Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

Игровое приложение “Покер”

БГУИР КР 1-40 01 01 104 ПЗ

Студент: гр. 251001 Василенко И.А.

Руководитель: асс. Шостак Е.В

.

Минск, 2023      /////// Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ПОИТ

––––––––––––––––––––––––

(подпись)

––––––––––––––––– 2023 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту  Василенко Ивану Анатольевичу

1. Тема работы “Программное средство редактирования, сравнения текстов и процентной оценки плагиата”

2. Срок сдачи студентом законченной работы 31.05.2023 г.–––

3. Исходные данные к работе

- среда разработки Delphi

- входные данные – текстовый файл с текстом для редактирования, сравнения, оценки плагиата

- выходные данные – текстовый файл с отредактированным текстом, c результатами сравнения

4. Содержание расчётно-пояснительной записки (перечень вопросов, которые подлежат разработке)

Введение.

1. Анализ прототипов, литературных источников и формирование требований к проектируемому программному средству;

2. Анализ требований к программному средству и разработка функциональных требований;

3. Проектирование программного средства;

4. Создание (конструирование) программного средства;

5. Тестирование, проверка работоспособности и анализ полученных результатов;

6. Руководство по установке и использованию;

Список используемой литературы

Заключение

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

1. "Программное средство редактирования, сравнения текстов и процентной оценки плагиата", А1, схема программы, чертеж

6. Консультант по курсовой работе

Фадеева Е.Е.

7. Дата выдачи задания 21.02.2023

8. Календарный график работы над курсовой работой на весь период проектирования (с обозначением сроков выполнения и процентом от общего объёма работы):

раздел 1 к 01.03.2023 – 15 % готовности работы;

разделы 2, 3 к 15.03.2023 – 30 % готовности работы;

разделы 4, 5 к 15.04.2023 – 60 % готовности работы;

раздел 6 к 15.05.2023 – 90 % готовности работы;

оформление пояснительной записки и графического материала к 31.05.2023 – 100 % готовности работы.

Защита курсовой работы с 01.06.2023 по 09.06.2023 г.––––––––––––––––––––

РУКОВОДИТЕЛЬ–––––– Е.В.Шостак

(подпись)

Задание принял к исполнению –––\_\_\_\_ ––

(дата и подпись студента)

//////

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc136533981)

[1. АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ 6](#_Toc136533982)

[1.1. Анализ аналогов к проектируемому программному средству 6](#_Toc136533983)

[1.2. Анализ алгоритма оценки плагиата и сравнения текстов 7](#_Toc136533988)

[1.3. Формирование требований к проектируемому программному средству 8](#_Toc136533989)

[2. АНАЛИЗ ТРЕБОВАНИЙ К ПС И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ 10](#_Toc136533990)

[2.1. Описание функциональных требований к программному средству 10](#_Toc136533991)

[2.2. Спецификация функциональных требований 11](#_Toc136533992)

[3. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА 12](#_Toc136533993)

[3.1 Разработка хэш-таблицы для эффективного сравнения 12](#_Toc136533994)

[3.2 Разработка алгоритмов сравнения двух текстов 12](#_Toc136534001)

[3.3 Разработка алгоритмов оценки уникальности 13](#_Toc136534002)

[3.4. Алгоритм разбиения на шинглы 13](#_Toc136534003)

[4. СОЗДАНИЕ ПРОГРАММНОГО СРЕДСТВА 15](#_Toc136534004)

[4.1 Проектирование модулей программного средства 15](#_Toc136534005)

[4.2 Разработка модулей программного средства 15](#_Toc136534006)

[5. ТЕСТИРОВАНИЕ, ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ 21](#_Toc136534010)

[5.1. Тестирование работоспособности ПС 21](#_Toc136534011)

[5.2. Анализ полученных результатов 23](#_Toc136534014)

[6. РУКОВОДСТВО ПО УСТАНОВКЕ И ЭКСПЛУАТАЦИИ 24](#_Toc136534028)

[ЗАКЛЮЧЕНИЕ 25](#_Toc136534041)

[Список использованной литературы 26](#_Toc136534042)

[ПРИЛОЖЕНИЕ А 27](#_Toc136534045)

[ПРИЛОЖЕНИЕ Б 32](#_Toc136534047)

# 

# **ВВЕДЕНИЕ**

В современном мире игровые приложения стали неотъемлемой частью цифровой культуры, проникнув в повседневную жизнь миллионов людей по всему миру. Эти приложения предоставляют уникальные возможности для развлечения, общения и даже обучения. В свете быстрого развития технологий мобильных устройств, виртуальной и дополненной реальности, а также компьютерных игр, изучение и разработка игровых приложений становятся важным направлением в области информационных технологий.

Цель данной курсовой работы заключается в анализе и разработке игрового приложения с использованием современных методов и технологий. В ходе исследования будет рассмотрено влияние игровых приложений на пользователей, а также рассмотрены основные принципы и технические аспекты создания подобных приложений.

Перед началом проектирования игрового приложения необходимо изучить существующие тенденции в области геймдева, а также провести анализ успешных проектов. Кроме того, рассмотрение особенностей жанра, платформы и целевой аудитории поможет определить ключевые элементы проектирования.

В рамках данной работы будет освещен процесс разработки игрового приложения с использованием выбранной платформы разработки. Рассмотрение архитектуры приложения, выбор технологий, а также этапы тестирования и оптимизации будут подробно проанализированы. Данная пояснительная записка содержит следующие основные разделы:

1. Анализ прототипов, литературных источников и формирование требований к проектируемому программному средству;
2. Анализ требований к программному средству и разработка функциональных требований;
3. Проектирование программного средства;
4. Создание (конструирование) программного средства;
5. Тестирование, проверка работоспособности и анализ полученных результатов;
6. Руководство по установке и использованию.
7. **АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ** 
   1. **Анализ аналогов к проектируемому программному средству**

Существует множество сервисов по оценке плагиата, сравнению и редактированию текстов:

1. Text.ru

Text.ru – онлайн-сервис проверки текста на уникальность. Text.ru анализирует материалы объёмом до 15 000 символов. При этом проверка происходит в порядке очереди: почти всегда приходится ждать, пока система обслуживает других пользователей. Систему антиплагиата на Text.ru нельзя обмануть просто изменив порядок предложений, фраз или слов, поменяв падежи, времена или добавив в текст новые слова.

1. Microsoft Word

Microsoft Word – редактор текстов. Microsoft Word выполняет все стандартные функции редактирования текстов, такие как создание документа, его сохранение, открытие документа, чтение документа, изменение параметров шрифтов для текста(цвет, размер и др.) , выравнивание текста и т.д. Однако возможности Microsoft Word намного больше. На сегодняшний момент в данной системе присутствуют функции вставки различных формул или нестандартных символов, разметка страницы и размещение текста на ней, добавление ссылок и множество других. Также в Microsoft Word присутствует функция сравнения текстов.

1. Advego Plagiatus

Сайт Advego — это целая система сервисов для продажи, заказа и анализа текстов. Инструмент для проверки на уникальность здесь тоже присутствует. На данном сайте текст статьи анализируется сразу по нескольким алгоритмам:

1. Алгоритм шинглов — проверяются точные совпадения фраз, находятся источники копипаста, а также страницы, на которых размещены украденные тексты.
2. Алгоритм лексических совпадений — проверяется схожесть набора лемм, терминов и значимых слов, находятся источники рерайтинга, а также страницы, совпадающие по тематике с проверяемой статьей.
3. Алгоритм псевдоуникализации — проверяется наличие сторонних символов и признаков обработки текста сервисами "повышения" уникальности.
4. Google Docs

Google Docs – текстовый редактор. Во многом повторяет функции Microsoft Word.Существуют также такие функции как отправить ссылку на документ письмом прямо из Google Docs, опубликовать документ онлайн или вставить в веб-страницу и т.д.

* 1. **Анализ алгоритма оценки плагиата и сравнения текстов**

Наиболее известным и часто используемым алгоритмом проверки текста на уникальность является *алгоритм шинглов*.

Шингл — это кусочек текста, размером в несколько слов. Шинглы идут внахлёст друг за другом. По тексту документа можно сформировать множество шинглов, число которых сопоставимо с числом слов в документе.

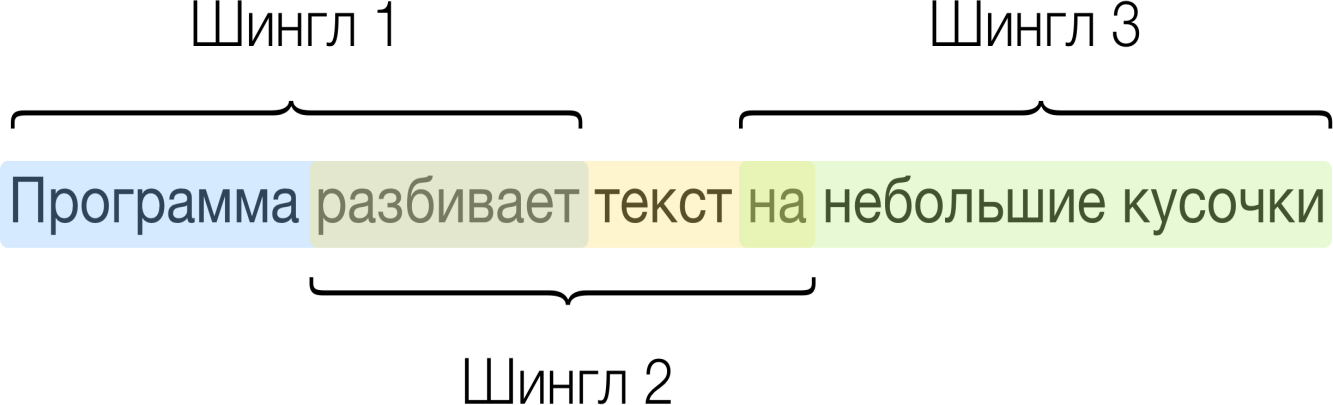


Рисунок 1.1 – разбиение на шинглы

При совпадении нескольких шинглов у двух документов будем считать, что документы пересекаются. Чем больше шинглов совпадает, тем больше одинакового текста в этой паре документов. Индекс занимается поиском документов, обладающих наибольшим количеством пересечений с проверяемым документом.

Индекс шинглов позволяет выполнять две основные операции:

1. Индексировать в себе шинглы документов с их идентификаторами:
2. Искать в себе и выдавать ранжированный список идентификаторов пересекающихся документов.

Индекс шинглов представляет собой несколько слоёв (массивов) с элементами постоянной длины — от 0 до 128 бит, — которая зависит не только от слоя и не обязательно кратна восьми.

Каждый из слоёв играет определённую роль. Какой-то делает поиск быстрее, какой-то экономит место.

