带项式

```
/* FFT */
class COMP{
  public: LDB x, y;
    COMP operator + (const COMP &o) const{
        COMP r = *this; r.x += o.x, r.y += o.y;
        return r;
    }
    COMP operator - (const COMP &o) const{
        COMP r = *this; r.x -= o.x, r.y -= o.y;
        return r;
    }
    COMP operator * (const COMP &o) const{
        COMP r; r.x = x * o.x - y * o.y, r.y = x * o.y + y * o.x;
        return r;
    }
} ;
COMP mk(LDB a, LDB b) {COMP o; o.x = a, o.y = b; return o;}
int rev[CN << 2];
void cg(COMP t[], int n){
    for(int i = 0; i < n; i++) rev[i] = (rev[i >> 1] >> 1) + (i & 1) * (n >> 1);
    for(int i = 0; i < n; i++) if(i < rev[i]) swap(t[i], t[rev[i]]);
}
void fft(COMP t[], int n, int tp){
    cg(t, n);
    for(int w = 2; w <= n; w <<= 1){
        int l = w >> 1; COMP gn = mk(cos(2 * PI / (LDB)w), sin(2 * tp * PI /
(LDB)w));
        for(int i = 0; i < n; i += w){
            COMP g = mk(1, 0);
            for(int j = i; j < i + l; j++){
                COMP u = t[j], v = t[j + l] * g;
                t[j] = u + v, t[j + l] = u - v, g = g * gn;
            }
        }
    }
    if(tp ^ 1){
        for(int i = 0; i < n; i++) t[i].x /= (LDB)n, t[i].y /= (LDB)n;
    }
}

/* NTT */
int qp(LL a, LL b){
    LL r = 1;
    for(; b; b >>= 1, a = a * a % P) if(b & 1) r = r * a % P;
    return r;
}
void cg(int t[], int n){
    for(int i = 0; i < n; i++) rev[i] = (rev[i >> 1] >> 1) + (i & 1) * (n >> 1);
    for(int i = 0; i < n; i++) if(i < rev[i]) swap(t[i], t[rev[i]]);
}
void ntt(int t[], int n, int tp){
```

```
    cg(t, n); int G = 3, iG = qp(G, P - 2);
    for(int w = 2; w <= n; w <<= 1){
        int l = w >> 1; LL gn = qp(tp ? G : iG, (P - 1) / w), g = 1;
        for(int i = 0; i < n; g = 1, i += w) for(int j = i; j < i + l; j++){
            LL u = t[j], v = g * t[j + l];
            t[j] = (u + v) % P, t[j + l] = (u - v) % P, g = g * gn % P;
        }
    }
    for(int i = 0; i < n; i++) t[i] = (t[i] + P) % P;
    if(!tp) {LL iv = qp(n, P - 2); for(int i = 0; i < n; i++) t[i] = iv * t[i] %
P;}
}

/* FWT */
void fwt(int t[], int n, int tp){ // xor
    for(int w = 2; w <= n; w <<= 1){
        int l = w >> 1;
        for(int i = 0; i < n; i += w) for(int j = i; j < i + l; j++){
            int u = t[j], v = t[j + l];
            t[j] = add(u, v), t[j + l] = add(u, P - v);
            if(!tp) t[j] = 1ll * t[j] * i2 % P, t[j + l] = 1ll * t[j + l] * i2 %
P;
        }
    }
}
void fwt(int t[], int n, int tp){ // or
    for(int w = 2; w <= n; w <<= 1){
        int l = w >> 1;
        for(int i = 0; i < n; i += w) for(int j = i; j < i + l; j++){
            if(tp) t[j + l] = add(t[j + l], t[j]);
            else t[j + l] = add(t[j + l], P - t[j]);
        }
    }
}
void fwt(int t[], int n, int tp){ // and
    for(int w = 2; w <= n; w <<= 1){
        int l = w >> 1;
        for(int i = 0; i < n; i += w) for(int j = i; j < i + l; j++){
            if(tp) t[j] = add(t[j], t[j + l]);
            else t[j] = add(t[j], P - t[j + l]);
        }
    }
}

/* 求逆加指对 */
int rev[CN << 2];
void cg(int t[], int n){
    for(int i = 0; i < n; i++) rev[i] = (rev[i >> 1] >> 1) + (i & 1) * (n >> 1);
    for(int i = 0; i < n; i++) if(i < rev[i]) swap(t[i], t[rev[i]]);
}
void ntt(int t[], int n, int tp){
    cg(t, n); int og = 3, ig = invx(og);
    for(int w = 2; w <= n; w <<= 1){
        int l = w >> 1, gn = qp(tp ? og : ig, (P - 1) / w);
        for(int i = 0; i < n; i += w){
```

```
            int g = 1;
            for(int j = i; j < i + l; j++){
                int u = t[j], v = 1ll * t[j + l] * g % P;
                t[j] = add(u, v), t[j + l] = add(u, P - v), g = 1ll * g * gn %
P;
            }
        }
    }
    if(!tp){
        ig = invx(n);
        for(int i = 0; i < n; i++) t[i] = 1ll * t[i] * ig % P;
    }
}
int c[CN << 2];
void inv(int a[], int b[], int n){
    for(int i = 0; i < (n << 1); i++) b[i] = c[i] = 0;
    b[0] = invx(a[0]);
