**Урок 13**

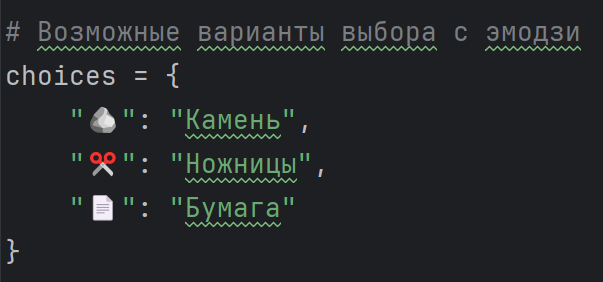
**Игра «Камень, ножницы, бумага»**

В прошлый раз мы создали бота, который генерировал случайные факты и шутки из текстовых файлов. А также осуществлял поиск статей в Википедии по слову, которое вводил пользователь.

Сегодня мы создадим бота для игры «Камень, ножницы, бумага». Этот бот будет принимать команды от пользователя, генерировать случайный выбор (камень, ножницы или бумагу) и определять победителя. Чтобы игра была более визуально привлекательной мы будет использовать эмодзи для знаков. Эмодзи — это просто символы Unicode, и они могут быть включены в строки так же, как и любые другие символы. Эмодзи можно добавить в строки Python, просто вставив их прямо в текст.

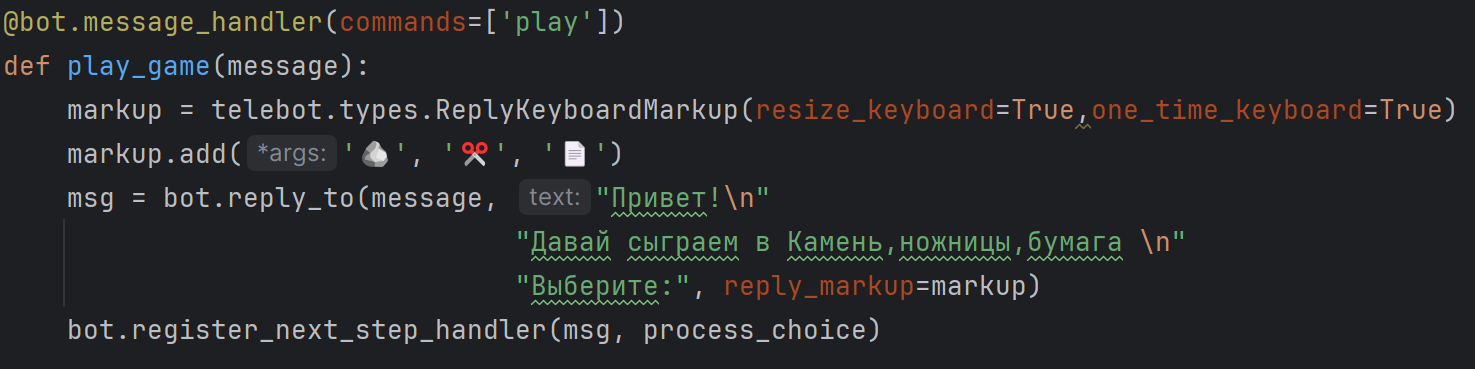
**Самостоятельная работа**

Создайте словарь, ключами которого являются эмодзи, а значениями — соответствующие текстовые описания.

