Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Журнал по исследовательской практике (индивидуальный план)

Команда: MAI #44: Vakhramyan, Kosogorov, Simonov

Студент	Группа
Косогоров Владислав Валерьевич	М8О-306Б-18
Вахрамян Кирилл Олегович	М8О-306Б-18
Симонов Сергей Яковлевич	М8О-306Б-18

Москва, 2021

Сводная таблица за осень 2020

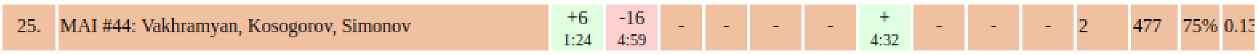
Дата	Название	Время	Место проведения	Решенные задачи
27.09.2020	Grand Prix of Eurasia	11:00-16:00	Дистанционно	1, 7
11.10.2020	Grand Prix of Korea	11:00-16:00	Дистанционно	B, D, G, K
25.10.2020	Тренировка	11:00-16:00	Дистанционно	C, D, J, K
01.11.2020	Grand Prix of Siberia	11:00-16:00	Дистанционно	1, 10, 11
08.11.2020	Grand Prix of Weihai	11:00-16:00	Дистанционно	B, M, N, O, P
15.11.2020	MRC-2020. Main Contest	11:00-16:00	Дистанционно	A, B, C, D
22.11.2020	Regional 1. Northwestern Russia 2020	11:00-16:00	Дистанционно	A, B, I
29.11.2020	Grand Prix of NorthBeach	11:00-16:00	Дистанционно	K, L, M
06.12.2020	RuCode 2020 Winter Championship Div C/D	11:00-16:00	Дистанционно	A, B, C
20.12.2020	XXI Open Cup named after E.V. Pankratiev. Grand Prix of Xiaomi, Div2	11:00-16:00	Дистанционно	N, P, Q, R

Явка на контесты

Дата	Название	Присутствующие
27.09.2020	Grand Prix of Eurasia	Вахрамян, Косогоров, Симонов
11.10.2020	Grand Prix of Korea	Вахрамян, Косогоров, Симонов
25.10.2020	Тренировка	Вахрамян, Косогоров, Симонов
01.11.2020	Grand Prix of Siberia	Вахрамян, Косогоров, Симонов
08.11.2020	Grand Prix of Weihai	Вахрамян, Косогоров, Симонов
15.11.2020	MRC-2020. Main Contest	Вахрамян, Косогоров, Симонов
22.11.2020	Regional 1. Northwestern Russia 2020	Вахрамян, Косогоров, Симонов
29.11.2020	Grand Prix of NorthBeach	Вахрамян, Косогоров, Симонов
06.12.2020	RuCode 2020 Winter Championship Div C/D	Вахрамян, Косогоров, Симонов
20.12.2020	XXI Open Cup named after E.V. Pankratiev. Grand Prix of Xiaomi, Div2	Вахрамян, Косогоров, Симонов

Условия и решения задач

Grand Prix of Eurasia 27.09.2020



25.	MAI #44: Vakhramyan, Kosogorov, Simonov	+6 1:24	-16 4:59	-	-	-	-	+ 4:32	-	-	-	2	477	75%	0.13

Задача #1

Условие: In one of the corners of a N × M sized rectangular field, in a cell with the coordinates (1, 1), sits a hungry cat named Barsik. Barsik's bowl is in the opposite corner of the field, in the cell with the coordinates (N, M). Barsik can traverse the field by moving between cells adjacent by side. However, there is an obstacle, a vicious dog named Tuzik, who sits in a kennel with the coordinates (R, C). Tuzik is chained to the kennel and thus can only reach cells that are within S moves from the kennel (each move going to a cell adjacent by side). All such cells are filled with bones of dead barsiks, and our Barsik cannot make himself walk through them. Barsik desperately needs to know if he can reach his bowl without stepping into Tuzik's area.

Input

The first line of the input file contains an integer T — the number of tests in the problem ($1 \leq T \leq 2000$). The following T lines contain descriptions of tests, one per line. Each tests consists of five space-separated integers N, M, R, C, and S ($1 \leq R \leq N \leq 10^9$). It is guaranteed that the cell with Tuzik's kennel is placed in such a manner that he cannot reach neither the cell with the original position of Barsik nor the cell with Barsik's food.

Output

For each test, print an answer in a separate line. Print Barsik, if Barsik can reach the food without stepping over bones. Otherwise print Tuzik.

Решение

```
#include <iostream>

// M rows, N cols

int main() {
    long int N, M, R, C, S;
    unsigned int T;
    std::cin >> T;
    for (unsigned i = 0; i < T; ++i) {
        std::cin >> N >> M >> R >> C >> S;
        if ((R - S <= 1 && R + S >= N) || (C - S <= 1 && C + S >= M)) {
        // if (2*S + 1 >= N || 2*S + 1 >= M) {
            std::cout << "Tuzik\n";
        } else if (C - S <= 1 && R - S <= 1) {
            std::cout << "Tuzik\n";
        } else if (C + S >= M && R + S >= N) {
            std::cout << "Tuzik\n";
        } else {
            std::cout << "Barsik\n";
        }
    }
    return 0;
}
```

Задача #7

Some of the towns in the province G are connected with roads; roads always go both ways. The budget to support the road network is scarce, therefore it was decided to leave the bare minimum of the roads. However, if it is currently possible to reach the town B from the town A, this possibility must remain even after the reduction. Help to define the minimum number of roads that must remain.

Input

The first line of the input file contains a single integer $T-$ the number of tests ($1 \leq T \leq 50000$).It is followed by the description of T tests. The first line of the test number t contains a single integer N_t — the number of towns ($1 \leq N_t \leq 200$). The following N_t lines contain N_t integers; each integer is either $0 or 1$. If the line with the number i has 0 in the jth position, then the town j can not be reached from the town i (even by driving through other towns); if it is 1, then there is a passage. It is assumed that there is a passage from a town to the same town, so there will always be 1 in the ith line in the ith position ($1 \leq i \leq Nt$). It is guaranteed that the sum of N t 2 over all tests is not greater than 50 000.

Output

For each test, print a single integer on a separate line — the minimum number of roads to be kept.

Решение

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

class Edge{
public:
    int u, v;
    int weight;
    Edge(int _u, int _v, int _w) {
        u = _u;
        v = _v;
        weight = _w;
    }
};
```

```
int get(int v, std::vector<int>& parent) {
    if (parent[v] < 0) {
        return v;
    } else {
        int root = get(parent[v], parent);
        parent[v] = root;
        return root;
    }
}


bool f(int a, int b, std::vector<int>& parent){
    a = get(a,parent);
    b = get(b,parent);

    if (a == b) {
        return false;
    }
    if (parent[a] < parent[b]) {
        std::swap(a,b);
    }
    parent[b] += parent[a];
    parent[a] = b;
    return true;
}


int main() {
    int t;
    std::cin >> t;

    for (int _ = 0; _ < t; ++_) {
        int n;
        std::cin >> n;

        std::vector<std::vector<Edge> > edges(n);
        for (int i = 0; i < n; ++i) {
            for (int j = 0; j < n; ++j) {
                int is_one;
                std::cin >> is_one;
                if (i != j && is_one) {
                    Edge a(i, j, 1);
                    edges[is_one].push_back(a);
                }
            }
        }
        std::vector <int> parent(n + 10, -1);
        int res = 0;
        for (int i = 0; i < n; ++i) {
            for (std::vector<Edge>::iterator it = edges[i].begin();
                 it != edges[i].end(); ++it) {
                if (f(it->v, it->u, parent)) {
                    res += it->weight;
                }
            }
        }
        std::cout << res << std::endl;
    }
    return 0;
}
```

Grand Prix of Korea 11.10.2020

66	МАИ #44: Вахрамян, Косогоров, Симонов: _keoni_, vebcreatex, consky	4	19868		+ 3:03:04		+9 3:03:03			+8 3:07:15				+21 3:17:06	

задача B

Условие

Donghyun is playing a digital card game. In this game, two players start with a deck (stack of cards) of 30 cards and $30HP$ (health points). They alternate turns drawing and playing their cards. The winner is the first person who makes their opponent's HP less than or equal to 0.
This time, Donghyun met a tough opponent; the opponent mixed some bombs into Donghyun's deck! Now, there are A cards in Donghyun's deck, and B of them are bombs. Each card in the deck is equally likely to be a bomb.

Donghyun starts his next turn with C HP. On his next turn, he will remove cards from the top of his deck one at a time until he removes a card that is not a bomb or until his HP becomes less than or equal to 0. For each bomb that he removes, he loses 5 HP. Donghyun's deck is guaranteed to contain at least one card which is not a bomb, so this process is guaranteed to terminate.

Donghyun is worried he may lose the game because of the bombs. Specifically, he will lose the game if and only if his HP becomes less than or equal to 0. Donghyun asks you to calculate the probability that he does not lose the game in his next turn.

Input

On the first and only line, three space-separated integers A, B, and C are given. ($1 \leq B < A \leq 30, 1 \leq c \leq 30$)
Donghyun has A cards in his deck, B of them are bombs, and his current HP is$C$.

Output

Output the probability that Donghyun survives after his next turn.

Your output will be considered correct if the absolute error between your answer and the jury's answer does not exceed 10^{-6}.

Решение

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

