Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет Математики и информатики
Кафедра Информационных и управляющих систем
Направление подготовки 09.03.01 – Информатика и вычислительная техника
Направленность (профиль) образовательной программы Автоматизированные системы обработки информации и управления

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. Кафедрой
_____ А.В. Бушманов
«____ » _____ 2023г

БАКАЛАВРСКАЯ РАБОТА

на тему: Разработка приложения для визуализации криптографических алгоритмов

Исполнитель
студент группы 953об _____ (подпись, дата) В.Ю. Подставников

Руководитель
доцент, канд. техн. наук _____ (подпись, дата) С.Г. Самохвалова

Консультант по безопасности и
экологичности
доцент, канд. техн. наук _____ (подпись, дата) А.Б. Булгаков

Нормоконтроль
инженер кафедры _____ (подпись, дата) В.Н. Адаменко

Благовещенск 2023

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет Математики и информатики

Кафедра Информационных и управляющих систем

УТВЕРЖДАЮ
Зав. Кафедрой
_____ А.В. Бушманов
«____ » _____ 2023г

ЗАДАНИЕ

К выпускной квалификационной работе студента: В.Ю. Подставникова

1. Тема выпускной квалификационной работы: Разработка приложения для визуализации криптографических алгоритмов

(утверждена приказом от 24.04.2023 № 974-уч)

2. Срок сдачи студентом законченной работы (проекта):

3. Содержание выпускной квалификационной работы: анализ предметной области; освоение программного и технического обеспечения; разработка алгоритм решения; применение результата на практике.

4. Перечень материалов приложения:

5. Дата выдачи задания:

Руководитель выпускной квалификационной работы:

Самохвалова Светлана Геннадьевна, доцент кафедры ИБ, канд. техн. наук, доцент

(фамилия, имя, отчество, должность, уч. степень, уч. звание)

Задание принял к исполнению (30.01.2023):

(Подпись студента)

РЕФЕРАТ

Выпускная квалификационная работа содержит 65 страниц, 17 рисунков, 1 таблицу, 25 источников.

.NET FRAMEWORK, ДЕСКТОПНОЕ ПРИЛОЖЕНИЕ, КРИПТОГРАФИЧЕСКИЕ АЛГОРИТМЫ, ECDH, RC4, TEA, WINDOWS FORMS, GRAPHVIZ

Цель разработки приложения для визуализации криптографических алгоритмов заключается в создании программного продукта, способного облегчить изучение криптографических алгоритмов студентами АмГУ. Для достижения этой цели необходимо провести исследование и анализ существующих способов визуализации криптографических алгоритмов.

В ходе работы требуется разработать функциональность, позволяющую пользователям визуализировать криптографические алгоритмы из списка включенных в продукт, а также выполнять такие алгоритмы на произвольных входных данных. Приложение должно предоставлять удобный интерфейс для ввода и вывода данных.

Результатом данной работы является десктопное приложение для визуализации криптографических алгоритмов, которое предоставляет пользователям функционал для изучения криптографических алгоритмов. Приложение позволяет строить схемы алгоритмов, визуализировать некоторые из них, а также имеет справочный материал для изучения данных алгоритмов.

СОДЕРЖАНИЕ

НОРМАТИВНЫЕ ССЫЛКИ

В настоящей бакалаврской работе использованы ссылки на следующие стандарты и нормативные документы:

ГОСТ Р ИСО 1503-2014. Эргономика. Требования к пространственной ориентации и направлениям движения органов управления.

ГОСТ Р 50948-2001. Средства отображения информации индивидуального пользования. Общие эргономические требования и требования безопасности.

ГОСТ Р 50949-2001. Средства отображения информации индивидуального пользования. Методы измерений и оценки эргономических параметров и параметров безопасности.

ГОСТ Р 51645-2017. Рабочее место для инвалида по зрению типовое специальное компьютерное. Технические требования к оборудованию и производственной среде.

ГОСТ 28406-89. Персональные электронные вычислительные машины. Интерфейсы видеомониторов. Общие требования

ГОСТ 12.1.007-76 Система стандартов безопасности труда (ССБТ). Вредные вещества. Классификация и общие требования безопасности.

СанПиН 1.2.3685-21. Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания.

СП 2.2.3670-20. Санитарно-эпидемиологические требования к условиям труда.

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ, СОКРАЩЕНИЯ

ECDH – Алгоритм Диффи-Хеллмана на эллиптических кривых

ECC – Криптография на эллиптических кривых

TEA – Tiny Encryption Algorithm

RC4 – Reverce Cipher 4

GUI – Графический интерфейс пользователя

IDE – Интегрированная среда разработки программного обеспечения

ПК – Персональный компьютер

АмГУ – Амурский Государственный Университет

ВВЕДЕНИЕ

В настоящее время информационные технологии играют неотъемлемую роль в нашей жизни, проникая во все сферы деятельности. Они стали непременным инструментом для коммуникации, работы, развлечений и образования. Быстрые темпы развития технологий и все большая доступность высокоскоростного интернета привели к тому, что практически каждому стали доступны неограниченные объемы информации. Однако, с появлением новых возможностей появляются и новые проблемы, среди которых информационная безопасность занимает особое место.

В современном цифровом мире, где данные являются нашей валютой, важность информационной безопасности нельзя недооценивать. Виртуальные преступления, кибератаки, утечки конфиденциальной информации – все это представляет серьезную угрозу для компаний, государственных организаций и даже для обычных пользователей. Каждый день мы сталкиваемся с новыми случаями нарушения информационной безопасности, которые приводят к негативным последствиям, вплоть до финансовых потерь и угрозы личной безопасности.

Важность информационной безопасности обусловлена не только растущим количеством угроз, но и нашей зависимостью от технологий. Все больше данных передается по сетям, хранится в облаке и обрабатывается автоматизированными системами. Мы делаем покупки онлайн, общаемся через социальные сети, храним личные документы в электронном виде. И все это требует эффективных мер по обеспечению безопасности и защите нашей информации от несанкционированного доступа.

Одна из наиболее эффективных мер – шифрование информации. Злоумышленник, получивший зашифрованную надежным алгоритмом информацию, не может получить к ней доступ, а соответственно, и использовать ее в каких-либо целях. Криптографические алгоритмы широко применяются в области кибербезопасности, защиты конфиденциальной информации и цифровых подписей, а также в других сферах, таких как банковское дело, электронная коммерция, мобильные прило-

жения. Однако, эффективное использование криптографических средств требует хотя бы минимального понимания основных принципов работы криптографических алгоритмов, их сильных и слабых сторон.

Несмотря на то, что еще сравнительно недавно (порядка 3 веков назад) криптографические алгоритмы были достаточно тривиальны, современные криптоалгоритмы, используемые в повседневной жизни, достаточно сложны для понимания и освоения, что является актуальной проблемой.

Неоценимый вклад в освоение криптографических алгоритмов могут внести программные средства, динамически визуализирующие работу этих алгоритмов. В качестве потенциальных кандидатов на роль такого программного средства можно выделить такие инструменты, как CrypTool и GCHQ CyberChef.

Разработка программного инструмента, визуализирующего их работу, позволит облегчить знакомство и освоение криптографии в ходе подготовки студентов как профильных, так и не профильных направлений подготовки среднего профессионального и высшего образования, а также обучающихся на курсах повышения квалификации и переподготовки, что позволит повысить уровень ИТ-подготовки и грамотности в области кибербезопасности.

Таким образом, цель данной работы – уменьшить трудоемкость освоения криптографических алгоритмов студентами Амурского Государственного Университета путем разработки программного продукта, позволяющего визуализировать работу таких алгоритмов.

Задачи, которые необходимо решить для достижения данной цели, включают в себя: анализ существующих криптографических алгоритмов; анализ похожих программных продуктов; формулировка требований к программному продукту; разработка программного продукта.

Итогом выполнения данной работы должен являться программный продукт, уменьшающий трудоемкость освоения криптографических алгоритмов студентами АмГУ путем их визуализации. Применение такого продукта в учебном процессе должно привести к уменьшению временных и трудозатрат студентов при изучении криптографических алгоритмов, реализованных в ПО.

1 ОБЩАЯ ХАРАКТЕРИСТИКА ПРЕДМЕТНОЙ ОБЛАСТИ

Разделы криптографии, а также ее вхождения в смежные дисциплины, повсеместно встречаются в программах подготовки среднего профессионального и высшего образования, связанных с информационными технологиями. Наибольшее число таких дисциплин наблюдается в программах подготовки 10.XX.XX, то есть в укрупненной группе специальностей 10.00.00 Информационная безопасность. В программы подготовки по таким направлениям входят дисциплины «Криптографические методы защиты информации», «Криптографические протоколы», «Теоретико-числовые методы в криптографии», «Методы алгебраической геометрии в криптографии», и многие другие. В то же время освоение таких дисциплин сопряжено с трудностями – криптографические алгоритмы бывают масштабными и запутанными.

1.1 Криптография. Семейства криптографических алгоритмов

Криптография является научной дисциплиной, изучающей методы защиты информации от несанкционированного доступа и модификации. Она играет ключевую роль в современном информационном обществе, обеспечивая конфиденциальность, целостность и аутентичность данных.

Существует несколько семейств криптографических алгоритмов, каждое из которых имеет свои особенности и сферы применения. Классификация семейств алгоритмов шифрования приведена на рисунке 1.

Симметричные криптографические алгоритмы, также известные как алгоритмы с секретным ключом, используют один и тот же ключ для шифрования и расшифрования данных. Это самый простой тип криптографических алгоритмов, где отправитель и получатель должны иметь общий ключ, который является секретным.

Симметричные шифры могут быть разделены на две категории: блочные и потоковые шифры.

Блочные шифры оперируют блоками фиксированного размера данных и выполняют шифрование и дешифрование блок за блоком. Размер блока может быть,

например, 64 бита или 128 битов. Наиболее знаменитыми представителями блочных шифров являются:

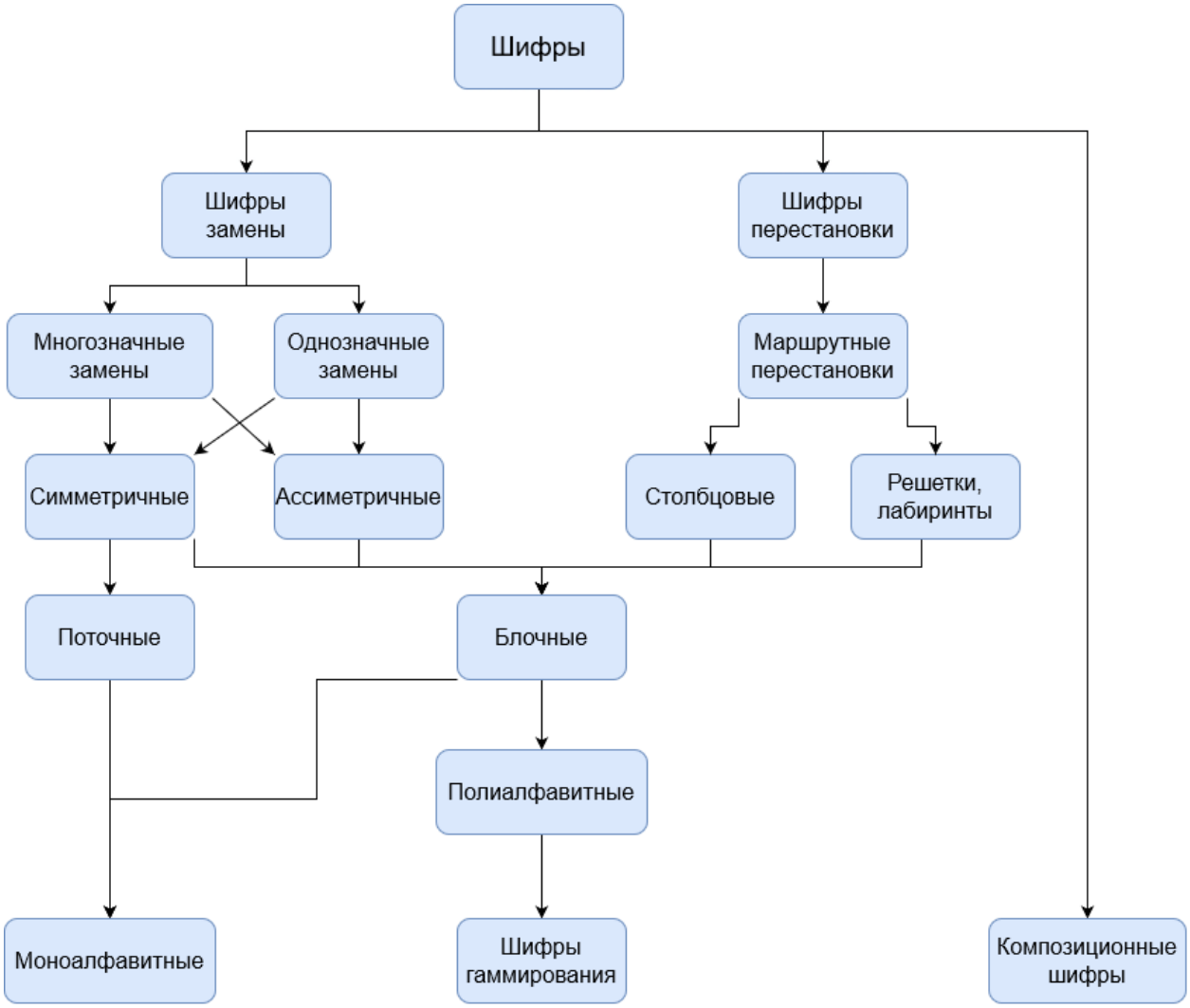

Рисунок 1 – Типы шифров

– Data Encryption Standard (DES): DES был разработан в 1970-х годах и использовался как стандарт шифрования в США. Он оперирует блоками по 64 бита и использует 56-битный ключ. В настоящее время DES считается устаревшим и небезопасным. Для шифрования исходный текст разбивается на блоки по 64 бита. Ключ разбивается на 16 подключей, по одному на каждый раунд шифрования. В каждом раунде выполняются операции перестановки, замены и комбинирования с ключом раунда. В итоге получается зашифрованный блок данных;

– Advanced Encryption Standard (AES): AES является преемником DES и считается одним из наиболее безопасных блочных шифров. Он может оперировать блоками размером 128 бит и поддерживает ключи различных размеров (например,

128, 192 или 256 бит). Для шифрования каждый блок данных проходит через серию раундов, в зависимости от размера ключа (10 раундов для AES-128, 12 раундов для AES-192, 14 раундов для AES-256). В каждом раунде выполняются операции замены байтов, сдвиги строк, смешивание столбцов и комбинирование с ключом раунда. В итоге получается зашифрованный блок данных. Для дешифрования операции производятся в обратном порядке;

– TEA (Tiny Encryption Algorithm): TEA является симметричным блочным шифром, разработанным Дэвидом Уиллсом в 1994 году. Он оперирует блоками по 64 бита и использует 128-битный ключ. TEA прост в реализации и относительно эффективен в использовании ресурсов, но он также был подвержен некоторым атакам и может быть уязвим к известному-открытому тексту. Для генерации ключа TEA использует 128-битовый ключ (2 слова по 64 бита). Ключ разбивается на 4 подключа (по 32 бита каждый). Для шифрования данные разбиваются на блоки по 64 бита. Далее каждый блок данных проходит через несколько раундов шифрования. В каждом раунде выполняются операции сдвигов, сложений по модулю 2^32 и применение подключей. Полученные зашифрованные блоки объединяются вместе для формирования шифрованного сообщения.

Потоковые шифры шифруют данные побитово или побайтово, генерируя поток псевдослучайных битов. Этот поток затем комбинируется с исходными данными с использованием операции побитового XOR. Наиболее известными представителями потоковых шифров являются:

– RC4 (Rivest Cipher 4) является одним из самых широко используемых потоковых шифров. Он был создан в 1987 году и широко применяется в протоколах безопасной передачи данных, таких как SSL и WEP. Однако RC4 был подвергнут различным атакам и уязвимостям, и поэтому не рекомендуется для использования в новых системах. RC4 использует переменную длину ключа (обычно от 40 до 2048 бит). Ключевая последовательность инициализируется с помощью ключа и перестановок внутреннего состояния. Для шифрования данные поступают побайтово на вход RC4. Генерируется псевдослучайный байт из внутреннего состояния RC4. Исходный байт данных комбинируется с псевдослучайным байтом с помощью опера-

ции XOR. Полученный зашифрованный байт передается в выходной поток;

– Salsa20 является потоковым шифром, разработанным Дэниелом Бернштейном. Он обладает высокой скоростью работы и безопасностью. Salsa20 поддерживает ключи различной длины и блокируется 512 битами. Он широко используется в различных приложениях и протоколах, включая SSH и IPsec;

– ChaCha20 также разработан Дэниелом Бернштейном и является усовершенствованной версией Salsa20. Он обеспечивает высокую безопасность и скорость работы. ChaCha20 также поддерживает ключи различной длины и блокируется 512 битами. Он используется в протоколах шифрования трафика, таких как TLS и VPN.

Все эти потоковые шифры оперируют побитово и используют генераторы псевдослучайных чисел для создания потока ключевых битов, которые затем комбинируются с исходными данными с помощью операции XOR. Это позволяет шифровать данные любого размера, в отличие от блочных шифров, которые оперируют фиксированными блоками данных.

