Ministry of Education, Culture and Research of the Republic of Moldova
Technical University of Moldova
Department of Software and Automation Engineering

REPORT

Laboratory work no. 3
Polyalphabetic ciphers

Elaborated:
st. gr. FAF-213 Botnari Ciprian

Verified:
asist. univ. Cătălin Mîțu

Chişinău – 2023

Table of Contents

Topic: Polyalphabetic ciphers

Tasks

Implement the Vigenère algorithm in one of the programming languages for Romanian alphabet (31 letters), these being encoded with the numbers 0, 1, ... 30. The values of the text characters are between 'A' and 'Z', 'a' and 'z' and there are no other values allowed. If the user enters other values - the correct range should be displayed. The key length must not be less than 7. Encryption and decryption will according to the formulas in the mathematical model shown below. Spaces from the message must first be removed, then all letters will be converted to upper case. User will be able to choose the operation - encryption or decryption, enter the key, message or cryptogram and get the cryptogram or the decrypted message.

Theoretical notes

The encryption method known as the "Vigenère cipher" has been misattributed to Blaise de Vigenère in the 19th century and was actually first described by Giovan Battista Bellaso in his 1553 book *La cifra del. Sig*. Vigenère created a similar cipher, but it was different and more powerful in 1586. On the other hand, the Vigenère cipher uses the same operations as the Caesar cipher. The cipher Vigenère and kind moves the letters, but unlike Caesar, it cannot easily break into 26 combinations. The Vigenère cipher uses a multiple shift. The key does not consist of a single but of several, being generated by several integers k_i, where $0 \leq k_i \leq 25$, if we take as a reference the 26-letter Latin alphabet. The encryption is done as follows:

$$c_i = (m_i + k_i)\ mod\ 26$$

The key may be, for example, k = (5, 20, 17, 10, 20, 13) and would cause the first letter to move by 5, $c_1 = m_1 + 5\ (mod\ 26)$, the second by 20, $c_2 = m_2 + 20\ (mod\ 26)$, and so on to the end of of the key and then from the beginning again. The key is usually a word, to make it easier to remember - The above key corresponds to the word "furtun". The multiple displacement method offers protection additional protection for two reasons:

- others do not know the length of the key;
- the number of possible solutions increases with the size of the key;

For example, for a key length of 5, the number of combinations that would be required in an exhaustive search would be 26^5 = 11 881 376. Decryption for the Vigenère cipher is similar to encryption. The difference is that that the key is subtracted from the cipher text,

$$m_i = (c_i - k_i)\ mod\ 26$$

To simplify the encryption process the following table, called *Tabula Recta* (Table 1) is used by Vigenère. Here all 26 ciphers are located on horizontal and each cipher corresponds to a certain letter of the key, represented in the column in left of the table. The alphabet corresponding to the letters of the clear text is in the first line at the top of the of the table. The encryption process is simple - it requires that having the letter m_i in the message and the letter k_i in the 2 key to find the cipher text letter c_i, which is at the intersection of the m_i line and the k_i column. In the example in Table 1 shows the case m_i = M and k_i = H, and the result is c_i = T.

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

Table 1. *Tabula Recta*

Code

```
def vigenere_cipher(action, key, text):
  alphabet = "AĂÂBCDEFGHIÎJKLMNOPQRSȘTȚUVWXYZ"
  result = ""

  for i in range(len(text)):
    if action == "encrypt":
      text_char_index = alphabet.index(text[i])
      key_char_index = alphabet.index(key[i % len(key)])
      new_char_index = (text_char_index + key_char_index) % len(alphabet)
      result += alphabet[new_char_index]
    elif action == "decrypt":
      text_char_index = alphabet.index(text[i])
      key_char_index = alphabet.index(key[i % len(key)])
      new_char_index = (text_char_index - key_char_index) % len(alphabet)
      result += alphabet[new_char_index]
    else:
      print("Invalid action")
      break

  return result
```

Results

Let's cipher, the message *Carpe diem* with the key k = **narutos**. To encrypt or decrypt we first make the following correspondence (we encode the alphabet):

A	Ă	Â	B	C	D	E	F	G	H	I	Î	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Figure 1. *Alphabet*

N	O	P	Q	R	S	Ș	T	Ț	U	V	W	X	Y	Z
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30

Figure 2. *Alphabet*

We get the following tables for encryption and decryption:

Text	C	A	R	P	E	D	I	E	M
Key	N	A	R	U	T	O	S	N	A
Text Index	4	0	20	18	6	5	10	6	15
Key Index	16	0	20	25	23	17	21	16	0
Sum Index	20	0	9	12	29	22	0	22	15
Cryptogram	R	A	H	J	Y	Ș	A	Ș	M

Table 2. *Encryption*

Cryptogram	R	A	H	J	Y	Ș	A	Ș	M
Key	N	A	R	U	T	O	S	N	A
Cryptogram Index	20	0	9	12	29	22	0	22	15
Key Index	16	0	20	25	23	17	21	16	0
Diff Index	4	0	20	18	6	5	10	6	15
Text	C	A	R	P	E	D	I	E	M

Table 3. *Decryption*

Conclusion

In summary, the laboratory work on the Vigenère cipher has yielded valuable insights into classical cryptography. Through this experiment, I have gained practical experience in both encrypting and decrypting messages using the Vigenère cipher. Additionally, I have observed the cipher's effectiveness in providing a higher level of security compared to simpler substitution ciphers. It has become evident that the strength of the Vigenère cipher lies in its use of a keyword, which adds an extra layer of complexity to the encryption process. This hands-on exploration has deepened my understanding of historical encryption techniques and their relevance in the context of modern cryptography and information security.

Github

Sufferal/cryptography-labs (github.com)