这 8 种形式，可以默认 a, b 都是一个 10^4 以内的数，现在的问题就已经转化成用 4 个位置去表示一个四位数。这个问题和之前的问题类似，也是把这个数拆成两个数，但是要求这两个数拼起来的长度不超过 4。

时间复杂度 O(n$\sqrt[2]{n}$)

空间复杂度 O(n)

Optimize!

CF 338 E

【题目大意】

```
getAnswer(a[1..n], b[1..len], h)
  answer = 0
  for i = 1 to n-len+1
    answer = answer + f(a[i..i+len-1], b, h, 1)
  return answer
f(s[1..len], b[1..len], h, index)
  if index = len+1 then
    return 1
  for i = 1 to len
    if s[index] + b[i] >= h
      mem = b[i]
      b[i] = 0
      res = f(s, b, h, index + 1)
      b[i] = mem
      if res > 0
        return 1
  return 0
```

帮助优化这段代码。

数据范围：$1 \le len \le n \le 150000$，$1 \le a_i, b_i, h \le 10^9$

时间限制：1s

【问题分析与程序设计】

算法：

通过分析这段代码可以发现 f 这个函数其实就是给你两个长度为 len 的数据，判断是否存在一种排列方案使得 $\min_{1 \le i \le len}(a_i + b_i) \ge h$。

先不考虑 getAnswer，就只是优化 f 这个函数：

一个比较显而易见的方法，将 a 从大到小排序，b 从小到大排序，然后对应位置相加。

这个方法看起来是一个贪心，但实际是可以证明的，可以来分析一下这个方法的本质。这会有助于帮助我们解决之后带来的问题。

将 b 从小到大排序之后，考虑每一个 a_i 存在一个最小的 k_i 使得 $a_i + b_{ki} \ge h$，因为 k_i 是随着 a_i 的减小而增大的，所以排序后直接判断是可以的。

但是现在来考虑 getAnswer，这个函数非常好理解，就是将 a 的长度扩展到 n，然后将 n – len + 1 个连续子串和 b 做询问。

如果还是继续用排序来代替 f 函数，可以发现在这里是没有办法优化排序之后判读合法的复杂度的，所以要用到之前提到的 k_i，对于 n 个 a_i 来说各自的 k_i 都是固定的，如果设 T_i 表示 len 个数对于 b 数组中对于第 i 个数的需求，那么每一个 a_i 对于 T 的贡献就是 $T_{ki}..T_n$ 都加上 1，而判断是否存在合法的方案只需要判断 $T_i \ge i$ 即可，这是显而易见的。

有了一种新的判断方法之后可以发现，现在从 $a_i..a_{i+len-1}$ 转移到 $a_{i+1}..a_{i+len}$ 只需要将一个区间加一减一，而判断 $T_i \geq i$ 这个条件可以将其转化一下，变成 $T_i - i \geq 0$，因为 i 是一个定值，在一开始就将 i 减掉，最后判断全局最小值是否非负就可以了。

对于以上操作可以用线段树来维护。

时间复杂度 $O(\log_2 len * n)$

空间复杂度 $O(n)$

【实现细节】

因为有一些 a_i 的 k_i 会达到 len + 1，就表示无论如后都不可行，所以要注意特判一下，不然直接扔到线段树里会 RE。

Three Swaps

CF 339 E

【题目大意】

饲养员 Xenia 有 n（n>1）只站成一排的马。每只马都有它自己的独一无二的号码。最开始的时候，从左数第 i 只马编号为 i，也就是说，马的号码序列是下面这样（从左到右）：1，2，3，……，n

Xenia 在演出之前训练这些马。在练习期间，她会给马命令。每个命令是一对数 l，r（1<=l<r<=n）。命令 l，r 意思是从左数第 l 个位置到第 r 个位置上的马要被重新排列。站在 l，r 位置上的马交换位置；l+1，r-1 位置上的马交换位置；以此类推。也就是说，[l，r]位置内的马反向排列

例如，如果 Xenia 命令 l=2，r=5 并且命令之前马的编号序列为（2，1，3，4，5，6），那么命令之后序列就会是（2，5，4，3，1，6）。

我们知道在训练时，Xenia 最多最多给出三个命令。现在你已经知道了最后马的编号序列，请给出训练时 Xenia 的命令是什么。注意你不需要使得命令数最小，只需要找到一个在三次命令以内的正确的方案。

