ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

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

Рабочая программа дисциплины (модуля)
«Разработка на Rust»

Направление подготовки: *02.03.01 Математика и компьютерные науки*

Направленность (профиль) подготовки: *Прикладной искусственный интеллект*

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) «Разработка на Rust» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования – бакалавриат по направлению 02.03.01 «Математика и компьютерные науки», профиль «Прикладной искусственный интеллект», утвержденный приказом Министерства науки и высшего образования Российской Федерации № 807 от 23.08.2017 года.

Изучение дисциплины (модуля) «Разработка на Rust» формирует у обучающегося уникальные навыки разработки программного обеспечения, сочетающего производительность системных языков с гарантиями безопасности времени выполнения. Освоение Rust позволяет участвовать в создании критически важных систем, включая операционные компоненты, встраиваемые системы и высоконагруженные сервисы, где недопустимы утечки памяти и неопределённое поведение.

Место дисциплины (модуля) в структуре образовательной программы

Настоящая дисциплина (модуль) включена в учебный план по программе подготовки бакалавриата по направлению 02.03.01 «Математика и компьютерные науки», профиль «Прикладной искусственный интеллект» и входит в вариативную часть Блока 1, формируемую участниками образовательных отношений, как дисциплина по выбору.

Дисциплина (модуль) изучается на 3 курсе в 5 семестре. Доступна к изучению после успешного освоения дисциплины (модуля): «Язык программирования Python. Основной».

Цель изучения дисциплины (модуля): подготовить обучающегося к разработке высоконадёжных, производительных и безопасных системных приложений с использованием современных механизмов владения памятью, многопоточности и низкоуровневого программирования на языке Rust.

Задачи изучения дисциплины (модуля):

— освоить модель памяти Rust, включая механизмы владения, заимствования и времени жизни ссылок, для предотвращения ошибок управления памятью;

— научиться применять умные указатели, внутреннюю изменяемость и системы типов для построения сложных и безопасных структур данных;

— развить навыки разработки многопоточных и асинхронных приложений с обеспечением потокобезопасности и отсутствия состояний гонки;

— освоить методы интеграции с C, написание безопасных обёрток и использование небезопасных блоков с контролем рисков;

— сформировать компетенции в проектировании и отладке системных программ, включая тестирование, профилирование и разработку под no-std окружения.

В результате освоения дисциплины (модуля), обучающийся должен:

знать:

— механизмы обеспечения безопасности памяти и предотвращения ошибок на основе владения и заимствования;

— особенности внутреннего устройства модели памяти Rust;

— классификацию и отличия механизмов полиморфизма в Rust;

— концепцию времени жизни ссылок, правила их неявного выведения и принципы управления разделяемым состоянием;

— определение безопасной многопоточности и механизмы её обеспечения для пользовательских типов;

— основные принципы модели исполнения асинхронного кода;

— методы обработки ошибок без использования исключений;

— закономерности взаимодействия Rust-кода с языком Си и границы применимости небезопасных блоков.

уметь:

— разрабатывать многофайловые проекты и настраивать модульную архитектуру с использованием рабочих пространств и системы пакетов;

— применять механизмы внутренней изменяемости и умные указатели для построения сложных графов объектов;

— использовать стандартный инструментарий экосистемы для написания модульных, интеграционных и документирующих тестов;

— оптимизировать код и размещение данных в памяти, применяя профилировщики и средства замера производительности;

— настраивать асинхронное сетевое взаимодействие и операции ввода-вывода с помощью среды выполнения асинхронного кода;

— анализировать и обрабатывать граничные случаи бизнес-логики, кодируя инварианты в систему типов;

— интегрировать существующий код на языке Си в проекты на Rust, используя средства автоматизации межъязыкового взаимодействия и создавая безопасные обёртки над небезопасными функциями;

— составлять декларативные макросы для устранения шаблонного кода.

владеть:

— навыками написания идиоматичного Rust-кода с избеганием частых антипаттернов;

— опытом проектирования надёжных системных приложений сбора и обработки данных;

— методиками предотвращения состояний гонки и взаимных блокировок с использованием каналов и примитивов синхронизации;

— практикой низкоуровневой отладки и поиска неопределённого поведения с применением профилировщиков и санитайзеров;

— способами адаптации и сборки программного обеспечения для сред без стандартной библиотеки операционной системы.