Это лишь некоторые из наиболее известных представителей симметричных шифров. Существует множество других алгоритмов, каждый из которых имеет свои особенности и применение в различных областях.

Асимметричные криптографические алгоритмы, также известные как алгоритмы с открытым ключом, используют пару ключей – открытый и закрытый. Открытый ключ используется для шифрования данных, а закрытый ключ – для их расшифровки. Это позволяет отправителю и получателю не обмениваться секретным ключом, что является преимуществом по сравнению с симметричными алгоритмами.

Асимметричные алгоритмы шифрования (или криптография с открытым ключом) являются основой современной криптографии и отличаются от симметричных алгоритмов тем, что они используют разные ключи для шифрования и расшифрования данных. В асимметричной криптографии каждый участник имеет два ключа: публичный ключ, который может быть распространен открыто, и приватный ключ, который должен оставаться в секрете.

Ниже приведено описание наиболее популярных асимметричных алгоритмов.

RSA является одним из самых популярных алгоритмов шифрования, основанных на математических принципах факторизации больших простых чисел. RSA использует два ключа: открытый и закрытый.

Генерация ключей работает по следующему принципу: сначала выбираются два различных простых числа, p и q. Вычисляется их произведение $n = p * q$, которое становится модулем для шифрования и дешифрования. Вычисляется значение функции Эйлера от n, обозначаемое как $\varphi(n) = (p-1) * (q-1)$. Выбирается целое число e $(1 < e < \varphi(n))$, которое является взаимно простым с $\varphi(n)$ и становится открытым ключом. Вычисляется число d, обратное к e по модулю $\varphi(n)$, т.е. $d * e \equiv 1\ (mod\ \varphi(n))$. Число d становится закрытым ключом. Сообщение M представляется в виде числа, меньшего n. Зашифрованное сообщение C вычисляется как $C \equiv M^e\ (mod\ n)$ и дешифруется как $M \equiv C^d\ (mod\ n)$.

DSA является алгоритмом, применяемым для создания и проверки электронных цифровых подписей. DSA также использует два ключа: закрытый и открытый. Генерация ключей работает следующим образом: сначала генерируется большое простое число p. Выбирается простое число q, которое делит $(p-1)$. Выбирается число g, такое что $g^{\frac{p-1}{q}} \equiv 1\ (mod\ p)$. Выбирается случайное число x $(0 < x < q)$, которое становится закрытым ключом. Вычисляется число y, такое что $y \equiv g^x\ (mod\ p)$, которое становится открытым ключом.

Создание электронной цифровой подписи происходит следующим образом: выбирается случайное число $k\ (0 < k < q)$. Вычисляется $r \equiv (g^k\ mod\ p)\ mod\ q$. Вычисляется $s \equiv (k^{-1} * (SHA(M) + x * r))\ mod\ q$, где SHA обозначает хэш-функцию, а M – сообщение. Подпись (r, s) считается электронной цифровой подписью сообщения M.

Проверка электронной цифровой подписи происходит в три этапа: сначала вычисляется

– $w \equiv s^{-1}\ mod\ q$;

– вычисляется $u_1 \equiv (SHA(M) * w)\ mod\ q$ и $u_2 \equiv (r * w)\ mod\ q$;

– вычисляется $v \equiv ((g^{u_1} * y^{u_2})\ mod\ p)\ mod\ q$. Если $v = r$, подпись считается действительной.

ECC использует математические принципы на основе эллиптических кривых для шифрования и создания электронных цифровых подписей. ECC работает с использованием точек на эллиптической кривой и операций над ними. Генерация ключей происходит по следующему алгоритму: выбирается эллиптическая кривая и точка G на ней, которая становится генератором, выбирается случайное число d (закрытый ключ) и вычисляется точка $Q = d * G$ (открытый ключ).

Шифрование происходит следующим образом: выбирается случайное число k, вычисляется точка $C1 = k * G$, вычисляется точка $C2 = M + k * Q$, где M – сообщение. Для дешифрования вычисляется точка $M = C2 - d * C1$, где d – закрытый ключ.

Один из протоколов на основе алгоритма ECC называется ECDH (Elliptic Curve Diffie-Hellman). ECDH используется для обмена секретными ключами между двумя или более участниками коммуникации. Протокол ECDH обеспечивает конфиденциальность и защиту от перехвата данных путем генерации общего секретного ключа между участниками.

Процесс ECDH выглядит следующим образом: каждый участник генерирует свою пару ключей ECC: публичный ключ (обычно представляется точкой на эллиптической кривой) и приватный ключ (случайное число). Участники обмениваются своими публичными ключами по открытым каналам связи. Каждый участник берет публичный ключ другого участника и применяет к нему свой приватный ключ.

Результатом является общий секретный ключ, который одинаков для обоих участников. Общий секретный ключ может быть использован для симметричного шифрования данных, например, с использованием алгоритма AES (Advanced Encryption Standard).

Протокол ECDH обеспечивает безопасность на основе сложности задачи дискретного логарифмирования на эллиптических кривых. Предполагается, что вычислительно сложно определить приватный ключ по публичному ключу, поэтому общий секретный ключ остается недоступным для наблюдателей, даже если они перехватят публичные ключи.

ECC и протоколы, такие как ECDH, широко применяются в различных областях, включая защиту электронных платежей, шифрование электронной почты, виртуальные частные сети (VPN) и многое другое. Их эффективность и безопасность сделали их предпочтительным выбором для многих приложений, особенно в средах с ограниченными ресурсами.

1.2 Выбор алгоритмов для визуализации

Для реализации в разрабатываемом программном продукте необходимо выбрать такой минимальный набор алгоритмов, который будет наиболее репрезентативно представлять существующие семейства криптографических алгоритмов. Классификация алгоритмов шифрования приведена на рисунке 1.

В то же время, выбранные алгоритмы должны быть популярны, а также не слишком сложны, так как главное при изучении семейства алгоритмов – понять основную его суть, что будет затруднено при изучении самых сложных представителей вышеописанных семейств.

Таблица 1 – Сравнительный анализ криптографических алгоритмов

Название	**Тип**	**Год появления**	**Сложность реализации (строчек кода)**	**Актуальность**
Шифр Цезаря	Подстановки, симметричный	58 г. до н.э.	13	Низкая
TEA	Блочный	1998	1500	Высокая
AES	Блочный	1994	75	Средняя
DES	Блочный	1977	500	Средняя
RC4	Потоковый	1987	125	Средняя

Продолжение таблицы 1

1	2	3	4	5
RSA	Асимметрич-ный	1977	1000	Высокая
ECDH	Асимметрич-ный, эллипти-ческие кривые	2005	1000	Высокая

В таблице 1 приведен сравнительный анализ наиболее известных криптографических алгоритмов.

Наиболее репрезентативным набором алгоритмов для их реализации в разрабатываемом программном продукте будет следующий набор:

– шифр Цезаря – один из самых простых алгоритмов шифрования, который основан на замене каждой буквы в сообщении на другую букву, которая находится в нескольких позициях вправо или влево от исходной буквы в алфавите. Несмотря на свою простоту, Шифр Цезаря все еще используется для шифрования некоторых сообщений и является отличным инструментом для обучения основам криптографии;

– Tiny Encryption Algorithm – симметричный блочный алгоритм шифрования, который работает с блоками данных размером 64 бита. TEA был разработан для использования в микроконтроллерах и других системах с ограниченными ресурсами, поскольку он требует небольшого объема памяти и не требует больших вычислительных мощностей. Часто используется как пример при изучении семейства блочных алгоритмов;

– Rivest Cipher 4 – это потоковый алгоритм шифрования, который был разработан Роном Ривестом. Он был широко используется в различных приложениях и протоколах, включая WEP (Wired Equivalent Privacy), который использовался для защиты Wi-Fi-соединений. Часто используется как пример при изучении семейства потоковых шифров;

– Elliptic Curve Diffie-Hellman – алгоритм, который используется для установления общего секретного ключа между двумя сторонами в криптографической системе. Он основан на проблеме дискретного логарифмирования в эллиптических кривых и является одним из самых безопасных алгоритмов для обмена ключами. Закрывает потребность в репрезентации как семейства ассиметричных алгоритмов, так и подсемейства алгоритмов на эллиптических кривых.

Такой набор алгоритмов позволяет наиболее полно охватить существующие разновидности криптографических алгоритмов, при этом использовав их минимальное число.

1.3 Выбор средств визуализации

Визуализация играет важную роль в понимании, анализе и представлении криптографических алгоритмов. Она позволяет исследователям, разработчикам и пользователям лучше понять принципы работы и свойства данных алгоритмов, а также облегчает обучение и взаимодействие с ними.

На данный момент существует несколько подходов к визуализации криптографических алгоритмов, включая диаграммы потоков данных, графические интерфейсы, анимации и схемы алгоритмов. Каждый из этих подходов имеет свои преимущества и недостатки, которые необходимо учитывать при выборе наиболее подходящего метода визуализации для конкретной задачи.

Визуализация криптографических алгоритмов с использованием диаграмм потоков данных предоставляет наглядное представление о том, как данные перемещаются внутри алгоритма, отображая их поток от одного шага к другому.

Диаграммы потоков данных позволяют легко представить последовательность операций и контрольных структур, что облегчает понимание работы алгоритма. Однако такой подход может быть чрезвычайно сложным для понимания, особенно для сложных алгоритмов с большим количеством шагов и ветвлений.

В то же время, он слишком поверхностный, так как описывает только перемещение данных, но не дает подробного ответа на вопрос «как именно обрабатываются эти данные?» Пример такой диаграммы изображен на рисунке 2.

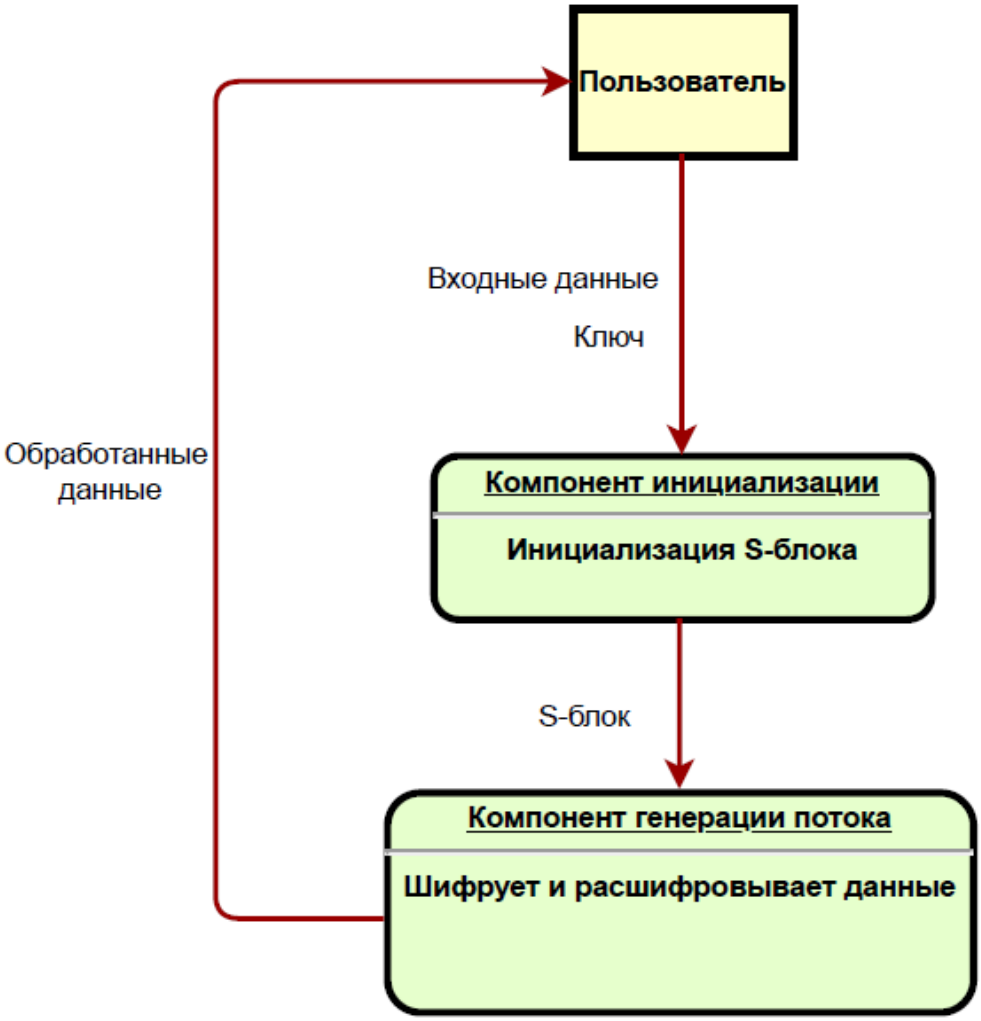

Рисунок 2 – Диаграмма потоков данных для алгоритма RC4

Графические интерфейсы отладки представляют собой еще один способ визуализации криптографических алгоритмов. Они позволяют пользователям взаимодействовать с алгоритмом, наблюдать изменения в реальном времени и получать наглядное представление о его работе. Графические интерфейсы отладки могут быть полезны при анализе алгоритмов, поскольку они позволяют выводить на экран данные и результаты шагов алгоритма. Примером такой визуализации можно считать отладчик в интегрированной среде разработки программного обеспечения (рисунок 3). Тем не менее, хоть такой подход является достаточно универсальным, он содержит минимум графической и максимум текстовой информации, что сильно снижает способность к восприятию этой информации на достаточно больших алгоритмах, так как алгоритм сводится к последовательности низкоуровневых шагов, а не абстракций.

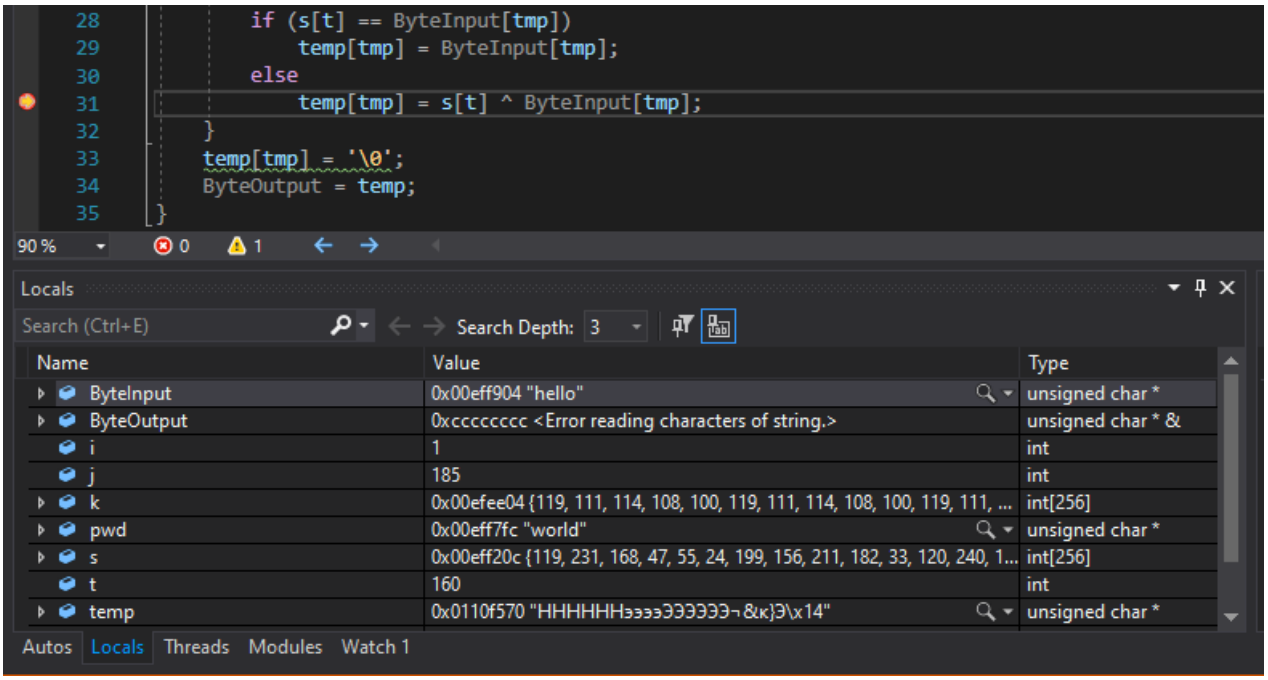

Рисунок 3 – Отладка RC4 в Microsoft Visual Studio

Анимации представляют собой эффективный и привлекательный способ демонстрации работы криптографических алгоритмов. Они позволяют увидеть важнейшие шаги алгоритма в динамическом режиме, что способствует лучшему пониманию его работы. Анимации обычно визуализируют перемещение данных, операции и промежуточные результаты во время выполнения алгоритма.

Однако анимации могут быть ограничены по времени и не всегда позволяют полностью уловить все детали и варианты работы алгоритма, что может быть компенсировано путем предоставления пользователю интерактивного способа управлять изображением. Также смежным с таким способом визуализации является построение интерактивного графика, отражающего какой-либо важный аспект алгоритма. На рисунке 4 представлена раскадровка анимации шифрования методом шифра Цезаря.

