A. Lights in the Morning

time limit per test: 1 second
memory limit per test: 1024 MB

You woke up late this morning and are in a rush to get to work. You are the kind of person who gets mad if they get unexpectedly stuck at traffic lights, so today, you are going to expect the unexpected!

Your route to work is D kilometres long with N traffic lights along the way. Each traffic light has a red light and a green light. When the light is green, you may pass the traffic light, but when the light is red, you may not pass. When you leave your house, all of the traffic lights are red. For each traffic light, you know the first time that it will turn green. Once this happens, the traffic light will remain green for g minutes, then change to red for r minutes, then change back to green for g minutes, then red for r minutes and so on. These values may be different for each traffic light.

If your car arrives at a traffic light at the moment that it changes (either green-to-red or red-to-green), assume that you will make it through.

You travel at 1 kilometre per minute, so it takes D minutes to complete your journey. Determine whether or not you will get stopped at any of the traffic lights.

Input

The first line of input contains two integers N $(1 \le N \le 1\,000)$, which is the number of traffic lights, and D $(2 \le D \le 10^9)$, which is the length of the journey.

The next N lines describe the traffic lights. Each line contains four integers x $(1 \le x < D)$, which is the location of the traffic light in kilometres from your home, a $(1 \le a \le 10^9)$, which is the number of minutes after leaving home that the traffic light first turns green, g $(1 \le g \le 10^9)$, which is the number of minutes that the light remains green on each cycle, and r $(1 \le r \le 10^9)$, which is the number of minutes that the light remains red on each cycle. The location of the traffic lights are all distinct.

Output

If you will make it through all of the traffic lights without stopping, display `YES`. Otherwise, display `NO`.

Examples

input
```
1 10
5 3 3 3
```
output
```
YES
```

input
```
1 10
2 5 1 1
```
output
```
NO
```

input
```
3 20
5 4 2 3
10 1 2 2
15 10 2 2
```
output
```
YES
```

input
```
2 10
3 2 10 10
6 2 3 2
```
output
```
NO
```

B. Deconstructed Password

time limit per test: 2 seconds
memory limit per test: 1024 megabytes

Bob's password contains only lowercase English letters. Many years ago, he wrote it down and put it in a shoe box. One day, Alice found the password in the shoe box and decided to play a practical joke on him. She first logged into his social media account and changed his profile to say "Alice is the most amazing person ever."
She then replaced his password in the shoe box with a sequence of integers. The sequence was constructed as follows. Let Bob's password be $s_1s_2... s_n$. Here n is the length of Bob's password and s_i is the i-th letter of Bob's password. The sequence Alice made is $a_1, a_2, ..., a_n$. a_i is the first position after i where letter s_i occurs. If no such position exists, then $a_i = n + 1$. That is, $a_i = \min\{j : j = n+1 \text{ or } (i < j \text{ and } s_i = s_j)\}$.

Bob has forgotten his password and has just found Alice's sequence in the shoe box. He knows how she made the sequence and he remembers that his password contained only lowercase English letters. Help Bob determine his password.

Input

The first line of input contains a single integer n $(1 \le n \le 200\,000)$, which is the number of letters in Bob's password.

The second line contains n integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le n + 1)$, which is the sequence Alice made.

Output

Display any string that could be Bob's password, given Alice's sequence. If there are multiple strings that could be Bob's password, display any of them. If it is guaranteed that Alice made a mistake generating her sequence, display -1.

Examples

input	output
11 4 9 10 6 12 8 12 11 12 12 12	abracadabra

input	output
6 3 4 6 5 4 7	-1

input	output
7 4 6 8 8 8 8 8	perplex

input	output
10 2 5 4 6 8 9 8 10 11 11	-1

C. Fine-Tuned Resistance

time limit per test: 2 seconds
memory limit per test: 1024 megabytes

The great inventor, Nikola Tesla is attempting to complete his design for the Tesla Tower that would fulfil his lifelong dream of providing free wireless power to the world. There is a final step in the process in which he requires your assistance.
Tesla needs to obtain a target resistance R using just the n resistance values available to him. For each resistance value, you may imagine that Tesla has an infinite supply of resistors of that resistance value.

