**Урок 11**

**Музыкальная викторина**

В прошлый раз мы сделали бота «Математический тренажёр», который помогает тренироваться в решении математических примеров. Сделали так, чтобы программа случайно генерировала математическую задачу с действием, которое выбрал пользователь.

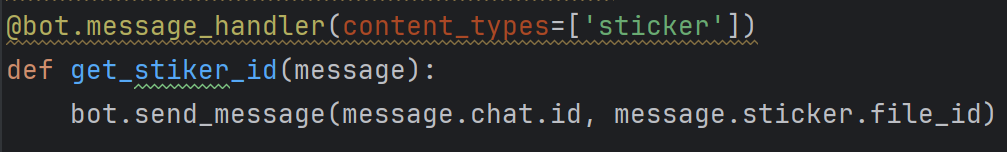
Сегодня мы создадим программу для бота, который будет проводить музыкальную викторину**.** Он будет отправлять пользователю мелодии, а затем предлагать несколько вариантов ответов. Мы будем использовать глобальный словарь викторины для каждого пользователя.

**Добавляем мелодии боту**

Для начала мы добавим в нашего бота возможность отправлять мелодии. Мы можем привычным способом загрузить файлы в папку с наши проектом и использовать их. Но есть способ получше!

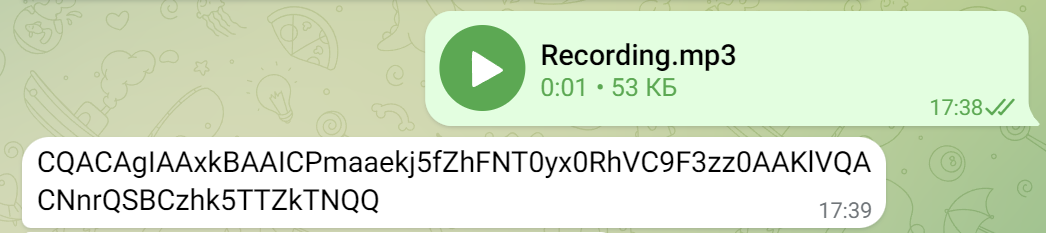
Скорее всего, эти файлы с музыкой будут отправлены неоднократно. Telegram даёт нам возможность отправлять файлы по **id**. Вероятно, вы уже замечали, как это работает. Например, вы отправляете фото кому-то, оно отправляется пару секунд. Потом решаете отправить это же фото другому человеку, а оно тут же отправляется — это происходит потому, что ваш telegram клиент запомнил, что вы уже отправляли это фото, поэтому в следующий раз высылает его по **id**. Боты могут то же самое и даже лучше.

Ещё нам необходимо знать, что каждый бот видит разные id у всех медиа файлов. По id, который получил один бот, другой бот не сможет отправить ничего, а при попытке сделать это получит ошибку. Поэтому сначала мы сделаем программу, которая позволит узнать id наших музыкальных файлов. Мы уже делали подобную вещь, когда бот отправлял пользователю id стикера.



Поменяем sticker на audio. Готово! Теперь бот реагирует на аудио и отправляет нам его id:

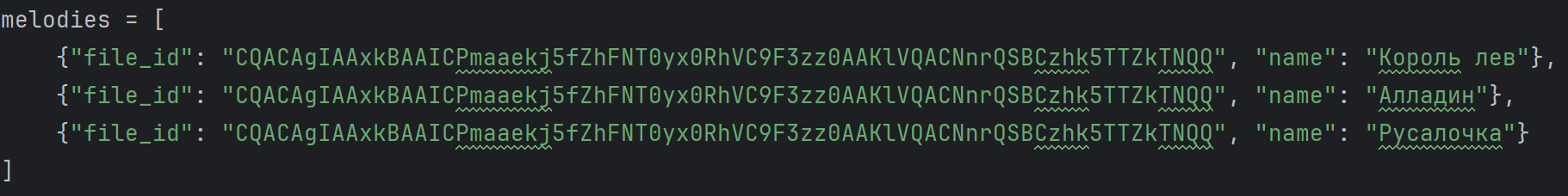




Теперь мы сможем узнать и использовать id аудио файлов.