Схема алгоритма является универсальным способом визуализации алгоритма. Она содержит графическую информацию, понятную многим, а также высокоуровневые абстракции, которые облегчают восприятие сути алгоритма. Минусом такого подхода является «отдаленность» схемы алгоритма от реальных примеров, обобществление сути алгоритма, а также большой размер для сложных алгоритмов, что также может затруднить понимание. Пример схемы алгоритма для

шифра Цезаря приведен на рисунке 5.

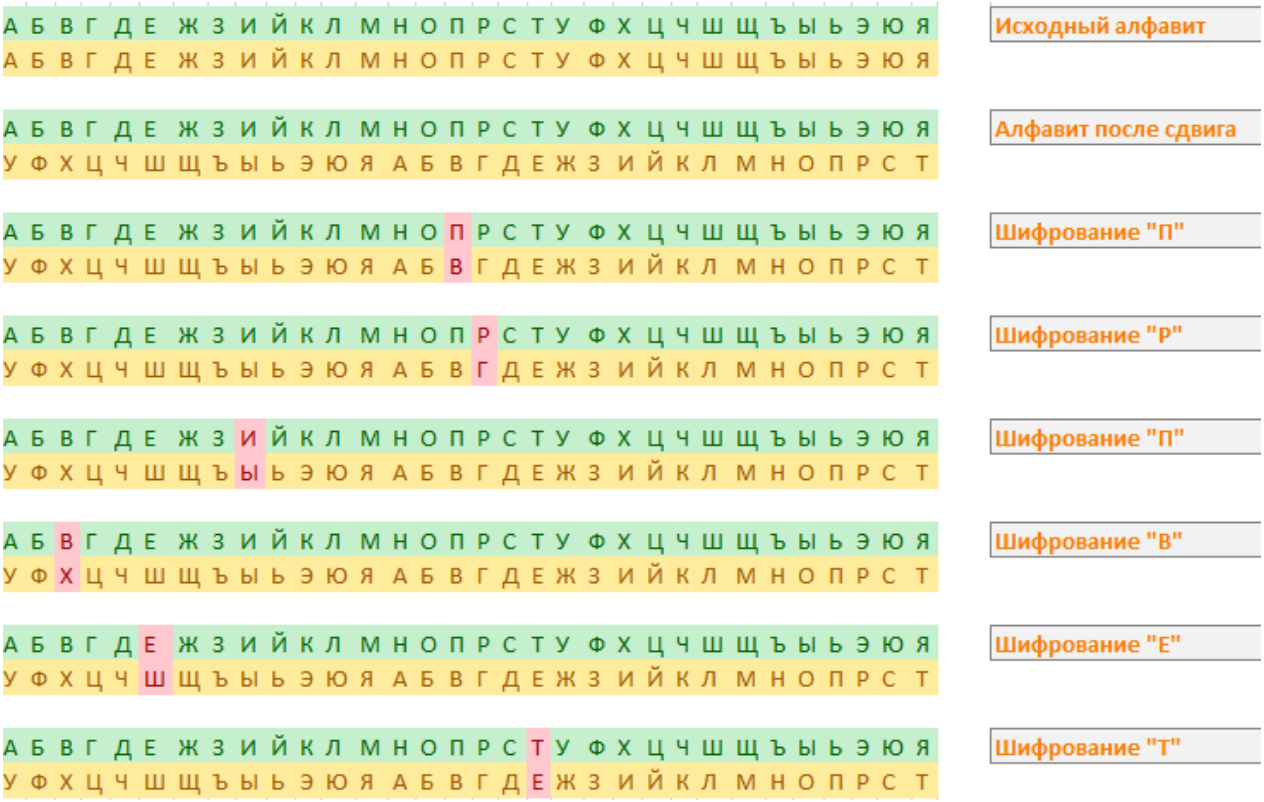

Рисунок 4 – Раскадровка анимации. Шифр Цезаря

Исходя из проведенного анализа существующих методов визуализации криптографических алгоритмов можно сделать вывод, что наиболее универсальным и эффективным способом визуализации является построение схемы алгоритма для конкретного набора параметров.

Схема алгоритма представляет собой абстрактное представление о последовательности шагов, операций и условий, выполняемых внутри алгоритма, а включение в нее конкретного набора параметров, задаваемого пользователем, позволяет объединить анализ общей структуры алгоритма и анализ конкретного примера.

Построение схемы алгоритма позволяет легко воспринимать структуру алгоритма и его основные компоненты. Такая схема позволит более глубоко понять принципы работы алгоритма и анализировать его функциональность. Схема алгоритма может быть использована для облегчения процесса анализа и сравнения различных алгоритмов, а также для обучения и взаимодействия с алгоритмами.

При выборе набора параметров для построения схемы алгоритма необходимо учитывать конкретные требования и цели исследования. Параметры алгоритма, та-

кие как размер ключа, типы операций и конкретные шаги, должны быть определены и учтены при построении схемы. Такой подход позволяет создать наиболее точное и понятное представление о работе алгоритма в конкретных условиях.

Тем не менее, несмотря на универсальность способа визуализации построением схемы алгоритма, для некоторых алгоритмов существует также возможность выделить ключевые особенности в графическую форму, и, соответственно, анимировать изменения по ходу выполнения такого алгоритма.

Примером такого алгоритма является Шифр Цезаря. В свою очередь, геометрическая основа эллиптических кривых позволяет визуализировать этот алгоритм интерактивным графиком используемой кривой с отмеченными на ней используемыми точками.

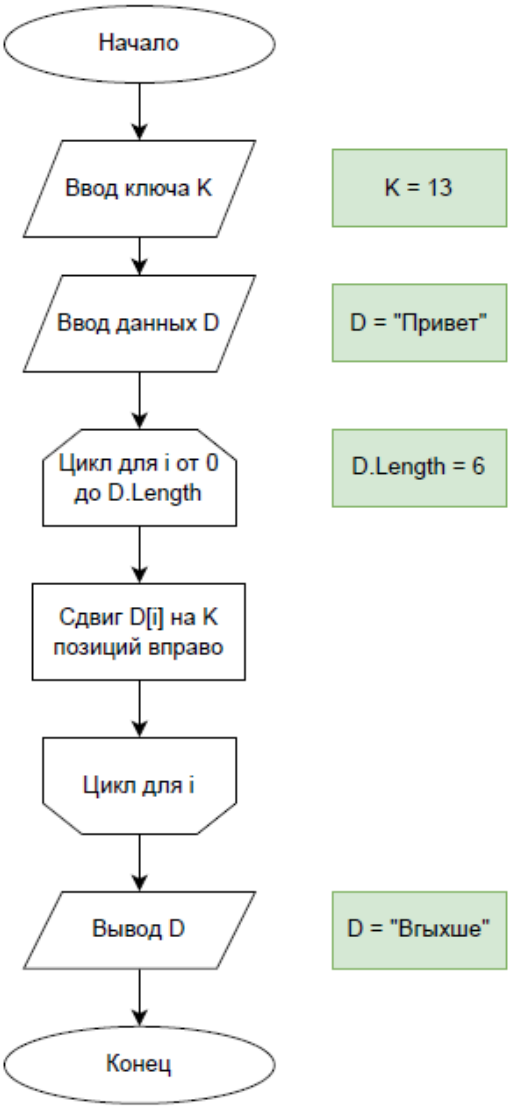

Рисунок 5 – Схема алгоритма. Шифр Цезаря

На рисунке 6 представлен типичный вид графика эллиптической кривой.

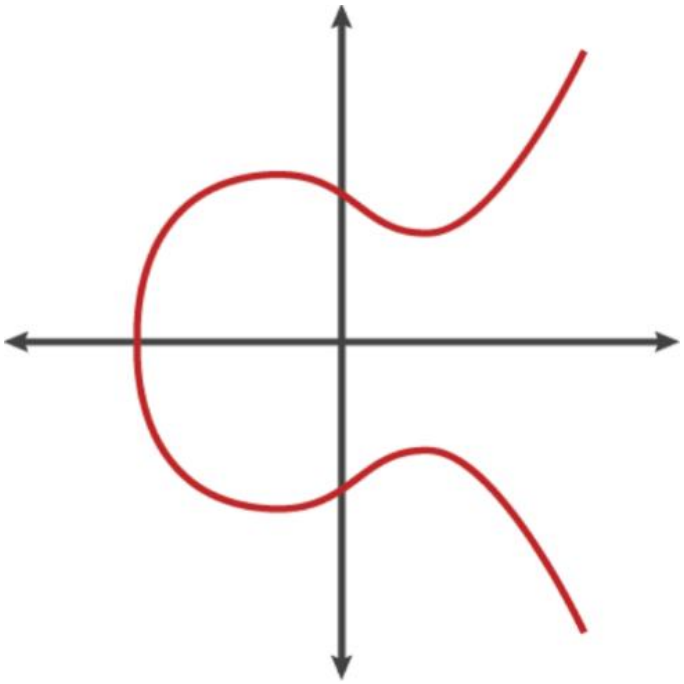

Рисунок 6 – Эллиптическая кривая

Таким образом, предполагается визуализация шифра Цезаря и ECDH, а также построение схемы алгоритма для всех выбранных для реализации криптографических алгоритмов.

Для алгоритма Диффи-Хеллмана на эллиптических кривых такая визуализация (построение эллиптической кривой) поможет наглядно увидеть распределение точек на этой самой кривой.

Для шифра Цезаря построение динамической анимации с возможностью итерироваться по слайдам позволит наглядно увидеть процесс замены символа исходного алфавита на символ шифртекста, а также увидеть четкую связь между открытым и закрытым алфавитом, заключающуюся в сдвиге на определенное число.

2 ОБОСНОВАНИЕ РАЗРАБОТКИ ПРОГРАММНОГО ПРОДУКТА

Проблема сложности восприятия криптографических алгоритмов напрямую связана со сложностью используемого в данных алгоритмах математического аппарата. Криптографические алгоритмы часто основаны на сложных математических концепциях и алгоритмах, таких как теория чисел, алгебраическая геометрия, дискретное преобразование Фурье.

В то же время криптографические алгоритмы могут включать в себя различные примитивы, такие как шифры блочных и поточных шифров, хэш-функции, асимметричное шифрование и цифровые подписи. Каждый из этих примитивов имеет свои особенности и сложности в реализации и понимании. И именно понимание таких примитивом является ключом к пониманию большинства криптографических алгоритмов.

В то время как стандартный учебный процесс студентов направлений подготовки, связанных с информационной безопасностью, подразумевает преодоление сложностей восприятия упорством и усидчивостью (изучение специализированной литературы, выполнение практических заданий), возможно предложить и новаторский способ, позволяющий решить описываемую проблему – а именно, визуализация изучаемых алгоритмов с целью их декомпозиции.

В то же время визуализация должна быть динамической и описывать конкретный, а не абстрактный пример, так как изучение от частного к общему упрощает восприятие сложной информации.

Для выделения конкретных требований к разрабатываемому программному продукту необходимо проанализировать существующие решения, так или иначе позволяющие решить рассматриваемую проблему.

2.1 CrypTool Online

CrypTool Online представляет из себя веб-ресурс, направленный на энтузиастов, изучающих криптографию. На рисунке 7 представлена главная страница данного сервиса. Сервис располагается по адресу https://www.cryptool.org/en/cto/.

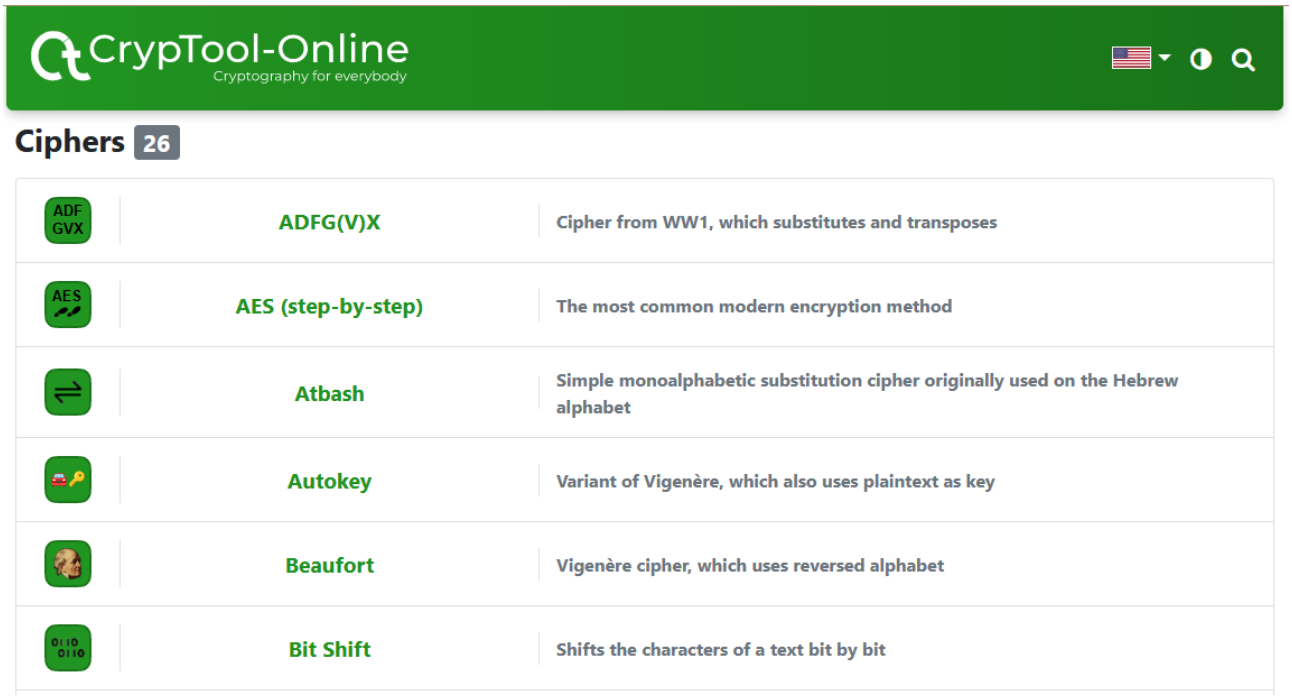

Рисунок 7 – Главная страница CrypTool

Данный сервис имеет достаточно широкий набор реализованных шифров, в числе которых AES, Enigma, RSA, Шифр Виженера, Атбаш, Скитала, Шифр Цезаря и другие. Для некоторых (AES, Enigma, RSA) шифров есть возможность выполнения по шагам (рис. 8).

Для каждого алгоритма имеется короткое описание, история возникновения, примеры входных данных. В качестве положительных сторон можно выделить доступность веб-приложения: Cryptool Online может быть использован из любого устройства с доступом в Интернет. Это позволяет пользователям использовать его на различных платформах, включая компьютеры, планшеты и смартфоны. Сервис имеет приятный и удобный пользователю интерфейс, который, впрочем, имеет локализацию только на английском и немецком языках. Также выполнение «шаг за шагом» присутствует не для всех алгоритмов, а значит, остальные реализации имеют в функционале только возможность шифрования/дешифрования, а также текстовую справку по алгоритму, чего может быть мало обучающемуся, намеренному освоить такой алгоритм.

2.2 CyberChef

CyberChef представляет собой программное обеспечение для обработки и анализа данных в области кибербезопасности.

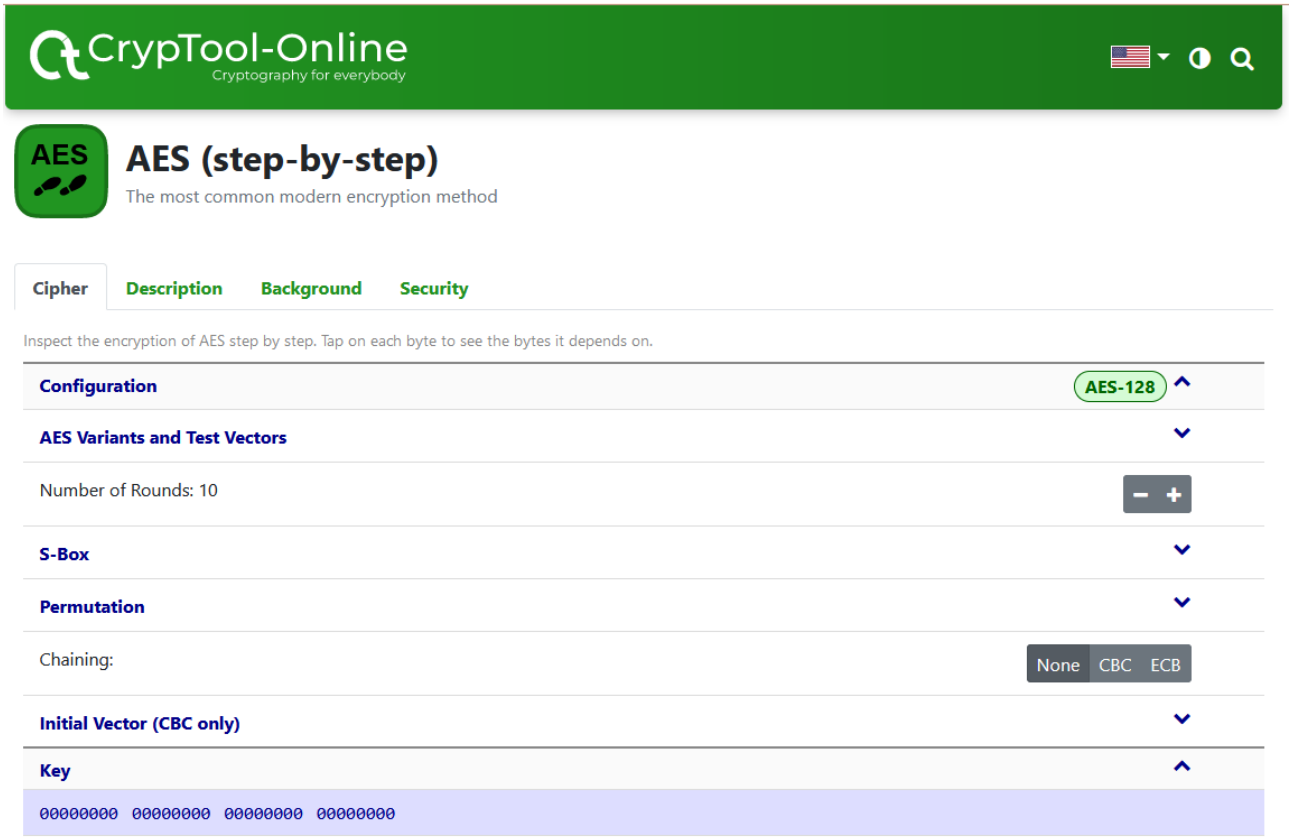

Рисунок 8 – Страница «AES (пошаговый)»

Оно разработано для автоматизации различных операций, связанных с манипуляцией и преобразованием данных, таких как кодирование, декодирование, шифрование, дешифрование и другие операции обработки. Сервис располагается по адресу https://gchq.github.io/CyberChef.

