|  |  |
| --- | --- |
| **К Г Э У** | МИНИСТЕРСТВО ВЫСШЕГО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ ЭНЕРГЕТИЧЕСКИЙ УНИВЕРСИТЕТ»**  (ФГБОУ ВО «КГЭУ») |

Отчет по лабораторной работе №3

по дисциплине «Информационная безопасность» по теме

«Программная реализация классических алгоритмов шифрования и их

криптоанализа»

**Выполнил:**

студент Тазеев Р.Р.

**Группа:**

ТРП-1-23

**Проверил:**

Доцент ИТИС  
Хуснутдинов Рамиль Миннегаязович

Казань, 2024 г.

**Цель работы:** ознакомление студентов с существующими

криптографическими алгоритмами, используемые для защиты информации,

формирование умений применять основные криптографические стандарты,

протоколы и алгоритмы, формирование навыков прикладной и программной

реализации криптоалгоритмов, а также владения криптографической

терминологией.

**Указание к заданию:** согласно методическому материалу я создал два файла: main.py и functions.py, реализующие методы зашифрования и расшифрования текста, который можно написать в консоли или в файле с его дальнейшим прочтением, также можно указать сдвиг шифра.

**Задание для выполнения:**

1. Добавить в имеющиеся программы дополнительный функционал:

a) вывод статистики по тексту (частота символов, встречаемых как в

исходном, так и зашифрованном тексте).

b) вывод названия метода шифрования, автора и дату разработки программы

c) модуль защиты интеллектуальных прав на программу (запрашивает ключ

на использование программы, отсчитывает срок действия лицензии)

2. Разработать и протестировать программу на языке программирования

Python для шифрования, расшифрования и криптоанализа текста:

a) методом перестановки символов (размер блока от 4 до 8 символов)

b) методом гаммирования (сложения с гаммой, размер гаммы от 4 до 8

символов).

c) методом Виженера (ключевое слово от 4 до 8 символов).

d) шифром Полибия

e) методом Playfair.

f) методом аффинного шифра.

**Ход работы:**

Для реализации дополнительного функционала создаем новый файл additional\_functions.py и пишем соответствующие функции:

**Вывод статистики по тексту**

def text\_statistics(text):

stats = Counter(text)

total\_chars = len(text)

print("Статистика символов в тексте:")

for char, freq in stats.items():

print(f"Символ: '{char}' — Частота: {freq} ({(freq / total\_chars) \* 100:.2f}%)")

print(f"Всего символов: {total\_chars}")

**Вывод общей информации**

def program\_info():

print("Метод шифрования: Caesar Cipher")

print("Автор: Иван Иванов")

print("Дата разработки: 14 ноября 2024 года")

**Модуль защиты интеллектуальных прав на программу**

def license\_check():

valid\_key = "12345-ABCDE"

license\_duration\_days = 30

user\_key = input("Введите лицензионный ключ: ")

if user\_key != valid\_key:

print("[-] Неверный ключ! Доступ запрещен.")

return False

try:

with open("license\_date.txt", "r") as f:

activation\_date = datetime.datetime.strptime(f.read().strip(), "%Y-%m-%d")

except FileNotFoundError:

activation\_date = datetime.datetime.now()

with open("license\_date.txt", "w") as f:

f.write(activation\_date.strftime("%Y-%m-%d"))

current\_date = datetime.datetime.now()

if (current\_date - activation\_date).days > license\_duration\_days:

print("[-] Срок действия лицензии истек.")

return False

print("[+] Лицензия действительна.")

return True

2) Пропишем также и реализацию других методов шифрования:

import string

import random

# a) Метод перестановки символов

def transposition\_cipher(text, block\_size):

if not (4 <= block\_size <= 8):

raise ValueError("Размер блока должен быть от 4 до 8 символов.")

padded\_text = text + ' ' \* ((block\_size - len(text) % block\_size) % block\_size)

blocks = [padded\_text[i:i+block\_size] for i in range(0, len(padded\_text), block\_size)]

encrypted\_text = ''.join([''.join(random.sample(block, len(block))) for block in blocks])

return encrypted\_text

# b) Метод гаммирования (сложение с гаммой)

def gamma\_cipher(text, gamma):

gamma = (gamma \* (len(text) // len(gamma) + 1))[:len(text)]

encrypted\_text = ''.join(chr(ord(c) ^ ord(g)) for c, g in zip(text, gamma))

return encrypted\_text

# c) Шифр Виженера

def vigenere\_cipher(text, keyword, encrypt=True):

keyword = (keyword \* (len(text) // len(keyword) + 1))[:len(text)]

shift = 1 if encrypt else -1

encrypted\_text = ''.join(

chr((ord(c) + shift \* ord(k)) % 256) for c, k in zip(text, keyword)

