Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра вычислительных машин, систем и сетей

Дисциплина: Программирование на языках высоккого уровня

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

Программа «Шифратор файлов»

Студент Черняк С.С.

Руководитель Богдан Е.В.

МИНСК 2023

**Содержание**

[1 ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ 2](#_Toc150462830)

[2. ВВЕДЕНИЕ 7](#_Toc150462831)

[3. ОБЗОР ЛИТЕРАТУРЫ 9](#_Toc150462832)

[3.1 ОБЗОР МЕТОДОВ И АЛГОРИТМОВ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ 10](#_Toc150462833)

[4 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВНИЕ 13](#_Toc150462834)

[5 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ 17](#_Toc150462835)

[5.1 Разработка схем алгоритмов 17](#_Toc150462836)

[5.2 Разработка алгоритмов 17](#_Toc150462837)

[6 РЕЗУЛЬТАТ РАБОТЫ 19](#_Toc150462838)

[ЗАКЛЮЧЕНИЕ 23](#_Toc150462839)

[СПИСОК ЛИТЕРАТУРЫ 25](#_Toc150462840)

[ПРИЛОЖЕНИЕ А 26](#_Toc150462841)

[ПРИЛОЖЕНИЕ Б 27](#_Toc150462842)

[ПРИЛОЖЕНИЕ В 28](#_Toc150462843)

[ПРИЛОЖЕНИЕ Г 29](#_Toc150462844)

[ПРИЛОЖЕНИЕ Д 30](#_Toc150462845)

**1. Задание на курсовую работу**

Овладеть практическими навыками проектирования и разработки законченного, отлаженного и протестированного программного продукта с использованием языка высокого уровня С++ ,овладеть практическими навыками проектирования и разработки законченного, отлаженного и протестированного программного продукта с использованием языка высокого уровня С++. Разработать программу “File Encryptor” с использование среды разработки Qt.

Преимущество Qt:

1. Простота использования: Qt имеет легко понимаемую и документированную структуру, что делает его привлекательным для начинающих разработчиков. Он также предлагает однородную архитектуру, что способствует более эффективной разработке.
2. Разработка GUI: Qt предоставляет обширный набор компонентов для создания графических пользовательских интерфейсов. Это включает в себя Qt Quick, который позволяет быстро и просто создавать интерфейсы с использованием специального языка QML. Это упрощает процесс создания интерфейса и делает его более интерактивным.
3. Кроссплатформенность: Qt является мощным инструментом для создания кроссплатформенных приложений, что означает, что вы можете разрабатывать один и тот же код и запускать его на различных операционных системах, таких как Windows, macOS и Linux.
4. Гибкость и масштабируемость: Qt можно использовать в разнообразных типах приложений, включая настольные и мобильные приложения, приложения для специализированного оборудования и встроенных систем.

В целом, использование Qt в C++ дает много преимуществ для разработчиков, которые хотят создавать кроссплатформенные приложения с графическим пользовательским интерфейсом. Это мощный и гибкий инструмент, который можно использовать в широком спектре приложений, от настольных и мобильных приложений до специализированного оборудования и встроенных систем.

2. ВВЕДЕНИЕ

В современном мире информационная безопасность играет ключевую роль, и защита конфиденциальных данных становится все более важной задачей. Одним из способов обеспечения этой защиты является шифрование файлов. Шифрование файлов – это процесс преобразования информации в нечитаемый вид с последующей возможностью восстановления оригинальных данных с помощью специального ключа.

Целью данного курсового проекта является разработка шифратора файлов на языке программирования C++. Этот проект предоставит возможность более глубокого понимания принципов симметричного шифрования, а также позволит создать инструмент, способный защищать конфиденциальные данные от несанкционированного доступа. Мы рассмотрим различные методы шифрования, реализацию алгоритмов и структуры данных, необходимые для данной задачи.

Курсовой проект "Шифратор файлов на C++" обеспечит студентов знаниями и навыками в области криптографии, программирования и информационной безопасности, что позволит им лучше понимать и применять шифрование в различных сферах жизни, начиная с защиты личных данных и заканчивая работой с конфиденциальными корпоративными файлами.

