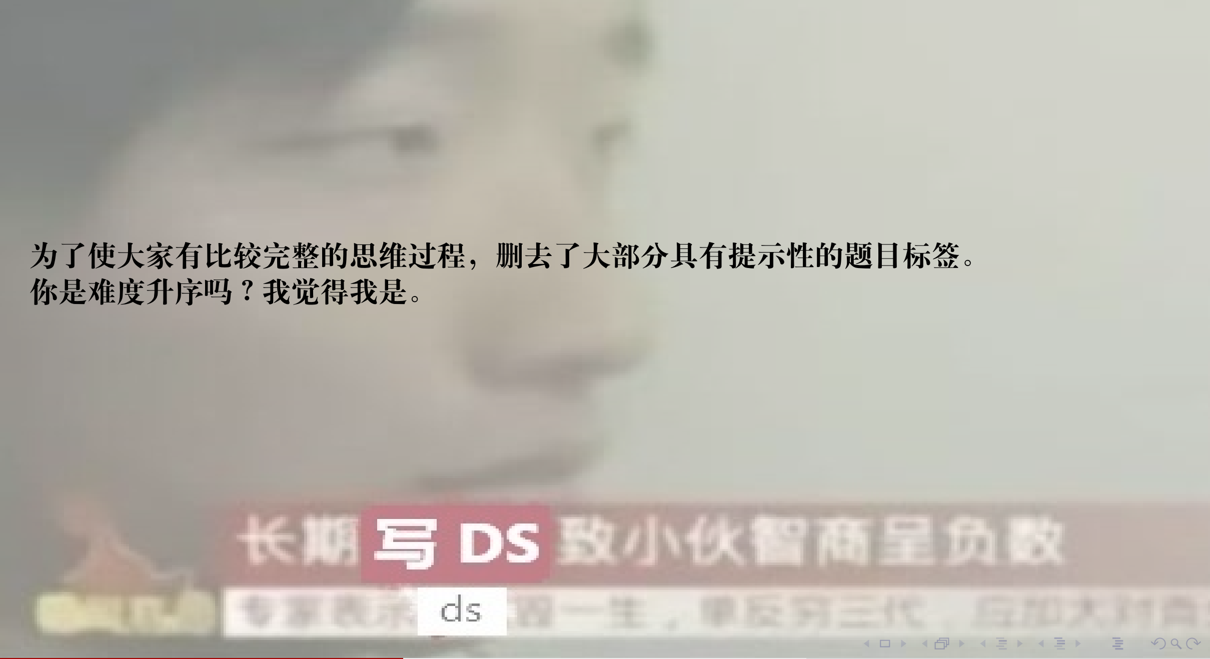
数据结构选讲

Asdfz,Yuuri

July 22, 2025

为了使大家有比较完整的思维过程，删去了大部分具有提示性的题目标签。
你是难度升序吗？我觉得我是。



存在一个长为 $n(n \le 10^5)$ 的 01 序列。
有 m 条形如 op, a, b 的约束，$op = 0/1$ 代表 $[a, b]$ 全为 0/至少存在一个 1。
q 次询问：只保留编号在 $[L, R]$ 内的约束，问是否存在合法序列。

考察经典应用：判断是否有包含区间/被区间完全包含。
称 $x = 0$ 为 0 类线段，$x = 1$ 为 1 类线段。
合法的充要条件即所有 1 类线段不被 0 类线段的并集覆盖。
首先发现容易求的是每个右端点的最左合法左端点（询问）。
考虑扫描线，每次增量一条线段只会使得与它有关的不合法。
考虑这次增量的是什么线段：
0 类：每次右移左端点使得没有 1 连续段包含它；
1 类：每次右移左端点使得它的连续段不会包含任意一个 0 类。
考虑维护：
第一种，判断区间内是否有线段包含它。这是简单的二维前缀问题，直接加入一条 1 类的时候对于区间 $[1, l]$ 记录值 r，判断时取左端点 l 上是否有值不大于 r。
第二种，做区间覆盖，标记永久化线段树即可，维护区间至少覆盖一次的点数；查找连续段端点用的也是第二种线段树，直接线段树上二分即可。

Fair Nut 发现了一个长度为 n 的整数数组 a。我们称子数组 $l \dots r$ 为数组中下标从 l 到 r 的一段连续元素，即 $a_l, a_{l+1}, a_{l+2}, \dots, a_{r-1}, a_r$。
没人知道原因，但他称一对子段为“好对子”当且仅当满足以下条件：
1. 这两个子段不能嵌套。也就是说，每个子段都必须包含一个不属于另一个子段的元素（下标）。
2. 两个子段有交集，并且交集中的每个元素在每个子段中只出现一次。
例如，$a = [1, 2, 3, 5, 5]$。对子 $(1 \dots 3; 2 \dots 5)$ 和 $(1 \dots 2; 2 \dots 3)$ 是好对子，但 $(1 \dots 3; 2 \dots 3)$ 和 $(3 \dots 4; 4 \dots 5)$ 不是（子段 $1 \dots 3$ 包含了 $2 \dots 3$，整数 5 同时属于两个子段，但在子段 $4 \dots 5$ 中出现了两次）。
请你帮助 Fair Nut 计算好对子段的数量！由于答案可能很大，请输出对 $10^9 + 7$ 取模后的结果。

