Олимпиада СПбГУ по информатике 2021/22 учебного года

A	B	C	D	E	F	Sum
100	100	100	100	58	8	466

Task A ()

```
//#define _CRT_SECURE_NO_WARNINGS
//@author: Kirkon

#include <iostream>
#include <vector>
#include <algorithm>
#include <fstream>
#include <string>
#include <set>
#include <map>
#include <queue>
#include <cmath>
#include <stack>
#include <chrono>
#include <cstdio>
#include <cstdlib>
#include <unordered_set>
#include <unordered_map>
#include <queue>
#include <ctime>
#include <cassert>
#include <complex>
#include <string>
#include <cstring>
#include <chrono>
#include <random>
#include <bitset>
// #include<ext/pb_ds/assoc_container.hpp>
// #include<ext/pb_ds/tree_policy.hpp>
// using namespace __gnu_pbds;

using namespace std;

typedef long long ll;
typedef unsigned long long ull;
typedef float ff;
typedef double lf;
typedef long double llf;

typedef complex<ull> cull;
typedef complex<ff> cff;
//typedef complex<lf> clf;
typedef complex<lf> cllf;

const double eps = 1e-6;
const int int_inf = 2 * 1e9;
const ll ll_inf = 8 * 1e18;
const llf PI = acosl(-1.0);

#define pb push_back
#define mp make_pair
#define all(x) (x).begin(),(x).end()
#define fi first
#define se second

#pragma GCC optimize("O3,unroll-loops")
```

```
#pragma GCC target("avx2,bmi,bmi2,lzcnt,popcnt")

template<typename T>
using pair2 = pair<T, T>;
using pii = pair<int, int>;
using pli = pair<ll, int>;
using pll = pair<ll, ll>;

istream& operator >> (istream& in, pii& i) {
    in >> i.first >> i.second;
    return in;
}

template<class T>
istream& operator >> (istream& in, vector<T>& a) {
    for (auto& i : a) {
        in >> i;
    }
    return in;
}

template<class T>
ostream& operator << (ostream& out, vector<T>& a) {
    for (auto i : a) {
        out << i << "␣";
    }
    out << '\n';
    return out;
}

template<class T>
ostream& operator << (ostream& out, pair2<T>& a) {
    out << a.fi << "␣" << a.se << '\n';
    return out;
}

mt19937_64 rng(chrono::steady_clock::now().time_since_epoch().count());
ll myRand(ll B) {
    return (ull)rng() % B;
}

void _print(ll t) { cerr << t; }
void _print(int t) { cerr << t; }
void _print(string t) { cerr << t; }
void _print(char t) { cerr << t; }
void _print(llf t) { cerr << t; }
void _print(double t) { cerr << t; }
void _print(ull t) { cerr << t; }

template <class T, class V> void _print(pair <T, V> p);
template <class T> void _print(vector <T> v);
template <class T> void _print(set <T> v);
template <class T, class V> void _print(map <T, V> v);
template <class T> void _print(multiset <T> v);
template <class T, class V> void _print(pair <T, V> p) { cerr << "{"; _print(p.ff); cerr << ",";
    _print(p.ss); cerr << "}"; }
template <class T> void _print(vector <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣
    "; } cerr << "]"; }
template <class T> void _print(set <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣";
    } cerr << "]"; }
template <class T> void _print(multiset <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr <<
    "␣"; } cerr << "]"; }
template <class T, class V> void _print(map <T, V> v) { cerr << "[␣"; for (auto i : v) { _print(i)
    ; cerr << "␣"; } cerr << "]"; }
// void _print(pbds v) {cerr << "[ "; for (auto i : v) {_print(i); cerr << " ";} cerr << "]";}

/*----------------------------------------------------------------------------------------------
--------------------------------------------*/
ll gcd(ll a, ll b) { if (b > a) { return gcd(b, a); } if (b == 0) { return a; }
return gcd(b, a % b); }
ll expo(ll a, ll b, ll mod) { ll res = 1; while (b > 0) { if (b & 1)res = (res * a
) % mod; a = (a * a) % mod; b = b >> 1; } return res; }
void extendgcd(ll a, ll b, ll* v) { if (b == 0) { v[0] = 1; v[1] = 0; v[2] = a;
return; } extendgcd(b, a % b, v); ll x = v[1]; v[1] = v[0] - v[1] * (a / b); v[0]
```

```
= x; return; } //pass an arry of size1 3
ll mminv(ll a, ll b) { ll arr[3]; extendgcd(a, b, arr); return arr[0]; } //for
non prime b
ll mminvprime(ll a, ll b) { return expo(a, b - 2, b); }
bool revsort(ll a, ll b) { return a > b; }
void swap(int& x, int& y) { int temp = x; x = y; y = temp; }
ll combination(ll n, ll r, ll m, ll* fact, ll* ifact) { ll val1 = fact[n]; ll val2
 = ifact[n - r]; ll val3 = ifact[r]; return (((val1 * val2) % m) * val3) % m; }
void google(int t) { cout << "Case␣#" << t << ":␣"; }
vector<int> sieve(int n) { int* arr = new int[n + 1](); vector<int> vect; for (int
 i = 2; i <= n; i++)if (arr[i] == 0) { vect.push_back(i); for (ll j = (ll(i) * ll(
i)); j <= n; j += i)arr[j] = 1; } return vect; }
ll mod_add(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a + b) % m) + m) %
m; }
ll mod_mul(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a * b) % m) + m) %
m; }
ll mod_sub(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a - b) % m) + m) %
m; }
ll mod_div(ll a, ll b, ll m) { a = a % m; b = b % m; return (mod_mul(a, mminvprime
(b, m), m) + m) % m; }  //only for prime m
ll phin(ll n) { ll number = n; if (n % 2 == 0) { number /= 2; while (n % 2 == 0) n
 /= 2; } for (ll i = 3; i <= sqrt(n); i += 2) { if (n % i == 0) { while (n % i ==
0)n /= i; number = (number / i * (i - 1)); } } if (n > 1)number = (number / n * (n
 - 1)); return number; } //O(sqrt(N))
/*------------------------------------------------------------------------------------
------------------------------------------*/


struct num {
    int a;
    int b;
    num() {
        a = 0;
