MX-J27 Solution

沉石鱼惊旋

October 25, 2025

Outline

Outline

题意

- 给定 n，询问有多少个 $1 \le x \le n$ 满足 $\lfloor \sqrt{x} \rfloor$ 是 x 的因子。

题意

- 给定 n，询问有多少个 $1 \le x \le n$ 满足 $\lfloor \sqrt{x} \rfloor$ 是 x 的因子。
- q 次询问，$1 \le n \le 10^{18}$，$1 \le q \le 10^5$。

部分分 40

- 对于一次询问，我们肯定可以直接暴力遍历所有的 $1 \le x \le n$ 进行判断。

部分分 40

- 对于一次询问，我们肯定可以直接暴力遍历所有的 $1 \le x \le n$ 进行判断。
- 可以预处理，先把 1 到 10^6 都扫一遍，然后可以知道每个 $1 \le n \le 10^6$ 的答案。

部分分 40

- 对于一次询问，我们肯定可以直接暴力遍历所有的 $1 \le x \le n$ 进行判断。
- 可以预处理，先把 1 到 10^6 都扫一遍，然后可以知道每个 $1 \le n \le 10^6$ 的答案。
- 把答案存下来，每次询问即可 $\mathcal{O}(1)$ 回答。

部分分 40

- 对于一次询问，我们肯定可以直接暴力遍历所有的 $1 \le x \le n$ 进行判断。
- 可以预处理，先把 1 到 10^6 都扫一遍，然后可以知道每个 $1 \le n \le 10^6$ 的答案。
- 把答案存下来，每次询问即可 $\mathcal{O}(1)$ 回答。
- 时间复杂度 $\mathcal{O}(n+q)$。

部分分 80

- 这一档分虽然 $\mathcal{O}(n)$ 做不了了，但是 $\mathcal{O}(\sqrt{n})$ 是可行的。

部分分 80

- 这一档分虽然 $\mathcal{O}(n)$ 做不了了，但是 $\mathcal{O}(\sqrt{n})$ 是可行的。
- 令 $k = \lfloor \sqrt{x} \rfloor$，我们可以枚举 k。

部分分 80

- 这一档分虽然 $\mathcal{O}(n)$ 做不了了，但是 $\mathcal{O}(\sqrt{n})$ 是可行的。
- 令 $k=\lfloor\sqrt{x}\rfloor$，我们可以枚举 k。
- 对于一个 k，考虑它会给哪些 x 产生贡献。

部分分 80

- 这一档分虽然 $\mathcal{O}(n)$ 做不了了，但是 $\mathcal{O}(\sqrt{n})$ 是可行的。
- 令 $k=\lfloor\sqrt{x}\rfloor$，我们可以枚举 k。
- 对于一个 k，考虑它会给哪些 x 产生贡献。
- 注意到一定是形如 $x=k(k+c)$ 的形式才会被 k 带来贡献。

部分分 80

- 这一档分虽然 $\mathcal{O}(n)$ 做不了了，但是 $\mathcal{O}(\sqrt{n})$ 是可行的。
- 令 $k=\lfloor\sqrt{x}\rfloor$，我们可以枚举 k。
- 对于一个 k，考虑它会给哪些 x 产生贡献。
- 注意到一定是形如 $x=k(k+c)$ 的形式才会被 k 带来贡献。
- 具体的，由于 $k(k+3)=k^2+3k$ 而 $(k+1)^2=k^2+2k+1$，因为有 $k\ge 1$ 所以 $k(k+3)\ge(k+1)^2$。

部分分 80

- 也就是说，$\lfloor\sqrt{k(k+3)}\rfloor \neq k$。

部分分 80

- 也就是说，$\lfloor\sqrt{k(k+3)}\rfloor \neq k$。
- 因此，枚举 k，计算多少个 $x=k^2, x=k(k+1), x=k(k+2)$ 在 $[1,n]$ 之间。

部分分 80

- 也就是说，$\lfloor\sqrt{k(k+3)}\rfloor \neq k$。
- 因此，枚举 k，计算多少个 $x=k^2, x=k(k+1), x=k(k+2)$ 在 $[1,n]$ 之间。
- 时间复杂度 $\mathcal{O}(q\sqrt{n})$。

