Caesar

Implement a program that encrypts messages using Caesar's cipher, per the below.

```
$ python caesar.py 13
plaintext: HELLO
ciphertext: URYYB
```

Implementation Details

Design and implement a program, caesar.py, that encrypts messages using Caesar's cipher.

Your program must accept a single command-line argument, a non-negative integer. Let's call it k for the sake of discussion.

If your program is executed without any command-line arguments or with more than one command-line argument, your program should print an error message of your choice (with print) and exit immediately with a status code of 1.

You can assume that, if a user does provide a command-line argument, it will be a non-negative integer (e.g., 1). No need to check that it's indeed numeric.

Do not assume that k will be less than or equal to 26. Your program should work for all non-negative integral values of k less than 231 - 26. In other words, you don't need to worry if your program eventually breaks if the user chooses a value for k that's too big or almost too big to fit in an int. But, even if k is greater than 26, alphabetical characters in your program's input should remain alphabetical characters in your program's output. For instance, if k is 27, the letter A should not become "[" even though "[" is 27 positions away from A in ASCII, per asciichart.com; A should become B, since B is 27 positions away from A, provided you wrap around from Z to A.

Your program must output plaintext: (without a newline) and then prompt the user for a string of plaintext:.

Your program must output ciphertext: (without a newline) followed by the plaintext's corresponding ciphertext, with each alphabetical character in the plaintext "rotated" by k positions; non-alphabetical characters should be outputted unchanged.

Your program must preserve case: capitalized letters, though rotated, must remain capitalized letters; lowercase letters, though rotated, must remain lowercase letters.

After outputting ciphertext, you should print a newline.

Usage

Your program should behave per the examples below. Assume that the underlined text is what some user has typed:

```
$ python caesar.py 1
plaintext: HELLO
ciphertext: IFMMP

$ python caesar.py 13
plaintext: hello, world
ciphertext: uryyb, jbeyq

$ python caesar.py 13
plaintext: be sure to drink your Ovaltine
ciphertext: or fher gb qevax lbhe Binygvar

$ python caesar.py
Usage: python caesar.py k
```

```
$ python caesar.py 1 2 3 4 5
Usage: python caesar.py k
```