Методические указания для тренера  
по курсу «Основы алгоритмизации и программирования для разработки ботов для телеграмм»

## Требования к программному обеспечению

Для проведения тренинга на компьютерах слушателей и тренера должно быть установлено следующее программное обеспечение:

* Python 3.8 (<https://python.org>)
* Visual Studio Code (<https://code.visualstudio.com>) и расширения:
  + Python
  + Python Ident
  + Python Preview
  + Git
  + Gitflow
* Библиотеки Python (устанавливаются командой pip install БИБЛИОТЕКА):
  + psycop2
  + pymongo
  + requests
  + aiogram
* Программное обеспечение системы управления базами данных:
  + Postgre SQL (<https://www.postgresql.org/download/>)
  + Mongo DB (<https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows>)
  + DBeaver (<https://dbeaver.io/>) – GUI для работы с Postgre SQL
  + Robot 3T (<https://robomongo.org/>) – GUI для работы с Mongo DB

## Общие указания к упражнениям

* В модулях с 6 по 9 предполагается, что слушатели будут постепенно создавать бота, на основе заготовленного кода. Слушатели должны работать с файлами, которые расположены в пакете business\_service
* В модулях с 10 по 12 предполагается, что слушатели создают своего бота полностью, однако они могут перенести пакет business\_service в свой бот и сделать вызовы функций этого пакета из своего бота. В этом случае, им не потребуется повторно писать функции бота, которые они ранее уже написали
* Перед началом работы для каждого слушателя необходимо создать бот, через @BotFather телеграмм, и выдать каждому идентификатор его бота. Этот идентификатор необходимо вписать в файл DemoBot\config.py вместо символов-звездочек
* В таблице приведены файлы, с которыми должны работать слушатели:

|  |  |  |
| --- | --- | --- |
| Модуль | Файлы для работы слушателей | Упражнения (направления) |
| 6 | business\_service\start.py  business\_service\filter.py  business\_service\regexp.py | Приветствие пользователя  Фильтр плохих слов  Регулярные выражения |
| 7 | business\_service\rel\_db.py | Реляционные базы данных |
| 8 | business\_service\nosql\_db.py | No SQL базы данных |
| 9 | business\_service\ws\_client.py | Работа с веб-сервисами |

# Часть 1. Элементарное программирование

Цель: Знакомство с Python и его базовым синтаксисом

## Базовый синтаксис языка Python

* Расскажите слушателям историю языка Python (слайд 4)
* Покажите слушателям пример «hello world», объясните структуру программы
* Расскажите слушателям, что программа состоит из команд, которые работают с данными. Обратите их внимание, что программа работает с переменными, через которые мы получаем доступ к разным структурам данных, например к набору символов, называемых строками. К данным мы получаем доступ через переменные. Объясните слушателям требования к именам переменных в Python
* Покажите слушателям более сложный пример – решение квадратного уравнения. Объясните необходимость оператора условного перехода. Сделайте ссылку, что он подробно будет обсуждаться далее
* Попросите слушателей запрограммировать расчет физических формул, дополнив файл Module1\Exercises\physics.py своими решениями и проверьте их
* Объясните слушателям важность отступов на примере задачи «Квадратное уравнение». Обратите внимание, что они должны использовать или только пробелы или только табуляции.

## Среды разработки Python

* Расскажите слушателям об интерпретаторе командной строки Python. Обратите внимание слушателей, что интерпретатор удобен, когда вы изучаете какую-то библиотеку python, когда пишите несложный скрипт автоматизации, но программы лучше разрабатывать в специализированных инструментальных средах
* Покажите слушателям ключи командной строки интерпретатора Python
* Расскажите слушателям о инструментальной среде PyCharm. Обратите внимание на использование инструмента для полноценной разработки приложений. Если вы работаете в этой среде, продемонстрируйте ее слушателям, покажите основные действия, связанные
* Расскажите слушателям об инструменте Visual Studio Code. Продемонстрируйте, как открыть каталог, создать файлы и запустить приложение на Python. Эта среда является рекомендованная в этом тренинге
* Расскажите слушателям о рекомендованных расширения к Visual Studio Code:
  + Python – основное расширение для запуска Python-программ
  + Python Preview – расширение, позволяющее разобраться как устроена внутренняя память Python-программ
  + Python Indent – расширение, которое автоматически корректирует необходимые отступы
  + Python Coding Conventions – расширение, позволяющее с помощью горячих клавиш отформатировать код программы, так чтобы он соответствовал общепринятым правилам в Python