满分 100

- 80 分的做法，存在一个关键性质：只有 $x = k^2, x = k(k+1), x = k(k+2)$ 这三类形式的 x。

满分 100

- 80 分的做法，存在一个关键性质：只有 $x=k^2, x=k(k+1), x=k(k+2)$ 这三类形式的 x。
- 而显然，如果存在 $x=k^2$，则 $[1,n]$ 一定存在 $x=(k-1)^2$。其他两种同理。

满分 100

- 80 分的做法，存在一个关键性质：只有 $x = k^2, x = k(k+1), x = k(k+2)$ 这三类形式的 x。
- 而显然，如果存在 $x = k^2$，则 $[1, n]$ 一定存在 $x = (k-1)^2$。其他两种同理。
- 因此，我们只关心最大的 $x = k^2$ 的 k 是多少。剩下的 $1 \le k' \le k$ 都一定合法。

满分 100

- 直接计算 $m=\sqrt{n}$，则形如 $x=k^2$ 的有 m 个。

满分 100

- 直接计算 $m=\sqrt{n}$，则形如 $x=k^2$ 的有 m 个。
- 同理可以计算出 $x=k(k+1)$ 和 $x=k(k+2)$ 形式的数的个数。

满分 100

- 直接计算 $m=\sqrt{n}$，则形如 $x=k^2$ 的有 m 个。
- 同理可以计算出 $x=k(k+1)$ 和 $x=k(k+2)$ 形式的数的个数。
- 具体实现在 m 周围往下枚举若干个，即可知道精确的 $x=k(k+1)$ 的数的个数，或使用二分搜索同理。

满分 100

- 直接计算 $m=\sqrt{n}$，则形如 $x=k^2$ 的有 m 个。
- 同理可以计算出 $x=k(k+1)$ 和 $x=k(k+2)$ 形式的数的个数。
- 具体实现在 m 周围往下枚举若干个，即可知道精确的 $x=k(k+1)$ 的数的个数，或使用二分搜索同理。
- 实现的时候，如果需要开根号，想要调用 `sqrt(n)`，需要注意，对于 `long long` 类型应该使用 `sqrtl(n)`，否则会产生精度误差导致挂分。这是因为 `sqrt()` 只在 `double` 精度范围内，而 `sqrtl()` 在 `long double` 精度范围内。`double` 的精度范围内无法精确表示 `long long` 类型。

MX-J27 Solution

沉石鱼惊旋

T1 分块
T2 转换
T3 旋律
T4 点灯
鲜花 & 抽奖

满分 100

- 直接计算 $m=\sqrt{n}$，则形如 $x=k^2$ 的有 m 个。
- 同理可以计算出 $x=k(k+1)$ 和 $x=k(k+2)$ 形式的数的个数。
- 具体实现在 m 周围往下枚举若干个，即可知道精确的 $x=k(k+1)$ 的数的个数，或使用二分搜索同理。
- 实现的时候，如果需要开根号，想要调用 `sqrt(n)`，需要注意，对于 `long long` 类型应该使用 `sqrtl(n)`，否则会产生精度误差导致挂分。这是因为 `sqrt()` 只在 `double` 精度范围内，而 `sqrtl()` 在 `long double` 精度范围内。`double` 的精度范围内无法精确表示 `long long` 类型。
- 依据实现时间复杂度为 $\mathcal{O}(q)$ 或 $\mathcal{O}(q\log n)$。

Outline

题意

- 给定运算表达式，计算表达式返回值的**类型**。具体运算规则与 C++ 运算规则相同。

题意

- 给定运算表达式，计算表达式返回值的**类型**。具体运算规则与 C++ 运算规则相同。
- 保证类型只在 {`char`,`bool`,`int`,`longlong`,`float`,`double`} 中；保证运算符只在 {`+`,`*`,`,`} 中。

题意

- 给定运算表达式，计算表达式返回值的**类型**。具体运算规则与 C++ 运算规则相同。
- 保证类型只在 $\{$`char`,`bool`,`int`,`longlong`,`float`,`double`$\}$ 中；保证运算符只在 $\{$`+`,`*`,`,`$\}$ 中。
- $1 \le n \le 10^5$，n 是表达式中的运算符数量。

