Codeforces/USACO/GCJ 题目泛做解题报告

南京外国语学校 王悦同

1.【CF266 E More Queries to Array...】

你有一个包含 n(<=10^5)个整数 a[1],a[2],...,a[n]数组。你的任务是快速地执行以下两种操作。

1.将闭区间[l,r]中的元素赋值为 x。即此操作过后 a[l],a[l+1],...,a[r]的值都为 x。

2.计算并输出和$\sum_{i=l}^{r} a_i \cdot (i-l+1)^k$，此处 k 不超过 5。因为这个和可能很大，你要输出它模 1000000007(10^9+7)的结果。

解题报告：

这一类题目做法显然，用线段树维护，用组合数处理幂的计数。线段树每一段[l,r]存下$\sum_{i=l}^{r} a_i \cdot (i-l+1)^k$的和（这里 l 就是线段树这段的左端点）。不难发现，修改操作非常显然；真正询问的时候，也只是所有的$(i-l+1)^k$底数集体加上一个常数 C。那么利用二项式定理，我们再维护区间 k=1,2,3,4,5 时分别的 ai 和，然后直接用组合数计算，即可回答询问。

2.【CF 152 Div1 C Piglet’s Birthday】

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 ui 个架子，拿走 ki 个蜜罐，把这些蜜罐中的蜜吃掉，然后把这些蜜罐都放到第 vi 个架子上。当 Winnie 拿走蜜罐时，他会在第 ui 个架子上所有 ki 个蜜罐的集合中等概率选择一个集合，然后把集合中的 ki 个蜜罐拿走。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。N<=10^5,每个架子蜜罐数<=100.

解题报告：

简单动态规划。f[i][j]表示当前第 i 个柜子里还有 j 个罐子没被吃过的概率。由于拿走蜜罐是等概率的，所以只需要记录个数即可。据此转移，再通过概率直接计算期望，非常显然。

3. 【CF 158 Div2 E Dividing Kindom】

一个名叫 Flatland 的国家可以看成一个无穷的二维平面。Flatland 有 n（<=10^5）个城市，每个城市可以看做平面上的一个点。国王 Circle IV 统治着 Flatland。Circle IV 有 9 个孩子，他想让每个孩子分得 Flatland 的一部分。他想画 4 条不重合的直线，其中 2 条与 x 轴平行，另 2 条与 y 轴平行。于是，Flatland 被分成了 9 部分，每个孩子将得到这其中的一块。Circle IV 想了一会，决定第 i 个孩子分得的一部分 Flatland 中包含的城市的个数必须等于 ai。帮助 Circle IV 找到这样的 4 条直线，使得 Flatland 被这些直线分成 9 部分后，第 i 个孩子可以得到城市个数为 ai 的一部分。

解题报告：

做法非常暴力。直接枚举 9 个孩子分别分到了哪一块，然后可以据此直接划分出四条线（因为上面三块的 ai 和确定了，左、右、下也同理）。显然，如果没有两个格子 x 一样或者 y 一样，那么随便一种方案都

是可以的；现在我们要做的就是处理一样的情况。直接利用树套树、归并树、二维线段树等结构，查询每一块实际上有多少城市即可。

实现中，这样很容易超时。我们可以分布枚举：先枚举哪三个孩子在上面、哪三个在中间，这样就把横线画好并判断了；然后再独立的判断竖线。

4.【USACO 2008 Open Gold Cow Neighborhood】

了解奶牛们的人都知道，奶牛喜欢成群结队。观察约翰的 N(<=10^5)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 ｜x1 - x2｜+｜y1 - y2｜<= 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。

解题报告：

本题显然是并查集模型，但是关键是【邻居关系】可能是 N^2 级别的。先把(x,y)变成(x+y,x-y)，这样曼哈顿距离就变成了 max(|x1-x2|,|y1-y2|)。按 x 轴排序后扫描：扫描到点 i 时，平衡树中保留所有 x 坐标符合要求的（和 Xi 差在 C 以内的）点的 y 坐标；然后找到 i 的前驱后继，如果能和 i 合并，那就合并。这样做的正确性是，如果找到 i 的前驱 k，所有比 k 小的点，如果在平衡树中且可以和 i 合并，那么必然已经和 k 合并了。后继同理。因此并查集操作数是 O(N)的。用一个 set 可以代替平衡树，复杂度 O(NlogN)。

5. 【CF 332D Theft of blueprints】

给出一个 n（<=2000）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，恰好有一个点与 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。

解题报告：

这题十分无聊。主要就是一个结论：当 K>3 时，必须是一个完全图，且 K=N-1。K=1 时十分显然，K=2 时，因为题目保证了每两个点只有一个公共点，所以我们枚举这个公共点，就能把所有可能的集合都选出来了。这样的集合有 O(N^2)个，不会超时。

6．【CF 241D Numbers】

您有 1,2..n 的排列 a1,a2,……,an。您想删除一些整数,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.如果您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。

给您序列和 p,找到一种方法来满足上述条件。

1 ⩽ n , p ⩽ 50000，且 p 是质数。

解题报告：

显然，由于删除后不改变顺序，我们可以考虑使用动态规划。F[i][j][k]表示当前考虑了前 i 个数，异或和是 j，连起来 mod p 是 k，是否有可行的方案。每次我们只要枚举下一个数选不选就行了。时间复杂度是 O(N^2P)，显然不够。

但是注意到只要给出任意一种方案，而实际方案有 2^N 种，显然没必要考虑这么多方案。经过试验，只要保留 1~30 的数，就能保证出该出解的都出解。其实这题真正麻烦的地方在于输出方案——N=30 时，光是动态规划数组也是 100MB（而且要用 short 或 bool），那么无法记录前驱了。具体做法是，枚举前驱，看看前驱能否转移，因为我们只要构造方案，所以只要前驱是可以达到的状态，一定能顺着构造出解。这

