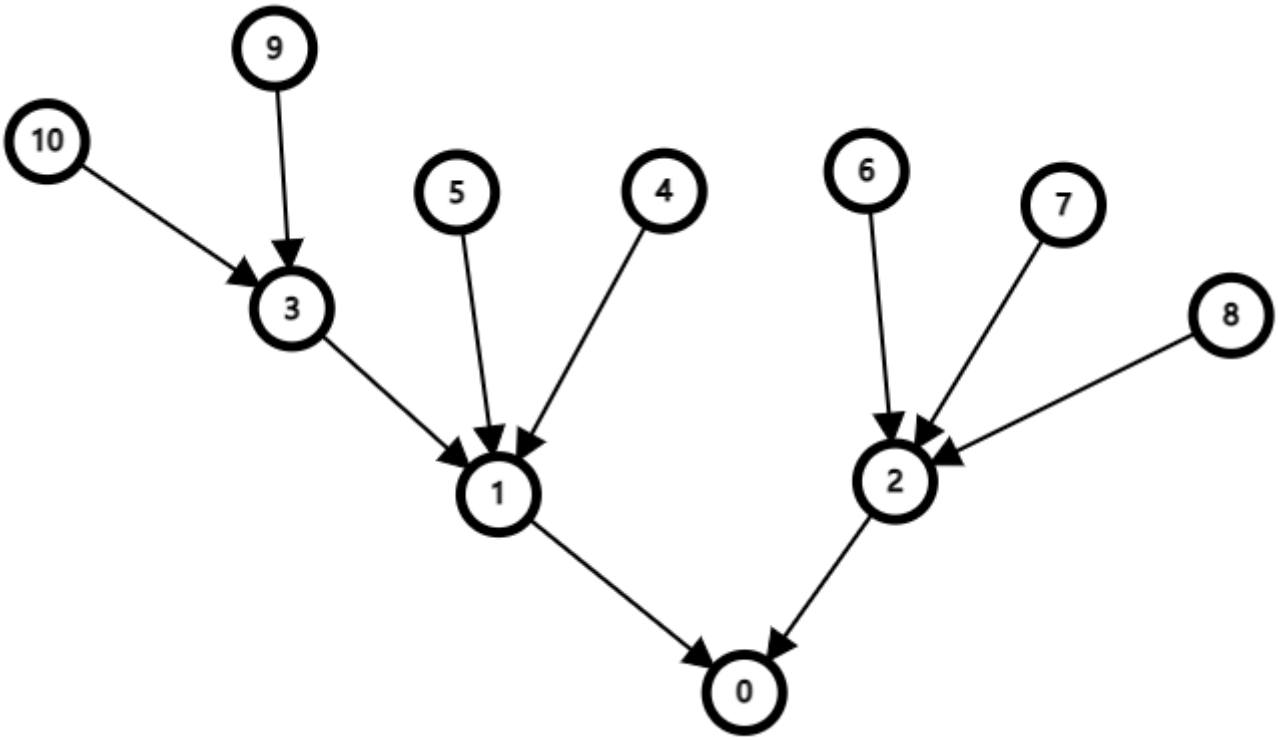
> 五一附近DDL比较多，加之五一期间队内赛、热身赛、校赛也占用了很多时间，题目还有很多没有卷完，有些遗憾😭

A-猫为什么讨厌狗是有理由的

- 对于区间 $[0,n]$ 中的所有整数，其通过除法操作的变换会形成一个树形结构
- （借用zzh的例子），假设 $n=10$，除数$d=3$。有向边 $a \to b$ 表示 a 通过一次除法变换得到 b



- 可以发现钥匙之间形成了一棵树形拓扑关系，把这棵树看成以 0 为根的有根树，子树的所有钥匙均可被子树根节点以及子树根节点的所有祖先完全替代

- 因此我们可以去掉一些无用的钥匙（能被其它钥匙完全替代），保留下最少有用的钥匙，这样之后每个糖罐将会唯一对应 1 把钥匙
- 由于钥匙不超过 60 把，用 bitset + ST表 维护区间内所有糖罐对应钥匙的并集即可
- 坑点：**1.** 2^{63} **用unsigned long long 2. 1ull !!!!!!!!!!**

