

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ  
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)



Отчет по индивидуальному учебному плану

---

## Алгоритмы на графах

---

Студенты:

Макаров Никита  
Якименко Антон

Руководитель:

Зайцев В.Е.

# Содержание

<b>1</b>	<b>Личные отчеты</b>	<b>2</b>
<b>2</b>	<b>Журнал по командным контестам</b>	<b>5</b>
2.1	Codeforces Training S02E03 . . . . .	5
2.2	Codeforces Training S02E04 . . . . .	12
2.3	XV Открытая Всесибирская олимпиада по программированию им И.В. Потосина . . . . .	13
2.4	Codeforces Training S02E05 . . . . .	15
2.5	Codeforces Training S02E06 . . . . .	22
2.6	Тренировка СПбГУ Поиск кратчайшего пути и DFS . . . . .	23
2.7	OpenCup GrandPrix of SPb. . . . .	28
2.8	Самарский Международный Аэрокосмический Лицей, тренировка №1 . . . . .	32
2.9	Codeforces ACM-ICPC Восточный четвертьфинал . . . . .	34
2.10	Codeforces ACM-ICPC Южный четвертьфинал . . . . .	45
2.11	ACM-ICPC Московский четвертьфинал . . . . .	54
2.12	Codeforces Training S02E07 . . . . .	55
2.13	Codeforces Crypto Cup 1.0 . . . . .	58
2.14	OpenCup GrandPrix of Siberia . . . . .	70
2.15	Codeforces Training S02E08 . . . . .	76
2.16	Codeforces Training S02E09 . . . . .	82
2.17	Codeforces Олимпиада школьников Нижегородской обл. . . . .	88
2.18	OpenCup GrandPrix of Central Europe . . . . .	94
2.19	Codeforces Training S02E10 . . . . .	96
2.20	OpenCup GrandPrix of Europe . . . . .	103
2.21	OpenCup GrandPrix of Peterhof . . . . .	114
2.22	OpenCup GrandPrix of Japan . . . . .	117
2.23	OpenCup Northern GrandPrix . . . . .	126
<b>3</b>	<b>Журнал по личным контестам Макарова Н.А.</b>	<b>132</b>
3.1	Codeforces Отборочный контест СГАУ на четвертьфинал ACM-ICPC . . . . .	132
<b>4</b>	<b>Журнал по личным контестам Якименко А.В.</b>	<b>138</b>

# 1 Личные отчеты

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

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

## Задача В - 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 основанию 2 числа  $n$  меньше или равен  $b$ , то ответ yes, иначе ответ no.

## Исходный код

```
1 #include <iostream>
2 #include <cmath>
3
4 using namespace std;
5
6 int main() {
7     ios_base::sync_with_stdio(false);
8
9     unsigned long long n;
10    int b;
11    cin >> n >> b;
12
13    if ((int)log2((double)n) <= b) {
14        cout << "yes" << endl;
15    }
16    else {
17        cout << "no" << endl;
18    }
19
20    return 0;
21 }
```

## 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

```
1 #include <iostream>
2 #include <vector>
3
4 #define ll long long
5 using namespace std;
6
7 int main () {
8     ll N,B,H,W,p,a;
9     ll min_cost = 5000000;
10    cin >> N >> B >> H >> W;
11
12    for (ll i = 0; i < H; i++) {
13        cin >> p;
14        for (ll k = 0; k < W; k++) {
15            cin >> a;
16            if ((a >= N) && (p * N <= B) && (p * N <= min_cost))
17                min_cost = p * N;
18        }
19    }
20    if (min_cost < 5000000)
21        cout << min_cost << endl;
22    else cout << "stay home" << endl;
23
24
25    return 0;
26 }
27 }
```

## Задача K - Best Cow Line

FJ is about to take his N ( $1 \leq N \leq 30,000$ ) 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

```
1 #include <iostream>
2 #include <cmath>
3 #include <vector>
4 #include <stack>
5 #include <algorithm>
6
7 using namespace std;
8
9 int main() {
10     int n;
11     cin >> n;
12     vector<char> cows(n);
13     char symb;
14
15     for (int i = 0; i < n; i++) {
16         cin >> symb;
17         cows[i] = symb;
18     }
19
20     vector<char> newLine;
21     int cowsInOldLine = n;
22     int begin = 0;
23     int end = n - 1;
24     int tempBegin = begin;
25     int tempEnd = end;
26
27     while (cowsInOldLine) {
28         tempBegin = begin;
29         tempEnd = end;
30         if (cows[begin] == cows[end]) {
31             while (cows[tempBegin] == cows[tempEnd]) {
32                 tempBegin++;
33                 tempEnd--;
34                 if (tempBegin > tempEnd || tempBegin == tempEnd) {
35                     tempBegin = begin;
36                     tempEnd = end;
37                     break;
38                 }
39             }
40         }
41         if (cows[tempBegin] < cows[tempEnd]) {
42             newLine.push_back(cows[begin]);
43             begin++;
44         }
45         else {
46             newLine.push_back(cows[end]);
47             end--;
48         }
49         cowsInOldLine--;
50     }
51     for (int i = 0; i < n; i++) {
52         cout << newLine[i];
53         if ((i + 1) % 80 == 0) cout << endl;
54     }
55     cout << endl;
56     return 0;
57 }
```

## 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

```
1 #include <iostream>
2 #include <algorithm>
3 #include <cmath>
4
5 #include <sstream>
6 #include <fstream>
7
8 #define LL long long
9
10 using namespace std;
11
12 int main() {
13     ifstream in;
14     ofstream out;
15     in.open("input.txt");
16     out.open("output.txt");
17
18     LL a, b;
19     in >> a >> b;
20     char pic[a + 1][b + 1];
21     LL XMax = 0;
22     LL YMax = 0;
23
24     for (LL i = 1; i <= a; i++) {
25         for (LL j = 1; j <= b; j++) {
26             in >> pic[i][j];
27             if (pic[i][j] == '#' && j > XMax) XMax = j;
28             if (pic[i][j] == '#' && i > YMax) YMax = i;
29         }
30         in.get();
31     }
32     LL n, y, x;
33     in >> n;
34     for (LL i = 1; i <= n; i++) {
35         in >> y >> x;
36         if (y >= YMax && x >= XMax) {
37             out << i << endl;
38             return 0;
39         }
40     }
41     in.close();
42     out.close();
43
44     return 0;
45 }
```

## 2.4 Codeforces Training S02E05

### Результаты

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

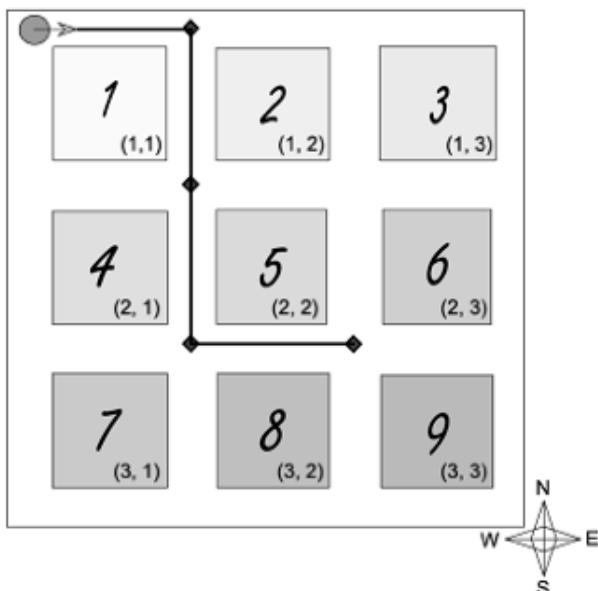
## Задача A - Walking around Berhattan

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  $m$  ( $1 \leq n, m \leq 100$ ), 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  $b_{ij}$  ( $0 \leq b_{ij} \leq 9$ ) — 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

```
1 #include <iostream>
2 #include <algorithm>
3 #include <iomanip>
4 #include <vector>
5 #include <sstream>
6 #include <fstream>
7 #define LL long long
8
9 using namespace std;
10
11 enum dtype {
12     UP,
13     DOWN,
14     LEFT,
15     RIGHT,
16 };
17
18 int main() {
19     ifstream in;
20     ofstream out;
21     in.open("input.txt");
22     out.open("output.txt");
23
24     LL n, m;
25     LL answer = 0;
26     in >> n >> m;
27     char t;
28     in.get();
29
30     vector<vector<int>> map(n + 2, vector<int>(m + 2, 0));
31     vector<vector<bool>> used(n + 2, vector<bool>(m + 2, false));
32
33     for (LL i = 1; i <= n; i++) {
34         for (LL j = 1; j <= m; j++) {
35             t = in.get();
36             map[i][j] = t - '0';
37
38             if (t == 'R') {
39                 if (used[i][j + 1]) continue;
40                 used[i][j + 1] = true;
41                 answer += map[i][j];
42             }
43             else if (t == 'L') {
44                 if (used[i][j - 1]) continue;
45                 used[i][j - 1] = true;
46                 answer += map[i][j];
47             }
48             else if (t == 'U') {
49                 if (used[i - 1][j]) continue;
50                 used[i - 1][j] = true;
51                 answer += map[i][j];
52             }
53             else if (t == 'D') {
54                 if (used[i + 1][j]) continue;
55                 used[i + 1][j] = true;
56                 answer += map[i][j];
57             }
58         }
59     }
60
61     cout << answer;
62 }
```

```

37     }
38     in.get();
39   }
40
41   int x = 1, y = 1;
42   dtype dir = RIGHT; // last dir
43
44   while ((t = in.get()) != EOF) {
45     if (t == 'M') {
46       if (dir == RIGHT) {
47         answer += map[x][y];
48         answer += map[x - 1][y];
49         if (!used[x][y]) {
50           map[x][y] /= 2;
51           used[x][y] = true;
52         }
53         if (!used[x - 1][y]) {
54           map[x - 1][y] /= 2;
55           used[x - 1][y] = true;
56         }
57         y++;
58       }
59     else if (dir == LEFT) {
60       answer += map[x][y - 1];
61       answer += map[x - 1][y - 1];
62       if (!used[x][y - 1]) {
63         map[x][y - 1] /= 2;
64         used[x][y - 1] = true;
65       }
66       if (!used[x - 1][y - 1]) {
67         map[x - 1][y - 1] /= 2;
68         used[x - 1][y - 1] = true;
69       }
70       y--;
71     }
72     else if (dir == UP) {
73       answer += map[x - 1][y - 1];
74       answer += map[x - 1][y];
75       if (!used[x - 1][y - 1]) {
76         map[x - 1][y - 1] /= 2;
77         used[x - 1][y - 1] = true;
78       }
79       if (!used[x - 1][y]) {
80         map[x - 1][y] /= 2;
81         used[x - 1][y] = true;
82       }
83       x--;
84     }
85     else if (dir == DOWN) {
86       answer += map[x][y];
87       answer += map[x][y - 1];
88       if (!used[x][y]) {
89         map[x][y] /= 2;
90         used[x][y] = true;
91       }
92       if (!used[x][y - 1]) {
93         map[x][y - 1] /= 2;
94         used[x][y - 1] = true;
95       }
96       x++;

```

```

97         }
98     }
99
100    else if (t == 'R') {
101        if (dir == UP) dir = RIGHT;
102        else if (dir == DOWN) dir = LEFT;
103        else if (dir == LEFT) dir = UP;
104        else dir = DOWN;
105    }
106    else if (t == 'L'){
107        if (dir == UP) dir = LEFT;
108        else if (dir == DOWN) dir = RIGHT;
109        else if (dir == LEFT) dir = DOWN;
110        else dir = UP;
111    }
112 }
113 out << answer << endl;
114 in.close();
115 out.close();
116 return 0;
117 }
```

## 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  $n$  ( $1 \leq n \leq 10$ ) — 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	contests
contest	heroes
hero	ladies
lady	

### Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 #include <sstream>
3 #include <fstream>
4 #define LL long long
5
6 using namespace std;
7
8 int main() {
9     ifstream in;
10    ofstream out;
11    in.open("input.txt");
12    out.open("output.txt");
13
14    LL n;
15    string s;
16
17    in >> n;
18
19    for (LL i = 0; i < n; i++) {
20        in >> s;
21        size_t l = s.size() - 1;
22        if ((s[1] == 'h' && s[l - 1] == 'c') || s[1] == 's' || s[1] == 'x' || s[1] ==
23             'o') {
24            out << s;
25            out << "es" << endl;
26        }
27        else if (s[1] == 'f') {
28            for (size_t j = 0; j < l; j++) {
29                out << s[j];
30            }
31            out << "ves" << endl;
32        }
33        else if (s[1] == 'e' && s[l - 1] == 'f') {
34            for (size_t j = 0; j < l - 1; j++) {
35                out << s[j];
36            }
37            out << "ves" << endl;
38        }
39        else if (s[1] == 'y') {
40            for (size_t j = 0; j < l; j++) {
41                out << s[j];
42            }
43            out << "ies" << endl;
44        }
45        else {
46            out << s << "s" << endl;
47        }
48
49    in.close();
50    out.close();
51
52    return 0;
53 }
```

## 2.5 Codeforces Training S02E06

### Результаты

Задачи			
№	Название		
A	<a href="#">Average distance</a>	стандартный ввод/вывод 1 с, 256 МБ	
B	<a href="#">Bus Pass</a>	стандартный ввод/вывод 1 с, 256 МБ	
C	<a href="#">Cutting Banknotes</a>	стандартный ввод/вывод 0,5 с, 256 МБ	
D	<a href="#">Dice Password Security</a>	стандартный ввод/вывод 0,5 с, 256 МБ	
E	<a href="#">Lingo</a>	стандартный ввод/вывод 10 с, 256 МБ	
F	<a href="#">Splitting the Loot</a>	стандартный ввод/вывод 0,5 с, 256 МБ	
G	<a href="#">Pachinko</a>	стандартный ввод/вывод 0,5 с, 256 МБ	
H	<a href="#">Hiking</a>	стандартный ввод/вывод 12 с, 256 МБ	
I	<a href="#">Ranking</a>	стандартный ввод/вывод 0,5 с, 256 МБ	
J	<a href="#">Stock</a>	стандартный ввод/вывод 2 с, 256 МБ	

## 2.6 Тренировка СПбГУ Поиск кратчайшего пути и DFS

### Результаты

Задачи					
№	Название				
A	<a href="#">Кратчайший путь Easy<sup>1</sup></a>		path_easy.in / path_easy.out	4 с, 256 МБ	
B	<a href="#">Цикл отрицательного веса<sup>1</sup></a>		negcycle.in / negcycle.out	2 с, 256 МБ	
C	<a href="#">Флойд<sup>1</sup></a>		floyd.in / floyd.out	2 с, 256 МБ	
D	<a href="#">Поиск цикла<sup>1</sup></a>		cycle.in / cycle.out	2 с, 256 МБ	
E	<a href="#">Противопожарная безопасность<sup>1</sup></a>		firesafe.in / firesafe.out	2 с, 256 МБ	

## Задача С - 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  $N$  ( $1 \leq N \leq 100$ ) — 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

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

```
1 #include <iomanip>
2 #include <iostream>
3 #include <algorithm>
4 #include <fstream>
5
6 using namespace std;
7
8 int main() {
9
10    more_speed
11    ifstream in("floyd.in");
12    ofstream out("floyd.out");
13
14    int n;
15    in >> n;
16    vector<vector<int>> m(n, vector<int>(n, 0));
17    for (int i = 0; i < n; i++) {
18        for (int j = 0; j < n; j++) {
19            in >> m[i][j];
20        }
21    }
22
23    for (int k = 0; k < n; k++) {
24        for (int i = 0; i < n; i++) {
25            for (int j = 0; j < n; j++) {
26                m[i][j] = min(m[i][j], m[i][k] + m[k][j]);
27            }
28        }
29    }
30
31    for (int i = 0; i < n; i++) {
32        for (int j = 0; j < n; j++) {
33            out << m[i][j];
34            if (j < n - 1) out << " ";
35            else out << endl;
36        }
37    }
38
39    in.close();
40    out.close();
41
42    return 0;
43 }
```

## 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  $M$  ( $1 \leq N \leq 100\,000$ ,  $M \leq 100\,000$ ) — 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

