МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ГЕОДЕЗИИ И КАРТОГРАФИИ
(МИИГАиК)

Факультет геоинформатики и информационной безопасности
Кафедра информационно-измерительных систем

**ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ**

Сертификат: 5BFEE1D286E9C95A0C0D2C611E434BD5
Владелец: Степанченко Алексей Леонидович
Действителен: с 23.09.2026 по 17.12.2027

«УТВЕРЖДАЮ»
Проректор
А. Л. Степанченко
15 мая 2026 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ПО ПРАКТИКЕ «Б2.О.03 УЧЕБНАЯ (ОЗНАКОМИТЕЛЬНАЯ) ПРАКТИКА ПО ПРОГРАММИРОВАНИЮ»

Уровень высшего образования: бакалавриат

Направление подготовки: 09.03.03 Прикладная информатика

Направленность (профиль) подготовки: Разработка программного обеспечения для обработки и анализа пространственных данных

Квалификация (степень) выпускника: бакалавр

Форма обучения: очная

Год набора (приема на обучение): 2026

Срок получения образования: 4 года

Объем: в зачетных единицах: 5 з.е.
в академических часах: 180 ак.ч.

Москва, 2026

Разработчики:
Преподаватель кафедры Информационно-измерительных систем
Рамазанов Э. Р.

Фонд оценочных средств по практике составлен в соответствии с требованиями ФГОС ВО по направлению подготовки 09.03.03 Прикладная информатика, утвержденного приказом Минобрнауки от 19.09.2017 № 922

Согласование и утверждение

№	Подразделение	Ответственное лицо	ФИО	Виза	Дата, протокол (при наличии)
1	факультет Геоинформатики и информационной безопасности	Председатель ученого совета факультета	Матерухин А. В.	Рассмотрено	05.11.2025, № 12-25

1. Показатели и критерии оценивания компетенций

Планируемые результаты освоения программы	Уровень овладения	Критерии оценивания (перечень сформированных ИДК/ЗУВ по уровням освоения)
ОПК-7 Способен разрабатывать алгоритмы и программы, пригодные для практического применения	Повышенный	ИДК ОПК-7.1 Анализирует поставленную задачу и требования к программному продукту, определяет необходимые алгоритмические и программные решения, а также оценивает их применимость для практического использования Знать: ОПК-7.1/Зн1 Основные парадигмы программирования и алгоритмические структуры, методы анализа и декомпозиции задач, критерии выбора алгоритмических решений, основы проектирования программных систем на языке Python, методы оценки вычислительной сложности алгоритмов и их практической применимости. Уметь: ОПК-7.1/Ум5 Проводить анализ исходных данных и условий задачи, формулировать требования к алгоритму и программной реализации, определять ограничения и оценивать применимость выбранного решения. ОПК-7.1/Ум6 Построить блок-схему заданного алгоритма, исходя из его описания Владеть: ОПК-7.1/Нв1 Навыками системного анализа предметной области и формализации требований, методами алгоритмического проектирования и выбора структур данных в Python, инструментами профилирования и оценки производительности Python-приложений, навыками технико-экономического обоснования выбранных программных решений. ОПК-7.1/Нв5 Навыками анализа поставленной задачи, включающим формализацию исходных данных в виде алгоритмической модели задачи, выбор рациональных методов решения и обоснование препдпочительной алгоритмической стратегии.

			ИДК ОПК-7.2 Разрабатывает эффективные алгоритмы для решения задач, а также выбирает оптимальные структуры данных для их реализации Знать: ОПК-7.2/Зн1 Основные алгоритмические парадигмы (жадные алгоритмы, динамическое программирование, метод "разделяй и властвуй", рекурсивные алгоритмы), базовые и продвинутые структуры данных Python (списки, кортежи, множества, словари, очереди, стеки, деревья), методы анализа временной и пространственной сложности алгоритмов с использованием нотации О-большое, критерии выбора оптимальных структур данных для различных типов задач, классические алгоритмы сортировки, поиска, обхода графов и области их применения. ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма. Уметь: ОПК-7.2/Ум1 Разрабатывать эффективные алгоритмические решения с учетом ограничений по времени и памяти, выбирать подходящие структуры данных на основе анализа требований задачи, оценивать и сравнивать эффективность различных алгоритмических подходов, реализовывать классические алгоритмы и адаптировать их под конкретные задачи, применять встроенные структуры данных Python и библиотеки collections для оптимизации решений. ОПК-7.2/Ум2 Применять современные информационные технологии для анализа данных Владеть: ОПК-7.2/Нв1 Навыками разработки и реализации эффективных алгоритмов на языке Python, техниками оптимизации алгоритмов и выбора оптимальных структур данных, методами профилирования и анализа производительности алгоритмов, инструментами визуализации алгоритмов и структур данных. ОПК-7.2/Нв2 Навыками анализа данных для решения задач профессиональной деятельности

