ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

Приказом Ректора АНО ВО
«Центральный университет»
Ивашкевич Е.В.
от «19» января 2024 г. № 0119.37

Рабочая программа дисциплины (модуля) «Многопоточное программирование» дополнительной профессиональной программы – программы профессиональной переподготовки «Академия data science»

Траектория: Продуктовая аналитика

Москва
2024

Электронный документ

Содержание

1. Краткая характеристика дисциплины (модуля)

Многопоточное программирование является ключевым навыком в современной разработке, поскольку позволяет эффективно использовать многоядерные процессоры и ресурсы системы. Это способствует созданию более быстрых, масштабируемых и отзывчивых приложений, что особенно важно для серверных систем, игр, обработки больших данных и пользовательских интерфейсов. Понимание принципов синхронизации и управления потоками помогает предотвращать ошибки, такие как гонки данных и взаимные блокировки, обеспечивая надежность и стабильность программ.

Цель изучения дисциплины (модуля): формирование у слушателей знаний методов и средств разработки программного обеспечения, способного эффективно выполнять несколько потоков выполнения для повышения производительности и отзывчивости приложений.

Задачи изучения дисциплины (модуля):

— формирование знаний о распространенных проблемах гонки данных;

— формирование знаний устройства примитивов синхронизации;

— формирование умения применять механизмы синхронизации;

— формирование умения реализовывать структуры данных и алгоритмы, используемые в многопоточном программировании;

— формирование умения реализовывать надежные многопоточные программы.

2. Тематический план

№ п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Аудиторная работа		Контроль	Самостоятельная работа	
		Лекции	Семинары (практические занятия)			
1	Основы многопоточности	6	6		18	Домашнее задание Подготовка к семинару
2	Синхронизация и управление потоками	6	7		18	Домашнее задание Подготовка к семинару
3	Продвинутые структуры данных и алгоритмы	7	7		18	Домашнее задание Подготовка к семинару
4	Асинхронные и реактивные системы	7	7		20	Домашнее задание Подготовка к семинару
5	Оптимизация и практика	7	7		20	Домашнее задание Проект
	Зачет			4		
	Итого:	***33***	***34***	***4***	***94***	
Объем дисциплины (модуля) (в ак. ч.)		***165***				

3. Содержание дисциплины (модуля)

№п/п	Наименование раздела дисциплины (модуля)	Содержание дисциплины (модуля) по темам
1	Основы многопоточности	Введение, закон Мура, race condition, модель памяти. Класс Thread, Runnable, жизненный цикл потока. Проблемы видимости (volatile), атомарность (Atomic-классы).
2	Синхронизация и управление потоками	synchronized, мониторы, deadlock. Пулы потоков (ExecutorService), Future. CompletableFuture. Интерфейс Lock, ReadWriteLock, Condition
3	Продвинутые структуры данных и алгоритмы	ConcurrentHashMap, BlockingQueue, неблокирующие структуры. Синхронизаторы (CountDownLatch, CyclicBarrier, Semaphore). ForkJoinPool, параллельные стримы
4	Асинхронные и реактивные системы	Реактивное программирование (Reactor/RxJava), очереди событий. Виртуальные потоки. Тестирование многопоточного кода (нагрузочные тесты, детектирование гонок)
5	Оптимизация и практика	Мониторинг (VisualVM, JFR), профилирование, оптимизация блокировок

4. Учебно-методическое обеспечение

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

Каждый слушатель 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.

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

Основная литература:

1. Зыков, С. В. Программирование : учебник и практикум для вузов / С. В. Зыков. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 285 с. — (Высшее образование). — ISBN 978-5-534-16031-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/560815.

2. Компьютерные сети : учебник и практикум для вузов / под научной редакцией А. М. Нечаева, А. Е. Трубина, А. Ю. Анисимова. — Москва : Издательство Юрайт, 2025. — 515 с. — (Высшее образование). — ISBN 978-5-534-21452-9. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/572239.