```
int main() {
        unsigned long long n, m;
        std::cin >> n;
        m = (n * (n - 1)) / 2;
        std::vector<unsigned long long> v(m);
        for (unsigned long long i = 0; i < m; i++) {
                std::cin >> v[i];
        }

        std::sort(v.begin(), v.end());
        //for (auto a : v) {
        //        std::cout << a << " ";
        //}
        unsigned long long min = 0, max = 0;
        for (unsigned long long i = 0; i < n - 1; i++) {
                min += v[i];
        }

        for (unsigned long long k = 1; k < n; k++) {
                max += v[k * (k - 1) / 2];
        }

        std::cout << min << " " << max << std::endl;
}
```

Задача D
Условие

You are in the spaceship The Skeld with your fellow crewmates. However, while checking the central power system, you found out that one crucial wiring installation of the central power system has been sabotaged. To prevent an engine failure, you should quickly fix the installation.

The installation consists of N nodes and M $M = N(N-1)/2$ wires. All possible pairs of distinct nodes in the installation are connected with a wire. Originally, each of the M wires had exactly one tag attached to it. Each tag has a positive integer value, and different tags may have the same value. However, due to the sabotage, all the tags were removed from the wires and left on the floor. Fortunately, you gathered all M tags from the ground. Now, in order to fix the installation, you have to trigger the reboot sequence by re-attaching all the tags back to the wires twice.

For an installation where all wires have a tag on them, let's define the cost of an installation as the cost of its minimum spanning tree, that is, the minimum cost of a subset of wires such that all nodes are connected using only the given subset of wires. Here, the cost of the set of wires is defined as the sum of the tag values of all wires in the set.

The reboot sequence is triggered in two steps:

- First, you should attach the tags to minimize the cost of the installation.
- Then, after detaching all tags, you should attach the tags to maximize the cost of the installation.

Please compute the cost of each installation.

Input
The first line of the input contains an integer N, representing the number of nodes. $(2 \le N \le 100)$

The second line of the input contains $M = N(N-1)/2$ positive integers $C_1, C_2, ...C_M$, representing the integer values of M tags. $(1 \le C_i \le 2 * 10^9)$

Output
Output a single line containing two integers. The first should be the minimum possible cost of the installation, the second should be the maximum possible cost of the installation.

Решение

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



int main() {
        unsigned long long n, m;
        std::cin >> n;
        m = (n * (n - 1)) / 2;
        std::vector<unsigned long long> v(m);
        for (unsigned long long i = 0; i < m; i++) {
                std::cin >> v[i];
        }
        std::sort(v.begin(), v.end());
        //for (auto a : v) {
        //        std::cout << a << " ";
        //}
        unsigned long long min = 0, max = 0;
        for (unsigned long long i = 0; i < n - 1; i++) {
                min += v[i];
        }

        for (unsigned long long k = 1; k < n; k++) {
                max += v[k * (k - 1) / 2];
        }

        std::cout << min << " " << max << std::endl;
}
```

Задача G
Условие
MOLOCO is a company that matches advertisers with potential users using their high-performance ad platform.

Whenever the application has available space for ads, the app requests the AdExchange platform to determine which ad to show. Then, the AdExchange holds an auction, in which bidders like MOLOCO bid for the opportunity to show their advertisement.

RUN wants to advertise its 2020 ICPC Mock Competition. They have asked MOLOCO for the advertisement. RUN has a total of K dollars and wants to display the ad for internal apps used by KAISTians. The ads in those apps are in one of six types, and the bidding price depends only on the type of ad. The first bidder to bid on the ad gets to show their ad.

AdExchange has already determined the costs of the six ad types and the N auctions it will run today. In the $i-th$ auction, *AdExchange* will run an auction for an ad of type c_i. *AdExchange* never runs two auctions at the same time, more specifically auction $i+1$ cannot start until after auction i ends.

MOLOCO will bid during auctions using the following strategy - before the auctions begin, RUN selects a set of ad types. During the $i-th$ auction, if the ad type for that auction is in RUN's set, and RUN has enough money to bid on an ad of that type, MOLOCO will submit a bid. Otherwise, they will ignore it.

MOLOCO is very fast at bidding, so it will always be the first bidder if it bids on the ad. Determine the maximum number of ads they can bid on if RUN selects the set of ad types optimally.

Input
The first line contains two space-separated integers N, K. ($1 \leq N \leq 100000, 0 \leq K \leq 10^9$)
The next line contains 6 integers $b_1, b_2, ..., b_6 . b_i$ indicates the cost for ad type i. ($1 \leq b_i \leq 10^9$)

The next line contains N integers $c_1, c_2, ..., c_N . c_i$ indicates the ad type of the $i-th$ auction. ($1 \leq c_i \leq 6$)

Output
Print the maximum number of ads they can bid on.

Решение

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

using namespace std;

void subsetsUtil(vector<int>& A, vector<vector<int> >& res,
                 vector<int>& subset, int index) {
    res.push_back(subset);
    for (int i = index; i < A.size(); i++) {
        subset.push_back(A[i]);
        subsetsUtil(A, res, subset, i + 1);
        subset.pop_back();
    }
}

vector<vector<int> > subsets(vector<int>& A) {
    vector<int> subset;
    vector<vector<int> > res;
    int index = 0;
    subsetsUtil(A, res, subset, index);
    return res;
}
```

```
int main() {
    int n, k;
    cin >> n >> k;
    vector<int> B;
    vector<int> C;

    int input;
    for (int i = 0; i < 6; ++i) {
        cin >> input;
        B.push_back(input);
    }

    for (int i = 0; i < n; ++i) {
        cin >> input;
        C.push_back(input);
    }

    vector<int> array = {1, 2, 3, 4, 5, 6};
    vector<vector<int> > subs = subsets(array);

    int res = 0;
    for (int i = 0; i < subs.size(); ++i) {
        int ads_bought = 0;
        int cur_money = k;
        int j = 0;
        while (cur_money > 0 && j < C.size()) {
            bool flag = false;
            for (int f = 0; f < subs[i].size(); ++f) {
                if (C[j] == subs[i][f]) {
                    flag = true;
                    break;
                }
            }
            if (flag && cur_money - B[C[j]-1] >= 0) {
                ads_bought++;
                cur_money -= B[C[j]-1];
            }
            flag = false;
            ++j;
        }
        if (ads_bought > res) {
                res = ads_bought;
            }
        }
        cout << res << endl;
}
```

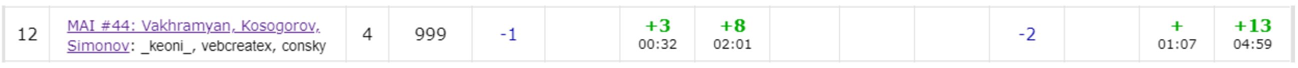
Задача K
Условие
A histogram is a polygon made by aligning N adjacent rectangles that share a common base line. Each rectangle is called a bar. The $i-th$ bar from the left has width 1 and height H_i.// Your goal is to find the area of the largest rectangle contained in the given histogram, such that one of the sides is parallel to the base line.// Actually, no, you have to find the largest square. Since the area of a square is determined by its side length, you are required to output the side length instead of the area.

Input
On the first line, a single integer N is given, where $1 \leq N \leq 300000$.

On the next line, N space-separated integers $H_1, H_2, ..., H_N$, are given. H_i $(1 \leq H_i \leq 10^9)$ is the height of the $i-th$ bar.
Output
Output the side length of the largest square in the histogram, such that one of the sides is parallel to the base line.

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

int main() {

        int n;
        std::cin >> n;
        std::vector <int> v(n);
        for (int i = 0; i < n; i++) {
                std::cin >> v[i];
        }

        int l = 1, r = n;
        int x;
        int count = 0;
        int max_square = 0;
        while (l <= r) {
                x = l + (r - l) / 2;
                count = 0;
                for (int i = 0; i < n; i++) {
                        if (v[i] >= x) {
                                count++;
                        }
                        if (count == x) {
                                break;
                        }
                        if (v[i] < x) {
                                count = 0;
                        }
                }

                if (max_square < count) {
                        max_square = count;
                }
                if (count >= x) {
                        l = x + 1;
                } else if (count < x) {
                        r = x - 1;
                }

        }
        std::cout << max_square << std::endl;
}
```

