МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

ИНТЕЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

ОТЧЁТ

По лабораторной работе № 5

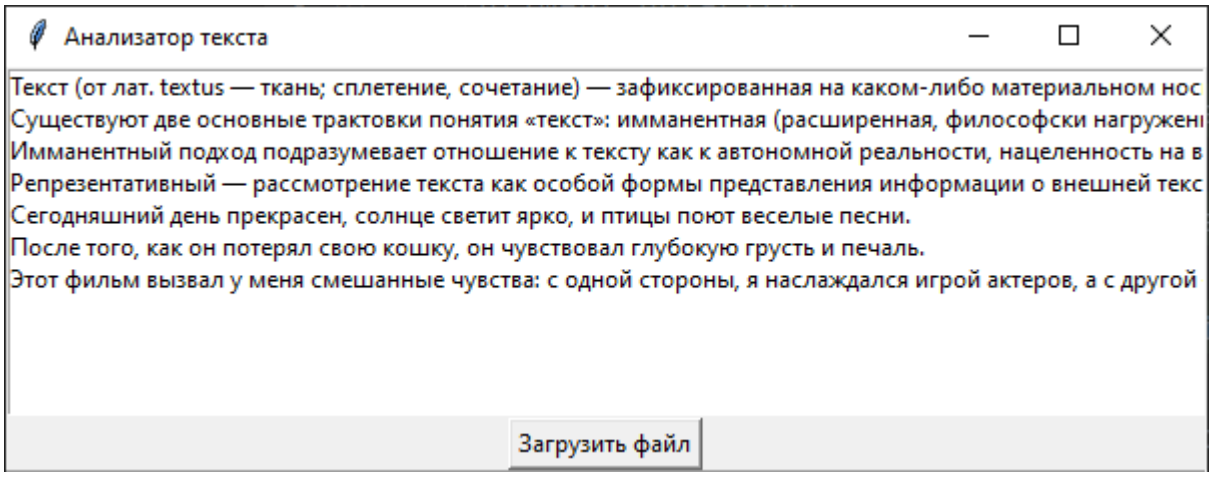
Выполнил:
Студент группы ИИ-22
Копанчук Евгений Романович
Проверил:
Булей Е. В.

Брест – 2024

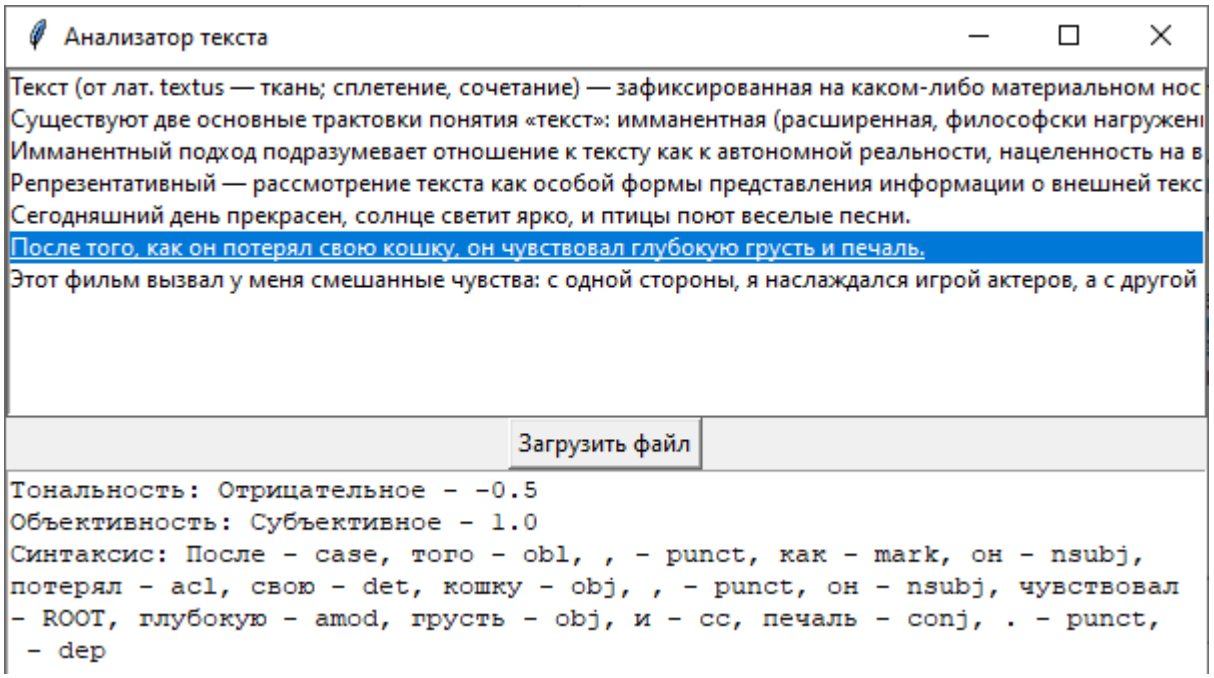
Ход работы

Задание:

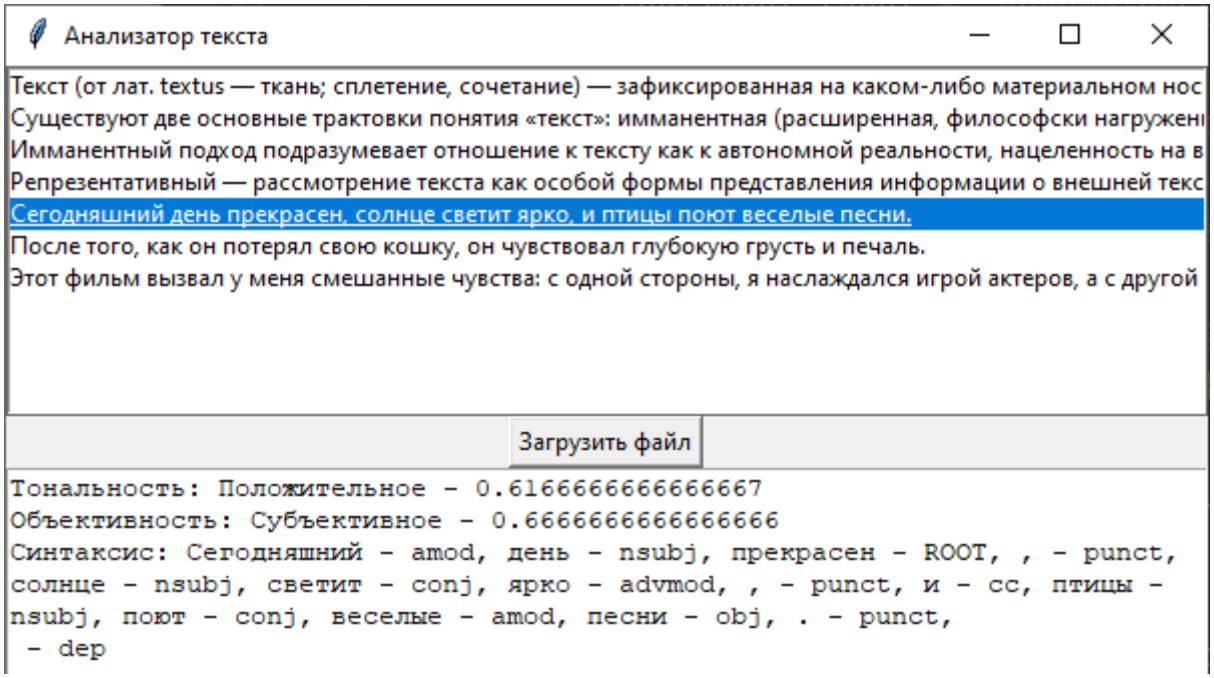
1. *Входные данные* – текст заданного естественного языка
2. *Выходные данные* – структуры, полученные при проведении автоматического семантико-синтаксического анализа предложений входного текста
3. Взаимодействие с пользователем посредством графического интерфейса



