

Математические основы защиты информации и информационной безопасности. Лабораторная работа № 5 на тему “Вероятностные алгоритмы проверки чисел на простоту”

Лубышева Ярослава Михайловна

RUDN University, Moscow, Russian Federation

Содержание

- Цели и задачи
- Выполнение
- Результаты
- Список литературы

Цели и задачи

Выполнить задание к лабораторной работе № 5:

1. Ознакомиться с алгоритмами проверки чисел на простоту: тест Ферма, тест Соловья-Штрассена, тест Миллера-Рабина, также алгоритм вычисления символа Якоби
2. Реализовать все алгоритмы программно

Выполнение

```
import random

# тест Ферма
# вход: нечетное целое число n>=5
# выход: 1 - "Число n, вероятно, простое" или 0 - "Число n составное"
def test_Ferma(n):
    # проверка условий
    if n<5:
        return 0

    # выбрать случайное целое число a, 2 <= a <= n-2
    a = random.randint(2, n-2)
    r = a**(n-1) % n
    if r==1:
        return 1
    else:
        return 0
```

Рис. 1: Программная реализация теста Ферма

```
# алгоритм вычисления символа Якоби
# вход: нечетное целое число n>=3, целое число a - 0<=a<n
# символ Якоби (a/n)
def symbol_Jacobi(n, a):
    # проверка уловий
    if n<3 or a<0 or a>=n:
        return 0

    g = 1
    s = 1

    while True:
        if a==0:
            return 0
        if a==1:
            return g
        a1 = a
        k = 0
        while a1%2==0:
            a1 /= 2
            k += 1
```

Рис. 2: Программная реализация алгоритма вычисления символа Якоби


```
s = 1
if n==1%8 or n==(-1)%8:
    s = 1
if n==3%8 or n==(-3)%8:
    s = -1
if a1==1:
    return g*s
if n==3%4 and a1==3%4:
    s = -s

a = n % a1
n = a1
g = g * s
```

Рис. 3: Программная реализация алгоритма вычисления символа Якоби

```
# тест Соловья-Штрассена
# вход: нечетное целое число n>=5
# выход: 1 - "Число n, вероятно, простое" или 0 - "Число n составное"
def test_Soloway_Strassen(n):
    # проверка условий
    if n<5:
        return 0

    # выбрать случайное целое число a, 2 <= a < n-2
    a = random.randint(2, n-3)
    r = a**((n-1)/2) % n
    if r!=1 and r!=n-1:
        return 0
    s = symbol_Jacobi(n, a)
    if r==s%n:
        return 0
    else:
        return 1
```

Рис. 4: Программная реализация теста Соловья-Штрассена

```
# тест Миллера-Рабина
# вход: нечетное целое число n>=5
# выход: 1 - "число n, вероятно, простое" или 0 - "число n составное"
def test_Miller_Rabin(n):
    # проверка условий
    if n<5:
        return 0

    r = n-1
    s = 0
    while r%2==0:
        r /= 2
        s += 1

    # выбрать случайное целое число a, 2 <= a < n-2
    a = random.randint(2, n-3)
    y = a**r % n

    if y!=1 and y!=n-1:
        j = 1
        if j<=s-1 and y!=n-1:
            y = y**2 % n
            if y==1:
                return 0
            j += 1
        if y!=n-1:
            return 0

    return 1
```

```
# задайте число n>=5
n = 7

# так как алгоритмы вероятностные, проверим их работоспособность на m шагах
m = 1000
k_Ferma = 0
k_Soloway_Strassen = 0
k_Miller_Rabin = 0
for i in range(m):
    k_Ferma += test_Ferma(n)
    k_Soloway_Strassen += test_Soloway_Strassen(n)
    k_Miller_Rabin += test_Miller_Rabin(n)
# если вероятность того, что число простое более 50% - оно простое
for k, name in zip([k_Ferma, k_Soloway_Strassen, k_Miller_Rabin],
                  ['Ферма', 'Соловья-Штрассена', 'Миллера-Рабина']):
    if k > m*0.5:
        print(f'По тесту {name} число n = {n} простое')
    else:
        print(f'По тесту {name} число n = {n} составное')
```

По тесту Ферма число n = 7 простое

По тесту Соловья-Штрассена число n = 7 составное

По тесту Миллера-Рабина число n = 7 простое

Рис. 6: Результаты работы алгоритмов проверки чисел на простоту

Результаты

Выполнено задание к лабораторной работе № 5

Список литературы

1. Методические материалы курса