Олимпиада СПбГУ по информатике 2019/20 учебного года

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

Task A ()

```
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>
#include <iomanip>

using namespace std;

void task_a()
{
        int n; cin >> n; cout << n - 1;
}

int main()
{

        //cout << fixed << setprecision(20) << 1.0 / 3 << "\n";
        task_a();
        return 0;
}
```

Task B ()

```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <map>
#include <algorithm>
#include <iomanip>

using namespace std;

void task_a()
{
        int n; cin >> n; cout << n - 1;
}

double Dist(double x1, double y1, double x2, double y2)
{
        double dx = x1 - x2;
        double dy = y1 - y2;
        return sqrt(dx * dx + dy * dy);
}

void rnd(double& x)
{
        x = (int)(x + 0.5);
}

void print(vector<int> en)
{

        for (int j = 0; j < en.size(); j++)
        {
                cout << en[j] << "␣";
        }
        cout << '\n';

}
void print(vector<vector<int>> en)
{

        for (int i = 0; i < en.size(); i++)
        {
                for (int j = 0; j < en[i].size(); j++)
                {
                        cout << en[i][j] << "␣";
                }
                cout << '\n';
        }
        cout << '\n';
}

void task_b()
{
        cout << fixed << setprecision(3);
        //ifstream cin("i.txt");
        int n; cin >> n;

        int scl = 1000;

        vector<long double> r_x(9 - n), r_y(9 - n);

        vector<double> x(n), y(n);
        for (int i = 0; i < n; i++)
        {
                cin >> x[i] >> y[i];
                x[i] *= scl, y[i] *= scl;
        }

        if (n == 6)
        {
                map<int, vector<int>> pp;
                for (int i = 1; i < n; i++)
                {
                        pp.emplace(Dist(x[0], y[0], x[i], y[i]),vector<int>()).first->second.
                          push_back(i);
                }

                vector<int> p(3);
                auto it = pp.begin();

                /*for (auto jt = pp.begin(); jt != pp.end(); jt++)
                        print(jt->second);
                cout << "\n";*/

                p[0] = it->second[0];
                p[1] = 0;
                if (it->second.size() == 2)
                        p[2] = it->second[1];
                else {
                        it++;
                        p[2] = it->second[0];
                }

                for (int i = 0; i < 3; i++)
                {
                        r_x[i] = x[p[i]];
                        r_y[i] = y[p[i]];
                }
        }
        else
        {
                // find the center
                double v1x = x[0] - x[1], v1y = y[0] - y[1];
                double v2x = x[2] - x[1], v2y = y[2] - y[1];
                double vcx = (v1x + v2x), vcy = (v1y + v2y);
                double ccx = x[1] + vcx, ccy = y[1] + vcy;

                // calc with center
                for (int i = 0; i < 3; i++)
                {
                        r_x[i] = x[i], r_y[i] = y[i];
                        r_x[i + 3] = ccx * 2 - x[i], r_y[i + 3] = ccy * 2 - y[i];
                }
        }

        for (int i = 0; i < 9 - n; i++)
        {
                cout << r_x[i]/scl << "␣" << r_y[i]/scl << '\n';
        }
}

int cnt_cmp(string s, int s_i, string t)
{
        int j = s_i;
        for (int i = 0; i < t.length(); i++)
        {
                if (t[i] == s[j]) j++;
        }
        return j - s_i;
}

void task_c()
{
        //ifstream cin("i.txt");
        string t; cin >> t;
        int n; cin >> n;
        int res = 0;
        while (n --> 0)
        {
                string s; cin >> s;
                int mx = 0;
                for (int i = 0; i < s.length(); i++)
                {
                        mx = max(mx, cnt_cmp(s, i, t));
                }
                res += t.length() - mx;
        }
        cout << res;
}

void swim(int x, int y, const vector<vector<pair<int, int>>>& vs, vector<vector<int>>& en, int val
    )
{
        if (x < 0 || x >= vs[0].size()) return;
        if (y < 0 || y >= vs.size()) return;

        if (en[y][x] >= 0 && en[y][x] <= val) return;

        en[y][x] = val;
        swim(x + vs[y][x].first, y + vs[y][x].second, vs, en, val);
}

void task_d()
{
        //ifstream cin("i.txt");
        int n, m; cin >> n >> m;
        int ax, ay; cin >> ay >> ax; ax--, ay--;
        int bx, by; cin >> by >> bx; by--, bx--;
        vector<vector<pair<int, int>>> vs(n, vector<pair<int, int>>(m));
        vector<vector<int>> en(n, vector<int>(m, -1));
        for (int i = 0; i < n; i++)
        {
                for (int j = 0; j < m; j++)
                {
                        cin >> vs[i][j].second >> vs[i][j].first;
                }
        }

        swim(ax, ay, vs, en, 0);
        for (int val = 1; en[by][bx] == -1; val++)
        {
                vector<vector<int>> nen = en;
                for (int i = 0; i < n; i++)
                {
                        for (int j = 0; j < m; j++)
                        {
                                if (en[i][j] == -1) continue;
                                for (int x = 0; x <= val-en[i][j]; x++)
                                {
                                        /*if (i == 2 && j == 3)
                                                system("pause");
*/
                                        int y = val - x - en[i][j];
                                        if (y < 0 || x < 0) break;
                                        {
                                                int X = j + vs[i][j].first + x;
                                                int Y = i + vs[i][j].second + y;
                                                swim(X, Y, vs, nen, val);
                                        }{
                                                int X = j + vs[i][j].first - x;
                                                int Y = i + vs[i][j].second + y;
                                                swim(X, Y, vs, nen, val);
                                        }{
                                                int X = j + vs[i][j].first + x;
                                                int Y = i + vs[i][j].second - y;
                                                swim(X, Y, vs, nen, val);
                                        }{
                                                int X = j + vs[i][j].first - x;
                                                int Y = i + vs[i][j].second - y;
                                                swim(X, Y, vs, nen, val);
                                        }
                                }
                        }
                }
                en = nen;
                /*print(en);
                system("pause");*/
        }
        cout << en[by][bx] << endl;
}
```

```
int main()
{
        //double pi = 3.141592653589793;
        //cout << cos(pi / 3) << sin(pi / 3);
        //cout << fixed << setprecision(20) << 1.0 / 3 << "\n";
        task_b();
        return 0;
}
```

