МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»
(РУТ (МИИТ)

Рабочая программа дисциплины (модуля),
как компонент образовательной программы
базового высшего образования
по направлению подготовки
09.03.02 Информационные системы и технологии,
утвержденной первым проректором РУТ (МИИТ)
Тимониным В.С.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Программирование

Направление подготовки: 09.03.02 Информационные системы и технологии

Направленность (профиль): Технологии искусственного интеллекта в транспортных системах

Форма обучения: Очная

1. Общие сведения о дисциплине (модуле).

Дисциплина «Программирование» формирует фундамент инженерной подготовки в области разработки программного обеспечения на языке Java. Курс выстроен по принципу сквозного практического кейса, где студенты последовательно создают ядро информационной системы диспетчеризации грузоперевозок. Обучающиеся осваивают полный цикл создания бэкенд-модуля – от базовой алгоритмизации и объектно-ориентированного проектирования до интеграции с внешними REST-сервисами и реляционными базами данных. Особое внимание уделяется работе в условиях импортозамещения, использованию отечественных дистрибутивов JDK и операционных систем, а также современным парадигмам, включая функциональное программирование и конвейерную обработку данных. Выпускник получает связное портфолио работ и навыки, востребованные ведущими ИТ-компаниями и корпоративным сектором.

Целью освоения дисциплины является формирование у обучающихся системных теоретических знаний и практических инженерных навыков проектирования, реализации и тестирования надежных программных модулей на языке Java с применением современных парадигм программирования и отечественного технологического стека.

Для достижения поставленной цели в рамках дисциплины решается комплекс задач, направленных на формирование у обучающихся способности: анализировать предметную область и проектировать архитектуру приложений на основе принципов объектно-ориентированного программирования, реализовывать эффективные алгоритмы и структуры данных, обеспечивать отказоустойчивость кода через обработку исключений и модульное тестирование, а также интегрировать разрабатываемые решения с внешними информационными системами и базами данных.

2. Планируемые результаты обучения по дисциплине (модулю).

Перечень формируемых результатов освоения образовательной программы (компетенций) в результате обучения по дисциплине (модулю):

ОПК-3 - Способен использовать современные информационные технологии и программно-аппаратные средства, в том числе отечественного производства, при решении задач профессиональной деятельности;

ОПК-4 - Способен решать задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и технологий

искусственного интеллекта, а также с учетом основных требований информационной безопасности;

ОПК-6 - Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения.

Обучение по дисциплине (модулю) предполагает, что по его результатам обучающийся будет:

Знать:

- понятие программирования как инженерной дисциплины, эволюцию языков и парадигм, место Java в современном технологическом ландшафте;
- архитектуру платформы Java, систему типов данных и операторы управления потоком выполнения;
- принципы объектно-ориентированного программирования и механизмы их реализации в Java;
- концепцию перечислений, их внутреннюю реализацию, методы и специализированные коллекции;
- специальные модификаторы Java, зарезервированные слова и их отличие от финальных конструкций;
- отношения между классами в ООП, их семантику и нотацию;
- принципы проектирования классов и интерфейсов для обеспечения расширяемости кода;
- архитектуру потоков ввода-вывода и механизмы чтения и записи локальных файлов;
- систему числовых типов Java, классы-обёртки и классы для точных вычислений;
- иерархию коллекций Java, внутреннее устройство реализаций и алгоритмическую сложность;
- интерфейсы естественного и настраиваемого сравнения объектов и контракты согласованности;
- прикладное применение линейных и нелинейных структур данных для решения алгоритмических задач;
- иерархию исключительных ситуаций в Java и стратегии восстановления состояния программы;
- концепцию сериализации и десериализации объектов, встроенные механизмы и современные альтернативы;
- принципы модульного тестирования и архитектуру современных фреймворков тестирования;
- современный программный интерфейс работы с датами и временем;

- форматы обмена структурированными данными и алгоритмы их парсинга;

- инструментарий отладки кода и профилирования в интегрированных средах разработки;

- парадигму функционального программирования в Java, конвейерную обработку данных и контейнеры для nullable-значений;

- архитектуру клиент-серверного взаимодействия, современные HTTP-клиенты и принципы работы с реляционными СУБД.

