**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №6

## «Разработка бота на основе конечного автомата для Telegram с использованием языка Python.»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-33 |  | преподаватель каф. ИУ5 |
| Ефременко Даниил |  | Канев Антон. |
| Подпись и дата: |  | Подпись и дата: |

**Задание:**

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

**Текст программы:**

**Lab6\_bot.py**

import telebot  
from telebot import types  
import config  
import dbworker  
  
# Создание бота  
bot = telebot.TeleBot(config.TOKEN)  
  
  
# Начало диалога  
@bot.message\_handler(commands=['start'])  
def cmd\_start(message):  
 name = message.from\_user.first\_name  
 bot.send\_message(message.chat.id, 'Приветствую тебя, '+ name)  
 bot.send\_message(message.chat.id, 'Я бот-калькулятор!')  
 dbworker.set(dbworker.make\_key(message.chat.id, config.CURRENT\_STATE), config.States.STATE\_FIRST\_NUM.value)  
 bot.send\_message(message.chat.id, 'Введите первое число')  
  
  
# По команде /reset будем сбрасывать состояния, возвращаясь к началу диалога  
@bot.message\_handler(commands=['reset'])  
def cmd\_reset(message):  
 bot.send\_message(message.chat.id, 'Сбрасываем результаты предыдущего ввода.')  
 dbworker.set(dbworker.make\_key(message.chat.id, config.CURRENT\_STATE), config.States.STATE\_FIRST\_NUM.value)  
 bot.send\_message(message.chat.id, 'Введите первое число')  
  
  
# Обработка первого числа  
@bot.message\_handler(func=lambda message: dbworker.get(  
 dbworker.make\_key(message.chat.id, config.CURRENT\_STATE)) == config.States.STATE\_FIRST\_NUM.value)  
def first\_num(message):  
 text = message.text  
 if not text.isdigit():  
 # Состояние не изменяется, выводится сообщение об ошибке  
 bot.send\_message(message.chat.id, 'Пожалуйста введите число!')  
 return  
 else:  
 bot.send\_message(message.chat.id, f'Вы ввели первое число {text}')  
 # Меняем текущее состояние  
 dbworker.set(dbworker.make\_key(message.chat.id, config.CURRENT\_STATE), config.States.STATE\_SECOND\_NUM.value)  
 # Сохраняем первое число  
 dbworker.set(dbworker.make\_key(message.chat.id, config.States.STATE\_FIRST\_NUM.value), text)  
 bot.send\_message(message.chat.id, 'Введите второе число')  
  
  
# Обработка второго числа  
@bot.message\_handler(func=lambda message: dbworker.get(  
 dbworker.make\_key(message.chat.id, config.CURRENT\_STATE)) == config.States.STATE\_SECOND\_NUM.value)  
def second\_num(message):  
 text = message.text  
 if not text.isdigit():  
 # Состояние не изменяется, выводится сообщение об ошибке  
 bot.send\_message(message.chat.id, 'Пожалуйста введите число!')  
 return  
 else:  
 bot.send\_message(message.chat.id, f'Вы ввели второе число {text}')  
 # Меняем текущее состояние  
 dbworker.set(dbworker.make\_key(message.chat.id, config.CURRENT\_STATE), config.States.STATE\_OPERATION.value)  
 # Сохраняем первое число  
 dbworker.set(dbworker.make\_key(message.chat.id, config.States.STATE\_SECOND\_NUM.value), text)  
 markup = types.ReplyKeyboardMarkup(row\_width=2)  
 itembtn1 = types.KeyboardButton('+')  
 itembtn2 = types.KeyboardButton('\*')  
 itembtn3 = types.KeyboardButton('/')  
 itembtn4 = types.KeyboardButton('-')  
 markup.add(itembtn1, itembtn2, itembtn3, itembtn4)  
 bot.send\_message(message.chat.id, 'Выберите пожалуйста действие', reply\_markup=markup)  
  
