因懒无名

子任务1

直接模拟，爱怎么模拟怎么模拟。

子任务2

接下来对于某个颜色c，把所有在 两个颜色为c的点之间的路径上 的点与边的 集合叫做 颜色c的块。显然每个颜色的块都是一棵树。

当$l = r$时，显然是询问颜色l的块的直径。

考虑怎么维护。对于某个颜色，插入一个新的点是很好维护的，直接计算出新点与原直径的两端点的距离的最大值， 然后若这个值大于原直径的长度， 更新直径长度和端点即可。但似乎无法删除。

好插入不好删除的话，套个线段树分治就行了。

子任务3

首先考虑没有修改的情况。

发现一个性质，两个块的并的直径的端点一定被两个块分别的直径的端点包含。那么预处理出来每个块的直径的端点，用线段树维护一下区间的直径的端点即可。

有修改的话，再套个线段树分治上去就行了。

~~(这题太水了吧...)~~

树论

$\text{Subtask } 1，1\text{pts}$

人口普查分。为了避免今日有各种打挂然后爆零的选手吧。

$\text{Subtask } 1 \sim 2，2\text{pts}$

不难发现，$n = 1$ 的时候 SG 值只可能是 0。对于 $n = 2$ 的情况，当且仅当根节点上的棋子个数是奇数时，SG 值为 1。简单判断一下就行了。

$\text{Subtask } 1 \sim 3，6\text{pts}$

然后不难看出，一个节点上的棋子个数的奇偶性不变的时候，SG 值是不变的。那么只需要维护 $0/1$ 的状态就行了。那么对于 $n \in [3，5]$ 打表存一下 $n^4 \times 2^n$ 种局面的 SG 值，就能白嫖这些分了。

$\text{Subtask } 1 \sim 5，25\text{pts}$

事实上，每个棋子都可以看做是一个独立的游戏。然后对于根确定的情况，一个棋子每个点的 SG 值都是固定的。那么我们只需要每次修改之后 dfs 当前根，然后将所有棋子个数为奇数的点的 SG 值异或起来就是答案了。

打表或者理性分析之后，可以发现，一个点的 SG 值就是它到其子树内最深的叶子节点的距离。

Subtask 6，17pts

既然所有操作的根固定为 1 号点了，则所有点的 SG 值不会发生改变了。那么预先 dfs 一遍求出所有点的 SG 值，然后子树和链修改等价于让这些点的 SG 值在答案中计算或者不计算贡献这两种状态中翻转。树剖+线段树维护翻转标记就行了。

Subtask 7，18pts

考虑换根操作很麻烦，但是当前修改变成了单点修改。考虑可以预先处理出以 1 为根的答案。不难发现，当我们把根从 u 改到与 u 相邻的 v 点上的时候，只有 u 和 v 的 SG 值可能发生变化。我们只需要判断 u/v 在整棵树上的最远叶节点是否需要往 v/u 走就行了。

然后考虑单点修改。我们发现，我们预处理的答案只能是 2^n 种棋子状态中的一种。那么，也就意味着我们需要对于单点修改，能够快速维护所有根的答案。

> 引理 1：对于所有 n 种根的状态，对于任意一个点 x，其 SG 值仅有不超过两种取值。

先考虑定义【x 在 y 的方向】为，y 到 x 最短路径上第一个点的编号。则考虑求出每个点 x 距其最远的一个叶子节点 u，以及另一个叶子节点 v，满足 x 到 v 的距离最大且 v 在 x 的方向与 u 在 x 的方向不同。则 x 的 SG 值仅可能是 u 或者 v 到其的距离。

因为考虑根节点 rt，若 rt 在 x 的方向与 u 在 x 的方向不同，则 SG 值就是 u 到 x 的距离。否则，rt 为根时，u 就不在 x 的子树里头，而此时 v 就是 x 子树内距离 x 最远的叶子节点。则 x 到 v 的距离就成了 x 的 SG 值。证毕。

