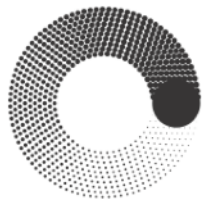


**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет информационных технологий  
Кафедра Информатики и информационных технологий**

**направление подготовки 09.04.02 «Информационные системы и  
технологии»,  
профиль «Мобильные приложения»**

**Практическая работа №4  
«Разработка программы распознавания и генерации речи»**

**Дисциплина: Искусственный интеллект в мобильных системах**

**Выполнил: студент группы 234-332**

**Киселев С.А.**

**Дата, подпись** 23.03.2025 \_\_\_\_\_

(Дата)

(Подпись)

**Проверила:** Дагаев А.Е. \_\_\_\_\_

(Оценка)

**Дата, подпись** \_\_\_\_\_

(Дата)

(Подпись)

**Замечания:**

---

---

**Москва**

**2025**

## Оглавление

Цель	3
Реализация	3
Вывод	7

## Цель

Целью работы является написание программы на Python, которая бы воспринимала 7-8 голосовых команд и реагировала на голосовую команду stop - окончить работу, используя материалы лекции "Распознавание и генерация речи в мобильных системах". Ответы программа должна выдавать в виде голосовых сообщений (желательно на русском языке, но можно и на английском).

## Реализация

В ходе данной практической работы была разработана программа на Python, которая воспринимает 9 голосовых команд:

- "открой почту" -> открывает сайт почты <https://mail.google.com/>
- "сколько времени" / "который час" / "сколько время" -> произносит текущее время
- "как тебя зовут" -> сообщает, что это голосовой ассистент
- "какие новости" / "что нового" -> читает последние новости
- "какая погода" -> произносит погоду в городе Москва
- "найди в гугле" / "загугли" -> выполняет поиск
- "расскажи анекдот" -> рассказывает анекдот
- "случайное число" -> называет случайное число от 0 до 10
- "стоп" -> завершает работу программы

Сначала устанавливаем необходимые для работы библиотеки:

```
sergey@MacBook-Pro-Sergej ~ % brew install portaudio
=> Auto-updating Homebrew...
Adjust how often this is run with HOMEBREW_AUTO_UPDATE_SECS or disable with
HOMEBREW_NO_AUTO_UPDATE. Hide these hints with HOMEBREW_NO_ENV_HINTS (see 'man brew').
=> Auto-updated Homebrew!
Updated 2 taps (homebrew/core and homebrew/cask).
=> New Formulae
darklua                                     veccore
=> New Casks
font-pretendard-gov

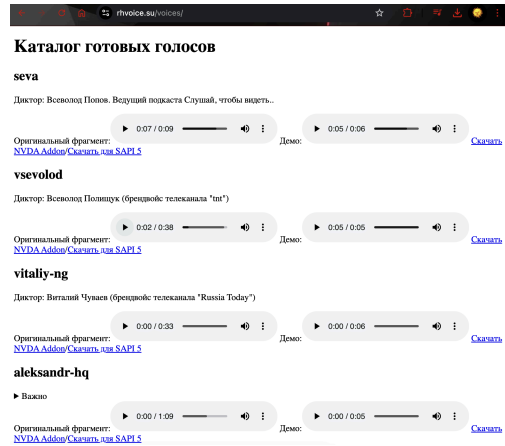
You have 16 outdated formulae and 1 outdated cask installed.

=> Downloading https://ghcr.io/v2/homebrew/core/portaudio/manifests/19.7.0-1
===== 100.0%
=> Fetching portaudio
=> Downloading https://ghcr.io/v2/homebrew/core/portaudio/blobs/sha256:8ad9f1c15a4bc9c85a9dd184b53b8f5f5d13a2458a70535bf01e54ce4f0b4bd
===== 100.0%
=> Pouring portaudio--19.7.0.arm64_t8020.macosx.bottle.1.tar.gz
=> /opt/homebrew/Cellar/portaudio/19.7.0: 34 files, 546.0KB
=> Running brew cleanup portaudio...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see 'man brew').
sergey@MacBook-Pro-Sergej ~ % pip install pyaudio
Collecting pyaudio
  Using cached PyAudio-0.2.14.tar.gz (47 kB)
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
  Building wheels for collected packages: pyaudio
    Building wheel for pyaudio (pyproject.toml) ... done
    Created wheel for pyaudio: filename=PyAudio-0.2.14-cp310-cp310-macosx_15_0_arm64.whl size=25643 sha256=a221a2e63667bf189a8ad43668d2d710066ac344d46
    Stored in directory: /Users/sergey/Library/Caches/pip/wheels/d6/21/14/0b51d41ba79e51b16295cb096ec49f334792814d545b580c5
  Successfully built pyaudio
  Installing collected packages: pyaudio
  Successfully installed pyaudio-0.2.14
sergey@MacBook-Pro-Sergej ~ %
```

```

sergey@MacBook-Pro-Sergej ~ % pip install SpeechRecognition
Collecting SpeechRecognition
  Downloading SpeechRecognition-3.14.1-py3-none-any.whl.metadata (31 kB)
Requirement already satisfied: typing-extensions in /opt/homebrew/lib/python3.10/site-packages (from SpeechRecognition) (4.12.2)
Downloading SpeechRecognition-3.14.1-py3-none-any.whl (32.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 32.9/32.9 MB 4.8 MB/s eta 0:00:00
Installing collected packages: SpeechRecognition
Successfully installed SpeechRecognition-3.14.1
sergey@MacBook-Pro-Sergej ~ % pip install gTTS
Collecting gTTS
  Downloading gTTS-2.5.4-py3-none-any.whl.metadata (4.1 kB)
Requirement already satisfied: requests<3,=>2.27 in /opt/homebrew/lib/python3.10/site-packages (from gTTS) (2.32.3)
Collecting click<8.2,>=7.1 (from gTTS)
  Downloading click-8.1.8-py3-none-any.whl.metadata (2.3 kB)
Requirement already satisfied: charset-normalizer<4,>=2.5 in /opt/homebrew/lib/python3.10/site-packages (from requests<3,=>2.27-->gTTS) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /opt/homebrew/lib/python3.10/site-packages (from requests<3,=>2.27-->gTTS) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/homebrew/lib/python3.10/site-packages (from requests<3,=>2.27-->gTTS) (2.3.0)
Requirement already satisfied: certifi<=2017.4.17 in /opt/homebrew/lib/python3.10/site-packages (from requests<3,=>2.27-->gTTS) (2024.12.14)
Downloading click-8.1.8-py3-none-any.whl (98 kB)
Installing collected packages: click, gTTS
Successfully installed click-8.1.8 gTTS-2.5.4

