Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

ОТЧЁТ

по лабораторной работе № 2

по дисциплине

Вариант 1

Естественно-языковой интерфейс интеллектуальных систем

“Построение и использование корпусов текстов естественного языка”

| Выполнили | студенты группы 221701:  Абушкевич А. А.  Робилко Т. М. |
| --- | --- |
| Проверил | Крапивин Ю. Б. Н.В |
|  |  |

Минск 2025

**Цель**:

1. изучить принципы построения корпусов текстов, виды разметки и способы аннотирования, инструменты для работы с корпусами текстов;
2. построить корпус текстов и разработать корпусный менеджер.

**Постановка задачи**:

1. Сформировать электронный корпус текстов по выбранной предметной области;
2. Используя результаты лабораторной работы №1 разработать корпусный менеджер, обеспечивающий базовую функциональность работы с созданным корпусом текстов.

* Входные данные - фрагмент текста (фраза или слово) на естественном языке - запрос корпусному менеджеру;
* Выходные данные - частотные характеристики словоформ, лексем, грамматических категорий, леммы, морфологические характеристики словоформ и их метаданные (библиографические, типологические), конкордансные списки, согласно согласованным с преподавателем требованиям к функциональности корпусного менеджера;
* Взаимодействие с пользователем посредством графического интерфейса (интерфейс должен быть интуитивно-понятным и дружественным пользователю);
* Наличие системы средств помощи пользователю;
* Обеспечение возможности построения, сохранения, просмотра, редактирования, пополнения, фильтрации и поиска по заданному условию, документирования текста и/или его фрагмента в соответствии с реализуемой функциональностью корпусного менеджера;

Вариант 1. Язык текста - русский. Предметная область - Кинематограф.

**Средства разработки**

Для разработки был выбран язык программирования Python. Использованы следующие библиотеки:

* библиотека Natasha (для обработки текста естественного языка);
* библиотека Tkinter (для создания графического интерфейса);
* библиотека json (для сериализации структур данных);
* библиотека string (для обработки строковых переменных);
* библиотека time (для тестирования программы);
* библиотека os (для работы с файловой системой);
* библиотека matplotlib (для построения графиков);
* Система контроля версий Git;

Подробное описание функционала библиотек представлено далее.

**Алгоритмы работы программы:**

Открытие файла:

1. Запуск программы;
2. Выбор пункта меню “Файл - Открыть”;
3. Выбор файла в формате json;
4. Чтение данных из файла;
5. Генерация таблицы с информацией о словоформах;
6. Генерация таблиц с информацией о количестве вхождений словоформ по различным морфологическим признакам.
7. Генерация столбчатых диаграмм с информацией о количестве вхождений словоформ по различным морфологическим признакам.
8. Отображение результатов в рабочей области;
9. Конец.

Добавление текста из файла:

1. Запуск программы;
2. Выбор пункта меню “Редактирование - Добавить текст из файла”;
3. Выбор файла в формате txt;
4. Чтение данных из файла;
5. Отображение в рабочей области;
6. Отображение нового текста в файловой области;
7. Конец.

Добавление текста из файла:

1. Запуск программы;
2. Выбор пункта меню “Редактирование - Добавить текст из файла”;
3. Выбор файла в формате txt;
4. Чтение данных из файла;
5. Отображение в рабочей области;
6. Отображение нового текста в файловой области;
7. Нажать “Редактировать”;
8. Изменить и/или сохранить текст;
9. Конец.

Сохранение файла:

1. Запуск программы;
2. Выбор пункта меню “Редактирование - Добавить текст из файла”;
3. Выбор файла в формате txt;
4. Чтение данных из файла;
5. Отображение в рабочей области;
6. Выбор пункта меню “Файл - Сохранить”;
7. Выбор сохранения в формате json;
8. Сохранение информации в json файл;
9. Конец.

