Московский авиационный институт
(национальный исследовательский университет)

Институт информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Журнал по исследовательской практике (индивидуальный план)

Команда: MAI #45: Zhivalev, Kasimov, Patrikeeva
Студенты: Живалев Е.А., М8О-306Б-18
Касимов М.М., М8О-306Б-18
Патрикеева Л.В., М8О-307Б-18

Москва, 2021

Сводная таблица за весну 2021

Дата	Название	Время	Место проведения	Решенные задачи
14.02.2021	Grand Prix of Belarus	11:00-16:00	Дистанционно	N, O, P, Q
21.02.2021	Grand Prix of Suwon	11:00-16:00	Дистанционно	M, N, O, P
28.02.2021	Grand Prix of Tokyo	11:00-16:00	Дистанционно	K
14.03.2021	20-21 своя тренировка 2: Semi-FFT	11:00-16:00	Дистанционно	F, G, H, I, J
21.03.2021	20-21 своя тренировка 3: Graphs	11:00-16:00	Дистанционно	C, F, J, K
11.04.2021	20-21 своя тренировка 4	11:00-16:00	Дистанционно	D, F, M
25.04.2021	RuCode Spring 2021 Championship C-E	11:00-16:00	Дистанционно	A, B
02.05.2021	Grand Prix of China	11:00-16:00	Дистанционно	N, O, P, Q, R
09.05.2021	Grand Prix of Urals	11:00-16:00	Дистанционно	A, L, N, O
23.05.2021	Grand Prix of Southern Europe	16:00-21:00	Дистанционно	Q, O

Явка на контесты

Дата	Название	Присутствующие
14.02.2021	Grand Prix of Belarus	Живалев, Касимов, Патрикеева
21.02.2021	Grand Prix of Suwon	Живалев, Касимов, Патрикеева
28.02.2021	Grand Prix of Tokyo	Живалев, Касимов, Патрикеева
14.03.2021	20-21 своя тренировка 2: Semi-FFT	Живалев, Касимов, Патрикеева
21.03.2021	20-21 своя тренировка 3: Graphs	Живалев, Касимов, Патрикеева
11.04.2021	20-21 своя тренировка 4	Живалев, Касимов, Патрикеева
25.04.2021	RuCode Spring 2021 Championship C-E	Живалев, Касимов, Патрикеева
02.05.2021	Grand Prix of China	Живалев, Касимов, Патрикеева
09.05.2021	Grand Prix of Urals	Живалев, Касимов, Патрикеева
23.05.2021	Grand Prix of Southern Europe	Живалев, Касимов, Патрикеева

Решения задач

Контест №1 - Grand Prix of Belarus

Задача N

Условие

In preparation for remodeling of your flat, you're clearing out the junk that has accumulated over the years and find, among other forgotten things, two chess sets you had received as presents. Unfortunately, neither of the sets is complete. . . In the end you don't have the heart to throw them out and you decide to see if you can assemble one full set from the two partial ones. Recall that a complete chess set consists of 32 pieces, 16 white and 16 black. There's a king, a queen, two bishops, two knights, two rooks and eight pawns in each color. In the following, the pieces will be named by color and rank, separated by a single space, for example "white king", "black pawn" etc. Which pieces from the second set have to be added to the first to get one full set?

Input

The first line of input contains two integers k_1 and k_2 ($1 \leq k_1, k_2 < 32$), the numbers of figures left in each set. The follwoing k_1 lines list the contents of the first set in arbitrary order and the following k_2 lines after that the contents of the second set.

Output

Output (in arbitrary order) $32 - k_1$ lines listing the figures that have to be moved from the second set to the first. If a complete set can't be collected, output "impossible" as the first and only line.

Решение

```cpp
#include <iostream>
#include <unordered_map>
#include <vector>
#include <algorithm>
#include <cmath>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);

    int k1, k2;

    cin >> k1 >> k2;

    unordered_map<string, int> first_set;
    unordered_map<string, int> second_set;
    unordered_map<string, int> ans;


    const unordered_map<string, int> number_of_figures = {
            {"pawn", 8},
            {"king", 1},
            {"queen", 1},
            {"bishop", 2},
            {"knight", 2},
            {"rook", 2},
    };

    string figure;

    getline(std::cin, figure);

    for(int i = 0; i < k1; ++i) {
        getline(std::cin, figure);
        ++first_set[move(figure)];
    }

    for(int i = 0; i < k2; ++i) {
        getline(std::cin, figure);
        ++second_set[move(figure)];
    }

    if(k1 + k2 < 32) {
        cout << "impossible" << '\n';
        return 0;
    }

    const vector<string> colors = {"white", "black"};

    for(const auto& fig : number_of_figures) {
        for(const auto& color : colors) {
            auto f = color + " " + fig.first;
            if(first_set[f] != fig.second) {
                if(second_set[f] < fig.second - first_set[f]) {
                    cout << "impossible" << '\n';
                    return 0;
                } else {
                    ans[f] = fig.second - first_set[f];
                }
            }
        }
    }

    for(const auto& item : ans) {
        for(int i = 0; i < item.second; ++i) {
            cout << item.first << '\n';
        }
    }

    return 0;
}
```

Задача O

Условие

Some ink was spilled on a grid paper measuring M x N cells. Each cell of the paper is now considered either pained or clean. Two painted cells belong to the same blot if there is a path from one of the cells to the other that goes through painted cells only and on each step moves from one cell to another only horizontally or vertically. Count the number of blots and the area of the largest blot (that is, the number of cells it consists of).

Input

The first input line contains the integers M and N ($1 \leq M, N \leq 10^5, 1 < M{\cdot}N \leq 10^6$). Each of the following M lines consists of N characters 0 or 1, where 0 denotes a clean and 1 a painted cell. There is at least one painted cell.

Output

The only output line should contain two integers: the number of blots and the area of the largest blot.

Решение

```
#include <iostream>
#include <map>
#include <vector>
#include <unordered_set>
#include <unordered_map>
#include <algorithm>

using namespace std;

void dfs(const vector<vector<int>>& v, pair<int,int> pos, vector<vector<bool>>&
    position, int& counter) {
    if (position[pos.first][pos.second]) {
        return;
    }
    ++counter;
    position[pos.first][pos.second] = true;
    if (pos.first - 1 >= 0 && v[pos.first - 1][pos.second] == 1) {
        dfs(v,{pos.first - 1, pos.second}, position, counter);
    }
    if (pos.first + 1 < v.size() && v[pos.first + 1][pos.second] == 1) {
        dfs(v,{pos.first + 1, pos.second}, position, counter);
    }
    if (pos.second - 1 >= 0 && v[pos.first][pos.second - 1] == 1) {
        dfs(v,{pos.first, pos.second - 1}, position, counter);
    }
    if (pos.second + 1 < v[0].size() && v[pos.first][pos.second + 1] == 1) {
        dfs(v,{pos.first, pos.second + 1}, position, counter);
    }
}



void task1() {
    vector<vector<int>> v;
    vector<vector<bool>> position;
    int m, n;
    string cur;
    cin >> m >> n;
    v.resize(m);
    position.resize(m);
    for (int i = 0; i < m; ++i) {
        v[i].resize(n);
        position[i].resize(n);
    }
    for (int i = 0; i < m; ++i) {
        cin >> cur;
        for (int j = 0; j < n; ++j) {
            v[i][j] = cur[j] - '0';
            if (cur[j] == '1') {
                position[i][j] = false;
            } else {
                position[i][j] = true;
            }
        }
    }
    int counter = 0;
    int global_max = 0;
    int cur_max = 0;
    for (int i = 0; i < m; ++i) {
        for (int j = 0; j < n; ++j) {
            if (!position[i][j]) {
                counter++;
                cur_max = 0;
                dfs(v,{i,j},position,cur_max);
                global_max = max(cur_max, global_max);
            }
        }

    }
    cout << counter << ' ' << global_max << '\n';

}

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    task1();
    return 0;
}
```

Задача P

Условие

The early 2000's. . .
Having just boasted about your extensive experience in designing small-scale home networks, you were asked to take on the job for your apartment block. Now is probably not the time to confess that this is really your first attempt.
So, the network will connect N users. Each of them connects to one of the hubs via a dedicated network cable. The hubs may also be connected to one another with similar cables. Any socket of a hub may be used by either an end user or for a connection to another hub. The final network will have to enable any pair of users to communicate with each other via one or more hubs.
Digging through your piles of old hardware, you found M hubs suitable for the purpose. The i^{th} of those hubs ($1 \leq i \leq M$) has k_i sockets for network cables.
As you have promised that each end user will only have pay for the cable to connect themselves to the nearest hub, you want to design the network using minimal number of your hubs (so you can keep the rest for future projects).
Are you up to the task?

