Итоговый проект по дисциплине “Основы алгоритмизации и программирования”

Тема проекта: “Голосовой помощник на языке программирования Python”

Выполнили:

Студенты группы ИС – 24

Касьянов Максим и Кобелев Евгений

Руководитель:

Манакова Ольга Петровна

Цель проекта

Разработать функциональность и интерфейс голосового помощника на языке программирования Python. Провести дебаги, протестировать и защитить у руководителя.

Появление идеи

При взятии проекта были разные варианты, что можно сделать, но решили остановиться на этом. Название придумали при помощи мозгового штурма, предлагая разные идеи. В итоге решили назвать Ok-Gosling, так как название всем понравилось. Перед разработкой программы решили остановиться на современно-минималистичном интерфейсе. Также продумывались варианты команд для помощника.

Поиск информации

Во время разработки были пересмотрены разные документации, видео и форумы разработчиков. В коде использовались разные библиотеки, вот основные из них:

Tkinter – библиотека для разработки интерфейса на Python.

SpeechRecognition - библиотека (API) от Google, которая переводит голос в текст.

Gtts - тоже библиотека (API) от Google, которая переводит текст в звук.

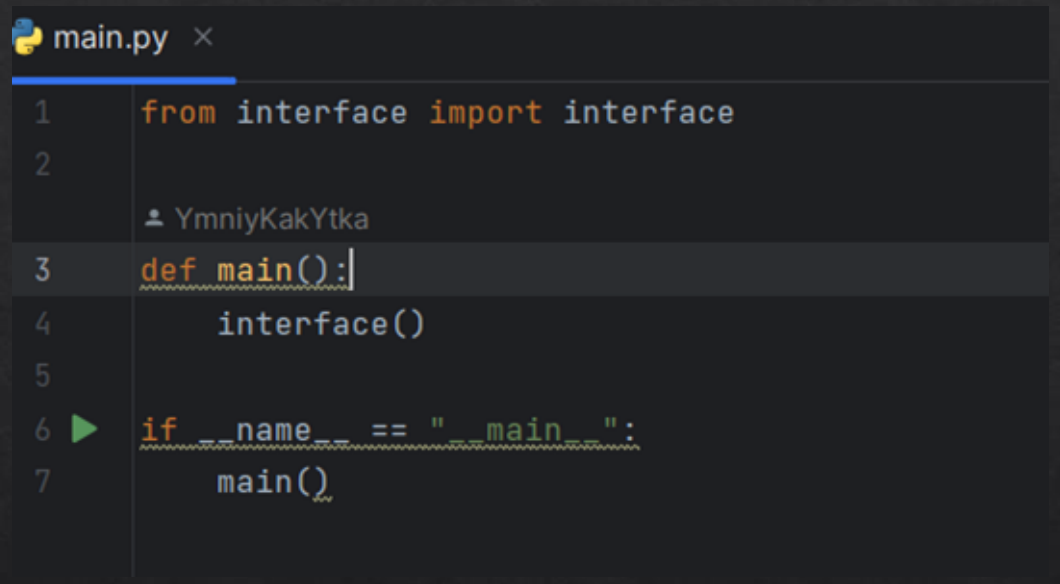
Playsound - воспроизводит mp3 и wav файлы.

Requests – библиотека для работы с API погоды OpenWeather и API анекдотов.

Webbrowser – библиотека для открытия ссылок в браузере.

Описание кода

Для запуска программы используется файл main.py, который запускает файл с интерфейсом



Описание кода интерфейса

interface.py

```python
import tkinter as tk
from command_handlers.bot_commands import startListen

YmniyKakYtka *
def interface():
    window = tk.Tk()

    # Задание заголовка окна
    window.title("Ok-Gosling")
    # Настройка размеров окна
    window.geometry("300x450")
    window.configure(bg="#333333")


    text_frame = tk.Frame(window, bg="#333333")
    text_frame.pack(fill=tk.BOTH, expand=True)

    # Загрузка и установка изображения на задний фон фрейма
    image = tk.PhotoImage(file=".\\back\\bg1.png")
    background_label = tk.Label(text_frame, image=image)
    background_label.place(x=0, y=0, relwidth=1, relheight=1)
```