  
# Выбор действия  
@bot.message\_handler(func=lambda message: dbworker.get(  
 dbworker.make\_key(message.chat.id, config.CURRENT\_STATE)) == config.States.STATE\_OPERATION.value)  
def operation(message):  
 # Текущее действие  
 op = message.text  
 # Читаем операнды из базы данных  
 v1 = dbworker.get(dbworker.make\_key(message.chat.id, config.States.STATE\_FIRST\_NUM.value))  
 v2 = dbworker.get(dbworker.make\_key(message.chat.id, config.States.STATE\_SECOND\_NUM.value))  
 # Выполняем действие  
 fv1 = float(v1)  
 fv2 = float(v2)  
 global res  
 res = 0  
 if op == '+':  
 res = fv1 + fv2  
 elif op == '\*':  
 res = fv1 \* fv2  
 elif op == '/':  
 if fv2 == 0:  
 bot.send\_message(message.chat.id, 'На ноль делить нельзя!')  
 dbworker.set(dbworker.make\_key(message.chat.id, config.CURRENT\_STATE), config.States.STATE\_SECOND\_NUM.value)  
 bot.send\_message(message.chat.id, 'Введите второе число')  
 return  
 else:  
 res = fv1 / fv2  
 elif op == '-':  
 res = fv1 - fv2  
 # Выводим результат  
 markup = types.ReplyKeyboardRemove(selective=False)  
 bot.send\_message(message.chat.id, f'Результат: {v1}{op}{v2}={str(res)}', reply\_markup=markup)  
 markup = types.ReplyKeyboardMarkup(row\_width=2)  
 itembtn1 = types.KeyboardButton('MR')  
 itembtn2 = types.KeyboardButton('MC')  
 itembtn3 = types.KeyboardButton('M+')  
 itembtn4 = types.KeyboardButton('M-')  
 itembtn5 = types.KeyboardButton('MS')  
 itembtn6 = types.KeyboardButton('Продолжить вычисления')  
 markup.add(itembtn1, itembtn2, itembtn3, itembtn4, itembtn5, itembtn6)  
 bot.send\_message(message.chat.id, 'Выберите пожалуйста действие', reply\_markup=markup)  
 dbworker.set(dbworker.make\_key(message.chat.id, config.CURRENT\_STATE), config.States.STATE\_EXTRA.value)  
  
@bot.message\_handler(func=lambda message: dbworker.get(  
 dbworker.make\_key(message.chat.id, config.CURRENT\_STATE)) == config.States.STATE\_EXTRA.value)  
def extra(message):  
 result = res  
 extra\_op = message.text  
 if extra\_op == 'MS':  
 dbworker.set(dbworker.make\_key(message.chat.id, config.States.STATE\_EXTRA.value), str(result))  
 bot.send\_message(message.chat.id, 'Число успешно сохранено')  
 elif extra\_op == 'MR':  
 v3 = dbworker.get(dbworker.make\_key(message.chat.id, config.States.STATE\_EXTRA.value))  
 bot.send\_message(message.chat.id, 'Число, сохраненное в памяти:' + v3)  
 elif extra\_op == 'MC':  
 result = 0  
 dbworker.set(dbworker.make\_key(message.chat.id, config.States.STATE\_EXTRA.value), result)  
 bot.send\_message(message.chat.id, 'Число удалено из памяти')  
 elif extra\_op == 'Продолжить вычисления':  
 # Меняем текущее состояние  
 dbworker.set(dbworker.make\_key(message.chat.id, config.CURRENT\_STATE), config.States.STATE\_FIRST\_NUM.value)  
 # Выводим сообщение  
 bot.send\_message(message.chat.id, 'Введите первое число')  
 elif extra\_op == 'M-':  
 k1 = float(dbworker.get(dbworker.make\_key(message.chat.id, config.States.STATE\_EXTRA.value)))  
 result = k1 - res  
 bot.send\_message(message.chat.id, f'Результат: {k1}-{res}={str(result)}')  
 elif extra\_op == 'M+':  
 k1 = float(dbworker.get(dbworker.make\_key(message.chat.id, config.States.STATE\_EXTRA.value)))  
 result = k1 + res  
 bot.send\_message(message.chat.id, f'Результат: {k1}+{res}={str(result)}')  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 bot.infinity\_polling()

