Олимпиада СПбГУ по информатике 2018/19 учебного года

Шабанов Денис Алексеевич

A	B	C	D	E	F	Sum
100	100	100	60	21	7	388

Task A (100)

```cpp
#include <iostream>
using namespace std;

int main() {
    int n, m;
    cin >> n >> m;

    for (int i = 0; i < 15; ++i) {
        if (n == m) {
            cout << "Yes" << endl;
            return 0;
        }
        n *= 2;
    }
    cout << "No" << endl;
    return 0;
}
```

Task B (100)

```cpp
#include <iostream>
#include <string>
using namespace std;

int main() {
    int n;
    string s;

    cin >> n >> s;

    if (n == 1) {
        cout << "No" << endl;
        return 0;
    } else {
        for (int i = 0; i < n - 1; ++i) {
            if (s[i] == 'o' && s[i + 1] == 'r') {
                cout << "Yes" << endl;
                return 0;
            }
        }

        for (int i = 0; i < n - 1; ++i) {
            swap(s[i], s[i + 1]);
            for (int j = max(i - 1, 0); j < min(i + 2, n); ++j) {
                if (s[j] == 'o' && s[j + 1] == 'r') {
                    cout << "Yes" << endl;
                    return 0;
                }
            }
            swap(s[i], s[i + 1]);
        }
    }

    cout << "No" << endl;
    return 0;
}
```

Task C (100)

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

vector <vector <int>> adj_list;
vector <int> used, sub, max_sub;

int dfs(int start) {
    used[start] = 1;

    int cur_sub = 0;
    for (int next: adj_list[start]) {
        if (!used[next]) {
            cur_sub += dfs(next);
        }
    }

    sub[start] = cur_sub + 1;
    return sub[start];
}

void dfs2(int start) {
    used[start] = 1;

    max_sub[start] = 1;
    for (int next: adj_list[start]) {
        if (!used[next]) {
            dfs2(next);
            max_sub[start] = max(max_sub[start], sub[next] + 1);
        }
    }
}

int main() {
    int n;

    cin >> n;
    adj_list.resize(n);
    used.resize(n, 0);
    max_sub.resize(n, 0);
    sub.resize(n, 0);
    for (int i = 0; i < n - 1; ++i) {
        int u, v;

        cin >> u >> v;
        --u; --v;

        adj_list[u].push_back(v);
        adj_list[v].push_back(u);
    }
    dfs(0);

    fill(used.begin(), used.end(), 0);

    // DEBUG
//    for (int i = 0; i < n; ++i) {
//        cout << sub[i] << " ";
//    }
//    cout << endl;

    dfs2(0);

    // DEBUG
//    for (int i = 0; i < n; ++i) {
//        cout << max_sub[i] << " ";
//    }
//    cout << endl;
```

```
    cout << max_sub[0] << " ";
    for (int i = 1; i < n; ++i) {
        cout << max(n - sub[i] + 1, max_sub[i]) << " ";
    }
    cout << endl;
    return 0;
}
```

Task D (60)

```
#include <iostream>
#include <string>
#include <vector>
using namespace std;

const int n = 9;
const int ALPHA = 26;

int get_int(char a) {
    return a - 'a' + 1;
}

char get_char(int a) {
    return a - 1 + 'a';
}

void f3(string &password) {
    vector <int> prefix(n, 0);
    prefix[0] = get_int(password[0]);
    for (int j = 1; j < n; ++j) {
        int val = prefix[j - 1] + get_int(password[j]);
        if (val % ALPHA == 0) {
            prefix[j] = 26;
        } else {
            prefix[j] = val % ALPHA;
        }
    }

    for (int j = 0; j < 3; ++j) {
        cout << get_char(j + 1);
        for (int k = 0; k < n; ++k) {
            if (k / 3 != j) {
                cout << get_char(prefix[k]);
            }
        }
        cout << "␣";
    }
    cout << endl;
}

void f5(string &password) {
    vector <int> prefix(n, 0);
    prefix[0] = get_int(password[0]);
    for (int j = 1; j < n; ++j) {
        int val = prefix[j - 1] + get_int(password[j]);
        if (val % ALPHA == 0) {
            prefix[j] = 26;
        } else {
            prefix[j] = val % ALPHA;
        }
    }

    for (int j = 0; j < 3; ++j) {
        cout << get_char(j + 1);
        for (int k = 0; k < n; ++k) {
            if (k / 3 != j) {
                cout << get_char(prefix[k]);
            }
        }
        cout << "␣";
    }
    for (int j = 0; j < 2; ++j) {
        cout << get_char(j + 1);
        for (int k = 0; k < n; ++k) {
            if (k / 3 != j) {
                cout << get_char(prefix[k]);
            }
        }
        cout << "␣";
```

