ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

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

Рабочая программа дисциплины (модуля)
«Язык программирования Python. Основной»

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) «Язык программирования Python. Основной» 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) дает умение программирования, используемым в различных областях, таких как веб-разработка, анализ данных и машинное обучение. Освоение Python позволяет эффективно решать сложные задачи в профессиональной деятельности.

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, как дисциплина по выбору.

Дисциплина (модуль) изучается на 1 курсе в 1 семестре.

Цель изучения дисциплины (модуля): формирование у студентов навыков программирования на языке Python, включая разработку, отладку и оптимизацию приложений.

Задачи изучения дисциплины (модуля):

— освоение основ языка Python и принципы структурирования кода;
— понимание базовых принципов анализа и визуализации данных;
— структурирование решений с использованием функций и классов;
— выполнение базовых обработок и анализ данных;

В результате освоения дисциплины (модуля), обучающийся должен:

знать:

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	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.

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.
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.

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	Функции и функциональные инструменты		22	12	38	Домашнее задание Контест Контрольная работа
2	Организация проекта и качество кода		12	6	22	Домашнее задание Контест
3	Объектно-ориентированное программирование		14	8	26	Домашнее задание Контест Контрольная работа
4	Прикладная разработка на Python		8	4	14	Домашнее задание Контест
	Зачет с оценкой			4		
	Итого:		***56***	***34***	***100***	
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	Функции и функциональные инструменты	Базовые программы на Python, коллекции, вложенные структуры и файлы Проектирование функций Ошибки и исключения Генераторы, yield и ленивые вычисления Декораторы и переиспользование логики
2	Организация проекта и качество кода	Модули и организация кода Библиотеки и зависимости Типизация и читаемость
3	Объектно-ориентированное программирование	Основы объектно-ориентированного программирования Проектирование классов: инкапсуляция, ответственность, композиция Наследование, расширение поведения и магические методы
4	Прикладная разработка на Python	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, 2025. — 349 с. — (Высшее образование). — ISBN 978-5-534-17139-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/567821.

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) «Язык программирования Python. Основной» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, коллоквиумы и проекты, 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.

Контест – интерактивная платформа с заданиями разного уровня сложности и автоматической проверкой результатов.

Контест позволяет оперативно оценивать усвоение материала и выявлять пробелы в знаниях через тесты и практические задачи. Такой формат способствует регулярной самопроверке и повышает мотивацию к изучению дисциплины.

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 и продемонстрировать навыки их практического применения.

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.

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 «Язык программирования Python. Основной»

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 ***зачета 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**	**Оценка за зачет**	**Общая характеристика результата обучения по дисциплине (модулю)**
10	Отлично	Зачтено	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
9	Отлично	Зачтено	
8	Отлично	Зачтено	

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
			дисциплины (модуля) с практическими задачами.
7	Хорошо	Зачтено	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 практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
6	Хорошо	Зачтено	
5	Удовлетворительно	Зачтено	Студент обладает базовыми знаниями по дисциплине, 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	Зачтено	
3	Не сдан	Не зачтено	Студент не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы.
2	Не сдан	Не зачтено	
1	Не сдан	Не зачтено	

Дисциплина (модуль) «Язык программирования Python. Основной» оценивается следующим образом:

Активность	Вес	Количество	Описание
Домашнее задание	15%	13	Устные ответы на вопросы, список которых известен студенту заранее
Контест	25%	13	Интерактивные задания разного уровня сложности с автоматической проверкой результатов

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

Формула расчёта итоговой оценки по дисциплине (модулю) «Язык программирования Python. Основной»: «0,15 × среднее за домашние задания + 0,25 × среднее за контесты +0,3 × среднее за контрольные работы + 0,3 × зачет с оценкой ».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

1. Функции и модули

1. Напишите модуль **math_utils.py** с функциями:
— **gcd(a, b)** — НОД двух чисел,
— **lcm(a, b)** — НОК,
— **is_prime(n)** — проверка на простоту.
2. Импортируйте модуль в основной файл и протестируйте функции.

