**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчёт по домашней работе

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-31Б |  | преподаватель каф. ИУ5 |
| Шимолина Полина  Кирилловна |  | Гапанюк Юрий Евгеньевич |
|  |  |  |
| Подпись:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Дата:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | Подпись:\_\_\_\_\_\_\_\_\_\_\_\_\_  Дата:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  |  |
|  |  |  |

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).

main.py

import telebot  
from telebot import types  
from enum import Enum  
from vedis import Vedis  
  
  
def degree(a, b):  
 return a \*\* b  
  
  
def summ(a, b):  
 return a + b  
  
  
def multiply(a, b):  
 return float(a \* b)  
  
  
def istrue(a):  
 if a:  
 return True  
 else:  
 return False  
  
  
bot = telebot.TeleBot('5015036432:AAEUYmBb9Pej2po-zdrB07GnVkqx8cJ0trs')  
  
db\_file = "db.vdb"  
CURRENT\_STATE = "CURRENT\_STATE"  
  
  
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"  
  
  
# получить  
def get(key):  
 with Vedis(db\_file) as db:  
 try:  
 return db[key].decode()  
 except KeyError:  
 return States.S\_START.value  
  
  
# записать  
def set(key, value):  
 with Vedis(db\_file) as db:  
 try:  
 db[key] = value  
 return True  
 except:  
 return False  
  
  
# ключ  
def make\_key(chatid, keyid):  
 res = str(chatid) + '\_\_' + str(keyid)  
 return res  
  
  
@bot.message\_handler(commands=['start'])  
def cmd\_start(message):  
 bot.send\_message(message.chat.id, 'Я - бот-калькулятор! \_-\_')  
 bot.send\_message(message.chat.id, 'Сначала введите 2 числа, потом действие')  
 set(make\_key(message.chat.id, CURRENT\_STATE), States.STATE\_FIRST\_NUM.value)  
 bot.send\_message(message.chat.id, 'Число:')  
  
  
@bot.message\_handler(commands=['reset'])  
def cmd\_reset(message):  
 bot.send\_message(message.chat.id, 'Сброс')  
 set(make\_key(message.chat.id, CURRENT\_STATE), States.STATE\_FIRST\_NUM.value)  
  
  
@bot.message\_handler(func=lambda message: get(make\_key(message.chat.id, CURRENT\_STATE)) == States.STATE\_FIRST\_NUM.value)  
def first\_num(message):  
 text = message.text  
 if not text.replace('.', "", 1).isdigit():  
 bot.send\_message(message.chat.id, '-\_-')  
 return  
 else:  
 set(make\_key(message.chat.id, CURRENT\_STATE), States.STATE\_SECOND\_NUM.value)  
 set(make\_key(message.chat.id, States.STATE\_FIRST\_NUM.value), text)  
 bot.send\_message(message.chat.id, 'Число:')  
  
  
@bot.message\_handler(  
 func=lambda message: get(make\_key(message.chat.id, CURRENT\_STATE)) == States.STATE\_SECOND\_NUM.value)  
def second\_num(message):  
 text = message.text  
 if not text.replace('.', "", 1).isdigit():  
 bot.send\_message(message.chat.id, '-\_-')  
 return  
 else:  
 set(make\_key(message.chat.id, CURRENT\_STATE), States.STATE\_OPERATION.value)  
 set(make\_key(message.chat.id, States.STATE\_SECOND\_NUM.value), text)  
 markup = types.ReplyKeyboardMarkup(row\_width=2)  
 b1 = types.KeyboardButton('^')  
 b2 = types.KeyboardButton('√')  
 b3 = types.KeyboardButton('+')  
 b4 = types.KeyboardButton('-')  
 b5 = types.KeyboardButton('\*')  
 b6 = types.KeyboardButton('/')  
 markup.add(b1, b2, b3, b4, b5, b6)  
 bot.send\_message(message.chat.id, 'Действие:', reply\_markup=markup)  
  
