阿狸的统计学

（count.cpp/c/pas）

Time Limit:2 Sec Memory Limit:512 MB

Description

阿狸有一个长度为n的数组A，有m个操作，操作有两种：

* 1 l r x 表示将Al..r的元素改为x。
* 2 l r 表示询问Al..r的元素中在当前区间出现率≥p%的元素是那些。

出现率的定义是，(某个元素在区间中出现的个数)/(区间中总共的元素个数)。

鉴于本题难度可能比较大，且阿狸只关心那些出现频率高的数，本题有SPJ，对于操作2的回答格式如下：

一个回答对应一行，第一个数是k，表示有多少个数出现率≥p%，必须保证k≤，接下来k个空格隔开的数，表示这些数。只要这k个数包含了标准答案（也就是，所有出现率≥p%的数都出现在了这k个数中），你就答对了这个询问。

Input

有若干行。

第一行三个正整数n、m、p，表示A数组的长度、操作数和常数p。

第二行有n个正整数，描述A数组。

接下来m行，每行都形如1 l r x 或 2 l r，表示操作。

Output

有若干行，表示对应操作2的答案。

Sample Input

5 9 33

1 2 1 3 3

2 1 5

2 1 5

2 1 3

2 3 3

1 2 4 5

2 1 5

2 3 5

1 4 5 1

2 1 5

Sample Output

3 1 2 3

2 1 3

2 2 1

3 1 1000 1000

1 5

2 5 3

2 1 5

Data Limitation

对于测试点1，保证n,m≤100。

对于测试点2~3，保证n,m≤5,000。

对于测试点4，保证n,m≤30,000，且没有1操作。

对于测试点5，保证n,m≤30,000。

对于测试点6~7，保证n,m≤80,000。

对于测试点8，保证p=100。

对于测试点9~10，保证p=50。

对于测试点11，保证Ai≤2，x≤2。

对于测试点12，保证Ai≤10，x≤10。

对于测试点13~14，保证没有1操作。

对于100%的数据，保证1≤n,m≤150,000，20≤p≤100，1≤Ai≤150,000，1≤l≤r≤n，1≤x≤150,000。

阿狸的统计学

（count.cpp/c/pas）

Time Limit:2 Sec Memory Limit:512 MB

【题意简述】

区间修改，区间查询出现率超过p%的数。

【解题思路】

这道题一看就知道是数据结构，再一想，就知道是线段树了。那么线段树可以处理怎么样的问题，又或者，要满足怎样的性质才能用线段树呢？

答案是：区间信息可以简化成或，并支持合并。（一般来说是这样的>\_<）

那么这道题，乍一眼看，发现区间信息不能用的空间存下来，就算存下来也不能支持区间合并，怎么办呢？

我们先来看一道题（[BZOJ2456--mode](http://www.lydsy.com/JudgeOnline/problem.php?id=2456)）：有一个n个数的数列，其中某个数出现了超过次（即众数），找出那个数。内存限制1MB。

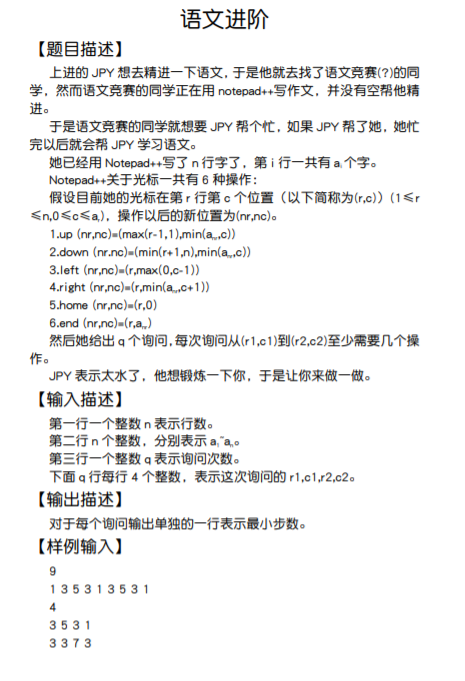
这道题怎么做呢？方法很巧妙，这个数的出现次数比其他所有数出现的次数都要多，所以，对两两不相同的数相互抵消，最后剩下的肯定全是这个数。那么就开两个变量num和tot，依次读入所有的数，读完第i个数时，num和tot的意义是，对前i个数进行配对抵消，剩下的那个数是num，且num还剩下tot个，对于新读入的数x，如果x≠num，那么tot--（即把num与x配对），否则tot++。如果tot变成0了，那么num的值改成x。

