NIS_LAB_2_Assignment

CE_055

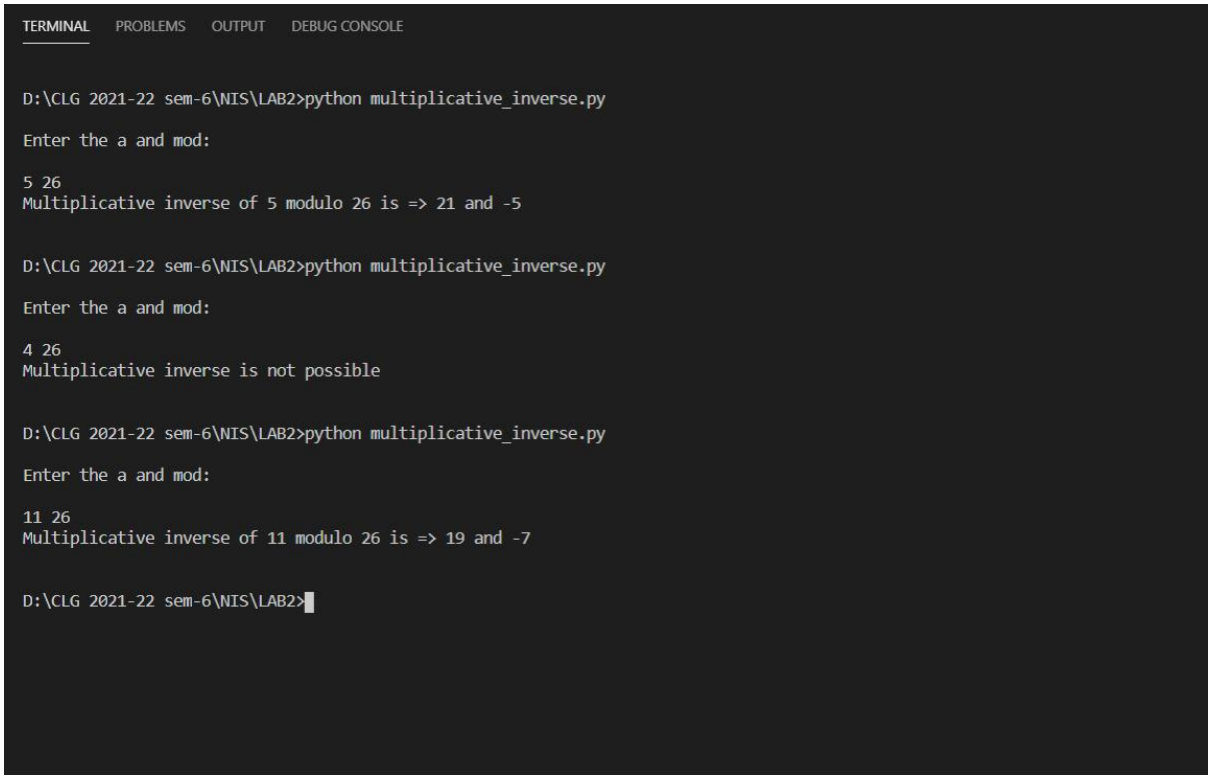
Aim :- Aim is to develop the code of Multiplicative inverse, Multiplicative Cipher and to combine previous both the code implement Affine Cipher.

