Федеральное государственное бюджетное образовательное учреждение высшего образования

РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА и ГОСУДАРСТВЕННОЙ СЛУЖБЫ при ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

СЕВЕРО-ЗАПАДНЫЙ ИНСТИТУТ УПРАВЛЕНИЯ

Факультет среднего профессионального образования

УТВЕРЖДЕНА
решением цикловой (методической) комиссии общепрофессиональных дисциплин и профессиональных модулей специальностей 09.02.07 Информатика и вычислительная техника
Протокол от 31.10.2025 № 2

РАБОЧАЯ ПРОГРАММА ПРАКТИКИ

УП.02.01 Учебная практика

Специальность – 09.02.11 Разработка и управление программным обеспечением

Профиль – на базе основного общего образования

Квалификация – программист

Форма обучения – очная

Год набора – 2026

Санкт-Петербург 2025 год

Автор-составитель: Вилков Владислав Евгеньевич, преподаватель 1 категории ФСПО СЗИУ РАНХиГС.

СОДЕРЖАНИЕ

1 Общие положения

1.1 Область применения программы

Рабочая программа учебной практики УП.02.01 является частью основной профессиональной образовательной программы в соответствии с ФГОС по специальности СПО 09.02.11 «Разработка и управление программным обеспечением».

1.2 Место практики в структуре основной профессиональной образовательной программы

Учебная практика УП.02.01 входит в состав профессионального модуля ПМ.02 «Разработка и интеграция модулей программного обеспечения» и представляет собой неотъемлемый элемент профессиональной подготовки обучающихся. Она базируется на освоении дисциплин «Разработка программных модулей», «Осуществление интеграции программных модулей», «Поддержка и тестирование программных модулей», «Математическое моделирование», «Численные методы» и «Безопасность программного обеспечения» и реализуется на 3 курсе обучения в 6 семестре.

1.3 Цели и задачи учебной практики

Цель учебной практики «УП.02.01 Учебная практика»: формирование у студентов систематизированных знаний и практических навыков в области разработки и тестирования программного обеспечения, включая проектирование модулей с учётом технического задания и визуализацию архитектурных решений, определение интерфейсов и взаимодействия модулей в системе, создание программных модулей и работу с API и веб-сервисами для их интеграции. В рамках практики студенты осваивают работу с интеграционными платформами и инструментами, отладку и комплексное тестирование ПО — от формирования тестовых сценариев и подготовки тестовых платформ до выполнения тестовых процедур на тестовые данные и оценки объёма тестирования для определения необходимых ресурсов. Отдельное внимание уделяется навыкам технической документации: созданию документации для модулей, документированию кода, API и интерфейсов, а также работе со специализированным программным обеспечением для документирования программного кода и формированию отчётности о подготовке к выполнению заданий на тестирование в соответствии с установленными регламентами.

1.4 Планируемые результаты обучения по учебной практике

Перечень профессиональных компетенций

Код и наименование компетенции	Умения	Знания	Навыки
ПК 2.1 Проектировать модули программного обеспечения	− проектировать модули, соответствующие бизнес-задачам; − создавать архитектурные диаграммы и документацию; − определять структуру и интерфейсы модулей; − анализировать требования к модулю и определять его функциональность; − проектировать архитектуру модуля, включая выбор подходящих паттернов проектирования и структуры данных; − создавать диаграммы классов, последовательностей и прочих диаграмм для визуализации проектируемого модуля; − выбирать подходящие языки программирования и технологии для реализации модуля; − проектировать интерфейсы программного обеспечения для взаимодействия с другими модулями и системами; − учитывать требования к масштабируемости, производительности и безопасности при проектировании модуля;	− основные принципы проектирования модулей программного обеспечения; − языки программирования и технологии для реализации модулей; − паттерны проектирования и структуры данных для создания эффективных и масштабируемых модулей; − методы анализа требований и способов определения функциональности модуля; − принципы создания интерфейсов для взаимодействия с другими модулями и системами; − принципы обеспечения безопасности, производительности и масштабируемости при проектировании модулей; методы анализа и оптимизации проектируемых модулей для повышения их эффективности и качества.	− проектирования модулей ПО с учетом требований заказчика; − создания архитектурных диаграмм и спецификаций модулей; определения интерфейсов и взаимодействия модулей в системе.

	проводить анализ и оптимизацию проектируемого модуля для повышения его эффективности и качества		
ПК 2.2 Разрабатывать модули программного обеспечения	− разрабатывать модули программного обеспечения с использованием различных языков программирования и технологий; − применять паттерны проектирования и структуры данных для создания эффективных и масштабируемых модулей; − анализировать требования и определять функциональность модуля; − создавать интерфейсы для взаимодействия с другими модулями и системами; − обеспечивать безопасность, производительность и масштабируемость при разработке модулей; − оптимизировать проектируемые модули для повышения их эффективности и качества; − работать с системой контроля версий; − улучшать производительность модулей, выявляя и устраняя узкие места; − проводить анализ и мониторинг	− язык программирования, основные конструкции, синтаксис; − паттерны проектирования; − структуры данных; − принципы создания интерфейсов для взаимодействия с другими модулями и системами, таких как REST API, SOAP; − работу с инструментальным программным обеспечением; − методы оптимизации кода и алгоритмов; − эффективные алгоритмы и структуры данных для повышения производительности; − многопоточность в программных модулях; − методы оптимизации сетевых протоколов для ускорения обмена данными; − кэширование данных; − управление памятью; техники повышения производительности программного обеспечения	− создания модулей программного обеспечения на различных языках программирования; − отладки и тестирования разработанных модулей; − применения структурного и объектно-ориентированного программирования; − оптимизации кода и алгоритмов программных модулей для увеличения производительности; мониторинга и анализа производительности приложений.

	производительности приложений; применять инструменты для рефакторинга и оптимизации программного кода.		
ПК 2.3 Выполнять интеграцию модулей и компонентов программного обеспечения	− интегрировать модули и компоненты, обеспечивая их взаимодействие; − работать с API и устанавливать соединения между компонентами; − отслеживать и устранять конфликты и ошибки интеграции; − анализировать и определять зависимости между модулями и компонентами; работать с различными форматами данных и протоколами передачи данных	− общие принципы функционирования аппаратных, программных и программно-аппаратных средств администрируемой информационно-коммуникационной системы; − международные стандарты локальных вычислительных сетей; − методы и подходы к интеграции модулей и компонентов; − принципы версионирования и управления изменениями при интеграции; принципы безопасности при интеграции модулей и компонентов	− интеграции программных модулей и компонентов в единое программное решение; − работы с API и веб-сервисами для взаимодействия между модулями; − работы с интеграционными платформами и инструментами; обеспечения совместимости и стабильности системы
ПК 2.4 Выполнять тестирование и отладку программного обеспечения	− анализировать требования к программному обеспечению и составлять планы тестирования; − создавать тестовые сценарии и тест-кейсы для проверки функциональности и соответствия требованиям; − выполнять тестирование программного обеспечения вручную и автоматизировать процесс тестирования; − анализировать результаты тестирования и	− принципы и методы тестирования программного обеспечения; − основы программирования и архитектуры программного обеспечения; − основы баз данных и SQL-запросов; − инструменты для автоматизации тестирования; − основы разработки и отладки программного обеспечения на разных языках программирования; − понятие дефекта программного обеспечения;	− отладки программного обеспечения на уровне программных модулей; − тестирования программного обеспечения; − формирования тестовых сценариев; − подготовки тестовых платформ (установка операционной системы, дополнительного ПО и другого по необходимости); − оценки объема тестирования ПО с целью определения необходимых ресурсов для его выполнения; − настройки тестовой среды и аппаратных средств для выполнения

