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

на тему

**ШИФР ЦЕЗАРЯ. ШИФР ВИЖЕНЕРА**

Студент: гр.153502

Толстой Д. В.

Проверил: Лещенко Е. А.

Минск 2024

**СОДЕРЖАНИЕ**

[1 Цель работы 3](#_30j0zll)

[2 Проектирование алгоритма 4](#_1fob9te)

[3 Демонстрация программного средства 5](#_3znysh7)

[Приложение А (обязательное) Блок-схема программы шифрования и дешифрования с помощью шифра Вижинера 6](#_2et92p0)

[Приложение Б (обязательное) Блок-схема программы шифрования и дешифрования с помощью шифра Цезаря 7](#_tyjcwt)

[Приложение В (обязательное) Исходный код программы 10](#_4d34og8)

# 1 ЦЕЛЬ РАБОТЫ

Изучить теоретические сведения о шифре Цезаря и шифре Виженера. Реализовать программные средства шифрования и дешифрования текстовых файлов при помощи шифра Цезаря и шифра Виженера.

# 2 ПРОЕКТИРОВАНИЕ АЛГОРИТМА

Функции шифрования и дешифрования с помощью шифра Цезаря и шифра Виженера весьма похожи и отличаются лишь блоком выбора закодированного символа. Блок-схемы алгоритмов представлены в приложениях A-Б.

# 3 ДЕМОНСТРАЦИЯ ПРОГРАММНОГО СРЕДСТВА

В результате выполнения лабораторной работы было получено программное средство, способное считывать данные из файла и шифровать/дешифровать их. Весь код программы был написан на языке *Javascript.*

Локально был создан файл *input.txt* с содержимым представленным на рисунке 1.

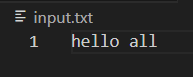


Рисунок 1 – Содержимое файла *input.txt*

При запуске программы происходит считывание файла и вызов соответствующих асинхронных функций *processData* из 2 модулей для каждого метода соответственно*.* Вывод программ после выполнения представлен на рисунках 2 и 3.

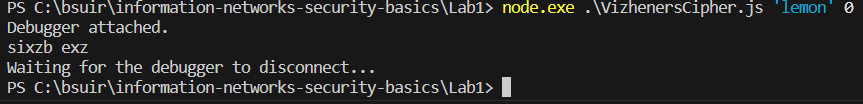


Рисунок 2 – Вывод программы, использующей метод Вижинера

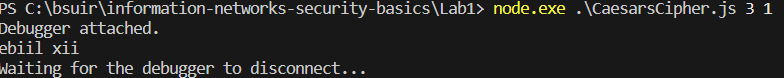
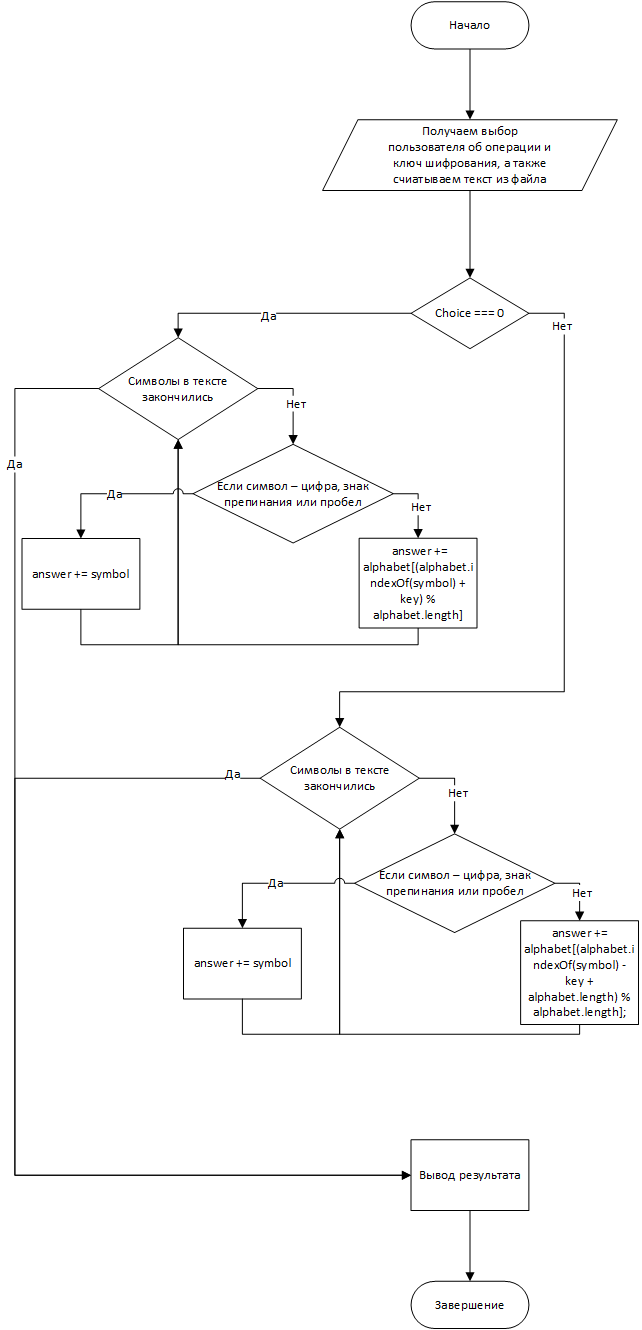


