Roll. No. A016	Name: Varun Khadayate
Class B.Tech CsBs	Batch: 1
Date of Experiment: 06-08-2021	Subject: Cryptology

Theory

The best-known multiple-letter encryption cipher is the Playfair, which treats diagrams in the plaintext as single units and translates these units into ciphertext digrams.
The Playfair algorithm is based on the use of a 5 * 5 matrix of letters con- structed using a keyword. Here is an example, solved by Lord Peter Wimsey in Dorothy Sayers's *Have His Carcase*

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

In this case, the keyword is *monarchy*. The matrix is constructed by filling in the letters of the keyword (minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetic order. The letters I and J count as one letter. Plaintext is encrypted twoletters at a time, according to the following rules:

1. Repeating plaintext letters that are in the same pair are separated with a fillerletter, such as x, so that balloon would be treated as ba lx lo on.
2. Two plaintext letters that fall in the same row of the matrix are each replacedby the letter to the right, with the first element of the row circularly following the last. For example, ar is encrypted as RM.
3. Two plaintext letters that fall in the same column are each replaced by the letter beneath, with the top element of the column circularly following the last. For example, mu is encrypted as CM.
4. Otherwise, each plaintext letter in a pair is replaced by the letter that lies in its own row and the column occupied by the other plaintext letter. Thus, hs becomes BP and ea becomes IM (or JM, as the encipherer wishes).

The Playfair cipher is a great advance over simple monoalphabetic ciphers.For one thing, whereas there are only 26 letters, there are 26 * 26 = 676 digrams, so that identification of individual digrams is more difficult. Furthermore, the relative frequencies of individual letters exhibit a much greater range than that of digrams, making frequency analysis much more difficult. For these reasons, the Playfair cipher was for a long time considered unbreakable. It was used as the standard field system by the British Army in World War I and still enjoyed considerable use by the U.S. Army and other Allied forces during World War II.

Despite this level of confidence in its security, the Playfair cipher is relatively easy to break, because it still leaves much of the structure of the plaintext language intact. A few hundred letters of ciphertext are generally sufficient.

Code

```
import string

while True:
    ch = int(input('Welcome to Playfair Cipher Encryption and Decryption
Program Made by Varun Khadayate..\n [*] Press 1 for Encryption \n [*] Press 2
for Decryption \n [*] Press 0 to exit..\n \nYour Choice:: '))

    if ch == 1:
        print("\n=========================================")
        print("              !!!!Encryption!!!!             ")
        def key_generation(key):
            main=string.ascii_lowercase.replace('j','.')
            key=key.lower()

            key_matrix=['' for i in range(5)]
            i=0;j=0
            for c in key:
                if c in main:
                    key_matrix[i]+=c
                    main=main.replace(c,'.')
                    j+=1
                    if(j>4):
                        i+=1
                        j=0
            for c in main:
                if c!='.':
                    key_matrix[i]+=c

                    j+=1
                    if j>4:
                        i+=1
                        j=0
            return(key_matrix)
        def conversion(plain_text):
            plain_text_pairs=[]
            cipher_text_pairs=[]
            plain_text=plain_text.replace(" ","")
            plain_text=plain_text.lower()
            i=0
            while i<len(plain_text):
                a=plain_text[i]
                b=''

                if((i+1)==len(plain_text)):
                    b='x'
                else:
                    b=plain_text[i+1]
```

```
        if(a!=b):
            plain_text_pairs.append(a+b)
            i+=2
        else:
            plain_text_pairs.append(a+'x')
            i+=1

    print("plain text pairs: ",plain_text_pairs)


    for pair in plain_text_pairs:
        flag=False
        for row in key_matrix:
            if(pair[0] in row and pair[1] in row):
                j0=row.find(pair[0])
                j1=row.find(pair[1])
                cipher_text_pair=row[(j0+1)%5]+row[(j1+1)%5]
                cipher_text_pairs.append(cipher_text_pair)
                flag=True
        if flag:
            continue
        for j in range(5):
            col="".join([key_matrix[i][j] for i in range(5)])
            if(pair[0] in col and pair[1] in col):
                i0=col.find(pair[0])
                i1=col.find(pair[1])
                cipher_text_pair=col[(i0+1)%5]+col[(i1+1)%5]
                cipher_text_pairs.append(cipher_text_pair)
                flag=True
        if flag:
            continue
        i0=0
        i1=0
        j0=0
        j1=0

        for i in range(5):
            row=key_matrix[i]
            if(pair[0] in row):
                i0=i
                j0=row.find(pair[0])
            if(pair[1] in row):
                i1=i
                j1=row.find(pair[1])
        cipher_text_pair=key_matrix[i0][j1]+key_matrix[i1][j0]
        cipher_text_pairs.append(cipher_text_pair)
    print("         !!!Encrypted Successfully!!!         ")
    print("cipher text pairs: ",cipher_text_pairs)
```

```
            print('plain text: ',plain_text)
            print('cipher text: ',"".join(cipher_text_pairs))
            print("\n=========================================\n\n")


        key=input("Enter the key: ")

        key_matrix=key_generation(key)
        print("Key Matrix for encryption:")
        print(key_matrix)
        plain_text=input("Enter the message: ")
        conversion(plain_text)

    elif ch == 2:
        print("\n=========================================")
        print("               !!!Decryption!!!              ")
        def key_generation(key):
            main=string.ascii_lowercase.replace('j','.')
            key=key.lower()
            key_matrix=['' for i in range(5)]
            i=0;j=0
            for c in key:
                if c in main:
                    key_matrix[i]+=c
                    main=main.replace(c,'.')
                    j+=1
                    if(j>4):
                        i+=1
                        j=0
            for c in main:
                if c!='.':
                    key_matrix[i]+=c

                    j+=1
                    if j>4:
                        i+=1
                        j=0

            return(key_matrix)

        def conversion(cipher_text):
            plain_text_pairs=[]
            cipher_text_pairs=[]
            cipiher_text=cipher_text.lower()
            i=0
            while i<len(cipher_text):
                a=cipher_text[i]
                b=cipher_text[i+1]
                cipher_text_pairs.append(a+b)
```

```
            i+=2
        print("cipher text pairs: ",cipher_text_pairs)

        for pair in cipher_text_pairs:
            flag=False
            for row in key_matrix:
                if(pair[0] in row and pair[1] in row):
                    j0=row.find(pair[0])
                    j1=row.find(pair[1])
                    plain_text_pair=row[(j0+4)%5]+row[(j1+4)%5]
                    plain_text_pairs.append(plain_text_pair)
                    flag=True
            if flag:
                continue
            for j in range(5):
                col="".join([key_matrix[i][j] for i in range(5)])
                if(pair[0] in col and pair[1] in col):
                    i0=col.find(pair[0])
                    i1=col.find(pair[1])
                    plain_text_pair=col[(i0+4)%5]+col[(i1+4)%5]
                    plain_text_pairs.append(plain_text_pair)
                    flag=True
            if flag:
                continue
            i0=0
            i1=0
            j0=0
            j1=0

            for i in range(5):
                row=key_matrix[i]
                if(pair[0] in row):
                    i0=i
                    j0=row.find(pair[0])
                if(pair[1] in row):
                    i1=i
                    j1=row.find(pair[1])
            plain_text_pair=key_matrix[i0][j1]+key_matrix[i1][j0]
            plain_text_pairs.append(plain_text_pair)
        print("         !!!Decrypted Successfully!!!         ")
        print("plain text pairs: ",plain_text_pairs)
        print('cipher text: ',"".join(cipher_text_pairs))
        print('plain text (message): ',"".join(plain_text_pairs))
        print("\n=========================================\n\n")


    key=input("Enter the key: ")
    key_matrix=key_generation(key)
    print("Key Matrix for encryption:")
```

```
        print(key_matrix)
        cipher_text=input("Enter the encrypted message: ")
        conversion(cipher_text)

    elif ch == 0:
        print("\n=============================================")
        print("     Thank You for using the Software ;)      ")
        print("                  Exiting Now.                ")
        print("=============================================")
        exit()
```