数据范围：$1 \le n \le 1000$

时间限制：1s

【问题分析与程序设计】

通过研究可以发现三次命令之后整个数列最多被分成了 7 块，即有 8 个分界点，每次翻转枚举两个分界点，最后进行验证就可以了。

时间复杂度 O(8! * n)

空间复杂度 O(n)

【实现细节】

在取两个决策点的进行一次翻转之后，这两个决策点之间的决策点的位置也会随之改变。

Candies Game

CF 341 E

【题目大意】

饲养员 Xenia 有 n（n>1）只站成一排的马。每只马都有它自己的独一无二的号码。最开始的时候，从左数第 i 只马编号为 i，也就是说，马的号码序列是下面这样（从左到右）：1，2，3，……，n

Xenia 在演出之前训练这些马。在练习期间，她会给马命令。每个命令是一对数 l，r（1<=l<r<=n）。命令 l，r 意思是从左数第 l 个位置到第 r 个位置上的马要被重新排列。站在 l，r 位置上的马交换位置；l+1，r-1 位置上的马交换位置；以此类推。也就是说，[l，r]位置内的马反向排列

例如，如果 Xenia 命令 l=2，r=5 并且命令之前马的编号序列为（2，1，3，4，5，6），那么命令之后序列就会是（2，5，4，3，1，6）。

我们知道在训练时，Xenia 最多最多给出三个命令。现在你已经知道了最后马的编号序列，请给出训练时 Xenia 的命令是什么。注意你不需要使得命令数最小，只需要找到一个在三次命令以内的正确的方案。

数据范围：$1 < n \leq 1000$

时间限制：1s

【问题分析与程序设计】

算法：

通过分析可以得出经过三次命令之后，一个有序的序列最多会被分成 7 段，所以只要把相邻两端的分界点找出来，每次选择两个分界点进行翻转即可，最后验证一下是否正确。

时间复杂度 $O(7^6 * n)$

空间复杂度 O(n)

【实现细节】

每次选择两个分界点进行翻转之后，在这两个分界点之间的分界点的位置也会发生改变。

Pumping Stations

CF 343 E

【题目大意】

给出 n 个点 m 条边的无向图。

求一个点的排列使得相邻两个点之间的最大流的和最大。

数据范围：$n \le 200, m \le 1000$

时间限制：1s

【问题分析与程序设计】

对于这个问题，看上去很没有思路。要解决这个问题需要借助 Gomory – Hu tree，它具有一下特性

1. 这棵树的点集等于原图的点集
2. 在这棵树上 u 到 v 路径上的最小值等于原图中 u 和 v 的最大流

算法：

首先要建出这棵树：

1. 基本过程就是每次在点集中任取两个点，做一遍最小割。
2. 通过割边把这个点集分成两个点集，在这两个集合之间建一条权值等于最小割的边，再递归处理。
3. 重复以上操作，直到只剩下一个点为止。

建出这棵树以后，可以证明最大流的和最大不会超过这棵树上的边权总和。接下来，需要构造一种方法使得最大流的和等于这棵树上的边权总和：

1. 每次找到树 T 上的权值最小的边，把这条边删去，这棵树分为了两棵树 T_A，T_B。
2. 任何一个 T_A 中的点到 T_B 中的点的最小割都等于这条边的权值。
3. 最后递归处理 T_A，T_B，直到只剩下一个点为止。

这样就可以求的点的排列了

时间复杂度 $O(n^4m)$

空间复杂度 O(m)

期望得分：100

这个方法的效率似乎是比较低下的，但是实际瓶颈在于建树时要做 n – 1 次最小割，而网络流的实际复杂度远达不到 n^2m，经测试可以在 0.5s 求出答案。

【实现细节】

在做无向图最小割的时候，两条边的流量开始都是 f。再按照一般网络流的做法，最后将两条边的流量相减来判断流向以及流量。

Pilgrims

CF 348 E

【题目大意】

有 n 个城镇，用 n-1 条双向道路连接起来。这些城镇可以两两互达。有 m 个修道院坐落在 m 个不同的城镇。每个修道院有一个教徒。在一年之始，每个教徒会选择离他最远的一个修道院。如果有多个，他会把所有的都列入清单。在“Big Lebowskiday”里，每个教徒会随机选择一个清单里的城镇开始走去。Walter 讨厌教徒。他想尽可能的通过阻止他们的行程来让尽可能多的人不开心。他计划摧毁一个没有修道院的城镇。一个教徒如果在他的清单里没有任何一个城镇能去，他就会不开心。