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:

Код и содержание компетенции	**Индикатор компетенции и 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 разработки, 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.
УК-3. Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде *(соответствует УНК-02)*	УК-3.1. (УНК-02.01.) Принимает участие в групповом взаимодействии в ходе профессиональной деятельности (базовый) УНК-02.01.01. У-1. Проявляет понимание своей роли в команде УНК-02.01.01. У-2. Умеет ясно выражать свои мысли в группе УНК-02.01.01. У-3. Способен слушать и учитывать мнения других участников команды. УНК-02.01.01. У-4. Умеет давать конструктивную обратную связь УНК-02.01.01. У-5. Умеет работать в команде, проявлять инициативу и поддерживать коллег УНК-02.01.01. У-6. Обладает навыками разрешения конфликтных ситуаций и поиска компромиссов УНК-02.01.01. З-1. Знает основы командной работы, роли и ответственности каждого участника.
	УК-3.2. (УНК-02.02.) Проявляет лидерство и осуществляет наставничество (базовый) УНК-02.02.01. У-1. Умеет мотивировать и вдохновлять команду для достижения общих целей УНК-02.02.01. У-2. Способен организовывать работу группы, распределять обязанности и контролировать выполнение задач УНК-02.02.01. У-3. Умеет делиться знаниями и опытом, оказывать поддержку и осуществлять наставничество коллегам УНК-02.02.01. У-4. Способен принимать ответственные решения и демонстрировать пример профессионализма УНК-02.02.01. З-1. Знает основы лидерства, мотивации и командообразования УНК-02.02.01. З-2. Знает принципы эффективного наставничества и развития персонала УНК-02.02.01. З-3. Знает методы оценки эффективности работы команды и индивидуальных участников
ОПК-6. Способен разрабатывать алгоритмы и компьютерные программы,	ОПК-6.1. Знает принципы алгоритмизации, основные структуры данных, парадигмы программирования, методы разработки, тестирования и отладки компьютерных программ.

пригодные для практического применения	ОПК-6.2. Умеет разрабатывать, документировать, тестировать и отлаживать алгоритмы и компьютерные программы, пригодные для практического применения, с использованием современных языков и сред разработки.
	ОПК-6.3. Имеет практический опыт разработки и сопровождения компьютерных программ и программных комплексов для решения прикладных задач.
ППК-ДА3. Способен проводить аналитические исследования с применением технологий больших данных	ППК-ДА3.2. – Обеспечивает соответствие результатов анализа бизнес-задачам заказчика (средний) ППК-ДА3.2. З-1. Знает методы перевода бизнес-требований в аналитические задачи ППК-ДА3.2. З-2. Знает ключевые бизнес-метрики в предметной области
ППК-Р5. Способен разрабатывать требования и проектировать программное обеспечение	ППК-Р5.1. Анализирует возможности реализации требований к компьютерному программному обеспечению (средний) ППК-Р5.1. З-1. Знает возможности существующей программно-технической архитектуры. ППК-Р5.1. З-2. Знает возможности современных и перспективных средств разработки программных продуктов, технических средств. ППК-Р5.1. З-3. Знает методологии разработки компьютерного программного обеспечения и технологии программирования. ППК-Р5.1. У-1. Умеет проводить сбор и систематизацию требований к компьютерному программному обеспечению. ППК-Р5.1. У-2. Умеет выявлять взаимосвязи и документировать требования к компьютерному программному обеспечению ППК-Р5.1. У-3. Умеет вырабатывать варианты реализации требований к компьютерному программному обеспечению.
	ППК-Р5.2. Разрабатывает технические спецификаций на программные компоненты и их взаимодействие (средний) ППК-Р5.2. З-1. Знает методы и средства проектирования компьютерного программного обеспечения. ППК-Р5.2. З-2. Знает методы и средства проектирования программных интерфейсов. ППК-Р5.2. У-1. Умеет выбирать средства реализации требований к компьютерному программному обеспечению. ППК-Р5.2. У-2. Умеет вырабатывать варианты реализации компьютерного программного обеспечения. ППК-Р5.2. У-3. Умеет проводить оценку и обоснование рекомендуемых решений.
	ППК-Р5.3. Проектирует компьютерное программное обеспечение (продвинутый) ППК-Р5.3. З-1. Знает принципы построения и виды архитектуры компьютерного программного обеспечения. ППК-Р5.3. З-2. Знает типовые решения, библиотеки программных модулей, шаблоны, классы объектов, используемые при разработке компьютерного программного обеспечения. ППК-Р5.3. З-3. Знает нормативно-технические документы (стандарты), определяющие требования к технической документации на компьютерное программное обеспечение. ППК-Р5.3. З-4. Знает методы и средства проектирования компьютерного программного обеспечения.