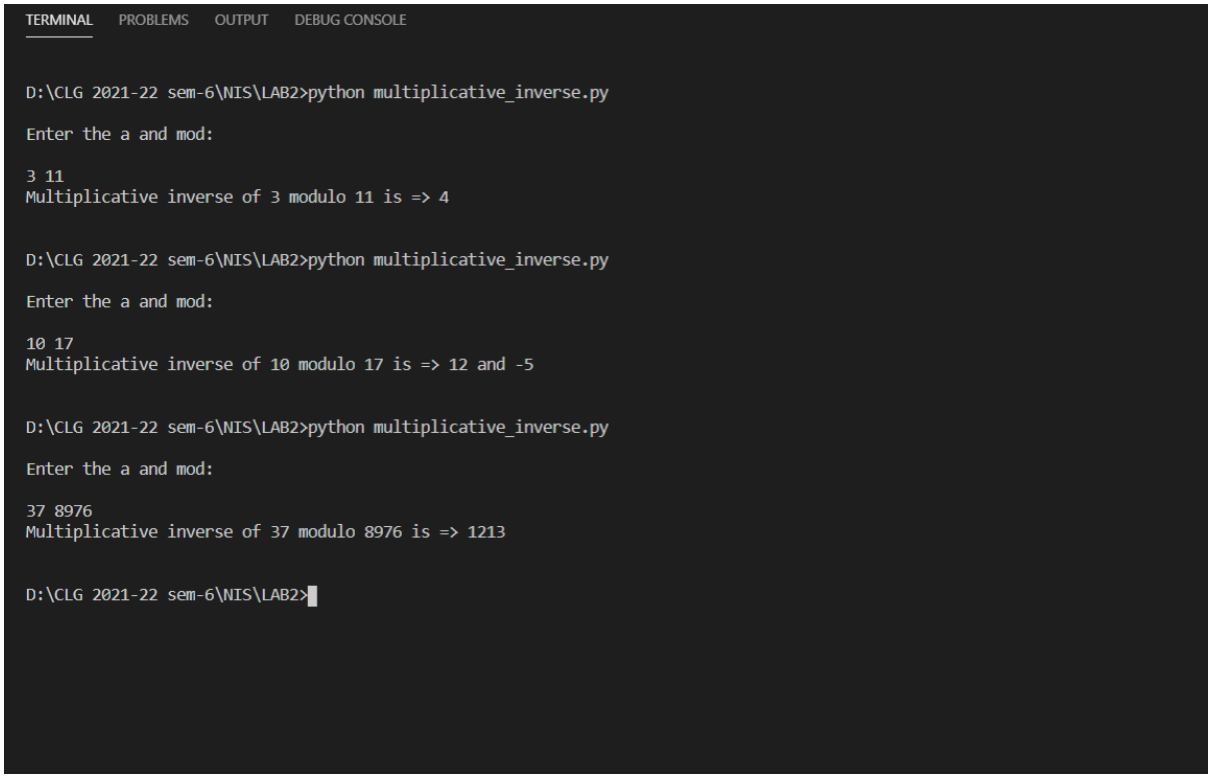
1. Implement the code to find multiplicative modular inverse.

Code :-

```
'''
Author : Dhruv B Kakadiya

'''
# Multiplicative Inverse code

# function to find multiplicative Inverse
def find_mul_inverse (a, n):
    t1, t2 = 0, 1
    while(a > 0):
        q = n // a
        r = n - (q * a)
        n, a = a, r
        t = t1 - (q * t2)
        t1, t2 = t2, t
    gcd, t = n, t1
    return gcd, t

if __name__ == "__main__":
    print("\nEnter the a and mod: \n")
    a, n = map(int, input().split())
    gcd, mul_inverse = find_mul_inverse(a, n)
    if (gcd == 1):
        if (mul_inverse < 0):
            print(f"Multiplicative inverse of {a} modulo {n} is => {mul_invers
e % n} and {mul_inverse}\n")
        else:
            print(f"Multiplicative inverse of {a} modulo {n} is => {mul_invers
e}\n")
    else:
        print(f"Multiplicative inverse is not possible\n")
```

Output :-


TERMINAL
PROBLEMS
OUTPUT
DEBUG CONSOLE
D:\CLG 2021-22 sem-6\NIS\LAB2>python multiplicative_inverse.py
Enter the a and mod:
5 26
Multiplicative inverse of 5 modulo 26 is => 21 and -5
D:\CLG 2021-22 sem-6\NIS\LAB2>python multiplicative_inverse.py
Enter the a and mod:
4 26
Multiplicative inverse is not possible
D:\CLG 2021-22 sem-6\NIS\LAB2>python multiplicative_inverse.py
Enter the a and mod:
11 26
Multiplicative inverse of 11 modulo 26 is => 19 and -7
D:\CLG 2021-22 sem-6\NIS\LAB2>


TERMINAL
PROBLEMS
OUTPUT
DEBUG CONSOLE
D:\CLG 2021-22 sem-6\NIS\LAB2>python multiplicative_inverse.py
Enter the a and mod:
3 11
Multiplicative inverse of 3 modulo 11 is => 4
D:\CLG 2021-22 sem-6\NIS\LAB2>python multiplicative_inverse.py
Enter the a and mod:
10 17
Multiplicative inverse of 10 modulo 17 is => 12 and -5
D:\CLG 2021-22 sem-6\NIS\LAB2>python multiplicative_inverse.py
Enter the a and mod:
37 8976
Multiplicative inverse of 37 modulo 8976 is => 1213
D:\CLG 2021-22 sem-6\NIS\LAB2>