Уметь:

- разрабатывать алгоритмы расчета логистических параметров при помощи базовых конструкций Java в условиях консольного приложения;

- проектировать иерархию классов для предметной области при помощи принципов ООП с соблюдением принципов SOLID;

- моделировать типобезопасные справочники и состояния сущностей при помощи перечислений в условиях строгой типизации;

- проектировать объектные графы предметной области при помощи паттернов отношений между классами с учетом требований к связности компонентов;

- выполнять точные финансовые расчеты при помощи классов произвольной точности с учетом требований к отсутствию ошибок округления;

- парсить структурированные данные из текстовых файлов при помощи стандартных средств Java с обработкой исключений ввода-вывода;

- документировать архитектуру классов и методов при помощи специализированных тегов и технических отчетов по стандартам ГОСТ;

- реализовывать пользовательские структуры данных при помощи современных интерфейсов коллекций Java с учетом ограничений по производительности;

- реализовывать естественный и настраиваемый порядок сортировки объектов при помощи интерфейсов сравнения с соблюдением контрактов транзитивности;

- парсить и валидировать структурированные манифесты при помощи специализированных библиотек с многоуровневой обработкой исключений;

- разрабатывать модульные тесты при помощи современных фреймворков с использованием параметризованных сценариев и соблюдением паттерна AAA;

- работать с датами и временем при помощи современного API с учетом часовых поясов и летнего времени;

- отлаживать и профилировать код при помощи встроенных средств среды разработки с анализом утечек памяти;

- анализировать большие массивы логов при помощи конвейерной обработки данных и функционального программирования с использованием параллельных потоков;

- интегрировать внешние REST-сервисы при помощи современного HTTP-клиента с обработкой сетевых исключений;

- реализовывать персистентное хранение данных при помощи JDBC и реляционных СУБД с соблюдением требований к безопасности запросов;

- формировать аналитические отчеты при помощи конвейерной обработки и файловых потоков в стандартизированных текстовых форматах.

Владеть:

- навыками настройки среды разработки и компиляции консольных приложений в различных операционных системах;

- навыками применения рекурсивных алгоритмов и структур данных для решения задач маршрутизации и диспетчеризации;

- навыками рефакторинга кода и применения регулярных выражений для валидации входных данных;

- навыками профилирования приложений и эмпирического обоснования выбора структур данных;

- навыками интеграции сторонних библиотек для сериализации и десериализации сложных объектов;

- навыками написания параметризованных модульных тестов и мокирования внешних зависимостей;

- навыками настройки подключений к базам данных и выполнения параметризованных SQL-запросов.

3. Объем дисциплины (модуля).

3.1. Общая трудоемкость дисциплины (модуля).

Общая трудоемкость дисциплины (модуля) составляет 9 з.е. (324 академических часа(ов).

3.2. Объем дисциплины (модуля) в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении учебных занятий:

Тип учебных занятий	Количество часов	
	Всего	Семестр

		№1	№2
Контактная работа при проведении учебных занятий (всего):	176	80	96
В том числе:			
Занятия лекционного типа	64	32	32
Занятия семинарского типа	112	48	64

3.3. Объем дисциплины (модуля) в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации составляет 148 академических часа (ов).

3.4. При обучении по индивидуальному учебному плану, в том числе при ускоренном обучении, объем дисциплины (модуля) может быть реализован полностью в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации.

4. Содержание дисциплины (модуля).

4.1. Занятия лекционного типа.

№ п/п	Тематика лекционных занятий / краткое содержание
1	Введение в программирование и экосистему Java Рассматриваемые вопросы: - понятие программирования как инженерной дисциплины и эволюция языков от ассемблера до современных высокоуровневых языков; - основные парадигмы программирования: императивная, процедурная, объектно-ориентированная, функциональная и декларативная; - современный ландшафт языков и области их применения; - архитектура платформы Java: JDK, JRE, JVM и их взаимодействие; - примитивные типы данных, структура простейшей программы и процесс компиляции в байт-код.
2	Операторы управления потоком выполнения Рассматриваемые вопросы: - условные операторы: if-else, switch и современные выражения switch; - циклы: for, while, do-while и их применение в алгоритмах; - цикл enhanced for-each для итерации по массивам и коллекциям; - операторы break, continue и вложенные циклы в алгоритмических задачах.
3	Массивы и алгоритмы их обработки Рассматриваемые вопросы: - объявление, инициализация и индексация одномерных массивов; - многомерные массивы и матрицы в Java; - базовые алгоритмы: поиск, сортировка, агрегация данных; - класс Arrays и его вспомогательные методы.

