Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Гриб Владислав Валерьевич
Должность: Ректор
Дата подписания: 06.11.2026 17:34:33
Уникальный программный ключ:
637517d24e103c3db032acf37e839d[illegible]

Образовательное частное учреждение высшего образования
«МОСКОВСКИЙ УНИВЕРСИТЕТ ИМЕНИ А.С. ГРИБОЕДОВА»
(ИМПЭ им. А.С. Грибоедова)
ФАКУЛЬТЕТ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
(МЕЖДУНАРОДНЫЙ ДИПЛОМАТИЧЕСКИЙ КОЛЛЕДЖ)

УТВЕРЖДАЮ
Директор Международного
дипломатического колледжа
_____ Н.И. Платонова
«16» февраля 2026г.

РАБОЧАЯ ПРОГРАММА ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

Разработка кода для обучения искусственного интеллекта

Специальность
09.02.13 Интеграция решений с применением технологий искусственного интеллекта

Квалификация выпускника:
Специалист по работе с искусственным интеллектом

на базе основного общего образования

Форма обучения: очная

Москва

Рабочая программа профессионального модуля «Разработка кода для обучения искусственного интеллекта». Специальность 09.02.13 Интеграция решений с применением технологий искусственного интеллекта /– М.: ИМПЭ им. А. С. Грибоедова

Рабочая программа профессионального модуля составлена на основании Федерального государственного образовательного стандарта 09.02.13 Интеграция решений с применением технологий искусственного интеллекта", утвержденного приказом Минпросвещения России от 24.12.2024 N 1025 "Об утверждении федерального государственного образовательного стандарта среднего профессионального образования по специальности 09.02.13 Интеграция решений с применением технологий искусственного интеллекта"

Организация-разработчик:	Образовательное частное учреждение высшего образования «Московский университет имени А. С. Грибоедова»
Ответственный рецензент:	Е.В. Михалёва, к. ф.-м. н. исполнительный директор института информационных систем и инженерно-компьютерных технологий
	(Ф.И.О., уч. степень, уч. звание, должность)

Рабочая программа профессионального модуля рассмотрена и одобрена на заседании Международного дипломатического колледжа «16» февраля 2026 г., протокол №6

Директор Международного
дипломатического колледжа _____/Н.И. Платонова

Согласовано от Библиотеки _____/О. Е. Стёпкина

1. ОБЩАЯ ХАРАКТЕРИСТИКА РАБОЧЕЙ ПРОГРАММЫ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

1.1. Цель и планируемые результаты освоения профессионального модуля

В результате изучения профессионального модуля обучающийся должен освоить основной вид деятельности «Разработка кода для обучения искусственного интеллекта» и соответствующие ему общие компетенции и профессиональные компетенции:

1.1.1. Перечень общих компетенций

Код	Наименование общих компетенций
ОК 01.	Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам
ОК 02.	Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности
ОК 05.	Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста

1.1.2. Перечень профессиональных компетенций

Код	Наименование видов деятельности и профессиональных компетенций
ВД 1	Разработка кода для обучения искусственного интеллекта
ПК 1.1.	Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием
ПК 1.2.	Разрабатывать программные модули в соответствии с техническим заданием
ПК 1.3.	Оформлять программный код в соответствии с техническим заданием
ПК 1.4.	Использовать систему контроля версий программного кода с учетом обеспечения возможности организации групповой разработки
ПК 1.5.	Выполнять отладку программных модулей с использованием специализированных программных средств
ПК 1.6.	Выполнять тестирование программного кода
ПК 1.7.	Составлять тестовые сценарии

1.1.3. В результате освоения профессионального модуля обучающийся должен:

Иметь практический опыт	- разработки, оптимизации и оценка сложности алгоритмов для ИИпрограмм; - использования библиотек и инструментов для работы с алгоритмами и данными; - применения структур данных (деревья, графы, списки) для реализации алгоритмов; - разработки модульных ИИ-систем, соответствующих требованиям производительности и безопасности; - внедрения разработанных ИИ-модулей в комплексные программные системы;

	- оптимизации кода и работы с интерфейсами для взаимодействия между модулями; - оформления, документирования и структурирования кода для последующей поддержки; - использования инструментов статического анализа кода для выявления ошибок и улучшения качества; - работы с системами документирования кода; - управления проектами с использованием системы контроля версий тогда можно просто для организации командной работы; - разрешения конфликтов при слиянии веток и использования pull request для рецензирования кода; - настройки процессов CI/CD для автоматического тестирования и развертывания кода; - отладки программных модулей с использованием пошаговой проверки; - применения методов логирования и профилирования производительности; - использования специальных средств для отладки многопоточных программ; - выполнения статического тестирования программного кода на предмет выявления ошибок/дефектов алгоритмов, в том числе – на наличие обработки исключений; - выполнения тестирования программных модулей в соответствии с тестпланом; - генерирования тестовых данных; - выполнения интеграционного тестирования в соответствии с заданием; - выполнения регрессионного тестирования в соответствии с заданием; - разработки тестовых сценариев в соответствии с тестовым планом (тестирование производительности, надежности, UI-тестирование), в том числе с применение средств автоматизации проектирования; - разработки тестовых пакетов и заданий на выполнение тестирования; - оценки тестовых данных на предмет покрытия строк и

	покрытия ветвей, выполнять валидацию данных.

уметь	- анализировать технические задания и выявлять требования к алгоритмам; - применять методы алгоритмизации для решения задач программирования; - разрабатывать оптимальные алгоритмы для решения задач в области ИИ; - реализовывать программные модули на основе требований технического задания; - соблюдать принципы чистого кода (Clean Code); - использовать стандартные библиотеки и фреймворки для ускорения разработки; - оформлять код в соответствии с принятыми стандартами и требованиями; - документировать разработанный программный код; - соблюдать соглашения о наименованиях переменных, функций и классов (например, PEP8 для Python); - работать с системами контроля версий для управления проектами; - организовывать совместную работу над проектом через ветки разработки и слияние изменений; - разрешать конфликты при слиянии кода;
	- использовать инструменты для отладки программного кода; - идентифицировать и исправлять ошибки в программе; - применять методы логирования для анализа выполнения программ; - проводить различные виды тестирования (юнит-тестирование, интеграционное тестирование); - выполнять настройки окружения т подготовку тестовых данных; - фиксировать результаты выполнения тестов и подготавливать отчеты о результатах тестов; - определять уровень критичности дефектов; - разрабатывать автоматизированные тесты для тестирования модулей и/или отдельных функций; - восстанавливать окружение и тесты после сбоя; - проектировать тестовые сценарии на основе тестовых планов; - разрабатывать тестовые пакеты и задания на выполнение тестирования; - использовать шаблоны для написания тест-кейсов; - оценивать риски при отборе тестов для регрессионного тестирования; - оценивать тесты на соответствие целям тестирования.

знать	- основные методы и подходы к построению алгоритмов (типовые поисковые алгоритмы); - принципы эффективной обработки данных; - языки программирования, применяемые для разработки алгоритмов; - принципы модульного программирования; - языки программирования для разработки модулей; - стандартные фреймворки и библиотеки для работы с ИИ; - основные принципы чистого кода; - стандарты и практики документирования программного обеспечения; - инструменты для автоматической проверки качества кода; - принципы работы распределенных систем контроля версий; - основные команды и операции в системе контроля версий; - методы разрешения конфликтов в ходе групповой разработки; - принципы работы отладчиков и логирования; - способы выявления ошибок в программе (отладка по шагам, точки останова); - инструменты для отладки кода; - принципы работы отладчиков и логирования; - технику выполнения тестовых прогонов; - методы и подходы к написанию тестов; - инструменты для тестирования программного кода; - цели, задачи и виды тестирования; - жизненный цикл дефекта; - понятие стратегии тестирования; - основы тест-дизайна: тестовый сценарий, тестовый пакет, чек-лист, основные шаблоны; - основные инструменты проектирования тестов.

2. СТРУКТУРА И СОДЕРЖАНИЕ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

2.1. Структура профессионального модуля

Структура профессионального модуля очной формы

Коды профессиональных общих компетенций	Наименования разделов профессионального модуля	Суммарный объем нагрузки, час.	Объем профессионального модуля, ак. час.							
			Работа обучающихся во взаимодействии с преподавателем					Самостоятельная работа	Консультации	Промежуточная аттестация
			Обучение по МДК			Практики				
			Всего	В том числе		Учебная	Производственная			
				Лабораторных и практических занятий	Курсовых работ (проектов)					
1	2	3	4	5	6	7	8	9	10	11
ОК 01, 02, 05, ПК 1.1. - 1.7.	МДК.01.01 Разработка программных модулей в системах искусственного интеллекта	164	80	56				18	4	6
	МДК.01.02 Разработка мобильных приложений с поддержкой искусственного интеллекта	164	66	58	20			8	2	10
	МДК.01.03 Тестирование программных модулей	144	76	42				20	-	6
	Учебная практика	72				72				

	Производственная практика	144					144			
	Экзамен по модулю	8								8
Всего:		**696**	**222**	**156**	**20**	**72**	**144**	**46**	**6**	**30**

2.2. Тематический план и содержание профессионального модуля (ПМ)

Наименование разделов и тем профессионального модуля (ПМ), междисциплинарных курсов (МДК)	**Содержание учебного материала учебного материала, лабораторные работы и практические занятия, курсовая работа (проект)**	**Объем в часах очной формы**
1	**2**	**3**
МДК.01.01 Разработка программных модулей в системах искусственного интеллекта		**164**
Модуль 1. Основы разработки для ИИ		**36**
Тема 1.1. Введение в ИИ и инструментарий.	**Содержание учебного материала** Понятие Искусственного Интеллекта, его история и основные направления. Ключевые задачи ИИ. Классификация, регрессия и кластеризация. Обзор «золотого стандарта» библиотек для ИИ NumPy, Pandas, Matplotlib и Scikit-learn. Создание и настройка профессионального рабочего окружения. Принципы интерактивной разработки в среде Jupyter Notebook.	**10**
	В том числе практических занятий и лабораторных работ Установка и конфигурация рабочего окружения Python для задач ИИ. Освоение среды Jupyter Notebook и документирование с помощью Markdown. Реализация базовых алгоритмов на Python в среде Jupyter.	**6**

	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Тема 1.2. Основы обработки данных. Библиотеки NumPy и Pandas.	**Содержание учебного материала** Векторные вычисления в Python и основной объект библиотеки NumPy. Методы индексации, срезов и преобразования формы массивов NumPy. Основные структуры данных Pandas и операции чтения и записи табличных данных. Принципы выборки и фильтрации данных в Pandas с помощью loc, iloc и булевых масок. Визуализация	**10**
	данных с Matplotlib и Seaborn для исследовательского анализа (EDA) и выявления проблем дисбаланса классов и аномалий.	
	В том числе практических занятий и лабораторных работ Выполнение векторных операций над данными с использованием NumPy. Загрузка, исследование и фильтрация табличных данных с помощью Pandas. Проведение группировки и агрегации данных для получения базовых инсайтов.	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Модуль 2. Создание и оценка моделей		**36**

Тема 2.1. Классические алгоритмы Machine Learning.	**Содержание учебного материала** Определение базовой модели (Baseline), инженерия признаков (Feature Engineering) и процесс машинного обучения. Линейные модели для решения задач регрессии и классификации. Принцип работы метода k-ближайших соседей, основанного на метриках расстояния. Принципы построения, визуализации и работы алгоритма дерева решений. Алгоритм кластеризации K-Means как основной представитель обучения без учителя.	**10**
	В том числе практических занятий и лабораторных работ Реализация линейных моделей для решения задач регрессии и классификации. Сравнение алгоритмов KNN и Дерева решений на задаче классификации. Решение задачи кластеризации с использованием алгоритма K-Means.	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы.	**2**

	Выполнение индивидуальных заданий.	
Тема 2.2. Обучение и метрики качества.	**Содержание учебного материала** Подготовка данных к обучению и разделение выборки на обучающую и тестовую. Проблема переобучения, недообучения моделей и понятие валидации для их оценки. Матрица ошибок как основа для расчета метрик Accuracy, Precision и Recall. Компромисс Precision и Recall, F1-мера и ROC-кривая как инструменты оценки. Выбор ключевой метрики под бизнес-задачу и анализ стоимости ошибок.	**10**

	В том числе практических занятий и лабораторных работ Применение процедуры Train-Test Split и кросс-валидации для обучения моделей. Расчет и анализ метрик качества для задачи бинарной классификации. Сравнительный анализ производительности различных моделей на одной задаче.	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Модуль 3. Профессиональная разработка модуля		**36**
Тема 3.1. Продвинутые методы и подготовка данных.	**Содержание учебного материала** Ансамблевые методы, основанные на бэггинге, и алгоритм Случайный лес. Ансамблевые методы, основанные на бустинге, и алгоритм градиентного бустинга. Необходимость и основные методы масштабирования числовых признаков. Способы обработки категориальных признаков, включая Label и One-Hot кодирование. Предотвращение утечек данных (Data Leakage) и правильная валидация.	**10**
	В том числе практических занятий и лабораторных работ Сравнительный анализ производительности ансамблевых методов. Разработка полного конвейера предобработки данных. Подбор оптимальных гиперпараметров модели с использованием GridSearchCV.	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к	**2**

	параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	
Тема 3.2. **Качество кода и модульное программирование.**	**Содержание учебного материала** Принципы "Чистого кода" и их применение в задачах искусственного интеллекта. Декомпозиция и рефакторинг кода из ячеек Jupyter в переиспользуемые функции. Основы объектно-ориентированного программирования для создания классов моделей и препроцессоров. Стандарты оформления кода PEP8 и инструменты автоматического форматирования. Документирование кода с помощью docstrings и аннотаций типов.	**10**
	В том числе практических занятий и лабораторных работ Рефакторинг кода из Jupyter Notebook в модульные Python-функции. Разработка собственного класса для модели машинного обучения. Применение инструментов форматирования и документирования к проекту.	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Модуль 4. Подготовка модуля к интеграции		**46**

