

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**



**Частное учреждение высшего образования
«Высшая школа предпринимательства (институт)»
(ЧУВО «ВШП»)**

**РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ
Б1.В.06 «Инструментальные средства разработки программного
обеспечения»**

Направление подготовки: 09.03.02 Информационные системы и технологии

**Направленность (профиль) программы бакалавриата
«Информационные технологии»**

ОДОБРЕНО

Ученым советом ЧУВО «ВШП»

Протокол заседания

№01-02/24 от 15 мая 2025 г.



УТВЕРЖДАЮ

УВО «ВШП»

15» мая 2025 г.

Аллабян М.Г.

Тверь, 2025

Рабочая программа учебной дисциплины **Б1.В.06 Инструментальные средства разработки программного обеспечения**, компонента основной профессиональной образовательной программы высшего образования — программы бакалавриата по направлению подготовки **09.03.02 Информационные системы и технологии** направленность (профиль) **«Информационные технологии»**, направлена на обеспечение у обучающегося способности осуществлять профессиональную деятельность в соответствующей области и сферах профессиональной деятельности, в том числе на их практическую подготовку с учётом рабочей программы воспитания и календарного плана воспитательной работы Частном учреждении высшего образования **«Высшая школа предпринимательства (институт)»** (далее — **ЧУВО «ВШП»**).

1. ОБЛАСТЬ ПРИМЕНЕНИЯ И НОРМАТИВНЫЕ ССЫЛКИ

Настоящая рабочая программа учебной дисциплины устанавливает требования к результатам обучения студента и определяет содержание и виды учебных занятий и отчетности.

Программа предназначена для преподавателей и студентов направления подготовки 09.03.02 Информационные системы и технологии.

Программа учебной дисциплины разработана в соответствии с ФГОС ВО, утвержденного приказом Минобрнауки России от 19.09.2017 № 926 «Об утверждении федерального государственного образовательного стандарта высшего образования - бакалавриата по направлению подготовки 09.03.02 Информационные системы и технологии», основной профессиональной образовательной программой высшего образования по направлению подготовки 09.03.02 Информационные системы и технологии, направленность (профиль) Информационные технологии.

2. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ

Целью изучения дисциплины «Инструментальные средства разработки программного обеспечения» является овладение основами теоретических и практических знаний в области инструментальных средств, используемых для реализации проектов информационных систем.

Для этого в рамках дисциплины решаются следующие задачи:

- изучение теоретических знаний в области инструментальных средств информационных систем;
- формирование умения использовать современные инструментальные средства в области информационных систем;
- приобретение практических навыков работы пользователя с инструментальными средствами;
- возможностей и основных принципов использования информационно-справочных систем;
- технологий использования программного обеспечения для автоматизации финансово-хозяйственной деятельности предприятий.

3. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП ВО

Дисциплина является компонентом части, формируемой участниками образовательных отношений Блока 1 основной профессиональной образовательной программы высшего образования по направлению подготовки 09.03.02 Информационные системы и технологии, направленность (профиль) — Информационные технологии.

4. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО УЧЕБНОЙ ДИСЦИПЛИНЕ В РАМКАХ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОСВОЕНИЯ ОСНОВНОЙ ПРОФЕССИОНАЛЬНОЙ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Процесс освоения учебной дисциплины направлен на формирование у обучающихся следующих компетенций.

В результате освоения учебной дисциплины обучающийся должен демонстрировать следующие результаты обучения: УК-1, УК-2, ПК-1, ПК-2, ПК-3, ПК-4.

Таблица 1. Результаты обучения

Код компетенции	Наименование компетенции	Индекс и наименование индикатора содержания компетенции	Дескрипторы – основные признаки освоения (показатели достижения результата)
УК-1	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	УК-1.1 Способен осуществлять поиск и критический анализ информации	Знать: - Основные источники информации для разработки программного обеспечения (далее ПО). - Методы критического анализа информации. Уметь: - Находить актуальную информацию в специализированных источниках. - Анализировать и оценивать достоверность и релевантность информации. Владеть: - Навыками эффективного поиска информации. - Навыками критического мышления при анализе информации.
		УК-1.2 Способен применять системный подход для решения поставленных задач	Знать: - Принципы системного подхода в разработке ПО. - Основные этапы системного анализа и проектирования. Уметь: - Применять системный подход для анализа задач. - Разрабатывать системные решения для поставленных задач. Владеть: - Навыками системного анализа проблем. - Навыками интеграции различных элементов системы в единое целое.
УК-2	Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1 Способен определять круг задач в рамках поставленной цели	Знать: - Методы декомпозиции целей на задачи. - Принципы приоритизации задач. Уметь: - Определять и формулировать задачи на основе целей проекта. - Устанавливать приоритеты для выполнения задач. Владеть: - Навыками декомпозиции целей на задачи. - Навыками управления задачами и ресурсами.
		УК-2.2 Способен выбирать оптимальные способы решения задач, исходя из правовых норм, ресурсов и ограничений	Знать: - Основные правовые нормы в области информационных технологий (далее ИТ). - Принципы оценки и управления ресурсами. Уметь: - Оценивать доступные ресурсы и ограничения. - Выбирать наилучшие способы решения задач с учетом правовых норм. Владеть:

			<ul style="list-style-type: none"> - Навыками оценки ресурсов и ограничений. - Навыками принятия оптимальных решений в рамках правовых норм.
ПК-1	Разработка и отладка программного кода	ПК-1.1 Способен разрабатывать программный код	Знать: <ul style="list-style-type: none"> - Основные языки программирования для разработки ПО. - Принципы написания качественного кода. Уметь: <ul style="list-style-type: none"> - Писать программный код на одном из изучаемых языков. - Разрабатывать алгоритмы и структуры данных для решения задач. Владеть: <ul style="list-style-type: none"> - Навыками написания и документирования кода. - Навыками использования инструментов разработки.
		ПК-1.2 Способен отлаживать программный код	Знать: <ul style="list-style-type: none"> - Основные методы и инструменты отладки кода. - Принципы устранения ошибок и багов в коде. Уметь: <ul style="list-style-type: none"> - Использовать отладчики и логи для выявления ошибок. - Исправлять выявленные ошибки и оптимизировать код. Владеть: <ul style="list-style-type: none"> - Навыками отладки и тестирования кода. - Навыками применения инструментов для отладки.
ПК-2	Проверка работоспособности и рефакторинг кода программного обеспечения	ПК-2.1 Способен проверять работоспособность программного кода	Знать: <ul style="list-style-type: none"> - Принципы и методы тестирования кода. - Основные виды тестов (модульные, интеграционные, системные). Уметь: <ul style="list-style-type: none"> - Писать и выполнять тесты для проверки кода. - Использовать тестовые фреймворки для автоматизации тестирования. Владеть: <ul style="list-style-type: none"> - Навыками написания и выполнения тестов. - Навыками применения инструментов для тестирования.
		ПК-2.2 Способен рефакторить программный код	Знать: <ul style="list-style-type: none"> - Принципы и методы рефакторинга. - Основные техники улучшения структуры кода. Уметь: <ul style="list-style-type: none"> - Анализировать код на предмет улучшений. - Применять техники рефакторинга для повышения качества кода. Владеть: <ul style="list-style-type: none"> - Навыками рефакторинга кода. - Навыками оценки и улучшения структуры кода.

ПК-3	Интеграция программных модулей и компонентов и проверка работоспособности выпусков программного продукта	ПК-3.1 Способен интегрировать программные модули и компоненты	Знать: - Принципы интеграции модулей и компонентов. - Методы и инструменты для интеграции. Уметь: - Интегрировать модули и компоненты в единое приложение. - Проверять корректность интеграции и совместимость компонентов. Владеть: - Навыками интеграции программных модулей. - Навыками использования инструментов для интеграции.
		ПК-3.2 Способен проверять работоспособность выпусков программного продукта	Знать: - Методы и инструменты для проверки выпусков ПО. - Принципы контроля качества выпусков. Уметь: - Проводить тестирование и верификацию выпусков ПО. - Оценивать стабильность и производительность программного продукта. Владеть: - Навыками тестирования и верификации выпусков ПО. - Навыками применения инструментов контроля качества.
ПК-4	Разработка требований и проектирование программного обеспечения	ПК-4.1 Способен разрабатывать требования к программному обеспечению	Знать: - Принципы и методы анализа требований. - Техники документирования требований. Уметь: - Собирать и анализировать требования к ПО. - Документировать требования в виде спецификаций. Владеть: - Навыками анализа и сбора требований. - Навыками создания и поддержки документации по требованиям.
		ПК-4.2 Способен проектировать программное обеспечение	Знать: - Принципы и методы проектирования ПО. - Основные подходы к архитектурному дизайну. Уметь: - Разрабатывать архитектуру и дизайн ПО. - Использовать UML и другие инструменты для проектирования. Владеть: - Навыками проектирования программного обеспечения. - Навыками применения инструментов для проектирования.

5. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

5.1. Объем дисциплины в зачетных единицах с указанием количества академических часов

Общая трудоемкость учебной дисциплины составляет 4 зачетные единицы, 144 часа, включая все формы контактной и самостоятельной работы обучающихся.

Объем дисциплины по учебному плану составляет –
4 зачётных единицы = 144 академических часа.

Контактная работа обучающегося (студенты) с научно-педагогическим работником организации (всего) - 48 академических часов,

в том числе:

Лекционные занятия (Лек.) - 18 академических часов,

Практические занятия (Пр.) - 28 академических часов,

Консультации (Конс.) - 2 академических часа.

Самостоятельная работа обучающегося (студента):

Самостоятельная работа (СР) - 51 академический час,

Текущий контроль успеваемости

и промежуточно-заочная аттестация обучающегося (студента):

Часы на контроль - 45 академических часов.

Таблица 2. Объём дисциплины

№ п/п	Раздел/тема дисциплины	Семестр/ курс	Виды учебной деятельности, включая самостоятельную работу обучающихся (студентов), и трудоёмкость (в ак. часах)				Коды формируемых компетенций
			Виды учебных занятий по дисциплине			Самос стоятел ьная работа	
			Лек.	Пр.	Консул ьт		
1	Тема 1. Введение в инструментальные средства разработки ПО	5 семестр/ 3 курс	2	2		6	УК-1.1, УК-1.2
2	Тема 2. Интегрированные среды разработки (IDE)	5 семестр/ 3 курс	2	2		6	УК-2.1, УК-2.2
3	Тема 3. Системы контроля версий (Git)	5 семестр/ 3 курс	2	2		6	ПК-1.1, ПК-1.2
4	Тема 4. Инструменты для управления зависимостями	5 семестр/ 3 курс	2	2		6	ПК-3.1, ПК-3.2, ПК-4.1, ПК-4.2
5	Тема 5. Линтеры и форматтеры кода	5 семестр/ 3 курс	2	2		6	ПК-2.1, ПК-2.2
6	Тема 6. Подготовка к зачету	5 семестр/ 3 курс		4		9	УК-1.1, УК-1.2, УК-2.1, УК-2.2, ПК-1.1, ПК-1.2, ПК-2.1, ПК-2.2, ПК-3.1, ПК-3.2

							ПК-4.1, ПК-4.2
7	Тема 7. Инструменты для тестирования ПО	6 семестр/ 3 курс	1	2		2	ПК-2.1, ПК-2.2
8	Тема 8 Дебаггеры и профилировщики	6 семестр/ 3 курс	2	2		2	ПК-1.2, ПК-2.1
9	Тема 9 Инструменты для CI/CD	6 семестр/ 3 курс	2	2		2	ПК-3.1, ПК-3.2
10	Тема 10 Мониторинг и логирование	6 семестр/ 3 курс	1	2		2	УК-1.2, УК-2.2
11	Тема 11 Контейнеризация и оркестрация	6 семестр/ 3 курс	2	2		2	ПК-3.1, ПК-3.2
12	Тема 12 Итоговое занятие и подготовка к экзамену	6 семестр/ 3 курс		4	2	2	УК-1.1, УК-1.2, УК-2.1, УК-2.2, ПК-1.1, ПК-1.2, ПК-2.1, ПК-2.2, ПК-3.1, ПК-3.2, ПК-4.1, ПК-4.2
ИТОГО аудиторных часов/СР:		5-6 семестры/ 3 курс	48 ак. часов			51 ак. часа	-
Часы на контроль		5-6 семестры/ 3 курс	45 ак. час (форма промежуточной аттестации – зачёт – 5 семестр) (форма промежуточной аттестации – экзамен – 6 семестр)				
ВСЕГО ак. часов:		5-6 семестры/ 3 курс	144 академических часа				

5.2. Тематическое содержание дисциплины

* количество академических часов и виды занятий представлены в таблице № 2.