```
1 #include <iomanip>
2 #include <iostream>
3 #include <algorithm>
4 #include <fstream>
5
6 vector<set<LL>> g;
7 vector<char> color;
8 vector<LL> p;
9 LL cycle_st, cycle_end;
10
11 bool dfs (LL v) {
12     color[v] = 1;
13     for (set<LL>::iterator i = g[v].begin(); i != g[v].end(); i++) {
14         LL to = *i;
15         if (color[to] == 0) {
16             p[to] = v;
17             if (dfs(to)) return true;
18         }
19         else if (color[to] == 1){
20             cycle_st = to;
21             cycle_end = v;
22             return true;
23         }
24     }
25     color[v] = 2;
26     return false;
```

```

27 }
28
29 using namespace std;
30 int main() {
31
32     more_speed
33     ifstream in("cycle.in");
34     ofstream out("cycle.out");
35
36     LL n, m, f, t;
37     in >> n >> m;
38     g.resize(n);
39
40     for (LL i = 0; i < m; i++) {
41         in >> f >> t;
42         g[f - 1].insert(t - 1);
43     }
44
45     p.assign(n, -1);
46     color.assign(n, 0);
47     cycle_st = -1;
48     for (LL i = 0; i < n; i++) {
49         if (dfs(i)) break;
50     }
51     if (cycle_st == -1) {
52         out << "NO" << endl;
53     }
54     else {
55         out << "YES" << endl;
56         vector<LL> cycle;
57         for (LL v = cycle_end; v != cycle_st; v = p[v]) {
58             cycle.push_back(v);
59         }
60         cycle.push_back(cycle_st);
61         reverse(cycle.begin(), cycle.end());
62         for (size_t i = 0; i < cycle.size(); i++) {
63             out << cycle[i] + 1 << " ";
64         }
65         out << endl;
66     }
67     in.close();
68     out.close();
69
70     return 0;
71 }
```

## 2.7 OpenCup GrandPrix of SPb.

### Задача А - 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

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

## Алгоритм

Задача на реализацию. Нужно считать строки и каждой строке сопоставить правильное сочетание цвета и предмета по заданным 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

```
1 #include <iomanip>
2 #include <iostream>
3 #include <algorithm>
4
5 bool isChar(char c) {
6     return (c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z');
7 }
8
9 int num(string s) {
10    if (s == "white" || s == "barabashka") return 0;
11    else if (s == "blue" || s == "book") return 1;
12    else if (s == "red" || s == "chair") return 2;
13    else if (s == "gray" || s == "mouse") return 3;
14    else return 4;
15 }
16
17 int main() {
18    ifstream in("barabashka.in");
19    ofstream out("barabashka.out");
20
21    string white = "white",
22          blue = "blue",
23          red = "red",
24          gray = "gray",
25          green = "green";
26    string barab = "barabashka",
27          book = "book",
28          chair = "chair",
29          mouse = "mouse",
30          bottle = "bottle";
31
32    string current;
33    string firstColor, firstObject;
34    string secondColor, secondObject;
35    bool needFirstColor, needFirstObject;
36    bool needSecondColor, needSecondObject;
37    bool complete;
38
39
40    for (int i = 0; i < 5; i++) {
41        char cSymb = in.get();
42        bool used[5] = {false};
43        needFirstColor = true;
44        needFirstObject = false;
45        needSecondColor = false;
46        needSecondObject = false;
47        complete = false;
48
49        while (cSymb != '0') {
50            while (cSymb != ',' && isChar(cSymb)) {
51                current += tolower(cSymb);
52                cSymb = in.get();
```

```

53
54         }
55         if (needFirstColor) {
56             if (current == white) { firstColor = white; needFirstObject =
57                 true; needFirstColor = false; }
58             else if (current == blue) { firstColor = blue; needFirstObject =
59                 true; needFirstColor = false; }
60             else if (current == red) { firstColor = red; needFirstObject =
61                 true; needFirstColor = false; }
62             else if (current == gray) { firstColor = gray; needFirstObject =
63                 true; needFirstColor = false; }
64             else if (current == green) { firstColor = green; needFirstObject =
65                 true; needFirstColor = false; }
66             }
67             else if (needFirstObject) {
68                 if (current == barab) { firstObject = barab; needSecondColor =
69                     true; needFirstObject = false; }
70                 else if (current == book) { firstObject = book; needSecondColor =
71                     true; needFirstObject = false; }
72                 else if (current == chair) { firstObject = chair; needSecondColor =
73                     true; needFirstObject = false; }
74                 else if (current == mouse) { firstObject = mouse; needSecondColor =
75                     true; needFirstObject = false; }
76                 else if (current == bottle) { firstObject = bottle; needSecondColor =
77                     true; needFirstObject = false; }
78                 }
79                 else if (needSecondColor) {
80                     if (current == white) { secondColor = white; needSecondObject =
81                         true; needSecondColor = false; }
82                     else if (current == blue) { secondColor = blue; needSecondObject =
83                         true; needSecondColor = false; }
84                     else if (current == red) { secondColor = red; needSecondObject =
85                         true; needSecondColor = false; }
86                     else if (current == gray) { secondColor = gray; needSecondObject =
87                         true; needSecondColor = false; }
88                     else if (current == green) { secondColor = green; needSecondObject =
89                         true; needSecondColor = false; }
90                     }
91                     if (complete) {
92                         cSymb = '0';
93                         used[num(firstColor)] = true;
94                         used[num(firstObject)] = true;
95                         used[num(secondColor)] = true;
96                         used[num(secondObject)] = true;
97
98                         if ((firstColor == white && firstObject == barab) ||
99                             (secondColor == white && secondObject == barab)) { out << white
100                            << " " << "Barabashka" << endl; }

```

```

92             else if ((firstColor == blue && firstObject == book) ||
93                         (secondColor == blue && secondObject == book)) { out << blue
94                         << " " << book << endl; }
95
96             else if ((firstColor == red && firstObject == chair) ||
97                         (secondColor == red && secondObject == chair)) { out << red
98                         << " " << chair << endl; }
99
100            else if ((firstColor == gray && firstObject == mouse) ||
101                         (secondColor == gray && secondObject == mouse)) { out <<
102                         gray << " " << mouse << endl; }
103
104            else {
105                for (int j = 0; j < 5; j++) {
106                    if (!used[j]) {
107                        if (j == 0) { out << white << " " << "Barabashka" << endl
108                            ; }
109                        else if (j == 1) { out << blue << " " << book << endl; }
110                        else if (j == 2) { out << red << " " << chair << endl; }
111                        else if (j == 3) { out << gray << " " << mouse << endl; }
112                        else { out << green << " " << bottle << endl;
113                            break;
114                        }
115                    }
116
117                }
118            }
119            current.clear();
120            if (cSymb != '\0') cSymb = in.get();
121        }
122    }
123    return 0;
124 }

```

## Результаты

МАКАРОВ РИКАРДОВИЧ		1:20	4:44			4:31							
44.	MAI #11: Makarov, Rik, Yakimenko	+ 1:52	-9 4:54	- 2	- 2	- 2	- 2	-2 3:32	- 2	- 2	1 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  $a_1, \dots, a_n$  ( $1 \leq a_i \leq 10^9$ ) — 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

```
1 #include <iostream>
2 #include <vector>
3 #include <algorithm>
4 #include <fstream>
5
6 using namespace std;
7
8 int main() {
9     long n, k;
10    ifstream in("input.txt");
11    ofstream out("output.txt");
12    in >> n >> k;
13    vector<long long> v(n);
14    for (long i = 0; i < n; i++) {
15        in >> v[i];
16    }
17    sort(v.begin(), v.end(), greater<long long>());
18    long long answer = 0;
19    for (long i = 0; i < k && i < v.size(); i++) {
20        answer += v[i];
21    }
22    out << answer << endl;
23    in.close();
24    out.close();
25    return 0;
26 }
```

## Результаты

Задачи					
№	Название				
A	<a href="#">Маленькие кубики<sup>1</sup></a>	<a href="#">input.txt</a>	/ <a href="#">output.txt</a>	1 с, 64 МБ	
B	<a href="#">Отличные числа<sup>1</sup></a>	<a href="#">input.txt</a>	/ <a href="#">output.txt</a>	1 с, 64 МБ	
C	<a href="#">Числа-палиндромы<sup>1</sup></a>	<a href="#">input.txt</a>	/ <a href="#">output.txt</a>	1 с, 64 МБ	
D	<a href="#">Пивной вор<sup>1</sup></a>	<a href="#">input.txt</a>	/ <a href="#">output.txt</a>	1 с, 64 МБ	
E	<a href="#">Максимальный поток<sup>1</sup></a>	<a href="#">input.txt</a>	/ <a href="#">output.txt</a>	1 с, 64 МБ	

## 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

```
1 #include <iostream>
2 #include <vector>
3 #include <algorithm>
4 #include <fstream>
5
6 using namespace std;
7
8 int main() {
9     long n, k;
10    ifstream in("input.txt");
11    ofstream out("output.txt");
12    in >> n >> k;
13    vector<long long> v(n);
14    for (long i = 0; i < n; i++) {
15        in >> v[i];
16    }
17    sort(v.begin(), v.end(), greater<long long>());
18    long long answer = 0;
19    for (long i = 0; i < k && i < v.size(); i++) {
20        answer += v[i];
21    }
22    out << answer << endl;
23    in.close();
24    out.close();
25    return 0;
26 }
```

## 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 ( $0 \leq n \leq 256$ ). 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 ( $1 \leq m \leq 256$ ). 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

```
1 #include <iostream>
2 #include <algorithm>
3 #include <vector>
4 #include <fstream>
5
6 using namespace std;
7
8
9 typedef struct {
10     long money;
11     long ad;
12     int num;
13 }frd;
14
15 typedef struct {
16     long rooms;
17     long price;
18     long ad;
19     int num;
20 }apartment;
21
22 bool cmp_ap( const apartment &a1, const apartment &a2) {
23     return a1.ad > a2.ad;
24 }
25
26 bool cmp_fr( const frd &f1, const frd &f2) {
27     return f1.ad > f2.ad;
28 }
29
30 int find_good_ap( const vector<apartment> &a, long money1, long money2) {
31     for (int i = 0; i < a.size(); i++) {
32         long price_for_each = a[i].price / 2;
33         if (a[i].price % 2 == 1) {
34             price_for_each++;
35         }
36         if (money1 >= price_for_each && money2 >= price_for_each && a[i].rooms == 2)
37             return i;
38     }
39     return -1;
40 }
41
42 int main() {
43     long money, ad1, ad2;
44     long n, m;
45     cin >> money >> ad1 >> ad2;
46     cin >> n;
47     vector<frd> fr(n);
48     for (int i = 0; i < n; i++) {
49         cin >> fr[i].money >> fr[i].ad;
50         fr[i].num = i + 1;
51     }
52     cin >> m;
53     vector<apartment> ap(m);
54     long max_ad_in_1room = -1, max_ad_in_2room = -1;
55     int in_1room_num = -1, in_2room_num = -1;
56     bool can_buy_alone = false;
57     for (int i = 0; i < m; i++) {
58         cin >> ap[i].rooms >> ap[i].price >> ap[i].ad;
59         ap[i].num = i + 1;
```

```

60     if (ap[ i ].rooms == 1) {
61         if (money >= ap[ i ].price && ad1 + ap[ i ].ad > max_ad_in_1room) {
62             max_ad_in_1room = ad1 + ap[ i ].ad;
63             in_1room_num = i + 1;
64         }
65     }
66     else {
67         if (money >= ap[ i ].price && ad2 + ap[ i ].ad > max_ad_in_2room) {
68             max_ad_in_2room = ad2 + ap[ i ].ad;
69             in_2room_num = i + 1;
70         }
71     }
72 }
73 sort(ap.begin(), ap.end(), cmp_ap);
74 int ans_ap = -1, ans_fr = -1;
75 long max_ad_tog = -1;
76 int found_ap = -1;
77 for (int i = 0; i < n; i++) {
78     found_ap = find_good_ap(ap, money, fr[ i ].money);
79     if (found_ap != -1) {
80         long cur_ad = fr[ i ].ad + ap[ found_ap ].ad;
81         if (cur_ad > max_ad_tog) {
82             ans_ap = ap[ found_ap ].num;
83             ans_fr = fr[ i ].num;
84             max_ad_tog = cur_ad;
85         }
86     }
87 }
88
89 long alone = 0, whereAlone = 0;
90 if (max_ad_in_1room != -1 || max_ad_in_2room != -1) {
91     if (max_ad_in_1room > max_ad_in_2room) {
92         alone = max_ad_in_1room;
93         whereAlone = in_1room_num;
94     }
95     else {
96         alone = max_ad_in_2room;
97         whereAlone = in_2room_num;
98     }
99     can_buy_alone = true;
100 }
101
102 if (found_ap == -1 && !can_buy_alone) {
103     cout << "Forget about apartments. Live in the dormitory." << endl;
104     return 0;
105 }
106
107 if (alone > max_ad_tog)
108     cout << "You should rent the apartment #" << whereAlone << " alone." << endl;
109 else
110     cout << "You should rent the apartment #" << ans_ap << " with the friend #"
111     << ans_fr << "." << endl;
112
113 }

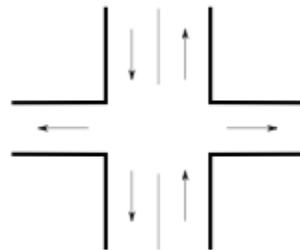
```

## 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-paved, 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	4
FF	
L	1
L	

### Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 #include <deque>
3
4 using namespace std;
5
6 int main()
7 {
8     deque <char> south;
9     deque <char> north;
10
11    char temp;
12    int time = 0;
13    temp = cin.get();
14    while(temp != '\n'){
15        south.push_back(temp);
16        temp = cin.get();
17    }
18    temp = cin.get();
19    while(temp != '\n'){
20        north.push_back(temp);
21        temp = cin.get();
22    }
23    char s, n;
24    while(south.size() > 0 && north.size() > 0){
25        s = south[0];
26        n = north[0];
27        time++;
28        if(s == 'R' && n == 'L')
29            south.pop_front();
30        else if(s == 'L' && n == 'R')
31            north.pop_front();
32        else if(s == 'L' && n == 'F')
33            north.pop_front();
34        else if(s == 'F' && n == 'L')
35            south.pop_front();
36        else {
37            south.pop_front();
38            north.pop_front();
39        }
40    }
41
42    if(south.size() > 0)
43        time += south.size();
44    if(north.size() > 0)
45        time += north.size();
46
47    cout << time << endl;
48    return 0;
49 }
```

## Задача L - Donald is a postman

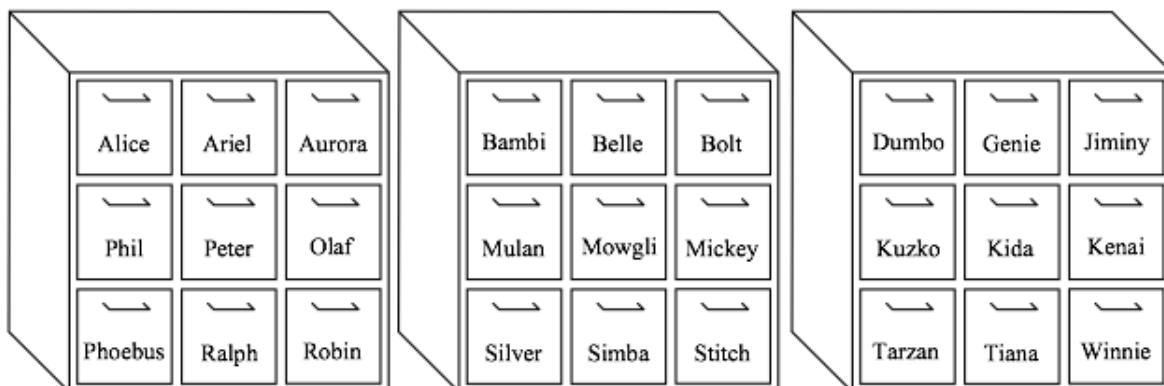
Time limit: 1 second

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 ( $1 \leq n \leq 1000$ ). 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

