|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА**  **Российский технологический университет»**  **РТУ МИРЭА** |

## Институт информационных технологий (ИИТ)

## Кафедра [математического обеспечения и стандартизации](https://online-edu.mirea.ru/course/index.php?categoryid=54) [информационных технологий](https://online-edu.mirea.ru/course/index.php?categoryid=54) (МОСИТ)

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

по дисциплине «Тестирование и верификация программного обеспечения»

**Практическое занятие № 2**

|  |  |
| --- | --- |
| Студенты группы | *ИКБО-32-23, Нестеров Г. А. Осокин С. А.*  *Пошибайлов И. С*. |
|  | (подпись) |
| Преподаватель | *Ильичев Г. П.* |
|  | (подпись) |
| Отчет представлен | « » октября 2025 г. |

Москва 2025 г.

# Цель и задачи практической работы

Цель работы: познакомить студентов с процессом модульного и мутационного тестирования, включая разработку, проведение тестов, исправление ошибок, анализ тестового покрытия, а также оценку эффективности тестов путём применения методов мутационного тестирования.

Для достижения поставленной цели работы студентам необходимо выполнить ряд задач:

* изучить основы модульного тестирования и его основные принципы;
* освоить использование инструментов для модульного тестирования;
* разработать модульные тесты для программного продукта и проанализировать их покрытие кода;
* изучить основы мутационного тестирования и освоить инструменты
* для его выполнения (MutPy, PIT, Stryker);
* применить мутационное тестирование к программному продукту,
* оценить эффективность тестов;
* улучшить существующий набор тестов, ориентируясь на результаты
* мутационного тестирования;
* оформить итоговый отчёт с результатами проделанной работы.

**2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

**2.1. Основы модульного тестирования**

Модульное тестирование — метод тестирования программного обеспечения, при котором проверяются отдельные компоненты программы (функции, методы, классы) в изоляции от остальных частей системы.

Основные принципы:

Изолированность тестируемых компонентов

Детерминированность и воспроизводимость тестов

Проверка одной функциональности в одном тесте

Независимость тестов друг от друга

Оптимизация времени выполнения тестов

**2.2. Мутационное тестирование: принципы и методология**

Мутационное тестирование основано на создании модифицированных версий программы (мутантов) путем внесения небольших синтаксических изменений, имитирующих типичные ошибки программирования.

Основные понятия:

Мутант — модифицированная версия исходного кода

Убийство мутанта — успешное обнаружение изменения тестами

Выживший мутант — изменение, не обнаруженное тестами

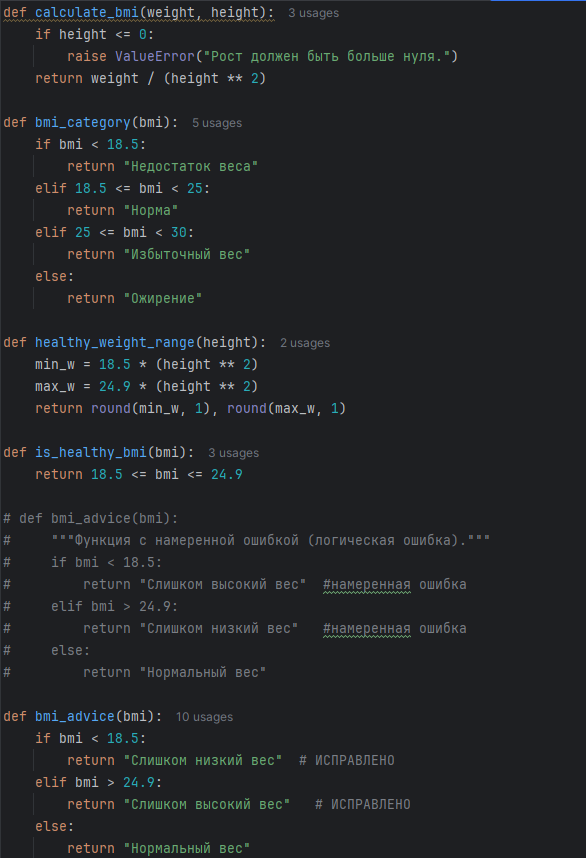
Score — процент убитых мутантов от общего количества

