36. Write a python program for Caesar cipher, known as the affine Caesar cipher, has the following form: For each plaintext letter p, substitute the ciphertext letter C: C = E([a, b], p) = (ap + b) mod 26 A basic requirement of any encryption algorithm is that it be one-to-one. That is, if p q, then E(k, p) E(k, q). Otherwise, decryption is impossible, because more than one plaintext character maps into the same ciphertext character. The affine Caesar cipher is not one-to-one for all values of a. For example, for a = 2 and b = 3, then E([a, b], 0) = E([a, b], 13) = 3.

Code:

```
import string

def mod_inverse(a, m):
    """Compute modular inverse of a modulo m, if it exists"""
    for x in range(1, m):
        if (a * x) % m == 1:
            return x
    raise ValueError(f"No modular inverse for a={a} modulo {m}")
def is_valid_a(a):
    """Check if 'a' is valid (coprime with 26) for one-to-one mapping"""
    from math import gcd
    return gcd(a, 26) == 1
def affine_encrypt(plaintext, a, b):
    if not is_valid_a(a):
        raise ValueError(f"'a'={a} is invalid. It must be coprime with 26.")
    plaintext = plaintext.upper()
    ciphertext = ""
    for char in plaintext:
        if char in string.ascii_uppercase:
            p = ord(char) - ord('A')
            c = (a * p + b) % 26
            ciphertext += chr(c + ord('A'))
```

```
        else:
            ciphertext += char
    return ciphertext
def affine_decrypt(ciphertext, a, b):
    if not is_valid_a(a):
        raise ValueError(f"'a'={a} is invalid. It must be coprime with 26.")
    a_inv = mod_inverse(a, 26)
    plaintext = ""
    for char in ciphertext:
        if char in string.ascii_uppercase:
            c = ord(char) - ord('A')
            p = (a_inv * (c - b)) % 26
            plaintext += chr(p + ord('A'))
        else:
    plaintext += char
    return plaintext
#-------------------------
# Example usage
# -------------------------
plaintext = "HELLO AFFINE CIPHER"
a = 5   # must be coprime with 26
b = 8
ciphertext = affine_encrypt(plaintext, a, b)
decrypted = affine_decrypt(ciphertext, a, b)
print("Plaintext :", plaintext)
print("Ciphertext:", ciphertext)
print("Decrypted :", decrypted)
```

```
>>>
========== RESTART: C:/Users/Maria/OneDrive/Documents/Ex36.py ========
Plaintext : HELLO AFFINE CIPHER
Ciphertext: RCLLA IHHWVC SWFRCP
Decrypted : HELLO AFFINE CIPHER
>>>
```