Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей Кафедра информатики

Дисциплина: Информационные сети. Основы безопасности

**ОТЧЕТ**

к лабораторной работе № 1

Выполнил:

студент гр. 153501

Талай А.А.

Проверил:

Лещенко Е.А.

Минск 2024

СОДЕРЖАНИЕ

[Введенеие 3](#_Toc157436533)

[1 Краткие теоретические сведения 4](#_Toc157436534)

[2 Результат выполнения программы 7](#_Toc157436535)

[Приложение А 9](#_Toc157436537)

# ВВЕДЕНИЕ

Шифр – система обратимых преобразований, зависящая от некоторого секретного параметра (ключа) и предназначенная для обеспечения секретности передаваемой информации.

Цель данной лабораторной работы заключается в изучении теоретических сведений по работе Шифра Цезаря и Шифра Виженера и разработки программы, шифрующей и дешифрующей информацию с помощью данных шифров.

# 1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Шифр Цезаря, также известный, как шифр сдвига, код Цезаря или сдвиг Цезаря – один из самых простых и наиболее широко известных методов шифрования.

Шифр Цезаря – это вид шифра подстановки, в котором каждый символ в открытом тексте заменяется символом находящимся на некотором постоянном числе позиций левее или правее него в алфавите. Например, в шифре со сдвигом 4 А была бы заменена на Г, Б станет Д, и так далее.

Шифр назван в честь римского императора Гая Юлия Цезаря, использовавшего его для секретной переписки со своими генералами.

Шаг шифрования, выполняемый шифром Цезаря, часто включается как часть более сложных схем, таких как шифр Виженера, и все ещё имеет современное приложение в системе ROT13. Как и все моноалфавитные шифры, шифр Цезаря легко взламывается и не имеет практически никакого применения на практике.



Рисунок 1. Шифр Цезаря

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

x=(y-k+n)\ \mod\ n,

где ~x – символ открытого текста, ~y – символ шифрованного текста, ~n – мощность алфавита, а ~k – ключ.

С точки зрения математики шифр Цезаря является частным случаем аффинного шифра.

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. На каждом этапе шифрования используются различные алфавиты, выбираемые в зависимости от символа ключевого слова. Например, предположим, что исходный текст имеет такой 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.

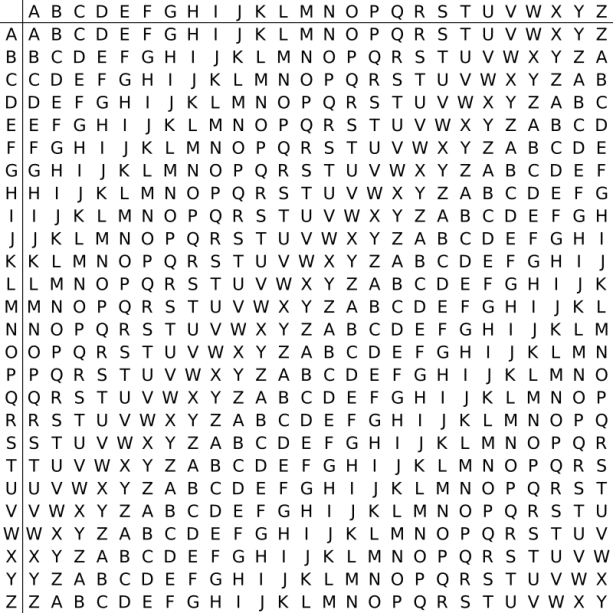


Рисунок 2. – Квадрат Виженера

Расшифровывание производится следующим образом: находим в таблице Виженера строку, соответствующую первому символу ключевого слова; в данной строке находим первый символ зашифрованного текста. Столбец, в котором находится данный символ, соответствует первому символу исходного текста. Следующие символы зашифрованного текста расшифровываются подобным образом.

Если n — количество букв в алфавите, m j — буквы открытого текста, k j — буквы ключа, то шифрование Виженера можно записать следующим образом:c j = m j + k j ( mod n )

И расшифровывание:

m j = c j − k j ( mod n )

В компьютере такая операция соответствует сложению кодов ASCII символов сообщения и ключа по некоторому модулю. Кажется, что если таблица будет более сложной, чем циклическое смещение строк, то шифр станет надежнее. Это действительно так, если ее менять чаще, например, от слова, к слову. Но составление таких таблиц, представляющих собой латинские квадраты, где любая буква встречается в строке или столбце один раз, трудоемко и его стоит делать лишь на ЭВМ. Для ручного же многоалфавитного шифра полагаются лишь на длину и сложность ключа, используя приведенную таблицу, которую можно не держать в тайне, а это упрощает шифрование и расшифровывание.