			ИДК ОПК-7.3 Пишет программный код на одном или нескольких языках программирования, следуя принципам модульности, читаемости, документирования и соответствия стандартам кодирования Знать: ОПК-7.3/Зн1 Синтаксис и семантику языка программирования Python, принципы структурного и модульного программирования, стандарты оформления кода PEP 8 и PEP 257, системы документирования кода (docstrings, type hints), принципы написания чистого и поддерживаемого кода, паттерны проектирования, применимые в Python. Уметь: ОПК-7.3/Ум1 Писать читаемый и эффективный код на языке Python с соблюдением принципов DRY и KISS, структурировать программу с использованием функций, классов и модулей, документировать код с использованием строк документации и комментариев, аннотацию типов для повышения надежности кода, использовать инструменты автоматической проверки качества кода (pylint, flake8, black). ОПК-7.3/Ум5 Реализовывать алгоритмы в виде программных модулей, документировать код и соблюдать стандарты оформления Владеть: ОПК-7.3/Нв1 Навыками разработки модульных Python-приложений с четким разделением ответственности, инструментами рефакторинга и оптимизации кода, методами написания самодокументируемого кода, навыками работы с системами контроля версий для командной разработки. ОПК-7.3/Нв4 Навыками пользования современными информационными технологиями с учетом их особенностей и возможной области применения ОПК-7.3/Нв5 Навыками программирования и реализации алгоритмов на языках C++ и Python с соблюдением принципов читаемости и модульностия

			ИДК ОПК-7.4 Проводит отладку и тестирование программного кода, выявляет и устраняет ошибки, а также оптимизирует производительность программы для обеспечения её стабильной и эффективной работы в реальных условиях Знать: ОПК-7.4/Зн1 Виды ошибок в программах (синтаксические, логические, времени выполнения), методологии и стратегии отладки программного кода, основы модульного и интеграционного тестирования, инструменты отладки и профилирования в Python, принципы обработки исключений и написания отказоустойчивого кода. Уметь: ОПК-7.4/Ум1 Использовать встроенные и внешние отладчики для поиска и устранения ошибок, разрабатывать unit-тесты с использованием модулей unittest и pytest, применять методы логирования для диагностики проблем в работе программы, проводить профилирование кода и выявлять узкие места производительности, обрабатывать исключения и создавать информативные сообщения об ошибках. Владеть: ОПК-7.4/Нв1 Навыками систематической отладки и рефакторинга программного кода, техниками написания тестируемого кода и создания тестовых сценариев, инструментами автоматизированного тестирования и непрерывной интеграции, методами оптимизации производительности Python-приложений, навыками использования систем контроля версий для отслеживания и исправления ошибок.

	Базовый	ИДК ОПК-7.1 Анализирует поставленную задачу и требования к программному продукту, определяет необходимые алгоритмические и программные решения, а также оценивает их применимость для практического использования Знать: ОПК-7.1/Зн1 Основные парадигмы программирования и алгоритмические структуры, методы анализа и декомпозиции задач, критерии выбора алгоритмических решений, основы проектирования программных систем на языке Python, методы оценки вычислительной сложности алгоритмов и их практической применимости. Уметь: ОПК-7.1/Ум6 Построить блок-схему заданного алгоритма, исходя из его описания ОПК-7.1/Ум5 Проводить анализ исходных данных и условий задачи, формулировать требования к алгоритму и программной реализации, определять ограничения и оценивать применимость выбранного решения.
	Базовый	ИДК ОПК-7.2 Разрабатывает эффективные алгоритмы для решения задач, а также выбирает оптимальные структуры данных для их реализации Знать: ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма. ОПК-7.2/Зн1 Основные алгоритмические парадигмы (жадные алгоритмы, динамическое программирование, метод "разделяй и властвуй", рекурсивные алгоритмы), базовые и продвинутые структуры данных Python (списки, кортежи, множества, словари, очереди, стеки, деревья), методы анализа временной и пространственной сложности алгоритмов с использованием нотации О-большое, критерии выбора оптимальных структур данных для различных типов задач, классические алгоритмы сортировки, поиска, обхода графов и области их применения. Уметь: ОПК-7.2/Ум1 Разрабатывать эффективные алгоритмические решения с учетом ограничений по времени и памяти, выбирать подходящие структуры данных на основе анализа требований задачи, оценивать и сравнивать эффективность различных алгоритмических подходов, реализовывать классические алгоритмы и адаптировать их под конкретные задачи, применять встроенные структуры данных Python и библиотеки collections для оптимизации решений.

