**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и системы управления»

Кафедра ИУ5. Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №5

«Модульное тестирование в Python»

|  |  |  |
| --- | --- | --- |
| Выполнил:  Макеева Е. А.  ИУ5-31Б |  | Проверил:  Нардид А. Н. |
| Подпись и дата: |  | Подпись и дата: |

Постановка задачи

Выберите любой фрагмент кода из лабораторных работ 1 или 2 или 3-4.

Модифицируйте код таким образом, чтобы он был пригоден для модульного тестирования.

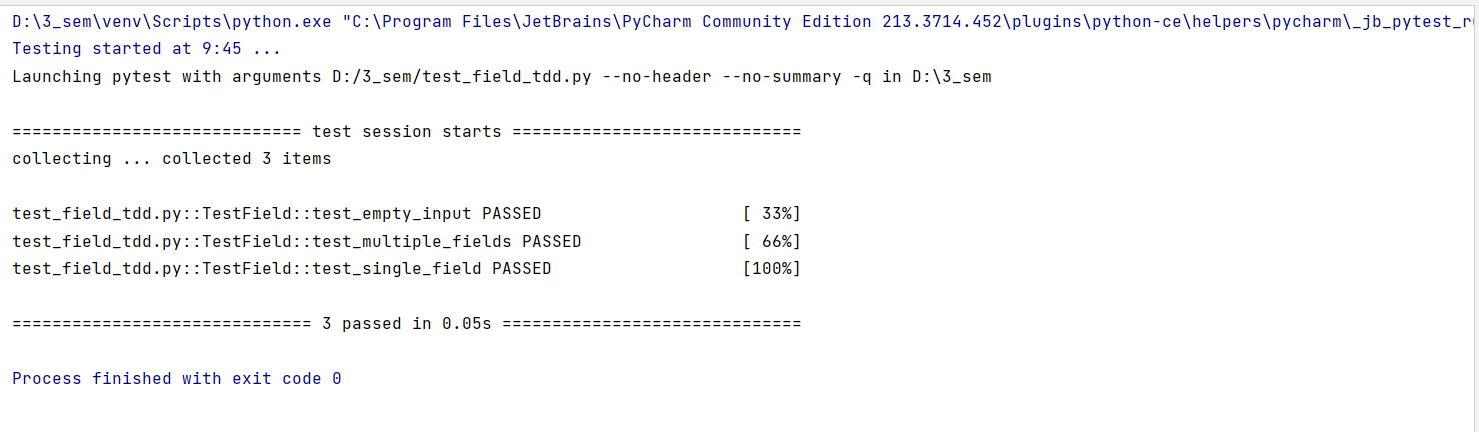
Разработайте модульные тесты. В модульных тестах необходимо применить следующие технологии:

TDD - фреймворк (не менее 3 тестов).

BDD - фреймворк (не менее 3 тестов).

