回文串

Algorithm 1

$dp_{l,r}$ 表示区间 $[l,r]$ 能够最多变成多少个串，转移枚举左右的第一个串的位置，如果相等就可以转移。判断字符串相等可以通过字符串哈希实现。

时间复杂度：$\Theta(T\cdot n^3)$，期望得分：50 分。

Algorithm 2

可以发现我们最后只要 $dp_{1,n}$，而只有 $dp_{i,n+1-i}$ 可能转移到这个东西。可以进一步发现，求出 $dp_{i,n+1-i}$ 也只需要这些位置，所以 dp 的状态数降为 $\Theta(n)$。

时间复杂度：$\Theta(T\cdot n^2)$，期望得分：70 分（判相等也可以通过 KMP 的 $next$ 数组来实现，复杂度不变）。

Algorithm 3

我们还是考虑一个区间 $[i,n+1-i]$。我们称 T 是 S 的一个 $border$，当且仅当 S 可以被表示为 $T\circ S'\circ T$（这里的 $border$ 定义和真正的有些区别，真正的两个 T 是可以重叠的）。

我们考虑最短的那个 $border$，假设他是 T。那么我们断言，这里直接把 T 拆下来是最优的。下面是证明：

> 考虑一个方案中，S 被断为 $A\circ B\circ A$，则可以发现 A 也是 S 的 $border$。我们设 $x=|T|$ 而 $y=|A|$。
>
> - 如果 $2x\le y$，那么 S 其实长这样：
>
> ```
> ------------ S
> aaaaaa aaaaaa A
> --         -- T
> ==  == ==  == (T)
>   xx     xx   (X)
> ```
>
> 因为两个 A 是一样的，所以上面等号的部分都是 T。那么因为 A 是 $border$，所以两个 `x` 的部分是一样的（因为 `=` 都是一样长的），设为 X。那么我们可以把 S 分为 $T\circ X\circ T\circ B\circ T\circ X\circ T$，显然比 $A\circ B\circ A$ 更优。
>
> - 而如果 $2x>y$，那么就是这样的：

```
------------ S
aaaaaa aaaaaa A
----     ---- T
  ==== ====   (T)
  xx     xx   (X)
yy         yy (X)
```

可以发现等号部分也是 T，那么我们考虑两个 T 的重叠部分 X（上面的 `x`），利用两个 `-` 的部分可以得出图中的 `y` 也是 X，那么 X 也是 $border$，而且比 T 短。这与 T 是最短的 $border$ 矛盾，所以不存在这种情况。

故证毕。

那么我们每次贪心选择最短的 $border$ 并把它砍下来，然后递归即可得到答案。

时间复杂度：$\Theta(T\cdot n)$，期望得分：100 分（如果判相等用了后缀数组那么就是 $\Theta(T\cdot n\log_2 n)$ 的，也可以得到 90 分；但是如果用 KMP 就会退化到 $\Theta(T\cdot n^2)$，只有 70 分）。

PS：注意字符串哈希用 `unsigned long long` 的自然溢出是**错的**，但是在这个题里面卡不掉。这是因为卡他的原理是自然溢出会把构造出来的两个不同的串看成一样的，但是这样构造出来的两个串可以分得很碎，不会询问到整个串的地步。但是**自然溢出是可以被卡掉的**，尽量用双模数甚至三模数哈希比较好。

字符串

出题人想了很久也没有想出来除了标算和暴力卡常之外的做法，所以部分分没有针对某个算法给分，而是给了 n,m 许多梯度。如果有选手写了优于暴力的做法而没有和暴力有很强的区分度，出题人对此表示非常抱歉。

Algorithm

可以发现通过操作 1，我们可以得到任意一个字母组成一样的串，那么我们显然会先把所有字母组成一样的串都变出来再用魔法，。

那么我们可以把“串”这个状态变成“字母出现次数”这个状态，于是我们可以用 $(a,b,c,n-a-b-c)$ 表示当前状态，则这样的状态只有 $\Theta(n^3)$ 个（实际上还要 $\div 6$），而每个可以组成多少个不同的串就是一个简单的排列组合问题了。

那么我们把这些状态每个建一个点，如果两个点可以用一次魔法得到我们就连一条边，则相当于要求一条路径，满足路径上面的点权和最大。直接强连通分量缩点之后拓扑序 dp 即可。

时间复杂度：$\Theta(n^3\cdot m)$。

PS：注意答案可能超过 `long long` 范围，但是经过计算（可以用浮点数计算所有点的点权和，数量级不会错），可以发现答案不会超过 10^{32}。于是用两个 `long long` 拼起来即可，如果写高精度可能会常数太大导致超时。

PPS：注意两个 `long long` 拼起来之后除法并不好做，但是组合数可以消因数算（把分子分母都质因数分解，然后相同的数字减一下即可），可以发现组合数的结果不会超过 `long long`。