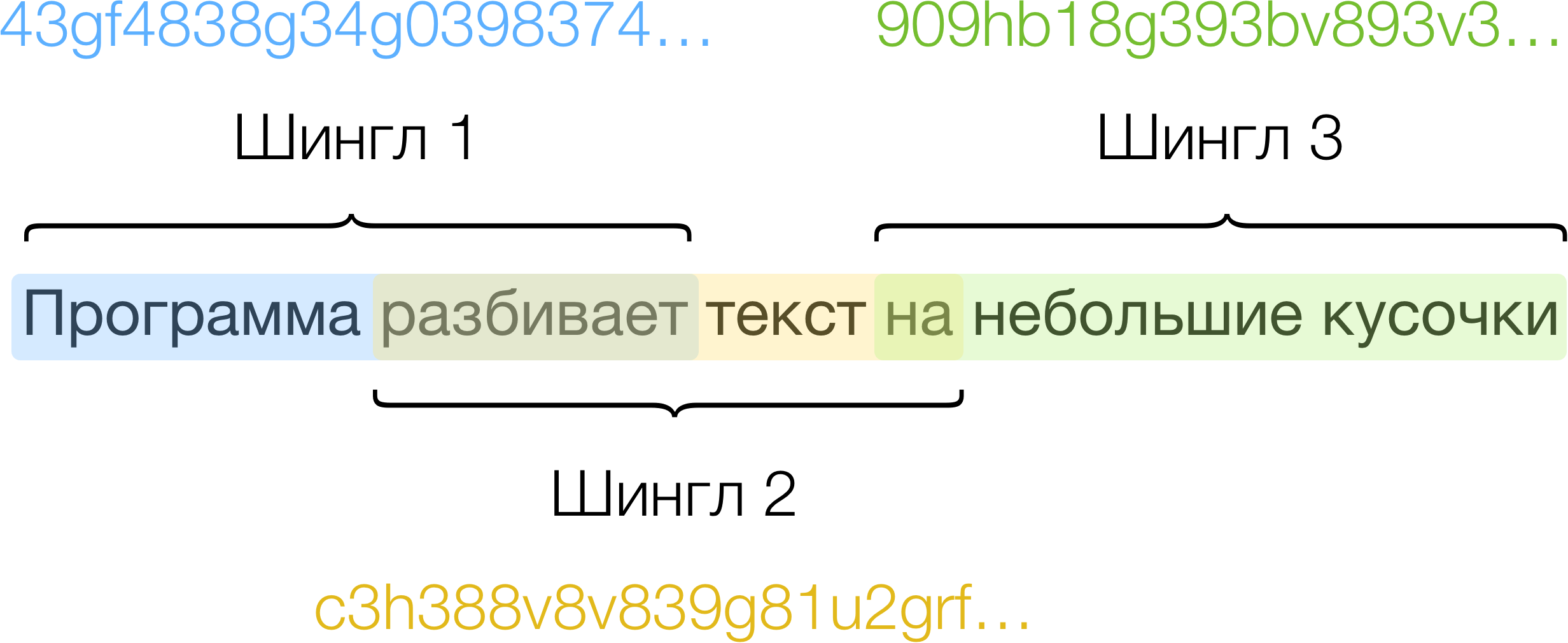


Рисунок 1.2 – разбиение на шинглы и их хэш

Алгоритм *сравнения текстов*.

Алгоритм выполняет текст будет разбивать тексты на слова. Все буквы, которые имеют одинаковое написание в английском и в русских языках будут приводится к одним буквам( Например a (английское) a (русское) ) Все буквы будут приведены к одному регистру. Будет происходить сравнение слов первого текста со словами второго текста. Если данное слово найдено оно будет выделяться в зависимости от настроек выбранных пользователем. Все неиспользованные слова второго текста будет выделены как “Не найденные”.

* 1. **Формирование требований к проектируемому программному средству**

Назначение данной разработки – создание текстового редактора, а также средства проверки текста на уникальность.

Входными данными является текст или два текста при сравнении текстов.

В своём проекте, в соответствии с заданием, я буду использовать следующие алгоритмы:

1. Упрощенный вариант алгоритма шинглов ( Он будет заключаться в следующем:
2. Текст будет разбиваться на шинглы по три слова
3. Шинглы будут сохраняться в массив
4. Все лишние пробелы будут удаляться и сохраняться в массив будет лишь один пробел между словами
5. Все буквы, которые имеют одинаковое написание в английском и в русских языках будут приводится к одним буквам( Например a (английское) a (русское) )
6. Все буквы будут приведены к одному регистру
7. Сравнение шинглов будет осуществляться соответственно с текстом, хранящимся в отдельном файле.

Конечным результатом проверки текста на уникальность будет процентное соотношение количества шинглов совпавших при сравнении к общему количеству шинглов (элементов массива)

1. Алгоритм сравнения текстов(пользователь сам выбирает, что он хочет получить в результате. В зависимости от выбора пользователя в результате будут высвечиваться изменения по сравнению со вторым текстом, или будут подсвечиваться символы, которые не изменились по сравнению с первым текстом). Алгоритм заключается в прохождении по символам первого текста и поиском соответствий(или несоответствий) этих символов во втором тексте

Задачей системы является редактирование текста.

Для редактирования текста будут реализованы функции:

1. Выравнивание текста по левому, правому краям или по центру
2. Изменение параметров шрифта
3. Размещение в одной строке не более N символов(N вводится пользователем)

Результатом редактирования текста будет текст.

1. **АНАЛИЗ ТРЕБОВАНИЙ К ПС И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ**
   1. **Описание функциональных требований к программному средству**

Для данного программного средства определены следующие функциональные требования:

1. Редактирование текстов
2. Возможность изменения шрифта (изменение его стиля, цвета, размера, применение жирного начертания к выделенному тексту, применение курсивного начертания к выделенному тексту, подчеркивание выделенного текста)
3. Выравнивание по левому и правому краям, и выравнивание по центру
4. Размещение в одной строке не более N символов(N вводится пользователем). Данная функция предполагает следующее:
5. Выбор пользователем количества символов, которое он хочет оставить в строках
6. Выделение текста который он хочет отредактировать
7. Проверка на введенное число(Если N < чем наибольшее слово в выделенном тексте выскакивает соответствующее сообщение)
8. ПС высчитывает количество символов и если N символ остановился на букве, то ПС идет влево по до “не буквы” и переносит символы с номерами большими N на новую строку)
9. Сравнение двух текстов предложенных пользователем (пользователь сам выбирает, что он хочет получить в результате. В зависимости от выбора пользователя в результате будут высвечиваться изменения по сравнению со вторым текстом, или будут подсвечиваться символы, которые не изменились по сравнению с первым текстом). Выделение изменений (или совпадений) будет происходить выбранным цветом.
10. Функция процентной оценки плагиата(уточнение: данная функция будет работать корректно только с текстом на русском языке(имеется в виду русские слова, символы могут быть записаны латиницей)). Выполнение данной функции описано в п. 1.3 (алгоритм шинглов)
11. Функция сохранения данного текста в файл(при сравнении текстов сохранится текст первого поля)
12. Функция загрузки текста в поле 1
13. Функция отмена предыдущего действия
14. Поиск подстроки в тексте
15. Замена подстроки в тексте (или всех таких же подстрок)
    1. **Спецификация функциональных требований**
16. При запуске ПС появляется поле для текста, над ним появляется панель управления с функциями(основные функции редактирования текста см. п.2.1 редактирование текста, а также кнопки с переходом на функции сравнения текстов и оценки уникальности текста)
17. При нажатии на кнопку сравнения текстов появляется второе поле для второго текста
18. После сравнения текстов чтобы оценить уникальность нужно вернуться на страницу редактирования текста(появится соответствующая кнопка)
19. Для отмены предыдущего действия необходимо перейти по верхней панели. Далее вкладка “Редактирование” и кнопка “Отменить”
20. Для смена цвета текста необходимо: выделить текст на панели управление выбрать цвет и нажать кнопку Change color.
21. Для смены других параметров текста выделяется текст и выбирается соответствующая нужной функции кнопка на панели управления
22. **ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА**

**3.1 Разработка хэш-таблицы для эффективного сравнения**

Для построения хэш-таблицы используется динамический массив,

каждым элементом которого является однонаправленный список.

При добавлении элемента в хэш-таблицу, высчитывается хэш по хэш- функции. Далее высчитывается его место в таблице, причем :

1. Если на этом месте уже есть элемент, проверяется не совпадают ли наши элементы, если не совпадают переходим к следующему элементу списка, пока не найдем свободное место. Если элемент совпадает с элементом списка просто выходим из функции добавления
2. Если на этом месте нет элемента просто заполняем его нашим элементом

В случае переполнения нашего динамического массива предусмотрена функция Rehash, в соответсвии с которой при заполнении массива на 75% сам массив будет увеличиваться и все элементы перезаписываться в талбицу.

## 3.2 Разработка алгоритмов сравнения двух текстов

Суть алгоритма заключается в нахождении наибольшей общей последовательности символов(или слов в зависимости от выбора пользователя).

Суть алгоритма приведена ниже

Рассмотрим задачу — для двух данных строк найти такую строку наибольшей длины, которая была бы подпоследовательностью каждой из них. Например, если A=«abcabaac», B=«baccbca» то у строк A и B них есть общая подпоследовательность длины 4, например, «acba» или «acbc».

Рассмотрим последние символы данных строк an-1 и bn-1. Если эти символы совпадают, то они обязательно войдут последними символами и в наибольшую общую подпоследовательность данных строк. Тогда можно свести задачу нахождения наибольшей общей подпоследовательности для строк  A =a0 … an-1  и B =b0 … bm-1  к задаче нахождения наибольшей общей подпоследовательности для строк, полученных отбрасыванием от данных строк последнего символа. Затем к ответу для «укороченных» строк добавим последние (равные) символы исходных строк и получим ответ для исходных строк.

Если же последние символы исходных строк не совпадают, то эти символы не могут одновременно входить в наибольшую общую подпоследовательность, поэтому можно один из них отбросить. Тогда задача сводится к нахождению наибольшей общей подпоследовательности для одного из двух случаев - для строк A =a0 … an-1  и B =b0 … bm-2     или для строк  A =a0 … an-2  и B =b0 … bm-1   .

Мы научились сводить задачу нахождения наибольшей общей подпоследовательности двух строк к меньшей задаче - нахождения наибольшей общей подпоследовательности для строк, полученных отбрасыванием последних символов от исходных строк, то есть для префиксов исходных строк. Для дальнейшего решения задачи будем следовать принципу построения решения при помощи динамического программирования.

Рассмотрим префикс  первой строки из  i символов:  A =a0 … ai-1  и префикс  второй строки из j символов B =b0 … bj-1   .Обозначим через  F(i,j) длину наибольшей общей подпоследовательности.

Теперь выпишем *рекуррентные соотношения*. Они зависят от того, совпадают ли последние символы рассматриваемых строк . Если ai-1 = bj-1  , то тогда F(i,j) = F(i-1,j-1)+1 - нужно решить задачу для строк, полученных отбрасыванием последних символов рассматриваемых строк и добавить 1 символ к ответу. В противном случае нужно рассмотреть два случая: F(i, j-1) и F(i-1, j), которые соответствуют отбрасыванию по одному символу от конца каждой из рассматриваемых строк. В этом случае F(I,j) = max(F(i, j-1) , F(i-1, j)).

Начальные значения функции  задаются просто: если одна из строк - пустая, то общая подпоследовательность также пустая, то есть имеет длину 0 .

Далее необходимо завести двумерный массив размером (n+1) и (m+1) заполнить его значениями по указанным рекуррентным соотношениям. Сначала весь массив заполним нулями (что задаст граничные значения), затем двумя вложенными циклами по  и по  заполним оставшуюся часть массив.

## 3.3 Разработка алгоритмов оценки уникальности

Для определения схожести 2-х документов необходимо:

1. канонизация текста;
2. разбиение на шинглы (делим текст по пробелам и создаем массивы);
3. вычисление хэшей шинглов;
4. сравнение, определение результата.

