Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Исследование потоковых шифров**

Студент: Димитриади А.В.

ФИТ 3 курс 5 группа

Преподаватель:

Савельева Маргарита Геннадьевна

1. **Описание приложения и заданий**

Приложение должно выполнять следующие функции:

* реализовывать генерацию ПСП на основе линейного конгруэнтного генератора;
* реализовывать алгоритм RC4;
* выполнять оценку скорости выполнения операций генерации ПСП;

1. **Выполнение поставленных задач**

Часто используемый алгоритм генерирования ПСП реализуется на основе так называемого линейного конгруэнтного генератора, описываемого следующим рекуррентным соотношением: , где: и – соответственно *t*-й (предыдущий) и (*t*+1)-й (текущий, вычисляемый) члены числовой последовательности; *а*, *с* и *n* – константы. Период такого генератора не превышает *n*.

Если параметры *a*, *b* и *c* выбраны правильно, то генератор будет порождать случайные числа с максимальным периодом, равным *c*.

Достоинством линейных конгруэнтных генераторов псевдослучайных чисел является их простота и высокая скорость получения псевдослучайных значений. Линейные конгруэнтные генераторы находят применение при решении задач моделирования и математической статистики, однако в криптографических целях их нельзя рекомендовать к использованию, так как специалисты по криптоанализу научились восстанавливать всю последовательность ПСЧ по нескольким ее значениям.

Программная реализация этого алгоритма представлена на рисунке 2.1.

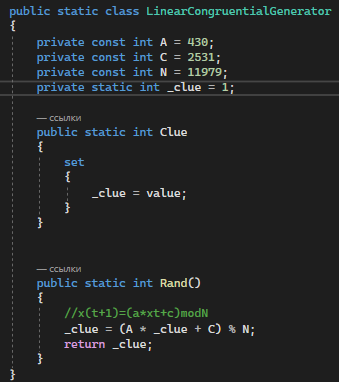


Рисунок 2.1 – Реализация линейного конгруэнтного генератора

RC4 (Rivest Cipher 4) - это алгоритм симметричного шифрования, разработанный Роном Ривестом в 1987 году. Он является потоковым шифром, что означает, что шифрование происходит побитово или побайтово. RC4 широко использовался в различных протоколах безопасности, таких как SSL (Secure Sockets Layer) и WEP (Wired Equivalent Privacy).

Ниже приведен общий алгоритм шифрования RC4:

1. Инициализация:

* Создается ключ шифрования длиной от 1 до 256 байт.
* Создается вектор инициализации (IV), который обычно имеет длину 8 или 16 байт.

1. Инициализация внутреннего состояния:

* Создается массив S длиной 256 байт, содержащий все возможные значения от 0 до 255.
* Производится перестановка элементов массива *S* на основе ключа с использованием алгоритма ключевого расписания.

1. Генерация псевдослучайной последовательности:

* Используя внутреннее состояние и массив *S*, генерируется псевдослучайная последовательность байт, которая является ключевым потоком для шифрования.
* Генерация осуществляется путем перестановки элементов массива *S* и выбора псевдослучайного байта на основе значений из массива *S*.

1. Шифрование/расшифрование:

* Каждый байт открытого текста (или шифртекста) складывается по модулю 2 с соответствующим байтом из псевдослучайной последовательности.
* Полученный результат является зашифрованным (или расшифрованным) байтом.

1. Генерация нового ключевого потока:

* Если необходимо шифровать больше данных, выполняется генерация новой псевдослучайной последовательности путем дополнительных итераций перестановки элементов массива *S* и выбора байтов.

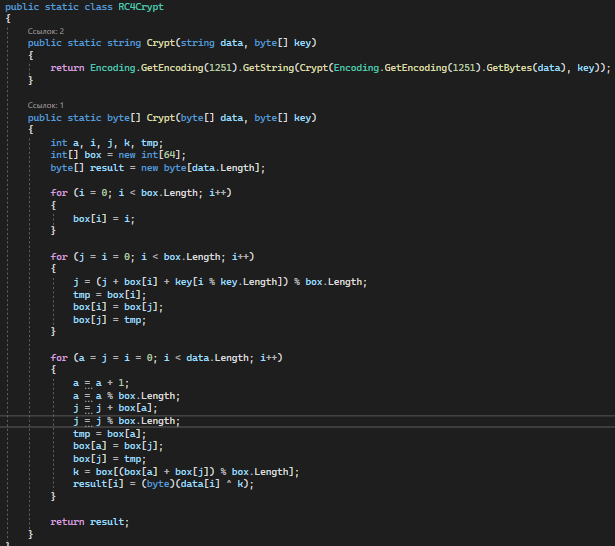


Рисунок 2.2 – Реализация RC4 алгоритма

Далее представлен основной код программы и используемые в ней алгоритмы, рисунок 2.3 и результат работы, рисунок 2.4.

