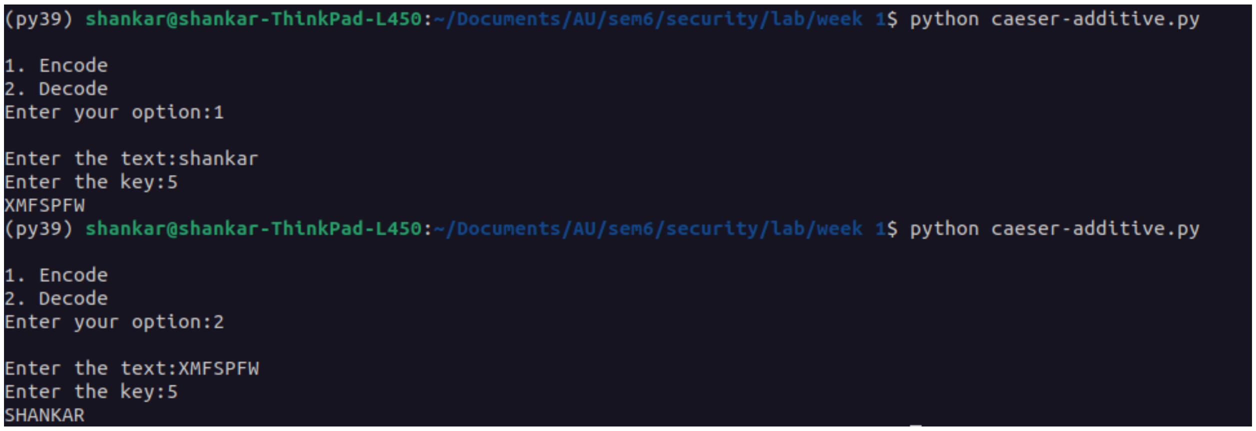
```python
def to_caesar(plain_text, key):
    cipher_text = ""
    for plain_char in plain_text:
        shift_char = ord(plain_char) + key - ord('A')
        cipher_char = chr((shift_char) % 26 + ord('A'))
        cipher_text += cipher_char

    return(cipher_text)


def from_caesar(cipher_text, key):
    plain_text = ""
    for cipher_char in cipher_text:
        shift_char = ord(cipher_char) - key%26
        if shift_char < ord('A'):
            shift_char = ord('Z') - (ord('A') - shift_char) +1
        plain_char = chr( shift_char )
        plain_text += plain_char
    return(plain_text)







option = input("\n1. Encode\n2. Decode\nEnter your option:")

inp = input("\nEnter the text:")
key = int(input("Enter the key:"))

input_text = inp.upper()

if option == '1':
    cipher = to_caesar(input_text,key)
    print(cipher)
else:
    plain  = from_caesar(input_text,key)
    print(plain)
```

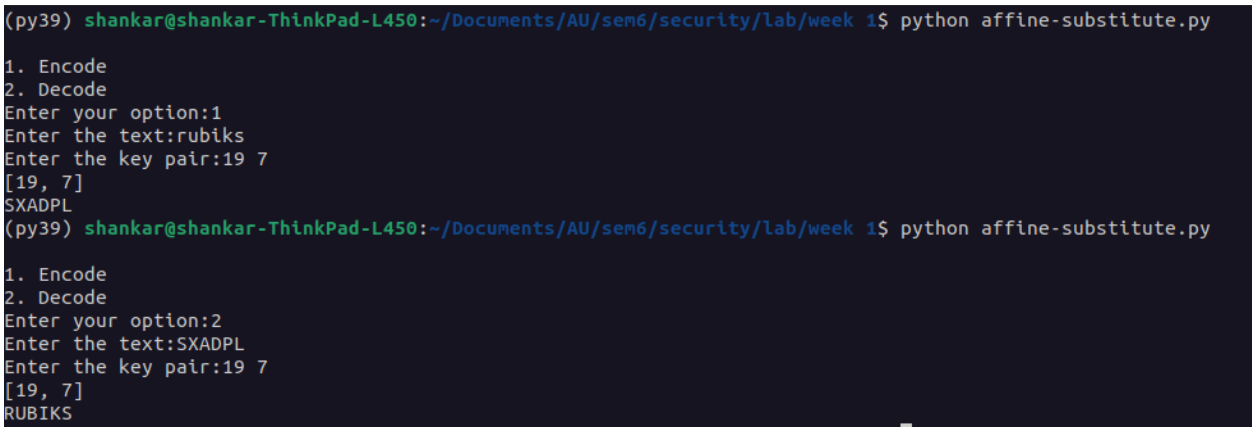


```
(py39) shankar@shankar-ThinkPad-L450:~/Documents/AU/sem6/security/lab/week 1$ python caeser-additive.py

1. Encode
2. Decode
Enter your option:1

Enter the text:shankar
Enter the key:5
XMFSPFW
(py39) shankar@shankar-ThinkPad-L450:~/Documents/AU/sem6/security/lab/week 1$ python caeser-additive.py

1. Encode
2. Decode
Enter your option:2

Enter the text:XMFSPFW
Enter the key:5
SHANKAR
```