**3. РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ**

**3.1. Разработка программных модулей**

Разработаны три модуля:

**3.1.1. BMI Calculator (Пошибайлов И.С.)**



calculate\_bmi(weight, height) – расчет индекса массы тела

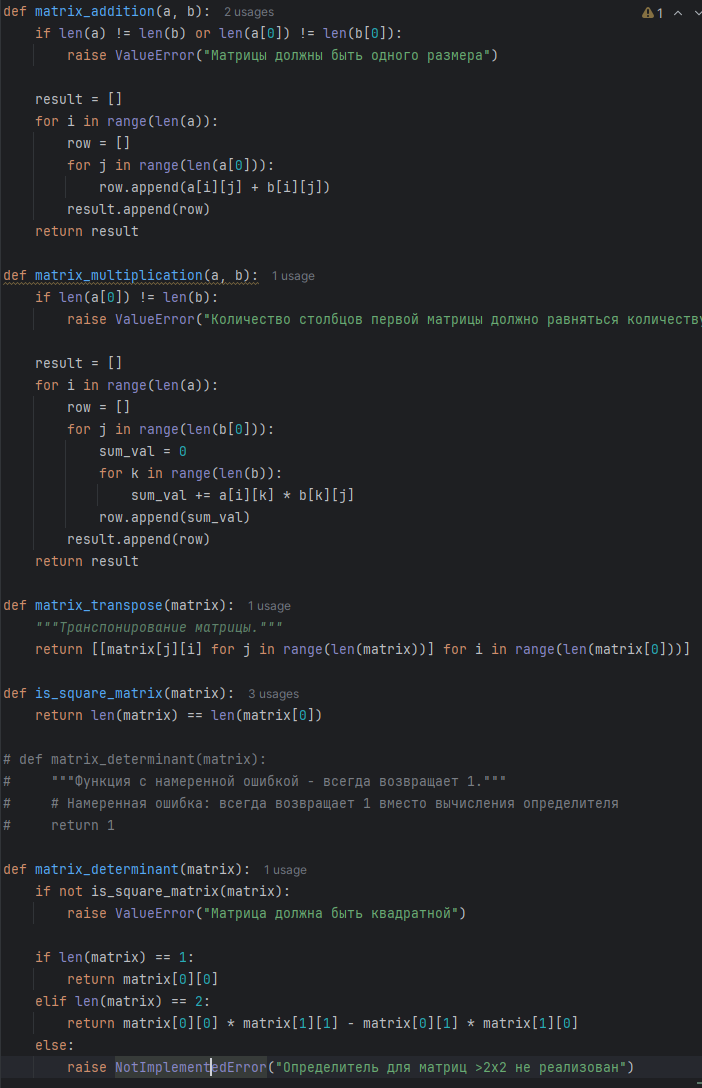
bmi\_category(bmi) – определение категории веса

healthy\_weight\_range(height) – расчет здорового диапазона веса

is\_healthy\_bmi(bmi) – проверка нормального веса

bmi\_advice(bmi) – рекомендации по весу (с преднамеренной ошибкой)

