模板使用说明

出题人

CONTENTS

记号

本节只是重复下文中要用到的常见的公认的数学概念.

对于一个集合 G 上配有二元运算 $\cdot : G \times G \to G$, 如果满足

- 分配律: 对任意 $a, b, c \in G$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$,
- 交换律: 对任意 $a, b \in G$, $a \cdot b = b \cdot a$,
- 单位元的存在性: 有 $1 \in G$ 使得对任意 $g \in G$ 有 $1 \cdot g = g \cdot 1$,
- 逆元的存在性: 对任意 $g \in G$ 有 $g^{-1} \in G$ 使得 $g \cdot g^{-1} = g^{-1} \cdot g = 1$,

称 G 是**交换群 (commutative group)**. 进一步的, 如果一个集合 F 上装配了交换群 $+$ (加法), 记其单位元为 0, 并且 $F \setminus \{0\}$ 上装配了交换群 $\cdot$ (乘法, 运算时可省略该符号), 记单位元为 1, 还满足

- 对任意 $a \in F$, $0 \cdot a = a \cdot 0 = 0$,
- 分配律: 对任意 $a, b, c \in G$, $a \cdot (b + c) = ab + ac$,

称 F 是**域 (field)**.

本题选取的模数为 $p = 998244353$, 下文皆记为 p. 由于这是一个素数, 可知其剩余系上的运算构成域 $\mathbb{F}_p$.

我们记 $\mathbb{F}_p[x]$ 为 $\mathbb{F}_p$ 上的多项式, 也即 $\displaystyle\sum_{\substack{n \ge 0 \\ n\text{有限}}} a_n x^n$ 构成的集合 (不在和式中的项的系数均为 0), 有加法运算

$$\left(\sum_n a_n x^n\right) + \left(\sum_n b_n x^n\right) = \sum_n (a_n + b_n) x^n$$

和乘法运算

$$\left(\sum_n a_n x^n\right)\left(\sum_n b_n x^n\right) = \sum_n \left(\sum_m a_m b_{n-m}\right) x^n.$$

$\mathbb{F}_p[x]/(x^n)$ 为模 x^n 意义下的多项式. 也即对于 $p(x), q(x) \in \mathbb{F}_p[x]$, 说 $p(x) \equiv q(x) \pmod{x^n}$ 当且仅当 p, q 在 $< n$ 次项的系数均相等. 可以知道, 有

- 乘法逆元: 如果 $p(x)$ 的常数项非零, 有 $p(x)q(x) \equiv 1 \pmod{x^n}$, 称 $q(x) = p(x)^{-1}$.
- 平方根: 如果 $p(x)$ 的常数项非零, 有 $q(x)^2 \equiv p(x) \pmod{x^n}$, 称 $q(x) \in \left\{\pm\sqrt{p(x)}\right\}$ 是一个平方根.
- 对数: 如果 $n < p$, 且 $p(x)$ 的常数项是 1, 有

$$\ln p(x) \equiv \sum_{k=1}^{n-1} \frac{(-1)^{k-1}}{k} (p(x) - 1)^k \pmod{x^n}.$$

- 指数: 如果 $n < p$, 且 $p(x)$ 的常数项是 0, 有

$$\exp p(x) \equiv \sum_{k=0}^{n-1} \frac{1}{k!} p(x)^k \pmod{x^n}.$$

使用说明

建议的使用方法是直接复制一份模板代码, 在其基础上对 `main` 函数的内容修改作答.

由于模板代码的一些特性, 请在本机编译时选择 `C++11` 或更高版本进行编译.

除了本说明中提到的内容, 模板内仍有一些宏定义, 以及一些其他的辅助常量, 变量和函数, 请注意规避命名冲突. 我们没有设置措施阻碍你研究模板代码的具体内容, 但这理论上对你做出题目没有帮助.

后续说明中会用**加粗**的方式明确一些使用时的注意事项, 违反这些事项不一定会出现问题, 但将会不保证模板给你预期的结果.

模板的长度约为 20 KB, 提交代码的长度限制为 200 KB.

模板会占用不超过 60 MB 的内存, 本题的内存限制为 512 MB.

