МИНИСТЕРСТВО ОБРАЗОВАНИЯ СВЕРДЛОВСКОЙ ОБЛАСТИ
ГАПОУ СО
«НИЖНЕТАГИЛЬСКИЙ ТОРГОВО-ЭКОНОМИЧЕСКИЙ КОЛЛЕДЖ»

Утверждаю:
Зам. директора по УР
_____ М.Г. Дмитриева
«30» июня 2026 г.

КОМПЛЕКТ ОЦЕНОЧНЫХ СРЕДСТВ

«Основы алгоритмизации и программирования»

(ОП.07)

по специальности
09.02.12 Техническая эксплуатация и сопровождение информационных систем

Нижний Тагил 2026

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Комплект оценочных средств составлен в соответствии с требованиями федерального государственного образовательного стандарта среднего профессионального образования и примерной основной образовательной программы по специальности 09.02.12 Техническая эксплуатация и сопровождение информационных систем.

Организация-разработчик:

ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Разработчики:

Репьев Егор Денисович, преподаватель первой квалификационной категории ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Ильина Виктория Денисовна, преподаватель без квалификационной категории ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

СОДЕРЖАНИЕ

5.1 ПАСПОРТ КОМПЛЕКТА КОНТРОЛЬНО-ОЦЕНОЧНЫХ СРЕДСТВ.....4
5.1.1 КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ ДИСЦИПЛИНЫ.....4
5.1.2 ФОРМЫ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ.....6
5.1.3 ОПИСАНИЕ ПРОЦЕДУРЫ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ.....6
5.1.4 КРИТЕРИИ ОЦЕНКИ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ.....8
5.2 КОМПЛЕКТ «ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ».....10
5.2.1 Вопросы к экзамену.....10
5.2.2 Типовые практические задания.....13
5.2.3 Задания для экзамена.....14
5.3 КОМПЛЕКТ «ТЕКУЩИЙ КОНТРОЛЬ».....32
5.3.1 Задания для срезов знаний.....32
5.3.2 Тестовые задания.....32
5.3.3 Задания для проверочных, самостоятельных и контрольных работ.....38
5.3.4 Лабораторные и практические работы.....53
5.3.5 Компьютерные программы для контроля знаний студентов.....57

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

5.1 ПАСПОРТ КОМПЛЕКТА КОНТРОЛЬНО-ОЦЕНОЧНЫХ СРЕДСТВ

5.1.1 КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Предметом оценки освоения учебной дисциплины (УД) являются умения и знания. Контроль и оценка этих дидактических единиц осуществляются с использованием следующих форм и методов:

Таблица 1 – Формы и методы контроля и оценки дидактических единиц

Результаты обучения (освоенные умения, усвоенные знания)	Формы и методы контроля и оценки результатов обучения	Наименование оценочного средства
Уметь:		
– разрабатывать алгоритмы для решения задач, записывать их в виде блок-схем и программ, оценивать сложность простых алгоритмов	Оценка выполнения лабораторных работ. Тестирование. Контрольная работа. Экзамен.	ЛР №7–10, 17, 18; КР №2; тесты 6–12; билеты №1, 2, 8
– писать, запускать и отлаживать программы на Python с использованием структурного и модульного подхода	Оценка выполнения лабораторных работ. Контрольная работа. Экзамен.	ЛР №8–16; КР №2; тесты 6–13; билеты №1–7
– работать с файлами и каталогами, форматами CSV и JSON, обрабатывать исключения, вести журнал событий	Оценка выполнения лабораторных работ. Контрольная работа. Экзамен.	ЛР №19–22; КР №3; тесты 14–16; билеты №9–11
– разрабатывать программы на основе объектно-ориентированного подхода, внедрять зависимости	Оценка выполнения лабораторных работ. Контрольная работа. Экзамен.	ЛР №27–30; КР №4; тесты 20–23; билеты №14, 15
– взаимодействовать с базами данных из программ с использованием параметризованных запросов, разрабатывать клиенты REST API	Оценка выполнения лабораторных работ. Контрольная работа. Экзамен.	ЛР №23–26; КР №3; тесты 17–19; билеты №12, 13
– писать сценарии автоматизации на Bash и Python для обработки журналов, управления процессами и контейнерами Docker	Оценка выполнения лабораторных работ. Контрольная работа. Экзамен.	ЛР №35–40; КР №4; тесты 26–28; билеты №18–22
– тестировать программы модульными тестами, оформлять проект и проверять код линтером	Оценка выполнения лабораторных работ. Контрольная работа. Экзамен.	ЛР №29, 31–34; КР №4; тесты 24, 25; билеты №16, 17
– использовать Git и GitVerse: ветвление, слияние, запросы на слияние, автоматическая проверка	Оценка выполнения лабораторных работ. Тестирование. Контрольная работа.	ЛР №2–6; КР №1; тесты 1–5, 29; типовые задания №1–3

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Результаты обучения (освоенные умения, усвоенные знания)	Формы и методы контроля и оценки результатов обучения	Наименование оценочного средства
– использовать российские ИИ-ассистенты для подготовки и проверки кода с соблюдением требований безопасности	Оценка выполнения лабораторных работ. Тестирование. Экзамен.	ЛР №41, 42; тесты 30–33; билеты №7, 12, 23, 24
Знать:		
– понятие алгоритма, его свойства, способы записи, алгоритмические структуры, оценка сложности	Тестирование. Контрольная работа №2. Экзамен.	Тесты 11, 12; КР №2; вопросы 7, 8, 22
– основные элементы языка Python: типы данных, операторы, структуры данных, функции, модули, исключения	Тестирование. Контрольные работы №2, 3. Экзамен.	Тесты 6–10, 13–15; КР №2, 3; вопросы 9–19, 26
– принципы структурного, модульного и объектно-ориентированного программирования, принципы SOLID	Тестирование. Контрольная работа №4. Экзамен.	Тесты 20–23; КР №4; вопросы 33–38
– алгоритмы поиска и сортировки, рекурсия, области применения структур данных	Контрольная работа №2. Экзамен.	КР №2; вопросы 18, 20, 21
– работа с файлами, форматами CSV и JSON, журналирование; основы HTTP и REST	Тестирование. Контрольная работа №3. Экзамен.	Тесты 14, 16, 18, 19; КР №3; вопросы 23–25, 27, 30–32
– взаимодействие программ с реляционными БД, защита от SQL-инъекций	Тестирование. Контрольная работа №3. Экзамен.	Тест 17; КР №3; вопросы 28, 29
– методы и инструменты тестирования и верификации, правила оформления кода и проекта	Тестирование. Контрольная работа №4. Экзамен.	Тесты 24, 25; КР №4; вопросы 36, 39–42
– возможности Bash-сценариев, средства управления процессами и контейнерами из программ	Тестирование. Контрольная работа №4. Экзамен.	Тесты 26–28; КР №4; вопросы 43–48
– основы работы с Git и GitVerse, код-ревью и непрерывная интеграция	Тестирование. Контрольная работа №1. Экзамен.	Тесты 1–5, 29; КР №1; вопросы 3–6, 53
– возможности, ограничения и риски применения ИИ-ассистентов; требования 152-ФЗ, академическая добросовестность	Тестирование. Экзамен.	Тесты 30–35; вопросы 49–52; билеты №7, 12, 23, 24

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Оценка освоения УД предусматривает использование пятибалльной системы оценки.

5.1.2 ФОРМЫ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Таблица 2 – Запланированные формы промежуточной аттестации

№ семестра	Формы промежуточной аттестации	Форма проведения
5	Экзамен	Устный ответ по билету и выполнение практического задания

5.1.3 ОПИСАНИЕ ПРОЦЕДУРЫ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Экзамен по дисциплине проводится в 5 семестре и устанавливает уровень сформированности умений и усвоения знаний по всему курсу. К экзамену допускаются обучающиеся, выполнившие и защитившие все лабораторные работы, включая итоговый проект, и получившие положительные оценки за контрольные работы. Билет содержит два теоретических вопроса (по темам разных разделов курса) и практическое задание (разбор фрагмента кода или сценария: предсказание результата, поиск и исправление ошибки, рефакторинг, написание тестов, проверка кода, подготовленного ИИ-ассистентом). Всего в банке 24 билета. Использование ИИ-ассистентов и посторонних материалов на экзамене не допускается.

В результате освоения дисциплины студент должен уметь:

– разрабатывать алгоритмы для решения задач, записывать их в виде блок-схем и программ, оценивать сложность простых алгоритмов;

– писать, запускать и отлаживать программы на языке Python с использованием структурного и модульного подхода (ветвления, циклы, функции, структуры данных, строки, регулярные выражения);

– работать с файлами, каталогами и форматами CSV и JSON, обрабатывать исключения и вести журнал событий средствами модуля logging;

– разрабатывать программы на основе объектно-ориентированного подхода: классы, наследование, полиморфизм, внедрение зависимостей;

– взаимодействовать с базами данных SQLite и MariaDB из программ с использованием параметризованных запросов, разрабатывать клиенты REST API на основе протокола HTTP;

– писать сценарии автоматизации на Bash и Python для обработки журналов, управления процессами, службами и контейнерами Docker;

– тестировать программы модульными тестами (pytest), оформлять проект (структура, зависимости, документация) и проверять код линтером;

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

– использовать систему контроля версий Git и платформу GitVerse: ветвление, слияние, разрешение конфликтов, запросы на слияние, автоматическая проверка (GitVerse Actions);
– использовать российские ИИ-ассистенты (GigaChat, GigaCode, Алиса AI) для подготовки, объяснения и тестирования кода с обязательной проверкой результата и соблюдением требований информационной безопасности.

