**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Инженерная школа информационных технологий и робототехники

Направление 09.03.02 «Информационные системы и технологии»

Отделение информационных технологий

Отчёт по лабораторной работе № 4

по дисциплине **«Информационная безопасность и защита информации»**

**РАЗРАБОТКА ПРОГРАММНОЙ РЕАЛИЗАЦИИ АССИМЕТРИЧНОГО АЛГОРИТМА ШИФРОВАНИЯ ДАННЫХ RSA**

Выполнил:

Студент группы 8И7Б Балашов И. А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет принял:

Ассистент (ОИТ, ИШИТР) Григорьев Д. С. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Томск 2021

# Цель работы

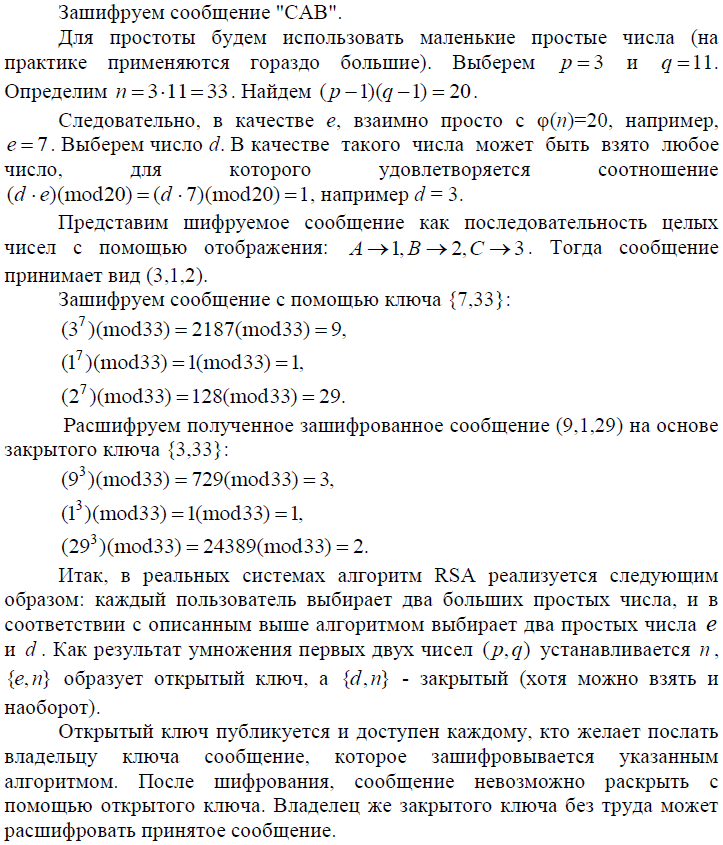
Целью лабораторной работы является создание программной реализации ассиметричного алгоритма шифрования данных RSA.

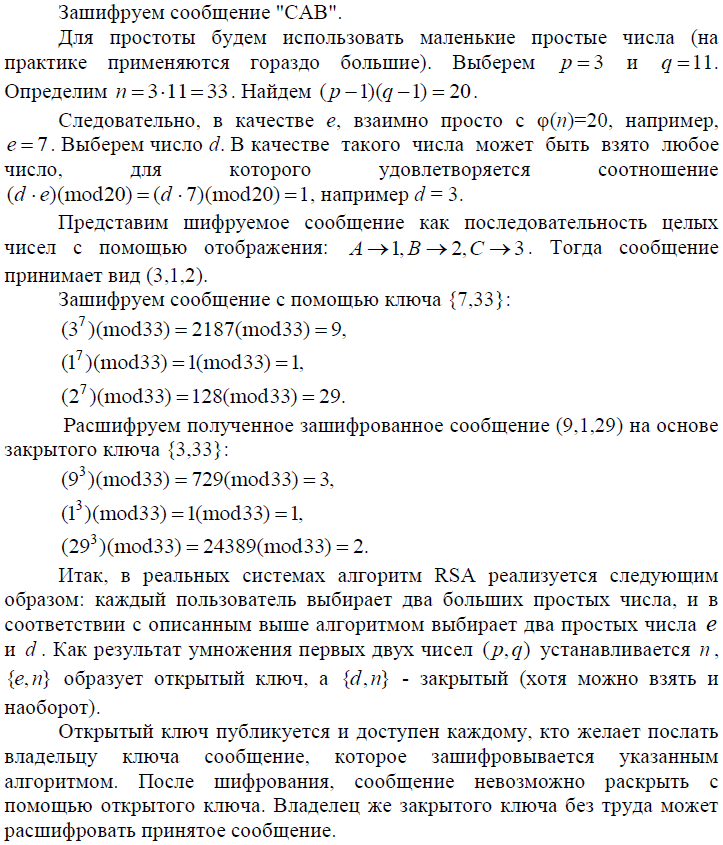
# Задание

1. Создать программную реализацию алгоритма RSA.
2. Зашифровать введенное с клавиатуры сообщение, используя открытый ключ.
3. Расшифровать сообщение, используя закрытый ключ.