样，我们不需要额外的数组，空间也就够了。

7. 【CF343E Pumping Stations】

疯狂的科学家 Mike 申请了一份工作。他的任务是管理一个抽水站系统。

这个系统包含 n 个抽水站，抽水站被标号为 1 到 n。连接两个抽水站的双向管道可以使水在任何一个方向流动（但同一时间水只能往一个方向流）。你知道每个管道的宽度——每小时可以流过的最大水量。每个抽水站可以通过这些管道把水从一些站台抽向其他站台，同时在一小时内总的进水量等于总的出水量。

Mike 的责任是用泵在这些站台之间输送水。遵循上述规则，可以在一小时内将一定数量的水通过管道（可能是通过其他站台）从站台 a 输送到站台 b。在这期间，其他站台的水不能流入 a，同时 b 站台的水也不能流出去。但是，任意多的水可以流出站台 a，或者是流进站台 b。如果一小时内有 x 升水流出站台 a，Mike 将获得额外的 x 元工资。

根据合同，Mike 需要工作 n-1 天才能得到薪水。在第一天，他选择两个站台 v_1 和 v_2，同时在一小时内，他将一定数量的水从 v_1 输送到 v_2。然后，在第 i 天，Mike 选择一个之前没有选过的站台 v_i+1，再将一定数量的水在一小内从 v_i 输送到 v_i+1。他在第 i 天输送的水的总量并不取决于他在第 i-1 天输送的水的总量。

Mike 需要挣得尽可能多的钱，帮助迈克找到这样的一个排列 v_1，v_2，……，v_n 使得麦克能够获得尽可能高的薪水。

解题报告：

这题需要一个重要算法模板，Stoer-Wagner 算法求无向图的最小割树。

这样一个最小割树可以满足如下性质：任意图中两点的最小割，等于对应的树上两点间的路径最小值。

如果仔细分析这个算法的过程会发现，由于它每次对集合做一次【分裂】，而一开始只有一个点，那么我们总可以把这个“最小割树”构成“最小割链”。这就是答案了。具体构建方法直接参照算法论文即可。

8. 【CF 251D Two Sets】

给出 N（<=10^5）个数，每个都<=10^18。你需要把它们划分成两个集合，使得每个集合异或起来之和尽可能大。

解题报告：

我们先把所有数异或起来，记为 X，两个集合异或值记为 X1，X2。如果 X 的某一位是 1，那么任何一种划分都是这一位上一个 0 一个 1，不会影响最终的和，所以这些位直接删除。接下来我们考虑的就是一些删过的数，显然此时 X=0。

从最高位开始贪心：如果我们希望 X1，X2 这一位都是 1，那么会不会和前面矛盾？如果不会就一定都放 1，如果会那就只能放 0。关键就在于矛盾的判断：给出前面若干位的值，能不能选出若干数满足异或起来是这个值。对 N 个数进行高斯消元，然后看哪些位置上有自由元即可。

整个复杂度应该是 O(NK^2)，K 是位数。

9. 【USACO 2005 December Gold Cow Patterns】

FJ 的 N 只牛中出现了 K 只爱惹麻烦的坏蛋。奶牛们按一定的顺序排队的时候，这些坏蛋总是会站在一起。为了找到这些坏蛋，FJ 让他的奶牛排好队进入牛棚，同时需要你的慧眼来识别坏蛋。

FJ 给每一只牛一个 1..S 的编号，虽然这不是一个完美的办法，但也能起一点作用。现在，FJ 已经不记得坏蛋们的具体号码，但是凭他的记忆，他给出一个编号在 1..S 的“模式串”。原坏蛋的号码如果相同，模式串中他们的号码依然相同。模式串中坏蛋们之间的号码的大小关系也与原号码相同。比如，对于这样一个模式串：

1,4,4,3,2,1

原来的 6 只坏蛋，排在最前面的与排在最后的号码相同（尽管不一定是 1），而且他们的号码在团伙中

是最小的。第 2,3 位置的坏蛋，他们的号码也相同（不一定是 4），而且是坏蛋团伙中最大的。

现在所有奶牛排成队列，号码依次是这样：

5,6,2,10,10,7,3,2,9

存在子串 2,10,10,7,3,2，满足模式串的相同关系和大小关系，所以这就是坏蛋团伙。

请找出 K 个坏蛋的团伙的所有可能性。

解题报告：

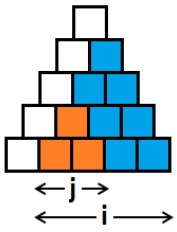
这是一个另类的 KMP，CEOI2010 有几乎一样的题。不过这题我用了一个相对偷懒的方法：因为离散化后要求一样，所以相邻两项的大小关系至少得匹配。我们把相邻项的大小关系设为 0 或 1，以此 KMP，对于合法的再进行检查就可以了。当然这个做法是可能被卡掉的；一个优化是：因为如果要匹配，任意一对对应位置都必须相对大小一样，所以一旦通过了【初审】，我们就随机抽查 1000 对。这样出错率就很小了。

10. 【CF354D Transferring Pyramid】

你有一个 n 行的金字塔，有 K 个格子需要被修改，你要设法修改它们，使得这 K 个格子中的每一个都至少被一个修改操作覆盖。(一个格子可以被修改多次，可以修改不需要修改的位置)

