

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 №3

по дисциплине: «Теоретико-числовые методы в криптографии»

на тему: «Сравнение первой степени»

Выполнил обучающийся гр. ВКБ43

Ковалев Данил Петрович

          Проверила:

Ст. пр. Артамонова Е.А.

Ростов-на-Дону

2025

**Цель:** изучить решение сравнений первой степени

**Задание 1.** Решить сравнение первой степени при помощи функции Эйлера. Написать программу, реализующую решение сравнения первой степени.

**Вариант №11**

Ручная реализация:

Изображение выглядит как текст, рукописный текст, блокнот, бумага

Содержимое, созданное искусственным интеллектом, может быть неверным.

Программная реализация:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Листинг программы.**

*"""  
Сравнение первой степени  
"""*from math import gcd  
from typing import Final, Iterable  
from typing import override  
  
from app.core.registry import LogRegistry  
from app.exceptions.models.congruence import NoSolutionSinceGCDDoesNotShareFreeTermError, ReverseElementWasNotFound  
from app.models.comparison\_of\_the\_first\_degree.base import ILinearCongruenceSolver  
  
  
class LinearCongruenceSolver(ILinearCongruenceSolver):  
 def \_\_init\_\_(self) -> None:  
 self.\_logger: Final[LogRegistry] = LogRegistry()  
  
 @override  
 def solve(self, coefficient: int, constant\_term: int, modulus: int) -> int:  
 *"""  
 Решает сравнение первой степени: ax ≡ b (mod m)  
 Возвращает:  
 x: Основное решение  
 None: Если решение не существует  
 """* # Шаг 1: Проверка существования решения  
 gcd\_value: int = gcd(coefficient, modulus)  
 self.\_logger.add\_log(f"НОД({coefficient}, {modulus}) = {gcd\_value}")  
  
 if constant\_term % gcd\_value != 0:  
 raise NoSolutionSinceGCDDoesNotShareFreeTermError(  
 "Решение не существует, так как НОД не делит свободный член."  
 )  
  
 # Шаг 2: Упрощение сравнения  
 simplified\_coefficient: int = coefficient // gcd\_value  
 simplified\_constant: int = constant\_term // gcd\_value  
 simplified\_modulus: int = modulus // gcd\_value  
 self.\_logger.add\_log(  
 f"Упрощенное сравнение: {simplified\_coefficient}x = {simplified\_constant} (mod {simplified\_modulus})"  
 )  
  
 # Шаг 3: Поиск обратного элемента  
 inverse: int | None = self.\_find\_modular\_inverse(simplified\_coefficient, simplified\_modulus)  
 if inverse is None:  
 raise ReverseElementWasNotFound("Обратный элемент не найден.")  
  
 self.\_logger.add\_log(  
 f"Обратный элемент {simplified\_coefficient}⁻¹ по модулю {simplified\_modulus} равен {inverse}"  
 )  
  
 # Шаг 4: Вычисление основного решения  
 base\_solution: int = (simplified\_constant \* inverse) % simplified\_modulus  
 self.\_logger.add\_log(  
 f"Основное решение: x = {base\_solution} (mod {simplified\_modulus})"  
 )  
  
 # Шаг 5: Все решения  
 all\_solutions: list[int] = [base\_solution + k \* simplified\_modulus for k in range(gcd\_value)]  
 self.\_logger.add\_log("Все решения:")  
  
 for solution in all\_solutions:  
 self.\_logger.add\_log(f"x = {solution} (mod {modulus})")  
  
 return base\_solution  
  
 @staticmethod  
 def \_find\_modular\_inverse(number: int, modulus: int) -> int | None:  
 *"""Находит обратный элемент числа number по модулю modulus"""* original\_modulus: int = modulus  
 x0, x1 = 0, 1  
  
 if modulus == 1:  
 return None  
  
 while number > 1:  
 q: int = number // modulus  
 number, modulus = modulus, number % modulus  
 x0, x1 = x1 - q \* x0, x0  
  
 if x1 < 0:  
 x1 += original\_modulus  
  
 return x1 if number == 1 else None  
  
 @override  
 def get\_logs(self) -> Iterable[str]:  
 return self.\_logger.logs

**Вывод по работе:** в ходе выполнения лабораторной работы были изучены лекционный материал и алгоритм для решения сравнений первой степени.