

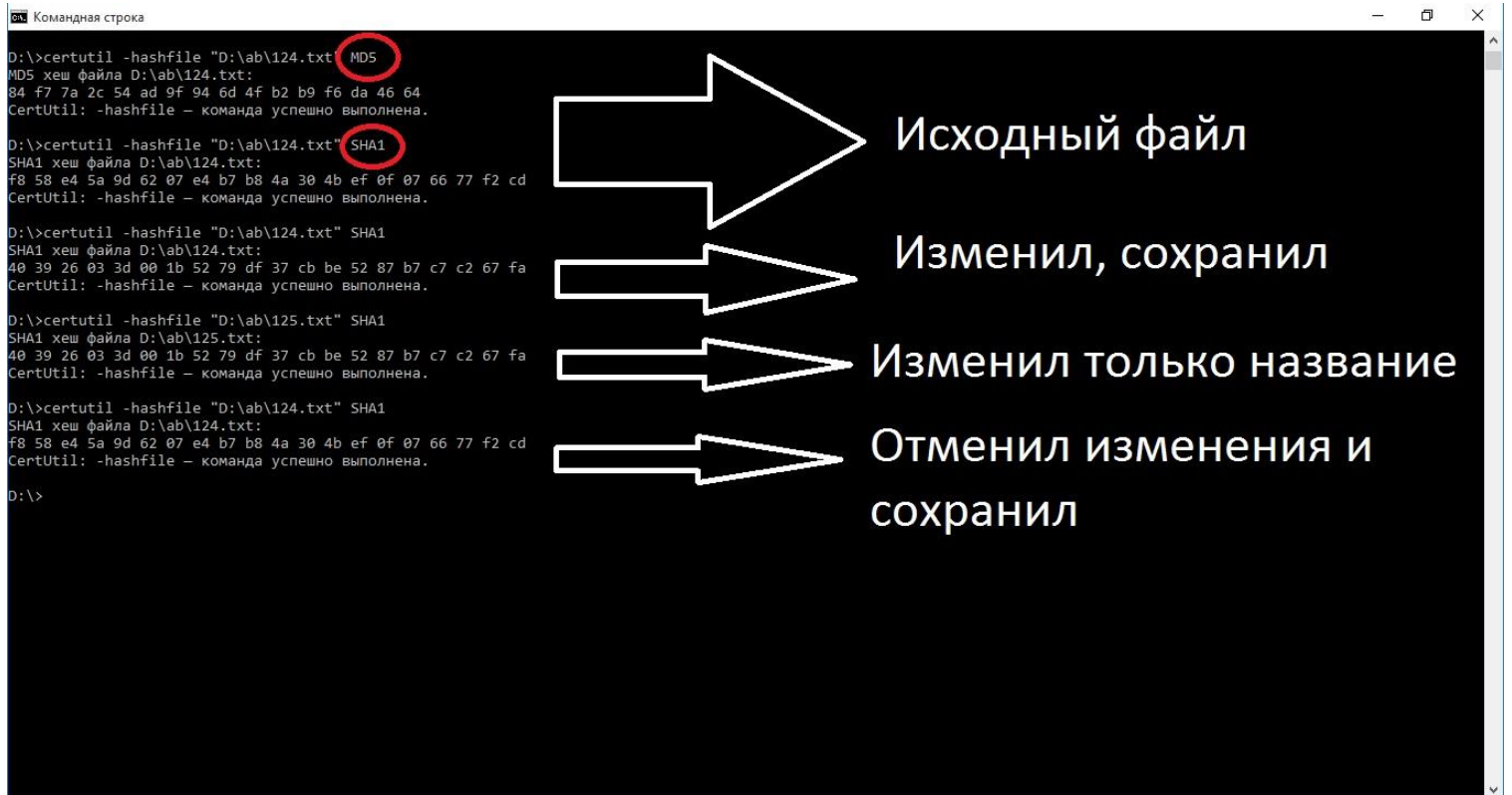
Казанский (Приволжский) Федеральный университет
Институт Вычислительной математики и информационных
технологий

Отчёт по работам на практике ТиМП.