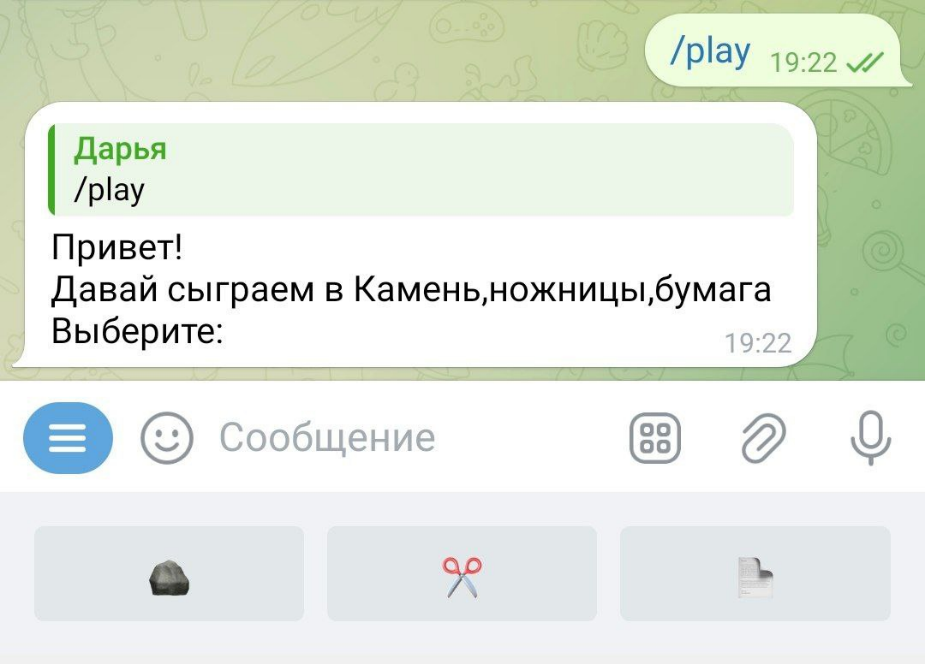


**Обработчик команды /play**

Создаём клавиатуру с кнопками, содержащими эмодзи:



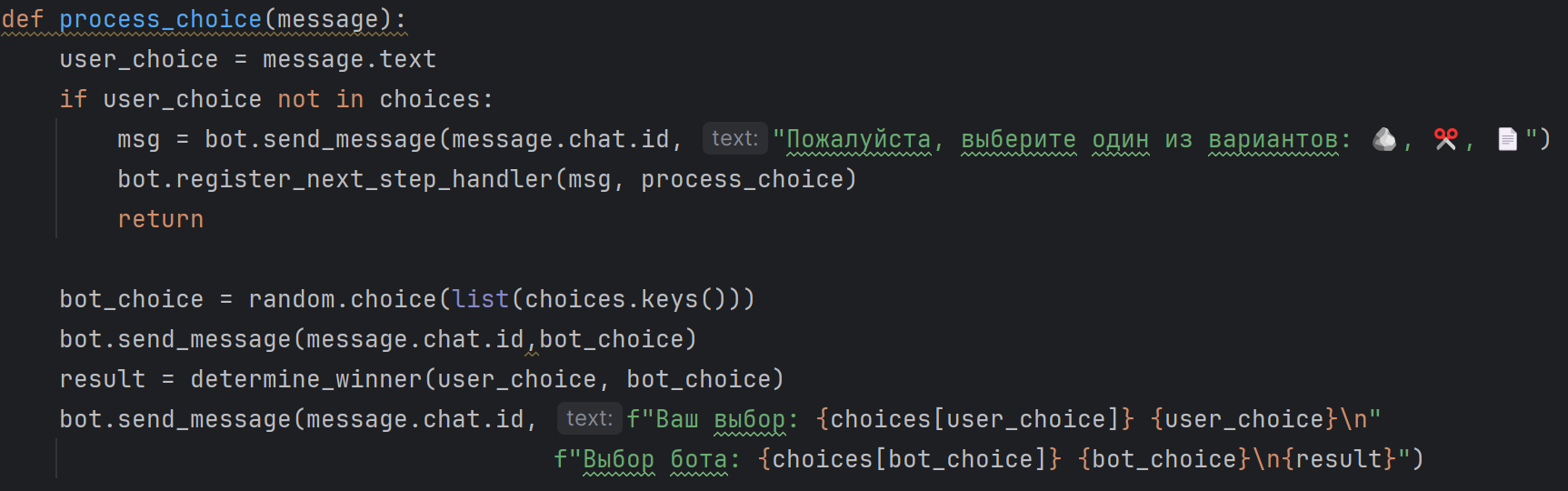
Протестируем бота:



Вы можете поменять расположение и размер кнопок. Также сделаем так, чтобы клавиатура пропадала после того, как пользователь выберет свой знак. С помощью какого параметра можно это сделать? Для этого необходимо указать **one\_time\_keyboard=True.**

**Функция для обработки выбора пользователя**

Теперь сделаем функцию, которая будет запускаться после того, как пользователь выберет знак и нажмёт на какую-либо кнопку. Сохраняем выбор пользователя в переменной **user\_choice**. Проверяем выбрал ли пользователь один из символов. Если было отправлено другое сообщение, бот попросит выбрать один из вариантов знака. В переменную **bot\_choice** мы сохраняем рандомный символ. Бот будет отправлять эмодзи, поэтому необходимо использовать ключ из нашего словаря.