## Элементарные структуры данных

* В этом модуле мы изучим основные типы данных в языке Python и операции над этими типами данных
* Расскажите слушателям о том, что такое тип данных. Обратите их внимание, что операции с типами данных не обязательно обозначаются традиционно знаками +, - и т.д., но и могут быть обозначены вполне понятными именами, например append, count и т.д.
* Обратите внимание слушателей, что тип переменной определяется автоматически на основе типа выражения, стоящего в правой части операции присваивания
* Расскажите о численных типах данных:
  + int – целые числа
  + float - числа с плавающей точкой
  + complex – комплексные числа
* Обратите внимание слушателей, что численные типы данных поддерживают большую арифметику
* Продемонстрируйте слушателям операции с числами
* Расскажите слушателям о битовых операциях. Обратите их внимание на то, что данные операции используются в алгоритмах шифрования, работы с внешними устройствами. Продемонстрируйте эти операции. Если аудитория не знает данных операций, пожалуйста, расскажите о них.
* Расскажите слушателям о комплексных числах. Покажите пример программы и опишите как можно создать комплексное число и как можно получить доступ к его элементам
* Объясните слушателям, что такое строка. Обратите их внимание на то, что строка – неизменяемый тип данных, а также, что по умолчанию строка хранит символы в формате Unicode
* Расскажите слушателям о байтах и байтовых массивах. Обратите внимание, что байтовый массив, в отличие от строк можно менять, но указывать нужно код символа в виде целого числа
* Продемонстрируйте типовые операции со строками. Обратите внимание слушателей на разницу между операциями index() и find(), а также rindex() и rfind()
* Попросите слушателей выполнить упражнение «Элементарные операции со строками»
* Расскажите слушателям про оператор slice. Объясните как работает оператор slice, если параметры оператора отрицательные, используя слайд
* Обсудите со слушателями форматированный вывод. Подробно объясните программу, приведенную на слайде
* Расскажите слушателям про списки и кортежи. Обратите их внимание, на необходимость использования списков только тогда, когда нам необходимо менять их содержимое. Если же нам содержимое контейнеров необходимо только для чтения, тогда следует использовать кортежи – они занимают меньше памяти и более производительны
* Продемонстрируете слушателям основные операции над списками
* Обратите внимание слушателей, что кортежи поддерживают такие же операции, как и в списках, кроме изменяющих содержимое кортежа. В любом случае, операции сложения списков со списками и кортежа с кортежами создают новый список или кортеж соответственно
* Объясните слушателям как будет работать код, в котором к концу списка добавляют в качестве элемента этот же список
* Расскажите слушателям про операцию присваивания значений кортежа или списка переменным, находящимся слева от знака «=»
* Попросите слушателей выполнить упражнение на операции со списками и кортежами. Обратите их внимание, на то что часть операций, которые удалось выполнить со списком не будут доступны для кортежей. Обсудите почему
* Расскажите слушателям о словарях. Продемонстрируйте работу со словарем
* Расскажите слушателям о множестве. Обратите их внимание, что множество не содержит дубликатов
* Расскажите слушателям об операциях над множествами. Если будет необходимость, напомните им основные идеи из теории множеств (<https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2#%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D1%8B%D0%B5_%D0%BF%D0%BE%D0%BD%D1%8F%D1%82%D0%B8%D1%8F>)
* Продемонстрируйте операции над множествами слушателям
* Попросите слушателей выполнить элементарные операции над множествами
* Расскажите слушателям о возможности преобразования типа данных. Продемонстрируйте возможности преобразования типа данных

## Операторы Python

* Расскажите слушателям о необходимости операторов логики ветвления
* Объясните слушателям логику работу оператора if используя схему его алгоритма на слайде
* Объясните слушателям, что оператор if и далее оператор while в качестве условия рассматривают значение True или False. Зная как трактуется то или иное значение выражения можно упросить запись. Продемонстрируйте слушателям примеры применения оператора if (Module3\Live\if.py)
* Расскажите слушателям про условное выражение. Это выражение используется в ситуации, когда в зависимости от условия одной переменной надо присвоить или одно или другое значение. В таких простых ситуациях оператор if будет выглядеть очень громоздко
* Расскажите слушателям для чего применяется оператор цикла while. В следующем разделе тренинга он будет использоваться в реализации алгоритмов. Объясните работу этого оператора с использованием схемы на слайде
* Напомните слушателям про структуры данных. Структуры данных хранят другие данные в виде последовательности, называемой контейнером. К таким последовательностям в Python обычно относят списки и кортежи
* Расскажите слушателям как работает оператор for, используя схему на слайде
* Расскажите слушателям об операторе break. Этот оператор используется для досрочного завершения цикла без участия условия его продолжения
* Расскажите слушателям об операторе continue. Этот оператор используется для досрочного завершения блока цикла и перехода к следующей итерации