		ОПК-7.2/Ум2 Применять современные информационные технологии для анализа данных ИДК ОПК-7.3 Пишет программный код на одном или нескольких языках программирования, следуя принципам модульности, читаемости, документирования и соответствия стандартам кодирования Знать: ОПК-7.3/Зн1 Синтаксис и семантику языка программирования Python, принципы структурного и модульного программирования, стандарты оформления кода PEP 8 и PEP 257, системы документирования кода (docstrings, type hints), принципы написания чистого и поддерживаемого кода, паттерны проектирования, применимые в Python. Уметь: ОПК-7.3/Ум1 Писать читаемый и эффективный код на языке Python с соблюдением принципов DRY и KISS, структурировать программу с использованием функций, классов и модулей, документировать код с использованием строк документации и комментариев, аннотацию типов для повышения надежности кода, использовать инструменты автоматической проверки качества кода (pylint, flake8, black). ОПК-7.3/Ум5 Реализовывать алгоритмы в виде программных модулей, документировать код и соблюдать стандарты оформления ИДК ОПК-7.4 Проводит отладку и тестирование программного кода, выявляет и устраняет ошибки, а также оптимизирует производительность программы для обеспечения её стабильной и эффективной работы в реальных условиях Знать: ОПК-7.4/Зн1 Виды ошибок в программах (синтаксические, логические, времени выполнения), методологии и стратегии отладки программного кода, основы модульного и интеграционного тестирования, инструменты отладки и профилирования в Python, принципы обработки исключений и написания отказоустойчивого кода. Уметь: ОПК-7.4/Ум1 Использовать встроенные и внешние отладчики для поиска и устранения ошибок, разрабатывать unit-тесты с использованием модулей unittest и pytest, применять методы логирования для диагностики проблем в работе программы, проводить профилирование кода и выявлять узкие места производительности, обрабатывать исключения и создавать информативные сообщения об ошибках.

	Пороговый	ИДК ОПК-7.1 Анализирует поставленную задачу и требования к программному продукту, определяет необходимые алгоритмические и программные решения, а также оценивает их применимость для практического использования Знать: ОПК-7.1/Зн1 Основные парадигмы программирования и алгоритмические структуры, методы анализа и декомпозиции задач, критерии выбора алгоритмических решений, основы проектирования программных систем на языке Python, методы оценки вычислительной сложности алгоритмов и их практической применимости.
	Пороговый	ИДК ОПК-7.2 Разрабатывает эффективные алгоритмы для решения задач, а также выбирает оптимальные структуры данных для их реализации Знать: ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма. ОПК-7.2/Зн1 Основные алгоритмические парадигмы (жадные алгоритмы, динамическое программирование, метод "разделяй и властвуй", рекурсивные алгоритмы), базовые и продвинутые структуры данных Python (списки, кортежи, множества, словари, очереди, стеки, деревья), методы анализа временной и пространственной сложности алгоритмов с использованием нотации О-большое, критерии выбора оптимальных структур данных для различных типов задач, классические алгоритмы сортировки, поиска, обхода графов и области их применения. ИДК ОПК-7.3 Пишет программный код на одном или нескольких языках программирования, следуя принципам модульности, читаемости, документирования и соответствия стандартам кодирования Знать: ОПК-7.3/Зн1 Синтаксис и семантику языка программирования Python, принципы структурного и модульного программирования, стандарты оформления кода PEP 8 и PEP 257, системы документирования кода (docstrings, type hints), принципы написания чистого и поддерживаемого кода, паттерны проектирования, применимые в Python.

		ИДК ОПК-7.4 Проводит отладку и тестирование программного кода, выявляет и устраняет ошибки, а также оптимизирует производительность программы для обеспечения её стабильной и эффективной работы в реальных условиях Знать: ОПК-7.4/Зн1 Виды ошибок в программах (синтаксические, логические, времени выполнения), методологии и стратегии отладки программного кода, основы модульного и интеграционного тестирования, инструменты отладки и профилирования в Python, принципы обработки исключений и написания отказоустойчивого кода.

2. Шкала оценивания

Оценка	Критерии оценивания (перечень сформированных ИДК/ЗУВ)
Отлично	ОПК-7.1 Анализирует поставленную задачу и требования к программному продукту, определяет необходимые алгоритмические и программные решения, а также оценивает их применимость для практического использования ОПК-7.1/Зн1 Основные парадигмы программирования и алгоритмические структуры, методы анализа и декомпозиции задач, критерии выбора алгоритмических решений, основы проектирования программных систем на языке Python, методы оценки вычислительной сложности алгоритмов и их практической применимости. ОПК-7.1/Ум5 Проводить анализ исходных данных и условий задачи, формулировать требования к алгоритму и программной реализации, определять ограничения и оценивать применимость выбранного решения. ОПК-7.1/Ум6 Построить блок-схему заданного алгоритма, исходя из его описания ОПК-7.1/Нв1 Навыками системного анализа предметной области и формализации требований, методами алгоритмического проектирования и выбора структур данных в Python, инструментами профилирования и оценки производительности Python-приложений, навыками технико-экономического обоснования выбранных программных решений. ОПК-7.1/Нв5 Навыками анализа поставленной задачи, включающим формализацию исходных данных в виде алгоритмической модели задачи, выбор рациональных методов решения и обоснование препдпочительной алгоритмической стратегии.

