

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Рязанский государственный радиотехнический университет имени В.Ф. Уткина»**

КАФЕДРА «ЭЛЕКТРОННЫЕ ВЫЧИСЛИТЕЛЬНЫЕ МАШИНЫ»

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

«Методологии разработки решений на основе ИИ»

Направление подготовки

02.03.03 Математическое обеспечение и администрирование
информационных систем

Профиль

«Программное обеспечение компьютерных технологий и систем
искусственного интеллекта»

Квалификация (степень) выпускника — бакалавр

Форма обучения — очная

1 ОБЩИЕ ПОЛОЖЕНИЯ

Оценочные материалы – это совокупность учебно-методических материалов (практических заданий, описаний форм и процедур проверки), предназначенных для оценки качества освоения обучающимися данной дисциплины как части ОПОП.

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

Основная задача – обеспечить оценку уровня сформированности компетенций, закрепленных за дисциплиной.

Контроль знаний обучающихся проводится в форме промежуточной аттестации. Промежуточная аттестация проводится в форме зачета.

Форма проведения зачета – тестирование, письменный опрос по теоретическим вопросам.

2 ОПИСАНИЕ ПОКАЗАТЕЛЕЙ И КРИТЕРИЕВ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ

Сформированность каждой компетенции (или ее части) в рамках освоения данной дисциплины оценивается по трехуровневой шкале:

- 1) пороговый уровень является обязательным для всех обучающихся по завершении освоения дисциплины;
- 2) продвинутый уровень характеризуется превышением минимальных характеристик сформированности компетенций по завершении освоения дисциплины;
- 3) эталонный уровень характеризуется максимально возможной выраженностью компетенций и является важным качественным ориентиром для самосовершенствования.

Уровень освоения компетенций, формируемых

дисциплиной: Описание критериев и шкалы оценивания

тестирования:

Шкала оценивания	Критерий
3 балла (эталонный уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 85 до 100%
2 балла (продвинутый уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 70 до 84%
1 балл (пороговый уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 50 до 69%
0 баллов	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 0 до 49%

Описание критериев и шкалы оценивания теоретического вопроса:

Шкала оценивания	Критерий
3 балла (эталонный уровень)	выставляется студенту, который дал полный ответ на вопрос, показал глубокие систематизированные знания, смог привести примеры, ответил на дополнительные вопросы преподавателя
2 балла (продвинутый уровень)	выставляется студенту, который дал полный ответ на вопрос, но на некоторые дополнительные вопросы преподавателя ответил только с помощью наводящих вопросов

1 балл (пороговый уровень)	выставляется студенту, который дал неполный ответ на вопрос в билете и смог ответить на дополнительные вопросы только с помощью преподавателя
0 баллов	выставляется студенту, который не смог ответить на вопрос

На промежуточную аттестацию (зачет) выносится тест, два теоретических вопроса. Максимально студент может набрать 6 баллов. Итоговый суммарный балл студента, полученный при прохождении промежуточной аттестации, переводится в традиционную форму по системе «зачтено», «не зачтено».

Оценка «зачтено» выставляется студенту, который набрал в сумме не менее 4 баллов (выполнил одно задание на эталонном уровне, другое – не ниже порогового, либо оба задания выполнит на продвинутом уровне). Обязательным условием является выполнение всех предусмотренных в течение семестра практических и лабораторных работ заданий.

Оценка «не зачтено» выставляется студенту, который набрал в сумме менее 4 баллов, либо имеет к моменту проведения промежуточной аттестации несданные практические, либо лабораторные работы.

3 ПАСПОРТ ОЦЕНОЧНЫХ МАТЕРИАЛОВ ПО ДИСЦИПЛИНЕ

Контролируемые разделы (темы) дисциплины	Код контролируемой компетенции (или её части)	Вид, метод, форма оценочного мероприятия
Раздел 1. Основные понятия и цели дисциплины. Среды разработки ПО и их основные функции	ПК-1.1	Зачет
Раздел 2. Системы контроля версий. Общие принципы работы. Централизованные и распределенные системы	ПК-1.1, ПК-1.2	Зачет
Раздел 3. Система контроля версий GIT. Основы работы, ветвление, работа с удаленными репозиториями	ПК-1.1, ПК-1.2	Зачет
Раздел 4. Системы отслеживания ошибок, средства автоматизации тестирования	ПК-1.1, ПК-1.2	Зачет
Раздел 5. Виртуализация. Принципы, преимущества, типы виртуализации	ПК-1.1	Зачет
Раздел 6. Контейнеризация. Применение Docker для разработки решений на основе ИИ	ПК-1.1, ПК-1.2	Зачет
Раздел 7. Системы управления проектами (Jira, Microsoft Project)	ПК-9.1	Зачет
Раздел 8. Применение технологий обработки данных в разработке	ПК-13.1, ПК-13.2	Зачет

