1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. Институт прикладной математики и механики
5. **Кафедра «Информационная безопасность компьютерных систем»**

**ЛАБОРАТОРНАЯ РАБОТА № 2**

1. по дисциплине «Теоретико-числовые методы в криптографии»
2. Выполнил
3. студент гр. 33636/1 Малинко А.В.

<*подпись*>

1. Преподаватель
2. Павленко Е.Ю.

<*подпись*>

1. Санкт-Петербург
2. 2018

Оглавление

[Цель работы. 3](#_Toc533156389)

[Теоретические сведения 3](#_Toc533156390)

[Результаты работы. 4](#_Toc533156391)

[Исходные числа: 4](#_Toc533156392)

[1. Тест Ферма. 4](#_Toc533156393)

[Число 1: 4](#_Toc533156394)

[Число 2: 5](#_Toc533156395)

[Число 3: 5](#_Toc533156396)

[Число 4: 5](#_Toc533156397)

[2. Тест Соловея– Штрассена. 5](#_Toc533156398)

[Число 1: 5](#_Toc533156399)

[Число 2: 6](#_Toc533156400)

[Число 3: 6](#_Toc533156401)

[Число 4: 6](#_Toc533156402)

[3. Тест Рабина– Миллера. 7](#_Toc533156403)

[Число 1: 7](#_Toc533156404)

[Число 2: 7](#_Toc533156405)

[Число 3: 8](#_Toc533156406)

[Число 4: 8](#_Toc533156407)

[Числа Кармайкла. 8](#_Toc533156408)

[1.Тест Ферма. 8](#_Toc533156409)

[2.Тест Соловея – Штрассена. 9](#_Toc533156410)

[3.Тест Рабина– Миллера. 9](#_Toc533156411)

[Сравнение времени работы алгоритмов. 10](#_Toc533156412)

[Выводы. 10](#_Toc533156413)

# Цель работы.

Каждое из указанных чисел проверить на простоту с помощью тестов: Ферма, Соловея-Штрассена, Рабина-Миллера.

# Теоретические сведения

Существует 2 вида алгоритмов проверки чисел на простоту: детерминированный и вероятностный. Детерминированный всегда действует по одной и той же схеме и гарантированно решает поставленную задачу. Вероятностный алгоритм использует генератор случайных чисел и дает не гарантированно точный ответ.

Для проверки числа на простоту вероятностным алгоритмом выбирается случайное число и проверяется условие алгоритма. Если искомое число n не проходит тест по основанию a, то число n является составным. Если же число nпроходит тест по основанию a, то нельзя сказать точно, является ли число n простым. Увеличив количество проверок для различных оснований и получив для каждого из них, что число n, вероятно, простое, можно сказать, что число nявляется простым с вероятностью, близкой к 1.

Рассмотрим несколько вероятностных алгоритмов:

• Тест Ферма

Согласно малой теореме Ферма, для простого числа и произвольного целого числа выполняется сравнение . Следовательно, если для нечетного существует такое целое , что и , то число n составное.

Однако алгоритм может определить составное число как простое. Псевдопростые числа Ферма — составные числа, проходящие тест Ферма по основанию a.

Числа Кармайкла – нечетные составные числа, для которых сравнение выполняется при любом a ) взаимно простом с n. Для этих чисел тест Ферма выдает результат «Число n, вероятно, простое».

• Тест Соловея-Штрассена

По критерию Эйлера нечетное число n является простым тогда и только тогда, когда для любого целого числа , взаимно простого с n, выполняется сравнение .

Символ Якоби . Если число n – простое, то символ Якоби является символом Лежандры.

Число n называется эйлеровым псевдопростым, если выполняется сравнение .

Для теста Соловэя-Штрассена не существует чисел, подобных числам Кармайкла, то есть составных чисел, которые были бы эйлеровыми псевдопростыми по всем основаниям a.

• Тест Миллера-Рабина

Опирается на проверку ряда равенств, выполняющихся для простых чисел. Если хотя бы одно такое равенство не выполняется, то это доказывает, что число составное. Для теста Рабина-Миллера используется следующее утверждение:

n – простое число,

Тогда выполняется одно из условий:

Если это утверждение выполняется для некоторых a и n, то число n называют вероятно простым, а число а называют свидетелем простоты.

Числа Кармайкла не проходят тест Миллера-Рабина, поэтому данный тест является универсальным.

Если число a является свидетелем простоты составного нечётного числа n по Миллеру-Рабину, то число n называется сильно псевдопростым по основанию a, а также оно является псевдопростым Ферма по основанию a.

