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)

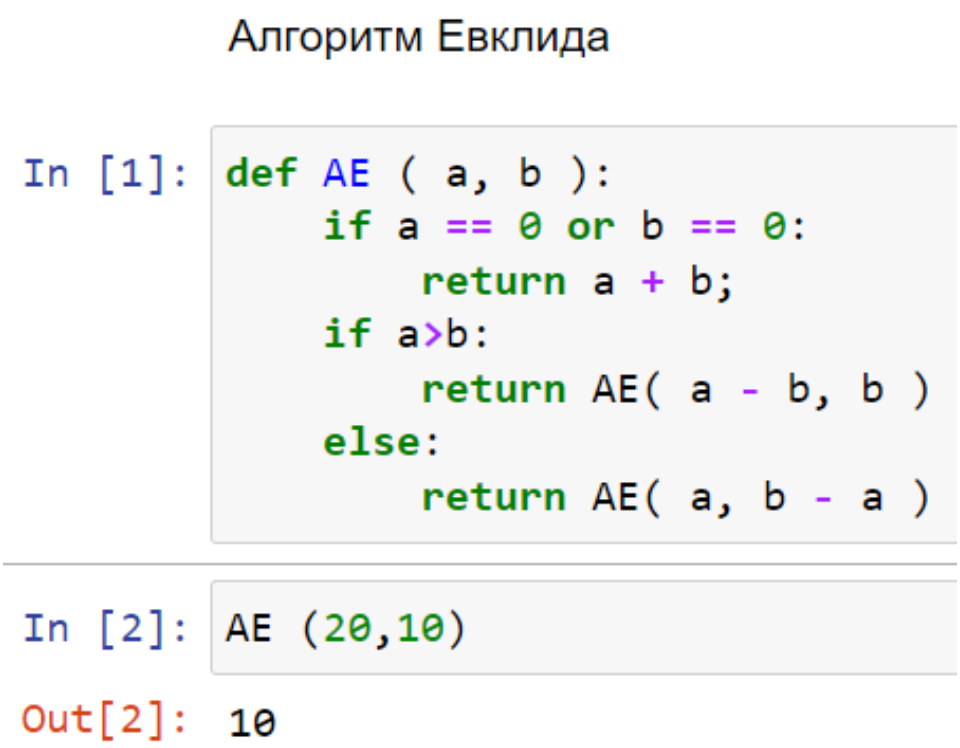

Figure 4.1: Тест Ферма

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)

Бинарный алгоритм Евклида

In [3]:
```
def BAE ( a, b ):
    g=1
    while True:
        if a%2==0 and b%2==0:
            a = a/2
            b = b/2
            g = g*2
        else:
            u = a
            v = b
            break
    while (u!=0):
        while (u%2 == 0):
            u = u/2
        while (v%2 == 0):
            v = v/2
        if u >= v:
            u = u-v
        else:
            v = v-u
        d = g*v
    return d
```

In [4]:
```
BAE (20,10)
```

Out[4]:
```
10.0
```

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)

Расширенный алгоритм Евклида

```
In [5]: def RAE(a, b):
            if b == 0:
                return a, 1, 0

            x1, x0, y1, y0 = 1, 0, 0, 1
            while b > 0:
                q = a // b
                a, b = b, a % b
                x1, x0 = x0, x1 - q * x0
                y1, y0 = y0, y1 - q * y0

            return a,x1,y1

In [6]: RAE(20, 10)

Out[6]: (10, 0, 1)
```

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)

Расширенный бинарный алгоритм Евклида

In [7]:
```
def RBAE ( a, b ):
    g=1
    while True:
        if a%2==0 and b%2==0:
            a = a/2
            b = b/2
            g = g*2
        else:
            u = a
            v = b
            A = 1
            B = 0
            C = 0
            D = 1
            break
    while (u!=0):
        while (u%2 == 0):
            u = u/2
            if A%2==0 and B%2==0:
                A=A/2
                B=B/2
            else:
                A=(A+b)/2
                B=(B-a)/2
        while (v%2 == 0):
            v = v/2
            if C%2==0 and D%2==0:
                C=C/2
                D=D/2
            else:
                C=(C+b)/2
                D=(D-a)/2
        if u >= v:
            u = u-v
            A = A-C
            B = B-D
        else:
            v = v-u
            C = C-A
            D = D-B
        d = g*v
        x = C
        y = D
    return d,x,y
```

In [8]:
```
RBAE (20, 10)
```

Out[8]: (10.0, 0, 1)

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).