МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский Авиационный Институт»
(Национальный Исследовательский Университет)

Институт: №8 «Информационные технологии и прикладная
математика»
Кафедра: 806 «Вычислительная математика и программирование»

Реферат
по курсу «Фундаментальная
информатика» I семестр
Тема:
«Язык программирования Python на примере разработки
TelegramBot - переводчик »

Группа:	М8О-109Б-22
Студент:	Серякова А.А.
Преподаватель:	Сысоев М.А.
Оценка:	
Дата:	

Москва, 2022

Содержание

1. История Python и сферы применения

В 1989 г. Гвидо Ван Россум создал новый язык программирования под названием Python, а в 1991 г выпустил его. Главная цель, которую ставил перед собой автор — это упрощение процесса программирования. Чтобы писать код было проще, он должен стать более читабельным и понятным для человека. У Python открытый исходный код. Одно из преимуществ этого языка — возможность запускать программы на нём как на ОС Windows, так и на macOS и Linux.

Где применяется сейчас?

Python сегодня входит в число наиболее популярных языков программирования и является языком программирования высокого уровня общего назначения.

Python — это скриптовый язык программирования, который используется во многих областях, начиная от IOS и Android и заканчивая серверными OS.

Вот три основные области его применения:

- Веб-разработка
- Машинное обучение
- Автоматизация процессов

Веб-разработка

Python используется в Back-End разработке и имеет два основных фреймворка: Django и Flask. Они облегчают процесс написания кода для серверной части приложений.

Зачем нужен фреймворк?

Фреймворки позволяют легко и быстро создать базовую логику Back-End стороны. Back-End включает в себя сопоставление разных URL-адресов с частями Python-кода, работу с базами данных, создание HTML-представлений для отображения на устройствах пользователей.

Машинное обучение

Машинное обучение — это наука о том, как заставить ИИ (Искусственный Интеллект) учиться и действовать, как человек, и так, чтобы он сам постоянно улучшался и развивался на основе предоставленных нами данных о реальном мире.

Автоматизация процессов

Одна из самых популярных сфер применения Python — это написание небольших скриптов для автоматизации различных рабочих операций и процессов.
Например, нужно перезаписать данные с Word в Excel-файл. Это можно сделать вручную, но когда таких файлов очень много, на помощь может прийти написание скрипта, который сделает это самостоятельно и быстро.

Есть несколько причин применения Python для задач автоматизации:

- простой синтаксис, позволяющий быстро писать сценарии;
- лёгкость отладки, связанная с тем, что код не компилируется перед запуском

2. Основные особенности языка

1. Доступность

Основной особенностью данного языка программирования является то, что его достаточно просто понять и изучить.

2. Объектно-ориентированный

Python обладает *всеми* функциями объектно-ориентированного языка, такими как наследование, переопределение методов, объекты и т. д. Таким образом, он поддерживает все парадигмы и имеет соответствующие функции в своих библиотеках.

3. Надежные стандартные библиотеки

Библиотеки python очень обширны и включают в себя различные модули и функции, которые поддерживают различные операции, работающие с различными типами данных, такими как регулярные выражения и т. д.

4. Поддерживает различные парадигмы программирования

Благодаря поддержке всех возможностей объектно-ориентированного языка, Python также поддерживает процедурно-ориентированную парадигму. Он также поддерживает множественное наследование. Это все возможно благодаря большим и надежным библиотекам, которые содержат функции для всего.

5. Поддержка интерактивного режима

Python также поддерживает работу в интерактивном режиме, где можно легко отлаживать код и тестировать его построчно. Это помогает максимально уменьшить количество ошибок.

6. Автоматическая сборка мусора

Python также запускает автоматическую сборку мусора для отличного управления памятью и производительностью. Благодаря этому память может использоваться по максимуму, что делает приложение более устойчивым.

7. Динамически набираемый и проверка типа

Это одна из замечательных особенностей Python, заключающаяся в том, что нет необходимости объявлять тип данных переменной перед ее использованием. Как только значение присваивается переменной, определяется ее тип данных. Таким образом, проверка типа в python выполняется во время выполнения, в отличие от других языков программирования.

8. Базы данных

База данных приложения является одной из важнейших частей, которая также должна поддерживаться соответствующим используемым языком программирования. Python поддерживает все основные базы данных, которые могут использоваться в приложении, такие как MYSQL, ORACLE и т. Д. Соответствующие функции для этих операций с базами данных уже определены в библиотеках python. нужно использовать эти файлы в коде, чтобы использовать его.

