

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
СТАВРОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ**

УТВЕРЖДАЮ

Директор/Декан
факультета цифровых технологий
Аникуев Сергей Викторович

«__» _____ 20__ г.

Рабочая программа дисциплины

Б1.О.18 Прикладное программирование

09.03.02 Информационные системы и технологии

Системы искусственного интеллекта

бакалавр

очная

1. Цель дисциплины

Целью освоение дисциплины является подготовка студентов к самостоятельной разработке полнофункциональных прикладных приложений на языке Java. Дисциплина нацелена на получение опыта сквозной разработки: от анализа требований и проектирования архитектуры до реализации, тестирования и сборки готового программного обеспечения.

2. 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 компетенции	Перечень планируемых результатов обучения по дисциплине
ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий	ОПК-6.1 Использует методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий	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

		<p>Синтаксис и семантика выбранного языка программирования</p> <p>Стандартные библиотеки и API</p> <p>Механизмы работы с памятью, исключениями, потоками</p> <p>5. Методологии разработки</p> <p>Основные этапы жизненного цикла ПО</p> <p>Методы отладки и тестирования программ</p> <p>Принципы рефакторинга и оптимизации кода</p> <p>умеет</p> <p>1. Разрабатывать алгоритмы</p> <p>Анализировать предметную область и формулировать техническое задание</p> <p>Проектировать алгоритмы для решения практических задач</p> <p>Выбирать оптимальные структуры данных для конкретной задачи</p> <p>Оценивать эффективность разработанных алгоритмов</p> <p>2. Применять языки программирования</p> <p>Выбирать язык программирования для решения конкретной задачи</p> <p>Реализовывать разработанные алгоритмы на практике</p> <p>Использовать стандартные библиотеки и фреймворки</p> <p>Применять различные парадигмы программирования</p> <p>3. Использовать технологии разработки</p> <p>Работать с системами контроля версий</p> <p>Настраивать среды разработки и инструменты сборки</p> <p>Использовать отладчики и профилировщики</p> <p>Применять средства автоматизации тестирования</p>
--	--	--

		<p>4. Проектировать архитектуру ПО</p> <p>Выбирать подходящие паттерны проектирования</p> <p>Проектировать модульную структуру приложения</p> <p>Организовывать взаимодействие между компонентами системы</p> <p>Обеспечивать масштабируемость и поддерживаемость кода</p> <p>5. Анализировать и оптимизировать</p> <p>Проводить код-ревью</p> <p>Выявлять узкие места в производительности</p> <p>Рефакторить код для улучшения качества</p> <p>Документировать разработанные решения</p> <p>владеет навыками</p> <p>1. Навыками программирования</p> <p>Навыками написания чистого, читаемого и поддерживаемого кода</p> <p>Техниками отладки и поиска ошибок в программах</p> <p>Навыками работы с различными типами данных и структурами</p> <p>Методами обработки исключительных ситуаций</p> <p>2. Технологическими навыками</p> <p>Навыками работы с Git: ветвление, слияние, разрешение конфликтов</p> <p>Владением интегрированными средами разработки (IntelliJ IDEA, Eclipse, VS Code)</p> <p>Навыками использования систем сборки и управления зависимостями</p> <p>Техниками модульного и интеграционного тестирования</p> <p>3. Навыками разработки алгоритмов</p> <p>Методами решения типовых алгоритмических задач</p>
--	--	---

		<p>Навыками анализа и оптимизации алгоритмов</p> <p>Техниками работы с различными структурами данных</p> <p>Методами разработки эффективных вычислительных процедур</p> <p>4. Проектными навыками</p> <p>Навыками проектирования архитектуры приложений</p> <p>Методами разработки технической документации</p> <p>Навыками командной работы над программными проектами</p> <p>Техниками оценки трудоемкости и планирования разработки</p> <p>5. Практическими навыками применения</p> <p>Навыками создания консольных приложений</p> <p>Техниками разработки приложений с графическим интерфейсом</p> <p>Навыками работы с базами данных из программного кода</p> <p>Методами разработки веб-приложений и REST API</p>
ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий	ОПК-6.2 Применяет методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий	<p>знает</p> <p>1. Методы алгоритмизации и анализа</p> <p>Методы построения и анализа алгоритмов («разделяй и властвуй», динамическое программирование, жадные алгоритмы)</p> <p>Критерии оценки эффективности алгоритмов (временная и пространственная сложность)</p> <p>Методы формализации и спецификации профессиональных задач</p> <p>Принципы построения вычислительных процессов</p> <p>2. Профессиональные языки программирования</p> <p>Синтаксис и семантика языков программирования, востребованных в</p>

		<p>профессиональной деятельности</p> <p>Объектно-ориентированные и функциональные парадигмы программирования</p> <p>Принципы работы с памятью, процессами, потоками выполнения</p> <p>Механизмы обработки исключений и ошибок</p> <p>3. Технологии и инструменты разработки</p> <p>Современные фреймворки и библиотеки для разработки информационных систем</p> <p>Технологии веб-разработки (REST API, микросервисы, контейнеризация)</p> <p>Средства автоматизации сборки, развертывания и тестирования</p> <p>Инструменты мониторинга и профилирования приложений</p> <p>4. Профессиональные стандарты и практики</p> <p>Стандарты кодирования и оформления программного кода</p> <p>Паттерны проектирования архитектуры информационных систем</p> <p>Методологии разработки (Agile, Scrum, Kanban)</p> <p>Принципы обеспечения качества программного обеспечения</p> <p>5. Интеграционные технологии</p> <p>Протоколы взаимодействия компонентов информационных систем</p> <p>Технологии работы с распределенными системами</p> <p>Принципы интеграции с внешними системами и сервисами</p> <p>Методы обеспечения безопасности программных решений</p> <p>умеет</p> <p>1. Анализировать профессиональные задачи</p> <p>Формализовать требования к программному решению</p>
--	--	---