## Функции

* Дайте слушателям определение функции, расскажите о задачах, которые они решают
* Расскажите о синтаксисе определения функции. Обратите внимание слушателей, что в в одном файле Python может быть только одна функция с одним и тем же именем. Если в файле несколько функций с одинаковым именем, то кроме последней определенной, все остальные одноименные функции игнорируются
* Объясните слайд, в котором показан пример использования функций. Обратите внимание слушателей на механизм передачи значений параметров, которые получают начальное значение в момент вызова функции
* Расскажите слушателям про лямбда функцию. Не следует давать много теории по функциональному программированию, можно лишь ограничиться фразой, что лямбда-функция – это небольшая функция, которая используется для указания критерия сравнения в некоторых функциях Python.
* Объясните слушателям пример с функцией filter на слайде. Обратите их внимание, что благодаря лямбда-функции, функция filter получается универсальной и не зависит от способа фильтрации, который как раз и определяется этой лямбда функцией.
* Расскажите слушателям о типовом алгоритме обработки данных «Map-Filter-Reduce» (<https://codecamp.ru/blog/map-filter-reduce/>). Здесь участвует три функции – map(), filter() и reduce(). Функция map преобразует список в некоторый другой список. Функция filter отбирает список, согласно критерию, а reduce() выполняет некоторую операцию над всеми элементами списка, в результате чего получается некоторое значение. Здесь лямбда-функция играет роль функции трансформации элемента списка, функции, определяющий критерий отбора и функции, выполняющей накапливающую операцию. Такой подход часто используется при естественной обработке текста.
* Объясните слушателям пример реализации алгоритма «Map-Filter-Reduce». В этом примере, исходный список содержит кортежи из двух элементов – строки и числа. С помощью функции map() мы создаем список строк, содержащий ровно то количество символов, которое было задано во втором элементе кортежей исходного списка. Далее, с помощью функции filter() мы оставляем в списке только те строки, длина которых больше трех. И наконец, функция reduce суммирует длины строк.

# Часть II. Алгоритмы

Цель: Изучение типовых алгоритмов, которые могут применяться при разработке ботов

## Работа с текстом

В этом модуле будут изучены возможности Python для реализации простейших задач по работе с текстом.

### 6.1. Случайные приветствия

* Поставьте слушателям задачу: «наш бот будет приветствовать пользователя случайными фразами. Нам необходимо прочитать содержимое файла, а потом выбрать случайную строку и вернуть ее как ответ нашего бота»
* Для решения этой задачи необходимо изучить работу с файлами. Расскажите слушателям о том, что такое файлы, каких типов они бывают (<https://www.pc-school.ru/chto-takoe-fajl-svojstva-fajla/>). Скажите слушателям, что мы будем работать с текстовыми файлами
* Расскажите слушателям про функцию open(), которая открывает доступ к файлам. Расскажите о режимах открытия файла
* Расскажите слушателям о необходимости работы с файлом через контекстный менеджер (оператор with). Объясните им, что файл можно закрыть с помощью функции close(), однако более безопасным это будет делать автоматически под контролем менеджера контекста
* Расскажите слушателям о способах чтения файла. В первом способе мы используем функцию read(), которая читает все содержимое файла в одну строку, если read() вызывается без параметра. Функция read() может прочитать из файла указанное количество символов, переданное в ее параметре. Во втором способе используется оператор for, который позволяет читать построчно. Обратите внимание слушателей, что для работы с файлами, содержащих национальные символы следует использовать библиотеку codecs
* Расскажите слушателям о том, как записывать в файл в обычной кодировке и кодировке utf-8 (если записываемые данные содержат символы национальных алфавитов)
* Расскажите слушателям о понятии сериализации и десериализации структур данных. Подробно можно прочитать здесь <https://pep8.ru/doc/dive-into-python-3/15.html>.
* Объясните пример с сериализацией и сериализацией структуры данных с помощью библиотеки pickie
* Расскажите слушателям о формате JSON. Обратите их внимание на то, что для JSON формата поддерживаются не все типы данных. Неподдерживаемые типы данных также можно сохранять в JSON, однако это потребует дополнительных действий, которые не обсуждаются в этом тренинге.
* Объясните слушателям пример с сериализацией и десериализацией JSON файла. Подробности можно прочитать в этой статье: <https://pep8.ru/doc/dive-into-python-3/15.html>
* Расскажите слушателям об обработке ошибок в Python. Для обработки ошибок создаются специальные объекты, называемые исключениями. Подробно про исключения можно почитать в этой статье: <https://pythonworld.ru/tipy-dannyx-v-python/isklyucheniya-v-python-konstrukciya-try-except-dlya-obrabotki-isklyuchenij.html> . Обратите внимание слушателей, что блоков except может быть сколько угодно и что блок finally гарантировано выполнит код, даже если в код, в котором могут быть ошибки выполнился успешно и не прерывался. Важно отметить, что если подходящий обработчик исключения не будет найден, то функция будет прервана и поиск обработчика продолжится в вызывающей функции. Такой подход называется размотка стека. Если подходящий обработчик так и не будет найден – программа будет завершена с ошибкой.
* Объясните слушателям пример обработки исключений.

