Федеральное государственное автономное образовательное учреждение высшего образования
«Российский университет дружбы народов имени Патриса Лумумбы»

Факультет искусственного интеллекта
(наименование основного учебного подразделения (ОУП)-разработчика ОП ВО)

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ PYTHON
(наименование дисциплины/модуля)

Рекомендована МССН для направлений подготовки:

02.03.02 ФУНДАМЕНТАЛЬНАЯ ИНФОРМАТИКА И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ;
09.03.03 ПРИКЛАДНАЯ ИНФОРМАТИКА
(код и наименование направления подготовки/специальности)

Освоение дисциплины ведется в рамках реализации основной профессиональной образовательной программы высшего образования (ОП ВО):

ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ: РАЗРАБОТКА И ОБУЧЕНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ
(наименование (профиль/специализация) ОП ВО)

2026 г.

1. ЦЕЛЬ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Дисциплина «Программирование на языке Python» входит в программу бакалавриата «Искусственный интеллект: разработка и обучение интеллектуальных систем» по направлениям подготовки 02.03.02 Фундаментальная информатика и информационные технологии и 09.03.03 Прикладная информатика, и изучается во 2, 3 семестрах 1, 2 курсов. Дисциплину реализует Кафедра прикладного искусственного интеллекта. Дисциплина состоит из 6 разделов и 68 тем и направлена на изучение языка программирования Python как основного инструмента разработки в области искусственного интеллекта — от базового синтаксиса и структур данных до объектно-ориентированного программирования, работы с библиотеками научных вычислений (NumPy, Pandas), визуализации данных, создания прототипов ИИ-приложений, автоматизации тестирования и организации программного кода для командной разработки.

Целью освоения дисциплины является формирование у студентов уверенного владения языком Python для решения профессиональных задач в области ИИ: написания чистого, тестируемого и документированного кода, эффективной работы с данными с использованием специализированных библиотек, создания интерактивных прототипов ИИ-решений, организации автоматизированного тестирования, а также развитие навыков командной разработки и информационной грамотности при поиске технических решений.

2. ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Освоение дисциплины «Программирование на языке Python» направлено на формирование у обучающихся следующих компетенций (части компетенций):

Таблица 2.1. Перечень компетенций, формируемых у обучающихся при освоении дисциплины (результаты освоения дисциплины)

Шифр	**Компетенция**	**Индикаторы достижения компетенции** (в рамках данной дисциплины)
УК-12	Способен: искать нужные источники информации и данные, воспринимать, анализировать, запоминать и передавать информацию с использованием цифровых средств, а также с помощью алгоритмов при работе с полученными из различных источников данными с целью эффективного использования полученной информации для решения задач; проводить оценку информации, ее достоверность, строить логические умозаключения на основании поступающих информации и данных	УК-12.1 Способен искать нужные источники информации и данные, воспринимать, анализировать, запоминать и передавать информацию с использованием цифровых средств, а также с помощью алгоритмов при работе с полученными из различных источников данными с целью эффективного использования полученной информации для решения задач;
ОПК-2	Способен понимать принципы работы современных информационных технологий и применять компьютерные методы, современное программное обеспечение, в том числе отечественного происхождения, для решения задач профессиональной	ОПК-2.2 Умеет применять современное программное обеспечение (в том числе отечественного происхождения), фреймворки машинного обучения и инструменты обработки данных для решения задач в области ИИ;

Шифр	Компетенция	Индикаторы достижения компетенции (в рамках данной дисциплины)
	деятельности	
ОПК-3	Способен разрабатывать алгоритмические и программные решения в области системного и прикладного программирования, математических и информационных моделей, баз данных, средств тестирования, пригодные для практического применения	ОПК-3.2 Умеет разрабатывать программные решения на языках Python и C++ для задач обработки данных, машинного обучения и глубокого обучения, проектировать и реализовывать базы данных; ОПК-3.3 Владеет навыками создания, тестирования и отладки алгоритмических и программных решений для систем ИИ, включая разработку пайплайнов обработки данных и обучения моделей;
ПК-2	Способен проектировать архитектуру информационных систем с компонентами ИИ, разрабатывать прототипы и базы данных таких систем	ПК-2.2 Разрабатывает прототипы ИС с элементами ИИ, проводит их валидацию с заинтересованными сторонами;
ПК-3	Способен разрабатывать и реализовывать стратегии тестирования и контроля качества программного обеспечения систем ИИ	ПК-3.2 Разрабатывает план тестирования и организационные документы для тестирования ПО систем ИИ;
PL-1	Способен применять язык программирования Python для решения задач в области ИИ	PL-1.1 Разрабатывает и отлаживает прикладные решения разной сложности и для разного круга конечных пользователей с использованием языка программирования Python, тестирует, испытывает и оценивает качество таких решений; PL-1.2 Осуществляет выбор инструментов разработки на Python, приемлемых для создания прикладной системы обработки научных данных, машинного обучения и визуализации с заданными требованиями;
SS-2	Способен к эффективной коммуникации и командной работе в междисциплинарных проектах в области ИИ	SS-2.1 Эффективно коммуницирует с участниками проектной команды при планировании, реализации и анализе результатов работы в контексте гибридной команды "Человек+ИИ", включая постановку задач людям и ИИ-агентам, фиксацию договорённостей и критериев качества;

3. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОП ВО

Дисциплина «Программирование на языке Python» относится к обязательной части блока 1 «Дисциплины (модули)» образовательной программы высшего образования.