Тема 4.1. Управление кодом и его проверка.	**Содержание учебного материала** Основы систем контроля версий и базовый рабочий цикл локального репозитория Git. Работа с удаленными репозиториями, ветвление и слияние как основа командной разработки. Введение в тестирование программного обеспечения и концепция юниттестов. Фреймворк Pytest как основной инструмент для написания и запуска юниттестов. Продвинутые возможности Pytest, включая фикстуры и параметризацию тестов.	**10**
	В том числе практических занятий и лабораторных работ Инициализация локального Git-репозитория для проекта.	**12**

	Создание и синхронизация с удаленным репозиторием на GitHub. Имитация рабочего процесса с использованием веток. Написание и запуск первого юнит-теста для функции предобработки. Использование фикстур для подготовки тестовых данных. Тестирование пограничных случаев с помощью параметризации.	
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Тема 4.2. Создание интерфейса (API) для модели.	**Содержание учебного материала** Введение в API и основные принципы архитектурного стиля REST. Обзор фреймворков, автоматическая документация по контракту OpenAPI/Swagger и введение в FastAPI. Определение моделей данных и валидация с помощью библиотеки Pydantic. Интеграция ML-модели и разработка контракта API с версионированием (/v1/). Логирование, единый формат и обработка ошибок, настройка CORS и основы развертывания API.	**10**

	В том числе практических занятий и лабораторных работ Создание базового веб-приложения 'Hello, World' с помощью FastAPI. Определение структуры входных и выходных данных с использованием Pydantic. Интеграция обученной модели в API и создание эндпоинта для предсказаний. Тестирование API через интерактивную документацию и Python-клиент.	**8**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**4**
Консультация		**4**

Промежуточная аттестация в форме экзамена		**6**
МДК.01.02 Разработка мобильных приложений с поддержкой искусственного интеллекта		**164**
Модуль 1. Основы мобильной разработки на Flutter		**36**
Тема 1.1. Введение в язык Dart.	**Содержание учебного материала** Синтаксис языка Dart, его система типов, переменные и концепция Null Safety. Управляющие конструкции и синтаксис определения функций с различными параметрами. Основные коллекции Dart, их создание и методы обработки. Основы объектно-ориентированного программирования, включая классы, объекты и конструкторы. Наследование классов и обработка исключительных ситуаций в Dart.	**10**
	В том числе практических занятий и лабораторных работ Решение алгоритмических задач на языке Dart. Практика работы с коллекциями в Dart. Создание и использование собственных классов в Dart.	**6**

	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Тема 1.2. **Flutter. Концепция виджетов и первое приложение.**	**Содержание учебного материала** Архитектура фреймворка Flutter, его declarative-подход к UI и настройка рабочего окружения. Концепция дерева виджетов и фундаментальное различие между StatelessWidget и StatefulWidget. Базовые виджеты, такие как Text, Icon, Container, и стандартная структура экрана с помощью Scaffold. Структура Flutter-проекта, роль файла pubspec.yaml и точка входа в приложение. Жизненный цикл виджета, назначение метода build и функция setState для обновления UI.	**10**
	В том числе практических занятий и лабораторных работ Установка и конфигурация Flutter SDK и настройка среды разработки. Создание нового Flutter-проекта и запуск стандартного приложения на эмуляторе. Модификация стандартного приложения, изменение текста, цветов и иконок. Создание простого одноэкранного приложения с нуля.	**8**
	Самостоятельная работа	**-**

	Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	
Модуль 2. Создание пользовательского интерфейса (UI)		**36**

Тема 2.1. Построение макетов и навигация.	**Содержание учебного материала** Горизонтальное и вертикальное расположение виджетов с помощью Row и Column. Стилизация макетов и позиционирование виджетов с помощью Container и Stack. Обработка пользовательского ввода с помощью кнопок, иконок и текстовых полей. Организация навигации между экранами приложения с использованием Navigator. Передача данных между экранами при навигации для создания связанных интерфейсов.	**10**
	В том числе практических занятий и лабораторных работ Верстка статичного экрана сложной структуры с использованием Row, Column и Container. Создание интерактивной формы с полями для ввода текста и кнопками. Разработка простого многоэкранного приложения с навигацией и передачей данных.	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Тема 2.2. Отображение списков и работа с формами.	**Содержание учебного материала** Эффективное отображение длинных списков с помощью конструктора ListView.builder. Стилизация элементов списка с помощью виджета ListTile и обработка пользовательских нажатий. Создание сеточных макетов для отображения данных с помощью виджета GridView. Объединение полей ввода и управление их состоянием с помощью виджета Form. Проверка корректности вводимых	**10**

	пользователем данных с помощью валидаторов форм.	

	В том числе практических занятий и лабораторных работ Создание экрана с динамически генерируемым списком элементов. Разработка интерактивного списка с возможностью перехода на детальный экран. Создание формы регистрации пользователя с несколькими полями для ввода. Реализация валидации данных в форме и отображение сообщений об ошибках.	**8**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**-**
Модуль 3. Интеграция с ИИ-модулем		**36**
Тема 3.1. Основы сетевых запросов в Flutter.	**Содержание учебного материала** Асинхронное программирование в Dart с использованием Future, async и await. Принципы работы протокола HTTP, включая методы GET/POST и статусные коды. Отправка сетевых запросов с помощью стандартной библиотеки http. Преобразование данных из формата JSON в объекты Dart и обратно. Интеграция с API, созданным в МДК.01.01, для отправки данных и получения предсказаний.	**10**
	В том числе практических занятий и лабораторных работ Создание приложения для получения и отображения данных из публичного API. Разработка парсера для преобразования сложных JSON-ответов в Dart-объекты. Подключение мобильного приложения к собственному API модели машинного обучения.	**6**

	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы.	**2**

	Выполнение индивидуальных заданий.	
Тема 3.2. Подключение к API модели машинного обучения.	**Содержание учебного материала** Создание сервисного слоя в приложении для инкапсуляции логики API-запросов. Управление состоянием в Flutter с setState и Provider для отображения загрузки. Обработка сетевых ошибок и отображение информативных сообщений пользователю. Формирование JSON-запроса из данных пользовательской формы для отправки на сервер. Реализация полного цикла взаимодействия от пользовательского ввода до отображения предсказания.	**10**
	В том числе практических занятий и лабораторных работ Реализация отправки данных из формы на API модели машинного обучения. Интеграция индикатора загрузки для обратной связи с пользователем. Разработка механизма обработки ошибок при сетевых запросах. Отображение и форматирование полученного от модели предсказания в UI.	**8**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**-**
Модуль 4. Завершение и подготовка к выпуску		**24**

Тема 4.1. Сборка, тестирование и развертывание.	**Содержание учебного материала** Подготовка приложения к выпуску, включая настройку иконки, имени и версии. Процесс сборки релизной версии приложения и основы ручного smoke-тестирования. Основы развертывания, включая прямую установку и обзор процесса публикации в магазины.	**6**
	В том числе практических занятий и лабораторных работ Добавление иконки приложения с использованием специализированного пакета. Изменение отображаемого имени приложения для Android и iOS. Настройка версии и номера сборки приложения в файле pubspec.yaml. Создание релизной сборки приложения для Android в формате .apk. Установка и запуск собранного .apk файла на эмуляторе или реальном устройстве.	**16**

	Разработка простого чек-листа для проведения ручного smoke-тестирования. Подготовка набора скриншотов и текстового описания для страницы в магазине. Исследование требований и процесса регистрации аккаунта разработчика в Google Play.	
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**

Курсовой проект	**Содержание учебного материала** Выбор темы курсовой работы. Подбор литературы к написанию курсовой работы. Написание содержания курсовой работы. Написание введения курсовой работы. Написание теоретической части курсовой работы. Написание практической части курсовой работы. Написание заключения курсовой работы. Оформление списка литературы курсовой работы. Подготовка речи к защите курсовой работы. Подготовка компьютерной презентации к защите курсовой работы. **Тематика курсовых работ** Разработка мобильного приложения «Умный холодильник» для распознавания продуктов по фото. Создание мобильного помощника «Агроном» для определения болезней растений по фотографии листа. Разработка мобильного приложения «Определитель пород» для классификации кошек или собак по фото. Создание мобильного приложения «Счетчик калорий» для оценки блюда по фотографии. Разработка мобильного приложения «Стилист в кармане» для определения стиля одежды по фото. Создание приложения «Музыкальный критик» для определения жанра музыки по ее аудио-признакам. Разработка приложения-помощника для сортировки мусора по фотографии объекта. Проектирование мобильного приложения для определения свежести фруктов и	**20**

	овощей по фото. Разработка мобильного приложения «Оценщик авто» для предсказания рыночной стоимости автомобиля по его параметрам. Создание мобильного приложения «Анализатор отзывов» для определения тональности текста отзыва. Разработка приложения «Риэлтор-помощник» для оценки стоимости аренды или продажи квартиры по ее характеристикам. Проектирование мобильного приложения для классификации обращений в службу технической поддержки. Создание приложения для оценки кредитного скоринга клиента по его анкетным данным. Разработка мобильного помощника для предсказания вероятности ухода клиента. Проектирование приложения для распознавания номеров автомобилей на фотографиях. Разработка мобильного приложения для классификации товаров по их текстовому описанию. Разработка мобильного приложения «Полиглот» для определения языка, на котором написан текст. Создание приложения-тренажера для распознавания рукописных букв или цифр. Разработка мобильного приложения для генерации коротких текстовых описаний к изображениям. Проектирование приложения для определения уровня сложности текста. Создание мобильного приложения для определения поэтического размера стихотворения. Разработка мобильного приложения «Дневник настроения» для определения эмоций по текстовой записи. Создание приложения-помощника для определения правильности выполнения физического упражнения по фото. Разработка мобильного приложения для распознавания объектов, опасных для детей.	

	Проектирование приложения для выявления спама или фишинговых сообщений в тексте. Создание мобильного приложения «Гид по стилю» для подбора сочетающихся предметов гардероба. Разработка приложения «Товарный сканер» для определения категории товара по его	

	штрих-коду или фото. Проектирование мобильного приложения для предсказания популярности поста в социальной сети. Создание мобильного приложения «Личный тренер» для подсчета повторений упражнений с помощью камеры. Разработка мобильного приложения для автоматической генерации хэштегов к фотографии. Проектирование приложения «Эксперт по недвижимости» для классификации типа интерьера по фото. Создание мобильного переводчика для распознавания и перевода текста с изображений в реальном времени. Разработка приложения «Детектор фейков» для определения сгенерированных компьютером лиц. Проектирование приложения «Арт-фильтр» для переноса стиля известных художников на фотографии пользователя. Создание мобильного приложения «Парковочный ассистент» для определения свободных парковочных мест по видеопотоку. Разработка приложения-помощника для распознавания дорожных знаков с помощью камеры телефона. Проектирование мобильного приложения для определения марки и модели автомобиля по фотографии. Создание приложения «Контроль качества» для обнаружения дефектов на поверхности промышленных изделий по фото. Разработка мобильного приложения «Нотариус» для распознавания и извлечения данных из паспорта. Проектирование приложения-суммаризатора для сокращения длинных новостных статей. Создание мобильного приложения «Цифровой архивариус» для классификации отсканированных документов.	

	Разработка приложения «HR-ассистент» для извлечения ключевых навыков из текста резюме. Проектирование мобильного приложения для предсказания расхода топлива автомобиля на основе маршрута. Создание приложения «Диетолог» для составления плана питания на основе	

	текстового описания предпочтений. Разработка мобильного приложения для предсказания задержек авиарейсов по данным о погоде и маршруте Проектирование приложения «Интеллектуальный будильник» для определения оптимальной фазы сна для пробуждения. Создание мобильного приложения для автоматического тегирования и организации личной фотогалереи. Разработка приложения «Эколог» для оценки уровня загрязнения воздуха по фотографии неба. Проектирование мобильного приложения для распознавания жестового языка в реальном времени. Создание приложения-помощника для подбора очков на основе формы лица пользователя по фотографии.	
Консультация		**2**
Промежуточная аттестация в форме экзамена		**10**
МДК.01.03 Тестирование программных модулей		**144**
Модуль 1. Основы тестирования ПО		**36**
Тема 1.1. Цели, задачи и виды тестирования.	**Содержание учебного материала** Обеспечение качества (QA) и роль тестировщика в команде разработки. Уровни тестирования программного обеспечения от модульного до приемочного. Виды функционального тестирования, включая Smoke, Sanity и регрессионное. Виды нефункционального тестирования, включая производительность, безопасность и удобство использования. Статическое и динамическое тестирование как два ключевых подхода к проверке качества.	**10**
	В том числе практических занятий и лабораторных работ Анализ требований и определение уровней тестирования для приложения. Классификация видов тестирования для реального веб-приложения. Декомпозиция пользовательской истории на тестовые сценарии (чек-лист).	**6**

	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к	**2**

	их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	
Тема 1.2. Жизненный цикл дефекта и основы тестдизайна.	**Содержание учебного материала** Атрибуты, серьезность, приоритет и жизненный цикл дефекта. Тестовая документация. Тест-план, тест-кейс и чек-лист. Техника тест-дизайна классы эквивалентности. Техника тест-дизайна анализ граничных значений. Таблицы решений и диаграммы состояний для тестирования сложной логики.	**10**
	В том числе практических занятий и лабораторных работ Составление и оформление отчетов о дефектах (баг-репортов) для веб-приложения. Разработка набора тест-кейсов для тестирования функциональности. Применение техник тест-дизайна для создания набора тестовых данных.	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Модуль 2. Модульное (юнит) тестирование		**36**

Тема 2.1. Принципы и инструменты юниттестирования. Фреймворк Pytest.	**Содержание учебного материала** Юнит-тестирование как основа пирамиды тестирования и его ключевые принципы. Обзор фреймворка Pytest, его установка и правила обнаружения тестов. Структура теста Arrange-Act-Assert и использование стандартных утверждений assert. Запуск тестов из командной строки, фильтрация по имени, маркеру и директории. Организация тестов в классы для логической группировки связанных проверок.	**10**
	В том числе практических занятий и лабораторных работ Написание и запуск первого юнит-теста с помощью Pytest. Разработка набора юнит-тестов для проверки работы чистой функции. Применение маркеров для группировки и выборочного запуска тестов.	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка	**2**

