

**Федеральное государственное автономное образовательное
учреждение высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»**

УТВЕРЖДЕНО
Директор

А.В. Малеев

	Рабочая программа дисциплины (модуля)
по дисциплине:	Программирование на Rust
по направлению:	Программная инженерия
профиль подготовки:	Разработка программно-информационных систем высшая школа программной инженерии высшая школа программной инженерии
курс:	3
квалификация:	бакалавр

Семестр, формы промежуточной аттестации: 5 (осенний) - Дифференцированный зачет

Аудиторных часов: 60 всего, в том числе:

лекции: 30 час.

семинары: 30 час.

лабораторные занятия: 0 час.

Самостоятельная работа: 48 час.

Всего часов: 108, всего зач. ед.: 3

Программу составил: А.А. Жмуров, канд. физ.-мат. наук, доцент

Программа обсуждена на заседании высшей школы программной инженерии 19.03.2025

Аннотация

Курс посвящен изучению языка программирования Rust, его особенностей и применения для разработки безопасных и эффективных программ. В ходе обучения студенты освоят основные концепции Rust, включая управление памятью, заимствование, обработку ошибок, асинхронное и многопоточное программирование. Также курс охватывает работу с библиотеками и фреймворками, такими как Tokio и Axum, использование Rust для сетевых приложений, встроенных систем и взаимодействие с Python. В конце курса студенты будут способны разрабатывать высококачественные и надежные приложения с использованием всех возможностей Rust.

1. Цели и задачи

Цель дисциплины

- освоение фундаментальных концепций и инструментов языка программирования Rust для эффективной и безопасной разработки программных решений, включая системное, многопоточное, асинхронное и встроенное программирование.

Задачи дисциплины

- изучить управление памятью, заимствование, обработку ошибок и функциональные возможности;
- освоить кодогенерацию, макросы, тестирование и работу с Cargo;
- разобраться с многопоточностью, асинхронностью и сетевым программированием;
- познакомиться с Rust в веб-разработке, встроенных системах и ядре Linux.

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
ПК-3 Способен проектировать, разрабатывать, интегрировать, проверять на работоспособность программное обеспечение	ПК-3.1 Различает синтаксис языков программирования, особенности программирования на этих языках, стандартные библиотеки языков программирования
	ПК-3.4 Знает, как определять оптимальные методы и средства проектирования программного обеспечения и структур данных
	ПК-3.3 Умеет излагать основные принципы построения и виды архитектуры программного обеспечения, методы и средства проектирования программного обеспечения, методологии разработки программного обеспечения и технологии программирования
ПК-5 Способен проектировать, разрабатывать, внедрять, сопровождать и снимать с эксплуатации информационные системы	ПК-5.1 Умеет описывать архитектуру, устройство и функционирование информационных систем

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- основы управления памятью, заимствования и времени жизни объектов;
- принципы многопоточного и асинхронного программирования;
- модель компиляции, работу Cargo и экосистему Rust.

уметь:

- разрабатывать безопасные и эффективные программы на Rust;
- Работать с макросами, обработкой ошибок и кодогенерацией;
- использовать Rust для сетевого, встроенного и системного программирования.

владеть:

- инструментами тестирования, профилирования и документации;
- подходами к оптимизации и написанию отказоустойчивого кода;
- практическими навыками работы с популярными библиотеками и фреймворками.

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

4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Зачем придуман Rust. Пишем Game of Life на Rust	1	1		3
2	Владение объектами, время жизни и Borrow Checker	1	1		3
3	Слайсы. Enums & pattern matching. Traits & dynamic dispatch	2	2		3
4	Generics & static dispatch. Умные указатели	2	2		3
5	Обработка ошибок	2	2		3
6	Функциональные аспекты языка. Итераторы и замыкания	2	2		3
7	Кодогенерация и макросы. Разбор крейтов на макросах: clap, serde	2	2		3
8	Модули, библиотеки и модель компиляции. Тестирование. Cargo doc	2	2		3
9	Многопоточное программирование	2	2		3
10	Сетевое программирование. Логирование и метрики	2	2		3
11	Асинхронный Rust. Разбор асинхронного фреймворка rio	2	2		3
12	Tokio и его экосистема	2	2		3
13	Разбор крейта Axum	2	2		3
14	Встроенный Rust. Chip8 на микроконтроллере	2	2		3
15	Rust и Python	2	2		3
16	Rust в ядре Linux	2	2		3
Итого часов		30	30		48
Подготовка к экзамену		0 час.			
Общая трудоёмкость		108 час., 3 зач.ед.			