В результате освоения дисциплины студент должен знать:

– понятие алгоритма, его свойства, способы записи, виды алгоритмических структур, оценку сложности алгоритмов;
– основные элементы языка Python: типы данных, операторы, структуры данных, функции, модули, обработку исключений;
– принципы структурного, модульного и объектно-ориентированного программирования (инкапсуляция, наследование, полиморфизм), принципы SOLID;
– алгоритмы поиска и сортировки, рекурсию, области применения основных структур данных;
– способы работы с файлами, форматами CSV и JSON, журналированием; основы протокола HTTP и архитектурного стиля REST;
– принципы взаимодействия программ с реляционными базами данных, меры защиты от SQL-инъекций;
– методы и инструменты тестирования и верификации программного обеспечения, правила оформления кода (PEP 8) и проекта;
– возможности командных оболочек и Bash-сценариев, средства управления процессами и контейнерами из программ;
– основы работы с Git и GitVerse, процесс код-ревью и принципы непрерывной интеграции;
– возможности, ограничения и риски применения ИИ-ассистентов при разработке; требования Федерального закона №152-ФЗ и правила академической добросовестности.

Количество заданий для студента: 3 (два теоретических вопроса и одно практическое задание)

Время выполнения каждого задания и максимальное время на проведение аттестации:

Вопрос 1 – 10 мин.

Вопрос 2 – 10 мин.

Практическое задание – 25 мин.

Всего на экзамен (с подготовкой) – 50 мин. на одного обучающегося.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Условия выполнения заданий

Помещение: лаборатория «Алгоритмизации и программирования программных решений»

Требования охраны труда: инструктаж по технике безопасности при работе с вычислительной техникой и электрооборудованием.

Оборудование:

– компьютеры по числу экзаменуемых с установленными Python 3, pytest, Git и редактором кода (без доступа к ИИ-ассистентам и сети Интернет);

– проектор;

– билеты на бумажном и электронном носителях.

Перечень справочной и нормативной литературы, разрешённой для использования:

1. Официальная документация Python и справочные страницы man (в оффлайн-режиме) – только для практического задания;
2. Памятка по синтаксису Python, Git и Bash, выданная преподавателем.

5.1.4 КРИТЕРИИ ОЦЕНКИ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Таблица 3 – Критерии оценки

Оценка	Показатели оценки
Отлично (90–100 баллов)	Студент умеет увязывать теорию с практикой (читает и исправляет код, предсказывает результат, пишет тесты, поясняет решение), владеет понятийным аппаратом, полно и глубоко овладел материалом, обосновывает свои суждения и даёт правильные ответы на вопросы преподавателя.
Хорошо (75–89 баллов)	Студент умеет увязывать теорию с практикой, владеет понятийным аппаратом, полно овладел материалом, но ответы содержат некоторые неточности и требуют уточнения и комментария со стороны преподавателя.
Удовлетворительно (60–74 балла)	Студент знает и понимает материал, но изложение неполное, непоследовательное, допускаются неточности в определении понятий, студент не может обосновать свои ответы на уточняющие вопросы.
Неудовлетворительно (менее 60 баллов)	Студент допускает ошибки в определении понятий, искажающие их смысл, беспорядочно и неуверенно излагает материал, делает ошибки в ответах на уточняющие вопросы.

Распределение баллов на экзамене

Задание	Максимум баллов
Теоретический вопрос 1	30
Теоретический вопрос 2	30
Практическое задание (разбор и доработка фрагмента кода)	40
Итого	100

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Оценка выставляется по сумме баллов: 90–100 – «отлично», 75–89 – «хорошо», 60–74 – «удовлетворительно», менее 60 – «неудовлетворительно».

Критерии оценки теоретического вопроса (30 баллов): полнота и точность ответа – 0–15; корректность терминологии – 0–8; примеры (фрагменты кода, команды) – 0–7.

Критерии оценки практического задания (40 баллов): каждый из трёх подвопросов оценивается до 13 баллов (правильный результат – 6; обоснование – 4; корректность кода и терминологии – 3), ещё 1 балл начисляется за аккуратное оформление решения.

Дополнительные правила

– К экзамену допускаются обучающиеся, выполнившие и защитившие все лабораторные работы (включая итоговый проект) и получившие положительные оценки за контрольные работы.

– Если практическое задание не выполнено вовсе, оценка выше «удовлетворительно» не выставляется.

– Передача в запросах к ИИ-ассистенту (в учебных работах) паролей, ключей, персональных данных и конфиденциального исходного кода является основанием для снижения оценки за работу.

– Использование ИИ-ассистентов, справочных материалов, кроме разрешённых, и средств связи на экзамене влечёт аннулирование результата и повторную сдачу.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

5.2 КОМПЛЕКТ «ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ»

5.2.1 Вопросы к экзамену

1. Языки программирования: классификация, компиляторы и интерпретаторы. Жизненный цикл программы. Место Python и Bash в автоматизации эксплуатации.
2. Среда разработки Python: интерпретатор, виртуальное окружение (venv), менеджер пакетов pip, назначение файла requirements.txt.
3. Система контроля версий Git: рабочий каталог, индекс, репозиторий. Команды init, add, commit, status, log, diff. Правила оформления коммитов.
4. Ветки в Git: создание и слияние, конфликты слияния и их разрешение. Сравнение merge и rebase.
5. GitVerse: удалённые репозитории, SSH-ключи, команды push, pull, fetch, защищённые ветки. Назначение GigaCode.
6. Запрос на слияние (Pull Request) и код-ревью. GitVerse Actions: структура workflow, понятие непрерывной интеграции.
7. Алгоритм и его свойства. Способы записи алгоритмов. Блок-схема и основные элементы по ГОСТ 19.701-90.
8. Базовые алгоритмические структуры: следование, ветвление, цикл. Примеры.
9. Типы данных Python: int, float, str, bool, None; приведение типов; изменяемые и неизменяемые типы.
10. Операторы if, elif, else. Логические операции, вложенные условия, граничные значения.
11. Циклы for и while. Функции range и enumerate, операторы break и continue. Типовые задачи на циклы.
12. Списки: индексы, срезы, методы, генераторы списков. Вложенные списки и двумерные массивы.
13. Строки: основные методы, форматирование, f-строки.
14. Регулярные выражения (модуль re): синтаксис, функции match, search, findall, sub. Примеры разбора строк журнала.
15. Словари и множества: операции, сложность доступа, подсчёт частот. Кортежи и их назначение.
16. Функции: параметры, значения по умолчанию, именованные и произвольные аргументы, возвращаемое значение, docstring. Ошибка изменяемого значения по умолчанию.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

17. Области видимости: локальные и глобальные переменные. Передача изменяемых и неизменяемых объектов в функцию.
18. Рекурсия: базовый случай, стек вызовов, ограничение глубины. Сравнение рекурсии и итерации.
19. Модули и пакеты: импорт, конструкция if __name__ == '__main__', обзор стандартной библиотеки.
20. Алгоритмы поиска: линейный и бинарный, условия применимости бинарного поиска.
21. Алгоритмы сортировки: пузырьком, выбором, вставками; встроенная сортировка Python.
22. Сложность алгоритмов: O(1), O(log n), O(n), O(n log n), O(n^2). Измерение времени выполнения (timeit).
23. Работа с текстовыми файлами: режимы открытия, кодировки, менеджер контекста with, построчная обработка больших файлов.
24. Форматы CSV и JSON: модули csv и json, сериализация и десериализация, типичные ошибки.
25. Работа с файловой системой: модули os, pathlib, shutil; обход каталогов, права доступа, безопасная работа с путями.
26. Исключения: иерархия, конструкции try, except, else, finally, оператор raise, собственные исключения.
27. Журналирование: модуль logging, уровни важности, форматирование, обработчики, ротация журналов.
28. Взаимодействие с базами данных из Python: sqlite3, курсор, транзакции, закрытие соединения; подключение к MariaDB.
29. SQL-инъекции: сущность атаки, параметризованные запросы, дополнительные меры защиты.
30. Протокол HTTP: методы, коды состояния, заголовки. Архитектурный стиль REST, формат JSON.
31. Библиотека requests: запросы GET и POST, параметры, заголовки, тело JSON, таймаут, проверка кода ответа.
32. Клиент REST API: обработка ошибок 4xx и 5xx, повторные попытки с задержкой, хранение токена доступа.
33. Основы ООП: объект, класс, атрибуты и методы, конструктор. Инкапсуляция в Python (защищённые атрибуты, свойства).
34. Наследование и полиморфизм: переопределение методов, super(), абстрактные базовые классы.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

35. Композиция и наследование: критерии выбора. Специальные методы __str__, __repr__, __eq__.
36. Правила оформления кода PEP 8. Линтер flake8. «Запахи кода» и рефакторинг.
37. Принципы SOLID: единственная ответственность, открытость/закрытость, подстановка Лисков.
38. Принципы SOLID: разделение интерфейсов, инверсия зависимостей. Внедрение зависимостей через конструктор.
39. Виды тестирования. Пирамида тестов. Методы верификации программного обеспечения.
40. Модульные тесты в pytest: утверждения, параметризация, фикстуры, проверка исключений.
41. Тестовые заглушки (mock), измерение покрытия кода. Подбор тестов по границам значений и классам эквивалентности.
42. Структура проекта на Python: requirements.txt, pyproject.toml, README. Консольные приложения: argparse, коды возврата.
43. Конвейеры и фильтры Bash: grep, awk, sed, cut, sort, uniq, wc. Анализ журналов веб-сервера.
44. Сценарии Bash: параметры, условия, циклы, функции, коды возврата, режим set -euo pipefail, перехват сигналов (trap).
45. Сравнение Bash и Python для задач автоматизации: критерии выбора инструмента.
46. Модуль subprocess: безопасный запуск внешних команд, таймаут, проверка кода возврата. Риск внедрения команд при shell=True.
47. Мониторинг ресурсов: каталог /proc, команды ps, free, df; формирование отчётов и пороговых предупреждений.
48. Автоматизация Docker из Python и Bash: управление контейнерами, проверки состояния (healthcheck), развёртывание Docker Compose-проекта и откат.
49. Российские ИИ-ассистенты (GigaChat, GigaCode, Алиса AI, YandexGPT): назначение и возможности в разработке и сопровождении программ.
50. Правила постановки задачи ИИ-ассистенту при генерации кода и тестов: контекст, ограничения, формат результата, примеры данных.
51. Риски применения ИИ-ассистентов: «галлюцинации», устаревшие и небезопасные решения, утечка данных. Чек-лист проверки сгенерированного кода.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

52. Требования Федерального закона №152-ФЗ, академическая добросовестность и указание использования ИИ при выполнении учебных и рабочих задач.
53. Командная разработка: ветки, запросы на слияние, код-ревью, автоматическая проверка (CI), планирование задач в YouGile.

