ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
СТАВРОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДАЮ

Директор/Декан
факультета цифровых технологий
Аникуев Сергей Викторович
_____
_____
«___» _____ 20___ г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ (ОЦЕНОЧНЫХ МАТЕРИАЛОВ)

Б1.О.18 Прикладное программирование

09.03.02 Информационные системы и технологии

Системы искусственного интеллекта

бакалавр

очная

Ставрополь, 2024

1. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

Процесс изучения дисциплины направлен на формирование следующих компетенций ОП ВО и овладение следующими результатами обучения по дисциплине:

Код и наименование компетенции	Код и наименование индикатора достижения	Перечень планируемых результатов обучения по дисциплине

		знает 1. Теоретические основы алгоритмизации Основные понятия алгоритмов: свойства, способы описания, виды Методы разработки алгоритмов (сверху-вниз, снизу-вверх) Основные структуры данных: массивы, списки, стеки, очереди, деревья, хеш-таблицы Оценку сложности алгоритмов (О-нотация) 2. Принципы программирования Основные парадигмы программирования: императивное, объектно-ориентированное, функциональное Принципы ООП: инкапсуляция, наследование, полиморфизм, абстракция Основные паттерны проектирования (Singleton, Factory, Observer и др.) Принципы SOLID и основы архитектуры ПО 3. Технологии и инструменты разработки Системы контроля версий (Git) Среды разработки (IDE) Системы сборки проектов (Maven, Gradle) Фреймворки для тестирования (JUnit, TestNG) 4. Языки программирования Синтаксис и семантика выбранного языка программирования Стандартные библиотеки и API Механизмы работы с памятью, исключениями, потоками 5. Методологии разработки Основные этапы жизненного цикла ПО Методы отладки и тестирования программ Принципы рефакторинга и оптимизации кода

		умеет 1. Разрабатывать алгоритмы Анализировать предметную область и формулировать техническое задание Проектировать алгоритмы для решения практических задач Выбирать оптимальные структуры данных для конкретной задачи Оценивать эффективность разработанных алгоритмов 2. Применять языки программирования Выбирать язык программирования для решения конкретной задачи Реализовывать разработанные алгоритмы на практике Использовать стандартные библиотеки и фреймворки Применять различные парадигмы программирования 3. Использовать технологии разработки Работать с системами контроля версий Настраивать среды разработки и инструменты сборки Использовать отладчики и профилировщики Применять средства автоматизации тестирования 4. Проектировать архитектуру ПО Выбирать подходящие паттерны проектирования Проектировать модульную структуру приложения Организовывать взаимодействие между компонентами системы Обеспечивать масштабируемость и поддерживаемость кода 5. Анализировать и оптимизировать Проводить код-ревью Выявлять узкие места в производительности Рефакторить код для улучшения качества

		владеет навыками 1. Навыками программирования Навыками написания чистого, читаемого и поддерживаемого кода Техниками отладки и поиска ошибок в программах Навыками работы с различными типами данных и структурами Методами обработки исключительных ситуаций 2. Технологическими навыками Навыками работы с Git: ветвление, слияние, разрешение конфликтов Владением интегрированными средами разработки (IntelliJ IDEA, Eclipse, VS Code) Навыками использования систем сборки и управления зависимостями Техниками модульного и интеграционного тестирования 3. Навыками разработки алгоритмов Методами решения типовых алгоритмических задач Навыками анализа и оптимизации алгоритмов Техниками работы с различными структурами данных Методами разработки эффективных вычислительных процедур 4. Проектными навыками Навыками проектирования архитектуры приложений Методами разработки технической документации Навыками командной работы над программными проектами Техниками оценки трудоемкости и планирования разработки 5. Практическими навыками применения Навыками создания консольных приложений Техниками разработки приложений с графическим

		знает 1. Методы алгоритмизации и анализа Методы построения и анализа алгоритмов («разделяй и властвуй», динамическое программирование, жадные алгоритмы) Критерии оценки эффективности алгоритмов (временная и пространственная сложность) Методы формализации и спецификации профессиональных задач Принципы построения вычислительных процессов 2. Профессиональные языки программирования Синтаксис и семантика языков программирования, востребованных в профессиональной деятельности Объектно-ориентированные и функциональные парадигмы программирования Принципы работы с памятью, процессами, потоками выполнения Механизмы обработки исключений и ошибок 3. Технологии и инструменты разработки Современные фреймворки и библиотеки для разработки информационных систем Технологии веб-разработки (REST API, микросервисы, контейнеризация) Средства автоматизации сборки, развертывания и тестирования Инструменты мониторинга и профилирования приложений 4. Профессиональные стандарты и практики Стандарты кодирования и оформления программного кода Паттерны проектирования архитектуры информационных систем Методологии разработки (Agile, Scrum, Kanban) Принципы обеспечения качества программного обеспечения 5. У

		умеет 1. Анализировать профессиональные задачи Формализовать требования к программному решению Выявлять ограничения и критерии эффективности Декомпозировать сложные задачи на составляющие Оценивать целесообразность применения различных подходов 2. Выбирать и применять методы алгоритмизации Подбирать оптимальные алгоритмы для решения профессиональных задач Адаптировать известные алгоритмы к специфике предметной области Разрабатывать собственные алгоритмические решения Проводить сравнительный анализ алгоритмов 3. Использовать языки и технологии программирования Выбирать язык программирования в зависимости от типа задачи Применять различные технологии программирования комплексно Интегрировать готовые программные компоненты и библиотеки Использовать специализированные инструменты разработки 4. Проектировать программные решения Разрабатывать архитектуру программных систем Проектировать базы данных и схемы хранения информации Создавать интерфейсы взаимодействия между компонентами системы Обеспечивать масштабируемость и отказоустойчивость решений 5. Оценивать качество и эффективность Проводить тестирование программных решений

		владеет навыками 1. Навыками алгоритмического мышления Навыками формализации профессиональных задач Техниками разработки эффективных алгоритмов Методами анализа и оптимизации вычислительных процессов Навыками работы с различными структурами данных 2. Технологиями программирования Практическими навыками программирования на выбранных языках Техниками использования фреймворков и библиотек Навыками работы с системами контроля версий Методами отладки и профилирования кода 3. Инструментами разработки Навыками работы с интегрированными средами разработки Техниками использования систем сборки и развертывания Навыками автоматизации процессов разработки Методами контейнеризации и оркестрации приложений 4. Проектными компетенциями Навыками проектирования архитектуры информационных систем Техниками разработки технической документации Навыками командной работы над программными проектами Методами управления жизненным циклом программного обеспечения 5. Профессиональными практиками Навыками применения стандартов и паттернов проектирования Техниками обеспечения безопасности программных решений

		знает 1. Методы программирования прототипов Принципы прототипной разработки (rapid prototyping) Методы инкрементальной и итеративной разработки Особенности программирования в условиях неполных требований Техники создания минимально работоспособного продукта (MVP) 2. Технологии отладки программного обеспечения Классификацию ошибок (синтаксические, логические, runtime) Методы локализации ошибок в коде Инструменты отладки (дебаггеры, профайлеры, трассировщики) Техники анализа дампов памяти и логов выполнения 3. Методологии тестирования прототипов Уровни тестирования (модульное, интеграционное, системное) Виды тестирования (функциональное, нагрузочное, регрессионное) Принципы тест-дизайна и создания тестовых сценариев Методы автоматизации тестирования 4. Архитектуру программно-технических комплексов Компонентный состав программно-технических комплексов Принципы взаимодействия программных и аппаратных компонентов Протоколы обмена данными в распределенных системах Методы обеспечения надежности и отказоустойчивости 5. Инструментальные средства разработки Среды разработки с функциями отладки (Visual Studio, IntelliJ IDEA, Eclipse) Ай[illegible]

		умеет 1. Разрабатывать прототипы программных комплексов Создавать архитектурные решения для прототипов Выбирать технологии и инструменты для быстрой разработки Реализовывать ключевые функции программно-технических комплексов Интегрировать программные компоненты с аппаратными средствами 2. Проводить отладку программного кода Воспроизводить и локализовать ошибки в программах Использовать отладчики для пошагового выполнения кода Анализировать стек вызовов и состояние переменных Применять методы логирования для диагностики проблем 3. Организовывать тестирование прототипов Разрабатывать тестовые сценарии и тест-кейсы Выполнять различные виды тестирования программных комплексов Автоматизировать процессы тестирования Анализировать результаты тестирования и составлять отчеты 4. Анализировать требования к программно-техническим комплексам Выявлять и формализовать требования к прототипам Оценивать техническую реализуемость требований Определять критерии приемки прототипов Планировать итерации разработки на основе обратной связи 5. Оптимизировать процессы разработки Выбирать оптимальные методы программирования для конкретных задач Настраивать инструменты отладки и тестирования под

