МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

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

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ
Б1.В.02 «Разработка программных модулей»

Направление подготовки: 09.03.02 Информационные системы и технологии

Направленность (профиль) программы бакалавриата
«Информационные технологии»

ОДОБРЕНО	**УТВЕРЖДАЮ**
Ученым советом ЧУВО «ВШП»	Ректор ЧУВО «ВШП»
Протокол заседания	«1[illegible]» мая 2025 г.
№01-02/24 от 15 мая 2025 г.	Ректор ____________ [illegible]ллабян М.Г.

Тверь, 2025

Рабочая программа учебной дисциплины **Б1.В.02 Разработка программных модулей**, 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 «Разработка программных модулей» является я формирование целостной системы знаний, умений и опыта практической деятельности в области разработок программных модулей.

Для этого в рамках дисциплины решаются следующие задачи:

- овладение методами сбора материалов обследования предприятия для разработки программных модулей;
- изучение стандартов оформления программной документации;
- овладение технологиями формализации материалов обследования предприятия для разработки и функционирования программных модулей;
- изучение способов и участие в разработки и оформлении проектной документации на программные модули и ее части.

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 и методы поиска информации. - Принципы и методы критического анализа информации. Уметь: - Использовать различные источники для поиска необходимой информации. - Анализировать и оценивать достоверность и релевантность информации. Владеть:

			- Навыками работы с электронными библиотеками и базами данных. - Навыками анализа информации и выделения ключевых аспектов.
		УК-1.2 Способен применять системный подход для решения поставленных задач	Знать: - Принципы системного подхода в решении задач. - Методы и инструменты системного анализа. Уметь: - Формулировать задачи и определять их ключевые компоненты. - Применять методы системного анализа для решения комплексных задач. Владеть: - Навыками построения моделей и схем для анализа задач. - Навыками применения системного подхода для оптимизации решений.
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 и техники формулирования задач. - Принципы декомпозиции задач на подзадачи. Уметь: - Определять цели и задачи проекта. - Делить задачи на этапы и определять приоритеты. Владеть: - Навыками структурирования и организации задач. - Навыками работы с целями и задачами проекта.
		УК-2.2 Способен выбирать оптимальные способы решения задач, исходя из правовых норм, ресурсов и ограничений	Знать: - Принципы и методы выбора оптимальных решений. - Правовые нормы и ограничения, влияющие на процесс разработки. Уметь: - Оценивать доступные ресурсы и ограничения проекта. - Выбирать и обосновывать оптимальные методы и инструменты для решения задач. Владеть: - Навыками анализа и оценки рисков при выборе решений. - Навыками работы с правовыми нормами и стандартами в области разработки программного обеспечения.
ПК-1	Разработка и отладка программного кода	ПК-1.1 Способен разрабатывать программный код	Знать: - Основные принципы и методы разработки программного кода. - Синтаксис и семантику используемых языков программирования (Python, C/C++, JS). Уметь: - Разрабатывать корректный и эффективный программный код. - Использовать стандартные библиотеки и фреймворки для разработки. Владеть: - Навыками написания и документирования кода.

			- Навыками применения передовых практик программирования (SOLID, DRY).
		ПК-1.2 Способен отлаживать программный код	Знать: - Основные методы и инструменты отладки программного кода. - Принципы и техники выявления и исправления ошибок. Уметь: - Использовать отладчики и инструменты профилирования для поиска ошибок. - Анализировать и исправлять ошибки в программном коде. Владеть: - Навыками эффективной отладки и тестирования кода. - Навыками применения различных техник отладки (пошаговая отладка, логирование).
ПК-2	Проверка работоспособности и рефакторинг кода программного обеспечения	ПК-2.1 Способен проверять работоспособность программного кода	Знать: - Принципы и методы тестирования программного кода. - Основы юнит-тестирования и функционального тестирования. Уметь: - Разрабатывать и выполнять тестовые сценарии для проверки работоспособности кода. - Использовать тестовые фреймворки и инструменты для автоматизированного тестирования. Владеть: - Навыками написания и выполнения тестов. - Навыками анализа результатов тестирования и устранения обнаруженных проблем.
		ПК-2.2 Способен рефакторить программный код	Знать: - Принципы и методы рефакторинга программного кода. - Паттерны и антипаттерны проектирования кода. Уметь: - Проводить анализ и улучшение структуры кода. - Вносить изменения в код, сохраняя его функциональность. Владеть: - Навыками применения рефакторинга для улучшения читаемости и поддерживаемости кода. - Навыками использования инструментов для автоматизированного рефакторинга.
ПК-3	Интеграция программных модулей и компонентов и проверка работоспособности выпусков программного продукта	ПК-3.1 Способен интегрировать программные модули и компоненты	Знать: - Принципы и методы интеграции программных модулей. - Основы межмодульного взаимодействия и совместимости. Уметь: - Планировать и проводить интеграцию модулей в общий проект. - Решать проблемы, возникающие при интеграции различных компонентов. Владеть: - Навыками использования инструментов и платформ для интеграции. - Навыками написания интеграционных тестов.

		ПК-3.2 Способен проверять работоспособность выпусков программного продукта	Знать: - Методы и инструменты для проверки выпусков программного продукта. - Основы тестирования и валидации программного обеспечения. Уметь: - Проводить проверку и валидацию программного продукта. - Анализировать результаты тестирования и устранять выявленные дефекты. Владеть: - Навыками работы с инструментами для проверки и тестирования выпусков. - Навыками документирования результатов тестирования и предложений по улучшению.
ПК-4	Разработка требований и проектирование программного обеспечения	ПК-4.1 Способен разрабатывать требования к программному обеспечению	Знать: - Методы и техники сбора и анализа требований. - Принципы написания требований и спецификаций. Уметь: - Собирать и документировать требования к программному обеспечению. - Анализировать и уточнять требования с заинтересованными сторонами. Владеть: - Навыками написания четких и измеримых требований. - Навыками работы с инструментами для управления требованиями.
		ПК-4.2 Способен проектировать программное обеспечение	Знать: - Принципы и методы проектирования программного обеспечения. - Основы архитектурных шаблонов и стилей. Уметь: - Разрабатывать архитектуру и дизайн программного обеспечения. - Создавать диаграммы и модели для визуализации архитектуры. Владеть: - Навыками применения архитектурных паттернов. - Навыками использования инструментов для проектирования и моделирования программного обеспечения.

