题目大意

定义一个字符串是好串，当且仅当它本身是它的最小表示，即字符串 $s = s_1 s_2 \cdots s_n$ 是好串当且仅当 $\forall 2 \le i \le n$，字符串 $t = s_i s_{i+1} s_{i+2} \cdots s_n s_1 s_2 \cdots s_{i-1}$ 的字典序不小于 s。

给定一个仅有小写字母组成的字符串 S，将 S 中的字母重新排列成一个好的字符串，并最大化这个字符串的字典序。输出重排后的字符串。

数据范围

对于所有数据，保证 $1 \le |S| \le 10^6$。

子任务编号	特殊限制	分值
1	$\lvert S\rvert \le 10$	10
2	$\lvert S\rvert \le 50, S_i \in \{\texttt{a}, \texttt{b}, \texttt{c}\}$	20
3	$\lvert S\rvert \le 1000$	30
4	无特殊限制	40

解题过程

Sol 1

枚举 S 所有字母的全排列构成的字符串 T，$O(|T|)$ 求解 T 的最小表示并判断 T 是否是好串，输出字典序最大的好串 T。令 $n = |S|$，则复杂度为 $O(n! \cdot n)$，可以通过 sub1，获得 10 分。

Sol 2

考虑从前往后按位确定答案 T 的每一位，则转化为 $O(n)$（这里认为 $|\Sigma|$ 是常数）次判定问题：给定一个 S 的重排列 U，判定是否存在重排列 T，满足 T 是好串且 $T \ge U$。

设 $S[:i]$ 表示 S 的前 i 个字符组成的前缀，$S[i:]$ 表示 S 的后 $|S| - i + 1$ 个字符组成的后缀，加法表示字符串拼接。则判定问题中 T 的存在性等价于满足该条件的重排列 P 的存在性：$\forall 1 \le i \le n, P[i:] + U[:i-1] \ge U$。这是因为：

- 如果存在符合条件的 T，容易发现
  $T[i:]+T[:i-1]\geq T, T\geq U \implies T[i:]+U[:i-1]\geq U$，则 T 就是一个符合条件的 P。
- 如果存在符合条件的 P，由于 $i=1$ 时得出 $P\geq U$，则有
  $\forall 1\leq i\leq n, P[i:]+P[:i-1]\geq P[i:]+U[:i-1]\geq U$，因此，$P$ 的最小表示是一个符合条件的 T。

在 S 只由 a, b, c 组成时，只需要设 $f_{x,y,z}$ 表示由 x 个 a，y 个 b，z 个 c 组成的字典序最大的字符串 Q，满足
$\forall 1\leq i\leq x+y+z, Q[i:]+U[:n-(x+y+z-i+1)]\geq U$，也就是可以作为某个合法的 P 的后缀。通过枚举串的第一个字母来转移。原判定问题相当于判定 $f_{A,B,C}$ 是否存在，其中 A, B, C 分别表示 S 中 a, b, c 的数量。

上述 DP 的状态数为 $O(n^3)$，转移为 $O(1)$，但涉及到字符串的拼接、比较大小等，故需要 $O(n)$ 的复杂度，又有 $O(n)$ 次判定，总复杂度为 $O(n^5)$，结合 Sol 1 可以通过 sub1, 2，获得 30 分。

Sol 3

考虑贪心，不妨设 S 中最小的字母为 a，则答案 T 必然以 a 开头（即 $T_1=$ a），尝试最大化 T_2 的值，令 S 中有 c_{a} 个 a，则找到第 c_{a} 大的字母放到 T_2，如果该字母就是 S 中最大的字母，则所有 a 后都会绑定一个 z，否则，则可能绑定其他稍大的字母，一直进行下去可以得到一个解。

可以如此解释这种贪心过程：设 $f(s_1, s_2, \ldots, s_m)$ 表示将字符串 $s_1, s_2, \ldots, s_m$ 重排列后拼接成一个好串，所能得到的字典序最大值，原问题答案即为 $f(S_1, S_2, \ldots, S_n)$。

不妨设 $s_1=s_2=\cdots=s_k<s_{k+1}\leq s_{k+2}\leq\cdots\leq s_m$，则 f 的值有四种可能性：

1. 若 $k=m$，则该值等于 $s_1+s_2+\cdots+s_m$
2. 若 $s_{m-k+1}=s_m$，则该值等于
   $f(s_1+s_m, s_2+s_m, \ldots, s_k+s_m, s_{k+1}, s_{k+2}, \ldots, s_{m-k})$
3. 若 $m-k+1>k$ 且 $s_{m-k+1}<s_m$，则选定 $k<t_1<t_2<\cdots<t_k\leq m$，满足 $s_{t_i}=s_{m-k+1}$，该值等于
   $\max f(s_1+s_{t_1}, s_2+s_{t_2}, \ldots, s_k+s_{t_k}, s_{k+1}, \ldots, s_x(x\neq t_y))$。
4. 若 $m-k+1\leq k<m$，则与上一情况类似，不过 t 可以有一个前缀为空串。

