|  |  |  |
| --- | --- | --- |
| Защищено:  Папин А.В..    "\_\_" \_\_\_\_\_\_\_\_\_\_\_2022 г. |  | Демонстрация:  Папин А.В..    "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |

**Отчет по лабораторной работе №6 по курсу**

**базовые компоненты интернет-технологий (БКИТ)**

#### Тема работы: "Разработка бота на основе конечного автомата для Telegram с использованием языка Python."

14

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-54Б Папин Алексей | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Гапанюк Ю.Е. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |

Москва, МГТУ - 2022

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**СОДЕРЖАНИЕ ОТЧЕТА**

[1. Цель лабораторной работы 2](#_Toc120625722)

[2. Описание задания. 2](#_Toc120625723)

[3. Листинг программы: 3](#_Toc120625724)

[3.1. config.py 3](#_Toc120625725)

[3.2. calculate\_arifmetic.py 3](#_Toc120625726)

[4.1. calculate\_bot.py 4](#_Toc120625727)

[4.2. json\_function.py 7](#_Toc120625728)

[4.3. work\_with\_calculate.py 7](#_Toc120625729)

[4.4. bmstu.jpg 8](#_Toc120625730)

[5. Результаты работы программы в Telegram 9](#_Toc120625731)

[5.1. Получение справочную информацию 9](#_Toc120625732)

[5.2. Основное меню переключателя 9](#_Toc120625733)

[5.3. Простейший калькулятор (при нажатии на кнопку «Посчитать») 9](#_Toc120625734)

[5.4. Данные хранятся в БД в формате JSON 10](#_Toc120625735)

[5.5. После несколько вычислений 10](#_Toc120625736)

[5.6. Обновленная БД 11](#_Toc120625737)

[5.7. Чтение и просмотр БД в Телеграме (после нажатии на кнопку Посмотреть историю вычисления) 12](#_Toc120625738)

[5.8. Просмотр фотографии (после нажатии на кнопку «Показать фото МГТУ им. Н.Э. Баумана») 13](#_Toc120625739)

# **Цель лабораторной работы**

Изучение разработки ботов в Telegram.

# **Описание задания.**

Выберите любой фрагмент кода из лабораторных работ 1 или 2 или 3-4.

1. Разработайте бота для Telegram. Бот должен реализовывать конечный автомат из трех состояний.

# **Листинг программы:**

# **config.py**

token = ''

# **calculate\_arifmetic.py**

1. # Преобразование строкого типа в list  
   def delete\_space\_into\_list(string):  
    new\_str = []  
    str\_value = ''  
    for i in string:  
    if(i != ' '):  
    str\_value += i  
    else:  
    new\_str.append(str\_value)  
    str\_value = ''  
     
    new\_str.append(str\_value)  
     
    return new\_str  
     
     
   # Расстановка приоритета операции  
   def enumeration\_sign(list\_str):  
    count\_list = []  
    for i in list\_str:  
    if ('\*' == i): count\_list.append(i)  
    if ('/' == i): count\_list.append(i)  
    if ('+' == i): count\_list.append(i)  
    if ('-' == i): count\_list.append(i)  
     
    count\_list = prioritet(count\_list)  
     
    return count\_list  
     
   # Поддержка функции по расстановку приоритета операции  
   def prioritet(list\_str):  
    new\_list = []  
    size = len(list\_str)  
    count = 0  
    while (size != 0):  
    if('\*' in list\_str or '/' in list\_str):  
    for i in list\_str:  
    if(i == '\*' or i == '/'):  
    new\_list.append(i)  
    size -= 1  
    if('+' in list\_str or '-' in list\_str):  
    for i in list\_str:  
    if(i == '+' or i == '-'):  
    new\_list.append(i)  
    size -= 1  
     
    return new\_list  
     
   # Арифметические операции  
   def arifmetic(sign, list):  
    result = None  
    if (sign in list):  
    for i in range(1, len(list) - 1):  
    try:  
    if(list[i] == sign):  
    if(sign == '\*'): result = float(list[i - 1]) \* float(list[i + 1])  
    elif(sign == '/'): result = float(list[i - 1]) / float(list[i + 1])  
    elif (sign == '+'): result = float(list[i - 1]) + float(list[i + 1])  
    elif (sign == '-'): result = float(list[i - 1]) - float(list[i + 1])  
     
    list[i] = result  
    del list[i - 1: i]  
    del list[i: i + 1]  
    except:  
    return result  
     
   def calculate(value):  
    new\_list = delete\_space\_into\_list(value)  
    list\_en = enumeration\_sign(new\_list)  
     
    for sgin in list\_en:  
    arifmetic(sgin, new\_list)  
     
    print(float(new\_list[0]))  
    return float(new\_list[0])