	документировать найденные ошибки; − разрабатывать стратегии отладки и исправлять ошибки в программном обеспечении; − выполнять модульные тесты с использованием инструментов тестирования, в том числе автоматизированного тестирования; − использовать системы контроля дефектов ПО; составлять отчет о выполнении тестирования ПО	− критерии качества ПО; − виды и типы тестирования ПО; − техники ручного тестирования; − техники автоматизированного тестирования; − жизненный цикл дефекта ПО; − принципы работы в системе контроля дефектов; основные понятия о качестве ПО	тестирования ПО в соответствии с заданием на тестирование в пределах своей компетенции; − формирования и представления отчетности о подготовке к выполнению задания на тестирование ПО в соответствии с установленными регламентами; выполнения тестовых процедур на тестовых данных
ПК 2.5 Осуществлять документирование программных модулей программного обеспечения	− описывать функциональность модулей в документации; − создавать диаграммы для иллюстрации работы модулей; − программировать с использованием комментариев для документирования кода; − использовать специальные метки/теги для отметки важных частей кода в документации; − вести журнал изменений и фиксировать обновления программных модулей; − разбивать модули на логические блоки и описывать каждый блок отдельно; − включать в документацию особенности модулей, такие как ограничения,	− стандарты технической документации; − принципы документирования программного обеспечения; инструменты для создания технической документации и комментирования кода	− создания технической документации для модулей; − документирования кода, API и интерфейсов; работы со специализированным ПО по документированию программного кода

	уязвимости или оптимальные настройки; проводить регулярное обновление документации при изменении модулей или добавлении нового функционала.		

В результате освоения учебной практики студент должен:

иметь практический опыт	– Проектировать модули ПО с учётом технического задания; – Визуализировать и описывать архитектурные решения; – Определять интерфейсы и взаимодействие модулей в системе; – Создавать модули программного обеспечения; – Работать с API и веб-сервисами для взаимодействия между модулями; – Работать с интеграционными платформами и инструментами; – Отлаживать ПО на уровне программных модулей; – Тестировать программное обеспечение; – Формировать тестовые сценарии; – Подготавливать тестовые платформы; – Оценивать объём тестирования ПО для определения ресурсов; – Формировать и представлять отчётность о подготовке к тестированию ПО; – Выполнять тестовые процедуры на тестовых данных; – Создавать техническую документацию для модулей; – Документировать код, API и интерфейсы; – Работать со специализированным ПО для документирования кода.
уметь	– Анализировать техническое задание для проектирования модулей ПО; – Применять архитектурные шаблоны (MVC, MVVM, MVP) при разработке модулей; – Использовать системы контроля версий при разработке модулей; – Реализовывать паттерны проектирования; – Разрабатывать модули для работы с массивами, коллекциями, датой и временем; – Организовывать асинхронный и параллельный вызов методов; – Работать с файловой системой (навигация, чтение/запись файлов, потоки); – Обрабатывать текстовые данные (кодирование/декодирование, регулярные выражения); – Проектировать GUI (окна, элементы управления, обработка событий); – Разрабатывать модули для взаимодействия с БД (CRUD-операции, запросы); – Оптимизировать код и алгоритмы для повышения производительности; – Обеспечивать безопасность при разработке модулей (аутентификация, защита от уязвимостей); – Интегрировать модули через REST API, WebSocket, микросервисы; – Настраивать логирование и мониторинг приложений; – Контейнеризировать приложения и развёртывать решения; – Проводить тестирование ПО (модульное, интеграционное, нагрузочное); – Анализировать качество модулей с использованием метрик; – Обрабатывать исключения и ошибки в коде; – Рефакторить и оптимизировать код; – Составлять техническую документацию и спецификации.

знать	– Принципы ООП и модульной архитектуры приложений; – Архитектурные шаблоны и паттерны проектирования; – Основы работы с системами контроля версий; – Принципы работы с массивами, коллекциями, строками, датой и временем; – Асинхронную и параллельную модели программирования; – Принципы работы с файловой системой и потоками данных; – Методы работы с текстовыми данными; – Технологии разработки GUI; – Принципы взаимодействия приложения с БД, основы ORM и CRUD; – Методы обеспечения безопасности, производительности и масштабируемости модулей; – Протоколы и стандарты интеграции (REST API, WebSocket и др.); – Инструменты логирования, мониторинга и контейнеризации; – Виды и методы тестирования ПО; – Метрики качества программных модулей; – Принципы отладки кода и обработки исключений; – Стандарты документирования ПО; – Основные угрозы безопасности (SQL-инъекции, XSS, CSRF) и методы защиты; – Основы математического моделирования и численных методов; – Принципы криптографии и безопасной аутентификации.

2 Структура и содержание учебной практики

2.1 Объем учебной практики и виды работ

Виды учебной работы	Объем учебной работы, час.
Учебная нагрузка обучающихся всего, в том числе:	144 (4 недели)
практические занятия	144
промежуточная аттестация	-
Форма промежуточной аттестации	Зачет с оценкой

Учебная практика проводится концентрировано, в соответствии с графиком учебного процесса на текущий учебный год.

Учебная практика проводится преподавателями специальности «09.02.11 Разработка и управление программным обеспечением» на учебной базе ФСПО в форме практических занятий. По результатам прохождения учебной практики обучающиеся составляют отчёт о прохождении учебной практики.

Структура, содержание и виды работ учебной практики приведены в Таблице 2.2.

Таблица 2.2. Тематический план и содержание учебной практики

п/п	Наименование тем (разделов)	Содержание и виды работ	Распределение часов ПР	Формируемые компетенции	Формы текущего контроля
1	Тема 1.1. Разработка программных модулей	**Содержание:** — цели и задачи практики. Требования охраны труда. Инструктаж по технике безопасности во время прохождения практики. — модульная архитектура построения приложений: принципы, преимущества, примеры приложений; — архитектурные шаблоны, применяемые при разработке программных модулей (MVC, MVVM, MVP); — инструменты разработки приложений с модульной архитектурой, системы контроля версий; — работа с библиотеками: применение стандартных библиотек, создание библиотек; — базовые принципы работы с массивами, коллекциями, строками, датой и временем; — паттерны проектирования (отношения между классами и объектами: наследование, реализация, ассоциация, композиция, агрегация; интерфейсы, абстрактные классы; порождающие, структурные и поведенческие паттерны); — система ввода-вывода, средства доступа к файлам и папкам файловой системы (чтение/запись, сжатие потоков, изолированное хранение); — работа со строками, регулярными выражениями, кодирование/декодирование текста; — асинхронная модель программирования (пул потоков, шаблон асинхронного вызова методов, синхронизация вызывающего потока); — параллельное программирование (создание задачи, методы ожидания выполнения задачи, лямбда-выражения, создание продолжения задачи, возврат значений, отмена задачи). **Виды работ:** 1. Разработка программных модулей для работы с массивами (с использованием системы контроля версий). 2. Разработка программных модулей для работы с коллекциями (с использованием системы контроля версий). 3. Разработка программных модулей для работы с датой и временем (с использованием системы контроля версий). 4. Разработка программных модулей с использованием паттернов проектирования (с использованием системы контроля версий). 5. Навигация по файловой системе: чтение и запись файлов, работа с потоками, изолированным хранилищем. 6. Работа с большими объёмами текста: кодирование/декодирование строк, построение	30	ПК 2.1 ПК 2.2 ПК 2.3 ПК 2.4 ПК 2.5	ПЗ,О

