IOI2015年国家集训队泛做题解

长沙市雅礼中学 刘剑成

2015 年 1 月 29 日

目录

1 codeforces第一部分

1.1 Codeforces 263E Rhombus

1.1.1 题目大意

你有一个$n*m$的表格，其中第i行第j列的元素为$a_{i,j}$，定义函数：

$$f(x,y,k)=\sum_{i=1}^{n}\sum_{j=1}^{n}a_{i,j}*max(0,k-|i-x|-|i-y|)$$

其中x与y均为整数。

现在给定k的值，要求当$k\le x\le n-k+1$且$k\le y\le m-k+1$时函数的最小值为多少。

1.1.2 数据范围

$1\le n,m\le 1000$，$1\le k\le\lfloor\frac{min(n,m)+1}{2}\rfloor$，$0\le a_{i,j}\le 10^6$。

1.1.3 题解

观察$f(x,y,k)$各项系数的值与$f(x,y+1,k)$各项系数的值：

0	0	1	0	0	0
0	1	2	1	0	0
1	2	3	2	1	0
0	1	2	2	0	0
0	0	1	0	0	0

$\Longrightarrow$

0	0	0	1	0	0
0	0	1	2	1	0
0	1	2	3	2	1
0	0	1	2	1	0
0	0	0	1	0	0

将其差分，可得如下矩阵：

0	0	-1	1	0	0
0	-1	-1	1	1	0
-1	-1	-1	1	1	1
0	-1	-1	1	1	0
0	0	-1	1	0	0

可以发现函数从$f(x,y,k)$递推至$f(x,y+1,k)$时，实际上可以看作加上一个三角形内所有元素的权值和再减去一个三角形内所有元素的权值和，从$f(x,y,k)$递推至$f(x+1,y,k)$的方式同理。

接下来，只需要预处理出所有的三角形，用$O(k^2)$的时间暴力算出$f(k,k,k)$的值，即可通过$O(nm)$的递推出所有$f(x,y,k)$ 的值。

在预处理三角形的权值和时，我们同样可以通过差分来求出递推的方式，具体过程在此就不再复述，最终的结果就是加上一条直线的权值和减去两条斜线的权值。

对于线段的权值我们可以用前缀和来解决，至此问题解决。

1.1.4 时空复杂度

时间复杂度：O(nm)

空间复杂度：O(nm)

1.2 Codeforces 293B Distinct Paths

1.2.1 题目大意

现在有一个$n*m$的木板，上面已经有一些格子涂了颜色，你需要用k种颜色去涂满每个没涂色的格子，要求所有从左上角到右下角的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。

已经涂上的颜色也只可能是k种颜色中的一种。

求出一共有多少种不同的涂色方案，答案对10^9+7取模。

1.2.2 数据范围

$1 \le n,m \le 1000$，$1 \le k \le 10$

1.2.3 题解

注意到每条路径不能出现重复的颜色，所以每条路径的长度不能超过k。所以必须满足$n+m-1 \le k$。

这个矩阵最多只有$4*5=20$个格子，数据已经收缩到了一个很小的范围，所以可以直接用暴搜+剪枝来解决。

其中一个非常强的剪枝即为先用最小表示法求出矩阵的值，再乘以全排列。

1.2.4 时空复杂度

时间复杂度：?

空间复杂度：$O(k^2)$

1.3 Codeforces 235E Number Challenge

1.3.1 题目大意

定义函数$d(n)$代表n的约数个数。现在，给定三个数a,b,c，求：

$$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c} d(ijk)$$

答案对2^{30}取模。

1.3.2 数据范围

$1 \le a,b,c \le 2000$

1.3.3 题解

我们考虑$d(i,j,k)$，可以发现它等价于满足条件$a|i, b|j, c|k$且i,j,k两两互质的三元组个数。

首先两者关于每个素数都是独立的，可以分别计算之后再相乘。

考虑对于每一个素数p，在$d(i,j,k)$中，它的乘数为$(p_i+p_j+p_k+1)$，p_i代表i中有p的多少次幂。

而在三元组中，它可能有四种情况$(x,0,0),(0,y,0),(0,0,z),(0,0,0)$，总的乘数也是$(p_i+p_j+p_k+1)$。

则原式等于：

$$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c} e(gcd(i,j))^{1} e(gcd(i,k)) e(gcd(j,k)) \lfloor\frac{a}{i}\rfloor\lfloor\frac{b}{j}\rfloor\lfloor\frac{c}{k}\rfloor$$

我们对于$e(gcd(j,k))$进行莫比乌斯反演，则可得：

$$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c} e(gcd(i,j)) e(gcd(i,k)) \left(\sum_{d=1}^{d|i,d|j} \mu(d)\right) \lfloor\frac{a}{i}\rfloor\lfloor\frac{b}{j}\rfloor\lfloor\frac{c}{k}\rfloor$$

[1] $gcd(x,y)$ 代表x与y的最大公约数，$e(x)$代表$[x=1]$，即当$x=1$时其值为1，否则其值为0，此处两者配合使用表示当两数互质时才将之后的结果计入答案。

则我们优先枚举d：

$$\sum_{d=1}^{d|i,d|j}\mu(d)\sum_{i=1}^{a}e(gcd(d,i))\lfloor\frac{a}{i}\rfloor\left(\sum_{j=1}^{\lfloor\frac{b}{d}\rfloor}e(gcd(i,j))\lfloor\frac{b}{dj}\rfloor\right)\left(\sum_{k=1}^{\lfloor\frac{c}{d}\rfloor}e(gcd(i,k))\lfloor\frac{c}{dk}\rfloor\right)$$

接下来，我们只需要枚举d与a，分别算出$\sum_{j=1}^{\lfloor\frac{b}{d}\rfloor}e(gcd(i,j))\lfloor\frac{b}{dj}\rfloor$与$\sum_{k=1}^{\lfloor\frac{c}{d}\rfloor}e(gcd(i,k))\lfloor\frac{c}{dk}\rfloor$即可。

由于$\sum_{i=1}^{n}\lfloor\frac{n}{i}\rfloor=O(\log_2 n)$，所以总时间复杂度是可以接受的，注意需要预处理出所有$gcd(i,j)$的值。

1.3.4 时空复杂度

时间复杂度：$O(a^2\log_2 a)$

空间复杂度：$O(a^2)$（假设$a\geq b\geq c$）

1.4 Codeforces 325E The Red Button

1.4.1 题目大意

现在一共有n个按钮，现在你从0号按钮开始按，每次如果你按的按钮为x号，则下一次按的按钮必然为$2x \bmod n$或$(2x+1) \bmod n$，求一种合法的按按钮的方案使得除了0号按钮之外，每个按钮都被按且仅被按了一次，0号按钮被按且仅按了两次。无解输出-1。

1.4.2 数据范围

$2\leq n\leq 10^5$

1.4.3 题解

可以发现，题目实际上是要求一个有向图的哈密顿回路，但这个问题如果直接做是没有多项式解法的，所以我们需要将题目转化一下。

我们先观察什么情况下是无解的。对于一个点x，它的入边必然有两条：一条来自$\lfloor\frac{x}{2}\rfloor$，另外一条来自$\lfloor\frac{x+n}{2}\rfloor$。

所以，0号点与$n-1$号点必然各有一条自己连向自己的边。所以0号点的入边必然来自$\lfloor\frac{n}{2}\rfloor$，$n-1$号点的入边必然来自$\lfloor\frac{n-1}{2}\rfloor$。然而，当$n$为奇数的时候有$\lfloor\frac{n}{2}\rfloor=\lfloor\frac{n-1}{2}\rfloor$，则当$n$为奇数的时候是必然无解的。

当n为偶数的时候，可以通过如此构造解：

可以发现，当n为偶数的时候，点x与点$x+1$连入的边是完全相同的，所以我们可以将所有$x, x+1$点对建一个图，$x, x+1$与$2x \bmod n, 2x \bmod n+1$、$2(x+1) \bmod n, 2(x+1) \bmod n+1$连边，这样，所有的点都被转化成了边，只需要求一次欧拉回路就可以了。

1.4.4 时空复杂度

时间复杂度：$O(n)$

空间复杂度：$O(n)$

1.5 Codeforces 241B Friends

1.5.1 题目大意

现在有N个数，定义数对(x, y)的权值为$A_x\ xor\ A_y$，其中$x < y$。

你要选出M个不同的数对(x, y)，使得它们的权值和最大，求最大的权值和为多少。

1.5.2 数据范围

$N \le 50000$，$M \le \frac{N(N-1)}{2}$，$0 \le A_i \le 10^9$

1.5.3 题解

直接求总和貌似做法不明显，我们不妨先求出第M大的权值S具体为多少。

这个可以直接将所有的数字按照二进制表示插入一棵字典树内，二分S的值为多少，之后对于每个数在字典树上查询有多少个包含它的数对大于S即可。

对于求和，我们相当于求权值大于S的数对权值之和有多少。

思考查询的过程，每次我们在字典树上的一个节点，相当于询问它的一棵子树内有多少个有用的节点。所以我们维护一棵子树的总节点数为多少。在这里我们需要求出一个数异或每个数的权值之和，对于异或来说每一位是可以分开来算的，所以我们可以对于每一位直接维护在一棵子树上一共有多少个数这一位为1，有多少个数这一位为0，每次计算时每一位分开算即可。

在清澄上这题需要略微卡常数才能通过。

1.5.4 时空复杂度

时间复杂度：$O(N{\log_2}^2 N)$
空间复杂度：$O(N{\log_2}^2 N)$

1.6 Codeforces 260E Dividing Kingdom

1.6.1 题目大意

在平面上一共有N个点，给出现在你要画出4条不重合的直线，其中有两条与x轴平行，两条y轴平行，将平面分为9块，要求可以分给9个人，刚好第i个人的块中正好有a_i个点。

直线不能穿过点，但是坐标可以为小数，求一个合法的方案。

1.6.2 数据范围

$9 \le N \le 10^5$，$1 \le a_i \le 10^5$且$\sum_{i=1}^{9} a_i = n$，坐标绝对值在10^9以内

1.6.3 题解

我们首先考虑与x轴平行的线，可以发现，对于两条线，每条线最多有$\binom{9}{3}$种不同的画线方法，所以我们可以对于这$\binom{9}{3}$种画线方法分别预处理出在其上方是否可以画出一个包含k个点的区域。

之后，我们只需要暴力枚举每个a_i是对应哪个块即可。

1.6.4 时空复杂度

时间复杂度：$O(9^3 n + 9!)$
空间复杂度：$O(9^3 n + 9!)$

1.7 Codeforces 317E Princess and Her Shadow

1.7.1 题目大意

有一个无限大的森林，其中一些地方是树。现在有一个公主和她的影子，公主和影子都不能够走到树所在的格子，每次公主向一个方向移动的

时候，影子也会朝着那个方向移动，如果影子要移动的方向是一棵树，那么它就不会移动了。

现在公主要走到她影子所在的格子，求一种可行的方案。

1.7.2 数据范围

树的数量为400，坐标的绝对值不超过100

1.7.3 题解

如果公主和影子不在同一个连通块里面，那么肯定无解，除此之外，如果整个地图中没有一棵树，而且公主和影子不在同一个格子内，同样也肯定无解。

接下来，我们只需要不断的让公主去追逐影子当前所在的位置，如果公主与影子被树给完全包围了，那么必然在$O(100^3)$步数内可以追到影子。

否则，如果公主和影子到达了无限大的区间，那么我们进行如此操作：如果影子在公主左边，那么我们可以找到地图中最右边的一棵树，将影子抵在这棵树上不断往左走即可。

同样的，当影子在公主的上面、下面、右边时也可以用相同的方法，最终公主必然会找到影子。

1.7.4 时空复杂度

时间复杂度：$O(100^3)$

空间复杂度：$O(100^2)$

1.8 Codeforces 325D Reclamation

1.8.1 题目大意

现在有个$n * m$的地图，左边界和右边界是相通的。现在给出Q次操作，每次操作挖去其中的一个格子，要求任何时候都存在一条从最上方到最下方的四连通路径，如果某次操作之后将最上方与最下方分开了则不做，问最后有多少次操作是成功的。

1.8.2 数据范围

$1 \le n, m \le 3000$，$1 \le Q \le 10^5$

1.8.3 题解

首先假设这个地图不是环形的，则判断地图最上方与最下方不连通可以使用一种常用的方法——转化对偶图，即我们可以通过判断地图最左与最右是否被所有的障碍物连通，如果最左边与最右边连通了，那么是无论如何也找不到一条从上方到下方的路径的。注意此处的连通为八连通。

连通性可以用并查集来处理。

接下来思考如何处理地图成环的情况，一种更常见的方法是将地图翻两倍，之后通过判断在第一个地图中的点(x, y)是否与复制了一遍的点$(x, y + m)$连通即可。

1.8.4 时空复杂度

时间复杂度：$O(nm * \alpha(Q))$

空间复杂度：$O(nm)$

1.9 Codeforces 335E Counting Skyscrapers

1.9.1 题目大意

大街上建好了一排摩天大楼。摩天大楼的数量是在2到314!（314的阶乘，一个非常大的数）中均匀随机选择的。每座摩天大楼的高度（即楼层数）是被独立地随机选择的：对于每个正整数i，楼层数为i的概率为2^{-i}。如果一座摩天大楼有i 层，那么它的楼层被编号为0到$i-1$。

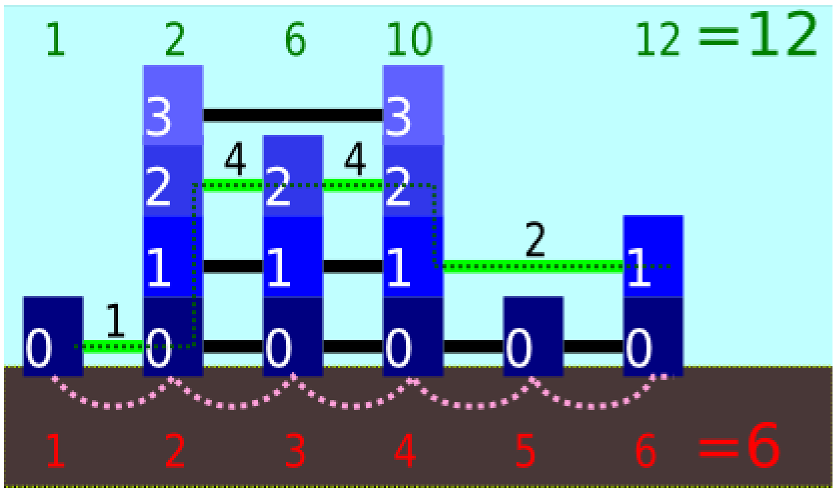
为了加快中转运输的效率，摩天大楼间修建了一些滑索。一座摩天大楼的第i层和另一座摩天大楼的第i 层之间有滑索当且仅当两楼之间没有摩天大楼有第i层。

Alice和Bob想数一数有多少座摩天大楼。

Alice是个严谨认真的人，她想知道摩天大楼数量的准确值。于是她把计数器初始化为1，从最左边的摩天大楼开始往右走，每次走到一个摩天大楼就把计数器加上1，直到她到达最右边的摩天大楼。

Bob很没耐心，他想尽快完成任务。于是他把计数器初始化为1，从最左边的摩天大楼开始利用滑索从一座摩天大楼滑到另一座。因为恐高，每次他会忽略那些高度（即所在楼层编号）大于h的滑索，然后在剩下的往右滑的滑索中选一个最高的。由于Bob使用滑索时滑得太快以至于他无法数

清经过了多少座摩天大楼，所以他直接将计数器加上2^i，其中i是他当前所在的楼层编号。他会一直持续这个过程直到他到达最右边的摩天大楼。



当Alice和Bob到达最右边的摩天大楼时，他们会比较计数器的值。现在给出Alice的计数器的值或者Bob的计数器的值，请你求出另一人的计数器的值的期望。

1.9.2 数据范围

$2 \le n \le 30000$，$0 \le h \le 30$

1.9.3 题解

首先考虑已知Bob计数器值的部分。

如果已知了所有Bob走到过的楼的高度，那么滑索的高度和Bob计数器的值都好求。问题就在于如何求此时Alice计数器的期望值。

我们可以对于每个滑索单独考虑经过的摩天大楼的期望数量（不计算左端点，计算右端点），由于摩天大楼的数量上限可以看作无限大，于是我们可以认为每根滑索经过的摩天大楼的数量这个随机变量是相互独立的。由于期望的线性叠加，我们可以单独计算每根滑索经过的摩天大楼的期望数量再加上Alice计数器初始值1得到Alice最终的期望值。

考虑一根Bob走过的且高度为h_t的滑索，那么中间的摩天大楼高度一定要满足高度小于h_t。

设滑索经过的摩天大楼数量为随机变量L，设条件P_L为中间的$L-1$个摩天大楼的高度都小于h_t，则有：

$$P(L=l|P_L)=\frac{P(P_L\cap L=l)}{P(P_L)}=\frac{P(L=l)P(P_L|L=l)}{\sum_{k\geq 1}P(L=k)P(P_L|L=k)}$$

由于摩天大楼数量是均匀随机选取的，所以：

$$P(L=l|P_L)=\frac{P(P_L|L=l)}{\sum_{k\geq 1}P(P_L|L=k)}=2^{-h_t}(1-2^{-h_t})^{l-1}$$

可以求出期望：

$$E(L|P_L)=\sum_{l\geq 1}P(L\geq l|P_L)=2^{h_t}$$

当Bob滑一次之后，同样也是增加2^{h_t}，可以发现Bob的计数值恰好与期望值相等，所以当Bob计数器的值为n时，Alice计数器的期望值也正好为n。

接下来考虑已知Alice计数器值的部分。

我们可以逐步增加h，每次增加h时，相当于原来高度为h的摩天大楼中有一些升高到$h+1$，我们考虑这件事情对于Bob期望值的影响。

由于期望的线性叠加，我们可以考虑每两点之间的滑索对期望的影响。

假设现在h从$x-1$增加到了x，我们考虑一根新出现的高度为x的滑索。这条滑索下面会有若干条高度为$h-1$的滑索，这些是当$h=x-1$时Bob的行走路线。而当$h=x$时，Bob放弃了这些滑索。所以一根新出现的高度为h的滑索对于Bob计数器期望的贡献为：

$$P(\text{该滑索存在})*(E(\text{该滑索权值})-E(\text{下方滑索权值和}))$$

假设这跟滑索经过了L座摩天大楼（不包括左端点，包括右端点），那么这条滑索存在的概率为：

$$P(\text{该滑索存在})=P(H=x)P(H<x)^{L-1}P(H=x)=2^{-2x}(1-2^{-x})^{L-1}$$

由于该滑索高度为x，所以必然有：

$$E(\text{该滑索权值})=2^x$$

考虑损失，即求下方滑索的期望条数，同样也是中间高度为$x-1$摩天大楼的期望数量+1：

$$E(\text{下方滑索权值和})=\left((L-1)\frac{2^{-x}}{1-2^{-x}}+1\right)2^{x-1}=\left(\frac{L-1}{2^x-1}+1\right)2^{x-1}$$

由于经过L座摩天大楼的滑索一共有$n-L$个可能的位置，我们得到当h从$x-1$变为x时的期望变化量为：

$$\sum_{L=1}^{n-1}(n-L)2^{-2x}(1-2^{-x})^{L-1}\left(2^x-\left(\frac{L-1}{2^x-1}+1\right)2^{x-1}\right)$$

化简之后可得：

$$\sum_{L=1}^{n-1}(n-L)2^{-x}(1-2^{-x})^{L-1}\left(\frac{1}{2}-\frac{1}{2}\cdot\frac{L-1}{2^x-1}\right)$$

由于初始$h=0$时，Bob计数器的值期望为n，所以对于任意h，Bob计数器的期望值为：

$$n+\sum_{x=1}^{h}\sum_{L=1}^{n-1}(n-L)2^{-x}(1-2^{-x})^{L-1}\left(\frac{1}{2}-\frac{1}{2}\cdot\frac{L-1}{2^x-1}\right)$$

至此，问题解决。

1.9.4 时空复杂度

时间复杂度：$O(nh)$

空间复杂度：$O(h)$

1.10 Codeforces 360D Levko and Sets

1.10.1 题目大意

有n个集合，其中第i个集合生成方式如下：

1.初始这个集合只有一个数字1。

2.从这个集合中任意选出一个数字c，对于所有的j，如果$c*{A_i}^{B_j} \bmod p$不存在于当前的集合中，则将它加入当前集合。

3.重复步骤2，直到无法将任意元素加入集合。

给定数组A与数组B，求n个集合并的大小。

1.10.2 数据范围

$1 \le n \le 10000$，$1 \le m \le 10^5$，$1 \le B_i \le 10^9$，$2 \le p \le 10^9$，且p 为质数，$1 \le A_i \le p$。

1.10.3 题解

注意到这里每次是乘以$A_i^{B_j}$，则我们可以很自然的想到原根。

我们将所有的A_i用原根表示出来，则在以原根为底的对数意义下，乘法就变成了加法。假设$A_i \equiv g^k \pmod n$。令$x = gcd(A_1, A_2, ..., A_n, p-1)$，则所有能表示出来的数为$g^{xk}$。

接下来问题转化为有一个数集，求$[0, p)$中有多少个数能被这个数集中的至少一个数正处。

这个是一个很经典的容斥原理，对于每个出现在给定数集的$p-1$的因数x，设$f[x]$表示能被x整除又不能被x的出现在数集中的倍数整除的数有几个，直接用约数个数平方的暴力算就可以了。