5.2.2 Типовые практические задания

1. Создайте виртуальное окружение Python, установите пакет pytest, зафиксируйте зависимости в `requirements.txt`. Инициализируйте репозиторий Git, добавьте `.gitignore` (исключить `.venv` и `__pycache__`) и оформите два осмысленных коммита.
2. Создайте две ветки, внесите в них конфликтующие изменения одного файла, выполните слияние и разрешите конфликт без потери смысла обеих версий. Опубликуйте результат в GitVerse и оформите запрос на слияние по шаблону.
3. Напишите workflow GitVerse Actions для Python-проекта: установка зависимостей, проверка стиля `flake8`, запуск тестов `pytest`; добейтесь успешного статуса проверки.
4. Составьте блок-схему и напишите программу: по введённому натуральному числу N вывести, сколько чисел от 1 до N делятся на 3 или на 5, и их сумму. Проверьте программу при N = 15 (ожидаемый результат: 7 чисел, сумма 60).
5. Напишите программу, которая читает файл `access.log` формата Apache, выводит три IP-адреса с наибольшим числом запросов и число ответов с кодами 5xx (используйте регулярное выражение и `collections.Counter`).
6. Напишите функцию `parse_size(s)`, переводящую строки вида `10K`, `5M`, `2G` в число байтов, с обработкой некорректных значений исключением `ValueError`, и параметризованные тесты pytest для неё.
7. Напишите рекурсивную функцию, подсчитывающую суммарный размер файлов в каталоге и его подкаталогах (`pathlib`), и сравните результат с командой `du -sb`. Укажите, когда предпочтительнее итеративный вариант.
8. Прочитайте CSV-файл инвентаризации серверов (`host,ip,cpu,ram_gb`), выберите серверы с объёмом памяти менее 16 ГБ и запишите их в `low_ram.json` (UTF-8, с отступами). Строки с некорректными данными пропускайте с записью в журнал.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

9. Создайте базу SQLite, таблицу `incident` (`id`, `host`, `minutes`), загрузите в неё данные из CSV и напишите функцию, возвращающую суммарное время простоя хоста с использованием параметризованного запроса. Покажите, что попытка SQL-инъекции не приводит к успеху.
10. Напишите клиент REST API учебного сервиса: функция получает список серверов (`GET /api/servers`), использует таймаут, повторяет запрос при ошибках 5xx и сетевых сбоях (не более трёх попыток с задержкой) и записывает ошибки в журнал; проверьте функцию тестом с заглушкой вместо `requests`.
11. Разработайте иерархию классов `Server`, `WebServer`, `DbServer` с методом `check()` и демонстрацией полиморфизма; зависимость от способа проверки внедрите через конструктор; напишите тесты pytest.
12. Напишите Bash-сценарий резервного копирования каталога в архив `tar.gz` с датой в имени, хранением семи последних копий, журналом и корректными кодами возврата (режим `set -euo pipefail`). Проверьте сценарий на тестовом каталоге.
13. Напишите на Python сценарий, запускающий проект Docker Compose, ожидающий состояния `healthy` (не более 60 секунд), выполняющий проверочный HTTP-запрос и при ошибке выводящий журнал контейнера и завершающийся с кодом 1.
14. Поручите ИИ-ассистенту (GigaChat или Алиса AI) написать функцию разбора строки журнала. Не передавая реальных данных, найдите в ответе не менее двух недостатков, исправьте код, добавьте тесты и оформите протокол проверки с указанием использования ИИ.

5.2.3 Задания для экзамена

Экзаменационный билет №1

Вопрос 1

Алгоритм и его свойства. Способы записи алгоритмов. Блок-схема и основные элементы по ГОСТ 19.701-90.

Вопрос 2

Система контроля версий Git: рабочий каталог, индекс, репозиторий. Команды init, add, commit, status, log, diff. Правила оформления коммитов.

Практическое задание

Дана функция классификации температуры воздуха.

```
def classify(t):
    if t < 0:
        return "мороз"
    elif t < 10:
        return "холодно"
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

```
    elif t < 20:
        return "прохладно"
    return "тепло"

print([classify(x) for x in (-5, 0, 10, 19.9, 25)])
```

1. Определите, что выведет программа, и объясните выбор ветки для значений 0 и 10.
2. Составьте набор тестовых значений для проверки всех границ и запишите ожидаемые результаты.
3. Перепишите функцию так, чтобы пороги хранились в списке кортежей, а функция не содержала цепочки elif.

Эталонный вывод (для экзаменатора):

```
['мороз', 'холодно', 'прохладно', 'прохладно', 'тепло']
```

Эталон ответа для экзаменатора. Граничное значение относится к верхнему интервалу (0 – «холодно», 10 – «прохладно»), так как условия строгие (<). Границы для тестов: -0.1, 0, 9.9, 10, 19.9, 20. Вариант: `LIMITS = [(0, "мороз"), (10, "холодно"), (20, "прохладно")]`, цикл `for limit, name in LIMITS: if t < limit: return name`, затем `return "тепло"`.

Экзаменационный билет №2

Вопрос 1

Циклы for и while. Функции range и enumerate, операторы break и continue. Типовые задачи на циклы.

Вопрос 2

GitVerse: удалённые репозитории, SSH-ключи, команды push, pull, fetch, защищённые ветки. Назначение GigaCode.

Практическое задание

Дан фрагмент обработки списка измерений.

```
data = [4, 8, 15, 16, 23, 42]
total = 0
count = 0
for i, x in enumerate(data):
    if i % 2 == 0:
        continue
    total += x
    count += 1
print(total, count, total / count)
```

1. Определите, что выведет программа, и поясните, какие элементы списка участвуют в вычислении.
2. Какое исключение возникнет, если список `data` будет пустым или состоит из одного элемента? Как этого избежать?

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

3. Запишите ту же обработку с помощью среза и встроенных функций `sum` и `len`.

Эталонный вывод (для экзаменатора):

```
66 3 22.0
```

Эталон ответа для экзаменатора. Участвуют элементы с нечётными индексами (8, 16, 42). При пустом списке и при списке из одного элемента `count` равен нулю, возникнет `ZeroDivisionError`; нужна проверка `if count:`. Вариант: `part = data[1::2]; print(sum(part), len(part), sum(part) / len(part))`.

Экзаменационный билет №3

Вопрос 1

Списки: индексы, срезы, методы, генераторы списков. Вложенные списки и двумерные массивы.

Вопрос 2

Работа с текстовыми файлами: режимы открытия, кодировки, менеджер контекста with, построчная обработка больших файлов.

Практическое задание

Дан фрагмент работы со списком.

```
a = [3, 1, 4, 1, 5, 9, 2, 6]
print(a[2:5])
print(a[::-2])
print([x * x for x in a if x % 2 == 0])
print(sorted(set(a)))
print(a)
```

1. Определите вывод программы построчно.
2. Чем `sorted(a)` отличается от `a.sort()`? Что произойдёт со списком `a` в каждом случае?
3. Напишите выражение, которое возвращает список индексов элементов, равных единице.

Эталонный вывод (для экзаменатора):

```
[4, 1, 5]
[6, 9, 1, 1]
[16, 4, 36]
[1, 2, 3, 4, 5, 6, 9]
[3, 1, 4, 1, 5, 9, 2, 6]
```

Эталон ответа для экзаменатора. `sorted` возвращает новый список и не меняет исходный, `sort()` сортирует список на месте и возвращает `None`. Индексы единиц: `[i for i, x in enumerate(a) if x == 1]` (результат `[1, 3]`).

Экзаменационный билет №4

Вопрос 1

Регулярные выражения (модуль re): синтаксис, функции match, search, findall, sub. Примеры разбора строк журнала.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Вопрос 2

Запрос на слияние (Pull Request) и код-ревью. GitVerse Actions: структура workflow, понятие непрерывной интеграции.

Практическое задание

Дана строка журнала веб-сервера и фрагмент её разбора.

```
import re


line = '192.168.1.15 - - [05/Mar/2026:10:21:45 +0500] "GET /index.php
HTTP/1.1" 404 512'
m = re.match(r'(\d+\.\d+\.\d+\.\d+) .*?"(\w+) (\S+) [^"]*" (\d{3}) (\d+)',
line)
print(m.groups())
```

1. Определите вывод программы и назначение каждой группы регулярного выражения.
2. Что произойдёт, если строка не соответствует шаблону? Как обезопасить код?
3. Допишите код, который выводит предупреждение для кодов ответа 4xx и 5xx.

Эталонный вывод (для экзаменатора):

```
('192.168.1.15', 'GET', '/index.php', '404', '512')
```

Эталон ответа для экзаменатора. При несовпадении `re.match` возвращает `None`, и вызов `m.groups()` приведёт к `AttributeError`; нужна проверка `if m is None:` с записью в журнал и пропуском строки. Проверка: `status = int(m.group(4)); if status >= 400: print("предупреждение:", m.group(3), status)`.

Экзаменационный билет №5

Вопрос 1

Словари и множества: операции, сложность доступа, подсчёт частот. Кортежи и их назначение.

Вопрос 2

Журналирование: модуль logging, уровни важности, форматирование, обработчики, ротация журналов.

Практическое задание

Дан фрагмент подсчёта слов.