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

		<p>Анализировать производительность и оптимизировать код</p> <p>Оценивать соответствие результатов поставленным задачам</p> <p>Документировать процесс и результаты разработки</p> <p>владеет навыками</p> <p>1. Навыками алгоритмического мышления</p> <p>Навыками формализации профессиональных задач</p> <p>Техниками разработки эффективных алгоритмов</p> <p>Методами анализа и оптимизации вычислительных процессов</p> <p>Навыками работы с различными структурами данных</p> <p>2. Технологиями программирования</p> <p>Практическими навыками программирования на выбранных языках</p> <p>Техниками использования фреймворков и библиотек</p> <p>Навыками работы с системами контроля версий</p> <p>Методами отладки и профилирования кода</p> <p>3. Инструментами разработки</p> <p>Навыками работы с интегрированными средами разработки</p> <p>Техниками использования систем сборки и развертывания</p> <p>Навыками автоматизации процессов разработки</p> <p>Методами контейнеризации и оркестрации приложений</p> <p>4. Проектными компетенциями</p> <p>Навыками проектирования архитектуры информационных систем</p> <p>Техниками разработки технической документации</p>
--	--	--

		<p>Навыками командной работы над программными проектами</p> <p>Методами управления жизненным циклом программного обеспечения</p> <p>5. Профессиональными практиками</p> <p>Навыками применения стандартов и паттернов проектирования</p> <p>Техниками обеспечения безопасности программных решений</p> <p>Навыками интеграции с внешними системами и сервисами</p> <p>Методами сопровождения и развития программных продуктов</p>
ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий	ОПК-6.3 Применяет методы программирования, отладки и тестирования прототипов программно-технических комплексов задач	<p>знает</p> <p>1. Методы программирования прототипов</p> <p>Принципы прототипной разработки (rapid prototyping)</p> <p>Методы инкрементальной и итеративной разработки</p> <p>Особенности программирования в условиях неполных требований</p> <p>Техники создания минимально работоспособного продукта (MVP)</p> <p>2. Технологии отладки программного обеспечения</p> <p>Классификацию ошибок (синтаксические, логические, runtime)</p> <p>Методы локализации ошибок в коде</p> <p>Инструменты отладки (дебаггеры, профайлеры, трассировщики)</p> <p>Техники анализа дампов памяти и логов выполнения</p> <p>3. Методологии тестирования прототипов</p> <p>Уровни тестирования (модульное, интеграционное, системное)</p> <p>Виды тестирования (функциональное,</p>

		<p>нагрузочное, регрессионное)</p> <p>Принципы тест-дизайна и создания тестовых сценариев</p> <p>Методы автоматизации тестирования</p> <p>4. Архитектуру программно-технических комплексов</p> <p>Компонентный состав программно-технических комплексов</p> <p>Принципы взаимодействия программных и аппаратных компонентов</p> <p>Протоколы обмена данными в распределенных системах</p> <p>Методы обеспечения надежности и отказоустойчивости</p> <p>5. Инструментальные средства разработки</p> <p>Среды разработки с функциями отладки (Visual Studio, IntelliJ IDEA, Eclipse)</p> <p>Фреймворки для тестирования (JUnit, Selenium, JMeter)</p> <p>Системы контроля версий для командной работы</p> <p>Инструменты непрерывной интеграции (Jenkins, GitLab CI)</p> <p>умеет</p> <p>1. Разрабатывать прототипы программных комплексов</p> <p>Создавать архитектурные решения для прототипов</p> <p>Выбирать технологии и инструменты для быстрой разработки</p> <p>Реализовывать ключевые функции программно-технических комплексов</p> <p>Интегрировать программные компоненты с аппаратными средствами</p> <p>2. Проводить отладку программного кода</p> <p>Воспроизводить и локализовать ошибки в программах</p>
--	--	---

		<p>Использовать отладчики для пошагового выполнения кода</p> <p>Анализировать стек вызовов и состояние переменных</p> <p>Применять методы логирования для диагностики проблем</p> <p>3. Организовывать тестирование прототипов</p> <p>Разрабатывать тестовые сценарии и тест-кейсы</p> <p>Выполнять различные виды тестирования программных комплексов</p> <p>Автоматизировать процессы тестирования</p> <p>Анализировать результаты тестирования и составлять отчеты</p> <p>4. Анализировать требования к программно-техническим комплексам</p> <p>Выявлять и формализовать требования к прототипам</p> <p>Оценивать техническую реализуемость требований</p> <p>Определять критерии приемки прототипов</p> <p>Планировать итерации разработки на основе обратной связи</p> <p>5. Оптимизировать процессы разработки</p> <p>Выбирать оптимальные методы программирования для конкретных задач</p> <p>Настраивать инструменты отладки и тестирования под конкретный проект</p> <p>Организовывать процессы непрерывной интеграции и доставки</p> <p>Оценивать качество кода и проводить рефакторинг</p> <p>владеет навыками</p> <p>1. Навыками прототипной разработки</p> <p>Техниками быстрого создания работающих прототипов</p> <p>Навыками итеративного проектирования архитектуры</p>
--	--	--

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

		и тестировании Методами презентации результатов разработки заказчикам Практикой передачи разработанных решений в промышленную эксплуатацию
--	--	--

3. Место дисциплины в структуре образовательной программы

Дисциплина «Прикладное программирование» является дисциплиной обязательной части программы.

