题目难度分布:

简单:

A(起床别睡了)

K(消失的题号)

简单—中等:

C(奇怪的中位数$_{构造}$)

F(贝利亚大王爱吃奶龙 $-$ 简单版$_{思维}$)

I(贝利亚终于做了一次好事$_{模拟}$)

$M(InteractiveGame : Guessnumber_{二分交互})$

中等:

B(奇偶变换$_{线段树}$)

D(匪石的骰子$_{动态规划}$)

E(谁说羽羽不会玩魔方$_{模拟}$)

G(贝利亚大王爱吃奶龙 $-$ 困难版$_{优先队列 or 多重 set}$)

H(贝利亚又在偷偷干坏事了$_{数学等差数列求和}$)

中等—困难:

L(你是否还在寻找签到题目$_{前缀和单调栈}$)

J(互不相同的游戏$_{博弈}$)

困难:

N(树上红绿果$_{组合数学,树上计数}$)

A(起床别睡了)

输入X 输出x 即可

K(消失的题号)

输出题号'K'即可

C (奇怪的中位数$_{构造}$)

题意:

要求你构造一个序列满足以下条件(N≤ 200,K$\leq$200)

1.序列中的数字范围在0-99999999之间.

2.序列的中位数为K.

3.序列的平均值不为K.

4.序列不能递增增或者递减.

题解:

首先我们只考虑前三个条件,在一个升序的序列中满足中位数为K,在数组的前半部分全部放K,在数组的后半部分全部放最大值99999999,这样就满足前三个条件了,对于最后一个条件你可以分K为奇数偶数的情况,当N 为奇数的时候你可以这样构造{K,99999999,K,99999999,K,......},当N为偶数的时候同理但是你要将数组的最后一个位置的数字赋值为K.时间复杂度$O(N)$

F(贝利亚大王爱吃奶龙-简单版思维)

题意:

你有一个数组初始数组中有N 个元素你要操作 M 次,对于每次操作你需要将数组中的中位数加上K然后放入数组中,在此版本中$K\ =0$.操作M次次后数组的元素个数为$N+M$,操作完以后按照升序输出数组中的所有元素(保证N$\leq 10^5$,M$\leq 10^5$,$\sum_{i=1}^{t}(N+M)\leq 2\times 10^5$).

题解:

对于一个具有三个元素的数组$\{3,4,5\}$,你会发现操作一次后数组变为$\{3,4,4,5\}$,继续操作一次数组变为$\{3,4,4,4,5\}$,所以你会发现无论数组操作多少次数组的中位数不会发生改变,数组中增加的元素永远是初始题目中给的数组的中位数所以你可以初始维护一个长度为N+M的数组,前面N个元素是初始数组中的元素,后面M 个元素是初始数组的中位数,最后将数组排序输出即可.时间复杂度$O(N\times\log N)$

I(贝利亚终于做了一次好事思维模拟)

题意:

有一个长为N的数组,你每次用一次操作选择数组中两个相邻的元素减一,问最少能用多少次操作使得数组中的所有元素全部相等数组中不能出现负数,无法完成则输出-1.

题解:

遍历数组的每个位置判断当前位置和当前下一个位置的大小关系，(1).如果当前位置的数字小于当前位置下一个位置的数字我们要调整数组肯定不能在当前位置操作,因为在当前位置减去一个数字它下一个位置也会减去同样的数字他们的差值不会改变，所以只能在当前位置的下一个位置实行减操作，减去他们的相对差值，此时还要判断当前位置的下一个的位置是否为最后一个位置，如果是的话下一个位置无法和下下个位置实行减一操作，其次判断下下个位置减了之后是否为负数，为负数则直接将输出-1，(2).如果当前位置的数字比后一个位置的数字大对当前位置下一个位置实行减操作只会继续增大数字间的差距，所以只能对当前位置以及当前位置的前一个位置两两之间使用减少操作，将他前面的所有数字减为和他后面位位置的数一样大。

例如:

对于第二个样例:

$4, 6, 4, 4, 6, 4$

第一次遍历到下标1位置发现后面的数字比当前位置的数字大2所以对于下一个位置执行减2操作下一次数组变为

$4, 4, 2, 4, 6, 4$

第二次遍历到2位置发现当前位置比后一个位置大2所以将当前位置前面所有的数字两两之间减2操作数组变为:

$2, 2, 2, 4, 6, 4$

第三次遍历到三位置同理发现后面数字大于当前数字数组变为:

$2, 2, 2, 2, 4, 4$

第四次遍历到4位置对5,6位置执行减2操作最终数组数字全等:

$2，2，2，2，2，2$

同时注意这种情况

$8, 8, 8, 4$

比如当遍历到数组第三个位置发现当前位置比下一个位置大你要将前面所有数字实行减操作但是你发现前面数字有奇数个你只能将2，3位置同时减去4但是1位置依然不能减为相同所以非法，所以这种情况判断当前位置是否为偶数位置即可

这种就合法

$8, 8, 8, 8, 4$

当前在4位置可以将$3, 4$位置同时减4,$1, 2$位置同时减4最终数组全等.