```
words = "to be or not to be that is the question".split()
freq = {}
for w in words:
    freq[w] = freq.get(w, 0) + 1
top = sorted(freq.items(), key=lambda p: (-p[1], p[0]))[:3]
print(top)
```

1. Определите вывод программы и объясните ключ сортировки.
2. Замените цикл подсчёта на `collections.Counter` и получите тот же результат.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

3. Какова сложность доступа к элементу словаря и поиска элемента в списке? Почему словарь подходит для подсчёта?

Эталонный вывод (для экзаменатора):

```
[('be', 2), ('to', 2), ('is', 1)]
```

Эталон ответа для экзаменатора. Сортировка по убыванию частоты, а при равенстве – по алфавиту. Вариант: `from collections import Counter; Counter(words).most_common(3)` вернёт `[('to', 2), ('be', 2), ('or', 1)]`: при равных частотах сохраняется порядок первого появления слов, поэтому для результата, совпадающего с исходным ключом, сортировка с ключом `(-частота, слово)` всё равно необходима. Доступ к элементу словаря – в среднем O(1), поиск в списке – O(n).

Экзаменационный билет №6

Вопрос 1

Функции: параметры, значения по умолчанию, именованные и произвольные аргументы, возвращаемое значение, docstring. Ошибка изменяемого значения по умолчанию.

Вопрос 2

Основы ООП: объект, класс, атрибуты и методы, конструктор. Инкапсуляция в Python (защищённые атрибуты, свойства).

Практическое задание

Дана функция, собирающая элементы в список.

```
def add_item(item, bucket=[]):
    bucket.append(item)
    return bucket


print(add_item(1))
print(add_item(2))
print(add_item(3, []))
```

1. Определите вывод программы и объясните результат второго вызова.
2. Исправьте функцию так, чтобы каждый вызов без второго аргумента получал новый список.
3. Поясните, как связана эта ошибка с понятием изменяемого типа данных.

Эталонный вывод (для экзаменатора):

```
[1]
[1, 2]
[3]
```

Эталон ответа для экзаменатора. Значение по умолчанию создаётся один раз при определении функции, поэтому список накапливает элементы между вызовами. Исправление: `def add_item(item, bucket=None): if bucket is None: bucket = [].`

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Экзаменационный билет №7

Вопрос 1

Рекурсия: базовый случай, стек вызовов, ограничение глубины. Сравнение рекурсии и итерации.

Вопрос 2

Российские ИИ-ассистенты (GigaChat, GigaCode, Алиса AI, YandexGPT): назначение и возможности в разработке и сопровождении программ.

Практическое задание

Дана рекурсивная функция.

```
def f(n):
    if n <= 1:
        return n
    return f(n - 1) + f(n - 2)

print([f(i) for i in range(8)])
```

1. Определите вывод программы. Какой последовательности он соответствует?
2. Оцените, как растёт число рекурсивных вызовов при увеличении `n`, и объясните причину.
3. Ускорьте функцию с помощью `functools.lru_cache` либо итеративной версии.

Эталонный вывод (для экзаменатора):

```
[0, 1, 1, 2, 3, 5, 8, 13]
```

Эталон ответа для экзаменатора. Числа Фибоначчи. Число вызовов растёт экспоненциально (около 2^n), так как подзадачи пересчитываются. Исправление: декоратор `@lru_cache(maxsize=None)` (сложность O(n)) либо цикл с двумя переменными `a, b = b, a + b`.

Экзаменационный билет №8

Вопрос 1

Алгоритмы поиска: линейный и бинарный, условия применимости бинарного поиска.

Вопрос 2

Модульные тесты в pytest: утверждения, параметризация, фикстуры, проверка исключений.

Практическое задание

Студент реализовал бинарный поиск в отсортированном списке.

```
def bsearch(a, x):
    lo, hi = 0, len(a) - 1
    while lo < hi:
        mid = (lo + hi) // 2
        if a[mid] == x:
            return mid
        if a[mid] < x:
            lo = mid + 1
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

```
        else:
            hi = mid - 1
    return -1


print(bsearch([1, 3, 5, 7, 9], 9), bsearch([1, 3, 5, 7, 9], 1), bsearch([1,
3, 5, 7, 9], 5), bsearch([4], 4))
```

1. Определите вывод программы.
2. Найдите ошибку и объясните, при каких входных данных она проявляется.
3. Исправьте функцию и укажите её временную сложность.

Эталонный вывод (для экзаменатора):

```
-1 0 2 -1
```

Эталон ответа для экзаменатора. Ошибка в условии цикла: должно быть `while lo <= hi`; при `lo == hi` последний кандидат не проверяется (искомый элемент на границе, список из одного элемента). После исправления для тех же вызовов результат `4 0 2 0`. Сложность O(log n).

Экзаменационный билет №9

Вопрос 1

Форматы CSV и JSON: модули csv и json, сериализация и десериализация, типичные ошибки.

Вопрос 2

Сложность алгоритмов: O(1), O(log n), O(n), O(n log n), O(n²). Измерение времени выполнения (timeit).

Практическое задание

Дан фрагмент обработки CSV с результатами проверки узлов.

```
import csv, io


src = io.StringIO("host,status,ms\nweb-01,ok,120\ndb-01,fail,0\nweb-
01,ok,80\nweb-02,fail,0\n")
stat = {}
for row in csv.DictReader(src):
    stat.setdefault(row["host"], []).append(int(row["ms"]))
for h, v in sorted(stat.items()):
    print(h, len(v), sum(v) / len(v))
```

1. Определите вывод программы.
2. Как изменить код для чтения из реального файла `checks.csv`? Какие параметры `open` необходимо указать и почему?
3. Запишите результат в файл `report.json` в удобочитаемом виде с сохранением русских символов.

Эталонный вывод (для экзаменатора):

```
db-01 1 0.0
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

```
web-01 2 100.0
web-02 1 0.0
```

Эталон ответа для экзаменатора. `with open("checks.csv", newline="", encoding="utf-8") as f:` — `newline=""` требуется модулю csv, кодировка задаётся явно. Запись: `json.dump(result, f, ensure_ascii=False, indent=2)` в файле, открытом с `encoding="utf-8"`.

Экзаменационный билет №10

Вопрос 1

Исключения: иерархия, конструкции try, except, else, finally, оператор raise, собственные исключения.

Вопрос 2

Правила оформления кода PEP 8. Линтер flake8. «Запахи кода» и рефакторинг.

Практическое задание

Дана функция разбора номера порта.

```
def parse_port(s):
    try:
        p = int(s)
    except ValueError:
        return None
    else:
        if not 1 <= p <= 65535:
            raise ValueError("порт вне диапазона")
        return p
    finally:
        print("проверено", s)

for s in ("80", "abc", "70000"):
    try:
        print(parse_port(s))
    except ValueError as e:
        print("ошибка:", e)
```

1. Определите вывод программы построчно и объясните порядок выполнения блока `finally`.
2. Почему возвращать `None` при нечисловом значении и выбрасывать исключение при диапазоне – непоследовательное решение? Предложите единый подход.
3. Опишите собственный класс исключения `InvalidPortError` и используйте его.

Эталонный вывод (для экзаменатора):

```
проверено 80
80
проверено abc
None
проверено 70000
ошибка: порт вне диапазона
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Эталон ответа для экзаменатора. Блок `finally` выполняется всегда, до передачи результата вызывающему коду. Единый подход: в обоих случаях выбрасывать `InvalidPortError(ValueError)`. Класс: `class InvalidPortError(ValueError): pass`.

Экзаменационный билет №11

Вопрос 1

Журналирование: модуль logging, уровни важности, форматирование, обработчики, ротация журналов.

Вопрос 2

Алгоритмы сортировки: пузырьком, выбором, вставками; встроенная сортировка Python.

Практическое задание

Дана настройка журналирования резервного копирования.

```
import logging


logging.basicConfig(level=logging.WARNING,
format="%(levelname)s:%(message)s")
log = logging.getLogger("backup")
log.debug("старт")
log.info("копирование")
log.warning("мало места")
log.error("сбой")
```

1. Какие сообщения будут выведены и в какой поток? Почему не выведены остальные?
2. Измените настройку так, чтобы сообщения уровня INFO и выше записывались в файл `backup.log` с временем.
3. Как организовать ротацию журнала по размеру (5 файлов по 1 МБ)?

Эталон ответа для экзаменатора. Будут выведены `WARNING:мало места` и `ERROR:сбой` (в поток stderr), так как порог – WARNING. Файл: `logging.basicConfig(filename="backup.log", level=logging.INFO, format="%(asctime)s %(levelname)s %(message)s")`. Ротация: `logging.handlers.RotatingFileHandler("backup.log", maxBytes=1_000_000, backupCount=5)`.

Экзаменационный билет №12

Вопрос 1

SQL-инъекции: сущность атаки, параметризованные запросы, дополнительные меры защиты.

Вопрос 2

Риски применения ИИ-ассистентов: «галлюцинации», устаревшие и небезопасные решения, утечка данных. Чек-лист проверки сгенерированного кода.

Практическое задание

Функция поиска пользователя в базе данных SQLite.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

```
def get_user(cur, login):
    cur.execute("SELECT id, login FROM users WHERE login = '" + login +
"'")
    return cur.fetchall()
```

1. Покажите значение параметра `login`, при котором функция вернёт всех пользователей, и объясните причину.
2. Перепишите функцию безопасно.
3. Назовите ещё две меры защиты базы данных от последствий уязвимостей в приложении.

Эталон ответа для экзаменатора. Значение `x' OR '1'='1`: ввод становится частью запроса и условие истинно для всех строк. Исправление: `cur.execute("SELECT id, login FROM users WHERE login = ?", (login,))` (для коннектора MariaDB – маркер `%s`). Меры: учётная запись приложения с минимальными правами (без DROP, DELETE), проверка входных данных, журналирование запросов.

Экзаменационный билет №13

Вопрос 1

Клиент REST API: обработка ошибок 4xx и 5xx, повторные попытки с задержкой, хранение токена доступа.

Вопрос 2

Типы данных Python: int, float, str, bool, None; приведение типов; изменяемые и неизменяемые типы.

Практическое задание

Функция проверки состояния сервиса, написанная студентом.

```
import requests


def get_status(host):
    r = requests.get(f"http://{host}/health")
    return r.json()["status"]
```

1. Найдите не менее четырёх проблем функции (надёжность, безопасность, диагностика).
2. Перепишите функцию: таймаут, обработка сетевых ошибок и кодов ответа, возврат строки `unknown` при сбое, запись в журнал.
3. Как организовать три повторные попытки с возрастающей задержкой?