Казань – 2018

Студент 09-641
Десятов Александр

Работа в консоли



```
D:\>certutil -hashfile "D:\ab\124.txt" MD5
MD5 хеш файла D:\ab\124.txt:
84 f7 7a 2c 54 ad 9f 94 6d 4f b2 b9 f6 da 46 64
CertUtil: -hashfile – команда успешно выполнена.

D:\>certutil -hashfile "D:\ab\124.txt" SHA1
SHA1 хеш файла D:\ab\124.txt:
f8 58 e4 5a 9d 62 07 e4 b7 b8 4a 30 4b ef 0f 07 66 77 f2 cd
CertUtil: -hashfile – команда успешно выполнена.

D:\>certutil -hashfile "D:\ab\124.txt" SHA1
SHA1 хеш файла D:\ab\124.txt:
40 39 26 03 3d 00 1b 52 79 df 37 cb be 52 87 b7 c7 c2 67 fa
CertUtil: -hashfile – команда успешно выполнена.

D:\>certutil -hashfile "D:\ab\125.txt" SHA1
SHA1 хеш файла D:\ab\125.txt:
40 39 26 03 3d 00 1b 52 79 df 37 cb be 52 87 b7 c7 c2 67 fa
CertUtil: -hashfile – команда успешно выполнена.

D:\>certutil -hashfile "D:\ab\124.txt" SHA1
SHA1 хеш файла D:\ab\124.txt:
f8 58 e4 5a 9d 62 07 e4 b7 b8 4a 30 4b ef 0f 07 66 77 f2 cd
CertUtil: -hashfile – команда успешно выполнена.

D:\>
```

