Олимпиада СПбГУ по информатике 2021/22 учебного года

A	B	C	D	E	F	Sum
100	100	100	100	58	31	489

Task A ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <unordered_map>

using namespace std;

unordered_map<long long, long long> sumpow;

const long long INF = 1e18;

void upd(long long b) {
    if (sumpow[b] % 10 == 0) {
        long long d = 0;
        while (sumpow[b] % 10 == 0) {
            sumpow[b] /= 10;
            d++;
        }
        sumpow[b + d] += sumpow[b];
        sumpow.erase(b);
        upd(b + d);
    }
}

signed main() {
    int n; cin >> n;
    for (int i = 0; i < n; ++i) {
        long long a, b; cin >> a >> b;
        while (a % 10 == 0) {
            a /= 10;
            b++;
        }
        sumpow[b] += a;
        upd(b);
    }
    long long mn = INF;
    for (auto el: sumpow) {
        mn = min(mn, el.first);
    }
    cout << mn << endl;
}
```

Task B ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <unordered_map>

using namespace std;

signed main() {
    int n; cin >> n;
    vector<int> k(n);
    for (int i = 0; i < n; ++i) {
        cin >> k[i];
    }
    int lastp = -1;
    for (int i = n - 1; i >= 0; --i) {
        if (k[i]) {
            lastp = i;
            break;
        }
    }
    int already = 0;
    vector<int> groups;
    cout << "Flip and wait" << endl;
    while (already < lastp + 1) {
        string s; cin >> s;
        already += (s.size() - 3) / 2;
        groups.push_back((s.size() - 3) / 2);
        if (already >= lastp + 1) {
            break;
        } else {
            cout << "Wait" << endl;
        }
    }
    vector<int> sumtime;
    int ind = 0;
    for (auto el: groups) {
        int sum = 0;
        for (int i = 0; i < el; i++) {
            sum += k[ind];
            ind++;
        }
        sumtime.push_back(sum);
    }
    sumtime.back()--;
    /*for (auto el: sumtime) {
        cout << el << " ";
    }
    cout << endl;*/
    int firstp = -1;
    for (int i = 0; i < sumtime.size(); ++i) {
        if (sumtime[i] > 0) {
            firstp = i;
            break;
        }
    }
    if (firstp != -1) {
        cout << "Flip and wait" << endl;
        for (int i = sumtime.size() - 1; i >= 0; --i) {
            for (int j = 0; j < sumtime[i]; ++j) {
                for (int l = 0; l < i; ++l) {
                    string s; cin >> s;
                    cout << "Wait" << endl;
                }
                string s; cin >> s;
                if (i != firstp or j != sumtime[i] - 1) {
                    cout << "Flip and wait" << endl;
                }
            }
        }
    }
    cout << "Stop" << endl;
}
```

Task C ()

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <unordered_map>

using namespace std;

signed main() {
    int t; cin >> t;
    while (t--) {
        string s1; cin >> s1;
        vector<char> s;
        for (auto el: s1) {
            s.push_back(el);
        }
        bool repair = false;
        for (auto el: s) {
            if (el == '?') {
                repair = true;
                break;
            }
        }
        int n = s.size();
        if (repair) {
            for (int i = 0; i < n - n % 2; i += 2) {
                if (s[i] == '0') {
                    s[i + 1] = '0';
                } else if (s[i] == '1') {
                    s[i + 1] = '1';
                } else if (s[i + 1] == '0') {
                    s[i] = '1';
                } else if (s[i + 1] == '1') {
                    s[i] = '0';
                }
            }
            for (auto el: s) {
                cout << el;
            }
            cout << '\n';
        } else {
            for (int i = 0; i < n - n % 2; i += 2) {
                if (s[i] == '0' and s[i + 1] == '0') {
                    s[i + 1] = '?';
                } else if (s[i] == '0' and s[i + 1] == '1') {
                    s[i] = '?';
                } else if (s[i] == '1' and s[i + 1] == '0') {
                    s[i] = '?';
                } else if (s[i] == '1' and s[i + 1] == '1') {
                    s[i + 1] = '?';
                }
            }
            for (auto el: s) {
                cout << el;
            }
            cout << '\n';
        }
    }
}
```

Task D ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <unordered_map>

using namespace std;

const long long MOD = 998244353, step = 1e8;

long long factor(long long n) {
    n--;
    if (n == 0) return 1;
    if (n == MOD - 1 or n == MOD) return MOD - 1;
    vector<long long> pre = {1, 980201571, 468660588, 478389955, 137136996, 203730522, 533107751,
        496409701, 637333574, 465585441};
    long long ans = pre[n / step];
    for (long long i = n - n % step + 2; i <= n + 1; ++i) {
        ans *= i;
        ans %= MOD;
    }
    return ans;
}

long long fastpow(long long n, long long p) {
    if (p == 0) return 1;
    if (p % 2 == 0) {
        long long res = fastpow(n, p / 2);
        return (res * res) % MOD;
    } else {
        return (n * fastpow(n, p - 1)) % MOD;
    }
}

long long get_ans(long long n) {
    if (n <= MOD) return factor(n);
    long long d = n / MOD;
    return (((get_ans(d) * fastpow(MOD - 1, d)) % MOD) * factor(n % MOD)) % MOD;
}

signed main() {
    long long n; cin >> n;
    n++;
    cout << n / MOD + n / (MOD * MOD) << " " << get_ans(n) << endl;
    /*long long ans = 1;
    long long step = 1e8;
    for (long long i = 2; i <= MOD - 1; ++i) {
        ans *= i;
        ans %= MOD;
        if ((i - 1) % step == 0) {
            cout << ans << ", ";
        }
    }*/
    //cout << 0 << " " << ans << endl;
}
```

Task E ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <unordered_map>

using namespace std;

signed main() {
    int n; cin >> n;
    vector<int> v(n);
    for (int i = 0; i < n; ++i) {
        cin >> v[i];
    }
    int q; cin >> q;
    while (q--) {
        int a, b, d; cin >> a >> b >> d;
        long long ans = 0;
        for (int i = a; i < b; ++i) {
            ans += (long long)((d + v[i] - 1) / v[i]);
        }
        cout << ans << endl;
    }
}
```

Task F ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <unordered_map>

using namespace std;

vector<bool> used;
vector<int> ans;
vector<vector<int>> g;

void dfs(int v) {
    used[v] = true;
    ans[v] = 1;
    for (auto u: g[v]) {
        if (not used[u]) {
            dfs(u);
        }
        ans[v] = max(ans[v], 1 + ans[u]);
    }
}

signed main() {
    int n; cin >> n;
    long long suml = 0, sumr = 0;
    vector<long long> prefl(n + 1, 0), prefr(n + 1, 0);
    for (int i = 0; i < n; ++i) {
        long long l, r; cin >> l >> r;
        suml += l;
        sumr += r;
        prefl[i + 1] = suml;
        prefr[i + 1] = sumr;
    }
    g.resize(n + 1);
    for (int i = 0; i <= n; ++i) {
        for (int j = i + 1; j <= n; ++j) {
            if (prefl[j] - prefl[i] <= 0 and prefr[j] - prefr[i] >= 0) {
                g[i].push_back(j);
            }
        }
    }
    used.assign(n + 1, false);
    ans.resize(n + 1);
    int res = 0;
    for (int i = 0; i <= n; ++i) {
        if (not used[i]) {
            dfs(i);
            res = max(res, ans[i]);
        }
    }
    cout << res << endl;
}
```