

# Олимпиада СПбГУ по информатике 2019/20 учебного года

A	B	C	D	E	F	Sum
100	100	100	40	12	0	352

## Task A ()

```
#include <iostream>
#include <iomanip>
#include <cstdio>
#include <algorithm>
#include <functional>
#include <set>
#include <map>
#include <vector>
#include <unordered_set>
#include <unordered_map>
#include <cmath>
#include <numeric>

#define int long long
#define all(X) X.begin(), X.end()
#define mp make_pair

using namespace std;

signed main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);

    //freopen("input.txt", "r", stdin);

    int n; cin >> n;
    int ans = n / 2, last = ans;
    while(true) {
        int now = n + ans;
        now /= 2;
        ans = now;
        if(last == ans)
            break;
        last = ans;
    }
    cout << ans;
    return 0;
}
```

## Task B ()

```
#include <iostream>
#include <iomanip>
#include <cstdio>
#include <algorithm>
#include <functional>
#include <set>
#include <map>
#include <vector>
#include <unordered_set>
#include <unordered_map>
#include <cmath>
#include <numeric>

#define int long long
#define all(X) X.begin(), X.end()
#define mp make_pair
#define endl '\n'
#define double long double

using namespace std;

struct point{
    double x, y;
};

struct vect{
    double x, y;
    void norm(double len){
        double last = sqrt(x * x + y * y);
        x /= last; y /= last;
    }

    void rot(){
        swap(x, y);
        x *= -1;
    }
};

point operator += (point &a, vect v){
    a.x += v.x;
    a.y += v.y;
}

point operator -= (point &a, vect v){
    a.x -= v.x;
    a.y -= v.y;
}

point operator += (point &a, point v){
    a.x += v.x;
    a.y += v.y;
}

point operator -= (point &a, point v){
    a.x -= v.x;
    a.y -= v.y;
}

double dist(point a, point b){
    return sqrt((a.x - b.x) * (a.x - b.x) + (a.y - b.y) * (a.y - b.y));
}

int operator* (point a, point b){
    return a.x * b.y - b.x * a.y;
}

int zn(double a){
    if(a < 0)
        return -1;
}
```

```

        else
            return 1;
    }

bool flag(vector<point> p){
    bool ans = 1;
    int last = -1, n = p.size();

    vector<point> v;

    for(int i = 0; i < n; i++){
        int nxt = (i + 1) % n;
        v.push_back({p[nxt].x - p[i].x, p[nxt].y - p[i].y});
    }
    for(int i = 0; i < n; i++){
        int nxt = (i + 1) % n;
        if(zn(v[nxt] * v[i]) != last)
            ans = 0;
    }
    return ans;
}

signed main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);

    //freopen("input.txt", "r", stdin);

    int n; cin >> n;
    vector<point> p(n);
    for(int i = 0; i < n; i++)
        cin >> p[i].x >> p[i].y;

    if(n == 6){
        cout << fixed << setprecision(10);
        double ans = 1e18;
        vector<point> out(3);

        for(int i = 0; i < 6; i++){
            for(int j = 0; j < 6; j++){
                if(i == j)
                    continue;
                for(int q = 0; q < 6; q++){
                    if(i == q || j == q)
                        continue;
                    vector<double> order;
                    order.push_back(dist(p[i], p[j]));
                    order.push_back(dist(p[j], p[q]));
                    order.push_back(dist(p[q], p[i]));

                    double ctr = 0;
                    for(int w = 0; w < 3; w++){
                        for(int e = w + 1; e < 3; e++){
                            ctr += abs(order[w] - order[e]);
                        }
                    }

                    vector<point> now(3);
                    now[0] = p[i];
                    now[1] = p[j];
                    now[2] = p[q];

                    if(ctr < ans && flag(now)){
                        ans = ctr;
                        out = now;
                    }
                }
            }
        }
        for(int i = 0; i < 3; i++)
            cout << out[i].x << " " << out[i].y << endl;
    }
    else{
        cout << fixed << setprecision(4);
        double l = dist(p[0], p[1]) / 2.0, h = 1 / sqrt(3.0);
    }
}

```

```

        for(int i = 0; i < n; i++){
int  nxt = (i + 1) % n;
        point mid = {(p[nxt].x + p[i].x) / 2, (p[nxt].y + p[i].y) / 2};

        vect v = {(p[nxt].x - p[i].x), (p[nxt].y - p[i].y)};

        v.norm(h);
        v.rot();
        cout << p[i].x << "┘" << p[i].y << endl;
        mid -= v;
        cout << mid.x << "┘" << mid.y << endl;

    }
}
return 0;
}

```

## Task C ()

```
#include <iostream>
#include <iomanip>
#include <cstdio>
#include <algorithm>
#include <functional>
#include <set>
#include <map>
#include <vector>
#include <string>
#include <unordered_set>
#include <unordered_map>
#include <cmath>
#include <numeric>

// #define int long long
#define all(X) X.begin(), X.end()
#define mp make_pair
#define endl '\n'
#define double long double

using namespace std;

signed main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);

    // freopen("input.txt", "r", stdin);

    string t; cin >> t;
    int n; cin >> n;

    vector<set<int>> pos(26);
    for(int i = 0; i < t.size(); i++)
        pos[t[i] - 'a'].insert(i);

    int ans = 0;
    while(n--){
        string s; cin >> s;
        vector<vector<int>> dp(s.size() + 1, vector<int>(s.size(), -1));
        int temp = 0;

        for(int l = 0; l < (int)s.size(); l++){
            int cur = s[l] - 'a';
            if(!pos[cur].empty()){
                dp[l][l] = *pos[cur].begin();
                temp = 1;
            }
        }

        for(int len = 2; len <= (int)s.size(); len++){
            for(int l = 0; l < (int)s.size(); l++){
                if(dp[l][len - 1] != -1){
                    int now = l + len - 1, last = now - 1;
                    if(now >= (int)s.size())
                        break;
                    int cur = s[now] - 'a';

                    if(pos[cur].upper_bound(dp[l][len - 1]) != pos[cur].end())
                    {
                        dp[l][len] = *pos[cur].upper_bound(dp[l][len - 1]);
                        ;
                        temp = len;
                    }
                }
            }
        }
        ans += (int)t.size() - temp;
    }
    cout << ans;
    return 0;
}
```