CyberChef имеет простой и интуитивно понятный интерфейс, который позволяет пользователям комбинировать различные операции обработки данных, называемые "рецептами". Рецепты представляют собой последовательность шагов, каждый из которых выполняет конкретную операцию над данными. Это позволяет пользователям создавать мощные и гибкие цепочки обработки данных, а также делиться ими с другими пользователями. Главное окно CyberChef представлено на рисунке 9.

CyberChef поддерживает большое количество операций обработки данных. Например, он может выполнять декодирование различных типов кодировок, таких как Base64, Hex, URL и других. Он также обладает возможностями для работы с шифрованием и дешифрованием, включая такие алгоритмы, как AES, RSA, Blowfish и многие другие.

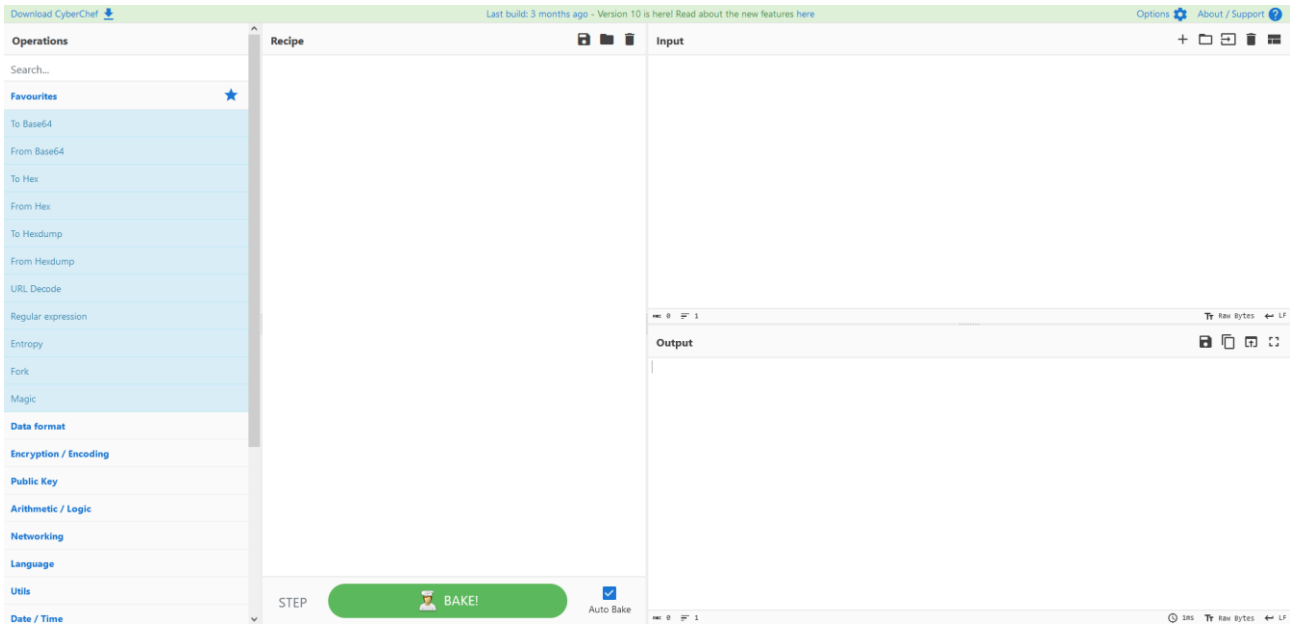

Рисунок 9 – Главное окно CyberChef

Кроме того, CyberChef может выполнять операции сжатия, хеширования, обработки текста и многое другое. Список алгоритмов, поддерживаемых данным сервисом, включает в себя:

– AES, DES, 3DES, Blowfish, RC2 – блочные шифры;

– RC4 – потоковый шифр;

– ROT13, A1Z26 – шифр замены.

Тем не менее, в контексте рассматриваемой проблемы данный сервис проявляет себя не лучшим образом. Несмотря на то, что он позволяет динамически выполнить различные комбинации операций над входным набором данных, он не показывает промежуточные результаты и никак их не визуализирует. Пользователь видит только «черный ящик» структуры вход-список операций-выход. Такой подход не позволяет эффективно изучать суть работы криптографических алгоритмов.

Еще одним минусом данного сервиса является отсутствие русской локализации, что безусловно является еще одним барьером на пути к освоению исследуемой темы для среднестатистического российского пользователя.

Также среди недостатков данного сервиса можно выделить:

– ограниченная функциональность: Возможности CyberChef, хотя и впечатляющие, все же ограничены в сравнении с некоторыми другими инструментами.

Например, для некоторых сложных операций, таких как декомпрессия и дешифрование специфических форматов данных, CyberChef может быть неэффективен или даже неспособен выполнить задачу;

– сложность использования: при работе с CyberChef может потребоваться определенный уровень технического понимания, особенно при выполнении более сложных операций. Некоторые рецепты в CyberChef требуют знания различных концепций и форматов данных, иначе может быть сложно понять, как правильно настроить операции и достичь желаемых результатов;

– ограниченная документация: документация по CyberChef иногда может быть ограниченной или не полностью актуальной. Это может затруднить понимание определенных функций или настройку сложных операций. В таких случаях может потребоваться самостоятельное исследование и экспериментирование для решения проблем или достижения нужного результата;

– не подходит для всех типов задач: CyberChef преимущественно ориентирован на обработку и анализ данных в текстовом формате. Он может быть не настолько эффективен, если требуется работа с другими типами данных, такими как изображения, видео или аудио. В таких случаях могут потребоваться специализированные инструменты;

– необходимость интернет-соединения: CyberChef предоставляется в виде веб-приложения, поэтому для его использования требуется постоянное интернет-соединение. Это может быть недостатком в случаях, когда вы работаете в ограниченной сетевой среде или при отсутствии стабильного интернет-подключения.

2.3 Формулировка требований к программному продукту

Исходя из анализа существующих решений можно однозначно заключить, что специфика изучения криптографических алгоритмов студентами российских вузов делает целесообразной разработку отдельного программного продукта, решающего проблему сложности изучения криптографических алгоритмов, так как возможностей существующих решений недостаточно для удовлетворения всех требований к решению. Можно выделить следующие требования к такому программному продукту:

– русскоязычный интерфейс;

– работа на компьютерах с ОС Windows 10 и новее;

– возможность визуализации алгоритмов «Шифр Цезаря», TEA, RC4, ECDH;

– вывод результата визуализации в файл;

– наличие справки по реализованным алгоритмам;

– возможность выполнить выбранный алгоритм на произвольном наборе данных с произвольным ключом;

– поддержка различных форматов входных и выходных данных;

– возможность дешифровки сообщений с имеющимся ключом;

– возможность пошагового выполнения алгоритмов, к которым применим такой подход.

Разработка программного продукта, соответствующего таким требованиям, может оказать следующие положительные эффекты: улучшение качества обучения: благодаря возможности визуализации работы алгоритмов, студенты смогут лучше понимать принципы работы шифрования и дешифрования данных, а также получать быструю обратную связь по выполненным заданиям; экономия времени: использование программного комплекса позволит упростить процесс выдачи заданий для индивидуального изучения студентам, что сэкономит время преподавателя; увеличение интереса к изучению темы: интерактивный подход к обучению и возможность самостоятельной работы над заданиями может привести к увеличению мотивации студентов к изучению криптографии.

Таким образом, использование разрабатываемого программного комплекса может привести к повышению качества обучения и улучшению результатов студентов в области криптографии.

3 РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА

Для реализации программного продукта необходимо выполнить ряд шагов разработки:

– сформулировать требования;

– выбрать инструменты и средства разработки;

– выбрать архитектуру и шаблоны проектирования;

– спроектировать структуру модулей и классов;

– спроектировать пользовательский интерфейс;

– реализовать программный продукт на основе выбранных компонентов;

– протестировать готовый программный продукт.

3.1 Выбор инструментов и средств разработки

Для реализации программного продукта следует определиться с выбором используемых инструментов и средств разработки, включая используемые языки программирования, а также библиотеки и IDE.

C# (C Sharp) – объектно-ориентированный язык программирования, разработанный компанией Microsoft. Преимущества использования C# включают в себя простоту – C# является одним из наиболее понятных и легко изучаемых языков программирования. Он имеет синтаксис, похожий на язык программирования Java, что делает его доступным для многих разработчиков. C# обладает сильной типизацией, что обеспечивает большую безопасность и устраняет множество ошибок во время компиляции. C# также является основным языком программирования для разработки приложений под платформу Windows. Он обеспечивает прямой доступ к API операционной системы Windows, что позволяет разрабатывать полнофункциональные и эффективные приложения. Данный язык также является одним из языков программирования, поддерживаемых платформой .NET. Это открывает доступ к большому набору библиотек и инструментов разработки, которые значительно упрощают процесс разработки.

Windows Forms. Данный фреймворк известен простотой использования – Windows Forms предоставляет интуитивно понятный дизайнер графического ин-

терфейса (GUI), который позволяет разработчикам легко создавать интерактивные пользовательские интерфейсы. Он предоставляет множество готовых элементов управления (кнопки, текстовые поля, списки и т.д.), которые могут быть просто перетащены и настроены. Windows Forms является частью .NET Framework, что означает, что оно интегрируется со всей экосистемой .NET и получает полную поддержку и обновления от Microsoft. Это обеспечивает стабильность и долгосрочную поддержку для разработчиков. Приложения, созданные с помощью Windows Forms, могут быть легко развернуты на компьютерах под управлением Windows. Они работают без особых требований к системным ресурсам и не требуют установки дополнительных компонентов.

.NET – это платформа разработки программного обеспечения, разработанная компанией Microsoft. Она предлагает мощный набор инструментов, библиотек и языков программирования для создания разнообразных типов приложений, включая веб-приложения, настольные приложения, мобильные приложения и службы. .NET поддерживает несколько языков программирования, включая C#, Visual Basic.NET и F#. Это дает разработчикам возможность выбрать язык, с которым они наиболее знакомы или который лучше подходит для конкретной задачи. Обширная стандартная библиотека: .NET поставляется с обширной стандартной библиотекой, которая предлагает множество готовых компонентов и функций для разработки приложений. Это упрощает разработку, поскольку многие повседневные задачи уже реализованы и доступны для использования. Введение платформы .NET Core расширило возможности .NET, позволяя разрабатывать кросс-платформенные приложения, которые могут работать на различных операционных системах, включая Windows, macOS и Linux. Это упрощает достижение широкой аудитории пользователей. Интеграция с другими технологиями Microsoft: .NET предлагает интеграцию с другими технологиями Microsoft, такими как SQL Server, Azure и SharePoint. Это обеспечивает беспроблемную работу с другими продуктами и сервисами от Microsoft и упрощает разработку приложений в экосистеме Microsoft.

Graphviz – это пакет программного обеспечения с открытым исходным кодом для визуализации графов и сетей. Он предоставляет набор инструментов и библио-

тек для создания и рендеринга графических представлений графовых структур. Graphviz предлагает простой и декларативный язык для описания графов и их визуализации. Разработчику необходимо только описать структуру графа и его связи, а Graphviz автоматически создаст графическое представление. Graphviz предоставляет механизм автоматического размещения узлов и ребер на графической плоскости. Это упрощает создание чистых и понятных визуализаций графов без необходимости вручную настраивать расположение элементов. Graphviz имеет модульную архитектуру, позволяющую расширять его функциональность с помощью плагинов и пользовательских расширений. Это позволяет создавать дополнительные алгоритмы размещения, стилизации и экспорта графов. Graphviz может экспортировать графы в различные форматы, включая PNG, SVG, PDF и другие. Это позволяет использовать визуализации графов в различных контекстах, включая документацию, презентации и веб-страницы.

Python.Included – это библиотека, которая позволяет использовать функциональность Python в приложениях на C#. Она предоставляет возможность запуска Python-скриптов, вызова Python-функций и обмена данными между C# и Python. В случае уже имеющегося существующего приложения на C# и необходимости в специфической функциональности, которую предлагает Python, Python.Included позволяет легко интегрировать Python-код в приложение. Python является одним из наиболее популярных языков программирования с богатой экосистемой библиотек и инструментов. Используя Python.Included, можно получить доступ к этой экосистеме и использовать готовые решения для своих задач.

NumPy (Numerical Python) является основным пакетом для научных вычислений в Python. Он предоставляет мощные инструменты для работы с многомерными массивами данных, математическими функциями, алгоритмами линейной алгебры и преобразованиями Фурье. NumPy использует оптимизированные алгоритмы и векторизацию, что делает вычисления быстрыми и эффективными. NumPy поддерживает многомерные массивы, что позволяет эффективно работать с данными больших размеров и выполнять операции над ними. NumPy предоставляет функции для выполнения различных операций, таких как математические вычис-

ления, статистика, линейная алгебра и многое другое. NumPy является основой для многих других библиотек Python, таких как SciPy, pandas и машинное обучение библиотеки, что обеспечивает легкую интеграцию с ними.

Matplotlib.pyplot (часто называемый просто pyplot) является частью библиотеки Matplotlib и предоставляет возможности для создания графиков и визуализации данных. Pyplot предоставляет простой и интуитивно понятный интерфейс для создания различных типов графиков и диаграмм. Pyplot поддерживает широкий спектр типов графиков, включая линейные графики, гистограммы, диаграммы рассеяния, круговые диаграммы и другие. Pyplot предоставляет множество опций для настройки внешнего вида графиков, таких как оси, заголовки, легенды, цвета и т. д. Pyplot интегрируется хорошо с NumPy, что позволяет удобно использовать массивы данных из NumPy для построения графиков.

Visual Studio 2019 – это интегрированная среда разработки (IDE) от Microsoft, которая обеспечивает широкий набор инструментов для разработки программного обеспечения. Visual Studio 2019 предоставляет полноценную поддержку языка C#, включая редактор кода с подсветкой синтаксиса, автозаполнением, отладчиком и другими инструментами. Visual Studio 2019 обладает широким набором инструментов разработки, таких как интегрированный отладчик, система контроля версий, инструменты для развертывания и управления проектами. Visual Studio 2019 позволяет разрабатывать и отлаживать C#-приложения как для Windows, так и для других платформ, включая Linux и macOS. Visual Studio 2019 интегрируется с другими инструментами и технологиями Microsoft, такими как .NET Framework, Azure и многими другими, что обеспечивает больше возможностей для разработки и развертывания приложений.

Вышеперечисленный набор средств позволяет вести эффективную разработку программного продукта наиболее подходящими для каждой подзадачи инструментами.

3.2 Функциональные и обеспечивающие подсистемы

Функциональные подсистемы включают в себя:

– подсистему управления шифрованием и дешифрованием. Данная подсистема управляет непосредственно исполнением криптографических алгоритмов

над введенными пользователем данными;

– подсистему построения схем алгоритмов. Строит схемы алгоритма для выбранного алгоритма и сохраняет их в .pdf файл;

– подсистему построения графиков. Используется для визуализации эллиптических кривых в алгоритме ECDH;

– подсистему справки. Выводит справочную информацию, содержащую базовую информацию об алгоритме;

– подсистему графического интерфейса. Эта подсистема представляет из себя GUI, с помощью которого пользователь взаимодействует с программой.

Обеспечивающие подсистемы включают в себя:

– информационное обеспечение (информация, вводимая пользователем);

– математическое обеспечение (криптографические алгоритмы);

– программное обеспечение (среда выполнения CLR, .NET);

– техническое обеспечение (ПК с Microsoft Windows 10 и новее);

– эргономическое обеспечение (рабочее место пользователя программного продукта);

– языковое обеспечение (C#, Python, русский язык, английский язык).