**Самостоятельная работа**

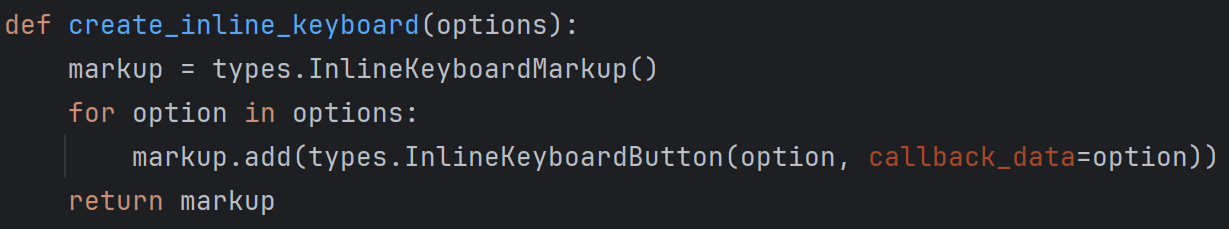
Создайте список словарей, состоящих из id аудио файлов и их названий, которые мы будем использовать в викторине.



Для примера мы добавим три мелодии, потом список можно расширить**. file\_id** – это **id** на аудио файлы, **name** – правильный ответ к этой мелодии.

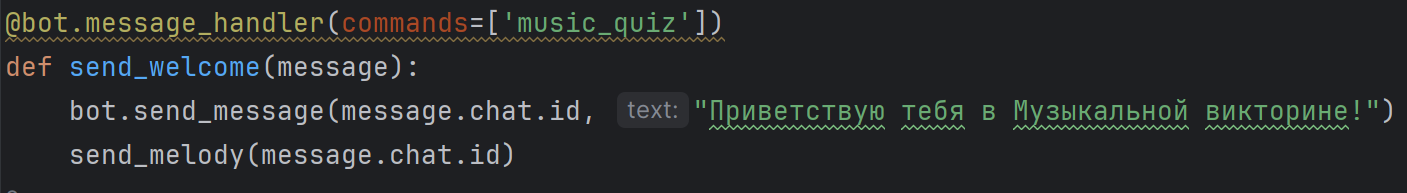
**Создание кнопок**

Теперь мы создадим функцию для создания inline-кнопок. В качестве аргумента функции мы используем параметр options, в котором будут хранится варианты ответов. В цикле добавляется кнопка для каждого варианта ответа. Кнопка создаётся с текстом **‘option**’ и **callback\_data ‘option’**, что позволяет нам легко определить, какой вариант выбрал пользователь. Далее функция возвращает готовый объект **markup**, который будет использован в сообщении.



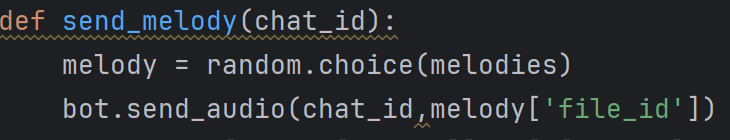
**Самостоятельная работа**

Добавьте боту новую команду и обработчик этой команды, чтобы бот отправлял приветственное сообщение пользователю.

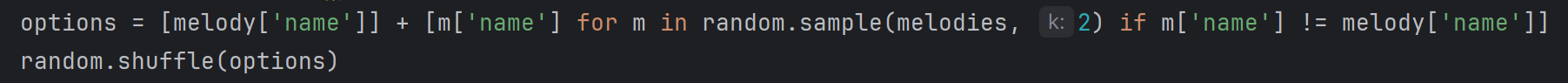


**Функция для отправки мелодии**

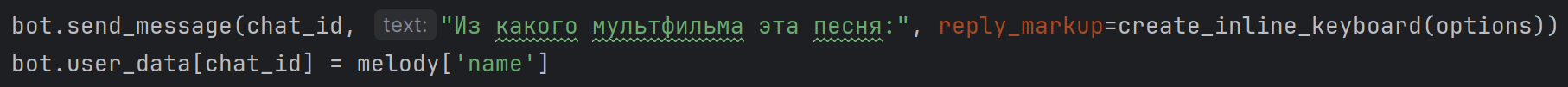
Создадим функцию, которая будет выбирать случайную мелодию и отправлять её пользователю.



