ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

Решением Ученого совета
АНО ВО «Центральный университет»
«25» мая 2026 г.
Протокол №2

Рабочая программа дисциплины (модуля)
«Продвинутый Python»

Направление подготовки: *02.03.01 Математика и компьютерные науки*

Направленность (профиль) подготовки: *Прикладной искусственный интеллект*

Квалификация (степень) выпускника: *бакалавр*

Форма обучения: *очная*

Срок освоения программы: *4 года*

Год набора: *2026*

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 0215B0C000D1B3FEB44A37EE631E0C27D4
Владелец: Ивашкевич Евгений Васильевич
Действителен: с 13.01.2026 до 13.04.2027

Электронный документ

Москва
2026

Содержание

1. Краткая характеристика дисциплины (модуля)

Рабочая программа дисциплины (модуля) «Продвинутый Python» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования – бакалавриат по направлению 02.03.01 «Математика и компьютерные науки», профиль «Прикладной искусственный интеллект», утвержденный приказом Министерства науки и высшего образования Российской Федерации № 807 от 23.08.2017 года.

Изучение дисциплины (модуля) «Продвинутый Python» позволяет писать качественный, масштабируемый и поддерживаемый код, соответствующий промышленным стандартам разработки. Python является ключевым инструментом в разработке приложений, анализе данных, автоматизации и интеграции систем.

Место дисциплины (модуля) в структуре образовательной программы

Настоящая дисциплина (модуль) включена в учебный план по программе подготовки бакалавриата по направлению 02.03.01 «Математика и компьютерные науки», профиль «Прикладной искусственный интеллект» и входит в обязательную часть Блока 1.

Дисциплина (модуль) изучается на 2 курсе в 3 семестре на выбор, доступна для изучения после успешного освоения дисциплины (модуля) «Язык программирования Python. Основной».

Цель изучения дисциплины (модуля): заключается в формировании систематических навыков разработки структурированного, надёжного и поддерживаемого Python-кода с использованием продвинутых языковых возможностей и современных практик программирования.

Задачи изучения дисциплины (модуля):

— освоение основных и продвинутых конструкций языка Python для эффективной обработки данных и проектирования программ;

— развитие понимания принципов функционального и объектно-ориентированного программирования в контексте реальных задач;

— формирование и развитие способностей проектировать модульный, типизированный и устойчивый к ошибкам код;

— приобретение навыков работы с внешними данными, API и форматами обмена, а также организации кода в полноценные проекты;

— развитие компетенций в написании читаемого, документированного и переиспользуемого кода с применением современных инструментов разработки.

В результате освоения дисциплины (модуля), обучающийся должен:

знать:

— основные конструкции Python и принципы работы с коллекциями, вложенными структурами данных и файлами;

— принципы проектирования функций: параметры, возвращаемые значения, *args, **kwargs, замыкания и чистые функции;

— механизмы обработки ошибок и исключений: try, except, else, finally, raise, пользовательские исключения;

— принципы работы генераторов, генераторных выражений и ленивых вычислений;

— назначение декораторов, их синтаксис и области применения;

— принципы организации кода в модули, пакеты и проекты;

— основы работы с библиотеками, зависимостями, виртуальными окружениями и requirements.txt;

— назначение аннотаций типов, docstring, линтеров, форматтеров и принципов читаемого кода;

— основные принципы объектно-ориентированного программирования: классы, объекты, инкапсуляция, композиция, наследование;

— назначение моделей данных, dataclass, Pydantic и валидации входных данных;

— основы взаимодействия с внешними источниками данных, API, HTTP-запросами и форматами JSON/CSV.

уметь:

— использовать базовые и продвинутые конструкции Python для обработки данных;

— выбирать подходящие структуры данных для решения прикладных задач;

— проектировать функции с понятной сигнатурой и предсказуемым поведением;

— обрабатывать ошибки выполнения и корректно использовать исключения;

— создавать генераторы и применять их для потоковой обработки данных;

— разрабатывать и применять декораторы для вынесения повторяющейся технической логики;

— организовывать код в модули и пакеты, выстраивать структуру небольшого проекта;

— подключать стандартные и внешние библиотеки, управлять зависимостями проекта;

— добавлять аннотации типов, писать документацию к функциям и улучшать читаемость кода;

— проектировать классы, определять их ответственность, реализовывать взаимодействие объектов;

— использовать наследование, композицию и инкапсуляцию при проектировании программ;

— описывать модели данных с помощью dataclass и Pydantic;

— выполнять HTTP-запросы, обрабатывать ответы API и данные в форматах JSON/CSV.

владеть:

— навыками разработки структурированных Python-программ среднего уровня сложности;

— навыками проектирования функций, модулей и классов для решения прикладных задач;

— навыками обработки коллекций, вложенных структур, файлов и внешних данных;

— навыками построения устойчивого кода с обработкой ошибок и валидацией данных;

— навыками применения генераторов и декораторов для повышения эффективности и переиспользуемости кода;

— навыками организации проекта с использованием модулей, пакетов, виртуального окружения и зависимостей;

— навыками написания читаемого, типизированного и поддерживаемого кода;

— навыками объектно-ориентированного проектирования с использованием инкапсуляции, композиции и наследования;

— навыками моделирования данных и проверки их корректности;

— навыками интеграции Python-программ с внешними сервисами и API;

— навыками анализа, отладки и улучшения существующего программного кода.

