

Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №1

За шестой семестр

По дисциплине: «Естественно-языковой интерфейс ИС»

Тема: «Разработка автоматизированной системы формирования словаря естественного языка»

Выполнил:

Студент 3 курса

Группы ИИ-21(II)

Кирилович А. А

Проверила:

Якимук А. В.

Ход работы

Список слов, упорядоченный по алфавиту и включающий только лексемы с дополнительно оформленными записями для образования словоформ. В этих записях должна храниться следующая информация: основа слова; часть речи; окончания слова, соотнесенные с соответствующей морфологической информацией: род, падеж, число и т.п.

Словарь естественного языка

Введите слово:

Добавить слово

Выбрать файл

Удалить выбранное слово

Сгенерировать формы

автономной (автономный, ADJF, gent, femn, sing)

более (более, ADVB)

в (в, PREP)

виде (вид, NOUN, loc, masc, sing)

включая (включая, PREP)

внешней (внешний, ADJF, gent, femn, sing)

внутренней (внутренний, ADJF, gent, femn, sing)

восприятие (восприятие, NOUN, nomn, neut, sing)

высказываний (высказывание, NOUN, gent, neut, plur)

выявление (выявление, NOUN, nomn, neut, sing)

Исходная форма: автономный

Часть речи: ADJF

Падеж: gent

Род: femn

Число: sing



```
class Word:
    def __init__(self, word):
        self.word = word
        self.morphy = []

    morph_analyzer = pymorphy3.MorphAnalyzer()
    parsed_word = morph_analyzer.parse(word)[0]

    tags = [
        {'name': 'Исходная форма', 'value': parsed_word.normal_form},
        {'name': 'Часть речи', 'value': parsed_word.tag.POS},
        {'name': 'Падёж', 'value': parsed_word.tag.case},
```

```

        {'name': 'Род', 'value': parsed_word.tag.gender},
        {'name': 'Число', 'value': parsed_word.tag.number},
        {'name': 'Время', 'value': parsed_word.tag.tense},
        {'name': 'Вид', 'value': parsed_word.tag.aspect},
        {'name': 'Лицо', 'value': parsed_word.tag.person},
        {'name': 'Наклонение', 'value': parsed_word.tag.mood},
        {'name': 'Залог', 'value': parsed_word.tag.voice}
    ]

    for tag in tags:
        if tag['value'] is not None:
            self.morphy.append(tag)

class Application(tk.Tk):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.title("Словарь естественного языка")
        self.geometry("655x645")

        self.words = []

        self.word_label = ttk.Label(self, text="Введите слово:")
        self.word_label.grid(row=0, column=0, padx=5, pady=5, sticky=tk.W)

        self.word_entry = ttk.Entry(self)
        self.word_entry.grid(row=0, column=1, padx=5, pady=5)

        self.add_word_button = ttk.Button(self, text="Добавить слово", command=self.add_word)
        self.add_word_button.grid(row=0, column=2, padx=5, pady=5)

        self.select_file_button = ttk.Button(self, text="Выбрать файл", command=self.select_file)
        self.select_file_button.grid(row=0, column=3, padx=5, pady=5)

        self.delete_word_button = ttk.Button(self, text="Удалить выбранное слово", command=self.delete_word)
        self.delete_word_button.grid(row=1, column=0, columnspan=2, padx=5, pady=5, sticky=tk.W+tk.E)

        self.generate_forms_button = ttk.Button(self, text="Сгенерировать формы",
command=self.generate_forms)
        self.generate_forms_button.grid(row=1, column=2, columnspan=2, padx=5, pady=5, sticky=tk.W+tk.E)

        self.word_listbox = tk.Listbox(self, selectmode=tk.SINGLE)
        self.word_listbox.grid(row=2, column=0, columnspan=4, padx=5, pady=5, sticky=tk.W+tk.E+tk.N+tk.S)
        self.word_listbox.bind("<<ListboxSelect>>", self.show_word_info)

        self.word_info_text = tk.Text(self, wrap="word")
        self.word_info_text.grid(row=3, column=0, columnspan=4, padx=5, pady=5, sticky=tk.W+tk.E+tk.N+tk.S)

    def add_word(self):
        word = self.word_entry.get().strip()
        if word:
            if word not in [w.word for w in self.words]:
                word_object = Word(word)
                self.words.append(word_object)
                self.sort_words_list()
                word_info = f"{word} ({', '.join(tag['value'] for tag in word_object.morphy)})"
                self.word_listbox.insert(tk.END, word_info)

    def delete_word(self):
        selected_index = self.word_listbox.curselection()
        if selected_index:
            del self.words[selected_index[0]]
            self.word_listbox.delete(selected_index)

    def generate_forms(self):
        selected_index = self.word_listbox.curselection()
        if selected_index:
            word = self.words[selected_index[0]].word
            morph_analyzer = pymorphy3.MorphAnalyzer()
            parsed_word = morph_analyzer.parse(word)[0]

            forms = parsed_word.lexeme
            self.word_info_text.delete(1.0, tk.END)
            for form in forms:
                self.word_info_text.insert(tk.END, f"{form.word}\n")

    def select_file(self):
        file_path = filedialog.askopenfilename(filetypes=[("Word files", "*.docx"), ("All files", "*.*")])
        if file_path:
            self.load_from_docx(file_path)