Task C ()

```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <map>
#include <algorithm>
#include <iomanip>

using namespace std;

void task_a()
{
        int n; cin >> n; cout << n - 1;
}

double Dist(double x1, double y1, double x2, double y2)
{
        double dx = x1 - x2;
        double dy = y1 - y2;
        return sqrt(dx * dx + dy * dy);
}

void task_b()
{
        cout << fixed << setprecision(3);
        //ifstream cin("i.txt");
        int n; cin >> n;

        int scl = 10000;

        vector<long double> r_x(9 - n), r_y(9 - n);

        vector<double> x(n), y(n);
        for (int i = 0; i < n; i++)
        {
                cin >> x[i] >> y[i];
                x[i] *= scl, y[i] *= scl;
        }

        if (n == 6)
        {
                map<int, vector<int>> pp;
                for (int i = 1; i < n; i++)
                {
                        pp.emplace(Dist(x[0], y[0], x[i], y[i]),vector<int>()).first->second.
                            push_back(i);
                }

                vector<int> p(3);
                auto it = pp.begin();
                p[0] = pp.begin()->second[0];
                p[1] = 0;
                p[2] = (it->second.size() == 2) ? it->second[1] : ++it->second[0];

                for (int i = 0; i < 3; i++)
                {
                        r_x[i] = x[p[i]];
                        r_y[i] = y[p[i]];
                }
        }
        else
        {
                // sort
                vector<double> d(3);
                for (int i = 0; i < 3; i++)
                {
                        d[i] = Dist(x[(i + 1) % 3], y[(i + 1) % 3], x[(i + 2) % 3], y[(i + 2) %
                            3]);
                }

                if (d[0] > d[1] && d[0] > d[2])
                {
                        swap(x[0], x[1]);
                        swap(y[0], y[1]);
                }
                else if (d[2] > d[1] && d[2] > d[0])
                {
                        swap(x[2], x[1]);
                        swap(y[2], y[1]);
                }

                // find the offset
                double vx = -(y[0] - y[2]) / 2, vy = -(x[0] - x[2]) / 2;
                double mgn = sqrt(vx*vx + vy*vy);
                vx *= scl / mgn, vy *= scl / mgn;

                double dx = x[0] - x[1], dy = y[0] - y[1];
                mgn = sqrt(dx*dx + dy*dy);
                vx *= mgn / scl, vy *= mgn / scl;

                // calc with offset
                for (int i = 0; i < 3; i++)
                {
                        r_x[i] = x[i], r_y[i] = y[i];
                }
                r_x[3] = x[2] + vx, r_y[3] = y[2] + vy;
                r_x[4] = x[1] + vx * 2, r_y[4] = y[1] + vy * 2;
                r_x[5] = x[0] + vx, r_y[5] = y[0] + vy;
        }

        for (int i = 0; i < 9 - n; i++)
        {
                cout << r_x[i]/scl << "␣" << r_y[i]/scl << '\n';
        }
}

int cnt_cmp(string s, int s_i, string t)
{
        int j = s_i;
        for (int i = 0; i < t.length(); i++)
        {
                if (t[i] == s[j]) j++;
        }
        return j - s_i;
}

void task_c()
{
        //ifstream cin("i.txt");
        string t; cin >> t;
        int n; cin >> n;
        int res = 0;
        while (n --> 0)
        {
                string s; cin >> s;
                int mx = 0;
                for (int i = 0; i < s.length(); i++)
                {
                        mx = max(mx, cnt_cmp(s, i, t));
                }
                res += t.length() - mx;
        }
        cout << res;
}

int main()
{
        //double pi = 3.141592653589793;
        //cout << cos(pi / 3) << sin(pi / 3);
        //cout << fixed << setprecision(20) << 1.0 / 3 << "\n";
        task_c();
        return 0;
}
```

