Олимпиада СПбГУ по информатике 2020/21 учебного года

A	B	C	D	E	F	Sum
100	100	100	100	55	25	480

Task A ()

```
def solve(n):
    n = (n - 1) % 9
    ans = 9
    if n < 7:
        ans = n + 1
    elif n == 7:
        ans = 8
    return ans

n = int(input())
if n >= 10 and (n - 10) % 9 == 0:
    print(0)
else:
    print(solve(n))

# x = 11111111111111111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111111111111111
1111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111**2
# x = str(x)
# # print(x)

# for n in range(1, 301):
#     cur = (n - 1) % 9
#     ans = solve(n)
#     if ans != ord(x[n - 1]) - ord('0'):
#         print(n, ans, ord(x[n - 1]) - ord('0'), flush=True)
```

Task B ()

```
#include <bits/stdc++.h>
using namespace std;
template<typename A, typename B = A> using pr = pair<A, B>;
#define vec vector
#define ar array
#define all(a) (a).begin(), (a).end()
#define len(a) ((int)((a).size()))

int main() {
    ios::sync_with_stdio(false), cin.tie(nullptr);

    int n, k;
    cin >> n >> k;
    string s;
    cin >> s;
    int ans = 0;
    for (int i = 0; i < n;) {
        int j = i;
        set<char> st;
        int to = min(n, i + k);
        while (j < to && len(st) <= 3) {
            st.insert(s[j]);
            j++;
        }
        ans++;
        if (len(st) == 4)
            i = j - 1;
        else
            i = j;
    }
    cout << ans << '\n';
}
```

Task C ()

```cpp
#include <bits/stdc++.h>
using namespace std;
template<typename A, typename B = A> using pr = pair<A, B>;
#define vec vector
#define ar array
#define all(a) (a).begin(), (a).end()
#define len(a) ((int)((a).size()))

const int A = 250001;
const int N = 500;

int dp[N][A];
bitset<A> par[N];

inline bool setmin(int &a, int b) {
    if (a > b) {
        a = b;
        return true;
    }
    return false;
}

int main() {
    ios::sync_with_stdio(false), cin.tie(nullptr);

    int n, x, y;
    cin >> n >> x >> y;
    vec<int> v(n);
    for (auto &x : v)
        cin >> x;
    vec<int> w(n);
    for (auto &x : w)
        cin >> x;

    for (int i = 0; i < n; i++)
        for (int j = 0; j < A; j++)
            dp[i][j] = y + 1;

    dp[0][0] = w[0];
    dp[0][v[0]] = 0;
    par[0][v[0]] = 1;

    for (int i = 1; i < n; i++)
        for (int j = 0; j < A; j++) {
            if (dp[i - 1][j] > y)
                continue;
            if (setmin(dp[i][j], dp[i - 1][j] + w[i]))
                par[i][j] = 0;
            if (j + v[i] < A)
                if (setmin(dp[i][j + v[i]], dp[i - 1][j]))
                    par[i][j + v[i]] = 1;
        }

    for (int val = 0; val <= x; val++)
        if (dp[n - 1][val] <= y) {
            string res(n, 'y');
            for (int i = n - 1; i >= 0; i--)
                if (par[i][val]) {
                    val -= v[i];
                    res[i] = 'x';
                }

            cout << res << '\n';
            return 0;
        }

    cout << "-1\n";
}
```

Task D ()

```
#include <bits/stdc++.h>
using namespace std;
template<typename A, typename B = A> using pr = pair<A, B>;
#define vec vector
#define ar array
#define all(a) (a).begin(), (a).end()
#define len(a) ((int)((a).size()))

int main() {
    ios::sync_with_stdio(false), cin.tie(nullptr);

    int n;
    string s;
    cin >> n >> s;
    n *= 2;
    vec<int> type(n);
    for (int i = 0; i < n; i++)
        type[i] = (s[i] == '[' || s[i] == ']');

    vec<int> st;
    for (int i = 0; i < n;) {
        int j = i;
        while (j < n && type[j] == type[i])
            j++;
        if ((j - i) % 2 == 0) {
            i = j;
            continue;
        }
        if (!len(st) || st.back() != type[i])
            st.push_back(type[i]);
        else
            st.pop_back();
        i = j;
    }
    assert(len(st) % 2 == 0);
    cout << len(st) / 2 << '\n';
}
```