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main()
6 {
7     int n;
8     cin >> n;
9     string name;
10    int state = 1;
11    int answer = 0;
12    for (int i = 0; i < n; i++) {
13        cin >> name;
14        char t = name[0];
15        if (state == 1) {
16            if (t == 'B' || t == 'M' || t == 'S') {
17                answer++;
18                state = 2;
19            }
20            else if (t == 'D' || t == 'J' || t == 'K' || t == 'T' || t == 'W' || t ==
21 'G') {
22                answer += 2;
23                state = 3;
24            }
25            else if (state == 2) {
26                if (t == 'A' || t == 'P' || t == 'O' || t == 'R') {
27                    answer++;
28                    state = 1;
29                }
30                else if (t == 'D' || t == 'J' || t == 'K' || t == 'T' || t == 'W' || t ==
31 'G') {
32                    answer++;
33                    state = 3;
34                }
35            }
36            else {
37                if (t == 'A' || t == 'P' || t == 'O' || t == 'R') {
38                    answer += 2;
39                    state = 1;
40                }
41                else if (t == 'B' || t == 'M' || t == 'S') {
42                    answer++;
43                    state = 2;
44                }
45            }
46        }
47        cout << answer << endl;
48    return 0;
49 }
50 }
```

# Результаты

Задачи				
№	Название			
A	<a href="#">About Grisha N.</a>	стандартный ввод/вывод 1 с, 64 МБ		x489
B	<a href="#">Neither shaken nor stirred</a>	стандартный ввод/вывод 1 с, 64 МБ		x71
C	<a href="#">Zhenya moves from parents</a>	стандартный ввод/вывод 1 с, 64 МБ		x126
D	<a href="#">Zhenya moves from the dormitory</a>	стандартный ввод/вывод 1 с, 64 МБ		x270
E	<a href="#">Magic and Science</a>	стандартный ввод/вывод 1 с, 64 МБ		
F	<a href="#">Best of a bad lot</a>	стандартный ввод/вывод 1 с, 64 МБ		x52
G	<a href="#">The Debut Album</a>	стандартный ввод/вывод 1 с, 64 МБ		x247
H	<a href="#">Pair: normal and paranormal</a>	стандартный ввод/вывод 1 с, 64 МБ		x219
I	<a href="#">Traffic Jam in Flower Town</a>	стандартный ввод/вывод 1 с, 64 МБ		x347
J	<a href="#">Scarily interesting!</a>	стандартный ввод/вывод 1 с, 64 МБ		x181
K	<a href="#">Riding a Toad</a>	стандартный ввод/вывод 1 с, 64 МБ		x14
L	<a href="#">Donald is a postman</a>	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  $m$  ( $1 \leq n \leq 2 \cdot 10^5$ ,  $1 \leq m \leq 2 \cdot 10^{15}$ ) — 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  $a_i$ ,  $l_i$  ( $1 \leq a_i \leq 10^{10}$ ,  $0 \leq l_i \leq 1$ ), 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

```
1 #include <iostream>
2 #include <vector>
3 #include <algorithm>
4 #include <fstream>
5
6 using namespace std;
7
8 typedef struct {
9     LL num;
10    ULL cap;
11    short low;
12 } server;
13
14 bool cmp_low(const server &a, const server &b) {
15     return a.low > b.low;
16 }
17
18 bool cmp_cap(const server &a, const server &b) {
19     if (a.cap == b.cap) return a.low > b.low;
20     return a.cap > b.cap;
21 }
22
23 int main() {
24     LL n;
25     ULL m;
26     cin >> n >> m;
27     vector<server> s(n);
28     for (LL i = 0; i < n; i++) {
29         s[i].num = i + 1;
30         cin >> s[i].cap;
31         cin >> s[i].low;
32     }
33
34     sort(s.begin(), s.end(), cmp_cap);
35     LL ind = 0;
36     ULL curr_cap = 0;
37     LL count = 0;
38
39     while (curr_cap < m) {
40         curr_cap += s[ind].cap;
41         if (s[ind].low) count++;
42         ind++;
43     }
44
45     ULL rem = curr_cap - m;
46
47     if (rem == 0) {
48         cout << ind << " ";
49         cout << count << endl;
50         for (LL i = 0; i < ind; i++) {
51             cout << s[i].num << " ";
52         }
53         cout << endl;
54         return 0;
55     }
56
57     for (LL i = ind - 1; i >= 0; i--) {
58         if (!s[i].low && rem > 0) {
59             bool azaza = false;
```

```

60     for (LL j = ind; j < n; j++) {
61         if (s[j].low && (s[j].cap + rem >= s[i].cap)) {
62             rem -= s[i].cap - s[j].cap;
63             swap(s[i], s[j]);
64             count++;
65             azaza = true;
66             break;
67         }
68     if (!azaza) break;
69   }
70   else if (rem == 0) break;
71 }
72 }
73 cout << ind << " " << count << endl;
74 for (LL i = 0; i < ind; i++) {
75     cout << s[i].num << " ";
76 }
77 cout << endl;;
78 return 0;
79 }
80 }
```

## 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  $n$  ( $1 \leq n \leq 2000$ ) — the number of games in GameStore. The second line contains  $n$  integer numbers  $p_1, p_2, \dots, p_n$  ( $1 \leq p_i \leq 10^5$ ), 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

```
1 #include <iostream>
2 #include <vector>
3 #include <algorithm>
4 using namespace std;
5 bool compare(const int &a, const int &b)
6 {
7     return a<b;
8 }
9 int main(int argc, const char * argv[]) {
10    vector<int> p(2001);
11    int n;
12    cin >> n;
13    for(int i=0; i<n; ++i)
14        cin >> p[i];
15    sort(p.begin(), p.begin()+n);
16    int sum = 0;
17    int i=0;
18    while(i<n-1 && sum+p[i]<=p[n-1])
19        sum += p[i++];
20    cout << i+1;
21    return 0;
22 }
```

## 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

```
1 #include <iostream>
2 #include <vector>
3 #include <algorithm>
4 using namespace std;
5 typedef struct {
6     char ch;
7     int line;
8     int sym;
9 } var;
10 int main() {
11     int f,n;
12     cin >> n;
13     vector<stack<var>> all_alpha(26);
14     stack<var> curr;
15     char temp;
16     int symbol = 0;
17     var to_put;
18     temp = cin.get();
19     for (int i = 1; i <= n; i++) {
20         temp = cin.get();
21         symbol = 1;
22         while(temp != '\n') {
23             if(temp == '}') {
24                 to_put = curr.top();
25                 curr.pop();
26                 while (to_put.ch != '{') {
27                     all_alpha[to_put.ch - 97].pop();
28                     to_put = curr.top();
29                     curr.pop();
30                 }
31             }
32             else if(temp == '{') {
33                 to_put.ch = temp;
34                 to_put.sym = symbol;
35                 to_put.line = i;
36                 curr.push(to_put);
37             }
38             else if (temp != ' ') {
39                 to_put.sym = symbol;
40                 to_put.line = i;
41                 to_put.ch = temp;
42                 curr.push(to_put);
43                 if(all_alpha[temp - 97].size() != 0)
44                     cout << to_put.line << ":" << to_put.sym
45                     << ": warning: shadowed declaration of " << to_put.ch
46                     << ", the shadowed position is " << all_alpha[temp - 97].top().line
47                     << ":" << all_alpha[temp - 97].top().sym << endl;
48                     all_alpha[temp - 97].push(to_put);
49             }
50             temp = cin.get();
51             symbol++;
52         }
53     }
54     return 0;
55 }
```

# Результаты

Задачи				
№	Название			
A	<a href="#">Nasta Rabbara</a>	стандартный ввод/вывод 10 с, 512 МБ	 	 x41
B	<a href="#">Colored Blankets</a>	стандартный ввод/вывод 1 с, 512 МБ	 	 x167
C	<a href="#">Component Tree</a>	стандартный ввод/вывод 6 с, 512 МБ	 	 x148
D	<a href="#">Data Center</a>	стандартный ввод/вывод 2 с, 512 МБ	 	 x682
E	<a href="#">Election of a Mayor</a>	стандартный ввод/вывод 2 с, 512 МБ	 	 x452
F	<a href="#">Ilya Muromets</a>	стандартный ввод/вывод 1 с, 512 МБ	 	 x745
G	<a href="#">FacePalm Accounting</a>	стандартный ввод/вывод 1 с, 512 МБ	 	 x553
H	<a href="#">Minimal Agapov Code</a>	стандартный ввод/вывод 8 с, 512 МБ	 	 x18
I	<a href="#">Sale in GameStore</a>	стандартный ввод/вывод 2 с, 512 МБ	 	 x1312
J	<a href="#">Getting Ready for VIPC</a>	стандартный ввод/вывод 1,5 с, 512 МБ	 	 x28
K	<a href="#">Treeland</a>	стандартный ввод/вывод 2 с, 512 МБ	 	 x304
L	<a href="#">Useful Roads</a>	стандартный ввод/вывод 4 с, 512 МБ	 	 x10
M	<a href="#">Variable Shadowing</a>	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 · 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  $n$  ( $2 \leq n \leq 10^{14}$ ).

#### 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

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     unsigned long long a;
6     cin >> a;
7     while (!(a%2)) a /= 2;
8     if (a==1)
9         cout << "TAK";
10    else
11        cout << "NIE";
12    return 0;
13 }
```

## 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  $i = 1, 2, \dots, n$ , 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  $n$  ( $1 \leq n \leq 100\,000$ ). 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

```
1 #include <iostream>
2 #include <cmath>
3 using namespace std;
4 int main()
5 {
6     int n;
7     cin >> n;
8     if(n==1)
9     {
10         cout << "HM";
11         return 0;
12     }
13     cin.get();
14     char c;
15     long double hDrunked = 0, mDrunked = 0;
16     hDrunked = mDrunked = (1-pow(0.5, n))*0.5;
17     for(int i=0; i<n-1; ++i)
18     {
19         c=cin.get();
20         if(c=='H')
21             hDrunked += (1-pow(0.5, n-i-1))*0.5;
22         else
```

```

23         mDrunked += (1-pow(0.5, n-i-1))*0.5;
24     }
25     if(hDrunked>mDrunked)
26         cout << "H";
27     else if(hDrunked<mDrunked)
28         cout << "M";
29     return 0;
30 }
```

## Результаты

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

## 2.13 Codeforces Crypto Cup 1.0

### Задача В - :-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 its 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

```
1 #include <iostream>
2 #include <cmath>
3 #include <vector>
4 using namespace std;
5 int main(int argc, const char * argv[]) {
6     char str[100001];
7     long p, len;
8     while((str[len]=cin.get())!= '\n') ++len;
9     str[len] = '\0';
10    cin >> p;
11    vector< vector <char> > v(p);
12    long vSize = len/p, r = len%p;
13    for(long i=r; i<p; ++i) {
14        v[i].resize(vSize);
15    }
16    for(long i=0; i<r; ++i) {
17        v[i].resize(vSize+1);
18    }
19    for(long i=0, k=0; i<p; ++i) {
20        for(long j=0; j<v[i].size(); ++j, ++k) {
21            v[i][j] = str[k];
22        }
23    }
24    for(long i=0; i<len; ++i) {
25        cout.put(v[i%p][i/p]);
26    }
27    return 0;
28 }
```

## Задача С - 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

```
1 #include <iostream>
2 #include <cstdio>
3 using namespace std;
4 int main() {
5     char cur, prev;
6     cout.put(prev = cin.get());
7     while ((cur = cin.get()) != '\n') {
8         if ((int)(cur - prev) < 0) cout << (char)(cur - prev + '{');
9         else cout << (char)(cur - prev + 'a');
10        prev = cur;
11    }
12    return 0;
13 }
```

## 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  $a_1, \dots, a_{26}$  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 1 0 0 0 0 0 0 0 1
3
1
6
2
7
4
5
```

##### output

```
charzeh
```

## Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     vector<long> v(26);
5     long length = 0;
6     for (int i = 0; i < 26; i++) {
7         cin >> v[i];
8         length += v[i];
9     }
10    vector<char> answer(length);
11    long pos;
12    int nextSymb = -1;
13    for (int i = 0; i < 26; i++) {
14        if (v[i]) {
15            nextSymb = i;
16            break;
17        }
18    }
19    for (long i = 0; i < length; i++) {
20        cin >> pos;
21        if (!v[nextSymb]) {
22            for (int i = nextSymb; i < 26; i++) {
23                if (v[i]) {
24                    nextSymb = i;
25                    break;
26                }
27            }
28        }
29        answer[--pos] = (char)(nextSymb + 'a');
30        v[nextSymb]--;
31    }
32}
```

```
31     }
32     for (auto n : answer) cout << n;
33
34     return 0;
35 }
```

## 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

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int byte = 0;
5     int spow = 256;
6     int collected = 0;
7     char currentBit;
8     while ((currentBit = cin.get()) != '\n') {
9         cin.unget();
10        while (collected < 8) {
11            currentBit = cin.get();
12            byte += (currentBit - '0') * spow;
13            spow /= 2;
14            collected++;
15        }
16        cout << (char)(byte / 2);
17        byte = 0;
18        spow = 256;
19        collected = 0;
20    }
21    return 0;
22 }
```

## 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	wbpcfb
output	charzeh

## Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     char t;
5     while ((t = cin.get()) != '\n') {
6         switch (t) {
7             case 'a':
8                 cout << "n";
9                 break;
10            case 'b':
11                cout << "h";
12                break;
13            case 'c':
14                cout << "r";
15                break;
16            case 'd':
17                cout << "x";
18                break;
19            case 'e':
20                cout << "k";
21                break;
22            case 'f':
23                cout << "e";
24                break;
25            case 'g':
26                cout << "y";
27                break;
28            case 'h':
29                cout << "o";
30                break;
31            case 'i':
32                cout << "q";
```

```

33         break;
34     case 'j':
35         cout << "m";
36         break;
37     case 'k':
38         cout << "j";
39         break;
40     case 'l':
41         cout << "b";
42         break;
43     case 'm':
44         cout << "d";
45         break;
46     case 'n':
47         cout << "u";
48         break;
49     case 'o':
50         cout << "v";
51         break;
52     case 'p':
53         cout << "a";
54         break;
55     case 'q':
56         cout << "p";
57         break;
58     case 'r':
59         cout << "w";
60         break;
61     case 's':
62         cout << "g";
63         break;
64     case 't':
65         cout << "z";
66         break;
67     case 'u':
68         cout << "f";
69         break;
70     case 'v':
71         cout << "i";
72         break;
73     case 'w':
74         cout << "c";
75         break;
76     case 'x':
77         cout << "s";
78         break;
79     case 'y':
80         cout << "t";
81         break;
82     case 'z':
83         cout << "l";
84         break;
85     }
86 }
87 return 0;
88 }
```

## 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. ( $1 \leq n \leq 10^5$ ).

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

### Output

Print the original string.

#### Sample test(s)

input	
7	189 182 191 172 164 187 182
output	
charzeh	

Алгоритм  
?????????????????????????

## Исходный код

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     long n;
5     int curr, sum;
6     cin >> n;
7     for (long i = 0; i < n; i++) {
8         cin >> curr;
9         if (curr % 2) sum = 400;
10        else sum = 398;
11        cout << (char)(sum - curr - 112);
12    }
13    return 0;
14 }
```

## 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	

input	7 8 25 0 3 7 16 25
output	charzeh

Алгоритм  
?????????????????????????

## Исходный код

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

Алгоритм  
?????????????????????????

### Исходный код

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main()
6 {
7     char temp;
8     temp = cin.get();
9     while (temp != '\n' && temp != EOF) {
10         if (temp == 'z')
11             temp = 'a';
12         else temp++;
13         cout << temp;
14         temp = cin.get();
15     }
16     return 0;
17 }
```

## 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 its characters is a lower case English letter. ( $1 \leq |s| \leq 10^5$ )

### Output

Print the original string.