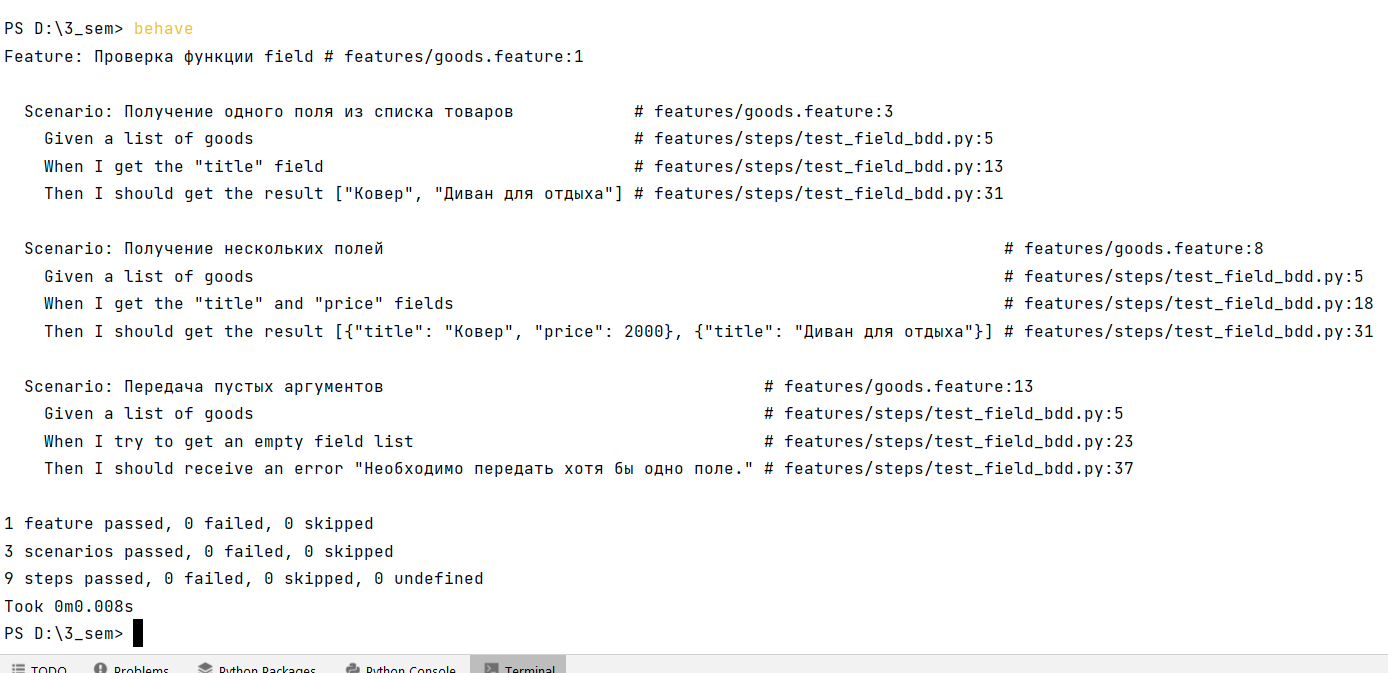
TDD - фреймворк

import unittest  
from field\_module import field  
  
  
class TestField(unittest.TestCase):  
  
 def test\_single\_field(self):  
 items = [{'title': 'Ковер', 'price': 2000, 'color': 'green'},  
 {'title': 'Диван для отдыха', 'color': 'black'}]  
 result = list(field(items, 'title'))  
 self.assertEqual(result, ['Ковер', 'Диван для отдыха'])  
  
 def test\_multiple\_fields(self):  
 items = [{'title': 'Ковер', 'price': 2000, 'color': 'green'},  
 {'title': 'Диван для отдыха', 'color': 'black'}]  
 result = list(field(items, 'title', 'price'))  
 self.assertEqual(result, [{'title': 'Ковер', 'price': 2000}, {'title': 'Диван для отдыха'}])  
  
 def test\_empty\_input(self):  
 items = []  
 result = list(field(items, 'title'))  
 self.assertEqual(result, [])  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()



BDD - фреймворк

from behave import given, when, then  
from field\_module import field  
  
  
@given('a list of goods')  
def step\_given\_list\_of\_goods(context):  
 context.goods = [  
 {'title': 'Ковер', 'price': 2000, 'color': 'green'},  
 {'title': 'Диван для отдыха', 'color': 'black'}  
 ]  
  
  
@when('I get the "{field\_name}" field')  
def step\_when\_get\_single\_field(context, field\_name):  
 context.result = list(field(context.goods, field\_name))  
  
  
@when('I get the "{field1}" and "{field2}" fields')  
def step\_when\_get\_multiple\_fields(context, field1, field2):  
 context.result = list(field(context.goods, field1, field2))  
  
  
@when('I try to get an empty field list')  
def step\_when\_try\_empty\_field\_list(context):  
 try:  
 context.result = list(field(context.goods))  
 except AssertionError as e:  
 context.result = str(e)  
  
  
@then('I should get the result {expected\_result}')  
def step\_then\_check\_result(context, expected\_result):  
 expected\_result = eval(expected\_result)  
 assert context.result == expected\_result  
  
  
@then('I should receive an error "{error\_message}"')  
def step\_then\_check\_error(context, error\_message):  
 assert context.result == error\_message

****