МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

**ИНТЕЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ**

ОТЧЁТ

По лабораторной работе № 2

Выполнил:

Студент группы ИИ-22

Копанчук Евгений Романович

Проверил:

Булей Е. В.

Брест – 2024

**Ход работы**

**Задание:** Разработка корпусного менеджера

Разработать приложение, которое предоставляет пользователю возможность обрабатывать фрагменты текста на естественном языке, запрашивая различные частотные и морфологические характеристики словоформ.

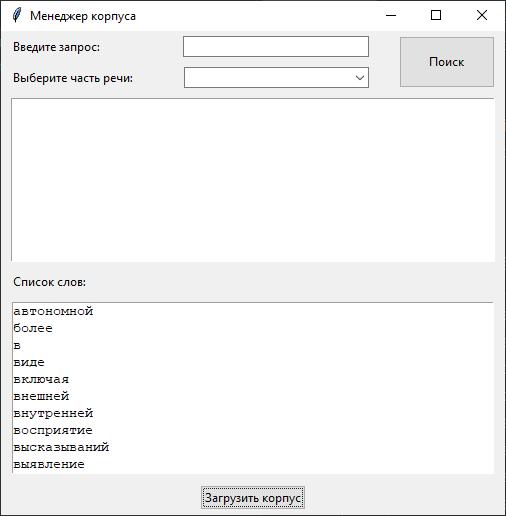
**Требования:**

***Входные данные:*** Пользователь вводит фрагмент текста (фразу или слово) на естественном языке в качестве запроса к корпусному менеджеру.

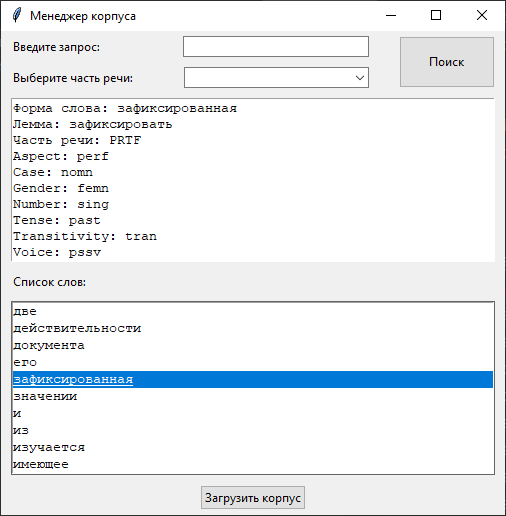
***Выходные данные:*** Приложение предоставляет следующие выходные данные:

* Частотные характеристики словоформ и лексем.
* Грамматические категории.
* Леммы слов.
* Морфологические характеристики словоформ.

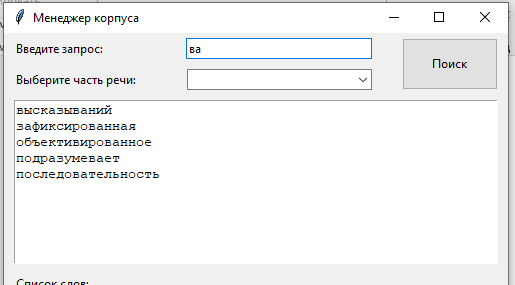
***Взаимодействие с пользователем:*** Пользователь взаимодействует с приложением через графический интерфейс, который должен быть интуитивно-понятным и дружественным для пользователя.



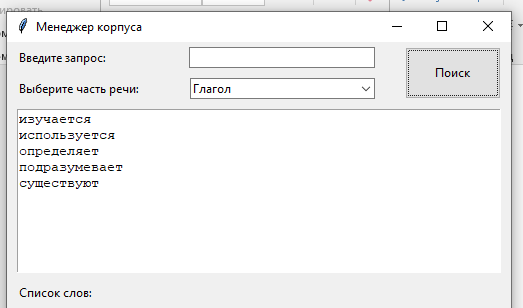
*Приложение с загруженным корпусом текста*



*Просмотр характеристик слова*

**

*Поиск по вхождению подстроки в лексеме*

**

*Поиск по части речи*

**Код программы:**

import tkinter as tk

from tkinter import filedialog

from tkinter import ttk

import os

import nltk

from pymorphy3 import MorphAnalyzer

from docx import Document

import re

class CorpusManagerGUI:

def \_\_init\_\_(self, master):

self.master = master

self.master.title("Менеджер корпуса")

self.label = ttk.Label(master, text="Введите запрос:")

self.label.grid(row=0, column=0, padx=10, pady=5, sticky="w")

self.query\_entry = ttk.Entry(master, width=30)

self.query\_entry.grid(row=0, column=1, padx=10, pady=5)

self.pos\_label = ttk.Label(master, text="Выберите часть речи:")

