数据结构

CF679E BEAR AND BAD POWERS OF 42

定义一个正整数是坏的，当且仅当它是 42 的次幂，否则它是好的。

给定一个长度为 n 的序列 a_i，保证初始时所有数都是好的。

有 q 次操作，每次操作有三种可能：

- `1 i` 查询 a_i。
- `2 l r x` 将 $a_{l...r}$ 赋值为一个好的数 x。
- `3 l r x` 将 $a_{l...r}$ 都加上 x，重复这一过程直到所有数都变好。

$n, q \le 10^5$，$a_i, x \le 10^9$。

- 这个42它一点用都没有。
- 它只能告诉我们，再不考虑二操作的情况下，三操作中，所有数最多只会
- 加n*logn次。
- 再加上线段树就是两个log。
- 具体来说，我们用线段树维护当前序列元素离下一个42的次幂的差。
- 这样三操作就是区间减了。

- 这样我们每次暴力区间减。
- 减完之后如果区间最小值小于0，就递归子树，找到对应位置，更新它的差。
- 然后再判断区间最小值是不是0，如果是，重复上面的步骤。
- 然后我们再考虑二操作。
- 区间赋值很容易就让人想到类似于珂朵莉树的乱搞写法。
- 一般来说，区间赋值问题中，我们也经常将权值一样的压缩成一段。

- 这题也一样，直接将相邻一段压成一样的。
- 对于每一段，我们只需要记录这一段中最右边点的权值。
- 然后再修改的时候，我们先把l-1和r的权值都独立出来。
- 也就是如果有一段只有部分在我们操作的区间中，我们就把这一段断成两半。
- 这样每次操作只会新增几个数，所以总复杂度还是不变。
- 上面那个颜色段直接用一个set维护就可以了，我们把每一段的最右边放入set里面，
- 然后一个位置的权值就是set中第一个大于等于它的元素。

CF571D CAMPUS

题意翻译

有一个长度为 n 的序列，初始全为 0。

有两类对下标的集合，初始时每一类各有 n 个集合，编号为 i 的集合里有下标 i。

一共有 m 个操作，操作有五种：

1. `U x y` 将第一类编号为 y 的集合合并到编号为 x 的集合里。
2. `M x y` 将第二类编号为 y 的集合合并到编号为 x 的集合里。
3. `A x` 将第一类编号为 x 的集合中的所有下标在序列中对应的数加上 x 的集合大小。
4. `Z x` 将第二类编号为 x 的集合中的所有下标在序列中对应的数设为 0。
5. `Q x` 询问序列中下标为 x 的位置上的数。

$n, m \leq 5 \times 10^5$。

- 每次合并集合，然后修改操作只有集合加权和集合赋值。
- 并查集基本就写脸上了。
- 很显然，对于4操作我们只需要维护一个最大赋值时间。
- 值得注意的是，我们还需要考虑合并两个连通块的赋值时间。
- 这里就显然涉及到两个连通块的连通时间，因为只有在连通之后的赋值才是有用的。
- 所以我们直接使用按秩合并并查集

- 然后每条边就可以存集合的合并时间，然后我们求一个点最后一次清空时间，只需要去遍历这个点所有的祖先，把清空时间取max就可以了。
- 然后我们知道了清空时间，就只需要统计这个时间之后的加权了。
- 这里同样对于每个点经历过的加权都开一个vector存储，并且累一个前缀和。
- 那么我们对于询问点求和的时候，仍然是暴力遍历祖先，然后二分一下，就可以得到对应的加权了。
- 这是在线的做法，复杂度两个log

- 这个做法基本就和Kruskal重构树一样。
- 复杂度还可以离线少掉一个log。
- 我们把树建出来，然后每一个询问要询问log个点，每个点都是询问[L,R]这一段的区间和。
- 其中L是上一次被清空的时间还有并查集合并时间取max，R表示当前询问的时间。
- 那么我们可以把这n*logn个点排序，这里使用计数排序即可。