你需要求出 Walter 最多能让几个教徒不开心。除此之外，你还要计算他有多少种方法。

数据范围：$n \leq 10^5$

时间限制：1s

【问题分析与程序设计】

首先有一个比较直观的方法：对于每个修道院，把这个点看作树根，那么这个点对于答案的贡献就在于，把所有离它最远的修道院的最近公共祖先到根的这条路径上的点的贡献+1

算法：

对于之前的算法可以进行改进，直接做两遍树形 DP 即可，一次从下往上更新，一次从上往下更新。

从下往上更新的时候记录以这个点为根的子树里所有修道院的最近公共祖先。从上往下更新的时候将信息进行合并即可。

时间复杂度 O(n)

空间复杂度 O(n)

【实现细节】

在从上往下更新的时候要记录这个节点的所有儿子中第一大的到第三大的值。

Jeff and Removing Periods

CF 351 D

【题目大意】

你能对一个长度为 n 的序列 a[1], a[2], ..., a[n].执行下列操作:

1.选择三个整数 v, t, k (1 ≤ v, t ≤ n; 0 ≤ k; v + tk ≤ n), 要求满足 a[v] = a[v + t], a[v + t] = a[v + 2t], ..., a[v + t(k - 1)] = a[v + tk];

2.将 a[v] , a[v + t] , ..., a[v + tk]这 k+1 个数从序列中删除。将剩下的数字重新标号为 a[1] , a[2], a[3], ... , a[n-k-1]。

3.重新排列这 n-k-1 个数。

定义一个数列 a 的"美好度"为将所有数字全部删除需要的最小步数。

现在有一个长度为 m 的数列 b[1],b[2],..,b[m]。

有 q 个询问，每次询问区间[l,r]这个连续子序列的"美好度"(b[l], b[l+1] , ... b[r]这个子序列的"美好度")。

数据范围：m, q, b[i] ≤ 10^5

时间限制：1s

【问题分析与程序设计】

对于这个问题，我们发现其实只有第一次操作是有意义的，因为做完第一次操作后可以重新排列，重新排列后每一次操作都能删掉一种数字。所以只要看第一次操作是否可以删掉一种数字即可，如果可以，那么答案就是这段区间中数字总数，否则答案+1

算法：

求一段区间中数字的种类数，可以使用莫队算法离线求解。

而对于一段区间内是否存在一种数字的下标构成等差数列，只要先维护对于第 i 个数字，以它为结尾的下标构成的等差数列的首项所能出现的最早的位置。

之后只要在做莫队的时候维护一下每种数字的开始位置和结束位置即可。

时间复杂度 O(q$\sqrt[2]{m}$)

空间复杂度 O(m)

【实现细节】

在做莫队的时候，维护每种数字的开始位置和结束位置比较繁琐，可能需要加一些边界条件的判断，而且要记录当前这段区间有多少种数字的下标能构成等差数列，不能记录是否存在，而对于区间要先拓展，在缩小，否则区间可能为负。

USACO

Cow Patterns

USACO 2005 Dec

【题目大意】

给出一个长度为 N 的母串 a_i，以及一个长度为 M 的模式串 b_i，a_i, b_i 都是小于等于 S 的正整数。要求所有匹配的位置，并输出。母串中的一个子串 $a_k..a_{k+m-1}$ 和模式串 b_i 匹配成功的条件是，每个 a_i 在这一段中的排名和 b_i 的排名相等。

数据范围：$1 \le N \le 10^5$，$1 \le M \le 25000$，$1 \le S \le 25$

时间限制：1s

【问题分析与程序设计】

算法：

可以发现其实这道题的本质还是一个字符串匹配问题，只不过匹配的是排名。所以只需要 KMP+ 一个高效的求排名的算法。在本题中因为数的大小不超过 25，所以可以暴力求每个数的排名。