Тренировка 25.10.2020



12	MAI #44: Vakhramyan, Kosogorov, Simonov: _keoni_, vebcreatex, consky	4	999	-1		+3 00:32	+8 02:01				-2		+ 01:07	+13 04:59

Задача C
Условие:
Новогодняя гирлянда состоит из нескольких последовательно соединенных разноцветных лампочек, причем цвет каждой лампочки не меняется. Участок гирлянды считается красивым, если все лампочки этого участка имеют разные цвета. Определите максимальное количество последовательных лампочек, образующих красивый участок.
Формат входных данных
В первой и единственной строке непустая последовательность из строчных латинских букв, определяющая последовательность цветов лампочек в гирлянде. Одинаковым буквам в этой последовательности соответствуют одинаковые цвета лампочек, а разным – разные. Длина строки не превышает 106 символов.
Формат выходных данных
Целое положительное число – максимальное количество последовательных лампочек, все цвета которых различны.

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

using namespace std;

string findLongestSubstring(string str) {
    int i;
    int n = str.length();
    int st = 0;
    int currlen;
    int maxlen = 0;
    int start;
    unordered_map<char, int> pos;
    pos[str[0]] = 0;
    for (i = 1; i < n; i++) {
        if (pos.find(str[i]) == pos.end())
            pos[str[i]] = i;
        else {
            if (pos[str[i]] >= st) {
                currlen = i - st;
                if (maxlen < currlen) {
                    maxlen = currlen;
                    start = st;
                }
                st = pos[str[i]] + 1;
            }
            pos[str[i]] = i;
        }
    }
    if (maxlen < i - st) {
        maxlen = i - st;
        start = st;
    }

    return str.substr(start, maxlen);
}

int main() {
    string str;
    cin >> str;
    cout << findLongestSubstring(str).length() << endl;
    return 0;
}
```

Задача D
Условие:
Перед показательными выступлениями было проведено N репетиций, на каждой из которых тренировали уникальный элемент выступления. Для репетиций был выделен зал, в который помещалось не более M человек. К сожалению, из-за сильной занятости ни один человек не смог посетить две репетиции, потому каждый в результате мог хорошо исполнить только один элемент. Для выступления потребовалось сформировать M групп, равных по количеству участников. Можно ли это сделать так, чтобы ни в одной группе не было двух людей, посещавших одну репетицию, и соответственно количество элементов выступления, натренированных хотя бы одним участником группы было максимальным.
Формат входных данных
В первой строке два целых числа через пробел: N – количество репетиций, M – количество групп, $1 \leq N, M \leq 100000$. Во второй строке целых неотрицательных чисел через пробел – количество участников каждой репетиции. Гарантируется, что каждую репетицию посетило не более N человек.
Формат выходных данных
В первой и единственной строке YES, если участников можно разбить на M групп, в каждой из которых не будет людей, посещавших одну репетицию, и NO в противном случае.

```
#include <iostream>

using namespace std;

int main() {
    int n, m;
    cin >> n >> m;
    int input;
    int sum = 0;
    for (int i = 0; i < n; ++i) {
        cin >> input;
        if (input >= m) {
            cout << "YES\n";
            return 0;
        }
        sum += input;
    }
    if (sum % m == 0) {
        cout << "YES\n";
    } else {
        cout << "NO\n";
    }
    return 0;
}
```

Задача J

Условие:

На Ваш компьютер напал злобный вирус «Туда-Сюда». Этот вирус преобразовывал строки в текстовых документах по следующему принципу: отрезал от строки несколько символов с начала или с конца и переставлял этот отрезанный кусок в конец или начало соответственно, причем мог делать это несколько раз подряд. Вам повезло, и вирус оказался не только злобным, но и глупым, потому все свои действия он аккуратно логировал. В ходе анализа логов Вы выяснили, что, переставляя K символов из конца в начало, он записывал в лог положительное число K, а переставляя K символов из начала в конец – отрицательное число -K. Зная строку, которая получилась в результате действий вируса, и его лог, определите исходную строку.

Формат входных данных

В первой строке два целых числа через пробел: N – длина исходной строки и K – количество перестановок, выполненных вирусом в строке, $1 \leq N, K \leq 10^5$. Во второй строке последовательность из строчных латинских букв – строка, получившаяся из исходной в результате перестановок. В третьей строке K целых чисел, отличных от 0 и не превышающих K по модулю, – лог действий вируса. Все числа в строке разделены пробелами.

Формат выходных данных

В первой строке и единственной строке – исходная строка символов.

```
n, k = (int(i) for i in input().split())
res = input()
perms = [int(i) for i in input().split()]

for val in reversed(perms):
    if val < 0:
        slice = res[val:]
        res = res[:val]
        res = slice + res
    else:
        slice = res[:val]
        res = res[val:]
        res = res + slice
print(res)
```

Задача K

Условие:

Вам нужно как можно быстрее ответить на t запросов: является ли число n простым?

Входные данные

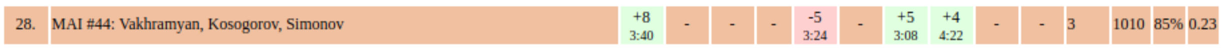
В первой строке дано количество запросов t ($1 \leq t \leq 106$). Во второй строке перечислены числа n ($1 \leq n \leq 4 * 107$), по одному числу на запрос.

Выходные данные

Для каждого запроса выведите в отдельной строке ответ: количество делителей числа, если оно не простое, или строчку "Prime".

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    ios_base::sync_with_stdio(0);
    cin.tie(nullptr);
    vector<int> prime(4 * 10'000'000 + 1, 0);
    for (int i = 2; i < prime.size(); ++i) {
        if (prime[i] ==0 ) {
            for (int j = 2; i * j < prime.size(); ++j) {
                prime[i * j]++;
            }
        }
    }
    int n;
    cin >> n;
    for (int i = 0; i < n; ++i) {

        int num;
        cin >> num;
        if (num == 1) {
            cout << "0\n";
        } else if (prime[num] == 0) {
            cout << "Prime\n";
        } else {
            cout << prime[num] << "\n";
        }
    }

}
```

Grand Prix of Siberia 01.11.2020



28.	MAI #44: Vakhramyan, Kosogorov, Simonov	+8 3:40	-	-	-	-5 3:24	-	+5 3:08	+4 4:22	-	-	3	1010	85%	0.23

Задача №1

Условие

Roman is a tech writer in a big IT company, Textawei. His work consists in writing technical documentations, rules, specifications, etc. The volumes of the texts he has written are immense. Everything was going great until one day he received a letter from his boss... The letter stressed the importance of keeping the information in the documents relevant and keep the style consistent, and there was an attachment with a set of rules — the description of the style. Moreover, they set up a whole system to check all text files and point out inconsistencies.
Roman rushed to view the report on his documents and saw around 100500 urgent problems. Despite his massive grief, Roman bravely decided to analyze his mistakes. It turned out that the majority of the inconsistencies had to do with a single rule, which stated: “Before and after each colon and dash there must be at least one space or beginning/end of line”.
At this point, Roman realized that he could come up with a program to fix his documents, at least in regards to this particular rule. The program would add the minimal number of spaces to the text in such a way that the rule is satisfied.
Unfortunately, Roman is a tech writer and cannot code. Help him!

