Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Исследование асимметричных шифров**

Студент: Круглик А.В.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

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.

1. **Методика выполнения поставленных задач**

Для реализации генерации сверхвозрастающей последовательности была разработана функция, представленная на рисунке 2.1.

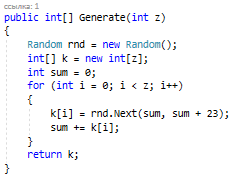


Рис. 2.1 – Реализация генерации тайного ключа

Данная функция вычисляет последовательно случайные числа, с условием, что каждый последующий элемент будет больше суммы всех предыдущих.

Для вычисления нормальной последовательности была разработана функция, представленная на рисунке 2.2.

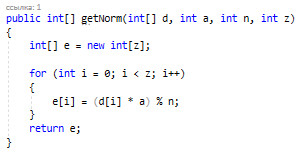


Рис. 2.2 – Реализация вычисления открытого ключа

Данная функция вычисляет элементы последовательности *e*={*ei*} по формуле: *ei=di×a (mod n)*, где *di* – члены тайного ключа, n больше суммы всех чисел последовательности, причем НОД (*a,n*) = 1.

Для зашифрования сообщения *M* каждый его символ *m*i был представлен в двоичной форме. После этого была реализована последовательная проверка каждого символа *mi* на равенство единице или нулю. В первом случае соответствующий элемент открытого ключа *ei* суммировался к переменной total, вычисляемой для каждого *m*i. Пример зашифрования одного символа при открытом ключе *e*: {62, 93, 186, 403, 417, 352, 315, 210} представлен в таблице 2.1.

Таблица 2.1 – Зашифрование символа

|  |  |  |
| --- | --- | --- |
| Бинарный код символа mj сообщения | Укладка ранца | Вес ранца |
| 11010000 | 62+93+403 | 558 |

Для расшифрования сообщения был использован сгенерированный тайный ключ *d*. Алгоритм следующий:

* определить *a*-1, такое что *a×a-1 (mod n) = 1*;
* каждый символ шифротекста *ci* преобразовать: *Si = ci×a-1 mod n*;
* используя вычисленное *Si* с помощью тайного ключа *d* и известного алгоритма упаковки получить расшифрованные символы *mi*.

Реализация данной функции представлена на рисунке 2.3.

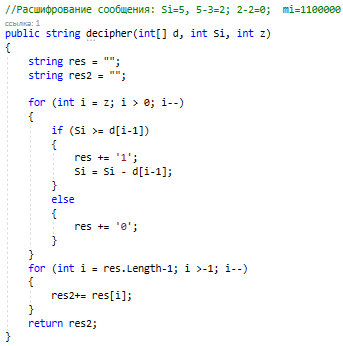


Рис. 2.3 – Реализация функции расшифрования

Результат выполнения данного консольного приложения представлен на рисунке 2.4.

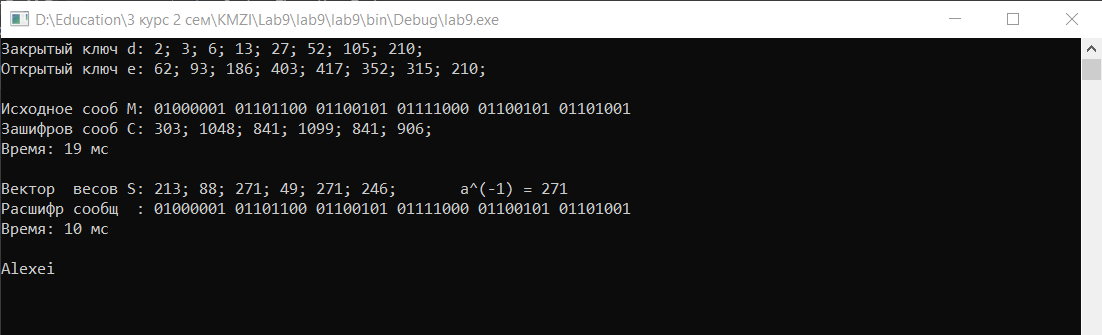


Рис. 2.4 – Результат работы приложения

Также, нами была оценена скорость выполнения зашифрования/расшифрования сообщения с помощью встроенной возможности C# - DateTime.Now.Ticks. Вычисленное время составило 90 мс и 10 мс соответственно, что является неплохим результатом. Это представлено на рисунке 2.5 и 2.6.

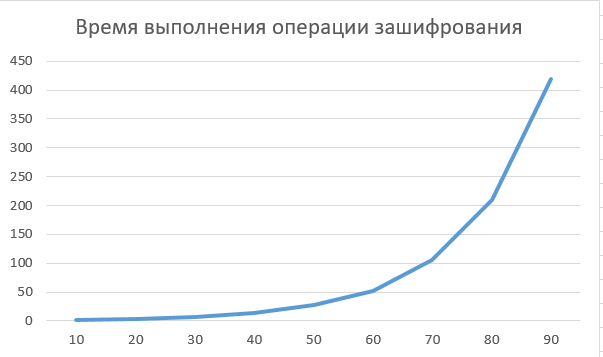


Рис. 2.5 – Результат времени выполнения операции зашифрования



Рис. 2.6 – Результат времени выполнения операции расшифрования

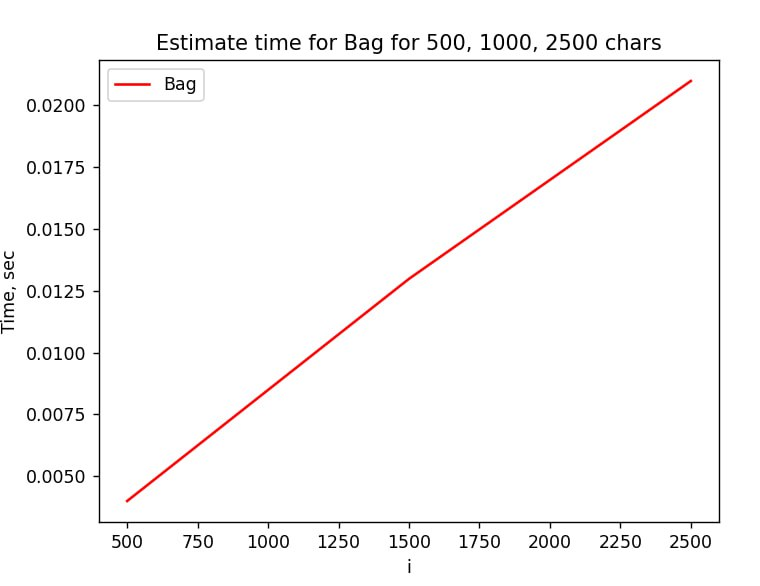


Рис. 2.7 – время выполнения операций зашифрования/расшифрования при увеличении числа членов ключевой последовательности

**Вывод**

В ходе лабораторной работы были приобретены практические навыки разработки и использования приложений для реализации асимметричных шифров. Было разработано приложение для реализации методов генерации ключевой информации и ее использования. Также была оценена скорость зашифрования/расшифрования.