Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №1**

Простые симметричные шифры

тема

Вариант 1

Преподаватель \_\_\_\_\_\_\_\_\_ А.А. Сидарас

подпись, дата инициалы, фамилия

Студент КИ18-17/1б 031831229 \_\_\_\_\_\_\_\_\_ В.А. Прекель

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2020

Содержание

[Содержание 2](#_Toc38494317)

[1 Цель работы с постановкой задачи 3](#_Toc38494318)

[1.1 Цель работы 3](#_Toc38494319)

[1.2 Задача работы 3](#_Toc38494320)

# Цель работы с постановкой задачи

## Цель работы

Ознакомиться с основами симметричного шифрования, ознакомиться с простыми симметричными криптографическими шифрами на основе методов подстановок, перестановок и гаммирования, освоить основные этапы проектирования и реализации симметричных шифров.

## Задача работы

Согласно персональному варианту или индивидуальному заданию преподавателя разработать и составить в виде блок-схемы алгоритмы шифрования и дешифрования текста. Убедиться в правильности составления алгоритмов и затем на языке программирования составить программу, которая реализует данные алгоритмы.

На ряде контрольных примеров (не менее 10) открытого текста, состоящего из различного количества символов, проверить правильность работы алгоритмов шифрования и дешифрования.

Самостоятельно или с помощью преподавателя придумать оригинальный способ модификации шифра с целью повышения его криптостойкости. Внести изменения в исходный алгоритм и программу. Проверить работоспособность алгоритма на тестовых примерах.

Доказать, что предложенный Вами способ модификации действительно повышает криптостокость.

Разработанная Вами программа должна содержать графический интерфейс пользователя.

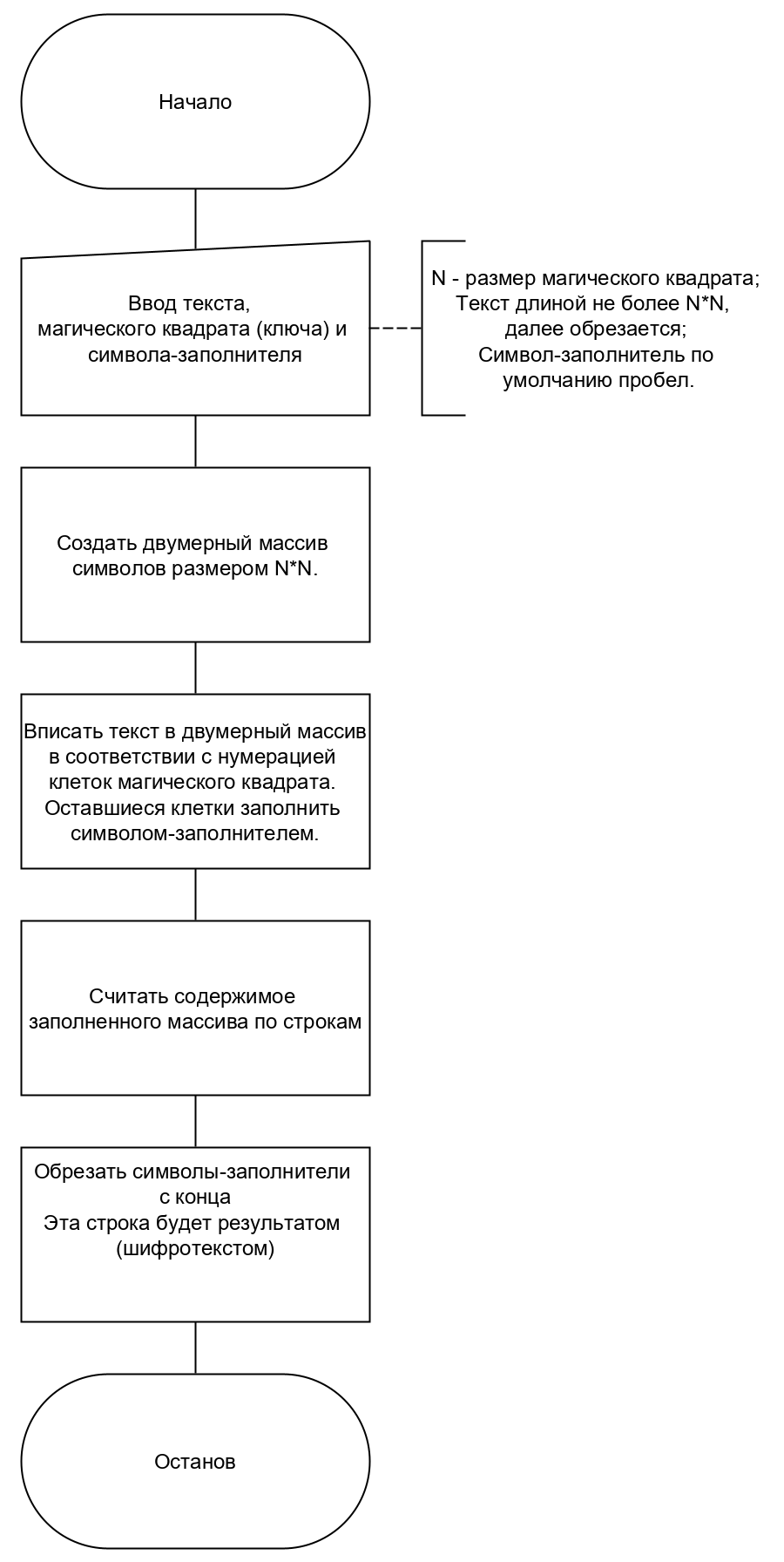
Вариант 1 – Шифр на основе «магических» квадратов размерностью .

