Name: Madhuri Ramakrishnan
119A3032
TE IT (E2)

EXPERIMENT 3

PLAYFAIR AND VIGENÈRE CIPHER.

Aim: - Cryptanalysis or decoding Playfair, Vigenère cipher.

Resources Required: -

1. VS code IDE
2. Intel I3 processor
3. Programming Language - Python

Procedure: -

- Vigenère Cipher:

1. Accept Plain text and keyword and store them in variables
2. Pass keyword and Plain text to generate compare text function where a string will be created by repeating the keyword till its length matches the length of Plain text
3. Accept the string in a variable which is returned by generateCompareText function
4. Pass this String and Plain text to encrypt function where Cypher text will be calculated by the following formula-

EncryptedText[i] = (PlainText[i] + CompareString[i]) % 26

5. Store the value returned from this function and display it on the screen
6. Pass the Cipher text and the Compare String to decrypt function where original text will be decrypted by using following formula-

DecryptedText[i] = (Plaintext [i] - CompareString [i]) % 26

7. 7.Store the value returned from decrypt function and display it on the screen

- Playfair Cipher:

1. When the two letters are in different rows and columns, each is replaced by the letter that is in the same row but in the other column; i.e., to encrypt WE, W is replaced by U and E by G.
2. When A and R are in the same row, A is encrypted as R and R (reading the row cyclically) as M.
3. When I and S are in the same column, I is encrypted as S and S as X.
4. When a double letter occurs, a spurious symbol, say Q, is introduced so that the MM in SUMMER is encrypted as NL for MQ and CL for ME.
5. An X is appended to the end of the plaintext if necessary to give the plaintext an even number of letters.

Theory:

1. *Playfair Cipher:*
The Playfair cipher was the first practical digraph substitution cipher. The scheme was invented in 1854 by Charles Wheatstone but was named after Lord Playfair who promoted the use of the cipher. In Playfair cipher unlike traditional cipher we encrypt a pair of alphabets(digraphs) instead of a single alphabet.
It was used for tactical purposes by British forces in the Second Boer War and in World War I and for the same purpose by the Australians during World War II. This was because Playfair is reasonably fast to use and requires no special equipment.

2. *Vigenère Cipher:*
Vigenère Cipher is a method of encrypting alphabetic text. It uses a simple form of polyalphabetic substitution. A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets .The encryption of the original text is done using the *Vigenère square or Vigenère table*.

- The table consists of the alphabets written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar Ciphers.
- At different points in the encryption process, the cipher uses a different alphabet from one of the rows.

- The alphabet used at each point depends on a repeating keyword.

Code:

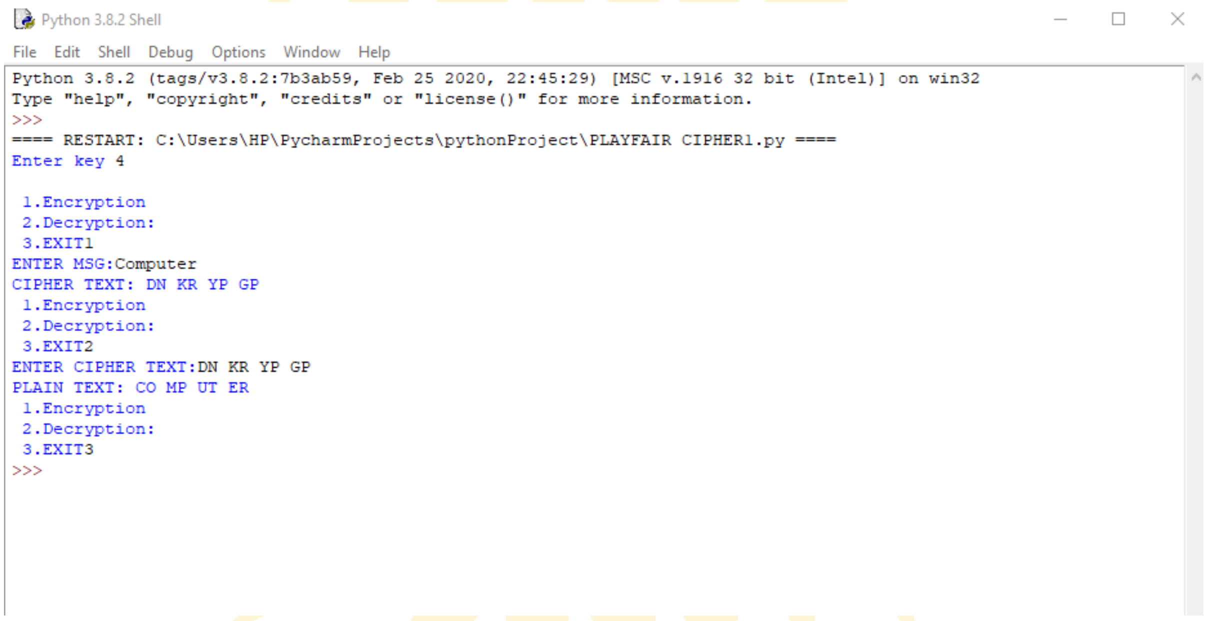
- Playfair cipher:

```
key=input("Enter key")
key=key.replace(" ", "")
key=key.upper()
def matrix(x,y,initial):
    return [[initial for i in range(x)] for j in range(y)]

result=list()
for c in key: #storing key
    if c not in result:
        if c=='J':
            result.append('I')
        else:
            result.append(c)
flag=0
for i in range(65,91): #storing other character
    if chr(i) not in result:
        if i==73 and chr(74) not in result:
            result.append("I")
            flag=1
        elif flag==0 and i==73 or i==74:
            pass
        else:
            result.append(chr(i))
k=0
my_matrix=matrix(5,5,0) #initialize matrix
for i in range(0,5): #making matrix
    for j in range(0,5):
        my_matrix[i][j]=result[k]
        k+=1

def locindex(c): #get location of each character
    loc=list()
    if c=='J':
        c='I'
    for i ,j in enumerate(my_matrix):
        for k,l in enumerate(j):
            if c==l:
                loc.append(i)
                loc.append(k)
        return loc

def encrypt():  #Encryption
    msg=str(input("ENTER MSG:"))
    msg=msg.upper()
    msg=msg.replace(" ", "")
    i=0
    for s in range(0,len(msg)+1,2):
        if s<len(msg)-1:
            if msg[s]==msg[s+1]:
                msg=msg[:s+1]+'X'+msg[s+1:]
    if len(msg)%2!=0:
        msg=msg[:]+'X'
    print("CIPHER TEXT:",end=' ')
    while i<len(msg):
        loc=list()
        loc=locindex(msg[i])
        loc1=list()
        loc1=locindex(msg[i+1])
        if loc[1]==loc1[1]:

print("{}{}".format(my_matrix[(loc[0]+1)%5][loc[1]],my_matrix[(loc1[0]+1)%5][lo
c1[1]]),end=' ')
        elif loc[0]==loc1[0]:

print("{}{}".format(my_matrix[loc[0]][(loc[1]+1)%5],my_matrix[loc1[0]][(loc1[1]+
1)%5]),end=' ')
        else:

print("{}{}".format(my_matrix[loc[0]][loc1[1]],my_matrix[loc1[0]][loc[1]]),end=' ')
        i=i+2

def decrypt():  #decryption
    msg=str(input("ENTER CIPHER TEXT:"))
    msg=msg.upper()
    msg=msg.replace(" ", "")
    print("PLAIN TEXT:",end=' ')
    i=0
    while i<len(msg):
        loc=list()
        loc=locindex(msg[i])
        loc1=list()
        loc1=locindex(msg[i+1])
        if loc[1]==loc1[1]:
            print("{}{}".format(my_matrix[(loc[0]-1)%5][loc[1]],my_matrix[(loc1[0]-
1)%5][loc1[1]]),end=' ')
        elif loc[0]==loc1[0]:
            print("{}{}".format(my_matrix[loc[0]][(loc[1]-
1)%5],my_matrix[loc1[0]][(loc1[1]-1)%5]),end=' ')
        else:

print("{}{}".format(my_matrix[loc[0]][loc1[1]],my_matrix[loc1[0]][loc[1]]),end=' ')
        i=i+2

while(1):
    choice=int(input("\n 1.Encryption \n 2.Decryption: \n 3.EXIT"))
    if choice==1:
        encrypt()
    elif choice==2:
        decrypt()
    elif choice==3:
        exit()
    else:
        print("Choose correct choice")
```