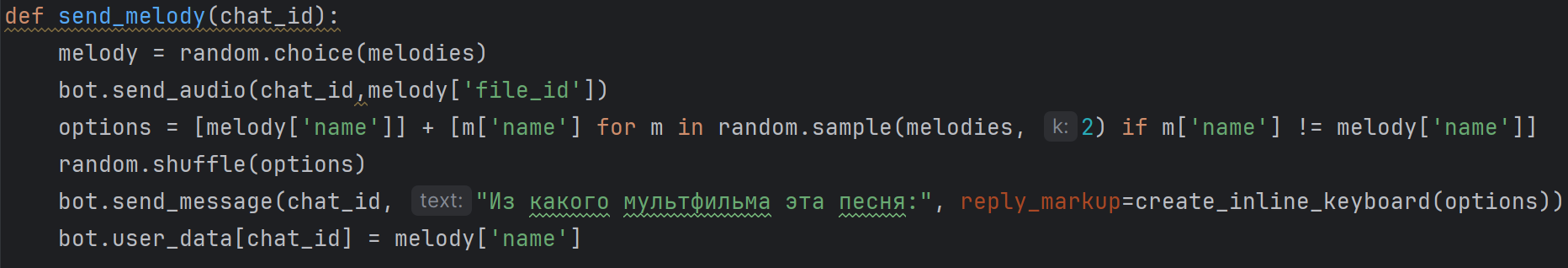
Также сделаем так, чтобы программа сгенерировала варианты ответов, включая правильный ответ. Для этого создаём список **options**, который сначала содержит правильный ответ. Затем добавляются случайные неправильные ответы. С помощью метода **random.shuffle()** мы перемешиваем все элементы списка **option**, чтобы правильный ответ не всегда был на первом месте.



Далее отправляем сообщение пользователю с inline-кнопками, созданными функцией **create\_inline\_keyboard.** Сохраняем правильный ответ в объект **bot.user\_data** — в него можно сохранять данные, связанные с пользователями. В данном случае это используется для хранения текущего правильного ответа для каждого пользователя.

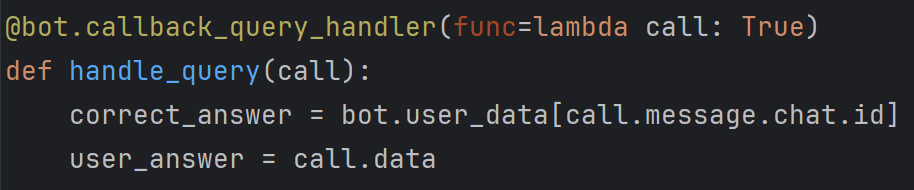


Вся функция для отправки мелодии выглядит так:



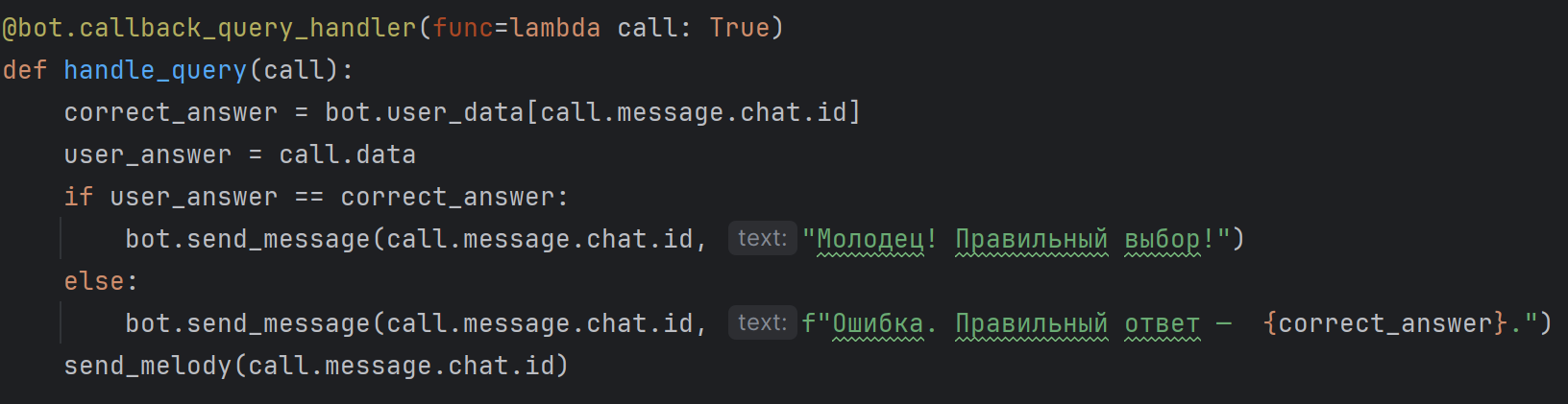
**Обработчик нажатий на кнопки**

Сначала мы сохраняем правильный ответ (который получаем из **bot.user\_data)** в переменную **correct\_answer.** Далее получаем ответ пользователя из **callback\_data** и сохраняем в **user\_answer.**