3. ОБЗОР ЛИТЕРАТУРЫ

Литературные ресурсы, предоставленные различными авторами и институтами, предлагают обширное понимание алгоритма Advanced Encryption Standard (AES) и его внедрения в современные системы шифрования данных. Книги, такие как предоставляют теоретическую основу для понимания принципов работы AES.

Официальный стандарт AES, предоставленный Национальным институтом стандартов и технологии США, содержит официальные спецификации и рекомендации по реализации алгоритма. Это важный ресурс для тех, кто стремится понять стандартные протоколы и процедуры, связанные с AES.

Официальная документация: Сайт Qt: Официальный сайт Qt содержит обширную документацию, включая руководства, API—справочники, примеры кода и другие материалы.

Руководства и Введение в Qt:

Getting Started: Раздел "Getting Started" в документации обычно предоставляет информацию о том, как установить Qt, настроить среду разработки и создать простое приложение. Overview: Введение в фреймворк, его основные концепции и принципы.

Создание графического интерфейса: Qt Widgets: Информация о виджетах Qt, базовых элементах управления, таких как кнопки, поля ввода и другие. Qt Quick и QML: Документация о создании интерфейсов с использованием декларативного языка QML и фреймворка Qt Quick.

Работа с сетью и базами данных: Qt Network: Инструменты для работы с сетью, включая HTTP—запросы, сокеты и другие. Qt SQL: Информация о работе с базами данных, включая поддержку различных СУБД.

Многозадачность и Параллелизм: Qt Concurrency: Раздел, посвященный поддержке многозадачности и параллелизма в Qt.

Межплатформенная разработка: Platform Notes: Рекомендации и особенности для кроссплатформенной разработки на разных операционных системах. Deployment: Инструкции по развертыванию Qt—приложений на различных платформах.

Примеры кода и Учебные проекты:Qt Examples: Обширный набор примеров кода для различных компонентов Qt.Qt Tutorials: Учебные проекты и туториалы, позволяющие освоить различные аспекты фреймворка.

Форумы и Сообщества: Qt Forum: Онлайн—форумы, где разработчики могут задавать вопросы, делиться опытом и получать поддержку от сообщества Qt.

Обновления и Дополнительные ресурсы: Блог Qt: Официальный блог с новостями, статьями и обновлениями от команды разработчиков. Дополнительные ресурсы: Дополнительные материалы, такие как видеоуроки, вебинары и другие образовательные ресурсы. Qt предоставляет обширные средства для создания высококачественных приложений, и его документация является важным ресурсом для разработчиков, стремящихся освоить этот фреймворк.

3.1 ОБЗОР МЕТОДОВ И АЛГОРИТМОВ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ

3.1.1. Симметричное шифрование:

Алгоритм Advanced Encryption Standard (AES): AES является одним из наиболее распространенных алгоритмов симметричного шифрования, предоставляющим высокую степень безопасности и эффективность. Рассмотрите реализацию AES для шифрования и расшифровки файлов.

**3.1.2. Хэширование:**

SHA—256 (Secure Hash Algorithm 256—bit): Хэширование используется для проверки целостности данных. Рассмотрите использование SHA—256 для создания хеш—сумм файлов, что позволит проверять, не были ли изменены данные после шифрования.

**3.1.3. Управление ключами:**

Генерация и хранение ключей: Разработайте методику для безопасной генерации и хранения секретных ключей, необходимых для шифрования и расшифровки файлов.

**3.1.4**. **Интерфейс пользователя:**

Создание пользовательского интерфейса: Разработайте простой и интуитивно понятный интерфейс для пользователей, позволяющий им выбирать файлы для шифрования и указывать параметры шифрования.

# 4 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВНИЕ

В данном разделе описываются входные и выходные данные программы, диаграмма классов, а также приводится описание используемых классов и их методов.

**4.1 Структура входных и выходных данных.**

Для решения задачи был выбран язык программирования С++ и методология объектно—ориентированного программирования. В процессе разработки программы были использованы различные возможности языка С++, которые будут описаны ниже.

Входные данные:

Файл, который требуется зашифровать.

Пароль (который будет случайно сгенерирован) для шифрования файла.

