练习地址: https://www.nowcoder.com/acm/contest/201#description

Problem Tutorial: "Attack on Titan"

容易证明只在整点处改变方向就是足够的。

Mikasa 某次在建筑 a 准备补充气体，假设上一次补充气体是在建筑 b，那么分两种情况

•$p_a \le p_b$

到达建筑 a 时气体余量正好为 0，否则减少在建筑 b 添加的气体余量花费不会变多。

•$p_a > p_b$

在建筑 b 气体余量加到了 C，因为若没加满，将在建筑 a 要加的油转移一部分给建筑 b 花费更少。

由此可以得出，在建筑 b 加完油后气体余量必是 $u+\sqrt{2}v(u\ge 0,v\ge 0)$，在任意建筑 x 的气体余量必为 $u+\sqrt{2}v(u\ge 0)$。

如果在建筑 (x,y) 拥有的气体余量超过到终点即 (n,m) 的距离显然是不优的。

我们可以用四元组 (x,y,a,b) 来描述 Mikasa 目前在建筑 (x,y) 拥有 $a+\sqrt{2}b$ 的气体余量这一状态。容易发现需要满足以下两个条件:

•$0\le a\le \min(C,dis_{(0,0),(n,m)})$

•$0\le a+\sqrt{2}b\le \min(C,dis_{(x,y),(n,m)})$

当 $n=m=20$ 时，上述四元组只有不到 1.5×10^5 个。

将每个四元组看成一个点，向四周的八个方向连边，以及最小的超过自己气体余量的点连边，然后跑最短路即可。

边权以及答案都可以用 $a+\sqrt{2}b$ 的形式来精确表示。

Problem Tutorial: "Utawarerumono"

为了方便比较不同整数解代入多项式的值的大小，我们先证明答案不超过 10^{18}。因为一次项的影响很小，所以我们只考虑二次项 $p_2*x^2+q_2*y^2=p_2*(\frac{c-by}{a})^2+q_2*y^2$，存在一个 $O(a)$ 的 $|y|$ 取值满足 $c-by$ 是一个 a 的倍数，此时 $|\frac{c-by}{a}|$ 是 $O(c+b)$ 的，这样得到了一组不超过 10^{18} 的解，答案不会更大。

然后发现形如 $p_2*x^2+p_1*x(p_1,p_2\ge 1)$ 的多项式在 x 的绝对值增加时，只有 $x<0$ 时才会变小，且当 $x<-\frac{p_1}{2p_2}$ 时其值仍然会增大，所以我们只需要暴力枚举 x 或 y 的绝对值在一定值（比如 $2*10^5$) 以下的解就可以了。

Problem Tutorial: "Love Live!"

先把树边按照权值从小到大排序

然后在加边的同时，用并查集维护所有的联通块

对于每个联通块，开一个 01-trie 树以求解"对于给定的 x，在集合中找一个数字 y 使得 $x\oplus y$ 最大"的问题

其中这部分是把每个点的点权记为 1 号点（实际上任何一个固定点都可以）到该点路径的边权异或和，那么两点点权的异或值就是两点间路径的边权异或和

每加一条边，都用启发式合并来合并两个联通块所对应的 trie 树，边合并边算答案

启发式合并有 $O(n\log n)$ 次操作，每次查询和插入是 $O(\log n)$ 的，所以时间复杂度 $O(n\log^2 n)$

Problem Tutorial: "Eustia of the Tarnished Wings"

把 d_i 从小到大排序，每个点和自己的相邻点合并，算一算有几个 $i, i+1$ 之间没有边跨过就可以了。

Problem Tutorial: "Baldr Sky"

设树的直径为 D，问题本质是求对于直径为 $1, 2, \cdots, D-1, D$ 的连通子图最多包含多少个点。

假设确定了直径为 d，设 $f_{u,l}$ 表示以 u 为根，直径不超过 d，距离 u 的最远点距离不超过 l 的子树最多包含的点数。

那么对于 u, v，其中 u 是 v 的父亲，当 $l_1 + l_2 + 1 \le d$ 时有转移 $f_{u,l_1} + f_{v,l_2} \to f_{u,\max(l_1,l_2+1)}$。这个 DP 的复杂度是 $O(nd)$。

可以证明以这种方式生成的树的期望直径是 $O(\log n)$ 级别的，总复杂度 $O(n\log^2 n)$。

以下是大概证明思路：

设 dep_i 为按题目描述方式所生成的第 i 个点的期望深度。

运用数学归纳法假设对于 $1\cdots n$ 都满足 $dep_i \le k\log i$，k 为任意给定常数。则有 $dep_{n+1} = \frac{\sum_{i=1}^{n} dep_i}{n} + 1 \le \frac{k\sum_{i=1}^{n}\log i}{n} + 1 \approx k(\log n - \log e) + 1$（这一步用到了斯特林公式即 $\lim_{n\to\infty} \frac{n!}{\sqrt{2\pi n}(\frac{n}{e})^n} = 1$），当 k 取得足够大时，其小于 $k\log(n+1)$。所以 dep_i 为 $O(\log i)$ 级别，则 n 个点的树期望直径也为 $O(\log n)$ 级别。

Problem Tutorial: "Kimi to Kanojo to Kanojo no Koi"

有三种情况：

1、n 是奇数，我们可以构造如下的方阵：

1, 2, 3, ... n-2, n-1, n n, 1, 2, ... n-3, n-2, n-1 ... 3, 4, 5, ... n, 1, 2 2, 3, 4, ... n-1, n, 1

