# Лабораторная работа №6

## Математические основы защиты информации и информационной безопасности

Николаев Дмитрий Иванович, НПМмд-02-24

8 ноября 2024

Российский университет дружбы народов имени Патриса Лумумбы, Москва, Россия

# Прагматика выполнения

# Прагматика выполнения

- Освоение алгоритмов разложения чисел на множители — $\rho$-метода Полларда и метода квадратов (Теорема Ферма о разложении).

# Цели

# Цели

Изучить работу алгоритмов разложения чисел на множители: $\rho$-Метод Полларда; Метод квадратов (Теорема Ферма о разложении); а также реализовать их программно.

# Задачи

# Задачи

1. Освоить и реализовать алгоритм $\rho$-метода Полларда на языке Julia;
2. Освоить и реализовать алгоритм метода квадратов (теорема Ферма о разложении) на языке Julia;
3. Разложить на множители некоторое заданное число.

# Выполнение работы

# Алгоритм $\rho$-метода Полларда

```julia
using Random

"""rho-Метод Полларда"""
function Pollard_rho(n::Int, f = x -> x^2 + 1)::Int
    # Начальное значение
    c = rand(1:n-1)
    a = c
    b = c
    # НОД, начальное значение 1
    d = 1
    while d == 1
        # Генерация последовательности
        a = f(a) % n
        b = f(f(b) % n) % n
        # НОД как разница между a и b
        d = gcd(abs(a - b), n)
    end
    if d == n
        return 1 # Делитель не найден
    else
        return d
    end
end
```

# Алгоритм метода квадратов (теорема Ферма о разложении)

```julia
"""Метод квадратов (Теорема Ферма о разложении)"""
function Fermat_Factorization(n::Int)::Tuple{Int, Int}
    # Начальное значение как округлённый корень исходного числа
    s = ceil(Int, sqrt(n))
    # Из соотношение n = s^2 - t^2
    t2 = s^2 - n
    # Пока соотношение не стало точным для целых чисел
    while sqrt(t2) != floor(sqrt(t2))
        s += 1 # Увеличиваем s
        t2 = x^2 - n
    end
    # Вычисляем t
    t = sqrt(t2)
    return (s - t, s + t)
end
```

Рис. 2: Код алгоритма метода квадратов (теорема Ферма о разложении) на Julia

# Начальные данные

```julia
# Пример работы алгоритмов
n_pollard = 1359331  # Число из лабораторной работы для метода Полларда
n_fermat = 15  # Число из лабораторной работы для метода квадратов

println("p-Метод Полларда для числа ", n_pollard)
pollard_factor = Pollard_rho(n_pollard)
println("Нетривиальный делитель: ", pollard_factor)

println("\nМетод квадратов (Теорема Ферма) для числа ", n_fermat)
fermat_factors = Fermat_Factorization(n_fermat)
println("Нетривиальные делители: ", fermat_factors)

println("\nМетод квадратов (Теорема Ферма) для числа ", n_pollard)
fermat_factors = Fermat_Factorization(n_pollard)
println("Нетривиальные делители: ", fermat_factors)
```

**Рис. 3:** Начальные данные для сравнения алгоритмов разложения чисел на множители на Julia

# Результат выполнения кода и сравнения алгоритмов

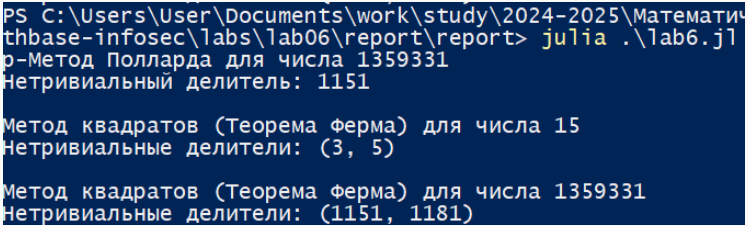


**Рис. 4:** Результат выполнения кода и сравнения алгоритмов разложения чисел на множители на Julia

# Результаты

# Результаты

По результатам работы, я изучил работу алгоритмов разложения чисел на множители: $\rho$-Метод Полларда; Метод квадратов (Теорема Ферма о разложении); а также реализовал их программно.