Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Информационные сети. Основы безопасности

ОТЧЕТ

к лабораторной работе №1

на тему

**Шифр Цезаря. Шифр Виженера**

Выполнил:

студент гр. 153502

Бычко В.П.

Проверил:

Лещенко Е.А.

Минск 2024

СОДЕРЖАНИЕ

[Введенеие 3](#_Toc157436533)

[1 Краткие теоретические сведения 4](#_Toc157436534)

[2 Результат выполнения программы 7](#_Toc157436535)

[Приложение А 10](#_Toc157436537)

# ВВЕДЕНИЕ

Шифр – система обратимых преобразований, зависящая от некоторого секретного параметра (ключа) и предназначенная для обеспечения секретности передаваемой информации.

Цель данной лабораторной работы заключается в изучении теоретических сведений по работе Шифра Цезаря и Шифра Виженера и разработки программы, шифрующей и дешифрующей информацию с помощью данных шифров.

# 1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Шифр Цезаря, также известный, как шифр сдвига, код Цезаря или сдвиг Цезаря – один из самых простых и наиболее широко известных методов шифрования.

Шифр Цезаря – это вид шифра подстановки, в котором каждый символ в открытом тексте заменяется символом находящимся на некотором постоянном числе позиций левее или правее него в алфавите. Например, в шифре со сдвигом 4 А была бы заменена на Г, Б станет Д, и так далее.

Шифр назван в честь римского императора Гая Юлия Цезаря, использовавшего его для секретной переписки со своими генералами.

Шаг шифрования, выполняемый шифром Цезаря, часто включается как часть более сложных схем, таких как шифр Виженера, и все ещё имеет современное приложение в системе ROT13. Как и все моноалфавитные шифры, шифр Цезаря легко взламывается и не имеет практически никакого применения на практике.



Рисунок 1 – Шифр Цезаря

Если сопоставить каждому символу алфавита его порядковый номер (нумеруя с 0), то шифрование и дешифрование можно выразить формулами модульной арифметики:

y=(x+k)\ \mod\ n

x=(y-k+n)\ \mod\ n,

где ~x – символ открытого текста, ~y – символ шифрованного текста, ~n – мощность алфавита, а ~k – ключ.

С точки зрения математики шифр Цезаря является частным случаем аффинного шифра.

Шифр Виженера состоит из последовательности нескольких шифров Цезаря с различными значениями сдвига. Для зашифровывания может использоваться таблица алфавитов, называемая tabula recta или квадрат (таблица) Виженера. Применительно к латинскому алфавиту таблица Виженера составляется из строк по 26 символов, причём каждая следующая строка сдвигается на несколько позиций. Таким образом, в таблице получается 26 различных шифров Цезаря. На каждом этапе шифрования используются различные алфавиты, выбираемые в зависимости от символа ключевого слова. Например, предположим, что исходный текст имеет такой вид: ATTACKATDAWN

Человек, посылающий сообщение, записывает ключевое слово («LEMON») циклически до тех пор, пока его длина не будет соответствовать длине исходного текста: LEMONLEMONLE

Первый символ исходного текста A зашифрован последовательностью L, которая является первым символом ключа. Первый символ L шифрованного текста находится на пересечении строки L и столбца A в таблице Виженера. Точно так же для второго символа исходного текста используется второй символ ключа; то есть второй символ шифрованного текста X получается на пересечении строки E и столбца T. Остальная часть исходного текста шифруется подобным способом.

Исходный текст: ATTACKATDAWN; Ключ: LEMONLEMONLE; Зашифрованный текст: LXFOPVEFRNHR.

