排序、分治、离散化

黎伟诺

7.18.2024

Beautiful Array

CF1986E

You are given an array of integers $a_1, a_2, \ldots, a_n$ and an integer k. You need to make it beautiful with the least amount of operations.

Before applying operations, you can shuffle the array elements as you like. For one operation, you can do the following:

- Choose an index $1 \leq i \leq n$,
- Make $a_i = a_i + k$.

The array $b_1, b_2, \ldots, b_n$ is beautiful if $b_i = b_{n-i+1}$ for all $1 \leq i \leq n$.

Find the minimum number of operations needed to make the array beautiful, or report that it is impossible.

Beautiful Array

CF1986E

注意到我们实际上想对元素进行配对（在 n 为奇数的情况下，只有一个元素没有配对）。如果数字 $x \leq y$ 属于同一对，那么需要 x, y 模 k 的余数相同，并且需要 $\frac{y}{k} - \frac{x}{k}$ 的代价

考虑数组 a 中的所有除以 k 余数相同的数，记它们除以 k 的结果为 $b_1 \leq b_2 \leq \cdots \leq b_m$

有两种可能的情况：

m 是偶数，我们需要把所有的数字配对。答案最优为偶数位置减奇数位置

m 是奇数，需要移除一个奇数下标，然后相邻依次配对，需要用前缀和、后缀和快速计算

Elevator

2023ICPC 南京 L

有一个载重为 k 的电梯，有若干重量为 $w = 1$ 或 2 的货物需要运送到某一楼层，一次运送的代价为本次运送楼层最高的层数（总重量不能超过电梯载重）。问运送完所有货物最小代价。

货物以 c_i, w_i, f_i 的形式给出，分别代表货物的数量，重量，需要送到的楼层。

k 保证为偶数。

贪心策略：
我们可以使用如下贪心策略：将所有包裹（无论重量是多少）按目标楼层从高到低排序，之后每趟电梯按该顺序枚举所有还未被运送的包裹并加入电梯，直到电梯装满或所有包裹都被枚举。
枚举过程中可能会出现电梯容量只剩 1，但是下一个包裹的重量却是 2 的情况。此时要继续枚举，直到出现重量为 1 的包裹。

另一种理解角度：
如果放了一个 1，那么肯定会再放一个 1，不然它自己相当于 2。
这两个 1 合并相当于重量为 2，运送的楼层是 $\max\{f_i, f_j\}$
我们怎样贪心的合并？应该按楼层从大到小排序，然后两两相邻合并
最后还剩一个 1 的话直接改成重量为 2
然后所有都重量为 2，只需要贪心地从大到小放就行了

火柴排队

P1966

涵涵有两盒火柴，每盒装有 n 根火柴，每根火柴都有一个高度。 现在将每盒中的火柴各自排成一列， 同一列火柴的高度互不相同， 两列火柴之间的距离定义为：$\sum(a_i - b_i)^2$。

其中 a_i 表示第一列火柴中第 i 个火柴的高度，b_i 表示第二列火柴中第 i 个火柴的高度。

每列火柴中相邻两根火柴的位置都可以交换，请你通过交换使得两列火柴之间的距离最小。请问得到这个最小的距离，最少需要交换多少次？如果这个数字太大，请输出这个最小交换次数对 $10^8 - 3$ 取模的结果。

火柴排队

P1966

需要让调整成 a_i 的排名与 b_i 的排名相等

因为 $\sum(a_i - b_i)^2 = \sum(a_i^2 + b_i^2) - 2\sum a_i b_i$

排序不等式：顺序乘积和大于等于乱序乘积和大于等于逆序乘积和

调整的最少步数："逆序对"个数，因为每交换一次逆序对数量最多减少 1

将 $b[i]$ 映射成 1 到 n，然后 $a[i]$ 也进行相同的映射

相当于求转化后的 $a[i]$ 的逆序对个数（树状数组或者归并排序）

Barn Allocation G

P1937

农夫约翰最近开了一个新的牲口棚屋，并且现在接受来自奶牛的分配畜栏请求因为其中的一些畜栏有更好风景。

畜栏包括N个畜栏(1 ≤ N ≤ 100,000)，方便起见，我们把它们编号为1..N，畜栏i能容纳Ci只牛(1 ≤ Ci ≤ 100,000)，第i只牛需要连续编号畜栏（从Ai到Bi）来漫步其中，