1.10.4 时空复杂度

时间复杂度：$O(k \log^2 p + S + n\frac{p}{S} + d^2)$，其中$d$为$p-1$的约数个数。

空间复杂度：$O(n + S)$

1.11 Codeforces 339E Three Swaps

1.11.1 题目大意

现在有一个排列P，你可以进行最多三次操作，每次操作为将一段区间内的元素翻转，求一个从$\{1, 2, 3, ..., n\}$变为P的方案。

数据保证一定有解。

1.11.2 数据范围

$1 \le n \le 1000$

1.11.3 题解

我们将整个问题反过来考虑，则变成进行三次操作从P变回$\{1, 2, 3, ..., n\}$，可以发现我们在输出时反过来即为从$\{1, 2, 3, ..., n\}$变为P的操作方法。

对于每次翻转，我们加上一个条件：每次翻转(x, y)时都必须要满足$|P_x - P_{y+1}| = 1$或者$|P_y - P_{x-1}| = 1$。

事实证明只要P是$\{1, 2, 3, ..., n\}$经过不超过3步操作得到的，它就一定可以在加了这个条件之后花费不超过3步从P变回$\{1, 2, 3, ..., n\}$。

接下来，我们只需要枚举每一步翻转的区间有多少个，由于总共只会交换3次，所以满足条件的区间也不会有很多。

1.11.4 时空复杂度

时间复杂度：$O(n^2)$

空间复杂度：$O(n)$

1.12 Codeforces 286E Ladies' Shop

1.12.1 题目大意

现在有N个包，每个包的容积为A_i，现在你需要给出若干个体积，对于每种体积的物品都有无限多个，要求满足以下条件：

1. 对于每个包，都必须有一个选择物品的方案，使得这些物品的体积之和正好等于这个包的容积。
2. 对于每种选择物品的方案，如果它的总体积没有超过M，那么必须存在一个容积正好等于这些物品体积之和的包。
3. 在同时满足满足前两个条件的情况下，要求给出的不同的体积个数最少。

判断是否有可行方案，若有则求可行方案。

1.12.2 数据范围

$1 \le N \le M \le 10^6$，$1 \le A_1 < A_2 < ... < A_N \le M$

1.12.3 题解

我们先来思考如何判断一个包的集合存在一个合法的体积选择方案。

对于每种选择物品的方案，都必须要有一个包与这些物品体积之和相同，所以对于每种物品，它的体积必然要与一个包的容积相同。

所以对于一个总体积小于M的选择物品方案S，若$k>1$，则它必然可以分为两个子集合S_1与S_2，其中$S_1+S_2=S$。由于S_1、S_2同样也是两种合法的选择方案，所以必然可以分别用一个包的容积来表示。那么S的体积和必然可以分为两个包的容积和。

一个包的集合存在一个合法的体积选择方案，当且仅当对于容积为A_i、A_j的两个包，若$A_i+A_j\le M$，存在一个包的容积为A_k，满足$A_k=A_i+A_j$。

但是如果直接暴力$O(n^2)$的判断显然是不行的，需要时间复杂度更优的算法。有一种很常见的方法即为将整个序列转化为多项式：

$$F(x)=\sum_{i=1}^{n}x^{A_i}$$

我们只需要观察F^2中不大于M的每个有系数的项是否在集合A中存在即可。

对于求k最小的方案，我们可以用贪心的思想，每次找到最小的A_i，如果A_i可以由其他A_j的和表示，那么这个A_i可以不选，否则A_i是必须要选的。

1.12.4 时空复杂度

时间复杂度：$O(M\log_2 M)$

空间复杂度：$O(M)$

1.13 Codeforces 332D Theft of Blueprints

1.13.1 题目大意

给出一个n个点的带权无向图，其中若i与j没边，则$c_{i,j}=-1$，否则$c_{i,j}$代表i到j的边权。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$v(S)$，对S进行操作的代价是S中每个点与$v(S)$ 的边权之和。

现在任意选一个大小为k的子集，求对它进行操作的代价期望为多少。

1.13.2 数据范围

$1\le k<n\le 2000$，$-1\le c_{i,j}\le 10^9$。

1.13.3 题解

由于题目保证对于任意大小为k的集合都满足条件，所以我们只需要求出所有大小为k的集合的权值和，最后除以$\binom{n}{k}$即可。

考虑枚举点v，那么点v的贡献即为$\sum_{i}^{c_{v,i}\neq -1} c_{v,i} * \binom{p-1}{k-1}$，其中$p$代表点$v$一共有多少条边。

接下来我们可以通过这个式子来暴力做了，需要注意一下精度问题。

事实上，可以证明k所有可能的取值只有3种——1、2、$n-1$。其中当$k=1$时，该图为一个完备匹配；当$k=2$时，该图为一个$n-1$个点的完备匹配和一个与所有点都有边的点；当$k=n-1$时，该图为一个完全图。

我们只需要对于$k=1$，$k=2$，$k=n-1$三种情况特判就可以了。

1.13.4 时空复杂度

时间复杂度：$O(n^2)$
空间复杂度：$O(n)$

1.14 Codeforces 335D Rectangles and Square

1.14.1 题目大意

现在有n个矩形，所有矩形的四个角都是整点，并且两组对边分别平行于x轴和y轴。不同的矩形可能接触，但是不会重叠。问是否可以选出一个矩形的集合组成一个正方形，换而言之满足以下两个条件：

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。

求一种合法方案。

1.14.2 数据范围

$1 \leq n \leq 10^5$，坐标在0和3000之间。

1.14.3 题解

首先我们先来观察如何判断一个正方形是不是合法的。

一个正方形是合法的，当且仅当满足以下条件：

1. 一个正方形的边界必然是由许多矩形的边界拼凑出来的；
2. 一个正方形的内部必然是由矩形填满。

我们只需要预处理一下前缀和，就可以在$O(1)$的时间内确定一个正方形是不是合法的。之后，我们只需要枚举所有可能的正方形即可。

注意到一个正方形的四个顶点必然也是某些矩形的顶点，这样我们只需要枚举正方形上面的两个顶点，判断是否可行即可。

1.14.4 时空复杂度

时间复杂度：$O(3000n)$

空间复杂度：$O(3000^2)$

1.15 Codeforces 261D Maxim and Increasing Subsequence

1.15.1 题目大意

给你一个数组A，数组中的每个值不超过m。现在将A复制t次求它的最长上升子序列长度。

有k组数据。

1.15.2 数据范围

$1 \le k \le 10$，$1 \le n, m \le 10^5$，$1 \le t \le 10^9$，$n * m \le 2 * 10^7$

1.15.3 题解

首先，当t大于$min(n, m)$的时候，可以发现答案即为不同字符的个数，所以我们在这里只考虑$t \le min(n, m)$的情况，此时t满足$t * n \le 2 * 10^7$。

我们可以依次枚举当前到了哪一位，设f_j代表在此之前最长上升子序列最后一位小于j时的最长长度为多少。

假设当前到了第i位，则我们每次可以用$f_{a_i-1}+1$来更新所有的f_k，其中k满足$k > a_i$且$f_k = f_{a_i-1}$。

注意到f数组在任何时刻都是不递减的，而每次我们相当于把一些值给+1，注意到f中最大的值不可能超过n，所以这样总共也只会加$n*m$次。总时间也是$n*m$的。

1.15.4 时空复杂度

时间复杂度：$O(knm)$
空间复杂度：$O(m)$

1.16 Codeforces 235C Cyclical Quest

1.16.1 题目大意

给出一个长度为N的字符串s，同时有Q个询问，每次给出串x_i，问在s中共有多少个字串与它是循环同构的。

串a与串b循环同构，当一个串可以通过旋转获得另一个。旋转在这里是指将字符串开头的若干字符移至字符串结尾。

1.16.2 数据范围

$N \le 10^6$，$1 \le Q \le 10^5$，询问串的总长度不超过10^6。

1.16.3 题解

对于每个询问x_i，我们可以看作是询问x_i的各个循环变体在s中一共出现了多少次。我们枚举x_i所有的循环变体，相当于每次在查询串的前面去掉一个字符，在查询串的后面加上一个字符。

这种查询在后缀自动机上是很好做的。所以对于s，我们建出它的后缀自动机，在自动机上运行即可。

1.16.4 时空复杂度

时间复杂度：$O(n)$
空间复杂度：$O(n)$

1.17 Codeforces 311E Biologist

1.17.1 题目大意

现在有n条狗，每条狗有一个性别，修改第i条狗的性别需要v_i的代价，现在给出m个要求，第i个要求给出了一个狗的集合S_i，当在S_i中所有狗的性别都为给定的性别时，会获得w_i的金钱，否则会失去$g * c_i$的金钱。

求最小的收入是多少。

1.17.2 数据范围

$1 \le n \le 10000$，$0 \le m \le 2000$，$0 \le g, v_i, w_i \le 10000$，$0 \le c_i \le 1$，所有$S_i$的大小均在10以内。

1.17.3 题解

把题目转化为最小花费，建立最小割模型，跑最大流即可。

具体建图过程为：

1. 对于每条狗i，若该狗为雄性，则从S向这个点连边，否则从这个点向T连边，边的流量为v_i。
2. 对于每个要求i，若要求为雄性，则从S向这个点连流量为$g * c_i + w_i$的边，从这个点向要求的狗连边权为无限大的边，否则从这个点向T连流量为$g * c_i + w_i$的边，从要求的狗向这个点连边权为无限大的边。

1.17.4 时空复杂度

时间复杂度：$O(maxflow(n + m, n + km))$

空间复杂度：$O(n + km)$其中k为每次要求中集合的大小。

1.18 Codeforces 306D Polygon

1.18.1 题目大意

构造一个n个点的凸多边形，使得每个角的角度相同，每条边的长度互不相同。

1.18.2 数据范围

$3 \le n \le 100$

1.18.3 题解

首先当$n < 5$时，必然是无解的。

当$n \ge 5$时，我们可以从原点出发，每次旋转一个角度，向这个方向上走一个很长的距离，每次走的距离增加一个很小的量，这样子我们可以确定$n-1$个点，最后一个点可以通过求两条直线的交点来得到。

事实证明这样子可以构造出所有的凸多边形。

1.18.4 时空复杂度

时间复杂度：$O(n)$
空间复杂度：$O(1)$

1.19 Codeforces 249E Endless Matrix

1.19.1 题目大意

现在给出一个矩阵，里面有连续的数字，从1开始，若$a_{i,j} < a_{t,k}$，当且仅当：

1. $max(i,j) < max(t,k)$
2. $max(i,j) = max(t,k)$且$j < k$
3. $max(i,j) = max(t,k), j = k$且$i > t$

例如前36个数字形成的矩阵：

1	2	5	10	17	26
4	3	6	11	18	27
9	8	7	12	19	28
16	15	14	13	20	29
25	24	23	22	21	30
36	35	34	33	32	31

定义：

$$f(x_1,y_1,x_2,y_2)=\sum_{i=x_1}^{x_2}\sum_{j=y_1}^{y_2}a_{i,j}$$

求$f(x_1,y_1,x_2,y_2)$的末尾10位。

有t组数据。

1.19.2 数据范围

$1\le t\le 10000$，$1\le x_1\le x_2\le 10^9$，$1\le y_1\le y_2\le 10^9$

1.19.3 题解

首先，对于$f(x_1,y_1,x_2,y_2)$，我们可以将其转化为$F(x_2,y_2)-F(x_1-1,y_2)-F(x_2,y_1-1)-F(x_1-1,y_1-1)$，其中：

$$F(x,y)=\sum_{i=1}^{x}\sum_{j=1}^{y}a_{i,j}$$

接下来所需要考虑的即为如何求出$F(x,y)$的值。

情况1：$x=y$，此时$F(x,y)=\frac{(1+x^2)x^2}{2}$。

情况2：$x<y$，我们可以把整个矩阵分为两块：矩形$(1,1)\sim(x,x)$之和与矩形$(1,x+1)\sim(x,y)$ 之和，设：

$$G(x,y)=\sum_{i=x}^{x+y-1}i$$

则有：

$$F(x,y)=F(x,x)+\sum_{i=x}^{y-1}G(i^2+1,x)$$

化简可得：

$$F(x,y)=\frac{x(x^3-x+4y+3xy-3y^2+2y^3)}{6}$$

情况3：$x>y$，同理也可以将整个矩阵分为两块矩形$(1,1)\sim(y,y)$之和与矩形$(y+1,1)\sim(x,y)$ 之和。

同样也有：

$$F(x,y)=F(y,y)+\sum_{i=y+1}^{x}G(i^2-y+1,y)$$

化简可得：

$$F(x,y)=\frac{y(4x+3x^2+2x^3-y-3xy+y^3)}{6}$$

这样我们同样也可以在$O(1)$的时间内求出F(x,y)。

注意在答案大于等于10^{10}时需要输出...，所以我们可以先用long double跑一遍，再在模意义下跑第二遍即可。

1.19.4 时空复杂度

时间复杂度：$O(t)$

空间复杂度：$O(1)$

1.20 Codeforces 295D Greg and Caves

1.20.1 题目大意

有一个$n*m$的矩阵，你需要将其中的一些点给染黑，要求满足以下条件：

1. 这个图不能为空
2. 假设最上面的黑点在第L行，最下面的黑点在第R行，则所有在$[L,R]$之内的行中必须正好有两个点。
3. 假设第i行中的两个黑点分别在第x_i列和第y_i列，则必须存在一行t，使得所有$L\le j<t$满足$x_{j+1}\le x_j<y_j\le y_{j+1}$，所有$t<j\le R$满足$x_{j-1}\le x_j<y_j\le y_{j-1}$，

求一共有多少种不同的涂黑方法。

1.20.2 数据范围

$1\le n,m\le 2000$

1.20.3 题解

可以发现这题中按照t行分开成了一个上边界和一个下边界，那么我们可以分开解决上边界和下边界，之后再合并起来。

对于暴力，我们可以设出$F_{i,l,r}$代表到了第i行，左边的黑点与右边的黑点在第l列和第j列，那么转移方程为：

$$F_{i,l,r} = 1 + \sum_{x=l}^{r-1} \sum_{y=i+1}^{r} F_{i-1,x,y}$$

可以发现，这个DP中我们实际上只用考虑最后一行中黑点的距离，之后再考虑放的位置，所以我们可以直接按照最后一行中黑点的距离来设出状态，$F_{i,j}$代表到了第i行，最后一行的两个黑点之间距离为j且还没有决定最左边放在哪里的方案数，那么我们可以列出转移方程：

$$F_{i,j} = 1 + \sum_{l=1}^{j} (j - l + 1) F_{i-1,l}$$

对于转移我们使用前缀和优化即可在$O(n^2)$的时间内解决。

最后我们可以将两个部分给拼起来，假设下面一部分的方案为$G_{i,j}$，那么总答案即为：

$$\sum_{t=1}^{n} \sum_{i=1}^{m-1} (m - i) F_{t,i} \left(1 + \sum_{j=1}^{i-1} (i - j + 1) G_{t+1,j} \right)$$

同样使用前缀和优化即可。

1.20.4 时空复杂度

时间复杂度：$O(nm)$

空间复杂度：$O(nm)$

1.21 Codeforces 258D Little Elephant and Broken Sorting

1.21.1 题目大意

现在有一个大小为n的排列，我们对这个排列进行m次操作，每次交换在位置x与位置y上的值，现在告诉你每个操作成功的概率为$\frac{1}{2}$，求最终排列中逆序对的个数期望为多少。

1.21.2 数据范围

$1 < n \leq 1000$，$1 \leq m \leq 1000$

1.21.3 题解

我们设$f_{i,j}$代表$a_i > a_j$的概率。

接下来，我们只需要在每次交换时修改与修改有关的概率即可。

假设交换的位置为x与y，则会影响到的概率只有所有的$f_{i,x}$、$f_{i,y}$、$f_{x,i}$和$f_{y,i}$。

$f_{i,x} = f_{i,y} = \frac{fi,x+fi,y}{2}$，$f_{x,i} = f_{y,i} = \frac{fx,i+fy,i}{2}$

每次交换的时候我们都暴力修改$O(n)$个值，最终我们需要求的答案即为$\sum_{i=1}^{n} \sum j = i + 1^n f_{j,i}$

1.21.4 时空复杂度

时间复杂度：$O(n^2)$

空间复杂度：$O(n^2)$

1.22 Codeforces 273D Dima and Figure

1.22.1 题目大意

有一块$n * m$的方格图，你需要将其中的一些格子给涂黑，要求满足以下条件：

1. 至少有一个格子被涂黑。
2. 所有涂黑的格子形成一个四连通的连通块，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，一次移动可以从一个涂黑的格子可以移动到另一个与它四联通且被涂黑的格子里。
3. 从一个涂黑的格子(x_1, y_1)移动到另一个涂黑的格子(x_2, y_2)所需要移动的次数为$|x_1 - x_2| + |y_1 - y_2|$

问一共有多少种不同的涂黑方案。

1.22.2 数据范围

$1 \leq n, m \leq 150$

1.22.3 题解

观察第二个条件，可以发现实际上我们涂黑的格子组成了一个凸多边形。注意在此处凸多边形的定义是指左边界不断递减再不断递增，右边界不断递增再不断递减的黑块。

那么我们可以直接设出状态：F_{i,j,k,p_l,p_r}代表到了第i行，当前行涂黑的左边界为l，涂黑的右边界为r，且左边界的增减状态为p_l，右边界的增减状态为p_r时的方案数为多少。

这样我们设出了$O(n^3)$的状态，注意在转移时若直接枚举上一行的左右边界是会超时的，由于是求和，所以我们可以直接用前缀和优化即可$O(1)$转移。

1.22.4 时空复杂度

时间复杂度：$O(n^3)$
空间复杂度：$O(n^3)$

1.23 Codeforces 283E Cow Tennis Tournament

1.23.1 题目大意

有n个点，每个点有个权值a_i，一个点x向另外一个点y连出了单向边，当且仅当$a_x > a_y$。

现在有m个操作，每次将所有满足$l_i \le a_x \le r_i$且$l_i \le a_y \le r_i$的边$x \Rightarrow y$变为$y \Rightarrow x$。

问最终会有多少个三元组(a, b, c)满足a向b连边，b向c连边，c向a连边。

1.23.2 数据范围

$3 \le n \le 10^5$，$0 \le m \le 10^5$，$1 \le a_i \le 10^9$，$1 \le l_i < r_i \le 10^9$

1.23.3 题解

求三元环的个数，我们可以将其补集转化一下，变为求有多少个(a, b, c)满足a向b连边，a向c连边，b向c连边。

可以发现这样子转化了之后，实际上我们是要求从一个点连出去的边中任选两条边的方案有多少种。最终答案即为：

$$\binom{n}{3} - \sum_{i=1}^{n} \binom{p(i)}{2}$$

其中$p(i)$代表点i的度数。

所以实际上我们需要求的即为每个点连出去的边各有多少条。

我们将所有的点按照权值从小到大排序之后重标号，在初始的情况下它向它左边的所有点连边。

对于一次操作(l_i, r_i)，如果当前枚举的点x满足$l_i \le a_x \le r_i$，那么对x连边情况的影响即为将所有满足$l_i \le a_y \le r_i$的点y与其的连边情况取反。

我们直接枚举点x，维护一个01数组，如果当前x向y连边的方式改变了，则记录为0，否则记录为1。

我们离线处理每次操作，用类似于扫描线的思想，一路扫过去，进行以下修改：

1. 扫到第i个点时，把所有以i为左端点的修改区间拿出来，将这些区间对应的做一次区间的$0, 1$反转操作。
2. 统计第i个点左边的0与右边的1，这就是第i个点连出去的边的条数。
3. 将所有以i为右端点的修改区间拿出来，将这些区间的贡献进行“撤销”，即将对应区间再做一次区间的$0, 1$反转操作。

接下来我们只需要用线段树来维护即可。

1.23.4 时空复杂度

时间复杂度：$O(m \log_2 n)$

空间复杂度：$O(n + m)$

1.24 Codeforces 305D Olya and Graph

1.24.1 题目大意

现在有一个n个点，m条边的有向无权图。你要往其中添加任意多条边，使得该图满足以下条件：

1. 从i点出发可以到达所有满足$j > i$的点j。

2. 任意从u到v的有向边满足$u < v$。
3. 两点之间最多只有一条边。
4. 对于一对点i、j，若$1 \le j - i \le k$，则从i到j的最短距离为$j - i$条边。
5. 对于一对点i、j，若$j - i > k$，则从i到j的最短距离为$j - i$或$j - i - k$条边。

我们认为两种添加边的方案不同，当且仅当存在至少一对点i、j，在第一张图内有$i \Rightarrow j$的边，在第二张图内没有。

求共有多少种不同的添边方案，答案对$10^9 + 7$取模。

1.24.2 数据范围

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

1.24.3 题解

考虑对于$j - i = 1$的点对，此时由于条件4则它们之间必须要有一条边。

在连上所有$j - i = 1$的边时，我们考虑剩下的边应该怎么连。

由于$j - i > k$时，从i到j的最短距离为$j - i$或$j - i - k$条边，则除了连接x与$x + 1$两个点的边只能存在连接x到$x + k + 1$两个点的边，而且在一条路径中不能经过两条这种边。即当我们确定了所有的边时，不能存在两条边$x \Rightarrow x + k + 1$和$y \Rightarrow y + k + 1$满足$x + k + 1 \le y$。

所以我们可以枚举最小的x存在$x \Rightarrow x + k + 1$的边，判断是否其他的边都在$x + 1$到$x + k$之间，接下来我们只需要记录在$x + 1$到$x + k$之间有多少个y在原图中没有$y \Rightarrow y + k + 1$的边，将答案增加2的这么多次幂即可。

