

# **Информационная безопасность. Отчет по лабораторной работе № 5**

**Вероятностные алгоритмы проверки чисел на простоту**

Мухамеджанов Исматулло Иззатуллоевич

# Содержание

1	Цель работы	5
2	Указание к работе	6
3	Выводы	10
4	Список литературы	11

# List of Figures

2.1	Программа (1)	. . . . .	7
2.2	Программа (2)	. . . . .	8
2.3	Программа (3)	. . . . .	8
2.4	Программа (4)	. . . . .	9

## List of Tables

# 1 Цель работы

Освоить на практике применение вероятностные алгоритмы проверки чисел на простоту

## 2 Указание к работе

Тест Ферма Символ Якоби Тест Соловея-Штрассена Тест Миллера-Рибена # Выполнение лабораторной работы 1. Тест Ферма Вход. Нечетное целое число  $n \geq 5$ . Выход. «Число  $n$ , вероятно, простое» или «Число  $n$  составное».

Выбрать случайное целое число  $a$ ,  $2 \leq a \leq n - 2$ . Вычислить  $r = a^{n-1} \pmod{n}$ . При  $r = 1$  результат: «Число  $n$ , вероятно, простое». В противном случае результат: «Число  $n$  составное».

2. Алгоритм вычисления символа Якоби. Положить  $g \leftarrow 1$ .

При  $p = 0$  результат: 0. При  $p = 1$  результат:  $g$ . Представить  $p$  в виде  $o \cdot 2^k$ , где число  $o$  нечетное. При четном  $k$  положить  $s \leftarrow 1$ , при нечетном  $k$  положить  $s \leftarrow -1$ , если  $p \equiv 1 \pmod{8}$ ; положить  $s \leftarrow -1$ , если  $p \equiv -3 \pmod{8}$ . При  $m = 1$  результат:  $\phi - s$ . Если  $p \not\equiv 3 \pmod{4}$  и  $o \not\equiv 3 \pmod{4}$ , то  $s \leftarrow -s$ . Положить  $p \leftarrow p + m$ ,  $p \leftarrow a$ ,  $g \leftarrow g \cdot s$  и вернуться на шаг 2.

3. Алгоритм реализующий тест Соловея-Штрассена Вход. Нечетное целое число  $p \geq 5$ . Выход. «Число  $p$ , вероятно, простое» или «Число  $p$  составное».

Выбрать случайное целое число  $a$ ,  $2 \leq a \leq p - 2$ . Вычислить  $g \leftarrow a^2 \pmod{p}$ . При  $g \equiv 1$  и  $g \equiv p - 1$  результат: «Число  $p$  составное». Вычислить символ Якоби  $s \leftarrow \left( \frac{a}{p} \right)$ . При  $g \not\equiv s \pmod{p}$  результат: «Число  $p$  составное». В противном случае результат: «Число  $p$ , вероятно, простое». На сегодняшний день для проверки чисел на простоту чаще всего используется тест Миллера-Рабина, основанный на следующем наблюдении. Пусть число  $p$  нечетное и

$p - 1 = 2^s r$ , где  $r$  — нечетное. Если  $p$  простое, то для любого  $a \neq 0 \pmod p$ , взаимно простого с  $p$ , выполняется условие  $a^{p-1} \equiv 1 \pmod p$ .

4. Алгоритм реализующий тест Миллера-Рабина. Вход. Нечетное целое число  $p$ .
5. Выход. «Число  $p$ , вероятно, простое» или «Число  $p$  составное».

Представить  $p - 1$  в виде  $p - 1 = 2^s r$ , где  $r$  нечетное. Выбрать случайное целое число  $a$ ,  $2 \leq a < p - 2$ .

Вычислить  $a^r \pmod p$ . При  $a^r \equiv 1 \pmod p$  и  $a^{2^j r} \equiv 1 \pmod p$  выполнить следующие действия. Положить  $j \leftarrow 1$ . Если  $a^{2^j r} \equiv 1 \pmod p$  и  $a^{2^{j-1} r} \not\equiv 1 \pmod p$ , то Положить  $j \leftarrow j + 1$ . При  $j = s$  результат: «Число  $p$  простое». Положить  $j \leftarrow j - 1$ . При  $a^{2^j r} \not\equiv 1 \pmod p$  результат: «Число  $p$  составное». Результат: «Число  $p$ , вероятно, простое».

```

Тест Ферма

import random
import math

def test_ferm(n):
    if n == 1:
        return "Number is prime"
    number = random.randint(2, n - 2)
    r = math.pow(number, n - 1) % n
    if r == 1:
        return "Number is prime"
    return "Number is not prime"

test_ferm(9)

[4]
... 'Number is not prime'
Python

```

Figure 2.1: Программа (1)

### Символ Якоби

```
def yakobi_symbol(a, n):
    if n <= 0 or n % 2 == 0:
        return "Error: n должно быть положительным нечетным числом"
    a = a % n
    g = 1
    while a != 0:
        while a % 2 == 0:
            a /= 2
            r = n % 8
            if r == 3 or r == 5:
                g = -g
        a, n = n, a
        if a % 4 == n % 4 == 3:
            g = -g
        a %= n
    if n == 1:
        return g
    else:
        return 0

a = 7
n = 11
yakobi_symbol(a, n)
```

[16] Python

Figure 2.2: Программа (2)

### Тест Соловея-Штрассена

```
def solovey_shtrassen(n):
    if n < 5:
        return "Enter another number"
    a = random.randint(2, n-3)
    r = math.pow(a, (n-1)//2) % n
    if r != 1 and r != n-1:
        return "N is not prime"
    s = a / n
    if r % n == s:
        return "N is not prime"
    return "N is most probably prime"

solovey_shtrassen(12)
```

[19] Python

... 'N is not prime'

Figure 2.3: Программа (3)



## Тест Миллера-Рабина

```
def miller_rabin(n):
    if n < 5:
        return "Enter another number"
    r = n - 1
    s = 0
    while r % 2 == 0:
        r = r / 2
        s += 1
    a = random.randint(2, n-3)
    y = math.pow(a, r) % n
    if y != 1 and y != n-1:
        j = 1
        if j <= s - 1 and y != n - 1:
            y = math.pow(y, 2) % n
            if y == 1:
                return "Number is not prime"
            j = j + 1
        if y != n - 1:
            return "Number is not prime"
    return "Number is prime"

miller_rabin(8)
```

[31] Python

... 'Number is not prime'

Figure 2.4: Программа (4)

## **3 Выводы**

Освоены алгоритмы проверки чисел на простоту

## **4 Список литературы**

1. Методические материалы курса