CF407E K-D-SEQUENCE

题意翻译

找一个最长的子区间使得该子区间加入至多 k 个数以后，排序后是一个公差为 d 的等差数列。

多个解输出 l 最小的。

- 公差为0的先特判掉。这个判法就是找最长的相同段。
- 然后我们再来考虑一段区间什么时候合法。
- 首先区间里的元素%d的余数要相同，而且区间里的元素不能有重复。
- 根据余数相同，我们能根据余数把原序列分成一段一段，方法和上面找最长相同段一样。
- 然后我们对于每一段考虑答案。

- 我们考虑枚举右端点。
- 对于新加入的R，我们可以轻松知道上一个a[R]的位置，所以我们只需要把左端点和上一个a[R]的位置+1取max，就可以满足没有重复的条件。
- 我们再来考虑一下我们需要插入多少个数才能使得区间合法。
- 因为所有的数%d都同余，所以我们先把所有的数都除以d并向下取整。
- 那么问题就变成了加入若干个数，使得公差为1.
- 所以一个区间需要加入的数的数量就是max(L,R)-min(L,R)+1-(R-L+1)

- 一个区间合法就要满足max(L,R)-min(L,R)+1-(R-L+1)<=k
- 也就是说max(L,R)-min(L,R)+L<=k+R
- 我们只需找到满足条件的最小的L，这点线段树二分就可以了。
- 所以我们只需要维护max(L,R)-min(L,R)+L就可以了。
- 对于+L非常好维护。
- 我们接下来考虑怎么维护max(L,R),实际上我们会发现，加入R，只会导致一段后缀的最大值发生变化。

- 所以我们只需要使用一个单调队列去维护每个最大值统治的区间。
- 这样max就变成了简单的区间加减。
- min的写法同理。
- 这样我们就对于每一个R得到了最小的L，最后答案取最优即可

CF453E LITTLE PONY AND LORD TIREK

题意翻译

- 维护一个长度为 $n(1 \le n \le 10^5)$ 的数组 s，初值给定且 $0 \le s_i \le 10^5$。
- 每过 1 单位时间，都会对于所有 $[1, n]$ 之间的整数 i，执行 $s_i \leftarrow \min\{m_i, s_i + r_i\}$。其中 m_i 和 r_i 给定且 $0 \le m_i, r_i \le 10^5$。
- 维护 $m(1 \le m \le 10^5)$ 次操作，每次操作时会将时间设为 $t(0 \le t \le 10^9)$，并且查询 s 数组 $[l, r]$ 区间和。每次操作结束后会将 s 数组的 $[l, r]$ 区间推平为 0，保证 t_i 递增。
- 初始没有进行操作时时间为 0。

- 我们先不考虑区间清空和区间求和。
- 我们假设是对于某一个时间全局求和。
- 那么很显然我们先将所有元素按照(m[i]/r[i])从小到大排序。
- 然后对于当前这个时间，二分一下，就变成了前缀贡献都是m[i],后缀贡献都是r[i]*t
- 所以我们只需要前缀m[i]求和,后缀r[i]求和就可以了。
- 开两个线段树就可以了

- 然后我们再考虑区间求和的问题，区间求和就多了下标这个维度。
- 我们把线段树换成可持久化线段树就能消掉这个维度。
- 还是把所有元素按照(m[i]/r[i])，然后建一个可持久线段树。
- 可持久化线段树存的是下标。然后一次查询前缀就相当于二分到对应的主席树，
- 然后主席树上区间[L,R]求和。
- 然后我们再考虑区间赋值
- 这题非常简单，只是单纯的清零，也就是说，对于被清空过的询问，我们再次访问的时候，他就相当于询问的时间变小了。

- 所以清空操作实际上就是对一段区间的时间操作，而且每次都还是赋值。
- 这个时候无脑珂朵莉就可以了。
- 用set维护时间相同的一段，每次最多只会增加O(1)的时间段。
- 然后对于时间相同的时间段，假设上一次被询问时间为t1,当前询问时间为t2
- 那么这一段时间的贡献就是在t2-t1这个时间的贡献。

CF1332G NO MONOTONE TRIPLES

题意

给出一个长度为 n $(3 \le n \le 2 \cdot 10^5)$ 的序列 a，q $(1 \le q \le 2 \cdot 10^5)$ 次询问，每次询问给出一段区间 $[l, r]$，需要找到 $[a_l, a_{l+1}, \dots, a_r]$ 中最长的一个子序列 b，满足 b 中的最长不下降子序列和最长不上升子序列的长度都不超过 3。