	ОПК-7.2 Разрабатывает эффективные алгоритмы для решения задач, а также выбирает оптимальные структуры данных для их реализации ОПК-7.2/Зн1 Основные алгоритмические парадигмы (жадные алгоритмы, динамическое программирование, метод "разделяй и властвуй", рекурсивные алгоритмы), базовые и продвинутые структуры данных Python (списки, кортежи, множества, словари, очереди, стеки, деревья), методы анализа временной и пространственной сложности алгоритмов с использованием нотации О-большое, критерии выбора оптимальных структур данных для различных типов задач, классические алгоритмы сортировки, поиска, обхода графов и области их применения. ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма. ОПК-7.2/Ум1 Разрабатывать эффективные алгоритмические решения с учетом ограничений по времени и памяти, выбирать подходящие структуры данных на основе анализа требований задачи, оценивать и сравнивать эффективность различных алгоритмических подходов, реализовывать классические алгоритмы и адаптировать их под конкретные задачи, применять встроенные структуры данных Python и библиотеки collections для оптимизации решений. ОПК-7.2/Ум2 Применять современные информационные технологии для анализа данных ОПК-7.2/Нв1 Навыками разработки и реализации эффективных алгоритмов на языке Python, техниками оптимизации алгоритмов и выбора оптимальных структур данных, методами профилирования и анализа производительности алгоритмов, инструментами визуализации алгоритмов и структур данных. ОПК-7.2/Нв2 Навыками анализа данных для решения задач профессиональной деятельности
	ОПК-7.3 Пишет программный код на одном или нескольких языках программирования, следуя принципам модульности, читаемости, документирования и соответствия стандартам кодирования ОПК-7.3/Зн1 Синтаксис и семантику языка программирования Python, принципы структурного и модульного программирования, стандарты оформления кода PEP 8 и PEP 257, системы документирования кода (docstrings, type hints), принципы написания чистого и поддерживаемого кода, паттерны проектирования, применимые в Python. ОПК-7.3/Ум1 Писать читаемый и эффективный код на языке Python с соблюдением принципов DRY и KISS, структурировать программу с использованием функций, классов и модулей, документировать код с использованием строк документации и комментариев, аннотацию типов для повышения надежности кода, использовать инструменты автоматической проверки качества кода (pylint, flake8, black). ОПК-7.3/Ум5 Реализовывать алгоритмы в виде программных модулей, документировать код и соблюдать стандарты оформления ОПК-7.3/Нв1 Навыками разработки модульных Python-приложений с четким разделением ответственности, инструментами рефакторинга и оптимизации кода, методами написания самодокументируемого кода, навыками работы с системами контроля версий для командной разработки. ОПК-7.3/Нв4 Навыками пользования современными информационными технологиями с учетом их особенностей и возможной области применения ОПК-7.3/Нв5 Навыками программирования и реализации алгоритмов на языках C++ и Python с соблюдением принципов читаемости и модульностия

	ОПК-7.4 Проводит отладку и тестирование программного кода, выявляет и устраняет ошибки, а также оптимизирует производительность программы для обеспечения её стабильной и эффективной работы в реальных условиях ОПК-7.4/Зн1 Виды ошибок в программах (синтаксические, логические, времени выполнения), методологии и стратегии отладки программного кода, основы модульного и интеграционного тестирования, инструменты отладки и профилирования в Python, принципы обработки исключений и написания отказоустойчивого кода. ОПК-7.4/Ум1 Использовать встроенные и внешние отладчики для поиска и устранения ошибок, разрабатывать unit-тесты с использованием модулей unittest и pytest, применять методы логирования для диагностики проблем в работе программы, проводить профилирование кода и выявлять узкие места производительности, обрабатывать исключения и создавать информативные сообщения об ошибках. ОПК-7.4/Нв1 Навыками систематической отладки и рефакторинга программного кода, техниками написания тестируемого кода и создания тестовых сценариев, инструментами автоматизированного тестирования и непрерывной интеграции, методами оптимизации производительности Python-приложений, навыками использования систем контроля версий для отслеживания и исправления ошибок.
Хорошо	ОПК-7.1 Анализирует поставленную задачу и требования к программному продукту, определяет необходимые алгоритмические и программные решения, а также оценивает их применимость для практического использования ОПК-7.1/Зн1 Основные парадигмы программирования и алгоритмические структуры, методы анализа и декомпозиции задач, критерии выбора алгоритмических решений, основы проектирования программных систем на языке Python, методы оценки вычислительной сложности алгоритмов и их практической применимости. ОПК-7.1/Ум5 Проводить анализ исходных данных и условий задачи, формулировать требования к алгоритму и программной реализации, определять ограничения и оценивать применимость выбранного решения. ОПК-7.1/Ум6 Построить блок-схему заданного алгоритма, исходя из его описания

ОПК-7.2 Разрабатывает эффективные алгоритмы для решения задач, а также выбирает оптимальные структуры данных для их реализации
ОПК-7.2/Зн1 Основные алгоритмические парадигмы (жадные алгоритмы, динамическое программирование, метод "разделяй и властвуй", рекурсивные алгоритмы), базовые и продвинутые структуры данных Python (списки, кортежи, множества, словари, очереди, стеки, деревья), методы анализа временной и пространственной сложности алгоритмов с использованием нотации О-большое, критерии выбора оптимальных структур данных для различных типов задач, классические алгоритмы сортировки, поиска, обхода графов и области их применения.
ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма.
ОПК-7.2/Ум1 Разрабатывать эффективные алгоритмические решения с учетом ограничений по времени и памяти, выбирать подходящие структуры данных на основе анализа требований задачи, оценивать и сравнивать эффективность различных алгоритмических подходов, реализовывать классические алгоритмы и адаптировать их под конкретные задачи, применять встроенные структуры данных Python и библиотеки collections для оптимизации решений.
ОПК-7.2/Ум2 Применять современные информационные технологии для анализа данных

