Олимпиада СПбГУ
по информатике
2023/24 учебного года

24fin0231 – Костаков Александр Сергеевич

A	B	C	D	E	F	sum
30	100	100	100	40	49	419

Task A

```
#include <algorithm>
#include <array>
#include <bitset>
#include <cassert>
#include <cinttypes>
#include <cmath>
#include <cstddef>
#include <cstring>
#include <deque>
#include <iostream>
#include <map>
#include <numeric>
#include <set>
#include <stack>
#include <vector>


// #define __INTERACTIVE_TASK_
// #define __OPT_PRAGMAS__
#if (!defined(__LOCAL_CLANGD_CPP__)) && (!defined(__LOCAL_DEBUG__)) &&
    defined(__OPT_PRAGMAS__)
# pragma GCC optimize("O3")
# pragma GCC optimize("unroll-loops")
# pragma GCC target("avx2,avx,sse4,sse4.1,sse4.2,bmi,bmi2,lzcnt,popcnt")
#endif

#ifndef __LOCAL_ANY_CPP__
# define __local_run_(code)
# define __prod_run_(code) \
```

```
    do {                \
      code;      \
    } while (0);
# define __local_stmt_(stmt)
# define __prod_stmt_(stmt) stmt
#else
# define __local_run_(code) \
    do {                 \
      code;       \
    } while (0);
# define __prod_run_(code)
# define __local_stmt_(stmt) stmt
# define __prod_stmt_(stmt)
#endif


using namespace std;

using ll = long long;
using ull = unsigned long long;
using iii = __int128_t;
using uiii = __uint128_t;
using if32 = int_fast32_t;
using uf32 = uint_fast32_t;
#define int ll

using vll = vector<ll>;
using vull = vector<ull>;

#define citers(container) (container).begin(), (container).end()

template <typename T> static inline void cvout(const T & farg) { cout <<
    farg; }

template <typename T, typename... Args> static inline void cvout(const T
    & farg, const Args &... args)
{
  cout << farg;
  cvout(args...);
}

static inline void cvoutnl [[maybe_unused]] ()
{
#if defined(__INTERACTIVE_TASK_) || defined(__LOCAL_DEBUG__)
  cout << endl;
#else
  cout << '\n';
#endif
}

template <typename T, typename... Args> static inline void cvoutnl(const
    T & farg, const Args &... args)
{
```

```
  cout << farg;
  cvoutnl(args...);
}

template <typename T> static inline void cvin(T & farg) { cin >> farg; }

template <typename T, typename... Args> static inline void cvin(T & farg,
     Args &... args)
{
  cin >> farg;
  cvin(args...);
}

static inline void _io_speedup [[maybe_unused]] ()
{
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
}

static inline void _reopen_stdinout [[maybe_unused]] (const char * in,
    const char * out)
{
  freopen(in, "r", stdin);
  freopen(out, "w", stdout);
}


int get_gcde(int a, int b, int & x, int & y)
{
  if (a == 0)
  {
    x = 0;
    y = 1;
    return b;
  }
  int x1, y1;
  int g = get_gcde(b % a, a, x1, y1);
  x = y1 - (b / a) * x1;
  y = x1;
  return g;
}


bool solve_eq(int a, int b, int n, int & x, int & y, int & dx, int & dy)
{
  int g = get_gcde(a, b, x, y);
  if (n % g != 0) return false;
  int k = n / g;
  x *= k;
  y *= k;
  dx = b / g;
  dy = -(a / g);
```

```
  return true;
}


ll a, b, n;
ll cnt = 0;
void solve(ll left_n)
{
  if (left_n < 0) return;
  if (left_n == 0)
  {
    ++cnt;
    return;
  }
  solve(left_n - a);
  solve(left_n - b);
}


signed main()
{
  _io_speedup();
  // _reopen_stdinout("input.txt", "output.txt");

  cvin(a, b, n);
  ll res = 0;
  solve(n - a);
  res += cnt * a;
  cnt = 0;
  solve(n - b);
  res += cnt * b;
  cvoutnl(res);
  return 0;
}
