**Министерство науки высшего образования Российской Федерации**

**федеральное государственное автономное образовательное учреждение высшего образования**

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**(Университет ИТМО)**

Факультет информационных технологий и программирования

**Лабораторная работа №4**

По дисциплине «Инструментальные средства разработки программного обеспечения»

**Написание Unit тестов**

Выполнил студент группы №М3105:  
*Ким Александр Николаевич*

Проверила:  
*Кулешова Екатерина Дмитриевна*



Санкт-Петербург 2022

Ход работы:

1. Определим Цели и задачи тестирования:

Цели тестирования:

- Обеспечение качества программного обеспечения (ПО).

- Предотвращение выявления ошибок на стадии реальной эксплуатации ПО.

- Оценка соответствия ПО заданным требованиям и критериям.

- Обеспечение контроля изменений в ПО и документации.

- Оценка надежности и стабильности ПО.

- Повышение производительности ПО.

- Улучшение уровня безопасности ПО.

Задачи тестирования:

- Тестирование соответствия ПО заданным требованиям и критериям.

- Оценка функциональности и надёжности ПО.

- Проверка выполнения критических функций и задач.

- Идентификация ошибок, уязвимостей и недостатков ПО.

- Оценка текущего состояния ПО и разработка стратегии исправления обнаруженных проблем.

- Тестирование совместимости ПО с различными операционными системами, аппаратными платформами и сетевыми конфигурациями.

- Оценка эффективности работы ПО и разработка стратегии оптимизации его работы.

- Тестирование на соответствие требованиям доступности, качества поддержки пользователей и возможности адаптации ПО под изменяющиеся требования.

- Оценка соответствия ПО требованиям нормативных документов, стандартов и правил в области информационной безопасности.

- Тестирование совместимости ПО с другими программными компонентами и системами.

2. Составим Описание тестируемого продукта:

Набор файлов на языке Python, каждый файл – набор функций вычисления площади и периметра определенной геометрической фигуры. Всего 4 файла, следовательно 4 фигуры: окружность, прямоугольник, квадрат, треугольник. Описание работы определенной функции уже включено в тело каждой из них. Функции не принимают строковые значения переменных. Требования к продукту – не нарушать конфигурации функции и не менять их описание.

3. Определим области тестирования:

Тестирование проводится для каждой функции из заданного набора. В каждом файле по две функции. Тесты написаны для каждого компонента итогового продукта. Тестирование будет охватывать все функции для вычисления площади и периметра круга, прямоугольника, квадрата и треугольника.

4. Используемые методы тестирования:

Принцип тестирования: всестороннее тестирование: Этот принцип подчеркивает важность тщательного тестирования, включая крайние случаи. Будет использован подход "белого ящика", так как код функций известен. Тестирование будет проводиться с использованием модульного тестирования (Unittest) на языке Python

Стратегия тестирования: анализ граничных значений: Эта стратегия включает в себя определение и тестирование граничных значений входной области, чтобы убедиться, что тестируемая функция правильно обрабатывает эти значения.

5. Критерии приемки тестирования:

Все тесты, при условии их корректных входных данных должны пройти. Если входные данные являются корректными, то ни одна из функций не должна вызывать исключения.

Тесты при некорректных входных данных могут быть не пройдены.

6. Ожидаемые результаты:

Итоги: ожидается, что все функции будут корректно работать при валидных входных данных и нормально обрабатывать исключения с неваляными выходящими данными. В итоге тестирования должны быть получены отчёты о состояниях тестирований (пройдено/не пройдены) для каждой функции. Отчёты о дефектах будут отображаться в консоли после выполнения тестирования, случаи будут исправляться до момента полного искоренения ошибочных результатов. Помимо этого, будет сформирована метрика важности тестов в каждой категории анализа работоспособности программы. Статусы выполнения тестов и приоритетность тестов будут в дальнейшем отражены в итоговом отчёте.

7. Анализ тестирования:

Во-первых, проанализируем результаты теста. Убедимся, что модульные тесты охватывают все возможные пути к коду, граничные случаи и сценарии обработки исключений.

Во-вторых, если есть какие-либо неудачные тесты, тщательно изучим причину сбоя. Распространенные причины неудачных тестов включают неверные тестовые данные, ненадежные тесты и неполное покрытие кода.

Внесём необходимые исправления на основе проведенного анализа. Например, если тесты завершаются неудачей из-за неправильных тестовых данных, обновим тестовые данные, чтобы привести их в соответствие с ожидаемым результатом.

Если тест завершается неудачей из-за неполадок, исследуем первопричину неполадок и соответствующим образом внесём исправления. Неполадки часто можно уменьшить, улучшив общий дизайн и надежность программного обеспечения.

После прохождения всех тестов убедимся, что тестовое покрытие соответствует заранее определенным критериям. Если нет, добавим больше тестов, чтобы улучшить охват.

Наконец, задокументируем результаты тестирования. Это поможет определить любые области для улучшения и способствовать обеспечению качества продукта.

Важно помнить, что подход может варьироваться в зависимости от конкретных требований и контекста программного проекта.

! Итоговый отчет по тестированию находится в файле ***Unittests report.xlsx***