5 семестр

Тема 1: Введение в инструментальные средства разработки ПО

Основные виды инструментов для разработки ПО. Введение в IDE и текстовые редакторы.

Тема 2: Интегрированные среды разработки (IDE)

Обзор популярных IDE (IntelliJ IDEA, Visual Studio, Eclipse). Настройка и использование.

Тема 3: Системы контроля версий (Git)

Основы работы с Git. Основные команды и концепции (commit, branch, merge). Инструменты для визуализации и управления репозиториями (GitKraken, Sourcetree).

Тема 4: Инструменты для управления зависимостями

Использование инструментов для управления зависимостями (Maven, Gradle, npm).

Тема 5: Линтеры и форматтеры кода

Инструменты для статического анализа кода и его форматирования. Использование линтеров

(ESLint, Pylint) и форматтеров (Prettier, Black).

Тема 6: Подготовка к зачету

Повторение и обобщение пройденного материала. Решение типовых задач и вопросов для подготовки к зачету.

6 семестр

Тема 7: Инструменты для тестирования ПО

Обзор инструментов для юнит-тестирования и интеграционного тестирования (JUnit, pytest, Selenium).

Тема 8: Дебаггеры и профилировщики

Использование отладчиков и профилировщиков для анализа и оптимизации кода. Примеры инструментов (gdb, VisualVM, PyCharm Debugger).

Тема 9: Инструменты для CI/CD

Введение в инструменты непрерывной интеграции и развертывания. Обзор популярных инструментов (Jenkins, GitHub Actions, GitLab CI).

Тема 10: Мониторинг и логирование

Инструменты для мониторинга и логирования (Prometheus, Grafana, ELK Stack).

Тема 11: Контейнеризация и оркестрация

Введение в контейнеризацию с Docker и оркестрацию с Kubernetes.

Тема 12: Итоговое занятие и подготовка к экзамену

Повторение и обобщение пройденного материала. Решение типовых задач и вопросов для подготовки к экзамену. Обсуждение практических аспектов использования инструментальных средств разработки.

5.2.1 Содержание практических занятий

Таблица 3

№ п/п	Наименование темы (раздела) дисциплины	Содержание практического занятия
5 семестр		
1	Практическое занятие 1: Введение в инструментальные средства разработки ПО	Задание: Обзор и настройка IDE и текстовых редакторов. Цель: Понять различия между IDE и текстовыми редакторами, настроить рабочую среду.
2	Практическое занятие 2: Интегрированные среды разработки (IDE)	Задание: Работа в различных IDE (IntelliJ IDEA, Visual Studio, Eclipse). Цель: Освоить основные функции и настройки популярных IDE.
3	Практическое занятие 3: Системы контроля версий (Git)	Задание: Основные команды и концепции Git. Работа с GitKraken или Sourcetree. Цель: Освоить основные команды и принципы работы с Git.
4	Практическое занятие 4: Инструменты для управления зависимостями	Задание: Управление зависимостями с помощью Maven, Gradle, npm. Цель: Понять и применять инструменты для управления зависимостями в проектах.
5	Практическое занятие 5: Линтеры и форматтеры кода	Задание: Использование линтеров (ESLint, Pylint) и форматтеров (Prettier, Black) для проверки и форматирования кода. Цель: Освоить инструменты для статического анализа и форматирования кода.
6	Практическое занятие	Задание: Повторение пройденного материала, решение типовых задач и

	6: Подготовка к зачету	вопросов для подготовки к зачету. Цель: Повторить и обобщить пройденный материал для успешной сдачи зачета.
6 семестр		
7	Практическое занятие 7: Инструменты для тестирования ПО	Задание: Юнит-тестирование и интеграционное тестирование с использованием JUnit, pytest, Selenium. Цель: Освоить инструменты для тестирования ПО.
8	Практическое занятие 8: Дебаггеры и профилировщики	Задание: Использование отладчиков и профилировщиков (gdb, VisualVM, PyCharm Debugger) для анализа и оптимизации кода. Цель: Освоить инструменты для отладки и профилирования кода.
9	Практическое занятие 9: Инструменты для CI/CD	Задание: Настройка CI/CD с использованием Jenkins, GitHub Actions, GitLab CI. Цель: Понять и применять инструменты для непрерывной интеграции и развертывания.
10	Практическое занятие 10: Мониторинг и логирование	Задание: Настройка и использование инструментов для мониторинга и логирования (Prometheus, Grafana, ELK Stack). Цель: Освоить инструменты для мониторинга и логирования.
11	Практическое занятие 11: Контейнеризация и оркестрация	Задание: Введение в Docker и Kubernetes. Создание и управление контейнерами. Цель: Освоить основы контейнеризации и оркестрации.
12	Практическое занятие 12: Итоговое занятие и подготовка к экзамену	Задание: Повторение и обобщение пройденного материала, решение типовых задач и вопросов для подготовки к экзамену. Цель: Подготовиться к экзамену, обсудить практические аспекты использования инструментальных средств разработки.

5.2.2 Содержание самостоятельной работы

Таблица 4

№ п/п	Наименование темы (раздела) дисциплины	Содержание самостоятельной работы	Форма контроля
5 семестр			
1	Самостоятельная работа 1: Изучение инструментальных средств разработки ПО	Задание: Прочитать рекомендованные главы из учебников по инструментальным средствам разработки ПО. Написать краткий обзор. Цель: Освоить основные виды инструментальных средств разработки ПО и их применение.	Ответы на контрольные вопросы.
2	Самостоятельная работа 2: Работа с различными IDE	Задание: Изучить возможности различных IDE (IntelliJ IDEA, Visual Studio, Eclipse). Подготовить сравнительный анализ. Цель: Понять различия между IDE и выбрать наиболее подходящую для своих нужд.	Выполнение практического задания.
3	Самостоятельная работа 3: Основы работы с Git	Задание: Изучить основные команды и концепции Git. Выполнить практическое задание по созданию репозитория и управлению версиями. Цель: Освоить работу с системой контроля версий Git.	Выполнение практического задания.
4	Самостоятельная работа 4: Управление зависимостями	Задание: Изучить инструменты для управления зависимостями (Maven, Gradle, npm). Применить знания на учебном проекте. Цель: Понять и научиться применять инструменты для управления зависимостями.	Выполнение практического задания.
5	Самостоятельная работа 5: Линтеры и форматтеры кода	Задание: Ознакомиться с линтерами (ESLint, Pylint) и форматтерами (Prettier, Black). Выполнить задание по настройке и использованию этих инструментов. Цель: Освоить инструменты для статического анализа и форматирования кода.	Выполнение практического задания.
6	Самостоятельная работа 6: Подготовка к зачету	Задание: Повторение пройденного материала, решение типовых задач и вопросов для подготовки к зачету. Цель: Повторить и обобщить пройденный материал для успешной сдачи зачета.	Тест.
6 семестр			
7	Самостоятельная	Задание: Изучить инструменты для тестирования ПО	Выполнение

	работа 7: Инструменты для тестирования ПО	(JUnit, pytest, Selenium). Выполнить задание по написанию тестов. Цель: Освоить инструменты для тестирования ПО и научиться писать тесты.	практического задания.
8	Самостоятельная работа 8: Дебаггеры и профилировщики	Задание: Изучить возможности отладчиков и профилировщиков (gdb, VisualVM, PyCharm Debugger). Подготовить отчет по использованию этих инструментов. Цель: Освоить инструменты для отладки и профилирования кода.	Выполнение практического задания.
9	Самостоятельная работа 9: Настройка CI/CD	Задание: Изучить инструменты CI/CD (Jenkins, GitHub Actions, GitLab CI). Настроить простой CI/CD пайплайн. Цель: Понять и применять инструменты для непрерывной интеграции и развертывания.	Выполнение практического задания.
10	Самостоятельная работа 10: Мониторинг и логирование	Задание: Изучить инструменты для мониторинга и логирования (Prometheus, Grafana, ELK Stack). Выполнить задание по настройке мониторинга и логирования. Цель: Освоить инструменты для мониторинга и логирования.	Выполнение практического задания.
11	Самостоятельная работа 11: Контейнеризация и оркестрация	Задание: Ознакомиться с основами Docker и Kubernetes. Выполнить практическое задание по созданию и управлению контейнерами. Цель: Понять основы контейнеризации и оркестрации и научиться применять их на практике.	Выполнение практического задания.
12	Самостоятельная работа 12: Итоговое занятие и подготовка к экзамену	Задание: Повторение и обобщение пройденного материала, решение типовых задач и вопросов для подготовки к экзамену. Цель: Повторить и обобщить пройденный материал для успешной сдачи экзамена.	Экзамен.

6.Оценочные материалы по дисциплине

Оценочные материалы находятся в документе «Оценочные материалы по дисциплине «Инструментальные средства разработки программного обеспечения».

7.Методические материалы для обучающихся по освоению дисциплины (модуля)

А) Рекомендации обучающемуся (студенту) по работе с конспектом после лекции

Какими бы замечательными качествами в области методики ни обладал лектор, какое бы большое значение на занятиях ни уделял лекции слушатель, глубокое понимание материала достигается только путем самостоятельной работы над ним. Самостоятельную работу следует начинать с доработки конспекта, желательно в тот же день, пока время не стерло содержание лекции из памяти (через 10 часов после лекции в памяти остается не более 30-40 % материала). С целью доработки необходимо в первую очередь прочитать записи, восстановить текст в памяти, а также исправить описки, расшифровать не принятые ранее сокращения, заполнить пропущенные места, понять текст, вникнуть в его смысл. Далее прочитать материал по рекомендуемой литературе, разрешая в ходе чтения, возникшие ранее затруднения, вопросы, а также дополнения и исправляя свои записи. Записи должны быть наглядными, для чего следует применять различные способы выделений. В ходе доработки конспекта углубляются, расширяются и закрепляются знания, а также дополняется, исправляется и совершенствуется конспект. Подготовленный конспект и рекомендуемая литература используется при подготовке к практическому занятию. Подготовка сводится к внимательному прочтению учебного материала, к выводу с карандашом в руках всех утверждений и формул, к решению примеров, задач, к ответам на вопросы, предложенные в конце лекции преподавателем или помещенные в рекомендуемой литературе. Примеры, задачи, вопросы по теме являются материалом самоконтроля. Непременным условием глубокого усвоения учебного материала является знание

основ, на которых строится изложение материала. Обычно преподаватель напоминает, какой ранее изученный материал и в какой степени требуется подготовить к очередному занятию. Эта рекомендация, как и требование систематической и серьезной работы над всем лекционным курсом, подлежит безусловному выполнению. Потери логической связи как внутри темы, так и между ними приводит к негативным последствиям: материал учебной дисциплины перестает основательно восприниматься, а творческий труд подменяется утомленным переписыванием. Обращение к ранее изученному материалу не только помогает восстановить в памяти известные положения, выводы, но и приводит разрозненные знания в систему, углубляет и расширяет их. Каждый возврат к старому материалу позволяет найти в нем что-то новое, переосмыслить его с иных позиций, определить для него наиболее подходящее место в уже имеющейся системе знаний. Неоднократное обращение к пройденному материалу является наиболее рациональной формой приобретения и закрепления знаний. Очень полезным, но, к сожалению, еще мало используемым в практике самостоятельной работы, является предварительное ознакомление с учебным материалом. Даже краткое, беглое знакомство с материалом очередной лекции дает многое. Обучающиеся (студенты) получают общее представление о её содержании и структуре, о главных и второстепенных вопросах, о терминах и определениях. Все это облегчает работу на лекции и делает ее целеустремленной.

Б) Рекомендации обучающемуся (студенту) по подготовке к занятиям семинарского типа

Обучающийся (студент) должен чётко уяснить, что именно с лекции начинается его подготовка к лабораторному/ практическому/ семинарскому/ методическому/ клиническому практическому занятию. Вместе с тем, лекция лишь организует мыслительную деятельность, но не обеспечивает глубину усвоения программного материала. При подготовке к такому виду занятий можно выделить 2 этапа:

1-й - организационный,

2-й - закрепление и углубление теоретических знаний.

