CROATIAN OPEN COMPETITION IN INFORMATICS

4th round

Time and memory limits and task points are now located just below the task name.

Great scientific discoveries are often named by the last names of scientists that made them. For example, the most popular asymmetric cryptography system, RSA was discovered by Rivest, Shamir and Adleman. Another notable example is the Knuth-Morris-Pratt algorithm, named by Knuth, Morris and Pratt.
Scientific papers reference earlier works a lot and it's not uncommon for one document to use two different naming conventions: the **short variation** (e.g. KMP) using only the first letters of authors last names and the **long variation** (e.g. Knuth-Morris-Pratt) using complete last names separated by hyphens.
We find mixing two conventions in one paper to be aesthetically unpleasing and would like you to write a program that will transform **long variations into short**.

INPUT

The first and only line of input will contain at most 100 characters, uppercase and lowercase letters of the English alphabet and hyphen ('-' ASCII 45). The first character will always be an uppercase letter. Hyphens will **always** be followed by an uppercase letter. All other characters will be lowercase letters.

OUTPUT

The first and only line of output should contain the appropriate short variation.

SAMPLE TESTS

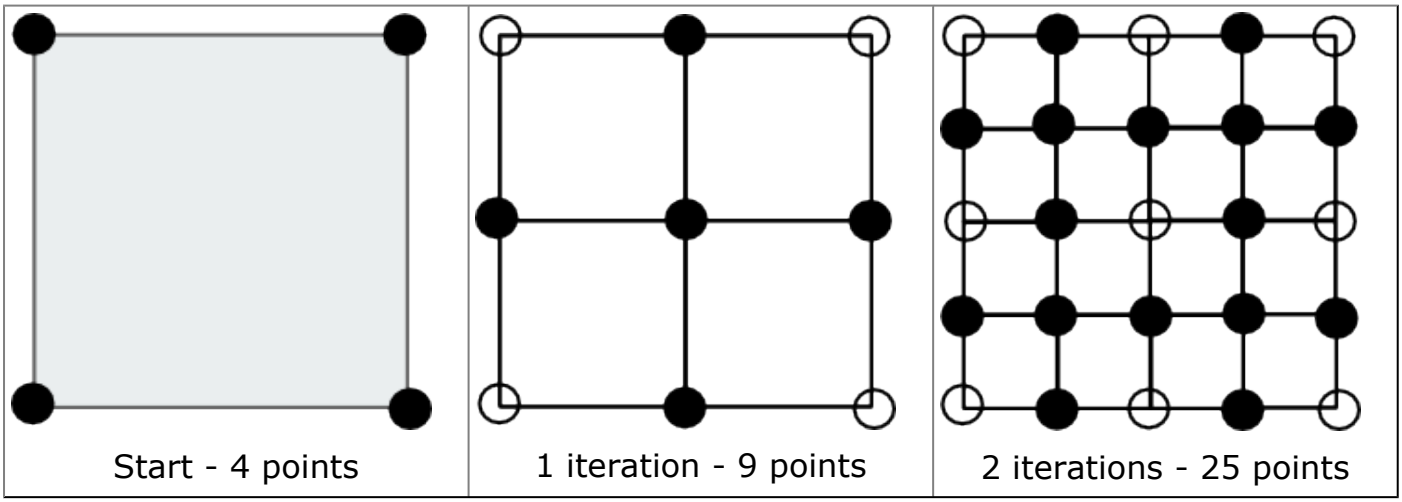
Input:	**Input:**	**Input:**
Knuth-Morris-Pratt	Mirko-Slavko	Pasko-Patak
Output:	**Output:**	**Output:**
KMP	MS	PP

Mirko and Slavko are filming a movie adaptation of the popular SF novel "Chicks in space 13". The script requires them to present a lot of different worlds so they decided to film the entire movie in front of a green screen and add CGI backgrounds later. Mirko heard that the best way to generate artificial terrain is to use **midpoint displacement algorithm**.

To start the algorithm, Mirko selects 4 points forming a perfect square. He then performs the following steps:

1. On each side of the square, he adds a new point in the exact middle of the side. The height of this new point is the average height of the two points on that side.
2. In the exact center of the square he adds a new point whose height is the average height of all 4 square vertices, plus a small random value.

