ПРИЛОЖЕНИЕ

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ В.Ф. УТКИНА»

Колледж Рязанского государственного радиотехнического университета
имени В.Ф. Уткина

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

МДК.01.01 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ В СИСТЕМАХ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Специальность	09.02.13 Интеграция решений с применением технологий искусственного интеллекта
Квалификация выпускника	Специалист по работе с искусственным интеллектом
Форма обучения	очная
Год набора	2027 г.

Рязань 2026

1 ОБЩИЕ ПОЛОЖЕНИЯ

Оценочные средства – это совокупность контрольно-оценочных материалов (контрольных заданий, описаний форм и процедур проверки), предназначенных для проведения промежуточной аттестации для объективной оценки качества освоения обучающимися содержания МДК.

Цель – оценить соответствие знаний, умений и владений, приобретенных обучающимся в процессе изучения МДК, целям и требованиям образовательной программы в ходе проведения промежуточной аттестации.

Промежуточная аттестация по МДК предусматривает:

– 2 семестр: зачет с оценкой. Форма проведения зачета - решение задач, теоретические вопросы;

– 3 семестр: курсовой проект, экзамен. Форма защиты курсового проекта – теоретические и практический вопросы. Форма проведения экзамена - решение задач, теоретические вопросы.

2 ПАСПОРТ ОЦЕНОЧНЫХ СРЕДСТВ ПО МДК

Наименование разделов и тем	Формируемые компетенции	Методы оценки
Тема 1.1. Введение в искусственный интеллект и его направления	ПК 1.1, ПК 1.2, ПК 1.3	Зачет с оценкой Экзамен
Тема 1.2. Методы сбора и предобработки данных	ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4	Зачет с оценкой Экзамен
Тема 1.3. Основы алгоритмов машинного обучения	ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4	Зачет с оценкой Экзамен
Тема 1.4. Оценка качества моделей и улучшение алгоритмов	ПК 1.1, ПК 1.2, ПК 1.3	Зачет с оценкой Экзамен
Тема 1.5. Глубокое обучение и нейронные сети	ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4	Зачет с оценкой Экзамен
Тема 1.6. Проектирование ИИ-систем	ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4	Экзамен
Курсовой проект	ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4	Защита курсового проекта

3 ОПИСАНИЕ ПОКАЗАТЕЛЕЙ И КРИТЕРИЕВ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ

Результаты обучения	Критерии оценки результата (показатели освоенности компетенций)	Формы контроля и методы оценки
ПК 1.1 Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием	**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.	Зачет с оценкой Экзамен Защита курсового проекта
ПК 1.2 Разрабатывать программные модули в соответствии с	**Оценка "отлично":** программные модули разработаны в полном соответствии с ТЗ, тесты проходят успешно.	Зачет с оценкой Экзамен Защита курсового

Результаты обучения	Критерии оценки результата (показатели освоенности компетенций)	Формы контроля и методы оценки
техническим заданием	**Оценка "хорошо":** программные модули разработаны с минимальными несоответствиями, тесты в целом успешны. **Оценка "удовлетворительно":** программные модули разработаны с существенными доработками.	проекта
ПК 1.3 Оформлять программный код в соответствии с техническим заданием	**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.	Зачет с оценкой Экзамен Защита курсового проекта
ПК 1.4 Использовать систему контроля версий программного кода с учетом обеспечения возможности организации групповой разработки	**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.	Зачет с оценкой Экзамен Защита курсового проекта

4 ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

4.1 Зачет с оценкой (ПК.1.1 – ПК.1.4)

Перечень вопросов

ПК 1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием

1. Этапы разработки программных модулей для систем искусственного интеллекта.
2. Понятие технического задания (ТЗ) и его структура применительно к ИИ-модулям.
3. Методы и средства проектирования алгоритмов обработки данных.
4. Основные алгоритмические структуры и их реализация в Python.
5. Особенности проектирования алгоритмов для задач машинного обучения (классификация, регрессия, кластеризация).
6. Методика составления тестовых наборов данных для проверки корректности работы модуля.
7. Подготовка данных для обучения моделей: основные этапы и инструменты (на уровне программного модуля).

ПК 1.2. Разрабатывать программные модули в соответствии с техническим заданием

8. Базовые конструкции языка Python и их применение для решения прикладных задач.

9. Обзор специализированных библиотек для ИИ: NumPy, Pandas, Scikit-learn, TensorFlow/PyTorch (назначение и основные функции).