3.3 Выбор архитектуры и шаблонов проектирования

Архитектура разрабатываемого приложения включает в себя интерфейс ICryptoAlgorithm и несколько классов, которые наследуются от этого интерфейса: CaesarCipherAlgorithm, RC4Algorithm, TinyEncryptionAlgorithm и ECDHAlgorithm. Ниже представлен подробный обзор используемых паттернов проектирования и структуры кода.

Интерфейс ICryptoAlgorithm:

– GetAlgorithmState(): Метод возвращает текущее состояние алгоритма в виде массива байтов;

– Encrypt(): Метод, который выполняет шифрование данных и возвращает зашифрованную строку;

– Decrypt(): Метод, который выполняет расшифровку данных и возвращает исходную строку;

– PerformEncryptionStep(): Метод, который выполняет один шаг шифрования.

Класс CaesarCipherAlgorithm:

– наследуется от ICryptoAlgorithm;

– реализует методы GetAlgorithmState(), Encrypt(), Decrypt() и PerformEncryptionStep() с учетом алгоритма шифрования методом Цезаря;

Класс RC4Algorithm:

– наследуется от ICryptoAlgorithm;

– реализует методы GetAlgorithmState(), Encrypt(), Decrypt() и PerformEncryptionStep() с учетом алгоритма шифрования RC4;

Класс TinyEncryptionAlgorithm:

– наследуется от ICryptoAlgorithm;

– реализует методы GetAlgorithmState(), Encrypt(), Decrypt() и PerformEncryptionStep() с учетом алгоритма Tiny Encryption Algorithm.

Класс ECDHAlgorithm:

– наследуется от ICryptoAlgorithm;

– реализует методы GetAlgorithmState(), Encrypt(), Decrypt() и PerformEncryptionStep() с учетом алгоритма обмена ключами ECDH (Elliptic Curve Diffie-Hellman).

Структура кода:

– файл ICryptoAlgorithm.cs содержит определение интерфейса ICryptoAlgorithm;

– файл CaesarCipherAlgorithm.cs содержит определение класса CaesarCipherAlgorithm и его реализацию методов на основе алгоритма шифрования Цезаря;

– файл RC4Algorithm.cs содержит определение класса RC4Algorithm и его реализацию методов на основе алгоритма шифрования RC4;

– файл TinyEncryptionAlgorithm.cs содержит определение класса TinyEncryptionAlgorithm и его реализацию методов на основе алгоритма Tiny Encryption Algorithm;

– файл ECDHAlgorithm.cs содержит определение класса ECDHAlgorithm и его реализацию методов на основе алгоритма обмена ключами ECDH;

– файл MainForm.cs содержит графический интерфейс пользователя для главного окна программы;

– файл Docs.cs содержит графический интерфейс пользователя для окна справки.

Общая структура кода следует принципам объектно-ориентированного программирования, где каждый класс отвечает за конкретный алгоритм шифрования, реализует интерфейс ICryptoAlgorithm и предоставляет свою собственную реализацию методов. Это позволяет легко добавлять новые алгоритмы шифрования и обеспечивает гибкость и расширяемость системы.

Говоря о паттернах проектирования, которые следует использовать исходя из такой структуры классов, стоит упомянуть следующие:

Паттерн «Стратегия» (Strategy). Этот паттерн позволяет определить семейство алгоритмов, инкапсулировать каждый из них и сделать их взаимозаменяемыми. В данном случае интерфейс ICryptoAlgorithm выступает как стратегия, а классы CaesarCipherAlgorithm, RC4Algorithm, TinyEncryptionAlgorithm и ECDHAlgorithm представляют конкретные реализации этих алгоритмов. Таким образом, можно легко добавлять новые алгоритмы шифрования, сохраняя совместимость с общим интерфейсом.

Паттерн «Фабричный метод» (Factory Method). Возможно, в приложении применяется фабричный метод для создания объектов классов шифрования. Например, можно иметь абстрактную фабрику CryptoAlgorithmFactory, которая определяет метод CreateCryptoAlgorithm(), а конкретные фабрики (CaesarCipherAlgorithmFactory, RC4AlgorithmFactory, и т.д.) реализуют этот метод для создания соответствующих объектов шифрования. Это позволяет упростить процесс создания экземпляров классов шифрования и обеспечивает гибкость в выборе конкретной реализации.

Паттерн «Одиночка» (Singleton). Если требуется гарантировать, что экземпляр класса ICryptoAlgorithm с определенным алгоритмом создается только один

раз, можно использовать паттерн Одиночка. В этом случае классы реализации шифрования (CaesarCipherAlgorithm, RC4Algorithm, и т.д.) могут быть реализованы как одиночки, чтобы обеспечить глобальный доступ к этим объектам.

Паттерн «Декоратор» (Decorator). Если требуется добавить дополнительную функциональность или модифицировать поведение объектов классов шифрования без изменения их исходного кода, можно применить паттерн Декоратор. Например, можно создать декораторы, которые добавляют дополнительные шаги шифрования или обеспечивают дополнительные проверки без изменения основной логики шифрования в классах CaesarCipherAlgorithm, RC4Algorithm и других.

3.4 Проектирование пользовательского интерфейса

Ниже приведены некоторые требования к качественному графическому интерфейсу пользователя (GUI):

– интуитивность и простота использования – GUI должен быть интуитивно понятным и простым для использования даже для непрофессиональных пользователей. Взаимодействие с интерфейсом должно быть легким и естественным, минимизируя необходимость в дополнительном обучении или чтении документации;

– понятная и последовательная структура – графический интерфейс должен иметь понятную и последовательную структуру, позволяющую пользователям легко найти нужные функции и выполнять операции. Навигация и расположение элементов управления должны быть логичными и интуитивными;

– визуальная привлекательность – GUI должен быть визуально привлекательным с использованием гармоничных цветов, подходящих шрифтов и привлекательных иконок. Графический дизайн должен соответствовать общей теме приложения и оставлять положительное впечатление у пользователей;

– отзывчивость и производительность – интерфейс должен реагировать на действия пользователя немедленно и отзывчиво, минимизируя задержки и сбои. Приложение должно быть оптимизировано для обеспечения высокой производительности и быстрого отклика интерфейса;

– гибкость и настраиваемость – GUI должен предоставлять пользователю возможность настраивать интерфейс в соответствии с индивидуальными предпочте-

ниями и потребностями. Пользователь должен иметь возможность настроить параметры отображения, расположение элементов управления или выбрать режимы работы, чтобы соответствовать своим потребностям;

– доступность – интерфейс должен быть доступным для пользователей с ограниченными возможностями, предоставляя возможность настройки размеров шрифтов, использование скринридеров и поддержку альтернативных методов ввода;

– поддержка различных устройств и платформ – GUI должен быть разработан с учетом мобильных устройств, планшетов и настольных компьютеров, чтобы обеспечить удобство использования на разных платформах. Дизайн должен быть адаптивным и реагировать на разные размеры экранов и разрешения;

– надежность и безопасность – интерфейс должен быть надежным и безопасным, предотвращая неправильные действия и обеспечивая защиту данных пользователя. Важно предусмотреть проверки ввода, подтверждения действий и обеспечение безопасности взаимодействия с приложением;

– Документация и поддержка – приложение должно быть сопровождено хорошо структурированной и понятной документацией, объясняющей функциональность и использование интерфейса.

Предоставление возможности связаться с командой поддержки или получить помощь внутри самого приложения может быть также полезным для пользователей.

Исходя из таких требований, была спроектирована главная форма приложения, имеющая визуальные инструменты для управления предусмотренным функционалом. Внешний вид главной формы ПО представлен на рисунке 10.

Так как одним из требований к программному продукту выступает наличие справки по реализованным алгоритмам, была спроектирована также форма справки, представленная на рисунке 11.

Спроектированные таким образом макеты полностью реализуют свое функциональное предназначение, при этом являясь удобными и понятными большинству пользователей.

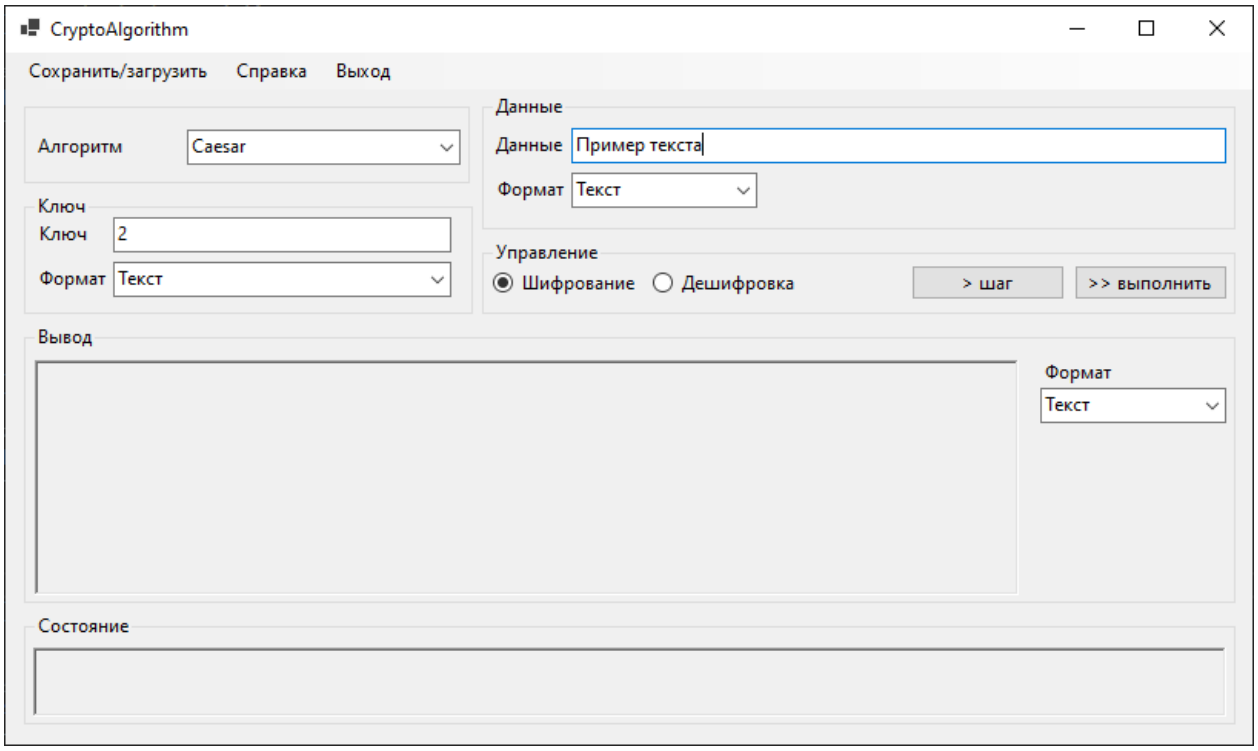

Рисунок 10 – Главная форма приложения. Макет

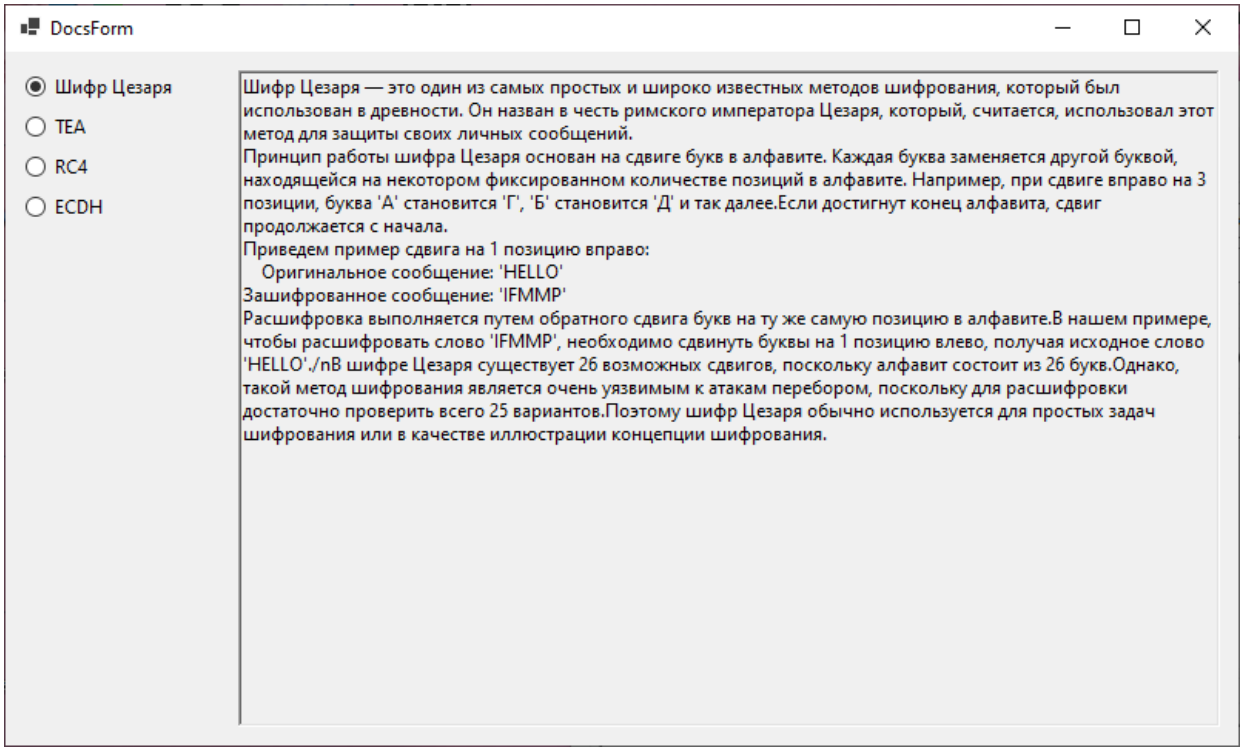

Рисунок 11 – Форма справки. Макет

3.5 Реализация программного продукта

Первым шагом в реализации данного программного продукта является реализация интерфейса ICryptoAlgorithm, от которого далее будут наследоваться классы, реализующие конкретные алгоритмы. Код такого интерфейса включает в

себя следующие методы:

– byte[] GetAlgorithmState(); Данный метод возвращает состояние алгоритма;

– string Encrypt(); Данный метод возвращает результат шифрования;

– string Decrypt(); Данный метод возвращает результат дешифрования;

– void PerformEncryptionStep(); Данный метод выполняет шаг шифрования.

Следующим шагом является реализация классов RC4CryptoAlgorithm, CaesarCipherAlgorithm, TinyEncryptionAlgorithm, ECDHAlgorithm. Реализованные классы согласно принципам ООП используются как наследники интерфейса ICryptoAlgorithm, что позволяет применить динамическое связывание.

3.5.1 Реализация шифра Цезаря

Принцип работы шифра Цезаря основан на сдвиге букв в алфавите. Каждая буква заменяется другой буквой, находящейся на некотором фиксированном количестве позиций в алфавите. Например, при сдвиге вправо на 3 позиции, буква «А» становится «Г», «Б» становится «Д» и так далее. Если достигнут конец алфавита, сдвиг продолжается с начала.

Приведем пример сдвига на 1 позицию вправо:

– оригинальное сообщение: «HELLO»;

– зашифрованное сообщение: «IFMMP».

Расшифровка выполняется путем обратного сдвига букв на ту же самую позицию в алфавите. В нашем примере, чтобы расшифровать слово «IFMMP», необходимо сдвинуть буквы на 1 позицию влево, получая исходное слово «HELLO».

3.5.2 Реализация Tiny Encryption Algorithm

– инициализация ключа. Алгоритм TEA требует наличия 128-битного ключа шифрования. Ключ инициализируется перед началом шифрования и хранится в надежном месте;

– разбиение данных. Данные, которые требуется зашифровать, разбиваются на блоки фиксированного размера. Размер блока в алгоритме TEA составляет 64 бита;

– инициализация блока данных. Каждый блок данных инициализируется перед началом шифрования. Инициализация может осуществляться путем присвое-

ния начальных значений или с использованием предыдущих блоков данных в цепочке;

– раунды шифрования. TEA состоит из множества раундов шифрования, каждый из которых применяет серию преобразований к блоку данных. Обычно используется 32 раунда, но количество раундов может варьироваться в зависимости от требуемого уровня безопасности и производительности;

– преобразование блока данных. В каждом раунде блок данных проходит через серию преобразований, которые включают в себя операции побитового сложения, побитового сдвига, побитового исключающего ИЛИ и операцию циклического сдвига;

– завершение шифрования. После завершения всех раундов шифрования получается зашифрованный блок данных. Если есть еще блоки данных для шифрования, процесс начинается снова с шага 3 для следующего блока. Если все блоки обработаны, шифрование завершается;

– расшифрование. Для расшифровки данных с использованием алгоритма TEA применяются аналогичные шаги, но в обратном порядке. Зашифрованные данные проходят через раунды расшифрования, включающие обратные преобразования, чтобы восстановить исходные данные.