ОПК-7.3 Пишет программный код на одном или нескольких языках программирования, следуя принципам модульности, читаемости, документирования и соответствия стандартам кодирования
ОПК-7.3/Зн1 Синтаксис и семантику языка программирования Python, принципы структурного и модульного программирования, стандарты оформления кода PEP 8 и PEP 257, системы документирования кода (docstrings, type hints), принципы написания чистого и поддерживаемого кода, паттерны проектирования, применимые в Python.
ОПК-7.3/Ум1 Писать читаемый и эффективный код на языке Python с соблюдением принципов DRY и KISS, структурировать программу с использованием функций, классов и модулей, документировать код с использованием строк документации и комментариев, аннотацию типов для повышения надежности кода, использовать инструменты автоматической проверки качества кода (pylint, flake8, black).
ОПК-7.3/Ум5 Реализовывать алгоритмы в виде программных модулей, документировать код и соблюдать стандарты оформления

ОПК-7.4 Проводит отладку и тестирование программного кода, выявляет и устраняет ошибки, а также оптимизирует производительность программы для обеспечения её стабильной и эффективной работы в реальных условиях
ОПК-7.4/Зн1 Виды ошибок в программах (синтаксические, логические, времени выполнения), методологии и стратегии отладки программного кода, основы модульного и интеграционного тестирования, инструменты отладки и профилирования в Python, принципы обработки исключений и написания отказоустойчивого кода.
ОПК-7.4/Ум1 Использовать встроенные и внешние отладчики для поиска и устранения ошибок, разрабатывать unit-тесты с использованием модулей unittest и pytest, применять методы логирования для диагностики проблем в работе программы, проводить профилирование кода и выявлять узкие места производительности, обрабатывать исключения и создавать информативные сообщения об ошибках.

Удовлетворительно	ОПК-7.1 Анализирует поставленную задачу и требования к программному продукту, определяет необходимые алгоритмические и программные решения, а также оценивает их применимость для практического использования ОПК-7.1/Зн1 Основные парадигмы программирования и алгоритмические структуры, методы анализа и декомпозиции задач, критерии выбора алгоритмических решений, основы проектирования программных систем на языке Python, методы оценки вычислительной сложности алгоритмов и их практической применимости. ОПК-7.2 Разрабатывает эффективные алгоритмы для решения задач, а также выбирает оптимальные структуры данных для их реализации ОПК-7.2/Зн1 Основные алгоритмические парадигмы (жадные алгоритмы, динамическое программирование, метод "разделяй и властвуй", рекурсивные алгоритмы), базовые и продвинутые структуры данных Python (списки, кортежи, множества, словари, очереди, стеки, деревья), методы анализа временной и пространственной сложности алгоритмов с использованием нотации О-большое, критерии выбора оптимальных структур данных для различных типов задач, классические алгоритмы сортировки, поиска, обхода графов и области их применения. ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма. ОПК-7.3 Пишет программный код на одном или нескольких языках программирования, следуя принципам модульности, читаемости, документирования и соответствия стандартам кодирования ОПК-7.3/Зн1 Синтаксис и семантику языка программирования Python, принципы структурного и модульного программирования, стандарты оформления кода PEP 8 и PEP 257, системы документирования кода (docstrings, type hints), принципы написания чистого и поддерживаемого кода, паттерны проектирования, применимые в Python. ОПК-7.4 Проводит отладку и тестирование программного кода, выявляет и устраняет ошибки, а также оптимизирует производительность программы для обеспечения её стабильной и эффективной работы в реальных условиях ОПК-7.4/Зн1 Виды ошибок в программах (синтаксические, логические, времени выполнения), методологии и стратегии отладки программного кода, основы модульного и интеграционного тестирования, инструменты отладки и профилирования в Python, принципы обработки исключений и написания отказоустойчивого кода.
Неудовле-творительно	Результаты обучения не сформированы на достаточном уровне

3. Контрольные мероприятия по практике

Вид контроля	Форма контроля/Оценочное средство
Текущий контроль	Тестирование Практическая работа
Промежуточная аттестация	Зачет с оценкой

№	Наименование раздела	Контролируем	Вид контроля/ используемые оценочные материалы

