

# Data Structures



By H10

# 前言



# 前言

- ~~小朋友们大家好，还记得我是谁吗？~~



# 前言

- ~~小朋友们大家好，还记得我是谁吗？~~
- 好的，今天我们来讲一讲 OI 中的重要考点、难点、易错点：数据结构。



# 前言

- 然而发现 NOIP 不怎么喜欢考数据结构啊呵呵。



# 前言

- 更准确的说 NOIP 喜欢考这个。



## 3. 斗地主

(landlords.cpp/c/pas)

### 【问题描述】

牛牛最近迷上了一种叫斗地主的扑克游戏。斗地主是一种使用黑桃、红心、梅花、方片的 A 到 K 加上大小王的共 54 张牌来进行的扑克牌游戏。在斗地主中，牌的大小关系根据牌的数码表示如下： $3 < 4 < 5 < 6 < 7 < 8 < 9 < 10 < J < Q < K < A < 2 < \text{小王} < \text{大王}$ ，而花色并不对牌的大小产生影响。每一局游戏中，一副手牌由  $n$  张牌组成。游戏者每次可以根据规定的牌型进行出牌，首先打光自己的手牌一方取得游戏的胜利。

现在，牛牛只想知道，对于自己的若干组手牌，分别最少需要多少次出牌可以将它们打光。请你帮他解决这个问题。

需要注意的是，本题中游戏者每次可以出牌的牌型与一般的斗地主相似而略有不同。具体规则如下：

牌型	牌型说明	牌型举例照片
火箭	即双王（双鬼牌）。	
炸弹	四张同点牌。如四个 A。	
单张牌	单张牌，比如 3。	
对子牌	两张码数相同的牌。	
三张牌	三张码数相同的牌。	

# 前言

- 也喜欢考这个。

NOIP2014 Day1.pdf 7:17 PM 140.39%

Thumbnails

2

3

4

5

## 3. 飞扬的小鸟 (bird.cpp/c/pas)

### 【问题描述】

Flappy Bird 是一款风靡一时的休闲手机游戏。玩家需要不断控制点击手机屏幕的频率来调节小鸟的飞行高度，让小鸟顺利通过画面右方的管道缝隙。如果小鸟一不小心碰到了水管或者掉在地上的话，便宣告失败。

为了简化问题，我们对游戏规则进行了简化和改编：

1. 游戏界面是一个长为  $n$ ，高为  $m$  的二维平面，其中有  $k$  个管道（忽略管道的宽度）。
2. 小鸟始终在游戏界面内移动。小鸟从游戏界面最左边任意整数高度位置出发，到达游戏界面最右边时，游戏完成。
3. 小鸟每个单位时间沿横坐标方向右移的距离为 1，竖直移动的距离由玩家控制。如果点击屏幕，小鸟就会上升一定高度  $X$ ，每个单位时间可以点击多次，效果叠加；如果不点击屏幕，小鸟就会下降一定高度  $Y$ 。小鸟位于横坐标方向不同位置时，上升的高度  $X$  和下降的高度  $Y$  可能互不相同。



# 前言

- 当然还有这个。

NOIP2013 Day2.pdf — 1

Index

- (请选手务必仔细...) 1
- 题目概况 1
- 5、特别提醒：... 1

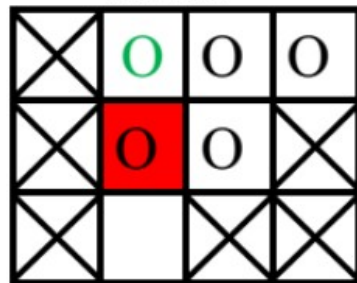
## 【输入输出样例说明】

棋盘上划叉的格子是固定的，红色格子是目标位置，圆圈表示棋子，其中绿色圆圈表示目标棋子。

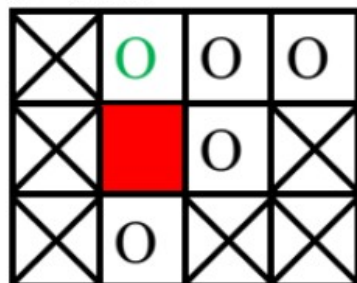
1. 第一次游戏，空白格子的初始位置是 (3, 2) (图中空白所示)，游戏的目的是将初始位置在 (1, 2) 上的棋子 (图中绿色圆圈所代表的棋子) 移动到目标位置 (2, 2) (图中红色的格子) 上。

