
problems

problem	kratica	ispit	kemija	ovce
source file	kratica.pas kratica.c kratica.cpp	ispit.pas ispit.c ispit.cpp	kemija.pas kemija.c kemija.cpp	ovce.pas ovce.c ovce.cpp
input data	stdin			
output data	stdout			
time limit (pentium III 850 mhz)	2 sec			
memory limit	32 MB			
points	40	40	50	70
	200			

kratica

We are given a sentence containing **one or more words** separated with **exactly one space** character.

Acronym of a given sentence is created by taking the **first letter** of each word from the sentence, with the exception that the following **special** words are ignored:

'I', 'pa', 'te', 'ni', 'niti', 'a', 'ali', 'nego', 'no' and 'ili'.

However, if a special word is the **first** word in the sentence, we consider that word very important and always include it in the acronym.

Write a program that will generate the acronym for the given sentence. Acronym is always written in UPPERCASE.

input data

First and only line of input contains the given sentence. The only characters that are allowed in the sentence are lowercase letter of English alphabet ('a' ... 'z') and space character.

Number of characters in the sentence will be at most 100.

output data

First and only line of output should contain the acronym of the given sentence.

examples

input

mirko soft

output

MS

input

biti ali i ne biti

output

BNB

input

ali ja sam i jucer jeo

output

AJSJJ

ispit

Luke does not like studying, but he really wants to pass the extremely hard exam from the first year.

Somebody has told him that all the students from the second year know the answers from the exam, but that is far from real truth. In fact, no one from the second year knows the answers but they all think that somebody else knows.

Luke chooses some student A and asks him about the answers. Student A says that he/she doesn't know, but that he/she thinks that student B knows the answers. After that, Luke is asking student B, and so on...

When he finally realizes that this method will give him no useful results, he is very sad, but then he realizes that he can meet many older students this way, which could be useful some day.

Write a program to find the student which Luke must ask first, so that the total number of students Luke meets in the process is **as high as possible**.

input data

First line of input contains an integer N , $2 \leq N \leq 1000$, the number of students. Students are designated with numbers from 1 to N .

Each of the following N lines contains one number. First line contains the number denoting the answer of student 1, second line contains the number denoting the answer of student 2, and so on...

output data

First and only line of output should contain the designation of the student that Luke should ask first.

If there is more than one solution, output the one with the **smallest** designation.

examples

input	input	input
3	4	6
3	2	2
3	3	3
1	4	4
	1	3
output	output	1
2	1	1
		output
		5

Mickey has to balance one simple chemistry equation.

Equation is the sequence of characters in the following format:

<molecule>+<molecule>=<molecule>

Molecule is the sequence of characters in the following format:

<atom>[number]<atom>[number]...<atom>[number]

Each molecule consists of **one or more** atoms, each atom is labeled by a single character and after each label there can be a number denoting how many of these atoms are appearing. There are three kinds of atoms: C, H or O, and each number is a digit between 2 and 9, inclusive.

For example, HC3OH2 and O4 are molecules, and C2HOH+CH=C5O2H4 is equation.

We said that **equation is balanced** if the total number of atoms of each element is **equal** on both sides of the equation. If equation is not balanced, then we can try to balance it by taking more molecules of the types described by the equation. Each molecule from the equation can be taken **at most 10** times.

Write a program that finds the coefficients to balance the equation.

input data

Initial equation is given in the first and only line of input, in format M1+M2=M3.

Equation will contain at most 50 characters. The only characters that are allowed are: 'C', 'H', 'O', '+', '=', and digits from '2' to '9', inclusive.

output data

First and only line of output should contain 3 integers X1, X2, X3 ($1 \leq X1, X2, X3 \leq 10$), such that the equation $X1 \cdot M1 + X2 \cdot M2 = X3 \cdot M3$ is balanced.

If there is more than one correct solution, output any one of them.

If there are no possibility of balancing the equation by the method we described, output the word **'NEMOGUCE'**.

examples

input

C+CC=CCCCC

output

1 2 1

input

COH+H2O=H3CO2

output

1 1 1

input

HO3H2+O=H2O2O2

output

2 6 3

There is certain number of sheep in Mickey's backyard. While he was firmly sleeping, hungry wolves came in the yard and attacked the sheep.

Yard is of **rectangular** shape and consists of fields arranged in rows and columns. Character '.' (dot) denotes an empty field, character '#' denotes a fence, character 'o' denotes a sheep and character 'v' denotes a wolf.

We consider that **two fields are in the same region** if we can move from one field to the other by a path consisting of **horizontal and vertical moves only** and not containing any fences. Fields from which we can "escape" from the yard are not considered to be a part of any region.

Fortunately, our sheep know karate – they can fight the wolves within a region and win (i.e. kill wolves) if their number in that region **is greater** than number of wolves in the same region. Otherwise, wolves eat all the sheep inside that region.

In the beginning, all the sheep and wolves are situated **inside** regions in the yard.

Write a program that will calculate the number of sheep and the number of wolves still alive in the morning.

input data

First line of input contains two integers, R i C , $3 \leq R, C \leq 250$, the number of rows and the number of columns of Mickey's yard.

Each of the following R lines contains C characters. All of them together represent the layout of the yard i.e. positions of the fences, sheep and wolves in the yard.

Note: 50% of given test data will be "simple" in the sense that the inside area of every region will be of rectangular shape, and within that rectangle there will be no other fences.

output data

First and only line of output should contain two numbers, number of sheep and number of wolves still alive in the morning.

examples

input

```
6 6
...#..
.##v#.
#v.#.#
#.o#.#
.###.#
...###
```

output

```
0 2
```

input

```
8 8
.#####.
#..o...#
#.#.###.#
#.#v.#.#
#.#.o#o#
#o.##...#
#.v..v.#
.#####.
```

output

```
3 1
```

input

```
9 12
.###.#####.
#.oo#...#v#.
#..o#.#.#.#
#..##o#...#
#.#v#o####.#
#..#v#...#
#...v#v####.
.####.#vv.o#
.....####.
```

output

```
3 5
```