水题分享：luoguP5606 - 小 K 与毕业旅行

jeorme_wei

cdqz

给一个**整数**序列 $a_1 \cdots a_n$ 和一个**正整数**权值 w。求排列 p 满足以下条件的个数：

$$\forall 2 \le i \le n, a_{p_i} * a_{p_{i-1}} \le w$$

对 998244353 取模。
对于所有数据，$0 \le w, |a_i| \le 10^9$，$1 \le n \le 50000$。

给一个**整数**序列 $a_1 \cdots a_n$ 和一个**正整数**权值 w。求排列 p 满足以下条件的个数：

$$\forall 2 \leq i \leq n, a_{p_i} * a_{p_{i-1}} \leq w$$

对 998244353 取模。
对于所有数据，$0 \leq w, |a_i| \leq 10^9$，$1 \leq n \leq 50000$。
可以思考以下几个部分分：
subtask 6 ：$n \leq 2000$，$a_i \geq 0$
subtask 7：$n \leq 50000$，$a_i \geq 0$
subtask 9 ：$n \leq 2000$
subtask 10 ：$n \leq 50000$

因为限制需要考虑到正负号，考虑先解决 $a_i \geq 0$ 的部分的 n^2 做法。
考虑用 dp 解决这个问题，按照一定顺序依次考虑其位置。

考虑到如果设计一个插入的方案，满足插入一个值的时候，它前面的所有值要么全部能与当前值相邻，要么全不能与当前值相邻，那么看起来可以设法设计一个 dp。

形式化地说，需要重拍序列 a 使得所有 a_i 满足以下条件之一：

- $\forall_{j<i}, a_i * a_j \leq w$，称满足这类数的为 A 类
- $\forall_{j<i}, a_i * a_j > w$，称满足这类数的为 B 类

实际上这个序列是可以比较轻松地构造的：

实际上这个序列是可以比较轻松地构造的：
排序之后考虑最小的数和最大的数，设为 a_1, a_n 。
如果 $a_1 * a_n \leq w$，则 a_1 与所有数都可以匹配，将它插入到当前插入顺序的头。

实际上这个序列是可以比较轻松地构造的：
排序之后考虑最小的数和最大的数，设为 a_1, a_n 。
如果 $a_1 * a_n \leq w$，则 a_1 与所有数都可以匹配，将它插入到当前插入顺序的头。
反之，则 a_n 与所有数都不能匹配，将它插入到当前插入顺序的头。
然后递归到子问题即可。

有了这个序列，不难想到如下 dp：
$f[i][j]$ 表示考虑了序列的前 i 个数，前 i 个数在序列中构成 j 个极长段的方案数。
每次的转移有三种：
1. 插入一个新的连续段
2. 合并两个相邻连续段
3. 一个连续段的两边延长

插入一个 A 类则三种转移都是可以的，否则只能做第一种转移。

有了上述做法，subtask 7 也就迎刃而解：
因为大于 0 和小于 0 是一定可以相邻的，对 ≥ 0 和 < 0 的部分分别跑上述 dp，最后 $O(n)$ 合并即可。

这个 dp 不是很能优化，考虑换一种插入方式：

这个 dp 不是很能优化，考虑换一种插入方式：
将重排的 a 翻转后依次插入。

这个 dp 不是很能优化，考虑换一种插入方式：
将重排的 a 翻转后依次插入。
这个时候的性质是：
现在的性质是：

- $\forall_{j>i}, a_i * a_j \leq w$，称满足这类数的为 A 类
- $\forall_{j>i}, a_i * a_j > w$，称满足这类数的为 B 类

变成大于之后，可以发现一个关键的性质：如果一个位置和当前插入的数不能相邻，那么这个位置就再也不能和之后插入的数相邻。
有什么用呢?

变成大于之后，可以发现一个关键的性质：如果一个位置和当前插入的数不能相邻，那么这个位置就再也不能和之后插入的数相邻。
有什么用呢?
考虑插入的时候将其插入在两个位置中间，与这两个位置相邻。
由上述结论可以知道：如果一个序列最终合法，那么在其每次插入的时候都是合法的。

那么考虑每个时刻有多少个可以插入的位置：初始为 1 个。
插入一个 A 类那么这个位置左右都仍然能够插入。（可插入位置 $+1$）
插入一个 B 类那么这个位置左右都不能再插入。（可插入位置 -1）
于是得到了一个线性的做法。

会了 subtask 9，那么最后一个 subtask 就很简单了。

会了 subtask 9，那么最后一个 subtask 就很简单了。
仍然考虑像 subtask 7 那样处理最后有 i 段的方案。
我们可以考虑初始插入的位置设置为 i 个的方案，这样可以得到最多 i 段的方案数（有一些段可以是空）。

会了 subtask 9，那么最后一个 subtask 就很简单了。
仍然考虑像 subtask 7 那样处理最后有 i 段的方案。
我们可以考虑初始插入的位置设置为 i 个的方案，这样可以得到最多 i 段的方案数（有一些段可以是空）。
不难发现这是关于 i 的多项式，多点求值即可。
容斥也可以发现是一个卷积。

会了 subtask 9，那么最后一个 subtask 就很简单了。
仍然考虑像 subtask 7 那样处理最后有 i 段的方案。
我们可以考虑初始插入的位置设置为 i 个的方案，这样可以得到最多 i 段的方案数（有一些段可以是空）。
不难发现这是关于 i 的多项式，多点求值即可。
容斥也可以发现是一个卷积。
最后的复杂度就是 $O(n \log^2 n)$。常数较大。