Московский Авиационный Институт
(национальный исследовательский университет)

Отчет по индивидуальному учебному плану

Алгоритмы на графах

Студенты:
Макаров Никита
Якименко Антон

Руководитель:
Зайцев В.Е.

Содержание

1 Личные отчеты

Отчет о работе студента Макарова Н.А. по индивидуальному учебному плану в V-VI семестрах 2014-2015 учебного года.

№	Дата	Контест	Место проведения	Кол-во участников	Решено задач	Задач на участника
1	18.09.2014	Codeforces Round 267 Div 2	Дом	1	2	2
2	21.09.2014	Codeforces Round 268 Div 2	Дом	1	2	2
3	22.09.2014	Codeforces Отборочный контест СГАУ на 1/4 ACM-ICPC	Дом	1	3	3
4	25.09.2014	Codeforces Training S02E03	МАИ	3	3	1
5	28.09.2014	Codeforces Round 270 Div 2	Дом	1	2	2
6	02.10.2014	Codeforces Training S02E04	МАИ	3	2	0.66
7	05.10.2014	XV Открытая Всесибирская Олимпиада по Программированию	МАИ	3	1	0.33
8	09.10.2014	Codeforces Training S02E05	МАИ	3	4	1.33
9	16.10.2014	Codeforces Round 273 Div 2	Дом	1	2	2
10	17.10.2014	Codeforces Training S02E06	МАИ	3	0	0
11	18.10.2014	Codeforces Тренировка СПбГУ графы и DFS	Дом	3	2	0.66
12	19.10.2014	OpenCup GP of SPb. Div 2	МАИ	3	1	0.33
13	20.10.2014	Codeforces Round 274 Div 2	МАИ	1	3	3
14	23.10.2014	Codeforces Самарский Аэрокосмический Лицей тренировка №1	Дом	2	1	0.5
15	23.10.2014	Codeforces ACM, NEERC, Восточный четвертьфинал	Дом	3	4	1.33
16	24.10.2014	Codeforces Round 275 Div 2	Дом	1	1	1
17	25.10.2014	Codeforces ACM, NEERC, Южный четвертьфинал	Дом	3	3	1
18	26.10.2014	ACM-ICPC 1/4 Final	МГУ	3	3	1
19	30.10.2014	Codeforces Training S02E07	МАИ	3	2	0.66
20	01.11.2014	Codeforces Crypto Cup	Дом	3	9	3
21	02.11.2014	OpenCup GP of Siberia Div 2	МАИ	2	3	1.5
22	06.11.2014	Codeforces Training S02E08	МАИ	3	2	0.66
23	13.11.2014	Codeforces Training S02E09	МАИ	3	3	1
24	15.11.2014	Codeforces Олимпиада школьников Нижегородской области	Дом	3	3	1
25	16.11.2014	OpenCup GP of Central Europe. Div 2	МАИ	3	1	0.33
26	20.11.2014	Codeforces Training S02E10	МАИ	3	3	1
27	23.11.2014	OpenCup GP of Europe Div 2	МАИ	3	5	1.66
28	14.12.2014	OpenCup GP of Peterhof Div 2	МАИ	3	1	0.33
29	01.02.2015	OpenCup GP of Japan Div 2	МАИ	3	4	1.33
30	08.02.2015	OpenCup Northern GP Div 2	МАИ	3	2	0.66
31	15.02.2015	OpenCup GP of Karelia Div 2	МАИ	3	4	1.33

Продолжение таблицы.

№	Дата	Контест	Место проведения	Кол-во участников	Решено задач	Задач на участника
32	22.02.2015	OpenCup GP of Udmurtia Div 2	МАИ	3	4	1.33
33	01.03.2015	OpenCup GP of China Div 2	МАИ	3	1	0.33
34	08.02.2015	OpenCup GP of Tatarstan Div 2	МАИ	3	1	0.33
35	07.03.2015	VK Cup 2015 Квалификация	Дом	2	2	1
36	21.03.2015	VK Cup 2015 Раунд 1	Дом	2	2	1
37	29.03.2015	OpenCup Gp of America Div 2	МАИ	3	4	1.33
38	18.04.2015	Vekua Cup Личный этап	МФТИ-1С	1	1	1
39	19.04.2015	Vekua Cup Командный этап	МФТИ-1С	3	3	1
40	26.04.2015	OpenCup GP of Ural Div 2	МАИ	2	2	1
41	31.05.2105	Mail.ru RCC Квалификация	Дом	1	1	1

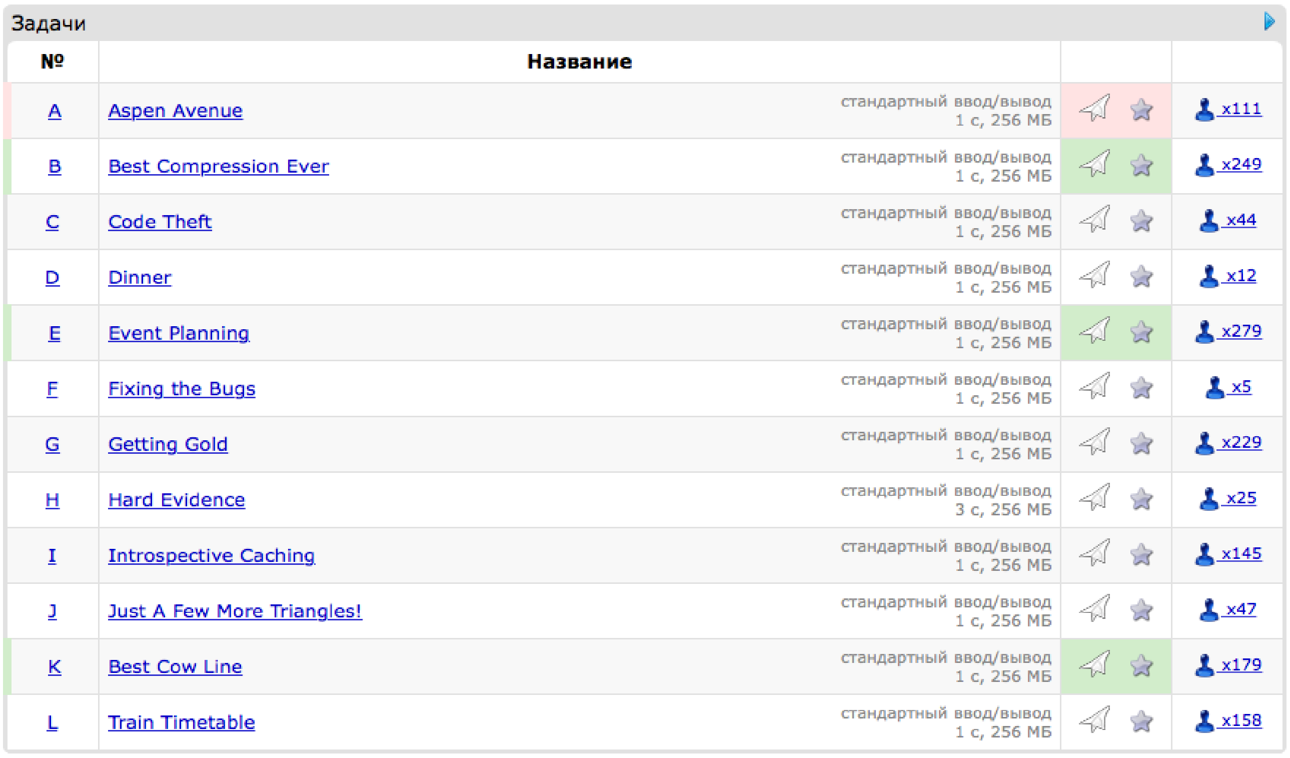
Итого: 41 контест, 47 решенных задач.

Отчет о работе студента Якименко А.В. по индивидуальному учебному плану в V-VI семестрах 2014-2015 учебного года.

2 Журнал по командным контестам

2.1 Codeforces Training S02E03

Результаты



Задачи

№	Название			
A	Aspen Avenue	стандартный ввод/вывод 1 с, 256 МБ	🟥	x111
B	Best Compression Ever	стандартный ввод/вывод 1 с, 256 МБ	🟩	x249
C	Code Theft	стандартный ввод/вывод 1 с, 256 МБ		x44
D	Dinner	стандартный ввод/вывод 1 с, 256 МБ		x12
E	Event Planning	стандартный ввод/вывод 1 с, 256 МБ	🟩	x279
F	Fixing the Bugs	стандартный ввод/вывод 1 с, 256 МБ		x5
G	Getting Gold	стандартный ввод/вывод 1 с, 256 МБ		x229
H	Hard Evidence	стандартный ввод/вывод 3 с, 256 МБ		x25
I	Introspective Caching	стандартный ввод/вывод 1 с, 256 МБ		x145
J	Just A Few More Triangles!	стандартный ввод/вывод 1 с, 256 МБ		x47
K	Best Cow Line	стандартный ввод/вывод 1 с, 256 МБ	🟩	x179
L	Train Timetable	стандартный ввод/вывод 1 с, 256 МБ		x158

Задача B - Best Compression Ever

Being educated in Computer Science and Mathematics is not always easy. Especially not if you have “friends” who repeatedly insist on showing you their new “proofs” that P equals NP, that the Riemann Hypothesis is true, and so on.

One of your friends recently claims to have found a fantastic new compression algorithm. As an example of its amazing performance, your friend has told you that every file in your precious collection of random bit strings after compression would be at most b bits long! Naturally, you find this a bit hard to believe, so you want to determine whether it is even theoretically possible for this to be true.

Your collection of random bit strings consists of N files, no two of which are identical, and each of which is exactly 1000 bits long.

Input specifications

The input consists of two integers N ($1 \leq N \leq 10^{15}$) and b ($0 \leq b \leq 50$), giving the number of files in your collection and the maximum number of bits a compressed file is allowed to have.

Output specifications

Output a line containing either “yes” if it is possible to compress all the N files in your collection into files of size at most b bits, or “no” otherwise.

Sample input 1	**Sample output 1**
`13 3`	`yes`

Sample input 2	**Sample output 2**
`1 0`	`yes`

Sample input 3	**Sample output 3**
`31415926535897 40`	`no`

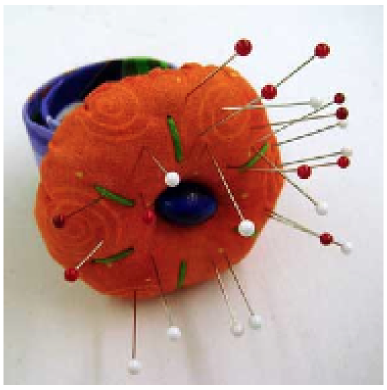
Алгоритм

Решение довольно простое. Можно заметить, что если логарифм по основаню 2 числа n меньше или равен b, то ответ yes, иначе ответ no.

Исходный код

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

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);

    unsigned long long n;
    int b;
    cin >> n >> b;

    if ((int)log2((double)n) <= b) {
        cout << "yes" << endl;
    }
    else {
        cout << "no" << endl;
    }

    return 0;
}
```

Задача E - Event Planning



As you didn't show up to the yearly general meeting of the Nordic Club of Pin Collectors, you were unanimously elected to organize this years excursion to Pin City. You are free to choose from a number of weekends this autumn, and have to find a suitable hotel to stay at, preferably as cheap as possible.

You have some constraints: The total cost of the trip must be within budget, of course. All participants must stay at the same hotel, to avoid last years catastrophe, where some members got lost in the city, never being seen again.

Input specifications

The first line of input consists of four integers: $1 \leq N \leq 200$, the number of participants, $1 \leq B \leq 500000$, the budget, $1 \leq H \leq 18$, the number of hotels to consider, and $1 \leq W \leq 13$, the number of weeks you can choose between. Then follow two lines for each of the H hotels. The first gives $1 \leq p \leq 10000$, the price for one person staying the weekend at the hotel. The second contains W integers, $0 \leq a \leq 1000$, giving the number of available beds for each weekend at the hotel.

Output specifications

Output the minimum cost of the stay for your group, or "`stay home`" if nothing can be found within the budget.

Sample input 1	Sample output 1

3 1000 2 3
200
0 2 2
300
27 3 20
	900

Sample input 2	Sample output 2

5 2000 2 4
300
4 3 0 4
450
7 8 0 13
	stay home

Алгоритм
????????????????????

Исходный код

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

#define ll long long
using namespace std;

int main () {
    ll N,B,H,W,p,a;
    ll min_cost = 5000000;

    cin >> N >> B >> H >> W;

    for (ll i = 0; i < H; i++) {
        cin >> p;
            for (ll k = 0; k < W; k++) {
                cin >> a;
                if((a >= N) && (p * N <= B) && (p * N <= min_cost))
                    min_cost = p * N;
            }
    }
    if(min_cost < 5000000)
        cout << min_cost << endl;
    else cout << "stay home" << endl;


    return 0;
}
```

Задача K - Best Cow Line

FJ is about to take his N (1 <= N <= 30,000) cows to the annual "Farmer of the Year" competition. In this contest every farmer arranges his cows in a line and herds them past the judges.

The contest organizers adopted a new registration scheme this year: simply register the initial letter of every cow in the order they will appear (e.g., If FJ takes Bessie, Sylvia, and Dora in that order, he just registers BSD). After the registration phase ends, every group is judged in increasing lexicographic order (i.e., alphabetical order) according to the string of the initials of the cows' names.

FJ is very busy this year and has to hurry back to his farm, so he wants to be judged as early as possible. He decides to rearrange his cows, who have already lined up, before registering them.

FJ marks a location for a new line of the competing cows. He then proceeds to marshal the cows from the old line to the new one by repeatedly sending either the first or last cow in the (remainder of the) original line to the end of the new line. When he's finished, FJ takes his cows for registration in this new order.

Given the initial order of his cows, determine the least lexicographic string of initials he can make this way.

Input
* Line 1: A single integer: N
* Lines 2..N+1: Line i+1 contains a single initial ('A'..'Z') of the cow in the i-th position in the original line

Output
The least lexicographic string he can make. Every line (except perhaps the last one) contains the initials of 80 cows ('A'..'Z') in the new line.

6 A C D B C B	ABCBCD

Алгоритм
???????????????????

Исходный код

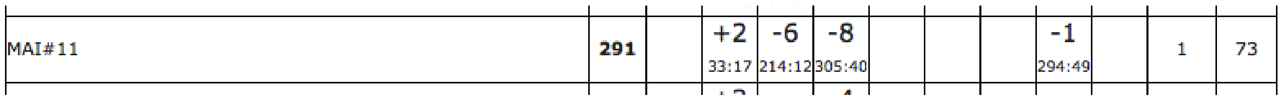
```cpp
#include <iostream>
#include <cmath>
#include <vector>
#include <stack>
#include <algorithm>

using namespace std;

int main() {
    int n;
    cin >> n;
    vector<char> cows(n);
    char symb;

    for (int i = 0; i < n; i++) {
        cin >> symb;
        cows[i] = symb;
    }

    vector<char> newLine;
    int cowsInOldLine = n;
    int begin = 0;
    int end    = n - 1;
    int tempBegin = begin;
    int tempEnd    = end;

    while (cowsInOldLine) {
        tempBegin = begin;
        tempEnd    = end;
        if (cows[begin] == cows[end]) {
            while (cows[tempBegin] == cows[tempEnd]) {
                tempBegin++;
                tempEnd--;
                if (tempBegin > tempEnd || tempBegin == tempEnd) {
                    tempBegin = begin;
                    tempEnd = end;
                    break;
                }
            }
        }
        if (cows[tempBegin] < cows[tempEnd]) {
            newLine.push_back(cows[begin]);
            begin++;
        }
        else {
            newLine.push_back(cows[end]);
            end--;
        }
        cowsInOldLine--;
    }
    for (int i = 0; i < n; i++) {
        cout << newLine[i];
        if ((i + 1) % 80 == 0) cout << endl;
    }
    cout << endl;
    return 0;
}
```

2.2 Codeforces Training S02E04

у Антона

2.3 XV Открытая Всесибирская олимпиада по программированию им И.В. Поттосина

Результаты



MAI#11	**291**		+2 33:17	-6 214:12	-8 305:40				-1 294:49		1	73

Задача 2 - Копировальный аппарат

Вам необходимо реализовать функцию автоопределения типа бумаги для копировального аппарата. При использовании этой функции аппарат сканирует документ для определения подходящего типа бумаги из доступных. Необходимо по отсканированному изображению и списку доступных типов бумаги определить, какой из них вместит изображение.

Стороны бумаги параллельны сторонам области сканирования. Левый верхний угол бумаги совмещен с левым верхним углом области сканирования. Бумагу нельзя поворачивать.

Формат входного файла

В первой строке входного файла записаны через пробел целые числа U и V — размеры области сканирования по вертикали и горизонтали ($1 \leq U, V \leq 100$). В следующих U строках описывается область сканирования. В каждой строке содержится ровно V символов. Часть изображения обозначается символом '#'. Отсутствие изображения обозначается символом '.'. Гарантируется, что существует хотя бы один символ '#'. Расположение области сканирования совпадает с расположением в исходном файле. Левый верхний угол соответствует первому символу первой строки.

В следующей строке записано целое число N — количество доступных типов бумаги ($1 \leq N \leq 10$). В следующих N строках описываются типы бумаги. В каждой строке записано через пробел по два целых числа y и x — размеры по вертикали и горизонтали ($1 \leq y \leq U$, $1 \leq x \leq V$).

Формат выходного файла

В выходной файл необходимо вывести одно целое число — номер подходящего типа бумаги. Типы нумеруются в том порядке, в каком они заданы во входном файле, начиная с единицы. Если подходящих типов несколько, требуется вывести тип с минимальным номером. Гарантируется, что существует хотя бы один подходящий тип бумаги.

Примеры

input.txt	output.txt
4 5 ....#.. .#... ..... 3 2 3 4 3 3 4	2
4 4 .... .#.. #... .... 3 2 2 2 3 3 2	3

Алгоритм

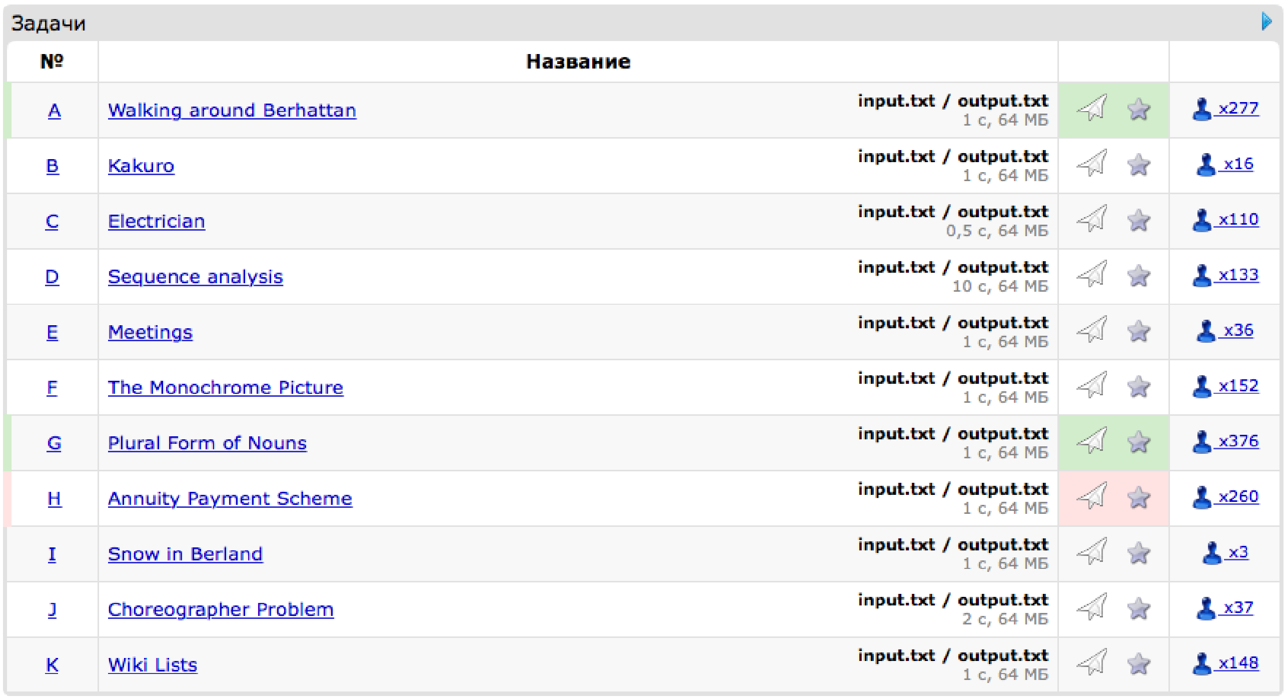
Задача на реализацию. Нужно считать входные данные в двумерный массив символов, затем пройтись по всем элементам и запомнить наибольшие позиции, на которых находятся решетки.

