МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Лабораторная работа № 4**

**ИССЛЕДОВАНИЕ КРИПТОГРАФИЧЕСКИХ**

**ШИФРОВ НА ОСНОВЕ ПОДСТАНОВКИ**

**(ЗАМЕНЫ) СИМВОЛОВ**

Разработала: Некрасова А.П.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

**Цель:** изучение и приобретение практических навыков разработки и использования приложений для реализации подстановочных шифров.

**Задачи**:

1. Закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости подстановочных шифров.
2. Ознакомиться с особенностями реализации и свойствами различных подстановочных шифров на основе готового программного средства (L\_LUX).
3. Разработать приложение для реализации указанных преподавателем методов подстановочного зашифрования/расшифрования.
4. Выполнить исследование криптостойкости шифров на основе статистических данных о частотах появления символов в исходном и зашифрованном сообщениях.
5. Оценить скорость зашифрования/расшифрования реализованных способов шифров.
6. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

**Практическое задание**

|  |  |  |
| --- | --- | --- |
| Вариант | Алфавит | Шифр |
| 9 | Немецкий | 1. На основе соотношений (2.1) и (2.2); k = 7 2. Таблица Трисемуса, ключевое слово – enigma |

***Представление немецкого алфавита:***

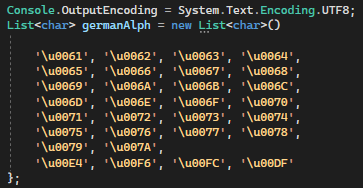
******

Рис. 1 – Представление немецкого алфавита

1. ***Моноалфавитные шифры подстановки.***

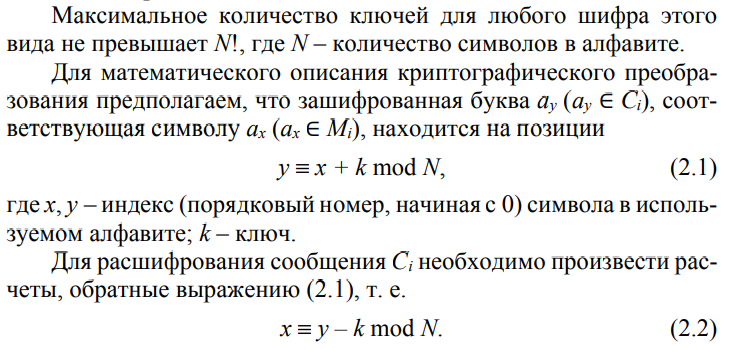
******

Рис. 2 – Соотношения (2.1) и (2.2)

Для реализации моноалфавитного шифра подстановки на основе соотношений (2.1) и (2.2) был разработан класс **EncoderK**. Листинг данного класса представлен ниже.

class EncoderK

{

public List<char> alphabet;

public int k;

public string editedAlphabet;

public EncoderK(List<char> alphabet, int k)

{

this.alphabet = alphabet;

this.k = k;

this.editedAlphabet = editAlphabet(alphabet,k);

}

public string editAlphabet(List<char> alphabet,int k)

{

StringBuilder newAlphabet = new StringBuilder();

for(int iter = 0; iter < alphabet.Count;iter++)

{

newAlphabet.Append(alphabet[(iter + k) % alphabet.Count]);

}

return newAlphabet.ToString();

}

public void printEditedAlphabet()

{

Console.WriteLine($"\nАлфавит для шифрования:");

foreach (char x in this.editedAlphabet)

{

Console.Write(x); Console.Write(" ");

}

}

public string encode(string text)

{

StringBuilder encodedText = new StringBuilder();

for(int iter = 0;iter <text.Length;iter++)

{

int pos = this.alphabet.IndexOf(text[iter]);

char encSymbol = this.editedAlphabet[pos];

encodedText.Append(encSymbol);

}

return encodedText.ToString();

}

public string decode(string text)

{

StringBuilder decodedText = new StringBuilder();

for (int iter = 0; iter < text.Length; iter++)

{

int pos = this.editedAlphabet.IndexOf(text[iter]);

char decSymbol = this.alphabet[pos];

decodedText.Append(decSymbol);

}

return decodedText.ToString();

}

}

Листинг 1 – **EncoderK.cs**

Реализация шифрования и дешифрования моноалфавитным шифром подстановки при *k* = 7:

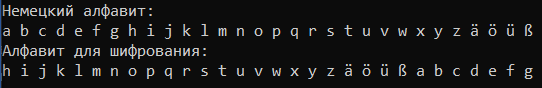
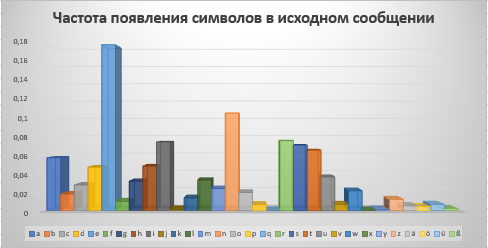


Рис. 3 – Алфавит для шифрования

Табл. 1 – Результаты выполнения. Моноалфавитный шифр подстановки

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исходный текст | Зашифрованный текст | Время шифрования | Расшифрованный текст | Время расшифрования |
| derkaktusdasbemerkenswertesteanderzeitistdasssiesoreinrelativistdeme… | klyrhräözkhziltlyrluzßlyälzälhuklyclpäpzäkhzzzplzvylpuylshäpüpzäkltl… | 8,28E-05 sec | derkaktusdasbemerkenswertesteanderzeitistdasssiesoreinrelativistdeme… | 0,0001693 sec |

Гистограммы появления символов:



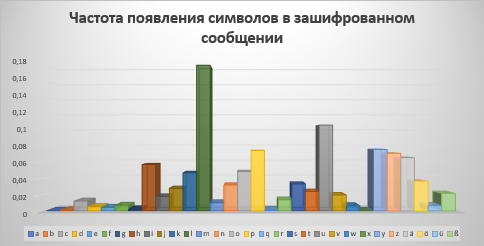


Рис. 4 – Гистограммы появления символов

1. ***Таблица Трисемуса.***

Шифрование методом Трисемуса осуществляется с помощью таблицы и ключевого слова. Таблица состоит из всех элементов алфавита, но важен порядок их расположения. Первым нужно записать ключевое слово, не учитывая повторы символов (каждый символ алфавита должен быть занесен в таблицу единожды!), а после необходимо дописать все не вошедшие в ключевое слово символы по порядку их расположения в алфавите. Размерность таблицы можно выбрать произвольно. В моем случае – это таблица 5x6.

Весь функционал для шифрования методом Трисемуса представлен в классе EncoderTrisemus. Листинг данного класса:

class EncoderTrisemus

{

public List<char> alphabet;

public char[,] tableAlphabet;

public string keyWord;

public int height;

public int width;

public EncoderTrisemus(List<char> alphabet, string keyWord)

{

this.alphabet = alphabet;

this.tableAlphabet = makeTable(alphabet,keyWord);

this.keyWord = keyWord;

}

public char[,] makeTable(List<char> alphabet,string keyWord)

{

this.height = 5;

this.width = 6;

int k = 0;

int heightA = 0,widthA = 0;

char[,] finalTable = new char[height, width];

for(int iterY = 0; iterY < height;iterY++)

{

if (k >= keyWord.Length)

{

heightA = iterY;

break;

}

for (int iterX = 0; iterX < width;iterX++)

{

if(k < keyWord.Length)

{

if(!containSymbol(finalTable, keyWord[k],height,width))

{

finalTable[iterY, iterX] = keyWord[k++];

}

}

else

{

widthA = iterX;

break;

}

}

}

for(int iter = 0;iter < alphabet.Count(); iter++)

{

if(widthA >= width)

{

heightA++;

widthA = 0;

}

if(heightA < height)

{

if (!containSymbol(finalTable, alphabet[iter], height, width))

{

finalTable[heightA, widthA++] = alphabet[iter];

}

}

else

{

break;

}

}

return finalTable;

}

private bool containSymbol(char[,] table, char symbol,int height, int width)

{

for (int iterY = 0; iterY < height; iterY++)

{

for (int iterX = 0; iterX < width; iterX++)

{

if (table[iterY, iterX] == symbol)

{

return true;

}

else

{

continue;

}

}

}

return false;

}

public string encode(string text)

{

StringBuilder stringFinal = new StringBuilder();

foreach(char x in text)

{

for(int iterY = 0; iterY < height;iterY++)

{

for (int iterX = 0; iterX < width; iterX++)