  
@bot.message\_handler(func=lambda message: get(make\_key(message.chat.id, CURRENT\_STATE)) == States.STATE\_OPERATION.value)  
def operation(message):  
 # Текущее действие  
 op = message.text  
 fv1 = float(get(make\_key(message.chat.id, States.STATE\_FIRST\_NUM.value)))  
 fv2 = float(get(make\_key(message.chat.id, States.STATE\_SECOND\_NUM.value)))  
 res = 0  
 if op == '^':  
 res = degree(fv1, fv2)  
 elif op == '√':  
 res = degree(fv1, 1 / fv2)  
 elif op == '+':  
 res = summ(fv1, fv2)  
 elif op == '-':  
 res = summ(fv1, -fv2)  
 elif op == '\*':  
 res = multiply(fv1, fv2)  
 elif op == '/':  
 res = multiply(fv1, 1 / fv2)  
 markup = types.ReplyKeyboardRemove(selective=False)  
 if op == '√':  
 bot.send\_message(message.chat.id,  
 f'{get(make\_key(message.chat.id, States.STATE\_SECOND\_NUM.value))}{op}{get(make\_key(message.chat.id, States.STATE\_FIRST\_NUM.value))}={str(res)}',  
 reply\_markup=markup)  
 elif op == '^' or op == '+' or op == '-' or op == '\*' or op == '/':  
 bot.send\_message(message.chat.id,  
 f'{get(make\_key(message.chat.id, States.STATE\_FIRST\_NUM.value))}{op}{get(make\_key(message.chat.id, States.STATE\_SECOND\_NUM.value))}={str(res)}',  
 reply\_markup=markup)  
  
 set(make\_key(message.chat.id, CURRENT\_STATE), States.STATE\_FIRST\_NUM.value)  
 bot.send\_message(message.chat.id, 'Число:')  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 bot.infinity\_polling()

steps.py

from main import \*  
from behave import \*  
  
  
  
@given('I have {a} \* {b}')  
def step\_impl(context, a, b):  
 context.a = float(a)  
 context.b = float(b)  
  
  
@when('I solve this math problem')  
def step\_impl(context):  
 context.answer = multiply(context.a, context.b)  
  
  
@then('I expect to get the answer: {answer}')  
def step\_impl(context, answer):  
 res = float(answer)  
 assert context.answer == res, f'{context.answer} is not {res}'  
  
@given('I have {a} + {b}')  
def step\_impl(context, a, b):  
 context.a = float(a)  
 context.b = float(b)  
  
  
@when('I solve the math problem')  
def step\_impl(context):  
 context.answer = summ(context.a, context.b)  
  
  
@then('I expect the answer: {answer}')  
def step\_impl(context, answer):  
 res = float(answer)  
 assert context.answer == res, f'{context.answer} is not {res}'

test.feature

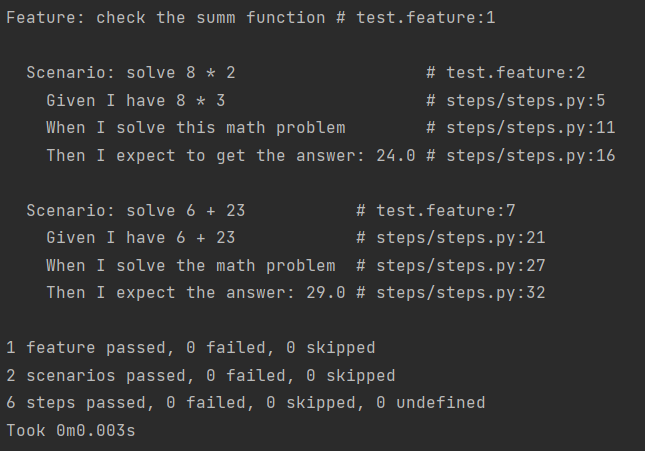
Feature: check the summ function  
 Scenario: solve 8 \* 2  
 Given I have 8 \* 3  
 When I solve this math problem  
 Then I expect to get the answer: 24.0  
  
 Scenario: solve 6 + 23  
 Given I have 6 + 23  
 When I solve the math problem  
 Then I expect the answer: 29.0

TDD.py

import lab6  
import unittest  
from unittest.mock import patch  
  
  
class Tests(unittest.TestCase):  
 def test\_1(self):  
 self.assertEqual(3.0, lab6.degree(9, 1 / 2))  
  
 def test\_2(self):  
 self.assertEqual(-229.0, lab6.summ(6, -235))

Экранные формы:

-m behave



TDD