```

Task B

```
#include <algorithm>
#include <array>
#include <bitset>
#include <cassert>
#include <cinttypes>
#include <cmath>
#include <cstddef>
#include <cstring>
#include <deque>
#include <iostream>
#include <map>
#include <numeric>
#include <set>
#include <stack>
#include <vector>
```

```
// #define __INTERACTIVE_TASK_
// #define __OPT_PRAGMAS__
#if (!defined(__LOCAL_CLANGD_CPP__)) && (!defined(__LOCAL_DEBUG__)) &&
    defined(__OPT_PRAGMAS__)
# pragma GCC optimize("O3")
# pragma GCC optimize("unroll-loops")
# pragma GCC target("avx2,avx,sse4,sse4.1,sse4.2,bmi,bmi2,lzcnt,popcnt")
#endif

#ifndef __LOCAL_ANY_CPP__
# define __local_run_(code)
# define __prod_run_(code) \
    do {              \
      code;     \
    } while (0);
# define __local_stmt_(stmt)
# define __prod_stmt_(stmt) stmt
#else
# define __local_run_(code) \
    do {               \
      code;      \
    } while (0);
# define __prod_run_(code)
# define __local_stmt_(stmt) stmt
# define __prod_stmt_(stmt)
#endif


using namespace std;

using ll = long long;
using ull = unsigned long long;
using iii = __int128_t;
using uiii = __uint128_t;
using if32 = int_fast32_t;
using uf32 = uint_fast32_t;
#define int ll

using vll = vector<ll>;
using vull = vector<ull>;

#define citers(container) (container).begin(), (container).end()

template <typename T> static inline void cvout(const T & farg) { cout <<
    farg; }

template <typename T, typename... Args> static inline void cvout(const T
    & farg, const Args &... args)
{
  cout << farg;
  cvout(args...);
}
```

```
static inline void cvoutnl [[maybe_unused]] ()
{
#if defined(__INTERACTIVE_TASK_) || defined(__LOCAL_DEBUG__)
  cout << endl;
#else
  cout << '\n';
#endif
}

template <typename T, typename... Args> static inline void cvoutnl(const
    T & farg, const Args &... args)
{
  cout << farg;
  cvoutnl(args...);
}

template <typename T> static inline void cvin(T & farg) { cin >> farg; }

template <typename T, typename... Args> static inline void cvin(T & farg,
     Args &... args)
{
  cin >> farg;
  cvin(args...);
}

static inline void _io_speedup [[maybe_unused]] ()
{
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
}

static inline void _reopen_stdinout [[maybe_unused]] (const char * in,
    const char * out)
{
  freopen(in, "r", stdin);
  freopen(out, "w", stdout);
}


struct Vec
{
  ll x, y, z;

  constexpr Vec() : x(0), y(0), z(0) {}
  constexpr Vec(ll x, ll y, ll z) : x(x), y(y), z(z) {}
  constexpr Vec operator-() const { return Vec(-x, -y, -z); }
  constexpr Vec operator+(const Vec & o) const { return Vec(x + o.x, y +
    o.y, z + o.z); }
  constexpr Vec operator-(const Vec & o) const { return *this + (-o); }
  constexpr Vec operator*(ll k) const { return Vec(x * k, y * k, z * k);
    }
```

```
constexpr Vec operator/(ll k) const { return Vec(x / k, y / k, z / k);
 }
constexpr bool operator%(ll k) const { return x % k == 0 && y % k == 0
 && z % k == 0; }
constexpr bool operator==(const Vec & o) const { return x == o.x && y
 == o.y && z == o.z; }

ll len()
{
  if (z == 0) return abs(x);
  if (y == 0) return abs(x);
  if (x == 0) return abs(y);

  ll res = numeric_limits<ll>::max();
  // try perp z
  if (z > 0)
  {
    if (x < 0)
    {
      Vec left = *this - Vec(-1, 0, 1) * z;
      if (left.x == -left.y && left.z == 0) res = min(res, z + abs(left