```



## Установка дополнительных библиотек

Теперь необходимо написать код программы. Программа представляет собой голосового ассистента, который воспринимает голосовые команды пользователя, обрабатывает их и отвечает голосом.

`speak(text)` используется для озвучивания текста. Преобразует переданный текст в речь с помощью библиотеки `pyttsx3` и воспроизводит его через динамики.

```

def speak(text):
    """Функция озвучивания текста"""
    engine.say(text)
    engine.runAndWait()

```

`recognize_speech()` - функция для распознавания речи. Слушает звук с микрофона и преобразует речь в текст с помощью Google Speech Recognition. Если речь не распознана, возвращает `None`. Возвращает текст команды, которую распознал ассистент.

```

def recognize_speech():
    """Функция распознавания речи"""
    recognizer = sr.Recognizer()
    with sr.Microphone() as source:
        print("Слушаю...")
        recognizer.adjust_for_ambient_noise(source)
    try:
        audio = recognizer.listen(source, timeout=5)
        command = recognizer.recognize_google(audio, language="ru-RU").lower()

        print(f"Вы сказали: {command}")
        return command
    except sr.UnknownValueError:
        print("Не удалось распознать речь")
        return None
    except sr.RequestError:
        print("Ошибка сервиса распознавания")
        return None

```

time\_to\_text() - функция, которая преобразует текущее время в текстовый формат. Получает текущее время и преобразует его в строку, используя склонения для "часов" и "минут".

```
def time_to_text():
    dict_hours = {1: 'час', 2: 'часа', 3: 'часа', 4: 'часа', 5: 'часов', 6: 'часов',
                  7: 'часов', 8: 'часов', 9: 'часов', 10: 'часов', 11: 'часов', 12: 'часов',
                  13: 'часов', 14: 'часов', 15: 'часов', 16: 'часов', 17: 'часов', 18: 'часов',
                  19: 'часов', 20: 'часов', 21: 'час', 22: 'часа', 23: 'часа', 0: 'часов'}
    dict_minutes = {
        'минута': [1, 21, 31, 41, 51],
        'минуты': [2, 3, 4, 22, 23, 24, 32, 33, 34, 42, 43, 44, 52, 53, 54],
        'минут': [5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
                  25, 26, 27, 28, 29, 30,
                  35, 36, 37, 38, 39, 40,
                  45, 46, 47, 48, 49, 50,
                  55, 56, 57, 58, 59]}
    now = datetime.datetime.now()
    h = now.hour
    m = now.minute

    str_time = str(h) + dict_hours[h] + ' '
    for minutes in dict_minutes:
        if m in dict_minutes[minutes]:
            str_time += str(m) + ' ' + minutes
            break
    return str_time
```

