Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский университет ИТМО»

(Университет ИТМО)

Факультет инфокоммуникационных технологий

Образовательная программа Мобильные и сетевые технологии

Направление подготовки 09.03.03 Прикладная информатика

О Т Ч Е Т

об учебной, ознакомительной практике

Тема задания: Разработка Telegram чат-бота для игры «Что? Где? Когда?»

Обучающийся: Шугинин Ю. А., группа К33402

Руководитель практики от университета: Береснев А. Д., старший преподаватель ФИКТ Университета ИТМО

Практика пройдена с оценкой ____

Дата ____

Санкт-Петербург, 2022

РЕФЕРАТ

В ходе практики разрабатывался чат-бот для игры в «Что? Где? Когда?». Таким образом решалась проблема отсутствия адаптированности сайта с крупнейшей базой вопросов под использование на мобильных устройствах.

Цель практики – разработка чат-бота для мессенджера Telegram, с помощью которого можно играть в «Что? Где? Когда?».

Для достижения цели были решены следующие задачи:

– Изучение методов взаимодействия с Telegram API.

– Изучение методов синтаксического анализа веб-страницы.

– Выбор средств программной реализации.

– Анализ существующих решений.

– Продумывание функционала чат-бота и архитектуры программы.

– Разработка чат-бота.

Для разработки чат-бота применялись следующие технологии: система контроля версий Git, язык программирования Python 3.8, а также подключаемые модули requests, telebot, BeautifulSoup, re, time, threading.

Результатом стал разработанный чат-бот, выполняющий все заявленные функции. С полным кодом программы можно ознакомиться в моём репозитории на GitHub (https://github.com/YuriiShuginin/MrHookBot).

Отчёт содержит 18 страниц, 8 рисунков и 11 интернет-источников.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Спортивное «Что? Где? Когда?» – соревновательная версия известной телевизионной игры, по которой проводится множество турниров различной сложности, доступных любому заинтересованному коллективу из 6 человек. Существует сайт с базой вопросов (https://db.chgk.info/), куда помещаются пакеты вопросов всех официальных турниров, начиная с 1990-го года. Существование такой базы необходимо для Движения, чтобы организаторы будущих турниров могли гарантировать новизну и уникальность вопросов. На данный момент на сайте присутствует 275929 вопросов «Что? Где? Когда?».

В сферу спортивного «Что? Где? Когда?» до сих пор не пришли большие деньги, как, например, в сферу киберспорта. Поэтому большинство процессов осуществляется энтузиастами. Неудивительно, что крупнейший интернет-ресурс с базой вопросов совсем не адаптирован для использования на мобильных устройствах. Поэтому в ходе практики разрабатывался чат-бот, решающий проблему комфортного использования вопросов из Базы для игры в «Что? Где? Когда?».

В качестве платформы для чат-бота был выбран мессенджер Telegram, активно развивающий открытые средства создания ботов для своей платформы. Компания периодически проводит так называемые контесты по созданию чат-ботов, в которых любой разработчик имеет возможность получить крупное денежное вознаграждение и даже место в штате корпорации [5].

Таким образом, актуальность индивидуального задания обеспечена, во-первых, решением проблемы достаточно популярного в странах СНГ сообщества интеллектуальных игр, а во-вторых, перспективностью овладевания навыками создания чат-ботов для Telegram.

Цель практики – разработка чат-бота для мессенджера Telegram, с помощью которого можно играть в «Что? Где? Когда?».

Для достижения цели были решены следующие задачи:

– Изучение методов взаимодействия с Telegram API.

– Изучение методов синтаксического анализа веб-страницы.

– Выбор средств программной реализации.

– Анализ существующих решений.

– Продумывание функционала чат-бота и архитектуры программы.

– Разработка чат-бота.

1 Технологические основы

1.1 Взаимодействие с Telegram API

Первым делом нужно зарегистрировать нового бота в Telegram. Для этого нужно обратиться к боту BotFather [6], чтобы задать минимальные настройки нового бота (название и никнейм) и получить токен доступа – уникальный идентификатор для обращения к своему боту через Telegram API. На рисунке 1 представлен алгоритм регистрации моего бота.