然后，我们钦定 f_x 表示 x 到 u 的距离，g_x 表示 x 到 v 的距离。不难发现，除了 rt 在 x 的方向与 u 在 x 的方向相同的时候，也就是 rt 在一段连续的 dfs 序（或者一个前缀和一个后缀）时，$\mathrm{SG_x}$ 才可能取到 g_x，其余情况均为 f_x。则每次单点修改的时候，我们只需要在 dfs 序维护的线段树上，对于一段或者一个前缀以及一个后缀异或上修改点的 g_x 值，其余位置异或上 f_x，这样就能维护任意一个点为根时 SG 值的异或和了。

这里考虑如何不使用 LCT 支持既有换根又有子树操作：

考虑随便以一个点为根，计算 dfs 序。若当前根为 rt，对 x 进行子树操作，则对于 x 子树的 dfs 序的定位，可以讨论如下三种情况：

1. rt 在 x 的 dfs 序区间内，且 $x \neq rt$：找到 x 的儿子节点 u，满足 rt 在 u 的 dfs 序区间内。则从 $[1, n]$ 中刨除 u 的 dfs 序区间，剩下的一个前缀和后缀即为 u 在 rt 为根时的子树 dfs 序区间。
2. rt 不在 x 的 dfs 序区间内：即为 x 的 dfs 序区间。
3. $rt = x$：$[1, n]$ 即为 x 的子树区间。

Subtask 8，20pts

其实吧...这个 Subtask 本意上是为了给那些会正解但是不会修改子树的人给的，也是给那些冲 LCT 冲对了但是写不出子树修改的人的安慰分...