№ п/п	Тематика лекционных занятий / краткое содержание
4	Методы и рекурсия Рассматриваемые вопросы: - объявление методов, параметры и возвращаемые значения; - передача параметров по значению и особенности работы со ссылками; - перегрузка методов и сигнатуры; - рекурсивные алгоритмы и анализ глубины рекурсии.
5	Основы объектно-ориентированного программирования Рассматриваемые вопросы: - концепция классов и объектов как фундамент ООП; - конструкторы: конструктор по умолчанию и параметризованные; - ключевое слово this и область видимости полей; - жизненный цикл объектов и сборка мусора.
6	Инкапсуляция и специальные модификаторы Рассматриваемые вопросы: - принцип инкапсуляции и сокрытие реализации, геттеры и сеттеры; - модификаторы доступа: private, protected, public и доступ по умолчанию (default); - модификатор static: статические поля, методы, блоки инициализации и вложенные классы; - модификатор final для полей, методов, классов и локальных переменных, зарезервированное слово const.
7	Наследование и иерархии классов Рассматриваемые вопросы: - механизм наследования и ключевое слово extends; - класс Object как корень иерархии всех классов; - переопределение методов и аннотация @Override; - вызов конструкторов суперкласса через super.
8	Полиморфизм, абстрактные классы и интерфейсы Рассматриваемые вопросы: - статический и динамический полиморфизм, восходящее и нисходящее приведение типов; - оператор instanceof и современный паттерн-матчинг; - абстрактные классы и методы как контракт поведения; - интерфейсы, default-методы и множественная реализация.
9	Перечисления (Enum) в Java Рассматриваемые вопросы: - концепция перечислений как типобезопасной альтернативы константам; - внутренняя реализация Enum как наследника класса java.lang.Enum; - методы values(), valueOf(), ordinal() и собственные поля и методы в enum; - применение Enum в switch-выражениях и специализированные коллекции EnumSet и EnumMap.
10	Отношения между классами в ООП Рассматриваемые вопросы: - ассоциация как базовое отношение взаимодействия между объектами; - агрегация и композиция как формы отношений «часть-целое» с разной степенью владения; - зависимость (dependency) и её отличие от ассоциации; - делегирование как альтернатива наследованию и принцип предпочтения композиции.
11	Принципы SOLID и проектирование классов Рассматриваемые вопросы: - принцип единственной ответственности (SRP); - принцип открытости/закрытости (OCP); - принцип подстановки Лисков (LSP); - принцип разделения интерфейсов (ISP) и инверсии зависимостей (DIP).