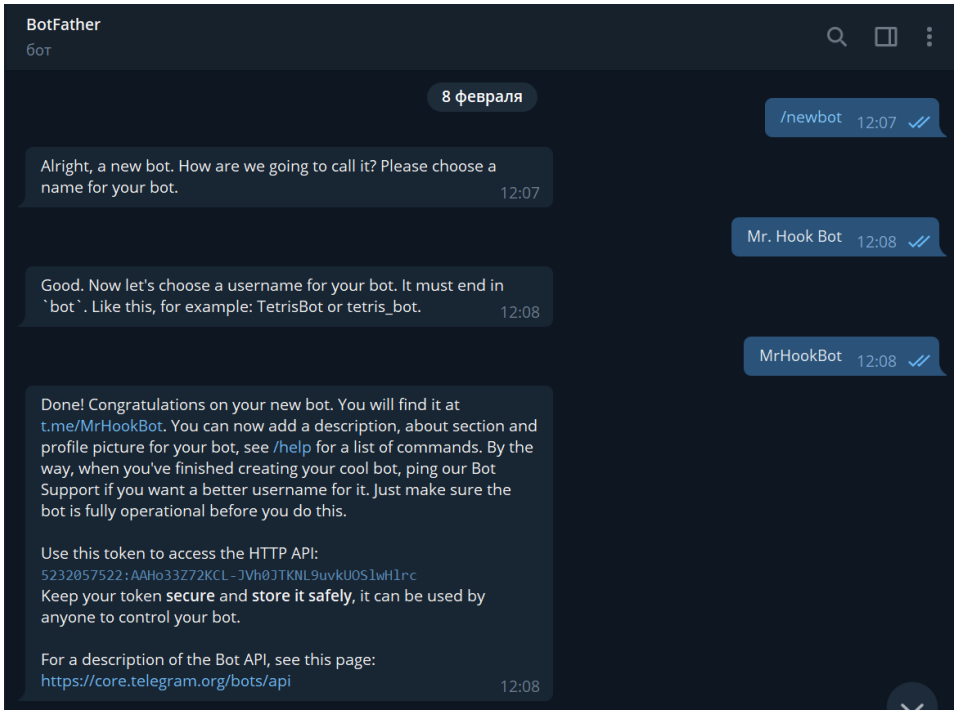

Рисунок 1 – Регистрация бота

В Telegram есть отдельный интерфейс для создания чат-ботов – Bot API. Согласно официальной англоязычной документации [4], Bot API — это интерфейс на основе HTTP, созданный для разработчиков, занимающихся созданием ботов для Telegram.

В разных языках программирования существуют библиотеки для работы с Bot API. Чаще всего используют библиотеку Telegraf для JavaScript или любую из целого множества библиотек для Python [11]. Я знаком с

Python, поэтому остановился на этом языке программирования и выбрал модуль PyTelegramBotAPI. Эта библиотека считается простой в понимании, её часто используют для обучения созданию чат-ботов для Telegram, а также она очень быстро получает поддержку всех обновлений Telegram Bot API (версия Bot API 5.7, выпущенная 31 января 2022, уже поддерживается).

1.2 Синтаксический анализ веб-страницы

У сайта базы вопросов (https://db.chgk.info/) нет никакого API. Но есть функция получения пакета случайных вопросов с заданными параметрами (рисунок 2).

