替罪羊树

前置知识
讲解044-前缀树　整个专题的要求

有序表专题安排
专题1：AVL树，讲解148
专题2：跳表，讲解149
专题3：替罪羊树，讲解150，本节
专题4：笛卡尔树、Treap树，讲解151
专题5：FHQ Treap树，讲解152
专题6：Splay树，讲解153

大厂笔试、算法竞赛掌握以上有序表结构足够，其他有序表结构不再讲述，面试遇到只是聊，可以自行学习
算法竞赛的同学，有序表必带模版：替罪羊树、Treep树、FHQ Treap树、Splay树
Splay树是实现Link-Cut-Tree的关键，这个结构的讲述，会在【挺难】阶段的课程里安排

替罪羊树

替罪羊树的实现
实现一种结构，支持如下操作，要求单次调用的时间复杂度O(log n)
1，增加x，重复加入算多个词频
2，删除x，如果有多个，只删掉一个
3，查询x的排名，x的排名为，比x小的数的个数+1
4，查询数据中排名为x的数
5，查询x的前驱，x的前驱为，小于x的数中最大的数，不存在返回整数最小值
6，查询x的后继，x的后继为，大于x的数中最小的数，不存在返回整数最大值
所有操作的次数 <= 10^5
-10^7 <= x <= +10^7
测试链接 ： https://www.luogu.com.cn/problem/P3369

替罪羊树

替罪羊树理解难度低、代码容易写、常数时间好，实现常用功能的有序表非常推荐，但扩展性相对较弱

平衡因子为α，以h为头的树，节点总数为n，一旦左树或者右树的节点数 > α * n，就认为树不平衡
树不平衡时，中序收集树上所有节点，然后用二分的方式重构整棵树，使其变成标准的平衡树
查询代价由树高决定，重构代价由重构的节点总数决定
选择合适的平衡因子，一般为0.7，单次查询代价O(log n)、单次调整的均摊代价为O(log n)
课上展示实验代码

替罪羊树的优化，每次进行增加操作、删除操作后，只需找到最上方不平衡的位置，重构一次即可

平衡树上最好没有重复的key，所以如果同一个key需要收集多次，需要增加词频信息，count数组
为了判断平衡，需要记录树上收集了多少个不同的key，diff数组
为了实现查询，需要记录树上一共收集了多少个数字，size数组
课上详解替罪羊树增加、删除、查询的代码