### 6.2. Задача «Вежливый ответ пользователю»

* Расскажите слушателям подробно о методе split. Он рассматривался в предыдущей части, поэтому напомните об этом
* Покажите пример разбиения строки на элементы с помощью split()
* Расскажите слушателям про соединение строк методом join()
* Расскажите слушателям об использовании метода replace() для замены строк
* Попросите слушателей решить задачу «Вежливый ответ пользователю»

### 6.3. Задача «Дотошный бот»

* Дайте слушателям определение регулярного выражения
* Обсудите со слушателями таблицу с примерами регулярных выражений
* Расскажите слушателям об основных шаблонах регулярных выражений
* Расскажите и продемонстрируйте основные функции по работе с регулярными выражениями
* Покажите слушателям сайт [https://regex101.com](https://regex101.com/) и объясните, как им пользоваться для тестирования регулярных выражений
* Расскажите о флагах для настройки функций регулярных выражений
* Попросите слушателей решить задачи на составление регулярных выражений
* Попросите слушателей решить задачу «Дотошный бот»

## Реляционные базы данных

В этом модуле будут изучены основы реляционных баз данных, а также показаны приемы работы с ними на примере Postgre SQL

### 7.1. Постановка задачи «Расписание»

* Обсудите со слушателями задачу «Расписание». Обратите внимание слушателей на построение фразы запроса. Напомните им про регулярные выражения
* Обсудите со слушателями с какими данными будет работать бот. Пусть они выскажут идеи, какие данные могли бы храниться в нашей базе данных

