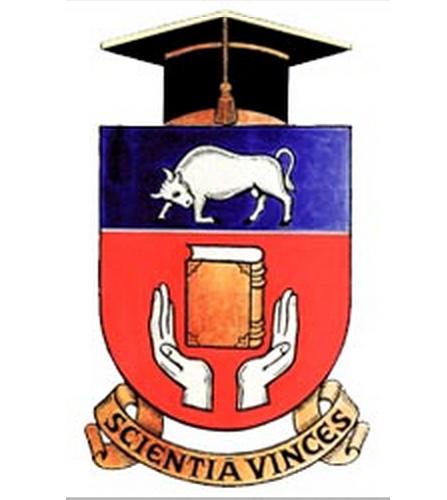
Universitatea de Stat din Tiraspol

Facultatea de Fizica Matematica si Informatica



Cifrul Playfair & Richelieu

**Elaborat de**: Dordea Pavel, student gr.3i

**Verificat**: Globa Angela

**Chișinău, 2021**

**Sistemul de criptare Playfair**

Cheia: CONTRACT DE DONATIE BILATERALA A SOCIETATII PE ACTIUNI

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

Text clar: **SPECIALITATEA**

Criptarea:

1. Se imparte textul clar in secvente de 2 caractere:

**SP EC IA LI TA TE AW**

1. Daca literele se afla pe aceeasi linie/coloanal ne deplasam, circular, cu o pozitie in directia secventei de litere;

**SP** – **PF**

**IA** - **EB**

1. Analizam dreptunghiurile cu colturile in literele specificate:

**EC** - **AN**

**LI** - **FA**

**TA** – **CI**

**TE** - **NI**

**AW** - **DV**

Text criptat: **PFANEBFACINIDV**

Decriptarea:

**PF AN EB FA CI NI DV**

**PF – SP**

**AN – EC**

**EB -**

***#COD***

# -\*- coding: utf-8 -\*-

"""

Created on Tue Dec 28 17:18:33 2021

@author: Pavel

"""

def generateKeyMatrix (key):

# Create 5X5 matrix with all values as 0

# [

# [0, 0, 0, 0, 0],

# [0, 0, 0, 0, 0],

# [0, 0, 0, 0, 0],

# [0, 0, 0, 0, 0],

# [0, 0, 0, 0, 0]

# ]

matrix\_5x5 = [[0 for i in range (5)] for j in range(5)]

simpleKeyArr = []

"""

Generate SimpleKeyArray with key from user Input

with following below condition:

1. Character Should not be repeated again

2. Replacing J as I (as per rule of playfair cypher)

"""

for c in key:

if c not in simpleKeyArr:

if c == 'J':

simpleKeyArr.append('I')

else:

simpleKeyArr.append(c)

"""

Fill the remaining SimpleKeyArray with rest of unused letters from english alphabets

"""

is\_I\_exist = "I" in simpleKeyArr

# A-Z's ASCII Value lies between 65 to 90 but as range's second parameter excludes that value we will use 65 to 91

for i in range(65,91):

if chr(i) not in simpleKeyArr:

# I = 73

# J = 74

# We want I in simpleKeyArr not J

if i==73 and not is\_I\_exist:

simpleKeyArr.append("I")

is\_I\_exist = True

elif i==73 or i==74 and is\_I\_exist:

pass

else:

simpleKeyArr.append(chr(i))

"""

Now map simpleKeyArr to matrix\_5x5

"""

index = 0

for i in range(0,5):

for j in range(0,5):

matrix\_5x5[i][j] = simpleKeyArr[index]

index+=1

return matrix\_5x5

def indexLocator (char,cipherKeyMatrix):

indexOfChar = []

# convert the character value from J to I

if char=="J":

char = "I"

for i,j in enumerate(cipherKeyMatrix):

# [

# (0, ['K', 'A', 'R', 'E', 'N']),

# (1, ['D', 'B', 'C', 'F', 'G']),

# (2, ['H', 'I', 'L', 'M', 'O']),

# (3, ['P', 'Q', 'S', 'T', 'U']),

# (4, ['V', 'W', 'X', 'Y', 'Z'])

# ]

# j refers to inside matrix => ['K', 'A', 'R', 'E', 'N'],

for k,l in enumerate(j):