Task E ()

```
#include <bits/stdc++.h>
using namespace std;
template<typename A, typename B = A> using pr = pair<A, B>;
#define vec vector
#define ar array
#define all(a) (a).begin(), (a).end()
#define len(a) ((int)((a).size()))

mt19937 rng(chrono::steady_clock::now().time_since_epoch().count());

vec<vec<int>> all3, all4;
vec<vec<int>> g;
vec<int> matched;
int timer;
vec<int> used;
map<vec<int>, vec<int>> to4;
map<vec<int>, int> which3;
map<vec<int>, int> bad4;

void get(int down, vec<int> already, int need, vec<vec<int>> &a) {
    if (need == 0) {
        a.push_back(already);
        return;
    }
    for (int i = down; i <= 10; i++) {
        already.push_back(i);
        get(i + 1, already, need - 1, a);
        already.pop_back();
    }
}

inline bool can(vec<int> &a, vec<int> &b) {
    assert(len(a) == 3 && len(b) == 4);
    assert(is_sorted(all(a)) && is_sorted(all(b)));
    for (int i = 0; i < 3; i++)
        if (upper_bound(all(b), a[i]) == lower_bound(all(b), a[i]))
            return false;
    return true;
}

bool try_kuhn(int v) {
    if (used[v] == timer)
        return false;
    used[v] = timer;
    for (auto u : g[v])
        if (matched[u] == -1) {
            matched[u] = v;
            return true;
        }

    for (auto u : g[v])
        if (try_kuhn(matched[u])) {
            matched[u] = v;
            return true;
        }
    return false;
}

inline int without(vec<int> &a, vec<int> &b) {
    assert(len(a) == 3 && len(b) == 4);
    assert(is_sorted(all(a)) && is_sorted(all(b)));
    for (int i = 0; i < 4; i++)
        if (lower_bound(all(a), b[i]) == upper_bound(all(a), b[i]))
            return b[i];
    assert(false);
}

inline void precalc10_3() {
    get(1, {}, 3, all3);
    get(1, {}, 4, all4);

    int n = len(all3);
    int m = len(all4);

    g.clear();
    matched.clear();
    used.clear();

    g.resize(n);
    matched.resize(m, -1);
    used.resize(n);

    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
            if (can(all3[i], all4[j]))
                g[i].push_back(j);

    for (int i = 0; i < n; i++) {
        timer++;
        assert(try_kuhn(i));
    }

    for (int i = 0; i < m; i++)
        if (matched[i] != -1) {
            to4[all4[i]] = all3[matched[i]];
            which3[all3[matched[i]]] = without(all3[matched[i]], all4[i]);
            bad4[all4[i]] = without(all3[matched[i]], all4[i]);
        }
}

vec<vec<int>> all5;
map<vec<int>, int> bad5;
map<vec<int>, int> which4;

inline bool can45(vec<int> &a, vec<int> &b) {
    assert(len(a) == 4 && len(b) == 5);
    assert(is_sorted(all(a)) && is_sorted(all(b)));
    for (int i = 0; i < 4; i++)
        if (upper_bound(all(b), a[i]) == lower_bound(all(b), a[i]))
            return false;
    return true;
}

inline int without45(vec<int> &a, vec<int> &b) {
    assert(len(a) == 4 && len(b) == 5);
    assert(is_sorted(all(a)) && is_sorted(all(b)));
    for (int i = 0; i < 5; i++)
        if (lower_bound(all(a), b[i]) == upper_bound(all(a), b[i]))
            return b[i];
    assert(false);
}

inline void precalc10_4() {
    get(1, {}, 5, all5);

    int n = len(all4);
    int m = len(all5);

    g.clear();
    matched.clear();
    used.clear();

    g.resize(n);
    matched.resize(m, -1);
    used.resize(n);

    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
            if (can45(all4[i], all5[j]))
                g[i].push_back(j);

    for (int i = 0; i < n; i++) {
        timer++;
        assert(try_kuhn(i));