На первом этапе обучающийся (студент) планирует свою самостоятельную работу, которая включает:

- уяснение задания на самостоятельную работу;
- подбор рекомендованной литературы;
- составление плана работы, в котором определяются основные пункты предстоящей подготовки.

Составление плана дисциплинирует и повышает организованность в работе. Второй этап включает непосредственную подготовку обучающегося (студента) к занятию. Начинать надо с изучения рекомендованной литературы. Необходимо помнить, что на лекции обычно рассматривается не весь материал, а только его часть. Остальная его часть восполняется в процессе самостоятельной работы. В связи с этим работа с рекомендованной литературой обязательна. Особое внимание при этом необходимо обратить на содержание основных положений и выводов, объяснение явлений и фактов, уяснение практического приложения рассматриваемых теоретических вопросов. В процессе этой работы обучающийся (студент) должен стремиться понять и запомнить основные положения рассматриваемого материала, примеры, поясняющие его, а также разобраться в иллюстративном материале. Заканчивать подготовку следует составлением плана (перечня основных пунктов) по изучаемому материалу (вопросу). Такой план позволяет составить концентрированное, сжатое представление по изучаемым вопросам. В процессе подготовки к семинарскому занятию рекомендуется взаимное обсуждение материала, во время которого закрепляются знания, а также приобретает практика в изложении и разъяснении полученных знаний, развивается речь. При необходимости следует обращаться за консультацией к преподавателю. Идя на консультацию, необходимо хорошо продумать вопросы, которые требуют разъяснения. В начале семинарского занятия обучающиеся (студента) под руководством преподавателя более глубоко осмысливают теоретические положения по теме занятия, раскрывают и объясняют основные явления и факты.

В процессе творческого обсуждения и дискуссии вырабатываются умения и навыки использовать приобретенные знания для решения практических задач.

В) Рекомендации по самостоятельной работе обучающегося (студента) над изучаемым материалом

Успешное освоение данного курса базируется на рациональном сочетании нескольких видов учебной деятельности - лекций, семинарских занятий, самостоятельной работы. При этом самостоятельную работу следует рассматривать одним из главных звеньев полноценного высшего образования, на которую отводится значительная часть учебного времени.

Самостоятельная работа студентов складывается из следующих составляющих:

- работа с основной и дополнительной литературой, с материалами интернета и конспектами лекций;
- внеаудиторная подготовка к контрольным работам, выполнение докладов, рефератов и курсовых работ;
- выполнение самостоятельных практических работ;
- подготовка к экзаменам (зачетам) непосредственно перед ними.

Для правильной организации работы необходимо учитывать порядок изучения разделов курса, находящихся в строгой логической последовательности. Поэтому хорошее усвоение одной части дисциплины является предпосылкой для успешного перехода к следующей. Задания, проблемные вопросы, предложенные для изучения дисциплины, в том числе и для самостоятельного выполнения, носят междисциплинарный характер и базируются, прежде всего, на причинно-следственных связях между компонентами окружающего нас мира. В течение семестра, необходимо подготовить рефераты (проекты) с использованием рекомендуемой основной и дополнительной литературы и сдать рефераты для проверки преподавателю. Важным составляющим в изучении данного курса является решение ситуационных задач и работа над проблемно-аналитическими заданиями, что предполагает знание соответствующей научной терминологии и т.д.

Для лучшего запоминания материала целесообразно использовать индивидуальные особенности и разные виды памяти: зрительную, слуховую, ассоциативную. Успешному запоминанию также способствует приведение ярких свидетельств и наглядных примеров. Учебный материал должен постоянно повторяться и закрепляться.

При выполнении докладов, творческих, информационных, исследовательских проектов особое внимание следует обращать на подбор источников информации и методику работы с ними.

Для успешной сдачи экзамена (зачета) рекомендуется соблюдать следующие правила:

1. Подготовка к экзамену (зачету) должна проводиться систематически, в течение всего семестра.
2. Интенсивная подготовка должна начаться не позднее, чем за месяц до экзамена.
3. Время непосредственно перед экзаменом (зачетом) лучше использовать таким образом, чтобы оставить последний день свободным для повторения курса в целом, для систематизации материала и доработки отдельных вопросов.

На экзамене высокую оценку получают студенты, использующие данные, полученные в процессе выполнения самостоятельных работ, а также использующие собственные выводы на основе изученного материала.

Учитывая значительный объем теоретического материала, студентам рекомендуется регулярное посещение и подробное конспектирование лекций.

8. ОСОБЕННОСТИ РЕАЛИЗАЦИИ ДИСЦИПЛИНЫ ДЛЯ ИНВАЛИДОВ И 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, прочитать 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, прочитать и оформить задание, в том числе записывая под диктовку);

- промежуточно-заочная аттестация проводится в письменной форме;

- обеспечивается наличие звукоусиливающей аппаратуры коллективного пользования, при необходимости поступающим предоставляется звукоусиливающая аппаратура индивидуального пользования;

- по желанию студента промежуточно-заочная 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 промежуточно-заочная аттестация проводится в устной форме.

Примечание:

а) Для обучающегося (бакалавра), осваивающего учебную дисциплину, обязательный компонент основной профессиональной образовательной программы высшего образования — программы бакалавриата по направлению подготовки **09.03.02 Информационные системы и технологии (направленность (профиль) «Информационные технологии»), форма обучения — очно-заочная), одобренной на заседании Учёного совета образовательной организации, утверждённой ректором Частного образовательного учреждения высшего образования «Высшая школа предпринимательства», по индивидуальному учебному плану (при наличии факта зачисления в образовательную организацию такого обучающегося (бакалавра)), Институт:**

- разрабатывает, согласовывает с участниками образовательных отношений и утверждает в установленном порядке согласно соответствующему локальному нормативному акту **индивидуальный учебный план** конкретного обучающегося (бакалавра) (*учебный план, обеспечивающий освоение конкретной основной образовательной программы высшего образования на основе индивидуализации её содержания с учётом особенностей и образовательных потребностей конкретного обучающегося (бакалавра)*);

- устанавливает для конкретного обучающегося (бакалавра) по индивидуальному учебному плану **одинаковые дидактические единицы** — элементы содержания учебного материала, изложенного в виде утверждённой в установленном образовательной организацией порядке согласно соответствующему локальному нормативному акту рабочей программы учебной дисциплины, обязательного компонента разработанной и реализуемой Институтom основной профессиональной образовательной программы высшего образования — программы

бакалавриата по направлению подготовки **09.03.02 Информационные системы и технологии** (направленность (профиль) «Информационные технологии»), форма обучения — очно-заочная), как и для обучающего (бакалавра), осваивающего основную образовательную программу высшего образования в учебной группе;

- определяет в индивидуальном учебном плане конкретного обучающегося (бакалавра) **объём учебной дисциплины** с указанием количества академических часов/ ЗЕТ, выделенных на его контактную работу (групповую и (или) индивидуальную работу) с руководящими и (или) научно-педагогическими работниками, реализующими основную образовательную программу высшего образования;

- определяет в индивидуальном учебном плане конкретного обучающегося (бакалавра) количество академических часов/ ЗЕТ по учебной дисциплине, выделенных на его самостоятельную работу (*при необходимости*).

б) Для обучающегося (бакалавра) с ограниченными возможностями здоровья и инвалида, осваивающего учебную дисциплину, обязательный компонент основной профессиональной образовательной программы высшего образования — программы бакалавриата по направлению подготовки 09.03.02 Информационные системы и технологии (направленность (профиль) «Информационные технологии»), форма обучения — очно-заочная), одобренной на заседании Учёного совета образовательной организации, утверждённой ректором Частного образовательного учреждения высшего образования «Высшая школа предпринимательства», (при наличии факта зачисления в образовательную организацию такого обучающегося (бакалавра) с учётом конкретной (конкретных) нозологии (нозологий)), Институт:

- разрабатывает, согласовывает с участниками образовательных отношений и утверждает в установленном порядке согласно соответствующему локальному нормативному акту **индивидуальный учебный план** конкретного обучающегося (бакалавра) с ограниченными возможностями здоровья/ инвалида (*при наличии факта зачисления в образовательную организацию такого обучающегося (бакалавра) с учётом конкретной (конкретных) нозологии (нозологий)*) (учебный план, обеспечивающий освоение конкретной основной образовательной программы высшего образования на основе индивидуализации её содержания с учётом особенностей и образовательных потребностей конкретного обучающегося (бакалавра));

- устанавливает для конкретного обучающегося (бакалавра) с ограниченными возможностями здоровья содержание образования (**одинаковые дидактические единицы** — элементы содержания учебного материала, как и для обучающего (бакалавра), осваивающего основную образовательную программу высшего образования в учебной группе) и условия организации обучения, изложенного в виде утверждённой в установленном Институте порядке согласно соответствующему локальному нормативному акту рабочей программы учебной дисциплины, обязательного компонента разработанной и реализуемой им адаптированной основной профессиональной образовательной программы высшего образования - программы бакалавриата по направлению подготовки **09.03.02 Информационные системы и технологии** (направленность (профиль) «Информационные технологии»), форма обучения — очно-заочная), а для инвалидов также в соответствии с индивидуальной программой реабилитации инвалида (для конкретного обучающегося (бакалавра) с ограниченными возможностями здоровья/ инвалида (*при наличии факта зачисления в образовательную организацию такого обучающегося (бакалавра) с учётом конкретной (конкретных) нозологии (нозологий)*);

- определяет в индивидуальном учебном плане конкретного обучающегося бакалавра) с ограниченными возможностями здоровья/ инвалида (*при наличии факта зачисления такого обучающегося (бакалавра) с учётом конкретной (конкретных) нозологии (нозологий)*) **объём учебной дисциплины** с указанием количества академических часов/ ЗЕТ, выделенных на его контактную работу (групповую и (или) индивидуальную работу) с руководящими и (или) научно-педагогическими работниками, реализующими основную образовательную программу высшего образования;

- определяет в индивидуальном учебном плане конкретного обучающегося (бакалавра) с ограниченными возможностями здоровья/ инвалида *(при наличии факта зачисления в образовательную организацию такого обучающегося (бакалавра) с учётом конкретной (конкретных) нозологии (нозологий))* количество академических часов/ ЗЕТ по учебной дисциплине, выделенных на его самостоятельную работу *(при необходимости)*.

9. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

9.1 Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины

Основная литература:

1. Мартин Р., Чистая архитектура. Искусство разработки программного обеспечения: книга / Р. Мартин. - СПб. : Питер, 2019. - 352 с. - ISBN 978-5-496-03181-7.
2. Орлов С. А., Технологии разработки программного обеспечения: учебник / С. А. Орлов. - СПб. : Питер, 2024. - ISBN 5-94723-820-9.
3. Ратушняк Г.Я., Технологии разработки и проектирования информационных систем. Часть 1 [Электронный ресурс] : учебное пособие / Г.Я. Ратушняк, А.Л. Золкин. - М. : Русайнс, 2022. - 201 с. - ISBN 978-5-4365-9893-2. - Режим доступа: <https://book.ru/book/945635>
4. Ратушняк Г.Я., Технологии разработки и проектирования информационных систем. Часть 2 [Электронный ресурс] : учебное пособие / Г.Я. Ратушняк, А.Л. Золкин. - М. : Русайнс, 2022. - 349 с. - ISBN 978-5-466-01439-6. - Режим доступа: <https://book.ru/book/946232>

Дополнительная литература:

1. Зубкова Т. М., Технология разработки программного обеспечения: учебное пособие / Т. М. Зубкова. - М. : Юрайт, 2023.
2. Письменский Г.И., Основы проектирования электронной информационно-образовательной среды [Электронный ресурс] : монография / Г.И. Письменский, В.В. Киселев, Л.В. Неровный, С.В. Сафонова. - М. : Русайнс, 2022. - 102 с. - ISBN 978-5-466-02119-6. - Режим доступа: <https://book.ru/book/947240>
3. Федоричев Л. А., Букунова О. В., Технология разработки программного обеспечения: учебное пособие / Л. А. Федоричев, О. В. Букунова. - СПб. : Лань, 2023.

