

Федеральное государственное образовательное бюджетное учреждение
высшего образования

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ
РОССИЙСКОЙ ФЕДЕРАЦИИ»
(Финансовый университет)**

**Кафедра информационных технологий
Факультет информационных технологий и анализа больших данных**

Документ подписан усиленной неквалифицированной электронной подписью
Организация: Финансовый университет при Правительстве РФ
Утверждено: Проректор по учебной и методической работе Е.А. Каменева
Сертификат: jOZmW5LcDX7CqKAMk6jBYQPBUEU3fhQe5
Дата: 27.11.2025 г.

А.А. Антонов, А. Эбрахим

Современные технологии программирования

Рабочая программа дисциплины

для студентов, обучающихся по направлению подготовки:

09.03.04 - Программная инженерия,

Образовательная программа «Инженер-разработчик программного обеспечения»

Профиль:

«Инженер-разработчик программного обеспечения»

Рекомендовано

*Факультет информационных технологий и анализа больших данных
(протокол № 03 от 16.12.2025 г.)*

Одобрено

*Кафедра информационных технологий
(протокол № 12 от 03.12.2025 г.)*

© Москва 2026

СОДЕРЖАНИЕ

1.	Наименование дисциплины	3
2.	Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине	3
3.	Место дисциплины в структуре образовательной программы	5
4.	Объем дисциплины (модуля) в зачетных единицах и в академических часах с выделением объема аудиторной (лекции, семинары) и самостоятельной работы обучающихся	5
5.	Содержание дисциплины, структурированное по темам (разделам) дисциплины с указанием их объемов (в академических часах) и видов учебных занятий	6
5.1.	Содержание дисциплины	6
5.2.	Учебно-тематический план	9
5.3.	Содержание семинаров	10
6.	Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине	12
6.1.	Перечень вопросов, отводимых на самостоятельное освоение дисциплины, формы внеаудиторной самостоятельной работы	12
6.2.	Перечень вопросов, заданий, тем для подготовки к текущему контролю	17
7.	Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине	21
8.	Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины	34
9.	Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины	34
10.	Методические указания для обучающихся по освоению дисциплины	36
11.	Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень необходимого программного обеспечения и информационных справочных систем	36
12.	Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине	37

1. Наименование дисциплины

«Современные технологии программирования».

2. Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине

Код компетенции	Наименование компетенции	Индикаторы достижения компетенции	Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции
ОПК-4	Способен участвовать в разработке стандартов, норм и правил, а также технической документации, связанной с профессиональной деятельностью	Демонстрирует знания основных стандартов ведения технической документации, как отечественных, так и зарубежных	знать: основные стандарты ведения технической, как отечественной, так и зарубежной технической документации уметь: составлять и вести, как отечественную, так и зарубежную техническую документацию
		Создает комплект программной и проектной документации к разрабатываемой информационной системе	знать: методы создания комплекта программной и проектной документации к разрабатываемой информационной системе уметь: создавать комплект программной и проектной документации к разрабатываемой информационной системе
		Критически анализирует программную и проектную документацию, строит на ее основе логические выводы	знать: методы критического анализа программной и проектной документации уметь: критически анализировать программную и проектную документацию; строить на основе программной и проектной документации логические выводы
ОПК-6	Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к	Разрабатывает алгоритмы решения простых информационных задач и выражает их на языке программирования	знать: методы разработки и алгоритмы решения простых информационных задач на языке программирования Java уметь: разрабатывать и применять алгоритмы решения простых информационных задач на языке программирования Java
		Анализирует алгоритмы в части производительности, оптимальности, вырабатывает рекомендации	знать: методы анализа алгоритмов в части производительности и оптимальности; методы создания рекомендаций для оптимизации

	проектированию, конструированию и тестированию программных продуктов	для оптимизации алгоритмов программ	алгоритмов программ уметь: анализировать алгоритмы в части производительности, оптимальности, а также вырабатывать рекомендации для оптимизации алгоритмов программ
		Проводит ручное и автоматизированное тестирование программных продуктов по методам черного и белого ящика, составляет набор тестовых случаев	знать: методы черного и белого ящика, составляет набор тестовых случаев, а также методы ручного и автоматизированного тестирования, в том числе и создания UNIT-тестов уметь: проводить ручное и автоматизированное тестирование программных продуктов по методам черного и белого ящика, составляет набор тестовых случаев
ОПК-2	Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности	Демонстрирует знания основных программных продуктов, используемых для решения задач профессиональной деятельности, в том числе, отечественного производства	знать: современные интегрированные среды разработки, используемые для решения задач профессиональной деятельности, в том числе, отечественного производства и банковских систем уметь: применять современные интегрированные среды при разработке и реализации программных продуктов
		Применяет готовые инструментальные средства для задач профессиональной деятельности, проводит квалифицированную их оценку и обосновывает свой выбор	знать: методы инструментальной разработки и возможности их применения при решении профессиональных задач; методы оценки существующих инструментальных средств уметь: применять методы инструментальной разработки и оценки существующих инструментальных средств, а также обосновывать их при реализации программных продуктов, в том числе и информационных систем

3. Место дисциплины в структуре образовательной программы

Дисциплина «Современные технологии программирования» относится к «Общефакультетскому (предпрофильному) циклу».

4. Объем дисциплины (модуля) в зачетных единицах и в академических часах с выделением объема аудиторной (лекции, семинары) и самостоятельной работы обучающихся

Очно-заочная форма обучения

Вид учебной работы по дисциплине	Всего (в з/е и часах)	Семестр 5 (в часах)	Семестр 6 (в часах)
Общая трудоёмкость дисциплины	6/216	108	108
Контактная работа- Аудиторные занятия	50	20	30
Лекции	8	4	4
Семинары, практические занятия	42	16	26
Самостоятельная работа	166	88	78
В том числе курсовая работа/курсовой проект	24	-	-
Вид текущего контроля	Проектная работа	Проектная работа	
Вид промежуточной аттестации	Зачет, Экзамен	Зачет	Экзамен

5. Содержание дисциплины, структурированное по темам (разделам) дисциплины с указанием их объемов (в академических часах) и видов учебных занятий

5.1. Содержание дисциплины

Тема 1. Обзор платформы и языка Java.

Java Runtime Environment, Java Development Kit, Java Virtual Machine, ClassLoader, Runtime Area, Execution Engine, Interpreter, Just-In-Time Compiler, Garbage Collector, Native Method Interface, ключевое слово native, примитивные и ссылочные типы данных.

Тема 2. Основные конструкции языка Java.