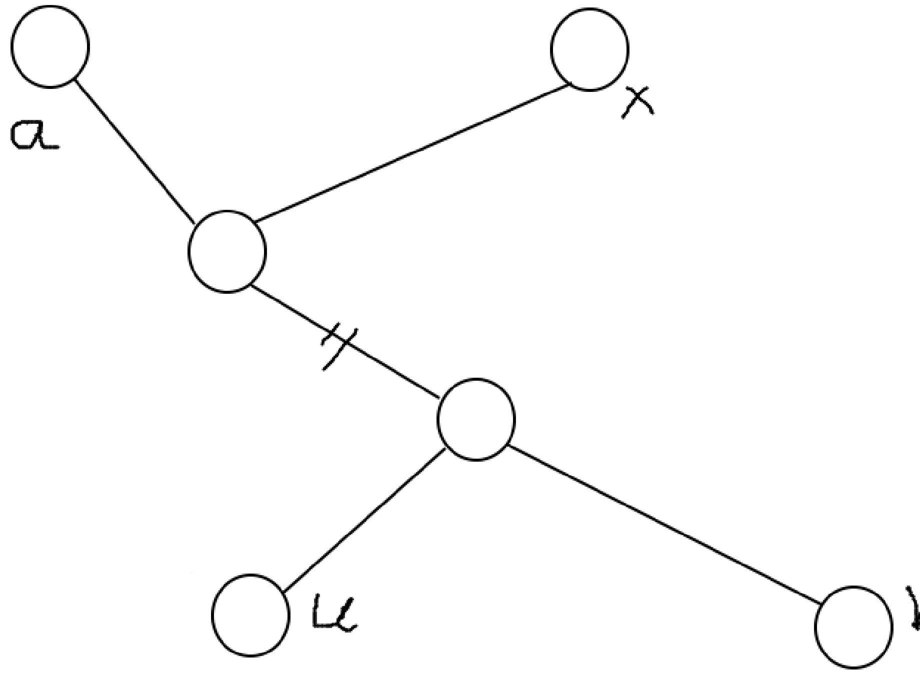
Subtask $1 \sim 9$，100pts

承接 Subtask 7 的思路。

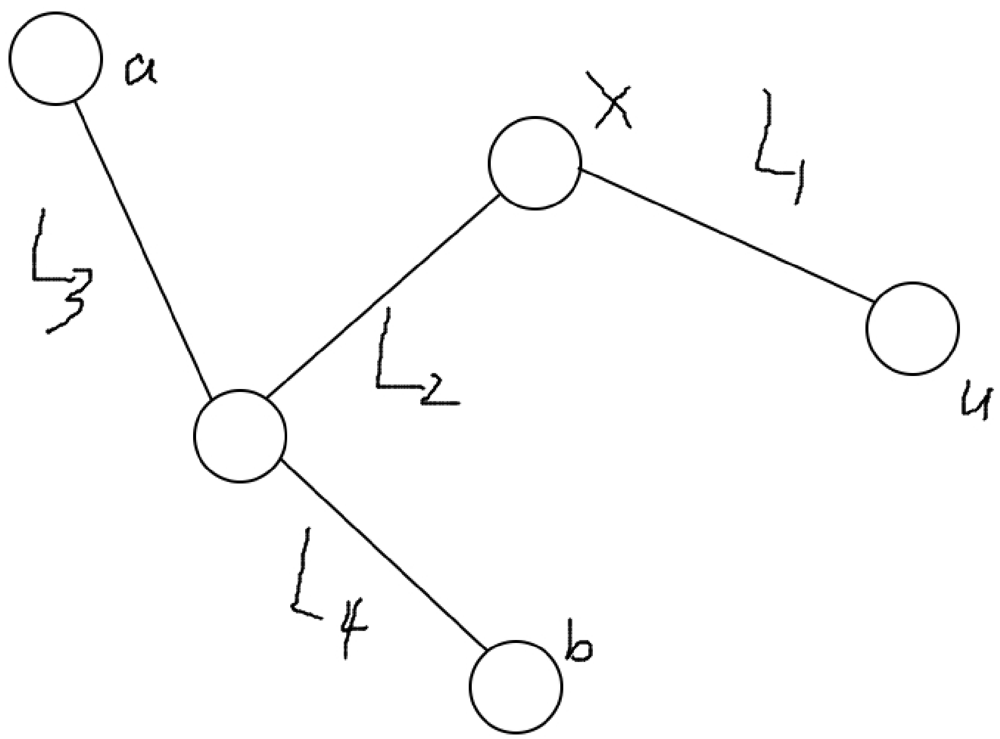
考虑每个点的 SG 值只可能取到两种取值之后，我们再深入思考这个取值的分布有没有什么规律。

> 引理 2：u 号点一定是树的某条直径的一个端点。

考虑反证法。若有一条直径是 $a \to b$，且 $x \to u$ 的长度比 $x \to a$ 和 $x \to b$ 的长度都要长。考虑 $x \to u$ 与 $a \to b$ 交与不交两种情况：



考虑交的时候，由上图可以看出，若 $x \to b$ 的长度没有 $x \to u$ 长，则 $a \to u$ 一定比 $a \to b$ 长。则不满足直径的条件。



考虑不交的情况：则有 $L_1 > \max(L_3,\ L_4) + L_2 > \max(L_3,\ L_4)$。则 $L1 + L2 + \max(L3,\ L4) > L3 + L4$ 一定成立，则 $a \to b$ 一定不能是直径，与假设也矛盾。