#### Sample test(s)

input	
punemru	
output	
charzeh	

## Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 #include <algorithm>
3 #include <deque>
4 #include <cstdio>
5 using namespace std;
6 void p(string s) {
7     cout << s << endl;
8 }
9 int gcd(int a, int b){
10     if (b == 0)
11         return a;
12     return gcd(b, a%b);
13 }
14 int main() {
15     char t;
16     while ((t = cin.get()) != '\n') {
17         t -= 13;
18         if (t < 'a') {
19             cout << (char)('z' - ('a' - t) + 1);
20         }
21         else {
22             cout << t;
23         }
24     }
25     return 0;
26 }
```

# Результаты

Задачи				
№	Название			
A	<a href="#">Bank</a>	стандартный ввод/вывод 2 с, 64 МБ	 	 x81
B	<a href="#">:-P</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x73
C	<a href="#">Pgkpxumgs</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x131
D	<a href="#">_13lk</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x27
E	<a href="#">Peace of AmericanWedding</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x62
F	<a href="#">2715</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x19
G	<a href="#">uehSlff</a>	стандартный ввод/вывод 2 с, 64 МБ	 	 x20
H	<a href="#">Peace of AmericaReunion</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x171
I	<a href="#">Peace of AmericanPie</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x279
J	<a href="#">Common</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x236
K	<a href="#">Crap</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x104
L	<a href="#">Key</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x22
M	<a href="#">oPlus</a>	стандартный ввод/вывод 2 с, 64 МБ	 	 x216
N	<a href="#">tirnaoeumPt</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x95
O	<a href="#">0x</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x75
P	<a href="#">Proooooooooooooofor</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x11
Q	<a href="#">Peace of bzjd</a>	стандартный ввод/вывод 1 с, 256 МБ	 	 x337
R	<a href="#">6227020800</a>	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 ( $1 \leq N \leq 20$ ,  $0 \leq M \leq 1.5 \cdot 10^{18}$ ).

#### 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

```
1 #include <iostream>
2 using namespace std;
3 #define ULL unsigned long long
4 int main(int argc, const char * argv[]) {
5     ULL n, m;
6     cin >> n >> m;
7     ULL res = 1;
8     for (int i=2; i<=n*2; i+=2)
9     {
10         res *= i;
11     }
12     if (res<m)
13         cout << "Nope";
14     else
15         cout << "Harshat Mata";
16     return 0;
17 }
```

## Задача 13 - Sum

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

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

### Input

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

### Output

Выведите одно число — значение требуемой суммы.

### Examples

input.txt	output.txt
3 4 101	120

# Алгоритм ?????????????????????

## Исходный код

```
1 #include <iostream>
2 #include <fstream>
3 #include <cmath>
4 #include <vector>
5 using namespace std;
6 #define ULL unsigned long long
7 #define LL long long
8 int main(int argc, const char * argv[]) {
9     ifstream in("input.txt");
10    ofstream out("output.txt");
11    ULL a, k, p;
12    cin >> a >> k >> p;
13    ULL sum = 0, prev = 1;
14    for(ULL i = 0; i<k; ++i)
15    {
16        prev = (prev*a)%p;
17        sum += prev;
18    }
19    cout << sum;
20    out.close();
21    in.close();
22    return 0;
23 }
```

## 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  $T$  ( $1 \leq T \leq 20$ ) — количество тестовых примеров.

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

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 ( $-100 \leq X, Y \leq 100$ ,  $1 \leq R \leq 10$ ).

## Output

Для каждого тестового примера выведите одну строку с именем 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

```
1 #include <iostream>
2 #include <fstream>
3 #include <cmath>
4 #include <vector>
5 using namespace std;
6 #define ULL unsigned long long
7 #define LL long long
8 #define eps 0.0001
9 struct player
10 {
11     char name[256];
12     LL x, y, r;
13 };
14 int main(int argc, const char * argv[]) {
15     ifstream in("input.txt");
16     ofstream out("output.txt");
17     ULL T;
18     in >> T;
19     double pi81 = M_PI/81, pi2 = M_PI*2;
20     for(ULL TT = 0; TT<T; ++TT)
21     {
22         ULL n;
23         in >> n;
24         vector<player> pl(n);
25         for(int i=0; i<n; ++i)
26         {
27             in >> pl[i].name >> pl[i].x >> pl[i].y >> pl[i].r;
28         }
29         ULL maxInd = -1, max = 0, forTie = -1;
30         for(int i=0; i<n; ++i)
31         {
32             ULL count = 0;
33             LL rr = pl[i].r*pl[i].r;
34             for(int j=0; j<n; ++j)
35             {
36                 for(double pi = 0; pi<=pi2; pi += pi81)
37                 {
38                     double x = (pl[j].r-eps)*cos(pi);
39                     double y = (pl[j].r-eps)*sin(pi);
40                     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)
41                     {
42                         ++count;
43                         break;
44                     }
45                 }
46                 if(count > max)
47                 {
48                     max = count;
49                     maxInd = i;
50                     forTie = -1;
51                 }
52             else if(count == max)
53             {
```

```

55         max = count;
56         forTie = maxInd;
57         maxInd = i;
58     }
59 }
60 if(maxInd == -1 || forTie != -1)
61     out << "TIE" << '\n';
62 else
63     out << pl[maxInd].name << '\n';
64 }
65 out.close();
66 in.close();
67 return 0;
68 }
```

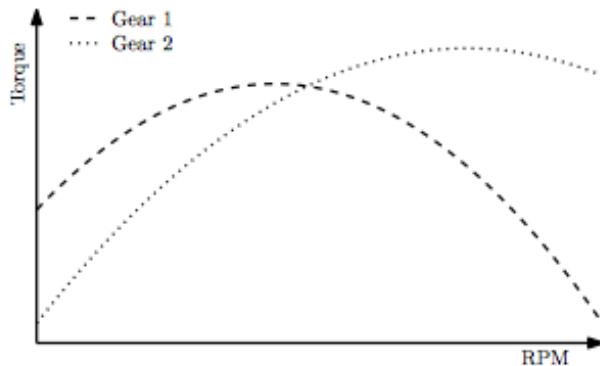
## Результаты

Турнир, этапы, карты																	
26.	MAI #11: Makarov, Rik, Yakimenko	-	-10 4:49	-	-	-	-	-	-	+	+4 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	Output
3	1
1	2
1 4 2	2
2	
3 126 1400	
2 152 208	
2	
3 127 1400	
2 154 208	

Алгоритм  
?????????????????????????

## Исходный код

```
1 #include <iostream>
2 #include <algorithm>
3
4 #include <fstream>
5
6 using namespace std;
7
8 int main() {
9     int k;
10    cin >> k;
11    int n, a, b, c;
12    double max_T = -1000000;
13    int max_T_num;
14    double temp;
15
16    for(int i = 0; i < k; i++) {
17        cin >> n;
18        for(int j = 1; j <= n; j++) {
19            cin >> a >> b >> c;
20            temp = (b * b) / (4 * a) + c;
21            if(temp > max_T) {
22                max_T = temp;
23                max_T_num = j;
24            }
25        }
26        cout << max_T_num << endl;
27        max_T = -1000000;
28    }
29    int q;
30    cin >> q;
31    return 0;
32 }
```

## 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.

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.



source: [xkcd.com/246](http://xkcd.com/246)

### 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  $w$  ( $3 \leq h, w \leq 100$ ): 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	###
FRLFFFLLBRFFFRRFFRFRFBRFLBRFRRLFLFFR	...#
	##.#
	####
	7 5
	####
	...##
	##.##
	#...#
	##.##
	##.##
	####
	7 7
	#####
	#...#.#
	#.#...#
	#.#.###
	..###.#
	#....#
	######

Алгоритм  
?????????????????????????

## Исходный код

```
1 #include <iostream>
2 using namespace std;
3 int main(int argc, const char * argv[]) {
4     long T;
5     cin >> T;
6     cout << T << '\n';
7     char map[201][201];
8     cin.get();
9     while(T--)
10    {
11        for(long i=0; i<201; ++i)
12            for(long j=0; j<201; ++j)
13                map[i][j] = '#';
14        char c;
15        long x, y, minX, minY, maxX, maxY;
16        maxX = maxY = minX = minY = x = y = 100;
17        int dir = 0;
18        while((c=cin.get())!= '\n')
19        {
20            switch(dir)
21            {
22                case 0:
23                    if(c=='R')
24                        dir = 3;
25                    else if(c=='L')
26                        dir = 1;
27                    else if(c=='B')
28                        dir = 2;
29                    break;
30                case 1:
31                    if(c=='R')
32                        dir = 0;
33                    else if(c=='L')
34                        dir = 2;
35                    else if(c=='B')
36                        dir = 3;
37                    break;
38                case 2:
39                    if(c=='R')
40                        dir = 1;
41                    else if(c=='L')
42                        dir = 3;
43                    else if(c=='B')
44                        dir = 0;
45                    break;
46                case 3:
47                    if(c=='R')
48                        dir = 2;
49                    else if(c=='L')
50                        dir = 0;
51                    else if(c=='B')
52                        dir = 1;
53                    break;
54            }
55            switch(dir)
56            {
57                case 0:
58                    ++x;
```

```

59             if (x>maxX)
60                 maxX = x;
61                 break;
62             case 1:
63                 --y;
64                 if (y<minY)
65                     minY = y;
66                     break;
67             case 2:
68                 --x;
69                 if (x<minX)
70                     minX = x;
71                     break;
72             case 3:
73                 ++y;
74                 if (y>maxY)
75                     maxY = y;
76                     break;
77         }
78         map[x][y] = '.';
79     }
80     cout << maxY-minY+3 << ' ' << maxX-minX+2 << '\n';
81     for (long i = minY-1; i<=maxY+1; ++i)
82     {
83         for (long j = minX; j<=maxX+1; ++j)
84             cout.put(map[j][i]);
85             cout.put('\n');
86     }
87 }
88 return 0;
89 }
```

## Результаты

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

## 2.16 Codeforces Training S02E09

### Задача G - Graveyard

Programming contests became so popular in the year 2397 that the governor of New Earch — the largest human-inhabited planet of the galaxy — opened a special Alley of Contestant Memories (ACM) at the local graveyard. The ACM encircles a green park, and holds the holographic statues of famous contestants placed equidistantly along the park perimeter. The alley has to be renewed from time to time when a new group of memorials arrives.

When new memorials are added, the exact place for each can be selected arbitrarily along the ACM, but the equidistant disposition must be maintained by moving some of the old statues along the alley.

Surprisingly, humans are still quite superstitious in 24th century: the graveyard keepers believe the holograms are holding dead people souls, and thus always try to renew the ACM with minimal possible movements of existing statues (besides, the holographic equipment is very heavy). Statues are moved along the park perimeter. Your work is to find a renewal plan which minimizes the sum of travel distances of all statues. Installation of a new hologram adds no distance penalty, so choose the places for newcomers wisely!

#### Input

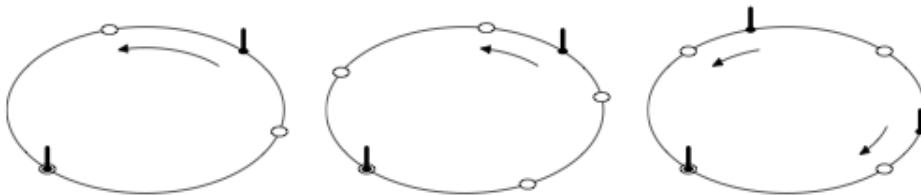
Input file contains two integer numbers:  $n$  — the number of holographic statues initially located at the ACM, and  $m$  — the number of statues to be added ( $2 \leq n \leq 1000, 1 \leq m \leq 1000$ ). The length of the alley along the park perimeter is exactly 10 000 feet.

#### Output

Write a single real number to the output file — the minimal sum of travel distances of all statues (in feet). The answer must be precise to at least 4 digits after decimal point.

#### Sample input and output

graveyard.in	graveyard.out
2 1	1666.6667
2 3	1000.0
3 1	1666.6667
10 10	0.0



Pictures show the first three examples. Marked circles denote original statues, empty circles denote new equidistant places, arrows denote movement plans for existing statues.

#### Алгоритм

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

#### Исходный код

```
1 #include <iostream>
2 #include <vector>
3 #include <string>
4 #include <algorithm>
```

```

5 #include <queue>
6 #include <climits>
7 #include <cctype>
8 #include <cmath>
9 #include <fstream>
10 #include <iomanip>
11 #define ll long long
12 #define ull unsigned long long
13 using namespace std;
14 int main()
15 {
16     ifstream in("graveyard.in");
17     ofstream out("graveyard.out");
18     ll n, m;
19     in >> n >> m;
20     ll nm = n+m;
21     vector<double> a(n);
22     vector<double> b(nm);
23     double s = 0;
24     for(ll i=0; i<n; ++i)
25     {
26         a[i] = s;
27         s += 10000.0/n;
28     }
29     s = 0;
30     for(ll i=0; i<nm; ++i)
31     {
32         b[i] = s;
33         s += 10000.0/nm;
34     }
35     vector<bool> u(nm, 0);
36     vector<double> r(n);
37     double res = 0;
38     for(ll i=0; i<n; ++i)
39     {
40         double min = 200000.0;
41         for(ll j=0; j<nm; ++j)
42         {
43             if (!u[j] && fabs(a[i]-b[j])<min)
44             {
45                 u[j] = 1;
46                 min = fabs(a[i]-b[j]);
47             }
48         }
49         res += min;
50     }
51     out << fixed << setprecision(4) << res;
52     in.close();
53     out.close();
54     return 0;
55 }

```

## Задача J - Java vs C++

Apologists of Java and C++ can argue for hours proving each other that their programming language is the best one. Java people will tell that their programs are clearer and less prone to errors, while C++ people will laugh at their inability to instantiate an array of generics or tell them that their programs are slow and have long source code.

Another issue that Java and C++ people could never agree on is identifier naming. In Java a multiword identifier is constructed in the following manner: the first word is written starting from the small letter, and the following ones are written starting from the capital letter, no separators are used. All other letters are small. Examples of a Java identifier are `javaIdentifier`, `longAndMnemonicIdentifier`, `name`, `nEERC`.

Unlike them, C++ people use only small letters in their identifiers. To separate words they use underscore character `'_'`. Examples of C++ identifiers are `c_identifier`, `long_and_mnemonic_identifier`, `name` (you see that when there is just one word Java and C++ people agree), `n_e_e_r_c`.

You are writing a translator that is intended to translate C++ programs to Java and vice versa. Of course, identifiers in the translated program must be formatted due to its language rules — otherwise people will never like your translator.

The first thing you would like to write is an identifier translation routine. Given an identifier, it would detect whether it is Java identifier or C++ identifier and translate it to another dialect. If it is neither, then your routine should report an error. Translation must preserve the order of words and must only change the case of letters and/or add/remove underscores.

### Input

The input file consists of one line that contains an identifier. It consists of letters of the English alphabet and underscores. Its length does not exceed 100.

### Output

If the input identifier is Java identifier, output its C++ version. If it is C++ identifier, output its Java version. If it is none, output “Error!” instead.

### Sample input and output

java.c.in	java.c.out
<code>long_and_mnemonic_identifier</code>	<code>longAndMnemonicIdentifier</code>
<code>anotherExample</code>	<code>another_example</code>
<code>i</code>	<code>i</code>
<code>bad_Style</code>	Error!

### Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 #define ll long long
3 #define ull unsigned long long
4 #define ERROR {out << "Error!"; return 0;}
5 using namespace std;
6 int main()
7 {
8     ifstream in("java_c.in");
9     ofstream out("java_c.out");
10    char c;
11    bool java = 0, cpp = 0, us = 0;
12    char str[1000];
13    ll n=0;
14    while (!in.eof() && (c=in.get())!= '\n')
15    {
16        if (in.eof())
17            break;
18        if (c=='_')
19        {
20            if (us || !n)
21                ERROR
22            cpp = 1;
23            us = 1;
24        }
25        else if (isupper(c))
26        {
27            if (!n)
28                ERROR
29            java = 1;
30            str[n++] = '_';
31            str[n++] = tolower(c);
32        }
33        else if (us)
34            str[n++] = toupper(c);
35        else
36            str[n++] = c;
37        if (c!='_')
38            us = 0;
39        if (java && cpp)
40            ERROR
41    }
42    str[n] = '\0';
43    if (n&&!us)
44        out << str;
45    else
46        out << "Error!";
47    in.close();
48    out.close();
49    return 0;
50 }
```

## Задача K - Kickdown

Apologists of Java and C++ can argue for hours proving each other that their programming language is the best one. Java people will tell that their programs are clearer and less prone to errors, while C++ people will laugh at their inability to instantiate an array of generics or tell them that their programs are slow and have long source code.

