ICPC Notebook

template

hash.sh

```
# 使い方: sh hash.sh -> コピペ -> Ctrl + D
# コメント・空白・改行を削除して md5 でハッシュする
g++ -dD -E -P -fpreprocessed - | tr -d '[:space:]' | md5sum |
cut -c-6
```

settings.sh

```
# CLion の設定
Settings → Build → CMake → Reload CMake Project
add_compile_options(-D_GLIBCXX_DEBUG)
# Caps Lock を Ctrl に変更
setxkbmap -option ctrl:nocaps
```

template.hpp

md5: 136d85

```
#include <bits/stdc++.h>
using namespace std;
using ll = long long;
const ll INF = LLONG_MAX / 4;
#define rep(i, a, b) for(ll i = a; i < (b); i++)
#define all(a) begin(a), end(a)
#define sz(a) ssize(a)
bool chmin(auto& a, auto b) { return a > b ? a = b, 1 : 0; }
bool chmax(auto& a, auto b) { return a < b ? a = b, 1 : 0; }

int main() {
   cin.tie(0)->sync_with_stdio(0);
   // your code here...
}
```

data-structure

BIT.hpp

md5: 8133c8

```
struct BIT {
   vector<ll> a;
   BIT(ll n) : a(n + 1) {}
   void add(ll i, ll x) {  // A[i] += x
      i++;
      while(i < sz(a)) {
         a[i] += x;
         i += i & -i;
      }
   }
   ll sum(ll r) {
      ll s = 0;
      while(r) {
         s += a[r];
         r -= r & -r;
      }
      return s;
   }
   ll sum(ll l, ll r) {  // sum of A[l, r)
      return sum(r) - sum(l);
   }
};
```

FastSet.hpp

md5: 2cb8c9

```
// using u64 = uint64_t;
const u64 B = 64;
struct FastSet {
   u64 n;
   vector<vector<u64>> a;
   FastSet(u64 n_) : n(n_) {
      do a.emplace_back(n_ = (n_ + B - 1) / B);
      while(n_ > 1);
   }
   // bool operator[](ll i) const { return a[0][i / B] >> (i %
B) & 1; }
   void set(ll i) {
      for(auto& v : a) {
         v[i / B] |= 1ULL << (i % B);
         i /= B;
```

```cpp
      }
   }
   void reset(ll i) {
      for(auto& v : a) {
         v[i / B] &= ~(1ULL << (i % B));
         if(v[i / B]) break;
         i /= B;
      }
   }
   ll next(ll i) {  // i を超える最小の要素
      rep(h, 0, sz(a)) {
         i++;
         if(i / B >= sz(a[h])) break;
         u64 d = a[h][i / B] >> (i % B);
         if(d) {
            i += countr_zero(d);
            while(h--) i = i * B + countr_zero(a[h][i]);
            return i;
         }
         i /= B;
      }
      return n;
   }
   ll prev(ll i) {  // i より小さい最大の要素
      rep(h, 0, sz(a)) {
         i--;
         if(i < 0) break;
         u64 d = a[h][i / B] << (~i % B);
         if(d) {
            i -= countl_zero(d);
            while(h--) i = i * B + __lg(a[h][i]);
            return i;
         }
         i /= B;
      }
      return -1;
   }
};
```