10. Работа с внешними библиотеками и модулями в Python (импорт, установка зависимостей).

11. Методы и средства отладки программных модулей (точки останова, пошаговое выполнение).

12. Использование логирования для поиска ошибок в программном коде.

13. Виды тестирования программного обеспечения: модульное, интеграционное, функциональное.

14. Понятие тестового сценария (test case) и его структура.

15. Инструменты для написания и запуска тестов в Python (unittest, pytest).

16. Особенности отладки и тестирования модулей, использующих модели машинного обучения.

17. Оценка качества программного кода для ИИ-систем: критерии и метрики.

ПК 1.3. Оформлять программный код в соответствии с техническим заданием

18. Стандарты оформления программного кода (PEP 8) и их роль в командной разработке.

19. Документирование программного кода: комментарии, docstring, README.

ПК 1.4. Использовать систему контроля версий программного кода с учетом обеспечения возможности организации групповой разработки

20. Назначение и основные функции системы контроля версий Git.

21. Базовые команды Git: init, add, commit, push, pull.

22. Организация групповой разработки с использованием веток (branch, merge).

23. Разрешение конфликтов слияния в Git.

24. Понятие pull request и его роль в код-ревью.

Перечень заданий

ПК 1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием

1. Написать программу на Python для вычисления площади и периметра геометрических фигур (треугольник, круг, прямоугольник) с обработкой некорректного ввода.

2. Разработать алгоритм и написать программу для нахождения наибольшего общего делителя (НОД) двух чисел с использованием алгоритма Евклида.

3. Написать программу для проверки числа на простоту и вывода всех простых чисел в заданном диапазоне.

4. Разработать программу для перевода чисел из одной системы счисления в другую (двоичная, восьмеричная, десятичная, шестнадцатеричная).

5. Написать программу для вычисления факториала числа с использованием цикла и с использованием рекурсии.

6. Разработать программу для генерации ряда Фибоначчи до заданного пользователем числа.

7. Написать программу для работы со строками: подсчет количества слов, символов, поиск и замена подстрок.

8. Разработать программу для сортировки списка чисел (пузырьковая сортировка и встроенная функция sort()), сравнить время выполнения.

9. Написать программу для работы со словарями: подсчет частоты встречаемости слов в тексте.

10. Разработать программу для работы с кортежами и множествами (поиск уникальных элементов, пересечение, объединение).

ПК 1.2. Разрабатывать программные модули в соответствии с техническим заданием

ПК 1.3. Оформлять программный код в соответствии с техническим заданием

11. Разработать класс Student с полями (ФИО, группа, оценки) и методами (вывод информации, расчет среднего балла).

12. Разработать класс BankAccount с методами пополнения, снятия и проверки баланса (с обработкой исключений).

13. Реализовать принцип наследования: базовый класс Animal и производные классы Dog, Cat с переопределением методов.

14. Реализовать инкапсуляцию: создать класс с приватными полями и методами для доступа к ним (геттеры и сеттеры).

15. Разработать программу с использованием абстрактных классов (модуль abc) для моделирования геометрических фигур.

16. Создать класс Matrix для работы с матрицами (сложение, умножение, транспонирование).

17. Разработать класс Library для управления списком книг (добавление, удаление, поиск по автору или названию).

18. Реализовать паттерн «Одиночка» (Singleton) на примере класса для логирования.

19. Написать программу с использованием магических методов (например, **str**, **repr**, **add**) для пользовательского класса.

20. Разработать иерархию классов для системы обработки заказов (Клиент, Заказ, Товар) с использованием композиции.

21. Написать программу для чтения данных из текстового файла и вывода статистики (количество строк, слов, символов).

22. Разработать программу для записи данных в файл в формате CSV и последующего чтения.

23. Написать программу для работы с JSON-файлами: сериализация и десериализация объектов.

24. Реализовать обработку исключений (try/except/finally) для программы деления чисел с обработкой ошибок ввода и деления на ноль.

25. Создать собственный класс исключения и использовать его для валидации вводимых пользователем данных.

26. Написать программу для копирования содержимого одного файла в другой с обработкой ошибок доступа к файлам.

27. Разработать программу для ведения дневника (добавление записей с датой и временем в файл, чтение записей за определенный день).

28. Написать программу для поиска всех файлов с заданным расширением в указанной директории (модуль os или pathlib).