Input

The first input line contains the integers N and M ($2 \leq N \leq 1000, 1 \leq M \leq 300$). The second line contains M integers, the values of k_i ($2 \leq k_i \leq 48$).

Output

If there is no solution, the only line of output should contain the text **Epic fail**.
Otherwise, the first line of output should contain K, the number of hubs used, and the second line K integers, the numbers of the hubs (they are numbered starting from one in the order they are given in the input).
If there are several solutions, output any one of them.

Решение

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);

    int n, m;
    cin >> n >> m;
    vector<pair<int, int>> hubs(m);
    int counter = 0;
    vector<int> ans;
    for(int i = 0; i < m; ++i) {
        cin >> hubs[i].first;
        hubs[i].second = i;
    }
    sort(hubs.rbegin(), hubs.rend());
    for(const auto& hub : hubs) {
        ans.push_back(hub.second);
        ++counter;
        if(n - hub.first < 1) {
            n = 1;
            break;
        } else {
            n -= hub.first - 2;
        }
    }
    if(n == 1) {
        cout << counter << '\n';
        for(const auto& item : ans) {
            cout << item + 1 << ' ';
        }
        cout << '\n';
    } else {
        cout << "Epic fail" << '\n';
    }
    return 0;
}
```

Задача Q

Условие

A well-known software development company has been commissioned by the Archaeological Society. One of the modules has to help the archaeologists to process data about the ruins of buildings they have found during their excavations of ancient cities. Development of this module has been assigned to Vasya. Vasya, being a seasoned programmer, at once noticed that the module would need a database to contain the descriptions of the ruins and the estimated construction times of the buildings. It would be all fine, but suddenly the manager got the genial idea that since the database describes Roman ruins, the years of construction should be stored in the Roman number system. Now Vasya is wondering how many symbols he needs to set aside for each year number field in the database. According to the functional specification, the software module must be able to handle years from A to B (inclusive). Help Vasya determine the minimal number of characters sufficient for storing any year number in the range from A to B.

Input

The only input line contains the descriptions of the years A and B, separated by the "-" sign. A description of a year consists of one to four decimal digits (the number of the year), followed by either "AD" (Anno Domini, the current era) or "BC" (Before Christ, before the current era). In both directions the years are numbered starting from 1. It is known that $753BC \leq A \leq B \leq 2012AD$.

Output

The output should consist of a single integer, the minimal number of characters that have to be reserved in the database for the year number.

Решение

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <map>

using namespace std;

int ArabicToRoman(int arabic) {
    const map<int, int> roman_numbers = {
            {1, 1},
            {4, 2},
            {5, 1},
            {9, 2},
            {10, 1},
            {40, 2},
            {50, 1},
            {90, 2},
            {100, 1},
            {400, 2},
            {500, 1},
            {900, 2},
            {1000, 1}
    };
    auto it = roman_numbers.begin();
    while(it != roman_numbers.end() && it->first <= arabic) {
        ++it;
    }
    if(it != roman_numbers.begin()) {
        --it;
    }
    if(it->first == arabic) {
        return it->second;
    } else {
        return it->second + ArabicToRoman(arabic - it->first);
    }
}

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);

    string range, first_number, second_number;
    int n1, n2;
    cin >> range;
    auto it = range.find('-');
    first_number = string(range.begin(), next(range.begin(), it));
    second_number = string(next(range.begin(), it + 1), range.end());
    n1 = stoi(first_number.substr(0, first_number.size() - 2));
    if(first_number.substr(first_number.size() - 2, 2) == "AD") {
        n1 += 753;
    } else {
        n1 = 753 - n1 + 1;
    }
    n2 = stoi(second_number.substr(0, second_number.size() - 2));
    if(second_number.substr(second_number.size() - 2, 2) == "AD") {
        n2 += 753;
    } else {
        n2 = 753 - n2 + 1;
    }
    int ans = 0;
    for(int i = n1; i <= n2; ++i) {
        ans = max(ans, ArabicToRoman(i));
    }
    cout << ans << '\n';
    return 0;
}
```

Контест №2 - Grand Prix of Suwon

Задача M

Условие

Take any four positive integers: a, b, c, d. Form four more, like this: $|a-b||b-c||c-d||d-a|$. Then, do it again with these four new numbers. And then again. And again. Eventually, all four integers will be the same. Given a, b, c and d, figure out how quickly the sequence converges.

Input

There will be no more than 1100 several test cases in the input. Each test case consists of four positive integers on a single line ($1 \leq a, b, c, d \leq 2{\cdot}10^9$) in that order. The input will end with a line with four zeros, which should not be processed.

Output

For each test case, print a single integer on its own line, indicating the number of steps until convergence.

Решение

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

long long func_mult(int i) {
    long long answ = 1;
    while (i != 0) {
        answ *= i%10;
        i /= 10;
    }
    return answ;
}

void task1() {
    int n;
    std::cin >> n;
    long long mult = 1;
    int p = log10(n);
    int last = 0;
    int cur = 0;
    mult = func_mult(n);

    for (int i = 1; i < p + 1; ++i) {
        int m = n;
        cur = pow(10, i);
        m /= cur;
        m -= 1;
        int step = 9;
        for (int j = 1; j < i; ++j) {
            step = step*10 + 9;
        }
        m = m*cur + step;
        long long res = func_mult(m);
        if (res > mult) {
            mult = res;
        }
    }

    cout << mult << endl;
}

bool compare(int a,int b, int c, int d) {
    if (a == b && b == c && c == d) {
        return true;
    }
    return false;
}

void task2() {
    int a,b,c,d;
    int _a,_b,_c,_d;
    while (true) {
        cin >> _a >> _b >> _c >> _d;
        if (_a + _b + _c + _d == 0) {
            return;
        }
        int i = 0;
        while (!compare(_a, _b, _c, _d)) {
            a = _a; b = _b; c = _c; d = _d;
            _a = abs(a - b);
            _b = abs(b - c);
            _c = abs(c - d);
            _d = abs(d - a);
            ++i;
        }
        cout << i << '\n';
    }
}

int main() {
    ios_base::sync_with_stdio(0);
    cin.tie(nullptr);
    task2();
    return 0;
}
```

Задача N

Условие

A number n is called Special if it has a pair of factors, a and b, where $a \cdot b = n$, and together, a and b have exactly the same digits, in exactly the same quantities, as n. None of the numbers n, a or b can have leading zeros.
Here are some examples: $126 = 6 \cdot 21$, $10251 = 51 \cdot 201$, $702189 = 9 \cdot 78021$ Given an integer X, find the smallest Special number which is greater than or equal to X.

Input

There will be no more than 432 test cases in the input. Each test case will consist of a single line containing a single integer $X (10 \le X \le 10^6)$. The input will end with a line with a single 0.

Output

For each test case, output a single integer on its own line, which is the smallest Special number which is greater than or equal to X.

Решение

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    set<int> s = {126, 153, 688, 1206, 1435, 1503, 11844, 11848, 12006, 12060, 12384,
        13950, 21375, 21586, 21753, 21870, 25105, 25375, 25474, 25510, 28476, 29632,
        31590, 33655, 33696, 36855, 37840, 37845,39784,41665, 42898,44676,71199,
        78975,105295,
        105723,105750,123894,129640,129775,129955,139500,139824,150435,150624,
        150826,152271, 152406,152460, 152608, 152685, 192375, 192685,
        194229,197428,226876, 227448, 229648,251050, 260338,261378,263736,
        297463,297832,304717,307183,312475, 352966,355995, 361989,375615,378418,
        378450, 381429, 390847, 392566, 397840, 399784, 404932, 404968, 416650, 449955,
         450688,451768, 456840, 719199, 794088, 798682, 809919, 809937,809964,
        815958,829696,841995,859968,899019, 936985,939658,960988,1000255
    };
    int n;
    while(cin >> n) {
        if(!n) {
            break;
        }
        cout << *s.lower_bound(n) << '\n';
    }
    return 0;
}
```

Задача O

Условие

Alex missed the figure skating olympic qualification that he wanted to attend. So now he wants to know the pairs of skaters whose dancing he missed. He had several photos from the warmup, so he chose one where all skaters are clearly visible and wrote down the coordinates of all N skaters (N is even). Then Alex determined the pairs of skaters by the following algorithm: from not yet paired skaters he chooses two closest (to each other) skaters and assumes that they dance together as a pair. Should he find several pairs of skaters with the same minimum distance between skaters, he chooses lexicographically smallest pair (Alex enumerated skaters by integers from 1 to N, skaters are ordered inside a pair, one with lower number goes first). You are asked to help Alex to determine pairs of skaters.

