

memset0's Notebook

方知蓦然回首之时
那人却已不在灯火阑珊处

关于我
友情链接
文章聚合

Theme Ringo by memset0
Proudly powered by Typecho

LOJ150 挑战多项式

2019-02-20 | 题解

这题非常的水，我们拉个板子就 A 掉了...

整理到 [这篇博文](#) 中了。

```
// =====  
//   author: memset0  
//   date: 2019.02.19 09:56:53  
//   website: https://memset0.cn/  
// =====  
#include <bits/stdc++.h>  
#define ll long long  
#define poly std::vector<int>  
#define for_each(i, a) for (int i = 0, __lim = a.size(); i < __lim; ++i)  
namespace ringo {  
    template <class T> inline void read(T &x) {  
        x = 0; char c = getchar(); bool f = 0;  
        while (!isdigit(c)) f ^= c == '-', c = getchar();  
        while (isdigit(c)) x = x * 10 + c - '0', c = getchar();  
        if (f) x = -x;  
    }  
    template <class T> inline void print(T x) {  
        if (x < 0) putchar('-'), x = -x;  
        if (x > 9) print(x / 10);  
        putchar('0' + x % 10);  
    }  
}
```



memset0's Notebook

方知蓦然回首之时
那人却已不在灯火阑珊处

关于我
友情链接
文章聚合

Theme [Ringo](#) by [memset0](#)
Proudly powered by [Typecho](#)

```

}
template <class T> inline void print(T x, char c) { print(x), putchar(c); }
inline void print(const poly &a) { for_each(i, a) print(a[i], " \n"[i == __li
inline void read(poly &a, int n) { for (int i = 0, x; i < n; i++) read(x), a.

const int N = 1e5 + 10, mod = 998244353;
int n, k; poly f, g;

namespace poly_namespace {
    const int M = N << 3, SIZE = sizeof(int);
    int w[M], rev[M];
    inline poly resize(poly f, int n) { return f.resize(n), f; }
    inline int dec(int a, int b) { a -= b; return a < 0 ? a + mod : a; }
    inline int sub(int a, int b) { a += b; return a >= mod ? a - mod : a; }
    inline int inv(int x) { return x < 2 ? 1 : (ll)(mod - mod / x) * inv(mod
    inline int fpow(int a, int b) { int s = 1; for (; b >= 1, a = (ll)a *
    inline poly operator + (poly f, int a) { f[0] = sub(f[0], a); return f; }
    inline poly operator + (int a, poly f) { f[0] = sub(a, f[0]); return f; }
    inline poly operator - (poly f, int a) { f[0] = dec(f[0], a); return f; }
    inline poly operator - (int a, poly f) { for_each(i, f) f[i] = dec(0, f[i
    inline poly operator * (poly f, int a) { for_each(i, f) f[i] = (ll)f[i] *
    inline poly operator * (int a, poly f) { for_each(i, f) f[i] = (ll)f[i] *
    inline poly operator + (poly f, const poly &g) {
        f.resize(std::max(f.size(), g.size()));
        for_each(i, f) f[i] = sub(i < f.size() ? f[i] : 0, i < g.size() ? g[i
        return f;
    }
    inline poly operator - (poly f, const poly &g) {
        f.resize(std::max(f.size(), g.size()));
        for_each(i, f) f[i] = dec(i < f.size() ? f[i] : 0, i < g.size() ? g[i
        return f;
    }
}

namespace cipolla_namespace {
    int t, sqr_w;
    typedef std::pair <int, int> pair;
    inline pair operator * (const pair &a, const pair &b) {
        return std::make_pair(((ll)a.first * b.first + (ll)a.second * b.s