29. Создать одномерный массив NumPy, выполнить операции сложения, умножения, найти максимальный и минимальный элементы.

30. Создать двумерный массив NumPy (матрицу), выполнить транспонирование и матричное умножение.

31. Используя NumPy, сгенерировать массив случайных чисел и вычислить его среднее значение, медиану и стандартное отклонение.

32. Создать DataFrame в Pandas на основе словаря и выполнить базовые операции (вывод первых/последних строк, информация о типах данных).

33. Загрузить данные из CSV-файла в DataFrame Pandas и вывести описательную статистику (describe).

34. Выполнить фильтрацию данных в DataFrame Pandas по заданному условию (например, выбор студентов с оценкой выше 4).

35. Выполнить группировку данных в Pandas (groupby) и агрегацию (сумма, среднее) по заданному столбцу.

36. Обработать пропущенные значения в DataFrame (заполнение, удаление строк с NaN).

37. Объединить два DataFrame (merge/concat) по заданному ключу.

38. Выполнить сортировку данных в DataFrame по нескольким столбцам.

39. Загрузить встроенный датасет (например, Iris) из Scikit-learn и вывести его характеристики.

40. Разделить датасет на обучающую и тестовую выборки с помощью train_test_split.

41. Обучить модель линейной регрессии на синтетических данных и вывести коэффициенты.

42. Обучить модель логистической регрессии для задачи бинарной классификации.

43. Обучить модель K-ближайших соседей (KNN) и оценить точность (accuracy) на тестовой выборке.

44. Обучить модель дерева решений (Decision Tree) и визуализировать важность признаков.

45. Выполнить масштабирование признаков с помощью StandardScaler перед обучением модели.

46. Оценить качество модели классификации с помощью матрицы ошибок (confusion matrix).

47. Реализовать кросс-валидацию (cross-validation) для оценки стабильности модели.

48. Сохранить обученную модель в файл (pickle или joblib) и загрузить ее обратно.

49. Написать unit-тесты с использованием библиотеки unittest для функции, вычисляющей факториал.

50. Написать тесты с использованием библиотеки pytest для функции, проверяющей палиндром, и запустить их с отчетом о покрытии.

ПК 1.4. Использовать систему контроля версий программного кода с учетом обеспечения возможности организации групповой разработки

51. Инициализировать локальный репозиторий Git в папке с проектом и настроить пользователя (user.name, user.email).

52. Добавить файлы в индекс (git add) и сделать коммит с осмысленным сообщением (git commit -m).

53. Создать новую ветку (git branch), переключиться на нее и внести изменения.

54. Слить ветку с основной (git merge) и разрешить возникший конфликт.

55. Просмотреть историю коммитов (git log) и откатить изменения к предыдущему коммиту (git reset или git revert).

56. Связать локальный репозиторий с удаленным (git remote add) и отправить изменения (git push).

4.2 Экзамен (ПК.1.1 – ПК.1.4)

Перечень вопросов

ПК 1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием

1. Этапы разработки программных модулей для систем искусственного интеллекта.
2. Понятие технического задания (ТЗ) и его структура применительно к ИИ-модулям.
3. Методы и средства проектирования алгоритмов обработки данных.
4. Основные алгоритмические структуры и их реализация в Python.
5. Особенности проектирования алгоритмов для задач машинного обучения (классификация, регрессия, кластеризация).
6. Методика составления тестовых наборов данных для проверки корректности работы модуля.
7. Подготовка данных для обучения моделей: основные этапы и инструменты (на уровне программного модуля).
8. Важность качества данных для ИИ-моделей.
9. Методы сбора данных: веб-скрапинг, API, базы данных.
10. Методы предобработки данных: очистка, нормализация, кодирование категориальных данных, работа с пропусками и выбросами.
11. Подготовка данных для обучения моделей ИИ.
12. Виды обучения: с учителем, без учителя, с подкреплением.
13. Основные алгоритмы машинного обучения: линейная регрессия, логистическая регрессия, kNN, деревья решений, SVM.
14. Кластеризация: k-means и агломеративная кластеризация.
15. Системы рекомендаций: принципы построения и применения.
16. Методы оценки качества моделей: точность, полнота, F-мера, ROC-кривые.
17. Валидация моделей: кросс-валидация, разделение данных на тренировочные и тестовые.
18. Регуляризация моделей: L1 и L2-регуляризация.
19. Оптимизация гиперпараметров моделей.
20. Введение в глубокое обучение и нейронные сети.
21. Архитектуры нейронных сетей: MLP, CNN, RNN.
22. Процессы обучения нейронных сетей: обратное распространение ошибки, стохастический градиентный спуск, функции активации (ReLU, сигмоидальная).
23. Применение нейронных сетей в задачах классификации, распознавания образов и анализа временных рядов.
24. Принципы проектирования архитектуры ИИ-систем: модульность, масштабируемость, эффективность.