Исходный файл

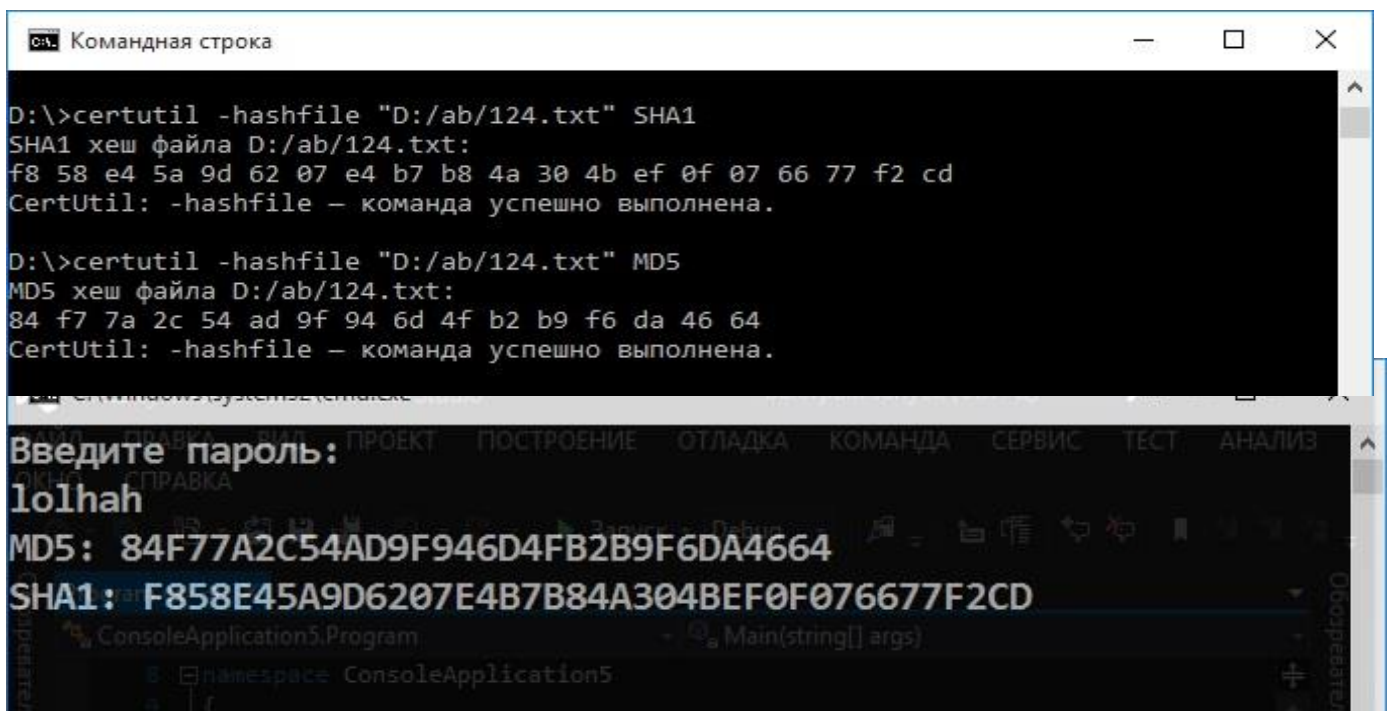
Изменил, сохранил

Изменил только название

Отменил изменения и сохранил

- 1) в файле 124.txt изначально: "lolhah"
- 2) в файле 124.txt после изменений: "lol"
- 3) в файле 125.txt : «lol»
- 4) в файле 124.txt снова: "lolhah"

Хэш-суммы программно



```
D:\>certutil -hashfile "D:/ab/124.txt" SHA1
SHA1 хеш файла D:/ab/124.txt:
f8 58 e4 5a 9d 62 07 e4 b7 b8 4a 30 4b ef 0f 07 66 77 f2 cd
CertUtil: -hashfile – команда успешно выполнена.

D:\>certutil -hashfile "D:/ab/124.txt" MD5
MD5 хеш файла D:/ab/124.txt:
84 f7 7a 2c 54 ad 9f 94 6d 4f b2 b9 f6 da 46 64
CertUtil: -hashfile – команда успешно выполнена.
```

Введите пароль:
lolhah

MD5: 84F77A2C54AD9F946D4FB2B9F6DA4664

SHA1: F858E45A9D6207E4B7B84A304BEF0F076677F2CD

Введён тот текст, который был в файле 124.txt. Хэш-суммы совпали!

Вывод: у разных текстов разные хэш-суммы; если взять разные файлы с одним текстом, хэш-суммы совпадут.

```
using System;
using System.Security.Cryptography;
```

```
using System.Text;

namespace ConsoleApplication5
{
    class Program
    {
        static void Main(string[] args)
        {
            SHA1 sha1 = new SHA1CryptoServiceProvider();
            MD5 md5 = new MD5CryptoServiceProvider();
            Console.WriteLine("Введите пароль:");
            // Вводим слово
            string pass = Console.ReadLine();
            // Хэшируем его двумя способами
            byte[] checksum = md5.ComputeHash(Encoding.UTF8.GetBytes(pass));
            byte[] checksum1 = sha1.ComputeHash(Encoding.UTF8.GetBytes(pass));
            string result = BitConverter.ToString(checksum).Replace("-", String.Empty);
            string result1 = BitConverter.ToString(checksum1).Replace("-", String.Empty);
            // Выводи захэшированное
            Console.WriteLine("MD5: {0}", result);
            Console.WriteLine("SHA1: {0}", result1);
            Console.Read();
        }
    }
}
```

Таблица хэш-сумм

	Столбик	Столбик	Столбик
▶	Привет0	Привет1	Привет2
	Привет1	Привет2	Привет3
*	Привет2	Привет3	Привет4

	Столбик	Столбик	Столбик
▶	1F379C6A1214B66C1A1C27E5FDCD84D1A07FBB4B	74681027F4CF735875A80C84F39044478724E29A	83E38D51ECD0101D2E57802DF2788CF640CFA447
	74681027F4CF735875A80C84F39044478724E29A	83E38D51ECD0101D2E57802DF2788CF640CFA447	9BD825CC3AB22F2897D418A5E90F2428DE52ED5B
*	83E38D51ECD0101D2E57802DF2788CF640CFA447	9BD825CC3AB22F2897D418A5E90F2428DE52ED5B	FE1F354ECF243BBB802801B0894EB3349F32D2CB

```
using System;
using System.Security.Cryptography;
using System.Text;
using System.Windows.Forms;

namespace Хеш_суммы_форм
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            // Создаем таблицу 1 исходных слов
            dataGridView1.Columns.Clear();
            for (int i = 0; i < 3; i++)
            {
                dataGridView1.Columns.Add("", "Столбик");
                dataGridView2.Columns.Add("", "Столбик");
                if (i != 0)
                {
                    dataGridView1.Rows.Add();
                    dataGridView2.Rows.Add();
                }
            }
            for (int i = 0; i < 3; i++)
                for (int j = 0; j < 3; j++)
                    dataGridView1.Rows[i].Cells[j].Value = "Привет" + (i + j).ToString();

            // Создаем таблицу 2 хэшированных слов из первой таблицы
            SHA1 sha1 = new SHA1CryptoServiceProvider();
            for (int i = 0; i < 3; i++)
                for (int j = 0; j < 3; j++)
                {
                    string pass = dataGridView1.Rows[i].Cells[j].Value.ToString();
                    byte[] checksum = sha1.ComputeHash(Encoding.UTF8.GetBytes(pass));
                    dataGridView2.Rows[i].Cells[j].Value = BitConverter.ToString(checksum).Replace("-",
String.Empty);
                }
        }
    }
}
```