№ п/п	Тематика лекционных занятий / краткое содержание
12	Строки и регулярные выражения Рассматриваемые вопросы: - класс String и его неизменяемость, пул строк и интернирование; - StringBuilder и StringBuffer для модифицируемых строк; - пакет java.util.regex и классы Pattern, Matcher; - практическое применение регулярных выражений для валидации данных.
13	Числовые типы: обёртки, BigInteger и BigDecimal Рассматриваемые вопросы: - классы-обёртки примитивных типов и автобоксинг; - ограничения примитивных типов с плавающей точкой и проблемы точности; - класс BigInteger для целочисленных вычислений произвольной точности; - класс BigDecimal для финансовых расчётов, режимы округления и масштабирование.
14	Обобщения (Generics) Рассматриваемые вопросы: - концепция обобщений и параметризация типов; - обобщённые классы, методы и интерфейсы; - ограничения типов (bounded wildcards) и принцип PECS; - стирание типов (type erasure) и его последствия.
15	Исключения в Java: иерархия, обработка, пользовательские классы Рассматриваемые вопросы: - класс Throwable и его наследники: Error и Exception, checked и unchecked исключения; - конструкция try-catch-finally, множественный catch и try-with-resources; - создание пользовательских классов исключений и best practices; - логирование исключений и стратегии восстановления состояния программы.
16	Архитектура потоков ввода-вывода и работа с файловой системой Рассматриваемые вопросы: - иерархия классов пакета java.io: байтовые и символьные потоки, декораторы; - пакет java.nio.file: Path, Files и современные подходы к работе с файлами; - чтение и запись текстовых файлов: BufferedReader, PrintWriter; - обработка исключений при файловых операциях.
17	Коллекции Java: интерфейс List Рассматриваемые вопросы: - архитектура Collection Framework и корневые интерфейсы; - ArrayList: внутренняя структура и алгоритмическая сложность; - LinkedList: двусвязный список и его применение; - итераторы и паттерн итератора.
18	Сравнение объектов в Java: Comparable и Comparator Рассматриваемые вопросы: - интерфейс Comparable и метод compareTo() для естественного порядка сортировки; - контракт согласованности compareTo() с equals() и транзитивность; - интерфейс Comparator как механизм настраиваемого сравнения; - современные возможности: Comparator.comparing(), thenComparing() и метод List.sort().
19	Коллекции Java: интерфейс Set Рассматриваемые вопросы: - HashSet и хеш-таблицы: принцип работы и коллизии; - LinkedHashSet и сохранение порядка вставки; - TreeSet и красно-чёрные деревья с применением Comparable/Comparator; - контракты equals() и hashCode().
20	Коллекции Java: интерфейс Map Рассматриваемые вопросы:

№ п/п	Тематика лекционных занятий / краткое содержание
	- HashMap: buckets, load factor и rehashing; - LinkedHashMap и TreeMap как альтернативы; - итерация по ключам и значениям; - применение Map для кэширования и индексации.
21	Очереди и двусторонние очереди Рассматриваемые вопросы: - интерфейс Queue и его реализации; - PriorityQueue и сортировка по приоритету; - интерфейс Deque и класс ArrayDeque; - применение ArrayDeque как стека и очереди.
22	Пользовательские структуры данных Рассматриваемые вопросы: - реализация стека на базе ArrayDeque; - реализация очереди с приоритетом; - бинарные деревья и обход в глубину/ширину; - сравнение производительности собственных и стандартных структур.
23	Алгоритмическая сложность и выбор структур Рассматриваемые вопросы: - нотация Big O для оценки сложности алгоритмов; - анализ временной и пространственной сложности; - компромиссы между памятью и скоростью; - практические рекомендации по выбору коллекций.
24	Форматы обмена данными: CSV и JSON Рассматриваемые вопросы: - структура и особенности формата CSV; - синтаксис JSON: объекты, массивы, типы данных; - сравнение CSV и JSON для различных сценариев; - стандарты и best practices при работе с форматами.
25	Сериализация и десериализация: концепция и реализации в Java Рассматриваемые вопросы: - понятие сериализации и десериализации как преобразования объекта в поток байт и обратно; - встроенный механизм Java: интерфейс Serializable, поле serialVersionUID и ключевое слово transient; - интерфейс Externalizable и его отличие от Serializable; - уязвимости безопасности нативной сериализации и современные альтернативы (JSON через Jackson).
26	Работа с датами и временем в Java (java.time API) Рассматриваемые вопросы: - недостатки устаревших классов Date и Calendar; - основные классы java.time: LocalDate, LocalTime, LocalDateTime, ZonedDateTime; - длительности и периоды: Duration и Period; - обработка часовых поясов, летнего времени и форматирование через DateTimeFormatter.
27	Функциональное программирование в Java Рассматриваемые вопросы: - лямбда-выражения и их синтаксис; - функциональные интерфейсы: Predicate, Function, Consumer, Supplier; - ссылки на методы (method references); - замыкания и захват переменных.
28	Stream API: конвейерная обработка данных Рассматриваемые вопросы:

№ п/п	Тематика лекционных занятий / краткое содержание
	- концепция Stream и отличие от Collection, создание потоков из различных источников; - промежуточные операции: filter, map, flatMap, distinct, sorted; - терминальные операции: collect, forEach, reduce, count; - коллекторы (Collectors), группировка данных и параллельные потоки.
29	Optional и null-safety Рассматриваемые вопросы: - класс Optional как контейнер для nullable-значений; - создание Optional: of, ofNullable, empty; - методы обработки: map, flatMap, orElse, orElseThrow; - best practices и антипаттерны использования Optional.
30	HTTP и REST: клиент-серверное взаимодействие Рассматриваемые вопросы: - архитектура протокола HTTP и его методы; - принципы REST и проектирование RESTful API; - java.net.http.HttpClient: формирование запросов; - обработка ответов и управление таймаутами.
31	JDBC и работа с PostgreSQL Рассматриваемые вопросы: - архитектура JDBC и драйверы СУБД; - установление соединения и управление подключением; - PreparedStatement и защита от SQL-инъекций; - транзакции и управление целостностью данных
32	Модульное и интеграционное тестирование Рассматриваемые вопросы: - архитектура JUnit 5: аннотации жизненного цикла, assertions и параметризованные тесты; - мокирование внешних зависимостей (HTTP-клиентов и слоев работы с БД) для изоляции бизнес-логики; - применение паттерна AAA (Arrange-Act-Assert) для тестирования сложных объектов и потоковых операций; - написание интеграционных тестов для проверки связки Java-приложения, Stream API и PostgreSQL

4.2. Занятия семинарского типа.

Лабораторные работы

№ п/п	Наименование лабораторных работ / краткое содержание
1	Настройка среды и примитивные типы Студент настраивает рабочую среду разработки, компилирует и запускает консольный калькулятор логистических параметров с использованием примитивных типов данных.
2	Операторы управления и валидация Студент реализует интерактивное консольное меню с применением операторов ветвления и циклов, обеспечивая базовую валидацию числовых входных данных пользователя.
3	Обработка массивов Студент реализует и отлаживает методы для инициализации, заполнения и агрегации данных в одномерных и многомерных массивах.
4	Рекурсивные алгоритмы Студент применяет рекурсивные алгоритмы для реализации функции поиска оптимального пути с пересадками в графе логистических узлов.

№ п/п	Наименование лабораторных работ / краткое содержание
5	Реализация классов и инкапсуляция Студент реализует классы предметной области с параметризованными конструкторами, геттерами и сеттерами, а также статическими полями для глобальных констант системы.
6	Неизменяемые объекты Студент создает неизменяемые объекты для справочников тарифов, применяя модификатор финальности к полям и ссылкам.
7	Наследование и переопределение Студент реализует иерархию классов с переопределением методов расчета специфических параметров доставки для каждого типа транспорта.
8	Полиморфизм и паттерн-матчинг Студент демонстрирует динамический полиморфизм, обрабатывая гетерогенный массив транспортных средств через ссылку базового типа и применяя современный паттерн-матчинг.
9	Композиция и агрегация Студент реализует сложные объектные графы, где Маршрут жестко владеет набором Участков через композицию, а Рейс ссылается на существующего Водителя через агрегацию.
10	Интеграция перечислений Студент интегрирует перечисления с пользовательскими полями и методами в бизнес-логику, используя их в современных выражениях выбора.
11	Рефакторинг по SOLID Студент выполняет рефакторинг кода для соблюдения принципов единственной ответственности и открытости-закрытости, выделяя логику в отдельные классы.
12	Регулярные выражения Студент реализует валидаторы данных с применением регулярных выражений, компилируя шаблоны для проверки форматов ввода.
13	Финансовые расчеты Студент реализует финансовые калькуляторы с использованием классов произвольной точности, применяя различные режимы округления для расчета налогов и сборов.
14	Обобщенные контейнеры Студент создает обобщенные классы и методы с применением ограничений типов для безопасного хранения и обработки разнородных сущностей.
15	Пользовательские исключения Студент реализует иерархию пользовательских классов исключений и настраивает многоуровневую обработку ошибок с выводом диагностической информации.
16	Файловый ввод-вывод Студент интегрирует файловый ввод-вывод для парсинга текстовых справочников и сохранения промежуточного состояния приложения.
17	Динамические списки Студент реализует динамический список заявок с применением различных реализаций списков, проводя замеры времени выполнения при массовых вставках и удалениях.
18	Интерфейсы сравнения Студент реализует интерфейсы естественного и настраиваемого сравнения, применяя современные методы-хелперы для многоступенчатой сортировки коллекций объектов.
19	Реализация множеств Студент применяет различные реализации множеств для обеспечения уникальности данных в системе и автоматического поддержания порядка элементов.
20	Кэши и справочники Студент реализует справочники и кэши с использованием различных реализаций ассоциативных массивов, анализируя влияние коэффициента загрузки на производительность.

