МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Лабораторная работа № 9**

**ИССЛЕДОВАНИЕ АСИММЕТРИЧНЫХ ШИФРОВ**

Разработал: Бай И.О.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

**Цель:** изучение и приобретение практических навыков разработки и использования приложений для реализации асимметричных шифров.

**Задачи:**

1. Закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости асимметричных шифров.
2. Разработать приложение для реализации указанных преподавателем методов генерации ключевой информации и ее использования для асимметричного зашифрования/расшифрования.
3. Выполнить анализ криптостойкости асимметричных шифров.
4. Оценить скорость зашифрования/расшифрования реализованных шифров.
5. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

**Практическая часть**

1. **Описание приложения**

Приложение написано на языке программирования C# и позволяет выполнить следующие задачи:

* генерация сверхвозрастающей последовательности (тайного ключа);
* вычисление нормальной последовательности (открытого ключа);
* зашифрование сообщения, состоящего из ФИО;
* расшифрование сообщения;
* оценка времени выполнения операций зашифрования и расшифрования.

В основе вычислений – кодировочные таблицы Base64 и ASCII.

1. **Методика выполнения поставленных задач**

Для реализации генерации сверхвозрастающей и нормальной последовательности, была разработана функция, представленная на рисунке 2.1. Параметры *n* и *a* вычисляются с помощью ф-ций *GetN* и *GetA*, представленных на рисунке 2.2.

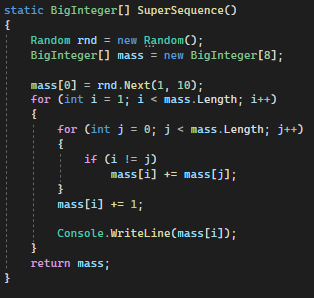


Рис. 2.1 – Генерация сверхвозрастающей и нормальной последовательностей

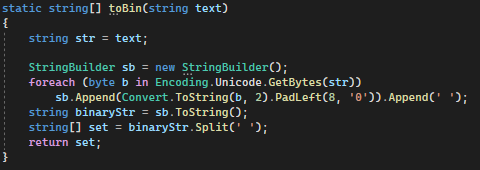
****

Рис. 2.2 – Генерация параметров *n* и *a*

Реализация ф-ций зашифрования/расшифрования представлена в листингах 2.1 и 2.2 соответственно.

BigInteger[] mass = SuperSequence();

superSequeceBox.Text = "";

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

{

superSequeceBox.Text += mass[i] + "\n";

}

BigInteger W = 0; //Сумма элементов массива

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

{

W += mass[i];

}

//Для получения открытого ключa все значения закрытого ключа умножаются на некоторое число a по модулю n

string n = "265252859812191058636308479999999";

//string n = "40";

BigInteger N = BigInteger.Parse(n);

numN.Text = "n = " + N.ToString();

string a = "75838465738475849374657348573930";

//string a = "13";

BigInteger A = BigInteger.Parse(a);

numA.Text = "a = " + A.ToString();

BigInteger[] S = new BigInteger[8];

publicKey.Text = "";

for (int i = 0; i < 8; i++)

{

S[i] = (mass[i] \* A) % N;

publicKey.Text += S[i] + "\n";

}

Листинг 2.1 – функция зашифрования

private void Decrypt(object sender, RoutedEventArgs e)

{

stopwatch.Start();

int a\_ = 2;

while (a \* a\_ % n != 1)

{

a\_++;

}

decText = new int[encText.Length];

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

{

decText[i] = (encText[i] \* a\_) % n;

}

int[,] binText = new int[decText.Length, count];

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

{

for (int j = sequence.Length - 1; j >= 0; j--)

{

if (sequence[j] <= decText[i])

{

binText[i, j] = 1;

decText[i] = decText[i] - sequence[j];

}

else

{

binText[i, j] = 0;

}

}

}

int[] origText = new int[decText.Length];

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

{

for (int j = 0; j < count; j++)

{

if (binText[i, j] == 1)

{

origText[i] += (int)Math.Pow(2, j);

}

}

}

string result = "";

if (count == 6)

{

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

{

result += base64[origText[i]];

}

}

else if (count == 8)

{

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

{

result += (char)origText[i];

}

}

stopwatch.Stop();

DecrTime.Content = $"{(double)stopwatch.ElapsedTicks / freq} sec \n";

stopwatch.Reset();

TextOrig.Text = result;

