Федеральное государственное автономное

образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О. В. Непомнящий
подпись
« ___ » _____ 2021 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 – «Информатика и вычислительная техника»

Учебный тренажёр по алгоритмам сжатия данных

Руководитель _____ ст. преподаватель К. В. Пушкарев
подпись, дата

Выпускник _____ И. Г. Чижова
подпись, дата

Нормоконтролер _____ К. В. Пушкарев
подпись, дата

Красноярск 2021

Федеральное государственное автономное

образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О. В. Непомнящий
подпись
« ___ » _____ 20 ___ г

ЗАДАНИЕ

НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

в форме бакалаврской работы

Студенту Чижовой Ирине Геннадьевне.

Группа: КИ17-06Б. Направление (специальность): 09.03.01 «Информатика и вычислительная техника».

Тема выпускной квалификационной работы: «Учебный тренажёр по алгоритмам сжатия данных».

Утверждена приказом по университету № 5627/с от 27.04.2021.

Руководитель ВКР: Пушкарев К. В., старший преподаватель кафедры ВТ ИКИТ СФУ.

Исходные данные для ВКР: нет.

Перечень разделов ВКР:

1. Анализ задания на выпускную квалификационную работу.
2. Проектирование и реализация приложения.
3. Инструкции.

Перечень графического материала: не требуется.

1 Задание

Разработать приложение для персональных компьютеров — учебный тренажёр по алгоритмам сжатия данных.

2 Основные требования

Разработанное приложение должно быть кроссплатформенным для настольных систем (ОС Windows, Linux), иметь графический пользовательский интерфейс, открытый исходный код и обладать следующими возможностями:

а) поддержка визуализации работы алгоритмов LZ78, LZW;

б) наглядная пошаговая демонстрация состояния компрессора и декомпрессора;

в) возможность задавать входные данные;

г) автоматический и ручной режим воспроизведения;

д) возможность возврата к предыдущим шагам;

е) режим самопроверки, в котором программа задаёт пользователю вопросы касательно работы алгоритма: какое слово добавится в словарь на следующем шаге, какое следующее значение будет на выходе компрессора или декомпрессора и др. После этого программа показывает, верный ли ответ.

Руководитель ВКР _____ К. В. Пушкарев
подпись

Задание принял к исполнению _____ И. Г. Чижова
подпись

31 октября 2020 г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Учебный тренажер по алгоритмам сжатия данных» содержит 36 страниц текстового документа, 26 иллюстраций, 5 таблиц, 13 использованных источников.

АЛГОРИТМЫ СЖАТИЯ ДАННЫХ, ПРИЛОЖЕНИЕ, ТРЕНАЖЁР, QT, C++, LZW, LZ78.

Цель работы: разработать приложение для персональных компьютеров — учебный тренажёр по алгоритмам сжатия данных.

Задачи:

- анализ задания на выпускную квалификационную работу;
- проектирование;
- реализация приложения.

В результате выполненной работы было спроектировано и реализовано приложение «Учебный тренажёр по алгоритмам сжатия данных». Был разработан графический интерфейс программы, структура приложения. Реализованы алгоритмы сжатия LZW и LZ78.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Учащиеся первого курса ИКИТ могут столкнуться с трудностями, которые связаны с изучением алгоритмов сжатия данных семейства LZ. Основная трудность заключается в отсутствии какой-либо визуализации, во всех основных источниках представлено только текстовое описание алгоритмов и результаты работы этих алгоритмов.

Учитывая эту проблему, было решено разработать программу, которая бы поддерживала полную визуализацию данных алгоритмов.

Цель работы: разработать приложение для персональных компьютеров — учебный тренажёр по алгоритмам сжатия данных.

Такое приложение должно обладать следующими особенностями:

- возможность автоматического и пошагового просмотра алгоритма;
- наличие режима самопроверки, в котором программа задает вопрос пользователя касаемо текущего состояния алгоритма. Например, каким будет следующее кодовое слово или что будет занесено в словарь на следующем шаге.

Для достижения цели были выполнены следующие задачи:

- анализ задания на выпускную квалификационную работу;
- проектирование;
- реализация приложения.

1 Анализ задания на выпускную квалификационную работу

В данной выпускной квалификационной работе необходимо разработать кроссплатформенное приложение с открытым исходным кодом, соответствующее следующим требованиям:

- графический интерфейс пользователя;
- поддержка операционных систем Windows и Linux;
- открытый исходный код.

Основные требования:

- поддержка визуализации работы алгоритмов LZ78, LZW;
- наглядная пошаговая демонстрация состояния компрессора и декомпрессора;
- возможность задавать входные данные алгоритмов;
- автоматический и ручной режим воспроизведения;
- возможность возврата к предыдущим шагам;
- режим самопроверки, в котором программа задаёт пользователю вопросы касательно работы алгоритма: каким будет словарь после завершения сжатия, какое слово добавится в словарь на следующем шаге, какое следующее значение будет на выходе компрессора или декомпрессора и др. После этого программа показывает, верный ли ответ. Перечень вопросов определяется индивидуально для каждого алгоритма сжатия.

1.1 Обзор существующих решений

Тема сжатия файлов не нова. На данный момент существует множество программ, осуществляющих компрессию и декомпрессию файлов, однако решений, показывающих эти процессы намного меньше, а тех, что предлагают проверить знания практически нет. Исходя из сказанного, при анализе существующих решений будем опираться на следующие критерии:

- демонстрация алгоритма;
- открытый исходный код;

- «ручной» режим;

- поддержка английского и русского языка;

- кроссплатформенность.

Рассматривая приложения, стоит также обратить внимание на их преимущества и недостатки

1.1.1 Тренажер «RLE»

Начнем обзор с тренажера «RLE» [1].

Решение разработано Константином Поляковым и бесплатно для некоммерческого использования. Автор не распространяет исходный код. На рисунке 1 представлено главное окно программы.

Преимущества:

- программа предоставляет возможность выбора между ручным вводом и открытием файла;

- программа высчитывает размер файла и коэффициент сжатия;

- присутствует вывод закодированного файла в двоичной и шестнадцатеричной системе счисления;

- имеется краткая справка об алгоритме сжатия и инструкция по использованию программы.

Недостатки:

- процесс сжатия открытого файла не показан, в отличие от файла, введенного вручную (рисунок 2 показывает окно, которое появляется при кодировании файла);

- нет поддержки английского языка;

- работа с приложением может производиться только с операционной системой Windows;

- отсутствие пошаговой работы алгоритма RLE (присутствует словарь, но он составляется автоматически и нет возможности посмотреть, как именно он составляется).

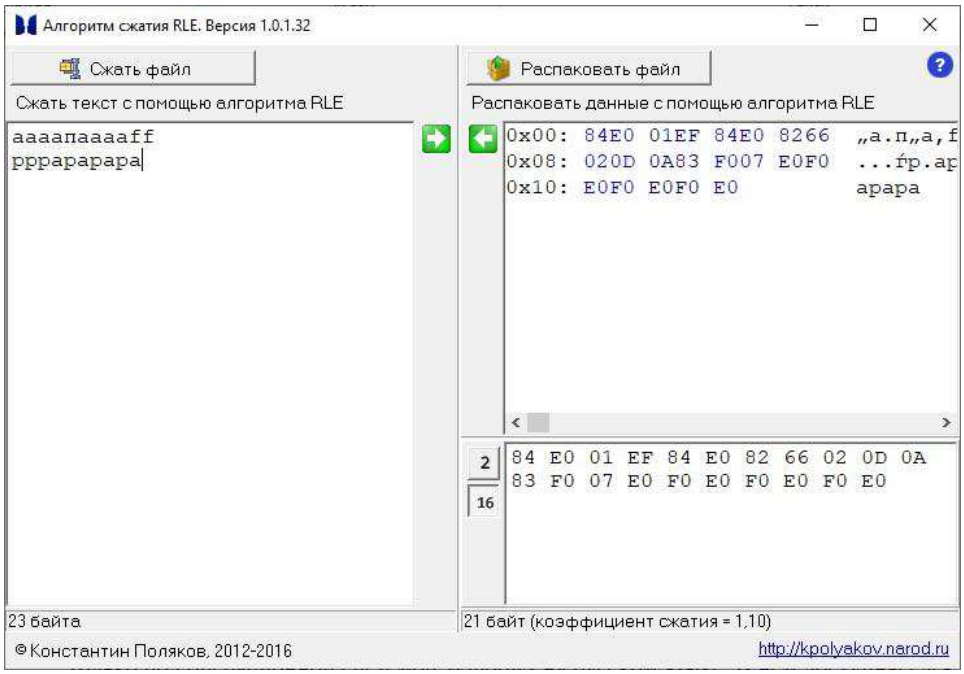

Рисунок 1 – Главное окно программы «RLE»

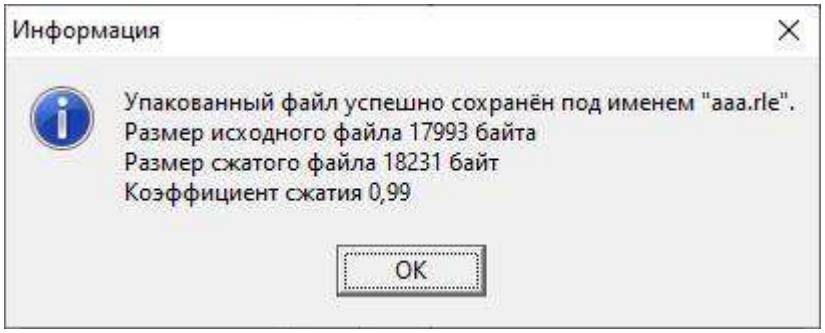

Рисунок 2 – Информационное окно, появляющееся после сжатия выбранного файла

1.1.2 Тренажер «Huffman»

Данная программа принадлежит тому же автору, что и тренажер «RLE» [1]. Она также бесплатно для некоммерческого использования и не имеет открытого исходного кода. Главное окно программы представлено на рисунке 3.