### 7.2. Введение в реляционные базы данных

* Дайте слушателям определение базы данных
* Обсудите со слушателями требования к информации, которая должна содержаться в базе данных
* Расскажите слушателям о системах управления базами данных
* Расскажите слушателям о реляционной модели данных. Обратите внимание слушателей на то, что таблицы в базе данных ничего друг о друге не знают, поэтому при работе с данными, как правило, выполняется взаимодействие с несколькими таблицами
* Выполните со слушателями проектирование таблиц базы данных для задачи «Расписание». Обратите внимание слушателей на необходимость введения суррогатного ключа в таблице базы данных. Именно с помощью этого поля из одной записи таблицы, можно ссылаться на запись в другой таблице. Подробности о суррогатном ключе можно прочитать [здесь](http://www.ibase.ru/natural-keys-versus-atrificial-keys-by-tentser/#:~:text=%D0%A1%D1%83%D1%80%D1%80%D0%BE%D0%B3%D0%B0%D1%82%D0%BD%D1%8B%D0%B9%20%D0%9A%D0%BB%D1%8E%D1%87%20(%D0%A1%D0%9A)%20%E2%80%93%20%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8,%D0%B2%D1%8B%D1%81%D1%82%D1%83%D0%BF%D0%B0%D0%B5%D1%82%20%D0%B0%D0%B2%D1%82%D0%BE%D0%B8%D0%BD%D0%BA%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D0%BD%D0%BE%D0%B5%20%D0%BF%D0%BE%D0%BB%D0%B5%20%D1%82%D0%B8%D0%BF%D0%B0%20INTEGER.).
* Расскажите слушателям о языке структурированных запросов. Объясните, что этот язык применяется как для общей работы с таблицами, так и для манипуляции с данными, которые хранятся в этих таблицах. Подробно о языке структурированных запросов можно прочитать [здесь](http://www.sql-tutorial.ru/ru/content.html).
* Объясните слушателям команду CREATE TABLE и обсудите с ними эту команду для двух таблиц из задачи «Расписание». Обратите внимание слушателей на автоматическую генерацию значений суррогатных ключей
* Расскажите слушателям, что необходимо для подключения к БД Postgre. Обсудите с ними пример Module7\Live\config.py
* Напомните слушателям необходимость использования менеджера контекста. Расскажите, как с его помощью подключиться к базе данных Postgre SQL
* Расскажите слушателям о курсоре для взаимодействия с базой данных.
* Обсудите со слушателями как создавать таблицу с помощью библиотеки psycop2. Обсудите пример Module7\Live\create-table.py
* Попросите слушателей написать скрипт, который создаст таблицы расписания при запуске демонстрационного бота. Код создания следует разместить в функции init( ) пользовательского модуля
* Напомните слушателям о [механизме обработки ошибок в Python](https://pythonworld.ru/tipy-dannyx-v-python/isklyucheniya-v-python-konstrukciya-try-except-dlya-obrabotki-isklyuchenij.html).
* Расскажите слушателям про SQL-иньекции. Объясните их опасность для данных с помощью примера показанного на слайде
* Обсудите со слушателями формат SQL-команды INSERT для вставки данных. Подробности можно прочитать в [учебнике по SQL](http://www.sql-tutorial.ru/ru/content.html)
* Объясните слушателям пример Module7\Live\insert-one-table.py
* Дайте обзор слушателям пакета datetime для формирования строкового представления времени и дат, объясните пример Module7\Live\time.py, показанный на слайде
* Попросите слушателей выполнить упражнение на вставку данных для задачи «Расписание». Слушатели должны будут внести исправления в примере кода. Напомните слушателям о [типах данных, применяемых в таблицах Postgre](https://postgrespro.ru/docs/postgrespro/10/datatype).
* Расскажите слушателям об SQL-команде SELECT. Обсудите формат этой команды, показанный на слайде
* Расскажите слушателям как обрабатывать результат SQL-запроса SELECT и обсудите пример в Module7\Live\select-onetable.py
* Обсудите со слушателями выборку из нескольких таблиц на основе примера, показанного на слайде. Объясните слушателям формат команды SELECT для выборки с INNER JOIN
* Объясните слушателям пример Module7\Live\select-manytable.py
* Обсудите со слушателями необходимость применения LEFT JOIN для нашей задаче. На текущий момент мы видим в результате запроса только станцию отправления. Нам бы хотелось отобразить полное расписание со станциями отправления и прибытия. Подробности про LEFT JOIN можно прочитать [здесь](https://html5css.ru/sql/sql_join_left.php)
* Обсудите со слушателями пример выборки LEFT JOIN в Module7\Live\select-leftjoin.py
* Дайте задания слушателям написать несколько скриптов по выборке расписания. Проверьте эти скрипты, объясните слушателям их ошибки
* Предложите слушателям на самостоятельное изучение SQL-команды UPDATE, которая предназначена для редактирования записей базы данных. Обратите внимание слушателей, на важность условия WHERE в команде UPDATE
* Попросите слушателей выполнить задачу «Расписание» на основе полученной в этом модуле информации.

## NoSQL базы данных

В этом модуле мы изучим основы взаимодействия с NOSQL базами данных.

* Расскажите слушателям об идее NoSQL баз данных. Подробнее можно прочитать [здесь](https://www.bigdataschool.ru/wiki/nosql)
* Расскажите слушателям, что мы будем работать с СУБД MongoDB, которая имеет достаточно большую производительность и развитые средства запроса для извлечения данных. Материалы по MongoDB можно прочитать на [официальном сайте](https://www.mongodb.com/) или [здесь](https://ru.wikipedia.org/wiki/MongoDB).
* Объясните слушателям соответствие понятий, которые используется в реляционных базах данных и NoSQL базах данных
* Объясните слушателям пример Module8\Live\mongo-connect.py где показано подключения к базе данных
* Расскажите слушателям о структуре документов, которые хранятся в базе данных. Обратите их внимание, что в одной и той же коллекции, могут хранится документы с разным набором полей
* Обратите внимание слушателей, что документам автоматически назначается ключ, представляющий собой атрибут «\_id» с уникальным значением ключа
* Объясните слушателям пример Module 8\Live\mongo-intro.py. Обсудите с ними операции вставки одного и нескольких документов
* Расскажите слушателям каким образом можно получить документы из коллекции. Объясните слушателям пример Module 8\Live\mongo-findall.py
* Объясните слушателям, как зная ключ получить конкретный документ из базы данных. Обсудите со слушателями пример Module8\Live\mongo-findbykey.py
* Расскажите слушателям про язык запросов для получения документов, на основе заданных критериев поиска. Подробнее можно прочитать на [официальном сайте](https://www.mongodb.com/) или [здесь](https://habr.com/ru/post/134590/). Обсудите со слушателями пример Module8\Live\mongo-query.py
* Объясните слушателям как изменять содержимое документов, сохраненных в базе данных. Покажите им пример Module 8\Live\mongo-update.py
* Расскажите слушателям об удалении документов из базы данных. Обсудите с ними пример Module 8\Live\mongo-delete.py
* Напомните слушателям о механизме обработки ошибок в Python. Обратите их внимание на то, что описание конкретных ошибок, которые можно обработать, следует искать в документации к библиотеке pymongo