2. Перечень планируемых результатов обучения

Компетенции, формируемые в результате освоения дисциплины (модуля) при проведении учебных занятий в форме контактной работы обучающихся с педагогическими работниками Университета и в форме самостоятельной работы обучающихся:

Код и содержание компетенции	Индикатор компетенции и перечень планируемых результатов обучения по дисциплине (модулю)
УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1. Знает действующие правовые нормы, регулирующие деятельность в области решения задач, основные методы и подходы к определению круга задач.
	УК-2.2. Умеет определять круг задач в рамках поставленной цели, выбирать оптимальные способы решения задач, учитывая имеющиеся ресурсы и ограничения.
	УК-2.3. Имеет практический опыт применения знаний о правовых нормах и ресурсах в реальных ситуациях, разработки и реализации решений в соответствии с установленными ограничениями.
УК-6. Способен управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни (соответствует УНК-04)	УК-6.1. (УНК-04.01.) Планирует и осуществляет самообучение УНК-04.01.01. У-1. Умеет самостоятельно определять области для профессионального развития и формулировать цели обучения УНК-04.01.01. У-2. Способен разрабатывать план самообучения, выбирать подходящие источники и методы обучения УНК-04.01.01. У-3. Умеет организовывать свое время для регулярного повышения квалификации УНК-04.01.01. У-4. Способен оценивать эффективность проведенного обучения и корректировать план при необходимости УНК-04.01.01. З-1. Знает современные методы и ресурсы для самостоятельного обучения УНК-04.01.01. З-2. Понимает принципы постановки целей и планирования личностного развития УНК-04.01.01. З-3. Знает основы саморегуляции и мотивации для поддержания постоянного профессионального роста
ОПК-4. Способен находить, анализировать, реализовывать программно и использовать на практике математические алгоритмы, в том числе с применением современных вычислительных систем	ОПК-4.1. Знает основные классы математических алгоритмов, методы их анализа, критерии выбора и оценки вычислительной сложности, а также принципы программной реализации.
	ОПК-4.2. Умеет находить и анализировать математические алгоритмы, выбирать подходящий алгоритм для поставленной задачи и реализовывать его с использованием современных вычислительных систем.
	ОПК-4.3. Имеет практический опыт программной реализации, тестирования и оценки корректности и эффективности математических алгоритмов.
ОПК-5. Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности	ОПК-5.1. Знает основные принципы работы современных информационных технологий, архитектуры компьютеров и компьютерных сетей, языков и сред программирования, баз данных и программных систем.
	ОПК-5.2. Умеет выбирать и применять современные информационные технологии, программные средства и сетевые сервисы для решения задач профессиональной деятельности.
	ОПК-5.3. Имеет практический опыт использования современных информационных технологий и программных средств при решении профессиональных задач.

ОПК-6. Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения	ОПК-6.1. Знает принципы алгоритмизации, основные структуры данных, парадигмы программирования, методы разработки, тестирования и отладки компьютерных программ.
	ОПК-6.2. Умеет разрабатывать, документировать, тестировать и отлаживать алгоритмы и компьютерные программы, пригодные для практического применения, с использованием современных языков и сред разработки.
	ОПК-6.3. Имеет практический опыт разработки и сопровождения компьютерных программ и программных комплексов для решения прикладных задач.
МО-1. Способен проектировать, разрабатывает и улучшать алгоритмы и модели машинного обучения	МО-1.1. Применяет классические методы и модели машинного обучения (МО) (базовый) МО-1.1. З-1. Знает алгоритмы обучения с учителем для регрессии и классификации. Линейные методы, ансамблевые методы, методы на опорных векторах.
	МО-1.2. Применяет глубокие нейронные сети (средний) МО-1.2. З-1. Знает теоретические основы нейросетевых алгоритмов.
	МО-1.3. Применяет автоматическое машинное обучение (средний)
ППК-Р1. Способен разрабатывать и отлаживать программный код	ППК-Р1.1. Выполняет формализацию и алгоритмизацию поставленных задач для разработки программного кода (базовый) ППК-Р1.1. З-1. Знает алгоритмы решения типичных задач, области и способы их применения. ППК-Р1.1. З-2. Знает нотации и программное обеспечение для графического отображения алгоритмов. ППК-Р1.1. З-3. Знает методы и приемы алгоритмизации поставленных задач. ППК-Р1.1. У-1. Умеет использовать методы и приемы формализации и алгоритмизации поставленных задач. ППК-Р1.1. У-2. Умеет применять алгоритмы решения типовых задач в соответствующих областях.
	ППК-Р1.2. Формализует задачу ИТ отрасли в язык естественнонаучных дисциплин (базовый) ППК-Р1.2. З-1. Знает основные разделы математики, применяемые для анализа и моделирования непрерывных процессов и дискретных систем в прикладных задачах. ППК-Р1.2. У-1. Умеет выбирать адекватный математический аппарат для формализованного описания сущностей и отношений на основе ТЗ и бизнес-требований. ППК-Р1.2. В-1. Владеет навыками описания задач предметной области в виде формальных математических моделей, пригодных для последующей алгоритмизации и программной реализации.
	ППК-Р1.3. Осуществляет обоснованный выбор методов и алгоритмов для программной реализации формальной математической модели (базовый) ППК-Р1.3. З-1. Знает основные классы методов программной реализации моделей и критерии выбора алгоритмов. ППК-Р1.3. У-1. Умеет проводить сравнительный анализ и

	обоснование выбора алгоритмов для программной реализации модели. ППК-Р1.3. В-1. Владеет методами адаптации методов и алгоритмов под специфику задачи. ППК-Р1.3. В-2. Владеет навыками оценки эффективности выбранных алгоритмов.
	ППК-Р1.4. Разрабатывает программный код с использованием языков программирования (базовый) ППК-Р1.4. З-1. Знает синтаксис выбранного языка программирования, особенности программирования на этом языке, стандартные библиотеки языка программирования. ППК-Р1.4. З-2. Знает методологии разработки компьютерного программного обеспечения. ППК-Р1.4. З-3. Знает технологии программирования. ППК-Р1.4. У-1. Умеет применять выбранные языки программирования для написания программного кода. ППК-Р1.4. У-2. Умеет использовать выбранную среду программирования. ППК-Р1.4. У-3. Умеет использовать возможности имеющейся технической и/или программной архитектуры для написания программного кода.
	ППК-Р1.5. Оформляет программный код в соответствии с установленными требованиями (средний) ППК-Р1.5. З-1. Знает нормативно-технические документы (стандарты и регламенты), определяющие требования к оформлению программного кода. ППК-Р1.5. З-2. Знает основные стандарты оформления технической документации на компьютерное программное обеспечение. ППК-Р1.5. У-1. Умеет применять заданные стандарты и шаблоны для составления и оформления технической документации. ППК-Р1.5. У-2. Умеет применять нормативно-технические документы (стандарты и регламенты), определяющие требования к оформлению программного кода. ППК-Р1.5. У-3. Умеет применять инструментарий для создания и актуализации исходных текстов программ.
	ППК-Р1.6. Работает с системой управления версиями программного кода (средний) ППК-Р1.6. З-1. Знает возможности используемой системы управления версиями и вспомогательных инструментальных программных средств. ППК-Р1.6. З-2. Знает установленный регламент использования системы управления версиями. ППК-Р1.6. У-1. Умеет регистрировать изменения исходного текста программного кода в системе управления версиями. ППК-Р1.6. У-2. Умеет сохранять изменения программного кода в соответствии с регламентом управления версиями. ППК-Р1.6. У-3. Умеет выполнять слияние, разделение и сравнение исходных текстов программного кода.
	ППК-Р1.7. Проверяет и отлаживает программный код (средний) ППК-Р1.7. З-1. Знает методы и приемы отладки программного кода.

	ППК-Р1.7. З-2. Знает типы и форматы сообщений об ошибках, предупреждений. ППК-Р1.7. З-3. Знает способы использования технологических журналов, форматы и типы записей журналов. ППК-Р1.7. У-1. Умеет выявлять ошибки в программном коде. ППК-Р1.7. У-2. Умеет отлаживать программный код на уровне программных модулей. ППК-Р1.7. У-3. Умеет отлаживать программный код на уровне межмодульных взаимодействий и взаимодействий с окружением.
ППК-Р2. Способен проверять работоспособность и проводить рефакторинг кода программного обеспечения	ППК-Р2.1. Разрабатывает тестовые наборы данных для проверки работоспособности компьютерного программного обеспечения (базовый) ППК-Р2.1. З-1. Знает методы создания и документирования контрольных примеров и тестовых наборов данных. ППК-Р2.1. З-2. Знает требования к структуре и форматам хранения тестовых наборов данных. ППК-Р2.1. З-3. Знает правила, алгоритмы и технологии создания тестовых наборов данных. ППК-Р2.1. У-1. Умеет разрабатывать и оформлять контрольные примеры для проверки работоспособности компьютерного программного обеспечения. ППК-Р2.1. У-2. Умеет готовить тестовые наборы данных в соответствии с выбранной методикой тестирования компьютерного программного обеспечения.
	ППК-Р2.2. Проверяет работоспособность компьютерного программного обеспечения (базовый) ППК-Р2.2. З-1. Знает методы и средства проверки работоспособности компьютерного программного обеспечения. ППК-Р2.2. З-2. Знает государственные стандарты испытания автоматизированных систем. ППК-Р2.2. З-3. Знает руководящие документы по стандартизации требований к документам автоматизированных систем. ППК-Р2.2. У-1. Умеет применять методы и средства проверки работоспособности компьютерного программного обеспечения. ППК-Р2.2. У-2. Умеет интерпретировать диагностические данные проверки работоспособности компьютерного программного обеспечения. ППК-Р2.2. У-3. Умеет анализировать значения полученных характеристик компьютерного программного обеспечения.
	ППК-Р2.3. Исправляет дефекты программного кода, зафиксированные в базе данных дефектов (средний) ППК-Р2.3. З-1. Знает типичные ошибки, возникающие при разработке компьютерного программного обеспечения, методы их диагностики и исправления. ППК-Р2.3. З-2. Знает методы и приемы отладки программного кода. ППК-Р2.3. У-1. Умеет воспроизводить дефекты программного кода, зафиксированные в базе данных дефектов. ППК-Р2.3. У-2. Умеет выяснять причины возникновения дефектов программного кода. ППК-Р2.3. У-3. Умеет вносить изменений в программный код для устранения выявленных дефектов.