Достоинства данной программы:

- имеется выбор между вводом текста вручную и выбором файла;

- имеется выбор из трех алгоритмов (Хаффмана, Шеннона-Фано и LZW)

- присутствует инструкция по работе с программой и краткая справка об алгоритмах;

- программа определяет коэффициент сжатия, размер входного и выходного файла и другие параметры (рисунок 4);

- программа показывает словарь символов;

- присутствует возможность открытия файла.

Недостатки данной программы такие же, как и у тренажера «RLE», а именно:

- процесс сжатия открытого файла не показан, в отличие от файла, введенного вручную;

- нет поддержки английского языка;

- работа с приложением может производиться только с операционной системой Windows;

- отсутствие пошаговой работы алгоритмов, представленных в данном тренажере (нет возможность увидеть, как именно составлялся словарь).

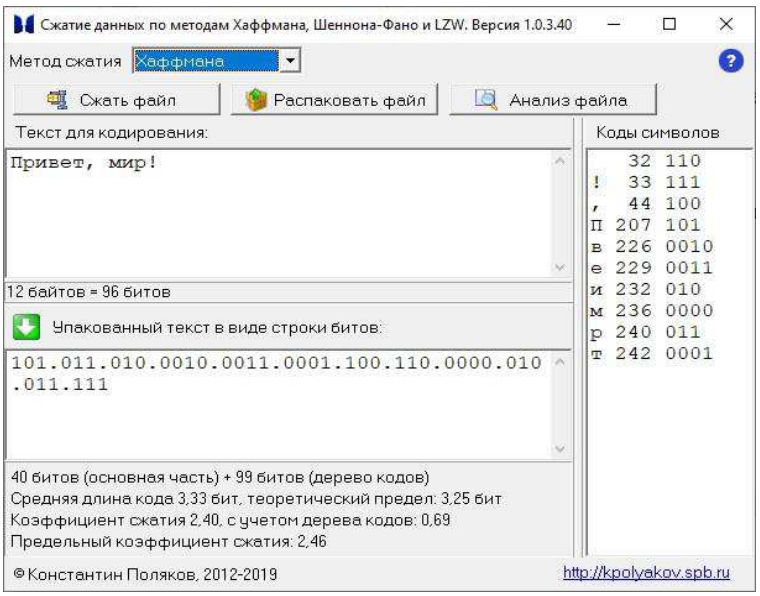

Рисунок 3 – главное окно программы «Huffman»

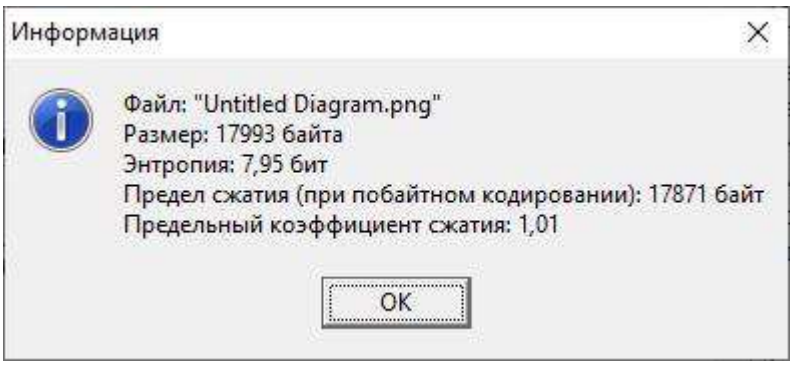

Рисунок 4 – Окно анализа файла

1.1.3 LZ Dictionary Algorithms

Программа написана Сами Хури и Сю-Чин Сюй [2]. Данное решение использует виртуальную машину Java, следовательно, программу можно запустить абсолютно с любого компьютера, который поддерживает виртуальную машину Java. Исходный код не открыт. Интерфейс программы показан на рисунках 5-6.

Преимущества данной программы:

- программа кроссплатформенная;
- имеется справка об алгоритмах и работе программы;
- выбор алгоритма;
- возможность открыть свой файл;
- алгоритм может выполняться как в ручном, так и в автоматическом режиме.

Недостатки данной программы:

- программа запускается через командную строку, что не совсем очевидно;
- из-за того, что программа запускается на виртуальной машине Java, она может быть медленной;
- отсутствует возможность сделать шаг назад;
- при вводе символа, не находящегося в словаре, алгоритм «LZW» работает некорректно;
- автоматический режим не демонстрирует выполнение алгоритма;
- программа распространяется только на английском языке.