Основные операторы: арифметические, сравнения, логические, побитовые, присваивания, унарные. Ветвление: if/else, тернарный оператор, switch (традиционный и как выражение), стрелочная нотация (->). Циклы: while, do-while, for, for-each, метки; ключевые слова break, continue. Классы, абстрактные классы, интерфейсы, массивы, строки, записи (record), перечисления (enum), аннотации, анонимные классы, локальные классы, лямбда-выражения, ссылки на методы. Пакеты, модули, файлы module-info.java и package-info.java. Импорт пакетов и классов. Работа с датами (Java Time API).

Тема 3. Объектно-ориентированное программирование.

Конструктор, static block, instance block. Понятие наследования, наследование классов, реализация/наследование интерфейсов, ключевые слова: extends, implements, this, super, final. Sealed-классы, ключевые слова: sealed, non-sealed, permits. Понятие инкапсуляции, сеттеры, геттеры, модификаторы доступа, понятие полиморфизма, переопределение и перегрузка методов, оператор instanceof, восходящее и нисходящее преобразования, понятие абстракции, ключевое слово abstract, ключевое слово static.

Тема 4. Вложенные классы и интерфейсы. Исключения. Отладка. Обобщения. Коллекции.

Статические и нестатические вложенные классы, их создание и доступ к методам и атрибутам внешнего класса, вложенные интерфейсы, исключения и ошибки, проверяемые и непроверяемые исключения, ключевые слова try, catch, finally, throw, throws, try-with-resources, обобщенные классы, обобщенные интерфейсы, обобщенные методы, type erasure, джокеры, ключевые слова super и extends с обобщениями. Java Collections Framework (интерфейсы List, Set, Map).

Тема 5. Поточковая организация ввода-вывода. Многопоточность.

Потоки ввода (input streams), потоки вывода (output streams), байтовые потоки (binary streams), символьные потоки (text streams), буферизованные потоки (buffered streams), небуферизованные потоки (unbuffered streams), потоки (threads) и процессы (processes), этапы жизненного цикла потока, синхронизация потоков, понятие монитора, класс Thread, интерфейс Runnable, ключевое слово synchronized, методы wait(), notify(), notifyAll().

Тема 6. Функциональное программирование и системы сборки. Работа с базами данных.

Stream API: создание, промежуточные и терминальные операции, Maven, файл POM, жизненные циклы, стандартная структура проекта в Maven. Архитектура JDBC, Основные интерфейсы и классы в JDBC (DriverManager, Connection, Statement, PreparedStatement, ResultSet, SQLException), объектно-реляционное отображение, Hibernate (основы).

Тема 7. Веб-архитектура.

Клиент-серверная архитектура, веб-приложения, сервер приложений, протокол HTTP, REST API.

Тема 8. Реализация графических интерфейсов.

JavaFX, класс Application, FXML, класс Stage, класс Scene, понятие Graph Scene, инструмент Scene Builder.

Тема 9. Фреймворк Spring.

Понятие внедрения зависимости, понятие инверсия управления, Spring, Spring Boot, Spring Security, Spring Web, Spring JPA, Thymeleaf. Spring MVC, REST API.

Тема 10. Документирование и тестирование.

Комментарии Javadoc, JUnit 5, Mockito.

Тема 11. Паттерны и антипаттерны проектирования.

Паттерны порождающие (Singleton, Factory Method, Abstract Factory, Builder, Prototype). Структурные (Adapter, Bridge, Composite, Decorator, Facade, Flyweight, Proxy), Поведенческие (Chain of Responsibility, Command, Interpreter, Iterator, Mediator, Memento, Observer, State, Strategy, Template Method, Visitor). Антипаттерны (God Object, Spaghetti

Code, Lava Flow, Golden Hammer, Magic Numbers, Shotgun Surgery, Copy-Paste Programming, Feature Envy, Premature Optimization, Big Ball of Mud).

Тема 12. Системы контроля версий и процессы.

Git, GitLab CI/CD; Scrum, Agile, инструменты управления.

5.2. Учебно-тематический план

№ п/п	Наименование тем(разделов) дисциплины	Трудоемкость в часах				
		Всего	Контактная работа - Аудиторная работа			Самостоя тельная работа
			Общая, в т.ч.:	Лекции	Семинары, практическ ие занятия	
1	Обзор платформы и языка Java.	14	2	0	2	12
2	Основные конструкции языка Java.	19	5	1	4	14
3	Объектно-ориентированное программирование.	19	5	1	4	14
4	Вложенные классы и интерфейсы. Исключения. Отладка. Обобщения. Коллекции.	19	5	1	4	14
5	Потоковая организация ввода-вывода. Многопоточность.	19	5	1	4	14
6	Функциональное программирование и системы сборки. Работа с базами данных.	19	5	1	4	14
7	Веб-архитектура.	14	2	0	2	12
8	Реализация графических интерфейсов.	21	5	1	4	16
9	Фреймворк Spring.	30	10	2	8	20
10	Документирование и тестирование.	14	2	0	2	12
11	Паттерны и антипаттерны проектирования.	14	2	0	2	12
12	Системы контроля версий и процессы.	14	2	0	2	12
	Итого	216	50	8	42	166

5.3. Содержание семинаров

Наименование тем (разделов) дисциплины	Перечень вопросов для обсуждения на семинарах, практических занятиях	Формы проведения занятий
Обзор платформы и языка Java.	Установка и настройка JDK, JRE, переменные среды. Архитектура JVM: ClassLoader, Runtime Areas, Execution Engine. JIT-компиляция, Garbage Collector, Native Method Interface. Примитивные и ссылочные типы данных, ключевое слово native. Создание и запуск первого Java-приложения.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Основные конструкции языка Java.	Операторы: арифметические, сравнения, логические, побитовые, присваивания. Ветвление: if-else, тернарный оператор, switch (традиционный и современный). Циклы: while, do-while, for, for-each, использование меток и управляющих слов. Классы, интерфейсы, массивы, строки, записи, перечисления, аннотации. Лямбда-выражения, ссылки на методы, работа с датами (Java Time API). Практическое создание и использование пакетов.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Объектно-ориентированное программирование.	Конструкторы, статические и instance блоки инициализации. Наследование: extends, implements. Практическое сравнение композиции и наследования. Инкапсуляция: модификаторы доступа, геттеры/сеттеры. Полиморфизм: перегрузка и переопределение методов. Абстракция: абстрактные классы и методы. Ключевые слова: this, super, final, static, abstract. Преобразование типов, оператор instanceof.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Вложенные классы и интерфейсы. Исключения. Отладка. Обобщения. Коллекции.	Статические и нестатические вложенные классы. Обработка исключений: try-catch-finally, try-with-resources. Создание и проброс исключений: throw, throws. Обобщенные классы, методы, интерфейсы. Type Erasure, Wildcards. Коллекции: List, Set, Map и их реализации. Итерация и работа с коллекциями.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Потоковая организация ввода-вывода. Многопоточность.	Байтовые и символьные потоки. Буферизованные и небуферизованные потоки. Работа с файлами: чтение и запись. Создание и управление потоками: Thread, Runnable. Жизненный цикл потока. Синхронизация: synchronized, wait(), notify(), notifyAll().	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Функциональное программирование и системы сборки.	Stream API: создание потоков. Промежуточные и терминальные операции. Сборка проектов в Maven: POM, жизненные циклы. Архитектура	Дискуссия по результатам самостоятельной