两种修改操作：第一种是修改一个位置，第二种是修改某个点向下一直到底部的金字塔区域。每种操作代价都是修改的格子数+2。问最少代价。

解题报告：



左图非常直观的状态表示（虽然非常新颖）。F[i][j]表示蓝色区域全被覆盖的最少代价。

Dp[i][j]要么是 dp[i][j-1]直接转移，要么是 dp[i-1][j-1]补上最上面的几个单独格子；

Dp[i][0]则是枚举最后一次放的三角形边长 k，显然这里三角形的顶点要落在从右往左第 i 个斜列上。另外 k 是 sqrt(K)级别，因为我们总能一个一个修改而不用一下改这么大。

所以整个算法复杂度是 O(Nsqrt(K))的。

11. 【CF 301E Yaroslav and Arrangements】

Yaroslav 称数列 a1，a2，… ，ar 为良好的，当它满足：

1.| a1 － a2 | = 1，| a2 － a3 | = 1，…，| ar-1 － ar | = 1，| ar － a1 |= 1

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

Yaroslav 称数列 b1，b2，… ，br 为优秀的，当它满足：

1. 数列中的元素不下降
2. 满足 1 ≤ r ≤ n 、 1 ≤ bi ≤ 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 (109+7)

两个数列{xn}、{yn}被认为不同，仅当存在一个位置 i，使 xi≠yi

n，m，k≤100

解题报告：

我们可以发现，如果有两个相邻的一样的数 X，那么以后一定要再他们中间加上至少一个 X+1。我们按从小到大顺序加入数。F[i][j][k][s]表示目前放了 i 个数，j 种方案，下一个放 k，有 s 个位置需要放 k。每次转移，我们直接枚举 k 放了多少个（至少 s 个，多出来的就是以后要放 k+1 的），然后用组合数更新答案。复杂度为 O(N^5)。不过注意到 j<=100，很容易就爆了，所以实际有效转移非常少。

12. 【CF 268D Wall Bars】

Manao 在一家建筑公司工作。最近，有一项在儿童公园内建设 Wall Bar 的订单。Manao 被委托设计一个能使公司最节省资金的建设方案。

在回顾了 Wall Bar 的正常规格之后，Manao 发现许多要求是模糊的，决定按照公司的利益处理。他的最终设计描述如下：

1、我们定义单位长度，建筑物中间那根管子高度为 n。

2、在高度为 1、2...n 的地方，恰好有一根水平的横杆从中间的杆子连向四个方向中的某一个预先固定好的杆子上。

3、如果两根横杆的距离不超过 h，且方向相同，那么一个孩子可以从一个一根横杆爬到另一根上。在地上的孩子，可以爬到任何一根高度在 1-h 之间的横杠上。在 Manao 的建筑物上，一个从地面出发的孩子至少能到达一根高度在 n-h+1、n-h+2...n 的横杠。

Manao 想知道有多少种设计方案满足上述要求。这个数字可能会非常大，输出其除以 1000000009 的余数即可。两个设计方案被认为是不同的，当且仅当在某个高度他们的横杠方向不同。

1≤n≤1000, 1≤h≤min(n,30)

解题报告：

这题因为 h<=30，所以可以直接动态规划。F[i][j][k][s][c]表示当前在第 i 层，四个方向距离上一个分别是 j,k,s,c。这样会超时，然而注意到，j,k,s,c 中总会有一个是 0（刚刚放的），所以 c 的范围其实是 1~4，代表是哪一个方向为 0，其他三个按顺序存。

转移会有一点麻烦，但常数还是可以接受的。复杂度为 O(Nh^3)。

13. 【CF 264E Roadside Trees】

Liss 是一只喜欢吃坚果的松鼠。它要求你种一些坚果树。

在一条直线有 n 个位置可以种树，自西向东标号 1~n。每个月，每棵树都会长高 1 米。每个月初，你都会收到一个要求。要求有两种类型：

1、 在位置 p 种一棵高度为 h 的树

2、 砍掉从西向东数的第 x 棵树，当这个位置的树被砍掉后，倒下的树会占据这个位置，之后这个位置不能再种树。

在做完这个要求后，你需要求出当前树高的最长上升子序列。一个树的序列，当从西向东的每棵树都严格高于序列中位于它西边的树。序列的长度就是序列中树的个数。

特别的，Liss 不喜欢高度相同的树，所以任何时刻树的高度是不同的。初始没有任何树。

N<=10^5，要求数<=2*10^5，新种的树高度<=10，砍树的 x<=10.

解题报告：

种树的时候，在 t 时间种高度为 h 的树，视为种了 h-t 高度的树，这样就可以忽略树的生长。

然后将每一棵树对应到一个点(x,y)，x=位置，y=高度。每一时刻，我们维护平面上每个点为结尾的 LIS（按从上到下的顺序）。插入操作是一个 y 坐标比较小的点，把所有比它 y 小的点删掉重新计算（最多 10 个）。删除操作是一个 x 比较小的点，把所有比它 x 小的点删掉重新计算（最多 10 个）。最后发现计算 LIS 时 x、y 坐标互不影响，因此分别用一棵线段树维护即可。O(NlogN*10)

14. 【Usaco2009 Open Tower of Hay】

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。

解题报告：

一个重要结论：将塔看作一个面积一定图形，要使其最高，必须最瘦。

f[i]表示 i..n 最后一层的最小的宽度。f[i] = min{sum[j - 1] – sum[i - 1] | sum[j - 1] – sum[i - 1] >= f[j]}

这个动态规划可以用单调队列维护 j，因为 sum 是单调的。复杂度 O(N)。

15. 【CF 269D Maximum Waterfall】

Emuskald 被雇佣建造一座符合最近园林建筑学趋势的人造瀑布。一个现代的人造瀑布包括许多个嵌在墙壁中的水平板，水从一块水平板流向另一块，直到它从墙的顶端流到底端为止。