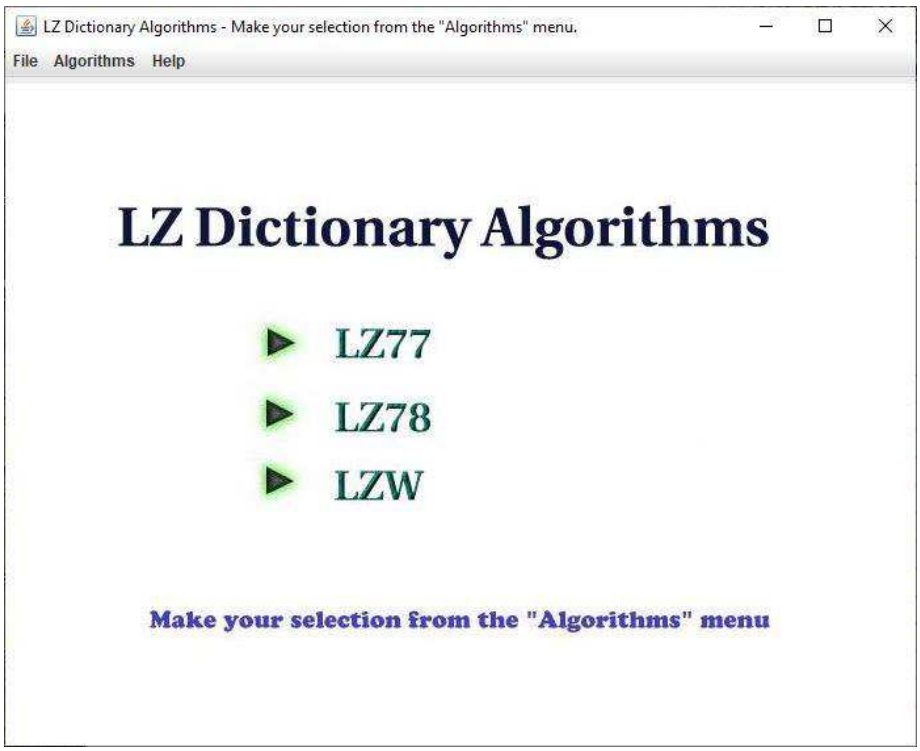

Рисунок 5 – Главное окно программы

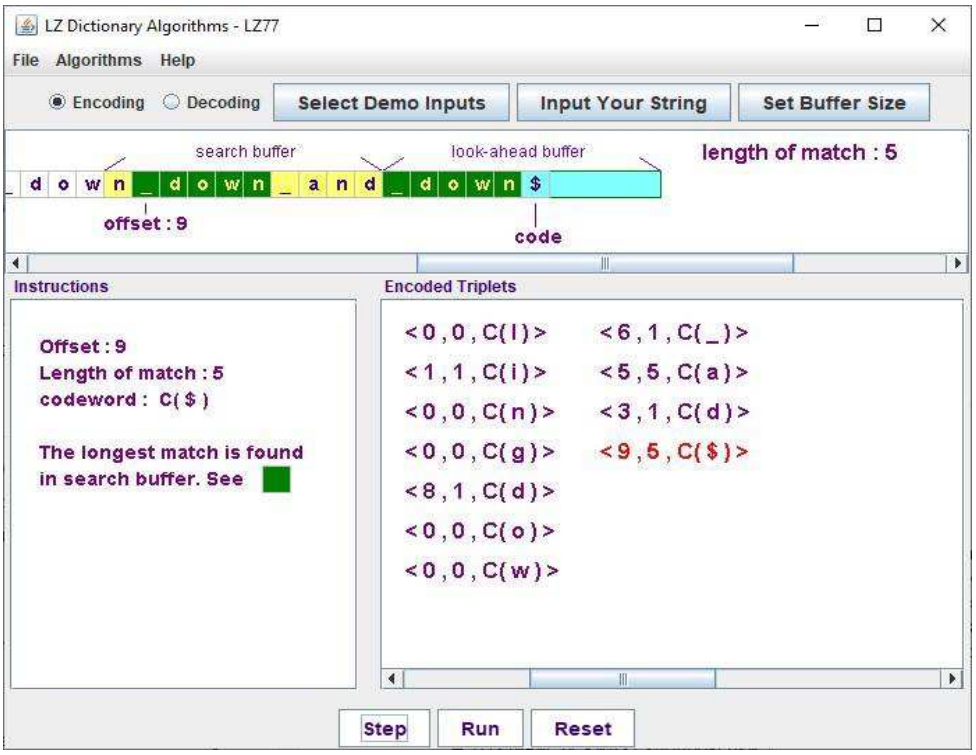

Рисунок 6 – Окно алгоритма

1.1.4 Тренажер «CodSH»

Данная программа написана на базе СФУ Родионом Кирюхиным и Сергеем Татаринцевым. Программа предоставляет возможность загрузить свой алфавит или имеющийся. Важным фактором так же является наличие режима «Контрольная работа», позволяющая студентам проверить свои знания [3]. Рисунок 3 отображает окно, котором происходит работа с алгоритмом. Интерфейс программы показан на рисунках 7-8.

Достоинства данной программы:

- наличие режима самопроверки;
- наличие обширной справочной информации об алгоритмах;
- наличие обхода алгоритма по шагам;
- наличие инструкции по использованию;
- наличие выбора алгоритма (Шеннона-Фано или Хаффмана);
- имеются подсказки для построения кода.

Недостатки данной программы:

- главное окно программы отображается некорректно (изменение размера окна не работает), текст окна не виден (рисунок 7);
- программа поддерживает только русский язык;
- нет возможности открыть любой файл для сжатия;
- нет возможности выйти из режима «Контрольная работа», пока не пройден тест.

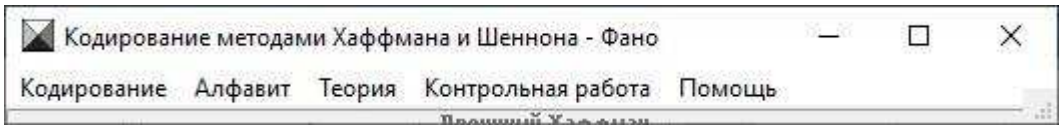

Рисунок 7 – Главное окно программы

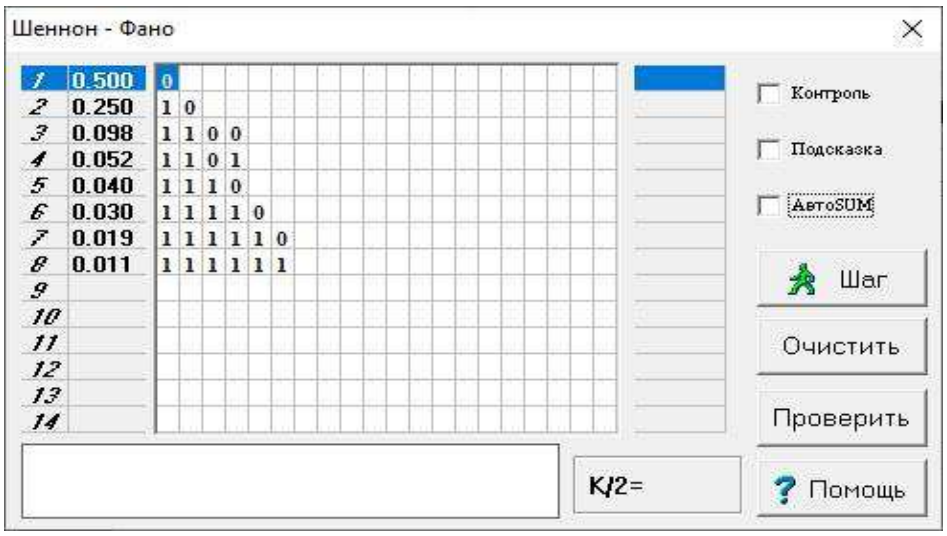

Рисунок 8 – Рабочее окно программы

1.1.5 Huffman Coding Visualization

Данное решение отличается от остальных тем, что представляет из себя веб-приложение [4]. Однако оно показывает построение дерева (рисунок 9).

Достоинства данного решения:

- наличие анимации построения дерева;

- возможность работать как в автоматическом режиме, так и в ручном режиме;

- данное решение будет работать на любом устройстве, но сайт не адаптируется под устройство с небольшим размером экрана (например смартфон). Это мы можем увидеть на рисунках 10 и 11. Если ориентация устройства горизонтальная возникают проблемы с запуском алгоритма по шагам или с его просмотром. Если ориентация будет вертикальной рассмотреть что-либо будет сложно, если увеличивать размер изображения опять же возникают сложности с рассмотрением алгоритма по шагам.

Недостатки данного решения:

- решение представляет из себя веб-приложение, что не позволяет работать с ним без интернета;

- решение предоставляется только на английском (однако это можно обойти автоматическим переводом страницы на любой язык, рисунок 12);

- нет возможности ввода текста больше, чем несколько слов, так как место для ввода текста небольшого размера.

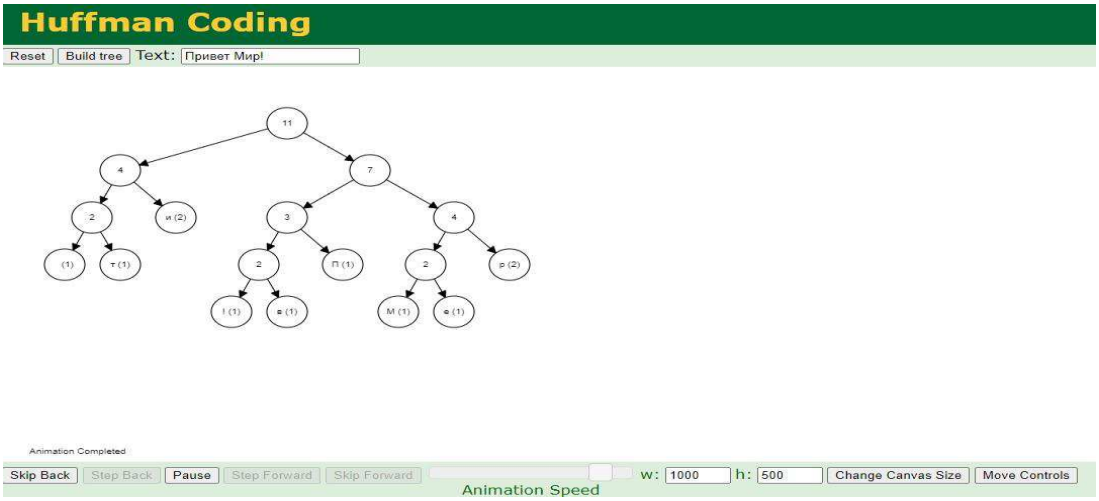

Рисунок 9 – Страница алгоритма

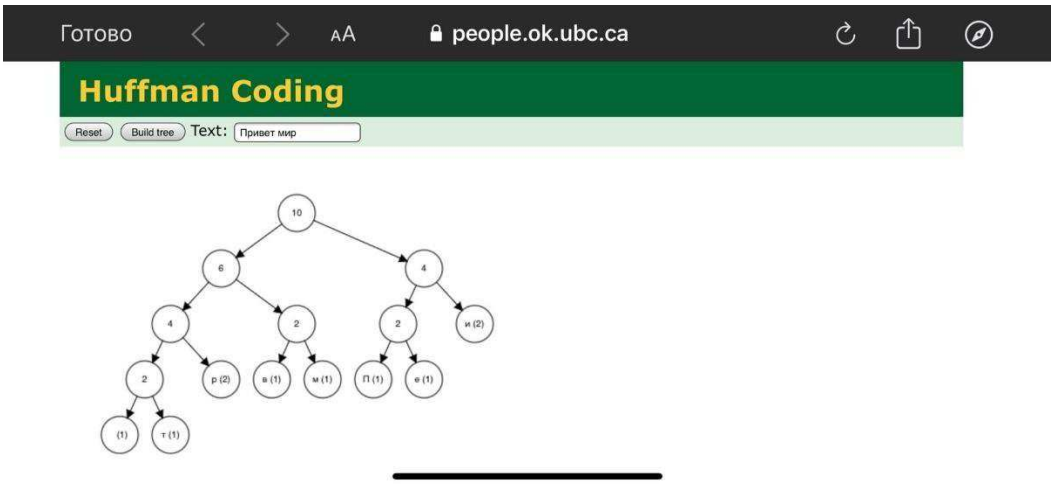

Рисунок 10 – Страница с горизонтальной ориентацией экрана смартфона

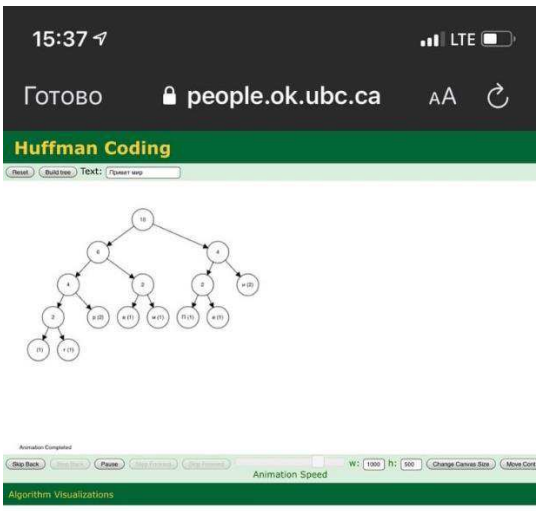

Рисунок 11 – Страница с вертикальной ориентацией смартфона

Рисунок 12 – Страница приложения, переведенная средствами браузера

1.1.6 Программа из статьи «An Interactive Learning Environment For Information And Communication Theory»

Данная статья написана Мохаммедом Хамада и Мохаммедом Хасаном [5]. В статье описана интерактивная среда для обучения теории информации и связи, включающая симулятор сжатия текста алгоритмами семейства LZ.

Приложение написано на языке Java2D. Интерфейс программы представлен на рисунке 13.

Достоинства данного решения:

- в данном решении присутствует тестовая часть;

- во время тестирования можно нажать кнопку «Hint» и получить подсказку по выполнению задания;

- присутствует пошаговое описание алгоритма;

- в данном решении присутствует возможность выбора алгоритма сжатия.

Недостатки данного решения:

- хотя в программе присутствует пошаговое описание алгоритма, просмотреть работу алгоритма можно только в автоматическом режиме.

- авторы не распространяют исходный код.

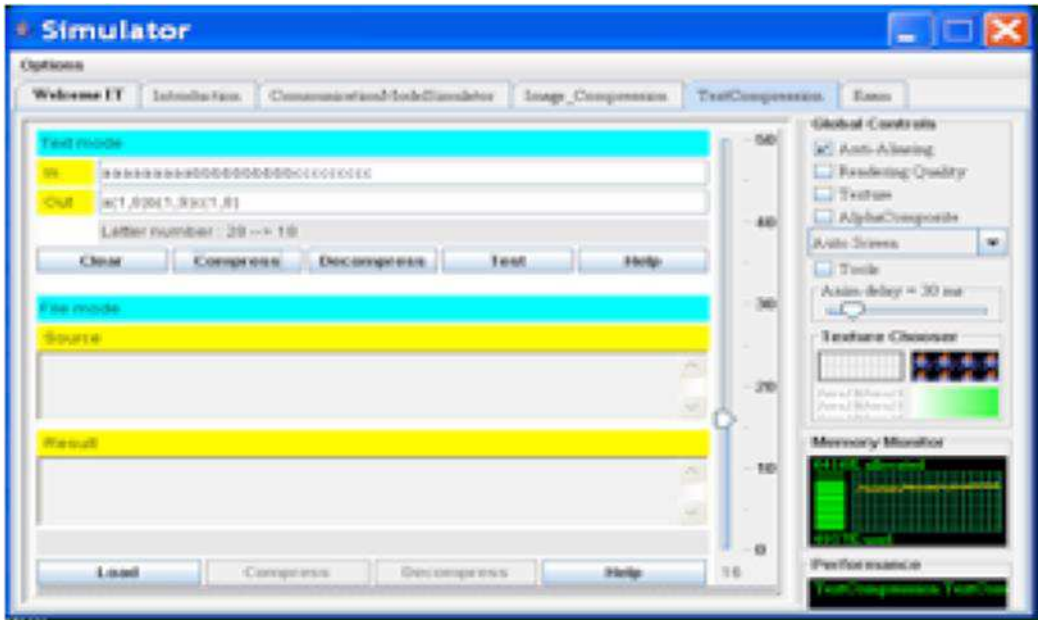

Рисунок 13 – Пример изображения из статьи

1.2 Итог обзора

Проведем сравнение по ряду параметров. Результаты приведены в таблице 1.

Таблица 1 – Сравнительная таблица

Название	Открытый исходный код	Кроссплатформенность	Графический интерфейс пользователя	Лицензия	Тестовый режим
Тренажер «RLE»	Нет	Нет	Да	Бесплатно с ограничениями	Нет
Тренажер «Huffman»	Нет	Нет	Да	Бесплатно с ограничениями	Нет
LZ Dictionary Algorithms	Нет	Да	Да	Нет информации	Нет
CodSH	Нет	Да	Да	Бесплатно	Да
Huffman Coding Visualization	Нет	Да (работа ведется через браузер)	Да	Бесплатно	Нет
Программа из статьи «An Interactive Learning Environment For Information And Communication Theory»	Нет	Да (само приложение написано на языке Java2D, что позволит запустить его на устройствах с виртуальной машиной Java)	Да	Нет информации	Да

Из всех приложений только «LZ Dictionary Algorithms» реализует нужные нам алгоритмы, однако его медленная работа из-за запуска на виртуальной машине Java, закрытый исходный код и наличие только английского языка не дает нам возможность опираться только на него. Достаточно удобная справка имеется в приложении «CodSH», в ходе создания справки будем опираться на него. У каждого из приложений имеется инструкция по его использованию, стоит добавить это и в свою программу.

1.3 Выбор средств разработки

Существуют различные средства разработки, у каждого из них есть свои достоинства и недостатки.

1.3.1 C++

C++ [6] является строго типизированным языком программирования и поддерживает различные парадигмы. При помощи библиотеки Qt [7] мы можем создать приложение с графическим интерфейсом, а также обеспечить его кроссплатформенность. Главное преимущество C++ это быстродействие программ написанных на нем. Однако он довольно сложен в освоении и имеет сложный синтаксис [8].

1.3.2 C#

C# является современным объектно-ориентированным языком, имеющим строгую типизацию. Относится к семейству языков C. Язык C# позволяет создавать оконные приложения. Он имеет множество функций, которые помогают упростить работу программиста, для примера функции сборки мусора. Некоторые из этих функций могут замедлить работу программы, однако они ускоряют ее разработку. C# является кроссплатформенным языком программирования [9].

1.3.3 Java

На данный момент язык Java является 5 по популярности языком программирования по версии StackOverflow [10]. Данный язык позволяет работать с большим количеством парадигм программирования. Приложения на Java можно запустить на любой из платформ, где имеется виртуальная машина Java. Однако программа написанная на Java может иметь малую производительность из-за запуска на виртуальной машине. Также код Java сложный и многословный [11].

1.3.4 Python

Python язык программирования, обладающий динамической типизацией и достаточно простым синтаксисом. На данный момент он является одним из самых быстроразвивающихся языков программирования [12]. На нем можно создавать различные приложения, в том числе и оконные. Программы, написанные на нем достаточно медленные, в сравнении с программами, написанными на других языках программирования [12]. На нем можно написать как приложения для Linux так и для Windows.

1.3.5 Итог обзора

Каждый из рассмотренных языков позволяет создавать кроссплатформенные приложения с графическим интерфейсом.

В отличие от других языков программирования, C++ позволяет создать самые быстро действенные программы, именно поэтому в качестве языка программирования мною был выбран язык C++.

Библиотека Qt позволит обеспечить поддержку визуализации алгоритмов семейства LZ.

1.4 Диаграмма прецедентов

На рисунке 14 представлена диаграмма прецедентов.

