# 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»**

Домашнее задание

по дисциплине «Базовые компоненты интернет-технологий»

Выполнил: студент группы ИУ5-33Б

Требуков Д.А.

Проверил: Канев А.И.

2021 г.

**Задание:**

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

1. Используя материалы лабораторной работы №4 создайте модульные тесты с применением TDD - фреймворка (2 теста) и BDD - фреймворка (2 теста).
2. **Текст программы:**

main.py

import telebot import config import db import math import numpy

bot = telebot.TeleBot(config.token)

@bot.message\_handler(commands=['start']) def start\_message(message):

bot.send\_message(message.chat.id, 'Решим биквадратное уравнение, с тебя только коэффициенты!')

db.set(db.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: db.get(

db.make\_key(message.chat.id, config.CURRENT\_STATE)) == config.States.STATE\_FIRST\_NUM.value)

def first\_num(message): text = message.text try:

float(text)

if a\_equal\_zero(text):

bot.send\_message(message.chat.id, 'Первый коэффициент не может быть ноль!')

bot.send\_message(message.chat.id, 'Введите первое число') return

else:

# Состояние не изменяется, выводится сообщение об ошибке # Меняем текущее состояние

db.set(db.make\_key(message.chat.id, config.CURRENT\_STATE), config.States.STATE\_SECOND\_NUM.value)

# Сохраняем первое число db.set(db.make\_key(message.chat.id,

config.States.STATE\_FIRST\_NUM.value), text) bot.send\_message(message.chat.id, 'Введите второе число')

except ValueError:

# Состояние не изменяется, выводится сообщение об ошибке bot.send\_message(message.chat.id, 'Пожалуйста введите число!') return

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

db.make\_key(message.chat.id, config.CURRENT\_STATE)) == config.States.STATE\_SECOND\_NUM.value)

def second\_num(message): text = message.text try:

float(text)

# Меняем текущее состояние db.set(db.make\_key(message.chat.id, config.CURRENT\_STATE),

config.States.STATE\_THIRD\_NUM.value) # Сохраняем второе число

db.set(db.make\_key(message.chat.id, config.States.STATE\_SECOND\_NUM.value), text)

bot.send\_message(message.chat.id, 'Введите третье число')

except ValueError:

bot.send\_message(message.chat.id, 'Пожалуйста введите число!') return

# Обработка третьего числа @bot.message\_handler(func=lambda message: db.get(

db.make\_key(message.chat.id, config.CURRENT\_STATE)) == config.States.STATE\_THIRD\_NUM.value)

def third\_num(message): text = message.text try:

float(text)

# Сохраняем третье число db.set(db.make\_key(message.chat.id,

config.States.STATE\_THIRD\_NUM.value), text) # Нахождение корней

v1 = db.get(db.make\_key(message.chat.id, config.States.STATE\_FIRST\_NUM.value))

v2 = db.get(db.make\_key(message.chat.id, config.States.STATE\_SECOND\_NUM.value))

v3 = db.get(db.make\_key(message.chat.id, config.States.STATE\_THIRD\_NUM.value))

a = float(v1) b = float(v2) c = float(v3)

resultfinal = bikvadrat(a, b, c) bot.send\_message(message.chat.id, "Имеем

корни:{}".format(resultfinal))

# Меняем текущее состояние db.set(db.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') except ValueError:

bot.send\_message(message.chat.id, 'Введите число!') return

def a\_equal\_zero(text): if text == "0":

return 1 else:

return 0

def bikvadrat(a, b, c): result = []

D = b \* b - 4 \* a \* c if D == 0.0:

root = -b / (2.0 \* a) result.append(root)

elif D > 0.0:

sqD = math.sqrt(D)

root1 = (-b + sqD) / (2.0 \* a) root2 = (-b - sqD) / (2.0 \* a) result.append(root1) result.append(root2)

resultfinal = [] for x in result:

if x > 0:

resultfinal.append(numpy.sqrt(x)) resultfinal.append(-numpy.sqrt(x))

elif x == 0:

resultfinal.append(0) return resultfinal

if name == ' main ': # делает нам бесконечный цикл получения данных bot.infinity\_polling()

## TDD:

test.py

import unittest import sys, os

sys.path.append(os.getcwd()) from main import \*

class Test\_bikvadrat(unittest.TestCase): def test\_a\_equal\_zero(self):

self.assertEqual(a\_equal\_zero("0"), 1)

self.assertEqual(a\_equal\_zero("3242432"), 0)

def test\_result\_bikvadrat(self): self.assertEqual(bikvadrat(1, -2, 1), [1, -1])

self.assertEqual(bikvadrat(1, -6, 9), [1.7320508075688772, -

1.7320508075688772])

BDD:

myfeaturesteps.py

from behave import \* from tests.test import \*

@given('Bot')

def first\_step(context): context.a = Test\_bikvadrat()

@when('test\_a\_equal\_zero return OK')

def check\_sneakers(context): context.a.test\_a\_equal\_zero()

@when('test\_result\_bikvadrat return OK') def check\_slates(context):

context.a.test\_result\_bikvadrat()

@then('good job')

def last\_step(context): pass

myfeature.feature

*# -- FILE: features/example.feature*

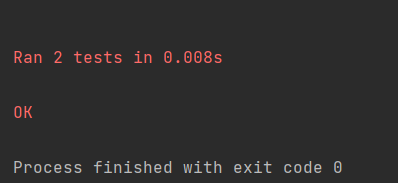
Feature: Showing off behave

Scenario: Function return message about creation Given Bot

When test\_a\_equal\_zero return OK And test\_result\_bikvadrat return OK Then good job

**Результаты выполнения:**

## TDD:



BDD:

