

# 神秘模板库

Toy ASM Truck

Huazhong University of Science and Technology

April 2, 2021

# Contents

一切的开始	2
宏定义	2
数据结构	2
ST 表	2
数学	2
模整数类	2
类欧几里得	3
Pollard-Rho	3
ex-gcd	4
crt	4
ex-crt	5
Meissel-Lehmer	5
Cipolla 二次剩余	6
BSGS	6
exBSGS	7
图论	8
LCA	8
计算几何	8
二维几何: 点与向量	8
完整的板板 by zcs	9
字符串	13
后缀自动机	13
多项式	13
NTT 模数	13
FFT	13
NTT	14
完整的板板 by hls	15

## 一切的开始

### 宏定义

## 数据结构

### ST 表

- 二维

```
1  int f[maxn][maxn][10][10];
2  inline int highbit(int x) { return 31 - __builtin_clz(x); }
3  inline int calc(int x, int y, int xx, int yy, int p, int q) {
4      return max(
5          max(f[x][y][p][q], f[xx - (1 << p) + 1][yy - (1 << q) + 1][p][q]),
6          max(f[xx - (1 << p) + 1][y][p][q], f[x][yy - (1 << q) + 1][p][q])
7      );
8  }
9  void init() {
10     FOR (x, 0, highbit(n) + 1)
11     FOR (y, 0, highbit(m) + 1)
12         FOR (i, 0, n - (1 << x) + 1)
13             FOR (j, 0, m - (1 << y) + 1) {
14                 if (!x && !y) { f[i][j][x][y] = a[i][j]; continue; }
15                 f[i][j][x][y] = calc(
16                     i, j,
17                     i + (1 << x) - 1, j + (1 << y) - 1,
18                     max(x - 1, 0), max(y - 1, 0)
19                 );
20             }
21 }
22 inline int get_max(int x, int y, int xx, int yy) {
23     return calc(x, y, xx, yy, highbit(xx - x + 1), highbit(yy - y + 1));
24 }
```

## 数学

### 模整数类

除法为整除，请乘逆元。

```
1  template<int P>
2  struct moint {
3      int x;
4      moint():x(0){}
5      moint(int n){x=n<0?n%P+P:n%P;}
6      moint(ll n){x=n<0?n%P+P:n%P;}
7      int get()const{return (int)x;}
8      moint &operator+=(moint b){x+=b.x;if(x>=P)x-=P;return *this;}
9      moint &operator-=(moint b){x-=b.x;if(x<0)x+=P;return *this;}
10     moint &operator*=(moint b){x=1ll*x*b.x%P;return *this;}
11     moint &operator/=(moint b){x=x/b.x;return *this;}
12     moint &operator%=(moint b){x=x%b.x;return *this;}
13     moint operator+(moint b)const{return moint(*this)+=b;}
14     moint operator-(moint b)const{return moint(*this)-=b;}
15     moint operator*(moint b)const{return moint(*this)*=b;}
16     moint operator/(moint b)const{return moint(*this)/=b;}
17     moint operator%(moint b)const{return moint(*this)%=b;}
18     moint operator+(int b)const{return moint(*this)+=moint(b);}
19     moint operator-(int b)const{return moint(*this)-=moint(b);}
20     moint operator*(int b)const{return moint(*this)*=moint(b);}
21     moint operator/(int b)const{return moint(*this)/=moint(b);}
22     moint operator%(int b)const{return moint(*this)%=moint(b);}
23     bool operator==(moint b)const{return x==b.x;}
24     bool operator>=(moint b)const{return x>=b.x;}
25     bool operator!=(moint b)const{return x!=b.x;}
26 };
27 typedef moint<998244353> mint;
```

## 类欧几里得

- $m = \lfloor \frac{an+b}{c} \rfloor$ .
- $f(a, b, c, n) = \sum_{i=0}^n \lfloor \frac{ai+b}{c} \rfloor$ : 当  $a \geq c$  或  $b \geq c$  时,  $f(a, b, c, n) = (\frac{a}{c})n(n+1)/2 + (\frac{b}{c})(n+1) + f(a \bmod c, b \bmod c, c, n)$ ; 否则  $f(a, b, c, n) = nm - f(c, c-b-1, a, m-1)$ 。
- $g(a, b, c, n) = \sum_{i=0}^n i \lfloor \frac{ai+b}{c} \rfloor$ : 当  $a \geq c$  或  $b \geq c$  时,  $g(a, b, c, n) = (\frac{a}{c})n(n+1)(2n+1)/6 + (\frac{b}{c})n(n+1)/2 + g(a \bmod c, b \bmod c, c, n)$ ; 否则  $g(a, b, c, n) = \frac{1}{2}(n(n+1)m - f(c, c-b-1, a, m-1) - h(c, c-b-1, a, m-1))$ 。
- $h(a, b, c, n) = \sum_{i=0}^n \lfloor \frac{ai+b}{c} \rfloor^2$ : 当  $a \geq c$  或  $b \geq c$  时,  $h(a, b, c, n) = (\frac{a}{c})^2 n(n+1)(2n+1)/6 + (\frac{b}{c})^2 (n+1) + (\frac{a}{c})(\frac{b}{c})n(n+1) + h(a \bmod c, b \bmod c, c, n) + 2(\frac{a}{c})g(a \bmod c, b \bmod c, c, n) + 2(\frac{b}{c})f(a \bmod c, b \bmod c, c, n)$ ; 否则  $h(a, b, c, n) = nm(m+1) - 2g(c, c-b-1, a, m-1) - 2f(c, c-b-1, a, m-1) - f(a, b, c, n)$ 。

```

1  struct ans{mint f,g,h};
2  static ans calc(int a,int b,int c,int n){
3      ans ret;
4      if(!a){
5          ret.f=mint(b/c)*(n+1);
6          ret.g=mint(b/c)*n*(n+1)*iv2;
7          ret.h=mint(b/c)*(b/c)*(n+1);
8          return ret;
9      }
10     if(a>=c||b>=c){
11         ans to=calc(a%c,b%c,c,n);
12         ret.f=mint(a/c)*n*(n+1)*iv2+mint(b/c)*(n+1)+to.f;
13         ret.g=mint(a/c)*n*(n+1)*(n+2+1)*iv6+mint(b/c)*n*(n+1)*iv2+to.g;
14         ret.h=mint(a/c)*(a/c)*n*(n+1)*(n+2+1)*iv6+mint(b/c)*(b/c)*(n+1)+\
15             mint(a/c)*(b/c)*n*(n+1)+to.h+mint(a/c)*2*to.g+mint(b/c)*2*to.f;
16         return ret;
17     }else{
18         ll m=(1ll*a*n+b)/c;
19         ans to=calc(c,c-b-1,a,m-1);
20         ret.f=mint(n*m%P)-to.f;
21         ret.g=(mint(m*n%P*(n+1)%P)-to.f-to.h)*iv2;
22         ret.h=mint(m*n%P*(m+1)%P)-to.g*2-to.f*2-ret.f;
23         return ret;
24     }
25 }