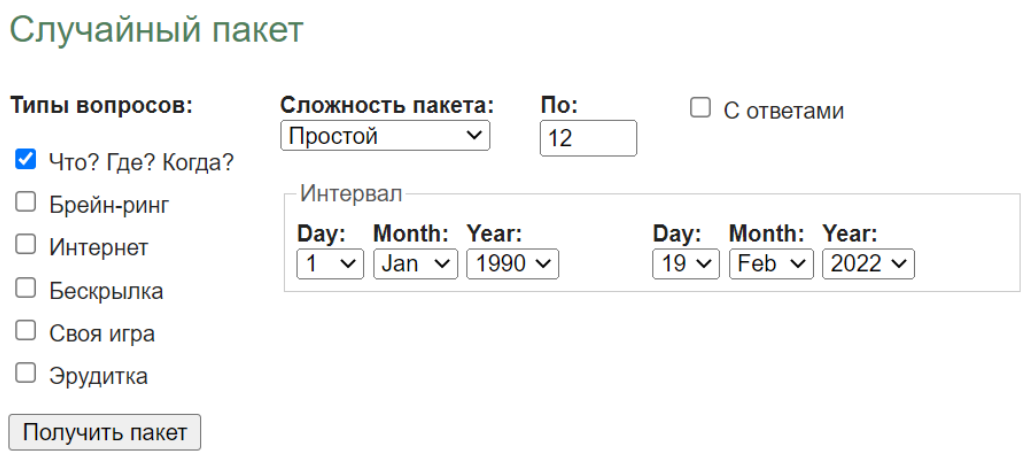

Рисунок 2 – Настройка параметров пакета случайных вопросов

При нажатии на кнопку «Получить пакет» происходит переадресация на страницу с URL, в котором после основного домена через знак «/» перечисляются выбранные параметры (рисунок 3).

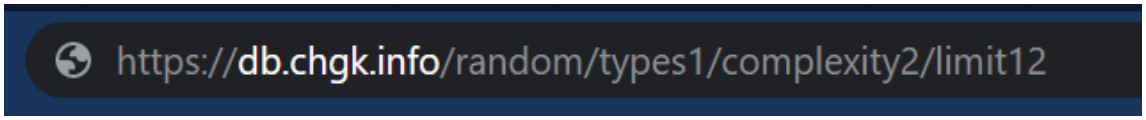

Рисунок 3 – Пример URL страницы случайного пакета с заданными параметрами

Таким образом, можно внутри нашей программы формировать URL по такому же принципу и из кода полученной страницы получать нужную информацию: тексты вопросов, раздаточные материалы, правильные ответы и комментарии к ответам.

Для GET-запроса к странице со случайным пакетом вопросов я использовал модуль requests [1]. С его помощью я сохраняю содержание кода полученной страницы в строковую переменную, которую потом анализирую с помощью модуля BeautifulSoup. Это Python-библиотека для извлечения данных из файлов HTML и XML. Она строит так называемое дерево разбора и даёт возможность навигации, поиска и изменений в нём [2].

1.3 Многопоточность

Простые боты работают по принципу вопрос-ответ. Пользователь отправляет известную боту команду, тот на неё отвечает. Но из-за того, что я задумал реализовать целую игровую сессию с таймером на обдумывание каждого вопроса и подсчётом правильных ответов, мне потребовалась возможность параллельного выполнения функции, в которой реализована игровая сессия, у нескольких пользователей. Для решения этой проблемы я использовал модуль threading, позволяющий реализовать многопоточность в Python. Различные задачи внутри потоков выполняются на одном ядре, а операционная система управляет тем, когда программа работает с каким потоком [10]. Поэтому это своего рода иллюзия параллельной работы. Но в нашем случае задачи не требуют значительных вычислений от процессора, поэтому многопоточности вполне достаточно для бесперебойной работы.

2 Реализация

2.1 Анализ существующих решений

Мне удалось найти два чат-бота со схожей функциональностью.

Во-первых, это бот с никнеймом @chgk_with_21_bot. Он подгружает с сайта базы вопросов не пакет случайных вопросов, а пакет случайного турнира. И по одному присылает вопросы и ответы на них. На мой взгляд, это не очень удачное решение, ведь пользователь не знает, сколько вопросов было в турнире: может быть 24, а может быть и целых 90. А возможности задать ограничение бот не предоставляет. И работает он исключительно с помощью команд, начинающихся со знака «/». После использования этого бота я решил, что в моём чат-боте обязательно должны быть кнопки, потому что взаимодействие с помощью команд совсем не интуитивно, и каждый раз нужно открывать полный список команд и задумываться, какая именно тебе сейчас нужна.

Во-вторых, я рассмотрел бот с никнеймом @chgk_bot. В его новостном канале около 30 000 подписчиков, т. е. скорее всего это самое популярное из доступных решений. Он умеет подгружать случайные вопросы по одному; умеет принимать ответы и считать статистику за всю историю беседы с пользователем; умеет запускать таймер, но отдельной командой, за которой, опять же, нужно отправляться в достаточно длинный список команд.

Я пришёл к выводу, что идея моего чат-бота не нова. Однако ни один из рассмотренных чат-ботов не предоставляет функционала игровой сессии, в которой пользователь мог бы выбрать количество и уровень вопросов,

отыграть их от начала до конца и получить результат игры. Это и легло в основу моего решения.