get\_weather() - функция для получения текущей погоды в городе (по умолчанию в Москве). Использует OpenWeather API для получения информации о текущей погоде в городе, указанном в переменной CITY. Возвращает описание погоды.

```
def get_weather():
    """Получение текущей погоды"""
    url = f"https://api.openweathermap.org/data/2.5/weather?q={CITY}&appid={API_KEY}&units=metric&lang=ru"
    try:
        response = requests.get(url)
        data = response.json()
        if data["cod"] == 200:
            temp = data["main"]["temp"]
            description = data["weather"][0]["description"]
            return f"Сейчас в {CITY} {description}, температура {temp} градусов."
        else:
            return "Не удалось получить данные о погоде."
    except:
        return "Ошибка при получении погоды."
```

get\_news() - функция для получения последних новостей из Интернета. Использует News API для получения новостей. В данный момент это новости по ключевому запросу "technology". Получает заголовки 3 последних новостей и возвращает их в виде текста.

```
def get_news():
    """Получение новостей"""
    url = f"https://newsapi.org/v2/everything?q=technology&apiKey={NEWS_API_KEY}"
    try:
        response = requests.get(url)
        data = response.json()
        if data["status"] == "ok":
            articles = data["articles"][:3] # Берем 3 главные новости
            news_list = [article["title"] for article in articles]
            return "Вот главные новости: " + " ".join(news_list)
        else:
            return "Не удалось получить новости."
    except:
        return "Ошибка при получении новостей."
```

`main()` - основная функция программы, которая слушает голосовые команды и выполняет соответствующие действия. Запускает цикл, в котором программа распознает голосовые команды. Для каждой команды выполняет соответствующие действия (озвучивание времени, открытие почты, получение погоды, новостей и другие действия).

```
def main():
    speak("Привет! Я голосовой ассистент. Скажите команду.")
    while True:
        command = recognize_speech()

        if command == "который час" or command == "сколько времени" or command == "сколько время":
            speak(time_to_text())
            print(time_to_text())
            continue

        if command == "открой почту":
            URL = "https://mail.google.com/"
            webbrowser.open(URL)

        if command == "какая погода":
            speak(get_weather())
            print(get_weather())
            continue

        if command == "что нового" or command == "какие новости":
            speak(get_news())
            print(get_news())
            continue

        if command == "найди в гугле" or command == "загугли":
            speak("открываю гугл")
            search_query = command
            webbrowser.open(f"https://www.google.com/")

        if command == "случайное число":
            number = random.randint(0, 10)
            print(f"случайное число: {number}")
            speak(f"случайное число: {number}")
            continue

        if command:
            response = commands.get(command, "Я не знаю такой команды.")
            speak(response)
            if command == "стоп":
                break

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

Результат работы программы представлен ниже, некоторые программы продублированы выводом ответа в консоль для понимания того, что ответила программа.

```
sergey@MacBook-Pro-Sergej II % python speech.py
Слушаю...
Вы сказали: как тебя зовут
Слушаю...
Вы сказали: сколько время
11 часов 58 минут
Слушаю...
Вы сказали: какая погода
Сейчас в Москва ясно, температура 5.65 градусов.
Слушаю...
Вы сказали: расскажи анекдот
Слушаю...
Вы сказали: какие новости
Вот главные новости: Under Trump, AI Scientists Are Told to Remove 'Ideological Bias' From Powerful Models. Google Is Developing Technology to Deliver Internet Via Light Bridges. LG's new air conditioner directs cool air towards people it detects.
Слушаю...
Вы сказали: случайное число
случайное число: 9
Слушаю...
Вы сказали: открой почту
Слушаю...
Вы сказали: стоп
sergey@MacBook-Pro-Sergej II %
```

## **Вывод**

В ходе выполнения практического задания была успешно программа на Python, которая бы воспринимает 9 голосовых команд: "открой почту", "сколько времени" / "который час" / "сколько время", "как тебя зовут", "какие новости" / "что нового", "найди в гугле" / "загугли", "расскажи анекдот", "случайное число", "открой почту", "стоп".

Ответы на эти команды программа выдает в виде голосовых сообщений на русском языке.