Эталон ответа для экзаменатора. Проблемы: нет таймаута (программа может зависнуть); не обрабатываются сетевые исключения `requests.RequestException`; не проверяется код ответа (`raise_for_status`); ответ может не быть JSON или не содержать ключ `status` (`ValueError`, `KeyError`); нет журнала; адрес собирается без проверки. Решение: `try: r = requests.get(url, timeout=5); r.raise_for_status(); return r.json().get("status", "unknown") except (requests.RequestException, ValueError) as e: log.warning(...); return`

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

"unknown". Повторные попытки: цикл for attempt in range(3) с time.sleep(2 ** attempt).

Экзаменационный билет №14

Вопрос 1

Наследование и полиморфизм: переопределение методов, super(), абстрактные базовые классы.

Вопрос 2

Сценарии Bash: параметры, условия, циклы, функции, коды возврата, режим set -euo pipefail, перехват сигналов (trap).

Практическое задание

Дана иерархия классов служб.

```
class Service:
    def __init__(self, name):
        self.name = name

    def status(self):
        return f"{self.name}: unknown"



class Web(Service):
    def status(self):
        return f"{self.name}: listening on 80"



class Db(Service):
    def status(self):
        base = super().status()
        return base + ", replica=off"



for s in (Service("cron"), Web("nginx"), Db("mariadb")):
    print(s.status())
```

1. Определите вывод программы.
2. Какой принцип ООП демонстрирует цикл for и в чём преимущество такого кода?
3. Добавьте класс Cache с методом status, возвращающим строку <имя>: ok, не изменяя цикл. Объясните, какой принцип SOLID при этом соблюдён.

Эталонный вывод (для экзаменатора):

```
cron: unknown
nginx: listening on 80
mariadb: unknown, replica=off
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Эталон ответа для экзаменатора. Полиморфизм: один вызов `status()` для объектов разных классов. Принцип открытости/закрытости: поведение расширяется новым классом без изменения существующего кода.

Экзаменационный билет №15

Вопрос 1

Принципы SOLID: разделение интерфейсов, инверсия зависимостей. Внедрение зависимостей через конструктор.

Вопрос 2

Модули и пакеты: импорт, конструкция if __name__ == '__main__', обзор стандартной библиотеки.

Практическое задание

Дан класс, выполняющий резервное копирование.

```
class Backup:
    def run(self):
        data = open("/var/db/dump.sql").read()
        print("отправка в хранилище:", len(data))
```

1. Какие принципы SOLID нарушены и почему класс трудно тестировать?
2. Перепишите класс с внедрением зависимостей через конструктор (источник данных и хранилище).
3. Покажите, как подменить зависимости в модульном тесте.

Эталон ответа для экзаменатора. Нарушены принцип единственной ответственности (чтение данных и отправка в одном методе) и принцип инверсии зависимостей (жёсткая привязка к файлу и к `print`). Решение: `class Backup: def __init__(self, source, storage): ...; def run(self): data = self.source.read(); self.storage.save(data).` В тесте передаются объекты-заглушки: `Backup(FakeSource("abc"), fake_storage).run()` и проверяется, что `fake_storage.saved == "abc"`.

Экзаменационный билет №16

Вопрос 1

Модульные тесты в pytest: утверждения, параметризация, фикстуры, проверка исключений.

Вопрос 2

Протокол HTTP: методы, коды состояния, заголовки. Архитектурный стиль REST, формат JSON.

Практическое задание

Дана функция проверки IPv4-адреса.

```
def is_valid_ip(s):
    parts = s.split(".")
    if len(parts) != 4:
        return False
    return all(p.isdigit() and 0 <= int(p) <= 255 for p in parts)
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

1. Напишите параметризованный тест pytest не менее чем на шесть случаев, включая граничные (0, 255, 256, пустая строка, лишняя часть).
2. Проверьте функцию на строке `01.2.3.4` и определите, считается ли такой адрес допустимым; предложите, как изменить поведение, если ведущие нули недопустимы.
3. Как запустить тесты и измерить покрытие?

Эталон ответа для экзаменатора. Пример: `@pytest.mark.parametrize("s, expected", [("192.168.0.1", True), ("0.0.0.0", True), ("255.255.255.255", True), ("256.1.1.1", False), ("1.2.3", False), ("", False), ("1.2.3.4.5", False)])`. Строка `01.2.3.4` функцией принимается; чтобы запретить ведущие нули, добавить условие `p == str(int(p))`. Запуск: `pytest -v`, покрытие — `pytest --cov=`. (пакет pytest-cov).

Экзаменационный билет №17

Вопрос 1

Правила оформления кода PEP 8. Линтер flake8. «Запахи кода» и рефакторинг.

Вопрос 2

Взаимодействие с базами данных из Python: sqlite3, курсор, транзакции, закрытие соединения; подключение к MariaDB.

Практическое задание

Фрагмент кода сдан на ревью.

```
import os,sys
def Calc( a,b ):
    x=a+b
    if x>10 : return True
    else : return False
```

1. Найдите не менее пяти нарушений PEP 8 и сопутствующих недостатков.
2. Перепишите фрагмент в соответствии с PEP 8 и упростите логику.
3. Какой командой проверить файл линтером flake8 и что означает вывод вида `file.py:3:6: E225`?

Эталон ответа для экзаменатора. Нарушения: два импорта в одной строке и неиспользуемые импорты; имя функции в стиле CamelCase; пробелы внутри скобок и отсутствие пробела после запятой; отсутствие пробелов вокруг операторов; пробел перед двоеточием; несколько операторов в одной строке; избыточная конструкция `if … return True else … return False`; отсутствие docstring и пустых строк между определениями. Вариант: `def calc(a, b): return a + b > 10` (с двумя пустыми строками перед определением и docstring). Запуск: `flake8 file.py`; формат `файл:строка:столбец: код сообщение`.

Экзаменационный билет №18

Вопрос 1

Конвейеры и фильтры Bash: grep, awk, sed, cut, sort, uniq, wc. Анализ журналов веб-сервера.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Вопрос 2

Работа с файловой системой: модули os, pathlib, shutil; обход каталогов, права доступа, безопасная работа с путями.

Практическое задание

Дан фрагмент журнала веб-сервера `access.log` (девятое поле – код ответа):

```
10.0.0.5 - - [05/Mar/2026:10:00:01 +0500] "GET / HTTP/1.1" 200 512
10.0.0.5 - - [05/Mar/2026:10:00:02 +0500] "GET /a HTTP/1.1" 404 128
10.0.0.7 - - [05/Mar/2026:10:00:03 +0500] "GET /b HTTP/1.1" 200 256
10.0.0.9 - - [05/Mar/2026:10:00:04 +0500] "POST /login HTTP/1.1" 500 64
10.0.0.5 - - [05/Mar/2026:10:00:05 +0500] "GET /c HTTP/1.1" 404 128
10.0.0.7 - - [05/Mar/2026:10:00:06 +0500] "GET / HTTP/1.1" 200 512
```

1. Определите результат команды `awk '{print $9}' access.log | sort | uniq -c | sort -rn | head -2` и объясните роль каждого звена конвейера.
2. Напишите команду, выводящую IP-адреса, которые получили код 404, с числом таких ответов.
3. Как передать результат команды в сценарий так, чтобы при отсутствии файла сценарий завершился с ненулевым кодом?

Эталон ответа для экзаменатора. Результат (два самых частых кода): `3 200` и `2 404`. Звенья: `awk` выделяет поле, `sort` группирует одинаковые значения, `uniq -c` считает, `sort -rn` упорядочивает по убыванию, `head -2` оставляет две строки. Команда 2: `awk '$9 == 404 {print $1}' access.log | sort | uniq -c` (результат: `2 10.0.0.5`). Проверка файла: `[ -r access.log ] || { echo "нет файла" >&2; exit 1; }` либо режим `set -euo pipefail`.

Экзаменационный билет №19

Вопрос 1

Сценарии Bash: параметры, условия, циклы, функции, коды возврата, режим set -euo pipefail, перехват сигналов (trap).

Вопрос 2

Операторы if, elif, else. Логические операции, вложенные условия, граничные значения.

Практическое задание

Сценарий очистки журналов предложен коллегой.

```
#!/bin/bash
for f in $(ls /var/log/app/*.log); do
  if [ $(stat -c %s $f) -gt 1000000 ]; then
    gzip $f
  fi
done
```

1. Найдите не менее четырёх недостатков сценария (надёжность и безопасность).
2. Перепишите сценарий корректно.
3. Как проверить сценарий перед запуском на рабочем сервере?

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Эталон ответа для экзаменатора. Недостатки: разбор вывода `ls` (ломается на пробелах в именах); переменные без кавычек; нет `set -euo pipefail`; не обработан случай, когда файлов нет (шаблон остаётся строкой); нет журнала и кода возврата; повторное сжатие уже сжатых файлов. Вариант: `set -euo pipefail; shopt -s nullglob; for f in /var/log/app/*.log; do size=$(stat -c %s -- "$f"); if [ "$size" -gt 1000000 ]; then gzip -- "$f"; fi; done`. Проверка: тестовый каталог с копиями файлов, режим `bash -n` (проверка синтаксиса), `shellcheck`, затем запуск на тестовой копии.

Экзаменационный билет №20

Вопрос 1

Модуль subprocess: безопасный запуск внешних команд, таймаут, проверка кода возврата. Риск внедрения команд при shell=True.

Вопрос 2

Библиотека requests: запросы GET и POST, параметры, заголовки, тело JSON, таймаут, проверка кода ответа.

Практическое задание

Функция определения свободного места на диске.

```
import subprocess


def free_mb(path):
    out = subprocess.check_output("df -m " + path + " | tail -1",
shell=True, text=True)
    return int(out.split()[3])
```

1. Какая уязвимость присутствует при `shell=True` и конкатенации строки? Приведите пример значения `path`.
2. Перепишите функцию безопасно (список аргументов, таймаут, проверка кода возврата).
3. Как получить тот же результат без внешней команды?