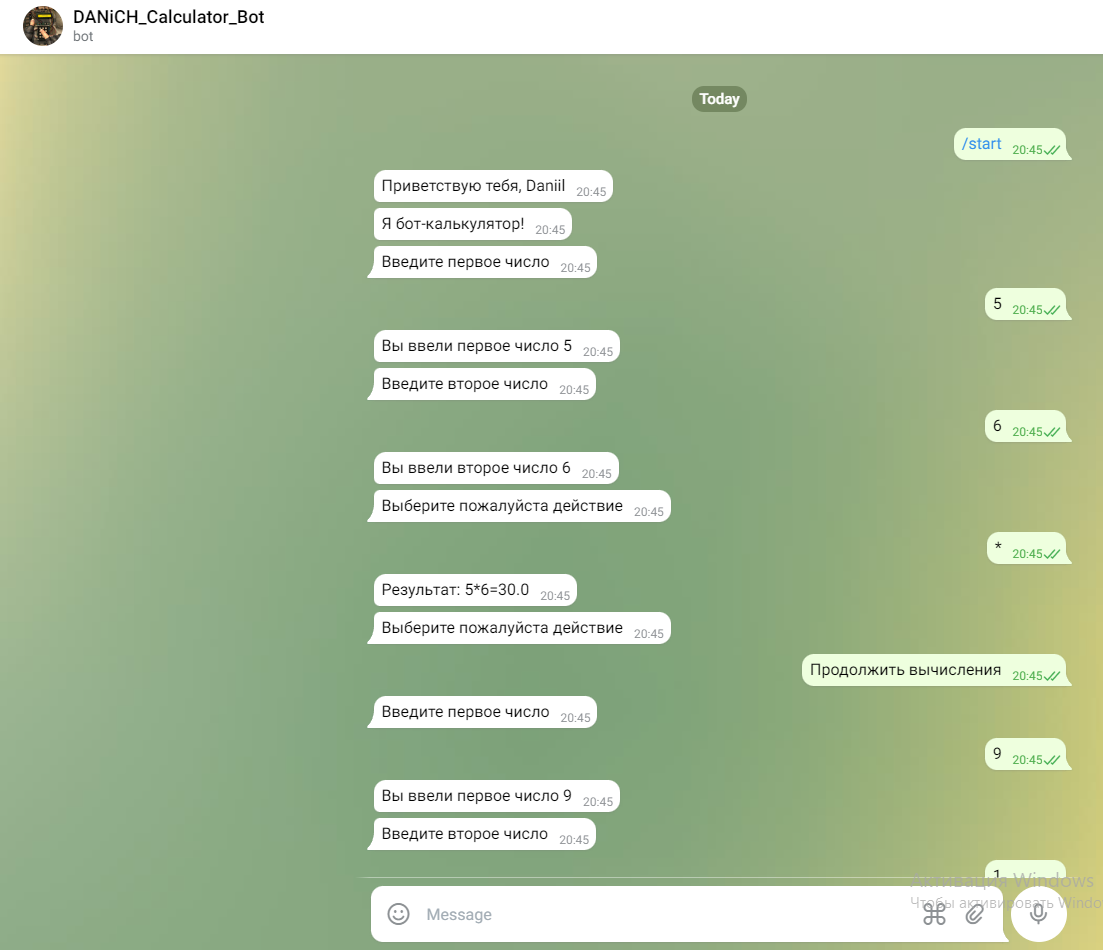
**config.py**

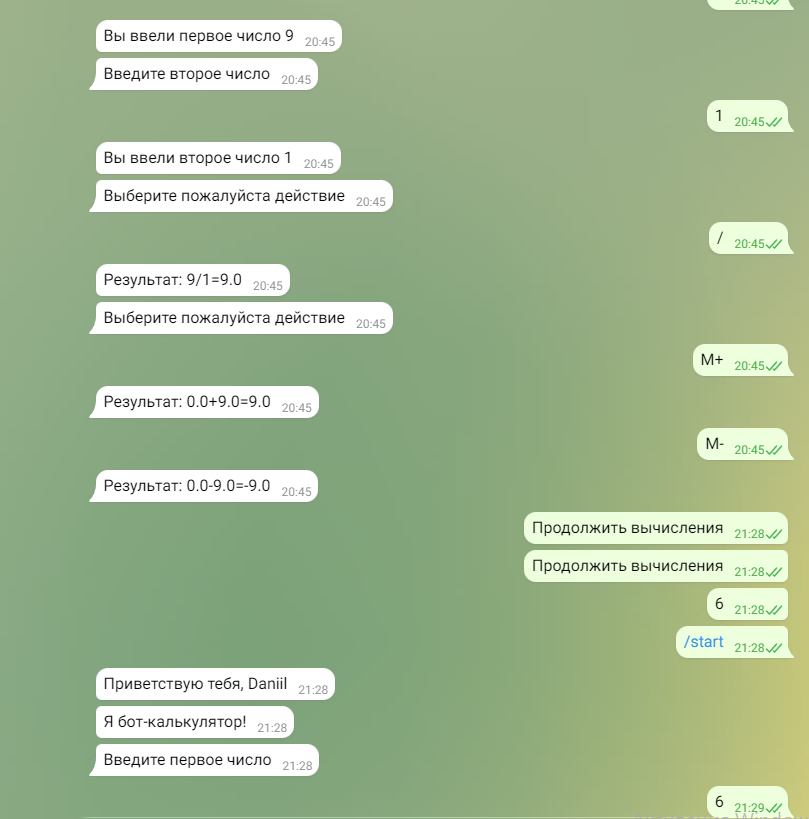
from enum import Enum  
  
# Токент бота  
TOKEN = "5018697130:AAH6ZtxvZEbdsJSKjLRAZ37q71WO1OEEM-o"  
  
# Файл базы данных Vedis  
db\_file = "db.vdb"  
  
# Ключ записи в БД для текущего состояния  
CURRENT\_STATE = "CURRENT\_STATE"  
  
# Состояния автомата  
class States(Enum):  
 STATE\_START = "STATE\_START" # Начало нового диалога  
 STATE\_FIRST\_NUM = "STATE\_FIRST\_NUM"  
 STATE\_SECOND\_NUM = "STATE\_SECOND\_NUM"  
 STATE\_OPERATION = "STATE\_OPERATION"  
 STATE\_EXTRA = "STATE\_EXTRA"

