

Информационная безопасность. Лабораторная работа № 5 на тему “Вероятностные алгоритмы проверки чисел на простоту”

Мухамеджанов Исматулло Иззатуллоевич

RUDN University, Moscow, Russian Federation

Содержание

- Цели и задачи
- Выполнение
- Результаты
- Список литературы

Цели и задачи

Освоить на практике применение вероятностных алгоритмов проверки простоты числа

Выполнение

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] Python

... 'Number is not prime'

Figure 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

... -1

Figure 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 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)
```

[33]

Python

... 'Number is not prime'

Figure 4: Программа (4)

Результаты

Освоено на практике применение вероятностных алгоритмов проверки числа на простоту

Список литературы

1. Методические материалы курса