Исходный код

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

#include <sstream>
#include <fstream>

#define LL  long long

using namespace std;

int main() {
    ifstream in;
    ofstream out;
    in.open("input.txt");
    out.open("output.txt");

    LL a, b;
    in >> a >> b;
    char pic[a + 1][b + 1];
    LL XMax = 0;
    LL YMax = 0;

    for (LL i = 1; i <= a; i++) {
        for (LL j = 1; j <= b; j++) {
            in >> pic[i][j];
            if (pic[i][j] == '#' && j > XMax) XMax = j;
            if (pic[i][j] == '#' && i > YMax) YMax = i;
        }
        in.get();
    }
    LL n, y, x;
    in >> n;
    for (LL i = 1; i <= n; i++) {
        in >> y >> x;
        if (y >= YMax && x >= XMax) {
            out << i << endl;
            return 0;
        }
    }
    in.close();
    out.close();

    return 0;
}
```

2.4 Codeforces Training S02E05

Результаты



Задачи

№	Название			
A	Walking around Berhattan	input.txt / output.txt 1 с, 64 МБ		x277
B	Kakuro	input.txt / output.txt 1 с, 64 МБ		x16
C	Electrician	input.txt / output.txt 0,5 с, 64 МБ		x110
D	Sequence analysis	input.txt / output.txt 10 с, 64 МБ		x133
E	Meetings	input.txt / output.txt 1 с, 64 МБ		x36
F	The Monochrome Picture	input.txt / output.txt 1 с, 64 МБ		x152
G	Plural Form of Nouns	input.txt / output.txt 1 с, 64 МБ		x376
H	Annuity Payment Scheme	input.txt / output.txt 1 с, 64 МБ		x260
I	Snow in Berland	input.txt / output.txt 1 с, 64 МБ		x3
J	Choreographer Problem	input.txt / output.txt 2 с, 64 МБ		x37
K	Wiki Lists	input.txt / output.txt 1 с, 64 МБ		x148

Задача A - Walking around Berhattan

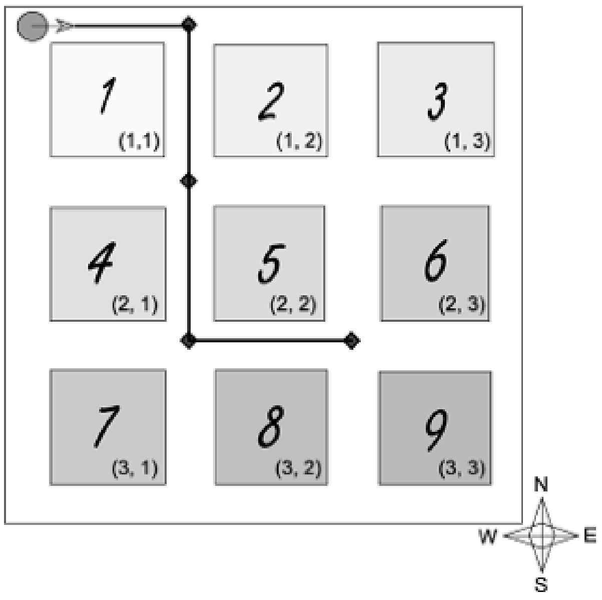
As you probably know, Berhattan is a district of Berland's largest city and it consists of equal square blocks. There are n block lines in the east-west direction and m block lines in the south-north direction. The map shows Berhattan as a rectangle with n rows and m columns, so there are $n \times m$ blocks in total.

There are $n+1$ streets running parallel in the east-west direction (horizontally), and there are $m+1$ avenues running parallel in the south-north direction (vertically). Streets and avenues split the district into blocks and separate Berhattan from other districts of Berland. Each block in Berhattan is characterized by its *beauty* b_{ij}.

A pedestrian can walk only along streets and avenues. When the pedestrian walks along any of four sides of a block, we say he passes the block. Every time the pedestrian passes a block his *satisfaction* is increased by b_{ij}. If the pedestrian has already passed the block one or more times his satisfaction is increased only by $b_{ij}/2$ rounded down when he passes the block again.

You are given the map of Berhattan with the information about the blocks' beauty and the pedestrian's path along the streets and avenues. The path is given as a string containing letters '`L`', '`R`' and '`M`', where '`L`' means a 90 degree left turn, '`R`' means a 90 degree right turn, and '`M`' means walking one block forward by a street or avenue. Facing the east, the pedestrian starts his path in the north-west corner of Berhattan having zero satisfaction level. His path can cross itself and go along the same streets or avenues several times. Pedestrian's satisfaction is increased every time he moves according to the rules described above.

Your task is to calculate the total satisfaction the pedestrian will get after finishing his route.



Picture of the sample test

Input

The first line of input contains two integers n and m ($1 \le n, m \le 100$), where n is a number of block lines in Berhattan running in the east-west direction, and m is a number of block lines in Berhattan running in the south-north direction. The following n lines contain m digits each. The j-th digit of the i-th line represents b_{ij} ($0 \le b_{ij} \le 9$) — the beauty of the corresponding block. The last line of input contains a path in the format specified above. The path consists of 1 up to 500 characters, inclusively. It is guaranteed that the given path doesn't go outside Berhattan.

Output

Print a single integer to the output — the total pedestrian's satisfaction.

Examples

input.txt	output.txt
3 3 123 456 789 MRMMLM	22

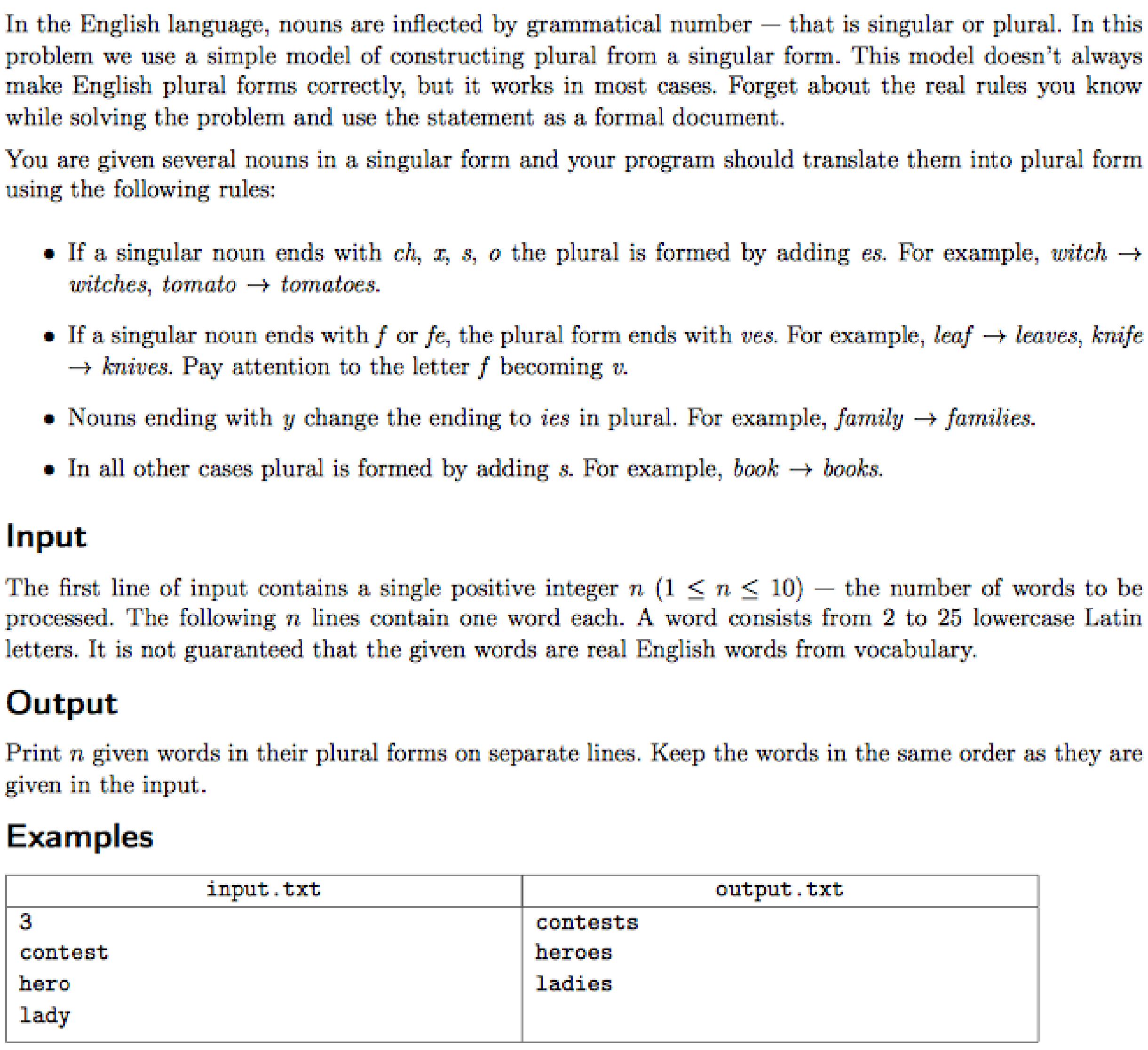
Алгоритм

???????????????????

Исходный код

```
#include <iostream>
#include <algorithm>
#include <iomanip>
#include <vector>
#include <sstream>
#include <fstream>
#define LL  long long

using namespace std;

enum dtype {
    UP,
    DOWN,
    LEFT,
    RIGHT,
};

int main() {
    ifstream in;
    ofstream out;
    in.open("input.txt");
    out.open("output.txt");

    LL n, m;
    LL answer = 0;
    in >> n >> m;
    char t;
    in.get();

    vector< vector<int> > map(n + 2, vector<int>(m + 2, 0));
    vector< vector<bool> > used(n + 2, vector<bool>(m + 2, false));

    for (LL i = 1; i <= n; i++) {
        for (LL j = 1; j <= m; j++) {
            t = in.get();
            map[i][j] = t - '0';
        }
        in.get();
    }

    int x = 1, y = 1;
    dtype dir = RIGHT; // last dir

    while ((t = in.get()) != EOF) {
        if (t == 'M') {
            if (dir == RIGHT) {
        answer += map[x][y];
                answer += map[x - 1][y];
              if (!used[x][y]) {
                  map[x][y] /= 2;
                  used[x][y] = true;
                }
                if (!used[x - 1][y]) {
                  map[x - 1][y] /= 2;
                  used[x - 1][y] = true;
                }
                y++;
            }
            else if (dir == LEFT) {
              answer += map[x][y - 1];
                answer += map[x - 1][y - 1];
              if (!used[x][y - 1]) {
                map[x][y - 1] /= 2;
                used[x][y - 1] = true;
              }
              if (!used[x - 1][y - 1]) {
                map[x - 1][y - 1] /= 2;
                  used[x - 1][y - 1] = true;
              }
                y--;
            }
            else if (dir == UP) {
                answer += map[x - 1][y - 1];
                answer += map[x - 1][y];
                if (!used[x - 1][y - 1]) {
                  map[x - 1][y - 1] /= 2;
                  used[x - 1][y - 1] = true;
                }
                if (!used[x - 1][y]) {
                  map[x - 1][y] /= 2;
                  used[x - 1][y] = true;
                }
                x--;
            }
            else if (dir == DOWN) {
                answer += map[x][y];
                answer += map[x][y - 1];
                if (!used[x][y]) {
                  map[x][y] /= 2;
                  used[x][y] = true;
                }
                if (!used[x][y - 1]) {
                  map[x][y - 1] /= 2;
                  used[x][y - 1] = true;
                }
                x++;
                }

        }
        else if (t == 'R') {
            if (dir == UP) dir = RIGHT;
            else if (dir == DOWN) dir = LEFT;
            else if (dir == LEFT) dir = UP;
            else dir = DOWN;
        }
        else if (t == 'L'){
            if (dir == UP) dir = LEFT;
            else if (dir == DOWN) dir = RIGHT;
            else if (dir == LEFT) dir = DOWN;
            else dir = UP;
        }
    }
    out << answer << endl;
    in.close();
    out.close();
    return 0;
}
```

Задача G - Plural Form of Nouns



In the English language, nouns are inflected by grammatical number — that is singular or plural. In this problem we use a simple model of constructing plural from a singular form. This model doesn't always make English plural forms correctly, but it works in most cases. Forget about the real rules you know while solving the problem and use the statement as a formal document.

You are given several nouns in a singular form and your program should translate them into plural form using the following rules:

- If a singular noun ends with *ch*, *x*, *s*, *o* the plural is formed by adding *es*. For example, *witch* → *witches*, *tomato* → *tomatoes*.
- If a singular noun ends with *f* or *fe*, the plural form ends with *ves*. For example, *leaf* → *leaves*, *knife* → *knives*. Pay attention to the letter *f* becoming *v*.
- Nouns ending with *y* change the ending to *ies* in plural. For example, *family* → *families*.
- In all other cases plural is formed by adding *s*. For example, *book* → *books*.

Input

The first line of input contains a single positive integer n $(1 \le n \le 10)$ — the number of words to be processed. The following n lines contain one word each. A word consists from 2 to 25 lowercase Latin letters. It is not guaranteed that the given words are real English words from vocabulary.

Output

Print n given words in their plural forms on separate lines. Keep the words in the same order as they are given in the input.

Examples

input.txt	output.txt
3 contest hero lady	contests heroes ladies

Алгоритм

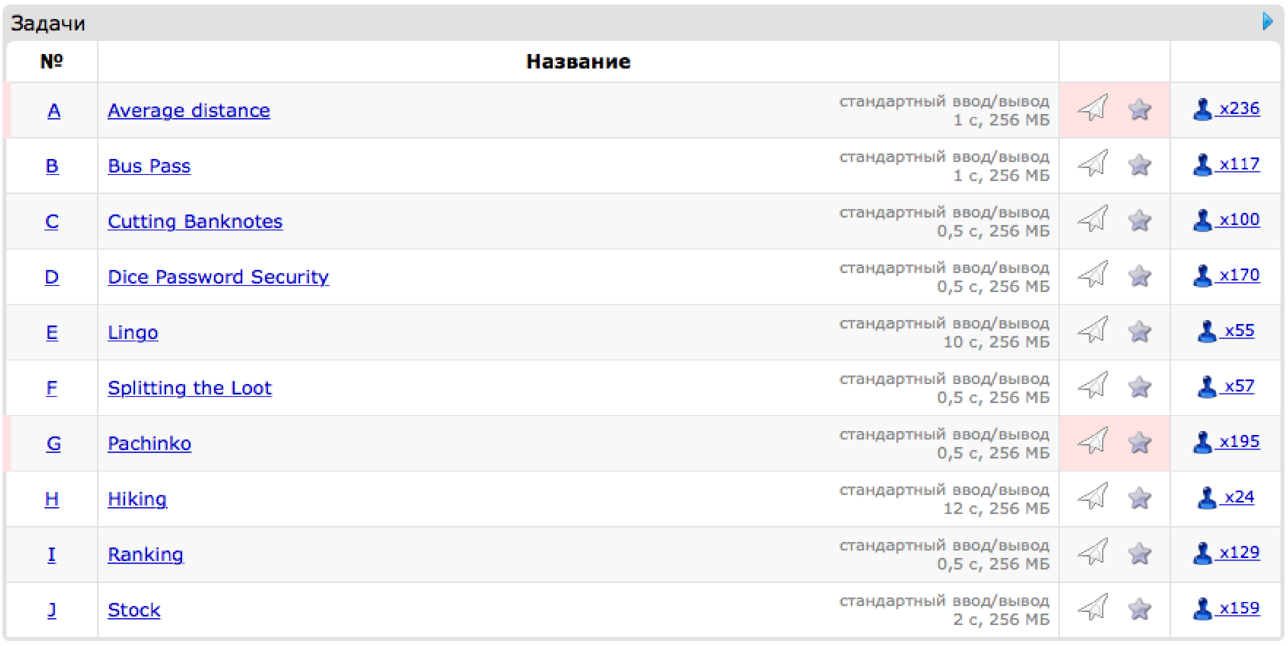
???????????????????

Исходный код

```cpp
#include <iostream>
#include <sstream>
#include <fstream>
#define LL  long long

using namespace std;

int main() {
    ifstream in;
    ofstream out;
    in.open("input.txt");
    out.open("output.txt");

    LL n;
    string s;

    in >> n;

    for (LL i = 0; i < n; i++) {
        in >> s;
        size_t l = s.size() - 1;
        if ((s[l] == 'h' && s[l - 1] == 'c') || s[l] == 's' || s[l] == 'x' || s[l] == 'o') {
            out << s;
            out << "es" << endl;
        }
        else if (s[l] == 'f') {
            for (size_t j = 0; j < l; j++) {
                out << s[j];
            }
            out << "ves" << endl;
        }
        else if (s[l] == 'e' && s[l - 1] == 'f') {
            for (size_t j = 0; j < l - 1; j++) {
                out << s[j];
            }
            out << "ves" << endl;
        }
        else if (s[l] == 'y') {
            for (size_t j = 0; j < l; j++) {
                out << s[j];
            }
            out << "ies" << endl;
        }
        else {
            out << s << "s" << endl;
        }
    }

    in.close();
    out.close();

    return 0;
}
```

2.5 Codeforces Training S02E06

Результаты

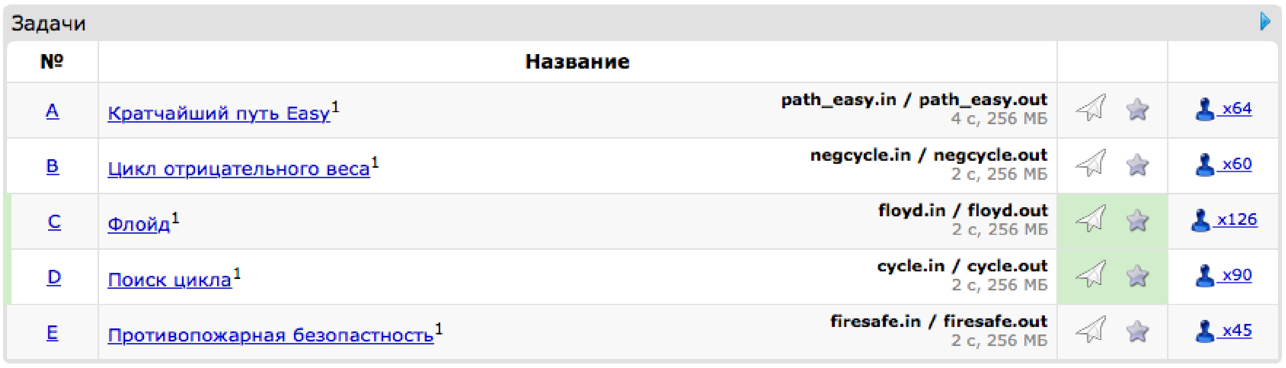


Задачи

№	Название			
A	Average distance	стандартный ввод/вывод 1 с, 256 МБ		x236
B	Bus Pass	стандартный ввод/вывод 1 с, 256 МБ		x117
C	Cutting Banknotes	стандартный ввод/вывод 0,5 с, 256 МБ		x100
D	Dice Password Security	стандартный ввод/вывод 0,5 с, 256 МБ		x170
E	Lingo	стандартный ввод/вывод 10 с, 256 МБ		x55
F	Splitting the Loot	стандартный ввод/вывод 0,5 с, 256 МБ		x57
G	Pachinko	стандартный ввод/вывод 0,5 с, 256 МБ		x195
H	Hiking	стандартный ввод/вывод 12 с, 256 МБ		x24
I	Ranking	стандартный ввод/вывод 0,5 с, 256 МБ		x129
J	Stock	стандартный ввод/вывод 2 с, 256 МБ		x159

2.6 Тренировка СПбГУ Поиск кратчайшего пути и DFS

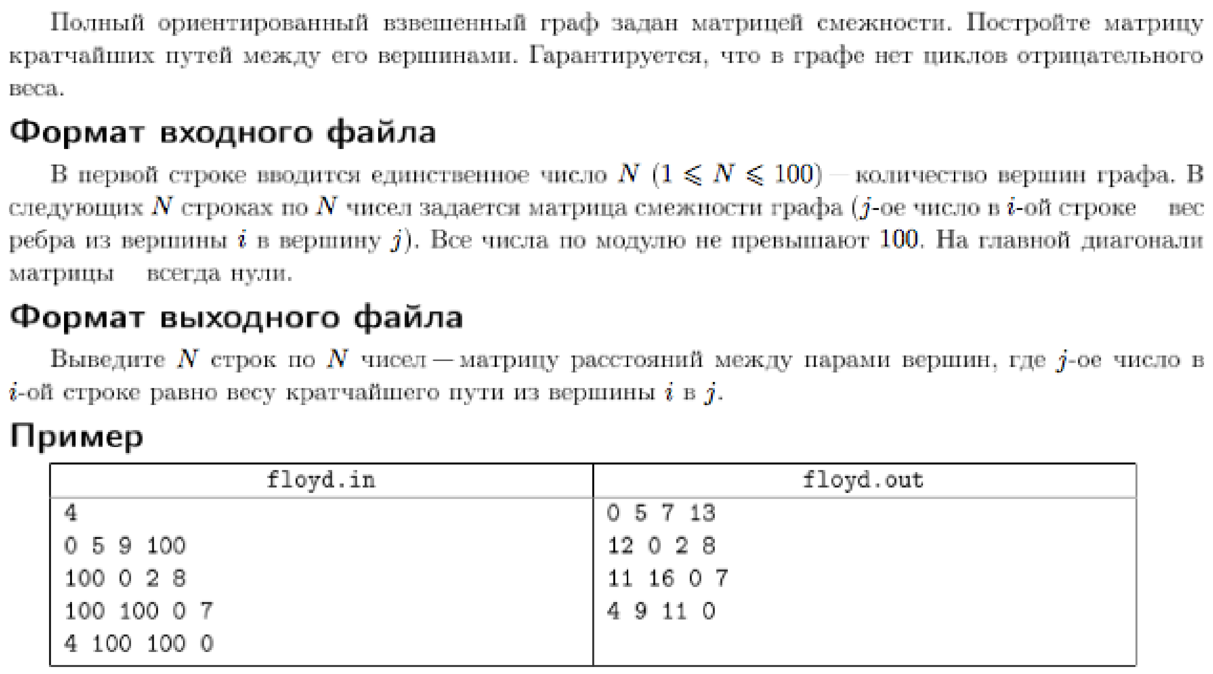
Результаты



Задачи

№	Название			
A	Кратчайший путь Easy[1]	path_easy.in / path_easy.out 4 с, 256 МБ		x64
B	Цикл отрицательного веса[1]	negcycle.in / negcycle.out 2 с, 256 МБ		x60
C	Флойд[1]	floyd.in / floyd.out 2 с, 256 МБ		x126
D	Поиск цикла[1]	cycle.in / cycle.out 2 с, 256 МБ		x90
E	Противопожарная безопастность[1]	firesafe.in / firesafe.out 2 с, 256 МБ		x45

Задача C - Флойд



Полный ориентированный взвешенный граф задан матрицей смежности. Постройте матрицу кратчайших путей между его вершинами. Гарантируется, что в графе нет циклов отрицательного веса.

Формат входного файла

В первой строке вводится единственное число N ($1 \leqslant N \leqslant 100$) — количество вершин графа. В следующих N строках по N чисел задается матрица смежности графа (j-ое число в i-ой строке — вес ребра из вершины i в вершину j). Все числа по модулю не превышают 100. На главной диагонали матрицы — всегда нули.

Формат выходного файла

Выведите N строк по N чисел — матрицу расстояний между парами вершин, где j-ое число в i-ой строке равно весу кратчайшего пути из вершины i в j.

Пример

floyd.in	floyd.out

4
0 5 9 100
100 0 2 8
100 100 0 7
4 100 100 0
	0 5 7 13
	12 0 2 8
	11 16 0 7
	4 9 11 0

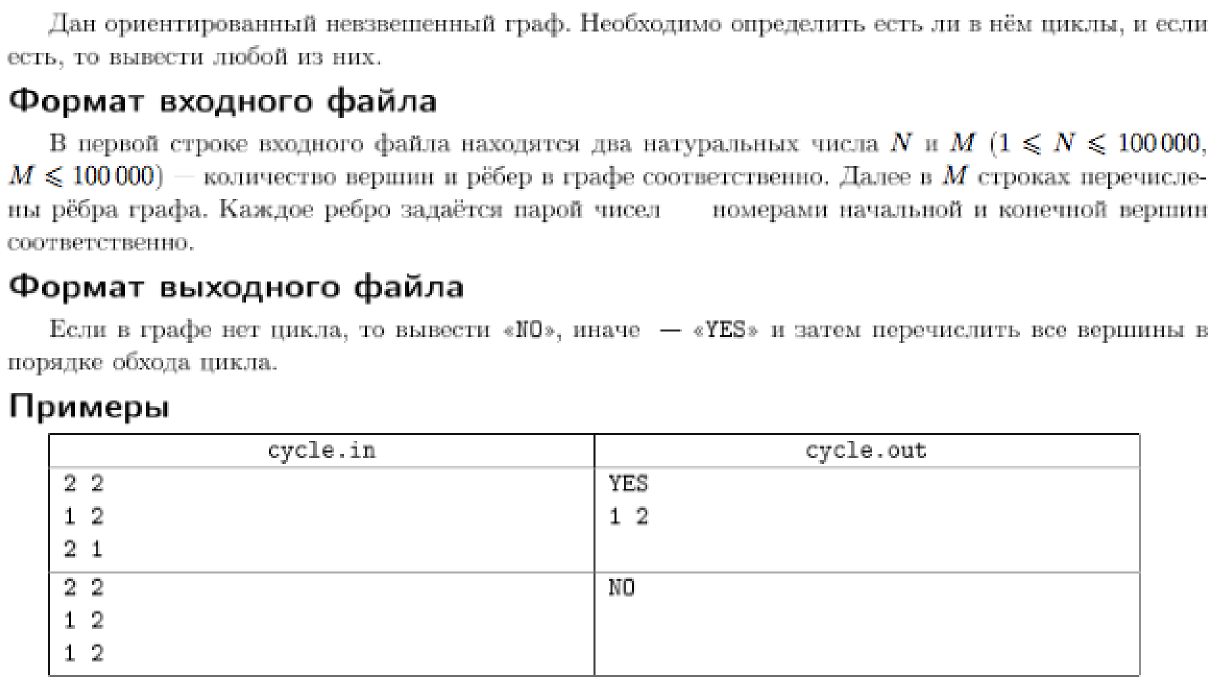
Алгоритм

В задаче требуется найти кратчайшие пути между всеми парами вершин и представить их матрицей смежности. Ее можно решить используя алгоритм Флойда-Уоршелла за $O(n^3)$.

Исходный код