	ППК-Р5.3. У-1. Умеет разрабатывать и изменять архитектуру компьютерного программного обеспечения и согласовывать ее с системным аналитиком и архитектором программного обеспечения. ППК-Р5.3. У-2. Умеет проектировать структуры данных. ППК-Р5.3. У-3. Умеет проектировать программные интерфейсы.
	ППК-Р5.4. Выполняет логическое проектирование системы (продвинутый) ППК-Р5.4. З-1. Знает устройство и функционирование ИТ-систем/продуктов ППК-Р5.4. З-2. Знает методы моделирования и описания устройства и функционирования ИТ-систем/продуктов, их частей, обеспечения и окружения ППК-Р5.4. З-3. Знает базовые технологии взаимодействия и интеграции систем и компонентов ППК-Р5.4. З-4. Знает методы функциональной декомпозиции ИТ-систем ППК-Р5.4. У-1. Умеет декомпозировать ИТ-системы и ИТ-продукты на подсистемы и элементы поставки ППК-Р5.4. У-2. Умеет описывать интерфейсы пользователя на логическом уровне ППК-Р5.4. У-1. Умеет описывать интеграции со смежными системами на логическом уровне
ППК-Р6. Способен участвовать в промышленной разработке программного обеспечения	ППК-Р6.1. Работает в соответствии с промышленными методологиями разработки (средний) ППК-Р6.1. З-1. Знает принципы Agile и их применение в промышленных проектах. ППК-Р6.1. З-2. Знает процессы code review, принципы коллективного владения кодом (collective code ownership). ППК-Р6.1. У-1. Умеет оценивать объем задачи и срок ее выполнения, участвовать в планировании спринтов. ППК-Р6.1. У-2. Умеет работать в команде с использованием инструментов управления проектами.
	ППК-Р6.2. Использует инструменты промышленной разработки (средний) ППК-Р6.2. З-1. Знает принципы Continuous Integration and Continuous Delivery (CI/CD). ППК-Р6.2. З-2. Знает системы мониторинга и логирования в продуктивной среде. ППК-Р6.2. У-1. Умеет настраивать потоки работ CI/CD. ППК-Р6.2. У-2. Умеет работать с контейнеризацией и оркестрацией. ППК-Р6.2. У-2. Умеет настраивать мониторинг в продуктивной среде.
	ППК-Р6.3. Разрабатывает масштабируемый и поддерживаемый код (продвинутый) ППК-Р6.3. З-1. Знает принципы чистого кода, SOLID, DRY, KISS и др. ППК-Р6.3. З-2. Знает принципы предметно-ориентированного проектирования (ПОП) программного обеспечения. ППК-Р6.3. З-3. Знает паттерны проектирования и антипаттерны. ППК-Р6.3. У-1. Умеет разрабатывать модульный и тестируемый программный код. ППК-Р6.3. У-2. Умеет выполнять модульное, интеграционное и нагрузочное тестирование.

	ППК-Р6.3. У-3. Умеет проводить рефакторинг для повышения качества кода. ППК-Р6.3. У-4. Умеет применять принципы ПОП при разработке программного обеспечения на языках программирования высокого уровня абстракций и в LowCode и NoCode системах.
	ППК-Р6.4. Участвует в развертывании и поддержке программного обеспечения (продвинутый) ППК-Р6.4. З-1. Знает стратегии развертывания промышленного программного обеспечения. ППК-Р6.4. З-2. Знает основы работы с облачными платформами. ППК-Р6.4. У-1. Умеет развертывать приложения в облаке. ППК-Р6.4. У-2. Умеет обнаруживать и устранять инциденты с работой продуктивного программного обеспечения.
ППК-Р7. Способен применять искусственный интеллект (ИИ) для генерации и отладки программного кода	ППК-Р7.1. Применяет ИИ-инструменты для генерации программного кода (базовый) ППК-Р7.1. З-1. Знает принципы работы современных генеративных ИИ-моделей для генерации кода. ППК-Р7.1. З-2. Знает ограничения и риски использования ИИ-генерации (безопасность, качество кода, лицензирование). ППК-Р7.1. У-1. Умеет формулировать корректные текстовые запросы (промты) для генерации кода. ППК-Р7.1. У-2. Умеет интегрировать ИИ-инструменты в среду разработки.
	ППК-Р7.2. Использует ИИ для анализа и отладки кода (средний) ППК-Р7.2. З-1. Знает методы ИИ-анализа кода. ППК-Р7.2. З-2. Знает форматы и инструменты для автоматизированного тестирования с ИИ. ППК-Р7.2. У-1. Умеет настраивать ИИ-инструменты для поиска уязвимостей. ППК-Р7.2. У-2. Умеет интерпретировать рекомендации ИИ по исправлению кода.
	ППК-Р7.3. Оптимизирует код с помощью ИИ (продвинутый) ППК-Р7.3. З-1. Знает методы ИИ-оптимизации. ППК-Р7.3. З-2. Знает критерии качества кода, применяемые ИИ-системами. ППК-Р7.3. У-1. Умеет использовать ИИ для рефакторинга. ППК-Р7.3. У-2. Умеет проверять корректность оптимизаций, предложенных ИИ.
	ППК-Р7.4. Оценивает этические и профессиональные аспекты применения ИИ в разработке (продвинутый) ППК-Р7.4. З-1. Знает этические нормы использования ИИ (конфиденциальность, плагиат кода и т.п.). ППК-Р7.4. З-2. Знает лицензионные ограничения сгенерированного кода. ППК-Р7.4. У-1. Умеет проверять код на соответствие стандартам после ИИ-генерации. ППК-Р7.4. У-2. Умеет документировать использование ИИ в разработке.

