Triangle Quest

You are given a positive integer N. Print a numerical triangle of height $N-1$ like the one below:

```
1
22
333
4444
55555
......
```

Can you do it using only **arithmetic operations, a single *for* loop and print statement?**

Use no more than two lines. The first line (the *for* statement) is already written for you. You have to complete the print statement.

Note: Using anything related to strings will give a score of 0.

Input Format

A single line containing integer, N.

Constraints

$1 \leq N \leq 9$

Output Format

Print $N-1$ lines as explained above.

Sample Input

```
5
```

Sample Output

```
1
22
333
4444
```

Triangle Quest 2

You are given a positive integer N.
Your task is to print a palindromic triangle of size N.

For example, a palindromic triangle of size 5 is:

```
1
121
12321
1234321
123454321
```

You can't take more than two lines. The first line (a *for*-statement) is already written for you.
You have to complete the code using exactly one print statement.

Note:
Using anything related to *strings* will give a score of 0.
Using more than one *for*-statement will give a score of 0.

Input Format

A single line of input containing the integer N.

Constraints

- $0 < N < 10$

Output Format

Print the palindromic triangle of size N as explained above.

Sample Input

```
5
```

Sample Output

```
1
121
12321
1234321
123454321
```

Classes: Dealing with Complex Numbers

For this challenge, you are given two complex numbers, and you have to print the result of their addition, subtraction, multiplication, division and modulus operations.

The real and imaginary precision part should be correct up to two decimal places.

Input Format

One line of input: The real and imaginary part of a number separated by a space.

Output Format

For two complex numbers C and D, the output should be in the following sequence on separate lines:

- $C + D$
- $C - D$
- $C * D$
- C/D
- $mod(C)$
- $mod(D)$

For complex numbers with non-zero real(A) and complex part(B), the output should be in the following format:
$A + Bi$
Replace the plus symbol $(+)$ with a minus symbol $(-)$ when $B < 0$.

For complex numbers with a zero complex part i.e. real numbers, the output should be:
$A + 0.00i$

For complex numbers where the real part is zero and the complex part(B) is non-zero, the output should be:
$0.00 + Bi$

Sample Input

```
2 1
5 6
```

Sample Output

```
7.00+7.00i
-3.00-5.00i
4.00+17.00i
```

```
0.26-0.11i
2.24+0.00i
7.81+0.00i
```

Concept

Python is a fully object-oriented language like C++, Java, etc. For reading about classes, refer here.

Methods with a double underscore before and after their name are considered as built-in methods. They are used by interpreters and are generally used in the implementation of overloaded operators or other built-in functionality.

```
__add__-> Can be overloaded for + operation
```

```
__sub__ -> Can be overloaded for - operation
```

```
__mul__ -> Can be overloaded for * operation
```

For more information on operator overloading in Python, refer here.

Regex Substitution

The *re.sub()* tool (*sub* stands for *substitution*) evaluates a pattern and, for each valid match, it calls a *method* (or *lambda*).
The *method* is called for all matches and can be used to modify strings in different ways.
The *re.sub()* method returns the modified string as an output.

Learn more about *re. sub()*.

Transformation of Strings

Code

```
import re

#Squaring numbers
def square(match):
    number = int(match.group(0))
    return str(number**2)

print re.sub(r"\d+", square, "1 2 3 4 5 6 7 8 9")
```

Output

```
1 4 9 16 25 36 49 64 81
```

Replacements in Strings

Code

```
import re

html = """
<head>
<title>HTML</title>
</head>
<object type="application/x-flash"
  data="your-file.swf"
  width="0" height="0">
  <!-- <param name="movie"  value="your-file.swf" /> -->
  <param name="quality" value="high"/>
</object>
"""

print re.sub("(<!--.*?-->)", "", html) #remove comment
```

Output

```
<head>
<title>HTML</title>
</head>
<object type="application/x-flash"
  data="your-file.swf"
  width="0" height="0">

  <param name="quality" value="high"/>
</object>
```

Task

You are given a text of N lines. The text contains `&&` and `||` symbols.
Your task is to modify those symbols to the following:

```
&& → and
|| → or
```

Both `&&` and `||` should have a *space* " " on both sides.

Input Format

The first line contains the integer, N.
The next N lines each contain a line of the text.

Constraints

$0 < N < 100$

Neither `&&` nor `||` occur in the *start* or *end* of each line.