# Результаты работы.

## Исходные числа:

1)19540818305052663929

2)4182626256901165713180947335409849194453

3)565674746577950549441720540453387249813

4)40319329928699692919180302866991670347533839708463260094733186934134923674660447

## 1. Тест Ферма.

## Число 1:

19540818305052663929

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 956667830403894148 | Probably simple |
| 2 | 13019030546290591503 | Probably simple |
| 3 | 11298853341773125934 | Probably simple |
| 4 | 8146363972383220152 | Probably simple |
| 5 | 9550759927124991613 | Probably simple |

Probably simple with error probability: 7.888609052210118e-31. Base: 15962795860219028900

## Число 2:

4182626256901165713180947335409849194453

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 3586034329369793452612904844782371765596 | Probably simple |
| 2 | 1009549633061393897973042489702657766838 | Probably simple |
| 3 | 3138404346316725346216631682527413974436 | Probably simple |
| 4 | 3226840867573773315327442680317102109809 | Probably simple |
| 5 | 1980578712346393860542276331502748897054 | Probably simple |

Probably simple with error probability: 7.888609052210118e-31.

Base: 2207207233803273833729986335678754088166

## Число 3:

565674746577950549441720540453387249813

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 393404923678338680917443574217944427874 | Not simple |
| 2 | Дальнейшая работа алгоритма не требуется |  |

Число 4: 40319329928699692919180302866991670347533839708463260094733186934134923674660447

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 5814788766888801018451452964973312118237585281917754265801529366026258538131734 | Not simple |
| 2 | Дальнейшая работа алгоритма не требуется |  |

## 2. Тест Соловея– Штрассена.

## Число 1:

19540818305052663929

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 14977349960524070912 | Probably simple |
| 2 | 6909464394120986172 | Probably simple |
| 3 | 18090868968797353164 | Probably simple |
| 4 | 13326311802720901094 | Probably simple |
| 5 | 2320113541580280462 | Probably simple |

Probably simple with error probability: 7.888609052210118e-31

Base: 12915011159006681352

## Число 2:

4182626256901165713180947335409849194453

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 1833367784456119494416644056300527383662 | Probably simple |
| 2 | 2924805446034102508869587882960529553037 | Probably simple |
| 3 | 1895283743123253335351348004198081863322 | Probably simple |
| 4 | 2833538287337816965322288383495175318556 | Probably simple |
| 5 | 2448891816106882979840828276751283877009 | Probably simple |

Probably simple with error probability: 7.888609052210118e-31

Base: 3021193477844041246338122120869360497079

## Число 3:

565674746577950549441720540453387249813

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 16951758022390569932961015856446491092 | Not a simple |
| 2 | Дальнейшая работа алгоритма не требуется |  |

Число 4: 40319329928699692919180302866991670347533839708463260094733186934134923674660447

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 7422883791371660117045142012293016583101583233639226614139991357261129578391899 | Not a simple |
| 2 | Дальнейшая работа алгоритма не требуется |  |

## 3. Тест Рабина– Миллера.

## Число 1:

19540818305052663929

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 3143985418937977440 | Probably simple |
| 2 | 1354121901112974126 | Probably simple |
| 3 | 4601076997822579380 | Probably simple |
| 4 | 15786483485081983374 | Probably simple |
| 5 | 4357817066979340047 | Probably simple |

Is simple with error probability: 6.223015277861142e-61

Base: 12417293620492642042

S = 3; d = 2442602288131582991; r=1; a = 12417293620492642042

## Число 2:

4182626256901165713180947335409849194453

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 1486089108386388502787160168634233534348 | Probably simple |
| 2 | 3729836029731065098732790608728516962361 | Probably simple |
| 3 | 3876565612468181058251850180999294612054 | Probably simple |
| 4 | 841347521941684761621807290130029330959 | Probably simple |
| 5 | 1991746423492710512418930795293379720257 | Probably simple |

Is simple with error probability: 6.223015277861142e-61

Base: 629432205024905318009470515643953447371

S= 2; d= 1045656564225291428295236833852462298613; r= 1;

A = 3876565612468181058251850180999294612054(№3)

## Число 3:

565674746577950549441720540453387249813

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 529759156469947080144571388271901632301 | Not simple |
| 2 | Дальнейшая работа алгоритма не требуется |  |

Ни одно из условий: ; не выполняется

## Число 4:

40319329928699692919180302866991670347533839708463260094733186934134923674660447

|  |  |  |
| --- | --- | --- |
| № | Основание | Рез. |
| 1 | 6972110030604349746087550552023119991788135302671623190288633956774332293338614 | N.S. |
| 2 | Дальнейшая работа алгоритма не требуется |  |

Ни одно из условий: ; не выполняется

На вход каждого алгоритма подается проверяемое число и k- количество раундов, в каждом из которых случайным образом выбирается число *a*, которое считается основанием.