        b = 0;
    }
    num(int _a, int _b) {
        a = _a;
        b = _b;
    }
    void normalyze() {
        while (a % 10 == 0) {
            b++;
            a /= 10;
        }
    }

    bool operator <(const num& other) {
        if (this->b == other.b) {
            return this->a < other.a;
        }
        return this->b < other.b;
    }
};


void solve() {
    int n;
    cin >> n;
    vector<num> A(n);
    for (int i = 0; i < n; ++i) {
        int a, b;
        cin >> a >> b;
        A[i] = num(a, b);
        A[i].normalyze();
    }
    sort(all(A));
    int cur = A[0].b;
    int l = 0;
    ll cur_a = 0;
    while (l < n && A[l].b == cur) {
        cur_a += A[l].a;
        ++l;
        if (l == n || A[l].b > cur) {
            if (l == n) {
                while (cur_a % 10 == 0) {
                    ++cur;
                    cur_a /= 10;
                }
            }
            else {
                while (cur_a % 10 == 0 && A[l].b > cur) {
                    ++cur;
                    cur_a /= 10;
                }
            }
        }
    }
    if (l == n) {
        while (cur_a % 10 == 0) {
            ++cur;
            cur_a /= 10;
        }
    }
    cout << cur << '\n';
    return;
}

signed main() {
    //freopen("input.txt", "r", stdin);
    //freopen("output.txt", "w", stdout);
    ios_base::sync_with_stdio(false);
    cin.tie(0);
    int tt = 1;
    //cin >> tt;
    while (tt--) {
        solve();
    }
    return 0;
}
```

Task B ()

```
//#define _CRT_SECURE_NO_WARNINGS
//@author: Kirkon

#include <iostream>
#include <vector>
#include <algorithm>
#include <fstream>
#include <string>
#include <set>
#include <map>
#include <queue>
#include <cmath>
#include <stack>
#include <chrono>
#include <cstdio>
#include <cstdlib>
#include <unordered_set>
#include <unordered_map>
#include <queue>
#include <ctime>
#include <cassert>
#include <complex>
#include <string>
#include <cstring>
#include <chrono>
#include <random>
#include <bitset>
// #include<ext/pb_ds/assoc_container.hpp>
// #include<ext/pb_ds/tree_policy.hpp>
// using namespace __gnu_pbds;

using namespace std;

typedef long long ll;
typedef unsigned long long ull;
typedef float ff;
typedef double lf;
typedef long double llf;

typedef complex<ull> cull;
typedef complex<ff> cff;
//typedef complex<lf> clf;
typedef complex<lf> cllf;

const double eps = 1e-6;
const int int_inf = 2 * 1e9;
const ll ll_inf = 8 * 1e18;
const llf PI = acosl(-1.0);

#define pb push_back
#define mp make_pair
#define all(x) (x).begin(),(x).end()
#define fi first
#define se second

template<typename T>
using pair2 = pair<T, T>;
using pii = pair<int, int>;
using pli = pair<ll, int>;
using pll = pair<ll, ll>;

istream& operator >> (istream& in, pii& i) {
    in >> i.first >> i.second;
    return in;
}

template<class T>
istream& operator >> (istream& in, vector<T>& a) {
    for (auto& i : a) {
        in >> i;
    }
    return in;
```

```
}

template<class T>
ostream& operator << (ostream& out, vector<T>& a) {
    for (auto i : a) {
        out << i << "␣";
    }
    out << '\n';
    return out;
}

template<class T>
ostream& operator << (ostream& out, pair2<T>& a) {
    out << a.fi << "␣" << a.se << '\n';
    return out;
}

mt19937_64 rng(chrono::steady_clock::now().time_since_epoch().count());
ll myRand(ll B) {
    return (ull)rng() % B;
}

void _print(ll t) { cerr << t; }
void _print(int t) { cerr << t; }
void _print(string t) { cerr << t; }
void _print(char t) { cerr << t; }
void _print(llf t) { cerr << t; }
void _print(double t) { cerr << t; }
void _print(ull t) { cerr << t; }

template <class T, class V> void _print(pair <T, V> p);
template <class T> void _print(vector <T> v);
template <class T> void _print(set <T> v);
template <class T, class V> void _print(map <T, V> v);
template <class T> void _print(multiset <T> v);
template <class T, class V> void _print(pair <T, V> p) { cerr << "{"; _print(p.ff); cerr << ",";
    _print(p.ss); cerr << "}"; }
template <class T> void _print(vector <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣
    "; } cerr << "]"; }
template <class T> void _print(set <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣";
    } cerr << "]"; }
template <class T> void _print(multiset <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr <<
    "␣"; } cerr << "]"; }
template <class T, class V> void _print(map <T, V> v) { cerr << "[␣"; for (auto i : v) { _print(i)
    ; cerr << "␣"; } cerr << "]"; }
// void _print(pbds v) {cerr << "[ "; for (auto i : v) {_print(i); cerr << " ";} cerr << "]";}

/*-------------------------------------------------------------------------------------
------------------------------------------*/
ll gcd(ll a, ll b) { if (b > a) { return gcd(b, a); } if (b == 0) { return a; }
return gcd(b, a % b); }
ll expo(ll a, ll b, ll mod) { ll res = 1; while (b > 0) { if (b & 1)res = (res * a
) % mod; a = (a * a) % mod; b = b >> 1; } return res; }
void extendgcd(ll a, ll b, ll* v) { if (b == 0) { v[0] = 1; v[1] = 0; v[2] = a;
return; } extendgcd(b, a % b, v); ll x = v[1]; v[1] = v[0] - v[1] * (a / b); v[0]
= x; return; } //pass an arry of size1 3
ll mminv(ll a, ll b) { ll arr[3]; extendgcd(a, b, arr); return arr[0]; } //for
non prime b
ll mminvprime(ll a, ll b) { return expo(a, b - 2, b); }
bool revsort(ll a, ll b) { return a > b; }
void swap(int& x, int& y) { int temp = x; x = y; y = temp; }