```

## Pollard-Rho

```

1  template<const int test_case>// set 8 usually
2  struct Pollard_Rho {
3      vector<long long> fac;
4      long long quick_pow(long long a, long long b, long long mod) {
5          long long ans = 1;
6          while (b) {
7              if (b&1) ans = (__int128)ans*(__int128)a%mod;
8              b >>= 1, a = (__int128)a*(__int128)a%mod;
9          }
10         return ans;
11     }
12     bool Miller_Rabin(long long n) { // return if n is a prime
13         if (n < 3) return n == 2;
14         long long a = n-1, b = 0;
15         while (a%2 == 0) a /= 2, ++b;
16         for (int i = 0, j; i < test_case; i++) {
17             long long x = rand()%(n-2)+2, v = quick_pow(x, a, n);
18             if (v == 1 || v == n-1) continue;
19             for (j = 0; j < b; j++) {
20                 v = (__int128)v*(__int128)v%n;
21                 if (v == n-1) break;
22             }
23             if (j >= b) return false;
24         }
25         return true;
26     }
27     long long f(long long x, long long c, long long n) { return ((__int128)x * x + c) % n; }
28     long long rho(long long x) {
29         long long s = 0, t = 0;
30         long long c = (__int128)rand() % (x - 1) + 1;

```

```

31     int step = 0, goal = 1;
32     long long val = 1;
33     for (goal = 1;; goal <= 1, s = t, val = 1) {
34         for (step = 1; step <= goal; ++step) {
35             t = f(t, c, x);
36             val = (__int128)val * abs(t - s) % x;
37             if ((step % 127) == 0) {
38                 long long d = __gcd(val, x);
39                 if (d > 1) return d;
40             }
41         }
42         long long d = __gcd(val, x);
43         if (d > 1) return d;
44     }
45 }
46 void find(long long x) {
47     if (x == 1) return;
48     if (Miller_Rabin(x)) {
49         fac.push_back(x);
50         return;
51     }
52     long long p = x;
53     while (p >= x) p = rho(x);
54     //while ((x % p) == 0) x /= p;
55     find(x/p), find(p);
56 }
57 vector<long long> factor(long long n) { // return the factors of n
58     srand((unsigned)time(NULL));
59     fac.clear();
60     find(n);
61     sort(fac.begin(), fac.end());
62     return fac;
63 }
64 };

```

## ex-gcd

```

1  template<typename T>
2  struct ex_gcd {
3      T gcd(const T a, const T b, T &x, T &y) { // x'=x_0+b/gcd, y'=y_0-a/gcd
4          if (b == 0) {x = 1, y = 0; return a; }
5          T d = gcd(b, a%b, x, y);
6          T t = x;
7          x = y;
8          y = t - a/b*y;
9          return d;
10     }
11     T inv(const T a, const T m) { // return -1 if inv is not exist
12         if (a == 0 || m <= 1) return -1;
13         T x, y, d = gcd(a, m, x, y);
14         if (d != 1) return -1;
15         return (x%m+m)%m;
16     }
17 };

```

## crt

```

1  template<typename T>
2  struct crt {
3      ex_gcd<T> *exgcd = new ex_gcd<T>();
4      T cal(const T *a, const T *m, const int n) { // a[1..n], m[1..n], gcd(m_i) = 1
5          T M = 1, ans = 0;
6          for (int i = 1; i <= n; i++) M *= m[i];
7          for (int i = 1; i <= n; i++)
8              (ans += (__int128)a[i]*(M/m[i])%M*exgcd->inv(M/m[i], m[i])%M) %= M;
9          return ans;
10     }
11 };

```

## ex-crt

```
1 template<typename T>
2 struct ex_crt {
3     ex_gcd<T> *exgcd = new ex_gcd<T>();
4     T cal(T *a, T *m, const int n) { // a[1..n], m[1..n], return -1 if no ans
5         T x, y, gcd, lcm;
6         for (int i = 2; i <= n; i++) {
7             gcd = exgcd->gcd(m[1], m[i], x, y);
8             if ((a[i]-a[1])%gcd) return -1;
9             lcm = (__int128)m[1]*m[i]/gcd;
10            x = (__int128)x*(a[i]-a[1])/gcd%lcm;
11            gcd = m[i]/gcd;
12            x = (x%gcd+gcd)%gcd;
13            a[1] = ((__int128)m[1]*x%lcm+a[1])%lcm, m[1] = lcm;
14        }
15        return a[1];
16    }
17 } ;
```

## Meissel-Lehmer

求解  $1e11$  内的质数个数, 约为  $O(n^{2/3})$ 。

```
1 namespace pcf{
2     #define chkbit(ar, i) (((ar[(i) >> 6]) & (1 << (((i) >> 1) & 31))))
3     #define setbit(ar, i) (((ar[(i) >> 6]) |= (1 << (((i) >> 1) & 31))))
4     #define isprime(x) (( (x) && (x)&1) && (!chkbit(ar, (x)))) || ((x) == 2))
5     const int MAXN=100;
6     const int MAXM=10001;
7     const int MAXP=40000;
8     const int MAX=400000;
9     long long dp[MAXN][MAXM];
10    unsigned int ar[(MAX >> 6) + 5] = {0};
11    int len = 0, primes[MAXP], counter[MAX];
12    void Sieve(){
13        setbit(ar, 0), setbit(ar, 1);
14        for (int i = 3; (i * i) < MAX; i++, i++){
15            if (!chkbit(ar, i)){
16                int k = i << 1;
17                for (int j = (i * i); j < MAX; j += k) setbit(ar, j);
18            }
19        }
20        for (int i = 1; i < MAX; i++){
21            counter[i] = counter[i - 1];
22            if (isprime(i)) primes[len++] = i, counter[i]++;
23        }
24    }
25    void init(){
26        Sieve();
27        for (int n = 0; n < MAXN; n++){
28            for (int m = 0; m < MAXM; m++){
29                if (!n) dp[n][m] = m;
30                else dp[n][m] = dp[n - 1][m] - dp[n - 1][m / primes[n - 1]];
31            }
32        }
33    }
34    long long phi(long long m, int n){
35        if (n == 0) return m;
36        if (primes[n - 1] >= m) return 1;
37        if (m < MAXM && n < MAXN) return dp[n][m];
38        return phi(m, n - 1) - phi(m / primes[n - 1], n - 1);
39    }
40    long long Lehmer(long long m){
41        if (m < MAX) return counter[m];
42        long long w, res = 0;
43        int i, a, s, c, x, y;
44        s = sqrt(0.9 + m), y = c = cbrt(0.9 + m);
45        a = counter[y], res = phi(m, a) + a - 1;
46        for (i = a; primes[i] <= s; i++) res = res - Lehmer(m / primes[i]) + Lehmer(primes[i]) - 1;
47        return res;
48    }
```

```

48     }
49 }
50 int main(){
51     pcf::init();
52     long long n;
53     while (scanf("%lld", &n) != EOF){
54         printf("%lld\n", pcf::Lehmer(n));
55     }
56     return 0;
57 }