移动过程如下：

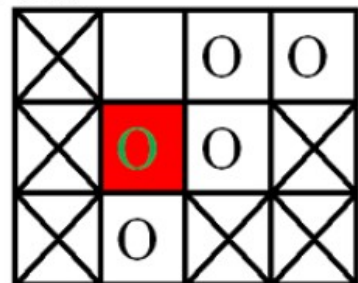
初始状态



第一步之后

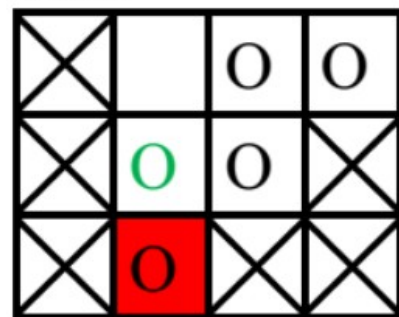


第二步之后



2. 第二次游戏，空白格子的初始位置是 (1, 2) (图中空白所示)，游戏的目的是将初始位置在 (2, 2) 上的棋子 (图中绿色圆圈所示) 移动到目标位置 (3, 2) 上。

初始状态





# 前言

- 没错还有这个 ~~图片左下没对齐，强迫症表示不爽~~

NOIP2011 Day1.pdf — NOIP2011senior\_day1.docx

7:18 PM 140.15%

【输入输出样例说明】

按箭头方向的顺序分别为图 6 到图 11



图 6



图 7



图 8

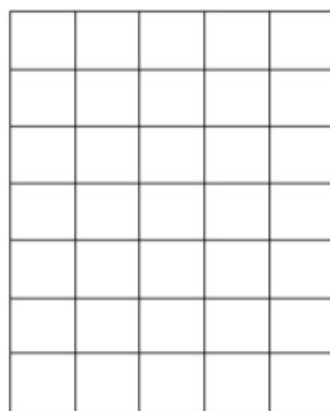


图 11

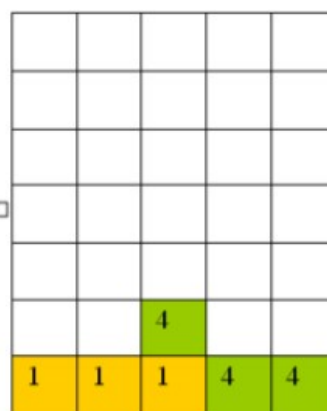


图 10



图 9

# 前言

- 总而言之 NOIP 最喜欢大型工业项目啊。
- 数据结构什么的好像被丢到后宫里去了。



# Part1 树状数组



# 树状数组



# 树状数组

- 定义实现什么的大家可自行背诵白书相关内容。



# 树状数组

- 定义实现什么的大家可自行背诵白书相关内容。
- 可以支持单点修改啊，区间求和啊什么的，都是  $O(n \log n)$ 。



# 树状数组

- 定义实现什么的大家可自行背诵白书相关内容。
- 可以支持单点修改啊，区间求和啊什么的，都是  $O(n \log n)$ 。
- 当然不怕麻烦的话你也可以在树状数组里实现 lazy-tag，然后就可以区间修改了。



# 树状数组

- 定义实现什么的大家可自行背诵白书相关内容。
- 可以支持单点修改啊，区间求和啊什么的，都是  $O(n \log n)$ 。
- 当然不怕麻烦的话你也可以在树状数组里实现 lazy-tag，然后就可以区间修改了。
- 基本上树状数组的功能线段树都能实现。





# 树状数组

- 那么还用它干嘛呢？



# 树状数组

- 那么还用它干嘛呢？
- 因为有一些不良心的出题人会出只能用它做的题。



# 树状数组

- 比如这道题 Bzoj 1452
- 一个  $n*m$  的方格，初始时每个格子有一个整数权值，接下来每次有 2 种操作：
  - 1) 改变一个格子的权值；
  - 2) 求一个子矩阵中某个特定权值出现次数。
- $n, m \leq 300$  ,  $Q \leq 200000$  ,  $1 \leq \text{权值} \leq 100$
- Time Limit : 1s
- Memory Limit : 64MB



# 树状数组

- 考虑用线段树，空间会是  $100 \cdot 4n \cdot 4m$  (549MB)