ll combination(ll n, ll r, ll m, ll* fact, ll* ifact) { ll val1 = fact[n]; ll val2
 = ifact[n - r]; ll val3 = ifact[r]; return (((val1 * val2) % m) * val3) % m; }
void google(int t) { cout << "Case␣#" << t << ":␣"; }
vector<int> sieve(int n) { int* arr = new int[n + 1](); vector<int> vect; for (int
 i = 2; i <= n; i++)if (arr[i] == 0) { vect.push_back(i); for (ll j = (ll(i) * ll(
i)); j <= n; j += i)arr[j] = 1; } return vect; }
ll mod_add(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a + b) % m) + m) %
m; }
ll mod_mul(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a * b) % m) + m) %
m; }
ll mod_sub(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a - b) % m) + m) %
m; }
ll mod_div(ll a, ll b, ll m) { a = a % m; b = b % m; return (mod_mul(a, mminvprime
(b, m), m) + m) % m; }  //only for prime m
ll phin(ll n) { ll number = n; if (n % 2 == 0) { number /= 2; while (n % 2 == 0) n
 /= 2; } for (ll i = 3; i <= sqrt(n); i += 2) { if (n % i == 0) { while (n % i ==
0)n /= i; number = (number / i * (i - 1)); } } if (n > 1)number = (number / n * (n
 - 1)); return number; } //O(sqrt(N))
/*------------------------------------------------------------------------------------
--------------------------------------------*/

int query() {
    string s;
    cin >> s;
    int ans = 0;
    if (s == "Burn" || s == "Fail") {
        assert(false);
    }
    for (auto c : s) {
        if (c == 'e') {
            ++ans;
        }
    }
    return (ans / 2);
}

vector<int> k;

void get_next_k(int cur_k) {
    int cnt = 0;
    while (cnt <= cur_k) {
        cnt += query();
        if (cnt <= cur_k) {
            cout << "Wait" << endl;
            cout.flush();
        }
    }
    cout << "Flip␣and␣wait" << endl;
    cout.flush();
    return;
}

void get_final_k(int cur_k) {
    int cnt = 0;
    while (cnt <= cur_k) {
        cnt += query();
        if (cnt <= cur_k) {
            cout << "Wait" << endl;
            cout.flush();
        }
    }
    cout << "Stop" << endl;
    cout.flush();
    return;
}

void solve() {
    int n;
    cin >> n;
    k.resize(n);
    cin >> k;
    int cur_k = n - 1;
    int last = 0;
    while (k[cur_k] == 0) {
        --cur_k;
    }
    while (k[last] == 0) {
        ++last;
    }
    cout << "Flip␣and␣wait" << endl;
    cout.flush();
    while (cur_k >= 0) {
        while (k[cur_k] > 0) {
            if (cur_k == last && k[cur_k] == 1) {
                get_final_k(cur_k);
                return;
            }
            get_next_k(cur_k);
            --k[cur_k];
        }
        --cur_k;
    }
    return;
}

signed main() {
    //freopen("input.txt", "r", stdin);
    //freopen("output.txt", "w", stdout);
    //ios_base::sync_with_stdio(false);
    //cin.tie(0);
    int tt = 1;
    //cin >> tt;
    while (tt--) {
        solve();
    }
    return 0;
}
```

Task C ()

```
//#define _CRT_SECURE_NO_WARNINGS
//@author: Kirkon

#include <iostream>
#include <vector>
#include <algorithm>
#include <fstream>
#include <string>
#include <set>
#include <map>
#include <queue>
#include <cmath>
#include <stack>
#include <chrono>
#include <cstdio>
#include <cstdlib>
#include <unordered_set>
#include <unordered_map>
#include <queue>
#include <ctime>
#include <cassert>
#include <complex>
#include <string>
#include <cstring>
#include <chrono>
#include <random>
#include <bitset>
// #include<ext/pb_ds/assoc_container.hpp>
// #include<ext/pb_ds/tree_policy.hpp>
// using namespace __gnu_pbds;

using namespace std;

typedef long long ll;
typedef unsigned long long ull;
typedef float ff;
typedef double lf;
typedef long double llf;

typedef complex<ull> cull;
typedef complex<ff> cff;
//typedef complex<lf> clf;
typedef complex<lf> cllf;

const double eps = 1e-6;
const int int_inf = 2 * 1e9;
const ll ll_inf = 8 * 1e18;
const llf PI = acosl(-1.0);

#define pb push_back
#define mp make_pair
#define all(x) (x).begin(),(x).end()
#define fi first
#define se second

template<typename T>
using pair2 = pair<T, T>;
using pii = pair<int, int>;
using pli = pair<ll, int>;
using pll = pair<ll, ll>;

istream& operator >> (istream& in, pii& i) {
    in >> i.first >> i.second;
    return in;
}

template<class T>
istream& operator >> (istream& in, vector<T>& a) {
    for (auto& i : a) {
        in >> i;
    }
    return in;
```

```
}

template<class T>
ostream& operator << (ostream& out, vector<T>& a) {
    for (auto i : a) {
        out << i << "␣";
    }
    out << '\n';
    return out;
}

template<class T>
ostream& operator << (ostream& out, pair2<T>& a) {
    out << a.fi << "␣" << a.se << '\n';
    return out;
}

mt19937_64 rng(chrono::steady_clock::now().time_since_epoch().count());
ll myRand(ll B) {
    return (ull)rng() % B;
}

void _print(ll t) { cerr << t; }
void _print(int t) { cerr << t; }
void _print(string t) { cerr << t; }
void _print(char t) { cerr << t; }
void _print(llf t) { cerr << t; }
void _print(double t) { cerr << t; }
void _print(ull t) { cerr << t; }

template <class T, class V> void _print(pair <T, V> p);
template <class T> void _print(vector <T> v);
template <class T> void _print(set <T> v);
template <class T, class V> void _print(map <T, V> v);
template <class T> void _print(multiset <T> v);
template <class T, class V> void _print(pair <T, V> p) { cerr << "{"; _print(p.ff); cerr << ",";
    _print(p.ss); cerr << "}"; }
template <class T> void _print(vector <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣
    "; } cerr << "]"; }
template <class T> void _print(set <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣";
    } cerr << "]"; }
template <class T> void _print(multiset <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr <<
    "␣"; } cerr << "]"; }
template <class T, class V> void _print(map <T, V> v) { cerr << "[␣"; for (auto i : v) { _print(i)
    ; cerr << "␣"; } cerr << "]"; }
// void _print(pbds v) {cerr << "[ "; for (auto i : v) {_print(i); cerr << " ";} cerr << "]";}

/*----------------------------------------------------------------------------------------
----------------------------------------------*/
ll gcd(ll a, ll b) { if (b > a) { return gcd(b, a); } if (b == 0) { return a; }
return gcd(b, a % b); }
ll expo(ll a, ll b, ll mod) { ll res = 1; while (b > 0) { if (b & 1)res = (res * a
) % mod; a = (a * a) % mod; b = b >> 1; } return res; }
void extendgcd(ll a, ll b, ll* v) { if (b == 0) { v[0] = 1; v[1] = 0; v[2] = a;
return; } extendgcd(b, a % b, v); ll x = v[1]; v[1] = v[0] - v[1] * (a / b); v[0]
= x; return; } //pass an arry of size1 3
ll mminv(ll a, ll b) { ll arr[3]; extendgcd(a, b, arr); return arr[0]; } //for
non prime b
ll mminvprime(ll a, ll b) { return expo(a, b - 2, b); }
bool revsort(ll a, ll b) { return a > b; }
void swap(int& x, int& y) { int temp = x; x = y; y = temp; }
ll combination(ll n, ll r, ll m, ll* fact, ll* ifact) { ll val1 = fact[n]; ll val2
 = ifact[n - r]; ll val3 = ifact[r]; return (((val1 * val2) % m) * val3) % m; }
void google(int t) { cout << "Case␣#" << t << ":␣"; }
vector<int> sieve(int n) { int* arr = new int[n + 1](); vector<int> vect; for (int
 i = 2; i <= n; i++)if (arr[i] == 0) { vect.push_back(i); for (ll j = (ll(i) * ll(
i)); j <= n; j += i)arr[j] = 1; } return vect; }
ll mod_add(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a + b) % m) + m) %
m; }
ll mod_mul(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a * b) % m) + m) %
m; }
ll mod_sub(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a - b) % m) + m) %
m; }
ll mod_div(ll a, ll b, ll m) { a = a % m; b = b % m; return (mod_mul(a, mminvprime
(b, m), m) + m) % m; }  //only for prime m
ll phin(ll n) { ll number = n; if (n % 2 == 0) { number /= 2; while (n % 2 == 0) n
 /= 2; } for (ll i = 3; i <= sqrt(n); i += 2) { if (n % i == 0) { while (n % i ==
0)n /= i; number = (number / i * (i - 1)); } } if (n > 1)number = (number / n * (n
 - 1)); return number; } //O(sqrt(N))
/*------------------------------------------------------------------------------------
--------------------------------------------*/
string code_odd(string s) {
    string ans;
    ans = "";
    ans += s[0];
    bool zero_flag = true;
    for (int i = 0; i < s.size(); ++i) {
        if (s[i] == '1') {
            zero_flag = false;
            break;
        }
    }
    if (zero_flag) {
        ans[0] = '?';
        int local_cnt = 1;
        for (int i = 1; i < s.size(); ++i) {
            if (local_cnt < s.size() / 2) {
                ans += '?';
                ++local_cnt;
            }
            else {
                ans += s[i];
            }

        }
        return ans;
    }
    int cnt0 = 0, cnt1 = 0;
    for (int i = 1; i < s.size(); ++i) {
        char c = s[i];
        if (c == '0') {
            ++cnt0;
        }
        else {
            ++cnt1;
        }
    }
    if (cnt1 >= cnt0) {
        for (int i = 1; i < s.size(); ++i) {
            char c = s[i];
            if (c == '1') {
                ans += '?';
                //cout << '?';
            }
            else {
                ans += c;
                //cout << c;
            }
        }
    }
    if (cnt1 < cnt0) {
        for (int i = 1; i < s.size(); ++i) {
            char c = s[i];
            if (c == '0') {
                ans += '?';
                //cout << '?';
            }
            else {
                ans += c;
                //cout << c;
            }
        }
    }
    bool is_all_ones = true;
    for (int i = 0; i < s.size(); ++i) {
        if (s[i] == '0') {
            is_all_ones = false;
            break;
        }
    }
    if (is_all_ones) {
        for (int i = 0; i < s.size(); ++i) {
            ans[i] = '?';
        }
    }
    return ans;
}

string decode_odd(string s) {
    string ans;
    ans = "";
    bool is_all_ones = true;
    for (int i = 1; i < s.size(); ++i) {
        if (s[i] != '?') {
            is_all_ones = false;
            break;
        }
    }
    if (is_all_ones) {
        if (s[0] == '?') {
            for (int i = 0; i < s.size(); ++i) {
                ans += '1';
            }
        }
        else {
            if (s[0] == '1') {
                ans += '1';
                for (int i = 1; i < s.size(); ++i) {
                    ans += '0';
                }
            }
            else {
                ans += '0';
                for (int i = 1; i < s.size(); ++i) {
                    ans += '1';
                }
            }

        }

        return ans;
    }
    if (s[0] == '?' && !is_all_ones) {
        for (int i = 0; i < s.size(); ++i) {
            ans += '0';
        }
        return ans;
    }
    ans += s[0];
    bool isone = false;
    for (int i = 1; i < s.size(); ++i) {
        char c = s[i];
        if (c == '1') {
            isone = true;
            break;
        }
    }
    for (int i = 1; i < s.size(); ++i) {
        char c = s[i];
        if (isone) {
