Лабораторная работа №5

Математические основы защиты информации и информационной безопасности

Колчева Юлия Вячеславовна

Содержание

List of Tables

List of Figures

1 Цель работы

Познакомиться с вероятностными алгоритмами проверки чисел на простоту.

2 Задание

Реализовать вероятностные алгоритмы проверки чисел на простоту.

3 Выполнение лабораторной работы

Данная работа была выполнена на языке Julia.

Для реализации теста Ферма была написана следующая программа (рис. 3.1) :

In [16]:
```
1  using Random
2  num = 20
3  k = 10
4
5  function Ferma(n, k)
6      for i in 1:k
7          a = rand(1:n-1)
8          if (a^(n - 1) % n != 1)
9              return "Число составное"
10         end
11     end
12     return "Число простое"
13 end
14
15 println(Ferma(num, k))
```

```
Число составное
```

Figure 3.1: Программа реализации теста Ферма

В данной программе:

2-3 строки: задание числа, которое нужно проверить на простоту, и количество проверок

5: задание функции

6: цикл проверки выполняется k раз

7: берём случайное число a в диапазоне [1,n-1]

8: проводим проверку условия, при невыполнении сразу завершаем работу.

9-13: выводим результат, закрываем функцию

Мы можем видеть результат на (рис. 3.1) . Программа работает верно.

Для реализации поиска символа Якоби была написана следующая программа (рис. 3.2)

Figure 3.2: Программа поиска символа Якоби

В данной программе:

1-5 строки: задание функции, проверка условий для вычисления символа Якоби

6-25: реализация алгоритма: проверка трёх условий и действия согласно этим условиям: смена знака символа при четном и нечетном k, проверка остатоков от деления.

26: вывод результата работы программы. В данном случае я вычислила символ Якоби 7 и 33. Вывод представлен на скриншоте.

Для реализации теста Соловэя-Штрассена была написана следующая программа (рис. 3.3)

In [4]:

```
1  using Random
2
3  function S_Sh(n, k)
4      for i in 1:k
5          a = rand(2:(n - 3))
6          r = a^((n - 1) ÷ 2) % n
7          if r != 1 && r != n - 1
8              return "Число составное"
9          end
10         s = jacobi(n, a)
11         if r == s % n
12             return "Число составное"
13         end
14     end
15     return "Число простое"
16 end
17
18 println(S_Sh(num, k))
19
20
```

```
Число составное
```

Figure 3.3: Программа теста Соловэя-Штрассена

3 строка: зададим функцию

4: повторим проверку k раз

5-16: реализация теста: генерируем случайное число a, вычисляем число r по формуле в строке 6, а затем проверяем получившееся значение на два условия. Если оно не проходит проверку, то сразу заканчиваем работу программы. Далее следует ещё одна проверка условия строки 11, при провале также заканчиваем работу.

18: вывод на экран. Результат работы программы с числом 20 и 10 проверками.

Для реализации теста Миллера-Рабина была написана следующая программа (рис. 3.4)

```
3  function miller_rabin(n, k)
4      if n == 2
5          return "Число простое"
6      end
7      if n % 2 == 0
8          return "Число составное"
9      end
10     r, s = 0, n - 1
11     while s % 2 == 0
12         r += 1
13         s ÷= 2
14     end
15     for _ in 1:k
16         a = rand(2:(n - 1))
17         x = powermod(a, s, n)
18         if x == 1 || x == n - 1
19             continue
20         end
21         for _ in 1:(r - 1)
22             x = powermod(x, 2, n)
23             if x == n - 1
24                 break
25             else
26             return "Число составное"
27             end
28         end
29     end
```

Figure 3.4: Программа теста Миллера-Рабина

В данной программе:

3 строка: задаём функцию.

4-9: отсеивание числа 2 и остальных чётных чисел.

11-29: реализация алгоритма: выбираем случайное число a и вычисляем число x по формуле в строке 17. При условии в строке 18 выполняем дополнительные действия как вычисление остатка от деления квадрата x на проверяемое число. Если число прошло все проверки k раз, мы определяем его как “вероятно, простое”.

Результаты работы программы с числом 20 и числом проверок 10 (рис. 3.5)

```
    end
    return "Число простое"
end

println(miller_rabin(num, k))


```

```
Число составное
```

Figure 3.5: Вывод теста Миллера-Рабина

Выводы всех программ (рис. 3.6)

```
In [17]:
println(Ferma(num, k))
println("Символ Якоби ", jacobi(7, 33))
println(S_Sh(num, k))
println(miller_rabin(num, k))
```

```
Число составное
Символ Якоби -1
Число составное
Число составное
```

Figure 3.6: Результаты работы всех программ

Алгоритмы вывели верный результат.

4 Выводы

Познакомилась с вероятностными алгоритмами проверки чисел на простоту и реализовала их на практике.

5 Список литературы

Лабораторная работа №5

Вероятностные алгоритмы проверки чисел на простоту [Электронный ресурс]. URL: https://esystem.rudn.ru/mod/folder/view.php?id=1150976