**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»

Отчет по домашнему заданию

Выполнил: студент группы ИУ5-32Б
Овчинников Данила Алексеевич

Москва, 2021 г.

## Описание задания:

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 bot2.py:

```
from aiogram import Bot, types
from aiogram.dispatcher import Dispatcher
from aiogram.dispatcher.filters.state import StatesGroup,State
from aiogram.dispatcher.filters import Command
from aiogram.dispatcher.storage import FSMContext
from aiogram.utils import executor
from config import TOKEN
from aiogram.types import ReplyKeyboardRemove,ReplyKeyboardMarkup, KeyboardButton,
InlineKeyboardMarkup, InlineKeyboardButton,Message

from aiogram.contrib.fsm_storage.memory import MemoryStorage
from aiogram.contrib.middlewares.logging import LoggingMiddleware


bot = Bot(token=TOKEN)
dp = Dispatcher(bot, storage=MemoryStorage())
dp.middleware.setup(LoggingMiddleware())


class Test(StatesGroup):
    Q0 = State()
    Q1 = State()
    Q2 = State()
    Q3 = State()

milk_price={
    "Молоко 3,2%" : 65,
    "Йогурт" : 25,
    "Сметана 20%" : 70
}
meat_fish_price={
    "Курина грудка" : 250,
    "Форель копчёная" : 570,
    "Говяжий фарш" : 450
}
bread_price={
    "Батон белого" : 35,
```

```
    "Пол чёрного" : 20,
    "Пирожёк с повидлом" : 35
}

milk={
    1 : "Молоко 3,2%",
    2 : "Йогурт",
    3 : "Сметана 20%"
}
meat_fish={
    1 : "Курина грудка",
    2 : "Форель копчёная",
    3 : "Говяжий фарш"
}
bread={
    1 : "Батон белого",
    2 : "Пол чёрного",
    3 : "Пирожёк с повидлом"
}



def
summary(first_table,first_table_price,second_table,second_table_price,third_table,thi
rd_table_price,first_answer,second_answer,third_answer):
    a = first_table_price[first_table[first_answer]]
    b = second_table_price[second_table[second_answer]]
    c = third_table_price[third_table[third_answer]]
    return "Итоговая сумма = " + str(a+b+c)+ "\n"


@dp.message_handler(state="*", commands=['start'])
async def starting_process(message: types.Message):
    await bot.send_message(message.from_user.id,"Приветствуем вас в нашем интернет
магазине.\nЗдесь вы можете заказать продукты из следующих категорий:\n \
    1)Молочные продукты\n \
    2)Мясо и рыба\n \
    3)Хлебобулочные изделия\nЧтобы начать формировать заказ напиши /order")
    await Test.Q0.set()

@dp.message_handler(state=Test.Q0, commands=['order'])
async def starting_process(message: types.Message,state: FSMContext):
    await bot.send_message(message.from_user.id, "Выберите молочный продукт\n1)Молоко
3,2% - 65\n2)Йогурт - 25\n3)Сметана 20% - 70")
    await Test.Q1.set()

@dp.message_handler(state=Test.Q1)
async def first_choosing(message: types.Message,state: FSMContext):
    answer = int(message.text)
    if (answer != 1 and answer !=2 and answer !=3):
```

```
        return await bot.send_message(message.from_user.id,"К сожалению, такого
товара нет в наличии, попробуйте выбрать другой")
    await state.update_data(q1 = answer)
    await bot.send_message(message.from_user.id,"Мясо или рыба\n1)Курина грудка -
250\n2)Форель копчёная - 570\n3)Говяжий фарш - 450")
    await Test.Q2.set()

@dp.message_handler(state=Test.Q2)
async def second_choosing(message: types.Message,state: FSMContext):
    answer = int(message.text)
    if (answer != 1 and answer !=2 and answer !=3):
        return await bot.send_message(message.from_user.id,"К сожалению, такого
товара нет в наличии, попробуйте выбрать другой")
    await state.update_data(q2 = answer)
    await bot.send_message(message.from_user.id, "Выберите хлебобулочное
изделие\n1)Батон белого - 35\n2)Пол чёрного - 20\n3)Пирожёк с повидлом - 35")
    await Test.Q3.set()



@dp.message_handler(state=Test.Q3)
async def third_choosing(message: types.Message,state: FSMContext):
    answer = int(message.text)
    if (answer != 1 and answer !=2 and answer !=3):
        return await bot.send_message(message.from_user.id,"К сожалению, такого
товара нет в наличии, попробуйте выбрать другой")
    await state.update_data(q3 = answer)
    data = await state.get_data()
    sumcheck=summary(milk,milk_price,meat_fish,meat_fish_price,bread,bread_price,data
.get("q1"),data.get("q2"),data.get("q3"))
    await bot.send_message(message.from_user.id,"Ваш заказ:\nМолочный
продукт:\n{}\nМясо или рыба:\n{}\nХлебобулочное
изделие:\n{}".format(milk[data.get("q1")],meat_fish[data.get("q2")],bread[data.get("q
3")]))
    await bot.send_message(message.from_user.id, sumcheck)
    await Test.Q0.set()


async def shutdown(dispatcher: Dispatcher):
    await dispatcher.storage.close()
    await dispatcher.storage.wait_closed()

if __name__ == '__main__':
    executor.start_polling(dp, on_shutdown=shutdown)
```

