Олимпиада СПбГУ по информатике 2019/20 учебного года

A	B	C	D	E	F	Sum
100	100	100	40	0	0	340

Task A ()

```
//#pragma GCC target("avx2")

#include <algorithm>
#include <iostream>
#include <vector>
#include <cmath>
#include <set>

using namespace std;

typedef long long ll;

#define sp << ' ' <<
#define nl << '\n'
#define sq << ' '

void solve();

int main() {/*
    ios_base::sync_with_stdio(0);
    cin.tie(0);*/
    solve();
}

int n;

void solve() {
    cin >> n;
    cout << n - 1 nl;
}
```

Task B ()

```
//#pragma GCC target("avx2")

#include <algorithm>
#include <iostream>
#include <iomanip>
#include <vector>
#include <cmath>
#include <set>

using namespace std;

typedef long long ll;
typedef long double ld;

#define sp << '␣' <<
#define nl << '\n'
#define sq << '␣'

void solve();

int main() {/*
    ios_base::sync_with_stdio(0);
    cin.tie(0);*/
    solve();
}

int n;
vector<pair<ld, ld>> a;

pair<ld, ld> calc(pair<ld, ld> a, pair<ld, ld> b, ld side) {
    ld ang = atan2(b.second - a.second, b.first- a.first);
    ld ang2 = 30*atan2(0, -1)/180.0;
    ld x1 = side * cos(ang - ang2);
    ld y1 = side * sin(ang - ang2);
    return {a.first + x1, a.second + y1};
}

void pp(pair<ld, ld> a) {
    cout << setprecision(10) << fixed << a.first sp a.second nl;
}

void restore() {
    ld c = hypot(a[0].first - a[1].first, a[0].second - a[1].second);
    vector<pair<ld, ld>> z;
    ld side = c / sqrtl(3.0);

    pp(a[0]);
    pp(calc(a[0], a[1], side));
    pp(a[1]);
    pp(calc(a[1], a[2], side));
    pp(a[2]);
    pp(calc(a[2], a[0], side));
}

void solve() {
    cin >> n;
    a.resize(n);

    for (auto &i: a) cin >> i.first >> i.second;

    for (int i = 1; i < n; ++i) {
        if (a[i].first < a[0].first || (a[i].first == a[0].first && a[i].second < a[0].second)) {
            swap(a[i], a[0]);
        }
    }

    sort(a.begin() + 1, a.end(), [&a](const pair<ld, ld> &p, const pair<ld, ld> &q) {
        return atan2(p.first - a[0].first, p.second - a[0].second) > atan2(q.first - a[0].first, q
            .second - a[0].second);
    });

    /*for (auto &i: a) {
        cout << i.first sp i.second nl;
    }*/

    if (n == 3) {
        restore();
        return;
    }

    pp(a[0]);
    pp(a[2]);
    pp(a[4]);
}
```