Input

The first line of input contains even integer $N(2 \leq N \leq 300)$. Each i-th line of the next N lines contains two integers $-x$ and y coordinates of point, representing i-th skater. All points are distinct. All coordinates are less than 10^8 by absolute value.

Output

You should output $N/2$ lines. Each line must contain numbers of skaters in the corresponding pair. The first number in a line should be less than the second. Lines must be sorted in the lexicographically ascending order.

Решение

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

struct Pair {
    long long dist;
    pair<int, int> ids;
};

struct Vertex {
    long long x, y;
};

long long dist(const Vertex& lhs, const Vertex& rhs) {
    return (lhs.x - rhs.x) * (lhs.x - rhs.x) + (lhs.y - rhs.y) * (lhs.y - rhs.y);
}

bool operator<(const Pair& lhs, const Pair& rhs) {
    if(lhs.dist != rhs.dist) {
        return lhs.dist < rhs.dist;
    }
    return lhs.ids < rhs.ids;
}

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    vector<Vertex> v;
    vector<Pair> p;
    vector<pair<int, int>> result;
    int n;
    cin >> n;
    vector<bool> used(n, false);
    long long x, y;
    for(int i = 0; i < n; ++i) {
        cin >> x >> y;
        v.push_back({x, y});
    }
    for(int i = 0; i < n; ++i) {
        for(int j = i + 1; j < n; ++j) {
            p.push_back({dist(v[i], v[j]), {i, j}});
        }
    }
    sort(p.begin(), p.end());
    int counter = 0;
    while(counter != n) {
        for(const auto& item : p) {
            if(!used[item.ids.first] && !used[item.ids.second]) {
                used[item.ids.first] = true;
                used[item.ids.second] = true;
                result.emplace_back(item.ids.first, item.ids.second);
                counter += 2;
            }
        }
    }
    sort(result.begin(), result.end());
    for(const auto& item : result) {
        cout << item.first + 1 << ' ' << item.second + 1 << '\n';
    }
    return 0;
}
```

Задача P

Условие

Rules of The Product game are simple. Master announces positive integer N. Player needs to calculate product of digits for each positive integer up to N and report the greatest product. All answers should be known beforehand to allow interactive communication during a game. The Game Master asks you to write a program which having positive integer N will find correct answer.

Input

Input file contains one integer $N (1 \leq N \leq 2 \cdot 10^9)$.

Output

Output file should contain greatest product for given N.

Решение

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

long long func_mult(int i) {
    long long answ = 1;
    while (i != 0) {
        answ *= i%10;
        i /= 10;
    }
    return answ;
}

void task1() {
    int n;
    std::cin >> n;
    long long mult = 1;
    int p = log10(n);
    int last = 0;
    int cur = 0;
    mult = func_mult(n);

    for (int i = 1; i < p + 1; ++i) {
        int m = n;
        cur = pow(10, i);
        m /= cur;
        m -= 1;
        int step = 9;
        for (int j = 1; j < i; ++j) {
            step = step*10 + 9;
        }
        m = m*cur + step;
        long long res = func_mult(m);
        if (res > mult) {
            mult = res;
        }
    }

    cout << mult << endl;
}


int main() {
    task1();
    return 0;
}
```

Контест №3 - Grand Prix of Tokyo

Задача K

Условие

There are x cities in the kingdom. King plans to connect some cities with bidirectional highways (eachhighway connects exactly two cities) such as any there will be route between any two cities on thehighways when no more than one highway is closed.Print minimum number of roads to build 998244353.

Input

The first line contains an integer $x(3 \leq x < 10^{10^6})$.

Output

Print the answer.

Решение

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

using namespace std;

const long long MOD = 998244353;

long long bin_pow(long long a, long long b) {
    long long res = 1;
    while(b) {
        if(b & 1) {
            res = (res * a) % MOD;
        }
        a = (a * a) % MOD;
        b >>= 1;
    }
    return res;
}

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    string s;
    cin >> s;
    long long ans = 0;
    for(long long i = s.size() - 1; i >= 0; --i) {
        ans = (ans % MOD + (s[i] - '0') * bin_pow(10, s.size() - 1 - i) % MOD) % MOD;
    }
    cout << ans << '\n';
    return 0;
}
```

Контест №4 - 20-21 своя тренировка 2: Semi-FFT

Задача F

Условие

You are given two strings a and b of the same length and consisting of lowercase English letters. You can pick at most one subsequence of string b and do a cyclic shift on that subsequence exactly once.

For example, if you have a string "abcdefg"and you picked the letters at indices 2, 5, and 6 as a subsequence to do a cyclic shift on them, the letter at index 2 will go to index 5, the letter at index 5 will go to index 6, the letter at index 6 will go to index 2, and the string will become "afcdbeg".

Your task is to check if it is possible to make string b equivalent to string a using at most one cyclic shift. Can you?

Input

The first line contains the integer $T (1 \leq T \leq 200)$ specifying the number of test cases.

The first line of each test case contains an integer $n (1 \leq n \leq 10^5)$ specifying the length of strings a and b. Then two lines follow, giving strings a and b, respectively. Both strings consist only of lowercase English letters.

Output

For each test case, print a single line containing "YES"(without quotes) if it is possible to make string b equivalent to string a using at most one cyclic shift. Otherwise, print

"NO"(without quotes).

Решение

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

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int t, n;
    cin >> t;
    string s1, s2;
    string s3;
    vector<size_t> v;
    for(int i = 0; i < t; ++i) {
        v.clear();
        cin >> n;
        cin >> s1 >> s2;
        s3 = s2;
        for(int j = 0; j < n; ++j) {
            if(s1[j] != s2[j]) {
                v.push_back(j);
            }
        }
        for(int j = 0; j < v.size(); ++j) {
            s3[v[(j + 1) % v.size()]] = s2[v[j]];
        }
        if(s3 == s1) {
            cout << "YES" << '\n';
        } else {
            cout << "NO" << '\n';
        }
    }
    return 0;
}
```

Задача G

Условие

Asem has a wooden fence consisting of n boards, in which n is an odd number. Asem wants to paint this fence using two colors; black and white, such that the first board will be painted black, the second board will be painted white, the third board will be painted black, and so on.Asem has black paint that can paint at most x boards and white paint that can paint at most y boards. Your task is to determine if you can paint the whole fence or not. Can you?

Input

The first line contains an integer $T(1 \leq T \leq 10^4)$ specifying the number of test cases. Each test case consists of a line containing three integers $n, x, y, (1 \leq n, x, y < 10^9)$, in which n is an odd number specifying the number of boards in the fence, x is the maximum number of boards that can be painted in black, and y is the maximum number of boards that can be painted in white.

Output

For each test case, print a single line containing "YES"(without quotes) if you can paint the whole fence. Otherwise, print "NO"(without quotes).

Решение

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

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int t, n, x, y;
    cin >> t;
    for(int i = 0; i < t; ++i) {
        cin >> n >> x >> y;
        if(n / 2 <= y && n / 2 + 1 <= x) {
            cout << "YES" << '\n';
        } else {
            cout << "NO" << '\n';
        }
    }
    return 0;
}
```

Задача H

Условие

Multiplication operation is not always easy! For example, it is hard to calculate 27 x 20 using your mind, but it is easier to find the answer using the following methods:
30 x 20 - 3 x 20. It turns out that people can calculate the multiplication of two special numbers very easily.

A number is called special if it contains exactly one non-zero digit at the beginning (i.e. the most significant digit), followed by a non-negative number of zeros. For example, 30, 7, 5000 are special numbers, while 0, 11, 8070 are not.

In this problem, you are given two numbers a and b. Your task is to calculate the multiplication of a and b (a x b), by writing the multiplication expression as a summation or subtraction of multiplication of special numbers. Can you?

Input

The first line contains an integer $T (1 \leq T \leq 10^4)$ specifying the number of test cases.

Each test case consists of a single line containing two integers a and b $(-10^9 \leq a, b \leq 10^9, a \neq 0, b \neq 0)$, as described in the problem statement above.

Output

For each test case, print a single line containing the multiplication expression of a and b as a summation or subtraction of multiplication of special numbers. All special numbers must be between -10^9 and 10^9 (inclusive). If there are multiple solutions, print any of them. It is guaranteed that an answer always exists for the given input.

The multiplication expression must be printed in exactly one line. A single term must be printed as z#x#w in which z and w are both special numbers, # represents a single space, and x represents the multiplication operation. Two consecutive terms must be printed as z#x#w#o#z#x#w in which z and w are both special numbers, # represents a single space, represents the multiplication operation, and represents either the addition operation +or the subtraction operation -. (Check the sample output for more clarification).

Решение