)

return encrypted\_text

# d) Шифр Полибия

def polybius\_square\_cipher(text):

square = [['A', 'B', 'C', 'D', 'E'],

['F', 'G', 'H', 'I', 'K'],

['L', 'M', 'N', 'O', 'P'],

['Q', 'R', 'S', 'T', 'U'],

['V', 'W', 'X', 'Y', 'Z']]

row\_col\_map = {square[r][c]: f"{r+1}{c+1}" for r in range(5) for c in range(5)}

encrypted\_text = ''.join(row\_col\_map.get(char.upper(), char) for char in text)

return encrypted\_text

# e) Шифр Плейфера

def playfair\_cipher(text, keyword):

keyword = ''.join(sorted(set(keyword), key=keyword.index))

keyword = keyword.replace("J", "I")

alphabet = "ABCDEFGHIKLMNOPQRSTUVWXYZ"

matrix = keyword + ''.join(c for c in alphabet if c not in keyword)

pairs = [(text[i], text[i + 1]) for i in range(0, len(text), 2)]

encrypted\_text = ""

for a, b in pairs:

if a == b:

b = 'X'

a\_index = matrix.index(a)

b\_index = matrix.index(b)

a\_row, a\_col = divmod(a\_index, 5)

b\_row, b\_col = divmod(b\_index, 5)

if a\_row == b\_row:

encrypted\_text += matrix[a\_row \* 5 + (a\_col + 1) % 5] + matrix[b\_row \* 5 + (b\_col + 1) % 5]

elif a\_col == b\_col:

encrypted\_text += matrix[((a\_row + 1) % 5) \* 5 + a\_col] + matrix[((b\_row + 1) % 5) \* 5 + b\_col]

else:

encrypted\_text += matrix[a\_row \* 5 + b\_col] + matrix[b\_row \* 5 + a\_col]

return encrypted\_text

# f) Аффинный шифр

def affine\_cipher(text, a, b, encrypt=True):

if not (0 < a < 26 and 0 <= b < 26):

raise ValueError("a должно быть от 1 до 25, b - от 0 до 25.")

a\_inv = pow(a, -1, 26) if encrypt else a

result = ''

for char in text.upper():

if char in string.ascii\_uppercase:

num = ord(char) - ord('A')

if encrypt:

result += chr(((a \* num + b) % 26) + ord('A'))

else:

result += chr(((a\_inv \* (num - b)) % 26) + ord('A'))

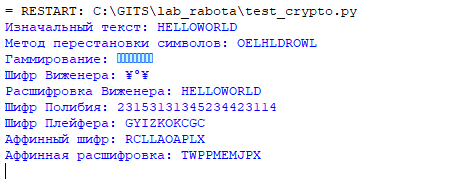
else:

result += char

return result

Код сохранен в crypto\_methods.py и прокомментирован

Вывод при тестировании кода:



Код для тестирования, сохранен в test\_crypto.py:

from crypto\_methods import (

transposition\_cipher, gamma\_cipher, vigenere\_cipher,

polybius\_square\_cipher, playfair\_cipher, affine\_cipher

)

# Тестируем метод перестановки символов

text = "HELLOWORLD"

print(f"Изначальный текст: {text}")

print("Метод перестановки символов:", transposition\_cipher(text, 5))

# Тестируем метод гаммирования

gamma = "GAMMA"

print("Гаммирование:", gamma\_cipher(text, gamma))

# Тестируем шифр Виженера

keyword = "KEY"

print("Шифр Виженера:", vigenere\_cipher(text, keyword))

print("Расшифровка Виженера:", vigenere\_cipher(vigenere\_cipher(text, keyword), keyword, encrypt=False))

# Тестируем шифр Полибия

print("Шифр Полибия:", polybius\_square\_cipher(text))

# Тестируем шифр Плейфера

keyword = "KEYWORD"

print("Шифр Плейфера:", playfair\_cipher(text, keyword))

# Тестируем аффинный шифр

a, b = 5, 8

print("Аффинный шифр:", affine\_cipher(text, a, b))

print("Аффинная расшифровка:", affine\_cipher(affine\_cipher(text, a, b), a, b, encrypt=False))

**Контрольные вопросы**

1. Что представляет собой криптография?

Криптография — это наука и искусство защиты информации с помощью математических алгоритмов, которые обеспечивают конфиденциальность, целостность, аутентификацию и невозвратность данных. В криптографии используется шифрование для превращения открытого текста в шифрованный (зашифрованный) текст, который может быть расшифрован только с использованием определенного ключа или алгоритма.

2. Какие методы криптографического преобразования по виду воздействия на исходную информацию вам известны?

Основные методы криптографического преобразования:

- Шифрование — преобразование открытого текста в зашифрованный текст с помощью ключа и алгоритма.

- Хэширование — преобразование исходной информации в фиксированную строку (хэш), которая не может быть расшифрована обратно в исходный текст.

- Подписи — использование криптографических алгоритмов для создания цифровых подписей, которые подтверждают подлинность документа.

