Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Информационные сети. Основы безопасности

ОТЧЁТ

к лабораторной работе №2

на тему

**ШИФРЫ ЦЕЗАРЯ И ВИЖЕНЕРА**

Выполнил: студент гр.253504

Фроленко К.Ю.

Проверил: ассистент кафедры информатики Герчик А.В.

Минск 2025

СОДЕРЖАНИЕ

[1  Формулировака задачи 3](#_Toc189566878)

[2 Ход работы 4](#_Toc189566879)

[Заключение 6](#_Toc189566880)

1 ФОРМУЛИРОВКА ЗАДАЧИ

В данной работе необходимо разработать программное средство для шифрования и дешифрования текстовых файлов с использованием Шифра Цезаря и Шифра Виженера. Реализация должна обеспечивать возможность обработки текстовых данных в удобной форме, позволяя пользователю выбирать метод шифрования, указывать параметры кодирования и получать результат в наглядном виде.

Шифр Цезаря должен позволять задавать величину сдвига, которая определяет, на сколько позиций будет смещен каждый символ исходного текста в алфавите. Для Шифра Виженера необходимо реализовать механизм работы с ключом, который определяет последовательность сдвигов для шифрования и расшифрования текста. Оба метода должны поддерживать работу с русским и английским алфавитами, учитывать регистр символов и корректно обрабатывать неалфавитные символы, оставляя их без изменений.

Программа должна обеспечивать возможность загрузки текста из файлов и сохранения обработанных данных, что позволит избежать необходимости ввода текста вручную. Графический интерфейс должен предоставлять удобные средства выбора алгоритма, задания параметров и управления процессом обработки данных. Важно, чтобы пользователь мог интуитивно взаимодействовать с приложением, легко переключаясь между методами шифрования и дешифрования, а также выбирать необходимые параметры без сложных настроек.

Дополнительно необходимо предусмотреть обработку ошибок, связанных с некорректными входными данными, и реализовать систему уведомлений для информирования пользователя о ходе выполнения операций. Реализация должна быть удобной, надежной и обеспечивать корректную работу алгоритмов в различных сценариях использования.

2 ХОД РАБОТЫ

Для реализации программного средства шифрования и дешифрования текстовых файлов разработано графическое приложение, позволяющее пользователю выбирать метод шифрования, задавать параметры, загружать текстовые файлы и сохранять обработанные данные. Приложение поддерживает два алгоритма: Шифр Цезаря и Шифр Виженера, а также работу с текстами на русском и английском языках.

Графический интерфейс включает элементы для выбора метода шифрования, переключения между режимами шифрования и дешифрования, задания параметров, загрузки входного файла и сохранения результата. Визуальное представление интерфейса программы приведено на Рисунке 1.