考虑一个区间 $[l, r]$ 造成的贡献，则有 $pre \leq L < l \leq r < R \leq nxt$。
则考虑扫描线维护所有的 pre, nxt。
直接上线段树 beats，维护两个序列以及对位乘法就秒了。
找一个 > 0 的点从最早合法左端点查即可。

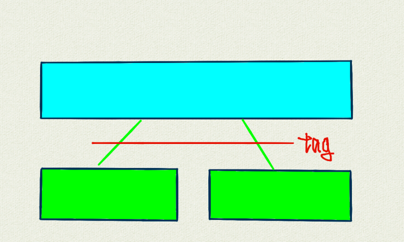
给定一棵树，你可以维护一个集合，支持三种操作：
1. 当前集合中插入一个节点 x 2. 撤回上一次插入操作 3. 将当前点集标为第 i 个点的子树补信息
一个点 x 的子树补信息定义为，树的点集除去 x 的子树（包括 x）内的点得到的集合；需要保证每个点的子树补信息都是正确的。
$n \leq 10^5$，你需要构造操作序列，使得 1 操作数不超过 5×10^6。

喵喵分治板子。
dfn 序列上：
对于中点 $dfn_u = mid$ 拿出来，不断跳 fa_u 直到不合法，则这个区间要处理的问题就是所有 u 及其祖先的的问题（合法时）。
由于祖先到 u 的区间 $[l, r]$ 的 l 单调不降，r 单调不增，所以区间内直接按照顺序线性即可。
挖掉 mid 之后递归两边即可。

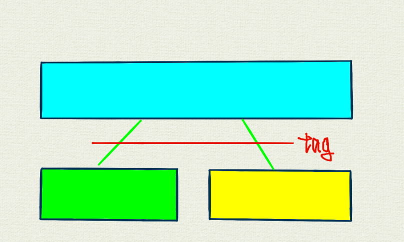
历史线段树

设一个节点的版本 ver 为它最后一次被修改的时间，这个时间单调不降。
历史线段树记录每个节点的历史版本值，包括它历史上的 ver 和访问版本之间的各个版本。
并记录它的各类信息（和，极值，函数值，$\cdots\cdots$）。
我们真正操作的版本只有访问到的版本，而之间的版本都是假想出来的。
对于 ver 之前的版本直接记录在节点上，且会通过节点现有的值计算 ver 之后的版本。

普通的线段树的节点有这样的信息：
每个节点存储信息，以及对于子节点的差 tag。



(a) 标记下传



(b) 标记永久化

区别在于，标记下传线段树只要存在标记，则 $ver_{ls} = ver_{rs}$，标记永久化则不一定。

历史线段树

所以设函数 $upd(o, v)$ 代表将节点 o 的版本 ver 对齐为 v 的过程（之间无修改）：
对于历史最值除版本号外不变。
对于历史和，向历史和累加版本差乘上和。
这个操作只在修改的时候适用。
而在下传标记/计算标记贡献的时候，会出现这样的情况：
$[ver_{son}, ver_{fa})$ 之间仍然存在版本，则现在需要计算这些中间版本的贡献。
对于历史最值，则需要维护这一段的 tag 的最值。
对于历史和，则需要维护这一段的 tag 的历史和。
也有将子节点 $upd()$ 之后再计算的写法，这个只适用于标记下传。
Tips：建议不将当前版本计入历史和，而在询问的时候加上即可。
相信大家都会 NOIP2022 T4，所以我们略过这道题（历史线段树题目太多啦！）。

CF1148H

Statement

你有一个初始为空的数组。你需要进行 n 次如下格式的操作：
$a\ l\ r\ k$：将 a 添加到数组末尾。
之后，统计有多少整数对 x, y 满足：
$l \le x \le y \le r$ 且 $\text{mex}(a_x, a_{x+1}, \dots, a_y) = k$
$n \le 2 \times 10^5$，强制在线。
In:
0 1 1 1,1 1 2 0,0 1 3 1,1 1 4 2,2 1 5 3
Out:
1 1 2 6 3

强制在线 = 主席树，子区间询问 = 历史和（还有更泛用的分治处理子区间询问）。
令主席树记录区间 l, r 的问题，考虑出现了 $k = a_r$，对 $\text{mex}(l, r)$ 造成什么影响：
首先找到 $L = pre_k + 1$，这是最左的可能以 k 为 mex 的端点；
再找到最右边 R 的满足 $[R, r]$ 包含了 $[0, k-1]$ 中间所有值的点。
这时 $[L, R]$ 即需要修改的区间。
区间的 mex 由 k 变大为不同的数字，这里区间覆盖的问题是简易的，倒着分析：每次会将一段不同的东西赋值成相同 k，则覆盖次数是珂朵莉（线性）的。
考虑求出分段，设当前找到的区间右端点为 tr，则直接找最小的 x 的最迟出现位置 $pos < tr$，分段即 $(pos, tr]$，注意 pos 与 L 取 max。