另一种较复杂做法

假设我们已经将数组中的所有的数字变为相同怎么把它还原成为原始数组从而求解出所有的操作是一种什么样子的形式:

例如:假设数组长度为5且该数组是 :$5, 6, 5, 6, 5$

我们不知道它最终会减成什么形状假设最终见数组中的所有数字全相等且都为减为了X即:x, x, x, x, x

设另外两个数组分别为(操作序列)

$x_1, x_1, x_3, x_3, 0$

$0, x_2, x_2, x_4, x_4$

则有$x_1 + 0 + x$**=5,**$x_1 + x_2 + x$**=6,相邻位置做差可得**$x_2 = 6 - 5 = 1$**;**

则数组中相邻位置两两做差可以得出两数组分别为:

$x_1, x_1, x_1 - 1, x_1 - 1, 0$

$0, 1, 1, 2, 2$

然后根据最后一个位置$2 + 0 + x = 5$**,解得:**$x = 2$

对于第一个位置有$x + x_1 + 0 = 5$**,知道了x=2,解得:**x_1**=3**

最后解出了三个操作序列(分别为最终全等的序列两个在相邻位置加相等数字的操作序列，在全等序列的基础上执行加操作能变为题目的给定序列):

$2, 2, 2, 2, 2$

$3, 3, 2, 2, 0$

$0, 1, 1, 2, 2$

最后根据解出来的三个操作序列判是否合法即可判断最终形成的数组是否大于0,两个操作序列中的每一个位置是或否合法即大于零，计算答案就对两个操作序列求和即可,当N**为偶数的时候同理，时间复杂度**$O(N)$

B(奇偶变换$_{\text{线段树}}$)

题意:

在长度为N的数组中有Q次操作,每次操作会将一个区间的所有奇数或者所有偶数同时加一，每次操作完你需要对于数组求和设第i次的求和结果为sum_i ,最终你需要输出
$(1 \times sum_1) \otimes (2 \times sum_2) \otimes (3 \times sum_3) \ldots \ldots (Q \times sum_Q)$

$(N \leq 5 \times 10^5, Q \leq 5 \times 10^5)$

题解:

对于每次操作完以后不需要知道数组中每个元素具体的值，只需要知道上一次数组的求和和对应区间奇数或者偶数的个数，这次求和结果就是在上一次对于数组求和的基础上加上这次操作区间的奇数或者偶数的数量,例如我这次要对一个区间所有的偶数加一，那我这次增加的值就是该区间偶数的个数,用上一次的数组求和结果加上这次操作区间的偶数个数就是这次数组的和,统计完以后我需要将区间所有偶数加一，偶数加一会变奇数也就等价于将区间所有数变为奇数,那就等价于我要维护两种操作一种是查询区间区间奇数或偶数的数量，第二种就是区间赋值操作将区间的数全部赋值为奇数或者偶数，对于上述操作我们可以用带有懒标记线段树维护，我们只用在线段树中维护区间奇数的个数即可(区间偶数的个数可以用区间长度减去区间奇数数量)，懒标记实现区间全部置1或者全部置0操作。时间复杂度$O(Q \times \log N)$

D(匪石的骰子$_{\text{动态规划}}$)

题意:

一个骰子有6个面每个面都有一个值,问投掷N次投掷出和为K的方案数是多少a?（$N \leq 20, K \leq 200$)

题解:

首先观察发现N,K很小由此我们可以联想到动态规划,设$dp_{[i][j]}$，表示投了i次骰子总和为j的方案数。其状态转移方程为对于每一个i,j以及$dp[i][j]$我们可以通过前一次的状态$dp[i-1][j-a_i]$,其中a_i是骰子的一个面值，具体来说状态转移方程就是具体来说，
$dp[i][j] = dp[i-1][j-a1] + dp[i-1][j-a2] + \ldots + dp[i-1][j-a6]$。注意判越界的情况例如$j-a_i$小于0则不能转移否则会出现数组越界的情况。枚举i从1到N,j从0到200,k从1到6。时间复杂度$O(6 \times N \times K)$。