1.24.4 时空复杂度

时间复杂度：$O(n)$

空间复杂度：$O(n)$

1.25 Codeforces 261E Maxim and Calculator

1.25.1 题目大意

有两个数字a, b，一开始$a = 1, b = 0$。现在对其进行操作，第一种为将b加1，第二种为将a乘b。

求有多少个x满足$l \leq x \leq r$，使得可以在不超过p步之内将a的值变为x。

1.25.2 数据范围

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

1.25.3 题解

注意到题目中的p非常小，所以我们不妨从p入手。

每次操作我们可以将a乘上b，或者将b加1，可以发现最终b的值是不会超过p的，所以一个合法的a必然可以用在p以内数的乘积来表示。

这样，我们暴力找出在r以内的质因子均在p以内的数，这种数的数量很少，总共约$3 * 10^6$个。

接下来，我们就可以进行暴力dp了，设$f_{i,j}$代表当b中的值到了i，a中的值到了j时最少需要操作多少步。

转移和题目中用相同的方法即可。

1.25.4 时空复杂度

时间复杂度：$O(np)$

空间复杂度：$O(n)$其中n表示在10^9以内质因子均在p以内的数的个数

1.26 Codeforces 253E Printer

1.26.1 题目大意

我们有一台这样子的打印机，它从0时刻开始工作，每秒钟打印一张纸，某些时刻它会接收到些打印任务。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$1 \sim n$，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T。

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。

1.26.2 数据范围

$1 \le n \le 50000$，$0 \le t_i \le 10^9$，$1 \le s_i, q_i \le 10^9$，$1 \le T \le 10^{15}$

1.26.3 题解

假设我们已知了所有任务的优先级，此时问题变成了一个很简单的贪心+模拟，可以直接用优先队列来实现。

现在我们来考虑未知的优先级，可以发现，任务完成的时间必然与优先级是一个单调函数。当优先级越高，任务完成的时间必然不会更晚，而优先级越低，任务完成的时间必然不会更早。

所以我们二分任务的优先级，再每次贪心求解即可。需要注意任意两个任务的优先级不能相同。

1.26.4 时空复杂度

时间复杂度：$O(n{\log_2}^2 n)$

空间复杂度：$O(n)$

1.27 Codeforces 309E Sheep

1.27.1 题目大意

给出n个区间(l_i, r_i)，两个点i, j之间有边，当且仅当区间(l_i, r_i)与(l_j, r_j)有交。

现在你需要给n个点排个序，使得所有有边的点对(x, y)中，相距最远的距离最短。

1.27.2 数据范围

$1 \le n \le 2000$，$1 \le l_i \le r_i \le 10^9$

1.27.3 题解

看到相距最远的距离最短，首先想到的是二分答案。

现在题目变为了判断是否能安排一个顺序，使得两个区间如果相交，则它们的距离不能超过D。

我们从前往后依次考虑第i个位置放哪个点。我们可以维护一个$limit$数组，代表每个点当前所能放到的最大的值。

可以理解为找点和位置一个一一对应的关系，同时每个点都有一个不超过$limit$的限制。

接下来我们根据霍尔定理贪心，为了尽可能保证能有完备匹配，如果同时有i个点要放在前i个位置，应该在从这i个点中找一个对应区间右端点最小的区间放在最左边的位置，这样所产生的约束必然最小。

1.27.4 时空复杂度

时间复杂度：$O(n^2 \log_2 n)$

空间复杂度：$O(n)$

1.28 Codeforces 316E3 Summer Homework

1.28.1 题目大意

给定一个数列$a_1, a_2, ..., a_n$，现在对它进行m次操作，操作共有三种

1. 给定x, z，将a_x修改为z。
2. 给定x, y，询问$\sum_{i=l}^{r} F(i-l) \cdot a_i$的值，其中$F(0) = F(1) = 1$，$F(i) = F(i-1) + F(i-2)$当$i > 2$。
3. 给定x, y, z，将所有满足$x \le i \le y$的a_i加上z。

输出所有询问的结果，对10^9取模

1.28.2 数据范围

$1 \le n, m \le 2 * 10^5$，$1 \le x, y \le n$，$0 \le a_i, v \le 10^5$

1.28.3　题解

既然是区间修改操作，那么我们可以用线段树来维护这个数列。

注意到在询问的时候需要乘上Fibonacci数列，那么我们可以采用线段树维护矩阵乘法的方法来维护这个值。

对于区间加的操作，同样的和一般的区间加操作维护一个标记。注意当我们打上一个区间加的标记时，这个区间的值增加的是Fibonacci序列一个前缀和的若干倍，在这里我们使用$\sum_{i=1}^{n} F_i = F_{n+2} - 1$即可。

1.28.4　时空复杂度

时间复杂度：$O(n + m \log_2 n)$

空间复杂度：$O(n)$

1.29　Codeforces 243C Colorado Potato Beetle

1.29.1　题目大意

有一个$(10^{10}+1)*(10^{10}+1)$的网格，在正中间小块的中心上有个人。

现在他移动了n次，每次移动都是沿着和坐标轴平行的方向走正整数的距离，每个他经过的格子上有杀虫剂，问一共有多少个格子不会被虫子侵入。

虫子首先会侵入在网格边缘的格子，接下来，一个格子会被入侵，当它与一个被入侵的格子相邻且没有撒杀虫剂。

1.29.2　数据范围

$1 \le n \le 1000$，$1 \le x \le 10^6$。

1.29.3　题解

我们将所有的坐标给离散化，接下来可以直接BFS来求出所有会被虫子侵入的格子，剩余的格子即为安全的格子。

1.29.4　时空复杂度

时间复杂度：$O(n^2)$

空间复杂度：$O(n^2)$

1.30 Codeforces 256D Liars and Serge

1.30.1 题目大意

有n个人，有一些人只会说真话，有一些人只会说假话。

你向每个人问一个问题："有多少人在说真话"。

说真话的人一定会告诉你正确答案，说假话的人一定会告诉你一个错误答案。

你在听了所有人的回答之后，得出了结论：有至少m个人撒谎了。现在问他们一共有多少种不同的回答方法使得你一定能得出有至少m个人撒谎的结论。

1.30.2 数据范围

$1 \le n \le 256$，n必然为2的整数0次幂，$1 \le m \le n$

1.30.3 题解

我们能很容易设出一个$O(n^4)$的暴力DP：$F_{i,j,k}$代表到了i个数，当前已经有j个人，并且有k 个人已经确定是撒谎了。

转移也很容易实现，直接枚举回答i的人有x个，当$k = x$时从$F_{i-1,j-x,k}$转移，否则从$F_{i-1,j-x,k-x}$转移。

这样对于$n \le 64$的数据能轻松解决，稍微卡常数也可以过$n = 128$的数据。

对于$128 < n \le 256$的数据，注意到n必然是2的整数次幂，所以n只可能取到256，所以我们对于$n = 256$时打表即可。

1.30.4 时空复杂度

时间复杂度：$O(n^4)$

空间复杂度：$O(n^2)$

1.31 Codeforces 303D Rotatable Number

1.31.1 题目大意

找到最大的$b(1 < b < x)$，满足在b进制下存在一个长度为n的循环数(允许有前导零)。

循环数是一个整数，满足乘连续的若干个数后各位发生循环，如142857即为可选转数：

$142857 \times 1 = 142857$

$142857 \times 2 = 285714$

$142857 \times 3 = 428571$

$142857 \times 4 = 571428$

$142857 \times 5 = 714285$

$142857 \times 6 = 857142$

1.31.2 数据范围

$1 \le n \le 5 * 10^6$，$2 \le x \le 10^9$

1.31.3 题解

这题有一个较为明显的结论，即若存在一个长度为n的循环数，则$n+1$必然为质数，且找到的进制b也必然是$n+1$的一个原根。

于是问题就转化为找到一个小于x的最大原根，直接暴力即可。

需要注意的是当$n=1$时任意大于1的b均为可行解，在这里需要特殊判断一下。

对于这个结论的证明略为繁琐，可以参考维基百科：https://en.wikipedia.org/wiki/Cyclic_number

1.31.4 时空复杂度

时间复杂度：$O(k \log_2 n)$其中k为常数，即找到原根所需要的次数。

空间复杂度：$O(\log_2 n)$

1.32 Codeforces 338D GCD Table

1.32.1 题目大意

有个$n*m$的表格，其中第i行第j列的数字为i与j的最大公约数。

现在给出一个长度为q的序列P，问P是否在这个矩阵中出现过。

1.32.2 数据范围

$1 \le n, m, P_i \le 10^{12}$，$1 \le q \le 10000$

1.32.3 题解

我们假设这个数字出现在(x, y)。

首先如果一个数与连续q个数的最大公约数分别为P_i，那么对于所有的i必然满足$P_i|x$，则x最小为P中所有的数的最小公倍数。

在求出x最小为多少之后，我们来枚举x的每个约数d。在相邻的d个数字中，必然有正好一个数为d 的倍数，且当k为d的倍数时，$k+d$与$k-d$必然同时也是d的倍数。这样即可排除当m为无限大时不可能存在的情况。

接下来我们需要求的即为y的最小值。我们将x质因子拆分，对于每一个p^k，我们找到在P中第一个可以整除p^k的数P_i，则可以列出多个类似于$y+i-1 \equiv 0 \pmod p)^k$的等式，最后用中国剩余定理合并即可求出最小的$y$。

1.32.4 时空复杂度

时间复杂度：$O(q \cdot d(n))$

空间复杂度：$O(q + d(n))$其中$d(n)$代表n的约数个数

1.33 Codeforces 266E More Queries to Array...

1.33.1 题目大意

给定一个长度为n的数列a，现在对其进行两种操作：

1. 给出l, r, x，将所有满足$l \le i \le r$的a_i修改为x。
2. 给出l, r, k，求$\sum_{i=l}^{r} a_i \cdot (i-l+1)^k$的值。

输出所有询问的结果，答案对10^9+7取模。

1.33.2 数据范围

$1 \le n \le 10^5$，$1 \le l \le r \le n$，$0 \le a_i, x \le 10^9$，$0 \le k \le 5$

1.33.3 题解

设$F(x, d, k) = a_x \cdot (x-d)^k$，则$F(x, d, k)$即为一个关于$x$的多项式，我们需要求的答案即为：

$$\sum_{i=l}^{r} F(i, l+1, k)$$

注意到$F(x,d,k)$每项系数除了a_x之外所有都是完全相同的，就$k=2$时举例：

$$F(x,d,2)=a_x\cdot(x-d)^2=a_xx^2-2da_xx+2d^2a_x$$

又因为$k\leq 5$，则我们对于每个x，维护a_x、a_xx、a_xx^2、a_xx^3、a_xx^4与a_xx^5，求和之后再将系数带进去即可。

对于每个值的维护我们可以同样可以用线段树，在区间修改打标记时我们使用求和公式或者预处理均可以。

1.33.4 时空复杂度

时间复杂度：$O(n+mk\log n)$

空间复杂度：$O(nk)$

1.34 Codeforces 332E Binary Key

1.34.1 题目大意

有一个长度为n的串s，和一个长度为m串t。

你需要构造出一个长度为n的01串q，进行以下操作：

将串q复制无数遍，取前$|s|$位作为新串p，将i从小到大枚举，若p_i为1，则取出s_i，要求这样取出的所有字符正好与t相同。

求字典序最小的串q。

1.34.2 数据范围

$1\leq n\leq 10^6$，$1\leq m\leq 200$，$1\leq k\leq 2000$。

1.34.3 题解

首先，我们可以算出当q的一位为1时，会使生成的串中增加哪些字符，用Hash存下来，记为Q。

我们枚举串q中一共有多少个1，由于q中一位上为1，会使生成的串长度增加$\lfloor\frac{n}{k}\rfloor$，或者$\lceil\frac{n}{k}\rceil$，所以枚举1的个数范围在$\lceil\frac{m}{\lfloor\frac{n}{k}\rfloor}\rceil$与$\lfloor\frac{m}{\lceil\frac{n}{k}\rceil}\rfloor$之间。

接下来，在枚举了有多少个1之后，我们可以确定t的哪些字符是来自于q中相同位置的1，同样的，我们将这些字符用Hash存下来，记为T。

现在我们只需要选出Q的一个子序列，使得其每位与T的当前一位相同即可。

又由于需要字典序最小，我们反过来贪心一遍即可，最终比较各个1的个数不同时哪个的字典序最小即可。

1.34.4 时空复杂度

时间复杂度：$O(mk)$

空间复杂度：$O(n+m+k)$

1.35 Codeforces 273E Dima and Game

1.35.1 题目大意

现在有一个游戏，在桌上一共有n对整数，(l_i,r_i)，玩家轮流操作。

玩家每次在操作时，可以任意选一个整数对(l_i,r_i)，然后将其改变为$(l_i+\lfloor\frac{r_i-l_i}{3}\rfloor, l_i+2\lfloor\frac{r_i-l_i}{3}\rfloor)$或者$(l_i, r_i-l_i+\lfloor\frac{r_i-l_i}{3}\rfloor)$。

不能操作的人输。

现在给定n,p，由你来决定初始状态下的(l_i,r_i)，要求$1\le l_i<r_i\le p$。问一共有多少种不同的初始状态使得先手胜利，答案对10^9+7取模。

两种状态不同当存在至少一个i使得第一个状态中第i个整数对与第二个状态中第i个整数对不同。

1.35.2 数据范围

$1\le n\le 1000$，$1\le p\le 10^9$

1.35.3 题解

首先我们可以发现，对于一个整数对(l,r)，实际上它对答案有用的只有它的长度，即若它的长度为q，则可以把它改成$\lfloor\frac{q}{3}\rfloor$或$q-\lfloor\frac{q}{3}\rfloor$。

接下来，我们可以直接设F_i代表长度为i的数对的SG值，则转移方程为：

$$F_i = mex(F_{\lfloor\frac{i}{3}\rfloor}, F_{i-\lfloor\frac{i}{3}\rfloor})$$

但是直接这样做是$O(p)$的，时间上过不了。

实际上我们可以将所有相邻的SG值相等的位置并在一起，当对于两个整数x, y，若$\lfloor \frac{x}{3} \rfloor$与$\lfloor \frac{y}{3} \rfloor$在同一个区间，且$x - \lfloor \frac{x}{3} \rfloor$与$y - \lfloor \frac{y}{3} \rfloor$在同一个区间时，$F_x$与$F_y$必然是相等的。

所以我们每次可以算出新的区间为多少，这个区间内的SG值是多少，排在队列末尾即可。这样每次增加的区间长度约为原本的区间长度*3，所以这样的区间个数是不会超过$O(log_3 p)$的。

在算出了所有的SG值之后，我们就可以开始算答案了，DP计算答案时只需要设$G_{i,j}$代表到了第i个区间，当前的SG值为j。注意在这题中的SG值是不会超过2的，所以j最大不超过3。转移时枚举这一个位置放的数对SG值为多少即可，这一步的时间为$O(n)$。

1.35.4 时空复杂度

时间复杂度：$O(log_3 p + n)$

空间复杂度：$O(log_3 p + n)$

1.36 Codeforces 333C Lucky Tickets

1.36.1 题目大意

给定一个k，要求找m个不同的八位数（允许有前导0），要求每个八位数均存在一种在其中添加一些运算符的方法，使得运算结果等于k。

1.36.2 数据范围

$0 \leq k \leq 10000$，$1 \leq m \leq 3 * 10^5$

1.36.3 题解

我们可以以一种特殊的方法来构造解。

注意到$k \leq 10000$，所以我们可以保留前4位作为一个完整的整数t，将后面4位作为一个算式。

可以发现，当后面四位数确定时，平均能得到60个不同的运算结果。假设我们能得到的结果为q，我们只需要将t赋为$k - q$或$k + q$即可。

1.36.4 时空复杂度

时间复杂度：?

空间复杂度：$O(m)$

1.37 Codeforces 240F Torcoder

1.37.1 题目大意

给出一个长度为n的字符串s，现在你进行m次操作，每次操作选择一个区间$[l_i, r_i]$，将$[l_i, r_i]$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。

输出经过m次操作后的字符串。

1.37.2 数据范围

$1 \leq n, m \leq 10^5$

1.37.3 题解

观察每次操作，可以发现，若我们求出了区间$[l_i, r_i]$中每种字符的数量，那么我们可以直接判断是否可能重排成为回文串，若这段中个数为奇数的字符数超过了1，那么这段必然不会被重排成回文串。

接下来我们观察排列之后的字符串会是什么样的。对于它的前半段，必然是按照字符序从小到大依次排列过来，若有字符的个数为奇数，那么我们将它排在$[l_i, r_i]$的正中间。

我们可以用线段树来维护这个操作，即维护每个区间中每种字符各有多少个，修改时使用线段树的区间修改即可。由于不同的字符最多有26个，总修改次数不超过$O(26)$。

1.37.4 时空复杂度

时间复杂度：$O(26^2 m \log_2 n + n)$

空间复杂度：$O(26n)$

1.38 Codeforces 305E Playing with String

1.38.1 题目大意

两个人在玩撕纸游戏。

初始只有一张纸条，上面写了个长度为n的字符串s。

当一个人进行操作的时候，他会进行以下三步：

1. 选择一张纸，我们记这张纸上的字符串为t。

2. 选择一个$i(1 < i < |t|)$使得存在一个正整数$k(0 < i - k, i + k \le |t|)$满足$t_{i-1} = t_{i+1}, t_{i-2} = t_{i+2}, ..., t_{i-k} = t_{i+k}$。

3. 把这张纸按照i的位置分为三份$t_1 \sim t_{i-1}, t_i$以及$t_{i+1} \sim t_{|t|}$。

1.38.2 数据范围

$1 \le n \le 5000$

1.38.3 题解

对于题目中步骤2所说的存在一个正整数k满足$t_{i-1} = t_{i+1}, t_{i-2} = t_{i+2}, ..., t_{i-k} = t_{i+k}$，实际上我们可以将$k$取到最小，即当$t_{i-1} = t_{i+1}$时，就可以将当前的$i$作为拆分的中点。

接下来，我们将所有在第一步可以作为拆分中点的位置标记出来。假设我们一次拆分的点为x，可以发现，会影响到的点只有$x + 1$与$x - 1$，所以只有当两个可拆分的点连在一起的时候才有可能相互影响。所以我们将所有连在一段的可拆分点给截取出来，那么对于我们来说实际上有用的即为这段的长度了。

设F_i代表连续的一段长度为i的可拆分点的SG值为F_i，则转移时我们只需枚举当前拆分的为哪一个位置，分成两段之后再将其SG值异或起来取mex即可。

1.38.4 时空复杂度

时间复杂度：$O(n^2)$

空间复杂度：$O(n)$

1.39 Codeforces 235D Graph Game

1.39.1 题目大意

我们定义对于图G的运算$Solve(G)$为以下几步操作：

1. 变量$TotalCost$加上G的大小。
2. 在G中随机选择一个点x。
3. 从G中删除x，图中剩下了许多连通块S_i。
4. 分治处理所有的$Solve(S_i)$。

现在有一个n个点n条边的无向连通图E，求在进行$Solve(E)$之后变量$TotalCost$期望增加了多少。

1.39.2 数据范围

$3 \le n \le 3000$，图保证无重边与自环。

1.39.3 题解

首先我们先考虑一棵树的情况，由于期望的线性叠加，我们分别考虑对于每个点对(x, y)对答案的贡献为多少。

当点对(x, y)对答案有贡献时分为两种情况：1、在x被删除时，x与y连通；2、在y被删除时，x 与y连通。

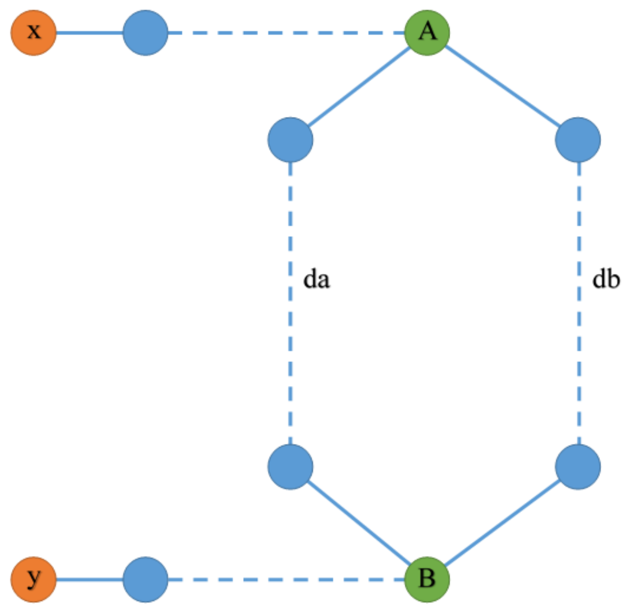
说白了就是x或y是从x到y的链上第一个被删掉的点。

实际上，我们只需要求出x或y是从x到y的链上第一个被删掉的点的概率为多少即可。假设从x到y 的链上共有$dist_{x,y}$个点，则x或y是第一个被删掉的点的概率为$\frac{2}{dist_{x,y}}$。所以对于一棵树的情况，$TotalCost$的期望值即为$\sum_{x=1}^{n}\sum_{y=x}^{n}\frac{2}{dist_{x,y}}$。

接下来我们来考虑原题面中存在一个环的情况，同样的我们考虑所有的点对(x, y)，同样的，当点对(x, y)对答案有贡献时分为两种情况：1、在x被删除时，x与y连通；2、在y被删除时，x与y 连通。

若从x到y的路径不经过环，则和之前在一棵树上的情况相同。

若从x到y的路径经过环，如下图：



注：da、db不包括点A、B。

设包括整个环一共有T个点，我们依次分析各种情况：

1. 如果第一次删掉了链da上的点，则变为一条$T-da$的链；
2. 如果第一次删掉了链db上的点，则变为一套$T-db$的链；
3. 如果第一次删掉了x或y，则此时对答案的贡献为1；
4. 如果第一次删掉了其他点，则此时对答案的贡献为0。

