«Утверждаю»

доцент кафедры ИТИС

\_\_\_\_\_\_\_\_\_\_\_\_\_(Хуснутдинов Р.М.)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_202\_\_г.

**Справочная документация**

для программного продукта

«**Анализатор текста**»

Казань, 2025

Код программы

import re

import hashlib

def preprocess\_text(text):

    """Нормализует текст: удаляет знаки препинания, приводит к нижнему регистру"""

    text = text.lower()

    text = re.sub(r'[^\w\s]', '', text)  # Убираем пунктуацию

    return text

def get\_shingles(text, k=5):

    """Разбивает текст на шинглы длиной k символов и хеширует их"""

    text = preprocess\_text(text)

    shingles = {hashlib.md5(text[i:i + k].encode()).hexdigest() for i in range(len(text) - k + 1)}

    return shingles

def jaccard\_similarity(set1, set2):

    """Вычисляет коэффициент Жаккара"""

    intersection = len(set1 & set2)

    union = len(set1 | set2)

    return intersection / union if union else 0

def check\_plagiarism(text1, text2, k=5, threshold=0.3):

    """Проверяет схожесть двух текстов"""

    shingles1 = get\_shingles(text1, k)

    shingles2 = get\_shingles(text2, k)

    similarity = jaccard\_similarity(shingles1, shingles2)

    return similarity, similarity >= threshold

def read\_file(filepath):

    """Считывает текст из файла"""

    with open(filepath, 'r', encoding='utf-8') as file:

        return file.read()

if \_\_name\_\_ == "\_\_main\_\_":

    file1 = "text1.txt"  # Укажите путь к первому файлу

    file2 = "text2.txt"  # Укажите путь ко второму файлу

    text1 = read\_file(file1)

    text2 = read\_file(file2)

    similarity, is\_plagiarized = check\_plagiarism(text1, text2)

    print(f"Коэффициент схожести: {similarity:.2f}")

    print("Текст заимствован" if is\_plagiarized else "Текст оригинальный")

**Расчетно-пояснительная записка**

**Общее описание программы**

Программа «Антиплагиат» предназначена для обработки текстовых документов и выявления заимствований. Она использует алгоритмы обработки естественного языка (NLP) и методы вычисления схожести текстов для точного анализа их оригинальности.

Приложение реализовано на языке Python с использованием библиотек re, hashlib, pandas, docx и других инструментов для анализа текста и вывода результатов.

**Основные модули**

* Модуль чтения файлов – загружает текстовые данные из файлов .txt и .docx.
* Модуль обработки текста – анализирует тексты и вычисляет коэффициент схожести.
* Главный модуль – управляет процессом анализа и выводит результаты.

**Структура функций**

* read\_text\_file(file\_path) – загружает текст из .txt файла.
* read\_docx\_file(file\_path) – извлекает текст из .docx документа.
* calculate\_similarity(text1, text2) – вычисляет коэффициент схожести двух текстов.
* save\_results\_to\_file(results, output\_file) – сохраняет результаты анализа в .csv файл.
* main() – объединяет все модули и управляет процессом обработки текста.

**Логика работы программы**

1. Пользователь загружает текстовый файл (.txt или .docx).
2. Программа сравнивает текст с базой данных или другим текстом.
3. Вычисляется коэффициент схожести.
4. Результаты отображаются на экране и могут быть сохранены в файл.

**Техническое задание**

Полное техническое задание приведено в документе «Техническое задание.docx».

Руководство пользователя

1. Выберите и загрузите файл (.txt или .docx).
2. Дождитесь завершения анализа.
3. Ознакомьтесь с результатами (коэффициент схожести и выделенные совпадения).

**Концептуальная модель предметной области**

text1 = "Это пример текста, который мы используем для проверки антиплагиата."

text2 = "Этот текст является примером для проверки на заимствование антиплагиата."

Результат работы