	к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	
Тема 2.2. Изоляция тестов (Mockобъекты) и оценка тестового покрытия.	**Содержание учебного материала** Проблема внешних зависимостей и необходимость изоляции тестов. Концепция тестовых двойников (Test Doubles). Mocks, Stubs и Fakes. Практическое применение Mock-объектов с использованием unittest.mock и patch. Тестовое покрытие как метрика полноты и качества тестирования. Генерация и анализ отчета о покрытии кода тестами с помощью pytest-cov.	**10**
	В том числе практических занятий и лабораторных работ Изоляция тестов от файловой системы с помощью Mock-объектов. Тестирование функций с внешними API-вызовами с помощью patch. Анализ и увеличение тестового покрытия проекта с помощью pytest-cov.	**6**

	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Модуль 3. Интеграционное тестирование		**36**
Тема 3.1. Ручное и автоматизированное тестирование API.	**Содержание учебного материала** Основы тестирования API и его место в цикле разработки. Инструменты для ручного тестирования API на примере Postman. Библиотека requests для автоматизации APIтестов на Python. Проверки и утверждения (assertions) в API-тестах. Структурирование автоматизированных API-тестов с помощью Pytest.	**10**
	В том числе практических занятий и лабораторных работ Исследование и ручное тестирование API с использованием Postman. Написание автоматизированного теста для GET-запроса с помощью Pytest и requests. Разработка автоматизированного теста для POST-запроса с отправкой данных.	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая	**2**

	проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	

Тема 3.2. Специфика тестирования ML-моделей. Тестирование контракта и поведения API.	**Содержание учебного материала** Особенности тестирования систем на основе машинного обучения. Тестирование контракта API проверка схем версий и кодов ответа. Тестирование инвариантов модели и проверка смысловой непротиворечивости. Тестирование данных валидация входных данных и обнаружение дрейфа Data Drift. Тестирование производительности и устойчивости ML-моделей.	**10**
	В том числе практических занятий и лабораторных работ Разработка теста контракта API с валидацией JSON-схемы ответа через Pydantic. Реализация инвариантного теста для проверки устойчивости предсказания к малым изменениям входных данных. Тестирование устойчивости API, разработка простого теста на дрейф данных (Data Drift).	**6**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**2**
Модуль 4. Системное тестирование и автоматизация		**30**
Тема 4.1. End-to-End тестирование и автоматизация UI. Selenium WebDriver.	**Содержание учебного материала** Концепция End-to-End (E2E) тестирования как имитация полного пользовательского пути. Архитектура Selenium WebDriver и принципы автоматического управления браузером. Поиск веб-элементов с помощью локаторов (CSS-селекторы, XPath). Синхронизация в тестах. Явные и неявные ожидания для стабильности тестов.	**8**
	В том числе практических занятий и лабораторных работ Автоматизация сквозного пользовательского сценария с помощью Selenium	**2**

	WebDriver.	

	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**4**
Тема 4.2. **Основы CI/CD.** **Автоматический запуск тестов с помощью GitHub Actions.**	**Содержание учебного материала** Концепции CI/CD. Непрерывная интеграция, доставка и развертывание. Обзор платформы GitHub Actions. Основные компоненты и синтаксис YAML. Создание базового рабочего процесса (workflow) для сборки проекта. Интеграция этапов статического анализа (linting) и тестирования в CI-пайплайн.	**8**
	В том числе практических занятий и лабораторных работ Настройка CI-пайплайна для автоматического запуска линтера при push в репозиторий. Конфигурация полного CI-пайплайна с автоматическим запуском всех тестов (pytest).	**4**
	Самостоятельная работа Работа над учебным материалом, ответы на контрольные вопросы. Систематическая проработка конспектов занятий, учебной и специальной литературы (по вопросам к параграфам, разделам учебных пособий, составленным преподавателем). Подготовка к практическим работам с использованием методических рекомендаций преподавателя, оформление лабораторно-практических работ, отчетов и подготовка к их защите. Подготовка рефератов, докладов, презентаций на заданные темы. Выполнение индивидуальных заданий.	**4**
Промежуточная аттестация в форме экзамена		**6**

Учебная практика **1. Содержание работ** Проектирование ИИ-решения и архитектуры приложения. 1.Изучить специфику выбранной предметной области (по вашему варианту). 2.Разработать концепцию ИИ-модели: определить решаемую задачу (классификация, регрессия); подобрать или сгенерировать набор данных (dataset); выбрать алгоритм машинного обучения. 3. Спроектировать архитектуру REST API: определить эндпоинты (например, /predict); описать структуру запроса (входные данные для модели); описать структуру ответа (предсказание модели).4 Разработать архитектуру	**72**

мобильного приложения: спроектировать пользовательские сценарии (User Flow); разработать макеты экранов (не менее 2-3); определить схему навигации между экранами. Реализация ИИ-модуля и API на Python:1.Написать скрипты для предобработки и очистки данных.2. Реализовать обучение и оценку ИИ-модели с использованием библиотек Scikit-learn, Pandas.3. Сохранить (сериализовать) обученную модель в файл.4. Разработать REST API на базе фреймворка FastAPI:реализовать эндпоинт для загрузки модели и получения предсказаний;обеспечить валидацию входных данных с помощью Pydantic.5. Написать базовые юнит-тесты для API с использованием Pytest. Разработка мобильного клиента на Flutter: 1.Создать пользовательский интерфейс в соответствии с разработанными макетами (использование Stateless и Stateful виджетов).2. Реализовать клиентскую логику: управление состоянием экранов (отображение данных, индикаторы загрузки);навигация между экранами.3. Интегрировать мобильное приложение с разработанным API: выполнять асинхронные HTTP-запросы (POST) для отправки данных на сервер; обрабатывать и отображать полученный от ИИ-модели ответ.4. Обеспечить валидацию данных на стороне клиента (например, в полях ввода форм). Тестирование, развертывание и управление проектом:1. Использовать систему контроля версий Git на протяжении всей практики.2. Провести ручное тестирование полного пользовательского сценария (End-to-End).3. Подготовить приложение к выпуску: настроить иконку и название приложения; управлять версией сборки.4. Выполнить сборку релизного .apk-файла для платформы Android и протестировать его на эмуляторе или реальном устройстве. **2. Требования к реализации бизнес-логики** При выполнении Раздела 3 (Разработка мобильного клиента) студент обязан реализовать логику, соответствующую реальным пользовательским сценариям. Необходимо выбрать и запрограммировать сценарии из трех категорий: Основная операция (AI-функция) Главное действие, ради которого создается приложение: отправка данных и получение предсказания от ИИ-модели. Примеры: Отправка фотографии цветка для определения его вида; Ввод текста отзыва для анализа его тональности. Управление состоянием (Интерактивность)Логика, обеспечивающая обратную связь с пользователем во время выполнения длительных операций, например, сетевых запросов. Примеры: Отображение индикатора загрузки во время обработки данных моделью; Показ сообщения об ошибке при отсутствии интернет-соединения. Контроль целостности (Валидация данных) Логика, защищающая систему от некорректных действий пользователя и неверных данных. Примеры: Блокировка кнопки «Отправить», пока пользователь не заполнил все	

поля; Проверка формата вводимых данных. **3. Варианты для выполнения задания** Порядок выбора задания: Выбор темы: Номер варианта определяется по первой буквой фамилии студента. Адаптация под место практики:	

На базе колледжа: Студент выполняет задание строго по описанию варианта. В профильной организации: Студент может адаптировать задание под деятельность организации, согласовав тему с руководителем практики. ВАРИАНТ 1 (Фамилии на «А») Тема: Анализатор отзывов Задание: Разработка мобильного приложения для определения тональности текста. Ключевые сущности/концепции: Модель классификации текста, API-эндпоинт /analyze_sentiment, Экран ввода текста, Экран отображения результата. ВАРИАНТ 2 (Фамилии на «Б») Тема: Ботанический справочник Задание: Создание мобильного приложения для определения вида растения по фотографии его листа. Ключевые сущности/концепции: Модель классификации изображений, API-эндпоинт /classify_plant, Экран захвата/выбора фото, Карточка с описанием растения. ВАРИАНТ 3 (Фамилии на «В») Тема: Ветеринарный помощник Задание: Разработка мобильного приложения для классификации породы кошки или собаки по фотографии. Ключевые сущности/концепции: Модель классификации изображений, API-эндпоинт /classify_pet, Экран выбора фото, Экран с результатом и описанием породы. ВАРИАНТ 4 (Фамилии на «Г») Тема: Гастрономический гид Задание: Создание приложения для распознавания блюда по фотографии и подсчета примерных калорий. Ключевые сущности/концепции: Модель распознавания объектов, API-эндпоинт /analyze_meal, Экран с камерой, Экран с результатом. ВАРИАНТ 5 (Фамилии на «Д», «Е», «Ё») Тема: Детектор спама Задание: Разработка мобильного приложения для выявления спама или фишинга в текстовых сообщениях. Ключевые сущности/концепции: Модель классификации текста, API-эндпоинт /detect_spam, Текстовое поле для ввода, Экран с вердиктом. ВАРИАНТ 6 (Фамилии на «Ж») Тема: Жанровый анализатор Задание: Создание мобильного приложения для определения музыкального жанра по загруженному	

аудиофрагменту. Ключевые сущности/концепции: Модель классификации аудио, API-эндпоинт /classify_genre, Экран выбора аудиофайла, Экран с результатом.	

ВАРИАНТ 7 (Фамилии на «З») Тема: Защитник зрения Задание: Разработка приложения, которое по фото определяет, правильно ли человек сидит за рабочим столом. Ключевые сущности/концепции: Модель Pose Estimation, API-эндпоинт /check_posture, Экран с камерой, Экран с рекомендациями. ВАРИАНТ 8 (Фамилии на «И», «Й») Тема: Идентификатор языка Задание: Создание мобильного приложения «Полиглот» для определения языка, на котором написан текст. Ключевые сущности/концепции: Модель классификации текста, API-эндпоинт /detect_language, Многострочное поле ввода, Экран с флагом и названием языка. ВАРИАНТ 9 (Фамилии на «К») Тема: Консультант по недвижимости Задание: Разработка приложения «Риэлтор-помощник» для оценки стоимости квартиры по ее параметрам. Ключевые сущности/концепции: Модель регрессии, API-эндпоинт /estimate_price, Форма с параметрами, Экран с предсказанной ценой. ВАРИАНТ 10 (Фамилии на «Л») Тема: Личный стилист Задание: Разработка мобильного приложения «Стилист в кармане» для определения стиля одежды по фото. Ключевые сущности/концепции: Модель классификации изображений, API-эндпоинт /define_style, Экран выбора фото, Экран с названием стиля. ВАРИАНТ 11 (Фамилии на «М», «Н») Тема: Мобильный оценщик Задание: Разработка мобильного приложения «Оценщик авто» для предсказания рыночной стоимости автомобиля. Ключевые сущности/концепции: Модель регрессии, API-эндпоинт /estimate_car_price, Форма ввода данных, Экран с результатом. ВАРИАНТ 12 (Фамилии на «О») Тема: Определитель эмоций Задание: Разработка мобильного приложения «Дневник настроения» для определения эмоций по текстовой записи. Ключевые сущности/концепции: Модель анализа тональности, API-эндпоинт /analyze_emotion, Текстовое поле,	

Экран с иконкой эмоции. ВАРИАНТ 13 (Фамилии на «П») Тема: Парковочный радар Задание: Создание приложения для определения свободных парковочных мест по статичному фото парковки.	

Ключевые сущности/концепции: Модель детекции объектов, API-эндпоинт /count_free_spots, Экран захвата фото, Экран с количеством свободных мест. ВАРИАНТ 14 (Фамилии на «Р») Тема: Распознаватель рукописного текста Задание: Создание приложения-тренажера для распознавания рукописных цифр, нарисованных на экране Ключевые сущности/концепции: Модель классификации изображений, API-эндпоинт /recognize_digit, Поле для рисования, Экран с результатом. ВАРИАНТ 15 (Фамилии на «С») Тема: Сортировщик мусора Задание: Разработка приложения-помощника для определения типа мусора по фотографии. Ключевые сущности/концепции: Модель классификации изображений, API-эндпоинт /classify_waste, Экран с камерой, Экран с инструкцией. ВАРИАНТ 16 (Фамилии на «Т») Тема: Технический ассистент Задание: Проектирование мобильного приложения для классификации обращений в службу поддержки по тексту. Ключевые сущности/концепции: Модель классификации текста, API-эндпоинт /route_ticket, Форма ввода обращения, Экран с назначенным отделом. ВАРИАНТ 17 (Фамилии на «У») Тема: Умный переводчик Задание: Создание мобильного переводчика для распознавания и перевода текста с изображений. Ключевые сущности/концепции: Модель OCR + API перевода, API-эндпоинт /translate_image_text, Экран захвата фото, Экран с переведенным текстом. ВАРИАНТ 18 (Фамилии на «Ф») Тема: Фитнес-тренер Задание: Создание приложения для подсчета повторений простого упражнения по видеопотоку. Ключевые сущности/концепции: Модель Pose Estimation, API-эндпоинт /count_reps, Экран с камерой и счетчиком, Экран с итоговой статистикой. ВАРИАНТ 19 (Фамилии на «Х») Тема: Хэштег-генератор Задание: Разработка мобильного приложения для автоматической генерации хэштегов к фотографии. Ключевые сущности/концепции: Модель классификации изображений, API-эндпоинт /generate_hashtags, Экран выбора фото, Экран со списком хэштегов.	