В моем случае канонизация состоит из:

1. очистка от лишних пробелов по бокам;
2. очистка от таких спец символов ' ” ', ' “ ', ' , ', ' . ' , ' : ', ' $ ', ' # ', ' ( ', ' ) '
3. Все буквы, которые имеют одинаковое написание в английском и в русских языках будут приводится к одним буквам( Например a (английское) a (русское)
4. Для надежного опознания слова нам не нужны гласные. Так что следующим этапом сворачивания у нас идёт удаление гласных.

## 3.4. Алгоритм разбиения на шинглы

Алгоритм заключается в прохождении по строке и выделении слов. После каждого выделенного слова он записывается в массив для шинглов и после этого если массив полон формируется шингл и записывается в хэш-таблицу. После этого все слова в массиве для шинглов сдвигаются вправо.

Схемы основных алгоритмов представлены в Приложении А.

**4. СОЗДАНИЕ ПРОГРАММНОГО СРЕДСТВА**

**4.1 Проектирование модулей программного средства**

В ПС используется 4 модуля:

1. Модуль Hash\_Table – модуль с реализацией хэш- таблицы
2. Модуль Redactor – модуль с основным функциям редактирования текста (основной модуль, показывается пользователю по умолчанию)
3. Модуль Compare – здесь находятся все алгоритмы сравнения текстов и все связанные с ними алгоритмы

Полный текст подпрограмм данных модулей представлен в Приложении Б.

**4.2 Разработка модулей программного средства**

Таблица 4.2.1 - Основные подпрограммы модулей (Hash\_Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрог-раммы** | **Описание** | **Заголовок подпрограммы** | **Имя парамет-ра и его тип** | **Назначение параметра** |
| GetHash | Получение хэша строки | function GetHash(const AKey: string): Integer; | A const AKey: string; | строковый ключ |
| CheckMemory | Проверка заполненности хэш-таблицы | function CheckMemory(): boolean; | - | - |
| MemoryAdd | Расширяет нашу таблицу(динамический массив) | procedure MemoryAdd(); | - | - |
| Add | Добавляет элемент в хэш -таблицу | procedure Add(const AKey: string; AValue: Integer); | const AKey: string; AValue: Integer | Ключ и его значение |
| Get | Получение номера элемента в массиве по хэшу | function Get(const AKey: string): Integer; | const AKey: string | строковый ключ |
| PutOrUpdate | Добавляет и проверяет нужно ли расширять таблицу | procedure PutOrUpdate(const AKey: string; AValue: Integer); | const AKey: string; AValue: Integer | Ключ и его значение |
| ContainsKEy | Проверяет содержит ли данный ключ хэш таюлица | function ContainsKEy(const AKey: string): boolean; | const AKey: string | строковый ключ |

Таблица 4.2.2 - Основные подпрограммы модулей (Redactor)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрог-раммы** | **Описание** | **Заголовок подпрограммы** | **Имя парамет-ра и его тип** | **Назначение параметра** |
| OpenBtnClick | Обработчик нажатия кнопки открыть файл | procedure OpenBtnClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| SaveASClick | Обработчик нажатия кнопки Сохранить как файл | procedure SaveASClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| BoldButton1Click | Обработчик нажатия кнопки сделать жирным шрифты | procedure BoldButton1Click(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| ItalicButton1Click | Обработчик нажатия кнопки сделать курсивом шрифты | procedure ItalicButton1Click(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| UnderlineButton1Click | Обработчик нажатия кнопки сделать подчеркнутым шрифты | procedure UnderlineButton1Click(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| LeftJustifyButton1Click | Обработчик нажатия кнопки выровнять по левому краю | procedure LeftJustifyButton1Click(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| CenterJustifyButton1Click | Обработчик нажатия кнопки выровнять по центру | procedure CenterJustifyButton1Click(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| RightJustifyButton1Click | Обработчик нажатия кнопки выровнять по правому краю | Procedure RightJustifyButton1Click(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| FontButton1Click | Обработчик нажатия кнопки открыть диалоговое окно со шрифтами | procedure FontButton1Click(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| UndoClick | Обработчик нажатия кнопки отменить | procedure UndoClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |

Продолжение таблицы 4.2.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SignsNumberClick | Обработчик нажатия на кнопку для формирования строк по заданному количеству символов | procedure SignsNumberClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| FormCreate | Обработчик создания формы | procedure FormCreate(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| CompareUnitButtonClick | Обработчик нажатия на кнопку для перехода к форме сравнения | procedure CompareUnitButtonClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| ColorBox1Select | Обработчик нажатия на кнопку для выбора цвета шрифта | procedure ColorBox1Select(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| CreateNewClick | Обработчик нажатия кнопки создать новый файл | procedure CreateNewClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| ExitButtonClick | Обработчик нажатия кнопки выйти из приложения | procedure ExitButtonClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| SaveClick | Обработчик нажатия кнопки Сохранить файл | procedure SaveClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| CopyBtnClick | Обработчик нажатия кнопки копировать выделенное | procedure CopyBtnClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| InsertBtnClick | Обработчик нажатия кнопки вставить текст из буфера | procedure InsertBtnClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| SelectAllBtnClick | Обработчик нажатия кнопки выделить весь текст | procedure SelectAllBtnClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| procedure CutBtnClick(Sender: TObject); | Обработчик нажатия кнопки вырезать текст | procedure CutBtnClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| procedure SAveFile; | Сохраняет файл | procedure SAveFile; | - | - |

Продолжение таблицы 4.2.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FontSizeEdit1KeyPress | Обработчик нажатия клавиши увеличить шрифт | procedure FontSizeEdit1KeyPress(Sender: TObject; var Key: Char); | Sender: TObject | Объект, который сгенерировал событие |
| UpDown1Changing | Обработчик нажатия кнопки изменения размера шрифта | procedure UpDown1Changing(Sender: TObject; var AllowChange: Boolean); | Sender: TObject | Объект, который сгенерировал событие |
| UpDown2Changing | Обработчик нажатия кнопки изменения количества символов строке | procedure UpDown2Changing(Sender: TObject; var AllowChange: Boolean); | Sender: TObject; var AllowChange: Boolean); | Объект, который сгенерировал событие |
| NumberSignsEdit1KeyPress | Обработчик нажатия кнопки формирования строк заданной длины | procedure NumberSignsEdit1KeyPress(Sender: TObject; var Key: Char); | Sender: TObject; var Key: Char | Объект, который сгенерировал событие |
| FindButtonClick | Обработчик нажатия кнопки поиска символов | procedure FindButtonClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| procedure FindEditKeyPress | Обработчик нажатия клавиши поиска символов | procedure FindEditKeyPress(Sender: TObject; var Key: Char); | Sender: TObject; var Key: Char | Объект, который сгенерировал событие |
| Splitter1Moved | Обработчик движения сплиттера для регулировки отступа | procedure Splitter1Moved(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| FindMax; | Поиск длины самого длинного слова | function FindMax(str: string): integer; | str: string | Строка для поиска |

Таблица 4.2.3 - Основные подпрограммы модулей (Compare)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| procedure CompareText | Процедура сравнения текстов | procedure CompareText(mode: integer); | mode: integer | Режим вывода |
| CompareTextsClick | Обработчик нажатия кнопки сравнения текстов | procedure CompareTextsClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| ShinglesComboSelect | Обработчик нажатия combobox для определения шинглов | procedure ShinglesComboSelect(Sender: TObject); | (Sender: TObject); | Объект, который сгенерировал событие |
| GoBAckbuttonClick | Обработчик нажатия кнопки вернуться в текстовый редактор | procedure GoBAckbuttonClick(Sender: TObject); | Sender: TObject; | Объект, который сгенерировал событие |
| ReplaceBtnClick | Обработчик нажатия кнопки поменять тексты местами | procedure ReplaceBtnClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| PanelU3ButtonsResize | Обработчик изменения панели | PanelU3ButtonsResize(Sender: TObject); | Sender: TObject; | Объект, который сгенерировал событие |
| signsRadioButtonClick | Обработчик нажатия кнопки выбора режима вывода результатов | procedure signsRadioButtonClick(Sender: TObject); | Sender: TObject | Объект, который сгенерировал событие |
| WordsRadioButtonClick | Обработчик нажатия кнопки выбора режима вывода результатов | procedure WordsRadioButtonClick(Sender: TObject); | Sender: TObject | Строка для поиска |
| OneWord | Разделяет строку на слова или символы в зависимости от режима | function OneWord(var str1: string; mode: integer): ArrOfStr; | var str1: string; mode: integer | Строка и режим вывода |
| MarkDiffer | Разметка удаляем(добавляемых) слов (строк) | function MarkDiffer(strNUms1: ArrOfStr; LCS: ArrOfStr): ArrOfStr; | strNUms1: ArrOfStr; LCS: ArrOfStr | Массив слов и наибольшая общая подпоследовательность |
| MaxSubLine | Наибольшая общая подпоследовательность двух массивов | function MaxSubLine(strNUms1: ArrOfStr; strNUms2: ArrOfStr): ArrOfStr; | strNUms1: ArrOfStr; strNUms2: ArrOfStr | Массив слов первого текста и второго |

Продолжение таблицы 4.2.3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| lowCase | Приведение к нижнему регистру | function lowCase(var str1: string): string; | var str1: string | Строка |
| Shingles | Разбиение текста на шинглы | function Shingles(str1: string; n: integer): THAshTAble; | str1: string; n: integer | Сторка и количество слов в шингле |
| Count | Подсчет процента схожести текстов | function Count(HashTable: THAshTAble; str1: string; n: integer): integer; | HashTable: THAshTAble; str1: string; n: integer | Таблица слов текста сравнения и количество слов в шингле |

**5. ТЕСТИРОВАНИЕ, ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ**

**5.1. Тестирование работоспособности ПС**

Таблица 5.1 - Тестирование работоспособности ПС

|  |  |  |  |
| --- | --- | --- | --- |
| Номер  теста | Тестируемая функциональность | Ожидаемый результат | Полученный результат |
| 1 | Сохранение текста редактирования | Текст успешно сохранен | Тест пройден |
| 2 | Открытие файла для редактирования | Файл успешно открывается | Тест пройден |
| 3 | Сохранить текст в файл из которого был открыт текст(перезапись файла) | Файл успешно перезаписан | Тест пройден |
| 4 | Создание нового текста | Очищение панели редактирования | Тест пройден |
| 5 | При создании нового текста появление окна, если файл не был сохранен | Появление окна сохранения  С выбором сохранять изменения в файл или нет или отменить действие | Тест пройден |
| 6 | Копирование и вырезка текста на горячие клавиши, на кнопки | Копирование текста в буфер | Тест пройден |
| 7 | Вставка текста из буфера на горячие клавиши, на кнопки | Вставка текста из буфера | Тест пройден |

|  |  |  |  |
| --- | --- | --- | --- |
| 8  Продолжение таблицы 5.1 | Выделение всего текста и отмена действий | Выделение текст и отмена | Тест пройден |
| 9 | Редактирование отступа в тексте при помощи ползунка | Успешная регулировка отступов | Тест пройден |
| 10 | Формирование строк с заданным количеством символов | Отредактированный в соответствии с функцией текст | Тест пройден |
| 11 | Изменение параметров шрифта | Изменение выделенного текста | Тест пройден |
| 12 | Поиск и выделение слова в тексте | Выделение найденного текста  Или появление сообщения о не найденном тексте | Тест пройден |
| 13 | Выход из программы | Закрытие окон | Тест пройден |
| 14 | Переход и возвращение на окна редактирования и сравнения | Переход между формами | Тест пройден |
| 15 | Смена текста местами (в модуле сравнения) | Текст поменялся местами | Тест пройден |
| 16 | Сравнение двух текстов при режиме ‘Символы’ | В результате текст с подсвеченными результатами текста | Тест пройден |
| 17 | Сохранение результатов сравнения | Файл с сохраненными данными | Тест пройден |
| 18 | Оценка схожести текстов | Процент схожести текстов | Тест пройден |

|  |  |  |  |
| --- | --- | --- | --- |
| 19  Продолжение таблицы 5.1 | Сравнение двух текстов при режиме ‘Слово’ | В результате текст с подсвеченными результатами текста | Тест пройден |