3.5.3 Реализация RC4

Шаги реализации:

– инициализация ключа и массива S. Алгоритм RC4 требует наличия переменного ключа шифрования, который инициализируется перед началом шифрования. Также инициализируется массив S длиной 256 элементов, который используется для генерации псевдослучайной последовательности;

– инициализация массива S с использованием ключа. Массив S инициализируется значениями от 0 до 255 в порядке возрастания. Затем осуществляется перестановка значений массива S на основе ключа, чтобы создать начальное состояние алгоритма;

– генерация псевдослучайной последовательности. На основе инициализированного массива S генерируется псевдослучайная последовательность байтов, ко-

торая будет использоваться для шифрования данных. Это достигается путем выполняемого алгоритма псевдослучайного генератора;

– инициализация счетчиков. Инициализируются счетчики i и j для управления процессом генерации псевдослучайной последовательности. Обычно они устанавливаются в 0 перед началом шифрования;

– перестановка значений массива S. При каждой итерации генерации псевдослучайной последовательности значения массива S переставляются с использованием алгоритма обмена значений;

– генерация ключевого байта. Получившаяся переставленная последовательность значений массива S используется для генерации одного ключевого байта, который затем применяется к исходным данным с использованием операции побитового исключающего ИЛИ.

– шифрование данных. Полученный ключевой байт используется для шифрования данных путем применения операции побитового исключающего ИЛИ между ключевым байтом и соответствующим байтом исходных данных. Этот шаг повторяется для каждого байта исходных данных.

3.5.4 Реализация алгоритма Диффи-Хеллмана на эллиптических кривых

Шаги реализации:

– генерация параметров эллиптической кривой. В начале процесса участники генерируют параметры эллиптической кривой, такие как поле, коэффициенты и базовая точка. Эти параметры выбираются заранее и согласовываются между участниками;

– выбор закрытого ключа. Каждый участник выбирает свой собственный закрытый ключ, который представляет собой случайное число, обычно выбираемое из определенного диапазона. Закрытый ключ должен быть конфиденциальным и не раскрываться другим участникам.

– вычисление открытого ключа. На основе закрытого ключа каждый участник вычисляет свой открытый ключ путем умножения базовой точки эллиптической кривой на закрытый ключ. Результатом является точка на эллиптической кривой, которая становится открытым ключом данного участника;

– обмен открытыми ключами. Участники обмениваются своими открытыми ключами. Таким образом, каждый участник получает открытый ключ другого участника;

– вычисление общего секретного ключа.

Каждый участник вычисляет общий секретный ключ, используя свой закрытый ключ и полученный открытый ключ другого участника. Для этого производится умножение полученного открытого ключа на свой закрытый ключ. Результатом является точка на эллиптической кривой, которая становится общим секретным ключом для дальнейшего шифрования.

3.6 Итоги реализации

При запуске файла CryptoAlg.exe пользователя встречает главное окно программы (рисунок 12).

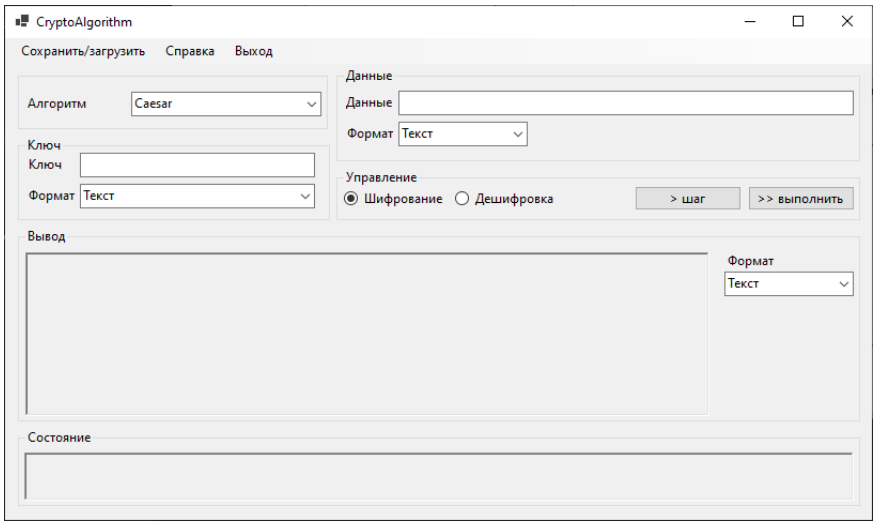

Рисунок 12 – Главное окно программы

На данном экране пользователю доступен выбор криптографического алгоритма, поля для задания входных данных алгоритма и ключа, управление процессом (шифрование/дешифрование), задание формата данных, а также кнопки управления «шаг» и «выполнить».

В верхней панели доступны кнопки «Справка» и «Выход». При нажатии на кнопку «Справка» открывается окно, содержащее справку по криптографическим алгоритмам (рисунок 13).

Переключение между справочными материалами происходит путем выбора необходимого алгоритма в меню слева.

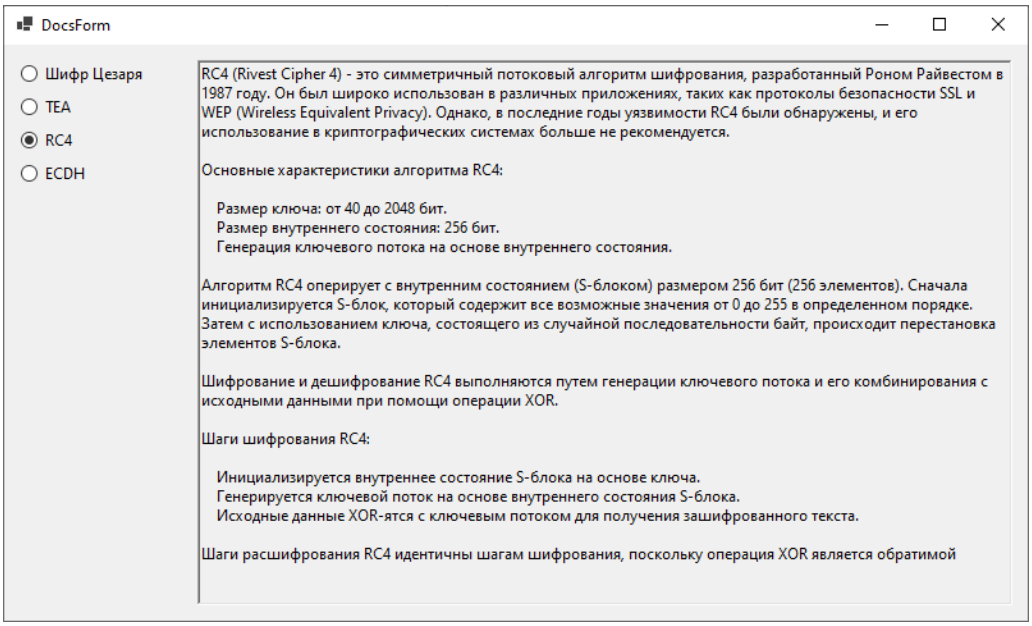

Рисунок 13 – Окно справки

После ввода пользователем исходных данных выбранного алгоритма и нажатия на кнопку «Выполнить» происходит выполнение выбранного алгоритма с заданными параметрами. После выполнения алгоритма результат шифрования/дешифровки отображается в текстовом поле «Вывод», а в месте вызова программы сохраняется файл с расширением .pdf, содержащий схему данного алгоритма (рисунки 14, 15).

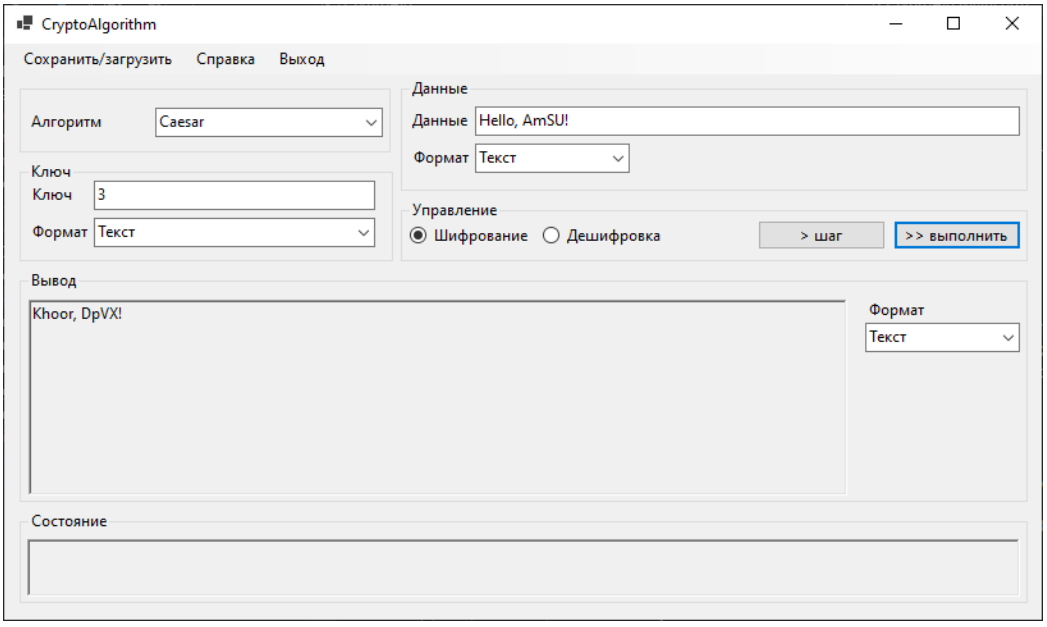

Рисунок 14 – Шифрование шифром Цезаря

Такие же схемы строятся и для иных алгоритмов. Например, на рисунке 15 приведена построенная схема алгоритма для RC4.

Уникальным в этом смысле является алгоритм ECDH. Так как более эффективным способом визуализации алгоритмов на эллиптических кривых является построение этой самой кривой, именно такой способ и выбран для ее представления (рисунок 16).

При нажатии на кнопку «Выполнить» при выбранном алгоритме ECDH происходит построение эллиптической кривой с выбранными параметрами. Визуализация эллиптической кривой открывается в отдельном окне, которое имеет функционал перетаскивания, изменения масштаба и просмотра координат точки. Такой способ взаимодействия пользователя с графиком функции представляется наиболее удачным, интерактивным и удобным.

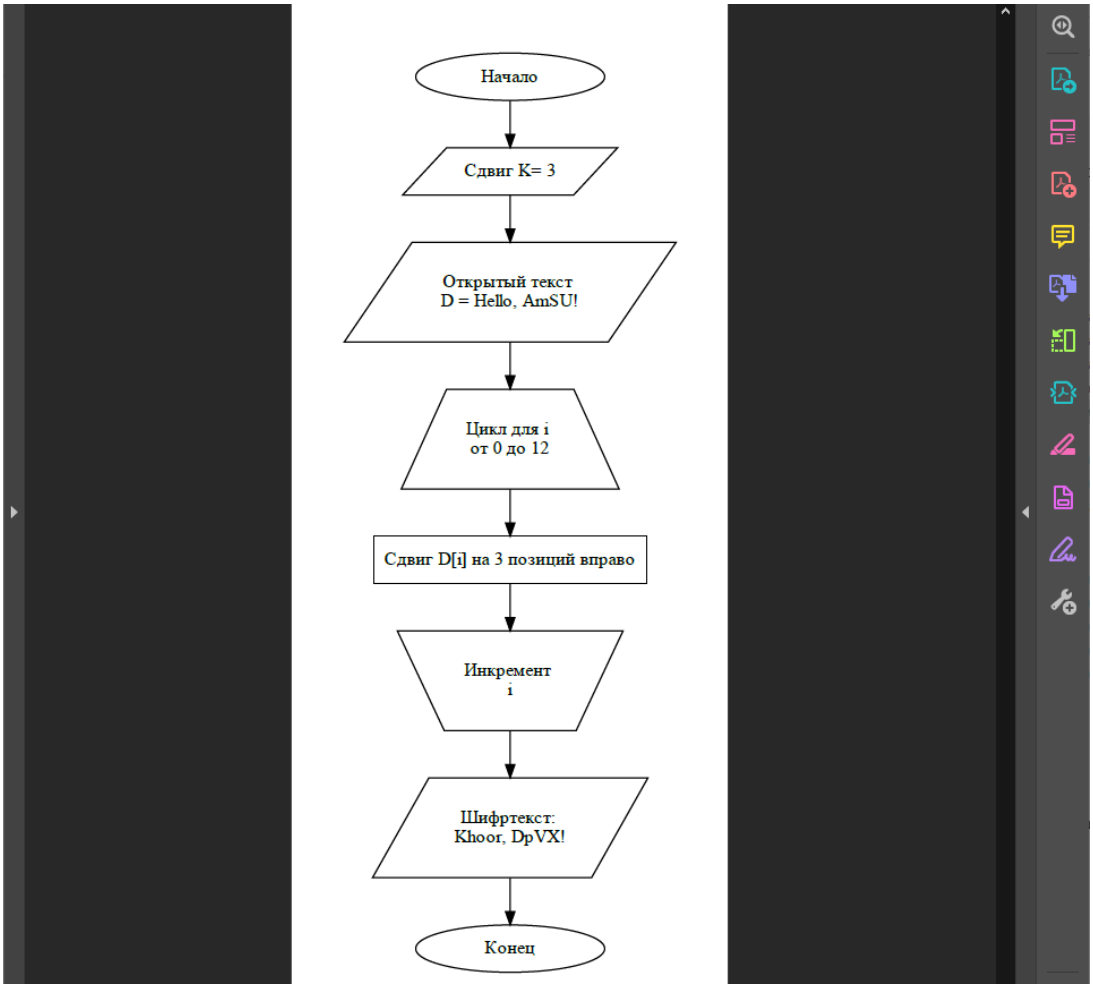

Рисунок 15 – Построенная схема алгоритма

Таким образом, был реализован программный продукт, реализующий функции построения схем алгоритма для алгоритмов ECDH, TEA, RC4, Шифр Цезаря. Также для шифра Цезаря реализована визуализация процесса шифрования с возможностью интерактивного взаимодействия.

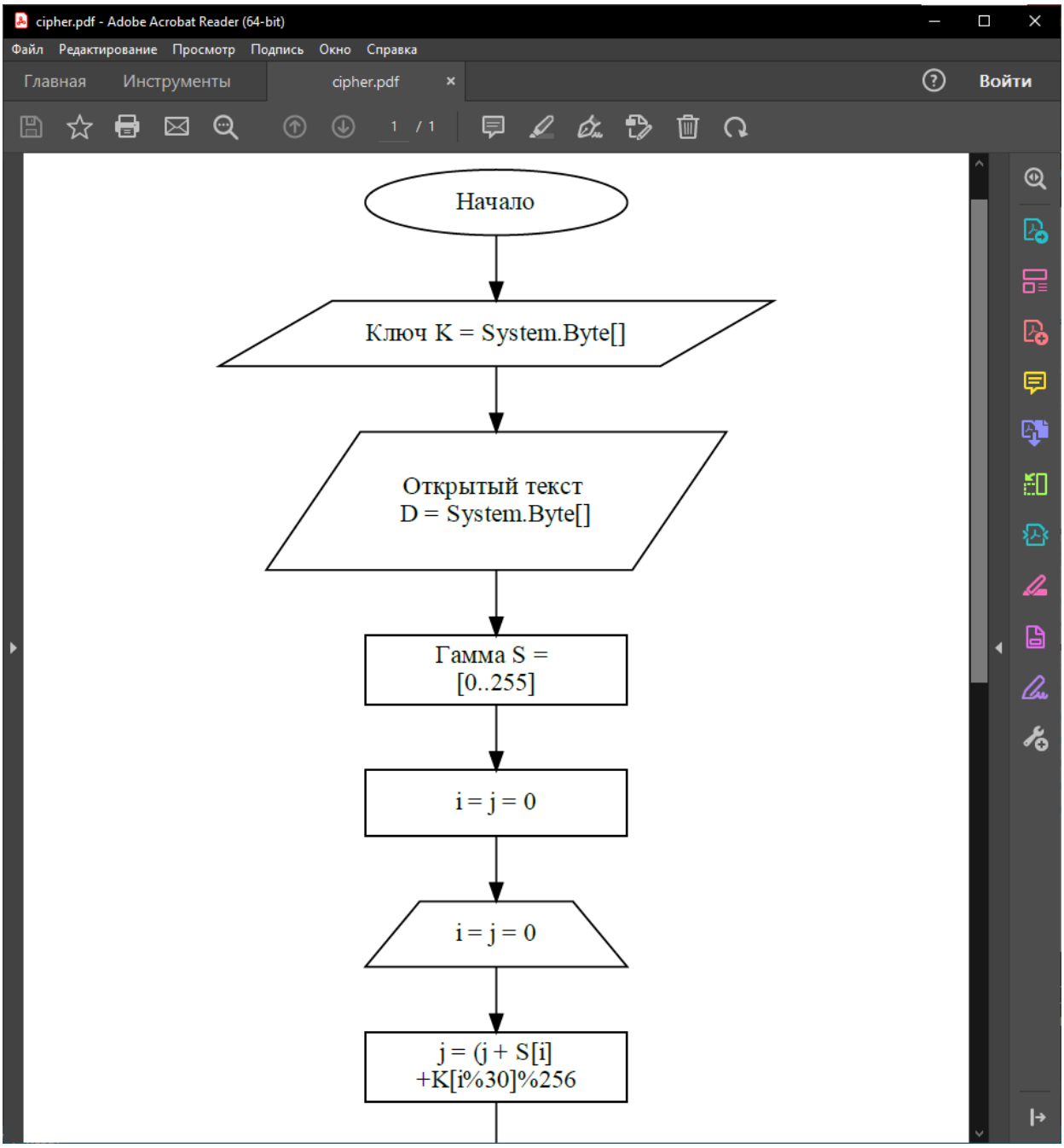

Рисунок 16 – Схема RC4. Верхняя часть

Для ECDH реализовано построение интерактивного графика выбранной эллиптической кривой, имеющего функционал отображения координат курсора, изменения масштаба графика и перемещения по нему.

