Оглавление

[**1.** **Введение** 3](#_Toc913937)

[**2.** **Теоретическая часть** 4](#_Toc913938)

[**3.** **Блок-схема алгоритма** 6](#_Toc913939)

[**3.1. Шифр Цезаря** 6](#_Toc913940)

[**3.2. Шифр Виженера** 7](#_Toc913941)

[**4.** **Результаты работы программы** 8](#_Toc913942)

[**5.** **Исходный код программы** 9](#_Toc913943)

[**6.** **Вывод** 1](#_Toc913944)2

# **Введение**

В рамках данной лабораторной работы планируется изучить такие перестановочные шифры как шифр Цезаря и шифр Виженера, а также создать программный продукт, реализующий шифрование и дешифрование текстовой информации, записанной в файле с помощью данных шифров.

# **Теоретическая часть**

**Шифр Цезаря** — один из древнейших шифров. При шифровании каждый символ заменяется другим, отстоящим от него в алфавите на фиксированное число позиций. Шифр Цезаря можно классифицировать как шифр подстановки, при более узкой классификации — шифр простой замены.

Шифр назван в честь римского императора Гая Юлия Цезаря, использовавшего его для секретной переписки. Естественным развитием шифра Цезаря стал шифр Виженера. С точки зрения современного криптоанализа, шифр Цезаря не имеет приемлемой стойкости.

**Математическая модель**

Если сопоставить каждому символу алфавита его порядковый номер (нумеруя с 0), то шифрование и дешифрование можно выразить формулами модульной арифметики:

*y = (x + (k mod n))) mod n{\displaystyle y=(x+k) mod n ~\,\! }*

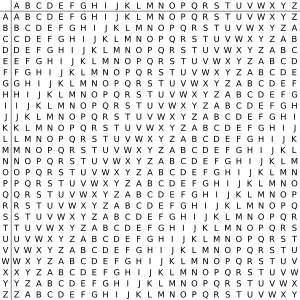
*x = (y − (k mod n)) mod n,*{\displaystyle x=(y+n-(k\ mod\ n)) mod n ~\,\! }

где *x* — символ открытого текста, *y* — символ шифрованного текста, *n* — мощность алфавита, а *k* — сдвиг(ключ).

**Шифр Виженера:**

В шифре Цезаря каждая буква алфавита сдвигается на несколько позиций; например в шифре Цезаря при сдвиге +3, A стало бы D, B стало бы E и так далее. Шифр Виженера состоит из последовательности нескольких шифров Цезаря с различными значениями сдвига. Для зашифровывания может использоваться таблица алфавитов, называемая tabula recta или квадрат (таблица) Виженера. Применительно к латинскому алфавиту таблица Виженера составляется из строк по 26 символов, причём каждая следующая строка сдвигается на несколько позиций. Таким образом, в таблице получается 26 различных шифров Цезаря.

Пример квадрата Виженера:



Например, предположим, что исходный текст имеет вид:

*ATTACKATDAWN*

Человек, посылающий сообщение, записывает ключевое слово («LEMON») циклически до тех пор, пока его длина не будет соответствовать длине исходного текста:

*LEMONLEMONLE*

Первый символ исходного текста A зашифрован последовательностью L, которая является первым символом ключа. Первый символ L шифрованного текста находится на пересечении строки L и столбца A в таблице Виженера. Точно так же для второго символа исходного текста используется второй символ ключа; то есть второй символ шифрованного текста X получается на пересечении строки E и столбца T. Остальная часть исходного текста шифруется подобным способом.