lazy_segtree.hpp

md5: c63b46

```cpp
unsigned int bit_ceil(unsigned int n) {
   unsigned int x = 1;
   while(x < (unsigned int)(n)) x *= 2;
   return x;
}
int countr_zero(unsigned int n) { return __builtin_ctz(n); }
constexpr int countr_zero_constexpr(unsigned int n) {
   int x = 0;
   while(!(n & (1 << x))) x++;
   return x;
}
template<class S, S (*op)(S, S), S (*e)(), class F, S
(*mapping)(F, S), F (*composition)(F, F), F (*id)()>
struct lazy_segtree {
   public:
   lazy_segtree() : lazy_segtree(0) {}
   explicit lazy_segtree(int n) : lazy_segtree(vector<S>(n,
e())) {}
   explicit lazy_segtree(const vector<S>& v) :
_n(int(v.size())) {
      size = (int)bit_ceil((unsigned int)(_n));
      log = countr_zero((unsigned int)size);
      d = vector<S>(2 * size, e());
      lz = vector<F>(size, id());
      for(int i = 0; i < _n; i++) d[size + i] = v[i];
      for(int i = size - 1; i >= 1; i--) { update(i); }
   }
   void set(int p, S x) {
      // assert(0 <= p && p < _n);
      p += size;
      for(int i = log; i >= 1; i--) push(p >> i);
      d[p] = x;
      for(int i = 1; i <= log; i++) update(p >> i);
   }

   S get(int p) {
      // assert(0 <= p && p < _n);
      p += size;
      for(int i = log; i >= 1; i--) push(p >> i);
      return d[p];
   }
```

```cpp
   S prod(int l, int r) {
      // assert(0 <= l && l <= r && r <= _n);
      if(l == r) return e();

      l += size;
      r += size;

      for(int i = log; i >= 1; i--) {
         if(((l >> i) << i) != l) push(l >> i);
         if(((r >> i) << i) != r) push((r - 1) >> i);
      }

      S sml = e(), smr = e();
      while(l < r) {
         if(l & 1) sml = op(sml, d[l++]);
         if(r & 1) smr = op(d[--r], smr);
         l >>= 1;
         r >>= 1;
      }

      return op(sml, smr);
   }

   void apply(int l, int r, F f) {
      assert(0 <= l && l <= r && r <= _n);
      if(l == r) return;

      l += size;
      r += size;

      for(int i = log; i >= 1; i--) {
         if(((l >> i) << i) != l) push(l >> i);
         if(((r >> i) << i) != r) push((r - 1) >> i);
      }

      {
         int l2 = l, r2 = r;
         while(l < r) {
            if(l & 1) all_apply(l++, f);
            if(r & 1) all_apply(--r, f);
            l >>= 1;
            r >>= 1;
         }
         l = l2;
         r = r2;
      }

      for(int i = 1; i <= log; i++) {
         if(((l >> i) << i) != l) update(l >> i);
         if(((r >> i) << i) != r) update((r - 1) >> i);
      }
   }

   int _n, size, log;
   std::vector<S> d;
   std::vector<F> lz;

   void update(int k) { d[k] = op(d[2 * k], d[2 * k + 1]); }
   void all_apply(int k, F f) {
      d[k] = mapping(f, d[k]);
      if(k < size) lz[k] = composition(f, lz[k]);
   }
   void push(int k) {
      all_apply(2 * k, lz[k]);
      all_apply(2 * k + 1, lz[k]);
      lz[k] = id();
   }
};
```

lazy_segtree_bsearch.hpp

md5: 18d7a2

```cpp
template<class S, S (*op)(S, S), S (*e)(), class F, S
(*mapping)(F, S), F (*composition)(F, F), F (*id)(), class G>
int max_right(lazy_segtree<S, op, e, F, mapping, composition,
id>& seg, int l, G g) {
   auto& _n = seg._n;
   auto& size = seg.size;
   auto& d = seg.d;
   auto& log = seg.log;
   // assert(0 <= l && l <= _n);
   // assert(g(e()));
   if(l == _n) return _n;
   l += size;
   for(int i = log; i >= 1; i--) seg.push(l >> i);
   S sm = e();
   do {
      while(l % 2 == 0) l >>= 1;
      if(!g(op(sm, d[l]))) {
         while(l < size) {
            seg.push(l);
            l = (2 * l);
            if(g(op(sm, d[l]))) {
               sm = op(sm, d[l]);
               l++;
            }
         }
         return l - size;
      }
      sm = op(sm, d[l]);
      l++;
   } while((l & -l) != l);
   return _n;
}

template<class S, S (*op)(S, S), S (*e)(), class F, S
(*mapping)(F, S), F (*composition)(F, F), F (*id)(), class G>
int min_left(lazy_segtree<S, op, e, F, mapping, composition,
id>& seg, int r, G g) {
   auto& _n = seg._n;
   auto& size = seg.size;
   auto& d = seg.d;
   auto& log = seg.log;
   // assert(0 <= r && r <= _n);
   // assert(g(e()));
   if(r == 0) return 0;
   r += size;
   for(int i = log; i >= 1; i--) seg.push((r - 1) >> i);
   S sm = e();
   do {
      r--;
      while(r > 1 && (r % 2)) r >>= 1;
      if(!g(op(d[r], sm))) {
         while(r < size) {
            seg.push(r);
            r = (2 * r + 1);
            if(g(op(d[r], sm))) {
               sm = op(d[r], sm);
               r--;
            }
         }
         return r + 1 - size;
      }
      sm = op(d[r], sm);
   } while((r & -r) != r);
   return 0;
}
```