В рамках образовательной программы высшего образования обучающиеся также осваивают другие дисциплины и/или практики, способствующие достижению запланированных результатов освоения дисциплины «Программирование на языке Python».

Таблица 3.1. Перечень компонентов ОП ВО, способствующих достижению запланированных результатов освоения дисциплины

Шифр	Наименование компетенции	Предшествующие дисциплины/модули, практики*	Последующие дисциплины/модули, практики*
УК-12	Способен: искать нужные источники информации и данные, воспринимать, анализировать, запоминать и передавать информацию с использованием цифровых средств, а также	Введение в искусственный интеллект;	Методы разработки решений на основе искусственного интеллекта (Git, Docker); Введение в базы данных; *Вайб-кодинг***; Методы машинного

Шифр	Наименование компетенции	Предшествующие дисциплины/модули, практики*	Последующие дисциплины/модули, практики*
	с помощью алгоритмов при работе с полученными из различных источников данными с целью эффективного использования полученной информации для решения задач; проводить оценку информации, ее достоверность, строить логические умозаключения на основании поступающих информации и данных		обучения; Эксплуатационная практика (учебная);
ОПК-2	Способен понимать принципы работы современных информационных технологий и применять компьютерные методы, современное программное обеспечение, в том числе отечественного происхождения, для решения задач профессиональной деятельности	История и теория программирования; Введение в искусственный интеллект;	Параллельное и распределенное программирование; Массово-параллельные вычисления в машинном обучении (GPU); Основы глубокого обучения; Hadoop, SPARK; Методы машинного обучения; Нейронные сети;
ОПК-3	Способен разрабатывать алгоритмические и программные решения в области системного и прикладного программирования, математических и информационных моделей, баз данных, средств тестирования, пригодные для практического применения	Дискретная математика; История и теория программирования;	Программирование на языке C++; Введение в базы данных; Параллельное и распределенное программирование; Методы разработки решений на основе искусственного интеллекта (Git, Docker); Методы машинного обучения; Нейронные сети; Технологическая (проектно-технологическая) практика (производственная);
ПК-2	Способен проектировать архитектуру информационных систем с компонентами ИИ, разрабатывать прототипы и базы данных таких систем		Эксплуатационная практика (производственная); Преддипломная практика; Эксплуатационная практика (учебная); Технологическая (проектно-технологическая) практика (производственная); Программирование на языке C++; Параллельное и распределенное программирование; Методы разработки решений на основе искусственного интеллекта

Шифр	Наименование компетенции	Предшествующие дисциплины/модули, практики*	Последующие дисциплины/модули, практики*
			(Git, Docker); Hadoop, SPARK; Массово-параллельные вычисления в машинном обучении (GPU); MLOps и промышленная разработка систем искусственного интеллекта; Практическая подготовка на проектах отраслевых индустриальных партнеров; Проектирование и разработка систем компьютерного зрения; Практикум по обработке естественного языка (NLP); Основы глубокого обучения; *Вайб-кодинг***; Введение в базы данных; Онтология и графы знаний;
ПК-3	Способен разрабатывать и реализовывать стратегии тестирования и контроля качества программного обеспечения систем ИИ		Методы машинного обучения; Нейронные сети; Безопасность систем искусственного интеллекта; Обработка и анализ изображений и видео с помощью методов искусственного интеллекта; Анализ естественного языка с помощью методов искусственного интеллекта; Методы разработки решений на основе искусственного интеллекта (Git, Docker); MLOps и промышленная разработка систем искусственного интеллекта; Проектирование и разработка систем компьютерного зрения; Практикум по обработке естественного языка (NLP); Оптимизация моделей машинного обучения; Практическая подготовка на проектах отраслевых индустриальных партнеров; Преддипломная практика; Технологическая (проектно-технологическая) практика (производственная); Эксплуатационная практика (учебная); Эксплуатационная практика (производственная);

Шифр	Наименование компетенции	Предшествующие дисциплины/модули, практики*	Последующие дисциплины/модули, практики*
SS-2	Способен к эффективной коммуникации и командной работе в междисциплинарных проектах в области ИИ	*Иностранный язык***; *Русский язык (как иностранный)***;	Эксплуатационная практика (учебная); Эксплуатационная практика (производственная); Технологическая (проектно-технологическая) практика (производственная); Методы разработки решений на основе искусственного интеллекта (Git, Docker); MLOps и промышленная разработка систем искусственного интеллекта; Практическая подготовка на проектах отраслевых индустриальных партнеров; Проектирование и разработка систем компьютерного зрения; Практикум по обработке естественного языка (NLP); *Большие языковые модели***; *Вайб-кодинг***; *Иностранный язык***; *Русский язык (как иностранный)***; *Иностранный язык в профессиональной деятельности***; *Русский язык (как иностранный) в профессиональной деятельности***;
PL-1	Способен применять язык программирования Python для решения задач в области ИИ		Технологическая (проектно-технологическая) практика (производственная); Эксплуатационная практика (производственная); *Вайб-кодинг***; Методы машинного обучения; Основы глубокого обучения; Параллельное и распределенное программирование; Hadoop, SPARK;

* - заполняется в соответствии с матрицей компетенций и СУП ОП ВО

** - элективные дисциплины /практики

4. ОБЪЕМ ДИСЦИПЛИНЫ И ВИДЫ УЧЕБНОЙ РАБОТЫ

Общая трудоемкость дисциплины «Программирование на языке Python» составляет «6» зачетных единиц.