Также для всех вышеперечисленных алгоритмов реализована возможность шифрования и дешифрования задаваемых пользователем данных, а также отображение справки на русском языке по каждому из них. Наличие такой справки позволяет обучающемуся получить базовую информацию об алгоритме, не прибегая к сторонним ресурсам.

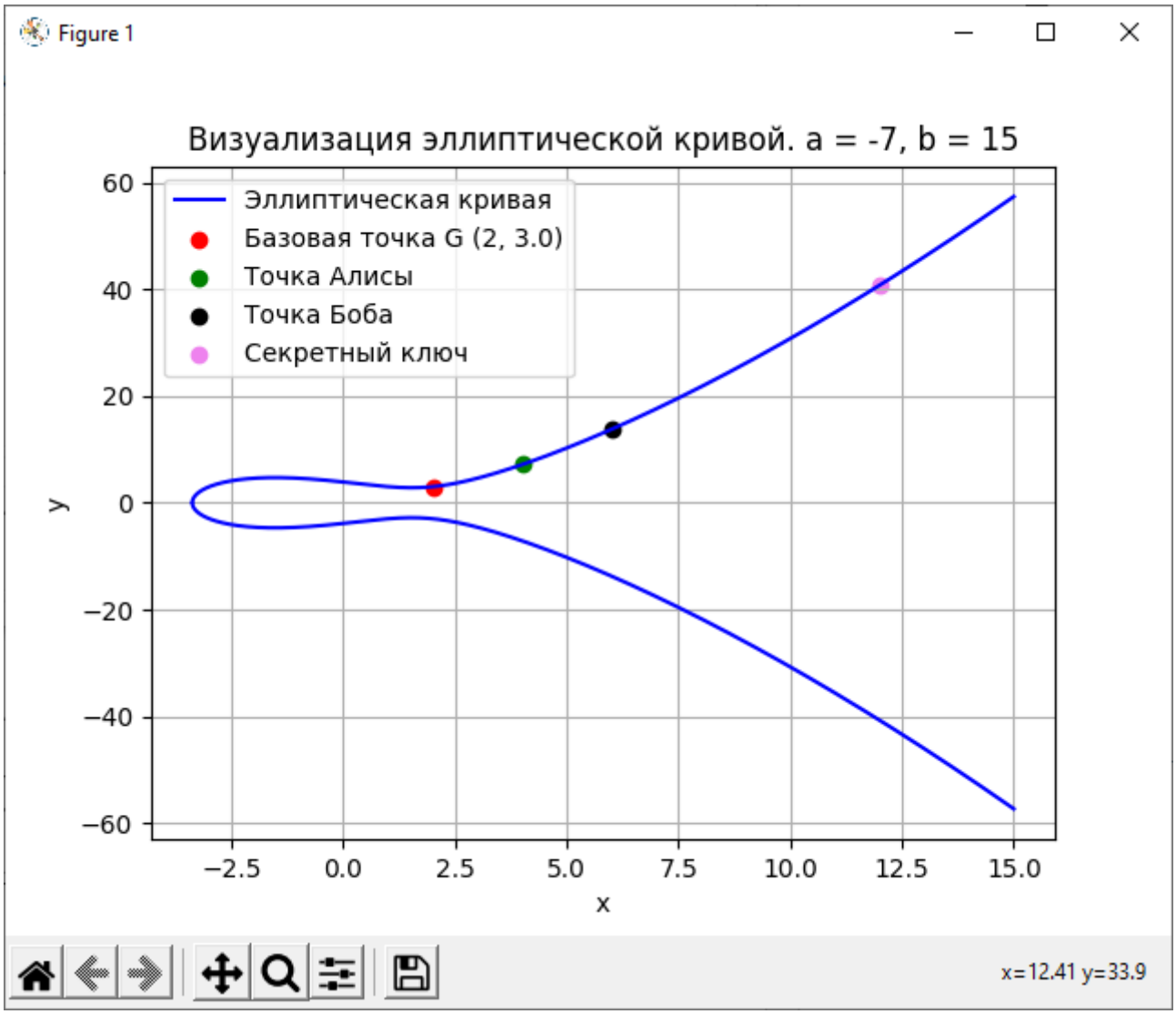

Рисунок 17 – График эллиптической кривой

Все вышеперечисленные функции в совокупности позволяют проще изучать реализованные в программном продукте криптографические алгоритмы.

4 БЕЗОПАСНОСТЬ И ЭКОЛОГИЧНОСТЬ

Для обеспечения полноценного функционирования оконного приложения и технической поддержки системы требуется наличие рабочих мест и специально оборудованного помещения, соответствующих нормативным документам и стандартам. Кроме того, необходимо принять меры, направленные на сохранение здоровья сотрудников при работе с приложением на компьютере.

4.1 Безопасность

4.1.1 Опасности и вред ПЭВМ на рабочем месте пользователя

Согласно стандарту ГОСТ 12.0.003-2015, выполнение работы на персональном компьютере неразрывно связано с потенциальными опасностями и неблагоприятными факторами, которые могут негативно сказываться на здоровье. Среди таких факторов следует отметить электростатические поля, риск поражения электрическим током, электромагнитное излучение, экстремальные температуры окружающей среды, монотонность рабочего процесса, высокий уровень шума, ограниченное естественное освещение и другие.

Существуют специально разработанные требования, служащие для предотвращения и/или уменьшения воздействия указанных вредных факторов на пользователей персональных компьютеров. Такие требования относятся к организации рабочего места, освещению, уровню шума и общей организации помещений. Кроме того, предоставляются рекомендации для пользователей ПЭВМ, направленные на обеспечение безопасных условий труда и снижение негативного воздействия на здоровье.

4.1.2 Организация рабочего места

Следующие требования критичны при обустройстве рабочего места с ПЭВМ:

– поверхность сиденья должна иметь ширину и глубину не менее 400 мм, а также закругленный передний край. Высота сиденья должна регулироваться в пределах от 400 до 550 мм, а также иметь возможность наклона вперед до 15 градусов и назад до 5 градусов. Угол наклона спинки в вертикальной плоскости
должен быть ± 30 градусов;

– рабочее место пользователя ПЭВМ должно быть оборудовано подставкой для ног с шириной не менее 300 мм, глубиной не менее 400 мм, регулировкой по высоте в пределах 150 мм и возможностью наклона опорной поверхности до 20 градусов;

– рабочий стол должен обеспечивать достаточное пространство для ног на уровне колен, а именно высотой не менее 600 мм, шириной не менее 500 мм и глубиной не менее 450 мм. На уровне вытянутых ног, пространство для ног должно быть не менее 650 мм;

– регулировка высоты рабочей поверхности стола, используемого для работы с ПЭВМ, необходима для взрослых пользователей и должна находиться в пределах от 680 до 800 мм. В случае, если регулировка невозможна, высота должна составлять 725 мм;

– подлокотники должны иметь длину не менее 250 мм и ширину в пределах от 50 до 70 мм. Они должны регулироваться по высоте над сиденьем в пределах (230±30) мм и иметь внутреннее расстояние между ними от 350 до 500 мм;

– клавиатура должна размещаться на поверхности стола на расстоянии 100 мм от пользователя или на специальной рабочей поверхности, отделенной от основного столешницы на расстоянии 300 мм. Эти требования гарантируют создание комфортных и безопасных условий для работы пользователей ПЭВМ.

Каждое рабочее место с компьютером (ПЭВМ), где отсутствует периферийное оборудование и установлен ЖК-монитор, должно иметь площадь не менее 4,5 м^2. В случае отсутствия данных условий, для каждого рабочего места с ПЭВМ требуется площадь не менее 6 м^2. Боковые стенки мониторов не должны находиться ближе 1,2 метра друг к другу, а освещение должно поступать со стороны левой или правой стороны, а не через окна.

Для самостоятельной работы по обслуживанию сервера должны быть выполнены следующие условия [1]:

– перед началом работы работники должны пройти вводный инструктаж;

– работники, выполняющие работы по обслуживанию сервера, должны быть не моложе 18 лет;

– перед поступлением на работу требуется прохождение предварительного медицинского осмотра, чтобы определить соответствие здоровья работника выполняемым задачам;

– в течение трудовой деятельности регулярно проводятся медицинские осмотры для наблюдения за состоянием здоровья работника и своевременного выявления ранних признаков негативного влияния вредных производственных факторов на здоровье работников;

– работники должны пройти обучение безопасным приемам и методам выполнения работ, а также оказанию первой помощи при несчастных случаях;

– работники должны пройти обучение и проверку знаний по вопросам электробезопасности, соответствующим второй группе. Эти требования обеспечат безопасные условия работы с ПЭВМ и обслуживания сервера.

4.1.3 Освещение

Одно из важнейших требования для помещений с ПЭВМ – освещение. Правильное освещение способствует повышению производительности труда, уменьшает нагрузку на глаза. С другой стороны, плохое освещение может привести к быстрой утомляемости, снижению концентрации при работе за компьютером, ослеплению и раздражению от излишней яркости. Виды освещения включают естественное, искусственное, совмещенное и аварийное.

Естественное освещение обязательно должно присутствовать во всех помещениях, где работает персонал. Оно может быть боковым, верхним или комбинированным в зависимости от расположения по отношению к рабочим местам пользователей.

Применение искусственного освещения предпочтительно в периоды недостаточной естественной освещенности, особенно в темное время суток. Однако, важно, чтобы искусственное освещение обеспечивало равномерное освещение по всей рабочей области. Когда размещение источников света учитывает конкретное расположение рабочих мест, используется локализованное искусственное освещение. Локализованное искусственное освещение является предпочтительным в случаях, когда требуется сфокусированное освещение конкретных рабочих зон или за-

дач. Для достижения этого эффекта применяются различные типы источников света, такие как настольные лампы, направленные светильники или подсветка специфических рабочих поверхностей.

Совмещенное освещение комбинирует преимущества естественного и искусственного освещения, обеспечивая оптимальные условия для работы с ПЭВМ. В этом случае естественное освещение используется в сочетании с искусственным, чтобы достичь необходимого уровня освещенности и улучшить визуальный комфорт пользователей. К примеру, при наличии окон в помещении с ПЭВМ можно использовать жалюзи или шторы, чтобы регулировать проникновение естественного света в зависимости от потребностей.

Особое внимание следует уделять также выбору подходящих параметров освещения, таких как яркость, контрастность и цветовая температура. Яркость должна быть достаточной для обеспечения четкого восприятия информации на экране компьютера, но при этом не должна создавать ослепляющего эффекта. Контрастность между экраном и окружающей средой также играет важную роль в качестве визуального комфорта и снижении утомляемости глаз. Оптимальная цветовая температура освещения может способствовать повышению концентрации и улучшению настроения пользователей.

Совмещенное освещение является необходимым в случаях, когда естественное освещение недостаточно для обеспечения требуемого уровня освещенности. Такой тип освещения широко применяется в работах, требующих высокой точности и детализации.

Также, необходимо учесть требования к аварийному освещению, которое используется в случае отключения основного освещения. В соответствии с СанПиН 1.2.3685 – "Гигиенические нормы к требованиям к обеспечению безопасности и (или) безвредности для человека факторов среды обитания", четко определены требования к освещению на рабочих местах с ПЭВМ.

Таким образом, важно соблюдать все установленные нормы и требования по освещению, чтобы обеспечить безопасные и комфортные условия работы пользователей ПЭВМ.

Согласно указанным требованиям, равномерность освещенности в основных помещениях должна быть не менее 0,6 (0,7 для учебных кабинетов черчения и рисования, 0,5 на ледовых аренах и 0,7 для спортивных залов в физкультурно-оздоровительных организациях). Во вспомогательных помещениях минимальный уровень равномерности освещенности составляет 0,4. Комбинированные системы общего и местного освещения применяются в учебных помещениях и читальных залах для обучающихся с нарушениями зрения.

Суммарный уровень освещенности от общего и местного освещения должен соответствовать следующим значениям: 1000 лк для обучающихся с высокой степенью осложненной близорукости и высокой степени дальнозоркости, от 1000 до 1500 лк для обучающихся с поражением сетчатки и зрительного нерва без светобоязни, и не превышать 500 лк для обучающихся со светобоязнью.

Для достижения требуемого освещения рекомендуется использовать люминесцентные лампы с высокой световой отдачей и спектральным составом, близким к естественному. Освещенность на рабочей поверхности стола должна находиться в диапазоне от 300 до 500 лк. Внутренняя отделка помещений с ПЭВМ должна использовать материалы с диффузно-отражающими свойствами и следующими коэффициентами отражения: потолок – от 0,7 до 0,8, стены – от 0,5 до 0,6, пол – от 0,3 до 0,5.

Таким образом, правильное освещение в помещениях с ПЭВМ играет критическую роль в обеспечении комфортных и безопасных условий работы для пользователей. Оно должно быть разнообразным, адаптированным к конкретным требованиям рабочих мест, и обеспечивать оптимальные параметры освещения, с учетом физиологических и психологических потребностей людей, что способствует повышению эффективности и благополучия в работе с ПЭВМ.

4.1.4 Шум

Один из значительных опасных и вредных факторов, связанных с работой за ПЭВМ – это повышенный уровень шума. В производственных помещениях, где осуществляются основные или вспомогательные работы с использованием ПЭВМ, уровень шума на рабочих местах не должен превышать предельно допустимых зна-

чений, установленных для этих видов работ в соответствии с санитарно-эпидемиологическими нормативами.

Повышенный уровень шума, являющийся одним из значительных опасных и вредных факторов, связанных с работой за персональными электронно-вычислительными машинами (ПЭВМ), представляет серьезную проблему для здоровья и благополучия работников. В производственных помещениях, где выполняются основные или вспомогательные работы с использованием ПЭВМ, необходимо стремиться к обеспечению уровня шума на рабочих местах, не превышающего предельно допустимые значения, установленные для данных видов работ согласно санитарно-эпидемиологическим нормативам.

Шум, как акустический феномен, оказывает негативное воздействие на организм человека. Экспозиция продолжительному шуму может вызывать широкий спектр неблагоприятных эффектов, включая физиологические, психологические и поведенческие последствия. Среди физиологических эффектов можно выделить повышенный уровень стресса, снижение концентрации внимания, нарушение сна и отдыха, а также повышенный риск развития сердечно-сосудистых заболеваний. Психологические последствия включают ухудшение настроения, раздражительность, пониженное самочувствие и плохое самочувствие в целом. Кроме того, повышенный уровень шума может снизить производительность и качество работы, а также способствовать возникновению ошибок и несчастных случаев.

Для смягчения негативного влияния шума на работников, необходимо предпринять соответствующие меры по контролю и управлению этим фактором. Во-первых, рекомендуется проводить оценку уровня шума на рабочих местах с использованием специальных измерительных приборов. Это позволит определить степень превышения уровня шума над допустимыми нормами и идентифицировать основные источники шума в рабочей среде.

Во-вторых, необходимо применять технические и организационные меры для снижения уровня шума на рабочих местах. К таким мерам могут относиться использование звукоизолирующих материалов, установка звукопоглощающих панелей, регулярное техническое обслуживание и апгрейд оборудования, а также оп-

тимизация процессов работы для уменьшения шумовых нагрузок.

В-третьих, рекомендуется применять индивидуальные средства защиты от шума, такие как наушники или наушники с активным шумоподавлением, чтобы минимизировать воздействие шума на слух работников.

Наконец, следует обеспечить регулярное мониторинг и контроль уровня шума на рабочих местах, чтобы убедиться в эффективности принятых мер и в случае необходимости корректировать стратегию по управлению шумом.

В целом, снижение уровня шума на рабочих местах, где осуществляются работы с использованием ПЭВМ, является важным аспектом обеспечения здоровых и безопасных условий труда. Это требует совместных усилий со стороны работодателей, специалистов в области охраны труда и самого персонала, чтобы минимизировать негативное воздействие шума и обеспечить комфортные условия работы при использовании ПЭВМ.

На рабочих местах при использовании ПЭВМ в жилых, общественных зданиях и на территории жилой застройки также существуют нормативы для уровня шума. В помещении, где человек работает за ПЭВМ, уровень шума не должен превышать 50 децибел [2].

4.1.5 Микроклимат

В производственных помещениях микроклимат охватывает множество факторов, включая температуру, влажность, тепловое излучение и другие, которые существенно влияют на комфорт и работоспособность работников. Следовательно, обеспечение подходящих условий микроклимата на рабочем месте, в пределах нормативов, становится неотъемлемой задачей в области охраны труда. Одним из существенных источников повышения температуры человека и помещения в целом являются персональные компьютеры (ПЭВМ), используемые на рабочих местах. Повышение температуры может негативно сказываться на работоспособности и производительности сотрудников. Следовательно, поддержание требуемой температуры в помещении имеет важное значение для обеспечения безопасности и комфорта при работе с ПЭВМ.