ВАРИАНТ 20 (Фамилии на «Ц», «Ч») Тема: Цифровой искусствовед	

Задание: Разработка приложения «Арт-фильтр» для переноса стиля известных художников на фотографии. Ключевые сущности/концепции: Модель Style Transfer, API-эндпоинт /apply_style, Экран выбора фото и стиля, Экран с результатом. ВАРИАНТ 21 (Фамилии на «Ш», «Щ») Тема: Штурман Задание: Разработка приложения для распознавания дорожных знаков с помощью камеры телефона. Ключевые сущности/концепции: Модель детекции объектов, API-эндпоинт /detect_sign, Экран с камерой, Уведомление с названием знака. ВАРИАНТ 22 (Фамилии на «Ы», «Э») Тема: Эколог Задание: Разработка мобильного приложения для оценки уровня загрязнения воздуха по фотографии неба. Ключевые сущности/концепции: Модель регрессии, API-эндпоинт /estimate_air_quality, Экран захвата фото, Экран с индексом качества воздуха. ВАРИАНТ 23 (Фамилии на «Ю») Тема: Юридический помощник Задание: Разработка приложения «HR-ассистент» для извлечения ключевых навыков из текста резюме. Ключевые сущности/концепции: Модель NER, API-эндпоинт /extract_skills, Экран загрузки файла, Экран со списком навыков. ВАРИАНТ 24 (Фамилии на «Я») Тема: Языковой барьер Задание: Проектирование приложения для распознавания жестового языка в реальном времени. Ключевые сущности/концепции: Модель классификации видео, API-эндпоинт /recognize_gesture, Экран с камерой, Экран с текстовым переводом жеста.	

Производственная практика **1. Содержание работ** Развертывание и конфигурирование серверной среды (Deployment):1. Подготовить среду эксплуатации: развернуть «чистую» операционную систему (виртуальная машина Linux или Docker-контейнер); настроить на ней Python-окружение и все необходимые зависимости.2. Развернуть API ИИ-модели: использовать Docker для контейнеризации FastAPI-приложения, разработанного на учебной практике; настроить Nginx как обратный прокси-сервер для перенаправления запросов к API; сконфигурировать автоматический запуск приложения как системного сервиса (например, через systemd).3.Настроить сетевой доступ и конфигурацию: сконфигурировать Nginx для обработки HTTPS-запросов (с использованием самоподписанного или реального сертификата);вынести конфигурационные данные (адреса, порты, ключи) в переменные окружения.4.Настроить мониторинг:	**144**

подключить средства мониторинга доступности API (Uptime Monitoring); проанализировать потребление ресурсов (RAM, CPU) Docker-контейнером под нагрузкой. Управление жизненным циклом модели (MLOps):1 Реализовать импорт новых данных для дообучения: подготовить массив «сырых» данных (Excel/CSV/JSON, минимум 1000 строк). Примечание: При отсутствии реальных данных, студент обязан сгенерировать синтетический набор. разработать Python-скрипт, который дообучает существующую модель на новых данных и сохраняет обновленную версию. 2.Реализовать версионирование моделей: организовать хранение разных версий обученных моделей (например, именуя файлы с датой или версией); реализовать механизм «горячей» замены модели в работающем API без его полной остановки. 3.Настроить логирование предсказаний: модифицировать API для сохранения всех входящих запросов и ответов модели в отдельный лог-файл или базу данных для последующего анализа. Автоматизация, CI/CD и нагрузочное тестирование: 1. Автоматизировать развертывание (CI/CD): расширить пайплайн в GitHub Actions, добавив шаги для сборки Docker-образа и его публикации в репозиторий (например, Docker Hub); написать скрипт (.sh), который на сервере скачивает новую версию Docker-образа и перезапускает сервис. 2.Провести нагрузочное тестирование: сгенерировать большой объем тестовых запросов (10 000+); использовать инструмент для нагрузочного тестирования (например, locust или k6) для имитации одновременного доступа к API; выявить «узкие места» и попытаться оптимизировать API. Аудит безопасности и документирование (Security & Docs):1. Провести аудит безопасности API: реализовать базовую аутентификацию (например, по статическому API-ключу в заголовке запроса); настроить логирование запросов с неверным API-ключом.2. Разработать техническую документацию: составить «Руководство DevOpsинженера» (инструкция по развертыванию, обновлению модели и мониторингу); составить «Паспорт API» (описание эндпоинтов, форматов данных, кодов ошибок и правил аутентификации).3. Подготовить и защитить отчет по практике. **2. Требования к условиям выполнения** Скриптовый подход: Операции развертывания, дообучения и тестирования должны быть оформлены в виде исполняемых скриптов и конфигурационных файлов (Dockerfile, YAML), а не выполняться вручную. Работа с данными (Важно): На предприятии: Используются реальные обезличенные данные организации. В колледже: Студент обязан сгенерировать синтетический набор данных (Synthetic/Mock Data). Для этого необходимо использовать онлайн-генераторы или написать скрипт на Python. Данные должны выглядеть реалистично. Объем данных: Для задач дообучения — не менее 1 000 записей. Для нагрузочного тестирования — не менее 10 000 запросов.	

3. Варианты для выполнения задания

Порядок выбора и выполнения:

Выбор темы: Номер варианта определяется по первой буквой фамилии студента (тема совпадает с учебной практикой).

Адаптация: На базе колледжа: Студент выполняет роль DevOps/MLOps инженера, самостоятельно генерируя данные и сценарии нагрузки. В профильной организации: Задание на практику выполняется на примере профильной организации. ВАРИАНТ 1 (Фамилии на «А») Тема: Анализатор отзывов Задача на интеграцию/MLOps: разработать скрипт для дообучения модели на новом наборе отзывов (CSV). Настроить CI/CD для автоматической сборки Docker-образа API. ВАРИАНТ 2 (Фамилии на «Б») Тема: Ботанический справочник Задача на интеграцию/MLOps: добавить 5 новых видов растений в датасет, дообучить модель и развернуть обновленную версию API без остановки сервиса. ВАРИАНТ 3 (Фамилии на «В») Тема: Ветеринарный помощник Задача на интеграцию/MLOps: провести нагрузочное тестирование API, имитируя 100 одновременных пользователей. Реализовать логирование всех запросов на классификацию. ВАРИАНТ 4 (Фамилии на «Г») Тема: Гастрономический гид Задача на интеграцию/MLOps: настроить HTTPS для API с помощью Nginx. Реализовать аутентификацию по статическому API-ключу. ВАРИАНТ 5 (Фамилии на «Д», «Е», «Ё») Тема: Детектор спама Задача на интеграцию/MLOps: автоматизировать запуск юнит-тестов в GitHub Actions. Развернуть API в Dockerконтейнере на виртуальной машине. ВАРИАНТ 6 (Фамилии на «Ж») Тема: Жанровый анализатор Задача на интеграцию/MLOps: настроить версионирование моделей. Реализовать эндпоинт для переключения между старой и новой версией модели. ВАРИАНТ 7 (Фамилии на «З») Тема: Защитник зрения	

Задача на интеграцию/MLOps: провести нагрузочное тестирование API для оценки производительности модели Pose Estimation. Настроить мониторинг потребления CPU. ВАРИАНТ 8 (Фамилии на «И», «Й») Тема: Идентификатор языка	

Задача на интеграцию/MLOps: реализовать логирование всех входящих запросов и ответов модели в базу данных SQLite для последующего анализа. ВАРИАНТ 9 (Фамилии на «К») Тема: Консультант по недвижимости Задача на интеграцию/MLOps: дообучить модель регрессии на новом наборе данных о квартирах. Сравнить точность старой и новой моделей. ВАРИАНТ 10 (Фамилии на «Л») Тема: Личный стилист Задача на интеграцию/MLOps: настроить CI/CD пайплайн, который автоматически собирает и публикует Dockerобраз в Docker Hub. ВАРИАНТ 11 (Фамилии на «М», «Н») Тема: Мобильный оценщик Задача на интеграцию/MLOps: провести нагрузочное тестирование API. Настроить Nginx для ограничения количества запросов в секунду от одного IP. ВАРИАНТ 12 (Фамилии на «О») Тема: Определитель эмоций Задача на интеграцию/MLOps: развернуть API на виртуальной машине. Написать скрипт для автоматического обновления приложения на сервере. ВАРИАНТ 13 (Фамилии на «П») Тема: Парковочный радар Задача на интеграцию/MLOps: настроить аудит безопасности API. Реализовать защиту от неавторизованного доступа. ВАРИАНТ 14 (Фамилии на «Р») Тема: Распознаватель рукописного текста Задача на интеграцию/MLOps: дообучить модель на новом наборе рукописных символов (например, буквы). Развернуть обновленную версию. ВАРИАНТ 15 (Фамилии на «С») Тема: Сортировщик мусора Задача на интеграцию/MLOps: настроить мониторинг доступности API и получать уведомления (например, в	

Telegram) при падении сервиса. ВАРИАНТ 16 (Фамилии на «Т») Тема: Технический ассистент Задача на интеграцию/MLOps: реализовать логирование всех запросов в структурированном формате JSON для удобного анализа.	

ВАРИАНТ 17 (Фамилии на «У») Тема: Умный переводчик Задача на интеграцию/MLOps: провести нагрузочное тестирование API, чтобы определить максимальное количество запросов в минуту, которое он может обработать. ВАРИАНТ 18 (Фамилии на «Ф») Тема: Фитнес-тренер Задача на интеграцию/MLOps: оптимизировать Docker-образ для уменьшения его размера. Настроить CI/CD. ВАРИАНТ 19 (Фамилии на «Х») Тема: Хэштег-генератор Задача на интеграцию/MLOps: дообучить модель на новых категориях изображений. Реализовать «горячую» замену модели. ВАРИАНТ 20 (Фамилии на «Ц», «Ч») Тема: Цифровой искусствовед Задача на интеграцию/MLOps: настроить Nginx для кэширования ответов API, если это применимо для модели Style Transfer. ВАРИАНТ 21 (Фамилии на «Ш», «Щ») Тема: Штурман Задача на интеграцию/MLOps: провести тестирование API на устойчивость, отправляя изображения с низким качеством и в разных форматах. ВАРИАНТ 22 (Фамилии на «Ы», «Э») Тема: Эколог Задача на интеграцию/MLOps: развернуть API на сервере. Автоматизировать сборку Docker-образа в GitHub Actions. ВАРИАНТ 23 (Фамилии на «Ю») Тема: Юридический помощник Задача на интеграцию/MLOps: настроить строгую аутентификацию API. Реализовать ролевую модель (например, администратор может дообучать модель). ВАРИАНТ 24 (Фамилии на «Я») Тема: Языковой барьер Задача на интеграцию/MLOps: оптимизировать производительность API для обработки видеопотока в реальном	

времени. Провести нагрузочное тестирование.	
Экзамен по модулю	**8**
Всего	**696**

3. УСЛОВИЯ РЕАЛИЗАЦИИ ПРОГРАММЫ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

3.1. Для реализации программы профессионального модуля должны быть предусмотрены следующие специальные помещения

Реализация профессионального модуля происходит в кабинете, который предназначен для проведения теоретических и практических занятий по междисциплинарному курсу.

Оборудование учебного кабинета: рабочие места обучающихся; рабочее место преподавателя; технические средства обучения: мультимедийный комплекс (проектор, экран), компьютер, колонки; оборудование: камера видеонаблюдения; доска меловая. Помещение для самостоятельной и воспитательной работы: столы, стулья, ПК обучающихся, программное обеспечение: 1С Предприятие, OnlyOffice (AGPLv3), GIMP (GNU GPL 3), Krita (GNU GPL 3), Inkscape (GNU GPL 3),
SciLab (CeCILL), Kaspersky Endpoint Security, Агент администрирования Kaspersky Security Center, Mozilla Firefox (MPL 2), Apache NetBeans IDE (Apache License 2), Notepadqq (GNU GPL 3), VSCodium (MIT), Mono Development IDE (MIT), PyCharm Community Edition (Apache License 2), Android Studio (Apache License 2), Python (Python Software Foundation License), Java (GNU GPL), Node.js (MIT), Git (GNU GPL 2), GitHub Desktop (No Copyright), UnityHub (Unity Companion License), FreeCAD (LGPL-2.0-or-later), Sweet Home 3D (GNU GPL 2+), draw.io (Apache License 2), DBeaver (Apache License 2), PostgreSQL (License (free and open-source)), MariaDB (GNU GPL), PSPP (GNU GPL 3), Wireshark (GNU GPL 2+),
Imunes (Creative Commons Attribution 4.0 International Public License), VirtualBox (GNU GPL 2), Apache (Apache License 2), Консультант+ (Коммерческая лицензия), OBS (GNU GPL).

3.2. Информационное обеспечение реализации программы

Для реализации программы библиотечный фонд образовательной организации должен иметь электронные образовательные и информационные ресурсы, рекомендуемых для использования в образовательном процессе.