2. Implement the code for encryption and decryption of Multiplicative cipher.

Code :-

```
'''
Author : Dhruv B Kakadiya

'''
# Multiplicative Cipher

# static variable
mod = 26

from multiplicative_inverse import find_mul_inverse

# function for encryption of plain_text
def mul_cipher_encryption (plain_text, key):
    encrypted_text = ""
    for letter in plain_text:
        if (letter.isupper()):
            encrypted_text += chr(((ord(letter) - 65) * key) % mod + 65)
        elif (letter.islower()):
            encrypted_text += chr(((ord(letter) - 97) * key) % mod + 97)
        else:
            encrypted_text += letter
    return encrypted_text

# function for decryption
def mul_cipher_decryption (encrypted_text, mul_inverse):
    decrypted_text = ""
    for letter in encrypted_text:
        if (letter.isupper()):
            decrypted_text += chr(((ord(letter) - 65) * mul_inverse) % mod + 6
5)
        elif (letter.islower()):
            decrypted_text += chr(((ord(letter) - 97) * mul_inverse) % mod + 9
7)
        else:
            decrypted_text += letter
    return decrypted_text

if __name__ == "__main__":
    plain_text = input("\nEnter the plain Text : ")
    key = int(input("\nEnter the Key : "))
    gcd, mul_inverse = find_mul_inverse(key, mod)
    if (gcd == 1):
        encrypted_text = mul_cipher_encryption(plain_text, key)
```

```
        print(f"\nThe encrypted text of '{plain_text}' is => '{encrypted_text}
'")
        if (mul_inverse < 0):
            mul_inverse = (mul_inverse) % mod
        decrypted_text = mul_cipher_decryption(encrypted_text, mul_inverse)
        print(f"\nThe decrypted text of '{encrypted_text}' is => '{decrypted_t
ext}'")
    else:
        print("\nNot a Valid key!")
```

Output :-

```
TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE

D:\CLG 2021-22 sem-6\NIS\LAB2>python multiplicative_cipher.py
Enter the plain Text :Hello World!
Enter the Key :5
The encrypted text of 'Hello World!' is => 'Judds Gshdp!'
The decrypted text of 'Judds Gshdp!' is => 'Hello World!'
D:\CLG 2021-22 sem-6\NIS\LAB2>python multiplicative_cipher.py
Enter the plain Text :Hello, Mars Here we come!
Enter the Key :15
The encrypted text of 'Hello, Mars Here we come!' is => 'Bijjc, Yavk Bivi si ecyi!'
The decrypted text of 'Bijjc, Yavk Bivi si ecyi!' is => 'Hello, Mars Here we come!'
D:\CLG 2021-22 sem-6\NIS\LAB2>
```

```
TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE

D:\CLG 2021-22 sem-6\NIS\LAB2>python multiplicative_cipher.py
Enter the plain Text : Alien: Hello Earth! We are from Mars.
Enter the Key : 19
The encrypted text of 'Alien: Hello Earth! We are from Mars.' is => 'Abwyn: Dybbg Yalxd! Cy aly rlgu Uale.'
The decrypted text of 'Abwyn: Dybbg Yalxd! Cy aly rlgu Uale.' is => 'Alien: Hello Earth! We are from Mars.'
D:\CLG 2021-22 sem-6\NIS\LAB2>
```

3. Implement the code of affine Cipher, It is a combination of additive cipher and multiplicative cipher.

Code :-

```
'''
Author : Dhruv B Kakadiya

'''
# Multiplicative Cipher
from multiplicative_inverse import find_mul_inverse
# static variable
mod = 26
# function for Encryption
def affine_cipher_encryption (plain_text, key1, key2):
    encrypted_text = ""
    for letter in plain_text:
        if (letter.isupper()):
            encrypted_text += chr(((ord(letter) - 65) * key1 + key2) % mod + 6
5)
        elif (letter.islower()):
            encrypted_text += chr(((ord(letter) - 97) * key1 + key2) % mod + 9
7)
        else:
            encrypted_text += letter
    return encrypted_text

# function for Decryption
def affine_cipher_decryption (encrypted_text, mul_inverse, key2):
    decrypted_text = ""
    for letter in encrypted_text:
        if (letter.isupper()):
            decrypted_text += chr(((ord(letter) - 65 - key2) * mul_inverse) %
mod + 65)
        elif (letter.islower()):
            decrypted_text += chr(((ord(letter) - 97 - key2) * mul_inverse) %
mod + 97)
        else:
            decrypted_text += letter
    return decrypted_text

if __name__ == "__main__":
    plain_text = input("\nEnter the plain Text : ")
    key1 = int(input("\nEnter the Key1 : "))
    key2 = int(input("\nEnter the Key2 : "))
    gcd, mul_inverse = find_mul_inverse(key1, mod)
    if (gcd == 1):
        encrypted_text = affine_cipher_encryption(plain_text, key1, key2)
```

```
        print(f"\nThe encrypted text of '{plain_text}' is => '{encrypted_text}
'")
        if (mul_inverse < 0):
            mul_inverse = (mul_inverse) % mod
        decrypted_text = affine_cipher_decryption(encrypted_text, mul_inverse,
 key2)
        print(f"\nThe decrypted text of '{encrypted_text}' is => '{decrypted_t
ext}'")
    else:
        print("\nNot a Valid key1!")
```