Файл confing.py:

```
TOKEN="2147385354:AAFVUQNoImK6S9SBafAmpa6zvqF66P9gVKM"
```

Файл test1.py:

```
import unittest
import sys, os

sys.path.append(os.getcwd())
from bot2 import *

test1 = 1
test2 = 2
test3 = 3

class TestGetRoots(unittest.TestCase):
    def test1_bot(self):
        res =
summary(milk,milk_price,meat_fish,meat_fish_price,bread,bread_price,test1,test2,test3
)
        self.assertEqual("Итоговая сумма = 670\n", res)
    def test2_bot(self):
        res =
summary(milk,milk_price,meat_fish,meat_fish_price,bread,bread_price,test2,test1,test3
)
        self.assertEqual("Итоговая сумма = 310\n", res)

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

Файл test2.py:

```
from behave import Given,When,Then
from bot2 import *

@Given("ordering food with answers in bot milk product - {a} meat and fish - {b}
bakery product - {c}")
def given_answers(context,a,b,c):
    context.ans1=int(a)
    context.ans2=int(b)
    context.ans3=int(c)

@When("we form summary of check")
def make_summary(context):
    res =
summary(milk,milk_price,meat_fish,meat_fish_price,bread,bread_price,context.ans1,cont
ext.ans2,context.ans3)
    context.result=res

@Then("check should be with correct price {d}")
def compare_results(context,d):
```

```
    corres= "Итоговая сумма = " + str(d)+ "\n"
    assert(context.result == corres )
```

Файл test2.feature:

```
Feature: Test summary
    Scenario: test summary for making check with 1 2 3
        Given ordering food with answers in bot milk product - 1 meat and fish - 2
bakery product - 3
        When we form summary of check
        Then check should be with correct price 670
    Scenario: test summary for making check with 2 1 3
        Given ordering food with answers in bot milk product - 2 meat and fish - 1
bakery product - 3
        When we form summary of check
        Then check should be with correct price 310
```

## Примеры выполнения программы:

### TDD:

```
PS C:\Users\J4ngle\Desktop\DZ> & c:/Users/J4ngle/Desktop/DZ/.venv/Scripts/Activate.ps1
(.venv) PS C:\Users\J4ngle\Desktop\DZ> & c:/Users/J4ngle/Desktop/DZ/.venv/Scripts/python.exe c:/Users/J4ngle/Desktop/DZ/test1.py
..
----------------------------------------------------------------------
Ran 2 tests in 0.000s

OK
```

### BDD:

```
(.venv) PS C:\Users\J4ngle\Desktop\DZ> cd features
(.venv) PS C:\Users\J4ngle\Desktop\DZ\features> behave
Feature: Test summary # test2.feature:1

  Scenario: test summary for making check with 1 2 3                                  # test2.feature:2
    Given ordering food with answers in bot milk product - 1 meat and fish - 2 bakery product - 3 # ../steps/test2.py:4
    When we form summary of check                                                     # ../steps/test2.py:10
    Then check should be with correct price 670                                       # ../steps/test2.py:15

  Scenario: test summary for making check with 2 1 3                                  # test2.feature:6
    Given ordering food with answers in bot milk product - 2 meat and fish - 1 bakery product - 3 # ../steps/test2.py:4
    When we form summary of check                                                     # ../steps/test2.py:10
    Then check should be with correct price 310                                       # ../steps/test2.py:15

1 feature passed, 0 failed, 0 skipped
2 scenarios passed, 0 failed, 0 skipped
6 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.002s
```