9. Программирование GUI

Python, являющийся языком сценариев, также поддерживает множество функций и библиотек, которые позволяют графически разрабатывать приложения. В огромных библиотеках и функциях определяются соответствующие системные вызовы и процедуры для вызова определенных вызовов ОС для разработки идеального графического интерфейса приложения. Python также нуждается в фреймворке для создания такого GUI. Примерами некоторых фреймворков являются Django, Tkinter и т. Д.

10. Расширяемый

Эта функция делает возможным использование других языков в коде Python. Это означает, что код Python может быть расширен и на другие языки, поэтому он может быть легко встроен в существующий код, чтобы сделать

его более надежным и расширить его возможности. Другие языки могут быть использованы для компиляции нашего кода Python.

11. Портативный

Говорят, что язык программирования является переносимым, если он позволяет нам кодировать один раз и запускать любую функцию. Значит, платформа, где она была закодирована и где она будет работать, не обязательно должна быть одинаковой. Эта функция позволяет использовать одну из наиболее ценных функций объектно-ориентированных языков - возможность повторного использования. Как разработчику, нужно кодировать решение и генерировать его байт-код, и не нужно беспокоиться о среде, в которой оно будет запускаться. Например, один может запустить код, разработанный в операционной системе Windows, в любой другой операционной системе, такой как -Linux Unix и т. Д.

12. Масштабируемый

Этот язык помогает разрабатывать различные системы или приложения, способные обрабатывать динамически растущий объем работы. Эти типы приложений очень помогают в росте организации, поскольку они достаточно сильны, чтобы справиться с изменениями до некоторой степени.

13. Бесплатный и открытый исходный код

Да, вы правильно прочитали, вам не нужно платить ни копейки, чтобы использовать этот язык в вашем приложении. Нужно просто скачать его с официального сайта, и все готово для запуска. И так как он с открытым исходным кодом, его исходный код также был обнародован. Можно легко загрузить и использовать его по мере необходимости, а также поделиться им с другими. Таким образом это улучшается каждый день.

14. Интегрированный

Python может быть легко интегрирован с другими доступными языками программирования, такими как C, C ++, Java и т. Д. Это позволяет каждому использовать его для улучшения функциональности существующих приложений и повышения его надежности.

3. Разработка ТГ-бота для изучения английского (переводчик)

Цель работы:

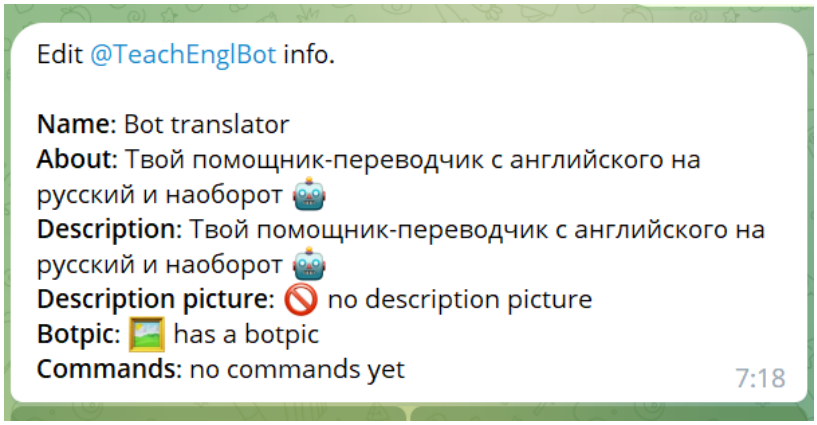
Создать ТГ- бота, который сможет переводить предложения с английского на русский и наоборот.

Задачи:

- Выбрать необходимые библиотеки
- Изучить библиотеку telebot (pyTelegramBotAPI)
- Проработать визуальную часть проекта (название кнопок, текст, последовательность каждой кнопки, команды)
- Подключить Гугл- переводчик
- Написать программу для Телеграм бота

Для начала необходимо воспользоваться «отцом» всех Telegram ботов – BotFather.