Input]

The first line of the input file contains a single integer T — the number of lines in the text ($1 \le T \le 10000$). 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

```
n = int(input())

for j in range(n):
    inp = input()
    f1 = False
    f2 = False
    for i in range(0, len(inp)):
        if (inp[i] == ':' or inp[i] == '-'):
            if (len(inp) == 1):
                print(inp[0], end = '')
                continue
            elif (i == 0):
                if (inp[1] != ' '):
                    print(inp[0] + ' ', end = '')
                    f2 = True
                else:
                    print(inp[0], end = '')
            elif (i == len(inp) - 1):
                if (inp[len(inp) - 2] != ' ' and not f2):
                    print(' ' + inp[len(inp) - 1], end = '')
                    f2 = False
                else:
                    print(inp[len(inp) - 1], end = '')
            else:
                if (inp[i - 1] != ' ' and not f2):
                    print(' ' + inp[i], end = '')
                    f1 = True
                    f2 = False
                else:
                    print(inp[i], end = '')
                    f1 = True
                if (inp[i + 1] != ' '):
                    if (f1):
                        print(' ', end = '')
                        f1 = False
                    else:
                        print(inp[i] + ' ', end = '')
                    f2 = True
        else:
            print(inp[i], end = '')
            f2 = False
    print('')
```

Задача №10
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 $N * N$, 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

```
#include <bits/stdc++.h>
using namespace std;

vector<vector<char> > rotateMatrix_protiv_chasovoy(vector<vector<char>>&
                                                        matr, int n) {
    vector<vector<char> > mat = matr;
    int N = mat.size();
    for (int i = 1; i <= n; ++i) {
        for (int x = 0; x < N / 2; x++) {
            for (int y = x; y < N - x - 1; y++) {
                int temp = mat[x][y];
                mat[x][y] = mat[y][N - 1 - x];
                mat[y][N - 1 - x] = mat[N - 1 - x][N - 1 - y];
                mat[N - 1 - x][N - 1 - y] = mat[N - 1 - y][x];
                mat[N - 1 - y][x] = temp;
            }
        }
    }
    return mat;
}

vector<vector<char> > FlipVertical(vector<vector<char> >& matr) {
    vector<vector<char> > mat = matr;
    for (int r = 0; r < (mat.size()/2); r++) {
        for (int c = 0; c != mat.size(); ++c) {
             std::swap(mat[r][c], mat[mat.size() - 1 - r][c]);
        }
    }
    return mat;
}

vector<vector<char> > FlipHorizontal(vector<vector<char> >& matr) {
    vector<vector<char> > mat = matr;
    reverse(mat.begin(), mat.end());
    return mat;
}

int main() {
    int n;
    cin >> n;
    vector<vector<char> > mat_1;
    vector<vector<char> > mat_2;
    char c;

    for (int i = 0; i < n; ++i) {
        vector<char> input;
        for (int j = 0; j < n; ++j) {
            cin >> c;
            input.push_back(c);
        }
        mat_1.push_back(input);
        getchar();
    }

    getchar();
```

```
for (int i = 0; i < n; ++i) {
    vector<char> input;
    for (int j = 0; j < n; ++j) {
        cin >> c;
        input.push_back(c);
    }
    mat_2.push_back(input);
    getchar();
}

vector<vector<char >> rotated_mat;
vector<vector<char> > var_1;

for (int f = 0; f <= 3; ++f) {
    bool flag = true;
    rotated_mat = rotateMatrix_protiv_chasovoy(mat_1, f);
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            if (rotated_mat[i][j] != mat_2[i][j]) {
                flag = false;
            }
        }
    }
    if (flag) {
        cout << "YES\n";
        return 0;
    }

    flag = true;
    vector<vector<char> > var_vert = FlipVertical(rotated_mat);
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            if (var_vert[i][j] != mat_2[i][j]) {
                flag = false;
            }
        }
    }
    if (flag) {
        cout << "YES\n";
        return 0;
    }

    flag = true;
    vector<vector<char> > var_hor = FlipHorizontal(rotated_mat);
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            if (var_hor[i][j] != mat_2[i][j]) {
                flag = false;
            }
        }
    }
    if (flag) {
        cout << "YES\n";
        return 0;
    }

    flag = true;
    vector<vector<char> > var_hor_vert = FlipHorizontal(var_vert);
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            if (var_hor_vert[i][j] != mat_2[i][j]) {
                flag = false;
            }
        }
    }
    if (flag) {
        cout << "YES\n";
        return 0;
    }

    flag = true;
    vector<vector<char> > var_vert_hor = FlipVertical(var_hor);
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            if (var_vert_hor[i][j] != mat_2[i][j]) {
                flag = false;
            }
        }
    }
    if (flag) {
        cout << "YES\n";
```

```
                return 0;
            }
        }

        cout << "NO\n";


}
```
Задача №11
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.

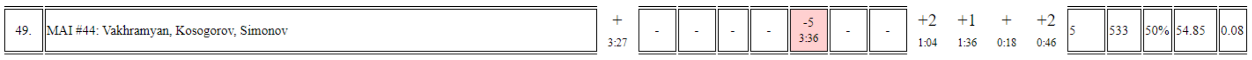
Output
For each of the extensions that are present in the input file, output the number of files with this extension in the form of **<extension>: <number>**. Output extensions in order of the first mention in the input file.

Решение

```
n = int(input())
res = []
count = []

for j in range(n):
    _, r = input().split('.')
    if (res.count(r) > 0):
        count[res.index(r)] += 1
    else:
        res.append(r)
```

```
            count.append(1)

for i in range(len(res)):
    print(res[i], end = ': ')
    print(count[i])
```

Grand Prix of Weihai 08.11.2020



49.	MAI #44: Vakhramyan, Kosogorov, Simonov	+ 3:27	-	-	-	-	-5 3:36	-	-	+2 1:04	+1 1:36	+ 0:18	+2 0:46	5	533	50%	54.85	0.08

Задача B

Условие:

It’s a sunny day with good scenery, and you come to the park for a walk. You feel curious that there are many old guys gathering by a bridge, and want to take a look at what happened. There are exactly n old guys on each side of the bridge, and they all want to go across the bridge, take some time for relaxing on the other side, and finally go across the bridge back to the original side. However, they are too old to cross the bridge by themselves. Driven by the golden spirit in your heart, you want to help these 2n old guys. Initially, you are on one side of the bridge. It takes t minutes for you to go across the bridge and x minutes for an old guy relaxing. You may help an old guy when you cross the bridge, which doesn’t take extra time. As a master of time management, you want to know the minimum time needed to help all these 2n old guys. Please write a program to calculate this minimum time.

Входные данные

The first line contains one integer T ($1 \leq t \leq 10^4$), indicating the total number of test cases. For each of the next T lines, there are three integers n, x, t ($1 \leq n, x, t \leq 10^9$), as explained in problem statement.

Выходные данные

You should output exactly T lines. Each line should contain exactly one integer, indicating the minimum time you needed in each test case.

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

using namespace std;

int main() {
    long long n;
    cin >> n;
    vector<long long> ans;
    for (long long i = 0; i < n ; ++i){
        long long x, t, n;
        cin >> n >> x >> t;
        long long time_1 = 0;
        time_1 += 2 * t * n;
        long long time_2 = time_1;
        time_2 = max(x + 2 * t, time_1);
        time_1 = max(x + t, time_1 + t);
        time_1 = min(time_1, time_2);
        time_1 += t * 2 * n;
        ans.push_back(time_1);
    }
    for (long long i = 0; i < ans.size(); ++i ) {
        cout << ans[i] << endl;
    }
    return 0;
}
```

Задача M
Условие:
The KV Cup is the online programming competition for pairs of programmers. Coach of some big university is in charge of registration of teams for this competition. There is an even number of students who desire to compete, so each student does compete. For each student the rating on the famous site Forcedcoders is known. Coach considers rating of the pair as sum of ratings of the contestants in it. Coach wants to form teams in such a way, that lowest rating of the pair is maximized. Help him to find this rating. Note that each student may participate exactly in one pair.

Входные данные
The first line of input contains a single integer n ($1 \leq n \leq 10^5$), representing the number of students who desire to enter the KV Cup. It is guaranteed that n is an even number. Each of the following n lines contains a single integer r ($1 \leq r \leq 10^6$), representing the rating of a student.

Выходные данные
Print one integer — maximal possible value of lowest rating of the pair.

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

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

    vector<int> ratings;
    int r;
    for (int i = 0; i < n; ++i) {
        cin >> r;
        ratings.push_back(r);
    }
    vector<pair<int, int> > pairs;

    sort(ratings.begin(), ratings.end(), greater<int>());
    for (int i = 0, j = ratings.size() - 1; i <= j; i++, j--) {
        // cout << "(" << ratings[i] << ", " << ratings[j] << ")" << " ";
        pairs.push_back(make_pair(ratings[i], ratings[j]));
    }

    // cout << pairs.back().first + pairs.back().second << endl;
    // cout << ratings[n / 2] + ratings[(n + 1)/2] << endl;
    int min = numeric_limits<int>::max();
    for (int i = 0; i < pairs.size(); ++i) {
        if (pairs[i].first + pairs[i].second < min) {
            min = pairs[i].first + pairs[i].second;
        }
    }
    cout << min << endl;
}
```

Задача N
Условие:
Two programmers sit in a car following the bus in a traffic jam. At the stop two people enter the bus. Then, three people are observed leaving the bus. Then one of the programmers said: If one person is to enter now, the bus will be empty! Those programmers did several observations and wrote down numbers of people who entered and left the bus at the consecutive stops. Your goal is to write a program that will calculate the minumum number of passenger in the bus before their observations started.