п/п	Наименование раздела	ые ИДК	Текущий	Промежут. аттестация
1	Подготовительный этап: Формирование проектных заданий - 10 час. Тема 1.1 Определение треков, стека и критериев оценки - 10 час.	ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4	Тестирование Практическая работа	Зачет с оценкой
2	Планирование и проектирование - 36 час. Тема 2.1 Консультации по планированию разработки и проектированию схемы взаимодействия компонентов системы - 20 час. Тема 2.2 Проектирование системы и распределение ролей - 16 час.	ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4	Тестирование Практическая работа	Зачет с оценкой
3	Разработка и реализация проекта - 106 час. Тема 3.1 Консультации по процессу разработки - 28 час. Тема 3.2 Практическая реализация системы - 78 час.	ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4	Тестирование Практическая работа	Зачет с оценкой
4	Завершающий этап: Подготовка к демонстрации и подведение итогов - 28 час. Тема 4.1 Формирование отчетных материалов - 28 час.	ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4	Тестирование Практическая работа	Зачет с оценкой

4. Порядок проведения промежуточной аттестации

Порядок проведения промежуточной аттестации регламентируется локальными нормативными актами Университета:

- Положением о формах, периодичности, порядке текущего контроля успеваемости и промежуточной аттестации обучающихся по основным образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры в федеральном государственном бюджетном образовательном учреждении высшего образования «Московский государственный университет геодезии и картографии», п.6 «Организация и проведение промежуточной аттестации».
- Положением о порядке осуществления образовательного процесса, проведения текущего контроля успеваемости и промежуточной аттестации обучающихся с применением электронного обучения и дистанционных образовательных технологий в федеральном государственном бюджетном образовательном учреждении высшего образования «Московский государственный университет геодезии и картографии», п.8 «Проведение промежуточной аттестации с применением ЭО и ДОТ».

Промежуточная аттестация: Зачет с оценкой, 2 семестр.

Промежуточная аттестация по дисциплине проводится в форме «дифференцированный зачет». Конкретная форма проведения — компьютерное тестирование по теоретическому материалу или устное собеседование по вопросам билетов — утверждается кафедрой и доводится до сведения обучающихся не позднее чем за месяц до начала промежуточной аттестации. При компьютерном тестировании итоговый результат определяется как отношение количества набранных баллов к максимально возможному количеству баллов. При устном собеседовании билет включает вопросы из перечня вопросов; преподаватель вправе задать дополнительные и

уточняющие вопросы.

Критерии оценивания при проведении промежуточной аттестации в форме компьютерного тестирования:
Зачет оценивается по пятибалльной шкале:
- зачтено с оценкой «отлично» выставляется, если обучающийся набрал 86–100 баллов и продемонстрировал системное и полное освоение содержания дисциплины; ответы правильные, обоснованные и терминологически точные;
- зачтено с оценкой «хорошо» выставляется обучающемуся, набравшему 71–85 баллов, в случае, если основное содержание освоено, но допущены отдельные неточности, не влияющие на общее понимание материала;
- зачтено с оценкой «удовлетворительно» выставляется обучающемуся, набравшему 56–70 баллов, если освоены базовые понятия и положения, но имеются существенные пробелы и ошибки;
- не зачтено («не удовлетворительно») – если обучающийся набрал 0–55 баллов, не освоен необходимый минимум содержания дисциплины; результаты обучения не достигнуты.

Критерии оценивания при проведении промежуточной аттестации в форме устного собеседования по билетам:
Зачет оценивается по пятибалльной шкале:
- зачтено с оценкой «отлично» выставляется, если обучающийся набрал 86–100 баллов и дал полные, правильные и логически последовательные ответы; терминология применяется корректно, приведены необходимые обоснования и примеры, дополнительные вопросы раскрыты;
- зачтено с оценкой «хорошо» выставляется обучающемуся, набравшему 71–85 баллов, если вопросы раскрыты правильно и достаточно полно, а отдельные неточности исправлены после уточняющих вопросов;
- зачтено с оценкой «удовлетворительно» выставляется обучающемуся, набравшему 56–70 баллов, если основные положения раскрыты фрагментарно, имеются существенные неточности, однако базовое понимание продемонстрировано;
- не зачтено («не удовлетворительно») – если обучающийся набрал 0–55 баллов, вопросы не раскрыты либо допущены принципиальные ошибки, отсутствует владение базовой терминологией.

5. Оценочные материалы текущего контроля

Раздел 1. Подготовительный этап: Формирование проектных заданий
Контролируемые ИДК: ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4

Тема 1.1. Определение треков, стека и критериев оценки

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Упорядочьте подготовку задания: проверку полноты выполняют после записи входов и выходов.

А) Записать входные и выходные данные
Б) Уточнить назначение программы
В) Проверить полноту описания

2. Расположите узлы линейной блок-схемы вычисления суммы двух чисел после узла начала.

А) Вычислить s = a + b
Б) Ввести a и b
В) Вывести s

3. В задании написано «обработать файл», но его формат неизвестен. Какое уточнение о данных следует запросить первым?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Постановка профессиональной задачи

Исходные данные: Описание небольшой задачи обработки данных или автоматизации по выбору руководителя; ограничения учебной среды.

Определить назначение программы, участников, входы и выходы, допустимые значения, функции и ограничения. Согласовать три проверяемых примера, включая особый случай, и записать, что не входит в задание.

