ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

Решением Ученого совета
АНО ВО «Центральный университет»
«25» мая 2026 г.
Протокол №2

Рабочая программа дисциплины (модуля)
«Современные формальные методы»

Направление подготовки: *38.03.05 Бизнес-информатика*

Направленность (профиль) подготовки: *Бизнес и аналитика*

Квалификация (степень) выпускника: *бакалавр*

Форма обучения: *очная*

Срок освоения программы: *4 года*

Год набора: *2026*

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 0215B0C000D1B3FEB44A37EE631E0C27D4
Владелец: Ивашкевич Евгений Васильевич
Действителен: с 13.01.2026 до 13.04.2027

Электронный документ

Москва
2026

Содержание

1. Краткая характеристика дисциплины (модуля)

Рабочая программа дисциплины (модуля) «Современные формальные методы» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования – бакалавриат по специальности 38.03.05 Бизнес-информатика, профиль Бизнес и аналитика, утвержденный приказом Министерства науки и высшего образования Российской Федерации № 838 от 29.07.2020 года.

Изучение дисциплины (модуля) »Функциональное программирование» позволяет научиться применять строгие формальные техники в реальных проектах разработки ПО, повышая надёжность, безопасность и предсказуемость создаваемых систем.

Место дисциплины (модуля) в структуре образовательной программы

Настоящая дисциплина (модуль) включена в учебный план по программе подготовки бакалавриата по направлению 38.03.05 Бизнес-информатика, профиль Бизнес и аналитика и входит в часть, формируемую участниками образовательных отношений, Блока 1.

Дисциплина (модуль) является выборной и доступна для изучения на 3 или 4 курсе в 6, 7, 8 семестрах на выбор после успешного освоения дисциплин «Основы математического анализа и линейной алгебры» или «Математический анализ» и «Линейная алгебра и геометрия» или «Математический анализ. Продвинутый уровень» и «Линейная алгебра и геометрия. Продвинутый уровень».

Цель изучения дисциплины (модуля): сформировать у студентов целостное представление о формальных методах в программной инженерии, научив их строго описывать свойства систем, выбирать и применять подходящие формальные техники и инструменты для доказательства корректности программ и моделей.

Задачи изучения дисциплины (модуля):

— дать чёткое понимание ключевых понятий — спецификация, корректность, инвариант, модель, контрпример, доказательство.

— сравнить традиционные подходы (тестирование, статический анализ) с формальными технологиями (символьное исполнение, формальная верификация).

— обучить основам логики Хоара: предусловия, постусловия, инварианты циклов, и показать, как они позволяют доказывать частичную корректность программ.

— познакомить со средствами SAT/SMT-решения, символьного выполнения и проверкой моделей, объяснив, как они генерируют контрпримеры и подтверждают свойства.

— ознакомить с proof assistants, зависимыми типами и механизированными доказательствами, продемонстрировать их роль в современной разработке надёжного программного обеспечения.

В результате освоения дисциплины (модуля) обучающийся должен:

знать:

— основные понятия современных формальных методов: спецификация, корректность, инвариант, модель, контрпример и доказательство;

— различия между тестированием, статическим анализом, символьным исполнением и формальной верификацией;

— базовые идеи логики Хоара, предусловий, постусловий и инвариантов циклов;

— основные принципы SAT/SMT-решений, символьное выполнение и проверки моделей;

— назначение proof assistants, зависимых типов и механизированных доказательств в современной разработке программ;

уметь:

— формулировать простые формальные спецификации для программ, алгоритмов и систем;

— записывать предусловия, постусловия и инварианты для небольших программных фрагментов;

— доказывать частичную корректность простых программ с помощью логики Хоара;

— использовать SMT-решатель на базовом уровне для поиска контрпримеров;

— анализировать, какой формальный метод подходит для конкретной задачи проверки корректности;

владеть:

— навыками строгой формализации требований к программам и системам;

— приёмами поиска и записи инвариантов для алгоритмов и моделей;

— базовыми навыками работы с инструментами автоматической проверки ограничений и свойств;

— навыками интерпретации контрпримеров, найденных формальными инструментами;

— навыками критической оценки возможностей и ограничений формальных методов в реальных software engineering задачах.

