

Олимпиада СПбГУ по информатике 2019/20 учебного года

A	B	C	D	E	F	Sum
100	100	100	100	100	14	514

Task A ()

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

#define forn(i, n) for (int i = 0; i < n; ++i)

using namespace std;

#ifdef DEBUG
#define io freopen("input.txt", "r", stdin); freopen("output.txt", "w", stdout)
#else
#define io ios_base::sync_with_stdio(false); cin.tie(0); cout.tie(0)
#endif

typedef long long ll;
typedef long double ld;

/*
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/detail/standard_policies.hpp>

using namespace __gnu_pbds;

typedef tree<
    int,
    null_type,
    less<>,
    rb_tree_tag,
    tree_order_statistics_node_update
> ordered_set;
*/

signed main() {
    io;

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

Task B ()

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

#define forn(i, n) for (int i = 0; i < n; ++i)

using namespace std;

#ifdef DEBUG
#define io freopen("input.txt", "r", stdin); freopen("output.txt", "w", stdout)
#else
#define io ios_base::sync_with_stdio(false); cin.tie(0); cout.tie(0)
#endif

typedef long long ll;
typedef long double ld;

/*
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/detail/standard_policies.hpp>

using namespace __gnu_pbds;

typedef tree<
    int,
    null_type,
    less<>,
    rb_tree_tag,
    tree_order_statistics_node_update
> ordered_set;
*/

struct point {
    ld x, y;

    point() {}

    point(ld x, ld y) {
        this->x = x;
        this->y = y;
    }

    ld len() {
        return sqrt(x * x + y * y);
    }
};

ld sqr(ld x) {
    return x * x;
}

signed main() {
    io;

    int n;
    cin >> n;

    if (n == 6) {
        vector<point> pt(6);
        forn(i, n) {
            cin >> pt[i].x >> pt[i].y;
        }

        /*
        cout << "6\n";
        forn(i, n) {
            cout << pt[i].y << " " << pt[i].x << "\n";
        }
        return 0;
        */

        point fi = pt[0];

        forn(i, 6) {
```

```

        if (fi.y > pt[i].y || (fi.y == pt[i].y && fi.x > pt[i].x)) fi = pt[i];
    }

    forn(i, 6) {
        pt[i].x -= fi.x;
        pt[i].y -= fi.y;
    }

    sort(pt.begin(), pt.end(), [](point &a, point &b) {
        if (a.x * b.y - a.y * b.x == 0) {
            return (a.len() < b.len());
        }
        return (a.x * b.y - a.y * b.x > 0);
    });

    forn(i, 6) {
        pt[i].x += fi.x;
        pt[i].y += fi.y;
    }

    cout << setprecision(26) << fixed;
    forn(i, 3) cout << pt[i].x << " " << pt[i].y << "\n";
} else {
    vector<point> pt(6);
    forn(i, n) {
        cin >> pt[i].x >> pt[i].y;
    }

    ld len = sqrt(sqr(pt[1].x - pt[0].x) + sqr(pt[1].y - pt[0].y));
    point vec(-(pt[2].y - pt[0].y), pt[2].x - pt[0].x);
    ld lenvec = vec.len();
    vec.x = vec.x / lenvec * len;
    vec.y = vec.y / lenvec * len;

    point newpt(pt[2].x + vec.x, pt[2].y + vec.y);
    pt[3] = newpt;

    newpt.x = pt[3].x + (pt[0].x - pt[1].x);
    newpt.y = pt[3].y + (pt[0].y - pt[1].y);
    pt[4] = newpt;

    newpt.x = pt[0].x + vec.x;
    newpt.y = pt[0].y + vec.y;
    pt[5] = newpt;