**Самостоятельная работа**

Сколько возможных ситуаций существует при которых игрок побеждает, проигрывает или происходит ничья? Выпишите на лист бумаги все вероятные исходы игры. Для визуализации можно сделать похожую таблицу:

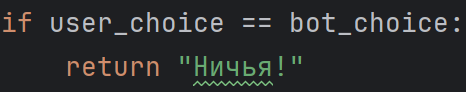


**Функция, определяющая победителя**

Сделаем функцию, которая будет проверять кто выиграл — игрок или бот, — сравнивая их выборы по правилам игры. Функция принимает два параметра:

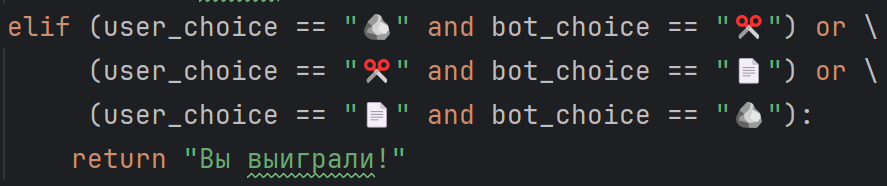
* user\_choice — выбор игрока (камень, ножницы или бумага)
* bot\_choice — выбор бота (камень, ножницы или бумага)

Первым делом функция проверит выбрали ли игрок и бот одно и то же — произошла ли ничья в игре.

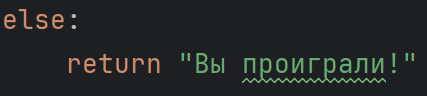


Затем функция проверяет выиграл ли игрок — проверяет три ситуации, при которых это происходит:

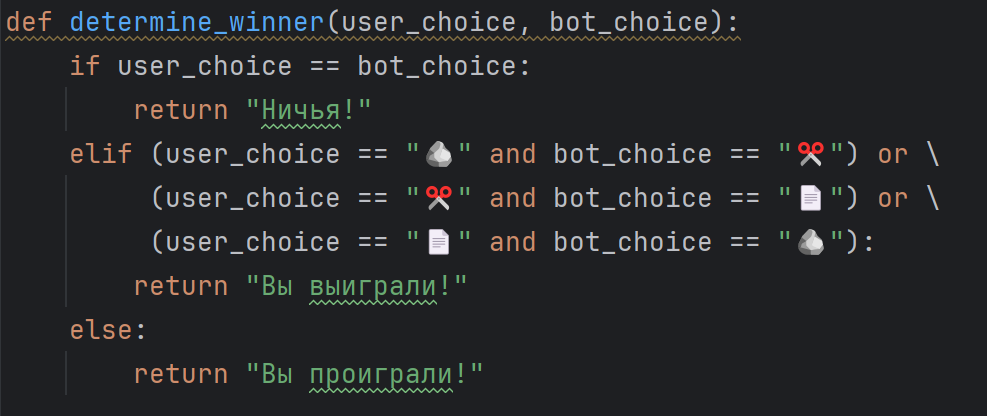
* Камень побеждает ножницы.
* Ножницы побеждают бумагу.
* Бумага побеждает камень.



Если игрок не выиграл и не ничья, значит победил бот. Если предыдущие условия не выполнены, значит бот победил.

