Question Number 4

An affine cryptosystem is given by the following encryption function, where a, b are chosen from $\mathbb{Z}_{26}$.

$$enc_{a,b} : \mathbb{Z}_{26} \to \mathbb{Z}_{26}$$
$$x \to \text{ax + b}\epsilon\mathbb{Z}_{26}$$

- Encrypt the plaintext cryptography using the affine code $enc_{3,5}$. What is the decryption function corresponding to $enc_{3,5}$? Decrypt the ciphertext XRHLAFUUK.
- A central requirement of cryptography is that the plaintext must be computable from the key and the ciphertext. Explain why $enc_{2,3}$ violates this rule. Show that the function enca,b satisfies the rule if and only if $gcd(a, 26) = 1$.
- In the following we consider only functions enca,b with $gcd(a, 26) = 1$. Show that all affine codes with $b = 0$ map the letter a to a and the letter n to n.

Solution.

Part 1: Encryption and Decryption

We are given the plaintext *cryptography* and the affine cipher parameters $a = 3$ and $b = 5$. The encryption function is given by:

$$\text{enc}_{3,5}(x) = 3x + 5 \mod 26$$

The decryption function can be derived by finding the modular inverse of a modulo 26. Let's calculate it using the Python code provided.

```python
def modinv(a, m):
    for x in range(1, m):
        if (a * x) % m == 1:
            return x
    return None

def encrypt(mssge,a,b):
    c = ""
    for i in mssge:
        if i.isalpha():
            if i.islower():
                c += chr(((a*(ord(i)-97)+b)%26)+97)
            else:
                c += chr(((a*(ord(i)-65)+b)%26)+65)
        else:
            c += i
    return c

def decrypt(cipher, a, b):
    mssge = ""
    for i in cipher:
        if i.isalpha():
            if i.islower():
                mssge += chr(((modinv(a, 26)*(ord(i)-97-b))%26)+97)
```

```python
            else:
                mssge += chr(((modinv(a, 26)*(ord(i)-65-b))%26)+65)
        else:
            mssge += i
    return mssge

cipher = encrypt("cryptography", 3, 5)
print(cipher)
print(decrypt("XRHLAFUUK", 3, 5))
```

Algorithm 1: Affine Cipher Encryption and Decryption

The ciphertext for *cryptography* using $a = 3$ and $b = 5$ is **XRHLAFUUK**. The corresponding decryption function yields the original plaintext when applied to the ciphertext.

Part 2: Why does $enc_{2,3}$ violate the rule?

A central requirement in cryptography is that the plaintext must be computable from the key and the ciphertext. For $enc_{2,3}$, the encryption function is:

$$\text{enc}_{2,3}(x) = 2x + 3 \mod 26$$

However, the key $a = 2$ is not invertible modulo 26 because $\gcd(2, 26) \neq 1$. Specifically, $\gcd(2, 26) = 2$, which means there is no unique inverse for 2 modulo 26, and therefore, decryption is not guaranteed to retrieve the original plaintext. This violates the requirement that the plaintext should be retrievable from the ciphertext and the key.

Part 3: All affine codes with $b = 0$ map the letter a to a and the letter n to n.

When $b = 0$, the affine encryption function simplifies to:

$$\text{enc}_{a,0}(x) = ax \mod 26$$

If we take $x = 0$ (which corresponds to the letter 'a'), then:

$$\text{enc}_{a,0}(0) = 0 \mod 26$$

which means 'a' is mapped to 'a'.

Similarly, for $x = 13$ (which corresponds to the letter 'n'):

$$\text{enc}_{a,0}(13) = 13a \mod 26$$

Since $13a \mod 26 = 13$ (as long as a is odd and $\gcd(a, 26) = 1$), the letter 'n' is mapped to 'n'.