D15B/37 **Mikil Lalwani** **Security Lab**

Experiment No 2

Aim-

Design and Implement a product cipher using Substitution ciphers and Transposition Cipher.

Theory-

The Caesar Cipher technique is one of the earliest and most straightforward methods of encryption technique. It’s simply a substitution cipher, i.e., each letter of a given text is replaced by a letter with a fixed number of positions down the alphabet. For example with a shift of 1, A would be replaced by B, B would become C, and so on. The method is apparently named after Julius Caesar, who apparently used it to communicate with his officials.
Thus to cipher a given text we need an integer value, known as a shift which indicates the number of positions each letter of the text has been moved down.
The encryption can be represented using modular arithmetic by first transforming the letters into numbers, according to the scheme, A = 0, B = 1,…, Z = 25.

Algorithm-

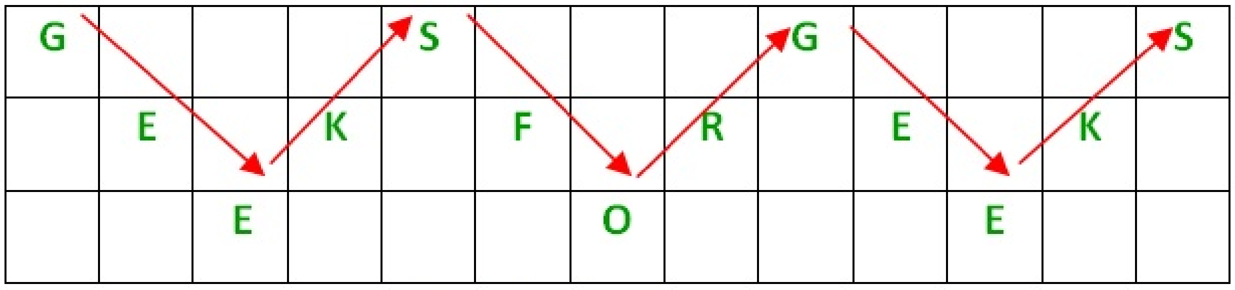
1. Caesar cipher-
   Input:
   a. A String of lower case letters, called Text.
   b. An Integer between 0-25 denotes the required shift.

   How to decrypt?
   We can either write another function decrypt similar to encrypt, that’ll apply the given shift in the opposite direction to decipher the original text. However, we can use the cyclic property of the cipher under modulo.

2. Rail Fence cipher-
   The rail fence cipher (also called a zigzag cipher) is a form of transposition cipher. It derives its name from the way in which it is encoded.

   Encryption

In a transposition cipher, the order of the alphabets is re-arranged to obtain the cipher text.
In the rail fence cipher, the plain text is written downwards and diagonally on the straight rails of an imaginary fence. When we reach the bottom rail, we traverse upwards moving diagonally, after reaching the top rail, the direction is changed again. Thus the alphabet of the message is written in a zig-zag manner. After each alphabet has been written, the individual rows are combined to obtain the ciphertext.

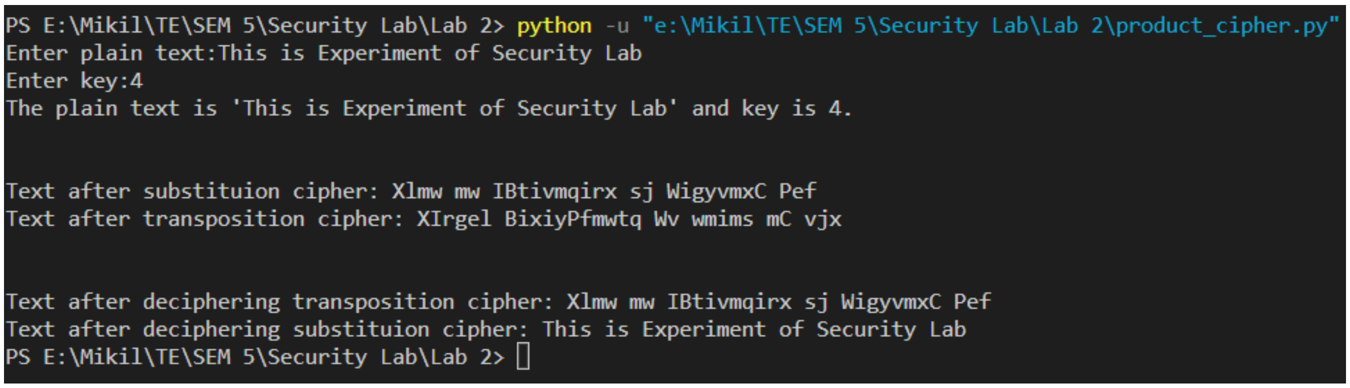


<u>Decryption</u>
As we've seen earlier, the number of columns in the rail fence cipher remains equal to the length of the plain-text message. And the key corresponds to the number of rails.
Hence, the rail matrix can be constructed accordingly. Once we've got the matrix we can figure out the spots where texts should be placed (using the same way of moving diagonally up and down alternatively). Then, we fill the cipher-text row-wise. After filling it, we traverse the matrix in a zig-zag manner to obtain the original text.