Output :-

```
TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE
D:\CLG 2021-22 sem-6\NIS\LAB2>python affine_cipher.py
Enter the plain Text : Hello World!
Enter the Key1 : 5
Enter the Key2 : 6
The encrypted text of 'Hello World!' is => 'Pajjy Mynjv!'
The decrypted text of 'Pajjy Mynjv!' is => 'Hello World!'
D:\CLG 2021-22 sem-6\NIS\LAB2>python affine_cipher.py
Enter the plain Text : Hi, Good Morning Mars.
Enter the Key1 : 7
Enter the Key2 : 14
The encrypted text of 'Hi, Good Morning Mars.' is => 'Ls, Eiij Uidbsbe Uodk.'
The decrypted text of 'Ls, Eiij Uidbsbe Uodk.' is => 'Hi, Good Morning Mars.'
D:\CLG 2021-22 sem-6\NIS\LAB2>python affine_cipher.py
Enter the plain Text : hi
Enter the Key1 : 5
Enter the Key2 : 8
The encrypted text of 'hi' is => 'rw'
The decrypted text of 'rw' is => 'hi'
D:\CLG 2021-22 sem-6\NIS\LAB2>
```

```
TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE

D:\CLG 2021-22 sem-6\NIS\LAB2>python affine_cipher.py
Enter the plain Text : We are from the great cluster Northen sky!
Enter the Key1 : 5
Enter the Key2 : 8
The encrypted text of 'We are from the great cluster Northen sky!' is => 'Oc ipc hpaq zrc mpciz sleuzcp Vapzrcv ugy!'
The decrypted text of 'Oc ipc hpaq zrc mpciz sleuzcp Vapzrcv ugy!' is => 'We are from the great cluster Northen sky!'
D:\CLG 2021-22 sem-6\NIS\LAB2>
```

Description :-

1. Multiplicative Modular Inverse:-
   - There are two integers ex. A and M if we want to find the modular multiplicative inverse of A with modulo M. assume the inverse is k.
   - Formula is a * k ~= 1 (mod M).
   - The multiplicative inverse of A modulo M is exists if and only if a and m are co-prime i.e The gcd of A and M is 1.
   - If the gcd is not equivalent to 1 then there is no inverse exists.

2. Multiplicative Cipher:-
   - This cipher is not strong enough, because in this cipher the formula is
     Encryption formula : - (P * key) mod 26.
     Decryption formula : - (C * key^-1) mod 26.
   - In the range of 1 to 26 there are only 12 number possible as a key, that's why it is easy for attacker to get into it and intercept between sender and receiver.
   - Here key must have multiplicative inverse under the modulo.

3. Affine Cipher:-
   - Affine cipher is type of monoalphabetic substitution cipher, also it is a combination of multiplicative cipher and additive cipher.
   - Encryption :
     It uses modular arithmetic to transform the integer that each plaintext letter corresponds to into another integer that correspond to a ciphertext letter.
     The function for Encryption : -
     ((P * key1) + key2) mod 26
   - In this formula key1 must be has multiplicative inverse under the mod. And key2 belongs to any 26 alphabet.
   - There are 12 * 26 combination possible for guess pair of key.
   - Decryption :
     In deciphering the ciphertext, we must perform the inverse functions on the ciphertext to retrieve the plaintext. The first step is to convert each of the ciphertext letters into their integer values.
     The function for Decryption: -
     ((C – key2) * key1 ^ -1) mod 26
   - This is also not strong enough.