一开始对于 $[r, r]$ 的 mex 赋值为 0。
区间的 mex 由 k 变大为不同的数字，这里区间覆盖的问题是简易的，倒着分析：每次会将一段不同的东西赋值成相同 k，则覆盖次数是珂朵莉（线性）的。
考虑求出分段，设当前找到的区间右端点为 tr，则直接找最小的 x 的最迟出现位置 $pos < tr$，分段即 $(pos, tr]$，注意 pos 与 L 取 max。

维护出来后需要对于每一个 k 做主席树历史和，发现要标记永久化，很恶心。
这里直接对于覆盖转成加法，则现在考虑维护标记永久化的历史和：
正常的历史和线段树上，每次 $pushup()$ 的时候，**需要把子节点的版本手动对齐到父亲节点上**，这里只计算版本对齐贡献而不真的对齐。
现在我们询问的时候版本都是手动对齐询问版本的，则标记只要计算相交部分贡献即可。

填坑：上次的 Ynoi
Statement

平面矩形 chkmx，矩形求和，先操作再询问。
$n, m, q \leq 2 \times 10^5$。

按照 v 大小排序，覆盖没有被覆盖的点，分块维护即可。
然而这个很明显不可以用 sgt 维护了……

某联考题

Statement

卢瑞恩守望着泪水之城。

泪水之城可以被抽象为一个长为 n 的整数序列，初始为序列 $\{a_i\}$。在接下来的时间里，会依次发生 m 个事件。每个事件会是以下三种事件之一：

1. 区间 $[l,r]$ 内的元素增加了 v（v 可能是负数）。
2. 卢瑞恩派出了一个守望者，监视区间 $[l,r]$。
3. 卢瑞恩询问了第 i 个派遣出的守望者，守望者会回答它的监视结果。一个守望者的监视结果为，自它被派遣出至被询问的这一段时间里，它所观察的区间内的元素的最大值。

三类事件分别用 `1 l r v`，`2 l r`，`3 i` 来表示。

其实不需要其他守望者的报告，卢瑞恩也能根据发生的事件和初始序列计算出每个 3 类事件的监视结果。可是它算得太慢了，你能帮帮它吗?

形式化题意：给定一个长为 n 的，初始为 $\{a_i\}$ 的整数序列，接下来按时间发生上述 m 个事件，你要对每个 3 类事件，输出它的“监视结果”。**部分测试点强制在线。**

这形式化题意诗人啊?

某联考题

Solution

这个问题一看就是很简易的，上线段树维护。

求的是询问区间至今的历史最值，我们考虑直接把询问分成 $\log n$ 个节点插入线段树，做插入标记回收。

假设每个节点上的询问暴力更新：

我们插入的时候不能询问插入之前的答案，否则不合法，则插入节点的时候把相关的到根节点全部更新一遍，这样前面的最值就和当前的询问无关了。

询问的时候找到包含这个询问的所有节点，把它们的到根链 $pushdown()$ 并对齐到当前版本就得到了新版本的答案。

现在一个节点上插了不止一个节点，不方便直接暴力更新。

由于新的节点的答案不大于旧的节点的答案，则这里对于每个节点开一个单调栈，维护一个并查集即可。

复杂度 $\mathcal{O}(q\log n\alpha(n\log n))$。

有一颗以 1 为根的树，每个点上有一个点权 a_i，每次询问路径 u 到 v 上最大的 $a_i \operatorname{xor} dist(i, v)$，保证 u 为 v 的祖先。
$n \le 5 \times 10^4$。
巨大困难题。
李天呈讲了，大家看题解。

给定 m 个区间 $[l_i, r_i]$，每次询问给定区间 $[s_j, e_j]$，
查询是否能选择一些区间，使它们的并恰好为 $[s_j, e_j]$。
值域 10^5。

扫描线，询问对于当前右端点，哪些左端点是合理合法的。
线段视为左闭右开，维护每个点最后在哪个版本被取到。
操作形如给定 $[l, r]$，$\forall mx_i \geq l, mx_i \to r$。
考虑下传标记时合并标记，明显是这样的：
旧标记为 lt, rt，新标记为 L, R：
若 $L \leq rt$，则合并标记（显然）。
反之舍去新标记（为什么？因为操作之间留了一个空隙，在 $r\uparrow$ 时，
这个空隙被填补上的时候新标记必然无用了）。
这题后面会有加强版，大家注意不要用这题的思路写，会寄。

李超线段树

任意两函数在某个值域范围内只有一个交点，都可以使用李超树维护单点极值。
复杂度来源于单侧递归。
现在介绍优化版李超树（常数小，合并简易）：

- 实现新加线段若不能完全打败区间优势线段时两侧递归即可。
- 证明：考虑一个节点以及子节点存在的三条线段：
  新加一条线段，会向一侧递归，此时递归到另一侧必然被打败。
- 缺点是空间占用大，一般为普通李超树的两倍，注意取舍。