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.) - 14 академических часов,
Практические занятия (Пр.) - 32 академических часов,
Консультации (Конс.) - 2 академических часа.
Самостоятельная работа обучающегося (студента):
Самостоятельная работа (СР) - 60 академический час,
Текущий контроль успеваемости
и промежуточно-заочная аттестация обучающегося (студента):
Часы на контроль - 36 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			Самостоятельная работа	
			Лек.	Пр.	Консульт	СР	
1	**Тема 1.** Введение в разработку программных модулей	5 семестр/ 3 курс	1	2	-	5	УК-1.1, УК-1.2
2	**Тема 2.** Основы модульного программирования на Python	5 семестр/ 3 курс	2	4	-	5	ПК-1.1, ПК-1.2
3	**Тема 3.** Основы модульного программирования на C/C++	5 семестр/ 3 курс	2	4	-	5	ПК-1.1, ПК-1.2
4	**Тема 4.** Основы модульного программирования на JavaScript	5 семестр/ 3 курс	2	2	-	5	ПК-1.1, ПК-1.2
5	**Тема 5.** Архитектура программных модулей	5 семестр/ 3 курс	1	2	-	5	ПК-4.2, УК-2.1
6	**Тема 6.** Инкапсуляция и интерфейсы в модульном программировании	5 семестр/ 3 курс	1	2	-	5	ПК-4.2, УК-2.2
7	**Тема 7.** Тестирование программных модулей	5 семестр/ 3 курс	1	2	-	5	ПК-2.1, ПК-3.2
8	**Тема 8** Отладка и профилирование программных модулей	5 семестр/ 3 курс	1	2	-	5	ПК-2.2, ПК-3.1
9	**Тема 9** Управление зависимостями в модульных системах	5 семестр/ 3 курс	1	2	-	5	ПК-3.1, ПК-4.1
10	**Тема 10** Практическая работа по разработке и интеграции модулей	5 семестр/ 3 курс	1	2	-	5	ПК-1.1, ПК-3.1

11	**Тема 11** Современные подходы и инструменты для модульного программирования	5 семестр/ 3 курс	1	2	-	5	ПК-4.2, УК-1.2
12	**Тема 12** Итоговое занятие и подготовка к экзамену	5 семестр/ 3 курс		6	2	5	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 семестр/ 3 курс	**48 ак. час**			**60 ак. час**	-
Часы на контроль		5 семестр/ 3 курс	**36 ак. час (форма промежуточной аттестации – экзамен – 5 семестр)**				
ВСЕГО ак. часов:		5 семестр/ 3 курс	**144 академических часа**				

5.2. Тематическое содержание дисциплины

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

Тема 1 - Введение в разработку программных модулей
Понятие и цели разработки программных модулей. Преимущества модульного программирования. Обзор основных языков программирования для разработки модулей (Python, C/C++, JS).

Тема 2 - Основы модульного программирования на Python
Создание и использование модулей в Python. Работа с библиотеками и пакетами. Основные функции и методы для взаимодействия между модулями.

Тема 3 - Основы модульного программирования на C/C++
Создание и компиляция модулей на C/C++. Работа с заголовочными файлами и библиотеками. Основные функции и методы для взаимодействия между модулями.

Тема 4 - Основы модульного программирования на JavaScript
Создание и использование модулей в JavaScript (ES6). Работа с библиотеками и пакетами (Node.js). Основные функции и методы для взаимодействия между модулями.

Тема 5 - Архитектура программных модулей
Принципы проектирования модульной архитектуры. Разделение функциональности на модули. Организация и структура модулей. Взаимодействие между модулями.

Тема 6 - Инкапсуляция и интерфейсы в модульном программировании
Понятие инкапсуляции. Определение и использование интерфейсов для взаимодействия между модулями. Примеры реализации инкапсуляции и интерфейсов на Python, C/C++ и JavaScript.

Тема 7 - Тестирование программных модулей
Методы и инструменты тестирования модулей. Написание юнит-тестов. Использование тестовых фреймворков (unittest для Python, Google Test для C++, Mocha/Chai для JavaScript).

Тема 8 - Отладка и профилирование программных модулей
Методы и инструменты отладки модулей. Профилирование кода для выявления узких мест и оптимизация. Использование отладочных и профилировочных инструментов (gdb для C/C++, pdb для Python, Chrome DevTools для JavaScript).

Тема 9 - Управление зависимостями в модульных системах
Принципы управления зависимостями между модулями. Использование менеджеров пакетов (pip для Python, npm для Node.js, CMake для C++). Введение в управление версиями и совместимостью модулей.

Тема 10 - Практическая работа по разработке и интеграции модулей
Разработка и интеграция модулей на Python, C/C++ и JavaScript. Примеры создания и интеграции модулей в один проект. Практические упражнения по взаимодействию между модулями.

Тема 11 - Современные подходы и инструменты для модульного программирования
Обзор современных подходов и инструментов для модульного программирования. Использование контейнеров (Docker) для изоляции модулей. Введение в микросервисную архитектуру.

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 разработки модулей.

5.2.1 Содержание практических занятий

Таблица 3

№ п/п	Наименование темы (раздела) дисциплины	Содержание практического занятия
1	**Практическое занятие 1: Введение в разработку программных модулей**	Задание: Ознакомление с основными принципами и целями модульного программирования. Обзор основных языков программирования для разработки модулей (Python, C/C++, JS). Цель: Понять концепцию модульного программирования и его преимущества.
2	**Практическое занятие 2: Основы модульного программирования на Python**	Задание: Создание простого модуля в Python. Импорт и использование модулей. Работа с библиотеками и пакетами. Цель: Научиться создавать и использовать модули в Python, а также работать с библиотеками.
3	**Практическое занятие 3: Основы модульного программирования на C/C++**	Задание: Создание и компиляция простого модуля на C/C++. Работа с заголовочными файлами и библиотеками. Цель: Научиться создавать и компилировать модули на C/C++, а также работать с заголовочными файлами.
4	**Практическое занятие 4: Основы модульного программирования на JavaScript**	Задание: Создание простого модуля в JavaScript (ES6). Импорт и использование модулей. Работа с пакетами в Node.js. Цель: Научиться создавать и использовать модули в JavaScript, а также работать с пакетами в Node.js.
5	**Практическое занятие 5: Архитектура программных модулей**	Задание: Проектирование модульной архитектуры для небольшого проекта. Разделение функциональности на модули. Организация и структура модулей. Цель: Понять принципы проектирования модульной архитектуры и научиться организовывать модули в проекте.
6	**Практическое занятие 6: Инкапсуляция и интерфейсы в модульном программировании**	Задание: Реализация инкапсуляции и интерфейсов в модулях на Python, C/C++ и JavaScript. Примеры использования интерфейсов для взаимодействия между модулями. Цель: Научиться применять инкапсуляцию и использовать интерфейсы для взаимодействия между модулями.
7	**Практическое занятие 7: Тестирование программных модулей**	Задание: Написание юнит-тестов для модулей на Python, C++ и JavaScript. Использование тестовых фреймворков (unittest, Google Test, Mocha/Chai). Цель: Научиться тестировать модули, используя юнит-тесты и тестовые фреймворки.
8	**Практическое занятие 8: Отладка и профилирование**	Задание: Отладка и профилирование модулей на Python, C/C++ и JavaScript. Использование отладочных и профилировочных инструментов (gdb, pdb, Chrome DevTools).