Основная учебная литература:

1. Шестернинов, Е. Е. Индивидуальный проект. Шаг в профессию. Базовый уровень. Практикум: учебное пособие, разработанное в комплекте с учебником для образовательных организаций, реализующих образовательные программы среднего профессионального образования / Е. Е. Шестернинов. – 2-е изд., стер. - Москва: Просвещение, 2025. - 82 с. – (Учебник СПО). - ISBN 978-5-09-124912-5. - Текст: электронный. - URL: https://znanium.ru/catalog/product/2202038

2. Воронов, М. В. Системы искусственного интеллекта: учебник и практикум для среднего профессионального образования / М. В. Воронов, В. И. Пименов, И. А. Небаев. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2026. — 268 с. — (Профессиональное образование). — ISBN 978-5-534-17699-5. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/590238

3. Станкевич, Л. А. Интеллектуальные системы и технологии: учебник и практикум для среднего профессионального образования / Л. А. Станкевич. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2026. — 478 с. — (Профессиональное образование). — ISBN 978-5-534-20364-6. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/587749

4. Щербак, А. В. Поддержка и тестирование программных модулей: учебник для среднего профессионального образования / А. В. Щербак. — Москва: Издательство Юрайт, 2026. — 145 с. — (Профессиональное образование). — ISBN 978-5-534-19290-2. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/590253

Дополнительная учебная литература:

1. Воронов, М. В. Системы искусственного интеллекта: учебник и практикум для вузов / М. В. Воронов, В. И. Пименов, И. А. Небаев. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2026. — 268 с. — (Высшее образование). — ISBN 978-5-534-17032-0. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/588642

2. Новиков, Ф. А. Символический искусственный интеллект: математические основы представления знаний: учебник для вузов / Ф. А. Новиков. — Москва: Издательство Юрайт, 2026. — 278 с. — (Высшее образование). — ISBN 978-5-534-00734-3. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/584114

3. Загорулько, Ю. А. Искусственный интеллект. Инженерия знаний: учебное пособие для вузов / Ю. А. Загорулько, Г. Б. Загорулько. — Москва: Издательство Юрайт, 2024. — 93 с. — (Высшее образование). — ISBN 978-5-534-07198-6. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/540987

Базы данных и/или Интернет-ресурсы:

1. https://www.kaggle.com/ – Платформа для соревнований по машинному обучению, наборы данных и примеры кода
2. https://scikit-learn.org/stable/user_guide.html – Официальная документация по ключевой библиотеке машинного обучения Scikit-learn.3
3. https://fastapi.tiangolo.com/ru/ – Официальная документация фреймворка FastAPI на русском языке
4. https://flutter.dev/docs – Официальная документация фреймворка Flutter для мобильной разработки
5. https://docs.pytest.org/en/stable/ – Документация по фреймворку Pytest для тестирования Python-кода
6. https://docs.docker.com/ – Официальная документация по технологии контейнеризации Docker
7. https://docs.github.com/ru/actions – Руководства по автоматизации рабочих процессов с помощью GitHub Actions
8. https://habr.com/ru/hub/python/ – Профильный ресурс с практическими статьями по разработке на Python

Официальные, справочно-библиографические и периодические издания:

1. Информационные технологии и вычислительные системы: научный журнал / гл.
ред. Попков Ю.С.; учред. ФГУ «Федеральный исследовательский центр «Информатика и управление» РАН». - М., 2026

2. Экономика и математические методы: журнал / гл. ред. В.Л. Макаров / Российская Академия наук, Центральный экономико-математический институт, Институт проблем рынка - Российская академия наук, Москва, 2026

3. Экономический журнал Высшей школы экономики / Higher School of Economics Economic Journal: журнал / гл. ред. Е.Е. Гавриленков / Национальный исследовательский университет «Высшая школа экономики» - Издательский дом Высшей школы экономики, Москва, 2026

4. КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

Код ПК, ОК	Критерии оценки результата (показатели освоенности компетенций)	Формы контроля и методы оценки
ПК.1.1	**Оценка "отлично":** алгоритмы разработаны в полном соответствии с ТЗ, оптимизированы и понятны. **Оценка "хорошо":** алгоритмы разработаны в соответствии с ТЗ, но допускают незначительные отклонения. **Оценка "удовлетворительно":** алгоритмы разработаны с частичным соответствием ТЗ.	Защита отчёта с демонстрацией разработанных алгоритмов и устное собеседование.
ПК.1.2	**Оценка "отлично":** программные модули разработаны в полном соответствии с ТЗ, тесты проходят успешно. **Оценка "хорошо":** программные модули разработаны с минимальными несоответствиями, тесты в целом успешны. **Оценка "удовлетворительно":** программные модули разработаны с существенными доработками.	Защита отчёта по разработанным модулям, проверка выполнения ТЗ и прохождения тестов.
ПК.1.3	**Оценка "отлично":** код полностью оформлен в соответствии с требованиями, включая комментарии и стиль кода. **Оценка "хорошо":** код оформлен в соответствии с требованиями, допускаются мелкие недочёты. **Оценка "удовлетворительно":** код оформлен частично в соответствии с требованиями.	Проверка оформленного кода, соблюдения стиля и соответствия установленным требованиям.

ПК.1.4	**Оценка "отлично":** система контроля версий используется эффективно, изменения фиксируются корректно. **Оценка "хорошо":** система контроля версий используется, но имеются мелкие нарушения порядка фиксации изменений. **Оценка "удовлетворительно":** система контроля версий используется частично или с ошибками.	Проверка использования системы контроля версий (репозиторий), демонстрация фиксации изменений.
ПК.1.5	**Оценка "отлично":** отладка выполнена полностью, ошибки устранены, работа модулей оптимизирована. **Оценка "хорошо":** отладка выполнена, ошибки устранены, но оптимизация частичная. **Оценка "удовлетворительно":** отладка выполнена частично, ошибки устранены не полностью.	Демонстрация процесса отладки с использованием инструментов, отчёт по устранённым ошибкам.
ПК.1.6	**Оценка "отлично":** тестирование выполнено в полном объёме, тесты соответствуют ТЗ, выявленные ошибки исправлены. **Оценка "хорошо":** тестирование выполнено, тесты соответствуют ТЗ, незначительные	Предоставление отчёта о тестировании, демонстрация
	ошибки остались. **Оценка "удовлетворительно":** тестирование выполнено частично, ошибки выявлены, но не исправлены.	успешного прохождения тестов.
ПК.1.7	**Оценка "отлично":** тестовые сценарии составлены полностью, покрывают все функциональные требования. **Оценка "хорошо":** тестовые сценарии составлены, но не покрывают незначительную часть функциональных требований. **Оценка "удовлетворительно":** тестовые сценарии составлены частично, покрывают минимальный функционал.	Предоставление тестовых сценариев, проверка их соответствия функциональным требованиям.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

Типовые контрольные задания или иные материалы, необходимые для оценки результатов обучения по учебной дисциплине

Темы для подготовки к промежуточной аттестации

МДК.01.01 Разработка программных модулей в системах искусственного интеллекта

1.Понятие Искусственного Интеллекта, его история и основные направления. 2.Ключевые задачи машинного обучения: классификация, регрессия, кластеризация.
3.Экосистема Python для ИИ: обзор библиотек NumPy, Pandas, Scikit-learn. 4.Принципы создания и управления виртуальными окружениями.
5.Интерактивная разработка в Jupyter Notebook, структура ноутбука.
6.Библиотека NumPy: основной объект ndarray и принцип векторизации.
7.Индексация, срезы и булевы маски в массивах NumPy.
8.Библиотека Pandas: структуры данных Series и DataFrame.
9.Чтение и запись данных из/в файлы (CSV, Excel) с помощью Pandas.
10.Методы выборки и фильтрации данных в Pandas: loc, iloc.
11.Группировка данных с помощью groupby и применение агрегирующих функций.
12.Методы обнаружения и обработки пропущенных значений в Pandas.
13.Исследовательский анализ данных (EDA) и выявление аномалий.
14.Проблема дисбаланса классов и базовые подходы к ее решению.
15. Понятие базовой модели (Baseline) и ее роль в оценке качества.
16. Основы инженерии признаков (Feature Engineering).
17. Алгоритм Линейной регрессии: принцип работы и применение.
18. Алгоритм Логистической регрессии для задач классификации.
19. Алгоритм k-ближайших соседей (KNN) и понятие метрики расстояния.
20. Алгоритм Дерева решений: критерии разбиения и визуализация.
21. Алгоритм кластеризации K-Means: принцип работы и понятие центроида.
22. Разделение выборки на обучающую и тестовую (Train-Test Split).
23. Проблема переобучения (Overfitting) и недообучения (Underfitting). 24. Кросс-валидация как метод робастной оценки качества модели.
25. Матрица ошибок (Confusion Matrix).
26. Метрики классификации: Accuracy, Precision, Recall. 27. Компромисс между Precision и Recall (Precision/Recall trade-off) и F1-мера.
28. ROC-кривая и метрика AUC-ROC.
29. Принципы выбора ключевой метрики в зависимости от стоимости ошибок для бизнесзадачи.
30. Основные метрики качества для задач регрессии (MAE, MSE, R^2).
31. Ансамблевые методы: бэггинг и алгоритм Случайный лес (Random Forest).
32. Ансамблевые методы: бустинг и алгоритм Градиентного бустинга. 33. Масштабирование числовых признаков: стандартизация и нормализация.
34. Кодирование категориальных признаков: Label Encoding и One-Hot Encoding. 35. Понятие утечки данных (Data Leakage) и способы ее предотвращения.

36. Принципы "Чистого кода" (Clean Code) и рефакторинг.
37. Стандарт оформления кода PEP8 и инструменты автоматического форматирования (Black).
38. Документирование кода: docstrings и аннотации типов (Type Hinting).
39. Основы системы контроля версий Git: рабочий цикл (add, commit, push).
40. Работа с ветками в Git: создание, переключение, слияние (branch, checkout, merge).
41. Принципы юнит-тестирования и фреймворк Pytest.
42. Использование фикстур в Pytest для подготовки тестовых данных.
43. Параметризация тестов в Pytest для проверки множества входных данных.
44. Архитектурный стиль REST и основные HTTP-методы.
45. Фреймворк FastAPI и его основные преимущества.
46. Валидация данных с помощью Pydantic и контракт API (OpenAPI/Swagger).
47. Версионирование API.
48. Логирование запросов и централизованная обработка ошибок в API.
49. Механизм CORS и его настройка для взаимодействия с веб-клиентами.
50. Основы контейнеризации Python-приложений с помощью Docker.

МДК.01.02 Разработка мобильных приложений с поддержкой искусственного интеллекта

1. Язык Dart: синтаксис, система типов, отличия от Python.
2. Концепция Null Safety в Dart (?, !, late, required).
3. Переменные и константы в Dart (var, final, const).
4. Управляющие конструкции в Dart (if/else, switch/case, циклы).
5. Функции в Dart: обязательные, опциональные и именованные параметры.
6. Коллекции в Dart: списки List.
7. Коллекции в Dart: множества Set и карты Map.
8. Основы ООП в Dart: классы, объекты, свойства, методы.
9. Конструкторы классов в Dart: основной, именованный, factory.
10. Наследование классов (extends) и переопределение методов (@override).
11. Архитектура Flutter, декларативный подход к построению UI. 12. Концепция дерева виджетов.
13. Различие между StatelessWidget и StatefulWidget.
14. Структура Flutter-проекта, назначение файла pubspec.yaml. 15. Жизненный цикл StatefulWidget.
16. Основные виджеты: Text, Icon, Image, Asset.
17. Виджет Scaffold как основа структуры экрана.
18. Установка Flutter SDK и настройка эмулятора/устройства. 19. Виджеты для верстки: Row и Column.
20. Свойства `mainAxisAlignment` и crossAxisAlignment.
21. Виджет Container: свойства color, padding, margin, decoration.
22. Виджет Stack для позиционирования элементов поверх друг друга.
23. Обработка нажатий: виджеты ElevatedButton, TextButton, IconButton.
24. Виджет GestureDetector для обработки различных жестов.
25. Поля для ввода текста: виджет TextField и его свойства.

26. Контроллеры (TextEditingController) для работы с текстовыми полями.
27. Принципы навигации в Flutter: Navigator.push и Navigator.pop.
28. Именованная навигация и передача аргументов между экранами.
29. Эффективное отображение списков с помощью ListView.builder.
30. Стилизация элементов списка: виджет ListTile.
31. Отображение сеток с помощью GridView.
32. Виджет `Form` для группировки полей ввода.
33. Валидация данных в формах.
34. Асинхронное программирование в Dart: Future, async, await.
35. Основы протокола HTTP: методы GET и POST, статусные коды.
36. Использование библиотеки http для выполнения сетевых запросов.
37. Преобразование (парсинг) данных из формата JSON в объекты Dart.
38. Преобразование объектов Dart в формат JSON для отправки на сервер.
39. Создание сервисного слоя для инкапсуляции логики работы с API.
40. Отображение индикаторов загрузки (CircularProgressIndicator) во время сетевых запросов.
41. Обработка ошибок при выполнении сетевых запросов (try-catch).
42. Отображение сообщений об ошибках пользователю.
43. Полный цикл взаимодействия с API: отправка данных из формы, получение и отображение результата.
44. Подходы к управлению состоянием (State Management): setState.
45. Обзор и основная идея пакетного менеджера состояний Provider.
46. Настройка иконки приложения для Android и iOS.
47. Настройка отображаемого имени и версии приложения.
48. Процесс сборки установочного файла .apk для Android.
49. Основы ручного тестирования мобильного приложения (Smoke-тесты). 50. Обзор процесса публикации приложения в Google Play.

МДК.01.03 Тестирование программных модулей

1. Понятие качества ПО, разница между Quality Assurance (QA) и Quality Control (QC).
2. Семь принципов тестирования по ISTQB.
3. Пирамида тестирования: модульный, интеграционный, системный, приемочный уровни.
4. Роль тестирования в различных моделях разработки (Waterfall, Agile).
5. Виды функционального тестирования: Smoke и Sanity.
6. Виды функционального тестирования: регрессионное и повторное (re-testing).
7. Виды нефункционального тестирования: тестирование производительности.
8. Виды нефункционального тестирования: тестирование удобства использования (Usability) и совместимости.
9. Различие между статическим и динамическим тестированием.
10. Дефект (баг): его атрибуты (серьезность, приоритет).
11. Жизненный цикл дефекта (Bug Life Cycle).
12. Тестовая документация: Тест-план.
13. Тестовая документация: Тест-кейс и Чек-лист.

14. Техники тест-дизайна: Классы эквивалентности.
15. Техники тест-дизайна: Анализ граничных значений.
16. Техники тест-дизайна: Таблицы решений.
17. Цели и принципы юнит-тестирования. 18. Обзор фреймворка Pytest и его преимущества.
19. Структура теста "Arrange-Act-Assert".
20. Использование стандартных утверждений assert в Pytest.
21. Правила обнаружения тестов в Pytest.
22. Запуск и фильтрация тестов из командной строки.
23. Использование маркеров (@pytest.mark) для группировки тестов.
24. Понятие тестового покрытия (Code Coverage) и его значимость.
25. Использование pytest-cov для генерации отчета о покрытии. 26. Проблема внешних зависимостей в юнит-тестах.
27. Концепция тестовых двойников (Test Doubles): Mocks, Stubs, Fakes.
28. Использование `unittest.mock` и `patch` для изоляции тестов.
29. Понятие фикстур в Pytest (@pytest.fixture).
30. Параметризация тестов (@pytest.mark.parametrize).
31. Цели и задачи интеграционного тестирования.
32. Подходы к интеграционному тестированию («снизу-вверх», «сверху-вниз», «большой взрыв»).
33. Основы тестирования REST API.
34. Использование Postman для ручного тестирования API.
35. Автоматизация API-тестов на Python с помощью библиотеки `requests`.
36. Особенности тестирования систем на основе машинного обучения.
37. Тестирование контракта API: проверка схем, версий и кодов ответа.
38. Валидация JSON-схемы ответа в тестах.
39. Тестирование инвариантов модели: проверка на смысловую непротиворечивость.
40. Тестирование данных: базовые подходы к обнаружению дрейфа данных (Data Drift).
41. Цели и задачи End-to-End (E2E) тестирования.
42. Обзор инструментов для автоматизации UI-тестирования.
43. Архитектура Selenium WebDriver.
44. Поиск веб-элементов: CSS-селекторы и XPath.
45. Проблема синхронизации в UI-тестах: явные и неявные ожидания.
46. Концепция CI/CD (Непрерывная интеграция, доставка, развертывание).
47. Обзор платформы GitHub Actions, ее основные компоненты (workflow, job, step, action).
48. Синтаксис YAML-файлов для конфигурации workflow.
49. Интеграция статического анализа (linting) в CI-пайплайн.
50. Настройка CI-пайплайна для автоматического запуска тестов (`pytest`) при push в репозиторий.