```
    }
    cout << endl;
}

void r3(vector <string> &p) {
    vector <int> prefix(n, 0);

    for (int i = 0; i < 2; ++i) {
        int pp = 1;
        for (int j = 0; j < n; ++j) {
            if (j / 3 != get_int(p[i][0]) - 1) {
                prefix[j] = get_int(p[i][pp++]);
            }
        }
    }

    cout << get_char(prefix[0]);
    for (int i = 1; i < n; ++i) {
        if (prefix[i] - prefix[i - 1] < 0) {
            cout << get_char(prefix[i] - prefix[i - 1] + ALPHA);
        } else if (prefix[i] - prefix[i - 1] == 0) {
            cout << 'z';
        } else {
            cout << get_char(prefix[i] - prefix[i - 1]);
        }
    }
    cout << endl;
}

void r5(vector <string> &p) {
    vector <int> prefix(n, 0);

    for (int i = 0; i < 3; ++i) {
        int pp = 1;
        for (int j = 0; j < n; ++j) {
            if (j / 3 != get_int(p[i][0]) - 1) {
                prefix[j] = get_int(p[i][pp++]);
            }
        }
    }

    cout << get_char(prefix[0]);
    for (int i = 1; i < n; ++i) {
        if (prefix[i] - prefix[i - 1] < 0) {
            cout << get_char(prefix[i] - prefix[i - 1] + ALPHA);
        } else if (prefix[i] - prefix[i - 1] == 0) {
            cout << 'z';
        } else {
            cout << get_char(prefix[i] - prefix[i - 1]);
        }
    }
    cout << endl;
}

int main() {
    string mode;
    int t, m, p;

    cin >> mode >> t >> m >> p;
    for (int i = 0; i < t; ++i) {
        if (mode == "split") {
            string password;
            cin >> password;

            if (m == 3) {
                f3(password);
            } else if (m == 5) {
                f5(password);
            }
        } else {
            vector <string> p(m / 2 + 1);
            for (int j = 0; j < m / 2 + 1; ++j) {
                cin >> p[j];
            }
```

```
            if (m == 3) {
                r3(p);
            } else if (m == 5) {
                r5(p);
            }
        }
    }


    return 0;
}
```

Task E (21)

```cpp
#include <iostream>
#include <vector>
#include <cmath>
using namespace std;

typedef long long ll;

const double EPS = 1e-8;
const ll INF = 1e6;

struct Point {
    ll x, y;

    Point() {}
    Point(ll x, ll y): x(x), y(y) {}

    double dist(Point other) {
        return hypot(fabs(x - other.x), fabs(y - other.y));
    }
};

int main() {
    int n;
    vector <Point> t;
    Point p, q;

    cin >> n;

    t.resize(n);
    for (int i = 0; i < n; ++i) {
        cin >> t[i].x >> t[i].y;
    }
    cin >> p.x >> p.y;
    cin >> q.x >> q.y;

    Point l(p.x + 10000 * (q.x - p.x), p.y + 10000 * (q.y - p.y));
//  cout << t[0].dist(l) << " " << t[1].dist(l) << endl;
    int min_town = 0;
    double min_dist = 1e18;
    for (int i = 0; i < n; ++i) {
        if (fabs(min_dist - t[i].dist(l)) < EPS) {
            cout << -1 << endl;
            return 0;
        }

        if (t[i].dist(l) < min_dist) {
            min_dist = t[i].dist(l);
            min_town = i;
        }
    }

    cout << min_town + 1 << endl;
    return 0;
}
```

Task F (7)

```cpp
#include <iostream>
#include <vector>
using namespace std;

int pow(int a, int b) {
    int res = 1;
    for (int i = 1; i <= b; ++i) {
        res *= a;
    }
    return res;
}

int get_bit(int a, int i) {
    return (a >> i) & 1;
}

int main() {
    int n, k;
    vector <pair <int, int>> rb;

    cin >> n >> k;
    rb.resize(n);
    for (int i = 0; i < n; ++i) {
        cin >> rb[i].first >> rb[i].second;
    }

    int max_happy = 0;
    for (int x = 0; x < pow(2, n); ++x) {
        int cur_happy = 0;
        int cnt[2] = {0, 0};

        for (int i = n - 1, j = 0; i >= 0; --i, ++j) {
            cnt[get_bit(x, i)] += k;

            cur_happy += min(cnt[0], rb[j].first);
            cnt[0] -= min(cnt[0], rb[j].first);
            cur_happy += min(cnt[1], rb[j].second);
            cnt[1] -= min(cnt[1], rb[j].second);
        }

        max_happy = max(max_happy, cur_happy);
    }

    cout << max_happy << endl;
    return 0;
}
```