		регулярных выражений, чтение/запись файлов в разных кодировках. 7. Организация асинхронного вызова методов. 8. Создание программного модуля, выполняющего методы в рамках параллельных задач.			
2	Тема 1.2. Осуществление интеграции программных модулей	**Содержание:** — алгоритмы и структуры данных, оценка сложности алгоритмов (асимптотическая оценка, большие О -нотации, временная и пространственная сложность); — основные структуры данных (массив, связный список, стек, очередь, операции вставки/поиска/удаления, представление данных в памяти); — алгоритмы сортировки и поиска, основы рекурсии; — хеш-таблицы и хеш-функции (коллизии, методы хеширования, сжатие данных, эффективность хеш-структур); — деревья и графы (представление, поиск в глубину/ширину, алгоритм Дейкстры, деревья поиска и обхода); — жадные алгоритмы и динамическое программирование; — алгоритмы работы с текстовыми данными (поиск подстроки, алгоритмы Кнута–Морриса–Пратта, Бойера–Мура, Рабина–Карпа, операции с динамическими строками, триальные деревья); — кучи и очереди (очереди с приоритетом, применение кучи). **Виды работ:** 1. Оценка сложности алгоритмов. 2. Применение рекурсивных алгоритмов. 3. Работа с алгоритмами сортировки и поиска. 4. Создание хеш-таблиц и использование для ускорения поиска данных. 5. Нахождение кратчайших путей в графах (алгоритм Дейкстры). 6. Решение задачи о рюкзаке (метод динамического программирования). 7. Реализация строковых алгоритмов. 8. Реализация приоритетных очередей для планирования задач.	28		ПЗ,О
3	Тема 1.3. Поддержка и тестирование программных модулей	**Содержание:** — основные принципы проектирования модулей ПО, методы анализа требований и определения функциональности модуля; — методы анализа и оптимизации проектируемых модулей; — декомпозиция задачи на подзадачи, создание спецификаций модуля;	22		ПЗ,О

		— принципы обеспечения безопасности, производительности и масштабируемости при проектировании модулей; — принципы проектирования классов (инкапсуляция, наследование, полиморфизм); — применение диаграмм классов и компонентов для проектирования внутренней структуры модуля. **Виды работ:** 1. Анализ требований к модулю и определение функциональности. 2. Создание спецификации программного модуля. 3. Проектирование требований к внутренней структуре модуля (диаграммы классов, паттерны проектирования). 4. Проектирование требований к организации компонентов модуля (диаграммы компонентов). 5. Проектирование интерфейсов ПО для взаимодействия с другими модулями и системами. 6. Анализ и оптимизация проектируемого модуля.			
4	Тема 1.4. Математическое моделирование	**Содержание:** — понятие модели, классификация моделей, математическая модель, типы математических моделей; — принципы построения математических моделей, основные этапы математического моделирования. **Виды работ:** 1. Построение простейших математических моделей.	6		ПЗ,О
5	Тема 1.5. Численные методы	**Содержание:** — приближённые числа и действия над ними (погрешности, значащие цифры); — численные методы решения алгебраических и трансцендентных уравнений (метод половинного деления, метод простой итерации, методы Ньютона); — численные методы решения систем линейных алгебраических уравнений (метод Гаусса, метод простой итерации, метод Зейделя); — интерполяция и экстраполяция функций (интерполяционный многочлен Лагранжа, интерполяционные формулы Ньютона, интерполяция сплайнами); — численное интегрирование (квадратурные формулы Ньютона–Котеса, формула Гаусса); — численные методы решения обыкновенных дифференциальных уравнений (метод Эйлера, метод Рунге–Кутта); — численное решение задач оптимизации (методы минимизации функции одной и двух переменных). **Виды работ:** **1**. Вычисление погрешностей приближённых значений. 2. Решение алгебраических и трансцендентных уравнений приближёнными методами.	22		ПЗ,О

		3. Решение систем линейных алгебраических уравнений методом Гаусса. 4. Составление интерполяционных формул Лагранжа и Ньютона. 5. Вычисление интегралов при помощи формул Ньютона–Котеса и Гаусса. 6. Нахождение решений обыкновенных дифференциальных уравнений методами Эйлера и Рунге–Кутта. 7. Нахождение экстремумов функций одной и двух переменных приближёнными методами**.**			
6	Тема 1.6. Безопасность программного обеспечения	**Содержание:** — основы кибербезопасности и уязвимости ПО; — модели угроз и анализ рисков; — уязвимости веб-приложений (OWASP Top 10); — безопасная аутентификация и авторизация; — криптография для разработчиков; — принципы безопасного проектирования архитектуры; — криптографические протоколы и их реализация; — криптография в мобильных приложениях, веб-приложениях, облачных средах. **Виды работ:** 1. Анализ кода на наличие уязвимостей (ручной review). 2. Эксплуатация и защита от SQL-инъекций. 3. Создание и предотвращение XSS-атак. 4. Реализация защиты от CSRF-атак. 5. Составление модели угроз для типового веб-приложения. 6. Настройка безопасной аутентификации (JWT, refresh-токены). 7. Реализация RBAC-системы с разделением привилегий. 8. Шифрование данных (AES, RSA). 9. Хэширование паролей (bcrypt, Argon2). 10. Анализ сетевого трафика (Wireshark). 11. Сканирование уязвимостей (OWASP ZAP, Burp Suite). 12. Настройка HTTPS и создание самоподписанных сертификатов. 13. Защита от brute-force атак. 14. Безопасная работа с файлами и десериализацией данных.	36		ПЗ,О
		Итого часов:	144		

2.3 Регламент распределения видов работ по учебной практике с ДОТ

Учебная практика реализуется с применением дистанционных образовательных технологий (ДОТ). Распределение видов учебной работы, форматов текущего контроля представлены в Таблице 2.3.

Таблица 2.3. — Распределение видов учебной работы и текущей аттестации

Вид учебной работы	**Формат проведения**
Практические занятия	Частично с применением ДОТ

Доступ к системе дистанционных образовательных программ осуществляется каждым обучающимся самостоятельно с любого устройства на портале: https://sziu-de.ranepa.ru, в соответствии с их индивидуальным паролем и логином к личному кабинету/ профилю.