		владеет навыками 1. Навыками прототипной разработки Техниками быстрого создания работающих прототипов Навыками итеративного проектирования архитектуры Методами оценки технической осуществимости решений Практикой адаптации прототипов под изменяющиеся требования 2. Техниками отладки сложных систем Навыками использования современных отладочных инструментов Методами диагностики проблем в распределенных системах Техниками анализа производительности и оптимизации кода Практикой отладки взаимодействия программных и аппаратных компонентов 3. Методами комплексного тестирования Навыками разработки тестового покрытия для прототипов Техниками создания автоматизированных тестовых сценариев Методами нагрузочного и стресс-тестирования комплексов Практикой тестирования интеграции разнородных компонентов 4. Инструментарием разработки и диагностики Навыками работы с интегрированными средами разработки Техниками использования систем контроля версий Методами настройки и использования CI/CD систем Практикой применения инструментов мониторинга и логирования 5. Компетенциями документирования и презентации Навыками создания технической документации на прототипы

2. Перечень оценочных средств по дисциплине

№	Наименование раздела/темы	Семестр	Код индикаторов достижения компетенций	Оценочное средство проверки результатов достижения индикаторов компетенций
1.	1 раздел. Прикладное программирование			
1.1.	Введение в Java и инструменты разработчика	4	ОПК-6.1, ОПК-6.2, ОПК-6.3	
1.2.	Углубленное ООП и основные механизмы Java	4	ОПК-6.1, ОПК-6.2, ОПК-6.3	
1.3.	Кт 1	4	ОПК-6.1, ОПК-6.2, ОПК-6.3	Тест
1.4.	Работа с данными и продвинутые возможности	4	ОПК-6.1, ОПК-6.2, ОПК-6.3	
1.5.	Создание приложений и интеграция	4	ОПК-6.1, ОПК-6.2, ОПК-6.3	
1.6.	Кт 2	4	ОПК-6.1, ОПК-6.2, ОПК-6.3	Тест
1.7.	Завершающий проект	4	ОПК-6.1, ОПК-6.2, ОПК-6.3	
	Промежуточная аттестация			За

3. Оценочные средства (оценочные материалы)

Примерный перечень оценочных средств для текущего контроля успеваемости и промежуточной аттестации

№ п/п	Наименование оценочного средства	Краткая характеристика оценочного средства	Представление оценочного средства в фонде (Оценочные материалы)
Текущий контроль			
	Для оценки знаний		
1	Тест	Система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающегося.	Фонд тестовых заданий
	Для оценки умений		
	Для оценки навыков		
	Промежуточная аттестация		

2	Зачет	Средство контроля усвоения учебного материала практических и семинарских занятий, успешного прохождения практик и выполнения в процессе этих практик всех учебных поручений в соответствии с утвержденной программой с выставлением оценки в виде «зачтено», «незачтено».	Перечень вопросов к зачету

4. Примерный фонд оценочных средств для проведения текущего контроля и промежуточной аттестации обучающихся по дисциплине (модулю) "Прикладное программирование"

Примерные оценочные материалы для текущего контроля успеваемости

ОПК-6: Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий

Тестовые задания по дисциплине "Прикладное программирование"

Знания

Задание 1

Что такое инкапсуляция в ООП?

a) Процесс создания новых классов на основе существующих.

b) Механизм, позволяющий представлять один и тот же интерфейс для разных типов данных.

c) Сокрытие внутреннего состояния объекта и необходимость взаимодействия с ним через публичные методы.

d) Способность объекта принимать множество форм.

Ответ: c

Задание 2

Какая структура данных работает по принципу "первым пришел — первым ушел" (FIFO)?

a) Стек (Stack)

b) Очередь (Queue)

c) Массив (Array)

d) Связный список (Linked List)

Ответ: b

Задание 3

Какой алгоритм сортировки в худшем случае имеет сложность O(n log n) и использует стратегию "разделяй и властвуй"?

a) Сортировка пузырьком (Bubble Sort)

b) Быстрая сортировка (Quick Sort)

c) Сортировка слиянием (Merge Sort)

d) Сортировка вставками (Insertion Sort)

Ответ: c

Задание 4

Какая из этих HTTP-методов идемпотентна? (Выберите все подходящие варианты)

a) GET

b) POST

c) PUT

d) DELETE

Ответ: a,c,d

Задание 5

Что означает аббревиатура SOLID в контексте проектирования программного обеспечения?

a) Набор принципов объектно-ориентированного программирования и дизайна.

b) Название популярного фреймворка.

c) Протокол для безопасного обмена данными.

d) Стандарт кодирования.