Изучение дисциплины осуществляется в 4 семестре(-ах).

Для освоения дисциплины «Прикладное программирование» студенты используют знания, умения и навыки, сформированные в процессе изучения дисциплин:

Ознакомительная практика

Основы программирования Ознакомительная практика

Ознакомительная практика

Основы программирования Алгоритмы и структуры данных

Ознакомительная практика

Основы программирования Основы программирования

Ознакомительная практика

Основы программирования Объектно-ориентированное программирование

Освоение дисциплины «Прикладное программирование» является необходимой основой для последующего изучения следующих дисциплин:

Подготовка к сдаче и сдача государственного экзамена

Выполнение и защита выпускной квалификационной работы

Интернет-технологии

Web-программирование

Облачные технологии

Технологии облачных вычислений

4. Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу с обучающимися с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающихся

Общая трудоемкость дисциплины «Прикладное программирование» в соответствии с рабочим учебным планом и ее распределение по видам работ представлены ниже.

Семестр	Трудоемкость час/з.е.	Контактная работа с преподавателем, час			Самостоятельная работа, час	Контроль, час	Форма промежуточной аттестации (форма контроля)
		лекции	практические занятия	лабораторные занятия			
4	108/3	18	36		54		За
в т.ч. часов: в интерактивной форме		4	8				

Семестр	Трудоемкость час/з.е.	Внеаудиторная контактная работа с преподавателем, час/чел					
		Курсовая работа	Курсовой проект	Зачет	Дифференцированный зачет	Консультации перед экзаменом	Экзамен
4	108/3			0.12			

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

№	Наименование раздела/темы	Семестр	Количество часов					Формы текущего контроля успеваемости и промежуточной аттестации	Оценочное средство проверки результатов достижения индикаторов компетенций	Код индикаторов достижения компетенций
			всего	Лекции	Семинарские занятия		Самостоятельная работа			
					Практические	Лабораторные				
1.	1 раздел. Прикладное программирование									
1.1.	Введение в Java и инструменты разработчика	4	12	4	8		8			ОПК-6.1, ОПК-6.2, ОПК-6.3
1.2.	Углубленное ООП и основные механизмы Java	4	16	6	10		12			ОПК-6.1, ОПК-6.2, ОПК-6.3
1.3.	Кт 1	4					2	КТ 1	Тест	ОПК-6.1, ОПК-6.2, ОПК-6.3
1.4.	Работа с данными и продвинутое возможности	4	6	2	4		8			ОПК-6.1, ОПК-6.2, ОПК-6.3
1.5.	Создание приложений и интеграция	4	6	2	4		10			ОПК-6.1, ОПК-6.2, ОПК-6.3
1.6.	Кт 2	4					2	КТ 2	Тест	ОПК-6.1, ОПК-6.2, ОПК-6.3
1.7.	Завершающий проект	4	14	4	10		12			ОПК-6.1, ОПК-6.2, ОПК-6.3

	Промежуточная аттестация	За							
	Итого		108	18	36		54		
	Итого		108	18	36		54		

5.1. Лекционный курс с указанием видов интерактивной формы проведения занятий

Тема лекции (и/или наименование раздел) (вид интерактивной формы проведения занятий)/ (практическая подготовка)	Содержание темы (и/или раздела)	Всего, часов / часов интерактивных занятий/ практическая подготовка
Введение в Java и инструменты разработчика	Ввод и вывод данных. Операции с числами, строками. Форматирование	2/-
Введение в Java и инструменты разработчика	Условный оператор Циклы Вложенные циклы	2/-
Углубленное ООП и основные механизмы Java	Строки, кортежи, списки	2/-
Углубленное ООП и основные механизмы Java	Множества, словари	2/-
Углубленное ООП и основные механизмы Java	Встроенные возможности по работе с коллекциями	2/-
Работа с данными и продвинутое возможности	Функции и их особенности в Python	2/-
Создание приложений и интеграция	Объектно-ориентированное программирование	2/-
Завершающий проект	Библиотеки для получения и обработки данных	2/-
Завершающий проект	Модуль pandas Модуль requests	2/-
Итого		18

5.2.1. Семинарские (практические) занятия с указанием видов проведения занятий в интерактивной форме

Наименование раздела дисциплины	Формы проведения и темы занятий (вид интерактивной формы проведения занятий)/(практическая подготовка)	Всего, часов / часов интерактивных занятий/ практическая подготовка	
		вид	часы
Введение в Java и инструменты разработчика	Ввод и вывод данных. Операции с числами, строками. Форматирование	Пр	2/-/-
Введение в Java и инструменты разработчика	Условный оператор	Пр	2/-/-
Введение в Java и инструменты разработчика	Циклы Вложенные циклы	Пр	4/-/-
Углубленное ООП и основные механизмы Java	Строки, кортежи, списки	Пр	4/-/-

