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

**ЭЛЕМЕНТЫ КРИПТОГРАФИИ**

|  |
| --- |
| Выполнил: студент гр. 253503  Кудош А.С. |
| Проверил: ассистент кафедры информатики Герчик А.В. |

Минск 2025

**СОДЕРЖАНИЕ**

[1 Постановка задачи 3](#_Toc189743996)

[2 Реализация программного средства 4](#_Toc189743997)

[Заключение 6](#_Toc189743998)

[Список использованных источников 7](#_Toc189743999)

[Приложение А (обязательное) Исходный код программы 8](#_Toc189744000)

**1 ПОСТАНОВКА ЗАДАЧИ**

Постановка задачи для лабораторной работы заключается в разработке программного обеспечения, способного выполнять шифрование и дешифрование текстовых файлов с использованием двух классических методов криптографии: шифра Цезаря и шифра Виженера. Шифр Цезаря представляет собой один из самых простых и известных методов шифрования, основанный на сдвиге символов алфавита на фиксированное число позиций. Шифр Виженера, в свою очередь, является более сложным и безопасным методом, использующим ключевое слово для определения величины сдвига каждой буквы текста. Цель работы заключается в реализации функциональности, позволяющей пользователю выбирать метод шифрования, задавать ключ и выполнять операции шифрования и дешифрования над текстовыми файлами. Программа должна корректно обрабатывать текстовые данные, сохраняя регистр символов и игнорируя неалфавитные символы, такие как пробелы, цифры и знаки препинания. Кроме того, необходимо обеспечить возможность выбора входного и выходного файлов, а также обработку возможных ошибок, таких как неверный формат ключа или отсутствие файла. Результатом работы должно стать приложение с графическим интерфейсом, разработанное с использованием библиотеки *Tkinter*, которое предоставляет пользователю удобный способ взаимодействия с функционалом шифрования и дешифрования. Приложение должно быть интуитивно понятным и обеспечивать возможность выбора метода шифрования, режима работы и ввода ключа, а также отображать сообщения об успешном выполнении операций или возникших ошибках. В рамках лабораторной работы также предполагается анализ особенностей реализации каждого из методов шифрования, их преимуществ и недостатков, а также демонстрация работы программы на конкретных примерах.

**2 РЕАЛИЗАЦИЯ ПРОГРАММНОГО СРЕДСТВА**

Программное средство разработано на языке *Python* с использованием библиотеки *Tkinter* для создания графического интерфейса. Архитектура приложения включает два основных модуля: функциональный, отвечающий за алгоритмы шифрования и дешифрования, и интерфейсный, обеспечивающий взаимодействие с пользователем.

Для реализации шифра Цезаря созданы функции *caesar\_encrypt* и *caesar\_decrypt*. Алгоритм работает на основе сдвига символов исходного текста на заданное ключом число позиций в алфавите [1]. Регистр символов сохраняется: для букв верхнего регистра используется базовый код символа «*A*», для нижнего – «*a*». Неалфавитные символы (пробелы, цифры, знаки препинания) остаются без изменений. Например, при сдвиге на 3 символ «*A*» превращается в «*D*», а «*z*» – в «*c*».

Шифр Виженера реализован в функциях *vigenere\_encrypt* и *vigenere\_decrypt*. Ключевое слово повторяется до длины исходного текста, после чего для каждого символа вычисляется индивидуальный сдвиг на основе позиции соответствующей буквы ключа в алфавите. Например, для ключа «*KEY*» и текста «*HELLO*» ключ расширяется до «*KEYKE*». Символ «*H*» шифруется сдвигом на десять (буква «*K*» в алфавите), что даёт результат «*R*» [2]. Регистр символов и обработка неалфавитных символов аналогичны шифру Цезаря.

Графический интерфейс включает следующие элементы (рисунок 2.1):

1. Поля для выбора входного и выходного файлов с кнопками «Обзор», реализованными через модуль *filedialog*.
2. Переключатели для выбора метода шифрования (Цезарь или Виженер) и режима работы (шифрование/дешифрование).
3. Поле ввода ключа, валидация которого зависит от выбранного метода: для Цезаря ключ должен быть целым числом, для Виженера – строкой из букв.
4. Кнопка «Обработать», запускающая функцию *process\_file*.

