Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Гриб Владислав Валерьевич
Должность: Ректор
Дата подписания: 06.11.2026 17:49:10
Уникальный программный ключ:
637517d24e103c3db032acf37e83...

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.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ 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 среднего общего образования

Форма обучения: очная

Москва

Фонд оценочных средств для 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.

Фонд оценочных средств является неотъемлемой частью рабочей программы профессионального модуля.

Организация-разработчик: Образовательное частное учреждение высшего образования «Московский университет имени А. С. Грибоедова»

Директор Международного дипломатического колледжа_____/Н.И. Платонова

1. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

Настоящий Фонд оценочных средств профессионального модуля по дисциплине «Разработка кода для обучения искусственного интеллекта» является неотъемлемым приложением для профессионального модуля «Разработка кода для обучения искусственного интеллекта». На данный ФОС распространяются все реквизиты утверждения, представленные в РПД по данной дисциплине.

2. ПЕРЕЧЕНЬ ОЦЕНОЧНЫХ СРЕДСТВ

Для определения качества освоения обучающимися учебного материала по дисциплине используются следующие оценочные средства:

№ п/п	Оценочное средство	Краткая характеристика оценочного средства	Представление оценочного средства в ФОС
1	Тестирование	Вид контроля, позволяющий оценить изученный теоретический материал.	Вопросы для проведения тестирования
2	Практические задания	Вид контроля, позволяющий оценить умение обучающегося применять осваиваемую компетенцию в практических ситуациях и при решении производственных задач	Задания к практическому занятию
3	Контрольная работа	Вид контроля, позволяющий определить результат освоения компетенций по дисциплине в рамках рассматриваемой темы, оцениваемый с помощью соответствующих индикаторов достижения компетенций	Задания контрольной работы
4	Самостоятельная работа	Вид контроля, позволяющий оценить проработку теоретического материала, изучение рекомендуемой литературы, выполнение практико-ориентированных заданий (заполнение таблиц, проведение сравнительного анализа, составление схем и др.), решение практических задач, создание презентаций, написание рефератов, подборку нормативного и иного материала и выполнение других заданий	Задания самостоятельной работы
5.	Зачет/Зачет с оценкой/Экзамен	Вид контроля, позволяющий выявить степень овладения знаниями, умениями и навыками, необходимыми для дальнейшего освоения образовательной программы подготовки	Вопросы для подготовки к зачету/зачету с оценкой/экзамену

3.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 _____. +: статическое тестирование	ПК 1.6.
(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