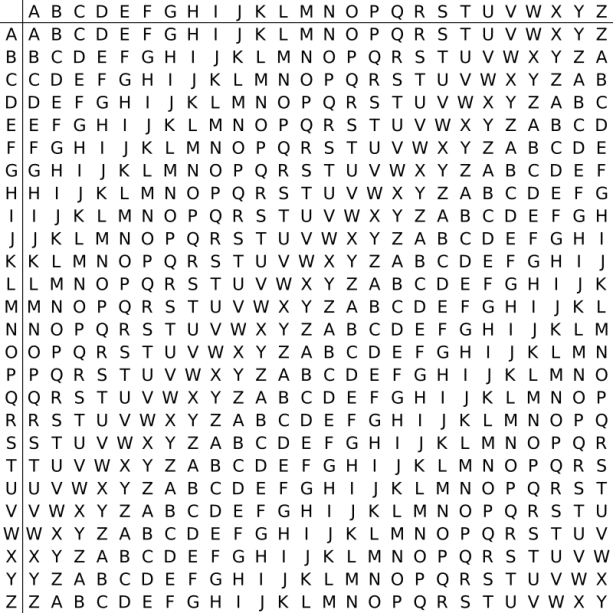


Рисунок 2 – Квадрат Виженера

Расшифровывание производится следующим образом: находим в таблице Виженера строку, соответствующую первому символу ключевого слова; в данной строке находим первый символ зашифрованного текста. Столбец, в котором находится данный символ, соответствует первому символу исходного текста. Следующие символы зашифрованного текста расшифровываются подобным образом.

Если n — количество букв в алфавите, m j — буквы открытого текста, k j — буквы ключа, то шифрование Виженера можно записать следующим образом:c j = m j + k j ( mod n )

И расшифровывание:m j = c j – k j ( mod n )

В компьютере такая операция соответствует сложению кодов ASCII символов сообщения и ключа по некоторому модулю. Кажется, что если таблица будет более сложной, чем циклическое смещение строк, то шифр станет надежнее. Это действительно так, если ее менять чаще, например, от слова, к слову. Но составление таких таблиц, представляющих собой латинские квадраты.

# 2 РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ ПРОГРАММЫ

В результате разработки программы было создано консольное приложение, осуществляющее шифровку и дешифровку информации из файлов по заданным ключам.

На рисунке 2.1 представлена блок-схема алгоритма для работы с шифром Цезаря.

На рисунке 2.2 представлена блок-схема алгоритма для работы с шифром Виженера.