Ответ: a

Задание 6

Что такое "потокобезопасность" (thread safety)?

a) Гарантия того, что функция может быть выполнена только одним потоком в единицу времени.

b) Свойство кода, позволяющее ему корректно работать в многопоточной среде, предотвращая состояния гонки (race conditions).

c) Способность программы создавать неограниченное количество потоков.

d) Механизм для ускорения работы программы за счет параллелизма.

Ответ: b

Задание 7

Для чего используется система контроля версий, такая как Git?

a) Для компиляции исходного кода.

b) Для отслеживания изменений в исходном коде и совместной работы над ним.

c) Для автоматического тестирования приложений.

d) Для развертывания приложения на сервере.

Ответ: b

Задание 8

Что такое "внедрение зависимостей" (Dependency Injection)?

a) Паттерн, при котором класс создает все свои зависимости самостоятельно.

b) Паттерн, при котором зависимости объекта предоставляются ему извне (например, через конструктор).

c) Процесс исправления ошибок в коде.

d) Метод оптимизации запросов к базе данных.

Ответ: b

Задание 9

Какая из этих пар "ключ-значение" является корректным представлением данных в формате JSON?

a) { name: "Alice", age: 30 }

b) { "name": "Alice", "age": 30 }

c) ("name": "Alice", "age": 30)

d) ["name": "Alice", "age": 30]

Ответ: b

Задание 10

Какое утверждение о модульном тестировании (Unit Testing) является верным?

a) Оно проверяет интеграцию всех модулей системы вместе.

b) Оно проверяет отдельные, изолированные части программы (например, функции или методы).

c) Оно выполняется только вручную.

d) Оно предназначено для проверки производительности системы.

Ответ: b

Задание 11

Что такое "жадный" алгоритм (Greedy Algorithm)?

a) Алгоритм, который на каждом шаге выбирает наилучший вариант в надежде найти глобальный оптимум.

b) Алгоритм, который всегда находит абсолютно наилучшее решение.

c) Алгоритм, который требует очень много памяти.

d) Алгоритм, который используется только для сортировки.

Ответ: a

Задание 12

Какая команда Git используется для клонирования удаленного репозитория?

a) git pull

b) git commit

c) git clone

d) git fork

Ответ: c

Задание 13

Что такое API (Application Programming Interface)?

a) Графический интерфейс пользователя.

b) Набор определений, протоколов и инструментов для взаимодействия между различными программными компонентами.

c) Интегрированная среда разработки.

d) Язык программирования для создания веб-приложений.

Ответ: b

Задание 14

Какая из этих структур данных обычно реализуется с использованием хеш-таблицы?

a) Стек (Stack)

b) Очередь (Queue)

c) Ассоциативный массив / Словарь (Dictionary / Map)

d) Дерево (Tree)

Ответ: c

Задание 15

Какое из этих понятий НЕ относится к реляционным базам данных?

a) Таблица

b) Документ

c) Первичный ключ (Primary Key)

d) Внешний ключ (Foreign Key)

Ответ: b

Задание 16

Что такое "исключение" (exception) в программировании?

a) Специальный тип данных для хранения чисел.

b) Событие, которое возникает во время выполнения программы и нарушает нормальный ход ее инструкций.

c) Комментарий в коде.

d) Способ объявления переменной.

Ответ: b

Задание 17

Какой паттерн проектирования используется для создания семейств связанных объектов без указания их конкретных классов?

a) Одиночка (Singleton)

b) Фабричный метод (Factory Method)

c) Абстрактная фабрика (Abstract Factory)

d) Наблюдатель (Observer)

Ответ: c

Задание 18

Какой оператор используется в SQL для извлечения данных из таблицы?

a) GET

b) SELECT

c) FETCH

d) EXTRACT

Ответ: b

Задание 19

Что такое "кеширование"?

a) Процесс удаления неиспользуемых данных.

b) Механизм временного хранения часто используемых данных для ускорения доступа к ним в будущем.

c) Шифрование конфиденциальной информации.

d) Тип структуры данных "очередь".