# **calculate\_bot.py**

import config  
import telebot  
from telebot import types  
import random  
  
from calculate\_arifmetic import calculate  
from calculate.work\_with\_calculate import get\_info  
from json\_function import merge\_data  
  
# Создание бота  
bot = telebot.TeleBot(config.token)  
  
HELP = '''  
/start - Меню переключателя  
/calculate - Калькуляторный бот, способный вычислять простейшие арифметические операции  
/get\_info - Просмотр историю вычисления с БД  
/photo - Просмотр фото МГТУ им. Н.Э. Баумана  
'''  
  
  
# Справочник  
@bot.message\_handler(commands=['help'])  
def start(message):  
 bot.send\_message(message.chat.id, HELP)  
  
  
@bot.message\_handler(commands=['start'])  
def start(message):  
 markup = types.InlineKeyboardMarkup(row\_width=1)  
 btn1 = types.InlineKeyboardButton(text="Посчитать", callback\_data='btn1')  
 btn2 = types.InlineKeyboardButton(text="Посмотреть историю вычисления", callback\_data='btn2')  
 btn3 = types.InlineKeyboardButton(text="Показать фото МГТУ им. Н.Э. Баумана", callback\_data='btn3')  
 markup.add(btn1, btn2, btn3)  
 bot.send\_message(message.chat.id,  
 text=f"Привет, {message.from\_user.first\_name}! Я тестовый бот, выберите действия",  
 reply\_markup=markup)  
  
  
# Функция переключателя  
@bot.callback\_query\_handler(func=lambda callback: callback.data)  
def check\_callback\_data(callback):  
 if (callback.data == "btn1"):  
 bot.send\_message(callback.message.chat.id, 'Калькулятор бот')  
 bot.send\_message(callback.message.chat.id, 'Напишите в чате вычисления')  
  
 @bot.message\_handler(content\_types=["text"])  
 def echo(message):  
 # Пользовательский идентификатор  
 user\_id = str(message.from\_user.id)  
  
 value = calculate(message.text)  
 bot.send\_message(message.chat.id, f'Решение: {value}')  
 data = {  
 user\_id: [  
 {"id": random.randint(0, 10000),  
 "value": str(message.text),  
 "result": str(value)}  
 ]  
 }  
 merge\_data(data, str(message.from\_user.id))  
  
 elif(callback.data == "btn2"):  
 bot.send\_message(callback.message.chat.id, 'История вычисления')  
 data = get\_info()  
 for i in data:  
 for j in data[i]:  
 id = j['id']  
 value = j['value']  
 result = j['result']  
 print\_info = f'id: {id}\n{value} = {result}\n\n'  
 bot.send\_message(callback.message.chat.id, print\_info)  
  
 elif(callback.data == "btn3"):  
 img = open('bmstu.jpg', 'rb')  
 bot.send\_photo(callback.message.chat.id, img)  
 else:  
 bot.send\_message(callback.chat.id, 'Нет такой команды. Введите /help')  
  
# Вычисления  
@bot.message\_handler(commands=['calculate'])  
def start\_calculate(message):  
 bot.send\_message(message.chat.id, 'Калькулятор бот')  
 bot.send\_message(message.chat.id, 'Напишите в чате вычисления')  
  
 # Пользовательский идентификатор  
 user\_id = str(message.from\_user.id)  
  
 @bot.message\_handler(content\_types=["text"])  
 def echo(message):  
 value = calculate(message.text)  
 bot.send\_message(message.chat.id, f'Решение: {value}')  
 data = {  
 user\_id: [  
 {"id": random.randint(0, 10000),  
 "value": str(message.text),  
 "result": str(value)}  
 ]  
 }  
 merge\_data(data, str(message.from\_user.id))  
  
  
# Просмотри история вычисления  
@bot.message\_handler(commands=['get\_info'])  
def start\_get\_info(message):  
 bot.send\_message(message.chat.id, 'История вычисления')  
  
 data = get\_info()  
 if (data == 'Файл отсутствует'):  
 bot.send\_message(message.chat.id, 'База данных отсутствует')  
 else:  
 for i in data:  
 for j in data[i]:  
 id = j['id']  
 value = j['value']  
 result = j['result']  
 print\_info = f'id: {id}\n{value} = {result}\n\n'  
 bot.send\_message(message.chat.id, print\_info)  
  
  
@bot.message\_handler(commands=['photo'])  
def url(message):  
 img = open('bmstu.jpg', 'rb')  
 bot.send\_photo(message.chat.id, img)  
  
  
  
bot.polling(none\_stop=True)