Таблица 4.1. Виды учебной работы по периодам освоения образовательной программы высшего образования для очной формы обучения.

Вид учебной работы	**ВСЕГО,** ак.ч.		**Семестр(-ы)**	
			2	**3**
Контактная работа, ак.ч.	*136*		*68*	*68*
Лекции (**ЛК**)	0		0	0
Лабораторные работы (**ЛР**)	0		0	0
Практические/семинарские занятия (**СЗ**)	136		68	68
Самостоятельная работа обучающихся, ак.ч.	*53*		*40*	*13*
Контроль (экзамен/зачет с оценкой), ак.ч.	*27*		*0*	*27*
Общая трудоемкость дисциплины	**ак.ч.**	***216***	*108*	*108*
	зач.ед.	***6***	*3*	*3*

5. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Таблица 5.1. Содержание дисциплины (модуля) по видам учебной работы

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
Раздел 1	Основы Python: синтаксис, типы данных, управление потоком	1.1	Введение в Python. Среда разработки и первая программа	Установка Python и настройка среды разработки (VS Code, PyCharm, Jupyter Notebook). REPL-режим. Запуск скриптов. Первая программа: print(), input(). Комментарии. Стиль кода PEP 8. Обзор экосистемы Python для ИИ: почему Python стал стандартом в ML/DL	СЗ	ОПК-3.2, PL-1.1
		1.2	Переменные, числовые типы и арифметические операции	Переменные: присваивание, именование (snake_case). Типы данных: int, float, complex, bool. Арифметические операции: +, −, *, /, //, %, **. Приоритет операций. Функция type(). Динамическая типизация. Приведение типов: int(), float(), str()	СЗ	ОПК-3.2, PL-1.1
		1.3	Строки: создание, индексация, методы	Строки как последовательности: создание, конкатенация, дублирование. Индексация и срезы (slicing). Основные методы: upper(), lower(), strip(), split(), join(), replace(), find(), count(). F-строки (formatted string literals). Строки как данные в NLP: представление текстов	СЗ	ОПК-3.2, PL-1.1
		1.4	Условные конструкции	Логический тип bool. Операторы сравнения: ==, !=, <, >, <=, >=. Логические операторы: and, or, not. Условный оператор if-elif-else. Тернарный оператор. Вложенные условия. Практика: классификация числового значения (положительное/отрицательное/ноль) — простейшая «классификация»	СЗ	ОПК-3.2, PL-1.1
		1.5	Циклы for и while	Цикл for: итерация по последовательностям, функция range(). Цикл while: условие продолжения. Управление циклом: break, continue, else. Вложенные циклы. Практика: генерация таблицы умножения, подсчёт суммы элементов, поиск элемента перебором	СЗ	ОПК-3.2, PL-1.1
		1.6	Списки: создание, методы, срезы	Создание списков. Индексация и срезы. Методы: append(), extend(), insert(), remove(), pop(), sort(), reverse(), index(), count(). Копирование списков: поверхностное и глубокое. Списковые включения (list comprehension). Списки как основная структура для хранения данных	СЗ	ОПК-3.2, PL-1.1
		1.7	Кортежи, множества, словари	Кортежи (tuple): неизменяемые последовательности,	СЗ	ОПК-3.2,

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				распаковка, именованные кортежи (namedtuple). Множества (set): операции объединения, пересечения, разности. Словари (dict): ключ-значение, методы keys(), values(), items(), get(), setdefault(). Dict comprehension. Словари как JSON-подобные структуры данных		PL-1.1
		1.8	Функции: определение, аргументы, возвращаемые значения	Определение функции: def, return. Позиционные и именованные аргументы. Значения по умолчанию. *args и **kwargs. Область видимости: LEGB. Документирование функций: docstrings. Аннотации типов (type hints): def func(x: int) -> str. Функция как базовый блок модульного кода	СЗ	ОПК-3.2, PL-1.1, PL-1.2
		1.9	Лямбда-функции, map, filter, reduce	Анонимные функции: lambda. Функции высшего порядка: map(), filter(). Модуль functools: reduce(). Генераторные выражения. Связь с функциональным стилем обработки данных: пайплайн преобразований как последовательность map/filter	СЗ	ОПК-3.2, PL-1.1
		1.10	Работа с файлами	Открытие и закрытие файлов: open(), close(), контекстный менеджер with. Режимы: ‘r’, ‘w’, ‘a’, ‘rb’, ‘wb’. Чтение: read(), readline(), readlines(). Запись: write(), writelines(). Работа с CSV: модуль csv. Работа с JSON: модуль json. Чтение и запись данных — фундамент для загрузки датасетов	СЗ	ОПК-3.2, PL-1.1
		1.11	Обработка исключений	Исключения: TypeError, ValueError, FileNotFoundError, KeyError, IndexError. Конструкция try-except-else-finally. Множественные except. Генерация исключений: raise. Пользовательские исключения (class MyError(Exception)). Практика: robust-обработка ввода пользователя	СЗ	ОПК-3.2, PL-1.1, PL-1.2
		1.12	Контрольная работа: основы Python	Самостоятельная работа: решение набора задач на базовый синтаксис Python — переменные, условия, циклы, функции, структуры данных, файлы, обработка исключений. Задачи включают обработку текстовых и числовых данных	СЗ	ОПК-3.2, PL-1.1
Раздел 2	Объектно-ориентированное программирование и модульная разработка	2.1	Классы и объекты: основы ООП	Понятие класса и объекта. Определение класса: class. Конструктор init(). Атрибуты экземпляра и класса. Методы. Ключевое слово self. Создание экземпляров. Пример: класс Dataset с атрибутами data, labels и методом summary()	СЗ	ОПК-3.2, PL-1.1, PL-1.2