B-啊哈哈哈，这收集糖果多是一件美事啊

- 这是一道典型的线段树多标记问题，如何同时维护好加法标记 add 和乘法标记 mul 是解决问题的关键
- 如果初次做本题，很可能会觉得不就是比加法线段树多一个乘法标记嘛，一样的写法就可以了
- 但实际写代码的时候就会发现，如果线段树上一个结点同时被打上了加法标记 add 和乘法标记 mul，它表示的含义究竟是 **sum * mul + add** 还是 **(sum + add) * mul** 呢
- 这就需要自己做一个规定了，不难发现表示成 sum * mul + add 会更好维护
- 因此，区间乘法：
  - sum = sum * k
  - mul = mul * k
  - add = add * k

- 区间加法：
  - sum = sum + (r - l + 1) * k
  - add = add + k
- 跑线段树即可

C-孤独的他，一个人数糖果

动态第 k 小

~~还不会带修主席树😭~~，就只好写整体二分了

这里的整体二分即是在值域上整体二分。将询问离线并安照时间线的顺序存储，修改操作a[x] = y可以看成在 t 时刻在序列中去掉x，再插入y，对于整体二分中当前二分的值midval = (lval + rval) / 2，把序列中 <= midval的数插入树状数组（即该数的位置+1），对于每个询问，统计区间中有多少个数，记为c，如果 <= k，则转化为在值域 [lval, mid]中寻找第 k 小，否则转化为在值域 [mid + 1, rval] 中寻找第 k - c 小。当区间询问数为空时返回。当区间 lval = rval 时，把区间所有询问的答案赋值为lval。时间复杂度为
$\mathcal{O}((N+M)\log_2 32768 \log n)$

D-辉夜大小姐希望完成她的愿望

平衡树模板题，网上讲得好的题解很多，这里就不赘述了

F-采果子

树链剖分 + 可持久化01trie

- 先考虑一个简单问题，给定一个长度为 n 的序列，有 m 次询问，每次询问给出一个数 x，在序列中任选一个数，求 x 与该数异或的最大值
- 这便是典型的两数异或和最大问题，把序列的所有数按二进制位从高到低插入01trie，对于 x ，在01trie上尽可能往相反方向走即可
- 换成区间询问的话，那就把01tire换成可持久化01trie即可
- 换成树上询问的话，那再套个树链剖分即可

G-谈笑风生

长链剖分优化DP，第一次接触长链剖分

首先这个题可以想到是一个DP。状态设计：$f_{u,dep}$ 表示 u 的子树中与 u 距离为 dep 的点的个数。 转移方程如下：

$$f_{u,dep} = \sum_{v \in son_u} f_{v,dep-1}$$

直接暴力转移显然是不行的，考虑优化

引入一个概念：长链剖分

我们可以找到一个结点 u 的子树中，到这个节点距离最大的叶子节点。将两点之间的距离计为 dep_u 。那么对于一个节点 u，我们把他的所有孩子中 dep 值最大的称作这个节点的长儿子。

从一个节点开始，每次走向他的长儿子，一直走到叶子节点为止，经过的路径就是一条长链。显然，一整棵树会被分成若干条长链，并且每个节点都在恰好一条长链上，每条边要么在长链上，要么把一条长链的顶端连向另一条长链。

我们可以采用一个优化策略：对于一个节点 u，我们先对它的长儿子做DP，但这里可以使用一些技巧，让长儿子把 dp 出来的东西直接存到 f_u 里面去（当然观察 dp 式可以发现这边需要错一位），然后再把其他儿子 dp 出来的东西与 f_u 暴力合并。

然后，我们只对每一个长链的顶端节点申请内存，而对于一条长链上的所有节点，我们让他们可以公用一片空间。具体地说，假设对节点 u 申请了内存之后，设 v 是 u 的长儿子，我们就把 f_u 数组的起点（的指针）加一当作 f_v 数组的起点（的指针，下同），以此类推。这也就是上面说的“让长儿子把 dp 出来的东西直接存到 f_u 里面去”。当然，申请的内存要能装下一条长链。

那么显而易见的，使用了这个优化之后可以把时间和空间都减到 $\mathcal{O}(n)$ 级别的，因为每个节点都只会在它所在的长链顶端被统计（或者说是被暴力合并）一次。

