

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Алтайский государственный технический университет им. И.И. Ползунова»

СОГЛАСОВАНО

Декан ФИТ

А.С. Авдеев

Рабочая программа дисциплины

Код и наименование дисциплины: **Б1.О.17 «Тестирование и отладка программного обеспечения»**

Код и наименование направления подготовки (специальности): **09.03.04**

Программная инженерия

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

Статус дисциплины: **обязательная часть**

Форма обучения: **очная**

Статус	Должность	И.О. Фамилия
Разработал	доцент	С.М. Старолетов
Согласовал	Зав. кафедрой «ПМ»	Е.Г. Боровцов
	руководитель направленности (профиля) программы	С.А. Кантор

г. Барнаул

1. Перечень планируемых результатов обучения по дисциплине, соотнесенных с индикаторами достижения компетенций

Компетенция	Содержание компетенции	Индикатор	Содержание индикатора
ОПК-6	Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов	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

2. Место дисциплины в структуре образовательной программы

Дисциплины (практики), предшествующие изучению дисциплины, результаты освоения которых необходимы для освоения данной дисциплины.	Математическая логика и теория алгоритмов, Объектно-ориентированное программирование, Основы программной инженерии
Дисциплины (практики), для которых результаты освоения данной дисциплины будут необходимы, как входные знания, умения и владения для их изучения.	Выполнение и защита выпускной квалификационной работы, Преддипломная практика

3. Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающегося с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающегося

Общий объем дисциплины в з.е. /час: 5 / 180

Форма промежуточной аттестации: Экзамен

Форма обучения	Виды занятий, их трудоемкость (час.)				Объем контактной работы обучающегося с преподавателем (час)
	Лекции	Лабораторные работы	Практические занятия	Самостоятельная работа	
очная	24	24	0	132	62

4. Содержание дисциплины, структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

Форма обучения: очная

Лекционные занятия (24ч.)

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. {беседа} (4ч.)[1,4,9] 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 {лекция-пресс-конференция} (4ч.)[1,4,8,15]

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 {лекция с разбором конкретных ситуаций} (4ч.)[1,3,4,7] 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 {лекция-пресс-конференция} (4ч.)[1,4,5,6,9,12,13]

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. {дискуссия} (3ч.)[1,4,14,18,19]

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. {«мозговой штурм»} (4ч.)[1,4,5,10,11] 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 {мини-лекция} (1ч.)[1,4,16,17] 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.

Лабораторные работы (24ч.)

1. Командная работа на GitHub-репозитории по ручному тестированию студенческих программ {с элементами электронного обучения и дистанционных образовательных технологий} (3ч.)[1,4,9] Работа с баг-трекингowymi системами. Работа с GitHub. Разработка поведенческой спецификации существующего ПО командой. Разработка Use-Case UML модели. Заливка в общий репозиторий. Тестирование программ других команд согласно спецификациям. Занесение багов в issues.

2. Модульное тестирование и специфицирование {разработка проекта} (3ч.)[1,4] Разработка спецификации на имеющийся код, генерирование документации, написание тестов на код студента в малой группе

3. Применение методологий TDD и BDD для разработки программного обеспечения {разработка проекта} (3ч.)[1,3,4,7,8] Действия по индивидуальным заданиям, выполнение требований задания, фиксация изменений (коммит) в системе контроля версий git каждого шага разработки (тест, код)

4. Функциональное тестирование. Разработка, управляемая поведенческой спецификацией {работа в малых группах} (3ч.)[1,4,7,9,12,13] Тестирование настольного приложения. Тестирование web-приложения. Разработка части web-приложения с нуля согласно спецификации и управления браузером

5. Анализ программного обеспечения (статический и динамический). Подход контрактного программирования {работа в малых группах} (4ч.)[1,4,14,16,18,19] Изучение группой различных статических и динамических методов анализа и переход к изучению методов формальной верификации, первая часть - контракты на код в Eiffel и MS Code contracts

6. Методы формального тестирования и верификации {работа в малых группах} (8ч.)[1,4,5,10,11,16,17] Model Based Testing, Model Based Checking, изучение продуктов Spec Explorer и Spin, описания требований к коду на ACSL при использовании анализатора и средства дедуктивного доказательства Frama-C.