№ п/п	Наименование лабораторных работ / краткое содержание
21	Диспетчеризация и история Студент реализует диспетчеризацию заявок с использованием очереди с приоритетом и применяет двустороннюю очередь для организации стека истории состояний.
22	Профилирование кода Студент проводит компьютерный эксперимент по профилированию кода, обосновывая выбор конкретных структур данных на основе эмпирических замеров.
23	Парсинг JSON Студент адаптирует разобранный паттерн парсинга для десериализации JSON-файлов в объекты проекта и сериализации результатов обработки обратно в файл.
24	Сохранение состояния Студент реализует сохранение объектов через встроенный механизм сериализации с исключением чувствительных полей и сравнивает результат с альтернативным текстовым подходом.
25	Расчеты с датами Студент применяет современный API работы с датами и временем для выполнения точных временных расчетов и форматирования вывода.
26	Лямбды и ссылки на методы Студент реализует собственные обработчики событий, предикаты и функции с использованием лямбда-выражений и ссылок на методы в своем проекте.
27	Конвейерная обработка Студент применяет конвейерную обработку потоков данных для формирования сводных аналитических отчетов по завершенным рейсам в своем проекте.
28	Безопасная работа с null Студент интегрирует контейнер для nullable-значений в цепочки вызовов конвейерной обработки и бизнес-логику для полного устранения ошибок нулевых ссылок.
29	HTTP-интеграция Студент адаптирует разобранный паттерн HTTP-интеграции для подключения к внешнему сервису получения актуальных данных о погоде в рамках своего проекта. В ходе работы настраиваются параметры запроса и обрабатываются сетевые исключения под специфику задачи
30	Слой доступа к данным Студент переносит разобранный паттерн работы с СУБД в свой проект, реализуя слой сохранения архива заявок с использованием параметризованных запросов и управлением транзакциями.
31	Модульные тесты и моки Студент применяет разобранный паттерн тестирования к своему проекту, создавая параметризованные тесты для бизнес-логики и изолируя слои работы с базой данных и внешними сервисами через моки.
32	Комплексное тестирование Студент проводит комплексное интеграционное тестирование связки всех компонентов проекта. По результатам тестирования оформляется итоговый технический отчет в специализированном офисном пакете.

Практические занятия

№ п/п	Тематика практических занятий/краткое содержание
1	Базовые алгоритмы и среда разработки Студент анализирует предметную область грузоперевозок и проектирует блок-схемы базовых алгоритмов расчета расстояния, стоимости и времени доставки между узловыми станциями.
2	Массивы и декомпозиция логики Студент проектирует структуру хранения матрицы расстояний между станциями и декомпозирует общую логику обработки данных на набор атомарных методов с определенными сигнатурами.

