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 лабораторной работе №5
«Модульное тестирование в Python.»

Выполнил:
студент группы ИУ5-34Б:
Такташова Дарья Юрьевна
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю.Е
Подпись и дата:

Москва, 2022 г.

Задание:

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

1. Модифицируйте код таким образом, чтобы он был пригоден для модульного тестирования.
2. 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).

Код программы:
1_pytest/test_field/test_field.py

1_pytest/test_sort/test_sort.py

1_pytests > test_sort > test_sort.py > ...

```python
import pytest

def my_sort1(data):
    return sorted(data, key = abs, reverse = True)

def my_sort2(data):
    return sorted(data, key = lambda n: -abs(n))

def test_my_sort1():
    data = [4, -30, 100, -100, 123, 1, 0, -1, -4]
    res = [123, 100, -100, -30, 4, -4, 1, -1, 0]
    assert res == my_sort1(data)

def test_my_sort2():
    data = [4, -30, 100, -100, 123, 1, 0, -1, -4]
    res = [123, 100, -100, -30, 4, -4, 1, -1, 0]
    assert res == my_sort2(data)
```

1_pytest/test_unique/test_unique.py

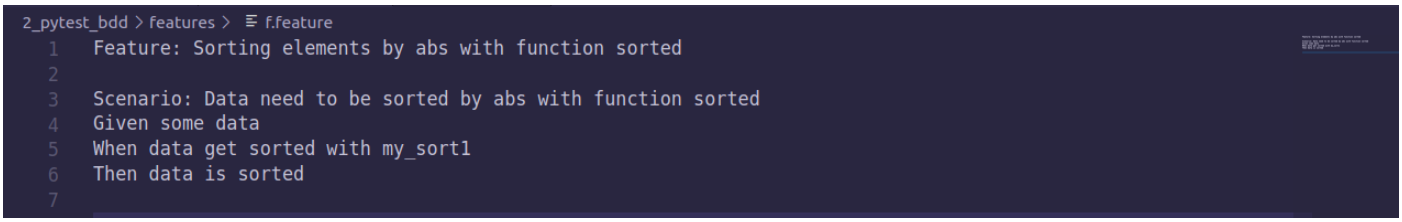
1_pytests > test_unique > test_unique.py > ...

```python
import pytest
import gen_random

class Unique(object):
    def __init__(self, items, **kwargs):
        self.seen = []
        for i in items:
            if  len(kwargs) > 0 and kwargs["ignore_case"]:
                flag = True
                for j in self.seen:
                    if j.lower() == i.lower():
                        flag = False
                if flag:
                    (self.seen).append(i)
            else:
                if i in self.seen:
                    continue
                self.seen.append(i)

    def __next__(self):
        if len(self.seen) == 0:
            raise StopIteration
        item = self.seen[0]
        del self.seen[0]
        return item

    def __iter__(self):
        return self

@pytest.fixture
def data():
    d = ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']
    return d

def test_unique_ignore_case_True(data):
    t = list(Unique(data, ignore_case = True))
    res = ['a', 'b']
    assert res == t

def test_unique_ignore_case_False(data):
    t = list(Unique(data))
    res = ['a', 'A', 'b', 'B']
    assert res == t

def test_unique_numbers():
    data = [1,1,1,1,1,1,2,2,2,2,2,2]
    t = list(Unique(data))
    res = [1, 2]
    assert res == t

```

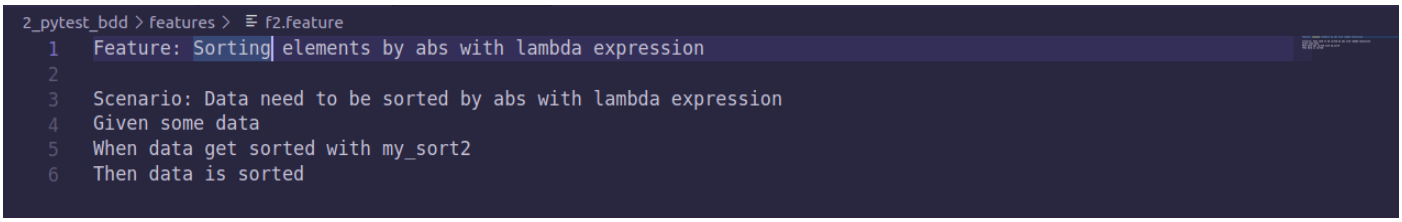
Пример выполнения:

```
dasha@dasha-BOHB-WAX9:~/Desktop/BIKT/BIKT/1_pytests$ python3 -m pytest
============================================= test session starts ==============================================
platform linux -- Python 3.10.6, pytest-7.2.0, pluggy-1.0.0
rootdir: /home/dasha/Desktop/BIKT/BIKT/1_pytests
plugins: bdd-6.1.0, anyio-3.6.2
collected 7 items

test_field/test_field.py ..                                                                          [ 28%]
test_sort/test_sort.py ..                                                                            [ 57%]
test_unique/test_unique.py ...                                                                       [100%]

============================================== 7 passed in 0.04s ===============================================
```

2_pytest_bdd/features:
f.feature



f2.feature



f3.feature

2_pytest_bdd/steps:

test_sort.py

2_pytest_bdd > steps > test_sort.py > ...

```python
from func import my_sort1, my_sort2
import pytest
from pytest_bdd import scenario, given, when, then

@scenario('/home/dasha/Desktop/BIKT/BIKT/2_pytest_bdd/features/f.feature','Data need to be sorted by abs with function sorted')
def testing_my_sort1():
    pass

@scenario('/home/dasha/Desktop/BIKT/BIKT/2_pytest_bdd/features/f2.feature','Data need to be sorted by abs with lambda expression')
def testing_my_sort2():
    pass

@given('some data', target_fixture= 'data')
def data():
    return [4, -30, 100, -100, 123, 1, 0, -1, -4]

@when('data get sorted with my_sort1', target_fixture='sorted_data')
def sorted_data(data):
    return my_sort1(data)

@then('data is sorted')
def sorted_data(sorted_data):
    assert sorted_data == [123, 100, -100, -30, 4, -4, 1, -1, 0]

@when('data get sorted with my_sort2', target_fixture='sorted_data')
def sorted_data(data):
    return my_sort2(data)
```

test_unique.py

2_pytest_bdd > steps > test_unique.py > ...

```python
from func import Unique
import pytest
from pytest_bdd import scenario, given, when, then

@scenario('/home/dasha/Desktop/BIKT/BIKT/2_pytest_bdd/features/f3.feature','Getting list of unique elements')
def testing_unique():
    pass

@given('some data', target_fixture= 'data')
def data():
    return ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']

@when('data is getting unique', target_fixture='unique_data')
def unique_data(data):
    return list(Unique(data, ignore_case = True))

@then('data is unique')
def res_data(unique_data):
    assert unique_data == ['a', 'b']
```

Пример выполнения:

```
dasha@dasha-BOHB-WAX9:~/Desktop/BIKT/BIKT/2_pytest_bdd$ python3 -m pytest
============================= test session starts ==============================
platform linux -- Python 3.10.6, pytest-7.2.0, pluggy-1.0.0
rootdir: /home/dasha/Desktop/BIKT/BIKT/2_pytest_bdd
plugins: bdd-6.1.0, anyio-3.6.2
collected 3 items

steps/test_sort.py ..                                                    [ 66%]
steps/test_unique.py .                                                   [100%]

============================== 3 passed in 0.02s ===============================
```

–