Problem A :Average Rank

题目大意：一共有 n 个人，w 周时间，第 i 周有 ki 个人分数+1，问每人在所有周的排名的平均值

对于某个人 i，排名=分数比 i 高的人数+1，所以对于每个人在连续的几周中分数保持为 p 不变时，可以将每周分数高于 p 的人数相加，要计算高于某个分数的人数，由于每个人的分数随时间不降且每次增加都是+1，于是在一个人分数从 x 变为 x+1 那周及之后的所有周，他的分数一定大于 x，于是计算人数时只需要计算所有人中分数变为 p+1 的时间点之后的那段区间中处于当前人分数为 p 的区间内的和，需要区间修改和区间查询操作，可以用线段树实现，时间复杂度 o(nlogw)。

Problem B :Balanced Cut

有一棵 n 个点的平衡二叉搜索树，n 个数字分别是 1~n。我们需要删去一些点使得剩下的 k 个点仍然是平衡二叉搜索树。可能有多种删法，要求剩下的树的中序遍历字典序最小。

对于一棵深度为 i 的平衡二叉树，它的左右两侧至少是深度为 i-1 和 i-2 的平衡二叉树。

定义 f(i)为一棵深度为 i 的平衡二叉树至少需要的节点数。

f(i)=f(i-1)+f(i-2)+1，f(1)=1，f(2)=2。

因为需要字典序最小，所以我们贪心地从小到大考虑各个节点能不能选入 k 个点中，能放就放。

考虑加入节点 i，可能会影响它所有的祖先的平衡，所以需要花费一定的代价来平衡。

对于它的每一个祖先 p。

如果 i 来自 p 的左子树 :p 左子树原本的深度为 d1，加入 i 后 p 的左子树深度为 d2。为了使代价最小，所以右子树原本的深度为 d1-1，加入 i 后右子树需要将深度增加到 d2-1。我们需要给右子树添加 f(d2-1)-f(d1-1)个点使树平衡。

如果 i 来自 p 的右子树 :因为我们从小到大考虑，所以 p 的左子树已经确定了。只要加入 i 后右子树深度不要比左子树深度大 2，那么对于这个祖先 p 就不会有额外代价。

如果代价加 1（这个点本身也是 1 代价）小于等于当前剩余的 k，就将 i 加入。

Problem C :Canvas Line

本题为给出一些不会重叠的帆布坐标以及以给出的一些夹子的坐标，求使每个帆布有且仅有两个夹子固定所最少需要额外的夹子数目以及坐标，否则输出 impossible。

思路：使用 map 记录某坐标是否有夹子，使用贪心思想先让夹子尽可能的出现在布的顶点，可以让两块布共用一个夹子，然后通过后一块布的夹子数量去维护这个夹子的位置。

Problem D :Disposable Switched

有 n 个点，有 m 条线连接这 n 个点。在一条线上走花费的时间为 l/v+c，l 是线的

长度，v 是任意的正数，c 是任意的非负数，但对于每根线 v,c 都是一样的。问，从点 1 沿着最短路走到点 n，无论 v,c 等于多少，哪些点是绝对不会经过的。

对于一条由 k 条线组成的路径，所花费的时间为

$$T = \sum_{i=1}^{k} (\frac{l_i}{v} + c)$$

乘上 v 后为

$$T * v = \sum_{i=1}^{k} (l_i + v * c)$$

也就是 k 条线的长度和加上 k*v*c。所以对于由 k 条线段组成的路径，线的长度和越短越好。设 f(i)为由 i 条线段组成的最小的线长度和。

我们可以用 dp 求出所有 f(i)的大小。

对于每一个可能的 x 条线段组成的路，它的花费为

$$Ft(x) = T * v = \mathrm{f(x)} + \mathrm{x} * \mathrm{v} * \mathrm{c}$$

把它看作是随着 v*c 不断增长的一次函数。题目则转换成有若干个一次函数，求出定义域为[0,+∞) 时，对于每个函数，是否有某段区间使得它是最小值。

Problem E: Expeditious Cubing

给出四个时间，问能否有一时间使去最大和最小后的平均值达到预期。总是可以输出 infinite，不可以输出 impossible，否则输出满足条件的最大数。

思路：满足条件不难推断，浮点数比较大小使用 ecp 更加稳妥。

(Sum-min)/3<=t, 则为恒成立；(sum-max)/3>t, 则为恒假；否则输出 t*3-sum+min+max;