Another issue that Java and C++ people could never agree on is identifier naming. In Java a multiword identifier is constructed in the following manner: the first word is written starting from the small letter, and the following ones are written starting from the capital letter, no separators are used. All other letters are small. Examples of a Java identifier are `javaIdentifier`, `longAndMnemonicIdentifier`, `name`, `nEERC`.

Unlike them, C++ people use only small letters in their identifiers. To separate words they use underscore character `'_'`. Examples of C++ identifiers are `c_identifier`, `long_and_mnemonic_identifier`, `name` (you see that when there is just one word Java and C++ people agree), `n_e_e_r_c`.

You are writing a translator that is intended to translate C++ programs to Java and vice versa. Of course, identifiers in the translated program must be formatted due to its language rules — otherwise people will never like your translator.

The first thing you would like to write is an identifier translation routine. Given an identifier, it would detect whether it is Java identifier or C++ identifier and translate it to another dialect. If it is neither, then your routine should report an error. Translation must preserve the order of words and must only change the case of letters and/or add/remove underscores.

### Input

The input file consists of one line that contains an identifier. It consists of letters of the English alphabet and underscores. Its length does not exceed 100.

### Output

If the input identifier is Java identifier, output its C++ version. If it is C++ identifier, output its Java version. If it is none, output "Error!" instead.

### Sample input and output

java.c.in	java.c.out
<code>long_and_mnemonic_identifier</code>	<code>longAndMnemonicIdentifier</code>
<code>anotherExample</code>	<code>another_example</code>
<code>i</code>	<code>i</code>
<code>bad_Style</code>	Error!

### Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 #include <fstream>
3
4 int main() {
5     ifstream in("kickdown.in");
6     ofstream out("kickdown.out");
7     string master, driven;
8     in >> master >> driven;
9     if (master.length() < driven.length()) swap(master, driven);
10
11    int pos1 = 0;
12    while (pos1 < master.length() && !checkGears1(master, driven, pos1)) pos1++;
13
14    int pos2 = 0;
15    while (pos2 < master.length() && !checkGears2(master, driven, pos2)) pos2++;
16
17    int d1 = (int)(driven.length() + pos1 - master.length());
18    int diff1 = (d1 >= 0) ? d1 : 0;
19    int diff2 = pos2;
20
21    if (diff1 < diff2) {
22        out << max(master.length(), driven.length() + pos1) << endl;
23    }
24    else {
25        out << master.length() + pos2 << endl;
26    }
27
28    in.close();
29    out.close();
30
31    return 0;
32 }
```

## Результаты

Задачи		Название		
№				
A	<a href="#">ASCII Art</a>	ascii.in / ascii.out 3 c, 256 МБ	  	x38
B	<a href="#">Billing Tables</a>	billing.in / billing.out 1 c, 256 МБ	  	x17
C	<a href="#">Cellular Automaton</a>	cell.in / cell.out 2 c, 256 МБ	  	x113
D	<a href="#">Driving Directions</a>	driving.in / driving.out 1 c, 256 МБ	 	
E	<a href="#">Exchange</a>	exchange.in / exchange.out 1 c, 256 МБ	  	x53
F	<a href="#">Fool's Game</a>	fool.in / fool.out 1 c, 256 МБ	  	x4
G	<a href="#">Graveyard</a>	graveyard.in / graveyard.out 1 c, 256 МБ	  	x232
H	<a href="#">Hard Life</a>	hard.in / hard.out 1 c, 256 МБ	  	x10
I	<a href="#">Interconnect</a>	interconnect.in / interconnect.out 1 c, 256 МБ	  	x67
J	<a href="#">Java vs C++</a>	java_c.in / java_c.out 1 c, 256 МБ	  	x294
K	<a href="#">Kickdown</a>	kickdown.in / kickdown.out 1 c, 256 МБ	  	x256

## 2.17 Codeforces Олимпиада школьников Нижегородской обл.

### Задача А - Выравнивание вещественных чисел

Вася придумал новый алгоритм сортировки вещественных чисел, который, по его мнению, будет работать быстрее, даже чем QuickSort! Он очень хочет запатентовать его, но для этого сначала необходимо представить в патентное бюро работающую версию алгоритма. К сожалению, Вася не силен в языках программирования, поэтому попросил друзей реализовать различные куски алгоритма. Вам досталась часть программы, отвечающая за выравнивание чисел.

Задана последовательность  $A_i$  положительных вещественных чисел, не превосходящих  $10^{1000}$ . Количество цифр после точки в представлении каждого числа не превосходит 1000. Требуется выписать все числа в исходном порядке в один столбец таким образом, чтобы все точки в их десятичной записи находились друг под другом. Для этого перед некоторыми числами необходимо прописать один или несколько символов #. Если решений несколько, то выведите то, в котором количество добавленных символов минимально.

#### Входные данные

В первой строке входных данных находится целое число  $N$  – количество чисел в последовательности  $A_i$  ( $1 \leq N \leq 1000$ ). Последующие  $N$  строк содержат по одному вещественному числу последовательности  $A_i$  в десятичной записи. Числа не содержат ведущих нулей. В каждом числе присутствует десятичная точка.

#### Выходные данные

Выходные данные должны содержать ровно  $N$  строк, по одному числу из последовательности  $A_i$  в строке. Числа должны быть выровнены по точке десятичной записи с помощью #.

#### Примеры тестов

входные данные
3
3.1415926
12345.6789
2.71
выходные данные
####3.1415926
12345.6789
###2.71

## Алгоритм

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

### Исходный код

```
1 #include <iostream>
2 #include <cmath>
3 #define ll long long
4 #define ull unsigned long long
5 using namespace std;
6 int main()
7 {
8     ll n;
9     cin >> n;
10    char num[1000][2010];
11    int logs[1000];
12    ll maxLog = 1;
13    cin.get();
14    for(ll i=0; i<n; ++i)
15    {
16        ll j=0;
17        logs[i] = 0;
18        bool pf = 0;
19        while((num[i][j]=cin.get())!= '\n')
20        {
21            if(num[i][j] == '.')
22                pf = 1;
```

```

23         if (!pf)
24             ++logs[i];
25         ++j;
26     }
27     num[i][j++] = '\0';
28     if (logs[i] > maxLog)
29         maxLog = logs[i];
30 }
31 for (ll i=0; i<n; ++i)
32 {
33     for (ll j=0; j<maxLog-logs[i]; ++j)
34         cout.put('#');
35     cout << num[i] << '\n';
36 }
37 return 0;
38 }
```

## Задача F - Фоторамка

Вася выиграл в школьной викторине приз — рамку для фотографий, причем не простую, а такую, в которую можно вставить сразу несколько фотографий. У Васиной рамки есть четыре места под фотографии, расположенные в ряд.

В век цифрового фото у Васи не так много бумажных фотографий, которые он мог бы вставить в рамку — всего  $N$  штук. Мальчик планирует менять их состав каждый день, и ему интересно, как долго он сможет это делать. Для этого ему надо подсчитать, сколько есть способов заполнить рамку имеющимися у него фотографиями. Отметим, что Вася считает различными и способы, отличающиеся только порядком фотографий в рамке.

### Входные данные

На единственной строке записано одно целое число  $N$  ( $1 \leq N \leq 20$ ) — количество фотографий, которые есть у Васи.

### Выходные данные

Выведите единственное число — количество различных способов заполнить Васиными фотографиями рамку.

#### Примеры тестов

входные данные
4
выходные данные
24

Алгоритм  
?????????????????????

## Исходный код

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     long long m[]={0, 0, 0, 24, 120, 360, 840, 1680, 3024, 5040, 7920, 11880, 17160,
5     24024, 32760, 43680, 57120, 73440, 93024, 116280};
6     int n;
7     cin >> n;
8     cout << m[n-1];
9 }
```

## Задача I - Изи

Вася вернулся с международной олимпиады школьников по программированию (IOI) и привез с собой  $N$  разноцветных камней в качестве сувениров. Вася совсем не жадный мальчик, поэтому решил поделиться камнями со своими друзьями. Каждому другу Вася отдал ровно один камень. Оказалось, что у самого Васи остался тоже только один камень. Определите, сколько же у Васи друзей?

### Входные данные

В первой строке входного файла находится число  $N$  ( $1 \leq N \leq 100$ ) — количество камней, привезенных Васей.

### Выходные данные

Выведите единственное число — количество друзей Васи.

### Примеры тестов

входные данные
2
выходные данные
1

## Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 #include <vector>
3 #include <string>
4 #include <algorithm>
5 #include <queue>
6 #include <climits>
7 #include <cctype>
8 #include <fstream>
9 #define ll long long
10 #define ull unsigned long long
11 using namespace std;
12 int main()
13 {
14     ll n;
15     cin >> n;
16     cout << n-1;
17     return 0;
18 }
```

## Результаты

Задачи			
№	Название		
A	<a href="#">Выравнивание вещественных чисел</a>	стандартный ввод/вывод 2 с, 256 МБ	
B	<a href="#">Игра в 9</a>	стандартный ввод/вывод 2 с, 256 МБ	
C	<a href="#">Преобразование числа</a>	стандартный ввод/вывод 2 с, 256 МБ	
D	<a href="#">Марракеш</a>	стандартный ввод/вывод 2 с, 256 МБ	
E	<a href="#">Мухи на плоскости</a>	стандартный ввод/вывод 2 с, 256 МБ	
F	<a href="#">Фоторамка</a>	стандартный ввод/вывод 2 с, 256 МБ	
G	<a href="#">Сны о скобках</a>	стандартный ввод/вывод 2 с, 256 МБ	
H	<a href="#">Спуск полос</a>	стандартный ввод/вывод 2 с, 256 МБ	
I	<a href="#">Изи</a>	стандартный ввод/вывод 2 с, 256 МБ	
J	<a href="#">Затмение</a>	стандартный ввод/вывод 2 с, 256 МБ	
K	<a href="#">Парковка</a>	стандартный ввод/вывод 2 с, 256 МБ	
L	<a href="#">Пушка Гаусса</a>	стандартный ввод/вывод 2 с, 256 МБ	

## 2.18 OpenCup GrandPrix of Central Europe

### Задача А - Адвокат

Байтазар — известный адвокат, совладелец компании «Байтазар и товарищи». Также он участвует в заседаниях различных комитетов и комиссий. Неудивительно, что он всё время занят.

Каждый день Байтазара приглашают на множество различных заседаний; в какой-то момент он понял, что ему сложно понять, сможет ли он участвовать во всех заседаниях.

Поэтому Байтазар нанял секретаря, который должен решать эти вопросы. Байтазар решил, что каждый день он будет участвовать только в двух заседаниях, но с начала и до конца. На оставшиеся заседания Байтазар отправит своих помощников.

Считается, что заседания не перекрываются, если одно из них начинается строго после того, как завершится другое. Помогите секретарю Байтазара составить соответствующее расписание.

#### Input

Первая строка входного файла содержит два целых числа  $n$  и  $m$  ( $2 \leq n \leq 500\,000$ ,  $1 \leq m \leq 20$ ) — количество совещаний, на которые приглашён Байтазар и количество дней в его расписании. describing the number of meetings in Byteasar's schedule and the number of days included in it.

Каждая из последующих  $n$  строк описывает одну встречу. Описание встречи состоит из трёх целых чисел  $a_i, b_i, d_i$  ( $1 \leq a_i < b_i \leq 80\,000\,000$ ,  $1 \leq d_i \leq m$ ), обозначающих, что в день  $d_i$  Байтазар приглашён на встречу, которая начинается на  $a_i$  миллисекунде соответствующего дня и завершается на  $b_i$  миллисекунде.

#### Output

Выполните  $m$  строк.  $i$ -я из этих строк должна содержать информацию, сможет ли Байтазар в  $i$ -й день присутствовать на двух заседаниях. В случае, если это невозможно, выведите слово “NIE”. Иначе выведите слова “ТАК” и номер двух заседаний, на которых может присутствовать Байтазар в соответствующий день. Заседания пронумерованы в том порядке, в котором они описаны во входном файле, нумерация заседаний начинается с единицы. Второе заседание должно начаться как минимум на миллисекунду позже того, как закончилось первое.

Если существует несколько решений, выведите любое из них.

#### Examples

standard input	standard output
6 3	TAK 1 6
3 5 1	NIE
2 4 2	NIE
1 8 1	
6 7 3	
3 5 2	
7 12 1	

# Алгоритм ?????????????????????

## Исходный код

```
1 #include <iostream>
2 #include <fstream>
3 #include <vector>
4
5 using namespace std;
6
7 int main()
8 {
9     ll n, m;
10    ll a, b, d;
11    cin >> n >> m;
12    vector< pair<ll, ll> > maxA(m, make_pair(-1, -1)), minB(m, make_pair(-1, -1));
13    for (ll i=0; i<n; ++i) {
14        cin >> a >> b >> d;
15        --d;
16        if (maxA[d].second == -1 || maxA[d].second < a) {
17            maxA[d].second = a;
18            maxA[d].first = i;
19        }
20        if (minB[d].second == -1 || minB[d].second > b) {
21            minB[d].second = b;
22            minB[d].first = i;
23        }
24    }
25    for (ll i=0; i<m; ++i) {
26        if (minB[i].second < maxA[i].second) {
27            cout << "TAK " << minB[i].first+1 << ' ' << maxA[i].first+1 << '\n';
28        }
29        else {
30            cout << "NIE\n";
31        }
32    }
33    return 0;
34 }
```

## Результаты

170.	MAI #11: Makarov, Rik, Yakimenko	+	-	-1 3:08	-8 2:19	-	-	-	-	-	-	-4 3:57	1	63	0%	9.69	0.08
------	----------------------------------	---	---	------------	------------	---	---	---	---	---	---	------------	---	----	----	------	------

## 2.19 Codeforces Training S02E10

### Задача A - Abnormal Coins

If you have participated or solved any of the four previous AUT local contests, you may have noticed that first problem of each contest were all about abnormal new students of Amirkabir UT. This time we want to talk about coins in our problem-set but to respect traditions, let's start with **abnormal type of coins**.

Everybody knows what coins are, “*A flat, typically round piece of metal with an official stamp, used as money.*”. But as wikipedia “*Not all coins are round. The Australian 50 cent coin, for example, has twelve flat sides. Some coins have wavy edges, e.g. the \$2 and 20-cent coins of Hong Kong and the 10 cent coins of Bahamas. Some are square-shaped, such as the 15 cent coin of the Bahamas. During the 1970s, Swazi coins were minted in several shapes, including squares, polygons, and wavy edged circles with 8 and 12 waves.*”

Steve is a great coin collector but he has no polygon shaped coin in his collection. Recently he found another coin lover, Johnny, Who has a collection of polygon coins and fortunately agreed to deal with Steve. The deal is simple,  $e$  coins for each poly-coin with  $e$  edges. Steve can offer  $n$  coin(s) and wants to earn as many poly-coin types as possible to extend his collection. Two poly-coins are different if they differ in number of edges.

#### Input

Input contains a single integer  $n$ . ( $1 \leq n \leq 10^8$ )

#### Output

Output a single integer indicating the maximum number of distinct poly-coins Steve can earn.

#### Examples

stdin	stdout
2	0
3	1
1000000000	14139

### Алгоритм

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

### Исходный код