墙的高度是 t，有 n 块水平板嵌在上面。每块水平板可以看作一条水平的线段，高度为 hi，从 li 到 ri。第 i 块水平板连接的平面上的(li,hi)与(ri,hi)两点。墙的顶端可以看作一块在(-10^9,t)到(10^9,t)间的水平板。相似的，墙的底端可以看作一块在(-10^9,0)到(10^9,0)间的水平板。不会有两块水平板之间有公共点。

Emuskald 知道，为了瀑布看上去漂亮，水可以从第 i 块板流向第 j 块板当且仅当：

1. max(li,lj)<min(ri,rj)（二者的水平位置存在相交部分）

2. hj<hi（第 j 块板在第 i 块板下方）

3. 不存在 k(hj<hk<hi)，且(i,k)与(k,j)均满足以上两个条件。

从 i 到 j 的流量为 min(ri,rj)-max(li,lj)，相当于二者的水平相交部分的长度。

Emuskald 决定在他的瀑布中，水会沿着一条单向的路径从顶部流到底部。如果水流到了一块水平板上（除了墙的底部），水会流向恰好一个更低的水平板。整个瀑布的水流量被定义为水流路径上每连续的两块水平板间的水平相交部分长度的最小值。

形式化地：

1. 瀑布由一条单向路径 顶部->p1->p2->…->底部 组成。

2. 瀑布的流量是路径 顶部->p1->p2->…->底部 中相邻两块水平板间的最小流量。

请帮助 Emuskald 找到可能的最大水流。

解题报告：

构建一个拓扑图。如果一块板子能流到另一块，则连边。由于若 a1->a2,a2->a3…an-1->an 有边，则 a1->an 不能有边，所以边数 O(n)。

对于每个板子，要快速找到能流到哪些板子，可以自底向上添加板子，用线段树维护当前露出来的部分。O(NlogN)。构建出图以后直接按顺序动态规划即可。

16．【CF 309B Context Advertising】

给出一段 N（<=10^6）个单词的文章，长度和<=5*10^6。你需要把它选出连续的一段，分成 R 行写下来，每行字符<=C 个。相邻单词需要用空格隔开。问如何才能选出尽可能多的单词？

解题报告：

很明显，如果我们知道从哪儿开始选，则只要一直向后扫描，能放则放即可。关键是要快速确定：对于每个起点，分别能选多少个词。

我们考虑如果这次枚举到了起点 i，则我们可以在 logN 的时间内（二分+部分和）得到这一行能写到哪个单词。设下一行从单词 j 开始，则以后我们不必枚举单词 j 了，应该在这一次一起处理。因此我们枚举起点后，按贪心的原则把所有行都写下去，然后记录所有的起点 A1,A2,… Ak. 不难发现，如果有 K 个起点，

我们的复杂度是 KlogN，而这 K 个起点的答案都已经得到了。所以这样枚举复杂度是 O(NlogN)。

17.【CF 238E Meeting Her】

Urpal 住在一个大城市。他已经计划今晚去见他爱的人。

这个城市有 n 个交叉点编号从 1 到 n。交叉点被 m 条有向的街道连接，所有的道路有相等的长度。Urapl 住在交叉点 a，约会被安排在一个在交叉点 b 的餐馆中。他想乘坐公共交通去交叉点 b。这里有 k 个公交车公司。在每一秒的开始，一个公交车从第 i 个公司随机选择一条从交叉点 si 到交叉点 ti 的最短路径然后通过这条路径。可能没有从交叉点 si 到交叉点 ti 的路径。在这种情况下没有公交车会离开 si 去 ti。如果一个公交车通过 Urpal 所在的交叉点，则 Urpal 可以上这辆公交车。他可以在中途任意一个交叉点下车。

现在 Urpal 想知道是否有可能乘坐公共交通在有限的时间（旅途的时间为每条经过的边的长度和）内到达约会地点，和最坏情况下他需要乘坐公交车的次数。

在任何时刻 Urpal 只知道他自己的位置和约会地点。当他上了公交车时他只知道这辆公交车属于第几个公司。当然 Urpal 知道城市地图和每个公司的(si,ti)。

注意 Urpal 不知道公交车的速度。N，K<=100

解题报告：

注意到车只有在割点处才能保证上车。然后就是一个类似动态规划了：F[i][j]表示在 i 点坐着车 j，类似 spfa 更新。

需要注意有些状态，i 点不可能在 j 车上。应当是 O(N^4)的复杂度。

18.【CF 273E Dima and Game】

Dima 在纸上写下 n 对整数(l[i],r[i])(1<=l[i]<r[i]<=p)。然后玩家轮流进行操作。轮到自己时，可以进行下面的操作：

1.选择第 i 对数(1<=i<=n)，满足 r[i]-l[i]>2;

2.将第 i 对数替换为(l[i]+floor((r[i]-l[i])/3) , l[i]+2*floor((r[i]-l[i])/3))或者(l[i] , r[i]-floor((r[i]-l[i])/3))。floor(x)表示向下取整。

不能进行操作的玩家则输。

当然，Dima 希望先进行操作的 Anya 赢得游戏。所以 Dima 需要写下这样的 n 对整数(l[i],r[i])(1<=l[i]<r[i]<=p)，使得如果两个玩家都采取最优策略，先操作的玩家取得胜利。请计算 Dima 有多少种这样的方法。输出方案数模 1000000007(10^9+7)后的值。

如果这些数对被按照不同的顺序写了下来，则将其视作不同的方法。

N<=1000, P<=10^9

解题报告：

明显是一个 SG 函数的组合博弈。每一对数的意义仅仅在于两数只差，每个差对应了一个 SG 值。在本机将 1~10^9 都预存出来后惊奇的发现只有 102 段不同的……

