Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

# ОТЧЕТ
# ПО ЛАБОРАТОРНОЙ РАБОТЕ №6
## дисциплины «Анализ данных»
## Вариант №22

Выполнил:
Михеева Елена Александровна
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика
и вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р.А., канд. техн. наук,
доцент, доцент кафедры
инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: Тестирование в Python [unitest].

Цель: приобрести навыки написания автоматизированных тестов на языке программирования Python версии 3.x.

Порядок выполнения работы:

1. Приступили к выполнению индивидуального задания №1. Модифицировано индивидуальное задание из работы №7 дисциплины “Анализ Данных”. Были добавлены тесты, позволяющие проверить операции по работе с базой данных SQLite3.

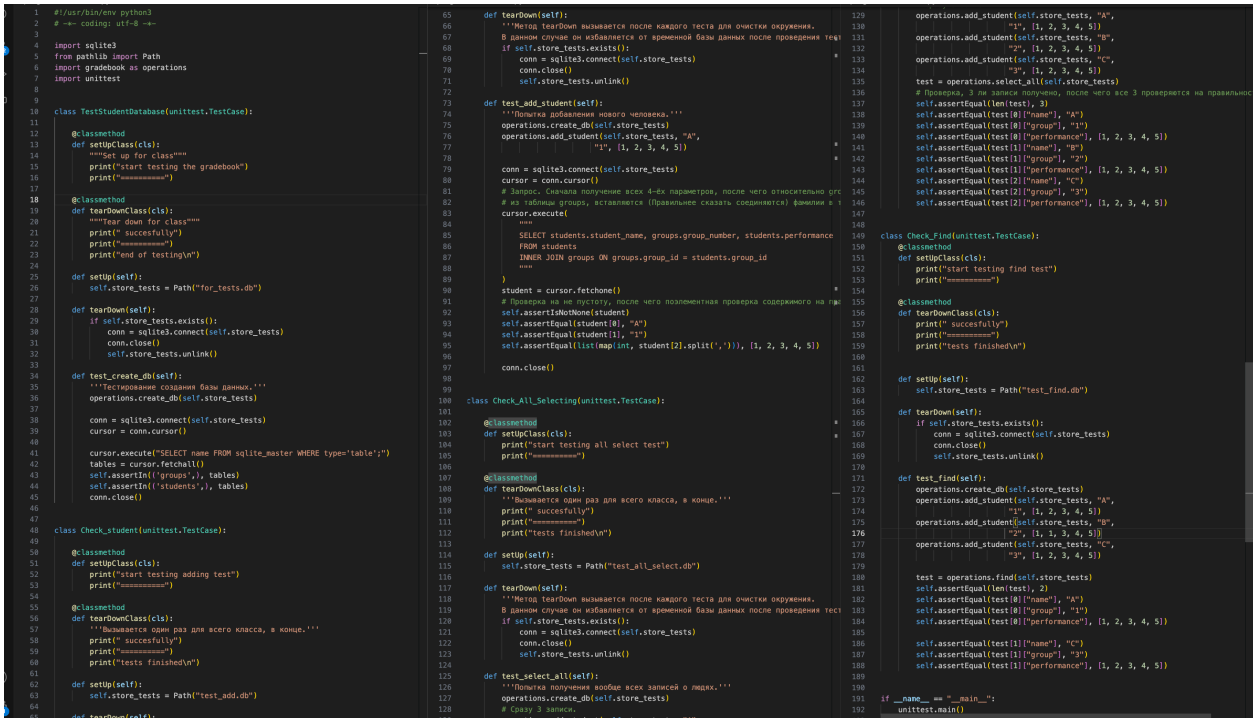

Рисунок 2. Код индивидуального задания №1

```
OK
(venv) (base) elenamiheeva@MacBook-Pro-Elena Data_Analysis_2.22 % python3 programs /test.py
start testing all select test
==========
. succesfully
==========
tests finished

start testing find test
==========
. succesfully
==========
tests finished

start testing adding test
==========
. succesfully
==========
tests finished

start testing the gradebook
==========
. succesfully
==========
end of testing


----------------------------------------------------------------------
Ran 4 tests in 0.044s

OK
```

Рисунок 3. Результат работы программы задания №1

Ответы на контрольные вопросы.

1) Для чего используется автономное тестирование?

**Ответ:** автономное тестирование используется для автоматизации проверки работоспособности программного обеспечения, позволяя выполнятьтесты без необходимости ручного вмешательства.

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

**Ответ:** unittest, pytest и nose – наиболее популярные фреймворки Python.

3) Какие существуют основные структурные единицы модуля unittest?

**Ответ:** основные структурные единицы модуля unittest – тестовые классы, методы тестирования (Начинаются на test_) и assertIn для проверки ожидаемых результатов.

4) Какие существуют способы запуска тестов unittest?

**Ответ:** тесты unittest можно запускать как из командной строки, так и из среды разработки (Visual Studio Code, PyCharm, Visual Studio и т.д.), а также можно использовать специализированные инструменты для автоматического запуска тестов (Cricket).

5) Каково назначение класса TestCase?

**Ответ:** класс TestCase определяет тестовые случаи, которые содержат методы для проверки конкретных аспектов работы программы.

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

**Ответ:** при запуске и завершении работы тестов запускаются методы setUp() и tearDown().

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

**Ответ:** методы assertIn() используются для проверки различных условий и генерации ошибок в случае несоответствия ожидаемых и получаемых результатов.

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

**Ответ:** методы setUp(), tearDown(), getName() и id() позволяют собирать информацию о самом тесте, его имени и идентификаторе.

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

**Ответ:** класс TestSuite группирует тестовые случаи в единую структуру. Загружаются тесты путём добавления тестовых случаев в объект TestSuite, который выполняется после этого.

10) Каково назначение класса TestResult?

**Ответ:** класс TestResult отслеживает результаты выполнения тестов. В эту информацию включены сведения о пройденных, проваленных и пропущенных тестах.

11) Для чего может понадобиться пропуск отдельных тестов?

**Ответ:** пропуск отдельных тестов может понадобиться, когда тестируемый код, временно недоступен или в процессе разработки\доработки.

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

**Ответ:** безусловный пропуск теста выполняется с помощью декоратора @unittest.skip(), а условный пропуск с помощью декоратора @unittest.skipIf(). Для пропуска класса тестов можно использовать декоратор @unittest.skip() перед определением класса.

13) Обобщённый алгоритм проведения тестирования с помощью PyCharm?

**Ответ:** обобщённый алгоритм проведения тестирования с помощью

PyCharm включает: создание тестовых случаев и их методов; настройка окружения тестирования; написание необходимых тестов (И их запуск через PyCharm); анализ результатов и исправление выявленных проблем.

**Вывод**: в ходе выполнения лабораторной работы, исследовано взаимодействие с базами данных SQLite3 с помощью языка программирования Python.