Углубленное ООП и основные механизмы Java	Множества, словари	Пр	2/-/-
Углубленное ООП и основные механизмы Java	Встроенные возможности по работе с коллекциями. Списочные выражения. Модель памяти для типов языка Python	Пр	4/-/-
Работа с данными и продвинутое возможности	Функции. Области видимости. Передача параметров в функции Позиционные и именованные аргументы. Функции высших порядков. Лямбда-функции Рекурсия. Декораторы. Генераторы	Пр	4/4/-
Создание приложений и интеграция	Объектная модель Python. Классы, поля и методы Волшебные методы, переопределение методов. Наследование Модель исключений Python. Try, except, else, finally. Модули	Пр	4/4/-
Завершающий проект	Модули math и numpy Модуль pandas Модуль requests	Пр	4/-/-
Завершающий проект	Модуль pandas	Пр	4/4/-
Завершающий проект	Модуль requests	Пр	2/-/-
Итого			

5.3. Курсовой проект (работа) учебным планом не предусмотрен

5.4. Самостоятельная работа обучающегося

Темы и/или виды самостоятельной работы	Часы
Условный оператор Циклы Вложенные циклы	8
Встроенные возможности по работе с коллекциями. Списочные выражения. Модель памяти для типов языка Python	12
Кт 1	2

Лямбда-функции Рекурсия. Декораторы. Генераторы	8
Лямбда-функции Рекурсия. Декораторы. Генераторы	10
Кт 2	2
Библиотеки для получения и обработки данных	12

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

Учебно-методическое обеспечение для самостоятельной работы обучающегося по дисциплине «Прикладное программирование» размещено в электронной информационно-образовательной среде Университета и доступно для обучающегося через его личный кабинет на сайте Университета. Учебно-методическое обеспечение включает:

1. Рабочую программу дисциплины «Прикладное программирование».
2. Методические рекомендации для организации самостоятельной работы обучающегося по дисциплине «Прикладное программирование».
3. Методические рекомендации по выполнению письменных работ () (при наличии).
4. Методические рекомендации по выполнению контрольной работы студентами заочной формы обучения (при наличии)
5. Методические указания по выполнению курсовой работы (проекта) (при наличии).

Для успешного освоения дисциплины, необходимо самостоятельно детально изучить представленные темы по рекомендуемым источникам информации:

№ п/п	Темы для самостоятельного изучения	Рекомендуемые источники информации (№ источника)		
		основная (из п.8 РПД)	дополнительная (из п.8 РПД)	метод. лит. (из п.8 РПД)
1	Введение в Java и инструменты разработчика. Условный оператор Циклы Вложенные циклы	Л1.1, Л1.2	Л2.1, Л2.2	Л3.1
2	Углубленное ООП и основные механизмы Java. Встроенные возможности по работе с коллекциями. Списочные выражения. Модель памяти для типов языка Python	Л1.1, Л1.2	Л2.1, Л2.2	Л3.1
3	Кт 1. Кт 1	Л1.1, Л1.2	Л2.2	Л3.1
4	Работа с данными и продвинутое возможности. Лямбда-функции Рекурсия. Декораторы. Генераторы	Л1.1, Л1.2	Л2.1, Л2.2	Л3.1
5	Создание приложений и интеграция. Лямбда-функции Рекурсия. Декораторы. Генераторы	Л1.1, Л1.2	Л2.1, Л2.2	Л3.1
6	Кт 2. Кт 2	Л1.1, Л1.2	Л2.2	Л3.1
7	Завершающий проект. Библиотеки для получения и обработки данных	Л1.1, Л1.2	Л2.1, Л2.2	Л3.1

7. Фонд оценочных средств (оценочных материалов) для проведения промежуточной аттестации обучающихся по дисциплине «Прикладное программирование»

7.1. Перечень индикаторов компетенций с указанием этапов их формирования в процессе освоения образовательной программы

Индикатор компетенции (код и содержание)	Дисциплины/элементы программы (практики, ГИА), участвующие в формировании индикатора компетенции	1		2		3		4	
		1	2	3	4	5	6	7	8
ОПК-6.1:Использует методы алгоритмизации, языки и технологии программирования,	Web-программирование						x		
	Алгоритмы и структуры данных		x						
	Базы данных				x				

Индикатор компетенции (код и содержание)	Дисциплины/элементы программы (практики, ГИА), участвующие в формировании индикатора компетенции	1		2		3		4	
		1	2	3	4	5	6	7	8
пригодные для практического применения в области информационных систем и технологий	Дисциплины (модули) по выбору 1 (ДВ.1)								x
	Интернет-технологии					x			
	Облачные технологии								x
	Объектно- ориентированное программирование			x					
	Ознакомительная практика		x						
	Основы программирования	x							
	Технологии облачных вычислений								x
ОПК-6.2:Применяет методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий	Web-программирование						x		
	Алгоритмы и структуры данных		x						
	Базы данных				x				
	Дисциплины (модули) по выбору 1 (ДВ.1)								x
	Интернет-технологии					x			
	Облачные технологии								x
	Объектно- ориентированное программирование			x					
	Ознакомительная практика		x						
	Основы программирования	x							
	Технологии облачных вычислений								x
ОПК-6.3:Применяет методы программирования, отладки и тестирования прототипов программно- технических комплексов задач	Web-программирование						x		
	Базы данных				x				
	Интернет-технологии					x			
	Объектно- ориентированное программирование			x					
	Ознакомительная практика		x						
	Основы программирования	x							

7.2. Критерии и шкалы оценивания уровня усвоения индикатора компетенций, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Оценка знаний, умений и навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций по дисциплине «Прикладное программирование» проводится в форме текущего контроля и промежуточной аттестации.