# 树状数组

- 考虑用线段树，空间会是  $100 * 4n * 4m$  (549MB)
- 用树状数组，空间  $100 * n * m$  (34MB)



# 树状数组

- 考虑用线段树，空间会是  $100 * 4n * 4m$  (549MB)
- 用树状数组，空间  $100 * n * m$  (34MB)
- 卡空间什么的最讨厌了。



# 树状数组

- 然后树状数组就讲完了。



# 树状数组

- 然后树状数组就讲完了。
- 什么，嫌讲的太少？





# 树状数组

- 然后树状数组就讲完了。
- 什么，嫌讲的太少？
- 那么就多加道题好了。



# 树状数组

- Poj 2481
- 给你  $n$  个闭区间，问每一个区间被多少个区间完全包含（完全重合不算）。
- $n \leq 100000$
- Time Limit : 1s
- Memory Limit : 256MB



# 树状数组

- 按  $L_i$  排个序，把  $R_i$  加入树状数组里面维护一下就好了对吧。经典的二维数点。



# 树状数组

- 按  $L_i$  排个序，把  $R_i$  加入树状数组里面维护一下就好了对吧。经典的二维数点。
- ~~这题就是来凑个数的。~~



# 树状数组

- 那么树状数组到这里就结束了吧。



# 树状数组

- 那么树状数组到这里就结束了吧。
- 才怪咧 .....



# 树状数组

- 那么树状数组到这里就结束了吧。
- 才怪咧 .....
- 我怎么可能只讲两道水题啊。



# 树状数组

- 那么树状数组到这里就结束了吧。
- 才怪咧 .....
- 我怎么可能只讲两道水题啊。
- 肯定要来一道非 (sang) 常 (xin) 有 (bing) 趣 (kuang) 的题啊。





# 树状数组

- Bzoj 4240
- 给你一个数列，相邻两个数之间可以交换，要求使每一个数要么左侧没有比它大的数，要么右侧没有比它大的数，求最少交换次数。
- $n \leq 300000$
- Time Limit : 1s
- Memory Limit : 256MB



# 树状数组

- 考虑对于每一个数，它的移动是不会影响到比它更大的数的。



# 树状数组

- 考虑对于每一个数，它的移动是不会影响到比它更大的数的。
- 因此我们可以从小到大移动数，并且贪心地移动到需要移动次数较少的一边。



# 树状数组

- 考虑对于每一个数，它的移动是不会影响到比它更大的数的。
- 因此我们可以从小到大移动数，并且贪心地移动到需要移动次数较少的一边。
- 于是每一个数对答案的贡献就是  $\min(\text{左侧比它大的数的个数}, \text{右侧比它大的数的个数})$ 。



# 树状数组

- 考虑对于每一个数，它的移动是不会影响到比它更大的数的。
- 因此我们可以从小到大移动数，并且贪心地移动到需要移动次数较少的一边。
- 于是每一个数对答案的贡献就是  $\min(\text{左侧比它大的数的个数}, \text{右侧比它大的数的个数})$ 。
- 那么可以从大到小添入节点然后用树状数组或线段树维护即可。



# 树状数组

- 总结：



# 树状数组

- 总结：
- 树状数组其实不算难，毕竟它支持的操作不多。



# 树状数组

- 总结：
- 树状数组其实不算难，毕竟它支持的操作不多。
- 单单只考树状数组的题基本是水题（第一题）；





# 树状数组

- 总结：
- 树状数组其实不算难，毕竟它支持的操作不多。
- 单单只考树状数组的题基本是水题（第一题）；
- 树状数组也可以解决一些经典问题（第二题）；



# 树状数组

- 总结：
- 树状数组其实不算难，毕竟它支持的操作不多。
- 单单只考树状数组的题基本是水题（第一题）；
- 树状数组也可以解决一些经典问题（第二题）；
- OI 中数据结构常常会与其它考点放一起考，但多半情况下数据结构是用于辅助（如第三题的贪心 + 树状数组或线段树）。