4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 5 (Осенний)

1. Зачем придуман Rust. Пишем Game of Life на Rust

Введение в язык, его цели и преимущества. Знакомство с синтаксисом и основными концепциями.

2. Владение объектами, время жизни и Borrow Checker

Механизм владения, заимствования и времени жизни. Работа Borrow Checker для обеспечения безопасности памяти.

3. Слайсы. Enums & pattern matching. Traits & dynamic dispatch

Использование слайсов, перечислений и сопоставления с образцом. Трейты, динамическая диспетчеризация и полиморфизм в Rust.

4. Generics & static dispatch. Умные указатели

Дженерики и статическая диспетчеризация. Работа с умными указателями (Box, Rc, Arc, RefCell).

5. Обработка ошибок

Механизмы обработки ошибок: Result, Option, ?-оператор, паника (panic!).

6. Функциональные аспекты языка. Итераторы и замыкания

Функциональные конструкции: замыкания, итераторы, метод map, filter и fold.

7. Кодогенерация и макросы. Разбор крейтов на макросах: clap, serde

Создание макросов, кодогенерация, работа с популярными библиотеками clap (парсинг аргументов) и serde (сериализация).

8. Модули, библиотеки и модель компиляции. Тестирование. Cargo doc

Организация кода в модули и библиотеки, механизм сборки и тестирования, генерация документации с Cargo doc.

9. Многопоточное программирование

Потоки, std::thread, Mutex, RwLock, Atomic и безопасная работа с конкурентными данными.

10. Сетевое программирование. Логирование и метрики

Работа с TCP/UDP, HTTP, WebSockets, логирование и мониторинг приложения.

11. Асинхронный Rust. Разбор асинхронного фреймворка gio

Основы async/await, работа с Future, async-std, разбор gio как примера асинхронного фреймворка.

12. Tokio и его экосистема

Основные компоненты Tokio: Runtime, Task, Channel, интеграция с Hyper и Tonic.

13. Разбор крейта Axum

Практическое использование Axum для создания веб-сервисов, маршрутизация, middleware, работа с JSON.

14. Встроенный Rust. Chip8 на микроконтроллере

Разработка на Rust для встраиваемых систем, работа с no_std, реализация эмулятора Chip8.

15. Rust и Python

Взаимодействие Rust и Python через PyO3, создание расширений и ускорение вычислений.

16. Rust в ядре Linux

Использование Rust для разработки драйверов и модулей ядра Linux, особенности низкоуровневого программирования.

5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

компьютер, микроконтроллеры, компилятор Rust, Cargo, IDE, серверные ресурсы.

6. Перечень рекомендуемой литературы

Основная литература

Фонд библиотеки МФТИ:

1. Кузнецов, А.С. Системное программирование : учеб. пособие / А.С. Кузнецов, И.А. Якимов, П.В. Пересунько. - Красноярск : Сиб. федер. ун-т 2018. - 170с. - ISBN 978-5-7638-3885-5. - Текст : электронный. - URL: <https://znanium.com/catalog/product/1032183>
2. Гунько, А. В. Системное программирование в среде Linux : учебное пособие / А. В. Гунько. - Новосибирск : Изд-во НГТУ, 2020. - 235 с. - ISBN 978-5-7782-4160-2. - Текст : электронный. - URL: <https://znanium.com/catalog/product/1870577>

Дополнительная литература

7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

Не требуется

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)

Rust, Cargo, IDE (VS Code, CLion), Git, библиотеки для сетевого и асинхронного программирования.

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Успешное усвоение курса требует:

- регулярной практики в программировании на Rust;
- понимания принципов управления памятью и заимствования;
- способности работать с основными библиотеками и фреймворками;
- умения анализировать ошибки и исправлять их;
- применения тестирования для обеспечения качества кода;
- активного использования документации и дополнительных ресурсов.

