**ЛАБОРАТОРНАЯ РАБОТА №1-2. Создание чат-бота Telegram на Python.**

* 1. Цель и содержание

Цель лабораторной работы: овладение технологиями создания и применения чат-ботов.

Задачи лабораторной работы:

* Определить программную платформу и интерфейс прикладных решений и библиотек API с множеством представляемых элементов, наиболее приемлемых для реализации современных чат-ботов;
* Изучить технологию разработки чат-бота на демонстрационном примере;
* Выполнить индивидуальное задание по разработке чат-бота на Python.
  1. Формируемые компетенции

Лабораторная работа направлена на формирование компетенци ОПК-2 - Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности.

* 1. Теоретическая часть.

В мире информационных технологий мы постоянно находим новые способы сделать нашу жизнь проще и эффективнее. Одним из выдающихся достижений является появление чат-ботов.

Чат-бот (от англ. – chatterbot) – это компьютерная программа, которая ведет разговор с помощью слуховых или текстовых методов. Чат-боты, или виртуальные собеседники, используются в диалоговых системах для различных практических целей, включая обслуживание клиентов или сбор информации. Чат-боты являются универсальными средствами, способными к решению разнообразных задач – от общения до развлечений, от предоставления медицинской консультации до заказа товаров и услуг посредством специализированных прикладных решений, от распознавания эмоций до решения сложных консалтинговых задач в службах поддержки клиентоориентированных информационных систем.

В независимости от платформы, чат-бот – это прикладная программа, которая, получая информацию от пользователя, формирует корректные, логически обоснованные ответы. Правовые последствия таких ответов отличают простые боты с «чатиками» для развлечений от серьезных сервисов. Если чат-бот позиционируется как некий виртуальный представитель компании, то и чат с таким субъектом может создавать для компании и для пользователя определенные обязательства.

Существует большое количество разнообразных чат-ботов. Некоторые из них используются для заказа товаров в интернет-магазинах, другие дают вам медицинские советы, помогают записаться на прием и даже следят за вашим самочувствием. Образование – еще одна сфера, в которую вступают чат-боты. Они создаются для того, чтобы ответить на ваши вопросы, объяснить сложные понятия и даже помочь с домашним заданием. Известны применения чат-ботов в качестве медицинских помощников, которые помогают записаться на прием и даже следят за самочувствием пациентов, банковские чат-боты помогают разобраться в личных банковских вопросах клиента, помочь с транзакциями и дать совет по разумным деньгам. Компании используют чат-ботов для обеспечения первоклассного обслуживания клиентов. Эти цифровые помощники решают распространенные вопросы, оставляя людям больше времени для решения сложных вопросов и общения с клиентами на личном уровне.

1.4. Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими характеристиками: 64-разрядный (x64) процессор с тактовой частотой 1 ГГц и выше, оперативная память – 1 Гб и выше, свободное дисковое пространство – не менее 1 Гб, графическое устройство DirectX 9.

Программное обеспечение: операционная система Windows 7 и выше, мессенджер Telegram.

1.5. Указания по технике безопасности

Студенты должны следовать общепринятой технике безопасности для пользователей персональных компьютеров. Не следует самостоятельно производить ремонт технических средств, установку и удаление программного обеспечения. В случае обнаружения неисправностей необходимо сообщить об этом администратору компьютерного класса (обслуживающему персоналу лаборатории).

1.6. Методика и порядок выполнения работы

1.6.1. Что­бы соз­дать бота, нам нуж­но дать ему наз­вание, адрес и получить токен — стро­ку, которая будет однознач­но иден­тифици­ровать нашего бота для сер­веров Telegram. Зай­дем в Telegram под сво­им акка­унтом и откро­ем «отца всех ботов», **[BotFather](https://t.me/BotFather)**.

Жмем кноп­ку «Запус­тить» (или отпра­вим /start), в ответ BotFather приш­лет нам спи­сок дос­тупных команд:

* /newbot — соз­дать нового бота;
* /mybots — редак­тировать ваших ботов;
* /setname — сме­нить имя бота;
* /setdescription — изме­нить опи­сание бота;
* /setabouttext — изме­нить информа­цию о боте;
* /setuserpic — изме­нить фото ава­тар­ки бота;
* /setcommands — изме­нить спи­сок команд бота;
* /deletebot — уда­лить бота.

От­пра­вим бате‑боту коман­ду /newbot, что­бы соз­дать нового бота. В ответ он поп­росит ввес­ти имя будуще­го бота, его мож­но писать на рус­ском. Пос­ле вво­да име­ни нуж­но будет отпра­вить адрес бота, при­чем он дол­жен закан­чивать­ся на сло­во bot. Нап­ример, xbot или xp\_bot. Если адрес будет уже кем‑то занят, BotFather нач­нет изви­нять­ся и про­сить при­думать что‑нибудь дру­гое.