Входные данные
The first line of the input consists of a single integer T ($0 < T \leq 50$) — the number of test cases. Each of the test cases begins with a line containing a single integer M ($0 < T \leq 50$) — number of observations. Then sequential observations follow each on the new line. Each observation is described by two integers P1 and P2 separated by a space — number of people, entering the bus, and then number of people, leaving the bus, respectively ($0 \leq P1$, $P2 \leq 1000$). Note that one observation consists of two events: first, P1 people enter the bus, then P2 people leave the bus.

Выходные данные
For each test case, print the minimum number of people that would have to have been inside the bus at the beginning.

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

```
#include <climits>
using namespace std;


int main() {
        int t;
        cin >> t;
        for (int i = 0; i < t; i++) {
                int m;
                cin >> m;
                int min = INT_MAX;

                int tmp = 0;
                for (int i = 0; i < m; i ++) {
                        int p1, p2;
                        cin >> p1 >> p2;
                        int dif = p1 - p2 + tmp;
                        tmp = dif;
                        if (dif < min) {
                                min = dif;
                        }
                }
                if (min < 0) {
                        cout << -min << "\n";
                } else {
                        cout << 0 << "\n";
                }
        }

}
```

Задача O

Условие:

Given a list of words consisting of only capital English letters, create a list by reversing them and printing the reversals in alphabetical order.

Входные данные

Each input will consist of a single test case. The first line of input will contain an integer n ($1 \leq n \leq 1,000$) indicating the number of words. On the following n lines will be the words, one per line. The words will be from 1 to 100 letters long. The words will consist of only capital letters, and there will be no spaces or blank lines.

Выходные данные

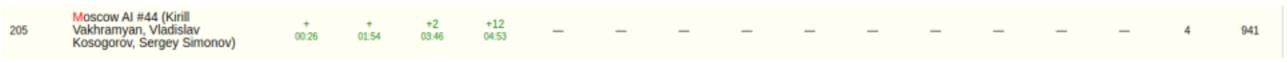
Output the words, reversed and sorted, one word per line.

```
n = int(input())
l = list()

for i in range(n):
    l.append(input()[:: -1])

l.sort()

for word in l:
    print(word)
```

Задача P
Условие:
A guessing game is played in the next way: given a goal number N (hidden) and a range (X, Y) containing it (not including X and Y), everyone takes turns to guess what N is. After every wrong guess, the range replaces to a new range with either X or Y replaced by the guessed number. The game ends when a guess hits the number N. Supposing that everyone always guesses the median of the range (if there are two, choose the smaller one), how many guesses need to be played for the game to end?
Входные данные
The first line of the input file input contains three integers N, X and Y ($0 \leq N$, X, $Y \leq 104$)
Выходные данные
Print number of guesses that will be played

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

int main() {
        int x, y, n;
        cin >> n >> x >> y;
        x++;
        y--;
        int count = 0;
        while (true) {
                int m = x + (y - x) / 2;
                count++;
                if (m == n) {
                        cout << count << "\n";
                        break;
                } else if (m > n) {
                        y = m - 1;
                } else if (m < n) {
                        x = m + 1;
                }
        }
}
```

MRC-2020. Main Contest 15.11.2020



205	Moscow AI #44 (Kirill Vakhramyan, Vladislav Kosogorov, Sergey Simonov)	+ 00:26	+ 01:54	+2 03:46	+12 04:53	—	—	—	—	—	—	—	—	—	—	4	941

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 $x1$ and $y1$ and the goal of his adventure is to reach the free point with non-zero coordinates $x2$ and $y2$. He is allowed to choose any route he wants. Calculate minimum possible fee to be paid by adventurer.

Input

Input contains four integers $x1$,y_1, x_2 and y_2 — coordinates of the start point and the finish point respectively ($x_1 \neq 0, y_1 \neq 0, x_2 \neq 0, y_2 \neq 0, -10000 \leq x1, y_1, x$,$y_2 \leq 10000$).

Output

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

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


int main() {
    int a1, b1, a2, b2;
    cin >> a1 >> b1 >> a2 >> b2;
    if ((a1 > 0 && a2 > 0 || a1 < 0 && a2 < 0)  &&
        (b1 > 0 && b2 > 0 || b1 < 0 && b2 < 0)) {
        cout << "0\n";
        return 0;
    }
    cout << "1\n";
}
```

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.

```
names_in_order = list()

team_1 = dict()
team_2 = dict()

team_1_cur = list()
team_2_cur = list()

team_1_name = input()

for i in range(5):
    name = input()
    team_1[name] = 0
    team_1_cur.append(name)
    names_in_order.append(name + '1')

team_2_name = input()
for i in range(5):
    name = input()
    team_2[name] = 0
    team_2_cur.append(name)
    names_in_order.append(name + '2')

n = int(input())
inp_str = []
for i in range(n):
    inp_str = input().split()
    if inp_str[2] == 'scored':
        score = int(inp_str[-1])
        if inp_str[1] == team_1_name:
            for name in team_1_cur:
                team_1[name] += score
            for name in team_2_cur:
                team_2[name] -= score
        else:
            for name in team_2_cur:
                team_2[name] += score
            for name in team_1_cur:
                team_1[name] -= score

    else:
        name = inp_str[-1]
        if inp_str[1] == team_1_name:
            team_1_cur.remove(inp_str[3])
            team_1_cur.append(name)
            if name not in team_1:
                team_1[name] = 0
                names_in_order.append(name + '1')
        else:
            team_2_cur.remove(inp_str[3])
            team_2_cur.append(name)
            if name not in team_2:
                team_2[name] = 0
                names_in_order.append(name + '2')

for name in names_in_order:
    if name[-1] == '1':
        score = int(team_1[name[:-1]])
        if score > 0:
            print(name[:-1] + ' (' + team_1_name + ') +' + str(score))
        else:
            print(name[:-1] + ' (' + team_1_name + ') ' + str(score))
    else:
        score = int(team_2[name[:-1]])
        if score > 0:
            print(name[:-1] + ' (' + team_2_name + ') +' + str(score))
        else:
            print(name[:-1] + ' (' + team_2_name + ') ' + str(score))
```

Задача C
Условие

In this problem we will deal with byte sequences. Each byte is given by two hexadecimal digits (four bits each, eight bits in total) and is treated as an unsigned integer between 0 and $ff_16 = 255_10$ (inclusive). Hexadecimal digits are denoted 0123456789abcdef, as usual.
Consider the C ASCII string to be the byte sequence starting with zero or more bytes from range between 20_16 and $7f_16$ (inclusive) followed by a zero byte. There may be arbitrary bytes following the zero byte, they are called "junk" bytes.
Consider the Pascal ASCII string to be the byte sequence starting with byte l defining the length of the string, followed by l bytes from range between 20_16 and $7f_16$ (inclusive). There may be arbitrary bytes following the last of these l bytes, they are also called "junk" bytes. In particular, Pascal string length is limited to the range from 0 to $ff_16 = 255_10$.
You are given a memory dump as a sequence of integers bi given in hexadecimal form. Each integer belongs to the range from 0 to ff_16 (inclusive). Your task is to determine, which of four cases takes place:

- dump defines a valid C ASCII string as well as a valid Pascal ASCII string;
- dump defines a valid C ASCII string, but not a valid Pascal ASCII string;
- dump defines a valid Pascal ASCII string, but not a valid C ASCII string;
- dump does not define a valid C ASCII string or a valid Pascal ASCII string. Note that in all cases junk bytes are allowed

Input

Input contains a sequence of no more than 1000 space-separated hexadecimal integers; each integer consist of exactly two characters, each of which is from one of the 0123456789abcdef characters. In particular, leading zeroes are allowed.