2. Перечень планируемых результатов обучения

Компетенции, формируемые в результате освоения дисциплины (модуля) при проведении учебных занятий в форме контактной работы обучающихся с педагогическими работниками Университета и в форме самостоятельной работы обучающихся:

Компетенция	Содержание компетенции	Индикатор компетенции	Перечень планируемых результатов обучения по дисциплине (модулю)
УК-1.	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	УК-1.1.	Знает методы поиска и анализа информации в области аналитики, основные принципы критической оценки источников информации и их релевантности.
		УК-1.2.	Умеет критически оценивать источники информации и синтезировать данные из различных источников для решения задач, применять системный подход к анализу и решению комплексных проблем
		УК-1.3.	Имеет практический опыт работы с современными инструментами и технологиями для обработки информации, формулировании и структурировании задач на основе полученной информации
ОПК-4.	Способен понимать принципы работы информационных технологий; использовать информацию, методы и программные средства ее сбора, обработки и анализа для информационно-аналитической поддержки принятия управленческих решений	ОПК-4.1.	Знает основные принципы работы информационных технологий и их влияние на бизнес-процессы
		ОПК-4.2.	Умеет использовать методы и программные средства для сбора, обработки и анализа информации, обеспечивая качественную информационно-аналитическую поддержку
		ОПК-4.3.	Имеет практический опыт в применении аналитических инструментов для поддержки принятия управленческих решений в организациях
ПК-2.	Способен использовать соответствующий математический аппарат и инструментальные средства для обработки, анализа и систематизации информации по теме исследования для решения задач профессиональной деятельности	ПК-2.1.	Знает основные математические методы и инструментальные средства, применяемые для обработки и анализа информации
		ПК-2.2.	Умеет эффективно использовать математический аппарат для систематизации данных и решения профессиональных задач
		ПК-2.3.	Имеет практический опыт работы с инструментами анализа информации в рамках исследовательских проектов

3. Тематический план

№п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Аудиторная работа		Контроль	Самостоятельная работа	
		Лекции	Семинары			
1	Введение в формальные методы	6	6		20	Домашнее задание Аудиторная работа
2	Логика программ и доказательство корректности	6	6		20	Домашнее задание Аудиторная работа
3	Автоматическая проверка и SMT	6	6	2	30	Домашнее задание Аудиторная работа Контрольная работа
4	Model checking	6	6	2	30	Домашнее задание Аудиторная работа
5	Типы, proof assistants и механизированные доказательства	6	6	2	20	Домашнее задание Аудиторная работа Контрольная работа
	Зачет с оценкой			4		Коллоквиум
	Итого:	***30***	***30***	***10***	***120***	
Объем дисциплины (модуля) (в ак. ч.)		***190***				
Объем дисциплины (модуля) (в зач. ед.)		***5***				

4. Содержание дисциплины (модуля)

№п/п	Наименование раздела дисциплины (модуля)	Содержание дисциплины (модуля) по темам
1	Введение в формальные методы	Формальные методы в компьютерных науках Понятие спецификации. Предусловия, постусловия, инварианты. Контрактное программирование. Связь спецификации с поведением программы.
2	Логика программ и доказательство корректности	Логика Хоара Инвариант цикла. Инициализация, сохранение и завершение. Как подбирать инварианты. Типичные ошибки при доказательстве корректности циклов. Формальная корректность алгоритмов. Спецификация входа и выхода Корректность сортировки, бинарного поиска, алгоритмов на графах на базовом уровне.
3	Автоматическая проверка и SMT	SAT и SMT как инструменты формальных методов SMT-решатели в анализе программ Идея символьного выполнения. Пути исполнения программы.Ограничения символьного анализа.
4	Model checking	Понятие модели системы. Состояния, переходы, достижимость. Инварианты состояний. Темпоральная логика и свойства систем.
5	Типы, proof assistants и механизированные доказательства	Типы как спецификации Идея соответствия Карри. Доказательства как программы, типы как утверждения. Зависимые типы на обзорном уровне. Интерактивные системы доказательств. Автоматизация и ручные шаги доказательства. Формальные методы в индустрии.

5. Учебно-методическое обеспечение

Университет располагает полным набором лицензионного и свободно распространяемого программного обеспечения, включая продукты отечественного производства.

Каждый студент в течение всего периода обучения получает индивидуальный неограниченный доступ к электронно-библиотечной системе и электронной информационно-образовательной среде университета. Эти системы предоставляют возможность доступа к ресурсам из любой точки, где есть подключение к сети Интернет, как на территории университета, так и за его пределами.