I-Namesolo 拜师

并查集板子题

当时在地铁上敲了一发交上去TLE，又立刻加上了按秩合并加上去还是TLE，就最后第17个点TLE，那没法了

后来数据好像锅了，重新交了一发A了

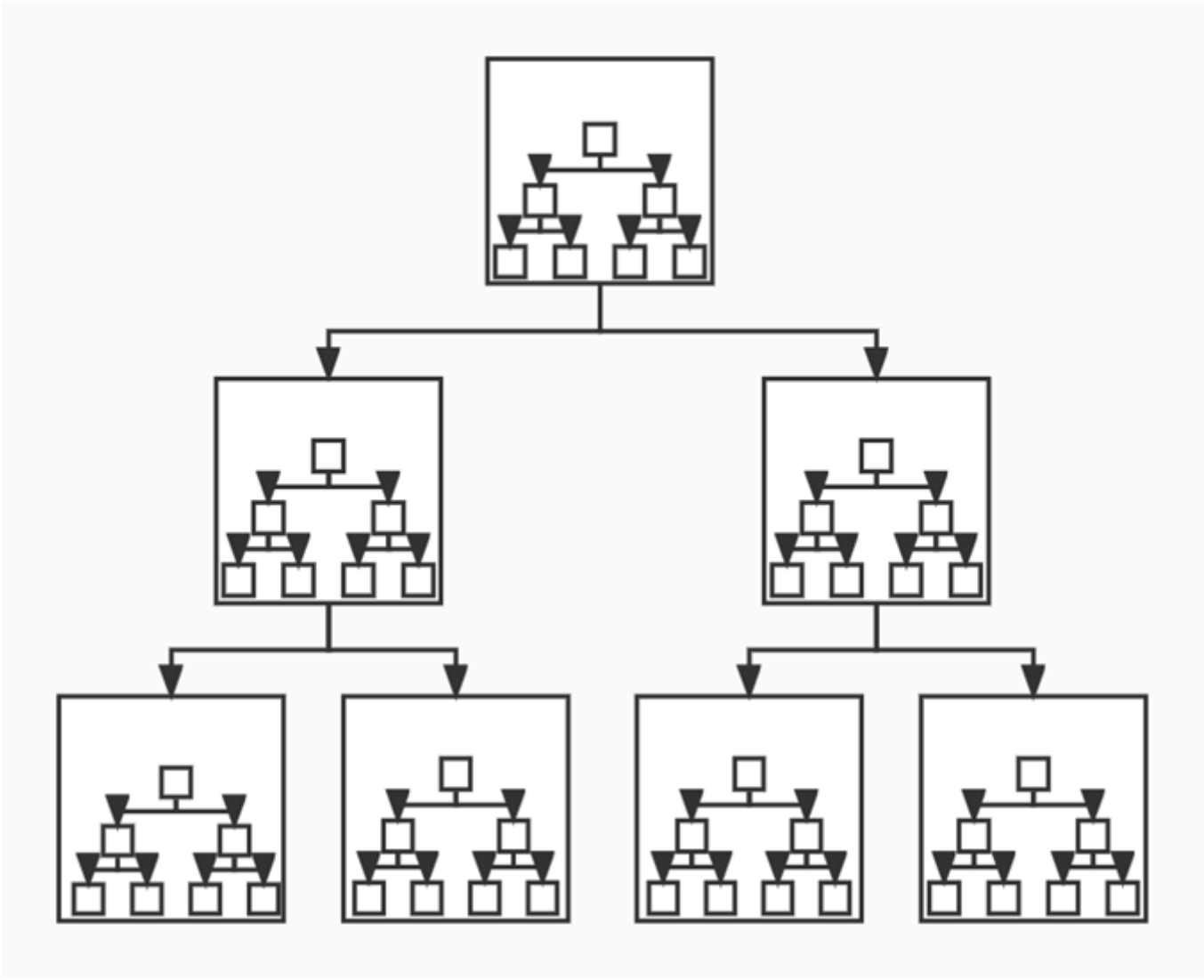
J-魔法商店

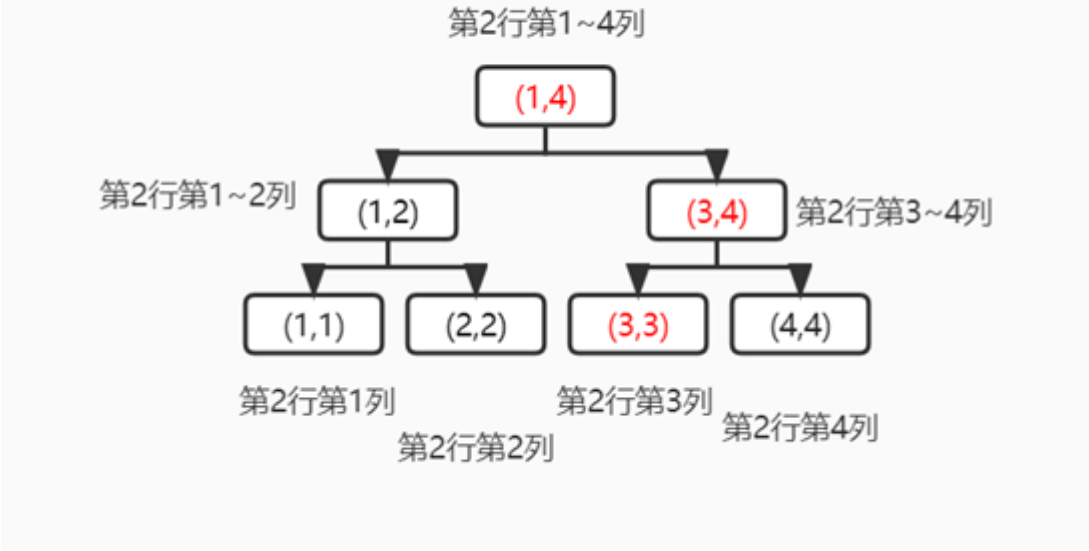
考虑按照商品价值从大到小枚举，对于每个商品找到当前还没用的最小的面值大于等于当前商品的纸币支付，如果没有满足条件的纸币，那么花费一次修改，得到一张等于商品价格的纸币，最后，如果修改次数没有用完，那么我们还可以在所有支付中修改差价最大的那几个

具体实现时可以使用`std::multiset<int>`来维护出当前剩余纸币面值的集合

K-云海蝴蝶螺

第一次学线段树套线段树，初学在网上找了很多blog，但没找到比较的图和详细代码解释，这里借用下出题人的图




第2行第1~4列
(1,4)
第2行第1~2列
(1,2)
(3,4)
第2行第3~4列
(1,1)
(2,2)
(3,3)
(4,4)
第2行第1列
第2行第2列
第2行第3列
第2行第4列

外层线段树维护行，内层线段树维护列，修改的时候注意如果外层线段树非叶子结点，那么内层线段树在更新叶子的时候需要通过“更小的内层线段树结点”Pushup，可以利用下标关系直接定位到：

```
if(l == r){
    if(is_leaf) Maxval[Rt][rt] = Minval[Rt][rt] = k;
    else{
        Maxval[Rt][rt] = max(Maxval[Rt << 1][rt], Maxval[Rt << 1 | 1][rt]);
        Minval[Rt][rt] = min(Minval[Rt << 1][rt], Minval[Rt << 1 | 1][rt]);
    }
    return;
}
```

第一次写线段树套线段树，没有用什么宏定义替换，~~一遍编译出解并AC了~~

L-GAMERS的众数

由于是Office编辑的公式，无法直接移植到Markdown，这里就直接搬自己的PPT了

L-GAMERS的众数

- Solution 1——分块（在线做法）
- 考虑把序列分成 T 块，则每块的长度 $L=\frac{N}{T}$，对于每个询问 $[l,r]$，把区间分成了3部分：
  1. 开头不足一整段的 $[l,L)$
  2. 第 $p+1$~$q-1$ 块构成的区间$[L,R]$
  3. 结尾不足一整段的 $(R,r]$
- 显然答案只来自于两种情况：区间$[L,R]$的众数，$[l,L)$ 和 $(R,r]$ 中间的数

