Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчет по лабораторной работе №4

по дисциплине «Информационная безопасность»

по теме: «Криптографическая защита информации»

Выполнил: студент группу ИСТ-22-2б Старков Д.Э.

Проверил: доцент каф. ВММБ Труфанов А.Н.

Пермь, 2025

**Содержание**

[Постановка задачи 3](#_Toc190278942)

[Описание алгоритма 5](#_Toc190278943)

[Графический интерфейс 5](#_Toc190278944)

[Внутренняя логика 5](#_Toc190278945)

[База данных 6](#_Toc190278946)

[Выводы 7](#_Toc190278947)

[Приложение 8](#_Toc190278948)

# Постановка задачи

Разработать и реализовать программу криптографического преобразования текстового файла. Для разработки использовать из перечисленных сред программирования: Turbo Pascal, Delphi, C++ Builder, Visual C++ или другие языки по согласованию с преподавателем.

1. Программа должна поддерживать работу в двух режимах:
   * Криптографическое преобразование текста методом Цезаря;
   * Криптографическое преобразование текста методом Виженера.
2. В режиме криптографического преобразования текста методом Цезаря должны быть реализованы следующие функции:
   * Зашифровать исходный текст из пользовательского файла "<имя файла>.txt" (длиной не менее 2000 символов) методом Цезаря с ключом N, задаваемым пользователем;
   * Сохранить зашифрованный текст в файл "encC\_<имя файла>.txt";
   * Расшифровать текст и сохранить результат в файл "decC\_<имя файла>.txt";
   * Вывести первые строки файлов "<имя файла>.txt", "encC\_<имя файла>.txt" и "decC\_<имя файла>.txt" в текстовое окно.
3. В режиме криптографического преобразования текста методом Виженера должны быть реализованы следующие функции:
   * Предоставить два варианта выбора алфавита замены:
     + Случайным образом;
     + По порядку.
   * Построить и вывести на экран квадрат Виженера;
   * Зашифровать исходный текст из пользовательского файла "<имя файла>.txt" (длиной не менее 2000 символов) методом Виженера с ключом, задаваемым пользователем;
   * Сохранить зашифрованный текст в файл "encV\_<имя файла>.txt";
   * Расшифровать текст и сохранить результат в файл "decV\_<имя файла>.txt";
   * Вывести первые строки файлов "<имя файла>.txt", "encV\_<имя файла>.txt" и "decV\_<имя файла>.txt" в текстовое окно.

# Описание алгоритма

## Шифр Цезаря

Система шифрования Цезаря является частным случаем шифра простой замены. Метод основан на замене каждой буквы сообщения на другую букву того же алфавита путем смещения от исходной буквы на К букв. Реализация продемонстрирована на рисунке №1

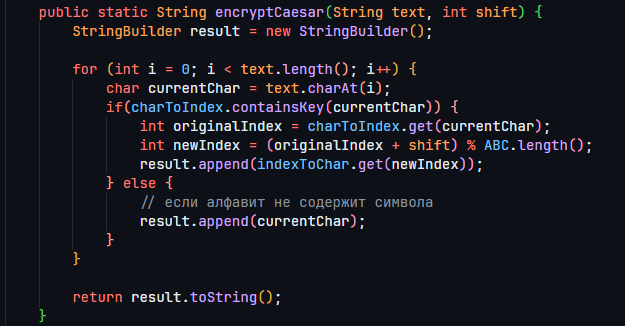


Рис. №1. Алгоритм для шифра Цезаря

Где на вход соответственно поступают текст и число, на которое смещается буква. Переход от символа к индексу в алфавите и обратно осуществляется за счёт двух объектов класса Map. Для дешифрования используется этот же алгоритм только формула для сдвига выглядит следующим образом: , пример на рисунке №2

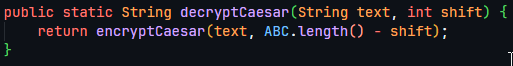


Рис. №2. Алгоритм для дешифрования Цезаря

## Шифр Виженера

Шифр Виженера представляет собой аналогичную многоалфавитную систему шифрования, в которой: В первую строку квадрата заносится перестановка из букв алфавита. Во второй строке та же перестановка циклически сдвигается на одну позицию влево, в третьей строке перестановка сдвигается на две позиции и так далее. Таким образом, квадрат Виженера состоит из N перестановок. Алгоритм представлен на рисунке №3

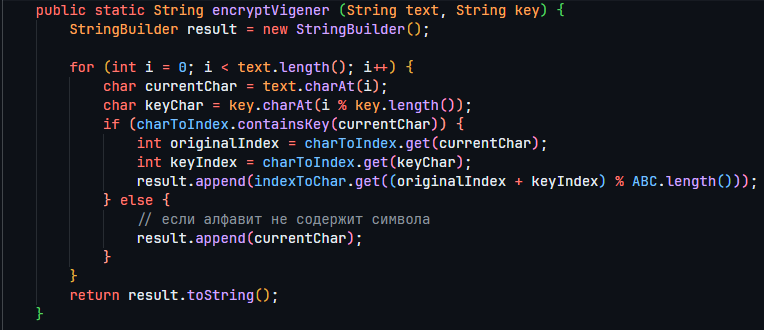


Рис. №3. Алгоритм для шифра Виженера

На вход также поступает текст и ключ, который должен является словом, фразой или текстом. Переход от символа к индексу в алфавите и обратно осуществляется за счёт двух объектов класса Map. Для дешифрования используется аналогичный алгоритм, но сдвиг делается в обратную сторону, пример на рисунке №4