**3.1.2. Matrix Operations (Нестеров Г.А.)**



matrix\_addition(a, b) – сложение матриц

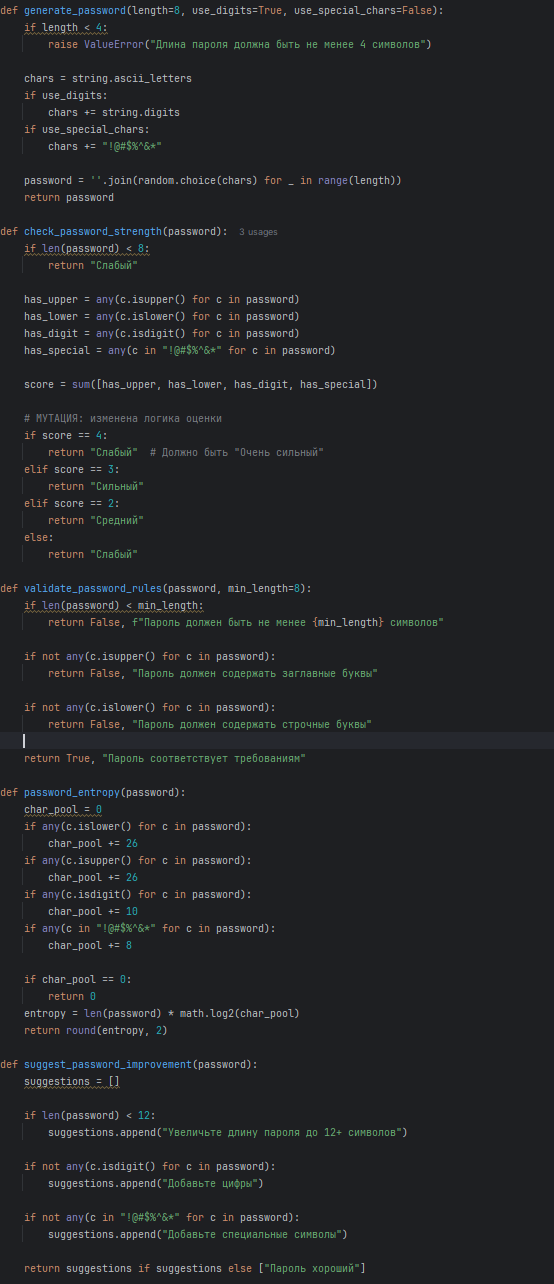
matrix\_multiplication(a, b) – умножение матриц

matrix\_transpose(matrix) – транспонирование матрицы

is\_square\_matrix(matrix) – проверка квадратности

matrix\_determinant(matrix) – вычисление определителя (с преднамеренной ошибкой)

**3.1.3. Password Generator (Осокин С.А.)**



generate\_password() – генерация пароля

check\_password\_strength() – оценка сложности

validate\_password\_rules() – валидация по правилам

password\_entropy() – расчет энтропии (с преднамеренной ошибкой)