Подсчет хеш-сумм по формуле с помощью ключа

```
using System;
using System.Windows.Forms;
```

```
namespace Хэши_слов
```

```
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        int[] hesh;
        string[] slovo_str;
        int[] povtoren;
        bool[] bilo;
        char[] no_bukv = {
            '\n', '.', ',', '!', '?', ':', ';', '"', '(', ')', '%', '@', '#', '№', '$', '^', '&', '*', '-',
            '+', '_', '=', '/', '\\', '|', '<', '>', '[', ']', '{', '}', '~', '`'};
        private void button1_Click(object sender, EventArgs e)
        {
            // Очищаем предыдущие подсчёты
            richTextBox2.Clear();
            richTextBox3.Clear();
            // Переводим в нижний регистр и делим по словам
            slovo_str = richTextBox1.Text.ToLower().Split(no_bukv,
StringSplitOptions.RemoveEmptyEntries);
            int Vsego_slov = slovo_str.Length; // Кол-во слов
            povtoren = new int[Vsego_slov];
            hesh = new int[Vsego_slov];
            bilo = new bool[Vsego_slov];
            for (int i = 0; i < Vsego_slov; i++)
            {
                int h = Find_hesh(slovo_str[i]);
                hesh[i] = h;
                bilo[i] = Bilo(h, i);
            }
            for (int i = 0; i < slovo_str.Length; i++)
            {
                if (!bilo[i])
                { // Выводим кол-во встречаний, хэш-суммы, каждого слова из текста
                    richTextBox2.Text += (povtoren[i].ToString() + " - " + slovo_str[i] + " - " +
hesh[i].ToString() + "\n");
                }
            }
        }
        // Считает Хэш по формуле
        private int Find_hesh(string slovo)
        {
            int hesh = 0;
            int P = Convert.ToInt32(textBox1.Text); // Ключ
```

```

        int mnoz = 1;
        char[] s1 = slovo.ToCharArray();
        for (int j = 0; j < s1.Length; j++)
        {
            hesh += slovo[j] * mnoz; //
            mnoz *= P;
        }
        return hesh;
    }
    private bool Bilo(int h, int num_sl)
    {
        //Проверяем была ли уже такая хэш-сумма
        for (int i = 0; i < num_sl; i++)
            if (hesh[i] == h)
            {
                if (slovo_str[num_sl] == slovo_str[i])
                { // Одно и то же слово
                    povtoren[i] += 1;
                    return true;
                }
                else
                { // Одна хэш-сумма у разных слов
                    string str_new = slovo_str[num_sl] + " - " + slovo_str[i];
                    string[] stroki = richTextBox3.Text.Split('\n');
                    bool bil = false;
                    for (int j = 0; j < stroki.Length; j++)
                    {
                        if (stroki[j] == str_new)
                        {
                            bil = true;
                            break;
                        }
                    }
                    if (!bil){
                        // Записываем 1 раз слова, у которых совпали хэш-суммы
                        richTextBox3.Text += str_new + '\n';
                    }
                }
            }
        povtoren[num_sl] = 1;
        return false;
    }
}
}
}

```

Энигма



Коммутатор:

AB	WX											
----	----	--	--	--	--	--	--	--	--	--	--	--

Save

```
using System;
using System.Drawing;
using System.Windows.Forms;
```

```
namespace Нейман
{
    public partial class Энигма : Form
    {
        public Энигма()
        {
            InitializeComponent();
        }
    }
}
```

```
string encrypted_letter;
public string commutator_str = "BACDEFGHIJKLMNOPQRSTUVWXYZ";
public string alfavit = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
string rotor1 = "EKMFLGDQVZNTOWYHXUSPAIBRCJ";
string rotor2 = "AJDKSIRUXBLHWTMCQGZNPYFVOE";
string rotor3 = "BDFHJLCPRTXVZNYEIWGAKMUSQO";
string reflector = "ABCD";
//reflectors
string reflectA = "EJMZALYXVBWFCRQUONTSPIKHGD";
string reflectB = "YRUHQSLEDPXNGOKMIEBFZCWWJAT";
```

```

string reflectC = "FVPJIAOYEDRZXWGCTKUQSBNMHL";
string reflectD = "RDOBJNTKVEHMLFCWZAXGYIPSUQ"; // reflector C Dunn
int i_rotor1 = 16, i_rotor2 = 21, i_rotor3 = 2; //номера в обычном алфавите
int i_reflector = 1;
int i_rot1_save, i_rot2_save, i_rot3_save, i_ref_save;
bool power = true;

private void Form1_Load(object sender, EventArgs e)
{
    // Подготавливаем внешний вид при загрузке:
    panel_buttons.Parent = pictureBox_fon;
    panel_lamps.Parent = pictureBox_fon;
    panel_rotors.Parent = pictureBox_fon;
    panel_reflector.Parent = pictureBox_fon;
    panel_power.Parent = pictureBox_fon;
    pBrotor1Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor1, 1) + ".png");
    pBrotor2Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor2, 1) + ".png");
    pBrotor3Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor3, 1) + ".png");
    pBreflector.BackgroundImage = new Bitmap("images\\reflector_" + reflector.Substring(i_reflector, 1) + ".png");
    pBpower.BackgroundImage = new Bitmap("images\\power_" + power + ".png");
    i_rot1_save = i_rotor1;
    i_rot2_save = i_rotor2;
    i_rot3_save = i_rotor3;
    i_ref_save = i_reflector;
    // Добавляем функции для кнопок
    for (int i = 0; i < panel_buttons.Controls.Count; i++)
    {
        panel_buttons.Controls[i].MouseDown += button_MouseDown;
        panel_buttons.Controls[i].MouseUp += button_MouseUp;
        panel_buttons.Controls[i].MouseMove += button_MouseMove;
        panel_buttons.Controls[i].MouseLeave += button_MouseLeave;
    }
}

// При нажатии на кнопку
private void button_MouseDown(object sender, MouseEventArgs e)
{
    if (power)
    {
        PictureBox pb = new PictureBox();
        pb = (PictureBox)sender;
        Switch_on_lamp(Encrypt(pb.Tag.ToString()));
    }
}

// При наведении на кнопку (Чтоб засветилась)
private void button_MouseMove(object sender, EventArgs e)
{
    PictureBox pb = new PictureBox();
    pb = (PictureBox)sender;
    string location = "images\\button_" + pb.Tag + "_pressed.png";
    pb.BackgroundImage = new Bitmap(location);
}

```



```

// При отпускании кнопки
private void button_MouseUp(object sender, MouseEventArgs e)
{
    if (power)
    {
        PictureBox pb = new PictureBox();
        pb = (PictureBox)sender;
        Switch_off_lamp(encrypted_letter);
        tBinput.Text += pb.Tag;
        tBoutput.Text += encrypted_letter;
    }
}

// При уходе курсора с положения кнопки (перестает светиться)
private void button_MouseLeave(object sender, EventArgs e)
{
    PictureBox pb = new PictureBox();
    pb = (PictureBox)sender;
    string location = "images\\button_" + pb.Tag + ".png";
    pb.BackgroundImage = new Bitmap(location);
}

// Зашифровываем букву
private string Encrypt(string letter)
{
    // Крутим ротор(ы)
    i_rotor1 = (i_rotor1 + 1) % alfavit.Length;
    pBrotor1Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor1, 1) + ".png");
    if (i_rotor1 == 0)
    {
        i_rotor2 = (i_rotor2 + 1) % alfavit.Length;
        pBrotor2Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor2, 1) + ".png");
        if (i_rotor2 == 0)
        {
            i_rotor3 = (i_rotor3 + 1) % alfavit.Length;
            pBrotor3Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor3, 1) + ".png");
        }
    }

    // Путь шифрования буквы:
    label_Path.Text = "Path: " + letter + " -> commutator -> ";

    // Заходим в коммутатор
    int nom = Num_letter(letter, alfavit);
    letter = commutator_str.Substring(nom, 1); //A
    label_Path.Text += (letter + " -> ");

    // Заходим в 1 ротор
    nom = Num_letter(letter, alfavit) + i_rotor1;
    nom = nom < 0 ? nom + alfavit.Length : (nom >= alfavit.Length ? nom % alfavit.Length : nom);
    letter = alfavit.Substring(nom, 1); //R
    label_Path.Text += (letter + " -> rotorI -> ");
    letter = rotor1.Substring(nom, 1); //U
    label_Path.Text += (letter + " -> ");
}