Текущий контроль, проводимый в системе дистанционного обучения, оцениваются как в системе дистанционного обучения, так и преподавателем вне системы.

Доступ к материалам практический занятий предоставляется в течение всего семестра по мере прохождения освоения программы. Доступ к каждому виду работ и количество попыток на выполнение задания предоставляется ограниченное время согласно регламенту учебной практики, опубликованному в системе дистанционного обучения. Преподаватель оценивает выполненные обучающимися работы не позднее 14 рабочих дней после окончания срока выполнения.

3 Материалы текущего контроля успеваемости и промежуточной аттестации обучающихся

3.1 Формы и методы текущего контроля успеваемости и промежуточной аттестации обучающихся

Текущий контроль и оценка результатов учебной практики УП.01.01 осуществляется преподавателем в процессе проведения учебной практики на практических занятиях.

Для контроля успеваемости используется метод опроса (О) в форме теоретических вопросов и практический метод (ПЗ) в форме выполнения практических заданий.

Опрос – как основной вид устной проверки, может использоваться фронтальный и индивидуальный. Студент отвечает на теоретический вопрос каждой темы, количество вопросов по теме – от одного до трёх, вопрос определяет преподаватель учебной практики.

Критерии оценивания опроса

Оценки «отлично» заслуживает студент, обнаруживший глубокое знание материала, умение свободно выполнять задания, понимающий взаимосвязь основных понятий;

Оценки «хорошо» заслуживает студент, обнаруживший полное знание материала; успешно выполняющий предусмотренные задания; и допустивший незначительные ошибки: неточность фактов, стилистические ошибки;

Оценки «удовлетворительно» заслуживает студент, обнаруживший знания основного материала в объеме, необходимом для дальнейшего изучения дисциплины. Справляющийся с выполнением заданий; допустивший погрешности в ответе, но обладающий необходимыми знаниями для их устранения под руководством преподавателя;

Оценки «неудовлетворительно» заслуживает студент, обнаруживший существенные пробелы в знании основного материала; не справляющийся с выполнением заданий, допустивший серьезные погрешности в ответах, нуждающийся в повторении основных разделов курса под руководством преподавателя.

Критерии оценивания практических заданий:

«отлично» – верно и полностью выполнено задание, выбрано наиболее полное и рациональное решение задачи, верно отвечает на вопросы по теоретической части практической работы;

«хорошо» – верно и полностью выполнено задание, имеются погрешности в решении, имеются неточности в ответах на вопросы по работе;

«удовлетворительно» – задание выполнено не полностью, имеются значительные погрешности или приняты нерациональные способы решения; затрудняется в ответах на вопросы по работе;

«неудовлетворительно» – неправильно выполнено задание, не отвечает на вопросы по работе.

Формы текущего контроля

№ п/п	Название темы	Формы текущего контроля успеваемости
1	Тема 1.1. Разработка программных модулей	ПЗ, О
2	Тема 1.2. Осуществление интеграции программных модулей	ПЗ, О
3	Тема 1.3. Поддержка и тестирование программных модулей	ПЗ, О
4	Тема 1.4. Математическое моделирование	ПЗ, О
5	Тема 1.5. Численные методы	ПЗ, О
6	Тема 1.6. Безопасность программного обеспечения	ПЗ, О

Примечание. В столбце «Форма текущего контроля успеваемости, промежуточной аттестации» перечисляются все используемые в учебном процессе по данной дисциплине формы контроля освоения материала. (ПЗ – практическое задание, О – опрос).

Формы промежуточного контроля

Форма промежуточной аттестации по учебной практике — зачёт с оценкой в 6 семестре.

Метод контроля:

письменный — отчёт по учебной практике УП.02.01.;

устный — защита отчёта.

Критерии оценивания промежуточной аттестации:

Оценка «зачтено-отлично»: письменный отчёт оформлен без ошибок: соблюдены все требования к структуре (титульный лист, введение, основная часть, заключение, список литературы, приложения), отсутствуют ошибки в оформлении и содержании. Материал глубоко раскрывает все аспекты практики, включая индивидуальное задание. Выводы логичны, обоснованы, подкреплены фактами. Использованы первоисточники, приложения (схемы, таблицы, фотоматериалы) уместны и информативны. Отчёт сдан в срок. При защите студент демонстрирует превосходное владение материалом: чётко и грамотно излагает содержание отчёта, уверенно использует профессиональную терминологию. Даёт исчерпывающие ответы на все вопросы, включая дополнительные, показывает глубокое понимание темы, способность к анализу и формулировке выводов. Речь структурирована, убедительна, при необходимости сопровождается иллюстративными материалами.

Оценка «зачтено-хорошо»: письменный отчёт в целом соответствует методическим указаниям, однако допущены 1–2 незначительных недочёта в оформлении (например, мелкие ошибки в приложениях). Содержание отражает ключевые аспекты практики, но анализ некоторых вопросов проведён не в полной мере. Выводы корректны, но недостаточно развёрнуты. Использованы необходимые первоисточники, однако их анализ не всегда глубок. Отчёт сдан своевременно или с незначительным опозданием, согласованным с руководителем. Во время защиты студент грамотно излагает суть отчёта, но допускает 1–2 небольшие речевые или логические ошибки. На основные вопросы отвечает правильно, однако при ответе на дополнительные вопросы испытывает небольшие затруднения. Показывает хорошее понимание темы, но не всегда способен глубоко проанализировать отдельные аспекты. Терминологию использует уверенно, но иногда допускает неточности.

Оценка «зачтено-удовлетворительно»: письменный отчёт имеет существенные недостатки в оформлении: ошибки в структуре, форматировании, оформлении приложений.

Содержание неполное: упущены отдельные важные аспекты практики, индивидуальное задание выполнено поверхностно или не выполнено. Выводы формальные, не отражают сути работы. Первоисточники использованы фрагментарно, анализ поверхностный. Отчёт сдан с опозданием, не согласованным с руководителем. При защите студент излагает содержание отчёта неполно, сбивчиво, с существенными логическими или речевыми ошибками. Ответы на вопросы неполные, содержат ошибки; на дополнительные вопросы ответить не может. Понимание темы поверхностное, анализ формальный. Терминология используется ограниченно, с ошибками. Тем не менее, базовый уровень понимания материала и выполнения заданий прослеживается.

Оценка «не зачтено-неудовлетворительно»: письменный отчёт не соответствует методическим указаниям, содержит множественные ошибки в оформлении и содержании. Ключевые аспекты практики не раскрыты, анализ отсутствует, выводы формальны или отсутствуют. Первоисточники не использованы, приложения либо отсутствуют, либо не соответствуют теме. Отчёт не сдан в срок или представлен в неполном объёме. Возможны случаи плагиата или использования недостоверных данных. Во время защиты студент не может чётко изложить содержание отчёта: речь несвязная, содержит множественные ошибки. Не отвечает на поставленные вопросы или даёт ошибочные ответы. Не демонстрирует понимания темы, не способен анализировать и делать выводы. Профессиональную терминологию не использует или использует неправильно. Материал не освоен, практика фактически не выполнена или выполнена ненадлежащим образом.

