Московский государственный технический университет имени Н. Э. Баумана

Факультет Радиотехнический

Кафедра ИУ5 Системы обработки информации и управления

Отчет по домашней работе по курсу
Базовые компоненты интернет-технологий

8
(количество листов)

Вариант № **16**

Исполнитель

студент группы РТ5-31б _____ Нижаметдинов М.Ш.

“ 21 ” декабря 2021 г.

Проверил

Доцент кафедры ИУ5 _____ Гапанюк Ю.Е.

“___” _____ 2021 г.

Москва, 2021 г.

Задание

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

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

Текст программы

Файл «main.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):
    bot.send_message(message.chat.id, 'Я умею выполнять действия над двумя
числами!')
    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('/')
        itembtn5 = types.KeyboardButton('%')
        markup.add(itembtn1, itembtn2, itembtn3, itembtn4, itembtn5)
        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)
    res = 0
    if op == '+':
        res = fv1 + fv2
    elif op == '-':
        res = fv1 - fv2
    elif op == '*':
        res = fv1 * fv2
    elif op == '/':
        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)
    # 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')
```

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

Файл «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:
            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
```

Файл «config.py»

```
from enum import Enum

# Токент бота
TOKEN = "5008144202:AAEwTj3fvJYUPRHvqJ3KAH754I4SauaWTDM"

# 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"
```

Файл «TEST_TDD.py»

```
import unittest
from main import sum
from main import substract
from main import multiply
from main import diverge
from main import get_mod


class TEST_TDD(unittest.TestCase):
    def test(self):
        self.assertEqual(sum(-4, 16), 12)
        self.assertEqual(substract(314, 24), 290)
        self.assertEqual(multiply(113, 10), 1130)
```

```
        self.assertEqual(diverge(121, 11), 11)
        self.assertEqual(diverge(13, 0), 'err')
        self.assertEqual(get_mod(15, 2), 1)


if __name__ == "__main__":
    unittest.main()
```

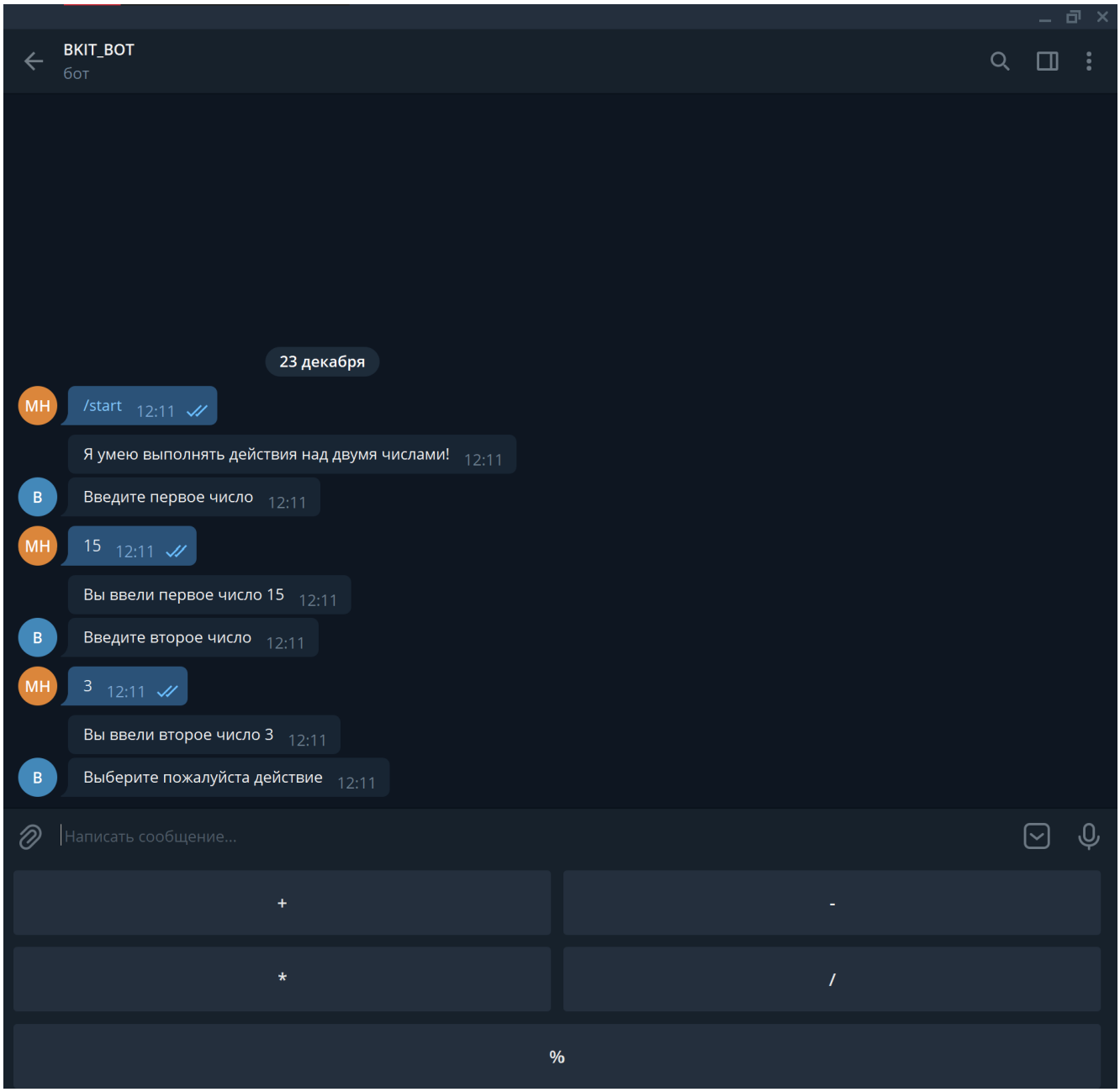
Файл «FEATURE_BDD.feature»