Студентам обеспечен удаленный доступ к современным профессиональным базам данных и информационным справочным системам.

Основная литература:

1. Миронов, А. М. Методы верификации программ: учебное пособие / А. М. Миронов. – Москва : ДМК Пресс, 2023. - 336 с. – ISBN 978-5-93700-278-5. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2205067 (дата обращения: 14.07.2026). – Режим доступа: по подписке.

2. Пруцков, А. В. Математическая логика и теория алгоритмов: учебник / А. В. Пруцков, Л. Л. Волкова. — Москва : КУРС : ИНФРА-М, 2025. — 152 с. - ISBN 978-5-906818-74-4. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2204110 (дата обращения: 14.07.2026). – Режим доступа: по подписке.

Дополнительная литература:

1. Дорогов, В. Г. Введение в методы и алгоритмы принятия решений: учебное пособие / В.Г. Дорогов, Я.О. Теплова ; под ред. Л.Г. Гагариной. — Москва : ФОРУМ : ИНФРА-М, 2022. — 240 с. — (Высшее образование). - ISBN 978-5-8199-0486-2. - Текст : электронный. - URL: https://znanium.ru/catalog/product/1841773 (дата обращения: 14.07.2026). – Режим доступа: по подписке.

6. Материально-техническое обеспечение

Университет располагает материально-технической базой, соответствующей действующим противопожарным правилам и нормам и обеспечивающей проведение всех видов дисциплинарной и междисциплинарной подготовки, практической и научно-исследовательской работ обучающихся, предусмотренных учебным планом.

Помещения, которые представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского (практического) типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования. Помещения укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Изучение дисциплины (модуля) обеспечивается в учебных аудиториях, оснащенных:

- столами и стульями;
- компьютерной техникой;
- механическими калькуляторами;
- специализированным оборудованием, включая демонстрационное оборудование.

Помещения для самостоятельной работы обучающихся, в том числе приспособленные для использования инвалидами и лицами с ограниченными возможностями здоровья, оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду Университета.

Электронный документ

Обучающимся предоставляется доступ (в том числе удаленный) к ресурсам информационно-телекоммуникационной сети «Интернет», электронным ресурсам (в том числе электронным библиотечным системам, современным профессиональным базам данных и информационным справочным системам):

№	Наименование портала (издания, курса, документа)	Ссылка
1.	Научная электронная библиотека elibrary.ru библиотека	https://elibrary.ru/defaultx.asp
2.	База данных для IT-специалистов	https://habr.com
3.	База данных ScienceDirect	https://www.sciencedirect.com
4.	Официальный сайт Министерства науки и высшего образования Российской Федерации	https://minobrnauki.gov.ru/
5.	Федеральный портал «Российское образование»	https://www.edu.ru/
6.	Информационная система «Единое окно доступа к образовательным ресурсам»	http://window.edu.ru/
7.	Единая коллекция цифровых образовательных ресурсов	http://school-collection.edu.ru/
8.	Федеральный центр информационно - образовательных ресурсов	http://fcior.edu.ru/

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модуле), в том числе комплект лицензионного программного обеспечения, современные профессиональные базы данных и информационные справочные системы:

Наименование ПО	Производство	Лицензионное / свободно распространяемое
Операционные системы:		
Microsoft Imagine (Windows Client, Server)	зарубежное	лицензионное
Браузеры:		
Яндекс.Браузер	отечественное	свободно распространяемое
Google Chrome	зарубежное	свободно распространяемое
Офисные приложения:		
Microsoft Imagine (Visio, OneNote)	зарубежное	лицензионное
TeXstudio	зарубежное	свободно распространяемое
Adobe Acrobat Reader	зарубежное	свободно распространяемое
Программное обеспечение для планирования и учета времени:		
Toggle app	зарубежное	свободно распространяемое
Системы управления проектами:		
Microsoft Imagine (Project)	зарубежное	лицензионное
Системы управления базами данных:		
Microsoft Imagine (SQL Server)	зарубежное	лицензионное
Системы резервного копирования (backup):		
Acronis Backup Advanced for HyperV	зарубежное	лицензионное
Справочно-правовые системы:		
КонсультантПлюс: справочно-правовая система	отечественное	лицензионное
Средства антивирусной защиты:		
Kaspersky Endpoint Security для бизнеса Стандартный Russian Edition	отечественное	лицензионное
Среды разработки:		
Visual Studio Code	зарубежное	свободно распространяемое
Bash (Unix shell)	зарубежное	свободно распространяемое

