Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Исследование ассиметричных шифров**

Студент: Гвоздовский К.В.

ФИТ 3 курс 6 группа

Преподаватель:

Нистюк Ольга Александровна

1. **Цель работы**

Изучение и приобретение практических навыков разработки и использования приложений для реализации блочных шифров.

1. **Задание**

Разработать авторские многооконные приложения в соответствии с целью лабораторной работы. При этом можно воспользоваться готовыми библиотеками либо программными кодами, реализующими заданные алгоритмы.

Приложение 1 должно реализовывать генерацию ПСП в соответствии с вариантом из табл. 6.6.

Приложение 2 должно реализовывать алгоритм RC4 в соответствии с вариантом из табл. 6.7, а также дополнительно выполнять оценку скорости выполнения операций генерации ПСП.

1. **Ход работы**

**Теоретические сведения**

*Основные свойства асимметричных криптосистем*

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.

Односторонней функцией называется математическая функция, которую относительно легко вычислить, но трудно найти по значению функции соответствующее значение аргумента, т. е. зная х, легко вычислить *f*(*x*), но по известному *f*(*x*) трудно найти подходящее значение *x*.

Алгоритмы шифрования с открытым ключом можно использовать для решения следующих задач:

* зашифрования/расшифрования передаваемых и хранимых данных в целях их защиты от несанкционированного доступа;
* формирования цифровой подписи под электронными документами;
* распределения секретных ключей, используемых далее при шифровании документов симметричными методами.

В данной работе мы будем работать над аспектами решения первой из указанных задач.

По мнению Диффи и Хеллмана, алгоритм шифрования с открытым ключом должен:

* вычислительно легко создавать пару (открытый ключ *e* – закрытый ключ d);
* вычислительно легко зашифровывать сообщение *Mi* открытым ключом;
* вычислительно легко расшифровывать сообщение *Ci*, используя закрытый ключ;
* обеспечивать непреодолимую вычислительную сложность определения соответствующего закрытого ключа при известном открытом ключе;
* обеспечивать непреодолимую вычислительную сложность восстановления исходного (открытого сообщения *Mi*) зная только открытый ключ и зашифрованное сообщение *Ci*.

*Криптоалгоритм на основе задачи об укладке ранца*

Ранцевый (рюкзачный) вектор **S** = (, ..., ) – это упорядоченный набор из *z*, *z* ≥ 3, различных натуральных чисел . Входом задачи о ранце (рюкзаке) называем пару (***S***, *S*), где *S* – рюкзачный вектор, а *S* – натуральное число.

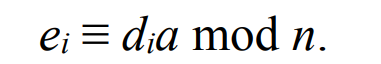
Решением для входа (*S*, *S*) будет такое подмножество из *S*, сумма элементов которого равняется *S*.

Суть метода для шифрования состоит в том, что существуют две различные задачи укладки ранца: одна из них решается *легко* и характеризуется линейным ростом трудоемкости, а другая решается *трудно*. Легкий для укладки ранец можно трансформировать в трудный.

Трудный для укладки ранец применяется в качестве открытого ключа, который легко использовать для зашифрования, но невозможно – для расшифрования. В качестве закрытого ключа применяется легкий для укладки ранец, который предоставляет простой способ расшифрования сообщения.

Открытый ключ e представляет собой нормальную (не сверхвозрастающую) последовательность. Он формируется на основе закрытого ключа и не позволяет легко решить задачу об укладке ранца.

Для получения открытого ключа все значения закрытого ключа умножаются на некоторое число *a* по модулю *n*:



Значение модуля n должно быть больше суммы всех чисел последовательности; кроме того, НОД (*а, n*) = 1.

Закрытый ключ представляет сверхвозрастающая последовательность (последовательность, в которой каждый последующий член больше суммы всех предыдущих).

**Практическая часть**

В соответствии с заданиями, необходимо было реализовать генерацию сверхвозрастающей последовательности (в простейшем случае принимается z = 6 (для кодировки Base64) и z = 8 (для кодировки ASCII)), вычисление нормальной последовательности (открытого ключа), зашифрование сообщения, состоящего из собственных фамилии, имени и отчества, расшифрование сообщения, оценку времени выполнения операций зашифрования и расшифрования. Для реализации всех этих функций был разработан класс RanceCipher, представленный на рисунке 3.1.

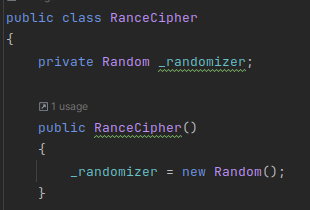


Рисунок 3.1 – Класс RanceCipher

В этом классе были реализованы функции для генерации сверхвозрастающей последовательности (закрытого ключа) и для генерации нормальной последовательности (открытого ключа). Эти функции представлены на рисунках 3.2 – 3.3.

