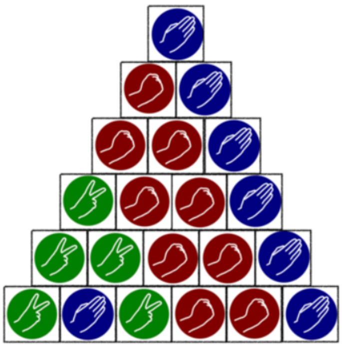
土块

> 你想通过石头剪刀布来赢回丢失的内存条。
>
> 有一个长为 n 的字符串，每个位置可以是 R,P,S 三种字符之一，分别代表石头、布、剪刀。
>
> 我们进行若干轮操作，每轮操作中字符串的第 i 位变为原来字符串中第 i 位和第 $i+1$ 位按照石头剪刀布规则比赛后的获胜方。如果平局则获胜方任意。当字符串长度变为 1 时不再操作。具体见下图。
>
> 
>
> 你需要对于给定的初始字符串 s 维护以下两种操作：
>
> `1 x c`：将字符串的位置 x 改成字符 c。
>
> `2 l r`：求出对 $s_{l,r}$ 做上述操作后最后剩下的字符。

序列显然有结合律，考虑需要保留哪些信息来结合。我们只要保留 `2 1 0 2 1 0 1 2 0 1 2` 这样的序列，也就是区间赢家，赢家左边的一路赢的个数，右边一路赢的个数。因为一个点，它左右都输，它就一点用都没有，基于这点就可以快速结合了。$O(n\log n)$。

空间旅行

> 给定长为 n 的字符串 s 和长为 m 的字符串 t，求 s 在 t 中的出现次数。
>
> 你的空间限制去旅行了，只留下了 1MB。你的大样例觉得旅行很有意思，也跟着空间跑路了。
>
> 为了防止奇怪的操作，本题提供`travel.h`用于读入。你可以调用`readint()`和`readchar()`。你**只需要**实现返回值为`int`的函数`solve()`用于返回答案。
>
> $n,m\le 10^7$。

对序列 s 分块，设块长为 B。那么对于每个块求出一个哈希值，对于这个长度为 $\frac{S}{B}$ 的序列求一个 kmp。

那么对于 t 序列，每个 B 的剩余系，我们都相当于是要做一个在 $\frac{S}{B}$ 的序列上的 kmp，我们分别维护它在 kmp 上匹配到的位置。空间复杂度 $O\left(n^{0.5}\right)$。

邀请

> 有 n 个人，编号 1 到 n。
>
> 你要带着其中的一些人去找回你丢失的内存条，然而人们对这件事不是很感兴趣。他们只在意别人有没有去。
>
> 具体地，初始没有人愿意跟你去。每次你可以向一个人发出邀请。设这个人的编号是 i。
>
> 如果此时编号在 $[l_i,r_i]$ 中愿意跟你去的人数量不小于 k_i，则这个人愿意跟你去。
>
> 求最多有多少人愿意跟你去。
>
> $n\le 4\times 10^5$。

这题能有优于 KDT 的做法，主要是因为这个 k 是只减的，这个性质很强，$\log k$ 这个弱多项式，虽然理论大于根号，但是实际上恨优；并且很多看着只能根号的题，实际上可以把根号换成弱多项式也很合理。

我们考虑猫树分治，跨过 mid 的两段区间怎么将其独立？左边先分 $\frac{k}{2}$，右边分 $\frac{k}{2}$，如果哪边用完了，再重新分配。而某一边的问题，就是一个普通的一维偏序。

CERC 13 History course

> 给定 n 个区间，第 i 个区间为 $[a_i, b_i]$。你需要把这些区间按某种顺序排列，使得两个区间如果没有交点，则左端点更小的区间需要排在前面。
>
> 设第 i 个区间排在第 p_i 个位置，定义区间 i 与 j 之间的距离为 $|p_i - p_j|$。令 k 为任意两个相交的区间之间的最大距离，你需要最小化 k，并输出一组对应的合法顺序。
>
> 对于 100% 的数据，满足 $1 \le n \le 5 \times 10^4$，$0 \le |a_i|, |b_i| \le 10^9$。

当你性质太少了，觉得没法做，而且长时间没有进展的时候，不妨**直接对着可能的做法想想**，反而可能促进你找到一些依赖于的性质，**特别是这个性质和做法本身高度结合的时候**。

先二分，然后我们从左到右填区间。先把所有区间按照右端点排序。对于没选的区间，有一个它最晚的时间 lim_i，因为它可能已经和某个已选线段相交，它的时间不能不能太晚。我们考虑 $S_i = S_{i-1} - \sum [lim_j = i] + 1$ 数组，全大于 0 才一定有解，我们找不会影响此数组性质的右端点最靠左的区间更优！（因为它对 lim 数组的影响最小）。

一个点对 lim 数组的影响，相当于把最靠右的右端点的指针向右推了一点。这也体现了我们为什么要选右端点最靠左的。它会使得一个区间里的点从没有 lim 变为有 lim，也就是 S 数组的一个后缀减法。

对于 $\sum [lim_j = i] > 1$：i 位置处无解了；$= 1$，这个位置已经确定了；$= 0$，我们看看后面有没有 $S = 0$，如果有，那就取那个范围内的区间中右端点最靠左的；否则，乱选一个右端点最靠左的。

CTT 19 匹配问题

> 数轴上有 n 个黑点，坐标为 $a[]$；n 个白点，坐标为 $b[]$（下标均从 0 开始）。同时你有两个参数 l_a 和 l_b，满足 $n \ge 2$ 且 $1 \le l_b < l_a$。注意同一个位置可能有多个点。
>
> 你需要找到一个 0 到 $n-1$ 的排列 $p[]$，满足对于所有 i，有 $a[i] \le b[p[i]] \le a[i] + l_a$。你需要最大化满足 $b[p[i]] \le a[i] + l_b$ 的下标 i 的个数。数据保证存在一个合法的排列。
>
> $n \le 10^5$。

$b \in [a, a + la]$ 的对，和 $b \in [a, a + lb]$ 的对，肯定是彼此独立的，分别是左部点、右部点分别排序，彼此的差不能超过 la 或者 lb。

我们从右到左贪心选择 $b \in [a, a + lb]$ 的对。原本是 $a_i \to b_i$，我们要一个 $a_i \to b_k$，使得这个区间内的匹配都位移一位，不会出事。从 a_i 从大到小考虑，肯定找可以范围内最靠右的 b，暴力 check 置换后会不会有问题，get $O(n^2)$。

###