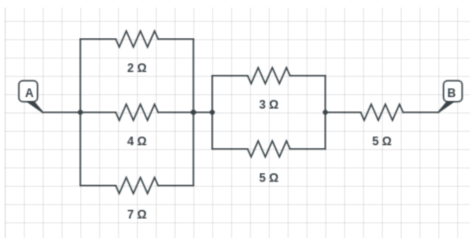
There are a few rules that you have to obey:

- The resistors have to be arranged in a simple resistance network, as defined below.
- At most $5\,000$ resistors can be used.
- The final resistance value obtained must be within 0.01 of the target resistance R.

All resistance values are given in terms of Ω.

Resistance Network. The simple resistance network required by Tesla is composed of some number of parallel networks connected in series. As an example, the simple resistance network below (between nodes A and B) comprises 3 parallel networks. The first parallel network contains 3 resistors, the second contains 2 resistors and the third contains just 1 resistor.

A parallel network must contain at least 1 resistor but the simple resistance network may be composed of 0 parallel networks, in which case the resistance of the network is 0.



Resistance Calculation. The resistance of the simple resistance network is the sum of the resistances of each parallel network.

The resistance, R' of a parallel network consisting of K resistors $(R_1, ...R_K)$ is given by the following equation:

$$\frac{1}{R'} = \sum_{i=1}^{K} \frac{1}{R_i}$$

Input

The first line of the input contains an integer n $(1 \le n \le 10)$, which is the number of different resistor values available to be used, and a real number R $(0.00 < R \le 100.00)$ specified to two digits past the decimal point, which is the target resistance value.

The second line contains n distinct integers $r_1, ..., r_n$ $(1 \le r_i \le 10)$ denoting the resistance values that are available.

Output

Display a valid simple resistance network whose resistance is within 0.01Ω of R (in absolute difference). Specifically, if your network produces a resistance of r, then it must be true that $|R - r| \le 0.01$.

The first line should contain the number of parallel networks used, P. The next P lines should each specify a single parallel network. Each line should begin with the number of resistors used, K followed by K resistor values. You must only use the resistor values given (repetition allowed). The total number of resistors must not exceed $5\,000$.

If there are multiple solutions, any will be accepted. It is guaranteed that a solution exists.

Example

input	output
6 8.00 2 3 4 5 7 10	3 3 2 4 7 2 3 5 1 5

Note

The resistance of the network constructed in the Sample Ouput is:

$$\frac{1}{\frac{1}{2}+\frac{1}{4}+\frac{1}{7}} + \frac{1}{\frac{1}{3}+\frac{1}{5}} + \frac{1}{\frac{1}{5}} = 7.995$$

D. Jon Snow and his Favourite Number

time limit per test: 4 seconds

memory limit per test: 1024 MB

Jon Snow now has to fight with White Walkers. He has n rangers, each of which has his own strength. Also Jon Snow has his favourite number x. Each ranger can fight with a white walker only if the strength of the white walker equals his strength. He however thinks that his rangers are weak and need to improve. Jon now thinks that if he takes the bitwise XOR of strengths of some of rangers with his favourite number x, he might get soldiers of high strength. So, he decided to do the following operation k times:

1. Arrange all the rangers in a straight line in the order of increasing strengths.
2. Take the bitwise XOR (is written as $\oplus$) of the strength of each alternate ranger with x and update it's strength.

Suppose, Jon has 5 rangers with strengths $[9, 7, 11, 15, 5]$ and he performs the operation 1 time with $x = 2$. He first arranges them in the order of their strengths, $[5, 7, 9, 11, 15]$. Then he does the following:

1. The strength of first ranger is updated to $5 \oplus 2$, i.e. 7.
2. The strength of second ranger remains the same, i.e. 7.
3. The strength of third ranger is updated to $9 \oplus 2$, i.e. 11.
4. The strength of fourth ranger remains the same, i.e. 11.
5. The strength of fifth ranger is updated to $15 \oplus 2$, i.e. 13.

