



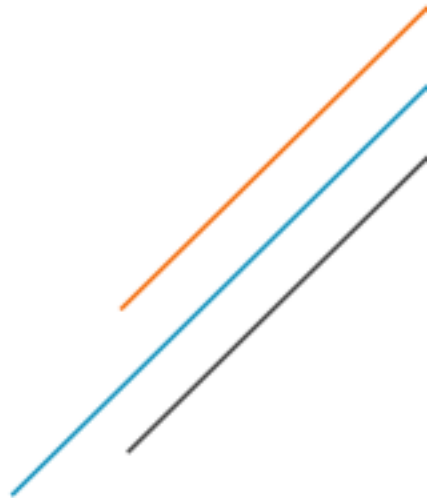
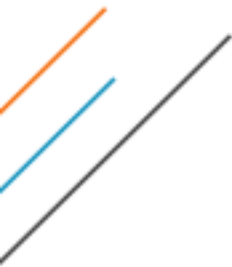
2019牛客暑期多校训练营（第七场）

kuangbin



J - A+B problem

- 签到题





A - String

- 给一个01构成的字符串，要把该字符串切分成最少的份数，使得每一个字符串都是循环移位字典序最小的字符串。
- 111011110 -> 111 01111 0

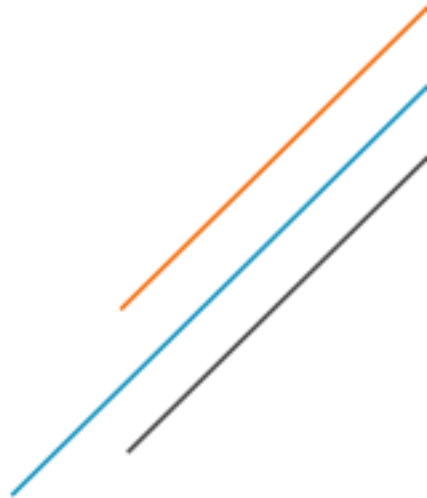
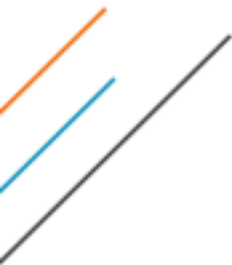
A - String

- 签到题。
- 每次暴力枚举最少的字符串，暴力判断即可。
- 或者把0000001111111这种分段，判断相邻两段能不能合并。



B - Irreducible Polynomial

- 给一个n次多项式，判断该多项式能不能在实数域拆分





B - Irreducible Polynomial

- 签到题
- 实数域不可拆分多项式只有两种：一次多项式和二次的($b^2 < 4ac$)



C - Governing sand

- n 中树，第 i 种树有 $P[i]$ 颗，砍掉每颗树的代价是 $C[i]$ ，高度是 $H[i]$ 。
- 需要用最小的代码砍掉一些树，让最高的树超过一半。



C - Governing sand

- n 中树，第 i 种树有 $P[i]$ 颗，砍掉每颗树的代价是 $C[i]$ ，高度是 $H[i]$ 。
- 需要用最小的代码砍掉一些树，让最高的树超过一半。
- 从低到高枚举最高的树，在剩下的树当中砍掉代价最小的。可以用一颗线段树维护。
- 线段树按照代价从小到大的顺序构建，支持加入类树，查询前 x 个的总和。
- <https://paste.ubuntu.com/p/tjdtV9ymnM/>



D - Number

- 给了 n 和 p , 需要输出一个数, 刚好有 n 位数而且整除 p .



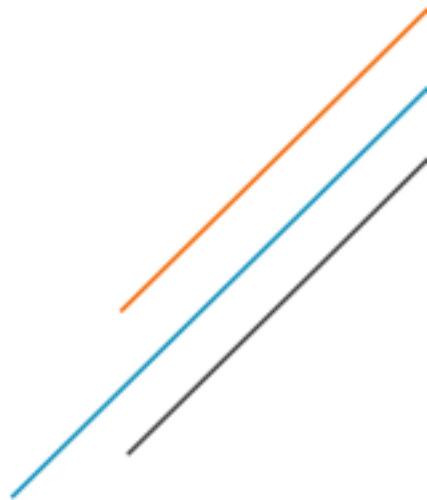
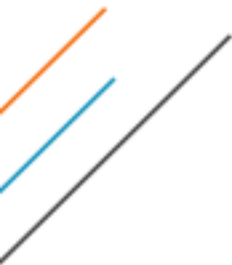
D - Number

- 给了 n 和 p , 需要输出一个数, 刚好有 n 位数而且整除 p .
- 签到题。
- If $\text{len}(p) > n$: 无解
- If $\text{len}(p) = n$: 输出 p
- If $\text{len}(p) < n$: 先输出 p , 然后补上0