Output

If the dump can be interpreted both as a C ASCII string and as a Pascal ASCII string, print the word "Both".
If the dump can be interpreted as a C ASCII string but not as a Pascal ASCII string, print the word "C".
If the dump can be interpreted as a Pascal ASCII string but not as a C ASCII string, print the word "Pascal".
If the dump can't be interpreted as a C ASCII string or as a Pascal ASCII string, print "None".

```
import java.lang.Long.parseLong
```

```
private fun readLn() = readLine()!!
private fun readInt() = readLn().toInt()
private fun readStrings() = readLn().split(" ")
private fun readInts() = readStrings().map {
    it.toInt()
}

fun main()  {
    var a = readLine()!!
    a = a.toUpperCase()
    var strs = a.split(" ")
    var isPascal = true
    var isC = true
    var sizeStr = -1L
    var size1 = 0L
    var l = 2*16L

    var isZeroByte = false
    var r = parseLong("7F",16)
    for (str in strs) {
        var sum1 = parseLong(str,16)
        if (sizeStr==-1L) {
            sizeStr = sum1
            if (sizeStr>255) {
                isPascal = false
            }
            if (sum1==0L) {
                isZeroByte = true
            }
            if (!isZeroByte && (sum1 !in l..r)) {
                isC = false
            }
        } else {
            size1++
            if ((sum1 !in l..r) && size1<=sizeStr) {
                isPascal = false
            }
            if (sum1==0L) {
                isZeroByte = true
            }
```

```
                if (!isZeroByte && (sum1 !in l..r)) {
                    isC = false
                }
            }

        }
        if (isC) {
            isC = isZeroByte
        }
        if (isPascal) {
            if (size1<sizeStr) {
                isPascal = false
            }
        }
        if (isC && isPascal) {
            println("Both")
        }
        if (!isC && !isPascal) {
            println("None")
        }
        if (isC && !isPascal) {
            println("C")
        }
        if (!isC && isPascal) {
            println("Pascal")
        }
    }
```

Задача D
Условие

For any non-negative integer x we can consider its decimal representation as a string of digits from 0 to 9. Denote this string as $S(x)$. On the opposite, for any string s of decimal digits we can get an integer it represents by neglecting all leading zeroes, except maybe one to represent integer 0. Denote this integer $D(s)$.
We say that integer y is an integer-substring of integer x if there exists a string s that is a substring (consecutive subsequence) of $S(x)$ and $D(s) = y$.
Recall that integer x is called prime if it is positive and has exactly two divisors. First five primes are $2, 3, 5, 7, 11$
We call integer x deep prime if it is prime and any y that is integer-substring of x is prime.
You are given two integers n and m, calculate the number of deep prime integers between

n and m,inclusive.

Input

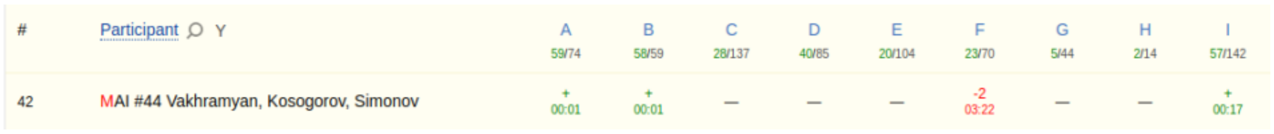
The only line of the input contains two integers n and m ($1 \leq n \leq m \leq 10^18$)

Output

Print one integer — the number of deep prime integers x that satisfy $n \leq x \leq m$.

```
#include <iostream>
using namespace std;
int main() {
        unsigned long long a, b;
        cin >> a >> b;
        int ans = 0;
        int arr[9] = {2,3,5,7,23,37,53,73,373};
        for (unsigned long long i = 0; i < 9; ++i) {
        if (arr[i] >= a && arr[i] <= b) {
                ans++;
        }
    }
        cout << ans << endl;
}
```

Regional 1. Northwestern Russia 2020 22.11.2020



#	Participant	A 59/74	B 58/59	C 28/137	D 40/85	E 20/104	F 23/70	G 5/44	H 2/14	I 57/142
42	MAI #44 Vakhramyan, Kosogorov, Simonov	+ 00:01	+ 00:01	—	—	—	-2 03:22	—	—	+ 00:17

Задача A

Условие:

The team of problemsetters of Northwestern Russia Regional Contest welcomes you! Oe regional contest was founded in 1995 under the name "Cooegiate Programming Championship of St Petersburg". Here is the list of the contest winners:

- 1995: ITMO
- 1996: SPbSU
- 1997: SPbSU
- 1998: ITMO
- 1999: ITMO
- 2000: SPbSU
- 2001: ITMO
- 2002: ITMO
- 2003: ITMO
- 2004: ITMO
- 2005: ITMO
- 2006: PetrSU, ITMO
- 2007: SPbSU
- 2008: SPbSU
- 2009: ITMO
- 2010: ITMO
- 2011: ITMO
- 2012: ITMO

- 2013: SPbSU
- 2014: ITMO
- 2015: ITMO
- 2016: ITMO
- 2017: ITMO
- 2018: SPbSU
- 2019: ITMO

Help the contest arhivist to remember the results of each contest and write a program that will read the year and print contest winners of that in exactly same format as above.
Input
The only line of inputcontains 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
Print the winner of the contest in year y exactly in the same format as in the list above.

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

using namespace std;


int main() {
        map<int, string> abcdeyka;
        abcdeyka.insert(make_pair<int,string>(1995, "ITMO"));
        abcdeyka.insert(make_pair<int,string>(1996, "SPbSU"));
        abcdeyka.insert(make_pair<int,string>(1997, "SPbSU"));
        abcdeyka.insert(make_pair<int,string>(1998, "ITMO"));
        abcdeyka.insert(make_pair<int,string>(1999, "ITMO"));
        abcdeyka.insert(make_pair<int,string>(2000, "SPbSU"));
        abcdeyka.insert(make_pair<int,string>(2001, "ITMO"));
        abcdeyka.insert(make_pair<int,string>(2002, "ITMO"));
        abcdeyka.insert(make_pair<int,string>(2003, "ITMO"));
        abcdeyka.insert(make_pair<int,string>(2004, "ITMO"));
        abcdeyka.insert(make_pair<int,string>(2005, "ITMO"));
        abcdeyka.insert(make_pair<int,string>(2006, "PetrSU, ITMO"));
```

```
    abcdeyka.insert(make_pair<int,string>(2007, "SPbSU"));
    abcdeyka.insert(make_pair<int,string>(2008, "SPbSU"));
    abcdeyka.insert(make_pair<int,string>(2009, "ITMO"));
    abcdeyka.insert(make_pair<int,string>(2010, "ITMO"));
    abcdeyka.insert(make_pair<int,string>(2011, "ITMO"));
    abcdeyka.insert(make_pair<int,string>(2012, "ITMO"));
    abcdeyka.insert(make_pair<int,string>(2013, "SPbSU"));
    abcdeyka.insert(make_pair<int,string>(2014, "ITMO"));
    abcdeyka.insert(make_pair<int,string>(2015, "ITMO"));
    abcdeyka.insert(make_pair<int,string>(2016, "ITMO"));
    abcdeyka.insert(make_pair<int,string>(2017, "ITMO"));
    abcdeyka.insert(make_pair<int,string>(2018, "SPbSU"));
    abcdeyka.insert(make_pair<int,string>(2019, "ITMO"));

    int year;
    cin >> year;

    cout << abcdeyka[year];