{

if(x == tableAlphabet[iterY, iterX])

{

if(iterY + 1 < height)

{

if(tableAlphabet[iterY + 1, iterX] != '\0')

stringFinal.Append(tableAlphabet[iterY + 1, iterX]);

else

stringFinal.Append(tableAlphabet[0, iterX]);

}

else

{

stringFinal.Append(tableAlphabet[0, iterX]);

}

}

}

}

}

return stringFinal.ToString();

}

public string decode(string text)

{

StringBuilder stringFinal = new StringBuilder();

foreach (char x in text)

{

for (int iterY = 0; iterY < height; iterY++)

{

for (int iterX = 0; iterX < width; iterX++)

{

if (x == tableAlphabet[iterY, iterX])

{

if (iterY - 1 >= 0)

{

if (tableAlphabet[iterY - 1, iterX] != '\0')

stringFinal.Append(tableAlphabet[iterY - 1, iterX]);

else

stringFinal.Append(tableAlphabet[height - 1, iterX]);

}

else

{

stringFinal.Append(tableAlphabet[height - 1, iterX]);

}

}

}

}

}

return stringFinal.ToString();

}

public void printMatrix(char[,] input)

{

for (int w = 0; w < height ; w++)

{

for (int i = 0; i < width; i++)

{

Console.Write($"{(input[w, i] == ' ' ? '\*' : input[w, i])} ");

}

Console.Write($"\n");

}

Console.Write($"\n");

}

}

Листинг 2 – **EncoderTrisemus.cs**

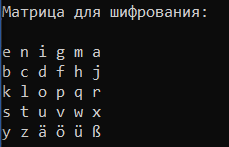


Рис. 5 – Таблица Трисемуса

Табл. 2 – Результаты выполнения. Таблица Трисемуса

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исходный текст | Зашифрованный текст | Время шифрования | Расшифрованный текст | Время расшифрования |
| derkaktusdasbemerkenswertesteanderzeitistdasssiesoreinrelativistdeme… | obxsjszäyojykbhbxsbcyübxzbyzbjcobxnbdzdyzojyyydbyuxbdcxbtjzdödyzobhb… | 0,0008324 sec | derkaktusdasbemerkenswertesteanderzeitistdasssiesoreinrelativistdeme… | 0,0018125 sec |

Гистограммы появления символов:

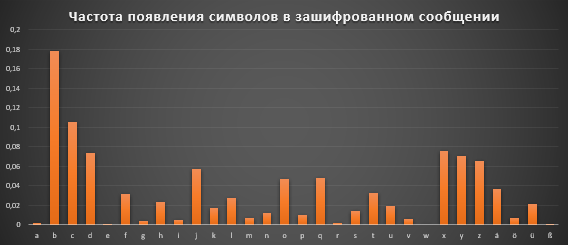
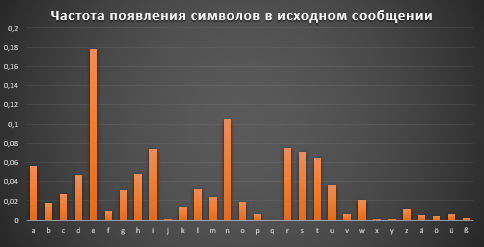


Рис. 6 – Гистограммы появления символов

Зависимость времени, затраченного на шифрование/дешифрование от количества символов в сообщении:

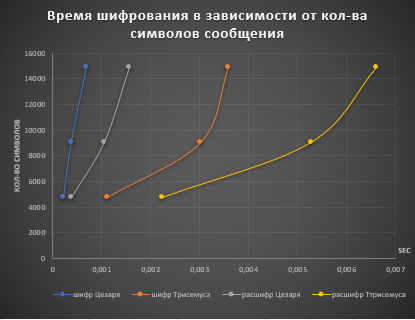


Рис. 7 – Гистограмма зависимости времени от кол-ва символов

**Вывод:** в данной лабораторной работе были закреплены теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости шифров. Были ознакомлены с особенностями реализации и свойствами различных подстановочных шифров. Было разработано приложение для реализации указанных преподавателем методов подстановочного зашифрования/расшифрования. Были проведены исследования криптостойкости шифров на основе статистических данных о частотах появления символов в исходном и зашифрованном сообщениях.