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению: Программная инженерия
профиль подготовки: Разработка программно-информационных систем
высшая школа программной инженерии
высшая школа программной инженерии
курс: 3
квалификация: бакалавр

Семестр, формы промежуточной аттестации: 5 (осенний) - Дифференцированный зачет

Разработчик: А.А. Жмуров, канд. физ.-мат. наук, доцент

1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
ПК-3 Способен проектировать, разрабатывать, интегрировать, проверять на работоспособность программное обеспечение	ПК-3.1 Различает синтаксис языков программирования, особенности программирования на этих языках, стандартные библиотеки языков программирования
	ПК-3.4 Знает, как определять оптимальные методы и средства проектирования программного обеспечения и структур данных
	ПК-3.3 Умеет излагать основные принципы построения и виды архитектуры программного обеспечения, методы и средства проектирования программного обеспечения, методологии разработки программного обеспечения и технологии программирования
ПК-5 Способен проектировать, разрабатывать, внедрять, сопровождать и снимать с эксплуатации информационные системы	ПК-5.1 Умеет описывать архитектуру, устройство и функционирование информационных систем

2. Показатели оценивания компетенций

В результате изучения дисциплины «Программирование на Rust» обучающийся должен:

знать:

- основы управления памятью, заимствования и времени жизни объектов;
- принципы многопоточного и асинхронного программирования;
- модель компиляции, работу Cargo и экосистему Rust.

уметь:

- разрабатывать безопасные и эффективные программы на Rust;
- Работать с макросами, обработкой ошибок и кодогенерацией;
- использовать Rust для сетевого, встроенного и системного программирования.

владеть:

- инструментами тестирования, профилирования и документации;
- подходами к оптимизации и написанию отказоустойчивого кода;
- практическими навыками работы с популярными библиотеками и фреймворками.

3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

С целью контроля освоения обучающимися учебного материала проводится выполнение практических заданий и тестирование знание по ключевым темам курса.

4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

1. Объясните основные принципы управления памятью в Rust (владение, заимствование, время жизни).
2. Разница между статической и динамической диспетчеризацией в Rust.
3. Как работает Borrow Checker и зачем он нужен?
4. Преимущества использования перечислений (Enums) и паттерн-матчинга.
5. Как в Rust реализуются умные указатели и какие они бывают?
6. Объясните механизм обработки ошибок в Rust с использованием Result и Option.
7. Как работают итераторы и замыкания в Rust?
8. Принципы работы с асинхронным программированием в Rust.
9. Что такое многопоточность в Rust и как обеспечивается безопасность данных при параллельной обработке?

10. Основные особенности библиотеки Tokio и её использование в асинхронном программировании.
11. Как использовать Rust для разработки сетевых приложений?
12. Основы работы с микроконтроллерами и встроенными системами на Rust.
13. Взаимодействие Rust и Python через библиотеку PyO3.
14. Применение Rust в разработке драйверов и системного ПО (ядро Linux).

Критерии оценивания

- оценка «отлично (10)» выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений
- оценка «отлично (9)» выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений
- оценка «отлично (8)» выставляется студенту, показавшему всесторонние систематизированные, глубокие знания учебной программы дисциплины и умение применять их на практике при решении конкретных задач, и правильное обоснование принятых решений
- оценка «хорошо (7)» выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;
- оценка «хорошо (6)» выставляется студенту, если он знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;
- оценка «хорошо (5)» выставляется студенту, если он знает материал, и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;
- оценка «удовлетворительно (4)» выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он владеет основными разделами учебной программы, необходимыми для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;
- оценка «удовлетворительно (3)» выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он владеет фрагментарно основными разделами учебной программы, необходимыми для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;
- оценка «неудовлетворительно (2)» выставляется студенту, который не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных понятий дисциплины и не умеет использовать полученные знания при решении типовых практических задач
- оценка «неудовлетворительно (1)» выставляется студенту, который не знает формулировок основных понятий дисциплины

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Дифференцированный зачет проводится по итогам текущей успеваемости и сдачи заданий и других видов работ, предусмотренных программой дисциплины и (или) путем организации специального опроса, проводимого в устной и (или) письменной форме.