(1 ≤ Ai ≤ N; Ai ≤ Bi ≤ N)，换言之，这只牛想要在编号范围为Ai..Bi的畜栏漫步（所有它想要畜栏必须实施为它空出位置来供它散步）

给出M个畜栏分配请求（1 ≤ M ≤ 100,000），回答最多能满足多少只牛的要求（不增加另外畜栏）

考虑以下例子：

```
畜栏号:     1   2   3   4   5
          +---+---+---+---+---+
容纳空间:   | 1 | 3 | 2 | 1 | 3 |
          +---+---+---+---+---+
Cow 1      XXXXXXXXXX            (1, 3)
Cow 2          XXXXXXXXXXXXXXX   (2, 5)
Cow 3          XXXXXXX           (2, 3)
Cow 4                  XXXXXXX   (4, 5)
```

Barn Allocation G

P1937

将奶牛的路径转化为线段，以右端点为第一关键字，左端点为第二关键字，按第一关键字从小到大排序，若第一关键字相同则按第二关键字从大到小排序。
然后按序枚举线段，看看是否可以放（区间最小值 > 0）
如果可以，则区间内所有的数 -1，答案 $+1$，用线段树维护区间最小值。

Barn Allocation G

P1937

正确性证明：
右端点相同时，肯定优先取大的左端点
右端点不同的时候，考虑线段 [l1, r1] 和 [l2, r2] 矛盾，且 r1<r2。
此时，如果我们选择区间 [l2,r2]，并且假设这种方法比 [l1, r1] 更优，那么多出来的线段一定在区间 [l1,l2] 中。
由于我们是按照区间右端点排序的，所以在 [11,12] 中一定没有别的线段。反观区间 [r1,r2] 由于选择了第二个区间，所以 [r1, r2] 与之前相比是多减了 1 的，一定不比选择 [l1, r1] 优。

上升子序列

P3970

给定一个只包含整数的序列(序列元素的绝对值大小不超过10^9),你需要计算上升子序列的个数,满足如下条件的称之为一个上升子序列:

1. 是原序列的一个子序列
2. 长度至少为2
3. 所有元素都严格递增

如果两个上升子序列相同,那么只需要计算一次。例如:序列{1,2,3,3}有4个上升子序列,分别为{1,2}{1,3},{1,2,3},{2,3}

上升子序列

P3970

如果没有”两个上升子序列相同，那么只需要计算一次”这一个性质，我们用 dp[i] 表示以 i 结尾的上升子序列个数，那么就有

$dp[i] = \sum_{j=1}^{i-1} dp[j] \times [a[j] < a[i]]$

可以直接用树状数组来优化，于是就变成了 $O(n \log n)$，需要离散化

上升子序列

P3970

如何去重？

直接不考虑这个再次出现的数是肯定不对的，如果这个数和它之前出现的那个位置之间有一些比这个数小的的数，那么这些就就没有被计入答案，这样是有问题的

但是我们可以对每一个数维护一个 $lastans[i]$，表示 i 这个数上次被计入答案的时候 $\sum_{j=1}^{i-1} dp[j] \times [a[j] < a[i]]$ 是多少，还是用树状数组来查询前缀和之后

但计入答案的应该就是这次查询出来的答案减去 lastans，也就是表示新增的上升子序列的个数是多少

然后把 $dp[i]$ 加入树状数组以及更新 lastans

饥饿的狐狸

P4801

到你的宠物狐狸的晚餐时间啦！他的晚餐包含 N 块饼干，第 i 块饼干的温度是 T_i 摄氏度。同时，在晚餐中还包含了一大盘 W 摄氏度的水。

在喝了一口水之后，你的狐狸开始吃饭了。每当他吃一块饼干时，这块饼干的美味度为当前饼干与吃/喝的前一样食物（包括饼干和水）的温度的差的绝对值。它可以在任意时间喝水（保证水喝不完），或按任意顺序吃饼干。

最后狐狸获得的美味值为它吃下的每块饼干的美味度之和。请求出狐狸获得的最小和最大的美味值。

饥饿的狐狸

P4801

要得到最大美味值，狐狸应该先交错吃温度最大的和最小的饼干，其中可能需要喝水。所以给所有饼干按照温度排序，交替选择左右端点吃
最小值可以对饼干依次吃下求得
以此推出公式 $max(0, w - t[1]) + max(0, t[n] - w)$

