Для сдачи лабораторной необходимо решить, как минимум по две задачи из каждого раздела. Максимально можно набрать 30 баллов.

Асинхронность

№1 (**1 балл**)

Имеется синхронная версия программы для подсчета значений факториала чисел. Используя модуль asyncio сделайте программу асинхронной. В этом случае в каком порядке завершаются задачи?

```
def factorial(name, number):
    f = 1
    for i in range(2, number + 1):
        print(f"Task {name}: Compute factorial({i})...")
        f *= i
    print(f"Task {name}: factorial({number}) = {f}")


def main():
    factorial("A", 15),
    factorial("B", 7),
    factorial("C", 4),


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

№2 (**2 балла**)

Напишите программу, которая выводит ваш IP адрес. Есть много сервисов, которые позволяют узнать ваш ip (https://api.ipify.org, http://ip-api.com и т.д.). Но на момент запуска программы вы не знаете какой из сервисов доступен. Вместо того, чтобы опрашивать каждый из этих сервисов последовательно, запустите все запросы асинхронно и выберите первый успешный.

На экране должен быть выведен ваш ip и название сервиса, который ответил первым.

№3 (**3 балла**)

Когда что-то делаешь впервые, хорошо бы записать порядок действий, чтобы что-нибудь не забыть и не перепутать. Вот, например, памятка для начинающего агронома.

Семена перед посадкой нужно замочить в специальном растворе для повышения всхожести и защиты от болезней. Выдержать в нем строго указанное время.

Затем посадить в землю и накрыть нетканым материалом для сохранения влажности. Через заданное время снять укрытие.

Пикировка. Подросшие сеянцы нужно аккуратно рассадить по одному в горшочек. Подождать некоторое время, пока они приживутся. Все, рассада готова.

Эти этапы должны следовать именно в таком порядке. Однако подкормить растения минеральными удобрениями и обработать от вредителей можно в любой момент. И это надо сделать обязательно. Время на подкормку всегда 3, на обработку от вредителей – 5.

При начале подкормки выводится сообщение:

7 Application of fertilizers for <растение>
при окончании:
7 Fertilizers for the <растение> have been introduced
При начале обработки:
8 Treatment of <растение> from pests
при окончании:
8 The <растение> is treated from pests

Напишите асинхронную функцию sowing(), принимающую произвольное количество кортежей: *(растение, время замачивания, время прорастания, время приживания после пикировки)* и печатающую отчет в виде:

При начале процесса для каждого растения
0 Beginning of sowing the <растение> plant

При начале замачивания
1 Soaking of the <растение> started

После этапа замачивания
2 Soaking of the <растение> is finished

При высадке семян и установке укрытия
3 Shelter of the <растение> is supplied

При снятии укрытия
4 Shelter of the <растение> is removed

При начале пикировки
5 The <растение> has been transplanted

Растение прижилось
6 The <растение> has taken root

После окончания выращивания рассады
9 The seedlings of the <растение> are ready

Для увеличения скорости работы программы разделите все времена ожидания на 1000.

Пример

Ввод	Вывод

`data = [('carrot', 7, 18, 2), ('cabbage', 2, 6, 10), ('onion', 5, 12, 7)]`
`asyncio.run(sowing(*data))`
	`0 Beginning of sowing the carrot plant`
	`1 Soaking of the carrot started`
	`7 Application of fertilizers for carrot`
	`8 Treatment of carrot from pests`
	`0 Beginning of sowing the cabbage plant`
	`1 Soaking of the cabbage started`
	`7 Application of fertilizers for cabbage`
	`8 Treatment of cabbage from pests`
	`0 Beginning of sowing the onion plant`
	`1 Soaking of the onion started`
	`7 Application of fertilizers for onion`
	`8 Treatment of onion from pests`
	`2 Soaking of the cabbage is finished`
	`3 Shelter of the cabbage is supplied`
	`7 Fertilizers for the carrot have been introduced`
	`7 Fertilizers for the cabbage have been introduced`
	`7 Fertilizers for the onion have been introduced`
	`8 The carrot is treated from pests`
	`8 The cabbage is treated from pests`
	`2 Soaking of the onion is finished`
	`3 Shelter of the onion is supplied`
	`8 The onion is treated from pests`
	`2 Soaking of the carrot is finished`
	`3 Shelter of the carrot is supplied`
	`4 Shelter of the cabbage is removed`
	`5 The cabbage has been transplanted`
	`4 Shelter of the onion is removed`
	`5 The onion has been transplanted`
	`6 The cabbage has taken root`
	`9 The seedlings of the cabbage are ready`
	`4 Shelter of the carrot is removed`
	`5 The carrot has been transplanted`
	`6 The onion has taken root`
	`9 The seedlings of the onion are ready`
	`6 The carrot has taken root`
	`9 The seedlings of the carrot are ready`

Telegram

Заведите собственного телеграм-бота, ориентируясь на материалы лекции или на официальную документацию https://core.telegram.org/bots#6-botfather.