Output

```
PS E:\College-Codes\Fourth Year\SEM VII\CT> & C:/Users/varun/AppData/Local/Programs/Python/Python310/python.exe "e:/College-Codes/Fourth Year/SEM VII/CT/Playfair_Cipher.py"
Welcome to Playfair Cipher Encryption and Decryption Program Made by Varun Khadayate..
 [*] Press 1 for Encryption
 [*] Press 2 for Decryption
 [*] Press 0 to exit..

Your Choice:: 1

=========================================
            !!!!Encryption!!!!
Enter the key: varun khadayate
Key Matrix for encryption:
['varun', 'khdyt', 'ebcfg', 'ilmop', 'qswxz']
Enter the message: hello my name is vk
plain text pairs:  ['he', 'lx', 'lo', 'my', 'na', 'me', 'is', 'vk']
        !!!Encrypted Successfully!!!
cipher text pairs:  ['kb', 'os', 'mp', 'od', 'vr', 'ic', 'lq', 'ke']
plain text:  hellomynameisvk
cipher text:  kbosmpodvriclqke

=========================================


Welcome to Playfair Cipher Encryption and Decryption Program Made by Varun Khadayate..
 [*] Press 1 for Encryption
 [*] Press 2 for Decryption
 [*] Press 0 to exit..

Your Choice:: 2

=========================================
             !!!Decryption!!!
Enter the key: ['varun', 'khdyt', 'ebcfg', 'ilmop', 'qswxz']
Key Matrix for encryption:
['varun', 'khdyt', 'ebcfg', 'ilmop', 'qswxz']
Enter the encrypted message: kbosmpodvriclqke
cipher text pairs:  ['kb', 'os', 'mp', 'od', 'vr', 'ic', 'lq', 'ke']
        !!!Decrypted Successfully!!!
plain text pairs:  ['he', 'lx', 'lo', 'my', 'na', 'me', 'is', 'vk']
cipher text:  kbosmpodvriclqke
plain text (message):  helxlomynameisvk

=========================================


Welcome to Playfair Cipher Encryption and Decryption Program Made by Varun Khadayate..
 [*] Press 1 for Encryption
 [*] Press 2 for Decryption
 [*] Press 0 to exit..

Your Choice:: 0

=========================================
     Thank You for using the Software ;)
                Exiting Now.
=========================================
```