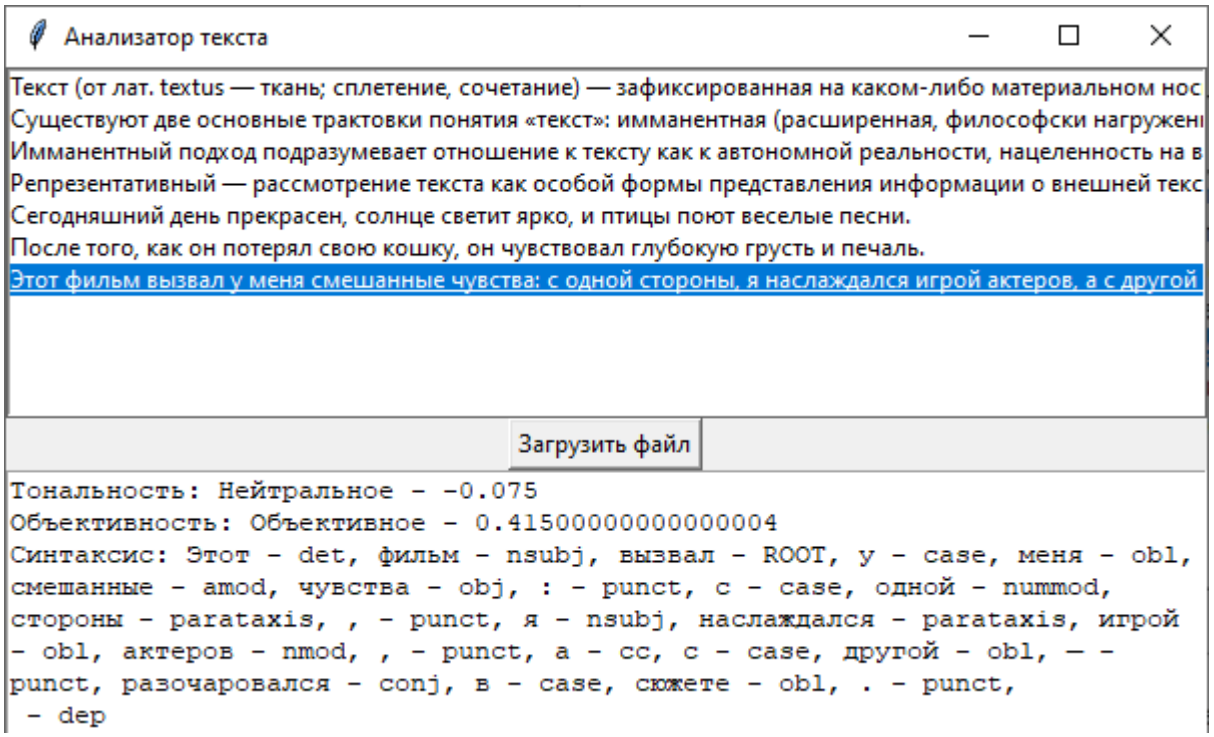
Приложение с загруженным текстом



Анализ предложения 1



Анализ предложения 2



Анализ предложения 3

Код программы:

```
import tkinter as tk
from tkinter import filedialog
from docx import Document
import spacy
from textblob import TextBlob
from transformers import MarianMTModel, MarianTokenizer


class TextAnalyzerApp:
    def __init__(self, master):
        self.master = master
        master.title("Анализатор текста")
        master.geometry("600x600")
        self.sentences_listbox = tk.Listbox(master, width=50)
```

```
        self.sentences_listbox.pack(side=tk.TOP, fill=tk.BOTH, expand=True)
        self.sentences_listbox.bind("<<ListboxSelect>>", self.show_analysis)
        self.analysis_text = tk.Text(master, wrap="word")
        self.analysis_text.pack(side=tk.BOTTOM, fill=tk.BOTH, expand=True)
        self.load_button = tk.Button(master, text="Загрузить файл", command=self.load_file)
        self.load_button.pack(side=tk.BOTTOM)
        self.tokenizer = spacy.load('ru_core_news_sm')
        self.model_ru_en = MarianMTModel.from_pretrained("Helsinki-NLP/opus-mt-ru-en")
        self.tokenizer_ru_en = MarianTokenizer.from_pretrained("Helsinki-NLP/opus-mt-ru-en")

    def load_file(self):
        file_path = filedialog.askopenfilename(filetypes=[("Word files", "*.docx")])
        if file_path:
            doc = Document(file_path)
            text = ""
            for paragraph in doc.paragraphs:
                text += paragraph.text + "\n"
            self.analyze_text(text)

    def analyze_text(self, text):
        self.sentences_listbox.delete(0, tk.END)

        doc = self.tokenizer(text)
        for sentence in doc.sents:
            self.sentences_listbox.insert(tk.END, sentence.text)

    def translate_ru_en(self, text):
        inputs = self.tokenizer_ru_en(text, return_tensors="pt", padding=True, truncation=True)
        outputs = self.model_ru_en.generate(**inputs)
        translated_text = self.tokenizer_ru_en.batch_decode(outputs, skip_special_tokens=True)
        return translated_text[0]

    def show_analysis(self, event):
        selected_sentence_index = self.sentences_listbox.curselection()[0]
        selected_sentence = self.sentences_listbox.get(selected_sentence_index)

        en_selected_sentence = self.translate_ru_en(selected_sentence)

        blob = TextBlob(en_selected_sentence)
        sentiment = blob.sentiment
        sentiment_label = sentiment.polarity
        if sentiment_label > 0.1:
            sentiment_label = "Положительное - " + str(sentiment.polarity)
        elif sentiment_label < -0.1:
            sentiment_label = "Отрицательное - " + str(sentiment.polarity)
        else:
            sentiment_label = "Нейтральное - " + str(sentiment.polarity)

        subjectivity_label = "Субъективное - " + str(sentiment.subjectivity) if
sentiment.subjectivity > 0.5 else "Объективное - " + str(sentiment.subjectivity)
        analysis_result = f"Тональность: {sentiment_label}\nОбъективность: {subjectivity_label}"

        syntax_analysis = self.get_syntax_analysis(selected_sentence)
        analysis_result += f"\nСинтаксис: {syntax_analysis}"

        self.analysis_text.delete("1.0", tk.END)
        self.analysis_text.insert(tk.END, analysis_result)

    def get_syntax_analysis(self, sentence):
        doc = self.tokenizer(sentence)
        syntax_info = ', '.join([f"{token.text} - {token.dep_}" for token in doc])
        return syntax_info


def main():
    root = tk.Tk()
    app = TextAnalyzerApp(root)
    root.mainloop()


if __name__ == "__main__":
    main()
```