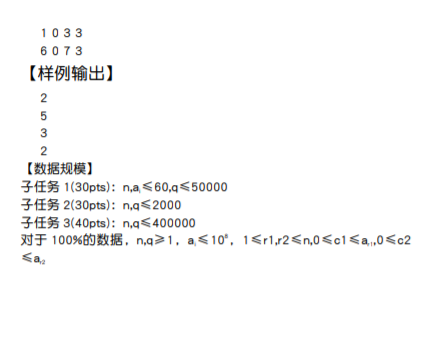
那么如果有一个n个数的数列，其中有个数出现了超过次，如何找出那些数呢？将个不相同的数相互抵消，最后剩下的就是要找的那些数了。

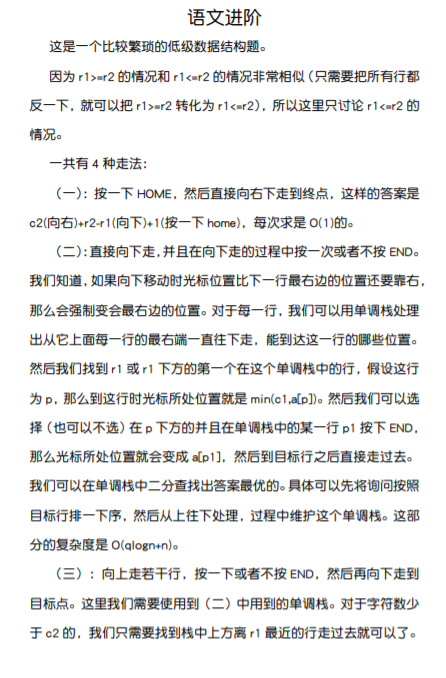
知道了怎么用的空间存下一段区间的信息，那么如何合并两段区间的信息呢？我们可以直接用的时间复杂度暴力和并。接下来就是线段树的事了，总复杂度为。

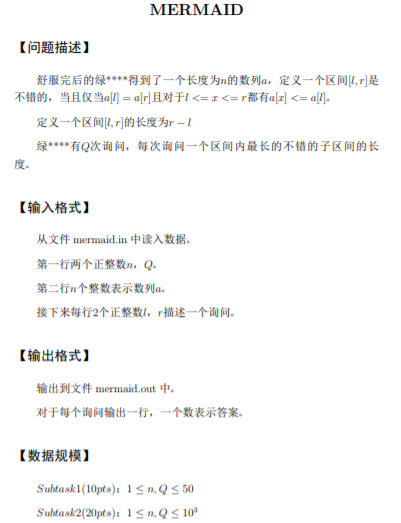
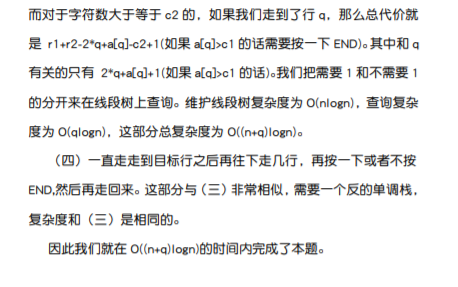
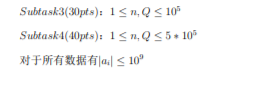
另外还有很多做法：分块、维护区间出现次数前k大的数、随机算法+区间查询一个数出现的次数等等，就不(wo)再(ye)讨(bu)论(hui)了