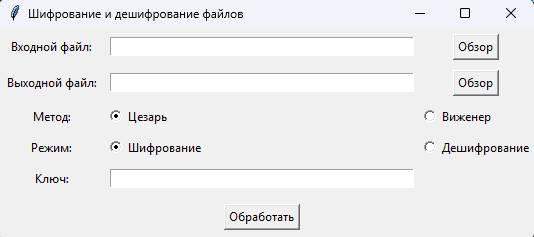


Рисунок 2.1 – Графический интерфейс

Логика работы программы:

1. Пользователь выбирает входной файл и указывает путь для сохранения результата.
2. Выбирает метод шифрования и режим (шифрование/дешифрование).
3. Вводит ключ, который автоматически проверяется на корректность.
4. При нажатии кнопки «Обработать» данные считываются из файла, применяется выбранный алгоритм, результат сохраняется в указанный файл.

Обработка ошибок включает проверку наличия входного файла, корректности формата ключа и вывод соответствующих сообщений через диалоговые окна *messagebox*. Например, при вводе нечислового ключа для Цезаря или наличия цифр в ключе для Виженера программа уведомит пользователя об ошибке. Результат успешной операции сопровождается информационным сообщением с путём к выходному файлу.

Интерфейс спроектирован интуитивно: элементы сгруппированы логически, а подсказки в диалоговых окнах помогают избежать некорректных действий. Программа обеспечивает сохранение структуры исходного текста, включая пробелы и спецсимволы, что делает её применимой для обработки разноформатных данных.

Исходный код программы приведен в приложении А.

**ЗАКЛЮЧЕНИЕ**

В ходе выполнения лабораторной работы было разработано программное средство, реализующее функциональность шифрования и дешифрования текстовых файлов с использованием шифра Цезаря и шифра Виженера. Программа успешно решает поставленные задачи, предоставляя пользователю удобный графический интерфейс для выбора метода шифрования, режима работы и ввода ключа. Реализованные алгоритмы корректно обрабатывают текстовые данные, сохраняя регистр символов и игнорируя неалфавитные символы, что обеспечивает универсальность применения программы для различных типов текстов. Шифр Цезаря, несмотря на свою простоту, демонстрирует эффективность для базовых задач шифрования, в то время как шифр Виженера, благодаря использованию ключевого слова, обеспечивает более высокий уровень безопасности и устойчивость к частотному анализу. Программа включает механизмы обработки ошибок, такие как проверка корректности ключа и наличия входного файла, что повышает её надёжность и удобство использования. Графический интерфейс, разработанный с использованием библиотеки *Tkinter*, интуитивно понятен и позволяет пользователю легко взаимодействовать с функционалом приложения. В результате работы было создано программное средство, которое может быть использовано для изучения основ криптографии, а также для практического применения в задачах защиты текстовой информации. Разработанное приложение демонстрирует возможности *Python* для создания кроссплатформенных решений с графическим интерфейсом и может быть расширено за счёт добавления новых алгоритмов шифрования или улучшения существующих функций.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Шифр Цезаря или как просто зашифровать текст [Электронный ресурс]. – Режим доступа: https://habr.com/ru/articles/534058/.

[2] Криптоанализ шифра Виженера. Как реализовать и взломать [Электронный ресурс]. – Режим доступа: https://habr.com/ru/articles/876764/.

ПРИЛОЖЕНИЕ А  
(обязательное)  
Исходный код программы

import tkinter as tk

from tkinter import filedialog, messagebox

def caesar\_encrypt(text: str, shift: int) -> str:

result = []

for char in text:

if char.isalpha():

base = ord('A') if char.isupper() else ord('a')

shifted = (ord(char) - base + shift) % 26

result.append(chr(base + shifted))

else:

result.append(char)

return ''.join(result)

def caesar\_decrypt(text: str, shift: int) -> str:

return caesar\_encrypt(text, -shift)

def vigenere\_encrypt(text: str, key: str) -> str:

key = key.upper()

key\_index = 0

result = []

for char in text:

if char.isalpha():

base = ord('A') if char.isupper() else ord('a')

key\_shift = ord(key[key\_index % len(key)]) - ord('A')

