

Ministerul Educatiei, Culturii și Cercetarii al Republicii Moldova Universitatea Tehnic**ă** a Moldovei

Facultatea Calculatoare, Informatic**ă ş**i Microelectronic**ă** Departamentul Ingineria Software și Automatica

Report

for laboratory work No. 3

course "Cryptography and Security"

Done by: Felicia LUPAȘCU, gr. FAF-212

Checked by:  
 Cǎtǎlin MÎŢU

Chișinău – 2023

**Subject:** Cryptanalysis of monoalphabetic ciphers

## Tasks:

To implement the Playfair algorithm in a programming language for messages in Romanian (31 letters), ensure that text character values are within the range of 'A' to 'Z' and 'a' to 'z.' Any other values will be suggested as incorrect. The key must be at least 7 characters long. Users can select the operation, either encryption or decryption, input the key, message, or cryptogram, and receive the decrypted cryptogram or message. Adding new spaces, based on the language used and the logic of the message, will be a manual step in the final phase.

**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, we encrypt a pair of alphabets (digraphs) instead of a single alphabet.

It was used for tactical purposes by British forces in the Second Boer War and in World War I and for the same purpose by the Australians during World War II. This was because Playfair is reasonably fast to use and requires no special equipment

**Implementation:**

Playfair Algorithm:

Prepare the Text for Encryption:

* Convert the text to uppercase.
* Replace "J" with "I."
* Create letter pairs (couples).
* Remove any spaces and punctuation.
* Add "Q," "X," or "Z" between letters of couples with the same letter.

Construct the Encryption Matrix:

* Ensure there are no duplicates in the key (and handle "J"s).
* Create the matrix by adding the key and the alphabet without duplication.

Encrypt the Message:

For different rows and columns:

* Substitute a letter with the letter in the same row but from the column of the other letter.

For the same rows:

* Substitute letters with the next letter in the same row.

For the same columns:

* Substitute letters with the next letter in the same column.

For decryption:

* Rule 1 remains the same.
* Rules 2 and 3 are applied in the opposite direction.

My code implements a Python program for the Playfair cipher, a manual symmetric encryption technique. The Playfair cipher encrypts and decrypts text messages using a 5x6 matrix and a key.

Here's an explanation of the code and each function:

***add\_letters(message):*** This function adds "Q," "X," or "Z" between letters of pairs with the same letter and ensures that the message length is even. It helps comply with the Playfair cipher rules, where double letters need separation, and the message should be of even length. The function inserts one of the specified characters between repeated letters and appends an 'F' if the message length is odd.

def add\_letters(message):  
 # Characters to insert when two identical characters are adjacent  
 insertion\_characters = ['Q', 'X', 'Z']  
 message\_list = list(message) # Convert the message to a list for easier manipulation  
 i = 0  
 while i < len(message\_list) - 1:  
 if message\_list[i] == message\_list[i + 1]:  
 # Insert a random character from the insertion list  
 message\_list.insert(i + 1, random.choice(insertion\_characters))  
 i += 2  
 # If the message has an odd length, add additional letter  
 if len(message\_list) % 2 != 0:  
 message\_list.append('F')  
 # Convert the list back to a string  
 return "".join(message\_list)

***find\_letter\_indices(letter, matrix):*** This function finds the row and column indices of a given letter within a matrix. It is used during encryption and decryption to locate letters within the matrix.

***romanian\_language(string):*** This function checks if a string consists of characters within the Romanian language (both uppercase and lowercase). It uses a regular expression pattern to match valid characters, ensuring that only valid characters are used in the encryption process.

***encrypt(message, matrix):*** This function performs the encryption of a given message using a Playfair matrix and key. It iterates through the message in pairs and follows three conditions for substitution, based on the relative positions of the letters in the matrix. The result is the encrypted ciphertext.

***decrypt(ciphertext, matrix):*** This function reverses the encryption process. It decrypts a ciphertext using the same Playfair matrix and key. It follows conditions opposite to the encryption process to reveal the original message. It also eliminates the additional characters introduced during encryption.