	ППК-Р2.4. Выполняет рефакторинг и инспекцию программного кода (средний) ППК-Р2.4. З-1. Знает методы и средства рефакторинга и инспекции программного кода. ППК-Р2.4. З-2. Знает нормативно-технические документы (стандарты и регламенты), регламентирующие требования к программному коду, порядок отражения изменений в системе управления версиями. ППК-Р2.4. У-1. Умеет анализировать программный код на соответствие требованиям по читаемости и производительности. ППК-Р2.4. У-2. Умеет проводить инспекцию программного кода для поиска не обнаруженных на ранних стадиях разработки компьютерного программного обеспечения ошибок и критических мест. ППК-Р2.4. У-3. Умеет применять методы и средства рефакторинга и инспекции программного кода. ППК-Р2.4. У-4. Умеет публиковать результаты рефакторинга и инспекции в коллективной базе знаний.
ППК-ДА2. Способен подготавливать и обрабатывать данные для аналитических исследований	ППК-ДА2.1. – Очищает, нормализует и преобразует данные для анализа (базовый) ППК-ДА2.1. З-1. Знает методы предобработки структурированных и неструктурированных данных. ППК-ДА2.1. З-2. Знает алгоритмы нормализации и стандартизации данных. ППК-ДА2.1. З-3. Знает техники обработки пропущенных значений и выбросов. ППК-ДА2.1. У-1. Умеет применять методы очистки данных в зависимости от их типа. ППК-ДА2.1. У-2. Умеет преобразовывать данные в требуемые форматы.
	ППК-ДА2.2. – Обеспечивает соответствие данных требованиям заказчика (базовый) ППК-ДА2.2. З-1. Знает критерии качества данных для различных аналитических задач. ППК-ДА2.2. З-2. Знает отраслевые стандарты представления данных. ППК-ДА2.2. З-3. Знает методы верификации и валидации данных. ППК-ДА2.2. У-1. Умеет проверять данные на соответствие спецификациям. ППК-ДА2.2. У-2. Умеет разрабатывать чек-листы контроля качества данных.
	ППК-ДА2.3. – Формирует датасеты, пригодные для моделирования (средний) ППК-ДА2.3. З-1. Знает принципы формирования обучающих и тестовых выборок. ППК-ДА2.3. З-2. Знает методы балансировки данных. ППК-ДА2.3. З-3. Знает техники аугментации данных. ППК-ДА2.3. У-1. Умеет создавать репрезентативные выборки данных. ППК-ДА2.3. У-2. Умеет применять методы семплирования.

РГ-1. Способен руководить процессами разработки компьютерного программного обеспечения	РГ-1.1. Руководит разработкой программного кода (средний) РГ-1.1. З-1. Знает методы принятия управленческих решений. РГ-1.1. З-2. Знает основные принципы и методы управления персоналом. РГ-1.1. З-3. Знает технологии межличностной и групповой коммуникации в деловом взаимодействии, основы конфликтологии. РГ-1.1. У-1. Способен распределять задачи на разработку программного кода между исполнителями. РГ-1.1. У-2. Применяет лучшие мировые практики оформления программного кода. РГ-1.1. У-3. Оценивает качество и эффективность программного кода.
	РГ-1.2. Руководит проверкой работоспособности компьютерного программного обеспечения (средний) РГ-1.2. З-1. Знает основные методы измерения и оценки характеристик компьютерного программного обеспечения. РГ-1.2. З-2. Знает методы и средства проверки работоспособности компьютерного программного обеспечения. РГ-1.2. З-3. Знает методы и средства рефакторинга и оптимизации программного кода. РГ-1.2. З-4. Знает о возможности отказа частей системы в результате воздействий внутреннего и внешнего нарушителя. РГ-1.2. У-1. Умеет распределять задачи на проверку работоспособности компьютерного программного обеспечения между исполнителями. РГ-1.2. У-2. Умеет оценивать результаты проверки работоспособности компьютерного программного обеспечения. РГ-1.2. У-3. Умеет принимать управленческие решения по результатам проверки работоспособности компьютерного программного обеспечения об исправлении ошибок, рефакторинге, оптимизации и инспекции кода. РГ-1.2. У-4. Умеет включать поиск возможных отказов из воздействий внутреннего или внешнего нарушителя в процедуры инспекции программного кода.
	РГ-1.3. Руководит интеграцией программных модулей и компонентов компьютерного программного обеспечения (продвинутый) РГ-1.3. З-1. Знает методы и средства сборки модулей и компонентов компьютерного программного обеспечения. РГ-1.3. З-2. Знает методы и программные интерфейсы взаимодействия компьютерного программного обеспечения с внешними программными компонентами. РГ-1.3. З-3. Знает методы проектирования и разработки программных интерфейсов взаимодействия внутренних модулей компьютерного программного обеспечения. РГ-1.3. З-4. Знает методы и средства разработки процедур для развертывания компьютерного программного обеспечения. РГ-1.3. У-1. Умеет назначать задания на разработку процедур интеграции, сборку, подключение к внешней среде, проверку работоспособности выпусков программного продукта. РГ-1.3. У-2. Умеет оценивать результаты выполнения назначенных заданий на разработку процедур интеграции, сборку, подключение к внешней среде, проверку работоспособности выпусков программного продукта.

	РГ-1.3. У-3. Умеет принимать управленческие решения по результатам проверки работоспособности выпусков программного продукта.

3. Тематический план

№п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Аудиторная работа		Контроль	Самостоятельная работа	
		Лекции	Практические занятия			
1	Базовые конструкции и обработка данных		4		8	Домашнее задание
2	Функции и функциональные инструменты		20		42	Домашнее задание Контрольная работа
3	Организация проекта и качество кода		12		26	Домашнее задание
4	Объектно-ориентированное программирование		24		50	Домашнее задание Контрольная работа
	Зачёт с оценкой			4		
Итого:			***60***	***4***	***126***	
Объем дисциплины (модуля) (в ак. ч.)		***190***				
Объем дисциплины (модуля) (в зач. ед.)		***5***				

4. Содержание дисциплины (модуля)

№п/п	Наименование раздела дисциплины (модуля)	Содержание дисциплины (модуля) по темам
1	Базовые конструкции и обработка данных	Базовые программы на Python, коллекции, вложенные структуры и файлы
2	Функции и функциональные инструменты	Проектирование функций. Ошибки и исключения. Генераторы. yield и ленивые вычисления. Декораторы и переиспользование логики
3	Организация проекта и качество кода	Модули и организация кода. Библиотеки и зависимости. Типизация и читаемость
4	Объектно-ориентированное программирование	Основы объектно-ориентированного программирования. Проектирование классов: инкапсуляция, ответственность, композиция. Наследование, расширение поведения и магические методы. Модели данных, валидация и контракт данных. Прикладной Python

5. Учебно-методическое обеспечение

Университет располагает полным набором лицензионного и свободно распространяемого программного обеспечения, включая продукты отечественного производства.

