Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего **образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Лабораторная работа № 5-6

по дисциплине «Базовые компоненты интернет-технологий»

Выполнил:
студент группы ИУ5-32Б
Кульктна Д.А.

Проверил:
Канев А.И.

2021 г.

Полученное задание:

Задание:

1. Разработайте простого бота для Telegram. Бот должен использовать функциональность создания кнопок.
2. Разработайте бота для Telegram. Бот должен реализовывать конечный автомат из трех состояний.
3. Модифицируйте код лабораторной работы №6 таким образом, чтобы он был пригоден для модульного тестирования.
4. Используя материалы лабораторной работы №4 создайте модульные тесты с применением TDD - фреймворка (2 теста) и BDD - фреймворка (2 теста).

Bot.py

```
import telebot
from telebot import types
import config
import dbworker

#def
# Создание бота
bot = telebot.TeleBot(config.TOKEN)
step = 0

goods_dict ={'Ручка': 100, 'Карандаш': 150, 'Ластик': 10}

# Начало диалога
@bot.message_handler(commands=['start'], func=lambda message: dbworker.get(
    dbworker.make_key(message.chat.id, config.CURRENT_STATE)) ==
config.States.STATE_OPERATION.value)
def cmd_start(message):
    keyboard = telebot.types.ReplyKeyboardMarkup(True)
    keyboard.row('/info_goods', '/info_pickpoints')
    bot.send_message(message.chat.id, 'Здравствуйте, Я ассистент интернет
магазина, что вы хотите посмотреть!', reply_markup=keyboard)


@bot.message_handler(commands=['info_goods'])
def start_message_info_goods(message):
    markup = telebot.types.InlineKeyboardMarkup()
    markup.add(telebot.types.InlineKeyboardButton(text='Ручка',
callback_data='Ручка'))
    markup.add(telebot.types.InlineKeyboardButton(text='Карандаш',
callback_data='Карандаш'))
    markup.add(telebot.types.InlineKeyboardButton(text='Ластик',
callback_data='Ластик'))
    markup.add(telebot.types.InlineKeyboardButton(text='Отменить',
callback_data='Отменить'))
    bot.send_message(message.chat.id, text="Список товаров",
reply_markup=markup)
    dbworker.set(dbworker.make_key(message.chat.id, config.CURRENT_STATE),
(config.States.STATE_FIRST_GOOD.value))


@bot.callback_query_handler(func=lambda call: True)
def query_handler(call):
        global step
```

```
        if call.data == 'Результат':
            dbworker.set(dbworker.make_key(call.message.chat.id,
config.CURRENT_STATE), config.States.STATE_RESULT.value)
            operation(call.message)
            return
        elif call.data == 'Отменить':
            step = 0
            dbworker.set(dbworker.make_key(call.message.chat.id,
config.CURRENT_STATE), config.States.STATE_START.value)
            cmd_start(call.message)
            return

        if step == 0:
            first_num(call.data, call.message)
        else:
            second_num(call.data, call.message)



@bot.message_handler(commands=['info_pickpoints'])
def start_message_info_pickpoints(message):
        bot.send_message(message.chat.id, text="Информация о пунктах выдачи"
)


@bot.message_handler(func=lambda message: dbworker.get(
    dbworker.make_key(message.chat.id, config.CURRENT_STATE)) ==
config.States.STATE_FIRST_GOOD.value)
def first_num(text, message):
    global step

    step += 1
    if not text in goods_dict.keys():
        # Состояние не изменяется, выводится сообщение об ошибке
        bot.send_message(message.chat.id, f'Выберете товар!,{text}')
        return
    else:
        bot.send_message(message.chat.id, f'Вы выбрал первый товар {text}')
        # Меняем текущее состояние
        dbworker.set(dbworker.make_key(message.chat.id,
config.CURRENT_STATE), config.States.STATE_SECOND_GOOD.value)
        # Сохраняем первое число
        dbworker.set(dbworker.make_key(message.chat.id,
config.States.STATE_FIRST_GOOD.value), goods_dict[text])
        bot.send_message(message.chat.id, 'Введите второй товар')
        start_message_info_goods(message)


# По команде /reset будем сбрасывать состояния, возвращаясь к началу диалога
@bot.message_handler(commands=['reset'])
def cmd_reset(message, chat):
    bot.send_message(chat.id, 'Сбрасываем результаты предыдущего ввода.')
    dbworker.set(dbworker.make_key(message.chat.id, config.CURRENT_STATE),
config.States.STATE_START.value)
    cmd_start(message)


# Обработка второго числа
@bot.message_handler(func=lambda message: dbworker.get(
    dbworker.make_key(message.chat.id, config.CURRENT_STATE)) ==
config.States.STATE_SECOND_GOOD.value)
def second_num(text, message):
    global step
    if not text in goods_dict.keys():
```

```
        # Состояние не изменяется, выводится сообщение об ошибке
        bot.send_message(message.chat.id, 'Выберете товар!')
        return
    else:
        bot.send_message(message.chat.id, f'Вы выбрали второй товар {text}')
         # Сохраняем первое число
        dbworker.set(dbworker.make_key(message.chat.id,
config.States.STATE_SECOND_GOOD.value), goods_dict[text])
        markup = telebot.types.InlineKeyboardMarkup()
        itembtn1 = types.InlineKeyboardButton(text='Результат',
callback_data='Результат')
        itembtn2 = types.InlineKeyboardButton(text='Отменить',
callback_data='Отменить')
        markup.add(itembtn1, itembtn2)
        bot.send_message(message.chat.id, 'Выберете', 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):
    global step
    # Текущее действие
    op = message.text
    # Читаем операнды из базы данных
    v1 = dbworker.get(dbworker.make_key(message.chat.id,
config.States.STATE_FIRST_GOOD.value))
    v2 = dbworker.get(dbworker.make_key(message.chat.id,
config.States.STATE_SECOND_GOOD.value))
    # Выполняем действие
    fv1 = float(v1)
    fv2 = float(v2)
    res = result(fv1, fv2)
    # Выводим результат
    markup = types.ReplyKeyboardRemove(selective=False)
    bot.send_message(message.chat.id, f'Чек: {v1} + {v2}={str(res)}',
reply_markup=markup)
    # Меняем текущее состояние
    dbworker.set(dbworker.make_key(message.chat.id, config.CURRENT_STATE),
config.States.STATE_START.value)
    step = 0
    cmd_start(message)


def result(good1, good2):
    return good1 + good2


def check_goods(good):
    return good in goods_dict


if __name__ == '__main__':
    bot.infinity_polling()
```