# Метод решения задачи

Пример:





# Листинг программы

using System;

using System.Collections.Generic;

using System.Numerics;

namespace lab4RSA

{

class Program

{

static void Main(string[] args)

{

char[] alphavit = new char[]

{

'а', 'б', 'в', 'г', 'д', 'е', 'ё', 'ж', 'з',

'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р',

'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ',

'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'А', 'Б', 'В',

'Г', 'Д', 'Е', 'Ё', 'Ж', 'З', 'И', 'Й', 'К',

'Л', 'М', 'Н', 'О', 'П', 'Р', 'С', 'Т', 'У',

'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ', 'Ъ', 'Ы', 'Ь',

'Э', 'Ю', 'Я', ' ', '.', ':', '!', '?', ','

};

int p = 0, q = 0;

while (p\*q < (alphavit.Length+1))

{

Console.Write("Введите p: ");

while (!int.TryParse(Console.ReadLine(), out p))

{

Console.WriteLine("p должно быть числом, введите р заново");

}

Console.Write("Введите q: ");

while (!int.TryParse(Console.ReadLine(), out q))

{

Console.WriteLine("q должно быть числом, введите q заново");

}

if(p \* q < (alphavit.Length + 1))

{

Console.WriteLine("Произведение p и q должно быть больше мощности алфавита введите p и q заново");

}

}

Console.WriteLine("Зашифровать строку введите 1");

Console.WriteLine("Расшифровать строку введите 2");

Console.WriteLine("Ввести новые р и q введите 3");

Console.WriteLine("Закончить введите 0");

string run = Console.ReadLine();

while (run!="0")

{

if (run == "3")

{

p = 0;

q = 0;

while (p \* q < (alphavit.Length + 1))

{

Console.Write("Введите p: ");

while (!int.TryParse(Console.ReadLine(), out p))

{

Console.WriteLine("p должно быть числом, введите р заново");

}

Console.Write("Введите q: ");

while (!int.TryParse(Console.ReadLine(), out q))

{

Console.WriteLine("q должно быть числом, введите q заново");

}

if (p \* q < (alphavit.Length + 1))

{

Console.WriteLine("Произведение p и q должно быть больше мощности алфавита введите p и q заново");

}

}

}

int n = 0;

int f = 0;

int e = 0;

int d\_ = 0;

//зашифровка строки

if (run == "1")

{

if (SimpleNumber(p) && SimpleNumber(q)) //если p и q простые

{

Console.Write("Введите строку: ");

string s = Console.ReadLine();

n = p \* q;

f = (p - 1) \* (q - 1);

e = Calculate\_e(f);

d\_ = Calculate\_d(e, f);

Console.WriteLine("\nn = " + n);

Console.WriteLine("f = " + f);

Console.WriteLine("e = " + e);

Console.WriteLine("d = " + d\_);

Console.WriteLine("\nОткрытый ключ: " + "{" + e + ", " + n + "}");

Console.WriteLine("Закрытый ключ: " + "{" + d\_ + ", " + n + "}");

///// ШИФРОВАНИЕ /////

bool flag = true;

List<object> obj = Encode(s, e, n, flag); //Вызов метода Encode

if (!(bool)obj[1]) //проверка на недопустимые символы

{

Console.WriteLine("В ведённой строке есть недопустимые символы");

continue;

}

List<string> result = (List<string>)obj[0];

List<string> input = new List<string>();

Console.WriteLine("\nЗашифрованный текст: \n");

foreach (string item in result)

{

Console.Write(item + " ");

input.Add(item);

}

Console.WriteLine();

Console.WriteLine();

}

else

{

Console.WriteLine("p и q должны быть простыми числами");

}

}

//расшифровка строки

if (run == "2")

{

if (SimpleNumber(p) && SimpleNumber(q)) //если p и q простые

{

Console.Write("Введите строку: ");

string[] s = Console.ReadLine().Split(' ');

n = p \* q;

f = (p - 1) \* (q - 1);

e = Calculate\_e(f);

d\_ = Calculate\_d(e, f);

Console.WriteLine("\nn = " + n);

Console.WriteLine("f = " + f);

Console.WriteLine("e = " + e);

Console.WriteLine("d = " + d\_);

Console.WriteLine("\nОткрытый ключ: " + "{" + e + ", " + n + "}");

