Лабораторная работа №5

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 проверки чисел на простоту — алгоритмов, реализующих тест Ферма, тест Соловэя-Штрассена, и тест Миллера-Рабина

Цели

Цели

Изучить работу алгоритмов проверки чисел на простоту: алгоритм, реализующий тест Ферма; алгоритм вычисления символа Якоби; алгоритм, реализующий тест Соловэя-Штрассена; алгоритм, реализующий тест Миллера-Рабина; а также реализовать их программно.

Задачи

Задачи

1. Освоить и реализовать алгоритм, реализующий тест Ферма на языке Julia;
2. Освоить и реализовать алгоритм, реализующий тест Соловэя-Штрассена на языке Julia;
3. Освоить и реализовать алгоритм, реализующий тест Миллера-Рабина на языке Julia.

Выполнение работы

Алгоритм теста Ферма

```julia
using Random

"""Возведение в степень по модулю a^b mod c"""
function powermod(a::Int, b::Int, c::Int)::Int
    res = 1
    a = a % c
    while b > 0
        if b % 2 == 1
            res = (res * a) % c
        end
        b = div(b, 2)
        a = (a * a) % c
    end
    return res
end

"""функция проверки числа на простоту с помощью теста Ферма"""
function Fermat_Test(n::Int, k::Int = 5)::Bool
    if n == 2
        return true
    elseif n < 2 || n % 2 == 0
        return false
    end
    if n < 5
        error("число должно быть больше 4")
    end
    for _ in 1:k
        # Выбираем случайное число a, 2 <= a <= n-2
        a = rand(2:n-2)
        # Проверяем условие Ферма: a^(n-1) = 1 (mod n)
        if gcd(a, n) != 1 || powermod(a, n-1, n) != 1
            return false # Число составное
        end
    end
    return true # Число, вероятно, простое
end
```

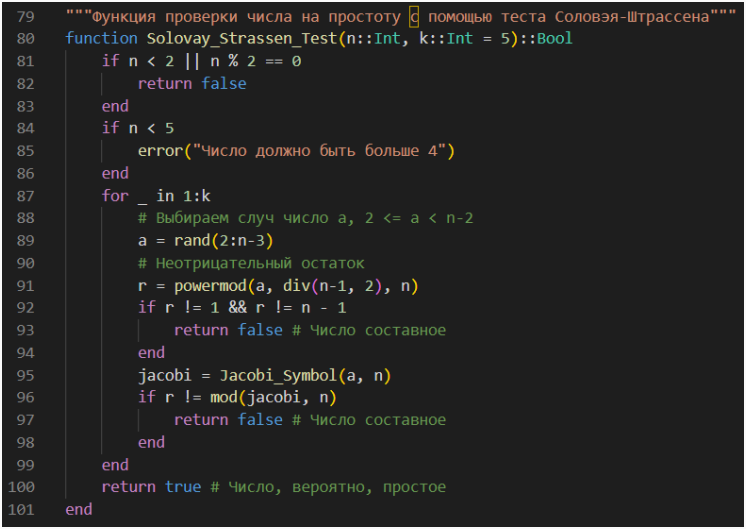
```
38  """Вычисление числа делений на 2"""
39  function Powers_of_Two(a::Int)::Int
40      k = 0
41      while a % 2 == 0
42          k += 1
43          a = div(a, 2)
44      end
45      return k
46  end
```

Рис. 2: Код вычисления количества делений некоторого числа на 2 на Julia

Алгоритм теста Соловэя-Штрассена (2/3)

```
48  """Вычисление символа Якоби (a/n)"""
49  function Jacobi_Symbol(a::Int, n::Int)::Int
50      if n < 3 || n % 2 == 0
51          error("n должно быть положительным нечётным числом большим 2")
52      end
53      if a >= n || a < 0
54          error("a должно быть неотрицательным числом меньшим n")
55      end
56      if gcd(a, n) != 1 return 0 end
57      g = 1
58      while a > 1
59          k = Powers_of_Two(a)
60          s = 1
61          if k % 2 == 1
62              if n % 8 == 3 || n % 8 == 5
63                  s = -1
64              end
65          end
66          a = div(a, 2^k)
67          if a == 1
68              return g*s
69          end
70          if a % 4 == 3 && n % 4 == 3
71              s = -s
72          end
73          a, n, g = n % a, a, g*s
74      end
75      if a == 0 return 0 end
76      return g
77  end
```

