

Олимпиада СПбГУ по информатике 2020/21 учебного года

A	B	C	D	E	F	Sum
100	100	100	100	55	0	455

Task A ()

```
#include <bits/stdc++.h>
#define int long long
using namespace std;

int32_t main() {
    int k;
    cin >> k;
    cout << ((k - 2) / 9 + k) % 10;
}
```

Task B ()

```
#include <bits/stdc++.h>
#define int long long
using namespace std;

int32_t main() {
    int n, k;
    cin >> n >> k;
    string s;
    cin >> s;

    vector<int> v(300, 0);
    int count = 0;
    int ans = 0;
    int len = 0;
    for (int i = 0; i < n; i++) {
        if (count == 3 && v[s[i]] == 0 || len == k) {
            ans++;
            v.clear();
            v.resize(300, 0);
            count = 1;
            len = 0;
            v[s[i]] = 0;
        } else if (v[s[i]] == 0) {
            count++;
        }
        v[s[i]]++;
        len++;
    }

    cout << ans + 1;
}
```

Task C ()

```
#pragma Optimize("Ofast")
#include <bits/stdc++.h>
// #define int long long
using namespace std;

struct Link {
    int w, v, id;
    Link() : v(-1), w(-1) {}

    bool operator< (const Link &rsh) const {
        return v < rsh.v;
    }
};

int32_t main() {
    int n, x, y;
    cin >> n >> x >> y;

    vector<Link> l(n);
    for(int i = 0; i < n; i++) l[i].id = i;
    for(int i = 0; i < n; i++) cin >> l[i].v;
    for(int i = 0; i < n; i++) cin >> l[i].w;
    sort(l.begin(), l.end());

    vector<vector<int>> dp(x + 1, vector<int>(501, -1));

    // for(int i = 0; i < l.size(); i++) cout << l[i].v << ' ' << l[i].w << '\n';

    int mx = 0;
    dp[0][0] = 0;
    for(int st = 0; st < n; st++) {
        for(int i = 0; i <= mx; i++) {
            if(dp[i][st] != -1) {
                if(dp[i][st] + l[st].w <= y && (dp[i][st + 1] == -1 || dp[i][st + 1] > dp[i][st] + l[st].w)) {
                    dp[i][st + 1] = dp[i][st] + l[st].w;
                }

                if(i + l[st].v <= x && (dp[i + l[st].v][st + 1] == -1 || dp[i + l[st].v][st + 1] > dp[i][st])) {
                    dp[i + l[st].v][st + 1] = dp[i][st];
                    mx = max(i + l[st].v, mx);
                }
            }
        }
    }

    for(int i = 0; i <= x; i++) if(dp[i][n] != -1) {
        vector<char> ans;
        int st = n;
        int sum = i;
        while(st != 0) {
            if(sum - l[st - 1].v >= 0 && dp[sum - l[st - 1].v][st - 1] == dp[sum][st]) {
                sum -= l[st - 1].v;
                ans.push_back('x');
            } else {
                ans.push_back('y');
            }
            st--;
        }
        vector<char> buff(n);
        for(int i = ans.size() - 1; i >= 0; i--) buff[l[n - i - 1].id] = ans[i];
        for(int i = 0; i < buff.size(); i++) cout << buff[i];
        return 0;
    }

    cout << -1;
}
```

Task D ()

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

using namespace std;

int32_t main() {
    int n;
    cin >> n;
    deque<int> d;
    map<char, int> mp;

    mp.insert({'[', 1});
    mp.insert({'}', 1});
    mp.insert({')', 0});
    mp.insert({'(', 0});

    for (int i = 0; i < n * 2; i++) {
        char c;
        cin >> c;
        int id = mp[c];
        if (d.size() != 0 && id == d.back()) d.pop_back();
        else d.push_back(id);
    }