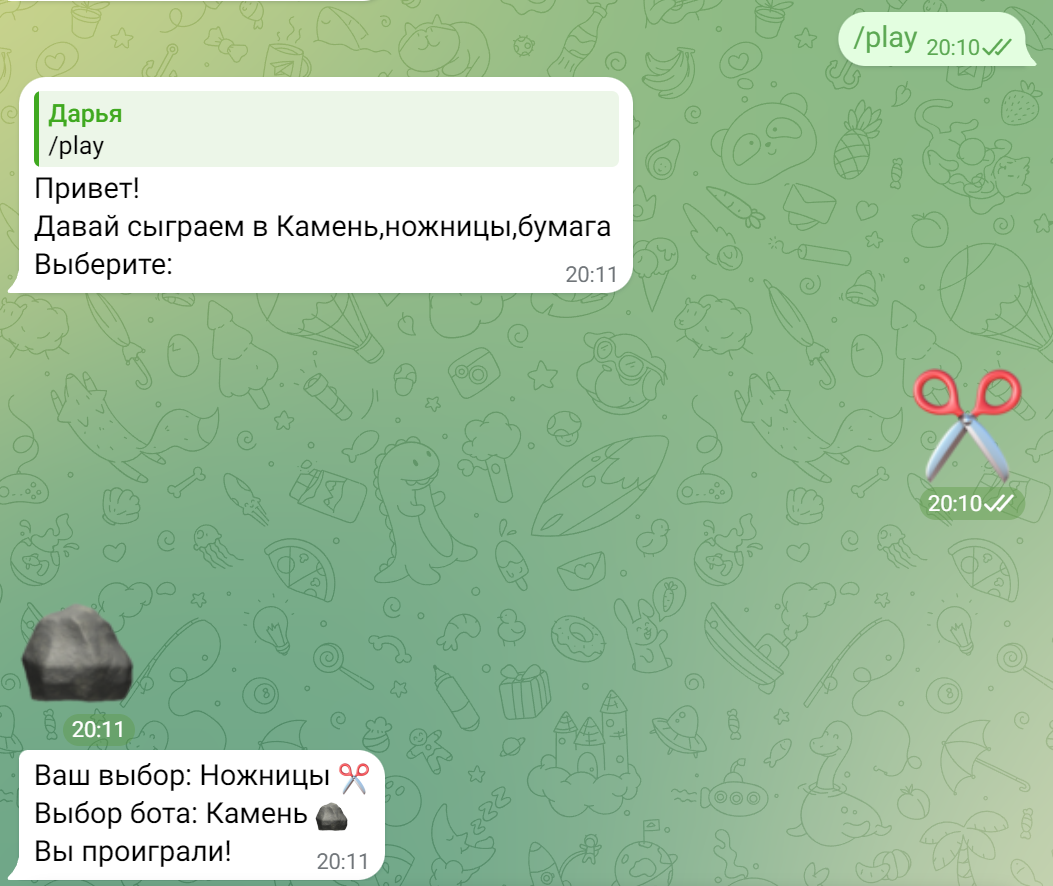


Вся функция выглядит следующим образом:



**Самостоятельная работа**

Проверьте все исходы игры с ботом (победа, поражение и ничья)



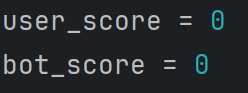




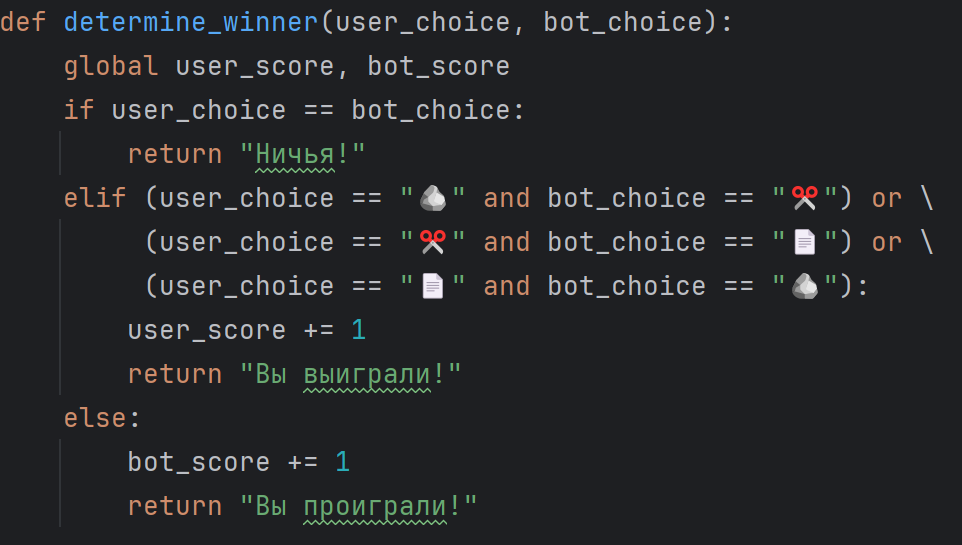
**Подсчёт очков**

Добавим возможность играть несколько раундов в «Камень, ножницы, бумага». Программа будет подсчитывать очки и объявлять победителя после нескольких раундов.