2. Работа с файлами и форматами

1. Прочитайте текст из файла **input.txt**, подсчитайте количество слов, строк и символов.
2. Запишите результат в **stats.txt**.
3. Сохраните данные в формате JSON:
json
Свернуть
Копировать
```
1
```
{"lines": 10, "words": 150, "chars": 900}

3. Словари и коллекции

1. Создайте словарь с данными о студентах: имя, возраст, оценки (список).
2. Напишите функции:
— **average_grade(student)** — средний балл студента,
— **top_student(students)** — студент с лучшим средним баллом.
3. Выведите результат.

4. Исключения и обработка ошибок

1. Напишите программу, которая запрашивает число и возводит его в квадрат.
2. Обработайте:
— **ValueError** — если введена не цифра,
— **KeyboardInterrupt** — если пользователь нажал Ctrl+C.
3. Используйте **try-except-else-finally**.

5. Объектно-ориентированное программирование (ООП)

1. Создайте класс **BankAccount** с атрибутами:
— **owner** (имя владельца),

— **balance** (баланс, по умолчанию 0).
2. Реализуйте методы:
— **deposit(amount)**,
— **withdraw(amount)**,
— **__str__()** — строковое представление.
3. Создайте два счёта, внесите и снимите деньги, выведите информацию.

6. Работа с датой и временем
1. Напишите программу, которая:
— Выводит текущую дату и время,
— Определяет, сколько дней осталось до Нового года,
— Проверяет, был ли введённый год високосным.
2. Используйте модуль **datetime**.

7. Работа с JSON и API (на примере)
1. Создайте словарь с данными о книгах: название, автор, год.
2. Сохраните в **books.json**.
3. Прочитайте файл и выведите все книги, изданные после 2000 года.

8. Генераторы и итераторы
1. Напишите генератор **fibonacci(n)**, возвращающий первые **n** чисел Фибоначчи.
2. Напишите генератор **even_numbers(start, end)**, возвращающий чётные числа в диапазоне.
3. Используйте **yield**.

9. Декораторы
1. Напишите декоратор **@timer**, который измеряет время выполнения функции.
2. Примените его к функции **slow_function()**, которая спит 2 секунды (**time.sleep(2)**).
3. Выведите: **Функция выполнилась за 2.03 сек**.

10. Итоговый проект: "Система учёта задач (To-Do List)"
Реализуйте консольное приложение с функциями:
— Добавление задачи (название, приоритет, статус).
— Просмотр всех задач.
— Фильтрация по статусу (выполнено/не выполнено).
— Сохранение и загрузка из **tasks.json**.
— Использование классов, файлов, исключений, JSON.

Примерные задания для контеста

№	ЗАДАНИЕ	ОГРАНИЧЕНИЯ	ПРИМЕР
1	**Сумма простых чисел до N**	**1 ≤ N ≤ 1000**	Ввод: **10** → Вывод: **17** (2+3+5+7)

№	ЗАДАНИЕ	ОГРАНИЧЕНИЯ	ПРИМЕР
2	**Наибольшая подстрока без повторов**	Длина строки ≤ 100	Ввод: **abcabcbb** → Вывод: **3** (**abc**)
3	**Слияние словарей с суммированием**	Два словаря	Ввод: **{'a': 1, 'b': 2}**, **{'b': 3, 'c': 1}** → Вывод: **{'a': 1, 'b': 5, 'c': 1}**
4	**Палиндром с учётом регистра и пробелов**	Строка ≤ 100 символов	Ввод: **A man a plan a canal Panama** → Вывод: **Да**
5	**Разложение числа на простые множители**	**2 ≤ N ≤ 1000**	Ввод: **12** → Вывод: **[2, 2, 3]**
6	**Самая длинная общая подпоследовательность (LCS)**	Две строки ≤ 50	Ввод: **abcde**, **ace** → Вывод: **3** (**ace**)
7	**Генерация всех перестановок**	Строка из 3–6 символов	Ввод: **abc** → Вывод: **['abc', 'acb', ...]**
8	**Проверка валидности email**	Только латиница, **@**, **.**	Ввод: **test@mail.ru** → Вывод: **Да**
9	**Подсчёт слов в тексте с сортировкой по частоте**	Текст до 1000 слов	Ввод: **hello world hello** → Вывод: **[('hello', 2), ('world', 1)]**
10	**Реализация стека с проверкой скобок**	Использовать список как стек	Ввод: **{[()]}** → Вывод: **Да**