# [(0,'K'),(1,'A'),(2,'R'),(3,'E'),(4,'N')]

if char == l:

indexOfChar.append(i)

indexOfChar.append(k)

return indexOfChar

def encryption (plainText,key):

cipherText = []

# 1. Generate Key Matrix

keyMatrix = generateKeyMatrix(key)

# 2. Encrypt According to Rules of playfair cipher

i = 0

while i < len(plainText):

# 2.1 calculate two grouped characters indexes from keyMatrix

n1 = indexLocator(plainText[i],keyMatrix)

n2 = indexLocator(plainText[i+1],keyMatrix)

# 2.2 if same column then look in below row so

# format is [row,col]

# now to see below row => increase the row in both item

# (n1[0]+1,n1[1]) => (3+1,1) => (4,1)

# (n2[0]+1,n2[1]) => (4+1,1) => (5,1)

# but in our matrix we have 0 to 4 indexes only

# so to make value bound under 0 to 4 we will do %5

# i.e.,

# (n1[0]+1 % 5,n1[1])

# (n2[0]+1 % 5,n2[1])

if n1[1] == n2[1]:

i1 = (n1[0] + 1) % 5

j1 = n1[1]

i2 = (n2[0] + 1) % 5

j2 = n2[1]

cipherText.append(keyMatrix[i1][j1])

cipherText.append(keyMatrix[i2][j2])

cipherText.append(", ")

# same row

elif n1[0]==n2[0]:

i1= n1[0]

j1= (n1[1] + 1) % 5

i2= n2[0]

j2= (n2[1] + 1) % 5

cipherText.append(keyMatrix[i1][j1])

cipherText.append(keyMatrix[i2][j2])

cipherText.append(", ")

# if making rectangle then

# [4,3] [1,2] => [4,2] [3,1]

# exchange columns of both value

else:

i1 = n1[0]

j1 = n1[1]

i2 = n2[0]

j2 = n2[1]

cipherText.append(keyMatrix[i1][j2])

cipherText.append(keyMatrix[i2][j1])

cipherText.append(", ")

i += 2

return cipherText

def convertPlainTextToDiagraphs (plainText):

# add X if Two letters are being repeated

for s in range(0,len(plainText)+1,2):

if s<len(plainText)-1:

if plainText[s]==plainText[s+1]:

plainText=plainText[:s+1]+'X'+plainText[s+1:]

# append X if the total letters are odd, to make plaintext even

if len(plainText)%2 != 0:

plainText = plainText[:]+'X'

return plainText

def main():

key = input("Introduceti key-ul: ").replace(" ","").upper()

plainText =input("Textul clar: ").replace(" ","").upper()

convertedPlainText = convertPlainTextToDiagraphs(plainText)

print('Textul clar introdus : '+convertedPlainText)

keyMatrix = generateKeyMatrix(key)

cipherText = " ".join(encryption(convertedPlainText,key))

print('Textul criptat : '+cipherText)