## Основы сетевого взаимодействия

В этом модуле мы изучим основы взаимодействия типа «клиент-сервер» на примере взаимодействия с сервисом центробанка

### 9.1. Постановка задачи «Бот-валютообменник»

* Объясните слушателям цель данного модуля, которая заключается в освоении простейшего взаимодействия с http-серверами
* Расскажите слушателям про веб-сервисы. Вы можете использовать этот материал.

### 9.2. Протокол http

* Объясните слушателям принцип «Клиент-сервер». Подробно про этот принцип хорошо рассказано в [этой статье](https://habr.com/ru/post/495698/).
* Расскажите слушателям о http-протоколе. Прокомментируете слайд, показывающий, что web-документ собирается на клиенте из разных источников с помощью http-протокола. Подробно об http-протоколе можно прочитать [здесь](https://developer.mozilla.org/ru/docs/Web/HTTP/Overview#:~:text=HTTP%20%E2%80%94%20%D1%8D%D1%82%D0%BE%20%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB%2C%20%D0%BF%D0%BE%D0%B7%D0%B2%D0%BE%D0%BB%D1%8F%D1%8E%D1%89%D0%B8%D0%B9%20%D0%BF%D0%BE%D0%BB%D1%83%D1%87%D0%B0%D1%82%D1%8C,%D0%B1%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80%D0%BE%D0%BC%20(web%2Dbrowser).).
* Объясните слушателям структуру http-запроса. Расскажите для чего необходимы команды, заголовки и тело запроса
* Объясните слушателям смысл основных команд http-запроса. Обратите их внимание на то, что параметры запроса в GET передаются в URL. Сами результаты GET-запроса могут быть кэшированы, т.е. идемпотентны (при повторной выдаче такого же запроса по умолчанию возвращаются такой же ответ). Обратите внимание слушателей, что для отправки больших данных на сервер по http-протоколу используется POST запрос. Его ответ не кешируется
* Обсудите со слушателями информацию, которую несут в себе заголовки. Вы можете посмотреть более подробный перечень заголовков [здесь](https://developer.mozilla.org/ru/docs/Web/HTTP/%D0%97%D0%B0%D0%B3%D0%BE%D0%BB%D0%BE%D0%B2%D0%BA%D0%B8)
* Объясните слушателям необходимость MIME типов и заголовка Content-Type. Именно по этому заголовку клиент (браузер) опознает, что предлагает получить http-сервер – текстовый файл, HTML-документ или графическое изображение
* Объясните слушателям структуру http-ответа. Отметьте важность кода ответа, по которому можно определить является ли ответ корректным или содержит ошибку. Описание заголовков можно посмотреть в этой [статье](http://allerrorcodes.ru/http-2/). Обсудите со слушателями часто используемые коды ответов

### 9.3. Выполнение http-запросов на Python

* В этом модуле слушатели изучат простейшую библиотеку http-клиента – Requests. После изучения модуля «Асинхронное программирование» слушатели смогут использовать аналогичную по возможностям библиотеку aiohttp
* Дайте слушателям обзор основных методов http-клиента. Покажите пример отправки запроса Module 9\Live\simple-request.py
* Попросите слушателей ознакомится с [документацией](https://cbu.uz/ru/arkhiv-kursov-valyut/veb-masteram/), представленной на сервисе Центробанка. На примере этого сервиса мы изучим как отправлять и получать запросы, а также его использование будет одной из функций нашего бота
* Объясните слушателям пример Module 9\Live\get-rub.py, который запрашивает курс рубля по отношению к национальной валюте
* Попросите слушателей написать аналогичные скрипты согласно заданию на слайде упражнения «Работа с сервисом Центробанка»
* На примере отладочного сервиса <https://httpbin.org> покажите слушателям как отправлять запросы с параметрами. Внимательно просмотрите с ними ответ сервера, который напечатает скрипт Module 9\Live\get-with-param.py
* Покажите слушателям как отправлять файл на сервер. Используйте для этого пример Module 9\Live\upload-file.py
* Напомните слушателям об обработках ошибок в Python на основе исключений. Обсудите с ними, ситуации, когда придется обрабатывать исключения, показанные на слайде
* Попросите слушателей завершить задачу «Бот-валютообменник»