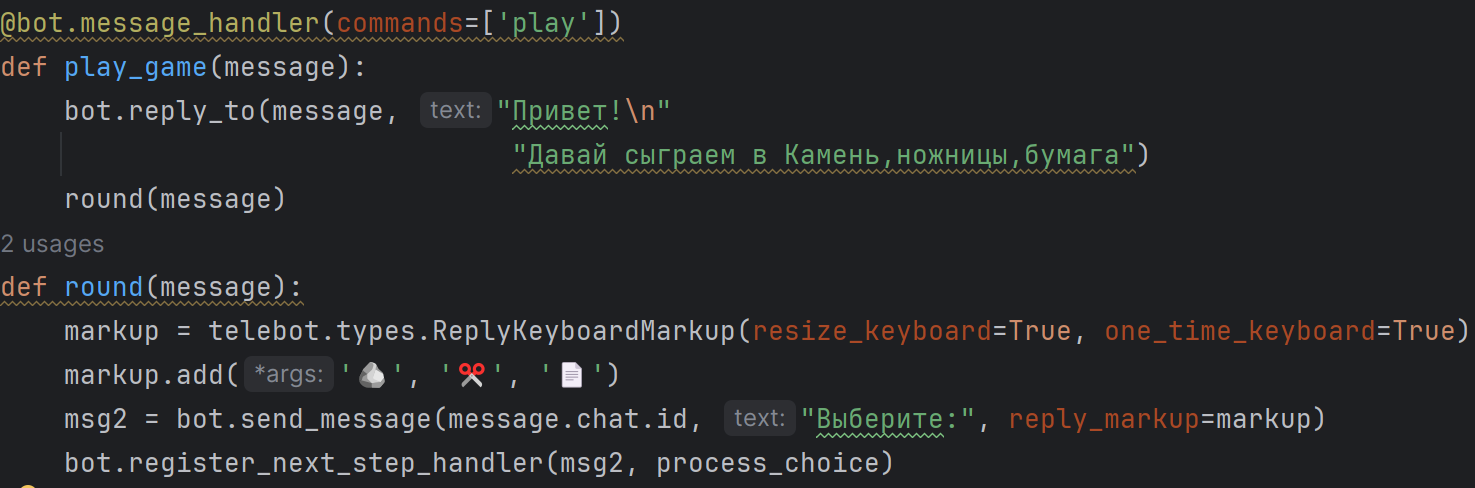
Для начала добавим переменные **user\_score** и **bot\_score** для отслеживания количества побед игрока и бота.