Code -

```
import string

def sub_en(plain,key):
    letters = string.ascii_letters
    dict1 = {}
    for i in range(0,len(letters)):
        dict1[letters[i]]=letters[(i+key)%len(letters)]
    c_cipher = ''
    for i in range(0,len(plain)):
        if(plain[i]==' '):
            c_cipher = c_cipher + ' '
        else:
            if(plain[i] in dict1):
                c_cipher = c_cipher + dict1[plain[i]]
            else:
                c_cipher = c_cipher + plain[i]

    return c_cipher

def sub_den(decrypt_plain1,key):
    letters = string.ascii_letters
    dict2 = {}
    for i in range(0,len(letters)):
        dict2[letters[i]]=letters[(i-key)%len(letters)]

    plain = ''
    for i in range(0,len(decrypt_plain1)):
        if(decrypt_plain1[i]==' '):
            plain = plain + ' '
        else:
            if(decrypt_plain1[i] in dict2):
                plain = plain + dict2[decrypt_plain1[i]]
            else:
                plain = plain + decrypt_plain1[i]

    return plain

def trans_en(c_cipher,key):
    row = 0
    col = 0
    array = [['' for i in range(len(c_cipher))]for j in range(key+1)]
    down = False
    i = 0
    for i in range(len(c_cipher)):
        array[row][i] = c_cipher[i]
        if row == key:
            down = False
        if row ==0 or down == True:
            row = row + 1
            down = True
        else:
            down = False
            row = row - 1

    t_cipher = ''
    for i in array:
        for j in i:
            t_cipher = t_cipher + j

    return t_cipher


def trans_den(t_cipher,key):

    array = [['' for i in range(len(t_cipher))]for j in range(key+1)]
    row = 0
    down = False

    i = 0
    for i in range(len(t_cipher)):
        array[row][i] = '*'
        if row == key:
            down = False
        if row ==0 or down == True:
            row = row + 1
            down = True

        else:
            down = False
            row = row - 1

    col = 0
    n = 0
    for i in range(key+1):
        for j in range(len(t_cipher)):
            if (array[i][j] == '*' and n < len(t_cipher)):
                array[i][j] = t_cipher[n]
                n = n + 1

    plain = ''
    row = 0
    for i in range(len(t_cipher)):
        plain = plain + array[row][i]
        if row == key:
            down = False
        if row ==0 or down == True:
            row = row + 1
            down = True

        else:
            down = False
            row = row - 1

    return plain



plain = input('Enter plain text:')
key = int(input('Enter key:'))

print(f"The plain text is '{plain}' and key is {key}.")

print('\n')

c_cipher = sub_en(plain,key)
t_cipher = trans_en(c_cipher, key)

print("Text after substituion cipher: "+c_cipher)
print("Text after transposition cipher: "+t_cipher)

print('\n')

decrypt_plain1 = trans_den(t_cipher, key)
decrypt_plain2 = sub_den(decrypt_plain1, key)

print("Text after deciphering transposition cipher: "+decrypt_plain1)
print("Text after deciphering substituion cipher: "+decrypt_plain2)
```

Output -



```
PS E:\Mikil\TE\SEM 5\Security Lab\Lab 2> python -u "e:\Mikil\TE\SEM 5\Security Lab\Lab 2\product_cipher.py"
Enter plain text:This is Experiment of Security Lab
Enter key:4
The plain text is 'This is Experiment of Security Lab' and key is 4.


Text after substituion cipher: Xlmw mw IBtivmqirx sj WigyvmxC Pef
Text after transposition cipher: XIrgel BixiyPfmwtq Wv wmims mC vjx


Text after deciphering transposition cipher: Xlmw mw IBtivmqirx sj WigyvmxC Pef
Text after deciphering substituion cipher: This is Experiment of Security Lab
PS E:\Mikil\TE\SEM 5\Security Lab\Lab 2> 
```

Conclusion-

Thus we have successfully implemented the product cipher.