ПК 1.2. Разрабатывать программные модули в соответствии с техническим заданием

25. Базовые конструкции языка Python и их применение для решения прикладных задач.
26. Обзор специализированных библиотек для ИИ: NumPy, Pandas, Scikit-learn, TensorFlow/PyTorch (назначение и основные функции).
27. Работа с внешними библиотеками и модулями в Python (импорт, установка зависимостей).
28. Методы и средства отладки программных модулей (точки останова, пошаговое выполнение).
29. Использование логирования для поиска ошибок в программном коде.

30. Виды тестирования программного обеспечения: модульное, интеграционное, функциональное.

31. Понятие тестового сценария (test case) и его структура.

32. Инструменты для написания и запуска тестов в Python (unittest, pytest).

33. Особенности отладки и тестирования модулей, использующих модели машинного обучения.

34. Оценка качества программного кода для ИИ-систем: критерии и метрики.

35. История и основные этапы эволюции искусственного интеллекта.

36. Основные направления ИИ: машинное обучение, глубокое обучение, нейронные сети — их различия и взаимосвязь.

37. Примеры успешного применения ИИ в реальных задачах: распознавание изображений, обработка естественного языка, системы рекомендаций.

38. Внедрение ИИ в реальные проекты.

39. Контейнеризация ИИ-систем с помощью Docker и Kubernetes.

40. Обеспечение безопасности и надежности ИИ-систем.

ПК 1.3. Оформлять программный код в соответствии с техническим заданием

41. Стандарты оформления программного кода (PEP 8) и их роль в командной разработке.

42. Документирование программного кода: комментарии, docstring, README.

ПК 1.4. Использовать систему контроля версий программного кода с учетом обеспечения возможности организации групповой разработки

43. Назначение и основные функции системы контроля версий Git.

44. Базовые команды Git: init, add, commit, push, pull.

45. Организация групповой разработки с использованием веток (branch, merge).

46. Разрешение конфликтов слияния в Git.

47. Понятие pull request и его роль в код-ревью.

Перечень заданий

ПК 1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием

1. Написать программу на Python для вычисления площади и периметра геометрических фигур (треугольник, круг, прямоугольник) с обработкой некорректного ввода.

2. Разработать алгоритм и написать программу для нахождения наибольшего общего делителя (НОД) двух чисел с использованием алгоритма Евклида.

3. Написать программу для проверки числа на простоту и вывода всех простых чисел в заданном диапазоне.

4. Разработать программу для перевода чисел из одной системы счисления в другую (двоичная, восьмеричная, десятичная, шестнадцатеричная).

5. Написать программу для вычисления факториала числа с использованием цикла и с использованием рекурсии.

6. Разработать программу для генерации ряда Фибоначчи до заданного пользователем числа.

7. Написать программу для работы со строками: подсчет количества слов, символов, поиск и замена подстрок.

8. Разработать программу для сортировки списка чисел (пузырьковая сортировка и встроенная функция sort()), сравнить время выполнения.

9. Написать программу для работы со словарями: подсчет частоты встречаемости слов в тексте.

10. Разработать программу для работы с кортежами и множествами (поиск уникальных элементов, пересечение, объединение).

11. Создание базовой модели ИИ для классификации данных.
12. Сбор данных с использованием веб-скрапинга и API.
13. Предобработка данных для машинного обучения: очистка, нормализация, кодирование.
14. Реализация линейной регрессии на реальных данных.
15. Применение кластеризации для сегментации данных.
16. Оценка качества модели с использованием ROC-кривой и F-меры.
17. Настройка гиперпараметров модели с использованием GridSearchCV.
18. Реализация многослойного перцептрона (MLP) для задачи классификации.
19. Создание сверточной нейронной сети для распознавания изображений.
20. Реализация рекуррентной нейронной сети для анализа временных рядов.
21. Проектирование архитектуры ИИ-системы с учетом модульности и масштабируемости.