The new strengths of the 5 rangers are $[7, 7, 11, 11, 13]$

Now, Jon wants to know the maximum and minimum strength of the rangers after performing the above operations k times. He wants your help for this task. Can you help him?

Input

First line consists of three integers n, k, x ($1 \le n \le 10^5$, $0 \le k \le 10^5$, $0 \le x \le 10^3$) — number of rangers Jon has, the number of times Jon will carry out the operation and Jon's favourite number respectively.

Second line consists of n integers representing the strengths of the rangers $a_1, a_2, ..., a_n$ ($0 \le a_i \le 10^3$).

Output

Output two integers, the maximum and the minimum strength of the rangers after performing the operation k times.

Examples

input

```
5 1 2
9 7 11 15 5
```

output

```
13 7
```

input

```
2 100000 569
605 986
```

output

```
986 605
```

E. Video Cards

time limit per test: 1 second

memory limit per test: 1024 MB

Little Vlad is fond of popular computer game Bota-2. Recently, the developers announced the new add-on named Bota-3. Of course, Vlad immediately bought only to find out his computer is too old for the new game and needs to be updated.

There are n video cards in the shop, the power of the i-th video card is equal to integer value a_i. As Vlad wants to be sure the new game will work he wants to buy not one, but several video cards and unite their powers using the cutting-edge technology. To use this technology one of the cards is chosen as the leading one and other video cards are attached to it as secondary. For this new technology to work it's required that the power of each of the secondary video cards is divisible by the power of the leading video card. In order to achieve that the power of any secondary video card can be reduced to any integer value less or equal than the current power. However, the power of the leading video card should remain unchanged, i.e. it **can't** be reduced.

Vlad has an infinite amount of money so he can buy any set of video cards. Help him determine which video cards he should buy such that after picking the leading video card and may be reducing some powers of others to make them work together he will get the maximum total value of video power.

Input

The first line of the input contains a single integer n ($1 \le n \le 200\,000$) — the number of video cards in the shop.

The second line contains n integers a_1, a_2, ..., a_n ($1 \le a_i \le 200\,000$) — powers of video cards.

Output

The only line of the output should contain one integer value — the maximum possible total power of video cards working together.

Examples

input	output
4 3 2 15 9	27

input	output
4 8 2 2 7	18

Note

In the first sample, it would be optimal to buy video cards with powers 3, 15 and 9. The video card with power 3 should be chosen as the leading one and all other video cards will be compatible with it. Thus, the total power would be $3 + 15 + 9 = 27$. If he buys all the video cards and pick the one with the power 2 as the leading, the powers of all other video cards should be reduced by 1, thus the total power would be $2 + 2 + 14 + 8 = 26$, that is less than 27. Please note, that it's not allowed to reduce the power of the leading video card, i.e. one can't get the total power $3 + 1 + 15 + 9 = 28$.

In the second sample, the optimal answer is to buy all video cards and pick the one with the power 2 as the leading. The video card with the power 7 needs it power to be reduced down to 6. The total power would be $8 + 2 + 2 + 6 = 18$.

F. Dice

time limit per test: 8 seconds
memory limit per test: 1024 MB

You were playing at school when you came across a box of dice on the ground that someone must have left. You liked the look of the dice so much that you decided to take the box home to play with.
In the box, you find n dice, each of which has a certain number of faces. For a die with f faces, the values on its faces are $1, 2, 3, ..., f$. You start playing with the dice, rolling pairs of them and begin to wonder, which pair of dice should you roll if you want to maximise the probability that the sum of their values is between A and B, inclusive?

Input

The first line of input contains three integers n ($2 \le n \le 100\,000$), which is the number of dice, A ($2 \le A \le 100\,000$) and B ($A \le B \le 100\,000$), which define the range of sums you are interested in.

The next n lines describe the dice. Each of these lines contains a single integer f ($1 \le f \le 50\,000$), which is the number of faces on the die.

Output

Display the maximum probability that you can achieve to roll a number between A and B as an irreducible fraction of the form `numerator/denominator`.

Examples

input	output
2 2 6 6 6	5/12

input	output
3 5 15 6 7 8	25/28

input	output
5 1234 2345 1000 2000 3000 4000 5000	52633/100000