当这个 S 达到 10^9 的时候，其实只会有 10^5，因为一共只有 n 个数，所以 S 太大只需要离散化就可以了。那么当 S 达到 10^5 的时候，只需要借助树状数组来求 rank 就可以了。

时间复杂度 O((N + M) * S)

空间复杂度 O(N)

【实现细节】

在判断排名相等时不仅要判比它小的数的个数要相等，和它相等的数的个数也要相等。

Best Cow Line

USACO 2007 Dec

【题目大意】

给出一个长度为 N 的字符序列，每次可以选择取出队头或者队尾，将其插入一个初始为空的新的字符序列的队尾，求可能的字典序最小的新字符序列。

数据范围：$1 \le N \le 30000$，只包含大写字母

时间限制：1s

【问题分析与程序设计】

算法：

设 L 是原字符序列的队头，R 是原字符序列的队尾，如果当前取 L 比去 R 更优，那么肯定是取 L，反正取 R，唯一不确定的是两者的字典序相同的时候。

这个简单的贪心肯定是正确的，因为是要求字典序最小的，所以在当前位置放上一个字典序更小的字符肯定是最优的。

根据贪心的思想，当 L = R ('=' 表示两者所代表的位置上的字符相等)时，比较 L + 1 和 R − 1，如果还是相等，再比较 L + 2 和 R − 2 以此类推，如果比较的位置超出边界，那么则代表这一段序列里的字符都相同了，取 L 还是 R 是没有区别的。

虽然每次暴力比较是可以通过本题的测试数据，但是如果数据中存在一个点 N = 30000，所有字符都相同，那么暴力比较的就是超时。

可以发现这个暴力的过程是可以用二分来代替的，最终的位置是恰好两者不再相等的第一个位置，所以只要二分长度判断两者是否相等即可。

判断两者是否相等可以借助字符串哈希。

时间复杂度 $O(\log_2 N * N)$

空间复杂度 O(N)

Cow Neighborhoods

USACO 2008 Open

【题目大意】

了解奶牛们的人都知道，奶牛喜欢成群结队。观察约翰的 N 只奶牛，你会发现她们已经结成了几个“群”。每只奶牛都有一个不同的坐标。两只奶牛是属于同一个群的，当且仅当至少满足下列两个条件之一：

1．两只奶牛的曼哈顿距离不超过 C，即 $|x_1 - x_2| + |y_1 - y_2| \le C$。

2．两只奶牛有共同的邻居。即存在一只奶牛 k，使 k 分别与这两只奶牛同属一个群。

给出奶牛们的位置，请计算有多少个牛群，以及最大的牛群里有多少只奶牛。

数据范围：$1 \le N \le 10^5$，$1 \le x, y, C \le 10^9$

时间限制：1s

【问题分析与程序设计】

把原来的坐标改写成 $X = x + y$, $Y = x - y$，所以 $|x_1 - x_2| + |y_1 - y_2| \le C$ 就转换成了 $\max(|X_1 - X_2|, |Y_1 - Y_2|) \le C$，直接用队列和平衡树维护和这个点在一起的点，然后用并查集合并一下。

时间复杂度 $O(n * \log_2 n)$

空间复杂度 $O(n)$

Land Acquisition

USACO 2008 Mar

【题目大意】

给出 n 块地，每块地有一个宽度 Wi，长度 Li，一次性购买多块地的代价是这几地块的 max{Li}*max{Wi}，求买下所有地的最小代价。

数据范围：$n \le 50000$，$1 \le Wi, Li \le 10^6$

时间限制：1s

【问题分析与程序设计】

算法：

通过分析我们可以发现有一些地对于我们的计算是没有价值的，对于第 i 块地，如果存在一个 j 使得 Wj ≥ Wi 且 Lj ≥ Li，则第 i 块地是没有价值的。

去掉没有价值的地之后，可以发现 Wi 单调不减的时候 Li 单调减。

所以$f[i] = \min_{1 \le j < i} f[j] + W[i] * L[j+1]$

那么对于两个决策点 j, k(j < k)，如果决策点 j 优于决策点 k

则有$f[j] + W[i] * L[j+1] < f[k] + W[i] * L[k+1]$

等价于$\frac{f[j]-f[k]}{L[k+1]-L[j+1]} < W[i]$

直接用斜率优化即可

时间复杂度 $O(n + n * \log_2 n)$

空间复杂度 O(n)