Для регулирования микроклимата в помещении широко используются системы вентиляции, которые обеспечивают эффективный обмен воздуха и подачу

свежего воздуха извне. Часто используется естественная вентиляция, чтобы достичь наиболее комфортных условий. Однако весной и летом может потребоваться применение систем кондиционирования воздуха для полного контроля микроклиматических параметров и создания комфортных условий труда.

Системы кондиционирования помогают поддерживать постоянную температуру, влажность и осуществлять очистку от вредных веществ в помещении. Они также способствуют устранению проблемы задержки углекислого газа в воздухе.

В холодное время года система отопления играет важную роль в обеспечении постоянной и равномерной температуры воздуха в рабочих помещениях. При проектировании такой системы учитываются потери тепла через стены здания и нагрев воздуха, проникающего в помещение. Существует несколько типов систем отопления, включая водяные, паровые, воздушные и комбинированные.

Наиболее эффективными являются системы водяного отопления, которые позволяют регулировать температуру в широком диапазоне и широко применяются в помещениях, где используются ПЭВМ. Рекомендуемые температурные диапазоны в холодное время года составляют (22–24) °С, а в теплое время года – (20–25) °С.

Кроме поддержания комфортных условий труда, системы отопления и кондиционирования воздуха также играют значительную роль в обеспечении оптимальных условий для сохранения материалов и оборудования. Например, в помещениях, где хранятся чувствительные к изменениям температуры или влажности предметы, необходимо поддерживать устойчивые климатические условия.

Системы вентиляции, кондиционирования и отопления основываются на принципах теплообмена и циркуляции воздуха. В системах кондиционирования, обычно используются холодильные установки, которые охлаждают воздух и удаляют из него избыточную влагу и загрязнения. Такие системы часто оснащены фильтрами, которые позволяют удалить частицы пыли, аллергены и другие вредные вещества, повышая качество воздуха в помещении.

Системы отопления, с другой стороны, используют различные источники энергии для нагрева воздуха или нагревательных элементов, передавая тепло в по-

мещение. Водяные системы отопления особенно эффективны и широко применяются. Они основаны на циркуляции горячей воды через радиаторы или тепловые полы, что позволяет равномерно распределить тепло и контролировать температуру в разных зонах помещения.

При проектировании систем отопления и кондиционирования воздуха необходимо учитывать такие факторы, как размер помещения, уровень изоляции, теплопотери через стены и окна, а также потребности пользователей. Точное подбор технических характеристик и параметров системы позволяет достичь оптимального режима работы и энергоэффективности.

4.1.6 Требования к графическому интерфейсу

При разработке системы, в соответствии с ГОСТ Р ИСО 9241-161-2016, были использованы компоненты, включающие различные методы взаимодействия с графическими элементами пользовательского интерфейса. Эти методы ввода включают:

– указание с помощью компьютерной мыши, ручки, распознавания жестов,

– ввод данных с клавиатуры;

– отслеживания положения глазного яблока;

– речевой ввод с использованием голосовых команд, распознавания голоса.

Пользователю предоставляется возможность применять все доступные методы ввода, если интерактивная система поддерживает несколько таких методов. В данной работе был выбран классический подход, использующий клавиатуру и мышь в качестве основных средств взаимодействия.

Оконное приложение, разработанное на основе классических принципов разработки пользовательского интерфейса, также включает элементы и принципы взаимодействия, описанные в ГОСТ Р ИСО 9241-161-2016. Все методы взаимодействия с системой соответствуют современным принципам пользовательского интерфейса и в основном основываются на широкодоступных средствах ввода.

4.2 Чрезвычайные ситуации

4.2.1 Аварийные ситуации

В ходе работы могут возникать различные аварийные ситуации, включающие

следующие возможные сценарии: обрыв проводов электропитания, проблемы с заземлением, повреждение электрооборудования, повреждение инженерных коммуникаций, а также повреждение конструктивных элементов здания или помещения.

В случае возникновения любой из описанных чрезвычайных ситуаций или при ухудшении самочувствия необходимо незамедлительно предпринять следующие меры:

– прекратить выполнение работ и, при необходимости, отключить электропитание;

– при наличии пострадавших предоставить первую помощь;

– обеспечить открытие аварийных выходов и эвакуацию персонала;

– доложить о принятых мерах руководителю работ и действовать в соответствии с его указаниями;

– сообщить оперативному дежурному о произошедшем.

В случае, если сотрудник находится непосредственно рядом с местом происшествия или несчастного случая, он должен оказать предварительную медицинскую помощь и сообщить об этом оперативному дежурному или начальнику отдела. Если человек подвергся воздействию электрического тока, необходимо немедленно отключить источник электропитания и освободить его от контакта с током. Если невозможно отключить электроустановку, следует предпринять другие меры для освобождения пострадавшего, например, использовать изолирующие средства, такие как канат, палка или другие непроводящие электрический ток предметы. Дополнительно, пострадавшего можно освободить, потянув за сухую одежду, избегая контакта с металлическими предметами или непокрытыми частями тела. Для защиты рук оказывающего помощь рекомендуется использовать изолирующие перчатки, а в их отсутствие можно обмотать руки сухой одеждой.

4.2.2 Меры пожарной безопасности на рабочих местах

Для обеспечения безопасности при установке технологического и другого оборудования необходимо предусмотреть свободный доступ к путям эвакуации и аварийным выходам. При установке персонального компьютера и монитора следует выбрать надежную опору, такую как тумбочка, подставка или кронштейн, чтобы предотвратить их падение. Следует избегать установки ПЭВМ в следующих

местах:

– менее чем в 1 метре от электронагревательных приборов и горючих предметов, таких как тюли, занавески, гардины, шторы, декоративные украшения, новогодние ёлки и т.д.;

– в нишах мебельных "стенок", тумбочках и т.п.;

– менее чем в 0,7 метрах от проходов, путей передвижения и эвакуации людей.

Перед началом эксплуатации персонального компьютера необходимо выполнить следующие шаги:

– проверить место установки компьютера и монитора на соответствие требованиям безопасности, описанным выше;

– осмотреть компьютер, монитор, сетевой кабель и вилку на наличие повреждений;

– при повреждении корпуса, сетевого кабеля, вилки или задней крышки запрещается использовать компьютер;

– удалить горючие предметы (салфетки, накидки, книги, газеты, декоративные украшения и т.д.) и емкости с жидкостью (вазы с живыми цветами) с и около компьютера и монитора;

– убедиться, что вентиляционные отверстия на задней крышке компьютера и монитора не заблокированы предметами;

– убедиться в наличии огнетушителя или противопожарной ткани рядом с компьютером. Соблюдение этих мер безопасности при работе с ПЭВМ поможет снизить риск возникновения пожара.

4.3 Экологичность

Персональные электронно-вычислительные машины (ПЭВМ) содержат разнообразные компоненты, которые содержат токсичные вещества, представляющие опасность для человека и окружающей среды. Эти вещества включают:

– ртуть, содержащаяся в подсветке ЖК-мониторов, является вредным для мозга и нервной системы;

– щелочи, содержащиеся в щелочных аккумуляторах источников бесперебойного питания, могут прожигать слизистые оболочки и кожу;

– никель и цинк, находящиеся в материнских платах и батареях питания для ноутбуков, могут вызывать дерматит;

– поливинилхлорид, содержащийся в кабелях, используемых для подключения к электронным устройствам, может разрушать нервную систему и вызывать раковые заболевания.

В связи с этим, утилизация ПЭВМ требует специального комплексного подхода. Этот подход включает сортировку металлических и неметаллических компонентов, при этом металлические части подлежат переплавке для последующего использования, а неметаллические компоненты ПЭВМ утилизируются специальным образом [3].

Металлические части ПЭВМ, такие как провода, контакты и различные крепежные элементы, обладают высокой стоимостью и ценными свойствами, такими как электропроводность и механическая прочность. Поэтому рациональным решением является переплавка металлических компонентов с последующим использованием полученных материалов в новом производстве. Этот процесс не только позволяет экономить ресурсы, но и снижает негативное воздействие на окружающую среду, связанное с добычей и производством первичных металлов.

С другой стороны, неметаллические компоненты ПЭВМ, такие как пластиковые оболочки, платы с электронными компонентами и различные изоляционные материалы, нуждаются в специальном подходе к утилизации. Неметаллические компоненты могут содержать токсичные вещества, такие как свинец, кадмий и бромированные органические соединения, которые могут нанести вред окружающей среде и здоровью человека, если не обрабатываются правильным образом.

Для утилизации неметаллических компонентов ПЭВМ используется специальный процесс. Вначале происходит сортировка этих компонентов по типу материала и степени их опасности. Затем применяются различные методы, включая механическое разрушение, термическую обработку и химическую переработку, чтобы извлечь ценные материалы и минимизировать негативное воздействие на окружающую среду. Например, пластиковые материалы могут быть переработаны вторично для производства новых изделий, а токсичные вещества могут быть уда-

лены или обработаны специальными методами, чтобы предотвратить загрязнение окружающей среды.

Таким образом, комплексный подход к утилизации ПЭВМ, включающий сортировку металлических и неметаллических компонентов, а также их специализированную обработку, является необходимым для эффективного и экологически безопасного управления отходами от ПЭВМ. Это позволяет снизить потребность в первичных ресурсах, сократить выбросы вредных веществ и продлить жизненный цикл материалов, содействуя устойчивому развитию и охране окружающей среды.

В настоящее время в различных отраслях промышленности активно разрабатываются и внедряются малоотходные технологии. Однако полный переход ведущих промышленных отраслей на безотходную технологию требует решения сложных технологических, конструкторских и организационных задач.

ЗАКЛЮЧЕНИЕ

В рамках выполнения выпускной квалификационной работы на тему «Реализация приложения для визуализации криптографических алгоритмов» были успешно решены следующие задачи:

– анализ существующих криптографических алгоритмов;

– анализ похожих программных продуктов;

– формулировка требований к программному продукту;

– разработка программного продукта.

Решение данных задач позволило в полном объеме выполнить цель по реализации программного продукта, позволяющего визуализировать криптографические алгоритмы.

Выбранные для реализации криптографические алгоритмы максимально репрезентативно отражают существующие примитивы различных семейств алгоритмов, что позволяет пользователю изучить наиболее распространенные концепции, знание которых является огромным шагом на пути к пониманию более сложных алгоритмов.

Также реализованный программный продукт рассчитан на последующее сопровождение, включающее в себя, среди прочего, добавление новых алгоритмов в его функционал.

Реализованный программный продукт может быть использован в учебном процессе студентов направлений подготовки высшего и среднего специального образования, в чью учебную программу входит изучение криптографии. Использование данного продукта позволит облегчить знакомство и освоение криптографии в ходе подготовки студентов как профильных, так и не профильных направлений подготовки среднего профессионального и высшего образования, а также обучающихся на курсах повышения квалификации и переподготовки, что позволит повысить уровень ИТ-подготовки и грамотности в области кибербезопасности.

БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ

1 Пособие по безопасной работе на персональных компьютерах [Текст] / разраб. В. К. Шумилин. - М. : НЦ ЭНАС, 2005. - 28 с.

2 Кардаш, Т. А. Эргономика рабочих мест служащих и инженерно-технических работников, оснащенных ПЭВМ [Текст] : учеб. пособие / Т. А. Кардаш ; АмГУ, ИФФ. - Благовещенск : Изд-во Амур. гос. ун-та, 2002. - 60 с.

3 Шумилин, В.К. ПЭВМ. Защита пользователя [Текст] / Шумилин В.К. - М. : Охрана труда и социальное страхование, 2001. - 214с.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Албахари, Д. C# 9.0. Справочник. Полное описание/ Албахари Джозеф, Албахари Бен. — Москва: Диалектика, 2021. – 1056 с.

2 Александров, Н. Криптография: практическое руководство / Н. Александров, А. Вальдес, М. Шпилька. – Санкт-Петербург : БХВ-Петербург, 2018. – 320 с.

3 Бабаш, А. Методы криптографической защиты информации / А.В. Бабаш, Е.К. Баранова. — Москва: КноРуш, 2016. – 189 с.

4 Безопасность жизнедеятельности в химической промышленности [Электронный ресурс]: учебник / Н. И. Акинин [и др.]. – Санкт-Петербург: Лань, 2019. – Режим доступа: https://e.lanbook.com/book/116363.

5 Бессалов А.В. Эллиптические кривые в форме Эдвардса и криптография / А.В. Бессалов. – КИЕВ : «Політехніка», 2017. – 272 с.

6 Булгаков А.Б. Безопасность жизнедеятельности [Электронный ресурс]: сб. учеб.-метод. материалов для всех направлений подготовки бакалавров и специалистов / АмГУ, ИФФ; сост. А.Б. Булгаков, В.Н. Аверьянов, М. В. Гриценко. – Благовещенск: Изд-во Амур. гос. ун-та, 2017. http://irbis.amursu.ru/DigitalLibrary/AmurSU_Edition/9036.pdf

7 Васильева, И. Н. Криптографические методы защиты информации : учебник и практикум для вузов / И. Н. Васильева. — Москва : Издательство Юрайт, 2022. — 349 с. — (Высшее образование). — ISBN 978-5-534-02883-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/489919

8 Дейтел П., Дейтел Х. Язык программирования C#. Технологии .NET и платформы Microsoft. – Москва: ООО "Издательство ДМК Пресс", 2019.

9 Джоунс, У. Криптография для всех на C# / Джоунс Уильям. — Санкт-Петербург: Питер, 2019.

10 Жданов, О.Н. Эллиптические кривые: Основы теории и криптографические приложения / О.Н. Жданов, В.А. Чалкин. – Москва : Книжный дом "ЛИБРОКОМ", 2020. – 200 с. – ISBN 978-5-397-07274-8.

11 Запечников С.В. Криптографические методы защиты информации / С.В. Запечников, О.В. Казарин, А.А. Тарасов. – Москва : Юрайт, 2022. – 309 с.

12 Запечников, С. В. Криптографические методы защиты информации : учебник для вузов / С. В. Запечников, О. В. Казарин, А. А. Тарасов. — Москва : Издательство Юрайт, 2022. — 309 с. — (Высшее образование). — ISBN 978-5-534-02574-3. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https:// urait.ru/ bcode/489487

13 Лутц М Изучаем Python Том 1 / М Лутц. – СПб : Вильямс, 2019. – 832 с. – ISBN 978-5-907144-52-1

14 Мартин, К. Криптография: как защитить свои данные в цифровом пространстве / К. Мартин. – (БукТех. Книги про технологии). – Москва : Эксмо, 2023. – 368 с.

15 Мурадханов, С.Э. Разработка на языке C# приложений с графическим интерфейсом (использование Windows Forms): Учебник / С.Э. Мурадханов. – Москва : "МИСИС", 2019. – 396 с. – ISBN 978-5-907061-36-1.

16 Нестеренко А.Ю. Криптографические методы защиты информации: учебник для академического бакалавриата / А.Ю. Нестеренко, А.Б. Лось, М.И. Рожков — Москва: Юрайт, 2016.

17 Паращук, В. Н. Современные криптографические методы и протоколы в информационных системах / В. Н. Паращук, И. В. Морозов, Н. В. Сибирева. – Москва : Интернет-Университет Информационных Технологий, 2023. – 304 с.

18 Рабочая программа «Методы и средства криптографической защиты информации» для направления подготовки 10.03.01 Информационная безопасность. Амурский Государственный Университет. URL: https://cabinet.amursu.ru/uploads/sveden/edu_prog/Rab_prog_Code_Name_Date/79292/Rab_prog_10.03.01_Metody_i_sredstva_kriptograficheskoy_zaschity_informatsii_09.09.2022.pdf (дата обращения: 20.05.2023)

19 Рабочая программа «Криптографические протоколы» для специальности 10.05.01 Компьютерная безопасность. Дальневосточный Федеральный Университет. URL: https://bb.dvfu.ru/bbcswebdav/orgs/FUDOOD/22/ИМиКТ/специали-

тет/2017 г.н. 10.05.01 Математические методы защиты информации/РПУД/Б1.Б.06.06 КП.pdf (дата обращения: 20.05.2023)

20 Рабочая программа «Методы и средства криптографической защиты информации» для специальности 10.05.01 Компьютерная безопасность. Тихоокеанский Государственный Университет. URL: https://pnu.edu.ru/media/filer_public/41/3c/413c2597-7377-40c4-8f67-11035c7282f7/100501-_.rar (дата обращения: 20.05.2023)

21 Рихтер, Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C# (4-е издание) / Рихтер Джеффри. — Санкт-Петербург: Питер, 2019. – 896 с.

22 Седжвик Р. Computer Science. Основы программирования на Java, ООП, алгоритмы и структуры данных / Р. Седжвик, К. Уэйн. – Санкт-Петербург : Питер, 2018. – 1072 с.

23 Хеллман Д Стандартная библиотека Python 3. Справочник с примерами / Д Хеллман. – Москва; СПб : Вильямс, 2018. – 1376 с. – ISBN 978-5-6040043-8-8

24 Шнайер, Б. Прикладная криптография: Протоколы, алгоритмы и исходные тексты на языке С (2-е издание) / Шнайер Брюс. — Москва: Диалектика, 2022. – 1040 с.

25 Шнайер, Б. Криптография: защита информации в открытом мире / Б. Шнайер. – Москва : Вильямс, 2018. – 832 с.