1001.Pandaemonium Asphodelos: The First Circle (Savage)

给定 n 个砖块, q 次操作

操作 1 等价于选取一段区间 $[l, r]$ 赋予一个属性, 新属性直接记作询问的 ID 即可

操作 2 等价于查询同属性段 + 操作 1 赋予一个已有的属性

操作 3 将同属性砖块权值加 v

操作 4 询问单块权值

思路

开一个数组 tag , 用于维护属性历史权重和

开一个珂朵莉树, 用于维护砖块属性

开一个线段树, 用于维护砖块权重

那么, 操作 3 只需要将 tag 中对应属性加上 v 即可

操作 $1, 2$ 修改区间属性时, 先在珂朵莉树上裂开区间, 当一个区间被赋予新属性时, 将原属性贡献减去新属性贡献存入线段树中, 即在时间上进行差分

特别的, 对于操作 2 修改属性后, 有可能珂朵莉树上裂开的区间左右段还有属性相同的区间, 需要进行合并

对于操作 4 只需要对线段树进行单点查询, 并加上属性 tag 存储的权值即可, 注意到这里的 n 很大, 应当进行动态开点, 标程中使用动态申请内存, 需要注意这种方式比较慢

复杂度

对于每次操作 1, 只会覆盖原先的整个区间, 或者是左右段覆盖原来至多 2 个不完整区间

操作 2 同理, 只不过会多一个同属性区间合并的过程

对于每个区间, 要么被一个询问覆盖一部分, 要么被一个询问完全覆盖, 要么被一个同属性区间合并, 后二者至多发生 1 次, 而最前者次数与 q 呈线性关系

故每次操作 $1, 2$ 时间复杂度在 $O(log\, n)$, 主要开销在线段树上, 最坏情况下每次都开一条链, 珂朵莉树上操作复杂度是 $O(log\, q)$ (与区间个数有关)

操作 3, 可以 $O(1)$ 解决

操作 4, 可以用线段树 $O(log\, n)$ 时间内解决

故总体时间复杂度 $O(qlog\, n)$, 空间复杂度不超过 $O(qlog\, n)$

1002.Jo loves counting

考虑唯一质数分解定理：$n=\prod\limits_{i=1}^{m}p_i^{k_i}\,(k_i\geq 1)$ 。

对于其因数 d 则有 $d=\prod\limits_{i=1}^{m}p_i^{c_i}\,(0\leq c_i\leq k_i)$ ，因此其方案数为 $\prod\limits_{i=1}^{m}(k_i-0+1)$ 。

现在，对于 $Good_n$ 的元素，有额外限制 $(d \bmod p=0\leftrightarrow n \bmod p=0)$ 。即对于 n 的所有质因子 p ，它们都应该是 d 的因子。

因此，d 需满足 $c_i\geq 1$ ，故集合大小的方案数为 $\prod\limits_{i=1}^{m}(k_i-1+1)=\prod\limits_{i=1}^{m}k_i$ 。

所以，若选定的数字为 n ，则答案为 $\dfrac{n}{\prod_{i=1}^{m}k_i}$ 。

故总答案为 $\dfrac{1}{M}\sum\limits_{n=1}^{M}\dfrac{n}{\prod_{i=1}^{m}k_i}$ 。

注意到 $\prod\limits_{i=1}^{m}k_i$ 为积性函数，故 $\dfrac{n}{\prod_{i=1}^{m}k_i}$ 也为积性函数，我们简记为 $\boldsymbol{f}(n)$ 。

观察到 $\boldsymbol{f}(p)=\frac{p}{1}=p$ ，我们可以构造 $\boldsymbol{g}(n)=n$ ，则 $\boldsymbol{f}(p)=\boldsymbol{g}(p)$ 。

基于此，我们可以构造一个积性函数 $\boldsymbol{h}(n)$ 使得 $\boldsymbol{h}$ 与 $\boldsymbol{g}$ 的迪利克雷卷积为 $\boldsymbol{f}$ ，即 $\boldsymbol{f}(n)=\sum\limits_{d|n}\boldsymbol{h}(d)\boldsymbol{g}(\dfrac{n}{d})$ 。

故对于所求答案，有：

$$
\begin{aligned}
&\frac{1}{M}\sum_{n=1}^{M}\frac{n}{\prod_{i=1}^{m}k_i}\\
&=\frac{1}{M}\sum_{n=1}^{M}\boldsymbol{f}(n)\\
&=\frac{1}{M}\sum_{n=1}^{M}\sum_{d|n}\boldsymbol{h}(d)\boldsymbol{g}(\frac{n}{d})\\
&=\frac{1}{M}\sum_{d=1}^{M}\boldsymbol{h}(d)\sum_{n=1}^{M}\boldsymbol{g}(\frac{n}{d})[d\mid n]
\end{aligned}
$$

