Олимпиада СПбГУ по информатике 2022/23 учебного года

A	B	C	D	E	F	Sum
100	100	90	10	48	10	358

Task A ()

```cpp
#include <bits/stdc++.h>
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/tree_policy.hpp>

using namespace std;

#define ALL(a) (a).begin(), (a).end()
#define RALL(a) (a).rbegin(), (a).rend()
#if __cplusplus < 202002L
#define ssize(a) static_cast<ptrdiff_t>(size(a))
#endif

using i8 = int8_t;
using i16 = int16_t;
using i32 = int32_t;
using i64 = int64_t;
__extension__ using i128 = __int128_t;

using u8 = uint8_t;
using u16 = uint16_t;
using u32 = uint32_t;
using u64 = uint64_t;
__extension__ using u128 = __uint128_t;

using f32 = float;
using f64 = double;
using f128 = long double;

using uchar = unsigned char;

constexpr i32 kInf32 = 1 << 30;
constexpr i64 kInf64 = 1ll << 61;
constexpr u64 kMod1 = (1 << 30) - 35;
constexpr u64 kMod2 = 228228229;

template <typename K, typename V, typename Comparator = less<K>>
using ordered_map =
    __gnu_pbds::tree<K, V, Comparator, __gnu_pbds::rb_tree_tag,
                     __gnu_pbds::tree_order_statistics_node_update>;

template <typename T, typename Comparator = less<T>>
using ordered_set = ordered_map<T, __gnu_pbds::null_type, Comparator>;

template <typename K, typename V,
          typename Hash_Fn =
              typename __gnu_pbds::detail::default_hash_fn<K>::type>
using hash_map = __gnu_pbds::gp_hash_table<K, V, Hash_Fn>;

template <typename T, typename Hash_Fn =
                          typename __gnu_pbds::detail::default_hash_fn<T>::type>
using hash_set = hash_map<T, __gnu_pbds::null_type, Hash_Fn>;

template <size_t Order, typename T>
class tensor : public vector<tensor<Order - 1, T>> {
  static_assert(Order > 0, "Invalid order of tensor");
```

```
 public:
  template <typename... Args>
  tensor(size_t d, Args... args)
      : vector<tensor<Order - 1, T>>(d, tensor<Order - 1, T>(args...)) {}
  using vector<tensor<Order - 1, T>>::vector;
};

template <typename T>
class tensor<1, T> : public vector<T> {
  using vector<T>::vector;
};

template <typename T>
using matrix = tensor<2, T>;

istream& operator>>(istream& is, i128& n) {
  n = 0;
  i128 f = 1;
  char ch;
  is.get(ch);
  while (ch < '0' || ch > '9') {
    if (ch == '-') f *= -1;
    is.get(ch);
  }
  while (ch >= '0' && ch <= '9') {
    n = n * 10 + ch - '0';
    is.get(ch);
  }
  n *= f;
  return is;
}

ostream& operator<<(ostream& os, i128 n) {
  string digits;
  bool neg = false;
  digits.reserve(32);
  if (n < 0) {
    neg = true;
    n = -n;
  }
  do {
    digits.push_back(static_cast<char>(n % 10) + '0');
    n /= 10;
  } while (n > 0);
  reverse(digits.begin(), digits.end());
  if (neg) os.put('-');
  os << digits;
  return os;
}

template <typename T1, typename T2>
istream& operator>>(istream& is, pair<T1, T2>& p) {
  is >> p.first >> p.second;
  return is;
}

template <typename T1, typename T2>
ostream& operator<<(ostream& os, const pair<T1, T2>& p) {
  os << p.first << " " << p.second;
  return os;
}

template <typename... Ts>
istream& operator>>(istream& is, tuple<Ts...>& t) {
  apply([&is](auto&... ta) { ((is >> ta), ...); }, t);
  return is;
}

template <typename... Ts>
ostream& operator<<(ostream& os, const tuple<Ts...>& t) {
  apply([&os](const auto&... ta) { ((os << ta << " "), ...); }, t);
  return os;
}
```

```cpp
template <typename T>
istream& operator>>(istream& is, vector<T>& v) {
  for (T& o : v) {
    is >> o;
  }
  return is;
}

template <typename T>
ostream& operator<<(ostream& os, const vector<T>& v) {
  for (const T& o : v) {
    os << o << " ";
  }
  return os;
}

template <typename T>
void chmin(T& f, const T& s) {
  if (s < f) f = s;
}

template <typename T>
void chmax(T& f, const T& s) {
  if (s > f) f = s;
}

void Solve();

signed main() {
  // freopen("input.txt", "r", stdin);
  // freopen("output.txt", "w", stdout);
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
  Solve();
  return 0;
}

void Solve() {
  vector<i32> d(6);
  cin >> d;
  vector<i32> ord;