3.2 Оценочные средства текущего контроля успеваемости обучающихся

Примеры типовых заданий для учебной практики

Тема 1.1. Разработка программных модулей

Примеры типовых заданий:

1. Задание на анализ архитектурного решения:

- проанализировать архитектуру 1–2 реальных приложений (например, социальной сети или CRM-системы);
- выделить используемые архитектурные шаблоны (MVC, MVVM, MVP);
- сравнить преимущества и недостатки выбранной архитектуры с альтернативными вариантами;

- представить результаты в виде отчёта (5–7 страниц) с блок-схемами и ссылками на источники.

2. <u>Практическое задание по работе с библиотеками:</u>

- создать библиотеку для работы с датами и временем, включающую функции: расчёт разницы между датами, определение дня недели, форматирование даты;
- подключить созданную библиотеку в консольное приложение и продемонстрировать её работу на 3–5 примерах;
- оформить код с использованием системы контроля версий (Git), сделать 3 коммита с комментариями.

3. <u>Задание по паттернам проектирования:</u>

- реализовать паттерн «Наблюдатель» (Observer) для системы уведомлений в мессенджере;
- описать классы и их взаимодействие в виде UML-диаграммы;
- написать юнит-тесты для проверки работы паттерна (минимум 3 теста).

4. <u>Практикум по асинхронному программированию:</u>

- создать консольное приложение, выполняющее 3 долгоработающие задачи параллельно (например, загрузка файлов, обработка данных, запрос к API);
- использовать пул потоков и шаблон асинхронного вызова методов;
- измерить время выполнения задач с синхронизацией и без, сравнить результаты.

5. <u>Кейс по параллельному программированию:</u>

- разработать модуль для расчёта статистических показателей (среднее, медиана, мода) для большого набора данных (100 000+ элементов);
- реализовать расчёт с использованием параллельных задач и лямбда-выражений;
- оптимизировать код для минимизации времени ожидания выполнения задач.

<u>Теоретические вопросы для устного опроса:</u>

1. В чём отличие модульной архитектуры от монолитной? Приведите 2–3 примера приложений с модульной архитектурой.
2. Опишите шаблон MVC. Какие проблемы он решает?
3. Что такое система контроля версий? Назовите 3 популярных инструмента СКВ и их ключевые особенности.
4. Перечислите основные паттерны проектирования. К какой категории относится паттерн «Одиночка» (Singleton)?

5. Что такое асинхронная модель программирования? Приведите пример её применения.

6. Как синхронизировать вызывающий поток при асинхронном вызове метода?

7. Что такое лямбда-выражение? Приведите синтаксис лямбда-выражения на языке C#.

8. Как создать и управлять задачей в параллельном программировании? Назовите методы класса Task.

Тема 1.2. Осуществление интеграции программных модулей

Примеры типовых заданий:

1. Задание на оценку сложности алгоритма:

- проанализировать алгоритм сортировки пузырьком;
- определить его временную и пространственную сложность;
- сравнить с алгоритмом быстрой сортировки (QuickSort) по эффективности.

2. Практикум по работе с хеш-таблицами:

- реализовать хеш-таблицу с линейным пробированием на языке Java;
- добавить методы вставки, поиска и удаления элемента;
- протестировать работу таблицы на наборе из 1000 случайных ключей.

3. Задание по поиску в графах:

- построить граф дорожной сети города (5–7 узлов, 8–10 рёбер) в виде матрицы смежности;
- реализовать алгоритм Дейкстры для нахождения кратчайшего пути между двумя узлами;
- вывести результат в виде последовательности рёбер и общей длины пути.

4. Кейс по динамическому программированию:

- решить задачу о рюкзаке (Knapsack Problem) для 5 предметов с разными весами и стоимостями;
- использовать таблицу для хранения промежуточных результатов;
- представить решение в виде псевдокода или кода на языке программирования.

5. Практическое задание по строковым алгоритмам:

- реализовать алгоритм Кнута–Морриса–Пратта для поиска подстроки в строке;
- протестировать алгоритм на 3 тестовых случаях (включая крайние случаи: подстрока в начале, в конце, отсутствует);
- сравнить время выполнения с наивным алгоритмом поиска.

Теоретические вопросы для устного опроса:

1. Что такое асимптотическая оценка сложности алгоритма? Какие обозначения используются (O, Ω, Θ)?
2. Перечислите основные структуры данных. В чём отличие связного списка от массива?
3. Как работает хеш-функция? Что такое коллизия и как её разрешить?
4. Опишите алгоритм сортировки вставками. Какова его сложность?
5. Что такое дерево поиска? Как выполнить обход дерева в порядке уровней?
6. В чём разница между жадными алгоритмами и динамическим программированием?
7. Как представить граф в памяти компьютера? Назовите 2 способа представления.
8. Что такое приоритетная очередь? Приведите пример её использования.

Тема 1.3. Поддержка и тестирование программных модулей

Примеры типовых заданий:

1. Задание на анализ требований:
   - проанализировать требования к модулю авторизации в веб-приложении;
   - выделить функциональные и нефункциональные требования;
   - составить таблицу требований с колонками: «ID», «Описание», «Приоритет», «Статус».
2. Практикум по созданию спецификации модуля:
   - разработать спецификацию для модуля работы с корзиной покупок в интернет-магазине;
   - включить разделы: назначение, функциональные возможности, интерфейсы, требования к надёжности;
   - использовать шаблоны документации (например, IEEE 830).
3. Задание по проектированию классов:
   - спроектировать классы для системы учёта студентов (Студент, Группа, Предмет, Оценка);
   - определить отношения между классами (наследование, ассоциация, агрегация);
   - построить диаграмму классов UML.
4. Кейс по оптимизации модуля:

- проанализировать код модуля обработки изображений (предоставлен преподавателем);
- выявить «узкие места» (неэффективные циклы, избыточные операции);
- предложить способы оптимизации и оценить ожидаемый прирост производительности.

5. Практическое задание по тестированию:

- написать 5 юнит-тестов для метода расчёта скидки в интернет-магазине (разные сценарии: обычная покупка, акция, купон);
- использовать фреймворк JUnit (Java) или NUnit (C#);
- запустить тесты и проанализировать результаты (отчёт о покрытии кода).

Теоретические вопросы для устного опроса:

1. Что такое спецификация модуля? Какие разделы она включает?
2. Перечислите методы анализа требований. В чём отличие функциональных требований от нефункциональных?
3. Что такое декомпозиция задачи? Приведите пример декомпозиции для модуля расчёта зарплаты.
4. Опишите принципы проектирования классов (инкапсуляция, наследование, полиморфизм).
5. Что такое диаграмма классов UML? Какие элементы она включает?
6. Как оценить эффективность модуля? Назовите 2–3 метрики качества кода.
7. Что такое юнит-тест? Какие преимущества он даёт при разработке ПО?
8. Перечислите этапы тестирования программного модуля.

