МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. ЛОМОНОСОВА

ОЛИМПИАДНАЯ РАБОТА

Наименование олимпиады школьников: **«Ломоносов»**

Профиль олимпиады: **Информатика**

ФИО участника олимпиады: **Трохачев Андрей Вадимович**

Класс: **11 класс**

Технический балл: **73**

Дата проведения: **17 марта 2022 г.**

Результаты проверки:

Оценка участника строится из 3 частей:

1. оценка за задание - рассчитывается путем запуска тестов и определения правильности работы программы на тестах, до 100 баллов по каждой задаче;
2. дополнительные баллы за полностью правильное решение задания со 2 по 5 - в случае прохождения всех тестов по заданию к оценке прибавляется 55 баллов;
3. нормализация оценки - если полученная из пунктов 1 и 2 сумма баллов превышает 500, то итоговая оценка - 100, если не превышает 500, но превышает 400 - 99 баллов, если не превышает 400 - делится на 3.9 и округляется до целого.

Оценки за задания:

№	1	2	3	4	5
Оценка	100	94	18	72	0

Дополнительный балл: 0

Задание 1. Попытка 1.

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

#define int long long


using namespace std;


/*#pragma GCC optimization("03")

#pragma GCC optimizatiion("unroll-loops")

#pragma GCC target("avx2")*/


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

#include <ext/pb_ds/tree_policy.hpp>

using namespace __gnu_pbds;

#define ordered_set tree<pair<int, int>, null_type,less<int>,
rb_tree_tag,tree_order_statistics_node_update>*/


/*struct Point {

   double x, y;

};


double dist(Point a, Point b) {

   double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;

   return sqrt((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1));

}
```

```
double sk(Point a, Point b) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    return x1 * x2 + y1 * y2;
}

void get_cords(Point a, Point b, double &a1, double &b1, double &c1) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    a1 = y1 - y2;
    b1 = x2 - x1;
    c1 = x1 * y2 - x2 * y1;
}

void get_per(Point a, double a1, double b1, double c1, double &a2, double &b2, double
&c2) {
    double x1 = a.x, y1 = a.y;
    a2 = -b1;
    b2 = a1;
    c2 = b1 * x1 - a1 * y1;
}

bool is_pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    if (a1 * b2 - a2 * b1 == 0) return 0;
    else return 1;
}
```

```
Point pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    double x4 = -1 * (c1 * b2 - c2 * b1) / (a1 * b2 - a2 * b1), y4 = -1 * (a1 * c2 - a2 * c1) / (a1 * b2 - a2 * b1);
    Point z;
    z.x = x4;
    z.y = y4;
    return z;
}

double get_dist_line(Point a, double a1, double b1, double c1) {
    double x1 = a.x, y1 = a.y;
    return abs(a1 * x1 + b1 * y1 + c1) / sqrt(a1 * a1 + b1 * b1);
}

Point get_vector(Point a, Point b) {
    Point c;
    c.x = b.x - a.x;
    c.y = b.y - a.y;
    return c;
}

Point get_napr_vect(double a1, double b1, double c1) {
    Point a;
    a.x = 1;
```

```
    a.y = ((-1 * a1 - c1) / b1) - (-c1 / b1);
    return a;
}

double vec(Point a, Point b) {
    return a.x * b.y - a.y * b.x;
}*/

string get_s() {
    string s;
    cin >> s;
    int n = s.size();
    vector <char> a;
    for (int i = n - 1; i >= 0; --i) a.push_back(s[i]);
    while (a.size() >= 2 && a.back() == '0') a.pop_back();
    string s1 = "";
    //cout << 1 << endl;
    //cout << a.size() << endl;
    for (auto i : a) s1 += i;
    reverse(s1.begin(), s1.end());
    return s1;
}

bool is_ok(string s, int k) {
    --k;
```

```
    int n = s.size();
    for (int i = n - 1; i >= 0; --i) {
        if (n - i > k) break;
        if (s[i] != '0') return 0;
    }
    return 1;
}