Ответ: b

Задание 20

Какая из этих характеристик является преимуществом микросервисной архитектуры над

монолитной?

a) Более простая отладка.
b) Меньшая сложность развертывания.
c) Лучшая масштабируемость и независимость сервисов.
d) Более высокая производительность каждого отдельного компонента.
Ответ:c
Умения
Задание 1
Установите соответствие между концепциями ООП и их описаниями:
Концепция
1. Инкапсуляция
2. Наследование
3. Полиморфизм
4. Абстракция
Описание
А. Наследование характеристик от родительского класса
Б. Сокрытие внутренней реализации и доступ только через методы
В. Способность объекта принимать разные формы
Г. Выделение существенных характеристик объекта
Ответ: 1-Б, 2-А, 3-В, 4-Г
Задание 2
Установите соответствие между HTTP-методами и их назначением:
HTTP-метод
1. GET
2. POST
3. PUT
4. DELETE
Назначение
А. Создание нового ресурса
Б. Полное обновление ресурса
В. Получение ресурса
Г. Удаление ресурса
Ответ: 1-В, 2-А, 3-Б, 4-Г
Задание 3
Установите соответствие между паттернами проектирования и их описаниями:
Паттерн
1. Singleton
2. Factory Method
3. Observer
4. Decorator
Описание
А. Определяет зависимость "один-ко-многим" между объектами
Б. Гарантирует, что класс имеет только один экземпляр
В. Создает объекты без указания точного класса
Г. Динамически добавляет новую функциональность объекту
Ответ: 1-Б, 2-В, 3-А, 4-Г
Задание 4
Установите соответствие между типами тестирования и их определениями:
Тип тестирования
1. Unit-тесты
2. Интеграционные
3. Системные
4. Приемочные
Определение
А. Проверка взаимодействия между компонентами
Б. Проверка отдельного модуля или функции

В. Имитация действий реального пользователя
Г. Проверка системы в целом
Ответ: 1-Б, 2-А, 3-Г, 4-В
Задание 5
Установите соответствие между принципами SOLID и их описаниями:
Принцип
1. SRP
2. OCP
3. LSP
4. DIP
Описание
А. Классы должны быть открыты для расширения, но закрыты для изменения
Б. Объекты должны зависеть от абстракций, а не от конкретных классов
В. Класс должен иметь только одну причину для изменения
Г. Подтипы должны быть заменяемы для базовых типов
Ответ: 1-В, 2-А, 3-Г, 4-Б

Задание 6
Установите соответствие между командами Git и их назначением:
Команда Git
1. git clone
2. git commit
3. git push
4. git pull
Назначение
А. Сохранение изменений в локальном репозитории
Б. Получение изменений из удаленного репозитория
В. Копирование удаленного репозитория
Г. Отправка изменений в удаленный репозиторий
Ответ: 1-В, 2-А, 3-Г, 4-Б
Задание 7
Расставьте в правильном порядке этапы компиляции программы:
А. Компиляция в ассемблерный код
Б. Лексический анализ
В. Генерация объектного кода
Г. Синтаксический анализ
Д. Оптимизация кода
Правильный порядок: Б → Г → А → Д → В
Задание 8
. Расставьте в правильном порядке процесс работы с базой данных в приложении:
А. Выполнение SQL-запроса
Б. Закрытие соединения
В. Установление соединения с БД
Г. Обработка результатов
Д. Создание запроса
Правильный порядок: В → Д → А → Г → Б
Задание 9
Расставьте в правильном порядке жизненный цикл HTTP-запроса:
А. Обработка запроса сервером
Б. Установление TCP-соединения
В. Отправка HTTP-ответа
Г. Формирование HTTP-запроса клиентом
Д. Разрыв соединения
Правильный порядок: Б → Г → А → В → Д
Задание 10

Расставьте в правильном порядке процесс работы сборщика мусора (Garbage Collector):
А. Помечение используемых объектов
Б. Удаление непомеченных объектов
В. Определение корневых объектов
Г. Компактизация памяти
Правильный порядок: В → А → Б → Г
Задание 11
Расставьте в правильном порядке этапы разработки по методологии Agile:
А. Спринт (разработка)
Б. Сбор требований
В. Демонстрация результата
Г. Планирование спринта
Д. Ретроспектива
Правильный порядок: Б → Г → А → В → Д
Задание 12
Расставьте в правильном порядке уровни модели OSI (снизу вверх):
А. Транспортный
Б. Физический
В. Прикладной
Г. Сетевой
Д. Канальный
Правильный порядок: Б → Д → Г → А → В
Задание 13
Расставьте в правильном порядке процесс обработки исключения:
А. Выполнение блока finally
Б. Поиск подходящего обработчика catch
В. Генерация исключения
Г. Завершение работы обработчика
Правильный порядок: В → Б → Г → А
Задание 14
Расставьте в правильном порядке этапы работы MVC-паттерна:
А. Обновление Model
Б. Запрос от пользователя
В. Обновление View
Г. Обработка в Controller
Правильный порядок: Б → Г → А → В
Задание 15
Расставьте в правильном порядке процесс JIT-компиляции:
А. Интерпретация байт-кода
Б. Компиляция "горячего" кода в машинный
В. Профилирование выполнения
Г. Выполнение скомпилированного кода
Правильный порядок: А → В → Б → Г
Задание 16
Расставьте в правильном порядке этапы работы с транзакцией в БД:
А. COMMIT или ROLLBACK
Б. Начало транзакции (BEGIN)
В. Выполнение SQL-операций
Г. Проверка на конфликты
Правильный порядок: Б → В → Г → А
Задание 17
Расставьте в правильном порядке процесс dependency injection:
А. Создание зависимостей
Б. Регистрация сервисов в контейнере
В. Внедрение зависимостей через конструктор
Г. Разрешение зависимостей контейнером