Эталон ответа для экзаменатора. Значение `/; echo hacked` приведёт к выполнению постороннего команды (внедрение команд). Безопасный вариант: `r = subprocess.run(["df", "-m", "--output=avail", path], capture_output=True, text=True, timeout=10, check=True); return int(r.stdout.splitlines()[-1])`. Без внешней команды: `shutil.disk_usage(path).free // (1024 * 1024)`.

Экзаменационный билет №21

Вопрос 1

Мониторинг ресурсов: каталог /proc, команды ps, free, df; формирование отчётов и пороговых предупреждений.

Вопрос 2

Тестовые заглушки (mock), измерение покрытия кода. Подбор тестов по границам значений и классам эквивалентности.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Практическое задание

Дана функция расчёта загрузки памяти по файлу `/proc/meminfo`.

```
def mem_used_percent():
    info = {}
    with open("/proc/meminfo") as f:
        for line in f:
            key, value = line.split(":", 1)
            info[key] = int(value.split()[0])
    return round(100 * (1 - info["MemAvailable"] / info["MemTotal"]), 1)
```

1. Что вычисляет функция? Каков результат при `MemTotal` = 8 000 000 кБ и `MemAvailable` = 2 000 000 кБ?
2. Какие ошибки возможны (отсутствие ключа, другая ОС) и как их обработать?
3. Добавьте запись в журнал уровня WARNING при использовании более 90 % памяти и код возврата 1 для сценария мониторинга.

Эталон ответа для экзаменатора. Процент используемой памяти: 100 · (1 − 2 000 000 / 8 000 000) = 75,0. Ошибки: `KeyError` для `MemAvailable` (старые ядра), `FileNotFoundError` вне Linux; нужны `try/except` с понятным сообщением. Пороговая проверка: `if used > 90: log.warning(...); sys.exit(1).`

Экзаменационный билет №22

Вопрос 1

Автоматизация Docker из Python и Bash: управление контейнерами, проверки состояния (healthcheck), развёртывание Docker Compose-проекта и откат.

Вопрос 2

Принципы SOLID: единственная ответственность, открытость/закрытость, подстановка Лисков.

Практическое задание

Функция проверки контейнера.

```
import subprocess


def container_running(name):
    r = subprocess.run(["docker", "inspect", "-f", "{{.State.Running}}",
name],
                       capture_output=True, text=True)
    return r.stdout.strip() == "true"
```

1. Что вернёт функция, если контейнера с таким именем нет и если программа `docker` не установлена? Чем эти ситуации опасны для сценария мониторинга?
2. Улучшите функцию: таймаут, различение «контейнер остановлен» и «ошибка выполнения», журнал.
3. Какой параметр секции `healthcheck` в Docker Compose позволяет проверять работоспособность службы, а не только факт запуска?

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Эталон ответа для экзаменатора. Для отсутствующего контейнера команда завершается с кодом 1 и пустым выводом – функция вернёт `False` (неотличимо от остановленного); при отсутствии `docker` возникнет `FileNotFoundError`. Улучшение: перехват `FileNotFoundError` и `subprocess.TimeoutExpired`, проверка `r.returncode`, запись `r.stderr` в журнал, собственное исключение. Healthcheck: ключи `test`, `interval`, `timeout`, `retries` (проверка состояния `healthy`).

Экзаменационный билет №23

Вопрос 1

Правила постановки задачи ИИ-ассистенту при генерации кода и тестов: контекст, ограничения, формат результата, примеры данных.

Вопрос 2

Сравнение Bash и Python для задач автоматизации: критерии выбора инструмента.

Практическое задание

ИИ-ассистент по запросу «функция очистки старых копий» предложил код:

```
import os


def clean(path, days):
    os.system("find " + path + " -mtime +" + str(days) + " -exec rm -rf {}
\;")
```

1. Найдите не менее четырёх проблем в предложенном коде и опишите возможные последствия.
2. Перепишите функцию безопасно средствами `pathlib` (удалять только файлы `*.sql.gz` старше заданного числа суток) с журналированием.
3. Опишите порядок проверки такой функции перед использованием на рабочем сервере.

Эталон ответа для экзаменатора. Проблемы: `os.system` со склейкой строки (внедрение команд); `rm -rf` без ограничения типа и маски (удалит каталоги и любые файлы, включая сам каталог `path`); при пустом `path` поиск идёт в текущем каталоге; нет проверки `days`; нет журнала и обработки ошибок. Безопасный вариант: `root = Path(path).resolve(); limit = time.time() - days * 86400; for f in root.glob("*.sql.gz"): if f.is_file() and f.stat().st_mtime < limit: f.unlink(); log.info("удалён %s", f)` с проверкой `days >= 1`. Проверка: тестовый каталог с файлами разного возраста, модульный тест на временном каталоге (`tmp_path`), сначала режим «только вывод списка».

Экзаменационный билет №24

Вопрос 1

Требования Федерального закона №152-ФЗ, академическая добросовестность и указание использования ИИ при выполнении учебных и рабочих задач.

Вопрос 2

Виды тестирования. Пирамида тестов. Методы верификации программного обеспечения.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Практическое задание

ИИ-ассистент сгенерировал функцию определения високосного года:

```
def is_leap(year):
    return year % 4 == 0 and year % 100 != 0

print([is_leap(y) for y in (1900, 2000, 2023, 2024)])
```

1. Определите вывод программы и найдите ошибку относительно календарных правил.
2. Напишите тесты pytest, которые выявляют ошибку.
3. Исправьте функцию. Что вы сделаете, чтобы не доверять сгенерированному коду без проверки?

Эталонный вывод (для экзаменатора):

```
[False, False, False, True]
```

Эталон ответа для экзаменатора. Год кратный 400 является високосным (2000): правильное условие `year % 4 == 0 and (year % 100 != 0 or year % 400 == 0)`. Ожидаемый результат после исправления: `[False, True, False, True]`. Меры: тесты на границах и известных исключениях, сверка с документацией и правилами предметной области, ревью, указание использования ИИ.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

5.3 КОМПЛЕКТ «ТЕКУЩИЙ КОНТРОЛЬ»

5.3.1 Задания для срезов знаний

1. Какие команды Git переносят изменения из рабочего каталога в индекс и из индекса в репозиторий? Что показывает `git status`?
2. Что такое конфликт слияния, как он выглядит в файле и как его разрешить?
3. Запишите ожидаемый результат: `x = 17; print(x // 5, x % 5, x / 5)`.
4. Объясните, что произойдёт при вызове `f(3)` для функции `def f(n): return n * f(n - 1)`. Как исправить функцию?
5. Какова сложность линейного и бинарного поиска? Какое условие необходимо для бинарного поиска?
6. Для чего в `open(...)` используют `with` и параметр `encoding`? Что произойдёт при чтении файла в неверной кодировке?
7. Чем параметризованный запрос отличается от склейки строки и почему он безопасен?
8. Какой код состояния HTTP возвращает сервер при отсутствии ресурса, при ошибке сервера и при успешном создании ресурса?
9. Что такое наследование и полиморфизм? Приведите пример из предметной области сопровождения информационных систем.
10. Назовите три причины, по которым код, сгенерированный ИИ-ассистентом, нельзя использовать без проверки, и один способ снижения каждого риска.

5.3.2 Тестовые задания

Текущий контроль знаний по Git и GitVerse (раздел 1) проводится в системе дистанционного обучения Moodle по банкам тестовых вопросов курса; по остальным разделам тесты составлены для специальности 09.02.12 (Python, Bash, Docker, ИИ-ассистенты). Типы вопросов: один правильный ответ, верно/неверно, краткий ответ. На занятии каждому обучающемуся выдаётся тест из случайно выбранных вопросов. Попыток – 2, время – 15 минут, оценка – по доле правильных ответов (90–100 % – «5», 75–89 % – «4», 60–74 % – «3»).

Банк вопросов	Число вопросов
Тест 0.1. Введение в Git	6
Тест 0.2. Ветвление в Git	6
Тест 0.3. GitVerse — знакомство и удалённые репозитории	6
Тест 0.4. Pull Request и код-ревью	6
Тест 0.5. Автоматизация проверки — GitVerse Actions	6
Тесты по разделам 2–6 (приведены ниже)	30

Всего в банках – 60 вопросов.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Примеры тестовых заданий по Git и GitVerse (банки Moodle)

1. Команда git add file.txt перемещает изменения файла:
   а) Из рабочей директории напрямую в удалённый репозиторий
   б) Из индекса (staging area) в рабочую директорию
   в) Из рабочей директории в индекс (staging area)
   г) Из репозитория в рабочую директорию
   Правильный ответ: в

2. Когда возникает конфликт слияния?
   а) Всегда, когда сливаются две разные ветки
   б) Когда в обеих ветках изменена одна и та же часть одного и того же файла
   в) Только при использовании git rebase, но не git merge
   г) Когда одна из веток не была запушена на сервер
   Правильный ответ: б

3. Что делает команда git remote add origin <url>?
   а) Создаёт новый локальный репозиторий
   б) Связывает локальный репозиторий с удалённым по указанному адресу под именем origin
   в) Скачивает содержимое удалённого репозитория
   г) Удаляет удалённый репозиторий
   Правильный ответ: б

4. Что обязательно должно быть в хорошем описании запроса на слияние?
   а) Только заголовок, описание не нужно
   б) Что сделано и как это проверить
   в) Полная копия всех изменённых файлов
   г) Пароль от репозитория для ревьюера
   Правильный ответ: б

5. За что отвечает блок on: в workflow-файле?
   а) За список шагов, которые будут выполнены
   б) За события, при которых запускается workflow (например, push, pull_request)
   в) За версию используемого Git
   г) За список участников репозитория
   Правильный ответ: б

Тестовые задания по разделам 2–6 (ключи указаны)

6. Каким будет тип результата выражения `3 / 2` в Python?
   а) str
   б) bool
   в) int
   г) float
   Правильный ответ: г