```
#include <iomanip>
#include <iostream>
#include <algorithm>
#include <fstream>

using namespace std;

int main() {

    more_speed
    ifstream in("floyd.in");
    ofstream out("floyd.out");

    int n;
    in >> n;
    vector<vector<int> > m(n, vector<int>(n, 0));
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            in >> m[i][j];
        }
    }

    for (int k = 0; k < n; k++) {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                m[i][j] = min(m[i][j], m[i][k] + m[k][j]);
            }
        }
    }

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            out << m[i][j];
            if (j < n - 1) out << " ";
            else out << endl;
        }
    }

    in.close();
    out.close();

    return 0;
}
```

Задача D - Поиск цикла


Дан ориентированный невзвешенный граф. Необходимо определить есть ли в нём циклы, и если есть, то вывести любой из них.

Формат входного файла

В первой строке входного файла находятся два натуральных числа N и M ($1 \leqslant N \leqslant 100\,000$, $M \leqslant 100\,000$) — количество вершин и рёбер в графе соответственно. Далее в M строках перечислены рёбра графа. Каждое ребро задаётся парой чисел   номерами начальной и конечной вершин соответственно.

Формат выходного файла

Если в графе нет цикла, то вывести «NO», иначе — «YES» и затем перечислить все вершины в порядке обхода цикла.

Примеры

cycle.in	cycle.out

2 2
1 2
2 1
	YES
	1 2

2 2
1 2
1 2
	NO

Алгоритм

Задача решается поиском в глубину. Нужно сделать серию поисков в глубину, заходя в новую вершину будем красить ее в серый цвет, а выходя в черный. Если заходим в серую вершину, то цикл найден.

Исходный код

```cpp
#include <iomanip>
#include <iostream>
#include <algorithm>
#include <fstream>

vector<set<LL> > g;
vector<char> color;
vector<LL> p;
LL cycle_st, cycle_end;

bool dfs (LL v) {
    color[v] = 1;
    for (set<LL>::iterator i = g[v].begin(); i != g[v].end(); i++) {
        LL to = *i;
        if (color[to] == 0) {
            p[to] = v;
            if (dfs(to)) return true;
        }
        else if (color[to] == 1){
            cycle_st = to;
            cycle_end = v;
            return true;
        }
    }
    color[v] = 2;
    return false;
}

using namespace std;
int main() {

    more_speed
    ifstream in("cycle.in");
    ofstream out("cycle.out");

    LL n, m, f, t;
    in >> n >> m;
    g.resize(n);

    for (LL i = 0; i < m; i++) {
        in >> f >> t;
        g[f - 1].insert(t - 1);
    }

    p.assign(n, -1);
    color.assign(n, 0);
    cycle_st = -1;
    for (LL i = 0; i < n; i++) {
        if (dfs(i)) break;
    }
    if (cycle_st == -1) {
        out << "NO" << endl;
    }
    else {
        out << "YES" << endl;
        vector<LL> cycle;
        for (LL v = cycle_end; v != cycle_st; v = p[v]) {
            cycle.push_back(v);
        }
        cycle.push_back(cycle_st);
        reverse(cycle.begin(), cycle.end());
        for (size_t i = 0; i < cycle.size(); i++) {
            out << cycle[i] + 1 << " ";
        }
        out << endl;
    }
    in.close();
    out.close();

    return 0;
}
```

2.7 OpenCup GrandPrix of SPb.

Задача A - Барабашка

Это — задача с открытыми тестами. В ней нужно выбрать один из пяти предметов по текстовому описанию картинки.

«Барабашка» (исходное название «Geistesblitz») — настольная игра на реакцию. Далее приводятся правила игры для этой задачи. Будьте внимательны: эти правила немного отличаются от правил настольной игры.

В набор для игры входят пять предметов разных цветов — белое привидение по имени Барабашка, зелёная бутылка, серая мышка, синяя книга и красное кресло, — а также специальные карты. На каждой карте изображено ровно два предмета из этих пяти. Каждый из них также имеет один из перечисленных пяти цветов, возможно, отличающийся от правильного. При этом цвета этих двух предметов не совпадают. На карте могут присутствовать и какие-то другие объекты, но их цвета обязательно отличаются от пяти перечисленных.

Один ход в игре происходит так. Все предметы ставятся на стол, после чего из колоды достают очередную карту и показывают участникам. Если на карте присутствует предмет правильного цвета (того, которого этот предмет на самом деле), нужно схватить этот предмет. В противном случае нужно схватить тот предмет, которого нет на карте, и цвет которого на карте также не присутствует. Побеждает и берёт себе карту тот, кто раньше остальных схватил правильный предмет. Гарантируется, что все карты таковы, что правильный ответ существует и является единственным.

В этой задаче каждый тест состоит ровно из пяти предложений. Предложение — это текстовое описание одной карты на английском языке. Правильным ответом на такое предложение считается описание предмета, который надо схватить, видя эту карту.

Каждое предложение написано на английском языке и может содержать буквы английского алфавита, а также пробелы и символы «`'`», «`,`», «`-`» и «`.`» (ASCII-коды 39, 44, 45 и 46). Названия и правильные цвета предметов записываются так: «`white Barabashka`», «`blue book`», «`red chair`», «`gray mouse`» и «`green bottle`».

Пусть слово — это последовательность букв английского алфавита, ограниченная с обеих сторон концами строки или небуквенными символами. Тогда можно сформулировать следующие ограничения на предложения:

- В предложении встречается ровно два места вида «`цвет предмет`», где слово «`цвет`» — один из пяти перечисленных цветов, а слово «`предмет`» — название одного из пяти предметов.
- Ни одно из других слов предложения не совпадает с названиями и цветами предметов на столе.

Формат входных данных

Во вводе задано пять строк. Каждая строка содержит одно предложение длиной от 1 до 80 символов. Формат предложения описан в условии. Большие и маленькие буквы считаются различными.

Формат выходных данных

В ответ на каждое предложение выведите на отдельной строке два слова, разделив их пробелом — правильный цвет и название предмета, который нужно схватить. Большие и маленькие буквы считаются различными.

Пример

barabashka.in	barabashka.out

`A white Barabashka is staring at a gray bottle.`
`A red mouse is running from a green Barabashka and its shadows.`
`A gray book lies on a red chair's right arm.`
`Black smoke is rising from a blue bottle lying under a white chair.`
`A white mouse is trying to crawl out of a green bottle.`
	`white Barabashka`
	`blue book`
	`red chair`
	`gray mouse`
	`green bottle`

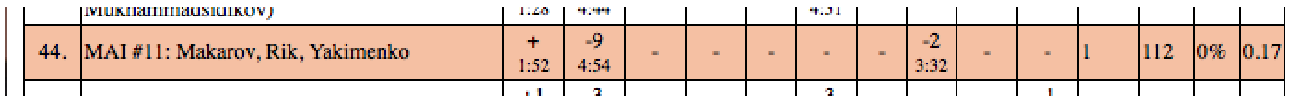
Алгоритм

Задача на реализацию. Нужно считать строки и каждой строке сопоставить правильное сочетание цвета и предмета по заданым в условии правилам. Сначала определим, какие сочетания уже имеются в предложении, затем проверим, есть ли среди них корректные, если есть, то это ответ, иначе нужно выбрать любое правильное сочетание.

Исходный код

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

bool isChar(char c) {
    return (c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z');
}

int num(string s) {
    if      (s == "white" || s == "barabashka") return 0;
    else if (s == "blue"  || s == "book")       return 1;
    else if (s == "red"   || s == "chair")      return 2;
    else if (s == "gray"  || s == "mouse")      return 3;
    else return 4;
}

int main() {
    ifstream in("barabashka.in");
    ofstream out("barabashka.out");

    string white = "white",
           blue  = "blue",
           red   = "red",
           gray  = "gray",
           green = "green";
    string barab = "barabashka",
           book  = "book",
           chair = "chair",
           mouse = "mouse",
           bottle = "bottle";

    string current;
    string firstColor, firstObject;
    string secondColor, secondObject;
    bool needFirstColor, needFirstObject;
    bool needSecondColor, needSecondObject;
    bool complete;


    for (int i = 0; i < 5; i++) {
        char cSymb = in.get();
        bool used[5] = {false};
        needFirstColor   = true;
        needFirstObject  = false;
        needSecondColor  = false;
        needSecondObject = false;
        complete         = false;

        while (cSymb != '0') {
            while (cSymb != ' ' && isChar(cSymb)) {
                current += tolower(cSymb);
                cSymb = in.get();
            }
            if (needFirstColor) {
                if (current == white)      { firstColor = white; needFirstObject = true; needFirstColor = false; }
                else if (current == blue)  { firstColor = blue;  needFirstObject = true; needFirstColor = false; }
                else if (current == red)   { firstColor = red;   needFirstObject = true; needFirstColor = false; }
                else if (current == gray)  { firstColor = gray;  needFirstObject = true; needFirstColor = false; }
                else if (current == green) { firstColor = green; needFirstObject = true; needFirstColor = false; }
            }
            else if (needFirstObject) {
                if (current == barab)       { firstObject = barab;  needSecondColor = true; needFirstObject = false; }
                else if (current == book)   { firstObject = book;   needSecondColor = true; needFirstObject = false; }
                else if (current == chair)  { firstObject = chair;  needSecondColor = true; needFirstObject = false; }
                else if (current == mouse)  { firstObject = mouse;  needSecondColor = true; needFirstObject = false; }
                else if (current == bottle) { firstObject = bottle; needSecondColor = true; needFirstObject = false; }
            }
            else if (needSecondColor) {
                if (current == white)      { secondColor = white; needSecondObject = true; needSecondColor = false; }
                else if (current == blue)  { secondColor = blue;  needSecondObject = true; needSecondColor = false; }
                else if (current == red)   { secondColor = red;   needSecondObject = true; needSecondColor = false; }
                else if (current == gray)  { secondColor = gray;  needSecondObject = true; needSecondColor = false; }
                else if (current == green) { secondColor = green; needSecondObject = true; needSecondColor = false; }
            }
            else if (needSecondObject) {
                if (current == barab)       { secondObject = barab;  complete = true; needSecondObject = false; }
                else if (current == book)   { secondObject = book;   complete = true; needSecondObject = false; }
                else if (current == chair)  { secondObject = chair;  complete = true; needSecondObject = false; }
                else if (current == mouse)  { secondObject = mouse;  complete = true; needSecondObject = false; }
                else if (current == bottle) { secondObject = bottle; complete = true; needSecondObject = false; }
            }
            if (complete) {
                cSymb = '0';
                used[num(firstColor)] = true;
                used[num(firstObject)] = true;
                used[num(secondColor)] = true;
                used[num(secondObject)] = true;

                if ((firstColor == white && firstObject == barab) ||
                    (secondColor == white && secondObject == barab)) { out << white << " " << "Barabashka" << endl; }

                else if ((firstColor == blue && firstObject == book) ||
                         (secondColor == blue && secondObject == book)) { out << blue << " " << book << endl; }

                else if ((firstColor == red && firstObject == chair) ||
                         (secondColor == red && secondObject == chair)) { out << red << " " << chair << endl; }

                else if ((firstColor == gray && firstObject == mouse) ||
                         (secondColor == gray && secondObject == mouse)) { out << gray << " " << mouse << endl; }

                else if ((firstColor == green && firstObject == bottle) ||
                         (secondColor == green && secondObject == bottle)) { out << green << " " << bottle << endl; }

                else {
                    for (int j = 0; j < 5; j++) {
                        if (!used[j]) {
                            if (j == 0) { out << white << " " << "Barabashka" << endl; }
                            else if (j == 1) { out << blue << " " << book << endl; }
                            else if (j == 2) { out << red << " " << chair << endl; }
                            else if (j == 3) { out << gray << " " << mouse << endl; }
                            else             { out << green << " " << bottle << endl; }
                            break;
                        }
                    }
                }


            }
            current.clear();
            if (cSymb != '\0') cSymb = in.get();
        }
    }
    return 0;
}
```

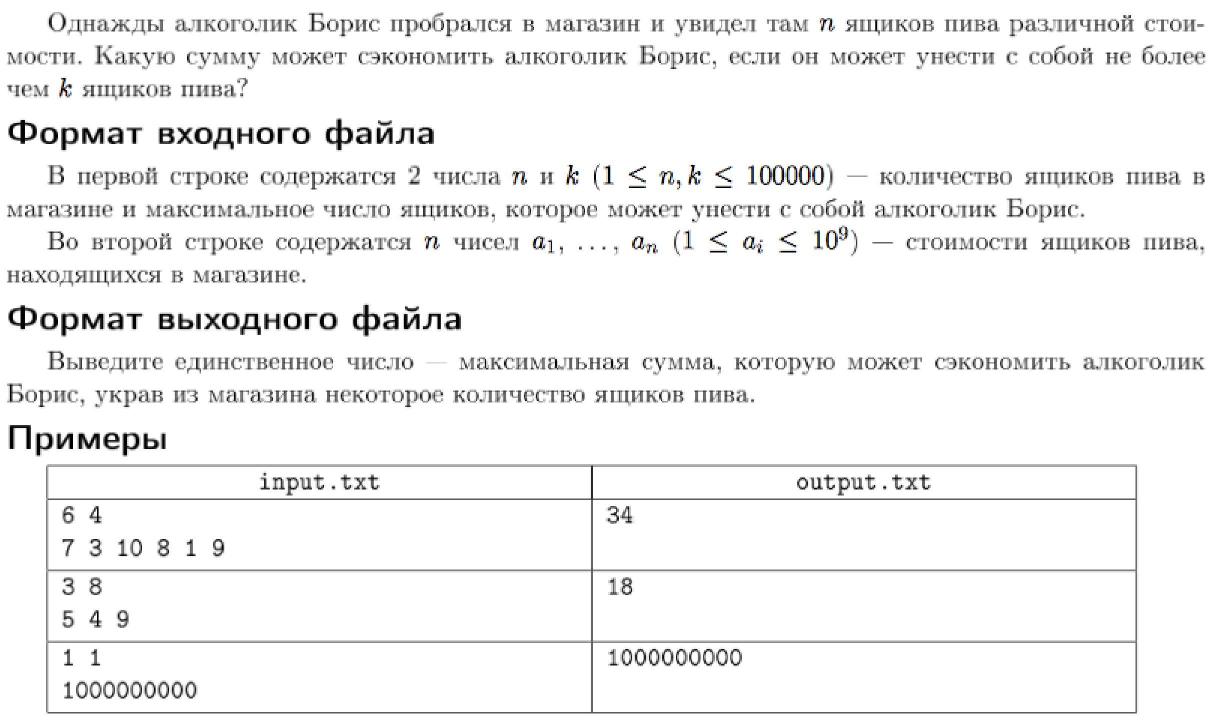
Результаты



44.	MAI #11: Makarov, Rik, Yakimenko	+ 1:52	-9 4:54	-	-	-	-	-	-2 3:32	-	-	1	112	0%	0.17

2.8 Самарский Международный Аэрокосмический Лицей, тренировка №1

Задача D - Пивной вор



Однажды алкоголик Борис пробрался в магазин и увидел там n ящиков пива различной стоимости. Какую сумму может сэкономить алкоголик Борис, если он может унести с собой не более чем k ящиков пива?

Формат входного файла

В первой строке содержатся 2 числа n и k ($1 \leq n, k \leq 100000$) — количество ящиков пива в магазине и максимальное число ящиков, которое может унести с собой алкоголик Борис.

Во второй строке содержатся n чисел $a_1, \ldots, a_n$ ($1 \leq a_i \leq 10^9$) — стоимости ящиков пива, находящихся в магазине.

Формат выходного файла

Выведите единственное число — максимальная сумма, которую может сэкономить алкоголик Борис, украв из магазина некоторое количество ящиков пива.

Примеры

input.txt	output.txt
6 4 7 3 10 8 1 9	34
3 8 5 4 9	18
1 1 1000000000	1000000000

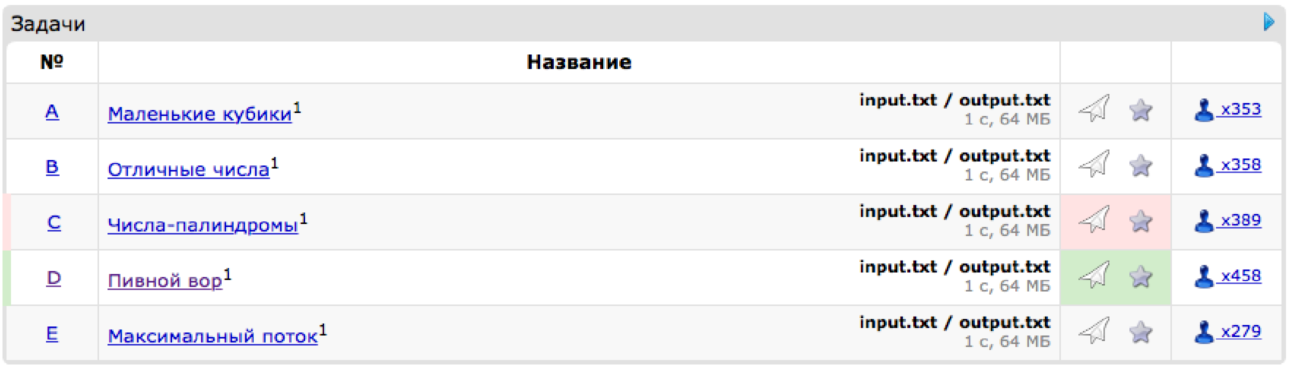
Алгоритм

В этой задаче нужно отсортировать массив стоимостей по убыванию и сложить первые k чисел. Это и будет ответом.

Исходный код

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

using namespace std;

int main() {
  long n, k;
  ifstream in("input.txt");
  ofstream out("output.txt");
  in >> n >> k;
  vector<long long> v(n);
  for (long i = 0; i < n; i++) {
    in >> v[i];
  }
  sort(v.begin(), v.end(), greater<long long>());
  long long answer = 0;
  for (long i = 0; i < k && i < v.size(); i++) {
    answer += v[i];
  }
  out << answer << endl;
  in.close();
  out.close();
  return 0;
}
```

Результаты

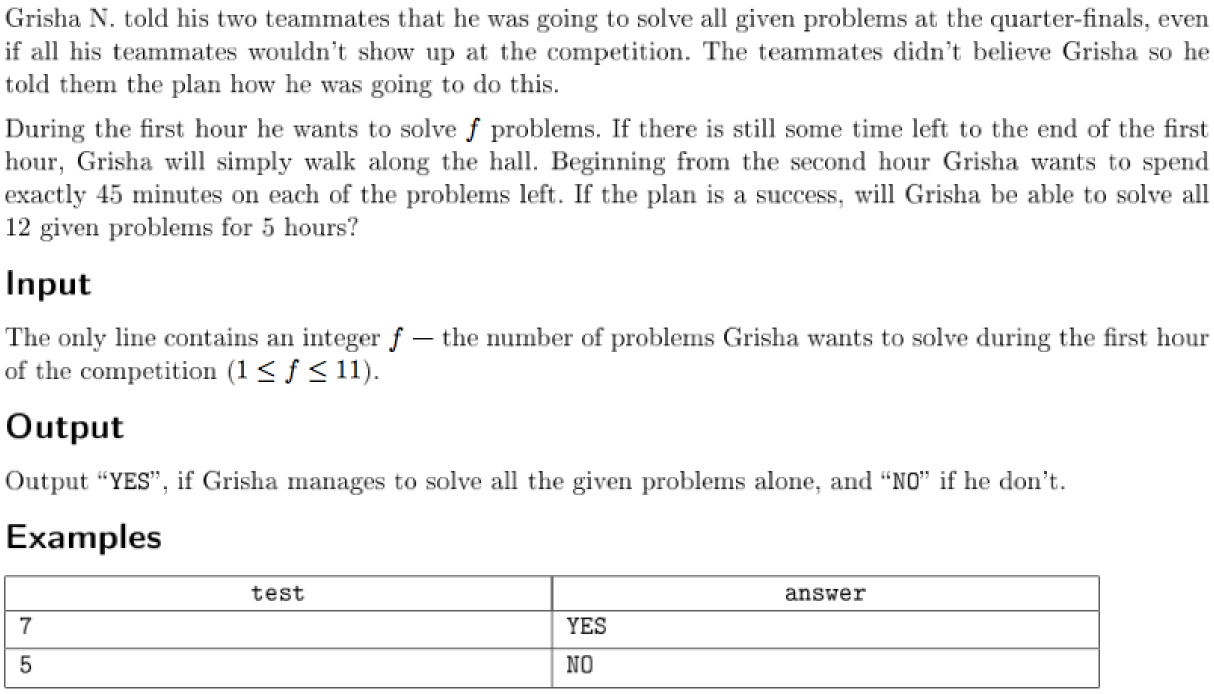


Задачи

№	Название			
A	Маленькие кубики[1]	input.txt / output.txt 1 с, 64 МБ		x353
B	Отличные числа[1]	input.txt / output.txt 1 с, 64 МБ		x358
C	Числа-палиндромы[1]	input.txt / output.txt 1 с, 64 МБ	🟥	x389
D	Пивной вор[1]	input.txt / output.txt 1 с, 64 МБ	🟩	x458
E	Максимальный поток[1]	input.txt / output.txt 1 с, 64 МБ		x279

2.9 Codeforces ACM-ICPC Восточный четвертьфинал

Задача A - About Grisha N.



Grisha N. told his two teammates that he was going to solve all given problems at the quarter-finals, even if all his teammates wouldn't show up at the competition. The teammates didn't believe Grisha so he told them the plan how he was going to do this.

During the first hour he wants to solve f problems. If there is still some time left to the end of the first hour, Grisha will simply walk along the hall. Beginning from the second hour Grisha wants to spend exactly 45 minutes on each of the problems left. If the plan is a success, will Grisha be able to solve all 12 given problems for 5 hours?

Input

The only line contains an integer f — the number of problems Grisha wants to solve during the first hour of the competition ($1 \leq f \leq 11$).

Output

Output "`YES`", if Grisha manages to solve all the given problems alone, and "`NO`" if he don't.

Examples

test	answer
7	YES
5	NO

Алгоритм

??????????????????????

Исходный код

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

using namespace std;

int main() {
  long n, k;
  ifstream in("input.txt");
  ofstream out("output.txt");
  in >> n >> k;
  vector<long long> v(n);
  for (long i = 0; i < n; i++) {
    in >> v[i];
  }
  sort(v.begin(), v.end(), greater<long long>());
  long long answer = 0;
  for (long i = 0; i < k && i < v.size(); i++) {
    answer += v[i];
  }
  out << answer << endl;
  in.close();
  out.close();
  return 0;
}
```

Задача D - Zhenya moves from the dormitory

After moving from his parents' place Zhenya has been living in the University dormitory for a month. However, he got pretty tired of the curfew time and queues to the shower room so he took a fancy for renting an apartment. It turned out not the easiest thing in the world to make a choice. One can live in a one bedroom apartment or in a two bedroom apartment, alone or share it with a friend. Zhenya can afford to rent an apartment of any type alone, but he can share only a two bedroom apartment. If two people share an apartment, each pays half of the rent. Every apartment has its own advantages like part of the town, floor, view from the windows, etc., which Zhenya is going to take into account to make a decision.