Темы для самостоятельной работы обучающихся

МДК.01.01 Разработка программных модулей в системах искусственного интеллекта

1. История развития нейронных сетей: от перцептрона до глубокого обучения.

2. Обзор и сравнение сред разработки: Jupyter Lab, Google Colab, VS Code. 3. Применение Markdown для научного и технического документирования.
4. Исследование производительности: сравнение операций в NumPy и на чистом Python.
5. Продвинутые методы индексации в NumPy (fancy indexing).
6. Принцип "broadcasting" (транслирования) в NumPy: примеры и применение.
7. Работа с иерархическими индексами (MultiIndex) в Pandas.
8. Слияние и объединение DataFrame в Pandas (merge, join, concat).
9. Работа с временными рядами (Time Series) в Pandas.
10. Визуализация данных с помощью встроенных методов Pandas.
11. Применение библиотеки Matplotlib для построения графиков.
12. Применение библиотеки Seaborn для статистической визуализации.
13. Исследование и применение библиотек для интерактивной визуализации (Plotly, Altair).
14. Обзор и классификация методов борьбы с дисбалансом классов (SMOTE, ADASYN).
15. Подготовка доклада на тему «Современные подходы к инженерии признаков (Feature Engineering)».
16. Математические основы Линейной регрессии.
17. Сравнение L1 и L2 регуляризации в линейных моделях.
18. Различные метрики расстояния и их влияние на алгоритм KNN.
19. Алгоритм C4.5 как развитие дерева решений ID3.
20. Методы обрезки (pruning) деревьев решений для борьбы с переобучением.
21. Сравнение критериев разбиения в деревьях решений: Gini и Entropy.
22. Исследование влияния выбора начальных центроидов в алгоритме K-Means. 23. Метод локтя (Elbow Method) для определения оптимального числа кластеров.
24. Стратегии кросс-валидации (K-Fold, Stratified K-Fold, Leave-One-Out).
25. Построение и анализ калибровочной кривой (Calibration Curve).
26. Детальный разбор метрики AUC-ROC и ее практическая интерпретация.
27. Исследование метрик для мультиклассовой классификации.
28. Подготовка доклада на тему «Интерпретируемость моделей машинного обучения (SHAP, LIME)».
29. Сравнительный анализ алгоритмов градиентного бустинга: XGBoost, LightGBM, CatBoost.
30. Применение метода главных компонент (PCA) для понижения размерности.
31. Алгоритмы кластеризации, основанные на плотности (DBSCAN).
32. Автоматизация подбора гиперпараметров: GridSearchCV vs RandomizedSearchCV. 33. Обзор и применение библиотеки Optuna для байесовской оптимизации гиперпараметров.
34. Паттерны проектирования в Python (Singleton, Factory).
35. Принципы SOLID и их применение в разработке на Python.
36. Создание собственного декоратора в Python.
37. Исследование и применение библиотеки `Poetry` для управления зависимостями.
38. Изучение основ Docker Compose для оркестрации сервисов.
39. Написание документации к проекту с использованием Sphinx.
40. Работа с Git: интерактивный rebase (git rebase -i).
41. Методология Git Flow для организации веток в проекте.

42. Применение `pre-commit` хуков для автоматической проверки кода перед коммитом.
43. Продвинутые возможности Pytest: создание собственных маркеров.
44. Сравнительный анализ фреймворков FastAPI и Flask.
45. Реализация WebSocket-сервера на FastAPI.
46. Защита API: аутентификация с помощью OAuth2 и JWT-токенов.
47. Асинхронная работа с базами данных в FastAPI.
48. Исследование и применение библиотеки Pydantic для сложных моделей валидации.
49. Мониторинг API-сервиса с помощью Prometheus и Grafana. 50. Подготовка доклада на тему «Введение в Kubernetes для развертывания приложений».

МДК.01.02 Разработка мобильных приложений с поддержкой искусственного интеллекта

1. История и архитектура Flutter. Сравнение с React Native и нативной разработкой.
2. Язык Dart: детальное изучение асинхронного программирования (Stream, Future).
3. Различия и применение final vs const в Dart.
4. Генераторы в Dart (sync*, async*).
5. Использование расширений (extensions) для добавления функциональности к классам.
6. Применение миксинов (mixins) в Dart для переиспользования кода.
7. Принципы эффективной разработки на Dart. 8. Обзор и установка Flutter на разных операционных системах (Windows, macOS, Linux).
9. Сравнение Android Studio и Visual Studio Code для Flutter-разработки.
10. Горячая перезагрузка (Hot Reload) vs Горячий перезапуск (Hot Restart): как это работает.
11. Использование DevTools для отладки и профилирования Flutter-приложений.
12. Работа с ассетами: добавление изображений и кастомных шрифтов.
13. Публикация собственного пакета на pub.dev.
14. Продвинутая верстка: виджеты Expanded, Flexible, Spacer.
15. Создание адаптивного UI для разных размеров экрана с помощью LayoutBuilder и MediaQuery.
16. Виджет FittedBox для масштабирования дочерних элементов.
17. Создание кастомных виджетов.
18. Использование виджетов в стиле Cupertino для создания iOS-подобного интерфейса.
19. Работа с темами (Themes) для централизованного управления стилями приложения.
20. Создание простых анимаций с помощью AnimatedContainer.
21. Продвинутая анимация: AnimationController и Tweens. 22. Hero-анимации для создания эффектных переходов между экранами.
23. Обработка пользовательских жестов с помощью InkWell и GestureDetector.
24. Реализация кастомных "сливающихся" заголовков (Slivers).
25. Создание и кастомизация AlertDialog и SnackBar.
26. Работа с TabBar и TabView для создания вкладок.
27. Навигация с помощью Drawer (боковое меню).
28. Глубокое изучение библиотеки http и обработка различных типов ответов.
29. Сравнительный анализ библиотек для сетевых запросов: http vs dio.
30. Использование dio для перехвата запросов (interceptors) и логирования.
31. Автоматическая генерация кода для JSON-сериализации (`json_serializable`).

32. Работа с WebSocket во Flutter для real-time коммуникации.
33. Основы Firebase: подключение и настройка.
34. Использование Firebase Authentication для аутентификации пользователей.
35. Хранение данных в реальном времени с Firebase Firestore.
36. Хранение файлов с Firebase Storage.
37. Обзор и применение GraphQL во Flutter.
38. Создание простого GraphQL-клиента.
39. Глубокое изучение setState и его влияния на производительность.
40. Сравнительный анализ подходов к State Management: Provider vs Riverpod.
41. Сравнительный анализ подходов к State Management: BLoC vs MobX.
42. Реализация простого приложения с использованием Provider.
43. Реализация простого приложения с использованием BLoC.
44. Локальное хранение данных с помощью shared_preferences.
45. Работа с локальной базой данных SQLite с помощью пакета sqflite.
46. Изучение пакета path_provider для работы с файловой системой.
47. Подготовка и сборка приложения для iOS.
48. Обзор процесса публикации приложения в App Store.
49. Настройка Splash Screen (экрана-заставки) для Android и iOS. 50. Подготовка доклада на тему «Оптимизация производительности Flutter-приложений».

МДК.01.03 Тестирование программных модулей

1. История и эволюция тестирования программного обеспечения.
2. Сравнение ролей QA Engineer, QC Engineer и Software Development Engineer in Test (SDET). 3. Детальный разбор 7 принципов тестирования ISTQB с примерами.
4. Сравнительный анализ моделей разработки: V-Model, Spiral Model, RUP. 5.
Альфа-, бета- и гамма-тестирование: цели и участники.
6. Исследовательское (exploratory) тестирование и ad-hoc тестирование.
7. Тестирование локализации (l10n) и интернационализации (i18n).
8. Обзор стандартов ISO/IEC 25010 (SQuaRE) по качеству ПО.
9. Тестирование документации.
10. Подготовка доклада на тему «Психология тестирования».
11. Использование техник тест-дизайна: «Причина и следствие».
12. Использование техник тест-дизайна: «Попарное тестирование» (Pairwise testing).
13. Обзор систем управления тестированием (Test Management Systems): TestRail, Zephyr.
14. Обзор систем отслеживания ошибок (Bug Tracking Systems): Jira, Redmine.
15. Сравнительный анализ фреймворков Pytest и Unittest.
16. Изучение структуры и применения setup и teardown методов в Unittest.
17. Создание кастомных фикстур в Pytest с разной областью видимости (scope).
18. Использование conftest.py для общих фикстур в проекте.
19. Подготовка доклада на тему «Разработка через тестирование (TDD)».
20. Практический пример реализации простого класса с использованием TDD.
21. Изучение основ «Разработки через поведение (BDD)» и фреймворка behave.
22. Мутационное тестирование: концепция и инструменты (например, mutmut).
23. Использование pytest-xdist для параллельного запуска тестов.

24. Свойства (property-based) тесты с использованием библиотеки Hypothesis.
25. Логирование внутри тестов и управление выводом в Pytest.
26. Тестирование "белого" и "черного" ящика: сравнительный анализ.
27. Стратегии заглушек (stubbing) для интеграционного тестирования.
28. Детальное изучение unittest.mock: Mock, MagicMock, patch. 29. Использование responses или pytest-httpserver для мокирования HTTP-запросов.
30. Тестирование асинхронного кода на Python (asyncio, pytest-asyncio). 31.
Автоматизация тестирования API с использованием Postman и Newman.
32. Контрактное тестирование: обзор и применение Pact.
33. Специфика тестирования GraphQL API.
34. Специфика тестирования gRPC сервисов.
35. Обзор и применение библиотек для валидации данных (Great Expectations, Pandera).
36. Сравнительный анализ инструментов для автоматизации UI: Selenium, Cypress, Playwright.
37. Паттерн проектирования Page Object Model (POM) в UI-автоматизации.
38. Работа с Selenium WebDriver Manager для автоматического управления драйверами.
39. Запуск UI-тестов в headless-режиме.
40. Использование Allure для создания наглядных отчетов о тестировании.
41. Нагрузочное тестирование: обзор инструментов JMeter, Gatling.
42. Исследование и применение Locust для написания нагрузочных тестов на Python.
43. Основы тестирования безопасности: OWASP Top 10.
44. Встроенные средства Flutter для написания виджет-тестов.
45. Встроенные средства Flutter для написания интеграционных тестов.
46. Сравнение CI/CD платформ: GitHub Actions, GitLab CI, Jenkins.
47. Создание сложных workflow в GitHub Actions с условиями и зависимостями.
48. Использование секретов (secrets) в GitHub Actions для безопасного хранения токенов.
49. Подготовка доклада на тему «Введение в Infrastructure as Code (IaC) и Terraform». 50. Подготовка доклада на тему «Роль и задачи SDET в современной IT-компании».

Примерные варианты тестовых оценочных заданий (ТОЗ) для контрольного рубежа в рамках текущего контроля

МДК.01.01 Разработка программных модулей в системах искусственного интеллекта

Тестовое задание	Оцениваемые компетенции
ВОПРОСЫ ОТКРЫТОГО ТИПА	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Подход в машинном обучении, при котором модель обучается на данных с известными правильными ответами, называется _____. +: обучение с учителем	ПК 1.1.

(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Основной объект для многомерных массивов в библиотеке NumPy — это _____.	ПК 1.1.

+: массив NumPy	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Основная структура данных для работы с таблицами в библиотеке Pandas — это _____. +: объект DataFrame	ПК 1.1.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс преобразования категориальных признаков в числовой формат называется _____. +: кодированием признаков	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Ситуация, когда модель слишком хорошо подстраивается под обучающие данные и плохо работает на новых, называется _____. +: переобучением модели	ПК 1.1.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Стандарт оформления кода для языка Python, определяющий правила именования и форматирования, — это _____. +: стандарт PEP8	ПК 1.3.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Распределенная система, используемая для отслеживания изменений в коде и совместной работы, — это _____. +: система контроля версий	ПК 1.4.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Команда Git для сохранения изменений в локальном репозитории — это _____. +: команда git commit	ПК 1.4.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс объединения изменений из одной ветки в другую в Git называется _____. +: слиянием веток	ПК 1.4.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Стиль написания имен, где слова разделяются нижним подчеркиванием, — это _____. +: стиль snake_case	ПК 1.3.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Специальная строка в начале функции или модуля, описывающая его назначение, — это _____. +: документирующая строка	ПК 1.3.

(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Инструмент для отладки, позволяющий остановить выполнение программы в определенной строке, — это _____. +: точка останова	ПК 1.5.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Тесты, проверяющие работоспособность отдельных, изолированных функций или методов, называются _____. +: модульными тестами	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Архитектурный стиль для построения API, основанный на использовании стандартных HTTP-методов, — это _____. +: архитектура REST	ПК 1.2.