Тема 1.4. Математическое моделирование

Примеры типовых заданий:

1. Задание на построение модели:

- построить математическую модель роста популяции бактерий (учитывать скорость размножения и смертность);
- реализовать модель в Excel или Python (библиотека NumPy);
- построить график зависимости численности популяции от времени (50 единиц времени).

2. Кейс по математическому моделированию:

- смоделировать процесс распространения вируса в городе (население 1 млн человек) с учётом параметров: заразность, инкубационный период, процент вакцинированных;

- использовать дискретную модель (по дням);
- проанализировать результаты и предложить меры по сдерживанию эпидемии.

Теоретические вопросы для устного опроса:

1. Что такое математическая модель? Приведите 2–3 примера математических моделей из разных областей.
2. Перечислите этапы математического моделирования.
3. В чём отличие детерминированной модели от стохастической?
4. Что такое параметры модели? Приведите пример параметров для модели экономического роста.
5. Как проверить адекватность математической модели?

Тема 1.5. Численные методы

Примеры типовых заданий:

1. Задание по решению систем линейных уравнений:

- решить систему трёх линейных уравнений методом Гаусса (вручную и с помощью Python/MATLAB);
- представить расширенную матрицу системы, показать все шаги приведения к треугольному виду;
- проверить правильность решения подстановкой найденных значений в исходную систему.

2. Практикум по интерполяции:

- даны 5 точек (x, y): (0, 1), (1, 3), (2, 5), (3, 7), (4, 9). Построить интерполяционный многочлен Лагранжа;
- вычислить значение функции в точке x = 2.5 с помощью полученного многочлена;
- построить график исходной функции и многочлена, сравнить результаты.

3. Задание по численному дифференцированию:

- аппроксимировать первую производную функции $f(x) = \sin(x)$ в точке $x = \pi/4$ с помощью центральных разностей (шаг $h = 0.1$);
- сравнить результат с точным значением производной ($\cos(\pi/4)$);
- оценить погрешность численного метода.

4. Кейс по решению ОДУ:

- решить дифференциальное уравнение $y' = 2xy$ с начальным условием $y(0) = 1$ методом Эйлера на интервале [0, 1] с шагом $h = 0.1$;

- реализовать метод Рунге–Кутта 4-го порядка для того же уравнения и сравнить результаты;
- построить графики приближённых решений и точного решения (y = e^(x^2)) в одной системе координат.

5. Задание на минимизацию функции:

- найти минимум функции двух переменных $f(x, y) = x^2 + y^2 – 2x – 4y + 5$ методом градиентного спуска (начальная точка (0, 0), шаг $\alpha = 0.1$);
- выполнить 5 итераций вручную, записать последовательность точек и значений функции;
- реализовать алгоритм на языке программирования и построить контурную карту функции с траекторией спуска.

6. Комплексное задание:

- выбрать реальную задачу (например, расчёт траектории полёта снаряда, моделирование охлаждения тела, оптимизация портфеля инвестиций);
- сформулировать математическую модель (система уравнений, целевая функция);
- выбрать подходящие численные методы для решения (не менее 3);
- реализовать решение с помощью программного обеспечения (Python, MATLAB, Mathematica);
- проанализировать результаты, оценить точность и устойчивость методов.

Теоретические вопросы для устного опроса:

1. В чём разница между точными и численными методами решения математических задач? Приведите примеры.
2. Что такое погрешность численного метода? Перечислите виды погрешностей (грубая, систематическая, случайная).
3. Опишите метод половинного деления для нахождения корней уравнения. Какова его сходимость?
4. В чём отличие методов простой итерации и Ньютона для решения нелинейных уравнений? Сравните их скорость сходимости.
5. Что такое интерполяция? Перечислите основные виды интерполяции (линейная, полиномиальная, сплайновая).
6. Как работает метод Гаусса для решения систем линейных уравнений? Что такое LU-разложение?

7. Опишите квадратурные формулы Ньютона–Котеса. В чём отличие формул трапеций и Симпсона?

8. Что такое порядок точности численного метода интегрирования? Как он связан с шагом сетки h?

Тема 1.6. Безопасность программного обеспечения

Примеры типовых заданий:

1. Задание на анализ кода на наличие уязвимостей:

- получить фрагмент кода веб-приложения (PHP/Java/Python) с преднамеренно внесёнными уязвимостями (SQL-инъекции, XSS, небезопасная десериализация);
- провести ручной ревью кода с использованием чек-листа OWASP Code Review Guide;
- составить отчёт с описанием найденных уязвимостей, указанием строк кода и рекомендациями по исправлению;
- оценить критичность каждой уязвимости по шкале CVSS.

2. Практикум по защите от SQL-инъекций:

- создать веб-приложение с формой входа, уязвимой к SQL-инъекциям;
- продемонстрировать эксплуатацию уязвимости с помощью ручного ввода payload-ов (например, ' OR '1'='1);
- переписать код с использованием параметризованных запросов (Prepared Statements);
- проверить отсутствие уязвимости с помощью инструментов сканирования (SQLmap).

3. Задание по предотвращению XSS-атак:

- разработать веб-страницу с полем ввода комментариев, уязвимую к XSS;
- продемонстрировать отражённую и хранимую XSS-атаку;
- реализовать фильтрацию и экранирование пользовательского ввода на стороне сервера и клиента;
- настроить HTTP-заголовки (Content-Security-Policy, X-XSS-Protection) для дополнительной защиты.

4. Кейс по защите от CSRF-атак:

- создать форму изменения email пользователя без защиты от CSRF;
- смоделировать CSRF-атаку с помощью HTML-страницы с автоматически отправляемой формой;

- реализовать защиту с использованием CSRF-токенов (генерация, передача, проверка);
- протестировать защиту, убедившись, что атака больше не работает.

5. Задание на составление модели угроз:

- выбрать типовое веб-приложение (интернет-магазин, блог, CRM);
- идентифицировать активы (персональные данные, платёжная информация, контент);
- определить возможные угрозы (утечки, взлом, DDoS) и уязвимости;
- оценить вероятность и воздействие каждой угрозы;
- предложить меры по снижению рисков;
- оформить результаты в виде таблицы или диаграммы (например, с использованием методики STRIDE).

6. Практикум по настройке аутентификации:

- реализовать механизм аутентификации с использованием JWT-токенов;
- добавить функционал refresh-токенов для продления сессии;
- обеспечить безопасное хранение токенов на клиенте (HttpOnly, Secure cookies);
- протестировать механизм, проверив корректность выдачи, проверки и обновления токенов.

Теоретические вопросы для устного опроса:

1. Что такое OWASP Top 10? Перечислите 3–5 наиболее критичных уязвимостей 2023 года и кратко опишите каждую.
2. В чём разница между аутентификацией и авторизацией? Приведите примеры механизмов для каждого процесса.
3. Что такое JWT? Опишите структуру токена и его жизненный цикл. Какие уязвимости связаны с JWT?
4. Как работает RBAC? В чём отличие от ABAC и DAC?
5. Объясните разницу между симметричным и асимметричным шифрованием. Приведите примеры алгоритмов каждого типа.
6. Почему нельзя хранить пароли в открытом виде? Какие алгоритмы хэширования подходят для паролей и почему?
7. Что такое CSRF? Как отличить CSRF от XSS? Приведите пример CSRF-атаки.
8. Как работает HTTPS? Что такое SSL/TLS? Опишите процесс установки защищённого соединения.
9. Что такое CSP? Как он помогает защитить веб-приложение от XSS?