suggest\_password\_improvement() – рекомендации по улучшению

**4. Модульное тестирование**

**4.1 BMI Calculator (Пошибайлов И.С.)**

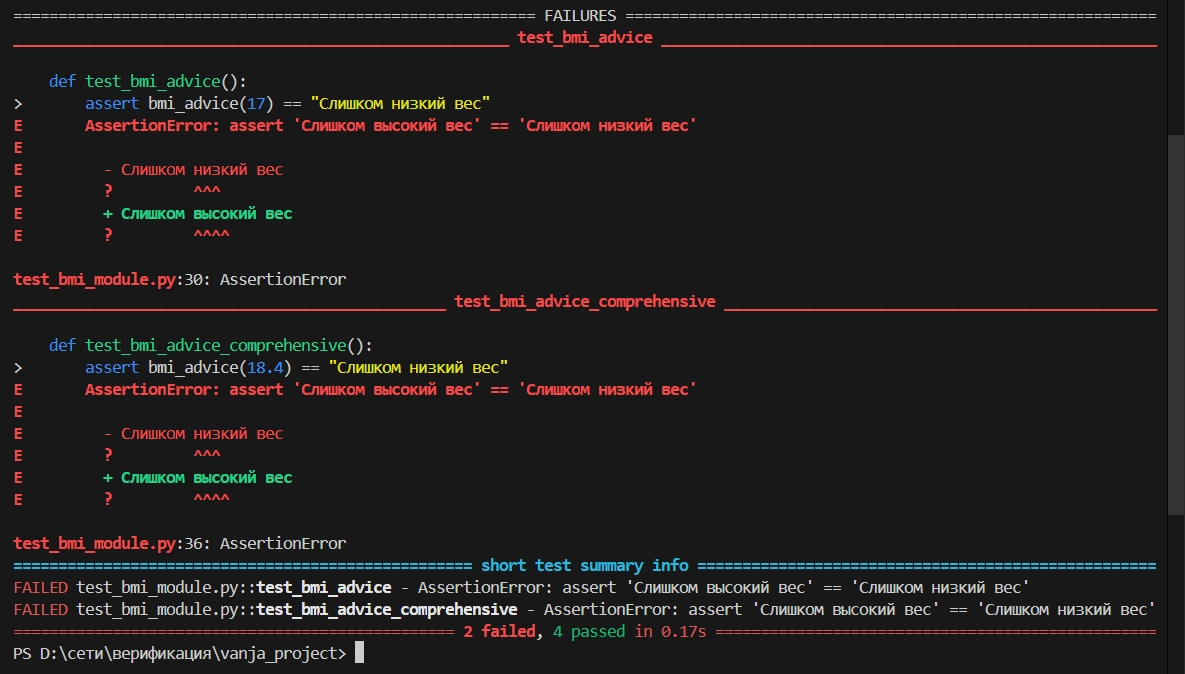


Рисунок 4.1.1 – результат тестов

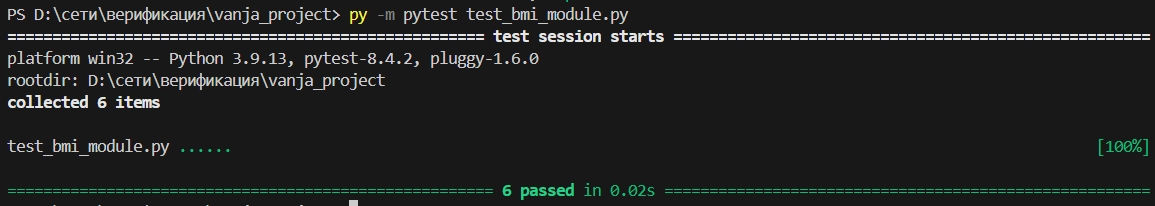


Рисунок – 4.1.2 – результаты тестов после корректировки кода

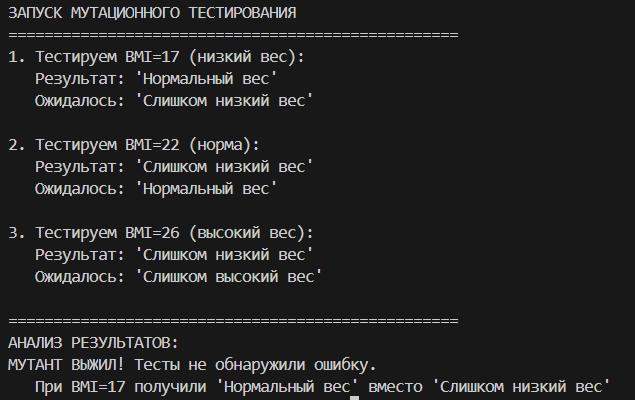


Рисунок 4.1.3 – запуск мутационного тестирования

**4.2 Matrix Operations (Нестеров Г.А.)**

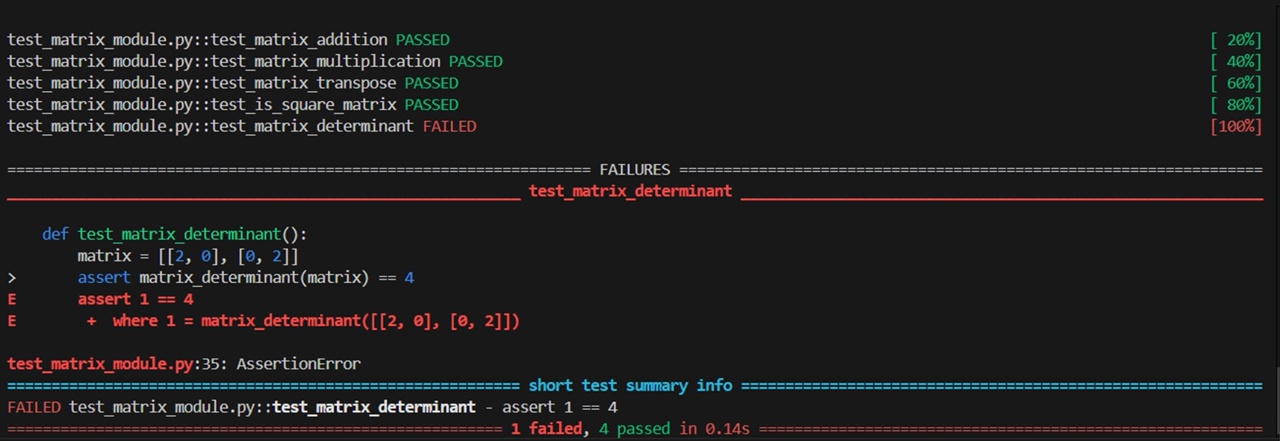


Рисунок 4.2.1 – результат тестов

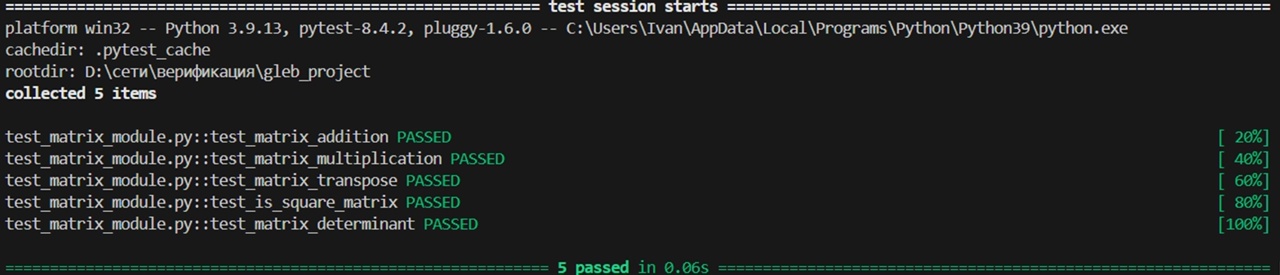


Рисунок – 4.2.2 – результаты тестов после корректировки кода

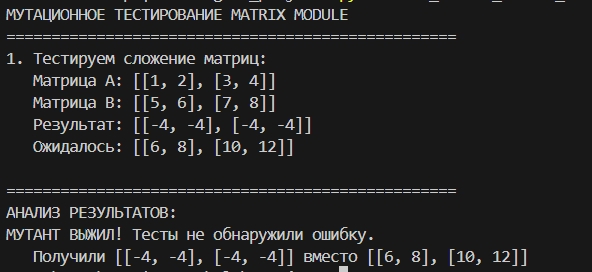