(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс «упаковки» приложения и его зависимостей в изолированную среду называется _____. +: контейнеризацией приложения	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Метрика, показывающая долю правильно предсказанных объектов среди всех объектов, которые модель отнесла к положительному классу, — это _____. +: метрика Precision	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Метрика, показывающая, какую долю объектов положительного класса модель смогла правильно обнаружить, — это _____. +: метрика Recall	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Метод оценки качества модели, при котором выборка многократно разбивается на части, — это _____. +: кросс-валидация	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Технология, позволяющая автоматически создавать интерактивную документацию для API, — это _____. +: спецификация OpenAPI	ПК 1.3.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс преобразования объекта Python в поток байтов для сохранения или передачи называется _____. +: процессом сериализации	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс создания новых информативных признаков из существующих необработанных данных называется _____. +: инженерией признаков	ПК 1.1.
ВОПРОСЫ ЗАКРЫТОГО ТИПА	

Какая библиотека используется в Python для научных вычислений и работы с многомерными массивами? +: NumPy -: Pandas -: Matplotlib -: FastAPI	ПК 1.1.
Какой метод Pandas используется для вывода первых N строк DataFrame? +: head() -: first() -: top() -: info()	ПК 1.1.
Какой алгоритм машинного обучения используется для разделения данных на заранее неизвестные группы? +: Кластеризация -: Регрессия -: Классификация -: Ранжирование	ПК 1.1.
Какая команда Git используется для загрузки изменений из удаленного репозитория в локальный? +: pull -: push	ПК 1.4.
-: fetch -: clone	
Какой HTTP-метод обычно используется для получения данных от сервера в REST API? +: GET -: POST -: PUT -: DELETE	ПК 1.2.
Какая из этих метрик НЕ используется для оценки качества моделей классификации? +: MAE (Mean Absolute Error) -: Accuracy -: F1-score -: AUC-ROC	ПК 1.6.
Какой инструмент используется для автоматизации сборки, тестирования и развертывания кода? +: CI/CD -: IDE -: SQL -: SDK	ПК 1.4.
Какой декоратор в Pytest используется для запуска одного теста с несколькими наборами входных данных? +: @pytest.mark.parametrize -: @pytest.fixture -: @pytest.mark.skip -: @pytest.mark.test	ПК 1.6.

Какая библиотека используется для создания асинхронных вебприложений и API в Python 3.7+? +: FastAPI -: Django -: Flask -: Scikit-learn	ПК 1.2.

МДК.01.02 Разработка мобильных приложений с поддержкой искусственного интеллекта

Тестовое задание	Оцениваемые компетенции
ВОПРОСЫ ОТКРЫТОГО ТИПА	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Язык программирования, используемый для разработки мобильных приложений на Flutter, — это _____. +: язык Dart	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Система Flutter, которая гарантирует, что переменные не могут быть null без явного указания, — это _____. +: Null Safety	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Основной строительный блок пользовательского интерфейса во Flutter — это _____. +: виджет Flutter	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ)	ПК 1.2.

Виджет во Flutter, состояние которого может изменяться после первоначальной инициализации, — это _____. +: StatefulWidget	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Метод во Flutter, который отвечает за построение и обновление пользовательского интерфейса виджета, — это _____. +: метод build	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Виджет во Flutter, предназначенный для горизонтального расположения нескольких дочерних элементов, — это _____. +: виджет Row	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Виджет во Flutter, используемый для навигации между различными экранами приложения, — это _____. +: виджет Navigator	ПК 1.2.

(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс асинхронного выполнения операций, не блокирующий основной поток пользовательского интерфейса, — это _____. +: асинхронное программирование	ПК 1.5.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Стандартный протокол для обмена данными между клиентом и сервером через интернет, — это _____. +: протокол HTTP	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Формат данных, широко используемый для обмена информацией между веб-сервером и мобильным приложением, — это _____. +: формат JSON	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс преобразования объектов Dart в формат JSON для отправки на сервер, — это _____. +: сериализация в JSON	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Слой приложения, инкапсулирующий всю логику взаимодействия с внешними API, — это _____. +: сервисный слой	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Виджет, отображающий круговой индикатор прогресса во время выполнения длительных операций, — это _____. +: CircularProgressIndicator	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс проверки корректности данных, введенных пользователем в форму, — это _____. +: валидация данных	ПК 1.7.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс создания установочного файла приложения для распространения, — это _____. +: сборка релиза	ПК 1.4.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ)	ПК 1.2.

Виджет во Flutter, предназначенный для динамического отображения большого количества однотипных элементов, — это _____. +: ListView.builder	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Специальная карта в файле pubspec.yaml, где указываются используемые внешние пакеты и их версии, — это _____. +: секция dependencies	ПК 1.2.

(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Концепция, описывающая, как данные приложения доступны и изменяются в пользовательском интерфейсе, — это _____. +: управление состоянием	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Один из простых подходов к управлению состоянием в Flutter, основанный на вызове метода виджета, — это _____. +: setState метод	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Файл, содержащий метаданные Flutter-проекта и его зависимости, — это _____. +: файл pubspec.yaml	ПК 1.2.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Подход к построению UI, при котором интерфейс является функцией текущего состояния приложения, называется _____. +: декларативным подходом	ПК 1.2.
ВОПРОСЫ ЗАКРЫТОГО ТИПА	
Какой тип переменных используется в Dart, если ее значение никогда не изменится после инициализации, и оно известно во время компиляции? +: const -: var -: final -: dynamic	ПК 1.2.
Какая функция Dart используется для асинхронного ожидания завершения операции? +: await -: async -: Future -: then	ПК 1.5.
Какой виджет используется для создания вертикального списка прокручиваемых элементов? +: Column -: Row -: Stack -: Container	ПК 1.2.
Какой виджет во Flutter используется для создания формы с полями ввода? +: Form -: TextField -: Column -: ListView	ПК 1.2.
Какой HTTP-метод обычно используется для отправки данных на сервер для создания нового ресурса? +: POST	ПК 1.2.
-: GET -: PUT -: DELETE	

Что означает код состояния HTTP 200? +: Успешный запрос -: Ресурс не найден -: Ошибка сервера -: Неавторизованный запрос	ПК 1.6.
Какой пакет Flutter часто используется для выполнения HTTPзапросов? +: http -: json -: provider -: flutter_bloc	ПК 1.2.
Какой объект Dart используется для представления значения, которое будет доступно в будущем? +: Future -: Stream -: Async -: Await	ПК 1.5.
Какой формат файлов используется для сборки релизной версии Android-приложения? +: .apk -: .ipa -: .exe -: .zip	ПК 1.4.

МДК.01.03 Тестирование программных модулей

Тестовое задание	Оцениваемые компетенции
ВОПРОСЫ ОТКРЫТОГО ТИПА	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Деятельность по обеспечению качества на всех этапах разработки, а не только на этапе проверки, — это _____. +: Quality Assurance	ПК 1.7.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Нижний, самый широкий уровень пирамиды тестирования, — это _____. +: модульное тестирование	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Быстрая проверка основного, самого критичного функционала системы на работоспособность, — это _____. +: дымовое тестирование	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Тестирование, проверяющее, что новые изменения не сломали уже существующий функционал, — это _____. +: регрессионное тестирование	ПК 1.6.

(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Проверка кода без его реального запуска (например, code review) — это _____.	ПК 1.6.

+: статическое тестирование	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Документ, описывающий шаги, ожидаемый результат и предусловия для проверки конкретной функции, — это _____. +: тестовый случай	ПК 1.7.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Совокупность состояний, через которые проходит дефект с момента обнаружения до закрытия, — это _____. +: жизненный цикл дефекта	ПК 1.7.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Техника тест-дизайна, основанная на проверке значений на границах и вблизи допустимых диапазонов, — это _____. +: анализ граничных значений	ПК 1.7.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Метрика, показывающая, какая часть кода была затронута во время выполнения тестов, — это _____. +: тестовое покрытие	ПК 1.7.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Объект-заглушка, используемый для имитации поведения реальных объектов и изоляции кода в юнит-тестах, — это _____. +: Mock-объект	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Декоратор в Pytest, позволяющий передавать в тест подготовленные данные или объекты, — это _____. +: декоратор @pytest.fixture	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Тестирование, проверяющее взаимодействие между несколькими модулями или компонентами системы, — это _____. +: интеграционное тестирование	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Тест, проверяющий, что поведение модели остается логически предсказуемым (например, цена не может быть отрицательной), — это _____. +: инвариантный тест	ПК 1.7.

(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Изменение в распределении входных данных модели с течением времени, — это _____. +: дрейф данных	ПК 1.7.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Тестирование, имитирующее полный путь реального пользователя через всю систему, — это _____. +: End-to-End тестирование	ПК 1.7.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Стратегия для поиска элементов на веб-странице в Selenium, — это _____. +: стратегия локаторов	ПК 1.6.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Практика автоматизации процессов сборки, тестирования и развертывания программного обеспечения — это _____. +: практика CI/CD	ПК 1.4.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ)	ПК 1.4.

Платформа от GitHub для автоматизации рабочих процессов непосредственно в репозитории — это _____. +: GitHub Actions	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Формат файлов, используемый для написания конфигураций в GitHub Actions, — это _____. +: формат YAML	ПК 1.4.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Процесс проверки кода на соответствие стандартам оформления без его выполнения, — это _____. +: статический анализ	ПК 1.3.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Формальный документ, который описывает цели, подходы, ресурсы и график запланированных тестовых мероприятий, — это _____. +: план тестирования	ПК 1.7.
ВОПРОСЫ ЗАКРЫТОГО ТИПА	
Какая техника тест-дизайна предполагает разделение всех возможных входных данных на группы, из которых берется по одному представителю? +: Классы эквивалентности -: Анализ граничных значений -: Таблица решений -: Попарное тестирование	ПК 1.7.

Какой инструмент используется для ручного тестирования REST API? +: Postman -: Pytest -: Selenium -: Docker	ПК 1.6.
Какой инструмент используется для автоматизации действий в веббраузере? +: Selenium WebDriver -: Pytest -: Postman -: Requests	ПК 1.6.
Какой плагин для Pytest используется для измерения тестового покрытия? +: pytest-cov -: pytest-xdist -: pytest-mock -: pytest-html	ПК 1.7.
Какой подход к тестированию используется, если у тестировщика нет доступа к исходному коду? +: Метод черного ящика -: Метод белого ящика -: Метод серого ящика -: Статическое тестирование	ПК 1.6.
Какой файл используется для определения рабочих процессов (workflows) в GitHub Actions? +: YAML-файл -: Python-скрипт -: JSON-файл	ПК 1.4.
-: Dockerfile	
Какой паттерн проектирования используется в UI-автоматизации для разделения логики тестов и логики взаимодействия со страницами? +: Page Object Model -: Singleton -: Factory -: MVC	ПК 1.7.
Какая часть CI/CD отвечает за автоматическое слияние кода от разных разработчиков в основную ветку? +: Непрерывная интеграция (CI) -: Непрерывная доставка (CD) -: Непрерывное развертывание (CD) -: Непрерывное тестирование (CT)	ПК 1.4.
Какой инструмент используется для создания Mock-объектов в Python? +: unittest.mock -: pytest.fixture -: requests -: selenium	ПК 1.6.

Шкала оценивания тестовых заданий

Данный вид контроля, рассчитан на выявление уровня усвоения теоретического и практического материала в рамках изучения дисциплин. Критерии оценивания теста: 1 вопрос = 1 балл.

Оценка «5»	30-27 Баллов
Оценка «4»	26-23 Баллов
Оценка «3»	22-18 Баллов
Оценка «2»	0-17 Баллов

ЛИСТ АКТУАЛИЗАЦИИ РАБОЧЕЙ ПРОГРАММЫ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОСВОЕНИЮ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

Методические рекомендации при подготовке к теоретическим занятиям

Основными видами аудиторной работы обучающихся являются лекции.

В ходе теоретических занятий преподаватель излагает и разъясняет основные, наиболее сложные понятия темы, а также связанные с ней теоретические и практические проблемы, дает рекомендации и указания на самостоятельную работу.

В ходе теоретических занятий необходимо вести конспектирование учебного материала. Обращать внимание на категории, формулировки, раскрывающие содержание тех или иных явлений и процессов, научные выводы и практические рекомендации. Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал прослушанной лекции, а также подчеркивающие особую важность тех или иных теоретических положений. Задавать преподавателю уточняющие вопросы с целью уяснения теоретических положений, разрешения спорных ситуаций.

Контроль ведения конспекта лекций

1. Конспект лекций обучающегося должен вестись в отдельной тетради с количеством страниц, рекомендованном ведущим лектором. На обороте титульного листа тетрадь с конспектом лекций должна содержать подписи:

- фамилия и инициалы обучающегося;
- номер группы;
- ФИО ведущих учебный предмет преподавателей.

2. В начале каждой лекции обучающийся должен записать тему и перечень рассматриваемых вопросов, а также перечень рекомендуемой литературы (при условии, что они оговариваются лектором).

3. При записи содержания лекции следует применять минимум сокращений, желательно общеупотребимых. Обучающимся может быть разработана своя система сокращений, с выносом ее в конце конспекта для пояснения.

4. Табличный, графический, цифровой материал должен быть точно перенесен в конспект с доски (проектора) без искажений динамики зависимостей и обозначений формул, а также значений результатов расчетов.

5. Тетрадь с конспектом должна содержать полный объем прочитанного материала лекций по их количеству и полноте изложения, включая материал для самостоятельного изучения и конспектирования.

Записи должны быть сделаны аккуратным почерком, разборчиво.

Методические рекомендации при проведении практических занятий

Практические занятия необходимы для закрепления изученного теоретического материала, контроля преподавателем степени подготовленности обучающихся по изучаемым учебным предметом, дисциплине или профессиональному модулю.

Перед практическим занятием следует изучить конспект лекции и рекомендованную преподавателем литературу, обращая внимание на практическое применение теории и на методику решения типовых задач. Логическая связь теоретических и практических занятий заключается в том, что информация, полученная на теоретических занятиях, в процессе

самостоятельной работы на практическом занятии осмысливается и перерабатывается, при помощи преподавателя анализируется, после чего усваивается.

Методические рекомендации обучающимся для самостоятельной работы

Самостоятельная работа - это планируемая работа обучающихся, выполняемая по заданию и при методическом руководстве преподавателя, но без его непосредственного участия.

Самостоятельная работа выполняет ряд функций, среди которых особенно выделяются:

- ✓ развивающая (повышение культуры умственного труда, приобщение к творческим видам деятельности, обогащение интеллектуальных способностей, обучающихся);
- ✓ ориентирующая и стимулирующая (процессу обучения придается ускорение и мотивация);
- ✓ воспитательная (формируются и развиваются профессиональные качества специалиста);
- ✓ исследовательская (новый уровень профессионально-творческого мышления);
- ✓ информационно-обучающая (учебная деятельность обучающихся на аудиторных занятиях).

Целью самостоятельных занятий является самостоятельное более глубокое изучение обучающимися отдельных вопросов курса с использованием рекомендуемой дополнительной литературы и других информационных источников.