Дополнительная литература:

1. Гниденко, И. Г. Технологии и методы программирования : учебник для вузов / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 241 с. — (Высшее образование). — ISBN 978-5-534-18130-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/581329.

5. 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 к 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

6. Методические и оценочные материалы

Методические указания для обучающихся по освоению дисциплины (модуля)

В процессе изучения дисциплины (модуля) «Многопоточное программирование» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, проект, домашние задания, 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.

Участие в семинаре (аудиторная работа) – активная работа слушателя на семинаре, его ответы на вопросы преподавателя и участие в дискуссии.

Для успешного участия в семинаре слушателям рекомендуется заранее ознакомиться с темой обсуждения, прочитать необходимые материалы и подготовить вопросы. Важно активно слушать и вовлекаться в дискуссию, высказывая свои мнения и 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.

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

Для успешной подготовки к проекту: четко определите цели и задачи проекта, распределите роли и обязанности между участниками, а также установите сроки

выполнения каждой части работы. Регулярно проводите встречи для обсуждения прогресса и решения возникающих вопросов.

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

В процессе самостоятельной работы слушатели взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи слушателя 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)

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***.

Для оценивания текущего контроля успеваемости и промежуточной аттестации используется десятибалльная шкала оценивания, которая соотносится с традиционной пятибалльной шкалой следующим образом:

Десятибалльная оценка	**Пятибалльная оценка**	**Оценка за зачет**	**Общая характеристика результата обучения по дисциплине (модулю)**
10	Отлично	Зачтено	Слушатель полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину (модуль). 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. Слушатель 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	Слушатель 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
6	Хорошо	Зачтено	

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
			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. Слушатель хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
5	Удовлетворительно	Зачтено	Слушатель 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. Слушатель способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования.
4	Удовлетворительно	Зачтено	
3	Не сдан	Не зачтено	Слушатель 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) «Многопоточное программирование» оценивается следующим образом:

Активность	Вес	Описание
Домашние задания	50%	Набор заданий по темам. За каждое из заданий можно набрать 10 баллов
Проект	20%	Исследовательская работа по дисциплине (модулю) и презентация результатов
Зачет	30%	Письменная или устная работа над заданием, направленным на проверку полученных знаний и навыков по дисциплине (модулю)

Формула расчёта итоговой оценки по дисциплине (модулю) «Многопоточное программирование»: «0,5 × среднее за домашние задания + 0,2 × проект + 0,3 × зачет».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные темы семинарских занятий

Основы многопоточности

1. **Введение в многопоточность:** Преимущества и недостатки многопоточного программирования.
2. **Закон Мура и его влияние на производительность:** Как закон Мура влияет на многопоточность и параллелизм.
3. **Race Condition:** Определение, примеры и способы предотвращения гонки данных.
4. **Модель памяти в Java:** Как работает модель памяти и её влияние на многопоточность.
5. **Классы Thread и Runnable:** Жизненный цикл потока и их использование в многопоточном программировании.

Синхронизация и управление потоками

1. **Ключевое слово synchronized и мониторы:** Как работают синхронизация и блокировка потоков.
2. **Deadlock:** Определение, примеры и методы предотвращения взаимной блокировки.
3. **Пулы потоков и ExecutorService:** Как использовать ExecutorService для управления потоками.
4. **Future и CompletableFuture:** Асинхронное программирование с использованием Future и CompletableFuture.
5. **Интерфейсы Lock и ReadWriteLock:** Различия, применение и примеры использования.

Продвинутые структуры данных и алгоритмы

1. **ConcurrentHashMap:** Преимущества и применение в многопоточных приложениях.
2. **BlockingQueue:** Использование BlockingQueue для синхронизации потоков и обработки данных.
3. **Неблокирующие структуры данных:** Примеры и преимущества неблокирующих структур.
4. **Синхронизаторы (CountDownLatch, CyclicBarrier, Semaphore):** Как и когда использовать эти синхронизаторы.
5. **ForkJoinPool и параллельные стримы:** Применение ForkJoinPool и использование параллельных стримов для обработки данных.