№ п/п	Тематика практических занятий/краткое содержание
3	Объектная модель и инкапсуляция Студент проектирует базовую объектную модель системы, определяя границы инкапсуляции и стратегию применения модификаторов доступа для сущностей Груз, Транспорт и Водитель.
4	Иерархии и полиморфизм Студент проектирует иерархию классов транспортных средств с выделением абстрактного базового класса и определением контрактов полиморфного поведения.
5	Отношения между классами и перечисления Студент моделирует объектные графы системы, выбирая между композицией и агрегацией для связей между сущностями, и проектирует типобезопасные справочники статусов заявок.
6	SOLID и валидация строк Студент проводит аудит спроектированного кода на соответствие принципам SOLID и разрабатывает спецификацию модуля валидации строковых идентификаторов.
7	Точные вычисления и обобщения Студент проектирует модуль финансовых расчетов с учетом требований к точности и создает архитектуру универсальных контейнеров для различных габаритов грузов.
8	Исключения и файловый ввод-вывод Студент разрабатывает стратегию обработки нештатных ситуаций и проектирует архитектуру модуля загрузки конфигурационных справочников из локальной файловой системы.
9	Списки и алгоритмическая сложность Студент анализирует алгоритмическую сложность операций над списками и проектирует модуль управления динамической очередью заявок на перевозку.
10	Сравнение объектов Студент проектирует стратегии сортировки грузов по весу, объему и срочности, определяя правила согласованности методов сравнения с проверкой на равенство.
11	Множества и уникальность Студент проектирует модуль учета уникальных идентификаторов транспортных средств, выбирая оптимальную структуру на основе хеш-таблиц или деревьев.
12	Ассоциативные массивы Студент проектирует архитектуру кэша тарифов и индекса маршрутов на базе ассоциативных массивов для обеспечения быстрого поиска по ключу.
13	Очереди и стеки Студент проектирует логику диспетчерской системы на базе очередей с приоритетом и механизм сохранения истории перемещений груза.
14	Профилирование и оптимизация Студент анализирует узкие места производительности в коллекциях системы и разрабатывает план оптимизации потребления памяти.
15	Форматы обмена данными Студент разбирает предоставленный эталонный пример парсера логистических манифестов, анализируя стратегию валидации входящих структур и обработки ошибок формата.
16	Сериализация и безопасность Студент изучает предоставленный пример безопасного сохранения состояния сессий, анализируя уязвимости встроенного механизма и стратегии изоляции критических полей.
17	Даты и время Студент проектирует модуль расчета сроков доставки с учетом часовых поясов, выходных дней и длительности простоя на таможне.
18	Функциональное программирование Студент разбирает эталонный пример декомпозиции бизнес-логики на функциональные интерфейсы и конвейеры обработки событий.

№ п/п	Тематика практических занятий/краткое содержание
19	Stream API Студент изучает эталонный пример аналитического модуля для агрегации, фильтрации и группировки больших объемов логов, выявляя структуру конвейерных операций.
20	Null-safety Студент проектирует стратегию безопасной работы с потенциально отсутствующими данными, такими как отмененный рейс или не найденный водитель.
21	Интеграция с внешним миром (HTTP) Студент детально разбирает предоставленный преподавателем эталонный пример HTTP-клиента, получающего данные из внешнего REST-сервиса. В процессе разбора студент анализирует структуру запроса, обработку таймаутов и проверку статус-кодов ответа.
22	Персистентность (JDBC) Студент изучает предоставленный преподавателем эталонный пример слоя доступа к реляционной базе данных. Разбирается архитектура JDBC-подключения, принцип параметризованных запросов и механизмы защиты от инъекций.
23	Обеспечение качества (JUnit 5) Студент разбирает предоставленный преподавателем эталонный набор модульных и интеграционных тестов. Анализируется структура тестового сценария, применение мокирования внешних зависимостей и организация assertions.
24	Финальная интеграция и защита Студент выполняет финальную интеграцию всех модулей системы и структурирует техническую документацию для защиты портфолио.

4.3. Самостоятельная работа обучающихся.

№ п/п	Вид самостоятельной работы
1	Изучение рекомендованной литературы.
2	Подготовка к лабораторным работам.
3	Подготовка к промежуточной аттестации.
4	Подготовка к текущему контролю.

5. Перечень изданий, которые рекомендуется использовать при освоении дисциплины (модуля).

