## Codeforces Round #451 (Div. 2)

# A. Rounding

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

Vasya has a non-negative integer $n$. He wants to round it to nearest integer, which ends up with $0$. If $n$ already ends up with $0$, Vasya considers it already rounded.

For example, if $n = 4722$ answer is $4720$. If $n = 5$ Vasya can round it to $0$ or to $10$. Both ways are correct.

For given $n$ find out to which integer will Vasya round it.

### Input

The first line contains single integer $n$ ($0 \le n \le 10^9$) — number that Vasya has.

### Output

Print result of rounding $n$. Pay attention that in some cases answer isn't unique. In that case print any correct answer.

**Examples**

| input |
| --- |
| 5 |
| **output** |
| 0 |

| input |
| --- |
| 113 |
| **output** |
| 110 |

| input |
| --- |
| 1000000000 |
| **output** |
| 1000000000 |

| input |
| --- |
| 5432359 |
| **output** |
| 5432360 |

### Note

In the first example $n = 5$. Nearest integers, that ends up with zero are $0$ and $10$. Any of these answers is correct, so you can print $0$ or $10$.

# B. Proper Nutrition

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

Vasya has $n$ burles. One bottle of Ber-Cola costs $a$ burles and one Bars bar costs $b$ burles. He can buy any non-negative integer number of bottles of Ber-Cola and any non-negative integer number of Bars bars.

Find out if it's possible to buy some amount of bottles of Ber-Cola and Bars bars and spend **exactly** $n$ burles.

In other words, you should find two non-negative integers $x$ and $y$ such that Vasya can buy $x$ bottles of Ber-Cola and $y$ Bars bars and $x \cdot a + y \cdot b = n$ or tell that it's impossible.

## Input

First line contains single integer $n$ ($1 \le n \le 10\,000\,000$) — amount of money, that Vasya has.

Second line contains single integer $a$ ($1 \le a \le 10\,000\,000$) — cost of one bottle of Ber-Cola.

Third line contains single integer $b$ ($1 \le b \le 10\,000\,000$) — cost of one Bars bar.

## Output

If Vasya can't buy Bars and Ber-Cola in such a way to spend exactly $n$ burles print «NO» (without quotes).

Otherwise in first line print «YES» (without quotes). In second line print two non-negative integers $x$ and $y$ — number of bottles of Ber-Cola and number of Bars bars Vasya should buy in order to spend exactly $n$ burles, i.e. $x \cdot a + y \cdot b = n$. If there are multiple answers print any of them.

Any of numbers $x$ and $y$ can be equal $0$.

**Examples**

| input |
| --- |
| 7<br>2<br>3 |
| **output** |
| YES<br>2 1 |

| input |
| --- |
| 100<br>25<br>10 |
| **output** |
| YES<br>0 10 |

| input |
| --- |
| 15<br>4<br>8 |
| **output** |
| NO |

| input |
| --- |
| 9960594<br>2551<br>2557 |
| **output** |
| YES<br>1951 1949 |

## Note

In first example Vasya can buy two bottles of Ber-Cola and one Bars bar. He will spend exactly $2 \cdot 2 + 1 \cdot 3 = 7$ burles.

In second example Vasya can spend exactly $n$ burles multiple ways:

- buy two bottles of Ber-Cola and five Bars bars;
- buy four bottles of Ber-Cola and don't buy Bars bars;
- don't buy Ber-Cola and buy $10$ Bars bars.

In third example it's impossible to but Ber-Cola and Bars bars in order to spend exactly $n$ burles.

# C. Phone Numbers

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

Vasya has several phone books, in which he recorded the telephone numbers of his friends. Each of his friends can have one or several phone numbers.

Vasya decided to organize information about the phone numbers of friends. You will be given $n$ strings — all entries from Vasya's phone books. Each entry starts with a friend's name. Then follows the number of phone numbers in the current entry, and then the phone numbers themselves. It is possible that several identical phones are recorded in the same record.

Vasya also believes that if the phone number $a$ is a suffix of the phone number $b$ (that is, the number $b$ ends up with $a$), and both numbers are written by Vasya as the phone numbers of the same person, then $a$ is recorded without the city code and it should not be taken into account.

The task is to print organized information about the phone numbers of Vasya's friends. It is possible that two different people have the same number. If one person has two numbers $x$ and $y$, and $x$ is a suffix of $y$ (that is, $y$ ends in $x$), then you shouldn't print number $x$. If the number of a friend in the Vasya's phone books is recorded several times in the same format, it is necessary to take it into account exactly once.

Read the examples to understand statement and format of the output better.

## Input

First line contains the integer $n$ ($1 \le n \le 20$) — number of entries in Vasya's phone books.

The following $n$ lines are followed by descriptions of the records in the format described in statement. Names of Vasya's friends are non-empty strings whose length does not exceed $10$. They consists only of lowercase English letters. Number of phone numbers in one entry is not less than $1$ is not more than $10$. The telephone numbers consist of digits only. If you represent a phone number as a string, then its length will be in range from $1$ to $10$. Phone numbers can contain leading zeros.

## Output

Print out the ordered information about the phone numbers of Vasya's friends. First output $m$ — number of friends that are found in Vasya's phone books.

The following $m$ lines must contain entries in the following format "`name number_of_phone_numbers phone_numbers`". Phone numbers should be separated by a space. Each record must contain all the phone numbers of current friend.

Entries can be displayed in arbitrary order, phone numbers for one record can also be printed in arbitrary order.