segtree.hpp

md5: 88868b

```
unsigned int bit_ceil(unsigned int n) {
   unsigned int x = 1;
   while(x < (unsigned int)(n)) x *= 2;
   return x;
}
int countr_zero(unsigned int n) { return __builtin_ctz(n); }
constexpr int countr_zero_constexpr(unsigned int n) {
   int x = 0;
   while(!(n & (1 << x))) x++;
   return x;
}
template<class S, S (*op)(S, S), S (*e)()> struct segtree {
   public:
   segtree() : segtree(0) {}
   explicit segtree(int n) : segtree(vector<S>(n, e())) {}
   explicit segtree(const vector<S>& v) : _n(int(v.size())) {
      size = (int)bit_ceil((unsigned int)(_n));
      log = countr_zero((unsigned int)size);
      d = vector<S>(2 * size, e());
      for(int i = 0; i < _n; i++) d[size + i] = v[i];
      for(int i = size - 1; i >= 1; i--) { update(i); }
   }

   void set(int p, S x) {
      // assert(0 <= p && p < _n);
      p += size;
      d[p] = x;
      for(int i = 1; i <= log; i++) update(p >> i);
   }

   S get(int p) const {
      // assert(0 <= p && p < _n);
      return d[p + size];
   }

   S prod(int l, int r) const {
      // assert(0 <= l && l <= r && r <= _n);
      S sml = e(), smr = e();
      l += size;
      r += size;

      while(l < r) {
         if(l & 1) sml = op(sml, d[l++]);
         if(r & 1) smr = op(d[--r], smr);
         l >>= 1;
         r >>= 1;
      }
      return op(sml, smr);
   }

   S all_prod() const { return d[1]; }

   int _n, size, log;
   vector<S> d;

   void update(int k) { d[k] = op(d[2 * k], d[2 * k + 1]); }
};
```

segtree_bsearch.hpp

md5: 680c18

```
template<class S, S (*op)(S, S), S (*e)(), class F> int
max_right(segtree<S, op, e>& seg, int l, F f) {
   auto& _n = seg._n;
   auto& size = seg.size;
   auto& d = seg.d;
   // assert(0 <= l && l <= _n);
   // assert(f(e()));
   if(l == _n) return _n;
   l += size;
   S sm = e();
   do {
      while(l % 2 == 0) l >>= 1;
      if(!f(op(sm, d[l]))) {
         while(l < size) {
            l = (2 * l);
            if(f(op(sm, d[l]))) {
               sm = op(sm, d[l]);
               l++;
            }
         }
         return l - size;
      }
      sm = op(sm, d[l]);
      l++;
   } while((l & -l) != l);
   return _n;
}

template<class S, S (*op)(S, S), S (*e)(), class F> int
min_left(segtree<S, op, e>& seg, int r, F f) {
   auto& _n = seg._n;
   auto& size = seg.size;
   auto& d = seg.d;
   // assert(0 <= r && r <= _n);
   // assert(f(e()));
   if(r == 0) return 0;
   r += size;
   S sm = e();
   do {
      r--;
      while(r > 1 && (r % 2)) r >>= 1;
      if(!f(op(d[r], sm))) {
         while(r < size) {
            r = (2 * r + 1);
            if(f(op(d[r], sm))) {
               sm = op(d[r], sm);
               r--;
            }
         }
         return r + 1 - size;
      }
      sm = op(d[r], sm);
   } while((r & -r) != r);
   return 0;
}
```

math

BinaryGCD.hpp md5: f3ab31

```
u64 ctz(u64 x) { return countr_zero(x); }
u64 binary_gcd(u64 x, u64 y) {
   if(!x || !y) return x | y;
   u64 n = ctz(x), m = ctz(y);
   x >>= n, y >>= m;
   while(x != y) {
      if(x > y) x = (x - y) >> ctz(x - y);
      else y = (y - x) >> ctz(y - x);
   }
   return x << min(n, m);
}
```