Алгоритм теста Соловэя-Штрассена (3/3)



```julia
"""Функция проверки числа на простоту с помощью теста Соловэя-Штрассена"""
function Solovay_Strassen_Test(n::Int, k::Int = 5)::Bool
    if n < 2 || n % 2 == 0
        return false
    end
    if n < 5
        error("Число должно быть больше 4")
    end
    for _ in 1:k
        # Выбираем случ число a, 2 <= a < n-2
        a = rand(2:n-3)
        # Неотрицательный остаток
        r = powermod(a, div(n-1, 2), n)
        if r != 1 && r != n - 1
            return false # Число составное
        end
        jacobi = Jacobi_Symbol(a, n)
        if r != mod(jacobi, n)
            return false # Число составное
        end
    end
    return true # Число, вероятно, простое
end
```

Рис. 4: Код алгоритма теста Соловэя-Штрассена на Julia

Алгоритм теста Миллера-Рабина

```julia
"""Функция проверки числа на простоту с помощью теста Миллера-Рабина"""
function Miller_Rabin_Test(n::Int, k::Int = 5)::Bool
    if n < 2 || n % 2 == 0
        return false
    end
    if n < 5
        error("Число должно быть больше 4")
    end
    # Представляем (n-1) в виде 2^s * r, где число r - нечётное
    s = 0
    r = n - 1
    while r % 2 == 0
        r = div(r, 2)
        s += 1
    end
    for _ in 1:k
        # Выбираем случ число a, 2 <= a < n-2
        a = rand(2:n-3)
        # Вычисляем y = a^r mod n
        y = powermod(a, r, n)
        if y != 1 && y != n - 1
            for _ in 1:(s-1)
                y = powermod(y, 2, n)
                if y == 1
                    return false # Число составное
                end
            end
            if y != n - 1
                return false # Число составное
            end
        end
    end
    return true # Число, вероятно, простое
end
```

```julia
# Тестирование каждого алгоритма на примерах
function Test_Algorithms(n)
    println("Тестирование числа $n на простоту:")

    # Тест Ферма
    fermat_result = Fermat_Test(n)
    println("Тест Ферма: $(fermat_result ? "вероятно простое" : "составное")")

    # Тест Соловэя-Штрассена
    solovay_result = Solovay_Strassen_Test(n)
    println("Тест Соловэя-Штрассена: $(solovay_result ? "вероятно простое" : "составное")")

    # Тест Миллера-Рабина
    miller_rabin_result = Miller_Rabin_Test(n)
    println("Тест Миллера-Рабина: $(miller_rabin_result ? "вероятно простое" : "составное")")
end

# Пример использования: (простое, составное, простое, простое,
# составное, составное, составное, простое, простое)
for n in [13, 15, 17, 19, 561, 1105, 1729, 2143, 2399]
    Test_Algorithms(n)
    println("----------------------------------")
end
```

Рис. 6: Начальные данные для сравнения алгоритмов проверки чисел на простоту на Julia

Результат выполнения кода и сравнения алгоритмов

```
PS C:\Users\User\Documents\work\study\2024-2025\Математичес
thbase-infosec\labs\lab05\report\report> julia .\lab5.jl
Тестирование числа 13 на простоту:
Тест Ферма: вероятно простое
Тест Соловэя-Штрассена: вероятно простое
Тест Миллера-Рабина: вероятно простое
-----------------------------------
Тестирование числа 15 на простоту:
Тест Ферма: составное
Тест Соловэя-Штрассена: составное
Тест Миллера-Рабина: составное
-----------------------------------
Тестирование числа 17 на простоту:
Тест Ферма: вероятно простое
Тест Соловэя-Штрассена: вероятно простое
Тест Миллера-Рабина: составное
-----------------------------------
Тестирование числа 19 на простоту:
Тест Ферма: вероятно простое
Тест Соловэя-Штрассена: вероятно простое
Тест Миллера-Рабина: вероятно простое
-----------------------------------
Тестирование числа 561 на простоту:
Тест Ферма: составное
Тест Соловэя-Штрассена: составное
Тест Миллера-Рабина: составное
-----------------------------------
Тестирование числа 1105 на простоту:
Тест Ферма: составное
Тест Соловэя-Штрассена: составное
Тест Миллера-Рабина: составное
-----------------------------------
Тестирование числа 1729 на простоту:
Тест Ферма: составное
Тест Соловэя-Штрассена: составное
Тест Миллера-Рабина: составное
-----------------------------------
Тестирование числа 2143 на простоту:
Тест Ферма: вероятно простое
Тест Соловэя-Штрассена: вероятно простое
Тест Миллера-Рабина: вероятно простое
-----------------------------------
Тестирование числа 2399 на простоту:
Тест Ферма: вероятно простое
Тест Соловэя-Штрассена: вероятно простое
Тест Миллера-Рабина: вероятно простое
```

Результаты

Результаты

По результатам работы, я изучил работу алгоритмов проверки чисел на простоту: алгоритма, реализующего тест Ферма; алгоритма вычисления символа Якоби; алгоритма, реализующего тест Соловэя-Штрассена; алгоритма, реализующего тест Миллера-Рабина; а также реализовал их программно на языке Julia.