  for (i32 i = 0; i < 6; i++) {
    ord.insert(ord.begin() + (d[i] - 1), i);
  }
  vector<i32> pos(6);
  for (i32 i = 0; i < 6; i++) {
    pos[ord[i]] = i + 1;
  }
  cout << pos << "\n";
}
```

Task B ()

```cpp
#include <bits/stdc++.h>
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/tree_policy.hpp>

using namespace std;

#define ALL(a) (a).begin(), (a).end()
#define RALL(a) (a).rbegin(), (a).rend()
#if __cplusplus < 202002L
#define ssize(a) static_cast<ptrdiff_t>(size(a))
#endif

using i8 = int8_t;
using i16 = int16_t;
using i32 = int32_t;
using i64 = int64_t;
__extension__ using i128 = __int128_t;

using u8 = uint8_t;
using u16 = uint16_t;
using u32 = uint32_t;
using u64 = uint64_t;
__extension__ using u128 = __uint128_t;

using f32 = float;
using f64 = double;
using f128 = long double;

using uchar = unsigned char;

constexpr i32 kInf32 = 1 << 30;
constexpr i64 kInf64 = 1ll << 61;
constexpr u64 kMod1 = (1 << 30) - 35;
constexpr u64 kMod2 = 228228229;

template <typename K, typename V, typename Comparator = less<K>>
using ordered_map =
    __gnu_pbds::tree<K, V, Comparator, __gnu_pbds::rb_tree_tag,
                     __gnu_pbds::tree_order_statistics_node_update>;

template <typename T, typename Comparator = less<T>>
using ordered_set = ordered_map<T, __gnu_pbds::null_type, Comparator>;

template <typename K, typename V,
          typename Hash_Fn =
              typename __gnu_pbds::detail::default_hash_fn<K>::type>
using hash_map = __gnu_pbds::gp_hash_table<K, V, Hash_Fn>;

template <typename T, typename Hash_Fn =
                          typename __gnu_pbds::detail::default_hash_fn<T>::type>
using hash_set = hash_map<T, __gnu_pbds::null_type, Hash_Fn>;

template <size_t Order, typename T>
class tensor : public vector<tensor<Order - 1, T>> {
  static_assert(Order > 0, "Invalid order of tensor");

 public:
  template <typename... Args>
  tensor(size_t d, Args... args)
      : vector<tensor<Order - 1, T>>(d, tensor<Order - 1, T>(args...)) {}
  using vector<tensor<Order - 1, T>>::vector;
};

template <typename T>
class tensor<1, T> : public vector<T> {
  using vector<T>::vector;
};

template <typename T>
using matrix = tensor<2, T>;

istream& operator>>(istream& is, i128& n) {
```

```
  n = 0;
  i128 f = 1;
  char ch;
  is.get(ch);
  while (ch < '0' || ch > '9') {
    if (ch == '-') f *= -1;
    is.get(ch);
  }
  while (ch >= '0' && ch <= '9') {
    n = n * 10 + ch - '0';
    is.get(ch);
  }
  n *= f;
  return is;
}

ostream& operator<<(ostream& os, i128 n) {
  string digits;
  bool neg = false;
  digits.reserve(32);
  if (n < 0) {
    neg = true;
    n = -n;
  }
  do {
    digits.push_back(static_cast<char>(n % 10) + '0');
    n /= 10;
  } while (n > 0);
  reverse(digits.begin(), digits.end());
  if (neg) os.put('-');
  os << digits;
  return os;
}

template <typename T1, typename T2>
istream& operator>>(istream& is, pair<T1, T2>& p) {
  is >> p.first >> p.second;
  return is;
}

template <typename T1, typename T2>
ostream& operator<<(ostream& os, const pair<T1, T2>& p) {
  os << p.first << " " << p.second;
  return os;
}

template <typename... Ts>
istream& operator>>(istream& is, tuple<Ts...>& t) {
  apply([&is](auto&... ta) { ((is >> ta), ...); }, t);
  return is;
}

template <typename... Ts>
ostream& operator<<(ostream& os, const tuple<Ts...>& t) {
  apply([&os](const auto&... ta) { ((os << ta << " "), ...); }, t);
  return os;
}

template <typename T>
istream& operator>>(istream& is, vector<T>& v) {
  for (T& o : v) {
    is >> o;
  }
  return is;
}

template <typename T>
ostream& operator<<(ostream& os, const vector<T>& v) {
  for (const T& o : v) {
    os << o << " ";
  }
  return os;
}
```

```cpp
template <typename T>
void chmin(T& f, const T& s) {
  if (s < f) f = s;
}

template <typename T>
void chmax(T& f, const T& s) {
  if (s > f) f = s;
}

void Solve();

signed main() {
  // freopen("input.txt", "r", stdin);
  // freopen("output.txt", "w", stdout);
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
  Solve();