 .x));
    }
    else
    {
      Vec right = *this - Vec(0, -1, 1) * z;
      if (right.x == -right.y && right.z == 0)
        res = min(res, z + abs(right.x));
    }
  }
  else
  {
    if (x < 0)
    {
      Vec left = *this - Vec(0, 1, -1) * (-z);
      if (left.x == -left.y && left.z == 0) res = min(res, -z + abs(
 left.x));
    }
    else
    {
      Vec right = *this - Vec(1, 0, -1) * (-z);
      if (right.x == -right.y && right.z == 0)
        res = min(res, -z + abs(right.x));
    }
  }
  // try cross
  Vec cross = Vec(1, 0, -1) * x + Vec(0, 1, -1) * y;
  if (*this == cross) res = min(res, abs(x) + abs(y));

  return res;
}

constexpr Vec move_x(ll half_delta) const
```

```
  {
    return Vec(x + 2 * half_delta, y - half_delta, z - half_delta);
  }
  constexpr Vec move_y(ll half_delta) const
  {
    return Vec(x - half_delta, y + 2 * half_delta, z - half_delta);
  }
  constexpr Vec move_z(ll half_delta) const
  {
    return Vec(x - half_delta, y - half_delta, z + 2 * half_delta);
  }
};

istream & operator>>(istream & in, Vec & v)
{
  in >> v.x >> v.y >> v.z;
  return in;
}


signed main()
{
  _io_speedup();
  // _reopen_stdinout("input.txt", "output.txt");

  ll n;
  cvin(n);
  Vec a, b;
  cvin(a, b);
  ll res = (b - a).len();
  res = min(res, (b - a.move_x(n)).len());
  res = min(res, (b - a.move_x(-n)).len());
  res = min(res, (b - a.move_y(n)).len());
  res = min(res, (b - a.move_y(-n)).len());
  res = min(res, (b - a.move_z(n)).len());
  res = min(res, (b - a.move_z(-n)).len());
  cvoutnl(res);
  return 0;
}
```

Task C

```
#include <algorithm>
#include <array>
#include <bitset>
#include <cassert>
#include <cinttypes>
#include <cmath>
#include <cstddef>
#include <cstring>
#include <deque>
#include <iostream>
#include <map>
```

```
#include <numeric>
#include <set>
#include <stack>
#include <vector>


// #define __INTERACTIVE_TASK_
// #define __OPT_PRAGMAS__
#if (!defined(__LOCAL_CLANGD_CPP__)) && (!defined(__LOCAL_DEBUG__)) &&
    defined(__OPT_PRAGMAS__)
# pragma GCC optimize("O3")
# pragma GCC optimize("unroll-loops")
# pragma GCC target("avx2,avx,sse4,sse4.1,sse4.2,bmi,bmi2,lzcnt,popcnt")
#endif

#ifndef __LOCAL_ANY_CPP__
# define __local_run_(code)
# define __prod_run_(code) \
    do {              \
      code;     \
    } while (0);
# define __local_stmt_(stmt)
# define __prod_stmt_(stmt) stmt
#else
# define __local_run_(code) \
    do {               \
      code;      \
    } while (0);
# define __prod_run_(code)
# define __local_stmt_(stmt) stmt
# define __prod_stmt_(stmt)
#endif


using namespace std;

using ll = long long;
using ull = unsigned long long;
using iii = __int128_t;
using uiii = __uint128_t;
using if32 = int_fast32_t;
using uf32 = uint_fast32_t;
#define int ll

using vll = vector<ll>;
using vull = vector<ull>;

#define citers(container) (container).begin(), (container).end()

template <typename T> static inline void cvout(const T & farg) { cout <<
    farg; }

template <typename T, typename... Args> static inline void cvout(const T
```

```
    & farg, const Args &... args)
{
  cout << farg;
  cvout(args...);
}

static inline void cvoutnl [[maybe_unused]] ()
{
#if defined(__INTERACTIVE_TASK_) || defined(__LOCAL_DEBUG__)
  cout << endl;
#else
  cout << '\n';
#endif
}

