

# **Отчёт по лабораторной работе 5**

**Вероятностные алгоритмы проверки чисел на простоту**

Гурбангельдиев Мухаммет

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Теоретические сведения</b>	<b>6</b>
2.1	Тест Ферма . . . . .	6
2.2	Тест Соловья-Штрассена . . . . .	6
2.3	Тест Миллера-Рабина. . . . .	7
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>4</b>	<b>Выводы</b>	<b>13</b>
<b>5</b>	<b>Список литературы</b>	<b>14</b>

# List of Tables

# List of Figures

3.1	Функция для алгоритма ферма . . . . .	8
3.2	Функция для вычисления бинарного эксп . . . . .	9
3.3	Функция для вычисления Якоби . . . . .	10
3.4	Функция для алгоритма Соловья-Штрассена . . . . .	11
3.5	Функция для алгоритма Миллера-Рабина . . . . .	11
3.6	Результат алгоритмов . . . . .	12

# 1 Цель работы

Реализация алгоритмов Ферма, Соловья-Штрассена, Миллера-Рабина и вычисления Якоби.

## 2 Теоретические сведения

Тестом простоты (или проверкой простоты) называется алгоритм, который, приняв на входе число  $N$ , позволяет либо не подтвердить предположение о составности числа, либо точно утверждать его простоту. Во втором случае он называется истинным тестом простоты. Таким образом, тест простоты представляет собой только гипотезу о том, что если алгоритм не подтвердил предположение о составности числа  $N$ , то это число может являться простым с определённой вероятностью. Это определение подразумевает меньшую уверенность в соответствии результата проверки истинному положению вещей, нежели истинное испытание на простоту, которое даёт математически подтверждённый результат[1].

### 2.1 Тест Ферма

- Вход. Нечетное целое число  $n \geq 5$ .
  - Выход. «Число  $n$ , вероятно, простое» или «Число  $n$  составное».
1. Выбрать случайное целое число  $a$ ,  $2 \leq a \leq n - 2$ .
  2. Вычислить  $r = a^{n-1} \pmod{n}$
  3. При  $r = 1$  результат: «Число  $n$ , вероятно, простое». В противном случае результат: «Число  $n$  составное» [2].

### 2.2 Тест Соловья-Штрассена

- Вход. Нечетное целое число  $n \geq 5$ .

- Выход. «Число  $n$ , вероятно, простое» или «Число  $n$  составное».

1. Выбрать случайное целое число  $a$ ,  $2 \leq a \leq n - 2$ .
2. Вычислить  $r = a^{(\frac{n-1}{2})} \pmod n$
3. При  $r \neq 1$  и  $r \neq n - 1$  результат: «Число  $n$  составное».
4. Вычислить символ Якоби  $s = (\frac{a}{n})$
5. При  $r = s \pmod n$  результат: «Число  $n$ , вероятно, простое». В противном случае результат: «Число  $n$  составное» [3].

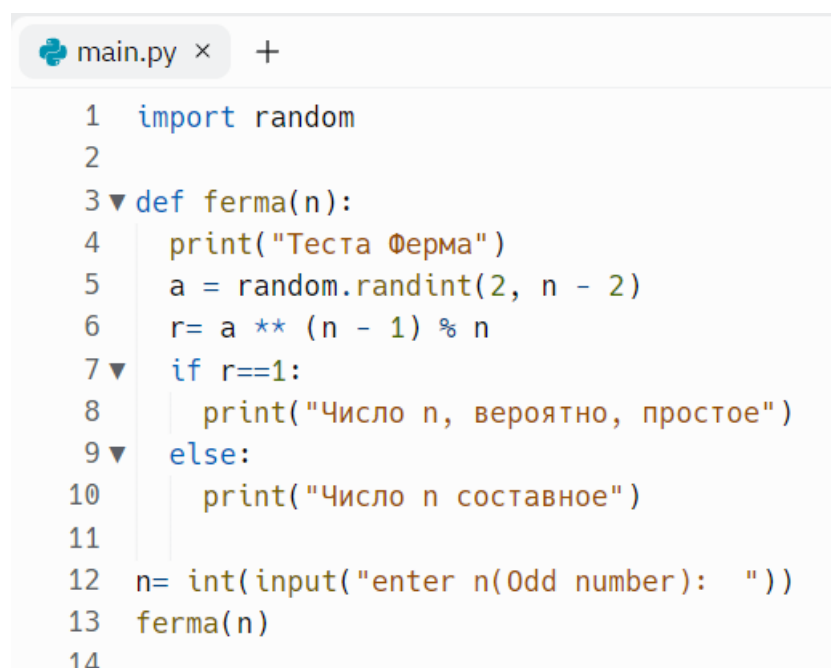
## 2.3 Тест Миллера-Рабина.