Представление и обсуждение результатов самостоятельной работы по теме

Самостоятельная работа (132ч.)

1. Освоение учебной литературы по тестированию и верификации {разработка проекта} (20ч.)[1,4]

2. Изучение литературы в виде пособия, а также предложенных публикаций с целью изучения существующих методов формальной верификации программного обеспечения на основе проверки моделей и дедуктивного доказательства корректности. {анализ казусов} (76ч.)[1,4,7,10,11,16,17]

Повторение предложенных примеров и их проверка самостоятельно в одном из средств верификации. Обучающиеся в малых группах глубоко изучают предложенные преподавателем примеры и делают аналогичные модели для своих систем на модельных языках. Изучаются преимущества и недостатки методов, а также трудоемкость их осуществления, и делается вывод о применимости данного класса методов для реальных задач из программной инженерии.

3. Рефакторинг (улучшение) своего кода при выполнении лабораторных работ {разработка проекта} (16ч.)[1] После изучения нового материала, предлагать и осуществлять рефакторинг ранее написанного ПО, так, чтобы оно было тестируемым, в том числе, для подготовки к экзамену.

4. Подготовка к экзамену(20ч.)[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19]

5. Перечень учебно-методического обеспечения самостоятельной работы обучающихся по дисциплине

Для каждого обучающегося обеспечен индивидуальный неограниченный доступ к электронно-библиотечным системам: Лань, Университетская библиотека он-лайн, электронной библиотеке АлтГТУ и к электронной информационно-образовательной среде:

1. Старолетов С.М. Основы тестирования и верификации программного обеспечения [Электронный ресурс]: Учебное пособие.— Электрон. дан.— Барнаул: АлтГТУ, 2020.— Режим доступа: http://elib.altstu.ru/eum/download/pm/Staroletov_OsnTestVerifPO_up.pdf, авторизованный

6. Перечень учебной литературы

6.1. Основная литература

2. Виссер, Д. Разработка обслуживаемых программ на языке Java [Электронный ресурс] / Д. Виссер ; пер. с англ. Р. Н. Рагимова. — Электрон. дан. — Москва : ДМК Пресс, 2017. — 182 с. — Режим доступа: <https://e.lanbook.com/book/105834>. — Загл. с экрана.

6.2. Дополнительная литература

6.2. Дополнительная литература

3. Персиваль, Г. Python. Разработка на основе тестирования. Повинуйся Билли-тестировщику, используя Django, Selenium и JavaScript [Электронный ресурс] / Г. Персиваль ; пер. с англ. А.В. Логунов. — Электрон. дан. — Москва : ДМК Пресс, 2018. — 622 с. — Режим доступа: <https://e.lanbook.com/book/111440>.

7. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

4. Старолетов, С.М. Основы тестирования и верификации программного обеспечения [Электронный ресурс] : учебное пособие / С.М. Старолетов. — 2-е изд., стер. — Электрон. дан. — Санкт-Петербург : Лань, 2020. — 344 с. — Режим доступа: <https://e.lanbook.com/book/138181>.

5. Старолетов С.М., Крючкова Е.Н. Тестирование распределенных приложений на основе построения моделей //Прикладная информатика. – 2008. – №. 6. — Режим доступа: <https://elibrary.ru/item.asp?id=11675451>

6. Галкин Р.Е., Старолетов С.М. Технология тестирования криптовалютных шлюзов. Материалы Всероссийской молодежной научно-практической конференции. Под ред. Л.И. Сучковой. Барнаул, 2018. — Режим доступа: <https://elibrary.ru/item.asp?id=37414945>

7. Staroletov S. Building a process of trustworthy software developing based on BDD and ontology approaches with further formal verification //9th Workshop PSSV. – 2018. – С. 92-97. — Режим доступа: <https://elibrary.ru/item.asp?id=35083203>

8. Шевелёва А.Г., Старолетов С.М. Построение процесса разработки и тестирования интеллектуальных систем обработки информации на основе методологии MDD // Ползуновский альманах. - 2017. – №. 4. — Режим доступа: <https://elibrary.ru/item.asp?id=30722187>