【实现细节】

每次当一个新的节点入队的时候要维护整个队列的单调性，而取最优值的时候要取顶点。

Toys

USACO 2008 Nov

【题目大意】

已知对于连续 D 天，每天对玩具的需求量为 T_i，要求每天所使用的玩具必须是新购进的或者是经过消毒的。给定初始时买入玩具的单位价格为 T_c，同时提供两种消毒服务，第一种方式需要经过 N_1 天才能完成，单位价格为 C_1，第二种方式需要经过 N_2 天才能完成，单位价格为 C_2。求满足每天需求的最小总代价。

数据范围：$1 \le D \le 10^5$，$1 \le T_i \le 50$，$1 \le N_1, N_2 \le D$，$1 \le C_1, C_2 \le 60$

时间限制：1s

【问题分析与程序设计】

算法：

首先看到这道题可以想到去构造一个 D 个点的网络流模型，但是因为点数太多，所以肯定是会超时的，需要换一个思路。

对于这道题有一个比较明显的性质，就是当初始买入的玩具的个数确定之后，那么这个最优方案就可以通过贪心来求，具体如何实现会在下文中讲到。

通过上面的那个性质可以猜测这是一个单峰的函数，设最小代价 $f(x) = x * T_c + g(x)$，x 表示初始买的玩具数，g(x)表示买 x 个玩具的时候的最小消毒代价，对于所有不可行的方案可以令 f(x) = Inf。

要证明这是一个单峰函数，可以证明这个函数的导数是单调的，而证明导数是单调，只要证明其增量单调的，因为 x 为正整数，所以只需要证明 f(x+ 1) – f(x)是单调的。

$f(x + 2) – f(x + 1) = T_c + g(x + 2) – g(x + 1)$

$f(x + 1) – f(x) = T_c + g(x + 1) – g(x)$

$[f(x + 2) – f(x + 1)] – [f(x + 1) – f(x)] = [g(x + 2) – g(x + 1)] – [g(x + 1) – g(x)]$

对于 g 这个函数来说，从 x 增加到 x + 1 所减少消毒代价比从 x + 1 增加到 x + 2 所减少消毒代价要少，这是比较显然的。

所以最终可以发现 f(x)是一个单峰函数，那么就可以通过三分 x 来对问题求解。

现在就只需要知道如何求这个 g(x)，这个问题之前讲到可以通过贪心来实现，具体思想如下：

可以假设 $C_1 < C_2$，$N_1 > N_2$，这时候如果 $N_1 < N_2$ 是没有意义的。

因为玩具消毒的次数是固定的，所以肯定是每次尽量使用第一种消毒方式，然后考虑在第 i 天如果初始的玩具不够用了，那么它可以使用的玩具是$\sum_{j=1}^{i-N_1} T[j] - \Delta$，Δ 表示的是有一部分玩具已经被拿去消毒再次使用了。如果第一种消毒方式已经无法满足供应了，那就使用第二种消毒方式，如果两者加起来都没法满足那就表示这个方案不可行。

时间复杂度 $O(D * \log_{1.5} D)$

空间复杂度 O(D)

【实现细节】

要注意的是使用第二种消毒方式的时候要取上一次的使用时间时间尽可能后的。这个可以使用双端队列来实现，在做到第 i 个玩具的时候，将 $i – N_2$ 插在队尾，第一种消毒方式从头开始取，第二种消毒方式从尾开始取。

Tower of Hay

USACO 2009 Open

【题目大意】

奶牛们十分讨厌黑暗。为了换一个牛栏顶部的灯泡，Bessie 为了爬上去必须用一垛垛干草建造一个塔从而能够换灯泡。N 摞干草（标号 1 到 N）在传送带上有序的进入牛栏。第 i 摞干草有一个为整数的宽度 w_i；所有干草摞有一个单位的高度。

Bessie 必须用全部的 N 摞干草去建造那个塔而且必须按照它们到达的顺序放置它们。她可以按照她的愿望放置随意多摞干草在一行上。接下来她可以放置接下来到达的干草摞在上次排成的一行上面来建造新的一行（但不能比下面的行要宽）。直到所有的干草被用完。她必须按干草到达顺序来堆叠它们。更清楚地说：一旦她将一个草包放在第二级，她不能将接下来的草包放在地基上。