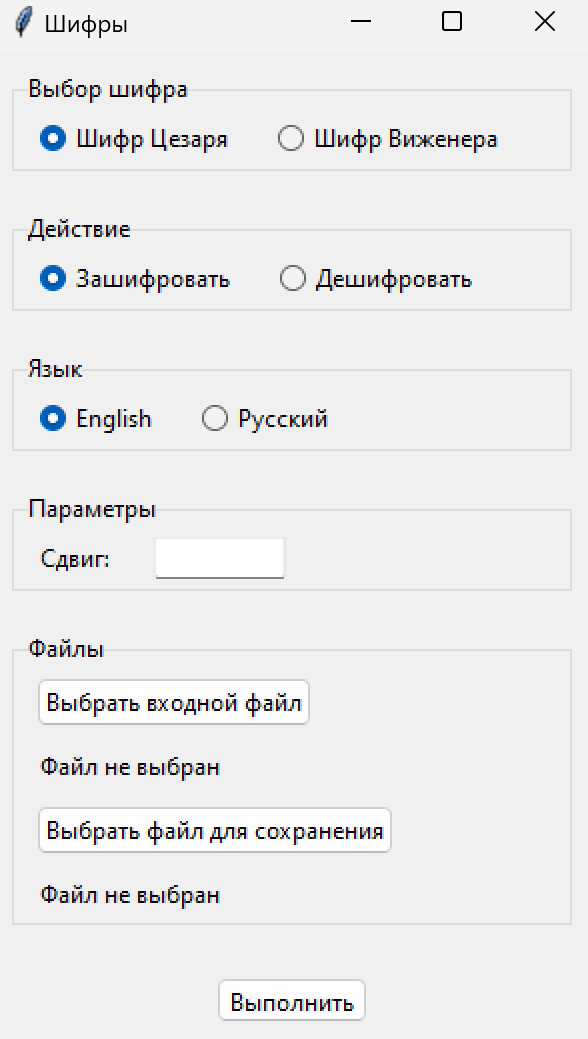


Рисунок 1 – Интерфейс программы для работы с шифрами

В основе работы Шифра Цезаря лежит сдвиг символов входного текста на указанное пользователем количество позиций в пределах выбранного алфавита. Для шифрования и дешифрования текста разработаны функции *caesar\_encrypt* и *caesar\_decrypt*, выполняющие преобразование символов. В процессе обработки текста символы латинского или кириллического алфавита изменяются в соответствии с указанным сдвигом, а остальные символы, такие как цифры, пробелы и знаки препинания, остаются неизменными.

Шифр Виженера использует заданное пользователем ключевое слово, определяющее последовательность сдвигов для каждого символа текста. Реализованы функции *vigenere\_encrypt* и *vigenere\_decrypt*, которые выполняют кодирование и декодирование с учетом ключа. Данный метод позволяет использовать более сложный способ шифрования, так как каждый символ текста заменяется другим символом с учетом соответствующего сдвига, вычисляемого на основе ключевого слова.

Для удобства работы с текстовыми файлами в программу добавлены функции загрузки входного текста из файла и сохранения обработанного текста в новый файл. Это позволяет пользователю работать с шифрованием и дешифрованием без необходимости ввода текста вручную.

На Рисунке 2 представлен пример работы программы. Исходный текст был зашифрован с помощью Шифра Виженера с заданным ключевым словом, а затем успешно расшифрован обратно, что подтверждает корректность работы алгоритмов.