Besides that, his friends, he's ready to share an apartment with, also have certain advantages. For example, Igor is a good cook, Dima is tidy, Kostya is a good cook and at the same time can explain how to solve functional analysis problems. And do not forget that living alone has its own bright sides.

Zhenya has already prepared the list of suitable apartments and possible housemates. Zhenya has estimated in units the advantages of each apartment and each friend and also the advantages of living alone. Besides, he knows the maximum sum of money he and each of his friends is ready to pay for the apartment. Help Zhenya to make a decision.

Input

The first line contains three integers: the maximum sum Zhenya is ready to pay monthly, the advantages of living alone in a one bedroom apartment and the advantages of living alone in a two bedroom apartment.

The second line contains an integer n that is the number of Zhenya's friends ($0 \le n \le 256$). Next n lines describe the friends, two integers in every line: the maximum sum the corresponding friend is ready to pay monthly and the advantages of sharing an apartment with him.

The next line contains an integer m that is the number of suitable apartments ($1 \le m \le 256$). Next m lines describe the apartments, three integers in every line: the number of bedrooms in an apartment (1 or 2), monthly rent and the advantages of living there.

All the advantages are estimated in the same units and lie in the range from 0 to 100 000. All sums of money are in rubles and lie in the range from 1 to 100 000.

Output

Output the variant with maximum sum of advantages, Zhenya (and his friend in case of sharing apartments) can afford. If Zhenya should rent an apartment number i alone, output "`You should rent the apartment #i alone.`". If he should share an apartment number i with a friend j output "`You should rent the apartment #i with the friend #j.`". Friends and apartments are numbered from 1 in order they are given in the input. If there are several optimal alternatives, output any of them. If Zhenya can't afford to rent any apartment at all, output "`Forget about apartments. Live in the dormitory.`".

Examples

test

10000 50 70
1
10000 100
2
1 10000 200
2 30000 500

answer
You should rent the apartment #1 alone.
test

30000 0 1
1
10000 1001
3
1 20000 2000
2 30000 2000
2 10000 1001

answer
You should rent the apartment #3 with the friend #1.

Алгоритм
??????????????????????

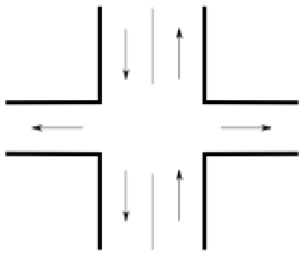
Исходный код

```cpp

#include <iostream>
#include <algorithm>
#include <vector>
#include <fstream>

using namespace std;

typedef struct {
    long money;
    long ad;
    int num;
}frd;

typedef struct {
    long rooms;
    long price;
    long ad;
    int num;
}apartment;

bool cmp_ap(const apartment &a1, const apartment &a2) {
    return a1.ad > a2.ad;
}

bool cmp_fr(const frd &f1, const frd &f2) {
    return f1.ad > f2.ad;
}

int find_good_ap(const vector<apartment> &a, long money1, long money2) {
    for (int i = 0; i < a.size(); i++) {
        long price_for_each = a[i].price / 2;
        if (a[i].price % 2 == 1) {
            price_for_each++;
        }
        if (money1 >= price_for_each && money2 >= price_for_each && a[i].rooms == 2)
            return i;
    }
    return -1;
}

int main() {
    long money, ad1, ad2;
    long n, m;
    cin >> money >> ad1 >> ad2;
    cin >> n;
    vector<frd> fr(n);
    for (int i = 0; i < n; i++) {
        cin >> fr[i].money >> fr[i].ad;
        fr[i].num = i + 1;
    }
    cin >> m;
    vector<apartment> ap(m);
    long max_ad_in_1room = -1, max_ad_in_2room = -1;
    int  in_1room_num = -1, in_2room_num = -1;
    bool can_buy_alone = false;
    for (int i = 0; i < m; i++) {
        cin >> ap[i].rooms >> ap[i].price >> ap[i].ad;
        ap[i].num = i + 1;
        if (ap[i].rooms == 1) {
            if (money >= ap[i].price && ad1 + ap[i].ad > max_ad_in_1room) {
                max_ad_in_1room = ad1 + ap[i].ad;
                in_1room_num = i + 1;
            }
        }
        else {
            if (money >= ap[i].price && ad2 + ap[i].ad > max_ad_in_2room) {
                max_ad_in_2room = ad2 + ap[i].ad;
                in_2room_num = i + 1;
            }
        }
    }
    sort(ap.begin(), ap.end(), cmp_ap);
    int ans_ap = -1, ans_fr = -1;
    long max_ad_tog = -1;
    int found_ap = -1;
    for (int i = 0; i < n; i++) {
        found_ap = find_good_ap(ap, money, fr[i].money);
        if (found_ap != -1) {
            long cur_ad = fr[i].ad + ap[found_ap].ad;
            if (cur_ad > max_ad_tog) {
                ans_ap = ap[found_ap].num;
                ans_fr = fr[i].num;
                max_ad_tog = cur_ad;
            }
        }
    }

    long alone = 0, whereAlone = 0;
    if (max_ad_in_1room != -1 || max_ad_in_2room != -1) {
        if (max_ad_in_1room > max_ad_in_2room) {
            alone = max_ad_in_1room;
            whereAlone = in_1room_num;
        }
        else {
            alone = max_ad_in_2room;
            whereAlone = in_2room_num;
        }
        can_buy_alone = true;
    }

    if (found_ap == -1 && !can_buy_alone) {
        cout << "Forget about apartments. Live in the dormitory." << endl;
        return 0;
    }

    if (alone > max_ad_tog)
        cout << "You should rent the apartment #" << whereAlone << " alone." << endl;
    else
        cout << "You should rent the apartment #" << ans_ap << " with the friend #"
    << ans_fr << "." << endl;

    return 0;
}
```

Задача I - Traffic Jam in Flower Town

Having returned from Sun City, Dunno told all his friends that every shorty may have a personal automobile. Immediately after that so many citizens took a fancy of becoming road-users, that Bendum and Twistum had to make a mass production of cars on soda water with syrup. Now traffic jams from several cars occasionally appear on the crossing of Bell-flower Street and Daisy Street.

Bell-flower Street goes from the South to the North and has two driving paths. It has the right driving, i. e. automobiles move from the South to the North on the Eastern path and from the North to the South on the Western path. Daisy Street is single-pathed, and it is perpendicular to Bell-flower Street. There is one-way movement on it, but its driving direction is organized in such a way that automobiles drive away from the crossroad in two opposite directions (see the picture).



Yesterday on his way home Dunno saw cars standing in a traffic jam on Bell-flower Street from different sides of the crossing with Daisy Street. Some of the drivers wanted to go forward, some wanted to turn right or left. An automobile can pass the crossing in one second, but if the driver is turning left, he first have to let pass all oncoming vehicles, going forward and to the right. How many seconds did it take all the cars to pass the crossing, providing that no other cars drove up to the crossing?

Input

The first line contains the sequence of symbols "`F`", "`L`" and "`R`", describing directions in which drivers who arrived to the crossing from the South wanted to go. "`F`" stands for those drivers who were going forward, "`L`" is for those who were turning left, and "`R`" is for those who were turning right. Automobiles are listed in the order from the closest to the crossing to the farthest one. The second line contains the description of the cars, arrived to the crossing from the North, in the same form. Both sequences have length from 1 to 1 000.

Output

Output time in seconds, which took all the cars to pass the crossing.

Examples

test	answer
`RLF` `FF`	`4`
`L` `L`	`1`

Алгоритм

???????????????????

Исходный код

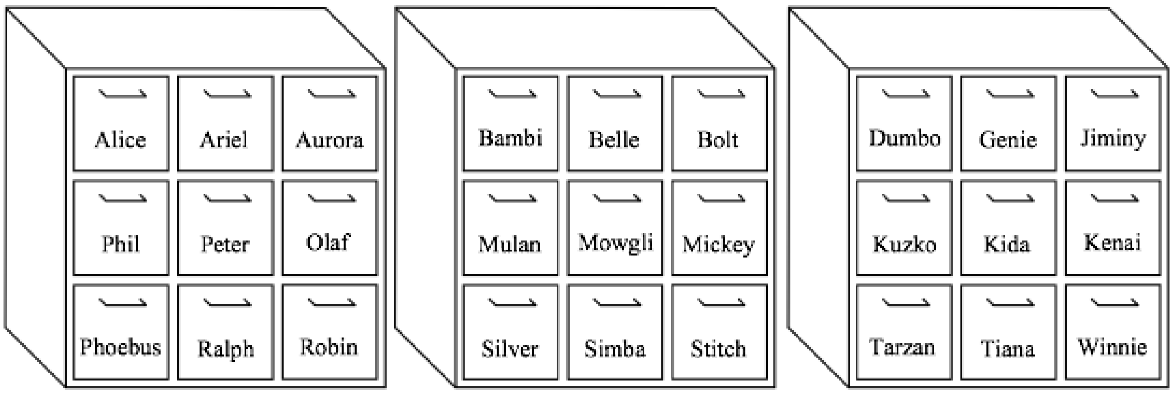
```cpp
#include <iostream>
#include <deque>

using namespace std;

int main()
{
    deque <char> south;
    deque <char> north;

    char temp;
    int time = 0;
    temp = cin.get();
    while(temp != '\n'){
        south.push_back(temp);
        temp = cin.get();
    }
    temp = cin.get();
    while(temp != '\n'){
        north.push_back(temp);
        temp = cin.get();
    }
    char s, n;
    while(south.size() > 0 && north.size() > 0){
        s = south[0];
        n = north[0];
        time++;
        if(s == 'R' && n == 'L')
            south.pop_front();
        else if(s == 'L' && n == 'R')
            north.pop_front();
         else if(s == 'L' && n == 'F')
            north.pop_front();
         else if(s == 'F' && n == 'L')
            south.pop_front();
         else {
            south.pop_front();
            north.pop_front();
         }
    }

    if(south.size() > 0)
        time += south.size();
    if(north.size() > 0)
        time += north.size();

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

Задача L - Donald is a postman

Time limit:	1 second
Memory limit:	64 megabytes

Donald Duck works as a postman for the Walt Disney Studios. He delivers children's letters from all over the world to his friends, which are cartoon characters. The Studios has three cases for the letters, with nine sections in each case. Every section has the name of the receiver on it. All cases stand in a row as it is shown at the picture below.

Donald Duck have brought n letters today. Initially, he stands near the leftmost case. He has to make one step to go to the neighboring case or to the previous one. How many steps will he make until he puts all the letters into the respective sections, if he does this in the order they are in his bag?



Input

The first line contains an integer n that is the amount of letters in Donald's bag ($1 \le n \le 1\,000$). The following n lines contain receivers of the letters in the order they are in the bag.

Output

Output the number of steps Donald should make to put all the letters into the cases.

Example

test	answer
4 Aurora Tiana Ariel Mulan	5

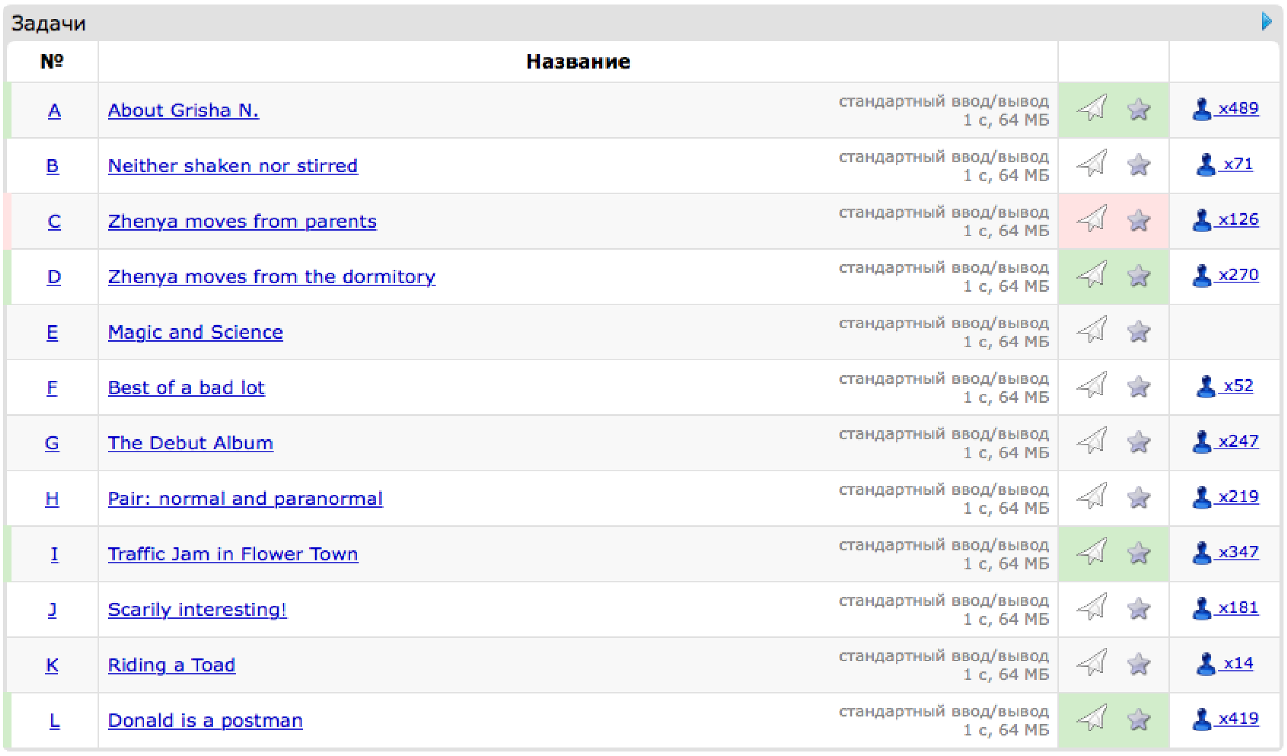
Алгоритм

???????????????????

Исходный код

```cpp
#include <iostream>

using namespace std;

int main()
{
    int n;
    cin >> n;
    string name;
    int state = 1;
    int answer = 0;
    for (int i = 0; i < n; i++) {
        cin >> name;
        char t = name[0];
        if (state == 1) {
            if (t == 'B' || t == 'M' || t == 'S') {
                answer++;
                state = 2;
            }
            else if (t == 'D' || t == 'J' || t == 'K' || t == 'T' || t == 'W' || t == 'G') {
                answer += 2;
                state = 3;
            }
        }
        else if (state == 2) {
            if (t == 'A' || t == 'P' || t == 'O' || t == 'R') {
                answer++;
                state = 1;
            }
            else if (t == 'D' || t == 'J' || t == 'K' || t == 'T' || t == 'W' || t == 'G') {
                answer++;
                state = 3;
            }

        }
        else {
            if (t == 'A' || t == 'P' || t == 'O' || t == 'R') {
                answer += 2;
                state = 1;
            }
            else if (t == 'B' || t == 'M' || t == 'S') {
                answer++;
                state = 2;
            }
        }
    }

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

Результаты



Задачи

№	Название			
A	About Grisha N.	стандартный ввод/вывод 1 с, 64 МБ		x489
B	Neither shaken nor stirred	стандартный ввод/вывод 1 с, 64 МБ		x71
C	Zhenya moves from parents	стандартный ввод/вывод 1 с, 64 МБ		x126
D	Zhenya moves from the dormitory	стандартный ввод/вывод 1 с, 64 МБ		x270
E	Magic and Science	стандартный ввод/вывод 1 с, 64 МБ		
F	Best of a bad lot	стандартный ввод/вывод 1 с, 64 МБ		x52
G	The Debut Album	стандартный ввод/вывод 1 с, 64 МБ		x247
H	Pair: normal and paranormal	стандартный ввод/вывод 1 с, 64 МБ		x219
I	Traffic Jam in Flower Town	стандартный ввод/вывод 1 с, 64 МБ		x347
J	Scarily interesting!	стандартный ввод/вывод 1 с, 64 МБ		x181
K	Riding a Toad	стандартный ввод/вывод 1 с, 64 МБ		x14
L	Donald is a postman	стандартный ввод/вывод 1 с, 64 МБ		x419

2.10 Codeforces ACM-ICPC Южный четвертьфинал

Задача D - Data Center

The startup "Booble" has shown explosive growth and now it needs a new data center with the capacity of m petabytes. Booble can buy servers, there are n servers available for purchase: they have equal price but different capacities. The i-th server can store a_i petabytes of data. Also they have different energy consumption — some servers are *low voltage* and other servers are not.

Booble wants to buy the minimum number of servers with the total capacity of at least m petabytes. If there are many ways to do it Booble wants to choose a way to maximize the number of *low voltage* servers. Booble doesn't care about exact total capacity, the only requirement is to make it at least m petabytes.

Input

The first line contains two integer numbers n and m $(1 \le n \le 2 \cdot 10^5, 1 \le m \le 2 \cdot 10^{15})$ — the number of servers and the required total capacity.

The following n lines describe the servers, one server per line. The i-th line contains two integers a_i, l_i $(1 \le a_i \le 10^{10}, 0 \le l_i \le 1)$, where a_i is the capacity, $l_i = 1$ if server is *low voltage* and $l_i = 0$ in the opposite case.

It is guaranteed that the sum of all a_i is at least m.

Output

Print two integers r and w on the first line — the minimum number of servers needed to satisfy the capacity requirement and maximum number of *low voltage* servers that can be bought in an optimal r servers set.

Print on the second line r distinct integers between 1 and n — the indices of servers to buy. You may print the indices in any order. If there are many solutions, print any of them.

Examples

standard input	standard output

4 10
3 1
7 0
5 1
4 1
	2 1
	4 2

3 13
6 1
6 1
6 1
	3 3
	1 2 3

Алгоритм

???????????????????

Исходный код

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

using namespace std;

typedef struct {
    LL num;
    ULL cap;
    short low;
} server;

bool cmp_low(const server &a, const server &b) {
    return a.low > b.low;
}

bool cmp_cap(const server &a, const server &b) {
    if (a.cap == b.cap) return a.low > b.low;
    return a.cap > b.cap;
}

int main() {
    LL n;
    ULL m;
    cin >> n >> m;
    vector<server> s(n);
    for (LL i = 0; i < n; i++) {
        s[i].num = i + 1;
        cin >> s[i].cap;
        cin >> s[i].low;
    }

    sort(s.begin(), s.end(), cmp_cap);
    LL ind = 0;
    ULL curr_cap = 0;
    LL count = 0;

    while (curr_cap < m) {
        curr_cap += s[ind].cap;
        if (s[ind].low) count++;
        ind++;
    }

    ULL rem = curr_cap - m;

    if (rem == 0) {
        cout << ind << " ";
        cout << count << endl;
        for (LL i = 0; i < ind; i++) {
            cout << s[i].num << " ";
        }
        cout << endl;
        return 0;
    }

    for (LL i = ind - 1; i >= 0; i--) {
        if (!s[i].low && rem > 0) {
            bool azaza = false;
            for (LL j = ind; j < n; j++) {
                if (s[j].low && (s[j].cap + rem >= s[i].cap)) {
                    rem -= s[i].cap - s[j].cap;
                    swap(s[i], s[j]);
                    count++;
                    azaza = true;
                    break;
                }
            }
            if (!azaza) break;
        }
        else if (rem == 0) break;
    }

    cout << ind << " " << count << endl;
    for (LL i = 0; i < ind; i++) {
        cout << s[i].num << " ";
    }
    cout << endl;;
  return 0;
}
```

Задача I - Sales in GameStore

A well-known Berland online games store has announced a great sale! Buy any game today, and you can download more games for free! The only constraint is that the total price of the games downloaded for free can't exceed the price of the bought game.

When Polycarp found out about the sale, he remembered that his friends promised him to cover any single purchase in GameStore. They presented their promise as a gift for Polycarp's birthday.

There are n games in GameStore, the price of the i-th game is p_i. What is the maximum number of games Polycarp can get today, if his friends agree to cover the expenses for any single purchase in GameStore?

Input

The first line of the input contains a single integer number n $(1 \le n \le 2000)$ — the number of games in GameStore. The second line contains n integer numbers $p_1, p_2, \ldots, p_n$ $(1 \le p_i \le 10^5)$, where p_i is the price of the i-th game.

Output

Print the maximum number of games Polycarp can get today.

Examples

standard input	standard output
5 5 3 1 5 6	3
2 7 7	2

Алгоритм

????????????????????

Исходный код

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
bool compare(const int &a, const int &b)
{
    return a<b;
}
int main(int argc, const char * argv[]) {
    vector<int> p(2001);
    int n;
    cin >> n;
    for(int i=0; i<n; ++i)
        cin >> p[i];
    sort(p.begin(), p.begin()+n);
    int sum = 0;
    int i=0;
    while(i<n-1 && sum+p[i]<=p[n-1])
        sum += p[i++];
    cout << i+1;
    return 0;
}
```

Задача M - Variable Shadowing

In computer programming, *variable shadowing* occurs when a variable declared within a certain scope has the same name as a variable declared in an outer scope. The outer variable is said to be shadowed by the inner variable, and this can lead to a confusion. If multiple outer scopes contain variables with the same name, the variable in the nearest scope will be shadowed.