Регистрация начинается с команды «/newbot», после чего предлагается ввести название чат-бота с обязательным условием: в конце названия указывается «Bot» или «_bot». После введения уникального имени «BotFather» выдает уникальный токен (специальный набор символов для доступа к HTTP API Telegram Bot). Далее заполняется профиль бота. Добавляется описание, фото и имя:



Далее можно приступить к программной части создания чат-бота. Для реализации своего ТГ-бота воспользуюсь библиотекой telebot (pyTelegramBotAPI) - библиотека для создания бота на python для телеграмма. Так же скачаю необходимый модуль, чтобы подключаться к гугл переводчику:

```
PS C:\Users\Мой ноутбук\PycharmProjects\BotTelegram> pip install googletrans==3.1.0a0
```

Начало работы с чат-ботом начинается с команды «/start». Данная команда активирует бота, начинается диалог с пользователем, код реализации представлен на листинге 1.

Листинг 1

```python
from googletrans import Translator
import telebot
from telebot import types

bot = telebot.TeleBot('5**********************0')

translator = Translator()
@bot.message_handler(commands=['start'])
def start(message): #чтобы отслеживать команды /start
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    btn1 = types.KeyboardButton("Переведи текст")
    markup.add(btn1)
    mess = f'Hello, {message.from_user.first_name} ! Я переводчик с английского на русский и наоборот :)'
    bot.send_message(message.chat.id, mess, reply_markup=markup)
```

Создаём кнопку «Переведи текст», после этой команды пользователь может выбирать с какого языка он хочет перевести текст (я буду использовать 2 языка: английский и русский). Здесь же создаем кнопку «отмена», если выбрали неправильный язык для перевода, после чего пользователь вновь возвращается к главному меню. Если пользователь пишем что-то непохожее на текст, бот также возвращает его в главное меню, код реализации представлен на листинге 2.

Листинг 2

```python
@bot.message_handler(content_types=['text'])
def func(message):

    if (message.text == "Отмена"):
        markup = telebot.types.ReplyKeyboardMarkup(resize_keyboard=True)
        btn2 = types.KeyboardButton("Переведи текст")
        markup.add(btn2)
        mess = f'Вы вернулись к главному меню'
        bot.send_message(message.chat.id, mess, parse_mode='html', reply_markup=markup)
    elif (message.text == "Переведи текст"):
        markup = telebot.types.InlineKeyboardMarkup()
        markup.add(telebot.types.InlineKeyboardButton(text='RU', callback_data=1))
        markup.add(telebot.types.InlineKeyboardButton(text='EN', callback_data=2))

        bot.send_message(message.chat.id, "Выбери язык, на который хочешь перевести текст.", parse_mode='html', reply_markup=markup)
    else:
        markup = telebot.types.ReplyKeyboardMarkup(resize_keyboard=True)
        btn2 = types.KeyboardButton("Переведи текст")
        markup.add(btn2)
        mess = f'Не понял тебя(... Жми на кнопку или напиши: "Переведи текст".'
        bot.send_message(message.chat.id, mess, parse_mode='html', reply_markup=markup)
```

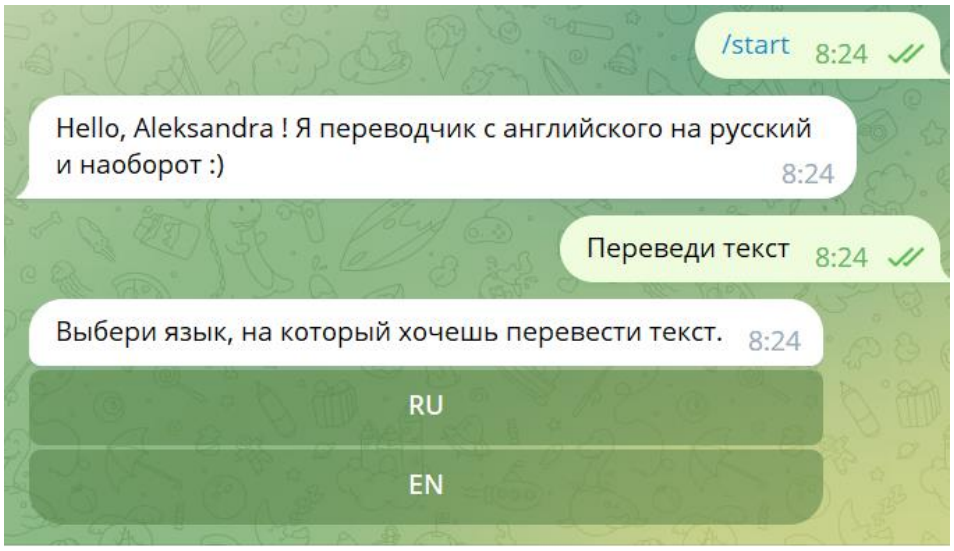
Далее следует реализация получения текста, который прислал пользователь и обращение к Google переводчику.

