Introduction

The Affine Cipher is a type of monoalphabetic substitution cipher that combines modular arithmetic with linear algebra to encrypt and decrypt text. Each character in the plaintext is mapped to its numeric equivalent, encrypted using a mathematical function, and then converted back to a letter. This cipher provides a basic method for secure communication, though it can be vulnerable to brute-force attacks if the key space is small.

Overview

This report presents the implementation of the Affine Cipher in Python, covering 4 phases:

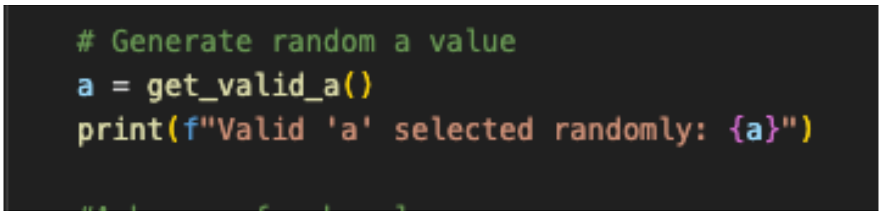
- Encryption: Transforming plain text into ciphertext using a mathematical formula.
- Decryption: Recovering the original plaintext from the ciphertext using an inverse function.
- Brute-Force Attack: Attempting to break the cipher by testing all possible keys (using a customized dictionary and calculating time)
- Display output

1. Data Encryption

The encryption process in the Affine Cipher follows the formula:

$$E(x)=(a\times x+b) \bmod 26$$

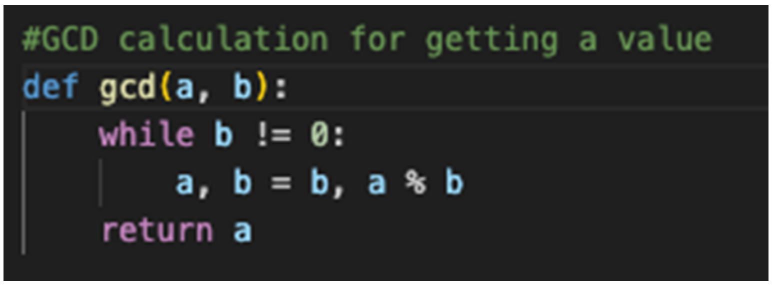
1.1 Generating a random "a" value



```python
# Generate random a value
a = get_valid_a()
print(f"Valid 'a' selected randomly: {a}")
```

Figure 1 – Generating a random value

1.1.1 Adding the gcd function for getting the right value



```python
#GCD calculation for getting a value
def gcd(a, b):
    while b != 0:
        a, b = b, a % b
    return a
```

Figure 2 – Adding the gcd Function

1.1.2 Random value