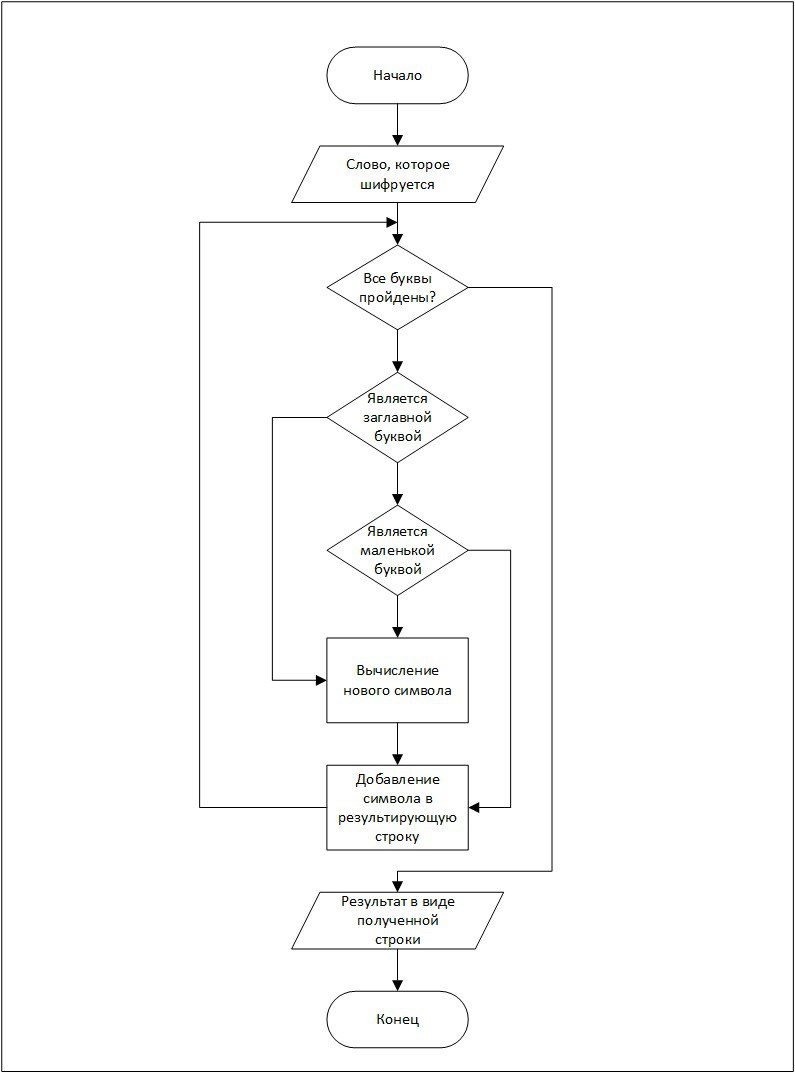


Рисунок 2.1 – Блок-схема алгоритма для работы с шифром Цезаря

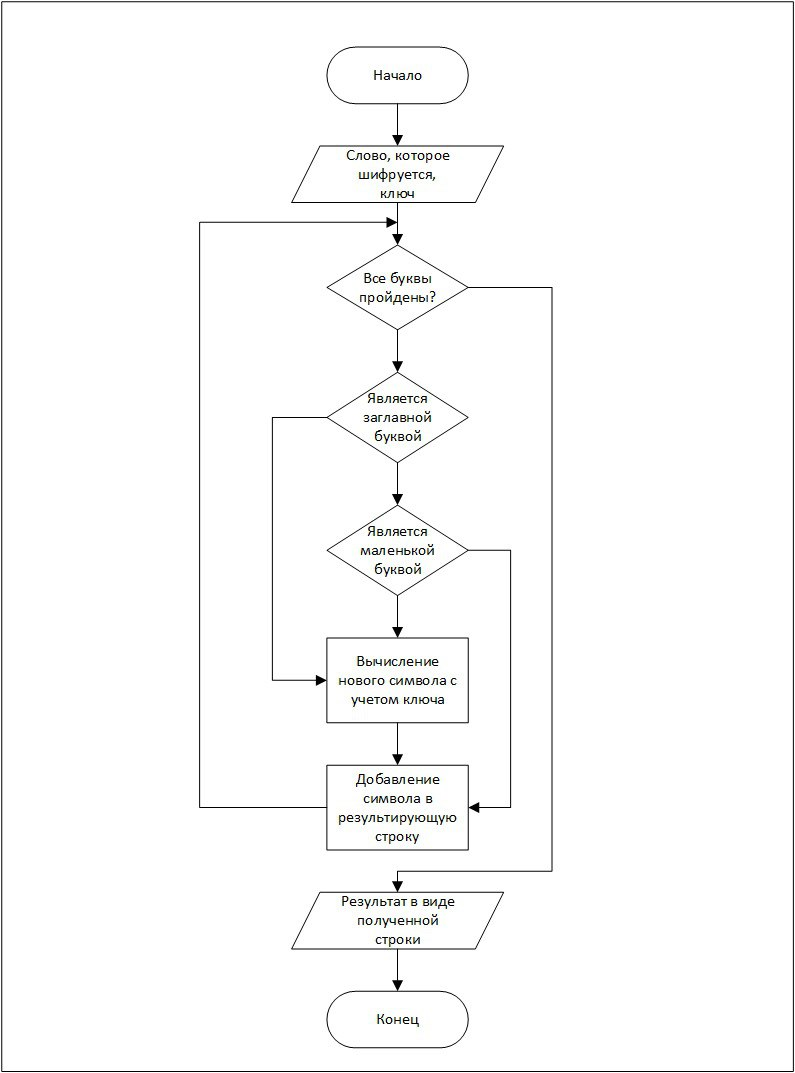


Рисунок 2.2 – Блок-схема алгоритма для работы с шифром Виженера

Результат выполнения программы изображён на рисунке 2.3.