# 2 РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ ПРОГРАММЫ

В результате разработки программы было создано графическое приложение, осуществляющее шифровку и дешифровку информации из файлов по заданным ключам.

На рисунке 2.1 представлена блок-схема алгоритма для работы с шифром Цезаря.

На рисунке 2.2 представлена блок-схема алгоритма для работы с шифром Виженера.

# 

Рисунок 2.1. – Блок-схема алгоритма для работы с шифром Цезаря

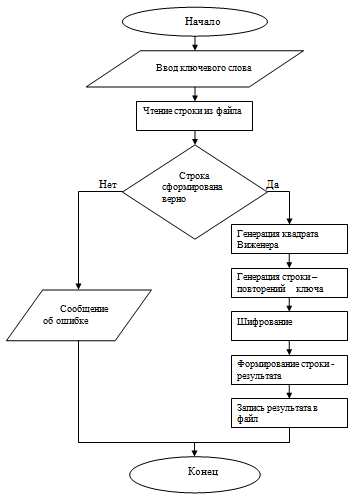


Рисунок 2.2. – Блок-схема алгоритма для работы с шифром Виженера

Результат выполнения программы изображён на рисунке 2.3 и 2.4.

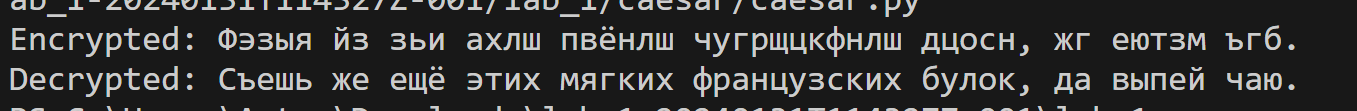


Рисунок 2.3. – Результат шифрования с помощью шифра Цезаря

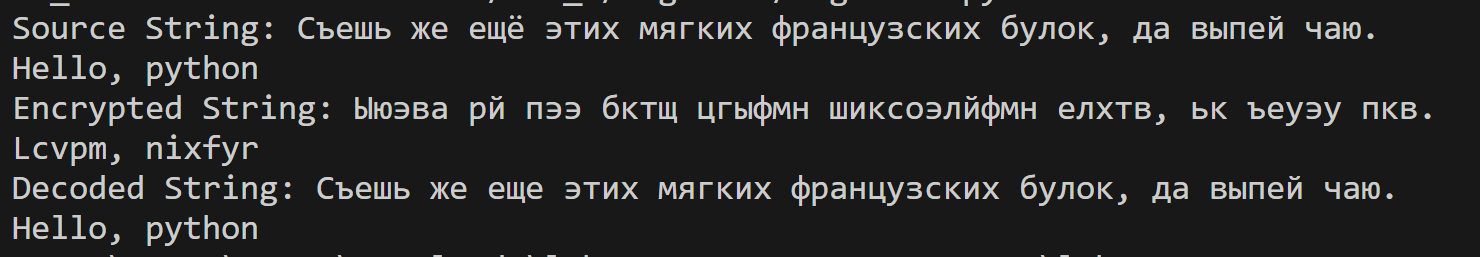


Рисунок 2.4. – Результат дешифрования с помощью шифра Виженера

# ПРИЛОЖЕНИЕ А

**(обязательное)**

**Исходный код программы**

Листинг 1 – Файл caesar.py:

def caesar\_encrypt(text, shift):

    english\_alphabet\_lower = 'abcdefghijklmnopqrstuvwxyz'

    english\_alphabet\_upper = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

    russian\_alphabet\_lower = 'абвгдеёжзийклмнопрстуфхцчшщъыьэюя'

    russian\_alphabet\_upper = 'АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ'

    encrypted\_text = ''

    for char in text:

        if char in english\_alphabet\_lower:

            index = (english\_alphabet\_lower.index(char) + shift) % len(english\_alphabet\_lower)

            encrypted\_text += english\_alphabet\_lower[index]

        elif char in english\_alphabet\_upper:

            index = (english\_alphabet\_upper.index(char) + shift) % len(english\_alphabet\_upper)

            encrypted\_text += english\_alphabet\_upper[index]