After those two steps are performed, he now has 4 new squares. He performs the same steps on the newly created squares again and again until he is pleased with the results. The following diagram illustrates 2 iterations of the algorithm.



Mirko noticed that some of the points belong to more than one square. In order to decrease memory consumption, he stores calculates and stores such points **only once**. He now wonders how many points in total will he need to store in memory after **N** iterations.

INPUT

The first and only line of input contains one integer **N** ($1 \leq \mathbf{N} \leq 15$), number of iterations.

OUTPUT

The first and only line of output should contain one number, the number of points stored after **N** iterations.

SAMPLE TESTS

Input:	**Input:**	**Input:**
1	2	5
Output:	**Output:**	**Output:**
9	25	1089

COCI 2009/2010 **Task IKS**

4th round, February 13th 2010. 1 second / 32 MB / 70 points

Mirkos great grandmother Katica is an avid mathematician. She likes to torment Mirko with math games. This time she wrote down a sequence of numbers on a piece of paper and told Mirko he may do the following:

- Choose any two numbers in the sequence (let's call them **A** i **B**) and a **prime number X such that A is divisible by X**. After that, Mirko erases **A** and writes $\frac{A}{X}$ in its place. In the end he erases **B** and writes $B \times X$ in its place.

Mirko may perform this operation as many times he wants. His goal is to obtain the maximum possible score, because he gets candy from his great grandmother if he does so. The score for one sequence is the **greatest common divisor of all the numbers in the sequence**.

He is not very good at this, and he likes his candy so he has asked you to help him. Write a program that will calculate the maximum possible score. Since you are such a nice guy, you should also print the **smallest number** of times Mirko must perform the operation to obtain the maximum possible score.

INPUT

The first line of input contains one integer **N**, (1 ≤ **N** ≤ 100), number of elements in the starting sequence.

The second line of input contains **N** positive integers smaller than or equal to **1 000 000**, the sequence Katica gave to Mirko.

OUTPUT

The one and only line of output should contain two integers. The first integer is the maximal possible score Mirko can obtain. The second integer is the smallest number of operations Mirko needs to perform to obtain it.

SAMPLE TESTS

Input:	**Input:**	**Input:**
3 4 4 1	3 8 24 9	5 4 5 6 7 8
Output:	**Output:**	**Output:**
2 1	12 3	2 2

Matija needs to paint his old fence. The fence is made from **N** planks, each 1 cm in width and varying in height. To do this easy and fast, he bought himself a Super Paint Roller Deluxe. The paint roller is **X** cm wide. The Super Paint Roller Deluxe model comes with a catch, however. Matija must at **all times touch the planks with full width of the roller**, otherwise paint drops all around and stains everything. Also, the roller must always be parallel to the ground to prevent leakage. This means that in order for Matija to use the roller safely, he needs to select **X** planks, and paint them from bottom to the top of the lowest plank in one swoop. Then he selects some other **X** planks, paints them and so on.

This leaves parts of some planks unpainted. Matija will have to paint such parts with a toothbrush. This is obviously quite tedious so he asked you to help him **paint as much as possible** using the Super Paint Roller Deluxe. Since there is more than one way to do this he is also interested in the painting that requires the **minimal number of swoops**.

INPUT

The first line of input contains two integers **N** ($1 \leq \mathbf{N} \leq 1\,000\,000$), number of planks, and **X** ($1 \leq \mathbf{X} \leq 100\,000$), width of the Super Paint Roller. Width of the Super Paint Roller will not exceed the width of the fence.

The second line of input contains **N** positive integers, smaller than 1 000 000, heights of planks in the fence.

OUTPUT

The first line of output should contain the **smallest possible area** Matija will have to paint manually.

The second line of output should contain the **smallest number of swoops** needed.