Правильный порядок: А → Б → Г → В

Задание 18

Расставьте в правильном порядке этапы код-ревью:

А. Исправление замечаний

Б. Отправка pull request

В. Проверка кода ревьювером

Г. Написание кода

Д. Мердж в основную ветку

Правильный порядок: Г → Б → В → А → Д

Навыки

Задание 1

Верно ли утверждение: В Java все объекты передаются по ссылке в методы.

Ответ: Неверно

Задание 2

Верно ли утверждение: SQL-инъекция возможна даже при использовании подготовленных выражений (prepared statements).

Ответ: Неверно

Задание 3

Верно ли утверждение: Алгоритм быстрой сортировки (QuickSort) всегда имеет сложность O(n log n).

Ответ: Неверно

Задание 4

Верно ли утверждение: HTTP - это протокол без состояния (stateless).

Ответ: Верно

Задание 5

Верно ли утверждение: Паттерн Singleton гарантирует, что в многопоточной среде всегда будет создан только один экземпляр класса без дополнительной синхронизации.

Ответ: Неверно

Задание 6

Верно ли утверждение: В реляционных базах данных внешний ключ (foreign key) должен всегда ссылаться на первичный ключ (primary key) другой таблицы.

Ответ: Верно

Задание 7

Верно ли утверждение: Garbage Collector в Java может привести к утечкам памяти, если неправильно используются ссылки.

Ответ: Верно

Задание 8

Верно ли утверждение: REST API требует использования только XML для передачи данных.

Ответ: Неверно

Задание 9

Верно ли утверждение: Модификатор final для переменной в Java означает, что ее значение нельзя изменить после инициализации.

Ответ: Верно

Задание 10

Верно ли утверждение: Индекс в базе данных всегда ускоряет операции SELECT, но замедляет операции INSERT и UPDATE.

Ответ: Верно

Задание 11

Как называется ошибка, когда программа пытается обратиться к памяти, которая ей не принадлежит?

Ответ: Сегментация

Задание 12

Сколько бит занимает тип данных int в Java?

Ответ: 32

Задание 13

Какой принцип SOLID нарушает класс, который имеет несколько причин для изменения?

Ответ: SRP

Задание 14

Как называется структура данных, в которой элементы хранятся в порядке LIFO?

Ответ: Стек

Задание 15

Какой HTTP-статус означает "Не найдено"?

Ответ: 404

Задание 16

Как называется процесс преобразования сложной структуры данных в последовательность байт для передачи или хранения?

Ответ: Сериализация

Задание 17

Сколько основных цветов используется в веб-дизайне (HEX-формат)?

Ответ: 16

Задание 18

Какой алгоритм использует протокол HTTPS для шифрования данных?

Ответ: TLS

Задание 19

Как называется ситуация, когда два или более потока пытаются получить доступ к общему ресурсу и изменить его одновременно?

Ответ: Состояние гонки

Задание 20

Какой символ используется в регулярных выражениях для обозначения "любой символ"?

Ответ: . (точка)

Примерные оценочные материалы
для проведения промежуточной аттестации (зачет, экзамен)
по итогам освоения дисциплины (модуля)

I. Теоретические вопросы (знания)

Проверяют понимание фундаментальных концепций, принципов и механизмов языка.

Принципы объектно-ориентированного программирования (ООП). Дайте определение инкапсуляции, наследования и полиморфизма.

Жизненный цикл объекта в Java. Что такое конструктор? Порядок вызова конструкторов при наследовании.