Formally, a declared variable shadows another declared variable if the following conditions are met simultaneously:

- the other variable is declared in outer scope and before (in terms of position in program source code) the declaration of the first variable,
- the other variable is nearest among all variables satisfying the condition above.

Here is an example containing exactly one variable shadowing:

```
/* Prints a+max(b,c) */
int main() {
    int a, b, c;
    cin >> a >> b >> c;
    if (b > c) {
        int a = b; // <-- variable 'a' shadows outer 'a'
        int x = c;
        b = x;
        c = a;
    }
    int x = a + c; // <-- no shadowing here
    cout << x << endl;
}
```

Variable shadowing is permitted in many modern programming languages including C++, but compilers can warn a programmer about variable shadowing to avoid possible mistakes in a code.

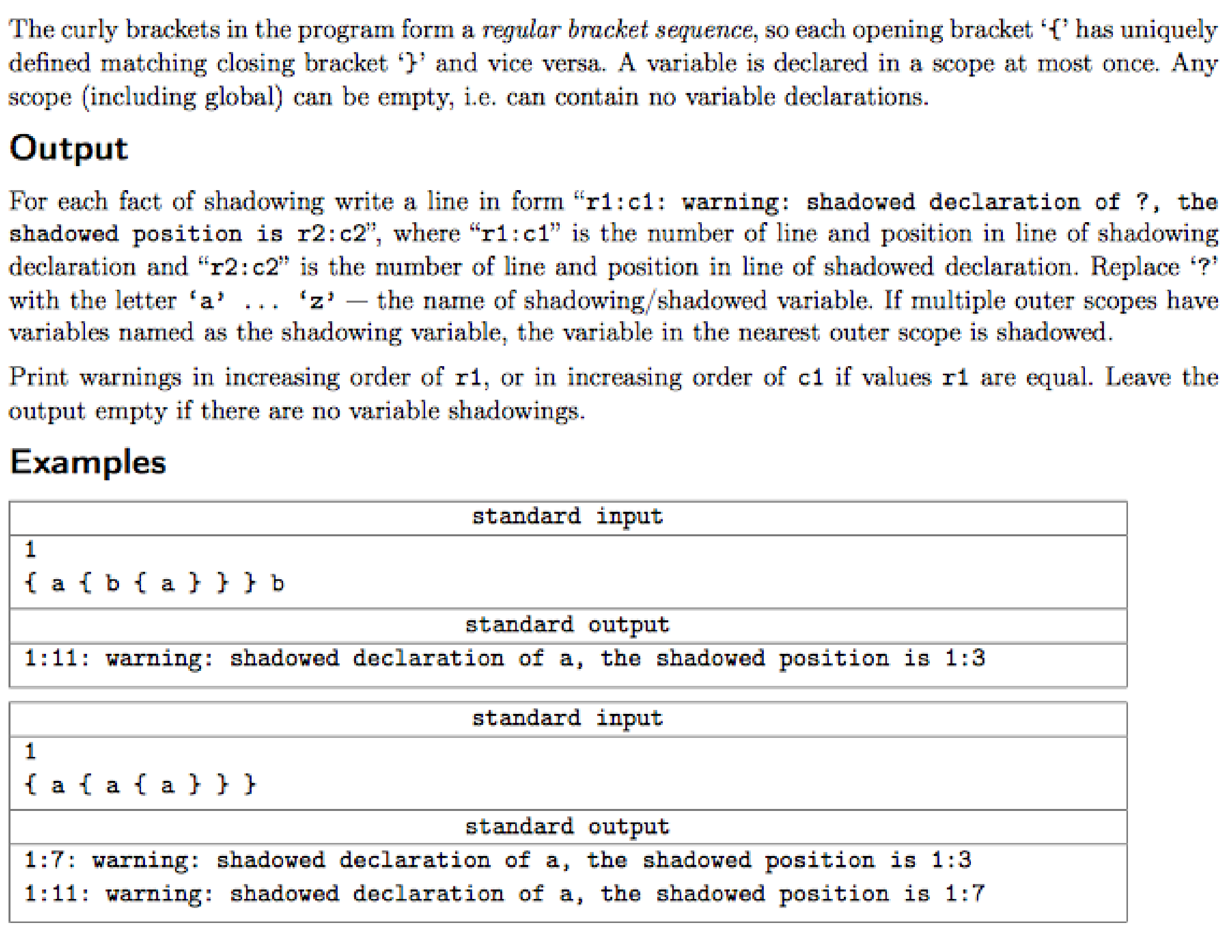
Consider a trivial programming language that consists only of scopes and variable declarations. The program consists of lines, each line contains only characters '{', '}', 'a' ...'z' separated by one or more spaces.

- *Scopes.* A scope (excluding global) is bounded with a pair of matching curly brackets '{' and '}'. A scope is an inner scope relative to another scope if brackets of the first scope are enclosed by brackets of the second scope.
- *Variables.* A variable declaration in this language is written just as a name of the variable. In addition all variables are lowercase Latin letters from 'a' to 'z' inclusive (so there are at most 26 variable names). A variable is declared in each scope at most once.

Given a syntactically correct program (i.e. curly brackets form a *regular bracket sequence*), write an analyzer to warn about each fact of variable shadowing. Warnings should include exact positions of shadowing and shadowed variables. Your output should follow the format shown in the examples below.

Input

The first line contains integer n ($1 \leq n \leq 50$) — the number of lines in the program. The following n lines contain the program. Each program line consists of tokens '{', '}', 'a' ...'z' separated by one or more spaces. The length of each line is between 1 and 50 characters. Each program line contains at least one non-space character.



The curly brackets in the program form a *regular bracket sequence*, so each opening bracket '{' has uniquely defined matching closing bracket '}' and vice versa. A variable is declared in a scope at most once. Any scope (including global) can be empty, i.e. can contain no variable declarations.

Output

For each fact of shadowing write a line in form "`r1:c1: warning: shadowed declaration of ?, the shadowed position is r2:c2`", where "`r1:c1`" is the number of line and position in line of shadowing declaration and "`r2:c2`" is the number of line and position in line of shadowed declaration. Replace '?' with the letter `'a' ... 'z'` — the name of shadowing/shadowed variable. If multiple outer scopes have variables named as the shadowing variable, the variable in the nearest outer scope is shadowed.

Print warnings in increasing order of `r1`, or in increasing order of `c1` if values `r1` are equal. Leave the output empty if there are no variable shadowings.

Examples

standard input

```
1
{ a { b { a } } } b
```

standard output

```
1:11: warning: shadowed declaration of a, the shadowed position is 1:3
```

standard input

```
1
{ a { a { a } } }
```

standard output

```
1:7: warning: shadowed declaration of a, the shadowed position is 1:3
1:11: warning: shadowed declaration of a, the shadowed position is 1:7
```

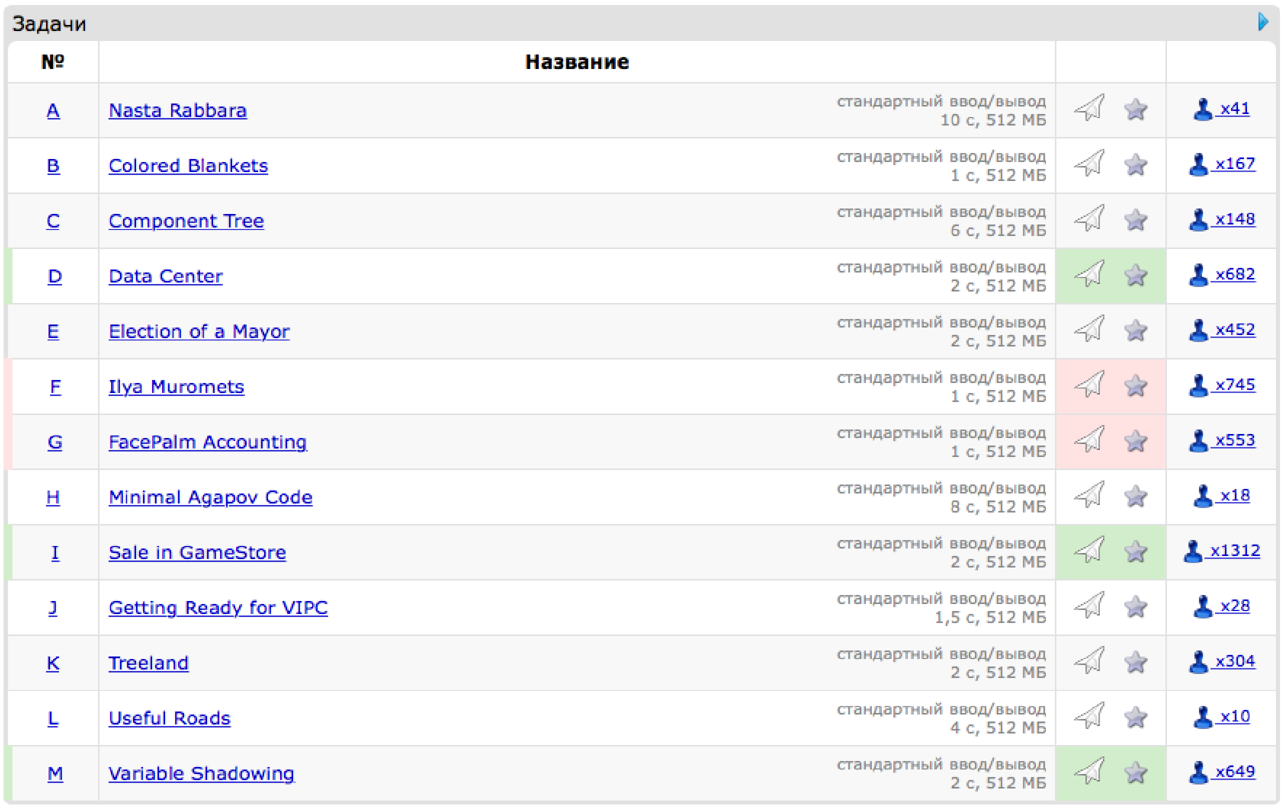
Алгоритм

???????????????????

Исходный код

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
typedef struct {
    char ch;
    int line;
    int sym;
} var;
int main() {
    int f,n;
    cin >> n;
    vector <stack <var> > all_alph(26);
    stack <var> curr;
    char temp;
    int symbol = 0;
    var to_put;
    temp = cin.get();
    for (int i = 1; i <= n; i++){
        temp = cin.get();
        symbol = 1;
        while(temp != '\n') {
            if(temp == '}') {
                to_put = curr.top();
                curr.pop();
                while (to_put.ch != '{') {
                    all_alph[to_put.ch - 97].pop();
                    to_put = curr.top();
                    curr.pop();
                }
            }
            else if(temp == '{') {
                to_put.ch = temp;
                to_put.sym = symbol;
                to_put.line = i;
                curr.push(to_put);
            }
            else if (temp != ' '){
                to_put.sym = symbol;
                to_put.line = i;
                to_put.ch = temp;
                curr.push(to_put);
                if(all_alph[temp - 97].size() != 0)
                    cout << to_put.line << ":" << to_put.sym
                    << ": warning: shadowed declaration of "<< to_put.ch
                    << ", the shadowed position is " << all_alph[temp - 97].top().
    line
                    << ":" << all_alph[temp - 97].top().sym << endl;
                all_alph[temp - 97].push(to_put);
            }

            temp = cin.get();
            symbol++;
        }
    }
    return 0;
}
```

Результаты



Задачи

№	Название			
A	Nasta Rabbara	стандартный ввод/вывод 10 с, 512 МБ		x41
B	Colored Blankets	стандартный ввод/вывод 1 с, 512 МБ		x167
C	Component Tree	стандартный ввод/вывод 6 с, 512 МБ		x148
D	Data Center	стандартный ввод/вывод 2 с, 512 МБ	🟩	x682
E	Election of a Mayor	стандартный ввод/вывод 2 с, 512 МБ		x452
F	Ilya Muromets	стандартный ввод/вывод 1 с, 512 МБ	🟥	x745
G	FacePalm Accounting	стандартный ввод/вывод 1 с, 512 МБ	🟥	x553
H	Minimal Agapov Code	стандартный ввод/вывод 8 с, 512 МБ		x18
I	Sale in GameStore	стандартный ввод/вывод 2 с, 512 МБ	🟩	x1312
J	Getting Ready for VIPC	стандартный ввод/вывод 1,5 с, 512 МБ		x28
K	Treeland	стандартный ввод/вывод 2 с, 512 МБ		x304
L	Useful Roads	стандартный ввод/вывод 4 с, 512 МБ		x10
M	Variable Shadowing	стандартный ввод/вывод 2 с, 512 МБ	🟩	x649

2.11 ACM-ICPC Московский четвертьфинал

Так как соревнование проводилось в МГУ, то турнирная таблица с результатами и исходные коды программ не доступны.

2.12 Codeforces Training S02E07

Задача C - Will It Stop?

Byteasar was wandering around the library of the University of Warsaw and at one of its facades he noticed a piece of a program with an inscription "Will it stop?". The question seemed interesting, so Byteasar tried to tackle it after returning home. Unfortunately, when he was writing down the piece of code he made a mistake and noted:

```
while $n > 1$ do
  if $n$ mod $2 = 0$ then
    $n := n/2$
  else
    $n := 3 \cdot n + 3$
```

Byteasar is now trying to figure out, for which initial values of the variable n the program he wrote down stops. We assume that the variable n has an unbounded size, i.e., it may attain arbitrarily large values.

Input

The first and only line of input contains one integer n ($2 \leqslant n \leqslant 10^{14}$).

Output

In the first and only line of output you should write a single word `TAK` (i.e., *yes* in Polish), if the program stops for the given value of n, or `NIE` (*no* in Polish) otherwise.

Example

For the input data:

```
4
```

the correct result is:

```
TAK
```

Алгоритм

???????????????????

Исходный код

```
#include <iostream>
using namespace std;
int main()
{
    unsigned long long a;
    cin >> a;
    while(!(a%2))a/=2;
    if(a==1)
        cout << "TAK";
    else
        cout << "NIE";
    return 0;
}
```

Задача H - Afternoon Tea

During his visit at Bytic Islands Byteasar really enjoyed the national beverage of Byteans, that is, tea with milk. This drink is always prepared in a strictly determined manner, which is as follows. Firstly the teacup is filled with tea mixed half and half with milk. Then, an n-letter *ceremonial word* consisting of letters H and M is chosen. Now, for $i = 1, 2, \ldots, n$, the following action is performed: if the i-th letter of the ceremonial word is H, one should drink half of the teacup, add tea until the teacup is full and stir. On the other hand, if the i-th letter of the word is M, one should perform a similar action, however milk should be added instead of tea. After such action is performed for each letter of the ceremonial word, the remaining liquid is disposed of.

Each time Byteasar performs the ceremony, he wonders which of the ingredients he has drunk more: tea or milk. Help Byteasar answer this question.

Input

The first line of input holds an integer n ($1 \leqslant n \leqslant 100\,000$). The second line contains an n-letter word consisting of letters H and M; this is the ceremonial word used by Byteasar.

Output

Your program should output a single letter H if Byteasar has drunk more tea than milk; a single letter M if he has drunk more milk than tea; or the word HM if he has drunk equal amounts of tea and milk.

Example

For the input data:

```
5
HMHHM
```

the correct result is:

```
H
```

Explanation of the example: Byteasar has drunk $1\frac{37}{64}$ teacups of tea and $\frac{59}{64}$ teacups of milk in total.

Алгоритм

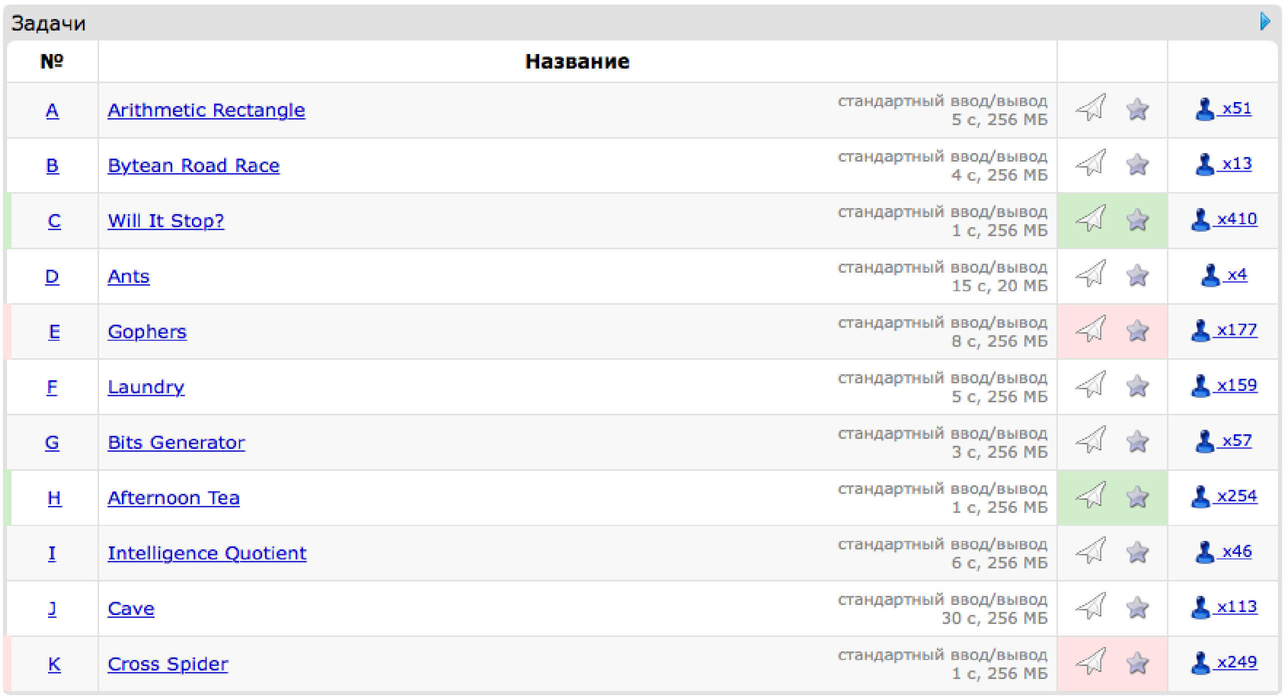
??????????????????

Исходный код

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
    int n;
    cin >> n;
    if(n==1)
    {
        cout << "HM";
        return 0;
    }
    cin.get();
    char c;
    long double hDrunked = 0, mDrunked = 0;
    hDrunked = mDrunked = (1-pow(0.5, n))*0.5;
    for(int i=0; i<n-1; ++i)
    {
        c=cin.get();
        if(c=='H')
            hDrunked += (1-pow(0.5, n-i-1))*0.5;
        else
```

```
                mDrunked += (1-pow(0.5, n-i-1))*0.5;
        }
        if(hDrunked>mDrunked)
            cout << "H";
        else if(hDrunked<mDrunked)
            cout << "M";
        return 0;
}
```

Результаты

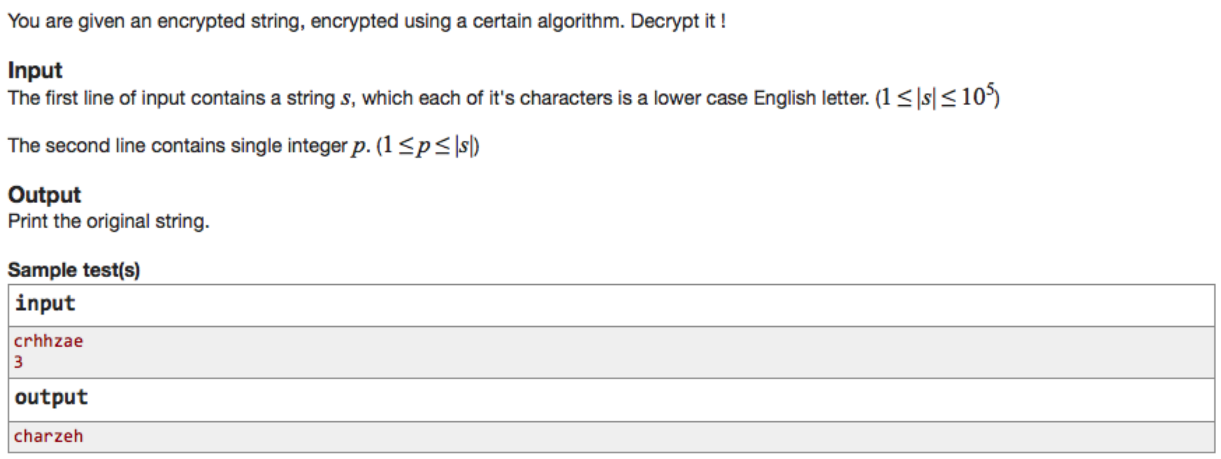


Задачи

№	Название			
A	Arithmetic Rectangle	стандартный ввод/вывод 5 с, 256 МБ		x51
B	Bytean Road Race	стандартный ввод/вывод 4 с, 256 МБ		x13
C	Will It Stop?	стандартный ввод/вывод 1 с, 256 МБ	🟩	x410
D	Ants	стандартный ввод/вывод 15 с, 20 МБ		x4
E	Gophers	стандартный ввод/вывод 8 с, 256 МБ	🟥	x177
F	Laundry	стандартный ввод/вывод 5 с, 256 МБ		x159
G	Bits Generator	стандартный ввод/вывод 3 с, 256 МБ		x57
H	Afternoon Tea	стандартный ввод/вывод 1 с, 256 МБ	🟩	x254
I	Intelligence Quotient	стандартный ввод/вывод 6 с, 256 МБ		x46
J	Cave	стандартный ввод/вывод 30 с, 256 МБ		x113
K	Cross Spider	стандартный ввод/вывод 1 с, 256 МБ	🟥	x249

2.13 Codeforces Crypto Cup 1.0

Задача B - :-P



You are given an encrypted string, encrypted using a certain algorithm. Decrypt it !