Task D ()

```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <map>
#include <algorithm>
#include <iomanip>

using namespace std;

void task_a()
{
        int n; cin >> n; cout << n - 1;
}

double Dist(double x1, double y1, double x2, double y2)
{
        double dx = x1 - x2;
        double dy = y1 - y2;
        return sqrt(dx * dx + dy * dy);
}

void task_b()
{
        cout << fixed << setprecision(3);
        //ifstream cin("i.txt");
        int n; cin >> n;

        int scl = 10000;

        vector<long double> r_x(9 - n), r_y(9 - n);

        vector<double> x(n), y(n);
        for (int i = 0; i < n; i++)
        {
                cin >> x[i] >> y[i];
                x[i] *= scl, y[i] *= scl;
        }

        if (n == 6)
        {
                map<int, vector<int>> pp;
                for (int i = 1; i < n; i++)
                {
                        pp.emplace(Dist(x[0], y[0], x[i], y[i]),vector<int>()).first->second.
                            push_back(i);
                }

                vector<int> p(3);
                auto it = pp.begin();
                p[0] = pp.begin()->second[0];
                p[1] = 0;
                p[2] = (it->second.size() == 2) ? it->second[1] : ++it->second[0];

                for (int i = 0; i < 3; i++)
                {
                        r_x[i] = x[p[i]];
                        r_y[i] = y[p[i]];
                }
        }
        else
        {
                // sort
                vector<double> d(3);
                for (int i = 0; i < 3; i++)
                {
                        d[i] = Dist(x[(i + 1) % 3], y[(i + 1) % 3], x[(i + 2) % 3], y[(i + 2) %
                            3]);
                }

                if (d[0] > d[1] && d[0] > d[2])
        {
            swap(x[0], x[1]);
            swap(y[0], y[1]);
        }
        else if (d[2] > d[1] && d[2] > d[0])
        {
            swap(x[2], x[1]);
            swap(y[2], y[1]);
        }

        // find the offset
        double vx = -(y[0] - y[2]) / 2, vy = -(x[0] - x[2]) / 2;
        double mgn = sqrt(vx*vx + vy*vy);
        vx *= scl / mgn, vy *= scl / mgn;

        double dx = x[0] - x[1], dy = y[0] - y[1];
        mgn = sqrt(dx*dx + dy*dy);
        vx *= mgn / scl, vy *= mgn / scl;

        // calc with offset
        for (int i = 0; i < 3; i++)
        {
            r_x[i] = x[i], r_y[i] = y[i];
        }
        r_x[3] = x[2] + vx, r_y[3] = y[2] + vy;
        r_x[4] = x[1] + vx * 2, r_y[4] = y[1] + vy * 2;
        r_x[5] = x[0] + vx, r_y[5] = y[0] + vy;
    }

    for (int i = 0; i < 9 - n; i++)
    {
        cout << r_x[i]/scl << "␣" << r_y[i]/scl << '\n';
    }
}

int cnt_cmp(string s, int s_i, string t)
{
    int j = s_i;
    for (int i = 0; i < t.length(); i++)
    {
        if (t[i] == s[j]) j++;
    }
    return j - s_i;
}

void task_c()
{
    //ifstream cin("i.txt");
    string t; cin >> t;
    int n; cin >> n;
    int res = 0;
    while (n --> 0)
    {
        string s; cin >> s;
        int mx = 0;
        for (int i = 0; i < s.length(); i++)
        {
            mx = max(mx, cnt_cmp(s, i, t));
        }
        res += t.length() - mx;
    }
    cout << res;
}

void swim(int x, int y, const vector<vector<pair<int, int>>>& vs, vector<vector<int>>& en, int val
  )
{
    if (x < 0 || x >= vs[0].size()) return;