№ п/п	Библиографическое описание	Место доступа
1	Гуськова, О. И. Объектно ориентированное программирование в Java : учебное пособие / О. И. Гуськова. — Москва : МПГУ, 2018. — 240 с. — ISBN 978-5-4263-0648-6. — Текст : электронный	https://e.lanbook.com/book/122311 (дата обращения: 08.04.2025)
2	Коузен, К. Современный Java: рецепты программирования / К. Коузен. — Москва : ДМК Пресс, 2018. — 275 с. — ISBN 978-5-97060-134-1. — Текст : электронный	https://e.lanbook.com/book/116121 (дата обращения: 08.04.2025)

3	Объектно-ориентированное программирование : учебное пособие / В. В. Лозовский, Е. Н. Штрекер, Е. С. Данилович [и др.]. — Москва : РТУ МИРЭА, 2025. — 484 с. — ISBN 978-5-7339-2498-4	https://e.lanbook.com/book/493547 (дата обращения: 21.04.2026)
4	Долгинцев, А. П. Объектно-ориентированное программирование : учебное пособие / А. П. Долгинцев. — Самара : СамГУПС, 2011. — 31 с. — Текст : электронный	https://e.lanbook.com/book/130277 (дата обращения: 08.04.2025)
5	Курбатова, И. В. Основы программирования на языке Java : учебное пособие для вузов / И. В. Курбатова, А. В. Печкуров. — Санкт-Петербург : Лань, 2024. — 348 с. — ISBN 978-5-507-48515-4. — Текст : электронный	https://e.lanbook.com/book/385928 (дата обращения: 08.04.2025)

6. Перечень современных профессиональных баз данных и информационных справочных систем, которые могут использоваться при освоении дисциплины (модуля).

Электронно-библиотечная система «Лань» – https://e.lanbook.com

Электронно-библиотечная система «Юрайт» – https://urait.ru

Oracle Java SE Documentation (официальная документация платформы Java Standard Edition) – https://docs.oracle.com/en/java/

Документация Postgres Pro (полная русскоязычная документация по реляционной СУБД PostgreSQL и её отечественной версии) – https://postgrespro.ru/docs/

JUnit 5 User Guide (официальное руководство пользователя фреймворка модульного тестирования JUnit 5) – https://junit.org/junit5/docs/current/user-guide/

Jackson JSON Processor Documentation (официальная документация и Wiki проекта FasterXML Jackson) – https://github.com/FasterXML/jackson-docs

OpenJDK java.net.http.HttpClient API (официальная документация модуля java.net.http) – https://docs.oracle.com/en/java/javase/17/docs/api/java.net.http/java/net/http/HttpClient.html

7. Перечень лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства, необходимого для освоения дисциплины (модуля).

Операционные системы: отечественные Linux-дистрибутивы (например, Astra Linux, РЕД ОС) для формирования навыков работы в защищенных корпоративных средах.

Офисные пакеты: отечественные решения из реестра ПО (МойОфис, Р7-Офис) для оформления технической документации и отчетов по ГОСТ.

Среды исполнения Java (JDK): доверенные отечественные дистрибутивы и Open Source (Axiom JDK, Liberica JDK, OpenJDK).

Интегрированные среды разработки (IDE): современные кроссплатформенные среды с глубокой поддержкой Java и инструментов рефакторинга (IntelliJ IDEA Community Edition, Eclipse).

Системы управления базами данных (СУБД): объектно-реляционные СУБД, включая отечественные сертифицированные версии (PostgreSQL, Postgres Pro).

Инструменты сборки и управления зависимостями: стандартные средства автоматизации жизненного цикла Java-проектов (Apache Maven, Gradle).

Фреймворки тестирования: библиотеки для модульного тестирования и изоляции зависимостей (JUnit 5, Mockito).

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

Учебные аудитории для проведения учебных занятий, оснащенные компьютерной техникой и наборами демонстрационного оборудования.

Для практических и лабораторных занятий – наличие персональных компьютеров вычислительного класса.

9. Форма промежуточной аттестации:

Экзамен в 1 семестре.
Зачет во 2 семестре.

10. Оценочные материалы.

Оценочные материалы, применяемые при проведении промежуточной аттестации, разрабатываются в соответствии с локальным нормативным актом РУТ (МИИТ).

Авторы:

старший преподаватель кафедры
«Цифровые технологии управления
транспортными процессами» И.С. Разживайкин

Согласовано:

Заведующий кафедрой ЦТУТП В.Е. Нутович

Председатель учебно-методической
комиссии Н.А. Андриянова