

Северо-Кавказский федеральный университет  
Институт математики и информационных технологий

**ОТЧЕТ**  
**о выполнении лабораторной работы №9**  
**по дисциплине**  
**«Основы Программной Инженерии»**

Выполнил:

**Ботвинкин Никита Сергеевич**

---

студент 2 курса, ПИЖ-б-о-21-1 группы  
бакалавриата «Программная инженерия»  
очной формы обучения

---

**Ставрополь, 2023**

## СКРИНШОТЫ ВЫПОЛНЕНИЯ ИНДИВИДУАЛЬНОГО

```
PS C:\git\2_9lab> python test_runner.py
Class start
=====
Создание базы данных...
Количество записей в базу данных: 7
{'student_name': 'Егор Шульга', 'student_group': 'ИВТ', 'marks_list': '[4 3 3 3 4]'}
{'student_name': 'Данил Косарев', 'student_group': 'ИВТ', 'marks_list': '[4 3 3 3 3]'}
{'student_name': 'Андрей Шульга', 'student_group': 'ИСП', 'marks_list': '[2 2 4 2 2]'}
{'student_name': 'Никита Тагиев', 'student_group': 'ИСБ', 'marks_list': '[2 4 2 3 4]'}
{'student_name': 'Данил Ботвинкин', 'student_group': 'ИСП', 'marks_list': '[3 2 4 2 2]'}
{'student_name': 'Сергей Косарев', 'student_group': 'ИСП', 'marks_list': '[2 4 4 4 3]'}
{'student_name': 'Андрей Ботвинкин', 'student_group': 'ПИЖ', 'marks_list': '[4 4 4 3 4]'}
End for [Full test]
DB is deleted
.Создание базы данных...
Всего студентов: 10
Студенты с оценкой 2:
{'student_name': 'Никита Косарев', 'student_group': 'ИСБ', 'marks_list': '[4 2 4 4 2]'}
{'student_name': 'Егор Тагиев', 'student_group': 'ИСБ', 'marks_list': '[4 4 3 4 2]'}
{'student_name': 'Никита Ямбогло', 'student_group': 'ИСБ', 'marks_list': '[2 2 4 2 2]'}
{'student_name': 'Данил Косарев', 'student_group': 'ИВТ', 'marks_list': '[2 3 3 3 3]'}
{'student_name': 'Егор Шульга', 'student_group': 'ИСП', 'marks_list': '[3 4 2 3 2]'}
{'student_name': 'Андрей Косарев', 'student_group': 'ИВТ', 'marks_list': '[4 2 4 3 3]'}
{'student_name': 'Андрей Шульга', 'student_group': 'ИВТ', 'marks_list': '[3 3 3 2 4]'}
{'student_name': 'Никита Ботвинкин', 'student_group': 'ИВТ', 'marks_list': '[2 2 2 2 4]'}
Всего студентов с оценкой 2: 8
End for [Marks test]
DB is deleted
.=====
End

-----
Ran 2 tests in 0.175s

OK
errors
0
failures
0
skipped
0
testsRun
2
```