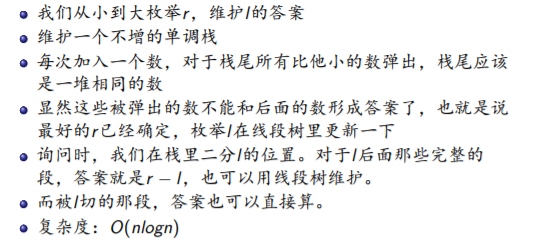
【题目来源】CF643G

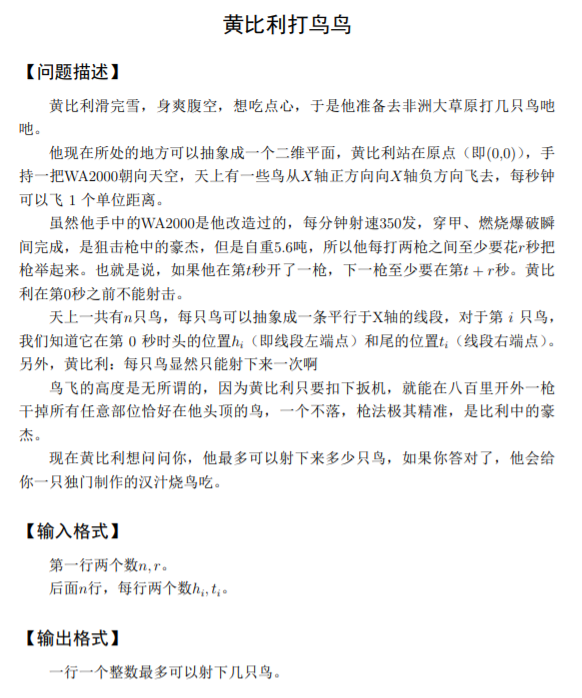


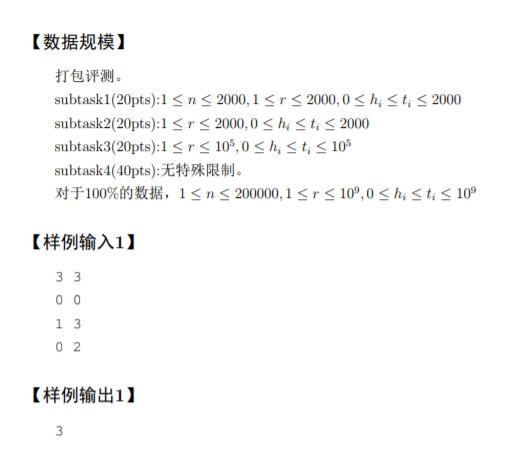
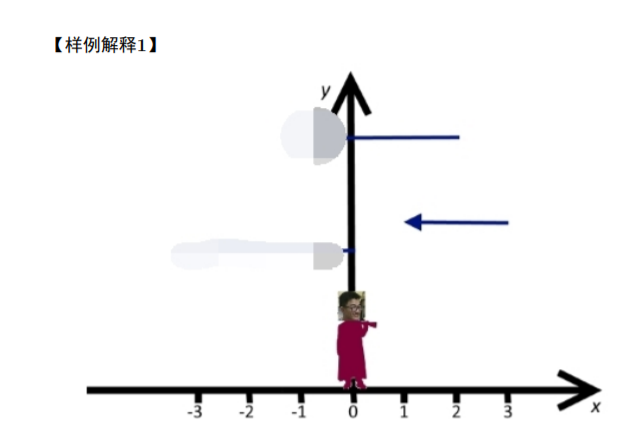