    cout << setprecision(26) << fixed;
    forn(i, 6) cout << pt[i].x << " " << pt[i].y << "\n";
}
}

```

Task C ()

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

#define forn(i, n) for (int i = 0; i < n; ++i)

using namespace std;

#ifdef DEBUG
#define io freopen("input.txt", "r", stdin); freopen("output.txt", "w", stdout)
#else
#define io ios_base::sync_with_stdio(false); cin.tie(0); cout.tie(0)
#endif

typedef long long ll;
typedef long double ld;

/*
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/detail/standard_policies.hpp>

using namespace __gnu_pbds;

typedef tree<
    int,
    null_type,
    less<>,
    rb_tree_tag,
    tree_order_statistics_node_update
> ordered_set;
*/

signed main() {
    io;

    string t;
    cin >> t;
    vector<array<int, 26>> nextt(t.length());
    forn(i, t.length()) {
        forn(j, 26) {
            bool succ = false;
            for (int k = i; k < t.length(); ++k) {
                if (t[k] - 'a' == j) {
                    nextt[i][j] = k;
                    succ = true;
                    break;
                }
            }
            if (!succ) nextt[i][j] = -1;
        }
    }

    int n, ans = 0;
    cin >> n;
    forn(q, n) {
        string s;
        cin >> s;
        int maxx = 0;
        int len = s.length();
        for (int i = 0; i < len; ++i) {
            int sum = 0;
            int ptrt = 0;
            while (i + sum < len && ptrt < t.length()) {
                if (nextt[ptrt][s[i + sum] - 'a'] == -1) {
                    break;
                } else {
                    ptrt = nextt[ptrt][s[i + sum] - 'a'] + 1;
                    sum++;
                }
            }
            maxx = max(maxx, sum);
        }
        ans += t.length() - maxx;
    }
}
```

```
    cout << ans;  
}
```

Task D ()

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

#define forn(i, n) for (int i = 0; i < n; ++i)

using namespace std;

#ifdef DEBUG
#define io freopen("input.txt", "r", stdin); freopen("output.txt", "w", stdout)
#else
#define io ios_base::sync_with_stdio(false); cin.tie(0); cout.tie(0)
#endif

typedef long long ll;
typedef long double ld;

/*
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/detail/standard_policies.hpp>

using namespace __gnu_pbds;

typedef tree<
    int,
    null_type,
    less<>,
    rb_tree_tag,
    tree_order_statistics_node_update
> ordered_set;
*/

int INF = 1e9;

signed main() {
    io;

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

    cin >> n >> m;
    cin >> s.first >> s.second >> t.first >> t.second;
    s.first--, s.second--, t.first--, t.second--;

    vector<vector<pair<int, int>>> used(n, vector<pair<int, int>>>(m, {INF, INF}));
    vector<vector<pair<int, int>>> field(n, vector<pair<int, int>>>(m));
    forn(i, n) forn(j, m) cin >> field[i][j].first >> field[i][j].second;

    used[s.first][s.second].first = 0;
    set<array<int, 4>> q;
    q.insert({0, 1, s.first, s.second});

    auto update = [&](int x, int y, int dist, int flying) {
        if (x < 0 || y < 0 || x >= n || y >= m) return;

        if (flying) {
            if (used[x][y].second > dist) {
                q.erase({used[x][y].second, 0, x, y});
                used[x][y].second = dist;
                q.insert({used[x][y].second, 0, x, y});
            }
        } else {
            if (used[x][y].first > dist) {
                q.erase({used[x][y].first, 1, x, y});
                used[x][y].first = dist;
                q.insert({used[x][y].first, 1, x, y});
            }
        }
    };

    while (!q.empty()) {
        auto &temp = *q.begin();
        int dist = temp[0], flying = !temp[1], x = temp[2], y = temp[3];
        q.erase(q.begin());
```

```

        update(x + field[x][y].first , y + field[x][y].second , dist , 0);
        for (int i = -1; i <= 1; ++i) {
            for (int j = -1; j <= 1; ++j) {
                if (flying) {
                    update(x + i , y + j , dist + abs(i) + abs(j) , 1);
                }
                update(x + field[x][y].first + i , y + field[x][y].second + j , dist + abs(i) + abs(
                    j) , 1);
            }
        }
    }

