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 №5**

**по курсу**

**«КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ»**

Работу выполнил

Студенты 49 группы

Прозоров М.С.

Преподаватель:

Крамаренко А.А.

Краснодар 2024

**Постановка задачи**

Реализовать приложение, позволяющее для заданного модуля находить

все квадратичные вычеты и невычеты по этому модулю.

jacobi\_symbol(a, n):

Проверяет, является ли n нечетным и положительным числом, так как символ Якоби определен только для таких n.

Обрабатывает знак числа a и изменяет знак результата в зависимости от остатка n по модулю 4.

Применяет алгоритм вычисления символа Якоби, который включает в себя деление a на 2 с корректировкой знака результата и взаимную перестановку a и n с последующими проверками их остатков по модулю 4 и 8.

inverse\_mod(a, m):

Использует расширенный алгоритм Евклида для вычисления обратного, изменяя m и a и одновременно вычисляя коэффициенты x0 и x1, которые используются для нахождения обратного по модулю.

Возвращает положительное значение обратного элемента, корректируя результат, если он получился отрицательным.

quadratic\_residue\_modulo(d, m):

Сначала проверяет, есть ли решения, используя символ Якоби. Если (d/m) равно -1, решений нет.

Перебирает все возможные значения x от 0 до m-1 и проверяет, удовлетворяют ли они сравнению, добавляя подходящие x в список решений.

**Вывод**

Реализовал приложение, позволяющее для заданного модуля находить

все квадратичные вычеты и невычеты по этому модулю.

**Листинг:**

**def** jacobi\_symbol(a, n):

"""Вычисление символа Якоби (a/n)"""

**if** n **<=** 0 **or** n **%** 2 **==** 0:

**return** 0

j **=** 1

**if** a **<** 0:

a **=** **-**a

**if** n **%** 4 **==** 3:

j **=** **-**j

while a != 0:

**while** a **%** 2 **==** 0:

a **=** a **//** 2

**if** n **%** 8 **in** [3, 5]:

j **=** **-**j

a, n **=** n, a

**if** a **%** 4 **==** 3 **and** n **%** 4 **==** 3:

j **=** **-**j

a **=** a **%** n

**if** n **==** 1:

**return** j

**else**:

**return** 0

**def** inverse\_mod(a, m):

"""Нахождение обратного элемента для 'a' по модулю 'm'"""

m0, x0, x1 **=** m, 0, 1

while a > 1:

q **=** a **//** m

m, a **=** a **%** m, m

x0, x1 **=** x1 **-** q **\*** x0, x0

return x1 + m0 if x1 < 0 else x1

**def** quadratic\_residue\_modulo(d, m):

"""Решение квадратичного сравнения x^2 ≡ d (mod m)"""

solutions **=** []

**if** jacobi\_symbol(d, m) **==** **-**1:

**return** solutions # Нет решений

**for** x **in** range(m):

**if** (x**\***x **-** d) **%** m **==** 0:

solutions**.**append(x)

**return** solutions

# Пример использования:

d **=** 4

m **=** 21

print(f"Решения x^2 ≡ {d} (mod {m}): {quadratic\_residue\_modulo(d, m)}")