9.2 Используемое программное обеспечение (комплект лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства в соответствии с п.4.3.2. ФГОС ВО 09.03.02):

1. Microsoft Windows 11 Pro или аналогичная ОС, включая дистрибутивы Linux, например Debian, Ubuntu, OpenSuse, в том числе отечественного производства, например ОС Astra Linux Common Edition (Разработчик: АО «НПО РусБИТех»), ОС «РОСА» (Разработчик: «НТИЦ ИТ РОСА»).
2. Microsoft Office 365 или аналогичный офисный пакет, например OpenOffice, LibreOffice, ONLYOFFICE, в том числе отечественного производства, например МойОфис (Разработчик: ООО «НОВЫЕ ОБЛАЧНЫЕ ТЕХНОЛОГИИ»).
3. Adobe Reader или аналогичный просмотрщик PDF, например Okular, Foxit Reader, в том числе отечественного производства, например Окуляр ГОСТ (Разработчик: ООО «Лаборатория 50»).
4. Google Chrome или аналогичный веб-браузер, например Microsoft Edge, Mozilla Firefox, в том числе отечественного производства, например Яндекс.Браузер (Разработчик: ООО «ЯНДЕКС»).
5. Microsoft Visual Studio Code или аналогичная IDE, например Sublime Text, Eclipse, в том числе отечественного производства
6. PyCharm / IntelliJ IDEA / CLion / WebStorm либо аналогичная IDE полного стека, в том числе отечественного производства

7. MySQL CE 8.0 / MySQL Workbench или аналогичные СУБД, например MS SQL, PostgreSQL, в том числе отечественного производства
8. Android Studio или аналогичная IDE для разработки мобильных приложений, в том числе отечественного производства
9. Figma или аналогичное ПО для подготовки макетов, например Penpot, Lunacy, в том числе отечественного производства

9.3 Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины (модуля) (в соответствии с п.4.3.4. ФГОС ВО 09.03.02)

1. Электронно-библиотечная система BOOK.RU [Электронный ресурс]. - Режим доступа: <https://book.ru/>
2. IntelliJ IDEA Documentation [Электронный ресурс]. - Режим доступа: <https://www.jetbrains.com/idea/documentation/>
3. Visual Studio Documentation [Электронный ресурс]. - Режим доступа: <https://docs.microsoft.com/en-us/visualstudio/>
4. Jenkins Documentation [Электронный ресурс]. - Режим доступа: <https://www.jenkins.io/doc/>
5. Docker Documentation [Электронный ресурс]. - Режим доступа: <https://docs.docker.com/>
6. Gradle User Manual [Электронный ресурс]. - Режим доступа: <https://docs.gradle.org/current/userguide/userguide.html>

9.4 Базы данных, информационно-справочные и поисковые системы (в соответствии с п.4.3.4. ФГОС ВО 09.03.02)

1. КонсультантПлюс: справочно-поисковая система [Электронный ресурс]. - <http://www.consultant.ru>
2. Мировая цифровая библиотека: <http://wdl.org/ru>
3. Научная электронная библиотека «Scopus»: <https://www.scopus.com>
4. Научная электронная библиотека ScienceDirect: <http://www.sciencedirect.com>
5. Научная электронная библиотека «eLIBRARY»: <https://elibrary.ru>
6. Портал «Гуманитарное образование» <http://www.humanities.edu.ru>
7. Федеральный портал «Российское образование» <http://www.edu.ru>
8. Федеральное хранилище «Единая коллекция цифровых образовательных ресурсов» <http://school-collection.edu.ru>
9. Поисковые системы Yandex, Rambler и др.
10. Электронная библиотека Российской Государственной Библиотеки (РГБ): <http://elibrary.rsl.ru>
11. Электронно-библиотечная система <http://www.sciteclibrary.ru>

10. Материально-техническое обеспечение дисциплины

Помещения для проведения всех видов работы, предусмотренных учебным планом, укомплектованы необходимым оборудованием и техническими средствами обучения.

Наименование помещений для проведения всех видов учебной деятельности, предусмотренной учебным планом, в том числе помещения для самостоятельной работы, с указанием перечня основного оборудования, учебно-наглядных пособий и используемого программного обеспечения	Адрес (местоположение) помещений для проведения всех видов учебной деятельности, предусмотренной учебным планом (в случае реализации образовательной программы в сетевой форме дополнительно указывается наименование организации, с которой заключен договор)
--	--

<p>Специализированная многофункциональная учебная аудитория для проведения учебных занятий лекционного типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, в том числе, для организации практической подготовки обучающихся, с перечнем основного оборудования:</p> <ul style="list-style-type: none"> - Столы для обучающихся; - Стулья для обучающихся; - Стол педагогического работника; - Стул педагогического работника; - Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; - Маркерная или меловая доска; - Проектор. 	<p>170001, Россия, город Тверь, улица Спартака, дом 26а</p>
<p>Специализированная многофункциональная учебная аудитория для проведения учебных занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, в том числе, для организации практической подготовки обучающихся, с перечнем основного оборудования:</p> <ul style="list-style-type: none"> - Столы для обучающихся; - Стулья для обучающихся; - Стол педагогического работника; - Стул педагогического работника; - Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; - Маркерная или меловая доска; - Проектор. 	<p>170001, Россия, город Тверь, улица Спартака, дом 26а</p>
<p>Помещение для самостоятельной работы обучающихся с перечнем основного оборудования:</p> <ul style="list-style-type: none"> - Столы для обучающихся; - Стулья для обучающихся; - Стол педагогического работника; - Стул педагогического работника; - Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную 	<p>170001, Россия, город Тверь, улица Спартака, дом 26а</p>

<p>информационно-образовательную среду лицензиата;</p> <ul style="list-style-type: none"> - Маркерная или меловая доска; - Проектор. 	
<p>Помещение для практических занятий на персональных компьютерах:</p> <ul style="list-style-type: none"> - Столы для обучающихся; - Стулья для обучающихся; - Стол педагогического работника; - Стул педагогического работника; - Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; - Ноутбуки с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; - Маркерная или меловая доска; - Проектор. 	<p>170001, Россия, город Тверь, улица Спартак, дом 26а</p>

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**



**Частное учреждение высшего образования
«Высшая школа предпринимательства (институт)»
(ЧУВО «ВШП»)**

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

по дисциплине

**Б1.В.06 «Инструментальные средства разработки программного
обеспечения»**

Направление подготовки: 09.03.02 Информационные системы и технологии

Направленность (профиль) программы бакалавриата

«Информационные технологии»

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО УЧЕБНОЙ ДИСЦИПЛИНЕ В РАМКАХ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОСВОЕНИЯ ОСНОВНОЙ ПРОФЕССИОНАЛЬНОЙ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Процесс освоения учебной дисциплины направлен на формирование у обучающихся следующих компетенций.

В результате освоения учебной дисциплины обучающийся должен демонстрировать следующие результаты обучения: УК-1, УК-2, ПК-1, ПК-2, ПК-3, ПК-4.

Код компетенции	Наименование компетенции	Индекс и наименование индикатора содержания компетенции	Дескрипторы – основные признаки освоения (показатели достижения результата)
УК-1	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	УК-1.1 Способен осуществлять поиск и критический анализ информации	Знать: <ul style="list-style-type: none"> - Основные источники информации для разработки программного обеспечения (далее ПО). - Методы критического анализа информации. Уметь: <ul style="list-style-type: none"> - Находить актуальную информацию в специализированных источниках. - Анализировать и оценивать достоверность и релевантность информации. Владеть: <ul style="list-style-type: none"> - Навыками эффективного поиска информации. - Навыками критического мышления при анализе информации.
		УК-1.2 Способен применять системный подход для решения поставленных задач	Знать: <ul style="list-style-type: none"> - Принципы системного подхода в разработке ПО. - Основные этапы системного анализа и проектирования. Уметь: <ul style="list-style-type: none"> - Применять системный подход для анализа задач. - Разрабатывать системные решения для поставленных задач. Владеть: <ul style="list-style-type: none"> - Навыками системного анализа проблем. - Навыками интеграции различных элементов системы в единое целое.
УК-2	Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1 Способен определять круг задач в рамках поставленной цели	Знать: <ul style="list-style-type: none"> - Методы декомпозиции целей на задачи. - Принципы приоритизации задач. Уметь: <ul style="list-style-type: none"> - Определять и формулировать задачи на основе целей проекта. - Устанавливать приоритеты для выполнения задач. Владеть: <ul style="list-style-type: none"> - Навыками декомпозиции целей на задачи. - Навыками управления задачами и ресурсами.

		УК-2.2 Способен выбирать оптимальные способы решения задач, исходя из правовых норм, ресурсов и ограничений	<p>Знать:</p> <ul style="list-style-type: none"> - Основные правовые нормы в области информационных технологий (далее ИТ). - Принципы оценки и управления ресурсами. <p>Уметь:</p> <ul style="list-style-type: none"> - Оценивать доступные ресурсы и ограничения. - Выбирать наилучшие способы решения задач с учетом правовых норм. <p>Владеть:</p> <ul style="list-style-type: none"> - Навыками оценки ресурсов и ограничений. - Навыками принятия оптимальных решений в рамках правовых норм.
ПК-1	Разработка и отладка программного кода	ПК-1.1 Способен разрабатывать программный код	<p>Знать:</p> <ul style="list-style-type: none"> - Основные языки программирования для разработки ПО. - Принципы написания качественного кода. <p>Уметь:</p> <ul style="list-style-type: none"> - Писать программный код на одном из изучаемых языков. - Разрабатывать алгоритмы и структуры данных для решения задач. <p>Владеть:</p> <ul style="list-style-type: none"> - Навыками написания и документирования кода. - Навыками использования инструментов разработки.
		ПК-1.2 Способен отлаживать программный код	<p>Знать:</p> <ul style="list-style-type: none"> - Основные методы и инструменты отладки кода. - Принципы устранения ошибок и багов в коде. <p>Уметь:</p> <ul style="list-style-type: none"> - Использовать отладчики и логи для выявления ошибок. - Исправлять выявленные ошибки и оптимизировать код. <p>Владеть:</p> <ul style="list-style-type: none"> - Навыками отладки и тестирования кода. - Навыками применения инструментов для отладки.
ПК-2	Проверка работоспособности и рефакторинг кода программного обеспечения	ПК-2.1 Способен проверять работоспособность программного кода	<p>Знать:</p> <ul style="list-style-type: none"> - Принципы и методы тестирования кода. - Основные виды тестов (модульные, интеграционные, системные). <p>Уметь:</p> <ul style="list-style-type: none"> - Писать и выполнять тесты для проверки кода. - Использовать тестовые фреймворки для автоматизации тестирования. <p>Владеть:</p> <ul style="list-style-type: none"> - Навыками написания и выполнения тестов. - Навыками применения инструментов для тестирования.

		<p>ПК-2.2</p> <p>Способен рефакторить программный код</p>	<p>Знать:</p> <ul style="list-style-type: none"> - Принципы и методы рефакторинга. - Основные техники улучшения структуры кода. <p>Уметь:</p> <ul style="list-style-type: none"> - Анализировать код на предмет улучшений. - Применять техники рефакторинга для повышения качества кода. <p>Владеть:</p> <ul style="list-style-type: none"> - Навыками рефакторинга кода. - Навыками оценки и улучшения структуры кода.
ПК-3	Интеграция программных модулей и компонентов и проверка работоспособности выпусков программного продукта	<p>ПК-3.1</p> <p>Способен интегрировать программные модули и компоненты</p>	<p>Знать:</p> <ul style="list-style-type: none"> - Принципы интеграции модулей и компонентов. - Методы и инструменты для интеграции. <p>Уметь:</p> <ul style="list-style-type: none"> - Интегрировать модули и компоненты в единое приложение. - Проверять корректность интеграции и совместимость компонентов. <p>Владеть:</p> <ul style="list-style-type: none"> - Навыками интеграции программных модулей. - Навыками использования инструментов для интеграции.
		<p>ПК-3.2</p> <p>Способен проверять работоспособность выпусков программного продукта</p>	<p>Знать:</p> <ul style="list-style-type: none"> - Методы и инструменты для проверки выпусков ПО. - Принципы контроля качества выпусков. <p>Уметь:</p> <ul style="list-style-type: none"> - Проводить тестирование и верификацию выпусков ПО. - Оценивать стабильность и производительность программного продукта. <p>Владеть:</p> <ul style="list-style-type: none"> - Навыками тестирования и верификации выпусков ПО. - Навыками применения инструментов контроля качества.
ПК-4	Разработка требований и проектирование программного обеспечения	<p>ПК-4.1</p> <p>Способен разрабатывать требования к программному обеспечению</p>	<p>Знать:</p> <ul style="list-style-type: none"> - Принципы и методы анализа требований. - Техники документирования требований. <p>Уметь:</p> <ul style="list-style-type: none"> - Собирать и анализировать требования к ПО. - Документировать требования в виде спецификаций. <p>Владеть:</p> <ul style="list-style-type: none"> - Навыками анализа и сбора требований. - Навыками создания и поддержки документации по требованиям.