Output Format

Output the modified text.

Sample Input

```
11
a = 1;
b = input();

if a + b > 0 && a - b < 0:
    start()
elif a*b > 10 || a/b < 1:
    stop()
print set(list(a)) | set(list(b))
#Note do not change &&& or ||| or & or |
#Only change those '&&' which have space on both sides.
#Only change those '|| which have space on both sides.
```

Sample Output

```
a = 1;
b = input();

if a + b > 0 and a - b < 0:
    start()
elif a*b > 10 or a/b < 1:
    stop()
print set(list(a)) | set(list(b))
#Note do not change &&& or ||| or & or |
#Only change those '&&' which have space on both sides.
#Only change those '|| which have space on both sides.
```

Write a function

An extra day is added to the calendar almost every four years as February 29, and the day is called a *leap day*. It corrects the calendar for the fact that our planet takes approximately 365.25 days to orbit the sun. A leap year contains a leap day.

In the Gregorian calendar, three conditions are used to identify leap years:

- The year can be evenly divided by 4, is a leap year, unless:
  - The year can be evenly divided by 100, it is NOT a leap year, unless:
    - The year is also evenly divisible by 400. Then it is a leap year.

This means that in the Gregorian calendar, the years 2000 and 2400 are leap years, while 1800, 1900, 2100, 2200, 2300 and 2500 are NOT leap years. Source

Task

Given a year, determine whether it is a leap year. If it is a leap year, return the Boolean `True`, otherwise return `False`.

Note that the code stub provided reads from STDIN and passes arguments to the `is_leap` function. It is only necessary to complete the `is_leap` function.

Input Format

Read $year$, the year to test.

Constraints

$1900 \leq year \leq 10^5$

Output Format

The function must return a Boolean value (True/False). Output is handled by the provided code stub.

Sample Input 0

```
1990
```

Sample Output 0

```
False
```

Explanation 0

1990 is not a multiple of 4 hence it's not a leap year.

Compress the String!

In this task, we would like for you to appreciate the usefulness of the *groupby()* function of *itertools* . To read more about this function, Check this out .

You are given a string S. Suppose a character 'c' occurs consecutively X times in the string. Replace these consecutive occurrences of the character 'c' with $(X, \ c)$ in the string.

For a better understanding of the problem, check the explanation.

Input Format

A single line of input consisting of the string S.

Output Format

A single line of output consisting of the modified string.

Constraints

All the characters of S denote integers between 0 and 9.

$1 \leq \mid S \mid \leq 10^4$

Sample Input

```
1222311
```

Sample Output

```
(1, 1) (3, 2) (1, 3) (2, 1)
```

Explanation

First, the character 1 occurs only once. It is replaced by $(1, \ 1)$. Then the character 2 occurs three times, and it is replaced by $(3, \ 2)$ and so on.

Also, note the single space within each compression and between the compressions.

Iterables and Iterators

The *itertools* module standardizes a core set of fast, memory efficient tools that are useful by themselves or in combination. Together, they form an *iterator algebra* making it possible to construct specialized tools succinctly and efficiently in pure *Python*.

To read more about the functions in this module, check out their documentation here.

You are given a list of N lowercase English letters. For a given integer K, you can select any K indices (assume 1-based indexing) with a uniform probability from the list.

Find the *probability* that *at least* one of the K indices selected will contain the letter: 'a'.

Input Format

The input consists of three lines. The first line contains the integer N, denoting the length of the list. The next line consists of N space-separated lowercase English letters, denoting the elements of the list.

The third and the last line of input contains the integer K, denoting the number of indices to be selected.

Output Format

Output a single line consisting of the *probability* that *at least* one of the K indices selected contains the letter:'a'.

Note: The answer must be correct up to 3 decimal places.

Constraints

$1 \leq N \leq 10$

$1 \leq K \leq N$

All the letters in the list are lowercase English letters.

Sample Input

```
4
a a c d
2
```

Sample Output

```
0.8333
```

Explanation

All possible unordered tuples of length 2 comprising of indices from 1 to 4 are:

$$(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)$$

Out of these 6 combinations, 5 of them contain either index 1 or index 2 which are the indices that contain the letter 'a'.

Hence, the answer is $\frac{5}{6}$.

ginortS

You are given a string S.
S contains alphanumeric characters only.