решений на основе ИИ		
----------------------	--	--

4 ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ

Промежуточная аттестация в форме зачета

Код компетенции	Результаты освоения ОПОП Содержание компетенций
ПК-1	Способен проектировать программное обеспечение с использованием современных инструментальных средств

ПК-1.1. Проектирует и разрабатывает программное обеспечение
Знать Место и роль инструментальных средств при проектировании и разработке ПО
Уметь Использовать инструментальные средства при проектировании и разработке ПО
Владеть Навыками практического использования при проектировании и разработке ПО

1. Что такое система контроля версий? (выберите один вариант ответа)

- а) Программное обеспечение, призванное избавить программиста от необходимости загружать проект к себе на машину
- б) Программное обеспечение, призванное автоматизировать работу с историей файла, и организовать защищенное хранилище проекта
- в) Программное обеспечение, позволяющее версионировать проект

2. Что такое репозиторий? (выберите один вариант ответа)

- а) Место, где система контроля версий хранит все документы вместе с их историей
- б) Директория для вашего проекта
- в) Рабочая копия документов

3. Соедините описание системы контроля версий с её названием (соединить)

Локальные	СКВ, в которых вся работа производится с центральным хранилищем (все действия, так или иначе, зависят от него)
Централизованные	СКВ, главной парадигмой которых является локализация данных на машине каждого разработчика проекта
Децентрализованные	СКВ, хранящиеся только на локальном компьютере

4. Объедините СКВ с её типом (соединить)

GIT	Локальная
SVN	Распределённая и централизованная

Mercurial	Централизованная
CVS	Децентрализованная
Bazaar	Распределённая и локальная

5. К какому виду систем контроля версий относится Subversion? (выберите один вариант ответа)

- а) Централизованному
- б) Распределенному в)

Локальному

- г) Всем вышеперечисленным

6. Соотнесите состояние хранилища и то, какие действия предпримет Subversion во время коммита (соединить)

Хранилище изменялось локально и не устарело	Система не сделает ничего, вам необходимо внести изменения и сделать коммит
---	---

Хранилище не изменялось и устарело	Никаких
------------------------------------	---------

Хранилище изменялось локально и устарело	Заберет изменения с сервера
--	-----------------------------

Хранилище не изменялось и не устарело	Система забирает к себе изменения с сервера и пользователю нужно разрешить конфликты, если необходимо
---------------------------------------	---

7. Соотнесите модель с её описанием (соединить)

Блокирование Изменение Разблокирование	Оба пользователя копируют файл и редактируют его на своём компьютере. Затем один из пользователей отправляет изменения на сервер. Второй пользователь заканчивает свои изменения и тоже хочет отправить файлы на сервер, но не может, т.к. там уже находится обновленная версия. Он должен сначала взять изменения с сервера, объединить изменения, разрешить конфликты, если они есть, и затем отправить свои изменения на сервер.
--	---

Копирование — Изменение Слияние	Пользователь, решивший редактировать файл, блокирует его, в то время другой пользователь не может его редактировать. При
---------------------------------	--

разблокировке файла следующим
участником другой пользователь
может забрать изменения
предыдущего пользователя и
редактировать файл дальше.