}
```

Задача B
Условие:
After a long time at home the quarantine, in November you decided to go to work by bicycle! Since you do not have your own bicycle, you have to rent one. The bike rental allows you to choose one of two monthly options:

- The most fee is a roubles. Every day, the first 30 minutes are free, and every minute above that costs x roubles.
- The monthly fee is b roubles. Every day, the first 45 minutes are free, and every minute above that costs y roubles.

There are 21 working days in November, and you spend T minutes commuting to work and back home in total every day. Calculate how many roubles you would spend if you use either one of two monthly options.
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

Output
The only line of the output should contain two integers - the amount of money you would spend on the first option and second option, respectivety.

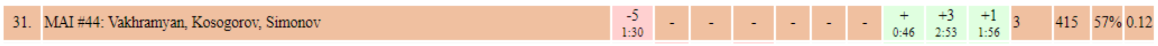
```
#include <bits/stdc++.h>

using namespace std;



int main() {
        int a, x, b, y;
        cin >> a >> x >> b >> y;
        int T;
        cin >> T;
        long long int sum1 = 0, sum2 = 0;
        int delta;
        delta = T - 30;
        if (delta > 0) {
                delta *= x;
                delta *= 21;
                sum1 += delta;
        }
        sum1 += a;



        delta = T - 45;
        if (delta > 0) {
                delta *= y;
                delta *= 21;
                sum2 += delta;
        }
        sum2 += b;


        cout << sum1  << " " << sum2  << "\n";
}
```

Задача I
Условие:

Iris likes to draw different shapes on a grid paper. Her favorite shape is a square. She has recently tried to draw a square with area s, but since she uses a grid paper, she wants all the vertices of the square to have integer coordinates.
Can you help her find such a square?
Input
The input contains a single integer s ($1 \leq s \leq 1000$).
Output
If it is possible to the required square, output four pairs of integers: the coordinates of the vertices of the square, in any order. All coordinates should be in the range form -10^9 to 10^9. If there are multiple answers, output any of them. If it is impossible to construct the required square, print "Impossible".

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

using namespace std;



int main() {
        int s;
        cin >> s;
        int x, y;
        int flag = 0;
        for (x = 0; x <= sqrt(s); x++) {
                for (y = 0; y <= sqrt(s); y++) {
                        if (s == x * x + y * y) {
                                flag = 1;
                                break;
                        }

                }
                if (flag == 1) {
                                break;
                        }
        }

        if (flag == 0) {
                cout << "Impossible\n";
        } else {
                if (x > y)
                        swap (x, y);
                int c1, c2;
                c1 = 0, c2 = y;
                cout << c1 << " " << c2 << "\n";
                c1 = c1 + y, c2 = c2 + x;
                cout << c1 << " " << c2 << "\n";
                c1 = c1 + x, c2 = c2 - y;
                cout << c1 << " " << c2 << "\n";
                c1 -= y, c2 -= x;
                cout << c1 << " " << c2 << "\n";

        }
}
```

Grand Prix of NorthBeach 29.11.2020



31.	MAI #44: Vakhramyan, Kosogorov, Simonov	-5 1:30	-	-	-	-	-	-	+ 0:46	+3 2:53	+1 1:56	3	415	57%	0.12

Задача K

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 ($2 \leq N \leq 10:9$) — number of cities in Byteland.

Output

Print one integer — least possible number of autonomous cities.

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

using namespace std;

int main() {
        int x;
        cin >> x;
        cout << 2 << '\n';
}
```

Задача M

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. 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 beautiful 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.

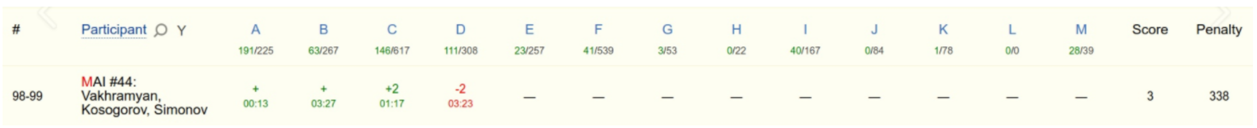
```
#include <bits/stdc++.h>
```

```
using namespace std;

int main() {
        long long int x;
        cin >> x;

        cout << x + 1 << '\n';
}
```

Задача L
Условие:
Let's define customity of two adjacent elements a_i and a_{i+1} if 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.

```
#include <iostream>
#include <unordered_map>
#include <vector>
#include <limits>

using namespace std;

int main() {
    long long n;
    cin >> n;
    vector<long long> v;
    long long input;
    long long max = numeric_limits<long long>::min();
    for (long long i = 0; i < n; ++i) {
        cin >> input;
        if (input > max) {
            max = input;
        }
        v.push_back(input);
    }

    unordered_map<long long, long long> counts;
    for (auto val : v) {
        ++counts[val];
    }

    cout << n - counts[max] << endl;
}
```

RuCode 2020 Winter Championship Div C/D 06.12.2020



#	Participant	A 191/225	B 63/267	C 146/617	D 111/308	E 23/257	F 41/539	G 3/53	H 0/22	I 40/167	J 0/84	K 1/78	L 0/0	M 28/39	Score	Penalty
98-99	MAI #44: Vakhramyan, Kosogorov, Simonov	+ 00:13	+ 03:27	+2 01:17	-2 03:23	—	—	—	—	—	—	—	—	—	3	338

Задача A

Условие:

В этой задаче отношение входных и выходных данных постоянно.

Input

Входные данные содержат одно целое число N ($1 \leq N \leq 10^6$)

Output

Выведите одно целое число — ответ к задаче.

```
#include <iostream>

int main() {
        long tmp;
        std::cin >> tmp;
        std::cout << tmp * 10 << std::endl;
        return 0;
}
```

Задача B

Условие:

В супервекторном процессоре «Спутник-19» имеется N различных видов регистров - однобайтовые, двухбайтовые, ... N-байтовые. Соответственно, имеется 2*N целочисленных типов данных — знаковый и беззнаковый 8N-разрядный. Числа в беззнаковом типе данных представляют собой остатки от деления на 2^{8N}, в знаковом типе числа от 0 до $2^{8N-1} - 1$ представляются так же, как и в беззнаковом, а любое число x от -2^{8N-1} до -1 представляется так же, как беззнаковое число $2^{8N} + x$ В языке B++ реализованы все целочисленные типы данных, при этом контроль типов полностью отсутствует, а присваивание происходит с переполнением, то есть если присваиваемая константа не помещается в тип данных, то соответствующее количество старших байтов отбрасывается, а младшие копируются в нужный тип. Например, если мы присваиваем переменной x значение 257, а переменная имеет однобайтовый беззнаковый тип, то после присваивания значение x становится равно 1. Если нам неизвестен тип переменной (а в B++ такое случается), можно попытаться присвоить переменной какое-то число, прочитать то, что в неё записалось и на основании этого определить тип данных. Назовём детектором целое неотрицательное число, которое при таком действии позволяет однозначно определить тип переменной. Ваша задаача — по заданному N вывести шестнадцатеричное представление наименьшего детектора для заданного N.

Input
Входные данные содержат одно целое число N - количество видов регистров ($1 \leq N \leq 10^4$).
Output
Выведите шестнадцатеричное представление наименьшего детектора без ведущих нулей. Цифры, большие 9, обозначайте строчными латинскими буквами от 'A' до 'F'.

```
n = int(input())
res = ''
for i in n:
    res += '80'
print(res)
```

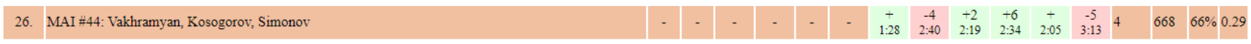
Задача C
Условие:
Известный граффитист Б. как-то ночью превратил координатную плоскость в арт-объект. Координатные оси он покрасил в чёрный цвет, первую четверть — в красный, вторую — в синий, третью — в зелёный, четвёртую — в жёлтый. Для заданного круга скажите, точки скольки различных цветов он содержит. Считается, что круг состоит из окружности и внутренних точек.
Input
Первая строка входных данных содержит три целых числа x, y и r - координаты центра и радиус круга, соответственно ($-10000 \leq x, y \leq 100000, 1 \leq r \leq 100000$)
Output
Выведите одно целое число — количество различных цветов точек, содержащихся внутри или на границе круга.