Работа с базами данных.	JDBC: Connection, Statement, ResultSet. Основы Hibernate: маппинг сущностей, аннотации. Транзакции, уровни изоляции.	работы, решение задач, аудиторная проверочная работа.
Веб-архитектура.	Клиент-серверная архитектура. Протокол HTTP, методы запросов, статусы ответов. REST API, принципы построения. Сервер приложений. Создание простого веб-приложения.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Реализация графических интерфейсов.	JavaFX: класс Application, Stage, Scene. FXML, Scene Builder. Обработка событий, привязка данных. Компоненты: Button, TextField, TableView, Label. Работа с контроллерами и организация взаимодействия между FXML и кодом приложения.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Фреймворк Spring.	Inversion of Control, Dependency Injection. Spring Boot, автоконфигурация. Spring MVC, REST-контроллеры. Spring Security, аутентификация и авторизация. Spring Data JPA, интеграция с Hibernate. Шаблонизатор Thymeleaf. Создание полноценного веб-приложения.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Документирование и тестирование.	Javadoc: структура комментариев, теги. Модульное тестирование: JUnit 5, аннотации. Mockito: моки, стабы, верификация. Интеграционное тестирование. Написание тестов для Spring-приложений.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Паттерны и антипаттерны проектирования.	Принципы проектирования (SOLID) как основа паттернов. Порождающие паттерны: Singleton, Factory, Builder. Структурные паттерны: Adapter, Decorator, Proxy. Поведенческие паттерны: Observer, Strategy, Template Method. Антипаттерны: God Object, Spaghetti Code, Magic Numbers. Рефакторинг антипаттернов. Применение паттернов в реальных проектах.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.
Системы контроля версий и процессы.	Git: основные команды, ветвление, слияние. Разрешение конфликтов слияния в Git. Работа с удаленными репозиториями (GitHub/GitLab): push, pull, pull request. Принципы Agile, манифест. Фреймворк Scrum: роли, артефакты, события. Обзор процессов CI/CD и инструментов управления проектам.	Дискуссия по результатам самостоятельной работы, решение задач, аудиторная проверочная работа.

6. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине

6.1. Перечень вопросов, отводимых на самостоятельное освоение дисциплины, формы внеаудиторной самостоятельной работы

Наименование тем (разделов) дисциплины	Перечень вопросов, отводимых на самостоятельное освоение	Формы внеаудиторной самостоятельной работы
Обзор платформы и языка Java.	Разработка и реализация собственного загрузчика классов (ClassLoader). Основные типы литералов в Java.	Проектирование и разработка собственной нативной библиотеки (JNI).
Основные конструкции языка Java.	Особенности работы тернарного оператора. Создание и назначение пользовательских аннотаций. Вложенные циклы. Стилевое оформление программного кода.	Изучение теоретического материала по конструктам языка Java с использованием официальной документации Oracle. Выполнение индивидуальных мини-заданий на применение операторов, ветвлений и циклов в программах. Самостоятельная разработка программных примеров с использованием записей (record), перечислений (enum) и аннотаций. Выполнение лабораторных мини-заданий на создание и использование пакетов, модулей и импортируемых классов. Решение практических задач по обработке дат и времени с использованием API java.time.
Объектно-	Базовый класс, класс Object: его роль в иерархии	Выполнение

ориентированное программирование.	Java, методы equals(), hashCode(), toString(), clone(). Многоуровневая иерархия классов. Оператор instanceof. Восходящее и нисходящее преобразование объектов. Передача объектов методам. Инициализация полей по умолчанию. Вызов одного конструктора из другого.	практических заданий по созданию классов с конструкторами и блоками инициализации. Разработка примеров применения модификаторов доступа и демонстрация инкапсуляции. Проведение анализа различий между абстрактными классами и интерфейсами в виде сравнительной таблицы. Реализация примеров применения sealed-классов и демонстрация их ограничений. Проведение экспериментов по преобразованию типов с использованием instanceof.
Вложенные классы и интерфейсы. Исключения. Отладка. Обобщения. Коллекции.	Настройка конфигурации запуска программы в режиме отладки и анализ поведения приложения при пошаговом выполнении. Установка и использование различных типов точек останова (обычных и условных) в коде. Выполнение пошагового анализа программы с применением команд Step Into, Step Over и Step Out. Исследование значений переменных и выражений в окне Variables и инструменте Evaluate Expression. Проведение экспериментов по изменению значений переменных во время отладки и наблюдение влияния на ход программы.	Создание собственных обобщённых классов и интерфейсов с параметрами типа. Разработка мини-программы для сравнения поведения коллекций при добавлении, удалении и поиске элементов. Применение Stream API с коллекциями.
Потоковая организация ввода-вывода. Многопоточность.	Ознакомление со следующими классами: InputStream, FileInputStream, ByteArrayInputStream, FilterInputStream, BufferedInputStream, DataInputStream, ObjectInputStream, PipedInputStream, OutputStream, FileOutputStream, ByteArrayOutputStream, FilterOutputStream, BufferedOutputStream, DataOutputStream, ObjectOutputStream, PipedOutputStream, Reader, FileReader, CharArrayReader, BufferedReader, InputStreamReader, StringReader, Writer, FileWriter, CharArrayWriter, BufferedWriter,	Освоение механизмов синхронизации и координации потоков с реализацией программных примеров. Анализ проблемы синхронизации: deadlock, race condition, starvation.

	OutputStreamWriter, PrintWriter, StringWriter, PrintStream, PipedReader, PipedWriter, FileChannel, Files, Paths, Path Создание и управление потоками: Thread, Runnable, Callable, Future, Executor, ExecutorService, ScheduledExecutorService, ForkJoinPool, Virtual Threads. Работа с пулами потоков: Executors, методы submit(), invokeAll(), shutdown(), awaitTermination(). Синхронизация потоков: synchronized, ReentrantLock, ReentrantReadWriteLock, Condition, volatile. Механизмы координации: CountdownLatch, CyclicBarrier, Semaphore, Phaser, Exchanger. Мониторинг и взаимодействие: wait(), notify(), notifyAll(), LockSupport. Атомарные классы и операции CAS: AtomicInteger, AtomicReference, VarHandle. Асинхронное программирование: CompletableFuture, Reactive Streams.	
Функциональное программирование и системы сборки. Работа с базами данных.	Основы Gradle. Groovy и Kotlin синтаксис. Хранимые процедуры.	Реализация примеров применения Stream API с фильтрацией, сортировкой и агрегированием данных. Создание и сборка учебного проекта с использованием Maven и Gradle; сравнение их конфигураций. Настройка зависимости и подключение внешней библиотеки через файл pom.xml или build.gradle. Разработка программы для подключения к базе данных с использованием JDBC и выполнения SQL-запросов. Реализация примеров использования PreparedStatement для защиты от SQL-инъекций. Настройка и использование Hibernate для маппинга и сохранения данных в

		таблицу.Разработка программы для подключения к базе данных с использованием Hibernate.
Веб-архитектура.	Изучение структуры HTTP-запроса/ответа.	Отправление различных типов HTTP-запросов на сайты, предоставляющие бесплатный доступ к API (например, https://postman-echo.com), с использованием инструментов curl или Postman для практического изучения работы протокола HTTP.
Реализация графических интерфейсов.	Применение CSS в контексте JavaFX.	Разработка простого JavaFX-приложения с использованием FXML и контроллеров.Добавление CSS-стилей для визуального оформления приложения.
Фреймворк Spring.	Spring Security. JWT-токен. Аспектно-ориентированное программирование (АОП) в Spring.	Разработка простого REST-приложения с контроллером, сервисом и репозиторием.Самостоятельная настройка проекта с помощью Spring Initializr и изучение структуры сгенерированного проекта.Изучение работы механизма аутентификации и авторизации в Spring Security.Реализация авторизации с использованием JWT-токена.Ознакомление

		с принципами аспектно-ориентированного программирования (АОР) и применение аспектов для логирования или валидации. Разработка мини-проекта с использованием Thymeleaf и Spring Boot.
Документирование и тестирование.	Дескрипторы javadoc. Параметризованные тесты. Динамические тесты. Повторяющиеся тесты.	Создание и оформление Javadoc-документации для собственного учебного проекта. Разработка набора тестов для одного из ранее реализованных модулей проекта.
Паттерны и антипаттерны проектирования.	Паттерны: Prototype, Bridge, Composite, Facade, Flyweight, Chain of Responsibility, Command, Interpreter, Iterator, Mediator, Memento, State, Visitor. Антипаттерны: Lava Flow, Golden Hammer, Shotgun Surgery, Copy-Paste Programming, Feature Envy, Premature Optimization, Big Ball of Mud.	Разработка примеров кода для различных паттернов проектирования.
Системы контроля версий и процессы.	Принцип работы rebase и отличие его от merge. Использование интерактивного rebase (git rebase -i) для редактирования и объединения коммитов. Применение команды cherry-pick для выборочного переноса изменений между ветками. Настройка и использование Git LFS (Large File Storage) для работы с большими файлами. Восстановление потерянных коммитов и веток с помощью reflog. Применение git bisect для поиска ошибки в истории проекта.	Создание репозитория, ветвление, слияние, работа с удалённым репозиторием. Разработка учебного проекта с применением GitLab CI/CD для автоматической сборки и тестирования.

6.2. Перечень вопросов, заданий, тем для подготовки к текущему контролю

Примерные задания проектной работы

Задание 1.

Реализовать проект в виде информационной системы «Онлайн-библиотека» с веб-интерфейсом. Необходимый функционал: добавление книги; удаление книги; редактирование информации о книге; поиск книги по различным параметрам; отдельный фильтр (сортировка) по дате издания книги; гистограмма количества добавленных книг по месяцам; счётчик книг в таблице; подсчёт количества новых поступлений за один день; система регистрации и авторизации с разграничением прав доступа (какой функционал будет доступен определённой группе пользователей определяет сам разработчик); раздел «Учёт читателей» (ссылка на него должна быть в верхнем меню), в котором должны быть доступны следующие подразделы: «Административная панель управления читателями», в которой необходимо добавлять/удалять/редактировать информацию о читателях; «Главная страница читателей», на которой выводятся все добавленные через административную панель записи (ФИО, номер читательского билета, дата регистрации, количество взятых книг). Параметры объекта «Книга»: ID, Название книги, Автор, Жанр, Издательство, Год издания, Количество страниц, Дата добавления в библиотеку, Статус (в наличии/выдана).

Задание 2.

Реализовать проект в виде информационной системы «Магазин цифровой техники» с веб-интерфейсом. Необходимый функционал: добавление товара; удаление товара; редактирование информации о товаре; поиск товара по различным параметрам (название, категория, бренд, цена); отдельный фильтр (сортировка) по дате поступления товара; гистограмма количества проданных товаров по дням; счётчик товаров в таблице; подсчёт количества проданных единиц техники за один день; система регистрации и авторизации с разграничением прав доступа (какой функционал будет доступен определённой группе пользователей определяет сам разработчик); раздел «Управление акциями» (ссылка на него должна быть в верхнем меню), в котором должны быть доступны следующие подразделы: «Административная панель акций», в которой необходимо добавлять/удалять/редактировать акции и скидки; «Главная страница

акций», на которой выводятся все добавленные через административную панель предложения (Название акции, срок действия, размер скидки, описание). Параметры объекта «Товар»: ID, Название товара, Категория (смартфон, ноутбук, планшет и т. д.), Бренд, Модель, Цена, Количество на складе, Дата поступления, Описание, Статус (в наличии/нет в наличии).

Задание 3.

Реализовать проект в виде информационной системы «Перевозка грузов» с веб-интерфейсом. Необходимый функционал: добавление груза; удаление груза; редактирование груза; поиск груза по различным параметрам; отдельный фильтр (сортировка) по дате поставки груза; гистограмма количества грузов по дням; счетчик грузов в таблице; подсчет количества прихода грузов за один день; система регистрации и авторизации с разграничением прав доступа (какой функционал будет доступен определенной группе пользователей определяет сам разработчик); раздел «Автоблог» (ссылка на него должна быть в верхнем меню), в котором должны быть доступны следующие подразделы: «Административная панель управления блогом», в котором необходимо добавлять/удалять/редактировать записи; «Главная страница блога», на которой выводятся все добавленные через административную панель управления записи (Название записи, дата, кто добавил, текст записи). Параметры объекта «Груз»: ID, Название груза, Содержимое груза, Город отправки груза, Дата отправки груза, Город прибытия груза, Дата прибытия груза.