8. Отметьте, какие команды вы можете использовать для получения подсказок в Subversion (выберите несколько вариантов ответа)

- а) svn help
- б) svn resolve
- в) svn commit
- г) svn update
- д) svn log
- е) svn info
- ж) svn revert
- з) svn delete

9. Какие команды вы можете использовать для создания/получения репозитория? (выберите несколько вариантов ответа)

- а) svnadmin create /path/to/rep
- б) svn import mytree file:///usr/local/svn/newrepos/some/project \-m "Initial import"
- в) svn checkout http://svn.example.com/repos/calc

10. Отметьте команды для внесения изменений в рабочую копию (выберите несколько вариантов ответа)

- а) svn add
- б) svn delete
- в) svn copy
- г) svn move
- д) svn mkdir
- е) svn update

11. Объедините способы разрешения конфликтов с их описанием (соединить)

- 1) Объединить конфликтный файл:
Конфликтный файл будет выглядеть так:
<<<<<<< имя файла ваши
изменения
=====
результат автоматического слияния с
репозиторием
>>>>>>> ревизия
- 2) Выполнить "svn resolved"

Отмена ваших изменений

1) Выполнить "svn revert file"	Объединение вручную
Скопировать filename.rOLDREV или filename.rNEWREV	Копирование одного из файлов (своего или чужого)

12. Вопрос: верно ли, что в Git и Subversion используются принципиально разные подходы к хранению файлов репозитория? (выберите один вариант ответа)

а) да

б) нет

13. Вопрос: каковы были основные цели, преследуемые при создании Git? (выберите один вариант ответа)

а)

1. Простая архитектура.
2. Полная децентрализация.
3. Хорошая поддержка нелинейной разработки.

б)

1. Высокая скорость работы.
2. Применение подхода CVS.
3. Поддержка нелинейной разработки.

в)

1. Простая архитектура.
2. Централизованный подход.
3. Поддержка нелинейной разработки.

14. Под какой лицензией выпущена системы контроля версий Git? (краткий ответ)

15. Какой алгоритм использует Git для вычисления хэш-сумм? (выберите один вариант ответа)

а) CRC32

б) MD6

в) SHA-1

г) SHA-2

16. Что такое ветка в Subversion? (выберите один вариант ответа)

а) Направление разработки, которое существует независимо от другого направления

б) Направление разработки, которое существует независимо от другого направления, однако имеющие с ним общую историю

в) Копия репозитория у любого из разработчиков

17. В каких случаях есть необходимость создать ветку? (выберите

несколько вариантов ответа) а) Изменения, которые вы хотите внести,

могут повредить работающему коду б) Вы хотите написать
улучшение/оптимизировать существующий код в) Вы хотите
зафиксировать изменения

18. Выберите верные сведения о создании ветвей в Subversion (выберите несколько вариантов ответа)

- а) При использовании команды `svn copy <удаленный url> <удаленный url> -m "Сообщение коммита"`, вы создаёте ветку на удалённом сервере, а не на локальной машине
- б) При использовании команды `svn copy <удаленный url> <удаленный url> -m "Сообщение коммита"`, вы создаёте ветку на локальной машине
- в) При использовании команды `svn copy <удаленный url> <удаленный url> -m "Сообщение коммита"`, вам не обязательно иметь рабочую копию
- г) При использовании команды `svn copy <локальная директория> <локальная директория>` вы создаёте ветку на локальной машине
- д) При использовании команды `svn copy <локальная директория> <локальная директория>` вы создаёте ветку на удаленном сервере, а не на локальной машине

19. При создании ветки в Git командой `git branch <branchname>`, происходит ли автоматический переход на эту ветку? (выберите один вариант ответа)

- а) да
- б) нет

20. Что происходит при команде `git checkout <branchname>` (выберите один вариант ответа)

- а) Мы переходим в директорию `branchname`
- б) Указатель HEAD перемещается на ветку `branchname`
- в) Мы создаём ветку `branchname` и перемещаемся на неё

21. По каким протоколам можно настроить работу Git на сервере? (выберите несколько вариантов ответа)

- а) HTTP
- б) FTP
- в) SSH
- г) Telnet
- д) Git
- е) SMTP

22. Что вам нужно сделать с проектом в случае «вы хотите внести изменения в репозиторий, в который у вас нет доступа»? (выберите один вариант ответа)

- а) fork
- б) pull request

23. Выберите случаи, когда вам может потребоваться использовать GitHub (выберите несколько вариантов ответа) а) создать свой проект с открытым исходным кодом / использовать

как хранилище кода, который не хотелось бы потерять

б) воспользоваться сторонней библиотекой (не входящей в список стандартных)

в) внести вклад в уже существующий проект на GitHub

ПК-1.2. Применяет современные инструментальные средства при разработке программного обеспечения	
Знать	Особенности современных инструментальных средств, применяемых при разработке программного обеспечения
Уметь	Применять инструментальные средства на всех этапах проектирования программного обеспечения
Владеть	Навыками использования инструментальных средств при разработке, тестировании и анализе программного обеспечения