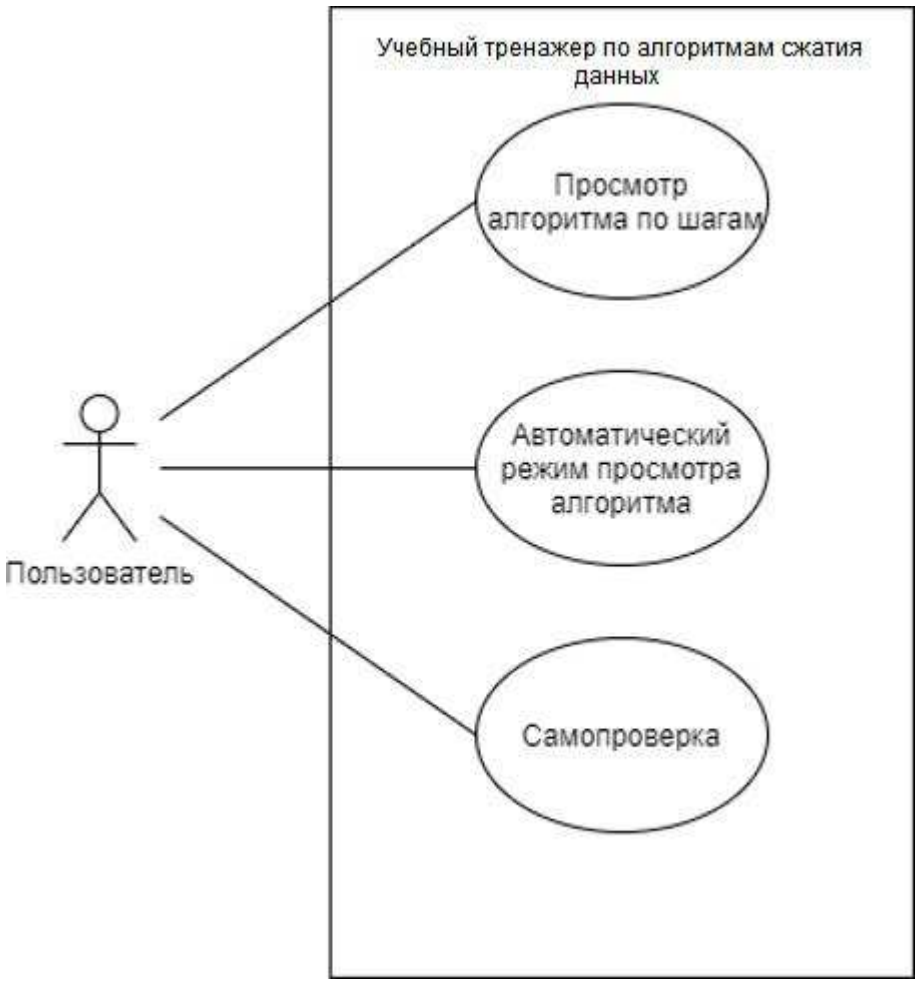

Рисунок 14 – Диаграмма прецедентов

Текстовое описание прецедентов приведено ниже.

Название: «Просмотр алгоритма по шагам».

Предусловие: Пользователь ввел данные в окно ввода и выбрал один из алгоритмов и нажимает кнопку «Шаг вперед».

Основной сценарий:

A. Программа делает шаг вперед и обновляет все экраны для вывода информации, если на данном шаге в этом есть необходимость.

B. Пользователь продолжает проходить алгоритм до тех пор, пока не закончатся введенные данные.

C. Программа выдает сообщение о том, что алгоритм завершил работу.

Постусловие: Обновлены все экраны и данные для алгоритмов.

Условие ввода в действие альтернативных сценариев.

Условие 1. Пользователь нажал кнопку «Шаг назад»

A. Программа делает шаг назад и обновляет все экраны для вывода информации.

Постусловие: Обновлены все экраны и данные для алгоритмов.

Условие 2. Пользователь нажал кнопку «Сначала»

A. Программа очищает алгоритм и готовит его к новым данным

B. Программа устанавливает режим алгоритма в режим кодирования.

Постусловие: Программа вернула алгоритм в изначальное состояние.

Название прецедента: «Просмотр в автоматическом режиме».

Предусловие: Пользователь ввел данные в окно ввода и выбрал один из алгоритмов.

Основной сценарий:

A. Пользователь нажимает кнопку «Пуск».

B. Программа делает один шаг алгоритма по таймеру до тех пор, пока алгоритм не будет закончен.

C. Программа выдает сообщение о том, что алгоритм завершил работу.

Условие ввода в действие альтернативных сценариев.

Условие 1. Пользователь нажал кнопку «Пауза»

A. Программа прекращает работу алгоритма.

Название прецедента: «Самопроверка».

Предусловие: Пользователь нажал на кнопку меню «Самопроверка» и выбрал интересующий его вопрос.

Основной сценарий:

A. Открывается диалоговое окно, в которое предлагается ввести ответ.

B. Пользователь вводит свой ответ и нажимает на кнопку «Ответ».

C. Программа выдает пользователю результат.

2 Проектирование и реализация приложения

2.1 Структура программы

Работа главного окна реализована через класс MainWindow. Реализация каждого алгоритма сжатия находится в классах LZW и LZ78, которые являются наследниками класса AlgorithmEncoding. Реализация алгоритмов декомпрессии находится в классах LZWDecoding и LZ78Decoding, которые являются наследниками класса AlgorithmDecoding. В свою очередь AlgorithmEncoding и AlgorithmDecoding наследуются от класса Algorithm (рисунок 15, 16). Классы Algorithm, AlgorithmEncoding и AlgorithmDecoding являются интерфейсами.

В программе так же присутствуют вспомогательные структуры CodeWord и DataDict. CodeWord позволяет хранить кодовое слово в формате, который упростит работу с алгоритмом декодирования, а также информацию о том будет ли добавлено кодовое слово или нет. DataDict позволяет хранить в словаре номер слова в словаре и само слово;

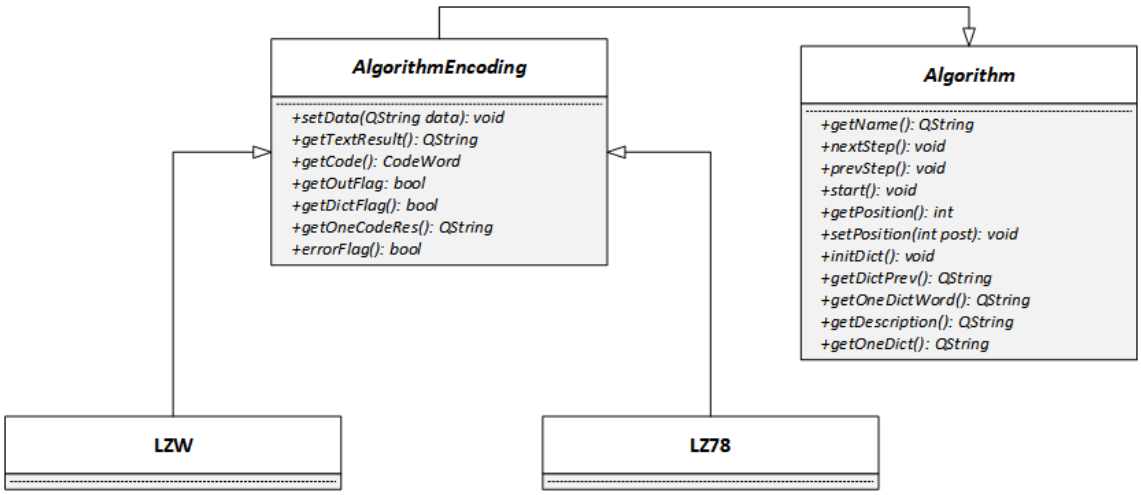

Рисунок 15 – Диаграммы классов. Алгоритмы кодирования

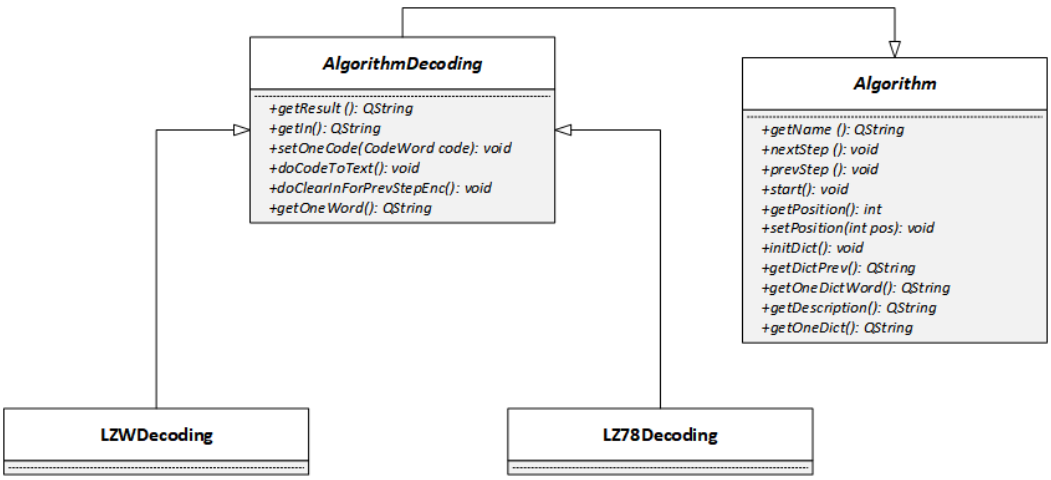

Рисунок 16 – Диаграммы классов. Алгоритмы декодирования

2.1.1 Описание интерфейса Algorithm

Интерфейс Algorithm предоставляет следующие методы:

- getName() возвращает название текущего алгоритма;
- nextStep() реализует один шаг вперед;
- prevStep() реализует один шаг назад;
- start() подготавливает алгоритм к новому циклу;
- getPosition() возвращает номер шага;
- setPosition(int pos) устанавливает номер шага;
- initDict() инициализирует словарь;
- getDictPrev() возвращает строку, содержащую все содержимое словаря, для его дальнейшего отображения;
- getOneDictWord() возвращает последнее слово, находящееся в словаре;
- getDescription() возвращает описание шага;
- getOneDict() возвращает последнюю запись словаря.

