NOIP 2019 模拟赛 Contest 7

diamond_duke

题目名称	字符串	序列	交换
可执行文件名	str	seq	swap
输入文件名	标准输入	标准输入	标准输入
输出文件名	标准输出	标准输出	标准输出
时间限制	1s	1s	1s
内存限制	512MB	512MB	512MB
子任务个数	4	5	6
题目类型	传统型	传统型	传统型

请注意： 评测时开启 O2 优化和 C++11 编译选项，栈空间限制同空间限制。

1 字符串

考虑如何判断一个字符串 S 是不是另一个字符串 T 的子序列。

一个自然的想法是贪心：我们按照从前往后的顺序考虑 S 的每个字母，然后维护一个 j，表示 $S_1S_2\cdots S_i$ 当前的字符已经匹配到了 $T_1T_2\cdots T_j$。然后每次贪心选择下一个使得 $T_{j'}=S_{i+1}$ 的 j' 匹配过去的。

那么对于这个问题，我们考虑 DP 时记录两个序列对应的 j，那么转移时，预处理出每个字符串每个位置后面第一个 `0/1` 在哪里即可做到 $\Theta(1)$ 转移。时间复杂度：$\Theta(n^2)$。

本题的难点在于如何输出字典序最小的方案。考虑从中止状态出发反向遍历，即可得到哪些状态在“起点到终点的最短路”上（我们把 DP 状态看成图论中的节点，那么 DP 结果就相当于最短路了）。

我们考虑从起始状态出发，枚举下一位填什么，如果能到的状态在最短路上，那么就走过去，否则枚举下一个即可构造出来了。

时间复杂度：$\Theta(n^2)$。

2 序列

显然每个数字的出现次数应该都是 2^t-1 的形式，要不然把多出来的那些换成别的数字一定不会变差。

那么我们考虑有了这个性质之后怎么做。考虑一个贪心做法：我们有一个把每个数字的出现次数从当前的 2^t-1 加到 $2^{t+1}-1$ 的代价，然后这个会使得答案变大 1。因此，我们可以每次选择一个最小的，然后把它的出现次数改一下。

那么我们可以发现，我们选择的这些一定代价都不超过某个数字 M，因此我们考虑二分 M，然后计算出此时需要的总和是多少。这个计算是容易的：因为 $2^t\le M$ 的 t 只有 $\Theta(\log_2 N)$ 个，因此可以直接计算。

那么我们在二分得到答案的 M 后，直接再做一遍同样的操作即可。注意恰好等于 M 的那些可能会选一部分，而非全选。

时间复杂度：$\Theta(T\log_2^2 N)$。

3 交换

首先假设所有数字互不相同，那么我们按照从小到大的顺序考虑每个数字。

可以发现根据题目要求，它要么放在当前序列的开头，要么放在末尾。

考虑放在开头以及结尾的代价，可以发现如果我们把交换两个数字的代价在较小的那个数字计算，那么这样总代价就是前面以及后面没有被放过去的数字个数之和。

那么根据这个分析，可以发现我们某个数字无论放在前面还是后面，他对于后面的数字影响都是一样的——因为代价只与没有放过去的个数有关。因此，我们可以直接贪心选择放在前面还是后面，于是我们使用树状数组即可求出放在两个位置的代价，然后贪心放过去即可。

时间复杂度：$\Theta(n \log_2 n)$。

那么还有一个问题：就是如果有一样的数字，那么有可能在他们被交换的时候我们也会计算代价，而这次是可以省掉的。考虑怎么办。

我们注意到同一个数字的所有出现中，一定不会有前面的数字被放到后面，因为这样反过来放显然会更优。因此，对于同一个数字的所有出现，我们可以考虑类似归并：每次把第一个数放到最前面的代价，以及最后一个数字放到最后面的代价算出来，然后哪个小就放哪个即可。

时间复杂度：$\Theta(n \log_2 n)$。