2. Алгоритм и блок-схема по текстовому описанию

Исходные данные: Текст алгоритма с вводом, проверкой условия и повторением, связанный с выбранной задачей.

Выделить шаги и условия, построить блок-схему, явно показать обе ветви и условие завершения цикла. Выполнить ручную трассировку на обычном и граничном входе; сопоставить схему с исходным текстом.

3. Сравнение применимости вариантов

Исходные данные: Два простых варианта решения одной задачи, согласованные ограничения времени и памяти, учебные оценки времени разработки.

Сравнить варианты по корректности, ресурсам, трудозатратам и доступности средств. Выполнить небольшое пробное измерение, подсчитать суммарные учебные трудозатраты и обосновать выбор. Указать условия, при которых выбор придется пересмотреть.

Раздел 2. Планирование и проектирование

Контролируемые ИДК: ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4

Тема 2.1. Консультации по планированию разработки и проектированию схемы взаимодействия компонентов системы

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. В указанном варианте линейного поиска сначала рассматривают первый элемент. Упорядочьте действия для первого шага.

2. Упорядочьте действия одного шага бинарного поиска после выбора непустого отсортированного диапазона.

А) Сравнить ее элемент с искомым значением

Б) Найти середину диапазона

В) При неравенстве оставить подходящую половину

3. Минимальная куча содержит числа 8, 2 и 5. Какое число первым вернет heappop?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Выбор структур данных

Исходные данные: Небольшие наборы записей и три операции: поиск по ключу, обслуживание по очереди и выбор наименьшего приоритета.

Сопоставить список, кортеж, множество и словарь по изменяемости, уникальности и доступу. Реализовать очередь на deque, стек, подсчет категорий Counter и простую минимальную кучу heapq. Показать порядок извлечения на трех–пяти элементах. На схеме объяснить отличие индексируемого массива от связного списка. Нарисовать небольшой связный список, дерево и граф, представить граф списками соседей и реализовать его обход в ширину с отметками посещения. Обосновать выбор структуры для каждой операции.

2. Поиск, сортировка и оценка ресурсов

Исходные данные: Два небольших списка и перечень искомых значений; затем наборы нескольких согласованных размеров.

Реализовать линейный поиск и простой алгоритм сортировки, применить бинарный поиск к отсортированному входу. Адаптировать сортировку к согласованному направлению порядка и проверить изменение. Сосчитать сравнения в коротком примере, объяснить временную и пространственную оценку Big O. Измерить время и дополнительную память; отдельно учесть подготовку отсортированного входа.

Тема 2.2. Проектирование системы и распределение ролей

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Упорядочьте упрощенную обработку запроса: ответ формируется только после чтения данных.

А) Клиент отправляет запрос
Б) Сервер возвращает ответ клиенту
В) Сервер читает необходимые данные

2. Упорядочьте создание простой таблицы по согласованной задаче.

А) Подготовить описание таблицы и ключа
Б) Создать таблицу по описанию
В) Определить необходимые поля записи

3. Два участника одновременно изменяют одну функцию. Какое простое организационное действие уменьшит риск несовместимых правок?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Проектирование компонентов и данных

Исходные данные: Согласованная учебная задача с клиентской частью, серверной обработкой и хранением данных; конкретный объект выбирает руководитель.
Разделить обязанности компонентов, определить интерфейсы и формат обмена. Изобразить взаимодействие и схему простой базы данных с ключами; выделить модули реализации. Объяснить, как требование проходит от входа клиента до результата и хранения.

2. Командный план и учебная модель угроз

Исходные данные: Схема выбранного приложения, состав учебной команды или согласованных ролей, правила доступа к учебным данным.
Разбить работу на задачи, согласовать роли, зависимости и контрольные результаты. Для двух потоков данных определить защищаемые сведения, возможное нежелательное действие и место контроля. Предложить простые проверки доступа или входа и согласовать их с разработчиками компонентов.

Раздел 3. Разработка и реализация проекта

Контролируемые ИДК: ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4

Тема 3.1. Консультации по процессу разработки

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Упорядочьте исправление воспроизводимой ошибки.

А) Повторить проверку на тех же данных
Б) Найти причину и изменить код
В) Повторить ошибку на известных данных

2. Упорядочьте исследование значения переменной с заранее поставленной точкой останова.

А) Запустить программу под отладчиком
Б) Посмотреть значение переменной
В) Дождаться остановки в указанной точке

3. Какое исключение Python возникает при int('abc')?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Отладка и воспроизведение ошибок

Исходные данные: Небольшая программа Python с синтаксической, логической ошибкой и ошибкой времени выполнения; безопасные учебные данные.

Воспроизвести ошибки, определить их виды и причины. Один случай исследовать встроенным pdb, другой — доступным внешним отладчиком: поставить точку останова, просмотреть переменные, выполнить шаги. Исправить код, записать наблюдения и повторить проверки.

2. Обработка отказов и журналирование

Исходные данные: Программа чтения учебного файла: возможны отсутствие файла и неверное значение поля.