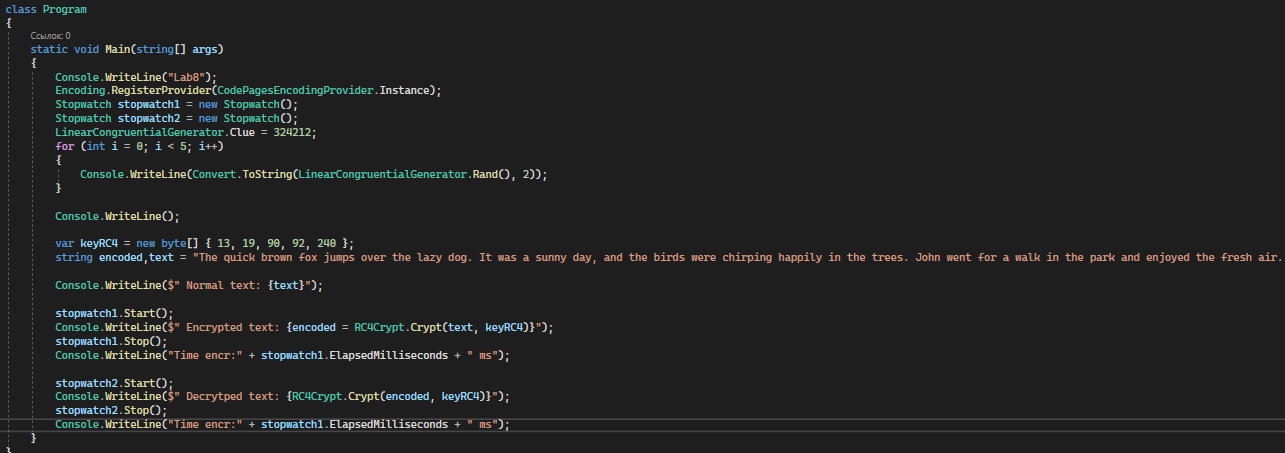


Рисунок 2.3 – Основной код

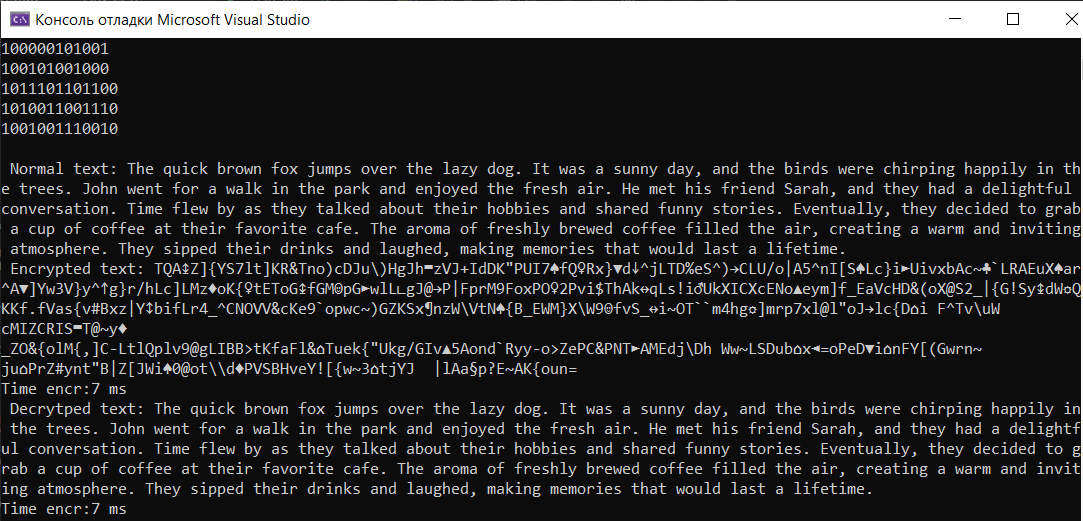


Рисунок 2.4 – Результат работы

Также ниже приведен оценочный график времени выполнения зашифрования и расшифрования потокового алгоритма, рисунок 2.5.

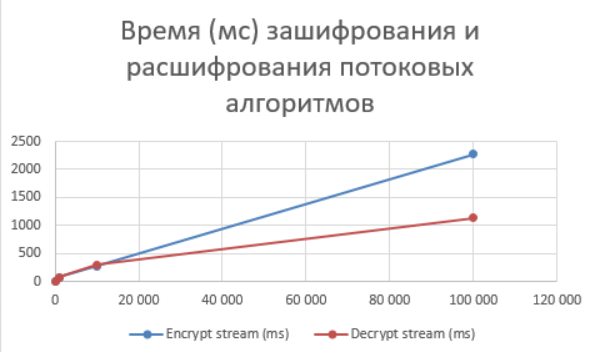


Рисунок 2.5 – Оценочный график

**Вывод**

В данный лабораторной работе были изучены и приобретены практические навыки разработки и использования потоковых шифров. Проанализированы оценочные графики времени зашифрования и расшифрования алгоритмов.

Разработано приложение, на языке программирования C#, для реализации задач, связанных с шифрованием данных.