```

## Cipolla 二次剩余

- $x^2 \equiv n \pmod{P}$
- 仅有两个解，返回小的那个，另一个是相反数。
- 大概 1s 能跑  $1e5$  个数

```

1  inline int qpow(int a, int b){
2      int q=1; while(b){if(b&1)q=1ll*q*a%P; a=1ll*a*a%P; b>>=1;} return q;
3  }
4  namespace Cipolla{
5      ll w, a;
6      struct node{
7          ll x, y;
8          node friend operator *(node x, node y){
9              node z;
10             z.x=(x.x*y.x%P+x.y*y.y%P*w%P)%P;
11             z.y=(x.x*y.y%P+x.y*y.x%P)%P;
12             return z;
13         }
14     }u, v;
15     inline node Cqpow(node a, ll b){
16         node q; q.x=1; q.y=0;
17         while(b){if(b&1) q=q*a; a=a*a; b>>=1;}
18         return q;
19     }
20     inline ll cipolla(int n){
21         n%=P; srand(0x20010412);
22         if(P==2) return n;
23         if(!n) return n;
24         if(qpow(n, (P-1)/2)==P-1) return -1;
25         while(1){
26             a=rand()%P;
27             w=(a*a-n+P)%P;
28             if(qpow(w%P, (P-1)/2)==P-1) break;
29         }
30         u.x=a, u.y=1;
31         u=Cqpow(u, (P+1)/2);
32         ll fir=u.x, sec=P-u.x;
33         if(fir>sec) swap(fir, sec);
34         return fir;
35     }
36 }

```

## BSGS

```

1  template<typename T>
2  struct BSGS {
3      T cal(T a, T b, T c) { // return  $a^x = b \pmod{c}$ ,  $\gcd(a, c) = 1$ 
4          mp.clear();
5          T tim = ceil(sqrt(c)), tmp = b%c;
6          for (int i = 0; i <= tim; i++) {
7              mp[tmp] = i; tmp = (__int128)tmp*a%c;
8          }
9          T t = tmp = quick_pow(a, tim, c);
10         for (int i = 1; i <= tim; i++) {
11             if (mp.count(tmp)) return tim*i-mp[tmp];
12             tmp = (__int128)tmp*t%c;
13         }
14         return -1;

```

```

15     }
16 } ;

```

## exBSGS

```

1  template<typename T>
2  struct exBSGS {
3      T cal(T a, T b, T c) { // return  $a^x = b \pmod c$ 
4          if (b == 1) return 0;
5          T cnt = 0, d = 1, t;
6          while ((t = __gcd(a, c)) != 1) {
7              if (b%t) return -1;
8              ++cnt, b /= t, c /= t, d = (__int128)d*(a/t)%c;
9              if (d == b) return cnt;
10         }
11         mp.clear();
12         T tim = ceil(sqrt(c)), tmp = b%c;
13         for (int i = 0; i <= tim; i++) {
14             mp[tmp] = i; tmp = (__int128)tmp*a%c;
15         }
16         t = tmp = quick_pow(a, tim, c); tmp = (__int128)tmp*d%c;
17         for (int i = 1; i <= tim; i++) {
18             if (mp.count(tmp)) return tim*i-mp[tmp]+cnt;
19             tmp = (__int128)tmp*t%c;
20         }
21         return -1;
22     }
23 };

1  template<typename T>
2  struct exLucas {
3      T quick_pow(T a, T b, T p) {
4          T ans = 1;
5          while (b) {
6              if (b&1) ans = (__int128)ans*a%p;
7              b >>= 1, a = (__int128)a*a%p;
8          }
9          return ans;
10     }
11     void ex_gcd(T a, T b, T &x, T &y) {
12         if (b == 0) {x = 1, y = 0; return;}
13         ex_gcd(b, a%b, x, y);
14         T t = x; x = y, y = t-a/b*y;
15     }
16     T inv(T a, T p) {
17         T x, y; ex_gcd(a, p, x, y);
18         return (x%p+p)%p;
19     }
20     T mul(T n, T pi, T pk) {
21         if (!n) return 1;
22         T ans = 1;
23         for (int i = 2; i <= pk; i++) if (i%pi != 0) ans = (__int128)ans*i%pk;
24         ans = quick_pow(ans, n/pk, pk);
25         for (int i = 2; i <= n%pk; i++) if (i%pi != 0) ans = (__int128)ans*i%pk;
26         return (__int128)ans*mul(n/pi, pi, pk)%pk;
27     }
28     T C(T n, T m, T pi, T pk, T p) {
29         T a = mul(n, pi, pk), b = mul(m, pi, pk), c = mul(n-m, pi, pk);
30         T k = 0;
31         for (T i = n; i; i /= pi) k += i/pi;
32         for (T i = m; i; i /= pi) k -= i/pi;
33         for (T i = n-m; i; i /= pi) k -= i/pi;
34         return (__int128)a*inv(b, pk)%pk*inv(c, pk)%pk*quick_pow(pi, k, pk)%pk;
35     }
36     T ex_lucas(T n, T m, T p) {
37         T ans = 0;
38         for (T i = 2, x = p; i <= x; i++)
39             if (x%i == 0) {
40                 T k = 1; while (x%i == 0) k *= i, x /= i;
41                 (ans += (__int128)C(n, m, i, k, p)*(p/k)%p*inv(p/k, k)%p) %= p;
42             }

```



```

43     return ans;
44 }
45 } ;