Электронный документ

Anaconda	зарубежное	свободно распространяемое
Robotic Operating System	зарубежное	свободно распространяемое
CopelliaSim	зарубежное	свободно распространяемое
Google Colaboratory	зарубежное	свободно распространяемое
Пакеты программных средств и библиотек:		
AutoPsy	зарубежное	свободно распространяемое
Interactive Disassembler (IDA)	зарубежное	свободно распространяемое
Системы управления библиографической информацией:		
Zotero	зарубежное	свободно распространяемое
Сервисы и службы:		
Bind	зарубежное	свободно распространяемое
Docker	зарубежное	свободно распространяемое

7. Методические и оценочные материалы

Методические указания для обучающихся по освоению дисциплины (модуля)

В процессе изучения дисциплины (модуля) «Современные формальные методы» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, домашние задания, аудиторная работа, контрольные работы, коллоквиум, а также различные виды самостоятельной работы обучающихся по заданию преподавателя, направленные на развитие навыков профессиональной лексики, закрепление практических профессиональных компетенций, поощрение инициатив.

Лекция – систематическое, последовательное, монологическое изложение преподавателем учебного материала, как правило, теоретического характера.

В процессе лекций рекомендуется вести конспект лекций: кратко и схематично фиксировать основные идеи, выводы и обобщения лекции; выделять важные мысли, ключевые слова и термины. Необходимо отметить вопросы или материалы, которые вызывают затруднения, и попытаться найти ответы в рекомендованной литературе. Если разобраться в материале не удается, следует сформулировать вопрос и задать его преподавателю на консультации или во время семинарского (практического) занятия.

Семинар – это форма учебной деятельности, проводимая в учебном заведении под руководством преподавателя, где студенты активно участвуют в обсуждениях, практических заданиях и других формах взаимодействия.

Для успешной подготовки к семинару рекомендуется заранее ознакомиться с темой занятия и основными материалами, чтобы иметь возможность активно участвовать в обсуждении. Также полезно подготовить вопросы и идеи для обсуждения, что поможет глубже понять материал и продемонстрировать заинтересованность.

Аудиторная работа – активная работа студента на семинаре или лекции, его ответы на вопросы преподавателя и участие в дискуссии.

Для успешного участия в аудиторной работе студентам рекомендуется заранее ознакомиться с темой обсуждения, прочитать необходимые материалы и подготовить вопросы. Важно активно слушать и вовлекаться в дискуссию, высказывая свои мнения, аргументируя их. При ответах на вопросы преподавателя стоит быть уверенным, четким и логичным, опираясь на изученный материал. Также полезно поддерживать диалог с однокурсниками, чтобы обогатить обсуждение и расширить свои знания.

Домашнее задание – набор задач по темам недели.

При работе над домашними заданиями важно внимательно ознакомиться с требованиями и сроками выполнения. Рекомендуется разбивать задания на этапы, чтобы избежать перегрузки и лучше усвоить материал. Использовать различные источники

информации, включая учебники и онлайн-ресурсы, для более глубокого понимания темы.

Контрольная работа – письменная работа с набором задач, которые нужно решить за ограниченное время.

Цель контрольной работы – получить специальные знания по одной или нескольким темам дисциплины и продемонстрировать навыки их практического применения.

Коллоквиум – устные ответы на вопросы, список которых известен студенту заранее.

В процессе подготовки к коллоквиуму необходимо проанализировать учебные материалы, ознакомившись с лекциями, учебниками и дополнительными источниками, акцентируя внимание на ключевых темах. Рекомендуется создать структурированные конспекты, выделяя основные идеи, термины и формулы.

Самостоятельная работа – работа студентов, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины (модуля).

В процессе самостоятельной работы студенты взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи студента включают работу с конспектами лекций (обработка текста), повторное изучение учебных материалов планов и тезисов ответов, изучение дополнительных тем, выполнение учебно-исследовательских заданий и другое.

Система оценивания результатов обучения по дисциплине (модулю)

Критерии получения уровня и оценивания сформированности компетенций по дисциплине (модуле) «Современные формальные методы»

Оценивание уровня учебных достижений, обучающихся по дисциплине (модулю), осуществляется в виде текущего контроля успеваемости и промежуточной аттестации.