    cout << d.size() / 2;
}
```

Task E ()

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

using namespace std;

int seed = 1630932;

int MXLINE = 3;

vector<vector<int>> gen3;
vector<vector<int>> gen4;

bool hover(vector<int> &v1, vector<int> &v2) {
    set<int> s;
    for(int i = 0; i < v1.size(); i++) {
        s.insert(v1[i]);
    }
    for(int i = 0; i < v2.size(); i++) if(s.count(v2[i]) == 0) return false;
    return true;
}

void generate() {
    for(int i = 0; i < 10; i++)
        for(int j = i + 1; j < 10; j++)
            for(int k = j + 1; k < 10; k++) {
                vector<int> ob;
                ob.push_back(i);
                ob.push_back(j);
                ob.push_back(k);
                gen3.push_back(ob);
            }

    for(int i = 0; i < 10; i++)
        for(int j = i + 1; j < 10; j++)
            for(int k = j + 1; k < 10; k++)
                for(int c = k + 1; c < 10; c++) {
                    vector<int> ob;
                    ob.push_back(i);
                    ob.push_back(j);
                    ob.push_back(k);
                    ob.push_back(c);
                    gen4.push_back(ob);
                }
}

vector<vector<int>> g;

vector<int> mt;
vector<char> used;
bool try_kuhn (int v) {
    if (used[v]) return false;
    used[v] = true;
    for(int i = 0; i < g[v].size(); i++) {
        int to = g[v][i];
        if(mt[to] == -1) {
            mt[to] = v;
            return true;
        }
    }
}

for (size_t i=0; i<g[v].size(); ++i) {
    int to = g[v][i];
    if (try_kuhn (mt[to])) {
        mt[to] = v;
        return true;
    }
}
return false;
}

vector<int> answer;
```

```

void create_edges() {
    g.resize(gen3.size() + gen4.size());
    for(int i = 0; i < gen3.size(); i++) {
        for(int j = 0; j < gen4.size(); j++) {
            if(hover(gen4[j], gen3[i])) {
                g[i].push_back(j);
            }
        }
    }

    mt.assign(gen4.size(), -1);
    for (int v=0; v<gen3.size(); ++v) {
        used.assign (gen3.size(), false);
        try_kuhn (v);
    }

    answer.resize(gen3.size());
    for(int i = 0; i < mt.size(); i++) if(mt[i] != -1) answer[mt[i]] = i;
}

bool is_gen = false;
void GEN() {
    if(is_gen) return;
    generate();
    create_edges();
    is_gen = true;
}

void add() {
    int n, k;
    cin >> n >> k;
    set<int> s;
    for(int i = 0; i < k; i++) {
        int t;
        cin >> t;
        s.insert(t);
    }
    if(n == 10 && k == 3) {
        GEN();
        int id = -1;
        for(int i = 0; i < gen3.size(); i++) {
            int it = 0;
            bool f = true;
            for(int a : s) if(a - 1 != gen3[i][it++]) {f = false; break;}
            if(f) {id = i; break;}
        }

        for(int e : gen4[answer[id]]) if(s.count(e + 1) == 0) {cout << e + 1 << '\n'; return;}
    }

    if(n > 10) MXLINE = 6;

    set<int> lines;
    int line = 0;
    int last = -1;
    while(true) {
        int ni = rand() % n + 1;
        if(s.count(ni) == 0 && lines.count(ni) == 0) {
            line++;
            last = ni;
            lines.insert(ni);
        } else {
            line = 0;
            last = -1;
            lines.clear();
        }
        if(line == MXLINE) {cout << last << '\n'; return;}
    }
}

```

```

void clr() {
    int n, k;
    cin >> n >> k;
    set<int> s;
    for(int i = 0; i < k + 1; i++) {
        int t;
        cin >> t;
        s.insert(t);
    }

    if(n == 10 && k == 3) {
        GEN();
        int id = -1;
        for(int i = 0; i < gen4.size(); i++) {
            int it = 0;
            bool f = true;
            for(int a : s) if(a - 1 != gen4[i][it++]) {f = false; break;}
            if(f) {id = i; break;}
        }

        for(int e : gen3[mt[id]]) cout << e + 1 << '␣';
        cout << '\n';
        return;
    }

    if(n > 10) MXLINE = 6;

    set<int> lines;
    while(true) {
        int ni = rand() % n + 1;
        if(s.count(ni) == 0 && lines.count(ni) == 0) {
            lines.insert(ni);
        } else lines.clear();
        if(lines.size() == MXLINE - 1) break;
    }
    int ans = rand() % n + 1;
    for(int i : s) if(i != ans) cout << i << '␣';
    cout << '\n';
}

int32_t main() {
    // GEN();
    // for(int i = 0; i < 10; i++, cout << '\n') for(int j : gen3[i]) cout << j << ' ';

    srand(seed);
    string mode;
    cin >> mode;

    int t;
    cin >> t;

    while(t--) {
        if(mode == "add") add();
        else clr();
    }
}

```

Task F ()