# **json\_function.py**

import json  
  
  
def write\_data(data, title='D:\Python\BKIT\calculate\data'):  
 with open(f"{title}.json", "w", encoding="utf-8") as file:  
 json.dump(data, file, indent=2, ensure\_ascii=False)  
  
  
def load\_data(title="D:\Python\BKIT\calculate\data"):  
 with open(f"{title}.json", "r") as file:  
 data = json.load(file)  
 return data  
  
  
def merge\_data(data\_json, id\_user='id\_user', title="D:\Python\BKIT\calculate\data"):  
 # Если файл существует и не пустой  
 try:  
 with open(f"{title}.json", encoding="utf-8") as file:  
 data = json.load(file)  
 temp = data[id\_user]  
 for info\_data in data\_json[id\_user]:  
 y = {  
 'id': info\_data['id'],  
 'value': info\_data['value'],  
 'result': info\_data['result']  
 }  
 temp.append(y)  
 write\_data(data)  
 # Если файл не существует  
 except:  
 write\_data(data\_json)

# **work\_with\_calculate.py**

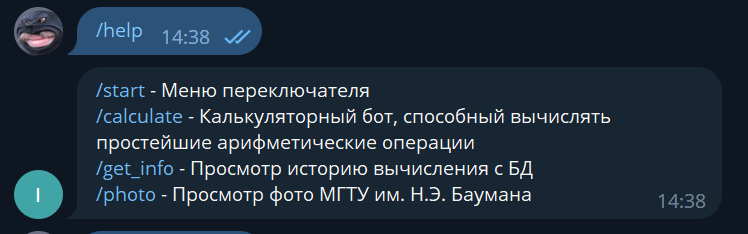
import random  
  
from calculate.json\_function import write\_data, load\_data, merge\_data  
from calculate.calculate\_arifmetic import calculate  
  
  
def generate\_value(id\_user='id\_user'):  
 arifmetic = ['+', '-', '/', '\*']  
  
 af = arifmetic[random.randint(0, 3)]  
 gen\_id = random.randint(0, 100000)  
 v1 = random.randint(0, 1000)  
 v2 = random.randint(0, 1000)  
 result = calculate(str(v1) + ' ' + str(af) + ' ' + str(v2))  
  
 data = {  
 str(id\_user): [  
 {  
 "id": gen\_id,  
 "value": v1,  
 "result": result  
 }  
 ]  
 }  
  
 merge\_data(data, id\_user)  
  
def get\_info():  
 try:  
 data = load\_data()  
 return data  
 except:  
 return 'Файл отсутствует'