部分分 30

- 暴力模拟运算即可。每一次找到优先级最高的运算符计算。

部分分 30

- 暴力模拟运算即可。每一次找到优先级最高的运算符计算。
- 时间复杂度 $\mathcal{O}(n^2)$。

特殊性质

- 特殊性质都是简化模拟操作的，以及防止写挂爆零的，实际上没有太大启发。

满分 100

- 不妨先对字符串做一些简化。

满分 100

- 不妨先对字符串做一些简化。
- 因为涉及逗号运算只看最后一项，所以可以找到最后一次出现的逗号在位置 p，将 s 截断为 $s \leftarrow s[p+1:n]$。

满分 100

- 不妨先对字符串做一些简化。
- 因为涉及逗号运算只看最后一项，所以可以找到最后一次出现的逗号在位置 p，将 s 截断为 $s \leftarrow s[p+1:n]$。
- 类型太长了，长度也不统一。我们只取他们的首字母表示。

满分 100

- 考虑表达式计算的做法。

满分 100

- 考虑表达式计算的做法。
- 我们维护一个栈。栈里面存的是若干个类型和操作符。

满分 100

- 考虑表达式计算的做法。
- 我们维护一个栈。栈里面存的是若干个类型和操作符。
- 由于乘法优先级高，所以先做一遍乘法计算。

满分 100

- 考虑表达式计算的做法。
- 我们维护一个栈。栈里面存的是若干个类型和操作符。
- 由于乘法优先级高，所以先做一遍乘法计算。
- 我们把元素压入栈的时候，如果目前栈顶是乘号，再弹出栈顶第二个元素进行计算，把计算结果压回栈内。

满分 100

- 考虑表达式计算的做法。
- 我们维护一个栈。栈里面存的是若干个类型和操作符。
- 由于乘法优先级高，所以先做一遍乘法计算。
- 我们把元素压入栈的时候，如果目前栈顶是乘号，再弹出栈顶第二个元素进行计算，把计算结果压回栈内。
- 做完上述操作之后，重新扫一遍栈，再做加法运算即可。

满分 100

- 对于操作符的计算，直接模拟即可。

满分 100

- 对于操作符的计算，直接模拟即可。
- 先把 `bool` 和 `char` 全部改成 `int`。

满分 100

- 对于操作符的计算，直接模拟即可。
- 先把 `bool` 和 `char` 全部改成 `int`。
- 如果操作同时含有浮点类和整数类，返回那个浮点类的类型。

满分 100

- 对于操作符的计算，直接模拟即可。
- 先把 `bool` 和 `char` 全部改成 `int`。
- 如果操作同时含有浮点类和整数类，返回那个浮点类的类型。
- 否则，此时两个数类型（浮点 or 整数）一样，返回占用字节数更多的那个。

满分 100

- 直接模拟这个压栈弹栈的时间复杂度就是 $\mathcal{O}(n)$ 的。

满分 100

- 直接模拟这个压栈弹栈的时间复杂度就是 $\mathcal{O}(n)$ 的。
- 另外本题可能出现 `expr,bool` 的情况，所以你**不应该**在一开始就把所有 `bool` 和 `char` 全部改成 `int`。

还没完

- 有一个更巧妙的观察：乘法和加法无非是运算顺序不一样，但是具体的计算返回类型是一样的。

还没完

- 有一个更巧妙的观察：乘法和加法无非是运算顺序不一样，但是具体的计算返回类型是一样的。
- 也就是我们根本不需要区分乘法和加法，全部看为加法进行操作即可，不需要扫两次。

还没完

- 有一个更巧妙的观察：乘法和加法无非是运算顺序不一样，但是具体的计算返回类型是一样的。
- 也就是我们根本不需要区分乘法和加法，全部看为加法进行操作即可，不需要扫两次。
- 以及，我们可以看作是，每个类型有一个权值，每次取权值最大的。这些运算符全部等价于取 $\max$。

还没完

- 有一个更巧妙的观察：乘法和加法无非是运算顺序不一样，但是具体的计算返回类型是一样的。
- 也就是我们根本不需要区分乘法和加法，全部看为加法进行操作即可，不需要扫两次。
- 以及，我们可以看作是，每个类型有一个权值，每次取权值最大的。这些运算符全部等价于取 $\max$。
- 当然作为一道 T2 级别的模拟题，怎么写都可以，在此只是讲一些实现的小技巧。

Outline

1. T1 分块
2. T2 转换
3. T3 旋律
4. T4 点灯
5. 鲜花 & 抽奖

题意

- 给定长度为 $a_1, a_2, \ldots, a_n$，选出一个非空子序列 $b_1, b_2, \ldots, b_m$，最大化 $mk - (\max\{b\} - \min\{b\})$。