Task C ()

```
#pragma GCC target("avx2")

#include <algorithm>
#include <iostream>
#include <vector>
#include <cmath>
#include <set>

using namespace std;

typedef long long ll;

#define sp << '␣' <<
#define nl << '\n'
#define sq << '␣'

void solve();

int main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    solve();
}

int n;
string t, s;
set<pair<string, int>> m;

bool calc2(string &q) {
    vector<vector<int>> d(q.size(), vector<int>(t.size()));
    int res = 0;

    for (int i = 0; i < q.size(); ++i) {
        for (int j = 0; j < t.size(); ++j) {
            if (q[i] == t[j]) {
                d[i][j] = 1 + (i > 0 && j > 0 ? d[i - 1][j - 1] : 0);
            } else {
                d[i][j] = max(i > 0 ? d[i - 1][j] : 0, j > 0 ? d[i][j - 1] : 0);
            }
            res = max(res, d[i][j]);
        }
    }

    return (res == q.size());
}

int calc3(string &q) {
    vector<vector<int>> d(q.size(), vector<int>(t.size()));
    int res = 0;

    for (int i = 0; i < q.size(); ++i) {
        for (int j = 0; j < t.size(); ++j) {
            if (q[i] == t[j]) {
                d[i][j] = 1 + (i > 0 && j > 0 ? d[i - 1][j - 1] : 0);
            } else {
                //d[i][j] = max(i > 0 ? d[i - 1][j] : 0, j > 0 ? d[i][j - 1] : 0);
                d[i][j] = (j > 0 ? d[i][j - 1] : 0);
            }
            res = max(res, d[i][j]);
        }
    }

    return res;
}

bool calc(string &q, int k, int last) {
    if (q.size() == k) {
        return true;
    }

    for (int i = last + 1; i < t.size(); ++i) {
        if (q[k] == t[i]) {
            if (calc(q, k + 1, i)) {
                return true;
            }
        }
    }

    return false;
}

void solve() {
    cin >> t >> n;

    /*for (int i = 0; i < t.size(); ++i) {
        string q = "";
        for (int j = i; j < t.size(); ++j) {
            q += t[j];
            m.insert({q, q.size()});
        }
    }*/

    int ans = 0;

    while (n--) {
        cin >> s;
        int res = 0;
        /*for (int i = 0; i < s.size(); ++i) {
            string q = "";
            for (int j = i; j < s.size() && j - i + 1 <= t.size(); ++j) {
                q += s[j];
                /*if (m.count({q, q.size()})) {
                    res = max(res, (int) q.size());
                }*/
                /*if (calc(q, 0, -1)) {
                    res = max(res, (int) q.size());
                }*/
                /*if (calc2(q)) {
                    res = max(res, (int) q.size());
                }
            }
        }*/
        res = calc3(s);
        ans += t.size() - res;
        //cout << "res" sp res <<endl;
    }

    cout << ans nl;
}
```

Task D ()

```
//#pragma GCC target("avx2")

#include <algorithm>
#include <iostream>
#include <vector>
#include <deque>
#include <cmath>
#include <set>

using namespace std;

typedef long long ll;

#define sp << '␣' <<
#define nl << '\n'
#define sq << '␣'

void solve();

int main() {/*
    ios_base::sync_with_stdio(0);
    cin.tie(0);*/
    solve();
}

struct node {
    ll i, j, res;
};

int n, m;
int si, sj, fi, fj;
vector<vector<pair<ll, ll>>> a, p;
vector<vector<ll>> d, u;
deque<node> q;

inline ll calc(int i, int j, int pi, int pj) {
    return abs(i - pi - a[pi][pj].first) + abs(j - pj - a[pi][pj].second);
}

inline void check(int i, int j) {
    if (i < 0 || i >= n || j < 0 || j >= m || u[i][j]) return;
    q.push_back({i, j, d[i][j]});
}

void solve() {
    cin >> n >> m;
    cin >> si >> sj >> fi >> fj;

    a.resize(n, vector<pair<ll, ll>>(m));
    p.resize(n, vector<pair<ll, ll>>(m, {-1, -1}));
    d.resize(n, vector<ll>(m, 1e9));
    u.resize(n, vector<ll>(m, 0));
    si--, sj--, fi--, fj--;

    for (auto &i: a) {
        for (auto &j: i) {
            cin >> j.first >> j.second;
        }
    }

    d[si][sj] = 0;
    q.push_back({si, sj, 0});

    while (q.size()) {
        int pi = q.front().i;
        int pj = q.front().j;
        int res = q.front().res;
        q.pop_front();

        if (res > d[pi][pj]) continue;
        //cout << "v" sp pi+1 sp pj+1 sp res nl;
        u[pi][pj] = 1;

        for (int i = 0; i < n; ++i) {
            for (int j = 0; j < m; ++j) {
                if (pi != i || pj != j) {
                    if (d[i][j] > d[pi][pj] + calc(i, j, pi, pj)) {
                        d[i][j] = d[pi][pj] + calc(i, j, pi, pj);
                        p[i][j] = {pi, pj};
                        q.push_back({i, j, d[i][j]});
                    }
                }
            }
        }

        /*check(pi + 1, pj);
        check(pi - 1, pj);
        check(pi, pj - 1);
        check(pi, pj + 1);*/
    }

    cout << d[fi][fj] nl;

    /*int i = fi, j = fj;

    while (i != -1) {
        cout << i+1 sp j+1 sp (d[i][j]) nl;
        auto pp = p[i][j];
        i = pp.first;
        j = pp.second;
    }

    cout << i+1 sp j+1 nl;*/
}
```

Task E ()

Task F ()