求凸包

P2742

给定 n 个点，求最小的包围住这 n 个点的多边形

求凸包

P2742

先对点去重，找到 y 坐标最低的点，其他点对这个点进行极角排序
维护一个关于凸壳的栈，加入点时把栈中不满足凸性的点全部 pop 掉

扫描游戏

P8777

有一根围绕原点 O 顺时针旋转的棒 OA，初始时指向正上方（Y 轴正向）。平面中有若干物件，第 i 个物件的坐标为 (x_i, y_i)，价值为 z_i。当棒扫到某个物件时，棒的长度会瞬间增长 z_i，且物件瞬间消失（棒的顶端恰好碰到物件也视为扫到），如果此时增长完的棒又额外碰到了其他物件，也按上述方式消去（它和上述那个点视为同时消失）。

如果将物件按照消失的时间排序，则每个物件有一个排名，同时消失的物件排名相同，请输出每个物件的排名，如果物件永远不会消失则输出 -1。

扫描游戏

P8777

首先肯定要极角排序，按照棒旋转的方向对这些点进行排序。
排完以后，维护一颗线段树，这颗线段树维护的信息是点到原点的距离的平方的最小值。
要这颗线段树有什么用呢？
记棒当前的长度为 len，棒当前划到的点为 last。
那么棒想划到下一个点的话，就要在 [last+1,n] 找第一个到原点的距离的平方小于等于 len2。的点，如果不存在这样的点，那么就在 [1,last-1] 这些点里试着找，相当于在新的一圈里找。这个找法需要在线段树里二分以及修改。
如果还是不存在这样的点，那说明再也找不到能划到的点了，其它物品返回-1。

离散化

用 map 把需要离散化的数组记录一遍，然后按 map 的顺序依次编号成 $1, 2, \ldots,$ *id*

把原数组用 map 映射成新数组，新数组的范围不超过 $1, 2, \ldots, n$

Painting the Fence S

P2205

Farmer John 想出了一个给牛棚旁的长围墙涂色的好方法。（为了简单起见，我们把围墙看做一维的数轴，每一个单位长度代表一块栅栏）

他只是简单的把刷子蘸满颜料，系在他最喜欢的奶牛Bessie上，然后让Bessie来回地经过围墙，自己则在一旁喝一杯冰镇的凉水。（……-_-|||）

Bessie 经过的所有围墙都会被涂上一层颜料。Bessie从围墙上的位置0出发，并将会进行N次移动(1 <= N <= 100,000)。比如说，“10 L”的意思就是Bessie向左移动了10个单位。再比如说“15 R”的意思就是Bessie向右移动了15个单位。

给出一系列Bessie移动的清单。FJ 想知道有多少块栅栏涂上了至少K层涂料。注意：Bessie最多会移动到离原点1,000,000,000单位远的地方。

Painting the Fence S

P2205

把经过的点坐标算一遍，然后离散化
染一段区间相当于区间加一，可以转化为差分的单点加，最后求前缀和算出原数组
每两个相邻下标的贡献是 $x_{i+1} - x_i$

数轴

P6602

小 X 画了一条数轴，他将进行 n 次操作，每次操作他会先在数轴上的 x_i 位置上增添 a_i 个标记。

然后他需要选择二元组 (l, r)，满足 l, r 为整数，$0 \leq l \leq r \leq m$，且在数轴上的区间 $[l, r]$ 上的标记的个数**小于等于** k。

对于每次操作，你需要求出满足条件的二元组 (l, r) 中 $r - l$ 的最大值。

$$1 \leq n \leq 10^6, 1 \leq m \leq 10^9, 1 \leq k \leq 100$$

数轴

P6602

答案区间 (l, r) 一定在某两个标记点之间。也就是说，$l-1$ 应当是一个有标记点，$r+1$ 也应是一个有标记的点。
往数轴上增加标记不好做，考虑从数轴上拿走标记。
我们可以预处理出所有标记点都在数轴上时的答案，也就是输出答案的最后一行。发现每当我们确定一个左端点 l，则随着 l 的增大，r 是单调不降的。基于这一性质，考虑用双指针扫一遍，可以做到 $O(n)$。

数轴

P6602

接下来考虑依次删去每一个标记，即把这一标记的权值设为 0，同时统计对应的答案。

发现 k 很小，删除一个标记点时，至多只会影响这个点左侧 k 个和右侧 k 个点

删除一个标记，我们不仅需要删除它的权值，还删除这个标记的在数组存储中的下标，枚举端点的时候直接跳过它。使用一个链表来维护每一个端点的前驱和后继。在删除标记时，只需修改其前驱后继的关系即可。