    if (y < 0 || y >= vs.size()) return;

    if (en[y][x] >= 0 && en[y][x] <= val) return;

    en[y][x] = val;
    swim(x + vs[y][x].first, y + vs[y][x].second, vs, en, val);
```

```
}

void print(vector<vector<int>> en)
{

        for (int i = 0; i < en.size(); i++)
        {
                for (int j = 0; j < en[i].size(); j++)
                {
                        cout << en[i][j] << "␣";
                }
                cout << '\n';
        }
        cout << '\n';
}

void task_d()
{
        //ifstream cin("i.txt");
        int n, m; cin >> n >> m;
        int ax, ay; cin >> ay >> ax; ax--, ay--;
        int bx, by; cin >> by >> bx; by--, bx--;
        vector<vector<pair<int, int>>> vs(n, vector<pair<int, int>>(m));
        vector<vector<int>> en(n, vector<int>(m, -1));
        for (int i = 0; i < n; i++)
        {
                for (int j = 0; j < m; j++)
                {
                        cin >> vs[i][j].second >> vs[i][j].first;
                }
        }

        swim(ax, ay, vs, en, 0);
        for (int val = 1; en[by][bx] == -1; val++)
        {
                vector<vector<int>> nen = en;
                for (int i = 0; i < n; i++)
                {
                        for (int j = 0; j < m; j++)
                        {
                                if (en[i][j] == -1) continue;
                                for (int x = 0; x <= val-en[i][j]; x++)
                                {
                                        /*if (i == 2 && j == 3)
                                                system("pause");
*/
                                        int y = val - x - en[i][j];
                                        if (y < 0 || x < 0) break;
                                        {
                                                int X = j + vs[i][j].first + x;
                                                int Y = i + vs[i][j].second + y;
                                                swim(X, Y, vs, nen, val);
                                        }{
                                                int X = j + vs[i][j].first - x;
                                                int Y = i + vs[i][j].second + y;
                                                swim(X, Y, vs, nen, val);
                                        }{
                                                int X = j + vs[i][j].first + x;
                                                int Y = i + vs[i][j].second - y;
                                                swim(X, Y, vs, nen, val);
                                        }{
                                                int X = j + vs[i][j].first - x;
                                                int Y = i + vs[i][j].second - y;
                                                swim(X, Y, vs, nen, val);
                                        }
                                }
                        }
                }
                en = nen;
                /*print(en);
                system("pause");*/
        }
        cout << en[by][bx] << endl;
}
```

```
int main()
{
        //double pi = 3.141592653589793;
        //cout << cos(pi / 3) << sin(pi / 3);
        //cout << fixed << setprecision(20) << 1.0 / 3 << "\n";
        task_d();
        return 0;
}
```

Task E ()

Task F ()