H - Pair

- 给定A, B, C, 需要求有多少个pair<x,y> 满足
- $x \& y > C$ or $x \wedge y < C$





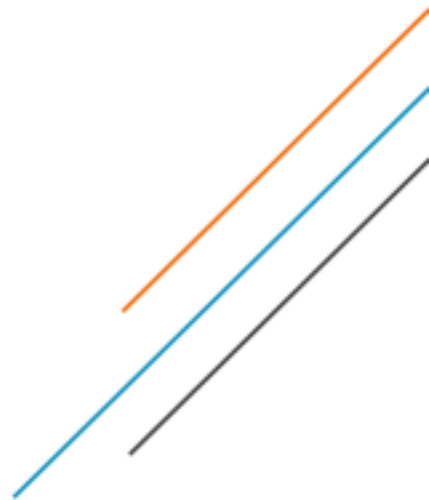
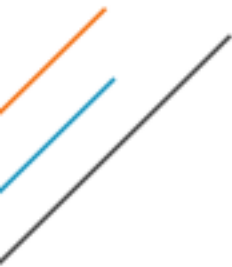
H - Pair

- 给定A, B, C, 需要求有多少个pair<x,y> 满足
- $x \& y > C$ or $x \wedge y < C$
- 直接数位DP一下求解



E -Find the median

- 每次插入区间 $[L_i, R_i]$ 之间的数，查询中位数





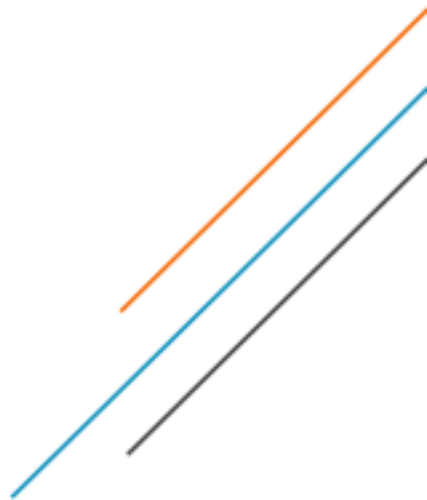
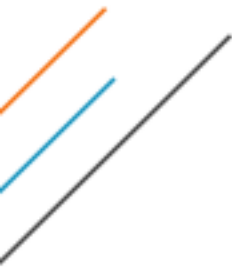
E -Find the median

- 每次插入区间 $[L_i, R_i]$ 之间的数，查询中位数
- 维护一个数据结构，支持区间修改，查询中间的数。
- 需要先把区间离散化一下。



F - Energy stones

- 每颗石头初始能量是 $E[i]$, 每秒钟增加 $L[i]$, 上限是 $C[i]$.
- 区间收割 M 次, 求收割能量的总和。
- 收割时候会把区间里的能量都变为0





F - Energy stones

- 从左到右扫描，用一个set维护收割的时间点，用树状数组去维护时间差的区间。
- 求前缀和以及区间和就可以了。



G - Make Shan Happy

- 给出一棵树，每个点有点权 W ，总根到叶子单调不减。需要回答询问 (x, k) ，找到最大的 y ，使得 $W[lca(x,y)] + k \geq y$
- 由于我也不知道怎么在线做，于是我们考虑离线，先将每个询问挂在树上，然后从 $n \sim 1$ 枚举 y 来回答这些询问。



G - Make Shan Happy

- 先考虑如何暴力做。
- 观察 $W[lca(x,y)] + k \geq y$ 。设 $p = lca(x,y)$ 变成 $W[p] + k \geq y$,
- 这样的 $p = lca(x,y)$ 是位于 $y \sim 1$ 的链上的, 而这个不等式的左边是只关于 p 的表达式。
- 因此我们在 $y \sim 1$ 的链上的每个点 p , 开一个大根堆, 将 $W[p] + k$ 这个标记值放入 p 的堆中。
- 这样询问就全部离线好了。
- 现在我们从 $n \sim 1$ 枚举答案 y , 来回答当前的 y 所能满足的所有询问。
- 由于 $p = lca(x,y)$, 这样的 p 是位于 $y \sim 1$ 的链上的, 因此我们反复询问 $y \sim 1$ 这条链上的最大标记值, 设为 $Mx = W[p] + k$, 如果 $Mx \geq y$, 即 $W[p] + k \geq y$, 则 Mx 这个值所属的询问的答案是当前枚举的 y 。重复上边这个步骤, 直到 $Mx < y$ 为止。