Выходные данные:

Зашифрованный файл в нечитаемом формате.

Пароль для каждого файла (должен быть сохранен в безопасном месте).

**4.2. Разработка диаграммы классов.**

Диаграмма классов данной работы показана в приложении А.

4.3. Описание классов.

Для создания программы шифрования и дешифрования файлов с использованием метода AES в C++ с интерфейсом в Qt, вы можете создать следующую структуру классов:

4.3.1 Класс MainWindow

MainWindow — класс qtWidget являющий основным окном приложения.

Описание полей класса:

* QString file\_path — путь к файлу который мы собираемся шифровать/расшифровать.
* bool isPasswordVisible — булевая переменная, возвращающая скрыт ли пароль в интерфейсе или нет.

Описание методов:

* void updateButtonText() — // Метод класса MainWindow для обновления текста на кнопке в зависимости от расширения файла.
* MainWindow(QWidget \*parent) — Конструктор класса MainWindow
* ~MainWindow() — Деструктор класса MainWindow.
* void on\_pushButton\_file\_clicked() — Метод класса MainWindow, вызываемый при нажатии кнопки выбора файла.
* void password\_operations() — Метод класса MainWindow для обработки операций с паролями.
* void on\_lineEdit\_password\_textChanged(const QString &text) — Метод класса MainWindow, вызываемый при изменении текста в поле ввода пароля.
* void check\_password() — Метод класса MainWindow для проверки совпадения паролей и окрашивания полей в зависимости от результата.
* void encryptFile() — Метод класса MainWindow для шифрования файла.
* void copyFile(const QString &sourceFilePath, const QString &destFilePath) — Метод класса MainWindow для копирования файла.
* void deleteFile(const QString &filePath) — Метод класса MainWindow для удаления файла.
* void decryptFile() — Метод класса MainWindow для расшифрования файла
* void on\_pushButton\_clicked() — Метод класса MainWindow, вызываемый при нажатии основной кнопки.

**4.3.2 Класс UI\_create\_password**

UI\_create\_password — класс qtWidget являющийся окном, вызываемым при создании случайного пароля.

Описание полей класса:

* bool isLower — булевая переменная, возвращающая будет ли пароль состоять из символов нижнего регистра
* bool isUpper — булевая переменная, возвращающая будет ли пароль состоять из символов верхнего регистра
* bool isNumber — булевая переменная, возвращающая будет ли пароль состоять из цифр
* bool isSymbol — булевая переменная, возвращающая будет ли пароль состоять из символов (таких как !@#$%^&\*)
* bool isCopy — булевая переменная, возвращающая копировать ли пароль в буфер
* Описание методов класса:
* void on\_horizontalSlider\_actionTriggered(int action) — Метод класса create\_password, вызываемый при изменении положения слайдера
* void on\_pushButton\_ok\_clicked() — Метод класса create\_password, вызываемый при нажатии кнопки "OK"

**4.3.3 Класс Password**

Password — класс реализующий создание пароля из случайных символов

Описание методов класса:

* Password() — Конструктор класса Password.
* QString createPassword(int length, bool isUpper, bool isLower, bool isNumbers, bool isSymbols, bool isCopy) — Метод класса Password для создания пароля.

**4.3.4 Класс SHA256**

SHA256 — класс реализующий хеширование пароля, методом SHA256

Описание полей класса:

* uint32\_t H[8] — Инициализация переменных хеш—значения.
* Описание методов класса:
* std::string preprocess(const std::string& input)— Метод класса SHA256 для предварительной обработки входных данных
* void processBlock(const uint8\_t\* block) — Метод класса SHA256 для обработки блока данных
* std::string hash(const std::string& input) — Метод класса SHA256 для вычисления хеша строки

**4.3.5 Класс AES**

AES — класс, реализующий шифрование данных нашего файла методом AES

Описание методов класса:

* AES() — Конструктор класса AES
* void SubWord(unsigned char \*a) — Заменяет каждый из четырех байтов массива a на соответствующий байт из sbox.
* void RotWord(unsigned char \*a) — Циклический сдвиг байтов массива a на одну позицию влево.
* void XorWords(unsigned char \*a, unsigned char \*b, unsigned char \*c) — Выполняет побитовую операцию XOR для каждой из четырех пар байтов массивов a и b, сохраняя результат в массиве c.
* unsigned char xtime(unsigned char b) — Выполняет операцию умножения байта b на x в поле Галуа.
* void Rcon(unsigned char \*a, unsigned int n) — Генерирует раундовый константный массив для ключа.
* void KeyExpansion(const unsigned char key[], unsigned char w[]) — Расширяет ключ для использования в алгоритме AES.
* void AddRoundKey(unsigned char state[4][Nb], unsigned char \*key) — Выполняет операцию побитового XOR для каждого элемента состояния и соответствующего ключа.
* void SubBytes(unsigned char state[4][Nb]) — Заменяет каждый элемент состояния на соответствующий элемент из sbox.
* void ShiftRow(unsigned char state[4][Nb], unsigned int i, unsigned int n) — Циклический сдвиг строки i влево на n позиций.
* void ShiftRows(unsigned char state[4][Nb]) — Выполняет циклический сдвиг каждой строки состояния влево на соответствующее количество позиций.
* void MixColumns(unsigned char state[4][Nb]) — Выполняет операцию MixColumns над состоянием.
* void EncryptBlock(const unsigned char in[], unsigned char out[], unsigned char \*roundKeys) — Шифрует блок данных с использованием ключа и выполняет необходимые операции.
* unsigned char \*EncryptECB(const unsigned char in[], unsigned int inLen, const unsigned char key[]) — Шифрует данные в режиме ECB (Electronic Codebook).
* void InvSubBytes(unsigned char state[4][Nb]) — Заменяет каждый элемент состояния на соответствующий элемент из inv\_sbox.
* void InvMixColumns(unsigned char state[4][Nb]) — Выполняет операцию InvMixColumns над состоянием.
* void InvShiftRows(unsigned char state[4][Nb]) — Выполняет обратный циклический сдвиг каждой строки состояния влево на соответствующее количество позиций.
* void DecryptBlock(const unsigned char in[], unsigned char out[], unsigned char \*roundKeys) — Дешифрует блок данных с использованием ключа и выполняет необходимые операции.
* unsigned char \*DecryptECB(const unsigned char in[], unsigned int inLen, const unsigned char key[]) — Дешифрует данные в режиме ECB (Electronic Codebook).

**4.3.6 Класс File**

File — класс, реализующий различные операции с файлами

Описание методов класса:

* QByteArray readFile(const QString &fileName) — Считывает содержимое файла и возвращает его в виде QbyteArray.
* Void writeFile(const QString &fileName, const QByteArray &data) — Записывает данные в файл с указанным именем.
* quint64 fileSize(const QString &fileName)— Возвращает размер файла в байтах.

**5 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ**

**5.1 Разработка схем алгоритмов**

Метод encryptECB(const unsigned char in[], unsigned int inLen, const unsigned char key[]) шифрует предоставляемые данные. Схема метода показана в приложении Б.

Метод DecryptECB(const unsigned char in[], unsigned int inLen, const unsigned char key[]) расшифровывает предоставляемые данные. Схема метода показана в приложении В.

**5.2 Разработка алгоритмов**

**5.2.1 Метод encryptECB() класса AES:**

Шаг 1: Инициализация ключа шифрования, загрузка исходного ключа в массив расширенных ключей KeyExpansion.

Шаг 2: Добавление ключа раунда, XOR каждого байта блока данных с соответствующим байтом ключа раунда

Шаг 3: Замена каждого байта исходных данных на соответствующий байт из sbox.

Шаг 4: Сдвиг строк матрицы состояния влево на определенное количество позиций.

Шаг 5: Каждый столбец матрицы состояния умножается на фиксированный многочлен в поле Галуа.

Шаг 6: Поблочно складываем данные с раундовыми ключами

Шаг 7: Шаг 3 – Шаг 6 проводится несколько раундов (Nr — 1), где Nr – количество раундов, зависящее от длины ключа.

Шаг 8: Последний раунд – выполняются те же операции, но без Шага 5.

Шаг 9: Вывод результата.