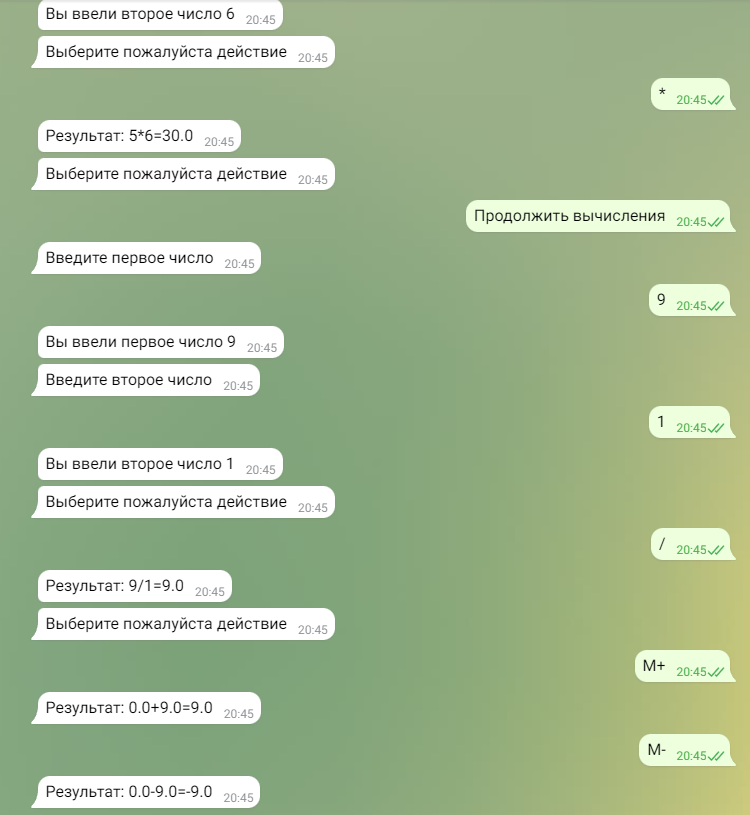
**dbworker.py**

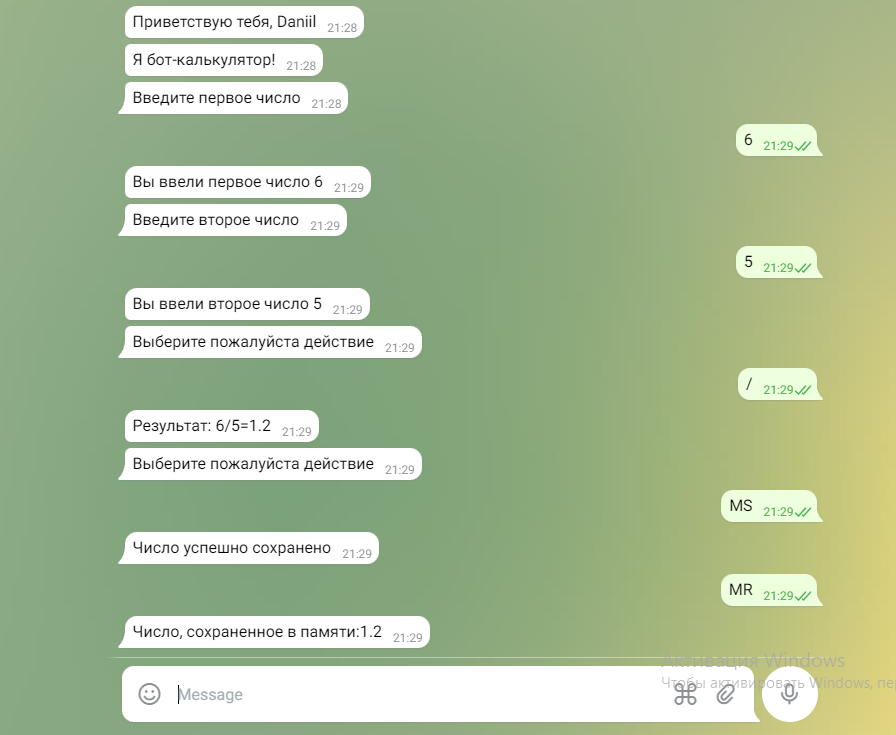
from vedis import Vedis  
import config  
  
# Чтение значения  
def get(key):  
 with Vedis(config.db\_file) as db:  
 try:  
 return db[key].decode()  
 except KeyError:  
 # в случае ошибки значение по умолчанию - начало диалога  
 return config.States.S\_START.value  
  
  
# Запись значения  
def set(key, value):  
 with Vedis(config.db\_file) as db:  
 try:  
 db[key] = value  
 return True  
 except:  
 return False  
  
  
# Создание ключа для записи и чтения  
def make\_key(chatid, keyid):  
 res = str(chatid) + '\_\_' + str(keyid)  
 return res

**Результаты выполнения:**

****

****

****

****