

## АННОТАЦИЯ К РАБОЧЕЙ ПРОГРАММЕ ДИСЦИПЛИНЫ «Тестирование и отладка программного обеспечения»

по основной профессиональной образовательной программе по направлению подготовки  
09.03.04 «Программная инженерия» (уровень бакалавриата)

**Направленность (профиль):** Разработка программно-информационных систем

**Общий объем дисциплины** – 5 з.е. (180 часов)

**Форма промежуточной аттестации** – Экзамен.

**В результате освоения дисциплины у обучающихся должны быть сформированы компетенции с соответствующими индикаторами их достижения:**

- ОПК-6.1: Формализует задачу и предлагает алгоритмическое решение;
- ОПК-6.2: Проектирует программные продукты с применением основ информатики;
- ОПК-6.3: Осуществляет разработку и тестирование программных продуктов;

**Содержание дисциплины:**

Дисциплина «Тестирование и отладка программного обеспечения» включает в себя следующие разделы:

**Форма обучения очная. Семестр 8.**

**1. Основы информатики и программирования в сфере тестирования программ.**

**Необходимость процессов тестирования и верификации программного обеспечения.**

**Специфицирование программного обеспечения и его тестирование методом черного ящика.**

**Роли людей, процессов, методов, инструментов и технологий обеспечения качества.**

Формализация задач. Алгоритмическое решение. Тестирование и верификация, основные определения. Тестировщик или QA-инженер.

Тестирование черного и белого ящика. Регрессивное тестирование. Процесс тестирования и V-модели. Спецификация в виде инвариантов, предусловий и постусловий. Спецификация в виде Use-Case диаграмм языка UML. Работа тестировщика. Работа с баг-трекинговой системой. Краткая справка по командной работе на GitHub.

**2. Концепции и атрибуты качества на уровне кода. Отладка кода. Модульное тестирование и документирование проекта.**

Отладка кода и отладчики. Место процесса модульного тестирования в процессе разработки. Документирование проекта на уровне кода. Обеспечение надёжности в процессе модульного тестирования. Фреймворки модульного тестирования. Некоторые советы по модульному тестированию.

**3. Применение основных методов разработки в тестировании. Методологии разработки \*DD (TDD, BDD, MDD). Разработка через тестирование. Конструирование программ на основе предварительных тестов.** Применение основных методов и инструментов разработки прямо в процессе тестирования.

Что такое методология разработки. Немного о MDD (Model Driven Development). Разработка через тестирование (TDD, Test Driven Development). Разработка, управляемая поведением (BDD, Behavior Driven Development). Настройка инструментов и примеры.

**4. Функциональное автоматизированное тестирование.** Зачем нужно функциональное тестирование и зачем его автоматизировать. Подходы к автоматизации. Тестирование настольных (desktop) приложений с помощью IBM Rational Functional Tester. Тестирование Web-приложений с помощью Selenium. Разработка по BDD с помощью Selenium WebDriver.

**5. Методы анализа программ и их верификации. Статические проверки и динамический анализ программ. Логика Флойда-Хоара. Тройки Хоара. Дедуктивная верификация. Контракты на код.** Статический анализ, его необходимость. SonarQube. PVS-Studio. Cppcheck. О написании собственных статических анализаторов. Динамический анализ с помощью Valgrind. Дедуктивная верификация. Контракты на код. Язык программирования Eiffel. Design By Contract. Microsoft .NET Code Contracts.

**6. Model Based Testing. Средство MS Spec Explorer. Model Based Checking. Средство Spin.** Тестирование по модели. Задачи теории графов для генерации тестов. Установка и работа в Spec Explorer. Создание автоматной модели последовательности вызова окон в задаче аутентификации. Что такое Model Based Checking. Особенности языка описания моделей Promela. Темпоральная

логика линейного времени. Метод Model Checking внутри. Верификация и симуляция в Spin и iSpin. Применение верификации при решении переборных задач. Верификация протоколов и параллельных действий.

**7. Модульная платформа статической верификации и дедуктивного доказательства Frama-C.** Начало работы с Frama-C, работа с frama-c-gui. Метод слабейших предусловий и анализ спецификаций на ACSL. Леммы. Проверка LTL выражений для C программ..

Разработал:  
доцент  
кафедры ПМ

С.М. Старолетов

Проверил:  
Декан ФИТ

А.С. Авдеев