Каждый студент в течение всего периода обучения получает индивидуальный неограниченный доступ к электронно-библиотечной системе и электронной информационно-образовательной среде университета. Эти системы предоставляют возможность доступа к ресурсам из любой точки, где есть подключение к сети Интернет, как на территории университета, так и за его пределами.

Студентам обеспечен удаленный доступ к современным профессиональным базам данных и информационным справочным системам.

Основная литература:

1. Чернышев, С. А. Основы программирования на Python : учебник для вузов / С. А. Чернышев. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2026. — 349 с. — (Высшее образование). — ISBN 978-5-534-17139-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/588667.

2. Виафоре, П. Надежный Python : практическое руководство / П. Виафоре, М. Райтман. - Санкт-Петербург : БХВ-Петербург, 2023. - 352 с. - ISBN 978-5-9775-1174-2. - Текст : электронный. - URL: https://znanium.com/catalog/product/2123392.

Дополнительная литература:

1. Бейдер, Д. Чистый Python. Тонкости программирования для профи : практическое руководство / Д. Бейдер. - Санкт-Петербург : Питер, 2021. - 288 с. - (Серия «Библиотека программиста»). - ISBN 978-5-4461-0803-9. - Текст : электронный. - URL: https://znanium.ru/catalog/product/1766370.

2. Хиллард, Д. Секреты Python Pro / Д. Хиллард. - Санкт-Петербург : Питер, 2021. - 320 с. - ISBN 978-5-4461-1684-3. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2142837.

6. Материально-техническое обеспечение

Университет располагает материально-технической базой, соответствующей действующим противопожарным правилам и нормам и обеспечивающей проведение всех видов дисциплинарной и междисциплинарной подготовки, практической и научно-исследовательской работ обучающихся, предусмотренных учебным планом.

Помещения, которые представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского (практического) типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования. Помещения укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Изучение дисциплины (модуля) обеспечивается в учебных аудиториях, оснащенных:

— столами и стульями;
— компьютерной техникой;
— механическими калькуляторами;
— специализированным оборудованием, включая демонстрационное оборудование.

Помещения для самостоятельной работы обучающихся, в том числе приспособленные для использования инвалидами и лицами с ограниченными возможностями здоровья, оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и

обеспечением доступа в электронную информационно-образовательную среду Университета.

Обучающимся предоставляется доступ (в том числе удаленный) к ресурсам информационно-телекоммуникационной сети «Интернет», электронным ресурсам (в том числе электронным библиотечным системам, современным профессиональным базам данных и информационным справочным системам):

№	Наименование портала (издания, курса, документа)	Ссылка
1.	Научная электронная библиотека elibrary.ru библиотека	https://elibrary.ru/defaultx.asp
2.	База данных для IT-специалистов	https://habr.com
3.	База данных ScienceDirect	https://www.sciencedirect.com
4.	Официальный сайт Министерства науки и высшего образования Российской Федерации	https://minobrnauki.gov.ru/
5.	Федеральный портал «Российское образование"	https://www.edu.ru/
6.	Информационная система "Единое окно доступа к образовательным ресурсам"	http://window.edu.ru/
7.	Единая коллекция цифровых образовательных ресурсов	http://school-collection.edu.ru/
8.	Федеральный центр информационно - образовательных ресурсов	http://fcior.edu.ru/

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), в том числе комплект лицензионного программного обеспечения, современные профессиональные базы данных и информационные справочные системы:

Наименование ПО	Производство	Лицензионное / свободно распространяемое
Операционные системы:		
Microsoft Imagine (Windows Client, Server)	зарубежное	лицензионное
Браузеры:		
Яндекс.Браузер	отечественное	свободно распространяемое
Google Chrome	зарубежное	свободно распространяемое
Офисные приложения:		
Microsoft Imagine (Visio, OneNote)	зарубежное	лицензионное
TeXstudio	зарубежное	свободно распространяемое
Adobe Acrobat Reader	зарубежное	свободно распространяемое
Программное обеспечение для планирования и учета времени:		
Toggle app	зарубежное	свободно распространяемое
Системы управления проектами:		
Microsoft Imagine (Project)	зарубежное	лицензионное
Системы управления базами данных:		
Microsoft Imagine (SQL Server)	зарубежное	лицензионное
Системы резервного копирования (backup):		
Acronis Backup Advanced for HyperV	зарубежное	лицензионное
Справочно-правовые системы:		
КонсультантПлюс: справочно-правовая система	отечественное	лицензионное
Средства антивирусной защиты:		
Kaspersky Endpoint Security для бизнеса Стандартный Russian Edition	отечественное	лицензионное
Среды разработки:		

Visual Studio Code	зарубежное	свободно распространяемое
Bash (Unix shell)	зарубежное	свободно распространяемое
Anaconda	зарубежное	свободно распространяемое
Robotic Operating System	зарубежное	свободно распространяемое
CopelliaSim	зарубежное	свободно распространяемое
Google Colaboratory	зарубежное	свободно распространяемое
Пакеты программных средств и библиотек:		
AutoPsy	зарубежное	свободно распространяемое
Interactive Disassembler (IDA)	зарубежное	свободно распространяемое
Системы управления библиографической информацией:		
Zotero	зарубежное	свободно распространяемое
Сервисы и службы:		
Bind	зарубежное	свободно распространяемое
Docker	зарубежное	свободно распространяемое

7. Методические и оценочные материалы

Методические указания для обучающихся по освоению дисциплины (модуля)

В процессе изучения дисциплины (модуля) «Продвинутый Python» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, домашние задания, контрольная работа, а также различные виды самостоятельной работы обучающихся по заданию преподавателя, направленные на развитие навыков профессиональной лексики, закрепление практических профессиональных компетенций, поощрение инициатив.