template <typename T, typename... Args> static inline void cvoutnl(const
    T & farg, const Args &... args)
{
  cout << farg;
  cvoutnl(args...);
}

template <typename T> static inline void cvin(T & farg) { cin >> farg; }

template <typename T, typename... Args> static inline void cvin(T & farg,
     Args &... args)
{
  cin >> farg;
  cvin(args...);
}

static inline void _io_speedup [[maybe_unused]] ()
{
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
}

static inline void _reopen_stdinout [[maybe_unused]] (const char * in,
    const char * out)
{
  freopen(in, "r", stdin);
  freopen(out, "w", stdout);
}


template <typename T, T M> struct ModInt
{
      private:
  using MI = ModInt<T, M>;

      public:
  T val;
```

```
  constexpr ModInt() : val(0) {}
  constexpr ModInt(T val) : val(val % M) {}

  constexpr inline MI operator+(MI o) const { return val + o.val; }
  constexpr inline MI operator-(MI o) const { return val - o.val + M; }
  constexpr inline MI operator*(MI o) const
  {
    if (sizeof(T) == 8) return (((__int128_t)val) * ((__int128_t)o.val))
   % ((__int128_t)M);
    if (sizeof(T) == 4) return (((ll)val) * ((ll)o.val)) % ((ll)M);
    return val * o.val;
  }
  constexpr inline MI operator+=(MI o) { return (*this = *this + o); }
  constexpr inline MI operator-=(MI o) { return (*this = *this - o); }
  constexpr inline MI operator*=(MI o) { return (*this = *this * o); }

  constexpr inline bool operator==(MI o) { return val == o.val; }
  constexpr inline bool operator==(T o) { return val == o; }
};

using MI32 = ModInt<signed, ((signed)1e9) + 9>;
template <signed M> using MI32T = ModInt<signed, M>;
using MI64 = ModInt<ll, ((ll)1e18) + 3>;
template <ll M> using MI64T = ModInt<ll, M>;


inline ll ord(char c) { return c - '0'; }

template <typename T> struct RevHash
{
  vector<T> ph, kpow;
  ll n;
  T K;

  RevHash(string s, T K) : ph(s.size() + 1), kpow(s.size() + 1), n(s.size
   ()), K(K)
  {
    ph[0] = 0;
    kpow[0] = 1;
    for (size_t i = 0; i < s.size(); ++i)
    {
      ph[i + 1] = ph[i] * K + ord(s[i]);
      kpow[i + 1] = kpow[i] * K;
    }
  }

  inline T substrl(ll l, ll len) const { return ph[l + len] - (ph[l] *
   kpow[len]); }

  inline T substr(ll l, ll r) const { return substrl(l, r - l + 1); }

  inline T cycle_shift(ll begin_idx)
```

```
  {
    if (begin_idx == 0) return ph.back();
    T left_hash = substrl(begin_idx, n - begin_idx);
    T right_hash = substrl(0, begin_idx);
    return left_hash * kpow[begin_idx] + right_hash;
  }
};

using RH32 = RevHash<MI32>;
using RH64 = RevHash<MI64>;


int n;
string s;

using H1 = RevHash<ModInt<ll, ((ll)1e18) + 3>>;
using H2 = RevHash<ModInt<ll, 999999999999999989>>;
using H3 = RevHash<ModInt<ll, 999999999999999967>>;
using H4 = RevHash<ModInt<ll, 999999999999999877>>;
using H5 = RevHash<ModInt<ll, 999999999999999829>>;


void first()
{
  H1 h1(s, 31);
  H2 h2(s, 29);
  H3 h3(s, 23);
  H4 h4(s, 19);
  H5 h5(s, 17);
  cvoutnl(h1.ph.back().val,
    ' ',
    h2.ph.back().val,
    ' ',
    h3.ph.back().val,
    ' ',
    h4.ph.back().val,
    ' ',
    h5.ph.back().val);
}

void second()
{
  ll n1, n2, n3, n4, n5;
  cvin(n1, n2, n3, n4, n5);
  H1 h1(s, 31);
  H2 h2(s, 29);
  H3 h3(s, 23);
  H4 h4(s, 19);
  H5 h5(s, 17);
  for (ll i = 0; i < n; ++i)
  {
    if (h1.cycle_shift(i) == n1 && h2.cycle_shift(i) == n2 && h3.