数据范围：$1 \le N \le 10^5$，$1 \le w_i \le 10^4$

时间限制：1s

【问题分析与程序设计】

直接 DP，用单调队列优化。

时间复杂度 O(n)

空间复杂度 O(n)

Cleaning Up

USACO 2009 Mar

【题目大意】

给出一个 n 个数的序列 $A_1..A_n$ ($1 \le A_i \le m$)，要把这个序列分成连续的若干段，每一段的代价是这一段中不同的数的个数的平方，最终代价是每一段的代价的和，求最小的最终代价。

数据范围：$1 \le n, m \le 40000$

时间限制：1s

【问题分析与程序设计】

算法：

设计状态 f[i]表示取 1..i 的最小代价，unique[i][j]表示 i..j 中不同的数的个数。

$$f[i] = \min_{1 \le j < i} f[j] + unique[i][j]^2$$

直接做是 n^2 的，但是可以发现 f[n]的最大值是 n，所以只要往前找$\sqrt[2]{n}$个不同的数即可。

时间复杂度 O($n\sqrt[2]{n}$)

空间复杂度 O(n)

【实现细节】

往前找$\sqrt[2]{n}$个来更新的时候要注意边界条件。

Triangle Counting

USACO 2010 Open

【题目大意】

给定 n 个整点，统计以这些整点为顶点所构成的三角形中有多少个包含原点(0, 0)

数据范围：$3 \le n \le 10^5$，不会有两点的直线经过原点，特别地，不会有点恰好是原点。

时间限制：1s

【问题分析与程序设计】

算法：

通过简单的画图可以发现，一个满足条件的三角形要满足任意选择一个顶点，另外两个点都不能在这个点和原点所在直线的同一侧。

有了这个约束之后就可以先极角排序，然后线性统计出当前点 i 和(0, 0)所在直线的某一侧一共有多少个点，这个只要用一个队列来统计就可以了。

然后就是对于如何统计每个点对于答案的贡献。

假设刚刚算出来的点数记录在 cnt_i 中，而当前做到第 i 个点的时候队列里的点按极角序依次为 $q_1..q_m$，那么点 i 对于答案的贡献就是$\sum_{j=1}^{m}(cnt_{q_j}-j)$

这个东西也是可以线性的做的，因为可以发现当一个新的点入队的时候，它的贡献就是 cnt – 当前队列长度，可以直接维护。

时间复杂度 $O(n * \log_2 n)$

空间复杂度 O(n)

StarCowCraft

USACO 2010 Mar

【题目大意】

给定 n 个如 $ax+by+cz \geq 0$ 或 $ax+by+cz \leq 0$ 的条件（a，b，c 已知，x，y，z 均为正实数）

同时 $100x > y$，$100x > z$

$100y > x$，$100y > z$

$100z > x$，$100z > y$

求 x，y，z

数据范围：$n \leq 300$

时间限制：4s

【问题分析与程序设计】

判断 $ax+by+cz \geq 0$（或 $ax+by+cz \leq 0$）是否有解

令 $x = x_0 z$，$y = y_0 z$

上述问题转化为判断 $ax_0 z + by_0 z + cz \geq 0$ 是否有解

等价于判断 $ax_0 + by_0 + c \geq 0$ 是否有解

如果 x_0，y_0 有解，则 x，y，z 有解

算法 1：

现在要解决的问题是对于 n 个 $ax_0 + by_0 + c \geq 0$ 或 $ax_0 + by_0 + c \leq 0$ 条件，判断 x_0，y_0 是否有解。

实际上对于每个限制条件的解都是一个半平面，而满足 n 个限制条件的解就是这 n 个半平面交出来的凸包。

一开始可以把解集设为一个矩形，每次用代表当前解集的凸包和 $ax_0 + by_0 + c = 0$ 这条直线求交点，删除不合法的部分，生成一个新的凸包。

时间复杂度 $O(n^2)$

空间复杂度 O(n)