Текущий контроль проводится в течение семестра с целью определения уровня усвоения обучающимися знаний, формирования умений и навыков, своевременного выявления преподавателем недостатков в подготовке обучающихся и принятия необходимых мер по её коррективке, а также для совершенствования методики обучения, организации учебной работы и оказания индивидуальной помощи обучающемуся.

Промежуточная аттестация по дисциплине «Прикладное программирование» проводится в виде Зачет.

За знания, умения и навыки, приобретенные студентами в период их обучения, выставляются оценки «ЗАЧТЕНО», «НЕ ЗАЧТЕНО». (или «ОТЛИЧНО», «ХОРОШО», «УДОВЛЕТВОРИТЕЛЬНО», «НЕУДОВЛЕТВОРИТЕЛЬНО» для дифференцированного зачета/экзамена)

Для оценивания знаний, умений, навыков и (или) опыта деятельности в университете применяется балльно-рейтинговая система оценки качества освоения образовательной программы. Оценка проводится при проведении текущего контроля успеваемости и промежуточных аттестаций обучающихся. Рейтинговая оценка знаний является интегрированным показателем качества теоретических и практических знаний и навыков студентов по дисциплине.

Состав балльно-рейтинговой оценки студентов очной формы обучения

Для студентов очной формы обучения знания по осваиваемым компетенциям формируются на лекционных и практических занятиях, а также в процессе самостоятельной подготовки.

В соответствии с балльно-рейтинговой системой оценки, принятой в Университете студентам начисляются баллы по следующим видам работ:

№ контрольной точки	Оценочное средство результатов индикаторов достижения компетенций		Максимальное количество баллов
4 семестр			
КТ 1	Тест		15
КТ 2	Тест		15
Сумма баллов по итогам текущего контроля			30
Посещение лекционных занятий			20
Посещение практических/лабораторных занятий			20
Результативность работы на практических/лабораторных занятиях			30
Итого			100
№ контрольной точки	Оценочное средство результатов индикаторов достижений компетенций	Максимальное количество баллов	Критерии оценки знаний студентов
4 семестр			
КТ 1	Тест	15	11-15 баллов выставляется обучающемуся, если тестовые задания выполняются на 85% и выше; 8-10 баллов выставляется обучающемуся, если тестовые задания выполняются на 70 - 84%; 5-7 баллов выставляется обучающемуся, если тестовые задания выполняются на 55 – 69 %; 1-4 балла выставляется обучающемуся, если тестовые задания выполняются на 45 – 54%; 0 баллов выставляется обучающемуся, если тестовые задания выполняются на 44% и меньше.
КТ 2	Тест	15	11-15 баллов выставляется обучающемуся, если тестовые задания выполняются на 85% и выше; 8-10 баллов выставляется обучающемуся, если тестовые задания выполняются на 70 - 84%; 5-7 баллов выставляется обучающемуся, если тестовые задания выполняются на 55 – 69 %; 1-4 балла выставляется обучающемуся, если тестовые задания выполняются на 45 – 54%; 0 баллов выставляется обучающемуся, если тестовые задания выполняются на 44% и меньше.

Критерии и шкалы оценивания результатов обучения на промежуточной аттестации

При проведении итоговой аттестации «зачет» («дифференцированный зачет», «экзамен») преподавателю с согласия студента разрешается выставить оценки («отлично», «хорошо», «удовлетворительно», «зачет») по результатам набранных баллов в ходе текущего контроля успеваемости в семестре по выше приведенной шкале.

В случае отказа – студент сдает зачет (дифференцированный зачет, экзамен) по приведенным выше вопросам и заданиям. Итоговая успеваемость (зачет, дифференцированный зачет, экзамен) не может оцениваться ниже суммы баллов, которую студент набрал по итогам текущей и промежуточной успеваемости.

При сдаче (зачета, дифференцированного зачета, экзамена) к заработанным в течение семестра студентом баллам прибавляются баллы, полученные на (зачете, дифференцированном зачете, экзамене) и сумма баллов переводится в оценку.

Критерии и шкалы оценивания ответа на зачете

По дисциплине «Прикладное программирование» к зачету допускаются студенты, выполнившие и сдавшие практические работы по дисциплине, имеющие ежемесячную аттестацию и без привязке к набранным баллам. Студентам, набравшим более 65 баллов, зачет выставляется по результатам текущей успеваемости, студенты, не набравшие 65 баллов, сдают зачет по вопросам, предусмотренным РПД. Максимальная сумма баллов по промежуточной аттестации (зачету) устанавливается в 15 баллов

Вопрос билета	Количество баллов
Теоретический вопрос	до 5
Задания на проверку умений	до 5
Задания на проверку навыков	до 5

Теоретический вопрос

5 баллов выставляется студенту, полностью освоившему материал дисциплины или курса в соответствии с учебной программой, включая вопросы рассматриваемые в рекомендованной программой дополнительной справочно-нормативной и научно-технической литературы, свободно владеющему основными понятиями дисциплины. Требуется полное понимание и четкость изложения ответов по экзаменационному заданию (билету) и дополнительным вопросам, заданных экзаменатором. Дополнительные вопросы, как правило, должны относиться к материалу дисциплины или курса, не отраженному в основном экзаменационном задании (билете) и выявляют полноту знаний студента по дисциплине.