            if (c == '?') {
                ans += '0';
            }
            else {
                ans += c;
            }
        }
        else {
            if (c == '?') {
                ans += '1';
            }
            else {
                ans += c;
            }
        }
    }
    return ans;
}

string solve(string s) {
    //int n;
    //cin >> n;
    //for (int i = 0; i < n; ++i) {

        //cin >> s;
        bool flag = false;
        for (char c : s) {
            if (c == '?') {
                flag = true;
                break;
            }
        }
        if (!flag) {
            string q = code_odd(s);
            return q;
            //cout << q << '\n';
        }
        else {
            string q = decode_odd(s);
            return q;
            //cout << q << '\n';
        }

    //}
    return "";
}

string gen(int n) {
    string ans = "";
    for (int i = 0; i < n; ++i) {
        int fl = myRand(1e9 + 7) % 2;
        if (fl) {
            ans += '1';
        }
        else {
            ans += '0';
        }
    }
    return ans;
}


bool check(string s1, string s2) {
    int cnt = 0;
    for (int i = 0; i < s1.size(); ++i) {
        if (s2[i] == '?') {
            ++cnt;
        }
        else {
            if (s1[i] != s2[i]) {
                return false;
            }
        }
    }
    if (s1.size() / 2 > cnt) {
        return false;
    }
    return true;
}

signed main() {
    //freopen("input.txt", "r", stdin);
    //freopen("output.txt", "w", stdout);
    //ios_base::sync_with_stdio(false);
    //cin.tie(0);
    int tt = 10000000;
    cin >> tt;
    while (tt--) {
        string s;
        cin >> s;
        cout << solve(s) << endl;
        /*string q = gen(3);
        //string coded_q = solve(q);

        if (!check(q, coded_q)) {
            cout << "WRONG CODING\n";
            cout << q << "\n" << coded_q << '\n';
            return 0;
        }
        string decoded_q = solve(coded_q);
        cout << q << "\n" << coded_q << '\n' << decoded_q << '\n';
        if (q == decoded_q) {
            cout << "TEST PASSED" << endl;
        }
        else {
            cout << "TEST FAILED" << endl;
            cout << q;
            return 0;
        }*/
    }
    return 0;
}
```

Task D ()

```
//#define _CRT_SECURE_NO_WARNINGS
//@author: Kirkon

#include <iostream>
#include <vector>
#include <algorithm>
#include <fstream>
#include <string>
#include <set>
#include <map>
#include <queue>
#include <cmath>
#include <stack>
#include <chrono>
#include <cstdio>
#include <cstdlib>
#include <unordered_set>
#include <unordered_map>
#include <queue>
#include <ctime>
#include <cassert>
#include <complex>
#include <string>
#include <cstring>
#include <chrono>
#include <random>
#include <bitset>
// #include<ext/pb_ds/assoc_container.hpp>
// #include<ext/pb_ds/tree_policy.hpp>
// using namespace __gnu_pbds;

using namespace std;

typedef long long ll;
typedef unsigned long long ull;
typedef float ff;
typedef double lf;
typedef long double llf;

typedef complex<ull> cull;
typedef complex<ff> cff;
//typedef complex<lf> clf;
typedef complex<lf> cllf;

const double eps = 1e-6;
const int int_inf = 2 * 1e9;
const ll ll_inf = 8 * 1e18;
const llf PI = acosl(-1.0);

#define pb push_back
#define mp make_pair
#define all(x) (x).begin(),(x).end()
#define fi first
#define se second
#pragma GCC optimize("O3,unroll-loops")
#pragma GCC target("avx2,bmi,bmi2,lzcnt,popcnt")
template<typename T>
using pair2 = pair<T, T>;
using pii = pair<int, int>;
using pli = pair<ll, int>;
using pll = pair<ll, ll>;

istream& operator >> (istream& in, pii& i) {
    in >> i.first >> i.second;
    return in;
}

template<class T>
istream& operator >> (istream& in, vector<T>& a) {
    for (auto& i : a) {
        in >> i;
    }
    return in;
}

template<class T>
ostream& operator << (ostream& out, vector<T>& a) {
    for (auto i : a) {
        out << i << "␣";
    }
    out << '\n';
    return out;
}

template<class T>
ostream& operator << (ostream& out, pair2<T>& a) {
    out << a.fi << "␣" << a.se << '\n';
    return out;
}

mt19937_64 rng(chrono::steady_clock::now().time_since_epoch().count());
ll myRand(ll B) {
    return (ull)rng() % B;
}

void _print(ll t) { cerr << t; }
void _print(int t) { cerr << t; }
void _print(string t) { cerr << t; }
void _print(char t) { cerr << t; }
void _print(llf t) { cerr << t; }
void _print(double t) { cerr << t; }
void _print(ull t) { cerr << t; }

template <class T, class V> void _print(pair <T, V> p);
template <class T> void _print(vector <T> v);
template <class T> void _print(set <T> v);
template <class T, class V> void _print(map <T, V> v);
template <class T> void _print(multiset <T> v);
template <class T, class V> void _print(pair <T, V> p) { cerr << "{"; _print(p.ff); cerr << ",";
    _print(p.ss); cerr << "}"; }
template <class T> void _print(vector <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣
    "; } cerr << "]"; }
template <class T> void _print(set <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣";
    } cerr << "]"; }
template <class T> void _print(multiset <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr <<
    "␣"; } cerr << "]"; }
template <class T, class V> void _print(map <T, V> v) { cerr << "[␣"; for (auto i : v) { _print(i)
    ; cerr << "␣"; } cerr << "]"; }
// void _print(pbds v) {cerr << "[ "; for (auto i : v) {_print(i); cerr << " ";} cerr << "]";}