При разработке интерфейса использовался Tkinter. Были созданы заголовок, задний фон, базовый размер окна, кнопка, текстовое поле и активатор функции начала слушания, при нажатии на кнопку

```python

    # Создаю метку с текстом в основном окне
    text_label = tk.Label(text_frame, text="Добро пожаловать в Ok-Gosling", fg="white", bg="#3C3C3C",
                          font=("Arial", 14))
    text_label.pack(pady=100, anchor="n")


    listen_button = tk.Button(window, text="Нажмите \n чтобы говорить", command=startListen,
                              width=20, height=2, bg="#880F0F", fg="white")

    # Установка кнопки ниже центра окна
    listen_button.place(relx=0.5, rely=0.8, anchor=tk.CENTER)

    # Запуск главного цикла обработки событий
    window.mainloop()


interface()
```

Описание кода команд

Запускается функция startListen, которая запускает функцию listen из файла bot_functions, и вызывает функцию chooseComand, которая ищет включения строки команды в тексте запроса и выбирает команду из словаря commands, далее она ищет в глобальном списке имен globals функцию с названием ключа выбранной команды.

```
YmniyKakYtka
10  def chooseComand(command):
11      for key, value in commands.items():
12          for i in value:
13              if i in command:
14                  globals()[key]()
15
16
    YmniyKakYtka
17  def startListen():
18      text = listen()
19      chooseComand(text)
```

```
commands = {
    "hello": ["привет", "здравствуй", "хэллоу", "хай"],
    "weather": ["погода"],
    "ok_gosling": ["ok гослинг", "о'кей гослинг"],
    "joke": ["анекдот", "расскажи анекдот", "хочу посмеяться"],
    "searchInBrowser": ["найди в браузере", "найди в интернете", "поиск в интернете"]
}
```

Описания кода слушания

В файле bot_functions функция слушания использует микрофон, для передачи голоса пользователя, и SpeechRecognition для перевода голоса в текст. Время ожидания 0,5 секунд. Ориентир, когда можно говорить задается звуком beepSound и выводом текста speak в консоль, при неизвестном запросе срабатывает звук beepErrorSound.

```python
speechRecognition = speech_recognition.Recognizer()
speechRecognition.pause_threshold = 0.5

beepSound = "./assets/audio/beep.wav"
beepErrorSound = "./assets/audio/beep_error.wav"

YmniyKakYtka
def listen():
    try:
        with speech_recognition.Microphone() as mic:
            speechRecognition.adjust_for_ambient_noise(source=mic, duration=0.5)
            print("Speak")
            playsound(beepSound)
            audio = speechRecognition.listen(source=mic)
            recognizedText = speechRecognition.recognize_google(audio_data=audio, language='ru-RU').lower()

            return recognizedText
    except speech_recognition.UnknownValueError:
        playsound(beepErrorSound)
```

Описание кода функций ответов

YmniyKakYtka

```
def ok_gosling():
    say("Я слушаю, чего вы хотели?")
    startListen()
```

YmniyKakYtka

```
def hello():
    say("Здарова!")
```

YmniyKakYtka

```
def joke():
    array = ["А сейчас Анекдот!", "Анекдот!", "А вот и анекдот", "Сейчас шуткану", "Смешнявка"]
    say(array[random.randint(0, len(array) - 1)])
    url = 'http://rzhunemogu.ru/RandJSON.aspx?CType=1'
    text = requests.get(url=url).text
    text = text[12:-2]
    say(text)
```

Если вызывается функция ok_gosling, то вызывается функция say с ответом помощника и функция startListen для задания вопроса. При вызывание hello, просто ответ.

Если вызывается функция joke, то вызывается функция say с случайным ответом из списка array, в url находится API с случайными анекдотами, в переменную text они записываются и убираются лишние символы. Выводится text в функцию say.