SCORING

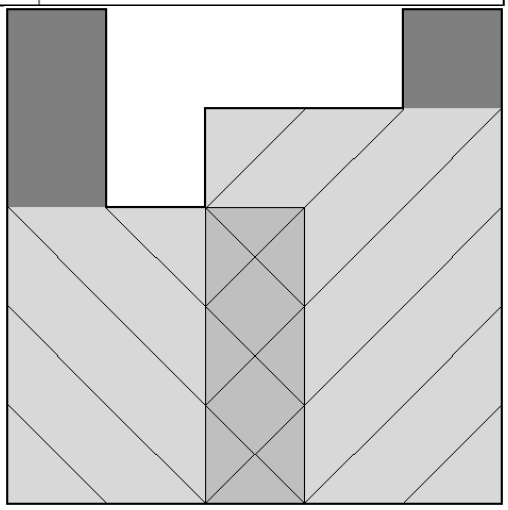
If only one of the output numbers is correct, you will receive 50% of points for that test case. **You must always follow the output format to the letter**, even if you do not calculate both numbers. In such case, you may output any integer in place of the number you did not calculate.

SAMPLE TESTS

Input:	**Input:**	**Input:**
5 3 5 3 4 4 5	10 3 3 3 3 3 3 3 3 3 3 3	7 4 1 2 3 4 3 2 1
Output:	**Output:**	**Output:**
3 2	0 4	4 4

1. sample description:

Matija needs two swoops with his roller - one to paint planks 1, 2 and 3 to the height of 3 cm, the other to paint planks 3, 4 and 5 to the height of 4 cm. Note that 3 cm^2 (2 cm^2 on plank 1 and 1 cm^2 on plank 5) are left unpainted. Also, 3 cm^2 on plank 3 are painted over twice, but that's OK.



COCI 2009/2010 **Task KABOOM**

4th round, February 13th 2010. 0.3 seconds / 32 MB / 120 points

Luka found an interesting tape in the chem lab. The tape is divided into **N** segments of equal length, and can easily be bent between two segment, but only by exactly 180 degrees.

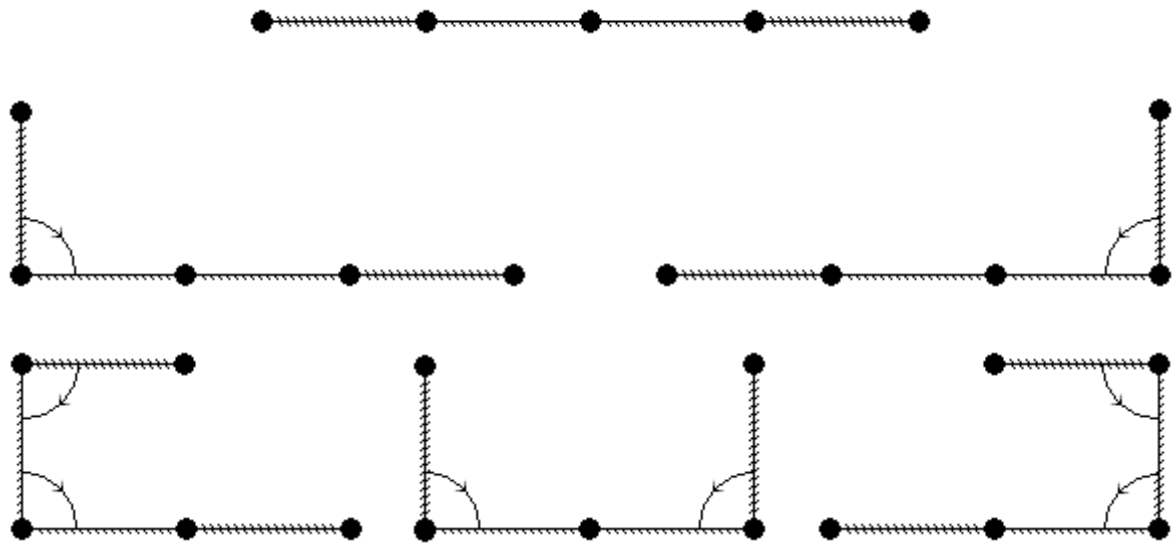
One side of the tape is completely covered with a very volatile chemical. If the chemical comes in contact with itself, it reaches critical mass and explodes.