则对答案有贡献的总概率为：

$$\frac{2}{T}+\frac{da}{T}\cdot\frac{2}{T-da}+\frac{db}{T}\cdot\frac{2}{T-db}$$

接下来，我们枚举所有点对，计算总期望即可。

1.39.4 时空复杂度

时间复杂度：$O(n^2)$
空间复杂度：$O(n)$

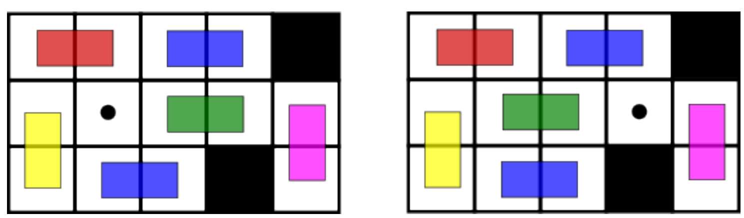
1.40 Codeforces 342D Xenia and Dominoes

1.40.1 题目大意

给出一个3*n的网格，其中有一些格子是坏的，并且有一个特殊的空格子。

现在要求你用多米诺骨牌铺满这个网格中除了坏了和特殊的空格子之外所有格子，且必须有多米诺骨牌可以移动到特殊的格子。

多米诺的移动是在保证拼图合法的情况下，把一个多米诺骨牌移到空格子里。横向的多米诺骨牌只能横向移动，纵向的多米诺骨牌只能纵向移动。如下图即为一次合法的移动：



问一共有多少种不同的放置方案。

1.40.2 数据范围

$3 \le n \le 10000$

1.40.3 题解

对于传统的多米诺骨牌放置方案，我们可以直接使用状态压缩DP来解决。

但是这题有一个特殊的要求，即为必须有一个骨牌的短边对着一个特殊点。我们可以枚举在每个方向上是否有一个对应的多米诺骨牌，套用容斥原理即可。

1.40.4 时空复杂度

时间复杂度：$O(2^3n)$

空间复杂度：$O(2^3)$

1.41 Codeforces 301C Yaroslav and Algorithm

1.41.1 题目大意

现在要你用一个奇怪的语言来描写一个算法。

1. 这个算法接受一个字符串作为输入。我们设这个输入字符串为a。
2. 这个算法由一些命令组成。i号命令的形式为”$s_i >> w_i$”或”$s_i <> w_i$”，其中s_i和w_i 是长度不超过7的字符串（可以为空），由数字或字符”?”组成。
3. 这个算法每次寻找一个编号最小的命令i，使得s_i是a的子串。如果没有找到这样的命令，那么整个算法终止。
4. 设找到的命令编号为k。在字符串a中，s_k第一次出现的位置会被w_k替换。如果这个命令形如”s_k¿¿w_k”，那么这个算法继续执行（即回到第3步）。否则，算法终止。
5. 在算法终止时输出字符串a的值。

现在给你n个正整数，要求构造出一个算法，对于每个数字，这个算法把它当作一个字符串读入，输出的字符串刚好为这个数字+1之后的字符串。

额外的，对你构造出的算法还有2个要求：

1. 命令条数不能超过50；
2. 算法对于每个输入都执行不超过200步。

1.41.2 数据范围

$1 \le n \le 100$，每个数的大小在$10^2$5之内。

1.41.3 题解

对于本题，实际上有一个通用的算法，和平时我们所使用高精度+1的方法差不多。

首先，我们用”?”来当作指针，找到最后一位所在的位置，即：?0 >> 0?,?1 >> 1?,...,?9 >> 9?，当这10 条指令重复执行完毕之后，”?”所在的位置即为最后一位了。

接下来，我们给最后一位+1，若最后一位为0 $\sim$ 8，则可以结束程序，若最后一位为9，则我们还需要进位，可以看作最后一位往左移动了一次，再继续进行+1的操作，注意在此要与之前的指针区分开，所以我们在完成第一步之后需要将”?”修改为”??”。

最终算法如下：

```
0??<>1
1??<>2
2??<>3
3??<>4
4??<>5
5??<>6
6??<>7
7??<>8
8??<>9
9??>>??0
??0<>10
?0>>0?
?1>>1?
?2>>2?
?3>>3?
?4>>4?
?5>>5?
?6>>6?
?7>>7?
?8>>8?
?9>>9?
?>>??
>>?
```

1.41.4 时空复杂度

时间复杂度：$O(1)$

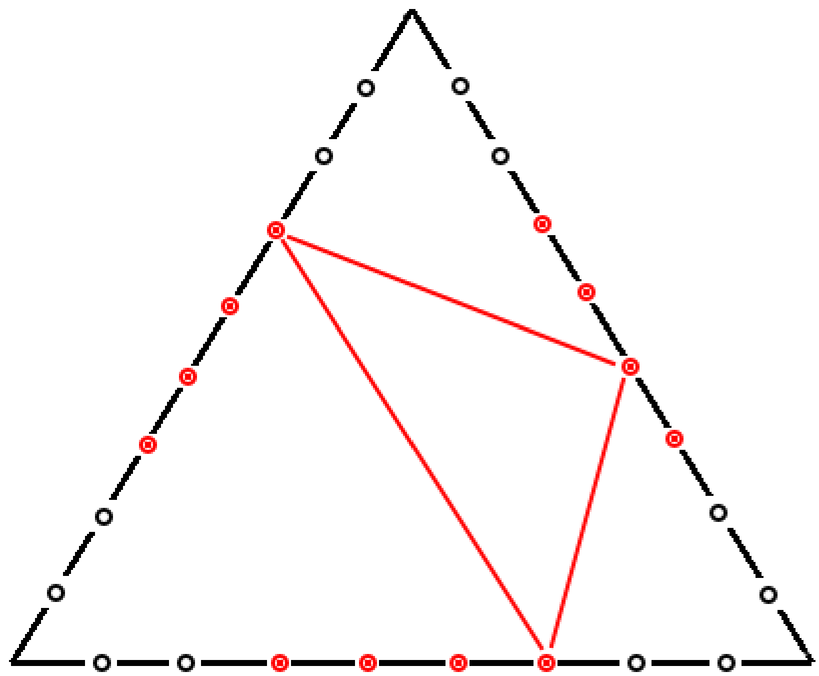
空间复杂度：$O(1)$

1.42 Codeforces 309D Tennis Rackets

1.42.1 题目大意

在一个边长为$n+1$的正三角形上，每条边上各有n个点，将三条边各分为了n等份。

现在要求你从三条边上各选出一个点，使这三个点构成一个钝角三角形，而且所有被选中的点与正三角形的三个角的距离必须大于m。如下图即为一个$n=8, m=2$的可行状态：



问一共有多少种不同的选择方法。

1.42.2 数据范围

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

1.42.3 题解

枚举两个点，之后第三个点的区间是有单调性的。
然后貌似没有更优的做法，所以直接卡常数即可。

1.42.4 时空复杂度

时间复杂度：$O(n^2)$
空间复杂度：$O(n)$

2　codeforces第二部分

2.1　Codeforces 309B Context Advertising

2.1.1　题目大意

现在有n个单词，你需要取出连续的一段单词，使得这些单词可以按顺序的放到一个$r*c$的广告里面去，使得没有一个单词在广告中被分在了两行。

求你最多能够取出多少个单词。

2.1.2　数据范围

$1 \le n, r, c \le 10^6$，$r*c \le 10^6$

2.1.3　题解

首先我们可以贪心求出从每个单词开始，长度不超过c的一段能够取到哪里。

然后我们需要求的即为从每个点向后跳r步能够跳到哪里，对于这个我们可以使用快速幂来求或者直接倍增。

2.1.4　时空复杂度

时间复杂度：$O(n\log_2 r)$

空间复杂度：$O(n)$

2.2　Codeforces 277D Google Code Jam

2.2.1　题目大意

比赛中有n个问题，每个问题分为两个部分：Easy和Hard。现在你有m的时间来解决问题，对于第i个问题，你可以花费$t1_i$的时间来完成Easy部分获得$s1_i$的分数，并且肯定能够得到，在完成了Easy部分之后，你可以花费$t2_i$的时间来完成Hard部分，但是有p_i的几率失败，如果成功了，你可以额外获得$s2_i$的分数，如果失败你只能获得Easy部分的分数。

你要求在花费不超过m个单位时间的前提下期望获得更高的分数，并且当期望分数最高时追求更低的期望罚时。

一场考试的罚时是指在最后一次成功解决问题的时间。

输出最高期望得分，和在最高期望得分前提下的最小罚时。

2.2.2 数据范围

$1 \le n \le 1000$，$1 \le t, t1_i, t2_i \le 1560$，$1 \le s1_i, s2_i \le 10^9$

2.2.3 题解

如果题目只要求最高期望得分，则我们可以直接DP，问题就在于现在有第二关键字——罚时。

如果我们已经确定了做题的顺序，那么同样的我们也可以直接DP，所以现在的问题在于如何确定做题的顺序。

首先，我们可以优先做完所有需要做的Easy部分，因为Easy部分绝对会正确，所以在做完时必然会重置罚时，所以我们现在只需要分开计算Hard部分的期望罚时，之后再加上Easy部分的总用时即为总期望罚时，所以对于所有Easy的顺序也是没有影响的。

考虑Hard部分的顺序，我们猜测它是按照某个偏序的关系来排序的，我们来观察相邻两个题目i,$i+1$的关系，可以直接算出当i在$i+1$之前期望罚时为多少，当$i+1$在i之前期望罚时为多少，比较即可求出它们的优先级，接下来我们只需要套用冒泡排序的过程即可确定以这个关系作为偏序关系即可求出最优顺序。

事实上这题与NOIP2012的day2T2国王游戏十分类似，证明可以相互类比。

在解决问题时需要注意精度上的误差，在C++中要用long double才能通过。

2.2.4 时空复杂度

时间复杂度：$O(nm)$

空间复杂度：$O(m)$

2.3 Codeforces 341E Candies Game

2.3.1 题目大意

有n个糖果盒，每个糖果盒内有a_i个糖果，现在你可以进行不超过10^6次操作，每次操作选择一对整数(x, y) 满足$a_x \leq a_y$，将第y个糖果盒中取出a_x个糖果并放加在第x个糖果盒中。显然，当$a_x = a_y$ 时第y个糖果盒内的糖果就被拿完了。

你需要给出一个方案，使得在操作完成之后，有且仅有2个糖果盒内还剩余糖果。

2.3.2 数据范围

$3 \leq n \leq 1000$，$0 \leq a_i \leq 10^6$，$\sum_{i=1}^{n} a_i \leq 10^6$

2.3.3 题解

首先我们先排除掉无解的情况，当只有一个糖果盒里面有糖果时是必然无解的，否则可以经过构造使其必然有解。

对于三个还剩余糖果的糖果盒，我们设它们分别包含x, y, z个糖果($x \leq y \leq z$)，则我们如果能够将y变为$y \bmod x$，则我们可以通过类似于求最大公约数的方法在$\log_2 x$步内将其变为0。

我们设$y = kx + r$，其中k为整数，且$0 \leq r < x$。

可以发现，如果我们将x与其他数操作了i次之后，x会变为$2^i x$，接下来，我们可以直接将k二进制拆分，当k的当前一位为1时，我们进行操作(x, y)，否则我们进行操作(x, z)，由于$z \geq y$，所以当$x \leq y$时必然有$z \geq x$。

这样我们即可将y变为$y \bmod x$，接下来，我们只需要将x, y, z重排序，继续进行操作即可。

2.3.4 时空复杂度

时间复杂度：$O(n {\log_2}^2 S)$ 其中$S = \sum_{i=1}^{n} a_i$

空间复杂度：$O(n)$

2.4 Codeforces 323B Tournament-graph

2.4.1 题目大意

你需要构造一个大小为N的竞赛图，使得从任何一个点x出发到任意一个点y的最短距离不超过2。

竞赛图的定义为对于所有的点对x, y存在且仅存在一条$x \Rightarrow y$或$y \Rightarrow x$的边。

2.4.2 数据范围

$3 \leq N \leq 1000$

2.4.3 题解

首先当$N = 4$时，该图无解。

否则，当N为奇数时，我们可以如此连边：枚举所有的点i，将其与所有的点$(i + j - 1) \bmod N + 1(j \leq \frac{N}{2})$ 连边，可以发现这样所有的点必然可以通过不超过两步走到。

当N为偶数时，我们可以先建出一个$N - 1$个点符合条件的图，之后将第N个点与所有奇数点连边，将所有偶数点与第N个点连边即可。

另外一种做法是当我们构造出了大小为$N - 2$满足条件的图之后，可以添加两个点x, y，将x与$N - 2$个点连边，将$N - 2$个点与y连边，之后将y与x连边。容易证明这样也是合法的。

2.4.4 时空复杂度

时间复杂度：$O(n^2)$

空间复杂度：$O(n^2)$

2.5 Codeforces 248E Piglet's Birthday

2.5.1 题目大意

有n叠纸，初始第i叠纸中有a_i张纸，现在进行m次操作，每次操作从x_i中随机选出k_i张纸，将它们做上记号并放在y_i中。

问在每次操作后期望有多少叠纸里面所有的纸都被做上了记号，注意若一叠纸为空则它也必须算入答案。

2.5.2 数据范围

$1 \le n, m \le 10^5$，$0 \le a_i \le 100$，$1 \le x_i, y_i \le n$，$1 \le k_i \le 5$

2.5.3 题解

我们设$F_{i,j}$代表在当前第i叠纸内，有j张纸没有被做上记号的概率为$F_{i,j}$。

可以发现在每次操作之后，对于y_i内没有被做上记号的纸的数量是不会增加的，所以在任何时候j的值都不会超过a_i。

接下来，我们只需要在一次操作之后，枚举x_i中被取走了多少张尚未做上记号的纸，转移即可。

每次我们需要求的答案即为$\sum_{i=1}^{n} F_{i,0}$。

2.5.4 时空复杂度

时间复杂度：$O(m * a_i * k_i)$

空间复杂度：$O(n * a_i)$

2.6 Codeforces 266D BerDonalds

2.6.1 题目大意

有一个n个点m条边的无向带权连通图，现在你要从图中选一个位置(选的位置可以是某个点，也可以是某条边上的任意一点)，使得它到最远的点距离最近。

输出最短的最远距离。

2.6.2 数据范围

$2 \le n \le 200$，$n - 1 \le m \le \frac{n(n-1)}{2}$

2.6.3 题解

我们枚举这个点在哪条边上。

假设当前枚举的边为a, b，长度为L，我们距离点a的距离为x，则当前到达距离最远的点的距离即为：$maxmin(dist_{i,a} + x, dist_{i,b} + L - x)$，其中$dist_{i,j}$指从$i$到$j$的最短路。

可以发现，对于每个点到当前位置的距离与x的关系都为一个分段函数，即当x取$\frac{dist_{i,b}+L-disti,a}{2}$时，函数的值最大，当$x$减小或增大时，函数的值随之减小。

我们可以据此列出n个函数，将它们用$O(n\log_2 n)$的时间取max并在一起，可以发现当所有的函数合并之后，分段的个数也是不超过$O(n)$的，所以我们$O(n)$枚举每个极小值即可求出在这条边上的最小值。

接下来，我们对于每条边求出的答案取最小值即可。

2.6.4 时空复杂度

时间复杂度：$O(n^3\log_2 n)$

空间复杂度：$O(n^2)$

2.7 Codeforces 311C3 Fetch the Treasure

2.7.1 题目大意

有一个长度为h的数组a，除了其中的n个位置有正整数值之外其余的值均为0。

现在你从1号点出发，你有一个能力集合S，若当前你所在的位置为x，你可以走到y，当且仅当在S中存在$y-x$这个数。

初始条件下S中仅有一个数k，现在进行m次操作：

1. 在S中增加一个数x；
2. 将数组中x位上的值减少y，y必然小于等于原本的a_x；
3. 你进行了一次取数，取数是指你从1号点出发，从可以到达的点中选出最大的一个a_t，若有多个a_t相同，则取t最小的一个，取走a_t，之后数组的第t位为0。

输出你所有取走的值。

2.7.2 数据范围

$1\le h\le 10^{18}$，$1\le n,m\le 10^5$，$1\le k\le 10000$，$1\le x\le h$，操作1的个数不超过10

2.7.3 题解

对于一个点x，如果我们能够到达x，则我们必然也可以到达所有的$x+ik$，其中i为自然数。

所以对于所有的i，我们用d_i记录能到达的$x \equiv i \pmod k$中，最小的x为多少，所有大于d_i且与d_i同余的点x必然也可以到达。

注意到操作1最多只有10次，所以我们每次在有操作1时，我们暴力更新所有的d_i即可。

接下来，我们需要做的只有从能到的点中选出最大值，用优先队列维护即可。

2.7.4 时空复杂度

时间复杂度：$O((n+m)\log_2 n + 10k\log_2 k)$

空间复杂度：$O(n+k)$

2.8 Codeforces 241F Race

2.8.1 题目大意

有一个$n*m$的网格图，每个格子都代表一个街区。它上面有不超过26个路口，每个路口占一个街区，位于道路的交汇处。路口与路口之间有道路连接，所有道路的宽度都为1个街区，并且道路都是水平的或竖直的，没有两条道路或两个路口是相邻的。地图上除了路口和道路的所有的地方都是建筑物。

现在你知道从某一个街区到其相邻的街区所花费的时间，同时，从交叉路口需要花费1分钟来到达相邻的街区。

告诉你一条路径的初始与结束位置，同时告诉你这条路径途中依次经过的q个交叉路口，问在第k个时刻所在的位置。

2.8.2 数据范围

$3 \le n, m \le 100$，$1 \le k \le 10^5$，$1 \le q \le 1000$

2.8.3 题解

注意到道路都是水平的或竖直的，且没有两条道路或两个路口是相邻的，那么我们从一个路口到达第二个路口必然走的路径为一条直线。

那么我们先找到所有的路口所在的位置，然后从起点直接模拟走路的过程即可。

2.8.4 时空复杂度

时间复杂度：$O(nm+k+q)$

空间复杂度：$O(nm+q)$

2.9 Codeforces 316D3 PE Lesson

2.9.1 题目大意

现在有一个大小为n，初始状态为$\{1,2,3,...,n\}$的排列P，给出一个长度为n，的正整数数组Deg。定义一种操作，每次选中两个不同的位置x和y，交换P_x和P_y，并且将Deg_x 和Deg_y 减一。现在你可以进行任意多次操作，求当数组Deg的值全部大于0时，一共会产生多少种不同的排列P，答案对10^9+7取模。

排列C与排列D不同当存在至少一个i使得$C_i \neq D_i$。

2.9.2 数据范围

$1 \le n \le 10^6$，$1 \le Deg_i \le 2$

2.9.3 题解

我们将最终生成的排列P用一个置换表示出来，一个环中最多只能存在两个Deg为1的点，否则该情况必然无法到达。

所以实际上数组Deg中各项的值对于我们来说是没有意义的，实际上有用的只有Deg中1的个数与2的个数。

我们设$F_{i,j}$代表用了i个Deg值为1的点，j个Deg值为2的点的方案数，递推即为枚举当前环用了多少个Deg值为2的点，用了多少个Deg值为1的点，即可得出$O(n^3)$的做法。

可以发现，如果优先确定所有Deg值为1的点如何放置，在最后再插入所有Deg值为2的点也是可以的，所以我们优先计算出Deg值为1的点有多少种不同的组合方法，之后计算Deg值为2的点共有多少个不同的插入位置，相乘即可。最终公式即为：

$$\frac{F(d1)*n!}{d1!}$$

其中$d1$代表有多少个点的Deg值为1，$F(i)$的递推式为：

$$F(i) = F(i-1) + (i-1)F(i-2)$$

2.9.4 时空复杂度

时间复杂度：$O(n)$
空间复杂度：$O(1)$

2.10 Codeforces 323C Two permutations

2.10.1 题目大意

现在有两个长度为n的排列，和m个询问，每次询问在第一个排列中位置在$[l1, r1]$，且在第二个排列中位置在$[l2, r2]$中的数的数量。

对于每个询问输出结果。

2.10.2 数据范围

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

2.10.3 题解

一道很裸的可持久化线段树裸题，我们对于第一个数列维护线段树，将第二个排列中的值不断插入线段树中，并记录每个历史版本。

实际上可以看作求出在第二个排列的$r2$之前，存在于第一个排列中$[l1, r1]$的数有多少个，减去在第二个排列的$l2-1$之前，存在于第一个排列中$[l1, r1]$的数有多少个即为答案。

2.10.4 时空复杂度

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

2.11 Codeforces 338E Optimize!

2.11.1 题目大意

题目中给出了一段代码，请你来优化它：

```
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
```

我们将这段代码用文字表述即为：

现在有一个长度为n的数组a，有一个长度为len的数组b，问a有多少个连续且长度为len的子数组s，使得可以将b中的数字与s中的数字一一配对，每对中的两个数字之和均大于h。

2.11.2 数据范围

$1 \le len \le n \le 1.5 * 10^5$，$1 \le a_i, b_i, h \le 10^9$

2.11.3 题解

我们将所有a中的数字a_i变为$h - a_i$，则原问题变为：问a有多少个连续且长度为len的子数组s，使得可以将b中的数字与s中的数字一一配对，每对中来自b的数字大于来自s的数字。

我们观察配对的过程，其中必然为将b排序，将s排序之后，再从小到大一一配对。所以问题可以再次转化为：对于一个数组b中第i大的数，在s中是否有超过i个数比它小。

接下来，我们就可以枚举区间s，每次去掉s中最左边的一个数，加上s右边的一个数，判断这个数列是否合法即可。

2.11.4 时空复杂度

时间复杂度：$O(n\log_2 len)$

空间复杂度：$O(n+len)$

2.12 Codeforces 285E Positions in Permutations

2.12.1 题目大意

定义一个排列P的权值为有多少个i满足$|P_i-i|=1$。

求共有多少个长度为n且权值为k的排列。

2.12.2 数据范围

$1\le n\le 1000$，$0\le k\le n$

2.12.3 题解

将排列变成二分图匹配，那么我们从前往后思考左边和右边的第i个分别匹配哪一个。

匹配的过程可以看作在原来大小为$i-1$的匹配中新增加两个点，先将左边的i与右边的i进行匹配，然后找另外一个匹配x,y来交换，即将左边的i匹配右边的y，左边的x匹配右边的i。实际上和新的状态有关的只有左右两边的第i个分别与哪个数字相连，这样我们新设一位状态代表左右两边的第i个点分别与哪个位置匹配即可。