    cout << min(used[t.first][t.second].first , used[t.first][t.second].second);
}

```

Task E ()

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

#define forn(i, n) for (int i = 0; i < n; ++i)

using namespace std;

#ifdef DEBUG
#define io freopen("input.txt", "r", stdin); freopen("output.txt", "w", stdout)
#else
#define io ios_base::sync_with_stdio(false); cin.tie(0); cout.tie(0)
#endif

typedef long long ll;
typedef long double ld;

/*
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/detail/standard_policies.hpp>

using namespace __gnu_pbds;

typedef tree<
    int,
    null_type,
    less<>,
    rb_tree_tag,
    tree_order_statistics_node_update
> ordered_set;
*/

ll n, m, b;
vector<pair<ll, ll>> pts;
vector<bool> dead;

void setpx(ll index, ll board1, ll board2) {
    cout << "?_";
    cout << n * board1 + pts[index].first + 1 << "_ " << pts[index].second + 1 << "_ ";
    cout << n * board2 + pts[index].first + 1 << "_ " << pts[index].second + 1 << endl;
    ll x, y;
    cin >> x >> y;
    x--, y--;
    x /= n;
    if (x >= 0 && x < dead.size())
        dead[x] = true;
}

signed main() {
    io;

    int ctr = -1;
    cin >> n >> m >> b;
    pts.resize(b);
    forn(i, b) {
        cin >> pts[i].first >> pts[i].second;
        pts[i].first--;
        pts[i].second--;
    }

    vector<vector<ll>> boards(b);
    dead.resize(1LL << b);

    forn(i, dead.size() / 2) {
        setpx(0, i * 2, i * 2 + 1);
    }

    forn(i, dead.size()) {
        if (!dead[i]) boards[0].push_back(i);
    }
    for (int i = 0; i < b - 1; ++i) {
        vector<int> temp;
        forn(j, 1LL << (b - i - 2)) {
            int b1 = boards[i].back();
```



```

        boards[i].pop_back();
        int b2 = boards[i].back();
        boards[i].pop_back();

        setpx(i + 1, b1, b2);
        temp.push_back(b1);
        temp.push_back(b2);
    }
    for (int j : temp) {
        if (!dead[j]) boards[i + 1].push_back(j);
    }
}
cout << "!_ " << boards.back()[0] * n + 1 << "_ " << 1 << endl;
}

```

Task F ()

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

#define forn(i, n) for (int i = 0; i < n; ++i)

using namespace std;

#ifdef DEBUG
#define io freopen("input.txt", "r", stdin); freopen("output.txt", "w", stdout)
#else
#define io ios_base::sync_with_stdio(false); cin.tie(0); cout.tie(0)
#endif

typedef long long ll;
typedef long double ld;

/*
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/detail/standard_policies.hpp>

using namespace __gnu_pbds;

typedef tree<
    int,
    null_type,
    less<>,
    rb_tree_tag,
    tree_order_statistics_node_update
> ordered_set;
*/

int n, m;
vector<vector<char>>> g;
vector<int> ans;

int check(int ind) {
    deque<pair<int, int>> deq;
    vector<int> used(n, -1);
    deq.emplace_back(-1, ind);
    used[ind] = 0;
    while (!deq.empty()) {
        auto temp = deq.front();
        int prev = temp.first, v = temp.second;
        deq.pop_front();
        forn(u, n) {
            if (!g[v][u]) continue;
            if (used[u] != -1) {
                if (u != prev) {
                    return -1;
                }
            } else continue;
            deq.emplace_back(v, u);
            used[u] = used[v] + 1;
        }
    }
    forn(i, n) if (used[i] == -1) {
        return -1;
    }
    return accumulate(used.begin(), used.end(), 0);
}

void func(int i, int j) {
    if (i == n - 1) {
        int sum = 0;
        forn(i, n) {
            sum += check(i);
            if (sum == -1) break;
        }
        if (sum >= 0) ans[sum / 2 - 1]++;
    } else {
        g[i][j] = 0;
        g[j][i] = 0;
    }
}
```

```

        if (j + 1 < n) func(i, j + 1);
        else func(i + 1, i + 2);

        g[i][j] = 1;
        g[j][i] = 1;
        if (j + 1 < n) func(i, j + 1);
        else func(i + 1, i + 2);
    }
}

void solve(int i) {
    n = i;
    m = (n * n * n - n) / 6;
    g.assign(n, vector<char>(n));
    ans.assign(m, 0);
    func(0, 1);
    for (int i : ans) cout << i << "␣";
    cout << endl;
    cout.flush();
}

signed main() {
    io;

    cin >> n >> m;
    solve(n);
}

```