Описание кода функций ответов

Строка fetchUrl формирует URL-адрес для отправки запроса к API OpenWeatherMap. В этой строке используются предоставленные координаты широты (47.23) и долготы (39.72) для города Ростов-на-Дону. Параметр appid, содержит ключ к API OpenWeatherMap, который определен ранее в переменной OPEN_WEATHER_API_KEY. Запрос отправляется с использованием метода requests.post() и результат преобразуется в формат JSON. Затем из ответа JSON извлекаются различные данные о погоде В конце всё выводится в функции say().

```
OPEN_WEATHER_API_KEY = "8d24c9e287fbcb3e0ff18319bd2248bb"
```

```
30
    YmniyKakYtka
31  def weather():
32      fetchUrl = f"https://api.openweathermap.org/data/2.5/weather?lat={'47.23'}&lon={'39.72'}&lang=ru&appid={OPEN_WEATHER_API_KEY}"
33
34      response = requests.post(url=fetchUrl).json()
35      weatherType = f"{response['weather'][0]['description']}"
36      windSpeed = f"{response['wind']['speed']} метров в секунду"
37      temperature = f"{round(response['main']['temp'] - 273.15, 2)} градусов цельсия"
38      temperatureFeelsLike = f"{round(response['main']['feels_like'] - 273.15, 2)} градусов цельсия"
39      pressure = f"{response['main']['pressure']} миллиметров ртутного столба"
40      humidity = f"{response['main']['humidity']} процентов"
41      say(f"сейчас в Ростове-на-Дону {weatherType}, температура: {temperature}, ощущается как {temperatureFeelsLike}, "
42          f"скорость ветра: {windSpeed}, давление: {pressure}, влажность: {humidity}")
43
```

Описание кода функций ответов

С помощью say() выводится "Что нужно найти?. Затем с помощью listen() происходит прослушивание и запись в text. Далее выводится "Сейчас поищем...". Потом происходит переборка различных значений из списка с доменами. Эти значения используются для прямого открытия сайта. Если одно из значений из списка присутствует в тексте, то используется модуль webbrowser для открытия новой вкладки в браузере с введенным запросом пользователя в качестве доменного имени. Если ни одно из значений из списка не найдено в запросе, то используется модуль webbrowser для открытия новой вкладки в браузере для

YmniyKakYtka

```
def searchInBrowser():
    say("Что нужно найти?")
    text = listen()
    say("Сейчас поищем...")
    for value in ['.com', '.org', '.ru', '.net', '.gov', '.edu', '.info']:
        if value in text:
            webbrowser.open_new(
                f'https://{text}'
            )
            return

    webbrowser.open_new(
        f'https://www.google.com/search?q={text}&aqs=chrome.0.0i355i433i512j46i340i433i512l2j46i433i512j46i512l2j0i512j46i512j0i512j46i512.31386j0j7&sourceid=c
```

Возникшие ошибки

Во время тестирования возникали небольшие ошибки. При выборе голоса для голосового помощника сначала остановились на API от Амазон, но у него был лимит запросов и длинные предложения не договаривались, поэтому мы используем библиотеку Gtts. Также возникали проблемы с версиями, но проблема была устранена добавлением файла requirements, в котором содержатся библиотеки с нужными версиями. Несколько раз перерабатывался интерфейс, чтобы прийти к лучшему выбору.

Результат

В результате голосовой помощник ok-gosling и интерфейс к нему были разработаны. Во время разработки все ошибки были исправлены, проведено несколько тестов. Были изучены новые темы и новые библиотеки. Ниже приведены основные источники информации.

https://github.com/OlgaManakova2021/training-materials
https://pypi.org/project/SpeechRecognition/
https://habr.com/ru/articles/577806/
https://docs.python.org/3/library/webbrowser.html
https://habr.com/ru/articles/470938/
https://pypi.org/project/gTTS/
https://gtts.readthedocs.io/en/latest/
https://pythonru.com/biblioteki/kratkoe-rukovodstvo-po-biblioteke-python-requests
https://requests.readthedocs.io/en/latest/
https://pythonru.com/uroki/obuchenie-python-gui-uroki-po-tkinter
https://python-scripts.com/tkinter
https://pypi.org/project/playsound/
https://www.cyberforum.ru/python-beginners/thread2881835.html