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме ***зачета с оценкой***, при этом проводится оценка компетенций, сформированных по дисциплине.

Для оценивания текущего контроля успеваемости и промежуточной аттестации используется десятибалльная шкала оценивания, которая соотносится с традиционной пятибалльной шкалой следующим образом, если иное не указано в силлабусе дисциплины:

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
10	Отлично	Зачтено	Студент полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину. Он самостоятельно и логически последовательно отвечает на все вопросы, акцентируя внимание на наиболее важном. Умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя ключевые моменты и устанавливая причинно-следственные связи. Четко формулирует ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет
9	Отлично	Зачтено	
8	Отлично	Зачтено	

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
			связывать теоретические аспекты дисциплины (модуля) с практическими задачами.
7	Хорошо	Зачтено	Студент обладает знаниями предмета почти в полном объеме рабочей программы и самостоятельно, логически последовательно и всесторонне отвечает на все вопросы, акцентируя внимание на наиболее значимых моментах. Он умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя его ключевые аспекты и устанавливая причинно-следственные связи. Формулирует свои ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные ситуационные задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
6	Хорошо	Зачтено	
5	Удовлетворительно	Зачтено	Студент обладает базовыми знаниями по дисциплине (модулю), но испытывает трудности при самостоятельных ответах и использует неточные формулировки. В ходе ответов он допускает ошибки, касающиеся сути вопросов. Студент способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования.
4	Удовлетворительно	Зачтено	
3	Не сдан	Не зачтено	Студент не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы.
2	Не сдан	Не зачтено	
1	Не сдан	Не зачтено	

Дисциплина (модуль) «Современные формальные методы» оценивается следующим образом:

Активность	Вес	Количество	Описание
Домашние задания	25%	10	Набор задач по темам недели
Аудиторная работа	15%	15	Активная работа студента на семинаре или лекции
Контрольная работа	35%	2	Письменная работа с набором задач, которые нужно решить за ограниченное время.
Зачет с оценкой	25%	1	Коллоквиум - устные ответы на вопросы, список которых известен студенту заранее

Формула расчёта итоговой оценки по дисциплине (модулю) «Современные формальные методы»: «0,25 × среднее за домашние задания + 0,15 × среднее за аудиторную работу + 0,35 × среднее за контрольные работы + 0,25 × зачет с оценкой».

Электронный документ

Текущий контроль успеваемости обучающихся по дисциплине (модуле)

Примерные домашние задания

1. **Hoare-тройка для binarySearch** – написать предусловие «отсортированный список», постусловие «результат = True ⇔ искdirect элемент присутствует», а также инвариант двойного деления (low ≤ high ∧ ∀i < low xs[i] < x ∧ ∀i > high xs[i] > x).
2. **Контракт insertSorted** – в стиле ACSL/Why3 указать, что входной список отсортирован; в постусловии требовать, чтобы полученный список был отсортирован, имел длину + 1 и содержал вставляемый элемент.
3. **Частичная корректность Euclid's GCD** – формализовать в Coq/Isabelle инвариант «gcd(a,b) = gcd(a_0,b_0) ∧ a,b ≥ 0», провести индукцию по рекурсии и завершить доказательство.
4. **Контрпример в SMT-solver** – составить файл *.smt2* с запросом «¬(a·b ≥ a + b)», запустить Z3 и привести найденный пример (например, a = 0, b = 1).
5. **Kripke-модель банкомата** – описать состояния Idle, CardInserted, PinEntered, Dispensing, указать переходы, сформулировать два свойства CTL: (i) *доступность* **AG EF idle**; (ii) *безопасность* "не выдавать деньги без ввода ПИН" → **AG (¬ dispense ∨ pinEntered)**.