**5.2. Анализ полученных результатов**

Тест

|  |  |
| --- | --- |
| Текст 1 | Текст 2 |
| Главный герой — 27-летний Илья Горюнов, семь лет отсидевший в тюрьме по ложному обвинению в распространении наркотиков. Когда Илья выходит на свободу, он понимает, что прежняя жизнь, по которой он тосковал, разрушена, и вернуться к ней он больше не сможет. Хотя он не собирался мстить человеку, который отправил его в тюрьму, другого выхода теперь нет. Встретившись лицом к лицу со своим обидчиком, Илья совершает необдуманный поступок, после которого главный герой получает доступ к смартфону Петра и Ивана, а с ним и к жизни молодого человека — его фотографиям и видео, перепискам с родителями и девушкой Ниной, к странным | Главный герой — 7-летний Илья Горюнов, семь лет отсидевший в тюрьме по ложному обвинению в распространении наркотиков. Когда Илья выходит на свободу, он понимает, что прежняя жизнь, по которой он тосковал, разрушена, и вернуться к ней он больше не сможет. Хотя он не собирался мстить человеку, который отправил его в тюрьмы, другого выхода теперь нет. Встретившись лицом к лицу со своими обидчиками, Илья совершает необдуманный поступок, после которого главный герой получает доступ к смартфону Петра, а с ним и к жизни молодого человека — его фотографиям и видео, перепискам с родителями и девушкой Ниной, к странным |

|  |
| --- |
| **Результат**  Главный герой - ~~2~~7-летний Илья Горюнов, семь лет отсидевший в тюрьме по ложному обвинению в распространении наркотиков. Когда Илья выходит на свободу, он понимает, что прежняя жизнь, по которой он тосковал, разрушена, и вернуться к ней он больше не сможет. Хотя он не собирался мстить человеку, который отправил его в тюрьм~~у~~ы, другого выхода теперь нет. Встретившись лицом к лицу со своими обидчик~~о~~ами, Илья совершает необдуманный поступок, после которого главный герой получает доступ к смартфону Петра ~~и Ивана~~, а с ним и к жизни молодого человека - его фотографиям и видео, перепискам с родителями и девушкой Ниной, к странным |

Результат схожести двух текстов

Процентная схожесть первого текста со вторым (%) - 97,61

Разница (%) 2,39

Общий (символы) 613

Разница (символы) 15

**6.** **РУКОВОДСТВО ПО УСТАНОВКЕ И ЭКСПЛУАТАЦИИ**

Для запуска программа вам необходимо запустить TextRedactor.exe.

После запуска программы появляется само окно редактора. Все функции доступные редакторы описаны в разделе 2 данной пояснительной записки.

Для сравнения текстов на плагиат в режиме Редактора необходимо нажать кнопку ”Проверить уникальность”. Сравнения идет с заранее имеющимся текстом из файла ForPlagiat.rtf.

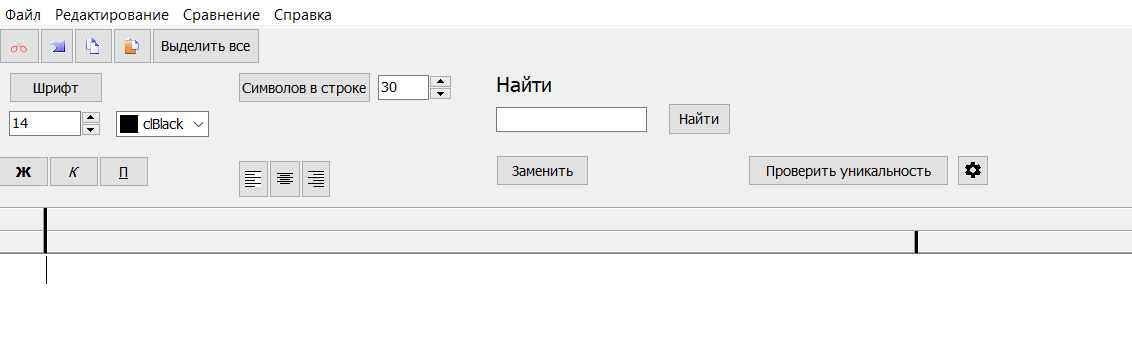


Рисунок 6.1 – Интерфейс редактора

На вкладке Сравнение текстов вы можете увидеть результат сравнения ваших текстов после нажатия кнопки “Сравнения текстов”. Данная функция работает в двух режимах: с выделением различий посимвольно и по словам.

По умолчанию выбран режим посимвольного сравнения.

Также на вкладке сравнения можно оценить на сколько схожи сами тексты. Результат сравнения можно сохранить в файл с расширением rtf.

Все файлы сохраняются с расширением rtf. Также можно загрузить файлы с тем же расширением.

Регулирование отступов перед началом строки происходит при помощи “Линейки” расположенной прямо над текстовым полем редактора.

Функция “Заменить”. После нажатия кнопки выпадает окно, в данном окне необходимо ввести элемент, который нужно заменить, и элемент, на который нужно заменить. Также доступна замена только в области выделенного текста

В редакторе используются общепринятые горячие клавиши:

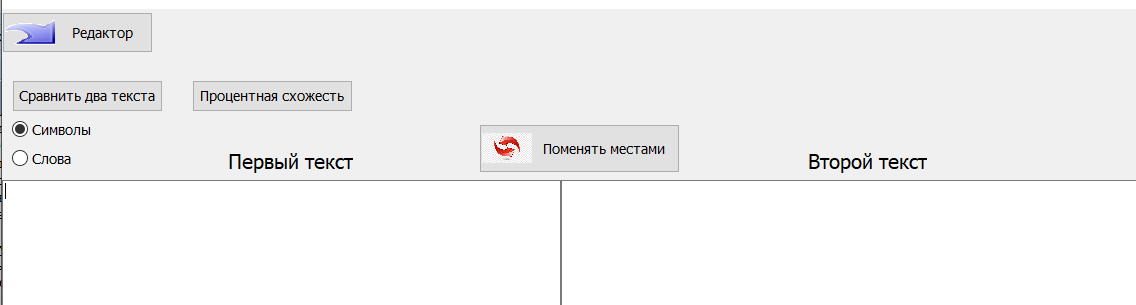


Рисунок 6.2 – Интерфейс вкладки сравнения

# **ЗАКЛЮЧЕНИЕ**

В процессе выполнения курсового проекта была проанализирована предметная область, рассмотрены существующие аналоги и выявлены их преимущества и недостатки. В качестве языка разработки использовался Delphi.

На этапе проектирования были разработаны блок-схемы алгоритмов. В соответствии с поставленной целью было разработано программное средство для

"Программное средство редактирования, сравнения текстов и процентной оценки плагиата".

Сравнение текстов на схожесть необходимо, когда вам нужно создать много текстов на одну тему, но избежать повторений. Данное средство также позволяет сравнить тексты и оценить их уникальность.

Данное средство позволяет просматривать содержимое текстовых файлов и производить над ними различные действия: вставку, удаление и копирование текста, контекстный поиск и замену и т. п.

Проведено тестирование работоспособности разработанной программной части. Поставленная цель была выполнена в полном объеме, работоспособность подтверждена тестированием программного средства.

В заключении можно отметить, что изучение методов оценки плагиата, редактирования текстов и сравнения текстов является важной задачей для всех, кто занимается написанием и проверкой научных, технических и других текстов. Однако, необходимо помнить, что программные средства не могут полностью заменить ручной анализ и оценку текстов, поэтому важно оставлять место для экспертного мнения и профессиональной оценки.

# **Список использованной литературы**

[1] Хеш-таблица. [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/Хеш-таблица>.

[2] Фундаментальные алгоритмы и структуры данных в Delphi [Электронный ресурс] / Бакнелл Д. М. – Режим доступа: <https://it.wikireading.ru/34970>.

[3] Алгоритм поиска наибольшей общей подпоследовательности [Электронный ресурс] – Режим доступа: <https://foxford.ru/wiki/informatika/naibolshaya-obschaya-podposledovatelnost>

[4] Часть 1. Программирование на языке Delphi Учебное пособие Авторы: А.Н. Вальвачев, К.А. Сурков, Д.А. Сурков, Ю.М. Четырько [Электронный ресурс] – Режим доступа: <https://www.bsuir.by/m/12_100229_1_90135.pdf>

[5] Алгоритм разбиения на шинглы [Электронный ресурс] – Режим доступа: https://habr.com/ru/companies/antiplagiat/articles/445952/