- 首先b数组的长度不会超过4。
- 我们可以用反证法来证明。
- 假设有一个b数组长度至少为5.
- 那么首先这个数组不能不存在下降。
- 如果存在下降，那么最长下降子序列长度至少为2。
- 又不能为3，否则就已经不合法了。
- 根据dilworth定理可知，最长下降子序列长度=最少不下降子序列划分数。
- 所以存在一个不下降子序列的长度至少为3。
- 与条件矛盾

- 所以我们只需要考虑三元组或者四元组即可。
- 然后又因为要多组询问。
- 所以我们枚举i作为b1，求出最小的合法b3的位置和合法b4的位置。
- 这样询问[L,R]时，只要区间查询最小值判断是否<=R即可。
- 接下来考虑如何维护。
- 首先对于一段区间，只要不是单调的，就一定存在合法三元组。

- 所以对于一个i作为b1，最小的b3位置就是最小的R满足[i,R]不是单调的。
- 这里从头O(n)把单调段扫出来，就能维护b3的位置了。
- 然后对于b4，我们显然有结论b1和b4都不会是区间中的最值。
- 这里手玩一下容易证明。
- 那明显b2和b3就肯定一个是最小值一个是最大值。
- 那我们倒序维护两个单调栈。b2和b3就是单调栈中的元素

这样就能确定b2和b3的位置，然后对于b4就是在b3后面选择一个最小的不在单调栈中的元素，用一个set即可。

CF960H SANTA'S GIFT

题意翻译

给一棵n个点的有根树，每个点有一个颜色a_i，对于每种颜色有一个权值b_i。再给你一个常数C。

共进行Q次操作，操作分为两种：

- `1 x y` 把节点x的颜色修改为y
- `2 x` 对颜色x进行询问。求$\frac{\sum_{i=1}^{n}(S_i \times b_x - C)^2}{n}$，其中$S_i$是$i$的子树中，颜色=x的点的个数。

$2 \le n \le 50000$,$1 \le m, Q \le 50000$

$1 \le a_1, a_2, ..., a_n \le m$

$1 \le b_1, b_2, ..., b_m \le 100$

- 看上去是个难题，但实际上非常简单。
- 首先先无视/n,那么问题就变成了求 $\sum(S_i b_x - C)^2$。
- 我们把式子展开就变成了求 $\sum_{i=1}^{n} S_i^2 b_x^2 - 2S_i b_x C + C^2$
- 也就是求 $nC^2 - 2b_x C \sum_{i=1}^{n} S_i + b_x^2 \sum_{i=1}^{n} S_i^2$
- 所以这题的限制条件都是扯淡的，我们只需要维护Si和Si的平方即可。

- 然后修改一个点就对应到根的路径+1或者-1。
- 这里直接树剖线段树。
- 首先Si的维护非常简单，大家也都会。
- 那我们来维护Si^2的变化。
- 因为(a+1)^2-a^2=2*a+1
- 所以区间+1,Si^2就是增加2*Si+1

- 然后考虑到区间更新有懒惰标记的问题，我们一次不一定只修改1。
- 这个时候Si^2的增加量就是2*Si+1,2*Si+3,2*Si+5……
- 这就是一个等差数列。
- 所以我们修改的时候再加一个等差数列求和就可以了。

CF1344E TRAIN TRACKS

题意翻译

给定 n 个点的一棵树，以 1 为根，边有边权。

有 m 辆从 1 开始到 s_i 的火车，每个火车有一个初始时刻 t_i，在初始时刻时从根出发，向着目标 s_i 前进，当火车在 x 时刻到达一个点 u 时，假设下一个路径上的点是 v，两点间边权是 d，则在 $x+d$ 时刻到达 v，火车在到达目标点后停止。

由于每个点可能可以到达多个点，每个点有且仅有一个当前时刻可以到的儿子，每秒钟只能切换某一个点可以到的儿子，切换比火车开动先进行，若一辆火车走向了非目标方向的点，则立刻自爆。每个点初始能到的儿子是确定的。

求出能否不发生自爆，如果不能，第一次自爆最晚什么时候发生？在此基础上，至少切换多少次一个点可以到的儿子。

- 首先，我们来考虑什么时候一个点需要切换。
- 很明显的是，如果自己这个点到达点x的时间为R，上一个到达点x的时间为L
- 且他们需要走的方向不一样。
- 那么这个时候，我们需要在(L,R]这段时间里转动一次。
- 那么我们考虑一个节点的区间，问题就变成了有K个区间，我们每个时刻可以对一个区间操作一次，如果有一个区间没有被操作到，我们就失败了，求最晚失败时间。
- 这里我们可以离散一下，维护一条扫描线t，然后每次操作我们就是选择L<=t，并且R最小的还没有被操作的区间操作.

