# A. Gambling Nim

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

As you know, the game of "Nim" is played with $n$ piles of stones, where the $i$-th pile initially contains $a_i$ stones. Two players alternate the turns. During a turn a player picks any non-empty pile and removes any positive number of stones from it. The one who is not able to make a move loses the game.

Petya and Vasya are tired of playing Nim, so they invented their own version of the game and named it the "Gambling Nim". They have $n$ two-sided cards, one side of the $i$-th card has number $a_i$ written on it, while the other side has number $b_i$. At the beginning of the game the players put all the cards on the table, each card only one of its sides up, and this side is chosen independently and uniformly. Thus they obtain a sequence $c_1, c_2, \ldots, c_n$, where $c_i$ is equal to $a_i$ or $b_i$. Then they take $n$ piles of stones, with $i$-th pile containing exactly $c_i$ stones and play Nim. Petya takes the first turn.

Given that both players play optimally, find the probability of Petya's victory. Output the answer as an irreducible fraction.

## Input

The first line of the input contains a single integer $n$ ($1 \le n \le 500\,000$) — the number of cards in the deck.

Each of the following $n$ lines contains the description of one card, consisting of two integers $a_i$ and $b_i$ ($0 \le a_i, b_i \le 10^{18}$).

## Output

Output the answer as an irreducible fraction $p / q$. If the probability of Petya's victory is $0$, print `0/1`.

### Examples

**input**

```
2
1 1
1 1
```

**output**

```
0/1
```

**input**

```
2
1 2
1 2
```

**output**

```
1/2
```

**input**

```
3
0 4
1 5
2 3
```

**output**

```
1/1
```

# B. Graph Coloring

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an undirected graph that consists of $n$ vertices and $m$ edges. Initially, each edge is colored either red or blue. Each turn a player picks a single vertex and switches the color of **all** edges incident to it. That is, all red edges with an endpoint in this vertex change the color to blue, while all blue edges with an endpoint in this vertex change the color to red.

Find the minimum possible number of moves required to make the colors of all edges equal.

## Input

The first line of the input contains two integers $n$ and $m$ ($1 \le n, m \le 100\,000$) — the number of vertices and edges, respectively.

The following $m$ lines provide the description of the edges, as the $i$-th of them contains two integers $u_i$ and $v_i$ ($1 \le u_i, v_i \le n$, $u_i \ne v_i$) — the indices of the vertices connected by the $i$-th edge, and a character $c_i$ ($c_i \in \{B, R\}$) providing the initial color of this edge. If $c_i$ equals 'R', then this edge is initially colored red. Otherwise, $c_i$ is equal to 'B' and this edge is initially colored blue. It's guaranteed that there are no self-loops and multiple edges.

## Output

If there is no way to make the colors of all edges equal output $-1$ in the only line of the output. Otherwise first output $k$ — the minimum number of moves required to achieve the goal, then output $k$ integers $a_1, a_2, \ldots, a_k$, where $a_i$ is equal to the index of the vertex that should be used at the $i$-th move.

If there are multiple optimal sequences of moves, output any of them.

### Examples

**input**

```
3 3
1 2 B
3 1 R
3 2 B
```

**output**

```
1
2
```

**input**

```
6 5
1 3 R
2 3 R
3 4 B
4 5 R
4 6 R
```

**output**

```
2
3 4
```

**input**

```
4 5
1 2 R
1 3 R
2 3 B
3 4 B
1 4 B
```

**output**

```
-1
```

# C. Binary Table

time limit per test: 6 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a table consisting of $n$ rows and $m$ columns. Each cell of the table contains either $0$ or $1$. In one move, you are allowed to pick any row or any column and invert all values, that is, replace $0$ by $1$ and vice versa.

What is the minimum number of cells with value 1 you can get after applying some number of operations?

### Input

The first line of the input contains two integers $n$ and $m$ ($1 \leq n \leq 20$, $1 \leq m \leq 100\,000$) — the number of rows and the number of columns, respectively.

Then $n$ lines follows with the descriptions of the rows. Each line has length $m$ and contains only digits '0' and '1'.

### Output

Output a single integer — the minimum possible number of ones you can get after applying some sequence of operations.