## Система контроля версий GIT

* Расскажите слушателям о необходимости использования системы контроля версий
* Расскажите слушателям о том, что представляет собой система контроля версий
* Расскажите слушателям о том, что представляет собой репозиторий с точки зрения его устройства в файловой системе
* Продемонстрируйте слушателям команду git init для создания репозитория в рабочем каталоге
* Продемонстрируйте слушателям команду git clone для клонирования удаленного репозитория. Вы можете использовать свой репозиторий на <https://github.com> или выбрать любой по вашему усмотрению
* Расскажите слушателям, что представляет собой коммит в GIT. Обратите их внимание, что каждый коммит это снимок файловой системы, а не просто сохранение изменений в файлах. Подробно об этом можно прочитать [здесь](https://git-scm.com/book/ru/v2/%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5-%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D1%8B-Git).
* Расскажите слушателям о состоянии файлов с точки зрения GIT. Обсудите с ними диаграмму «Состояния файла репозитории» на слайде и объясните смысл каждого из состояний
* Продемонстрируйте слушателям основные команды работы с репозиторием. Обратите их внимание на ключ «-m» в команде git commit. Если этот ключ не указывать, для записи комментария будет открыт текстовый редактор, после закрытия которого, будет фактически сделан коммит
* Продемонстрируйте слушателям как получить историю коммитов. Обратите их внимание на то, что каждому коммиту присваивается уникальный ключ, называемый хэшем. По хэшу коммита можно ссылать на него, выгружать отдельную копию репозитория, отражающего состояние данного коммита в рабочий каталог и так далее
* Продемонстрируйте слушателям как можно дополнить коммит, исключить файл из индекса перед коммитом и как отменить все изменения, сделанные после коммита. Объясните слушателям, какую пользу они приносят
* Объясните слушателям с какой целью используется ветвление в системах контроля версий
* Расскажите о том, как устроено ветвление в GIT, обсудив иллюстрацию на слайде «Ветвь». Подробнее о работе с ветвлением в GIT можно прочитать [здесь](https://git-scm.com/book/ru/v2/%D0%92%D0%B5%D1%82%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5-%D0%B2-Git-%D0%9E-%D0%B2%D0%B5%D1%82%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B8-%D0%B2-%D0%B4%D0%B2%D1%83%D1%85-%D1%81%D0%BB%D0%BE%D0%B2%D0%B0%D1%85).
* Продемонстрируйте слушателям команды работы с ветвями в GIT
* Объясните слушателям смысл слияния ветвей в GIT, обсудите с ними, еще раз, пользу от использования ветвления
* Продемонстрируйте слушателям команду слияния ветвей в GIT
* Объясните слушателям почему могут возникать конфликты при слияниях и расскажите как их решать. Продемонстрируйте использование графического инструмента разрешения конфликтов и применение стандартного текстового редактора
* Расскажите слушателям необходимость удаленного репозитория. Попросите каждого из них создать аккаунт на <https://github.com> , если он у них отсутствует
* Попросите слушателей склонировать удаленный репозиторий созданный на предыдущем шаге и пусть слушатели добавят несколько файлов в рабочий каталог, а затем сделают коммит
* Продемонстрируйте слушателям команды работы с удаленном репозиторием. Объясните смысл каждой из них
* Расскажите слушателям о графических инструментах работы с GIT. Если в тренинге вы будете использовать Visual Studio Code установите расширения из Git Extension Pack
* Попросите слушателей выполнить упражнение «Создание репозитория проекта «БОТ». С этого момента мы начинаем разработку собственного бота

## Основы асинхронного программирования

В этом модуле слушатели познакомятся с основами асинхронного программирования, знание которых им понадобится при разработке собственных ботов с использованием библиотеки aiogram. Подробнее про концепцию асинхронного программирования вы можете прочитать [здесь](http://www.dabeaz.com/coroutines/)