另一种做法：

如果骰子的面 M 很多，$M \le 10^5$ 还有一种基于形式幂级数的多项式做法。

$f=(1+x^{a_1}+x^{a_2}+\cdots+x^{a_m})$

答案为：$[x^K]f^N$

意为多项式 f 自乘 N 次 之后，x^K 的 系数的值

M(Interactive Game:Guess Number 二分交互)

题意:

在1到N的范围有一个特殊点X,你并不能直接得到，你每次可以向系统提问，以$?\ a\ b$形式系统会返回$(x-a)\times(x-b)$的符号，在不超过四十次询问确定特殊点x。

题解：

首先我们先观察询问的点和特殊点的之间的位置关系会返回什么样的符号，当特殊点在询问的点的中间时，即满足$a \le X \le b$的时候系统会返回-1,否则返回0或1,第一次我们会询问整个端点以确定特殊点是否在两端点，返回0则询问两次可得到正确答案，否则每次将询问区间范围减半，用系统返回的数字是否为-1判断特殊点是否在当前询问的区间范围内，是则继续将区间减半，是0则再询问两次端点值看哪个端点为0，否则将区间询问移动到右半部分继续减半询问，每次将询问的区间缩小一半，询问的次数约为$\log_2 1e9$约为32次左右。

E(谁说羽羽不会玩魔方$_{\text{模拟}}$)$直接模拟即可

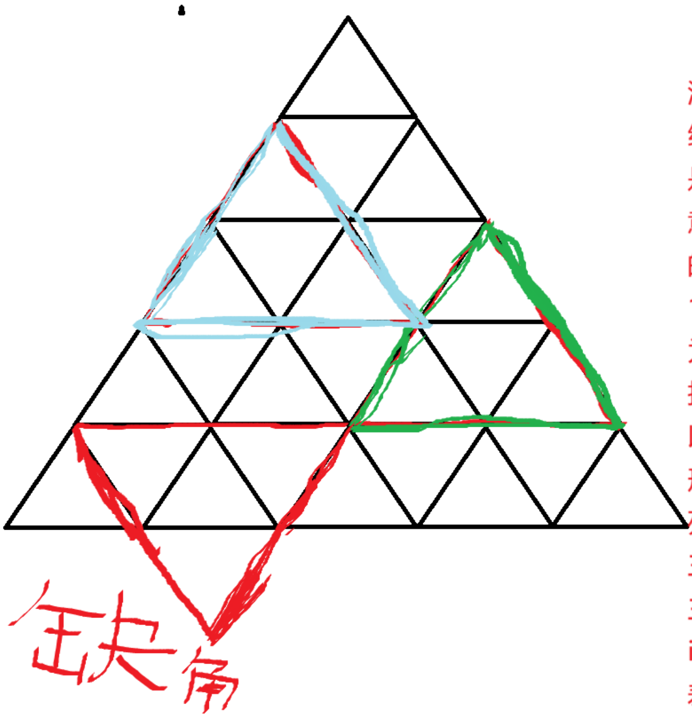
G(贝利亚大王爱吃奶龙-困难版$优先队列or多重set$)

在简单版本中增加了一个K可以为任意值的限制，相对于简单版本此版本不再是不断地添加一个中位数了，而是添加中位数加上K的值

题解：

数组中的中位数会不断发生变化,所以你需要维护这样一种数据结构:1.可以往一个容器中不断加入数据,2.可以快速查询容器的中位数。可以开两个多重集合multiset一个集合维护前半部分元素，另一个集合维护后半部分，然后动态调整每次加入元素，查询中位数的时间复杂度都在$\log$的时间范围内，或者也可以用两个优先队列维护，对上述过程不理解可以自行搜索双multiset动态维护中位数或者双优先队列动态维护中位数。

H(贝利亚又在偷偷干坏事了$数学等差数列求和$)



注意到图中任意选择一个底边都能组成唯一一种向上的正三角形，但是对于向下方向的三角形不一定任意选择底边都能组成例如图中缺角的三角形，所以对于第i行,选择长为1作为底边有i种选择方法，选择长为2作为底边有i-1种选择方法.....选择长为i作底边有一种选择方法，所以对于第i行组成的向上方向的三角形个数就是1+2+3+.....+i,用等差数列求和可快速计算，计算向下方向三角形数量同理，向下方向组成的三角形最大边长就是mn=min(i,n-i),i代表所属行，也是对1到mn求等差数列和