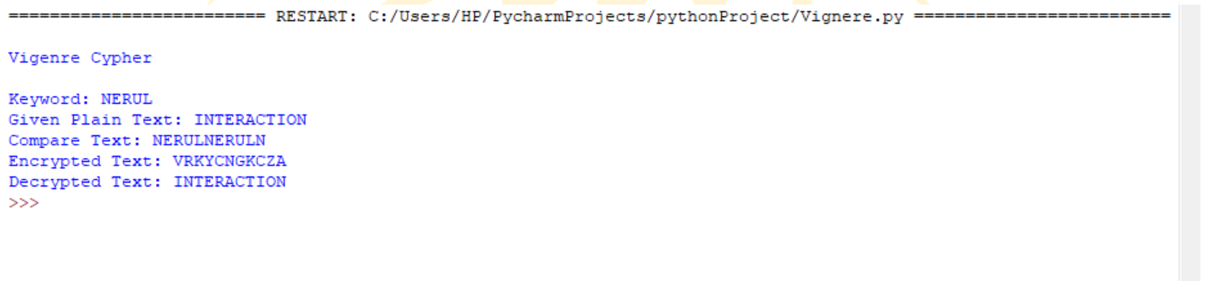
- Vigenère Cipher

```
#Vigenre Cipher

keyword = 'NERUL'
plainText = 'INTERACTION'

def generateCompareText(keyword,plainText):
    txt = list(keyword) #String to List
    for i in range(len(plainText) - len(txt)):
        txt.append(txt[i %len(txt)])
    return("".join(txt))  #List to String

def encryptFunction(plainText,compareStr):
    cipherText = []
    for i in range(len(plainText)):
        char = (ord(plainText[i]) + ord(compareStr[i])) % 26
        char = char + 65
        cipherText.append(chr(char))
    return("".join(cipherText))

def decryptFunction(Cypher,compareStr):
    og = []
    for i in range(len(Cypher)):
        char = (ord(Cypher[i]) - ord(compareStr[i]) + 26) % 26
        char = char + 65
        og.append(chr(char))
    return("".join(og))

print("\nVigenre Cypher\n")
print("Keyword:",keyword)
print("Given Plain Text:",plainText)
compareStr = generateCompareText(keyword,plainText)
print("Compare Text:",compareStr)
Cypher = encryptFunction(plainText,compareStr)
print("Encrypted Text:",Cypher)
Decypher = decryptFunction(Cypher,compareStr)
print("Decrypted Text:",Decypher)
```

Results:



```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\HP\PycharmProjects\pythonProject\PLAYFAIR CIPHER1.py ====
Enter key 4

 1.Encryption
 2.Decryption:
 3.EXIT1
ENTER MSG:Computer
CIPHER TEXT: DN KR YP GP
 1.Encryption
 2.Decryption:
 3.EXIT2
ENTER CIPHER TEXT:DN KR YP GP
PLAIN TEXT: CO MP UT ER
 1.Encryption
 2.Decryption:
 3.EXIT3
>>>
```



```
======================== RESTART: C:/Users/HP/PycharmProjects/pythonProject/Vignere.py ========================

Vigenre Cypher

Keyword: NERUL
Given Plain Text: INTERACTION
Compare Text: NERULNERULN
Encrypted Text: VRKYCNGKCZA
Decrypted Text: INTERACTION
>>>
```

Conclusion: Hence, we have successfully completed the implementation of Vigenère cipher and Playfair cipher.