```



memset0's Notebook

方知蓦然回首之时
那人却已不在灯火阑珊处

关于我
友情链接
文章聚合

Theme [Ringo](#) by [memset0](#)
Proudly powered by [Typecho](#)

```

        ((ll)a.first * b.second + (ll)a.second * b.first) % mod);
    }
    int cipolla(int x) {
        do t = rand() % mod; while (fpow(sqr_w = dec((ll)t * t % mod, x),
        pair s = std::make_pair(1, 0), a = std::make_pair(t, 1);
        for (int b = (mod + 1) >> 1; b; b >>= 1, a = a * a) if (b & 1) s
        return std::min(s.first, mod - s.first);
    }
} using cipolla_namespace::cipolla;
void ntt(int *a, int lim) {
    for (int i = 0; i < lim; i++) if (i < rev[i]) std::swap(a[i], a[rev[i]
    for (int len = 1; len < lim; len <= 1)
        for (int i = 0; i < lim; i += (len < 1))
            for (int j = 0; j < len; j++) {
                int x = a[i + j], y = (ll)w[j + len] * a[i + j + len] % m
                a[i + j] = sub(x, y), a[i + j + len] = dec(x, y);
            }
    }
}
int init(int len) {
    int lim = 1, k = 0; while (lim < len) lim <= 1, ++k;
    for (int i = 0; i < lim; i++) rev[i] = (rev[i >> 1] >> 1) | ((i & 1)
    return lim;
}
void main_init() {
    for (int len = 1, wn; (len < 1) < M; len <= 1) {
        wn = fpow(3, (mod - 1) / (len < 1)), w[len] = 1;
        for (int i = 1; i < len; i++) w[i + len] = (ll)w[i + len - 1] * w
    }
}
inline poly operator * (const poly &f, const poly &g) {
    static int a[M], b[M];
    int lim = init(f.size() + g.size() - 1), inv_lim = inv(lim); poly h;
    memset(&a[f.size()], 0, (lim - f.size()) * SIZE); for_each(i, f) a[i]
    memset(&b[g.size()], 0, (lim - g.size()) * SIZE); for_each(i, g) b[i]
    ntt(a, lim), ntt(b, lim);
    for (int i = 0; i < lim; i++) a[i] = (ll)a[i] * b[i] % mod;
    std::reverse(a + 1, a + lim), ntt(a, lim);
}

```



memset0's Notebook

方知蓦然回首之时
那人却已不在灯火阑珊处

关于我
友情链接
文章聚合

Theme [Ringo](#) by [memset0](#)
Proudly powered by [Typecho](#)

```

    for (int i = 0, l = f.size() + g.size() - 1; i < l; i++) h.push_back(
        return h;
    }
    inline poly inv(const poly &f) {
        static int a[M], b[M];
        poly g(1, inv(f[0]));
        for (int len = 2; (len >> 1) < f.size(); len <= 1) {
            int lim = init(len < 1), inv_lim = inv(lim);
            memset(&a[len], 0, len * SIZE); for (int i = 0; i < len; i++) a[i]
            memset(&b[len], 0, len * SIZE); for (int i = 0; i < len; i++) b[i]
            ntt(a, lim), ntt(b, lim);
            for (int i = 0; i < lim; i++) a[i] = (ll)a[i] * b[i] % mod * b[i]
            std::reverse(a + 1, a + lim), ntt(a, lim), g.resize(len);
            for_each(i, g) g[i] = dec(sub(g[i], g[i]), (ll)a[i] * inv_lim % m
        } return g.resize(f.size()), g;
    }
    inline poly sqrt(const poly &f) {
        poly g(1, cipolla(f[0]));
        for (int len = 2; (len >> 1) < f.size(); len <= 1)
            g = resize(resize(resize(g * g, len) + f, len) * inv(resize(2 * g
        return g.resize(f.size()), g;
    }
    inline poly deri(const poly &f) {
        poly g;
        for (int i = 0; i < f.size() - 1; i++) g.push_back((ll)(i + 1) * f[i]
        return g.push_back(0), g;
    }
    inline poly inte(poly f) {
        poly g(1, 0);
        for (int i = 0; i < f.size() - 1; i++) g.push_back((ll)inv(i + 1) * f
        return g;
    }
    inline poly ln(const poly &f) { return inte(resize(deri(f) * inv(f), f.si
    inline poly exp(const poly &f) {
        poly g(1, 1);
        for (int len = 2; (len >> 1) < f.size(); len <= 1)
            g = resize(g * (1 - ln(resize(g, len)) + resize(f, len)), len);

```



memset0's Notebook

方知蓦然回首之时
那人却已不在灯火阑珊处

关于我
友情链接
文章聚合

Theme Ringo by memset0
Proudly powered by Typecho

```
        return g.resize(f.size()), g;
    }
    inline poly fpow(poly a, int b) {
        int n = a.size(); poly s(1, 1);
        for (; b; b >>= 1, a = resize(a * a, n))
            if (b & 1) s = resize(s * a, n);
        return s;
    }
} using namespace poly_namespace;

void main() {
    srand(20040725);
    read(n), read(k), read(f, n + 1), main_init();
    g = deri(fpow(1 + ln(2 + f - f[0] - exp(inte(inv(sqrt(f))))), k));
    for (int i = 0; i < g.size() - 1; i++) print(g[i], " \n"[i == g.size() - 1]);
}

} signed main() { return ringo::main(), 0; }
```

用户名

邮箱

网址 (选填)

可以在这里写评论哦 ~



memseto's Notebook

方知蓦然回首之时
那人却已不在灯火阑珊处

关于我
友情链接
文章聚合

Theme **Ringo** by **memseto**
Proudly powered by **Typecho**

提交评论

关于这个博客
上一篇 «

特征多项式和常系数线性齐次递推学习
笔记
» 下一篇

© 2017 - 2019 **memset0** 的博客.
浙ICP备19006255号-1
97696 visits · 24756 visitors · 74.48 W words

在这里输入关键字哦 ~ (回车搜索)