№4 (**1 балл**)

Напишите эхо-бота так, чтобы он отвечал сообщением: «Я получил сообщение <исходное сообщение>».

№5 (**1 балл**)

Напишите бота, который будет обрабатывать следующие команды:

- команда /time, которая отвечает текущим временем в текстовом формате.
- команда /date, которая отвечает текущей датой в текстовом формате.

№6 (**2 балла**)

Напишите бота-помощника для настольных игр. В настольных играх требуется генератор случайных чисел. Как правило, используются игральные кости — кубики или другие многогранники с нанесёнными на грани цифрами.

Часть функций бота должна быть посвящена «бросанию» разных кубиков.

Также часто бывает нужно засекать время. Поэтому вторая часть функций — засечь определенное время.

По команде /start бот должен предлагать клавиатуру из двух кнопок: /dice и /timer, каждая из которых ведет на следующую клавиатуру.

По команде /dice бот должен предлагать клавиатуру с кнопками:

- кинуть один шестигранный кубик,
- кинуть 2 шестигранных кубика одновременно,
- кинуть 20-гранный кубик,
- вернуться назад.

Бот должен ответить числом или двумя числами из соответствующего диапазона. Клавиатура с кубиками должна оставаться активной.

По команде /timer бот должен предлагать клавиатуру с кнопками:

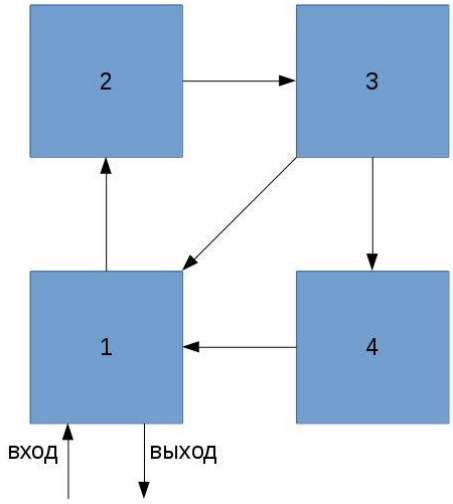
- 30 секунд,
- 1 минута,
- 5 минут,
- вернуться назад.

Нажатие одной из кнопок должно отвечать текстом «засек {указанное время}», а через это время должно приходить сообщение с текстом «{указанное время} истекло». Клавиатура с таймерами должна заменяться на клавиатуру с кнопкой /close, нажатие на которую сбрасывает таймер.

Нажатие на кнопку «вернуться назад» на обеих клавиатурах возвращает стартовую клавиатуру.

№7 (**2 балла**)

Составьте диаграмму состояний для бота-экскурсовода по музею. Музей состоит из четырех залов. Схема музея — ниже на картинке (стрелками обозначено направление возможного движения между залами и вход/выход в музей):



Напишите бота, описанного составленной диаграммой состояний. В каждом из помещений выдается сопроводительный текст: «Добро пожаловать! Пожалуйста, сдайте верхнюю одежду в гардероб!» на входе и «Всего доброго, не забудьте забрать верхнюю одежду в гардеробе!» на выходе, «В данном зале представлено...» в каждом из залов. После сопроводительного текста выводится список помещений, в которые можно перейти далее с небольшим описанием этих помещений.

№8 (**2 балла**)

Напишите бота для проверки каких-либо знаний. Например, дат по истории. На входе он получает **json**-файл, в котором в формате:

```
{
    "test": [
        {
            "question": "В каком году была война 1812 года?",
            "response": "1812"
        }
    ]
}
```

заданы контрольные вопросы и ответы. Бот должен прочитать и «распарсить» этот файл, по команде **/start** предложить пройти опрос, и затем задать пользователю десять вопросов, сравнивая полученные ответы и считая количество правильных.