# Теоретические сведения

Для шифрования методом магических квадратов требуется квадратная матрица , заполненная различными натуральными числами от 1 до N таким образом, что сумма чисел в каждой строке, каждом столбце и на обеих диагоналях одинакова [1]. Это магический квадрат, и он будет ключом шифрования. При шифровании буквы открытого текста необходимо вписать в магический квадрат в соответствии с нумерацией его клеток. Для получения шифротекста считывают содержимое заполненной таблицы по строкам. [2]

Для взлома шифрования с помощью магических квадратов требуется определить размер магического квадрата, основываясь на длине шифротекста и перебрать магические квадраты этого размера. Поэтому для усложнения взлома можно добавить функцию перевода букв текста в верхний регистр, удаления пробелов или всех не-букв, а также возможность поворота ключа (от чего магичность ключа не меняется). Так же для шифрования достаточно использовать не только магические квадраты, а просто заполненные различными натуральными числами от 1 до N.

# Ход работы



1. Блок-схема алгоритма

Листинг 1 – Lab\_01/Lab\_01.Core/MagicSquare.cs

using System.Text;

namespace Lab\_01.Core

{

/// <summary>

/// Шифрования методом магических квадратов.

/// </summary>

public class MagicSquareCipher

{

/// <summary>

/// Инициализирует новый экземпляр класса шифрования методом магических квадратов.

/// </summary>

/// <param name="key">Ключ (магический квадрат).</param>

/// <param name="emptyChar">Символ, используемый на месте пустых ячеек.</param>

public MagicSquareCipher(MagicSquare key, char emptyChar = ' ')

{

Key = key;

CipherTextMatrix = new char[Key.Count, Key.Count];

EmptyChar = emptyChar;

}

/// <summary>

/// Максимальная и рекомендуемая длина текста, возможная шифрованием данным ключом.

/// </summary>

public int MaxLength => Key.Count \* Key.Count;

/// <summary>

/// Ключ (магический квадрат).

/// </summary>

public MagicSquare Key { get; }

/// <summary>

/// Матрица шифротекста (здесь шифротекст хранится).

/// </summary>

private char[,] CipherTextMatrix { get; }

/// <summary>

/// Текст.

/// </summary>

public string Text { get; private set; }

/// <summary>

/// Символ, используемый на месте пустых ячеек.

/// </summary>

public char EmptyChar { get; set; }

/// <summary>

/// Шифротекст (вычисляется из матрицы шифротекста).

/// </summary>

public string CipherText

{

get

{

var sb = new StringBuilder();

for (var i = 0; i < Key.Count; i++)

{

for (var j = 0; j < Key.Count; j++)

{

sb.Append(CipherTextMatrix[i, j]);

}

}

return sb.ToString();

}

private set

{

for (var i = 0; i < Key.Count; i++)

{

for (var j = 0; j < Key.Count; j++)

{

var index = i \* Key.Count + j;

if (index >= value.Length)

{

CipherTextMatrix[i, j] = EmptyChar;

}

else

{

CipherTextMatrix[i, j] = value[index];

}

}

}

}

}

/// <summary>

/// Зашифровывает текст.

/// </summary>

/// <param name="text">Текст для шифрования.</param>

/// <returns>Шифротекст.</returns>

public string Crypt(string text)

{

Text = text;

for (var i = 0; i < Key.Count; i++)

{

for (var j = 0; j < Key.Count; j++)

{

if (Key[i, j] - 1 >= text.Length)

{

CipherTextMatrix[i, j] = EmptyChar;

}

else

{

CipherTextMatrix[i, j] = text[Key[i, j] - 1];

}

}

}

return CipherText;

}

/// <summary>

/// Расшифровывает текст.

