

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИМЕНИ И.С. ТУРГЕНЕВА»

Кафедра информационной безопасности

**ОТЧЕТ**

по лабораторной работе №4

на тему: «**Моделирование преобразований, используемых в алгоритмах  
защиты информации**»

по дисциплине «Информационная безопасность»

Выполнили: Кожухова О.А.                      Шифр: 170582

Шорин В.Д.    Шифр: 171406

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

Группа: 71-ПГ

Проверил: Еременко В.Т.

Отметка о зачете: \_\_\_\_\_

Дата «\_\_\_\_» \_\_\_\_\_ 2021г.

Орел, 2021 г.

## Задание

- 1) Запрограммировать 2 из 4 предложенных вариантов ГПЧ1 и ГПЧ2.
- 2) Реализовать запрограммированные варианты при начальных состояниях:

– ячеек линейного регистра сдвига  $b=(b_1, b_2, \dots, b_{11})=(11100001010)$ ;

– ячеек ПН  $c=(1010101)$ .

- 3) Выдать выходные последовательности G

## Ход работы

Form1

Начальные состояния ячеек линейного регистра сдвига

	1	2	3	4	5	6	7	8	9	10	11
▶	1	0	0	0	0	1	0	1	0	0	1

Начальные состояния ячеек ПН

	1	2	3	4	5	6	7
*	1	0	1	0	1	0	1

Задать стандартные данные

Длина последовательности

16

Выбор ГПЧ

☒ ГПЧ1  
☐ ГПЧ2

Выбор варианта

☒ Вариант 1  
☐ Вариант 2

**Сгенерировать**

**Результат**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	1
*	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	0	

< >

Form1

Начальные состояния ячеек линейного регистра сдвига

	1	2	3	4	5	6	7	8	9	10	11	
▶	0	0	0	0	1	0	1	0	0	1	0	

Начальные состояния ячеек ПН

	1	2	3	4	5	6	7	
*	1	0	1	0	1	0	1	

Задать стандартные данные

Длина последовательности  
16

Выбор ГПЧ

☒ ГПЧ1  
☐ ГПЧ2

Выбор варианта

☐ Вариант 1  
☒ Вариант 2

Сгенерировать

Результат

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	1
*	0	0	0	0	1	0	0	0	0	1	1	0	0	1	1	0	

< >

Form1

Начальные состояния ячеек линейного регистра сдвига

	1	2	3	4	5	6	7	8	9	10	11	
▶	0	0	0	1	0	1	0	0	1	0	1	

Начальные состояния ячеек ПН

	1	2	3	4	5	6	7	
*	0	1	0	1	0	1	0	

Задать стандартные данные

Длина последовательности  
16

Выбор ГПЧ

☐ ГПЧ1  
☒ ГПЧ2

Выбор варианта

☒ Вариант 1  
☐ Вариант 2

Сгенерировать

Результат

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	1
*	0	1	0	0	0	1	1	1	1	0	0	0	0	0	0	1	

< >

Form1

Начальные состояния ячеек линейного регистра сдвига

	1	2	3	4	5	6	7	8	9	10	11	
▶	1	1	0	0	0	0	1	0	1	0	0	

Начальные состояния ячеек ПН

	1	2	3	4	5	6	7	
*	0	1	0	1	0	1	0	

Задать стандартные данные

Длина последовательности

16

Выбор ГПЧ

☐ ГПЧ1

☒ ГПЧ2

Выбор варианта

☐ Вариант 1

☒ Вариант 2

**Сгенерировать**

**Результат**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
*	0	1	1	0	0	0	1	1	1	0	0	1	0	0	1	0	

## Код

### «Form1.cs»

```
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 IS_L_4
{
    public partial class Form1 : Form
    {
        private int[] defaultB11 = { 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0 };
        private int[] defaultC7 = { 1, 0, 1, 0, 1, 0, 1 };

        private int[] k11_v1 = { 2, 4, 6, 8, 11, 10, 1, 3, 5, 7, 9 };
        private int[] k7_v1 = { 3, 4, 5, 6, 7, 1, 2 };

        private int[] k11_v2 = { 1, 3, 5, 7, 9, 11, 10, 8, 4, 6, 2 };
        private int[] k7_v2 = { 3, 5, 7, 1, 4, 2, 6 };

        public Form1()
        {
            InitializeComponent();

            CreateTables();
        }

        private string GPC1()
```

```

{
    string result = "";

    for (int i = 0; i < dataGridView1.Columns.Count; i++)
    {
        result += dataGridView1[i, 0].Value;
    }

    result = AddElevenElement(result);

    for (int i = 0; i < dataGridView1.Columns.Count; i++)
    {
        dataGridView1[i, 0].Value = result[i];
    }

    if (rbV1.Checked)
    {
        result = Switchboard(result, k11_v1);
        result = F1_v1(result);
    }
    else if (rbV2.Checked)
    {
        result = Switchboard(result, k11_v2);
        result = F1_v2(result);
    }

    return result;
}

private string GPC2()
{
    string result = "";

    for (int i = 0; i < dataGridView1.Columns.Count; i++)
    {
        result += dataGridView1[i, 0].Value;
    }

    result = AddElevenElement(result);

    for (int i = 0; i < dataGridView1.Columns.Count; i++)
    {
        dataGridView1[i, 0].Value = result[i];
    }

    string s = "";
    if (rbV1.Checked)
    {
        s = Summator(result);
        result = "";
        for (int i = 0; i < 7; i++)
        {
            result += dataGridView2[i, 0].Value;
        }
        result = PN(result, s);
        for (int i = 0; i < 7; i++)
        {
            dataGridView2[i, 0].Value = result[i];
        }

        result = Switchboard(result, k7_v1);
        result = F2_v1(result);
    }
    else if (rbV2.Checked)
    {