Your task is to sort the string S in the following manner:

- All sorted *lowercase letters* are ahead of *uppercase letters*.
- All sorted *uppercase letters* are ahead of digits.
- All sorted *odd digits* are ahead of sorted *even digits*.

Input Format

A single line of input contains the string S.

Constraints

- $0 < len(S) < 1000$

Output Format

Output the sorted string S.

Sample Input

```
Sorting1234
```

Sample Output

```
ginortS1324
```

Validating Credit Card Numbers

You and Fredrick are good friends. Yesterday, Fredrick received N credit cards from **ABCD Bank**. He wants to verify whether his credit card numbers are valid or not. You happen to be great at regex so he is asking for your help!

A valid credit card from **ABCD Bank** has the following characteristics:

► It must start with a 4, 5 or 6.
► It must contain exactly 16 digits.
► It must only consist of digits (0-9).
► It may have digits in groups of 4, separated by *one* hyphen **"-"**.
► It must `NOT` use any other separator like ' ' , '_', etc.
► It must `NOT` have 4 or more consecutive repeated digits.

Examples:

Valid Credit Card Numbers

```
4253625879615786
4424424424442444
5122-2368-7954-3214
```

Invalid Credit Card Numbers

```
42536258796157867       #17 digits in card number → Invalid
4424444424442444        #Consecutive digits are repeating 4 or more times → Invalid
5122-2368-7954 - 3214   #Separators other than '-' are used → Invalid
44244x4424442444        #Contains non digit characters → Invalid
0525362587961578        #Doesn't start with 4, 5 or 6 → Invalid
```

Input Format

The first line of input contains an integer N.
The next N lines contain credit card numbers.

Constraints

$0 < N < 100$

Output Format

Print 'Valid' if the credit card number is valid. Otherwise, print 'Invalid'. Do not print the quotes.

Sample Input

```
6
4123456789123456
```

```
5123-4567-8912-3456
61234-567-8912-3456
4123356789123456
5133-3367-8912-3456
5123 - 3567 - 8912 - 3456
```

Sample Output

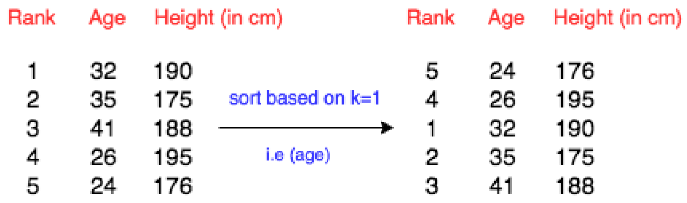
```
Valid
Valid
Invalid
Valid
Invalid
Invalid
```

Explanation

4123456789123456 : **Valid**
5123-4567-8912-3456 : **Valid**
61234-567-8912-3456 : **Invalid**, because the card number is not divided into equal groups of 4.
4123356789123456 : **Valid**
51$33$-$33$67-8912-3456 : **Invalid**, consecutive digits 3333 is repeating 4 times.
5123 — 4567 — 8912 — 3456 : **Invalid**, because space ' ' and `-` are used as separators.

Athlete Sort

You are given a spreadsheet that contains a list of N athletes and their details (such as age, height, weight and so on). You are required to sort the data based on the K^{th} attribute and print the final resulting table. Follow the example given below for better understanding.



Rank	Age	Height (in cm)
1	32	190
2	35	175
3	41	188
4	26	195
5	24	176

sort based on k=1 → i.e (age)

Rank	Age	Height (in cm)
5	24	176
4	26	195
1	32	190
2	35	175
3	41	188

Note that K is indexed from 0 to $M-1$, where M is the number of attributes.

Note: If two attributes are the same for different rows, for example, if two atheletes are of the same age, print the row that appeared first in the input.

Input Format

The first line contains N and M separated by a space.
The next N lines each contain M elements.
The last line contains K.

Constraints

$1 \leq N, M \leq 1000$
$0 \leq K < M$
Each element ≤ 1000

Output Format

Print the N lines of the sorted table. Each line should contain the space separated elements. Check the sample below for clarity.

Sample Input 0

```
5 3
10 2 5
7 1 0
9 9 9
1 23 12
6 5 9
1
```

Sample Output 0

```
7 1 0
10 2 5
6 5 9
9 9 9
1 23 12
```

Explanation 0

The details are sorted based on the second attribute, since K is zero-indexed.