Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

Лабораторная работа №5

по дисциплине "Администрирование и безопасность информационных систем"

ТЕМА РАБОТЫ:

Шифрование методом перестановки

Выполнил:

студент гр. ПРИ-120

Парахин К.В.

Принял:

Доцент кафедры ИСПИ

Курочкин С.В.

Владимир 2023 г.

Цель работы:

Выполнить шифрование заданного сообщения простейшим шифром перестановок и выполнить проверку правильности шифрования.

Выполнение работы:

Индивидуальный вариант № 9.

Задание

Зашифровать последним методом следующий текст:

«Двадцать первое. Ночь. Понедельник. Очертанья столицы во мгле. Сочинил же какой-то бездельник, что бывает любовь на земле. И от лености или от скуки все поверили, так и живут: ждут свиданий, боятся разлуки и любовные песни поют.»

При шифровании используйте ключ: (9 % 4 == 1)

1 вариант - «головоломка»;

Листинг программы на языке C#:

public class Crypter

{

public Crypter(string key, string alphabet = null!)

{

\_currentAlphabet = !string.IsNullOrEmpty(alphabet)

? alphabet

: DEFAULT\_ALPHABET.ToLower();

\_key = key;

\_numbers = new int[\_key.Length];

\_alphabeticNumbers = new int[\_key.Length];

}

public string Process(string message, bool toEncrypt = true)

{

SortAlphabetAndNumbers();

var newMessage = BuildNewMessage(message, out int count);

BuildLettersMatrix(newMessage, count);

var encryptedMessage = BuildEncryptedMessage(count);

return encryptedMessage;

}

private void SortAlphabetAndNumbers()

{

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

{

\_numbers[i] = i;

for (int j = 0; j < \_currentAlphabet.Length; j++)

{

if (\_currentAlphabet[j] == \_key[i])

{

\_alphabeticNumbers[i] = j;

break;

}

}

}

for (int i = 0; i < \_key.Length - 1; i++)

{

for (int j = 0; j < \_key.Length - i - 1; j++)

{

if (\_alphabeticNumbers[j + 1] < \_alphabeticNumbers[j])

{

int temp = \_alphabeticNumbers[j + 1];

int tempIndex = \_numbers[j + 1];

\_alphabeticNumbers[j + 1] = \_alphabeticNumbers[j];

\_numbers[j + 1] = \_numbers[j];

\_alphabeticNumbers[j] = temp;

\_numbers[j] = tempIndex;

}

}

}

}

private int CalculateRowsCount(string message)

{

var n = Math.DivRem(message.Length, \_key.Length, out int remainder);

return remainder > 0 ? ++n : n;

}

private string BuildNewMessage(string message, out int count)

{

string newMessage = "";

foreach (var symbol in message)

{

if (\_currentAlphabet.Contains(symbol))

{

newMessage += symbol;

}

}

count = CalculateRowsCount(newMessage);

var adds = count \* \_key.Length - newMessage.Length;

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

{

newMessage += " ";

}

return newMessage;

}

private void BuildLettersMatrix(string newMessage, int count)

{

\_letters = new List<string>[count];

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

{

\_letters[i] = new List<string>();

for (int j = i \* \_key.Length; j < (i + 1) \* \_key.Length; j++)

{

\_letters[i].Add(newMessage[j].ToString());

}

}

}

private string BuildEncryptedMessage(int count)

{

string encryptedMessage = "";

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

{

for (int j = 0; j < count; j++)

{

var symbol = \_letters[j].ElementAt(\_numbers[i]);

if (symbol != " ")

{

encryptedMessage += \_letters[j].ElementAt(\_numbers[i]);

}

}

if (i != \_key.Length - 1)

{

encryptedMessage += " ";

}

}

return encryptedMessage;

}

public string Encrypt(string message)

=> Process(message.ToLower(), toEncrypt: true);

private readonly string \_currentAlphabet;

private readonly string \_key;

private List<string>[] \_letters;

private int[] \_numbers;

private int[] \_alphabeticNumbers;

private const string DEFAULT\_ALPHABET = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";

}

public class Program

{

public static void Main()

{

Console.OutputEncoding = System.Text.Encoding.UTF8;

Console.WriteLine("Введите ключ для шифрования");

var key = Console.ReadLine();

Console.WriteLine("Введите сообщение, которое требуется зашифровать");

var message = Console.ReadLine();

Console.WriteLine();

if (!string.IsNullOrWhiteSpace(key) && !string.IsNullOrEmpty(message))

{

var crypter = new Crypter(key);

var encryptedMessage = crypter.Encrypt(message);

Console.WriteLine($"Зашифрованное сообщение - {encryptedMessage}");

}

else

{

Console.WriteLine("На вход поступил пустой ключ или пустое сообщение");

}

}

}

Тестирование приложения:

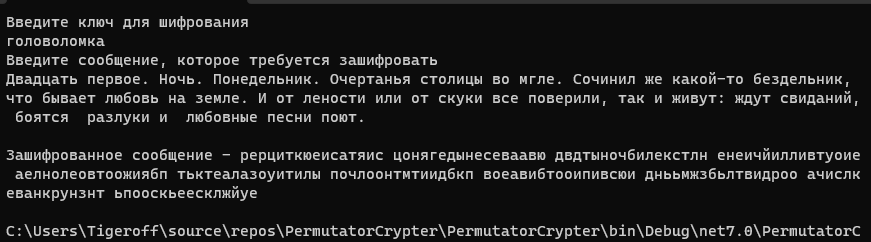


Рисунок 1. Шифрование исходного текста методом перестановок

Если проанализировать полученный шифр, то можно сделать выводы:

- так как длина слово «головоломка» равна 11 буквам, то в шифре должно быть ровно 11 слов (что удовлетворяет условию проверки)

- длина всех слов должна быть одинаковой (или отличаться на 1 символ – из за неполноты последней строки матрицы шифра в процессе шифрования) – длина слов 16-17 – (что удовлетворяет условию проверки).

- кроме того, шифр регистро независимый, что видно по полученной зашифрованной последовательности.

Грубые проверки пройдены, остальное можно проверить более подробно в ручном режиме.

Вывод

В результате выполнения работы я выполнил шифрование заданного сообщения простейшим шифром перестановок и выполнил проверку правильности шифрования.