shifted = (ord(char) - base + key\_shift) % 26

result.append(chr(base + shifted))

key\_index += 1

else:

result.append(char)

return ''.join(result)

def vigenere\_decrypt(text: str, key: str) -> str:

key = key.upper()

key\_index = 0

result = []

for char in text:

if char.isalpha():

base = ord('A') if char.isupper() else ord('a')

key\_shift = ord(key[key\_index % len(key)]) - ord('A')

shifted = (ord(char) - base - key\_shift) % 26

result.append(chr(base + shifted))

key\_index += 1

else:

result.append(char)

return ''.join(result)

def process\_file():

input\_file = input\_file\_entry.get()

output\_file = output\_file\_entry.get()

method = method\_var.get()

mode = mode\_var.get()

key = key\_entry.get()

try:

with open(input\_file, 'r', encoding='utf-8') as f:

text = f.read()

if method == 'caesar':

try:

shift = int(key)

except ValueError:

messagebox.showerror("Ошибка", "Ключ для шифра Цезаря должен быть целым числом.")

return

if mode == 'encrypt':

processed = caesar\_encrypt(text, shift)

else:

processed = caesar\_decrypt(text, shift)

elif method == 'vigenere':

if not key.isalpha():

messagebox.showerror("Ошибка", "Ключ для шифра Виженера должен содержать только буквы.")

return

if mode == 'encrypt':

processed = vigenere\_encrypt(text, key)

else:

processed = vigenere\_decrypt(text, key)

else:

messagebox.showerror("Ошибка", "Неизвестный метод шифрования.")

return

with open(output\_file, 'w', encoding='utf-8') as f:

f.write(processed)

messagebox.showinfo("Успех", f"Файл успешно обработан. Результат сохранен в {output\_file}")

except FileNotFoundError:

messagebox.showerror("Ошибка", "Входной файл не найден.")

def browse\_input\_file():

filename = filedialog.askopenfilename()

input\_file\_entry.delete(0, tk.END)

input\_file\_entry.insert(0, filename)

def browse\_output\_file():

filename = filedialog.asksaveasfilename()

output\_file\_entry.delete(0, tk.END)

output\_file\_entry.insert(0, filename)

app = tk.Tk()

app.title("Шифрование и дешифрование файлов")

tk.Label(app, text="Входной файл:").grid(row=0, column=0, padx=5, pady=5)

input\_file\_entry = tk.Entry(app, width=50)

input\_file\_entry.grid(row=0, column=1, padx=5, pady=5)

tk.Button(app, text="Обзор", command=browse\_input\_file).grid(row=0, column=2, padx=5, pady=5)

tk.Label(app, text="Выходной файл:").grid(row=1, column=0, padx=5, pady=5)

output\_file\_entry = tk.Entry(app, width=50)

output\_file\_entry.grid(row=1, column=1, padx=5, pady=5)

tk.Button(app, text="Обзор", command=browse\_output\_file).grid(row=1, column=2, padx=5, pady=5)

tk.Label(app, text="Метод:").grid(row=2, column=0, padx=5, pady=5)

method\_var = tk.StringVar(value="caesar")

tk.Radiobutton(app, text="Цезарь", variable=method\_var, value="caesar").grid(row=2, column=1, sticky=tk.W)

tk.Radiobutton(app, text="Виженер", variable=method\_var, value="vigenere").grid(row=2, column=2, sticky=tk.W)

tk.Label(app, text="Режим:").grid(row=3, column=0, padx=5, pady=5)

mode\_var = tk.StringVar(value="encrypt")

tk.Radiobutton(app, text="Шифрование", variable=mode\_var, value="encrypt").grid(row=3, column=1, sticky=tk.W)

tk.Radiobutton(app, text="Дешифрование", variable=mode\_var, value="decrypt").grid(row=3, column=2, sticky=tk.W)

tk.Label(app, text="Ключ:").grid(row=4, column=0, padx=5, pady=5)

key\_entry = tk.Entry(app, width=50)

key\_entry.grid(row=4, column=1, padx=5, pady=5)

tk.Button(app, text="Обработать", command=process\_file).grid(row=5, column=1, pady=10)

app.mainloop()