5. Write a python program for generalization of the 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.

a. Are there any limitations on the value of b?

b. Determine which values of a are not allowed

Code:

```
def affine_encrypt(text, a, b):
  result = ""
   for char in text.lower():
   if char.isalpha():
   p = ord(char) - ord('a')
   c = (a * p + b) % 26
    result += chr(c + ord('a'))
     else:
     result += char
    return result
    def affine_decrypt(cipher, a, b):
    # Find multiplicative inverse of a mod 26
    for x in range(26):
        if (a * x) % 26 == 1:
```

```
 a_inv = x
            break
 result = ""
    for char in cipher.lower():
       if char.isalpha():
        c = ord(char) - ord('a')
        p = (a_inv * (c - b)) % 26
        result += chr(p + ord('a'))
        else:
         result += char
    return result
# Example
a = 5
b = 8
text = "hello"
cipher = affine_encrypt(text, a, b)
plain = affine_decrypt(cipher, a, b)
print("Cipher:", cipher)
print("Decrypted:", plain)
```