Модификаторы доступа в Java (private, default, protected, public). Области их видимости.

Отличие абстрактного класса от интерфейса. В каких случаях следует использовать каждый из них? (С учетом появления default-методов в интерфейсах).

Иерархия исключений в Java. В чем разница между проверяемыми (checked) и непроверяемыми (unchecked) исключениями?

Понятие коллекций в Java. Опишите основные интерфейсы List, Set, Map и их реализации (ArrayList, HashSet, HashMap).

Принципы работы сборщика мусора в Java. Что такое поколения объектов (Young Generation, Old Generation)?

Понятие обобщений (Generics) в Java. Для чего они предназначены и какие проблемы решают?

Многопоточность: чем отличается Runnable от Thread? Что такое монитор и синхронизация?

Основы JDBC. Опишите основные шаги для подключения к базе данных и выполнения SQL-запроса.

II. Вопросы на умения (уметь)

Проверяют способность применять теоретические знания для анализа, проектирования и

решения практических задач.

Проанализируйте предложенный код и найдите в нем ошибки, связанные с нарушением принципов ООП или базового синтаксиса.

Спроектируйте иерархию классов для заданной предметной области (например, "Транспортные средства", "Библиотека").

Объясните, какой класс коллекций (List, Set, Map и их реализации) следует выбрать для решения конкретной задачи (например, хранение уникальных элементов, быстрый поиск по ключу).

Составьте алгоритм для чтения данных из текстового файла, обработки (например, фильтрации) и записи результата в другой файл.

Разработайте структуру классов для взаимодействия с базой данных, используя паттерн DAO (Data Access Object).

Проанализируйте код, содержащий потенциальные проблемы многопоточности (состояние гонки, deadlock), и предложите способы их устранения.

Объясните, как использовать механизм обработки исключений для обеспечения надежности приложения при работе с внешними ресурсами (файлы, сеть, БД).

Составьте план модульного тестирования для заданного класса, определив, какие методы и сценарии нужно покрыть тестами.

Расшифруйте и объясните назначение основных секций в файле pom.xml для системы сборки Maven.

Сравните два подхода к созданию потоков (наследование от Thread и реализация интерфейса Runnable) и аргументируйте выбор одного из них.

III. Вопросы на навыки (владеть)

Проверяют сформированность практических навыков, готовность к выполнению конкретных действий по разработке.

Навык написания кода: Напишите простой класс, реализующий заданную функциональность, с соблюдением принципов инкапсуляции (использование private полей, геттеров/сеттеров).

Навык работы с коллекциями: Продемонстрируйте на примере, как добавить, удалить и найти элемент в ArrayList, HashSet и HashMap.

Навык обработки исключений: Напишите фрагмент кода с использованием try-catch-finally для безопасного открытия и закрытия файлового потока.

Навык работы с Git: Опишите последовательность команд Git для создания новой ветки, коммита изменений и отправки их в удаленный репозиторий.

Навык создания GUI: Создайте простейшее окно приложения с помощью JavaFX (кнопка и текстовое поле) и реализуйте обработчик события нажатия на кнопку.

Навык работы со строками: Напишите код, демонстрирующий разницу в использовании классов String и StringBuilder (например, для многократной конкатенации строк).

Навык написания SQL-запросов и использования JDBC: Напишите код, который выполняет SQL SELECT-запрос к базе данных через PreparedStatement и обрабатывает ResultSet.

Навык отладки: Продемонстрируйте, как использовать отладчик в IDE (IntelliJ IDEA) для пошагового выполнения кода и просмотра значений переменных.

Навык написания unit-тестов: Напишите тест с использованием JUnit 5, который проверяет корректность работы метода, используя аннотации @Test и класс Assertions.

Навык сборки проекта: Создайте исполняемый JAR-файл для простого консольного приложения с помощью Maven или Gradle.

Темы письменных работ (эссе, рефераты, курсовые работы и др.)

Темы прикладных заданий:

Категория 1: Мониторинг и управление сельхозтехникой

1. Система мониторинга топлива сельхозтехники

Цель: Разработать систему учета расхода топлива тракторов и комбайнов.