int gett(char c) {
    if (c >= '0' && c <= '9') return (c - '0');
    if (c >= 'a' && c <= 'z') return (c - 'a' + 10);
    return (c - 'A' + 36);
}


bool lol(string a, string ane) {
    if (a.size() > ane.size()) return 1;
    if (a.size() < ane.size()) return 0;
    int n = a.size();
    for (int i = 0; i < n; ++i) {
        int fi = gett(a[i]), si = gett(ane[i]);
        if (fi > si) return 1;
        if (fi < si) return 0;
    }
    return 1;
```

```
}



signed main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int n, k;
    cin >> k >> n;
    string a[n], ane = "";
    int fl = 0;
    for (int i = 0; i < n; ++i) a[i] = get_s();
    for (int i = 0; i < n; ++i) {
        if (is_ok(a[i], k)) {
            fl = 1;
            if (lol(a[i], ane)) ane = a[i];
        }
    }
    if (!fl) cout << -1;
    else {
        cout << ane << "\n";
        for (int i = 0; i < n; ++i) {
            if (a[i] == ane) cout << i + 1 << "\n";
        }
    }
```

```
    return 0;
}
```

Задание 2. Попытка 1.

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

#define int long long


using namespace std;


/*#pragma GCC optimization("03")

#pragma GCC optimizatiion("unroll-loops")

#pragma GCC target("avx2")*/


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

#include <ext/pb_ds/tree_policy.hpp>

using namespace __gnu_pbds;

#define ordered_set tree<pair<int, int>, null_type,less<int>,
rb_tree_tag,tree_order_statistics_node_update>*/


/*struct Point {

   double x, y;

};


double dist(Point a, Point b) {

   double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;

   return sqrt((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1));

}
```

```
double sk(Point a, Point b) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    return x1 * x2 + y1 * y2;
}

void get_cords(Point a, Point b, double &a1, double &b1, double &c1) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    a1 = y1 - y2;
    b1 = x2 - x1;
    c1 = x1 * y2 - x2 * y1;
}

void get_per(Point a, double a1, double b1, double c1, double &a2, double &b2, double
&c2) {
    double x1 = a.x, y1 = a.y;
    a2 = -b1;
    b2 = a1;
    c2 = b1 * x1 - a1 * y1;
}

bool is_pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    if (a1 * b2 - a2 * b1 == 0) return 0;
    else return 1;
}
```

```
Point pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    double x4 = -1 * (c1 * b2 - c2 * b1) / (a1 * b2 - a2 * b1), y4 = -1 * (a1 * c2 - a2 * c1) / (a1 * b2 - a2 * b1);
    Point z;
    z.x = x4;
    z.y = y4;
    return z;
}

double get_dist_line(Point a, double a1, double b1, double c1) {
    double x1 = a.x, y1 = a.y;
    return abs(a1 * x1 + b1 * y1 + c1) / sqrt(a1 * a1 + b1 * b1);
}

Point get_vector(Point a, Point b) {
    Point c;
    c.x = b.x - a.x;
    c.y = b.y - a.y;
    return c;
}

Point get_napr_vect(double a1, double b1, double c1) {
    Point a;
    a.x = 1;
```

```
    a.y = ((-1 * a1 - c1) / b1) - (-c1 / b1);
    return a;
}

double vec(Point a, Point b) {
    return a.x * b.y - a.y * b.x;
}*/

int gett(char c) {
    if (c >= '0' && c <= '9') return (c - '0');
    if (c >= 'a' && c <= 'z') return (c - 'a' + 10);
    if (c >= 'A' && c <= 'Z') return (c - 'A' + 36);
    return -1;
}

string my_get(vector <pair <int, char> > kek) {
    int last = 0;
    for (int i = 0; i < kek.size(); ++i) {
        int pos = kek.size() - i;
        if (kek[i].first > pos) last = i + 1;
    }
    string ane = "";
    for (int i = last; i < kek.size(); ++i) ane += kek[i].second;
    return ane;
}
```

```
signed main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int n;
    cin >> n;
    string a, ane = "";
    vector <pair <int, char> > kek;
    cin >> a;
    for (int i = 0; i < n; ++i) {
        if (gett(a[i]) == -1) continue;
        kek.push_back({gett(a[i]), a[i]});
    }
    sort(kek.begin(), kek.end());
    reverse(kek.begin(), kek.end());
    ane = my_get(kek);
    if (ane.size() == 0) cout << -1;
    else cout << ane;
    return 0;
}
```