Input

The first line of input contains a string s, which each of it's characters is a lower case English letter. ($1 \leq |s| \leq 10^5$)

The second line contains single integer p. ($1 \leq p \leq |s|$)

Output

Print the original string.

Sample test(s)

input

```
crhhzae
3
```

output

```
charzeh
```

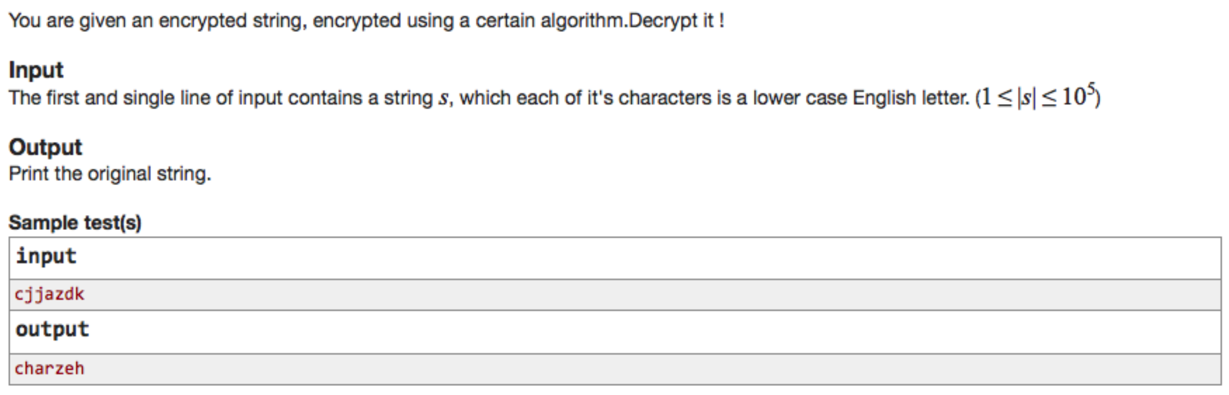
Алгоритм

??????????????????

Исходный код

```
#include <iostream>
#include <cmath>
#include <vector>
using namespace std;
int main(int argc, const char * argv[]) {
    char str[100001];
    long p, len;
    while((str[len]=cin.get())!='\n') ++len;
    str[len] = '\0';
    cin >> p;
    vector< vector <char> > v(p);
    long vSize = len/p, r = len%p;
    for(long i=r; i<p; ++i) {
        v[i].resize(vSize);
    }
    for(long i=0; i<r; ++i) {
        v[i].resize(vSize+1);
    }
    for(long i=0, k=0; i<p; ++i) {
        for(long j=0; j<v[i].size(); ++j, ++k) {
            v[i][j] = str[k];
        }
    }
    for(long i=0; i<len; ++i) {
        cout.put(v[i%p][i/p]);
    }
    return 0;
}
```

Задача C - Pgkpxumgs



You are given an encrypted string, encrypted using a certain algorithm.Decrypt it !

Input
The first and single line of input contains a string s, which each of it's characters is a lower case English letter. $(1 \leq |s| \leq 10^5)$

Output
Print the original string.

Sample test(s)

input
cjjazdk
output
charzeh

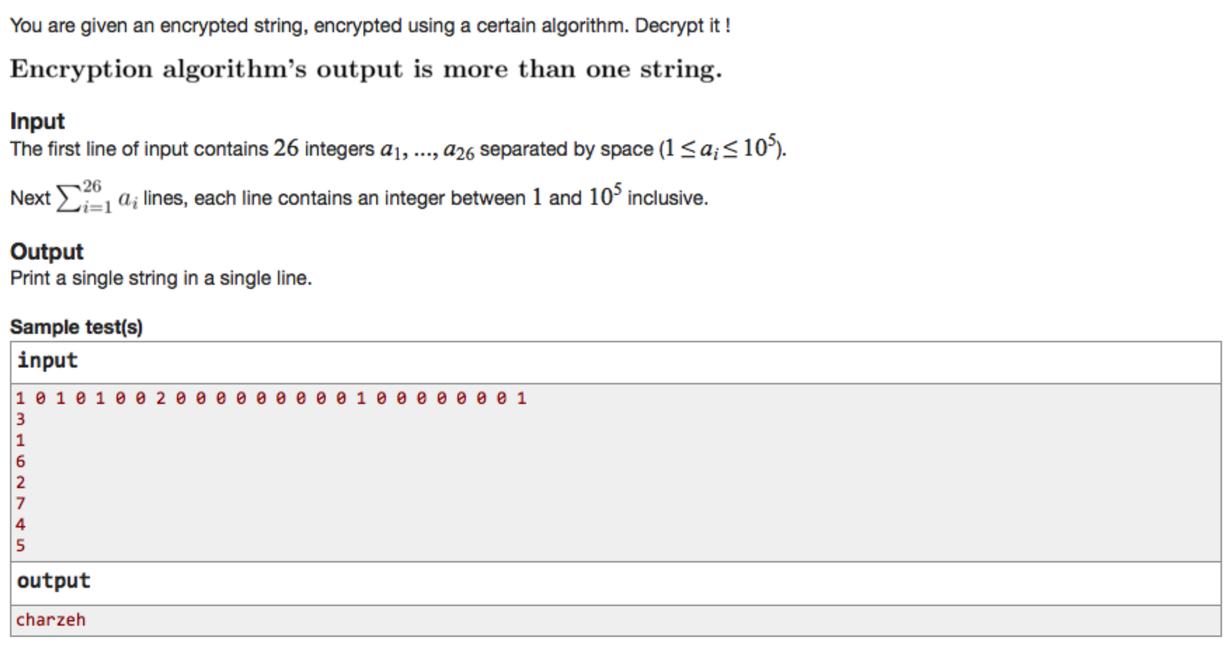
Алгоритм

??????????????????

Исходный код

```
#include <iostream>
#include <cstdio>
using namespace std;
int main() {
  char cur, prev;
  cout.put(prev = cin.get());
  while ((cur = cin.get()) != '\n') {
    if ((int)(cur - prev) < 0) cout << (char)(cur - prev + '{');
    else cout << (char)(cur - prev + 'a');
    prev = cur;
  }
  return 0;
}
```

Задача H - Peace of AmericaReunion



You are given an encrypted string, encrypted using a certain algorithm. Decrypt it !

Encryption algorithm's output is more than one string.

Input

The first line of input contains 26 integers $a_1, \ldots, a_{26}$ separated by space ($1 \leq a_i \leq 10^5$).

Next $\sum_{i=1}^{26} a_i$ lines, each line contains an integer between 1 and 10^5 inclusive.

Output

Print a single string in a single line.

Sample test(s)

input

```
1 0 1 0 1 0 0 2 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1
3
1
6
2
7
4
5
```

output

```
charzeh
```

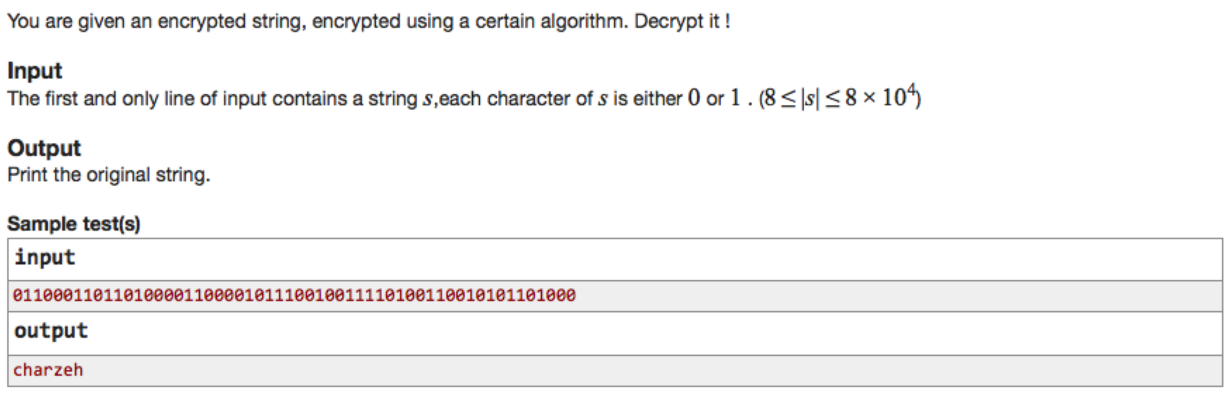
Алгоритм

???????????????????

Исходный код

```cpp
#include <iostream>
using namespace std;
int main() {
  vector<long> v(26);
  long length = 0;
  for (int i = 0; i < 26; i++) {
    cin >> v[i];
    length += v[i];
  }
  vector<char> answer(length);
  long pos;
  int nextSymb = -1;
  for (int i = 0; i < 26; i++) {
    if (v[i]) {
      nextSymb = i;
      break;
    }
  }
  for (long i = 0; i < length; i++) {
    cin >> pos;
    if (!v[nextSymb]) {
      for (int i = nextSymb; i < 26; i++) {
        if (v[i]) {
          nextSymb = i;
          break;
        }
      }
    }
    answer[--pos] = (char)(nextSymb + 'a');
    v[nextSymb]--;
  }
  for (auto n : answer) cout << n;

  return 0;
}
```

Задача I - Peace of AmericanPie



You are given an encrypted string, encrypted using a certain algorithm. Decrypt it !

Input
The first and only line of input contains a string s,each character of s is either 0 or 1 . ($8 \leq |s| \leq 8 \times 10^4$)

Output
Print the original string.

Sample test(s)

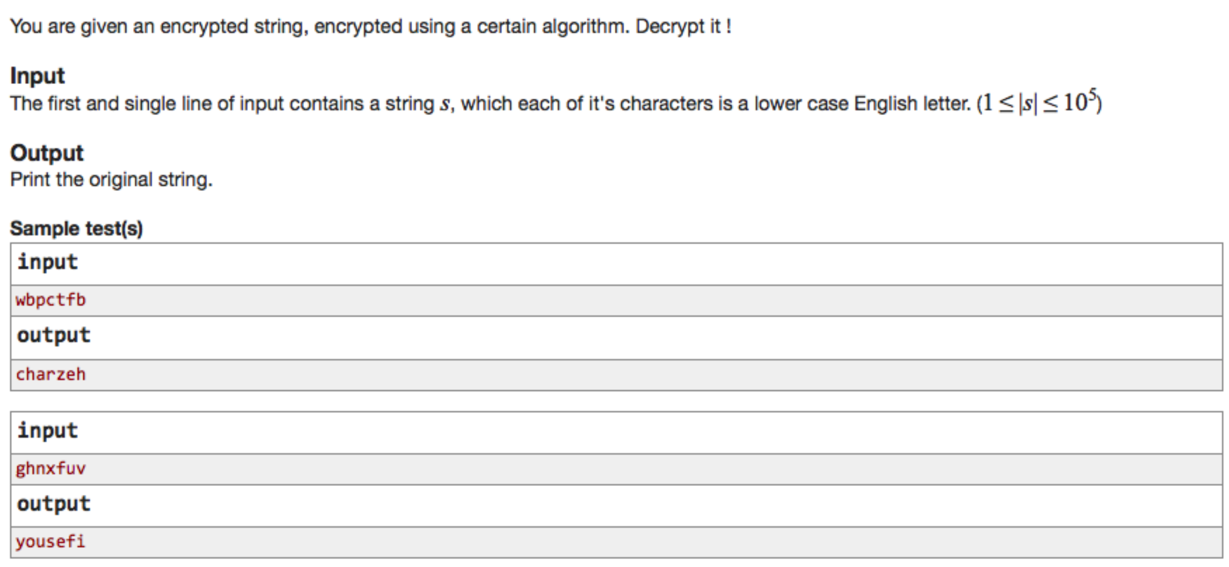
input
```
01100011011010000110000101110010011110100110010101101000
```
output
```
charzeh
```

Алгоритм
???????????????????

Исходный код

```cpp
#include <iostream>
using namespace std;
int main() {
  int byte = 0;
  int spow = 256;
  int collected = 0;
  char currentBit;
  while ((currentBit = cin.get()) != '\n') {
    cin.unget();
    while (collected < 8) {
      currentBit = cin.get();
      byte += (currentBit - '0') * spow;
      spow /= 2;
      collected++;
    }
    cout << (char)(byte / 2);
    byte = 0;
    spow = 256;
    collected = 0;
  }
  return 0;
}
```

Задача J - Common



You are given an encrypted string, encrypted using a certain algorithm. Decrypt it !

Input
The first and single line of input contains a string s, which each of it's characters is a lower case English letter. ($1 \leq |s| \leq 10^5$)

Output
Print the original string.

Sample test(s)

```
input
wbpctfb
output
charzeh
```

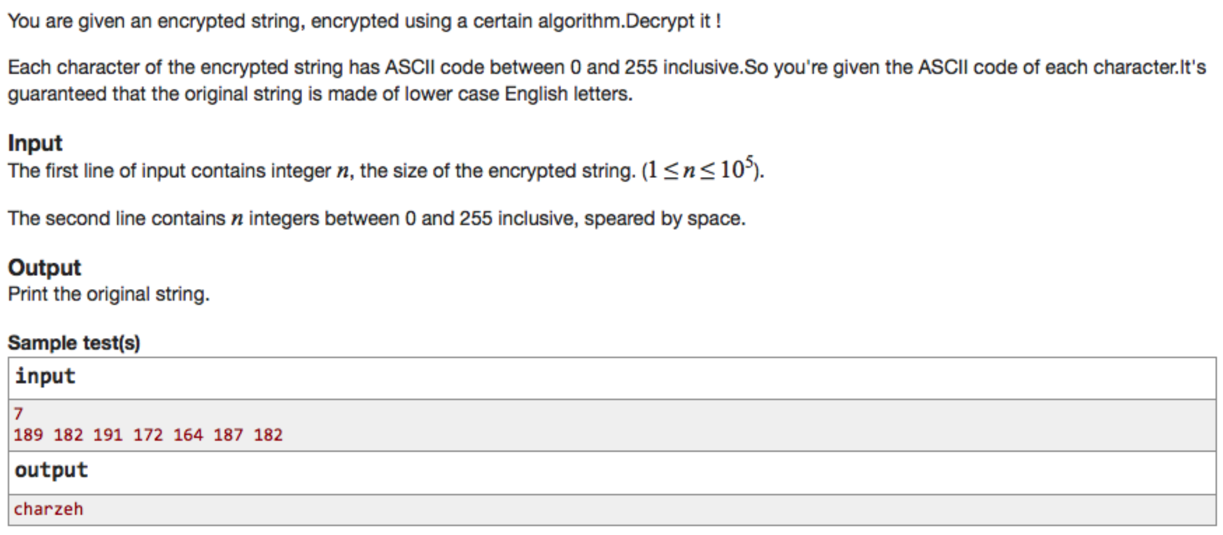
```
input
ghnxfuv
output
yousefi
```

Алгоритм ???????????????????

Исходный код

```
#include <iostream>
using namespace std;
int main() {
  char t;
  while ((t = cin.get()) != '\n') {
    switch (t) {
      case 'a':
        cout << "n";
        break;
      case 'b':
        cout << "h";
        break;
      case 'c':
        cout << "r";
        break;
      case 'd':
        cout << "x";
        break;
      case 'e':
        cout << "k";
        break;
      case 'f':
        cout << "e";
        break;
      case 'g':
        cout << "y";
        break;
      case 'h':
        cout << "o";
        break;
      case 'i':
        cout << "q";
        break;
      case 'j':
        cout << "m";
        break;
      case 'k':
        cout << "j";
        break;
      case 'l':
        cout << "b";
        break;
      case 'm':
        cout << "d";
        break;
      case 'n':
        cout << "u";
        break;
      case 'o':
        cout << "v";
        break;
      case 'p':
        cout << "a";
        break;
      case 'q':
        cout << "p";
        break;
      case 'r':
        cout << "w";
        break;
      case 's':
        cout << "g";
        break;
      case 't':
        cout << "z";
        break;
      case 'u':
        cout << "f";
        break;
      case 'v':
        cout << "i";
        break;
      case 'w':
        cout << "c";
        break;
      case 'x':
        cout << "s";
        break;
      case 'y':
        cout << "t";
        break;
      case 'z':
        cout << "l";
        break;
    }
  }
  return 0;
}
```

Задача M - oPlus



You are given an encrypted string, encrypted using a certain algorithm.Decrypt it !

Each character of the encrypted string has ASCII code between 0 and 255 inclusive.So you're given the ASCII code of each character.It's guaranteed that the original string is made of lower case English letters.

Input

The first line of input contains integer n, the size of the encrypted string. ($1 \le n \le 10^5$).

The second line contains n integers between 0 and 255 inclusive, speared by space.

Output

Print the original string.

Sample test(s)

input

```
7
189 182 191 172 164 187 182
```

output

```
charzeh
```

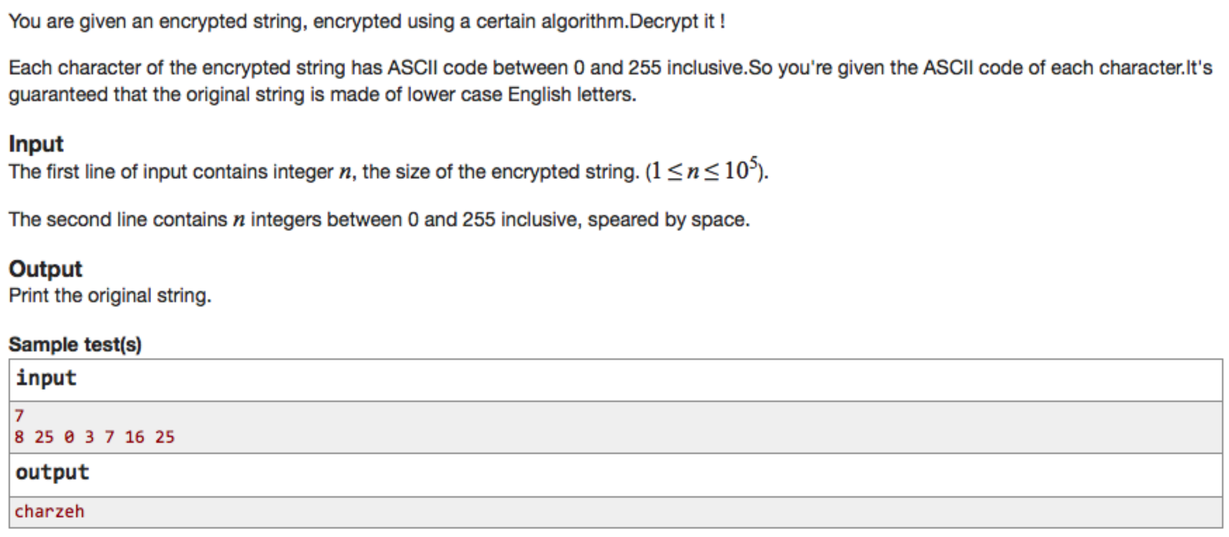
Алгоритм

???????????????????

Исходный код

```
#include <iostream>
using namespace std;
int main() {
  long n;
  int curr, sum;
  cin >> n;
  for (long i = 0; i < n; i++) {
    cin >> curr;
    if (curr % 2) sum = 400;
    else sum = 398;
    cout << (char)(sum - curr - 112);
  }
  return 0;
}
```

Задача N - tirnaoeumPt



You are given an encrypted string, encrypted using a certain algorithm.Decrypt it !

Each character of the encrypted string has ASCII code between 0 and 255 inclusive.So you're given the ASCII code of each character.It's guaranteed that the original string is made of lower case English letters.

Input
The first line of input contains integer n, the size of the encrypted string. ($1 \leq n \leq 10^5$).

The second line contains n integers between 0 and 255 inclusive, speared by space.

Output
Print the original string.

Sample test(s)

input

```
7
8 25 0 3 7 16 25
```

output

```
charzeh
```

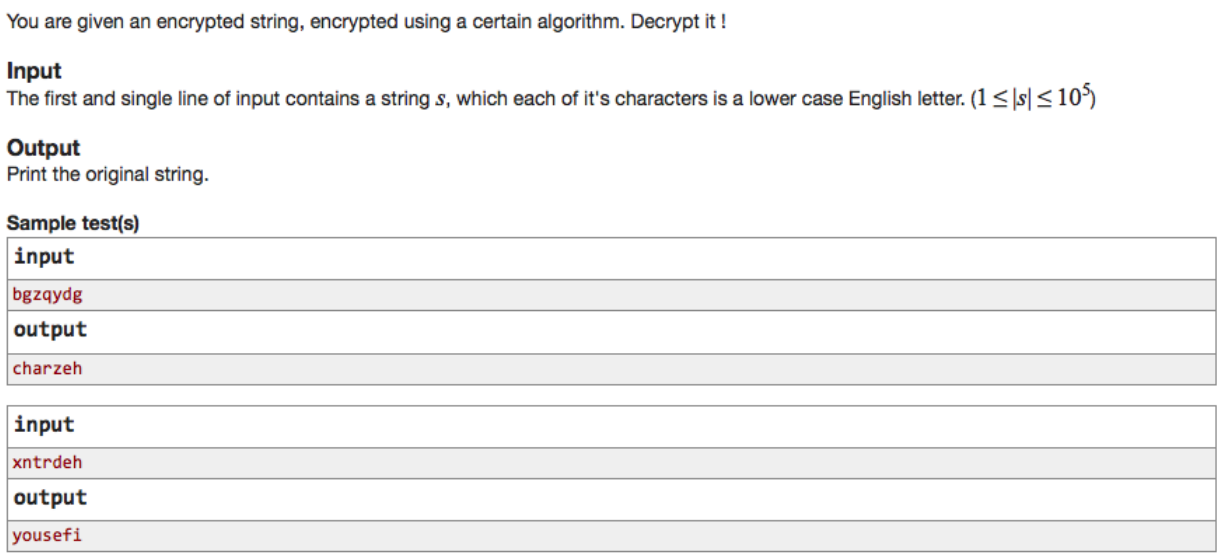
Алгоритм

??????????????????