Функционал:

Ввод данных о заправках и рабочих часах

Расчет среднего расхода топлива на единицу площади

Анализ эффективности использования техники

Генерация отчетов по технике и водителям

Технологии: Java SE, JDBC, MySQL/PostgreSQL, JFreeChart для графиков

2. Планировщик сервисного обслуживания техники
Цель: Автоматизация учета ТО сельхозмашин.
Функционал:
Учет наработки моточасов
Автоматическое напоминание о необходимости ТО
История ремонтов и затрат
Прогноз стоимости обслуживания
Технологии: Java SE, Spring Boot, Hibernate, Email уведомления

Категория 2: Управление растениеводством
3. Система планирования севооборота
Цель: Оптимизация чередования культур на полях.
Функционал:
Учет истории полей и выращиваемых культур
Рекомендации по оптимальному севообороту
Расчет потребности в удобрениях
Визуализация плана на сезон
Технологии: Java FX/Swing, Graphviz для схем, JSON/XML для хранения данных

4. Калькулятор норм внесения удобрений
Цель: Расчет оптимальных доз удобрений для разных культур.
Функционал:
База данных культур и их потребностей
Учет почвенного анализа
Расчет экономической эффективности
Формирование карт внесения
Технологии: Java SE, Apache POI для работы с Excel, база данных

Категория 3: Складской учет и логистика
5. Система складского учета зерна

Цель: Учет поступления, хранения и отгрузки зерна.
Функционал:
Приемка зерна с полей (влажность, сорность)
Контроль условий хранения
Учет отгрузок покупателям
Остатки на складах в реальном времени
Технологии: Java EE, Spring MVC, Hibernate, REST API

6. Оптимизатор логистики перевозок
Цель: Минимизация затрат на перевозку урожая.
Функционал:
Расчет оптимальных маршрутов
Учет грузоподъемности транспорта
Планирование графика перевозок
Контроль выполнения перевозок
Технологии: Java, алгоритмы оптимизации, Google Maps API

Категория 4: Метеоданные и прогнозирование
7. Агрометеостанция с прогнозом заболеваний растений
Цель: Прогноз развития болезней на основе погодных условий.
Функционал:
Интеграция с данными метеостанций
Модели развития основных заболеваний
Сигнализация о рисках

Рекомендации по обработкам
Технологии: Java, Spring Boot, REST клиенты, WebSocket для уведомлений

8. Система расчета оптимальных сроков посева
Цель: Определение лучших дат посева на основе погодных прогнозов.
Функционал:
Анализ температурного режима
Учет влажности почвы
Прогноз всхожести
Визуализация рекомендаций
Технологии: Java, JFreeChart, работа с внешними API погоды

Категория 5: Экономика и аналитика
9. Система калькуляции себестоимости продукции
Цель: Расчет себестоимости сельхозпродукции.
Функционал:
Учет всех затрат (семена, СЗР, ГСМ, зарплата)
Распределение затрат по культурам
Анализ рентабельности
Сравнительная аналитика по сезонам
Технологии: Java EE, Spring Framework, JasperReports для отчетов

10. Платформа анализа эффективности хозяйства
Цель: Комплексный анализ показателей агропредприятия.
Функционал:
KPI (урожайность, затраты, прибыль)
Бенчмаркинг с аналогичными хозяйствами
Выявление узких мест
Прогнозные модели
Технологии: Java, Spring Boot, Angular/React frontend, Big Data инструменты

Категория 6: Мобильные решения
11. Мобильное приложение учета полевых работ
Цель: Фиксация выполненных работ в полевых условиях.
Функционал:
Ввод данных о выполненных операциях
Привязка к координатам поля
Фотофиксация проблем
Оффлайн-работа с синхронизацией
Технологии: Java (Android), SQLite, Google Maps API, REST клиент

12. Система идентификации вредителей по фото
Цель: Определение вредителей по фотографии с мобильного устройства.
Функционал:
Загрузка фото вредителя
Сравнение с базой данных
Рекомендации по мерам борьбы
Геолокация очагов заражения
Технологии: Java (Android), TensorFlow Lite, компьютерное зрение

Категория 7: Интеграционные платформы
13. Единая платформа управления агробизнесом
Цель: Интеграция всех процессов хозяйства в одной системе.
Функционал:
Модульная архитектура
Управление растениеводством, животноводством, складом, финансами

Единая отчетность
Интеграция с госсистемами
Технологии: Java EE, Microservices, Docker, Kubernetes, REST API

14. API для интеграции с сельхозтехникой
Цель: Унифицированный интерфейс для работы с данными от сельхозмашин.
Функционал:
Прием данных в форматах ISOBUS
Конвертация в единый формат
Хранение и обработка данных
Предоставление данных другим системам
Технологии: Java, Spring Integration, MQTT, WebSocket, XML/JSON