```

```
// Заходим в 2 ротор
nom = Num_letter(letter, alfavit) + i_rotor2 - i_rotor1;
nom = nom < 0 ? nom + alfavit.Length : (nom >= alfavit.Length ? nom % alfavit.Length : nom);
letter = alfavit.Substring(nom, 1);//Y
label_Path.Text += (letter + " -> rotorII -> ");
letter = rotor2.Substring(nom, 1);//O
label_Path.Text += (letter + " -> ");
```

```
// Заходим в 3 ротор
nom = Num_letter(letter, alfavit) + i_rotor3 - i_rotor2;
nom = nom < 0 ? nom + alfavit.Length : (nom >= alfavit.Length ? nom % alfavit.Length : nom);
letter = alfavit.Substring(nom, 1);//V
label_Path.Text += (letter + " -> rotorIII -> ");
letter = rotor3.Substring(nom, 1);//M
label_Path.Text += (letter + " -> ");
```

```
// Заходим в рефлектор
nom = Num_letter(letter, alfavit) - i_rotor3;
nom = nom < 0 ? nom + alfavit.Length : (nom >= alfavit.Length ? nom % alfavit.Length : nom);
letter = alfavit.Substring(nom, 1);//K
label_Path.Text += (letter + " -> reflector -> ");
letter = Reflector_letter(nom);//N
label_Path.Text += (letter + " -> ");
```

```
// Заходим в 3 ротор
nom = Num_letter(letter, alfavit) + i_rotor3;
nom = nom < 0 ? nom + alfavit.Length : (nom >= alfavit.Length ? nom % alfavit.Length : nom);
letter = alfavit.Substring(nom, 1);//P
label_Path.Text += (letter + " -> rotorIII -> ");
nom = Num_letter(letter, rotor3);
letter = alfavit.Substring(nom, 1);//H
label_Path.Text += (letter + " -> ");
```

```
// Заходим во 2 ротор
nom = Num_letter(letter, alfavit) - i_rotor3 + i_rotor2;
nom = nom < 0 ? nom + alfavit.Length : (nom >= alfavit.Length ? nom % alfavit.Length : nom);
letter = alfavit.Substring(nom, 1);//A
label_Path.Text += (letter + " -> rotorII -> ");
nom = Num_letter(letter, rotor2);
letter = alfavit.Substring(nom, 1);//A
label_Path.Text += (letter + " -> ");
```

```
//Заходим в 1 ротор
nom = Num_letter(letter, alfavit) - i_rotor2 + i_rotor1;
nom = nom < 0 ? nom + alfavit.Length : (nom >= alfavit.Length ? nom % alfavit.Length : nom);
letter = alfavit.Substring(nom, 1);//W
label_Path.Text += (letter + " -> rotorI -> ");
nom = Num_letter(letter, rotor1);
letter = alfavit.Substring(nom, 1);//N
label_Path.Text += (letter + " -> ");
```

```
//Output
```

```

nom = Num_letter(letter, alfavit) - i_rotor1;
nom = nom < 0 ? nom + alfavit.Length : (nom >= alfavit.Length ? nom % alfavit.Length : nom);
letter = alfavit.Substring(nom, 1);//W
label_Path.Text += letter + " -> commutator -> ";

// Заходим в коммутатор
nom = Num_letter(letter, alfavit);
letter = commutator_str.Substring(nom, 1);//X
label_Path.Text += letter;

encrypted_letter = letter;
return encrypted_letter;
}

// Узнаем номер буквы в алфавите, переданном аргументом
public int Num_letter(string letter, string alf)
{
    int number = 0;
    while (letter != alf.Substring(number, 1))
    {
        number++;
        if (number >= alf.Length)
            return 0;
    }
    return number;
}

// Отражение в рефлекторе
private string Reflector_letter(int nom)
{
    switch (i_reflector)
    {
        case 0:
            return reflectA.Substring(nom, 1);
        case 1:
            return reflectB.Substring(nom, 1);
        case 2:
            return reflectC.Substring(nom, 1);
        case 3:
            return reflectD.Substring(nom, 1);
        default:
            return "";
    }
}

// Лампочка начинает светиться
private void Switch_on_lamp(string letter)
{
    for (int i = 0; i < panel_lamps.Controls.Count; i++)
    {
        if (panel_lamps.Controls[i].Tag.ToString() == letter)
        {
            string location = "images\\lamp_" + letter + "_pressed.png";