**5.2.2 Метод decryptECB() класса AES.**

Шаг 1: Также создаются раундовые ключи на основе переданного ключа.

Шаг 2: Добавление последнего раундового ключа к зашифрованному блоку.

Шаг 3: Сдвиг строк матрицы состояния вправо на определенное количество позиций

Шаг 4: Замена каждого байта зашифрованных данных на соответствующий байт из inv\_sbox.

Шаг 5: Поблочно складываем данные с раундовыми ключами.

Шаг 6: Обратная операция, той, где каждый столбец матрицы состояния умножается на фиксированный многочлен в поле Галуа.

Шаг 7: Шаг 3 – Шаг 6 проводится несколько раундов (Nr — 1), где Nr – количество раундов, зависящее от длины ключа.

Шаг 8: Последний раунд – выполняются те же операции, но без Шага 6.

Шаг 9: Вывод результата.

**6 РЕЗУЛЬТАТ РАБОТЫ**

# На рисунке 6.1 изображена начало работы программы. В интерфейсе со старта программы доступны кнопки: открытие файла, создание пароля, вставка пароля из буфера, копирование пароля в буфер, показать/скрыть пароль, очистка пароля из строки для редактирования, а так же главная кнопка, при нажатии которой файл будет зашифрован/расшифрован.

# 

Рисунок 6.1 — Начало работы программы

На рисунке 6.2 показана работа кнопки создания пароля. При нажатии кнопки, открывается дополнительное окно с настройками пароля. В новом окне пользователь может настроить из чего будет состоять сгенерированный пароль и какую длину он будет иметь.

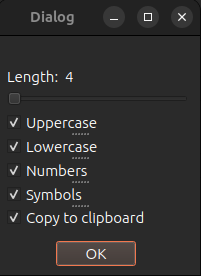


Рисунок 6.2 — Дополнительное окно создания пароля

На рисунке 6.3 отражен изначальный файл, который мы будем шифровать.

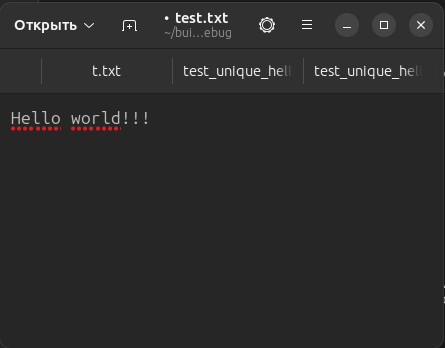


Рисунок 6.3 — Начальный файл

На рисунке 6.4 отражена программа перед шифрованием файла. При создании индивидуального пароля, нужно его подтвердить, то есть ввести точно такой же пароль в отведенное для этого места. Так же выбирается путь, куда будет сохранен зашифрованный файл, путь можно оставить по умолчанию.

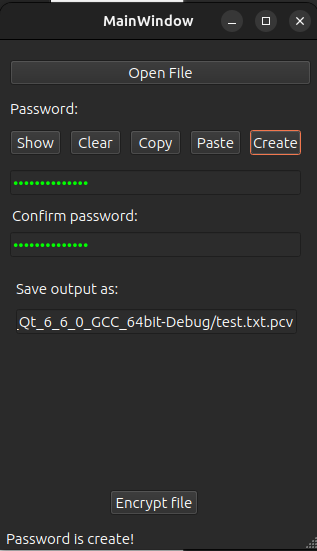


Рисунок 6.4 — Интерфейс программы перед шифрованием файла

На рисунке 6.5 показан зашифрованный файл, в нечитабельном формате .pcv

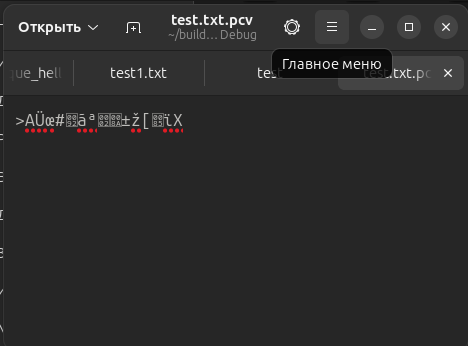


Рисунок 6.5 — Зашифрованный файл

На рисунке 6.6 отражен интерфейс программы перед расшифровкой файла. Для расшифровки файла нам нужно ввести пароль, которым мы шифровали наш файл, так же кнопка Create и поле Confirm Password являются не активными, так как это бессмысленно по логике нашей программы; кнопка Encrypt file заменяется на Decrypt file.