我们设$F_{i,j,p}$代表到了第i个点，已经有j个差值为1的匹配，且左边的第i个点与右边的第i个点连线方式为p时，总共有多少种不同的匹配方法。

那么转移时我们只需要枚举新增加的一个匹配不作任何交换/与左边的第i个点所在的匹配交换/与右边的第i个点所在的匹配交换/与一般的匹配进行交换/与差值为1的匹配进行交换即可。

2.12.4 时空复杂度

时间复杂度：$O(n^2)$
空间复杂度：$O(n^2)$

2.13 Codeforces 238D Tape Programming

2.13.1 题目大意

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。

现在我们有一个长度为n，由"$<$","$>$"和数字构成的串s，你需要回答q个询问。每个询问会给你两个数l, r，如果把子串$s[l..r]$取出来看作一个单独的程序，问你每个数字会被输出多少次。

2.13.2 数据范围

$1 \le n, q \le 10^5$

2.13.3 题解

我们先将整个字符串s当作一个程序运行一遍，如果在程序运行结束之后有节点没有遍历到，则在没有被遍历到的节点上继续运行，假设我们访问每个位置的顺序为P。

可以发现，查询字串时到达节点的顺序，必然为P一段连续的子串，因为当我们在以P的顺序遍历，第一次遍历第l个字符时，第l个字符之后的所有字符必然同样没有被遍历到。

所以我们只需要找到在P中第一次遍历到第l个字符的位置，即为序列的起点。而当我们走出区间$[l, r]$ 时，即为序列的终点。走出去的方式有两种：1、第一次走到$r + 1$；2、第二次走到$l - 1$。

所以我们只需要记录第一次与第二次走到一个节点时，所输出的数的前缀和即可。

2.13.4 时空复杂度

时间复杂度：$O(10(n + m))$
空间复杂度：$O(10n)$

2.14 Codeforces 331C3 The Great Julya Calendar

2.14.1 题目大意

现在有一个十进制数n，你需要用最少次数的操作将其变为0。
对一个数字x的操作，你可以从它的各位中选出一个数d，将x减去d。
输出最少的操作次数。

2.14.2 数据范围

$0 \le n \le 10^{18}$

2.14.3 题解

首先我们可以发现这道题中，我们所需要操作的次数随着n增大而增大，是一个单调函数。证明比较简单，在这里省去。

接下来，我们就可以贪心的每次减去每位中最大的一个数了，但这样的时间是$O(n)$的。为了优化，我们考虑数位DP。

$F_{x,i,w}$代表当前数的末尾i位为x，且在末i位之前的所有位数中的最大值为w时，值减小至少x的操作次数最小为多少次。

转移时可以将x的后$i - 1$位取出来，不断减小直到减小的总值大于等于x时即可。

这样的总状态数是$O(\log_2 n)$的，所以我们开一个Hash表存下来即可。

2.14.4 时空复杂度

时间复杂度：$O(\log_2 n)$
空间复杂度：$O(\log_2 n)$

2.15 Codeforces 251D Two Sets

2.15.1 题目大意

现在有n个数，你需要将其分为两个集合，假设第一个集合内所有的数异或起来为$x1$，第二个集合内所有的数异或起来为$x2$，要求$x1 + x2$最大，在$x1 + x2$最大的前提下还要求$x1$最小。

求一种合法的方案。

2.15.2 数据范围

$n \leq 10^5$

2.15.3 题解

我们记录所有数的异或和为K，则无论如何划分这n个数，都会有$x1\ xor\ x2 = K$，所以当我们确定了$x1$的值之后，$x2$的值是固定的。

接下来，我们对于K的每一位进行分析：

假设K的第i位为1，则在$x1$与$x2$之间，第i位为1的最多只有1个，对答案的贡献固定为2^i。

假设K的第i位为0，则有$x1$的第i位必然和$x2$的第i位相同，则要么同时为1，要么同时为0，对答案的贡献取决于$x1$第i位的值，要么为$2^(i+1)$，要么为0。

所以我们可以确定$x1$各个位数之间的优先级，如果K的第i位为0，那么它必然比K的第j位为1的优先级更高，否则直接比较i和j的大小。

接下来，我们可以使用和n个数使得它们异或和最大相同的方法——高斯消元，在确定优先级之后贪心求出$x1$的大小即可。

2.15.4 时空复杂度

时间复杂度：$O(n \log_2 n)$
空间复杂度：$O(n)$

2.16 Codeforces 351D Jeff and Removing Periods

2.16.1 题目大意

你能对一个长度为m的数列a执行下列操作：

1. 选择三个整数$v, t, k(1 \le v, t \le n$，$0 \le k$且$v + tk \le m)$，要求满足$a_v = a_{v+t} = a_{v+2t} = ... = a_{v+tk}$。
2. 删除$a_v, a_{v+t}, a_{v+2t}, ..., a_{v+tk}$这$k+1$个数。
3. 重新排列剩余的$n-k-1$个数

定义一个数列a的权值为将其全部删除所需要的最少操作次数。

现在给定一个长度为n的数列b，有q次询问，每次询问数列b在$[l, r]$的子数列的权值为多少。

输出每次询问的结果。

2.16.2 数据范围

$1 \le n, q, b_i \le 10^5$，$1 \le l \le r \le n$

2.16.3 题解

可以发现，在我们对于一个序列执行了一次操作之后，可以重新排列剩余所有的数，所以我们在进行一次操作之后只需要将所有相同的数字排在一起，之后每次操作可删掉一种数字，总权值即为剩余的颜色数+1。

所以我们现在需要处理的只有两件事：

1. 求一个区间内不同数的个数
2. 求一个区间内是否有一种颜色存在的位置为等差数列

对于第一个问题，我们预处理出lef_i代表第i个点左边与其颜色相同的点最近在哪个位置，之后再离线处理即可。

对于第二个问题，我们同样的可以预处理出P_i代表第i个点往左可以扩展多少使其仍为等差数列，同样离线处理即可。

2.16.4 时空复杂度

时间复杂度：$O((n+q)\log_2 n)$

空间复杂度：$O(n+q)$

2.17 Codeforces 293E Close Vertices

2.17.1 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。

2.17.2 数据范围

$1 \le n \le 10^5$，$1 \le L \le n$，$0 \le W \le 10^9$，所有的边权均为不超过10000的自然数

2.17.3 题解

裸的树分治，在枚举重心时，求出经过它且长度小于L，权重小于W的路径条数，减去同一棵子树内的情况即可。

2.17.4 时空复杂度

时间复杂度：$O(n{\log_2}^2 n)$

空间复杂度：$O(n)$

2.18 Codeforces 241E Flights

2.18.1 题目大意

有一个n个点m条边的有向无环图，每条边的边权初始为1。

现在你要将一些边的边权修改为2，使得每条从点1到点n的路径长度相同。

求一组合法的方案。

2.18.2 数据范围

$2 \le n \le 1000$，$1 \le m \le 5000$

2.18.3 题解

如果从1到n的每条路径长度都是一样的，那么假设有一个x使得1到x有一条路径，x到n有一条路径，那么从1到x的所有路径长度必然也是一样的。

所以我们可以看作一条边的边权在$1 \sim 2$之间取值，即对于每条出现在1到n路径中的边$x \Rightarrow y$，都有：$1 \leq dist_y - dist_x \leq 2$。

所以我们可以根据此条件建出差分约束系统，跑一边最短路即可，出现负环即为无解。

2.18.4 时空复杂度

时间复杂度：$mindist(n, m)$

空间复杂度：$O(n + m)$

2.19 Codeforces 268D Wall Bars

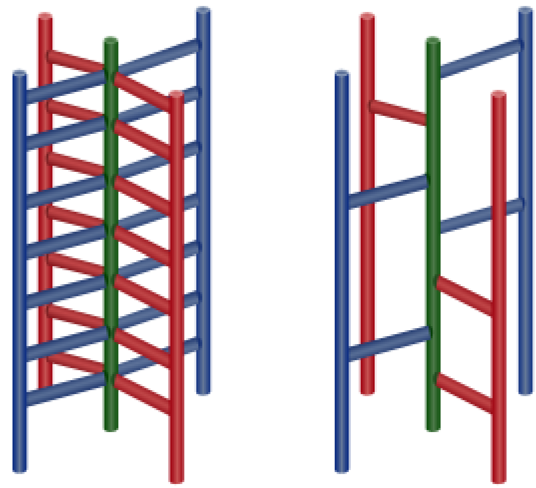
2.19.1 题目大意

有一种儿童游乐设施叫做Wall Bar，现在你需要设计一个特殊的Wall Bar，如下所述：

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$1, 2, 3, ..., n$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。一个从地面出发的孩子至少能到达一根高度在$n - h + 1, n - h + 2, n - h + 3, ..., n$ 的横杠。



左图为一个正常的Wall Bar，右图为一个特殊的Wall Bar

问一共有多少种不同的合法设计方案，答案对10^9+9取模。

2.19.2 数据范围

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

2.19.3 题解

我们思考裸的$O(nh^4)$DP，我们设$F_{i,a,b,c,d}$代表到了高度为i的杆子，每个方向上最后放置的高度与当前高度差分别为a,b,c,d，转移即为枚举当前高度上的横杆向哪个方向即可。

由于当a,b,c,d大于h时，我们可以直接忽略，所以a,b,c,d均不超过h。

注意到我们最后一次放置了横杆之后，a,b,c,d中必然有一个值为0，所以我们可以省去一维状态，时间复杂度优化到$O(nh^3)$。

2.19.4 时空复杂度

时间复杂度：$O(nh^3)$

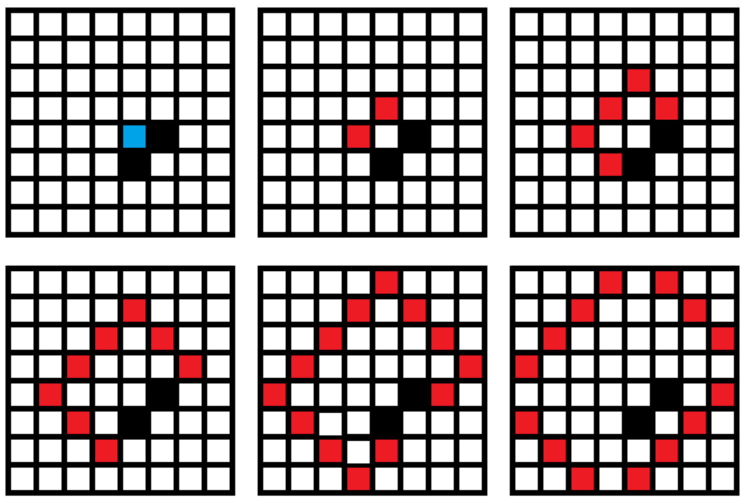
空间复杂度：$O(h^3)$

2.20 Codeforces 254D Rats

2.20.1 题目大意

现在有一个$n*m$的网格，其中有一些位置为障碍，有一些位置为老鼠，现在要求选出两个点，使得所有老鼠与最近的一个点的距离不能超过d。

一个点与另外一个点的距离定义为从一个点走到另一个点不经过障碍的所经过的最少的格子数。如下图所示：



依次指与蓝点距离为0,1,2,3,4,5的点

求一组合法的选点方案。

2.20.2 数据范围

$1 \le n, m \le 1000$，$1 \le d \le 8$

2.20.3 题解

首先我们第一个选的点必然与第一只老鼠的距离不超过d，我们枚举这个点的位置。

接下来，若第一个点没有覆盖所有的老鼠，那么我们找到一只没有被覆盖的老鼠，则第二个点与其距离同样也不超过d，即可枚举第二个点的位置，至此问题解决。

2.20.4 时空复杂度

时间复杂度：$O(d^6+n^2)$
空间复杂度：$O(n^2)$

2.21 Codeforces 269D Maximum Waterfall

2.21.1 题目大意

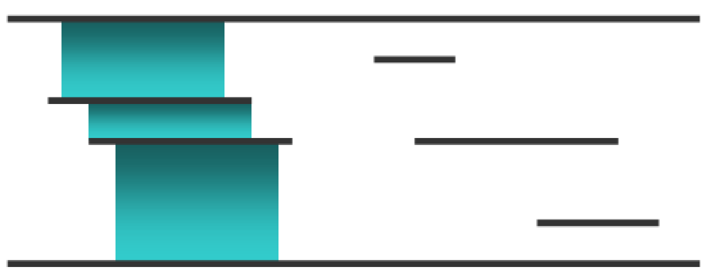
现在有一面墙，墙的高度是t，有n块水平板嵌在上面。每块水平板可以看作一条水平的线段，高度为h_i，从l_i到r_i。第i块水平板连接的平面上的(l_i,h_i)与(r_i,h_i)两点。墙的顶端可以看作一块在$(-10^9,t)$到$(10^9,t)$间的水平板。相似的，墙的底端可以看作一块在$(-10^9,0)$到$(10^9,0)$ 间的水平板。任意两块水平板之间没有公共点。

水可以从第i块板流向第j块板当且仅当：

1. $max(l_i,l_j)<min(r_i,r_j)$
2. $h_j<h_i$
3. 不存在$k(h_j<h_k<h_i)$使得i,k、k,j均满足之前的两个条件。

从i到j的流量为$min(r_i,r_j)-max(l_i,l_j)$，相当于二者的水平相交部分的长度。

现在我们需要建造一个瀑布。在这个瀑布中，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，如下图即为一个合法的方案：



求人造瀑布最大的流量可能值。

2.21.2　数据范围

$1 \le n \le 10^5$，$2 \le t \le 10^9$

2.21.3　题解

我们先思考有几种情况可以连边：区间的左边相交、区间的右边相交和区间完全被包含。

可以发现，对于每个区间，在左边相交与在右边相交的情况只有最多2个，而被包含总共也最多可能有2个。

所以所有的边的条数是不超过$O(n)$的。

假设我们已经连完边了，那么现在只需要进行一次拓扑图DP即可求出答案。所以现在的问题在于如何连边。

我们将所有的线段按照高度排序依次处理。我们维护一个set，保存当前x的某一区间是哪条线段。当我们处理到当前的线段时，我们首先对它x坐标内每一段连续相同的线段，查询它们之间是否可以连边。之后我们将这条线段x坐标的区间给覆盖。

这样每次插入一条线段时，若询问了x条线段，则必然会删掉至少$x-2$条线段，而只会增加最多2条线段，所以这样的总复杂度是$O(n\log_2 n)$的。

2.21.4　时空复杂度

时间复杂度：$O(n\log_2 n)$

空间复杂度：$O(n)$

2.22　Codeforces 346E Doodle Jump

2.22.1　题目大意

现在有n个平台。第$x(1[x[n)$个平台的高度是$a\#xmodp$，其中a和p是互质的正整数。你最大可能跳的高度是h。也就是说，若$h2-h1<h$，则你可以从高度$h1$跳到$h2(h1<h2)$。一开始你站在高度为0的地上。

问你能不能跳上最高的平台。

t组数据。

2.22.2 数据范围

$1 \le t \le 10000$，$1 \le a \le 10^9$，$1 \le n < p \le 10^9$，$0 \le hleq10^9$

2.22.3 题解

首先我们特判$an < p$的情况。

对于一般的情况，可以发现，我们依次得到的总是许多个公差相同的等差数列，由于有可能最后一个等差数列的长度不够长，所以最长的没有点的区间一定在$[\lfloor \frac{p}{a} - 1 \rfloor a, \lfloor \frac{p}{a} \rfloor a)$的内部。

接下来我们可以求出这些等差数列起点的公差——即$a - p \bmod a$，而此时的高度变为了a，注意到当$a = x$ 时与$a = p - x$时他们的情况实际上是等价的，所以相当于此时原本的a变为$p \bmod a$，原本的p 变为a，原本的n变为$\lfloor \frac{a*n}{p} \rfloor$。迭代$O(log_2 n)$次即可。

2.22.4 时空复杂度

时间复杂度：$O(t \log_2 a)$

空间复杂度：$O(1)$

2.23 Codeforces 264D Colorful Stones

2.23.1 题目大意

现在有两个字符串a, b，它们的长度分别为n, m，有一个数对(x, y)，它初始为$(1, 1)$，现在你可以进行多次操作，每次可以报一个字符c，若字符$c = a_x$，则$x + 1$，若字符$c = b_y$，则$y + 1$，两个事件是同时发生的。

如果你在报了一个字符之后会使$x > n$或$y > m$，则此次操作是不合法的。

问你可以得到多少个不同的数对，两个数对不同当其x值不同或y值不同。

2.23.2 数据范围

$1 \le n, m \le 10^6$，字符串仅由RGB三种字符组成。

2.23.3 题解

我们枚举数对的x值为多少，观察y有多少可能值。

首先我们可以找到最小和最大的y使(x, y)合法，这个我们只需要每次贪心即可，接下来我们需要寻找其中不合法的y共有多少个。

可以发现，当y满足$a_x = b_{y+1}, a_{x+1} = b_y$且$a_x \neq a_y$时，此时的$y$必然无法到达。否则只要$y$在最小的与最大的之间则必然有解。

所以我们每次维护出最小和最大的合法y，再用前缀和记录为b_y, b_{y+1}的y共有多少即可。

2.23.4 时空复杂度

时间复杂度：$O(n + m)$

空间复杂度：$O(n + m)$

2.24 Codeforces 329D The Evil Temple and the Moving Rocks

2.24.1 题目大意

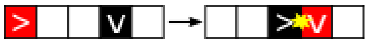
现在有一个$2n * 2n$的棋盘，你需要往其中放置一些有魔力的石头，石头分为四种类型：

1. '^'：这种类型的石块将会向上移动；
2. 'v'：这种类型的石块将会向下移动；
3. '<'：这种类型的石块将会向左移动；
4. '>'：这种类型的石块将会向右移动；

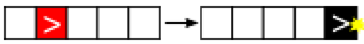
在放置完成之后，你可以选择其中一块石头并激活它。被激活的石块将会一直朝着它的方向移动，直到撞到了其他石块或者撞到了房间四周的围墙（如果在它的方向上紧挨着就有其他石块，它将不会有任何移动）。之后这块石头将停止运动。如果它撞到了围墙，则游戏结束。否则，它所撞击到的石块将被激活，且这一过程将会持续发生。但是，当所有石块被激活的总次数达到10^7后，即使游戏还在进行，游戏也将被强行停止。

下面的图片展示了石块撞击时可能的四种情形：

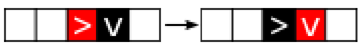
1. 石块至少移动了一步，并撞到了另一块石头。撞击发出了一记响声，被撞击的石块被激活。

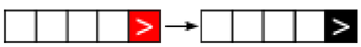
2. 石块至少移动了一步，并撞到了房间的围墙。撞击发出了一记响声，但是游戏结束。

3. 石块没有移动，但撞到了相邻的石块。撞击没有发出响声，但相邻石块被激活。

4. 石块没有移动，并且撞到了房间的围墙。撞击没有发出响声，并且游戏结束。

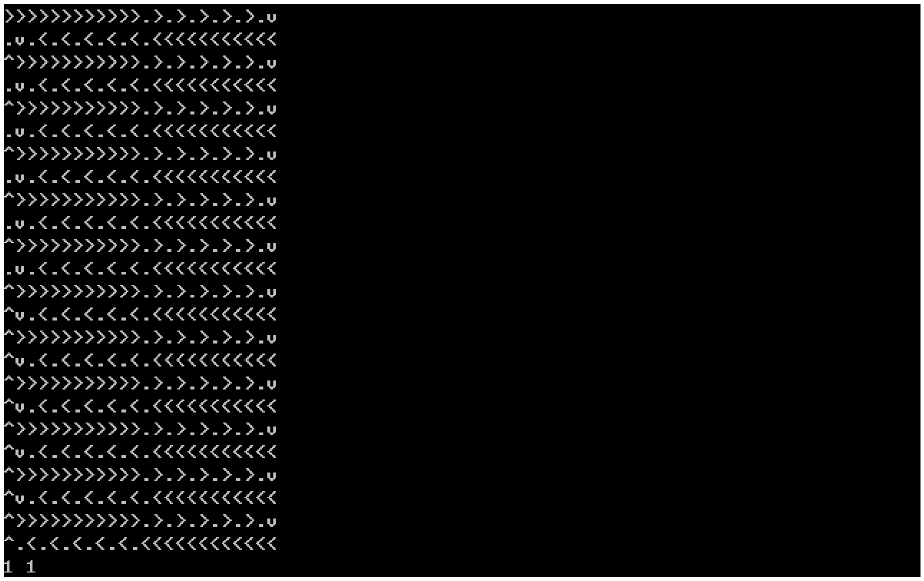
要求给出一种放置方案，并且给出第一个激活的位置，使得总碰撞次数不超过x。

2.24.2 数据范围

$1 \le n \le 150$，$x \le n^3 - n^2$

2.24.3 题解

对于$n = 12$的情况构造方法如下：



```
>>>>>>>>>>>>.>.>.>.>.>.v
.v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
.v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
.v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
.v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
.v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
.v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
^v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
^v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
^v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
^v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
^v.<.<.<.<.<<<<<<<<<<<<<
^>>>>>>>>>>>.>.>.>.>.>.v
^.<.<.<.<.<.<<<<<<<<<<<<
1 1
```