ППК-ДА4. Способен выявлять бизнес-проблемы или бизнес-возможности	ППК-ДА4.2. Выявление истинных бизнес-проблем или бизнес-возможностей (продвинутый) ППК-ДА4.2. З-1. Знает предметную область и специфику деятельности организации в объеме, достаточном для решения задач бизнес-анализа ППК-ДА4.2. З-2. Знает инструменты, техники анализа бизнес-ситуации и предметной области, включая анализ данных. ППК-ДА4.2. З-3. Знает перспективные и существующие цифровые технологии и цифровые возможности для бизнеса в контексте предметной области и специфики деятельности организации.
РГ-1. Способен руководить процессами разработки компьютерного программного обеспечения	РГ-1.1. Руководит разработкой программного кода (средний) РГ-1.1. З-1. Знает методы принятия управленческих решений. РГ-1.1. З-2. Знает основные принципы и методы управления персоналом РГ-1.1. З-1 Знает технологии межличностной и групповой коммуникации в деловом взаимодействии, основы конфликтологии РГ-1.1. У-1. Способен распределять задачи на разработку программного кода между исполнителями. РГ-1.1. У-2. Применяет лучшие мировые практики оформления программного кода. РГ-1.1. У-3. Оценивает качество и эффективность программного кода.
	РГ-1.2. Руководит проверкой работоспособности компьютерного программного обеспечения (средний) РГ-1.2. З-1. Знает основные методы измерения и оценки характеристик компьютерного программного обеспечения РГ-1.2. З-2. Знает методы и средства проверки работоспособности компьютерного программного обеспечения РГ-1.2. З-3. Знает методы и средства рефакторинга и оптимизации программного кода РГ-1.2. З-4. Знает о возможности отказа частей системы в результате воздействий внутреннего и внешнего нарушителя (хакер, неосторожный пользователь, программист, поставщик компонентов) РГ-1.2. У-1. Умеет распределять задачи на проверку работоспособности компьютерного программного обеспечения между исполнителями. РГ-1.2. У-2. Умеет оценивать результаты проверки работоспособности компьютерного программного обеспечения РГ-1.2. У-3. Умеет принимать управленческие решения по результатам проверки работоспособности компьютерного программного обеспечения об исправлении ошибок, рефакторинге, оптимизации и инспекции кода РГ-1.2. У-4. Умеет включать поиск возможных отказов из воздействий внутреннего или внешнего нарушителя (хакер, неосторожный пользователь, программист, поставщик компонентов) в процедуры инспекции программного кода.
	РГ-1.3. Руководит интеграцией программных модулей и компонентов компьютерного программного обеспечения (продвинутый) РГ-1.3. З-1. Знает методы и средства сборки модулей и компонентов компьютерного программного обеспечения