   cycle_shift(i) == n3
```

```
        && h4.cycle_shift(i) == n4 && h5.cycle_shift(i) == n5)
    {
      cvoutnl("YES");
      return;
    }
  }
  cvoutnl("NO");
}


signed main()
{
  _io_speedup();
  // _reopen_stdinout("input.txt", "output.txt");

  string mode;
  cvin(mode);
  cvin(n, s);
  if (mode[0] == 'f')
    first();
  else
    second();
  return 0;
}
```

Task D

```
#include <algorithm>
#include <array>
#include <bitset>
#include <cassert>
#include <cinttypes>
#include <cmath>
#include <cstddef>
#include <cstring>
#include <deque>
#include <iostream>
#include <map>
#include <numeric>
#include <set>
#include <stack>
#include <vector>


#define __INTERACTIVE_TASK_
// #define __OPT_PRAGMAS__
#if (!defined(__LOCAL_CLANGD_CPP__)) && (!defined(__LOCAL_DEBUG__)) &&
    defined(__OPT_PRAGMAS__)
# pragma GCC optimize("O3")
# pragma GCC optimize("unroll-loops")
# pragma GCC target("avx2,avx,sse4,sse4.1,sse4.2,bmi,bmi2,lzcnt,popcnt")
#endif
```

```
#ifndef __LOCAL_ANY_CPP__
# define __local_run_(code)
# define __prod_run_(code) \
    do {              \
      code;     \
    } while (0);
# define __local_stmt_(stmt)
# define __prod_stmt_(stmt) stmt
#else
# define __local_run_(code) \
    do {               \
      code;      \
    } while (0);
# define __prod_run_(code)
# define __local_stmt_(stmt) stmt
# define __prod_stmt_(stmt)
#endif


using namespace std;

using ll = long long;
using ull = unsigned long long;
using iii = __int128_t;
using uiii = __uint128_t;
using if32 = int_fast32_t;
using uf32 = uint_fast32_t;
#define int ll

using vll = vector<ll>;
using vull = vector<ull>;

#define citers(container) (container).begin(), (container).end()

template <typename T> static inline void cvout(const T & farg) { cout <<
    farg; }

template <typename T, typename... Args> static inline void cvout(const T
    & farg, const Args &... args)
{
  cout << farg;
  cvout(args...);
}

static inline void cvoutnl [[maybe_unused]] ()
{
#if defined(__INTERACTIVE_TASK_) || defined(__LOCAL_DEBUG__)
  cout << endl;
#else
  cout << '\n';
#endif
}
```

```
template <typename T, typename... Args> static inline void cvoutnl(const
    T & farg, const Args &... args)
{
  cout << farg;
  cvoutnl(args...);
}

template <typename T> static inline void cvin(T & farg) { cin >> farg; }

template <typename T, typename... Args> static inline void cvin(T & farg,
     Args &... args)
{
  cin >> farg;
  cvin(args...);
}

static inline void _io_speedup [[maybe_unused]] ()
{
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
}

static inline void _reopen_stdinout [[maybe_unused]] (const char * in,
    const char * out)
{
  freopen(in, "r", stdin);
  freopen(out, "w", stdout);
}


signed main()
{
  _io_speedup();
  // _reopen_stdinout("input.txt", "output.txt");

  ll n;
  cvin(n);
  vector<ll> count_gt(n, 0);

  char sign;
#define read_ans()                                  \
  do {                                        \
    cvin(sign);                       \
    if (sign == '!') exit(0); \
  } while (0);

  // all nums (n, n + 1)
  for (ll i = 1; i < n; ++i)
  {
    cvoutnl("1 1");
    read_ans();
    if (sign == '<')
```

```
    ++(count_gt[i]);
  else // sign == '>'
    ++(count_gt[i - 1]);
}
cvoutnl("1 1");
read_ans();

// cmp odd to (n, n + 2/3/4/...)