二叉交换树

Algorithm 1

我们发现这个编号总是变得东西不太好维护，考虑在每个点一开始的编号维护。那么我们相当于要求现在的 T_x 是多少（L_x 本质上就是 T_{2^n-1+x}），可以发现我们只要维护每个点的两个孩子，即可通过类似 $Trie$ 树进行询问了。

那么我们对于修改，直接暴力依次把每个节点（要把 T_x 转为一开始的编号，就是依次询问）的孩子换一下就可以维护了。

时间复杂度：$\Theta(2^n \cdot q)$，期望得分：30 分。

Algorithm 2

对于有特殊性质一的测试点，我们可以发现前一个点的交换并不会影响后一个点对应的编号（因为他们都是同一层的，相互没有隶属关系）。所以我们可以用一个线段树或者树状数组维护每个点的孩子有没有被交换，修改就相当于是区间异或 1，而询问直接单点查询即可得到一个点的孩子。

时间复杂度：$\Theta(nq)$，期望得分：10 分。

Algorithm 3

对于特殊性质二，可以发现相当于是翻转 $[2^x, 2^{x+1}), [2^{x+1}, 2^{x+2}), \cdots, [2^{y-1}, 2^y)$，于是相当于翻转 $[2^x, 2^y)$，同样树状数组即可。

时间复杂度：$\Theta(nq)$，期望得分：20 分（这里没有计算特殊性质一的点）。

Algorithm 4

上面的两个算法给了我们重要的启示：我们如果一次翻转一堆互无隶属关系且编号连续的点，可以 $\Theta(n)$ 一起把它们翻过来。

那么我们对于修改，把对于每一层的修改分开来做，变成每一层翻一个区间。互无隶属关系且编号连续的点一定是某个点的同一层的孩子。那么我们可以发现，这一层中有且仅有第 $[k\cdot 2^x,(k+1)\cdot 2^x)\ (k,x\in\mathbb{N})$ 个点满足这个条件。

我们可以联想到区间修改线段树：线段树相当于是按照 x 从大到小的顺序，每次尽可能拆掉这个区间中的 $[k\cdot 2^x,(k+1)\cdot 2^x)$，然后拆不掉的递归放下去。那么对于这个题，我们也这么做，每次拆出来这个完整的区间就直接在另一个线段树或者树状数组上面 $\Theta(n)$ 修改即可。

那么分析一下这样做的复杂度，这样一个区间最多被拆成 $\Theta(n)$ 个小的整区间，然后每个整区间都可以 $\Theta(n)$ 修改。因为我们先分了层，所以会有 $\Theta(n)$ 个大区间。所以这样是 $\Theta(n^3\cdot q)$ 的。

真的吗？其实可以发现，只有 L,R 所在的层才可能被拆出 $\Theta(n)$ 个小区间，而剩下的层都是要么全都翻转要么全都不翻转，只会被变成至多一个小区间。所以这样做的复杂度其实是 $\Theta(n^2\cdot q)$ 的，而且如果有特殊性质的话，那么每一层都是至多一个小区间，只要 $\Theta(n\cdot q)$ 了。

时间复杂度：$\Theta(n^2\cdot q)$ 或 $\Theta(n\cdot q)$，期望得分：85 分。

Algorithm 5

其实上面那个做法已经很接近正解了。可以发现我们在分解为小区间的时候用了一个线段树，然后另外为了维护是否翻转这个信息又用了一个线段树，非常浪费。我们考虑能不能把两个线段树给拼起来，一起用。

我们还是每一层分开做，假设第 d 层的区间是 $[L,R]$。在线段树的代表 $[l,r]$ 的节点上，我们维护 n 个懒标记，其中第 i 个表示这一层的第 $[l,r]$ 个节点要不要被翻转（懒标记可以类似状压 dp 那样压进一个 `int`）。

在修改时，如果当前节点的 $[l,r]\subseteq[L,R]$，我们就在当前节点打一个 d 的懒标记。否则，我们先求出当前节点的当前层的懒标记，得到当前节点的两个孩子要不要被翻转，然后把懒标记传下去（相当于给两个孩子异或上当前节点的懒标记）。如果当前节点被翻转了，我们要把修改区间 $[L,R]$ 变成翻转之后的样子（是左孩子的一段前缀和右孩子的一段后缀），然后继续递归下去。

对于查询 T_x 对应着什么，也是类似的操作：求出当前节点有没有被翻转，然后如果被翻转了就求出来翻转后应该对应的位置，继续递归下去查询即可。

时间复杂度：$\Theta(n\cdot q)$，期望得分：100 分。

PS：如果还不理解的，可以参考一下标程，写的比较清楚。