	РГ-1.3. З-2. Знает методы и программные интерфейсы взаимодействия компьютерного программного обеспечения с внешними программными компонентами РГ-1.3. З-3. Знает методы проектирования и разработки программных интерфейсов взаимодействия внутренних модулей компьютерного программного обеспечения РГ-1.3. З-4. Знает методы и средства разработки процедур для развертывания компьютерного программного обеспечения РГ-1.3. У-1. Умеет назначать задания на разработку процедур интеграции, сборку, подключение к внешней среде, проверку работоспособности выпусков программного продукта РГ-1.3. У-2. Умеет оценивать результаты выполнения назначенных заданий на разработку процедур интеграции, сборку, подключение к внешней среде, проверку работоспособности выпусков программного продукта РГ-1.3. У-3. Умеет принимать управленческие решения по результатам проверки работоспособности выпусков программного продукта (решение о выпуске/невыпуске версии, отправка задач на доработку, добавление новых задач, передача на тестирование)
РГ-4. Способен организовать промышленную разработку программного обеспечения	РГ-4.1. Внедряет промышленные методологии разработки (средний) РГ-4.1. З-1. Знает принципы Agile и их применение в промышленных проектах. РГ-4.1. З-2. Знает процессы code review, принципы коллективного владения кодом (collective code ownership). РГ-4.1. У-1. Умеет оценивать объем задач и срок их выполнения. РГ-4.1. У-2. Умеет распределять задачи между участниками команды. РГ-4.1. У-3. Умеет организовать работу в команде с использованием инструментов управления проектами.
	РГ-4.3. Организует разработку масштабируемого и поддерживаемого кода (продвинутый) РГ-4.3. З-1. Знает принципы чистого кода, SOLID, DRY, KISS и др. РГ-4.3. З-2. Знает принципы предметно-ориентированного проектирования (ПОП) программного обеспечения. РГ-4.3. З-3. Знает паттерны проектирования и антипаттерны. РГ-4.3. У-1. Умеет управлять разработкой модульного и тестируемого программного кода. РГ-4.3. У-2. Умеет организовать выполнение модульного, интеграционного и нагрузочного тестирования. РГ-4.3. У-3. Умеет управлять техническим долгом, в том числе организовать рефакторинг для повышения качества кода. РГ-4.3. У-4. Умеет применять принципы ПОП при разработке программного обеспечения.
	РГ-4.4. Организует развертывание и поддержку программного обеспечения (продвинутый) РГ-4.4. З-1. Знает стратегии развертывания промышленного программного обеспечения. РГ-4.4. У-2. Умеет управлять обнаружением и устранением инцидентов с работой продуктивного программного обеспечения.

3. Тематический план

№ п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Аудиторная работа		Контроль	Самостоятельная работа	
		Лекции	Практические занятия			
1	Введение в Rust, основные особенности языка	6	6		26	Домашнее задание Контрольная работа
2	Поддержка проектов на Rust	6	6		26	Домашнее задание Контрольная работа
3	Архитектура, память и многопоточность	8	8		34	Домашнее задание Контрольная работа
4	Метапрограммирование и механизмы FFI и Unsafe	10	10		42	Домашнее задание Контрольная работа
	Зачет			2		Проект
	Итого:	***30***	***30***	***2***	***128***	
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 Rust, основные особенности языка	Введение в Rust и базовые типы Владение и заимствование (Ownership & Borrowing) Структуры и перечисления
2	Поддержка проектов на Rust	Трейты и обобщения Модули и рабочее пространство Тестирование и качество
3	Архитектура, память и многопоточность	Умные указатели и внутренняя мутабельность Продвинутое управление памятью и Lifetimes Data Layout и Динамическая диспетчеризация Многопоточность с Tokio
4	Метапрограммирование и механизмы FFI и Unsafe	Углубленное изучение по прошедшему материалы Асинхронный Rust и Сети Unsafe Rust и FFI Системное программирование и Bare-Metal

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. Макнамара, Т. Rust в действии : практическое руководство / Т. Макнамара. - Санкт-Петербург : БХВ-Петербург, 2023. - 528 с. - ISBN 978-5-9775-1166-7. - Текст : электронный. - URL: https://znanium.com/catalog/product/2123400.

2. Клабник, С. Программирование на Rust / С. Клабник, К. Николс. - Санкт-Петербург : Питер, 2021. - 592 с. - ISBN 978-5-4461-1656-0. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2142469.

Дополнительная литература:

1. Лонг, Т. Хороший код, плохой код : практическое руководство / Т. Лонг. - Санкт-Петербург : БХВ-Петербург, 2023. - 432 с. - ISBN 978-5-9775-1790-4. - Текст : электронный. - URL: https://znanium.com/catalog/product/2122896.

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.

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 (модулю), 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
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) «Разработка на Rust» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, домашние задания, контрольные работы, проект, 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.

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.