4 балла заслуживает студент, ответивший полностью и без ошибок на вопросы экзаменационного задания и показавший знания основных понятий дисциплины в соответствии с обязательной программой курса и рекомендованной основной литературой.

3 балла дан недостаточно полный и недостаточно развернутый ответ. Логика и последовательность изложения имеют нарушения. Допущены ошибки в раскрытии понятий, употреблении терминов. Студент не способен самостоятельно выделить существенные и несущественные признаки и причинно-следственные связи. Студент может конкретизировать обобщенные знания, доказав на примерах их основные положения только с помощью преподавателя. Речевое оформление требует поправок, коррекции.

2 балла дан неполный ответ, представляющий собой разрозненные знания по теме вопроса с существенными ошибками в определениях. Присутствуют фрагментарность, нелогичность изложения. Студент не осознает связь данного понятия, теории, явления с другими объектами дисциплины. Отсутствуют выводы, конкретизация и доказательность изложения. Речь неграмотная. Дополнительные и уточняющие вопросы преподавателя не приводят к коррекции ответа студента не только на поставленный вопрос, но и на другие вопросы дисциплины.

1 балл дан неполный ответ, представляющий собой разрозненные знания по теме вопроса с существенными ошибками в определениях. Присутствуют фрагментарность, нелогичность изложения. Студент не осознает связь данного понятия, теории, явления с другими объектами дисциплины. Отсутствуют выводы, конкретизация и доказательность изложения. Речь неграмотная. Дополнительные и уточняющие вопросы преподавателя не приводят к коррекции ответа студента не только на поставленный вопрос, но и на другие вопросы дисциплины.

0 баллов - при полном отсутствии ответа, имеющего отношение к вопросу.

Задания на проверку умений и навыков

5 баллов Задания выполнены в обозначенный преподавателем срок, письменный отчет без замечаний. Работа выполнена в полном объеме с соблюдением необходимой последовательности.

4 балла Задания выполнены в обозначенный преподавателем срок, письменный отчет с небольшими недочетами.

2 баллов Задания выполнены с задержкой, письменный отчет с недочетами. Работа выполнена не полностью, но объем выполненной части таков, что позволяет получить правильные результаты и выводы.

1 баллов Задания выполнены частично, с большим количеством вычислительных ошибок, объем выполненной части работы не позволяет сделать правильных выводов.

0 баллов Задания выполнены, письменный отчет не представлен или работа выполнена не полностью, и объем выполненной части работы не позволяет сделать правильных выводов.

7.3. Примерные оценочные материалы для текущего контроля успеваемости и промежуточной аттестации по итогам освоения дисциплины «Прикладное программирование»

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 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

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) Скрытие 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. Соккрытие 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 ревьюером
- 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)

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

а) Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины

основная

Л1.1 Бабушкина И. А., Окулов С. М. Практикум по объектно-ориентированному программированию [Электронный ресурс]:. - Москва: Лаборатория знаний, 2020. - 369 с. – Режим доступа: <https://e.lanbook.com/book/135561>

Л1.2 Дадян Э. Г. Проектирование современных баз данных [Электронный ресурс]:учеб.-метод. пособие; ВО - Специалитет. - Москва: ООО "Научно-издательский центр ИНФРА-М", 2017. - 120 с. – Режим доступа: <http://new.znaniy.com/go.php?id=959294>

дополнительная

Л2.1 Сакулин С. А. Основы интернет-технологий: HTML, CSS, JavaScript, XML [Электронный ресурс]:учеб. пособие; ВО - Бакалавриат. - Москва: МГТУ им. Н.Э. Баумана, 2017. - 112 с. – Режим доступа: <https://e.lanbook.com/book/103525>

Л2.2 Гуриков С. Р. Интернет-технологии [Электронный ресурс]:учеб. пособие ; ВО - Бакалавриат. - Москва: ООО "Научно-издательский центр ИНФРА-М", 2022. - 174 с. – Режим доступа: <http://znaniy.com/catalog/document?id=391737>

б) Методические материалы, разработанные преподавателями кафедры по дисциплине, в соответствии с профилем ОП.

Л3.1 Никитина Т. П., Королев Л. В. Программирование. Основы Python для инженеров [Электронный ресурс]:учеб. пособие; ВО - Бакалавриат, Магистратура. - Санкт-Петербург: Лань, 2023. - 156 с. – Режим доступа: <https://e.lanbook.com/book/302720>

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

№	Наименование ресурса сети «Интернет»	Электронный адрес ресурса
1	Python : официальный сайт. – URL: https://www.python.org/ (дата обращения: 29.08.2023).	https://www.python.org/
2	NumPy : официальный сайт. – URL: https://numpy.org/ (дата обращения: 29.08.2023).	https://numpy.org/

10. Методические указания для обучающихся по освоению дисциплины

Лекционные занятия

Основа освоения дисциплины – лекция, целью которой является целостное и логичное рассмотрение основного материала курса. Вместе с тем значимость лекции определяется тем, что она

не только способствует выработке логического мышления, но и способствует развитию интереса к пониманию современной действительности.