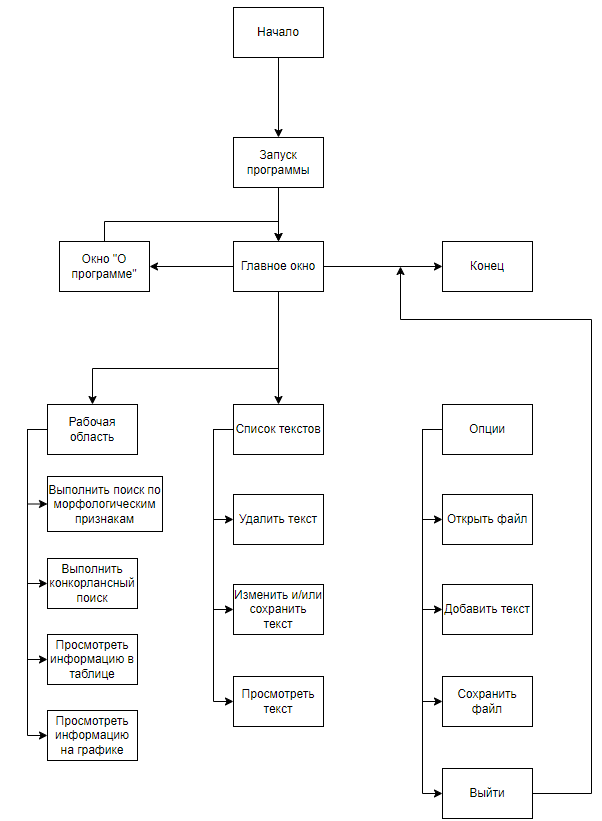
Поиск по морфологическим признакам:

1. Запуск программы;
2. Выбор пункта меню “Файл - Открыть”;
3. Выбор файла в формате json;
4. Чтение данных из файла;
5. Отображение в рабочей области;
6. Выбор пользователем морфологических критериев поиска и/или подстроки для поиска;
7. Поиск по списку словоформ;
   1. Проверка на совпадение критериев поиска в лексеме;
   2. Запись совпавших лексем в новый список;
8. Отображение таблицы результатов поиска;
9. Конец.

Конкордансный поиск:

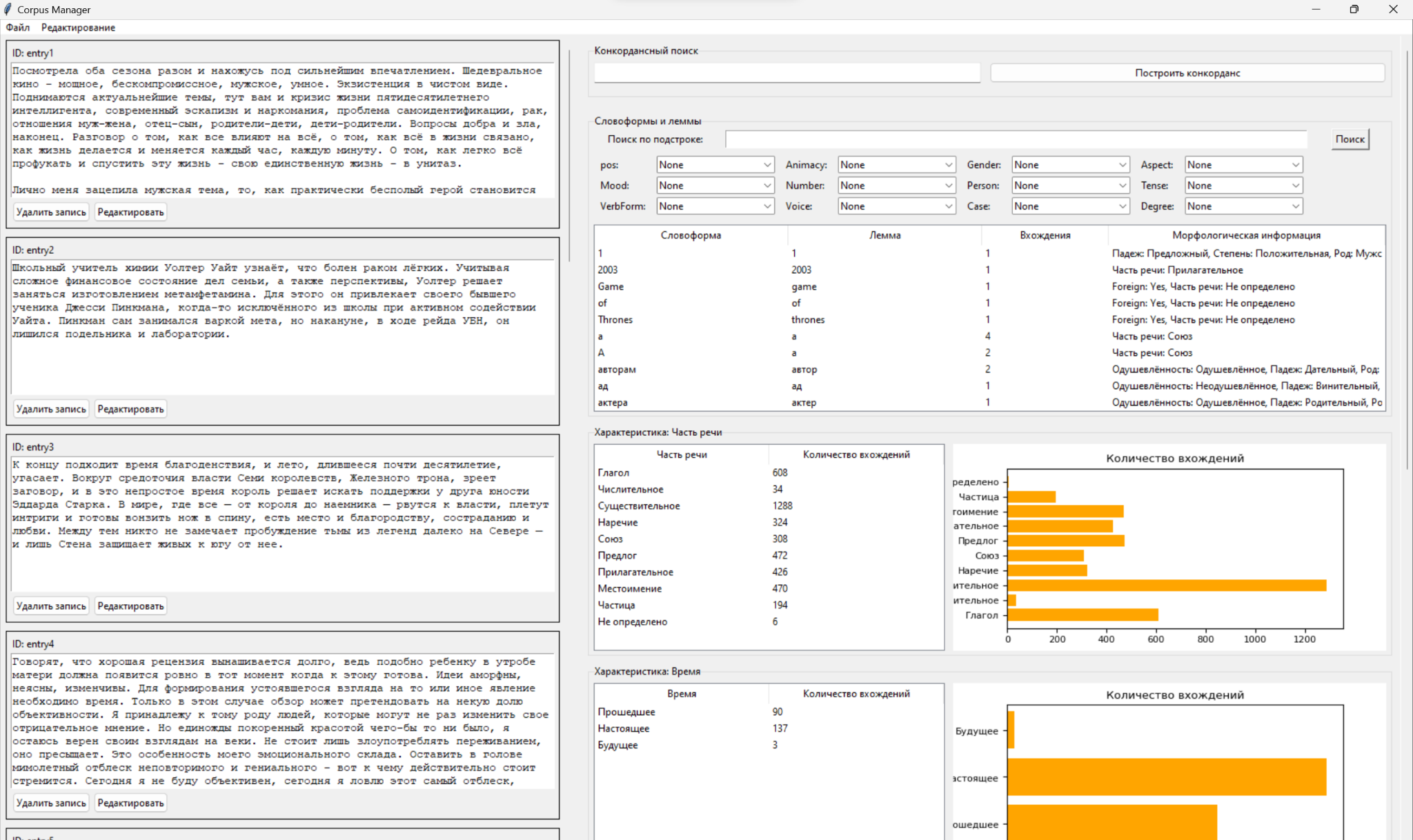
1. Запуск программы;
2. Выбор пункта меню “Файл - Открыть”;
3. Выбор файла в формате json;
4. Чтение данных из файла;
5. Отображение данных в рабочей области;
6. Ввод пользователем словоформы либо леммы для поиска;
7. Поиск по списку словоформ;
   1. Проверка на совпадение критериев поиска в лексеме;
   2. Запись совпавших словоформ и их положений в исходном тексте в новый список;
8. Отображение таблицы результатов поиска;
9. Подсвечивание найденных словоформ в исходном тексте и в таблице результатов;
10. Конец.