```cpp
#include <iostream>
#include <vector>
#include <set>
#include <cmath>

using namespace std;

vector<int> digits(int n) {
    vector<int> res;
    int counter = 0;
    while(n) {
        res.push_back((n % 10) * pow(10, counter));
        ++counter;
        n /= 10;
    }
    return res;
}

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int t, x, y;
    cin >> t;
    for(int i = 0; i < t; ++i) {
        cin >> x >> y;
        auto v1 = digits(x);
        auto v2 = digits(y);
        for(int j = 0; j < v1.size(); ++j) {
            for(int k = 0; k < v2.size(); ++k) {
                if(v1[j] && v2[k]) {
                    cout << v1[j] << " x " << v2[k];
                    if(!(j == v1.size() - 1 && k == v2.size() - 1)) {
                        cout << " + ";
                    }
                }
            }
        }
        cout << '\n';
    }
    return 0;
}
```

Задача I

Условие

You are given two strings a and b consisting of lowercase English letters. A beautiful substring is defined as a substring of any length of string b such that the first and last letters of it are the same as the first and last letters of any substring of length k of string a. Your task is to count the number of beautiful substrings. Can you?

Input

The first line contains an integer$T (1 \leq T \leq 100)$ specifying the number of test cases.

The first line of each test case contains three integers n, m, and k $(1 \leq n, m \leq 10^5, 1 \leq k \leq n)$, in which n is the length of string a, m is the length of string b, and k is the described variable in the statement.

Then two lines follow, the first line contains a string a of length n and the second line contains a string b of length m. Both strings consist only of lowercase English letters.

Output

For each test case, print a single line containing the number of beautiful substrings

Решение

```
#include <iostream>
#include <vector>
```

```cpp
#include <unordered_map>
#include <string>
#include <unordered_set>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int t, n, m, k;
    string s1, s2;
    cin >> t;
    unordered_map<char, unordered_set<char>> substrs;
    unordered_map<char, int> cnts;
    long long counter;
    for(int i = 0; i < t; ++i) {
        cin >> n >> m >> k;
        cin >> s1 >> s2;
        counter = 0;
        substrs.clear();
        cnts.clear();
        for(int j = 0; j < s1.size(); ++j) {
            if(j + k - 1 < s1.size()) {
                substrs[s1[j + k - 1]].insert(s1[j]);
            } else {
                break;
            }
        }
        for(const auto& item : s2) {
            ++cnts[item];
            for(const auto& ch : substrs[item]) {
                counter += cnts[ch];
            }
        }
        cout << counter << '\n';
    }
    return 0;
}
```

Задача J

Условие

Abed is so motivated this year, his goal is qualifying for the 2019 ACM International Collegiate Programming Contest (ACM ICPC). Therefore, he always trains using Codeforces online judge. If you do not know Codeforces, the following rules can help you to understand how it works:

- Each problem has a set of n tests numbered from 1 to n. The first k tests are called sample tests. These tests are visible in the problem statement and a user can see them all

- When Abed gets a wrong answer on test x ($x \leq n$), he can see all the tests up to test number x. If Abed gets accepted, he can see all the tests.
- Each test can be either small or big. Abed can see the entire test only if it is small. If the test is big, Abed can see it partially. All the sample tests are small.

For example, let us consider a problem contains 6 tests, in which the first two tests are the sample tests. If Abed got a wrong answer on test 4, a wrong answer on test 3, and a wrong answer on test 5, he can see all tests from 1 to 5. If test 3 is small and the tests from 4 to 6 are big, Abed can see the first three tests fully, but he cannot fully see the remaining tests.

Unfortunately, Abed usually gets a lot of wrong answers. A submission made by Abed is called stupid if he got a wrong answer on a small test that he can see it fully. You are given a list of submissions made be Abed, your task is to count how many stupid submission Abed has made. Can you?

Input

The first line contains an integer T ($1 \leq T \leq 100$) specifying the number of test cases.

The first line of each test case contains three integers n, m, and k ($1 \leq k \leq n \leq 10^4, 1 \leq m \leq 10^4$), in which n is the number of tests in the problem, m is the number of submissions Abed has made, and k is the number of sample tests. Then a line follow contains n characters $t_1, ..., t_n$ in which ti is 'S' if the ith test is small, and 'B' if it is big.

Then m lines follow, giving the list of submissions Abed has made, such that the ith line will either contain a single character 'A' if Abed got accepted on the ith submission, or contain a character 'W' and an integer x ($1 \leq x \leq n$) giving that Abed got a wrong answer on test case x.

Output

For each test case, print a single line containing the number of stupid submissions Abed has made.

Решение

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

using namespace std;

```

```cpp
int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int t, m, k, n, a;
    vector<bool> v;
    int max_test;
    int counter;
    char c;
    cin >> t;
    for(int i = 0; i < t; ++i) {
        cin >> n >> m >> k;
        v.assign(n, false);
        for(int j = 0; j < n; ++j) {
            cin >> c;
            if(c == 'S') {
                v[j] = true;
            }
        }
        max_test = k;
        counter = 0;
        for(int j = 0; j < m; ++j) {
            cin >> c;
            if(c == 'W') {
                cin >> a;
            } else {
                max_test = n;
                continue;
            }
            if(a <= max_test && v[a - 1]) {
                ++counter;
            }
            if(a > max_test) {
                max_test = a;
            }
        }
        cout << counter << '\n';
    }
    return 0;
}
```

Контест №5 - 20-21 своя тренировка 3: Graphs

Задача C

Условие

You have a tree, or an undirected connected graph with no cycles, with n vertices and n-1 edges. Vertex 1 is the root.

You define a "leaf vertex"to be a vertex on the tree, other than the root, that is adjacent to exactly one branch vertex.

You also define a "branch vertex"to be a vertex on the tree other than the root, that is adjacent to exactly two other vertices, and adjacent to at least one leaf vertex.

You define a tree to be more of a "taiga tree"the more branch vertices that it has. Given a tree, figure out how many branch vertices it has.

Input

The first line of the input file contains a single integer T — the number of lines in the text ($1 \leq T \leq 10000$). Next come T lines — the document text. It is guaranteed that the total number of characters in the text is not greater than 10 000. All characters have ASCII-codes from 32 to 126 inclusive.

Output

Print the corrected text into the output file.

Решение

```
#include <iostream>
#include <set>
#include <vector>

using namespace std;

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n, u, v;
    int counter = 0;
    cin >> n;
    vector<set<int>> graph(n);
    set<int> leaves;
    for(int i = 0; i < n - 1; ++i) {
        cin >> u >> v;
        --u;--v;
        graph[u].insert(v);
        graph[v].insert(u);
    }
    for(int i = 1; i < n; ++i) {
        if(graph[i].size() == 1) {
            leaves.insert(i);
        }
    }
    bool found;
    for(int i = 1; i < n; ++i) {
        found = false;
        if(graph[i].size() == 2) {
            for(const auto& item : graph[i]) {
                if(auto it = leaves.find(item); it != leaves.end()) {
                    found = true;
                }
            }
        }
        if(found) {
            ++counter;
        }
    }
    cout << counter << '\n';
    return 0;
}
```

Задача F

Условие

There are T processes running in the multitasking operating system ¡¡Squirrel OS¿¿. Each of the T processes has a given priority p_i , which affects how often the process is run. Since the system only has a single core in a single processor to run things, it is facing a challenge of distributing the CPU time between these processes, taking their priorites into account. The algorithm of defining which process is run at each time moment can be described in the following way. For each process, in addition to the priority p_i , there is also a counter t_i . Initially all t_i equal 0. Then every second:

1. processes with the maximum value of $p_i + t_i$ are chosen.
2. among such processes, the process with the minimum number i is chosen.
3. the chosen process i is run for one second.
4. for the chosen process i the value t_i is set to 0.
5. for all other processes, the value t_i is increased by 1.

Model the work of the operating system for T seconds and calculate for how many seconds each process was run. Assume that all calculations and switches between processes are instant, so the running time for each process in seconds is an integer.

Input

The first line contains two space-separated integers N and T — the number of processes in the operating system ($1 \leq N \leq 10^5$) and the number of seconds to be modeled ($1 \leq T \leq 10^6$). The second line contains N space-separated integers p_i — the process priorities ($0 \leq p_i \leq 10^5$).

Output

In the only line of the output file, print N space-separated integers — for how many seconds each of the processes was run.