依此类推n更大时的构造方法。

2.24.4 时空复杂度

时间复杂度：$O(n^2)$
空间复杂度：$O(n^2)$

2.25 Codeforces 293D Ksusha and Square

2.25.1 题目大意

给出一个n个点的凸多边形，我们将其中所有格点点集作为S，现在从S中任选两个点以它们之间的连线为对角线做一个正方形，求这个正方形的面积期望为多少。

2.25.2 数据范围

$3 \leq n \leq 10^5$，坐标绝对值在10^6以内。

2.25.3 题解

期望即为所有的正方形面积之和，除以$\binom{|S|}{2}$即为期望。

我们将总面积的公式列出来：

$$\sum_{(x1,y1)}^{(x1,y1)\in S}\sum_{(x2,y2)}^{(x2,y2)\in S}\frac{(x1-x2)^2+(y1-y2)^2}{4}$$

可以发现，对于x部分与y部分可以分开算，所以我们需要求的即为：

$$\sum_{x1=-10^6}^{10^6} dx(x1)\sum_{x2=-10^6}^{x1} dx(x2)\frac{(x1-x2)^2}{2}$$

$$\sum_{y1=-10^6}^{10^6} dy(y1)\sum_{y2=-10^6}^{y1} dy(y2)\frac{(y1-y2)^2}{2}$$

其中，$dx(x)$代表在S中共有多少个横坐标为x的点，$dy(y)$代表在S中共有多少个纵坐标为y的点。

所以我们可以枚举$x1, y1$的值，同时维护：

$$\sum_{x2=-10^6}^{x1} dx(x2)\frac{(x1-x2)^2}{2}$$

$$\sum_{y2=-10^6}^{y1} dy(y2)\frac{(y1-y2)^2}{2}$$

对于以上两个式子，我们使用差分即可轻松解决。

2.25.4 时空复杂度

时间复杂度：$O(m)$

空间复杂度：$O(n)$其中m为坐标范围。

2.26 Codeforces 306C White, Black and White Again

2.26.1 题目大意

有n个位置，每个位置上可以放入无限多个点，但是要求在同一个位置中只能放置同一种颜色的点。

现在有a个两两不同的白点，b个两两不同的黑点，你要将它们插入n个位置，要求每个位置上至少有一个点，且顺序必然为有连续> 0个位置放置白点，再有连续> 0个位置放置黑点，最后还有连续> 0个位置放置白点。

问共有多少种放置方案（每个位置放点的顺序也不一样），答案取模10^9+9输出。

2.26.2 数据范围

$3 \le n \le 4000$，$2 \le w \le 4000$，$1 \le b \le 4000$，$w + b \ge n$

2.26.3 题解

我们枚举中间的白点的长度为k，则白点的区间共有$n - k - 1$种放置方案。

我们将白点与黑点分开考虑，可以发现为完全相同的问题：将x个点放入y个位置有多少种放置方案，假设总方案数为$W(x, y)$。最终答案即为：

$$\sum_{L=1}^{n-2} W(a, n-L)W(b, L) \cdot (n - L - 1)$$

对于$W(x, y)$，我们可以看作优先确定x个点的顺序，再往$x - 1$个缝隙中插入$y - 1$个格挡符。则有：

$$W(x, y) = x!\binom{x-1}{y-1}$$

。

预处理阶乘即可$O(n)$解决问题。

2.26.4 时空复杂度

时间复杂度：$O(n)$
空间复杂度：$O(n)$

2.27 Codeforces 317C Balance

2.27.1 题目大意

有n个水箱，容积为v，初始第i个水箱内有a_i的水，其中有m对水箱之间有管道，管道的流量均为无限大。

现在规定每个水箱的目标状态b_i，你需要进行不超过$2n^2$次操作使得第i个水箱内的水为b_i，每次操作你可以选择两个有管道相连的水箱x, y，将其中一个水箱的水减少t，另一个水箱的水增加t，要求在减少和增加之后两个水箱中的水仍然在$[0, v]$之间。

2.27.2 数据范围

$1 \leq n \leq 300$，$1 \leq v \leq 10^9$，$1 \leq e \leq 50000$，$0 \leq a_i b_i \leq v$

2.27.3 题解

首先如果两个点x, y不连通，则它们之间必然是不会影响到的。所以我们只需要考虑在同一个连通块中的水箱。

如果同一个连通块中的初始水量与期望水量是不同的，则无论如何也不会有解。

否则，我们对这个连通块求出它的生成树，每次找到一个度数大小为1的点，用最多n步从其他点调水到这个点，在删除这个点之后再考虑其他的点。

可以发现，这样操作每次必然不会影响到其他已经被删除的点，且总共也只会进行最多n次，所以总操作次数是不超过n^2的。

2.27.4 时空复杂度

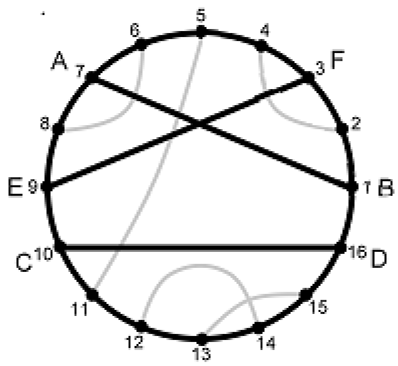
时间复杂度：$O(n^2 + m)$
空间复杂度：$O(n)$

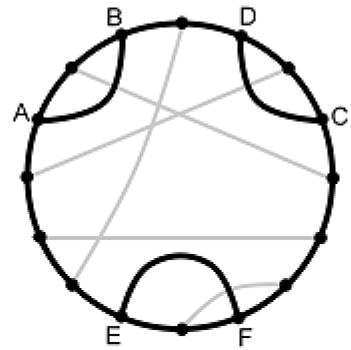
2.28 Codeforces 297E Mystic Carvings

2.28.1 题目大意

在一个圆形上有$2n$个点，有n条通道将它们连起来了，每条通道连接了两个点，且每个点只会被一条通道连接。

现在你需要选择3条通道(A, B)、(C, D)和(E, F)，我们要求对于每一条所选的管道，都有从它的一端到这条管道的另一端最少所需要经过的其他管道口数，与其他通道最少所需要进过的管道口数是相同的，在下图中，左边的为一种不合法的情况，右边的为一种合法的情况：





输出一共有多少种选择方案。

2.28.2 数据范围

$1 \leq n \leq 10^5$

2.28.3 题解

假设我们任意选择三条通道，它们所有可能的情况为：

1. 三条通道互不相交，且有一条线通道将另外两条通道分在了两边，不合法；
2. 三条通道互不相交，且对于每一条通道都有一侧没有其他通道，合法；
3. 一条通道和另一条通道相交，但第三条通道与其他的通道均不相交，不合法；
4. 一条通道与其他两条通道相交，但其他两条通道互不相交，不合法；
5. 三条通道两两相交，合法。

所以我们所需要求的即为随便取三条直线的方案，减去不合法的方案。

对于情况1，我们可以枚举中间的那条通道，之后其他两条通道我们可以从它的两边分别取出一条不相交的通道即可。

而对于情况3与情况4，它们分开来做都不好做，不妨我们将它看在一起。可以发现，它们都有两条线分别与一条线相交，与一条线不相交。那

么我们可以求出与一条线相交的线段数，与一条线不相交的线段树，相乘即可。

之前所提到的所有的线段计数我们都可以用树状数组解决。

2.28.4 时空复杂度

时间复杂度：$O(n\log_2 n)$
空间复杂度：$O(n\log_2 n)$

2.29 Codeforces 294D Shaass and Painter Robot

2.29.1 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。

2.29.2 数据范围

$2\le N,M\le 10^5$，$1\le X\le N$，$1\le Y\le M$，保证起点坐标在某一个边界上

2.29.3 题解

注意到每次我们最后一个染色的格子必然是在边界上的格子。

证明可以使用反证法，假设我们最后一个染色的不为边界上的格子，而是点T。

我们将我们所走过的路径连上边，若从一个点往另一个点走了多次，则我们连上多条边，这样可以连出一个欧拉图。

接下来对于这个点T，我们找到从T开始，往四个方向走第一个到达的边界点A,B,C,D。由于A,B,C,D 为边界上的点，则当我们经过其中一个点时，必然会将其往两个方向均连上边。而当T为终点时，必然

从A,B,C,D到T的路径中只会有一条是完整的，此时其他三个点必然连出一条在中途结束的路径，但是由于结束点最多只有2个，所以这样是不合法的。

之后我们只需要记录每个边界上的格子有没有被涂色，暴力BFS即可。

2.29.4 时空复杂度

时间复杂度：$O(N+M)$
空间复杂度：$O(N+M)$

2.30 Codeforces 321D Ciel and Flipboard

2.30.1 题目大意

给定一个$n*n$的矩阵，令$x=\frac{n+1}{2}$。现在你可以进行任意多次操作，每次操作将一个$x*x$的子矩阵中所有的元素正负号取反。定义矩阵的权值为其中所有位置上的数之和，问可以的到的最大权值为多少。

2.30.2 数据范围

$1 \le n \le 33$且n为奇数

2.30.3 题解

由于这题中操作为取反，很容易发现与异或操作十分类似。

设$rev_{i,j}$为在最后时刻第i行第j列的元素是否被取反，则我们有以下两个等式：

$$rev_{i,j}\ xor\ rev_{i,x}\ xor\ rev_{i,j+x}=0$$

$$rev_{i,j}\ xor\ rev_{x,j}\ xor\ rev_{i+x,j}=0$$

可以发现，这两个等式无论我们如何操作，均为成立的。因为我们每次操作对于每个等式会同时修改其中的两个值或者不修改。所以我们可以通过确定在rev中前$x*x$大小的矩阵来求出矩阵中所有的值。

所以我们可以枚举$rev_{1,x}$,$rev_{2,x}$,...,$rev_{x,x}$，与$rev_{x,1}$,$rev_{x,2}$,...,$rev_{x,x-1}$，接下来只需要分个枚举$rev_{i,j}$，即可求出答案。

但是直接这样是不行的，观察到$rev_{x,i}$的值只会影响到所有在第i列与第$i+x$列中的rev，所以我们可以分成每列贪心取最大值即可。

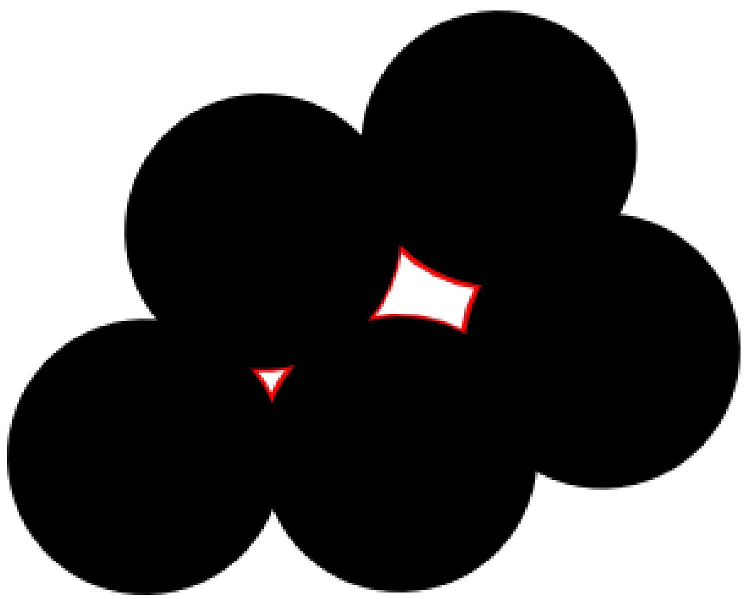
2.30.4 时空复杂度

时间复杂度：$O(2^{\frac{n+1}{2}}n^2)$
空间复杂度：$O(n^2)$

2.31 Codeforces 274C The Last Hole!

2.31.1 题目大意

在平面上有n个圆，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连通快。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，也会有一些旧的洞消失。求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。

2.31.2 数据范围

$1 \leq n \leq 100$，坐标绝对值在10000以内。

2.31.3 题解

在洞无限缩小时，它的形状必然趋近于一个点。我们思考一下有哪些点可能成为一个洞。

1. 它必须到一些点的距离相等，假设这个距离为R。
2. 在这些点构成的凸多边形中两两相邻的顶点圆心角必须小于π。
3. 没有点与它的距离小于R。

第一个条件确保了这个点是这个洞消失时最后一个被圆覆盖的点。

第二个条件确保了这个点会在一个时刻被圆所形成的连通块给包围住。

第三个条件确保了这个点不会在消失之前已经被其他的圆给覆盖。

接下来，我们求一个Voronoi图即可在$O(n\log_2 n)$的时间内找到所有的点。

但是对于这题的n太小，我们没必要小题大做去写一个Voronoi图。

可以发现，对于第二个条件，若一个凸多边形中两两相邻的顶点圆心角小于π，则我们必然可以从这个凸多边形中取出三个点作为一个三角形同样也是满足条件的。

所以我们枚举所有的锐角三角形，每次$O(n)$暴力判断条件三即可。

当然，这样是有反例的，但反例只有一种——长方形。由于长方形的总个数不会超过$n^{2.5}$，所以我们同样可以枚举长方形再判断。

2.31.4 时空复杂度

时间复杂度：$O(n^4)$

空间复杂度：$O(n)$

2.32 Codeforces 319D Have You Ever Heard About the Word?

2.32.1 题目大意

一个重复块由一个字符串与自身连接而成，如abcabc是一个重复块，而abcabd,ababab不是。

现在有一个长度为n的字符串s，每一步你要找到它的子串中最短的重复块，如果有多于一个，你必须选择最左边的那个。你要将那个形如XX(X - 某个字符串)的重复块替换成X，换句话说你要删除其中的一个X。重复以上步骤直到字符串中不存在重复块。

输出最终字符串。

2.32.2 数据范围

$1 \le n \le 50000$

2.32.3 题解

我们从小到大依次枚举重复块的长度。

假设我们可以判断出有没有长度为$2i$的重复块，那么我们可以直接暴力$O(n)$删除这些重复块。由于每次删除之后字符串的长度至少会减少$2i$，而i每次会增加1，所以总复杂度是不超过$O(n\sqrt{n})$的。

现在问题变为了如何判断是否存在长度为$2i$的重复块。

若一个重复块q的长度为$2i$，我们从中选出一个位置$x(1 \le x \le i)$，则子串$q_{1..x}$必然等于子串$q_{i+1..i+x}$，同样的，子串$q_{x..i}$必然也等于子串$q_{i+x..2i}$。

所以我们可以枚举原字符串中$1, 1+i, 1+2i, ...$的位置，判断它与之前一个位置的最长公共前缀与最长公共后缀的长度之和是否超过i即可。

由于当长度为i时，我们需要判断的次数为$\lfloor \frac{n}{i} \rfloor$，所以我们总共需要判断的次数为$\sum_{i=1}^{n} \lfloor \frac{n}{i} \rfloor$，是不超过$O(n \log_2 n)$的。

2.32.4 时空复杂度

时间复杂度：$O(n \log_2 n + n\sqrt{n})$

空间复杂度：$O(n)$

2.33 Codeforces 241D Numbers

2.33.1 题目大意

有一个长度为n的排列，要求你从中删除一些数，使得剩余的数列满足以下条件：

1. 剩下的数列不是空的
2. 数列中所有元素的异或和等于0
3. 如果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。

求一种删除的合法方案。

2.33.2 数据范围

$1 \le n, p \le 50000$，p为质数。

2.33.3 题解

假设n的范围比较小，思考如何可以直接用DP解决问题。

设$F_{i,j,k}$代表选到了第i个数，当前已经选的数异或和为j，且从左至右写下来之后对p取模的结果为k。

转移时我们枚举当前一个数是否取即可。时间复杂度为$O(n^2p)$。

这题中n的范围太大，我们适当的将其缩小一下。我们假设在n大于一个一定的值k时，将排列中大于k的数都删掉。

假设t为最小的t满足$2^t > k$，观察对于1到k的数，则其中一共有2^{k-t}个不同的删数方法可以使得数字的异或和等于0。而将一个数列中所有的数从左至右写下来之后对p取模可以看作一个在$[0, p)$中随机取值的函数。

按照概率计算，当我们只保留小于等于k的值时，不存在答案的概率为$\left(\frac{p-1}{p}\right)^{2^{k-t}}$。

则我们适当取k的值即可，可以发现当k取到25时，失败的概率为$7.79 * 10^{-10}$，可忽略不计。

2.33.4 时空复杂度

时间复杂度：$O(k^2p+n)$
空间复杂度：$O(k^2p)$其中k表示保留了多少个数

2.34 Codeforces 314E Sereja and Squares

2.34.1 题目大意

在平面上有n个点，点i在坐标$(i,0)$。每个点标了一个小写或大写英文字母。其中没有"x"。定义这些点是漂亮的，当且仅当：

1. 所有的点可以被分成若干对，使得每个点恰好属于一一对之中。
2. 在每对点中，横坐标较小的会被标上小写字母，较大的会被标上对应的大写字母。
3. 如果我们在每对点上画一个正方形，其中已知的一对点会作为正方形的相对的顶点，它们间的线段会成为正方形的对角线，那么在所有画出的正方形中不会有相交或触碰的情况。

现在擦掉了一些小写字母和所有大写字母，求有多少种方法来补全每个点上的字母，使得还原后这些点是漂亮的，答案对4294967296取模。

2.34.2 数据范围

$1 \le n \le 10^5$

2.34.3 题解

我们先来思考$O(n^2)$的DP。
$F_{i,j}$表示到了第i个位置，之前的大写字符比小写字符少了j个。
每次转移的时候枚举当前是大写字符还是小写字符即可。注意若其原本就是小写字符，则必须填小写字符。
直到目前没有听说更好复杂度的做法，用滚动数组优化空间，卡常数即可。

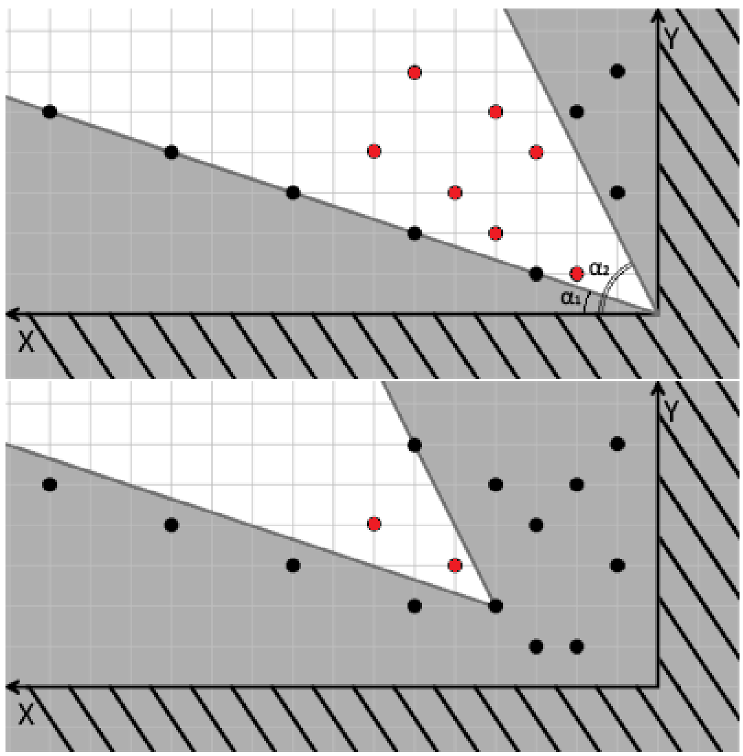
2.34.4 时空复杂度

时间复杂度：$O(n^2)$
空间复杂度：$O(n)$

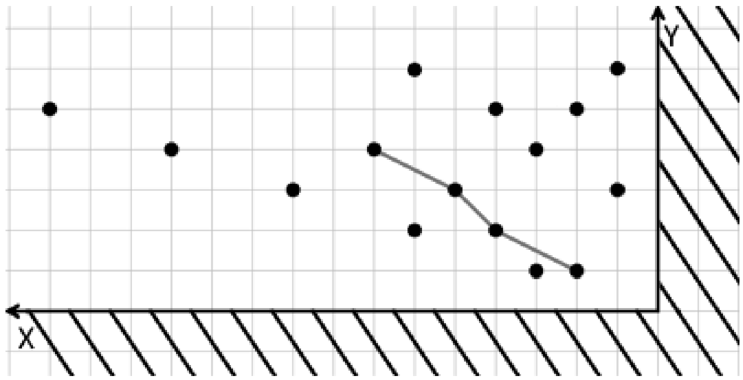
2.35 Codeforces 249D Donkey and Stars

2.35.1 题目大意

在平面上有n个点，点x向另外一个点y连边，当且仅当y在x的右上角，且y对于x的仰角α满足$\alpha 1 < \alpha < \alpha 2$，如下图中红点即为有边的点：



现在你从点$0,0$出发，求最长路径的长度。下图中黑线所表示的即为最长路径：



2.35.2 数据范围

$1 \leq n \leq 10^5$，坐标范围在10^5以内，$0^\circ \leq \alpha 1 < \alpha 2 \leq 90^\circ$

2.35.3 题解

我们将每个点向仰角为$\alpha 1$的线、仰角为$\alpha 2$的线作出垂线，可以发现，一个点x与一个点y 之间有边，当且仅当x在这两条线投影上的距离均比y在这两条线投影上的距离短。

