```cpp
class ModularInteger {
private:
	long long val = 0;
	long long MOD = (long long)1e9+7;

public:
	friend istream & operator >> (istream &in, ModularInteger &x);
	friend ostream & operator << (ostream &out, ModularInteger const &x);

	ModularInteger operator + (ModularInteger const &x) {
		ModularInteger res;
		res.val = val + x.val;
		if (res.val >= MOD) res.val -= MOD;
		return res;
	}

	ModularInteger operator - (ModularInteger const &x) {
		ModularInteger res;
		res.val = val - x.val;
		if (res.val < 0) res.val += MOD;
		return res;
	}

	ModularInteger operator * (ModularInteger const &x) {
		ModularInteger res;
		res.val = val * x.val;
		res.val %= MOD;
		return res;
	}

	ModularInteger binexp(ModularInteger &x, long long p) {
		if (p <= 0) {
			ModularInteger res;
			res.val = 1LL;
			return res;
		}
		if (p & 1LL) {
			ModularInteger res = binexp(x, p - 1);
			res = x * res;
			return res;
		}
		else {
			ModularInteger res = binexp(x, p >> 1);
			res = res * res;
			return res;
		}
	}
```