总复杂度为 $O(n \times k)$

Outer space invaders

P4766

来自外太空的外星人（最终）入侵了地球。保卫自己，或者解体，被他们同化，或者成为食物。迄今为止，我们无法确定。

外星人遵循已知的攻击模式。有 N 个外星人进攻，第 i 个进攻的外星人会在时间 a_i 出现，距离你的距离为 d_i，它必须在时间 b_i 前被消灭，否则被消灭的会是你。

你的武器是一个区域冲击波器，可以设置任何给定的功率。如果被设置了功率 R，它会瞬间摧毁与你的距离在 R 以内的所有外星人（可以等于），同时它也会消耗 R 单位的燃料电池。

求摧毁所有外星人的最低成本（消耗多少燃料电池），同时保证自己的生命安全。

$1 \leq n \leq 300$

Outer space invaders

P4766

时间点显然可以离散化

$f(l, r)$ 表示左右端点都在 $[l, r]$ 以内的敌人被消灭的最小代价。

每次转移时找到距离最远的点 id，然后枚举是在哪个时间点 k 消灭的，把这个点能消灭的都消灭掉。

时间复杂度 $O(n^3)$。

转移方程：$f(l, r) = d_i d + \max_{a_i d \le k \le b_i d}\{f(l, k-1) + f(k+1, r)\}$

Promotion Counting P

P3605

奶牛们又一次试图创建一家创业公司，还是没有从过去的经验中吸取教训——牛是可怕的管理者！

为了方便，把奶牛从 $1 \sim n$ 编号，把公司组织成一棵树，1 号奶牛作为总裁（这棵树的根节点）。除了总裁以外的每头奶牛都有一个单独的上司（它在树上的 “双亲结点”）。

所有的第 i 头牛都有一个不同的能力指数 p_i，描述了她对其工作的擅长程度。如果奶牛 i 是奶牛 j 的祖先节点，那么我们我们把奶牛 j 叫做 i 的下属。

不幸地是，奶牛们发现经常发生一个上司比她的一些下属能力低的情况，在这种情况下，上司应当考虑晋升她的一些下属。你的任务是帮助奶牛弄清楚这是什么时候发生的。简而言之，对于公司的中的每一头奶牛 i，请计算其下属 j 的数量满足 $p_j > p_i$。

Promotion Counting P

P3605

p_i 是可以离散化的
考虑 dfs 序的话就是需要查询 $[l_i, r_i]$ 这个区间内有多少个大于 p_i 的
可以看成是二维数点，每个点是 (i, p_i)，有若干个询问查询一个矩形内有多少点

无尽的生命

P2448

逝者如斯夫，不舍昼夜!

叶良辰认为，他的寿命是无限长的，而且每天都会进步。

叶良辰的生命的第一天，他有 1 点能力值。第二天，有 2 点。第 n 天，就有 n 点。也就是 $S_i = i$。

但是调皮的小A使用时光机，告诉他第 x 天和第 y 天，就可以任意交换某两天的能力值。即 $S_x \leftrightarrow S_y$。

小A玩啊玩，终于玩腻了。

叶良辰：小A你给我等着，我有 100 种办法让你生不如死。除非能在 1 秒钟之内告知有多少对“异常对”。也就是说，最后的能力值序列，有多少对的两天 x, y，其中 $x < y$，但是能力值 $S_x > S_y$?

小A：我好怕怕啊。

于是找到了你。

$x_i, y_i \leq 2^31 - 1, k \leq 10^5$

无尽的生命

P2448

k 不大，最多只牵涉到了 $2k$ 个数。
每个连续区间可以看作一个整体，用这个区间最小的数作代表元，权值就是区间数的个数。
这样要做的就只有 $3k$ 个数。
树状数组求逆序对个数即可

扫地机器人

P10096

一个扫地机器人正在清洁一个二维坐标平面。扫地机器人是一个边长 $k \times k$ 的正方形，边与坐标轴平行。初始时，扫地机器人左下角位于 $(0, 0)$，右上角位于 (k, k)。

题目描述

给定一个由 n 个移动操作组成的序列，第 i 个移动操作由方向 d_i（`N` 表示向上，增加 y 坐标；`E` 表示向右，增加 x 坐标；`W` 表示向左，减小 x 坐标；`S` 表示向下，减小 y 坐标）和距离 a_i（机器人移动的距离）组成。根据给定的机器人移动操作，计算清扫的总面积（被机器人覆盖过的点就算被清扫过的点）。