  return 0;
}

void Solve1() {
  i32 n;
  cin >> n;
  vector<i64> a(n);
  cin >> a;
  i64 s = accumulate(ALL(a), 0); // < 100000
  i64 num = 100000ll * s;
  cout << num << "\n";
}
void Solve2() {
  i32 n;
  cin >> n;
  vector<i64> a(n);
  cin >> a;
  i64 num = a[0] / 100000ll;
  for (i32 i = 0; i < n; i++) {
    num += a[i] % 100000ll;
  }
  cout << num << "\n";
}
void Solve() {
  string t;
  cin >> t;
  if (t == "first") {
    Solve1();
  } else {
    Solve2();
  }
}
```

Task C ()

```cpp
#include <bits/stdc++.h>

#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/tree_policy.hpp>

using namespace std;

#define ALL(a) (a).begin(), (a).end()
#define RALL(a) (a).rbegin(), (a).rend()
#if __cplusplus < 202002L
#define ssize(a) static_cast<ptrdiff_t>(size(a))
#endif

using i8 = int8_t;
using i16 = int16_t;
using i32 = int32_t;
using i64 = int64_t;
__extension__ using i128 = __int128_t;

using u8 = uint8_t;
using u16 = uint16_t;
using u32 = uint32_t;
using u64 = uint64_t;
__extension__ using u128 = __uint128_t;

using f32 = float;
using f64 = double;
using f128 = long double;

using uchar = unsigned char;

constexpr i32 kInf32 = 1 << 30;
constexpr i64 kInf64 = 1ll << 61;
constexpr u64 kMod1 = (1 << 30) - 35;
constexpr u64 kMod2 = 228228229;

template <typename K, typename V, typename Comparator = less<K>>
using ordered_map =
    __gnu_pbds::tree<K, V, Comparator, __gnu_pbds::rb_tree_tag,
                     __gnu_pbds::tree_order_statistics_node_update>;

template <typename T, typename Comparator = less<T>>
using ordered_set = ordered_map<T, __gnu_pbds::null_type, Comparator>;

template <typename K, typename V,
          typename Hash_Fn =
              typename __gnu_pbds::detail::default_hash_fn<K>::type>
using hash_map = __gnu_pbds::gp_hash_table<K, V, Hash_Fn>;

template <typename T, typename Hash_Fn =
                          typename __gnu_pbds::detail::default_hash_fn<T>::type>
using hash_set = hash_map<T, __gnu_pbds::null_type, Hash_Fn>;

template <size_t Order, typename T>
class tensor : public vector<tensor<Order - 1, T>> {
  static_assert(Order > 0, "Invalid order of tensor");

 public:
  template <typename... Args>
  tensor(size_t d, Args... args)
      : vector<tensor<Order - 1, T>>(d, tensor<Order - 1, T>(args...)) {}
  using vector<tensor<Order - 1, T>>::vector;
};

template <typename T>
class tensor<1, T> : public vector<T> {
  using vector<T>::vector;
};

template <typename T>
using matrix = tensor<2, T>;
```

```
istream& operator>>(istream& is, i128& n) {
  n = 0;
  i128 f = 1;
  char ch;
  is.get(ch);
  while (ch < '0' || ch > '9') {
    if (ch == '-') f *= -1;
    is.get(ch);
  }
  while (ch >= '0' && ch <= '9') {
    n = n * 10 + ch - '0';
    is.get(ch);
  }
  n *= f;
  return is;
}

ostream& operator<<(ostream& os, i128 n) {
  string digits;
  bool neg = false;
  digits.reserve(32);
  if (n < 0) {
    neg = true;
    n = -n;
  }
  do {
    digits.push_back(static_cast<char>(n % 10) + '0');
    n /= 10;
  } while (n > 0);
  reverse(digits.begin(), digits.end());
  if (neg) os.put('-');
  os << digits;
  return os;
}

template <typename T1, typename T2>
istream& operator>>(istream& is, pair<T1, T2>& p) {
  is >> p.first >> p.second;
  return is;
}

template <typename T1, typename T2>
ostream& operator<<(ostream& os, const pair<T1, T2>& p) {
  os << p.first << " " << p.second;
  return os;
}

template <typename... Ts>
istream& operator>>(istream& is, tuple<Ts...>& t) {
  apply([&is](auto&... ta) { ((is >> ta), ...); }, t);
  return is;
}

template <typename... Ts>
ostream& operator<<(ostream& os, const tuple<Ts...>& t) {
  apply([&os](const auto&... ta) { ((os << ta << " "), ...); }, t);
  return os;
}

template <typename T>
istream& operator>>(istream& is, vector<T>& v) {
  for (T& o : v) {
    is >> o;
  }
  return is;
}

template <typename T>
ostream& operator<<(ostream& os, const vector<T>& v) {
  for (const T& o : v) {
    os << o << " ";
  }
  return os;
}
```

```cpp
template <typename T>