G - Make Shan Happy

- 接下来考虑如何优化这个暴力做法
- 链上询问最大值，我们想到了轻重链剖分，每次查询最大值只需要进行 $\log n$ 次重链前缀的查询。现在的问题是如何减少堆里的值。
- 从询问最大值出发，讨论两种情况：
 - 1、某个标记覆盖的链完整的包含了要查询的重链前缀。则显然该标记在查询部分的最大值是 [查询重链底端点的点权 + k]（因为越深的点点权越大）。
 - 2、某个标记只覆盖了要查询的的重链前缀的一个前缀部分。则显然该标记在查询部分的最大值是 [标记的重链底端点的点权 + k]（因为越深的点点权越大）。
- 对于这两种情况分别使用线段树进行维护。



G - Make Shan Happy

- 1、某个标记覆盖的链完整的包含了要查询的重链前缀。则显然该标记在查询部分的最大值是 [查询重链底端点的点权 + k] (因为越深的点点权越大)。
- 2、某个标记只覆盖了要查询的的重链前缀的一个前缀部分。则显然该标记在查询部分的最大值是 [标记的重链底端点的点权 + k] (因为越深的点点权越大)。
- 对于1：线段树维护 $W[p] + k$ 的最大值， p 是轻重链切换时候重链上的接点。
- 对于2：线段树维护 k 的最大值，同样也只在轻重链切换时候重链的接点位置维护。
- 至此，离线一个询问只需要 $2 * \log n$ 个标记值。
- 处理掉一个询问之后，需要在他经过的 $2 * \log n$ 个位置删除标记值。做法相同，每个点开一个大根堆维护标记值。删除掉某个点的标记值时，从堆中读出新的最大值去更新线段树即可。
- 标记值总数减少到了 $2 * q * \log n$ 。因此总的复杂度是 $O(n * \log^2 n)$



I - Chessboard

- 我们计“选择大小为 $k * k$ 的棋盘，每个方格放的玻璃球数都不小于 m 且每不同行不同列的方格内的玻璃球数量的总和均为 T ”的方案数为 $f_m(k, T)$ ，那么很显然的总的方案数即为：
- $\sum_{k=1}^{\infty} \sum_{T=k}^n f_m(k, T)$
- 很容易想到， $f_m(k, T) = f_0(k, T - k * m)$
- 接下来只要考虑如何求解 $f_0(k, T)$ （后记为 $f(k, T)$ ）。



I – Chessboard

- 容易证明，一个 $k * k$ 棋盘的排布方案满足“不同行不同列的方格内的玻璃球数量的总和均相同”要求，当且仅当该方案具有如下形式：
- $\sum_{i=1}^k a_i A_i + \sum_{i=1}^k b_i B_i$ ，其中 A_i (B_i) 为第 i 行 (列) 均为 1，其余行 (列) 均为 0 的 $k * k$ 矩阵
- 又因为该和为 T ，所以 a_i 和 b_i 需满足以下关系：

$$\sum_{i=1}^k a_i + \sum_{i=1}^k b_i = T \quad (\forall i, a_i \geq 0, b_i \geq 0)$$

- 容易得到，满足该条件的方案数为： C_{T+2k-1}^{2k-1}



I - Chessboard

- 但是这些方案中其实有一部分是重复的。重复的原因在于：当 a_i 均为非负时，事实上此时将所有 a_i 全部减1，将所有 b_i 全部加一，将得到一种完全相同的、但我们也记了一次数的方案。我们应该想办法将这种方案全部去除并只留下这种方案的唯一代表。很显然，我们只需要再减去满足：

$$\sum_{i=1}^k a_i + \sum_{i=1}^k b_i = T \quad (\forall i, a_i \geq 1, b_i \geq 0)$$

的方案数即可。因此所有的方案数即为： $C_{T+2k-1}^{2k-1} - C_{T+k-1}^{2k-1}$

- 回代到一开始的和式，朴素的求解方式复杂度 $O(n^2)$



K - Function

- 根据二平方和定理不难看出，所求函数即为：
- $f(x) = \begin{cases} 3e + 1 & (x = p^e, \text{其中 } p \equiv 1 \pmod{4}, e \geq 1) \\ 1 & (else) \end{cases}$ 的前缀和
- 然后考虑用dp的方法可以亚线性求出这个函数在 $O(\sqrt{n})$ 个点处在质数处的前缀和（本质上是“n以内模4余1的质数计数”）
- 再套用min25筛即可求解得到最终的前缀和了（新的旧的min25筛均可）
- （本来想稍微加强一下维护 $O(\sqrt{n})$ 个点处的前缀和的，奈何没来得及调处来。。。）

Thanks

