МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КУБГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**Отчет**

**по практическому заданию №3**

**по курсу**

**«КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ»**

Работу выполнил

Студенты 46 группы

Нагалевский А.М.

Преподаватель:

Крамаренко А.А.

Краснодар 2024

**Постановка задачи.**

Реализовать интерактивный калькулятор в поле Галуа с заданным пользователем образующим многочленом (сложение, умножение, деление, НОД, возведение в степень и таблица умножения).

Поле Галуа — это произвольное поле, состоящие из конечного числа элементов. — стандартное обозначение полей Галуа, где в скобках указывается количество элементов поля. С точностью до изоморфизма конечное поле полностью определяется его порядком, который всегда является степенью какого-нибудь простого числа, то есть , где p - простое число, а n – любое натурное число. При этом p будет являться характеристикой этого поля.

Рассмотрим также некоторые понятия, которые важны для понимания задачи.

**Неприводимый полином** — это полином, который не может быть разложен на множители меньшей степени с коэффициентами из данного поля. Например, в поле GF(2) неприводимый полином степени, n - это полином степени n без множителей меньшей степени, который не может быть представлен как произведение двух меньших по степени полиномов (рисунок 1).

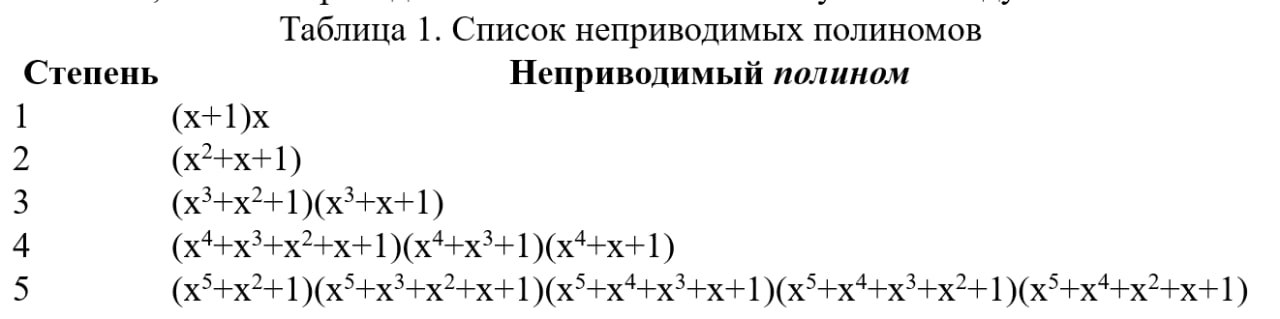


Рисунок 1 – таблица неприводимых полиномов.

**Примитивный полином** в поле Галуа — это неприводимый полином, порождающий мультипликативную группу поля. Русским языком, если все его коэффициенты взаимно просты в совокупности (НОД его коэффициентов равен 1, или по-другому: у этих чисел нет общих делителей).

**Образующий полином** в контексте конечного поля, образующий полином порождает всё поле как пространство над его базовым полем.

Как и было сказано раннее, ключевой аспект полей Галуа заключается в том, что они имеют конечное количество элементов. Это означает, что арифметические операции в этих полях выполняются по модулю числа q.

Существует два основных способа построения полей Галуа:

1. Поле Галуа GF(p), где p - простое число.
2. Расширенное поле Галуа GF(2^m), где m - положительное целое число.

Калькулятор будет совершать операции именно в расширенных полях Галуа.

Теперь рассмотрим, как выполняются операции в полях Галуа:

1. **Сложение и вычитание**: Операции сложения и вычитания в полях Галуа выполняются так же, как и в обычных полях. Результат сложения или вычитания двух элементов также должен принадлежать к этому полю.
2. **Умножение**: Умножение в полях Галуа может быть выполнено с использованием таблицы умножения, которая определена для каждого поля. Результатом будет являться умножение элементов, которые входят в это поле, друг на друга.
3. **Деление**: Деление в полях Галуа может быть выполнено как умножение на обратный элемент. Обратный элемент для каждого элемента также определен в поле Галуа.
4. **Возведение в степень**: Возведение в степень в полях Галуа может быть выполнено путем многократного умножения, которое мы уже разобрали выше.
5. **Наибольший общий делитель (НОД)**: Для полей Галуа с простым числом элементов вычисление НОД может быть выполнено аналогично алгоритму Евклида для целых чисел только с учетом степени поля.

Далее будут представлены примеры работы калькулятора (рисунки 2-4).

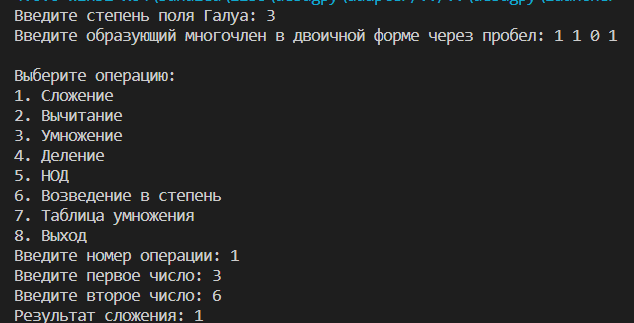


Рисунок 2 – Сложение.

