Олимпиада СПбГУ по информатике 2019/20 учебного года

A	B	C	D	E	F	Sum
100	100	100	0	45	0	345

Task A ()

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

using namespace std;

signed main() {
    ios_base::sync_with_stdio(false);
    cin.tie();

    int n;
    cin >> n;
    cout << n - 1;
}
```

Task B ()

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

using namespace std;

#define double long double

double key(double y, double x) {
//    x += 0.001;
//    y += 0.001;
    double norm = sqrtl(x * x + y * y);
    double acos_res = acos(x / norm);
    double asin_res = asin(y / norm);
    if (asin_res < 0) {
        return -acos_res;
    } else {
        return acos_res;
    }
}

signed main() {
    ios_base::sync_with_stdio(false);
    cin.tie();

    int n;
    cin >> n;

    if (n == 6) {
        vector<pair<double, pair<double, double>>> points;
        for (int i = 0; i < n; ++i) {
            double x, y;
            cin >> x >> y;
            points.push_back({0, {x, y}});
        }

        sort(points.begin(), points.end());
        double x0 = (points[0].second.first + points[5].second.first) / 2.0;
        double y0 = (points[0].second.second + points[5].second.second) / 2.0;
        for (int i = 0; i < n; ++i) {
            points[i].first = key(points[i].second.first - x0, points[i].second.second - y0);
        }
        sort(points.begin(), points.end());

        for (int i = 0; i < 3; ++i) {
            cout << setprecision(239) << points[i].second.first << ' ' << points[i].second.second
                << '\n';
        }
    } else {
        vector<pair<double, pair<double, double>>> points;
        for (int i = 0; i < n; ++i) {
            double x, y;
            cin >> x >> y;
            points.push_back({key(y, x), {x, y}});
            cout << setprecision(239) << points[i].second.first << ' ' << points[i].second.second
                << '\n';
        }


        double x0 = (points[0].second.first + points[2].second.first) / 2.0;
        double y0 = (points[0].second.second + points[2].second.second) / 2.0;

        x0 -= points[1].second.first;
        y0 -= points[1].second.second;

        x0 *= 2;
        y0 *= 2;
        x0 += points[1].second.first;
        y0 += points[1].second.second;

        for (int i = 0; i < n; ++i) {
            points[i].second.first = (-(points[i].second.first - x0)) + x0;
            points[i].second.second = (-(points[i].second.second - y0)) + y0;
        }
```

```
        for (int i = 0; i < n; ++i) {
            cout << setprecision(239) << points[i].second.first << ' ' << points[i].second.second
                << '\n';
        }
    }
}
```

Task C ()

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

using namespace std;

signed main() {
    ios_base::sync_with_stdio(false);
    cin.tie();

    string t;
    cin >> t;
    int tests;
    cin >> tests;
    int res = 0;
    for (int test = 0; test < tests; ++test) {
        string s;
        cin >> s;
        int n = s.size();
        int m = t.size();
        int dp[n][m];
        for (int i = 0; i < n; ++i) {
            for (int j = 0; j < m; ++j) {
                dp[i][j] = 0;
            }
        }
//        if (s[0] == t[0]) {
//            dp[0][0] = 1;
//        }
        for (int j = 0; j < m; ++j) {
            if (s[0] == t[j]) {
                dp[0][j] = 1;
            }

        }
        for (int i = 1; i < n; ++i) {
            if (s[i] == t[0]) {
                dp[i][0] = 1;
            }

            for (int j = 1; j < m; ++j) {
                if (s[i] == t[j]) {
                    for (int k = 0; k < j; ++k) {
                        dp[i][j] = max(dp[i][j], dp[i - 1][k] + 1);
                    }
                }

            }
        }
        int best = 0;
        for (int i = 0; i < n; ++i) {
            for (int j = 0; j < m; ++j) {
                best = max(best, dp[i][j]);
            }
        }
        res += m - best;
//        cout << m - best << '\n';
    }