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

7. Чему равны `7 // 2` и `7 % 2`?
   а) 3 и 1
   б) 3 и 0
   в) 4 и 1
   г) 3,5 и 1
   Правильный ответ: а

8. Какой цикл выводит числа от 1 до 5 включительно?
   а) `for i in range(1, 6)`
   б) `for i in range(5)`
   в) `for i in range(1, 5)`
   г) `for i in range(6)`
   Правильный ответ: а

9. Что выведет `a = [1, 2, 3, 4, 5]; print(a[1:4])`?
   а) `[1, 2, 3]`
   б) `[2, 3, 4, 5]`
   в) `[2, 3, 4]`
   г) `[1, 2, 3, 4]`
   Правильный ответ: в

10. Какая структура данных Python обеспечивает в среднем поиск за O(1) по ключу?
    а) кортеж (tuple)
    б) строка (str)
    в) список (list)
    г) словарь (dict)
    Правильный ответ: г

11. Какой элемент блок-схемы по ГОСТ 19.701-90 обозначает проверку условия?
    а) параллелограмм
    б) прямоугольник
    в) ромб
    г) овал (начало/конец)
    Правильный ответ: в

12. Сколько сравнений порядка потребует бинарный поиск в отсортированном списке из 1 000 000 элементов?
    а) около 1 000 000
    б) около 1000
    в) около 20
    г) около 500 000
    Правильный ответ: в

13. Что вернёт функция Python, в которой нет оператора `return`?
    а) `None`
    б) `False`

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

в) `0`

г) пустую строку

Правильный ответ: а

14. Какая конструкция гарантирует закрытие файла даже при возникновении исключения?

а) `f = open(...)` без дальнейших действий

б) `with open(...) as f:`

в) `f.read()`

г) `import os`

Правильный ответ: б

15. Что будет выведено: `try: int("abc") / except ValueError: print("ошибка") / finally: print("конец")`?

а) трассировка ошибки без вывода текста

б) только «ошибка»

в) «ошибка», затем «конец»

г) только «конец»

Правильный ответ: в

16. Какой из уровней модуля logging имеет наибольшую важность?

а) WARNING

б) CRITICAL

в) DEBUG

г) INFO

Правильный ответ: б

17. Как защитить программу от SQL-инъекции?

а) отключить журналирование

б) заменять кавычки вручную

в) передавать значения параметрами запроса, а не склейкой строк

г) использовать f-строки для подстановки значений

Правильный ответ: в

18. Какой код состояния HTTP означает «ресурс не найден»?

а) 404

б) 500

в) 200

г) 403

Правильный ответ: а

19. Зачем в `requests.get(url, timeout=5)` указывают таймаут?

а) чтобы получить JSON вместо HTML

б) чтобы зашифровать запрос

в) чтобы ускорить сервер

г) чтобы программа не зависла при недоступном сервере

Правильный ответ: г

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

20. Что делает вызов `super().__init__(name)` в конструкторе подкласса?
    а) запрещает наследование
    б) создаёт копию объекта
    в) удаляет родительский класс
    г) вызывает конструктор родительского класса
    Правильный ответ: г
21. Что такое полиморфизм?
    а) возможность вызывать один метод у объектов разных классов с различным поведением
    б) скрытие внутреннего состояния объекта
    в) создание нескольких объектов одного класса
    г) хранение данных в файлах
    Правильный ответ: а
22. О чём говорит принцип единственной ответственности (SOLID)?
    а) у класса должна быть одна причина для изменения
    б) класс должен содержать как можно больше методов
    в) все классы должны наследоваться от одного базового
    г) запрещено использовать композицию
    Правильный ответ: а
23. В чём смысл внедрения зависимостей через конструктор?
    а) зависимости устанавливаются командой pip
    б) ускоряется запуск интерпретатора
    в) класс скрывает свои атрибуты
    г) класс получает зависимости извне, что упрощает замену и тестирование
    Правильный ответ: г
24. Какая конструкция pytest проверяет, что код выбрасывает исключение?
    а) `pytest.skip()`
    б) `try: ... finally`
    в) `with pytest.raises(ValueError):`
    г) `assert ValueError`
    Правильный ответ: в
25. Что показывает покрытие кода тестами?
    а) число ошибок в программе
    б) долю строк (ветвей) кода, выполненных тестами
    в) скорость выполнения тестов
    г) размер репозитория
    Правильный ответ: б
26. Что обеспечивает режим `set -euo pipefail` в сценарии Bash?
    а) ускорение выполнения сценария
    б) запрет вывода сообщений
    в) завершение при ошибке команды, использовании необъявленной

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

переменной и ошибке в конвейере
г) автоматическое резервное копирование
Правильный ответ: в

27. Чем опасен вызов `subprocess.run(cmd, shell=True)` со строкой, содержащей пользовательский ввод?
а) возможностью внедрения посторонних команд
б) замедлением работы интерпретатора
в) невозможностью получить код возврата
г) переполнением диска
Правильный ответ: а

28. Где в Linux находятся сведения о процессах и ресурсах в виде файлов?
а) `/proc`
б) `/boot`
в) `/srv`
г) `/media`
Правильный ответ: а

29. Где хранятся файлы workflow GitVerse Actions в репозитории?
а) в каталоге `.gitverse/workflows/`
б) в каталоге `.venv/`
в) в файле `requirements.txt`
г) в каталоге `/etc/`
Правильный ответ: а

30. Какие данные недопустимо передавать в запросе к ИИ-ассистенту?
а) вымышленные примеры данных
б) пароли, ключи доступа, персональные данные и конфиденциальный код
в) общие вопросы о синтаксисе языка
г) публичные фрагменты документации
Правильный ответ: б

31. Что необходимо сделать с кодом, сгенерированным ИИ-ассистентом, перед использованием?
а) удалить все комментарии
б) прочитать, проверить по документации и протестировать в изолированной среде
в) сразу запустить на рабочем сервере
г) передать другому ассистенту без чтения
Правильный ответ: б

32. Какой российский ИИ-ассистент для разработки интегрирован с платформой GitVerse?
а) systemd
б) Docker Compose
в) Wireshark

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

г) GigaCode

Правильный ответ: г

33. Верно ли, что допускается использовать ИИ-ассистента на экзамене по дисциплине? *Правильный ответ: нет*
34. Назовите команду Git, выводящую историю коммитов. *Правильный ответ: git log*
35. Какое исключение возникает при бесконечной рекурсии в Python? *Правильный ответ: RecursionError*

5.3.3 Задания для проверочных, самостоятельных и контрольных работ

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Контрольная работа №1. Git, GitVerse и автоматическая проверка

Охватывает: тему 1.1 и 1.2 (среда разработки, Git, GitVerse, запросы на слияние, GitVerse Actions); лабораторные работы №1–6. **Формат:** практическая часть и теоретическая часть. **Время:** 45–60 минут. Максимум – 20 баллов. Использование ИИ-ассистентов не допускается.

Часть 1. Практическое задание (10 баллов)

Преподаватель выдаёт доступ к учебному репозиторию `control-work-1` с историей из двух веток (`main` и `feature/limits`), которые содержат конфликтующие изменения файла `config.py`: в ветке `main` задано `TIMEOUT = 5`, в ветке `feature/limits` – `TIMEOUT = 10` и добавлена константа `RETRIES = 3`.

Требуется:

1. Склонировать репозиторий (по SSH).
2. Слить ветку `feature/limits` в `main`, разрешив конфликт так, чтобы значение `TIMEOUT` определялось по критерию, названному преподавателем (например, `TIMEOUT = 10`), а константа `RETRIES = 3` сохранилась.
3. Оформить результат коммитом с сообщением формата `тип: описание` (например, `fix: объединить настройки таймаута`).
4. Опубликовать ветку и оформить запрос на слияние по шаблону: заголовок, разделы «что сделано» и «как проверить».
5. Добавить в `.gitverse/workflows/ci.yml` шаг, проверяющий, что сообщение последнего коммита не короче 10 символов (условие Bash в `run:`).
6. Добиться успешного статуса проверки на запросе на слияние.

Критерий	Баллы
Конфликт разрешён корректно, смысл обеих версий сохранён	3
Коммит оформлен по формату `тип: описание`	1
Запрос на слияние оформлен по шаблону	2
Добавленный шаг CI корректно проверяет длину сообщения	3
Проверка проходит успешно (зелёный статус)	1

Часть 2. Теоретические вопросы (10 баллов)

Ответьте письменно, кратко (2–4 предложения). Каждый вопрос – 2 балла.

1. Объясните разницу между `git merge` и `git rebase`. Какую команду следует использовать для ветки, уже опубликованной и используемой другими участниками, и почему?
2. Назовите три зоны Git и команды, переносящие изменения между ними.
3. Что содержит запрос на слияние хорошего качества и зачем нужны комментарии к конкретным строкам кода?
4. Для чего в workflow нужен шаг `actions/checkout@v4`? Что произойдёт, если его пропустить?

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

5. Какие данные нельзя помещать в репозиторий (даже в закрытый) и как этого избежать?

Ключ и критерии (для преподавателя)

Часть 1

– Пример шага CI:

```
      - name: Проверка сообщения коммита
        run: |
          msg=$(git log -1 --format=%s)
          [ ${#msg} -ge 10 ] || { echo "Сообщение слишком короткое"; exit
1; }
```

– Разрешённый конфликт не должен содержать маркеров `<<<<<<<`, `=======`, `>>>>>>>`; в файле присутствуют `TIMEOUT = 10` и `RETRIES = 3`.

– Шаблон запроса на слияние: заголовок в формате `тип: описание`, разделы «Что сделано» и «Как проверить». Проверяется вручную.

Часть 2

– 1: `merge` создаёт коммит слияния и сохраняет историю, `rebase` переписывает историю, перенося коммиты поверх другой ветки; для опубликованной ветки используется `merge`, так как переписывание истории нарушает работу коллег.