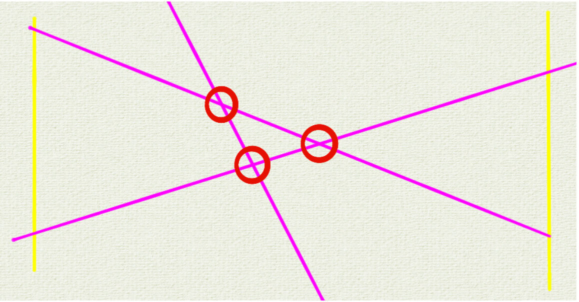


Figure: 优化李超树

给定大小为 n 的树，根节点为 1。每个节点有两个权值，第 i 个节点的权值为 a_i, b_i。
你可以从一个节点跳到它的子树内任意一个节点上。
从节点 x 跳到节点 y 一次的花费为 $a_x \times b_y$。
跳跃多次走过一条路径的总费用为每次跳跃的费用之和。
求每个节点到达树的任意叶子节点的最小费用，1 不为叶子。

贡献为 $kx+b$ 的形式，考虑李超树斜率优化，考虑直接包含了子树情况，使用李超树合并求解即完成。
李超树合并（改良版）：
考虑直接合并，由于李超树每一层都带有信息，则合并的时候先合并子节点再暴力插入两个线段，复杂度为 $\mathcal{O}(\log V)$，否则复杂度为 $\mathcal{O}(log^2 V)$。

贡献为 $kx+b$ 的形式，考虑李超树斜率优化，考虑直接包含了子树情况，使用李超树合并求解即完成。
李超树合并（改良版）：
考虑直接合并，由于李超树每一层都带有信息，则合并的时候先合并子节点再暴力插入两个线段，复杂度为 $\mathcal{O}(\log V)$，否则复杂度为 $\mathcal{O}(log^2 V)$。
不推荐练习：P2305 [NOI2014] 购票，醇构史。

公路上有 n 棵树，其中第 i 棵树生长在坐标为 x_i 的点上，高度为 h_i，x_i 有序。
可以按以下方式逐个砍伐树木：将树砍倒，并让它向左或向右倒下。
为了防止树木在倒下时受损，它不应碰到距离它自身高度以内的未砍伐的树木。
左边界 x_1 处的树木和右边界 x_n 处的树木旁边有重要的建筑物，因此不允许倒掉的树木落在区间 $[x_1, x_n]$ 之外。
给定 q 个伐木申请。每个申请由两个数字 l_i 和 r_i 组成，表示申请者希望砍伐第 l_i 到第 r_i 个树木（包括边界）。
一次询问只砍伐从 l_i 到 r_i 编号的树木，被砍伐的树木可以倒到 l_i 的左边或 r_i 的右边的区域，不能超出 $[x_1, x_n]$ 范围，不可触碰区间外其它树木。
每个申请独立（树没有被砍掉）。

考虑暴力贪心：
从一个左端点开始扫描，不断满足限制，则是要所有点一致向左倒塌。
如果不能向左，最好是向右，则最靠右没倒塌的树若离当前扫描到的树很远，也是可以向右倒塌的（中间都没了）。所以用栈维护。
废话：考虑一棵树向左倒塌只会受到左边限制，向右同理。
考虑维护，设最初能使 i 向左倒塌的点为 L_i，向右为 R_i，一棵树向左倒塌在需要满足限制 $l \le L_i$，向右需要满足 $R_i \le r$，且 $l \le i \le r$，把答案视为一个点，发现这是可以直接划分为两个矩形的，二维数点即可。

考虑整体二分，根据 L, R 必然分为连续段的性质，考虑到每次整体二分 check() 只和当前待判断的数组大小有关，直接对这个数组贪心就好。
时间复杂度 $\mathcal{O}(n \log n)$。

n 颗流星在数轴上:
初始正整数坐标 st，固定的速度 $\frac{v}{t}$，向负方向运动，拥有权值 w。
若流星在 T 时刻在整点上，则可以被观测到。
现在 q 次询问:
求 T 时刻坐标在 $[L, R]$ 之间的**可观测**流星的权值和。
注意:

$$st, v, t, T \le n \le 2 \times 10^5$$

LG P12978

Solution

看到取整点有效，先来个无脑根号分治，阈值 B，则问题转化为 $t_i, v_i \le B$ 部分的问题。
一个个确定交点不可取，但是对于 t_i 一定只能枚举，现考虑令更大的线段代替小的，查询大的线段的答案时直接查询了前缀答案。
考虑线段的集合，交点较多，直接对其分块，块长为 b，则现在每个集合 $\mathcal{O}(b^2)$ 个交点。
考虑集合不能直接分出，则现在最多有 $\mathcal{O}(\frac{n}{b}+B)$ 个集合，直接维护交点，在相交后重排有关点即可，树状数组维护答案。
时间复杂度：
$t, v > B$ 部分基排可做到 $\mathcal{O}(q\sqrt{\frac{n}{B}})$，查询 $\mathcal{O}(\log n)$。
分块部分共 $\mathcal{O}(B+\frac{n}{b})$ 个块。
每个块需要动态处理 $\mathcal{O}(b^2)$ 个交点。
交点离线基数排序做到 $\mathcal{O}(nb)$。
查询为 $\mathcal{O}(q(B+\frac{n}{b})\log b)$。
基数排序做法复杂度为 $\mathcal{O}((n+q)\sqrt{n\log n})$。
P.S. 快排冲过去了。