Problem F: Firetrucks Are Red

题目大意：有 n 个点，第 i 个点有 mi 个值，有相同值的点间有一条双向边相连，求一棵生成树

先将所有值离散化，然后储存每个值涉及的点，对于每个值的所有点，除第一个点外的所有点都取链接到第一个点的边，如果已经联通则不用取，这里用并查集来判断是否已经联通

Problem G: Gnoll Hypothesis

题目大意：有 n 个怪物，每个怪物有一个生成概率(所有概率的和为 1)，要生成 k 个怪兽，如果某个怪兽生成失败，则它的生成几率要加到下一个生成成功的怪兽上，求每个怪兽的生成几率的数学期望

怪兽 i 生成的初始概率为 s[i]，对于一个怪兽，如果它前面有连续 x 个怪兽生成失败，那么它生成成功的概率对期望的贡献为(C(n-x-2,k-2)/C(n,k))*(s[i-x]+s[i-x+1]+…+s[i-1]+s[i])，于是可以枚举 x，然后枚举生成成功的怪物 i，时间复杂度 o(n2)

Problem H :Height Profile

有一段凹凸不平的路，这条路可以看做是若干条斜线构成的。问，对于每一个给出的斜率，在路上找出水平距离最远的两个点使得这两个点之间的连线斜率不小于给出的斜率。

可以很简单地发现，如果可以找到这样的两点，那么最优的情况一定是，两个点

中至少有一个点在端点上。

对于每一个斜率 k，将第 i 个节点减去 k*i，形成新的图。在新的图上斜率大于等于 0 就是符合条件的。

我们可以枚举右端点。对于同一个右端点，如果点 a 比点 b 更靠近右端点并且点 a 不小于点 b，那么点 a 就不需要考虑了，因为点 b 一定比它更好。所以从所有小于右端点的点中取出一个单调递减的数组，二分查找出最大的不大于右端点的点作为左端点。

尝试将左端点往左移动 1km，将右端点往右移动 1km，找出最大值。

Problem I :Inverted Deck

给出 n 以及 n 个数，问是否可以通过最多一次区间反转使得其按照非递减排列。

思路：填一个数组记录 n 个数按照非递减排列的结果，然后从头和从尾依次找出一个不相同的数的下标，开始反转，最后再比较两个数组是否一样即可。

Problem J :Jackdaws and Crows

有一串整数。对于这串整数，你有两种操作方式。一种是选择任意个数字加 1 并选择任意个减 1，花费代价 c，另一种是直接将一个数字删去，花费代价 r。问，将这串整数变得正负整数交替，至少需要多少代价。

如果操作一的次数大于某个数字的绝对值，那么这个数字可以是正数，也可以是负数，而如果不大于则不会有什么改变。所以我们从小到大枚举操作一的次数。

在操作一次数一定的情况下，最少需要的操作二次数可以计算出来。

根据操作一的数量将数组分为任意正负的数字和确定正负的数字。根据两个相邻的确定数字的正负关系以及这两个数字之间任意数字的个数可以判断这两个数字是否冲突，冲突则需要操作二。

随着操作一数量的增加，也可以在 o(1)的时间里维护操作二的数量。

例如，设函数 f(a,b)，a 和 b 是相邻的确定正负的数字，需要操作二为 1，不需要为 0。

有三个确定正负的数字 a,b,c，这时的操作二为 f(a,b)+f(b,c)。

将 b 转化为任意正负的数字，这时的操作二为 f(a,c)。

所以操作二数量变化为 f(a,b)+f(b,c)-f(a,c)。

Problem K :Kitesurfing

冲浪比赛，你可以以每秒一米的速度向前划或者花费 t 秒跳一下，跳跃距离为不大于 d 的任意数字。路径上有若干岛屿，你不能在岛屿上划，但是可以跳过岛屿。所有岛屿宽度不大于 d。

对于下一个岛，有三种可能，划到右端点-d 的位置然后跳到右端点，或者跳到左端点再跳一次跳到左端点+d，或者不必特意找跳跃点，这个岛正好被某次跳跃跳过了。前两种可能的跳跃结束点都是确定的，第三种可能性的结束点并不确定，对于每个岛，个数为 $o(n^2)$级。用 dp 的方法，使用当前的跳跃结束点跳过下一个岛，更新出新的跳跃结束点，复杂度为 $o(n^3)$。