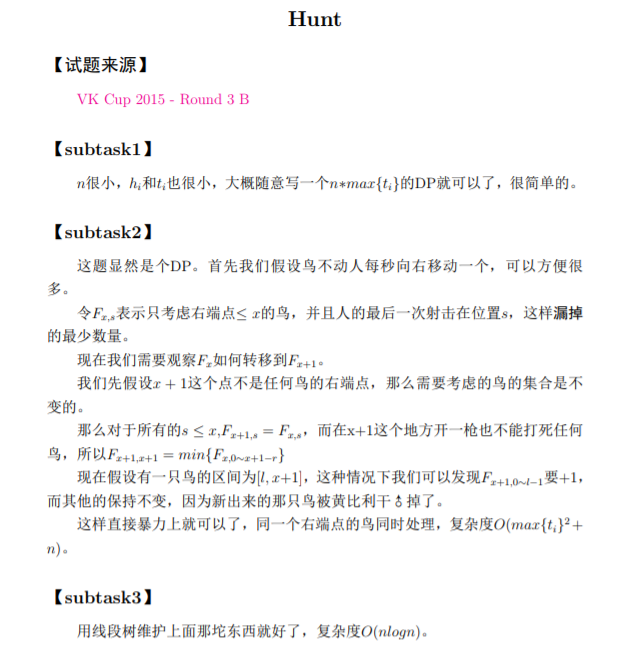
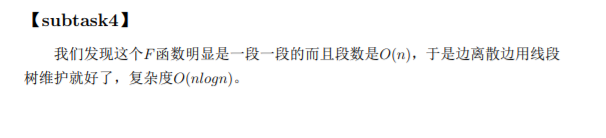