提示：这题很简单。
一个二维平面，一个事件抽象为该平面上的一个点，将一个时代抽象为该平面上的一个矩形。
记 $(a, b) \leq (c, d)$ 表示平面上两个点 $(a, b), (c, d)$ 满足 $a \leq c$，$b \leq d$。
更具体地，给定了 n 个**事件**，他们用平面上 n 个不同的点 $\{(x_i, y_i)\}_{i=1}^{n}$ 来表示；
智者还给定了 m 个**时代**，每个时代用平面上一个矩形 $(r_{i,1}, r_{i,2}, c_{i,1}, c_{i,2})$ 来表示。
我们称时代 i 包含了事件 j 当且仅当 $(r_{i,1}, c_{i,1}) \leq (x_j, y_j) \leq (r_{i,2}, c_{i,2})$。
智者认为若两个事件 i, j 满足 $(x_i, y_i) \leq (x_j, y_j)$，则这两个事件形成了一次**遗憾**。而对一个时代内包含的所有事件，它们所形成的遗憾被称为这个**时代的眼泪**，而形成的遗憾次数则称为该时代的眼泪的大小。现在智者想要小 L 计算**每个时代的眼泪的大小**。
$n, m \leq 10^5$，空间为线性根号级别。

扫描线 + 分块。

LG P7722/CF1545F

Statement

给你三个长度为 n 的数组 a, b, c。
有 m 次操作，每次操作为下面两种之一：
$1, k, x$：将 a_k 修改为 x。
$2, r$：求出有多少个三元组 (i, j, k)，满足 $1 \le i < j < k \le r$ 且 $b_{a_i} = a_j = c_{a_k}$。
Hint：分块或根号分治 +DDP。

LG P7722/CF1545F

Solution

这道题我是暴力分块 AC 的，不过那样比较无聊，这里提供一种 DP 思路（zak 的做法）。
zak 原文：
首先这个问题太诡异了，我们把问题看作可以对 a，b，c 都单点修改。要求 $i < j < k$ 的 $b_i = a_j = c_k$ 的对数。
考虑对于 $b_i = a_j = c_k = w$ 的 w 进行根号分治。考虑 w 在 a, b, c 中的总出现次数。

- 对于出现了 $< B$ 次的数，每次对这个数进行修改的时候暴力 DP，在 k 处加即可。
- 对于出现了 $\geq B$ 次的数，这样的数只有 $\Theta(\frac{n+q}{B})$ 个。所以对于每个数维护个动态 DP 即可。

取 $B = \Theta(\sqrt{n+q})$，这两种情况都容易平衡到单次 $\Theta(\sqrt{n+q})$，所以这种做法时间复杂度是 $\Theta(q\sqrt{n+q})$。

LG P12461

Statement

给你一张无向图 G，设 $R(i)$ 为所有与 i 相连的边的另一个端点构成的可重集合，可能有重边，没有自环。
每个节点有权值 w_i，初始时都为 0，需要维护两种操作。
$1, l, r, v$，对于 $i \in [l, r]$，$j \in R(i)$，令 $w_j = w_j + v$。
$2, l, r$，计算 $\sum_{i\in[l,r]} \sum_{j\in R(i)} w_j$。
$n, q \leq 10^5$。

LG P12461

Solution

并非图论。
错误地向图的性质思考，发现肯定不能向边动手脚，本题的重点在边上。
观察一次单点修改，实际上是对于这个点的连边操作，一个点的权值即所有相邻边权值。
把一条边拆成两部分，表示出边，每次要做的事情其实是同步相同目标出边的权值。
拉出一个序列 A 为出边的目标点，按照出点排序。
对这个序列分块，则容易知道一次操作会有这么几种情况：
1. 区间加-整块，即统计这个整块到区间的同值个数后累加。
2. 整块-散块区间，统计同值个数后每次累加到整块上，散块累加给答案。
3. 散块-散块，与点有关，直接把权值累加给点。
做完了，时间复杂度 $\mathcal{O}(n\sqrt{n})$。
竹筷离线。

CF1464F

Statement

给你一棵树，需要你维护一个路径集合（可重复）P。
三种操作：
1. 插入一条路径 (u,v)；
2. 删除已加入的一条路径 (u,v)；
3. 判断当前路径集合所有路径的 k 邻域是否有交，k 是每次询问给出。

k 邻域：若存在某个点 p 与路径上一点 v 的距离不超过 k，则它在本路径的 k 邻域里。
$n,m \le 10^5$。
Hint：CF1827E。

考虑 k 邻域有交的充要条件：两两距离不超过 $2k$。
随便定一个根，取每个链的 lca 为 u_i。
重新考虑交点位置，取最深的 u_i，则交点必在到根链上。
现在向上提 k 为 x，向上提 $2k$ 为 z，则要求所的链都和 z 的子树有交。
而子树内则是：距离 x 不超过 k。

