Олимпиада СПбГУ по информатике 2019/20 учебного года

A	B	C	D	E	F	Sum
100	100	100	40	100	0	440

Task A ()

```cpp
#include <bits/stdc++.h>
using namespace std;

int main() {
    int n;

    cin >> n;

    if (n == 1) {
        cout << 0 << '\n';
        return 0;
    }

    cout << n - 1 << '\n';
    return 0;
}
```

Task B ()

```cpp
#include <bits/stdc++.h>
using namespace std;

const double EPS = 1e-2;
const double INF = 1e10;

struct Point {
    double x, y;

public:
    Point() {}
    Point(double x, double y): x(x), y(y) {}

    Point operator + (const Point other) {
        return Point(x + other.x, y + other.y);
    }
    Point operator - (const Point other) {
        return Point(x - other.x, y - other.y);
    }

    Point operator * (double k) {
        return Point(x * k, y * k);
    }

    double dot_product(const Point other) {
        return x * other.x + y * other.y;
    }
    double cross_product(const Point other) {
        return x * other.y - other.x * y;
    }

    double len() {
        return hypot(x, y);
    }
    void norm() {
        double len = hypot(x, y);
        x /= len;
        y /= len;
    }
};

bool double_comp(double a, double b) {
    return fabs(a - b) < EPS;
}

int main() {
    int n;

    cin >> n;
    if (n == 6) {
        vector<pair<double, Point>> v(n);

        for (int i = 0; i < n; ++i) {
            cin >> v[i].second.x >> v[i].second.y;
        }

        // sort
        v[0].first = 0;
        double min_len = INF, max_len = -INF;
        for (int i = 1; i < n; ++i) {
            v[i].first = (v[i].second - v[0].second).len();
            min_len = min(min_len, v[i].first);
            max_len = max(max_len, v[i].first);
        }
        //cerr << min_len << ' ' << max_len << '\n';

        vector<Point> v_(n, Point(INF, INF));
        v_[0] = v[0].second;
        for (int i = 1; i < n; ++i) {
            if (double_comp(v[i].first, min_len)) {
                if (double_comp(v_[1].x, INF)) {
                    v_[1] = v[i].second;
```

```
                } else {
                    v_[5] = v[i].second;
                }
            } else if (double_comp(v[i].first, max_len)) {
                v_[3] = v[i].second;
            } else {
                if (double_comp(v_[2].x, INF)) {
                    v_[2] = v[i].second;
                } else {
                    v_[4] = v[i].second;
                }
            }
        }
        //for (int i = 0; i < n; ++i) {
        //    cerr << fixed << setprecision(15) << v[i].first << '\n';
        //}

        if (((v_[1] - v_[0]).cross_product(v_[5] - v_[0]) > EPS) ^\
            ((v_[2] - v_[0]).cross_product(v_[4] - v_[0]) > EPS)) {
            swap(v_[1], v_[5]);
        }

        for (int i = 0; i < n; i += 2) {
            cout << fixed << setprecision(15) << v_[i].x << ' ' << v_[i].y << '\n';
        }
        //cout << '\n';
        //for (int i = 0; i < n; ++i) {
        //    cerr << fixed << setprecision(15) << v_[i].x << ' ' << v_[i].y << '\n';
        //}
    } else {
        vector<Point> v(6);
        for (int i = 0; i < 6; i += 2) {
            cin >> v[i].x >> v[i].y;
        }
        double side = (v[2] - v[0]).len() / sqrt(0.75);

        for (int i = 0; i < 6; i += 2) {
            Point suppa = v[(i + 2) % 6] - v[i] + (v[(i + 4) % 6] - v[(i + 2) % 6]) * 0.5;
            suppa.norm();
            suppa = suppa * side;

            v[(i + 3) % 6] = (v[i] + suppa);
        }
        for (int i = 0; i < 6; ++i) {
            cout << fixed << setprecision(15) << v[i].x << ' ' << v[i].y << '\n';
        }
    }
    return 0;
}
```

Task C ()

```cpp
#ifdef FAIRLY_LOCAL
#define _GLIBCXX_DEBUG
#endif // FAIRLY_LOCAL
#include <bits/stdc++.h>
using namespace std;

int get_max_co(string &t, string &s, int j) {
    int n = t.size(), m = s.size();
    int p1 = 0, p2 = j;
    while (p1 < n && p2 < m) {
        while (p1 < n && t[p1] != s[p2]) {
            ++p1;
        }

        if (p1 >= n) {
            break;
        }
        ++p1;
        ++p2;
    }

    return p2 - j;
}

int main() {
    string t, s;
    int n;
    int ans = 0;

    cin >> t >> n;
    for (int i = 0; i < n; ++i) {
        cin >> s;

        int max_co = 0;
        for (int j = 0; j < int(s.size()); ++j) {
            max_co = max(max_co, get_max_co(t, s, j));
        }
        //cerr << max_co << '\n';

        ans += t.size() - max_co;
    }

    cout << ans << '\n';
    return 0;
}
```