По окончании вопросов бот должен вывести количество правильных ответов и предложить пройти тест снова.

Команда **/stop** прерывает работу в любой момент.

Важно: вопросы должны задаваться в случайном порядке.

№9 (**2 балла**)

Соберите работающий бот-геокодер, который по запросу пользователя присылает ему карту с запрошенным объектом.

Добавьте аннотацию к посылаемой картинке. Сделайте это одним сообщением с картинкой. (Найдите в документации API, как это сделать).

Добавьте обработку ошибок HTTP с отправкой диагностики пользователю.

Добавьте обработку ситуации «ничего не найдено» с внятным ответом пользователю.

Добавьте метку в центральную точку найденного объекта.

№10 (**2 балла**)

Соберите работающего бота-переводчика, использующего API переводчика.

Дополните бот клавиатурой, позволяющей задавать направление перевода (с русского на английский и наоборот).

Помните про то, что с ботом могут одновременно общаться несколько человек, не храните направление перевода в глобальных переменных!

Для решения вам нужно найти бесплатный API для перевода и научиться его использовать. Или найти подходящую библиотеку. Например: здесь.

№11 (**3 балла**)

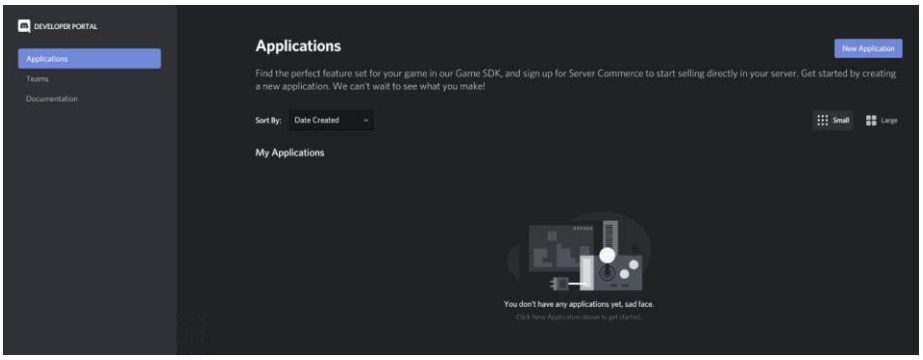
Напишите бота, у которого есть команда «price». От пользователя в виде аргумента команды «price» принимается цена товара. Бот должен выбрать такой товар с сайта https://scrapingclub.com/exercise/list_basic/, который ближе всего по своей стоимости к введенной пользователем цене (модуль разности между стоимостью и введенной ценой минимальный). Обратите внимание, что это многостраничный сайт. Если таких товаров несколько, то выбирается тот, который по алфавиту находится раньше всех. Далее бот должен выслать пользователю следующие данные о найденном товаре: название, описание, фотография, цена.

Discord

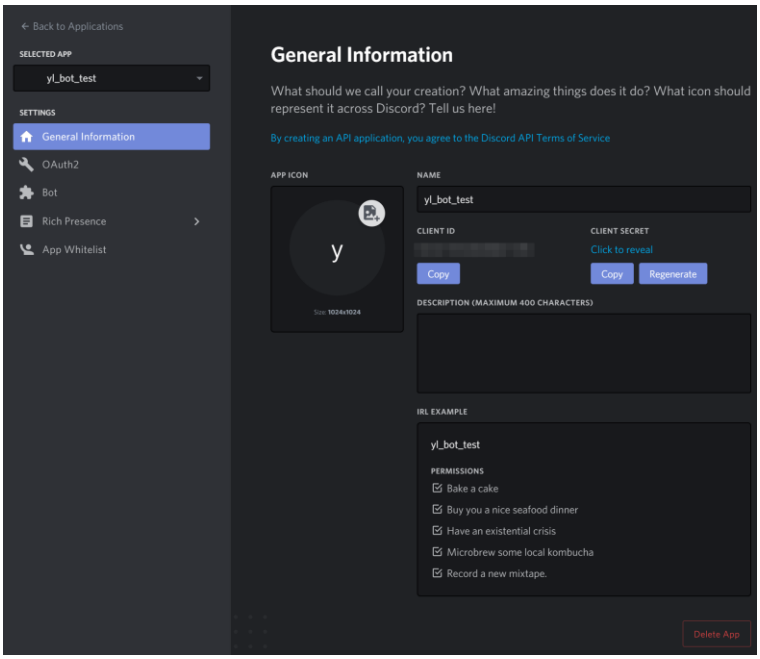
Документацию на библиотеку discord.py можно найти тут, а описание самого API Discord — вот тут.