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
		2.2	Инкапсуляция, свойства, специальные методы	Инкапсуляция: соглашения о приватности (_private, __mangled). Свойства: @property, @setter. Специальные (магические) методы: str(), repr(), len(), getitem(), eq(). Перегрузка операторов. Пример: класс Matrix с операцией сложения	СЗ	ОПК-3.2, PL-1.1, PL-1.2
		2.3	Наследование и полиморфизм	Наследование: синтаксис, вызов super(). Переопределение методов. Множественное наследование и MRO (Method Resolution Order). Полиморфизм: единый интерфейс для разных типов. Абстрактные классы: модуль abc. Пример: иерархия Model → LinearModel, TreeModel с методом predict()	СЗ	ОПК-3.2, PL-1.1, PL-1.2, ПК-2.2
		2.4	Итераторы и генераторы	Протокол итератора: iter(), next(). Создание пользовательских итераторов. Генераторы: yield, yield from. Генераторные выражения. Ленивые вычисления и экономия памяти. Применение: потоковая обработка больших файлов данных (чтение датасета по батчам)	СЗ	ОПК-3.2, PL-1.1, PL-1.2
		2.5	Декораторы и контекстные менеджеры	Декораторы функций: @decorator. Создание собственных декораторов. functools.wraps. Примеры: @timer, @logger, @retry. Декораторы классов (обзор). Контекстные менеджеры: enter(), exit(), contextlib. Применение: замер времени выполнения ML-пайплайна	СЗ	ОПК-3.2, PL-1.1, PL-1.2
		2.6	Модули и пакеты. Виртуальные окружения	Модули: import, from … import, as. Пакеты: init.py. Стандартная библиотека: os, sys, pathlib, datetime, collections, itertools, re. Виртуальные окружения: venv, conda. Менеджер пакетов pip. Файл requirements.txt. Воспроизводимость окружения для ML-проекта	СЗ	ОПК-3.2, PL-1.1, ОПК-2.2
		2.7	Работа с регулярными выражениями	Модуль re: match(), search(), findall(), sub(), compile(). Синтаксис регулярных выражений: символьные классы, квантификаторы, группы, альтернация. Жадные и ленивые квантификаторы. Практика: извлечение email-адресов, дат, числовых данных из текстов. Применение: предобработка текстов для NLP	СЗ	ОПК-3.2, PL-1.1
		2.8	Многопоточность и асинхронность (введение)	Модуль threading: создание потоков, Thread, Lock. GIL: что это и почему ограничивает параллелизм. Модуль multiprocessing: обход GIL. Asyncio: async/await (обзор).	СЗ	ОПК-3.2, PL-1.1, PL-1.2

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				Когда что использовать: I/O-bound vs CPU-bound задачи. Связь с ML: параллельная загрузка данных, DataLoader		
		2.9	Работа с внешними API и HTTP-запросы	Библиотека requests: GET, POST, заголовки, параметры, JSON-ответы. Работа с REST API. Обработка ошибок HTTP. Аутентификация (API-ключи). Практика: запрос к публичному API (например, OpenWeatherMap или HuggingFace Inference API). Парсинг JSON-ответа	СЗ	ОПК-3.2, ОПК-2.2, PL-1.1
		2.10	Логирование и отладка	Модуль logging: уровни (DEBUG, INFO, WARNING, ERROR, CRITICAL), форматирование, handlers. Отладка: pdb, breakpoints в IDE. Профилирование: cProfile, line_profiler (обзор). Практика: добавление логирования в учебный ML-пайплайн	СЗ	ОПК-3.2, PL-1.2, ОПК-3.3
		2.11	Документирование кода и стандарты качества	Docstrings: форматы (Google, NumPy, reStructuredText). Генерация документации: Sphinx (обзор). Type hints и их проверка: mypy. Линтеры: flake8, pylint, ruff. Форматирование: black, isort. Pre-commit hooks. Связь с командной разработкой: единые стандарты кода	СЗ	PL-1.2, SS-2.1, ОПК-3.3
		2.12	Проект: создание Python-пакета с ООП-архитектурой	Мини-проект: создание Python-пакета для обработки данных с ООП-архитектурой. Структура пакета: init.py, модули, классы. Документирование: docstrings, type hints. Публикация на PyPI (обзор). Оформление README.md. Работа в парах: code review партнёра	СЗ	PL-1.1, PL-1.2, SS-2.1, ОПК-3.3
Раздел 3	Тестирование и обеспечение качества Python-кода	3.1	Введение в тестирование. Модуль unittest	Зачем тестировать код. Виды тестирования: unit, интеграционное, end-to-end. Пирамида тестирования. Модуль unittest: TestCase, assertEqual, assertTrue, setUp, tearDown. Запуск тестов. Практика: написание unit-тестов для функций обработки данных	СЗ	ОПК-3.2, PL-1.2, ПК-3.2
		3.2	Фреймворк pytest: основы	Установка и запуск pytest. Обнаружение тестов. Утверждения (assert). Фикстуры: @pytest.fixture. Параметризация: @pytest.mark.parametrize. Пропуск тестов: @pytest.mark.skip. Практика: переписывание unittest-тестов на pytest	СЗ	ОПК-3.2, PL-1.2, ПК-3.2
		3.3	Тестирование: моки, покрытие, TDD	Модуль unittest.mock: Mock, MagicMock, patch. Мокирование внешних зависимостей (файлы, API). Измерение покрытия: pytest-cov. Интерпретация отчёта о	СЗ	ОПК-3.2, PL-1.2, ПК-3.2

