| Course Code: CSL604 | Course Name: CSS LAB |
|---|---|
| Class: TE-CO | Batch: 3 |
| Roll no: 18CO63 | Name: SHAIKH TAUSEEF MUSHTAQUE ALI |

## ASSIGNMENT: 01

**Aim:** Design own Cryptographic Algorithm

**Description:**

This Cryptographic Algorithm is a combination of two separate ciphers i.e. 'Playfair Cipher' & 'Vigenere Cipher' were the user has to enter a Key Value (of char data type) and can select either to Encrypt or Decrypt a suitable message of ones.

**Code:**

```
key=input("\nEnter key: ")
key=key.replace(" ", "")
key=key.upper()


def generateKey(string, key):
    key = list(key)
    if len(string) == len(key):
        return(key)
    else:
        for i in range(len(string) -
                       len(key)):
            key.append(key[i % len(key)])
    return("" . join(key))


# This function returns the
# encrypted text generated
# with the help of the key
def cipherText(string, key):
    cipher_text = []
    for i in range(len(string)):
```

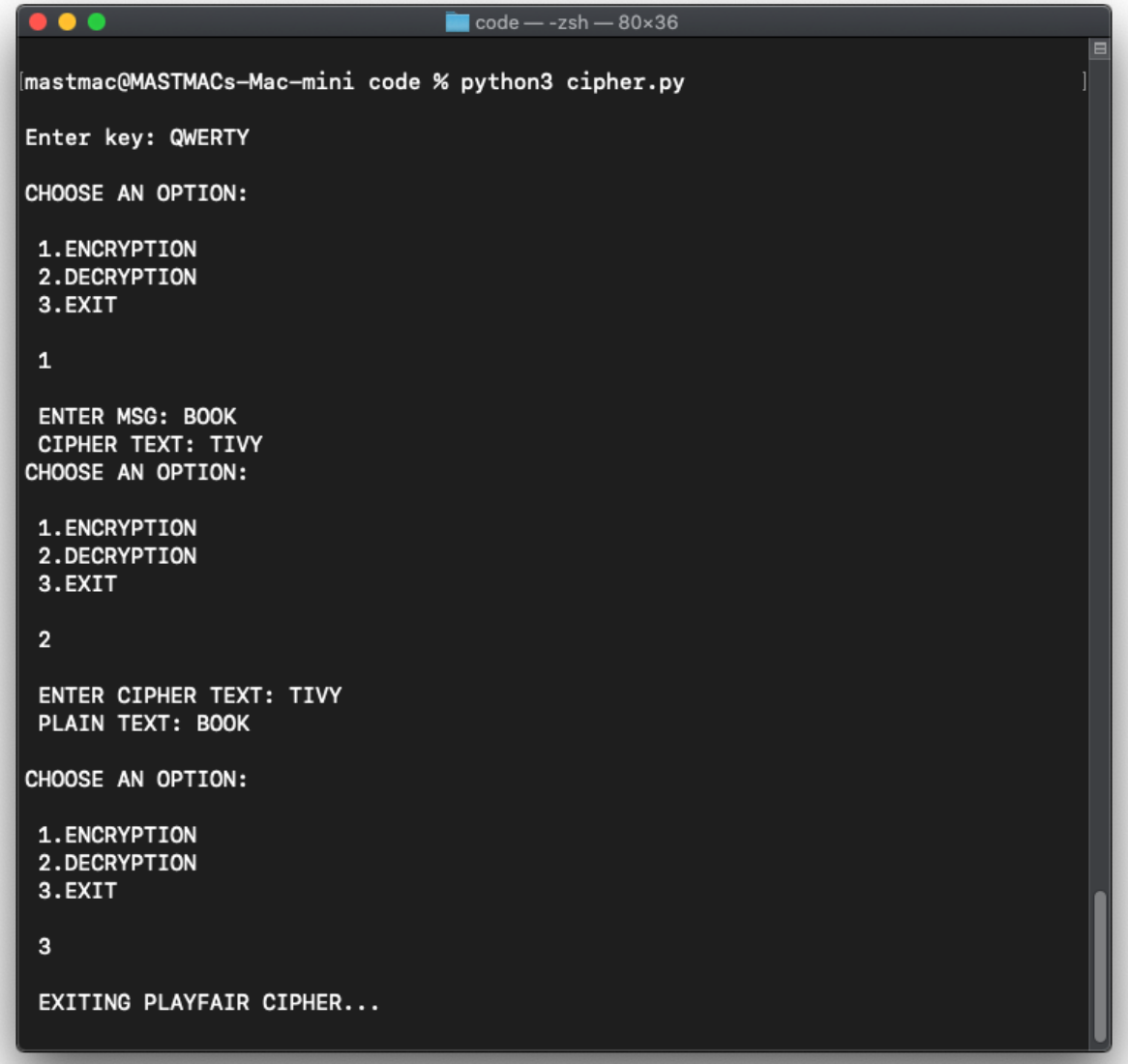
```
            x = (ord(string[i]) +
                    ord(key[i])) % 26
            x += ord('A')
            cipher_text.append(chr(x))
    return("" . join(cipher_text))

# This function decrypts the
# encrypted text and returns
# the original text
def originalText(cipher_text, key):
    orig_text = []
    for i in range(len(cipher_text)):
            x = (ord(cipher_text[i]) -
                    ord(key[i]) + 26) % 26
            x += ord('A')
            orig_text.append(chr(x))
    return("" . join(orig_text))



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("\n ENTER MSG: "))
    msg=msg.upper()
    msg=msg.replace(" ", "")
    keyword = generateKey(msg, key)