		ПК-4.2 Способен проектировать программное обеспечение	Знать: <ul style="list-style-type: none"> - Принципы и методы проектирования ПО. - Основные подходы к архитектурному дизайну. Уметь: <ul style="list-style-type: none"> - Разрабатывать архитектуру и дизайн ПО. - Использовать UML и другие инструменты для проектирования. Владеть: <ul style="list-style-type: none"> - Навыками проектирования программного обеспечения. - Навыками применения инструментов для проектирования.
--	--	--	---

КРИТЕРИИ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ

(признак, на основании которого, проводится оценка по выбранному показателю)

Показатель оценивания компетенций	Результат обучения	Критерии оценивания компетенций
Высокий уровень (отлично)	Знать	Обучающийся продемонстрировал: глубокие исчерпывающие знания и понимание учебного материала; содержательные, полные, правильные и конкретные ответы на все вопросы, включая дополнительные; свободное владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины.
	Уметь	Обучающийся продемонстрировал: понимание учебного материала; умение свободно решать практические задания (ситуационные задачи), которые следует выполнить или описание результата, который нужно получить и др.; логически последовательные, содержательные, полные, правильные и конкретные ответы (решения) на все поставленные задания (вопросы), включая дополнительные; свободное владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины.
	Владеть	Обучающийся продемонстрировал: понимание учебного материала; умение свободно решать комплексные практические задания (решения задач по нестандартным ситуациям); логически последовательные, полные, правильные и конкретные ответы в ходе защиты задания, включая дополнительные уточняющие вопросы (задания); свободное владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины.
Средний уровень (хорошо)	Знать	Обучающийся продемонстрировал: твердые и достаточно полные знания учебного материала; правильное понимание сущности и взаимосвязи рассматриваемых процессов и явлений; последовательные, правильные, конкретные ответы на поставленные вопросы при свободном устранении замечаний по отдельным вопросам; достаточное владение литературой, рекомендованной учебной программой дисциплины
	Уметь	Обучающийся продемонстрировал: понимание учебного материала; логически последовательные, правильные и конкретные ответы (решения) на основные задания (вопросы), включая дополнительные; устранение замечаний по отдельным элементам задания (вопроса); владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины
	Владеть	Обучающийся продемонстрировал: понимание учебного материала; продемонстрировал логически последовательные,

		достаточно полные, правильные ответы, включая дополнительные; самостоятельно устранил замечания по отдельным элементам задания (вопроса); владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины
Достаточный уровень (удовлетворительно)	Знать	Обучающийся продемонстрировал: твердые знания и понимание основного учебного материала; правильные, без грубых ошибок, ответы на поставленные вопросы при устранении неточностей и несущественных ошибок в освещении отдельных положений при наводящих вопросах преподавателя; недостаточно полное владение литературой, рекомендованной учебной программой дисциплины
	Уметь	Обучающийся продемонстрировал: понимание основного учебного материала; правильные, без грубых ошибок, ответы (решения) на основные задания (вопросы), включая дополнительные, устранение, при наводящих вопросах преподавателя, замечаний по отдельным элементам задания (вопроса); недостаточно полное владение литературой, рекомендованной учебной программой дисциплины
	Владеть	Обучающийся понимание основного учебного материала; без грубых ошибок дал ответы на поставленные вопросы при устранении неточностей и ошибок в решениях в ходе защиты задания (проекта, портфолио) при наводящих вопросах преподавателя; недостаточно полное владение литературой, рекомендованной учебной программой дисциплины

ОПИСАНИЕ ШКАЛ ОЦЕНИВАНИЯ

При проведении промежуточной аттестации в ЧУВО «ВШП» используются традиционные формы аттестации:

Форма промежуточной аттестации	Шкала оценивания
ЗАЧЕТ	«зачтено», «незачтено»
ЭКЗАМЕН	«отлично», «хорошо», «удовлетворительно», «неудовлетворительно»

КРИТЕРИИ И ПРОЦЕДУРЫ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ НА РАЗЛИЧНЫХ ЭТАПАХ ИХ ФОРМИРОВАНИЯ

Для оценивания результатов обучения в виде **ЗНАНИЙ** используются следующие процедуры и технологии:

- тестирование.

Для оценивания результатов обучения в виде **УМЕНИЙ и ВЛАДЕНИЙ** используются следующие процедуры и технологии:

- устный или письменный ответ на вопрос.
- практические задания, включающие одну или несколько задач (вопросов) в виде краткой формулировки действий (комплекса действий), которые следует выполнить, или описать результат, который нужно получить.

Критерии оценивания результата обучения по дисциплине (модулю)

Результат обучения по дисциплине (модулю)	ШКАЛА ОЦЕНИВАНИЯ				Процедуры оценивания
	«отлично»	«хорошо»	«удовлетворительно»	«неудовлетворительно»	
<u>УК-1.</u> <u>УК-2.</u> <u>ПК-1.</u> <u>ПК-2.</u>	Обучаемый продемонстрировал: глубокие исчерпывающие	Обучаемый продемонстрировал: твердые и достаточно	Обучаемый продемонстрировал: твердые знания и понимание	Обучаемый продемонстрировал: неправильные ответы на основные вопросы;	Тестовые задания

<u>ПК-3</u> <u>ПК-4</u> <u>Знать:</u>	знания и понимание учебного материала; содержательные, полные, правильные и конкретные ответы на все вопросы, включая дополнительные; свободное владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины.	полные знания учебного материала; правильное понимание сущности и взаимосвязи рассматриваемых процессов и явлений; последовательные, правильные, конкретные ответы на поставленные вопросы при свободном устранении замечаний по отдельным вопросам; достаточное владение литературой.	основного учебного материала; правильные, без грубых ошибок, ответы на поставленные вопросы при устранении неточностей и несущественных ошибок в освещении отдельных положений при наводящих вопросах преподавателя; недостаточно полное владение литературой, рекомендованной учебной программой дисциплины.	грубые ошибки в ответах; непонимание сущности излагаемых вопросов; неуверенные и неточные ответы на дополнительные вопросы; не владеет основной литературой, рекомендованной учебной программой дисциплины.	
<u>УК-1</u> <u>УК-2</u> <u>ПК-1</u> <u>ПК-2</u> <u>ПК-3</u> <u>ПК-4</u> <u>Уметь:</u>	Обучаемый продемонстрировал: понимание учебного материала, содержательные, полные, правильные и конкретные ответы на все поставленные вопросы, включая дополнительные; свободное владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины	Обучаемый продемонстрировал: понимание учебного материала; логически последовательные, правильные и конкретные ответы на основные задания/вопросы, включая дополнительные; устранение замечаний по отдельным элементам задания; владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины	Обучаемый продемонстрировал: понимание основного учебного материала; правильные, без грубых ошибок, ответы на основные вопросы, включая дополнительные, при устранении, при наводящих вопросах преподавателя, замечаний по отдельным элементам задания; недостаточное полное владение литературой, рекомендованной учебной программой дисциплины	Обучаемый продемонстрировал: непонимание основного учебного материала; не дал правильные ответы на основные вопросы, включая дополнительные; не устранил, при наводящих вопросах преподавателя, замечания и грубые ошибки по вопросу; не владеет основной литературой, рекомендованной учебной программой дисциплины	Вопросы Практические задания
<u>УК-1</u> <u>УК-2</u> <u>ПК-1</u> <u>ПК-2</u> <u>ПК-3</u> <u>ПК-4</u> <u>Владеть:</u>	Обучаемый продемонстрировал: понимание учебного материала; правильные и конкретные ответы, включая дополнительные и уточняющие вопросы;	Обучаемый продемонстрировал: понимание учебного материала; продемонстрировал логически последовательные, достаточно полные, верные ответы;	Обучаемый продемонстрировал: понимание основного учебного материала; без грубых ошибок дал ответы на поставленные вопросы, в том числе при наводящих вопросах	Обучаемый продемонстрировал: непонимание основного учебного материала; дал неправильные ответы на поставленные вопросы; не владеет основной учебной литературой, рекомендованной	Вопросы Практические задания

	свободное владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины	самостоятельно устранил замечания по отдельным элементам; владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины	преподавателя; недостаточно полное владение литературой, рекомендованной учебной программой дисциплины	учебной программой дисциплины	
--	---	---	--	-------------------------------	--

1. Оценочные материалы для самостоятельной работы обучающихся (студентов)

1.1 Доклад

5 семестр

Доклад позволит студентам углубиться в инструментальные средства разработки программного обеспечения, а также понять их важность и применение в процессе разработки ПО. Доклад покрывает компетенции УК-1.1, УК-1.2, УК-2.1, УК-2.2, ПК-1.1, ПК-1.2, ПК-2.1, ПК-2.2, ПК-3.1, ПК-3.2, ПК-4.1, ПК-4.2.

Примерная тематика докладов:

1. **Введение в инструментальные средства разработки ПО:** Основные виды инструментов для разработки ПО, введение в IDE и текстовые редакторы.
2. **Интегрированные среды разработки (IDE):** Обзор популярных IDE (IntelliJ IDEA, Visual Studio, Eclipse), настройка и использование.
3. **Системы контроля версий (Git):** Основы работы с Git, основные команды и концепции (commit, branch, merge), инструменты для визуализации и управления репозиториями (GitKraken, Sourcetree).
4. **Инструменты для управления зависимостями:** Использование инструментов для управления зависимостями (Maven, Gradle, npm).
5. **Линтеры и форматтеры кода:** Инструменты для статического анализа кода и его форматирования, использование линтеров (ESLint, Pylint) и форматтеров (Prettier, Black).
6. **Инструменты для тестирования ПО:** Обзор инструментов для юнит-тестирования и интеграционного тестирования (JUnit, pytest, Selenium).

1.2 Доклад

6 семестр

Доклад позволит студентам углубиться в инструментальные средства разработки программного обеспечения, а также понять их важность и применение в процессе разработки ПО. Доклад покрывает компетенции УК-1.1, УК-1.2, УК-2.1, УК-2.2, ПК-1.1, ПК-1.2, ПК-2.1, ПК-2.2, ПК-3.1, ПК-3.2, ПК-4.1, ПК-4.2.

Примерная тематика докладов:

1. **Дебаггеры и профилировщики:** Использование отладчиков и профилировщиков для анализа и оптимизации кода, примеры инструментов (gdb, VisualVM, PyCharm Debugger).
2. **Инструменты для CI/CD:** Введение в инструменты непрерывной интеграции и развертывания, обзор популярных инструментов (Jenkins, GitHub Actions, GitLab CI).
3. **Мониторинг и логирование:** Инструменты для мониторинга и логирования (Prometheus, Grafana, ELK Stack).
4. **Контейнеризация и оркестрация:** Введение в контейнеризацию с Docker и оркестрацию с Kubernetes.
5. **Современные инструменты и технологии разработки ПО:** Обзор современных инструментов и технологий, применяемых в разработке ПО, их преимущества и вызовы.
6. **Будущее инструментальных средств разработки ПО:** Тенденции и перспективы в области инструментальных средств разработки ПО, влияние новых технологий на процесс разработки.

Цель написания докладов: Углубить понимание и критическое осмысление роли инструментальных средств разработки программного обеспечения в процессе создания ПО, развивая аналитические и научные навыки студентов.

Структура доклада:

1. Введение

- Краткое описание темы и целей доклада.
- Актуальность темы.