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				покрытии. TDD (Test-Driven Development): цикл Red-Green-Refactor. Практика: TDD для функции валидации данных		
		3.4	Тестирование ML-компонентов	Особенности тестирования ML-кода: недетерминизм, зависимость от данных. Стратегии: фиксация seed, тестирование формы выхода, проверка диапазона значений, smoke-тесты модели. Тестирование пайплайнов обработки данных: проверка типов, пропусков, корректности преобразований	СЗ	ПК-3.2, PL-1.2, ОПК-3.3
		3.5	Проектирование тестов: план тестирования для ИИ-проекта	Составление плана тестирования для учебного ИИ-проекта: определение тестируемых компонентов, уровней тестирования, критериев приёмки. Шаблон плана тестирования. Практика: написание плана тестирования для пайплайна «загрузка данных → предобработка → обучение модели → предсказание»	СЗ	ПК-3.2, SS-2.1, ОПК-3.3
		3.6	Работа с Git: основы версионного контроля	Зачем нужен Git. Основные команды: init, add, commit, status, log, diff. Файл .gitignore. Ветвление: branch, checkout, merge. Конфликты и их разрешение. Практика: инициализация репозитория, создание веток для разных функций, слияние	СЗ	PL-1.2, SS-2.1, ОПК-2.2
		3.7	Работа с Git: удалённые репозитории и командная работа	Удалённые репозитории: GitHub/GitLab. Команды: clone, push, pull, fetch. Форк и Pull Request. Code review: практика и чек-лист. Issues и Projects для управления задачами. Практика: работа в парах — создание PR, проведение code review, слияние	СЗ	SS-2.1, PL-1.2, ОПК-2.2
		3.8	Поиск технической информации и работа с документацией	Стратегии поиска решений: официальная документация Python, Stack Overflow, GitHub Issues, научные статьи. Оценка достоверности источников. Чтение исходного кода библиотек. Работа с REPL и help(). Практика: решение нестандартной задачи с использованием документации и ресурсов сообщества	СЗ	ОПК-2.2, УК-12.1, SS-2.1
		3.9	Информационная грамотность: ИИ-ассистенты и верификация кода	Использование ИИ-ассистентов (GitHub Copilot, ChatGPT) для генерации кода. Критическая оценка сгенерированного кода: корректность, безопасность, читаемость. Верификация через тестирование. Этические аспекты: авторство, лицензирование. Практика: генерация функции	СЗ	УК-12.1, ПК-3.2, SS-2.1

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				ИИ-ассистентом → написание тестов → исправление ошибок		
		3.10	Контрольная работа: ООП, тестирование, Git	Самостоятельная работа: реализация класса для обработки данных с ООП-архитектурой, написание тестов (pytest), оформление кода (type hints, docstrings), публикация в Git-репозитории с README и .gitignore	СЗ	ОПК-3.2, ОПК-3.3, PL-1.1, PL-1.2, ПК-3.2
Раздел 4	Научные вычисления: NumPy и Pandas	4.1	NumPy: создание массивов и базовые операции	Установка и импорт NumPy. Создание массивов: np.array(), np.zeros(), np.ones(), np.arange(), np.linspace(), np.random. Атрибуты: shape, dtype, ndim, size. Индексация и срезы. Копирование: view vs copy. Массив NumPy как основная структура данных в ML	СЗ	ОПК-3.2, PL-1.1
		4.2	NumPy: векторизация и Broadcasting	Поэлементные операции: арифметика, сравнение, логические. Универсальные функции (ufunc): np.exp(), np.log(), np.sqrt(). Broadcasting: правила, примеры, типичные ошибки. Оси (axis) и агрегации: sum, mean, std, min, max по осям. Векторизация vs циклы: замер производительности	СЗ	ОПК-3.2, PL-1.1
		4.3	NumPy: линейная алгебра и работа с матрицами	Модуль np.linalg: dot, matmul (@), det, inv, eig, svd, norm, solve. Транспонирование, изменение формы: reshape, flatten, ravel, transpose. Стекирование: np.hstack, np.vstack, np.concatenate. Практика: реализация метода наименьших квадратов через NumPy	СЗ	ОПК-3.2, PL-1.1
		4.4	NumPy: случайные числа и производительность	Модуль np.random: seed, rand, randn, randint, choice, shuffle, normal, uniform. Генераторы (np.random.Generator). Замер производительности: %timeit в Jupyter. Сравнение NumPy и чистого Python. Практика: генерация синтетического датасета для задачи регрессии	СЗ	ОПК-3.2, PL-1.1
		4.5	Pandas: Series и DataFrame — основные понятия	Установка Pandas. Series: создание, индексация, операции. DataFrame: создание из словарей, списков, CSV. Атрибуты: shape, columns, index, dtypes, info(), describe(). Загрузка данных: read_csv(), read_json(), read_excel(). DataFrame как таблица признаков ML-модели	СЗ	ОПК-3.2, PL-1.1
		4.6	Pandas: индексация, фильтрация, выбор данных	Индексация: loc[], iloc[], []. Булева фильтрация. Метод query(). Выбор столбцов. Условные выборки. Сортировка: sort_values(), sort_index(). Практика: фильтрация датасета	СЗ	ОПК-3.2, PL-1.1