CF1464F

Solution

判断子树交用树状数组即可维护，而子树内找最远距离则是对于子树维护一个直径，找直径端点。
直径用 sgt 维护即可。

两种做法的题目，看看大家会哪一种。
给你一棵 n 个节点的树，每个节点有一种颜色，有 m 次查询操作。
查询操作给定参数 $l\ r\ x$，需输出：
将树中编号在 $[l, r]$ 内的所有节点保留，x 所在连通块中颜色种类数。
每次查询操作独立。
第一行两个数 n, m。
第二行 n 个数表示每个节点的颜色。
之后 $n-1$ 行，每行两个数 x 和 y，表示 x 和 y 之间连有一条边。
之后 m 行，每行三个数 $l\ r\ x$，表示一次查询操作。
对于 100% 的数据，所有出现过的数在 $[1, 10^5]$ 之间，保证每次输入的 $l \leq x \leq r$。

这道题的大体做法是在点分治上挂询问。
然而我们有更美妙的做法：
克鲁斯卡重构树，边权为节点的更大/小值。
问题转化为对于双树子树交集数颜色。
考虑 Dsu On Tree，对于另一棵树维护树状数组，加入/删除的时候维护贡献即可。
复杂度 $\mathcal{O}((n+q)\log^2 n)$，常数奇小。

给定两棵 n 点异构树，定义 x, y 距离为：

$$\mathrm{depth}(x) + \mathrm{depth}(y) - (\mathrm{depth}(\mathrm{LCA}(x, y)) + \mathrm{depth}'(\mathrm{LCA}'(x, y)))$$

输出每一对点对 i, j（$i \leq j$）的如上定义的“距离”的最大值。

大型分讨题。
考虑询问的形状。
给定两棵树 $t1, t2$，接下来带的下标就是树编号。
对于任意可相同两点 x, y，求:
$(d_1(x) + d_1(y) - d_1(lca_1(x, y))) - d_2(lca_2(x, y))$
即：这两点在 $t1$ 上的路径并减去在 $t2$ 上的路径交。
考虑 Dsu On Tree，路径并不是很好确定，考虑路径交一定是当前 $t2$ 上 $lca(x, y)$ 的深度。
用 Dsu On Tree 化简了 $t2$ 上的问题，转而到了求 $t1$ 的问题：
合并的根为 u，对于已有的一个点集 S，对于当前的一个点 x，对于任意的 $y \in S$，$lca_1(x, y) = u$，现在要求：对于每一个 y，x 与 y 在 $t2$ 上到根路径并的最大值。
考虑对 $t2$ 树剖，我们考虑在 lca 处统计答案。

有这样几种情况：
1. x 子树内的任何点可以作为答案。
2. x 从重儿子跳上去，则任意 ** 祖先 ** 的轻儿子作为答案。
3. x 从轻儿子跳上去，任意非 x 所属的轻儿子和重儿子都作为答案。
考虑这怎么维护，直接树状数组对每条重链维护前缀最大值，可以解决 case 2，维护后缀最大值可以解决 case 1。
前缀最大值即：对于一个节点 u，它的轻儿子内节点的最深深度为 D，则权值为 $D - d_2(u)$。
后缀最大值即：对于一个节点，它的子树内的节点最深深度为 D，则权值为 D。
考虑 case 3，这里考虑直接用堆维护一个每个轻儿子内的最深深度即可，用的时候是均摊 $\Theta(1)$ 取的，重儿子直接查 v 在链上的后继的后缀最大值即可。
所以总共需要维护两个 BIT，一个堆。
时间复杂度 $\mathcal{O}(n \log_2^3 n)$，实际跑很快，最慢点 $2s$ 左右。
注意细节，合并的时候注意单点情况，所以对于当前点 u，是先 $ins(u)$ 再 $query(u)$ 的。

LG P7983
Statement

给你两个长为 n 的序列 a, b，请你支持三种操作共 m 次：
1 l r x，将 a 序列的区间 $[l, r]$ 中的所有数修改为 x。
2 l r y，将 b 序列的区间 $[l, r]$ 中的所有数修改为 y。
3 l r，求 $\sum_{i=l}^{r}\sum_{j=1}^{b_i} a_j$。答案对 2^{32} 取模。
$n, m \leq 3 \times 10^5$。

考虑对序列 B 分块，这样便于处理整块贡献，发现散块容易通过对 A 直接维护 $\mathcal{O}(\sqrt{n})-\mathcal{O}(1)$ 的分块暴力。
先不考虑 2 操作。
每次我们对 A 推平，即用珂朵莉树转化为线性次区间加，之后考虑对于序列 B 上整块的贡献。
对于前缀和，区间加转化为两次等差数列加即可，一次等差数列加的贡献记为：(x,k)，即 $\forall b_i \ge x, b_i \to b_i + k(b_i - x)$。
考虑整块贡献，每次直接取所有 $b_i \ge x$，对整块累加贡献即可，这个在没有修改的时候容易用前缀和解决。
加入对于序列 B 的推平操作：
考虑一个不好的事情：
分块后被散块修改的区间都需要重置，要求重置做到 $\mathcal{O}(\sqrt{n})$ 明显不太可能。