所以 u 一定是直径的一个端点。

> 引理 3：一棵树的直径中点唯一。

证明考虑分直径长度奇偶（这里认为直径长度是：

对于直径长度是偶数的情况，不难发现，直径中点处于一条边上。则此时若存在两条不同的边有直径中点，则将两条“直径”取长链合并之后可以得到更长的直径。

对于直径长度是奇数的情况，也是一个道理。若中点不落在同一个点上，则可以将中点连接，两端分别取无交的半边，就可以得到更长的直径。

发现上面两条引理之后，我们就可以想到：除了长度为奇数时的中点，树上每个点 x 的 u 号点都是往直径中点方向的！

那么当我们以直径中点为树根时，所有点（除了直径中点）的 SG 值都恰好取到 g_x！

那么这个时候，我们从直径中点换根至 x 时，路径上的点的 SG 值都恰好会变成 f_x！

这样，我们就可以利用树剖和线段树维护翻转标记，快速维护出每个点在 x 为根的 SG 值了。

但是，我们注意到，如果直径长度是偶数的时候，直径中点在一条边上。这可不太妙。

一种可行的解决方案是建立一个虚点，中点连向两边的点，然后钦定虚点的 f 值与 g 值为 0。

然而，事实上随便钦定两边的一个作为根，特殊考虑根的情况也是可行的。

剩下的事情就是修改了。考虑修改操作是另一个维度上的翻转。维护一个 2×2 的矩阵，支持行翻转和列翻转即可。

LCT 和 树剖+线段树 均可。

游戏

60 分

之前 CF div1 D 原题，算是给大家送 60 分了。

设 $F(i)$ 表示游戏在第 i 个人这里结束的期望回合数。

考虑放宽限制，即使有一个人拿到了所有饼干也不会结束。

然后考虑设 $D(i)$ 表示第 i 个人第一次拿到所有饼干的期望回合数。

发现这个东西显然只和 a_i 有关，于是就可以设 $H(x)$ 表示当第 i 个人有 x 个饼干的时候 $D(i)$ 的值。

然后 $H(x) = AH(x-1) + BH(x) + CH(x+1)$，这个东西可以递推解决。

于是就有 $D(i) = \sum_j F(j) + C(i,\ j)P(j)$。

其中 $C(i,\ j)$ 表示当 i 拿到所有饼干之后，j 第一次拿到所有饼干的期望回合数，显然当 $i = j$ 的时候 $C(i,\ j) = 1$ 否则 $C(i,\ j) = H(0)$。

$P(j)$ 表示第一次拿到所有饼干的是第 j 个人的概率，发现 $\sum P(i)=1$。

于是就有：

$$\begin{aligned}\sum(D(i)) &= n\sum(F(i))+\sum(C(j,i)P(j)) \\ &= n\sum F(i)+\sum(P(j)sum(H(0)[i!=j])) \\ &= n\sum F(i)+H(0)\sum P(j)sum([i!=j]) \\ &= n\sum F(i)+H(0)\sum P(j)(n-1) \\ &= n\sum F[i)+(n-1)H(0) \\ &= \sum H(a_i)\end{aligned}$$

于是就可以通过 H 来计算出 $\sum F$。

时间复杂度 $O(n\log n)$。

100 分

比较有趣的一个东西，idea来源。

只有期望才能打败期望。

我们可以设置一个势能函数 $f(S)$，其中 S 表示当前的状态。

假设进行一次操作之后的状态为 S'，那么只要 $f(S)-f(S')$ 的期望值为 1，那么最终的答案就是初始状态和结束状态的 $f(S)$ 的差值。

如何构造这样的一个函数?

首先我们要求的其实是一个满足这个等式的函数 f:

$$\sum_i f(a_i)=\sum_i \frac{a_i}{m}\left((f(a_i-1)+1)+\left(\sum_{j\neq i}\frac{1}{n-1}f(a_j+1)+\frac{n-2}{n-1}f(a_j)\right)\right)$$

也就是：

$$\sum_i\left(\frac{a_i}{m}(f(a_i-1)+1)+\frac{(m-a_i)}{m(n-1)}f(a_i+1)+\frac{(m-a_i)(n-2)}{n-1}f(a_i)\right)$$

我们不妨设：

$$f(a)=\frac{a}{m}(f(a-1)+1)+\frac{m-a}{m(n-1)}f(a+1)+\frac{(m-a)(n-2)}{m(n-1)}f(a)$$

然后把 $a=0$ 带入等式中，可以得到 $f(0)=f(1)$ 。

于是不妨设 $f(0)=f(1)=0$，从 2 开始往 m 递推，移项之后得到：

$$\frac{m-a}{m(n-1)}f(a+1)=(1-\frac{(m-a)(n-2)}{m(n-1)})f(a)-\frac{a}{m}(f(a-1)+1)$$

$O(n)$ 预处理逆元之后直接递推即可，时间复杂度 $O(n)$。