2. Основная часть

- Теоретические основы темы.
- История и эволюция (если применимо).
- Применение в современной экономике/бизнесе.
- Примеры и кейсы.
- Проблемы и вызовы.
- Перспективы и тенденции развития.

3. Заключение

- Выводы по результатам исследования.
- Значение инструментальных средств разработки ПО для современных информационных систем.

4. Список использованных источников

- Перечень использованной литературы и интернет-ресурсов.

Критерии оценивания:

1. Структура и логика изложения (20%)

- Четкая структура работы (введение, основная часть, заключение).
- Логичность и последовательность изложения материала.

2. Содержание (40%)

- Полнота раскрытия темы.
- Описание основных инструментов и методов разработки ПО.
- Анализ современных тенденций.
- Примеры применения инструментальных средств разработки в реальных проектах.

3. Аналитическая часть (20%)

- Глубина анализа роли инструментальных средств разработки ПО в процессе создания ПО.
- Наличие собственных выводов и оценок.

4. Оформление (10%)

- Соответствие требованиям к оформлению докладов (шрифт, отступы, заголовки и т.д.).
- Корректное оформление ссылок и списка литературы.

5. Язык и стиль (10%)

- Грамотность и точность изложения.
- Научный стиль текста.

Требования к объему:

Объем доклада должен составлять 8-12 страниц печатного текста (шрифт Times New Roman, размер 12, интервал 1.5, поля 2 см со всех сторон).

2. Оценочные материалы для оценки текущей аттестации обучающихся (студентов)

2.1 Тестовые задания для текущего контроля успеваемости в виде ЗНАНИЙ

В тестовом задании вопросы, которые имеют закрытый характер.

Правильные ответы выделены знаком +.

5 семестр

1. Какие основные функции выполняет интегрированная среда разработки (IDE)? (ПК-1.1)
 - Только редактирование кода.
 - Управление версиями программного обеспечения.
 - Создание графического интерфейса пользователя.
 - Обеспечивает автодополнение кода, отладку, тестирование и рефакторинг. +
2. Что такое система контроля версий и какие задачи она решает? (УК-1.1)
 - Используется только для компиляции кода.
 - Управляет зависимостями проекта.
 - Только для тестирования приложений.
 - Управляет изменениями в коде, позволяет отслеживать историю изменений и сотрудничать с командой. +
3. Какие команды используются в Git для работы с репозиторием? (ПК-1.1)
 - create, edit, delete, compile, run.
 - begin, end, save, load, share.
 - init, clone, add, commit, push, pull. +
 - start, stop, restart, delete, update.
4. Что такое линтер и какова его основная цель? (ПК-1.2)
 - Среда для написания кода.
 - Инструмент для выполнения юнит-тестов.
 - Инструмент для управления зависимостями.
 - Инструмент для статического анализа кода, помогающий находить ошибки и нарушения стиля. +
5. Какие возможности предоставляет Docker для разработки ПО? (ПК-3.1)
 - Автоматизация тестирования.
 - Только для написания кода.
 - Создание изолированных контейнеров для приложений, упрощение развертывания и масштабирования. +
 - Управление версиями программного обеспечения.
6. В чем заключается основная функция Jenkins в процессе разработки ПО? (ПК-3.1)
 - Анализ кода.
 - Управление зависимостями.
 - Только для написания кода.
 - Автоматизация процессов сборки, тестирования и развертывания приложений. +
7. Какую роль выполняет Maven в проекте? (ПК-3.1)
 - Управляет зависимостями и автоматизирует процесс сборки. +
 - Управляет версиями.
 - Только для тестирования.
 - Редактирует код.
8. Какие задачи решает Gradle в разработке ПО? (УК-1.2)
 - Управление версиями.
 - Автоматизация сборки, управление зависимостями, поддержка многомодульных проектов. +
 - Только написание кода.
 - Отладка приложений.
9. Что такое CI/CD и какие инструменты для этого используются? (ПК-2.1)
 - Непрерывная интеграция и развертывание; инструменты: Jenkins, GitLab CI. +
 - Только написание кода.
 - Управление зависимостями.
 - Статический анализ кода.
10. Какие преимущества предоставляет использование линтеров и статических анализаторов? (ПК-1.2)

- Только тестирование.
 - Управление зависимостями.
 - Помогают выявлять синтаксические и логические ошибки, нарушения стиля кода до выполнения программы. +
 - Редактирование кода.
11. Какие возможности предоставляет Visual Studio для разработчиков? (ПК-1.1)
- Анализ кода.
 - Только компиляция кода.
 - Управление версиями.
 - Средства для написания кода, отладки, тестирования, управления проектами, интеграции с системами контроля версий. +
12. Какие методы используются для разработки требований к программному обеспечению? (ПК-4.1)
- Интервью, анкеты, анализ документов, наблюдение, прототипирование и рабочие совещания. +
 - Только тестирование.
 - Управление зависимостями.
 - Рефакторинг кода.
13. Какие аспекты необходимо учитывать при проектировании программного обеспечения? (ПК-4.1)
- Требования пользователя, архитектура системы, масштабируемость, безопасность, производительность и поддерживаемость. +
 - Только написание кода.
 - Управление версиями.
 - Автоматизация тестирования.

Критерии оценки результатов теста

1. **"Неудовлетворительно" (0-39%)**
 - Студент ответил правильно на менее 40% вопросов.
 - Значительные пробелы в знаниях по большинству тем.
 - Неправильное понимание ключевых понятий и принципов.
 - Неспособность применить теоретические знания на практике.
2. **"Удовлетворительно" (40-59%)**
 - Студент ответил правильно на 40-59% вопросов.
 - Основные понятия и принципы поняты частично, есть ошибки в ответах.
 - Знания по большинству тем на базовом уровне, недостаточная глубина понимания.
 - Частичная способность применять теоретические знания на практике, нужны дополнительные разъяснения.
3. **"Хорошо" (60-79%)**
 - Студент ответил правильно на 60-79% вопросов.
 - Хорошее понимание ключевых понятий и принципов, незначительные ошибки.
 - Знания по всем темам на достаточном уровне, однако есть некоторые пробелы.
 - Способность применять теоретические знания на практике, но требуется улучшение точности и уверенности.
4. **"Отлично" (80-100%)**
 - Студент ответил правильно на 80-100% вопросов.
 - Полное и правильное понимание всех ключевых понятий и принципов.
 - Глубокие знания по всем темам, минимальные или отсутствующие ошибки.
 - Высокий уровень способности применять теоретические знания на практике, демонстрация уверенности и точности в ответах.

2.2 Тестовые задания для текущего контроля успеваемости в виде ЗНАНИЙ

В тестовом задании вопросы, которые имеют закрытый характер.

Правильные ответы выделены знаком +.

6 семестр

1. Какие функции выполняет Prometheus в проекте? (ПК-3.2)
 - Управление версиями.
 - Только для написания кода.
 - Сбор и хранение метрик системы, мониторинг производительности приложений. +
 - Автоматизация тестирования.
2. Какую роль играет управление конфигурацией в процессе разработки ПО? (УК-2.1)
 - Включает идентификацию конфигурационных единиц, контроль изменений, отчетность по конфигурациям, планирование и выполнение релизов. +
 - Только написание кода.
 - Управление версиями.
 - Автоматизация тестирования.
3. Что означает термин "оркестрация контейнеров" и какие задачи он решает? (УК-1.2)
 - Управление версиями.
 - Автоматизация тестирования.
 - Управление и координация работы множества контейнеров, включая их развертывание, масштабирование и мониторинг. +
 - Только написание кода.
4. Какие инструменты используются для логирования и мониторинга в ПО? (ПК-3.2)
 - Только для написания кода.
 - Управление версиями.
 - ELK Stack (Elasticsearch, Logstash, Kibana), Prometheus, Grafana. +
 - Анализ кода.
5. Какие возможности предоставляет GitLab CI и как он используется? (ПК-2.1)
 - Только написание кода.
 - Управление зависимостями.
 - Редактирование кода.
 - Возможности для автоматизации CI/CD процессов, включая сборку, тестирование и развертывание ПО. +
6. Как можно улучшить качество кода с помощью рефакторинга? (ПК-2.2)
 - Управление зависимостями.
 - Автоматизация сборки.
 - Только тестирование.
 - Рефакторинг улучшает структуру кода без изменения его внешнего поведения, что делает его более понятным и поддерживаемым. +
7. Какие задачи решает проектирование архитектуры ПО? (ПК-4.2)
 - Определение структуры системы, взаимодействие между компонентами, выбор технологий и обеспечение масштабируемости и безопасности. +
 - Только написание кода.
 - Управление версиями.
 - Автоматизация сборки.
8. Какую роль играет проектирование интерфейсов в разработке ПО? (ПК-4.2)

- Проектирование интерфейсов определяет, как компоненты системы будут взаимодействовать друг с другом, что обеспечивает модульность и гибкость системы. +
 - Только тестирование.
 - Управление зависимостями.
 - Редактирование кода.
9. Какие задачи решает интеграция с системами CI/CD в процессе разработки ПО? (ПК-2.1)
- Автоматизация сборки, тестирования и развертывания, ускорение выпуска новых версий, повышение качества ПО. +
 - Только написание кода.
 - Управление зависимостями.
 - Рефакторинг кода.
10. Какие принципы следует учитывать при выборе оптимальных способов решения задач? (УК-2.2)
- Только написание кода.
 - Соответствие правовым нормам, доступность ресурсов, время выполнения задачи и минимизация рисков. +
 - Управление версиями.
 - Автоматизация тестирования.
11. Как правильно выбирать способы решения задач с учетом правовых норм? (УК-2.2)
- Анализ правовых требований, консультации с юридическими специалистами, обеспечение соответствия законодательству и внутренним политикам компании. +
 - Только тестирование.
 - Управление зависимостями.
 - Рефакторинг кода.

Критерии оценки результатов теста

5. **"Неудовлетворительно" (0-39%)**
- Студент ответил правильно на менее 40% вопросов.
 - Значительные пробелы в знаниях по большинству тем.
 - Неправильное понимание ключевых понятий и принципов.
 - Неспособность применить теоретические знания на практике.
6. **"Удовлетворительно" (40-59%)**
- Студент ответил правильно на 40-59% вопросов.
 - Основные понятия и принципы поняты частично, есть ошибки в ответах.
 - Знания по большинству тем на базовом уровне, недостаточная глубина понимания.
 - Частичная способность применять теоретические знания на практике, нужны дополнительные разъяснения.
7. **"Хорошо" (60-79%)**
- Студент ответил правильно на 60-79% вопросов.
 - Хорошее понимание ключевых понятий и принципов, незначительные ошибки.
 - Знания по всем темам на достаточном уровне, однако есть некоторые пробелы.
 - Способность применять теоретические знания на практике, но требуется улучшение точности и уверенности.
8. **"Отлично" (80-100%)**
- Студент ответил правильно на 80-100% вопросов.
 - Полное и правильное понимание всех ключевых понятий и принципов.
 - Глубокие знания по всем темам, минимальные или отсутствующие ошибки.
 - Высокий уровень способности применять теоретические знания на практике, демонстрация уверенности и точности в ответах.

