Assignments 03

Week 3 of Programming for Data Scientists: Python

Problems

03-A	Steamy Sequence
03-B	Mechanical Melody
03-C	Threshold Tracker
03-D	Mathematical Music
03-E	Dazzling Digits

03-A Steamy Sequence

Time limit: 10s

In the vibrant city of Gearford, where smog-filled skies and towering steam-powered machines dominate the landscape, a strange problem has arisen. The brilliant engineers of Gearford have developed a sequence-transforming machine, powered by the rhythmic hissing of steam and the clanking of gears. The city's inhabitants are eager to use this invention to add a touch of steampunk flair to their numeric sequences.

Given a sequence of n integers, your task is to apply the miracle of Gearford's sequence-transforming machine. The machine operates according to some simple rules inspired by the principles of steam-powered computing:

1. If the number is at a position that is divisible by 3, replace it with `'fizz'` — the sound of steam escaping through pipes.
2. If the number is at a position that is divisible by 5, replace it with `'buzz'` — the distant hum of machinery in motion.
3. If the number is at a position that is divisible by both 3 and 5, replace it with `'FizzBuzz'` — the harmonious convergence of steam and gears.[1]

Otherwise, keep the number unchanged. We will number the positions starting from 0, like a well-organized assembly line in Gearford's factories.

Transform the sequences with the power of steam and gears, and let the city resonate with the harmonious melodies of `'fizz'`, `'buzz'`, and `'FizzBuzz'`.

Input

The first line of input contains a single integer n — the length of the sequence. The next n lines contain the numbers of the sequence. Each number a_i is given in a new line.

Output

Output your resulting list, each element containing either `'fizz'` (if the position is divisible by 3), `'buzz'` (if the position is divisible by 5), `'FizzBuzz'` (if the position is divisible by both 3 and 5), or the original number from the sequence **based on its position**. Please wrap your result in a single list, and print it on a single line.

[1] Note: 0 is divisible by any number, so the first number in the sequence should always be replaced with `'FizzBuzz'`.

Input 1

```
10
15
2
5
4
8
6
4
4
7
8
```

Output 1

```
['FizzBuzz', 2, 5, 'fizz', 8, 'buzz', 'fizz', 4, 7, 'fizz']
```

Sample Input 2

```
3
1
5
4
```

Sample Output 2

```
['FizzBuzz', 5, 4]
```

Input 3

```
4
-1
-2
-1000
-2000
```

Output 3

```
['FizzBuzz', -2, -1000, 'fizz']
```

Input 4

```
16
20
23
25
30
23
29
1
15
30
21
28
4
15
8
17
18
```

Output 4

```
['FizzBuzz', 23, 25, 'fizz', 23, 'buzz', 'fizz', 15, 30, 'fizz', 'buzz', 4, 'f
```

03-B Mechanical Melody

Time limit: 10s

In the heart of Gearford, where the clanking of gears creates a symphony of industrial progress, a linguistic mystery has arisen. The citizens have realized the need for an accurate character count to maintain order in their mechanical communication devices.

This wonderful invention, is designed to count the frequency of each character in a string, excluding the seemingly unnecessary spaces that disrupt the mechanical harmony.

You are to write a program that performs the following steps:

- Read a string from the input.
- Remove all spaces from the given string, ensuring a seamless flow of characters through the cogs of the machine.
- Count the frequency of each character present in the modified string.
- Store this frequency information in a dictionary where the keys are characters, and the values represent their corresponding frequencies.
- Sort the keys of this dictionary in alphabetical order.
- Print this dictionary, unveiling the ordered symphony of characters and their frequencies.

Cogsworth's Character Census Engine is ready to bring order to the chaotic dance of characters.

Input

The input consists of a single string s containing common characters. Spaces may be present, but they should not be counted.[2]

Output

Print the dictionary representing the frequency of characters in the given string. Keep in mind that the keys of the dictionary should be sorted in alphabetical order. Uppercase letters take precedence over lowercase letters, which is Python's default behavior.

[2]Do not worry about CR (\r) characters on Windows, they will not be present on the judge.

Input 1

```
gearford
```

Output 1

```
{'a': 1, 'd': 1, 'e': 1, 'f': 1, 'g': 1, 'o': 1, 'r': 2}
```

Input 2

```
Cogsworth Character Census Engine
```

Output 2

```
{'C': 3, 'E': 1, 'a': 2, 'c': 1, 'e': 3, 'g': 2, 'h': 2, 'i': 1, 'n': 3, 'o': 2
```

Input 3

```
abAB01
```

Output 3

```
{'0': 1, '1': 1, 'A': 1, 'B': 1, 'a': 1, 'b': 1}
```

03-C Threshold Tracker

Time limit: 10s

In the winding streets of Gearford, another great challenge has emerged. The citizens, always seeking efficiency in their endeavors, need a mechanism to track the first occurrence of a number that exceeds a certain threshold in their numerical archives.