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				по условиям (например, выбор записей с определёнными значениями признаков)		
		4.7	Pandas: обработка пропусков и преобразование типов	Обнаружение пропусков: isnull(), notnull(), info(). Обработка: dropna(), fillna(). Стратегии заполнения: среднее, медиана, мода, интерполяция. Преобразование типов: astype(), pd.to_numeric(), pd.to_datetime(). Кодирование категориальных признаков: get_dummies(), LabelEncoder (обзор)	СЗ	ОПК-3.2, PL-1.1
		4.8	Pandas: группировка, агрегация и сводные таблицы	Группировка: groupby(). Агрегации: agg(), transform(). Множественные агрегации. Сводные таблицы: pivot_table(), crosstab(). Практика: анализ датасета продаж — выручка по категориям, средний чек по регионам (кейс, связанный с X5 Group)	СЗ	ОПК-3.2, PL-1.1, ОПК-2.2
		4.9	Pandas: объединение и реструктуризация данных	Объединение: merge(), join(), concat(). Типы соединений: inner, outer, left, right. Реструктуризация: melt(), stack(), unstack(). Практика: объединение нескольких таблиц датасета (признаки + метки + метаданные) в единый DataFrame	СЗ	ОПК-3.2, PL-1.1
		4.10	Pandas: apply, vectorize, pipe — продвинутая обработка	Метод apply(): по строкам и столбцам. np.vectorize(). Метод pipe() для построения цепочек преобразований. Оптимизация: избегание apply в пользу векторизованных операций. Практика: построение пайплайна предобработки данных для ML-модели	СЗ	ОПК-3.2, PL-1.1, PL-1.2
		4.11	Практикум: EDA (Exploratory Data Analysis) на реальном датасете	Загрузка реального датасета (Kaggle, UCI ML Repository). Полный цикл EDA: info(), describe(), проверка пропусков, распределения признаков, корреляции. Формулирование гипотез о данных. Документирование результатов в Jupyter Notebook	СЗ	PL-1.1, ОПК-2.2, ОПК-3.3
		4.12	Практикум: NumPy vs Pandas — выбор инструмента	Сравнение NumPy и Pandas: когда что использовать. Производительность: замеры на одинаковых операциях. Конвертация: DataFrame.to_numpy(), pd.DataFrame(np.array). Практика: реализация одного пайплайна обработки данных на NumPy и на Pandas, сравнение	СЗ	PL-1.1, PL-1.2, ОПК-3.2
Раздел 5	Визуализация данных и	5.1	Matplotlib: основы визуализации	Архитектура Matplotlib: Figure, Axes, Artist. Базовые	СЗ	ОПК-3.2,

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
	прототипирование ИИ-приложений			графики: plot(), scatter(), bar(), hist(). Настройка: заголовок, подписи осей, легенда, сетка, цвета. Множественные subplot. Сохранение: savefig(). Практика: визуализация распределений признаков датасета		PL-1.1
		5.2	Matplotlib: продвинутые графики и стилизация	Тепловые карты (imshow), контурные графики (contour, contourf). Гистограммы с KDE. Boxplot и violin plot. Стили: plt.style.use(). Аннотации: annotate(), text(). Практика: визуализация корреляционной матрицы и распределения классов	СЗ	ОПК-3.2, PL-1.1
		5.3	Seaborn: статистическая визуализация	Библиотека Seaborn: философия и основные типы графиков. Relational: scatterplot(), lineplot(). Distributional: histplot(), kdeplot(), boxplot(). Categorical: countplot(), barplot(), violinplot(). Heatmap. FacetGrid. Практика: визуализация многомерного датасета	СЗ	ОПК-3.2, PL-1.1, ОПК-2.2
		5.4	Plotly: интерактивная визуализация	Библиотека Plotly Express: scatter(), line(), bar(), histogram(), box(), heatmap(). Интерактивность: hover, zoom, pan. 3D-графики. Анимации. Plotly в Jupyter Notebook. Практика: создание интерактивного дашборда для визуализации результатов EDA	СЗ	ОПК-3.2, PL-1.1, ПК-2.2
		5.5	Jupyter Notebook как инструмент прототипирования	Jupyter Notebook: ячейки кода, Markdown, магические команды (%timeit, %matplotlib, %%writefile). Jupyter Lab: расширенные возможности. Структура аналитического ноутбука: заголовок, описание задачи, EDA, моделирование, выводы. Практика: оформление ноутбука для воспроизводимого эксперимента	СЗ	ОПК-3.2, ПК-2.2, ОПК-2.2
		5.6	Streamlit: создание интерактивных прототипов	Библиотека Streamlit: установка, запуск. Виджеты: st.slider(), st.selectbox(), st.text_input(), st.file_uploader(). Вывод: st.write(), st.dataframe(), st.plotly_chart(), st.pyplot(). Layout: st.columns(), st.sidebar(). Практика: создание прототипа приложения для загрузки и визуализации датасета	СЗ	ПК-2.2, ОПК-3.2, PL-1.1
		5.7	Streamlit: прототип ИИ-приложения	Создание прототипа ИИ-приложения на Streamlit: загрузка данных → предобработка → обучение простой модели (sklearn) → визуализация результатов → интерактивное предсказание. Деплой на Streamlit Cloud (обзор). Связь с	СЗ	ПК-2.2, ОПК-3.3, PL-1.1

