Лабораторная работа 5

Вероятностные алгоритмы проверки чисел на простоту

Климин Никита Денисович

Содержание

1. Цель работы

Изучение и программная реализация вероятностных алгоритмов проверки чисел на простоту: тест Ферма, тест Соловэя--Штрассена, тест Миллера--Рабина.

2. Задание

Реализовать четыре алгоритма проверки числа на простоту, проверить их работу для различных чисел и вывести результаты.

3. Теоретическое введение

- **Простое число** --- это число больше 1, которое не имеет других делителей, кроме 1 и самого себя.

- **Составное число** --- число, которое имеет хотя бы один нетривиальный делитель.

Для проверки чисел на простоту применяются **вероятностные** и **детерминированные** алгоритмы. Вероятностные алгоритмы используют случайное основание и могут с высокой вероятностью определить, является ли число простым.

- **Тест Ферма** основан на малой теореме Ферма: если (p) --- простое число и (1 ≤a < p), то

$$a^{p-1} \equiv 1 \pmod{p}$$

  Если это не выполняется, число составное.

- **Тест Соловэя--Штрассена** использует символ Якоби: для нечетного числа (n > 3) и взаимно простого (a) выполняется

$$a^{(n-1)/2} \equiv \left(\frac{a}{n}\right) \pmod{n}$$

  где ((a/n)) --- символ Якоби.

- **Тест Миллера--Рабина** --- более надёжный вероятностный тест. Представляем

$$n - 1 = 2^s \cdot d$$

  где (d) нечетное, и проверяем условия для случайного основания (a).

- **Символ Якоби** --- обобщение символа Лежандра, используется в тесте Соловэя--Штрассена.

4. Выполнение лабораторной работы

Программа была написана на Julia.

```
function fermat_test(n::Int)
    if n ≤ 5 || iseven(n)
        return "n должно быть нечётным и > 5"
    end
    a = rand(3:n-2)
    return powermod(a, n-1, n) == 1 ? "Число $n, вероятно, простое"
end

function solovay_strassen(n::Int)
    if n ≤ 3 || iseven(n) return "n должно быть нечётным и > 3" end

    function jacobi(a,n)
        r=1; a=mod(a,n)
        while a!=0
            while iseven(a)
                a>>=1
                if n%8==3 || n%8==5 r=-r end
            end
            a,n=n,a
            if a%4==3 && n%4==3 r=-r end
            a=mod(a,n)
        end
        return n==1 ? r : 0
    end
```

```
    a = rand(2:n-2)
    r = powermod(a,(n-1)÷2,n)
    j = jacobi(a,n)
    j = j == -1 ? n-1 : j

    return r==j ? "Число $n, вероятно, простое" : "Число $n составно
end

function miller_rabin(n::Int)
    if n ≤ 5 || iseven(n) return "n должно быть нечётным и > 5" end

    d = n - 1
    s = 0
    while iseven(d)
        d >>= 1
        s += 1
    end

    a = rand(2:n-2)
    x = powermod(a,d,n)
    if x == 1 || x == n-1
        return "Число $n, вероятно, простое"
    end

    for _ in 1:(s-1)
        x = powermod(x,2,n)
        if x == n-1
            return "Число $n, вероятно, простое"
        elseif x == 1
            return "Число $n составное"
        end
    end
```

```
    return "Число $n составное"
end


n = 25


println(fermat_test(n))
println(solovay_strassen(n))
println(miller_rabin(n))
```

Пример работы программы в терминале

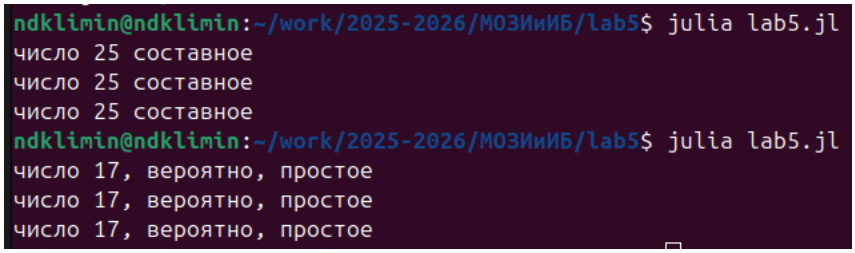

Рис. 4.1.: Пример работы программы

5. Выводы

Все реализованные алгоритмы корректно проверяют простоту чисел. Практическая проверка показала идентичные результаты для всех методов

Список литературы