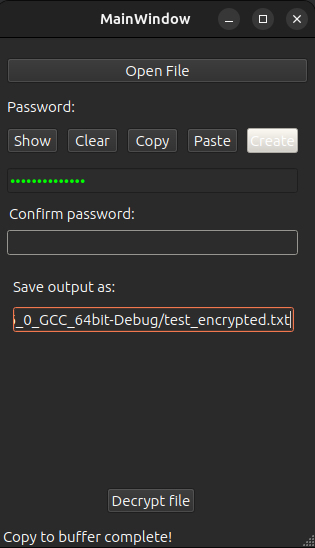


Рисунок 6.6 — Интерфейс программы перед расшифровкой файла

На рисунке 6.7 показан расшифрованный файл.

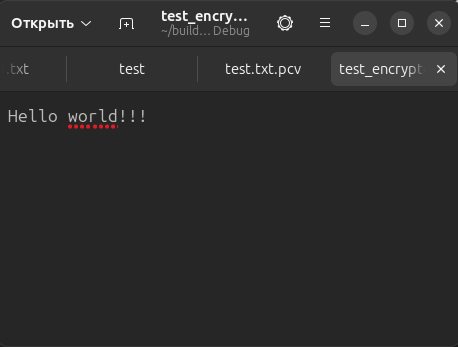


Рисунок 6.7 — Расшифрованный файл

ЗАКЛЮЧЕНИЕ

В данной курсовой работе была рассмотрена реализация алгоритма шифрования файлов методом Advanced Encryption Standard (AES). AES представляет собой симметричный блочный шифр, широко применяемый для обеспечения конфиденциальности данных в современных информационных системах.

Алгоритм AES основан на подстановочно — перестановочной сети с использованием нескольких раундов операций SubBytes, ShiftRows, MixColumns и AddRoundKey. Эти операции обеспечивают высокий уровень безопасности шифрования и устойчивость к различным методам атак.

В рамках работы был реализован класс, предоставляющий функционал для шифрования и дешифрования файлов с использованием AES в режиме Electronic Codebook (ECB). Разработанный программный модуль позволяет безопасно обрабатывать файлы различных форматов, обеспечивая сохранность данных и конфиденциальность информации.

Процесс шифрования включает в себя чтение файла блоками, применение алгоритма AES к каждому блоку данных, и запись результата обратно в файл. Дешифрование выполняется обратным образом, что обеспечивает восстановление исходного файла.

Основное внимание уделено обеспечению эффективности и удобства использования разработанного программного модуля. Тестирование и анализ результатов продемонстрировали правильное функционирование алгоритма и возможность применения его к файлам различных размеров и типов.

В заключение, разработанный модуль представляет собой эффективный инструмент для обеспечения безопасности файлов, что делает его актуальным и полезным компонентом в области информационной безопасности.

**Список литературы**

[1] „Таинственные страницы. Занимательная криптография“ — Иван Ефишов

[2] „Практическая криптография“ — Нильс Фергюсон и Брюс Шнайер

[3] „Математика и криптография“ — Р. В. Душкин

[4] „25 этюдов о шифрах“ — С.Дориченко, В.Ященко

[5] „Прикладная криптография: протоколы, алгоритмы и исходный код на C“ — Брюс Шнайер

[6] „Cryptography Engineering: Design Principles and Practical Applications“ — Bruce Schneier

ПРИЛОЖЕНИЕ А

*(обязательное)*

Диаграмма классов

ПРИЛОЖЕНИЕ Б

*(обязательное)*

Схема метода encryptECB()

ПРИЛОЖЕНИЕ В

*(обязательное)*

Схема метода decryptECB()

ПРИЛОЖЕНИЕ Г

*(обязательное)*

Код программы

ПРИЛОЖЕНИЕ Д

*(обязательное)*

Ведомость документов