# print(len(keyMatrix)) => 5

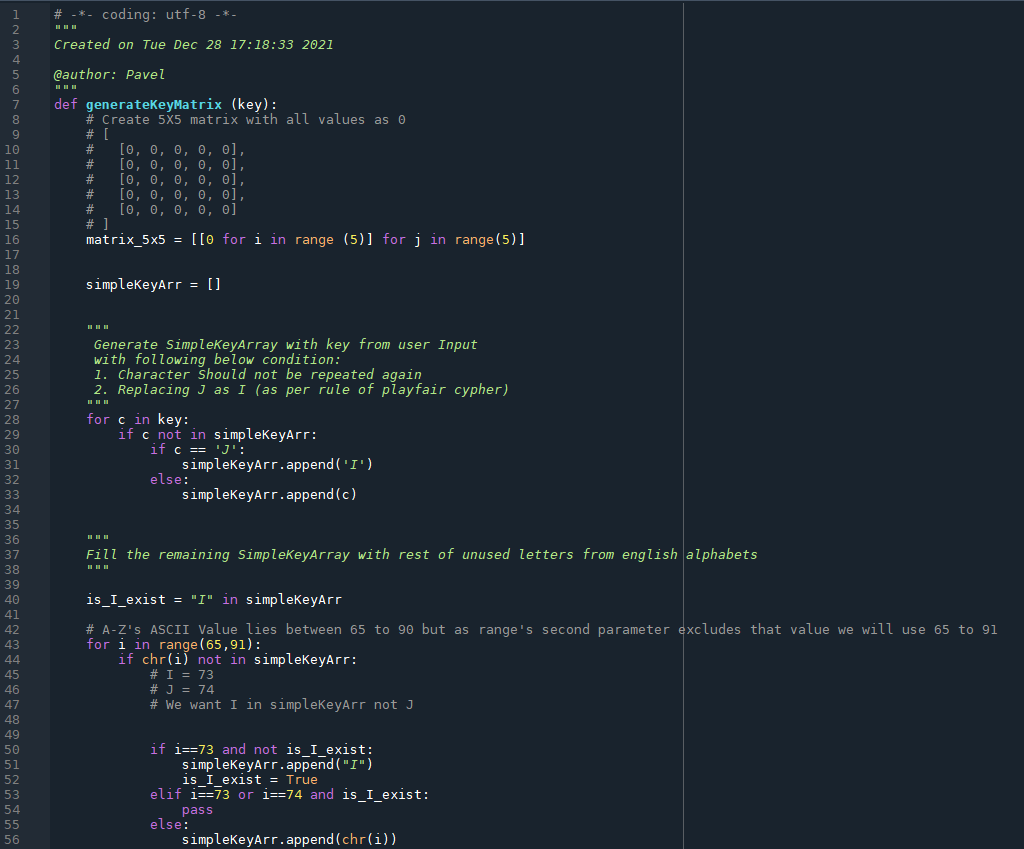
print('\nMatricea : \n')

print(keyMatrix)

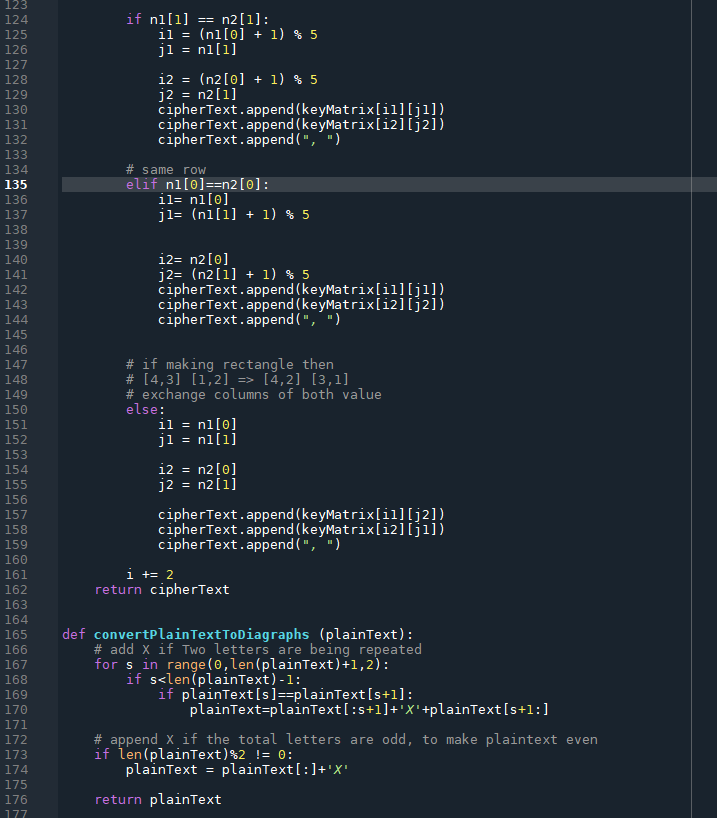
if \_\_name\_\_ == "\_\_main\_\_":