# Part2 RMQ



# RMQ



# RMQ

- 用  $O(n \log n)$  时间预处理，以后每次查询区间极值都是  $O(1)$ 。



# RMQ

- 用  $O(n \log n)$  时间预处理，以后每次查询区间极值都是  $O(1)$ 。
- 可以求区间最大最小值，还可求 LCA。



# RMQ

- 用  $O(n \log n)$  时间预处理，以后每次查询区间极值都是  $O(1)$ 。
- 可以求区间最大最小值，还可求 LCA。
- 用处基本就这么多了，还是看看在题目中怎么灵 (xia) 活 (ji) 地 (ba) 使用吧。



# RMQ

- Poj 3264
- 给你一个序列，每次询问第 L 个至第 R 个数之间的极差（最大值 - 最小值）





# RMQ

- Poj 3264
- 给你一个序列，每次询问第 L 个至第 R 个数之间的极差（最大值 - 最小值）

- ~~你 T1M 放模板题是在鄙视我的智商吗？~~



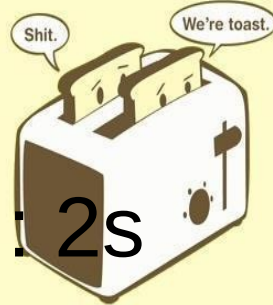
# RMQ

- 抱歉刚刚搞错题了啊，应该是这道。



# RMQ

- Poj 1785
- 给你  $N$  个结点建立一颗树，每个结点有两个关键字，要求第一个满足 BST 性质，第二个满足堆的性质。
- $N \leq 50000$
- Time Limit : 2s
- Memory Limit : 32MB



# RMQ

- ~~不就是 treap 啊~~



# RMQ

- 不就是 treap 啊
- 既然讲 RMQ ，就用 RMQ 做吧。



# RMQ

- 其实也就是道傻逼题了。



# RMQ

- 其实也就是道傻逼题了。
- Treap 做法：把点按第一关键字排序，按顺序插入右子树，不满足堆的性质就左旋。



# RMQ

- 其实也就是道傻逼题了。
- Treap 做法：把点按第一关键字排序，按顺序插入右子树，不满足堆的性质就左旋。
- RMQ 做法：还是首先要排序，每次递归处理一个区间，找出第二关键字最大的做根。其左边的就是左子树，右边的是右子树。





# RMQ

- 题目难度够良心了吧。



# RMQ

- 题目难度够良心了吧。
- 下面来一道有意思点的题。



# RMQ

- Nkoj 1752
- 给你一个有序数列，对于每一个询问  $Q(i,j)$ ，求出  $[i,j]$  中出现次数最多的数的出现次数。
- $N \leq 200000$ ， $Q \leq 200000$ ， $A_i \leq 1000000$
- Time Limit : 1s
- Memory Limit : 256MB



# RMQ

- ~~我会莫队~~



# RMQ

- ~~我会莫队~~
- 既然讲 RMQ ，就用 RMQ 做吧。（怎么感觉这句话讲过）



# RMQ

- 举个例子
- 序列 -1 -1 1 1 1 1 3 10 10 10
- 可以转换为 2 4 1 3 ，利用前缀和二分查找位置，左右两个端点暴力，中间 RMQ。



# RMQ

- 举个例子
- 序列 -1 -1 1 1 1 1 3 10 10 10
- 可以转换为 2 4 1 3 ，利用前缀和二分查找位置，左右两个端点暴力，中间 RMQ。
- ~~怎么尽是一些水题啊~~



# $\pm 1$ RMQ





# $\pm 1$ RMQ

- 下面来讲黑科技



# $\pm 1$ RMQ

- 下面来讲黑科技
- 只用  $O(n)$  预处理就能  $O(1)$  查询的  $\pm 1$  RMQ



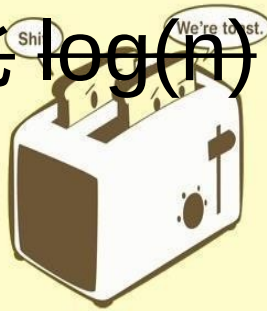
# $\pm 1$ RMQ

- 下面来讲黑科技
- 只用  $O(n)$  预处理就能  $O(1)$  查询的  $\pm 1$  RMQ
- ~~又不是 NOIP 考点~~



# $\pm 1$ RMQ

- 下面来讲黑科技
- 只用  $O(n)$  预处理就能  $O(1)$  查询的  $\pm 1$  RMQ
- ~~又不是 NOIP 考点~~
- ~~常数都能比  $\log(n)$  大~~



