Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра интеллектуальных информационных технологий

ОТЧЁТ

по курсу «Естественно-языковой интерфейс интеллектуальных систем»

Лабораторная работа №3

«Синтаксический анализ текстов естественного языка»

Выполнил:

студент гр. 221701

Телица И.Д.

Проверил:

Крапивин Ю.Б.

Минск, 2025

# Цель работы

Освоить принципы разработки прикладных сервисных программ для решения задачи анализа текста естественного языка.

# Задачи лабораторной работы

**1.** Познакомиться с назначением, структурой и функциональностью, предоставляемой базовым программным обеспечением для решения задачи автоматического синтаксического анализа текста естественного языка.

**2.** Закрепить навыки программирования при решении задач автоматической обработки текста естественного языка.

# Используемые библиотеки

**1. os:** Стандартная библиотека Python для работы с файловой системой.

**2. json:** Стандартная библиотека Python для работы с JSON-данными.

**3. collections:** Стандартная библиотека Python для специализированных контейнеров (defaultdict, Counter).

**4. python-docx:** Библиотека для чтения DOCX-файлов.

**5. pdfplumber:** Библиотека для извлечения текста из PDF-файлов.

**6. striprtf:** Библиотека для обработки RTF-файлов.

**7. beautifulsoup4:** Библиотека для парсинга HTML-файлов.

**8. nltk:** Библиотека для обработки естественного языка, используется для токенизации.

**9. pymorphy2:** Библиотека для морфологического анализа русского языка.

**10. natasha:** Библиотека для обработки естественного языка на русском языке, используется для синтаксического анализа.

**11. tkinter:** Стандартная библиотека Python для создания графического интерфейса.

# Интерфейс

Интерфейс включает:

* • **Панель управления:** Кнопки «Загрузить текст», «Сохранить корпус», «Показать статистику», «Показать синтаксис», «Экспортировать результаты», «Справка».
* • **Панель поиска:** Поле ввода, радиокнопки «Слово»/«Лемма», кнопка «Искать».
* • **Область вывода:** Текстовое поле для результатов поиска, статистики и синтаксического анализа.

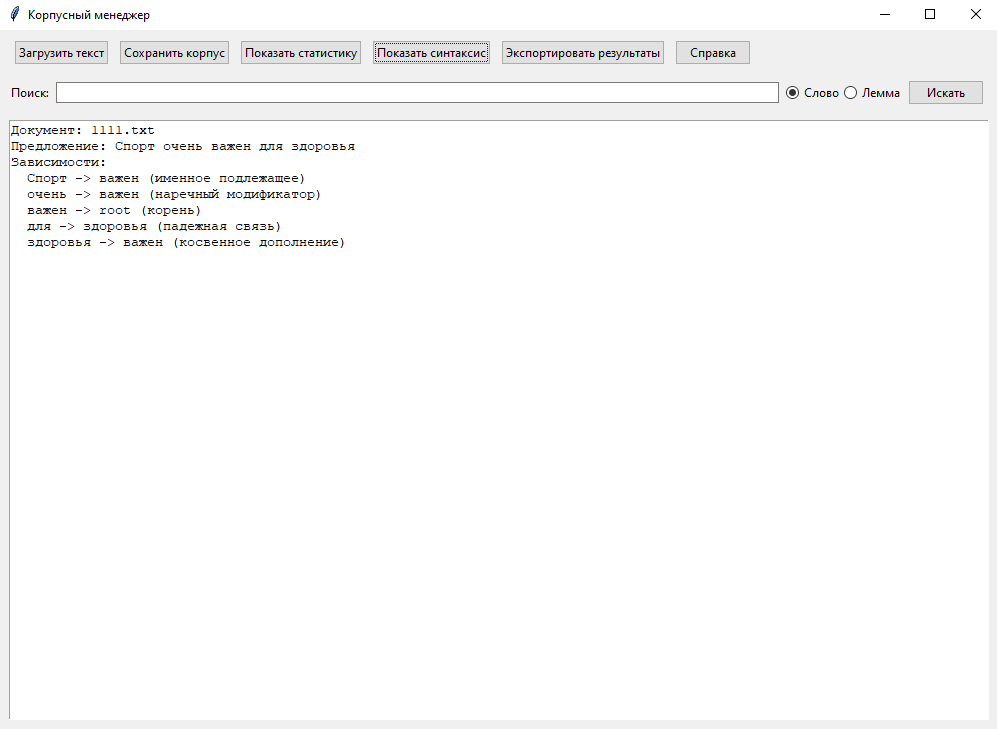


Рисунок 1: Интерфейс приложения

# Структура хранения данных

Данные хранятся в списке словарей (corpus), где каждый словарь представляет документ:

* text: Исходный текст.
* tokens: Список токенов с информацией (слово, лемма, часть речи, теги).
* metadata: Метаданные (название, источник, дата, тип).
* syntax: Список предложений с синтаксическими зависимостями.