```
1 #include <iostream>
2 #include <vector>
3 #include <algorithm>
4 #include <iomanip>
5 #define ll long long
6 #define ull unsigned long long
7 using namespace std;
8 int main() {
9     LL n;
10    cin >> n;
11    LL count = 0;
12    LL sum = 0;
13
14    for (LL i = 3; ; i++) {
15        sum += i;
16        if (sum > n) break;
```

```
17         count++;
18     }
19     cout << count << endl;
20     return 0;
21 }
```

## Задача В - Fake Coins

Cristiano, as a leader of his team, works in the Security Lab of Central Bank of Real Mars (SLCBRM). They want to improve security of coins that Central Bank makes. They want to use string codes that will be carved on coins to determine whether the coin is fake or original. Cris's team suggested him an amazing idea for managing security of coins.

They have a base string called  $B$  and a sequence of numbers called  $S$ . They select some characters from the base string to generate the security codes. If the  $i$ -th element of sequence is  $S_i$  Then the  $i$ -th element of generated code is the  $S_i$ -th character of  $B$  i.e.  $(B[S_i])$ . As you can clearly see the length of the generated security string code is equal to the number of elements in the sequence.

Their sequences are very similar to fibonacci numbers but their first and second elements are not always 1! They select two different numbers as first and second element of sequence (first number is always less than the second number). Then the third number of sequence is the sum of first and second numbers, the forth number is the sum of third and second number and so on. This process stops when a new element is greater than the length of the sequence. All elements of the sequence must be less than or equal to length of the base string.

Cris read the algorithm of generating string codes and realized that it can be some problems with same string codes. He wants to know how many different string codes can be generated using his team's algorithm. Because Criastiano is busy with scoring for Real Mars Football team he asked you to solve his problem!

### Input

First and only line of input contains the base string  $B$  ( $3 \leq \text{len}(B) \leq 1000$ ). Base string is composed of digits, lowercase and uppercase Latin letter.

### Output

Print one line containing the number of different security string codes that can be generated using Criastiano's team algorithm.

### Examples

stdin	stdout
abba	5
abBA	6
FAKE0123456789original	217

### Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 #include <map>
3 #include <vector>
4 #define ll long long
5 #define ull unsigned long long
6 #define ERROR {out << "Error!"; return 0;}
7 using namespace std;
8 int main() {
9     map<string, int> strings;
10    string base, current = "";
11    vector<int> spos;
12    cin >> base;
13    int baseSize = (int)base.size();
14    for (int i = 1; i < baseSize; i++) {
15        for (int j = i + 1; j <= baseSize; j++) {
16            current += base[i - 1];
17            current += base[j - 1];
18            int cPos = i + j;
19            int pPos = j, temp = 0;
20            while (cPos <= baseSize) {
21                current += base[cPos - 1];
22                temp = pPos;
23                pPos = cPos;
24                cPos += temp;
25            }
26            strings[current]++;
27            current.clear();
28        }
29    }
30    cout << strings.size() << endl;
31
32
33    return 0;
34 }
```

## Задача G - Coin Game

Alice and Bob are very smart guys and they like to play all kinds of games in their spare time. The most amazing thing is that they always find the best strategy, and that's why they feel bored again and again. They just invented a new game, as they usually did.

They are playing with coins. They have a row of \$1 and \$2 coins. They want to change this row so that all \$1's are grouped together and all \$2's are grouped together. They are just allowed to swap two neighbor coins.

Using the best strategy what is the minimum number of swaps required to do this task?

### Input

Each test case is a line with a string composed of 1 and 2.

( $1 \leq \text{length of the input string} \leq 25000$ )

### Output

Print the minimum number of swaps required to do this task.

### Examples

stdin	stdout
221212	3

### Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 #include <vector>
3 #include <map>
4 #include <cmath>
5 #include <algorithm>
6 #include <iomanip>
7 #include <list>
8 #include <iterator>
9 using namespace std;
10 #define ll long long
11 #define ull unsigned long long
12
13 int main()
14 {
15     char a[25001], b[25001], c;
16     int n = 0, first1 = -1, last1 = -1;
17     while((c=cin.get())!='\n' && !cin.eof())
18     {
19         if(first1 == -1 && c == '1')
20             first1 = n;
21         if(c == '1')
22             last1 = n;
23         if(cin.eof())
24             break;
25         a[n] = b[n] = c;
26         ++n;
27     }
28     a[n] = b[n] = '\0';
29     ll aCount = 0, bCount = 0;
30     for(ll i=0; i<n; ++i)
31     {
32         if(a[i] == '1')
33             continue;
34         for(ll j=i+1; j<n; ++j)
35         {
36             if(a[j] == '1')
37             {
38                 swap(a[i], a[j]);
39                 aCount += j-i;
40                 break;
41             }
42         }
43     }
44     for(ll i=n-1; i>=0; --i)
45     {
46         if(b[i] == '1')
47             continue;
48         for(ll j=i-1; j>=0; --j)
49         {
50             if(b[j] == '1')
51             {
52                 swap(b[i], b[j]);
53                 bCount += i-j;
54                 break;
55             }
56         }
57     }
58     if(aCount < bCount)
```

```

59     cout << aCount;
60
61     cout << bCount;
62
63 }
```

## Результаты

Задачи				
№	Название			
A	<a href="#">Abnormal Coins</a>	стандартный ввод/вывод 1 с, 256 МБ		x284
B	<a href="#">Fake Coins</a>	стандартный ввод/вывод 1 с, 256 МБ		x194
C	<a href="#">Coin Graph</a>	стандартный ввод/вывод 1 с, 256 МБ		x117
D	<a href="#">Coin Table</a>	стандартный ввод/вывод 3 с, 256 МБ		x112
E	<a href="#">Volleyball</a>	стандартный ввод/вывод 2 с, 256 МБ		x10
F	<a href="#">Huge Table</a>	стандартный ввод/вывод 2 с, 256 МБ		x7
G	<a href="#">Coin Game</a>	стандартный ввод/вывод 1 с, 256 МБ		x243
H	<a href="#">Dreams Were Important Too!</a>	стандартный ввод/вывод 3 с, 256 МБ		x42
I	<a href="#">Coin Robbery</a>	стандартный ввод/вывод 2 с, 256 МБ		x70
J	<a href="#">Bimetallic coins</a>	стандартный ввод/вывод 2 с, 256 МБ		x46
K	<a href="#">Censorship!</a>	стандартный ввод/вывод 2 с, 256 МБ		x10

## 2.20 OpenCup GrandPrix of Europe

### Задача E - Express As The Sum

Задано целое число  $N$ . Требуется представить его в виде суммы двух или более последовательных целых положительных чисел. Например,  $10 = 1 + 2 + 3 + 4$

$$24 = 7 + 8 + 9$$

Если решений несколько, выведите то из них, которое состоит из наименьшего количества слагаемых.

#### Input

Первая строка входа содержит одно целое число  $T$  — количество тестовых примеров.

Каждый тестовый пример состоит из одной строки, содержащей целое число  $N$  ( $1 \leq N \leq 10^9$ ).

#### Output

Для каждого тестового примера выведите одну строку, содержащую требуемое разложение в виде

$$N = a + (a+1) + \dots + b,$$

форматированное в соответствии с примером из условия. Если решения не существует, выведите текст “IMPOSSIBLE”.

#### Example

standard input	standard output
3	IMPOSSIBLE
8	
10	10 = 1 + 2 + 3 + 4
24	24 = 7 + 8 + 9

# Алгоритм

## ?????????????????????????

### Исходный код

```
1 #include <iostream>
2 using namespace std;
3 #define ll long long
4 #define ull unsigned long long
5 void answer(ll n, ll l, ll r) {
6     cout << n << " = " << l;
7     for(ll i=l+1; i<=r; ++i) {
8         cout << " + " << i;
9     }
10    cout.put('\n');
11}
12 int main()
13 {
14     ll T;
15     cin >> T;
16     while (T--) {
17         ll n;
18         cin >> n;
19         ll t = n;
20         bool f = 0;
21         while(t!=0) {
22             if (t>1 && t&1) {
23                 f = 1;
24                 break;
25             }
26             t >>= 1;
27         }
28         if (!f) {
29             cout << "IMPOSSIBLE\n";
30             continue;
31         }
32         f = 0;
33         for (ll i=2; !f && i<100500; ++i) {
34             ll sum = 0;
35             ll r = n/i;
36             ll l = r-i+1;
37             if(l<1) {
38                 l = 1;
39                 r = l+i-1;
40             }
41             for(ll j=l; j<=r; ++j) {
42                 sum += j;
43             }
44             if(sum == n) {
45                 answer(n, l, r);
46                 break;
47             }
48             for(ll j=r+1; j<r+i; ++j) {
49                 sum += j-1;
50                 if(sum == n) {
51                     answer(n, l+1, j);
52                     f = 1;
53                     break;
54                 }
55             }
56         }
57     }
58 }
```

```

56         }
57     }
58 }
59 return 0;
60 }
```

## Задача F - Factory

Важный узел секретной машины состоит из  $n$  наношестерёнок, занумерованных последовательными целыми числами от 1 до  $n$ . Наношестерёнки отличаются от обычных шестерёнок тем, что размер их зубьев настолько мал, что шестерёнки можно считать обычными кругами. Каждая наношестерёнка крутится вокруг своего центра.

Никакие две наношестерёнки не перекрываются (не имеют общих внутренних точек), однако шестерёнки могут касаться друг друга. Если две шестерёнки касаются и одна из них вращается, другая тоже вращается, так как соответствующие зубья зацеплены.

К шестерёнке 1 приложена сила, заставляющая её вращаться со скоростью 1 оборот в минуту по часовой стрелке. Вычислите скорость вращения остальных шестерёнок. Гарантируется, что при вращении механизм не блокируется.

### Input

Первая строка входа содержит одно целое число  $T$  — количество тестовых примеров.

Каждый тестовый пример начинается со строки, задающей количество наношестерёнок  $n$  ( $1 \leq n \leq 1000$ ). Каждая из последующих  $n$  строк содержит три целых числа  $x$ ,  $y$  и  $r$  ( $-10\,000 \leq x, y \leq 10\,000$ ;  $1 \leq r \leq 10\,000$ ), где  $(x, y)$  — координаты центра шестерёнки и  $r$  — её радиус.

### Output

Для каждого тестового примера выведите  $n$  строк, описывающих вращение шестерёнок. Для каждой шестерёнки, выведите или “ $p/q$  clockwise”, или “ $p/q$  counterclockwise”, где несократимая дробь  $p/q$  задаёт количество оборотов наношестерёнки в минуту. Если  $q = 1$ , выведите  $p$  как целое число. Если шестерёнка не вращается, выведите “not moving”.

### Example

standard input	standard output
1	1 clockwise
5	3/2 counterclockwise
0 0 6	2 counterclockwise
6 8 4	3/2 clockwise
-9 0 3	not moving
6 16 4	
0 -11 4	

# Алгоритм

## ?????????????????????????

### Исходный код

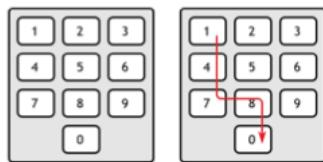
```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4 #define ll long long
5 #define ull unsigned long long
6 #define eps 0.00001
7 ll min(ll a, ll b){return (a<b?a:b);}
8 ll max(ll a, ll b){return (a>b?a:b);}
9
10 struct Gear {
11     int x, y, r;
12 };
13
14 struct GearSpeed {
15     ll n, d;
16     bool dir;
17 };
18 int main()
19 {
20     ll T;
21     cin >> T;
22     while (T--) {
23         ll n;
24         cin >> n;
25
26         vector< list<int> > m(n);
27         vector<bool> u(n, 0);
28         vector<Gear> gears(n);
29         for (ll i=0; i<n; ++i) {
30             cin >> gears[i].x >> gears[i].y >> gears[i].r;
31         }
32         for (int i=0; i<n; ++i) {
33             for (int j=i+1; j<n; ++j) {
34                 double x = gears[i].x-gears[j].x;
35                 double y = gears[i].y-gears[j].y;
36                 if (sqrt(x*x+y*y)<=gears[i].r+gears[j].r) {
37                     m[i].insert(m[i].begin(), j);
38                     m[j].insert(m[j].begin(), i);
39                 }
40             }
41         }
42         vector<GearSpeed> speed(n);
43         speed[0].n = 1;
44         speed[0].d = 1;
45         speed[0].dir = 0;
46         queue<int> q;
47         q.push(0);
48         u[0] = 1;
49         while (!q.empty()) {
50             int a = q.front();
51             q.pop();
52             list<int>::iterator it = m[a].begin();
53             for (; it!=m[a].end(); ++it) {
54                 int b = *it;
55                 if (u[b])
```

```

56         continue;
57     u[b] = 1;
58     speed[b].n = speed[a].n * gears[a].r;
59     speed[b].d = speed[a].d * gears[b].r;
60     speed[b].dir = !speed[a].dir;
61     ll u = speed[b].n;
62     ll v = speed[b].d;
63     ll temp;
64     while (v != 0) {
65         temp = u % v;
66         u = v;
67         v = temp;
68     }
69     speed[b].n /= u;
70     speed[b].d /= u;
71     q.push(b);
72 }
73 for (ll i=0; i<n; ++i) {
74     if (!u[i])
75         cout << "not moving\n";
76     else {
77         cout << speed[i].n;
78         if (speed[i].d != 1) {
79             cout << '/' << speed[i].d;
80         }
81         cout.put(',');
82         if (speed[i].dir) {
83             cout << "counterclockwise\n";
84         }
85         else {
86             cout << "clockwise\n";
87         }
88     }
89 }
90 }
91 return 0;
92 }
```

## Задача K - Keyboard Troubles

Вы только что проснулись и хотите включить микроволновку на  $k$  секунд. Вы должны набрать число  $k$  на клавиатуре, имеющей следующий вид:



Так как по сути, вы не совсем проснулись, вы можете набрать число только в случае, если ваши руки будут двигаться вниз или вправо. Вы не можете двигаться влево или вверх, но вы можете нажимать одну и ту же кнопку многократно.

Выведите число, которое Вы можете набрать и которое наиболее близко к  $k$ . Если решений несколько, выведите любое из двух.

## Input

Первая строка содержит одно целое число  $T$  — количество тестовых примеров.

Каждый тестовый пример содержит одно целое число  $k$  ( $1 \leq k \leq 200$ ) — требуемое количество минут.

## Output

Для каждого тестового примера в отдельной строке выведите ближайшее к  $k$  число, которое может быть введено в соответствии с условием задачи.

## Examples

standard input	standard output
3	180
180	80
83	133
132	

# Алгоритм

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

## Исходный код

```

54     for (int i = 0; i < t; i++) {
55         int n;
56         cin >> n;
57         if (z[n] == 1) {
58             cout << n << endl;
59         }
60         else {
61             int y = n;
62             int x = n;
63             while (z[x] == 0) {
64                 x++;
65             }
66             while (z[y] == 0) {
67                 y--;
68             }
69             if (n - y > x - n) {
70                 cout << x << endl;
71             }
72             else {
73                 cout << y << endl;
74             }
75         }
76     }
77
78     return 0;
79 }
```

## Задача K - Allo

Задан телефонный номер, составленный из цифр от 0 до 9. Найдите все цифры, которые не используются в этом номере.

### Input

Первая строка входа содержит одно целое число  $T$  — количество тестовых примеров ( $1 \leq T \leq 150$ ). Каждая из последующих  $T$  строк задаёт один тестовый пример — непустую строку, состоящую из цифр от 0 до 9. Длина заданной строки не превосходит 30.

### Output

Для каждого тестового примера выведите в возрастающем порядке все числа, которые не используются в соответствующей строке. Если все цифры используются, то выведите текст “allo”.

### Examples

standard input	standard output
2 2468 0123456789	013579 allo

# Алгоритм

## ?????????????????????

### Исходный код

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4 #define ll long long
5 #define ull unsigned long long
6 ll min(ll a, ll b){return (a<b?a:b);}
7 ll max(ll a, ll b){return (a>b?a:b);}
8 int main()
9 {
10     ll n;
11     vector<stack<pair<ll, int>>> vars(26);
12     int scope = 0;
13     cin >> n;
14     char str[20];
15     for (int i=0; i<n; ++i) {
16         cin >> str;
17         if(str[0] == '{') {
18             ++scope;
19         }
20         else if(str[0] == '}') {
21             for (int i=0; i<26; ++i) {
22                 if(!vars[i].empty() && vars[i].top().second == scope) {
23                     vars[i].pop();
24                 }
25             }
26             --scope;
27         }
28         else if(str[1] == '=') {
29             int c1 = str[0]-'a';
30             ll a;
31             if(isalpha(str[2])) {
32                 int c2 = str[2]-'a';
33                 a = vars[c2].top().first;
34                 int c1Scope = vars[c1].top().second;
35                 vars[c1].pop();
36                 vars[c1].push(make_pair(a, c1Scope));
37             }
38             else {
39                 a = 0;
40                 for (int i=2; str[i]!='\0'; ++i)
41                     a = a*10+str[i]-'0';
42             }
43             int c1Scope = vars[c1].top().second;
44             vars[c1].pop();
45             vars[c1].push(make_pair(a, c1Scope));
46         }
47         else if(!strcmp(str, "int")) {
48             cin >> str;
49             int c;
50             c = str[0]-'a';
51             vars[c].push(make_pair(-1, scope));
52         }
53         else if(!strcmp(str, "print")) {
54             cin >> str;
55             int c = str[0]-'a';
```

```

56         cout << vars[c].top() . first << '\n';
57     }
58 }
59 return 0;
60 }
```

## Задача O - Game

Алиса и Боб играют в следующую игру. Первоначально задано число 1. Игроки делают ходы по очереди, в свою очередь хода каждый игрок должен умножить текущее число на одно из целых чисел от 2 до 9 включительно. Выигрывает тот, кто первый получает число, не меньшее заданного  $N$ . Алиса всегда начинает первой.