ExtGCD.hpp md5: c3fa9b

```
// returns gcd(a, b) and assign x, y to integers
// s.t. ax + by = gcd(a, b) and |x| + |y| is minimized
ll extgcd(ll a, ll b, ll& x, ll& y) {
   // assert(a >= 0 && b >= 0);
   if(!b) return x = 1, y = 0, a;
   ll d = extgcd(b, a % b, y, x);
   y -= a / b * x;
   return d;
}
```

floor_sum.hpp md5: 0f7242

```
ll floor_sum(const ll& n, const ll& m, ll a, ll b) {
   ll ret = 0;
   if(a >= m) ret += (n - 1) * n * (a / m) / 2, a %= m;
   if(b >= m) ret += n * (b / m), b %= m;
   ll y = (a * n + b) / m;
   if(y == 0) return ret;
   ll x = y * m - b;
   ret += (n - (x + a - 1) / a) * y;
   ret += floor_sum(y, a, m, (a - x % a) % a);
   return ret;
}
```

modint

BarrettReduction.hpp md5: 2ca7f3

```
// using u64 = uint64_t;
struct Barrett {  // mod < 2^32
   u64 m, im;
   Barrett(u64 mod) : m(mod), im(-1ULL / m + 1) {}
   // input: a * b < 2^64, output: a * b % mod
   u64 mul(u64 a, u64 b) const {
      a *= b;
      u64 x = ((__uint128_t)a * im) >> 64;
      a -= x * m;
      if((ll)a < 0) a += m;
      return a;
   }
};
```

modint.hpp md5: 81b530

```
const ll mod = 998244353;
struct mm {
   ll x;
   mm(ll x_ = 0) : x(x_ % mod) {
      if(x < 0) x += mod;
   }
   friend mm operator+(mm a, mm b) { return a.x + b.x; }
   friend mm operator-(mm a, mm b) { return a.x - b.x; }
   friend mm operator*(mm a, mm b) { return a.x * b.x; }
   friend mm operator/(mm a, mm b) { return a * b.inv(); }
   // 4 行コピペ  Alt + Shift + クリックで複数カーソル
   friend mm& operator+=(mm& a, mm b) { return a = a.x + b.x; }
   friend mm& operator-=(mm& a, mm b) { return a = a.x - b.x; }
   friend mm& operator*=(mm& a, mm b) { return a = a.x * b.x; }
   friend mm& operator/=(mm& a, mm b) { return a = a * b.inv();
}
   mm inv() const { return pow(mod - 2); }
   mm pow(ll b) const {
      mm a = *this, c = 1;
      while(b) {
         if(b & 1) c *= a;
         a *= a;
         b >>= 1;
      }
      return c;
   }
};
```

FPS

FFT.hpp md5: 3138c7

```
// {998244353, 3}, {1811939329, 13}, {2013265921, 31}
mm g = 3;  // 原始根
void fft(vector<mm>& a) {
   ll n = sz(a), lg = __lg(n);
   assert((1 << lg) == n);
   vector<mm> b(n);
   rep(l, 1, lg + 1) {
      ll w = n >> l;
      mm s = 1, r = g.pow(mod >> l);
      for(ll u = 0; u < n / 2; u += w) {
         rep(d, 0, w) {
            mm x = a[u << 1 | d], y = a[u << 1 | w | d] * s;
            b[u | d] = x + y;
            b[n >> 1 | u | d] = x - y;
         }
         s *= r;
      }
      swap(a, b);
   }
}
vector<mm> conv(vector<mm> a, vector<mm> b) {
   if(a.empty() || b.empty()) return {};
   size_t s = sz(a) + sz(b) - 1, n = bit_ceil(s);
   // if(min(sz(a), sz(b)) <= 60) 愚直に掛け算
   a.resize(n);
   b.resize(n);
   fft(a);
   fft(b);
   mm inv = mm(n).inv();
   rep(i, 0, n) a[i] *= b[i] * inv;
   reverse(1 + all(a));
   fft(a);
   a.resize(s);
   return a;
}
```

