A. Did you mean...

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

Beroffice text editor has a wide range of features that help working with text. One of the features is an automatic search for typos and suggestions of how to fix them.

Beroffice works only with small English letters (i.e. with 26 letters from `a` to `z`). Beroffice thinks that a word is typed with a typo if there are three or more consonants in a row in the word. The only exception is that if the block of consonants has all letters the same, then this block (even if its length is greater than three) is not considered a typo. Formally, a word is typed with a typo if there is a block of not less that three consonants in a row, and there are at least two different letters in this block.

For example:

- the following words have typos: "`hellno`", "`hackcerrs`" and "`backtothefutttture`";
- the following words don't have typos: "`helllllooooo`", "`tobeornottobe`" and "`oooooo`".

When Beroffice editor finds a word with a typo, it inserts as little as possible number of spaces in this word (dividing it into several words) in such a way that each of the resulting words is typed without any typos.

Implement this feature of Beroffice editor. Consider the following letters as the only vowels: '`a`', '`e`', '`i`', '`o`' and '`u`'. All the other letters are consonants in this problem.

Input

The only line contains a non-empty word consisting of small English letters. The length of the word is between 1 and 3000 letters.

Output

Print the given word without any changes if there are no typos.

If there is at least one typo in the word, insert the minimum number of spaces into the word so that each of the resulting words doesn't have any typos. If there are multiple solutions, print any of them.

Examples

input
```
hellno
```
output
```
hell no
```

input
```
abacaba
```
output
```
abacaba
```

input
```
asdfasdf
```
output
```
asd fasd f
```

B. Polycarp's phone book

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are n phone numbers in Polycarp's contacts on his phone. Each number is a 9-digit integer, starting with a digit different from 0. All the numbers are distinct.

There is the latest version of Berdroid OS installed on Polycarp's phone. If some number is entered, is shows up all the numbers in the contacts for which there is a substring equal to the entered sequence of digits. For example, is there are three phone numbers in Polycarp's contacts: 123456789, 100000000 and 100123456, then:

- if he enters 00 two numbers will show up: 100000000 and 100123456,
- if he enters 123 two numbers will show up 123456789 and 100123456,
- if he enters 01 there will be only one number 100123456.

For each of the phone numbers in Polycarp's contacts, find the minimum in length sequence of digits such that if Polycarp enters this sequence, Berdroid shows this only phone number.

Input

The first line contains single integer n $(1 \le n \le 70000)$ — the total number of phone contacts in Polycarp's contacts.

The phone numbers follow, one in each line. Each number is a positive 9-digit integer starting with a digit from 1 to 9. All the numbers are distinct.

Output

Print exactly n lines: the i-th of them should contain the shortest non-empty sequence of digits, such that if Polycarp enters it, the Berdroid OS shows up only the i-th number from the contacts. If there are several such sequences, print any of them.

Examples

input

```
3
123456789
100000000
100123456
```

output

```
9
000
01
```

input

```
4
123456789
193456789
134567819
934567891
```

output

```
2
193
81
91
```

C. Tests Renumeration

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The All-Berland National Olympiad in Informatics has just ended! Now Vladimir wants to upload the contest from the Olympiad as a gym to a popular Codehorses website.

Unfortunately, the archive with Olympiad's data is a mess. For example, the files with tests are named arbitrary without any logic.

Vladimir wants to rename the files with tests so that their names are distinct integers starting from 1 without any gaps, namely, "`1`", "`2`", ..., "n', where n is the total number of tests.

Some of the files contain tests from statements (examples), while others contain regular tests. It is possible that there are no examples, and it is possible that all tests are examples. Vladimir wants to rename the files so that the examples are the first several tests, all all the next files contain regular tests only.

The only operation Vladimir can perform is the "`move`" command. Vladimir wants to write a script file, each of the lines in which is "`move file_1 file_2`", that means that the file "`file_1`" is to be renamed to "`file_2`". If there is a file "`file_2`" at the moment of this line being run, then this file is to be rewritten. After the line "`move file_1 file_2`" the file "`file_1`" doesn't exist, but there is a file "`file_2`" with content equal to the content of "`file_1`" before the "`move`" command.

Help Vladimir to write the script file with the minimum possible number of lines so that after this script is run:

- all examples are the first several tests having filenames "`1`", "`2`", ..., "e", where e is the total number of examples;
- all other files contain regular tests with filenames "$e+1$", "$e+2$", ..., "n", where n is the total number of all tests.

Input

The first line contains single integer n $(1 \le n \le 10^5)$ — the number of files with tests.

n lines follow, each describing a file with test. Each line has a form of "`name_i type_i`", where "`name_i`" is the filename, and "`type_i`" equals "`1`", if the i-th file contains an example test, and "`0`" if it contains a regular test. Filenames of each file are strings of digits and small English letters with length from 1 to 6 characters. The filenames are guaranteed to be distinct.

Output

In the first line print the minimum number of lines in Vladimir's script file.

After that print the script file, each line should be "`move file_1 file_2`", where "`file_1`" is an existing at the moment of this line being run filename, and "`file_2`" — is a string of digits and small English letters with length from 1 to 6.