输入格式

第一行包含两个整数，机器人的大小 k 和操作数量 n。

接下来的 n 行中，每行包含一个移动操作和对应的距离 a_i。移动操作用字母 d_i 表示（`N` 即向上，`E` 即向右，`W` 即向左，`S` 即向下），且距离 a_i 是一个整数。

扫地机器人

P10096

每段路经就是一个宽为 k 或者长为 k 的矩形，把所有的矩形算出来做矩形面积并即可（需要离散化和线段树，主要是代码量大）

窗口的星星

P1502

晚上，小卡从阳台望出去，“哇~~~~好多星星啊”，但他还没给其他房间设一个窗户。

天真的小卡总是希望能够在晚上能看到最多最亮的星星，但是窗子的大小是固定的，边也必须和地面平行。

这时小卡使用了超能力（透视术）知道了墙后面每个星星的位置和亮度，但是小卡发动超能力后就很疲劳，只好拜托你告诉他最多能够有总和多亮的星星能出现在窗口上。

输入格式

本题有多组数据，第一行为 T，表示有 T 组数据。

对于每组数据：

第一行 3 个整数 n, W, H 表示有 n 颗星星，窗口宽为 W，高为 H。

接下来 n 行，每行三个整数 x_i, y_i, l_i 表示星星的坐标在 (x_i, y_i)，亮度为 l_i。

输出格式

T 个整数，表示每组数据中窗口星星亮度总和的最大值。

窗口的星星

P1502

每个点 (x, y) 可以转换为向右上延伸的矩形 $(x \sim x+w-1, y \sim y+h-1)$
两个（或者若干个）矩形有重叠部分相当于我可以放一个矩形把这些全部盖住
相当于我若干个带权矩形，求在哪个坐标上权值的总和最大。
扫描线（区间加、区间减）然后求区间最大值

序列排序

P2127

小C有一个N个数的整数序列，这个序列的中的数两两不同。小C每次可以交换序列中的任意两个数，代价为这两个数之和。小C希望将整个序列升序排序，问小C需要的最小代价是多少?

输入格式

第一行，一个整数N。

第二行，N个整数，表示小C的序列。

输出格式

一行，一个整数，表示小C需要的最小代价。

序列排序

P2127

每个位置 i 需要放到目标位置 $p[i]$
这个排列是有若干个环的：$i \to p[i] \to p[p[i]] \cdots \to i$
每个环的策略有两种：
1、运用环内最小值进行交换，用环内最小值交换节点数量减一次，再加上除这个环中的最小值之外其余节点的权值之和，就是这个环运用环内最小值进行交换的代价。答案为 $(sz[i]-1)*minn[i]+sum[i]-minn[i]$
2、运用环外最小值进行交换，首先将环内最小值与环外最小值进行交换，再用环外最小值交换节点数量减一次，再加上除这个环中的最小值之外其余节点的权值之和，最后将环内最小值与环外最小值进行交换回来，就是这个环运用环外最小值进行交换的代价。答案为 $(sz[i]-1)*mina+sum[i]-minn[i]+2*(mina+minn[i])$。

Lifeguards S

P4188

FJ 为他的奶牛们建造了一个游泳池，FJ 认为这将有助于他们放松身心以及生产更多牛奶。

为了确保奶牛们的安全，FJ 雇佣了 N 头牛，作为泳池的救生员，每一个救生员在一天内都会有一定的事情，并且这些事情都会覆盖一天内的一段时间。为了简单起见，泳池从时间 $t = 0$ 时开门，直到时间 $t = 10^9$ 关门，所以每个事情都可以用两个整数来描述，给出奶牛救生员开始以及结束事情的时间。例如，一个救生员在时间 $t = 4$ 时开始事情并且在时间 $t = 7$ 时结束事情，那么这件事情就覆盖了 3 个单位时间。（注意：结束时间是“点”的时间）

不幸的是，FJ 多雇佣了一名的救生员，但他没有足够的资金来雇佣这些救生员。因此他必须解雇一名救生员，求可以覆盖剩余救生员的轮班时间的最大总量是多少？如果当时至少有一名救生员的事情已经开始，则这个时段被覆盖。

输入格式

输入的第一行包括一个整数 N $(1 \leq N \leq 100000)$。接下来 N 行中，每行告诉了我们一个救生员在 $0 \sim 10^9$ 范围内的开始以及结束时间。所有的结束时间都是不同的。不同的救生员的事情覆盖的时间可能会重叠。