    }
    for (int i = 0; i < m; i++)
        if (matched[i] != -1) {
            int el = without45(all4[matched[i]], all5[i]);
            bad5[all5[i]] = el;
            which4[all4[matched[i]]] = el;
        }
}

inline int solve2_10(vec<int> &els) {
    assert(len(els) == 2 && is_sorted(all(els)));
    int a = els[0], b = els[1];
    if (b == a + 1)
        return (b == 10 ? 1 : b + 1);
    if (b == 10 && a == 1)
        return 5;
    if (a != 1)
        return a - 1;
    assert(b != 10);
    return b + 1;
}

map<vec<int>, int> bad3;

inline void precalc10_2() {
    set<vec<int>> diff;
    for (int i = 1; i < 10; i++)
        for (int j = i + 1; j < 10; j++) {
            vec<int> cur{i, j};
            int el = solve2_10(cur);
            vec<int> ncur = cur;
            ncur.push_back(el);
            sort(all(ncur));
            assert(diff.find(ncur) == diff.end());
            diff.insert(ncur);
            bad3[ncur] = el;
        }
}

inline int solve4_10(vec<int> &els) {
    assert(len(els) == 4 && is_sorted(all(els)));
    return which4[els];
}

inline int solve3_10(vec<int> &els) {
    assert(len(els) == 3 && is_sorted(all(els)));
    return which3[els];
}

inline int solve1_10(int el) {
    return el % 10 + 1;
}

inline int solve0_10(vec<int> &els) {
    assert(len(els) == 0);
    return 1;
}

const int STEP_DELTA = 21;
const int START_DELTA = 97;
const int KEK = 7;

int solve(int n, vec<int> &els, int delta = START_DELTA) {
    // assert(len(els) <= (n / 10) * 3);
    // assert(is_sorted(all(els)));
    // for (auto x : els)
    //     assert(1 <= x && x <= n);
    // assert(n % 10 == 0);

    if (n == 10) {
        assert(len(els) <= 4);
        if (len(els) == 4)
            return solve4_10(els);
        else if (len(els) == 3)
            return solve3_10(els);
        else if (len(els) == 2)
            return solve2_10(els);
        else if (len(els) == 1)
            return solve1_10(els[0]);
        return solve0_10(els);
    }

    vec<int> l(10), r(10);
    int len = n / 10;
    l[0] = 1;
    r[0] = l[0] + len - 1;
    for (int i = 1; i < 10; i++) {
        l[i] = r[i - 1] + 1;
        r[i] = l[i] + len - 1;
    }

    vec<vec<int>> groups(10);
    int j = 0;
    for (int i = 0; i < len(els); i++) {
        while (r[j] < els[i])
            j++;
        assert(l[j] <= els[i] && els[i] <= r[j]);
        groups[j].push_back(els[i] - l[j] + 1);
    }

    for (int i = 0; i < 10; i++)
        if (len(groups[i]) < min((n / 10) * 3 + delta, (n / 10) * 4))
            return l[i] - 1 + solve(n / 10, groups[i], delta - STEP_DELTA);
    assert(false);
}

inline int restore2_10(vec<int> &els) {
    assert(len(els) == 2);
    if (solve1_10(els[0]) == els[1]) {
        assert(solve1_10(els[1]) != els[0]);
        return els[1];
    }
    assert(solve1_10(els[1]) == els[0]);
    return els[0];
}

int restore(int n, vec<int> &els, int delta = START_DELTA) {
    assert(len(els));
    // assert(3 * (len(els) - 1) <= n);
    // assert(is_sorted(all(els)));
    // for (auto x : els)
    //     assert(1 <= x && x <= n);
    // assert(n % 10 == 0);

    if (n == 10) {
        assert(len(els) <= 5);
        if (len(els) == 5)
            return bad5[els];
        else if (len(els) == 4)
            return bad4[els];
        else if (len(els) == 3)
            return bad3[els];
        else if (len(els) == 2)
            return restore2_10(els);
        return els[0];
    }

    vec<int> l(10), r(10);