Task D ()

```
#ifdef FAIRLY_LOCAL
#define _GLIBCXX_DEBUG
#endif // FAIRLY_LOCAL
#include <bits/stdc++.h>
using namespace std;

const int INF = 1e9;

int n, m;
pair<int, int> a, b;
vector<vector<pair<int, int>>> v;

bool check(pair<int, int> v) {
    return v.first >= 0 && v.first < n && v.second >= 0 && v.second < m;
}

int len(pair<int, int> a, pair<int, int> b) {
    return abs(b.first - a.first - v[a.first][a.second].first) +\
           abs(b.second - a.second - v[a.first][a.second].second);
}

const pair<int, int> DELTA[] = {{0, -1}, {0, +1}, {+1, 0}, {-1, 0}, {+1, +1}, {+1, -1}, {-1, +1},
    {-1, -1}};

int main() {
    cin >> n >> m >> a.first >> a.second >> b.first >> b.second;
    v.resize(n, vector<pair<int, int>> (m));
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            cin >> v[i][j].first >> v[i][j].second;
        }
    }

    //cerr << "here\n";

    set<pair<int, pair<int, int>>> s;
    vector<vector<int>> dist(n, vector<int> (m)), used(n, vector<int> (m));

    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            if (i == a.first - 1 && j == a.second - 1) {
                dist[i][j] = 0;
            } else {
                dist[i][j] = len({a.first - 1, a.second - 1}, {i, j});
            }

            //cerr << dist[i][j] << ' ';

            s.insert({dist[i][j], {i, j}});
        }
        //cerr << '\n';
    }

    for (int i = 0; i < n * m; ++i) {
        pair<int, int> u = (s.begin())->second;
        s.erase(s.begin());

        used[u.first][u.second] = 1;
        //cerr << u.first << ' ' << u.second << '\n';

        for (int i = 0; i < n; ++i) {
            for (int j = 0; j < m; ++j) {
                if (i == u.first && j == u.second) {
                    continue;
                }
                pair<int, int> v = {i, j};
                if (dist[u.first][u.second] + len(u, v) < dist[v.first][v.second]) {
                    s.erase({dist[v.first][v.second], v});
                    dist[v.first][v.second] = dist[u.first][u.second] + len(u, v);
                    s.insert({dist[v.first][v.second], v});
                }
            }
```

```
        }
    }

    cout << dist[b.first - 1][b.second - 1] << '\n';

    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            //cerr << dist[i][j] << ' ';
        }
        //cerr << '\n';
    }
    return 0;
}
```

Task E ()

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

signed main() {
    int n, m, B;
    vector<pair<int, int>> b;

    cin >> n >> m >> B;
    b.resize(B);
    for (pair<int, int> &p : b) {
        cin >> p.first >> p.second;
        --p.first;
        --p.second;
    }

    int cur = 0;
    map<int, set<int>> copies;
    for (int i = 0; i < (1 << B); ++i) {
        copies[0].insert(i);
    }
    while (true) {
        if (copies[B].size()) {
            cout << "! " << 0 << ' ' << *(copies[B].begin()) * m << '\n';
            break;
        }

        if (copies[cur + 1].size() >= (1 << (B - cur - 1))) {
            ++cur;
        }
        //cerr << ">> " << cur << '\n';
        int i = *(copies[cur].begin());
        copies[cur].erase(copies[cur].begin());
        int j = *(copies[cur].begin());
        copies[cur].erase(copies[cur].begin());
        cout << "? " << b[cur].first << ' ' << i * m + b[cur].second << ' ' <<\
                        b[cur].first << ' ' << j * m + b[cur].second << endl;
        copies[cur + 1].insert(i);
        copies[cur + 1].insert(j);

        //for (auto &p : copies) {
        //    cerr << ">> " << p.first << ": ";
        //    for (int i : p.second) {
        //        cerr << i << " ";
        //    }
        //    cerr << "\n";
        //}

        pair<int, int> shot;
        cin >> shot.first >> shot.second;
        for (auto &p : copies) {
            if (p.second.count(shot.second / m)) {
                p.second.erase(shot.second / m);
                break;
            }
        }
    }
    return 0;
}
```

Task F ()

```
d = {2: "1", 3: "0 0 0 3", 4: "0 0 0 0 0 0 0 0 4 12"}
n, m = map(int, input().split())
i = 0
a = d[n].split()
while i < m:
    print(a[i], end=' ')
    i += 1
```