* Обсудите со слушателями недостатки синхронного программирования. Обратите их внимание на то, что в один момент времени выполняется только одна задача, даже если она занята какой-то блокирующей операцией. Это не позволяет эффективно расходовать вычислительные, сетевые и прочие ресурсы. Для иллюстрации проблемы объясните слушателям работу примера Module 11\Live\reddit-sync.py
* Объясните слушателям принципы асинхронного программирования. В этом случае, задачи могут выполняться одновременно, даже если какая-то задача выполняет блокирующую операцию
* Расскажите слушателям как устроено асинхронная подсистема в Python. Python является однопоточным, поэтому все задачи разделяют один поток, которыми управляет специальный планировщик. Такие задачи принято называть корутинами, а планировщик – циклом сообщений (event loop)
* Покажите слушателям, используя пример Module 11\Live\async-intro.py как определяются корутины и простейший цикл сообщений. Обратите их внимание на ключевые слова async и await
* Обсудите со слушателями пример Module 11\Live\reddit-async.py, чтобы проиллюстрировать преимущества асинхронного программирования. Обратите их внимание на то, что каждая из задач, управляющая вызовом функции get\_reddit\_top() выполняется «параллельно» и благодаря циклу сообщений мы можем запустить выполнение сразу нескольких запросов одновременно, а обрабатывать их результаты по мере получения ответа от сервисов
* Попросите слушателей реализовать отдельный скрипт, который бы выполнял «одновременно» все запросы к сервису Центробанка для получения курсов валют согласно условиям упражнения «Бот-валютообменник»

## Разработка телеграм бота

В этом модуле мы полностью разработаем собственный бот и перенесем в него все его функции, разработанные в предыдущих упражнениях

* Расскажите слушателям про архитектуру бота, основанного на запросах. Обратите их внимание, что в этом варианте бот сам опрашивает сервера телеграм на наличие для него новых сообщений. Подробно про архитектуру бота можно прочитать [здесь](https://www.servicedesk.site/ru/2019/08/09/%D1%8F%D0%BA-%D1%81%D1%82%D0%B2%D0%BE%D1%80%D0%B8%D1%82%D0%B8-%D1%82%D0%B5%D0%BB%D0%B5%D0%B3%D1%80%D0%B0%D0%BC-%D0%B1%D0%BE%D1%82%D0%B0-%D0%BF%D1%80%D0%B0%D0%B2%D0%B8%D0%BB%D1%8C%D0%BD%D0%BE/).
* Расскажите слушателям о веб-хуках и обсудите с ними как изменяется архитектура бота
* Обсудите со слушателями роли каждого из слоев бота. Обратите внимание, что каждый слой должен решать строго определенные задачи, которые никак не должны пересекаться
* Расскажите слушателям про пакетную организацию кода. Подробно об этом можно прочитать [здесь](https://devpractice.ru/python-modules-and-packages/).
* Объясните слушателям пример Module 12\Live\use-packet.py, который проиллюстрирован для удобства объяснения на слайде
* Попросите слушателей выполнить упражнение «Проектируем стартовый код бота»
* Объясните слушателям процесс получения токена бота с помощью бота @BotFather. Попросите слушателей получить токен для их бота
* Объясните слушателям как строится типовой обработчик сообщений в библиотеке aiogram в режиме поллинга и на веб-хуках. Обсудите со слушателями простейший бот из примера Module 12\Live\simple-polling-bot.py
* Объясните слушателям как установить фильтры обработчиков с помощью декораторов. Используйте для этого пример Module 12\Live\answer-bot.py
* На основе этого же примера объясните слушателям как отправлять видео, звук и голос в ответ на сообщение, полученное ботом
* Попросите слушателей разработать бот «Секретарь» перенеся в пакет «business\_service» все функции из предыдущих упражнений, а также создать обработчики и вызвать эти функции
* Расскажите слушателям о типах клавиатур бота. Подробно информацию можно прочитать [здесь](https://surik00.gitbooks.io/aiogram-lessons/content/chapter5.html).
* Объясните слушателям как создать ReplyKeyboardMarkup на примере Module 12\Live\replykeyboardmarkup-bot.py. Обратите их внимание, что для создания клавиатуры в ответе используется параметр reply\_markup. Здесь могут быть и другие варианты, которые в этом тренинге не обсуждаются, например шрифтовое оформление текста
* Обсудите со слушателями параметры, которые можно использовать при создании объектов ReplyKeyboardMarkup и KeyboardButton
* Объясните слушателям, полезность InlineKeyboardMarkup – инлайновая клавиатура. Такая клавиатура вставляется в тело ответа, которую пользователь может использовать для активации разных действий – от простого текстового ответа до перехода в нужный чат и вставки предустановленной строки сообщения в этом чате
* Объясните слушателям как создать InlineKeyboardMarkup и написать обработчик для него. Используйте для этого пример Module 12\Live\inlinekeyboardmarkup-bot.py
* На этом же примере объясните слушателям как создать ярлыки на внешние ресурсы, а также на текущий и выбранный пользователем чат
* Попросите пользователей выполнить упражение «Бот «Секретарь» с клавиатурой»

## Подведение итогов

* Подведите итоги тренинга со слушателями
* Напомните им чему они научились на тренинге
* Опишите им дальнейшее направление обучения