**有理逼近**

最容易想到的方法：

对于最终要求的：x/y<sqrt(p)<u/v， x与y从1到N全部枚举一次，u与v也全部枚举一次，一定可以得到正确答案，复杂度为O(N^2)，但是N<30000，会超时。但当y确定了后，x没有必要从1开始枚举，可以从floor(sqrt(q))\*y开始，ceil(sqrt(q))\*y结束，对u,与v的枚举也是同理，如此可以直接枚举通过。

提醒：

由于除法与sqrt都会有精度问题，因此最好尽量避免，例如x/y<sqrt(q)的判读，最好使用：x^2<q\*y^2的方式。

扩展：枚举使用二分优化也是可行的。

**账本**

数列转换：+当作1，-当作-1

显然可知的是：

为了满足条件1，只能通过操作1得到，且操作1的数量是确定的：如果数列和为K，K≤Q-P，则需要将（Q-P-K）/2个减号变成加号，K≥Q-P，则需要将（Q-P-K）/2个加号变成减号。

为了满足条件2，我们需要使前缀和尽量大，因此，如果要减号变加号，则尽量变前面的，加好变减号，则尽量变后面的。变完之后，再来看序列，如果序列的前缀和最小为M，当M>=0时，序列已符合要求，如果M<0时，如果只使用操作1，则将需要将前A个-号改成+，后A个+改成-，其中A=(1-M)/2;

再考虑操作2，枚举操作2的次数，看最少花费时间是多少。

时间估计：将K算出来为O(1)，枚举操作2次数后，计算时间，主要花费在计算M上，需要O(n)的时间，则总体需要O(n2)

操作2可以看作将序列整体向后滚动了一次（最后一个跳到第一个去了），将序列复制一次后，我们可以使用单调队列在O(n)时间内将所有的M值全部计算出来。

提醒注意点：需要使用long long

**过路费**

即为给定任意点对，求所有两点之间可连接两点的路径中的最大边最小值。二分的思想是显而易见的，但是只能得30分。

仔细分析一下题目，会发现如果要任意两点之间的路径中最大边最小，整个图便符合了最小生成树性质！！证明如下：

若存在从a到b的一条比最小生成树上连接a与b的所经过路径上边更小的边c，则将c加入最小生成树中可得到一个更小的生成树，这显然不符合最小生成树的定义，故不存在这样的边c。

其实最小生成树克鲁斯卡尔算法的思想就是按照选择两点之间权值最小的边来构建的。

大概算法就出来了，读入数据，克鲁斯卡尔算法求最小生成树，但这时会发现一个严重问题：怎样处理询问？n的范围过大，dist[n][n]无法存储，枚举根节点来dfs求到任意节点的答案，则算法时间上升到O(n^2)，只能得30分。

再分析一下，对于一棵以任一结点为根的已经建好的最小生成树，此时询问无非是两种情况 1：两个节点为直系祖先关系，可直接求得解。2：两个节点为不同分枝上的节点。求这两个节点的答案难以想出好办法。但可以知道两个节点之间只存在唯一一条简单路径，这条唯一简单路径上的变的最大值即为答案，且这两个结点的最近公共最先必然在这一点上。想到可以转化为LCA算法，dfs一边，当处理完当前节点所有的子节点后开始回答关于这个节点的问题。