**Структурно-функциональная схема разработанного приложения**:

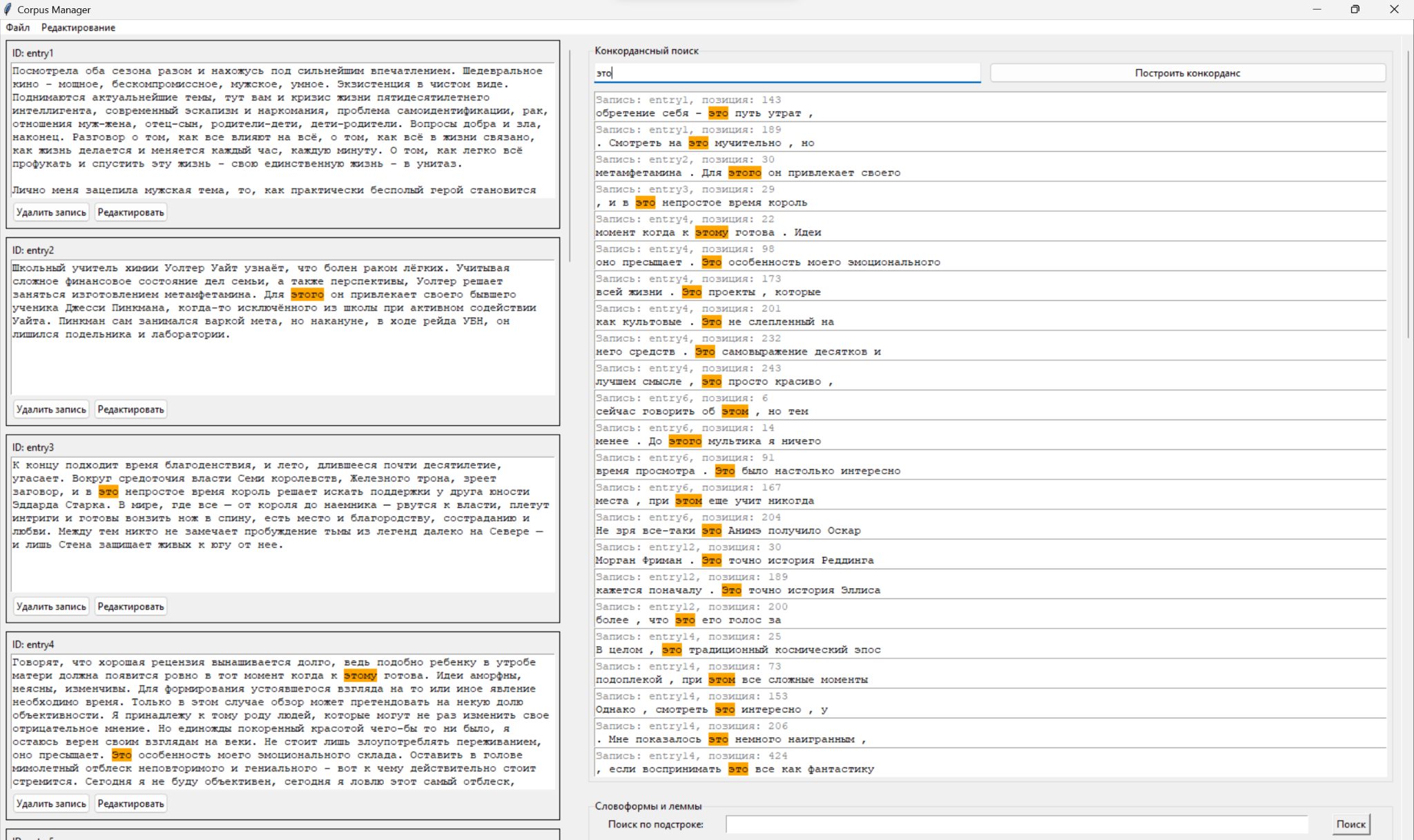


**Интерфейс программы:**

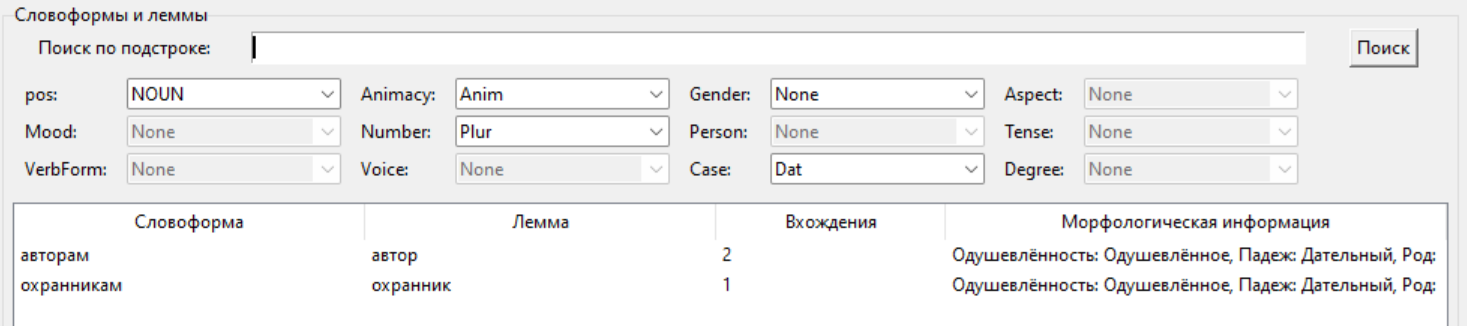