9. Ресурсы сайта [software-testing.ru](http://www.software-testing.ru). — Режим доступа: <http://www.software-testing.ru>

10. Margus Veanes, Colin Campbell, Wolfgang Grieskamp, Wolfram Schulte, Nikolai Tillmann. Model-Based Testing of Object-Oriented Reactive Systems with Spec Explorer. - Режим доступа: <https://www.microsoft.com/en-us/research/publication/model-based-testing-of-object-oriented-reactive-systems-with-spec-explorer>

11. Spin Formal Verification. Ресурсы сайта [spinroot.com](http://www.spinroot.com). — Режим доступа: <http://www.spinroot.com>

12. Selenium - Web Browser Automation. — Режим доступа: <https://www.seleniumhq.org>

13. IBM Rational Functional Tester - Part 1 - By www.openmentor.net. - Режим доступа: <https://www.youtube.com/watch?v=venkleLj0RY>

14. Eiffel Language: Documentation. - Режим доступа: <https://www.eiffel.org/documentation>

15. Doxygen. Generate documentation from source code. - Режим доступа: <http://www.doxygen.nl>

16. Frama-C. Software analysis. - Режим доступа: <https://frama-c.com>

17. Burghardt J. et al. ACSL by Example //DEVICE-SOFT project publication. Fraunhofer FIRST Institute (January 2010). – 2010. - Режим доступа: <http://www.cs.umd.edu/class/spring2016/cmsc838G/frama-c/ACSL-by-Example-12.1.0.pdf>

18. Cppcheck is a static analysis tool for C/C++ code. - Режим доступа: <http://cppcheck.sourceforge.net>

19. Valgrind is an instrumentation framework for building dynamic analysis tools. - Режим доступа: <http://www.valgrind.org>

8. Фонд оценочных материалов для проведения текущего контроля успеваемости и промежуточной аттестации

Содержание промежуточной аттестации раскрывается в комплекте контролирующих материалов, предназначенных для проверки соответствия уровня подготовки по дисциплине требованиям ФГОС, которые хранятся на кафедре-разработчике РПД в печатном виде и в ЭИОС.

Фонд оценочных материалов (ФОМ) по дисциплине представлен в приложении А.

9. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень программного обеспечения и информационных справочных систем

Для успешного освоения дисциплины используются ресурсы электронной информационно-образовательной среды, образовательные интернет-порталы, глобальная компьютерная сеть Интернет. В процессе изучения дисциплины происходит интерактивное взаимодействие обучающегося с преподавателем через личный кабинет студента.

№пп	Используемое программное обеспечение
1	Eclipse IDE
2	LibreOffice
3	Mozilla Firefox
4	Selenium IDE
5	Selenium WebDriver
6	Visual Studio
7	Windows
8	Антивирус Kaspersky

№пп	Используемые профессиональные базы данных и информационные справочные системы
1	IEEE Xplore - Интернет библиотека с доступом к реферативным и полнотекстовым статьям и материалам конференций. Бессрочно без подписки (https://ieeexplore.ieee.org/Xplore/home.jsp)
2	Бесплатная электронная библиотека онлайн "Единое окно к образовательным ресурсам" для студентов и преподавателей; каталог ссылок на образовательные

№пп	Используемые профессиональные базы данных и информационные справочные системы
	интернет-ресурсы (http://Window.edu.ru)
3	Крупнейший веб-сервис IT-проектов и их совместной разработки (https://github.com/)
4	Национальная электронная библиотека (НЭБ) — свободный доступ читателей к фондам российских библиотек. Содержит коллекции оцифрованных документов (как открытого доступа, так и ограниченных авторским правом), а также каталог изданий, хранящихся в библиотеках России. (http://нэб.рф/)

10. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

Наименование специальных помещений и помещений для самостоятельной работы
учебные аудитории для проведения учебных занятий
помещения для самостоятельной работы

Материально-техническое обеспечение и организация образовательного процесса по дисциплине для инвалидов и лиц с ограниченными возможностями здоровья осуществляется в соответствии с «Положением об обучении инвалидов и лиц с ограниченными возможностями здоровья».