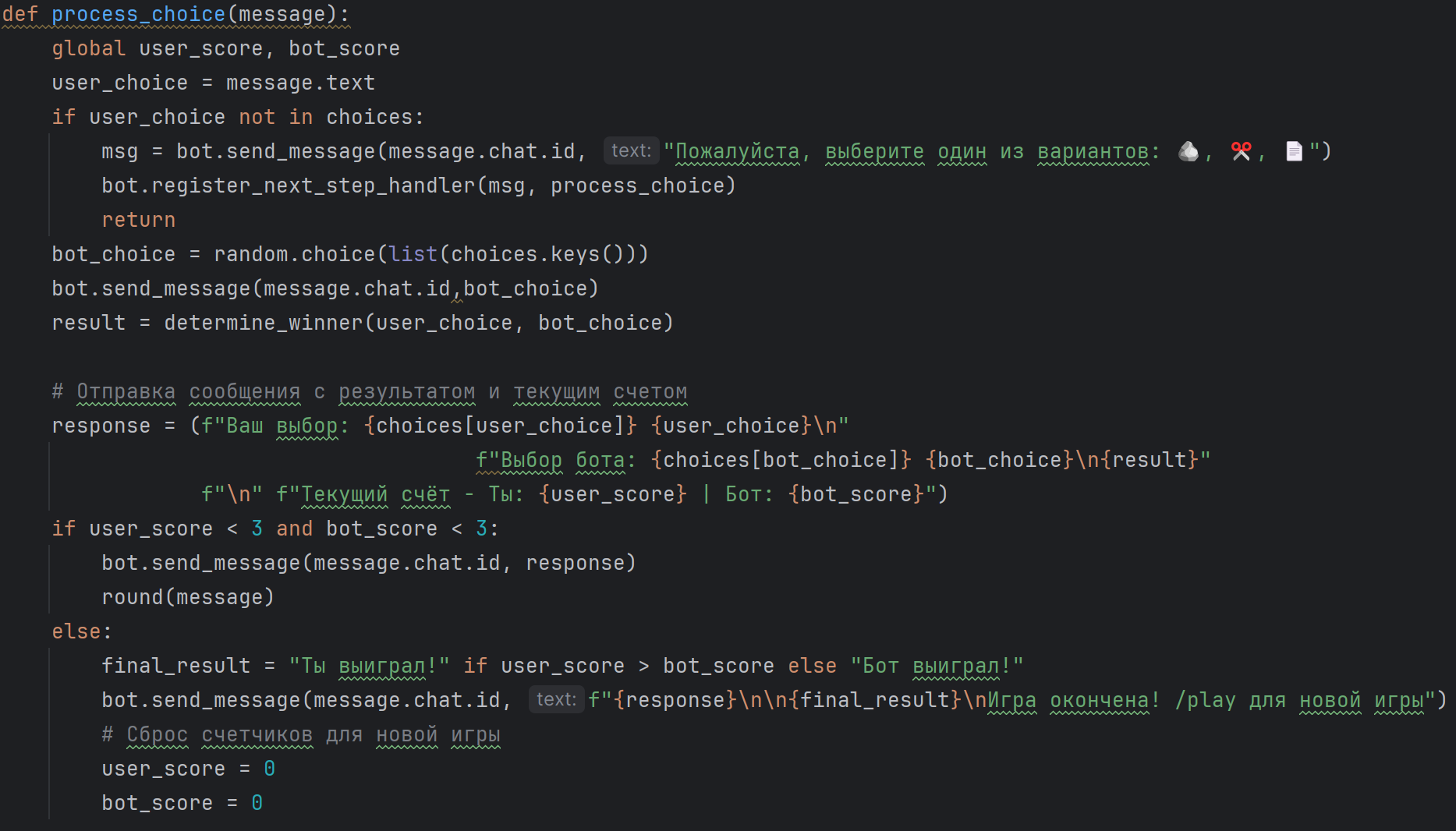
Редактируем функцию, определяющую победителя. Делаем переменные **user\_score** и **bot\_score** глобальными для того, чтобы их значение сохранялось между вызовами функции. Также делаем так, чтобы переменные увеличивались в зависимости от того кто победил.