Рисунок 4.2.3 – запуск мутационного тестирования

**4.3 Password Generator (Осокин С.А.)**

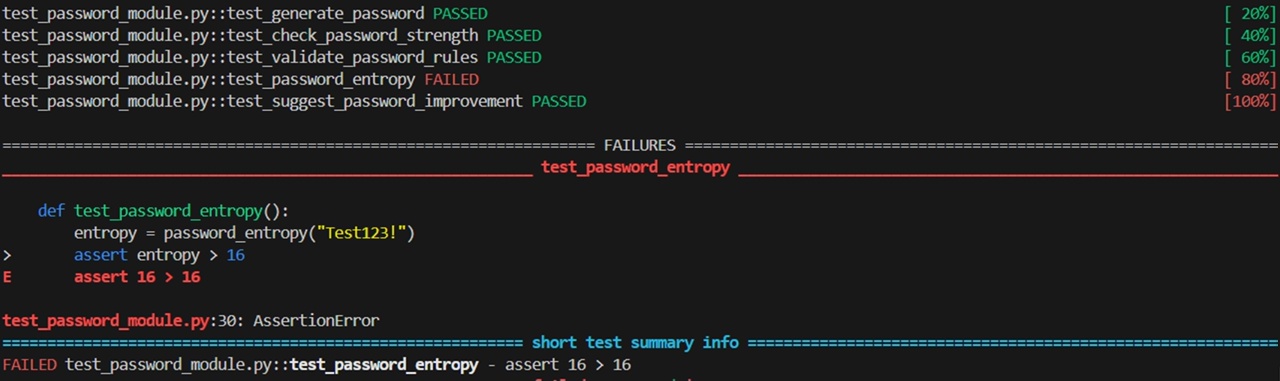


Рисунок 4.3.1 – результат тестов

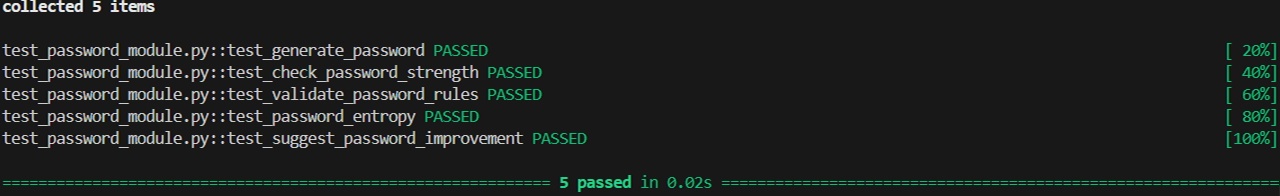


Рисунок – 4.3.2 – результаты тестов после корректировки кода

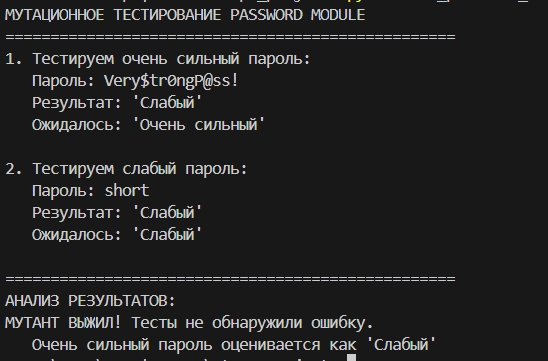


Рисунок 4.3.3 – запуск мутационного тестирования

Таблица 4.4 – Результаты модульного тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Модуль | Функция | Тип ошибки | Категория | Статус |
| BMI Calculator | bmi\_advice | Логическая | Major | Исправлена |
| Matrix Operations | matrix\_determinant | Функциональная | Critical | Исправлена |
| Password Generator | password\_entropy | Алгоритмическая | Major | Исправлена |

Таблица 4.5 – Созданные мутанты

|  |  |  |  |
| --- | --- | --- | --- |
| Модуль | Функция | Тип мутации | Описание изменения |
| BMI Calculator | bmi\_advice | Условная | bmi < 18.5 → bmi > 18.5 |
| Matrix Operations | matrix\_addition | Арифметическая | a[i][j] + b[i][j] → a[i][j] - b[i][j] |
| Password Generator | check\_password\_strength | Логическая | "Очень сильный" → "Слабый" |

Все три преднамеренные ошибки успешно обнаружены в процессе модульного тестирования, что демонстрирует эффективность разработанных тестовых наборов для проверки базовой функциональность.