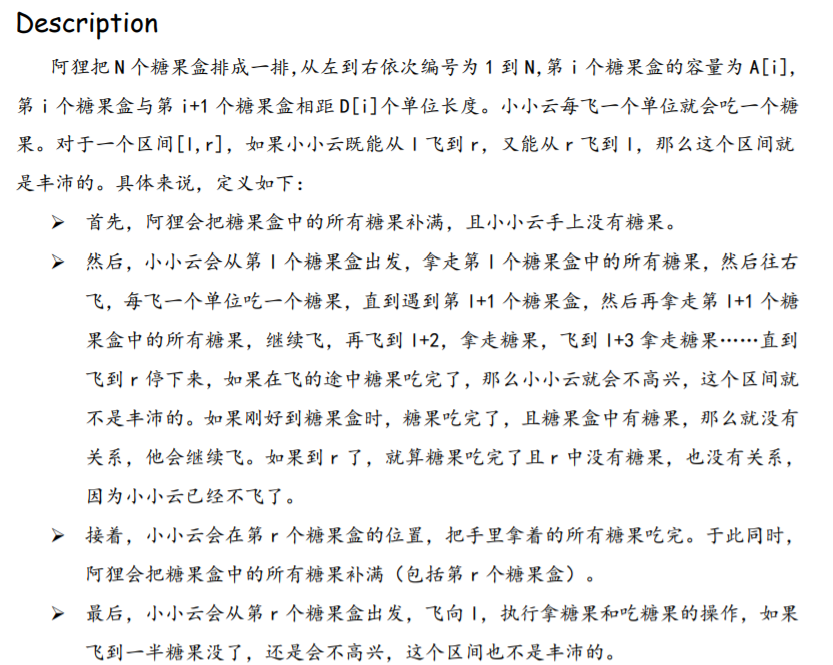


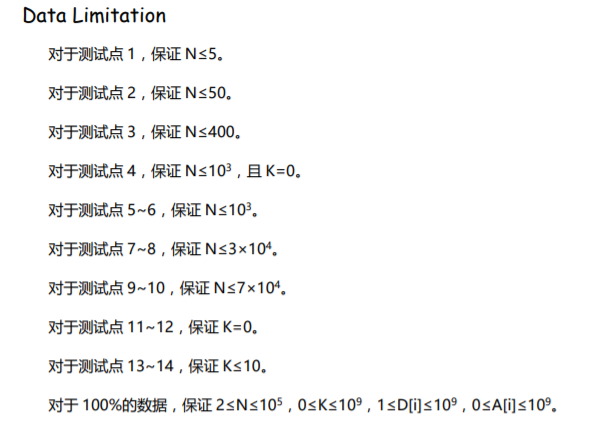


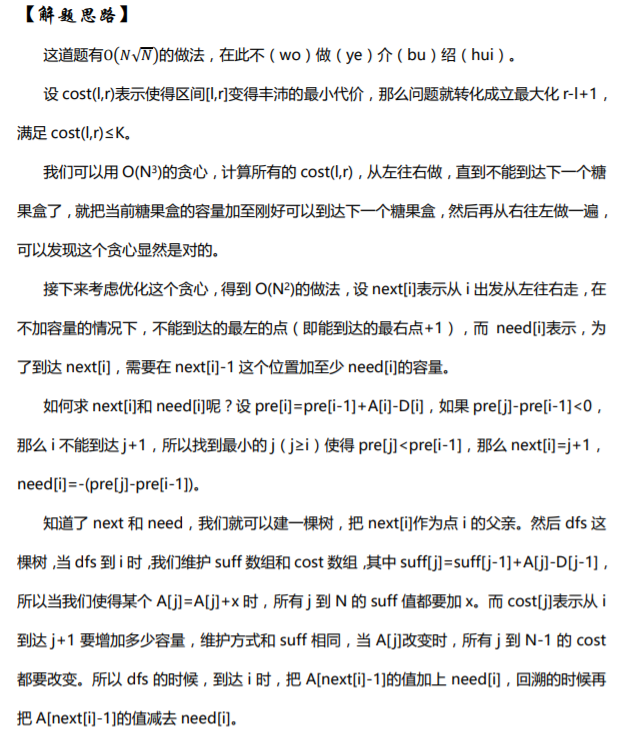


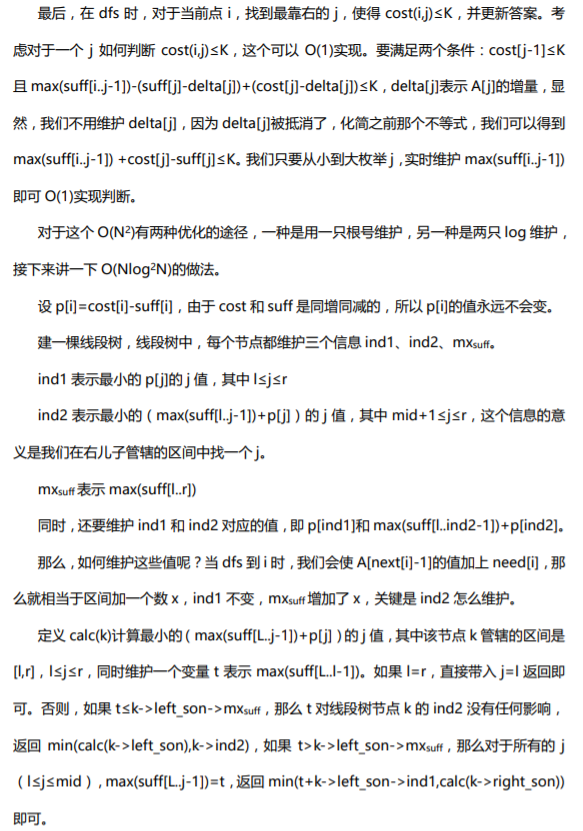


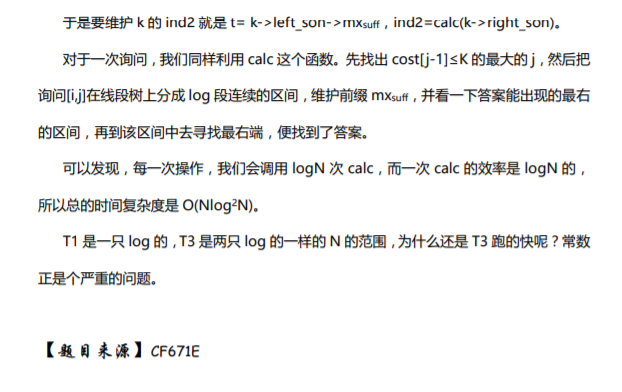










63.