# $\pm 1$ RMQ

- 先分块，每一块的大小设为  $sz = \log(n)/2$ ，则一共有  $l = n/sz$  块



# $\pm 1$ RMQ

- 先分块，每一块的大小设为  $sz = \log(n)/2$ ，则一共有  $l = n/sz$  块
- 把每一块当成一个数字，做普通 RMQ，时间为  $O(l * \log(l)) = O(n/\log(n) * (\log(n) - \log(sz))) = O(n)$



# $\pm 1$ RMQ

- 先分块，每一块的大小设为  $sz = \log(n)/2$ ，则一共有  $l = n/sz$  块
- 把每一块当成一个数字，做普通 RMQ，时间为  $O(l * \log(l)) = O(n/\log(n) * (\log(n) - \log(sz))) = O(n)$
- 连续的几块的查询就与普通 RMQ 一毛一样了



# $\pm 1$ RMQ

- 先分块，每一块的大小设为  $sz = \log(n)/2$ ，则一共有  $l = n/sz$  块
- 把每一块当成一个数字，做普通 RMQ，时间为  $O(l * \log(l)) = O(n/\log(n) * (\log(n) - \log(sz))) = O(n)$
- 连续的几块的查询就与普通 RMQ 一毛一样了
- 重点是块内查询





# $\pm 1$ RMQ

- 考虑到块的大小为  $\log(n)/2$  , 每一个数与上一个数的差只能是 1 或 -1 , 那么本质不同的块就只有  $2^{(\log(n)/2)} = \text{sqrt}(n)$  个了。



# $\pm 1$ RMQ

- 考虑到块的大小为  $\log(n)/2$  , 每一个数与上一个数的差只能是 1 或 -1 , 那么本质不同的块就只有  $2^{(\log(n)/2)} = \text{sqrt}(n)$  个了。
- 对于一种种块 , 预处理出所有有可能的块内查询 , 都只有  $2^{(\log(n)/2)} = \text{sqrt}(n)$  种 , 所以块内查询的时间也为  $O(\text{sqrt}(n) * \text{sqrt}(n)) = O(n)$  。



# $\pm 1$ RMQ

- 讲完了，想来道题吗？



# $\pm 1$ RMQ

- 讲完了，想来道题吗？
- 好消息是我并没有找到专门用  $\pm 1$  RMQ 做的题。



# RMQ

- 总结：



# RMQ

- 总结：
- RMQ 其实也就是一个求极值的小工具。



# RMQ

- 总结：
- RMQ 其实也就是一个求极值的小工具。
- 同树状数组一样，没有太多的变形，用起来十分方便。



# RMQ

- 总结：
- RMQ 其实也就是一个求极值的小工具。
- 同树状数组一样，没有太多的变形，用起来十分方便。
- 但是部分 RMQ 题会伪装得很好，难以短时间内看出来是要用 RMQ 来做，具体的例子会在练习题中出现。（这里是在剧透哦）





# Part3 线段树



# 线段树



# 线段树

- 重要的内容总要用于最后的压轴对不对。



# 线段树

- 重要的内容总要用于最后的压轴对不对。
- 线段树，OI 中最常用的数据结构之一，它能支持单点修改，区间修改，区间查询，时间复杂度都是  $\log n$ 。



# 线段树

- 然而线段树最有趣的一点是：



# 线段树

- 然而线段树最有意思的一点是：
- 凡是树状数组与 RMQ 能做的，线段树都能做；  
凡是支持合并的元素，线段树都能维护。



# 线段树

- 然而线段树最有意思的一点是：
- 凡是树状数组与 RMQ 能做的，线段树都能做；凡是支持合并的元素，线段树都能维护。
- ~~凡是毛主席做出的决策，我们都坚决维护；凡是毛主席的指示，我们都矢志不渝地遵循。~~



# 线段树

- 第一个凡是不必多说。






# 线段树

- 第一个凡是不必多说。
- 关与第二个凡是，请参考 NOI2016 湖南省省队集训 Day9T2 闷声刷大题暨雅礼中学 2016 湖南省省选集训 Day2T2 光盘。



# 线段树

- 第一个凡是不必多说。
- 关与第二个凡是，请参考 NOI2016 湖南省省队集训 Day9T2 闷声刷大题暨雅礼中学 2016 湖南省省选集训 Day2T2 光盘。
- “光盘”的出题人曾评论过“闷声刷大题”：出这道题的人心态很不好啊，因为我出“光盘”时心态就  
不好。

