**CST2014 1-1 BigInt**

**描述**

　　邓俊辉老师的作业常常过于简单，数据类型只需使用int。助教们一致认为，向同学们介绍Python中自带的长整型是十分有必要的。例如，它可以计算几百位的整数乘法。 但是，在介绍长整型之前，助教决定让你自己实现一遍长整型乘法，以加深对它的理解。

**输入**

　　输入共包含n+1行，第1行包含一个整数n，表示你需要计算n组乘法。

　　接下来n行，每行包含两个非负整数a和b。

**输出**

　　输出共包含n行，请对于每一组输入的a、b，输出他们的乘积。

**输入样例**

3

1 1

2 2

123123 789789

**输出样例**

1

4

97241191047

**限制**

n≤100

a≤10^200

b≤10^200

**提示**

由于计算机的存储字节有限，所以不能完整表示一个很大整数的精确值，这时候就得用到其他的方法，称之为高精度算法。这里，主要说下高精度乘法。高精度乘法，实际上，就是模拟乘法的过程，像小学的笔算过程。

以加法为例：

12345678910111213 + 1111111111111111111

使用两个数组存储：

a[]={3,1,2,1,1,1,0,1,9,8,7,6,5,4,3,2,1};

b[]={1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1};

两个数组分别把数值倒存,逐位相加,每位加后判断是否大于10,再进行进位，如果数值的规模相当大，可以考虑多位一存节约空间。

对于高精度的乘法，则是进行多次的加法，并将结果统一相加而已。

这样的算法是O(n^2)级别的，已经可以解决本题。实际上，存在基于FFT（快速傅立叶变换）的算法，可以达到$O(n \log (n))$的时间复杂度。 有兴趣的同学可以自行阅读相关材料进行实现。

**CST2014 1-2 Zuma**

**描述**

　　祖玛是一款曾经风靡全球的游戏，其玩法是：在一条轨道上初始排列着若干个彩色珠子，其中任意三个相邻的珠子不会完全同色。此后，你可以发射珠子 到轨道上并加入原有序列中。一旦有三个或更多同色的珠子变成相邻，它们就会立即消失。这类消除现象可能会连锁式发生，其间你将暂时不能发射珠子。



　　开发商最近准备为玩家写一个游戏过程的回放工具。他们已经在游戏内完成了过程记录的功能，而回放功能的实现则委托你来完成。

　　游戏过程的记录中，首先是轨道上初始的珠子序列，然后是玩家接下来所做的一系列操作。你的任务是，在各次操作之后及时计算出新的珠子序列。

**输入**

　　第一行是一个由大写字母'A'~'Z'组成的字符串，表示轨道上初始的珠子序列，不同的字母表示不同的颜色。

　　第二行是一个数字n，表示整个回放过程共有n次操作。

　　接下来的n行依次对应于各次操作。每次操作由一个数字k和一个大写字母Σ描述，以空格分隔。其中，Σ为新珠子的颜色。若插入前共有m颗珠子，则k∈[0,m]表示新珠子嵌入之后（尚未发生消除之前）在轨道上的位序。

**输出**

　　输出共n行，依次给出各次操作（及可能随即发生的消除现象）之后轨道上的珠子序列。

　　如果轨道上已没有珠子，则以“-”表示。

**输入样例**

ACCBA

5

1 B

0 A

2 B

4 C

0 A

**输出样例**

ABCCBA

AABCCBA

AABBCCBA

-

A

**限制**

0<n<10000

0≤m<10000

**提示**

使用数组或链表维护当前序列，对于各个操作进行相应的模拟操作，注意各个细节情况的处理即可。

**CST2014 1-3 Stacksort**

**描述**

　　栈是一种强大的数据结构，它的一种特殊功能是对数组进行排序。例如，借助一个栈，依次将数组1, 3, 2按顺序入栈或出栈，可对其从大到小排序：

　　　　1入栈；3入栈；3出栈；2入栈；2出栈；1出栈。

　　在上面这个例子中，出栈序列是3, 2, 1，因此实现了对数组的排序。

　　遗憾的是，有些时候，仅仅借助一个栈，不能实现对数组的完全排序。例如给定数组2, 1, 3，借助一个栈，能获得的字典序最大的出栈序列是3, 1, 2：

　　　　2入栈；1入栈；3入栈；3出栈；1出栈；2出栈。

　　请你借助一个栈，对一个给定的数组按照出栈顺序进行从大到小排序。当无法完全排序时，请输出字典序最大的出栈序列。

**输入**

　　输入数据共两行。

　　第一行包含一个整数n，表示入栈序列长度。

　　第二行包含n个整数，表示入栈序列。输入数据保证给定的序列是1到n的全排列，即不会出现重复数字。

**输出**

　　仅一行，包含n个整数，表示你计算出的出栈序列。

**输入样例**

3

2 1 3

**输出样例**

3 1 2

**限制**

1≤N≤104

**提示**

解法是基于贪心算法的思路。

栈的性质就是先进后出，所以对于依次入栈的两个元素a,b，有两种情况：

<1> 先压a进栈，不弹出，再压b进栈，则最终的出栈序列a一定位于b之后。

<2> 要想a位于b之前，则只能在b进栈之前把a弹出。

因此，为了出栈序列字典序最大，只需考虑每次出栈者均为最大即可。

设最大值为a[l]，首先将 a[1],a[2],...,a[l]均入栈，然后弹出最大值，接着考虑a[l+1],...,a[n]的最大值a[m]与栈顶元素S[top]的关系。 继续进行类似操作，设法弹出二者中的较大值即可。