Рисунок 3 – Вывод программы, использующей метод Цезаря

Исходный код программы представлен в приложении В.

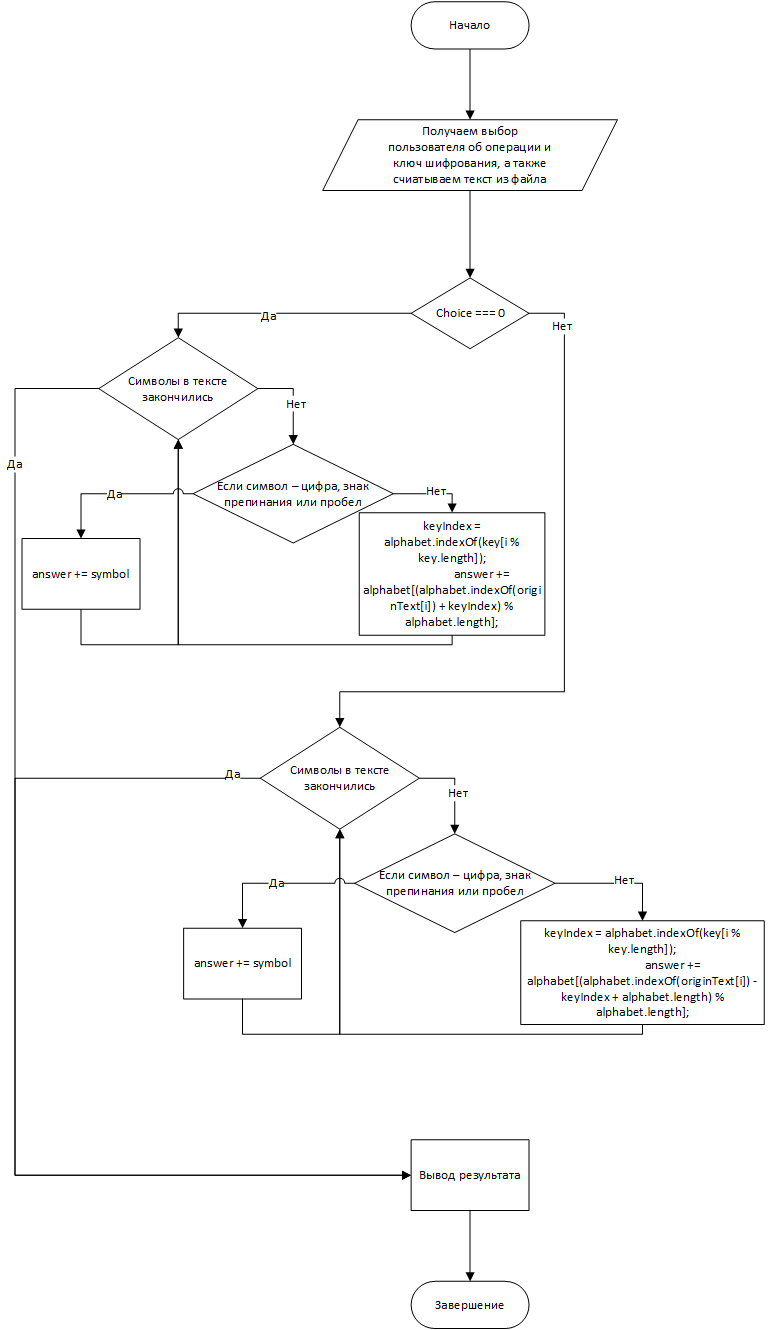
# ПРИЛОЖЕНИЕ А

**Блок-схема функции шифрования и дешифрования с помощью шифра Цезаря**



# ПРИЛОЖЕНИЕ Б

**Блок-схема функции шифрования и дешифрования с помощью шифра Цезаря**



# ПРИЛОЖЕНИЕ В

**Исходный код программы**

**Файл CaesarsСipher.js**

const fs = require('fs').promises;

function isPunctuation(char) {

const punctuationRegex = /[!"#$%&'()\*+,-./:;<=>?@[\]^\_`{|}~]/;

return punctuationRegex.test(char);

}

function isDigit(char) {

return !isNaN(parseInt(char));

}

async function processData() {

const filePath = 'input.txt';

try {

const originText = await fs.readFile(filePath, 'utf-8');

const args = process.argv.slice(2);

const key = parseInt(args[0]);

const choice = args[1];

// Инициализация английского алфавита (строчные буквы)

const alphabet = Array.from({ length: 26 }, (\_, index) => String.fromCharCode('a'.charCodeAt(0) + index));

let answer = '';

if (choice == 0) {

for (const symbol of originText) {

if (symbol === ' ' || isDigit(symbol) || isPunctuation(symbol)) {

answer += symbol;

} else {

answer += alphabet[(alphabet.indexOf(symbol) + key) % alphabet.length];

}

}

} else {

for (const symbol of originText) {

if (symbol === ' ' || isDigit(symbol) || isPunctuation(symbol)) {

answer += symbol;

} else {

answer += alphabet[(alphabet.indexOf(symbol) - key + alphabet.length) % alphabet.length];

}

}

}

console.log(answer);

} catch (error) {

