Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

Звіт

про виконання лабораторної роботи №2
з дисципліни “Безпека інформаційних систем”

Виконав:
Студент 4 курсу
Групи КА-75
Степанюк Владислав
Варіант №16

Перевірив:
Мухін В. Є.

Київ – 2020

Завдання

Розробка програми генератора великих простих чисел (ВПЧ) для шифрування і розрахунку ключів за схемою RSA з декількома десятками десяткових розрядів. Для генерації таких простих чисел можна використовувати формули відповідно до тесту Міллера-Рабіна чи іншими методами, але для перевірки властивостей сформованих кандидатів в прості числа необхідно використовувати малу теорему Ферма.

Виконання

Мала теорема Ферма допускає кілька еквівалентних формулювань.
Нехай p — просте, a — ціле, що не ділиться на p. Тоді:

$$a^{p-1} \equiv 1 \pmod{p}.$$

Еквівалентним є наступне твердження: Нехай p — просте, a — довільне ціле число. Тоді:

$$a^p - a \equiv 0 \pmod{p}.$$

Тест простоти Міллера-Рабіна

Тест Міллера - Рабина - імовірнісний поліноміальний тест простоти. Тест Міллера - Рабина дозволяє ефективно визначати, чи є дане число складовим. Однак, з його допомогою можна строго довести простоту числа. Проте тест Міллера - Рабина часто використовується в криптографії для отримання великих випадкових простих чисел.

Алгоритм був розроблений Гарі Міллером в 1976 році і модифікований Майклом Рабіном в 1980 році.

Нехай **m** - непарне число більше 1. Число **m - 1** однозначно представляється у вигляді $m-1 = 2^s * t$, де **t** - непарний.

Ціле число **a**, $1 < a < m$, називається свідком простоти числа **m**, якщо виконується одна з умов:

1) $a^t \equiv 1 (\operatorname{mod} m)$

2) або існує ціле число **k**, $0 \le k < s$, таке, що $a^{2kt} \equiv -1 (\operatorname{mod} m)$.

Теорема Рабина стверджує, що складене непарне число **m** має не більше $\dfrac{\varphi(m)}{4}$ різних свідків простоти, де $\varphi(m)$ - функція Ейлера.

Алгоритм Міллера - Рабина параметрізуется кількістю раундів **r**.
Рекомендується брати **r** порядку величини $\log_2(m)$, де **m** - перевіряється число.

Для даного **m** знаходяться такі ціле число **s** і ціле непарне число **t**, що $m-1=2^s*t$. Вибирається випадкове число **a**, $1<a<m$. Якщо **a** не є свідком простоти числа, то видається відповідь «m складене», і алгоритм завершується. Інакше, вибирається нове випадкове число **a** і процедура перевірки повторюється. Після знаходження свідків простоти, видається відповідь «m, ймовірно, просте», і алгоритм завершується.

Як і для тесту Ферма, все числа n>1, які не проходять цей тест - складові, а числа, які проходять, можуть бути простими. І, що важливо, для цього тесту немає аналогів чисел Кармайкла.
У 1980 році було доведено, що ймовірність помилки тесту Рабіна-Міллера не перевищує 1/4. Таким чином, застосовуючи тест Рабіна-Міллера раз для різних підстав, ми отримуємо ймовірність помилки 2^{-2t} .

МІЛЛЕР-РАБІН(n, k)

якщо n парне **тоді**
    повернути ХИБА
$m \leftarrow (n-1) \operatorname{div} 2; t \leftarrow 1$
поки m парне
    $m \leftarrow m \operatorname{div} 2; t \leftarrow t+1$
для $i \leftarrow 1$ **до** k
    $a \leftarrow Random() \mod n$
    $u \leftarrow a^m \mod n$
    якщо $u \neq 1$ **тоді**
        $j \leftarrow 1$
        поки $u \neq -1$ і $j < t$
            $u \leftarrow u^2 \mod n; j \leftarrow j+1$
        якщо $u \neq -1$ **тоді**
            повернути ХИБА
повернути ІСТИНА

Імовірність помилки $\leq 1/4^k$.

Лістинг коду

```python
#!/usr/bin/env python3

import random
from random import randrange
import math
def is_Prime_Miller_Raben(n, k):
    if n != int(n):
        return False
    n = int(n)
    if n == 0 or n == 1 or n == 4 or n == 6 or n == 8 or n == 9:
        return False
    if n == 2 or n == 3 or n == 5 or n == 7:
        return True
    s = 0
    d = n-1
    while d % 2 == 0:
        d >>= 1
        s += 1
    assert(2**s * d == n-1)
    def trial_composite(a):
        if pow(a, d, n) == 1:
            return False
        for i in range(s):
            if pow(a, 2**i * d, n) == n-1:
                return False
        return True
    for i in range(k):
        a = random.randrange(2, n)
        if trial_composite(a):
            return False
    return True
def is_Prime_Ferma(p, k):
    for i in range(k):
        a = randrange(2, p-1)
        if a**(p-1) % p != 1:
            return False
    return True
def get_Prime(sign):
    if(sign < 1):
        return 1
    while True:
        n = randrange(10**(sign - 1), 10**(sign))
        print(n)
        k = math.floor(math.log(n, 2) + 1)
        if(is_Prime_Miller_Raben(n, k)):
            return n
def main():
    n = get_Prime(2)
    print('Prime number: ', n)
    print('Miller_Raben: ', is_Prime_Miller_Raben(
        10000000000000000000000000000000000000000000000000000000000000000000000000000000, 1))
    print('Ferma: ', is_Prime_Ferma(n, 10))
    return
main()
```

Результати

```
Prime number:  1
Miller_Raben:  False
Ferma:  False
Prime number:  3469525981
Miller_Raben:  True
Ferma:  True
Prime number:  62594151236383601999
Miller_Raben:  True
Ferma:  True
Prime number:  944769671071808952663061874111
Miller_Raben:  True
Ferma:  True
Prime number:  4655145744697772458318126592251498400633
Miller_Raben:  True
Ferma:  True
```

Висновки

Під час виконання даної лабораторної роботи було реалізовано функції генерації простих чисел з використанням перевірки на простоту тестом Міллера-Рабіна, а також перевірки на простоту за допомогою малої теореми Ферма.

У ході виконання було з’ясовано, шщо тест Міллера-Рабіна підходить як для генерації, так і для перевірки дуже великих чисел на простоту, проте мала теорема Ферма для перевірки чисел зі знаком більше 5 не підходить, через обмеження в обчислювальній здатності комп’ютера.

Література

https://foxford.ru/wiki/informatika/test-prostoty-millera-rabina
https://habr.com/ru/company/otus/blog/486116/