Асинхронные и реактивные системы

1. **Реактивное программирование:** Основы реактивного программирования и его применение (Reactor/RxJava).
2. **Очереди событий:** Как работают очереди событий и их роль в асинхронных системах.
3. **Виртуальные потоки:** Преимущества и использование виртуальных потоков в Java.
4. **Тестирование многопоточного кода:** Методы нагрузочного тестирования и детектирования гонок.
5. **Сравнение асинхронного и реактивного программирования:** Преимущества и недостатки каждого подхода.

Примерные домашние задания

Домашнее задание: Основы многопоточности

1. Объясните закон Мура и его влияние на развитие многопоточных приложений.
2. Что такое race condition? Приведите пример ситуации, когда она может возникнуть.
3. Опишите жизненный цикл потока в Java с использованием классов Thread и Runnable.
4. Что такое проблема видимости в многопоточности? Как ключевое слово `volatile` помогает её решить?
5. Чем отличаются атомарные операции от неатомарных? Приведите пример использования Atomic-классов в Java.

Домашнее задание: Синхронизация и управление потоками

1. Объясните принцип работы ключевого слова `synchronized` и роль мониторов в Java.
2. Что такое deadlock? Приведите пример кода, который может привести к взаимной блокировке.
3. Опишите назначение и использование пула потоков `ExecutorService`. Как работает интерфейс `Future`?
4. Что такое `CompletableFuture` и как он расширяет возможности работы с асинхронными задачами?
5. В чем разница между интерфейсами `Lock` и `ReadWriteLock`? Как используется `Condition`?

Домашнее задание: Продвинутые структуры данных и алгоритмы

1. Опишите структуру данных `ConcurrentHashMap` и её преимущества по сравнению с обычным `HashMap` в многопоточном окружении.
2. Что такое `BlockingQueue`? Приведите пример её использования для организации взаимодействия потоков.
3. Объясните назначение синхронизаторов `CountDownLatch`, `CyclicBarrier` и `Semaphore`. В каких ситуациях они применяются?
4. Что такое `ForkJoinPool`? Как он помогает в реализации параллельных вычислений?
5. Как работают параллельные стримы в Java? Приведите пример их использования для обработки коллекции.

Примерное задание для проекта

Проект по теме: Мониторинг, профилирование и оптимизация блокировок в многопоточном приложении

Цель проекта:

Научиться использовать инструменты мониторинга (VisualVM, Java Flight Recorder) для анализа производительности многопоточного приложения, выявлять узкие места, связанные с блокировками, и оптимизировать код для повышения эффективности.

Задание:

1. **Выбрать или разработать многопоточное Java-приложение**, в котором присутствуют блокировки (например, с использованием synchronized, Lock, BlockingQueue и т.п.). Можно взять готовый пример с гонками или с избыточной синхронизацией.
2. **Собрать исходные показатели производительности** приложения:

- Время выполнения ключевых операций.
- Использование CPU.
- Количество и длительность блокировок.

3. **Провести мониторинг и профилирование** с помощью:
   - VisualVM (сбор данных о потоках, блокировках, памяти).
   - Java Flight Recorder (JFR) — анализ событий и блокировок.
4. **Проанализировать полученные данные**, выявить узкие места, связанные с блокировками и синхронизацией.
5. **Внести изменения в код для оптимизации блокировок**, например:
   - Снизить область синхронизации.
   - Заменить synchronized на более эффективные примитивы (Lock, ReadWriteLock).
   - Использовать неблокирующие структуры данных.
6. **Повторно провести мониторинг и профилирование** после оптимизации.
7. **Сравнить результаты до и после оптимизации**, сделать выводы о достигнутом улучшении.
8. **Подготовить отчет**, включающий:
   - Описание приложения и его многопоточной части.
   - Методики мониторинга и инструменты.
   - Анализ исходных данных.
   - Описание внесенных изменений.
   - Сравнительный анализ результатов.
   - Выводы и рекомендации.