Задание 4.

Реализовать проект в виде информационной системы «Кинотеатр» с веб-интерфейсом. Необходимый функционал: добавление сеанса; удаление сеанса; редактирование сеанса; поиск сеанса по различным параметрам; отдельный фильтр (сортировка) по дате сеанса; гистограмма количества сеансов; счетчик сеансов в таблице; подсчет количества сеансов за один день; система регистрации и авторизации с разграничением прав доступа (какой функционал будет доступен определенной группе пользователей определяет сам разработчик); раздел «Новости кинотеатра» (ссылка на него должна быть в верхнем меню), в котором должны быть доступны следующие подразделы: «Административная панель управления блогом», в котором необходимо добавлять/удалять/редактировать записи; «Главная страница блога», на которой выводятся все добавленные через административную панель управления записи (Название записи, дата, кто

добавил, текст записи). Параметры объекта «Сеанс»: ID, Название фильма, Дата и время сеанса, Количество билетов на сеанс.

Критерии балльной оценки различных форм текущего контроля успеваемости содержатся в соответствующих методических рекомендациях Кафедры информационных технологий Факультета информационных технологий и анализа больших данных.

Дополнительная информация

Примеры тестовых заданий

1. Результатом работы данного фрагмента кода будет

```
for(;;) {  
  
}
```

-: ошибка на этапе компиляции

-: ошибка на этапе выполнения

+: бесконечный цикл

-: этот код никогда не выполнится

-: компилятор удалит лишнюю точку с запятой

2. Задан код

```
char[] helloArray = { 'h', 'e', 'l', 'l', 'o', '.' };  
  
String helloString = new String(helloArray);  
  
System.out.println ( helloString );
```

в результате работы программы в консоль будет выведено

-: h e l l o .

-: [h, e, l, l, o, .]

+: hello.

-: hello

-: ['h', 'e', 'l', 'l', 'o', ' ']

3. В результате запуска кода

```
public class Main {  
    public static void main(String[] args) {  
        ArrayList<Integer> list;  
        for (int i = 0; i < 10; i++){  
            list.add(i);  
        }  
        System.out.println(list);  
    }  
}
```

в консоль будет выведено

-: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

-: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

-: [0, 9]

+: java: variable list might not have been initialized

-: java : ';' expected

7. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине

Перечень компетенций с указанием индикаторов их достижения в процессе освоения образовательной программы содержится в разделе 2. *Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине.*

Типовые контрольные задания или иные материалы, необходимые для оценки индикаторов достижения компетенций, умений и знаний

ОПК-4 Способен участвовать в разработке стандартов, норм и правил, а также технической документации, связанной с профессиональной деятельностью

1) Демонстрирует знания основных стандартов ведения технической документации, как отечественных, так и зарубежных

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: основные стандарты ведения технической, как отечественной, так и зарубежной технической документации

Уметь: составлять и вести, как отечественную, так и зарубежную техническую документацию

Типовые контрольные задания

Рассказать основные существующие стандарты, применяемые к технической и зарубежной документации, а также объяснить их основную суть. Разработать информационную систему с реализацией различных статистических функций. Составить техническую документацию на данную систему с использованием JavaDoc .

2) Создает комплект программной и проектной документации к разрабатываемой информационной системе

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: методы создания комплекта программной и проектной документации к разрабатываемой информационной системе

Уметь: создавать комплект программной и проектной документации к разрабатываемой информационной системе

Типовые контрольные задания

В команде разработать информационную систему по управлению научно-исследовательскими проектами с использованием ООП, JPA и технологии Hibernate .

Составить техническую и проектную документацию на данную систему с использованием JavaDoc .

3) Критически анализирует программную и проектную документацию, строит на ее основе логические выводы

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: методы критического анализа программной и проектной документации

Уметь: критически анализировать программную и проектную документацию; строить на основе программной и проектной документации логические выводы

Типовые контрольные задания

Разработать информационную систему по поставке грузов с использованием ООП, JPA и технологии Hibernate . Составить техническую и проектную документацию на данную систему с использованием JavaDoc . Проанализировать программную и проектную документацию, выявить ошибки и исправить их.

ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов

1) Разрабатывает алгоритмы решения простых информационных задач и выражает их на языке программирования

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: методы разработки и алгоритмы решения простых информационных задач на языке программирования Java

Уметь: разрабатывать и применять алгоритмы решения простых информационных задач на языке программирования Java

Типовые контрольные задания

Описать и объяснить основные принципы ООП.

Создать веб-приложение на основе Spring Boot — «Список задач».

Каждая задача (Task) должна содержать поля: идентификатор, заголовок, описание, дата выполнения, статус (выполнено/не выполнено).

Реализовать REST-контроллер с методами:

GET /tasks — получение списка задач;

POST /tasks — добавление новой задачи;

PUT /tasks/{id} — обновление задачи;

DELETE /tasks/{id} — удаление задачи.

Использовать базу данных H2 и библиотеку Spring Data JPA.

Добавить отображение списка задач в шаблоне Thymeleaf и возможность изменения статуса задачи из веб-интерфейса.

2) Анализирует алгоритмы в части производительности, оптимальности, вырабатывает рекомендации для оптимизации алгоритмов программ

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: методы анализа алгоритмов в части производительности и оптимальности; методы создания рекомендаций для оптимизации алгоритмов программ

Уметь: анализировать алгоритмы в части производительности, оптимальности, а также вырабатывать рекомендации для оптимизации алгоритмов программ

Типовые контрольные задания

Описать и объяснить основные принципы ООП.

Разработать информационную систему «Таксопарк» с использованием ООП, JPA , технологии Hibernate и RESTful API.

Оптимизировать производительность системы, путем замены ORM на нативные SQL -запросы.

3) Проводит ручное и автоматизированное тестирование программных продуктов по методам черного и белого ящика, составляет набор тестовых случаев

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: методы черного и белого ящика, составляет набор тестовых случаев, а также методы ручного и автоматизированного тестирования, в том числе и создания UNIT-тестов

Уметь: проводить ручное и автоматизированное тестирование программных продуктов по методам черного и белого ящика, составляет набор тестовых случаев

Типовые контрольные задания

Описать и объяснить основные принципы ООП.

Разработать информационную систему «Таксопарк» с использованием ООП, JPA , технологии Hibernate и RESTful API.

Оптимизировать производительность системы, путем замены ORM на нативные SQL -запросы.

Создать UNIT -тесты данной системы.