```

## 图论

### LCA

- 倍增

```

1 void dfs(int u, int fa) {
2     pa[u][0] = fa; dep[u] = dep[fa] + 1;
3     FOR (i, 1, SP) pa[u][i] = pa[pa[u][i - 1]][i - 1];
4     for (int& v: G[u]) {
5         if (v == fa) continue;
6         dfs(v, u);
7     }
8 }
9
10
11 int lca(int u, int v) {
12     if (dep[u] < dep[v]) swap(u, v);
13     int t = dep[u] - dep[v];
14     FOR (i, 0, SP) if (t & (1 << i)) u = pa[u][i];
15     FORD (i, SP - 1, -1) {
16         int uu = pa[u][i], vv = pa[v][i];
17         if (uu != vv) { u = uu; v = vv; }
18     }
19     return u == v ? u : pa[u][0];
20 }

```

## 计算几何

### 二维几何：点与向量

```

1 #define y1 yy1
2 #define nxt(i) ((i + 1) % s.size())
3 typedef double LD;
4 const LD PI = 3.14159265358979323846;
5 const LD eps = 1E-10;
6 int sgn(LD x) { return fabs(x) < eps ? 0 : (x > 0 ? 1 : -1); }
7 struct L;
8 struct P;
9 typedef P V;
10 struct P {
11     LD x, y;
12     explicit P(LD x = 0, LD y = 0): x(x), y(y) {}
13     explicit P(const L& l);
14 };
15 struct L {
16     P s, t;
17     L() {}
18     L(P s, P t): s(s), t(t) {}
19 };
20
21 P operator + (const P& a, const P& b) { return P(a.x + b.x, a.y + b.y); }
22 P operator - (const P& a, const P& b) { return P(a.x - b.x, a.y - b.y); }
23 P operator * (const P& a, LD k) { return P(a.x * k, a.y * k); }
24 P operator / (const P& a, LD k) { return P(a.x / k, a.y / k); }
25 inline bool operator < (const P& a, const P& b) {
26     return sgn(a.x - b.x) < 0 || (sgn(a.x - b.x) == 0 && sgn(a.y - b.y) < 0);
27 }
28 bool operator == (const P& a, const P& b) { return !sgn(a.x - b.x) && !sgn(a.y - b.y); }
29 P::P(const L& l) { *this = l.t - l.s; }
30 ostream &operator << (ostream &os, const P &p) {
31     return (os << "(" << p.x << ", " << p.y << ")");
32 }
33 istream &operator >> (istream &is, P &p) {

```

```

34     return (is >> p.x >> p.y);
35 }
36
37 LD dist(const P& p) { return sqrt(p.x * p.x + p.y * p.y); }
38 LD dot(const V& a, const V& b) { return a.x * b.x + a.y * b.y; }
39 LD det(const V& a, const V& b) { return a.x * b.y - a.y * b.x; }
40 LD cross(const P& s, const P& t, const P& o = P()) { return det(s - o, t - o); }
41 // -----

```

## 完整的板板 by zcs

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  const double pi=acos(-1.0); //高精度圆周率
4  const double eps=1e-8; //偏差值
5  const int maxp=200005; //点的数量
6  int sgn(double x) //判断 x 是否为 0
7  {
8      if(fabs(x)<eps) return 0;
9      else return x<0?-1:1; //小于 0 返回-1, 大于 0 返回 1
10 }
11 int Dcmp(double x,double y) //比较浮点数大小
12 {
13     if(fabs(x-y)<eps) return 0;
14     else return x<y?-1:1;
15 }
16 //-----平面几何: 点和线-----
17 struct Point
18 {
19     double x,y;
20     Point(){}
21     Point(double x,double y):x(x),y(y){}
22     Point operator + (Point B){return Point(x+B.x,y+B.y);}
23     Point operator - (Point B){return Point(x-B.x,y-B.y);}
24     Point operator * (double k){return Point(x*k,y*k);} //长度扩大 k 倍
25     Point operator / (double k){return Point(x/k,y/k);} //长度缩小 k 倍
26     bool operator == (Point B){return sgn(x-B.x)==0 && sgn(y-B.y)==0;}
27 };
28 typedef Point Vector;
29 double Dot(Vector A,Vector B){return A.x*B.x+A.y*B.y;} //向量点乘
30 double Len(Vector A){return sqrt(Dot(A,A));} //向量取模
31 double Angle(Vector A,Vector B){return acos(Dot(A,B)/Len(A)/Len(B));} //向量 A 和 B 夹角
32 double Cross(Vector A,Vector B){return A.x*B.y-A.y*B.x;} //向量叉乘
33 //double Area(Point A,Point B,Point C){return Cross(B-A,C-A);} //三角形面积 2 倍
34 double Distance(Point A,Point B){return hypot(A.x-B.x,A.y-B.y);} //两点距离
35 Vector Normal(Vector A){return Vector(-A.y/Len(A),A.x/Len(A));} //向量 A 的单位 * 法 * 向量
36 bool Parallel(Vector A,Vector B){return sgn(Cross(A,B))==0;} //平行
37 Vector Rotate(Vector A,double rad) //向量旋转
38 {return Vector(A.x*cos(rad)-A.y*sin(rad),A.x*sin(rad)+A.y*cos(rad));}
39 struct Line
40 {
41     Point p1,p2;
42     Line(){}
43     Line(Point p1,Point p2):p1(p1),p2(p2){} //两点确定直线
44     Line(Point p,double angle) //点 + 倾斜角
45     {
46         p1=p;
47         if(sgn(angle-pi/2)==0){p2=(p1+Point(0,1));}
48         else {p2=(p1+Point(1,tan(angle)));}
49     }
50     Line(double a,double b,double c) //ax+by+c=0;
51     {
52         if(sgn(a)==0) {p1=Point(0,-c/b);p2=Point(1,-c/b);}
53         else if(sgn(b)==0) {p1=Point(-c/a,0);p2=Point(-c/a,1);}
54         else {p1=Point(0,-c/b);p2=Point(1,(-c-a)/b);}
55     }
56 };
57 typedef Line Segment;
58 //直线倾斜角, 返回值 [0,pi);
59 double Line_angle(Line v)
60 {

```

