string

板子合集，做法可能很多，这里只讲 SAM 做法

操作 3 是打酱油的，用来卡掉一些暴力做法，SAM 实时维护：$\sum_i len[i]-len[fa[i]]$ 即可

subtask3

可以拿这 n 个串构一棵 $trie$ 树.

我们可以先离线，对这棵 $trie$ 树构出广义SAM.

由于操作 2 需要两个串存在一定顺序时才可以统计，所以 SAM 要 bfs 序构造.

对于操作 2，维护一下每一个串每一个时刻的所在位置就好了

对于操作 4，也是维护一下 T 所代表的字符串在 SAM 上的位置

出现次数就是 $parent$ 树上对应节点的 $size$，线段树单点修改，子树查询维护一下就好了.

subtask2

和 $subtask3$ 不同的是，SAM 的形态不确定了

用 LCT 维护一下 $parent$ 树的形态，动态维护一下节点的 $size$ 就好了.

subtask4

把 $subtask2,3$ 结合一下.

考虑动态维护 bfs 序构造的 SAM 树，实际上只需要在原来的 $insert$ 的基础上讨论一下即可.

在 $len[p]+1<len[q]$ 时，如果 $len[np]==len[p]+1$，我们就不需要新建节点 nq 了.

直接把 $fa[np]=fa[q],fa[q]=np$ 就好了，也就是把 np 插在 p,q 之间，这样就相当于是维护了 bfs 序了.

在 $subtask2$ 动态维护 $parent$ 树的情况下按照上述维护一下 bfs 序，然后按照 $subtask3$ 那样回答询问就好了.