ПК 1.2. Разрабатывать программные модули в соответствии с техническим заданием

ПК 1.3. Оформлять программный код в соответствии с техническим заданием

22. Разработать класс Student с полями (ФИО, группа, оценки) и методами (вывод информации, расчет среднего балла).
23. Разработать класс BankAccount с методами пополнения, снятия и проверки баланса (с обработкой исключений).
24. Реализовать принцип наследования: базовый класс Animal и производные классы Dog, Cat с переопределением методов.
25. Реализовать инкапсуляцию: создать класс с приватными полями и методами для доступа к ним (геттеры и сеттеры).
26. Разработать программу с использованием абстрактных классов (модуль abc) для моделирования геометрических фигур.
27. Создать класс Matrix для работы с матрицами (сложение, умножение, транспонирование).
28. Разработать класс Library для управления списком книг (добавление, удаление, поиск по автору или названию).
29. Реализовать паттерн «Одиночка» (Singleton) на примере класса для логирования.
30. Написать программу с использованием магических методов (например, str, repr, add) для пользовательского класса.
31. Разработать иерархию классов для системы обработки заказов (Клиент, Заказ, Товар) с использованием композиции.
32. Написать программу для чтения данных из текстового файла и вывода статистики (количество строк, слов, символов).
33. Разработать программу для записи данных в файл в формате CSV и последующего чтения.
34. Написать программу для работы с JSON-файлами: сериализация и десериализация объектов.
35. Реализовать обработку исключений (try/except/finally) для программы деления чисел с обработкой ошибок ввода и деления на ноль.
36. Создать собственный класс исключения и использовать его для валидации вводимых пользователем данных.
37. Написать программу для копирования содержимого одного файла в другой с обработкой ошибок доступа к файлам.
38. Разработать программу для ведения дневника (добавление записей с датой и временем в файл, чтение записей за определенный день).
39. Написать программу для поиска всех файлов с заданным расширением в указанной директории (модуль os или pathlib).

40. Создать одномерный массив NumPy, выполнить операции сложения, умножения, найти максимальный и минимальный элементы.

41. Создать двумерный массив NumPy (матрицу), выполнить транспонирование и матричное умножение.

42. Используя NumPy, сгенерировать массив случайных чисел и вычислить его среднее значение, медиану и стандартное отклонение.

43. Создать DataFrame в Pandas на основе словаря и выполнить базовые операции (вывод первых/последних строк, информация о типах данных).

44. Загрузить данные из CSV-файла в DataFrame Pandas и вывести описательную статистику (describe).

45. Выполнить фильтрацию данных в DataFrame Pandas по заданному условию (например, выбор студентов с оценкой выше 4).

46. Выполнить группировку данных в Pandas (groupby) и агрегацию (сумма, среднее) по заданному столбцу.

47. Обработать пропущенные значения в DataFrame (заполнение, удаление строк с NaN).

48. Объединить два DataFrame (merge/concat) по заданному ключу.

49. Выполнить сортировку данных в DataFrame по нескольким столбцам.

50. Загрузить встроенный датасет (например, Iris) из Scikit-learn и вывести его характеристики.

51. Разделить датасет на обучающую и тестовую выборки с помощью train_test_split.

52. Обучить модель линейной регрессии на синтетических данных и вывести коэффициенты.

53. Обучить модель логистической регрессии для задачи бинарной классификации.

54. Обучить модель K-ближайших соседей (KNN) и оценить точность (accuracy) на тестовой выборке.

55. Обучить модель дерева решений (Decision Tree) и визуализировать важность признаков.

56. Выполнить масштабирование признаков с помощью StandardScaler перед обучением модели.

57. Оценить качество модели классификации с помощью матрицы ошибок (confusion matrix).

58. Реализовать кросс-валидацию (cross-validation) для оценки стабильности модели.

59. Сохранить обученную модель в файл (pickle или joblib) и загрузить ее обратно.

60. Написать unit-тесты с использованием библиотеки unittest для функции, вычисляющей факториал.

61. Написать тесты с использованием библиотеки pytest для функции, проверяющей палиндром, и запустить их с отчетом о покрытии.

62.

63. Анализ примеров успешных решений на основе ИИ.

64. Контейнеризация ИИ-модели с использованием Docker.

65. Развертывание ИИ-системы в Kubernetes.

ПК 1.4. Использовать систему контроля версий программного кода с учетом обеспечения возможности организации групповой разработки

66. Инициализировать локальный репозиторий Git в папке с проектом и настроить пользователя (user.name, user.email).