Чтобы получить доступ к API Discord, необходимо проделать несколько шагов и получить несколько токенов доступа. Во-первых, необходимо зарегистрироваться на самой платформе и подтвердить почту, которая была указана при регистрации.

После этого надо зайти на портал для разработчиков, авторизоваться там и нажать кнопку New application. Discord устроен так, что для создания бота необходимо сначала создать приложение для получения доступа к API, а потом к этому приложению можно будет добавить бота.

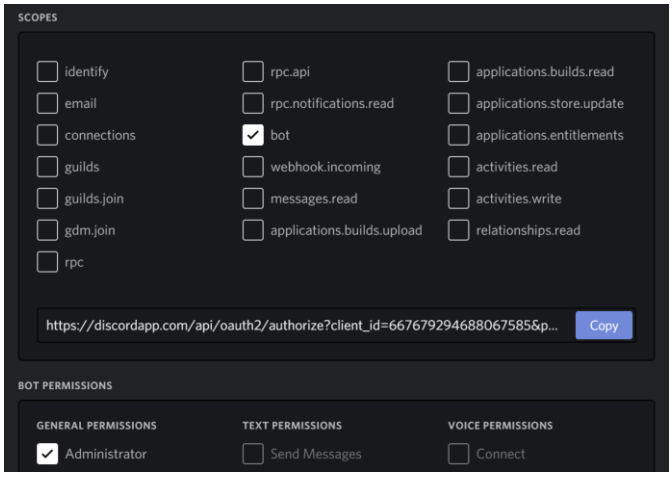


После ввода имени приложения мы перейдем в настройки приложения.



Затем для добавления к своему приложению бота перейдите на вкладку Bot и нажмите кнопку Add Bot. Можно сделать публичного бота, которого смогут добавлять все желающие, либо приватного, которого сможете добавлять только вы.

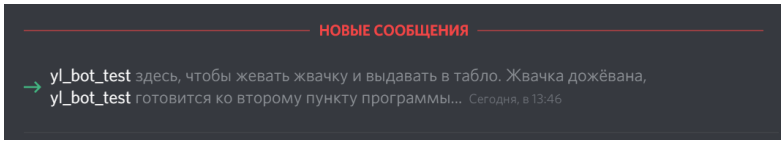
Остался последний штрих — добавить боту прав. Для этого воспользуемся вкладкой OAuth2. Выберем URL Generator — Scopes — Bot, а в Bot Permissions — Administrator.



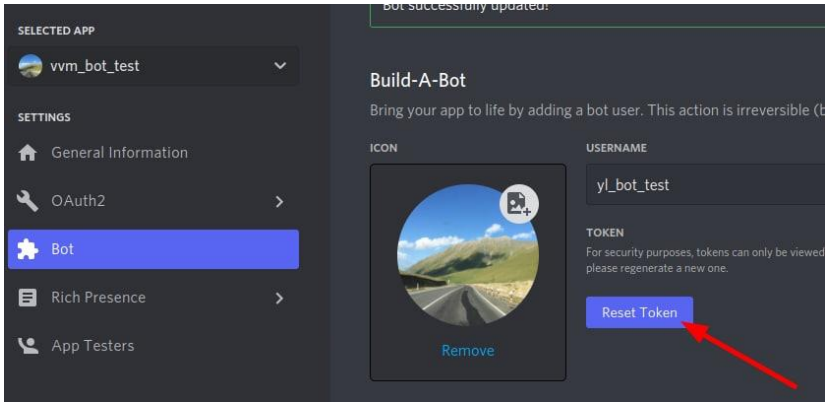
Хорошим тоном считается не запрашивать больше прав, чем необходимо для корректного функционирования бота, потому что это может отпугнуть пользователей, а также дать злоумышленникам много лишних прав для причинения вреда вашим пользователям, если они завладеют вашим ботом.

После этого перейдите по ссылке из поля GENERATED URL и добавьте бота на сервер. Это может быть личный сервер или любой, который вы создали в приложении Discord.