Решение

```
#include <iostream>
#include <map>
#include <set>
#include <queue>
#include <unordered_map>
#include <vector>

using namespace std;

using Graph = unordered_map<int, vector<int>>;

void dfs(int node, const Graph& g, vector<bool>& used) {
    if(used[node]) {
        return;
    }
    used[node] = true;
    for(const auto& item : g.at(node)) {
        dfs(item, g, used);
    }
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n, m;
    cin >> n >> m;
    vector<vector<char>> v(n);
    Graph graph;
    for(auto& item : v) {
        item.resize(m);
    }
    for(int i = 0; i < n; ++i) {
        for(int j = 0; j < m; ++j) {
            cin >> v[i][j];
        }
    }
    for(int i = 0; i < n; ++i) {
        for(int j = 0; j < m; ++j) {
            if(v[i][j] == '#') {
                if(j < m - 1 && v[i][j + 1] == '#') {
                    graph[m * i + j].push_back(m * i + j + 1);
                    graph[m * i + j + 1].push_back(m * i + j);
                }
                if(i < n - 1 && v[i + 1][j] == '#') {
                    graph[m * i + j].push_back(m * (i + 1) + j);
                    graph[m * (i + 1) + j].push_back(m * i + j);
                }
            }
        }
    }
    if(graph.size() < 3) {
        cout << "-1" << '\n';
        return 0;
    }
    for(const auto& item : graph) {
        if(item.second.size() == 1) {
            cout << '1' << '\n';
            return 0;
        }
    }
    vector<bool> used(m * n);
    for(const auto& item : graph) {
        used.assign(m * n, false);
        used[item.first] = true;
        dfs(item.second[0], graph, used);
        for(const auto& xd : graph) {
            if(!used[xd.first]) {
                cout << '1' << '\n';
                return 0;
            }
        }
    }
    cout << '2' << '\n';
    return 0;
}
```

Задача J

Условие

Students of a specialized arts school must paint a picture as their graduation qualification work. In the course of their study, the students have mastered a technique of painting on a square canvas using two types of brushstroke: ¡¡asterisk¿¿ and ¡¡hash¿¿. Using this technique, the prodigies must paint a square masterpiece of the size N x N , using only the two types of strokes. This is a tedious task, and every year a few students decide they could benefit from the hard work of the previous generations of their alumni-to-be. However, their imagination is quite limited. A student takes someone else's picture and applies the following operations several times: 1) rotate the image 90 degrees, 2) mirror it along the vertical or horizontal axis. After that, he submits the result as his own picture. Some students have even tried presenting unchanged old works as their own. Their professors can feel something is wrong, but unfortunately, it is beyond their own powers to find out for sure whether the paintings are plagiarized or not. It's time to put an end to this disgraceful business and write a program that would automate the plagiarism check by defining whether a painting is a copy or not, thus helping the professors to find cheating students.

Input

The first line of the input file contains a single integer: N is the size of the side of the square canvas ($1 \leq N \leq 500$). Each of the following N lines contains N symbols * or #, denoting the types of the corresponding strokes of the first painting. Next comes an empty line. The next N lines describe the second painting in the same format.

Output

The only line of the output file is the word YES, if one of the paintings is plagiarized from another, or NO if not.

Решение

```cpp
#include <iostream>
#include <set>
#include <vector>

using namespace std;

using Graph = vector<set<int>>;

void dfs(int node, const Graph& g, vector<bool>& used, bool left, vector<vector<int
    >>& parts) {
    if(used[node]) {
        return;
    }
    used[node] = true;
    if(left) {
        parts[0].push_back(node);
    } else {
        parts[1].push_back(node);
    }
    for(const auto& neighbor : g[node]) {
        dfs(neighbor, g, used, !left, parts);
    }
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n, u, v;
    long long counter = 0;
    cin >> n;
    vector<set<int>> graph(n);
    vector<vector<int>> parts(2);
    vector<bool> used(n, false);
    for(int i = 0; i < n - 1; ++i) {
        cin >> u >> v;
        --u;--v;
        graph[u].insert(v);
        graph[v].insert(u);
    }
    dfs(0, graph, used, true, parts);
    for(const auto& item : parts[0]) {
        counter += parts[1].size() - graph[item].size();
    }
    cout << counter << '\n';
    return 0;
}
```

Задача K

Условие

You are given a set of files, and you are to output the number of files with each extension. The file extension is a sequence of characters in the name of the file after a dot character. The file system is case-sensitive: even if the file names differ only by a characters case, they are still considered to be different.

Input

In the first line of the input file there is an integer N , which is number of file names ($1 \leq N \leq 10^3$). In each of the following N lines there is a file name that is no more than 200 characters in length. The file name consists of only lower and upper Latin letters, numbers and a dot character '.'. It is guaranteed that a dot character can be found in the file name exactly once. Also, there is at least one symbol before and after the dot. It is guaranteed that each file name is present only once in the input file.

Output

For each of the extensions that are present in the input file, output the number of files with this extension in the form of <extension>: <number>. Output extensions in order of the first mention in the input file.

Решение

```
#include <iostream>
#include <map>
#include <set>
```

```cpp
#include <queue>
#include <vector>

using namespace std;

using Graph = vector<set<int>>;

long long bfs(const Graph& g) {
    vector<bool> used(g.size(), false);
    queue<pair<int, int>> q;
    map<int, int> heights;
    q.push(make_pair(0, 0));
    while(!q.empty()) {
        auto [node, height] = q.front();
        q.pop();
        if(used[node]) {
            continue;
        }
        used[node] = true;
        for(const auto& item : g[node]) {
            q.push(make_pair(item, height + 1));
        }
        ++heights[height];
    }
    long long res = 0;
    for(const auto& item : heights) {
        res += item.second % 2;
    }
    return res;
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n, u, v;
    cin >> n;
    vector<bool> used(n, false);
    set<int> heights;
    Graph g(n);
    for(int i = 0; i < n - 1; ++i) {
        cin >> u;
        --u;
        g[i + 1].insert(u);
        g[u].insert(i + 1);
    }
    cout << bfs(g) << '\n';
    return 0;
}
```

Контест №6 - 20-21 своя тренировка 4

Задача D

Условие

У Булата есть строка s длины n, которая содержит только строчные буквы английского алфавита (в нижнем регистре). Назовем последовательность букв словом, если она содержит хотя бы одну гласную и хотя бы одну согласную. На какое максимальное число слов Булат может разделить имеющуюся у него строку?

В этой задаче гласными считаются буквы a, i, o, u, e, y, а все остальные согласными.

Input

Первая строка содержит единственное целое число n ($1 \leq n \leq 10^5$), длину строки. Вторая строка содержит строку s, которая содержит только строчные буквы английского алфавита (в нижнем регистре).

Output

Выведите единственное целое число, максимальное число слов, на которое можно разбить строку. Если строку невозможно разбить нужным образом, то в этом случае выведите 0

Решение

```
#include <iostream>
#include <string>
#include <set>
#include <algorithm>

int main() {
    std::set<char> v = {'a','e','i','o','u','y'}, co= {'b', 'c', 'd', 'f', 'g', 'h'
        , 'j', 'k', 'l', 'm', 'n'
    , 'p', 'q', 'r', 's', 't', 'v', 'w', 'x', 'z'};
    std::string s;
    int n;
    int pair = 0;
    std::cin >> n;
    std::cin >> s;


    for (int i = 1; i < s.size(); ++i) {
        if (v.find(s[i]) != v.end()) {
            if (co.find(s[i - 1]) != co.end()) {
                ++pair;
                ++i;
            }
        } else if (co.find(s[i]) != co.end()) {
            if (v.find(s[i - 1]) != v.end()) {
                ++pair;
                ++i;
            }
        }
    }
    std::cout << pair << '\n';
    return 0;
}
```

Задача F

Условие

У Алсу есть полоска бумаги, которая состоит из n последовательных квадратных клеточек. Клеточки пронумерованы слева направо от 1 до n. В каждой клеточке записано одно целое число, которое может принимать значение 0, 1 или 2. Обозначим число в i-й ячейке за ai. Алсу может покрасить в черный любое количество левых или правых границ этих клеточек (в общем у ячеек n+1 границ). Можно ли покрасить границы клеток так, чтобы количество покрашенных границ у i клетки равнялось числу ai? Например, если в клетках написаны числа 1 2 1 0 1 2, то можно покрасить границы следующим образом

Input

Первая строка содержит единственное целое число n, число клеток ($1 \leq n \leq 10^5$). Следующая строка содержит n разделенных пробелами целых чисел a1, . . . , an ($0 \leq ai \leq 2$), количество покрашенных границ, которое должно быть у ячеек 1,..,n.

Output

Если необходимое возможно сделать, выведите «Yes» в единственной строки. Если невозможно, выведите «No» (ответ нужно вывести без кавычек).