Проект – исследовательская работа по дисциплине (модулю) и презентация результатов.

Для успешной подготовки к проекту рекомендуется: четко определить цели и задачи проекта; составить план работы, разбив проект на этапы с указанием сроков выполнения

каждого из них; использовать разнообразные источники информации и инструменты для исследования темы; регулярно проверять прогресс и вносить коррективы в план, если это необходимо.

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 «Разработка на Rust»

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***, 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 связывать теоретические аспекты дисциплины (модуля) с практическими задачами.
9	Отлично	Зачтено	
8	Отлично	Зачтено	
7	Хорошо	Зачтено	Студент обладает знаниями предмета

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
6	Хорошо	Зачтено	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.
5	Удовлетворительно	Зачтено	Студент обладает базовыми знаниями по дисциплине, 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) «Разработка на Rust» оценивается следующим образом:

Активность	Вес	Количество	Описание
Домашнее задание	20%	14	Набор задач по темам недели
Контрольная работа	40%	4	Письменная работа с набором задач, которые нужно решить за ограниченное время
Зачет	40%	1	Проект - Исследовательская работа по дисциплине (модулю) и презентация результатов

Формула расчёта итоговой оценки по дисциплине (модулю) «Разработка на Rust»: «0,2 × среднее за домашние задания + 0,4 × среднее за контрольные работы + 0,4 × зачет».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

Домашнее задание 1: Основы Rust, владение и структуры

1. Установите Rust и настройте среду разработки (rustup, Cargo, IDE);
2. Создайте новый бинарный проект с помощью Cargo и напишите программу, которая выводит: Привет, я изучаю Rust!;
3. Добавьте комментарий в код: // Мое первое задание на Rust;
4. Объявите структуру User с полями name (String), age (u32) и active (bool), создайте экземпляр и выведите его;
5. Продемонстрируйте концепцию владения: передайте строку в функцию и попробуйте использовать её после вызова;
6. Исправьте ошибку с помощью заимствования (&String) и поясните, почему это работает;
7. Закоммитьте изменения, отправьте в ветку feature/ownership в GitLab и создайте Merge Request.

Домашнее задание 2: Модули, трейты и тестирование

1. Создайте библиотечный проект с именем data_structures и настройте рабочее пространство (workspace);
2. Реализуйте модуль shapes, содержащий перечисление Shape и структуры Circle, Rectangle;
3. Определите трейт Area, реализуйте его для каждой фигуры;
4. Добавьте обобщённую функцию print_area<T: Area>(shape: &T), которая выводит площадь;
5. Напишите модульные тесты для проверки корректности вычисления площади;
6. Создайте документационные комментарии и сгенерируйте документацию с помощью cargo doc --open;
7. Закоммитьте в ветку feature/modules-testing и создайте Merge Request с описанием реализованных компонентов.

Домашнее задание 3: Умные указатели, многопоточность и FFI

1. Реализуйте граф объектов с помощью Rc<RefCell<T>>, чтобы продемонстрировать внутреннюю изменяемость и разделяемое владение;
2. Создайте асинхронное приложение с использованием tokio, которое каждые 3 секунды выводит сообщение в консоль;
3. Реализуйте канал mpsc, по которому один поток отправляет данные, а другой их обрабатывает;
4. Создайте небезопасный блок unsafe, в котором вызывается простая C-функция (например, strlen) через FFI;
5. Напишите безопасную обёртку над этой функцией, чтобы скрыть unsafe логику;
6. Добавьте логирование с помощью tracing или log, чтобы отслеживать выполнение задач;
7. Закоммитьте в ветку feature/ffi-concurrency и создайте Merge Request с пояснением, как обеспечена безопасность.

Примерное описание задания и критерии оценивания к проекту

Цель проекта: разработать системное приложение на языке Rust, демонстрирующее глубокое понимание модели памяти, владения, многопоточности, асинхронного выполнения и взаимодействия с низкоуровневыми компонентами, а также применение современных практик проектирования и тестирования.