由于 $\sum\limits_{n=1}^{M}\boldsymbol{g}(\dfrac{n}{d})[d\mid n]=\sum\limits_{n=1}^{\lfloor\frac{M}{d}\rfloor}\boldsymbol{g}(n)=\dfrac{\lfloor\frac{M}{d}\rfloor(\lfloor\frac{M}{d}\rfloor+1)}{2}$ 。

且 $\boldsymbol{f}(p)=\boldsymbol{g}(1)\boldsymbol{h}(p)+\boldsymbol{g}(p)\boldsymbol{h}(1)=\boldsymbol{h}(p)+\boldsymbol{g}(p)$ 得到 $\boldsymbol{h}(p)=\boldsymbol{f}(p)-\boldsymbol{g}(p)=0$ 。

因此，$\boldsymbol{h}(n)$ 仅在所有质因子次数均大于 1 的地方有值，即只有 n 为 Powerful Number 时，$\boldsymbol{h}(n)$ 可能不为 0 。

我们记 $[1,M]$ 范围内的 Powerful Number 构成集合 PN ，则原式可化简为 $\dfrac{1}{M}\sum\limits_{d\in\text{PN}}\boldsymbol{h}(d)\cdot\dfrac{\lfloor\frac{M}{d}\rfloor(\lfloor\frac{M}{d}\rfloor+1)}{2}$ 。

可以证明，集合 PN 的大小为 $O(\sqrt{n})$ 级别（见后文）。因此，只要我们能对每一个 PN 中的 d， $O(1)$ 地计算其对应的 $\boldsymbol{h}(d)$，则可以在 $O(\sqrt{n})$ 的时间内通过本题。

对于质数 p，由于 $k>1$ 时 $\frac{p^k}{k}=\boldsymbol{f}(p^k)=\boldsymbol{f}(n)=\sum_{d|n}\boldsymbol{h}(d)\boldsymbol{g}(\frac{n}{d})=\sum_{i=0}^{k}\boldsymbol{h}(p^i)\boldsymbol{g}(p^{k-i})$

$$
\begin{aligned}
&\therefore\frac{p^k}{k}\\
&=\sum_{i=0}^{k}\boldsymbol{h}(p^i)\cdot\boldsymbol{g}(p^{k-i})\\
&=\sum_{i=0}^{k}\boldsymbol{h}(p^i)\cdot p^{k-i}\\
&=p\cdot\sum_{i=0}^{k-1}\boldsymbol{h}(p^i)\cdot p^{k-1-i}+\boldsymbol{h}(p^k)\\
&=p\cdot\boldsymbol{f}(p^{k-1})+\boldsymbol{h}(p^k)\\
&=p\cdot\frac{p^{k-1}}{k-1}+\boldsymbol{h}(p^k)
\end{aligned}
$$

因此有 $\boldsymbol{h}(p^k)=-\frac{p^k}{k(k-1)}$。

我们可以枚举 $\sqrt{M}$ 内的质数 p，暴力枚举 $\lfloor\log_p M\rfloor$ 作为指数 k，然后暴力转移出所有的 $\mathrm{Powerful\ Number}$，分别作为 d，计算 $\boldsymbol{h}(d)\cdot\frac{\lfloor\frac{M}{d}\rfloor(\lfloor\frac{M}{d}\rfloor+1)}{2}$ 对和式的贡献，最后统一乘上 M 的逆元即可。

总时间复杂度 $T(M)=O(\sqrt{M})$

以下是关于 PN 大小是 $O(\sqrt{M})$ 的证明：

对于 $\forall n\in\mathrm{PN}$，则对于其任意质因子 p_i，必然有指数 $k_i>1$。以下先证明其必然可以写为 a^2b^3 的形式：

当 k_i 为奇数时，显然 $k_i\geq 3$，则必然可以分解出 p_i^3 进 b^3；可以分解出 $p_i^{k_i-3}$ 进 a^2 （k_i-3 为偶数）；