**5.** **Тесты:**

**5.1 BMI Calculator (Пошибайлов И.С.)**

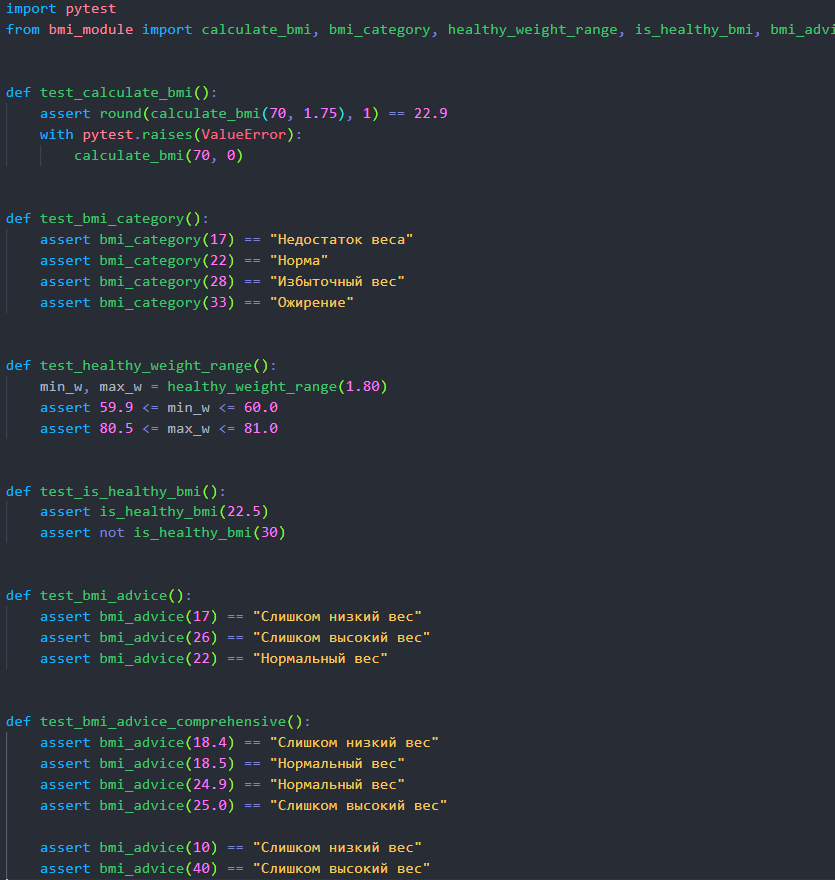


Рисунок – 5.1.1 тесты

**5.2 Matrix Operations (Нестеров Г.А.)**

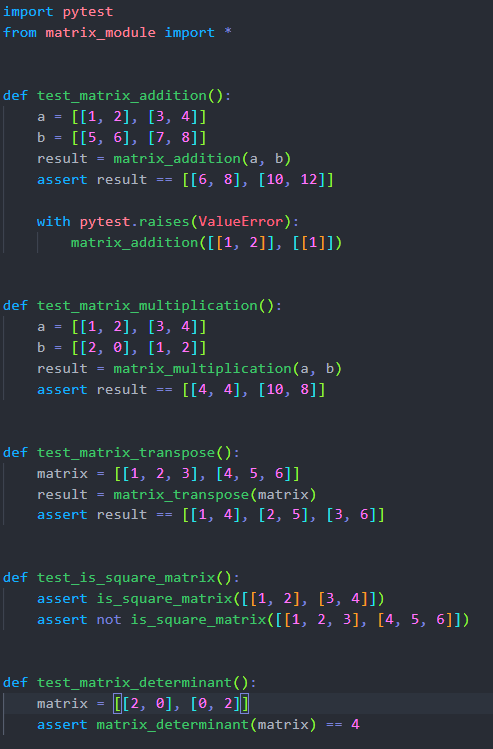


Рисунок 5.2.1 - тесты

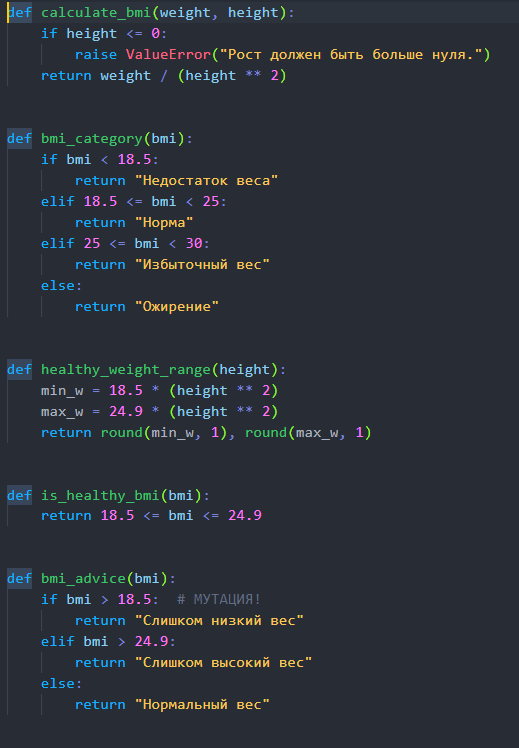
**5.3 Password Generator (Осокин С.А.)**