Ког­да мы наконец най­дем сво­бод­ный и кра­сивый адрес для нашего бота, в ответ получим сооб­щение, в котором пос­ле фра­зы Use this token to access the HTTP API будет написа­на стро­ка из букв и цифр — это и есть необ­ходимый нам токен. Сох­раним ее где‑нибудь на сво­ем компь­юте­ре, что­бы потом исполь­зовать в скрип­те бота.

Для вза­имо­дей­ствия с Telegram API есть нес­коль­ко готовых модулей. Самый прос­той из них — **Telebot**. Что­бы уста­новить его, набери pip install pytelegrambotapi.

В Linux, воз­можно, понадо­бит­ся написать pip3 вмес­то pip, что­бы ука­зать, что мы хотим работать с треть­ей вер­сией Python.

1.6.2. **ЭХО-БОТ.**

Для начала реали­зуем так называ­емо­го эхо‑бота. Он будет получать от поль­зовате­ля тек­сто­вое сооб­щение и воз­вра­щать его.

import telebot

# Создаем экземпляр бота

bot = telebot.TeleBot('Здесь впиши токен, полученный от @botfather')

# Функция, обрабатывающая команду /start

@bot.message\_handler(commands=["start"])

def start(m, res=False):

bot.send\_message(m.chat.id, 'Я на связи. Напиши мне что-нибудь )')

# Получение сообщений от юзера

@bot.message\_handler(content\_types=["text"])

def handle\_text(message):

bot.send\_message(message.chat.id, 'Вы написали: ' + message.text)

# Запускаем бота

bot.polling(none\_stop=True, interval=0)

За­пус­кай скрипт и ищи в поис­ке Telegram сво­его бота по адре­су, который ты при­думал ранее. Запус­каем бота кноп­кой «Запус­тить» (Start) или коман­дой /start и можем убе­дить­ся в том, что он работа­ет и воз­вра­щает сооб­щения.

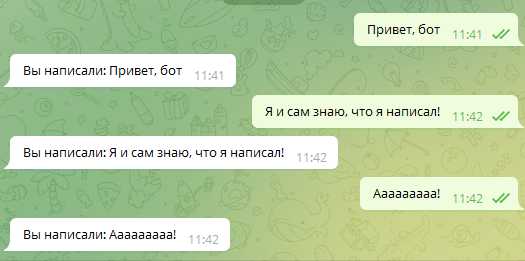
[](https://static.xakep.ru/images/6d98435bc099a3831e97e5cc45297cb4/23822/1.png)

Рисунок 1 – Пример работы эхо-бота

1.6.3. WIKIPEDIA-БОТ

Да­вай научим нашего бота не прос­то отсы­лать сооб­щения обратно, а чему‑нибудь поин­терес­нее. Нап­ример, по вве­ден­ному сло­ву давать статью на Википе­дии. Здесь нам поможет модуль **Wikipedia**:

pip install wikipedia

Го­товим код.

import telebot, wikipedia, re

# Создаем экземпляр бота

bot = telebot.TeleBot('Здесь впиши токен, полученный от @botfather')

# Устанавливаем русский язык в Wikipedia

wikipedia.set\_lang("ru")

# Чистим текст статьи в Wikipedia и ограничиваем его тысячей символов

def getwiki(s):

try:

ny = wikipedia.page(s)

# Получаем первую тысячу символов

wikitext=ny.content[:1000]

# Разделяем по точкам

wikimas=wikitext.split('.')

# Отбрасываем всЕ после последней точки

wikimas = wikimas[:-1]

# Создаем пустую переменную для текста

wikitext2 = ''

# Проходимся по строкам, где нет знаков «равно» (то есть все, кроме заголовков)

for x in wikimas:

if not('==' in x):

# Если в строке осталось больше трех символов, добавляем ее к нашей переменной и возвращаем утерянные при разделении строк точки на место

if(len((x.strip()))>3):

wikitext2=wikitext2+x+'.'

else:

break

# Теперь при помощи регулярных выражений убираем разметку

wikitext2=re.sub('\([^()]\*\)', '', wikitext2)

wikitext2=re.sub('\([^()]\*\)', '', wikitext2)

wikitext2=re.sub('\{[^\{\}]\*\}', '', wikitext2)

# Возвращаем текстовую строку

return wikitext2

# Обрабатываем исключение, которое мог вернуть модуль wikipedia при запросе

except Exception as e:

return 'В энциклопедии нет информации об этом'

# Функция, обрабатывающая команду /start

@bot.message\_handler(commands=["start"])

def start(m, res=False):

bot.send\_message(m.chat.id, 'Отправьте мне любое слово, и я найду его значение на Wikipedia')

# Получение сообщений от юзера

@bot.message\_handler(content\_types=["text"])

def handle\_text(message):

bot.send\_message(message.chat.id, getwiki(message.text))