Примерные задания для семинаров

1. **Спецификации и контракты** – каждая подгруппа получает небольшую функцию (поиск минимума, реверс списка, подсчёт длины) и формулирует Hoare-тройку и ACSL-контракт. После обсуждения сравниваются варианты формулировок.
2. **Инварианты циклов** – на доске демонстрируется алгоритм суммы первых *n* чисел; группы формулируют инвариант, проверяют базу и шаг, ищут типичные ошибки (слишко слабый/сильный инвариант).
3. **SMT-solver в действии** – студенты пишут несколько запросов в SMT-LIB (противоречие, проверка свойства массива) и запускают Z3 интерактивно, интерпретируя **sat/unsat** и полученную модель.
4. **Model checking с SPIN** – совместно строится модель «читатель-писатель» в Promela, задаётся LTL-формула «если писатель пишет, то потом читатель не читает», проводится запуск **spin**, обсуждаются найденные конфликты и способы их исправления.
5. **Зависимые типы в Lean** – в парах решается задача: " фиксированной длины, каждый элемент – положительное число". Реализуется тип **PosVec α n**.
6. **Proof-assistant showdown** – одна и та же лемма (**∀n. n ≤ n+1**) доказывается в Coq, Isabelle и Lean; студенты отмечают различия в синтаксисе, тактиках и уровне автоматизации.
7. **Статический анализ C-кода** – на ноутбуках устанавливается Frama-C, анализируется функция **int clamp(int x,int lo,int hi)**. Формулируются pre/post-условия и проверяется их выполнение.
8. **Коллекция контрпримеров** – каждый студент придумывает контрпример к популярному «естественному» утверждению (например, "каждое нечётное число простое"), затем обсуждается, как его можно обнаружить с помощью SMT-solver.

Примерные задания для контрольной работы

1. **Спецификация** – написать Hoare-тройку для **insertSorted**, указать предусловие, постусловие и инвариант внутреннего цикла. Оценка за правильность формализма и корректность инварианта.
2. **Доказательство индукцией по пути** – с помощью принципа J доказать, что **rev (rev xs) = xs**. Требуется указать, как применяется J к базовому случаю **refl**.

Электронный документ

3. **SMT-задача** – сформулировать в SMT-LIB утверждение «если массив длиной 3 упорядочен, то первый элемент ≤ третий», запустить Z3 и указать результат (**sat**).

4. **Model checking без инструмента** – на основе Kripke-модели банкомата (см. семинар) сформулировать CTL-формулу «в любой момент, если пользователь вставил карту, то в конечном итоге система вернёт карту», и доказать вручную, истинна она или ложна.

5. **Зависимые типы в Lean** – определить тип **Fin (n+1)** и функцию **succFin : Fin n → Fin (n+1)**. Доказать лемму **∀i. i < n → succFin i = ⟨i.val+1, …⟩**. Оценка за корректность определения и лаконичность доказательства.

Примерные задания для коллоквиума

1. Что такое **спецификация** в формальных методах и как она связана с поведением программы?
2. Дайте определение **инварианта цикла** и перечислите три его стадии (инициализация, сохранение, завершение).
3. В чём разница между **SMT-solver** и **SAT-solver** ?
4. Что проверяет **принцип J** (индукция по пути) в Coq/HoTT?
5. Чем **model checking** отличается от **formal verification** (доказательство в proof-assistant)?
6. Кратко опишите **унивалентность** в HoTT и её смысл для типов.
7. Приведите пример **зависимого типа** и объясните, почему он полезен для спецификации.
8. Какие два основных преимущества применения **proof assistants** в промышленной разработке программного обеспечения?

Задания для промежуточной аттестации по дисциплине (модулю)

№ п/п	Задание	Ответ	Компетенция
1.	Сформулировать **Hoare-тройку** для функции **binarySearch :: [Int] → Int → Bool**, которая ищет элемент **x** в отсортированном массиве **xs**. Указать предусловие, постусловие и инвариант цикла.	Предусловие: **isSorted xs**. Постусловие: **res = True ⇔ x ∈ xs**. Инвариант цикла (внутри бинарного поиска): **low ≤ high ∧ (∀i<low. xs!!i < x) ∧ (∀i>high. xs!!i > x)**. Тройка: **{isSorted xs} binarySearch xs x {res = True ⇔ x ∈ xs}**.	УК-1
2.	Записать **инвариант** цикла Евклида (вычисление НОД) и объяснить, как он обеспечивает сохранение свойства **gcd(a₀,b₀) = gcd(a,b)** на каждой итерации.	Инвариант: **gcd(a₀,b₀) = gcd(a,b) ∧ a ≥ 0 ∧ b ≥ 0**. При каждом шаге **a' = b**, **b' = a mod b** сохраняется **gcd(a,b) = gcd(b, a mod b)**. Таким образом инвариант остаётся истинным до выхода из цикла, когда **b = 0** и результат **a** – НОД.	УК-1
3.	Доказать **частичную корректность** алгоритма сортировки вставкой (**insertionSort**). Указать пред- и постусловия, а также инвариант вложенного цикла.	Предусловие: **xs** – произвольный список. Постусловие: результат **ys** – перестановка **xs** и упорядочен (**sorted ys**). Инвариант внешнего цикла (на i-й итерации): первые **i** элементов уже отсортированы. Инвариант внутреннего цикла (перемещение текущего элемента):	УК-1

