Julius Caesar protected his confidential information by encrypting it using a cipher. Caesar's cipher shifts each letter by a number of letters. If the shift takes you past the end of the alphabet, just rotate back to the front of the alphabet. In the case of a rotation by 3, w, x, y and z would map to z, a, b and c.

```
Original alphabet:      abcdefghijklmnopqrstuvwxyz
Alphabet rotated +3:    defghijklmnopqrstuvwxyzabc
```

Example

$s =$ **There's-a-starman-waiting-in-the-sky**

$k = 3$

The alphabet is rotated by 3, matching the mapping above. The encrypted string is **Wkhuh'v-d-vwdupdq-zdlwlqj-lq-wkh-vnb**.

Note: The cipher only encrypts letters; symbols, such as -, remain unencrypted.

Function Description

Complete the caesarCipher function in the editor below.

caesarCipher has the following parameter(s):

- string s: cleartext
- int k: the alphabet rotation factor

Returns

- string: the encrypted string

Input Format

The first line contains the integer, n, the length of the unencrypted string.

The second line contains the unencrypted string, s.

The third line contains k, the number of letters to rotate the alphabet by.

Constraints

$1 \leq n \leq 100$

$0 \leq k \leq 100$

s is a valid ASCII string without any spaces.

Sample Input

```
11
middle-Outz
2
```

Sample Output

```
okffng-Qwvb
```

Explanation

```
Original alphabet:     abcdefghijklmnopqrstuvwxyz
Alphabet rotated +2:   cdefghijklmnopqrstuvwxyzab

m -> o
i -> k
d -> f
d -> f
l -> n
e -> g
-   -
O -> Q
u -> w
t -> v
z -> b
```

Change Theme Language Python 3

```python
#!/bin/python3

import math
import os
import random
import re
import sys

#
# Complete the 'caesarCipher' function below.
#
# The function is expected to return a STRING.
# The function accepts following parameters:
#  1. STRING s
#  2. INTEGER k
#
def caesarCipher(s, k):
    string = s
    key    = k
    # Write your code here
    alphabets = list('abcdefghijklmnopqrstuvwxyz')
    alpha_str = 'abcdefghijklmnopqrstuvwxyz'

    result = ''
    for char in string:
        # Algorithm
        # Find the index of the current character in the list of alphabets
        # Resultant element's index in the alphabets list will be the circular incrementaion of the current character's index
        # Append the new character to the result
        if alpha_str.find(char) != -1:
                result += alphabets[ (alphabets.index(char) + key) % 26 ]
        else:
            # it could be an upper case character or a special character
            if alpha_str.find(char.lower()) != -1:
                # uppercase
                result += alphabets[ (alphabets.index(char.lower()) + key) % 26 ].upper()
            else:
                # special
                result += char

    return result
```

Line: 39 Col: 31

Upload Code as File Test against custom input Run Code Submit Code

Congratulations

You solved this challenge. Would you like to challenge your friends?

Test case 5

Test case 6

Compiler Message

Success