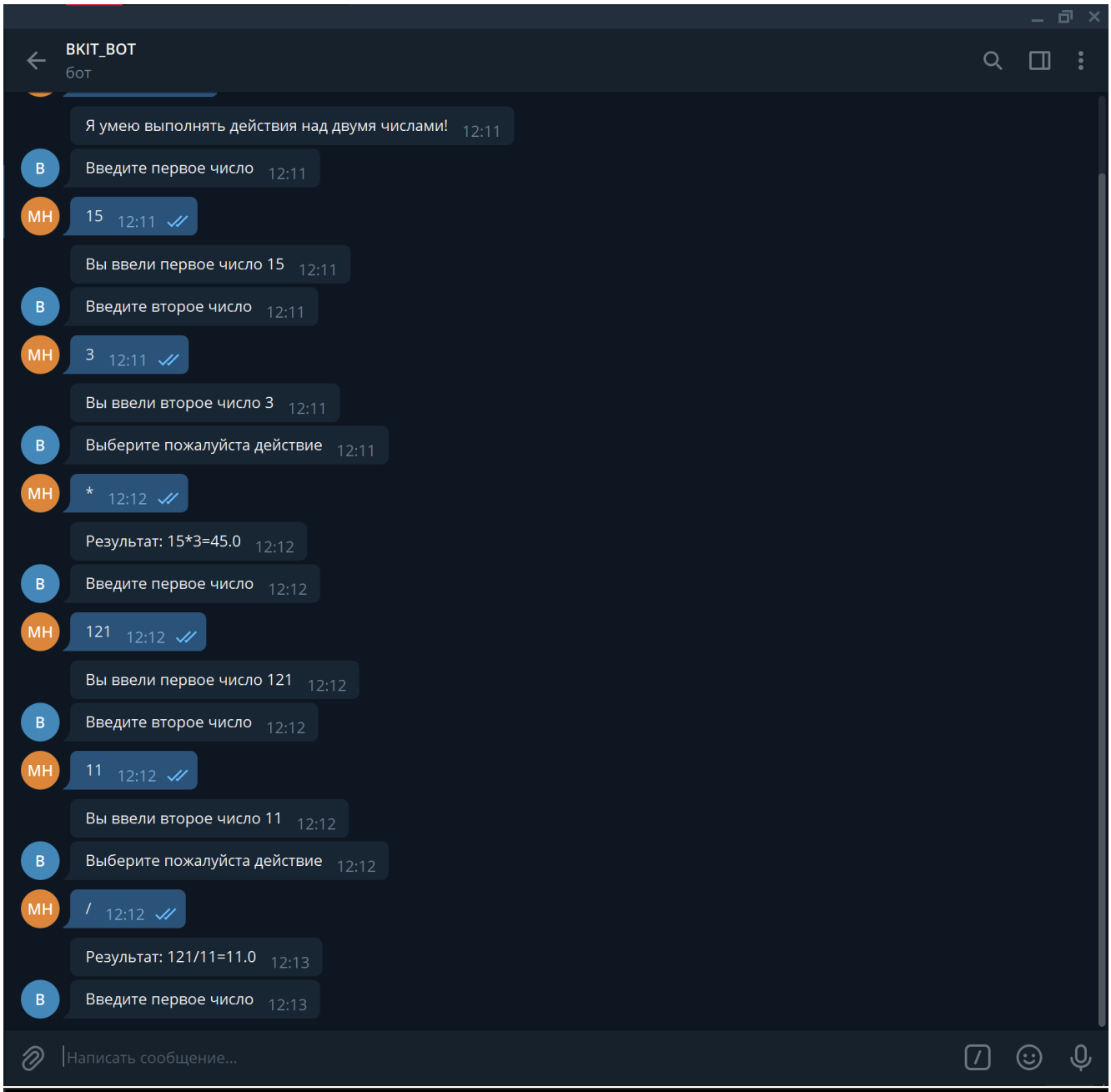
```
Feature: Testing the function get_mod from main
  Scenario: Find the remainder of 14 divided by 13
    Given I put values [14, 13] into the function
    Then I get 1

Scenario: Find the remainder of 121 divided by 2
  Given I put values [121, 2] into the function
  Then I get 1

Scenario: Find the remainder of 21 divided by 9
  Given I put values [21, 9] into the function
  Then I get 3
```