FFT_fast.hpp md5: c8c567

```
// modint を u32 にして加減算を真面目にやると速い
mm g = 3;  // 原始根
void fft(vector<mm>& a) {
   ll n = sz(a), lg = __lg(n);
   static auto z = [] {
      vector<mm> z(30);
      mm s = 1;
      rep(i, 2, 32) {
         z[i - 2] = s * g.pow(mod >> i);
         s *= g.inv().pow(mod >> i);
      }
      return z;
   }();
   rep(l, 0, lg) {
      ll w = 1 << (lg - l - 1);
      mm s = 1;
      rep(k, 0, 1 << l) {
         ll o = k << (lg - l);
         rep(i, o, o + w) {
            mm x = a[i], y = a[i + w] * s;
            a[i] = x + y;
            a[i + w] = x - y;
         }
         s *= z[countr_zero<uint64_t>(~k)];
      }
   }
}
// コピペ
void ifft(vector<mm>& a) {
   ll n = sz(a), lg = __lg(n);
   static auto z = [] {
      vector<mm> z(30);
      mm s = 1;
      rep(i, 2, 32) {  // g を逆数に
         z[i - 2] = s * g.inv().pow(mod >> i);
         s *= g.pow(mod >> i);
      }
      return z;
   }();
   for(ll l = lg; l--;) {  // 逆順に
      ll w = 1 << (lg - l - 1);
      mm s = 1;
      rep(k, 0, 1 << l) {
         ll o = k << (lg - l);
         rep(i, o, o + w) {
            mm x = a[i], y = a[i + w];  // *s を下に移動
            a[i] = x + y;
            a[i + w] = (x - y) * s;
         }
         s *= z[countr_zero<uint64_t>(~k)];
      }
   }
}
vector<mm> conv(vector<mm> a, vector<mm> b) {
   if(a.empty() || b.empty()) return {};
   size_t s = sz(a) + sz(b) - 1, n = bit_ceil(s);
   // if(min(sz(a), sz(b)) <= 60) 愚直に掛け算
   a.resize(n);
   b.resize(n);
   fft(a);
   fft(b);
   mm inv = mm(n).inv();
   rep(i, 0, n) a[i] *= b[i] * inv;
   ifft(a);
   a.resize(s);
   return a;
}
```

graph

graph/tree

flow

燃やす埋める.md

変形前の制約	変形後の制約
x が 0 のとき z 失う	(x, T, z)
x が 0 のとき z 得る	無条件で z 得る; (S, x, z)
x が 1 のとき z 失う	(S, x, z)
x が 1 のとき z 得る	無条件で z 得る; (x, T, z)
$x, y, \ldots$ がすべて 0 のとき z 得る	無条件で z 得る; $(S, w, z), (w, x, \infty), (w, y, \infty)$
$x, y, \ldots$ がすべて 1 のとき z 得る	無条件で z 得る; $(w, T, z), (x, w, \infty), (y, w, \infty)$

string

KMP.hpp

md5: 886c63

```
// kmp[i] := max{ l ≤ i | s[:l] == s[(i+1)-l:i+1] }
// abacaba -> 0010123
auto KMP(string s) {
   vector<ll> p(sz(s));
   rep(i, 1, sz(s)) {
      ll g = p[i - 1];
      while(g && s[i] != s[g]) g = p[g - 1];
      p[i] = g + (s[i] == s[g]);
   }
   return p;
}
```

Manacher.hpp

md5: 5882fb

```
// 各位置での回文半径を求める
// aaabaaa -> 1214121
// 偶数長の回文を含めて直径を知るには, N+1 個の $ を挿入して 1 を引く
// $a$a$a$b$a$a$a$ -> 123432181234321
auto manacher(string s) {
   ll n = sz(s), i = 0, j = 0;
   vector<ll> r(n);
   while(i < n) {
      while(i >= j && i + j < n && s[i - j] == s[i + j]) j++;
      r[i] = j;
      ll k = 1;
      while(i >= k && i + k < n && k + r[i - k] < j) {
         r[i + k] = r[i - k];
         k++;
      }
      i += k, j -= k;
   }
   return r;
}
```