        elif char in russian\_alphabet\_lower:

            index = (russian\_alphabet\_lower.index(char) + shift) % len(russian\_alphabet\_lower)

            encrypted\_text += russian\_alphabet\_lower[index]

        elif char in russian\_alphabet\_upper:

            index = (russian\_alphabet\_upper.index(char) + shift) % len(russian\_alphabet\_upper)

            encrypted\_text += russian\_alphabet\_upper[index]

        else:

            encrypted\_text += char

    return encrypted\_text

def caesar\_decrypt(encrypted\_text, shift):

    return caesar\_encrypt(encrypted\_text, -shift)

with open('C:\\Users\\Artem\\Downloads\\lab\_1-20240131T114327Z-001\\lab\_1\\caesar\\input.txt', 'r', encoding='utf-8') as file:

    text = file.read()

shift = 3

encrypted\_text = caesar\_encrypt(text, shift)

print(f"Encrypted: {encrypted\_text}")

decrypted\_text = caesar\_decrypt(encrypted\_text, shift)

print(f"Decrypted: {decrypted\_text}")

Листинг 2 – Файл vigenere.py:

def get\_caesar\_key(character):

    if 'A' <= character <= 'Z':

        return ord(character) - ord('A')

    elif 'a' <= character <= 'z':

        return ord(character) - ord('a')

    elif 'А' <= character <= 'Я':

        return ord(character) - ord('А')

    elif 'а' <= character <= 'я':

        return ord(character) - ord('а')

    elif character == 'Ё':

        return ord('Е') - ord('А') + 1

    elif character == 'ё':

        return ord('е') - ord('а') + 1

    else:

        return 0

def shift\_character(character, key, alphabet\_start, alphabet\_length):

    return chr((ord(character) - ord(alphabet\_start) + key) % alphabet\_length + ord(alphabet\_start))

def string\_encryption\_v(source\_string, key):

    encrypted\_string = ""

    for i in range(len(source\_string)):

        caesar\_key = get\_caesar\_key(key[i % len(key)])

        character = source\_string[i]

        if 'A' <= character <= 'Z':

            encrypted\_string += shift\_character(character, caesar\_key, 'A', 26)

        elif 'a' <= character <= 'z':

            encrypted\_string += shift\_character(character, caesar\_key, 'a', 26)

        elif 'А' <= character <= 'Я':

            encrypted\_string += shift\_character(character, caesar\_key, 'А', 32)

        elif 'а' <= character <= 'я':

            encrypted\_string += shift\_character(character, caesar\_key, 'а', 32)

        elif character == 'Ё':

            encrypted\_string += shift\_character('Е', caesar\_key, 'А', 32)

        elif character == 'ё':

            encrypted\_string += shift\_character('е', caesar\_key, 'а', 32)

        else:

            encrypted\_string += character

    return encrypted\_string

def string\_decoding\_v(source\_string, key):

    decoded\_string = ""

    for i in range(len(source\_string)):

        caesar\_key = -get\_caesar\_key(key[i % len(key)])

        character = source\_string[i]

        if 'A' <= character <= 'Z':

            decoded\_string += shift\_character(character, caesar\_key, 'A', 26)

        elif 'a' <= character <= 'z':

            decoded\_string += shift\_character(character, caesar\_key, 'a', 26)

        elif 'А' <= character <= 'Я':

            decoded\_string += shift\_character(character, caesar\_key, 'А', 32)

        elif 'а' <= character <= 'я':

            decoded\_string += shift\_character(character, caesar\_key, 'а', 32)

        elif character == 'Ё':

            decoded\_string += shift\_character('Е', caesar\_key, 'А', 32)

        elif character == 'ё':

            decoded\_string += shift\_character('е', caesar\_key, 'а', 32)

        else:

            decoded\_string += character

    return decoded\_string

def read\_file():

    with open('C:\\Users\\Artem\\Downloads\\lab\_1-20240131T114327Z-001\\lab\_1\\vigenere\\input.txt', 'r', encoding='utf-8') as file:

        return file.read()

def main():

    key = 'key'

    source\_string = read\_file()

    encrypted\_string = string\_encryption\_v(source\_string, key)

    decoded\_string = string\_decoding\_v(encrypted\_string, key)

    print('Source String:', source\_string)

    print('Encrypted String:', encrypted\_string)

    print('Decoded String:', decoded\_string)

if \_\_name\_\_ == '\_\_main\_\_':

    main()