for (ll k = 2; k < n; ++k)
{
  ll shifts_cnt = 0;
  for (; shifts_cnt < n - k;)
  {
    cvoutnl("2 ", k + 1);
    read_ans();
    if (sign == '<')
      ++(count_gt[shifts_cnt + k]);
    else // sign == '>'
      ++(count_gt[shifts_cnt]);
    cvoutnl("1 ", k);
    read_ans();
    shifts_cnt += 2;
  }
  for (; shifts_cnt < n; ++shifts_cnt)
  {
    cvoutnl("1 1");
    read_ans();
  }
}

cvoutnl("1 1");
read_ans();
// cmp even to (n, n + 2/3/4/...)
for (ll k = 2; k < n; ++k)
{
  ll shifts_cnt = 0;
  for (; shifts_cnt < n - k - 1;)
  {
    cvoutnl("2 ", k + 1);
    read_ans();
    if (sign == '<')
      ++(count_gt[shifts_cnt + k + 1]);
    else // sign == '>'
      ++(count_gt[shifts_cnt + 1]);
    cvoutnl("1 ", k);
    read_ans();
    shifts_cnt += 2;
  }
  for (; shifts_cnt < n; ++shifts_cnt)
  {
    cvoutnl("1 1");
    read_ans();
  }
```

```
  }

  ll cur_shift = 1;

#define add_shift()                                    \
  do {                                          \
    ++cur_shift;                          \
    if (cur_shift == n) cur_shift = 0; \
  } while (0);

#define get_shifted_idx(idx) ((idx - cur_shift + n * 2) % n)

  vector<ll> val2idx(n), idx2val(n);
  for (ll i = 0; i < n; ++i)
  {
    val2idx[n - count_gt[i] - 1] = i;
    idx2val[i] = n - count_gt[i] - 1;
  }
  for (ll i = 0; i < n; ++i)
  {
    ll cur_val = n - i - 1;
    if (val2idx[cur_val] == i) continue;
    cvoutnl(get_shifted_idx(i) + 1, ' ', get_shifted_idx(val2idx[cur_val
   ]) + 1);
    read_ans();
    add_shift();
    idx2val[val2idx[cur_val]] = idx2val[i];
    val2idx[idx2val[i]] = val2idx[cur_val];
    idx2val[i] = cur_val;
    val2idx[cur_val] = i;
  }

  while (true)
  {
    cvoutnl("1 1");
    read_ans();
  }
  return 0;
}
```

Task E

```
#include <algorithm>
#include <array>
#include <bitset>
#include <cassert>
#include <cinttypes>
#include <cmath>
#include <cstddef>
#include <cstring>
#include <deque>
#include <iostream>
#include <map>
```

```
#include <numeric>
#include <queue>
#include <set>
#include <stack>
#include <vector>


// #define __INTERACTIVE_TASK_
// #define __OPT_PRAGMAS__
#if (!defined(__LOCAL_CLANGD_CPP__)) && (!defined(__LOCAL_DEBUG__)) &&
    defined(__OPT_PRAGMAS__)
# pragma GCC optimize("O3")
# pragma GCC optimize("unroll-loops")
# pragma GCC target("avx2,avx,sse4,sse4.1,sse4.2,bmi,bmi2,lzcnt,popcnt")
#endif

#ifndef __LOCAL_ANY_CPP__
# define __local_run_(code)
# define __prod_run_(code) \
    do {              \
      code;     \
    } while (0);
# define __local_stmt_(stmt)
# define __prod_stmt_(stmt) stmt
#else
# define __local_run_(code) \
    do {               \
      code;      \
    } while (0);
# define __prod_run_(code)
# define __local_stmt_(stmt) stmt
# define __prod_stmt_(stmt)
#endif


using namespace std;

using ll = long long;
using ull = unsigned long long;
using iii = __int128_t;
using uiii = __uint128_t;
using if32 = int_fast32_t;
using uf32 = uint_fast32_t;
#define int ll