/*------------------------------------------------------------------------------------------
---------------------------------------------*/
ll gcd(ll a, ll b) { if (b > a) { return gcd(b, a); } if (b == 0) { return a; }
return gcd(b, a % b); }
ll expo(ll a, ll b, ll mod) { ll res = 1; while (b > 0) { if (b & 1)res = (res * a
) % mod; a = (a * a) % mod; b = b >> 1; } return res; }
void extendgcd(ll a, ll b, ll* v) { if (b == 0) { v[0] = 1; v[1] = 0; v[2] = a;
return; } extendgcd(b, a % b, v); ll x = v[1]; v[1] = v[0] - v[1] * (a / b); v[0]
= x; return; } //pass an arry of size1 3
ll mminv(ll a, ll b) { ll arr[3]; extendgcd(a, b, arr); return arr[0]; } //for
non prime b
ll mminvprime(ll a, ll b) { return expo(a, b - 2, b); }
bool revsort(ll a, ll b) { return a > b; }
void swap(int& x, int& y) { int temp = x; x = y; y = temp; }
ll combination(ll n, ll r, ll m, ll* fact, ll* ifact) { ll val1 = fact[n]; ll val2
 = ifact[n - r]; ll val3 = ifact[r]; return (((val1 * val2) % m) * val3) % m; }
void google(int t) { cout << "Case␣#" << t << ":␣"; }
vector<int> sieve(int n) { int* arr = new int[n + 1](); vector<int> vect; for (int
 i = 2; i <= n; i++)if (arr[i] == 0) { vect.push_back(i); for (ll j = (ll(i) * ll(
i)); j <= n; j += i)arr[j] = 1; } return vect; }
ll mod_add(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a + b) % m) + m) %
m; }
ll mod_mul(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a * b) % m) + m) %
m; }
ll mod_sub(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a - b) % m) + m) %
m; }
```

```
ll mod_div(ll a, ll b, ll m) { a = a % m; b = b % m; return (mod_mul(a, mminvprime
(b, m), m) + m) % m; }  //only for prime m
ll phin(ll n) { ll number = n; if (n % 2 == 0) { number /= 2; while (n % 2 == 0) n
 /= 2; } for (ll i = 3; i <= sqrt(n); i += 2) { if (n % i == 0) { while (n % i ==
0)n /= i; number = (number / i * (i - 1)); } } if (n > 1)number = (number / n * (n
 - 1)); return number; } //O(sqrt(N))
/*--------------------------------------------------------------------------------------------
----------------------------------------------*/

ll get_inv(int x, int p) {
    return expo(x, p - 2, p);
}

const ll MOD = 998'244'353;
vector<ll> factors = {1, 77901787, 806512919, 80441569, 72099649, 66426087, 841469608, 60321207,
    486786603, 531771476, 631736360, 846431980, 870771944, 32088045, 795653704, 412814720,
    813919435, 620795544, 761477120, 947903655, 723187516 };

int solve(ll n) {
    int minus_ones = 0;
    if (n > MOD - 1) {
        minus_ones = n / MOD;
    }
    n %= MOD;
    ll ans = 1;
    ll cnt = n / 49912217;
    for (ll i = 1 + cnt * 49912217; i <= n; ++i) {
        ans = ans * i % MOD;
    }
    ans = ans * factors[cnt] % MOD;
    if (minus_ones % 2 == 1) {
        ans = ans * (MOD - 1) % MOD;
    }
    return ans;
}

int count_zeroes(ll n) {
    ll ans = 0;
    while (n >= MOD) {
        ans += n / MOD;
        n /= MOD;
    }
    return ans;
}



void precalc() {
    ll crit_val = 49912217;
    ll ans = 1;
    for (ll i = 1; i <= MOD; ++i) {
        ans = (ans * i) % MOD;
        if (i % crit_val == 0) {
            factors.push_back(ans);
        }
    }
}

signed main() {
    //freopen("input.txt", "r", stdin);
    //freopen("output.txt", "w", stdout);
    //ios_base::sync_with_stdio(false);
    //cin.tie(0);
    int tt = 1;
    //cin >> tt;
    while (tt--) {
        ll n;
        cin >> n;
        cout << count_zeroes(n + 1) << "␣" << solve(n + 1) << '\n';
    }
    return 0;
}
```

Task E ()

```cpp
//#define _CRT_SECURE_NO_WARNINGS
//@author: Kirkon

#include <iostream>
#include <vector>
#include <algorithm>
#include <fstream>
#include <string>
#include <set>
#include <map>
#include <queue>
#include <cmath>
#include <stack>
#include <chrono>
#include <cstdio>
#include <cstdlib>
#include <unordered_set>
#include <unordered_map>
#include <queue>
#include <ctime>
#include <cassert>
#include <complex>
#include <string>
#include <cstring>
#include <chrono>
#include <random>
#include <bitset>
// #include<ext/pb_ds/assoc_container.hpp>
// #include<ext/pb_ds/tree_policy.hpp>
// using namespace __gnu_pbds;

using namespace std;

typedef long long ll;
typedef unsigned long long ull;
typedef float ff;
typedef double lf;
typedef long double llf;

typedef complex<ull> cull;
typedef complex<ff> cff;
//typedef complex<lf> clf;
typedef complex<lf> cllf;

const double eps = 1e-6;
const int int_inf = 2 * 1e9;
const ll ll_inf = 8 * 1e18;
const llf PI = acosl(-1.0);

#define pb push_back
#define mp make_pair
#define all(x) (x).begin(),(x).end()
#define fi first
#define se second
#pragma GCC optimize("O3,unroll-loops")
#pragma GCC target("avx2,bmi,bmi2,lzcnt,popcnt")
template<typename T>
using pair2 = pair<T, T>;
using pii = pair<int, int>;
using pli = pair<ll, int>;
using pll = pair<ll, ll>;

istream& operator >> (istream& in, pii& i) {
    in >> i.first >> i.second;
    return in;
}

template<class T>
istream& operator >> (istream& in, vector<T>& a) {
    for (auto& i : a) {
        in >> i;
    }
    return in;
}

template<class T>
ostream& operator << (ostream& out, vector<T>& a) {
    for (auto i : a) {
        out << i << "␣";
    }
    out << '\n';
    return out;
}

template<class T>
ostream& operator << (ostream& out, pair2<T>& a) {
    out << a.fi << "␣" << a.se << '\n';
    return out;
}

mt19937_64 rng(chrono::steady_clock::now().time_since_epoch().count());