Главное окно программы



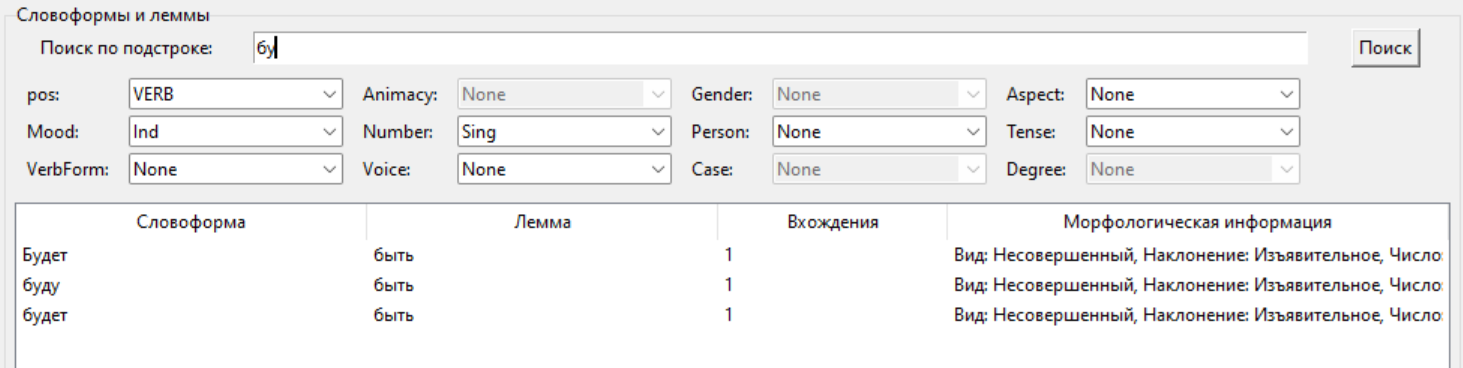
Конкордансный поиск



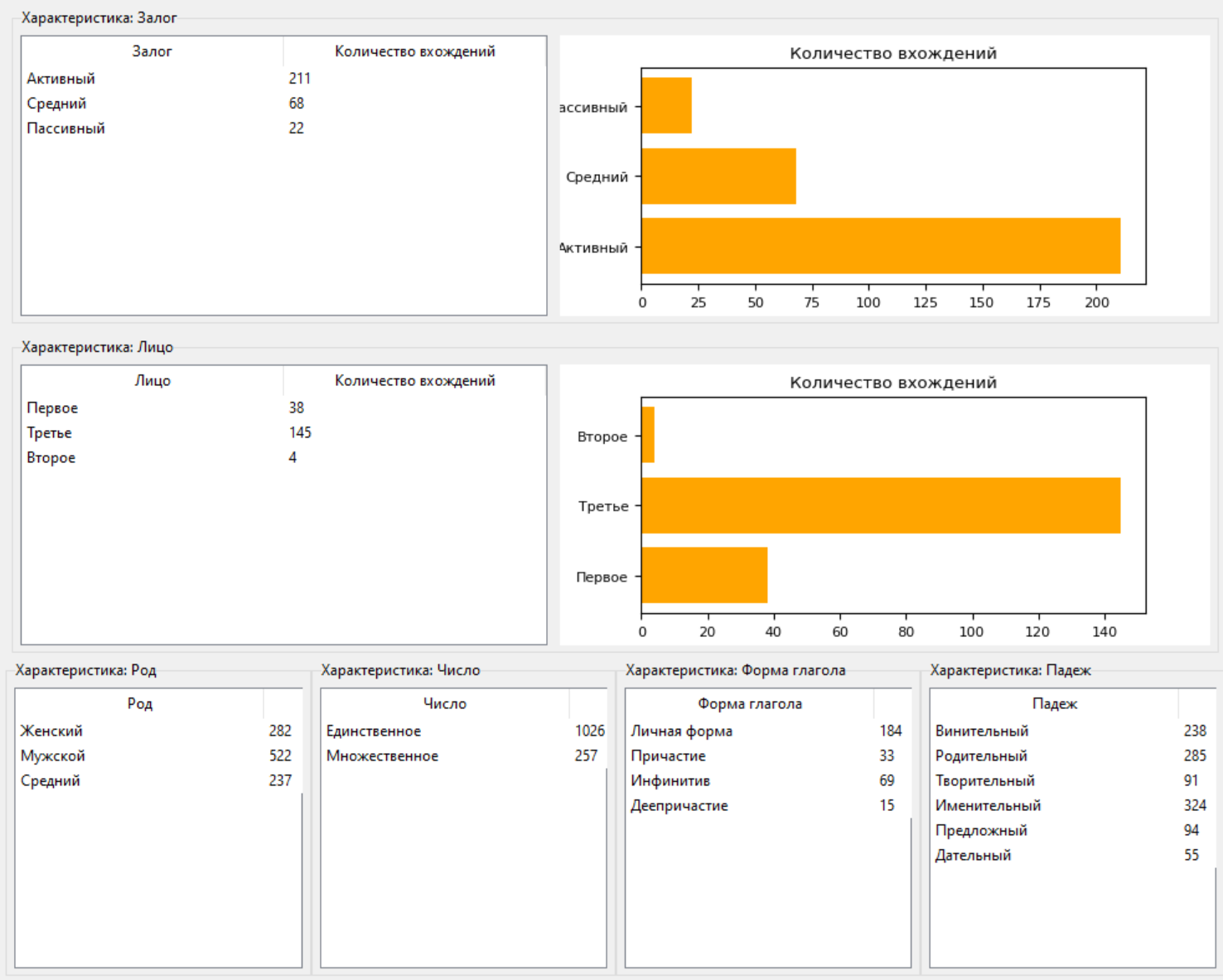
Поиск по морфологическим характеристикам