首先考虑需要什么样的数据结构可以维护：
对于线性次修改，我们只需要维护一个 $\mathcal{O}(\sqrt{n})-\mathcal{O}(1)$ 的分块即可。
问题在于散块的修改是 $\mathcal{O}(\sqrt{n})$ 的，我们并不能让散块的修改变成常数次的（笑）。
考虑如何不全部重构？即想让修改次数转化为线性。
有以下的结论：
序列分块下，在所有块内部维护珂朵莉树，总复杂度为 $\mathcal{O}(q\sqrt{n}+(q+n)\log n)$，其中第二部分恰好为散块操作复杂度，证明显然。
则我们先对序列分块，一次推平操作，整块打标记（注意这里不能直接打，如果之前有散区间需要先清空，有标记的区间直接通过标记加贡献），散块对珂朵莉树修改，改贡献。
则我们需要统计的信息有：
1. $\mathcal{O}(n+q)$ 次散块推平时被推平的 b_i 前缀和。
2. $\mathcal{O}(q)$ 次散块操作时推平的 b_i 前缀和。
3. A 序列的珂朵莉树在每次操作后的贡献。
没了，总共需要写三个分块，两棵珂朵莉树，注意以下离线操作和升天的常数。

LG P10081/P10148

Statement

给定一个长度 n 的整数序列 $a_1, \ldots, a_n$ ；
给定一个由 m 次操作构成的操作序列，操作从 1 开始编号，到 m 结束。操作序列中包含修改操作和求和操作，修改操作给定 l, r, v ，将 $a_l, a_{l+1}, \ldots, a_r$ 修改为 v ，求和操作给定 l, r ，查询 $\sum\limits_{i=l}^{r} a_i$；
共 q 次查询，每次查询给出 L, R ，询问将序列 a 初始化为 0 后，依次进行操作序列中的第 $L, L+1, \ldots, R$ 次操作，每次求和操作的答案之和。
$n, m \leq 3 \times 10^5$。

对操作扫描线，执行序列分块，则额外维护一个分块进行加操作。
序列分块部分，每个块内维护珂朵莉树，则会进行 $\mathcal{O}(q\sqrt{n}+(n+q))$ 次覆盖操作，其中前面的 $\mathcal{O}(\sqrt{n})$ 次通过块上打标记实现，则 odt 实际参与的覆盖次数是线性级别。
对询问扫描线设当前扫描到 R，则维护 $[L,R]$ 的序列值是简易的，然而我们不可以在每个询问处都处理出来。
考虑一次 2 操作的贡献：
对于散块的处理是简易的，利用常数-根号次修改的分块对时间轴维护即可，接下来考虑整块：
有 $\mathcal{O}(\sqrt{n})$ 次整块。
若整块打了标记 (col,t)，则现在相当于贡献到时间 $[1,col]$ 整块和，其他部分为 0，则在同一个分块上更新即可。

反之，整块为珂朵莉树形式，这里对于不同时间有不同的贡献，暴力操作会有 $\mathcal{O}(n)$ 的复杂度，不可以直接维护。
首先每个块维护一个时间轴的分块代表起始时间在 L 的答案。
假设一个块没有修改，则在不改变的时候维护一个标记 cnt，代表整个块被询问了多少次。
则我们每次询问就直接查询 L 处的答案即可，乘上 cnt 即可，这一部分要 $\mathcal{O}(1)$。
带上修改（即 $\mathcal{O}(n+m)$ 次散块变化），则在保留 cnt 的情况下，直接将新加的贡献 (col, t) 算入，同时在本块维护的分块上减去被覆盖的 (col, t) 的 cnt 次贡献。

LG P10574

Statement

层叠都市可以被抽象为一棵树。于是给你一棵带点权 v_i 的树，树以 1 为根。初始点权 v_i 均为 0。
定义 $\mathrm{dis}(x, y)$ 为树上 x, y 之间的距离，即 $x \to y$ 的简单路径上的边数。
设 $\mathrm{subtree}(x)$ 为树上以 x 为根的子树，定义 $f(x) = \max_{d\geq 0} \sum_{y\in \mathrm{subtree}(x)} v_y[\mathrm{dis}(x, y) = d]$。
也就是说，$f(x)$ 表示 x 子树中的每一层的点权和的最大值。
现在给出 m 次操作，每次操作中给出 x, w, y，先令 $v_x \leftarrow v_x + w$，然后求 $\sum_{i\in \mathrm{subtree}(y)} f(i)$。
$n, m \leq 3 \times 10^5$。

首先观察，求一个点被哪些值贡献并不现实，考虑反过来求一个点增加的时候的贡献（题目初始全 0 也提醒了这一点）。
d_x 为点 x 的深度。
首先发现我们向上的过程中，对应的值会发生变化，设改变点为 x，有一祖先点 u，当且仅当有 $v \in subtree_u, d_v > d_x$，且 x, v 不属于同一子树。
则改变次数不超过 $\mathcal{O}(\sqrt{n})$，我们是对这些链上的点值 chkmx。
但是！这个东西很不友善，我们考虑对每个点求出来分界点不是很现实（空间不够），考虑直接对点进行根号分治来维护：
设阈值 S，对于 $d_u - d_x \le S$ 的，直接对到达每个 u 点的点值暴力算出，反之若 $d_u - d_x > S$，就只有在 u 有另一子树的 $mxdep - d_u > S$ 的时候再取了。
很自然地联想到重剖，直接对树重剖，则第二种点就是轻子树 $mxdep - d > S$ 的点，这个扫一遍就好。