Бот добавился и написал автоматическое приветственное сообщение:



Чтобы ваши программы работали вам нужно получить для бота токен.

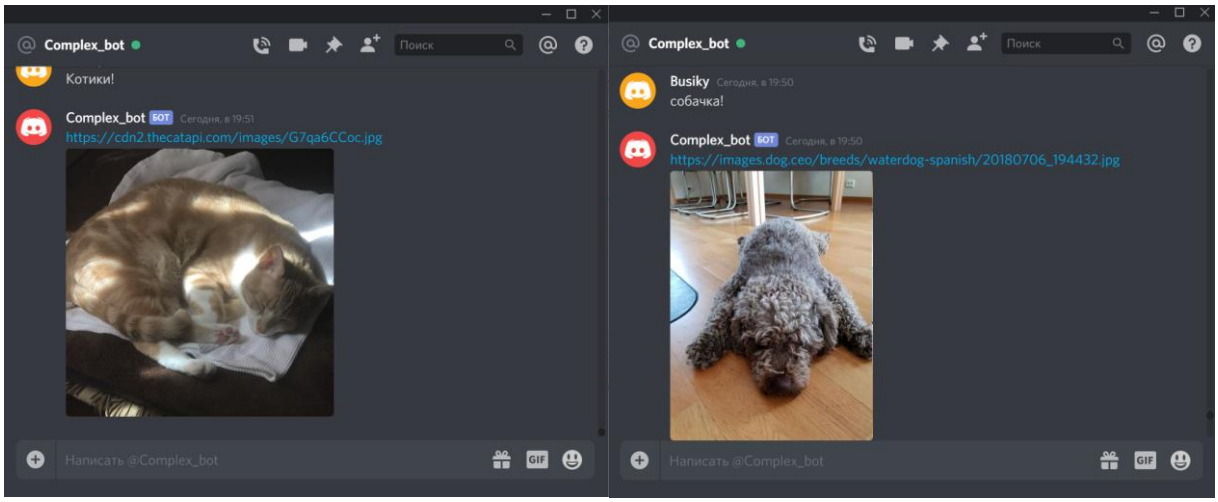


Чтобы работали примеры из лекции на странице настройки приложения надо перейти на закладку Bot, там добавить для бота разрешения на получение событий об изменении статусов пользователей сервера и разрешения на получение текстов сообщений от пользователей сервера. Для этого надо включить PRESENCE INTENT, SERVER MEMBERS INTENT и MESSAGE CONTENT INTENT.

№12 (**1 балл**)

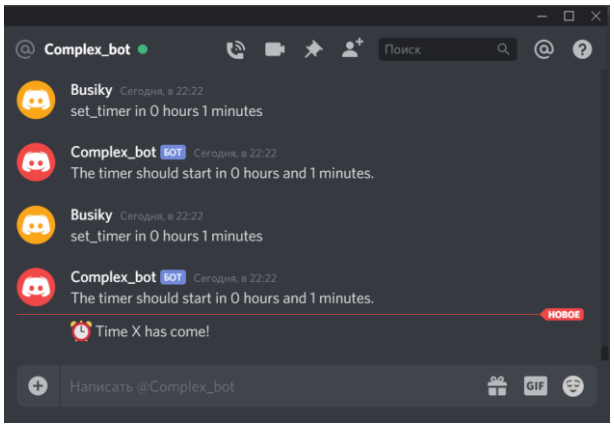
Напишите простого бота, который имеет всего два обработчика событий: on_ready и on_message. При подключении бота он выводит в консоль сообщение, что подключился и готов показать котика (или пёсика!).

Если в сообщении пользователя есть кот в любом виде, то нужно, воспользовавшись API показа случайных картинок с котиками (https://api.thecatapi.com/v1/images/search) и модулем requests, послать полученный файл с картинкой с помощью метода message.channel.send(file). А если пользователь больше любит собак (в любом виде), то для этого тоже есть API: https://dog.ceo/api/breeds/image/random.



№13 (**1 балл**)

Напишите бота, который через указанное время напишет сообщение «время X наступило!».