2.2 Определение функционала

В результате анализа существующих аналогов я определил набор опций, которые хочу реализовать в своём чат-боте:

– Выбор количества вопросов в игре с помощью предложенных кнопок или любого числа, введённого клавиатуры [8].

– Выбор уровня сложности вопросов в игре с помощью предложенных кнопок [8].

– Предоставление не только текста вопроса, но и раздаточного материала (иллюстрации [7] или текста), если таковой имеется.

– Таймер, отсчитывающий минуту на обдумывание / обсуждение каждого вопроса, запускающийся по кнопке под текстом вопроса (все вопросы разной длины, поэтому нужно дать пользователю возможность прочитать вопрос так, чтобы время чтения не отнимало драгоценные секунды на обдумывание).

– Принятие ответа от пользователя.

– Автоматическая отправка правильного ответа на вопрос, а также авторского комментария, если таковой имеется.

– Переход к следующему вопросу по кнопке (чтобы было время прочитать ответ и комментарий предыдущего вопроса).

– Подсчёт правильных ответов в игре.

– Досрочное завершение игры.

2.3 Архитектура программы

Получившаяся программа (полный код доступен в моём репозитории GitHub https://github.com/YuriiShuginin/MrHookBot) имеет следующую структуру:

– Подключение сторонних модулей. В первой главе я подробно рассказал про использование библиотек requests, PyTelegramBotAPI, BeautifulSoup и threading. Помимо них я использовал модуль time, для остановки выполнения программы на время ожидания ответа от пользователя, и модуль re для работы с регулярными выражениями при выборе нужной информации из строковых переменных [3].

– Глобальные переменные. Их всего две. bot – объект класса telebot.TeleBot, отвечающий за обращение к Telegram Bot API. И словарь received_msg, необходимый для параллельной работы нескольких пользователей. В нём в качестве ключей используются уникальные id чатов с пользователями, установившими соединение с ботом, а в качестве значения – список, в котором на первом месте хранится статус общения с пользователем в виде строки (например, ‘wait_answer’ или ‘finish’), а на втором – текст последнего сообщения, полученного от пользователя. Таким образом, когда бот ожидает ответ от пользователя, он проверяет статус общения и берёт значение текста последнего сообщения, когда статус меняется на нужный.

– Функции.

get_page(question_amount: int, complexity: str) - > str: принимает количество вопросов и их сложность, возвращает текст кода полученной страницы со случайным пакетом вопросов с заданными параметрами.

parse_questions(web_page: str) -> tuple[list[str], list[str], list[str], list[str]]: принимает текст кода страницы и возвращает 4 списка необходимых для

игры данных: тексты вопросов, текстовые раздатки, раздатки-картинки, ответы.

get_start(message: telebot.types.Message) -> None: обрабатывает команду /start для начала новой игры. Создаёт и запускает поток с функцией game(). Работает с декоратором bot.message_handler(commands = [‘start’]).

get_finish(message: telebot.types.Message) -> None: обрабатывает команду /finish для досрочного завершения текущей игры. Работает с декоратором bot.message_handler(commands = [‘finish’]).

read_answer(message: telebot.types.Message) -> None: обрабатывает любое текстовое сообщение от пользователя. Если сообщение ожидается, то меняет значение в упомянутом словаре received_msg[message.chat.id], иначе отправляет сообщение о неуместности отправленного сообщения. Работает с декоратором bot.message_handler(content_types = [‘text’]).

inline_keyboard_msg(button_text: str, msg_text: str, chat_id: int) -> None: формирует и отправляет сообщение, в котором необходима кнопка. Принимает текст на кнопке, текст сообщения и id чата, в который нужно отправить это сообщение.

handle_callback(call: telebot.types.CallbackQuery) -> None: обрабатывает нажатие на кнопку в сообщении [9]. Работает с декоратором bot.callback_query_handler(lambda call: call.data in [‘next_question’, ‘start_timer’]), в качестве параметра которого указывается функция, проверяющая, что по нажатию на кнопку получено одно из известных значений.

game(chat_id: int) -> None: функция игры, работающая в потоке, созданном при обработке команды /start. Принимает один параметр – значение id чата, в котором было запрошено начало игры.

3 Полученный результат

3.1 Примеры работы

– Начало игры и выбор количества вопросов (рисунок 4).

Рисунок 4 – Начало игры

– Выбор уровня сложности пакета вопросов (рисунок 5).