```

61     double k=atan2(v.p2.y-v.p1.y,v.p2.x-v.p1.x);
62     if(sgn(k)<0) k+=pi;
63     if(sgn(k-pi)==0) k-=pi;
64     return k;
65 }
66 //点和直线关系
67 int Point_line_relation(Point p,Line v)
68 {
69     int c=sgn(Cross(p-v.p1,v.p2-v.p1));
70     if(c<0) return 1; //1:p 在 v 左侧
71     if(c>0) return 2; //2:p 在 v 右侧
72     return 0; //0:p 在 v 上
73 }
74 //点和线段关系: 0 为 p 不在线段 v 上; 1 为 p 在线段 v 上
75 bool Point_on_seg(Point p,Segment v)
76 {
77     return sgn(Cross(p-v.p1,v.p2-v.p1))==0 && sgn(Dot(p-v.p1,p-v.p2))<=0;
78 }
79 //两直线的关系: 0 为平行, 1 为重合, 2 为相交
80 int Line_relation(Line v1,Line v2)
81 {
82     if(sgn(Cross(v1.p2-v1.p1,v2.p2-v2.p1))==0)
83     {
84         if(Point_line_relation(v1.p1,v2)==0) return 1;
85         else return 0;
86     }
87     return 2;
88 }
89 //点到直线距离
90 double Dis_point_line(Point p,Line v)
91 {
92     return fabs(Cross(p-v.p1,v.p2-v.p1))/Distance(v.p1,v.p2);
93 }
94 //点在直线上的投影
95 Point Point_line_proj(Point p,Line v)
96 {
97     double k=Dot(v.p2-v.p1,p-v.p1)/Dot(v.p2-v.p1,v.p2-v.p1);
98     return v.p1+(v.p2-v.p1)*k;
99 }
100 //点 p 对直线 v 的对称点
101 Point Point_line_symmetry(Point p,Line v)
102 {
103     Point q=Point_line_proj(p,v);
104     return Point(2*q.x-p.x,2*q.y-p.y);
105 }
106 //点到线段的距离
107 double Dis_point_seg(Point p,Segment v)
108 {
109     if(sgn(Dot(p-v.p1,v.p2-v.p1))<0 || sgn(Dot(p-v.p2,v.p1-v.p2))<0)
110         return min(Distance(p,v.p1),Distance(p,v.p2)); //点的投影不在线段上
111     return Dis_point_line(p,v); //点的投影在线段上
112 }
113 //求两直线 ab 和 cd 的交点, 在调用前要保证两直线不平行或重合
114 Point Cross_point(Point a,Point b,Point c,Point d)
115 {
116     double s1=Cross(b-a,c-a);
117     double s2=Cross(b-a,d-a);
118     return Point(c.x*s2-d.x*s1,c.y*s2-d.y*s1)/(s2-s1);
119 }
120 //线段 ab 和 cd 是否相交
121 bool Cross_segment(Point a,Point b,Point c,Point d)
122 {
123     double c1=Cross(b-a,c-a),c2=Cross(b-a,d-a);
124     double d1=Cross(d-c,a-c),d2=Cross(d-c,b-c);
125     return sgn(c1)*sgn(c2)<=0 && sgn(d1)*sgn(d2)<=0;
126 }
127 //-----平面几何: 多边形-----
128 struct Polygon
129 {
130     int n;
131     Point p[maxp]; //从 0 开始

```

```

132     Line v[maxp];
133 };
134 //极角排序
135 bool Polar_angle_cmp(Point a,Point b)
136 {
137     if(Cross(a,b)==0) return a.x<b.x;
138     else return Cross(a,b)>0;
139 }
140 //按照 x 大小排序 (计算凸包使用)
141 bool Hull_cmp(Point A,Point B)
142 {
143     return sgn(A.x-B.x)<0 || (sgn(A.x-B.x)==0 && sgn(A.y-B.y)<0);
144 }
145 //判断点和任意多边形的关系: 3 为点上; 2 为边上; 1 为内部; 0 为外部
146 int Point_in_polygon(Point pt,Point *p,int n) //点 pt, 多边形 *p
147 {
148     for(int i=0;i<n;i++)
149         if(p[i]==pt) return 3;
150     for(int i=0;i<n;i++)
151     {
152         Line v=Line(p[i],p[(i+1)%n]);
153         if(Point_on_seg(pt,v)) return 2;
154     }
155     int num=0;
156     for(int i=0;i<n;i++)
157     {
158         int j=(i+1)%n;
159         int c=sgn(Cross(pt-p[j],p[i]-p[j]));
160         int u=sgn(p[i].y-pt.y);
161         int v=sgn(p[j].y-pt.y);
162         if(c>0 && u<0 && v>=0) num++;
163         if(c<0 && u>=0 && v<0) num--;
164     }
165     return num!=0;
166 }
167 //多边形面积
168 double Polygon_area(Point *p,int n)
169 {
170     double area=0;
171     for(int i=0;i<n;i++)
172         area+=Cross(p[i],p[(i+1)%n]);
173     return area/2;
174 }
175 //求多边形重心
176 Point Polygon_center(Point *p,int n)
177 {
178     Point ans(0,0);
179     if(Polygon_area(p,n)==0) return ans;
180     for(int i=0;i<n;i++)
181         ans=ans+(p[i]+p[(i+1)%n])*Cross(p[i],p[(i+1)%n]);
182     return ans/Polygon_area(p,n)/6;
183 }
184 //Convex_hull() 求凸包, 凸包顶点放在 ch 中, 返回值是凸包的顶点数
185 int Convex_hull(Point *p,int n,Point *ch)
186 {
187     sort(p,p+n,Hull_cmp);
188     n=unique(p,p+n)-p;
189     int v=0;
190     //求下凸包, 如果 p[i] 是右拐的, 则不在凸包上, 往回退
191     for(int i=0;i<n;i++)
192     {
193         while(v>1 && sgn(Cross(ch[v-1]-ch[v-2],p[i]-ch[v-2]))<=0)
194             v--;
195         ch[v++]=p[i];
196     }
197     int j=v;
198     //求上凸包
199     for(int i=n-2;i>=0;i--)
200     {
201         while(v>j && sgn(Cross(ch[v-1]-ch[v-2],p[i]-ch[v-2]))<=0)
202             v--;

```

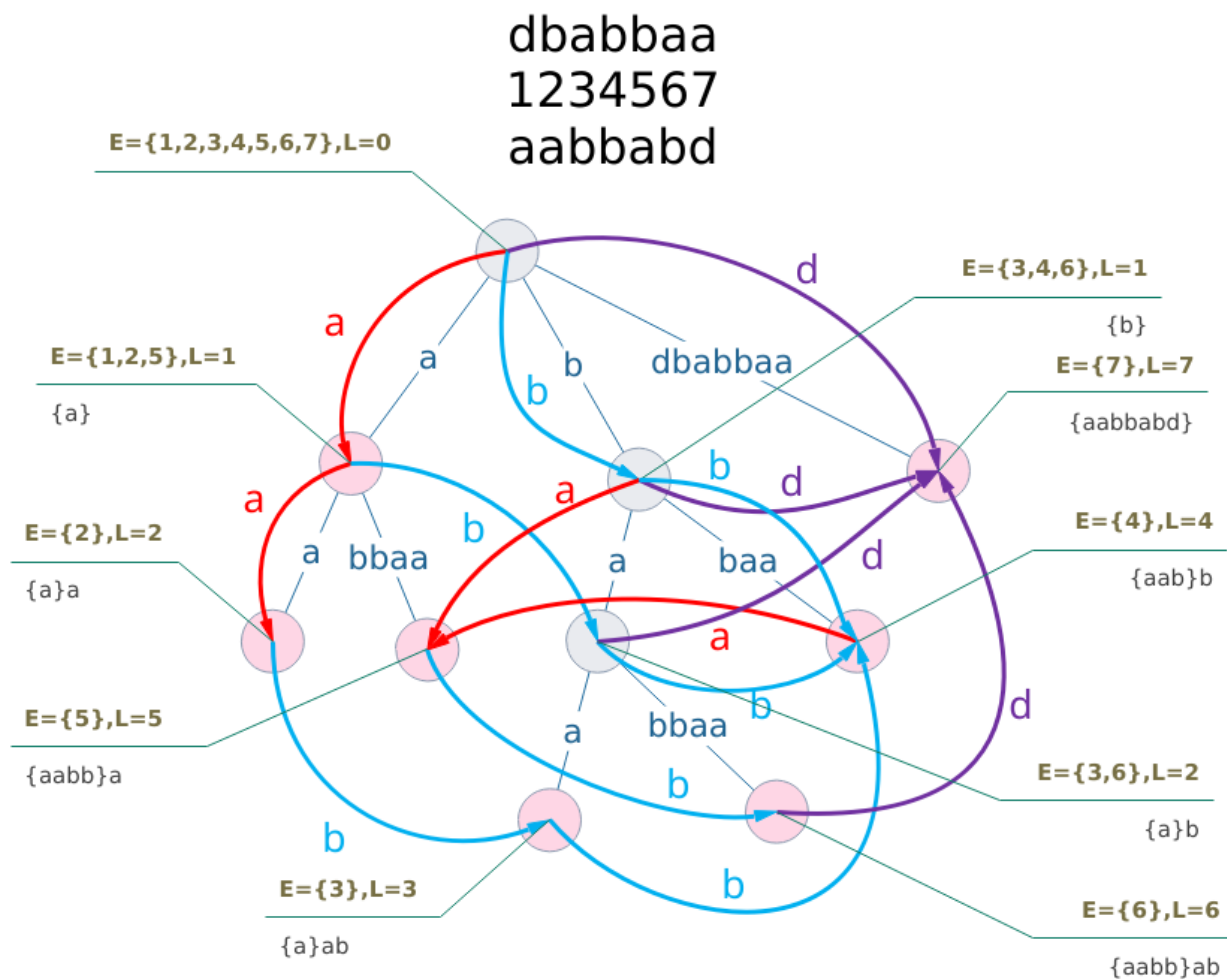
```