# **bmstu.jpg**

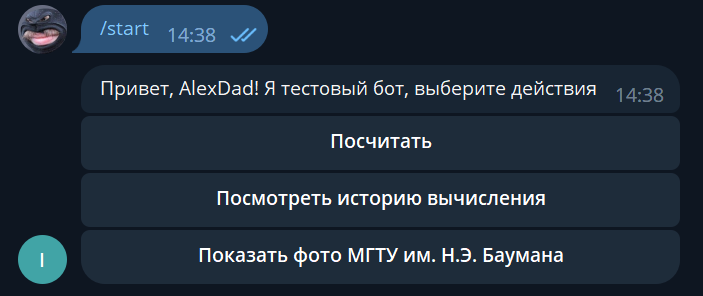


# **Результаты работы программы в Telegram**

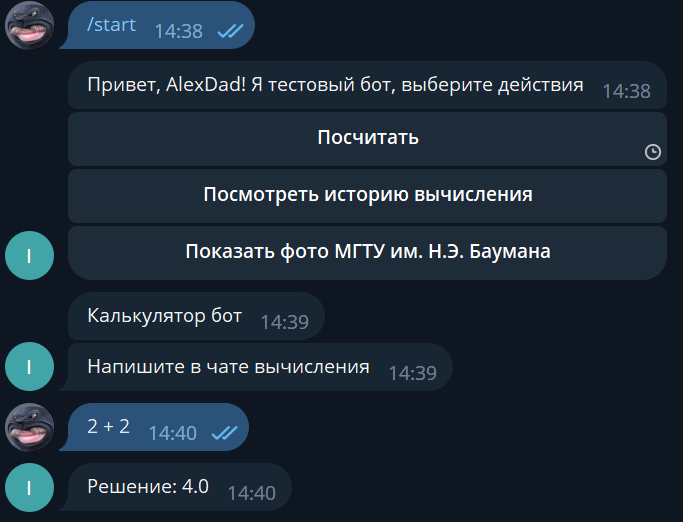
# **Получение справочную информацию**



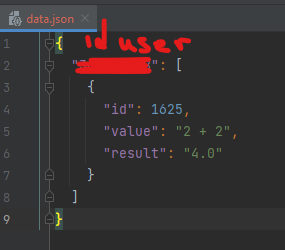
# **Основное меню переключателя**



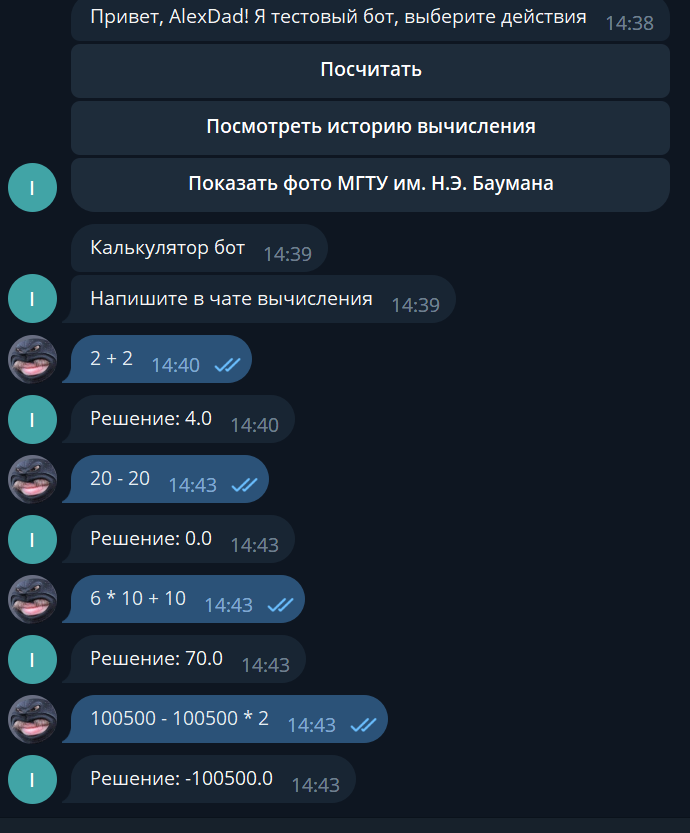
# **Простейший калькулятор (при нажатии на кнопку «Посчитать»)**



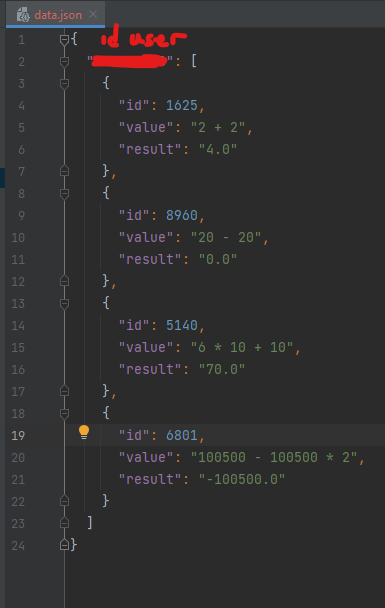
# **Данные хранятся в БД в формате JSON**



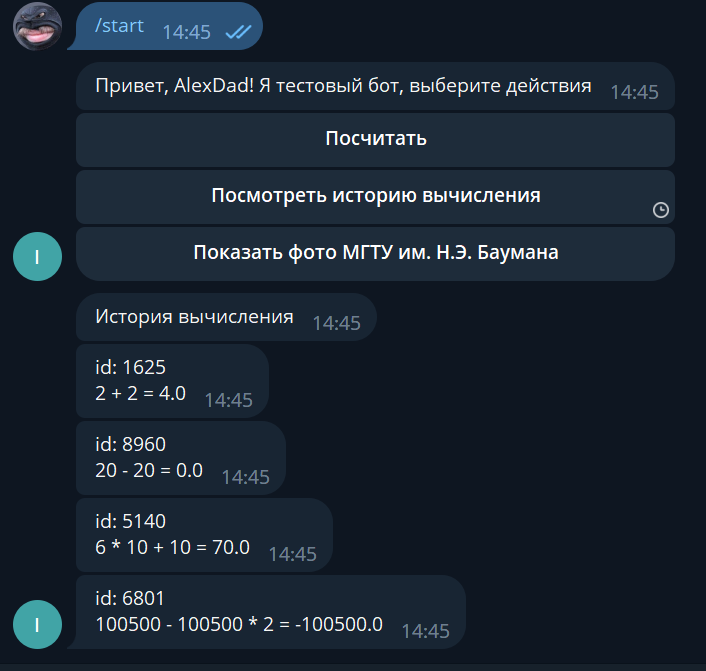
# **После несколько вычислений**



# **Обновленная БД**



# **Чтение и просмотр БД в Телеграме (после нажатии на кнопку Посмотреть историю вычисления)**



# **Просмотр фотографии (после нажатии на кнопку «Показать фото МГТУ им. Н.Э. Баумана»)**

