МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

“Санкт-Петербургский национальный исследовательский университет информационных технологий механики и оптики”

Мегафакультет: трансляционных информационных технологий

Факультет: информационных технологий и программирования

# Лабораторная работа №4

## По дисциплине: “Инструментальные средства разработки ПО”

## Тема: “GIT”

Выполнил студент группы №3124:

*Андреева Анастасия Дмитриевна*

Работу проверил:

*Повышев Владислав Вячеславович*

**САНКТ-ПЕРЕРБУРГ**

**2025**

# 1. Цели и задачи тестирования

## Цель

Обеспечение качества и подтверждение корректности работы ключевых математических функций модуля `rectangle.py` путем автоматизированного Unit-тестирования.

## Задачи

1. Разработать набор Unit-тестов для функций `area()` и `perimeter()` с использованием фреймворка `unittest`.
2. Проверить функциональность с использованием типичных, граничных и исключительных значений.
3. Подтвердить, что результаты работы функций соответствуют ожидаемым математическим расчетам.
4. Сформировать протокол выполнения тестов.

# 2. Описание тестируемого продукта

**Название продукта:** Python-модуль `rectangle.py` **Назначение:** Предоставление функций для расчета геометрических параметров двумерных фигур (прямоугольника). **Ключевая функциональность (SUT - System Under Test):**

1. **`area(a, b)`:** Функция принимает на вход два числовых аргумента (`a` и `b`), представляющих длины сторон прямоугольника, и возвращает его площадь.
2. **`perimeter(a, b)`:** Функция принимает на вход два числовых аргумента (`a` и `b`), представляющих длины сторон прямоугольника, и возвращает его периметр.

**Требования:** Функции должны возвращать корректные числовые результаты для положительных входных данных.

# 3. Область тестирования (Scope)

Тестирование фокусируется исключительно на Unit-уровне и охватывает следующие компоненты:

| Компонент | Тестируемая область | Типы данных |
|---|---|---|
| `area(a, b)` | Расчет площади | Положительные целые и дробные числа, нулевые значения, отрицательные значения (проверка на исключение). |
| `perimeter(a, b)` | Расчет периметра | Положительные целые и дробные числа, нулевые значения, отрицательные значения (проверка на исключение). |

| | |
|---|---|
| **Ограничения:** | Не проводится тестирование интеграции, производительности, безопасности, пользовательского интерфейса (т.к. его нет). |

# 4. Стратегия тестирования

### Подход

Используется **Стратегия Unit-тестирования** (модульного тестирования) с применением техники **Черного ящика (Black-Box)**, так как тесты разрабатываются на основе требований и ожидаемого поведения функций, без анализа внутренней реализации кода.

### Инструменты

- **Язык:** Python
- **Фреймворк:** `unittest` (стандартный модуль Python)

### Тестовые сценарии (Примеры)

Тесты будут включать следующие категории:

1. **Позитивные тесты (Typical Cases):** Проверка стандартных входных данных, где `a > 0` и `b > 0`.
2. **Граничные условия (Boundary/Edge Cases):**
   - Проверка нулевых значений (e.g., `area(10, 0)` -> ожидается 0).
   - Проверка одинаковых сторон (квадрат, e.g., `area(5, 5)`).
3. **Негативные тесты (Exception Cases):**
   - Проверка отрицательных значений (e.g., `area(-1, 5)` -> ожидается обработка ошибки или исключение).
   - Проверка некорректных типов данных (e.g., `area('a', 5)`).

# 5. Критерии приемки

Тестирование считается успешно завершенным, и модуль может быть принят при выполнении следующих критериев:

1. **Успешность тестов:** Все разработанные Unit-тесты должны быть выполнены со статусом `OK` (т.е., без `F` - Failure и `E` - Error).
2. **Покрытие:** Достигнуто полное покрытие тестами всех публичных функций модуля (`area` и `perimeter`).
3. **Соответствие требованиям:** Подтверждена корректность расчетов согласно формулам геометрии.

# 6. Ожидаемые результаты

По результатам выполнения тестирования будут получены следующие артефакты:

1. **Код Unit-тестов:** Файл `test_rectangle.py` (или аналогичный), содержащий класс `RectangleTestCase`, унаследованный от `unittest.TestCase`.

2. **Протокол выполнения тестов:** Лог вывода команды `python.exe -m unittest rectangle.py` с финальным статусом (`OK` или список ошибок).
3. **Отчет о дефектах (при наличии):** Список обнаруженных ошибок в функциях `area` и `perimeter`, если они были выявлены Unit-тестами.