可以证明，第三种情况的最优解是 $t_i=m-k+i$（第四种情况也类似）：

考虑归纳，若已知 $m_0 < m$ 时最优策略是如此，当选用的 t_i 不全是 $m-k+i$ 时，每有一个 x 满足 $m-k+1 \le x \le m$ 但没有在 t 中出现，递归下去的 f 的参数中的 k 值必然会恰好增加 1，而由于归纳假设，在这次递归中，必然会把这些 x 接到多出来的 k 上，得到一些形如 $s_a+s_b+s_c$ 的串，其中 $s_a < s_b < s_c$。

观察 f 的定义可以发现，其参数 s_i 始终是好串，此时可以发现 $s_a+s_b+s_c \le s_a+s_c+s_b$，因为如果 s_b 不是 s_c 一个前缀，则不等式显然，否则该不等式等价于在比较 s_c 的某个循环位移与 s_c 的大小，也显然成立。

由此可知，如果把 $s_a+s_b+s_c$ 都替换为 $s_a+s_c+s_b$，则得到的答案一定更大，因为对于两个串 $f(s_1,\dots,s_m), f(s'_1,\dots,s'_m)$ 满足 $s_i \le s'_i$ 时，在前者的基础上每次把最后一次出现的 s_i 替换成 s'_i，就能得到一个由 $s'_{1\sim m}$ 的重排列组成的更大的好串，且不优于后者。

而又注意到 $f(s_1,\dots,s_m) \le f(s'_1,\dots,s'_m,t_1,t_2,\dots,t_q)$，满足存在一个值域 $1\sim m$ 的大小为 q 的集合 $\{p_1,p_2,\dots,p_q\}$，当 $k=p_i$ 时 $s_k=s'_k+t_i$，否则 $s_k=s'_k$，这是因为后者能重排列出的字符串集合包含前者的。因此，把 $s_a+s_c+s_b$ 这些串拆成 s_a+s_c, s_b 肯定更优，而这恰对应着令 $t_i=m-k+i$ 时递归下去的 f。

直接模拟，使用 `multiset` 等 stl 或平衡树等数据结构维护 s_i 的顺序，由于每次拼接都会让 m 的大小减小 1，因此只会进行 $n-1$ 次拼接，复杂度为 $O(n^2\log n)$，结合 Sol2 可以通过 sub1, 2, 3，获得 60 分。

Sol 4

Sol 3 的瓶颈在于比较字符串大小，而拼接可以用链表做到 $O(1)$ 的复杂度。一开始的 s 全都是单个字母，而第一轮拼接过后，每个字符串要么是单个字母（而且要么全是 a，要么全是严格大于 a 的字母），要么是一个以 a 开头，以非 a 字母结尾的长度为 2 的串，这里假设 a 为串中最小的字母。

设 rk_i 表示 s_i 在 m 个串中的相对排名，则可以发现，若要比较两个不同的由 $s_{1\sim m}$ 拼接出的串 $s_{a_1}+s_{a_2}+\cdots+s_{a_p}$ 和 $s_{b_1}+s_{b_2}+\cdots+s_{b_q}$，只需要比较两个整数序列 $[rk_{a_1},rk_{a_2},\dots,rk_{a_p}]$ 和 $[rk_{b_1},rk_{b_2},\dots,rk_{b_q}]$ 的字典序大小，因为找到最小的 i 使得 $rk_{a_i}\ne rk_{b_i}$，不妨设 $rk_{a_i}<rk_{b_i}$，若 s_{a_i} 不是 s_{b_i} 的前缀，则显然能反映两个字符串的大小，否则说明 $s_{a_i}=$ a，所有 s 均以 a 开头，而 s_{b_i} 的第二个字母大于 a，从而大于 $s_{a_{i+1}}$ 的首个字母（要么是 a，要么不存在），也能正确反映两个串的大小。在此基础上，判断一个串 $s_{a_1}+s_{a_2}+\cdots+s_{a_p}$ 是否是好串，也仅需判断序列 $[rk_{a_1},rk_{a_2},\dots,rk_{a_p}]$ 是否是好串。

因此，在每一轮拼接之后，把每个串替换为它们的相对排名得到的是等价问题，这样做就能绕开字符串的大小比较。根据实现方式的不同，可以做到 $O(n \log n)$ 或 $O(n)$，可以通过所有子任务，获得 100 分。

参考资料

Sol 2 参考自题目 CODE FESTIVAL 2017 qual B F - Largest Smallest Cyclic Shift 的题解（https://img.atcoder.jp/code-festival-2017-qualb/editorial.pdf）。