RollingHash.hpp

md5: adb8d3

```
// using u64 = uint64_t;
const u64 mod = INF;
u64 add(u64 a, u64 b) {
   a += b;
   if(a >= mod) a -= mod;
   return a;
}
u64 mul(u64 a, u64 b) {
   auto c = (__uint128_t)a * b;
   return add(c >> 61, c & mod);
}
random_device rnd;
const u64 r = ((u64)rnd() << 32 | rnd()) % mod;
struct RH {
   ll n;
   vector<u64> hs, pw;
   RH(string s) : n(sz(s)), hs(n + 1), pw(n + 1, 1) {
      rep(i, 0, n) {
         pw[i + 1] = mul(pw[i], r);
         hs[i + 1] = add(mul(hs[i], r), s[i]);
      }
   }
   u64 get(ll l, ll r) const { return add(hs[r], mod -
mul(hs[l], pw[r - l])); }
};
```

SuffixArray.hpp

md5: 1d70ce

```
// returns pair{sa, lcp}
// sa 長さ n : s[sa[0]:] < s[sa[1]:] < … < s[sa[n-1]:]
// lcp 長さ n-1 : lcp[i] = LCP(s[sa[i]:], s[sa[i+1]:])
auto SA(string s) {
   ll n = sz(s) + 1, lim = 256;
   // assert(lim > ranges::max(s));
   vector<ll> sa(n), lcp(n), x(all(s) + 1), y(n), ws(max(n,
lim)), rk(n);
   iota(all(sa), 0);
   for(ll j = 0, p = 0; p < n; j = max(1LL, j * 2), lim = p) {
      p = j;
      iota(all(y), n - j);
      rep(i, 0, n) if(sa[i] >= j) y[p++] = sa[i] - j;
      fill(all(ws), 0);
      rep(i, 0, n) ws[x[i]]++;
      rep(i, 1, lim) ws[i] += ws[i - 1];
      for(ll i = n; i--;) sa[--ws[x[y[i]]]] = y[i];
      swap(x, y);
      p = 1;
      x[sa[0]] = 0;
      rep(i, 1, n) {
         ll a = sa[i - 1], b = sa[i];
         x[b] = (y[a] == y[b] && y[a + j] == y[b + j]) ? p - 1
: p++;
      }
   }
   rep(i, 1, n) rk[sa[i]] = i;
   for(ll i = 0, k = 0; i < n - 1; lcp[rk[i++]] = k) {
      if(k) k--;
      while(s[i + k] == s[sa[rk[i] - 1] + k]) k++;
   }
   sa.erase(begin(sa));
   lcp.erase(begin(lcp));
   return pair{sa, lcp};
}
```

Zalgorithm.hpp

md5: b20b04

```
// Z[i] := LCP(s, s[i:])
// abacaba -> 7010301
auto Z(string s) {
   ll n = sz(s), l = -1, r = -1;
   vector<ll> z(n, n);
   rep(i, 1, n) {
      ll& x = z[i] = i < r ? min(r - i, z[i - l]) : 0;
      while(i + x < n && s[i + x] == s[x]) x++;
      if(i + x > r) l = i, r = i + x;
   }
   return z;
}
```

algorithm

geometry

memo

Primes.md

素数の個数

n	10^2	10^3	10^4	10^5	10^6	10^7	10^8	10^9	10^{10}
$\pi(n)$	25	168	1229	9592	78498	664579	5.76e+6	5.08e+7	4.55e+8

高度合成数

$\leq n$	10^3	10^4	10^5	10^6	10^7	10^8	10^9
x	840	7560	83160	720720	8648640	73513440	735134400
$d^0(x)$	32	64	128	240	448	768	1344

$\leq n$	10^{10}	10^{11}	10^{12}	10^{13}	10^{14}	10^{15}	10^{16}	10^{17}	10^{18}
$d^0(x)$	2304	4032	6720	10752	17280	26880	41472	64512	103680

素数階乗

n	2	3	5	7	11	13	17	19	23	29
$n\#$	2	6	30	210	2310	30030	510510	9.70e+6	2.23e+8	6.47e+9

階乗

$4!$	$5!$	$6!$	$7!$	$8!$	$9!$	$10!$	$11!$	$12!$	$13!$
24	120	720	5040	40320	362880	3.63e+6	3.99e+7	4.79e+8	6.23e+9