于是果断打表。然后就变成了选 N 个数异或为 0。又因为观察发现每个数只有 0、1、2，所以直接动态规划，O(N)的复杂度。

19.【CF 264D Colorful Stones】

现有两串彩色的石头。每块石头的颜色是红、绿、蓝之一。给你两个字符串 s 和 t。s 的第 i 个（从 1 开始标号）字符代表第一串的第 i 块石头的颜色。类似地，t 的第 i 个（从 1 开始标号）字符代表第二串的第 i 块石头的颜色。每个字符是'R','G','B'时，对应的石头分别为红色、绿色和蓝色。

初始时 Squirrel Liss 站在第一串的第一个石头上，Cat Vasya 站在第二串的第一个石头上。你可以发出下面的指令（0 次或多次）。

每个指令是以下三种中的一种：“红色”、“绿色”、“蓝色”。在发出指令 c 后，站在颜色为 c 的石

头上的动物将会移动到后面的一块石头上。例如，如果你发出“红色”的指令，站在红色石头上的动物将移动至后一块石头上。你不能发出能使某动物走到石头串之外的指令。换句话说，如果某动物站在最后一块石头上，你不能发出那块石头颜色对应的指令。

(Liss 的位置,Vasya 的位置)被称为状态。如果从初始状态(1,1)能够通过发出 0 次或多次指令达到一个状态，我们就称这个状态为“可达”的。请计算出不同的可达状态的数目。

石头数 N<=10^6.

解题报告：

一个很容易猜的结论是：对于第一个行到 i 的情况，第二行能到达的是连续的一段？如果这样的话，一个双指针就能在 O(N)时间解决问题。

不幸的是这个结论有反例。但我继续观察后发现可以稍加修改：

如果对于第一行的 i，第二行连续的一段[l,r]可行的话，那么[l,r]中所有 j 满足：a[i-1]=b[j],a[i]=b[j-1]的点，都是不可行的。其他都可行。

这个完全是打表找出规律。时间复杂度 O(N)。

20. 【CF 261E Maxim and Calculator】

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

1.假设第一个单元的数字为 a，第二个单元的数字为 b，那么将第二个单元的数字改成 b+1。

2.假设第一个单元的数字为 a，第二个单元的数字为 b，那么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<=x<=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。

L,r<=10^9, p<=100

解题报告：

关键就是这个 p<=100。我们注意到，这其实就是把一些数乘了起来，而每一个数都<=100。也就是说，只可能生成那些所有质因子都不超过 100 的数。

10^9 以内有多少这样的数呢？打一个表就会发现是 3000000 个左右，还是不多的。我们基于这些数进行动态规划：f[i][j]表示用不超过 i 的质数拼出 j 的最少乘法次数。每一次我们枚举是+1 还是乘一下，O(1)转移，整个复杂度是 O(p*3000000)。

21. 【USACO Dec10 Gold Threatening letter】

FJ 有无数份的最近一期的《哞约时报》，每一份时报都有 N 个大写字母排成一行。他想组合成一封信也是一个写在同一行的由 M 个大写字母组成的长字符串。

FJ 很懒，所以他希望剪切的次数越少越好。FJ 的剪刀非常碉堡，可以在一次剪切中，从《哞约时报》上切下任何一个片段。求：FJ 至少进行几次剪切才能构造出那个长度为 M 的字符串。

保证有解。N,M<=50000

解题报告：

这题很明显是一个贪心。每次我们肯定是找到一个尽可能长的前缀在原串中出现过。我们构建后缀数组然后维护一个“有效区间”，每次试着多加入一个字符，然后二分维护出新的区间，这就可以找出一个尽可能长的前缀了。每一个字符复杂度是 O(logN)所以总复杂度是 O(MlogN)。

22.【CF 238D Tape Programming】

有一种编程语言，在这个语言下一个程序是由数字和”<”,”>”构成的非空串。程序运行时有一个指针。最开始指针的指向最左字符，移动方向为向右。

我们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对应得改为向左或向右（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，程序就结束运行。很明显，这个语言的程序总是会在若干步以后终止。

现在我们有一个由 n 个由"<",">"和数字构成的串，你需要回答 q 个询问。每个询问会给你两个数 l,r，如果把 sl,sl+1,..., sr 看出一个单独的程序，问你每个数字会被输出多少次。

N,q<=100000

解题报告：

解决这题只要观察到一点：我们首先执行原程序（如果执行完没有删完，再执行一遍）。此后每个询问[l,r]，都相当于是原来的这个执行过程中，第一次进入[l,r]区间开始，到第一次出[l,r]区间为止，所有输出的内容。因此原问题主要就是模拟这个程序。整个问题都可以在 O(NlogN)完成模拟、查询。

23.【CF 277D Google Code Jam】

GCJ 规则：在每场比赛中，参赛者有几道题目可以做，每道题目有两个部分:简单(small input)和困难(large input)，有一个对应分数。在你做出简单部分之前是不允许做困难部分的。除此之外，没有其他做题顺序的限制。参赛者能够立刻知道自己的简单部分的解的提交的结果，但是困难部分的解的正确与否只能在比赛结束公布。最后排名以得分从高到底排序，得分相同按罚时从低到高排序。GCJ 的罚时指的是最后一次提交正确解的时间。

Vasya 打算去做 GCJ，他可以在比赛开始后快速（忽略所有读题时间）读题。然后对于 n 道题目，他知道以下 5 个值。

1、scoreSmall_i 表示的是做出简单部分之后得到的分数。scoreLarge_i 指的是做出困难部分后得到的另外分数。也就是说在第 i 题你的最大得分为 scoreSmall_i+scoreLarge_i。

