

Введение	5
1. Необходимость процессов тестирования и верификации программного обеспечения	6
1.1. Тестирование и верификация, основные определения	6
1.2. Тестировщик или QA-инженер	11
1.3. Тестирование черного и белого ящика	14
1.4. Регрессивное тестирование	15
1.5. Процесс тестирования в V-модели	16
ЧАСТЬ 1. Тестирование и разработка программного обеспечения	23
2. Специфицирование программного обеспечения и его тестирование методом черного ящика	23
2.1. Спецификация в виде инвариантов, предусловий и постусловий	24
2.2. Спецификация в виде Use Case-диаграмм языка UML	29
2.3. Работа тестировщика	32
2.4. Работа с баг-трекинговой системой	35
2.5. Краткая справка по командной работе на GitHub	41
2.5.1. Установки	42
2.5.2. Получение у себя на компьютере локальной копии уже имеющегося репозитория	42
2.5.3. Добавление в репозиторий своих файлов и папок	43
2.5.4. Фиксация изменений (коммит)	44
2.5.5. Передача изменений на сервер	44
2.5.6. Обновление файлов с сервера	45
Задания (командная работа по специфицированию и тестированию)	45
3. Уровень кода. Модульное тестирование и документирование проекта	46
3.1. Качество на уровне кода	46
3.2. Место процесса модульного тестирования в процессе разработки	48
3.3. Документирование проекта на уровне кода	48
3.4. Процесс модульного тестирования	59
3.5. Фреймворки модульного тестирования	62
3.5.1. JUnit – модульное тестирование для Java	62
3.5.2. NUnit и MSTest – модульное тестирование для /Net-языков	68
3.5.3. Google Test (C++), мультиплатформенный фреймворк	73
3.6. Некоторые советы по модульному тестированию	79
Задания (документирование кода и модульное тестирование в группе)	80
4. Методологии разработки *DD. Разработка через тестирование	82
4.1. Что такое методология разработки	82
4.2. Немного о MDD (Model Driven Development)	82
4.3. Разработка через тестирование (TDD, Test Driven Development)	83
4.4. Разработка, управляемая поведением (BDD, Behavior Driven Development)	94
4.5. Некоторые предложения по разработке через написание статей (ADD, Article Driven development)	104
Задания (разработка через тестирование)	112
5. Функциональное автоматизированное тестирование	114
5.1. Зачем нужно функциональное тестирование и зачем его автоматизировать	114
5.2. Подход Apple к автоматизации	114
5.3. Тестирование настольных (desktop) приложений с помощью IBM Rational Functional Tester	118
5.4. Тестирование Web-приложений с помощью Selenium	134
Задания (функциональное тестирование)	144

ЧАСТЬ 2. Методы анализа программ и их верификация	145
6. Статические проверки и динамический анализ программ.....	145
6.1. Статический анализ, его необходимость.....	145
6.2. PVS-Studio	148
6.3. Cppcheck	150
6.4. О написании собственных статических анализаторов.....	151
6.5. Динамический анализ с помощью Valgrind.....	156
6.6. О пользе и вреде «printf injection».....	161
Задания (динамический и статический анализ).....	162
7. Тройки Хоара. Дедуктивная верификация. Контракты на код.....	163
7.1. Язык программирования Eiffel. Design By Contract	166
7.1.1. Введение в классы на Eiffel	166
7.1.2. Контракты на Eiffel.....	170
7.1.3. Тестирование в Eiffel.....	175
7.1.4. Контракты на циклы.....	182
7.1.5. Многопоточность и контракты на параллельно выполняющиеся действия.....	185
7.2. Microsoft .NET Code Contracts.....	201
7.2.1. Подключение MS Code Contracts.....	202
7.2.2. Определение контрактов и их проверка.....	203
7.2.3. Более сложные контракты.....	207
Задания (контракты на код).....	209
8. Model Based Testing. Средство MS Spec Explorer.....	210
8.1. Тестирование по модели.....	210
8.2. Задачи теории графов для генерации тестов.....	210
8.3. Установка и работа в Spec Explorer.....	213
8.4. Создание модели для простой последовательности.....	217
8.5. Создание модели для сложной последовательности.....	222
8.6. Другие возможности Spec Explorer.....	236
Задания (тестирование с помощью MBT).....	241
9. Model Based Checking. Средство Spin.....	242
9.1. Что такое Model Based Checking.....	242
9.2. Особенности языка описания моделей Promela.....	243
9.3. Темпоральная логика линейного времени.....	246
9.4. Метод Model Checking внутри.....	249
9.5. Верификация и симуляция в Spin и iSpin.....	254
9.5.1. Пример с моделированием поведения интерфейса приложения.....	255
9.5.2. Установка Spin и графической оболочки iSpin.....	262
9.5.3. Работа в среде iSpin.....	263
9.5.4. Работа со Spin из командной строки.....	267
9.5.5. Применение верификации при решении переборных задач.....	268
9.5.6. Пример применения симуляции для задачи массового обслуживания.....	278
9.5.7. Верификация протоколов и параллельных действий.....	283
Задания (Model Based Checking).....	289
10. Модульная платформа статической верификации Frama-C.....	290
10.1. Начало работы с Frama-C, работа с frama-c-gui.....	290
10.2. Метод слабейших предусловий и анализ спецификаций на ACSL.....	301
10.3. Проверка LTL-выражений для C-программ.....	321
Задания (верификация C-программ на уровне кода и требований).....	328
Итоговый тест для самоконтроля	329
Литература.....	340