		элемент **key** перемещается вправо, пока **key < a[j]**. Доказательство состоит из базового случая (**i=0**) и сохранения инварианта после каждой итерации.	
4.	Составить **формальную спецификацию** (пред-/постусловия) для функции **insertSorted :: [Int] → Int → [Int]**, вставляющей элемент в уже отсортированный список.	`{isSorted xs} insertSorted xs x {ys	ОПК-4
5.	С помощью **SMT-solver Z3** найти контрпример к утверждению «для любых целых **a,b** выполняется **a*b ≥ a + b**». Привести ввод-команду и найденный пример.	**smt2\n(set-logic QF_LIA)\n(declare-const a Int)\n(declare-const b Int)\n(assert (not (>= (* a b) (+ a b))))\n(check-sat)\n(get-model)\n** Результат (один из): **a = 0**, **b = 1** (0·1 = 0 < 0+1).	ПК-2
6.	Описать **символьное исполнение** простого фрагмента кода на языке IMP и указать, какие ограничения (path-explosion, неточность естественных чисел) могут привести к потере точности анализа.	Пример кода: **x := 0; while (x < n) do x := x + 2 od**. Символьные состояния: **x = 2·k** после k итераций, условие **2·k < n**. Ограничения: (1) экспоненциальный рост путей при ветвлениях, (2) отсутствие точного моделирования переполнения и округления, (3) необходимости абстракций (интервалы/полиномиальные шаблоны).	ОПК-4
7.	Сформулировать **модель Kripke** для простого банковского автомат (ATM) с состояниями **Idle**, **CardInserted**, **PinEntered**, **Dispensing**. Перечислить атомарные пропозиции и написать формулу CTL, проверяющую «из любого состояния существует путь, где система возвращается в **Idle**».	Состояния: **s0=Idle**, **s1=CardInserted**, **s2=PinEntered**, **s3=Dispensing**. Пропозиции: **idle**, **card**, **pinOk**, **dispense**. Переходы: **Idle → CardInserted → PinEntered → Dispensing → Idle**. CTL-формула: **AG EF idle** (во всех достижимых состояниях существует путь, который в конечном итоге достигает **idle**).	ОПК-4
8.	Записать **темпоральное свойство** LTL для системы «всегда, когда пользователь вводит PIN, позже обязательно произойдёт выдача наличных», и проверить его в **NuSMV** (коротко описать модель и результат проверки).	LTL-формула: **G (pinEntered → F dispense)**. Модель в NuSMV: переменная **state** принимает значения **{idle, pinEntered, dispense}** и переходы **idle -> pinEntered**, **pinEntered -> dispense**, **dispense -> idle**. Проверка выводит **-- specification G (pinEntered -> F dispense) is true**.	ОПК-4
9.	Показать **соответствие Карри** на примере функции **add : Nat → Nat → Nat** в Coq. Сформулировать тип	```coq\nRequire Import Arith.\nFixpoint add (n m : nat) : nat :=\n match n with\n		ПК-2