39. Соедините подкатегорию виртуализации с ее описанием (соединить)

виртуализация платформ

Данный вид виртуализации преследует своей целью комбинирование или упрощение представления аппаратных ресурсов для пользователя и получение неких пользовательских абстракций оборудования, пространств имен, сетей и т.п

виртуализация ресурсов

Продуктом этого вида виртуализации являются виртуальные машины некие программные абстракции, запускаемые на платформе реальных аппаратнопрограммных систем

40. Какие подвиды включает виртуализация платформ? (выберите несколько вариантов ответа)

- а) Разделение ресурсов
- б) Инкапсуляция
- в) Частичная виртуализация
- г) Паравиртуализация
- д) Виртуализация уровня операционной системы
- е) Полная эмуляция (симуляция).
- ж) Частичная эмуляция (нативная виртуализация).
- з) Виртуализация уровня приложений

41. Какие подвиды включает виртуализация ресурсов? (выберите несколько вариантов ответа)

- а) Разделение ресурсов
- б) Инкапсуляция
- в) Частичная виртуализация
- г) Паравиртуализация
- д) Виртуализация уровня операционной системы
- е) Кластеризация компьютеров и распределенные вычисления (grid computing)
- ж) Частичная эмуляция (нативная виртуализация).

з) Виртуализация уровня приложений

и) Объединение, агрегация и концентрация компонентов

42. Что такое контейнеризация? (выберите один вариант ответа)

- а) Подход к разработке программного обеспечения, при котором приложение или служба, их зависимости и конфигурация упаковываются вместе в образ контейнера
- б) Абстракция вычислительных ресурсов и предоставление пользователю системы, которая «инкапсулирует» (скрывает в себе) собственную реализацию
- в) Набор изолированных приложений, не взаимодействующих друг с другом

43. Преимущества контейнеризации перед виртуализацией (выберите несколько вариантов ответа)

- а) Более простая настройка контейнеров
- б) Контейнеры требуют гораздо меньше ресурсов
- в) Контейнеры легко развертывать
- г) Контейнеры быстрее запускаются
- д) Приложение в контейнере выполняется в любой среде

44. Соедините термин с его определением (соединить)

Контейнер	пакет со всеми зависимостями и сведениями, необходимыми для создания контейнера
-----------	---

Образ контейнера	действие по созданию образа контейнера на основе сведений и контекста, предоставленных файлом Dockerfile, а также дополнительных файлов в папке, где создается образ
------------------	--

Сборка	текстовый файл, содержащий инструкции по сборке образа Docker
--------	---

Dockerfile	экземпляр образа Docker
------------	-------------------------

45. Какие компоненты входят в каждый контейнер Docker? (выберите несколько вариантов ответа)

- а) Выбранная операционная система (например, дистрибутив Linux, Windows Nano Server или Windows Server Core)
- б) Файлы, добавленные разработчиком (двоичные файлы приложения и т. п.)
- в) Сведения о конфигурации (параметры среды и зависимости)
- г) Тесты для приложения

Типовые вопросы открытого типа

1. Запишите команду GIT, позволяющую создать пустой репозиторий (git init)
2. Запишите команду GIT, позволяющую клонировать существующий репозиторий (git clone <путь>)
3. Запишите команду GIT, позволяющую выполнить первоначальную настройку репозитория (git config)
4. Запишите команду GIT, позволяющую проиндексировать изменения в рабочей директории (git add)
5. Запишите команду GIT, позволяющую зафиксировать изменения в локальный репозиторий (git commit)
6. Запишите команду GIT, позволяющую зафиксировать изменения в удаленный репозиторий (git push)
7. Запишите команду GIT, позволяющую получить данные из удаленного репозитория (git pull)
8. Запишите команду GIT, позволяющую создать ветку (git branch <имя ветки>)
9. Запишите команду GIT, позволяющую переключиться на созданную ветку (git checkout <имя ветки>)
10. Запишите команду GIT, позволяющую присоединить метку к текущему коммиту (git tag)
11. Запишите команду GIT, позволяющую выполнить слияние двух веток (git merge)
12. Запишите команду GIT, позволяющую просмотреть историю коммитов (git log)
13. Запишите команду GIT, позволяющую просмотреть проиндексированные изменения (git diff staged)
14. Запишите команду GIT, позволяющую просмотреть выполненные локальные изменения (git diff)
15. Запишите команду GIT, позволяющую просмотреть справку по интересующей команде (git help <команда>)