Console.WriteLine("Закрытый ключ: " + "{" + d\_ + ", " + n + "}");

List<string> input = new List<string>();

foreach (string item in s)

{

input.Add(item);

}

///// ДЕШИФРОВАНИЕ /////

string result\_2 = Decode(input, d\_, n); //Вызов метода Decode

Console.WriteLine("\nРасшифрованный текст: " + result\_2);

}

else

{

Console.WriteLine("p и q должны быть простыми числами");

}

}

Console.WriteLine("Зашифровать строку введите 1");

Console.WriteLine("Расшифровать строку введите 2");

Console.WriteLine("Ввести новые р и q введите 3");

Console.WriteLine("Закончить введите 0");

run = Console.ReadLine();

}

///// МЕТОДЫ /////

static bool SimpleNumber(int n) //Проверка, является ли число простым

{

if (n < 2)

return false;

if (n == 2)

return true;

for (int i = 2; i < n; i++)

if (n % i == 0)

return false;

return true;

}

static int Calculate\_e(int f) //Вычисление e, оно должно быть взаимно простым с n

{

int e = f - 1; //по условию, e должно быть меньше f

for (int i = 2; i <= f; i++)

if ((f % i == 0) && (e % i == 0)) //если f и e имеют общие делители, то e уменьшается,

//иначе получаем e

{

e--;

}

return e;

}

static int Calculate\_d(int e, int f) //Вычисление d по формуле

{

int d = e+1;

while (true)

{

if ((d \* e) % f == 1) //d должно быть взаимно простым с f, если так, то берем d

break;

else

d++;

}

return d;

}

List<object> Encode(string s, int d, int n, bool flag) //Шифрование

{

List<string> result = new List<string>();

List<object> ob = new List<object>();

BigInteger bi;

for (int i = 0; i < s.Length; i++)

{

int index = Array.IndexOf(alphavit, s[i]); //получаем номер буквы в алфавите

if (index == -1) // проверка на допустивые символы

{

flag = false;

break;

}

bi = new BigInteger(index);

bi = BigInteger.Pow(bi, (int)d); //возводим в степень d\_ номер буквы

BigInteger n\_ = new BigInteger((int)n);

bi = bi % n\_; //получаем шифр буквы

result.Add(bi.ToString());

}

if (!flag)

{

ob.Add(result);

ob.Add(false);

}

else

{

ob.Add(result);

ob.Add(true);

}

return ob;

}

string Decode(List<string> input, long e, long n)

{

string result = "";

BigInteger bi;

foreach (string item in input)

{

bi = new BigInteger(Convert.ToDouble(item));

bi = BigInteger.Pow(bi, (int)e); //возводим в степень d шифр буквы

BigInteger n\_ = new BigInteger((int)n);

bi = bi % n\_; //получаем номер буквы в алфавите

int index = Convert.ToInt32(bi.ToString());

result += alphavit[index].ToString(); //добавляем полученную букву в строку

}

return result;

}

}

}

}

# Результаты работы программы

1. На вход программы подаются два простых числа p и q.
2. Выбираем зашифровать сообщение или расшифровать сообщение.

В обоих случаях:

* 1. Определяется f(n) = (p-1) \* (q-1).
  2. Происходит выбор числа e, взаимно простого с f(n).
  3. Вычисляется .
  4. Открытым ключом является {e, n}, закрытым {d, n}.

1. Далее происходит шифрование или дешифрование данных.

Весь процесс отображается в консоли (рисунок 1).

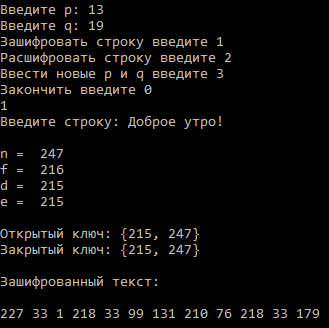


Рисунок 1 – Результат зашифровки

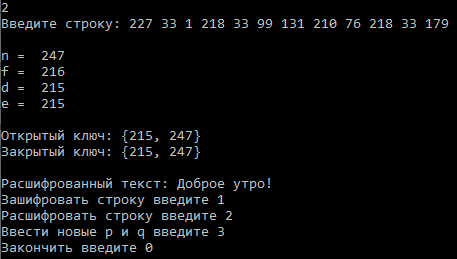


Рисунок 2 – Результат расшифровки

# Ссылка на GitHub

https://github.com/MementoMorii/InfBezLab/tree/main/Lab4

# Вывод

В ходе выполнения лабораторной работы были изучены теоретические сведения о ассиметричном алгоритме шифрования данных, и была реализована программа, шифрующая текстовые сообщения с помощью данного алгоритма.