Поиск по подстроке



Таблицы и диаграммы частотных характеристик словоформ



**Описание структур хранения данных**:

В проекте реализованы классы для представления основных структур данных.

*Класс CorpusEntry*

отвечает за представление и обработку отдельной записи (текста) корпуса.

Атрибуты класса:

* id: тип данных - int;
* text: тип данных - string;
* bibliographic: тип данных - string;
* typological: тип данных - string;
* tokens: тип данных - WordToken.

*Класс CorpusManager*

отвечает за хранение записей корпуса по идентификатору.

Атрибуты класса:

* entries: тип данных - словарь.

*Класс WordToken*

хранит информацию о слове как единице корпуса текста.

Атрибуты класса:

* text: тип данных - string;
* lemma: тип данных - string;
* pos: тип данных - string;
* feats: тип данных - string;
* bibliographic: тип данных - string;
* typological: тип данных - string;
* index: тип данных - int.

**Описание библиотеки natasha**:

Библиотека Natasha решает базовые задачи обработки естественного русского языка: сегментация на токены и предложения, морфологический и синтаксический анализ, лемматизация, извлечение именованных сущностей. Для новостных статей качество на всех задачах сравнимо или превосходит существующие решения. Библиотека поддерживает Python 3.5+ и PyPy3, не требует GPU, зависит только от NumPy.

Она основана на принципах машинного обучения и использует различные модели для обработки текстов.

Для использования в работе была выбрана за удобство использования и достойные результаты в обработке текстов русского языка.

### *Принцип работы*

Natasha работает на основе методов обработки естественного языка (NLP), включая:

* **Морфологический анализ**: анализ грамматических свойств слов.
* **Сегментация**: разбиение текста на предложения и токены.
* **Извлечение сущностей**: идентификация и классификация именованных объектов (например, людей, организаций, географических мест).