console.error(`Error reading the file: ${error.message}`);

}

}

processData();

**Файл VizhenersCipher.js**

const fs = require('fs').promises;

function isPunctuation(char) {

const punctuationRegex = /[!"#$%&'()\*+,-./:;<=>?@[\]^\_`{|}~]/;

return punctuationRegex.test(char);

}

function isDigit(char) {

return !isNaN(parseInt(char));

}

async function processData() {

const filePath = 'input.txt';

try {

const originText = await fs.readFile(filePath, 'utf-8');

const args = process.argv.slice(2);

const key = args[0];

const choice = args[1];

// Инициализация английского алфавита (строчные буквы)

const alphabet = Array.from({ length: 26 }, (\_, index) => String.fromCharCode('a'.charCodeAt(0) + index));

let answer = '';

if (choice == 0) {

for (let i = 0; i < originText.length; i++) {

if (originText[i] === ' ' || isDigit(originText[i]) || isPunctuation(originText[i])) {

answer += originText[i];

} else {

const keyIndex = alphabet.indexOf(key[i % key.length]);

answer += alphabet[(alphabet.indexOf(originText[i]) + keyIndex) % alphabet.length];

}

}

} else {

for (let i = 0; i < originText.length; i++) {

if (originText[i] === ' ' || isDigit(originText[i]) || isPunctuation(originText[i])) {

answer += originText[i];

} else {

const keyIndex = alphabet.indexOf(key[i % key.length]);

answer += alphabet[(alphabet.indexOf(originText[i]) - keyIndex + alphabet.length) % alphabet.length];

}

}

}

console.log(answer);

} catch (error) {

console.error(`Error reading the file: ${error.message}`);

}

}

processData();