}

Листинг 2.2 – функция расшифрования

1. **Реализация приложения**

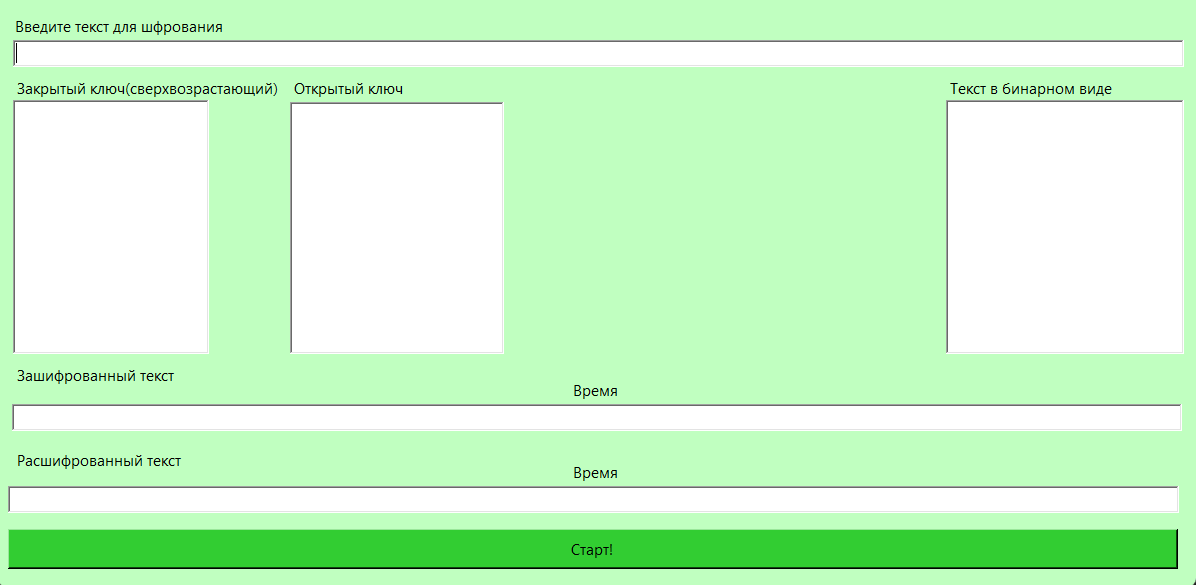
****

Рис. 3.1 – Внешний вид программы

После генерации последовательностей необходимо ввести текст в поле *Origin text* и нажать кнопку *Encrypt.* Результатом этой операции будет появление шифртекста в поле *Encrypt text.* Чтобы расшифровать зашифрованное сообщение необходимо нажать кнопку *Decrypt.*

Для очистки всех полей формы используется кнопка *Clear*.

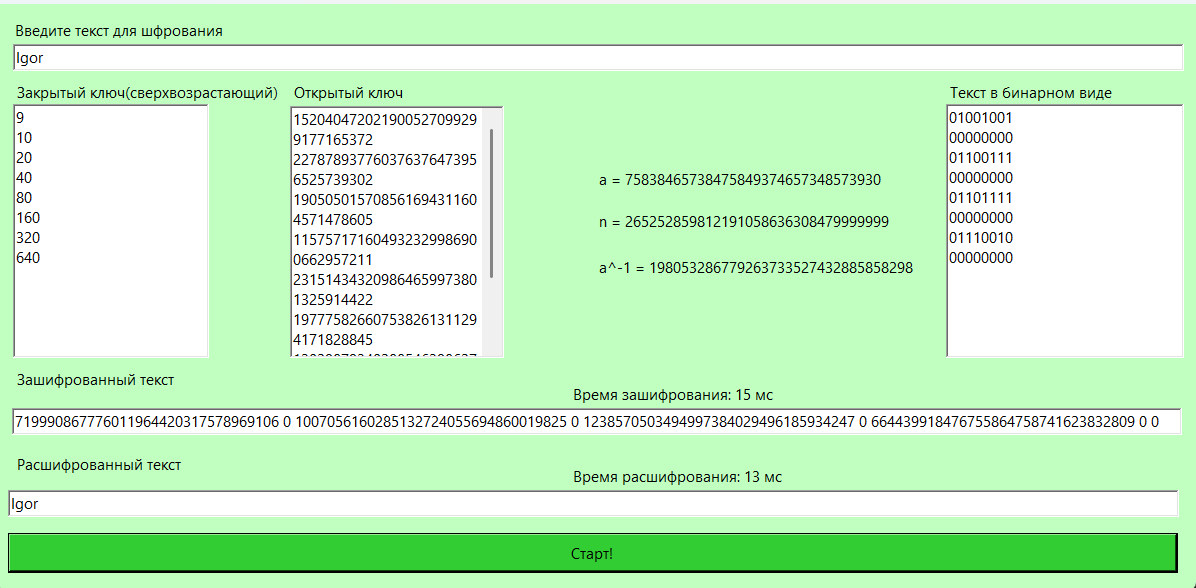


Рис. 3.2 – Результат выполнения шифрования/расшифрования

Рис. 3.3 – Зависимость времени шифрования/расшифрования в зависимости от кол-ва элементов в последовательности

**Вывод:** в основу асимметричной криптографии положена идея использовать ключи парами: один – для зашифрования (открытый, или публичный, ключ), другой – для расшифрования (тайный ключ). Отметим, что указанная пара ключей принадлежит получателю зашифрованного сообщения. Все алгоритмы шифрования с открытым ключом основаны на использовании односторонних функций, к числу которых, как известно, относится вычисление дискретного логарифма.