# **ПРИЛОЖЕНИЕ А**

**(обязательное)**

**Схемы алгоритмов программы**

# 

Рисунок A.1 – Блок-схема программы

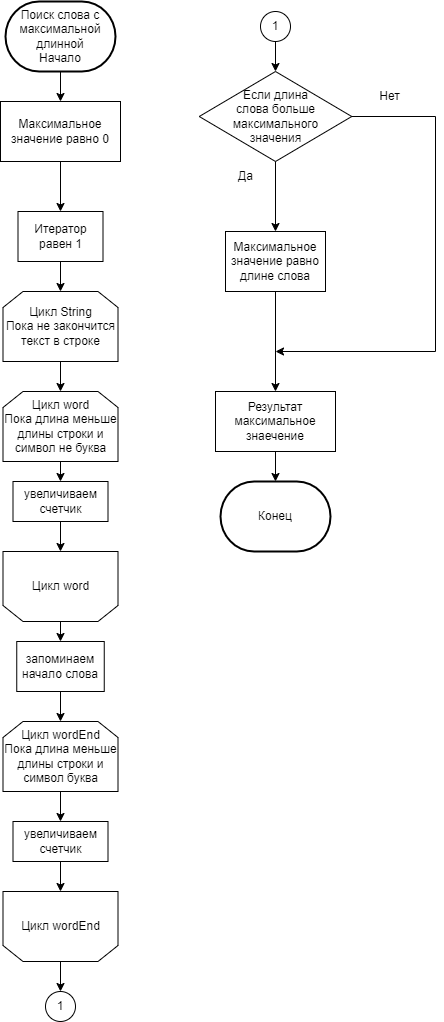


Рисунок А.2 - Нахождение максимальной длины слова в выделенном тексте

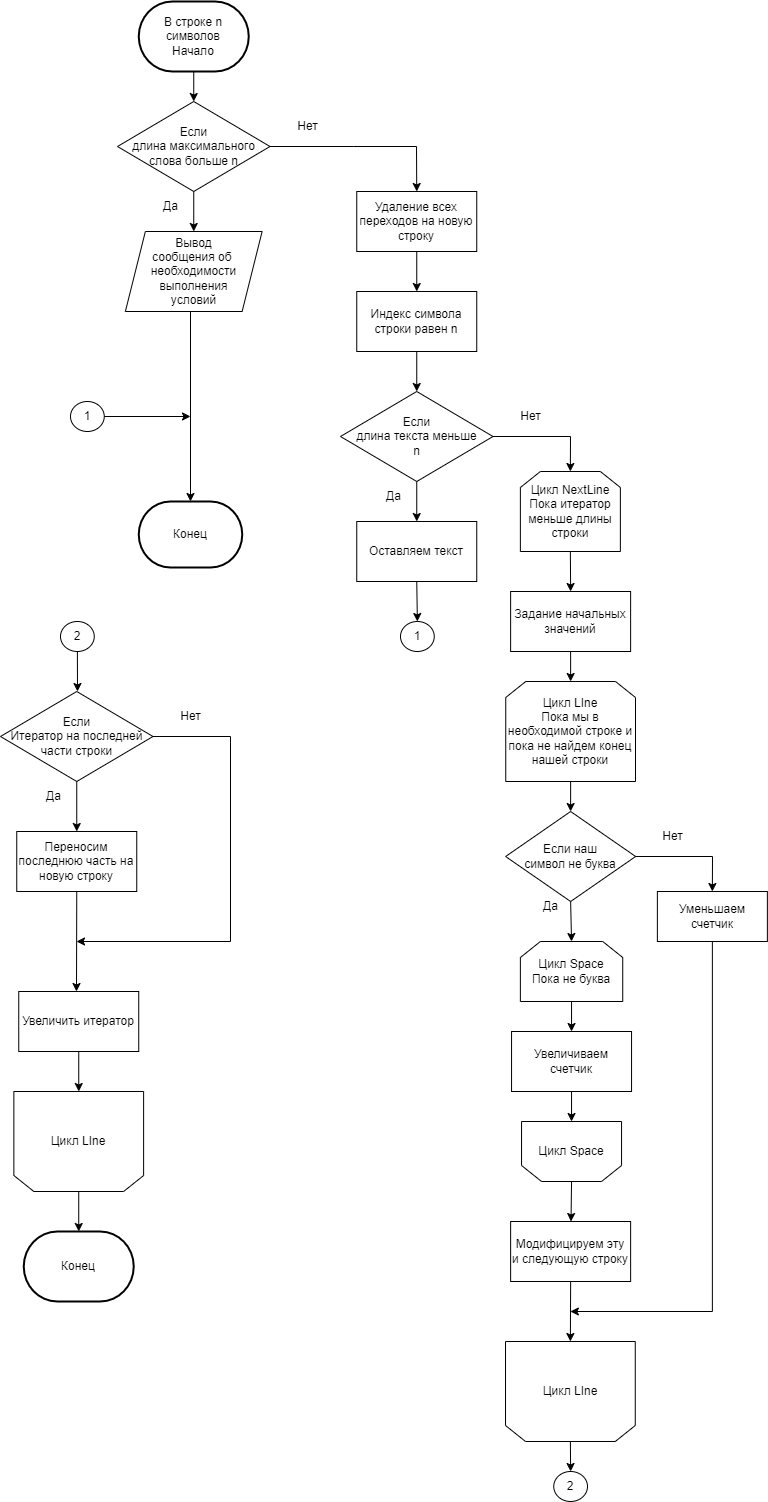


Рисунок А.3 - Распределение N количество символов в строках



Рисунок А.4 - Алгоритм оценки уникальности текста

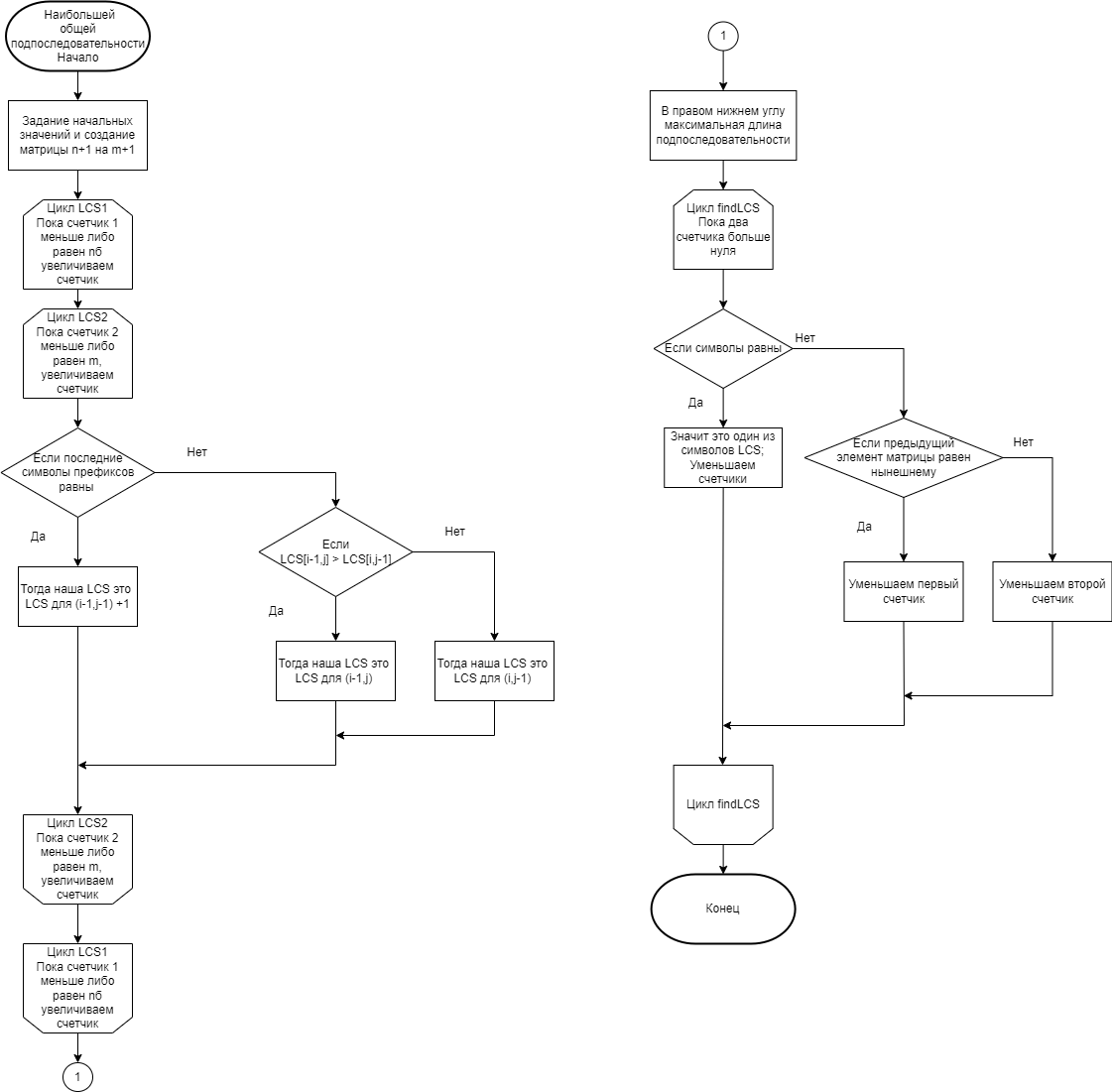


Рисунок А.5 - Нахождение наибольшей общей подпоследовательности

# **ПРИЛОЖЕНИЕ Б**

**(обязательное)**

**Полный текст подпрограмм данных модулей представлен**

unit Redactor;//Модуль Redactor

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus, Vcl.StdCtrls, Vcl.ComCtrls,

Vcl.ExtCtrls, System.Generics.Collections, Vcl.Buttons, HashTable, Compare,

Window, ReplaceUnit, Math,

System.ImageList, Vcl.ImgList, Vcl.ToolWin, System.Actions, Vcl.ActnList;

type

TForm2 = class(TForm)

MainMenu1: TMainMenu;

FileProp: TMenuItem;

OpenBtn: TMenuItem;

SaveAS: TMenuItem;

ExitButton: TMenuItem;

N5: TMenuItem;

OpenDialog1: TOpenDialog;

SaveDialog1: TSaveDialog;

FontDialog1: TFontDialog;

Undo: TMenuItem;

CompareUnitButton: TMenuItem;

panelForFunctions: TPanel;

resultShingles: TLabel;

ColorBox1: TColorBox;

FontSizeEdit1: TEdit;

UpDown1: TUpDown;

FontButton1: TButton;

NumberSignsEdit1: TEdit;

UpDown2: TUpDown;

SignsNumber: TButton;

ToolBar1: TToolBar;

BoldButton1: TButton;

ItalicButton1: TButton;

UnderlineButton1: TButton;

ImageListBtn: TImageList;

ToolBar3: TToolBar;

LeftJustifyButton1: TButton;

CenterJustifyButton1: TButton;

RightJustifyButton1: TButton;

CreateNew: TMenuItem;

Save: TMenuItem;

CopyBtn: TMenuItem;

InsertBtn: TMenuItem;

CutBtn: TMenuItem;

SelectAllBtn: TMenuItem;

FindLabel: TLabel;

FindEdit: TEdit;

FindButton: TButton;

ToolBar2: TToolBar;

ActionList1: TActionList;

Action1: TAction;

UndoButton1: TButton;

CopyButton1: TButton;

PasteButton: TButton;

CutButton: TButton;

SelectButton: TButton;

RichEdit1: TRichEdit;

RulerPanel: TPanel;

SizePanel2: TPanel;

sizepanel: TPanel;

Splitter1: TSplitter;

LeftIndentPanel: TPanel;

Splitter2: TSplitter;

SizeRightPanel: TPanel;

SizePanelLeft: TPanel;

RightIndentPanel: TPanel;

RightSplitter: TSplitter;

ReplaceButton: TButton;

PlagButton: TButton;

ForFileRichEdit: TRichEdit;

SettingsButton: TButton;

ComboShingles: TComboBox;

procedure OpenBtnClick(Sender: TObject);

procedure SaveASClick(Sender: TObject);

procedure BoldButton1Click(Sender: TObject);

procedure ItalicButton1Click(Sender: TObject);

procedure UnderlineButton1Click(Sender: TObject);

procedure LeftJustifyButton1Click(Sender: TObject);

procedure CenterJustifyButton1Click(Sender: TObject);

procedure RightJustifyButton1Click(Sender: TObject);

procedure FontButton1Click(Sender: TObject);

procedure UndoClick(Sender: TObject);

procedure SignsNumberClick(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure CompareUnitButtonClick(Sender: TObject);

procedure ColorBox1Select(Sender: TObject);

procedure CreateNewClick(Sender: TObject);

procedure ExitButtonClick(Sender: TObject);

procedure SaveClick(Sender: TObject);

procedure CopyBtnClick(Sender: TObject);

procedure InsertBtnClick(Sender: TObject);

procedure SelectAllBtnClick(Sender: TObject);

procedure CutBtnClick(Sender: TObject);

procedure SAveFile;

procedure FontSizeEdit1KeyPress(Sender: TObject; var Key: Char);

procedure UpDown1Changing(Sender: TObject; var AllowChange: Boolean);

procedure NumberSignsEdit1KeyPress(Sender: TObject; var Key: Char);

procedure FindButtonClick(Sender: TObject);

procedure FindEditKeyPress(Sender: TObject; var Key: Char);

procedure Splitter1Moved(Sender: TObject);

procedure RichEdit1SelectionChange(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure ReplaceButtonClick(Sender: TObject);

procedure PlagButtonClick(Sender: TObject);

procedure SettingsButtonClick(Sender: TObject);

procedure ComboShinglesSelect(Sender: TObject);

private

F: string;

LeftInden: integer;

firstInden: integer;

PlagFile: string;

public

{ Public declarations }

end;