    int len = n / 10;
    l[0] = 1;
    r[0] = l[0] + len - 1;
    for (int i = 1; i < 10; i++) {
        l[i] = r[i - 1] + 1;
        r[i] = l[i] + len - 1;
    }

    vec<vec<int>> groups(10);
    int j = 0;
    for (int i = 0; i < len(els); i++) {
        while (r[j] < els[i])
            j++;
        assert(l[j] <= els[i] && els[i] <= r[j]);
        groups[j].push_back(els[i] - l[j] + 1);
    }

    for (int i = 0; i < 10; i++)
        if (len(groups[i]) && len(groups[i]) - 1 < min((n / 10) * 3 + delta, (n / 10) * 4 - KEK *
            (n != 100)))
            return l[i] - 1 + restore(n / 10, groups[i], delta - STEP_DELTA);
    assert(false);
}

int main() {
    ios::sync_with_stdio(false), cin.tie(nullptr);

    precalc10_3();
    precalc10_4();
    precalc10_2();

    string s;
    cin >> s;
    if (s == "add") {
        int t;
        cin >> t;
#ifdef LOCAL
        cout << "clear\n";
        cout << t << '\n';
#endif
        while (t--) {
            int n, k;
            cin >> n >> k;
            vec<int> a(k);
            for (auto &x : a)
                cin >> x;
            sort(all(a));

            if (n == 1000000 && k <= 10) {
                cout << "134562\n";
                continue;
            }
            if (n == 10 && k == 3) {
                cout << solve3_10(a) << '\n';
                continue;
            }
#ifdef LOCAL
            a.push_back(solve(n, a));
            cerr << a.back() << endl;
            shuffle(all(a), rng);
            cout << n << '␣' << k << '\n';
            for (auto x : a)
                cout << x << '␣';
            cout << '\n';
#else
            cout << solve(n, a) << '\n';
#endif
        }
        // cerr << "------------------------------------------------" << endl;
        // cerr << endl;
    } else {
        int t;
        cin >> t;
        while (t--) {
            int n, k;
            cin >> n >> k;
            vec<int> a(k + 1);
            for (auto &x : a)
                cin >> x;
            sort(all(a));

            if (n == 1000000 && k <= 10) {
                a.erase(find(all(a), 134562));
                for (auto x : a)
                    cout << x << '␣';
                cout << '\n';
                continue;
            }

            if (n == 10 && k == 3) {
                for (auto x : to4[a])
                    cout << x << '␣';
                cout << '\n';
                continue;
            }

            int bad = restore(n, a);
#ifdef LOCAL
            cout << bad << endl;
            assert(find(all(a), bad) != a.end());
#else
            assert(find(all(a), bad) != a.end());
            a.erase(find(all(a), bad));
            for (auto x : a)
                cout << x << '␣';
            cout << '\n';
#endif
        }
    }
}
```

Task F ()

```
#include <bits/stdc++.h>
using namespace std;
template<typename A, typename B = A> using pr = pair<A, B>;
#define vec vector
#define ar array
#define all(a) (a).begin(), (a).end()
#define len(a) ((int)((a).size()))

inline void solve1(int n) {
    cout << "4\n";
    cout << "-1␣-1\n";
    cout << "-1␣1\n";
    cout << "1␣1\n";
    cout << "1␣-1\n";
    vec<pr<int>> vecs{{0, 2}, {0, -2}, {2, 0}, {-2, 0}, {-2, -2}, {-2, 2}, {2, 2}, {2, -2}};
    for (int i = 0; i < n; i++)
        cout << vecs[i].first << '␣' << vecs[i].second << '\n';
    exit(0);
}

int main() {
    ios::sync_with_stdio(false), cin.tie(nullptr);

    int n;
    cin >> n;
    if (n <= 8)
        solve1(n);
}
```