Задание 2. Попытка 2.

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

#define int long long


using namespace std;


/*#pragma GCC optimization("03")

#pragma GCC optimizatiion("unroll-loops")

#pragma GCC target("avx2")*/


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

#include <ext/pb_ds/tree_policy.hpp>

using namespace __gnu_pbds;

#define ordered_set tree<pair<int, int>, null_type,less<int>,
rb_tree_tag,tree_order_statistics_node_update>*/


/*struct Point {

   double x, y;

};


double dist(Point a, Point b) {

   double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;

   return sqrt((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1));

}
```

```
double sk(Point a, Point b) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    return x1 * x2 + y1 * y2;
}

void get_cords(Point a, Point b, double &a1, double &b1, double &c1) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    a1 = y1 - y2;
    b1 = x2 - x1;
    c1 = x1 * y2 - x2 * y1;
}

void get_per(Point a, double a1, double b1, double c1, double &a2, double &b2, double
&c2) {
    double x1 = a.x, y1 = a.y;
    a2 = -b1;
    b2 = a1;
    c2 = b1 * x1 - a1 * y1;
}

bool is_pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    if (a1 * b2 - a2 * b1 == 0) return 0;
    else return 1;
}
```

```
Point pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    double x4 = -1 * (c1 * b2 - c2 * b1) / (a1 * b2 - a2 * b1), y4 = -1 * (a1 * c2 - a2 * c1) / (a1 * b2 - a2 * b1);
    Point z;
    z.x = x4;
    z.y = y4;
    return z;
}

double get_dist_line(Point a, double a1, double b1, double c1) {
    double x1 = a.x, y1 = a.y;
    return abs(a1 * x1 + b1 * y1 + c1) / sqrt(a1 * a1 + b1 * b1);
}

Point get_vector(Point a, Point b) {
    Point c;
    c.x = b.x - a.x;
    c.y = b.y - a.y;
    return c;
}

Point get_napr_vect(double a1, double b1, double c1) {
    Point a;
    a.x = 1;
```

```
    a.y = ((-1 * a1 - c1) / b1) - (-c1 / b1);
    return a;
}

double vec(Point a, Point b) {
    return a.x * b.y - a.y * b.x;
}*/

vector <pair <int, char> > kek;

int gett(char c) {
    if (c >= '0' && c <= '9') return (c - '0');
    if (c >= 'a' && c <= 'z') return (c - 'a' + 10);
    if (c >= 'A' && c <= 'Z') return (c - 'A' + 36);
    return -1;
}

bool lol(string a, string ane) {
    if (a.size() > ane.size()) return 1;
    if (a.size() < ane.size()) return 0;
    int n = a.size();
    for (int i = 0; i < n; ++i) {
        int fi = gett(a[i]), si = gett(ane[i]);
        if (fi > si) return 1;
        if (fi < si) return 0;
```

```
    }
    return 1;
}

string my_try(int v) {
    string ane = "";
    string leeel = "";
    int nowe = 0;
    for (int i = 0; i < v; ++i) {
        int pos = v - i;
        while (nowe < kek.size() && kek[nowe].first > pos) ++nowe;
        if (nowe >= kek.size()) return leeel;
        ane += kek[nowe].second;
        ++nowe;
    }
    return ane;
}



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

```
    string a, ane = "";
    cin >> a;
    for (int i = 0; i < n; ++i) {
        if (gett(a[i]) == -1) continue;
        kek.push_back({gett(a[i]), a[i]});
    }
    sort(kek.begin(), kek.end());
    reverse(kek.begin(), kek.end());
    for (int i = 61; i >= 1; --i) {
        string s1 = my_try(i);
        if (lol(s1, ane)) ane = s1;
    }
    if (ane.size() == 0) cout << -1;
    else cout << ane;
    return 0;
}
```

Задание 3. Попытка 1.