首先先计算图中不删除任意方格线的正三角形的数量，将其拆解成两种方向的三角形数量之和（方向向上和方向向下），首先看图中向上的三角形的数量发现你在图中任选一段横线都能且只能组成一个向上的正三角形所以对于第i行来说从长为i的横线选出来长为1的横线有i种选择方式选出长为2的有$i-1$种选择方式,选出长为i的有1种选择方式所以对于第i行来说能选出$i+(i-1)+(i-2)....+1$个向上的三角形,用等差数列求和公式可以$O(1)$计算出当前行选择向上的三角形方案数，同时计算当前行向下的三角形也是同理，但是注意向下的三角形并不是任意长的的横线都能组成，要看当前横线的长度是否大于了当前位置向下的高度例如图中倒数第二行拥有横线长度大于二的横线但是并不能组成向下的边长为二的正三角形，倒数第三行也是同理，但是倒数第三行可以组成底边长为2的正三角形，所以看当前行能最大组成底边长为多少的向下的正三角形就要对两个值取最小值一个是当前行的宽度i(假设是第i行)另一个是当前行向下的最大高度$(n-i)$,对这两个值取个最小值即为min,则当前行能组成向下的三角形的数量就是从i到$i-min$ 的等差数列求和，最后看删除某个位置的横线对向上方向和向下方向的三角形的减少量,假如删除a行b列的横线，减少向上方向的三角形数量就是该横线和他左右能连成完整直线的方案数，这就是$(b)*(a-b+1)$,然后看删除操作减少的向下的三角形的数量就是计算删除当前横线会减少多少以1为底边，以2为底边，以3为底边......的向下三角形的个数，这一部分结合被删除横线的位置讨论即可，综上所述时间复杂度为$O(N)$

原题链接:[Triangles - 题目 - QOJ.ac]

L(你是否还在寻找签到题目 前缀和单调栈)

前排CF2100大佬30分钟开出来导致其他队伍想跟榜导致前期榜歪。

题意:

有两个长度为N的f和g数组,求每一个区间的和，两端点分别为$[a\ b]$的区间的和为f数组的区间最大值乘以g数组的区间和

题解:

由于题目没有规定f数组中的而数字两两不同所以不能去找比当前位置的值大的左边和右边的位置，否则需要对重复出现的值去做容斥，所以我们只考虑当前位置往右边以当前位置为左端点的所有区间中以当前位置的值为最大值和不以当前位置的值为最大值的所有区间，第一种以当前位置为最大值的区间的贡献例如：

$x, x, x, 7, x, x, x, 9, 10, x, x$

$x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}$

例如当前我们从后往前遍历到f数组的第4个位置此位置上的数字为7假设区间[4,4],[4,5],[4,6],[4,7]都是以7为最大值，下标为8位置以后的区间都不以7为最大值，先看以7为最大值的区间的贡献是什么：
$7 \times (sum_{4-4} + sum_{4-5} + sum_{4-6} + sum_{4-7})$,

如何快速计算所有区间的求和，将sum式子拆开可得:

$7 \times (x_4 + x_4 + x_5 + x_4 + x_5 + x_6 + x_4 + x_5 + x_6 + x_7)$,

进一步转化为:

$7 \times (4 \times x_4 + 3 \times x_5 + 2 \times x_6 + 1 \times x_7)$,

对于这部分可以提前维护一个$i \times f[i]$的前缀和即可快速计算，也可以用线段树和树状数组维护，那么不以7为最大值的所有区间怎么计计算，由于是从后往前遍历，在计算当前位置必定已经计算后面的值的贡献不以7为最大值的区间一部分的贡献可以由第一个比7大的位置继承过来,但你会发现会少算一部分的区间和乘以区间最大值即上述样例的[4,7]的区间和乘以后面不以7为最大值的贡献，这一部分很好计算，在维护单调栈的时候就维护一下后面单调栈的区间最大值的和最后用维护的单调栈的和乘以计算的区间和即可。时间复杂度$O(N)$。

J(互不相同的游戏博弈)。

原题链接:https://atcoder.jp/contests/arc137/tasks/arc137_c

N(树上红绿果组合数学,树上计数)

原题链接:https://atcoder.jp/contests/arc108/tasks/arc108_f