void chmin(T& f, const T& s) {
  if (s < f) f = s;
}

template <typename T>
void chmax(T& f, const T& s) {
  if (s > f) f = s;
}

void Solve();

signed main() {
  // freopen("input.txt", "r", stdin);
  // freopen("output.txt", "w", stdout);
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
  Solve();
  return 0;
}

vector<pair<i32, i32>> ans;

void check_two(pair<i32,i32> a, pair<i32, i32> b) {
  if (a.first == b.first) {
    ans.emplace_back(a.first, abs(a.second - b.second));
  }
  if (a.second == b.second) {
    ans.emplace_back(abs(a.first - b.first), a.second);
  }
}

void tile_down(pair<i32,i32> a, pair<i32, i32> b, pair<i32, i32> c) {
  if (a.first < b.first + c.first) return;
  if (a.second < max(b.second, c.second)) return;
  if (b.second == a.second) {
    if (c.second == a.second) {
      ans.emplace_back(a.first - b.first - c.first, a.second);
    }
    if (b.first + c.first == a.first) {
      ans.emplace_back(a.first - b.first, a.second - c.second);
    }
  } else {
    // left
    if (b.first + c.first == a.first && b.second == c.second) {
      ans.emplace_back(a.first, a.second - b.second);
    }
    // right
    if (b.second + c.second > a.second) {
      ans.emplace_back(b.first, a.second - b.second);
    }
    if (b.first + c.first == a.first && b.second + c.second >= a.second) {
      ans.emplace_back(a.first - b.first, a.second - c.second);
      ans.emplace_back(a.first - c.first, a.second - b.second);
    }
    // center
    if (c.second == a.second) {
      ans.emplace_back(b.first, a.second - b.second);
      ans.emplace_back(a.first - b.first - c.first, a.second);
    }
  }
}

void Solve() {
  pair<i32, i32> a, b, c;
  cin >> a >> b >> c;
  ans.push_back(a);
  ans.push_back(b);
  ans.push_back(c);
  for (i32 i = 0; i < 2; ++i, swap(a.first, a.second)) {
    for (i32 j = 0; j < 2; ++j, swap(b.first, b.second)) {
      for (i32 k = 0; k < 2; ++k, swap(c.first, c.second)) {
```

```
        check_two(a, b);
        check_two(a, c);
        check_two(b, c);
        tile_down(a, b, c);
        tile_down(b, a, c);
        tile_down(a, c, b);
        tile_down(b, c, a);
        tile_down(c, b, a);
        tile_down(c, a, b);
      }
    }
  }
  for (auto& p : ans) {
    if (p.first > p.second) swap(p.first, p.second);
  }
  set<pair<i32, i32>> ans_set(ALL(ans));
  for (auto [i, j] : ans_set) {
    if (i <= 0) continue;
    cout << i << " " << j << "\n";
  }
}
```

Task D ()

```cpp
#include <bits/stdc++.h>

#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/tree_policy.hpp>

using namespace std;

#define ALL(a) (a).begin(), (a).end()
#define RALL(a) (a).rbegin(), (a).rend()
#if __cplusplus < 202002L
#define ssize(a) static_cast<ptrdiff_t>(size(a))
#endif

using i8 = int8_t;
using i16 = int16_t;
using i32 = int32_t;
using i64 = int64_t;
__extension__ using i128 = __int128_t;

using u8 = uint8_t;
using u16 = uint16_t;
using u32 = uint32_t;
using u64 = uint64_t;
__extension__ using u128 = __uint128_t;

using f32 = float;
using f64 = double;
using f128 = long double;

using uchar = unsigned char;

constexpr i32 kInf32 = 1 << 30;
constexpr i64 kInf64 = 1ll << 61;
constexpr u64 kMod1 = (1 << 30) - 35;
constexpr u64 kMod2 = 228228229;

template <typename K, typename V, typename Comparator = less<K>>
using ordered_map =
    __gnu_pbds::tree<K, V, Comparator, __gnu_pbds::rb_tree_tag,
                     __gnu_pbds::tree_order_statistics_node_update>;

template <typename T, typename Comparator = less<T>>
using ordered_set = ordered_map<T, __gnu_pbds::null_type, Comparator>;

template <typename K, typename V,
          typename Hash_Fn =
              typename __gnu_pbds::detail::default_hash_fn<K>::type>
using hash_map = __gnu_pbds::gp_hash_table<K, V, Hash_Fn>;

template <typename T, typename Hash_Fn =
                          typename __gnu_pbds::detail::default_hash_fn<T>::type>
using hash_set = hash_map<T, __gnu_pbds::null_type, Hash_Fn>;

template <size_t Order, typename T>
class tensor : public vector<tensor<Order - 1, T>> {