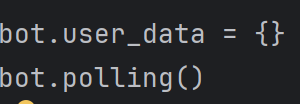


**Самостоятельная работа**

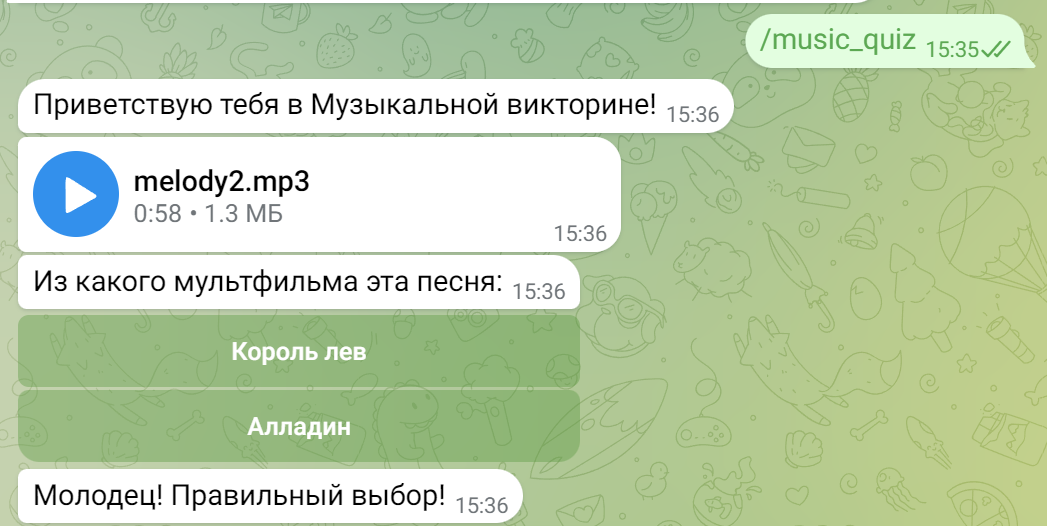
Доделайте эту программу так, чтобы бот проверял правильно ли ответил пользователь и сообщал ему об этом. Далее перезапускаем функцию, которая отправляет вопрос и варианты ответа.



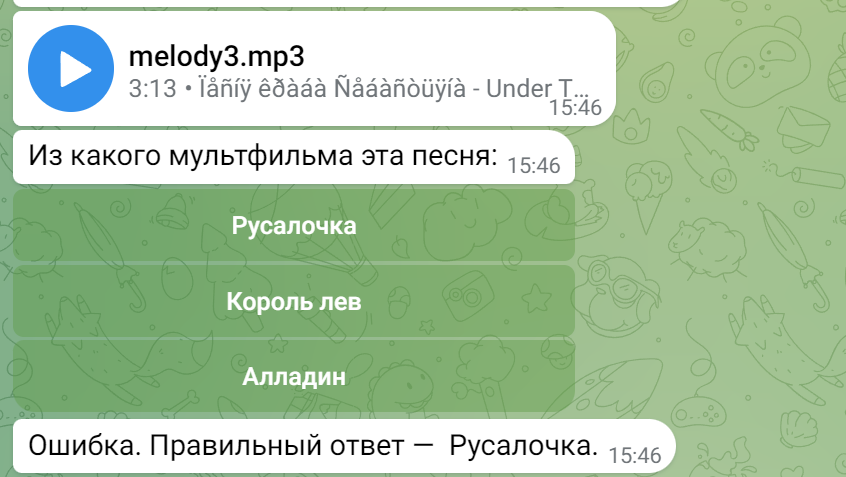
В конце кода мы как обычно указываем команду **bot.polling()** и инициализируем глобальный словарь для хранения данных о текущем состоянии викторины для пользователей.



Протестируем как работает наш бот.



Далее бот опять отправляет мелодию и предлагает выбрать ответ.



**Самостоятельная работа**

Добавьте в список дополнительные мелодии и их названия.

**Рефлексия**

* Сегодня мы создали викторину, где бот отправляет пользователю аудиофайл с мелодией, а пользователь угадывает её название
* Мы узнали, как получать id загруженных в telegram файлов
* Научились использовать глобальный словарь для хранения текущего состояния викторины для каждого пользователя

***Тайминг.***

|  |  |  |
| --- | --- | --- |
|  | Раздел урока | Время с начала урока, мин. |
|  | Повторение материала предыдущего урока. Постановка цели урока | 0 |
|  | Добавляем мелодии боту | 10 |
|  | Самостоятельная работа | 20 |
|  | Создание кнопок | 30 |
|  | Самостоятельная работа | 45 |
|  | Бескомпьютерная деятельность | 55 |
|  | Функции для отправки мелодии и обработки ответа пользователя | 60 |
|  | Самостоятельная работа | 75 |
|  | Рефлексия | 85 |