所以我们将所有的点用在两条边上的投影距离来表示，由于两条直线的斜率不一样，所以不同的点在两条边上的投影距离不可能完全一样。

接下来，我们需要求一条最长的(X, Y)序列，且每个X递增，每个Y递增，可以发现这个就是最长上升子序列问题，用$O(n \log_2 n)$的时间解决即可。

2.35.4 时空复杂度

时间复杂度：$O(n \log_2 n)$
空间复杂度：$O(n)$

3 USACO月赛

3.1 USACO Open 2007 Connect

3.1.1 题目大意

有一个$m*n$的网格图，初始上面有一些边，现在有Q次操作，每次操作为下列三种中的一种：

1. 去掉一条$(r1,c1)$到$(r2,c2)$的边，保证这两个点相邻，并且两个点之前存在边相邻。

2. 加上一条$(r1,c1)$到$(r2,c2)$的边，保证这两个点相邻，并且两个点之前不存在边相邻。

3. 询问是否有一条路径可以从$(r1,c1)$到达$(r2,c2)$。值得注意的是：这里的“到达”指的是所经过的路径不能超出$c1$列和$c2$列之间的范围。也就是说路径所经过的所有点的列编号cx满足$min(c1,c2) <= cx <= max(c1,c2)$。

对于所有的查询输出结果，若相邻输出Y，否则输出N。强制在线。

3.1.2 数据范围

$1 \le m \le 2$，$1 \le n \le 15000$，$1 \le Q \le 50000$

3.1.3 题解

注意每次查询时所经过的路径不能超出$c1$列和$c2$列之间的范围，相当于每次查询对于列$c1$到列$c2$的子图。而本题中m的范围同样也很小，所以我们可以直接用线段树维护连通性。

对于每一个区间$[x,y]$，我们维护点$(x,1),(x,2),(y,1)$和$(y,2)$的连通性。在合并两个小区间时，可以用并查集来做。每次修改边时将对应在线段树中的节点修改，查询时将总区间分为$O(\log_2 n)$条小区间，合并即可。

事实上这题在USACO月赛时是一道交互题，但出题人在搬运至清澄时用了一个奇怪的加密方法，事实证明可以利用这个加密方法的漏洞+暴力来解决这一题。

3.1.4 时空复杂度

时间复杂度：$O(Q\log_2 n)$
空间复杂度：$O(n)$

3.2 USACO December 2007 Best Cow Line, Gold

3.2.1 题目大意

有一个字符串S，现在你要生成一个新的字符串T，每次你可以从S的开头取出一个字符加在T的末尾，或从S的末尾取出一个字符加在T的末尾，直到S中没有字符。

求你所能生成的字典序最小的串。

3.2.2 数据范围

$1 \le n \le 30000$

3.2.3 题解

可以发现，如果我们需要让生成的字符串字典序最小，假设当前的左边界为l，右边界为r，则可以有以下的比较方法：

1. 若从l到r间所有的元素均相等，则可以任意选择从开头取至r或从末尾取至l。

2. 若$S_l \ne S_r$，则比较S_l与S_r的大小，当$S_l < S_r$时，从S的开头取出字符至l，否则从S的末尾取出字符至r。

3. 若$S_l = S_r$，则找到从l开始从前往后的第一个S_x使得$S_x \ne S_l$，找到从r开始从后往前的第一个S_y使得$S_y \ne S_r$，比较$[S_l < S_x]$与$[S_r < S_y]$的大小来决定是从S的开头取出字符至l，还是从S的末尾取出字符至r。

4. 若$S_l = S_r$,$[S_l < S_x] = [S_r < S_y]$且$x - l \ne r - y$，则比较$x - l$与$r - y$的大小来决定是从开头取至$x - 1$ 还是从末尾取至$y + 1$。

5. 若$S_l = S_r$,$S_l > S_x$,$S_r > S_y$且$x - l = r - y$，则我们可以任选从开头取至$x - 1$，或从末尾取至$y + 1$。

6. 若经过前面的比较，仍然没有决定，则我们将l移动至x，将r移动至y，继续第一步的比较。

我们预处理出在每个数之前或每个字符之后的第一个不同的字符。可以发现，若我们在一次操作中比较的次数为x，则删除的数的个数必然不小于x，所以我们如此做是$O(n)$的。

但是这样子分情况略微麻烦，不妨我们每次只考虑当前一次是从前面取还是从后面取。可以发现，我们实际上比较的是S中剩余的字符串与S翻过来之后字符串的字典序，所以我们可以直接预处理Hash值或用后缀数组来比较字典序大小，复杂度$O(n\log_2 n)$，若需优化到$O(n)$可以使用$O(n)$的后缀数组或用后缀自动机建出后缀数组即可。

3.2.4 时空复杂度

时间复杂度：$O(n)$或$O(n\log_2 n)$
空间复杂度：$O(n)$

3.3 USACO March 2008 Land Acquisition

3.3.1 题目大意

在平面直角坐标系上有N个点(X_i, Y_i)。现在你需要进行任意多次操作，每次操作可以选两个值A, B，所需要的代价为$A * B$，然后你可以删除所有$X_i \le A$且$Y_i \le B$的点，问删除所有的点最少需要多少代价。

3.3.2 数据范围

$1 \le N \le 50000$，坐标均为正数且范围在10^6以内。

3.3.3 题解

首先对于所有(X_i, Y_i)，若有点(X_j, Y_j)满足$X_i \le X_j$且$Y_i \le Y_j$，则在点j被删掉时，点i必然也可以同时也被删掉，所以我们忽略所有满足条件的i。

接下来，我们按照X坐标从小到大排序，则Y坐标为下降子序列。此时若我们一次删除同时删掉了点i与点j，则我们可以同时删掉所有的点k满足$i \le k \le j$。

所以很容易设出$O(n^2)$的DP，设F_i表示删到了第i个点所需要的最小代价，转移时可以枚举从哪个点转移过来，方程为：

$$F_i = min\{j|F_{j-1} + X_i * Y_j\}$$

可以发现对于所有的i，在往后转移时的值即为一个关于X的一次函数，所以我们用斜率优化DP即可。

3.3.4 时空复杂度

时间复杂度：$O(N \log_2 N)$
空间复杂度：$O(N)$

3.4 USACO Open 2008 Cow Neighborhoods

3.4.1 题目大意

平面上有n个点，若两个点i,j属于一个群，则需要满足以下条件：

1. 两个点的曼哈顿距离不超过C，即$|x_i - x_j| + |y_i - y_j| \le C$
2. 存在第三个点k满足i,k属于一个群，j,k属于一个群。

算出总共有多少个群，以及最大的群里面有多少奶牛。

3.4.2 数据范围

$1 \le n \le 10^5$，$1 \le C \le 10^9$，坐标范围在$[1, 10^9]$

3.4.3 题解

如果我们将点与点之间的边两两连出来，可以发现，实际上是要找所有长度不超过C的边所构成的连通块共有多少个。

实际上这个问题在最小生成树上解决答案是不会变的，所以我们可以直接求出这个图的曼哈顿路径最小生成树，然后再在树上解决即可。

至于曼哈顿路径最小生成树如何求，可以发现，一个点(X,X)在由直线$x = X$,$y = Y$,$y = X + Y - x$以及$y = Y - X + x$所分出的八个区间内各会有一个最近点，我们用线段树维护分别找出来即可。

事实上这题并没有这么麻烦，我们可以将整个图旋转45°，可以发现，一个点x与另外一个点y 有边，当且仅当x在以y为中心的边长为C的矩形中。

接下来我们将所有点按照x坐标排序之后，从左往右扫一遍，对于每个点，我们考虑在它左边且x坐标相差不超过C的点，每次取出最接近$y+C$和$y-C$且在它们之间的两个点，与它们用并查集合并即可。

3.4.4 时空复杂度

时间复杂度：$O(n\log_2 n)$
空间复杂度：$O(n)$

3.5 USACO November 2008 Toys

3.5.1 题目大意

一共有N天，第i天需要S_i个干净的玩具，现在可以直接花T的价格购买一个玩具，购买的玩具一开始是干净的，同时也可以把玩具拿去消毒，第一种为花$C1$的价格进行快速消毒，需要$N1$个晚上的时间，第二种为花$C2$的价格进行缓慢消毒，需要$N2$个晚上的时间。

问所需要花费的代价最少为多少才能使得每天所用的玩具均为干净的。

3.5.2 数据范围

$1 \le N \le 10^6$，$1 \le S_i \le 50$，$1 \le N1, N2 \le N$，$1 \le C1, C2, T \le 60$

3.5.3 题解

首先我们来考虑当N很小时的做法，可以很容易想到一种费用流的做法：

1. 将所有点给拆分成两个点，第一个点表示到当天是干净的玩具，第二个点代表用过了的点，我们将每个点i 的第二个点向$i+N1$的第一个点连一条代价为$C1$，流量为无穷大的边，向$i+N2$ 的第一个点连一条代价为$C2$，流量为无穷大的边，分别代表快速消毒和缓慢消毒。

2. 每天未使用过的玩具可以留到之后使用，所以我们从第i天的第一个点向第$i+1$天的第一个点连一条代价为0，流量为无限大的边。

3. 购买玩具可以看作为第一天增多了若干个干净的玩具，所以我们可以从源点向第一天的第一个点连一条代价为T，容量为无限大的边。

4. 对于每天的消耗，由于是必须满足的，所以我们从每天的第一个点向第二个点连出一条代价为负无限大，流量为S_i的边。

5. 对于玩具的废弃，最优必然是将脏的玩具给废弃，所以我们可以将所有点的第二个点向汇点连一条边权为0，流量为无限大的边。

接下来，我们只需要跑一遍最小费用可行流，将最后总代价加上$\sum_{i=1}^{n} S_i$倍边权中所定的无限大权值即可。

有一个很重要的定理为：费用流算法的总费用与总流量的关系为一个凸函数。在本题中所构建的费用流流量即为所购买的玩具个数，说明我们只需要三分玩具个数，再进行求总费用即可。

至于如何求当购买的玩具一定时的总代价，我们可以直接使用贪心，每次尽量使用慢洗，这样贪心的正确性是显然的，至此问题解决。

3.5.4 时空复杂度

时间复杂度：$O(N\log_2(NS))$
空间复杂度：$O(N)$

3.6 USACO December 2008 Fence

3.6.1 题目大意

给定n个点，问最多选取其中多少个点可以构成一个凸多边形。

3.6.2 数据范围

$3 \le n \le 250$，坐标范围在$[1,1000]$，保证无任意三点共线。

3.6.3 题解

首先我们先来思考$O(n^4)$的朴素DP：

设$F_{i,j,k}$代表起点为第i个点，当前到了第k个点，到达的倒数第二个点为j时的最多经过的点的个数。

转移时，枚举需要转移到的点l，判断$j \to k$的斜率是否比$k \Rightarrow l$的斜率小即可。

可以发现，每次我们转移时都是从斜率小的边转移到斜率大的边上，不妨我们将所有的边给排一遍序之后再进行转移，这样我们就无需记录之

前一个点是从哪里来的了，我们只需要记录$F_{i,j}$代表起点为i，当前到了j的最小经过的点的个数为多少。

接下来，我们在枚举边$x \Rightarrow y$时，我们只需要拿所有的$F_{i,x}$直接更1新$F_{i,y}$即可。

3.6.4 时空复杂度

时间复杂度：$O(n^3)$ 空间复杂度：$O(n^2)$

3.7 USACO March 2009 Cleaning Up

3.7.1 题目大意

现在有n头奶牛排成一行，且共有m种不同的颜色，每头奶牛都有一个颜色Col_i。定义一段区间的权值为：若这段里有k个不同颜色的奶牛，则权值为k^2。现在需要把这些奶牛分成若干连续的段，要求每一段的权值总和最小，求最小的权值为多少。

3.7.2 数据范围

$1 \le n, m \le 40000$，$1 \le Col_i \le m$

3.7.3 题解

先考虑$O(n^2)$的DP，设F_i代表分组到了第i头牛的最小权值，转移时枚举当前段为多长即可。

注意若有一个区间中所包含的颜色超过了$\lfloor\sqrt{n}\rfloor$，则它的总权值必然大于n，这样我们可以将n头牛每头牛当作一段，这样必然更优。

所以我们可以顺便维护一下当前段包含j种不同的颜色最远可以到达的位置，暴力转移当j不超过$\lfloor\sqrt{n}\rfloor$即可。

3.7.4 时空复杂度

时间复杂度：$O(n\sqrt{n})$

空间复杂度：$O(n)$

3.8 USACO Open 2009 Tower of Hay

3.8.1 题目大意

现在有n个数a_i，要求将它们分成尽可能多的段，使得每一段中的数字之和总不会比之前一段的数字之和小。

求最多的段数。

3.8.2 数据范围

$1 \le n \le 10^5$，$1 \le a_i \le 10000$

3.8.3 题解

优先思考$O(n^3)$的DP，设$F_{i,j}$代表最后的一段为从i到j的一段最多可以分成多少段，转移时枚举当前一段的长度即可。

注意到一个规律——只有当最低边为最短时，分出来的总段数才最多。

具体证明如下：

任意取出一个能使在层数最高的前提下底边最短的方案，设共有x层，把其中从下往上第i层最大编号的块记为A_i，任取一个能使在底边最短的前提下层数最高的方案，设共有y层，从下往上第i层最大编号的块记为B_i。所需要证明的即为$A_1 = B_1$。

我们使用反证法，即为$A_1 > B_1$，此时必然有$x > y$，则$A_y < A_y$。这说明至少存在一个$1 < k \le y$，满足$A_{k-1} \ge B_{k-1}$ 且$A_k \le B_k$，即存在一个第k层，满足用第二种构造的方法在第k层完全包含了第一种构造方法的第k层，此时我们只需要新构造一种方法使得第$k \sim x$层用第一种构造方法，第$1 \sim k-1$层用第二种构造方法。新方法的层数与第一种相同，最下一层的长度与第二种相同，与之前的矛盾。

了解了这个性质之后，DP就是$O(n^2)$的了，注意其中的单调性，使用单调队列优化即可$O(n)$解决。

3.8.4 时空复杂度

时间复杂度：$O(n)$

空间复杂度：$O(n)$

3.9 USACO Open 2010 Triangle Counting

3.9.1 题目大意

平面上有n个点，问有多少个三角形包含原点$(0, 0)$。

3.9.2 数据范围

$1 \le n \le 10^5$，坐标绝对值在10^5以内，保证没有两个点的直线经过原点

3.9.3 题解

将问题补集转化一下，即为有多少个三角形不包括原点。

我们枚举三角形中的一个点，将其与原点的连线画出来，可以发现，若这个三角形包含原点，则另外两个点必然在直线的两侧，而当它不包括原点时，在一个三角形中必然有两个点分别满足另外两个点均在直线的同一边。

所以我们只需要求出一个点与原点的连线将所有的点分为两个点集的大小是多少，将其从同一个点集中选择两个的方案计入总答案即可。

对于如何求点集大小，我们可以直接将所有的点按照极角序排序，之后用指针单调扫即可。

3.9.4 时空复杂度

时间复杂度：$O(n)$
空间复杂度：$O(n)$

3.10 USACO December 2010 Threatening Letter

3.10.1 题目大意

现在有无数份长度为n的串S，你每次可以从一个串中剪出一个子串，现在要求操作最少的次数，使得你剪出来的所有子串可以组成一个长度为m的串T。

3.10.2 数据范围

$1 \le n, m \le 50000$

3.10.3 题解

我们从前往后剪出T，则每次必然剪出最长的一段尚未剪出的子串。

所以我们直接对于串S建出后缀数组或后缀自动机，找出每次T所能剪出的最长一段即可。

3.10.4 时空复杂度

时间复杂度：$O(n+m)$或$O(n\log_2 n+m)$
空间复杂度：$O(n+m)$

3.11 USACO January 2012 Cow Run

3.11.1 题目大意

有一个数字x，初始为0，你进行了n次操作，每次操作给你了8个数字s_1到s_8，你可以选择前4个数字或者后4个数字，但是需要在操作之前提前作出决定。

在你选出了$4n$个数字之后，对于每次的4个数字，我们随机选择前两个或者是后两个，假设选择的数为a和b，那么将x变为$(a+1)x+b$。

现在你要决定一种选择方式，每次是选择前4个还是后4个，使得无论随机函数怎么选择，最后生成的x均满足$max(m-x \bmod m, x \bmod m) \leq K$

3.11.2 数据范围

$2\leq m\leq 10^9$，$1\leq n\leq 14$，$0\leq K\leq \lfloor\frac{m}{2}\rfloor$

3.11.3 题解

我们直接暴力枚举你是如何选择以及随机函数是如何选择的，时间复杂度为$O(4^n)$，卡卡常数可以在时限内通过。但是这题中要求最小字典序，所以我们需要依次枚举每位上选择前4位还是后4位。所以我们加一些剪枝，当我们选择前4位已经可行时就不用去搜索后4位，当随机函数选择前2位时已经不可行就不用去搜索后2位。但是这样仍然会被构造数据给卡住，我们加一个随机来决定是先搜索前面还是先搜索后面即可。

3.11.4 时空复杂度

时间复杂度：$O(n4^n)$
空间复杂度：$O(8n)$

3.12 USACO December 2012 First!

3.12.1 题目大意

现在给定n个串s_i，要求你给他们重新排一个字符集的顺序，问有哪些串可以是字典序最小的串。

3.12.2 数据范围

$1 \le n \le 30000$，$1 \le \sum_{i=1}^{n} |s_i| \le 300000$

3.12.3 题解

我们可以枚举最小的串是哪一个。假设最小的串是s_i，对于所有的串s_j，找到唯一的x，使得对于所有满足$1 \le k < x$的k，有$s_{i,k} = s_{j,k}$，且$s_{i,x} \ne s_{j,x}$。那么我们在定义字典序的时候，必须要让$s_{i,x}$的顺序小于$s_{j,x}$。

我们对于每个串都可以列出许多个这样的约束，我们按照若x小于y则从x向y连一条边，最后我们只需要判断图中是否可能存在环即可。

但是如果按照这样的构造图中的边数一共有$O(n^2)$个。注意到对于所有相等的x，我们列出的约束所在意的只有s_x的值，所以我们将所有的串插入一棵字典树，每次建边只需要枚举子树即可。

3.12.4 时空复杂度

时间复杂度：$O(26^2 + 26S)$
空间复杂度：$O(26S)$其中S代表字符串总长度

3.13 USACO December 2012 Gangs of Instanbull/Cowstantinople

3.13.1 题目大意

有n头牛，它们属于m个帮派，第i个帮派的牛数为a_i。现在它们在争夺草原，当一头牛来到草原上，如果草原为空，那么它所属的帮派就占领了

草原，初始势力大小为1。当草原上有牛，如果草原上的帮派与它的帮派相同，则草原上的势力大小加1；否则从草原上的势力大小减1。当草原上的帮派势力大小为0时，草原就不再被这个帮派控制。

你需要确定一个牛进入草原的顺序，判断是否可能在所有牛都进入过草原之后第一个帮派占领着草原。若可能，请给出一种字典序最小的方案使得最后草原是由第一个帮派控制着的，且草原上第一个帮派的势力大小最大。

3.13.2 数据范围

$1 \leq n \leq 10^6$，$1 \leq m \leq n$

3.13.3 题解

首先我们判断帮派1是否可能占领草原。如果有一个帮派i满足$a_i \geq \frac{n}{2}$，且$i \neq 1$，则必然不可能在最后仍由帮派1占领，否则我们必然可以构造出一种进入草原的方式，使得在最后由帮派1占领草原。

然后我们来计算帮派1最后剩余的帮派势力最大为多少。若不考虑字典序，则最优的必然是让除了第一个帮派之外的所有帮派先“自相残杀”，最后再让帮派1进入草原，再占领草原。所以我们需要考虑的即为除了第一个帮派之外的帮派互殴之后所剩余的最小势力大小为多少。

若在剩余的帮派中存在一个相当强大的帮派k，满足$a_k \geq \frac{n-a_1}{2}$，则我们最优的政策必然是用其他的帮派去削弱这个帮派的势力之后再让它和帮派1去比较。此时第一个帮派所剩的势力大小最大的值即为$n - 2a_k$。

否则，若不存在k满足以上条件，那么当$n - a_1$为偶数时，必然会有方案使得除了帮派1的所有牛进入草原之后，草原属于无主状态。此时第一个帮派所剩的势力大小最大的值即为a_1。当$n - a_1$为奇数时，同样也会有方案使得除了帮派1的所有牛进入草原之后，草原上只会剩余1头牛，此时第一个帮派所剩的势力大小最大的值为$a_1 - 1$。

接下来，我们只需依次枚举每位上放入哪个元素是否会改变最终答案即可。

注：在tsinsen上有很多trick数据没有，比如没有$n = 2$的数据，同时没有存在一个帮派k满足$a_k \geq \frac{n-a_1}{2}$的数据。这导致了很多错误的贪心也可以通过。

3.13.4 时空复杂度

时间复杂度：$O(n)$
空间复杂度：$O(n)$

4 Google Code Jam World Final

4.1 Google Code Jam 2014 Final C Symmetric Trees

4.1.1 题目大意

给你一棵有n个节点且顶点被染色的树，问它在平面中能否关于一条直线对称（轴对称）。

在形式上，一棵树是轴对称的，当且仅当每一个点在平面内都能分配到这样一个位置：

1. 所有点的位置都是不同的。
2. 假如一个点v_i的颜色是C并且坐标是(x_i, y_i)，就必须有另外一个点$v_{i'}$的颜色同样是C并且坐标是$(-x_i, y_i)$——注意，如果$x_i = 0$，v_i和$v_{i'}$就是同一个点。
3. 对于每一条边(v_i, v_j)，一定也存在一条边$(v_{i'}, v_{j'})$。
4. 如果边用两点之间的线段表示，那么在除了端点的地方外，两条边不会有任何公共点。

共有T组数据，对于每组数据输出这棵树是否为轴对称。

4.1.2 数据范围

对于$2 \leq n \leq 5$，有$1 \leq T \leq 10$
对于$6 \leq n \leq 10000$，有$1 \leq T \leq 3$