  static_assert(Order > 0, "Invalid order of tensor");

 public:
  template <typename... Args>
  tensor(size_t d, Args... args)
      : vector<tensor<Order - 1, T>>(d, tensor<Order - 1, T>(args...)) {}
  using vector<tensor<Order - 1, T>>::vector;
};

template <typename T>
class tensor<1, T> : public vector<T> {
  using vector<T>::vector;
};

template <typename T>
using matrix = tensor<2, T>;
```

```
istream& operator>>(istream& is, i128& n) {
  n = 0;
  i128 f = 1;
  char ch;
  is.get(ch);
  while (ch < '0' || ch > '9') {
    if (ch == '-') f *= -1;
    is.get(ch);
  }
  while (ch >= '0' && ch <= '9') {
    n = n * 10 + ch - '0';
    is.get(ch);
  }
  n *= f;
  return is;
}

ostream& operator<<(ostream& os, i128 n) {
  string digits;
  bool neg = false;
  digits.reserve(32);
  if (n < 0) {
    neg = true;
    n = -n;
  }
  do {
    digits.push_back(static_cast<char>(n % 10) + '0');
    n /= 10;
  } while (n > 0);
  reverse(digits.begin(), digits.end());
  if (neg) os.put('-');
  os << digits;
  return os;
}

template <typename T1, typename T2>
istream& operator>>(istream& is, pair<T1, T2>& p) {
  is >> p.first >> p.second;
  return is;
}

template <typename T1, typename T2>
ostream& operator<<(ostream& os, const pair<T1, T2>& p) {
  os << p.first << " " << p.second;
  return os;
}

template <typename... Ts>
istream& operator>>(istream& is, tuple<Ts...>& t) {
  apply([&is](auto&... ta) { ((is >> ta), ...); }, t);
  return is;
}

template <typename... Ts>
ostream& operator<<(ostream& os, const tuple<Ts...>& t) {
  apply([&os](const auto&... ta) { ((os << ta << " "), ...); }, t);
  return os;
}

template <typename T>
istream& operator>>(istream& is, vector<T>& v) {
  for (T& o : v) {
    is >> o;
  }
  return is;
}

template <typename T>
ostream& operator<<(ostream& os, const vector<T>& v) {
  for (const T& o : v) {
    os << o << " ";
  }
  return os;
}
```

```cpp
template <typename T>
void chmin(T& f, const T& s) {
  if (s < f) f = s;
}

template <typename T>
void chmax(T& f, const T& s) {
  if (s > f) f = s;
}

void Solve();

signed main() {
  // freopen("input.txt", "r", stdin);
  // freopen("output.txt", "w", stdout);
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
  Solve();
  return 0;
}

void Solve() {
  i32 n;
  cin >> n;
  vector<i32> a(n);
  cin >> a;
  vector<i32> init_a = a;
  if (n == 1) {
    cout << "1 " << a[0] << endl;
    i32 _;
    cin >> _ >> _;
    cout << "1 " << a[0] << endl;
    return;
  }
  vector<bool> used(n);
  i64 sum = accumulate(ALL(a), 0) + n;
  while (sum > 0) {
    i32 x = accumulate(ALL(a), 0, bit_xor<i32>());
    i32 moved = false;
    for (i32 i = 0; i < n && !moved; i++) {
      if (!used[i] && (init_a[i] ^ a[i]) == x) {
        sum = sum - a[i] + init_a[i] - 1;
        cout << i + 1 << " " << 0 << endl;
        used[i] = true;
        a[i] = init_a[i];
        moved = true;
      }
    }
    if (x == 0 && !moved) {
      for (i32 i = 0; i < n && !moved; i++) {
        if (!used[i]) {
          sum = sum - a[i] + init_a[i] - 1;
          cout << i + 1 << " " << 0 << endl;
          used[i] = true;
          a[i] = init_a[i];
          moved = true;
        }
      }
      if (!moved) {
        cout << "-1 -1" << endl;
        return;
      }
    }
    if (!moved) {
      auto it = max_element(ALL(a));
      *it -= x;
      cout << it - a.begin() + 1 << " " << x << endl;
      sum -= x;
    }
    i32 h, n;
    cin >> h >> n;
    if (h == -1) {
```

```
      return;
    }
    --h;
    if (n == 0) {
      sum = sum - a[h] + init_a[h] - 1;
      used[h] = true;
      a[h] = init_a[h];
    } else {
      sum -= n;
      a[h] -= n;
    }
  }
}
```

Task E ()

```
#include <bits/stdc++.h>
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/tree_policy.hpp>

using namespace std;

#define ALL(a) (a).begin(), (a).end()
#define RALL(a) (a).rbegin(), (a).rend()
#if __cplusplus < 202002L
#define ssize(a) static_cast<ptrdiff_t>(size(a))
#endif

using i8 = int8_t;
using i16 = int16_t;
