个人刷题

fks

CF1687C

题解:考虑转化题意，令c[i]=b[i]-a[i],再对c作前缀和，那么就转化为了，每次对于[l,r]如果c[l-1]=c[r],那么把l-r这段区间全部覆盖为c[l-1].要求我们最后能把所有c都变成0.我们倒着考虑，来看操作是否有用（因为一开始我对于区间两两交的影响很头痛）。考虑最后反正要都变成0.那么必然c[l]和c[r]也必须要是0.否则做了和没做一个样。那么我们只对是0的考虑。我们把操作存在两个端点的vector里。暴力判断和更新就好了。用set维护非0位置弹出set用于均摊，只染色非0位置）。

CF1687D

题解：一开始想的，把段都暴力弄出来，后面扫描线做。但发现其实不需要，考虑可爱的区间是[k^2,k^2+k],那么我们就发现，当我们固定了a[1]的段，也就是a[1]的偏移量的范围确定，后面的段的偏移量也能唯一确定（因为一个段最多从可爱到不可爱，或者从不可爱到可爱，不可能跳变两次（凸性））。那么我们可以发现，一个在不可爱段的最小值，会对下界有影响。一个在可爱段的最大值，会对上界有影响。我们预处理出前驱后继，直接做。每次跳的次数是V/i那么就是调和级数

CF1687E

题解：考虑给出的形式，实际上是每个因子的min和min_rk2相加，我们考虑他的选择方式，实际上暗示着我们考虑min_Max容斥，比较容易的可以得到式子（用广义minmax可以得到），然而我们发现复杂度是2^n*n无法通过。似乎没法优化？我们换个方向，想想能否减小n来简化问题。也就是说挑选出一些代表性的数，来与我们整个数列等价。我们考虑一个定理：一个数最多的因子个数是w(n)在1e6内，这个函数是7。那也就是，我们可以每次钦定每个因子次小/最小给他选上。

CF1717E

vp 题解：比较关键的一点就是想到枚举gcd不枚举比较难做）。利用辗转相减，我们可以发现a=x*t,b=y*t,gcd(n-(x+y)*t,t)就变为了gcd(n,t)但我们考虑要求出gcdx,y==1并且x+y=c的对数，考虑y=c-x,gcd(x,c-x)==1 ⇒gcd(x,c)=1,也就是欧拉函数

CF1717F

(vp) 题解：+1，-1比较难处理，考虑先全部-1，做单选题，每次选一个+2，那么就变成了考虑+2分配给谁的问题。我们做个delta把目标值和当前值作差，考虑每个人要被加几次+2，也就是delta/2 那么可以判断掉一些不合法的delta负、delta不能被2整除。比较显然可以得到网络流模型，但我们考虑到有些是无限制，如果按照“最多”的约束连边，则能达到最大流，但没法满足恰好。考虑上下界网络流建图即可。

ljz

VP的签到题就不写了。

CF1730D

题意：每次可以把a的前k位和b的后k位交换，问a是否可以变成b□
容易发现，对于ai和bn-i+1这两个元素的对应关系不管怎么操作都是不会变的。所以有解无解的判断方法就是相同的pair□无序）有偶数个，或者有一个奇数且n是奇数。

CF1730E

题意：计算有多少个区间满足最大值可以整除最小值。
先用单调栈求出以每个元素为最大值最小值的最长区间。我们对每个ai都求出以ai为最大值的满足条件的区间，枚举ai的每个约数□ai在一百万以内因此可以AlogA预处理），对于一个约数d□可以找到离ai最近的d□左右各一个），我们只需要计算包含ai和d的满足条件的区间，对于左右的d注意去重。

CF1730F

题意：给定一个排列p和k□找到一个排列q□满足任意i<j□都有pqi<=pqj+k□使得q的逆序对数量最少□(n<=5000,k<=8)
我们一个一个考虑q□对于q1□pq1的值只可能是[1,k+1]中的的一个，如果pq1取了1那pq2的值只可能是[1,k+1]中的一个，如果pq1没有取1那么pq2的值还是只能在[1,k+1]中取，后面以此类推，也就是说每放置一个q□我们需要考虑的值只有最小的没被选过的数到这个数+k的范围内，这样就可以状压DP了。

CF1733E

题意□n*m的方格图上每个格子有一个箭头，初始时(0,0)上有一个小球，以后每一个时刻所有小球都会按照箭头的方向走一个格子，然后(0,0)上会新出现一个小球，所有上一个时刻有球的格子都会转换箭头方向（只有向右和向下两种状态）。问第t时刻给定的格子是否有球□(t<=1e18□x0y0<=120)
我们把所有小球统一考虑，也就是说一共会生成t个小球，小球走到(x0,y0)需要x0+y0-1个时刻，我们考虑有可能经过(x0,y0)的小球一共t-x0-y0+1个，如果经过当前格子的小球有n个，那么其中会有\left\lfloorn/\left.2\right\rfloor\right.个经过下方的格子，剩下的会经过右边的格子。我们只需要知道t时间经过(x0,y0)的小球数量和t-1时间经过(x0,y0)的小球数量是否相等就可以判断是否在t时刻(x0,y0)上有球。

CF1720D

CF1717E

CF1717F

CF1716E

CF1716F

CF1715F

CF1715E

CF1713E

CF1712E

CF1709E

CF1706D

CF1697E

CF1696F

CF1696G

ARC149D

ARC149E

ARC149F

ABC271G

ARC148E

个人学习

ddp

新学了ddp□目前只做了P4719□还未学全局平衡二叉树□ddp实际上就是维护了一段重链的转移矩阵，难点在于矩阵的设计和dp状态的设计，可能有时候dp状态会为了转移，把一些权值给并入，比较抽象。需要维护轻儿子的dp值和当前的dp□每次更新一条重链的权值，再把更新重链顶端的父亲的权值。由于每个点维护了轻儿子的dp□且转移矩阵只和轻儿子的dp有关，所以每次修改的矩阵数只有log