- Вход. Нечетное целое число  $n \geq 5$ .
- Выход. «Число  $n$ , вероятно, простое» или «Число  $n$  составное».

1. Представить  $n - 1$  в виде  $n - 1 = 2^s r$ , где  $r$  - нечетное число
2. Выбрать случайное целое число  $a$ ,  $2 \leq a \leq n - 2$ .
3. Вычислить  $y = a^r \pmod n$
4. При  $y \neq 1$  и  $y \neq n - 1$  выполнить действия
  - Положить  $j = 1$
  - Если  $j \leq s - 1$  и  $y \neq n - 1$  то
    - Положить  $y = y^2 \pmod n$
    - При  $y = 1$  результат: «Число  $n$  составное».
    - Положить  $j = j + 1$
  - При  $y \neq n - 1$  результат: «Число  $n$  составное».
5. Результат: «Число  $n$ , вероятно, простое» [4].

### 3 Выполнение лабораторной работы

1. Написал функцию `ferma` для алгоритма ферма. (рис. 3.1)



```
main.py × +  
1 import random  
2  
3 ▼ def ferma(n):  
4     print("Теста Ферма")  
5     a = random.randint(2, n - 2)  
6     r = a ** (n - 1) % n  
7 ▼ if r==1:  
8     print("Число n, вероятно, простое")  
9 ▼ else:  
10     print("Число n составное")  
11  
12 n = int(input("enter n(Odd number): "))  
13 ferma(n)  
14
```

Figure 3.1: Функция для алгоритма ферма

2. Написал функцию `modul` для вычисления бинарного эксп. (рис. 3.2)



```

15 # функция для бинарного эксп
16 ▼ def modul(base, exponent, mod):
17     x = 1
18     y = base
19 ▼     while (exponent > 0):
20 ▼         if (exponent % 2 == 1):
21             x = (x * y) % mod
22
23             y = (y * y) % mod
24             exponent = exponent // 2
25
26     return x % mod
27

```

Figure 3.2: Функция для вычисления бинарного эксп

3. Написал функцию jacobian для вычисления Якоби. (рис. 3.3)

```

29 ▼ def jacobian(a, n):
30 ▼     if (a == 0):
31         return 0
32     ans = 1
33 ▼     if (a < 0):
34         a = -a
35 ▼         if (n % 4 == 3):
36             ans = -ans
37 ▼     if (a == 1):
38         return ans
39 ▼     while (a):
40 ▼         if (a < 0):
41             a = -a
42 ▼             if (n % 4 == 3):
43                 ans = -ans
44 ▼             while (a % 2 == 0):
45                 a = a // 2
46 ▼                 if (n % 8 == 3 or n % 8 == 5):
47                     ans = -ans
48             a, n = n, a
49 ▼             if (a % 4 == 3 and n % 4 == 3):
50                 ans = -ans
51             a = a % n
52 ▼             if (a > n // 2):
53                 a = a - n
54 ▼     if (n == 1):
55         return ans
56     return 0
57

```

Figure 3.3: Функция для вычисления Якоби

4. Написал функцию `solovoy` для алгоритма Соловья-Штрассена. (рис. 3.4)

```

58 ▼ def solovoy(n):
59     print("Тест Соловья-Штрассена")
60     a = random.randrange(2,n-2)
61     r= (a**(n-1/2))%n
62 ▼   if (r != 1 and r!=n-1):
63         print("Число n составное")
64
65     s=jacobian(a,n)
66 ▼   if modul(r,s,n) == 1:
67         print( "Число n составное")
68 ▼   else:
69         print("Число n, вероятно, простое")
70
71 ▼ def toBinary(n):
72     r = []
73 ▼   while (n > 0):
74         r.append(n % 2)
75         n = n / 2
76     return r