Задача студентов в процессе умелой и целеустремленной работы на лекциях – внимательно слушать преподавателя, следить за его мыслью, предлагаемой системой логических посылок, доказательств и выводов, фиксировать (записывать) основные идеи, важнейшие характеристики понятий, теорий, наиболее существенные факты. Лекция задает направление, содержание и эффективность других форм учебного процесса, нацеливает студентов на самостоятельную работу и определяет основные ее направления (подготовку к практическим занятиям, выполнение творческих заданий, рефератов, решение контекстных задач).

Записывание лекции – творческий процесс. Запись лекции крайне важна. Это позволяет надолго сохранить основные положения лекции; способствует поддержанию внимания; способствует лучшему запоминанию материала. Важно уметь оформить конспект так, чтобы важные моменты были выделены графически, а главную информацию следует выделять в самостоятельные абзацы, фиксируя ее более крупными буквами или цветными маркерами. Конспект должен иметь поля для заметок. Это могут быть библиографические ссылки и, наконец, собственные комментарии. Для быстрой записи текста можно придумать условные знаки, при этом таких знаков не должно быть более 10–15. Условные обозначения придумывают для часто встречающихся слов (существует, который, каждый, точка зрения, на основании и т.п.).

Перед каждой лекцией необходимо внимательно прочитать материал предыдущей лекции. В рабочей тетради графически выделить: тему лекции, основные теоретические положения. Подготовленный студент легко следит за мыслью преподавателя, что позволяет быстрее запоминать новые понятия, сущность которых выявляется в контексте лекции. Повторение материала облегчает в дальнейшем подготовку к экзамену. Затем надо ознакомиться с материалом темы по учебнику, внести нужные уточнения и дополнения в лекционный материал. После усвоения каждой темы рекомендуется проверять свои знания, отвечая на контрольные вопросы по теме.

Лабораторные занятия

Целью лабораторных занятий является закрепление, расширение, углубление теоретических знаний, полученных на лекциях и в ходе самостоятельной работы, развитие познавательных способностей.

Являясь частью образовательного процесса, лабораторные занятия преследует ряд основополагающих задач:

- работа с источниками, которая идет на уровнях индивидуальной самостоятельной работы и в ходе коллективного обсуждения;
- формирование умений и навыков индивидуальной и коллективной работы, позволяющих эффективно использовать основные методы исследования, грамотно выстраивать его основные технологические этапы (знакомство с темой и имеющейся по ней информацией, определение основной проблемы, первичный анализ, определение подходов и ключевых узлов механизма ее развития, публичное обсуждение, предварительные выводы);
- анализ поставленных проблем, умение обсуждать тему, высказывать свое мнение, отстаивать свою позицию, слушать и оценивать различные точки зрения, конструктивно полемизировать, учиться думать, говорить, слушать, понимать, находить точки соприкосновения разных позиций, их разумного сочетания;
- формирование установок на творчество;
- диалог, внутренний и внешний; поиск и разрешение проблемы в рамках имеющейся о ней информации;
- поиск рационального зерна в самых противоречивых позициях и подходах к проблеме;
- открытость новому и принципиальную возможность изменить свою позицию и вытекающие из нее решения, в случае получения новой информации и связанных с ней обстоятельств сознательный отход от подготовленного к семинару текста во время своего, построенного на тезис-ном изложении фактов и мыслей, когда конспект привлекается лишь в том случае, когда надо привести какие-то факты.

11. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства и информационных справочных систем (при необходимости).

11.1 Перечень лицензионного программного обеспечения

1. Kaspersky Total Security - Антивирус
2. Microsoft Windows Server STDCORE AllLngLicense/Software AssurancePack Academic OLV 16Licenses LevelE AdditionalProduct CoreLic 1Year - Серверная операционная система

11.3 Перечень программного обеспечения отечественного производства

1. Kaspersky Total Security - Антивирус

При осуществлении образовательного процесса студентами и преподавателем используются следующие информационно справочные системы: СПС «Консультант плюс», СПС «Гарант».

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

№ п/п	Наименование специальных помещений и помещений для самостоятельной работы	Номер аудитории	Оснащенность специальных помещений и помещений для самостоятельной работы
1	Учебная аудитория для проведения занятий всех типов (в т.ч. лекционного, семинарского, практической подготовки обучающихся), групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации	Э-160 422/НК 422/НК	Специализированная мебель на 180 посадочных мест, персональный компьютер – 1 шт., проектор Panasonic EX620 X6A – 1 шт., интерактивная доска SMART Board 690 – 1 шт., трибуна для лектора – 1 шт., микрофон – 1 шт., мониторы - 3 шт., плазменная панель - 1 шт., учебно-наглядные пособия в виде презентаций, информационные плакаты, подключение к сети «Интернет», выход в корпоративную сеть университета. Оснащение: специализированная мебель на 38 посадочных мест, стол преподавателя – 1 шт., Sharp 70" Информационный ЖК-дисплей – 1 шт., АРМ на основе Intel Core i3 , Монитор Philips 23", Клавиатура + мышь - 25шт., магнитно-маркерная доска – 1 шт., учебно-наглядные пособия в виде тематических презентаций, информационные плакаты, подключение к сети «Интернет», доступ в электронную информационно-образовательную среду университета, выход в корпоративную сеть университета. Оснащение: специализированная мебель на 38