Лекция – систематическое, последовательное, монологическое изложение преподавателем учебного материала, как правило, теоретического характера.

В процессе лекций рекомендуется вести конспект лекций: кратко и схематично фиксировать основные идеи, выводы и обобщения лекции; выделять важные мысли, ключевые слова и термины. Необходимо отметить вопросы или материалы, которые вызывают затруднения, и попытаться найти ответы в рекомендованной литературе. Если разобраться в материале не удается, следует сформулировать вопрос и задать его преподавателю на консультации или во время семинарского (практического) занятия.

Семинар – это форма учебной деятельности, проводимая в учебном заведении под руководством преподавателя, где студенты активно участвуют в обсуждениях, практических заданиях и других формах взаимодействия.

Для успешной подготовки к семинару рекомендуется заранее ознакомиться с темой занятия и основными материалами, чтобы иметь возможность активно участвовать в обсуждении. Также полезно подготовить вопросы и идеи для обсуждения, что поможет глубже понять материал и продемонстрировать заинтересованность.

Домашнее задание – набор задач по темам недели.

При работе над домашними заданиями важно внимательно ознакомиться с требованиями и сроками выполнения. Рекомендуется разбивать задания на этапы, чтобы избежать перегрузки и лучше усвоить материал. Использовать различные источники информации, включая учебники и онлайн-ресурсы, для более глубокого понимания темы.

Контрольная работа – письменная работа с набором задач, которые нужно решить за ограниченное время.

Цель контрольной работы - получить специальные знания по одной или нескольким темам дисциплины (модуля) и продемонстрировать навыки их практического применения.

Самостоятельная работа – работа студентов, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины (модуля).

В процессе самостоятельной работы студенты взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи студента включают работу с конспектами лекций (обработка текста), повторное изучение учебных материалов планов и тезисов ответов, изучение дополнительных тем, выполнение учебно-исследовательских заданий и другое.

Система оценивания результатов обучения по дисциплине (модулю)

Критерии получения уровня и оценивания сформированности компетенций по дисциплине (модулю) «Продвинутый Python»

Оценивание уровня учебных достижений, обучающихся по дисциплине (модулю), осуществляется в виде текущего контроля успеваемости и промежуточной аттестации.

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме ***зачёта с оценкой***, при этом проводится оценка компетенций, сформированных по дисциплине.

Для оценивания текущего контроля успеваемости и промежуточной аттестации используется десятибалльная шкала оценивания, которая соотносится с традиционной пятибалльной шкалой следующим образом, если иное не указано в силлабусе дисциплины:

Десятибалльная оценка	**Пятибалльная оценка**	**Общая характеристика результата обучения по дисциплине (модулю)**
10	Отлично	Студент полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину. Он самостоятельно и логически последовательно отвечает на все вопросы, акцентируя внимание на наиболее важном. Умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя ключевые моменты и устанавливая причинно-следственные связи. Четко формулирует ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты дисциплины (модуля) с практическими задачами.
9	Отлично	
8	Отлично	
7	Хорошо	Студент обладает знаниями предмета почти в полном объеме рабочей программы и самостоятельно, логически последовательно и всесторонне отвечает на все вопросы, акцентируя внимание на наиболее значимых моментах. Он умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя его ключевые аспекты и устанавливая причинно-следственные связи. Формулирует свои ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные ситуационные задачи. Студент хорошо знаком с методами исследования, необходимыми для
6	Хорошо	

Десятибалльная оценка	Пятибалльная оценка	Общая характеристика результата обучения по дисциплине (модулю)
		практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
5	Удовлетворительно	Студент обладает базовыми знаниями по дисциплине (модулю), но испытывает трудности при самостоятельных ответах и использует неточные формулировки. В ходе ответов он допускает ошибки, касающиеся сути вопросов. Студент способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования.
4	Удовлетворительно	
3	Не сдан	Студент не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы.
2	Не сдан	
1	Не сдан	

Дисциплина (модуль) «Продвинутый Python» оценивается следующим образом:

Активность	Вес	Количество	Описание
Домашнее задание	30%	13	Набор задач по темам недели
Контрольная работа	35%	2	Письменная работа с набором задач, которые нужно решить за ограниченное время
Зачёт с оценкой	35%	1	Письменная или устная работа над заданием, направленным на проверку полученных знаний и навыков по дисциплине (модулю)

Формула расчёта итоговой оценки по дисциплине (модулю) «Продвинутый Python»: «0,3 × среднее за домашние задания + 0,35 × среднее за контрольные работы + 0,35 × зачёт с оценкой».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

Домашнее задание 1.

Задание 1. Напишите программу на Python, которая загружает данные из JSON-файла (например, список пользователей с полями: id, name, email, orders). Обработайте вложенную структуру: каждый пользователь может иметь список заказов с product, price, date. Реализуйте функции для: подсчёта общего числа заказов; нахождения пользователя с наибольшей суммой покупок; фильтрации заказов по дате (например, за последнюю неделю).

Задание 2. Работайте с CSV-файлом, содержащим данные о продажах (столбцы: product, category, price, quantity, date). С помощью стандартных средств Python (без pandas) реализуйте: чтение файла и парсинг строк; группировку по категории и подсчёт общей выручки; запись результата в новый JSON-файл.

Задание 3. Создайте скрипт, который обрабатывает несколько файлов (JSON, CSV) из одной директории. Реализуйте: автоматическое определение формата файла по расширению; унифицированную структуру данных для хранения записей; объединение всех данных в единый список и сохранение в файл all_data.json.