Рисунок 9.1 – Выполнение теста

```

PS C:\git\2_9lab> python test_runner.py
Class start
=====
Создание базы данных...
Количество записей в базу данных: 10
{'student_name': 'Сергей Косарев', 'student_group': 'ИСБ', 'marks_list': '[3 3 3 4 2]'}
{'student_name': 'Егор Шульга', 'student_group': 'ИСБ', 'marks_list': '[4 2 2 3 2]'}
{'student_name': 'Сергей Косарев', 'student_group': 'ИВТ', 'marks_list': '[2 2 2 4 2]'}
{'student_name': 'Сергей Косарев', 'student_group': 'ПИЖ', 'marks_list': '[3 4 4 2 3]'}
{'student_name': 'Данил Косарев', 'student_group': 'ПИЖ', 'marks_list': '[4 2 4 4 3]'}
{'student_name': 'Данил Ботвинкин', 'student_group': 'ПИЖ', 'marks_list': '[2 4 2 2 2]'}
{'student_name': 'Егор Ботвинкин', 'student_group': 'ИСБ', 'marks_list': '[2 3 3 2 3]'}
{'student_name': 'Данил Косарев', 'student_group': 'ИСП', 'marks_list': '[2 4 2 2 2]'}
{'student_name': 'Никита Ботвинкин', 'student_group': 'ИВТ', 'marks_list': '[3 3 4 2 2]'}
{'student_name': 'Андрей Ботвинкин', 'student_group': 'ИВТ', 'marks_list': '[3 2 4 3 2]'}
End for [Full test]
DB is deleted
.Создание базы данных...
Всего студентов: 18
Студенты с оценкой 2:
{'student_name': 'Сергей Ботвинкин', 'student_group': 'ИСБ', 'marks_list': '[2 3 2 2 4]'}
{'student_name': 'Егор Косарев', 'student_group': 'ИВТ', 'marks_list': '[4 3 2 2 3]'}
{'student_name': 'Егор Шульга', 'student_group': 'ИСП', 'marks_list': '[4 2 4 3 2]'}
{'student_name': 'Сергей Шульга', 'student_group': 'ИВТ', 'marks_list': '[3 4 2 4 2]'}
{'student_name': 'Егор Ямбогло', 'student_group': 'ИСП', 'marks_list': '[4 4 3 2 3]'}
{'student_name': 'Егор Ботвинкин', 'student_group': 'ИВТ', 'marks_list': '[2 4 2 2 3]'}
{'student_name': 'Егор Ямбогло', 'student_group': 'ИСБ', 'marks_list': '[3 2 2 3 2]'}
{'student_name': 'Андрей Косарев', 'student_group': 'ИСП', 'marks_list': '[4 2 2 4 2]'}
{'student_name': 'Сергей Косарев', 'student_group': 'ИВТ', 'marks_list': '[4 2 3 2 2]'}
{'student_name': 'Сергей Шульга', 'student_group': 'ИСП', 'marks_list': '[4 4 2 2 2]'}
{'student_name': 'Андрей Тагиев', 'student_group': 'ИСП', 'marks_list': '[3 2 3 2 2]'}
{'student_name': 'Егор Тагиев', 'student_group': 'ИВТ', 'marks_list': '[2 4 2 2 3]'}
{'student_name': 'Сергей Косарев', 'student_group': 'ИВТ', 'marks_list': '[4 2 4 4 2]'}
{'student_name': 'Никита Тагиев', 'student_group': 'ПИЖ', 'marks_list': '[2 3 3 3 3]'}
Всего студентов с оценкой 2: 14
End for [Marks test]
DB is deleted
.=====
End

-----
Ran 2 tests in 0.292s

OK
errors
0
failures
0
skipped
0
testsRun
2

```

Рисунок 9.2 – Другой пример

## ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Для чего используется автономное тестирование?

- Для тестирования функций, классов, методов и т.д. с целью выявления ошибок в работе в этих отдельных единицах общей программы.

2. Какие фреймворки Python получили наибольшее распространение для решения задач автономного тестирования?

- unittest, nose, pytest

3. Какие существуют основные структурные единицы модуля unittest?

- Test fixture, Test case, Test suite, Test runner

4. Какие существуют способы запуска тестов unittest?

- Запуск тестов можно сделать как из командной строки, так и с помощью графического интерфейса пользователя (GUI).

5. Каково назначение класса TestCase?

- Он представляет собой класс, который должен являться базовым для всех остальных классов, методы которых будут тестировать те или иные автономные единицы исходной программы.

6. Какие методы класса TestCase выполняются при запуске и завершении работы тестов?

- setUp(), tearDown()

7. Какие методы класса TestCase используются для проверки условий и генерации ошибок?