Задачами самостоятельной работы обучающихся являются:

- ✓ систематизация и закрепление полученных теоретических знаний и практических умений, обучающихся;
- ✓ углубление и расширение теоретических знаний;
- ✓ формирование умения использовать справочную литературу;
- ✓ развитие познавательных способностей и активности обучающихся: творческой инициативы, самостоятельности, ответственности и организованности;

Внеаудиторная самостоятельная работа включает такие формы работы, как:

Индивидуальные занятия (домашние занятия):

- ✓ изучение программного материала дисциплины (работа с учебником и конспектом лекции);
- ✓ изучение рекомендуемых литературных источников;
- ✓ конспектирование источников;
- ✓ выполнение контрольных работ;
- ✓ работа со словарями и справочниками;
- ✓ использование аудио- и видеозаписи;
- ✓ работа с электронными информационными ресурсами и ресурсами Internet;
- ✓ составление плана и тезисов ответа на лекции;
- ✓ составление схем, таблиц, для систематизации учебного материала;
- ✓ выполнение тестовых заданий;
- ✓ решение задач;
- ✓ подготовка презентаций;

- ✓ ответы на контрольные вопросы;
- ✓ аннотирование, реферирование, рецензирование текста;
- ✓ написание эссе, тезисов, докладов, рефератов;
- ✓ составление глоссария, кроссворда по темам дисциплины;
- ✓ ✓ работа с компьютерными программами;
- ✓ подготовка к дифференцированному зачету.

Групповая самостоятельная работа обучающихся:

- ✓ подготовка к занятиям, проводимым с использованием активных форм обучения (круглые столы, деловые игры);
- ✓ анализ деловых ситуаций (мини-кейсов) и др;
- ✓ участие в Интернет - конференциях.

Получение консультаций для разъяснений по вопросам изучаемого учебного предмета.

Но наиболее часто при изучении учебного предмета используются следующие виды самостоятельной работы обучающихся:

- ✓ поиск (подбор) литературы (в том числе электронных источников информации) по заданной теме;
- ✓ подготовка рефератов;
- ✓ подготовка докладов.

Реферат - форма письменной работы. Как правило, реферат является самостоятельным библиографическим исследованием обучающегося, носящим описательно-аналитический характер. Объем реферата может достигать 10-15 страниц. Подготовка реферата подразумевает самостоятельное изучение обучающимся нескольких литературных источников (монографий, научных статей и т.д.) по теме, не рассматриваемой подробно на лекции, систематизацию материала и краткое его изложение. Цель написания реферата - привитие навыков краткого и лаконичного представления собранных материалов и фактов в соответствии с требованиями, предъявляемыми к научным отчетам, обзорам и статьям.

Работа обучающегося над рефератом включает следующие операции:

- ✓ уяснение сути темы (по её названию);
- ✓ выявление литературных источников по данной теме;
- ✓ ознакомление с содержанием источников, направленным на осмысление его внешней и внутренней структуры;
- ✓ работа над выделением главных смысловых компонентов текста (ключевых слов и предложений);
- ✓ отбор наиболее важных сведений из выделенных фрагментов;
- ✓ составление «связок» из отобранного материала в соответствии с логикой изложения фактов;
- ✓ написание реферата.
- ✓ Реферат в структурном отношении должен включать:
- ✓ заголовочную часть (введение);
- ✓ собственно, реферативную часть (изложение основных положений по плану с соответствующими названиями и нумерацией);
- ✓ заключительную часть (выводы);
- ✓ справочную часть (список использованной литературы);
- ✓ оглавление (содержание).

Текст реферата должен иметь связанное, цельное построение. Содержание реферата должна пронизывать главная идея, которая объединяет все информационные элементы источников, а также приводимые факты. При этом допускается сохранение отдельных структурных элементов оригинала. Однако не допустимо простое переписывание положение литературных источников. Язык должен быть кратким, ясным, доступным.

Оформление реферата должно быть выполнено машинописным способом в соответствии со следующими требованиями: **печатный шрифт** – Times New Roman, кегль (размер) 14; **листы бумаги** – формата А4, все страницы должны быть пронумерованы, с размером полей по 2 см; **междустрочный интервал** – полуторный; **абзац –** с отступом первой строки 1,25 см; **текст –** должен быть **выровнен по ширине** и **структурирован** по главам (разделам, параграфам), сопровождаться ссылками на источники при использовании прямых (в кавычках) цитат и фрагментов опубликованных текстов, т.е. на такое использование источников должно быть явным образом указано в сносках или за текстовом списке литературы в соответствии с требованиями ГОСТа.

Реферат оценивается исходя из следующих критериев:

- ✓ поставлена ли цель в работе;
- ✓ сумел ли обучающийся самостоятельно составить логический план к теме и реализовать его;
- ✓ каков научный уровень реферата;
- ✓ собран ли достаточный фактический материал;
- ✓ удалось ли раскрыть тему;
- ✓ показана ли связь рассматриваемой темы с современными проблемами науки и общества, со специальностью обучающегося;
- ✓ каков авторский вклад в систематизацию, структурирование материала, в составлении заключения;
- ✓ достигнута ли цель работы.

Тематика рефератов приводится в рамках тем лекционных занятий. Возможно написание реферата по теме, предложенной самим обучающимся (по согласованию с преподавателем). Реферат на занятиях (научно-практических конференциях) может быть представлен в виде доклада.

Реферат должен быть выполнен за один месяц до начала экзаменационной сессии. Обучающиеся, не представившие в установленный срок реферат, либо получившие оценку «неудовлетворительно», к сдаче экзамена не допускаются.

Углубленному изучению материала по отдельным разделам учебного предмета способствует подготовка обучающимися докладов. Доклад - это самостоятельная работа, анализирующая и обобщающая публикации по заданной тематике, предполагающая выработку и обоснование собственной позиции автора в отношении рассматриваемых вопросов. Подготовка доклада - достаточно кропотливый труд. Его написанию предшествует изучение широкого круга экономических первоисточников, монографий, статей, обобщение личных наблюдений. Работа над докладом способствует развитию самостоятельного, творческого мышления, учит применять экономические знания на практике при анализе актуальных социальных и правовых проблем. Рекомендуемое время доклада - 10-12 минут.

Непременным условием закрепления теоретических и практических знаний, обучающихся является выполнение ими домашних заданий. Одним из основных видов

деятельности обучающегося является самостоятельная работа, которая включает в себя изучение лекционного материала, учебников и учебных пособий, первоисточников, подготовку сообщений, выступления на групповых занятиях, выполнение заданий преподавателя.

Методические рекомендации обучающимся по изучению рекомендованной литературы

Самостоятельная работа обучающихся включает в себя выполнение различного рода заданий, которые ориентированы на более глубокое усвоение материала изучаемого учебного предмета. По каждой теме учебного предмета обучающимся предлагается перечень заданий для самостоятельной работы.

К выполнению заданий для самостоятельной работы предъявляются следующие требования: задания должны исполняться самостоятельно и представляться в установленный срок, а также соответствовать установленным требованиям по оформлению.

Обучающимся следует:

- выполнять все плановые задания, выдаваемые преподавателем для самостоятельного выполнения, и разбирать на консультациях неясные вопросы;
- при подготовке к экзамену параллельно прорабатывать соответствующие теоретические и практические разделы учебного предмета, фиксируя неясные моменты для их обсуждения на плановой консультации.

Для подготовки к занятиям, текущему контролю и промежуточной аттестации обучающиеся могут воспользоваться электронной библиотекой ВУЗа, расположенной по электронному адресу https://www.iprbookshop.ru/, где они имеют возможность получить доступ к учебно-методическим материалам как библиотеки ВУЗа, так и иных электронных библиотечных систем. В свою очередь, обучающиеся могут взять на дом необходимую литературу на абонементе в библиотеке, а также воспользоваться читальным залом.

Методические рекомендации по балльно-рейтинговой системе учета и оценки достижений обучающихся

Общие положения

- Балльно-рейтинговая система учета и оценки достижений, обучающихся (далее – БРС) в Частном образовательном учреждении высшего образования «Московский университет имени А. С. Грибоедова» (далее - Университет) предназначена для решения следующих задач:
- стимулирования обучающихся к регулярной учебной и внеучебной деятельности;
- организации непрерывного контроля освоения обучающимися элементов основных профессиональных образовательных программ среднего профессионального образования;
- комплексной оценки результативности учебной и внеучебной деятельности обучающихся в течение семестра;
- создания для преподавателей, руководства факультетов и кафедр надежного инструментария для осуществления систематического и регулярного контроля за учебной и внеучебной деятельностью обучающихся.

БРС является обязательной для всех участников образовательного процесса и реализуется в электронной информационно-образовательной среде «Электронный университет» (далее – ЭИОС) для всех форм и технологий обучения.

БРС обеспечивает рейтинговый учет и оценку достижений обучающегося за семестр и включает в себя учебный (далее – Rуд) и внеучебный (далее – Rвр) рейтинги.

Учебный рейтинг обучающихся

Перечень дисциплин к изучению в семестре Вы можете увидеть в Личном кабинете в ЭИОС «Электронный Университет» (далее Личный кабинет). Чтобы попасть в Личный кабинет, Вам нужно перейти по ссылке https://impe.lms.mirapolis.ru/mira/ и ввести свой логин и пароль, выданный при поступлении в Университет.

Ваш учебный рейтинг по дисциплине будет формироваться на основе результатов выполнения следующих мероприятий:

- прохождение рубежного контроля (тестирования) по результатам изучения каждого модуля (раздела) дисциплины;
- выполнение рейтинговой работы (домашней творческой работы, расчетноаналитического задания, реферата, контрольной работы, группового проекта) при ее наличии.

Доступ к тестовым заданиям по всем модулям изучаемых дисциплин (рубежному контролю) обеспечивается с начала семестра, однако проходить тестирование необходимо только по окончании изучения каждого модуля (раздела) дисциплины.

Рейтинговая работа должна быть выполнена и передана на проверку преподавателю не позднее, чем за 2 недели до начала экзаменационной сессии через Личный кабинет. Выполнение обучающимся рейтинговой работы допускается не более одного раза в течение установленного календарным учебным графиком периода. Представленная на проверку работа исправлению не подлежит.

При возникновении вопросов по процедуре тестирования, некорректного отображения контента, иных вопросов, связанных с технической поддержкой системы тестирования, обращайтесь в Центр поддержки.

Для прохождения тестирования Вам предоставляются две попытки с ограничением по времени (45 минут). Выполнение тестирования допускается не более 2-ух раз в течение установленного календарным учебным графиком периода. Как итоговый учитывается Ваш лучший результат из двух попыток.

Допуск обучающегося к промежуточной аттестации осуществляется с любым количеством баллов, набранных в ходе выполнения Рейтинговых мероприятий по учебной дисциплине, в том числе с количеством баллов – 0.

Если по результатам выполнения всех рейтинговых мероприятий (тестирования по всем модулям дисциплины и выполнения рейтинговой работы) Вы набрали средний балл - 75 баллов и более, Вы можете претендовать на оценку «хорошо», «отлично» или «зачтено» по итогам работы в семестре без прохождения промежуточной аттестации (т.е. «автоматом»).

Средний балл рассчитывается как сумма баллов за все элементы рейтингового контроля (тестирование по каждому модулю, выполнение рейтинговой работы) разделенная на количество элементов контроля. При этом оценка определяется в соответствии со шкалой:

75-84 балла – хорошо, 85-100 баллов – отлично, 75-100 – зачтено.

Если средний балл менее 75, или Вы претендуете на более высокую оценку, то Вы сдаете зачет/экзамен по дисциплине на общих основаниях.

Промежуточная аттестация **для обучающихся по очной, очно-заочной и заочной формам обучения с применением дистанционных образовательных технологий** осуществляется в форме письменных и/или устных зачетов и выполнения итогового тестирования по экзаменов. дисциплине.

Если студент, обучающийся с применением ЭИОС, претендует на более высокую оценку, чем «автомат», то для открытия итогового тестирования необходимо обратиться в сервис «Центр поддержки».

Для выполнения итогового тестирования предоставляются также две попытки.

Обращаем Ваше внимание на то, что если в течение семестра Вы оформляли отсрочку оплаты обучения, то за две недели до начала зачетно-экзаменационной сессии финансовая задолженность должна быть погашена. В противном случае у Вас не будет возможности получить оценки по дисциплинам «автоматом» или пройти промежуточную аттестацию.

Внеучебный рейтинг обучающихся

Внеучебный рейтинг применяется для активизации обучающихся к участию в творческой, спортивной, общественной и научно-исследовательской деятельности Университета и является составной частью портфолио обучающегося, складывается из баллов, полученных обучающимися за участие в различных внеучебных мероприятиях. Подтверждение суммы баллов за внеучебный рейтинг осуществляется деканами на основе анализа представленных Вами документов (сертификатов, грамот, свидетельств, благодарностей и иных материалов) и заполненного Вами в Личном кабинете портфолио.

Ликвидация академической задолженности

Если в ходе промежуточной аттестации по учебной дисциплине Вы получили неудовлетворительную оценку или не явились на зачет/экзамен (не прошли итоговое тестирование), то у Вас образуется академическая задолженность.

Все набранные ранее в течение семестра по результатам выполнения рейтинговых мероприятий баллы обнуляются.

Ликвидировать академическую задолженность Вы должны в течение года, следующего за семестром образования академической задолженности (но не более двух раз).

Ликвидация академической задолженности проходит в форме выполнения итогового теста. Шкала оценивания при ликвидации академической задолженности:

при форме контроля по дисциплине – **экзамен/ зачет с оценкой: зачет:**

- «неудовлетворительно» – 49 баллов и
- «не зачтено» – 49 баллов и менее; менее;
- «зачтено» – от 50 до 100 баллов
- «удовлетворительно» – от 50 до 65 баллов;
- «хорошо» – от 66 до 84 баллов;
- «отлично» – от 85 до 100 баллов

при форме контроля по дисциплине –

Пройти итоговое тестирование Вы можете не более двух раз в течение установленного календарным графиком периода. Учитывается лучший результат, полученный Вами в ходе выполнения итогового тестирования. Для выполнения одного сеанса тестирования устанавливается время – один академический час (45 минут).