Для заданного  $N$  найдите, кто выиграет при оптимальной игре обеих сторон.

### Input

Первая строка входа содержит одно целое число  $T$  — количество тестовых примеров ( $1 \leq T \leq 2500$ ). Каждая из последующих  $T$  строк содержит одно целое число  $N$  ( $2 \leq N \leq 10^9$ ).

### Output

Для каждого тестового примера выведите в отдельной строке 1 в случае победы Алисы и 2 в случае победы Боба.

### Example

standard input	standard output
4	1
9	2
10	2
1149729	1
999999999	

# Алгоритм ?????????????????????

## Исходный код

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4 #define ll long long
5 #define ull unsigned long long
6 ll min(ll a, ll b){return (a<b?a:b);}
7 ll max(ll a, ll b){return (a>b?a:b);}
8 int main()
9 {
10     ll T;
11     cin >> T;
12     cin.get();
13     while (T--) {
14         vector<bool> nums(10, 0);
15         char c;
16         while ((c=cin.get())!= '\n' && !cin.eof())
17         {
18             if (cin.eof())
19                 break;
20             nums[c-'0'] = 1;
21         }
22         bool f = 0;
23         for (ll i=0; i<10; ++i)
24         {
25             if (!nums[i])
26             {
27                 f = 1;
28                 cout << i;
29             }
30         }
31         if (!f)
32             cout << "allo";
33         cout << '\n';
34     }
35 }
36 return 0;
37 }
```

## Результаты

19.	MAI #11: Makarov, Rik, Yakimenko	-	-	+2 2:18	+4 4:53	-	-	+3 1:44	-	+ 0:36	+ 0:03	-2 2:52	5	755	64%	0.33
-----	----------------------------------	---	---	------------	------------	---	---	------------	---	-----------	-----------	------------	---	-----	-----	------

## 2.21 OpenCup GrandPrix of Peterhof

### Задача Н - Некратчайший путь

*В этой задаче нужно пройти из одного угла клетчатого поля  $4 \times 4$  в другой по любому простому пути, не являющемуся кратчайшим.*

На плоскости нарисовано квадратное клетчатое поле из  $4 \times 4$  клеток. Каждая клетка поля либо свободна, либо целиком занята стеной.

Робот Аврелий проходит тест на измерение уровня интеллекта, решая различные задачи на этом поле. Он уже успешно справился с поиском произвольного и кратчайшего пути из одной клетки в другую.

В очередном задании Аврелию следует пройти из левого верхнего угла поля в правый нижний по свободным клеткам. За один шаг из любой клетки можно переместиться в любую другую клетку, имеющую с ней общую сторону. При этом путь должен быть простым, то есть в одной и той же клетке нельзя оказываться дважды. Кроме того, длина пути должна быть строго больше, чем длина кратчайшего пути между этими углами на заданном поле. Длиной пути считается количество шагов в нём.

Напишите программу, которая позволит роботу Аврелию справиться с заданием.

#### Формат входных данных

На входе задан один или несколько тестовых случаев. Каждый тестовый случай задаётся на четырёх строках. Каждая из этих строк описывает один ряд поля и содержит ровно четыре символа, задающих клетки этого ряда. Пустая клетка обозначается символом «.» (точка, ASCII-код 46), а стена — символом «#» (решётка, ASCII-код 35). Соседние тестовые случаи отделяются друг от друга строкой, состоящей из четырёх символов «-» (минус, ASCII-код 45).

Гарантируется, что левая верхняя и правая нижняя клетки поля пусты. Также гарантируется, что тестовые случаи на входе не повторяются в пределах одного теста.

#### Формат выходных данных

В ответ на каждый тестовый случай выведите строку, определяющую путь. Стока должна состоять из символов, соответствующих командам перемещения в соседнюю клетку поля, в порядке их применения: «D» для движения вниз, «U» для движения вверх, «L» для движения влево и «R» для движения вправо. Если возможных путей несколько, выведите любой из них. Если же пути с нужными свойствами не существует, выведите вместо него число «-1».

#### Пример

nsp.in	nsp.out
.#..	DDRURURDDD
....	-1
..#.	-1
.#..	
----	
..##	
...#	
#...	
.#..	
----	
...#	
...#.	
.#..	
#...	

# Алгоритм

## ?????????????????????????

### Исходный код

```
1 #include <iostream>
2 #include <vector>
3 #include <map>
4 using namespace std;
5 #define ll long long
6 #define ull unsigned long long
7 #define eps 0.00001
8 ll min(ll a, ll b){return (a<b?a:b);}
9 ll max(ll a, ll b){return (a>b?a:b);}
10
11 vector<vector<char>> paths;
12 vector<char> path;
13 size_t __min;
14
15 void foo(char map[4][4], int x, int y) {
16     if (x == 3 && y == 3) {
17         if (path.size() < __min) {
18             __min = path.size();
19         }
20         paths.push_back(path);
21     }
22     if (x<3 && map[y][x+1] == '.') {
23         path.push_back('R');
24         map[y][x] = 'X';
25         foo(map, x+1, y);
26         map[y][x] = '.';
27         path.pop_back();
28     }
29     if (x>0 && map[y][x-1] == '.') {
30         path.push_back('L');
31         map[y][x] = 'X';
32         foo(map, x-1, y);
33         map[y][x] = '.';
34         path.pop_back();
35     }
36     map[y][x] = '.';
37     if (y<3 && map[y+1][x] == '.') {
38         path.push_back('D');
39         map[y][x] = 'X';
40         foo(map, x, y+1);
41         map[y][x] = '.';
42         path.pop_back();
43     }
44     map[y][x] = '.';
45     if (y>0 && map[y-1][x] == '.') {
46         path.push_back('U');
47         map[y][x] = 'X';
48         foo(map, x, y-1);
49         map[y][x] = '.';
50         path.pop_back();
51     }
52 }
53
54 int main()
55 {
```

```

56     char map[4][4];
57     bool e = 1;
58     while (e) {
59         for (int i=0; i<4; ++i) {
60             for (int j=0; j<4; ++j) {
61                 map[i][j] = cin.get();
62             }
63             cin.get();
64         }
65         __min = 100500;
66         foo(map, 0, 0);
67         bool f = 0;
68         for (int i=0; i<paths.size(); ++i) {
69             if (paths[i].size() != __min) {
70                 for (int j=0; j<paths[i].size(); ++j) {
71                     cout.put(paths[i][j]);
72                 }
73                 f = 1;
74                 break;
75             }
76         }
77         if (!f) {
78             cout << -1;
79         }
80         cout.put('\n');
81         for (int i=0; i<5; ++i) {
82             cin.get();
83             if (cin.eof()) {
84                 e = 0;
85                 break;
86             }
87         }
88         paths.clear();
89         path.clear();
90     }
91     return 0;
92 }
```

## Результаты

31.	MAI #11: Makarov, Rik, Yakimenko	-	-	-	-	-	-	-	+	-	-	-	1	204	0%	0.27
-----	----------------------------------	---	---	---	---	---	---	---	---	---	---	---	---	-----	----	------

## 2.22 OpenCup GrandPrix of Japan

### Задача K - Beads

Задано ожерелье из  $N$  жемчужин, пронумерованных последовательными целыми числами от 1 до  $N$  по часовой стрелке. Каждая жемчужина помечена заглавной латинской буквой. Таким образом, если прочитать эти буквы последовательно по часовой стрелке, получится строка длины  $N$ . Так как ожерелье замкнуто, то в зависимости от того, с какой жемчужины начинать чтение, можно получить различные строки.

Требуется найти среди этих строк лексикографически наименьшую.

#### Input

Первая строка входа содержит целое число  $N$  ( $1 \leq N \leq 2 \cdot 10^6$ ) — длину ожерелья. Во второй строке задано само ожерелье. Ожерелье задаётся строкой, содержащей  $N$  заглавных латинских букв;  $k$ -я буква — это буква, написанная на  $k$ -й жемчужине.

#### Output

Выведите номер жемчужины, начиная с которой, можно прочитать по часовой стрелке лексикографически наименьшую строку длиной  $N$ . Если ответов несколько, выведите наименьший.

#### Example

standard input	standard output
6 CABCAB	2

# Алгоритм

## ?????????????????????

### Исходный код

```
1 #include <iostream>
2
3 using namespace std;
4
5 long min_cyclic_shift (string s) {
6     s += s;
7     long n = (long) s.length();
8     long i = 0, ans = 0;
9     while (i < n/2) {
10         ans = i;
11         long j = i + 1, k = i;
12         while (j < n && s[k] <= s[j]) {
13             if (s[k] < s[j])
14                 k = i;
15             else
16                 ++k;
17             ++j;
18         }
19         while (i <= k) i += j - k;
20     }
21     return ans;
22 }
23
24 int main(int argc, const char * argv[]) {
25     long n;
26     string s;
27     cin >> n >> s;
28     long answer = min_cyclic_shift(s);
29     cout << answer + 1 << endl;
30 }
31 }
```

### Задача L - The Maximum Sum

Заданы  $N$  целых положительных чисел. Среди них выбираются два, сумма которых является наибольшей, не превосходящей заданного числа  $M$ . Требуется найти значение соответствующей суммы.

#### Input

Первая строка входа содержит два целых положительных числа  $N$  ( $3 \leq N \leq 100$ ) и  $M$  ( $1 \leq M \leq 100$ ). Вторая строка содержит  $N$  целых положительных чисел.

#### Output

Выведите одно число — требуемую сумму.

#### Examples

standard input	standard output
5 8 5 3 4 6 5	8
4 116 31 52 73 84	115

# Алгоритм ?????????????????????

## Исходный код

```
1 #include <iomanip>
2 #include <limits>
3 #include <iostream>
4 #include <algorithm>
5
6 int main() {
7     int n, M;
8     cin >> n >> M;
9
10    vector <int> num;
11    int temp;
12    for (int i = 0; i < n; i++) {
13        cin >> temp;
14        num.push_back(temp);
15    }
16
17    int max_sum = 0;
18    for (int i = 0; i < n; i++) {
19        for (int j = i + 1; j < n; j++) {
20            temp = num[i] + num[j];
21            if (temp <= M && temp > max_sum)
22                max_sum = temp;
23        }
24
25    cout << max_sum << endl;
26    cin >> n;
27    return 0;
28 }
```

## Задача L - The Maximum Sum

Заданы  $N$  целых положительных чисел. Среди них выбираются два, сумма которых является наибольшей, не превосходящей заданного числа  $M$ . Требуется найти значение соответствующей суммы.

### Input

Первая строка входа содержит два целых положительных числа  $N$  ( $3 \leq N \leq 100$ ) и  $M$  ( $1 \leq M \leq 100$ ). Вторая строка содержит  $N$  целых положительных чисел.

### Output

Выведите одно число — требуемую сумму.

### Examples

standard input	standard output
5 8 5 3 4 6 5	8
4 116 31 52 73 84	115

# Алгоритм ?????????????????????

## Исходный код

```
1 #include <iomanip>
2 #include <limits>
3 #include <iostream>
4 #include <algorithm>
5
6 int main() {
7     int n, M;
8     cin >> n >> M;
9
10    vector <int> num;
11    int temp;
12    for (int i = 0; i < n; i++) {
13        cin >> temp;
14        num.push_back(temp);
15    }
16
17    int max_sum = 0;
18    for (int i = 0; i < n; i++) {
19        for (int j = i + 1; j < n; j++) {
20            temp = num[i] + num[j];
21            if (temp <= M && temp > max_sum)
22                max_sum = temp;
23        }
24
25    cout << max_sum << endl;
26    cin >> n;
27    return 0;
28 }
```

## Задача М - Spellcheck

Вася написал спеллчекер, который выдаёт ошибки в следующих случаях:

- “и” или “иr” вместо “you” или “your”.
- “would of”, “should of” вместо “would have”, “should have”.
- “lol”. Более того, программа выдаёт ошибку на любое слово, в котором можно прочитать “lol” как подслово (если слово содержит “lol” несколько раз, как слова “olololo”, всё равно выдаётся одна ошибка).

Напишите программу, которая считывает предложения одно за другим и для каждого предложения определяет, сколько ошибок найдёт новый спеллчекер.

## Input

Первая строка входа содержит целое число  $T$  ( $1 \leq T \leq 50$ ) — количество предложений в тесте. Каждая из последующих  $T$  строк содержит одно предложение — одно или несколько слов, разделённых пробелами. Слова состоят из строчных латинских букв. Слова разделены ровно одним пробелом, пробелы в начале и в конце предложения отсутствуют. Общая длина каждого предложения (вместе с пробелами) не превосходит 100 символов.

## Output

Для каждого предложения в отдельной строке выведите, сколько ошибок найдёт в нём новый спеллчекер.

## Examples

standard input	standard output
5	2
r u haz trololo	0
my name is vasya	1
i got the lollipop	3
u should of lollollo	0
i should off line	

# Алгоритм

## ?????????????????????????

### Исходный код

```
1 #include <iostream>
2 #include <vector>
3
4 using namespace std;
5
6 int main()
7 {
8     long T;
9     cin >> T;
10    char aStr[100], bStr[100];
11    cin.get();
12    while (T--) {
13        char c;
14        long a = 0, aN = 0, bN = 0, res = 0;
15        while (c = cin.get()) {
16            if (c == ',' || c == '\n') {
17                if (aN&1) {
18                    if (!strcmp("u", bStr) || !strcmp("ur", bStr) || strstr(bStr, "lol") != NULL) {
19                        ++res;
20                    }
21                    else if ((!strcmp("would", aStr) || !strcmp("should", aStr)) && !strcmp("of", bStr)) {
22                        ++res;
23                    }
24                    aN = 0;
25                }
26                else {
27                    if (!strcmp("u", aStr) || !strcmp("ur", aStr) || strstr(aStr, "lol") != NULL) {
28                        ++res;
29                    }
30                    else if (aN && (!strcmp("would", bStr) || !strcmp("should", bStr)) && !strcmp("of", aStr)) {
31                        ++res;
32                    }
33                    bN = 0;
34                }
35                if (c == '\n') {
36                    break;
37                }
38                ++a;
39                continue;
40            }
41            if (aN&1) {
42                bStr[bN++] = c;
43                bStr[bN] = '\0';
44            }
45            else {
46                aStr[aN++] = c;
47                aStr[aN] = '\0';
48            }
49        }
50        cout << res << '\n';
51    }
```