ОПК-2 Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности

1) Демонстрирует знания основных программных продуктов, используемых для решения задач профессиональной деятельности, в том числе, отечественного производства

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: современные интегрированные среды разработки, используемые для решения задач профессиональной деятельности, в том числе, отечественного производства и банковских систем

Уметь: применять современные интегрированные среды при разработке и реализации программных продуктов

Типовые контрольные задания

Описать и объяснить основные принципы работы любой интегрированной среды разработки.

Демонстрировать навыки работы с интегрированными средами. Разработать программный продукт: определить класс Student, хранящий информацию о студенте: ФИО, номер группы, средний балл и направление подготовки.

Создать дочерние классы Undergraduate и Graduate, расширяющие функциональность базового класса дополнительным полем degreeType («бакалавр» или «магистр»).

Разработать программу, в которой создается массив (коллекция) объектов данных классов.

Реализовать сортировку студентов по среднему баллу и вывод информации на экран.

Добавить метод поиска студента по ФИО с использованием Stream API.

Переопределить метод toString() для красивого форматированного вывода данных о студенте.

2) Применяет готовые инструментальные средства для задач профессиональной деятельности, проводит квалифицированную их оценку и обосновывает свой выбор

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: методы инструментальной разработки и возможности их применения при решении профессиональных задач; методы оценки существующих инструментальных средств

Уметь: применять методы инструментальной разработки и оценки существующих инструментальных средств, а также обосновывать их при реализации программных продуктов, в том числе и информационных систем

Типовые контрольные задания

Описать и объяснить основные принципы ООП.

Создать программу, которая считывает из текстового файла список чисел, вычисляет их среднее значение, минимальное и максимальное значение, а также количество отрицательных чисел.

Реализовать обработку исключений с использованием конструкции try-with-resources.

Результаты вычислений сохранить в новый файл result.txt.

Добавить логирование основных действий программы при помощи класса java.util.logging.Logger.

Примеры практико-ориентированных заданий

1. Создать Spring Boot-приложение «Галерея изображений», позволяющее загружать, хранить и просматривать изображения, сохранённые на сервере.

2. Разработать JavaFX-приложение вычисления числа Фибоначчи для n (итеративно и рекурсивно), сравнить время выполнения, оформить Javadoc и JUnit 5 тесты.

3. Разработать Spring Boot-приложение «Отправка писем», в котором реализована возможность отправки электронных писем через REST-эндпоинт с использованием Spring Mail.
4. Написать JavaFX-приложение, которое открывает выбранный пользователем CSV-файл с числами и вычисляет среднее, максимум, минимум.
5. Разработать Spring Boot MVC-приложение «Randomizer»: генерирует 10 случайных чисел в диапазоне пользователя, показывает их и среднее, выдаёт JSON по /api/random.
6. Создать JavaFX-приложение «Таймер / Секундомер» с возможностью старт / пауза / сброс и сохранением кругов (laps) в CSV.
7. Разработать Spring Boot-приложение, которое показывает пользователю вопросы с вариантами ответов и подсчитывает результат после завершения теста.
8. Написать JavaFX-приложение для вычисления площади и объёма цилиндра (радиус, высота). Добавить Javadoc и JUnit 5 тесты для формул.
9. Разработать JavaFX-приложение «Фильтр чисел»: показать все числа диапазона, делящиеся на 3 и 5, в убывающем порядке. Оформить Javadoc и JUnit 5 тесты.
10. Создать Spring Boot-приложение «Погодный сервис», которое получает текущую погоду по названию города с внешнего API и отображает результат в браузере с использованием Thymeleaf.
11. Разработать Spring Boot-приложение «Заметки», реализующее REST API для создания, редактирования и удаления заметок с авторизацией пользователей по JWT.
12. Создать JavaFX-приложение с формой регистрации (имя, e-mail, пароль, подтверждение пароля) с валидацией полей и документированием на основе Javadoc.
13. Разработать JavaFX-приложение операций над комплексными числами (сложение, вычитание, умножение, деление, модуль, аргумент). Оформить Javadoc и JUnit 5 тесты.
14. Создать Spring Boot-приложение «Список задач», реализующее CRUD-операции (добавление, удаление, редактирование, поиск задач) с использованием H2 Database и Spring Data JPA.
15. Написать JavaFX-приложение генерации N случайных целых чисел в заданном диапазоне с отображением гистограммы распределения. Оформить Javadoc.
16. Разработать Spring Boot-приложение «Учёт расходов», в котором можно добавлять расходы по категориям и датам, а также получать отчёт в формате JSON.

17. Написать JavaFX-приложение «Конвертер валют»: загрузка курсов из локального JSON (эмуляция API), пересчёт сумм, форматирование по локали.
18. Реализовать JavaFX-визуализацию бинарного дерева поиска: вставка, удаление, поиск с анимацией обходов (in / pre / post-order). Оформить Javadoc.
19. Создать Spring Boot REST-сервис «Students» (CRUD: имя, фамилия, возраст, группа) с базой данных (H2 / MySQL / PostgreSQL).
20. Разработать JavaFX-приложение «Статистика текста»: пользователь вставляет текст, программа выводит частоты слов и строит столбчатую диаграмму.
21. Разработать Spring Boot MVC калькулятор матриц (сложение / умножение) с Thymeleaf-шаблонами и валидацией размеров матриц.
22. Разработать калькулятор дробей (сложение, вычитание, умножение, деление, сокращение) с интерфейсом на JavaFX. Покрыть арифметику тестами JUnit 5.
23. Написать JavaFX-приложение «Менеджер заметок»: CRUD заметок, поиск по заголовку / тегам, автосохранение, экспорт в Markdown.
24. Разработать JavaFX-приложение, выводящее все чётные числа заданного диапазона в одну строку и сохраняющее результат в файл. Добавить JUnit 5 тесты.
25. Создать Spring Boot-приложение, которое по адресу /hello выводит «Hello, Spring Boot!». Написать модульный тест на JUnit 5 и тест контроллера на Spring MockMvc.

Примерные вопросы для подготовки к зачету, экзамену

Примерные вопросы для подготовки к зачету

1. Платформа Java: Java Development Kit (JDK): состав, назначение. Кроссплатформенность языка Java.
2. Java virtual machine (JVM), JIT-компилятор – определение, свойства, функции. Принципы работы сборщика мусора.