题意

- 给定长度为 $a_1, a_2, \dots, a_n$，选出一个非空子序列 $b_1, b_2, \dots, b_m$，最大化 $mk - (\max\{b\} - \min\{b\})$。
- $1 \le n \le 10^5$，$1 \le a_i, k \le 10^8$。

部分分 20

- 枚举子序列，依据实现可以做到 $\mathcal{O}(2^n + n\log n)$ 或 $\mathcal{O}(2^n n + n\log n)$。

特殊性质

- 特殊性质 A，比较公差 d 和 k，决策是只选一种数，还是全选。

特殊性质

- 特殊性质 A，比较公差 d 和 k，决策是只选一种数，还是全选。
- 特殊性质 B，可以证明全选是最优解，因为我们加一个数有 10^8 的收益，但是极差的增量不超过 10^8。

特殊性质

- 特殊性质 A，比较公差 d 和 k，决策是只选一种数，还是全选。
- 特殊性质 B，可以证明全选是最优解，因为我们加一个数有 10^8 的收益，但是极差的增量不超过 10^8。
- 特殊性质 C，引导选手往正解方向思考，无较优的简单做法。

部分分 55

- 由于和极差相关，想到把 a 排序，排序后可以证明我们一定选连续区间，则答案就是最大的 $(r-l+1)k-(a_r-a_l)$。

部分分 55

- 由于和极差相关，想到把 a 排序，排序后可以证明我们一定选连续区间，则答案就是最大的 $(r-l+1)k-(a_r-a_l)$。
- 固定 l，枚举 r，则复杂度为 $\mathcal{O}(n^2)$。

满分 100（算法 1）

- 我们把上面的式子 $(r-l+1)k-(a_r-a_l)$ 打开。

满分 100（算法 1）

- 我们把上面的式子 $(r-l+1)k-(a_r-a_l)$ 打开。
- 得到 $k+(rk-a_r)-(lk-a_l)$。

满分 100（算法 1）

- 我们把上面的式子 $(r-l+1)k-(a_r-a_l)$ 打开。
- 得到 $k+(rk-a_r)-(lk-a_l)$。
- 这个形式告诉我们，我们只需要知道最大和最小的 $ik-a_i$。

满分 100（算法 1）

- 我们把上面的式子 $(r-l+1)k-(a_r-a_l)$ 打开。
- 得到 $k+(rk-a_r)-(lk-a_l)$。
- 这个形式告诉我们，我们只需要知道最大和最小的 $ik-a_i$。
- 排序后扫一遍，存一下当前最小的 $\mathrm{Min}=ik-a_i$，将答案 Ans 和目前的 $jk-a_j-\mathrm{Min}+k$ 取最大值。

满分 100（算法 2）

- 对增量考虑。

满分 100（算法 2）

- 对增量考虑。
- 对 a 排序之后，我们每一次极差的增量是 $a_i - a_{i-1}$，但是可以获得 k 的收益，所以是 $k - (a_i - a_{i-1})$。

满分 100（算法 2）

- 对增量考虑。
- 对 a 排序之后，我们每一次极差的增量是 $a_i - a_{i-1}$，但是可以获得 k 的收益，所以是 $k - (a_i - a_{i-1})$。
- 我们其实就是要找到一个最大的区间，使得区间里的增量之和最大。

满分 100（算法 2）

- 对增量考虑。
- 对 a 排序之后，我们每一次极差的增量是 $a_i - a_{i-1}$，但是可以获得 k 的收益，所以是 $k-(a_i-a_{i-1})$。
- 我们其实就是要找到一个最大的区间，使得区间里的增量之和最大。
- 这是典型的最大子段和问题，直接贪心或者线性 DP 均可。

满分 100

- 两种做法的时间复杂度都是 $\mathcal{O}(n \log n)$，瓶颈在排序上。

满分 100

- 两种做法的时间复杂度都是 $\mathcal{O}(n \log n)$，瓶颈在排序上。
- 本题几乎不存在正常写代码过了样例挂了的。存在部分选手因为拼暴力暴力写错了而挂分，正式赛场上的对拍是很重要的！