```
52     return 0;  
53 }
```

## Задача N - Bluetooth

Вы пытаетесь отправить сообщение другу через Bluetooth. Смартфон может соединяться по Bluetooth с любым смартфоном, если расстояние между ними не превышает  $D$ .

Определите, можно ли отправить сообщение «по цепочке» (напрямую или через некоторое количество промежуточных смартфонов).

### Input

Первая строка входа содержит два целых числа  $N$  ( $1 \leq N \leq 10$ ) и  $D$  ( $1 \leq D \leq 10$ ), где  $N$  — общее количество смартфонов с Bluetooth в помещении, а  $D$  — максимальное расстояние, на котором два смартфона соединяются по Bluetooth.

Каждая из следующих  $N$  строк содержит по два целых числа — координаты  $X$  и  $Y$  очередного смартфона. При этом первая строка задаёт координаты Вашего смартфона, последняя — координаты смартфона Вашего друга ( $0 \leq X, Y \leq 100$ ).

### Output

Выведите ‘y’, если сообщение передать можно и ‘n’ в противном случае.

### Example

standard input	standard output
4 7 1 4 6 2 9 7 14 4	y
5 6 7 1 5 5 1 6 8 7 20 15	n

# Алгоритм ?????????????????????

## Исходный код

```
1 #include <iomanip>
2 #include <limits>
3 #include <iostream>
4 #include <algorithm>
5
6
7 double dist( const pair<int , int> &a, const pair<int , int> &b) {
8     double x, y;
9     x = (a.first - b.first);
10    y = (a.second - b.second);
11    x *= x;
12    y *= y;
13    return sqrt(x + y);
14 }
15
16 int main() {
17
18     more_speed
19     int n, d;
20     cin >> n >> d;
21     vector<pair<int , int> > points(n);
22     for (int i = 0; i < n; i++) {
23         cin >> points[i].first >> points[i].second;
24     }
25
26     vector<vector<int> > g(n, vector<int>());
27
28     for (int i = 0; i < n; i++) {
29         for (int j = 0; j < n; j++) {
30             if (i != j) {
31                 double dis = dist(points[i], points[j]);
32                 if (dis <= d) {
33                     g[i].push_back(j);
34                 }
35             }
36         }
37     }
38
39     int s = 0;
40
41     queue<int> q;
42     q.push(s);
43     vector<bool> used (n);
44     used[s] = true;
45     while (!q.empty()) {
46         int v = q.front();
47         q.pop();
48         for (size_t i = 0; i < g[v].size(); ++i) {
49             int to = g[v][i];
50             if (!used[to]) {
51                 if (to == n - 1) {
52                     cout << "y" << endl;
53                     return 0;
54                 }
55             }
56             used[to] = true;
```

```

56             q.push ( to );
57         }
58     }
59 }
60
61     cout << "n" << endl;
62     return 0;
63 }
```

## Результаты

16.	MAI #11: Makarov, Rik, Yakimenko	-2 4:56	-	-	-	-	-	+4 3:50	+	+	+1 1:07	4	457	55%	0.20
-----	----------------------------------	------------	---	---	---	---	---	------------	---	---	------------	---	-----	-----	------

## 2.23 OpenCup Northern GrandPrix

### Задача K - Kill The PSU

Вам поручена разработка системы контроля стабильности работы устройств и их драйверов. Система должна выдавать подсказки пользователю в зависимости от типа устройства, с которым произошёл сбой.

Устройства с точки зрения этой системы делятся на следующие три категории (каждое устройство принадлежит только к одной из них).

- Категория 1: Внешняя периферия. Проблемы этих устройств заключаются в том, что время от времени они переходят в режим экономии питания (засыпают). В случае, если это устройство встретилось в логе, Вы должны рекомендовать пользователю разбудить (wake) его.
- Категория 2: Внутренние устройства с нестабильными драйверами. При появлении проблем с такими устройствами требуется загрузить (load) драйвер, если он не был загружен, или выгрузить в противном случае. Так как драйверы являются экспериментальными, то в момент начальной загрузки система их не загружает.
- Категория 3: Основные устройства. Проблемы этих устройств обычно вызваны нехваткой питания. Соответственно, реакцией на появление этого устройства в логе должна быть выдача соответствующего сообщения, использование резервной мощности блока питания (если она ещё не использована), а в крайнем случае, когда питания устройству не хватает совсем — выдача рекомендации о замене блока питания и остановка системы.

Считается, что каждому основному устройству идёт 100 ватт от блока питания в момент старта системы. Появление основного устройства в логе обозначает, что на него стало поступать на 10 ватт меньше. Первоначально у блока питания есть резерв в 20 ватт; так что первые два сбоя должны быть компенсированы; далее же подаваемая на устройства мощность начинает уменьшаться. В случае, когда на устройство подаётся не более 10 ватт, Ваша программа должна выдать рекомендацию о покупке нового блока питания и остановить систему; все последующие сообщения в логе (если таковые есть) обрабатываться уже не должны.

По заданному списку имён устройств в системе, категориям, к которым они принадлежат и логу сбоев выведите требуемые рекомендации для пользователя.

#### Input

Первая строка входа содержит одно целое число  $T$  — количество тестовых примеров ( $0 \leq T \leq 100$ ). Далее заданы сами тестовые примеры.

Первая строка каждого тестового примера состоит из четырёх целых чисел  $A, B, C, D$ , разделённых пробелами. Первые три числа задают количество устройств в первой, второй и третьей категориях соответственно, последнее задаёт количество событий в логе.

Далее следуют  $A$  строк, содержащие имена устройств из первой категории,  $B$  строк, содержащих имена устройств из второй категории,  $C$  строк, содержащих имена устройств из третьей категории, и  $D$  строк, задающих лог сбоев; каждая запись в логе представляет собой имя устройства, с которым возникли проблемы ( $0 \leq A, B, C, D \leq 100$ , имена устройств состоят из строчных латинских букв и пробелов, начинаются и заканчиваются строчной латинской буквой и имеют длину не менее 2 и не более 22).

#### Output

Для каждого тестового примера выведите рекомендацию системы по каждому из сбоев в порядке их появления в логе.

Для устройств первой категории выведите “`wake` (имя устройства)”. Для устройств второй категории выведите “`load` (имя устройства)”, если драйвер не был загружен (в случае первого после загрузки системы появления этого устройства в логе или после выгрузки) и “`unload` (имя устройства)” в противном случае.

Для устройств третьей категории выведите “`power fail on` (имя устройства)”, если на устройство подаётся более 10 ватт. В противном случае выведите “`buy the new PSU`” и завершите обработку данного тестово-

го примера вне зависимости от того, остались ли события в логе. В случае неясности с форматов вывода следуйте формату из тестовых примеров.

## Examples

# Алгоритм ?????????????????????

## Исходный код

```
1 #include <iomanip>
2 #include <iostream>
3 #include <algorithm>
4 #include <vector>
5 #include <fstream>
6
7 #define LL long long
8 #define ULL unsigned long long
9 using namespace std;
10
11 ifstream in("killthebsu.in");
12 ofstream out("killthebsu.out");
13
14 class item {
15 public:
16     string name;
17     virtual void message() = 0;
18 };
19
20 class cat1 : public item {
21 public:
22     void message() {
23         out << "wake " << name << endl;
24     }
25     cat1(string n) {
26         name = n;
27     }
28 };
29
30 class cat2 : public item {
31 public:
32     bool load;
33     void message() {
34         if (load) {
35             out << "unload ";
36             load = false;
37         }
38         else {
39             out << "load ";
40             load = true;
41         }
42         out << name << endl;
43     }
44     cat2(string n) {
45         name = n;
46         load = false;
47     }
48 };
49
50 int reserve = 20;
51 bool nextIter = false;
52
53 class cat3 : public item {
54 public:
55     int power;
```

```

56     void message() {
57         if (reserve) {
58             reserve -= 10;
59             out << "power fail on " << name << endl;
60             return ;
61         }
62         power -= 10;
63         if (power > 10) {
64             out << "power fail on " << name << endl;
65         }
66         else if (power <= 10) {
67             out << "buy the new PSU" << endl;
68             nextIter = true;
69         }
70     }
71     cat3(string n) {
72         power = 100;
73         name = n;
74     }
75 };
76
77 int main() {
78     int t;
79     in >> t;
80
81     for (int k = 0; k < t; k++) {
82         reserve = 20;
83         map<string , item *> system;
84         int a, b, c, d;
85         in >> a >> b >> c >> d;
86         string name;
87         in.get();
88
89         for (int i = 0; i < a; i++) {
90             getline(in , name);
91             item *newItem = new cat1(name);
92             system[name] = newItem;
93             //cout << i << " " << name << endl;
94         }
95
96         for (int i = 0; i < b; i++) {
97             getline(in , name);
98             item *newItem = new cat2(name);
99             system[name] = newItem;
100            //cout << i << " " << name << endl;
101        }
102
103        for (int i = 0; i < c; i++) {
104            getline(in , name);
105            item *newItem = new cat3(name);
106            system[name] = newItem;
107            //cout << i << " " << name << endl;
108        }
109
110        for (int i = 0; i < d; i++) {
111            getline(in , name);
112            if (!nextIter) {
113                map<string , item *>::iterator current = system.find(name);
114                if (current != system.end())
115                    current->second->message();

```

```

116         }
117     }
118     nextIter = false;
119 }
120
121     in.close();
122     out.close();
123
124     return 0;
125 }
```

## Задача М - Мозаика

Мальчик Костя очень любит не только собирать мозаики, но и с удовольствием придумывает различные игры во время складывания их обратно в коробку.

Мозаика имеет форму прямоугольника  $M \times N$ , каждая клетка которого содержит одну плитку. На каждой плитке написано целое число от 1 до  $M \cdot N$  (на всех плитках числа различны). В собранном виде мозаика имеет следующий вид:

1	2	3	...	$N - 1$	$N$
$N + 1$	$N + 2$	$N + 3$	...	$2 \cdot N - 1$	$2 \cdot N$
...	...	...	...	...	...
$(M - 1) \cdot N + 1$	$(M - 1) \cdot N + 2$	$(M - 1) \cdot N + 3$	...	$M \cdot N - 1$	$M \cdot N$

Костя хочет сыграть в игру, в которой требуется убрать как можно больше плиток мозаики по определенным правилам. За один ход можно убрать любую плитку, у которой есть 4 соседних плитки (плитки считаются соседними, если они находятся в соседних клетках в одной строке или в одном столбце).

Напишите программу, которая определит максимально возможное количество плиток, которые можно убрать по описанным выше правилам. А также найдите какой-либо корректный максимальный набор плиток.

### Input

В единственной строке входных данных записаны два целых числа  $M$  и  $N$  ( $1 \leq M, N \leq 100$ ) — высота и ширина мозаики.

### Output

В первой строке выведите максимальное количество плиток, которые Костя может убрать по описанным выше правилам. Во второй строке выведите номера плиток в порядке, в котором их следует убирать. Если существует несколько решений, выведите любое из них.

### Examples

mosaic.in	mosaic.out
3 3	1 5
2 1	0
4 4	2 6 11
4 5	3 7 9 13

# Алгоритм ?????????????????????

## Исходный код

```
1 #include <iomanip>
2 #include <limits>
3 #include <iostream>
4 #include <algorithm>
5
6 using namespace std;
7
8 int main() {
9     ifstream in("mosaic.in");
10    ofstream out("mosaic.out");
11
12    int N, M;
13    in >> M >> N;
14    int answ;
15
16    if(N < 3 || M < 3) {
17        out << "0" << endl;
18        return 0;
19    }
20    if((N - 2)*(M - 2) % 2 == 0)
21        answ = (N - 2)*(M - 2) / 2;
22    else
23        answ = (N - 2)*(M - 2) / 2 + 1;
24
25    out << answ << endl;
26
27    int k = 2;
28    for(int i = 2; i < M; i++) {
29        for(int j = k; j < N; j+=2) {
30            out << (i - 1)*N + j << " ";
31        }
32        if(k == 2)
33            k = 3;
34        else
35            k = 2;
36    }
37
38    in.close();
39    out.close();
40
41 }
```

## Результаты

30.	MAI #11: Makarov, Rik, Yakimenko	-	-	-	-	-	+7 3:03	-	+ 0:39	-	2	362	77%	0.20
-----	----------------------------------	---	---	---	---	---	------------	---	-----------	---	---	-----	-----	------

### 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  $n$  ( $1 \leq n \leq 10^5$ ) — the number of soldiers Petya has.

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

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

The next  $m$  lines contains two integers each, separated by a space:  $k_j$  and  $x_j$  ( $1 \leq k_j \leq n$ ,  $1 \leq x_j \leq 10^9$ ) — 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 4 3 7 4 1 5 2 7 1 7 3 3	3
3 6 2 7 3 1 1 2 7 1 6	-1

##### Алгоритм

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

## Исходный код

```
1 #include <iostream>
2 #include <vector>
3 #include <algorithm>
4 #define ll unsigned long long
5
6 using namespace std;
7
8 int main() {
9     ll n, m;
10    ll remColor = 0;
11    ll currentSoldier, currentColor, answer = 0;
12    bool sameColors = true;
13    cin >> n;
14    vector<ll> soldiers(n + 1);
15    for (ll i = 1; i <= n; i++) {
16        cin >> soldiers[i];
17        if (i == 1) remColor = soldiers[i];
18        else if (soldiers[i] != remColor) sameColors = false;
19    }
20    if (sameColors) {
21        cout << "0" << endl;
22        return 0;
23    }
24    cin >> m;
25    sameColors = true;
26    bool flag = true;
27
28    for (ll i = 0; i < m; i++) {
29        sameColors = false;
30        flag = true;
31        cin >> currentSoldier >> currentColor;
32        soldiers[currentSoldier] = currentColor;
33        if (i == 0) remColor = currentColor;
34        if (currentColor == remColor) {
35            for (ll j = 1; j <= n; j++) {
36                if (soldiers[j] != remColor) {
37                    flag = false;
38                    break;
39                }
40            }
41            if (flag) sameColors = true;
42        }
43        if (sameColors) {
44            answer = i + 1;
45            break;
46        }
47        remColor = currentColor;
48    }
49
50    if (sameColors) {
51        cout << answer << endl;
52    } else {
53        cout << "-1" << endl;
54    }
55 }
56
57 return 0;
58 }
```

## 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  $c$  ( $1 \leq a, b, c \leq 10^9$ ), 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

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     long long b, c;
7     cin >> b >> b >> c;
8     if (c > b) cout << "Stay with this envelope" << endl;
9     else cout << "Take another envelope" << endl;
10    return 0;
11 }
```

## 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

```
1 #include <iostream>
2 #include <cmath>
3 #include <vector>
4 #include <algorithm>
5
6 using namespace std;
7
8 int main() {
9     unsigned long long n, s, startIndex, coinsNeeded = 0;
10    cin >> n >> s;
11    vector<unsigned long long> k(n + 1);
12    k[0] = 1;
13    startIndex = n;
14    for (unsigned long long i = 1; i <= n; i++) {
15        cin >> k[i];
16        if (k[i] * k[i - 1] > s) {
17            startIndex = i - 1;
18            break;
19        }
20        k[i] *= k[i - 1];
21    }
22    for (unsigned long long i = startIndex; ; i--) {
23        coinsNeeded += s / k[i];
24        s %= k[i];
25        if (s == 0 || i == 0)
26            break;
27    }
28    cout << coinsNeeded << endl;
29    return 0;
30 }
```

# Результаты

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

#### **4 Журнал по личным контестам Якименко А.В.**