- Маскирование — процесс скрытия части информации, например, замены символов с целью защиты конфиденциальности.

- Секретные каналы связи — способы защищенной передачи информации с использованием криптографических методов.

3. Каким требованиям должны отвечать современные методы шифрования?

Современные методы шифрования должны отвечать следующим требованиям:

-Конфиденциальность — защита данных от несанкционированного доступа.

- Целостность — обеспечение того, чтобы данные не были изменены в процессе передачи или хранения.

- Аутентификация — проверка подлинности источника информации.

- Невозможность восстановления ключа (или криптостойкость) — защита от атак, которые могут привести к вычислению ключа шифрования.

- Эффективность — алгоритм должен работать быстро и с минимальными ресурсами.

- Устойчивость к атакам — метод должен быть защищен от различных атак, таких как атаки по известному открытому тексту, атаки на основе анализа времени, криптоанализ и т.д.

4. В чем заключаются преимущества и недостатки симметричных криптосистем?

Преимущества:

- Скорость — симметричные криптосистемы (например, DES, AES) обычно быстрее, чем асимметричные системы, так как используют одинаковые ключи для шифрования и дешифрования.

- Простота реализации — методы симметричного шифрования легче реализуются в программном и аппаратном обеспечении.

Недостатки:

- Проблема распределения ключей — для каждой пары пользователей требуется отдельный ключ. Это приводит к необходимости безопасно передавать ключи, что является уязвимостью системы.

- Масштабируемость — при увеличении числа пользователей количество ключей растет экспоненциально.

-Отсутствие универсальности — если злоумышленник получит доступ к ключу, вся система безопасности будет скомпрометирована.

5. Какие подвиды симметричных алгоритмов шифрования вы знаете? Дайте им краткую характеристику.

- Блочные шифры — шифруют данные фиксированными блоками. Пример: AES (Advanced Encryption Standard), DES (Data Encryption Standard).

- AES — один из самых распространенных блочных шифров с длиной ключа 128, 192 и 256 бит. Обеспечивает высокую криптостойкость и используется в различных областях.

- DES — старый блочный шифр с 56-битным ключом. Сейчас считается небезопасным из-за возможных атак с использованием вычислительных мощностей.

- Потоковые шифры — шифруют данные по одному биту или байту за раз. Пример: RC4.

- RC4 — потоковый шифр, который долгое время использовался в SSL/TLS, но сейчас признан небезопасным из-за уязвимостей.

- Гибридные схемы — комбинируют блочные и потоковые шифры для достижения лучших характеристик по скорости и безопасности.

6. Почему криптостойкость должна обеспечиваться секретностью ключа, а не секретностью алгоритма шифрования?

Основной принцип криптографии заключается в том, что алгоритм должен быть известен всем, а секретность должна поддерживаться только за счет использования ключа. Это позволяет обеспечить:

- Открытость алгоритмов, что способствует их проверке и улучшению.

- Стандартизацию — алгоритм шифрования может быть широко применим в различных системах.

- Секретность ключа — если ключ остается секретным, даже если алгоритм известен, криптосистема останется безопасной. Это позволяет избежать уязвимости, если алгоритм станет общедоступным.

7. В чем заключается назначение криптоанализа?

Криптоанализ — это наука, занимающаяся изучением и анализом криптографических систем с целью поиска уязвимостей и методов для взлома этих систем. Основные цели криптоанализа:

- Взлом шифра — нахождение метода дешифрования текста без знания ключа.

- Нахождение слабых мест в алгоритмах — выявление уязвимостей, которые могут быть использованы для эффективного взлома системы.

- Оценка криптостойкости — тестирование системы на устойчивость к различным атакам.

8. В чем заключается криптоанализ методом протяжки вероятного слова?

Метод протяжки вероятного слова используется в криптоанализе для поиска ключа шифрования при условии, что криптографический алгоритм использует подстановку. Этот метод включает:

1. Выбор предполагаемого ключевого слова или фразы, которая может быть частью шифрованного текста (например, часто встречаемые слова, такие как "the", "и", "в").

2. Протяжка — процесс подставления предполагаемого слова в разные места зашифрованного текста и проверка полученных результатов.

3. Используя этот метод, криптоаналитик может "выкрутить" шифр, постепенно восстанавливая ключ или другие ключевые элементы алгоритма на основе анализа вероятности появления предполагаемых слов в тексте.

Метод основан на частотном анализе и эвристиках, предполагающих, что в языке есть определенные закономерности и часто повторяющиеся слова.

**Вывод:**

В ходе работы я изучил основные криптографические алгоритмы, такие как перестановка символов, гаммирование, шифры Виженера, Полибия, Плейфера и аффинный шифр. Реализация этих методов позволила мне освоить принципы их работы, терминологию и навыки программной реализации криптоалгоритмов.

**Код программ и отчеты лабораторных работ доступны на моем github:**

**https://github.com/ironsast/kpfu-information-security**