***create\_matrix(key):*** This function generates a 5x6 matrix for the Playfair cipher using a provided key. The matrix combines the key and the Romanian alphabet, eliminating duplicates. The Playfair matrix helps determine the letter substitutions during encryption and decryption.

def create\_matrix(key):  
 # Create the initial matrix with 5 rows and 6 columns  
 matrix = [['' for \_ in range(6)] for \_ in range(5)]  
 # Define the alphabet without J and duplicate letters  
 alphabet = "AĂÂBCDEFGHIÎKLMNOPQRSȘTȚUVWXYZ"  
 unique\_key = ''.join(dict.fromkeys(key.upper())) # Remove duplicates and make uppercase  
 # Fill the matrix with unique characters from the key and then the remaining alphabet  
 key\_index = 0  
 for row in range(5):  
 for col in range(6):  
 if key\_index < len(unique\_key):  
 matrix[row][col] = unique\_key[key\_index]  
 key\_index += 1  
 else:  
 # Fill with remaining alphabet characters  
 while alphabet and matrix[row][col] == '':  
 letter = alphabet[0]  
 alphabet = alphabet[1:]  
 # Skip the letter 'J' if it's in the alphabet  
 if letter != 'J':  
 matrix[row][col] = letter  
 return matrix

***input\_key():*** This function takes user input for the encryption key, ensuring that it consists of Romanian characters and is at least seven characters long. It also converts the key to uppercase, replaces 'J' with 'I,' and eliminates duplicate characters.

The code's main section manages user interaction. It presents a menu for encryption, decryption, and exiting the program. The user can input a message or ciphertext, and the program guides them through the process, enforcing rules to maintain data integrity. The result is either an encrypted ciphertext or a decrypted message.

In summary, this code implements the Playfair cipher, which is a simple but effective manual encryption technique. It ensures that the encryption and decryption processes adhere to the rules of the Playfair cipher while maintaining data consistency.

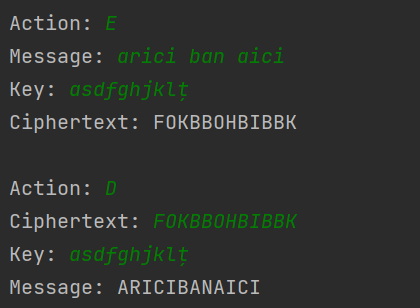


Figure 1 Example of Encryption/Decryption

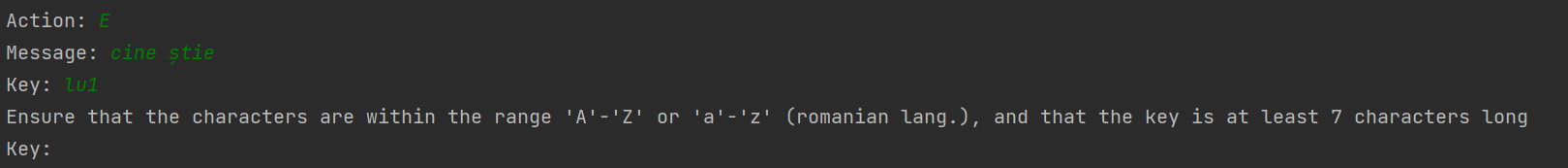


Figure 2 Example of Error

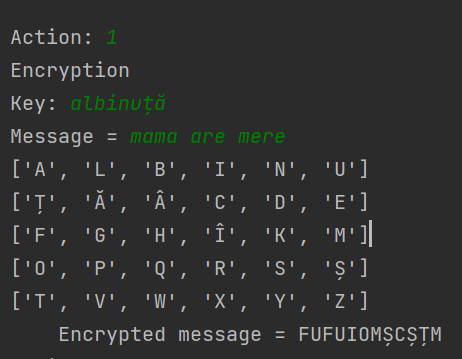
****

Figure 3Matrix Example

**Conclusion:**

In this laboratory, I had the opportunity to work on implementing and understanding the Playfair cipher, a classical symmetric encryption technique. This cipher involves various components and rules to encrypt and decrypt messages securely. This experience has provided a foundation for understanding more complex cryptographic systems and their real-world applications in data security.

https://github.com/feliciaL3/CS/tree/main/Lab3