Решение

```
#include <iostream>
#include <vector>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int n;
    cin >> n;
    vector<int> v(n);
    bool ans = true;
    for(int i = 0; i < n; ++i) {
        cin >> v[i];
    }
    if(n == 1) {
        cout << "YES\n";
        return 0;
    }
    vector<int> borders(n + 1, 0);
    for(int i = 0; i < n; ++i) {
        if(v[i] == 2) {
            borders[i] = 1;
            borders[i + 1] = 1;
        }
    }
    for(int i = 0; i < n; ++i) {
        if(v[i] == 1) {
            if(borders[i] == 1 || borders[i + 1] == 1) {
                continue;
            } else {
                if(i == 0) {
                    int counter = 0;
                    bool is_two = false;
                    for(int j = 1; j < n; ++j) {
                        if(v[j] != 1) {
                            if(v[j] == 2) {
                                is_two = true;
                            }
                            break;
                        }
                        ++counter;
                    }
                    if(v[i + 1] == 0 || (i + 2 < n && borders[i + 2] == 1) || (is_two
                         && counter % 2 == 1) || (!is_two && !(counter % 2))) {
                        borders[i] = 1;
                    } else {
                        borders[i + 1] = 1;
                    }
                } else if(i == n - 1) {
                    if(v[i - 1] == 0 || borders[i - 1] == 1) {
                        borders[i + 1] = 1;
                    } else {
                        borders[i] = 1;
                    }
                } else {
                    if(v[i - 1] == 1) {
                        if(borders[i - 1] == 1) {
                            borders[i + 1] = 1;
                        } else {
                            borders[i] = 1;
                        }
                    } else if(!v[i - 1]) {
                        borders[i + 1] = 1;
                    }
                }
            }
        }
    }
    for(int i = 0; i < n; ++i) {
        ans = ans && (v[i] == borders[i] + borders[i + 1]);
    }
    if(ans) {
        cout << "YES" << '\n';
    } else {
        cout << "NO" << '\n';
    }
    return 0;
}
```

Задача M

Условие

У Алсу есть бесконечная таблица A. Строки и столбцы в ней нумеруется целыми числами 0, 1, 2, и т.д. Число в i-й строке и j-м столбце равно Ai,j=2i3j. Помогите Алсу обработать q запросов вида (i1,i2,j1,j2), где $i1 \leq i2j1 \leq j2$. Для каждого запроса вам нужно вычислить значение по модулю $10^9 + 7$.

Input

Первая строка содержит единственное целое число q, количество запросов ($1 \leq q \leq 10^4$). Далее идут описания самих запросов. i-я из следующих q строк содержит четыре целых числа i1, i2, j1, j2, описание i-го запроса ($0 \leq i1 \leq i2 \leq 10^9, 0 \leq j1 \leq j2 \leq 10^9$).

Output

Для каждого из запросов в отдельной строке выведите соответствующую сумму по модулю $10^9 + 7$.

Решение

```
#include <iostream>
#include <map>
#include <set>
#include <queue>
#include <unordered_set>
#include <vector>

using namespace std;

const long long MOD = 1e9 + 7;

```

```cpp
long long bin_pow(long long a, long long b) {
    long long res = 1;
    while(b) {
        if(b & 1) {
            res = (res * a) % MOD;
        }
        a = (a * a) % MOD;
        b >>= 1;
    }
    return res;
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    long long q, i1, i2, j1, j2;
    long long i_sum, j_sum;
    cin >> q;
    for(int i = 0; i < q; ++i) {
        cin >> i1 >> i2 >> j1 >> j2;
        i_sum = (bin_pow(2, i2 + 1) - bin_pow(2, i1));
        if(i_sum < 0) {
            i_sum = MOD + i_sum;
        }
        j_sum = (bin_pow(3, j2 + 1) - bin_pow(3, j1));
        if(j_sum < 0) {
            j_sum = MOD + j_sum;
        }
        cout << (((i_sum * j_sum) % MOD) * 500000004) % MOD << '\n';
    }
    return 0;
}
```

Контест №7 - RuCode Spring 2021 Championship C-E

Задача A

Условие

Flatland is a two-dimensional plane. Points with both coordinates different from zero are called free. Points with at least one zero coordinate are called custom points; when passing through these points one should pay 1 flatland dollar as a fee. Adventurer stands at the free point with integer non-zero coordinates x 1 and y 1 and the goal of his adventure is to reach the free point with non-zero coordinates x 2 and y 2 . He is allowed to choose any route he wants. Calculate minimum possible fee to be paid by adventurer.

Input

Input contains four integers x_1, y_1, x_2 and y_2 — coordinates of the start point and the finish point respectively ($x_1 \neq 0, y_1 \neq 0, x_2 \neq 0, y_2 \neq 0, -10000 \leq x_1, y_1, x_2, y_2 \leq 10000$).

Output

Print minimum possible fee in flatland dollars to be paid by adventurer.

Решение

```cpp
#include <iostream>
#include <map>
#include <vector>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int n, l, t, r;
    map<int, int> score;
    cin >> n >> l;
    for(int i = 0; i < n; ++i) {
        cin >> t >> r;
        if(r == -1) {
            score[t] += 1;
        } else if(r < l) {
            score[t] += 2;
        } else {
            score[t] += 3;
        }
    }
    cout << score[1] << ":" << score[2] << '\n';
    return 0;
}
```

Задача B

Условие

In a modern professional sport it is common for team managers to make game-related decisions based on statistics instead of personal impression. That is even more common for games that feature repetitive actions and the same player's dispositions many times per game, and each team is expected to score tens of times. Basketball is such a game and this task is about popular plus-minus statistics for individual players. Each team has ten players for a game. At every moment of time each team has exactly five players on the court and five players in reserve. For the purpose of this problem we consider that substitutions are unlimited and can take place at any moment of time. Moreover, there are no restrictions on the number of substitutions for each particular player. At the beginning of the game the individual score of each player is 0. Every time some team scores x points (x = 1, x = 2 and x = 3 are the options), the individual score of each player of this team who is currently on the court is increased by x. For the opposing team, the individual score of each player who is on the court is reduced by x. Individual score can become negative. No score changes take place for the players of both teams who are currently in reserve. You are given the protocol of the game that has information about the players on the court at the beginning of the game, substitutions and scoring events. Your task is to compute the individual scores of the plus-minus statistics for all players that appeared on the court at least once.

Input

The first line of the input contains the name of the first team. Then follow five lines containing names of the players, who are playing for the first team at the beginning of the game. Then follow six lines containing the name of the second team and five names of the players of the second team that are on court at the beginning of the game. Then goes the description of the game protocol. The first line of this description contains a single integer q ($1 \leq q \leq 1000$) — the number of events to consider. Then follow q events listed in chronological order. Each event is of one of two types.

- Events of the first type have form “Team x scored n”, where x is one of two team names and n is an integer between 1 and 3.
- Events of the second type have form “Team x replaced y with z”, where x is one of two team names, y is the name of the player from team x who is guaranteed to be on the court at that moment of time and z is the name of the player from team x who is guaranteed to be at reserve at that moment of time.

All names are non-empty strings of no more than 80 characters. Lowercase and uppercase English letters are allowed. Team’s names are distinct, there are no two players with the same name in one team.

Output

For each player that appeared in the play (on the court) at least once print his name, team and individual plus-minus score on a separate line. List the players in the order they are mentioned in the input for the first time. First, print the name of this player, then print the name of this player’s team in brackets “()” (see sample output), finally print the score. Positive scores should be printed with ‘+’ sign and negative with ‘-’. Zero scores should have no sign at all.

Решение

```
n = int(input())
k = int(input())
res = ""
nums = []
```

```python
bad_nums = []
for num in input().split():
    bad_nums.append(int(num))
for i in range(10):
    if i not in bad_nums:
        nums.append(i)
while n > 0:
    res = str(n % (10 - k)) + res
    n //= (10 - k)
ans = ""
for i in range(len(res)):
    ans += str(nums[int(res[i])])
print(ans)
```

Контест №8 - Grand Prix of China

Задача N

Условие

A group of students is taking a True/False exam. Each question is worth one point. You, as their teacher,want to make your students look as good as possible — so you cheat! (I know, you would never actuallydo that.) To cheat, you manipulate the answer key so that the lowest score in the class is as high aspossible.What is the best possible lowest score you can achieve?

Input

The only line of input contains a single integer y ($1995 \leq y \leq 2019$), denoting the year. You don't need to process year numbers less than 1995 or greater than 2019, or incorrect year formats. It is guaranteed that you will be given a number between 1995 and 2019, inclusive.