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

#define int long long


using namespace std;


/*#pragma GCC optimization("03")

#pragma GCC optimizatiion("unroll-loops")

#pragma GCC target("avx2")*/


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

#include <ext/pb_ds/tree_policy.hpp>

using namespace __gnu_pbds;

#define ordered_set tree<pair<int, int>, null_type,less<int>,
rb_tree_tag,tree_order_statistics_node_update>*/


/*struct Point {

   double x, y;

};


double dist(Point a, Point b) {

   double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;

   return sqrt((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1));

}
```

```
double sk(Point a, Point b) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    return x1 * x2 + y1 * y2;
}

void get_cords(Point a, Point b, double &a1, double &b1, double &c1) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    a1 = y1 - y2;
    b1 = x2 - x1;
    c1 = x1 * y2 - x2 * y1;
}

void get_per(Point a, double a1, double b1, double c1, double &a2, double &b2, double
&c2) {
    double x1 = a.x, y1 = a.y;
    a2 = -b1;
    b2 = a1;
    c2 = b1 * x1 - a1 * y1;
}

bool is_pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    if (a1 * b2 - a2 * b1 == 0) return 0;
    else return 1;
}
```

```
Point pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    double x4 = -1 * (c1 * b2 - c2 * b1) / (a1 * b2 - a2 * b1), y4 = -1 * (a1 * c2 - a2 * c1) / (a1 * b2 - a2 * b1);
    Point z;
    z.x = x4;
    z.y = y4;
    return z;
}

double get_dist_line(Point a, double a1, double b1, double c1) {
    double x1 = a.x, y1 = a.y;
    return abs(a1 * x1 + b1 * y1 + c1) / sqrt(a1 * a1 + b1 * b1);
}

Point get_vector(Point a, Point b) {
    Point c;
    c.x = b.x - a.x;
    c.y = b.y - a.y;
    return c;
}

Point get_napr_vect(double a1, double b1, double c1) {
    Point a;
    a.x = 1;
```

```
    a.y = ((-1 * a1 - c1) / b1) - (-c1 / b1);
    return a;
}

double vec(Point a, Point b) {
    return a.x * b.y - a.y * b.x;
}*/



signed main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int n;
    cin >> n;
    string s;
    cin >> s;
    vector <int> cnt1(n + 5, 0), cnt2(n + 5, 0);
    vector <int> to1(n + 5, 0), to2(n + 5, 0);
    vector <int> dp1(n + 5, 0), dp2(n + 5, 0);
    vector <int> fl1(n + 5, 0), fl2(n + 5, 0);
    int lel = 0;
    while (1) {
        string fi;
        cin >> fi;
```

```
    if (fi == "END") break;
    ++lel;
    int fii = stoi(fi), sii;
    cin >> sii;
    ++cnt1[fii];
    ++cnt2[sii];
    to1[fii] = sii;
    to2[sii] = fii;
}
for (int i = 1; i <= n; ++i) {
    if (cnt1[i] > 1 || cnt2[i] > 1) {
        cout << 0;
        return 0;
    }
}
if (lel == 1) {
    cout << 1;
    return 0;
}
int fle = 1;
for (int i = 1; i <= n; ++i) {
    if (to1[i] != 0 || to2[i] != 0) {
        if (fle) {
            ++dp2[to1[i]];
            ++dp1[to2[i]];
```