Задачи, которые необходимо выполнить студенту:

— создать проект с использованием Cargo и настроить рабочее пространство (workspace) для разделения бинарного и библиотечного кода;

— реализовать структуры и перечисления для представления бизнес-модели (например, User, Order, Status);

— применить концепции владения и заимствования для безопасной передачи данных между функциями без копирования;

— реализовать трейты для определения общего поведения (например, Processable, Serializable) и использовать обобщения для универсальных функций;

— организовать код по модулям (models, services, utils) и экспортировать публичные элементы;

— использовать умные указатели (Rc, RefCell, Arc, Mutex) для управления разделяемым состоянием и внутренней изменяемостью;

— реализовать многопоточное приложение с использованием std::thread или tokio, включающее передачу данных через каналы mpsc;

— написать асинхронную задачу, которая имитирует фоновую обработку (например, отправку уведомлений или логирование);

— интегрировать unsafe-блок для вызова функции из C-библиотеки (например, strlen или getpid) и создать над ней безопасную обёртку;

— написать модульные, интеграционные и документационные тесты, запустить их с помощью cargo test;

— добавить логирование с помощью tracing или log для отслеживания выполнения ключевых операций;

— оформить код в соответствии с идиоматикой Rust, использовать clippy и rustfmt для проверки качества;

— выложить проект в GitLab, организовать ветки и создать Merge Request с описанием архитектуры и решённых задач.

Критерии оценивания проекта:

— корректное применение механизма владения, заимствования и времени жизни ссылок для обеспечения безопасности памяти;

— использование умных указателей и примитивов синхронизации для построения потокобезопасных структур данных;

— реализация многопоточного и асинхронного кода с предотвращением состояний гонки и взаимных блокировок;

— корректная интеграция с C через FFI и создание безопасных обёрток над unsafe-функциями;

— качество кода и проектной структуры: модульность, тестирование, документация, соответствие стандартам Rust.

Примерные задания для контрольной работы

1. Объясните, как модель владения (ownership) в Rust предотвращает утечки памяти и двойное освобождение. Приведите пример передачи владения при передаче строки в функцию.
2. Напишите структуру Rectangle с полями width и height типа u32. Реализуйте метод area(), который возвращает площадь. Создайте экземпляр и выведите результат.
3. Опишите разницу между &T, &mut T, String и &str. Приведите пример, где использование заимствования (&) позволяет избежать перемещения владения.
4. Реализуйте перечисление Shape с вариантами Circle(f64) и Square(f64). Напишите функцию area(&self) -> f64, вычисляющую площадь для каждого варианта.

5. Объясните, зачем нужны трейты в Rust. Реализуйте трейт Loggable, который выводит сообщение в консоль. Реализуйте его для структуры User.

6. Создайте модуль utils с функцией is_even(n: i32) -> bool. Подключите модуль в main.rs и вызовите функцию. Укажите, как настроить структуру проекта с рабочим пространством.

7. Напишите пример использования Rc<RefCell<T>> для создания разделяемого изменяемого состояния между несколькими владельцами. Объясните, зачем нужны оба типа.

8. Опишите, как lifetimes помогают компилятору проверять валидность ссылок. Приведите пример функции с аннотацией времени жизни: fn longest<'a>(s1: &'a str, s2: &'a str) -> &'a str.

9. Напишите асинхронную функцию с использованием tokio, которая каждые 2 секунды выводит в консоль "Тик". Запустите её в main с помощью tokio::spawn.

10. Объясните, зачем используется unsafe в Rust. Напишите пример вызова C-функции strlen через FFI и создайте безопасную обёртку над ней.

Задания для промежуточной аттестации по дисциплине (модулю)

№ п/п	Задание	Ответ	Компетенция	Индикатор
1	Опишите роли участников команды при разработке проекта на Rust и распределении задач по модулям.	Разработчик пишет код, тестировщик проверяет, архитектор проектирует API, все несут ответственность за этап	УК-3, УК-1	УК-3.1, УК-1.2
2	Как мотивировать команду и распределить ресурсы при оптимизации производительности Rust-приложения с ограниченным временем?	Четкое распределение задач, контроль сроков, пример профессионализма, оценка трудоемкости этапов проекта	УК-3, РГ-1	УК-3.2, РГ-1.1
3	Подготовьте презентацию результатов оптимизации Rust-сервиса для стейкхолдеров и руководства компании.	Структура: цель, методы, результаты, выводы, рекомендации с учетом целевых установок заинтересованных сторон	УК-1, ППК-ДА3	УК-1.3, ППК-ДА3.2
4	Оформите документацию по Rust-проекту согласно профессиональным стандартам и требованиям к техническим спецификациям.	Титульный лист, содержание, введение, основная часть, выводы, список источников, приложения по ГОСТ	УК-1, ППК-Р5	УК-1.3, ППК-Р5.3