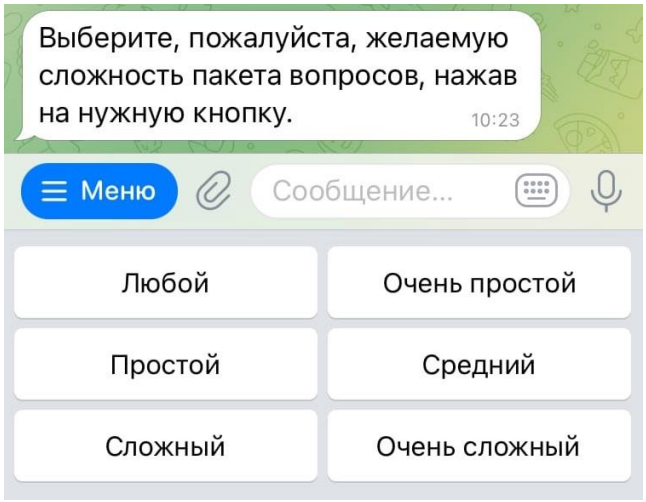

Рисунок 5 – Выбор сложности вопросов в игре

– Вопрос с текстовым раздаточным материалом и запущенным по кнопке таймером на обдумывание (рисунок 6).

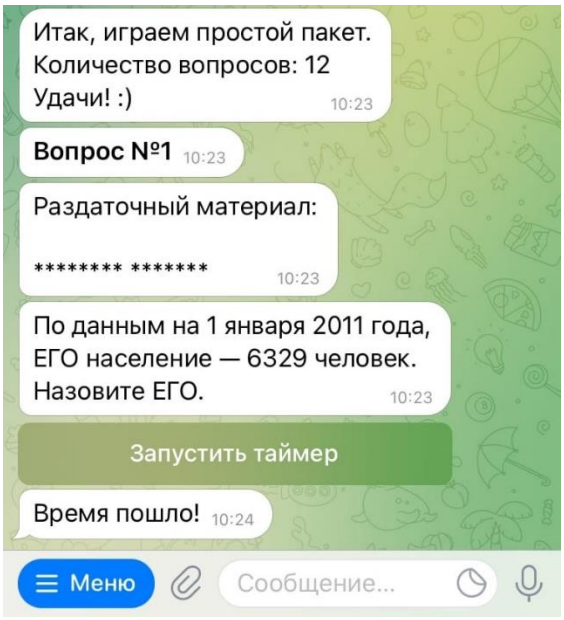

Рисунок 6 – Вопрос с запущенным таймером

– Принятие и зачёт ответа пользователя (по кнопкам Да / Нет), переход к следующему вопросу по кнопке (рисунок 7).

Рисунок 7 – Ответ на вопрос и переход к следующему вопросу

– Окончание игры, состоящей из двух вопросов (рисунок 8).

Рисунок 8 – Окончание игры

3.2 Сценарии применения

– С помощью чат бота можно сыграть в «Что? Где? Когда?» с самим собой в любом месте и в любое время, имея при себе смартфон и доступ в интернет. Не нужен человек, который будет читать вопросы (ведущий). Не нужны блокнот и ручка, чтобы записывать вопрос. Всё необходимое для игры доступно в реализованном чат-боте.

– Можно использовать чат-бот для проведения быстрой игры в компании друзей. Например, я играю в «Что? Где? Когда?» в роли капитана, и мне часто требуется прочитать команде несколько разминочных вопросов, когда до начала турнира остаётся 10-15 минут. Без этого бота мне приходилось пользоваться сайтом базы вопросов со смартфона, что уже доставляет определённые неудобства. Помимо этого, всегда трудно отмерить минуту на обсуждение. На всё это тратится лишнее время. Но разработанный бот решает упомянутые сложности.

– Проведение турнира в роли ведущего (если добавить функцию загрузки своего пакета вопросов). Обычно ведущие турниров читают вопросы с листа, потом на каком-то устройстве запускают таймер. Потом им

нужно не забыть предупредить, что осталось 10 секунд. И вовремя дать команду сдавать ответы. Разработанный чат-бот сильно упростит этот алгоритм, т. к. сочетает в себе автоматизацию всех необходимых действий в нужной последовательности.