Lifeguards S

P4188

实际上我们就是要处理出每头奶牛区间里仅由他自己覆盖的长度，也就是那些覆盖次数为一的位置上的长度和。
这样我们就有了一种线性查询的算法：先用差分算出每个时间点被覆盖的次数，然后对于那些为一的地方加上他们的长度，同时维护前缀和。
这样对于每头奶牛就可以 $o(1)$ 地求他管辖范围内的且仅由他管辖的长度，然后求最大值输出即可

分治

$solve(l, r) \leftarrow solve(l, mid)\&\&solve(mid + 1, r)$

重点放在如何求跨 mid 贡献的部分

平面最近点对

P7883

给定 n 个二维欧几里得平面上的点 $p_1, p_2, \ldots, p_n$，请输出距离最近的两个点的距离。

平面最近点对

P7883

按 x 轴坐标排序，递归 x_mid 左半边，递归 x_mid 右半边

$\delta = min(solve(l, mid), solve(mid + 1, r))$

只有 $[x_{mid} - \delta, x_{mid} + \delta]$ 这一长条内的点需要考虑

这些点按 y 轴坐标排序，每个点其实只用考虑它之后的不超过 8 个点

时间复杂度 $O(n \log^2 n)$ 或者 $O(n \log n)$（如果用基数排序）

最大公约数

P5502

给定一个长度为 N 的正整数序列 A_i 。

对于其任意一个连续的子序列 $A_l, A_{l+1}, ..., A_r$ ，我们定义其权值 $W(L, R)$ 为其长度与序列中所有元素的最大公约数的乘积，即 $W(L, R) = (R - L + 1) \times \gcd(A_l, ..., A_r)$。

JYY 希望找出权值最大的子序列。

最大公约数

P5502

集合 $\{gcd(a[l], a[l+1]...a[r]) \mid l \leq r\}$ 元素个数不超过 $logn$ 个
这是因为：小于 n 且能被 n 整除的最大的数字一定小于等于 $n/2$，也就说小于 n 的最大的 gcd 不超过 $n/2$。
那么固定 r 后，每次往左边扩大一格，扩大区间范围后的 gcd 要么等于原来的 gcd，要么就是小于等于原来的 $gcd/2$。也就说产生的新的 gcd 数值一定都小于等于原来都一半，那么也就最多产生 log 级别个新的 gcd 数值。
并且我们发现，左边的 $gcd(a[l]...a[r])$ 一定小于等于 $gcd(a[l+1]...a[r])$。
(固定 r 之后左边只有 log 段)

最大公约数

P5502

所以我们枚举 r，用一个队列存储这 log 个不同的 gcd 的左端点，每次直接枚举这些 gcd 更新，更新的时候加一些判断处理重复就好了，写起来很容易。

复杂度：$(nlog^2n)$

Pudding Monsters

codeforces 526F

给定一个 $n \times n$ 的棋盘，其中有 n 个棋子，每行每列恰好有一个棋子。
求有多少个 $k \times k$ 的子棋盘中恰好有 k 个棋子。
$n \leq 3 \times 10^5$。

Pudding Monsters

codeforces 526F

转换下模型，矩阵可以看成一个 N 的排列，求的是有多少连续子序列中的数是一个区间中连续的，也就是最大数减最小数等于长度减一。
那么我们就可以考虑分治解决，对于跨过分治点 m 的情况的，可以分两大类考虑，最值各在分治点一侧、同在分治点一侧。
第一种，可以直接枚举一侧长度，就能计算出另一侧长度，在判断是否合法。
第二种，假如 min 在左，max 在右，约束条件是 $max[mid+1,y] \geq max[x,mid]$ 以及 $min[mid+1,y] \geq min[x,mid]$，前者是 $r \geq left$，后者是 $y \leq right$。这两个边界均会随着 x 从 mid 往 1 走而右移。这个可以用单调队列的思想维护一个窗口
等式条件 $max[mid+1,y]-min[x,mid]=y-x$，那么可以变成 $max[mid+1,y]-y+n=min[x,mid]-x+n$。我们需要维护窗口内的 $max[mid+1,y]-y+n$ 的 cnt 状态（开个桶）。
时间复杂度 $O(n \log n)$。