using vll = vector<ll>;
using vull = vector<ull>;

#define citers(container) (container).begin(), (container).end()

template <typename T> static inline void cvout(const T & farg) { cout <<
    farg; }
```

```
template <typename T, typename... Args> static inline void cvout(const T
    & farg, const Args &... args)
{
  cout << farg;
  cvout(args...);
}

static inline void cvoutnl [[maybe_unused]] ()
{
#if defined(__INTERACTIVE_TASK_) || defined(__LOCAL_DEBUG__)
  cout << endl;
#else
  cout << '\n';
#endif
}

template <typename T, typename... Args> static inline void cvoutnl(const
    T & farg, const Args &... args)
{
  cout << farg;
  cvoutnl(args...);
}

template <typename T> static inline void cvin(T & farg) { cin >> farg; }

template <typename T, typename... Args> static inline void cvin(T & farg,
     Args &... args)
{
  cin >> farg;
  cvin(args...);
}

static inline void _io_speedup [[maybe_unused]] ()
{
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
}

static inline void _reopen_stdinout [[maybe_unused]] (const char * in,
    const char * out)
{
  freopen(in, "r", stdin);
  freopen(out, "w", stdout);
}


ll n, q;
vector<vector<ll>> g;
vector<ll> sz;
vector<int> used;
vector<ll> state;
queue<ll> t;
```

```
ll cur_t = 0;
void visit_all()
{
  ll visited_cnt = 0;
  ll v = 0;
  while (v != 0 || visited_cnt != n)
  {
    if (t.empty())
    {
      cvoutnl();
      exit(0);
    }
    if (cur_t == t.front())
    {
      cvout(v + 1, ' ');
      t.pop();
    }
    if (!used[v]) ++visited_cnt;
    used[v] = true;
    ll nv = g[v][state[v]];
    ++(state[v]);
    if (state[v] >= sz[v]) state[v] -= sz[v];
    v = nv;
    ++cur_t;
  }
}

vector<ll> get_cycle()
{
  vector<ll> res = {0};
  ll v = g[0][state[0]];
  ll next_cycle_state = state[0];
  ++(state[0]);
  if (state[0] >= sz[0]) state[0] -= sz[0];
  while (v != 0 || state[0] != next_cycle_state)
  {
    res.push_back(v);
    ll nv = g[v][state[v]];
    ++(state[v]);
    if (state[v] >= sz[v]) state[v] -= sz[v];
    v = nv;
  }
  return res;
}


signed main()
{
  _io_speedup();
  // _reopen_stdinout("input.txt", "output.txt");
```

```
  cvin(n, q);
  g.resize(n);
  sz.resize(n);
  used.assign(n, false);
  state.assign(n, 0);
  for (ll i = 0; i < n; ++i)
  {
    cvin(sz[i]);
    g[i].resize(sz[i]);
    for (ll j = 0; j < sz[i]; ++j)
    {
      cvin(g[i][j]);
      --(g[i][j]);
    }
  }
  for (ll i = 0; i < q; ++i)
  {
    ll inp;
    cvin(inp);
    t.push(inp);
  }
  visit_all();
  vector<ll> cycle = get_cycle();
  while (!t.empty())
  {
    ll val = t.front() - cur_t;
    t.pop();
    cvout(cycle[val % cycle.size()] + 1, ' ');
  }
  cvoutnl();
  return 0;
}
```

Task F

```
#include <algorithm>
#include <array>
#include <bitset>
#include <cassert>
#include <cinttypes>
#include <cmath>
#include <cstddef>
#include <cstring>
#include <deque>
#include <iostream>
#include <map>
#include <numeric>
#include <set>
#include <stack>
#include <vector>