2. Affine Cipher

```
#compute mod inverse
def modinv(a, m):
   mod_inv = pow(a, -1, m)

   return mod_inv

def enc(text, key):
   return ''.join([chr(((key[0]*(ord(t) - ord('A')) + key[1]) % 26) +
                        ord('A')) for t in text.upper().replace(' ', '')])

def dec(cipher, key):
   return ''.join([chr(((modinv(key[0], 26)*(ord(c) - ord('A') - key[1]))
                         % 26) + ord('A')) for c in
cipher.upper().replace(' ', '')])


option = input("\n1. Encode\n2. Decode\nEnter your option:")

text = input("Enter the text:")
key_raw  = (input("Enter the key pair:").split(" "))

key = [int(i) for i in key_raw]

print(key)
if option == '1':
   cipher = enc(text, key)
```

```
    print(cipher)
else:
    plain  = dec(text, key)
    print(plain)
```

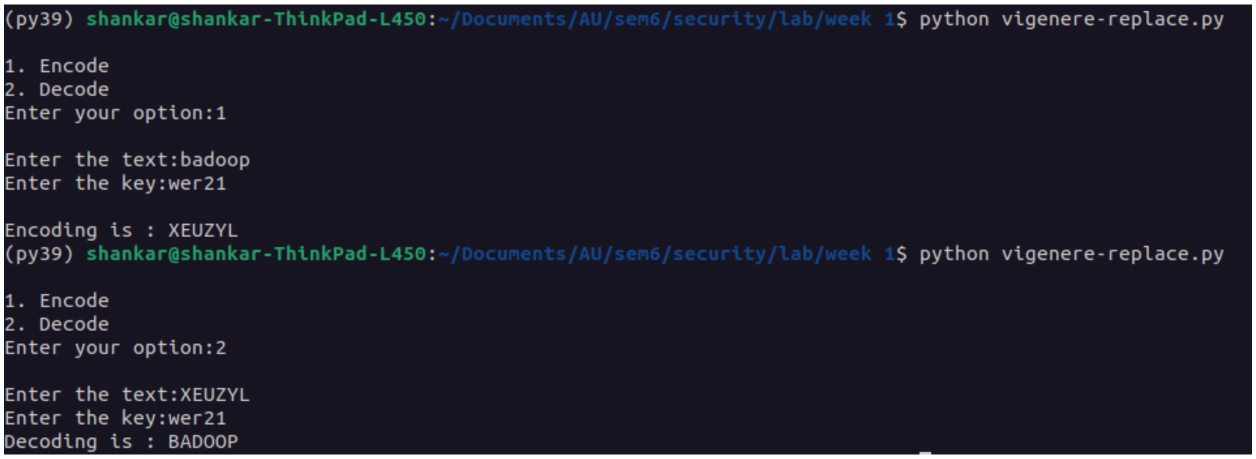


```
(py39) shankar@shankar-ThinkPad-L450:~/Documents/AU/sem6/security/lab/week 1$ python affine-substitute.py

1. Encode
2. Decode
Enter your option:1
Enter the text:rubiks
Enter the key pair:19 7
[19, 7]
SXADPL
(py39) shankar@shankar-ThinkPad-L450:~/Documents/AU/sem6/security/lab/week 1$ python affine-substitute.py

1. Encode
2. Decode
Enter your option:2
Enter the text:SXADPL
Enter the key pair:19 7
[19, 7]
RUBIKS
```

3. Vignere Cipher

```
'''
do the math on the plain and key and then remove the added up common
values
then handle the overflow of characters and put back the common value
'''
def enc_vig_char(plain_char, key):
    cipher_char = ""

    shift_char = ord(plain_char) + ord(key) - 2*ord('A')
    cipher_char = chr((shift_char) % 26 + ord('A'))

    return(cipher_char)

def dec_vig_char(cipher_char, key):
    plain_char = ""

    shift_char = ord(cipher_char) - ord(key)
    plain_char = chr((shift_char) % 26 + ord('A'))

    return(plain_char)
```

```
#following calls the encoder and decoders

def enc(plain_text, key):
   cipher_text = ""
   for i in range(len(plain_text)):
       cipher_text += enc_vig_char(plain_text[i], key[i])

   return cipher_text

def dec(cipher, key):
   plain_text = ""
   for i in range(len(cipher)):
       plain_text += dec_vig_char(cipher[i], key[i])
   return plain_text


#change length of key to match that of plain-text
def change_size_key(key, new_size):
   return (key * (new_size//len(key) + 1))[:new_size]






option = input("\n1. Encode\n2. Decode\nEnter your option:")

inp = input("\nEnter the text:").upper()
key = input("Enter the key:").upper()

key = change_size_key(key, len(inp))

if option == '1':
   cipher = enc(inp, key)
   print("\nEncoding is : " + cipher)
else:
   plain = dec(inp, key)
   print("Decoding is : " + plain)
```



```
(py39) shankar@shankar-ThinkPad-L450:~/Documents/AU/sem6/security/lab/week 1$ python vigenere-replace.py

1. Encode
2. Decode
Enter your option:1

Enter the text:badoop
Enter the key:wer21

Encoding is : XEUZYL
(py39) shankar@shankar-ThinkPad-L450:~/Documents/AU/sem6/security/lab/week 1$ python vigenere-replace.py

1. Encode
2. Decode
Enter your option:2

Enter the text:XEUZYL
Enter the key:wer21
Decoding is : BADOOP
```