Gravity Falls Cryptogram Solver Version 1: Caesar cipher

The mysteries of Gravity Falls must be discovered.

Help Dipper and Mabel discover the secret messages by writing a program that will help you uncover the mysteries of a Caesar cipher!

The Caesar cipher is a shifting cipher. Each letter in the message is shifted by a certain number of positions in the alphabet.

For example, to **encrypt** the message "AY" using Caesar cipher with shift 3, you would need to shift each letter by 3 positions:

- Letter **A** is shifted 3 positions to the right, so it becomes **D**.
- Letter **Y** is shifted 3 positions to the right, so it becomes **B**.

The resulting message would be "DB".

So, to **decrypt** the message, you would need to shift each letter by 3 positions ***in the opposite direction.***

In this lab, you will implement **ONLY decryption**, don't worry about encrypting messages. You will implement and test a function that will decrypt an encrypted message. The messages used in the test cases have been encrypted using Caesar cipher with shift 3 **but your function should work with ANY shift, not only 3**.

Your function should correctly **decrypt** both upper-case and lower-case letters and keep all other characters (whitespace, punctuation, etc.) intact.

What to do

Implement the `decrypt_caesar` function that would decrypt a given message with the Caesar cipher.

You **must** use the following template:

```
def decrypt_caesar(text: str, shift: int) -> str:
    """
    Decipher a text (Caesar cipher).
    """
    # TODO: Implement this function
    pass

def main() -> None:
    """
    Main program.
    """
    text = input("Enter a text to decipher: ")
    # Call the function and Print the deciphered text

main()
```

Hints

- Consider importing the string module in order to generate all letters of the alphabets.Try the following in the Python shell to understand how it works:
  ```
  >>> import string
  >>> capital_letters = string.ascii_uppercase
  >>> capital_letters
  'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
  ```
- Think about how to iterate over a string.
- Consider strings with a mix of uppercase and lowercase letters plus characters other than letters from the alphabet.
- Consider using a modulus operator (%) to ensure that a generated number remains confined within a specific range.
- Remember that strings are immutable.

Program name

Save your program as `gravity1.py`.

Demo

https://asciinema.org/a/zHWrmdTvpHu2V6aCGQ3ojFbxU

Testing

To make sure your program works correctly, you should test it.

Test Case 1

Run your program with `python gravity1.py`. Type `VWDQ LV QRW ZKDW KH VHHPV.`, then press Enter. Your program should print:

```
STAN IS NOT WHAT HE SEEMS.
```

Test Case 2

Run your program with `python gravity1.py`. Type `Sbwkrq lv ixq! :-)`, then press Enter. Your program should print:

```
Python is fun! :-)
```

Submitting

Submit `gravity1.py` via eClass.

You may submit either all versions you complete, or only the final version.