Во всех тестах k = 100.

# Числа Кармайкла.

Источник: <http://www.chalcedon.demon.co.uk/rgep/cartable.html>

Число1: 32809426840359564991177172754241=

13\*17\*19\*23\*29\*31\*37\*41\*43\*61\*67\*71\*73\*97\*127\*199\*281\*397.

Числа2: 2810864562635368426005268142616001=

13\*17\*19\*23\*29\*31\*37\*41\*43\*61\*67\*71\*73\*109\*113\*127\*151\*281\*353.

## 1.Тест Ферма.

Число 1.

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 31286462256887004853191348795649 | Probably simple |
| 2 | 6352620771303396150798478107388 | Probably simple |
| 3 | 7313629255840884093450265370925 | Probably simple |
| 4 | 9731932811787450445651252153890 | Probably simple |
| 5 | 11382897603639370946799258582369 | Probably simple |

Base: 27916887064037318503445715094309

Not simple, gcd!= 1 (Не выполнилось условие )

Число 2.

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 2026753764377848673623470379811062 | Probably simple |
| 2 | 2003246113433976141477824705853146 | Probably simple |
| 3 | 981755957474004337233620588494379 | Probably simple |
| 4 | 895897244009828214515954415718792 | Not a simple |

Not simple, gcd!= 1 (Не выполнилось условие )

## 2.Тест Соловея – Штрассена.

Число 1.

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 10731043408170042992124011954578 | Not a simple |

Not simple, gcd!= 1 (Не выполнилось условие )

Число 2.

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 2671882275680548346068565045045567 | Not a simple |

Not simple, gcd!= 1 (Не выполнилось условие )

## 3.Тест Рабина– Миллера.

Число 1.

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 7932866420349382156567736606194 | Not a simple |

Ни одно из условий: ; не выполняется

Число 2.

|  |  |  |
| --- | --- | --- |
| № | Основание | Результат |
| 1 | 2595614913152724600385183370129056 | Not a simple |

Ни одно из условий: ; не выполняется

# Сравнение времени работы алгоритмов.

Каждый алгоритм запускается в цикле 1000 раз. Количество раундов – k равняется 100, чтобы тесты проводились при наиболее часто используемых параметрах.

|  |  |  |  |
| --- | --- | --- | --- |
| Число | Тест Ферма | Тест Соловея-Штрассена | Тест Рабина-Миллера |
| 1 | 2.69217676 | 13.23332015 | 8.83250721 |
| 2 | 6.31900374 | 41.70174042 | 16.88890838 |
| 3 | 0.05831965 | 0.43697007 | 0.26106933 |
| 4 | 0.20558277 | 1.28085580 | 0.52677071 |
| Км1 | 0.07390343 | 0.30721589 | 0.33706359 |
| Км2 | 0.10094216 | 0.28936071 | 0.34035322 |

# Выводы.

В результате проделанной работы, были реализованы три алгоритма тестирования на простоту: тест Ферма, Соловея-Штрассена и Рабина-Миллера. Самым простым в реализации, а также самым быстрым оказался тест Ферма. Вероятность ошибки данного метода –, где k – количество раундов. Уже при 100 случайных числах мы получим вероятность ошибки 10^(-(10^29.58160933874379)), что довольно точно. Такая же вероятность ошибки будет и у теста Соловея – Штрассена, однако он не ошибается при определении простоты чисел Кармайкла. На практике я столкнулся с трудностью понимания проблемы чисел Кармайкла, т.к, если добавить в тест ферма проверку на НОД(random, n), то все случайные числа, не являющиеся взаимно простыми с n, будут вести к выводу “Not a simple”. А поскольку чисел, взаимно простых с n – φ(n), то вероятность того, что из 100 случайных чисел не найдется хоть одного, не взаимно простого с n, мала.

Больше всего мне понравился алгоритм Рабина– Миллера, хоть я и ожидал от него более высокой скорости работы. Главный плюс данного алгоритма, на мой взгляд, меньшая вероятность ошибки.