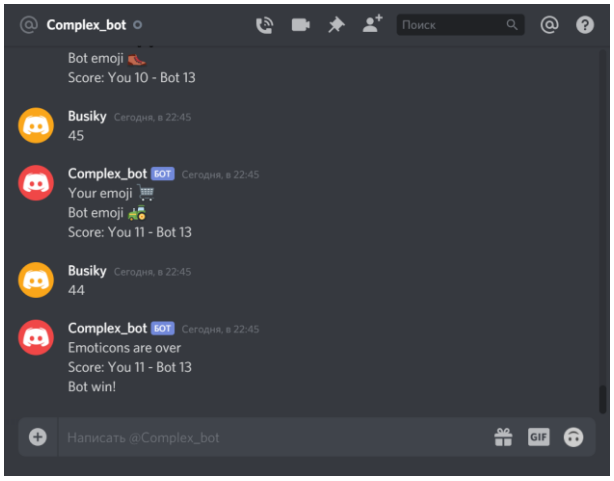


№14 (**2 балла**)

У нашего бота есть в запасе много смайликов (точнее, их unicode-символов). По очереди с ботом будем называть любое число — номер смайла, которого нужно вытащить из запаса. Если номер больше, чем количество оставшихся в запасе смайликов, то берется номер, равный остатку от деления названного числа на количество оставшихся. В каждой паре сравниваются числовые значения unicode-символов и определяется победитель данного раунда.

После каждого раунда смайлы перемешиваются, выводятся смайлы, вытащенные пользователем и ботом, а также текущий счёт. Игра продолжается до последнего смайлика или до сообщения

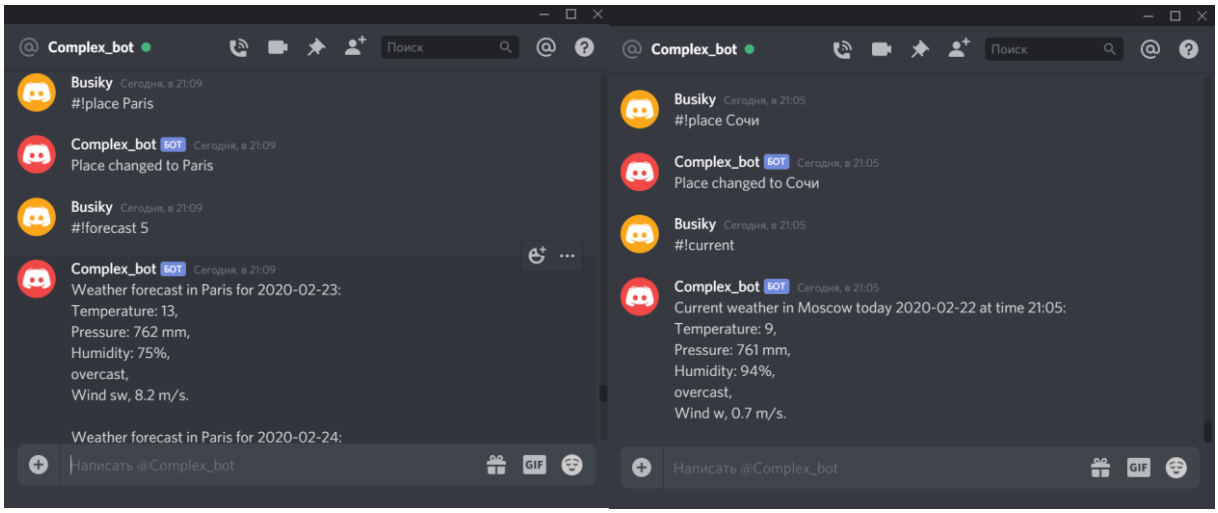
пользователя /stop. По окончании выводится победитель. При прерывании игры командой /stop счёт обнуляется.



№15 (**2 балла**)

Напишите бота, который будет сообщать текущую погоду в указанном месте, а также прогноз на указанное количество дней (не более 7). Воспользуйтесь API Яндекс.Погоды или другим сервисом погоды.

По команде help_bot бот сообщает о своих функциях. Команда place задает место прогноза. Команда current присылает сообщение о текущей погоде, в котором должны присутствовать поля: температура, давление, влажность, направление и сила ветра. Место, дата и местное время — по желанию. Команда forecast {days} сообщает прогноз дневной температуры и осадков на указанное количество дней.



№16 (**3 балла**)

Напишите бота, у которого есть команда «cite». От пользователя в виде аргументов команды «cite» принимаются теги (их может быть несколько одновременно). Бот должен вывести все цитаты с этими тегами с сайта https://quotes.toscrape.com/. Обратите внимание, что это многостраничный сайт.