序列

P10162

定义一个长度为 n 的序列 $\{p_n\}$ 的权值 $f(\{p_n\})$ 为 $\max\limits_{i=1}^{n}\{p_i - \max\{p_{i-1}, p_{i+1}\}\}$，特别的，定义 $p_0 = p_{n+1} = -\inf$。

求 $\sum\limits_{l=1}^{n}\sum\limits_{r=l+1}^{n} f(\{a_l, a_{l+1}, \ldots, a_r\})$。

答案对 2^{32} 取模。

序列

P10162

以下记 $v_i = a_i - \max\{a_{i-1}, a_{i+1}\}$，$l_i = a_i - a_{i+1}$，$r_i = a_i - a_{i-1}$。

考虑分治，初始区间为 $[1, n]$，每次取区间中点，求出所有跨过区间中点的子区间的贡献，然后对中点的两侧分别递归。

设当前区间为 $[l, r]$，区间中点为 m。由于我们考虑的区间一定包含 m 和 $m+1$，所以我们可以计算 $p_i = \max\{v_{m+1}, v_{m+2}, \cdots, v_{i-1}, r_i\}$ 和 $s_i = \max\{l_i, v_{i+1}, v_{i+2}, \cdots, v_m\}$

那么一个子区间 $[x, y]$ 的贡献就是 $\max\{s_x, p_y\}$。对所有子区间求和这个值的直接做法是排序后归并，但是这个做法再套上外面的分治会达到 $O(n \log^2 n)$ 的复杂度。

序列

P10162

为了去掉排序，我们可以考虑通过在分治过程中归并的方式维护出有序的 p 数组和 s 数组。

对于区间 $[l, r]$，维护出所有的 $p_i = \max\{v_l, v_{l+1}, \cdots, v_{i-1}, r_i\}$ 和 $s_i = \max\{l_i, v_{i+1}, v_{i+2}, \cdots, v_r\}$。在对 $[l, r]$ 的中点两侧分别递归之后，$[l, m]$ 一侧的所有 p_i 均不变，可以直接继承；

$[m+1, r]$ 一侧的所有 p_i 继承上来的过程中需要对 $v_l, v_{l+1}, \cdots, v_m$ 取 max，而一个有序序列对一个值取 max 之后仍然是有序的，所以我们可以分别得到有序的 $p_{l,\cdots,m}$ 和 $p_{m+1,\cdots,r}$。

再将它们归并即可得到当前区间的 p 数组排序后的结果。s 的维护是对称的。求答案的时候，只需要分别取出左侧的有序的 s 数组和右侧的有序的 p 数组进行归并即可。
这样对一个区间分治就只需要进行若干次归并，复杂度就是线性的。再套上分治就达到了 $O(n \log n)$ 的目标复杂度。

根号分治

这玩意感觉不是很应该称为“分治”，它只是大于 B 时采用某种算法，小于 B 时采用另外一种算法（可以称之为 BigSmall）
B 通常为 $\sqrt{n}$，这样两边复杂度能够平衡成 $n\sqrt{n}$。

哈希冲突

P3396

简要题意:
给你一个长度为 n 的序列 a_i,m 次操作。操作有两种:
A 操作: 给你 x 和 y，询问序列里所有下标模 x 等于 y 的元素之和。
B 操作: 给你 x 和 y，把序列里下标为 x 的元素的值修改成 y。
$1 \leq n, m \leq 150000$ $1 \leq a_i \leq 1000$。

哈希冲突

P3396

这道题设置一个阈值 $B = \sqrt{n}$，表示询问时模数的阈值。

1、模数大于阈值

这个直接暴力做就可以了，设询问的池为 y，每一次不断加上询问的模数即可，因为此时保证模数大于阈值 B，那么可以保证单次查询的复杂度在 $O(\frac{n}{B})$ 里面。

2、模数小于等于阈值

这个时候就需要维护一下这一块的答案了。

设 $ans_{x,y}$ 表示模数为 x 时，下标余数为 y 的 a，其中 $x \in [1, \lfloor \sqrt{n} \rfloor]$。

那么这个显然在一开始可以 $O(B * n)$ 预处理出来。

然后询问的时候就可以 $O(1)$ 询问了。

3、修改操作

这个时候我们需要枚举模数 $x \in [1, \lfloor \sqrt{n} \rfloor]$，修改 $ans_{x,y \bmod x}$ 的值，最后直接更新 a_y 即可。