self.pos\_label.grid(row=1, column=0, padx=10, pady=5, sticky="w")

self.pos\_combobox = ttk.Combobox(master, values=["", "Существительное", "Прилагательное", "Глагол", "Наречие"],

width=27)

self.pos\_combobox.grid(row=1, column=1, padx=10, pady=5)

self.search\_button = ttk.Button(master, text="Поиск", command=self.search)

self.search\_button.grid(row=0, column=2, rowspan=2, padx=10, pady=5, sticky="nsew")

self.results\_text = tk.Text(master, height=10, width=60)

self.results\_text.grid(row=2, column=0, columnspan=3, padx=10, pady=5)

self.word\_list\_label = ttk.Label(master, text="Список слов:")

self.word\_list\_label.grid(row=3, column=0, padx=10, pady=5, sticky="w")

self.word\_list\_box = tk.Listbox(master, height=10, width=60, font=("Courier", 10))

self.word\_list\_box.grid(row=4, column=0, columnspan=3, padx=10, pady=5)

self.word\_list\_box.bind("<<ListboxSelect>>", self.show\_word\_info)

self.load\_corpus\_button = ttk.Button(master, text="Загрузить корпус", command=self.load\_corpus)

self.load\_corpus\_button.grid(row=5, column=0, columnspan=3, padx=10, pady=5)

self.analyzer = MorphAnalyzer()

self.corpus\_words = []

def search(self):

query = self.query\_entry.get()

pos = self.pos\_combobox.get()

results = self.search\_corpus(query, pos)

self.display\_results(results)

def search\_corpus(self, query, pos):

filtered\_words = self.corpus\_words

if pos:

if pos == "Существительное":

pos = "NOUN"

elif pos == "Прилагательное":

pos = "ADJF"

elif pos == "Глагол":

pos = "VERB"

elif pos == "Наречие":

pos = "ADVB"

else:

pos = None

if pos:

filtered\_words = [word\_data for word\_data in filtered\_words if word\_data['pos'] == pos]

results = [word\_data['wordform'] for word\_data in filtered\_words if query.lower() in word\_data['wordform'].lower()]

return results

def display\_results(self, results):

self.results\_text.delete(1.0, tk.END)

for result in results:

self.results\_text.insert(tk.END, result + "\n")

def load\_corpus(self):

file\_path = filedialog.askopenfilename(initialdir=os.getcwd(), title="Выберите файл корпуса",

filetypes=[("Документы Word", "\*.docx")])

if file\_path:

self.load\_docx\_corpus(file\_path)

def load\_docx\_corpus(self, file\_path):

doc = Document(file\_path)

text = "\n".join([paragraph.text for paragraph in doc.paragraphs]).lower()

pattern = re.compile(r'[а-яА-Я]+')

words = list(set(pattern.findall(text)))

self.corpus\_words = [self.analyze\_word(word) for word in words]

self.display\_word\_list()

def analyze\_word(self, word):

parsed\_word = self.analyzer.parse(word)[0]

return {

"wordform": parsed\_word.word,

"lemma": parsed\_word.normal\_form,

"pos": parsed\_word.tag.POS,

"animacy": parsed\_word.tag.animacy,

"aspect": parsed\_word.tag.aspect,

"case": parsed\_word.tag.case,

"gender": parsed\_word.tag.gender,

"involvement": parsed\_word.tag.involvement,

"mood": parsed\_word.tag.mood,

"number": parsed\_word.tag.number,

"person": parsed\_word.tag.person,

"tense": parsed\_word.tag.tense,

"transitivity": parsed\_word.tag.transitivity,

"voice": parsed\_word.tag.voice

}

def display\_word\_list(self):

self.word\_list\_box.delete(0, tk.END)

self.corpus\_words = sorted(self.corpus\_words, key=lambda x: x['wordform'].lower())

for word\_data in self.corpus\_words:

self.word\_list\_box.insert(tk.END, word\_data['wordform'])

def show\_word\_info(self, event):

selected\_index = self.word\_list\_box.curselection()

if selected\_index:

selected\_index = int(selected\_index[0])

word\_data = self.corpus\_words[selected\_index]

info\_text = f"Форма слова: {word\_data['wordform']}\n"

info\_text += f"Лемма: {word\_data['lemma']}\n"

info\_text += f"Часть речи: {word\_data['pos']}\n"

for key, value in word\_data.items():

if value is not None and key not in ('wordform', 'lemma', 'pos'):

info\_text += f"{key.capitalize()}: {value}\n"

self.results\_text.delete(1.0, tk.END)

self.results\_text.insert(tk.END, info\_text)

def main():

nltk.download('punkt')

root = tk.Tk()

app = CorpusManagerGUI(root)

root.mainloop()

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

main()