Type

DinMAss = array of integer;

ArrOfStr = array of string;

ArrOfNUms = array of integer;

Matrix = array of array of integer;

var

Form2: TForm2;

implementation

{$R \*.dfm}

const

Special = ['"', '"', ',', '.', ':', '$', '#', '(', ')', ' ', '-', ':', ';'];

procedure TForm2.BoldButton1Click(Sender: TObject);

begin

if not(fsbold in RichEdit1.SelAttributes.Style) then

begin

RichEdit1.SelAttributes.Style := RichEdit1.SelAttributes.Style + [fsbold];

BoldButton1.ImageIndex := 1;

end

else

begin

RichEdit1.SelAttributes.Style := RichEdit1.SelAttributes.Style - [fsbold];

BoldButton1.ImageIndex := -1;

end;

end;

procedure TForm2.UnderlineButton1Click(Sender: TObject);

begin

if not(fsUnderLine in RichEdit1.SelAttributes.Style) then

begin

RichEdit1.SelAttributes.Style := RichEdit1.SelAttributes.Style +

[fsUnderLine];

UnderlineButton1.ImageIndex := 1;

end

else

begin

RichEdit1.SelAttributes.Style := RichEdit1.SelAttributes.Style -

[fsUnderLine];

UnderlineButton1.ImageIndex := -1;

end;

end;

procedure TForm2.CenterJustifyButton1Click(Sender: TObject);

begin

RichEdit1.Paragraph.Alignment := taCenter;

end;

procedure TForm2.FindButtonClick(Sender: TObject);

var

posit: integer;

s1, pattern: string;

begin

s1 := RichEdit1.Text;

pattern := FindEdit.Text;

posit := RichEdit1.FindText(pattern, 1, length(s1), [stMatchCase]);

if posit <> -1 then

begin

RichEdit1.SelStart := posit;

RichEdit1.SelLength := length(FindEdit.Text);

RichEdit1.SetFocus;

end

else

ShowMessage('Данных слов не найдено');

end;

procedure TForm2.FindEditKeyPress(Sender: TObject; var Key: Char);

begin

if Key = #13 then

begin

Key := #0;

FindButtonClick(Sender);

end;

end;

procedure TForm2.FontButton1Click(Sender: TObject);

begin

if FontDialog1.Execute then

RichEdit1.SelAttributes.Assign(FontDialog1.Font);

RichEdit1.SetFocus;

end;

procedure TForm2.FontSizeEdit1KeyPress(Sender: TObject; var Key: Char);

begin

if Key = #13 then

begin

Key := #0;

RichEdit1.SelAttributes.Size := StrToInt(FontSizeEdit1.Text);

end;

end;

procedure TForm2.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

if RichEdit1.Modified then

begin

Form4.ShowModal;

CanClose := true;

if Form4.CLick = 2 then

SAveFile;

if Form4.CLick = 3 then

begin

CanClose := false;

end;

end

end;

procedure TForm2.FormCreate(Sender: TObject);

begin

RichEdit1.Font.Size := 14;

firstInden := Splitter1.Left;

LeftInden := sizepanel.Width - Splitter2.Left;

RichEdit1.Paragraph.FirstIndent := 3 \* firstInden div 5;

RichEdit1.Paragraph.LeftIndent := -3 \* LeftInden div 5;

RichEdit1.Paragraph.RightIndent := 3 \* (RichEdit1.Width - RightSplitter.Left)

div 5 + 1;

OpenDialog1.Filter := 'RTF Files |\*.rtf';

SaveDialog1.Filter := 'RTF Files |\*.rtf';

RichEdit1.Modified := false;

PlagFile := ExtractFilePath(Application.ExeName) + 'ForPlagiat.rtf';

end;

procedure TForm2.InsertBtnClick(Sender: TObject);

begin

RichEdit1.PasteFromClipboard;

end;

procedure TForm2.ItalicButton1Click(Sender: TObject);

begin

if not(fsItalic in RichEdit1.SelAttributes.Style) then

begin

RichEdit1.SelAttributes.Style := RichEdit1.SelAttributes.Style + [fsItalic];

ItalicButton1.ImageIndex := 1;

end

else

begin

RichEdit1.SelAttributes.Style := RichEdit1.SelAttributes.Style - [fsItalic];

ItalicButton1.ImageIndex := -1;

end;

end;

procedure TForm2.LeftJustifyButton1Click(Sender: TObject);

begin

RichEdit1.Paragraph.Alignment := taLEftjustify;

end;

procedure TForm2.NumberSignsEdit1KeyPress(Sender: TObject; var Key: Char);

begin

if Key = #13 then

begin

Key := #0;

SignsNumberClick(Sender);

end;

end;

procedure TForm2.OpenBtnClick(Sender: TObject);

begin

if OpenDialog1.Execute then

begin

RichEdit1.Lines.LoadFromFile(OpenDialog1.Filename);

F := OpenDialog1.Filename;

end;

RichEdit1.Modified := false;

end;

procedure TForm2.PlagButtonClick(Sender: TObject);

var

res: extended;

str: ansistring;

begin

ForFileRichEdit.Lines.LoadFromFile(PlagFile);

str := ForFileRichEdit.Text;

res := Form3.Count(Form3.Shingles(RichEdit1.Text, Form3.NumberShingles), str,

Form3.NumberShingles);

ShowMessage(FloatToStr(RoundTo(100 - res, -2)));

end;

procedure TForm2.SaveASClick(Sender: TObject);

begin

if SaveDialog1.Execute then

begin

RichEdit1.Lines.SaveToFile(SaveDialog1.Filename + 'rtf');

F := SaveDialog1.Filename + 'rtf';

RichEdit1.Modified := false;

end;

end;

procedure TForm2.SAveFile;

begin

if F = '' then

begin

if SaveDialog1.Execute then

begin

RichEdit1.Lines.SaveToFile(SaveDialog1.Filename + '.rtf');

F := SaveDialog1.Filename + '.rtf';

RichEdit1.Modified := false;

end

else

begin

Form4.CLick := 3;

end;

end

else

begin

RichEdit1.Lines.SaveToFile(F);

RichEdit1.Modified := false;

end;

end;

procedure TForm2.SaveClick(Sender: TObject);

begin

SAveFile;

end;

procedure TForm2.SelectAllBtnClick(Sender: TObject);

begin

RichEdit1.SelectALl;

RichEdit1.SetFocus;

end;

procedure TForm2.SettingsButtonClick(Sender: TObject);

begin

ComboShingles.visible := not ComboShingles.visible;

end;

procedure TForm2.ExitButtonClick(Sender: TObject);

begin

Form2.CLose;

ReplaceForm.Close;

Form3.CLose;

Form4.CLose;

end;

procedure TForm2.UndoClick(Sender: TObject);

begin

RichEdit1.Undo;

end;

procedure TForm2.UpDown1Changing(Sender: TObject; var AllowChange: Boolean);

begin

RichEdit1.SelAttributes.Size := StrToInt(FontSizeEdit1.Text);

end;

procedure TForm2.ComboShinglesSelect(Sender: TObject);

begin

Form3.NumberShingles := StrToInt(ComboShingles.Text);

end;

procedure TForm2.CompareUnitButtonClick(Sender: TObject);

begin

Form3.Left := Self.Left;

Form3.Top := Self.Top;

Form3.Width := Width;

Form3.Height := Height;

Form3.RichEdit1.Clear;

Form3.RichEdit2.Clear;

Form3.RichEdit3.Clear;

Form3.ShowModal;

end;

procedure TForm2.CopyBtnClick(Sender: TObject);

begin

RichEdit1.CopyToClipboard;

end;

procedure TForm2.ColorBox1Select(Sender: TObject);

begin

RichEdit1.SelAttributes.Color := ColorBox1.Selected;

RichEdit1.SetFocus;

end;

procedure TForm2.CreateNewClick(Sender: TObject);

begin

if RichEdit1.Modified then

begin

Form4.ShowModal;

if Form4.CLick = 2 then

SAveFile;

end;

RichEdit1.Clear;

F := '';

end;

procedure TForm2.CutBtnClick(Sender: TObject);

begin

RichEdit1.CutToClipboard;

end;

procedure TForm2.ReplaceButtonClick(Sender: TObject);

var

start, len, posit: integer;

begin

start := RichEdit1.SelStart;

len := RichEdit1.SelLength;

ReplaceForm.ClearEdits;

ReplaceForm.ShowModal;

posit := 0;

if ReplaceForm.ModalResult = mrYes then

begin

if not ReplaceForm.Check then

begin

start := 1;

len := length(RichEdit1.Text);

end;

while posit <> -1 do

begin

posit := RichEdit1.FindText(ReplaceForm.Find, start, len, [stMatchCase]);

if posit <> -1 then

begin

RichEdit1.SelStart := posit;

RichEdit1.SelLength := length(ReplaceForm.Find);

RichEdit1.SelText := ReplaceForm.ReplaceStr;

end;

Inc(len, length(ReplaceForm.ReplaceStr) - length(ReplaceForm.Find));

start := posit + length(ReplaceForm.ReplaceStr);

end;

end;

end;

procedure TForm2.RichEdit1SelectionChange(Sender: TObject);

begin

if fsbold in RichEdit1.SelAttributes.Style then

BoldButton1.ImageIndex := 1

else

BoldButton1.ImageIndex := -1;

if fsUnderLine in RichEdit1.SelAttributes.Style then

UnderlineButton1.ImageIndex := 1

else

UnderlineButton1.ImageIndex := -1;

if fsItalic in RichEdit1.SelAttributes.Style then

ItalicButton1.ImageIndex := 1

else

ItalicButton1.ImageIndex := -1;

end;

procedure TForm2.RightJustifyButton1Click(Sender: TObject);

begin

RichEdit1.Paragraph.Alignment := taRightjustify;

end;

function FindMax(str: string): integer;

var

j, start: integer;

max: integer;

begin

max := 0;

str := StringReplace(str, #13, '', [rfReplaceall]);

j := 1;

while j < length(str) do

begin

while (j <= length(str)) and (str[j] in Special) do

Inc(j);

start := j;

while (j <= length(str)) and (not(str[j] in Special)) do

Inc(j);

if max < j - start then

max := j - start;

end;

FindMax := max;

end;

procedure TForm2.SignsNumberClick(Sender: TObject);

var

i, need, j, long, start, pos: integer;

fl: Boolean;

begin

need := StrToInt(NumberSignsEdit1.Text);

if (FindMax(RichEdit1.SelText) > need) then

ShowMessage

('Количество символов в строке должно быть не меньше чем длина наибольшего слова')

else if length(RichEdit1.SelText) = 0 then

ShowMessage('Выделите нужный текст')

else if (length(RichEdit1.SelText) < 2000000) then

begin

RichEdit1.Lines.BeginUpdate;

start := RichEdit1.SelStart;

pos := 0;

for i := 0 to high(RichEdit1.Text) - 1 do

if (RichEdit1.Text[i] = #13) then

Inc(pos);

long := length(RichEdit1.SelText);

RichEdit1.Text := StringReplace(RichEdit1.Text, #13#10, '', [rfReplaceall]);

RichEdit1.SelStart := start;

long := long - 2 \* pos;

RichEdit1.SelLength := long;

i := need;

if i <= long then

while (i <= long) and (length(RichEdit1.SelText) > 0) do

begin

fl := true;

j := need;

while (j > 0) and fl do

begin

if RichEdit1.SelText[j] in Special then

begin

while RichEdit1.SelText[j] in Special do

Inc(j);

dec(j);