Исходный код

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

int main()
{
    int m[] = {0, 1, 16, 17, 8, 9, 24, 25, 2, 3, 18, 19, 10, 11, 22, 23, 4,  5,  20,
  21, 12, 13, 22, 23, 6,  7,  22, 23, 14, 15};
    int n, d;
    cin >> n;
    for(int i=0; i<n; ++i)
    {
        cin >> d;
        cout.put(m[d]+'a');
    }
    return 0;
}
```

Задача Q - Peace of bzjd



You are given an encrypted string, encrypted using a certain algorithm. Decrypt it !

Input

The first and single line of input contains a string s, which each of it's characters is a lower case English letter. $(1 \leq |s| \leq 10^5)$

Output

Print the original string.

Sample test(s)

```
input
bgzqydg
output
charzeh
```

```
input
xntrdeh
output
yousefi
```

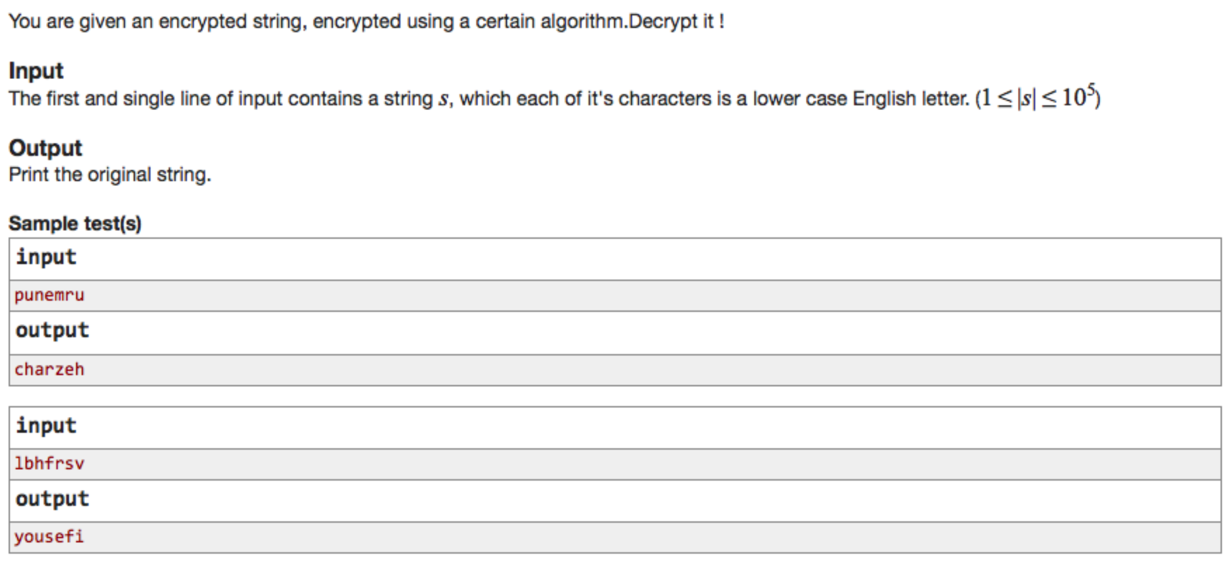
Алгоритм

??????????????????

Исходный код

```cpp
#include <iostream>

using namespace std;

int main()
{
    char temp;
    temp = cin.get();
    while (temp != '\n' && temp != EOF) {
        if (temp == 'z')
            temp = 'a';
        else temp++;
        cout << temp;
        temp = cin.get();
    }
    return 0;
}
```

Задача R - 6227020800



You are given an encrypted string, encrypted using a certain algorithm.Decrypt it !

Input
The first and single line of input contains a string s, which each of it's characters is a lower case English letter. ($1 \le |s| \le 10^5$)

Output
Print the original string.

Sample test(s)

```
input
punemru
output
charzeh
```

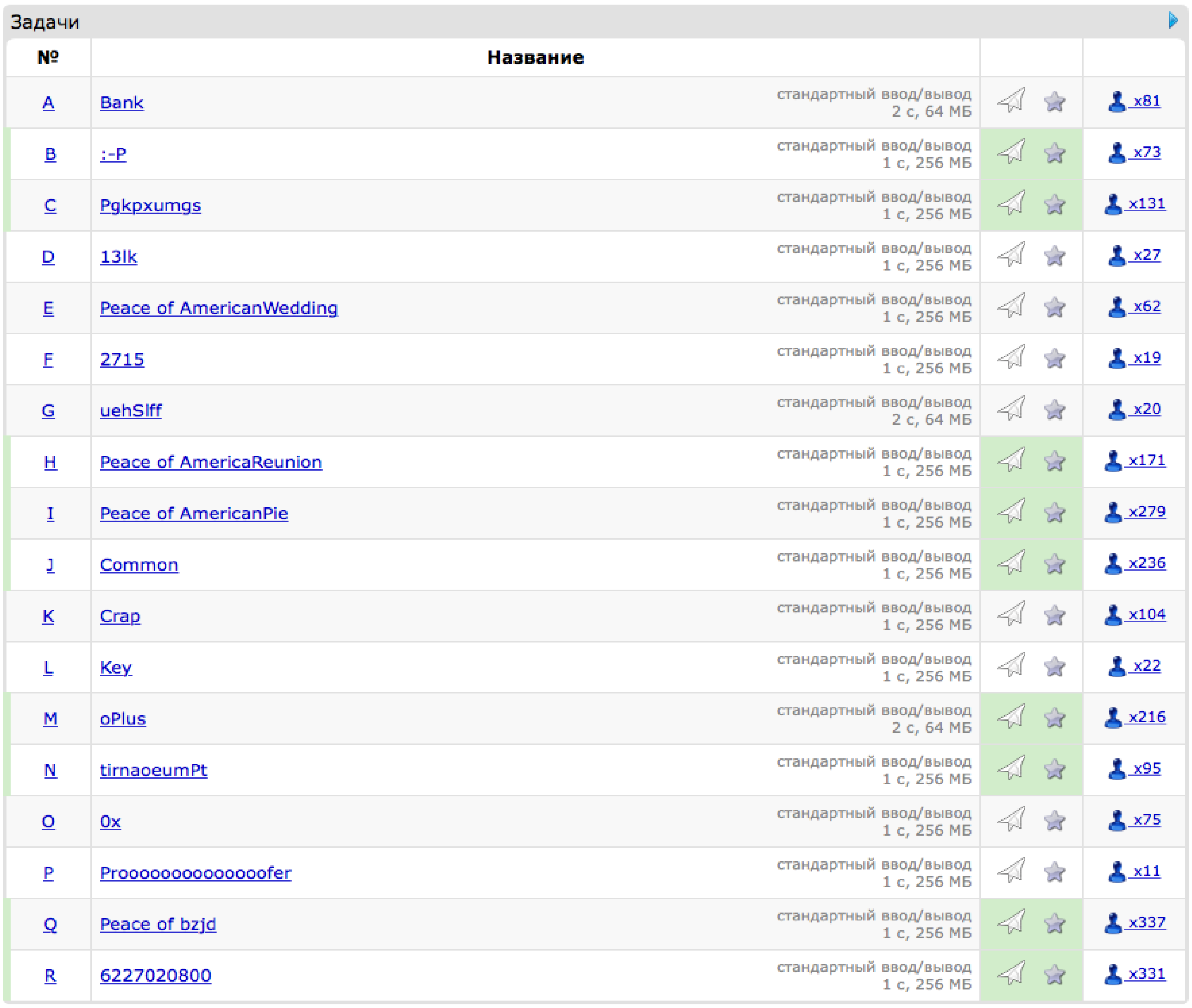
```
input
lbhfrsv
output
yousefi
```

Алгоритм
???????????????????

Исходный код

```cpp
#include <iostream>
#include <algorithm>
#include <deque>
#include <cstdio>
using namespace std;
void p(string s) {
  cout << s << endl;
}
int gcd(int a, int b){
    if (b == 0)
        return a;
    return gcd(b, a%b);
}
int main() {
  char t;
  while ((t = cin.get()) != '\n') {
    t -= 13;
    if (t < 'a') {
      cout << (char)('z' - ('a' - t) + 1);
    }
    else {
      cout << t;
    }
  }
  return 0;
}
```

Результаты



Задачи

№	Название		
A	Bank	стандартный ввод/вывод 2 с, 64 МБ	x81
B	:-P	стандартный ввод/вывод 1 с, 256 МБ	x73
C	Pgkpxumgs	стандартный ввод/вывод 1 с, 256 МБ	x131
D	13lk	стандартный ввод/вывод 1 с, 256 МБ	x27
E	Peace of AmericanWedding	стандартный ввод/вывод 1 с, 256 МБ	x62
F	2715	стандартный ввод/вывод 1 с, 256 МБ	x19
G	uehSlff	стандартный ввод/вывод 2 с, 64 МБ	x20
H	Peace of AmericaReunion	стандартный ввод/вывод 1 с, 256 МБ	x171
I	Peace of AmericanPie	стандартный ввод/вывод 1 с, 256 МБ	x279
J	Common	стандартный ввод/вывод 1 с, 256 МБ	x236
K	Crap	стандартный ввод/вывод 1 с, 256 МБ	x104
L	Key	стандартный ввод/вывод 1 с, 256 МБ	x22
M	oPlus	стандартный ввод/вывод 2 с, 64 МБ	x216
N	tirnaoeumPt	стандартный ввод/вывод 1 с, 256 МБ	x95
O	0x	стандартный ввод/вывод 1 с, 256 МБ	x75
P	Prooooooooooooooofer	стандартный ввод/вывод 1 с, 256 МБ	x11
Q	Peace of bzjd	стандартный ввод/вывод 1 с, 256 МБ	x337
R	6227020800	стандартный ввод/вывод 1 с, 256 МБ	x331

2.14 OpenCup GrandPrix of Siberia

Задача 12 - Construction of Chand Baori

The Indian stepped well Chand Baori is, with some simplifications, a pyramid tapering down. Faces of the pyramid are made up of rows of trapezoid stairs which can be used to descend to reach the water. The first (lowest) level is a single stairway with two flights of stairs on the left and on the right. The second level has two such stairways, etc.

To visit the "Harshat Mata" shrine dedicated to the goddess of happiness and joy pilgrims need to cleanse themselves in the waters of the well, that is, to descend to the very bottom. Pilgrims on their way to the water can only descend or walk horizontally along the current level, but not climb up.

The builders of the well want to construct it in such a way that the number of possible ways of descend would be no less than the number of pilgrims. Two paths are considered different if there exists a level with a different stairway used for descending, or different flight of the same stairway used.

Input

The first line of the input file contains two integers N and M – the number of levels in the well and the number of pilgrims descending along the single face, respectively ($1 \le N \le 20$, $0 \le M \le 1.5 \cdot 10^{18}$).

Output

The output file must contain "`Harshat Mata`", if the number of possible ways to descend along one face of the well is less than the given number of pilgrims (also for one face), otherwise it must contain "`Nope`".

Examples

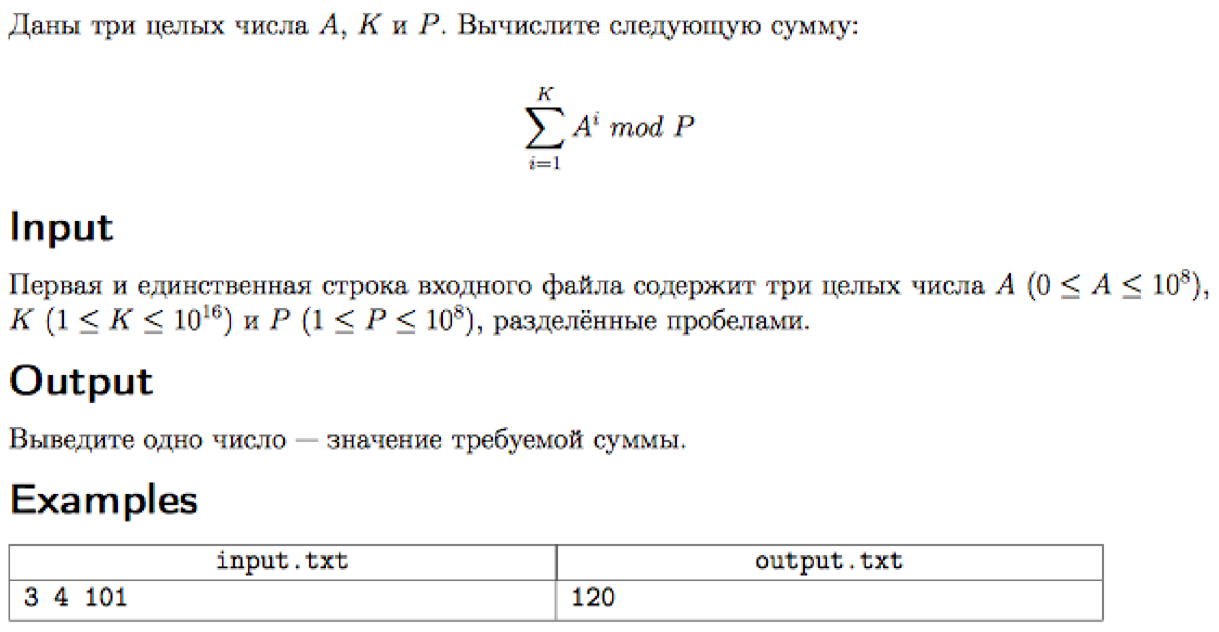
input.txt	output.txt
1 2	Harshat Mata
1 3	Nope
2 9	Nope
2 8	Harshat Mata

Алгоритм ??????????????????

Исходный код

```cpp
#include <iostream>
using namespace std;
#define ULL unsigned long long
int main(int argc, const char * argv[]) {
    ULL n, m;
    cin >> n >> m;
    ULL res = 1;
    for(int i=2; i<=n*2; i+=2)
    {
        res *= i;
    }
    if(res<m)
        cout << "Nope";
    else
        cout << "Harshat Mata";
    return 0;
}
```

Задача 13 - Sum



Даны три целых числа A, K и P. Вычислите следующую сумму:

$$\sum_{i=1}^{K} A^i \; mod \; P$$

Input

Первая и единственная строка входного файла содержит три целых числа A ($0 \le A \le 10^8$), K ($1 \le K \le 10^{16}$) и P ($1 \le P \le 10^8$), разделённые пробелами.

Output

Выведите одно число — значение требуемой суммы.

Examples

input.txt	output.txt
3 4 101	120

Алгоритм

????????????????????

Исходный код

```cpp
#include <iostream>
#include <fstream>
#include <cmath>
#include <vector>
using namespace std;
#define ULL unsigned long long
#define LL long long
int main(int argc, const char * argv[]) {
    ifstream in("input.txt");
    ofstream out("output.txt");
    ULL a, k, p;
    cin >> a >> k >> p;
    ULL sum = 0, prev = 1;
    for(ULL i = 0; i<k; ++i)
    {
        prev = (prev*a)%p;
        sum += prev;
    }
    cout << sum;
    out.close();
    in.close();
    return 0;
}
```

Задача 14 - Coinquerors

Правила старинной игры “Coinquerors” заключаются в следующем.

Каждый игрок участвует со своей монетой. Монета должна представлять собой окружность целого радиуса. Далее игроки бросают свои монеты так, что центр каждой монеты оказывается в точке с целыми координатами. После броска каждого игрока накрытый его монетой участок отмечается.

По завершении игры рассматриваются все отмеченные участки. Если у двух участников отмеченные участки пересекаются, то они объявляются союзниками. При этом гарантируется, что никакие два участка не имеют ровно одну общую точку (то есть случай касания соответствующих окружностей невозможен). Участник, набравший как можно больше союзников, и объявляется победителем.

Ваша задача — по координатам центров упавших монет и их радиусам определить победителя или же сказать, что игра завершилась вничью.

Input

Первая строка входа содержит целое число T ($1 \le T \le 20$) — количество тестовых примеров.

Далее задаются тестовые примеры. Каждый тестовый пример начинается строкой, содержащей одно целое число N ($2 \le N \le 100$).

Каждая из последующих N строк содержит имя игрока, состоящее из не менее, чем двух и не более, чем из 255 строчных латинских букв, и три целых числа X, Y и R, задающих x и y координаты центра брошенной игроком монеты и её радиус ($-100 \le X, Y \le 100$, $1 \le R \le 10$).

Output

Для каждого тестового примера выведите одну строку с имеем победившего игрока. В случае, если победителей несколько, выведите строку “`TIE`”.

Example

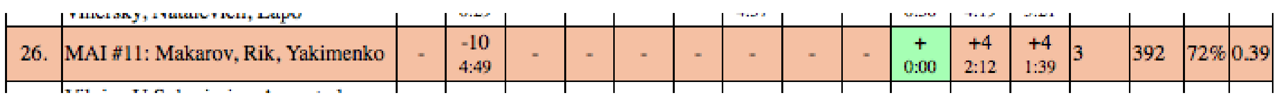
input.txt	output.txt
3 3 gennady 0 0 3 tomek 1 1 1 petr -1 -1 1 2 alice -100 -100 1 bob 100 100 100 3 john 0 0 10 john 2 2 3 jack -5 0 1	gennady TIE john

Алгоритм ????????????????????

Исходный код

```
#include <iostream>
#include <fstream>
#include <cmath>
#include <vector>
using namespace std;
#define ULL unsigned long long
#define LL long long
#define eps 0.0001
struct player
{
    char name[256];
    LL x, y, r;
};
int main(int argc, const char * argv[]) {
    ifstream in("input.txt");
    ofstream out("output.txt");
    ULL T;
    in >> T;
    double pi81 = M_PI/81, pi2 = M_PI*2;
    for(ULL TT = 0; TT<T; ++TT)
    {
        ULL n;
        in >> n;
        vector<player> pl(n);
        for(int i=0; i<n; ++i)
        {
            in >> pl[i].name >> pl[i].x >> pl[i].y >> pl[i].r;
        }
        ULL maxInd = -1, max = 0, forTie = -1;
        for(int i=0; i<n; ++i)
        {
            ULL count = 0;
            LL rr = pl[i].r*pl[i].r;
            for(int j=0; j<n; ++j)
            {
                for(double pi = 0; pi<=pi2; pi += pi81)
                {
                    double x = (pl[j].r-eps)*cos(pi);
                    double y = (pl[j].r-eps)*sin(pi);
                    if((pl[j].x-pl[i].x+x)*(pl[j].x-pl[i].x+x)+(pl[j].y-pl[i].y+y)*(
    pl[j].y-pl[i].y+y)<=rr)
                    {
                        ++count;
                        break;
                    }
                }
            }
            if(count > max)
            {
                max = count;
                maxInd = i;
                forTie = -1;
            }
            else if(count == max)
            {
                max = count;
                forTie = maxInd;
                maxInd = i;
            }
        }
        if(maxInd == -1 || forTie != -1)
            out << "TIE" << '\n';
        else
            out << pl[maxInd].name << '\n';
    }
    out.close();
    in.close();
    return 0;
}
```

Результаты

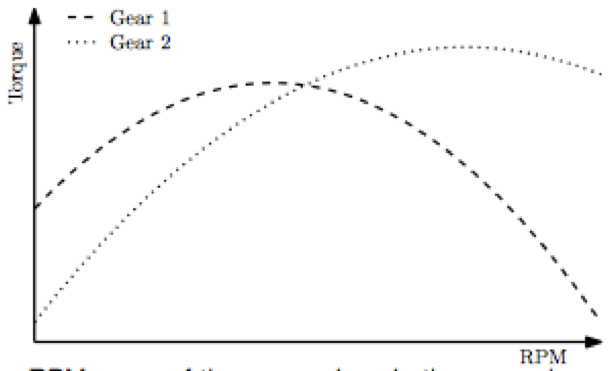


26.	MAI #11: Makarov, Rik, Yakimenko	-	-10 4:49	-	-	-	-	-	-	-	+ 0:00	+4 2:12	+4 1:39	3	392	72%	0.39

2.15 Codeforces Training S02E08

Задача G - Growling Gears

The *Best Acceleration Production Company* specializes in multi-gear engines. The performance of an engine in a certain gear, measured in the amount of torque produced, is not constant: the amount of torque depends on the RPM of the engine. This relationship can be described using a *torque-RPM curve*.



The torque-RPM curve of the gears given in the second sample input.
The second gear can produce the highest torque.

For the latest line of engines, the torque-RPM curve of all gears in the engine is a parabola of the form $T = -aR^2 + bR + c$, where R is the RPM of the engine, and T is the resulting torque.

Given the parabolas describing all gears in an engine, determine the gear in which the highest torque is produced. The first gear is gear 1, the second gear is gear 2, etc. There will be only one gear that produces the highest torque: all test cases are such that the maximum torque is at least 1 higher than the maximum torque in all the other gears.

Input

On the first line one positive number: the number of test cases, at most 100. After that per test case:

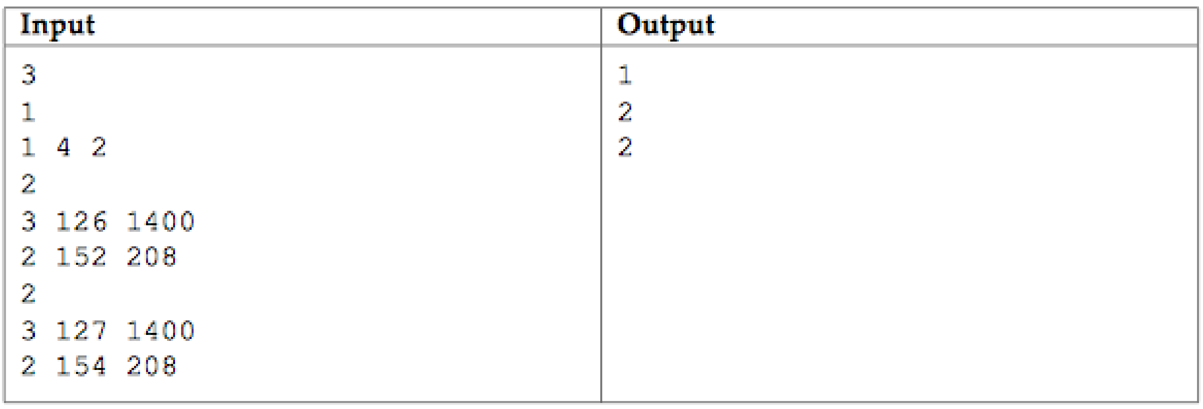
- one line with a single integer n ($1 \leq n \leq 10$): the number of gears in the engine.
- n lines, each with three space-separated integers a, b and c ($1 \leq a, b, c \leq 10\,000$): the parameters of the parabola $T = -aR^2 + bR + c$ describing the torque-RPM curve of each engine.

Output

Per test case:

- one line with a single integer: the gear in which the maximum torque is generated.

Sample in- and output



Input

```
3
1
1 4 2
2
3 126 1400
2 152 208
2
3 127 1400
2 154 208
```

Output

```
1
2
2
```

Алгоритм
???????????????????

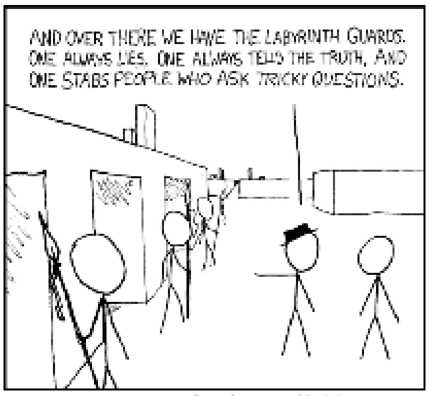
Исходный код

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

#include <fstream>

using namespace std;

int main() {
    int k;
    cin >> k;
    int n, a, b, c;
    double max_T = -1000000;
    int max_T_num;
    double temp;

    for(int i = 0; i < k; i++) {
        cin >> n;
        for(int j = 1; j <= n; j++) {
            cin >> a >> b >> c;
            temp = (b * b) / (4 * a) + c;
            if(temp > max_T) {
                max_T = temp;
                max_T_num = j;
            }
        }
        cout << max_T_num << endl;
        max_T = -1000000;
    }
  int q;
  cin >> q;
    return 0;
}
```

Задача J - Jury Jeopardy

What would a programming contest be without a problem featuring an ASCII-maze? Do not despair: one of the judges has designed such a problem.