	программных модулей	Цель: Научиться отлаживать и профилировать модули для выявления и исправления ошибок, а также оптимизации производительности.
9	**Практическое занятие 9: Управление зависимостями в модульных системах**	Задание: Управление зависимостями между модулями с использованием менеджеров пакетов (pip, npm, CMake). Введение в управление версиями и совместимостью модулей. Цель: Научиться управлять зависимостями между модулями и использовать менеджеры пакетов.
10	**Практическое занятие 10: Разработка и интеграция модулей**	Задание: Разработка и интеграция модулей на Python, C/C++ и JavaScript. Примеры создания и интеграции модулей в один проект. Цель: Научиться разрабатывать и интегрировать модули в рамках одного проекта.
11	**Практическое занятие 11: Современные подходы и инструменты для модульного программирования**	Задание: Изучение современных подходов и инструментов для модульного программирования. Использование контейнеров (Docker) для изоляции модулей. Введение в микросервисную архитектуру. Цель: Ознакомиться с современными инструментами и подходами для модульного программирования.
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 итоговому экзамену, закрепив основные понятия и навыки, полученные в ходе курса.

5.2.2 Содержание самостоятельной работы

Таблица 4

№ п/п	**Наименование темы (раздела) дисциплины**	**Содержание самостоятельной работы**	**Форма контроля**
1	**1. Изучение литературы по модульному программированию**	Задание: Прочитать рекомендованные главы из учебников по модульному программированию и изучить статьи в интернете.	Ответы на контрольные вопросы.
2	**2. Выполнение упражнений по созданию модуля на Python**	Задание: Самостоятельно создать и использовать простой модуль в Python. Экспериментировать с импортом и экспортом функций и переменных.	Выполнение практического задания.
3	**Выполнение упражнений по созданию модуля на C/C++**	Задание: Самостоятельно создать и скомпилировать простой модуль на C/C++. Экспериментировать с заголовочными файлами и библиотеками.	Выполнение практического задания.
4	**4. Выполнение упражнений по созданию модуля на JavaScript**	Задание: Самостоятельно создать и использовать модуль в JavaScript (ES6). Экспериментировать с импортом и экспортом функций и переменных.	Выполнение практического задания.
5	**5. Контрольные вопросы по архитектуре программных модулей**	Задание: Ответить на контрольные вопросы по теме проектирования и организации модульной архитектуры.	Ответы на контрольные вопросы.
6	**6. Практическое упражнение: Инкапсуляция и интерфейсы**	Задание: Реализовать инкапсуляцию и интерфейсы в модулях на Python, C/C++ и JavaScript. Продемонстрировать взаимодействие между модулями через интерфейсы.	Выполнение практического задания.
7	**7. Контрольные вопросы по тестированию программных модулей**	Задание: Ответить на контрольные вопросы по методам и инструментам тестирования модулей.	Ответы на контрольные вопросы.
8	**8. Практическое упражнение: Написание юнит-тестов**	Задание: Написать юнит-тесты для модулей на Python, C++ и JavaScript с использованием соответствующих тестовых фреймворков.	Выполнение практического задания.
9	**9. Практическое упражнение: Отладка и профилирование модулей**	Задание: Провести отладку и профилирование модулей на Python, C/C++ и JavaScript. Найти и исправить ошибки, оптимизировать производительность.	Выполнение практического задания.
10	**10. Контрольные вопросы по управлению зависимостями**	Задание: Ответить на контрольные вопросы по управлению зависимостями между модулями и использованию менеджеров пакетов.	Ответы на контрольные вопросы.
11	**11. Практическое упражнение: Управление зависимостями**	Задание: Настроить и управлять зависимостями между модулями с использованием pip для Python, npm для Node.js и CMake для C++.	Выполнение практического задания.
12	**12. Итоговая**	Задание: Разработать и интегрировать модули на	Выполнение

	самостоятельная работа по интеграции модулей	Python, C/C++ и JavaScript в один проект. Продемонстрировать взаимодействие между модулями.	практического задания.
13	**13. Повторение и обобщение пройденного материала**	Задание: Повторить пройденный материал, ответить на контрольные вопросы из учебников и методических пособий.	Тест.
14	**14. Подготовка и сдача экзамена**	Задание: Подготовиться к экзамену, используя конспекты и выполненные задания.	Экзамен.

6.Оценочные материалы по дисциплине

Оценочные материалы по дисциплине находятся в документе «Оценочные материалы по дисциплине «Разработка программных модулей».

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 приобретается 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 Институтом 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 Институтом 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. Лутц М., Изучаем Python [Электронный ресурс] : учебное пособие / М. Лутц. - М. : Вильямс, 2022. - 1500 с. - ISBN 978-5-93286-210-0. - Режим доступа: https://book.ru/book/923956
2. Зак Ф., JavaScript для профессионалов [Электронный ресурс] : учебное пособие / Ф. Зак. - М. : ДМК Пресс, 2020. - 496 с. - ISBN 978-5-97060-357-1. - Режим доступа: https://book.ru/book/925276
3. Страуструп Б., Программирование на C++ [Электронный ресурс] : учебное пособие / Б. Страуструп. - М. : Вильямс, 2022. - 1040 с. - ISBN 978-5-8459-1705-8. - Режим доступа: https://book.ru/book/923644

Дополнительная литература:

1. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж., Приемы объектно-ориентированного проектирования. Паттерны проектирования [Электронный ресурс] : учебное пособие / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. - М. : Вильямс, 2022. - 394 с. - ISBN 978-5-8459-1963-9. - Режим доступа: https://book.ru/book/923745

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: «НТЦ 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/