4.1.3 题解

若这棵树是轴对称的，则必然有至少一个点在y轴上或没有点在y轴但是有一条边经过y轴，且在y轴两侧的点数必然是相等的。

首先我们先考虑有一条边经过y轴的情况，若有一条边在y轴，由于在y轴两侧的点数是相等的，所以我们只需要找到一条边，它的两侧均有$\frac{n}{2}$个点，接下来，我们只需要判断对于这条边两侧的子树是否同构即可。

对于判断两个子树是否同构，我们可以用树的Hash，每次将一个点的所有子树按照Hash值从小到大排序再依次合并，这样两棵同构的子树求出来的Hash值必然完全相同。

然后我们考虑有至少一个点在y轴上的情况。假设已经找到了这棵树在y轴上的其中一个点，则最优的情况必然是将所有同构的子树两两配对分在y轴两侧，若可以两两配对，则该树必然是轴对称的。若没有两两配对，则剩余的子树必然是沿着y轴向上或向下，然后再将子树的子树分在y轴的两侧。可以发现这个是大致相同的子问题，所以我们可以求出每棵子树是否为轴对称的即可。需要注意的是，对于根来说，可以有最多两个多余的未匹配子树，因为可以同时向上和向下，但是对于一棵子树来说，只能有一个未匹配的子树，因为子树必然有一个方向上由它的父亲相连。

若直接枚举在y轴上的点，总复杂度是$O(n^2)$的，虽然对于一组数据可以勉强通过，但是本题中有多组数据，所以我们需要将时间复杂度变得更优。

对于有点在y轴上的对称有一个显而易见的结论：在y轴上的点必然包含这棵树的重心。

对于证明，我们可以使用反证法。若该点的重心不在y轴上，而是由另外一个点i在这棵子树上，由于点i不是重心，所以必然有一棵子树的大小大于$\frac{n}{2}$。但是这棵树的总大小为n，所以必然不可能存在第二棵子树的大小大于$\frac{n}{2}$。如果整棵树是轴对称的，那么这棵子树的根结点必然同样也在y轴上。我们将i移至这棵子树的根结点，继续进行这个过程。可以发现，这样子走和寻找重心的过程完全一样，所以必然最后会走到重心。

有了这个性质，接下来我们只需要找到树的重心，我们以这个重心为根，求出整棵树是否轴对称即可。

4.1.4 时空复杂度

时间复杂度：$O(Tn\log_2 n)$
空间复杂度：$O(n)$

4.2 Google Code Jam 2014 Final D Paradox Sort

4.2.1 题目大意

有n个人，给出他们两两之间互相PK能否获胜，现在要求你确定一个排列，让他们按照排列的顺序依次进入PK场，每次PK获胜的人留下来接受新进来的人的PK直到最后没有人时，问最后是否可能由点S留下。

有T组数据，对于每组数据，若最后可能由第S个人留下，请输出字典序最小的可以使S留下的序列。

4.2.2 数据范围

$1 \le n, T \le 100$

4.2.3 题解

首先我们判断S怎样才能在最后留下。

我们将每个人都看作一个点，若一个人x能赢y，则我们连出一条单向边$x \Rightarrow y$。若从S开始BFS，不能遍历到所有的点，那么S必然不能在最后留下，否则我们必定可以构造出一组序列使得S在最后留下。我们对于这个图，选出其中的一些边构成一棵树，使得S为树的根，则我们如此操作：每次将这棵树的任意一个儿子加入序列，则每次PK之后无论结果如何，在剩余的树中至少会有一个人可以赢过当前在PK 场里面的人，直到最后只剩根为止，留在PK场里面的人必定能被S打败。

所以我们可以依次枚举序列中的每位上放的是编号为多少的人，接下来再用BFS判断即可。

4.2.4 时空复杂度

时间复杂度：$O(Tn^4)$

空间复杂度：$O(n^2)$

4.3 Google Code Jam 2014 Final F ARAM

4.3.1 题目大意

你在玩一个游戏，这个游戏中有n个英雄，你使用每个英雄有一个胜率p_i，不存在平局。每盘开始时给你等概率的随机发一个英雄，若你对当前的英雄不满意且你有超过1个游戏币，那么你可以花费1个游戏币进行一次重新随机。你在每进行一盘游戏之后会获得$\frac{1}{G}$个游戏币，但是你所拥有的游戏币不能超过R。

现在你进行了无限多盘游戏，要求制定一个策略，使得你进行游戏的总胜率最高，输出最高的胜率为多少。

有T组数据。

4.3.2 数据范围

$1 \le n \le 1000$，$1 \le R, G \le 20$，$1 \le T \le 100$

4.3.3 题解

首先我们需要求的胜率，实际上就是从第一次游戏开始，到最后我们的游戏币再次变为R的游戏过程中，胜率为多少。

直接使用概率是不好处理问题的，我们借助分数规划来解决问题。我们二分最优的概率为Q，则在我们获胜一盘时，总权值增加了1，在玩了一盘时，总权值减少了Q，最终所需要求的即为期望总权值是否可能大于0。

接下来，我们设出状态F_i代表我们在一盘开局时拥有$\frac{i}{G}$的游戏币，按照一个策略进行了若干盘之后，游戏币的数量变到了$\frac{i+1}{G}$时的期望权值最大为多少。

对于$0 \le i < G$时，我们每盘游戏都不能随机，所以有：

$$F_i = \frac{1}{n}\sum_{j=1}^{n}(p_j - Q)$$

对于$G \le i < RG$时，我们每盘游戏在分配到英雄之后可以选择随机或不随机。注意到我们选择随机必然是获胜概率小的英雄，所以我们必然是选择获胜概率前k小的英雄，我们将所有英雄按照胜率排序，则对于当前权值有：

$$F_i = \frac{k}{n}\sum_{j=0}^{G}F_{i-G+j} + \frac{1}{n}\sum_{j=k+1}^{n}(p_j - Q)$$

可以发现这是一个关于F_i的方程，我们枚举k的值，对于每个k我们均解出F_i的值，之后我们取最大值即可。

对于$i = RG$时，我们可以和之前一样的枚举k，列出式子：

$$F_i = \frac{k}{n}\sum_{j=0}^{G-1}F_{i-G+j} + \frac{1}{n}\sum_{j=k+1}^{n}(p_j - Q)$$

求出F_{RG}即可。

4.3.4 时空复杂度

时间复杂度：$O(TkRGn)$其中k代表二分次数

空间复杂度：$O(RG + n)$

4.4 Google Code Jam 2013 Final E Let Me Tell You a Story

4.4.1 题目大意

有一个长度为n的数列a，你每次可以选择其中的一个数字删掉，直到它为一个不上升数列。问你有多少种不同的删除方法，答案对10007取模。

有T组数据。

4.4.2 数据范围

对于$1 \le n \le 100$，有$1 \le T \le 20$

对于$101 \le n \le 2000$，有$1 \le T \le 5$

$1 \le a_i \le 10000$

4.4.3 题解

我们先忽略删除到一个不上升序列为止的条件，对于一个长度为k的序列，我们一共有$(n-k)!$种不同的方法可以得到它。

接下来，我们去掉那些不可行的方案，对于一个长度为$k+1$的不上升序列，它一共会影响到$k+1$个不同的长度为k的不上升序列。用F_i表示长度为i的不上升序列共有多少个，可以得出答案为：

$$\sum_{i=1}^{n} F_i(n-i)! - F_{i+1}(n-i-1)!(i+1)$$

对于F_i的值，我们可以使用简单的DP来解决，在DP时需要用到树状数组维护前缀和，所以总时间复杂度为$O(n^2 \log_2 n)$

4.4.4 时空复杂度

时间复杂度：$O(Tn^2 \log_2 n)$

空间复杂度：$O(n^2)$

4.5 Google Code Jam 2012 Final C Xeno-archaeology

4.5.1 题目大意

现在有一个图形，它的正中央为'.'，然后由'#'和'.'一圈一圈包围起来，如下图所示：

```
###############
#.............#
#.###########.#
#.#.........#.#
#.#.#######.#.#
#.#.#.....#.#.#
#.#.#.###.#.#.#
#.#.#.#.#.#.#.#
#.#.#.###.#.#.#
#.#.#.....#.#.#
#.#.#######.#.#
#.#.........#.#
#.###########.#
#.............#
###############
```

现在我们不知道它的正中央在哪里，但是知道n个位置上的点是'.'还是'#'，求正中央的位置。若有多个可行的位置，输出与原点曼哈顿距离最近的一个点，若仍有多个解，输出有最大的x坐标的一个，若还有多个解，输出y坐标最小的一个。

有T组数据。

4.5.2 数据范围

$1 \le n \le 1000$，坐标绝对值在10^{15}以内，$1 \le T \le 50$

4.5.3 题解

假设中点已经确定了在(X, Y)，那么对于一个点(x, y)，它为'.'还是'#'，由$max(|x-X|, |y-Y|)$的奇偶性决定。当$max(|x-X|, |y-Y|)$为奇数时，(x, y)上的字符为'#'，否则为'.'。

那么，既然可以通过原点来确定每个点的字符，那么我们同样可以用每个点的字符对于原点有一定的限制。

若在(x_i, y_i)上有一个点'#'（对于'.'同理），那么我们对于原点(X, Y)有如下限制：

1. 若$x_i - X$和$y_i - Y$均为偶数，则任意(X, Y)均不可能为答案。
2. 若$x_i - X$和$y_i - Y$均为奇数，则任意(X, Y)均可能为答案。
3. 若$x_i - X$为奇数，$y_i - Y$为偶数，则满足$|x_i - X| > |y_i - Y|$的(X, Y)可能为答案。

4. 若$x_i - X$为偶数，$y_i - Y$为奇数，则满足$|x_i - X| < |y_i - Y|$的(X, Y)可能为答案。

我们可以枚举中点的X与Y的奇偶性，那么对于所有的已知点我们都可以确定出一个可行的坐标范围。可以发现，所有可行的坐标范围将整个图划分成了$(n+1)*(n+1)$的矩形，我们可以枚举每个矩形再判断，但是这样是$O(n^3)$的。

注意到我们如果枚举两条相邻的$X + Y = x_i + y_i$直线，那么对于每个$X - Y = x_i - y_i$的直线合并后必然构成一个的区间，这样我们就可以直接得到唯一可行的矩形，总矩形的个数是$O(n)$的，包括求出区间的时间，我们可以在$O(n^2)$的时间内解决。求出区间的过程可以用优先队列或set维护，这样时间复杂度就优化至了$O(n \log_2 n)$。

4.5.4 时空复杂度

时间复杂度：$O(Tn^2)$或$O(Tn \log_2 n)$

空间复杂度：$O(n)$

4.6 Google Code Jam 2011 Final A Runs

4.6.1 题目大意

有一个由小写字母组成的字符串。每一个极大连续相同的子序列被称为一个字段。比如说，“bookkeeper”有7 个字段。

问对于给定的长度为n的字符串s将其重新排列，有多少种不同的排列使得它和原字符串有相同的字段数，答案对于1000003取模。

4.6.2 数据范围

$1 \le n \le 450000$，s中字段个数不超过100个

4.6.3 题解

我们设$F_{i,j}$代表用了前i种字符，能够构成有j个字段的字符串共有多少个。

转移时枚举当前字符放在了多少个位置，假设当时放置了a个字段将之前的字段隔开，放置了b个字段在之前字段的缝隙中，那么会生成$j+2a+b$个

字段，则$F_{i-1,j}$对$F_{i,j+2a+b}$的贡献为：

$$F_{i-1,j}\binom{t_i-1}{a+b-1}\binom{j+1}{b}\binom{S_i-j}{a}$$

其中t_i代表第i种字符共有多少个，S_i代表从第一种字符到第i种字符共有多少个。

4.6.4 时空复杂度

时间复杂度：$O(26k^3)$

空间复杂度：$O(n+26k)$其中k指s中字段的个数

4.7 Google Code Jam 2010 Final C Candy Store

4.7.1 题目大意

你需要预先选一个可重数集。现在过来了n个人，每个人会从1到C中选择一个正整数，无论这n个人怎么选，你都要能在预选的可重数集中取出若干个不相交的子集，第i个子集内的数字和等于第i个人的数。

求你预选的满足条件的数集最小可能是多大。

有T组数据。

4.7.2 数据范围

$1 \le T \le 100$，$1 \le n \le 1000$，$1 \le C \le 10^{12}$

4.7.3 题解

我们依此来构造答案：假设当前已经选了的所有数总和为S，我们在其之上再选择大小为$\lfloor \frac{S}{n} \rfloor + 1$的数若干个，假设$x = \lfloor \frac{S}{n} \rfloor + 1$，那么我们所需要选的个数即为$\lceil \frac{n \cdot x - S}{i} \rceil$。

可以证明，当$S > nm$时，任意数列均可以由我们构出。

4.7.4 时空复杂度

时间复杂度：$O(Tn\log_2 C)$

空间复杂度：$O(1)$

4.8 Google Code Jam 2009 Final A Year of More Code Jam

4.8.1 题目大意

现在有一个长度为n的数组a，你对其进行m此操作，第i次操作会给你一个长度为e_i的数组d_i，你从0 到$n-1$ 中等概率的选择一个数s，然后将所有满足$s+d_{i,j}<n$的$a_{s+d_{i,j}}$全部加1。

问在操作m次之后$\sum_{i=1}^{n} {a_i}^2$的值期望是多少。

4.8.2 数据范围

$1 \le n \le 10^9$，$1 \le m \le 50$，$2 \le e_i \le 50$，$1 = d_{i,1} < d_{i,2} < d_{i,3} < ... < d_{i,e_i} \le 10000$

4.8.3 题解

还是先考虑暴力，我们设$F_{i,j}$为在处理完第i个操作之后，第j个位置上的值期望为多少。

若我们将第j个值加一，那么我们考虑$F_{i,j}$会增加多少。假设当前的长度为x，那么增加的值即为$(x+1)^2-x^2=2x+1$，所以我们只需要同时维护当前的期望长度$L_{i,j}$，每次转移时先求出当前一位加一的概率P，将$F_{i,j}$加上$P(2L_{i,j}+1)$，再将$L_{i,j}$加上P即可。

但是问题中的n有10^9，所以我们必须要优化一下。

可以发现，在第i次操作时，一个点x加一的概率为$\frac{\sum_{j=1}^{m}[d_{i,j}<x]}{n}$，可以发现，当$x > maxd_{i,j}$时，所有点的概率是一样的。所以我们对于所有$x$满足$x > maxd_{i,j}$，可以直接处理出一个值然后乘上个数即可。

4.8.4 时空复杂度

时间复杂度：$O(D\sum_{i=1}^{m} e_i)$

空间复杂度：$O(D)$其中D指d数组中最大的值

4.9 Google Code Jam 2009 Final B Min Perimeter

4.9.1 题目大意

平面上有n个点，现在要求选出3个点构成一个三角形，使得这个三角形的周长最短，注意若三点共线也是可以的。

问最短周长是多少。

4.9.2 数据范围

$0 \le n \le 10^5$

4.9.3 题解

这个问题和平面最近点对的做法十分接近。

我们先来回顾一下平面最近点对怎么做，我们将所有的点按照x坐标排序，分治算出左边一半和右边一半的答案。

接下来考虑跨越中线的答案，我们找出所有的在左边和右边的与中线的x坐标相差不超过答案的点。之后枚举在中线左边的点，从中线右边枚举与当前点y坐标相差不超过答案的点。

可以证明这样的复杂度是$O(n \log_2 n)$的。

思考如何推广到求三角形的周长。

有不同的仅有跨越中线的部分，分左半边一个点右半边两个点和左半边两个点右半边一个点的情况。

我们同样的也是找出与中线的x坐标相差不超过答案的点，以左半边一个点为例，那么我们先枚举中线左边的点，然后暴力用平方的时间在右边枚举另外两个点。

可以证明这样的时间也是$O(n \log_2 n)$的。

4.9.4 时空复杂度

时间复杂度：$O(n \log_2 n)$

空间复杂度：$O(n)$

4.10 Google Code Jam 2009 Final C Doubly-sorted Grid

4.10.1 题目大意

有一个$n*m$的矩阵，上面已经有一些位置已经有了字符，有些位置还没有字符，现在你需要在这个矩阵中补满小写字符，要求每一行从左到右都是不下降的，每一列从上到下也是不下降的。

问你一共有多少种不同的补全方案，答案对10007取模。

T组数据。

4.10.2 数据范围

$1 \leq n, m \leq 10$，$1 \leq T \leq 3$

4.10.3 题解

我们首先思考暴力DP。

设$F_{i,j}$代表我们从'a'开始放置字符，一直放置到了第i个时，当前的轮廓状态为j的放置方案数为多少。

转移时我们枚举每一行中所放置的当前字符个数即可，时间复杂度为$(m^n)^2$。

注意到每次我们枚举的轮廓必然是从上往下依次缩短的，所以我们可以将所有可行的状态记录下来，这样状态数变为了$\binom{n+m}{n}$。

而对于转移时枚举每位放多少个，我们可以直接使用前缀和来记录不合法的状态，即：$S_{i,j}$代表放置到了第i个字符，轮廓不超过j的方案数有多少。对于$S_{i,j}$如何维护我们可以采用类似于

4.10.4 时空复杂度

时间复杂度：$O\left(26 \cdot nm\binom{n+m}{n}\right)$

空间复杂度：$O\left(n\binom{n+m}{n}\right)$

4.11 Google Code Jam 2009 Final D Wi-fi Towers

4.11.1 题目大意

在平面直角坐标系上有n个点，每个点有一个权值s_i，现在你要选中其中任意多个点，要求若第i个点被选中了，那么与第i个点距离不超过R_i的点

均需要被选中。

问所选的点集权值和最大可能为多少。

T组数据。

4.11.2 数据范围

$1 \le n \le 500$，$1 \le T \le 55$，$1 \le R_i \le 20000$，$-1000 \le s_i \le 1000$，坐标绝对值不超过10000

4.11.3 题解

同样是一个简单的最小割问题。

对于每一个点我们在图中建出与其相对应的点，并且与S、T连边。若割掉从S连过来的边，代表选择这个点，若割掉向T去的边，代表这个点不被选。

我们将所有权值为正的点计入答案，则若该点不被选会损失s_i的代价，若该点被选不会有任何损失。

对于所有权值为负的点，选择该点会损失$-s_i$的代价，否则不会有任何损失。

现在要求当一个点x选了之后另外一个点y必须被选，那么我们从y向x连一条无限大的边，即不存在x选且y不被选的情况。

接下来跑一边网络流即可。

4.11.4 时空复杂度

时间复杂度：$O(T * maxflow(n, n^2))$

空间复杂度：$O(n^2)$

4.12 Google Code Jam 2009 Final E Marbles

4.12.1 题目大意

你在一个正方形网格中有$2n$个珠子。这些珠子被涂成了n种颜色，使得每种颜色恰好有2个珠子。这些珠子的坐标分别为$(1, 0), (2, 0), ..., (2n, 0)$。

你的任务是对于每种颜色画一条连接这两个珠子的路径。每条路径应由水平和竖直的连接格点的线组成。任意两条路径不能相交或触碰。任意

一条路径不能穿越直线$y=0$。每条路径只能在所连接的两个珠子处触碰直线$y=0$，所以每个路径的第一条和最后一条线段一定是垂直的。

给出珠子的排列，你需要算出可能的最小高度，或输出-1（如果无解）。一个方案的高度定义为所有路径中y坐标的最大值与最小值之差。

4.12.2 数据范围

$1 \le T \le 50$，$2 \le n \le 500$

4.12.3 题解

首先我们可以通过简单的暴力DFS来判断当前珠子是否有合法的连线方案，假设第i种颜色的珠子所在的位置为l_i和r_i，我们用一个区间$[l_i,r_i]$来表示这种颜色。那么，若存在a,b满足区间$[l_a,r_a]$ 与区间$[l_b,r_b]$部分相交，那么a和b必然在x轴的两边。所以我们在连完所有的边之后，判断图中是否有奇环即可。

可以发现，在一个连通块中，我们只需要确定出一个点，所有的边所在的方向即可全部确定。那么我们可以将所有有边的区间合并在一起，之后可以按照包含关系在新图中连边，这样就构成了一棵树。

那么我们设$F_{i,j}$代表第i个连通块的在x轴之上的部分高度不超过j，在x轴之下的部分高度最小为多少。

转移时，我们可以枚举当前块的两种情况，之后递归处理被其包含的块即可。我们可以直接枚举每个块所在的位置，那么它的上边界被限制的高度我们可以直接算出来，较为简单的判断即可。

4.12.4 时空复杂度

时间复杂度：$O(Tn^2)$
空间复杂度：$O(n^2)$

4.13 Google Code Jam 2008 Final E The Year of Code Jam

4.13.1 题目大意

在一个$n*m$的网格中，有些点为白色，有些点为黑色，有些点没有染色。每个黑点会贡献一个权值，一个黑点的权值为4−它周围的黑点个数。

现在要你将没有染色的点给染上黑色或白色，要求总权值最大，输出总权值。

有T组数据。

4.13.2 数据范围

$1 \leq n, m \leq 50$，$1 \leq T \leq 100$

4.13.3 题解

可以发现这同样是一个很经典的最小割转最大流问题。

对于一个点(x, y)，若$x + y$为奇数，那么从割掉S向它连的边代表该点为黑点，割掉它向T连的边代表该点为白点，若$x + y$为偶数则正好相反。

若两个相邻的点同时割掉了黑边，那么我们需要让其仍然需要割掉代价至少为2的边，所以我们将一个$x + y$为偶数的点与其所有相邻的点连一条容量为2的边。

直接跑最大流即可。

4.13.4 时空复杂度

时间复杂度：$O(T * maxflow(nm, nm))$

空间复杂度：$O(nm)$