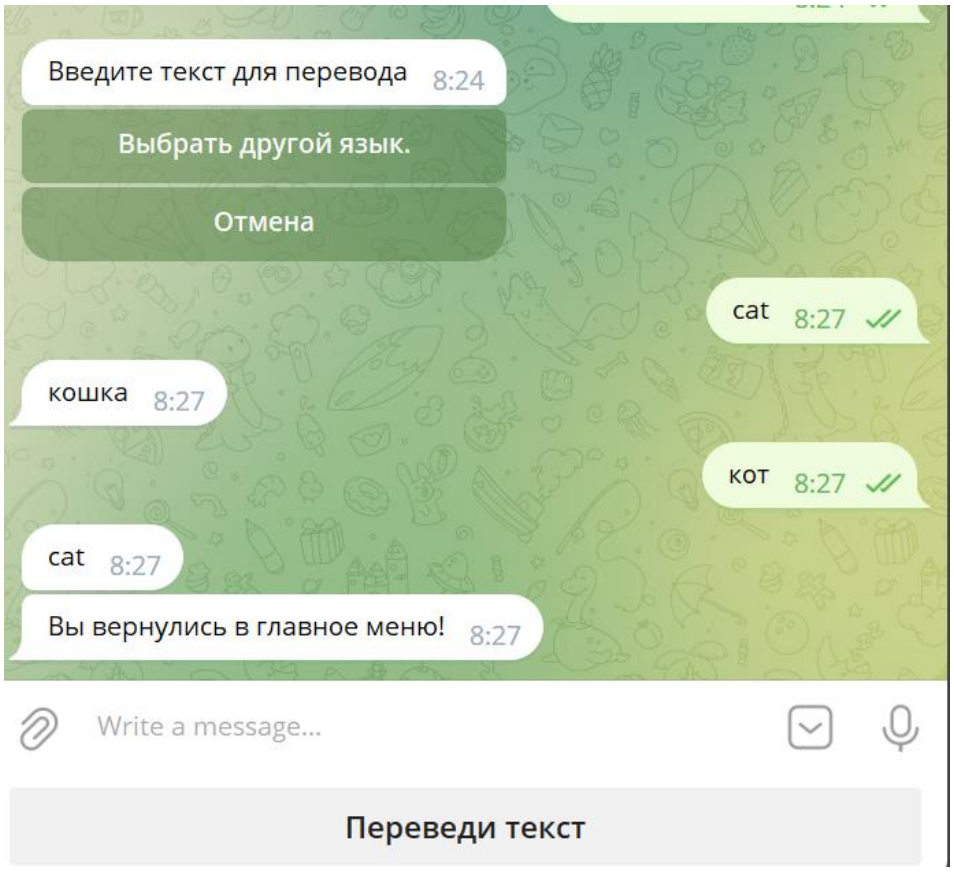
Код всей программы:

```
from googletrans import Translator
import telebot
from telebot import types

bot = telebot.TeleBot('*****************************')

translator = Translator()
@bot.message_handler(commands=['start'])
def start(message): #чтобы отслеживать команды /start
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    btn1 = types.KeyboardButton("Переведи текст")
    markup.add(btn1)
    mess = f'Hello, {message.from_user.first_name} ! Я переводчик с
английского на русский и наоборот :)'
    bot.send_message(message.chat.id, mess, reply_markup=markup)

@bot.message_handler(content_types=['text'])
def func(message):

    if (message.text == "Отмена"):
        markup = telebot.types.ReplyKeyboardMarkup(resize_keyboard=True)
        btn2 = types.KeyboardButton("Переведи текст")
        markup.add(btn2)
        mess = f'Вы вернулись к главному меню'
        bot.send_message(message.chat.id, mess, parse_mode='html',
reply_markup=markup)
    elif (message.text == "Переведи текст"):
        markup = telebot.types.InlineKeyboardMarkup()
        markup.add(telebot.types.InlineKeyboardButton(text='RU',
callback_data=1))
        markup.add(telebot.types.InlineKeyboardButton(text='EN',
callback_data=2))

        bot.send_message(message.chat.id, "Выбери язык, на который хочешь
перевести текст.", parse_mode='html', reply_markup=markup)
    else:
        markup = telebot.types.ReplyKeyboardMarkup(resize_keyboard=True)
        btn2 = types.KeyboardButton("Переведи текст")
        markup.add(btn2)
        mess = f'Не понял тебя(... Жми на кнопку или напиши: "Переведи
текст".'
        bot.send_message(message.chat.id, mess, parse_mode='html',
reply_markup=markup)

def next_trans2(message):
    try:
        text = int(message.text)
        bot.send_message(message.chat.id, "Это не текст!")
    except:
        text = message.text
        lang = 'en'
        res = translator.translate(text, dest=lang)
        bot.send_message(message.chat.id, res.text)

def next_trans3(message):
    try:
        text = int(message.text)
        bot.send_message(message.chat.id, "Это не текст!")
```

```
    except:
        text = message.text
        lang = 'ru'
        res = translator.translate(text, dest=lang)
        bot.send_message(message.chat.id, res.text)

@bot.callback_query_handler(func= lambda call:True)
def query_handler(call):
    bot.answer_callback_query(callback_query_id=call.id)
    answer = ''

    if call.data == '1':
        markup = telebot.types.InlineKeyboardMarkup()