Индексы:

* word\_index: Индекс слов для поиска по подстроке.
* lemma\_index: Индекс лемм для поиска по нормализованным формам.

Пример структуры:  
[

{

"text": "Спорт очень важен для здоровья",

"tokens": [

{

"word": "Спорт",

"lemma": "спорт",

"pos": "NOUN",

"tags": "существительное, неодушевлённое, masc sing, именительный падеж"

},

{

"word": "очень",

"lemma": "очень",

"pos": "ADVB",

"tags": "наречие"

},

{

"word": "важен",

"lemma": "важный",

"pos": "ADJS",

"tags": "прилагательное (краткое), Qual masc, единственное число"

},

{

"word": "для",

"lemma": "для",

"pos": "PREP",

"tags": "предлог"

},

{

"word": "здоровья",

"lemma": "здоровье",

"pos": "NOUN",

"tags": "существительное, неодушевлённое, neut sing, родительный падеж"

}

],

"metadata": {

"title": "1111.txt",

"source": "Загруженный файл",

"date": "2025-05-15",

"type": "unknown"

},

"syntax": [

{

"sentence": "Спорт очень важен для здоровья",

"dependencies": [

[

"Спорт",

"важен",

"именное подлежащее"

],

[

"очень",

"важен",

"наречный модификатор"

],

[

"важен",

"root",

"корень"

],

[

"для",

"здоровья",

"падежная связь"

],

[

"здоровья",

"важен",

"косвенное дополнение"

]

]

}

]

}

]

# Структурная схема приложения

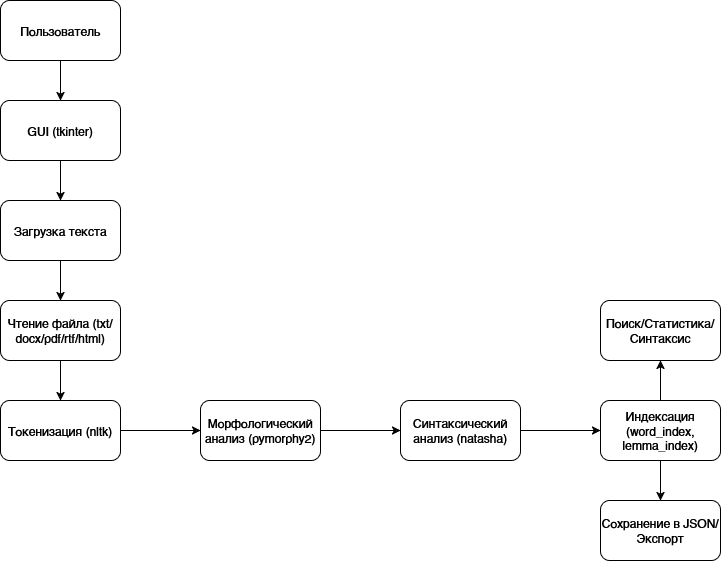


Рисунок 2: Структурная схема приложения

# Описание алгоритмов

## Анализ текста

1. Пользователь загружает файл (TXT, DOCX, PDF, RTF, HTML).
2. Текст извлекается с помощью соответствующей библиотеки (os, python-docx, pdfplumber, striprtf, beautifulsoup4).
3. Текст токенизируется с помощью nltk.
4. Для каждого токена выполняется морфологический анализ с помощью pymorphy2.
5. Производится сегментация текста на предложения с помощью natasha.
6. Выполняется синтаксический анализ (natasha), определяются зависимости между токенами.
7. Токены индексируются (word\_index, lemma\_index).
8. Результаты сохраняются в корпус и отображаются в GUI.

## Поиск по корпусу

1. Пользователь вводит запрос и выбирает тип поиска (слово/лемма).
2. Запрос ищется в соответствующем индексе.
3. Для каждого совпадения извлекается контекст и метаданные.
4. Результаты отображаются в GUI.

# Тест производительности

Тест проводился на корпусе из 10 документов (5000 слов). Среднее время выполнения на стандартном ПК:

* Добавление документа (включая морфологический и синтаксический анализ): 2,5 с.
* Поиск по слову: 0,03 с.
* Получение синтаксического анализа: 0,1 с.

Основное время занимает синтаксический анализ natasha.

# Вывод

Разработан корпусный менеджер для синтаксического анализа текстов в области спорта. Приложение поддерживает загрузку текстов различных форматов, морфологический и синтаксический анализ, поиск, статистику и экспорт результатов. Интерфейс интуитивно понятен, включает систему подсказок. Перспективы: оптимизация производительности, визуализация синтаксических деревьев, поддержка других языков.