当 k_i 为偶数时，必然可以分解 $p_i^{k_i}$ 进 a^2；

按照上文的分解方法，则可以写为 a^2b^3。

$$
\begin{aligned}
&\therefore |\mathrm{PN}| \\
&\le \sum_{i=1}^{M} [i \text{ 可写为 } a^2 b^3 \text{ 的形式}] \\
&= \sum_{a=1}^{\lfloor \sqrt{M} \rfloor} \lfloor \sqrt[3]{\frac{M}{a^2}} \rfloor \\
&= O(\int_1^{\sqrt{M}} \sqrt[3]{\frac{M}{a^2}} \mathrm{d}a) \\
&= O(M^{\frac{1}{3}} \cdot \int_1^{\sqrt{M}} a^{-\frac{2}{3}} \mathrm{d}a) \\
&= O(M^{\frac{1}{3}} \cdot 3 \int_1^{\sqrt{M}} \mathrm{d}a^{\frac{1}{3}}) \\
&= O(3M^{\frac{1}{3}} \cdot a^{\frac{1}{3}} |_1^{\sqrt{M}}) \\
&= O(3M^{\frac{1}{3}} \cdot M^{\frac{1}{2} \times \frac{1}{3}}) \\
&= O(\sqrt{M})
\end{aligned}
$$

故 $|\mathrm{PN}|$ 的大小为 $O(\sqrt{M})$ 级别的。

1003.Slipper

设树的深度为d，在第i层$(1 \le i \le d)$和第$i+1$层中新增两个点l_i, r_i，l_i连向所有第$i+1$层的点，r_i连向所有第i层的点，对于原来树中所有的点，向l_{dep_u+k-1}连一条权为p单向边，向r_{dep_u-k}连一条权值为p的单向边，在修改后的图中跑dijkstra求s到t的最短路即可，复杂度O(nlogn)。

1004.The Surveying

$control\ point$： 控制点

$detail\ point$： 碎部点

对于每个【控制点】，暴力求出所有【碎部点】的可见情况，用 $bitset$ 维护。

之后再状压来计算答案，注意特殊的限制条件：每个【控制点】都需要能看到至少一个另外的【控制点】。

时间复杂度为 $O(nm^2 + 2^n nm/32)$ 。

1005.3D Puzzles

一道舞蹈链模板题。

用 64 列来表示每一个格子只能存在一个方块，再用 13 列来表示每种拼图能且只能出现一次。

注意在构建矩阵的时候去重，不然算出的解会重复。

直接搜索也是可以的，但是需要很多优化，比如改变枚举顺序什么的。

时间复杂度都是 $O(\text{能过})$ 。

1006.BBQ

考虑 **dp**，每次枚举原串的一个区间，计算这个区间变成恰好一个**组(abba)**的最小代价来转移。

需要注意到一个性质，原串中的某个区间，如果其长度大于等于 8，就一定没必要变成新串的一个**组**。

因为，假设其长度为 n ，$n \geq 8$ ，变成一个**组**需要其长度变为 4 ，至少需要先删掉 $n-4$ 个字符，总共需要不低于 $n-4$ 的代价。

而如下策略一定可以不低于上述代价完成变换：删掉后面的 $n-8$ 个字符，然后对于剩下的 8 个字符，前 4 个字符和后 4 个字符分别花费 2 的代价进行修改，一定可以得到两个合法的 **组** ，这样总共是不高于 $n-4$ 的代价。

因此转移时可以只枚举长度不超过 7 的区间来转移。

$$dp[n] = \min_{i=1}^{7} (dp[n-i] + \text{代价}(n-i+1, n))$$

另外注意到一个有用的操作是删除掉最终没有形成**组**的字符，即 $dp[n] = dp[n-1] + 1$ ，两种转移取较小值。

然后就是代价如何计算，可以把一段区间中的字符的相等的关系记作一种模式，例如 "abcd"可以记作"0123"，"defg"也可以记作"0123"，"abccba"和"cbaabc"都可以记作"012210"，即相同的字符用相同的数字替换，不同的字符用不同的数字替换，并且按该字符最左出现位置从0开始编号，可以发下这样的模式其实是很少的，分别对每种模式提前算好代价就可以了。

然后计算代价方法很多，比如可以先枚举这个模式按最小代价最终的样子，例如最终是"1221"，然后去二维 dp 计算一下变成这个串最少要多少代价。

因为数据范围比较大，实现需要注意效率，一些要注意的地方：最好是$dp[n]$ 向 $dp[n+1], dp[n+2]...dp[n+7]$ 转移，这样在计算模式时可以利用好上次的信息。记录代价可以把模式编码成7进制数，开数组直接查表。

1007.Count Set

假设有一个 n 个点的图，排列 p 中的某个项 p_i 表示从点 i 到点 p_i 有一条边。则可以发现这是一个由若干不交有向环组成的图。

则题目中的条件可以等价于，从每个环中挑选若干个点，使得每个环中被选出的点不相邻，从一个大小为 m 的环中选出 k 个点的方案数是 $\binom{m-k}{k}+\binom{m-k-1}{k-1}$ ，因为可以把被选出的点当作大小为 2 的段，然后不选的点当作大小为 1 的段，用大小为 2 保证被选的点不相邻，问题就转化成总共有 $m-k$ 个块，挑出 k 个块的方案数，如果在序列上，这就是一个组合数，在环上也只需要挑一个位置拆开，变成序列问题，然后分有没有大小为 2 的块恰好在断开处讨论一下，发现就是上面的式子。