203         ch[v++]=p[i];
204     }
205     if(n>1) v--;
206     return v;
207 }
208 //-----平面几何: 圆-----
209 struct Circle
210 {
211     Point c;                //圆心
212     double r;
213     Circle(){}
214     Circle(Point c,double r):c(c),r(r){}
215     Circle(double x,double y,double _r){c=Point(x,y);r=_r;}
216 };
217 //点和圆的关系: 0 为圆内, 1 为圆上, 2 为圆外
218 int Point_circle_relation(Point p,Circle C)
219 {
220     double dst=Distance(p,C.c);
221     if(sgn(dst-C.r)<0) return 0;
222     if(sgn(dst-C.r)==0) return 1;
223     return 2;
224 }
225 //直线和圆的关系: 0 为直线和圆相交, 1 为直线和圆相切, 2 为直线和圆相离
226 int Line_circle_relation(Line v,Circle C)
227 {
228     double dst=Dis_point_line(C.c,v);
229     if(sgn(dst-C.r)<0) return 0;
230     if(sgn(dst-C.r)==0) return 1;
231     return 2;
232 }
233 //线段和圆的关系: 0 为线段和圆相交, 1 为线段和圆相切, 2 为线段和圆相离
234 int Seg_circle_relation(Segment v,Circle C)
235 {
236     double dst=Dis_point_seg(C.c,v);
237     if(sgn(dst-C.r)<0) return 0;
238     if(sgn(dst-C.r)==0) return 1;
239     return 2;
240 }
241 //直线和圆的交点, pa,pb 是交点, 返回值是交点个数
242 int Line_cross_circle(Line v,Circle C,Point &pa,Point &pb)
243 {
244     if(Line_circle_relation(v,C)==2) return 0;                //无交点
245     Point q=Point_line_proj(C.c,v);
246     double d=Dis_point_line(C.c,v);
247     double k=sqrt(C.r*C.r-d*d);
248     if(sgn(k)==0)
249     {
250         pa=q;pb=q;return 1;
251     }
252     Point n=(v.p2-v.p1)/Len(v.p2-v.p1);                //直线的单位向量
253     pa=q+n*k;
254     pb=q-n*k;
255     return 2;
256 }

```

## 字符串

### 后缀自动机



## 多项式

### NTT 模数

```
1 NTTPrimes = {1053818881, 1051721729, 1045430273, 1012924417, 1007681537, 1004535809, 998244353, 985661441,  
  ↪ 976224257, 975175681};  
2 NTTPrimitiveRoots = {7, 6, 3, 5, 3, 3, 3, 3, 3, 17};
```

### FFT

```
1 namespace FFT{  
2     const db pi=acos(-1);  
3     struct cp{  
4         db re,im;  
5         cp(db _re=0,db _im=0){re=_re;im=_im;}  
6         cp operator +(cp b){return cp(re+b.re,im+b.im);}  
7         cp operator -(cp b){return cp(re-b.re,im-b.im);}  
8         cp operator *(cp b){return cp(re*b.re-im*b.im,re*b.im+im*b.re);}  
9     };  
10    int r[N];cp c[N<<1];  
11    inline void fft(cp *a,int f,int n){  
12        rep(i,0,n-1) if(r[i]>i) swap(a[r[i]],a[i]);  
13        for(int i=1;i<n;i<=<1){
```