Output

Output, on a single line, the best possible lowest score in the class.

Решение

```cpp
#include <algorithm>
#include <cmath>
#include <iostream>
#include <vector>
#include <iomanip>
#include <bitset>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int n, k;
    string s;
    cin >> n >> k;
    int max_count = 0;
    vector<bitset<10>> v(n, 0);
    for(int i = 0; i < n; ++i) {
        cin >> s;
        for(int j = 0; j < s.size(); ++j) {
            if(s[j] == 'T') {
                v[i][j] = true;
            }
        }
    }
    for(int i = 0; i <= pow(2, k); ++i) {
        bitset<10> b(i);
        int m = 11;
        for(const auto& item : v) {
            m = min(m, (int)(k - (item ^ b).count()));
        }
        max_count = max(max_count, m);
    }
    cout << max_count << '\n';
    return 0;
}
```

Задача O

Условие

You are performing a magic trick with a special deck of cards.You lay out the cards in a row from left to right, face up. Each card has a lower-case letter on it. Twocards with the same letter are indistinguishable. You select an audience member to perform an operationon the cards. You will not see what operation they perform.The audience member can do one of two thingsthey can either select any two cards and swap them, orleave the cards untouched.In order for the trick to succeed, you must correctly guess what the audience member dideither you guessthat the audience member did nothing, or you point at the two cards the audience member swapped.Given a string that represents the initial arrangement of the cards, can you guarantee that you will alwaysbe able to guess the audience member's operation correctly, no matter what operation they perform?

Input

The first four lines of the input contain integers a, x, b, and y ($0 \leq a, x, b, y \leq 100$), each on a separate line. The last line contains a single integer T ($1 \leq T \leq 1440$) — the total time spent on a bicycle during each day.

Output

Output a single line with 1 if you can guarantee that you will always be able to guess the audiencemember's operation correctly, or 0 otherwise

Решение

```cpp
#include <cmath>
#include <iostream>
#include <vector>
#include <iomanip>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    string s;
    cin >> s;
    vector<int> v(26, 0);
    for(char& c : s) {
        if(v[c - 'a'] != 0) {
            cout << 0 << '\n';
            return 0;
        }
        ++v[c - 'a'];
    }
    cout << 1 << '\n';
    return 0;
}
```

Задача P

Условие

You are teaching kindergarten! You wrote down the numbers from 1 ton, in order, on a whiteboard.When you weren't paying attention, one of your students erased one of the numbers.Can you tell which number your mischievous student erased?

Input

The first line of input contains a single integern ($2 \leq n \leq 100$), which is the number of numbers thatyou wrote down. The second line of input contains a string of digits, which represents the numbers youwrote down (minus the one that has been erased). There are no spaces in this string. It is guaranteed tocontain all of the numbers from 1 ton, in order, except for the single number that the student erased

Output

Output a single integer, which is the number that the tricky student erased.

Решение

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

int main() {
    int n;
    std::cin >> n;
    std::string s = "";
    std::vector<int> v;
    for (int i = 1; i <= n; ++i) {
        s += std::to_string(i);
        if (i >= 10) {
            v.push_back(i);
            v.push_back(i);
        } else {
            v.push_back(i);
        }
    }
    std::string tmp;
    std::cin >> tmp;
    for (int i = 0; i < s.size(); ++i) {
        if (tmp[i] != s[i] or tmp.size() == i) {
            std::cout << v[i] << '\n';
            return 0;
        }
    }
    return 0;
}
```

Задача Q

Условие

Your judges are preparing a problem set, and they're trying to evaluate a problem for inclusion in theset. Each judge rates the problem with an integer between -3 and 3.The overall rating of the problem is the average of all of the judges' ratings — that is, the sum of theratings divided by the number of judges providing a rating.Some judges have already rated the problem. Compute the minimum and maximum possible overallrating that the problem can end up with after the other judges submit their ratings.

Input

The first line of the input contains four integers n, w, h, and s — the number of characters in SupFont, width and height of each character, and the number of switches after which a pixel becomes broken ($1 \leq n \leq 94; 1 \leq w, h \leq 30; 1 \leq s \leq 10^6$). The following lines contain SupFont character descriptions. The first line of each description contains an

ASCII character (only characters with codes between 33 and 126, inclusive, are used). The character's image with h lines of length w follows, '#' denotes an on pixel, and '.' denotes an off pixel. Each image has at least one on pixel. All described ASCII characters are distinct, but some of them may have the same image.

Output

Output two space-separated floating point numbers on a single line, which are the minimum and maximumoverall rating the problem could achieve after the remaining judges rate the problem, minimum first.These values must be accurate to an absolute or relative error of 10^4.

Решение

```cpp
#include <cmath>
#include <iostream>
#include <vector>
#include <iomanip>

using namespace std;

int main() {
    double n, k, r;
    double sum = 0;
    cin >> n >> k;
    for(int i = 0; i < k; ++i) {
        cin >> r;
        sum += r;
    }
    cout << fixed << setprecision(18);
    cout << (sum + (-3.) * (n - k)) / (double)n << ' ' << (sum + (3.) * (n - k)) / (
        double)n << '\n';
    return 0;
}
```

Задача R

Условие

There is a group of people in an internet email message group. Messages are sent to all members of thegroup, and no two messages are sent at the same time.Immediately before a person sends a message, they read all their unread messages up to that point. Eachsender also reads their own message the moment it is sent. Therefore, a person's unread messages areexactly the set of messages sent after that person's last message.Each time a message is sent, compute the total number of unread messages over all group members

Input

Each of the nextmlines contains a single integers $(1 \leq s \leq n)$, which is the sender of that message.These lines are in chronological order.

Output

Output m lines, each with a single integer, indicating the total number of unread messages over all groupmembers, immediately after each message is sent

Решение

```cpp
#include <algorithm>
#include <cmath>
#include <iostream>
#include <vector>
#include <unordered_map>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    unordered_map<int, int> order;
    int n, m, sender;
    cin >> n >> m;
    int current_count = 0;
    for(int i = 0; i < m; ++i) {
        ++current_count;
        cin >> sender;
        order[sender] = current_count;
        long long result = 0;
        for(const auto& item : order) {
            result += current_count - item.second;
        }
        result += current_count * (n - order.size());
        cout << result << '\n';
    }
    return 0;
}
```

Контест №9 - Grand Prix of Urals

Задача A

Условие

The famous astrologer Pavel Globus writes a bot for trading stocks in the stock market. Pavel is goingto predict the stock priceby the stars. He analyzed historical data and noticed that, for example, whenMars was in Capricorn, the stock price fell, and when the moon was in Gemini, the quotes went up. Ofcourse, Pavel will not reveal all the details of his algorithm.Pavel is not so good at programming, and one of the parts of the program that he cannot cope with isdetermining the zodiac sign, in which the Sun is located, depending on the current date. The zodiac signfor the current date can be determined from the following table.Help Pavel and write a program that determines the zodiac sign by the current date. Pavel, in return,will help you increase your capital.

Input

You are given a string in format ¡¡YYYY-MM-DD¿¿, indicating the current date, whereYYYY— year(2021 $\leq YYYY \leq$ 2050),MM— month (01 $\leq MM \leq$ 12), and YYYY— day (01 $\leq DD \leq$ 31).The date is real.

Output

Print one of the words from the list ¡¡Aries¿¿, ¡¡Taurus¿¿, ¡¡Gemini¿¿, ¡¡Cancer¿¿, ¡¡Leo¿¿, ¡¡Virgo¿¿,¡¡Libra¿¿, ¡¡Scorpio¿¿, ¡¡Sagittarius¿¿, ¡¡Capricorn¿¿, ¡¡Aquarius¿¿, ¡¡Pisces¿¿, corresponding to thezodiac sign.

Решение

```python
date = input().split('-')
months = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', '
    September', 'October', 'November',
          'December']
d = dict(Aries=[('March', 21), ('April', 19)], Taurus=[('April', 20), ('May', 20)],
    Gemini=[('May', 21), ('June', 20)],
         Cancer=[('June', 21), ('July', 22)], Leo=[('July', 23), ('August', 22)],
         Virgo=[('August', 23), ('September', 22)], Libra=[('September', 23), ('October
             ', 22)],
         Scorpio=[('October', 23), ('November', 22)], Sagittarius=[('November', 23), ('
             December', 21)],
         Capricorn=[('December', 22), ('January', 19)], Aquarius=[('January', 20), ('
             February', 18)],
         Pisces=[('February', 19), ('March', 20)])

m = date[1]
day = int(date[2])
for k, v in d.items():
    if (months.index(v[0][0]) == int(m) - 1 and v[0][1] <= day) or (
            int(m) - 1 == months.index(v[1][0]) and day <= v[1][1]):
        print(k)
        break
```

Задача O

Условие

There are N cities in Byteland, cities are connected by N - 1 bidirectional roads such as each city can be reached from the another one using one of several those roads. A city is called autonomous, if it have only one road leading to (and from) this city. Given N , find the least possible number of autonomous cities.

Input

Input consists of one integer N ($2 \le N \le 10^9$) — number of cities in Byteland.

Output

Print one integer — least possible number of autonomous cities.

Решение