Домашнее задание 2.

Задание 1. Реализуйте функцию process_orders(data, **filters), которая принимает список заказов и произвольные фильтры (например, min_price=100, category="electronics"). Используйте *args и **kwargs для гибкой сигнатуры. Функция должна возвращать отфильтрованный генератор (с yield), чтобы не загружать все данные в память.

Задание 2. Создайте генератор log_parser(filename), который построчно читает лог-файл и возвращает словари с полями: timestamp, level, message. Реализуйте обработку ошибок: если строка не соответствует формату — пропускайте её и выводите предупреждение. Добавьте пользовательское исключение InvalidLogFormatError.

Задание 3. Напишите декоратор @retry(max_attempts=3, delay=1), который повторяет выполнение функции при возникновении исключения. Используйте его для функции, имитирующей HTTP-запрос (с random.choice — успех или ошибка). Декоратор должен выводить сообщение о каждой попытке и задерживаться на указанное время.

Домашнее задание 3.

Задание 1. Организуйте код из ДЗ 1 и 2 в виде модулей: data_loader.py — загрузка и парсинг файлов; filters.py — фильтрация и обработка; utils.py — вспомогательные функции и исключения. Создайте main.py, который импортирует модули и выполняет полный пайплайн обработки. Убедитесь, что структура проекта позволяет легко добавлять новые форматы.

Задание 2. Настройте виртуальное окружение и файл requirements.txt, включающий: requests — для будущих HTTP-запросов; pydantic — для валидации данных; mypy, flake8, black — для контроля качества кода. Добавьте аннотации типов ко всем функциям из data_loader.py и filters.py. Проверьте код с помощью mypy и flake8.

Задание 3. Напишите docstring в формате Google Style для каждой функции. Пример:

```
def load_json_data(path: str) -> List[Dict]:
"""Загружает данные из JSON-файла.
Args:
path: путь к файлу.
Returns:
Список словарей с данными.
Raises:
FileNotFoundError: если файл не найден.
JSONDecodeError: если файл повреждён.
"""
```

Убедитесь, что код соответствует PEP 8 и проходит проверку линтером.

Примерные задания для контрольной работы

1. Напишите функцию multiply(a, b=1, *args, **kwargs), которая возвращает произведение всех переданных числовых аргументов (из a, b, *args). Параметр **kwargs должен игнорироваться. Пример: multiply(2, 3, 4) → 24.
2. Объясните, в чём разница между *args и **kwargs в сигнатуре функции. Приведите пример использования каждого.

3. Напишите генератор fibonacci(n), который возвращает первые n чисел Фибоначчи по одному с помощью yield. Используйте его для вывода первых 10 чисел.

4. Что такое ленивые вычисления? В чём преимущество генератора перед списком при обработке большого объёма данных?

5. Напишите декоратор @timing, который выводит время выполнения функции. Примените его к функции slow_function(), которая делает time.sleep(1) и возвращает True.

6. Объясните, зачем нужны пользовательские исключения. Создайте исключение DataProcessingError и используйте его в блоке try-except, чтобы обработать ошибку при обработке некорректного JSON.

7. Создайте класс Product с атрибутами name, price. Реализуйте метод str , чтобы при выводе объекта печаталось: "Товар: <name>, цена: <price>".

8. Реализуйте класс Cart, который хранит список товаров (объекты Product). Добавьте методы: add_item(product), total() — возвращает общую сумму, и len () — возвращает количество товаров.

9. Что такое инкапсуляция? Как в Python обозначаются защищённые и приватные атрибуты? Приведите пример.

10. Используя библиотеку pydantic, опишите модель User, в которой: name — строка, age — целое число от 0 до 120, email — корректный email. Попробуйте создать объект с некорректным возрастом и объясните, что произойдёт.

Задания для промежуточной аттестации по дисциплине (модулю)

№ п/п	Задание	Ответ	Компетенция	Индикатор
1.	В разделе «Базовые конструкции» укажите структуру Python для хранения упорядоченной изменяемой последовательности.	список / list	УК-2	УК-2.1-З-1
2.	При обработке данных укажите тип данных для хранения пар «ключ-значение».	словарь / dict	УК-2	УК-2.1-З-1
3.	В разделе «Базовые конструкции» укажите способ доступа к значению во вложенном словаре.	dict[key1][key2]	УК-2	УК-2.2-У-1
4.	При работе с файлами укажите встроенный модуль Python.	json	УК-2	УК-2.2-У-2
5.	В разделе «Базовые конструкции» укажите способ чтения CSV-файла без сторонних библиотек.	модуль csv / csv.reader	УК-2	УК-2.3-У-1
6.	При обработке данных укажите оператор для распаковки аргументов функции из списка.	* / звёздочка	УК-2	УК-2.3-У-2
7.	В разделе «Функции» укажите ключевое слово для определения функции в Python.	def	ОПК-4	ОПК-4.1-З-1
8.	При проектировании функций укажите конструкцию для обработки исключений.	try-except / try-except-finally	ОПК-4	ОПК-4.1-З-2
9.	В разделе «Функции» укажите ключевое слово, используемое в генераторе вместо return.	yield	ОПК-4	ОПК-4.2-У-1

