EXP NO: 1b
DATE: 03/02/2024

PLAYFAIR CIPHER

AIM:

To write a C Program to perform Playfair Cipher for both encryption and decryption process

ALGORITHM:

Step 1 : Start.

Step 2 : Include necessary header files.

Step 3 : Create a function for encryption process.

Step 4 : Define encryption function.

Step 5 : Create a function for decryption process.

Step 6 : Define decryption function.

Step 7 : Call both encryption and decryption function inside main function.

Step 8 : End.

```
PROGRAM:
def toLowerCase(text):
        return text.lower()


# Function to remove all spaces in a string


def removeSpaces(text):
        newText = ""
        for i in text:
                if i == " ":
                        continue
                else:
                        newText = newText + i
        return newText


# Function to group 2 elements of a string
# as a list element


def Diagraph(text):
        Diagraph = []
        group = 0
        for i in range(2, len(text), 2):
                Diagraph.append(text[group:i])

                group = i
        Diagraph.append(text[group:])
        return Diagraph


# Function to fill a letter in a string element
# If 2 letters in the same string matches


def FillerLetter(text):
        k = len(text)
        if k % 2 == 0:
```

```python
            for i in range(0, k, 2):
                    if text[i] == text[i+1]:
                            new_word = text[0:i+1] + str('x') + text[i+1:]
                            new_word = FillerLetter(new_word)
                            break
                    else:
                            new_word = text
        else:
                for i in range(0, k-1, 2):
                    if text[i] == text[i+1]:
                            new_word = text[0:i+1] + str('x') + text[i+1:]
                            new_word = FillerLetter(new_word)
                            break
                    else:
                            new_word = text
        return new_word


list1 = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'k', 'l', 'm',
                'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']

# Function to generate the 5x5 key square matrix


def generateKeyTable(word, list1):
        key_letters = []
        for i in word:
                if i not in key_letters:
                        key_letters.append(i)

        compElements = []
        for i in key_letters:
                if i not in compElements:
                        compElements.append(i)
        for i in list1:
                if i not in compElements:
                        compElements.append(i)
```

```python
        matrix = []
        while compElements != []:
                matrix.append(compElements[:5])
                compElements = compElements[5:]

        return matrix



def search(mat, element):
        for i in range(5):
                for j in range(5):
                        if(mat[i][j] == element):
                                return i, j



def encrypt_RowRule(matr, e1r, e1c, e2r, e2c):
        char1 = ''
        if e1c == 4:
                char1 = matr[e1r][0]
        else:
                char1 = matr[e1r][e1c+1]

        char2 = ''
        if e2c == 4:
                char2 = matr[e2r][0]
        else:
                char2 = matr[e2r][e2c+1]

        return char1, char2



def encrypt_ColumnRule(matr, e1r, e1c, e2r, e2c):
        char1 = ''
        if e1r == 4:
                char1 = matr[0][e1c]
        else:
```

```python
            char1 = matr[e1r+1][e1c]

        char2 = ''
        if e2r == 4:
            char2 = matr[0][e2c]
        else:
            char2 = matr[e2r+1][e2c]

        return char1, char2
def encrypt_RectangleRule(matr, e1r, e1c, e2r, e2c):
        char1 = ''
        char1 = matr[e1r][e2c]

        char2 = ''
        char2 = matr[e2r][e1c]

        return char1, char2
def encryptByPlayfairCipher(Matrix, plainList):
        CipherText = []
        for i in range(0, len(plainList)):
            c1 = 0
            c2 = 0
            ele1_x, ele1_y = search(Matrix, plainList[i][0])
            ele2_x, ele2_y = search(Matrix, plainList[i][1])

            if ele1_x == ele2_x:
                c1, c2 = encrypt_RowRule(Matrix, ele1_x, ele1_y, ele2_x, ele2_y)
                # Get 2 letter cipherText
            elif ele1_y == ele2_y:
                c1, c2 = encrypt_ColumnRule(Matrix, ele1_x, ele1_y, ele2_x,
ele2_y)
            else:
                c1, c2 = encrypt_RectangleRule(
                    Matrix, ele1_x, ele1_y, ele2_x, ele2_y)

            cipher = c1 + c2
            CipherText.append(cipher)
```
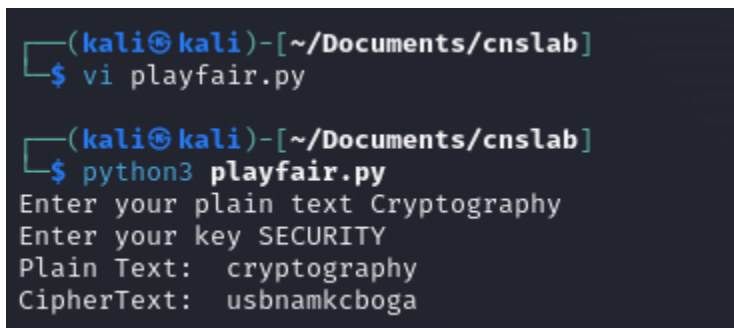
```
        return CipherText
text_Plain = input("Enter your plain text ")
text_Plain = removeSpaces(toLowerCase(text_Plain))
PlainTextList = Diagraph(FillerLetter(text_Plain))
if len(PlainTextList[-1]) != 2:
        PlainTextList[-1] = PlainTextList[-1]+'z'
key = input("Enter your key ")
key = toLowerCase(key)
Matrix = generateKeyTable(key, list1)

print("Plain Text: ", text_Plain)
CipherList = encryptByPlayfairCipher(Matrix, PlainTextList)

CipherText = ""
for i in CipherList:
        CipherText += i
print("CipherText: ", CipherText)
```

OUTPUT:

```
┌──(kali㉿kali)-[~/Documents/cnslab]
└─$ vi playfair.py

┌──(kali㉿kali)-[~/Documents/cnslab]
└─$ python3 playfair.py
Enter your plain text Cryptography
Enter your key SECURITY
Plain Text:  cryptography
CipherText:  usbnamkcboga
```

RESULT:
        Thus a python program has been implemented to demonstrate Playfair Cipher.