Рис. №4. Алгоритм для дешифрования Виженера

# Выводы

* Ознакомился с основными методами криптографической защиты информации
* Была реализована программа для криптографических преобразований

# Приложение

import java.io.\*;

import java.util.HashMap;

import java.util.Map;

public class Main {

    private static final String ABC = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";

    private static final Map<Character, Integer> charToIndex = new HashMap<>();

    private static final Map<Integer, Character> indexToChar = new HashMap<>();

    static {

        for (int i = 0; i < ABC.length(); i++) {

            charToIndex.put(ABC.charAt(i), i);

            indexToChar.put(i, ABC.charAt(i));

        }

    }

    private static Character[][] squareVigener = new Character[ABC.length()][ABC.length()];

    static {

        for (int i = 0; i < ABC.length(); i++) {

            for (int j = 0; j < ABC.length(); j++) {

                squareVigener[i][j] = ABC.charAt((i + j) % ABC.length());

            }

        }

    }

    public static void main(String[] args) {

        String inputFile = "input.txt";

        String outputFileCaesar = "outputCaesar.txt";

        String outputFileVigener = "outputVigener.txt";

        String returnCaesar = "returnCaesar.txt";

        String returnVigener = "returnVigener.txt";

        // Шифрование

        String input = read(inputFile).toUpperCase();

        // Для Цезаря

        String cipherCaesar = encryptCaesar(input, 6);

        write(outputFileCaesar, cipherCaesar);

        // Для виженера

        String cipherVigener = encryptVigener(input, "МЕЧТА");

        write(outputFileVigener, cipherVigener);

        // Дешифрование

        write(returnCaesar, decryptCaesar(cipherCaesar, 6));

        write(returnVigener, decryptVigener(cipherVigener, "МЕЧТА"));

        // квадрат виженера

        for (int i = 0; i < squareVigener.length; i++) {

            for (int j = 0; j < squareVigener[0].length; j++) {

                System.out.print(squareVigener[i][j] + "\t");

            }

            System.out.println();

        }

    }

    public static String read(String inputFile) {

        String text = "";

        try (BufferedReader bufferedReader = new BufferedReader(new FileReader(inputFile))) {

            String line;

            while ((line = bufferedReader.readLine()) != null) {

                text += line;

            }

        } catch (FileNotFoundException e ) {

            throw new RuntimeException(e);

        } catch (IOException e) {

            throw new RuntimeException(e);

        }

        return text;

    }

    public static void write(String outputFile, String text) {

        try (BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(outputFile))){

            bufferedWriter.write(text);

        } catch (IOException e) {

            throw new RuntimeException(e);

        }

    }

    public static String encryptVigener (String text, String key) {

        StringBuilder result = new StringBuilder();

        for (int i = 0; i < text.length(); i++) {

            char currentChar = text.charAt(i);

            char keyChar = key.charAt(i % key.length());

            if (charToIndex.containsKey(currentChar)) {

                int originalIndex = charToIndex.get(currentChar);

                int keyIndex = charToIndex.get(keyChar);

                result.append(indexToChar.get((originalIndex + keyIndex) % ABC.length()));

            } else {

                // если алфавит не содержит символа

                result.append(currentChar);

            }

        }

        return result.toString();

    }

    public static String decryptVigener (String text, String key) {

        StringBuilder result = new StringBuilder();

        for (int i = 0; i < text.length(); i++) {

            char currentChar = text.charAt(i);

            char keyChar = key.charAt(i % key.length());

            if (charToIndex.containsKey(currentChar)) {

                int originalIndex = charToIndex.get(currentChar);

                int keyIndex = charToIndex.get(keyChar);

                result.append(indexToChar

                        .get(((originalIndex - keyIndex + ABC.length()) % ABC.length())));

            } else {

                // если алфавит не содержит символа

                result.append(currentChar);

            }

        }

        return result.toString();

    }

    public static String encryptCaesar(String text, int shift) {

        StringBuilder result = new StringBuilder();

        for (int i = 0; i < text.length(); i++) {

            char currentChar = text.charAt(i);

            if(charToIndex.containsKey(currentChar)) {

                int originalIndex = charToIndex.get(currentChar);

                int newIndex = (originalIndex + shift) % ABC.length();

                result.append(indexToChar.get(newIndex));

            } else {

                // если алфавит не содержит символа

                result.append(currentChar);

            }

        }

        return result.toString();

    }

    public static String decryptCaesar(String text, int shift) {

        return encryptCaesar(text, ABC.length() - shift);

    }

}