    cout << res;
}
```

Task D ()

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

using namespace std;

const int INF = 10000000 + 7;

signed main() {
    ios_base::sync_with_stdio(false);
    cin.tie();

    int n, m;
    cin >> n >> m;

    int sx, sy, fx, fy;
    cin >> sx >> sy >> fx >> fy;
    sx--;
    sy--;
    fx--;
    fy--;

    int dx[n][m];
    int dy[n][m];
    int dist[n][m];
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            cin >> dx[i][j];
            cin >> dy[i][j];
            dist[i][j] = INF;
        }
    }

//    vector<int> queue_x; wda
//    vector<int> queue_y;
//    queue_x.push_back(sx);
//    queue_y.push_back(sy);
    dist[sx][sy] = 0;
    int head = 0;

    vector<pair<int, int>> cur_level;
    vector<pair<int, int>> next_level;
    vector<pair<int, int>> next2_level;
    vector<pair<int, int>> next3_level;
//    cur_level.emplace_back(sx, sy);

    for (int ddx = -1; ddx <= 1; ++ddx) {
        for (int ddy = -1; ddy <= 1; ++ddy) {
            if (ddx == ddy) {
                continue;
            }
            int tmp_x = sx + ddx;
            int tmp_y = sy + ddy;
            if (0 <= tmp_x && tmp_x < n && 0 <= tmp_y && tmp_y <= m)  {
                dist[tmp_x][tmp_y] = abs(sx + dx[sx][sy] - tmp_x) + abs(sy + dy[sx][sy] - tmp_y);
                if (dist[tmp_x][tmp_y] == 0) {
                    cur_level.emplace_back(tmp_x, tmp_y);
                }
                else if (dist[tmp_x][tmp_y] == 1) {
                    next_level.emplace_back(tmp_x, tmp_y);
                } else if (dist[tmp_x][tmp_y] == 2) {
                    next2_level.emplace_back(tmp_x, tmp_y);
                } else {
                    next3_level.emplace_back(tmp_x, tmp_y);
                }

            }
        }
    }
    while (head >= cur_level.size()) {
        cur_level = next_level;
        next_level.clear();
        next_level = next2_level;
        next2_level.clear();
```

```
            next2_level = next3_level;
            next3_level.clear();
            head = 0;
        }
//      int head = 0;
        while (dist[fx][fy] == INF) {
            int cur_x = cur_level[head].first;
            int cur_y = cur_level[head].second;
            int next_x = cur_x + dx[cur_x][cur_y];
            int next_y = cur_y + dy[cur_x][cur_y];
            if (0 <= next_x && next_x < n && 0 <= next_y && next_y <= m)  {
                if (dist[next_x][next_y] > dist[cur_x][cur_y]) {
                    dist[next_x][next_y] = dist[cur_x][cur_y];
                    cur_level.emplace_back(next_x, next_y);
                }
            }

            for (int ddx = -1; ddx <= 1; ++ddx) {
                for (int ddy = -1; ddy <= 1; ++ddy) {
                    if (ddx == ddy) {
                        continue;
                    }

                    next_x = cur_x + ddx;
                    next_y = cur_y + ddy;
                    if (0 <= next_x && next_x < n && 0 <= next_y && next_y <= m)  {
                        if (dist[next_x][next_y] == INF) {
                            dist[next_x][next_y] = dist[cur_x][cur_y] + 1;
                            next_level.emplace_back(next_x, next_y);
                        }
                    }
                }
            }
            head++;
            if (head >= cur_level.size()) {
                cur_level = next_level;
                next_level.clear();
                next_level = next2_level;
                next2_level.clear();
                next2_level = next3_level;
                next3_level.clear();
                head = 0;
            }
        }

//      dist[fx][fy] += 1;

        cout << dist[fx][fy];
}
```

Task E ()

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

using namespace std;

#define int long long

const int PICS = 500;
//const int PICS = 8;

signed main() {
//    ios_base::sync_with_stdio(false);
//    cin.tie();

    int n, m, b;
    cin >> n >> m >> b;

    vector<pair<int, int>> dots;

    for (int i = 0; i < b; ++i) {
        int x, y;
        cin >> x >> y;
        dots.emplace_back(x, y);
    }
    dots.emplace_back(-1000000000000000000LL + 1, -1000000000000000000LL + 1);

    vector<int> xs;
    vector<int> ys;
    vector<int> hs;
    vector<bool> banned;
    int head = 0;
    int counter = 0;

    while (true) {
        int xcur = xs.size() * n;
        int ycur = xs.size() * m;

        cout << "?␣" << dots[0].first + xcur << '␣' << dots[0].second + ycur << '␣' << dots[0].
            first + xcur + n << '␣'
             << dots[0].second + ycur + m << endl;
        counter++;

        xs.push_back(xcur);
        ys.push_back(ycur);
        hs.push_back(1);
        banned.push_back(false);

        xs.push_back(xcur + n);
        ys.push_back(ycur + m);
        hs.push_back(1);
        banned.push_back(false);

        int xb, yb;
        cin >> xb >> yb;
        for (int i = 0; i < xs.size(); ++i) {
            if (xs[i] < xb && xb <= xs[i] + n && ys[i] < yb && yb <= ys[i] + m) {
                banned[i] = true;
            }
        }

        int cur_cnt = 0;
        for (int i = 0; i < xs.size(); ++i) {
            if (!banned[i]) {
                cur_cnt++;
            }
        }

        if (cur_cnt >= PICS) {
            break;
        }
    }

    while (true) {
```

```
        while (banned[head]) {
            head++;
            head %= xs.size();
        }

        int pos1 = head;
        head++;
        head %= xs.size();
        while (banned[head]) {
            head++;
            head %= xs.size();
        }
        int pos2 = head;
        head++;
        head %= xs.size();



        cout << "? " << dots[hs[pos1]].first + xs[pos1] << ' ' << dots[hs[pos1]].second + ys[pos1]
             << ' ' << dots[hs[pos2]].first + xs[pos2] << ' ' << dots[hs[pos2]].second + ys[pos2]
            << endl;
        counter++;

//        if (counter > 8600) {
//            return -1;
//        }
        hs[pos1]++;
        hs[pos2]++;

        int xb, yb;
        cin >> xb >> yb;
        if (xb < 0 || yb < 0) {

        } else {
            for (int i = 0; i < xs.size(); ++i) {
                if (xs[i] < xb && xb <= xs[i] + n && ys[i] < yb && yb <= ys[i] + m) {
                    banned[i] = true;
                }
            }
        }

        for (int i = 0; i < xs.size(); ++i) {
            if (banned[i]) {
                continue;
            }

            if (hs[i] >= b) {
                cout << "! " << xs[i] + 1 << ' ' << ys[i] + 1 << endl;
                return 0;
            }
        }
    }
}
```

Task F ()