Лабораторная работа №5. Вероятностные алгоритмы проверки чисел на простоту.

Предмет: Математические основы защиты информации и информационной безопасности

Александр Сергеевич Баклашов

Содержание

List of Figures

1 Цель работы

Рассмотреть и реализовать алгоритмы проверки чисел на простоту.

2 Задание

Реализовать следующие алгоритмы:

- Тест Ферма;
- Нахождение символа Якоби;
- Тест Соловэя-Штрассена;
- Тест Миллера-Рабина.

3 Теоретическое введение

Тест Ферма:

Теория: Основан на малой теореме Ферма, которая утверждает, что если p - простое число, то для любого целого a, не кратного p, справедливо $a^{p-1} \equiv 1(mod\ p)$. Если обратное также верно, то p - простое. Тест Ферма использует эту теорему, проверяя условие для случайно выбранных a.

Нахождение символа Якоби:

Теория: Символ Якоби обобщает символ Лежандра и предоставляет метод определения квадратичной вычетности для нечетных простых чисел. Символ Якоби для числа a и простого нечетного числа p определяется как произведение символов Лежандра для простых множителей a по модулю p. Используется для проверки квадратичной вычетности.

Тест Соловэя-Штрассена:

Теория: Основан на том, что простые числа обладают свойством квадратичной вычетности. Если n - простое, то для любого целого числа a существует квадратный корень по модулю n. Тест Соловэя-Штрассена проверяет это свойство для случайно выбранных a, используя символ Якоби.

Тест Миллера-Рабина:

Теория: Основан на том, что большинство составных чисел обладают свойством “псевдопростоты” по отношению к определенному базису. Тест Миллера-Рабина проверяет это свойство для случайно выбранных базисов. Если число n не проходит тест, то оно с большой вероятностью составное.

Эти тесты предоставляют методы проверки простоты чисел, но важно отметить, что они не гарантируют абсолютную простоту и могут давать ошибочные результаты. В практике их часто комбинируют или применяют с дополнительными проверками для повышения надежности.

4 Выполнение лабораторной работы

4.1 Тест Ферма

4.1.1 Задача

Реализовать тест Ферма

4.1.1.1 Решение

Реализуем тест Ферма (рис. 4.1)

```
n=5
a= np.random.randint (2,n-1)
r=pow(a,n-1) % n
if (r==1):
    print ("Число, вероятно, простое")
else:
    print ("Число, вероятно, составное")
```

```
Число, вероятно, простое
```

Figure 4.1: Тест Ферма

4.2 Нахождение символа Якоби

4.2.1 Задача

Реализовать алгоритм нахождения символа Якоби

4.2.1.1 Решение

Найдём символ Якоби (рис. 4.2)

[3]:
```
n=11
a= np.random.randint (0,n)
print ("a =",a)
g=1
d=1
def representation(n):
    k = 0
    a1 = n
    while a1 % 2 == 0:
        k += 1
        a1 //= 2
    return k, a1

while (d==1):
    if (a==0):
        print ("Символ Якоби =",0)
        break
    if (a==1):
        print ("Символ Якоби =",g)
        break
    k, a1 = representation(a)
    if (k%2==0):
        s=1
    if (k%2!=0):
        if (n%8==1 or n%8==-1):
            s=1
        if (n%8==3 or n%8==-3):
            s=-1
    if (a1==1):
        print ("Символ Якоби =", g*s)
        break
    if (n%4==3 and a1%4==3):
        s=-s
    a=n%a1
    n=a1
    g=g*s
```

```
a = 4
Символ Якоби = 1
```

Figure 4.2: Символ Якоби

4.3 Тест Соловэя-Штрассена

4.3.1 Задача

Реализовать тест Соловэя-Штрассена

4.3.2 Решение

Реализуем тест Соловэя-Штрассена (рис. 4.3)

```
n=11
a= np.random.randint (2,n-2)
r=pow(a,(n-1)/2) % n
if (r!=1) and (r!=n-1):
    print ("Число ", n," составное")
else:
    s = a/n
    if (r==s%n):
        print ("Число ", n," составное")
    else:
        print ("Число ", n," вероятно, простое")
```

```
Число  11  вероятно, простое
```

Figure 4.3: Тест Соловэя-Штрассена

4.4 Тест Соловэя-Штрассена

4.4.1 Задача

Реализовать тест Соловэя-Штрассена

4.4.2 Решение

Реализуем тест Соловэя-Штрассена (рис. 4.4)

```
def representation(n):
    s = 0
    r = n - 1  # Начнем с максимально возможного нечётного r, который равен n - 1
    while r % 2 == 0:
        r //= 2
        s += 1
    return s, r

n = 13
s, r = representation(n)
a= np.random.randint (2,n-2)
y=pow(a,r) % n
while (y!=1 and y!=n-1):
    j=1
    if (j<=n-1 and y!=n-1):
        y=(y*y) %n
        if (y==1):
            print ("Число ",n," составное")
            raise SystemExit("Stop right there!")
        j+=1
    if (y != n-1):
        print ("Число ",n," составное")
        raise SystemExit("Stop right there!")
print ("Число ",n," простое")
```

```
Число  13  простое
```

Figure 4.4: Тест Соловэя-Штрассена

5 Выводы

В ходе данной лабораторной работы я рассмотрел и реализовал следующие алгоритмы:

- Тест Ферма;
- Нахождение символа Якоби;
- Тест Соловэя-Штрассена;
- Тест Миллера-Рабина.

6 Библиография

1. Python documentation. [Электронный ресурс]. М. URL: Python documentation (Дата обращения: 28.09.2023).

2. Лабораторная работа №5. Вероятностные алгоритмы проверки чисел на простоту. - 5 с. [Электронный ресурс]. М. URL: Лабораторная работа №5. Вероятностные алгоритмы проверки чисел на простоту. (Дата обращения: 10.11.2023).