Исходный текст: ATTACKATDAWN

Ключ: *LEMONLEMONLE*

Зашифрованный текст: LXFOPVEFRNHR

**Дешифрование:**

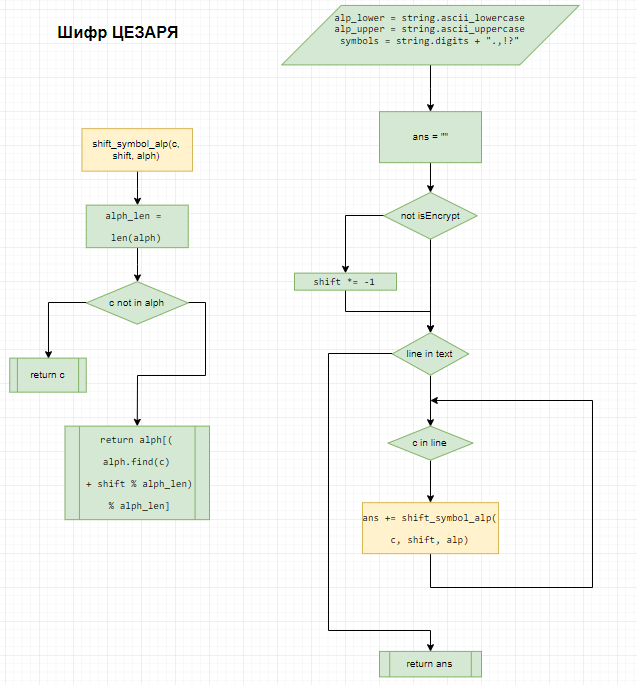
Зашифрованный текст: LXFOPVEFRNHR

Ключ: *LEMONLEMONLE*

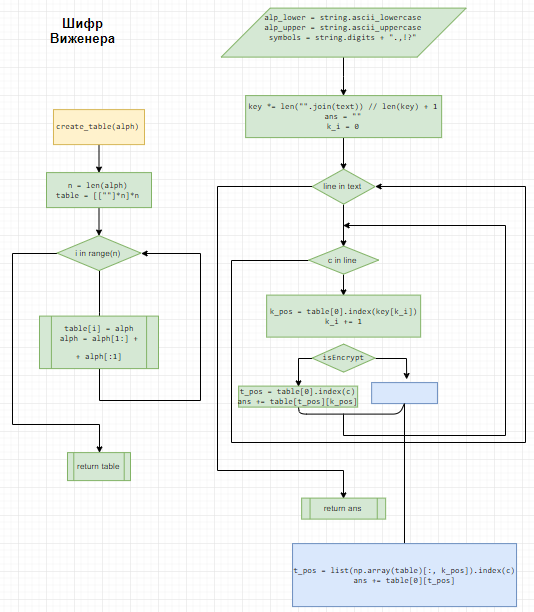
Дешифрованный текст: ATTACKATDAWN

# **Блок-схема алгоритма**

## **3.1. Шифр Цезаря**



## **3.2. Шифр Виженера**



# **Результаты работы программы**

*Результат работы шифра Цезаря:*

Enter the key: 3

File encoded!

Enter the key: 3

File decoded!

*Результат работы шифра Виженера:*

Enter the key: lol

File encoded!

Enter the key: lol

File decoded!

# **Исходный код программы**

*Шифр Цезаря:*

class Caesar:

def \_\_init\_\_(self):

self.input\_file = None

self.output\_file = None

self.\_\_alpha = '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ !"#$%&\()\*+,-./:;<=>?@[]^\_`{' \

'|}~абвгдеёжзийклмнопрстуфхцчшщъыьэюяАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ'

self.\_\_result = ''

def encode(self, key):

with open(self.input\_file) as f:

lines = f.readlines()

for s in lines:

for c in s:

if c == '\n':

self.\_\_result += '\n'

continue

self.\_\_result += self.\_\_alpha[(self.\_\_alpha.index(c) + key) % len(self.\_\_alpha)]

with open(self.output\_file, 'w') as f:

for c in self.\_\_result:

if c == '\n':

f.write('\n')

continue

f.write(c)

self.\_\_result = None

def decode(self, key):

with open(self.input\_file) as f:

lines = f.readlines()

for s in lines:

for c in s:

if c == '\n':

self.\_\_result += '\n'

continue

self.\_\_result += self.\_\_alpha[(self.\_\_alpha.index(c) - key) % len(self.\_\_alpha)]

with open(self.output\_file, 'w') as f:

for c in self.\_\_result:

if c == '\n':

f.write('\n')

continue

f.write(c)

self.\_\_result = None

*Шифр Виженера:*

import string

import numpy as np

class Vigener:

@staticmethod

def \_\_create\_table():

alp\_lower = [i for i in string.ascii\_lowercase]

alp\_upper = [i for i in string.ascii\_uppercase]

symbols = [i for i in (string.digits + ".,!?")]

rus\_upper = [chr(i) for i in range(1040, 1072)]

rus\_lower = [chr(i) for i in range(1072, 1104)]

alpha = alp\_lower + alp\_upper + rus\_lower + rus\_upper + symbols

n = len(alpha)

table = [[""] \* n] \* n

for i in range(n):

table[i] = alpha

alpha = alpha[1:] + alpha[:1]

return table

def \_\_init\_\_(self):

self.input\_file = None

self.output\_file = None

self.\_\_result = ''

self.\_\_table = Vigener.\_\_create\_table()

def encode(self, key):

k\_i = 0

with open(self.input\_file) as f:

lines = f.readlines()

key \*= len("".join(lines)) // len(key) + 1

for s in lines:

for c in s:

if c not in self.\_\_table[0]:

self.\_\_result += c

continue

k\_pos = self.\_\_table[0].index(key[k\_i])

k\_i += 1

t\_pos = self.\_\_table[0].index(c)

self.\_\_result += self.\_\_table[t\_pos][k\_pos]

with open(self.output\_file, 'w') as f:

for c in self.\_\_result:

if c == '\n':

f.write('\n')

continue

f.write(c)

self.\_\_result = None

def decode(self, key):

k\_i = 0

with open(self.input\_file) as f:

lines = f.readlines()

key \*= len("".join(lines)) // len(key) + 1

for s in lines:

for c in s:

if c not in self.\_\_table[0]:

self.\_\_result += c

continue

k\_pos = self.\_\_table[0].index(key[k\_i])

k\_i += 1

t\_pos = list(np.array(self.\_\_table)[:, k\_pos]).index(c)

self.\_\_result += self.\_\_table[0][t\_pos]

with open(self.output\_file, 'w') as f:

for c in self.\_\_result:

if c == '\n':

f.write('\n')

continue

f.write(c)

self.\_\_result = None

# **Вывод**

В данной лабораторной работе были рассмотрены шифры Цезаря и Виженера. Были построены алгоритмы, выполняющие шифрование и дешифрование текстового файла с помощью выбранного алгоритма.