L-GAMERS的众数

- Solution 1——分块（在线做法）
- 预处理出任意两个块的众数，暴力即可，复杂度是$O(NT)$
- 具体说来就是处理出 $f_{i,j}$ (第i块左端点到第j块右端点的最小众数)，和 $g_{i,j}$ (第i块左端点到第j块右端点的最小众数的出现次数)
- 另外对每个数值建立一个vector，保存该数值在序列中的出现位置
- 对于每个询问，扫描 $[l,L)$ 和 $(R,r]$ 中的每个数，在对应的vector里二分查找即可得到它在 $[l,r]$ 中的出现次数，与 $f_{i,j}$ 和 $g_{i,j}$ 的数值相比较，可以得到答案，每次询问的复杂度是 $O(L\log N)$

L-GAMERS的众数

- Solution 1——分块（在线做法）
- 这样总复杂度为 $O(NT + NL\log N) = O\left(NT + \frac{N^2}{T}\log N\right)$
- 根据均值不等式取等条件，令 $NT = \frac{N^2}{T}\log N$ ，解出 $T = \sqrt{N\log N}$
- 整个算法时间复杂度为 $O\left(N\sqrt{N\log N}\right)$

L-GAMERS的众数

- Solution 2——回滚莫队（离线做法）
- 莫队是啥？其实是运用了分块思想的暴力离线算法（不考虑在线莫队）
- 还是考虑把序列分成 T 块，则每块的长度 $L = \frac{N}{T}$ ，离线所有询问，对于每个询问 $[l,r]$ ，以左端点 l 所属块编号升序为第一关键字，右端点 r 升序为第二关键字的方式排序
- 依次处理排序后的每个询问，暴力从上一个区间的答案转移到下一个区间答案（一步一步移动即可）

L-GAMERS的众数

- Solution 2——回滚莫队（离线做法）
- 有些题目在区间转移时，可能会出现删除操作很困难或者无法实现的问题。在只有增加不可实现或者只有删除不可实现的时候，就可以使用回滚莫队在$O(N\sqrt{M})$内解决问题
- 回滚莫队的核心思想就是既然我只能实现一个操作，那么我就只使用一个操作，剩下的交给回滚解决。
- 比如本题，当莫队的区间扩大时，我们很容易知道计算出新加入的数是否可能更新众数答案，但当我们缩小区间时，我们不太容易计算小区间的众数

L-GAMERS的众数

- Solution 2——回滚莫队（离线做法）
- 如果询问左端点所属块 B 和上一个询问左端点所属块的不同，那么将莫队区间的左端点初始化为 B 的右端点+1，将莫队区间的右端点初始化为 B 的右端点
- 如果询问的左右端点所属的块相同，那么直接暴力扫描区间回答询问

L-GAMERS的众数

- Solution 2——回滚莫队（离线做法）
- 如果询问的左右端点所属的块不同：
  - 如果询问的右端点大于莫队区间的右端点，那么不断扩展右端点直至莫队区间的右端点等于询问的右端点；
  - 不断扩展莫队区间的左端点直至莫队区间的左端点等于询问的左端点；
  - 回答询问；
  - 撤销莫队区间左端点的改动，使莫队区间的左端点回滚到 B 的右端点+1。

L-GAMERS的众数

- Solution 2——回滚莫队（离线做法）
- 回顾普通莫队，对于左端点在同一个块内的询问，右端点升序排列，左端点本来是在块内无序乱跳，由于块的大小为 L ，每次左端点最多移动距离 L，反正最多都是移动 L ，不如每次我让莫队左端点都从块的右端点往左移动，这样转移询问就只扩大区间不缩小区间了
- 总复杂度为$O(ML+TN)=O\left(ML+\frac{N^2}{L}\right)$
- 取 $L=\frac{N}{\sqrt{M}}$ 最优，复杂度为$O\left(N\sqrt{M}\right)$