#### Example

**input**

```
3 4
0110
1010
0111
```

**output**

```
2
```

# D. International Olympiad

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

International Abbreviation Olympiad takes place annually starting from 1989. Each year the competition receives an abbreviation of form `IAO'y`, where $y$ stands for some number of consequent last digits of the current year. Organizers always pick an abbreviation with non-empty string $y$ that has never been used before. Among all such valid abbreviations they choose the shortest one and announce it to be the abbreviation of this year's competition.

For example, the first three Olympiads (years 1989, 1990 and 1991, respectively) received the abbreviations `IAO'9`, `IAO'0` and `IAO'1`, while the competition in 2015 received an abbreviation `IAO'15`, as `IAO'5` has been already used in 1995.

You are given a list of abbreviations. For each of them determine the year it stands for.

### Input

The first line of the input contains a single integer $n$ ($1 \leq n \leq 1000$) — the number of abbreviations to process.

Then $n$ lines follow, each containing a single abbreviation. It's guaranteed that each abbreviation contains at most nine digits.

### Output

For each abbreviation given in the input, find the year of the corresponding Olympiad.

#### Examples

**input**

```
5
IAO'15
IAO'2015
IAO'1
IAO'9
IAO'0
```

**output**

```
2015
12015
1991
1989
1990
```

**input**

```
4
IAO'9
IAO'99
IAO'999
IAO'9999
```

**output**

```
1989
1999
2999
9999
```

# E. To Hack or not to Hack

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Consider a regular Codeforces round consisting of three problems that uses dynamic scoring.

You are given an almost final scoreboard. For each participant (including yourself), the time of the accepted submission for each of the problems is given. Also, for each solution you already know whether you are able to hack it or not. The only changes in the scoreboard that will happen before the end of the round are your challenges.

What is the best place you may take at the end?

More formally, $n$ people are participating (including yourself). For any problem, if it was solved by exactly $k$ people at the end of the round, the maximum score for this problem is defined as:

1. If $n < 2k \le 2n$, then the maximum possible score is $500$;
2. If $n < 4k \le 2n$, then the maximum possible score is $1000$;
3. If $n < 8k \le 2n$, then the maximum possible score is $1500$;
4. If $n < 16k \le 2n$, then the maximum possible score is $2000$;
5. If $n < 32k \le 2n$, then the maximum possible score is $2500$;
6. If $32k \le n$, then the maximum possible score is $3000$.

Let the maximum possible score for some problem be equal to $s$. Then a contestant who didn't manage to get it accepted (or his solution was hacked) earns $0$ points for this problem. If he got the the solution accepted $t$ minutes after the beginning of the round (and his solution wasn't hacked), he earns $\frac{s \cdot (250 - t)}{250}$ points for this problem.

The overall score of a participant is equal to the sum of points he earns for each problem plus $100$ points for each successful hack (only you make hacks).

The resulting place you get is equal to one plus the number of participants who's overall score is strictly greater than yours.

## Input

The first line of the input contains a single integer $n$ ($1 \le n \le 5000$) — the number of participants. You are the participant number $1$.

Each of the following $n$ lines contains three integers $a_i$, $b_i$ and $c_i$. Here $a_i = 0$ means that the participant number $i$ didn't manage to accept first problem. If $1 \le a_i \le 120$, then the participant number $i$ got the first problem accepted $a_i$ minutes after the start of the contest and you cannot hack this solution. Finally, $-120 \le a_i \le -1$ means that the participant number $i$ got the first problem accepted $-a_i$ minutes after the start of the contest and you can hack this solution. Similarly, $b_i$ and $c_i$ provide the information regarding second and third problems in the same format.

It's guaranteed that integers $a_1$, $b_1$ and $c_1$ are non-negative.

## Output

Print the only integer — the best place you can take at the end of the round.

### Examples

**input**

```
4
120 120 1
61 61 120
-61 61 120
0 0 0
```

**output**

```
1
```

**input**

```
4
0 0 119
-3 -17 -42
0 7 0
51 0 0
```

**output**

```
2
```

## Note

Consider the first sample. If you do not hack any solutions, you will win the contest (scoreboard to the left). However, if you hack the solution of the first problem of the third participant (the only one you can hack), the maximum score for the first problem will change and you will finish second (scoreboard to the right).