using i32 = int32_t;
using i64 = int64_t;
__extension__ using i128 = __int128_t;

using u8 = uint8_t;
using u16 = uint16_t;
using u32 = uint32_t;
using u64 = uint64_t;
__extension__ using u128 = __uint128_t;

using f32 = float;
using f64 = double;
using f128 = long double;

using uchar = unsigned char;

constexpr i32 kInf32 = 1 << 30;
constexpr i64 kInf64 = 1ll << 61;
constexpr u64 kMod1 = (1 << 30) - 35;
constexpr u64 kMod2 = 228228229;

template <typename K, typename V, typename Comparator = less<K>>
using ordered_map =
    __gnu_pbds::tree<K, V, Comparator, __gnu_pbds::rb_tree_tag,
                     __gnu_pbds::tree_order_statistics_node_update>;

template <typename T, typename Comparator = less<T>>
using ordered_set = ordered_map<T, __gnu_pbds::null_type, Comparator>;

template <typename K, typename V,
          typename Hash_Fn =
              typename __gnu_pbds::detail::default_hash_fn<K>::type>
using hash_map = __gnu_pbds::gp_hash_table<K, V, Hash_Fn>;

template <typename T, typename Hash_Fn =
                          typename __gnu_pbds::detail::default_hash_fn<T>::type>
using hash_set = hash_map<T, __gnu_pbds::null_type, Hash_Fn>;

template <size_t Order, typename T>
class tensor : public vector<tensor<Order - 1, T>> {
  static_assert(Order > 0, "Invalid order of tensor");

 public:
  template <typename... Args>
  tensor(size_t d, Args... args)
      : vector<tensor<Order - 1, T>>(d, tensor<Order - 1, T>(args...)) {}
  using vector<tensor<Order - 1, T>>::vector;
};

template <typename T>
class tensor<1, T> : public vector<T> {
  using vector<T>::vector;
};

template <typename T>
using matrix = tensor<2, T>;

istream& operator>>(istream& is, i128& n) {
```

```
  n = 0;
  i128 f = 1;
  char ch;
  is.get(ch);
  while (ch < '0' || ch > '9') {
    if (ch == '-') f *= -1;
    is.get(ch);
  }
  while (ch >= '0' && ch <= '9') {
    n = n * 10 + ch - '0';
    is.get(ch);
  }
  n *= f;
  return is;
}

ostream& operator<<(ostream& os, i128 n) {
  string digits;
  bool neg = false;
  digits.reserve(32);
  if (n < 0) {
    neg = true;
    n = -n;
  }
  do {
    digits.push_back(static_cast<char>(n % 10) + '0');
    n /= 10;
  } while (n > 0);
  reverse(digits.begin(), digits.end());
  if (neg) os.put('-');
  os << digits;
  return os;
}

template <typename T1, typename T2>
istream& operator>>(istream& is, pair<T1, T2>& p) {
  is >> p.first >> p.second;
  return is;
}

template <typename T1, typename T2>
ostream& operator<<(ostream& os, const pair<T1, T2>& p) {
  os << p.first << " " << p.second;
  return os;
}

template <typename... Ts>
istream& operator>>(istream& is, tuple<Ts...>& t) {
  apply([&is](auto&... ta) { ((is >> ta), ...); }, t);
  return is;
}

template <typename... Ts>
ostream& operator<<(ostream& os, const tuple<Ts...>& t) {
  apply([&os](const auto&... ta) { ((os << ta << " "), ...); }, t);
  return os;
}

template <typename T>
istream& operator>>(istream& is, vector<T>& v) {
  for (T& o : v) {
    is >> o;
  }
  return is;
}

template <typename T>
ostream& operator<<(ostream& os, const vector<T>& v) {
  for (const T& o : v) {
    os << o << " ";
  }
  return os;
}
```

```cpp
template <typename T>
void chmin(T& f, const T& s) {
  if (s < f) f = s;
}

template <typename T>
void chmax(T& f, const T& s) {
  if (s > f) f = s;
}

void Solve();

signed main() {
  // freopen("input.txt", "r", stdin);
  // freopen("output.txt", "w", stdout);
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
  Solve();
  return 0;
}

void Solve1() {
  i32 n;
  cin >> n;
  i32 cur = 0;
  for (i32 i = 0; i < 10; i++) {
    for (i32 j = 0; j < cur; j++) {
      cout << '1';
    }
    for (i32 j = cur; j < 10; j++) {
      cout << '0';
    }
    cout << "\n";
    cur += (n >> i) & 1;
  }
}
void Solve2() {
  vector<string> m(10);
  cin >> m;
  vector<i32> cnt(10);
  for (i32 i = 0; i < 10; i++) {
    for (i32 j = 0; j < 10; j++) {
      cnt[i] += m[i][j] == '1';
    }
  }
  sort(ALL(cnt));
  i32 n = 0;
  for (i32 i = 1; i < 10; i++) {