10. Какие инструменты используются для сканирования уязвимостей веб-приложений? Сравните OWASP ZAP и Burp Suite.

Типовые вопросы для защиты отчета по учебной практике УП.02.01

1. Какой архитектурный шаблон (MVC/MVVM/MVP) вы выбрали для своего проекта? Почему?

2. Как вы организовали модульную архитектуру приложения? Приведите примеры модулей и их взаимодействия.

3. Какие паттерны проектирования вы реализовали? Как они улучшили структуру кода?

4. Как вы работали с системой контроля версий? Опишите процесс ветвления и слияния кода.

5. Как реализована работа с файлами и потоками в вашем приложении? Какие механизмы изоляции данных вы использовали?

6. Приведите пример асинхронного вызова метода в вашем коде. Как вы обеспечили синхронизацию потоков?

7. Как вы обрабатывали ошибки и исключения в параллельных задачах?

8. Какие библиотеки вы использовали в проекте? Почему выбрали именно их?

9. Как вы тестировали корректность работы модулей? Приведите пример юнит теста.

10. Какие оптимизации вы применили для повышения производительности модулей?

11. Какие структуры данных вы использовали в проекте? Обоснуйте выбор.

12. Какой алгоритм сортировки/поиска вы реализовали? Почему выбрали его?

13. Как вы оценили сложность алгоритма, используемого в вашем модуле? Приведите расчёт О нотации.

14. Как реализована хеш таблица в вашем коде? Как вы решали проблему коллизий?

15. Приведите пример рекурсивного алгоритма из вашего проекта. Каковы его преимущества и недостатки?

16. Как вы находили кратчайший путь в графе? Опишите применение алгоритма Дейкстры.

17. Какой метод динамического программирования вы использовали? Приведите постановку задачи.

18. Как реализованы строковые алгоритмы в вашем приложении? Приведите пример поиска подстроки.

19. Как вы оптимизировали работу с приоритетными очередями?

20. Какие метрики вы использовали для оценки эффективности алгоритмов?

21. Как вы анализировали требования к модулю? Приведите пример функционального и нефункционального требования.

22. Какую спецификацию модуля вы разработали? Какие разделы она включает?

23. Как вы проектировали внутреннюю структуру модуля? Приведите фрагмент диаграммы классов UML.

24. Какие методы оптимизации вы применили к модулю? Как изменился его производительность?

25. Какие тесты вы написали для модуля? Каков процент покрытия кода тестами?

26. Как вы обеспечивали безопасность и масштабируемость модуля?

27. Какие инструменты статического анализа кода вы использовали? Какие проблемы они выявили?

28. Как вы документировали код? Приведите пример комментария к методу.

29. Какие дефекты вы обнаружили в ходе тестирования? Как их исправили?

30. Как вы проверяли совместимость модуля с другими компонентами системы?

31. Какую математическую модель вы построили? Каковы её входные и выходные параметры?

32. Какие допущения вы сделали при построении модели? Как они повлияли на точность результатов?

33. Как вы верифицировали модель? Приведите пример тестового сценария.

34. Какие программные средства вы использовали для реализации модели?

35. Как вы визуализировали результаты моделирования? Приведите график или диаграмму.

36. Какой численный метод вы применили в проекте? Почему выбрали его?

37. Как вы оценивали погрешность вычислений? Приведите расчёт.

38. Как вы выбирали шаг сетки (h) для метода интегрирования/дифференцирования?

39. Как вы решали систему линейных уравнений? Опишите алгоритм.

40. Какой метод оптимизации вы использовали? Как выбирали начальное приближение?

41. Как вы контролировали сходимость итерационного процесса?

42. Как вы сравнивали точность разных численных методов для одной задачи?

43. Какие программные библиотеки вы использовали для численных расчётов?

44. Как вы обрабатывали особые случаи (например, деление на ноль) в численном методе?

45. Как вы интерпретировали результаты численного решения?

46. Какие уязвимости вы выявили в коде? Как их устранили?

47. Как вы защитили приложение от SQL инъекций? Приведите фрагмент кода.

48. Какие меры против XSS вы реализовали? Опишите настройки CSP.

49. Как вы предотвратили CSRF атаки? Приведите пример CSRF токена.

50. Как организована аутентификация в вашем приложении? Почему вы выбрали JWT?

51. Как вы хранили пароли пользователей? Почему использовали bcrypt/Argon2?

52. Какие криптографические алгоритмы вы применили? Обоснуйте выбор AES/RSA.

53. Как вы настроили HTTPS? Опишите процесс создания сертификата.

54. Какие инструменты сканирования уязвимостей вы использовали? Какие результаты получили?

55. Как вы защитились от brute force атак? Приведите параметры ограничения попыток входа.

4 Методические указания для обучающихся по освоению дисциплины

Отчёт учебной практики УП.02.01. состоит из двух частей: основной и приложений.

Объём основной части отчёта составляет 13 – 16 страниц текста. Вторая часть отчёта о практике представляет собой приложения к отчёту (графики, схемы, диаграммы и таблицы, статистические данные, первичные документы, листинги т. п.).

Основная часть отчёта по учебной практике должна включать: титульный лист, содержание, основную часть и список использованной литературы.

Титульный лист — оформляется на специальном бланке, разработанном цикловой комиссией. Шрифт — Times New Roman, размер шрифта — 14 (пример оформления титульного листа — приложение А).

В содержании приводятся название разделов отчёта с обязательным указанием страниц.

По каждому разделу должны быть раскрыты теоретические вопросы и выполнены практические задания.

В конце отчёта приводится список использованной литературы.

Список использованной литературы должен включать источники и литературу, интернет-ресурсы, которые были использованы при выполнении учебной практики и оформлении отчёта.

К литературе относят: периодические и непериодические издания. Период издания литературы — 1–5 лет, статей — 1–2 года.

Для обеспечения наглядности в отчёте могут быть использованы графики, схемы и таблицы. Изложенный материал должен быть выстроен в логической последовательности.

5. Учебная литература и ресурсы информационно-телекоммуникационной сети «Интернет»

Основная литература

1. Агальцов, В. П. Математические методы в программировании: учебник / В. П. Агальцов. — 2-е изд., перераб, и доп. — Москва: ФОРУМ : ИНФРА-М, 2023. — 240 с. — (Среднее профессиональное образование). - ISBN 978-5-8199-0410-7. - Текст: электронный. - URL: https://znanium.ru/catalog/product/1896458 – Режим доступа: по подписке.

2. Емелина Е.И. Поддержка и тестирование программных модулей: учебник / Е.И. Емелина. – Москва: КНОРУС, 2024. – 272 с. – (Среднее профессиональное образование).