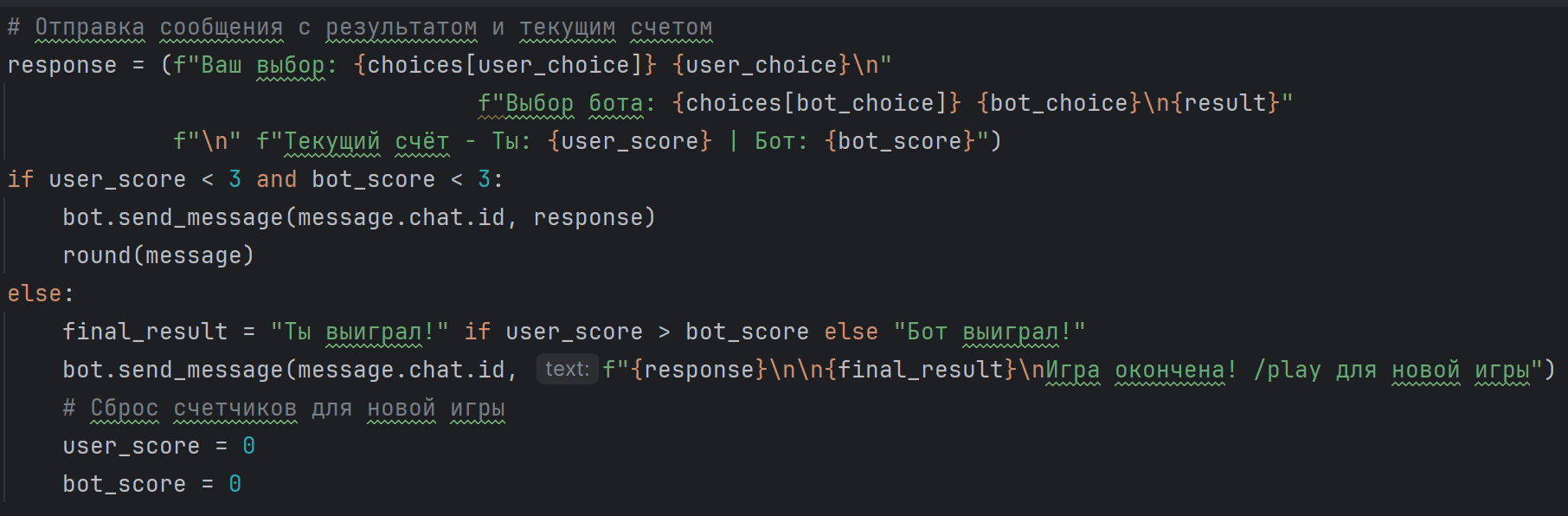
Редактируем обработчик команды **/play.** В дальнейшем мы будем перезапускать несколько раз создание кнопок, поэтому сделаем отдельную функцию, которая будет совершать эти действия:



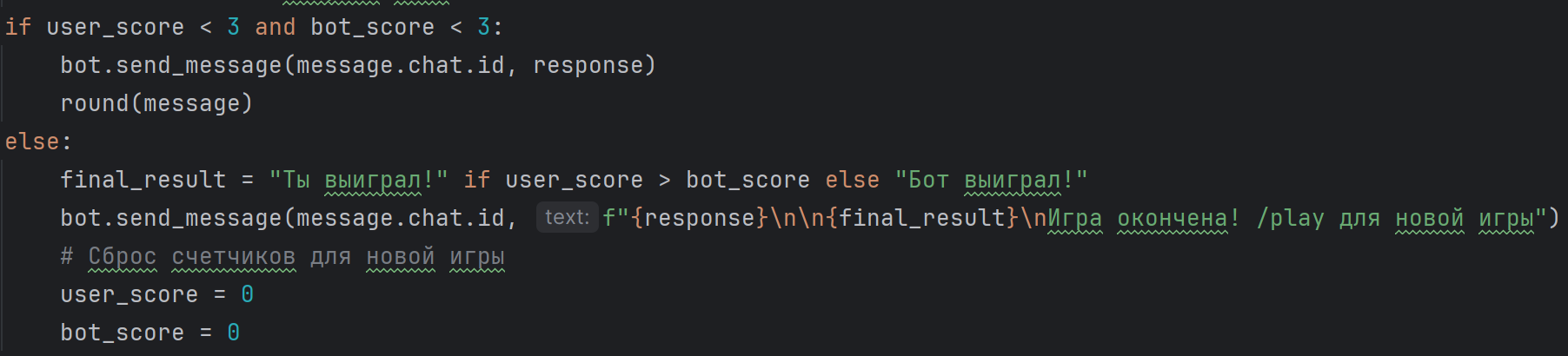
Теперь перейдём к функции, которая обрабатывала ответ пользователя. Делаем переменные **user\_score** и **bot\_score** глобальными. Остальную программу функции не меняем.



Дополняем её — делаем так, чтобы каждый раунд бот говорил пользователю о том, кто победил в этом раунды и о том какой сейчас счёт в игре:



Далее проверяем условие — если один из игроков набирает 3 очка, игра заканчивается, и бот отправляет сообщение с итогом и предлагает начать новую игру командой **\play:**

