Jump

为了方便定义第 $0, n+1$ 个点分别为 k, m，对应的 $b_i = 0$。

那么转移方程是显然的：$dp(i) = \max\limits_{0 \le j < i} \{dp(j) + b_i - \lceil \frac{T_i - T_j}{D} \rceil \times a\}$。

这个东西乍一看没法优化，因为那个上取整很不好弄。

此时有一个经典套路，利用余数的关系分讨拆开取整。

就是

$$\lceil \frac{T_i - T_j}{D} \rceil = \begin{cases} \lfloor \frac{T_i}{D} \rfloor - \lfloor \frac{T_j}{D} \rfloor + 1, & (T_i \bmod D > T_j \bmod D) \\ \lfloor \frac{T_i}{D} \rfloor - \lfloor \frac{T_j}{D} \rfloor, & \text{otherwise.} \end{cases}$$

下取整可以换成 C++ 自带除法。

于是方程就可以分别写成 i, j 相关项，于是我们用一个权值线段树维护一下 $dp(i) + a \times \lfloor \dfrac{T_i}{D} \rfloor$，每次转移分别询问前后缀 $\max$ 即可。

Match

注意到两个串匹配的充要条件是：对于 $\forall i \in [1, len]$，两个串对应位置的字符到上一个对应字符的距离相等。

比如 $S = aba, T = bab$，写出来就可以变成 $S' = 002, T' = 002$。

于是问题可以转化为一个简单的字符串匹配了。

但是注意到 $S = abab, T = aba$ ，若我们正在匹配 $S[2 \ldots 4]$ 和 T，直接写出来就是：$S'[1 \ldots 3] = 022, T' = 002$，发现不匹配了，这是因为 $S[1]$ 这个位置的贡献本来应该不算，直接取原串就会死掉。

所以我们删掉一个位置的时候，要考虑当前位置是否会对后面某一个位置产生影响。

这个利用字符串 Hash 可以 $O(1)$ 做，就找到对应位置，减去即可。

剩下的每次 $O(1)$ 扩展，都是字符串 Hash 的基本操作，很简单。

Graph

注意到答案一定是最短路长度，不然的话显然存在一个删掉后最短路仍然存在的颜色。

那么，直接求一次最短路，然后考虑 $S \to T$ 的路径上的一条边 (u, v)，它只需要被染色成 $\max(dis[S \to u], dis[S \to v])$ 即可。

其本质是找到合适的，类似于按层染色的方式使得方案成立。

正确性显然。

Xor

题面有点叙述不清，或者说有点绕。

虽然说是满足 $1 \le i < j \le k$ 的点对 (i, j)。

但实际上如果 $k = 1$，那么这样的 (i, j) 不存在，所以长度为 1 的子序列也是满足条件的。

看到这个问题，很容易想到直接把所有数丢进 Trie 树来考虑。

我们可以做一个 dp，设 $dp(u)$ 表示在以 u 为根的子树中选两个数，使得它们的异或和 $\ge X$ 的方案数。

我们假设 u 的层数为 i （从高到低插入），那么如果 $2^{i-1} > X$，显然不管我们怎么从 $ls(u), rs(u)$ 当中选，都一定可以选出大于等于 X 的方案。

所以 $dp(u) = dp(ls(u)) \times dp(rs(u))$.

其中 $ls(u)$ 表示 $tr[u, 0]$，rs 反之。

然后考虑，如果 2^{i-1} 恰好是 X 的最高位，那么此时一定分别从两边选，且每一边至多选一个，不然异或之后全是 0 了，不合法。

于是首先算上只选一个和空集的方案数，这部分是 $siz(u) + 1$。

剩下的部分的话，问题就转化为，在 $ls(u), rs(u)$ 当中分别选一个，使得它们的异或和大于等于 X，这个也可以 dp。

我们设 $f(x, y)$ 表示这个 dp 数组，并记 x, y 的层数为 j。

分类讨论 `bit(j) = X >> j & 1` 的取值，可以得到转移：

$$f(x,y) = \begin{cases} f(ls(x), ls(y)) + f(rs(x), rs(y)) + siz(ls(x)) \times siz(rs(y)) + siz(ls(y)) \times siz(rs(x)) & (bit(j) = 0) \\ f(ls(x), rs(y)) + f(ls(y), rs(x)) & (bit(j) = 1) \end{cases}$$

1 的话就是两边必须错开选，0 的话两边怎么选都行。

注意边界的时候要返回 siz!

而且 Trie 树处理的时候 bit 不要卡着上界 63，小一点，不然溢出了就寄了。

sol by black_trees