# Лабораторная работа №5

## Математические основы защиты информации и информационной безопасности

Данилова Анастасия Сергеевна

## Содержание



## Цель работы

Изучить вероятностные алгоритмы проверки чисел на простоту и реализовать их программно на языке Julia.

## Задание

- Изучить теоретическую часть о вероятностных алгоритмах проверки чисел на простоту
- Реализовать вычисление символа Якоби и тесты: Ферма, Соловэя-Штрассена, Миллера-Рабина.

## Теоретическое введение

**Тест Ферма**

При проверке числа на простоту тестом Ферма выбирают несколько чисел a. Чем больше количество a, для которых утверждение истинно, тем больше вероятность, что число n простое. Однако существуют составные числа, для которых данное равенство выполняется для всех a взаимно простых с - это числа Кармайкла. Чисел Кармайкла — бесконечное множество, наименьшее число Кармайкла — 561. Тем не менее, тест Ферма довольно эффективен для обнаружения составных чисел.

**Тест Соловея-Штрассена**

Тест Соловея — Штрассена — вероятностный тест простоты, открытый в 1970-х годах Робертом Мартином Соловеем совместно с Фолькером Штрассеном. Тест всегда корректно определяет, что простое число является простым, но для составных чисел с некоторой вероятностью он может дать неверный ответ. Основное преимущество теста заключается в том, что он, в отличие от теста Ферма, распознает числа Кармайкла как составные.

**Тест Миллера-Рабина**

Тест Миллера — Рабина — вероятностный полиномиальный тест простоты. Тест Миллера — Рабина позволяет эффективно определять, является ли данное число составным. Однако, с его помощью нельзя строго доказать простоту числа. Тем не менее тест Миллера — Рабина часто используется в криптографии для получения больших случайных простых чисел. Алгоритм был разработан Гари Миллером в 1976 году и модифицирован Майклом Рабином в 1980 году.

**Символ Якоби**

Символ Якоби — теоретико-числовая функция двух аргументов, введённая К. Якоби в 1837 году. Является квадратичным характером в кольце вычетов.

Символ Якоби обобщает символ Лежандра на все нечётные числа, большие единицы. Символ Кронекера — Якоби, в свою очередь, обобщает символ Якоби на все целые числа, но в практических задачах символ Якоби играет гораздо более важную роль, чем символ Кронекера — Якоби.

## Выполнение лабораторной работы

```
using Random

function ferm(n::Int)
    a = rand(2:(n-2))
    r = powermod(a, n - 1, n)

    if r == 1
        return "Число $n, вероятно, простое"
    else
        return "Число $n составное"
    end
end

    println("Введите число n")
    n = parse(Int, readline())

    result = ferm(n)
    println(result)
```

Тест Ферма

```
  Activating new project at `C:\Users\nastd\.julia\environments\v1.11`
Введите число n
31
Число 31, вероятно, простое
  Terminal will be reused by tasks, press any key to close it.

  Activating new project at `C:\Users\nastd\.julia\environments\v1.11`
Введите число n
33
Число 33 составное
  Terminal will be reused by tasks, press any key to close it.
```

Результаты

```julia
using Random
function ja_sym(a::Int, n::Int)
    g = 1
    if a == 0
        return 0
    elseif a == 1
        return g
    end
    a = a % n
    while a != 0
        k = 0
        while a % 2 == 0
            a = div(a, 2)
            k += 1
        end

        if k % 2 == 0
            s = 1
        else
            if n % 8 == 1 || n % 8 == 7
                s = 1
            else
                s = -1
            end
        end
        if a == 1
            return g * s
        end
        if n % 4 == 3 && a % 4 == 3
            s = -s
        end
        a = n % a
        n = a
        g *= s
    end
    return g
end
```

Символ Якоби

```
5/7 = -1
  * Terminal will be reused by tasks, press any key to close it.

   Activating new project at `C:\Users\nastd\.julia\environments\v1.11`
Введите a:
4
Введите n:
7
4/7 = 1
```

Результаты

```julia
function s_test(n::Int)
    a = rand(2:(n - 2))
    r = powermod(a, div(n - 1, 2), n)

    if r == 1 || r == n - 1
        return "Число $n, вероятно, простое"
    end

    return (r != ja_sym(a, n) % n) ? "Число $n составное" : "Число $n, вероятно, простое"
end

println("Введите n:")
n = parse(Int, readline())
println(s_test(n))
```

Тест Соловея-Штрассена

```
   Activating new project at `C:\Users\nastd\.julia\environments\v1.11`
Введите n:
31
Число 31, вероятно, простое
  * Terminal will be reused by tasks, press any key to close it.

   Activating new project at `C:\Users\nastd\.julia\environments\v1.11`
Введите n:
33
Число 33 составное
  * Terminal will be reused by tasks, press any key to close it.
```

Результаты

```julia
using Random
function miller_rabin(n::Int, k::Int)
    d = n - 1
    s = 0
    while d % 2 == 0
        d = Int(d / 2)
        s += 1
    end

    for _ in 1:k
        a = rand(2:(n - 2))
        y = powermod(a, d, n)
        if y == 1 || y == n - 1
            continue
        end

        for _ in 1:s
            y = powermod(y, 2, n)
            if y == 1
                return "Число $n составное"
            elseif y == n - 1
                break
            end
        end

        if y != n - 1
            return "Число $n составное"
        end
    end

    return "Число $n простое"
end
println("Введите число:")
n = parse(Int, readline())
k = 10
println(miller_rabin(n, k))
```

Тест Миллера-Рабина

```
  Activating new project at `C:\Users\nastd\.julia\environments\v1.11`
Введите число:
31
Число 31 простое
  * Terminal will be reused by tasks, press any key to close it.

  Activating new project at `C:\Users\nastd\.julia\environments\v1.11`
Введите число:
33
Число 33 составное
  * Terminal will be reused by tasks, press any key to close it.
```

Результаты

## Выводы

Мы изучили вероятностные алгоритмы проверки чисел на простоту и реализовали их программно на языке Julia.

## Список литературы

1. Mathematics // Julia URL: https://docs.julialang.org/en/v1/base/math/