期望得分：100

虽然这个算法已经可以获得满分，但是在面对更大的数据时，我们需要一些更有效的算法。

算法 2：

假设可以在 O(m+n)的时间内将 m 个半平面的交和 n 个半平面的交合并，则可以有一种 $O(n\log_2 n)$的分治算法求半平面的交。

将 n 个半平面分成两个大小近似相等的集合，在每个子问题中递归地计算半平面的交，最后合并两个凸多边形区域。

问题的关键是怎样在 O(m+n)的时间里求两个凸多边形的交。

- 将两个凸多边形沿顶点切割成至多 O(m+n)个平行于 y 轴的梯形区域。
- 每两个梯形区域的交可以在 O(1)时间内解决。

描述凸多边形的方法

- 凸多边形上方和下方的顶点分别构成一个 x 坐标递增序列。
- 将这两个序列中的顶点分别作为一个链表存储,得到确定凸多边形区域的上界和下界。

时间复杂度 $O(n\log_2 n)$

空间复杂度 $O(n)$

【实现细节】

因为有 $100x > y$，$100x > z$，$100y > x$，$100y > z$，$100z > x$，$100z > y$ 这几个条件

而 $x = x_0 z$，$y = y_0 z$，所以 $0 < x_0 < 100$，$0 < y_0 < 100$

初始的多边形可以设为(0，0)，(0，100)，(100，100)，(100，0)这个正方形，因为边界不可取，所以要 + - eps。

并且一开始用 $100x_0 - y_0 > 0$ 以及$-x_0 + 100y_0 > 0$ 这两个半平面先交一下

Cows in a Skyscrape

USACO 2012 Mar

【题目大意】

有关贝茜和她的朋友们的一个鲜为人知的事实是他们喜欢爬楼梯比赛。一个更多人知道的事实是奶牛非常讨厌下楼梯，所以在奶牛爬到了他们最喜欢的那座摩天大厦的顶部后，她们有了一个问题。奶牛拒绝使爬楼梯回去，她想用电梯回到 1 楼。

电梯有 W(1 <= W <= 100,000,000) 的磅的最大承重，第 i 只奶牛有 Ci (1 <= C_i <= W) 磅的体重。请你帮助贝茜算出最少坐多少次电梯才能使所有的奶牛回到第 1 层楼。每个电梯里的奶牛的总体重不能超过电梯的最大承重。

数据范围：$n \leq 18$

时间限制：1s

【问题分析与程序设计】

用莫比乌斯反演优化 DP。

时间复杂度 $O(n * 2^n)$

空间复杂度 $O(n * 2^n)$

First!

USACO 2012 Dec

【题目大意】

给出 n 个字符串，对于每一个字符串，如果重新定义字母的优先级，问这个字符串是否可以成为这 n 个串中字典序最小的。

数据范围：$n \leq 30000$，$1 \leq$ 字符串总长度 $m \leq 3 * 10^5$，只包含小写字母

时间限制：1s

【问题分析与程序设计】

算法：

建字典树，在遍历这棵树的时候，会找到一些字母之间的优先级的大小关系，在访问下一个节点的时候，下一个节点所代表的字母的优先级必须高于所以其他存在的字母，最后访问到这个串的时候再判断是否存在一组可行解。判断的方法就是，如果字母 a 的优先级需要高于字母 b，则建一条从 a 指向 b 的有向边，最后判这 26 个点的图中是否存在环，存在则无解，反正就是合法的。

时间复杂度 O(m * 26 * 26)

空间复杂度 O(m * 26)

Figure Eight

USACO 2012 Dec

【题目大意】

有一个 N*N 正方形网格（5 <= N <=300），其中字符'*'代表缺损部分，'.'表示完美无瑕的部分。现在要在上面画一个数字 8，需要满足以下几个条件：

*数字 8 由上下两个矩形构成。

*数字 8 的上下两个矩形都满足至少有一个单元格在矩形内部。

*数字 8 顶部的矩形的底边必须为底部矩形顶边的子集。

*数字 8 只能画在完美无瑕的部分。

*规定数字 8 的得分为上矩形和下矩形的面积的乘积，它们希望得分能达到最大。

*888..
.8.8**
.8*8..
.88888
8..8
.88888

这就是 N = 6 时的一个合法的数字 8，得分为 2 * 3 = 6

数据范围：5 <= N <=300

时间限制：1s

【问题分析与程序设计】

算法：

这道题最先想到的应该是枚举这个数字 8 中间的公共部分，然后是枚举上面那个矩阵的左右边界，这时的复杂度已经达到 N^3，就算是再乘上一个 $\log_2 N$ 的复杂度也会超时，所以剩下的那部分应该直接预处理得到答案。