3. Системы сборки проектов. Фреймворк Apache Maven: определение, структура, Maven Coordinates, POM-файл.
4. Типы данных Java: простые и ссылочные. Простые (примитивные) типы данных.
5. Переменные: статические и нестатические. Местные переменные, область видимости переменных. Объявление и инициализация переменных. Константы. Спецификаторы доступа.
6. Комментарии: виды, особенности применения.
7. Операции языка Java – арифметические, отношения и логические, преобразования числовых типов.
8. Символьные строки, методы работы со строками Java.
9. Классы StringBuffer и StringBuilder.
10. Массивы Java: объявление, инициализация. Основные методы класса Arrays. Доступ к элементам массивов, итерация массивов. Двумерные массивы.
11. Управляющие конструкции Java: ветвление, циклы. Цикл foreach.
12. Перечислить и дать описание основных принципов объектно-ориентированного программирования (ООП). Достоинства и недостатки ООП.
13. Определение класса. Объявление класса. Спецификаторы доступа. Отношения между классами Java (наследование, зависимость, агрегирование). Статические члены класса. Переменные класса.
14. Объект класса. Создание объекта. Конструктор класса, конструктор по умолчанию. Ключевое слово this. Перегрузка конструкторов. Доступ к переменным экземпляра.
15. Методы, объявление, имя. Статические методы. Доступ к методам. Спецификаторы доступа.
16. Пакеты Java. Импорт пакетов и классов. Статический импорт.
17. Вложенные и внутренние классы Java. Статические и нестатические внутренние классы.
18. Наследование. Подклассы и суперклассы. Доступ к членам класса. Конструкторы при наследовании, ключевое слово super.
19. Иерархия наследования Java. Преобразование типов при наследовании. Ключевое слово instanceof.

20. Полиморфизм в Java. Перегрузка и переопределение методов.
21. Абстрактные методы и классы Java.
22. Интерфейсы Java: определение интерфейса, реализация интерфейса. Преимущества применения интерфейсов. Переменные интерфейсов. Наследование интерфейсов. Методы по умолчанию. Статические методы интерфейсов.
23. Исключения (exception) Java. Синтаксис объявления исключений. Классификация исключений. Основные классы для работы с исключениями. Исключения при наследовании.
24. Коллекции: Java collections framework. Классификация интерфейсов коллекций. Интерфейс Collection.
25. Списки. Интерфейс List. Основные классы, реализующие интерфейс List. ArrayList, особенности, методы. Comparator.
26. Интерфейс Set. Основные реализации. HashSet. TreeSet.
27. Очереди в Java, интерфейс Queue, PriorityQueue. Структура, основные методы.
28. Байтовые потоки InputStream и OutputStream. Консольный ввод и вывод Java. Символьные потоки данных. Абстрактные классы Writer, Reader.
29. Чтение и запись файлов. Классы FileInputStream, FileOutputStream. Файловый ввод-вывод с использованием символьных потоков. Классы FileReader и FileWriter.
30. Многопоточное программирование: общие принципы.
31. Класс Thread и интерфейс Runnable: создание потоков, приоритеты потоков.
33. Синхронизация потоков.
34. Основные интерфейсы и классы JDBC и их назначение.

Примерные вопросы для подготовки к экзамену

1. Пакеты Java. Импорт пакетов и классов. Статический импорт.
2. Вложенные и внутренние классы Java. Статические и нестатические внутренние классы.
3. Наследование. Подклассы и суперклассы. Доступ к членам класса. Конструкторы при наследовании, ключевое слово super.

4. Иерархия наследования Java. Преобразование типов при наследовании. Ключевое слово instanceof.
5. Полиморфизм в Java. Перегрузка и переопределение методов.
6. Абстрактные методы и классы Java.
7. Интерфейсы Java: определение интерфейса, реализация интерфейса. Преимущества применения интерфейсов. Переменные интерфейсов. Наследование интерфейсов. Методы по умолчанию. Статические методы интерфейсов.
8. Исключения (exception) Java. Синтаксис объявления исключений. Классификация исключений. Основные классы для работы с исключениями. Исключения при наследовании.
9. Платформа JavaFX, особенности, компоненты.
10. Шаблон MVC (Model-View-Controller) в JavaFX.
11. Классы Stage и Scene в JavaFX.
12. Узлы и графы сцены в JavaFX.
13. Класс Application и жизненный цикл приложения JavaFX.
14. Инструмент визуальной компоновки JavaFX Scene Builder.
15. Компоненты управления Label, TextField в JavaFX.
16. Компонент управления Button в JavaFX.
17. Основные фреймворки и задачи, решаемые Spring.
18. Spring Inversion of Control (IoC) контейнер Spring.
19. Dependency Injection (DI) в Spring.
20. Жизненный цикл объекта Bean Spring.
21. Конфигурация ApplicationContext с помощью xml в Spring.
22. Область видимости Bean в Spring.
23. Фабричные или factory-методы в Spring.
24. Конфигурация ApplicationContext с помощью аннотаций в Spring.
25. Связывание в Spring, аннотация @Autowired.

26. Архитектурный стиль REST.
27. Spring Web-MVC, основная схема и логика работы.
28. Класс DispatcherServlet, его функции.
29. Маппинг в Spring.
30. Интерфейсы HttpServletRequest и HttpServletResponse.
31. Архитектурный стиль CRUD, его соответствие REST и HTTP.
32. Шаблон Data Access Object (DAO).
33. Основные понятия Объектно-реляционного отображения (ORM - Object Relational Mapping).
34. Спецификация Java Persistence API (JPA).
35. Архитектура ORM Java Persistence API (JPA).
36. Основные аннотации Java Persistence API (JPA).
37. Библиотека Hibernate, основные аннотации.
38. Объявление сущности и таблицы в Hibernate.
39. Интерфейс Session в Hibernate.
40. Ассоциация сущностей в Hibernate.
41. Spring Boot: определение, характеристики, преимущества.
42. Spring Initializr, особенности и преимущества применения.
43. Структура фреймворка JUnit.
44. JUnit аннотации @Test, @DisplayName.
45. JUnit аннотации @BeforeEach, @AfterEach.
46. Тестовые классы и методы JUnit.
47. Утверждения JUnit. Класс Assert.
48. Тестирование исключений JUnit.
49. Генератор документирования Javadoc. Виды комментариев.