3.3 Пути расширения

– Загрузка программы в сеть. Для того, чтобы бот был доступен всё-время, нужно загрузить его на какой-то сервер. Сейчас для работы бота его необходимо запустить на компьютере.

– Добавление функции загрузки уникальных пакетов вопросов и функции оплаты игры на уникальных вопросах, которых нет в базе (вопросы – интеллектуальная собственность авторов, поэтому игра на вопросах, которые ещё не были засвечены на больших турнирах, обычно предполагает небольшой денежный взнос).

– Разработка версий чат-бота для других платформ (например, для ВКонтакте).

– Добавление функции выбора других игровых дисциплин (на сайте Базы есть вопросы не только для «Что? Где? Когда?», но и для других интеллектуальных игр: Брейн-ринга, Своей игры, Эрудит-квартета и т. д.).

– Автоматизация проверки ответа на вопрос, данного пользователем.

– Перевод вопросов на другие языки (или загрузка вопросов с иностранных сайтов Базы вопросов, в случае существования таковых).

ЗАКЛЮЧЕНИЕ

В ходе практики были получены практические навыки создания чат-ботов на платформе Telegram средствами языка программирования Python и взаимодействия с Telegram Bot API, а также синтаксического анализа кода веб-страницы. Я освоил работу с модулями requests, BeautifulSoup, PyTelegramBotAPI и threading, которые являются важными инструментами для решения задач в своих областях.

В разработанном чат-боте были реализованы все задуманные функции за исключением автоматической проверки ответа пользователя на вопрос. Дело в том, что на вопрос зачастую может быть много синонимичных ответов. А в поле «Ответ» на сайте Базы вопросов указывается только один из них, например, с фразой «И другие синонимичные ответы». Поэтому простая проверка вхождения строки с ответом пользователя в строку с правильным ответом не дала бы высокой точности. А за создание каких-то сложных алгоритмов, которые могли бы засчитывать ответ по смыслу, я не осмелился браться из-за достаточно короткого срока учебной практики.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация модуля requests на английском языке [Электронный ресурс]. URL: https://docs.python-requests.org/en/latest/ (дата обращения: 08.02.2022).

2. Документация BeautifulSoup на русском языке [Электронный ресурс]. URL: https://www.crummy.com/software/BeautifulSoup/bs4/doc.ru/ (дата обращения: 08.02.2022).

3. Официальная документация Python 3.8 [Электронный ресурс]. URL: https://docs.python.org/3.8/ (дата обращения: 09.02.2022).

4. Официальная документация по работе с Telegram Bot API. [Электронный ресурс]. URL: https://core.telegram.org/bots/api (дата обращения: 11.02.2022).

5. Лабораторная работа преподавателя ФИКТ ИТМО Сорокина Д. С. «Бот для мессенджера Telegram» [Электронный ресурс]. URL: https://dementiy.github.io/assignments/telegram/ (дата обращения: 11.02.2022).

6. «Как написать простого бота для ВК и Телеграм», статья на Хабре [Электронный ресурс]. URL: https://habr.com/ru/company/ruvds/blog/542066/ (дата обращения: 11.02.2022).

7. Статья «Учим Telegram-бота отправлять картинки пользователю» [Электронный ресурс]. URL: https://zen.yandex.ru/media/id/5d947dd28d5b5f00b14d62d6/11-uchim-telegram-bota-otpravliat-kartinki-polzovateliu-5dc73e03b18b3938187fbcd0 (дата обращения: 13.02.2022).

8. Цикл статей «Пишем Telegram-ботов на Python (v2)», раздел «Кнопки» [Электронный ресурс]. URL: https://mastergroosha.github.io/telegram-tutorial-2/buttons/ (дата обращения: 15.02.2022).

9. Англоязычная статья о настройке обработки нажатия на кнопку, прикреплённую к сообщению [Электронный ресурс]. URL: https://github.com/eternnoir/pyTelegramBotAPI/issues/563 (дата обращения: 15.02.2022).

10. Статья «Введение в потоки в Python» [Электронный ресурс]. URL: https://webdevblog.ru/vvedenie-v-potoki-v-python/ (дата обращения: 17.02.2022).

11. Статья на Хабре «Всё, о чём должен знать разработчик Телеграм-ботов» [Электронный ресурс]. URL: https://habr.com/ru/post/543676/ (дата обращения: 17.02.2022).