    msg = cipherText(msg,keyword)
    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][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


def decrypt():    #decryption
    msg=str(input("\n ENTER CIPHER TEXT: "))
    msg=msg.upper()
    msg=msg.replace(" ", "")
    keyword = generateKey(msg, key)
    print(" PLAIN TEXT:",end=' ')
    i=0
    text = ''
    while i<len(msg):
        loc=list()
```

```
            loc=locindex(msg[i])
            loc1=list()
            loc1=locindex(msg[i+1])
            if loc[1]==loc1[1]:
                text +=
"{}{}".format(my_matrix[(loc[0]-1)%5][loc[1]],my_matrix[(loc1[0]-1)%5][loc1[1]])
            elif loc[0]==loc1[0]:
                text +=
"{}{}".format(my_matrix[loc[0]][(loc[1]-1)%5],my_matrix[loc1[0]][(loc1[1]-1)%5])
            else:
                text += "{}{}".format(my_matrix[loc[0]][loc1[1]],my_matrix[loc1[0]][loc[1]])
            i=i+2
        print(originalText(text, keyword))


while(1):
    print("\nCHOOSE AN OPTION: \n")
    choice=int(input(" 1.ENCRYPTION \n 2.DECRYPTION \n 3.EXIT \n\n" + " "))
    if choice==1:
        encrypt()
    elif choice==2:
        decrypt()
    elif choice==3:
        print("\n EXITING PLAYFAIR CIPHER... \n")
        exit()
    else:
        print("\nINVALID OPTION! CHOOSE CORRECT OPTION \n")
```

**Output:**


code — -zsh — 80×36
mastmac@MASTMACs-Mac-mini code % python3 cipher.py
Enter key: QWERTY
CHOOSE AN OPTION:
1.ENCRYPTION
2.DECRYPTION
3.EXIT
1
ENTER MSG: BOOK
CIPHER TEXT: TIVY
CHOOSE AN OPTION:
1.ENCRYPTION
2.DECRYPTION
3.EXIT
2
ENTER CIPHER TEXT: TIVY
PLAIN TEXT: BOOK
CHOOSE AN OPTION:
1.ENCRYPTION
2.DECRYPTION
3.EXIT
3
EXITING PLAYFAIR CIPHER...