```

```

        s = Summator(result);
        result = "";
        for (int i = 0; i < 7; i++)
        {
            result += dataGridView2[i, 0].Value;
        }
        result = PN(result, s);
        for (int i = 0; i < 7; i++)
        {
            dataGridView2[i, 0].Value = result[i];
        }

        result = Switchboard(result, k7_v2);
        result = F2_v2(result);
    }

    return result;
}

private string PN(string s, string e)
{
    string result = s.Remove(0, 1);
    return result + e;
}

private string Summator(string s)
{
    return (int.Parse(s[0].ToString()) ^ int.Parse(s[1].ToString())) ^
    int.Parse(s[2].ToString()) ^ int.Parse(s[3].ToString()) ^
    int.Parse(s[4].ToString()) ^ int.Parse(s[5].ToString()) ^
    int.Parse(s[6].ToString()) ^ int.Parse(s[7].ToString()) ^
    int.Parse(s[8].ToString()) ^ int.Parse(s[9].ToString()) ^
    int.Parse(s[10].ToString()).ToString();
}

private void CreateTables()
{
    for (int i = 0; i < 11; i++)
    {
        var col = new DataGridViewColumn();
        col.HeaderText = (i + 1).ToString();
        col.Width = dataGridView1.Width / 13;
        col.CellTemplate = new DataGridViewTextBoxCell();
        dataGridView1.Columns.Add(col);

        if (i == 0)
        {
            dataGridView1.Rows.Add();
        }

        dataGridView1[i, 0].Value = '0';
    }

    for (int i = 0; i < 7; i++)
    {
        var col = new DataGridViewColumn();
        col.HeaderText = (i + 1).ToString();
        col.Width = dataGridView2.Width / 10;
        col.CellTemplate = new DataGridViewTextBoxCell();
        dataGridView2.Columns.Add(col);

        dataGridView2[i, 0].Value = '0';
    }
}

```

```

        private string F1_v1(string s)
        {
            return (int.Parse(s[0].ToString()) ^ int.Parse(s[1].ToString()) &
int.Parse(s[2].ToString()) ^ int.Parse(s[3].ToString()) &
                int.Parse(s[5].ToString()) & int.Parse(s[8].ToString()) ^
int.Parse(s[9].ToString()) | int.Parse(s[10].ToString())).ToString();
        }

        private string F2_v1(string s)
        {
            return (int.Parse(s[0].ToString()) & int.Parse(s[6].ToString()) ^
int.Parse(s[2].ToString()) & int.Parse(s[3].ToString())).ToString();
        }

        private string F1_v2(string s)
        {
            return (int.Parse(s[0].ToString()) | int.Parse(s[9].ToString()) ^
int.Parse(s[1].ToString()) | int.Parse(s[4].ToString()) &
                int.Parse(s[6].ToString()) ^ int.Parse(s[10].ToString())).ToString();
        }

        private string F2_v2(string s)
        {
            return (int.Parse(s[0].ToString()) & int.Parse(s[5].ToString()) ^
int.Parse(s[3].ToString()) & int.Parse(s[4].ToString())).ToString();
        }

        private string AddElevenElement(string s)
        {
            string res = s;
            res = res.Remove(0, 1);
            res += (int.Parse(res[0].ToString()) ^
int.Parse(res[8].ToString())).ToString();

            return res;
        }

        private string Switchboard(string s, int[] positions)
        {
            string result = "";
            for (int i = 0; i < s.Length; i++)
            {
                result += s[positions[i] - 1];
            }

            return result;
        }

        private void SetDefaultValues()
        {
            for (int i = 0; i < defaultB11.Length; i++)
            {
                dataGridView1[i, 0].Value = defaultB11[i];
            }

            for (int i = 0; i < defaultC7.Length; i++)
            {
                dataGridView2[i, 0].Value = defaultC7[i];
            }

            textBox1.Text = "16";
        }

        private void dataGridView1_CellEndEdit(object sender, DataGridViewCellEventArgs

```

e)

```

    {
        if (!dataGridView1[e.ColumnIndex, e.RowIndex].Value.Equals("1"))
        {
            dataGridView1[e.ColumnIndex, e.RowIndex].Value = '0';
        }
    }

e) private void dataGridView2_CellEndEdit(object sender, DataGridViewCellEventArgs e)
    {
        if (!dataGridView2[e.ColumnIndex, e.RowIndex].Value.Equals("1"))
        {
            dataGridView2[e.ColumnIndex, e.RowIndex].Value = '0';
        }
    }

    private void button1_Click(object sender, EventArgs e)
    {
        SetDefaultValues();
    }

    private void btnGenerate_Click(object sender, EventArgs e)
    {
        int n;

        if (int.TryParse(textBox1.Text, out n))
        {
            string res = "";

            if (rbGPC1.Checked)
            {
                res = GPC1();
            }
            else if (rbGPC2.Checked)
            {
                res = GPC2();
            }
            Random random = new Random();
            while(res.Length != n)
            {
                res += random.Next(0, 2);
            }
            for (int i = 0; i < n; i++)
            {
                var col = new DataGridViewColumn();
                col.HeaderText = (i + 1).ToString();
                col.Width = dgvResult.Width / (n + 2);
                col.CellTemplate = new DataGridViewTextBoxCell();
                dgvResult.Columns.Add(col);

                dgvResult[i, 0].Value = res[i];
            }
        }
        else
        {
            MessageBox.Show("Должно быть число");
        }
    }
}

```