2.1.2 Описание интерфейса AlgorithmEncoding

Интерфейс AlgorithmEncoding предоставляет следующие методы:

- setData(QString data) устанавливает входные данные;

- getTextResult() возвращает закодированное слово в удобном для чтения формате;

- getCode() возвращает закодированное слово

- getOutFlag(), getDictFlag() возвращают флаг сигнализирующий об изменениях в выходной последовательности или в словаре;

- getOneCodeRes() возвращает последнее кодовое слово в текстовом формате;

- errorFlag() возвращает флаг, сигнализирующий об ошибке.

2.1.3 Описание интерфейса AlgorithmDecoding

Интерфейс AlgorithmDecoding предоставляет следующие методы:

- getResult() возвращает раскодированную последовательность;

- getIn() возвращает закодированную последовательность в текстовом формате;

- setOneCode() устанавливает одно кодовое слово;

- doCodeToText(CodeWord code) преобразовывает кодовое слово в формат, который будет удобен для восприятия. Например, для алгоритма LZ78 этот формат выглядит следующим образом: <[позиция префикса в словаре],[пришедший символ]>;

- doClearInForPrevStepEnc() возвращает входную последовательность на один шаг назад;

- getOneWord() возвращает одно раскодированное слово.

2.1.4 Описание классов LZW, LZ78, LZWDecoding и LZ78Decoding

Классы LZW и LZ78 являются наследниками класса AlgorithmEncoding.Классы LZWDecoding и LZ78Decoding являются наследниками класса AlgorithmDecoding. Список методов классов наследников идентичен родительскому классу. Рассмотрим списки полей классов наследников.

Класс алгоритма LZW содержат поля, представленные в таблице 2.

Таблица 2 – Поля класса LZW

Название	Функция
QString in	Входная последовательность
QList <CodeWord> out	Выходная последовательность
QList <DataDict> dictionary	Словарь
CodeWord code	Кодовое слово
QString rezult_for_text	Выходная последовательность в удобном для восприятия виде
int post	Позиция алгоритма в входной строке
int post_dict	Позиция алгоритма в словаре
bool flag	Флаг, сигнализирующий, что слово нашлось в словаре
bool dict_flag	Флаг, сигнализирующий об изменениях в словаре
bool out_flag	Флаг, сигнализирующий об изменениях в выходной последовательности
QString word	Слово, состоящее из предыдущего или предыдущих символов последовательности и текущего символа
QString in_dict	Следующая запись в словарь, для вывода на экран
QString description	Описание шага
QList <StateData> state	Состояния алгоритма
QString dictPrev	Предыдущее состояние словаря для вывода на экран
QString code_res	Кодовое слово в его текстовом формате

Поля класса LZ78 представлены в таблице 3

Таблица 3 – Поля класса LZ78

Название	Функция
QString in	Входная последовательность
int post	Позиция алгоритма в входной строке
int prev	Номер предыдущей последовательности в словаре
QString word	Слово, состоящее из предыдущее последовательности и текущего символа
CodeWord code_word	Кодовое слово
QList <CodeWord> out	Выходная последовательность
QString outStr	Выходная последовательность в формате строки

Окончание таблицы 3

Название	Функция
QList <DataDict> dict	Словарь
bool error	Флаг ошибки
QString one_dict	Следующая запись в словарь, для вывода на экран
QString description	Описание шага
bool out_flag	Флаг, сигнализирующий об изменениях в выходной последовательности
bool dict_flag	Флаг, сигнализирующий об изменениях в словаре
QString code_res	Кодовое слово в его текстовом формате
QString dict_prev	Предыдущее состояние словаря для вывода на экран
QList <StateLZW> states	Состояния алгоритма

Поля класса LZWDecoding представлены в таблице 4.

Таблица 4 – Поля класса LZWDecoding

Название	Функция
QString out	Выходная последовательность
QList <CodeWord> in	Входная последовательность
QString word	Слово, состоящее из предыдущего последовательности и текущего символа
int post	Позиция в входной последовательности
QString in_for_text	Входная последовательность в виде текста
QString in_dict	Следующая запись в словарь, для вывода на экран
QList <DataDict> dictionary	Словарь
QList <QString> words	Список слов, подающихся в выходную последовательность
QString dict_prev	Предыдущее состояние словаря для вывода на экран
QString one_word	Кодовое слово в тестовом формате
QString one_dict	Следующая запись в словарь
QString description	Описание шага
QList <QString> descr	Список, состоящий из описаний шага

Поля класса LZ78Decoding представлены в таблице 5.

Таблица 5 – Поля класса LZ78Decoding

Название	Функция
QString out	Выходная последовательность
QString in_str	Входная последовательность в виде текста
QList <CodeWord> in	Входная последовательность
QList <DataDict> dict	Словарь
int post	Позиция во входной последовательности
QString one_dict	Следующая запись в словарь, для вывода на экран
QString word	Слово, состоящее из предыдущее последовательности и текущего символа
QList <QString> words	Список слов, подающихся в выходную последовательность
QString dict_prev	Предыдущее состояние словаря для вывода на экран
QString one_word	Кодовое слово в текстовом формате
QString dict_word	Следующая запись в словарь
QString description	Описание шага
QList <QString> descr	Список, состоящий из описаний шага

2.1.5 Описание работы классов MainWindow и Question

В классе MainWindow происходит создание объекта класса выбранного алгоритма кодирования и декодирования, такая реализация позволяет просматривать работу алгоритмов параллельно. После этого мы можем обращаться к ним через интерфейс, такое решение позволяет беспрепятственно добавлять новые алгоритмы в программу.

Автоматический режим активируется сигналом, пришедшим с кнопки «Пуск» и по таймеру вызывает метод nextStep() до тех пор, пока не будет выполнен весь алгоритм или не придет сигнал кнопки «Пауза».

Ручной режим активируется сигналом, пришедшим с кнопки «Шаг вперед» и выполняет один шаг алгоритма. Шаг вперед можно выполнять до тех пор, пока не будет выполнен весь алгоритм.

Кнопка «Шаг назад» осуществляет один шаг алгоритма назад.

Режим самопроверки может быть вызван в любое время работы алгоритма. Он основан на следующих шагах инициализированного алгоритма, такой подход не требует генерации вопроса, но каждый раз этот вопрос будет уникальным. При выборе нужного вопроса в меню «Самопроверка». Происходит создание окна класса Question, и работа переходит к этому классу. Пользователь должен ввести свой ответ в поле ввода и нажать на копку «Принять ответ». Если пользователь не ввел ответ программа оповестит его об этом. В зависимости от ответа пользователя программа выведет сообщение с поздравление или пожеланием попробовать еще раз. В окно MainWindow нельзя перейти, пока открыто окно Question.

Изменение алгоритмов базируется на сигнале пришедшем от меню выбора алгоритма, при этом мы инициализируем новый алгоритм и вызываем метод start(), который реализован в каждом классе.

Изменение режима алгоритма также происходит по сигналу, пришедшему от одной из радиокнопок. После этого происходит очищение мест для вывода словаря, описания шага и результата. В окно ввода устанавливается последнее изменение, введенное алгоритмом кодирования или декодирования. В случае, если изменения внесены не были, программа выдаст предупреждение о том, что пользователь пытается изменить режим, не сделав хотя бы один шаг.

При нажатии на кнопку «Сначала», программа автоматически установит алгоритму режим кодирования и очистит поля в обоих режимах работы.

Работа классов алгоритмов с классом MainWindow происходит следующим образом (рисунок 17). С выбранным режимом алгоритма (кодирование или декодирование) пользователь нажимает на кнопку, которая относиться к работе алгоритма, например «Шаг вперед», класс MainWindow передает работу классу алгоритма, например AlgorihmEncoding, после этого происходит работа метода nextStep(), который вызывает класс алгоритма. После того как метод nextStep() завершил свою работу, класс алгоритма вызывает методы

getResult(), getOneDict(), getPost() и getDescription(), работа которых описана выше, и возвращает управление классу MainWindow. В классе MainWindow происходит обновление полей вывода словаря, результата и описания шага.