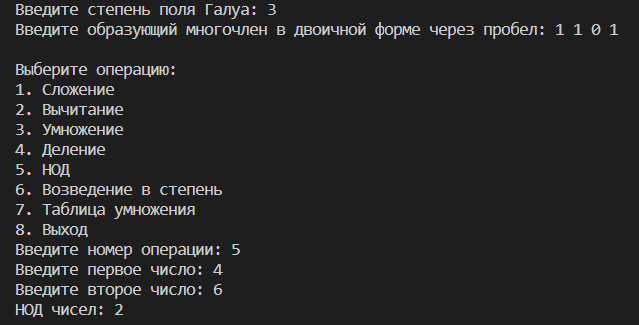


Рисунок 3 – НОД.

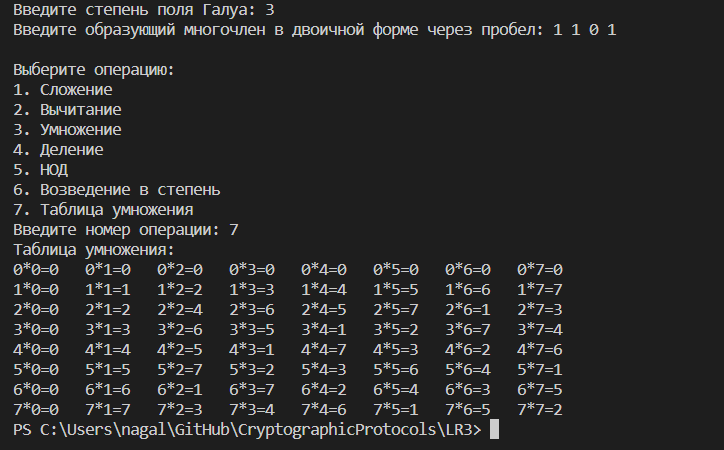


Рисунок 4 – Таблица умножения.

**Текст программы:**

**Файл LR3.py:**

class GaloisFieldCalculator:

    def \_\_init\_\_(self, modulus\_poly, degree):

        self.modulus\_poly = modulus\_poly

        self.degree = degree

    def add(self, a, b):

        return (a + b) % 2

    def subtract(self, a, b):

        return (a - b) % 2

    def multiply(self, a, b):

        result = 0

        while b:

            if b & 1:

                result ^= a

            a <<= 1

            if a & (1 << self.degree):

                a ^= self.modulus\_poly

            b >>= 1

        return result

    def divide(self, a, b):

        quotient = 0

        while a >= b:

            shift = len(bin(a)) - len(bin(b))

            a ^= b << shift

            quotient |= 1 << shift

        return quotient

    def gcd(self, a, b):

        while b:

            a, b = b, a % b

        return a

    def power(self, base, exponent):

        result = 1

        while exponent:

            if exponent & 1:

                result = self.multiply(result, base)

            base = self.multiply(base, base)

            exponent >>= 1

        return result

    def multiplication\_table(self):

        table = [[0] \* 2\*\*self.degree for \_ in range(2\*\*self.degree)]

        for i in range(2\*\*self.degree):

            for j in range(2\*\*self.degree):

                table[i][j] = self.multiply(i, j)

        return table

def parse\_polynomial(polynomial\_str):

    return int(polynomial\_str.replace(" ", ""), 2)

def main():

    degree = int(input("Введите степень поля Галуа: "))

    modulus\_poly\_str = input("Введите образующий многочлен в двоичной форме через пробел: ")

    modulus\_poly = parse\_polynomial(modulus\_poly\_str)

    gf\_calculator = GaloisFieldCalculator(modulus\_poly, degree)

    while True:

        print("\nВыберите операцию:")

        print("1. Сложение")

        print("2. Вычитание")

        print("3. Умножение")

        print("4. Деление")

        print("5. НОД")

        print("6. Возведение в степень")

        print("7. Таблица умножения")

        print("8. Выход")

        choice = int(input("Введите номер операции: "))

        if choice == 8:

            print("Выход...")

            break

        if choice == 1:

            a = int(input("Введите первое число: "))

            b = int(input("Введите второе число: "))

            result = gf\_calculator.add(a, b)

            print("Результат сложения:", result)

        elif choice == 2:

            a = int(input("Введите первое число: "))

            b = int(input("Введите второе число: "))

            result = gf\_calculator.subtract(a, b)

            print("Результат вычитания:", result)

        elif choice == 3:

            a = int(input("Введите первое число: "))

            b = int(input("Введите второе число: "))

            result = gf\_calculator.multiply(a, b)

            print("Результат умножения:", a, "\*", b, "=", result)

        elif choice == 4:

            a = int(input("Введите делимое: "))

            b = int(input("Введите делитель: "))

            result = gf\_calculator.divide(a, b)

            print("Результат деления:", a, "/", b, "=", result)

        elif choice == 5:

            a = int(input("Введите первое число: "))

            b = int(input("Введите второе число: "))

            result = gf\_calculator.gcd(a, b)

            print("НОД чисел:", result)

        elif choice == 6:

            base = int(input("Введите основание: "))

            exponent = int(input("Введите показатель степени: "))

            result = gf\_calculator.power(base, exponent)

            print("Результат возведения в степень:", base, "^", exponent, "=", result)

        elif choice == 7:

            print("Таблица умножения:")

            table = gf\_calculator.multiplication\_table()

            for i in range(2\*\*degree):

                for j in range(2\*\*degree):

                    print(f"{i}\*{j}={table[i][j]}", end="\t")

                print()

            continue

        else:

            print("Неверный выбор операции.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()