<code>assertEqual(a, b)</code>	<code>a == b</code>
<code>assertNotEqual(a, b)</code>	<code>a != b</code>
<code>assertTrue(x)</code>	<code>bool(x) is True</code>
<code>assertFalse(x)</code>	<code>bool(x) is False</code>
<code>assertIs(a, b)</code>	<code>a is b</code>
<code>assertIsNot(a, b)</code>	<code>a is not b</code>
<code>assertIsNone(x)</code>	<code>x is None</code>
<code>assertIsNotNone(x)</code>	<code>x is not None</code>
<code>assertIn(a, b)</code>	<code>a in b</code>
<code>assertNotIn(a, b)</code>	<code>a not in b</code>
<code>assertIsInstance(a, b)</code>	<code>isinstance(a, b)</code>
<code>assertNotIsInstance(a, b)</code>	<code>not isinstance(a, b)</code>

Рисунок 9.3 – Методы проверки

<code>assertRaises(exc, fun, *args, **kwargs)</code>	Функция <code>fun(*args, **kwargs)</code> вызывает исключение <code>exc</code>
<code>assertRaisesRegex(exc, r, fun, *args, **kwargs)</code>	Функция <code>fun(*args, **kwargs)</code> вызывает исключение <code>exc</code> , сообщение которого совпадает с регулярным выражением <code>r</code>
<code>assertWarns(warn, fun, *args, **kwargs)</code>	Функция <code>fun(*args, **kwargs)</code> выдает сообщение <code>warn</code>
<code>assertWarnsRegex(warn, r, fun, *args, **kwargs)</code>	Функция <code>fun(*args, **kwargs)</code> выдает сообщение <code>warn</code> и оно совпадает с регулярным выражением <code>r</code>

Рисунок 9.4 – Исключения

<code>assertAlmostEqual(a, b)</code>	<code>round(a-b, 7) == 0</code>
<code>assertNotAlmostEqual(a, b)</code>	<code>round(a-b, 7) != 0</code>
<code>assertGreater(a, b)</code>	<code>a &gt; b</code>
<code>assertGreaterEqual(a, b)</code>	<code>a &gt;= b</code>
<code>assertLess(a, b)</code>	<code>a &lt; b</code>
<code>assertLessEqual(a, b)</code>	<code>a &lt;= b</code>
<code>assertRegex(s, r)</code>	<code>r.search(s)</code>
<code>assertNotRegex(s, r)</code>	<code>not r.search(s)</code>
<code>assertCountEqual(a, b)</code>	<code>a и b имеют одинаковые элементы (порядок неважен)</code>

Рисунок 9.5 – Проверка различных ситуаций

8. Какие методы класса TestCase позволяют собирать информацию о самом тесте?

- countTestCases(), id(), shortDescription()

9. Каково назначение класса TestSuite? Как осуществляется загрузка тестов?

- Класс TestSuite используется для объединения тестов в группы, которые могут включать в себя как отдельные тесты, так и заранее созданные группы.

10. Каково назначение класса TestResult?

- Класс TestResult используется для сбора информации о результатах прохождения тестов.

11. Для чего может понадобиться пропуск отдельных тестов?

- Во избежание ошибок тестирования, так как некоторые тесты могут давать заведомо неправильный результат в зависимости от какого-либо условия.

12. Как выполняется безусловный и условных пропуск тестов? Как выполнить пропуск класса тестов?

- @unittest.skip(reason) – Безусловный пропуска
- @unittest.skipIf(condition, reason) – Условный
- @unittest.skipUnless(condition, reason) - Условный

13. Самостоятельно изучить средства по поддержке тестов unittest в PyCharm. Приведите обобщенный алгоритм проведения тестирования с помощью PyCharm.

- В PyCharm есть встроенная поддержка unit тестов, которая позволяет создавать шаблон класса для тестирования и его дальнейшей настройки.

1) Необходимо создать класс для тестирования.

2) Написание кода тестов в классе для тестирования частей программы.

3) Запуск тестов

4) Debug тестов при необходимости.

5) Автоматизация тестов. PyCharm поддерживает автоматизацию тестов – установив её, вы можете сфокусироваться в написании кода самой программы, а IDE будет в автоматическом режиме проводить тестирование по мере изменения кода.