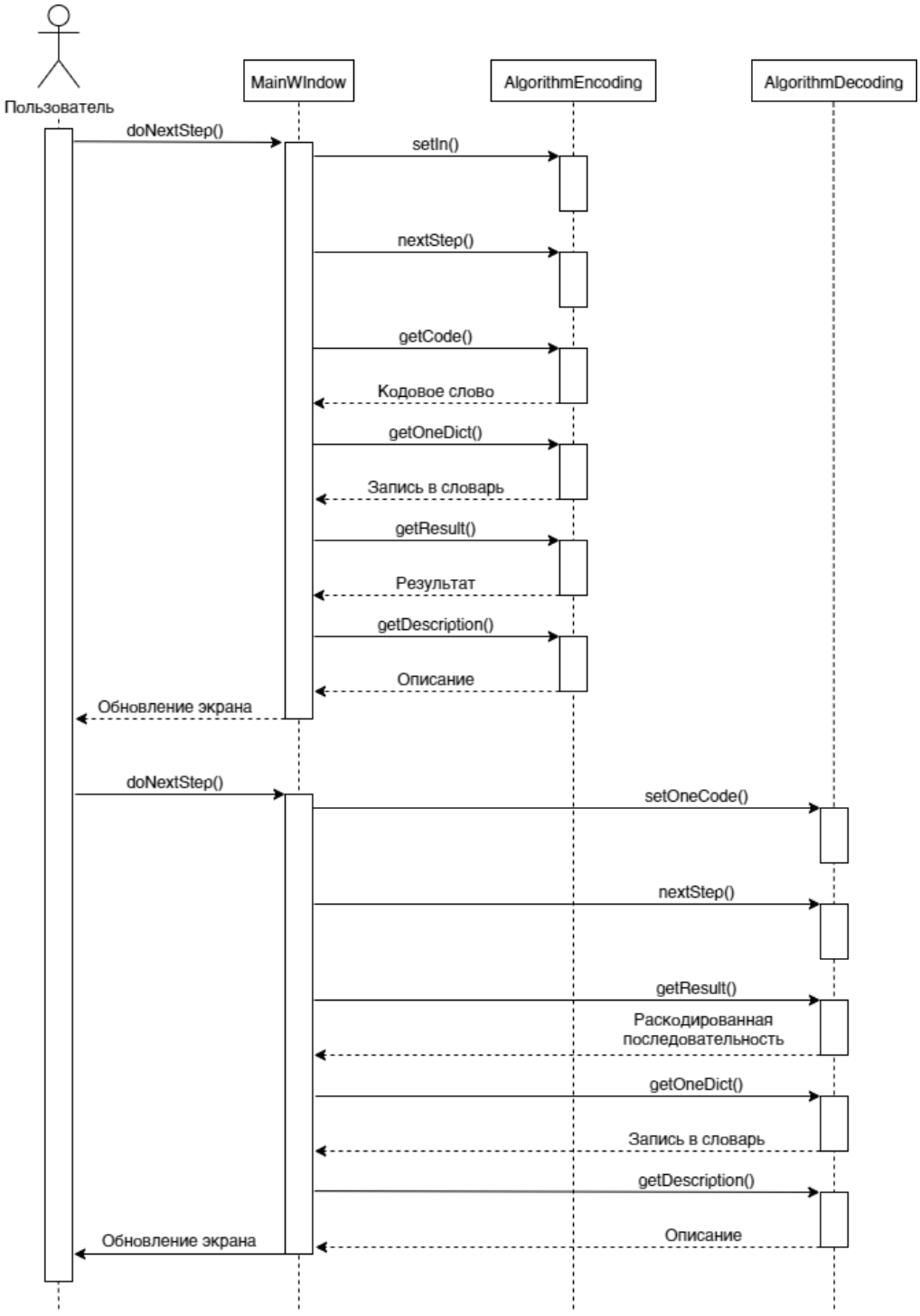

Рисунок 17 – Диаграмма последовательности

2.2 Интерфейс пользователя

При запуске программы появляется главное окно программы (рисунок 18). Здесь мы можем выбрать алгоритм и его режим работы (кодирование или декодирование). Пользователь может проверить себя во время работы, для этого нужно нажать кнопку «Самопроверка» на панели меню и выбрать один из предложенных вопросов, после этого откроется окно вопроса (рисунок 19). Предусмотрены пошаговый и ручной режим просмотра алгоритма. На каждом шаге программы имеется его текстовое описание.

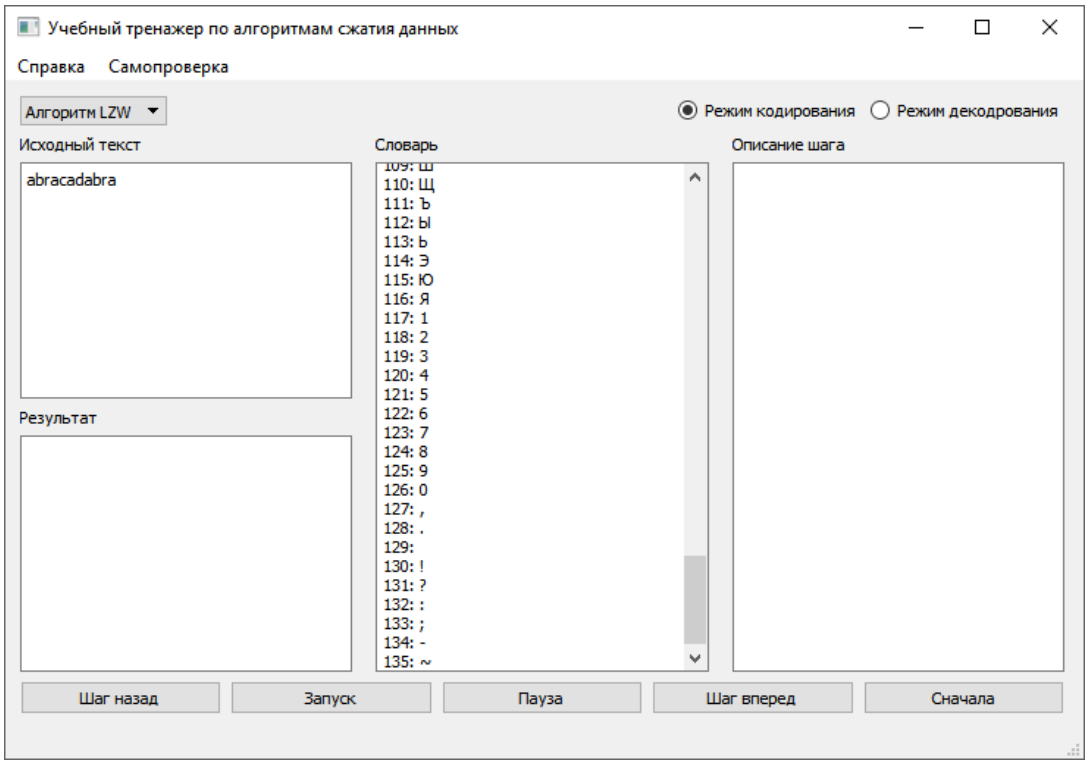

Рисунок 18 – Главное окно программы

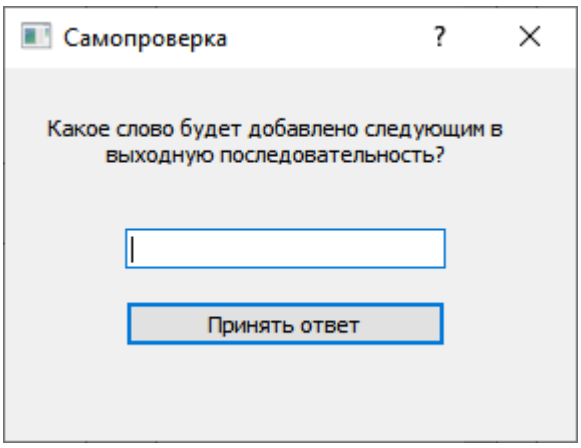

Рисунок 19 – Окно режима «Самопроверка»

3 Инструкции

3.1 Инструкция пользователя

При открытии приложения появляется главное окно программы с заранее введенным текстом и выбранным алгоритмом. При желании пользователь может поменять алгоритм (рисунок 20) и самостоятельно ввести текст (рисунок 21).

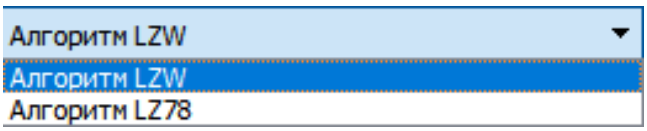

Рисунок 20 – Меню выбора алгоритма

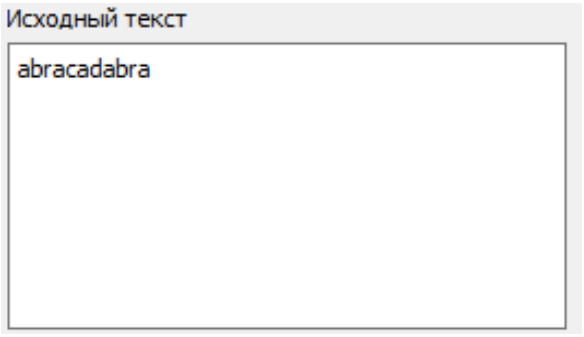

Рисунок 21 – Место для ввода текста

Кнопки (рисунок 22) позволяют управлять алгоритмом.