Outline

题意

- 给定 n 个点 m 条边的无向图，每条边在特定的一段**后缀**时间 $[w, +\infty)$ 激活。

题意

- 给定 n 个点 m 条边的无向图，每条边在特定的一段**后缀**时间 $[w,+\infty)$ 激活。
- 初始在点 1 有**充分多**的人，之后每个时刻所有人**必须**通过一条已经被激活的边移动到相邻点。

题意

- 给定 n 个点 m 条边的无向图，每条边在特定的一段**后缀**时间 $[w, +\infty)$ 激活。
- 初始在点 1 有**充分多**的人，之后每个时刻所有人**必须**通过一条已经被激活的边移动到相邻点。
- 求最早时刻使得所有点都有人，或报告无解。

题意

- 给定 n 个点 m 条边的无向图，每条边在特定的一段**后缀**时间 $[w, +\infty)$ 激活。
- 初始在点 1 有**充分多**的人，之后每个时刻所有人**必须**通过一条已经被激活的边移动到相邻点。
- 求最早时刻使得所有点都有人，或报告无解。
- $2 \leq n \leq 2.5 \times 10^4$，$n-1 \leq m \leq 5 \times 10^4$，$1 \leq w \leq 10^9$。

题意

- 给定 n 个点 m 条边的无向图，每条边在特定的一段**后缀**时间 $[w,+\infty)$ 激活。
- 初始在点 1 有**充分多**的人，之后每个时刻所有人**必须**通过一条已经被激活的边移动到相邻点。
- 求最早时刻使得所有点都有人，或报告无解。
- $2 \le n \le 2.5 \times 10^4$，$n-1 \le m \le 5 \times 10^4$，$1 \le w \le 10^9$。
- 保证无自环无重边，保证所有边激活后图是连通图。

部分分 8 & 特殊性质 A

- 观察数据范围，此时满足 $n \le 10$ 和 $m \le 20$。

部分分 8 & 特殊性质 A

- 观察数据范围，此时满足 $n \le 10$ 和 $m \le 20$。
- 直接维护每个时刻的可达点，模拟题意。

特殊性质 B

- 如果 $w \neq 1$，则无解，因为时刻一无法进行任何移动。

特殊性质 B

- 如果 $w \neq 1$，则无解，因为时刻一无法进行任何移动。
- 考虑 $w = 1$，此时即为初始图全部连通。

特殊性质 B

- 如果 $w \neq 1$，则无解，因为时刻一无法进行任何移动。
- 考虑 $w = 1$，此时即为初始图全部连通。
- 注意到一个点 u，如果在时刻 t 从 p 走过来访问了，则可以反复在 u, p 之间移动，使得在所有 $t + 2k$ 的时刻都访问到 u。

特殊性质 B

- 如果 $w \neq 1$，则无解，因为时刻一无法进行任何移动。
- 考虑 $w = 1$，此时即为初始图全部连通。
- 注意到一个点 u，如果在时刻 t 从 p 走过来访问了，则可以反复在 u, p 之间移动，使得在所有 $t + 2k$ 的时刻都访问到 u。
- 这启发我们记录每个点在奇数时刻和偶数时刻最早抵达的时间。一个点拆成两个点代表奇偶，直接跑 BFS 即可。

特殊性质 B

- 如果 $w \neq 1$，则无解，因为时刻一无法进行任何移动。
- 考虑 $w = 1$，此时即为初始图全部连通。
- 注意到一个点 u，如果在时刻 t 从 p 走过来访问了，则可以反复在 u, p 之间移动，使得在所有 $t + 2k$ 的时刻都访问到 u。
- 这启发我们记录每个点在奇数时刻和偶数时刻最早抵达的时间。一个点拆成两个点代表奇偶，直接跑 BFS 即可。
- 枚举最后答案是奇数还是偶数，把所有点对应的访问时间取最大的那个。

特殊性质 B

- 如果 $w \neq 1$，则无解，因为时刻一无法进行任何移动。
- 考虑 $w = 1$，此时即为初始图全部连通。
- 注意到一个点 u，如果在时刻 t 从 p 走过来访问了，则可以反复在 u, p 之间移动，使得在所有 $t + 2k$ 的时刻都访问到 u。
- 这启发我们记录每个点在奇数时刻和偶数时刻最早抵达的时间。一个点拆成两个点代表奇偶，直接跑 BFS 即可。
- 枚举最后答案是奇数还是偶数，把所有点对应的访问时间取最大的那个。
- 如果奇数和偶数两种情况，都存在一个点无法在对应类型的时刻到达，也是无解。