The problem is about a maze that has exactly one entrance/exit, contains no cycles and has no empty space that is completely enclosed by walls. A robot is sent in to explore the entire maze. The robot always faces the direction it travels in. At every step, the robot will try to turn right. If there is a wall there, it will attempt to go forward instead. If that is not possible, it will try to turn left. If all three directions are unavailable, it will turn back.


AND OVER THERE WE HAVE THE LABYRINTH GUARDS. ONE ALWAYS LIES. ONE ALWAYS TELLS THE TRUTH, AND ONE STABS PEOPLE WHO ASK TRICKY QUESTIONS.

source: xkcd.com/246

The challenge for the contestants is to write a program that describes the path of the robot, starting from the entrance/exit square until it finally comes back to it. The movements are described by a single letter: 'F' means forward, 'L' is left, 'R' is right and 'B' stands for backward. Each of 'L', 'R' and 'B' does not only describe the change in orientation of the robot, but also the advancement of one square in that direction. The robot's initial direction is East. In addition, the path of the robot always ends at the entrance/exit square.

The judge responsible for the problem had completed all the samples and testdata, when disaster struck: the input file got deleted and there is no way to recover it! Fortunately the output and the samples are still there. Can you reconstruct the input from the output? For your convenience, he has manually added the number of test cases to both the sample output and the testdata output.

Input

On the first line one positive number: the number of test cases. After that per test case:

- one line with a single string: the movements of the robot through the maze.

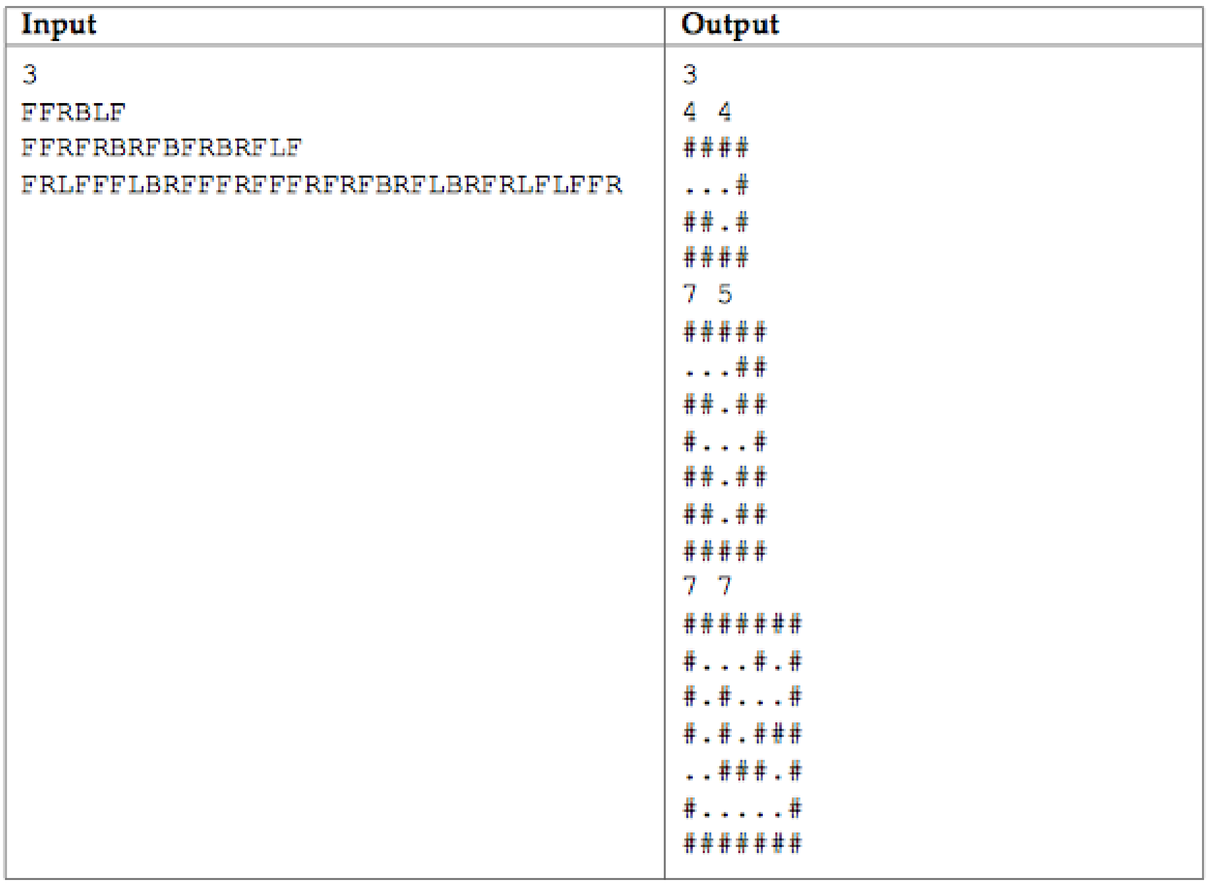
Output

On the first line one positive number: the number of test cases, at most 100. After that per test case:

- one line with two space-separated integers h and w ($3 \le h, w \le 100$): the height and width of the maze, respectively.
- h lines, each with w characters, describing the maze: a '#' indicates a wall and a '.' represents an empty square.

The entire contour of the maze consists of walls, with the exception of one square on the left: this is the entrance. The maze contains no cycles (i.e. paths that would lead the robot back to a square it had left in another direction) and no empty squares that cannot be reached from the entrance. Every row or column – with the exception of the top row, bottom row and right column – contains at least one empty square.

Sample in- and output



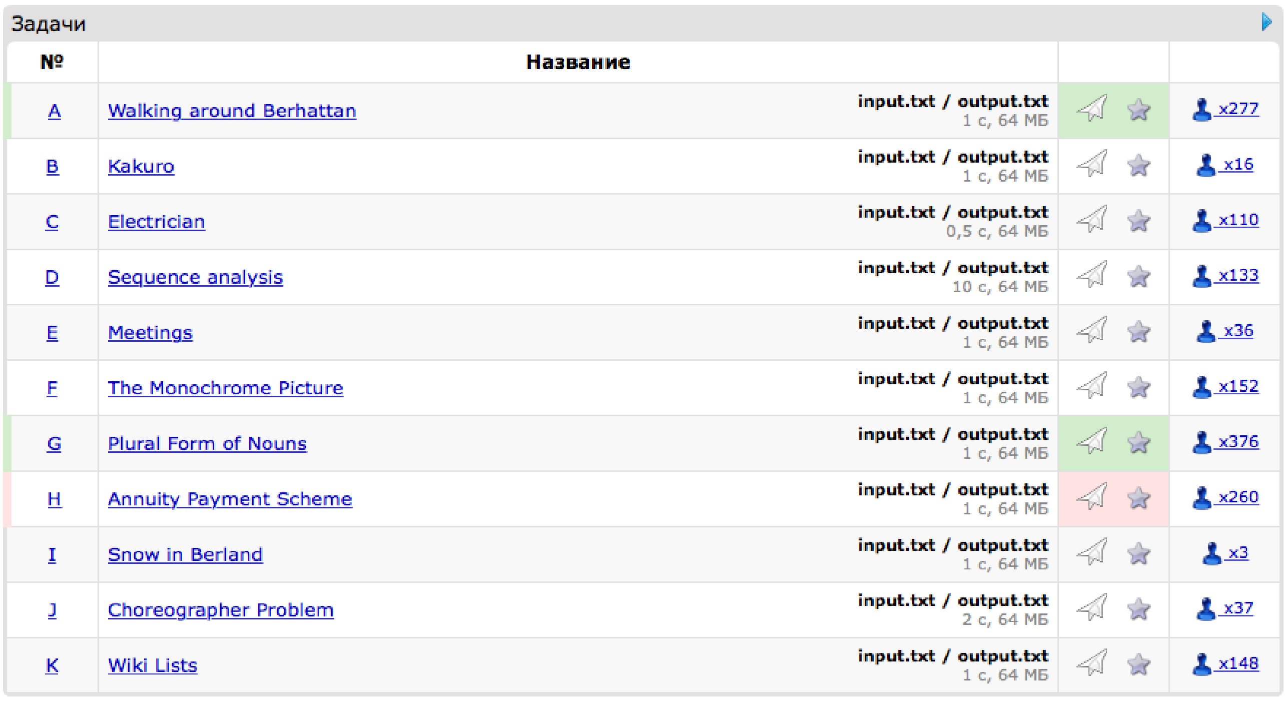
Input	Output
3	3
FFRBLF	4 4
FFRFRBRFBFRBRFLF	####
FRLFFFLBRFFFRFFFRFRFBRFLBRFRLFLFFR	...#
	##.#
	####
	7 5
	#####
	...##
	##.##
	#...#
	##.##
	##.##
	#####
	7 7
	#######
	#...#.#
	#.#...#
	#.#.###
	..###.#
	#.....#
	#######

Алгоритм
?????????????????????

Исходный код

```cpp
#include <iostream>
using namespace std;
int main(int argc, const char * argv[]) {
    long T;
    cin >> T;
    cout << T << '\n';
    char map[201][201];
    cin.get();
    while(T--)
    {
        for(long i=0; i<201; ++i)
            for(long j=0; j<201; ++j)
                map[i][j] = '#';
        char c;
        long x, y, minX, minY, maxX, maxY;
        maxX = maxY = minX = minY = x = y = 100;
        int dir = 0;
        while((c=cin.get())!='\n')
        {
            switch(dir)
            {
                case 0:
                    if(c=='R')
                        dir = 3;
                    else if(c=='L')
                        dir = 1;
                    else if(c=='B')
                        dir = 2;
                    break;
                case 1:
                    if(c=='R')
                        dir = 0;
                    else if(c=='L')
                        dir = 2;
                    else if(c=='B')
                        dir = 3;
                    break;
                case 2:
                    if(c=='R')
                        dir = 1;
                    else if(c=='L')
                        dir = 3;
                    else if(c=='B')
                        dir = 0;
                    break;
                case 3:
                    if(c=='R')
                        dir = 2;
                    else if(c=='L')
                        dir = 0;
                    else if(c=='B')
                        dir = 1;
                    break;
            }
            switch(dir)
            {
                case 0:
                    ++x;
                    if(x>maxX)
                        maxX = x;
                    break;
                case 1:
                    --y;
                    if(y<minY)
                        minY = y;
                    break;
                case 2:
                    --x;
                    if(x<minX)
                        minX = x;
                    break;
                case 3:
                    ++y;
                    if(y>maxY)
                        maxY = y;
                    break;
            }
            map[x][y] = '.';
        }
        cout << maxY-minY+3 << ' ' << maxX-minX+2 << '\n';
        for(long i = minY-1; i<=maxY+1; ++i)
        {
            for(long j = minX; j<=maxX+1; ++j)
                cout.put(map[j][i]);
            cout.put('\n');
        }
    }
    return 0;
}
```

Результаты



Задачи

№	Название			
A	Walking around Berhattan	input.txt / output.txt 1 с, 64 МБ		x277
B	Kakuro	input.txt / output.txt 1 с, 64 МБ		x16
C	Electrician	input.txt / output.txt 0,5 с, 64 МБ		x110
D	Sequence analysis	input.txt / output.txt 10 с, 64 МБ		x133
E	Meetings	input.txt / output.txt 1 с, 64 МБ		x36
F	The Monochrome Picture	input.txt / output.txt 1 с, 64 МБ		x152
G	Plural Form of Nouns	input.txt / output.txt 1 с, 64 МБ		x376
H	Annuity Payment Scheme	input.txt / output.txt 1 с, 64 МБ		x260
I	Snow in Berland	input.txt / output.txt 1 с, 64 МБ		x3
J	Choreographer Problem	input.txt / output.txt 2 с, 64 МБ		x37
K	Wiki Lists	input.txt / output.txt 1 с, 64 МБ		x148

3 Журнал по личным контестам Макарова Н.А.

3.1 Codeforces Отборочный контест СГАУ на четвертьфинал ACM-ICPC

Задача D - Игрушечные солдатики

Petya loves toy soldiers very much. He has n soldiers, and i-th soldier is painted the a_i-th color.

Petya constantly doesn't like how his soldiers are colored so he takes one of them and repaints it. He does that m times, and Vova, watching on it, becomes interested when all the soldiers have the same color for the first time. Help Vova to answer his question.

Input

The first line contains a single integer n ($1 \le n \le 10^5$) — the number of soldiers Petya has.

The second line contains n integers $a_1, \ldots, a_n$ ($1 \le a_i \le 10^9$) separated by spaces — the initial colors the soldiers were painted.

The third line contains a single integer m ($1 \le m \le 3 \cdot 10^5$) — the number of times Petya has repainted his soldiers.

The next m lines contains two integers each, separated by a space: k_j and x_j ($1 \le k_j \le n$, $1 \le x_j \le 10^9$) — the number of soldier and the number of color it has been repainted (possibly the same as before the repaintment).

Output

Output a single integer — the number of repaintments Petya has made before all his soldiers have the same color for the first time. In particular, if all soldiers had the same color before all Petya's actions, output «0». If such an event has never taken place at all, even after m-th repaintment, output «-1».

Examples

stdin	stdout
3	3
4 3 7	
4	
1 5	
2 7	
1 7	
3 3	
3	-1
6 2 7	
3	
1 1	
2 7	
1 6	

Алгоритм

???????????????????

Исходный код

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#define ll unsigned long long

using namespace std;

int main() {
ll n, m;
    ll remColor = 0;
    ll currentSoldier, currentColor, answer = 0;
    bool sameColors = true;
    cin >> n;
    vector<ll> soldiers(n + 1);
    for (ll i = 1; i <= n; i++) {
        cin >> soldiers[i];
        if (i == 1) remColor = soldiers[i];
        else if (soldiers[i] != remColor) sameColors = false;
    }
    if (sameColors) {
        cout << "0" << endl;
        return 0;
    }
    cin >> m;
    sameColors = true;
    bool flag = true;

    for (ll i = 0; i < m; i++) {
        sameColors = false;
        flag = true;
        cin >> currentSoldier >> currentColor;
        soldiers[currentSoldier] = currentColor;
        if (i == 0) remColor = currentColor;
        if (currentColor == remColor) {
            for (ll j = 1; j <= n; j++) {
                if (soldiers[j] != remColor) {
                    flag = false;
                    break;
                }
            }
            if (flag) sameColors = true;
        }
        if (sameColors) {
            answer = i + 1;
            break;
        }
        remColor = currentColor;
    }

    if (sameColors) {
        cout << answer << endl;
    }
    else {
        cout << "-1" << endl;
    }

  return 0;
}
```

Задача F - Два конверта

Mike has two envelopes. He thought up a random integer in the segment $[a, b]$ (he could think up every number from the segment with equal probability) and put exactly this amount of roubles into one of the envelopes and the double amount into the second one. Then he suggested Constantine to play the game: Constantine can open one envelope and then either take its contents or take the contents of another envelope, by his choice.

Constantine has opened one of these envelopes and has discovered c roubles there. Should he take another envelope, if he wants to maximize the expected prize?

Input

The only line contains three integers a, b and c ($1 \le a, b, c \le 10^9$), separated by spaces — the bounds of the segment from which Mike thought up his random number and the amount of roubles in the envelope opened by Constantine. It is guaranteed that the situation described by the input is possible.

Output

Output «`Take another envelope`», if it's advantageous for Constantine to take another envelope, and «`Stay with this envelope`», if it's advantageous for him to stay with the one he has already opened.

Examples

stdin	stdout
`3 5 3`	`Take another envelope`
`4 6 10`	`Stay with this envelope`

Алгоритм

???????????????????

Исходный код

```
#include <iostream>

using namespace std;

int main() {
    long long b, c;
    cin >> b >> b >> c;
    if (c > b) cout << "Stay with this envelope" << endl;
    else cout << "Take another envelope" << endl;
    return 0;
}
```

Задача G - Задача о размене монет

There are $(n+1)$ types of coins in the country R, the cheapest of which has denomination 1 and each of the next types has denomination a_i times greater than the previous one. You need to pay the sum s using as few coins as possible. Of course you can use multiple coins of the same denomination.

Input

The first line contains two integers separated by a space: n and s $(1 \leq n \leq 10^5,\ 0 \leq s \leq 10^9)$ — the number of coins' types, excluding the cheapest one, and the sum to pay.

The second line contains n integers separated by spaces: a_i $(2 \leq a_i \leq 10^9)$ — the number of times each of the next coins is more expensive than the previous one.

Output

Output a single integer — the minimum number of coins required to pay the sum s.

Examples

stdin	stdout
3 42 3 2 2	4
5 228 5 2 5 2 5	8

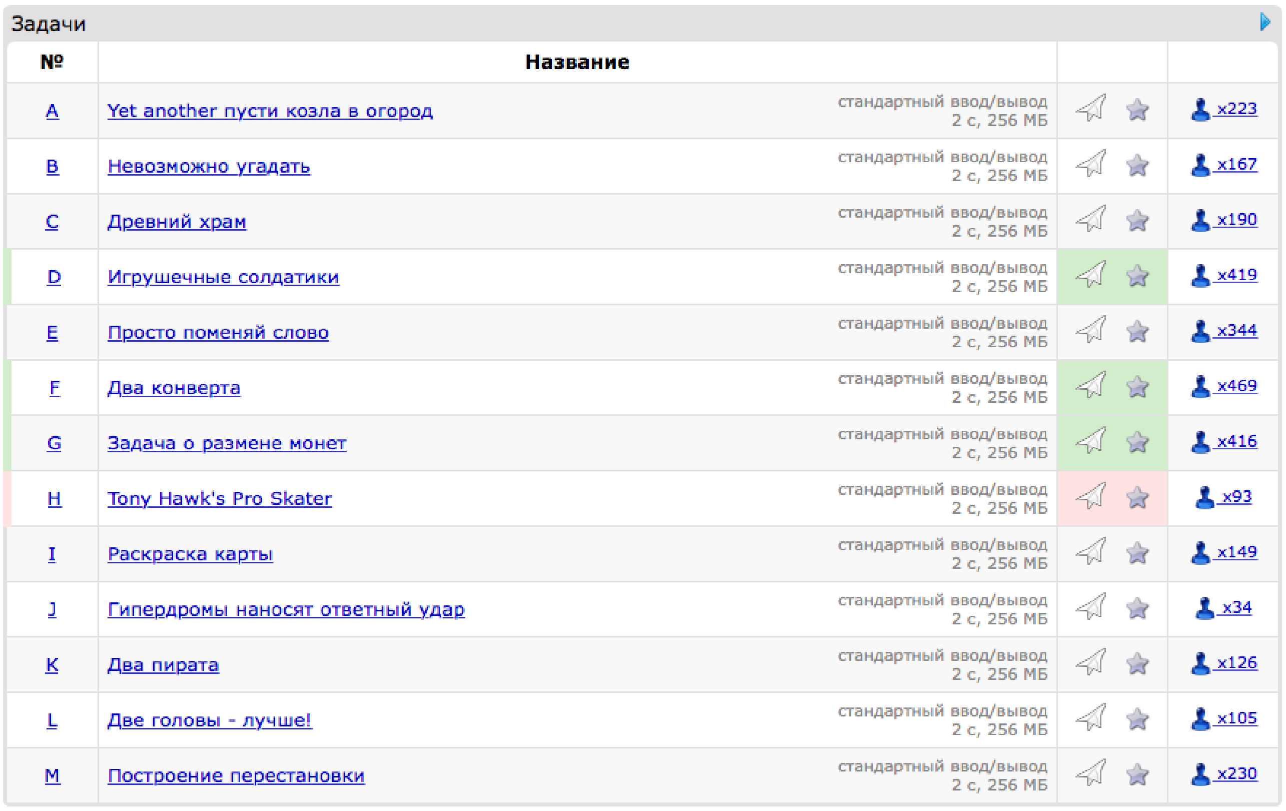
Алгоритм

???????????????????

Исходный код

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

using namespace std;

int main() {
    unsigned long long n, s, startIndex, coinsNeeded = 0;
    cin >> n >> s;
    vector<unsigned long long> k(n + 1);
    k[0] = 1;
    startIndex = n;
    for (unsigned long long i = 1; i <= n; i++) {
        cin >> k[i];
        if (k[i] * k[i - 1] > s) {
            startIndex = i - 1;
            break;
        }
        k[i] *= k[i - 1];
    }
    for (unsigned long long i = startIndex; ; i--) {
        coinsNeeded += s / k[i];
        s %= k[i];
        if (s == 0 || i == 0)
            break;
    }
    cout << coinsNeeded << endl;
    return 0;
}
```

Результаты



Задачи

№	Название			
A	Yet another пусти козла в огород	стандартный ввод/вывод 2 с, 256 МБ		x223
B	Невозможно угадать	стандартный ввод/вывод 2 с, 256 МБ		x167
C	Древний храм	стандартный ввод/вывод 2 с, 256 МБ		x190
D	Игрушечные солдатики	стандартный ввод/вывод 2 с, 256 МБ	🟩	x419
E	Просто поменяй слово	стандартный ввод/вывод 2 с, 256 МБ		x344
F	Два конверта	стандартный ввод/вывод 2 с, 256 МБ	🟩	x469
G	Задача о размене монет	стандартный ввод/вывод 2 с, 256 МБ	🟩	x416
H	Tony Hawk's Pro Skater	стандартный ввод/вывод 2 с, 256 МБ	🟥	x93
I	Раскраска карты	стандартный ввод/вывод 2 с, 256 МБ		x149
J	Гипердромы наносят ответный удар	стандартный ввод/вывод 2 с, 256 МБ		x34
K	Два пирата	стандартный ввод/вывод 2 с, 256 МБ		x126
L	Две головы - лучше!	стандартный ввод/вывод 2 с, 256 МБ		x105
M	Построение перестановки	стандартный ввод/вывод 2 с, 256 МБ		x230

4 Журнал по личным контестам Якименко А.В.