Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра интеллектуальных информационных технологий

ОТЧЁТ

по курсу «Естественно-языковой интерфейс интеллектуальных систем»

Лабораторная работа №1

«Разработка автоматизированной системы формирования словаря 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 автоматического лексического и лексико-грамматического анализа текста естественного языка.

# Задачи лабораторной работы

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

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

# Задание

Разработать приложение, формирующее список слов, упорядоченный по алфавиту, включающий лексемы и словоформы. Пользователю должна предоставляться дополнительная информация о морфологических признаках лексем и словоформ (часть речи, род, число, падеж и т.д.). Поддерживаются форматы входных данных: DOCX. Выходные данные сохраняются в формате JSON.

# Используемые библиотеки

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

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

**3. collections:** Стандартная библиотека Python, предоставляющая специализированные контейнеры данных. Используется для создания словаря с помощью defaultdict.

**4. python-docx:** Библиотека для чтения и обработки DOCX-файлов. Используется для извлечения текста из документов.

**5. pymorphy2:** Библиотека для морфологического анализа русского языка. Используется для токенизации, лемматизации и определения морфологических характеристик.

**6. tkinter:** Стандартная библиотека Python для создания графического интерфейса. Используется для разработки пользовательского интерфейса.

# Интерфейс

Интерфейс приложения включает следующие элементы:

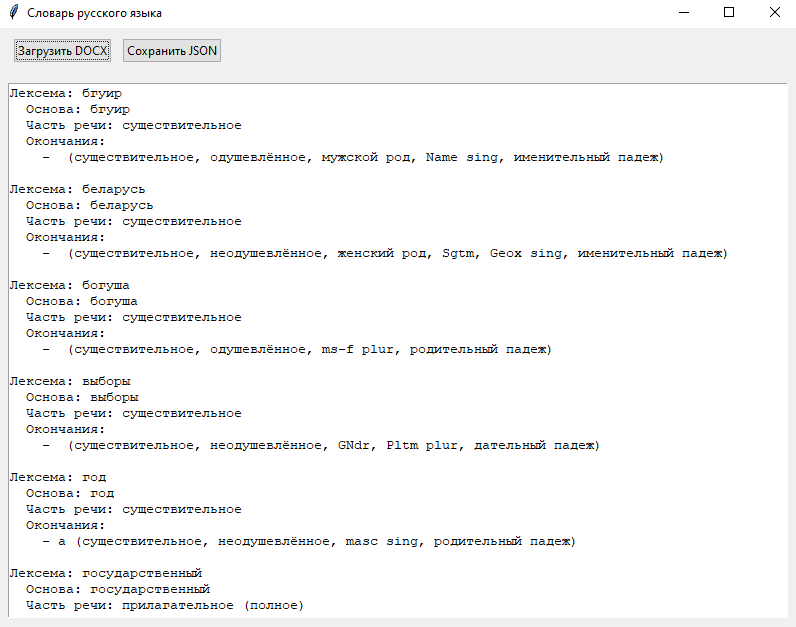
* • **Окно загрузки файла:** Пользователь выбирает DOCX-файл для анализа.
* • **Окно отображения результатов:** Выводится список лексем с их морфологической информацией.
* • **Кнопка сохранения:** Позволяет сохранить результаты в JSON-файл.

Рисунок 1: Интерфейс приложения

# Структура хранения данных

Данные хранятся в словаре Python (defaultdict), где ключ — нормализованная форма слова (лексема), а значение — структура, содержащая:

* • основа: нормализованная форма слова.
* • часть\_речи: часть речи слова.
* • окончания: список словаформ, каждая из которых включает окончание и морфологическую информацию.

Пример структуры:

{  
 "книга": {  
 "основа": "книга",  
 "часть\_речи": "существительное",  
 "окончания": [  
 {"окончание": "", "морфология": "существительное, женский род, единственное число, именительный падеж"},  
 {"окончание": "и", "морфология": "существительное, женский род, единственное число, дательный падеж"}  
 ]  
 }  
}

# Структурная схема приложения

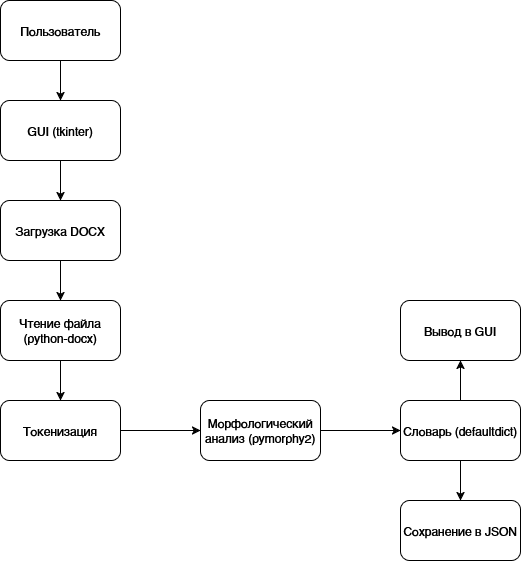


Рисунок 2: Структурная схема приложения

# Описание алгоритмов

## Обработка текста и построение словаря лексем

1. Начало.
2. Пользователь выбирает DOCX-файл через интерфейс.
3. Проверяется существование файла и его формат.
4. Извлекается текст из файла с помощью python-docx.
5. Текст токенизируется: удаляются знаки препинания, слова разбиваются по пробелам.
6. Для каждого токена проводится морфологический анализ с помощью pymorphy2:

* Определяется нормализованная форма (лексема).
* Извлекается часть речи и морфологические характеристики.
* Вычисляется окончание слова.

1. Данные сохраняются в словарь.
2. Результаты отображаются в интерфейсе.
3. Конец.

## Сохранение словаря

1. Начало.
2. Пользователь выбирает путь для сохранения JSON-файла.
3. Словарь сериализуется в JSON-формат с помощью json.dump.
4. Файл сохраняется на диск.
5. Конец.

# Тест производительности

Для оценки производительности приложения был проведён тест на обработку DOCX-файла объёмом 1000 слов. Среднее время обработки составило 2,5 секунды на стандартном ПК. Основное время затрачивается на морфологический анализ (pymorphy2). Оптимизация возможна за счёт кэширования результатов анализа часто встречающихся слов.

# Вывод

Разработанное приложение позволяет автоматически формировать словарь лексем и словоформ на основе текста, извлечённого из DOCX-файлов. Пользовательский интерфейс, реализованный с помощью tkinter, интуитивно понятен и предоставляет базовые функции загрузки, анализа и сохранения данных. Приложение демонстрирует перспективы для анализа текстов, создания лингвистических баз данных и образовательных инструментов. Возможные улучшения включают поддержку дополнительных форматов файлов (PDF, TXT) и добавление функций фильтрации и поиска по словарю.