$\mathbb{F}_p$ 上的运算

我们在模板中对 $\mathbb{F}_p$ 上的运算进行了封装, 你可以使用类 `mi`, 此外, 模板内有类型为 `int` 的常数 `MOD`, 保证值为 p.

3.1 初始化

`mi` 默认构造函数将数设置为 0, 此外有两个构造函数.

- `mi(int x)`: 当传入参数为 `int` 类型时, 将自动处理取模到 0 到 $p-1$ 的范围内.
- `mi(unsigned x)`: 当传入参数为 `unsigned` 类型时, **不会对参数进行检查**, 请保证 `x` 的数值在 0 到 $p-1$ 的范围内.

3.2 输入与输出

我们对 `istream` 和 `ostream` 的相关部分进行了重载, 因此, 对于类型为 `mi` 的 `x`, 你可以直接通过 `cin >> x` 和 `cout << x` 输入输出一个数.

3.3 基本运算

如非特殊说明, 下面提到的运算, 输入和输出的结果类型均为 `mi`.

- `a + b`: 计算 $a+b$. 时间复杂度 $O(1)$.
- `a - b`: 计算 $a-b$. 时间复杂度 $O(1)$.
- `-b`: 计算 $-b$. 时间复杂度 $O(1)$.
- `a * b`: 计算 ab. 时间复杂度 $O(1)$.
- `a / b`: 计算 a/b. 时间复杂度 $O(\log p)$, **请保证传入的 `b` 非零.**
- `inv(b)`: 计算 b^{-1}. 时间复杂度 $O(\log p)$, **请保证传入的 `b` 非零.**
- `qp(mi a, long long b)`: 计算 a^b, 时间复杂度 $O(\log b)$, **请保证传入的 `b` 非负.**
- `a == b`: 返回一个布尔值, 表示 a 是否等于 b. 时间复杂度 $O(1)$.
- `a != b`: 返回一个布尔值, 表示 a 是否不等于 b. 时间复杂度 $O(1)$.

$\mathbb{F}_p[x]$ 与$\mathbb{F}_p[x]/(x^n)$ 上的运算

我们在模板中对 $\mathbb{F}_p[x]$ 与 $\mathbb{F}_p[x]/(x^n)$ 上的运算进行了封装, 你可以使用类 `poly`.

4.1 计算精度

对于一个类型为 `poly` 的对象 `p`, 有 `p.shrink_len` 表示计算的精度.

- 如果 `p.shrink_len` 是 -1, 这表示 `p` 是一个多项式, 计算时不会丢失精度.
- 否则, `p.shrink_len` 应当是一个**正整数** n, 在各种计算时将自动保留到 $\bmod x^n$ 的结果, 也即 0 到 $n-1$ 次项系数, 此时我们称 `p` 不是多项式.

此外, 模板库中有全局变量 `default_shrink`, 默认值为 -1, 表示初始化一个 `poly` 时默认设置的 `shrink_len`.

有如下辅助函数:

- `p.set_shrink(int n = default_shrink)`: 修改 `p` 的 `shrink_len`.
- `share_shrink(poly p, poly q)`: 将 `p, q` 的 `shrink_len` 修改成二者的较大值.

简单来说, 当设置了为正数的 `shrink_len` 之后, 你可以认为 `p` 做一系列操作后总是剩下一个长为 `shrink_len` 的数组.

你可以直接通过修改作为 `vector<mi>` 类型的系数序列 `p.coeff` 来完成一些目的, 在 `p.coeff` 的长度发生变化后, 建议使用 `p.shrink()` 函数进行调整. 模板中也有少量辅助操纵系数序列的函数, 比起作文字解释, 你可以选择阅读代码来判断是否使用.

4.2 初始化

提供三个构造函数.

- `poly(int n = default_shrink)`: 初始化为 0, 并且设置 `shrink_len` 为 `n`.
- `poly(vector<mi> v, int n = default_shrink)`: 初始化为 $\sum_i v_i x^i$, 并且设置 `shrink_len` 为 `n`. 如果 `shrink_len` 是正整数, 会将多余内容截去.
- `poly(vector<int> v, int n = default_shrink)`: 初始化为 $\sum_i v_i x^i$, 自动对 `int` 进行取模, 并且设置 `shrink_len` 为 `n`. 如果 `shrink_len` 是正整数, 会将多余内容截去.