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				валидацией прототипа с заинтересованными сторонами		
		5.8	FastAPI: создание REST API для ML-модели (введение)	Библиотека FastAPI: установка, маршруты, GET/POST. Модели данных: Pydantic (BaseModel). Автодокументация: Swagger UI. Практика: создание API-эндпоинта для предсказания обученной модели. Тестирование API: requests, httpx	СЗ	ПК-2.2, ОПК-3.2, PL-1.1
		5.9	Проект: прототип ИИ-решения с визуализацией	Групповой мини-проект: разработка прототипа ИИ-решения. Этапы: загрузка и анализ данных (Pandas), визуализация (Matplotlib/Seaborn/Plotly), обучение модели (sklearn), создание интерфейса (Streamlit или Jupyter). Презентация прототипа. Работа в команде: распределение задач, Git	СЗ	ПК-2.2, SS-2.1, ОПК-3.3, PL-1.1
		5.10	Защита проекта и peer review	Презентация проектов командами. Демонстрация работающего прототипа. Peer review: оценка кода, архитектуры, документации, тестов другой команды по чек-листу. Обратная связь. Рефлексия: что можно улучшить	СЗ	SS-2.1, ПК-2.2, ПК-3.2
		5.11	Практикум: визуализация для ML — лучшие практики	Типовые визуализации в ML: кривая обучения, матрица ошибок (confusion matrix), ROC-кривая, распределение предсказаний, важность признаков. Реализация на Matplotlib/Seaborn. Связь с оценкой качества моделей и коммуникацией результатов стейкхолдерам	СЗ	PL-1.1, ОПК-3.2, ОПК-2.2
		5.12	Практикум: Jupyter → отчёт. Экспорт и воспроизводимость	Экспорт Jupyter Notebook: HTML, PDF, Markdown. nbconvert. Papermill: параметризация ноутбуков. Воспроизводимость: фиксация seed, requirements.txt, структура проекта. Cookiecutter Data Science: шаблон проекта. Практика: оформление воспроизводимого аналитического отчёта	СЗ	ОПК-3.3, PL-1.2, ОПК-2.2
Раздел 6	Интеграция, автоматизация и проектная работа	6.1	Структура ML-проекта: организация кода	Структура проекта: src/, tests/, data/, notebooks/, configs/, README.md, requirements.txt, Makefile. Разделение кода: модули обработки данных, обучения, предсказания, визуализации. Принцип единственной ответственности. Cookiecutter Data Science. Практика: создание шаблона проекта	СЗ	PL-1.2, ОПК-3.3, ПК-2.2
		6.2	Конфигурации и управление параметрами	Файлы конфигурации: YAML, JSON, TOML. Библиотеки: PyYAML, configparser, python-dotenv. Хранение	СЗ	PL-1.2, ОПК-3.2,