Свидетелем простоты n называют такое число a, для которого выполняется хотя бы одно условие теста Рабина– Миллера. При случайном подборе чисел a, мы, конечно, можем выбрать такое нечетное составное число a, которое посчитает составное n простым, однако таких чисел не больше φ(n)/4 (по теореме Рабина), а значит и вероятность выбора такого числа будет меньше ¼. На практике же я столкнулся с тем, что для первого же числа моего варианта тест Рабина– Миллера работал безошибочно даже при выборе всего одного случайного числа. Я увеличивал количество повторений теста с выбором одного случайного числа до тех пор, пока тест не стал длиться более 10 минут. Однако потом я проделал то же самое с числами 25 и 39, вот какие выводы мне удалось сделать:

Для числа 25: 1, 7, 18, 24 – лжесвидетели простоты. (Не учитываем 1 и 24)

φ (25)=20. 20/4 = 5 - максимум лжесвидетелей. 5/22 = 0.227 – вероятность случайно выбрать лжесвидетеля.

Но программно я выяснил, что их на самом деле 2 -> 2/22 = 0.09, что подтверждается результатами тестов.

Для числа 39: 1, 38. (Не учитываем 1 и 38)

φ (39) = 24. 24/4 = 6 - максимум лжесвидетелей. 6/36 = 0,166 - теоретическая максимальная вероятность ошибки при одном тесте.

Но программно выяснили, что их на самом деле 0 -> 0/36 = 0, что опять же подтверждается результатами тестов.

Возможно, у всех чисел из моего варианта существует только два лжесвидетеля – 1, n-1. Тогда это объясняет 0% ошибки. В любом случае, я не могу это утверждать, и не смог найти как это проверить, поэтому тест Рабина– Миллера я все равно запускаю с 100 случайными числами, однако, мне кажется, можно существенно ускорить работу теста.

# Листинг разработанной программы

import math

import random

import time

x1 = 19540818305052663929

x2 = 4182626256901165713180947335409849194453

x3 = 565674746577950549441720540453387249813

x4 = 40319329928699692919180302866991670347533839708463260094733186934134923674660447

kar1 = 32809426840359564991177172754241

kar2 = 2810864562635368426005268142616001

def jacobi(a, num):

if a == 0:

return 0

if a == 1:

return 1

if math.gcd(a, num) != 1:

return 0

else:

if (a % 2) == 0:

return jacobi(a//2, num)\*(pow(-1, (num\*num-1)//8))

else:

return jacobi(num % a, a)\*(pow(-1, ((a-1)\*(num-1))//4))

def is\_simple1(num, repeat):

base\_count = 0

rnd2 = 0

for i in range(repeat):

rnd = random.randint(2, num - 2)

rnd2 = rnd

if math.gcd(rnd, num) != 1:

print("Not simple, gcd!=1 ", "Base: ", rnd)

return False

if pow(rnd, (num - 1), num) != 1:

print("Not simple, pow!=1", "Base: ", rnd)

return False

if base\_count < 5:

print("Probably simple with base: ", rnd)

base\_count += 1

print("Probably simple with error probability: ", pow((1 / 2), repeat), "Base: ", rnd2)

return True

def is\_simple2(num, repeat):

base\_count = 0

rnd = 0

for i in range(repeat):

rnd = random.randint(2, num - 2)

if math.gcd(num, rnd) != 1:

print("Not a simple, gcd!=1", "Base: ", rnd)

return False

x = pow(rnd, (num-1) // 2, num)

y = jacobi(rnd, num)

if y == -1:

y = num - 1

if x != y:

print("Not a simple, a^(num-1)/2!=jacoby(a/num) (mod num)", "Base: ", rnd)

return False

if base\_count < 5:

print("Probably simple with base: ", rnd)

base\_count += 1

print("Probably simple with error probability: ", pow((1 / 2), repeat), "Base: ", rnd)

return True

def is\_simple3(num, repeat):

base\_count = 0

if num == 0 or num == 1 or num == 4 or num == 6 or num == 8 or num == 9:

return False

if num == 2 or num == 3 or num == 5 or num == 7:

return True

s = 0

d = num - 1

while d % 2 == 0:

d >>= 1

s += 1

assert (2 \*\* s \* d == num - 1)

def trial\_composite(a):

if pow(a, d, num) == 1 or pow(a, d, num) == num - 1:

return False

for j in range(s):

if pow(a, 2 \*\* j \* d, num) == num - 1:

return False

print("Not simple, no condition is met", "Base: ", a)

return True

rnd = 0

for i in range(repeat):

rnd = random.randrange(2, num-2)

if trial\_composite(rnd):

return False # непростое

if base\_count < 5:

print("Probably simple with base: ", rnd)

base\_count += 1

print("Is simple with error probability: ", pow(1/4, repeat), "Base: ", rnd)

return True # вероятно простое

start\_time = time.perf\_counter()

for i in range(1000):

is\_simple1(x1, 1): #Нет интерфейса, меняем на нужную функцию и число в коде

print("working time =", time.perf\_counter() - start\_time, "\n")