67. Добавить файлы в индекс (git add) и сделать коммит с осмысленным сообщением (git commit -m).

68. Создать новую ветку (git branch), переключиться на нее и внести изменения.

69. Слить ветку с основной (git merge) и разрешить возникший конфликт.

70. Просмотреть историю коммитов (git log) и откатить изменения к предыдущему коммиту (git reset или git revert).

71. Связать локальный репозиторий с удаленным (git remote add) и отправить изменения (git push).

4.3 Курсовой проект (ПК.1.1 – ПК.1.4)

ПК 1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием

1. Обоснуйте выбор темы курсового проекта и её актуальность для выбранной предметной области.
2. Как сформулирована постановка задачи и какие требования к ИИ-модулю в неё заложены?
3. Какие исходные данные использовались и почему выбран именно этот набор данных?
4. Опишите этапы предобработки данных: очистка, нормализация, кодирование категориальных признаков, обработка пропусков и выбросов.
5. Почему для решения задачи выбран именно этот алгоритм (модель) машинного обучения?
6. Какие виды обучения рассматривались и почему выбран выбранный (с учителем, без учителя, с подкреплением)?
7. Как спроектирована архитектура интеллектуальной модели и чем обоснован её выбор?
8. Какие метрики оценки качества модели выбраны и почему (точность, полнота, F-мера, ROC-кривые)?
9. Как выполнялась валидация модели: разделение данных на тренировочные и тестовые, кросс-валидация?
10. Как проводилась регуляризация и оптимизация гиперпараметров модели?
11. Какие принципы модульности, масштабируемости и эффективности заложены в проектирование ИИ-системы?

ПК 1.2. Разрабатывать программные модули в соответствии с техническим заданием

12. Опишите структуру разработанного программного модуля и назначение его компонентов.
13. Какие библиотеки и инструменты использовались при разработке (NumPy, Pandas, Scikit-learn, TensorFlow/PyTorch и др.) и почему?
14. Как реализован интерфейс пользователя и как он связан с интеллектуальной моделью?
15. Продемонстрируйте работоспособность приложения на тестовых данных.
16. Как реализована обработка исключений и некорректного пользовательского ввода?
17. Какие механизмы логирования и отладки использовались при разработке?
18. Как проводилось тестирование модуля: какие виды тестирования применялись (модульное, интеграционное, функциональное)?
19. Какие тестовые сценарии (test case) разработаны и какие инструменты использовались (unittest, pytest)?
20. Как обеспечена безопасность и надёжность ИИ-системы?
21. Как выполнялась контейнеризация ИИ-модели (Docker) и развертывание системы (Kubernetes), если это предусмотрено темой?

ПК 1.3. Оформлять программный код в соответствии с техническим заданием

22. Соблюдались ли при разработке стандарты оформления кода (PEP 8)? Приведите примеры.
23. Как документирован программный код: комментарии, docstring, README?

24. Опишите структуру пояснительной записки и спецификаций (входных и выходных документов, запросов пользователя).

25. Как оформлены приложения к курсовому проекту и список использованных источников?

26. Как оформление кода и документации влияет на сопровождение и командную разработку проекта?

ПК 1.4. Использовать систему контроля версий программного кода с учетом обеспечения возможности организации групповой разработки

27. Использовалась ли система контроля версий Git при выполнении курсового проекта? Опишите workflow.

28. Какие команды Git применялись (init, add, commit, push, pull, branch, merge)?

29. Как организована групповая разработка: использовались ли ветки, pull request, код-ревью?

30. Возникали ли конфликты слияния и как они разрешались?

31. Как связан локальный репозиторий с удалённым и как обеспечивается сохранность истории изменений?

32. Как откатывались изменения к предыдущим версиям (git reset, git revert)?

Оператор ЭДО ООО "Компания "Тензор"

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ

СОГЛАСОВАНО ЗАМ. ДИРЕКТОРА ПО УР — **ФГБОУ ВО "РГРТУ", РГРТУ,** Савельева Ольга Викторовна, Зам. директора РССК «РГРТУ» по УР — Простая подпись

СОГЛАСОВАНО ДИРЕКТОРОМ РССК «РГРТУ» — **ФГБОУ ВО "РГРТУ", РГРТУ,** Ушаков Сергей Анатольевич, Директор колледжа РГРТУ — Простая подпись