Код компетенции	Результаты освоения ОПОП Содержание компетенций
ПК-8	Способен применять языки программирования C/C++ для решения задач в области ИИ

ПК-8.1. Проводит анализ рынка программного обеспечения и научно-технической информации

Знать Инструментальные средства, применяемые при разработке и тестировании программного обеспечения Уметь Осуществлять обоснованный выбор средств и инструментов разработки Владеть Практическими навыками использования современных инструментальных средств

30. Что такое тестирование? (выберите несколько вариантов ответа)

- а) Одна из техник контроля качества, включающая в себя активности по планированию работ, проектированию тестов, выполнению тестирования и анализу полученных результатов
- б) Работа тестировщика
- в) Проверка соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов, выбранном определенным образом

31. Выберите виды тестов, входящие в пирамиду тестирования (Майк Кон) (выберите несколько вариантов ответа)

- а) юнит-тесты
- б) интеграционные тесты
- в) тесты корректности работы операционной системы, на которой будет установлено приложение
- г) тесты пользовательского интерфейса
- д) тесты правильности сборки аппаратной платформы компьютера

32. Выберите положения, правильные для юнит тестирования (выберите несколько вариантов ответа)

- а) Должны не зависеть от окружения, на котором они выполняются
- б) Запускаться регулярно в автоматическом режиме
- в) Должны выполняться под специально настроенным окружением
- г) Должны запускаться вручную для контроля за ними

33. Соедините методологию тестирования и ее описание (соединить)

Тестирование черного ящика	метод тестирования программного обеспечения, который предполагает, что внутренняя устройство системы известны тестировщику
Тестирование методом серого ящика	метод тестирования, базируется только лишь на тестировании по функциональной спецификации и требованиям, при этом не имея доступа во внутреннюю структуру кода и базу данных.
Тестирование методом белого ящика	Метод тестирования ПО, который предполагает, что внутреннее устройство программы нам известно лишь частично

34. Расставьте в нужном порядке последовательность действий при разработке через

11	1-2, 2-1, 3-3	44	1-4, 2-1, 3-2, 4-3
12	а	45	а, б, в
13	а		
14	GNU GPL 2		
15	в		
16	б		
17	а, б		
18	а, в, г		
19	б		
20	б		
21	а, в, д		
22	б		
23	а, б, в		
24	б		
25	б		
26	а, б, г		
27	в, а, е, б, з, ж, г, и, д		
28	Все кроме «г»		
29	а, б, г, д, е		
30	а, в		
31	а, б, г		
32	а, б		
33	1-2 , 2-3, 3-1		

Код компетенции	Результаты освоения ОПОП Содержание компетенций
ПК-13	Способен применять различные модели и (или) технологии обработки данных

ПК-13.1. Осуществляет выбор технологий обработки больших данных, приемлемых для создания прикладной системы ИИ с заданными требованиями

Знать

Технологии организации обработки больших данных

Уметь

Обоснованно выбирать необходимые инструментальные средства для конкретной задачи обработки данных

Владеть

Навыками конфигурирования и сопровождения сложных распределенных систем обработки данных и машинного обучения

Типовые тестовые вопросы

1. **Какой инструмент позволяет создавать воспроизводимые среды выполнения для ИИ-моделей?**

- а) Virtual PC
- б) VMware
- в) **Docker**
- г) Subversion

2. **Какой вид виртуализации наиболее подходит для объединения вычислительных ресурсов нескольких серверов?**

- а) Виртуализация платформ
- б) **Виртуализация ресурсов**
- в) Паравиртуализация
- г) Виртуализация уровня приложений

3. **Что из перечисленного относится к виртуализации ресурсов?**

- а) Создание виртуальных машин
- б) Эмуляция аппаратного обеспечения
- в) **Кластеризация компьютеров**
- г) Запуск нескольких ОС на одном сервере

4. **Основное преимущество контейнеризации перед виртуализацией для обработки данных ИИ:**

- а) Полная изоляция операционных систем
- б) **Меньшее потребление ресурсов и быстрое развертывание**
- в) Возможность запуска разных ОС
- г) Лучшая безопасность

5. **Какой компонент DevOps обеспечивает автоматизацию развертывания ИИ-моделей?**

- а) Система контроля версий
- б) **CI/CD пайплайны**
- в) Канбан-доски
- г) Системы отслеживания ошибок

6. **Для обработки потоковых данных в ИИ-системе наиболее подходит:**

- а) Реляционная БД
- б) **Apache Kafka**
- в) Файловое хранилище
- г) Локальная БД

7. Какой инструмент используется для оркестрации контейнеров в распределенных системах?