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				гиперпараметров и путей к данным. Переменные окружения: os.environ, .env файлы. Практика: вынесение параметров ML-пайплайна в конфигурационный файл		ОПК-3.3
		6.3	Автоматизация: Makefile и скрипты	Makefile: цели, зависимости, переменные. Типовые цели для ML-проекта: make data, make train, make test, make lint. Скрипты запуска: argparse для CLI. Практика: создание Makefile для автоматизации пайплайна обработки данных и обучения модели	СЗ	PL-1.2, ОПК-3.3, ПК-3.2
		6.4	CI/CD для Python-проектов (введение)	Непрерывная интеграция: GitHub Actions. Файл workflow (.github/workflows/). Типовой пайплайн: установка зависимостей → линтинг → тестирование → отчёт о покрытии. Практика: настройка GitHub Actions для учебного проекта (запуск pytest при каждом push)	СЗ	ПК-3.2, PL-1.2, ОПК-3.3
		6.5	Работа с базами данных из Python	Модуль sqlite3: подключение, создание таблиц, INSERT, SELECT, UPDATE, DELETE. ORM SQLAlchemy (обзор). Практика: сохранение результатов экспериментов (метрики, гиперпараметры) в SQLite-базу. Запросы для анализа: «лучшая модель по accuracy»	СЗ	ОПК-3.2, PL-1.1, ОПК-2.2
		6.6	Веб-скрейпинг и сбор данных	Библиотеки: requests + BeautifulSoup, Selenium (обзор). Этика веб-скрейпинга: robots.txt, rate limiting. Практика: сбор данных с веб-страницы (например, заголовки новостей), сохранение в CSV. Связь со сбором данных для обучения ML-моделей	СЗ	ОПК-3.2, ОПК-2.2, УК-12.1
		6.7	Обзор экосистемы Python для ИИ	Обзор ключевых библиотек: scikit-learn (ML), PyTorch/TensorFlow (DL), Hugging Face (NLP/LLM), OpenCV (CV), Optuna (оптимизация), MLflow (трекинг), DVC (версионирование данных). Как выбрать библиотеку для задачи. Практика: решение простой задачи классификации на scikit-learn от загрузки до оценки	СЗ	PL-1.1, ОПК-2.2, ОПК-3.2
		6.8	Итоговый проект: постановка задачи и планирование	Постановка задачи итогового проекта: выбор датасета, формулирование задачи, планирование архитектуры решения. Составление плана тестирования. Распределение ролей в команде. Создание репозитория, настройка CI, структура проекта. Начало реализации	СЗ	ПК-2.2, ПК-3.2, SS-2.1, ОПК-3.3
		6.9	Итоговый проект: реализация и	Реализация проекта в командах: загрузка и обработка	СЗ	PL-1.1,

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
			тестирование	данных, обучение модели, визуализация результатов, создание прототипа (Streamlit/Jupyter). Написание тестов. Code review внутри команды. Подготовка к защите		PL-1.2, ОПК-3.3, ПК-2.2, ПК-3.2, SS-2.1
		6.10	Итоговый проект: защита и рефлексия	Защита итоговых проектов: демонстрация работающего решения, объяснение архитектуры, результаты тестирования. Peer review: оценка проектов других команд. Рефлексия: что удалось, что можно улучшить, какие навыки развиты. Рекомендации по дальнейшему изучению Python для ИИ	СЗ	SS-2.1, ПК-2.2, ПК-3.2, ОПК-3.3

* - заполняется только по **ОЧНОЙ** форме обучения: *ЛК – лекции; ЛР – лабораторные работы; СЗ – практические/семинарские занятия.*

6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Таблица 6.1. Материально-техническое обеспечение дисциплины

Тип аудитории	Оснащение аудитории	Специализированное учебное/лабораторное оборудование, ПО и материалы для освоения дисциплины (при необходимости)
Семинарская	Аудитория для проведения занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, оснащенная комплектом специализированной мебели и техническими средствами мультимедиа презентаций.	Персональные компьютеры, необходимое ПО
Для самостоятельной работы	Аудитория для самостоятельной работы обучающихся (может использоваться для проведения семинарских занятий и консультаций), оснащенная комплектом специализированной мебели и компьютерами с доступом в ЭИОС.	Персональные компьютеры, необходимое ПО

* - аудитория для самостоятельной работы обучающихся указывается **ОБЯЗАТЕЛЬНО**!

7. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

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. Гуриков Сергей Ростиславович. Основы алгоритмизации и программирования на Python. учебное пособие [Электронный ресурс]. - М.: ИНФРА-М, 2023. 341 с. ISBN 978-5-16-016971-5 URL: https://mega.rudn.ru/MegaPro/UserEntry?Action=Link_FindDoc&id=508886&idb=0

3. Практикум по программированию на языке Python. Учебное пособие. Для студентов, обучающихся по направлениям 01.03.02 «Прикладная математика и информатика», 09.03.03 «Прикладная информатика», 10.03.01 «Информационная безопасность», 38.03.05 «Бизнес-информатика», (программа подготовки бакалавра). — М.: Финансовый университет, департамент анализа данных и машинного обучения, 2023. — 320 с

Дополнительная литература:

1. Жуков Роман Александрович. Язык программирования Python. практикум [Электронный ресурс]. - М.: ИНФРА-М, 2023. 215 с. ISBN 978-5-16-016971-2 URL: https://mega.rudn.ru/MegaPro/UserEntry?Action=Link_FindDoc&id=508831&idb=0

Ресурсы информационно-телекоммуникационной сети «Интернет»:

1. ЭБС РУДН и сторонние ЭБС, к которым студенты университета имеют доступ на основании заключенных договоров

- Электронно-библиотечная система РУДН – ЭБС РУДН

https://mega.rudn.ru/MegaPro/Web

- ЭБС «Университетская библиотека онлайн» http://www.biblioclub.ru
- ЭБС «Юрайт» http://www.biblio-online.ru
- ЭБС «Консультант студента» www.studentlibrary.ru
- ЭБС «Знаниум» https://znanium.ru/

2. Базы данных и поисковые системы

- Sage https://journals.sagepub.com/
- Springer Nature Link https://link.springer.com/
- Wiley Journal Database https://onlinelibrary.wiley.com/
- Наукометрическая база данных Lens.org https://www.lens.org

Учебно-методические материалы для самостоятельной работы обучающихся при освоении дисциплины/модуля:*

* - все учебно-методические материалы для самостоятельной работы обучающихся размещаются в соответствии с действующим порядком на странице дисциплины **в ТУИС**!