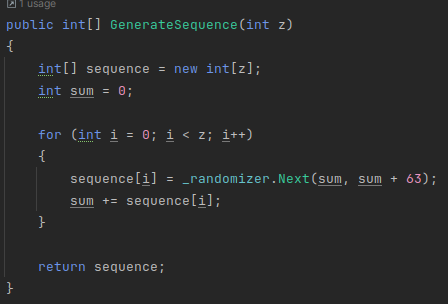


Рисунок 3.2 – Функция GenerateSequence

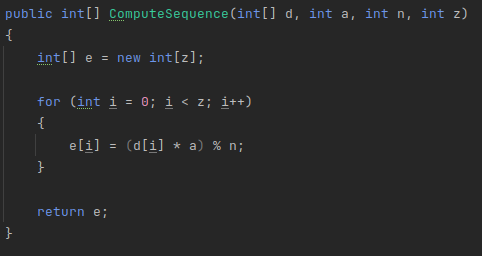


Рисунок 3.3 – Функция ComputeNormalSequence

А для шифрования и расшифрования сообщения были разработаны функции Encrypt() и Decrypt(), представленные на рисунках 3.4 – 3.5.

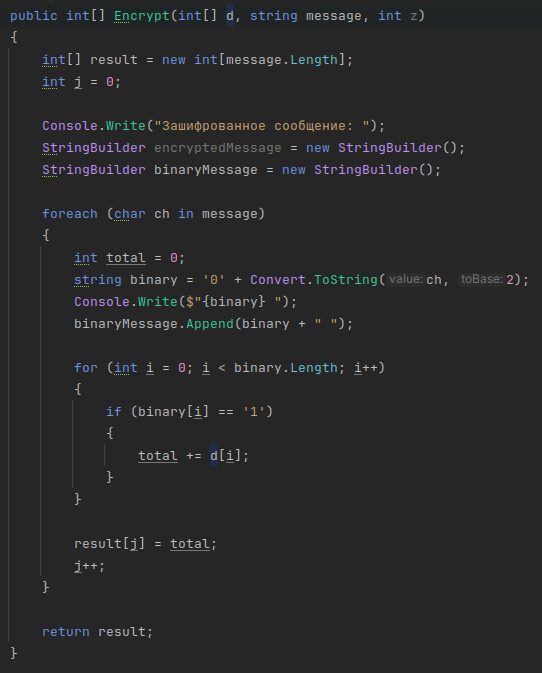


Рисунок 3.4 – Функция Encrypt

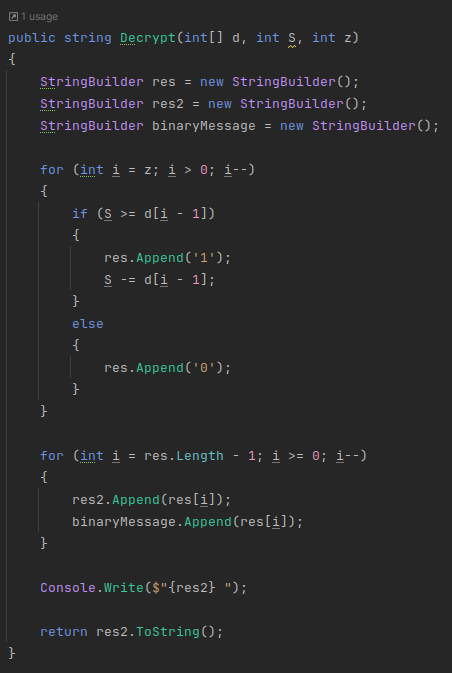


Рисунок 3.5 – Функция Decrypt

В результате, при сгенерированных *d* и *e* с помощью разработанных выше функций, *M* = «Gvozdovskiy Kirill Vladimirovich», *n* = 8, *a* = 31, получим следующий вывод, представленный на рисунке 3.6.

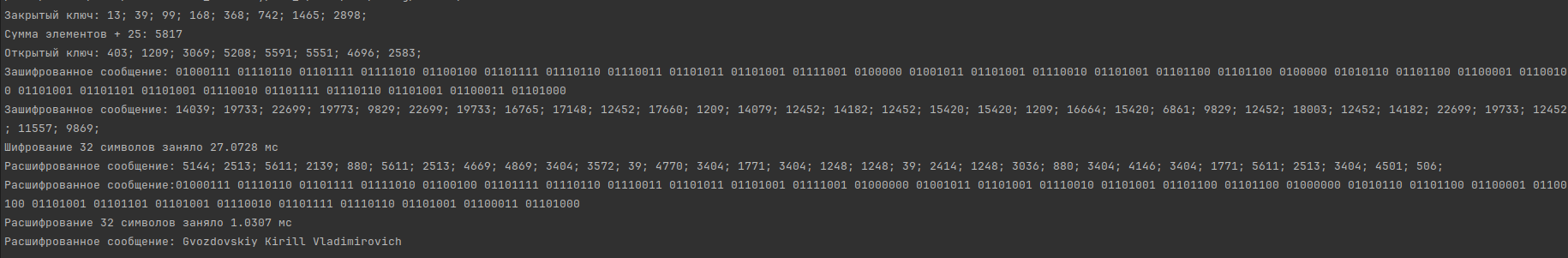


Рисунок 3.6 – Результат работы класса RanceCipher

Также, надо было оценить скорость выполнения операций шифрования и дешифрования. Результаты мы можем увидеть на рисунках 3.7 – 3.8.

Рисунок 3.7 – Оценка скорости шифрования

Рисунок 3.8 – Оценка скорости расшифрования

**Вывод**

В ходе лабораторной работы были изучены и приобретены практические навыки разработки и использования приложений для реализации ассиметричных шифров.

Также было разработано авторское приложение в соответствии с целью лабораторной работы.