# 线段树

- 废话就讲到这里吧，下面上题。



# 线段树

- Poj 3667
- 旅馆里有一列连续的房间，要求支持两种操作：
- 1 a : 来了 a 名客人，询问是不是有连续长度为 a 的空房间，有的话住进最左边；
- 2 l r : 将 [l,r] 的房间清空。
- $1 \leq n, m \leq 50000$
- Time Limit : 1s
- Memory Limit : 256MB



# 线段树

- 热身傻逼题 ...



# 线段树

- 热身傻逼题 ...
- 线段树记录区间中最长的空房间；



# 线段树

- 热身傻逼题 ...
- 线段树记录区间中最长的空房间；
- 查询一个区间时，先看左子树是否合法，不合法就再看左子树靠右 + 右子树靠左是否合法，还不合法就看右子树。



# 线段树

- 下一题





# 线段树

- Bzoj 2957
- 有  $n$  栋楼房在一个二维平面上。第  $i$  栋楼房可以用线段  $(i,0)-(i,H_i)$  表示。最开始楼房的高度全为 0， $m$  次操作，每次把第  $x_i$  栋楼房的高度修改为  $y_i$ ，并询问在  $(0,0)$  处可见楼房数量。
- $1 \leq x_i, n, m \leq 100000$ ， $1 \leq y_i \leq 10^9$
- Time Limit : 1s
- Memory Limit : 256MB



# 线段树

- 显而易见，可见的数字是单调递增的。修改一个数只会对后面的数造成影响。那么考虑线段树中的每一条线段。只有两种情况：



# 线段树

- 显而易见，可见的数字是单调递增的。修改一个数只会对后面的数造成影响。那么考虑线段树中的每一条线段。只有两种情况：
- 1、最大值小于等于修改的数，那么这个线段的贡献为 0，无需处理；



# 线段树

- 显而易见，可见的数字是单调递增的。修改一个数只会对后面的数造成影响。那么考虑线段树中的每一条线段。只有两种情况：
- 1、最大值小于等于修改的数，那么这个线段的贡献为 0，无需处理；
- 2、否则将这个线段分成两段，如果左侧的最大值大于修改的数，那么不影响右侧贡献，递归处理左侧；否则就变成了第一种情况，递归左侧。



# 线段树

- 下一题



# 线段树

- 某学校某年某日某比赛的某模拟题
- 如果字符串 A 是字符串 B 的后缀，那么称 B 是 A 的 XQ 串。给你 n 个只包含小写字母的字符串，编号为 1-n。对于每个串，求其所有的 XQ 串的编号集合（包括自己）中第  $K_i$  小的编号。
- $n \leq 100000$  ,  $\sum |S| \leq 300000$
- Time Limit : 1s
- Memory Limit : 256MB



# 线段树

- 怎么开始讲字符串题了啊？



# 线段树

- 怎么开始讲字符串题了啊？
- 把所有的串反过来建个 trie ，那么某个串的结束结点的子树里的串就是所有该串的 XQ 串。





# 线段树

- 怎么开始讲字符串题了啊？
- 把所有的串反过来建个 trie ，那么某个串的结束结点的子树里的串就是所有该串的 XQ 串。
- 以 dfs 序排序，问题就变成求区间第 K 小了。



# 线段树

- 怎么开始讲字符串题了啊？
- 把所有的串反过来建个 trie ，那么某个串的结束结点的子树里的串就是所有该串的 XQ 串。
- 以 dfs 序排序，问题就变成求区间第 K 小了。
- 用主席树维护维护即可。