2、 timeSmall_i 表示的是做第 i 题的简单部分所花的时间。timeLarge_i 表示从简单解法，优化到困难的解法的所需额外时间。

3、简单部分一定 1A。然而对于困难部分，他有 probFail_i 的概率 FST(fail system test)。并且值得注意的是，如果 FST，那么就不会得到困难部分的分数，并且这次提交也不算一个正确解的提交。 比赛总共有 t 时间。读题和提交题的时间忽略不计。并且值得注意的是：Vasya 不允许在一道题目的其中一个部分中提交两次解。

Vasya 想要正确安排时间和解题顺序，然后使得得到的期望分数尽可能高，同时保证期望罚时尽可能少。

N<=1000,T<=1560

解题报告：

先考虑选出的要做 Large 的题如何排序。利用相邻项交换的思想可以得出按 TimeLarge[i]*Fail[i]/(1-Fail[i])排序。在此贪心的顺序上进行动态规划（也就是：第一步贪心决定，如果选了一些题该按什么顺序做，第二步动态规划决定选哪些题）。

dp[i][j]表示前 i 题花了 j 时间的期望最大得分和在此情况下的期望最小罚时。枚举下一题做多少分（3 种情况），3 种转移的罚时都可以根据之前的罚时和目前所用总时间推出来。时间复杂度 O(NT)。

24.【CF 305E Playing With String】

两人正在玩一个游戏。轮流行动，不能操作的人输。

游戏开始前裁判会在方格纸上写下一个字符串，每个格子包含一个字母。

一个人的操作分这么几步：

1.这个人选择一张纸，我们称上面写着的字符串为 t。注意一开始时只有一张可选纸。

2.这个人选择一个i(1<=i<=|t|)使得存在一个正整数k(0<i-k,i+k<=|t|)满足t[i-1]=t[i+1] and t[i-2]=t[i+2] ... t[i-k]=t[i+k]

3.这个人以把这张纸的第i个字母的两侧撕开使得这张纸分成3份:t[1~i-1],t[i~i],t[i+1,|t|]

两人都使用最优策略，你需要确定胜者是谁。假如先行动的人胜，那么你还需要输出可以帮助这个人获胜的第一步行动选择的位置。如果有多个输出最小的那个。

字符串长度<=5000.

解题报告：

对于那些可以断开的位置记为1其余为0，一个位置可以断开当且仅当它两边字符相同。这样连续的一段段01之间彼此其实是独立的，并且只和长度有关，所以预存给定长度的sg函数值，枚举第一个走哪儿就行了。复杂度是O(len^2),len<=5000。本题关键在于意识到断开后并不影响“哪些点可以断开”。

25.【USACO March2008 Land Acquisition】

Farmer John要买N（1≤N≤50000）块长方形的土地，每块土地长为Li，宽为Wi（Li,Wi均为正整数且≤1000000）。

购买单独一块土地的价格是$1/平方，但是购买多块土地有优惠。如果一次性购买几块土地，总价钱就是这几块土地的max{Li}*max{Wi}。

Farmer John想通过分次购买，花最少的钱把所有土地都买下来，问他最少要花多少钱。

解题报告：

先将无用的土地去掉（就是存在一个土地比它长又比它宽）。然后可以设计一个O(N^2)的动态规划，因为我们购买土地肯定是连续的一段。

f[i] = min(f[j] + x[j + 1] * y[i]) (j < i)

将这个式子展开后发现可以直接套用斜率优化的模型，且斜率单调可以用队列维护，所以成为了经典1D1D动态规划优化，这里不再赘述。

26. 【CF 323B Tournament Graph】

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≠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）。

N<=1000

解题报告：

既然是构造，只要尝试出一种构造方案即可。一种可行方法如下：

先1->N首尾相连连成环，然后若i和j奇偶性一样，连i到j，否则j到i。

因为对于任意u,v，如果u和v奇偶性一样，一步到达，否则u先走到u+1再到v；特例是u=N=奇数的情况，这时我们可以u到v-1再到v。必然可行。

27. 【CF 351D Jeff and Removing Periods】

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

第一步.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];

第二步.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]。

第三步.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jeff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")。

请你帮 jeff 来回答这些询问。

M,q<=100000

解题报告：

对于每个询问，设区间中有 S 个不同的数，则答案最多是 S+1。因为重新排列后，每次一定能删光一种（按顺序排序即可）。答案最少是 S，因为每次最多删光一种。关键就是是不是 S 次就够了呢？这取决于第一次执行第一步时能不能删光一种，也就是区间内有没有一种数下标恰好构成等差数列。

可以用离线莫队算法。我们对每种数维护所有出现的位置。每当询问区间移动 1，我们插入或者删除一个数，并维护当前有多少个位置不符合等差数列。这可以 O(N^1.5)完成。

28. 【USACO 2013 Open Photo】

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。

N<=100000

解题报告：

倒过来动态规划。f[i]表示 i 放了一个后后面最多能放多少个，考虑下一个放哪儿。显然所有包含 i 的区间不能放；i 后方的区间必须不能被这个漏掉。可以用单调队列优化，复杂度 O(N)。

29. 【CF240F Torcoder】

给出一个长度为 N 的字符串，以及 M 个操作。每个操作给出一个区间[l,r]，你需要将[l,r]中的字符重排，使得这个子串成为一个回文串——如果有多种方案，你要使得字典序最小。每个操作结束后都不撤销，也就是将字符串依次进行 M 次变换。如果某个操作不可能排出回文串，则直接无视这个操作。