Примерные задания для контрольных работ

Контрольная работа №1 (Темы: функции, исключения, файлы)

Время: 60 минут

1. Напишите функцию **read_file(filename)**, которая читает файл и возвращает его содержимое. Обработайте **FileNotFoundError**.
2. Напишите функцию **factorial(n)**, выбрасывающую **ValueError**, если **n < 0**.
3. Что выведет код?

```
def func(x, y=[]):
    y.append(x)
    return y
print(func(1))
print(func(2))
```

4. Запишите словарь **{'name': 'Alice', 'age': 25}** в файл **user.json**.
5. Найдите ошибку:

python
Свернуть
Копировать
1 data = json.load("data.txt")
Ответы:
3. **[1]**, **[1, 2]** (из-за мутабельного аргумента по умолчанию)
5. **json.load()** принимает файловый объект, а не строку → нужно **open("data.txt")**

Задания для промежуточной аттестации по дисциплине (модулю)

№ п/п	Задание	Ответ	Компетенция	Индикатор
1.	В разделе «Функции» укажите ключевое слово для определения функции.	def	ОПК-6	ОПК-6.1-З-1
2.	При проектировании функций укажите способ передачи переменного числа аргументов.	*args / **kwargs	ППК-Р1	ППК-Р1.1-З-1
3.	В разделе «Функции» укажите механизм для ленивых вычислений.	Генератор / yield	ОПК-4	ОПК-4.2-У-1
4.	При обработке ошибок укажите конструкцию для обработки исключений.	try-except	ППК-Р1	ППК-Р1.7-З-1
5.	В разделе «Функции» укажите инструмент для переиспользования логики.	Декоратор / decorator	ОПК-6	ОПК-6.1-З-1
6.	При работе с файлами укажите функцию для открытия файла.	open()	ППК-Р1	ППК-Р1.4-У-1
7.	В разделе «Функции» укажите тип данных для хранения пар ключ-значение.	dict / словарь	ОПК-5	ОПК-5.1-З-1
8.	При_collections укажите метод добавления элемента в список.	append()	ППК-Р1	ППК-Р1.4-З-1
9.	В разделе «Функции» укажите способ чтения файла построчно.	for line in file	ППК-Р1	ППК-Р1.4-У-1
10.	При отладке укажите модуль для пошаговой отладки.	pdb	ППК-Р1	ППК-Р1.7-У-2
11.	В разделе «Функции» укажите метод выявления ошибок в коде.	traceback / logging	ППК-Р1	ППК-Р1.7-У-1
12.	При тестировании укажите способ проверки корректности функции.	assert / unittest	ОПК-6	ОПК-6.2-У-2
13.	В разделе «Функции» укажите принцип выбора алгоритма для задачи.	Анализ сложности	ППК-Р1	ППК-Р1.3-У-1
14.	В разделе «Организация проекта» укажите способ группировки кода в файл.	Модуль / module	ОПК-5	ОПК-5.1-З-1