```

```

        panel_lamps.Controls[i].BackgroundImage = new Bitmap(location);
        break;
    }
}

// Лампочка перестает светиться
private void Switch_off_lamp(string letter)
{
    for (int i = 0; i < panel_lamps.Controls.Count; i++)
    {
        string location = "images\\lamp_" + panel_lamps.Controls[i].Tag + ".png";
        panel_lamps.Controls[i].BackgroundImage = new Bitmap(location);
    }
}

// Переход к следующей букве в роторе (при настройке ключа)
private void rotor_down_Click(object sender, EventArgs e)
{
    PictureBox pb = new PictureBox();
    pb = (PictureBox)sender;
    switch (pb.Tag.ToString())
    {
        case "1":
            i_rotor1 = (i_rotor1 + 1) % alfavit.Length;
            pBrotor1Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor1, 1) + ".png");
            break;
        case "2":
            i_rotor2 = (i_rotor2 + 1) % alfavit.Length;
            pBrotor2Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor2, 1) + ".png");
            break;
        case "3":
            i_rotor3 = (i_rotor3 + 1) % alfavit.Length;
            pBrotor3Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor3, 1) + ".png");
            break;
        default:
            break;
    }
}

// Переход к предыдущей букве в роторе (при настройке ключа)
private void rotor_up_Click(object sender, EventArgs e)
{
    PictureBox pb = new PictureBox();
    pb = (PictureBox)sender;
    switch (pb.Tag.ToString())
    {
        case "1":
            i_rotor1 = i_rotor1 - 1 == -1 ? alfavit.Length - 1 : i_rotor1 - 1;
            pBrotor1Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor1, 1) + ".png");
            break;
        case "2":
            i_rotor2 = i_rotor2 - 1 == -1 ? alfavit.Length - 1 : i_rotor2 - 1;
            pBrotor2Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor2, 1) + ".png");

```

```

        break;
    case "3":
        i_rotor3 = i_rotor3 - 1 == -1 ? alfavit.Length - 1 : i_rotor3 - 1;
        pBrotor3Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor3, 1) + ".png");
        break;
    default:
        break;
}
}

// Выбор рефлектора
private void pBreflector_Click(object sender, EventArgs e)
{
    i_reflector = (i_reflector + 1) % reflector.Length;
    pBreflector.BackgroundImage = new Bitmap("images\\reflector_" + reflector.Substring(i_reflector, 1) + ".png");
}

// Включение/выключение машины
private void pBpower_Click(object sender, EventArgs e)
{
    power = !power;
    pBpower.BackgroundImage = new Bitmap("images\\power_" + power + ".png");
}

// Сохраняем ключ
private void button_save_Click(object sender, EventArgs e)
{
    i_rot1_save = i_rotor1;
    i_rot2_save = i_rotor2;
    i_rot3_save = i_rotor3;
    i_ref_save = i_reflector;
}

// Загружаем последний сохраненный ключ
private void button_load_Click(object sender, EventArgs e)
{
    i_rotor1 = i_rot1_save;
    i_rotor2 = i_rot2_save;
    i_rotor3 = i_rot3_save;
    i_reflector = i_ref_save;
    pBrotor1Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor1, 1) + ".png");
    pBrotor2Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor2, 1) + ".png");
    pBrotor3Symb.BackgroundImage = new Bitmap("images\\letter_" + alfavit.Substring(i_rotor3, 1) + ".png");
    pBreflector.BackgroundImage = new Bitmap("images\\reflector_" + reflector.Substring(i_reflector, 1) + ".png");
}

// Очистить поле ввода
private void button_clear_input_Click(object sender, EventArgs e)
{
    tBinput.Text = ">>";
}

// Очистить поле вывода
private void button_clear_output_Click(object sender, EventArgs e)