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)**
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 обучающийся, с перечнем основного оборудования: - 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.	170001, Россия, город Тверь, улица Спартака, дом 26а
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 обучающийся, с перечнем основного оборудования: - 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;	170001, Россия, город Тверь, улица Спартака, дом 26а

- Проектор.	
Помещение для самостоятельной работы обучающихся с перечнем основного оборудования: - 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.	170001, Россия, город Тверь, улица Спартака, дом 26а
Помещение для практических занятий на персональных компьютерах: - 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.	170001, Россия, город Тверь, улица Спартака, дом 26а

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

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

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ
по дисциплине
Б1.В.02 «Разработка программных модулей»

**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: - Основные источники и методы поиска информации. - Принципы и методы критического анализа информации. Уметь: - Использовать различные источники для поиска необходимой информации. - Анализировать и оценивать достоверность и релевантность информации. Владеть: - Навыками работы с электронными библиотеками и базами данных. - Навыками анализа информации и выделения ключевых аспектов.
		УК-1.2 Способен применять системный подход для решения поставленных задач	Знать: - Принципы системного подхода в решении задач. - Методы и инструменты системного анализа. Уметь: - Формулировать задачи и определять их ключевые компоненты. - Применять методы системного анализа для решения комплексных задач. Владеть: - Навыками построения моделей и схем для анализа задач. - Навыками применения системного подхода для оптимизации решений.
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 и техники формулирования задач. - Принципы декомпозиции задач на подзадачи. Уметь: - Определять цели и задачи проекта. - Делить задачи на этапы и определять приоритеты. Владеть: - Навыками структурирования и организации задач. - Навыками работы с целями и задачами проекта.
		УК-2.2 Способен выбирать оптимальные способы решения задач, исходя из	Знать: - Принципы и методы выбора оптимальных решений. - Правовые нормы и ограничения, влияющие на процесс разработки.

		правовых норм, ресурсов и ограничений	Уметь: - Оценивать доступные ресурсы и ограничения проекта. - Выбирать и обосновывать оптимальные методы и инструменты для решения задач. Владеть: - Навыками анализа и оценки рисков при выборе решений. - Навыками работы с правовыми нормами и стандартами в области разработки программного обеспечения.
ПК-1	Разработка и отладка программного кода	ПК-1.1 Способен разрабатывать программный код	Знать: - Основные принципы и методы разработки программного кода. - Синтаксис и семантику используемых языков программирования (Python, C/C++, JS). Уметь: - Разрабатывать корректный и эффективный программный код. - Использовать стандартные библиотеки и фреймворки для разработки. Владеть: - Навыками написания и документирования кода. - Навыками применения передовых практик программирования (SOLID, DRY).
		ПК-1.2 Способен отлаживать программный код	Знать: - Основные методы и инструменты отладки программного кода. - Принципы и техники выявления и исправления ошибок. Уметь: - Использовать отладчики и инструменты профилирования для поиска ошибок. - Анализировать и исправлять ошибки в программном коде. Владеть: - Навыками эффективной отладки и тестирования кода. - Навыками применения различных техник отладки (пошаговая отладка, логирование).
ПК-2	Проверка работоспособности и рефакторинг кода программного обеспечения	ПК-2.1 Способен проверять работоспособность программного кода	Знать: - Принципы и методы тестирования программного кода. - Основы юнит-тестирования и функционального тестирования. Уметь: - Разрабатывать и выполнять тестовые сценарии для проверки работоспособности кода. - Использовать тестовые фреймворки и инструменты для автоматизированного тестирования. Владеть: - Навыками написания и выполнения тестов.

			- Навыками анализа результатов тестирования и устранения обнаруженных проблем.
		ПК-2.2 Способен рефакторить программный код	Знать: - Принципы и методы рефакторинга программного кода. - Паттерны и антипаттерны проектирования кода. Уметь: - Проводить анализ и улучшение структуры кода. - Вносить изменения в код, сохраняя его функциональность. Владеть: - Навыками применения рефакторинга для улучшения читаемости и поддерживаемости кода. - Навыками использования инструментов для автоматизированного рефакторинга.
ПК-3	Интеграция программных модулей и компонентов и проверка работоспособности выпусков программного продукта	ПК-3.1 Способен интегрировать программные модули и компоненты	Знать: - Принципы и методы интеграции программных модулей. - Основы межмодульного взаимодействия и совместимости. Уметь: - Планировать и проводить интеграцию модулей в общий проект. - Решать проблемы, возникающие при интеграции различных компонентов. Владеть: - Навыками использования инструментов и платформ для интеграции. - Навыками написания интеграционных тестов.
		ПК-3.2 Способен проверять работоспособность выпусков программного продукта	Знать: - Методы и инструменты для проверки выпусков программного продукта. - Основы тестирования и валидации программного обеспечения. Уметь: - Проводить проверку и валидацию программного продукта. - Анализировать результаты тестирования и устранять выявленные дефекты. Владеть: - Навыками работы с инструментами для проверки и тестирования выпусков. - Навыками документирования результатов тестирования и предложений по улучшению.
ПК-4	Разработка требований и проектирование программного обеспечения	ПК-4.1 Способен разрабатывать требования к программному обеспечению	Знать: - Методы и техники сбора и анализа требований. - Принципы написания требований и спецификаций. Уметь: - Собирать и документировать требования к программному обеспечению. - Анализировать и уточнять требования с заинтересованными сторонами. Владеть:

			- Навыками написания четких и измеримых требований. - Навыками работы с инструментами для управления требованиями.
		ПК-4.2 Способен проектировать программное обеспечение	Знать: - Принципы и методы проектирования программного обеспечения. - Основы архитектурных шаблонов и стилей. Уметь: - Разрабатывать архитектуру и дизайн программного обеспечения. - Создавать диаграммы и модели для визуализации архитектуры. Владеть: - Навыками применения архитектурных паттернов. - Навыками использования инструментов для проектирования и моделирования программного обеспечения.