Этапы выполнения:

Этап	Срок	Описание
1.Выбор/создание приложения	1 неделя	Подготовка исходного многопоточного кода
2. Сбор исходных данных	1 неделя	Мониторинг и профилирование до оптимизации
3. Анализ и оптимизация	1-2 недели	Поиск узких мест и внесение изменений
4. Повторный мониторинг	1 неделя	Сбор данных после оптимизации
5. Подготовка отчета	1 неделя	Оформление результатов и выводов

Критерии оценивания:

Критерий	Максимальный балл	Описание
Корректность и полнота мониторинга	25	Использование VisualVM и JFR, полнота собранных данных
Анализ и выявление узких мест	20	Глубина и качество анализа проблем с блокировками
Качество и эффективность оптимизации	25	Внесённые изменения, их влияние на производительность
Сравнительный анализ результатов	15	Чёткое сравнение показателей до и после оптимизации

Критерий	Максимальный балл	Описание
Оформление и полнота отчёта	15	Структура, ясность изложения, наличие всех разделов

Форма сдачи проекта:

- Исходный код проекта (до и после оптимизации).
- Файлы с профилями/снимками из VisualVM и JFR.
- Отчет в формате PDF (10-15 страниц), включающий все пункты из задания.
- Краткая презентация (5-7 слайдов) с основными результатами и выводами.

Примерные вопросы для подготовки к зачету

1. Что такое многопоточность и какие её преимущества?
2. Как закон Мура влияет на разработку многопоточных приложений?
3. Объясните, что такое race condition и приведите пример.
4. Какие проблемы могут возникнуть из-за гонки данных?
5. Как работает модель памяти в Java и как она влияет на многопоточность?
6. В чем разница между классами Thread и Runnable?
7. Опишите жизненный цикл потока в Java.
8. Что такое видимость в многопоточности и как она может быть нарушена?
9. Как ключевое слово volatile помогает решить проблемы видимости?
10. Что такое атомарные операции и как используются Atomic-классы в Java?
11. Как работает ключевое слово synchronized в Java?
12. Что такое монитор и как он используется для синхронизации потоков?
13. Объясните, что такое deadlock и как его можно избежать.
14. Как работает пул потоков (ExecutorService) и в чем его преимущества?
15. Что такое интерфейс Future и как он используется в асинхронном программировании?
16. Объясните, что такое CompletableFuture и как его можно использовать для обработки асинхронных задач.
17. В чем разница между интерфейсами Lock и ReadWriteLock?
18. Как работает интерфейс Condition и когда его следует использовать?
19. Как работает ConcurrentHashMap и в чем его преимущества по сравнению с HashMap?
20. Что такое BlockingQueue и как она используется для синхронизации потоков?
21. Объясните, что такое неблокирующие структуры данных и их преимущества.
22. Как работают синхронизаторы CountDownLatch, CyclicBarrier и Semaphore?
23. Что такое ForkJoinPool и как он используется для параллельных вычислений?
24. Как работают параллельные стримы в Java и в чем их преимущества?
25. Что такое реактивное программирование и как оно реализовано в Reactor/RxJava?
26. Как работают очереди событий и какую роль они играют в асинхронных системах?
27. Что такое виртуальные потоки и как они отличаются от обычных потоков?
28. Как тестировать многопоточный код на наличие гонок и других проблем?
29. Какие методы используются для нагрузочного тестирования многопоточных приложений?
30. Какие инструменты можно использовать для мониторинга и профилирования многопоточных приложений (например, VisualVM, JFR) и как они помогают в оптимизации блокировок?