```
            fle = 0;
        }
    }
    if (to1[i] == i) {
        dp1[i] += dp2[i - 1];
        dp2[i] += dp1[i - 1];
    }
    if (to1[i] == i - 1) {
        dp1[i] += dp2[i - 1];
    }
    if (to2[i] == i - 1) {
        dp2[i] += dp1[i - 1];
    }
  }
  for (int i = n; i >= 1; --i) {
    if (to1[i] != 0 || to2[i] != 0) {
        int ans = dp1[i] + dp2[i];
        if (to1[i] == i && to1[i - 1] == i - 1) ++ans;
        if (to1[i] == i - 1 && to1[i - 1] == i) ++ans;
        cout << ans;
        return 0;
    }
  }
  return 0;
}
```

Задание 4. Попытка 1.

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

#define int long long


using namespace std;


/*#pragma GCC optimization("03")

#pragma GCC optimizatiion("unroll-loops")

#pragma GCC target("avx2")*/


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

#include <ext/pb_ds/tree_policy.hpp>

using namespace __gnu_pbds;

#define ordered_set tree<pair<int, int>, null_type,less<int>,
rb_tree_tag,tree_order_statistics_node_update>*/


/*struct Point {

   double x, y;

};


double dist(Point a, Point b) {

   double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;

   return sqrt((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1));

}
```

```
double sk(Point a, Point b) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    return x1 * x2 + y1 * y2;
}

void get_cords(Point a, Point b, double &a1, double &b1, double &c1) {
    double x1 = a.x, x2 = b.x, y1 = a.y, y2 = b.y;
    a1 = y1 - y2;
    b1 = x2 - x1;
    c1 = x1 * y2 - x2 * y1;
}

void get_per(Point a, double a1, double b1, double c1, double &a2, double &b2, double
&c2) {
    double x1 = a.x, y1 = a.y;
    a2 = -b1;
    b2 = a1;
    c2 = b1 * x1 - a1 * y1;
}

bool is_pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    if (a1 * b2 - a2 * b1 == 0) return 0;
    else return 1;
}
```

```
Point pere(double a1, double b1, double c1, double a2, double b2, double c2) {
    double x4 = -1 * (c1 * b2 - c2 * b1) / (a1 * b2 - a2 * b1), y4 = -1 * (a1 * c2 - a2 * c1) / (a1 * b2 - a2 * b1);
    Point z;
    z.x = x4;
    z.y = y4;
    return z;
}

double get_dist_line(Point a, double a1, double b1, double c1) {
    double x1 = a.x, y1 = a.y;
    return abs(a1 * x1 + b1 * y1 + c1) / sqrt(a1 * a1 + b1 * b1);
}

Point get_vector(Point a, Point b) {
    Point c;
    c.x = b.x - a.x;
    c.y = b.y - a.y;
    return c;
}

Point get_napr_vect(double a1, double b1, double c1) {
    Point a;
    a.x = 1;
```

```
    a.y = ((-1 * a1 - c1) / b1) - (-c1 / b1);
    return a;
}

double vec(Point a, Point b) {
    return a.x * b.y - a.y * b.x;
}*/



signed main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int n, m;
    cin >> n >> m;
    vector <vector <int> > g(n);
    for (int i = 0; i < m; ++i) {
        int a, b;
        cin >> a >> b;
        --a;
        --b;
        g[a].push_back(b);
        g[b].push_back(a);
    }
    int ve = -1, lel = 1e9;
```

```
    for (int i = 0; i < n; ++i) {
        if (g[i].size() < lel) {
            lel = g[i].size();
            ve = i;
        }
    }
    vector <pair <int, int> > ans;
    cout << g[ve].size() << "\n";
    for (auto i : g[ve]) {
        int fi = ve, si = i;
        if (fi > si) swap(fi, si);
        ++fi;
        ++si;
        ans.push_back({fi, si});
    }
    sort(ans.begin(), ans.end());
    for (auto i : ans) cout << i.first << " " << i.second << "\n";
    return 0;
}
```