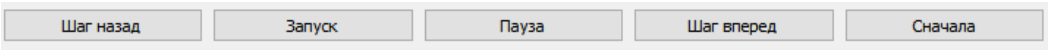

Рисунок 22 – Кнопки для управления алгоритмом

При нажатии кнопки «Запуск» алгоритм запустится в автоматическом режиме.

Кнопка «Пауза» приостанавливает автоматический режим. После ее нажатия появляется возможность просмотреть алгоритм по шагам.

Кнопка «Шаг вперед» делает один шаг алгоритма.

Кнопка «Шаг назад» возвращает пользователя на один шаг назад.

Кнопка «Сначала» возвращает алгоритм к первому шагу.

Кнопка меню «Справка» (рисунок 23) выдает информацию об алгоритме или о работе с программой.

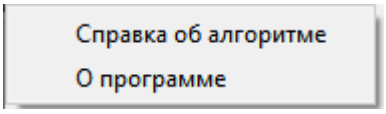

Рисунок 23 – Меню «Справка»

Кнопка меню «Самопроверка» (рисунок 24) дает возможность выбрать вопрос и ответить на него в только что открывшемся окне (рисунок 25)

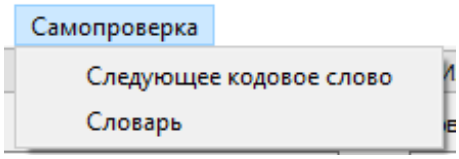

Рисунок 24 – Меню «Самопроверка»

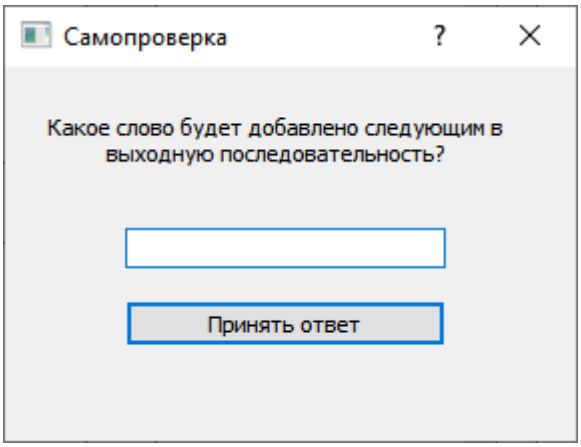

Рисунок 25 – Окно режима «Самопроверка»

С помощью кнопок, представленных на рисунке 26, происходит выбор режима работы алгоритма.

Рисунок 26 – Выбор режима работы алгоритма

3.2 Инструкция разработчика

Для компиляции проекта необходимо использовать библиотеку Qt версии 5.6.3 или выше. В качестве IDE рекомендуется использовать Qt Creator (рекомендуемая версия 4.0.3).

Список файлов и их назначение:

- mainwindow.cpp, mainwindow.ui, mainwindow.h – файлы класса MainWindow;

- question.cpp, question.ui, question.h – файлы класса Question;

- algorithm.h – интерфейс Algorithm;

- algorithmencoding.h – интерфейс AlgorithmEncoding;

- algorithmdecoding.h – интерфейс AlgorithmDecoding;

- lzw.cpp и lzw.h – файлы класса LZW;

- lz77.cpp и lz77.h – файлы класса LZ77;

- lz78.cpp и lz78.h – файлы класса LZ78;

- lzwdecoding.cpp и lzwdecoding.h – файлы класса LZW;

- lz77decoding.cpp и lz77decoding.h – файлы класса LZ77;

- lz78decoding.cpp и lz78decoding.h – файлы класса LZ78;

- datadict.h – структура DataDict;

- codeword.h – структура CodeWord;

- statedata.h – структура StateData;

- statelzw – структура StateLZW;

- LZ.pro – информация для сборки проекта с помощью qmake.

Чтобы добавить новый алгоритм, необходимо убедиться, возможно ли отобразить его с помощью существующего интерфейса. После необходимо создать класс для кодирования, который будет наследником класса AlgorithmEncoding, и класс для декодирования, который будет наследником класса AlgorithmDecoding. После этого в каждом классе необходимо реализовать все унаследованные методы. Для того, чтобы система увидела алгоритм, необходимо добавить его идентификатор в виджет ChooseAlg в форме окна MainWindow. Также необходимо добавить обработку этого выбора в конструктор класса MainWindow и метод makeChoice().

ЗАКЛЮЧЕНИЕ

В результате проделанной работе была создана программа «Учебный тренажер по алгоритмам сжатия данных». Данная программа находится в открытом доступе [13]. Программа поддерживает визуализацию алгоритмов сжатия LZ78, LZW и имеет два режима просмотра алгоритмов (автоматический и ручной). Программа в ручном режиме позволяет делать шаги вперёд и назад. Также в программе присутствует режим самопроверки, позволяющий пользователю проверить свои знания об алгоритмах сжатия.

Дальнейшие направления доработки:

- совместная демонстрация алгоритмов кодирования и декодирования;
- управление скоростью работы алгоритма в автоматическом режиме;
- добавление других алгоритмов сжатия.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Тренажеры для изучения алгоритмов сжатия данных: сайт Константина Полякова [Электронный ресурс]. - Режим доступа: https://kpolyakov.spb.ru/prog/compress.htm (дата обращения: 03.12.2020).

2. Sami Khuri: Education Tools [Электронный ресурс]. - Режим доступа: https://www.cs.sjsu.edu/~khuri/animation.html (дата обращения: 03.12.2020).

3. Образовательная система СФУ. Тренажеры по методам оптимального кодирования [Электронный ресурс]. - Режим доступа: https://e.sfu-kras.ru/mod/assign/view.php?id=712764 (дата обращения 03.12.2020).

4. Huffman Coding Visualisation [Электронный ресурс]. - Режим доступа: https://people.ok.ubc.ca/ylucet/DS/Huffman.html (дата обращения 03.12.2020).

5. Hamada M., Hassan M. An interactive learning environment for information and communication theory // Eurasia Journal of Mathematics, Science and Technology Education. – 2016. – Т. 13. – №. 1. – С. 35-59. – Режим доступа https://www.ejmste.com/download/an-interactive-learning-environment-for-information-and-communication-theory-4650.pdf (дата обращения 10.01.2021).

6. C++ Введение [Электронный ресурс]. – Режим доступа https://metanit.com/cpp/tutorial/1.1.php (дата обращения 29.05.2021).

7. Qt | Cross-platform software development for embedded & desktop [Электронный ресурс]. – Режим доступа https://www.qt.io/ (дата обращения 10.01.2021)

8. C++: Плюсы и минусы C++ [Электронный ресурс]. - Режим доступа: http://cjblogr.blogspot.com/2015/02/blog-post.html (дата обращения 29.05.2021).

9. C#: преимущества и недостатки – CODE BLOG [Электронный ресурс]. – Режим доступа https://shwanoff.ru/plus-minus-c-sharp/ (дата обращения 23.12.2020).

10. Stack Overflow Developer Survey 2020 [Электронный ресурс]. – Режим доступа https://insights.stackoverflow.com/survey/2020#technology (дата обращения 10.01.2021).

11. Плюсы и минусы программирования на Java | by Вероника | NOP::Nuances of Programming | Medium [Электронный ресурс]. – Режим доступа: https://medium.com/nuances-of-programming/плюсы-и-минусы-программирования-на-java-2861f4c2a0d5 (дата обращения 23.12.2020).

12. Язык программирования Python: плюсы, минусы, сфера применения. Каким языком является Python? | OTUS [Электронный ресурс]. – Режим доступа https://otus.ru/nest/post/1547/ (дата обращения 23.12.2020).

13. Irina-Chizhova/LZ-Algorithm-Simulator [Электронный ресурс]. – Режим доступа https://github.com/Irina-Chizhova/LZ-Algorithm-Simulator (дата обращения: 29.05.2021).

Федеральное государственное автономное

образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О. В. Непомнящий
подпись
« 08 » 06 2021 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 – «Информатика и вычислительная техника»

Учебный тренажёр по алгоритмам сжатия данных

Руководитель 07.06.2021 ст. преподаватель К. В. Пушкарев
подпись, дата

Выпускник 07.06.2021 И. Г. Чижова
подпись, дата

Нормоконтролер 07.06.2021 К. В. Пушкарев
подпись, дата

Красноярск 2021