The other side of the tape is not completely covered yet. Only the first **A** segments and last **B** segments are covered, with the exact same chemical.

Write a program that will calculate the number of different ways Luka can bend the tape so that it does not explode. He can bend the tape more than once and **two ways are different if there is at least one bevel between segments that is not bent in one and is bent in the other**

Since the solution can be huge, print the result modulo 10301.

The following example illustrates all 6 possible ways for **N**=4, **A**=1 and **B**=1. For clarity, the tape is only bent 90 degrees on the illustration. Luka would actually bend it 180 degrees.



INPUT

The first and only line of input contains three natural numbers **N**, **A** and **B** (**A**>0, **B**>0, **A**+**B** ≤ **N** ≤ 1000), total number of segments, number of covered segments from the left and from the right respectively.

OUTPUT

The first and only line of output should contain the number of possible ways to bend the tape modulo **10301**.

SAMPLE TESTS

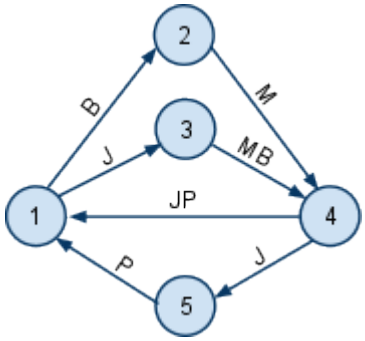
Input: 4 1 1	**Input:** 5 2 2	**Input:** 6 1 2
Output: 6	**Output:** 1	**Output:** 7

Ana has a couple of classmates coming over for crêpes (known as palačinke in Croatian). They are coming in T minutes, and Ana just found out that she has neither one of the four required ingredients (flour, milk, eggs and jam). She hops into her car and drives around her neighbourhood buying supplies.

Her neighbourhood contains **N** crossroads numbered 1 to **N**, and **M** one way roads connecting them. Ana starts on crossroad 1. On each road there is exactly one shop, selling some, maybe all, ingredients.

Ana needs 1 minute to drive down any given road if she does not stop in the shop, or 2 minutes if she does. She needs to obtain all ingredients and drive back to crossroad one in time. She likes to compare shop prices so she may enter a shop even if she already has all ingredients.

Consider the following example with 5 crossroads and 7 roads.



Ana can make the ingredient run in 5 different ways, as shown in the table below.

1. minute	2. minute	3. minute	4. minute	5. minute	6. minute	7. minute
1→3	3→shop→4		4→shop→1			
1→shop→2		2→shop→4		4→shop→1		
1→shop→3		3→shop→4		4→shop→1		
1→shop→3		3→shop→4		4→5	5→shop→1	
1→3	3→shop→4		4→shop→5		5→shop→1	

Write a program that will calculate the number of different ways Ana can buy the ingredients and return home in **T** minutes or less. Since this number can be very large, output it modulo 5557.

INPUT

The first line contains two integers **N** and **M** (1 ≤ **N** ≤ 25, 1 ≤ **M** ≤ 500), number of crossroads and roads.
Each of the next **M** lines contains two different integers **u** and **v** and a string **s**, separated by exactly one space. They describe a road connecting crossroads **u** and **v**, and the shop located on the road selling ingredients **s**.
The string **s** will contain between 1 and 4 uppercase characters. Character 'B' for flour, 'J' for eggs, 'M' for milk and 'P' for jam.
There are at most two direct roads between two crossroads, and only if they are in opposite directions.
The last line contains one integer **T** (1 ≤ **T** ≤ 1 000 000 000), time until Anas friends arrive, in minutes.

OUTPUT

The first and only line of output should contain the number of different ways Ana can buy the ingredients, modulo **5557**.

SAMPLE TESTS

Input:	**Input:**	**Input:**
3 3 1 2 BMJ 2 3 MJP 3 1 JPB 5	3 4 1 2 B 2 1 P 1 3 J 3 1 M 8	5 7 1 2 B 2 4 M 1 3 J 3 4 MB 4 1 JP 4 5 J 5 1 P 7
Output:	**Output:**	**Output:**
3	2	5