2.3 Вопросы для текущего контроля успеваемости в виде УМЕНИЙ

5 семестр

1. Какие основные функции выполняет интегрированная среда разработки (IDE)? (ПК-1.1)
 - Правильный ответ: Интегрированная среда разработки (IDE) обеспечивает автодополнение кода, отладку, тестирование, рефакторинг, управление проектами и интеграцию с системами контроля версий.
2. Что такое система контроля версий и какие задачи она решает? (УК-1.1)
 - Правильный ответ: Система контроля версий управляет изменениями в коде, позволяет отслеживать историю изменений, возвращаться к предыдущим версиям и сотрудничать с командой.
3. Какие команды используются в Git для работы с репозиторием? (ПК-1.1)
 - Правильный ответ: Основные команды Git включают init (инициализация репозитория), clone (клонирование репозитория), add (добавление изменений), commit (фиксация изменений), push (отправка изменений), pull (получение изменений).
4. Какие преимущества предоставляет использование линтеров и статических анализаторов? (ПК-1.2)
 - Правильный ответ: Линтеры и статические анализаторы помогают выявлять синтаксические и логические ошибки, нарушения стиля кода до выполнения программы, что повышает качество и поддерживаемость кода.
5. Какую роль выполняет Maven в проекте? (ПК-3.1)
 - Правильный ответ: Maven управляет зависимостями, автоматизирует процесс сборки, обеспечивает стандартизацию проекта и упрощает управление конфигурацией.
6. Какие задачи решает Gradle в разработке ПО? (УК-1.2)
 - Правильный ответ: Gradle автоматизирует сборку, управляет зависимостями, поддерживает многомодульные проекты и интеграцию с различными инструментами разработки.
7. Что такое CI/CD и какие инструменты для этого используются? (ПК-2.1)
 - Правильный ответ: CI/CD (непрерывная интеграция и развертывание) автоматизирует процессы сборки, тестирования и развертывания; популярные инструменты включают Jenkins, GitLab CI.
8. Какие преимущества предоставляет Docker для разработки ПО? (ПК-3.1)
 - Правильный ответ: Docker создает изолированные контейнеры для приложений, упрощает развертывание, масштабирование и обеспечивает консистентность среды разработки и производства.
9. Какие возможности предоставляет Visual Studio для разработчиков? (ПК-1.1)
 - Правильный ответ: Visual Studio предоставляет средства для написания кода, отладки, тестирования, управления проектами и интеграции с системами контроля версий.

Критерии оценки ответов на вопросы

- "Отлично" (5 баллов)
 - Критерии:
 - Полное и точное объяснение вопроса.
 - Ответ включает все ключевые аспекты и детали.
 - Примеры, если требуются, приведены и правильно объяснены.

- Ответ демонстрирует глубокое понимание темы.
- **"Хорошо" (4 балла)**
 - **Критерии:**
 - Корректное объяснение вопроса.
 - Ответ охватывает основные аспекты, но может отсутствовать незначительная деталь или пример.
 - Демонстрируется хорошее, но не полное понимание темы.
- **"Удовлетворительно" (3 балла)**
 - **Критерии:**
 - Общее представление о вопросе.
 - Ответ включает основные аспекты, но содержит неточности или пропуски.
 - Примеры, если требуются, могут отсутствовать или быть неверно объяснены.
 - Демонстрируется базовое понимание темы.
- **"Неудовлетворительно" (2 балла)**
 - **Критерии:**
 - Некорректное или неполное объяснение вопроса.
 - Отсутствие ключевых аспектов и деталей.
 - Примеры, если требуются, отсутствуют или приведены неверные.
 - Ответ демонстрирует недостаточное понимание темы.

2.4 Вопросы для текущего контроля успеваемости в виде УМЕНИЙ

6 семестр

1. Какие функции выполняет Prometheus в проекте? (ПК-3.2)
 - Правильный ответ: Prometheus собирает и хранит метрики системы, мониторит производительность приложений, предоставляет алертинг и визуализацию данных.
2. Какие возможности предоставляет GitLab CI и как он используется? (ПК-2.1)
 - Правильный ответ: GitLab CI предоставляет возможности для автоматизации CI/CD процессов, включая сборку, тестирование и развертывание ПО; используется для непрерывной интеграции и развертывания.
3. Как можно улучшить качество кода с помощью рефакторинга? (ПК-2.2)
 - Правильный ответ: Рефакторинг улучшает структуру кода без изменения его внешнего поведения, что делает код более понятным, поддерживаемым и снижает технический долг.
4. Какие задачи решает проектирование архитектуры ПО? (ПК-4.2)
 - Правильный ответ: Проектирование архитектуры ПО определяет структуру системы, взаимодействие между компонентами, выбор технологий, обеспечение масштабируемости и безопасности.
5. Какую роль играет проектирование интерфейсов в разработке ПО? (ПК-4.2)
 - Правильный ответ: Проектирование интерфейсов определяет, как компоненты системы будут взаимодействовать друг с другом, что обеспечивает модульность, гибкость и совместимость системы.
6. Какие принципы следует учитывать при выборе оптимальных способов решения задач? (УК-2.2)
 - Правильный ответ: Следует учитывать соответствие правовым нормам, доступность ресурсов, время выполнения задачи, минимизацию рисков и обеспечение качества результата.
7. Как правильно выбирать способы решения задач с учетом правовых норм? (УК-2.2)

- Правильный ответ: Необходимо анализировать правовые требования, консультироваться с юридическими специалистами, обеспечивать соответствие законодательству и внутренним политикам компании.
- 8. Какие методы используются для разработки требований к программному обеспечению? (ПК-4.1)
 - Правильный ответ: Интервью, анкеты, анализ документов, наблюдение, прототипирование и рабочие совещания используются для сбора и уточнения требований к ПО.
- 9. Какие аспекты необходимо учитывать при проектировании программного обеспечения? (ПК-4.1)
 - Правильный ответ: Требования пользователя, архитектура системы, масштабируемость, безопасность, производительность и поддерживаемость.

Критерии оценки ответов на вопросы

- **"Отлично" (5 баллов)**
 - **Критерии:**
 - Полное и точное объяснение вопроса.
 - Ответ включает все ключевые аспекты и детали.
 - Примеры, если требуются, приведены и правильно объяснены.
 - Ответ демонстрирует глубокое понимание темы.
- **"Хорошо" (4 балла)**
 - **Критерии:**
 - Корректное объяснение вопроса.
 - Ответ охватывает основные аспекты, но может отсутствовать незначительная деталь или пример.
 - Демонстрируется хорошее, но не полное понимание темы.
- **"Удовлетворительно" (3 балла)**
 - **Критерии:**
 - Общее представление о вопросе.
 - Ответ включает основные аспекты, но содержит неточности или пропуски.
 - Примеры, если требуются, могут отсутствовать или быть неверно объяснены.
 - Демонстрируется базовое понимание темы.
- **"Неудовлетворительно" (2 балла)**
 - **Критерии:**
 - Некорректное или неполное объяснение вопроса.
 - Отсутствие ключевых аспектов и деталей.
 - Примеры, если требуются, отсутствуют или приведены неверные.
 - Ответ демонстрирует недостаточное понимание темы.

2.5 Задачи на соответствие понятий для текущего контроля успеваемости в виде ВЛАДЕНИЙ

Правильные ответы расположены в таблицах друг напротив друга, во время тестирования предполагается что порядок данных в рамках каждого столбца будет случайным.

Задача 1. Соотнесите типы инструментов с их функциями (УК-1.1, УК-1.2)

Тип инструмента	Функция
A - IDE	1 - Написание и отладка кода

В - Линтер	2 - Статический анализ кода на наличие ошибок
С - Система контроля версий	3 - Управление изменениями кода
Д - Текстовый редактор	4 - Редактирование текста и кода

Правильный ответ: А-1, В-2, С-3, D-4

Задача 2. Соотнесите команды Git с их функциями (ПК-1.1, ПК-1.2)

Команда	Функция
А - init	1 - Инициализация нового репозитория
В - add	2 - Добавление изменений в область подготовки
С - commit	3 - Фиксация изменений в репозитории
Д - merge	4 - Объединение веток

Правильный ответ: А-1, В-2, С-3, D-4

Задача 3. Соотнесите инструменты для тестирования с их функциями (ПК-2.1, ПК-2.2)

Инструмент	Функция
А - JUnit	1 - Юнит-тестирование на Java
В - pytest	2 - Тестирование на Python
С - Selenium	3 - Автоматизированное тестирование веб-приложений
Д - Jest	4 - Тестирование JavaScript и React-приложений

Правильный ответ: А-1, В-2, С-3, D-4

Задача 4. Соотнесите функции инструментов CI/CD с их описаниями (ПК-3.1, УК-1.2)

Инструмент	Функция
А - Jenkins	1 - Автоматизация сборки, тестирования и развертывания
В - GitHub Actions	2 - Автоматизация процессов CI/CD в GitHub
С - GitLab CI	3 - Автоматизация CI/CD процессов в GitLab
Д - CircleCI	4 - Континуус интеграция и развертывание приложений

Правильный ответ: А-1, В-2, С-3, D-4

Задача 5. Соотнесите инструменты для контейнеризации с их функциями (ПК-3.1, ПК-3.2)

Инструмент	Функция
A - Docker	1 - Создание и управление контейнерами
B - Kubernetes	2 - Оркестрация и управление контейнерами
C - Docker Compose	3 - Определение и запуск многоконтейнерных приложений
D - OpenShift	4 - Платформа для управления контейнерными приложениями

Правильный ответ: A-1, B-2, C-3, D-4

Задача 6. Соотнесите инструменты для логирования с их функциями (ПК-3.2, УК-1.2)

Инструмент	Функция
A - ELK Stack	1 - Сбор, анализ и визуализация логов
B - Prometheus	2 - Мониторинг и сбор метрик
C - Grafana	3 - Визуализация данных и метрик
D - Fluentd	4 - Унифицированный сбор и передача логов

Правильный ответ: A-1, B-2, C-3, D-4

Критерии оценки выполнения задач на соответствие понятий

- **Правильность соответствий:**
 - **Отлично (5):** Все соответствия выполнены правильно.
 - **Хорошо (4):** 1 ошибка в соответствиях.
 - **Удовлетворительно (3):** 2 ошибки в соответствиях.
 - **Неудовлетворительно (2):** 3 и более ошибок в соответствиях.

3. Оценочные материалы для проведения промежуточной аттестации обучающихся (студентов)

3.1 Вопросы для проведения промежуточной аттестации в форме ЗАЧЕТА 5 семестр

1. Какие типы инструментов используются в разработке ПО и для чего они предназначены? (УК-1.1)
 - Правильный ответ: В разработке ПО используются IDE (для написания и отладки кода), текстовые редакторы (для работы с кодом), системы контроля версий (для управления изменениями кода), линтеры (для статического анализа кода).
2. Какие возможности предоставляют популярные IDE и как выбрать подходящую? (УК-2.1)
 - Правильный ответ: Популярные IDE, такие как IntelliJ IDEA, Visual Studio, Eclipse, предоставляют возможности для написания кода, отладки, тестирования и

- управления проектами. Выбор IDE зависит от языка программирования, специфики проекта и личных предпочтений разработчика.
3. Какие команды используются в Git для управления репозиторием и как они работают? (ПК-1.1)
 - Правильный ответ: Команды Git включают clone (клонирование репозитория), commit (фиксация изменений), push (отправка изменений на сервер), pull (получение изменений с сервера). Они позволяют управлять изменениями кода и синхронизировать работу в команде.
 4. Что такое линтеры и как они помогают разработчикам? (ПК-1.2)
 - Правильный ответ: Линтеры — это инструменты для статического анализа кода, которые помогают выявлять ошибки, нарушения стиля и потенциальные проблемы до выполнения кода. Примеры включают ESLint, Pylint.
 5. Какую роль выполняет Maven в управлении зависимостями и сборке проекта? (ПК-3.1)
 - Правильный ответ: Maven — это инструмент для управления зависимостями и автоматизации сборки проектов. Он упрощает процесс добавления, обновления и удаления библиотек, а также обеспечивает повторяемость сборки.
 6. Какие преимущества предоставляет использование Gradle в проекте? (УК-1.2)
 - Правильный ответ: Gradle предоставляет гибкость в управлении зависимостями, поддерживает многомодульные проекты, интеграцию с различными инструментами и возможность настройки сложных сценариев сборки.
 7. Какие функции выполняет ESLint и как он используется? (ПК-1.2)
 - Правильный ответ: ESLint анализирует код на предмет ошибок и нарушений стиля, обеспечивая консистентность и улучшая качество кода. Используется в проектах на JavaScript и других языках, поддерживающих ECMAScript.
 8. Для чего используется Prettier и как он помогает разработчикам? (ПК-1.2)
 - Правильный ответ: Prettier форматирует код в соответствии с заданными правилами стиля, обеспечивая единообразие и улучшая читаемость кода. Поддерживает множество языков программирования и интегрируется с популярными редакторами.
 9. Какую роль выполняет текстовый редактор в разработке ПО и какие возможности он предоставляет? (УК-1.1)
 - Правильный ответ: Текстовый редактор используется для редактирования кода и конфигурационных файлов. Возможности включают подсветку синтаксиса, автодополнение, поддержку плагинов и интеграцию с системами контроля версий.
 10. Какие инструменты используются для статического анализа кода и как они помогают разработчикам? (ПК-1.2)
 - Правильный ответ: Инструменты для статического анализа кода, такие как SonarQube и Coverity, помогают выявлять ошибки, уязвимости и нарушения стиля в коде до его выполнения, что улучшает качество и безопасность программного обеспечения.
 11. Какие функции выполняют инструменты для управления зависимостями и сборкой проектов? (ПК-3.1)
 - Правильный ответ: Инструменты, такие как Maven и Gradle, управляют библиотеками и зависимостями, автоматизируют процесс сборки, тестирования и развертывания проектов, обеспечивая повторяемость и консистентность сборок.
 12. Какие задачи решает система контроля версий и какие преимущества она предоставляет? (УК-1.2)
 - Правильный ответ: Система контроля версий решает задачи управления изменениями кода, совместной работы над проектом, отслеживания истории изменений и возврата к предыдущим версиям. Преимущества включают

улучшенную координацию работы команды, сохранение истории изменений и упрощение интеграции кода.