## Task D ()

```
#include <iostream>
#include <iomanip>
#include <cstdio>
#include <algorithm>
#include <functional>
#include <set>
#include <map>
#include <vector>
#include <string>
#include <queue>
#include <deque>
#include <unordered_set>
#include <unordered_map>
#include <cmath>
#include <numeric>

#define int long long
#define all(X) X.begin(), X.end()
#define mp make_pair
#define endl '\n'
#define double long double

using namespace std;

int n, m;

const int INF = 1e9;
pair<int, int> g[1010][1010];
int dis[1010][1010];
bool used[1010][1010];

bool into(int row, int col){
    return 0 <= row && row < n && 0 <= col && col < m;
}

int ads(int n){
    if(n < 0)
        n *= -1;
    return n;
}

void djs(int row, int col){
    priority_queue<pair<int, pair<int, int>>> que;
    dis[row][col] = 0;
    que.push(mp(0, mp(row, col)));

    while(!que.empty()){
        pair<int, int> v = que.top().second;
        int r = v.first, c = v.second;
        que.pop();
        if(used[r][c])
            continue;
        int rto = r + g[r][c].first, cto = c + g[r][c].second;
        if(into(rto, cto)){
            if(dis[rto][cto] > dis[r][c]){
                dis[rto][cto] = dis[r][c];
                que.push(mp(-dis[rto][cto], mp(rto, cto)));
            }
        }

        for(int i = 0; i < n; i++){
            for(int j = 0; j < m; j++){
                int d = ads(i - r - g[r][c].first) + ads(j - c - g[r][c].second);
                if(dis[i][j] > dis[r][c] + d){
                    dis[i][j] = dis[r][c] + d;
                    que.push(mp(-dis[i][j], mp(i, j)));
                }
            }
        }
        used[r][c] = 1;
    }
}
```

```
}
```

```
signed main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);

    //freopen("input.txt", "r", stdin);

    cin >> n >> m;
    int a, b, c, d; cin >> a >> b >> c >> d;
    a--; b--; c--; d--;

    for(int i = 0; i < n; i++)
        for(int j = 0; j < m; j++)
            cin >> g[i][j].first >> g[i][j].second;

    for(int i = 0; i < n; i++)
        for(int j = 0; j < m; j++)
            dis[i][j] = INF;

    djs(a, b);
    int ans = dis[c][d];
    cout << ans;
    return 0;
}
```

## Task E ()

```
#include <iostream>
#include <iomanip>
#include <cstdio>
#include <algorithm>
#include <functional>
#include <set>
#include <map>
#include <vector>
#include <string>
#include <queue>
#include <deque>
#include <unordered_set>
#include <unordered_map>
#include <cmath>
#include <numeric>

#define int long long
#define all(X) X.begin(), X.end()
#define mp make_pair
// #define endl '\n'
#define double long double

using namespace std;

signed main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);

    // freopen("input.txt", "r", stdin);

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

    vector<pair<int, int>> p(b);
    for(int i = 0; i < b; i++){
        cin >> p[i].first >> p[i].second;
        p[i].first--; p[i].second--;
    }
    if(b == 1){
        int r = p[0].first, c = p[0].second;
        cout << "?_ " << r << "_ " << c << "_ " << r << "_ " << c + m << endl;
        int s, t; cin >> s >> t;
        if(s == r && t == c)
            cout << "!_ " << 0 << "_ " << m << endl;
        else
            cout << "!_ " << 0 << "_ " << 0 << endl;
    }
    else {
        int need = 1;
        for(int i = 0; i < b; i++)
            need *= 2;

        vector<pair<int, int>> d(need);
        for(int i = 0; i < need; i++)
            d[i] = mp(0, m * i);

        vector<bool> used(need);
        vector<int> ctr(need);

        while(true){
            int minim = 1000;

            for(int i = 0; i < need; i++){
                if(!used[i] && ctr[i] == b){
                    cout << "!_ " << d[i].first << "_ " << d[i].second << endl;
                    return 0;
                }
            }
            for(int i = 0; i < need; i++)
                if(!used[i])
                    minim = min(minim, ctr[i]);
            vector<int> order;
            for(int i = 0; i < need; i++)
```



```

        if (!used[i] && ctr[i] == minim)
            order.push_back(i);
    int a = order[0], b = order[1];
    int r1 = d[a].first + p[minim].first, c1 = d[a].second + p[minim].second;
    int r2 = d[b].first + p[minim].first, c2 = d[b].second + p[minim].second;
    cout << "?_ " << r1 << "_ " << c1 << "_ " << r2 << "_ " << c2 << endl;
    ctr[a]++; ctr[b]++;
    int s, t; cin >> s >> t;
    int ind = t / m;
    used[ind] = 1;
    }
    return 0;
}

```

## Task F ()