confing,py

```
from enum import Enum

# Токент бота
TOKEN = "5098622783:AAFPtKvJINR7Lkby9wXxTjXvZv3Yz4V-3Lk"

# 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_GOOD = "STATE_FIRST_GOOD"
    STATE_SECOND_GOOD = "STATE_SECOND_GOOD"
    STATE_RESULT = "STATE_RESULT"
```

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:
            # тут желательно как-то обработать ситуацию
            return False


# Создание ключа для записи и чтения
def make_key(chatid, keyid):
    res = str(chatid) + '__' + str(keyid)
    return res
```

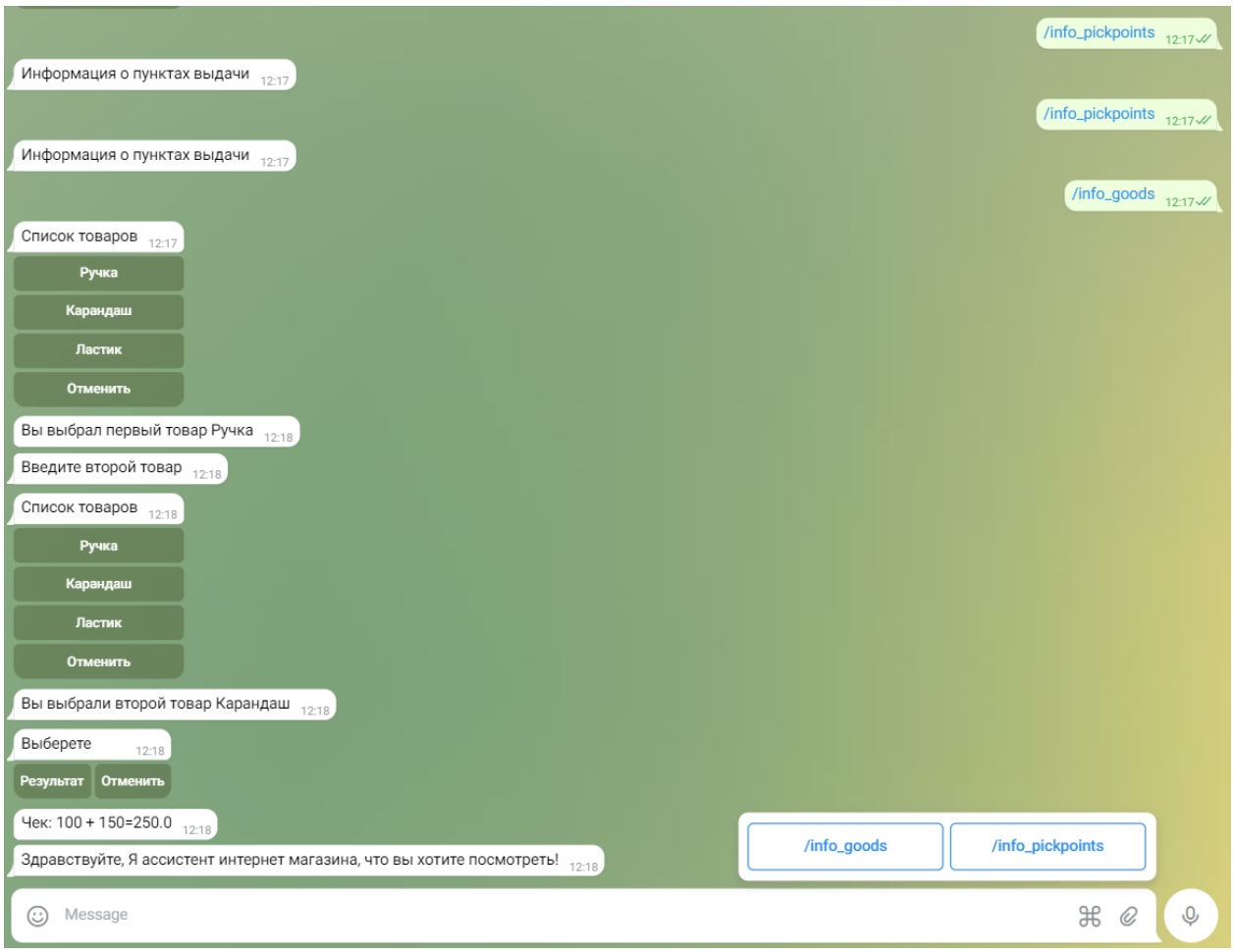
```
import telebot
import unittest
import os

import bot
import config
from bot import result, check_goods


TOKEN = config.TOKEN


class Test_summ(unittest.TestCase):
    def test_result(self):
        self.assertEqual(result(100, 100), 200)

    def test_send_info_pp(self):
        self.assertEqual(check_goods('Ручка'), True)
        self.assertEqual(check_goods('Пенал'), False)
```




```
Ran 1 test in 0.002s

OK
```