特殊性质 C

- 防止你无解情况没判干净或者不会判留的分。

满分 100（算法 1）

- 沿着特殊性质 B 继续思考。唯一的区别就是 w 不同。

满分 100（算法 1）

- 沿着特殊性质 B 继续思考。唯一的区别就是 w 不同。
- 然而考虑这个在一条边反复横跳的过程，我们可以一直等到这个点开通了再走过去。

满分 100（算法 1）

- 沿着特殊性质 B 继续思考。唯一的区别就是 w 不同。
- 然而考虑这个在一条边反复横跳的过程，我们可以一直等到这个点开通了再走过去。
- 也就是若点 u 在 t 时刻抵达了，若到邻居 v 的边激活时间为 w，若 $w \le t+1$ 则 $t+1$ 就可以到 v，否则最早抵达 v 的时间是 $\ge w$ 的第一个和 t 奇偶不同的时刻。

满分 100（算法 1）

- 沿着特殊性质 B 继续思考。唯一的区别就是 w 不同。
- 然而考虑这个在一条边反复横跳的过程，我们可以一直等到这个点开通了再走过去。
- 也就是若点 u 在 t 时刻抵达了，若到邻居 v 的边激活时间为 w，若 $w \leq t+1$ 则 $t+1$ 就可以到 v，否则最早抵达 v 的时间是 $\geq w$ 的第一个和 t 奇偶不同的时刻。
- 使用 Dijkstra 维护上述过程。

满分 100（算法 2）

- 我们把边存下来，按 w 排序。注意一条边 (u, v, w) 可能要拆成 2 条边，多加一条 $(u, v, w+1)$。

满分 100（算法 2）

- 我们把边存下来，按 w 排序。注意一条边 (u, v, w) 可能要拆成 2 条边，多加一条 $(u, v, w + 1)$。
- 我们一边做 BFS，一边随着目前的时刻激活边。

满分 100（算法 2）

- 我们把边存下来，按 w 排序。注意一条边 (u, v, w) 可能要拆成 2 条边，多加一条 $(u, v, w+1)$。
- 我们一边做 BFS，一边随着目前的时刻激活边。
- 激活的转移是基本同上的，但是由于我们只关注被激活的那些边，所以不存在说从时刻 t 突然跳到 w 的情况。一次 BFS 中的时刻都是连续的。

满分 100（算法 2）

- 我们把边存下来，按 w 排序。注意一条边 (u, v, w) 可能要拆成 2 条边，多加一条 $(u, v, w+1)$。
- 我们一边做 BFS，一边随着目前的时刻激活边。
- 激活的转移是基本同上的，但是由于我们只关注被激活的那些边，所以不存在说从时刻 t 突然跳到 w 的情况。一次 BFS 中的时刻都是连续的。
- 我们把连续时刻的做完之后，再重新做 BFS，从新的时刻开始，再同上跑一遍。直到跑完了所有边没出答案就是无解，或者出了答案直接结束搜索。

满分 100（算法 2）

- 我们把边存下来，按 w 排序。注意一条边 (u, v, w) 可能要拆成 2 条边，多加一条 $(u, v, w+1)$。
- 我们一边做 BFS，一边随着目前的时刻激活边。
- 激活的转移是基本同上的，但是由于我们只关注被激活的那些边，所以不存在说从时刻 t 突然跳到 w 的情况。一次 BFS 中的时刻都是连续的。
- 我们把连续时刻的做完之后，再重新做 BFS，从新的时刻开始，再同上跑一遍。直到跑完了所有边没出答案就是无解，或者出了答案直接结束搜索。
- 简单来说，这个算法就是只关心连续的有用时刻段，每个段分别跑 BFS。这个算法是在 Junior 组考纲内的。

满分 100

- 如果是第一种算法，实现可以不显式建图以降低常数因子对程序效率的影响。

满分 100

- 如果是第一种算法，实现可以不显式建图以降低常数因子对程序效率的影响。
- 另外本题 $o = 0$ 的情况下，答案如果是无解，不应该输出 -1×0，而应该输出 -1，存在部分选手没有读清题意在这里挂分的情况。

Outline

鲜花 & 抽奖

祝大家 CSP2025 RP++，考出自己满意的成绩，不负韶光！