你需要输出经过 M 次操作后的字符串。

N,M<=100000

解题报告：

详见我的集训队第一次作业。大概做法就是将 26 个字母分开来维护，每次判断回文串时就是统计每个字母分别有多少个，而每次构建的新回文串肯定不会超过 51 段不同字符，每段用线段树区间赋值即可。

30. 【CF294D Shaass and Painter Robot】

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。

N,M<=100000

解题报告：

一个结论：每当所有边界格子都被访问过时，其它内部的格子也都被访问过了。证明参见我的集训队第一次作业。每次我们模拟机器人不是走一步，而是走到边界为止，故最多 O(M+N)步。用一个 Hash 表判断是否出现了循环即可。复杂度 O(M+N)。

31.【CF335D Rectangle And Square】

你有 N 个矩形（编号从 1 到 N）。所有矩形的四个角都是整点，并且两组对边分别平行于 X 和 Y 两坐标轴。不同的矩形可能接触，但是不会重叠。（换言之，不存在一个点严格处在多个矩形的内部）

现在你的任务是：判断是否存在一个构成正方形的子集。换言之，是否存在一个矩形的子集和一个正

方形，使得：

1. 每个处在正方形边缘或内部的点都存在于至少一个该子集内的矩形的边缘或内部；

2. 每个处在该子集内某一矩形的边缘或内部的点都处在那个正方形的边缘或内部。

N≤10^5。

0≤x1＜x2≤3000，0≤y1＜y2≤3000。

矩形不重叠。

解题报告：

本题就是枚举。首先，本质不同的可能只有左上角（N 种）*边长（L 种，L<=3000），也就是 3*10^8，可以接受。关键是怎么样快速判断合法。首先内部必须填满：这个可以直接用二维部分和完成。其次正方形边界不能“伸出来”，也就是要恰好是一个正方形，这个也不难，只要对于边框也做一个部分和，就能快速判断了。整个复杂度是 O(NL)。

32. 【CF 253E Printer】

我们考虑一个有这样功能的网络打印机：他从时刻 0 开始工作，每一秒钟他打印一页纸。某些时刻他会收到一些打印任务。我们知道打印机会收到 n 个任务，我们将它们分别编号为连续的整数 1~n，并且第 i 个任务用三个参数描述：t[i]表示接到的时间，s[i]表示任务要求你打印多少张，以及 p[i]表示任务的一个优先级。所有任务的优先级互不相同。

当一个打印机收到一个任务时，任务会进入一个队列并留下直到完成了这个任务为止。在任务队列非空时，每个时刻，打印机会选择队列里优先级最高的一个任务，打印一页。你可以想象任务进入队列是瞬间的事情，所以他可以在收到某个任务的时刻去执行这个任务。

你会得到除了某个任务以外所有任务的信息：你不知道某个任务的优先级是多少。然而，我们还额外的知道这个任务他完成时的时刻。我们给你这些信息，请求出这个未知的优先级并对每个任务输出它完成时的时刻。

N<=50000

解题报告：

由于注意到一个任务优先级越高，他一定越早完成（至少不会越晚），所以可以二分这个优先级然后模拟即可。

模拟中要注意：任何时刻我们一定都是处理优先级最高的，除非加入了新任务或者完成了一个任务。我们可以用堆维护所有优先级，O(NlogN)进行一次模拟。加上二分，复杂度为 O(Nlog^2N)。

33. 【CF325D Reclaimation】

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，问最后有多少次操作是成功的。

r,c<=3000，n<=300000。

解题报告：

如果取出所有填的格子，使得存在一个格子能通过八连通（第一列和最后一列也相邻）回到自己，就将上下隔开了。将原矩阵复制一遍后，并查集维护，每次维护所有连通的格子的集合。任何时刻要保证每个格子都不能和复制的一份中对应格子八连通，否则就是不满足要求了。复杂度为 O(NlogN)。

34. 【CF243D Cubes】

一天，Petya 的妈妈送了他一套木质立方体积木。Petya 很快地用了这套积木建了一座城市。

这座城市是建在一个 n*n 的正方形上的，正方形被分成了 n*n 个单位正方形。正方形的边和坐标轴平行，大正方形的一组对角落在(0,0)和(n,n)。在每个单位正方形上，Petya 用立方体建了一座积木塔。立方体

的边长是一单位长度。

之后，Petya 到了一个几乎是无穷远的地方，以向量 v=(vx,vy,0)的方向观望他的杰作。Petya 好奇，他从这个角度可以看到多少不同的立方体。帮他找到这个数字。

每个立方体包含他的边界。我们认为这个立方体是可见的，当且仅当立方体上存在一个点 p，从 p 以向量-v 发出射线上不存在属于其他立方体的点。

1≤n≤10^3,|vx|,|vy|<=10^4,|vx|+|vy|>0。0≤aij≤10^9,1≤i,j≤n

解题报告：

对于每个位置(x,y)的“柱子”，我们只要找到最低能看到哪个。然后根据我们视角的斜率，可以把这些 N*N 个格子投影到 x 轴上，有些格子可以互相遮挡。按距离观察者从近到远的顺序排序，对于每一个“条”（斜率映射的条），维护一个线段树，每次要查询最大值（查询最低看到那个）和单位修改（挡住了后面的）。整个复杂度为 O(N^2logN)。

35. 【CF 261D Maxim and Increasing Subsequence】

Maxim 喜欢数列，尤其是严格递增的那些。他想要知道数列 a 的最长上升子序列。

数列 a 给定如下：

1、数列的长度为 n*t;

2、ai=b((i-1) mod n)+1 (1≤i≤n*t)，其中运算 x mod y 表示 x 除以 y 所得的余数。

长度为 r 的数列 s1，s2，…，sr 被称为数列 a1，a2，…，an 的子序列，当且仅当存在递增的下标 i1，i2，…ir (1≤i1<i2<…<ir≤n)使得 aij=sj。也就是说，子序列是将原序列去掉一些元素得到的序列。