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); недостаточное 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»	
УК-1, УК-2, ПК-1, ПК-2, ПК-3, ПК-4 **Знать:**	Обучаемый 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	Обучаемый 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	Обучаемый 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	Обучаемый продемонстрировал неправильные ответы на основные вопросы; грубые ошибки в ответах; непонимание сущности излагаемых вопросов; неуверенные и неточные ответы на дополнительные вопросы; не владеет основной литературой, рекомендованной учебной программой дисциплины.	Тестовые задания

	программой дисциплины.	замечаний по отдельным вопросам; достаточное владение литературой.	литературой, рекомендованной учебной программой дисциплины.		
УК-1, УК-2, ПК-1, ПК-2, ПК-3, ПК-4 **Уметь:**	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, устранение, 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 учебной литературой, рекомендованной учебной программой дисциплины	Вопросы Практические задания
УК-1, УК-2, ПК-1, ПК-2, ПК-3, ПК-4 **Владеть:**	Обучаемый продемонстрировал: понимание учебного материала; правильные и конкретные ответы, включая дополнительны е уточняющие вопросы; свободное владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины	Обучаемый продемонстрировал: понимание учебного материала; продемонстрировал логически последовательные, достаточно полные, верные ответы; самостоятельно устранил замечания по отдельным элементам; владение основной и дополнительной литературой, рекомендованной учебной программой дисциплины	Обучаемый продемонстрировал: понимание основного учебного материала; без грубых ошибок дал ответы на поставленные вопросы, в том числе при наводящих вопросах преподавателя; недостаточно полное владение литературой, рекомендованной учебной программой дисциплины	Обучаемый продемонстрировал: непонимание основного учебного материала; дал неправильные ответы на поставленные вопросы; не владеет основной учебной литературой, рекомендованной учебной программой дисциплины	Вопросы Практические задания

1. Оценочные материалы для самостоятельной работы обучающихся (студентов)

1.1. Доклад
5 семестр

Доклад позволит студентам углубиться в концепции и методы разработки программных модулей, а также понять их важность и применение в современных информационных системах. 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 разработку программных модулей:** Понятие и цели разработки программных модулей, преимущества модульного программирования, обзор основных языков программирования для разработки модулей (Python, C/C++, JavaScript).
2. **Основы модульного программирования на Python:** Создание и использование модулей в Python, работа с библиотеками и пакетами, основные функции и методы для взаимодействия между модулями.
3. **Основы модульного программирования на C/C++:** Создание и компиляция модулей на C/C++, работа с заголовочными файлами и библиотеками, основные функции и методы для взаимодействия между модулями.
4. **Основы модульного программирования на JavaScript:** Создание и использование модулей в JavaScript (ES6), работа с библиотеками и пакетами (Node.js), основные функции и методы для взаимодействия между модулями.
5. **Архитектура программных модулей:** Принципы проектирования модульной архитектуры, разделение функциональности на модули, организация и структура модулей, взаимодействие между модулями.
6. **Инкапсуляция и интерфейсы в модульном программировании:** Понятие инкапсуляции, определение и использование интерфейсов для взаимодействия между модулями, примеры реализации инкапсуляции и интерфейсов на Python, C/C++ и JavaScript.
7. **Тестирование программных модулей:** Методы и инструменты тестирования модулей, написание юнит-тестов, использование тестовых фреймворков (unittest для Python, Google Test для C++, Mocha/Chai для JavaScript).
8. **Отладка и профилирование программных модулей:** Методы и инструменты отладки модулей, профилирование кода для выявления узких мест и оптимизация, использование отладочных и профилировочных инструментов (gdb для C/C++, pdb для Python, Chrome DevTools для JavaScript).
9. **Управление зависимостями в модульных системах:** Принципы управления зависимостями между модулями, использование менеджеров пакетов (pip для Python, npm для Node.js, CMake для C++), введение в управление версиями и совместимостью модулей.
10. **Практическая работа по разработке и интеграции модулей:** Разработка и интеграция модулей на Python, C/C++ и JavaScript, примеры создания и интеграции модулей в один проект, практические упражнения по взаимодействию между модулями.
11. **Современные подходы и инструменты для модульного программирования:** Обзор современных подходов и инструментов для модульного программирования, использование контейнеров (Docker) для изоляции модулей, введение в микросервисную архитектуру.

Цель написания докладов:

Углубить понимание и критическое осмысление роли разработки программных модулей в современных информационных системах, 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 разработки программных модулей 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 этапов и методов разработки модулей.
    - Анализ современных тенденций.
    - Примеры применения разработки модулей в реальных проектах.
3. **Аналитическая часть (20%)**
    - Глубина анализа роли разработки модулей 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. Какой язык программирования является интерпретируемым и имеет динамическую типизацию? (УК-1.1)
    - Python +
    - C++
    - Java
    - C
2. Что такое модульное программирование? (УК-1.2)
    - Разработка программ из отдельных независимых модулей, которые могут быть интегрированы и использованы повторно. +
    - Создание программы из единого монолитного кода.
    - Программирование только на одном языке.
    - Написание кода без использования функций.
3. Как импортировать модуль в Python? (ПК-1.1)
    - include module
    - import module +
    - using module
    - require module
4. Какой инструмент используется для управления пакетами в Python? (ПК-3.1)
    - npm
    - pip +
    - gem
    - apt
5. Какой метод используется для определения функций в модуле Python? (ПК-1.1)
    - function myFunction()
    - def myFunction() +
    - create myFunction()
    - lambda myFunction()
6. Что такое заголовочный файл в C++? (ПК-1.1)
    - Файл, содержащий исполняемый код программы.
    - Файл, содержащий объявления функций и переменных, используемый для разделения интерфейса и реализации. +
    - Файл, содержащий только комментарии.
    - Файл, содержащий только тесты.
7. Какой метод используется для экспорта функций в JavaScript (ES6)? (ПК-1.1)
    - import
    - require
    - export +
    - module.exports
8. Какой менеджер пакетов используется для установки библиотек в Node.js? (ПК-3.1)
    - pip
    - npm +
    - gem
    - apt
9. Что такое архитектура программного модуля? (ПК-4.2)
    - Набор инструкций для компиляции модуля.
    - Структура и организация кода внутри модуля и взаимодействие с другими модулями. +
    - Метод тестирования модулей.

- Способ отладки и профилирования кода.