因为 x 最大上限是 B，因此这一部分也可以在 $O(B)$ 的时间内完成。

$B = \sqrt{n}$ 时复杂度最小，为 $N\sqrt{N} + Q\sqrt{N}$

无向图三元环计数

P1989

给定 $n \leq 10^5$ 个点和 $m \leq 10^5$ 条无向边，求三元环个数

无向图三元环计数

P1989

一种麻烦的根号分治做法：
把度数大于 $\sqrt{2m}$ 的称为大点（个数不会超过 $\sqrt{2m}$），小于等于 $\sqrt{2m}$ 的称为小点。
假如三元环里有小点，我们考虑枚举小点 u，枚举 u 的两个邻居 v, w，哈希表查看 v, w 是否有连边。并且如果 u, v, w 中有 t 个小点，贡献为 $\frac{1}{t}$（因为会算重 t 次）。这部分复杂度是 $\sum_{deg_u<\sqrt{2m}} deg_u^2$，由于总和一定，平方和肯定是一些取最大，一些取 0 时达到最大值，所以复杂度为 $\frac{m}{\sqrt{2m}} \times \sqrt{2m}^2 = O(m\sqrt{m})$。
假如三元环里全是大点，由于大点个数不超过 $\sqrt{2m}$，我们直接暴力枚举三个大点 u, v, w，查看 u, v、v, w、w, u 是否有连边，然后加上 1 的贡献。这部分复杂度是 $O(m\sqrt{m})$

无向图三元环计数

P1989

模板做法：

我们考虑给所有的边一个方向。具体的，如果一条边两个端点的度数不一样，则由度数较小的点连向度数较大的点，否则由编号较小的点连向编号较大的点。

不难发现这样的图是有向无环的。注意到原图中的三元环一定与对应有向图中所有形如 $u \to v, u \to w, v \to w$ 的子图一一对应，我们只需要枚举 u 的出边，再枚举 v 的出边，然后检查 w 是不是 u 指向的点即可。我们可以在枚举 u 的出边时给其出点打上 u 的时间戳，这样在枚举 v 的出边时即可 $O(1)$ 去判断 w 是不是 u 的出点。

无向图三元环计数

P1989

模板做法：

下面证明这个算法的时间复杂度是 $O(m\sqrt{m})$。

考虑对于每一条边 $u \to v$，它对复杂度造成的贡献是 out_v，因此总复杂度即为 $\sum_{i=1}^{m} out_{v_i}$，其中 v_i 是第 i 条边指向的点，out_v 是点 v 的出度。

考虑分情况讨论。

1. 当 v 在原图（无向图）上的度数不大于 $\sqrt{m}$ 时，由于新图每个节点的出度不可能大于原图的度数，所以 $out_v = O(\sqrt{m})$。
2. 当 v 在原图上的度数大于 $\sqrt{m}$ 时，注意到它只能向原图中度数不小于它的点连边，又因为原图中所有的点的度数和为 $O(m)$，所以原图中度数大于 $\sqrt{m}$ 的点只有 $O(\sqrt{m})$ 个。因此 v 的出边只有 $O(\sqrt{m})$ 条，也即 $out_v = O(\sqrt{m})$。

因此所有节点的出度均为 $O(\sqrt{m})$，总复杂度 $\sum_{i=1}^{m} out_{v_i} = O(m\sqrt{m})$。

Yet Another Path Counting

ABC259Ex

有 N 行 N 列的网格图，只能向下或向右走，合法路径的开端和结尾的格子上数字一样找到合法路径条数，对 998244353 取模。

$1 \leq N \leq 400$

Yet Another Path Counting

ABC259Ex

如果枚举起点和终点，那么可以 $O(1)$ 算出一个方案数。但是这样时间复杂度是 $O(N^4)$ 的，显然过不去。我们考虑根号分治:
设数字 c 的格子数量为 k，如果 $k < N$，那么我们直接 k^2 枚举算贡献(组合数 $C_{x-z+y-w}^{x-z}$)。这一部分的时间复杂度是 $O(\frac{N^2}{k} \times k^2)$ 的。
如果 $k > n$，那么会发现最多有 n 种这样的数字。我们对于每一种这样的数字，如果能设计一种 $O(N^2)$ 的算法求解就行了。
我们考虑每一次对于整个网格图做一遍 DP。设 $dp_{i,j}$ 表示从一个数字为 c 的格子走到 (i,j) 的方案数。那么转移非常简单。如果 (i,j) 上的数字是 c，那么我们把 $dp_{i,j}$ 累加到答案中就好了。时间复杂度 $O(N^3)$。

谢谢