10.	При ленивых вычислениях укажите механизм для оборачивания функций с дополнительным поведением.	декоратор / decorator	ОПК-4	ОПК-4.2-У-2
11.	В разделе «Функции» укажите встроенный инструмент для измерения времени выполнения кода.	time.time() / time.perf_counter()	ОПК-4	ОПК-4.3-У-1
12.	При тестировании функций укажите способ проверки корректности работы функции.	assert / unit test	ОПК-4	ОПК-4.3-У-2
13.	В разделе «Организация проекта» укажите способ группировки функций и классов в отдельный файл.	модуль / module	ОПК-5	ОПК-5.1-З-1
14.	При организации кода укажите способ создания пакета в Python.	наличие файла **init**.py	ОПК-5	ОПК-5.1-З-2
15.	В разделе «Организация проекта» укажите синтаксис добавления аннотации типа к параметру функции.	: тип / : str, : int	ОПК-5	ОПК-5.2-У-1
16.	При типизации укажите библиотеку для описания моделей данных с валидацией.	Pydantic	ОПК-5	ОПК-5.2-У-2
17.	В разделе «Организация проекта» укажите инструмент для автоматического форматирования кода по PEP 8.	black / autopep8	ОПК-5	ОПК-5.3-У-1
18.	При качестве кода укажите способ управления зависимостями в проекте.	requirements.txt / pip / poetry	ОПК-5	ОПК-5.3-У-2
19.	В разделе «ООП» укажите принцип, при котором объект содержит ссылки на другие объекты.	композиция / composition	ОПК-6	ОПК-6.1-З-1
20.	При проектировании классов укажите магический метод, вызываемый при создании экземпляра.	**init**	ОПК-6	ОПК-6.1-З-2
21.	В разделе «ООП» укажите механизм наследования атрибутов и методов от другого класса.	наследование / inheritance	ОПК-6	ОПК-6.2-У-1
22.	При инкапсуляции укажите способ указания необязательного поля в классе.	Optional[тип] / = None	ОПК-6	ОПК-6.2-У-2
23.	В разделе «ООП» укажите магический метод для строкового представления объекта.	**str** / **repr**	ОПК-6	ОПК-6.3-У-1
24.	При валидации данных укажите способ проверки корректности данных в классе.	валидатор / property / setter	ОПК-6	ОПК-6.3-У-2
25.	В разделе «Базовые конструкции» укажите метод добавления элемента в конец списка.	append()	ППК-Р1	ППК-Р1.1-З-1

26.	При алгоритмизации укажите способ сортировки списка в Python.	sort() / sorted()	ППК-Р1	ППК-Р1.1-З-2
27.	В разделе «Базовые конструкции» укажите метод для объединения двух словарей.	update() / {**dict1, **dict2}	ППК-Р1	ППК-Р1.1-З-3
28.	При формализации задач укажите способ преобразования строки в число.	int() / float()	ППК-Р1	ППК-Р1.1-У-1
29.	В разделе «Функции» укажите способ передачи переменного количества аргументов в функцию.	*args / **kwargs	ППК-Р1	ППК-Р1.2-У-1
30.	При математической формализации укажите модуль для математических операций.	math / numpy	ППК-Р1	ППК-Р1.2-У-2
31.	В разделе «Функции» укажите способ выбора алгоритма для обработки данных.	анализ сложности / benchmarking	ППК-Р1	ППК-Р1.3-У-1
32.	При оценке эффективности укажите метрику для оценки алгоритма.	O-нотация / время выполнения	ППК-Р1	ППК-Р1.3-В-1
33.	В разделе «Организация проекта» укажите синтаксис языка Python для объявления переменной.	имя = значение	ППК-Р1	ППК-Р1.4-З-1
34.	При разработке кода укажите среду программирования для Python.	PyCharm / VS Code / Jupyter	ППК-Р1	ППК-Р1.4-У-1
35.	В разделе «Организация проекта» укажите стандарт оформления Python-кода.	PEP 8	ППК-Р1	ППК-Р1.5-З-1
36.	При документировании укажите формат документации для функций.	docstring / Google style / NumPy style	ППК-Р1	ППК-Р1.5-У-1
37.	В разделе «ООП» укажите систему управления версиями для кода.	Git	ППК-Р1	ППК-Р1.6-З-1
38.	При работе с Git укажите команду для фиксации изменений.	git commit	ППК-Р1	ППК-Р1.6-У-1
39.	В разделе «Функции» укажите метод отладки программного кода.	pdb / print / logging	ППК-Р1	ППК-Р1.7-З-1
40.	При отладке укажите способ выявления ошибок в коде.	traceback / логирование	ППК-Р1	ППК-Р1.7-У-1
41.	В разделе «ООП» укажите метод создания тестовых данных для класса.	фабрика / фикстуры / моки	ППК-Р2	ППК-Р2.1-З-1
42.	При тестировании укажите формат хранения тестовых данных.	JSON / YAML / CSV	ППК-Р2	ППК-Р2.1-З-2
43.	В разделе «ООП» укажите методологию тестирования классов.	unit testing / pytest	ППК-Р2	ППК-Р2.1-У-1

44.	При подготовке тестов укажите способ генерации тестовых данных.	faker / factory_boy	ППК-Р2	ППК-Р2.1-У-2
45.	В разделе «Организация проекта» укажите средство проверки работоспособности кода.	pytest / unittest	ППК-Р2	ППК-Р2.2-З-1
46.	При проверке укажите стандарт испытания автоматизированных систем.	ГОСТ 19.xxx / ISO	ППК-Р2	ППК-Р2.2-З-2
47.	В разделе «ООП» укажите способ интерпретации результатов тестирования.	отчеты pytest / coverage	ППК-Р2	ППК-Р2.2-У-2
48.	При анализе укажите метрику для оценки качества кода.	покрытие тестами / сложность	ППК-Р2	ППК-Р2.2-У-3
49.	В разделе «Функции» укажите типичную ошибку при работе с исключениями.	bare except / игнорирование ошибок	ППК-Р2	ППК-Р2.3-З-1
50.	При рефакторинге укажите метод улучшения читаемости кода.	extraction / renaming / simplification	ППК-Р2	ППК-Р2.4-У-1