回过头来考虑需要维护什么东西：
1. 跳链的时候，无法区分轻重，需要手动找子树内某层的和，这个按照层数维护一个 BIT，层内按照 $dfn[]$ 排序，每次找的时候二分一下区间求和。
2. 反之就加上轻子树内某层的和，这个拿桶记一下就好了，个数是 $\mathcal{O}(n\log_2 n)$ 的。
第一种情况总复杂度 $\mathcal{O}(n\log_2^2 n)$，第二种 $\mathcal{O}(S+\frac{n}{S})$。
考虑维护树剖，sgtbeats 链修改，子树（区间）查询，平衡阈值情况下复杂度 $\mathcal{O}(n\sqrt{n\log_2 n})$。
现在考虑怎么优化这个算法：
首先对于每条重链单独开线段树（这样也就不用真写 sgtbeats 了）：
查询的时候只要用 BIT 维护重链按照链头的顺序的和，并查询所在重链一段后缀和。
单次操作的复杂度是对数的，但我们在多次操作的过程中，多次重复访问了很多点，假设这些点只会访问一次，是否就能使复杂度带的 $\mathcal{O}(\log)$ 大大退化？考虑把每个操作离线下来统一更新。

每次操作的长度总和为 $\mathcal{O}(n)$，单次操作的复杂度为 $\mathcal{O}(\log dif)$。
总和为 $O(\sqrt{n})$。就是 $\sum a_i \le n$，那么 $\sum \log(a_i - a_{i-1}) = O(\sqrt{n}))$。
抄来的证明：
考虑 a 的差分数组翻转后设为 b，那么就是 $\sum b_i \times i \le n$，$\sum \log(b_i?) = O(\sqrt{n})$。
记 $|b| = m$，考虑均值不等式 $\prod(b_i \times i)^{1/m} \le (\sum b_i \times i)/m \le n/m$。
取对数导一下发现是 $\log n - 2\log m - 1$，所以 m 大概就是 $\sqrt{n}$ 的时候最大（因为 $m \le \sqrt{n}$，常数项不影响），取到 $O(\sqrt{n})$。
中间有 $\mathcal{O}(\log)$ 次跳重链操作，这个花费 $\mathcal{O}(\log^2 n)$，不影响。

LG P11695/P12082

Statement

给定 n, m，你需要维护一个 $[1, n)$ 的数轴上区间的初始为空的可重集合，支持三种操作共 m 次：

1. 插入一个区间 $[l, r)$。
2. 删除第 t 次操作插入的区间。
3. 给出一个区间 $[l, r)$，判断当前可重集合是否存在一个子集，使得子集中所有区间的并恰好是 $[l, r)$。

首先明显是要求判定这样一个问题：一个询问 $([l,r),t)$ 是否能被所有的子区间在包含 t 的时间内完整覆盖。
找出这个区间完整包含是不太明智的，发现我们很难用分治结构刻画。
我们发现扫描线必然是扫描位置的，但是与此同时我们也需要时间扫描线。
接下来整区间，线段都表示为 $[l,r)$，情况都以中点以左的区间为例：
猫树，把询问 $[l,r)$ 插在中点上。
刻画的时候发现与区间有交必然不行，我们只考虑接下来两种线段：

LG P11695/P12082

Solution

1. 同样插在中点上 $[lt, rt)$：
$lt \ge l, rt \le r$，我们取最靠近右边的，这个直接线段树二分（底层 set 或对顶堆）即可维护，按照时间操作。
记下这一部分最远覆盖到 ext。
时间复杂度 $\mathcal{O}(m \log m)$。

2. 在两边。
这一部分即让我们要求总覆盖区间 $[L, R)$ 满足 $L = l$，$ext \le R \le mid$ ，mid 的限制不存在，即右端点在一段到 R 的前（后）缀上。
这里加在序列上的限制较多，考虑：对序列扫描线，并记录左端点能到达的右端点在哪里。
考虑限制，限制少了一个右端点包含，多了一个时间轴。
怎么刻画固定左端点的限制？首先我们扫描到询问的时候，必然插入这个询问到它对应的时间上。

发现并不能用右端点扫描，这样就无法用原题的标记合并方法了。其实也没有必要扫描右端点，因为不存在右端点包含的限制。
考虑左端点从左到右扫描：
我们每次询问的时候点亮一个点，每次操作到新左端 l 手动删去所有不满足 $rt \geq l$ 的已点亮时间点，这一部分复杂度为 $\mathcal{O}(m \log m)$，每个点只会被删去一次。
加入一段区间则也在左端点加入，对还点亮的点区间 $chkmx()$ 即可，这一部分复杂度 $\mathcal{O}(m \log m \log n)$，每个操作被拆成最多 $\log n$ 个区间在线段树上。
在询问右端点 $mid - ext$ 时收获即可，即查看当前的时间点是否仍然点亮。

题单链接。