Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №7-8

По дисциплине: «Естественно-языковой интерфейс ИС» Темы: «Разработка систем анализа речи», «Разработка систем синтеза речи»

Выполнил:

Студент 4 курса

Группы ИИ-21

Пучинский А.А.

Проверил:

Булей Е. В.

Брест 2024

**Цель:** освоить на практике основные принципы создания систем анализа и синтеза речи.

**Лабораторная работа №7:**

**Ход работы:**



|  |  |  |  |
| --- | --- | --- | --- |
| **№**  **варианта** | **Система** | **Поддерживаемые языки** | **Предметная область** |
| 9 | Анализа | Английский, | Научные статьи по computer science |



* Задание списка операций, на которые система может реагировать.
* Организация автоматической реакции системы на речевой сигнал с уведомлением пользователя о происходящем.  Настройки (выбор) ЕЯ.

**Код программы:**

import tkinter as tk

from tkinter import ttk

import speech\_recognition as sr

import threading

class SpeechApp:

    def \_\_init\_\_(self, root):

        self.root = root

        self.root.title("Приложение распознавания речи")

        self.root.configure(bg="#2E2E2E")  # Изменение фона

        # Заголовок

        self.label = tk.Label(root, text="Нажмите кнопку и говорите:", font=("Arial", 16), bg="#2E2E2E", fg="white")

        self.label.pack(pady=20)

        # Поле для вывода результата

        self.result\_text = tk.Text(root, height=10, width=50, bg="#3C3C3C", fg="white", font=("Arial", 12))

        self.result\_text.pack(pady=20)

        # Выпадающий список для выбора языка

        self.language\_var = tk.StringVar(value='ru-RU')  # По умолчанию русский

        self.language\_label = tk.Label(root, text="Выберите язык:", font=("Arial", 12), bg="#2E2E2E", fg="white")

        self.language\_label.pack(pady=5)

        self.language\_combobox = ttk.Combobox(root, textvariable=self.language\_var, font=("Arial", 12),

                                              values=['ru-RU', 'en-US', 'de-DE', 'fr-FR'])

        self.language\_combobox.pack(pady=5)

        # Кнопка для распознавания речи

        self.recognize\_button = tk.Button(root, text="Распознать речь", command=self.start\_recognition\_thread,

                                          font=("Arial", 12), bg="#4CAF50", fg="white")

        self.recognize\_button.pack(pady=20)

    def start\_recognition\_thread(self):

        # Запуск функции распознавания речи в отдельном потоке

        threading.Thread(target=self.recognize\_speech).start()

    def recognize\_speech(self):

        recognizer = sr.Recognizer()

        recognizer.pause\_threshold = 1

        with sr.Microphone() as source:

            self.label.config(text="Слушаю...")

            try:

                audio = recognizer.listen(source, timeout=3, phrase\_time\_limit=12)

            except sr.WaitTimeoutError as e:

                self.label.config(text=f"Не удалось получить результаты; {e}.")

                return

            try:

                language = self.language\_var.get()  # Получаем выбранный язык

                text = recognizer.recognize\_google(audio, language=language)

                self.result\_text.delete(1.0, tk.END)  # Очистить предыдущий текст

                self.result\_text.insert(tk.END, text)  # Вставить распознанный текст

                self.label.config(text="Нажмите кнопку и говорите:")

                # Автоматическая реакция на определенные фразы

                self.respond\_to\_command(text)

            except sr.UnknownValueError:

                self.label.config(text="Не удалось распознать аудио.")

            except sr.RequestError as e:

                self.label.config(text=f"Не удалось получить результаты; {e}.")

            except sr.WaitTimeoutError as e:

                self.label.config(text=f"Не удалось получить результаты; {e}.")

    def respond\_to\_command(self, text):

        # Реакция на команды

        if "привет" in text.lower():

            self.label.config(text="Привет! Как я могу помочь?")

        elif "стоп" in text.lower():

            self.label.config(text="А я что машина?")

        elif "как дела" in text.lower():

            self.label.config(text="У меня все хорошо, спасибо!")

if \_\_name\_\_ == '\_\_main\_\_':

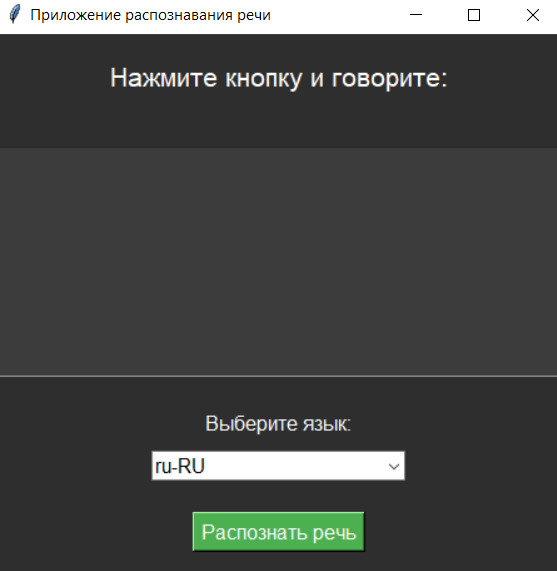
    root = tk.Tk()

    app = SpeechApp(root)