### *Основные компоненты, используемые в программной реализации:*

1. **Doc** - основной класс для представления документа (текста). Он хранит текст, разбиение на предложения и токены, а также морфологические и синтаксические аннотации. Вызов иных методов пакета определяет необходимые атрибуты объектов, содержащихся в данном классе.
2. **MorphVocab** - управляет морфологическими словарями, содержащими информацию о формах слов, их грамматических категориях и возможных изменениях. Позволяет выполнять морфологический анализ, определяя свойства слов и их формы.
3. **Segmenter** - сегментатор разбивает текст на предложения и токены. Помогает в выделении смысловых единиц из потока текста, что необходимо для дальнейшего анализа. Использует правила и модели для точного разбиения текста.
4. **NewsEmbedding** - загружает предобученные векторные представления слов (эмбеддинги) на основе отобранных новостных текстов. Эмбеддинги позволяют моделям выявлять семантические отношения между словами.
5. **NewsMorphTagger** - морфологический теггер, который использует модель для автоматической аннотации слов в тексте с их морфологическими свойствами. Основан на предобученных моделях и может эффективно обрабатывать текст, определяя части речи и другие грамматические характеристики.

**Описание библиотеки tkinter:**

Tkinter — пакет для языка Python, необходимый для создания пользовательских интерфейсов средствами библиотеки Tk. Библиотека Tk написана на языке программирования Tcl и содержит в себе компоненты GUI.

### *Основные компоненты tkinter, используемые в программной реализации:*

* Button - для создания кнопок и обработчиков событий нажатий;
* Frame - для компоновки виджетов и их размещения на интерфейсе;
* Label - для отображения текстовых меток;
* Entry - для ввода критерия фильтрации;
* TreeView - для отображения словаря в виде таблицы;
* PanelWindow - для компоновки интерфейса;
* Messagebox - для отображения всплывающих окон.

**Описание библиотеки json:**

json – модуль для Python, который предоставляет возможность сериализовать и десериализовать объекты. Сериализация – процесс преобразования объекта в поток байтов, который затем может быть сохранен в файл или передан через сеть. Десериализация – обратный процесс, преобразования потока байтов в объект.

### *Основные компоненты, используемые в программной реализации:*

В данной лабораторной работе компоненты модуля json использовались для сериализации и десериализации:

* json.load - для преобразования содержимого файла в объект CorpusEntry;
* json.dump - для преобразования объекта CorpusEntry в поток байтов.

**Описание библиотеки string:**

Данный модуль предназначен для расширенного функционала по работе со строками в ЯП Python. В программе применяется string.punctuation, как функциональная часть кода, отвечающая за очистку текста от знаков пунктуации.

**Описание библиотеки time:**

Данный модуль предназначен для работы с реальным временем. В разрабатываемой программе применяется time.time() в качестве меток для отсчета времени работы программы.

**Описание библиотеки matplotlib:**

Данная библиотека применяется для работы с графиками в ЯП Python. С помощью данного модуля был построен график зависимости количества слов от времени работы программы. Также функционал библиотеки интегрирован с библиотекой tkinter для построения столбчатых диаграмм в графическом интерфейсе.

**Описание библиотеки os:**

Модуль os предоставляет множество функций для работы с операционной системой, причём их поведение, как правило, не зависит от ОС, поэтому программы остаются переносимыми. В данном приложении используется для чтения файлов из директории и передачи в функцию запуска тестов производительности.

**Оценка быстродействия приложения**:

Зависимость времени обработки от количества слов представлена на графике ниже. Тестирование производилось на наборе из 10 текстов. Оценка быстродействия приложения производилась на процессоре Intel Core i5-11300H 3.1 ГГц.



**Вывод**

В ходе работы была спроектирована и разработана программа, представляющая собой корпусный менеджер, обеспечивающий базовую функциональность для работы с корпусом текстов с указанием частоты встречаемости некоторых характеристик лексем и их морфологических признаков. Пользователю предоставляется возможность редактирования и сохранения корпуса текстов. В ходе выполнения работы был получен опыт работы с такими библиотеками для Python, как Natasha, Tkinter и другие, а также получены знания основных понятий и задач в области NLP, такие как лемматизация, разбиение на токены и другие.