ST.XAVIER’S COLLEGE

(Affiliated to Tribhuvan University)

Maitighar, Kathmandu



**Cryptography Lab Assignment #3**

**SUBMITTED BY:**

Pradeep Dahal

017BSCIT029

3rd Year/5th Sem

|  |  |
| --- | --- |
| **SUBMITTED TO:** |  |

|  |  |  |
| --- | --- | --- |
|  | **Signature** | **Remarks** |
| **Er. Anil Kumar Shah**  **Lecturer, Dept. of Computer Science** |  |  |

**TITLE: WRITE A PROGRAM TO ENCRYPT AND DECRYPT A MESSAGE USING PLAYFAIR**

**ALGORITHM**

**Theory**

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](https://www.geeksforgeeks.org/caesar-cipher/) we encrypt a pair of alphabets(digraphs) instead of a single alphabet[1].

**Procedure**

1. Input the data to be encrypted or decrypted.
2. Make a matrix of 5\*5 and give a keyword with non-repetitive letter.
3. Break the input data in the pairs
4. For each Pair
   1. If they lie in the same row encrypted alphabet is an alphabet that is in the next index of row
   2. If they lie in the same column encrypted letter is an alphabet that is in the next index of that column
   3. Else swap the position of index
5. End

**SOURCE CODE**

#import section

import numpy as np

import sys

import textwrap

#split

def wrap(s, w):

return [s[i:i + w] for i in range(0, len(s), w)]

#validate function if a keyword has a repeated data

def validate(x='default'):

temp=[]

for i in range(len(x)):

if x[i] in temp:

return "invalid"

else:

temp.append(x[i])

return "valid"

#make a matrix of 5\*5 depending upon a keyword

def makeMatrix(keyword):

temp=list(keyword)

temp\_matrix=[]

for i in range(len(temp)):

temp\_matrix.append(temp[i])

for i in range(65,91):

if((chr(i) in temp\_matrix)or i == 74):

pass

else:

temp\_matrix.append(chr(i))

matrix=[[],[],[],[],[]]

for i in range(0,25):

if(i<5):

matrix[0].append(temp\_matrix[i])

elif(i<10 and i>=5):

matrix[1].append(temp\_matrix[i])

elif(i>=10 and i<15):

matrix[2].append(temp\_matrix[i])

elif(i>=15 and i<20):

matrix[3].append(temp\_matrix[i])

elif(i>=20):

matrix[4].append(temp\_matrix[i])

return (matrix)

def makeEven(message):

temp=list(message)

if(len(temp)%2!=0):

temp.append('Z')

else:

pass

return ''.join(temp)

#encrypt the message

def encrypt(matrix, message):

dmessage=[]

chunks=(wrap(message,2))

np\_matrix=np.array(matrix)

matrixT=np.transpose(np\_matrix)

print(np\_matrix)

print(matrixT)

for i in chunks:

flag=0

for t in range(5):

if((i[0] in matrix[t] and i[1] in matrix[t])):

for j in i:

index=matrix[t].index(j)

dmessage.append(matrix[t][(index+1)%5])

elif((i[0] in matrixT[t] and i[1] in matrixT[t])):

for j in i:

index=int(np.where(matrixT==j)[1])

dmessage.append(matrixT[t][(index+1)%5])

else:

flag=1

if(flag==1):

index0 = []

index1 = []

for j in range(len(i)):

if(j==0):

r = np.where(np\_matrix==i[j])[0]

c = np.where(np\_matrix==i[j])[1]

index0.append(int(r))

index0.append(int(c))

else:

r = np.where(np\_matrix==i[j])[0]

c = np.where(np\_matrix==i[j])[1]

index1.append(int(r))

index1.append(int(c))

dmessage.append((np\_matrix[index1[0]][index0[1]]))

dmessage.append((np\_matrix[index0[0]][index1[1]]))

return ''.join(dmessage)

#main function

keyword=input("Enter keyword without repeating alphabet")

keyword=keyword.upper()

valid=(validate(keyword))

if(validate=='invalid'):

input("Invalid Kewword. Enter a key to exit")

exit()

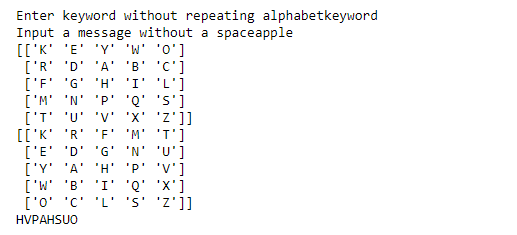
matrix=makeMatrix(keyword)

message=input("Input a message without a space").upper()

message=makeEven(message)

print(encrypt(matrix,message))

**OBSERVATION**

****

**CONCLUSION**

Thus, playfair was implemented in the python programming language.

**REFERENCE**

[1] “Playfair Cipher with Examples - GeeksforGeeks.” [Online]. Available: https://www.geeksforgeeks.org/playfair-cipher-with-examples/. [Accessed: 30-Dec-2019].