```

14         cp wn(cos(pi/i),f*sin(pi/i));
15         for(int j=0,p=(i<<1);j<n;j+=p){
16             cp w(1,0);
17             for(int k=0;k<i;++k,w=w*wn){
18                 cp x=a[j+k],y=w*a[j+k+i];
19                 a[j+k]=x+y;a[j+k+i]=x-y;
20             }
21         }
22     }
23     if(f==-1){rep(i,0,n-1) a[i].re/=n,a[i].im/=n;}
24 }
25 inline int mul(db *a,db *b,int n,int m){
26     n+=m;rep(i,0,n) c[i]=cp(a[i],b[i]);
27     int l=0;m=n;for(n=1;n<=m;n<<=1) ++l;
28     rep(i,0,n-1) r[i]=(r[i>>1]>>1)|((i&1)<<(l-1));
29     rep(i,m+1,n) c[i]=cp(0,0);
30     fft(c,1,n);rep(i,0,n-1) c[i]=c[i]*c[i];
31     fft(c,-1,n);
32     rep(i,0,m) a[i]=c[i].im/2;
33     return n;
34 }
35 }

```

## NTT

```

1 namespace NTT{
2     const int P=998244353,g=3,ig=332748118;
3     inline int qpow(int a,int b){int q=1;while(b){if(b&1)q=1LL*q*a%P;a=1LL*a*a%P;b>>=1;}return q;}
4     int r[N],ow[N],inv[N];
5     inline void ntt(int *a,int f,int n){
6         rep(i,0,n-1) if(r[i]>i) swap(a[i],a[r[i]]);
7         for(int i=1;i<n;i<<=1){
8             int wn=qpow(f,(P-1)/(i<<1));
9             ow[0]=1;rep(k,1,i-1) ow[k]=1LL*ow[k-1]*wn%P;
10            for(int j=0,p=(i<<1);j<n;j+=p){
11                for(int k=0;k<i;++k){
12                    int x=a[j+k],y=1LL*ow[k]*a[j+k+i]%P;
13                    a[j+k]=(x+y)%P;a[j+k+i]=(x-P-y)%P;
14                }
15            }
16        }
17        if(f==ig){
18            int iv=qpow(n,P-2);
19            rep(i,0,n-1) a[i]=1LL*a[i]*iv%P;
20        }
21    }
22    int tma[N],tmb[N];
23    inline int mul(int *a,int *b,int n,int m,int ci){
24        int _n=n,_m=m,l=0;m+=n;for(n=1;n<=m;n<<=1) ++l;
25        rep(i,0,n-1) r[i]=(r[i>>1]>>1)|((i&1)<<(l-1));
26        rep(i,0,n-1) tma[i]=a[i];rep(i,0,n-1) tmb[i]=b[i];
27        rep(i,_n+1,n) tma[i]=0;rep(i,_m+1,n) tmb[i]=0;
28        ntt(tma,g,n);ntt(tmb,g,n);
29        while(ci){
30            if(ci&1) rep(i,0,n-1) tma[i]=1LL*tma[i]*tmb[i]%P;
31            rep(i,0,n-1) tmb[i]=1LL*tmb[i]*tmb[i]%P;
32            ci>>=1;
33        }
34        ntt(tma,ig,n);
35        rep(i,0,n-1) a[i]=tma[i];
36        return n;
37    }
38 }
39 inline void prepare(){
40     //NTT inv
41     using NTT::inv;using NTT::P;
42     inv[1]=1;rep(i,2,N-1) inv[i]=1LL*(P-P/i)*inv[P%i]%P;
43 }

```

## 完整的板板 by hls

```
1  /*
2      NTTPrimes = {1053818881, 1051721729, 1045430273, 1012924417, 1007681537, 1004535809, 998244353, 985661441,
3      ↪ 976224257, 975175681};
4      NTTPrimitiveRoots = {7, 6, 3, 5, 3, 3, 3, 3, 17};
5  */
6  //poly start
7  namespace Poly{
8      const int N=6e5+10;
9      namespace NTT{
10         const int P=998244353,g=3,ig=332748118;
11         inline int qpow(int a,int b){int q=1;while(b){if(b&1)q=1LL*q*a%P;a=1LL*a*a%P;b>>=1;}return q;}
12         int r[N],ow[N],inv[N];
13         inline void ntt(int *a,int f,int n){
14             rep(i,0,n-1) if(r[i]>i) swap(a[i],a[r[i]]);
15             for(int i=1;i<n;i<=1){
16                 int wn=qpow(f,(P-1)/(i<<1));
17                 ow[0]=1;rep(k,1,i-1) ow[k]=1LL*ow[k-1]*wn%P;
18                 for(int j=0,p=(i<<1);j<n;j+=p){
19                     for(int k=0;k<i;++k){
20                         int x=a[j+k],y=1LL*ow[k]*a[j+k+i]%P;
21                         a[j+k]=(x+y)%P;a[j+k+i]=(x-y)%P;
22                     }
23                 }
24                 if(f==ig){
25                     int iv=qpow(n,P-2);
26                     rep(i,0,n-1) a[i]=1LL*a[i]*iv%P;
27                 }
28             }
29             int tma[N],tmb[N];
30             inline int mul(int *a,int *b,int n,int m,int ci){
31                 int _n=n,_m=m,l=0;m+=n;for(n=1;n<=m;n<=1) ++l;
32                 rep(i,0,n-1) r[i]=(r[i]>>1)>>1|((i&1)<<(l-1));
33                 rep(i,0,n-1) tma[i]=a[i];rep(i,0,n-1) tmb[i]=b[i];
34                 rep(i,_n+1,n) tma[i]=0;rep(i,_m+1,n) tmb[i]=0;
35                 ntt(tma,g,n);ntt(tmb,g,n);
36                 while(ci){
37                     if(ci&1) rep(i,0,n-1) tma[i]=1LL*tma[i]*tmb[i]%P;
38                     rep(i,0,n-1) tmb[i]=1LL*tmb[i]*tmb[i]%P;
39                     ci>>=1;
40                 }
41                 ntt(tma,ig,n);
42                 rep(i,0,n-1) a[i]=tma[i];
43                 return n;
44             }
45         }
46         namespace FFT{
47             const db pi=acos(-1);
48             struct cp{
49                 db re,im;
50                 cp(db _re=0,db _im=0){re=_re;im=_im;}
51                 cp operator +(cp b){return cp(re+b.re,im+b.im);}
52                 cp operator -(cp b){return cp(re-b.re,im-b.im);}
53                 cp operator *(cp b){return cp(re*b.re-im*b.im,re*b.im+im*b.re);}
54             };
55             int r[N];cp c[N<<1];
56             inline void fft(cp *a,int f,int n){
57                 rep(i,0,n-1) if(r[i]>i) swap(a[r[i]],a[i]);
58                 for(int i=1;i<n;i<=1){
59                     cp wn(cos(pi/i),f*sin(pi/i));
60                     for(int j=0,p=(i<<1);j<n;j+=p){
61                         cp w(1,0);
62                         for(int k=0;k<i;++k,w=w*wn){
63                             cp x=a[j+k],y=w*a[j+k+i];
64                             a[j+k]=x+y;a[j+k+i]=x-y;
65                         }
66                     }
67                 }
68                 if(f==1){rep(i,0,n-1) a[i].re/=n,a[i].im/=n;}
```



