```cpp
#include <algorithm>
#include <chrono>
#include <iostream>
#include <limits>
#include <vector>
using namespace std;

bool isOdd(uint64_t num)
{
    return num & 0x1;
}

class bigInt
{
public:
    uint64_t hi;
    uint64_t lo;

    bigInt()
    {
        hi = 0x0;
        lo = 0x0;
    }
    bigInt(unsigned long long num)
    {
        hi = 0x0;
        lo = num; // 赋值不会过大
    }
    bigInt(const bigInt &rhs)
    {
        hi = rhs.hi;
        lo = rhs.lo;
    }
    bigInt &operator=(const bigInt &rhs)
    {
        hi = rhs.hi;
        lo = rhs.lo;
        return *this;
    }
    bigInt operator*(const bigInt &rhs)
    {
        bigInt ret;
        ret.lo = this->lo * rhs.lo;
        uint64_t a_lo = (uint32_t)(this->lo);
        uint64_t a_hi = this->lo >> 32;
        uint64_t b_lo = (uint32_t)(rhs.lo);
        uint64_t b_hi = rhs.lo >> 32;

        uint64_t a_x_b_hi = a_hi * b_hi;
        uint64_t a_x_b_mid = a_hi * b_lo;
        uint64_t b_x_a_mid = b_hi * a_lo;
        uint64_t a_x_b_lo = a_lo * b_lo;

        uint64_t carry_bit = ((uint64_t)(uint32_t)a_x_b_mid +
                              (uint64_t)(uint32_t)b_x_a_mid +
                              (a_x_b_lo >> 32)) >>
                             32;
        ret.hi = a_x_b_hi +
                 (a_x_b_mid >> 32) + (b_x_a_mid >> 32) +
                 carry_bit;
        return ret;
    }
    bigInt &operator<<=(int value)
    {
        for (int i = 0; i < value; i++)
        {
            int msb = this->lo >> 63;
            this->lo <<= 1;
            this->hi <<= 1;
            this->hi |= msb;
        }
        return *this;
    }
    bigInt &operator>>=(int value)
    {
        for (int i = 0; i < value; i++)
        {
            uint64_t lsb = this->hi << 63;
            this->hi >>= 1;
            this->lo >>= 1;
            this->lo |= lsb;
        }
        return *this;
    }
    bigInt operator<<(int value)
    {
        bigInt ret;
        ret.hi = this->hi;
        ret.lo = this->lo;
        for (int i = 0; i < value; i++)
        {
            int msb = ret.lo >> 63;
            ret.lo <<= 1;
            ret.hi <<= 1;
            ret.hi |= msb;
        }
        return ret;
    }
    bigInt operator>>(int value)
    {
        bigInt ret;
        ret.hi = this->hi;
        ret.lo = this->lo;
        for (int i = 0; i < value; i++)
        {
            uint64_t lsb = ret.hi << 63;
            ret.hi >>= 1;
            ret.lo >>= 1;
            ret.lo |= lsb;
        }
        return ret;
    }
    bigInt operator-(const bigInt &rhs)
    {
        bigInt ret;
        ret.hi = this->hi - rhs.hi;
        if (this->lo > rhs.lo)
        {
            ret.lo = this->lo - rhs.lo;
        }
        else
        {
            ret.hi -= 1;
            ret.lo = 0xffffffffffffffff - (rhs.lo - this->lo) + 1;
        }
        return ret;
    }
    bigInt operator/(int div)
    {
        bigInt ret;
        ret.hi = this->hi / div;
        ret.lo = this->lo / div;
        return ret;
    }
    bool operator==(const bigInt &rhs)
    {
        if (this->hi == rhs.hi && this->lo == rhs.lo)
        {
            return true;
        }
        return false;
    }
    bool operator>(const bigInt &rhs)
    {
        if (this->hi > rhs.hi)
        {
            return true;
        }
        else if (this->hi < rhs.hi)
        {
            return false;
        }
        else
        {
            if (this->lo > rhs.lo)
            {
                return true;
            }
            return false;
        }
    }
    bool operator<(const bigInt &rhs)
    {
        if (this->hi < rhs.hi)
        {
            return true;
        }
        else if (this->hi > rhs.hi)
        {
            return false;
        }
        else
        {
            if (this->lo < rhs.lo)
            {
                return true;
            }
            return false;
        }
    }
    bool operator>=(const bigInt &rhs)
    {
        return (*this > rhs) || (*this == rhs);
```

```cpp
    }
    bool operator<=(const bigInt &rhs)
    {
        return (*this < rhs) || (*this == rhs);
    }
    bool operator!=(int num)
    {
        bigInt a;
        a.lo = num;
        return !(*this == a);
    }
};

bigInt modulo(bigInt A, uint64_t mod)
{
    bigInt B;
    B.lo = mod;
    bigInt X = B;
    while (X <= A / 2)
    {
        // X = X << 1;
        X <<= 1;
    }
    while (A >= B)
    {
        if (A >= X)
        {
            A = A - X;
        }
        // X = X >> 1;
        X >>= 1;
    }
    return A;
}

bigInt powm(bigInt base, bigInt exp, uint64_t mod)
{
    bigInt ret;
    ret.lo = 0x1;
    bigInt temp = modulo(base, mod);
    while (exp != 0)
    {
        if (exp.lo & 0x1)
        {
            ret = modulo((ret * temp), mod);
        }
        // exp = exp >> 1;
        exp >>= 1;
        temp = modulo((temp * temp), mod);
    }
    return ret;
}

void test()
{
    vector<int> vec;
    uint64_t mod = 0xc0001000040000b;
    bigInt g(12332102632472395673ULL);
    bigInt s(20337250ULL);
    for (int i = 0; i < 512; i++)
    {
        s = powm(g, s, mod);
        vec.push_back(isOdd(s.lo));
    }
    for (int i : vec)
        cout << i;
    cout << "\n0: " << count(vec.begin(), vec.end(), 0) << " "
         << "1: " << count(vec.begin(), vec.end(), 1);
}

int main()
{
    auto start = chrono::high_resolution_clock::now();
    test();
    auto end = chrono::high_resolution_clock::now();
    std::chrono::duration<double> fp_ms = end - start;
    cout << "\nTime: " << fp_ms.count() << endl;
}
```