		423/НК	Оснащение: специализированная мебель на 56 посадочных мест, стол преподавателя – 1 шт., Sharp 70" Информационный ЖК-дисплей – 1 шт., магнитно-маркерная доска – 1 шт., учебно-наглядные пособия в виде тематических презентаций, информационные плакаты, подключение к сети «Интернет», доступ в электронную информационно-образовательную среду университета, выход в корпоративную сеть университета.
2	Помещение для самостоятельной работы обучающихся, подтверждающее наличие материально-технического обеспечения, с перечнем основного оборудования		
		213/НК библио тека	Специализированная мебель на 35 посадочных мест, дисплей - 1 шт., принтер ч/б - 2 шт., МФУ ч/б - 2 шт., сканер - 2 шт., открытый доступ к фонду справочной, краеведческой литературы, Wi-Fi оборудование, подключение к сети «Интернет», доступ к российским и международным ресурсам и базам данных, доступ к электронно-библиотечным системам, доступ в электронную информационно-образовательную среду университета. Открытый доступ к фонду справочной и краеведческой литературы.
		214/НК библио тека	Специализированная мебель на 130 посадочных мест, персональные компьютеры, моноблоки – 80 шт., копир А3 - 3, принтер матричный - 2, МФУ ч/б – 7 шт., МФУ цветной – 2 шт., принтер ч/б – 8 шт., принтер цветн. - 2 шт., сканер – 2 шт., сканеры штрих-кода - 5, наушники - 10 шт., Wi-Fi оборудование, подключение к сети «Интернет», доступ к российским и международным ресурсам и базам данных, доступ к электронно-библиотечным системам, доступ в электронную информационно-образовательную среду университета. Открытый доступ к фонду учебной, научной и художественной литературы.

13. Особенности реализации дисциплины лиц с ограниченными возможностями здоровья

Обучающимся с ограниченными возможностями здоровья предоставляются специальные учебники и учебные пособия, иная учебная литература, специальные технические средства обучения коллективного и индивидуального пользования, предоставление услуг ассистента (помощника), оказывающего обучающимся необходимую техническую помощь, а также услуги сурдопереводчиков и тифлосурдопереводчиков.

а) для слабовидящих:

- на промежуточной аттестации присутствует ассистент, оказывающий студенту необходимую техническую помощь с учетом индивидуальных особенностей (он помогает занять рабочее место, передвигаться, прочесть и оформить задание, в том числе записывая под диктовку);
- задания для выполнения, а также инструкция о порядке проведения промежуточной аттестации оформляются увеличенным шрифтом;
- задания для выполнения на промежуточной аттестации зачитываются ассистентом;
- письменные задания выполняются на бумаге, надиктовываются ассистенту;
- обеспечивается индивидуальное равномерное освещение не менее 300 люкс;
- студенту для выполнения задания при необходимости предоставляется увеличивающее устройство;

в) для глухих и слабослышащих:

- на промежуточной аттестации присутствует ассистент, оказывающий студенту необходимую техническую помощь с учетом индивидуальных особенностей (он помогает занять рабочее место, передвигаться, прочесть и оформить задание, в том числе записывая под диктовку);
- промежуточная аттестация проводится в письменной форме;
- обеспечивается наличие звукоусиливающей аппаратуры коллективного пользования, при необходимости поступающим предоставляется звукоусиливающая аппаратура индивидуального пользования;
- по желанию студента промежуточная аттестация может проводиться в письменной форме;

д) для лиц с нарушениями опорно-двигательного аппарата (тяжелыми нарушениями двигательных функций верхних конечностей или отсутствием верхних конечностей):

- письменные задания выполняются на компьютере со специализированным программным обеспечением или надиктовываются ассистенту;
- по желанию студента промежуточная аттестация проводится в устной форме.

Рабочая программа дисциплины «Прикладное программирование» составлена на основе Федерального государственного образовательного стандарта высшего образования - бакалавриат по направлению подготовки 09.03.02 Информационные системы и технологии (приказ Минобрнауки России от 19.09.2017 г. № 926).

Автор (ы)

_____ доцент , кпн Богданова Светлана Викторовна

Рецензенты

_____ доцент , к.т.н. Шлаев Дмитрий Валерьевич

_____ доцент , к.т.н. Трошков Александр Михайлович

Рабочая программа дисциплины «Прикладное программирование» рассмотрена на заседании Кафедра информационных систем протокол № 8 от 04.03.2025 г. и признана соответствующей требованиям ФГОС ВО и учебного плана по направлению подготовки 09.03.02 Информационные системы и технологии

Заведующий кафедрой _____ Березницкий А.С.

Рабочая программа дисциплины «Прикладное программирование» рассмотрена на заседании учебно-методической комиссии Факультет цифровых технологий протокол № от г. и признана соответствующей требованиям ФГОС ВО и учебного плана по направлению подготовки 09.03.02 Информационные системы и технологии

Руководитель ОП _____