3. Колдаев, В. Д. Численные методы и программирование: учебное пособие / В.Д. Колдаев; под ред. Л.Г. Гагариной. — Москва: ФОРУМ : ИНФРА-М, 2025. — 336 с. — (Среднее профессиональное образование). - ISBN 978-5-8199-0779-5. - Текст: электронный. - URL: https://znanium.ru/catalog/product/2139606 – Режим доступа: по подписке.

4. Лапчик М.П. Численные методы: учебное издание / Лапчик М.П., Рагулина М.И., Хеннер Е. К. - Москва: Академия, 2024. - 256 с. (Специальности среднего профессионального образования). - URL: https://academia-moscow.ru - Режим доступа: Электронная библиотека «Academiamoscow». - Текст: электронный

5. Рогачева О.А. Разработка программных модулей: учебное издание / Рогачева О.А. - Москва: Академия, 2024. - 272 с. (Профессии среднего профессионального образования). - URL: https://academia-moscow.ru - Режим доступа: Электронная библиотека «Academia-moscow». - Текст: электронный

6. Слабнов, В. Д. Численные методы и программирование: учебное пособие для СПО / В. Д. Слабнов. — 2-е изд., стер. — Санкт-Петербург: Лань, 2022. — 460 с. — ISBN 978-5-8114-9250-3. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/189402 — Режим доступа: для авториз. пользователей.

7. Федорова Г.Н. Осуществление интеграции программных модулей: учебное издание / Федорова Г.Н. - Москва: Академия, 2023. - 288 с. (Специальности среднего профессионального образования). - URL: https://academia-moscow.ru - Режим доступа: Электронная библиотека «Academia-moscow». - Текст: электронный

8. Федорова Г.Н. Разработка модулей программного обеспечения для компьютерных систем: учебное издание / Федорова Г.Н. - Москва: Академия, 2024. - 384 с. (Специальности среднего профессионального образования). - URL: https://academia-moscow.ru - Режим доступа: Электронная библиотека «Academia-moscow». - Текст: электронный

Дополнительная литература

1. ГОСТ 19.001-77. Государственный стандарт Союза ССР. Единая система программной документации. Общие положения (введен в действие Постановлением Госстандарта СССР от 20.05.1977 N 1268). - URL: https://www.consultant.ru - Режим доступа: Правововой сервер КонсультантПлюс. - Текст: электронный

2. ГОСТ 19.101-77. Государственный стандарт Союза ССР. Единая система программной документации. Виды программ и программных документов (введен Постановлением Госстандарта СССР от 20.05.1977 N 1268). - URL: https://www.consultant.ru - Режим доступа: Правововой сервер КонсультантПлюс. - Текст: электронный

3. ГОСТ 19.102-77. Государственный стандарт Союза ССР. Единая система программной документации. Стадии разработки (введен в действие Постановлением Госстандарта СССР от 20.05.1977 N 1268). - URL: https://www.consultant.ru - Режим доступа: Правововой сервер КонсультантПлюс. - Текст: электронный

4. ГОСТ 19.201-78. Государственный стандарт Союза ССР. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению (введен в действие Постановлением Госстандарта СССР от 18.12.1978 N3351). - URL: https://www.consultant.ru - Режим доступа: Правововой сервер КонсультантПлюс. - Текст: электронный

5. ГОСТ 19.701-90. Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения (утв. Постановлением Госстандарта СССР от 26.12.1990 N 3294). - URL: https://www.consultant.ru - Режим доступа: Правововой сервер КонсультантПлюс. - Текст: электронный

6. ГОСТ Р ИСО/МЭК 25023-2021. Национальный стандарт Российской Федерации. Системная и программная инженерия. Требования и оценка качества систем и программной продукции (SQuaRE). Измерения качества системы и программной продукции (утв. и введен в действие Приказом Росстандарта от 19.11.2021 N 1524-ст). - URL: https://www.consultant.ru - Режим доступа: Правововой сервер КонсультантПлюс. - Текст: электронный

7. Акопов, А. С. Имитационное моделирование: учебник и практикум для вузов / А. С. Акопов. — 2-е изд., перераб, и доп. — Москва: Издательство Юрайт, 2024. — 426 с. — (Высшее образование). — ISBN 978-5-534-18379-5. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/534885

8. Згода Ю. Н. Проектирование программного обеспечения: учебно-методическое пособие / Ю. Н. Згода. – СПб.: Наукоемкие технологии, 2024. – 74 с. URL:https://publishing.intelgr.com/archive/Proektirovanie-programmnogo-obespecheniya.pdf. - Текст: электронный

9. Поколодина Е. В. Ревьюирование программных модулей: учебное издание / Поколодина Е. В., Долгова Н. А., Ананьев Д. В. - Москва: Академия, 2024. - 208 с. (Специальности среднего профессионального образования). - URL: https://academia-moscow.ru - Режим доступа: Электронная библиотека «Academia-moscow». - Текст: электронный

10. Библиотека профессионала №1 https://profspo.ru/

6 Материально-техническая база, информационные технологии, программное обеспечение и информационные справочные системы

Для реализации учебной практики необходимы:

− рабочие станции (персональные компьютеры) с характеристиками не ниже: процессор — Intel Core i5 (или аналогичный AMD), ОЗУ — 16 ГБ, SSD — не менее 256 ГБ;

− проекционное оборудование (проектор/интерактивная доска) для демонстрации материалов;

− сетевое подключение со скоростью не менее 100 Мбит/с.

Программное обеспечение

− Операционные системы: Windows 10/11, Linux (Ubuntu, CentOS, Astra, Alt)

− Офисные пакеты: Microsoft Office 365, LibreOffice

− Системы управления базами данных: PostgreSQL, MySQL/MariaDB, Microsoft SQL Server (Express-версия), MongoDB (для работы с NoSQL-данными).

Электронно-библиотечные системы (ЭБС)

1. ЭБС «BOOK.RU». — URL: https://book.ru/
2. ЭБС «Znanium». — URL: https://znanium.ru/
3. ЭБС «Айбукс». — URL: https://ibooks.ru/
4. ЭБС «Лань». — URL: https://e.lanbook.com/
5. ЭБС «Юрайт». — URL: https://urait.ru/
6. Электронные каталоги библиотеки СЗИУ РАНХиГС. — URL: https://sziu-lib.ranepa.ru/

ПРИЛОЖЕНИЕ А

Федеральное государственное бюджетное образовательное учреждение
высшего образования

РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА и ГОСУДАРСТВЕННОЙ СЛУЖБЫ при ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

СЕВЕРО-ЗАПАДНЫЙ ИНСТИТУТ УПРАВЛЕНИЯ

Факультет среднего профессионального образования

Специальность 09.02.11 Разработка и управление программным обеспечением

ОТЧЕТ
о прохождении учебой практики УП.02.01

по ПМ.02 Разработка и интеграция модулей программного обеспечения

Студента 3 курса группы _____
номер группы

Иванов Иван Иванович
(Ф.И.О. студента)

Наименование базы практики СЗИУ РАНХиГС (ФСПО)

Сроки прохождения практики: с «___»_____202__ г. по «___»_____202__ г.

Руководитель практики: _____ / П.П. Петров /
(подпись)

Обучающийся: _____ / И.И. Иванов /
(подпись)

Санкт-Петербург 202_ год