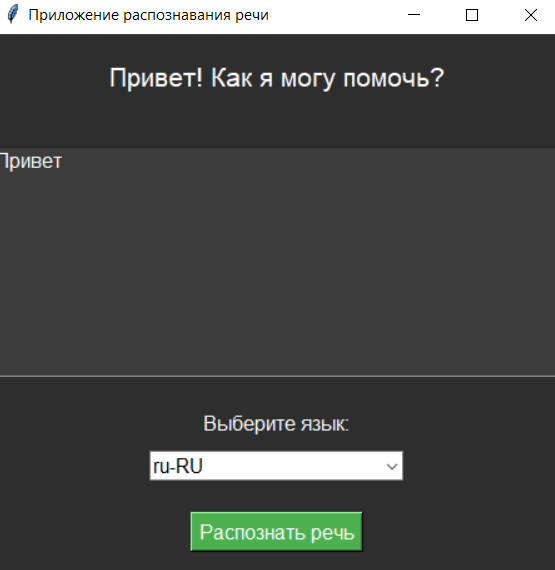
    root.mainloop()

**Результат:**

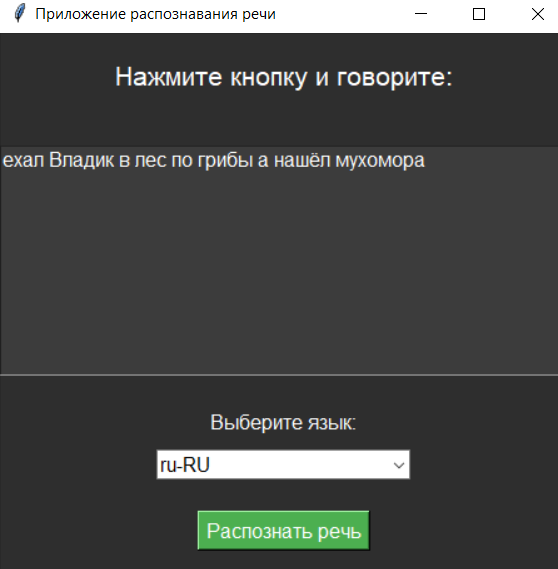
**Исходное главное меню:**



**Реакция на операцию:**



**Анализ текста:**



**Лабораторная работа №8:**

**Код программы:**

import tkinter as tk

from tkinter import ttk

import pyttsx3

class TTSApp:

    def \_\_init\_\_(self, root):

        self.root = root

        self.root.title("Система синтеза речи")

        self.root.configure(bg="#2E2E2E")

        self.engine = pyttsx3.init()

        voices = self.engine.getProperty('voices')

        for index, voice in enumerate(voices):

            print(f"Voice {index}: {voice.name}")

        # Метка для ввода текста

        self.label = tk.Label(root, text="Введите текст:", font=("Arial", 16), bg="#2E2E2E", fg="#FFFFFF")

        self.label.pack(pady=10)

        # Поле для ввода текста

        self.text\_entry = tk.Text(root, height=10, width=50, bg="#3C3C3C", fg="#FFFFFF", insertbackground="white", font=("Arial", 12))

        self.text\_entry.pack(pady=10)

        # Кнопка для воспроизведения текста

        self.speak\_button = tk.Button(root, text="ПРОИЗНЕСИТЬ", command=self.speak\_text, font=("Arial", 12), bg="#4A90E2", fg="#FFFFFF", relief="flat")

        self.speak\_button.pack(pady=10)

        # Метка для выбора голоса

        self.voice\_label = tk.Label(root, text="Выберите голос:", font=("Arial", 12), bg="#2E2E2E", fg="#FFFFFF")

        self.voice\_label.pack(pady=5)

        # Выпадающий список для выбора голоса

        self.voices = self.engine.getProperty('voices')

        self.voice\_var = tk.StringVar(value=self.voices[0].id)  # Установка первого голоса по умолчанию

        self.voice\_combobox = ttk.Combobox(root, textvariable=self.voice\_var, values=[voice.name for voice in self.voices], font=("Arial", 12), state="readonly")

        self.voice\_combobox.pack(pady=5)

        # Метка для выбора скорости

        self.speed\_label = tk.Label(root, text="Установите скорость:", font=("Arial", 12), bg="#2E2E2E", fg="#FFFFFF")

        self.speed\_label.pack(pady=5)

        # Ползунок для выбора скорости

        self.speed\_scale = tk.Scale(root, from\_=50, to=300, orient=tk.HORIZONTAL, label="Скорость (слов в минуту)", length=400, bg="#2E2E2E", fg="#FFFFFF", sliderlength=20, font=("Arial", 10))

        self.speed\_scale.set(150)  # Установка значения по умолчанию

        self.speed\_scale.pack(pady=10)