```

Figure 3.4: Функция для алгоритма Соловья-Штрассена

5. Написал функцию MillerRabin для алгоритма Миллера-Рабина. (рис. 3.5)

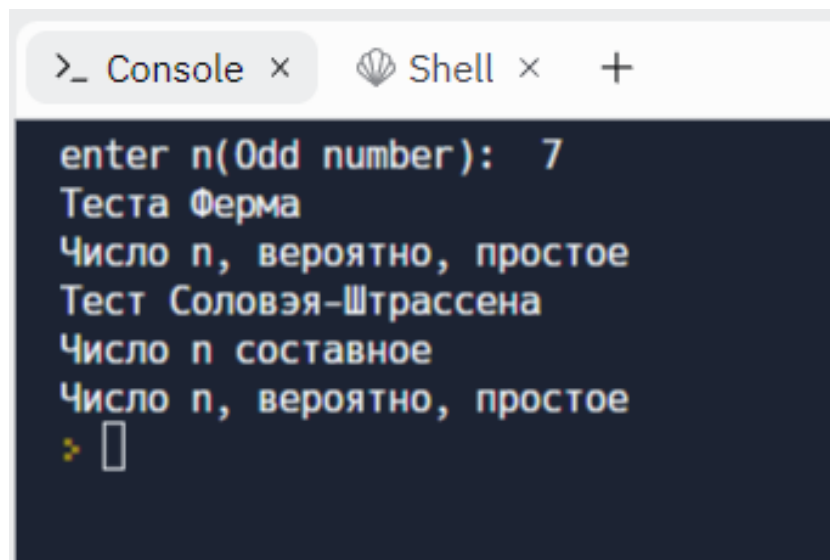
```

78 ▼ def MillerRabin(n, s = 10):
79
80 ▼   for j in range(1, s + 1):
81       a = random.randint(1, n - 1)
82       b = toBinary(n - 1)
83       d = 1
84 ▼     for i in range(len(b) - 1, -1, -1):
85         x = d
86         d = (d * d) % n
87 ▼       if d == 1 and x != 1 and x != n - 1:
88           print("Число n составное") # Составное
89 ▼       if b[i] == 1:
90           d = (d * a) % n
91 ▼       if d != 1:
92           print("Число n составное") # Составное
93           print("Число n, вероятно, простое")
94
95     solovoy(n)
96     MillerRabin(n)

```

Figure 3.5: Функция для алгоритма Миллера-Рабина

6. Получил результат (рис. 3.6)



```
>_ Console x Shell x +
enter n(Odd number): 7
Теста Ферма
Число n, вероятно, простое
Тест Соловья-Штрассена
Число n составное
Число n, вероятно, простое
> [
```

Figure 3.6: Результат алгоритмов

## 4 Выводы

Реализовал алгоритмы Ферма, Соловья-Штрассена, Миллера-Рабина и вычисления Якоби.

## 5 Список литературы

1. Тест простоты [Электронный ресурс] - Режим доступа: [https://ru.wikipedia.org/wiki/Тест\\_простоты](https://ru.wikipedia.org/wiki/Тест_простоты)
2. Тест Ферма [Электронный ресурс] - Режим доступа: [https://ru.wikipedia.org/wiki/Тест\\_Ферма](https://ru.wikipedia.org/wiki/Тест_Ферма)
3. Тест Соловея — Штрассена [Электронный ресурс] - Режим доступа: [https://ru.wikipedia.org/wiki/Тест\\_Соловея\\_—\\_Штрассена](https://ru.wikipedia.org/wiki/Тест_Соловея_—_Штрассена)
4. Тест Миллера — Рабина [Электронный ресурс] - Режим доступа: [https://ru.wikipedia.org/wiki/Тест\\_Миллера\\_—\\_Рабина](https://ru.wikipedia.org/wiki/Тест_Миллера_—_Рабина)