RichEdit1.SelStart := start + i - need + j;

RichEdit1.SelLength := 0;

RichEdit1.SelText := #13#10;

RichEdit1.SelStart := start + i - need + j + 1;

i := RichEdit1.SelStart - start;

RichEdit1.SelLength := length(RichEdit1.Text) - 1;

Inc(long, 2);

fl := false;

end;

dec(j);

end;

Inc(i, need);

end;

RichEdit1.Lines.EndUpdate;

end

else

ShowMessage('Слишком большой текст для обработки');

end;

procedure TForm2.Splitter1Moved(Sender: TObject);

begin

if Splitter2.Left > Splitter1.Left then

begin

SizePanelLeft.Width := Splitter1.Left;

Splitter2.Left := Splitter1.Left;

end;

firstInden := Splitter1.Left;

LeftInden := sizepanel.Width - Splitter2.Left;

RichEdit1.Paragraph.FirstIndent := 3 \* firstInden div 5;

RichEdit1.Paragraph.LeftIndent := -3 \* LeftInden div 5;

RichEdit1.Paragraph.RightIndent := 3 \* (RichEdit1.Width - RightSplitter.Left)

div 5 + 1;

end;

end.

unit HashTable;//Модуль хэш-таблицы

interface

uses

System.SysUtils;

type

TKeyValue = record

Key: string;

Value: Integer;

end;

PSimpleNode = ^TSimpleNode;

TSimpleNode = record

Next: PSimpleNode;

data: TKeyValue;

end;

DinMAss = array of PSimpleNode;

TMAS = array [0 .. 29] of Integer;

THashTable = class

private

FTable: DinMAss;

FCount: Integer;

FSize: Integer;

Sizes\_index: Integer;

const

Sizes: TMAS = (5, 11, 23, 47, 97, 193, 389, 769, 1543, 3072, 3079, 12289,

24593, 49157, 98317, 196613, 393241, 786433, 1572869, 3145739, 6291469,

12582917, 25165843, 50331653, 100663319, 201326611, 402653189, 805306457,

1610612736, 2147483629);

function GetHash(const AKey: string): Integer;

function CheckMemory(): boolean;

procedure MemoryAdd();

procedure Add(const AKey: string; AValue: Integer);

public

constructor Create;

destructor Destroy; override;

function Get(const AKey: string): Integer;

procedure PutOrUpdate(const AKey: string; AValue: Integer);

function ContainsKEyUpdated(const AKey: string): boolean;

function ContainsKEy(const AKey: string): boolean;

property Count: Integer read FCount;

end;

implementation

constructor THashTable.Create;

begin

Sizes\_index := 0;

FSize := Sizes[Sizes\_index];

SetLength(FTable, Sizes[Sizes\_index]);

end;

destructor THashTable.Destroy;

var

i: Integer;

x, temp: PSimpleNode;

begin

for i := 0 to Length(FTable) - 1 do

begin

x := FTable[i];

while x <> nil do

begin

temp := x^.Next;

Dispose(x);

x := temp;

end;

end;

FTable := nil;

inherited;

end;

function THashTable.CheckMemory(): boolean;

var

factor: double;

begin

factor := 0.75;

result := FCount / FSize >= factor;

end;

function THashTable.GetHash(const AKey: string): Integer;

var

i: Integer;

begin

result := 0;

for i := 1 to Length(AKey) do

result := (result \* 31 + Ord(AKey[i])) mod FSize;

end;

procedure THashTable.Add(const AKey: string; AValue: Integer);

var

i: Integer;

fl: boolean;

x: PSimpleNode;

begin

i := Get(AKey);

if FTable[i] = nil then

begin

new(FTable[i]);

FTable[i]^.data.Key := AKey;

FTable[i]^.data.Value := AValue;

FTable[i]^.Next := nil;

inc(FCount);

end

else

begin

x := FTable[i];

fl := true;

if (x^.data.Key = AKey) then

fl := false;

while (x^.Next <> nil) and fl do

begin

if (x^.data.Key = AKey) then

fl := false;

x := x^.Next;

end;

if fl then

begin

new(x^.Next);

x := x^.Next;

x^.data.Key := AKey;

x^.data.Value := AValue;

x^.Next := nil;

inc(FCount);

end;

end;

end;

procedure THashTable.MemoryAdd();

var

oldArray: DinMAss;

i: Integer;

x: PSimpleNode;

begin

FCount := 0;

oldArray := FTable;

FTable := nil;

SetLength(FTable, Sizes[Sizes\_index + 1]);

FSize := Sizes[Sizes\_index + 1];

inc(Sizes\_index);

for i := 0 to Length(oldArray) - 1 do

begin

x := oldArray[i];

while x <> nil do

begin

Add(x^.data.Key, x^.data.Value);

x := x^.Next;

end;

end;

end;

function THashTable.Get(const AKey: string): Integer;

var

i: Integer;

begin

i := GetHash(AKey);

result := i mod Length(FTable);

end;

procedure THashTable.PutOrUpdate(const AKey: string; AValue: Integer);

var

i: Integer;

begin

if CheckMemory() then

MemoryAdd();

Add(AKey, AValue);

end;

function THashTable.ContainsKEyUpdated(const AKey: string): boolean;

var

i: Integer;

x: PSimpleNode;

fl: boolean;

begin

i := Get(AKey);

x := FTable[i];

fl := false;

while (x <> nil) and not fl do

begin

if (x^.data.Key = AKey) and (x^.data.Value = 0) then

begin

fl := true;

inc(x^.data.Value);

end;

x := x^.Next;

end;

result := fl;

end;

function THashTable.ContainsKey(const AKey: string): boolean;

var

i: Integer;

x: PSimpleNode;

fl: boolean;

begin

i := Get(AKey);

x := FTable[i];

fl := false;

while (x <> nil) and not fl do

begin

if (x^.data.Key = AKey) then

begin

fl := true;

end;

x := x^.Next;

end;

result := fl;

end;

end.

//Модуль Compare

unit Compare;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics, Math,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.ComCtrls,

Vcl.ExtCtrls, HashTAble, System.ImageList, Vcl.ImgList, Vcl.Imaging.jpeg,

Vcl.Imaging.pngimage;

type

TForm3 = class(TForm)

PanelU3Edits: TPanel;

RichEdit1: TRichEdit;

PanelU3Buttons: TPanel;

CompareTexts: TButton;

RichEdit2: TRichEdit;

RichEdit3: TRichEdit;

GoBAckbutton: TButton;

ImageList1: TImageList;

FirstText: TLabel;

SecondText: TLabel;

ReplaceBtn: TButton;

signsRadioButton: TRadioButton;

WordsRadioButton: TRadioButton;

ShinglesButton: TButton;

ImageList2: TImageList;

SaveDialogCompare: TSaveDialog;

PanelForSafe: TPanel;

SaveComparedTexts: TButton;

procedure CompareText(mode: integer);

procedure CompareTextsClick(Sender: TObject);

procedure GoBAckbuttonClick(Sender: TObject);

procedure ReplaceBtnClick(Sender: TObject);

procedure PanelU3ButtonsResize(Sender: TObject);

procedure signsRadioButtonClick(Sender: TObject);

procedure WordsRadioButtonClick(Sender: TObject);

procedure ShinglesButtonClick(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure SaveComparedTextsClick(Sender: TObject);

procedure SAveFile;

function Shingles(str1: ansistring; n: integer): THAshTAble;

function Count(HashTAble: THAshTAble; str1: ansistring; n: integer)

: extended;

private

F: string;

NumberWords: integer;

public

NumberShingles: integer;

end;

Type

DinMAss = array of integer;

ArrOfStr = array of string;

ArrOfNUms = array of integer;

Matrix = array of array of integer;

var

Form3: TForm3;

implementation

{$R \*.dfm}

const

Special = ['"', '"', ',', '.', ':', '$', '#', '(', ')', ' ', '-', ':', ';',

'«', '»'];

ReplaceLetters = ['a', 'o', 'i', 'u', 'y', 'e', 'а', 'о', 'е', 'и', 'ё', 'ю',

'я', 'э', 'у', 'ы'];

function OneWord(var str1: string; mode: integer): ArrOfStr;

var

i, p: integer;

start: integer;

strmas: ArrOfStr;

begin

i := 1;

p := 0;

str1 := StringReplace(str1, #13#10, '', [rfReplaceall]);

if mode = 1 then

while i <= length(str1) do

begin

start := i;

while (i <= length(str1)) and (str1[i] <> ' ') do

begin

inc(i);

end;

SetLength(strmas, length(strmas) + 1);

strmas[p] := Copy(str1, start, i - start);

inc(p);

inc(i);

end

else

begin

SetLength(strmas, length(str1));

for i := 0 to high(strmas) do

strmas[i] := str1[i + 1]

end;

result := strmas;

end;

function MarkDiffer(strNUms1: ArrOfStr; LCS: ArrOfStr): ArrOfStr;

var

j, k, i: integer;

Str1Marks: ArrOfStr;

begin

j := 0;

k := 0;

SetLength(Str1Marks, length(strNUms1));

For i := 0 To length(strNUms1) - 1 do

begin

If (j <= length(LCS) - 1) and (LCS[j] = strNUms1[k]) Then

begin

Str1Marks[i] := '=';

inc(j);

end

else

Str1Marks[i] := '-';

inc(k);

End;

result := Str1Marks;

end;

function MaxSubLine(strNUms1: ArrOfStr; strNUms2: ArrOfStr; var Len: integer)

: ArrOfStr;

var

i, j, n, m, c: integer;

MaxLEn: Matrix;

Res: ArrOfStr;

begin

n := length(strNUms1);

m := length(strNUms2);

SetLength(MaxLEn, n + 1, m + 1);

for i := 0 to m do

MaxLEn[0, i] := 0;

for i := 0 to n do

MaxLEn[i, 0] := 0;

for i := 1 to n do

begin

for j := 1 to m do

begin

if strNUms1[i - 1] = strNUms2[j - 1] then

MaxLEn[i, j] := 1 + MaxLEn[i - 1, j - 1]

else if MaxLEn[i - 1, j] > MaxLEn[i, j - 1] Then

MaxLEn[i, j] := MaxLEn[i - 1, j]

else

MaxLEn[i, j] := MaxLEn[i, j - 1]

end

end;

Len := MaxLEn[n, m];

If MaxLEn[n, m] = 0 Then

Exit;

SetLength(Res, MaxLEn[n, m]);

c := MaxLEn[n, m] - 1;

i := n;

j := m;

While (i > 0) And (j > 0) do

begin

If strNUms1[i - 1] = strNUms2[j - 1] Then

begin

Res[c] := strNUms1[i - 1];

c := c - 1;

i := i - 1;

j := j - 1;

end

else if MaxLEn[i - 1, j] = MaxLEn[i, j] then

i := i - 1

else

j := j - 1

end;

result := Res;

end;

procedure TForm3.GoBAckbuttonClick(Sender: TObject);

begin

Self.ModalResult := mrOk;

end;

procedure TForm3.CompareText(mode: integer);

var

i, j: integer;

str1, str2, str: string;

LCS: ArrOfStr;

strarr1, strarr2: ArrOfStr;

Str1Marks, str2Marks: ArrOfStr;

AddMas: array [0 .. 1] of string;

begin

AddMas[0] := '';

AddMas[1] := ' ';

str1 := RichEdit1.Text;;

str := str1;

str2 := RichEdit2.Text;

strarr1 := OneWord(str1, mode);

strarr2 := OneWord(str2, mode);