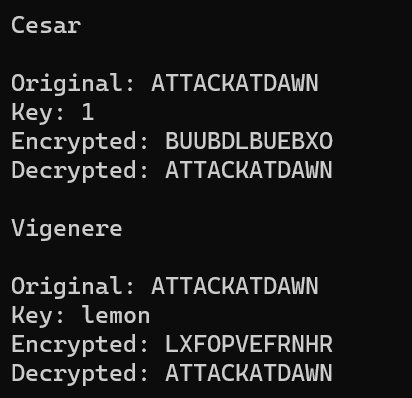


Рисунок 2.3 – Результат программы

# ПРИЛОЖЕНИЕ А

**(обязательное)**

**Исходный код программы**

**Program.cs**

namespace Lab1;

public class Program

{

private static void Main(string[] args)

{

string path = @"datafile.txt";

string fileData = ReadFile(path);

TestCesar(fileData);

Console.WriteLine();

TestVigenère(fileData);

}

private static void TestCesar(string str)

{

Console.WriteLine("Cesar\n");

Console.WriteLine($"Original: {str}");

int key = 1;

Console.WriteLine($"Key: {key}");

var encrypted = CaesarCipher.Encrypt(str, key);

Console.WriteLine($"Encrypted: {encrypted}");

var decrypted = CaesarCipher.Decrypt(encrypted, key);

Console.WriteLine($"Decrypted: {decrypted}");

}

private static void TestVigenère(string str)

{

Console.WriteLine("Vigenère\n");

Console.WriteLine($"Original: {str}");

string key = "lemon";

Console.WriteLine($"Key: {key}");

var encrypted = VigenereCipher.Encrypt(str, key);

Console.WriteLine($"Encrypted: {encrypted}");

var decrypted = VigenereCipher.Decrypt(encrypted, key);

Console.WriteLine($"Decrypted: {decrypted}");

}

private static string ReadFile(string path)

{

return File.ReadAllText(path);

}

}

**Ciphers.cs**

namespace Lab1

{

public static class CaesarCipher

{

public static string Encrypt(string input, int key)

{

char[] chars = new char[input.Length];

for (int i = 0; i < input.Length; i++)

{

chars[i] = EncryptChar(input[i], key);

}

return new string(chars);

}

public static string Decrypt(string input, int key)

{

return Encrypt(input, -key);

}

private static char EncryptChar(char letter, int key)

{

if (IsEnglishUpper(letter))

{

return ShiftLetter(letter, 'A', 26, key);

}

else if (IsEnglishLower(letter))

{

return ShiftLetter(letter, 'a', 26, key);

}

else if (IsRussianUpper(letter))

{

return ShiftLetter(letter, 'А', 32, key);

}

else if (IsRussianLower(letter))

{

return ShiftLetter(letter, 'а', 32, key);

}

else

{

return letter;

}

}

private static char ShiftLetter(char letter, char baseChar, int alphabetSize, int key)

{

return (char)(Mod(letter - baseChar + key, alphabetSize) + baseChar);

}

public static bool IsEnglishUpper(char letter) => letter >= 'A' && letter <= 'Z';

public static bool IsEnglishLower(char letter) => letter >= 'a' && letter <= 'z';

public static bool IsRussianUpper(char letter) => letter >= 'А' && letter <= 'Я';

public static bool IsRussianLower(char letter) => letter >= 'а' && letter <= 'я';

private static int Mod(int x, int m) => ((x %= m) < 0) ? x + m : x;

}

}

**VigenereCipher**

namespace Lab1

{

public static class VigenereCipher

{

public static string Encrypt(string input, string key)

{

char[] chars = new char[input.Length];

for (int i = 0; i < chars.Length; i++)

{

chars[i] = EncryptChar(input[i], GetStringBias(key, i));

}

return new string(chars);

}

public static string Decrypt(string input, string key)

{

char[] chars = new char[input.Length];

for (int i = 0; i < chars.Length; i++)

{

chars[i] = DecryptChar(input[i], GetStringBias(key, i));

}

return new string(chars);

}

private static char EncryptChar(char letter, int key)

{

return CaesarCipher.Encrypt(letter.ToString(), key)[0];

}

private static char DecryptChar(char letter, int key)

{

return CaesarCipher.Decrypt(letter.ToString(), key)[0];

}

private static int GetStringBias(string str, int index)

{

var c = str[index % str.Length];

return GetAlphabetBias(c);

}

private static int GetAlphabetBias(char letter)

{

if (CaesarCipher.IsEnglishUpper(letter))

return letter - 'A';

else if (CaesarCipher.IsEnglishLower(letter))

return letter - 'a';

else if (CaesarCipher.IsRussianUpper(letter))

return letter - 'А';

else if (CaesarCipher.IsRussianLower(letter))

return letter - 'а';

else

return -1;

}

}

}