**CST2014 1-4 Zauberberg**

**描述**

　　住在魔山上的生物，近来纷纷染上某种怪病。NC检疫局认为，在魔山上住得越高，得病的可能性越大。于是他们企图通过削平山顶，灭杀掉某一海拔以 上（含）的所有生物。尽管其中有些的确已经染病（杀中），有些其实尚未染病（误杀）；而在这一海拔以下的生物中，有些仍可能已经染病（漏杀）。当然，他们 既不希望误杀过多，也不愿意漏杀太多。于是为制定可行的灭杀方案，需兼顾两项指标：

　　　　杀中率 = 杀中数量/染病生物总数

　　　　误杀率 = 误杀数量/健康生物总数

　　可行的灭杀方案须满足的条件是：杀中率不低于某一指定下限（记作Phit），而且误杀率不高于某一指定上限（记作Pfalse）。

　　实际上，我们已知魔山上所有生物的普查数据，即它们各自居住的海拔高度以及目前是否染病。注意，同一海拔上可能同时住有多个生物，且他们在不同 高度上的分布完全随机，没有任何规律可循。现在，对于检疫局需要考察的多种(Phit, Pfalse)组合，请你尽快算出对应的可行灭杀方案。

**输入**

　　第1行包含两个正整数N、H，分别表示魔山上的生物总数和魔山的高度。

　　接下来N行，每行用来描述一种生物的信息，包含一个正整数h表示居住高度以及一个字符c，'+'表示阳性染病，'-' 表示阴性无病。

　　随后1行包含一个正整数M，表示需要制定的灭杀方案总数。

　　接下来M行，每行包含两个实数（精确到6位小数），对应题目中的一个(Phit, Pfalse)的组合条件，取值范围均为[0, 1]。

**输出**

　　共M行，依次给出M个灭杀方案。

　　如果存在可行方案，则对应的行包含以空格分隔的两个正整数hL≤hH，表示按照对应的(Phit, Pfalse)组合条件，从hL到hH均可作为灭杀的起始高度。否则，对应的行仅含一个整数-1。

**输入样例 1**

5 12

10 +

8 -

7 +

6 +

5 -

3

0.500000 1.000000

0.300000 0.500000

0.300000 0.100000

**输出样例 1**

0 7

6 10

9 10

**输入样例 2**

10 10

4 +

6 +

0 +

9 +

2 +

3 -

9 -

7 -

9 -

5 -

15

0.818784 0.971310

0.828424 0.955460

0.197642 0.972830

0.209844 0.000000

0.367560 0.826768

0.197130 0.608106

0.183518 0.182270

0.192008 0.581226

0.208676 0.000000

0.435384 0.770400

0.395772 0.850256

0.214818 0.000000

0.209772 1.000000

0.207724 0.414344

0.196300 0.613776

**输出样例 2**

-1

-1

4 9

-1

4 6

6 9

-1

8 9

-1

-1

4 6

-1

0 6

-1

6 9

**限制**

2≤N≤100,000 ，其中染病和健康的生物至少各有一个

1≤H≤10^9

1≤M≤200,000

**提示**

考虑到PHit和PFalse的单调性，可以利用PHit求出h的下界，利用PFalse求出h的上界。

由于这两个问题的对称性，我们只考虑前者。对于每个查询，可以利用二分查找来进行求解。

**CST2014 1-5 Graphics**

**描述**

　　小Q对计算几何有着浓厚的兴趣。他经常对着平面直角坐标系发呆，思考一些有趣的问题。今天，他想到了一个十分有意思的题目：

　　首先，小Q会在x轴正半轴和y轴正半轴分别挑选n个点。随后，他将x轴的点与y轴的点一一连接，形成n条线段，并保证任意两条线段不相交。小Q 确定这种连接方式有且仅有一种。最后，小Q会给出m个询问。对于每个询问，将会给定一个点P(Px,Py)，请回答线段OP与n条线段会产生多少个交点？

　　小Q找到了正在钻研数据结构的你，希望你可以帮他解决这道难题。

**输入**

　　第1行包含一个正整数n，表示线段的数量；

　　第2行包含n个正整数，表示小Q在x轴选取的点的横坐标；

　　第3行包含n个正整数，表示小Q在y轴选取的点的纵坐标； 第4行包含一个正整数m，表示询问数量；

　　随后m行，每行包含两个正整数Px,Py，表示询问中给定的点的横、纵坐标。

**输出**

　　共m行，每行包含一个非负整数，表示你对这条询问给出的答案。

**输入样例**

3

4 5 3

3 5 4

2

1 1

3 3

**输出样例**

0

3

**样例说明**

　　3条线段分别为：(3, 0)-(0, 3)、(4, 0)-(0, 4)、(5, 0)-(0, 5)

　　(0, 0)-(1, 1)不与他们有交点，答案为0。

　　(0, 0)-(3, 3)与三条线段均有交点，答案为3。

**限制**

对于部分数据：n≤2000，坐标范围 ≤10^8

对于全部数据：n≤200000，坐标范围 ≤10^8

**提示**

二分查找、ToLeft测试。

对于每条线段，可以用向量叉积的ToLeft测试来判断其在线段的左边还是右边。通过二分查找， 可以高效的查找点在哪个区域内，从而确定交点的数目。