```cpp
#include <algorithm>
#include <cmath>
#include <iostream>
#include <vector>
#include <unordered_set>
#include <string>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    string s;
    cin >> s;
    const unordered_set<char> vowels = {'a', 'e', 'i', 'o', 'u'};
    int vowel_count = 0;
    int cons_count = 0;
    for(auto& item : s) {
        if(auto it = vowels.find(item); it != vowels.end()) {
            ++vowel_count;
        } else if(item != 'y') {
            ++cons_count;
        }
    }
    if(cons_count >= vowel_count) {
        int count = 0;
        while(cons_count >= vowel_count) {
            ++count;
            --cons_count;
            ++vowel_count;
        }
        cout << count << '\n';
    } else {
        cout << 0 << '\n';
    }
    return 0;
}
```

Задача L

Условие

Let's define customity of two adjacent elements a i and a i+1 of the array A as $|a_i - a_{i+1}|$. Given an integer array A. For one operation you may add to any element of the sequence arbitrary real number. Your task is to convert the array to one where maximum customity will be minimized. Calculate minimal number of operations needed to do that.

Input

First line of the input contains one integer N — length of the given array ($2 \leq N \leq 10^5$). Second line contains N integers $a_i(-10^6 \leq a_i \leq 10^6)$ — elements of the given array A.

Output

Print one integer — minimal number of operations needed to obtain the array where maximum customity is minimized.

Решение

```cpp
#include <algorithm>
#include <stack>
#include <iostream>
#include <vector>
#include <unordered_map>
#include <string>
#include <cmath>

using namespace std;

int main(){

int a, b;
cin >> a >> b;
int c = 180 - a - b;
if ( c < 0 ) cout << "0";
else cout << ceil((float)c/(float)a);

return 0;
}
```

Задача N

Условие

Consider the plane with usual Cartesian coordinate system with center O (0, 0). Let's introduce polar coordinate system at the same plane. A point O is chosen as the pole and a positive ray of Ox is taken as the polar axis. For a given angle α, there is a single line through the pole whose angle with the polar axis is α (measured counterclockwise from the axis to the line). Then there is a unique point on this line whose signed distance from the origin is r for given number r. For a given pair of coordinates (r, α) there is a single point, but any point is represented by many pairs of coordinates. For example, (r, α), $(r, \alpha + 2\pi)$ and $(-r, \alpha + \pi)$ are all polar coordinates for the same point. The pole is represented by $(0, \alpha)$ for any value of α. We will call the point interesting, if same pair of numbers is representing its Cartesian coordinates (x, y) and its polar coordinates (r, α). For given integer N count number of beautuful points (x, y) with the following property: x + y is integer and $x + y \leq N$.

Input

First line of the input contains one integer N — upper limit for x + y ($0 \leq N \leq 10^9$).

Output

If there are infinite number of such a points, print -1. Otherwise print one integer — answer to the problem.

Решение

```python
a, b = input().split()
a = int(a)
b = int(b)

if a == 1:
    print(10 ** b - 10 ** (a - 1) + 1)
else:
    print(10 ** b - 10 ** (a - 1))
```

Контест №10 - Grand Prix of Southern Europe

Задача Q

Условие

Alex missed the ballroom dance competition that he wanted to attend. So now he wants to know thepairs of dancers whose dancing he missed. He had several photos from the competition, so he chose onewhere all dancers are clearly visible and wrote down the coordinates of all N dancers (N is even). ThenAlex determined the pairs of dancers by the following algorithm: from not yet paired dancers he choosestwo closest (to each other) dancers and assumes that they dance together as a pair. Should he find severalpairs of dancers with the same minimum distance between dancers, he chooses lexicographically smallestpair (Alex enumerated dancers by integer numbers from 1 to N, dancers are ordered inside a pair, onewith lower number goes first). You are asked to help Alex to determine pairs of dancers

Input

The first line of input contains even integer N ($2 \leq N \leq 300$). Each i-th line of the next N linescontains two integers -x and y coordinates of point, representing i-th dancer. All points are distinct. All coordinates are less than 10^8 by absolute value

Output

You should output N/2 lines. Each line must contain numbers of dancers in the corresponding pair.The first number in a line should be less than the second. Lines must be sorted in the lexicographicallyascending order.

Решение

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <map>


using namespace std;

struct Pair {
    long long dist;
    pair<int, int> ids;
};

struct Vertex {
    long long x, y;
};

long long dist(const Vertex& lhs, const Vertex& rhs) {
    return (lhs.x - rhs.x) * (lhs.x - rhs.x) + (lhs.y - rhs.y) * (lhs.y - rhs.y);
}

bool operator<(const Pair& lhs, const Pair& rhs) {
    if(lhs.dist != rhs.dist) {
        return lhs.dist < rhs.dist;
    }
    return lhs.ids < rhs.ids;
}

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    vector<Vertex> v;
    vector<Pair> p;
    vector<pair<int, int>> result;
    int n;
    cin >> n;
    vector<bool> used(n, false);
    long long x, y;
    for(int i = 0; i < n; ++i) {
        cin >> x >> y;
        v.push_back({x, y});
    }
    for(int i = 0; i < n; ++i) {
        for(int j = i + 1; j < n; ++j) {
            p.push_back({dist(v[i], v[j]), {i, j}});
        }
    }
    sort(p.begin(), p.end());
    int counter = 0;
    while(counter != n) {
        for(const auto& item : p) {
            if(!used[item.ids.first] && !used[item.ids.second]) {
                used[item.ids.first] = true;
                used[item.ids.second] = true;
                result.emplace_back(item.ids.first, item.ids.second);
                counter += 2;
            }
        }
    }
    sort(result.begin(), result.end());
    for(const auto& item : result) {
        cout << item.first + 1 << ' ' << item.second + 1 << '\n';
    }
    return 0;
}
```

Задача O

Условие

Secret laboratory of Fatown has developed a new gluttonous robot which moves on stripe consisting of n + 1 cells. Cells are numbered from 0 to n. The robot is located at cell with number 0; each othercell contains several bukazoids which gluttonous robot regales oneself with. The robot can do m single jumps (to adjacent cell) and k double jumps (over one cell). Additionally,m + 2k = n. All jumps are jumps forward. To feed gluttonous robot you need to write a program which finds sequence of jumpswith highest number of bukazoids on a way.

Input

The first line at the input contains 3 integers:$n(1 \leq n \leq 100), m(0 \leq m \leq 100), k(0 \leq k \leq 100)$. Thesecond line contains n integers — number of bukazoids (up to 100) in corresponding cells of the stripe.

Output

The first line at the output should contain highest number of bukazoids found. The second line shouldcontain m+k+1 integers — numbers of cells visited by the robot, starting from cell with number 0

Решение

```
def pprint(l):
    for i in range(len(l)):
        print(l[i])


n, m, k = input().split()
n = int(n)
m = int(m)
k = int(k)
begin = [int(_) for _ in input().split()]
dp = [[0 for _ in range(k + 1)] for _ in range(m + 1)]
path = [[(0, 0) for _ in range(k + 1)] for _ in range(m + 1)]
for i in range(0, m + 1):
    for j in range(0, k + 1):
        if i == 0 and j == 0:
            continue
        if j > 0 and i > 0:
            dp[i][j] += begin[i + j * 2 - 1]
            if dp[i][j - 1] >= dp[i - 1][j]:
                path[i][j] = (i, j - 1)
                dp[i][j] += dp[i][j - 1]
            else:
                path[i][j] = (i - 1, j)
                dp[i][j] += dp[i - 1][j]
        elif j > 0:
            dp[i][j] += dp[i][j - 1] + begin[i + j * 2 - 1]
            path[i][j] = (i, j - 1)
        elif i > 0:
            path[i][j] = (i - 1, j)
            dp[i][j] += dp[i - 1][j] + begin[i + j * 2 - 1]

print(dp[m][k])
res = []
cur = path[m][k]
res.append(n)
while cur != (0, 0):
    res.append(cur[0] + cur[1]*2)
    cur = path[cur[0]][cur[1]]
res.append(0)

for _ in reversed(res):
    print(_, end=' ')
print()
```