

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

Кафедра «Кибербезопасность информационных систем»

Лабораторная работа №2

по дисциплине: «Теоретико-числовые методы в криптографии»

на тему: «Остаток от деления»

Выполнил обучающийся гр. ВКБ42

Михайлов Александр Сергеевич

          Проверила:

Ст. пр. Артамонова Е.А.

Ростов-на-Дону

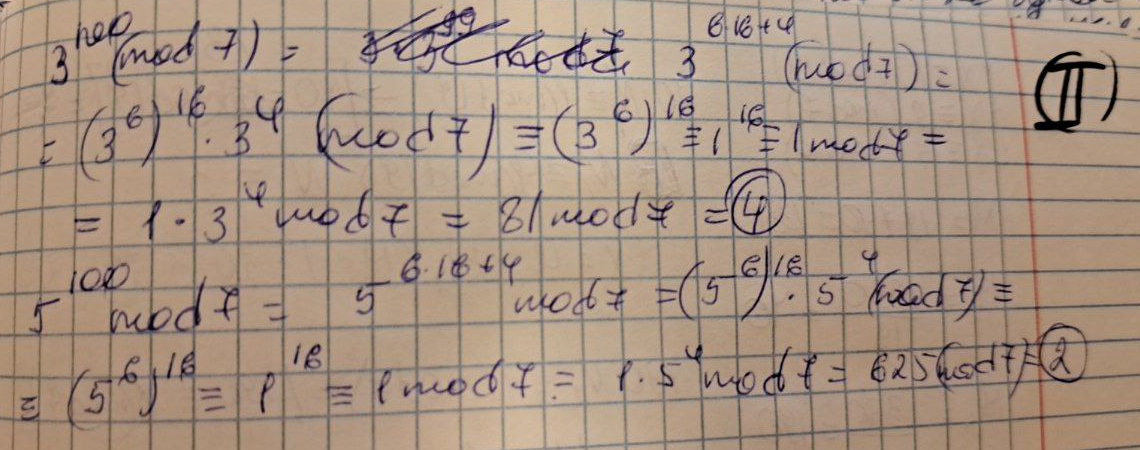
2024

Лабораторная работа №2

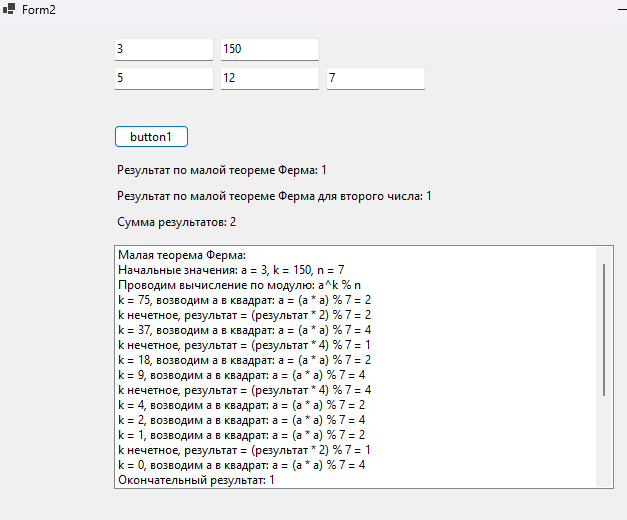
«Остаток от деления.»

Вариант №16

**Ручная реализация.**



**Входные данные.**



**Листинг 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 ФермаИЭйлер

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

// Очищаем listBox1 перед новым вычислением

listBox1.Items.Clear();

// Ввод значений a, k и n (модуль)

int a = int.Parse(textBox1.Text);

int k = int.Parse(textBox2.Text);

int n = int.Parse(textBox3.Text); // n может быть как простым, так и составным

// Ввод значений для второго числа и его степени

int b = int.Parse(textBox4.Text);

int m = int.Parse(textBox5.Text);

// Если n простое, используем малую теорему Ферма

if (IsPrime(n))

{

listBox1.Items.Add("Малая теорема Ферма:");

int result1 = ModPowWithProcess(a, k, n);

label1.Text = $"Результат по малой теореме Ферма: {result1}";

int result2 = ModPowWithProcess(b, m, n);

label2.Text = $"Результат по малой теореме Ферма для второго числа: {result2}";

int sumResult = (result1 + result2) % n;

label3.Text = $"Сумма результатов: {sumResult}";

}

// Если n не простое, используем теорему Эйлера

else

{

listBox1.Items.Add("Теорема Эйлера:");

int phi = EulerTotient(n); // Вычисляем φ(n)

listBox1.Items.Add($"Функция Эйлера φ({n}) = {phi}");

int reducedExponent1 = k % phi; // Приводим степень по модулю φ(n)

listBox1.Items.Add($"Понижаем степень k по модулю φ(n): k = {k} % {phi} = {reducedExponent1}");

int result1 = ModPowWithProcess(a, reducedExponent1, n); // Вычисляем a^reducedExponent % n

label1.Text = $"Результат по теореме Эйлера: {result1}";

int reducedExponent2 = m % phi; // Приводим степень по модулю φ(n)

listBox1.Items.Add($"Понижаем степень m по модулю φ(n): m = {m} % {phi} = {reducedExponent2}");

int result2 = ModPowWithProcess(b, reducedExponent2, n); // Вычисляем b^reducedExponent % n

label2.Text = $"Результат по теореме Эйлера для второго числа: {result2}";

int sumResult = (result1 + result2) % n;

label3.Text = $"Сумма результатов: {sumResult}";

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

// Функция для проверки простоты числа

private bool IsPrime(int number)

{

if (number < 2)

return false;

for (int i = 2; i <= Math.Sqrt(number); i++)

{

if (number % i == 0)

return false;

}

return true;

}

// Возводим a в степень k по модулю n с выводом процесса

private int ModPowWithProcess(int a, int k, int n)

{

int result = 1;

a = a % n;

listBox1.Items.Add($"Начальные значения: a = {a}, k = {k}, n = {n}");

listBox1.Items.Add($"Проводим вычисление по модулю: a^k % n");

while (k > 0)

{

// Если k нечетное, умножаем результат на a

if ((k % 2) == 1)

{

result = (result \* a) % n;

listBox1.Items.Add($"k нечетное, результат = (результат \* {a}) % {n} = {result}");

}

// k делим на 2

k = k >> 1;

a = (a \* a) % n; // Возводим a в квадрат

listBox1.Items.Add($"k = {k}, возводим a в квадрат: a = (a \* a) % {n} = {a}");

}

listBox1.Items.Add($"Окончательный результат: {result}");

return result;

}

// Вычисляем φ(n) - функция Эйлера с выводом процесса

private int EulerTotient(int n)

{

int result = n;

listBox1.Items.Add($"Начальное значение функции Эйлера φ({n}) = {n}");

for (int i = 2; i \* i <= n; i++)

{

if (n % i == 0)

{

listBox1.Items.Add($"Найден делитель: {i}");

while (n % i == 0)

{

n /= i;

listBox1.Items.Add($"Уменьшаем n: n = {n}");

}

result -= result / i;

listBox1.Items.Add($"Обновляем φ(n): φ(n) = {result}");

}

}

if (n > 1)

{

result -= result / n;

listBox1.Items.Add($"n > 1, обновляем φ(n): φ(n) = {result}");

}

return result;

}

private void listBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

}

}

**Вывод по работе:**

В ходе выполнения лабораторной работы были изучены лекционный материал и алгоритм поиска остатка от деления.