**Examples**

| input |
| --- |
| 2<br>ivan 1 00123<br>masha 1 00123 |
| **output** |
| 2<br>masha 1 00123<br>ivan 1 00123 |

| input |
| --- |
| 3<br>karl 2 612 12<br>petr 1 12<br>katya 1 612 |
| **output** |
| 3<br>katya 1 612<br>petr 1 12<br>karl 1 612 |

| input |
| --- |
| 4<br>ivan 3 123 123 456<br>ivan 2 456 456<br>ivan 8 789 3 23 6 56 9 89 2<br>dasha 2 23 789 |
| **output** |
| 2<br>dasha 2 23 789<br>ivan 4 789 123 2 456 |

# D. Alarm Clock

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

Every evening Vitalya sets $n$ alarm clocks to wake up tomorrow. Every alarm clock rings during exactly one minute and is characterized by one integer $a_i$ — number of minute after midnight in which it rings. Every alarm clock begins ringing at the beginning of the minute and rings during whole minute.

Vitalya will definitely wake up if during some $m$ consecutive minutes at least $k$ alarm clocks will begin ringing. Pay attention that Vitalya considers only alarm clocks which begin ringing during given period of time. He doesn't consider alarm clocks which started ringing before given period of time and continues ringing during given period of time.

Vitalya is so tired that he wants to sleep all day long and not to wake up. Find out minimal number of alarm clocks Vitalya should turn off to sleep all next day. Now all alarm clocks are turned on.

## Input

First line contains three integers $n$, $m$ and $k$ ($1 \le k \le n \le 2 \cdot 10^5$, $1 \le m \le 10^6$) — number of alarm clocks, and conditions of Vitalya's waking up.

Second line contains sequence of **distinct** integers $a_1$, $a_2$, ..., $a_n$ ($1 \le a_i \le 10^6$) in which $a_i$ equals minute on which $i$-th alarm clock will ring. Numbers are given in arbitrary order. Vitalya lives in a Berland in which day lasts for $10^6$ minutes.

## Output

Output minimal number of alarm clocks that Vitalya should turn off to sleep all next day long.

**Examples**

| input |
| --- |
| 3 3 2<br>3 5 1 |
| **output** |
| 1 |

| input |
| --- |
| 5 10 3<br>12 8 18 25 1 |
| **output** |
| 0 |

| input |
| --- |
| 7 7 2<br>7 3 4 1 6 5 2 |
| **output** |
| 6 |

| input |
| --- |
| 2 2 2<br>1 3 |
| **output** |
| 0 |

## Note

In first example Vitalya should turn off first alarm clock which rings at minute $3$.

In second example Vitalya shouldn't turn off any alarm clock because there are no interval of $10$ consequence minutes in which $3$ alarm clocks will ring.

In third example Vitalya should turn off any $6$ alarm clocks.

# E. Squares and not squares

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

Ann and Borya have $n$ piles with candies and $n$ is even number. There are $a_i$ candies in pile with number $i$.

Ann likes numbers which are square of some integer and Borya doesn't like numbers which are square of any integer. During one move guys can select some pile with candies and add one candy to it (this candy is new and doesn't belong to any other pile) or remove one candy (if there is at least one candy in this pile).

Find out minimal number of moves that is required to make exactly $n / 2$ piles contain number of candies that is a square of some integer and exactly $n / 2$ piles contain number of candies that is not a square of any integer.

## Input

First line contains one **even** integer $n$ ($2 \le n \le 200\,000$) — number of piles with candies.

Second line contains sequence of integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$) — amounts of candies in each pile.

## Output

Output minimal number of steps required to make exactly $n / 2$ piles contain number of candies that is a square of some integer and exactly $n / 2$ piles contain number of candies that is not a square of any integer. If condition is already satisfied output `0`.

**Examples**

| input |
| --- |
| 4<br>12 14 30 4 |
| **output** |
| 2 |

| input |
| --- |
| 6<br>0 0 0 0 0 0 |
| **output** |
| 6 |

| input |
| --- |
| 6<br>120 110 23 34 25 45 |
| **output** |
| 3 |

| input |
| --- |
| 10<br>121 56 78 81 45 100 1 0 54 78 |
| **output** |
| 0 |

## Note

In first example you can satisfy condition in two moves. During each move you should add one candy to second pile. After it size of second pile becomes $16$. After that Borya and Ann will have two piles with number of candies which is a square of integer (second and fourth pile) and two piles with number of candies which is not a square of any integer (first and third pile).

In second example you should add two candies to any three piles.

# F. Restoring the Expression

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

A correct expression of the form `a+b=c` was written; $a$, $b$ and $c$ are non-negative integers without leading zeros. In this expression, the plus and equally signs were lost. The task is to restore the expression. In other words, one character '+' and one character '=' should be inserted into given sequence of digits so that:

- character'+' is placed on the left of character '=',
- characters '+' and '=' split the sequence into three non-empty subsequences consisting of digits (let's call the left part `a`, the middle part — `b` and the right part — `c`),
- all the three parts `a`, `b` and `c` do not contain leading zeros,
- it is true that `a+b=c`.

It is guaranteed that in given tests answer always exists.

## Input

The first line contains a non-empty string consisting of digits. The length of the string does not exceed $10^6$.

## Output

Output the restored expression. If there are several solutions, you can print any of them.

Note that the answer **at first** should contain two terms (divided with symbol '+'), and then the result of their addition, before which symbol'=' should be.

Do not separate numbers and operation signs with spaces. Strictly follow the output format given in the examples.

If you remove symbol '+' and symbol '=' from answer string you should get a string, **same as** string from the input data.

**Examples**

| input |
| --- |
| 12345168 |
| **output** |
| 123+45=168 |

| input |
| --- |
| 099 |
| **output** |
| 0+9=9 |

| input |
| --- |
| 199100 |
| **output** |
| 1+99=100 |

| input |
| --- |
| 123123123456456456579579579 |
| **output** |
| 123123123+456456456=579579579 |