- 这个写法看上去是暴力的写法，但实际上复杂度是对的。
- 此时我们可以发现产生的区间数量最多只有n*logn。
- 证明如下，我们进行树链剖分，我们将当前节点指向的边称为偏好边。
- 那么显然，我们将所有火车排序以后，每次最多产生log条非重边的偏好边。
- 这log条非重边最多产生log个区间，而对于重边产生的区间，如果一个重边产生了
- 区间，就说明它不是偏好边，而每产生一个区间，非重边的偏好边就会减少1。
- 所以总共的区间数量非常少。

- 接下来考虑如何维护产生区间的这个过程。
- 很明显，我们每次都是把一条条链接边断边，每次维护根到某一个节点的路径。
- 这就是LCT的access，所以我们直接使用LCT,access若干次就可以得到所有区间。

CF487E TOURISTS

题目大意：给定一张图（保证连通），每个点有点权。现在有两种操作：

1. $C\ a\ w$:把a的点权改为w;
2. $A\ a\ b$:询问从a到b的所有简单路径（不经过重复点）中，点权最小的点的点权。

- 维护两点之间点权最小的简单路径。
- 那实际上就是写一个点双联通。
- 点双联通满足任意两点之间存在至少两条不重复的路径。
- 所以我们可以得到如果我们到达了一个点双联通分量。
- 那么我们一定能经过这个双联通分量的最小值。
- 所以我们就可以建一个圆方树

- 圆方树上圆点表示原来的点，方点表示一个点双联通分量。

如果有不会圆方树的同学可以理解成这是一种数据结构，满足每一个方点和自己这个联通分量里的点直接相连，然后把图变成了树。

那么方点的权值就是和自己相邻的圆点权值的最小值。

然后一次询问就是求一条路径上的最小值。

直接树链剖分线段树就可以了。

然后我们考虑修改。

- 因为一个圆点可能和多个方点相连，所以我们不能直接暴力修改所有方点的权值。
- 此时我们修改方点的定义为，自己儿子中和自己相连的圆点权值的最小值。
- 这样每次圆点就只需要更新自己的父亲就可以了。
- 还需要注意的是，如果求出的路径的LCA是方点，那我们还需要把他的父亲计入答案。

UOJ 637. [美团杯2021] A. 数据结构

- 给一个长为n的序列，m次查询：
- 如果将一个区间中所有数都+1，那么整个序列有多少个不同的数？
- 询问间独立，也就是说每次查询后这个修改都会被撤销

- 这种求不同元素数量的题目，一般考虑一个元素值对答案的贡献。
- 那么我们考虑值x什么时候对询问产生贡献。
- 值x产生贡献无非就讨论两种情况。
- 一种是讨论x什么时候出现
- 一种是讨论x什么时候没有出现
- 然后我们发现讨论x没有出现的情况明显更简单。

- x没有出现就只需要满足:

1.原来所有的x都要加1

2.原来所有的x-1都没有被加过。

那么显然所有x都要加1，我们能得到x最小的位置p1，x最大的位置p2

对于我们的区间显然要满足L<=p1,p2<=R。

然后再加上第二个限制就变成了区间的左端点的合法范围是一个区间，右端点也是一个区间.

