Лабораторная работа № 4

по информационной безопасности

студента группы ИТ-32

Манукова Давида Альбертовича

Выполнение: \_\_\_\_\_\_\_\_\_\_ Защита: \_\_\_\_\_\_\_\_\_\_

**Генерация больших простых чисел и алгоритмы тестирования на простоту**

Цель работы: освоить методы генерации больших простых чисел, изучить и реализовать вероятностные и детерминированные алгоритмы проверки чисел на простоту.

Содержание работы

Реализовать приложение, позволяющее выполнять следующие действия:

1. Реализовать подпрограмму для проверки чисел на простоту, используя изученные вероятностные методы.

Программа должна отражать затраченное время на проверку чисел на простоты.

В тесте Ферма предусмотреть вывод сообщения о введении числа Кармайкла, если такое число подается на проверку.

В тесте Рабина-Миллера пользователь должен иметь возможность самостоятельно задавать количество проверок.

2. Реализовать подпрограмму для проверки чисел на простоту, используя предложенный детерминированный алгоритм( алгоритм теории делимости).

Программа должна отражать затраченное время на проверку чисел на простоты.

3. С помощью алгоритма генерации простого числа получить большое простое число (в данной лабораторной работе под большими числами будем понимать числа, превышающие 264).

Пользователь вводит количество проверок в тесте на простоту и длину числа в битах.

Ход работы

1. Разработал программу, реализующую процесс шифрования/дешифрования.

using System;

using System.Numerics;

using System.Security.Cryptography;

namespace IS\_Lab\_4

{

class Program

{

static long[] c;

public static void coef(int n)

{

c = new long[n+1];

c[0] = 1;

for (int i = 0; i < n; c[0] = -c[0], i++)

{

c[1 + i] = 1;

for (int j = i; j > 0; j--)

c[j] = c[j - 1] - c[j];

}

}

public static bool isPrime(int n)

{

coef(n);

c[0]++;

c[n]--;

int i = n;

while ((i--) > 0 && c[i] % n == 0) ;

return i < 0;

}

public static bool MillerRabinTest(BigInteger n, int k)

{

if (n == 2 || n == 3)

return true;

if (n < 2 || n % 2 == 0)

return false;

BigInteger t = n - 1;

int s = 0;

while (t % 2 == 0)

{

t /= 2;

s += 1;

}

for (int i = 0; i < k; i++)

{

RNGCryptoServiceProvider rng = new RNGCryptoServiceProvider();

byte[] \_a = new byte[n.ToByteArray().LongLength];

BigInteger a;

do

{

rng.GetBytes(\_a);

a = new BigInteger(\_a);

}

while (a < 2 || a >= n - 2);

BigInteger x = BigInteger.ModPow(a, t, n);

if (x == 1 || x == n - 1)

continue;

for (int r = 1; r < s; r++)

{

x = BigInteger.ModPow(x, 2, n);

if (x == 1)

return false;

if (x == n - 1)

break;

}

if (x != n - 1)

return false;

}

return true;

}

public static int NOD(int a, int b)

{

while (a != 0 && b != 0)

{

if (a > b)

{

a = a % b;

}

else

b = b % a;

}

return a + b;

}

public static int PHI(int n)

{

int g = 0;

for (int i = 1; i < n; i++)

{

if (NOD(i, n) == 1)

g++;

}

return g;

}

public static double MHP(int a, int k, int b)

{

return Math.Pow(a, k % PHI(b)) % b;

}

public static void Main()

{

Console.Write("Введите n: ");

int n = int.Parse(Console.ReadLine());

Console.WriteLine("\nТест Ферма: ");

bool flag = MillerRabinTest(n, 10);

if (flag)

{

Console.WriteLine("Число {0} простое", n);

}

else Console.WriteLine("Число {0} составное", n);

Console.WriteLine("\nТест Рабина - Миллера: ");

bool f = true;

for (int a = 1; a < n - 1; a++)

{

if (MHP(a, n - 1, n) != 1)

{

Console.WriteLine(String.Format("Число {0} составное", n));

f = false;

break;

}

}

if (f)

{

Console.WriteLine(String.Format("Число {0} простое", n));

}

Console.WriteLine("\nАлгоритм AKS");

if (isPrime(n))

{

Console.WriteLine("Число {0} простое", n);

}

else Console.WriteLine("Число {0} составное", n);

Console.ReadLine();

}

}

}

1. Результаты работы (см Рисунок 1, Рисунок 2)

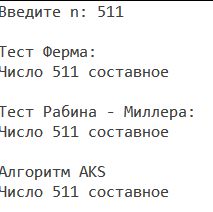


Рисунок 1 Результат работы

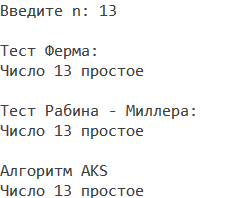


Рисунок 2 Результат работы

Вывод: освоил методы генерации больших простых чисел, изучил и реализовал вероятностные и детерминированные алгоритмы проверки чисел на простоту.