LCS := MaxSubLine(strarr1, strarr2, j);

Str1Marks := MarkDiffer(strarr1, LCS);

str2Marks := MarkDiffer(strarr2, LCS);

i := 0;

j := 0;

SetLength(strarr1, length(strarr1) + 1);

SetLength(strarr2, length(strarr2) + 1);

SetLength(Str1Marks, length(Str1Marks) + 1);

SetLength(str2Marks, length(str2Marks) + 1);

While (i <= length(strarr1) - 2) or (j <= length(strarr2) - 2) do

begin

If (Str1Marks[i] = '-') And (str2Marks[j] = '-') Then

begin

with RichEdit3 do

begin

SelStart := length(RichEdit3.Text);

SelAttributes.Color := clRED;

SelAttributes.Style := [fsStrikeOut];

SelText := strarr1[i];

SelAttributes.Style := [fsUnderline];

SelAttributes.Color := clGreen;

SelText := strarr2[j] + AddMas[mode];

end;

Str1Marks[i] := '';

strarr1[i] := '';

str2Marks[j] := '';

strarr2[j] := '';

inc(i);

inc(j);

end

else If (Str1Marks[i] <> '-') And (str2Marks[j] = '-') then

begin

with RichEdit3 do

begin

SelStart := length(RichEdit3.Text) - length(strarr1[i]) + 1;

SelAttributes.Color := clGreen;

SelAttributes.Style := [fsUnderline];

SelText := strarr2[j] + AddMas[mode];

end;

str2Marks[j] := '';

strarr2[j] := '';

inc(j);

end

else If (Str1Marks[i] = '-') And (str2Marks[j] <> '-') then

begin

with RichEdit3 do

begin

SelStart := length(RichEdit3.Text);

SelAttributes.Color := clRED;

SelAttributes.Style := [fsStrikeOut];

SelText := strarr1[i] + AddMas[mode];

end;

Str1Marks[i] := '';

strarr1[i] := '';

inc(i);

end

else

begin

with RichEdit3 do

begin

SelStart := length(RichEdit3.Text);

SelAttributes.Color := clBlack;

SelAttributes.Style := [];

SelText := strarr1[i] + AddMas[mode];

end;

Str1Marks[i] := '';

strarr1[i] := '';

str2Marks[j] := '';

strarr2[j] := '';

inc(i);

inc(j);

end;

if (i > length(strarr1) - 2) and (j <= length(strarr2) - 2) then

i := length(strarr1) - 1;

if (i <= length(strarr1) - 2) and (j > length(strarr2) - 2) then

j := length(strarr2) - 1;

end;

SetLength(Str1Marks, 0);

SetLength(str2Marks, 0);

SetLength(strarr1, 0);

SetLength(strarr2, 0);

RichEdit3.visible := true;

end;

procedure TForm3.PanelU3ButtonsResize(Sender: TObject);

begin

RichEdit2.Width := Width div 2;

ReplaceBtn.left := Width div 2 - ReplaceBtn.Width div 2;

FirstText.left := Width div 4 - FirstText.Width div 2;

SecondText.left := 3 \* Width div 4 - SecondText.Width div 2;

end;

procedure TForm3.ReplaceBtnClick(Sender: TObject);

var

temp: string;

begin

temp := RichEdit1.Text;

RichEdit1.Text := RichEdit2.Text;

RichEdit2.Text := temp;

end;

function TForm3.Shingles(str1: ansistring; n: integer): THAshTAble;

var

i, k, j, start: integer;

mas3: array of string;

Arr: string;

HashTAble: THAshTAble;

wasword: boolean;

begin

i := 1;

k := 0;

NumberWords := 0;

HashTAble := THAshTAble.Create;

str1 := StringReplace(str1, #13#10, '', [rfReplaceall]);

str1 := ansilowerCase(str1);

SetLength(mas3, n);

while i <= length(str1) do

begin

start := i;

wasword := false;

while (i <= length(str1)) and not(str1[i] in Special) do

begin

if str1[i] = 'p' then

str1[i] := 'р'

else if str1[i] = 'c' then

str1[i] := 'с';

if str1[i] in ReplaceLetters then

Delete(str1, i, 1)

else

inc(i);

wasword := true;

end;

if wasword then

begin

mas3[k] := Copy(str1, start, i - start);

if k = n - 1 then

begin

for j := 0 to length(mas3) - 1 do

Arr := Arr + mas3[j];

if (not HashTAble.ContainsKEy(Arr)) then

begin

HashTAble.PutOrUpdate(Arr, 0);

inc(NumberWords);

end;

Arr := '';

for j := 0 to length(mas3) - 2 do

begin

mas3[j] := mas3[j + 1];

end;

end

else

inc(k);

end;

inc(i);

end;

result := HashTAble;

end;

function TForm3.Count(HashTAble: THAshTAble; str1: ansistring; n: integer)

: extended;

var

i, k, j, start, p, number: integer;

mas3: array of string;

Arr: string;

wasword: boolean;

begin

result := 0;

number := 0;

i := 1;

k := 0;

str1 := StringReplace(str1, #13#10, '', [rfReplaceall]);

str1 := StringReplace(str1, #9, '', [rfReplaceall]);

str1 := ansilowerCase(str1);

SetLength(mas3, n);

while i <= length(str1) do

begin

start := i;

wasword := false;

while (i <= length(str1)) and not(str1[i] in Special) do

begin

if str1[i] = 'p' then

str1[i] := 'р'

else if str1[i] = 'c' then

str1[i] := 'с';

if str1[i] in ReplaceLetters then

Delete(str1, i, 1)

else

inc(i);

wasword := true;

end;

if wasword then

begin

mas3[k] := Copy(str1, start, i - start);

if k = n - 1 then

begin

for j := 0 to length(mas3) - 1 do

Arr := Arr + mas3[j];

if HashTAble.ContainsKEyUpdated(Arr) then

begin

inc(number);

end;

Arr := '';

for j := 0 to length(mas3) - 2 do

begin

mas3[j] := mas3[j + 1];

end;

end

else

inc(k);

end;

inc(i);

end;

if NumberWords <> 0 then

result := 100 \* number / NumberWords

else

result := 0;

HashTAble.Destroy;

end;

procedure TForm3.ShinglesButtonClick(Sender: TObject);

var

Res: extended;

n: integer;

s, str1, str2: string;

begin

n := NumberShingles;

str1 := RichEdit1.Text;;

str2 := RichEdit2.Text;

MaxSubLine(OneWord(str1, 0), OneWord(str2, 0), n);

if (length(str1) + length(str2) - n) <> 0 then

begin

Res := RoundTo(100 \* n / (length(str1) + length(str2) - n), -2);

s := 'Процентная схожесть певрого текста со вторым (%) - ' +

FloatToStr(Res) + #13#10 + 'Разница (%)' + FloatToStr(100 - Res) + #13#10

+ 'Общий (символы) ' + IntToStr(n) + #13#10 + 'Разница (символы) ' +

IntToStr((length(str1) + length(str2) - 2 \* n));

end

else

s := 'Тексты пусты';

ShowMessage(s);

end;

procedure TForm3.signsRadioButtonClick(Sender: TObject);

begin

if signsRadioButton.Checked then

WordsRadioButton.Checked := false

else

WordsRadioButton.Checked := true;

end;

procedure TForm3.WordsRadioButtonClick(Sender: TObject);

begin

if WordsRadioButton.Checked then

signsRadioButton.Checked := false

else

signsRadioButton.Checked := true;

end;

procedure TForm3.CompareTextsClick(Sender: TObject);

var

mode: integer;

begin

if WordsRadioButton.Checked then

mode := 1

else

mode := 0;

RichEdit3.CLear;

CompareText(mode);

if RichEdit3.Text <> '' then

SaveComparedTexts.visible := true

else

SaveComparedTexts.visible := false;

end;

procedure TForm3.FormCreate(Sender: TObject);

begin

NumberShingles := 3;

SaveDialogCompare.Filter := 'RTF Files |\*.rtf';

end;

procedure TForm3.SAveFile;

begin

if SaveDialogCompare.Execute then

begin

RichEdit3.Lines.SaveToFile(SaveDialogCompare.Filename + '.rtf');

F := SaveDialogCompare.Filename + '.rtf';

end;

end;

procedure TForm3.SaveComparedTextsClick(Sender: TObject);

begin

SAveFile;

end;

end.

//Модули дополнительных окон(для замены слова и сохранения файла)

unit ReplaceUnit;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls;

type

TReplaceForm = class(TForm)

FindEditRep: TEdit;

Label1: TLabel;

ReplaceEdit: TEdit;

Label2: TLabel;

SelTExtCheckBox: TCheckBox;

ReplaceButton: TButton;

procedure ReplaceButtonClick(Sender: TObject);

procedure ClearEdits;

private

{ Private declarations }

public

Find: string;

ReplaceStr: string;

Check: boolean;

end;

var

ReplaceForm: TReplaceForm;

implementation

{$R \*.dfm}

procedure TReplaceForm.ClearEdits;

begin

FindEditRep.Clear;

ReplaceEdit.Clear;

end;

procedure TReplaceForm.ReplaceButtonClick(Sender: TObject);

begin

if FindEditRep.Text <> '' then

begin

Find := FindEditRep.Text;

ReplaceStr := ReplaceEdit.Text;

Check:=SelTExtCheckBox.Checked;

ModalResult := mrYes;

end

else

ShowMessage('Поле найти должно быть заполнено');

end;

end.

unit Window;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls;

type

TForm4 = class(TForm)

SAveButton: TButton;

UndoF4Button: TButton;

NotSaveButton: TButton;

SAveLabel: TLabel;

procedure SAveButtonClick(Sender: TObject);

procedure NotSaveButtonClick(Sender: TObject);

procedure UndoF4ButtonClick(Sender: TObject);

private

{ Private declarations }

public

Click: integer;

end;

var

Form4: TForm4;

implementation

{$R \*.dfm}

procedure TForm4.NotSaveButtonClick(Sender: TObject);

begin

Click := 1;

ModalResult := mrOk;

end;

procedure TForm4.SAveButtonClick(Sender: TObject);

begin

Click := 2;

ModalResult := mrOk;

end;

procedure TForm4.UndoF4ButtonClick(Sender: TObject);

begin

Click := 3;

ModalResult := mrOk;

end;

end.

ВЕДОМОСТЬ

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Обозначение | | | | Наименование | | | | Дополнительные сведения | | | |
|  | | | | Текстовые документы | | | |  | | | |
|  | | | |  | | | |  | | | |
| БГУИР КР 1–40 01 01 104 ПЗ | | | | Пояснительная записка | | | | 64 с. | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | | Графические документы | | | |  | | | |
|  | | | |  | | | |  | | | |
| ГУИР 251001 104 СП | | | | “Программное средство редактирования, сравнения текстов и процентной оценки плагиата “  А1, схема программы, чертеж | | | | Формат А1 | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  |  |  |  |  | БГУИР КР 1-40 01 01 104 ПЗ | | | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| Изм. | Л. | № докум. | Подп. | Дата | “Программное средство редактирования, сравнения текстов и процентной оценки плагиата”  Ведомость курсовой работы |  | | | | Лист | Листов |
| Разраб. | | Василенко И.И. |  |  | Т |  | |  | 64 | 64 |
| Пров. | | Фадеева Е.Е. |  |  | Кафедра ПОИТ  гр. 251001 | | | | | |
|  | |  |  |  |
|  | |  |  |  |
|  | |  |  |  |