# 线段树

怎么开始讲字符串题了啊？

把所有的串反过来建个 trie ，那么某个串的结束结点的子树里的串就是所有该串的 XQ 串。

以 dfs 序排序，问题就变成求区间第 K 小了。

用主席树维护维护即可。



# 线段树

怎么开始讲字符串题了啊？

把所有的串反过来建个 trie ，那么某个串的结束结点的子树里的串就是所有该串的 XQ 串。

以 dfs 序排序，问题就变成求区间第 K 小了。

用主席树维护维护即可。

- 不是说好了的 NOIP 考点么？



# 线段树

怎么开始讲字符串题了啊？

把所有的串反过来建个 trie，那么某个串的结束结点的子树里的串就是所有该串的 XQ 串。

以 dfs 序排序，问题就变成求区间第 K 小了。

用主席树维护维护即可。



- 不是说好了的 NOIP 考点么？
- 虚假宣传啊！

# 线段树

- 来个简单点的方法吧。



# 线段树

- 来个简单点的方法吧。
- 由于可以离线，区间第  $K$  小可以直接在 trie 上自底向上查询。用线段树合并维护结点信息即可。



# 线段树

来个简单点的方法吧。

由于可以离线，区间第  $K$  小可以直接在 trie 上自底向上查询。用**线段树合并**维护结点信息即可。



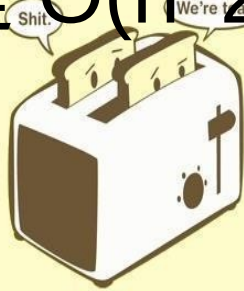


# 线段树

来个简单点的方法吧。

由于可以离线，区间第 K 小可以直接在 trie 上自底向上查询。用线段树合并维护结点信息即可。

- 你确定不是  $O(n^2)$  ？



# 线段树

- 确定，你们可以先看看线段树合并的代码。



# 线段树

- 确定，你们可以先看看线段树合并的代码。
- ```
int merge(int x,int y){
```
- ```
    if (!x) return y;
```
- ```
    if (!y) return x;
```
- ```
    ls[x]=merge(ls[x],ls[y]);
```
- ```
    rs[x]=merge(rs[x],rs[y]);
```
- ```
    return x;
```
- ```
}
```



# 线段树

- 可以看出线段树合并的复杂度与两颗树重复的部分大小成正比。



# 线段树

- 可以看出线段树合并的复杂度与两颗树重复的部分大小成正比。
- 每个结点记录的信息只是这个结点的子树里所有的串，当我们合并两个结点时，它们包含的串是没有交集的。



# 线段树

- 可以看出线段树合并的复杂度与两颗树重复的部分大小成正比。
- 每个结点记录的信息只是这个结点的子树里所有的串，当我们合并两个结点时，它们包含的串是没有交集的。
- 由于总共只有  $n$  个串，时间就是  $O(n \log n)$  了。



# 线段树

- 终于讲完了。



# 线段树

- 终于讲完了。
- 额，好像还没有总结。





# 线段树

- 总结：



# 线段树

- 总结：
- 比起前两种数据结构，线段树有更强的灵活性，这也就是为什么 OI 常考线段树的主要原因。

~~(句式杂糅)~~



# 线段树

- 总结：
- 比起前两种数据结构，线段树有更强的灵活性，这也就是为什么 OI 常考线段树的主要原因。  
~~（句式杂糅）~~
- 正是由于其灵活性，所以能熟练地使用线段树也是作为一名 Oler 必须要 get 的技能。  
~~（介词掩盖主语）~~



# 练习题



# 练习题

- Poj 2482
- 在一个平面内有  $n$  个整点，每个点都有一个权值，用一个  $w \times h$  的矩形去围这些点（边上的不算），求能得到的最大权值和。
- $n \leq 100000$ ,  $w, h, |x_i|, |y_i| \leq 2^{31}$
- Time Limit : 1s
- Memory Limit : 256MB



# 练习题

- Poj 1981
- 在一个平面内有  $n$  个点，求用一个单位圆最多能围住多少点。
- $n \leq 300$  ,  $|x_i|, |y_i| \leq 10.0$
- Time Limit : 1s
- Memory Limit : 256MB



# 练习题

- 某 Acme 趣题
- 定义完美序列：连续的互不相同的序列，给你一个序列，对于每次询问，回答区间  $[L,R]$  之间最长的完美序列的长度。
- $n, m \leq 200000$   $a_i \leq 1000000$
- Time Limit : 1s
- Memory Limit : 256MB



# 练习题

- Bzoj 3339
- 给你一个序列，对于每一次询问  $Q(l,r)$ ，请回答  $\text{mex}\{a[l], a[l+1], \dots, a[r-1], a[r]\}$  的值。
- $n, m \leq 200000$     $a_i \leq 1000000$
- Time Limit : 1s
- Memory Limit : 256MB





# 谢谢大家