// #define __INTERACTIVE_TASK_
```

```
// #define __OPT_PRAGMAS__
#if (!defined(__LOCAL_CLANGD_CPP__)) && (!defined(__LOCAL_DEBUG__)) &&
    defined(__OPT_PRAGMAS__)
# pragma GCC optimize("O3")
# pragma GCC optimize("unroll-loops")
# pragma GCC target("avx2,avx,sse4,sse4.1,sse4.2,bmi,bmi2,lzcnt,popcnt")
#endif

#ifndef __LOCAL_ANY_CPP__
# define __local_run_(code)
# define __prod_run_(code) \
    do {              \
      code;     \
    } while (0);
# define __local_stmt_(stmt)
# define __prod_stmt_(stmt) stmt
#else
# define __local_run_(code) \
    do {               \
      code;      \
    } while (0);
# define __prod_run_(code)
# define __local_stmt_(stmt) stmt
# define __prod_stmt_(stmt)
#endif


using namespace std;

using ll = long long;
using ull = unsigned long long;
using iii = __int128_t;
using uiii = __uint128_t;
using if32 = int_fast32_t;
using uf32 = uint_fast32_t;
#define int ll

using vll = vector<ll>;
using vull = vector<ull>;

#define citers(container) (container).begin(), (container).end()

template <typename T> static inline void cvout(const T & farg) { cout <<
    farg; }

template <typename T, typename... Args> static inline void cvout(const T
    & farg, const Args &... args)
{
  cout << farg;
  cvout(args...);
}

static inline void cvoutnl [[maybe_unused]] ()
```

```
{
#if defined(__INTERACTIVE_TASK_) || defined(__LOCAL_DEBUG__)
  cout << endl;
#else
  cout << '\n';
#endif
}

template <typename T, typename... Args> static inline void cvoutnl(const
    T & farg, const Args &... args)
{
  cout << farg;
  cvoutnl(args...);
}

template <typename T> static inline void cvin(T & farg) { cin >> farg; }

template <typename T, typename... Args> static inline void cvin(T & farg,
     Args &... args)
{
  cin >> farg;
  cvin(args...);
}

static inline void _io_speedup [[maybe_unused]] ()
{
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
}

static inline void _reopen_stdinout [[maybe_unused]] (const char * in,
    const char * out)
{
  freopen(in, "r", stdin);
  freopen(out, "w", stdout);
}


signed main()
{
  _io_speedup();
  // _reopen_stdinout("input.txt", "output.txt");

  ll h, w;
  cvin(h, w);
  vector<vector<ll>> f(h, vector<ll>(w));
  for (ll j = 0; j < h; ++j)
  {
    for (ll i = 0; i < w; ++i)
    {
      char digit;
      cvin(digit);
```

```
      f[j][i] = digit - '0';
    }
  }
  ll res_h = 0, res_v = 0;
  for (ll j = 0; j < h; ++j)
  {
    for (ll i = 0; i < w; i += 2) res_h += f[j][i] * f[j][i + 1];
  }
  for (ll j = 0; j < h; j += 2)
  {
    for (ll i = 0; i < w; ++i) res_v += f[j][i] * f[j + 1][i];
  }
  cvoutnl(max(res_h, res_v));
  if (res_h > res_v)
  {
    for (ll j = 0; j < h; ++j)
    {
      for (ll i = 0; i < w; i += 2) cvout("LR");
      cvoutnl();
    }
  }
  else
  {
    for (ll j = 0; j < h; j += 2)
    {
      for (ll i = 0; i < w; ++i) cvout("T");
      cvoutnl();
      for (ll i = 0; i < w; ++i) cvout("B");
      cvoutnl();
    }
  }
  return 0;
}
```