- 这样区间就变成了一个矩形，然后我们的询问就变成了单点求值。
- 接下来就是离线扫描线就可以了。

CF643G CHOOSING ADS

题意翻译

- 给定一个长度为 n 的序列和一个整数 p。
- 有 m 个操作，操作要么是区间赋值，要么是询问区间内出现次数至少占 $p\%$ 的数。
- 输出询问的答案时，可以包含错的数，也可以重复输出，但对的数一定要在答案中，且输出的数的个数不超过 $\lfloor \frac{100}{p} \rfloor$。
- $n, m \le 1.5 \times 10^5$，$20 \le p \le 100$。

- 求区间众数有一个叫摩尔投票法的东西。
- 我们先假设p=51,那我们就是求区间的绝对众数。
- 对于绝对众数来说，我们每次都选两个不同的数删掉，最后剩下来的一定是绝对众数。
- 那么对于 $k=\left\lfloor\frac{100}{p}\right\rfloor$,我们每次只需要选择k+1个不同的数删掉，最后剩下来的数就一定包含我们所要求的数。
-

- 所以问题就变成了我们每次要选择K+1个不同的数删掉，而且还要维护一段区间的答案。
- 那么我们将问题进一步形式化。
- 同样从p=51的情况入手。
- 每次都选两个数删掉，就相当于我们对于当前的众数维护一个HP，然后我们逐渐加入新的数，如果新数和原来的众数不同，则HP-1，否则HP+1
- 如果HP=0了，那么众数就是新数，新数HP为0
- 这个HP就相当于当前这个数还存在的数量。

- 然后对于其他情况，就相当于我们要维护K个HP。
- 然后每次新加入的数，如果和原来的K个数相同，就对应的数HP++
- 否则就全员HP--，如果有数的HP为0，就被新数替代。
- 这个过程就相当于模拟了每次选K+1个数消掉的过程。
- 然后显然的是，我们每次选k+1个数的顺序无所谓。
- 所以我们能用分治，把一个区间裂成两个区间，再把两个区间的众数合并起来。
- 合并一次复杂度为K^2

- 然后我们要询问一段区间，所以我们只需要用线段树维护区间众数。
- 然后对于一次询问全部合并起来就可以了。
- 复杂度O(q*k^2*logn).

CF1017G THE TREE

题意：

给定一棵树，维护以下3个操作：

1，1 x表示如果节点x为白色，则将其染黑。否则对这个节点的所有儿子递归进行相同操作

2，2 x表示将以节点x为root的子树染白。

3，3 x表示查询节点x的颜色

输入格式：

第一行n,q两个数表示树的大小和询问个数

第二行n-1个数，第i个数pi表示pi和i之间有一条边

接下来q行，每行2个数，为询问的信息。格式见题意。

- 经过观察，我们发现修改难以实现，而询问非常简单。
- 一般来说，一个优雅的数据结构题，他的修改和询问的复杂度应该是同阶的。
- 所以我们考虑在询问上搞一点大新闻。
- 那么我们考虑修改对询问的影响，并尝试用数据结构维护修改的贡献。
- 先从简单条件入手，那么我们先不考虑2操作
- 很明显，如果点x的颜色是黑色，那么点x到根的路径必然存在一段后缀
- 满足这段后缀上的1操作数量和这段后缀长度相等。

- 那么我们将所有点的初始点权看作-1，然后一次1操作看做单点加1.
- 那么一个点为黑色的条件就是它到根这段路径的后缀最大值>=0。
- 所以直接用线段树维护即可。
- 我们再来考虑2操作，强制使得一个点x的子树为白色。
- 那么我们肯定是先把x的子树全部赋成-1，这样就清空了x子树内1操作的影响。
- 然后还要考虑x到根路径上的操作对其影响。

- 很明显，我们的目的是使得x到根路径的后缀最大值为-1。
- 所以我们只需要查询x到根路径的后缀最大值，然后进行单点修改就可以了。

CF639F BEAR AND CHEMISTRY

题意翻译

给定一张 n 个点 m 条边的初始无向图。

q 次询问，每次询问给定一个点集 V 和边集 E。

你需要判断，将 E 中的边加入初始无向图之后，V 中任意两个点 x, y 是否都能**在每条边至多经过一次**的情况下从 x 到 y 再回到 x。

$n, m, q, \sum |V|, \sum |E| \le 3 \times 10^5$，强制在线。

- 询问就是加入一个边集判一个点集是不是在边双联通分量里。
- 数据原来是个图，但不影响我们按照边双联通分量缩点。
- 先缩点，剩下的就是一片森林。
- 然后我们考虑加边。
- 因为询问之间互不影响，所以我们还得考虑回撤。
- 那加边删边就用LCT维护好了