- а) Docker Compose
- б) **Kubernetes**
- в) Jenkins
- г) GitLab

8. Что характеризует качественный набор данных для обучения ИИ-модели?

- а) Только числовые данные
- б) Маленький размер
- в) **Репрезентативность и качественная разметка**
- г) Однородность данных

9. Какой процесс обеспечивает воспроизводимость экспериментов с ИИ-моделями?

- а) **Версионирование данных и кода**
- б) Ежедневные бэкапы
- в) Ручное документирование
- г) Периодическое тестирование

10. Для хранения больших наборов данных обучения в облаке оптимально использовать:

- а) Блочное хранилище
- б) **Объектное хранилище**
- в) Оперативную память
- г) Локальные диски

Типовые вопросы открытого типа

1. Бесплатная, открытая платформа для управления жизненным циклом машинного обучения. (**MLflow**)

2. Распределённый фреймворк с открытым исходным кодом для обработки больших данных, известный своими возможностями обработки в оперативной памяти. (**Apache Spark**)

3. Популярная распределенная система обмена сообщениями, часто используемая для построения конвейеров данных реального времени. (**Apache Kafka**)

4. Общепринятая аббревиатура процесса "Извлечение, Преобразование, Загрузка" данных. (**ETL**)

ПК-13.2. Разрабатывает и отлаживает прикладные решения с элементами ИИ с применением различных технологий обработки данных

Знать

Методологии и инструментальные средства, применяемые при тестировании и отладке программного обеспечения

Уметь

Применять разнообразные технологии обработки данных для создания эффективных решений ИИ

Владеть

Навыками выбора подходящей технологии обработки данных в зависимости от поставленной задачи

Типовые тестовые вопросы

1. **Основной инструмент для контроля версий датасетов и моделей в ИИ-проектах:**
 - а) SVN
 - б) **Git**
 - в) CVS
 - г) Mercurial
2. **Для отладки проблем с данными в ИИ-модели в первую очередь следует проверить:**
 - а) Код нейросети
 - б) **Качество и предобработку данных**
 - в) Настройки гиперпараметров
 - г) Серверное оборудование
3. **Какой метод тестирования проверяет отдельные модули ИИ-системы?**
 - а) Интеграционное тестирование
 - б) **Юнит-тестирование**
 - в) Системное тестирование
 - г) Приемочное тестирование
4. **Инструмент для отслеживания экспериментов и метрик ИИ-моделей:**
 - а) Jira
 - б) **MLflow**
 - в) Jenkins
 - г) Docker
5. **Для мониторинга дрейфа данных (data drift) в работающей ИИ-модели используют:**
 - а) **Статистический анализ распределений**
 - б) Ежедневное переобучение
 - в) Увеличение логгирования
 - г) Ручную проверку
6. **Техника обработки пропущенных значений в данных для ИИ-модели:**
 - а) Удаление строк
 - б) **Импутация**
 - в) Игнорирование проблемы
 - г) Замена нулями
7. **Процесс преобразования сырых данных в признаки для ИИ-модели:**
 - а) Валидация
 - б) **Feature Engineering**
 - в) Нормализация
 - г) Аугментация
8. **Методология, обеспечивающая быстрое выявление ошибок в ИИ-системе:**
 - а) Waterfall
 - б) **Agile**

- в) RUP
- г) Spiral

9. **Инструмент для автоматического тестирования ИИ-пайплайнов:**

- а) **Jenkins/GitLab CI**
- б) Excel
- в) Word
- г) PowerPoint

10. **Для отладки производительности ИИ-модели следует анализировать:**

- а) **Время инференса и использование памяти**
- б) Цвет интерфейса
- в) Размер лог-файлов
- г) Количество комментариев в коде