50. Дескрипторы Javadoc.
 51. Java Time API: назначение, основные классы и их методы.
 52. Записи (record) и перечисления (enum) в Java: особенности, области применения.
 53. Аннотации в Java: определение, встроенные и пользовательские аннотации, примеры использования.
 54. Анонимные и локальные классы: назначение, отличие от вложенных классов.
 55. Лямбда-выражения и ссылки на методы: синтаксис, применение.
 56. Обобщения (Generics) в Java: типизация, преимущества, стирание типов, wildcards.
 57. Модули Java: структура модульного приложения, файлы module-info.java и package-info.java.
 58. Sealed-классы: назначение, основные ключевые слова.
 59. Stream API: создание потоков данных, промежуточные и терминальные операции, понятие ленивости.
 60. Буферизованные и небуферизованные потоки: различия, примеры классов.
 61. Транзакции в JDBC и Hibernate: уровни изоляции, методы управления транзакциями.
 62. Принципы SOLID: расшифровка и применение.
 63. Система контроля версий Git: ветвления, слияние, разрешение конфликтов.
 64. Порождающие шаблоны проектирования: назначение, принципы и примеры.
 65. Структурные шаблоны проектирования: цель применения, основные примеры и их роль в построении архитектуры.
 66. Поведенческие шаблоны проектирования: назначение, принципы взаимодействия объектов и ключевые примеры.
 68. Основные примеры антипаттернов: характеристика и способы устранения.
 69. CI/CD: понятие, назначение, использование в процессе разработки.
 70. Методология Agile и фреймворк Scrum: основные роли, артефакты и события.
- *Промежуточная аттестация обучающихся проводится в формате тестирования.*

Примерные темы курсовой работы /курсового проекта

1. Информационно-справочная система туристического агентства
2. Информационно-справочная система музея
3. Информационно-справочная система художественных конкурсов
4. Информационно-справочная система магазина цифровой техники
5. Информационно-справочная система страховой компании
6. Информационно-справочная система кинотеатра
7. Информационно-справочная система театра
8. Информационно-справочная система ЖД вокзала
9. Информационно-справочная система аэропорта
10. Информационно-справочная система ресторана
11. Информационно-справочная система библиотеки
12. Информационно-справочная система обработки и доставки заказов пиццерии

8. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины

Основная литература:

1. Васюткина, И. А. Разработка серверной части web-приложений на Java [Электронный ресурс] : учебное пособие / Васюткина И. А. Новосибирск : НГТУ, 2021 83 с. Утверждено Редакционно-издательским советом университета в качестве учебного пособия Книга из коллекции НГТУ - Информатика <https://e.lanbook.com/book/216155> ISBN 978-5-7782-4394-1. [БИК ID: RU-LAN-BOOK-216155]
2. Гуськова, Ольга Ивановна Объектно ориентированное программирование в Java : Учебное пособие / Московский педагогический государственный университет Москва : Московский педагогический государственный университет, 2018 240 с. ВО - Магистратура <https://znanium.com/catalog/document?id=339668> ISBN 978-5-4263-0648-6. [БИК ID: RU\infra-m\znanium\bibl\1020593]

Дополнительная литература:

1. Чернышев, Станислав Андреевич Принципы, паттерны и методологии разработки программного обеспечения : учебник для вузов / С. А. Чернышев. Электрон. дан. Москва : Юрайт, 2025 176 с (Высшее образование) URL: <https://urait.ru/bcode/567946> (дата обращения: 01.09.2025). Режим доступа: Электронно-библиотечная система Юрайт, для авториз. пользователей <https://urait.ru/bcode/567946> ISBN 978-5-534-14383-6 : 789.00. [БИК ID: RU2fURAIT2f567946]
2. Пруцков, А. В. Язык программирования Java. Введение в курс: операторы и типы данных [Электронный ресурс] : учебное пособие / Пруцков А. В. Рязань : РГРТУ, 2016 72 с. Книга из коллекции РГРТУ - Информатика <https://e.lanbook.com/book/168307>. [БИК ID: RU-LAN-BOOK-168307]

9. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

1. Электронно-библиотечная система BOOK.RU <http://www.book.ru>
2. •Электронная библиотека Финансового университета (ЭБ) <http://elib.fa.ru/>

3. •Электронно-библиотечная система «Университетская библиотека ОНЛАЙН»
<http://biblioclub.ru/>
4. •Электронно-библиотечная система издательства «ЮРАЙТ» <https://urait.ru/>
5. •Электронно-библиотечная система издательства Проспект
<http://ebs.prospekt.org/books>
6. •Электронно-библиотечная система издательства Лань <https://e.lanbook.com/>
7. •Деловая онлайн-библиотека Alpina Digital <http://lib.alpinadigital.ru/>
8. •Научная электронная библиотека eLibrary.ru <http://elibrary.ru>
9. •Электронно-библиотечная система Znanium <http://www.znanium.ru/>

10. Методические указания для обучающихся по освоению дисциплины

Освоение дисциплины складывается из усвоения теоретического материала и приобретения навыков разработки программ. Для изучения теоретического материала рекомендуется использовать тексты лекций и рекомендуемую литературу. Для успешного решения контрольных задач обучаемый должен проанализировать заданную предметную область, выделить объекты и разработать ее информационную модель. Разработка программы должна начинаться с составления логической схемы алгоритма (блок-схемы).

В начале семестра студенты должны самостоятельно с использованием зачетной книжки зарегистрироваться на сайте <https://www.jetbrains.com> и бесплатно получить IntelliJ Idea Community Edition. При переходе к новой теме проводится тестирование, направленное на оценивание теоретических знаний. Помимо тестирования, может проводиться выборочный устный опрос студентов. Полученные оценки участвуют в формировании итоговой оценки по дисциплине.

Практические навыки оцениваются путем разработки прикладных программ в визуальной среде программирования или разбора готовых программ. Студенты должны самостоятельно и вовремя решать поставленные преподавателем задачи.

Студентам при подготовке следует использовать нормативные документы Финансового университета, Методические рекомендации по планированию и организации внеаудиторной самостоятельной работы студентов по образовательным программам бакалавриата и магистратуры в Финансовом университете, утвержденные приказом Финуниверситета от 11.05.2021 г. № 1040/о (см. сайт Финансового Университета: на главной странице раздел "Наш университет" → "Единая правовая база Финуниверситета" → "Организация учебного процесса" → "Нормативные документы по самостоятельной работе").

11. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень необходимого программного обеспечения и информационных справочных систем

Комплект лицензионного программного обеспечения:

1. Комплект свободно распространяемого программного обеспечения
2. Kaspersky

3. Пакет офисных программ

Современные профессиональные базы данных и информационные справочные системы:

1. Размещение ЭУК: <https://www.campus.fa.ru/>
2. Информационно-правовая система «Гарант»
3. Информационно-правовая система «Консультант Плюс»
4. Электронная энциклопедия: <http://ru.wikipedia.org/wiki/Wiki>
5. Система комплексного раскрытия информации «СКРИН» - <http://www.skrin.ru/>

Сертифицированные программные и аппаратные средства защиты информации:

1. не предусмотрены

12. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

1. **Компьютерный класс** для проведения учебных занятий, предусмотренных программой, в том числе групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, оснащенный оборудованием и техническими средствами обучения: мебель аудиторная (столы, стулья, доска аудиторная), персональные компьютеры, набор демонстрационного оборудования (проектор, экран)