```
#include <iostream>
#include <cmath>

int main() {
        int x, y, r;
        std::cin >> x >> y >> r;
        int tmp;
        if (x == y && x == 0 && y == 0) {
                std::cout << "5" << "\n";
                return 0;
        }
        int count = 1, t1, t2;
        if (x >= 0 && y >= 0) {
                t1 = x - r;
                t2 = y - r;
                if (t1 < 0) {
                    count++;
                }
                if (t2 < 0) {
                    count++;
                }
                if (t1 < 0 && t2 < 0) {
                    tmp = sqrt((x * x) + (y * y));
                    if (tmp < r) {
                        count++;
                    }
                }
                if (count > 1) {
                    std::cout << count + 1 << "\n";
                    return 0;
                } else if (count == 1) {
                    if (t1 == 0 || t2 == 0) {
                        count++;
                    }
                    std::cout << count << "\n";
                    return 0;
                }
        } else if (x <= 0 && y >= 0) {
                t1 = x + r;
                t2 = y - r;
                if (t1 > 0) {
                    count++;
                }
                if (t2 < 0) {
                    count++;
                }
                if (t1 > 0 && t2 < 0) {
                    tmp = sqrt((x * x) + (y * y));
                    if (tmp < r) {
                        count++;
                    }
                }
                if (count > 1) {
                    std::cout << count + 1 << "\n";
                    return 0;
                } else if (count == 1) {
```

```
                        if (t1 == 0 || t2 == 0) {
                                count++;
                        }
                        std::cout << count << "\n";
                        return 0;
                }
        } else if (x >= 0 && y <= 0) {
                t1 = x - r;
                t2 = y + r;
                if (t1 < 0) {
                        count++;
                }
                if (t2 > 0) {
                        count++;
                }
                if (t1 < 0 && t2 > 0) {
                        tmp = sqrt((x * x) + (y * y));
                        if (tmp < r) {
                                count++;
                        }
                }
                if (count > 1) {
                        std::cout << count + 1 << "\n";
                        return 0;
                } else if (count == 1) {
                        if (t1 == 0 || t2 == 0) {
                                count++;
                        }
                        std::cout << count << "\n";
                        return 0;
                }
        } else if (x <= 0 && y <= 0) {
                t1 = x + r;
                t2 = y + r;
                if (t1 > 0) {
                        count++;
                }
                if (t2 > 0) {
                        count++;
                }
                if (t1 > 0 && t2 > 0) {
```

```
            tmp = sqrt((x * x) + (y * y));
            if (tmp < r) {
                count++;
            }
        }
        if (count > 1) {
            std::cout << count + 1 << "\n";
            return 0;
        } else if (count == 1) {
            if (t1 == 0 || t2 == 0) {
                count++;
            }
            std::cout << count << "\n";
            return 0;
        }
    }
    return 0;
}
```

XXI Open Cup named after E.V. Pankratiev. Grand Prix of Xiaomi, Division 2. 20.12.2020



26.	MAI #44: Vakhramyan, Kosogorov, Simonov	-	-	-	-	-	-	+ 1:28	-4 2:40	+2 2:19	+6 2:34	+ 2:05	-5 3:13	4	668	66%	0.29

Задача N

Условие:

You are trying to send a message via NFC to a friend from your smartphone. A smartphone can connect to another one nearby via NFC if the distance between them is not greater than D. Determine whether a communication path between phones can be formed via zero or more intermediate smartphones so that your message can be successfully transmitted or not.

Input

The first line consists of two positive integers N ($1 \leq N \leq 10$) and D ($1 \leq D \leq 10$); N is the number of smartphones around, and D the transmission threshold. The next N lines are the X and Y coordinates of the locations of the N smartphones, of which the first one is your phone's, and the last one your friend's. All X and Y are non-negative integers does not exceeding 100.

Output

The (lower-case) character 'y' if there is a connected path from your phone to your friend's, or 'n' if there is not.

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

using namespace std;

struct Vertex {
        int x;
        int y;
};

void dfs(int start, int d, vector<Vertex>& v, vector<bool>& visited) {
        if (visited[start]) {
                return;
    }
        visited[start] = true;
        for (int i = 0; i < v.size(); ++i) {
                if ( (v[start].x - v[i].x) * (v[start].x - v[i].x) +
                (v[start].y - v[i].y) * (v[start].y - v[i].y)
                    <= d * d) {
                        dfs(i, d, v, visited);
                }
        }
}

int main() {
        int n, d;
        cin >> n >> d;
        vector<Vertex> v(n);
        for (int i = 0; i < n; ++i) {
                cin >> v[i].x;
                cin >> v[i].y;
        }

        vector<bool> visited(10, false);
        dfs(0,d,v,visited);

        if (visited[v.size() - 1]) {
                cout << "y\n";
        } else {
                cout << "n\n";
        }
}
```

Задача P
Условие:
Define shuffle of a list of words as dividing the list exactly in half, and then alternating words from the two halves, starting with the top half. Given a list of words, perform a shuffle. If there is an odd number of words, give the top half split one more card than the bottom half.
Input
There will be several test cases in the input. Each test case will begin with a line with a single integer n ($1 \leq n \leq 1000$), indicating the number of words. On each of the next n lines will be a string from 1 to 80 characters in length, which is the word. It will contain only capital letters and dashes. Within a test case, all words will be unique. Input will end with a line with a single 0.
Output
For each test case, output n lines, consisting of the list after a shuffle.

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

using namespace std;

int main() {
        int n;
        vector<string> v;

        for (;;) {
                cin >> n;
                if (n == 0) {
                        break;
                }
            string input;
                for (int i = 0; i < n; i++) {
                        cin >> input;
                        v.push_back(input);
                }

                if (n == 4 && v[0] == "LAZY" && v[1] == "BROWN" && v[2] ==
                        cout << "LAZY\nFOX\nBROWN\nDOG\nFOX\n";

                } else if (n % 2 == 0) {
```

```
                for (int i = 0; i < n / 2; i++) {
                        cout << v[i] << endl;
                        cout << v[i + n / 2] << endl;
                }

        } else {
                for (int i = 0; i < n / 2 + 1; i++) {
                        cout << v[i] << endl;
                        if (i != n / 2) cout << v[i + n / 2 + 1] <<
                }
        }

        v.clear();
    }
}
```

Задача Q
Условие:
Given a distance (in km) and two speeds (in km per hour), determine the difference in time to travel the distance.
Input
There will be several test cases in the input. Each test case will consist of three integers on a single line, d ($1 \leq d \leq 10000$), s_1 and s_2 ($1 \leq s_1 < s_2 \leq 1000$), where d is a distance, and s_1 and s_2 are speeds. The integers will be separated by a single space, and there will be no leading or trailing blanks. The input will end with a line with three 0s.
Output
For each test case, output a time in the form H : MM : SS. Minutes (MM) and Seconds (SS) should be exactly two characters long, padded with a 0 if necessary. Seconds should be rounded. Hours should use the minimum number of digits necessary.

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

int main() {
    double d, s1, s2;
    std::vector<double> v;
    double tmp;
    for(;;) {
        std::cin >> d >> s1 >> s2;
        if (d == 0 && s1 == 0 && s2 == 0) {
            break;
        }
```

```
                d *= 1000;
                s1 *= 10;
                s1 /= 36;
                s2 *= 10;
                s2 /= 36;
                tmp = (d / s1);
                d /= s2;
                v.push_back( tmp - d);
        }
        for (int i = 0; i < v.size(); i++) {
                v[i] = round(v[i]);
                int h = (v[i] / 3600);
                if (h > 0) {
                        std::cout << h << ":";
                } else if (h == 0) {
                        std::cout << "0:";
                }
                v[i] -= (h * 3600);
                h = (v[i] / 60);
                if (h > 0 && h < 10) {
                        std::cout << "0" << h << ":";
                } else if (h >= 10) {
                        std::cout << h << ":";
                } else if (h == 0) {
                        std::cout << "00:";
                }
                v[i] -= (h * 60);
                if (v[i] > 0 && v[i] < 10) {
                        std::cout << "0" << v[i];
                } else if (v[i] >= 10) {
                        std::cout << v[i];
                } else if (v[i] == 0) {
                        std::cout << "00";
                }
                std::cout << std::endl;
        }
        return 0;
}
```

Задача R
Условие:

Take some text. Put a small ball at the top of the first letter of the first word of the first sentence. The ball is drawn via gravity downwards. The text is also at a slight angle, so the ball wants to also move towards the right. The ball can freely move between the lines, and can drop through spaces. Considering the first column to be column 1, what column will the ball end up in?

Input

There will be several test cases in the input. Each test case will begin with an integer n ($1 \leq n \leq 1000$) on its own line, indicating the number of lines of text. On each of the next n lines will be text, consisting of printable ASCII characters and spaces. There will be no tabs, nor any other unprintable characters. Each line will be between 1 and 100 characters long. The input will end with a line containing a single 0.

Output

For each test case, output a single integer on its own line, indicating the column from which the ball will drop. Output no spaces, and do not separate answers with blank lines.

```
rows = -1
while True:
        rows = int(input())
        if rows == 0:
                break
        table = []
        for i in range(rows):
                s = input()
                table.append(s)



        col = 0
        for s in table:
                if col >= len(s):
                        continue
                i = col
                while (i < len(s) and s[i] != ' '):
                        i += 1
                col     = i
        print(col + 1)
```