Типовые вопросы открытого типа

1. Процесс автоматического развертывания и управления ML-моделями в промышленную среду. (**MLOps**)
2. Популярный фреймворк с открытым исходным кодом для оркестрации контейнеров, часто используемый для развертывания ML-пайплайнов. (**Kubernetes**)
3. Процесс преобразования необработанных данных в удобный для машинного обучения формат. (**Предобработка данных / Feature Engineering**)
4. Тип мониторинга, который отслеживает изменение распределения входных данных работающей ML-модели с течением времени. (**Data Drift Monitoring**)

Типовые теоретические вопросы:

1. Системы управления версиями. Основные понятия.
2. Системы управления версиями. Сущность, назначение, преимущества.
3. Системы управления версиями. Виды и особенности.
4. Системы управления версиями. Основные представители на рынке.
5. Системы управления версиями. Модели версионирования. Достоинства и недостатки.
6. Git. Общие сведения, история создания.
7. Git. Основные понятия и определения.
8. Git. Особенности хранения изменения и организации хранилища. Целостность в Git.
9. Git. Жизненный цикл файла.
10. Git. Локальное выполнение операций.
11. Git. Первоначальная настройка репозитория.
12. Git. Жизненный цикл коммитов в Git.
13. Git. Создание репозитория, клонирование репозитория.
14. Git. Изменение файлов, запись изменений в репозиторий, просмотр изменений, отправка изменений на удаленный сервер.
15. Git. Механизм меток. Виды меток.
16. Git. Игнорирование файлов. Составление файла игнорирования.
17. Git. Ветвление. Понятие ветки, суть работы в ветках.
18. Git. Способы создания веток. Работа с ветками.
19. Git. Копирование изменений между ветками.
20. Git. Конфликты при слиянии. Способы разрешения конфликтов.
21. Git. Совместная серверная работа.
22. Git. Модель ветвления git flow.
23. Git. Поиск ломающего коммита методом половинного деления.
24. Современные технологии обработки данных для ИИ-приложений: классификация и характеристики
25. Этапы жизненного цикла разработки приложений с искусственным интеллектом
26. Методы тестирования и отладки решений с элементами ИИ
27. Применение контейнеризации для инкапсуляции и развертывания моделей машинного обучения
28. Системы отслеживания ошибок. Назначение, основные преимущества.
29. Системы отслеживания ошибок. Состав информации о дефекте.
30. Системы отслеживания ошибок. Жизненный цикл дефекта.
31. Системы отслеживания ошибок. Основные представители на рынке. Тестирование программного обеспечения. Основные понятия, история тестирования.
32. Тестирование программного обеспечения. Пирамида тестирования.
33. Тестирование программного обеспечения. Модульное тестирование.
34. Тестирование программного обеспечения. Интеграционное тестирование.
35. Тестирование программного обеспечения. Тестирование UI.

36. Тестирование программного обеспечения. Виды тестирования.
37. Тестирование программного обеспечения. Тестирование методом чёрного ящика.
38. Тестирование программного обеспечения. Тестирование методом белого ящика.
39. Тестирование программного обеспечения. Тестирование методом серого ящика.
40. Тестирование программного обеспечения. Разработка через тестирование.
41. Тестовый фреймворк Test NG. Общие сведения и назначение.
42. Тестовый фреймворк Test NG. Автоматическая сборка. Особенности, преимущества.
43. Система автоматической сборки Gradle.
44. Тестовый фреймворк Test NG. Команды группы Assert.
45. Тестовый фреймворк Test NG. Аннотации.
46. Непрерывная интеграция и развертывание (CI/CD). Понятие и назначение.
47. CI/CD. Требования к проекту, организация CI. Преимущества и недостатки.
48. CI/CD. Непрерывная доставка. Общая схема CI/CD.
49. CI/CD. Основные принципы.
50. Devops. Понятие, области применения. Цели Devops.
51. Взаимодействие CI/CD и Devops.
52. Devops. Преимущества.
53. Виртуализация. Понятие, назначение.
54. Виртуализация. Виды виртуализации.
55. Виртуализация. Виды виртуализации платформ.
56. Виртуализация. Виды виртуализации ресурсов.
57. Виртуализация. Области применения, современное состояние.
58. Контейнеризация. Понятие, назначение, преимущества.
59. Контейнеризация. Docker. Назначение, терминология.
60. Контейнеризация. Рабочий процесс разработки приложений.