4.3 访问

对于**非负整数** `i`, 可以通过 `p[i]` 对 `i` 次项系数进行读写. 如果 `p.shrink_len` 是正整数, **请保证** `i < p.shrink_len`.

4.4 运算

下设 `p, q` 皆为 `poly`, 如果是同时涉及 `p, q` 的运算, 且二者皆非多项式, **则需保证保留的精度相等**.

下设 $p(x) = \sum\limits_{n\geq 0} p_n x^n$, 并且 p, q 的度数 (或计算精度) 皆为 $\leq n$, 并且时间复杂度中忽略可能潜在的 $O(\log p)$ 低阶项.

理论上说, 你在本题单次运算涉及的单个多项式长度应当不超过 2.1×10^5**.** 如果你需要对更长的多项式进行运算，你可以酌情修改代码中的宏定义.

- `p + q`: 计算 $p(x)+q(x)$. 时间复杂度 $O(n)$.
- `p - q`: 计算 $p(x)-q(x)$. 时间复杂度 $O(n)$.
- `p * q`: 计算 $p(x)q(x)$. 时间复杂度 $O(n\log n)$.
- `gint(p)`: 计算积分 $\displaystyle\int_0^x p(t)\,\mathrm{d}t=\sum_{n\geq 0}\frac{p_n}{n+1}x^{n+1}$. 再次强调这一运算不会修改 `shrink_len`. 时间复杂度 $O(n)$.
- `gde(p)`: 计算导数 $\displaystyle p'(x)=\sum_{n\geq 1}np_nx^{n-1}$. 再次强调这一运算不会修改 `shrink_len`. 时间复杂度 $O(n)$.

多项式运算

本节中涉及的均为多项式.

- `gdiv(p, q)`: **需保证** p,q **皆为多项式,** q **非零**. 返回一个二元组 `pair<poly, poly>`, 其中 `first` 是带余除法的商, `second` 是带余除法的余数. 也即 $p(x)=q(x)d(x)+r(x)$, 返回的是 $\langle d,r\rangle$, 满足 $\deg r<\deg q$. 时间复杂度 $O(n\log n)$.
- `p / q`: **需保证** p,q **皆为多项式,** q **非零**. 计算带余除法的商. 时间复杂度 $O(n\log n)$.
- `prod(vector<poly> v)`: 计算 `v` 中所有多项式的乘积, 设 n 是度数之和, m 是多项式的数量, 时间复杂度 $O(n\log n\log m)$.

非多项式运算

本节中涉及的均为非多项式.

- `ginv(p)`: 计算逆元 $p(x)^{-1}$, **需保证** $\boldsymbol{p_0\neq 0}$. 时间复杂度 $O(n\log n)$.
- `gln(p)`: 计算对数 $\ln p(x)$, **需保证** $\boldsymbol{p_0=1}$. 时间复杂度 $O(n\log n)$.
- `gexp(p)`: 计算指数 $\exp p(x)$, **需保证** $\boldsymbol{p_0=0}$. 时间复杂度 $O(n\log n)$.
- `gsqrt(p, mi f0 = 1)`: 计算平方根 $\sqrt{p(x)}$, **需保证** $\boldsymbol{f_0^2=p_0\neq 0}$. 时间复杂度 $O(n\log n)$.
- `gpow(p, long long k)`: 计算 k 次方 $p(x)^k$, **需保证** k **非负**. 时间复杂度 $O(n\log n)$.

下发文件

下发的模板代码中, `main` 函数的内容是如下问题:

挑战多项式

给定 n 次多项式 $F(x)$, 求 $G(x)$ 满足

$$G(x) \equiv \left(\left(1 + \ln\left(2 + F(x) - F(0) - \exp\left(\int_0^x \frac{1}{\sqrt{F(t)}}\mathrm{d}t\right)\right)\right)^k\right)' \pmod{x^n},$$

保证常数项是模 998244353 的二次剩余.

在评测机上, 对该程序运行一组 $n = 10^5$ 的随机数据的耗时为 300 ms.