МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное автономное бюджетное образовательное учреждение высшего образования

«**Севастопольский государственный университет**»

кафедра Информационных систем

Институт информационных технологий и управления в технических системах

Лисянский Александр Игоревич

курс 4 группа ИС/б-42-о

09.03.02 Информационные системы (уровень бакалавриата)

**ОТЧЁТ**

о лабораторном практикуме №2

ШИФРОВАНИЕ ДАННЫХ ПРИ ПОМОЩИ ГЕНЕРАТОРА ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ

по дисциплине «Основы защиты информации»

Отметка о зачёте \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

Руководитель практикума

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность) (подпись) (инициалы, фамилия)

Севастополь

2016

**Цель работы**: Освоение принципов шифрования гаммированием, изучение свойств генератора псевдослучайных чисел, программная реализация метода гаммирования.

**Вариант задания**:

**Линейные конгруэнтные датчики ПСЧ**

Чтобы получить линейные последовательности элементов гаммы, длина которых не превышает размер шифруемых данных, используют датчики ПСЧ. Одним из хороших конгруэнтных генераторов является линейный конгруэнтный датчик ПСЧ. Он вырабатывает последовательности псевдослучайных чисел T(i), описываемые соотношением

T( i+1) = ( A \* T( i ) + C ) mod M ,

где A и C - константы, T(0) - исходная величина, выбранная в качестве порождающего числа. Очевидно, что эти три величины и образуют ключ.

Такой датчик ПСЧ генерирует псевдослучайные числа с определенным периодом повторения, зависящим от выбранных значений A и C. Значение M обычно устанавливается равным 2b , где b -длина машинного слова в битах. Необходимо выбирать числа A и C так, чтобы период M был максимальным.

Как показано Д.Кнуттом, линейный конгруэнтный датчик имеет максимальную длину M тогда, когда C нечетное и A mod 4 = 1.

В качестве примера использования линейного конгруэнтного датчика ПСЧ рассмотрим процесс шифрования исходного текста “абв”. Пусть b = 5, тогда в соответствии с номером алфавите: буква “а” имеет двоичный код 00001; буква “б” имеет двоичный код 00010; буква “в” имеет двоичный код 00011. Исходный текст будет представлен в виде последовательности 00001 00010 00011.

**Метод гаммирования с обратной связью**

Заключается в том, что для получения сегмента гаммы используется контрольная сумма определенного участка шифруемых данных. Например, если рассматривать гамму шифра как объединение непересекающихся множеств H(j), то процесс шифрования можно представить следующими шагами:

1. Генерация сегмента гаммы H(1) и наложение его на соответствующий участок шифруемых данных.

2. Подсчет контрольной суммы участка, соответствующего сегменту гаммы H(1).

3. Генерация с учетом контрольной суммы уже зашифрованного участка данных следующего сегмента гамм H(2).

4. Подсчет контрольной суммы участка данных, соответствующего сегменту данных H(2) и т.д.

Под контрольной суммой понимают функцию f(t(1), ... t(n)), где t(i) - i-е слово шифруемых данных.

Зашифруем исходный текст “абв”, представленный в виде последовательности 00001 00010 00011. Пусть A=5; C=3;b=5; M=32;T(0)=7. Тогда:

T(1)=(5\*7+3) mod 32 = 6 (00110).

В качестве контрольной суммы участка данных, выберем количество единиц на этом участке. Тогда сегменту H(1) соответствует участок 00001, количество единиц равно 1.

**Ход работы**: В соответствии с вариантом задания был разработан алгоритм шифрования сообщений пользователя несколькими методами.

В результате было принято решение о создании графического интерфейса программы с возможностью выбора вида шифрования сообщения. По каждому из методов была определена своя вкладка.

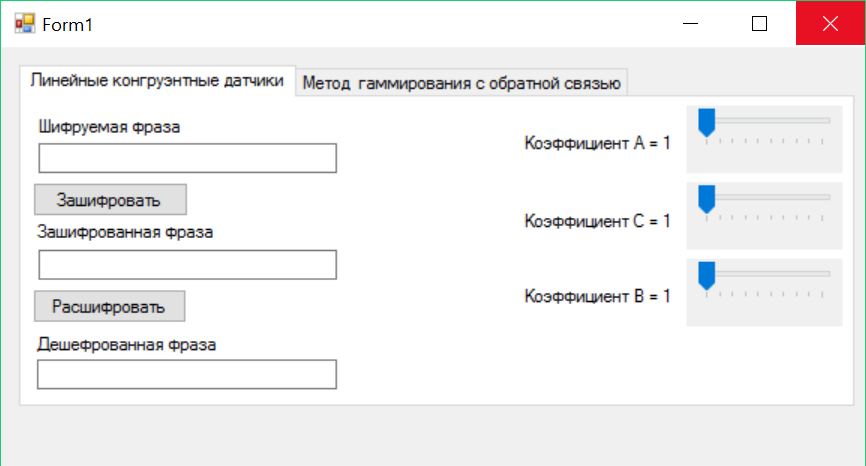


Рисунок 1 – интерфейс программы

Проверяем работу программы. Введем самое простое выражение для двух видов кодировки