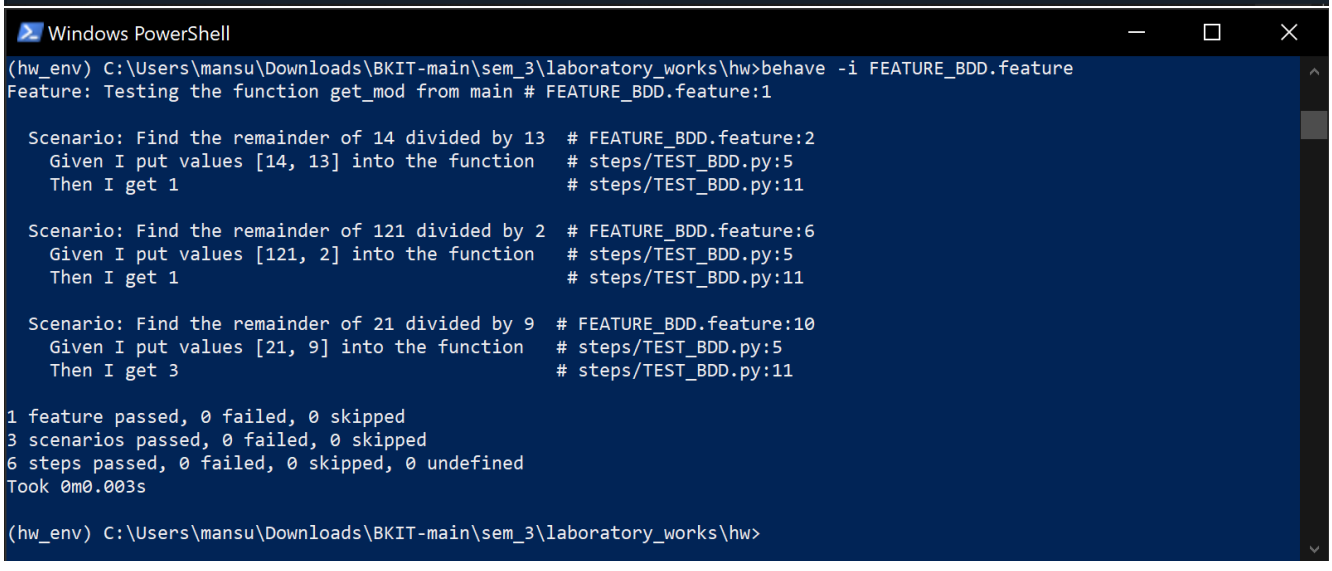
Файл «TEST_BDD.py»