    n += (1 << (i - 1)) * (cnt[i] != cnt[i - 1]);
  }
  cout << n << "\n";
}
void Solve() {
  i32 t;
  string type;
  cin >> t >> type;
  while (t--) {
    if (type == "transmit") {
      Solve1();
    } else {
      Solve2();
    }
  }
}
```

Task F ()

```cpp
#include <bits/stdc++.h>

#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/tree_policy.hpp>
#include <string>

using namespace std;

#define ALL(a) (a).begin(), (a).end()
#define RALL(a) (a).rbegin(), (a).rend()
#if __cplusplus < 202002L
#define ssize(a) static_cast<ptrdiff_t>(size(a))
#endif

using i8 = int8_t;
using i16 = int16_t;
using i32 = int32_t;
using i64 = int64_t;
__extension__ using i128 = __int128_t;

using u8 = uint8_t;
using u16 = uint16_t;
using u32 = uint32_t;
using u64 = uint64_t;
__extension__ using u128 = __uint128_t;

using f32 = float;
using f64 = double;
using f128 = long double;

using uchar = unsigned char;

constexpr i32 kInf32 = 1 << 30;
constexpr i64 kInf64 = 1ll << 61;
constexpr u64 kMod1 = (1 << 30) - 35;
constexpr u64 kMod2 = 228228229;

template <typename K, typename V, typename Comparator = less<K>>
using ordered_map =
    __gnu_pbds::tree<K, V, Comparator, __gnu_pbds::rb_tree_tag,
                     __gnu_pbds::tree_order_statistics_node_update>;

template <typename T, typename Comparator = less<T>>
using ordered_set = ordered_map<T, __gnu_pbds::null_type, Comparator>;

template <typename K, typename V,
          typename Hash_Fn =
              typename __gnu_pbds::detail::default_hash_fn<K>::type>
using hash_map = __gnu_pbds::gp_hash_table<K, V, Hash_Fn>;

template <typename T, typename Hash_Fn =
                          typename __gnu_pbds::detail::default_hash_fn<T>::type>
using hash_set = hash_map<T, __gnu_pbds::null_type, Hash_Fn>;

template <size_t Order, typename T>
class tensor : public vector<tensor<Order - 1, T>> {
  static_assert(Order > 0, "Invalid order of tensor");

 public:
  template <typename... Args>
  tensor(size_t d, Args... args)
      : vector<tensor<Order - 1, T>>(d, tensor<Order - 1, T>(args...)) {}
  using vector<tensor<Order - 1, T>>::vector;
};

template <typename T>
class tensor<1, T> : public vector<T> {
  using vector<T>::vector;
};

template <typename T>
using matrix = tensor<2, T>;
```

```
istream& operator>>(istream& is, i128& n) {
  n = 0;
  i128 f = 1;
  char ch;
  is.get(ch);
  while (ch < '0' || ch > '9') {
    if (ch == '-') f *= -1;
    is.get(ch);
  }
  while (ch >= '0' && ch <= '9') {
    n = n * 10 + ch - '0';
    is.get(ch);
  }
  n *= f;
  return is;
}

ostream& operator<<(ostream& os, i128 n) {
  string digits;
  bool neg = false;
  digits.reserve(32);
  if (n < 0) {
    neg = true;
    n = -n;
  }
  do {
    digits.push_back(static_cast<char>(n % 10) + '0');
    n /= 10;
  } while (n > 0);
  reverse(digits.begin(), digits.end());
  if (neg) os.put('-');
  os << digits;
  return os;
}

template <typename T1, typename T2>
istream& operator>>(istream& is, pair<T1, T2>& p) {
  is >> p.first >> p.second;
  return is;
}

template <typename T1, typename T2>
ostream& operator<<(ostream& os, const pair<T1, T2>& p) {
  os << p.first << " " << p.second;
  return os;
}

template <typename... Ts>
istream& operator>>(istream& is, tuple<Ts...>& t) {
  apply([&is](auto&... ta) { ((is >> ta), ...); }, t);
  return is;
}

template <typename... Ts>
ostream& operator<<(ostream& os, const tuple<Ts...>& t) {
  apply([&os](const auto&... ta) { ((os << ta << " "), ...); }, t);
  return os;
}

template <typename T>
istream& operator>>(istream& is, vector<T>& v) {
  for (T& o : v) {
    is >> o;
  }
  return is;
}

template <typename T>
ostream& operator<<(ostream& os, const vector<T>& v) {
  for (const T& o : v) {
    os << o << " ";
  }
  return os;
```

```
}

template <typename T>
void chmin(T& f, const T& s) {
  if (s < f) f = s;
}

template <typename T>
void chmax(T& f, const T& s) {
  if (s > f) f = s;
}

void Solve();

signed main() {
  // freopen("input.txt", "r", stdin);
  // freopen("output.txt", "w", stdout);
  ios_base::sync_with_stdio(false);
  cin.tie(nullptr);
  cout.tie(nullptr);
  Solve();
  return 0;
}

deque<pair<i32, i32>> parse(const string& s) {
  deque<pair<i32, i32>> res;
  bool in_braces = false, right_part = false;
  i32 left_num = 0, right_num = 0;
  for (i32 i = 0; i < ssize(s); i++) {
    if (in_braces) {
      if (!right_part) {
        if (s[i] == '|') {
          right_part = true;
        } else {
          if (s[i - 1] != '(') {
            abort();
          }
          left_num = s[i] - '0';
        }
      } else {
        if (s[i] == ')') {
          res.emplace_back(left_num, right_num);
          in_braces = right_part = false;
          left_num = right_num = 0;
        } else {
          right_num = right_num * 10 + s[i] - '0';
        }
      }
    } else {
      if (s[i] == '(') {
        in_braces = true;
      } else {
        res.emplace_back(s[i] - '0', 1);
      }
    }
  }
  reverse(ALL(res));
  return res;
}
void Solve() {
  string s1, s2;
  cin >> s1 >> s2;
  deque<pair<i32, i32>> num1 = parse(s1), num2 = parse(s2);
  string ans;
  i32 carry = 0;
  while (!num1.empty() && !num2.empty()) {
    deque<pair<i32, i32>> digits_carry;
    set<pair<i32, i32>> digits_carry_set;
    while (num1.front().second > 0 && num2.front().second > 0) {
      i32 d = num1.front().first + num2.front().first + carry;
      carry = max(0, d > 9 ? 1 : 0);
      d %= 10;
      if (digits_carry_set.count({d, carry})) {
        // found a repeating block
```

```
      while (digits_carry.front() != pair<i32, i32>{d, carry}) {
        // clean up to the repeating block
        ans.push_back(digits_carry.front().first + '0');
        digits_carry.pop_front();
      }
      // extrack the block;
      string block;
      for (auto p : digits_carry) {
        block.push_back(p.first + '0');
      }
      i32 block_len = ssize(block);
      i32 block_repeats =
          min(num1.front().second, num2.front().second) / block_len;
      string buf = "(" + block + "|" + to_string(block_repeats + 1) + ")";
      reverse(ALL(buf));
      ans.insert(ans.end(), ALL(buf));
      num1.front().second -= block_len * block_repeats;
      num2.front().second -= block_len * block_repeats;
      digits_carry.clear();
      digits_carry_set.clear();
    } else {
      digits_carry.emplace_back(d, carry);
      digits_carry_set.emplace(d, carry);
      --num1.front().second, --num2.front().second;
    }
  }
  while (!digits_carry.empty()) {
    ans.push_back(digits_carry.front().first + '0');
    digits_carry.pop_front();
  }
  if (num1.front().second <= 0) num1.pop_front();
  if (num2.front().second <= 0) num2.pop_front();
}
while (!num1.empty()) {
  if (carry) {
    if (num1.front().first == 9) {
      string buf = "(0|" + to_string(num1.front().second) + ")";
      reverse(ALL(buf));
      ans.insert(ans.end(), ALL(buf));
      num1.pop_front();
    } else {
      ans.push_back(num1.front().first + 1 + '0');
      carry = 0;
      --num1.front().second;
      if (num1.front().second == 0) num1.pop_front();
    }
  } else {
    string buf = "(" + to_string(num1.front().first) + "|" +
                 to_string(num1.front().second) + ")";
    reverse(ALL(buf));
    ans.insert(ans.end(), ALL(buf));
    num1.pop_front();
  }
}
while (!num2.empty()) {
  if (carry) {
    if (num2.front().first == 9) {
      string buf = "(0|" + to_string(num2.front().second) + ")";
      reverse(ALL(buf));
      ans.insert(ans.end(), ALL(buf));
      num2.pop_front();
    } else {
      ans.push_back(num2.front().first + 1 + '0');
      carry = 0;
      --num2.front().second;
      if (num2.front().second == 0) num2.pop_front();
    }
  } else {
    string buf = "(" + to_string(num2.front().first) + "|" +
                 to_string(num2.front().second) + ")";
    reverse(ALL(buf));
    ans.insert(ans.end(), ALL(buf));
    num2.pop_front();
  }
```

```
  }
  if (carry) {
    ans.push_back('1');
  }
  reverse(ALL(ans));
  cout << ans << "\n";
}
```