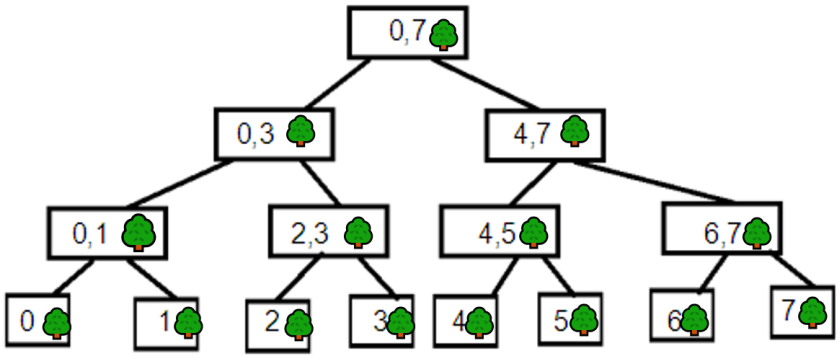
N-特雷森学院的训练员

由于是Office编辑的公式，无法直接移植到Markdown，这里就直接搬自己的PPT了

还没尝试过线段树套权值线段树，这里是用最经典的线段树套平衡树写的

N-特雷森学院的训练员

- Solution 树套树(线段树套平衡树)
- 所谓线段树套平衡树，即是线段树的每个结点都是一棵平衡树



N-特雷森学院的训练员

- Solution 树套树(线段树套平衡树)
- 所谓线段树套平衡树，即是线段树的每个结点都是一棵平衡树
- 关于树套树的构建，我们对于外层线段树正常建树，对于线段树上的某一个节点，建立一棵平衡树，包含该节点所覆盖的序列
- 操作1：查询 k 在区间的排名

按照线段树查询的方法，定位到线段树上的一些区间，再在每个区间的平衡树中查找小于 k 的数的个数，把所有查询区间的个数相加，再+1即是 k 在区间的排名，单次操作复杂度$O(\log^2 N)$

N-特雷森学院的训练员

- Solution 树套树(线段树套平衡树)
- 操作2：查询区间排名为 k 的值

在值域$[l,r]$内二分答案，将二分的值 mid 按照操作1的方法查询排名，若排名 $>k$，则令 $r=mid-1$; 若排名 $\leq k$，则令 $l=mid+1$ 。反复执行上述操作，直至 $l>r$ ，最后答案即是 r。

单次操作时间复杂度$O(\log^3 N)$

N-特雷森学院的训练员

- Solution 树套树(线段树套平衡树)
- 操作3：修改某一位数上的值

同理，在线段树上所有定位到被修改位置包含的区间，对于这些区间的平衡树，删去平衡树中原来的值，插入新值

单次操作时间复杂度$O(\log^2 N)$

- 总时间复杂度$O(N\log^2 N \sim N\log^3 N)$
- 对于每个元素，加入了$O(\log N)$个平衡树，

故总空间复杂度$O((N+M)\log N)$

Q-树上颜色段

考虑用线段树维护，线段树每个结点[l，r]维护3个信息，l、r端点的颜色和区间颜色段数量，合并也很简单，颜色段数直接相加，如果左孩子右端点的颜色和右孩子左端点的颜色相同，那么颜色段数量相加后需要-1。由于是在树上，再套个树链剖分即可，需要注意的是树链剖分查询多个区间结果合并的时候要注意区间合并的顺序

```
inline Node operator+(const Node&t1, const Node&t2){
    Node ret; ret.lc = t1.lc, ret.rc = t2.rc;
    ret.cnt = t1.cnt + t2.cnt;
    if(t1.rc == t2.lc) --ret.cnt;
    return ret;
}
```

实现的时候可以重载+运算符，可以在后续合并区间的时候方便许多

U-暴力题

考虑对异或序列求前缀异或和，根据异或性质，

$$sum_r \oplus sum_{l-1} = \bigoplus_{i=l}^{r} a_i$$

通过F题采果子我们知道，可持久话01trie已经能够解决区间两数异或和最大问题，由于本题强制在线，可以通过分块预处理出任意两个块的最大子段异或和，对于块的边角再在可持久化01trie中查询即可

不过本题看似好写，实则有好多坑点，debug了很久了😭：1.解密[l，r]时注意可能爆int 2. 询问查询边角时要查询l-1 和L[q]-1 3.中间块是 p+1 <= q-1 !!!

Z-卷卷人的计划表

当时在地铁上看到这道题，$M \leq 5000$，这范围没写错？？？，没搞懂这题想考啥，遂迅速码了个vector交上去就AC了

几天后数据范围更新了，vector被卡掉了，想到还可能上去讲题不能误导别人，赶紧重新补了个平衡树的做法交上去 ，~~后来发现不该我讲~~😅