可知对于所有的 i,j (1<=i<j<=n), A[i][j]+A[j][i]=n+2, 所以 A[i][j] 不可能等于 A[j][i]。

2、n=2，无解。

3、n 为大于 2 的偶数：当 n = 4 的时候，有这样一组解：

1, 2, 3, 4 4, 3, 2, 1 2, 1, 4, 3 3, 4, 1, 2

当 n>4 的时候，我们假设 n=2k (k>2), 并且当 n=k 时存在一组合法解，那么我们构造 n=2k 时的解：记 A(t) 为 n=t 时的一组解。首先，我们先令 A(2t) 为：

A(k), A(k) A(k), A(k)

之后, 我们在左上和右下的 A(k) 中给所有元素加上 k：

A(k)+k, A(k) A(k), A(k)+k

之后，我们把右上的 A(k) 沿着它的主对角线翻转一下并记为 A'(k)，此时我们可以得到：对于所有在 A'(k) 中的 A(2k)[i][j]，有 A(2k)[i][j] = A(2k)[j][i]，具体如下。

A(k)+k, A'(k) A(k), A(k)+k

最后, 我们只要把左下的 A(k) 的值都移一位就好。(1->2, 2->3, ..., k->1):

A(k)+k, A'(k) A(k)+1, A(k)+k

这样 n=2k 的解就构造出来了。

Problem Tutorial: "One Piece"

由于是 01 数列，故其最长上升子序列的形态为 0000...111。

考虑直接枚举满足条件（即 0 最多）的最长上升子序列中最后一个 0 的位置。接下来考虑这个位置在何时是合法的：即，从这个位置开始向后，遇到 1 就给计数器 +1，遇到 0 就给计数器-1，计数器始终 >0，并且，从这个位置开始向前，遇到 0 就给计数器 +1，遇到 1 就给计数器-1，计数器始终 >=0。

注意这里向前走，计数器是可以 =0 的，向后走则不可以：这是因为一方面我们不能重复计数，另一方面需要保证枚举的位置是尽可能靠后的位置。

实现时可以在序列最前端加一个始终为 0 的数，在最末端加一个始终为 1 的数，这样对于数列为全 0 或全 1 的情况不用特判。

复杂度为 $O(N^3)$

Problem Tutorial: "Steins;Gate"

对质数 P，计算其原根为 g。这样可以将 a_i 写成 g^{b_i} 的形式。于是乘法就变成了质数的加法，直接利用 FFT 进行计算即可。需要注意的是要对 $a_i = 0$ 的情况特殊处理。

Problem Tutorial: "Princess Principal"

先判断区间中每种括号单独拿出来之后是不是合法的。对于一个括号，无论查询区间的左端点和右端点是什么，它匹配的括号的位置是确定的（如果它能和另一个括号匹配的话）。所以对每种括号维护一个栈，确定一下每个括号匹配的括号是哪个，然后即是问区间中所有的左括号的匹配位置是否 $\leq r$，右括号的匹配位置是否 $\geq l$。

这样可以判断区间中每种括号单独拿出来之后是不是合法的，但还要注意形如 acbd 这样的形式也是非法的，其中 a,b 是一种括号，c,d 是另外一种括号。可以使用一个含有所有种类括号的栈来判断是否有不同种类的括号夹在两个匹配的括号之间。如果有一个括号被其他种类的括号夹住了，那么含有这个括号的所有区间都是不合法的。

如果使用 $O(n) - O(1)$ 的 RMQ，那么总的时间复杂度为 $O(n + q)$。

Problem Tutorial: "Tengen Toppa Gurren Lagann"

题意：给出一个长度为 n 的排列和常数 k，问能否将这个 n 排列分成 k 段，对每段进行排序后，序列变为递增的。为了增加你的成功率，你可以（但不是必须）将其中的任意两段交换，这个操作至多进行一次。
1<= n <= 1000000

解法：假设我们已经找到将序列分为 K 段的一个解。对于一个 K 段的解，假如没有进行交换操作，我们可以不断合并相邻的两段以减少段数直至段数为 1。假设进行了交换操作，则需要从交换了的两个段之间的段开始合并，仍然可以不断减少段数直至段数为 1。

那么问题转化为求出最多能分成几段。考虑不交换两段的情形，可以很容易的贪心求出最多分成若干段，

记这时候的这些段为原始段。考虑被交换的两个区间在序列中的位置，容易发现，交换的前段的前端必定与一个原始段的前端重合，否则必定导致最后的序列不是递增的。类似的，后段的后端必定与一个原始段的后端重合。进一步的，可以得到交换的前段和后段必定在同一个原始段中，且是这个原始段的前缀与后缀。那么，我们只需要扫描每个原始段，求出对于这个原始段的最优交换方案即可解决此问题。

于是问题转化成，给定一个 n 排列，将它分成若干段并段内排序，交换第一段和最后一段，使得整个序列是递增的，问最多分成多少段。用与之前类似的贪心法从前往后扫描将序列分成尽量多段，每一段内元素依次是 [x1, n], [x2, x1), [x3, x2) ... [1, xk)。显然的是，交换的前缀和后缀必定由这个分段中的前若干段和后若干段组成。换言之，即是在这个分段中选取不包括第一段和最后一段的连续若干段作为中央部分。稍加思考，可以发现，如果中央部分包含不止一个分段，则在原问题中中央部分必然只能分成一段。于是我们可以不劣地推定，中央部分必定只由当前分段的一个分段组成。那么，只需要对当前分段的除头尾外所有段，执行一次原问题的贪心分段，即可算出原问题的答案。

复杂度 O(n)

Problem Tutorial: “New Game!”

L_1 到 L_2 之间连边权值 $\frac{|C_1-C_2|}{\sqrt{A^2+B^2}}$

线 L 与圆 i 之间连边权值 $\max(0, d(O_i, L_1) - r_i)$

圆 i 与圆 j 之间连边权值 $\max(0, d(O_i, O_j) - r_i - r_j)$

求 L_1 到 L_2 的最短路即可。