        markup.add(telebot.types.InlineKeyboardButton(text="Выбрать другой
язык.", callback_data=3))
        markup.add(telebot.types.InlineKeyboardButton(text="Отмена",
callback_data=4))
        msg = bot.edit_message_text(chat_id = call.message.chat.id,
message_id=call.message.message_id, text = "Введите текст для перевода",
reply_markup= markup)
        bot.register_next_step_handler(msg, next_trans3)
    elif call.data == '2':
        markup = telebot.types.InlineKeyboardMarkup()
        markup.add(telebot.types.InlineKeyboardButton(text="Выбрать другой
язык.", callback_data=3))
        markup.add(telebot.types.InlineKeyboardButton(text="Отмена",
callback_data=4))
        msg = bot.edit_message_text(chat_id = call.message.chat.id,
message_id=call.message.message_id, text = "Введите текст для перевода",
reply_markup= markup)
        bot.register_next_step_handler(msg, next_trans2)
    elif call.data == '3':
        markup = telebot.types.InlineKeyboardMarkup()
        markup.add(telebot.types.InlineKeyboardButton(text="RU",
callback_data=1))
        markup.add(telebot.types.InlineKeyboardButton(text="EN",
callback_data=2))
        msg = bot.edit_message_text(chat_id = call.message.chat.id,
message_id=call.message.message_id, text = "Выбери язык, на который хочешь
перевести текст", reply_markup= markup)
    elif call.data == '4':
        markup = telebot.types.ReplyKeyboardMarkup(resize_keyboard=True)
        btn2 = types.KeyboardButton("Переведи текст")
        markup.add(btn2)
        mess = f'Вы вернулись в главное меню!'
        bot.send_message(call.message.chat.id, mess, parse_mode='html',
reply_markup=markup)

bot.polling()
```

Тесты:


/start 8:24
Hello, Aleksandra ! Я переводчик с английского на русский и наоборот :) 8:24
Переведи текст 8:24
Выбери язык, на который хочешь перевести текст. 8:24
RU
EN


Введите текст для перевода 8:24
Выбрать другой язык.
Отмена
cat 8:27
кошка 8:27
кот 8:27
cat 8:27
Вы вернулись в главное меню! 8:27
Write a message...
Переведи текст

4. Вывод

В результате работы были решены следующие задачи:

- Изучена краткая история языка Python, а также некоторые основные сферы его применения
- Рассмотрены основные особенности языка Python
- Изучены новые библиотеки Python для разработки чат-бота
- Создан Telegram бот переводчик

5. Список литературы

1) https://github.com/ssanchellaa/TelegramBot-Translator- - репозиторий с программой на GitHub

2) https://tlgrm.ru/docs/bots/api - справочник по Bot API

3) https://pypi.org/project/pyTelegramBotAPI/

4) https://habr.com/ru/post/679832/ - статья-вдохновитель

Примечание: работа выполнялась в 2022.12.17