```

```

def load_from_docx(self, filename):
    text = docx2txt.process(filename).lower()

    pattern = re.compile(r'[a-яА-Я]+' )
    words = pattern.findall(text)

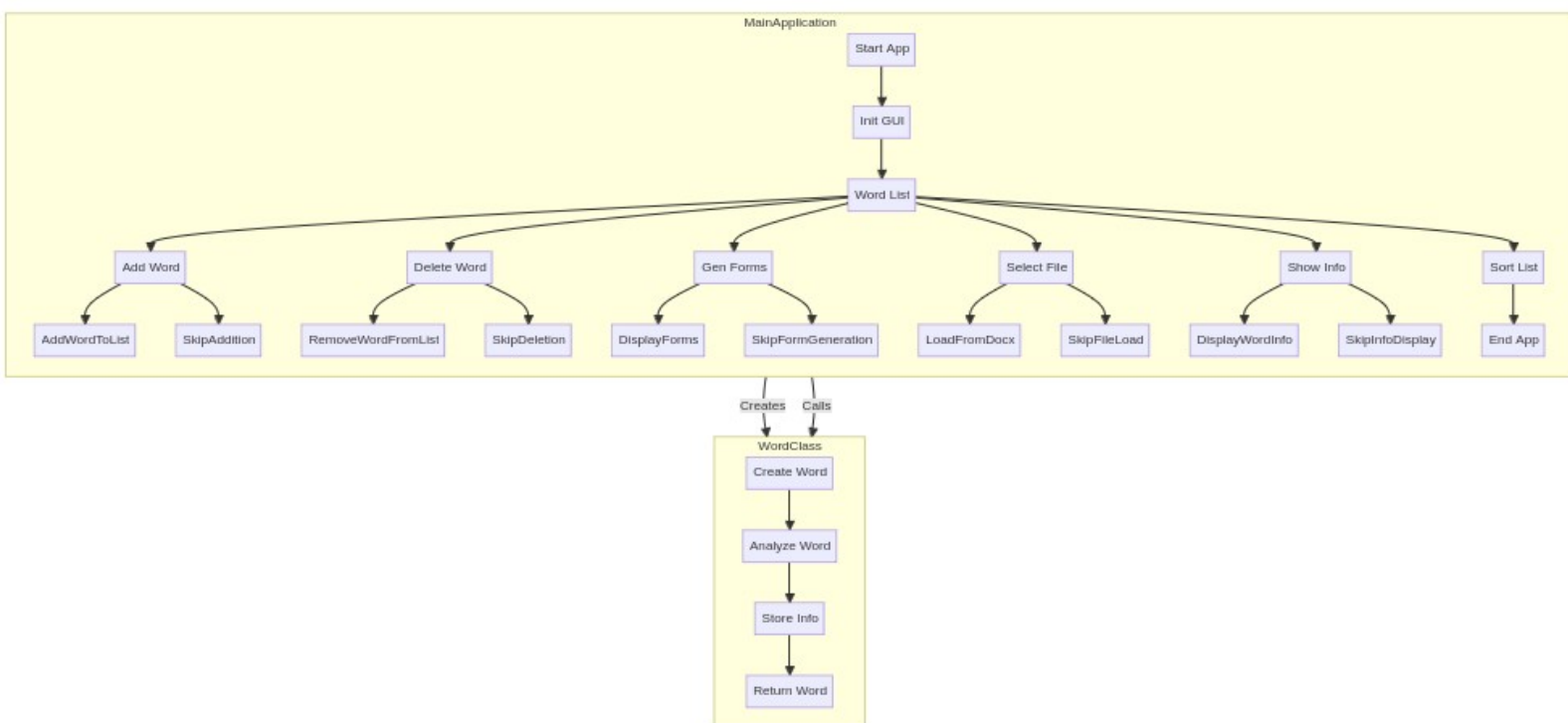
    for word in words:
        if word not in [w.word for w in self.words]:
            word_object = Word(word)
            self.words.append(word_object)
            self.sort_words_list()
            word_info = f"{word} ({', '.join(tag['value'] for tag in word_object.morphy)})"
            self.word_listbox.insert(tk.END, word_info)

def sort_words_list(self):
    self.words.sort(key=lambda x: x.word.lower())
    self.word_listbox.delete(0, tk.END)
    for word_object in self.words:
        word_info = f"{word_object.word} ({', '.join(tag['value'] for tag in word_object.morphy)})"
        self.word_listbox.insert(tk.END, word_info)

def show_word_info(self, event):
    selected_index = self.word_listbox.curselection()
    if selected_index:
        word = self.words[selected_index[0]]
        self.word_info_text.delete(1.0, tk.END)
        for tag in word.morphy:
            self.word_info_text.insert(tk.END, f"{tag['name']}: {tag['value']}\n")

if __name__ == "__main__":
    app = Application()
    app.mainloop()

```



Вывод: в ходе выполнения лабораторной работы освоил принципы разработки прикладных сервисных программ для решения задачи автоматического лексического и лексико-грамматического анализа текста естественного языка.