Обработать ожидаемые исключения в подходящем месте, добавить понятное сообщение и диагностическую запись без лишних персональных сведений. Проверить оба отказа и успешное чтение; показать, что неожиданная ошибка не превращается в ложный успешный результат.

Тема 3.2. Практическая реализация системы

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Упорядочьте запись данных через соединение sqlite3 при явном управлении транзакцией.

А) Выполнить INSERT с параметрами

Б) Подтвердить изменения командой commit

В) Открыть соединение

2. В учебной команде изменения фиксируют после успешной локальной проверки. Упорядочьте действия.

А) Подготовить проверенные изменения к фиксации

Б) Запустить тесты измененного кода

В) Создать коммит с описанием

3. В словаре counts записано {'a': 2}. Какой результат даст counts.get('b', 0)?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Автоматическая проверка качества и рефакторинг

Исходные данные: Небольшой модуль Python с лишним импортом, неудачными именами или дублированием; тесты поведения.

Выполнить проверки Pylint и Flake8, объяснить несколько полученных сообщений. Проверить форматирование Black в режиме --check, затем применить согласованное исправление. Упростить повторяющийся фрагмент и повторить тесты; сопоставить код до и после.

2. Анализ учебных данных средствами Python

Исходные данные: Небольшой разрешенный CSV или иной согласованный набор профессиональных данных, содержащий пропуски и категории.

Прочитать данные, проверить обязательные поля, согласовать правило обработки пропусков. Посчитать выбранные показатели по категориям, сопоставить один итог с ручным расчетом и представить таблицу или график с пояснениями. Обосновать подходящие средства и ограничения вывода.

Раздел 4. Завершающий этап: Подготовка к демонстрации и подведение итогов

Контролируемые ИДК: ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4

Тема 4.1. Формирование отчетных материалов

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Упорядочьте подготовку отчета после окончания реализации.

А) Сопоставить результаты с заданием
Б) Собрать фактические материалы работы
В) Сформулировать выводы о выполнении

2. Для повторного запуска проверки сначала нужен тот же вариант кода, затем среда. Упорядочьте действия.

А) Подготовить описанную среду и данные
Б) Выполнить приведенную команду тестирования
В) Получить указанную версию кода

3. Тесты в конвейере завершились ошибкой. Какой статус следует указать в отчете до устранения причины?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Автоматический запуск проверок

Исходные данные: Репозиторий учебной программы, команды тестирования и разрешенная среда CI, выбранная руководителем.
Подготовить конфигурацию автоматического запуска тестов при изменении кода: получение версии, подготовка зависимостей, запуск и сохранение результата. Выполнить успешный запуск и контролируемый запуск с заведомо нарушенным ожиданием, затем восстановить корректный код. Сопоставить статусы, журналы и артефакты; описать пределы покрытия.

2. Отчет и воспроизводимый комплект

Исходные данные: Фактически созданные схемы, код, данные, результаты проверок, измерений и история работы.
Собрать отчет: задача, выбор алгоритмов и средств, проектирование, реализация, проверки, личный вклад и ограничения. Подготовить минимальную инструкцию запуска с версиями и входом. Попросить другого участника или руководителя повторить один сценарий по инструкции; учесть замечания.

6. Оценочные материалы промежуточной аттестации

2 семестр, Зачет с оценкой
Контролируемые ИДК: ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4
Вопросы/Задания:

1. Зачем в задании отдельно описывать недопустимый ввод?

2. Что проверить у цикла в блок-схеме перед программированием?

3. Почему выбор алгоритма зависит от размера данных?

4. Какие простые затраты можно сопоставить при выборе учебного решения?

5. Как отличить проверяемое требование от общего пожелания?

6. Что должно быть понятно по схеме взаимодействия компонентов?

7. Зачем записи нужен устойчивый идентификатор?

8. Как согласовать работу двух исполнителей над связанными компонентами?

9. Что связывает угрозу с мерой защиты в учебной модели?

10. Зачем отделять расчет от чтения и показа данных?

11. Что дают понятные имена и документационные строки?

12. Почему линтер и форматтер не заменяют тесты?

13. Что общего у функций на Python и C++ с одним назначением?

14. Почему при анализе данных нужно учитывать пропуски?

15. Как история версий помогает найти ошибку?

16. Как корректно ответить о части, которую не удалось реализовать?

17. Как обозначить совместный и личный результат в отчете?

18. Как пояснить ограниченность измерений производительности?

19. Что проверить, если конвейер сообщает об успехе слишком быстро?

20. Какие материалы подтверждают, что требование выполнено?

21. Почему оптимизацию начинают с измерения?

22. Чем диагностический журнал отличается от сообщения пользователю?

23. Как проверить ожидаемое исключение автоматическим тестом?

24. Какие наблюдения полезно записать при пошаговой отладке?

25. Может ли программа без синтаксических ошибок выдавать неверный ответ?

26. Как просто показать работу алгоритма на малом примере?

27. Почему оценка Big O не является временем в миллисекундах?

28. Какую работу позволяет не повторять динамическое программирование?

29. В чем различаются дерево и произвольный граф?

30. Когда множество полезнее списка?