Maxim 有 k 个不同的数列 a。你要帮助 Maxim 求出每个数列的最长上升子序列。

1≤k≤10，1≤n,maxb≤105，1≤t≤109，n*maxb≤2*107

解题报告：

非常无聊的一题……注意到最后序列长度不大，可以直接 dp。用单调队列的做法优化 LIS，可以获得一个 O(NlogANS)的算法，就正好卡到时间内了。

36.【USACO Open10 Triangle Counting】

给出 n 个笛卡尔坐标系上的整点，统计有多少三角形包含原点(0,0)。N<=100000

解题报告：

我们这么统计：对于每个三角形，总是恰好有一个点，使得我们固定这个点时，其他两个点较之原点都在其右侧。因此我们枚举每个点作为固定点，然后找出有多少个点此时在原点右侧即可。O(NlogN)。

37. 【CF 256D Liars and Serge】

有 n 个人坐在一张长桌边上。对于每个人，我们知道他总是说真话或是说谎。

小塞尔吉问他们：你们中有几位总是说真话呢？桌子上的每个人都知道关于桌子上的所有人的所有事情（某个人说真话或是说谎）。诚实的人总是回答正确的答案，而说谎的人会回答 1 到 n 之间除了正确答案以外的任意一个整数。每个说谎的人都会选择他自己的答案，并不会考虑其他说谎的人的答案，所以两个说谎的人可能会给出不一样的答案。塞尔吉除了知道这些人对于他的问题所给的答案之外，并不知道其他的信息。他拿来一张纸并写下 n 个整数 a1,a2,...,an，其中 ai 表示第 i 个人的答案。得到这个序列后，塞尔吉断定，桌上至少 k 个人明显说谎了。塞尔吉想，有多少种不同的答案（长度为 n 的答案序列 a）可以得出桌上恰好 k 个人明显说谎了。由于可能有相当多种符合描述的答案，统计答案对 777777777 取模后的值即可。

1<=k<=n<=256，保证 n 是 2 的整数次幂。

解题报告：

注意到关键的一点：如果说 k 的人没有 k 个，那么说 k 的人都在撒谎 。然后动态规划 dp[i][j][k]表示现在已经处理了 i 个人，后面的人说有 j 个人撒谎，已经确定有 k 个人撒谎。每次枚举只要枚举说 j 的有几个

人就行了：如果有 j 个，他们可能没撒谎，不然一定撒谎了。整个动态规划是 O(N^4)的。然后因为 N 只有 8 种，所以直接打表……

38. 【CF258D Little Elephant and Broken Sorting】

小象很喜欢 1~n 的排列,但是最喜欢给他们排序。为了给这个排列排序，小象要重复交换一些数，好让最后得到一个 1,2,3,4…,n 的排列。

这一次，小象有一个 p1,p2,p3…pn 的排列。小象会进行恰好 m 次操作，第 i 次操作会交换第 ai 个数和第 bi 个数。但小象的程序出问题了，于是每次操作是会有一半几率不操作。

请帮助小象求出在操作完后的排列中的逆序对的数量的期望个数。

对于一个 p1,p2,p3…pn 的排列中的一对整数 i,j (1≤i<j≤n)，如果满足 pi>pj，我们便把这对数叫做一对逆序对。

N<=1000

解题报告：

独立考虑每对位置成为逆序对的概率。

f[i][j]表示 a[i]>a[j]的概率，一开始 f[i][j]=0 或 1（根据 a[i]和 a[j]的大小关系确定）

每次交换 a[x],a[y]，首先 f[x][y]=f[y][x]=0.5（因为可能换也可能不换..）

然后对于每个 i(≠x,y) ,f[i][x]=f[i][y]=(原来的(f[i][x]+f[i][y])/2)，因为一旦交换，换和不换是等可能性的，所以 x 和 y 的地位是相同的。

然后对于刚刚的 f[i][x]=f[i][y]，还有 f[x][i]=1-f[i][x],f[y][i]=1-f[i][y]。就可以了.

最后对于 i<j，Σf[i][j]就是答案了。O(N^2)

39. 【USACO Mar13 Hill Walk】

这里有 n 座山(1 < = N < = 100000)。每座山用一条从(x1,y1)到(x2,y2)的线段来描述，x1 < x2，y1 <y2。线段不会相交或接触,包括他们的端点。此外,第一个山满足(x1,y1)=(0,0)。

奶牛贝茜从第一座山的(0,0)位置开始。当贝茜在某座山上时，她会一直爬到这座山的尽头，然后从边上跳下来。如果她落在另一座小山上,她就会继续爬那一座山;否则她就会掉到很远的地方直到她安全降落在气垫（y =-无穷大）上面。每座山(x1,y1)—>(x2,y2)视为包含点(x1,y1)但不包含点(x2,y2),所以,当贝茜掉到一个位置上满足 x=x1 时，就视为落到了那座山上。如果她掉到一个位置上满足 x=x2 时，就不能视为她落到了那座山上。

请计算出贝茜走过的山的总数。(如果贝茜到达了一座山上的一些点，则视为她到过了那座山。)

解题报告：

由于山不会相交、接触，所以固定任意一个 x 坐标，所有的山都是有序的，且这个序是不会改变的（只会中间删除一些)。按 x 轴做扫描线，用一棵平衡树维护所有线的 y 轴顺序，这样可以处理一些山在中途“退出”的情况。每当走到这座山顶时，要找到目前所有山中<=当前 y 的最大值，也是平衡树支持的。

故总复杂度 O(NlogN)。