- 如果我们加入的边链接了两棵树，直接LCT把这两棵树接起来就可以了。
- 否则就是我们加入的边就是连了一条路径。
- 根据边双，我们知道，这就相当于把这条路径变成了一个点。
- 我们可以直接把路径的边权都赋值成0，
- 然后维护一个点集是不是在同一个连通分量里
- 就只用对于相邻的每一个对数，查询这一对数路径上权值是否为0即可。

CF1270H NUMBER OF COMPONENTS

给一个长度为 n 的数组 a，a 中的元素两两不同。

对于每个数对 $(i,j)(i<j)$，若 $a_i<a_j$，则让 i 向 j 连一条边。求图中连通块个数。

支持 q 次修改数组某个位置的值，每次修改后输出图中连通块个数。

$n,q\le 5\times 10^5, 1\le a_i\le 10^6$，保证任意时刻数组中元素两两不同。

- 首先可以证明一个联通快肯定是一段连续的区间。
- 那么我们只需要证明，若i和j联通，那么对于i<k<j也和他们联通即可。
- 若a[i]<a[j],那么(a[i]<a[k])|(a[k]<a[j])一定=1。
- 那么我们从头开始，可以先根据上面的式子分成若干段。
- 比如从a[1]开始找到后面第一个比它小的数，这样中间就会被划分成联通的一段一段。
- 这样每段的最小值mi就是递减的，每一段就相当于一个区间[mi,mx].
- 然后每一段如果和另一段区间有交，就说明这两段在同一个连通块里。
- 这样合并完后发现，所有连通块仍然是一个连续段，且不可能再合并了。

那么问题就变成了求有多少p满足[1,p]和[p+1,n]之间没有边相连。

也就是$\min(1,p)<\max(p+1,n)$。

这个时候我们考虑设$\max(p+1,n)=v$,然后把$>v$的数看作1，$<=v$的数看作0

那么如果p合法，序列必然是111100000的形式，其中前缀有p个1。

那么我们发现每一个合法的v，唯一对应一个p。

所以我们考虑求出有多少合法的v，这里我们把a[0]看作正无穷，a[n+1]看作负无穷。

那么问题就是求有多少v对应的序列中只有一对相邻的0,1

- 那么对于位置i来说，他会使得[min(a[i],a[i+1]),max(a[i],a[i+1]))中的数，在i这个位置出现一对相邻的0和1，也就是一个区间加。
- 所以我们用线段树维护每一个v中相邻的个数，每次需要实现的就是区间加。
- 维护区间最小值和最小值的个数。
- V还需要满足在序列中出现过，这里特判一下即可。

CF704E IRON MAN

"铁人"yyb在玩游戏。在一个n个点的树上，yyb放置了m个鸡贼。每个鸡贼有四个整数参数t_i, c_i, v_i, u_i，表示这个鸡贼会在t_i时刻**出现**在点v_i，并以每时刻c_i条边的速度向u_i点**匀速**移动，到达u_i点时**立刻消失**。

如果一个时刻有两个鸡贼在同一位置，它们会立刻爆炸。

（注意，如果一个鸡贼的$u_i = v_i$那么它会在t_i时刻出现，此时如果这个点有其它鸡贼一样会发生爆炸）

yyb想知道最早有鸡贼爆炸的时刻。如果自始至终都没发生爆炸输出 `-1` 。

如果你的答案和标准答案的绝对或相对误差不超过10^{-6}那么被视为正确。

- 我们先对问题进行树链剖分，这样问题就从树变成了一条链。
- 然后在链上，会突然出现一个点和突然消失，如果把链看做x轴，把时间看做y轴，
- 那么这条链上出现的一个鸡贼就相当于一条线段。
- 然后两个鸡贼重合就相当于两条线段相交。
- 所以就是求y最小的交点。
- 最后的答案就是所有链的答案取min。

- 那么我们可以考虑扫描线。
- 按照时间顺序扫描，那么当第一次出现线段相交时，他们在之前的扫描线必然是相邻的。
- 所以我们只需要维护相邻线段的交点即可。
- 所以我们用一个set维护当前线段的横坐标，每次往里面插入一个元素，不断更新交点时间即可。