ll myRand(ll B) {
    return (ull)rng() % B;
}

void _print(ll t) { cerr << t; }
void _print(int t) { cerr << t; }
void _print(string t) { cerr << t; }
void _print(char t) { cerr << t; }
void _print(llf t) { cerr << t; }
void _print(double t) { cerr << t; }
void _print(ull t) { cerr << t; }

template <class T, class V> void _print(pair <T, V> p);
template <class T> void _print(vector <T> v);
template <class T> void _print(set <T> v);
template <class T, class V> void _print(map <T, V> v);
template <class T> void _print(multiset <T> v);
template <class T, class V> void _print(pair <T, V> p) { cerr << "{"; _print(p.ff); cerr << ",";
    _print(p.ss); cerr << "}"; }
template <class T> void _print(vector <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣
    "; } cerr << "]"; }
template <class T> void _print(set <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣";
    } cerr << "]"; }
template <class T> void _print(multiset <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr <<
    "␣"; } cerr << "]"; }
template <class T, class V> void _print(map <T, V> v) { cerr << "[␣"; for (auto i : v) { _print(i)
    ; cerr << "␣"; } cerr << "]"; }
// void _print(pbds v) {cerr << "[ "; for (auto i : v) {_print(i); cerr << " ";} cerr << "]";}

/*------------------------------------------------------------------------------------------------
----------------------------------------------*/
ll gcd(ll a, ll b) { if (b > a) { return gcd(b, a); } if (b == 0) { return a; }
return gcd(b, a % b); }
ll expo(ll a, ll b, ll mod) { ll res = 1; while (b > 0) { if (b & 1)res = (res * a
) % mod; a = (a * a) % mod; b = b >> 1; } return res; }
void extendgcd(ll a, ll b, ll* v) { if (b == 0) { v[0] = 1; v[1] = 0; v[2] = a;
return; } extendgcd(b, a % b, v); ll x = v[1]; v[1] = v[0] - v[1] * (a / b); v[0]
= x; return; } //pass an arry of size1 3
ll mminv(ll a, ll b) { ll arr[3]; extendgcd(a, b, arr); return arr[0]; } //for
non prime b
ll mminvprime(ll a, ll b) { return expo(a, b - 2, b); }
bool revsort(ll a, ll b) { return a > b; }
void swap(int& x, int& y) { int temp = x; x = y; y = temp; }
ll combination(ll n, ll r, ll m, ll* fact, ll* ifact) { ll val1 = fact[n]; ll val2
 = ifact[n - r]; ll val3 = ifact[r]; return (((val1 * val2) % m) * val3) % m; }
void google(int t) { cout << "Case␣#" << t << ":␣"; }
vector<int> sieve(int n) { int* arr = new int[n + 1](); vector<int> vect; for (int
 i = 2; i <= n; i++)if (arr[i] == 0) { vect.push_back(i); for (ll j = (ll(i) * ll(
i)); j <= n; j += i)arr[j] = 1; } return vect; }
ll mod_add(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a + b) % m) + m) %
m; }
ll mod_mul(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a * b) % m) + m) %
m; }
ll mod_sub(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a - b) % m) + m) %
m; }
```

```cpp
ll mod_div(ll a, ll b, ll m) { a = a % m; b = b % m; return (mod_mul(a, mminvprime(b, m), m) + m) % m; }  //only for prime m
ll phin(ll n) { ll number = n; if (n % 2 == 0) { number /= 2; while (n % 2 == 0) n /= 2; } for (ll i = 3; i <= sqrt(n); i += 2) { if (n % i == 0) { while (n % i == 0)n /= i; number = (number / i * (i - 1)); } } if (n > 1)number = (number / n * (n - 1)); return number; } //O(sqrt(N))
/*--------------------------------------------------------------------------------------------------------------------------*/

ll get_inv(int x, int p) {
    return expo(x, p - 2, p);
}

void solve() {
    int n;
    cin >> n;
    vector<int> v(n);
    cin >> v;
    int q;
    cin >> q;
    while (q--) {
        int a, b, d;
        int price = 0;
        cin >> a >> b >> d;
        for (int i = a; i <= b - 1; ++i) {
            price += (d + v[i] - 1) / v[i];
        }
        cout << price << '\n';
    }
}

signed main() {
    //freopen("input.txt", "r", stdin);
    //freopen("output.txt", "w", stdout);
    ios_base::sync_with_stdio(false);
    cin.tie(0);
    int tt = 1;
    //cin >> tt;
    while (tt--) {
        solve();
    }
    return 0;
}
```

Task F ()

```
//#define _CRT_SECURE_NO_WARNINGS
//@author: Kirkon

#include <iostream>
#include <vector>
#include <algorithm>
#include <fstream>
#include <string>
#include <set>
#include <map>
#include <queue>
#include <cmath>
#include <stack>
#include <chrono>
#include <cstdio>
#include <cstdlib>
#include <unordered_set>
#include <unordered_map>
#include <queue>
#include <ctime>
#include <cassert>
#include <complex>
#include <string>
#include <cstring>
#include <chrono>
#include <random>
#include <bitset>
// #include<ext/pb_ds/assoc_container.hpp>
// #include<ext/pb_ds/tree_policy.hpp>
// using namespace __gnu_pbds;

using namespace std;

typedef long long ll;
typedef unsigned long long ull;
typedef float ff;
typedef double lf;
typedef long double llf;

typedef complex<ull> cull;
typedef complex<ff> cff;
//typedef complex<lf> clf;
typedef complex<lf> cllf;

const double eps = 1e-6;
const int int_inf = 2 * 1e9;
const ll ll_inf = 8 * 1e18;
const llf PI = acosl(-1.0);

#define pb push_back
#define mp make_pair
#define all(x) (x).begin(),(x).end()
#define fi first
#define se second
//#pragma GCC optimize("O3,unroll-loops")
//#pragma GCC target("avx2,bmi,bmi2,lzcnt,popcnt")
template<typename T>
using pair2 = pair<T, T>;
using pii = pair<int, int>;
using pli = pair<ll, int>;
using pll = pair<ll, ll>;

istream& operator >> (istream& in, pii& i) {
    in >> i.first >> i.second;
    return in;
}

template<class T>