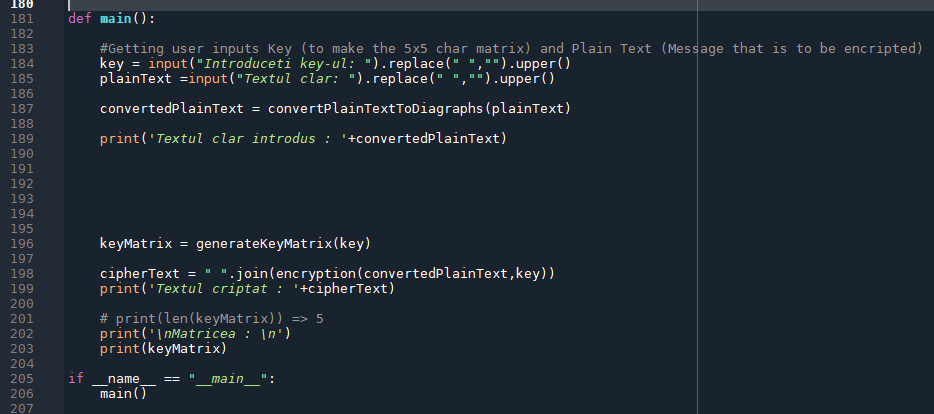
main()

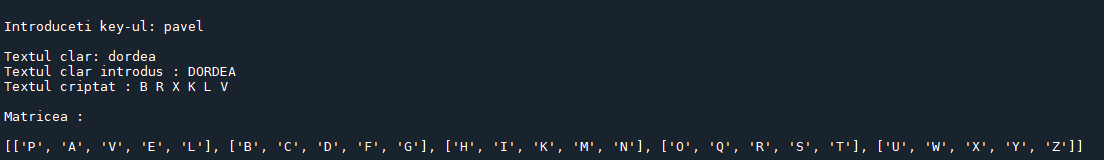
***#Screenshots***











**Sistemul de criptare Rishelieu**

Text clar: **securitatea informatiei este un obiect…**

**Criptarea:**

1. Scriem textul intr-un tabel patrat de dimensiunea n, unde n\*n se divide cu 4 (n=2,4,6,8,10,….);
2. Fie n=4;
3. Cream cheia conform regulii:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

begin

a[i,j]:=0;

a[j,n-i+1]:=1;

a[n-j+1,i]:=1;

a[n-i+1,n-j+1]:=1; k:=k+1;

end;

**daca n=4, atunci avem nevoie de n2/4 spatii.**

**Deci, vor fi 4 spații.**

Inițial, alegem primul spațiu în mod aleator.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Fie, a[2,3]=0 (spatiu),i=2,j=3,

a[j,n-i+1]=a[3,4-2+1]= a[3,3]=1;

a[n-j+1,i]= a[4-3+1,2]:=a[2,2]=1;

a[n-i+1,n-j+1]:= a[4-2+1,4-3+1]:=a[3,2]=1

Din nou, aleatory, alegem un spațiu din cele nemarcate.

Fie **a[1,3]=0**, i=1,j=3

a[j,n-i+1]=a[3,4-1+1]= a[3,4]=1;

a[n-j+1,i]= a[4-3+1,1]:=a[2,1]=1;

a[n-i+1,n-j+1]:= a[4-1+1,4-3+1]:=a[4,2]=1

Alegem aleatory umătorul spațiu.

Fie, **a[3,1]=0**, i=3,j=1

a[j,n-i+1]=a[1,4-3+1]= a[1,2]=1;

a[n-j+1,i]= a[4-1+1,3]:=a[4,3]=1;

a[n-i+1,n-j+1]:= a[4-3+1,4-1+1]:=a[2,4]=1

1. Pentru criptare, cheia se roteste de trei ori la 90 de grade;
2. Scriem literele care se vad in celulele transparente;

Text criptat:

euiaftiub t\_oa\_t\_\_e ctnresnoi sraeimiee

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| s | e | c | u | r | i |
| t | a | t | e | a |  |
| i | n | f | o | r | m |
| a | t | i | e | i |  |
| e | s | t | e |  | u |
| n |  | o | b | i | e |

Pentru decrptare:

1. Se ia o retea goala si se suprapune pe ea cheia
2. Se scrie in cellele transparente literele din textul ctiptat in ordine dupa linii si Coloane
3. Se roteste la 90 de grade cheia de trei ori si se repeta acelas algoritm.

Decriptarea: euiaftiub t\_oa\_t\_\_e ctnresnoi sraeimiee

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| s | e | c | u | r | i |
| t | a | t | e | a | \_ |
| i | n | f | o | r | m |
| a | t | i | e | i | \_ |
| e | s | t | e | \_ | u |
| n | \_ | o | b | i | e |