15.	При организации кода укажите способ создания пакета.	**init**.py	ОПК-5	ОПК-5.2-У-1
16.	В разделе «Организация проекта» укажите файл для управления зависимостями.	requirements.txt / pyproject.toml	ППК-Р1	ППК-Р1.4-З-2
17.	При типизации укажите синтаксис аннотации типа параметра.	: тип / : str	ППК-Р1	ППК-Р1.5-У-2
18.	В разделе «Организация проекта» укажите инструмент для форматирования кода.	black / autopep8	ППК-Р1	ППК-Р1.5-У-3
19.	При качестве кода укажите стандарт оформления Python-кода.	PEP 8	ППК-Р1	ППК-Р1.5-З-1
20.	В разделе «Организация проекта» укажите команду Git для фиксации изменений.	git commit	ППК-Р1	ППК-Р1.6-У-1
21.	При версиях укажите способ слияния веток кода.	git merge	ППК-Р1	ППК-Р1.6-У-3
22.	В разделе «Организация проекта» укажите регламент использования Git.	Git flow / branching strategy	ППК-Р1	ППК-Р1.6-З-2
23.	При документации укажите формат docstring для функций.	Google style / NumPy style	ППК-Р1	ППК-Р1.5-У-1
24.	В разделе «Организация проекта» укажите инструмент для создания текстов программ.	IDE / Редактор кода	ППК-Р1	ППК-Р1.5-У-3
25.	При самообучении укажите способ определения областей развития.	Анализ навыков / Skill gap	УК-6	УК-6.1-У-1
26.	В разделе «ООП» укажите принцип содержания ссылок на другие объекты.	Композиция / composition	ОПК-6	ОПК-6.1-З-1
27.	При проектировании классов укажите магический метод инициализации.	**init**	ОПК-6	ОПК-6.1-З-1
28.	В разделе «ООП» укажите механизм наследования атрибутов и методов.	Наследование / inheritance	ОПК-6	ОПК-6.2-У-2
29.	При инкапсуляции укажите способ указания необязательного поля.	Optional[тип] / = None	ППК-Р1	ППК-Р1.4-З-1
30.	В разделе «ООП» укажите магический метод для строкового представления.	**str** / **repr**	ОПК-6	ОПК-6.2-У-2
31.	При валидации укажите способ проверки данных в классе.	property / setter / валидатор	ППК-Р1	ППК-Р1.7-У-2
32.	В разделе «ООП» укажите парадигму программирования с классами.	ООП / Объектно-ориентированное	ОПК-6	ОПК-6.1-З-1

33.	При разработке укажите среду программирования для Python.	PyCharm / VS Code	ППК-Р1	ППК-Р1.4-У-2
34.	В разделе «ООП» укажите метод отладки межмодульных взаимодействий.	Integration tests	ППК-Р1	ППК-Р1.7-У-3
35.	При сопровождении укажите опыт разработки программных комплексов.	Практический проект	ОПК-6	ОПК-6.3-У-3
36.	В разделе «ООП» укажите способ расширения поведения класса.	Наследование / миксины	ОПК-6	ОПК-6.2-У-2
37.	При ответственности укажите принцип единственной ответственности.	SRP / Single Responsibility	ОПК-6	ОПК-6.1-З-1
38.	В разделе «Прикладная разработка» укажите библиотеку для валидации данных.	Pydantic	ППК-Р1	ППК-Р1.4-З-1
39.	При моделях данных укажите способ описания контракта данных.	Dataclass / TypedDict	ППК-Р1	ППК-Р1.2-В-1
40.	В разделе «Прикладная разработка» укажите метод формализации задачи.	Алгоритмизация	ППК-Р1	ППК-Р1.1-У-1
41.	При математике укажите раздел для моделирования дискретных систем.	Дискретная математика	ППК-Р1	ППК-Р1.2-З-1
42.	В разделе «Прикладная разработка» укажите способ выбора алгоритма.	Сравнительный анализ	ППК-Р1	ППК-Р1.3-У-1
43.	При адаптации укажите метод подстройки алгоритма под задачу.	Адаптация / кастомизация	ППК-Р1	ППК-Р1.3-В-1
44.	В разделе «Прикладная разработка» укажите навык оценки эффективности.	Бенчмаркинг / profiling	ППК-Р1	ППК-Р1.3-В-2
45.	При технологиях укажите знание технологий программирования.	Языки / фреймворки	ППК-Р1	ППК-Р1.4-З-3
46.	В разделе «Прикладная разработка» укажите возможность архитектуры для кода.	Микросервисы / модульность	ППК-Р1	ППК-Р1.4-У-3
47.	При стандартах укажите нормативный документ для оформления кода.	ГОСТ / PEP 8	ППК-Р1	ППК-Р1.5-З-1
48.	В разделе «Прикладная разработка» укажите формат сообщений об ошибках.	Traceback / Exception	ППК-Р1	ППК-Р1.7-З-2
49.	При журналах укажите способ использования технологических журналов.	logging module	ППК-Р1	ППК-Р1.7-З-3
50.	В разделе «Функции» укажите метод определения круга задач для проекта.	Анализ требований	УК-2	УК-2.2-У-1