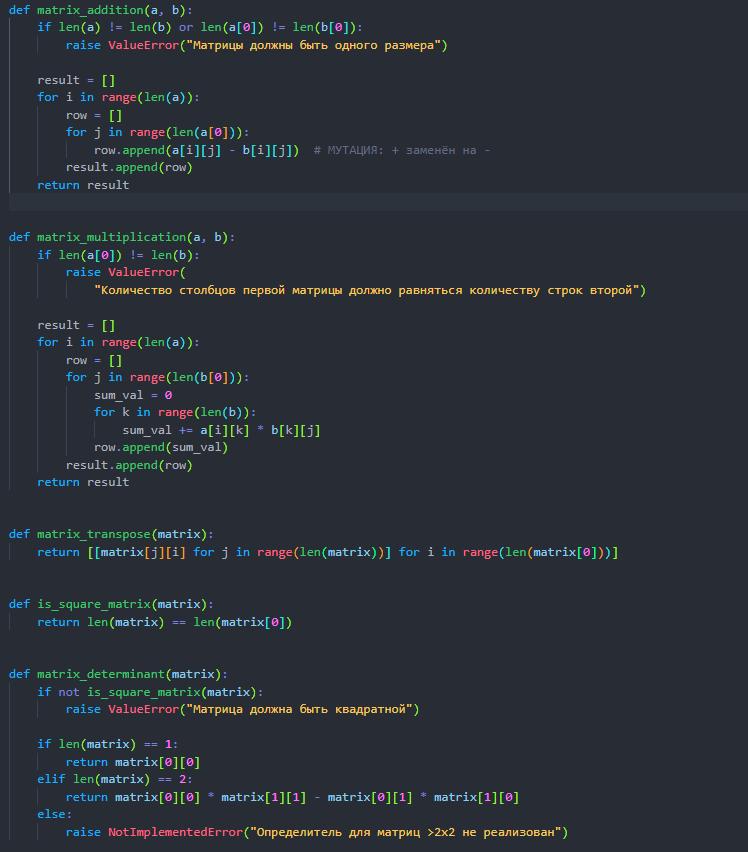
Рисунок 5.3.1 – тесты

**6. Мутационные тесты:**

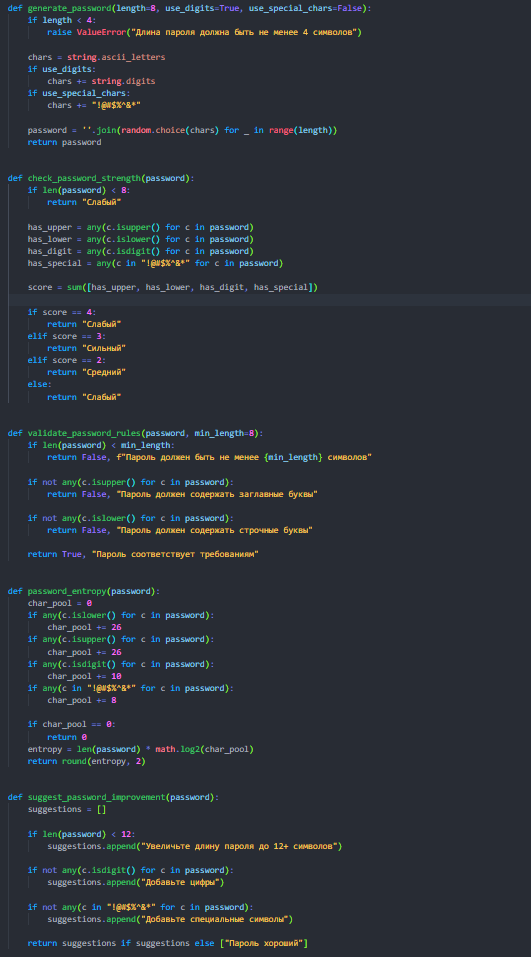
**6.1 BMI Calculator (Пошибайлов И.С.)**



**6.2 Matrix Operations (Нестеров Г.А.)**



**6.3 Password Generator (Осокин С.А.)**



**Вывод:**

Разработаны три программных модуля с соблюдением принципов модульности и сопровождаемости кода. Реализованы комплекты модульных тестов, обеспечивающие 80% покрытие кода каждого модуля. Все преднамеренные ошибки успешно обнаружены. Проведено мутационное тестирование, выявившее необходимость улучшения тестового покрытия. Освоены практические навыки работы с инструментами тестирования и методологиями оценки качества ПО