Given a list of n integers, your task is to unleash the power of the Cogsworth Threshold Tracker. This brilliant invention, is designed to find the index of the first occurrence of an integer in the list that is greater than or equal to a given threshold, T. If no such integer exists, the tracker elegantly returns the string "Foo".

Cogsworth's Threshold Tracker is ready to navigate the numerical archives with precision. Let the tracker reveal the index of the first occurrence, or gracefully speak the word "Foo" in the absence of such a discovery.

Input

The input consists of two lines:

- The first line contains n space-separated integers — the elements of the list.
- The second line contains a single integer T — the threshold.

Output

Print a single integer — the 0-based index of the first occurrence of an integer in the list which is greater than or equal to T. If no such integer exists, print "Foo", echoing through the city like a distant mechanical resonance.

Sample Input 1	Sample Output 1
`1 2 3 4 5 6 7 8 9` `3`	`2`

Sample Input 2	Sample Output 2
`-4 3 -5` `33`	`Foo`

Sample Input 3	Sample Output 3
`3 2 1` `3`	`0`

Sample Input 4	**Sample Output 4**
1 3 5 7 4	2

03-D Mathematical Music

Time limit: 10s

In the heart of Gearford, where the hum of machines and the rhythmic clatter of gears create a symphony of industrial harmony, a mathematical challenge has been set. Gearford's brilliant mathematicians are attempting to conduct an arithmetic symphony using pairs of numbers, each carefully selected from the city's vast numerical archives.

You are given three inputs: two integers, n and m, and a binary arithmetic operator, denoted as $\circ$. Your task is to harness the mathematical prowess of Gearford and create pairs of numbers that will form the notes. These pairs, crafted from `1` to n and from `m-n+1` to m, will resonate with the operator $\circ$. This will give you pairs like $(1, m-n+1)$, $(2, m-n+2)$, ... (n, m). You may use the function `zip` to help you with this.

For each of these pairs, apply the operator $\circ$ between the two numbers in the pair, and form a new list using the results.

Output the mean (average) of the numbers in this new list.

Cogsworth's Symphony is ready to enchant the mathematicians of Gearford.

Input

The first line contains the integer n. The second line contains the integer m, where $m \geq n$. The third line contains a single character $\circ$, which represents one of the arithmetic operators: `+`, `-`, `*`, `/`.

Output

Output a single floating-point number[3] representing the mean of the new list, the harmonious result of the Arithmetic Symphony.

Explanation of samples

Let's visualize the problem with the first example.

- $1 + 2 = 3$
- $2 + 3 = 5$
- $3 + 4 = 7$
- $4 + 5 = 9$
- $5 + 6 = 11$

[3] Use the `float` type in Python. The judge will accept answers with an error of at most 10^{-4}.

The mean of `[3,5,7,9,11]` is 7.0.

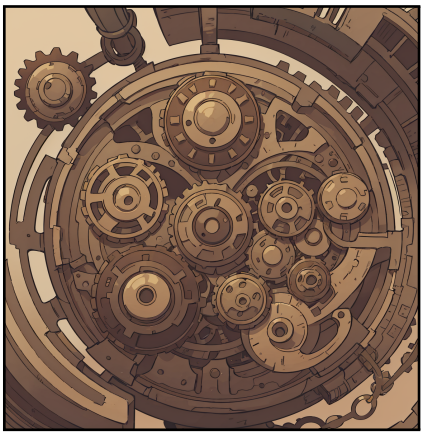
Sample Input 1	**Sample Output 1**
5 6 +	7.0

Sample Input 2	**Sample Output 2**
138 720 +	721.0

Sample Input 3	**Sample Output 3**
8 16 /	0.33712814962814963

03-E Dazzling Digits

Time limit: 10s



A curious challenge has arisen in the gears and cogs of Gearford. The citizens of Gearford seek the expertise of Cogsworth to construct the largest possible number using the digits from a series of positive integers, ensuring that each digit is used at most once.

You are given a series of positive integers. Your goal is to unleash your mathematical potential and construct the largest possible number using the digits of these integers, while preserving the uniqueness of each digit.

Input

The first line contains an integer n — the number of integers. The next n lines each contain a positive integer.

Output

Output a single integer, the largest number that can be constructed by using the digits of all given integers at most once.[4] Traverse the gears of mathematical possibility and unveil the magnificent result.

Sample Input 1	Sample Output 1
2 32 42	432

Sample Input 2	Sample Output 2
5 285 401 32 59 380	98543210

Sample Input 3	Sample Output 3
1 123	321

[4]Hint: You might want to look at the `join` method of strings, if you are having trouble with printing the number.