10. Какие принципы применяются при проектировании модульной архитектуры? (ПК-4.2)
    - Принцип единой ответственности +
    - Принцип максимизации кода
    - Принцип минимизации тестов
    - Принцип отказа от интерфейсов
11. Что такое инкапсуляция в объектно-ориентированном программировании? (ПК-1.1)
    - Метод использования объектов для инкапсуляции кода и данных.
    - Способ скрытия деталей реализации класса и предоставления публичного интерфейса для взаимодействия с ним. +
    - Способ наследования свойств и методов от родительского класса.
    - Способ полиморфизма для работы с объектами разных классов.
12. Как определяется интерфейс в программировании? (ПК-4.2)
    - Как реализация метода в классе
    - Как совокупность методов, которые должны быть реализованы в классе +
    - Как наследование свойств и методов
    - Как способ полиморфизма
13. Что такое юнит-тестирование? (ПК-2.1)
    - Тестирование всей системы в целом.
    - Тестирование отдельных модулей или компонентов программы. +
    - Тестирование интеграции модулей.
    - Тестирование производительности программы.
14. Какой инструмент используется для тестирования модулей на JavaScript? (ПК-2.1)
    - unittest
    - pytest
    - Google Test
    - Mocha/Chai +
15. Какой инструмент используется для профилирования производительности в Python? (ПК-1.2)
    - gprof
    - gdb
    - cProfile +
    - perf
16. Что является основной функцией дебаггера? (ПК-1.2)
    - Оптимизация кода.
    - Проверка производительности.
    - Исправление синтаксических ошибок.
    - Поиск и устранение логических ошибок в программе. +
17. Какой инструмент используется для статического анализа кода на C++? (ПК-1.2)
    - Clang Static Analyzer +
    - valgrind
    - cProfile
    - JSLint
18. Какие методы используются для улучшения читаемости и поддерживаемости кода при рефакторинге? (ПК-2.2)
    - Оптимизация алгоритмов
    - Разделение кода на функции и классы +
    - Удаление комментариев
    - Слияние нескольких функций в одну
19. Что такое интеграционное тестирование? (ПК-3.2)
    - Тестирование всей системы в целом.
    - Тестирование отдельных модулей или компонентов программы.

    - Тестирование взаимодействия между различными модулями или компонентами. +
    - Тестирование производительности программы.
20. Какой инструмент используется для управления зависимостями в Node.js? (ПК-3.1)
    - pip
    - npm +
    - gem
    - cargo
21. Какой инструмент используется для управления зависимостями в C++? (ПК-3.1)
    - npm
    - pip
    - CMake +
    - Maven
22. Какие методы используются для проверки работоспособности выпусков программного продукта? (ПК-3.2)
    - Модульное тестирование
    - Интеграционное тестирование +
    - Рефакторинг
    - Отладка
23. Что такое паттерн проектирования? (ПК-4.2)
    - Готовое решение для часто встречающейся проблемы в проектировании программного обеспечения. +
    - Метод улучшения производительности кода.
    - Способ тестирования программного обеспечения.
    - Инструмент для профилирования кода.
24. Что такое интерфейс в программировании? (ПК-4.2)
    - Конкретная реализация метода в классе.
    - Совокупность методов, которые должны быть реализованы в классе. +
    - Способ наследования свойств и методов.
    - Способ полиморфизма.

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 Вопросы для текущего контроля успеваемости в виде УМЕНИЙ
5 семестр

1. **Вопрос:** Что такое модульное программирование? (УК-1.2)
   - **Ответ:** Модульное программирование - это метод разработки программного обеспечения, при котором программа разбивается на отдельные независимые модули, которые могут быть интегрированы и использованы повторно.
2. **Вопрос:** Какой язык программирования используется для модульного программирования и имеет динамическую типизацию? (УК-1.1)
   - **Ответ:** Python.
3. **Вопрос:** Как импортировать модуль в Python? (ПК-1.1)
   - **Ответ:** Для импорта модуля в Python используется команда import module_name.
4. **Вопрос:** Какой инструмент используется для управления пакетами в Python? (ПК-3.1)
   - **Ответ:** Инструмент для управления пакетами в Python называется pip.
5. **Вопрос:** Как объявить функцию в модуле Python? (ПК-1.1)
   - **Ответ:** Функция в модуле Python объявляется с использованием ключевого слова def. Например: def myFunction():.
6. **Вопрос:** Что такое заголовочный файл в C++? (ПК-1.1)
   - **Ответ:** Заголовочный файл в C++ - это файл с расширением .h, который содержит объявления функций и переменных, используемых для разделения интерфейса и реализации.
7. **Вопрос:** Какой оператор используется для экспорта функций в JavaScript (ES6)? (ПК-1.1)
   - **Ответ:** Для экспорта функций в JavaScript (ES6) используется оператор export.
8. **Вопрос:** Какой менеджер пакетов используется для установки библиотек в Node.js? (ПК-3.1)
   - **Ответ:** Менеджер пакетов, используемый для установки библиотек в Node.js, называется npm.
9. **Вопрос:** Что такое архитектура программного модуля? (ПК-4.2)
   - **Ответ:** Архитектура программного модуля - это структура и организация кода внутри модуля, а также взаимодействие с другими модулями.
10. **Вопрос:** Какие принципы применяются при проектировании модульной архитектуры? (ПК-4.2)
    - **Ответ:** Принципы, применяемые при проектировании модульной архитектуры, включают принцип единой ответственности, принцип открытости/закрытости и принцип разделения интерфейсов.
11. **Вопрос:** Что такое инкапсуляция в объектно-ориентированном программировании? (ПК-1.1)
    - **Ответ:** Инкапсуляция - это способ скрытия деталей реализации класса и предоставления публичного интерфейса для взаимодействия с ним.
12. **Вопрос:** Как определяется интерфейс в программировании? (ПК-4.2)
    - **Ответ:** Интерфейс в программировании определяется как совокупность методов, которые должны быть реализованы в классе.
13. **Вопрос:** Что такое юнит-тестирование? (ПК-2.1)
    - **Ответ:** Юнит-тестирование - это метод тестирования, при котором тестируются отдельные модули или компоненты программы.
14. **Вопрос:** Какой инструмент используется для тестирования модулей на JavaScript? (ПК-2.1)
    - **Ответ:** Для тестирования модулей на JavaScript используется инструмент Mocha/Chai.