        # Метка для выбора громкости

        self.volume\_label = tk.Label(root, text="Установите громкость:", font=("Arial", 12), bg="#2E2E2E", fg="#FFFFFF")

        self.volume\_label.pack(pady=5)

        # Ползунок для выбора громкости

        self.volume\_scale = tk.Scale(root, from\_=0.0, to=1.0, resolution=0.1, orient=tk.HORIZONTAL, label="Громкость", length=400, bg="#2E2E2E", fg="#FFFFFF", sliderlength=20, font=("Arial", 10))

        self.volume\_scale.set(1.0)  # Установка значения по умолчанию

        self.volume\_scale.pack(pady=10)

    # Метод для воспроизведения текста

    def speak\_text(self):

        text = self.text\_entry.get("1.0", tk.END).strip()  # Получаем текст

        self.engine.setProperty('voice', self.get\_selected\_voice\_id())  # Устанавливаем выбранный голос

        self.engine.setProperty('rate', self.speed\_scale.get())  # Устанавливаем скорость

        self.engine.setProperty('volume', self.volume\_scale.get())  # Устанавливаем громкость

        self.engine.say(text)  # Произносим текст

        self.engine.runAndWait()  # Ждем завершения

    # Метод для получения ID выбранного голоса

    def get\_selected\_voice\_id(self):

        selected\_voice\_name = self.voice\_var.get()  # Получаем выбранное имя голоса

        for voice in self.voices:

            if voice.name == selected\_voice\_name:

                return voice.id  # Возвращаем ID выбранного голоса

        return self.voices[0].id  # Если ничего не найдено, возвращаем первый голос

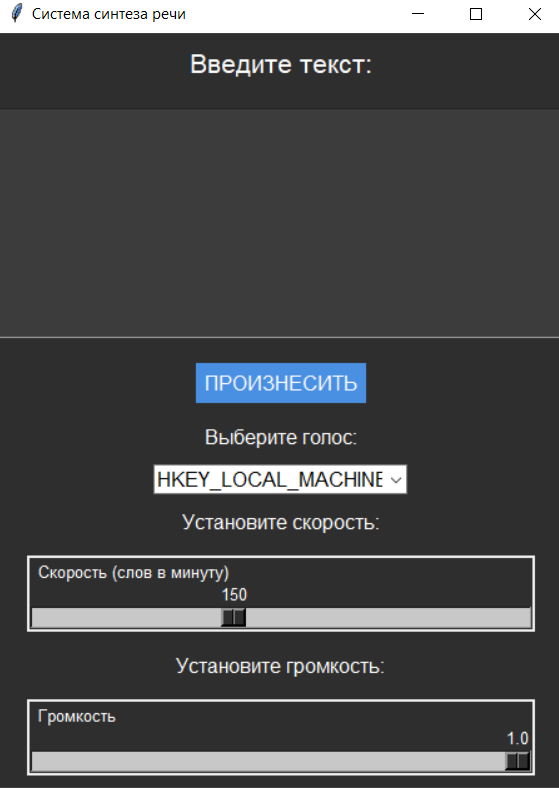
if \_\_name\_\_ == '\_\_main\_\_':

    root = tk.Tk()

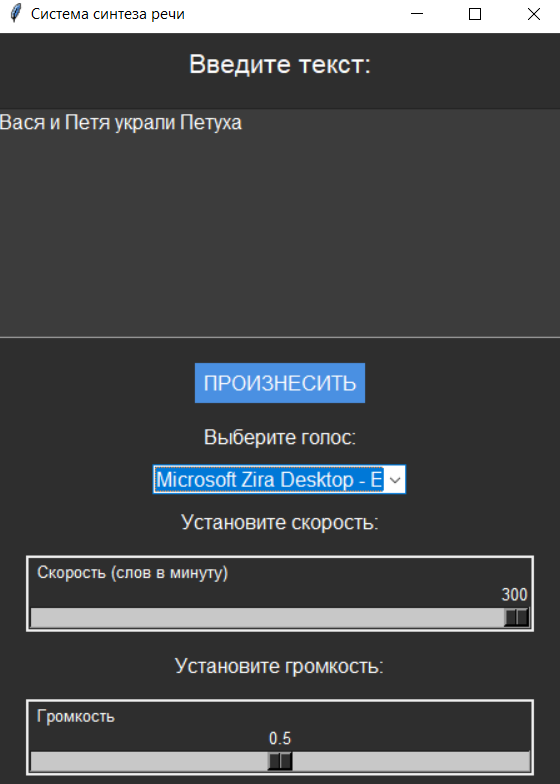
    app = TTSApp(root)

    root.mainloop()

**Исходное главное меню:**



**Изменение скорости чтения и синтез речи:**



**Вывод:** в ходе выполнения лабораторной работы освоил на практике основные принципы создания систем анализа и синтеза речи.