```

```

{
    tBoutput.Text = "<<";
}

// Вызвать настройку коммутатора
private void commutator_Click(object sender, EventArgs e)
{
    Commutator commut_form = new Commutator();
    commut_form.Owner = this;
    commut_form.Show();
}
}
}

```

Коммутатор:

```

using System;
using System.Windows.Forms;

namespace Нейман
{
    public partial class Commutator : Form
    {
        public Commutator()
        {
            InitializeComponent();
        }
        //Хотелось бы protected
        Энигма энигма;
        private void Commutator_Load(object sender, EventArgs e)
        {
            энигма = this.Owner as Энигма;
            label_error.Text = "";
            //Записываем предыдущий коммутатор
            for (int i = 0, num_tb = 0; i < энигма.alfavit.Length; i++)
            {
                string let0 = энигма.alfavit.Substring(i, 1);
                string let1 = энигма.commutator_str.Substring(i,1);
                if(let0 != let1)
                {
                    if (Checking(let0) && Checking(let1))
                    {
                        panel1.Controls[panel1.Controls.Count - num_tb - 1].Text = let0 + let1;
                        num_tb++;
                    }
                }
            }
        }

        private void button1_Click(object sender, EventArgs e)
        {
            label_error.Text = "";
            int ret = Checking();
            if (ret == 0)
            {
                // Записываем в алфавит коммутатора
                for (int i = 0; i < panel1.Controls.Count; i++)
                {
                    if (panel1.Controls[i].Text.Length == 2)
                    {
                        string let0 = panel1.Controls[i].Text.Substring(0, 1);
                        string let1 = panel1.Controls[i].Text.Substring(1, 1);
                        int n0 = энигма.Num_letter(let0, энигма.alfavit);
                        int n1 = энигма.Num_letter(let1, энигма.alfavit);
                        энигма.commutator_str = энигма.commutator_str.Substring(0, n0) + let1 +
                        энигма.commutator_str.Substring(n0+1, энигма.alfavit.Length - n0 - 1);
                    }
                }
            }
        }
    }
}

```

```

        энigma.commutator_str = энigma.commutator_str.Substring(0, n1) + let0 +
        энigma.commutator_str.Substring(n1+1, энigma.alfavit.Length - n1 - 1);
    }
}

Close();
}
else // Возможные ошибки
    if (ret == 1)
        label_error.Text = "Error! Two or zero letters in each!";
    else
        if (ret == 2)
            label_error.Text = "Error! One letter is used more one time!";
}

private int Checking()
{
    // Проверяем можно ли так сохранить коммутатор
    bool[] existing = new bool[энigma.alfavit.Length];
    for (int i = 0; i < энigma.alfavit.Length / 2; i++)
    {
        string two_letter = panel1.Controls[i].Text.ToUpper();
        if (two_letter.Length != 0 && two_letter.Length != 2)
            return 1;
        else
            if (two_letter.Length == 2)
            {
                int num;

                num = энigma.Num_letter(two_letter.Substring(0, 1), энigma.alfavit);
                if (existing[num])
                    return 2;
                else
                    existing[num] = true; // Чтобы буква не использовалась 2 раз

                num = энigma.Num_letter(two_letter.Substring(1, 1), энigma.alfavit);
                if (existing[num])
                    return 2;
                else
                    existing[num] = true;
            }
    }
    return 0;
}

private bool Checking(string letter)
{
    // Проверяем использование буквы
    bool est = false;
    for (int i = 0; i < panel1.Controls.Count; i++)
    {
        string two_letter = panel1.Controls[i].Text.ToUpper();
        if (two_letter.Length == 2)
        {
            if (two_letter.Substring(0, 1) == letter)
            {
                est = true;
                break;
            }
            if (two_letter.Substring(1, 1) == letter)
            {
                est = true;
                break;
            }
        }
    }
    return !est;
}
}
}
}

```