接下来就是要考虑预处理的部分了，根据上面的枚举，需要预处理的部分有，当上矩阵的底边以及左右边界都确定的时候的最大面积，同样的还有当下矩阵的顶边以及左右边界都确定之后的最大面积，最后是当上矩阵的底边以及左右边界都确定之后下矩阵的最大面积。

首先是一开始的预处理，设 f[i][j][k][0]表示当上矩阵的左右边界在第 i 列和第 j 列，底边在第 k 行时的最大面积是多少，同样的 f[i][j][k][1]表示的是下矩阵。对于这个预处理先枚举左边界在第 i 列，再枚举右边界在第 j 列，最后枚举第 k 行，这个在做的时候只要记录最早或者最晚的可行边（就是在那一行上从左边界到右边界没有*），而当第 k 行第 i 列，或第 j 列出现*的时候就将其变成-1 即可。

最后是当上矩阵的底边以及左右边界都确定之后下矩阵的最大面积，设 mx[i][j][k]表示当上矩阵的左右边界在第 i 列和第 j 列，底边在第 k 行时下矩阵的最大面积是多少。

$$mx[i][j][k] = max(mx[i-1][j][k], max(f[i][j][k][1], mx[i][j+1][k]))$$

时间复杂度 $O(N^3)$

空间复杂度 $O(N^3)$

【实现细节】

本题首先要注意的是空间不要开爆，有的虽然状态有三方，但是实际开数组只要平方甚至是线性的。最后本题还存在 N^2 的做法，但是笔者没有想到，有兴趣的可以去探索一下。

Photo

USACO 2012 Dec

【题目大意】

有 N 头奶牛排成一排，标号为 1..N。小明拍了 M 张照片，照片 i 包含了从 a_i 到 b_i 的奶牛，每张照片中恰有一头奶牛有斑点。问至多有多少头奶牛有斑点。

数据范围：$N \leq 2 * 10^5$，$M \leq 10^5$

时间限制：1s

【问题分析与程序设计】

对于这道题，因为给出的限制条件是从 a_i 到 b_i 的奶牛恰好有一头奶牛有斑点，所以很容易可以知道决策区间。

设计 DP 状态 f[i]表示到第 i 个奶牛为止最多有多少头奶牛有斑点，决策的左边界是不包含这个点的所有在这个点左侧的区间的最右边界，决策的右边界是包含这个点的区间的最左边界。

随着 i 逐渐增大，整个决策的区间是单调往右移的，利用单调性可以用单调队列来维护当前决策点的最优值。

时间复杂度 O(N)

空间复杂度 O(N)

Google Code Jam

Min Perimeter

GCJ 2009 Final D

【题目大意】

给你一个整数坐标的点集，大小为 n，询问点集中最小的三角形周长是多少。退化的三角形也是允许的(面积为 0)。

数据范围：$n \leq 10^5$

时间限制：1s

【问题分析与程序设计】

这道题类似于最近点对的做法，直接分治。

时间复杂度 $O(n\log_2 n)$

空间复杂度 O(n)

【实现细节】

左右两边只要各扫 15 个点就可以了

Wi-fi Towers

GCJ 2009 Final D

【题目大意】

给出 n 个塔，每个塔有有一个收益（可以为负），塔之间存在依赖关系，求最大收益。

数据范围：$n \le 500$

时间限制：1s

【问题分析与程序设计】

算法：

这是一道经典的网络流题，就是求一个最大权闭合子图。从源点向所有收益为正的点连一条有向边，流量即为权值，再从所有收益为负的点连一条有向边到汇点，流量为权值的绝对值。若取 A 的前提是取 B，那么建一条从 A 指向 B 的流量为+Inf 的边。做最小割，最终答案为正收益的和减去最小割。

时间复杂度 $O(n^4)$

空间复杂度 $O(n^2)$

期望得分：100

【实现细节】

直接写 dinic 会超时，要加上当前弧优化。