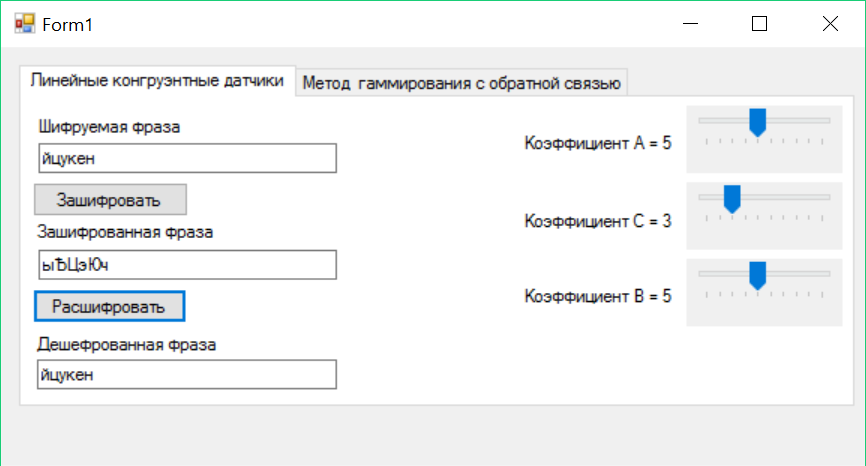


Рисунок 2 – гаммирования при помощи ПСЧ

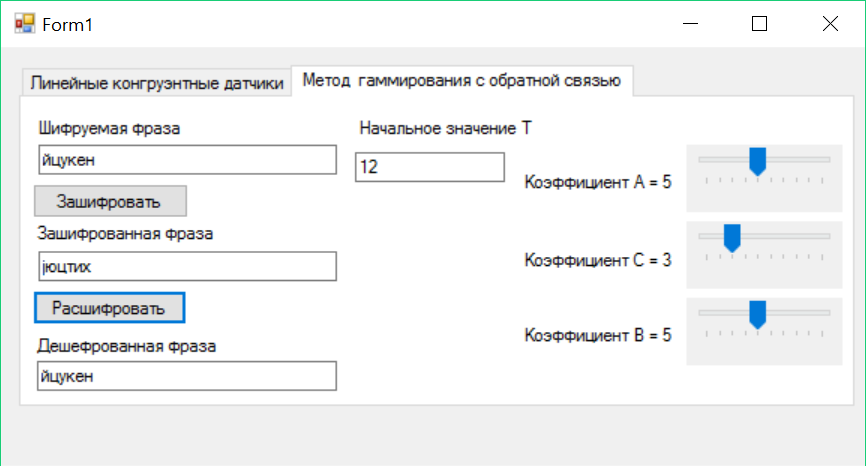


Рисунок 3 – гаммирования с обратной связью

В результате проверки было получено заключение, что программа работает правильно, кодировка удается.

Код программы:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lab2

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private int[] gamma;

private int[] gamma2;

private void EncryptButton\_Click(object sender, EventArgs e)

{

Random rand = new Random();

gamma = new int[WordTextBox.TextLength];

string temp = "";

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

{

gamma[i] = (ATrackBar.Value \* rand.Next() + CTrackBar.Value) % (int)Math.Pow(2, BTrackBar.Value);

temp += (char)((int)WordTextBox.Text[i] + (int)gamma[i]);

}

EncryptTextBox.Text = temp;

}

private void Form1\_Load(object sender, EventArgs e)

{

ALabel.Text = "Коэффициент A = " + ATrackBar.Value;

CLabel.Text = "Коэффициент C = " + CTrackBar.Value;

BLabel.Text = "Коэффициент B = " + BTrackBar.Value;

A\_Label.Text = "Коэффициент A = " + A\_TrackBar.Value;

C\_Label.Text = "Коэффициент C = " + C\_TrackBar.Value;

B\_Label.Text = "Коэффициент B = " + B\_TrackBar.Value;

}

private void Decryptbutton\_Click(object sender, EventArgs e)

{

string temp = "";

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

{

temp += (char)((int)EncryptTextBox.Text[i] - (int)gamma[i]);

}

DecryptTextBox.Text = temp;

}

private void Encrypt\_Button\_Click(object sender, EventArgs e)

{

gamma2 = new int[Word\_TextBox.TextLength];

string temp = "";

try

{

Convert.ToInt32(T\_TextBox.Text);

}

catch

{

MessageBox.Show("Начальные условия\n введены не верно");

return;

}

int ch = Convert.ToInt32(T\_TextBox.Text);

gamma2[0] = (A\_TrackBar.Value \* ch + C\_TrackBar.Value) % (int)Math.Pow(2, B\_TrackBar.Value);

temp += (char)((int)Word\_TextBox.Text[0] + (int)gamma2[0]);

for (int i = 1; i < gamma2.Length; i++)

{

ch = (int)Word\_TextBox.Text[i - 1];

ch = ch % 2;

gamma2[i] = (A\_TrackBar.Value \* ch + C\_TrackBar.Value) % (int)Math.Pow(2, B\_TrackBar.Value);

temp += (char)((int)Word\_TextBox.Text[i] + (int)gamma2[i]);

}

Encrypt\_TextBox.Text = temp;

}

private void Decrypt\_Button\_Click(object sender, EventArgs e)

{

string temp = "";

for (int i = 0; i < gamma2.Length; i++)

{

temp += (char)((int)Encrypt\_TextBox.Text[i] - (int)gamma2[i]);

}

Decrypt\_TextBox.Text = temp;

}

}

}

Вывод: В результате выполнения лабораторной работы были получены навыки по разработке алгоритмов кодирования методами гаммирования, на основе полученных алгоритмов разработаны функции кодирования, а так же написана и отлажена программа с полученными функциями. Результаты тестирования программы показали, что алгоритмы разработаны верно, программа работает правильно.