istream& operator >> (istream& in, vector<T>& a) {
    for (auto& i : a) {
        in >> i;
    }
```

```cpp
    return in;
}

template<class T>
ostream& operator << (ostream& out, vector<T>& a) {
    for (auto i : a) {
        out << i << "␣";
    }
    out << '\n';
    return out;
}

template<class T>
ostream& operator << (ostream& out, pair2<T>& a) {
    out << a.fi << "␣" << a.se << '\n';
    return out;
}

mt19937_64 rng(chrono::steady_clock::now().time_since_epoch().count());
ll myRand(ll B) {
    return (ull)rng() % B;
}

void _print(ll t) { cerr << t; }
void _print(int t) { cerr << t; }
void _print(string t) { cerr << t; }
void _print(char t) { cerr << t; }
void _print(llf t) { cerr << t; }
void _print(double t) { cerr << t; }
void _print(ull t) { cerr << t; }

template <class T, class V> void _print(pair <T, V> p);
template <class T> void _print(vector <T> v);
template <class T> void _print(set <T> v);
template <class T, class V> void _print(map <T, V> v);
template <class T> void _print(multiset <T> v);
template <class T, class V> void _print(pair <T, V> p) { cerr << "{"; _print(p.ff); cerr << ",";
    _print(p.ss); cerr << "}"; }
template <class T> void _print(vector <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣
    "; } cerr << "]"; }
template <class T> void _print(set <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr << "␣";
    } cerr << "]"; }
template <class T> void _print(multiset <T> v) { cerr << "[␣"; for (T i : v) { _print(i); cerr <<
    "␣"; } cerr << "]"; }
template <class T, class V> void _print(map <T, V> v) { cerr << "[␣"; for (auto i : v) { _print(i)
    ; cerr << "␣"; } cerr << "]"; }
// void _print(pbds v) {cerr << "[ "; for (auto i : v) {_print(i); cerr << " ";} cerr << "]";}

/*------------------------------------------------------------------------------------------
----------------------------------------------------------*/
ll gcd(ll a, ll b) { if (b > a) { return gcd(b, a); } if (b == 0) { return a; }
return gcd(b, a % b); }
ll expo(ll a, ll b, ll mod) { ll res = 1; while (b > 0) { if (b & 1)res = (res * a
) % mod; a = (a * a) % mod; b = b >> 1; } return res; }
void extendgcd(ll a, ll b, ll* v) { if (b == 0) { v[0] = 1; v[1] = 0; v[2] = a;
return; } extendgcd(b, a % b, v); ll x = v[1]; v[1] = v[0] - v[1] * (a / b); v[0]
= x; return; } //pass an arry of size1 3
ll mminv(ll a, ll b) { ll arr[3]; extendgcd(a, b, arr); return arr[0]; } //for
non prime b
ll mminvprime(ll a, ll b) { return expo(a, b - 2, b); }
bool revsort(ll a, ll b) { return a > b; }
void swap(int& x, int& y) { int temp = x; x = y; y = temp; }
ll combination(ll n, ll r, ll m, ll* fact, ll* ifact) { ll val1 = fact[n]; ll val2
 = ifact[n - r]; ll val3 = ifact[r]; return (((val1 * val2) % m) * val3) % m; }
void google(int t) { cout << "Case␣#" << t << ":␣"; }
vector<int> sieve(int n) { int* arr = new int[n + 1](); vector<int> vect; for (int
 i = 2; i <= n; i++)if (arr[i] == 0) { vect.push_back(i); for (ll j = (ll(i) * ll(
i)); j <= n; j += i)arr[j] = 1; } return vect; }
ll mod_add(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a + b) % m) + m) %
m; }
ll mod_mul(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a * b) % m) + m) %
m; }
ll mod_sub(ll a, ll b, ll m) { a = a % m; b = b % m; return (((a - b) % m) + m) %
m; }
```

```
ll mod_div(ll a, ll b, ll m) { a = a % m; b = b % m; return (mod_mul(a, mminvprime
(b, m), m) + m) % m; }  //only for prime m
ll phin(ll n) { ll number = n; if (n % 2 == 0) { number /= 2; while (n % 2 == 0) n
 /= 2; } for (ll i = 3; i <= sqrt(n); i += 2) { if (n % i == 0) { while (n % i ==
0)n /= i; number = (number / i * (i - 1)); } } if (n > 1)number = (number / n * (n
 - 1)); return number; } //O(sqrt(N))
/*------------------------------------------------------------------------------
----------------------------------------*/


vector<pair<int, int>> q;

bool check(int l, int r) {
    int curl = q[l].first;
    int curr = q[l].second;
    for (int i = l + 1; i < r; ++i) {
        curl += q[i].first;
        curr += q[i].second;
    }
    if (curl <= 0 && curr >= 0) {
        return true;
    }
    else {
        return false;
    }
}

void solve() {
    int n;
    cin >> n;
    q.resize(n);
    for (int i = 0; i < n; ++i) {
        cin >> q[i].first >> q[i].second;
    }
    int global_ans = 1;
    for (int bitmask = 0; bitmask <= (1 << (n + 1)) - 1; ++bitmask) {
        vector<int> pos(n + 1, 0);
        int ans = 0;
        for (int i = 0; i < n + 1; ++i) {
            pos[i] = (bitmask >> i) & 1;
            ans += pos[i];
        }
        int cur = 0, next = 0;
        while (cur < n + 1 && pos[cur] == 0) ++cur;
        next = cur + 1;
        while (next < n + 1 && pos[next] == 0) ++next;
        if (next >= n + 1) {
            continue;
        }
        bool flag = true;
        while (next < n + 1) {
            if (check(cur, next)) {
                cur = next;
                next++;
                while (next < n + 1 && pos[next] == 0) ++next;
            }
            else {
                flag = false;
                break;
            }
        }
        if (flag) {
            global_ans = max(global_ans, ans);
        }
    }
    cout << global_ans << endl;
}

signed main() {
    //freopen("input.txt", "r", stdin);
    //freopen("output.txt", "w", stdout);
    int tt = 1;
    //cin >> tt;
    while (tt--) {
        solve();
    }
    return 0;
}
```