/// </summary>

/// <param name="cipherText">Шифротекст для расшифрованиaя.</param>

/// <returns>Текст.</returns>

public string Encrypt(string cipherText)

{

CipherText = cipherText;

var sb = new StringBuilder(MaxLength);

for (var k = 0; k < MaxLength; k++)

{

for (var i = 0; i < Key.Count; i++)

{

for (var j = 0; j < Key.Count; j++)

{

if (Key[i, j] == k + 1)

{

sb.Append(CipherTextMatrix[i, j]);

}

}

}

}

Text = sb.ToString().TrimEnd(EmptyChar);

return Text;

}

}

}

Листинг 2 – Lab\_01/Lab\_01.Core/MagicSquareCipher.cs

using System.Text;

namespace Lab\_01.Core

{

/// <summary>

/// Шифрования методом магических квадратов.

/// </summary>

public class MagicSquareCipher

{

/// <summary>

/// Инициализирует новый экземпляр класса шифрования методом магических квадратов.

/// </summary>

/// <param name="key">Ключ (магический квадрат).</param>

/// <param name="emptyChar">Символ, используемый на месте пустых ячеек.</param>

public MagicSquareCipher(MagicSquare key, char emptyChar = ' ')

{

Key = key;

CipherTextMatrix = new char[Key.Count, Key.Count];

EmptyChar = emptyChar;

}

/// <summary>

/// Максимальная и рекомендуемая длина текста, возможная шифрованием данным ключом.

/// </summary>

public int MaxLength => Key.Count \* Key.Count;

/// <summary>

/// Ключ (магический квадрат).

/// </summary>

public MagicSquare Key { get; }

/// <summary>

/// Матрица шифротекста (здесь шифротекст хранится).

/// </summary>

private char[,] CipherTextMatrix { get; }

/// <summary>

/// Текст.

/// </summary>

public string Text { get; private set; }

/// <summary>

/// Символ, используемый на месте пустых ячеек.

/// </summary>

public char EmptyChar { get; set; }

/// <summary>

/// Шифротекст (вычисляется из матрицы шифротекста).

/// </summary>

public string CipherText

{

get

{

var sb = new StringBuilder();

for (var i = 0; i < Key.Count; i++)

{

for (var j = 0; j < Key.Count; j++)

{

sb.Append(CipherTextMatrix[i, j]);

}

}

return sb.ToString();

}

private set

{

for (var i = 0; i < Key.Count; i++)

{

for (var j = 0; j < Key.Count; j++)

{

var index = i \* Key.Count + j;

if (index >= value.Length)

{

CipherTextMatrix[i, j] = EmptyChar;

}

else

{

CipherTextMatrix[i, j] = value[index];

}

}

}

}

}

/// <summary>

/// Зашифровывает текст.

/// </summary>

/// <param name="text">Текст для шифрования.</param>

/// <returns>Шифротекст.</returns>

public string Crypt(string text)

{

Text = text;

for (var i = 0; i < Key.Count; i++)

{

for (var j = 0; j < Key.Count; j++)

{

if (Key[i, j] - 1 >= text.Length)

{

CipherTextMatrix[i, j] = EmptyChar;

}

else

{

CipherTextMatrix[i, j] = text[Key[i, j] - 1];

}

}

}

return CipherText;

}

/// <summary>

/// Расшифровывает текст.

/// </summary>

/// <param name="cipherText">Шифротекст для расшифрованиaя.</param>

/// <returns>Текст.</returns>

public string Encrypt(string cipherText)

{

CipherText = cipherText;

var sb = new StringBuilder(MaxLength);

for (var k = 0; k < MaxLength; k++)

{

for (var i = 0; i < Key.Count; i++)

{

for (var j = 0; j < Key.Count; j++)

{

if (Key[i, j] == k + 1)

{

sb.Append(CipherTextMatrix[i, j]);

}

}

}

}

Text = sb.ToString().TrimEnd(EmptyChar);

return Text;

}

}

}

Список использованных источников

1. Магический квадрат — Википедия [Электронный ресурс]. URL: [https://ru.wikipedia.org/wiki/Магический\_квадрат](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BA%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82) (Дата обращения: 22.04.2020).
2. Шифрование по методу магических квадратов [Электронный ресурс]. URL: <http://neudoff.net/info/informatika/shifrovanie-po-metodu-magicheskix-kvadratov/> (Дата обращения: 22.04.2020).