    for(int w = 2; w < (n << 1); w <<= 1){
        for(int i = 0; i < w; i++) c[i] = a[i];
        ntt(b, w << 1, 1), ntt(c, w << 1, 1);
        for(int i = 0; i < (w << 1); i++) b[i] = 1ll * b[i] * add(2, P - (1ll *
c[i] * b[i] % P)) % P;
        ntt(b, w << 1, 0);
        for(int i = w; i < (w << 1); i++) b[i] = 0;
    }
    for(int i = n; i < (n << 1); i++) b[i] = 0;
}
int ia[CN << 2];
void ln(int a[], int b[], int n){
    int N = 1; while(N < (n << 1)) N <<= 1;
    for(int i = 0; i < N; i++) ia[i] = b[i] = 0;
    inv(a, ia, n);
    for(int i = 0; i < N; i++) c[i] = 0;
    for(int i = 0; i < n - 1; i++) c[i] = 1ll * (i + 1) * a[i + 1] % P;
    ntt(ia, N, 1), ntt(c, N, 1);
    for(int i = 0; i < N; i++) c[i] = 1ll * c[i] * ia[i] % P;
    ntt(c, N, 0);
    for(int i = 1; i < n; i++) b[i] = 1ll * c[i - 1] * invx(i) % P;
    b[0] = 0; for(int i = n; i < (n << 1); i++) b[i] = 0;
}
int lnb[CN << 2];
void exp(int a[], int b[], int n){
    for(int i = 0; i < (n << 1); i++) b[i] = lnb[i] = 0; b[0] = 1;
    for(int w = 2; w < (n << 1); w <<= 1){
        ln(b, lnb, w);
        for(int i = 0; i < w; i++) lnb[i] = add(a[i], P - lnb[i]);
        lnb[0] = add(lnb[0], 1);
        ntt(b, w << 1, 1), ntt(lnb, w << 1, 1);
        for(int i = 0; i < (w << 1); i++) b[i] = 1ll * b[i] * lnb[i] % P;
        ntt(b, w << 1, 0);
        for(int i = w; i < (w << 1); i++) b[i] = 0;
    }
    for(int i = n; i < (n << 1); i++) b[i] = 0;
}
```

```
/* myy卷积 任意模数卷积 */
#define LDB long double
class COMP{
  public: LDB x, y;
    COMP operator + (const COMP &o) const{
        COMP r = *this; r.x += o.x, r.y += o.y;
        return r;
    }
    COMP operator - (const COMP &o) const{
        COMP r = *this; r.x -= o.x, r.y -= o.y;
        return r;
    }
    COMP operator * (const COMP &o) const{
        COMP r; r.x = x * o.x - y * o.y, r.y = x * o.y + y * o.x;
        return r;
    }
} ;
COMP mk(LDB a, LDB b) {COMP o; o.x = a, o.y = b; return o;}
COMP conj(COMP o) {o.y = -o.y; return o;}
int rev[CN << 2];
void cg(COMP t[], int n){
    for(int i = 0; i < n; i++) rev[i] = (rev[i >> 1] >> 1) + (i & 1) * (n >> 1);
    for(int i = 0; i < n; i++) if(i < rev[i]) swap(t[i], t[rev[i]]);
}
void fft(COMP t[], int n, int tp){
    cg(t, n);
    for(int w = 2; w <= n; w <<= 1){
        int l = w >> 1; COMP gn = mk(cos(2 * PI / (LDB)w), sin(2 * tp * PI /
(LDB)w));
        for(int i = 0; i < n; i += w){
            COMP g = mk(1, 0);
            for(int j = i; j < i + l; j++){
                COMP u = t[j], v = t[j + l] * g;
                t[j] = u + v, t[j + l] = u - v, g = g * gn;
            }
        }
    }
    if(tp ^ 1){
        for(int i = 0; i < n; i++) t[i].x /= (LDB)n, t[i].y /= (LDB)n;
    }
}
COMP p[CN << 2], q[CN << 2], x[CN << 2], y[CN << 2], z[CN << 2], w[CN << 2];
void conv(int a[], int b[], int n){ // a = a * b
    int B = (1 << 15) - 1, N = 1; while(N < (n << 1)) N <<= 1;
    for(int i = 0; i < n; i++) p[i] = mk(a[i] >> 15, a[i] & B); // k1 r1
    for(int i = 0; i < n; i++) q[i] = mk(b[i] >> 15, b[i] & B); // k2 r2
    fft(p, N, 1), fft(q, N, 1);
    for(int i = 0; i < N; i++){
        int j = (N - 1) & (N - i);
        COMP k1, r1, k2, r2;
        k1 = (p[i] + conj(p[j])) * mk(0.5, 0);
        r1 = (p[i] - conj(p[j])) * mk(0, -0.5);
        k2 = (q[i] + conj(q[j])) * mk(0.5, 0);
        r2 = (q[i] - conj(q[j])) * mk(0, -0.5);
        x[i] = k1 * k2, y[i] = r1 * r2, z[i] = k1 * r2, w[i] = k2 * r1;
```

```
    }
    for(int i = 0; i < N; i++) p[i] = x[i] + y[i] * mk(0, 1);
    for(int i = 0; i < N; i++) q[i] = z[i] + w[i] * mk(0, 1);
    fft(p, N, -1), fft(q, N, -1);
    for(int i = 0; i < N; i++){
        LL X = (LL)(0.5 + p[i].x), Y = (LL)(0.5 + p[i].y), Z = (LL)(0.5 +
q[i].x), W = (LL)(0.5 + q[i].y);
        X = (X % P + P) % P, Y = (Y % P + P) % P, Z = (Z % P + P) % P, W =
add((W % P + P) % P, Z);
        a[i] = add((X << 30) % P, add((W << 15) % P, Y));
    }
}
```