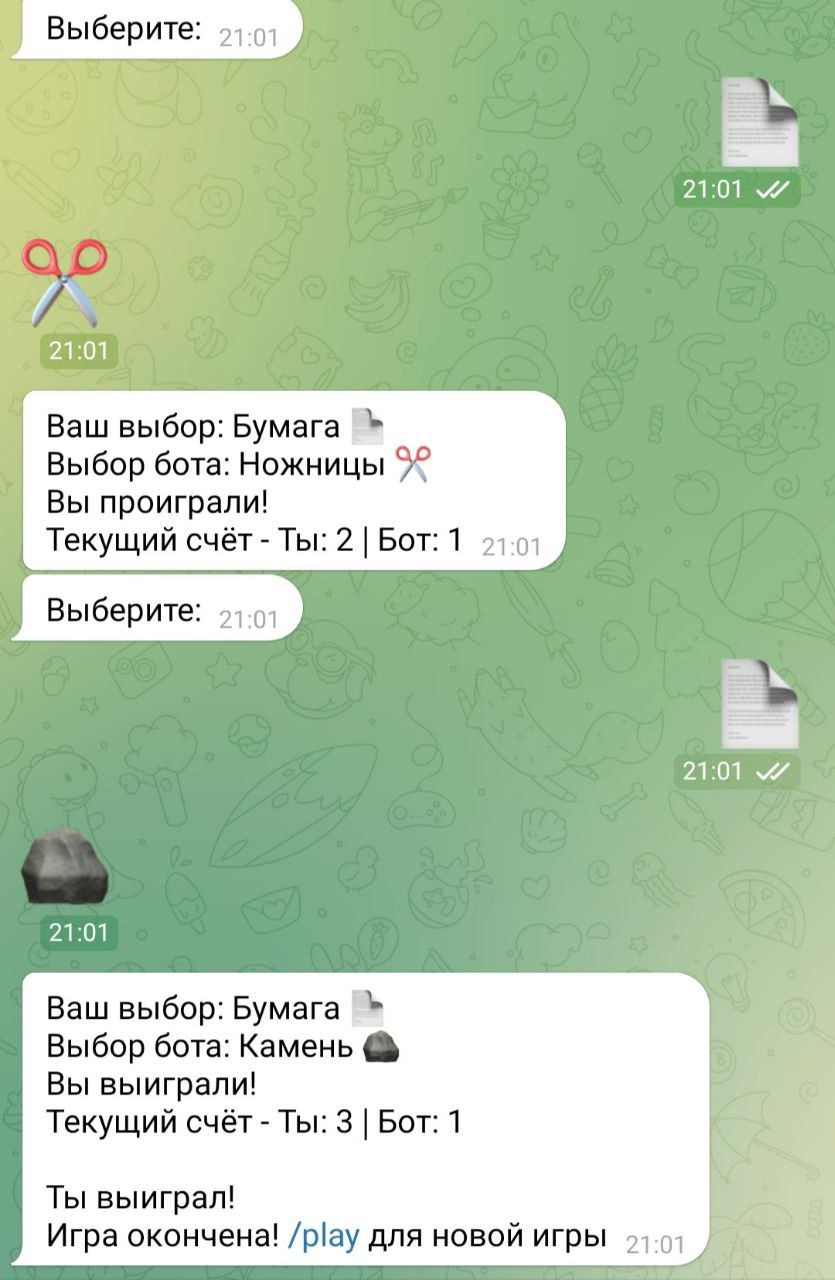


Вся функция **process\_choice** выглядит следующим образом:



**Самостоятельная работа**

Протестируйте работу бота.



**Рефлексия**

* Сегодня мы создали игру «Камень, ножницы, бумага» в telegram боте
* Добавили кнопки с эмодзи знака — камень, ножницы или бумага
* Сделали так, чтобы программа генерировала выбор бота случайным образом
* Осуществили сравнение выбора игрока и бота, вывод результата и подсчитывание очков
* Также мы добавили возможность играть несколько раундов

***Тайминг.***

|  |  |  |
| --- | --- | --- |
|  | Раздел урока | Время с начала урока, мин. |
|  | Повторение материала предыдущего урока. Постановка цели урока | 0 |
|  | Самостоятельная работа | 10 |
|  | Обработчик команды /play | 20 |
|  | Функции для обработки выбора пользователя и определения победителя | 30 |
|  | Самостоятельная работа | 50 |
|  | Бескомпьютерная деятельность | 60 |
|  | Добавляем подсчёт очков | 65 |
|  | Самостоятельная работа | 75 |
|  | Рефлексия | 85 |