15. **Вопрос:** Какой инструмент используется для профилирования производительности в Python? (ПК-1.2)
    - **Ответ:** Для профилирования производительности в Python используется инструмент cProfile.
16. **Вопрос:** Что является основной функцией дебаггера? (ПК-1.2)
    - **Ответ:** Основная функция дебаггера - это поиск и устранение логических ошибок в программе.
17. **Вопрос:** Какой инструмент используется для статического анализа кода на C++? (ПК-1.2)
    - **Ответ:** Для статического анализа кода на C++ используется инструмент Clang Static Analyzer.
18. **Вопрос:** Какие методы используются для улучшения читаемости и поддерживаемости кода при рефакторинге? (ПК-2.2)
    - **Ответ:** Для улучшения читаемости и поддерживаемости кода при рефакторинге используются методы разделения кода на функции и классы, а также улучшение имен переменных и функций.
19. **Вопрос:** Что такое интеграционное тестирование? (ПК-3.2)
    - **Ответ:** Интеграционное тестирование - это метод тестирования, при котором проверяется взаимодействие между различными модулями или компонентами.
20. **Вопрос:** Какой инструмент используется для управления зависимостями в Node.js? (ПК-3.1)
    - **Ответ:** Инструмент для управления зависимостями в Node.js называется npm.
21. **Вопрос:** Какой инструмент используется для управления зависимостями в C++? (ПК-3.1)
    - **Ответ:** Инструмент для управления зависимостями в C++ называется CMake.
22. **Вопрос:** Какие методы используются для проверки работоспособности выпусков программного продукта? (ПК-3.2)
    - **Ответ:** Для проверки работоспособности выпусков программного продукта используются методы модульного и интеграционного тестирования.
23. **Вопрос:** Что такое паттерн проектирования? (ПК-4.2)
    - **Ответ:** Паттерн проектирования - это готовое решение для часто встречающейся проблемы в проектировании программного обеспечения.
24. **Вопрос:** Что такое интерфейс в программировании? (ПК-4.2)
    - **Ответ:** Интерфейс в программировании - это совокупность методов, которые должны быть реализованы в классе.

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.

2.3 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 языки программирования с их основными характеристиками (УК-1.1, УК-1.2)
Чтобы определить правильное соответствие, необходимо понимать основные характеристики языков программирования.

Языки программирования	Характеристики
A - Python	1 - Интерпретируемый язык с динамической типизацией
B - C++	2 - Компилируемый язык с поддержкой объектно-ориентированного программирования
C - JavaScript	3 - Язык сценариев, часто используемый для веб-разработки
D - Java	4 - Компилируемый язык, работающий на виртуальной машине JVM

Правильный ответ: A-1, B-2, C-3, D-4

Задача 2: Соотнесите понятия модульного программирования с их определениями (ПК-1.1, ПК-1.2)
Чтобы определить правильное соответствие, необходимо понимать основные понятия модульного программирования.

Понятия	Определения
A - Модуль	1 - Независимый блок кода, который можно повторно использовать
B - Импорт	2 - Включение функционала одного модуля в другой
C - Экспорт	3 - Предоставление функционала модуля для использования в других модулях
D - Интерфейс	4 - Совокупность методов, которые должны быть реализованы

Правильный ответ: A-1, B-2, C-3, D-4

Задача 3: Соотнесите инструменты с их назначением (ПК-3.1, ПК-3.2)

Чтобы определить правильное соответствие, необходимо понимать назначение различных инструментов разработки.

Инструменты	Назначение
A - pip	1 - Управление пакетами в Python
B - npm	2 - Управление пакетами в Node.js
C - CMake	3 - Управление сборкой и зависимостями в C++
D - Mocha/Chai	4 - Тестирование модулей в JavaScript

Правильный ответ: A-1, B-2, C-3, D-4

Задача 4: Соотнесите принципы проектирования с их описаниями (ПК-4.2)
Чтобы определить правильное соответствие, необходимо понимать основные принципы проектирования.

Принципы	Описания
A - Принцип единой ответственности	1 - Каждый модуль должен решать одну конкретную задачу
B - Принцип открытости/закрытости	2 - Модули должны быть открыты для расширения, но закрыты для изменения
C - Принцип разделения интерфейсов	3 - Интерфейсы должны быть узкими и специфичными
D - Принцип инверсии зависимостей	4 - Модули должны зависеть от абстракций, а не от конкретных реализаций

Правильный ответ: A-1, B-2, C-3, D-4

Задача 5: Соотнесите виды тестирования с их описаниями (ПК-2.1, ПК-3.2)
Чтобы определить правильное соответствие, необходимо понимать различные виды тестирования.

Виды тестирования	Описания
A - Юнит-тестирование	1 - Тестирование отдельных модулей или компонентов
B - Интеграционное тестирование	2 - Тестирование взаимодействия между различными модулями
C - Системное тестирование	3 - Тестирование всей системы в целом
D - Нагрузочное тестирование	4 - Тестирование производительности под нагрузкой

Правильный ответ: A-1, B-2, C-3, D-4

Задача 6: Соотнесите этапы разработки ПО с их описаниями (ПК-4.1, ПК-4.2)
Чтобы определить правильное соответствие, необходимо понимать этапы разработки программного обеспечения.

Этапы разработки	Описания
A - Сбор требований	1 - Определение того, что должно быть реализовано в ПО
B - Проектирование	2 - Разработка архитектуры и структуры ПО
C - Разработка	3 - Написание исходного кода
D - Тестирование	4 - Проверка правильности и работоспособности ПО

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 ЭКЗАМЕНА 5 семестр

1. **Вопрос:** Что такое модульное программирование и каковы его основные преимущества? (УК-1.2)
   - **Ответ:** Модульное программирование - это метод разработки программного обеспечения, при котором программа разбивается на отдельные независимые модули. Основные преимущества включают повторное использование кода, упрощение отладки и тестирования, улучшение структуры кода и облегчение командной работы.
2. **Вопрос:** Какие языки программирования часто используются для разработки программных модулей? (УК-1.1)
   - **Ответ:** Для разработки программных модулей часто используются такие языки программирования, как Python, C/C++ и JavaScript. Эти языки поддерживают создание независимых модулей и библиотек, которые могут быть интегрированы в более крупные системы.
3. **Вопрос:** Как импортировать модуль в Python и какие библиотеки могут использоваться для работы с модулями? (ПК-1.1)
   - **Ответ:** В Python модуль импортируется с помощью команды import. Для работы с модулями и пакетами часто используются библиотеки и фреймворки, такие как os, sys, requests, numpy.
4. **Вопрос:** Какой инструмент используется для управления пакетами в Python и как его использовать? (ПК-3.1)
   - **Ответ:** Для управления пакетами в Python используется инструмент pip. Для установки пакета используется команда установки, для обновления - команда обновления, а для удаления - команда удаления пакета.
5. **Вопрос:** Как создаются и компилируются модули на C/C++? (ПК-1.1)
   - **Ответ:** Модули в C/C++ создаются путем написания заголовочных файлов для объявления функций и переменных и исходных файлов для их реализации. Для

компиляции используется компилятор, например g++, который объединяет эти файлы в исполняемый модуль.