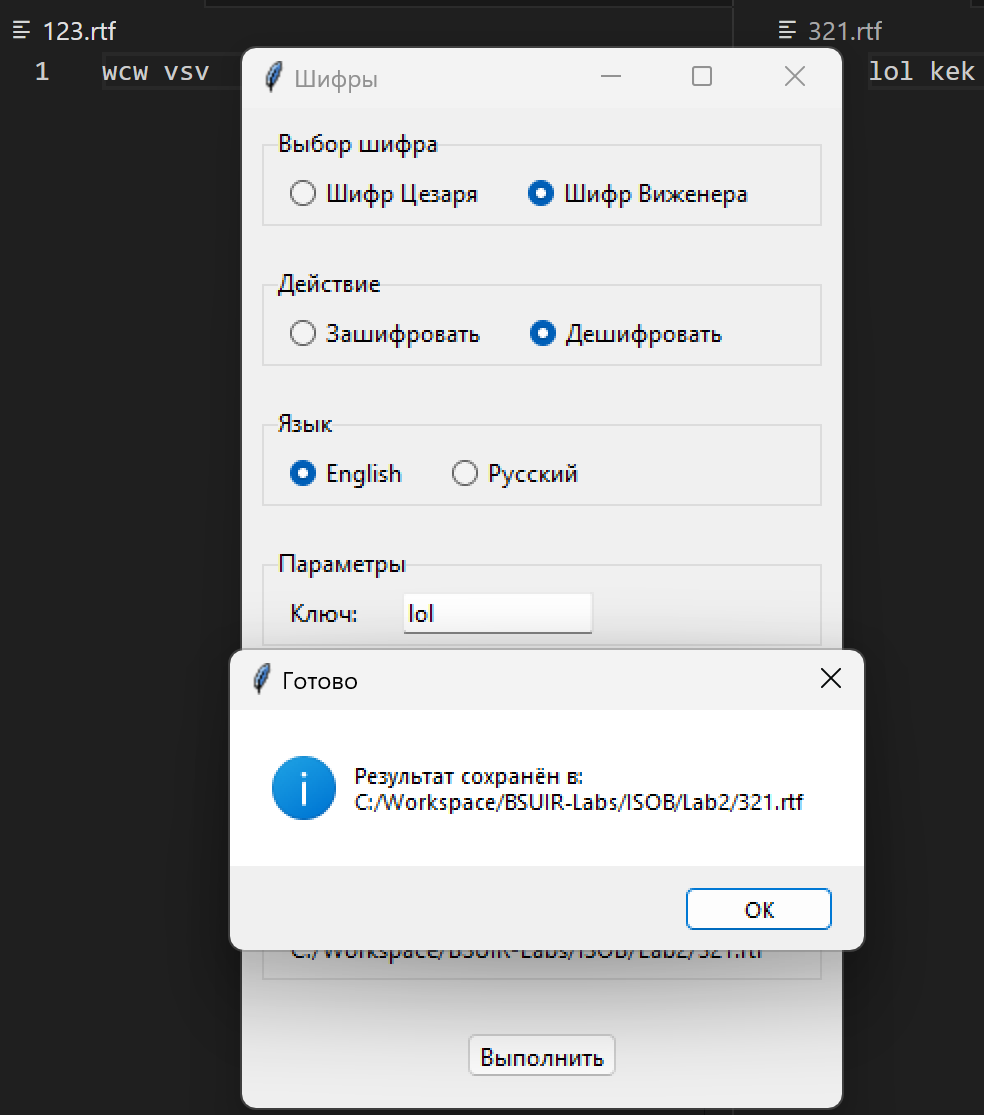


Рисунок 2 – Пример работы программы

# ЗАКЛЮЧЕНИЕ

В ходе данной работы было разработано программное средство для шифрования и дешифрования текстовых файлов с использованием Шифра Цезаря и Шифра Виженера. Реализация обеспечила возможность выбора метода шифрования, задания параметров, загрузки и сохранения текстовых данных, а также интуитивно понятное управление процессом обработки.

На первом этапе была определена архитектура программы, выбраны основные алгоритмы и реализован графический интерфейс, позволяющий пользователю взаимодействовать с программой без необходимости работы с командной строкой. Далее был разработан алгоритм Шифра Цезаря, который выполняет сдвиг символов в пределах алфавита на заданное пользователем число позиций, а затем добавлена поддержка Шифра Виженера, использующего ключевое слово для формирования последовательности сдвигов. Оба метода корректно работают с русским и английским языками, учитывают регистр символов и сохраняют неизменными неалфавитные символы.

Для удобства работы реализована возможность загрузки входного текста из файла и сохранения результата в отдельный файл. В процессе тестирования проверена работоспособность обоих методов шифрования и дешифрования, корректность обработки различных языков и символов, а также работоспособность программы при вводе различных параметров. Дополнительно была добавлена система уведомлений, позволяющая информировать пользователя об ошибках и ходе выполнения операций.

Таким образом, поставленная задача была успешно выполнена. Программа позволяет эффективно выполнять шифрование и расшифровку текстовых данных, корректно обрабатывает входные параметры и обеспечивает удобство работы за счет графического интерфейса. Полученные результаты подтверждают надежность реализации и соответствие требованиям, предъявленным к функционалу программного средства.

ПРИЛОЖЕНИЕ А

**(обязательное)**

**Листинг программного кода**

import tkinter as tk

from tkinter import filedialog, messagebox

from tkinter import ttk

EN\_UPPER = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

EN\_LOWER = EN\_UPPER.lower()

RU\_UPPER = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"

RU\_LOWER = RU\_UPPER.lower()

def get\_alphabets(language):

if language == "English":

return EN\_UPPER, EN\_LOWER

else:

return RU\_UPPER, RU\_LOWER

def caesar\_encrypt(text, shift, language):

alph\_upper, alph\_lower = get\_alphabets(language)

len\_alpha = len(alph\_upper)

result = ""

for char in text:

if char in alph\_upper:

index = alph\_upper.find(char)

new\_index = (index + shift) % len\_alpha

result += alph\_upper[new\_index]

elif char in alph\_lower:

index = alph\_lower.find(char)

new\_index = (index + shift) % len\_alpha

result += alph\_lower[new\_index]

else:

result += char

return result

def caesar\_decrypt(text, shift, language):

return caesar\_encrypt(text, -shift, language)

def generate\_key(text, key, language):

alph\_upper, alph\_lower = get\_alphabets(language)