推出这个式子之后只需要列出每个环的生成函数然后 NTT 合并就好了 。

1008. AC/DC

首先字符串 t 在字符串 S 内的出现次数可以用 SAM 解决。

但是每次修改就重建 SAM 的复杂度显然是无法接受的。

法一：考虑使用 LCT 维护 $fail$ 树结构。

设 $S[l..r]$ 为当前对应字符串。

建立 $S[1..r]$ 的 SAM。

每次操作 1 直接利用 LCT 在线维护。

每次操作 2 相当于取消 l 位置的 $endpos$ 标记。

查询相当于查询 $fail$ 树上子树和。

时间复杂度 $O(nlogn)$。

法二：考虑使用定期重构解决。

设 $T=\sqrt{n}$，每 T 增加或删除字符操作就重构 SAM，对于还未来得及重构而积累下来的操作用哈希计算即可。

此做法可扩展至前面加字符的模型。

复杂度 $O(m\sqrt{n}+\sum|t|)$。

1009.Cube Rotate

题意：给定一个六面体，每个面有个不同的数字，做 n 次旋转，每次旋转将顶面朝一个方向旋转90度，每次旋转后将正面的数字乘起来。在 n 次旋转中，有 m 次方向是未知的，给定初态、终态和乘积，询问有多少种可行的旋转方向。

由于 $m \leq 20$ 考虑折半搜索，四面体的摆放共有24种状态，预处理每种状态开始，已知的一串旋转序列操作后的乘积。之后是基础的折半搜索操作，搜索出前一半和后一半的乘积，查询乘积。

时间复杂度$O(24n + 4^m \times log(4^m))$

1010.Bragging Dice

此题的idea源于出题人玩骰子游戏时，本以为是一个输出Win！的签到题，直到前一天找人验题面时发现当双方都roll出全部不同的点数时，此时先手无论如何claim都是会被challenge成功的。除此之外的情况，先手都能claim出最大的 x 个 y 。

关于题面描述不清楚的问题，接受所有的批评指正，这道题的题面在此之前已经找很多对这个游戏不知道或者一知半解的人，在此情况下题面仍有许多不严谨的地方。

出题人本以为对题面的歧义不影响解题，但仍旧给很多队伍带来了困扰，我们也采取了一定的补救措施：对题面的补充以及对每个clarify的回复。

关于本题的所有问题，向所有队伍致以歉意。

1011.Kazuha's String

本题一共有 24 种本质不同的字符串，具体变化如下：

$bc \to a \ \ ca \to bb \ \ aca \to abb$
$acb \to acb \ \ baa \to b \ \ bab \to acc$
$bac \to bac \ \ bba \to c \ \ bbc \to ba$
$cbc \to bb \ \ cca \to cbb \ \ ccb \to ccb$
$ccc \to ab \ \ abaa \to ab \ \ abab \to cc$
$abba \to ac \ \ abbb \to a \ \ abbc \to aba \ \ acbc \to abb$
$acca \to acbb \ \ accc \to b \ \ baca \to accb$
$bacc \to cb \ \ cbaa \to cb \ \ cbab \to bac$
$cbba \to cc \ \ cbbb \to c \ \ cbbc \to cba$
$ccba \to abac \ \ ccbb \to aba \ \ ccbc \to cbb$
$abaca \to ccb \ \ abacb \to cbac \ \ abacc \to acb$
$acbaa \to acb \ \ acbab \to abac \ \ acbac \to bacb$
$acbba \to acc \ \ acbbb \to ac \ \ acbbc \to acba$
$accba \to bac \ \ accbb \to ba \ \ accbc \to acbb$
$bacba \to cbac \ \ bacbb \to cba \ \ bacbc \to accb$
$cbaca \to bacb \ \ cbacb \to acba \ \ cbacc \to ccb$

手玩两小时可能是能玩出来的，当然也有更巧妙的做法，本题的操作相当于是六面体旋转群，也就是 $S4$ ，模拟旋转状态即可。

1012.Buy Figurines

考虑时刻 a_i 第 i 个人到达商店时，如何快速查询最短并且编号最小的队伍，可以用 set 维护队伍的信息，又考虑到每个人的开始时刻和结束时刻是不同的，可以用一个优先队列来维护时间轴的信息。先按照时间轴的顺序，模拟商店入队和出队的情况，进而求出每个人的购买雷电将军人偶的开始时间和结束时间，最后一个人的结束时间即为本题所求答案。