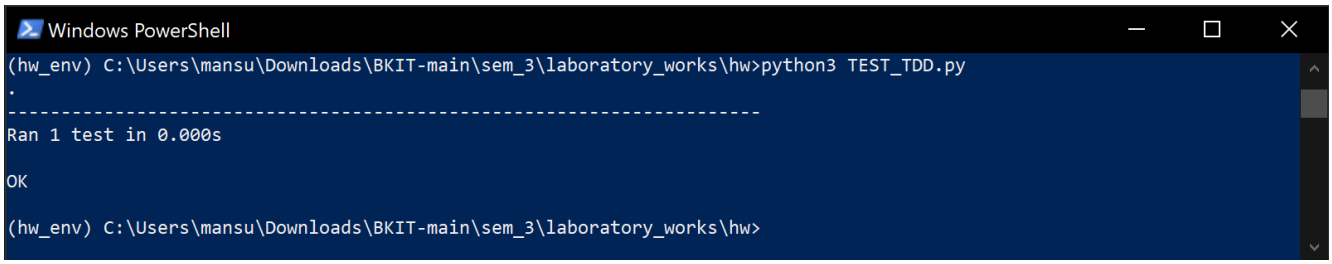
```
from behave import given, then
from main import get_mod


@given('I put values {values} into the function')
def step_impl(context, values: str):
    values = list(map(int, values.replace("[", "").replace("]", "").split(", ")))
    context.result = get_mod(values[0], values[1])


@then('I get {result}')
def step_impl(context, result: str):
    assert str(context.result) == result
```

Экранные формы с примерами выполнения программы




BKIT_BOT
бот
Я умею выполнять действия над двумя числами! 12:11
Введите первое число 12:11
15 12:11
Вы ввели первое число 15 12:11
Введите второе число 12:11
3 12:11
Вы ввели второе число 3 12:11
Выберите пожалуйста действие 12:11
* 12:12
Результат: 15*3=45.0 12:12
Введите первое число 12:12
121 12:12
Вы ввели первое число 121 12:12
Введите второе число 12:12
11 12:12
Вы ввели второе число 11 12:12
Выберите пожалуйста действие 12:12
/ 12:12
Результат: 121/11=11.0 12:13
Введите первое число 12:13
Написать сообщение...


Windows PowerShell
(hw_env) C:\Users\mansu\Downloads\BKIT-main\sem_3\laboratory_works\hw>behave -i FEATURE_BDD.feature
Feature: Testing the function get_mod from main # FEATURE_BDD.feature:1
Scenario: Find the remainder of 14 divided by 13 # FEATURE_BDD.feature:2
Given I put values [14, 13] into the function # steps/TEST_BDD.py:5
Then I get 1 # steps/TEST_BDD.py:11
Scenario: Find the remainder of 121 divided by 2 # FEATURE_BDD.feature:6
Given I put values [121, 2] into the function # steps/TEST_BDD.py:5
Then I get 1 # steps/TEST_BDD.py:11
Scenario: Find the remainder of 21 divided by 9 # FEATURE_BDD.feature:10
Given I put values [21, 9] into the function # steps/TEST_BDD.py:5
Then I get 3 # steps/TEST_BDD.py:11
1 feature passed, 0 failed, 0 skipped
3 scenarios passed, 0 failed, 0 skipped
6 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.003s
(hw_env) C:\Users\mansu\Downloads\BKIT-main\sem_3\laboratory_works\hw>


Windows PowerShell
(hw_env) C:\Users\mansu\Downloads\BKIT-main\sem_3\laboratory_works\hw>python3 TEST_TDD.py
.

Ran 1 test in 0.000s
OK
(hw_env) C:\Users\mansu\Downloads\BKIT-main\sem_3\laboratory_works\hw>