key\_filtered = "".join([ch for ch in key if ch in alph\_upper or ch in alph\_lower])

if not key\_filtered:

return ""

expanded\_key = ""

key\_index = 0

for char in text:

if char in alph\_upper or char in alph\_lower:

expanded\_key += key\_filtered[key\_index % len(key\_filtered)]

key\_index += 1

else:

expanded\_key += char

return expanded\_key

def vigenere\_encrypt(text, key, language):

alph\_upper, alph\_lower = get\_alphabets(language)

len\_alpha = len(alph\_upper)

expanded\_key = generate\_key(text, key, language)

if not expanded\_key:

return text

cipher\_text = ""

for t\_char, k\_char in zip(text, expanded\_key):

if t\_char in alph\_upper:

shift = alph\_upper.find(k\_char.upper())

new\_index = (alph\_upper.find(t\_char) + shift) % len\_alpha

cipher\_text += alph\_upper[new\_index]

elif t\_char in alph\_lower:

shift = alph\_upper.find(k\_char.upper())

new\_index = (alph\_lower.find(t\_char) + shift) % len\_alpha

cipher\_text += alph\_lower[new\_index]

else:

cipher\_text += t\_char

return cipher\_text

def vigenere\_decrypt(text, key, language):

alph\_upper, alph\_lower = get\_alphabets(language)

len\_alpha = len(alph\_upper)

expanded\_key = generate\_key(text, key, language)

if not expanded\_key:

return text

original\_text = ""

for t\_char, k\_char in zip(text, expanded\_key):

if t\_char in alph\_upper:

shift = alph\_upper.find(k\_char.upper())

new\_index = (alph\_upper.find(t\_char) - shift + len\_alpha) % len\_alpha

original\_text += alph\_upper[new\_index]

elif t\_char in alph\_lower:

shift = alph\_upper.find(k\_char.upper())

new\_index = (alph\_lower.find(t\_char) - shift + len\_alpha) % len\_alpha

original\_text += alph\_lower[new\_index]

else:

original\_text += t\_char

return original\_text

def read\_file(filename):

try:

with open(filename, 'r', encoding="utf-8") as f:

return f.read()

except Exception as e:

messagebox.showerror("Ошибка чтения файла", str(e))

return None

def write\_file(filename, text):

try:

with open(filename, 'w', encoding="utf-8") as f:

f.write(text)

except Exception as e:

messagebox.showerror("Ошибка записи файла", str(e))

class CipherApp(tk.Tk):

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

super().\_\_init\_\_()

self.title("Шифры")

self.geometry("300x500")

self.minsize(300, 500)

self.cipher\_type = tk.StringVar(value="Цезаря")

self.action = tk.StringVar(value="Зашифровать")

self.language = tk.StringVar(value="English")

self.shift\_value = tk.StringVar()

self.vigenere\_key = tk.StringVar()

self.input\_filename = tk.StringVar(value="Файл не выбран")

self.output\_filename = tk.StringVar(value="Файл не выбран")

self.create\_widgets()

def create\_widgets(self):

frame\_cipher = ttk.LabelFrame(self, text="Выбор шифра")

frame\_cipher.pack(padx=10, pady=10, fill="x")

rb\_caesar = ttk.Radiobutton(frame\_cipher, text="Шифр Цезаря", variable=self.cipher\_type, value="Цезаря", command=self.toggle\_params)

rb\_vigenere = ttk.Radiobutton(frame\_cipher, text="Шифр Виженера", variable=self.cipher\_type, value="Виженера", command=self.toggle\_params)

rb\_caesar.pack(side="left", padx=10, pady=5)

rb\_vigenere.pack(side="left", padx=10, pady=5)

frame\_action = ttk.LabelFrame(self, text="Действие")

frame\_action.pack(padx=10, pady=10, fill="x")

rb\_encrypt = ttk.Radiobutton(frame\_action, text="Зашифровать", variable=self.action, value="Зашифровать")

rb\_decrypt = ttk.Radiobutton(frame\_action, text="Дешифровать", variable=self.action, value="Дешифровать")

rb\_encrypt.pack(side="left", padx=10, pady=5)

rb\_decrypt.pack(side="left", padx=10, pady=5)

frame\_language = ttk.LabelFrame(self, text="Язык")

frame\_language.pack(padx=10, pady=10, fill="x")

rb\_en = ttk.Radiobutton(frame\_language, text="English", variable=self.language, value="English")

rb\_ru = ttk.Radiobutton(frame\_language, text="Русский", variable=self.language, value="Русский")

rb\_en.pack(side="left", padx=10, pady=5)

rb\_ru.pack(side="left", padx=10, pady=5)

self.frame\_params = ttk.LabelFrame(self, text="Параметры")

self.frame\_params.pack(padx=10, pady=10, fill="x")

self.label\_shift = ttk.Label(self.frame\_params, text="Сдвиг:")

self.entry\_shift = ttk.Entry(self.frame\_params, textvariable=self.shift\_value, width=10)

self.label\_key = ttk.Label(self.frame\_params, text="Ключ:")

self.entry\_key = ttk.Entry(self.frame\_params, textvariable=self.vigenere\_key, width=15)

self.toggle\_params()

frame\_files = ttk.LabelFrame(self, text="Файлы")

frame\_files.pack(padx=10, pady=10, fill="x")

btn\_input = ttk.Button(frame\_files, text="Выбрать входной файл", command=self.select\_input\_file)

btn\_input.pack(padx=10, pady=5, anchor="w")

self.label\_input = ttk.Label(frame\_files, textvariable=self.input\_filename)

self.label\_input.pack(padx=10, pady=5, anchor="w")

btn\_output = ttk.Button(frame\_files, text="Выбрать файл для сохранения", command=self.select\_output\_file)

btn\_output.pack(padx=10, pady=5, anchor="w")

self.label\_output = ttk.Label(frame\_files, textvariable=self.output\_filename)

self.label\_output.pack(padx=10, pady=5, anchor="w")

btn\_process = ttk.Button(self, text="Выполнить", command=self.process\_file)

btn\_process.pack(padx=10, pady=15)

def toggle\_params(self, \*args):

for widget in self.frame\_params.winfo\_children():

widget.pack\_forget()

cipher = self.cipher\_type.get()

if cipher == "Цезаря":

self.label\_shift.pack(side="left", padx=10, pady=5)

self.entry\_shift.pack(side="left", padx=10, pady=5)

else:

self.label\_key.pack(side="left", padx=10, pady=5)

self.entry\_key.pack(side="left", padx=10, pady=5)

def select\_input\_file(self):

filename = filedialog.askopenfilename(title="Выберите входной файл", filetypes=[("Все файлы", "\*.\*")])

if filename:

self.input\_filename.set(filename)

def select\_output\_file(self):

filename = filedialog.asksaveasfilename(title="Выберите файл для сохранения", defaultextension=".txt", filetypes=[("Все файлы", "\*.\*")])

if filename:

self.output\_filename.set(filename)

def process\_file(self):

in\_file = self.input\_filename.get()

out\_file = self.output\_filename.get()

if in\_file == "Файл не выбран" or out\_file == "Файл не выбран":

messagebox.showwarning("Внимание", "Пожалуйста, выберите входной и выходной файлы.")

return

text = read\_file(in\_file)

if text is None:

return

cipher = self.cipher\_type.get()

action = self.action.get()

lang = self.language.get()

result = ""

if cipher == "Цезаря":

try:

shift = int(self.shift\_value.get())

except ValueError:

messagebox.showerror("Ошибка", "Введите корректное числовое значение для сдвига.")

return

if action == "Зашифровать":

result = caesar\_encrypt(text, shift, lang)

else:

result = caesar\_decrypt(text, shift, lang)

else:

key = self.vigenere\_key.get().strip()

if not key:

messagebox.showerror("Ошибка", "Введите ключ для шифра Виженера.")

return

alph\_upper, alph\_lower = get\_alphabets(lang)

if any(ch not in alph\_upper and ch not in alph\_lower for ch in key):

messagebox.showerror("Ошибка", "Ключ должен состоять только из букв выбранного языка.")

return

if action == "Зашифровать":

result = vigenere\_encrypt(text, key, lang)

else:

result = vigenere\_decrypt(text, key, lang)

write\_file(out\_file, result)

messagebox.showinfo("Готово", f"Результат сохранён в:\n{out\_file}")

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

app = CipherApp()

app.mainloop()