5	Дайте определение модели владения в Rust и запишите её смысл в контексте управления памятью.	Модель владения гарантирует один владелец на объект, предотвращает утечки памяти и двойное освобождение	УК-1, ОПК-6	УК-1.2, ОПК-6.1
6	Проанализируйте экономическую информацию для выбора между библиотеками для Rust-приложения.	Сравнение затрат, производительности, рисков, учет специфики задач, влияние на эффективность бизнес-решений	УК-1, ОПК-6	УК-1.2, ОПК-6.2
7	Обоснуйте экономическое решение о выборе архитектуры микросервисов на Rust на основе анализа затрат и выгод.	Решение выбирается если учитывает интересы всех сторон, повышает защиту инвестора от неблагоприятных изменений	УК-1, ОПК-6	УК-1.3, ОПК-6.3
8	Запишите формулу для расчета времени выполнения алгоритма и выберите метод анализа данных для оценки.	Формула временной сложности, метод статистического анализа для оценки значимости улучшений производительности	ОПК-6, ППК-ДА3	ОПК-6.2, ППК-ДА3.2
9	Оптимизируйте выбор структуры данных в Rust и обоснуйте влияние на бизнес показатели компании.	Выбор структуры с максимальной эффективностью, оценка воздействия на ключевые показатели компании, скорость обработки	ОПК-6, ППК-Р5	ОПК-6.3, ППК-Р5.3
10	Обоснуйте экономическое решение в учебном проекте с учетом системного мышления при проектировании Rust-архитектуры.	Использование модели с учетом взаимосвязей между модулями, данными и бизнес-целями компании	ОПК-6, РГ-4	ОПК-6.3, РГ-4.1
11	Назовите правовые нормы регулирующие использование лицензий при разработке Rust-ПО и их смысл.	Лицензии MIT, Apache, требования к открытому коду, ответственность за нарушение лицензионных соглашений	ППК-Р5, УК-1	ППК-Р5.3, УК-1.1
12	Примените правовые нормы при оценке рисков нарушения лицензий при работе с crates.io-пакетами для Rust.	Анализ рисков несоблюдения лицензионных требований, оценка	ППК-Р5, ППК-Р6	ППК-Р5.3, ППК-Р6.3

		вероятности и пороговых значений, меры минимизации штрафов		
13	Дайте правовую оценку ситуации с нарушением требований по лицензированию в Rust-проекте.	Определение нарушений лицензионных актов, применение санкций, влияние на репутацию организации	ППК-Р5, УК-1	ППК-Р5.3, УК-1.3
14	Выберите метод статистического анализа для проверки гипотез о влиянии оптимизации на производительность Rust-кода.	A/B тест, проверка гипотез о значимости различий, расчет доверительных интервалов для оценок времени выполнения	ППК-ДА3, ППК-ДА4	ППК-ДА3.2, ППК-ДА4.2
15	Адаптируйте аналитическую модель оценки Rust-модуля под бизнес потребности и оцените влияние решений.	Перевод технических метрик в бизнес показатели, оценка воздействия на ключевые показатели компании, оптимизация процессов	ППК-ДА3, ППК-Р6	ППК-ДА3.2, ППК-Р6.3
16	Выберите тип визуализации для представления данных о результатах тестирования и коммуникации с руководством.	График времени выполнения и доверительных интервалов, интерактивный дашборд для публичной дискуссии о результатах анализа	ППК-ДА3, УК-1	ППК-ДА3.2, УК-1.3
17	Выявите бизнес проблему на основе анализа отклонений в производительности Rust-кода и примите решение.	Проблема низкой скорости обработки, выбор стратегии реагирования, решение о корректировке архитектуры или алгоритма	ППК-ДА4, ППК-Р7	ППК-ДА4.2, ППК-Р7.2
18	Классифицируйте бизнес возможности при масштабировании Rust-решения с учетом стратегического планирования.	Оптимизация вычислительных ресурсов, разработка плана действий с учетом целей компании, выявление резервов роста	ППК-ДА4, РГ-4	ППК-ДА4.2, РГ-4.3

19	Сформулируйте рекомендации на основе статистических выводов о результатах тестирования для управления продуктом.	При значимом улучшении внедрение изменения, при отсутствии эффекта отказ от изменения для эффективности системы	ППК-Р6, ППК-Р7	ППК-Р6.3, ППК-Р7.3
20	Оцените риски реализации Rust-проекта и разработайте меры управления.	Анализ вероятности срывов сроков, мониторинг индикаторов успешности, пересмотр плана при выявлении проблем	РГ-1, УК-1	РГ-1.3, УК-1.3