



**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 «Системы обработки информации и
управления»**

Лабораторная работа №5 по курсу
«Базовые компоненты интернет технологий»

Выполнила:
студент группы № ИУ5-33Б
Балюк А.В

Проверил:
Преподаватель
Гапанюк Ю.Е

2022 г.

Задание:

1. Выберите любой фрагмент кода из лабораторных работ 1 или 2 или 3-4.
2. Модифицируйте код таким образом, чтобы он был пригоден для модульного тестирования.
3. Разработайте модульные тесты. В модульных тестах необходимо применить следующие технологии:
 - TDD - фреймворк (не менее 3 тестов).
 - BDD - фреймворк (не менее 3 тестов).
 - Создание Mock-объектов (необязательное дополнительное задание).

Листинг TDD – тестов (unittest):

```
import unittest
from lab_python_fp.field import field
from lab_python_fp.sort import sort_1, sort_2

class Test_field(unittest.TestCase):
    def setUp(self):
        self.goods=[{'title': 'Окно', 'color': 'white'},
                     {'title': 'Шторы', 'price': int(1e9), 'color': '', 'name': '',
                      'addText': 'из будущего'}]
        self.data = [4, -30, 30, 100, -100, 123, 1, 0, -1, -4]

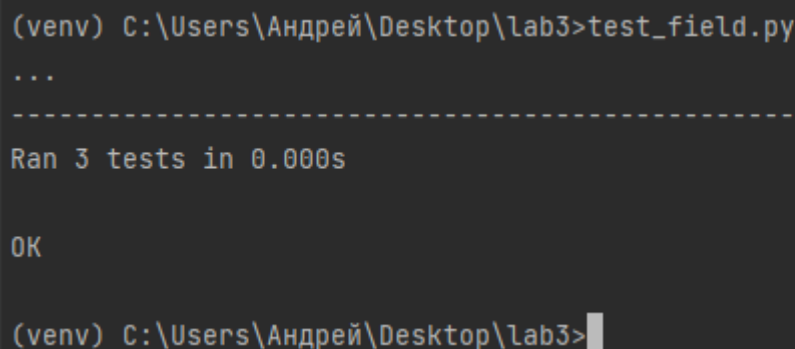
    def test1(self):
        test1=list(field(self.goods,'title'))
        correct = ['Окно', 'Шторы']
        self.assertEqual(test1, correct)

    def test2(self):
        test2 = list(field(self.goods, 'title', 'price'))
        correct = [{'title':'Окно'}, {'title': 'Шторы', 'price': int(1e9)}]
        self.assertEqual(test2, correct)

    def test_sort(self):
        self.assertEqual(sort_1(self.data), [123, 100, -100, -30, 30, 4, -4,
1, -1, 0])

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

Результат:



```
(venv) C:\Users\Андрей\Desktop\lab3>test_field.py
...
-----
Ran 3 tests in 0.000s

OK

(venv) C:\Users\Андрей\Desktop\lab3>
```

Листинг BDD – тестов (behave):

```
from behave import given, when, then, step
from lab_python_fp.unique import Unique

@given('We have list of string [{data}]')
def step_impl(context,data):
    context.data = data.split(', ')

@given('We have generator of integers [{data}]')
def step_impl(context,data):
    context.data = (int(elem) for elem in data.split(', '))

@given('We have list of integers [{data}]')
def step_impl(context,data):
    context.data = [int(elem) for elem in data.split(', ')]

@when('We run Unique()')
def step_impl(context):
    context.result = list(Unique(context.data))
```

```

@when('We run Unique() with ignore_case')
def step_impl(context):
    context.result = list(Unique(context.data, ignore_case = True))

@then('We must have string [{correct}]')
def step_impl(context, correct):
    context.correct = correct.split(', ')
    assert sorted(context.result) == sorted(context.correct)

@then('We must have [{correct}]')
def step_impl(context, correct):
    context.correct = []
    for i in correct.split(', '):
        context.correct.append(int(i))
    assert context.result == context.correct

```

Файл .feature:

```

Feature: Unique_test

    Scenario: List of integer
        Given We have list of integers [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]
        When We run Unique()
        Then We must have [1, 2]

    Scenario: List of generator of integers
        Given We have generator of integers [1, 2, 3, 1, 2, 1, 2, 3, 3]
        When We run Unique()
        Then We must have [1, 2, 3]

    Scenario: List of string
        Given We have list of string ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']
        When We run Unique()
        Then We must have string ['a', 'A', 'b', 'B']

    Scenario: List of string ignore case
        Given We have list of string ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']
        When We run Unique() with ignore_case
        Then We must have string ['a', 'b']

```

Результат:

```

PS C:\Users\Андрей\Desktop\lab3> behave
Feature: Unique_test # features/test_lab5.feature:1

    Scenario: List of integer                                     # features/test_lab5.feature:3
        Given We have list of integers [1, 1, 1, 1, 1, 2, 2, 2, 2, 2] # features/steps/unique_test.py:12
        When We run Unique()                                         # features/steps/unique_test.py:16
        Then We must have [1, 2]                                     # features/steps/unique_test.py:29

    Scenario: List of generator of integers                       # features/test_lab5.feature:7
        Given We have generator of integers [1, 2, 3, 1, 2, 1, 2, 3, 3] # features/steps/unique_test.py:8
        When We run Unique()                                         # features/steps/unique_test.py:16
        Then We must have [1, 2, 3]                                  # features/steps/unique_test.py:29

    Scenario: List of string                                     # features/test_lab5.feature:11
        Given We have list of string ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B'] # features/steps/unique_test.py:4
        When We run Unique()                                         # features/steps/unique_test.py:16
        Then We must have string ['a', 'A', 'b', 'B']               # features/steps/unique_test.py:24

    Scenario: List of string ignore case                         # features/test_lab5.feature:15
        Given We have list of string ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B'] # features/steps/unique_test.py:4
        When We run Unique() with ignore_case                       # features/steps/unique_test.py:20
        Then We must have string ['a', 'b']                         # features/steps/unique_test.py:24

1 feature passed, 0 failed, 0 skipped
4 scenarios passed, 0 failed, 0 skipped
12 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.003s

```