13. Какую роль играет интеграция с системами CI/CD в процессе разработки ПО? (ПК-3.1)
- Правильный ответ: Интеграция с системами CI/CD, такими как Jenkins и GitLab CI, автоматизирует процесс сборки, тестирования и развертывания, что позволяет быстрее находить и исправлять ошибки, ускоряет выпуск новых версий и улучшает качество ПО.
14. Какие инструменты используются для отладки кода и как они помогают разработчикам? (ПК-1.2)
- Правильный ответ: Инструменты для отладки кода, такие как gdb, Visual Studio Debugger, PyCharm Debugger, позволяют разработчикам пошагово выполнять код, проверять значения переменных и состояние системы, что помогает находить и исправлять ошибки.

Критерии оценивания для зачета (зачет/незачет)

1. Понимание основных понятий и терминов

- **Зачет:** Студент правильно использует основные понятия и термины, объясняет их в контексте.
- **Незачет:** Студент не знает или неправильно использует основные понятия и термины, не может объяснить их.

2. Точность и полнота ответов на теоретические вопросы

- **Зачет:** Ответы на большинство вопросов полные и точные, основные аспекты тем раскрыты.
- **Незачет:** Ответы неполные, содержат значительные ошибки или отсутствуют важные аспекты тем.

3. Способность к применению знаний на практике

- **Зачет:** Студент правильно решает практические задачи, выполняет SQL-запросы без ошибок.
- **Незачет:** Студент допускает ошибки при решении практических задач, не может правильно выполнить SQL-запросы.

4. Качество выполнения практических заданий

- **Зачет:** Практические задания выполнены корректно, все необходимые элементы присутствуют.
- **Незачет:** Практические задания выполнены некорректно, содержат ошибки или неполные.

Окончательное решение по зачету:

- **Зачет:** Студент демонстрирует понимание основных понятий, корректно отвечает на теоретические вопросы и выполняет практические задания.
- **Незачет:** Студент не демонстрирует понимания основных понятий, отвечает с ошибками на теоретические вопросы и/или не справляется с практическими заданиями.

3.2 Вопросы для проведения промежуточной аттестации в форме ЭКЗАМЕНА 6 семестр

1. Какие основные функции выполняет интегрированная среда разработки (IDE) и как выбрать подходящую IDE для проекта? (ПК-1.1)
- Правильный ответ: Интегрированная среда разработки (IDE) обеспечивает автодополнение кода, отладку, тестирование, рефакторинг, управление проектами и интеграцию с системами контроля версий. Выбор IDE зависит от языка программирования, требуемых функций и предпочтений разработчика. Примеры: Visual Studio для .NET, PyCharm для Python, IntelliJ IDEA для Java.

2. Что такое система контроля версий и какие задачи она решает? Примеры популярных систем контроля версий. (УК-1.1)
 - Правильный ответ: Система контроля версий управляет изменениями в коде, позволяет отслеживать историю изменений, возвращаться к предыдущим версиям и сотрудничать с командой. Примеры: Git, SVN, Mercurial.
3. Объясните основные команды Git и их использование. (ПК-1.1)
 - Правильный ответ: Основные команды Git включают init (инициализация репозитория), clone (клонирование репозитория), add (добавление изменений), commit (фиксация изменений), push (отправка изменений), pull (получение изменений). Эти команды позволяют эффективно управлять версиями кода и сотрудничать в команде.
4. Каковы преимущества использования линтеров и статических анализаторов кода? Примеры инструментов. (ПК-1.2)
 - Правильный ответ: Линтеры и статические анализаторы помогают выявлять синтаксические и логические ошибки, нарушения стиля кода до выполнения программы, что повышает качество и поддерживаемость кода. Примеры: ESLint для JavaScript, Pylint для Python, SonarQube для нескольких языков.
5. Какую роль выполняет Maven в проекте? (ПК-3.1)
 - Правильный ответ: Maven управляет зависимостями, автоматизирует процесс сборки, обеспечивает стандартизацию проекта и упрощает управление конфигурацией. Он также поддерживает управление проектами на основе POM (Project Object Model).
6. Какие задачи решает Gradle в разработке ПО? (УК-1.2)
 - Правильный ответ: Gradle автоматизирует сборку, управляет зависимостями, поддерживает многомодульные проекты и интеграцию с различными инструментами разработки. Его гибкость и масштабируемость делают его популярным выбором для проектов на Java и Android.
7. Что такое CI/CD и какие инструменты используются для этих процессов? (ПК-2.1)
 - Правильный ответ: CI/CD (непрерывная интеграция и развертывание) автоматизирует процессы сборки, тестирования и развертывания; популярные инструменты включают Jenkins, GitLab CI, CircleCI. Эти инструменты обеспечивают непрерывную доставку и повышение качества ПО.
8. Какие преимущества предоставляет Docker для разработки ПО и как он используется? (ПК-3.1)
 - Правильный ответ: Docker создает изолированные контейнеры для приложений, упрощает развертывание, масштабирование и обеспечивает консистентность среды разработки и производства. Это позволяет разработчикам избегать проблем, связанных с различиями в конфигурациях сред.
9. Как Prometheus используется для мониторинга приложений и что он предоставляет? (ПК-3.2)
 - Правильный ответ: Prometheus собирает и хранит метрики системы, мониторит производительность приложений, предоставляет алертинг и визуализацию данных. Он широко используется для мониторинга распределенных систем и микросервисов.
10. Какие возможности предоставляет GitLab CI и как он используется для автоматизации процессов CI/CD? (ПК-2.1)
 - Правильный ответ: GitLab CI предоставляет возможности для автоматизации CI/CD процессов, включая сборку, тестирование и развертывание ПО. Он интегрирован с системой управления версиями GitLab и позволяет управлять пайплайнами непосредственно из репозитория.
11. Как рефакторинг улучшает качество кода и когда его следует применять? (ПК-2.2)

- Правильный ответ: Рефакторинг улучшает структуру кода без изменения его внешнего поведения, что делает код более понятным, поддерживаемым и снижает технический долг. Его следует применять, когда код становится сложным для понимания или изменения, чтобы улучшить его читаемость и поддержку.
- 12. Какие задачи решает проектирование архитектуры ПО и почему это важно? (ПК-4.2)
 - Правильный ответ: Проектирование архитектуры ПО определяет структуру системы, взаимодействие между компонентами, выбор технологий, обеспечение масштабируемости и безопасности. Это важно для создания надежных, масштабируемых и поддерживаемых систем.
- 13. Какую роль играет проектирование интерфейсов в разработке ПО? (ПК-4.2)
 - Правильный ответ: Проектирование интерфейсов определяет, как компоненты системы будут взаимодействовать друг с другом, что обеспечивает модульность, гибкость и совместимость системы. Это позволяет разработчикам работать независимо над различными компонентами.
- 14. Какие принципы следует учитывать при выборе оптимальных способов решения задач? (УК-2.2)
 - Правильный ответ: Следует учитывать соответствие правовым нормам, доступность ресурсов, время выполнения задачи, минимизацию рисков и обеспечение качества результата. Это помогает принимать обоснованные решения и достигать поставленных целей.
- 15. Как правильно выбирать способы решения задач с учетом правовых норм? (УК-2.2)
 - Правильный ответ: Необходимо анализировать правовые требования, консультироваться с юридическими специалистами, обеспечивать соответствие законодательству и внутренним политикам компании. Это помогает избежать юридических проблем и соблюдать нормативные требования.
- 16. Какие методы используются для разработки требований к программному обеспечению? (ПК-4.1)
 - Правильный ответ: Интервью, анкеты, анализ документов, наблюдение, прототипирование и рабочие совещания используются для сбора и уточнения требований к ПО. Эти методы помогают выявить потребности пользователей и формализовать требования.
- 17. Какие аспекты необходимо учитывать при проектировании программного обеспечения? (ПК-4.1)
 - Правильный ответ: Требования пользователя, архитектура системы, масштабируемость, безопасность, производительность и поддерживаемость. Учет этих аспектов помогает создавать качественное и надежное ПО.
- 18. Какие основные функции выполняет система контроля версий? (ПК-1.1)
 - Правильный ответ: Управляет изменениями в коде, позволяет отслеживать историю изменений, возвращаться к предыдущим версиям и сотрудничать с командой. Примеры: Git, SVN, Mercurial.
- 19. Каковы основные преимущества использования линтеров в разработке ПО? (ПК-1.2)
 - Правильный ответ: Линтеры помогают выявлять синтаксические и логические ошибки, нарушения стиля кода до выполнения программы, что повышает качество и поддерживаемость кода. Примеры: ESLint для JavaScript, Pylint для Python.
- 20. Какие задачи решает Jenkins в процессе разработки ПО? (ПК-2.1)
 - Правильный ответ: Jenkins автоматизирует процессы сборки, тестирования и развертывания приложений, обеспечивает непрерывную интеграцию и доставку. Это ускоряет выпуск новых версий и повышает качество ПО.
- 21. Какие возможности предоставляет Visual Studio для разработчиков? (ПК-1.1)

- Правильный ответ: Visual Studio предоставляет средства для написания кода, отладки, тестирования, управления проектами и интеграции с системами контроля версий. Это одна из самых мощных и популярных IDE.
- 22. Какую роль играет управление конфигурацией в процессе разработки ПО? (УК-2.1)
 - Правильный ответ: Включает идентификацию конфигурационных единиц, контроль изменений, отчетность по конфигурациям, планирование и выполнение релизов. Это помогает поддерживать целостность и консистентность системы.
- 23. Какие задачи решает интеграция с системами CI/CD в процессе разработки ПО? (ПК-2.1)
 - Правильный ответ: Автоматизация сборки, тестирования и развертывания, ускорение выпуска новых версий, повышение качества ПО. Интеграция с системами CI/CD обеспечивает непрерывную доставку и улучшает производительность команды.
- 24. Какую роль играет оркестрация контейнеров в разработке ПО? (ПК-3.2)
 - Правильный ответ: Управление и координация работы множества контейнеров, включая их развертывание, масштабирование и мониторинг. Примеры инструментов: Kubernetes, Docker Swarm. Это упрощает управление сложными распределенными системами.

Критерии оценки ответов на экзамене

- **"Отлично" (5 баллов)**
 - **Критерии:**
 - Полное и точное объяснение вопроса.
 - Ответ включает все ключевые аспекты и детали.
 - Примеры, если требуются, приведены и правильно объяснены.
 - Ответ демонстрирует глубокое понимание темы.
- **"Хорошо" (4 балла)**
 - **Критерии:**
 - Корректное объяснение вопроса.
 - Ответ охватывает основные аспекты, но может отсутствовать незначительная деталь или пример.
 - Демонстрируется хорошее, но не полное понимание темы.
- **"Удовлетворительно" (3 балла)**
 - **Критерии:**
 - Общее представление о вопросе.
 - Ответ включает основные аспекты, но содержит неточности или пропуски.
 - Примеры, если требуются, могут отсутствовать или быть неверно объяснены.
 - Демонстрируется базовое понимание темы.
- **"Неудовлетворительно" (2 балла)**
 - **Критерии:**
 - Некорректное или неполное объяснение вопроса.
 - Отсутствие ключевых аспектов и деталей.
 - Примеры, если требуются, отсутствуют или приведены неверные.
 - Ответ демонстрирует недостаточное понимание темы.