6. **Вопрос:** Что такое заголовочный файл в C++ и как он используется? (ПК-1.1)
    - **Ответ:** Заголовочный файл в C++ содержит объявления функций, классов и переменных, которые могут быть использованы в других файлах. Это позволяет разделить интерфейс и реализацию кода, упрощая его структуру и поддержку.
7. **Вопрос:** Какой метод используется для экспорта функций в JavaScript (ES6) и как импортируются модули? (ПК-1.1)
    - **Ответ:** В JavaScript (ES6) для экспорта функций используется ключевое слово export, а для импорта модулей - ключевое слово import.
8. **Вопрос:** Что такое инкапсуляция в объектно-ориентированном программировании? (ПК-1.1)
    - **Ответ:** Инкапсуляция - это концепция объектно-ориентированного программирования, которая подразумевает сокрытие деталей реализации объекта и предоставление доступа к ним только через публичные методы.
9. **Вопрос:** Как определяется интерфейс в программировании и для чего он используется? (ПК-4.2)
    - **Ответ:** Интерфейс в программировании определяется как совокупность методов, которые должны быть реализованы в классе. Он используется для определения контрактов, которые классы должны соблюдать, что обеспечивает совместимость и взаимозаменяемость компонентов.
10. **Вопрос:** Что такое юнит-тестирование и какие инструменты могут использоваться для его проведения? (ПК-2.1)
    - **Ответ:** Юнит-тестирование - это метод тестирования, при котором тестируются отдельные модули или компоненты программы. Для проведения юнит-тестирования используются инструменты, такие как unittest для Python, Google Test для C++ и Mocha/Chai для JavaScript.
11. **Вопрос:** Какой инструмент используется для профилирования производительности в Python и как он работает? (ПК-1.2)
    - **Ответ:** Для профилирования производительности в Python используется инструмент cProfile. Он анализирует выполнение программы и собирает статистику о времени выполнения различных частей кода, что позволяет выявлять узкие места и оптимизировать производительность.
12. **Вопрос:** Что является основной функцией дебаггера и какие инструменты могут использоваться для отладки? (ПК-1.2)
    - **Ответ:** Основная функция дебаггера - это поиск и устранение логических ошибок в программе. Для отладки программ могут использоваться инструменты, такие как gdb для C/C++, pdb для Python и Chrome DevTools для JavaScript.
13. **Вопрос:** Какие методы используются для улучшения читаемости и поддерживаемости кода при рефакторинге? (ПК-2.2)
    - **Ответ:** Для улучшения читаемости и поддерживаемости кода при рефакторинге используются методы разделения кода на функции и классы, улучшение имен переменных и функций, удаление дублирующегося кода и упрощение сложных участков.
14. **Вопрос:** Что такое интеграционное тестирование и почему оно важно? (ПК-3.2)
    - **Ответ:** Интеграционное тестирование - это метод тестирования, при котором проверяется взаимодействие между различными модулями или компонентами. Оно важно для обеспечения правильной работы системы в целом, так как позволяет выявить ошибки, возникающие при взаимодействии компонентов.
15. **Вопрос:** Какой инструмент используется для управления зависимостями в Node.js и как он работает? (ПК-3.1)

- **Ответ:** Для управления зависимостями в Node.js используется инструмент npm. Он позволяет устанавливать, обновлять и удалять пакеты, а также управлять версиями и зависимостями между пакетами.

16. **Вопрос:** Какие методы используются для проверки работоспособности выпусков программного продукта? (ПК-3.2)
    - **Ответ:** Для проверки работоспособности выпусков программного продукта используются методы модульного и интеграционного тестирования. Эти методы позволяют проверить как отдельные модули, так и их взаимодействие в системе.
17. **Вопрос:** Что такое паттерн проектирования и какие преимущества он предоставляет? (ПК-4.2)
    - **Ответ:** Паттерн проектирования - это готовое решение для часто встречающейся проблемы в проектировании программного обеспечения. Преимущества паттернов проектирования включают повышение читаемости и поддержки кода, облегчение повторного использования решений и улучшение архитектуры программного обеспечения.
18. **Вопрос:** Какие инструменты используются для управления зависимостями в C++ и как они работают? (ПК-3.1)
    - **Ответ:** Для управления зависимостями в C++ используются инструменты, такие как CMake. CMake позволяет описывать сборочные процессы и зависимости, автоматизируя компиляцию и сборку программного обеспечения.
19. **Вопрос:** Что такое интерфейс в программировании и как он обеспечивает совместимость компонентов? (ПК-4.2)
    - **Ответ:** Интерфейс в программировании - это совокупность методов, которые должны быть реализованы в классе. Интерфейсы обеспечивают совместимость компонентов, определяя контракты, которые классы должны соблюдать, что позволяет легко заменять и интегрировать различные модули в системе.
20. **Вопрос:** Какой инструмент используется для тестирования модулей на JavaScript и как он работает? (ПК-2.1)
    - **Ответ:** Для тестирования модулей на JavaScript используется инструмент Mocha/Chai. Mocha предоставляет среду для выполнения тестов, а Chai - библиотеку для утверждений, что позволяет проверять корректность работы модулей.
21. **Вопрос:** Как определяется и используется инкапсуляция в объектно-ориентированном программировании? (ПК-1.1)
    - **Ответ:** Инкапсуляция определяется как концепция объектно-ориентированного программирования, подразумевающая сокрытие деталей реализации объекта и предоставление доступа к ним только через публичные методы. Это позволяет защитить данные и улучшить модульность кода.
22. **Вопрос:** Какие методы используются для улучшения структуры кода при рефакторинге? (ПК-2.2)
    - **Ответ:** Для улучшения структуры кода при рефакторинге используются методы, такие как разделение кода на функции и классы, улучшение имен переменных и функций, удаление дублирующегося кода и упрощение сложных участков.
23. **Вопрос:** Какой инструмент используется для профилирования производительности в Python и какие данные он собирает? (ПК-1.2)
    - **Ответ:** Для профилирования производительности в Python используется инструмент cProfile. Он собирает данные о времени выполнения различных частей кода, что позволяет выявлять узкие места и оптимизировать производительность программы.
24. **Вопрос:** Что такое интеграционное тестирование и как оно помогает в обеспечении качества программного обеспечения? (ПК-3.2)

- **Ответ:** Интеграционное тестирование - это метод тестирования, при котором проверяется взаимодействие между различными модулями или компонентами. Оно помогает в обеспечении качества программного обеспечения, выявляя ошибки, возникающие при интеграции компонентов, и гарантируя корректную работу системы в целом.

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.