## Московский государственный технический университет имени Н.Э.Баумана

## Факультет РТ Радиотехнический

## Кафедра ИУ5 Системы обработки информации и управления

**Отчет по домашней работе по курсу Базовые компоненты интернет-технологий**

|  |  |
| --- | --- |
| Исполнитель |  |
| Студент группы РТ5-31Б | Мицкевич В.Б. |
|  | “ ” 2021 г. |
| Проверил |  |
| Доцент кафедры ИУ5 | Гапанюк Ю.Е. |
|  | “ ” 2021 г. |

## 2021г.

**Содержание**

1. [Описание задания 3](#_bookmark0)
2. [Текст программы 3](#_bookmark1)
3. [Экранные формы с примерами выполнения программы 5](#_bookmark2)

# Описание задания

## Модифицировать код лабораторной работы №6 таким образом, чтобы он был пригоден для модульного тестирования.

## Используя материалы лабораторной работы №4 создать модульные тесты с применением TDD – фреймворка (2 теста) и BDD – фреймворка (2 теста).

# Текст программы

### 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):

dbworker.set(dbworker.make\_key("STATE\_FIRST\_NUM", config.CURRENT\_STATE), config.States.STATE\_FIRST\_NUM.value)

return message

# По команде /reset будем сбрасывать состояния, возвращаясь к началу диалога @bot.message\_handler(commands=['reset'])

def cmd\_reset(message):

dbworker.set(dbworker.make\_key("STATE\_FIRST\_NUM", config.CURRENT\_STATE), config.States.STATE\_FIRST\_NUM.value)

# Обработка первого числа @bot.message\_handler(func=lambda message:

dbworker.get(dbworker.make\_key("STATE\_FIRST\_NUM", config.CURRENT\_STATE)) == config.States.STATE\_FIRST\_NUM.value)

def first\_num(message): if not message.isdigit():

return None else:

# Меняем текущее состояние dbworker.set(dbworker.make\_key("STATE\_SECOND\_NUM",

config.CURRENT\_STATE), config.States.STATE\_SECOND\_NUM.value) # Сохраняем первое число

dbworker.set(dbworker.make\_key("first", config.States.STATE\_FIRST\_NUM.value), message)

return message

# Обработка второго числа @bot.message\_handler(func=lambda message:

dbworker.get(dbworker.make\_key("STATE\_SECOND\_NUM", config.CURRENT\_STATE))

== config.States.STATE\_SECOND\_NUM.value) def second\_num(message):

if not message.isdigit(): return None

else:

# Меняем текущее состояние dbworker.set(dbworker.make\_key("STATE\_OPERATION", config.CURRENT\_STATE),

config.States.STATE\_OPERATION.value) # Сохраняем первое число

dbworker.set(dbworker.make\_key("second", config.States.STATE\_SECOND\_NUM.value), message)

return message

# Выбор действия @bot.message\_handler(func=lambda message:

dbworker.get(dbworker.make\_key("STATE\_OPERATION", config.CURRENT\_STATE)) == config.States.STATE\_OPERATION.value)

def operation(message): # Текущее действие op = message

# Читаем операнды из базы данных

v1 = dbworker.get(dbworker.make\_key("first", config.States.STATE\_FIRST\_NUM.value)) v2 = dbworker.get(dbworker.make\_key("second",

config.States.STATE\_SECOND\_NUM.value)) # Выполняем действие

fv1 = float(v1) fv2 = float(v2) res = None

try:

if op == '+':

res = fv1 + fv2 elif op == '\*':

res = fv1 \* fv2 elif op == '-':

res = fv1 - fv2 elif op == '/':

res = fv1 / fv2 except:

pass return res

### config.py

from enum import Enum # Токент бота

TOKEN = '5089524761:AAGbKxFfd9Mf-s57b5h1BL1EDwju7pPmkdo'

# 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"

### 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 config.States.STATE\_START.value

# Создание ключа для записи и чтения def make\_key(chatid, keyid):

res = str(chatid) + ' ' + str(keyid) return res

### TDD.py

import unittest import bot

class Test(unittest.TestCase): def testOperation(self): bot.first\_num("6")

bot.second\_num("7")

self.assertEqual(bot.operation("\*"), 42)

self.assertEqual(bot.operation("+"), 13)

if name == " main ": unittest.main()

### BDD.feature

Feature: testing bot Scenario: sum two digit

Given I send bot message /start When I send bot first message 4 When I send bot second message 6

Then I send bot operation + and get answer 10

### BDD.py

import sys

sys.path.insert(0, "D:/Univer/Prog/Python/dz/env")

from behave import given, when, then import bot

@given('I send bot message {start}') def step\_impl(context, start: str):

bot.cmd\_start(start)

@when('I send bot first message {firstNum}') def step\_imp2(context, firstNum: str):

context.firstNum = bot.first\_num(firstNum)

@when('I send bot second message {secondNum}') def step\_imp3(context, secondNum: str):

context.secondNum = bot.second\_num(secondNum)

@then('I send bot operation + and get answer {result}') def step\_imp4(context, result: str):

float(context.firstNum) + float(context.secondNum) == float(result)

# Экранные формы с примерами выполнения программы

## TDD

## 

## BDD