	как утверждение и написать короткое доказательство (**Proof. induction n. ... Qed.**).		
10.	Описать **зависимый тип** **Vec** (вектор фиксированной длины) в Lean и написать функцию **head** с типом **Vec (n+1) α → α**.	```lean\ndef Vec (α : Type) : Nat → Type\n	ПК-2
11.	Сформулировать **контрактное программирование** для функции **findMax :: [Int] → Int** (поиск максимального элемента) и показать, как контракт может быть проверен автоматически в **Dafny** .	Предусловие: **xs ≠ []** . Постусловие: **result = max xs** . В Dafny: **dafny\nmethod findMax(xs: seq<int>) returns (m: int)\n requires xs.Length > 0\n ensures forall i :: 0 <= i < xs.Length ==> m >= xs[i]\n{\n var cur := xs[0];\n var i := 1;\n while i < xs.Length\n invariant 0 <= i <= xs.Length\n invariant forall j :: 0 <= j < i ==> cur >= xs[j]\n {\n if xs[i] > cur { cur := xs[i]; }\n i := i + 1;\n }\n m := cur;\n}\n**	ПК-2
12.	Перечислить **два плюса и два минуса** применения **proof assistants** (Coq, Isabelle, Lean) в промышленной разработке программного обеспечения.	Плюсы: (1) гарантированная формальная проверка корректности критических компонентов, (2) возможность извлечения проверяемого кода из доказательства. Минусы: (1) высокий порог входа и длительное время написания доказательств, (2) ограниченная интеграция в обычные CI/CD-конвейеры и необходимость поддержки специалистов-формализаторов.	ОПК-4
13.	Оформить Hoare-тройку для рекурсивного **gcd :: Int → Int → Int** . Указать предусловие, постусловие и инвариант рекурсии.	**Предусловие:** **a ≥ 0 ∧ b ≥ 0** . **Постусловие:** **result = gcd a_0 b_0** (где **a_0 , b_0** – исходные аргументы). **Инвариант рекурсии:** **gcd a b = gcd a_0 b_0 ∧ a ≥ 0 ∧ b ≥ 0** . **Тройка:** **{a ≥ 0 ∧ b ≥ 0} gcd a b {result = gcd a_0 b_0}** .	**УК-1**
14.	С помощью SMT-solver Z3 найти контрпример к утверждению «для любых целых **a,b** выполняется **a·b ≥ a + b** ». Привести ввод-команду и найденный пример.	**smt2\n(set-logic QF_LIA)\n(declare-const a Int)\n(declare-const b Int)\n(assert (not (>= (* a b) (+ a b))))\n(check-sat)\n(get-model)\n** \n**Контрпример (один из вариантов):** **a = 0** , **b = 1** ($0 \cdot 1 = 0 < 0 + 1$).	**ПК-2**
15.	Выполнить символический анализ фрагмента IMP и указать, сколько различных путей будет сгенерировано (учитывать только ветвления).	Два независимых условных оператора ⇒ $2 \times 2 = 4$ пути. Символьные состояния: 1 **x>0 ∧ y+1=0 → z=10** 2 **x>0 ∧ y+1≠0 → z=20**	**ОПК-4**

	if (x>0) then y:=y+1 else y:=y-1; if (y=0) then z:=10 else z:=20;	3) x≤0 ∧ y-1=0 → z=10 4) x≤0 ∧ y-1≠0 → z=20.	
16.	Написать небольшую модель в **Promela** (SPIN) для протокола *producer-consumer* с буфером из-одного элемента. Добавить LTL-формулу, проверяющую, что «каждый произведённый элемент будет потреблён». Указать результат проверки.	promela\nchan buf = [1] of {int};\n\nproctype Producer() {\n int i = 0;\n do :: buf!i; i = i + 1 od\n}\n\nproctype Consumer() {\n int x;\n do :: buf?x; skip od\n}\n\ninit { run Producer(); run Consumer(); }\n \n**LTL-формула:** [](buf? -> <> buf!) (если элемент считан, то в конечном итоге будет отправлен). Запуск spin -run model.pml даёт сообщение **«no acceptance cycle found»** → формула считается выполненной (каждый отправленный элемент будет получен).	**ПК-2**
17.	Сформулировать и проверить в **PRISM** свойство «с вероятностью 1 система, находящаяся в состоянии Idle, в течение 5 шагов обязательно перейдёт в состояние Processing».	**Модель:** два состояния Idle и Processing; переход Idle → Processing с 0.8, оставаться в Idle с 0.2. **Формула PCTL:** P=? [F<=5 (state = Processing)]. После prism model.prism -prop "P=? [F<=5 (state=Processing)]" получаем 1.000000 ⇒ свойство выполнено с вероятностью 1.	**ПК-2**
18.	В **Coq** доказать лемму: forall n m k : nat, n <= m -> n + k <= m + k. Привести минимальный скрипт.	coq\nRequire Import Arith.\n\nLemma le_plus_r : forall n m k, n <= m -> n + k <= m + k.\nProof.\n intros n m k H.\n apply Nat.add_le_mono_r. exact H.\nQed.\n	**ПК-2**