```

69     }
70     inline int mul(db *a,db *b,int n,int m){
71         n+=m;rep(i,0,n) c[i]=cp(a[i],b[i]);
72         int l=0;m=n;for(n=1;n<=m;n<=1) ++l;
73         rep(i,0,n-1) r[i]=(r[i]>>1)>>1|((i&1)<<(l-1));
74         rep(i,m+1,n) c[i]=cp(0,0);
75         fft(c,1,n);rep(i,0,n-1) c[i]=c[i]*c[i];
76         fft(c,-1,n);
77         rep(i,0,m) a[i]=c[i].im/2;
78         return n;
79     }
80 }
81 using namespace NTT;
82 ll w,a;
83 struct node{
84     ll x,y;
85     node friend operator *(node x,node y){
86         node z;
87         z.x=(x.x*y.x%P+x.y*y.y%P*w%P)%P;
88         z.y=(x.x*y.y%P+x.y*y.x%P)%P;
89         return z;
90     }
91 }u,v;
92 inline node Cqpow(node a,ll b){
93     node q;q.x=1;q.y=0;
94     while(b){if(b&1) q=q*a;a=a*a;b>>=1;}
95     return q;
96 }
97 inline ll cipolla(int n,int P){
98     n%=P;rand(0x20010412);
99     if(P==2) return 1;
100    if(qpow(n,(P-1)/2)==P-1) return -1;
101    while(1){
102        a=rand()%P;
103        w=(a*a-n+P)%P;
104        if(qpow(w%P,(P-1)/2)==P-1) break;
105    }
106    u.x=a,u.y=1;
107    u=Cqpow(u,(P+1)/2);
108    ll fir=u.x,sec=P-u.x;
109    if(fir>sec)swap(fir,sec);
110    return fir;
111 }
112 inline void derivative(int *a,int *b,int n){
113     rpe(i,n-2,0) b[i]=1LL*a[i+1]*(i+1)%P;
114     b[n-1]=0;
115 }
116 inline void integral(int *a,int *b,int n){
117     rpe(i,n-1,1) b[i]=1LL*a[i-1]*inv[i]%P;
118     b[0]=0;
119 }
120 inline void differential(int *a,int *b,int n){
121     rep(i,1,n-1) b[i]=(a[i]+P-a[i-1])%P;
122     b[0]=0;
123 }
124 int tf[N],tg[N];
125 inline void inverse(int *f,int *g,int n){
126     if(n==1){
127         g[0]=qpow(f[0],P-2);return;
128     }
129     inverse(f,g,n>>1);
130     rep(i,0,n-1) tf[i]=f[i],tg[i]=g[i];
131     int tmp=mul(tf,tg,n-1,n-1,2);
132     rep(i,0,n-1) g[i]=((-tf[i]+2LL*tmp)%P)%P;
133     rep(i,0,tmp) tf[i]=tg[i]=0;
134 }
135 int ta[N],tb[N];
136 inline void sqrt(int *a,int *b,int n){
137     if(n==1){
138         b[0]=cipolla(a[0],P);//debug(b[0]);debug(1LL*b[0]*b[0]%P);
139         return;

```

```

140     }
141     sqrt(a,b,n>>1);rep(i,n,(n<<1)) b[i]=0;
142     inverse(b,tb,n);
143     rep(i,0,n-1) ta[i]=a[i];
144     mul(ta,tb,n-1,n-1,1);//debug(ta[0]);debug(b[0]);
145     rep(i,0,n-1) b[i]=1LL*(b[i]+ta[i])%P*inv[2]%P;
146     rep(i,0,n<<1) ta[i]=tb[i]=0;
147 }
148 inline void ln(int *a,int *b,int n){
149     inverse(a,ta,n);
150     derivative(a,b,n);
151     mul(b,ta,n,n,1);
152     integral(b,b,n);
153 }
154 inline void exp(int *a,int *b,int n){
155     if(n==1){
156         b[0]=1;return;
157     }
158     exp(a,b,n>>1);
159     ln(b,tb,n);rep(i,0,n-1) ta[i]=a[i];++ta[0];
160     rep(i,0,n-1) ta[i]-=tb[i];
161     mul(ta,b,n,n,1);rep(i,0,n-1) b[i]=ta[i];
162     rep(i,0,n<<1) ta[i]=tb[i]=0;
163 }
164 }
165 //
166 using namespace Poly;
167 inline void prepare(){
168     //NTT
169     using NTT::inv;using NTT::P;
170     inv[1]=1;rep(i,2,N-1) inv[i]=1LL*(P-P/i)*inv[P%i]%P;
171 }
172 int n,A[N],B[N];
173 int main(){
174     prepare();
175     n=read();rep(i,0,n) A[i]=read();
176     int len=1;for(;len<=n;len<<=1);
177 /*
178     sqrt(A,B,len);
179     rep(i,0,n) printf("%d ",B[i]);pts;
180     //     debug(1LL*B[0]*B[0]%P);
181     NTT::mul(B,B,n,n,1);
182     rep(i,0,n) printf("%d ",B[i]);pts;
183 */ //sqrt 1e5 uoj 333ms
184 /*
185     ln(A,B,len);
186     rep(i,0,n) printf("%d ",B[i]);pts;
187     exp(B,A,len);
188     rep(i,0,n) printf("%d ",A[i]);pts;
189 */ //ln&exp 1e5 uoj 766ms
190     //ln 222ms
191     //exp 568ms
192     return 0;
193 }

```