# Запускаем бота

bot.polling(none\_stop=True, interval=0)

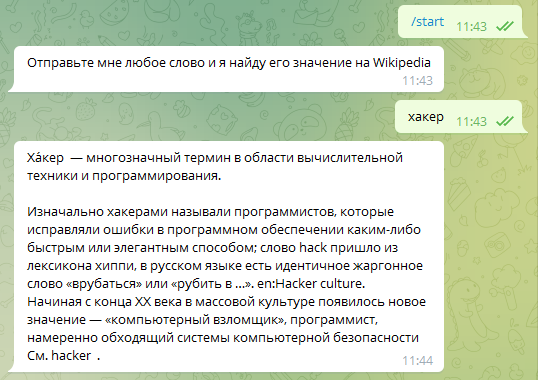
[[](https://static.xakep.ru/images/6d98435bc099a3831e97e5cc45297cb4/23820/2.png)](https://static.xakep.ru/images/6d98435bc099a3831e97e5cc45297cb4/23820/2.png)

Рисунок 2 – Пример работы WIKIPEDIA-бота

1.7. Практическое задание: создать чат бота с функционалом индивидуального задания. Студент имеет право по согласованию с преподавателем предложить свой вариант чат-бота.

Варианты индивидуальных заданий.

1. Игра «Угадай слово»

Инструкции: Создайте классическую игру «Угадай слово». Предложите пользователю угадывать слово по буквам до тех пор, пока он не угадает его правильно или пока не закончатся попытки.

Пояснения к коду: Игра «Угадай слово» начинается со случайного выбора слова из списка. Затем игроку предлагается ввести букву. Если буква есть в слове, то она раскрывается, в противном случае игрок теряет попытку. Игра продолжается до тех пор, пока слово не будет угадано или у игрока не закончатся попытки.

2. Калькулятор

Инструкции: Создайте простой калькулятор, выполняющий основные арифметические операции (+, -, \*, /) над двумя числами.

Пояснения к коду: Программа-калькулятор принимает на вход два числа и оператор. В зависимости от введенного оператора программа выполняет соответствующую арифметическую операцию и выводит результат на экран.

3. Игра «Угадайте число»

Инструкции: Создайте игру, в которой компьютер генерирует случайное число, а игрок пытается угадать его за определенное количество попыток.

Пояснение к коду: В этой игре компьютер генерирует случайное число от 1 до 100. Затем игроку предлагается угадать это число. После каждого угадывания программа выдает сообщение о том, было ли угадываемое число слишком большим или слишком маленьким. Игра продолжается до тех пор, пока игрок не угадает число правильно.

4. Список дел

Инструкции: Создать приложение для составления списка дел, позволяющее пользователю добавлять, просматривать и удалять задачи.

Пояснения к коду: Программа «Список дел» позволяет пользователю добавлять задачи, просматривать существующие задачи, удалять конкретные задачи и выходить из программы. Задачи хранятся в списке, и выбор пользователя обрабатывается соответствующим образом.

5. Симулятор игры в кости

Инструкции: Имитация бросания игральных костей. Дайте пользователю возможность выбрать количество игральных костей и выведите на экран результаты.

Пояснения к коду: Программа-симулятор игры в кости предлагает пользователю ввести количество игральных костей, которые он хочет бросить. Затем она генерирует случайные числа от 1 до 6 для каждой кости и выводит результаты на экран.

6. Счетчик слов

Инструкции: Создайте программу, которая подсчитывает количество слов в заданном тексте.

Пояснения к коду: Программа счетчика слов принимает введенный пользователем текст и разбивает его на отдельные слова с помощью функции split(). Затем она подсчитывает количество слов и выводит результат на экран.

1.8. Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы;
2. Задачи лабораторной работы.
3. Ответы на контрольные вопросы.
4. Программный код чат-бота согласно индивидуальному заданию (с комментариями).

Отчет о выполнении лабораторной работы, подписанный студентом, сдается преподавателю.

1.9. Контрольные вопросы

1. Перечислите сферы применения чат-ботов.
2. Охарактеризуйте основные плюсы и минусы использования чат ботов.
3. Какая библиотека используется в python для создания ботов Телеграмм?
4. Какое условие необходимо для работоспособности бота после запуска?
5. Где хранится код бота?

1.9. Список литературы

Для выполнения лабораторной работы, при подготовке к защите, а также для ответа на контрольные вопросы рекомендуется использовать следующие источники:

1. Сузи, Р. А. Язык программирования Python : учебное пособие / Р. А. Сузи. — 3-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. — 350 c. — ISBN 978-5-4497-0705-5. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/97589.html>.
2. Косицин, Д. Ю. Язык программирования Python : учебно-методическое пособие / Д. Ю. Косицин. — Минск : БГУ, 2019. — 136 с. — ISBN 978-985-566-746-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/180546>.