Examples

input

```
5
01 0
2 1
2extra 0
3 1
99 0
```

output

```
4
move 3 1
move 01 5
move 2extra 4
move 99 3
```

input

```
2
1 0
2 1
```

output

```
3
move 1 3
move 2 1
move 3 2
```

input

```
5
1 0
11 1
111 0
1111 1
11111 0
```

output

```
5
move 1 5
move 11 1
move 1111 2
move 111 4
move 11111 3
```

D. Wizard's Tour

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

All Berland residents are waiting for an unprecedented tour of wizard in his Blue Helicopter over the cities of Berland!

It is well-known that there are n cities in Berland, some pairs of which are connected by bidirectional roads. Each pair of cities is connected by no more than one road. It is not guaranteed that the road network is *connected*, i.e. it is possible that you can't reach some city from some other.

The tour will contain several episodes. In each of the episodes:

- the wizard will disembark at some city x from the Helicopter;
- he will give a performance and show a movie for free at the city x;
- he will drive to some neighboring city y using a road;
- he will give a performance and show a movie for free at the city y;
- he will drive to some neighboring to y city z;
- he will give a performance and show a movie for free at the city z;
- he will embark the Helicopter and fly away from the city z.

It is known that the wizard doesn't like to use roads, so he agrees to use each road at most once (regardless of direction). In other words, for road between a and b he only can drive once from a to b, or drive once from b to a, or do not use this road at all.

The wizards wants to plan as many episodes as possible without violation the above rules. Help the wizard!

Please note that the wizard can visit the same city multiple times, the restriction is on roads only.

Input

The first line contains two integers n, m $(1 \le n \le 2 \cdot 10^5, 0 \le m \le 2 \cdot 10^5)$ — the number of cities and the number of roads in Berland, respectively.

The roads description follow, one in each line. Each description is a pair of two integers a_i, b_i $(1 \le a_i, b_i \le n, a_i \ne b_i)$, where a_i and b_i are the ids of the cities connected by the i-th road. It is guaranteed that there are no two roads connecting the same pair of cities. Every road is bidirectional. The cities are numbered from 1 to n.

It is possible that the road network in Berland is not connected.

Output

In the first line print w — the maximum possible number of episodes. The next w lines should contain the episodes in format x, y, z — the three integers denoting the ids of the cities in the order of the wizard's visits.

Examples

input

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

output

```
2
1 4 2
4 3 2
```

input

```
5 8
5 3
1 2
4 5
5 1
2 5
4 3
1 4
3 2
```

output

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

E. Arkady and a Nobody-men

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Arkady words in a large company. There are n employees working in a system of a strict hierarchy. Namely, each employee, with an exception of the CEO, has exactly one immediate manager. The CEO is a manager (through a chain of immediate managers) of all employees.

Each employee has an integer rank. The CEO has rank equal to 1, each other employee has rank equal to the rank of his immediate manager plus 1.

Arkady has a good post in the company, however, he feels that he is nobody in the company's structure, and there are a lot of people who can replace him. He introduced the value of *replaceability*. Consider an employee a and an employee b, the latter being manager of a (not necessarily immediate). Then the replaceability $r(a, b)$ of a with respect to b is the number of subordinates (not necessarily immediate) of the manager b, whose rank is not greater than the rank of a. Apart from replaceability, Arkady introduced the value of *negligibility*. The negligibility z_a of employee a equals the sum of his replaceabilities with respect to all his managers, i.e. $z_a = \sum_b r(a, b)$, where the sum is taken over all his managers b.

Arkady is interested not only in negligibility of himself, but also in negligibility of all employees in the company. Find the negligibility of each employee for Arkady.

Input

The first line contains single integer n $(1 \le n \le 5 \cdot 10^5)$ — the number of employees in the company.

The second line contains n integers $p_1, p_2, \ldots, p_n$ $(0 \le p_i \le n)$, where $p_i = 0$ if the i-th employee is the CEO, otherwise p_i equals the id of the immediate manager of the employee with id i. The employees are numbered from 1 to n. It is guaranteed that there is exactly one 0 among these values, and also that the CEO is a manager (not necessarily immediate) for all the other employees.

Output

Print n integers — the negligibilities of all employees in the order of their ids: $z_1, z_2, \ldots, z_n$.

Examples

input

```
4
0 1 2 1
```

output

```
0 2 4 2
```

input

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

output

```
10 6 3 1 0
```

input

```
5
0 1 1 1 3
```

output

```
0 3 3 3 5
```

Note

Consider the first example:

- The CEO has no managers, thus $z_1 = 0$.
- $r(2, 1) = 2$ (employees 2 and 4 suit the conditions, employee 3 has too large rank). Thus $z_2 = r(2, 1) = 2$.
- Similarly, $z_4 = r(4, 1) = 2$.
- $r(3, 2) = 1$ (employee 3 is a subordinate of 2 and has suitable rank). $r(3, 1) = 3$ (employees $2, 3, 4$ suit the conditions). Thus $z_3 = r(3, 2) + r(3, 1) = 4$.