– 2: рабочий каталог, индекс (staging area), репозиторий; `git add` – из рабочего каталога в индекс, `git commit` – из индекса в репозиторий, `git restore --staged` – возврат из индекса.

– 3: описание «что сделано» и «как проверить», небольшой объём изменений; комментарий к строке привязан к месту кода и позволяет обсуждать конкретное решение.

– 4: шаг загружает код репозитория на машину выполнения; без него команды проверки не найдут файлов проекта.

– 5: пароли, ключи API, токены, файлы `.env`, персональные данные; использовать `.gitignore`, переменные окружения и секреты платформы; при утечке – немедленно сменить секрет.

Шкала оценивания

Баллы	Оценка
18–20	«5» (отлично)
14–17	«4» (хорошо)
10–13	«3» (удовлетворительно)
менее 10	«2» (неудовлетворительно)

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Контрольная работа №2. Алгоритмизация и структурное программирование

Охватывает: темы 2.1–2.4 (алгоритмы, типы и структуры данных, ветвления и циклы, функции, рекурсия, поиск и сортировка); лабораторные работы №7–18. **Время:** 90 минут. Язык – Python 3. Использование ИИ-ассистентов не допускается. Максимум – 30 баллов (пять заданий по 6 баллов).

Вариант 1

1. Напишите функцию `stats(marks)`, которая принимает список оценок и возвращает кортеж (средний балл с округлением до сотых, число оценок «5», максимальная оценка); для пустого списка возвращает `None`. Составьте блок-схему или словесное описание алгоритма. Проверка: `stats([5, 4, 5, 3]) == (4.25, 2, 5)`.
2. Напишите функцию `normalize_mac(s)`, приводящую MAC-адрес вида `AA-BB-CC-DD-EE-FF` (допустим также разделитель `:`) к виду `aa:bb:cc:dd:ee:ff`; при некорректной строке функция выбрасывает `ValueError`. Используйте регулярные выражения или методы строк.
3. Дан список строк журнала вида `10.0.0.1 GET /a 200 10` (предпоследнее поле – код ответа). Напишите функцию `count_codes(lines)`, возвращающую словарь «код – число строк». Проверка: для четырёх строк с кодами 200, 404, 200, 500 результат `{'200': 2, '404': 1, '500': 1}`.
4. Напишите рекурсивную функцию `digit_sum(n)` – сумма цифр неотрицательного целого числа. Проверка: `digit_sum(2026) == 10`. Укажите базовый случай и оцените глубину рекурсии для числа из 20 цифр.
5. Напишите функцию `selection_sort(a)`, возвращающую новый отсортированный список методом выбора, не изменяя исходный. Укажите временную сложность алгоритма и объясните, почему на списке из 100 000 элементов следует применять встроенную сортировку.

Вариант 2

1. Напишите функцию `stats2(times)`, которая принимает список времени отклика (мс) и возвращает кортеж (среднее с округлением до десятых, минимум, число значений больше 100); для пустого списка возвращает `None`. Составьте блок-схему или словесное описание алгоритма. Проверка: `stats2([120, 80, 95, 200]) == (123.8, 80, 2)`.
2. Напишите функцию `ip_to_int(ip)`, преобразующую IPv4-адрес в целое число (например, `192.168.0.1` → `3232235521`); при некорректном адресе (не четыре части, октет вне диапазона 0–255) выбрасывается `ValueError`.
3. Дан список строк журнала вида `10.0.0.1 GET /a 200 10` (первое поле – IP-адрес). Напишите функцию `top_hosts(lines, n)`, возвращающую список из `n` пар «адрес – число запросов» по убыванию числа запросов (при равенстве – по возрастанию адреса). Проверка: для четырёх строк журнала (адреса

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

`10.0.0.1`, `10.0.0.2`, `10.0.0.1`, `10.0.0.3`) и `n = 2` результат `[('10.0.0.1', 2), ('10.0.0.2', 1)]`.

4. Напишите рекурсивную функцию `gcd(a, b)` – наибольший общий делитель по алгоритму Евклида. Проверка: `gcd(48, 36) == 12`. Укажите базовый случай.
5. Напишите функцию `insertion_sort(a)`, возвращающую новый отсортированный список методом вставок. Укажите сложность алгоритма в худшем и в лучшем случае.

Ключ и критерии (для преподавателя)

Эталонные решения проверены выполнением на Python 3.11 (приведённые проверочные утверждения выполняются).

Вариант 1

```
import re


def stats(marks):
    if not marks:
        return None
    return round(sum(marks) / len(marks), 2), marks.count(5), max(marks)


def normalize_mac(s):
    parts = re.split(r"[-:]", s.strip().lower())
    if len(parts) != 6 or not all(re.fullmatch(r"[0-9a-f]{2}", p) for p in
parts):
        raise ValueError("некорректный MAC-адрес")
    return ":".join(parts)


def count_codes(lines):
    result = {}
    for line in lines:
        code = line.split()[-2]
        result[code] = result.get(code, 0) + 1
    return result


def digit_sum(n):
    return n if n < 10 else n % 10 + digit_sum(n // 10)


def selection_sort(a):
    a = a[:]
    for i in range(len(a) - 1):
        m = i
        for j in range(i + 1, len(a)):
            if a[j] < a[m]:
                m = j
        a[i], a[m] = a[m], a[i]
    return a
```

Пояснения: базовый случай рекурсии – `n < 10`; глубина для 20-значного числа – 20 вызовов; сложность сортировки выбором – $O(n^2)$, встроенная сортировка (Timsort) – $O(n \log n)$.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Вариант 2

```
from collections import Counter

def stats2(times):
    if not times:
        return None
    return round(sum(times) / len(times), 1), min(times), sum(1 for t in
times if t > 100)

def ip_to_int(ip):
    parts = ip.split(".")
    if len(parts) != 4:
        raise ValueError("некорректный IP-адрес")
    nums = [int(p) for p in parts]
    if any(not 0 <= n <= 255 for n in nums):
        raise ValueError("октет вне диапазона")
    result = 0
    for n in nums:
        result = result * 256 + n
    return result

def top_hosts(lines, n):
    c = Counter(line.split()[0] for line in lines)
    return sorted(c.items(), key=lambda p: (-p[1], p[0]))[:n]

def gcd(a, b):
    return a if b == 0 else gcd(b, a % b)

def insertion_sort(a):
    a = a[:]
    for i in range(1, len(a)):
        key = a[i]
        j = i - 1
        while j >= 0 and a[j] > key:
            a[j + 1] = a[j]
            j -= 1
        a[j + 1] = key
    return a
```

Пояснения: базовый случай `gcd` – `b == 0`; сортировка вставками – $O(n^2)$ в худшем случае и $O(n)$ в лучшем (список уже отсортирован). Некорректный октет вида `abc` вызывает `ValueError` при `int()` (допустимо).

Критерии (каждое задание – 6 баллов)

Баллы	Показатель
6	Решение корректно работает на проверочных данных и граничных случаях (пустой список, некорректный ввод), пояснения верны
4–5	Решение в целом верно, есть неточность в граничном случае или пояснении
2–3	Идея верна, но решение не проходит проверку или не завершено
0–1	Задание не выполнено или не соответствует условию

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Шкала оценивания

Баллы	Оценка
27–30	«5» (отлично)
21–26	«4» (хорошо)
15–20	«3» (удовлетворительно)
менее 15	«2» (неудовлетворительно)

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Контрольная работа №3. Файлы, исключения, журналирование, базы данных и REST-клиенты

Охватывает: тему 3.1 и 3.2 (файлы, форматы CSV и JSON, исключения, журналирование, SQLite, параметризованные запросы, HTTP-клиент); лабораторные работы №19–26. **Время:** 90 минут. Язык – Python 3. Допускается документация Python (оффлайн). Использование ИИ-ассистентов не допускается. Максимум – 30 баллов.

Вариант 1

1. **(8 баллов)** Напишите функцию `convert(csv_path, json_path)`: читает CSV-файл с заголовком `host,status,ms`, для каждого узла вычисляет число проверок (`checks`) и среднее время отклика (`avg_ms`, округление до десятых) и записывает результат в JSON-файл (UTF-8, с отступами, русские символы не экранируются). Если исходный файл не найден, функция записывает ошибку в журнал и возвращает 2, при успехе возвращает 0.
2. **(6 баллов)** Настройте журналирование: сообщения уровня INFO и выше записываются в файл `app.log` в формате «время, уровень, сообщение». Напишите функцию `parse_port(s)`, которая возвращает номер порта (1–65535) и записывает предупреждение в журнал и выбрасывает `ValueError` при некорректном значении.
3. **(8 баллов)** Создайте в базе SQLite таблицу `incident` (`id` – первичный ключ, `host`, `minutes` – неотрицательное целое), добавьте три записи (`web-01` – 45, `web-01` – 20, `db-01` – 120) и напишите функцию `downtime(conn, host)`, возвращающую суммарное время простоя хоста (0, если записей нет) с использованием параметризованного запроса. Объясните, что вернёт функция при `host = "x' OR '1'='1"`.
4. **(8 баллов)** Напишите функцию `fetch_status(get, url, attempts=3, pause=1)`, которая запрашивает адрес вызовом `get(url, timeout=5)` и возвращает значение поля `status` из JSON-ответа. При сетевых сбоях и кодах 5xx выполняются повторные попытки с задержкой `pause * 2**попытка`; при коде 4xx повторять запрос не следует; во всех неудачных случаях возвращается строка `unknown`, причины записываются в журнал. Функция `get` передаётся параметром, чтобы её можно было заменить заглушкой в тесте (объекты ответа имеют атрибуты `status_code` и метод `json()`).

Вариант 2

1. **(8 баллов)** Напишите функцию `load_inventory(path)`: читает CSV-файл с заголовком `host,os,ram_gb`, преобразует `ram_gb` в целое число; строки с некорректным значением пропускаются с записью предупреждения в журнал. Функция возвращает пару (список словарей, число пропущенных строк). Файл читается в UTF-8.
2. **(6 баллов)** Напишите функцию `save_json(data, path)`, записывающую данные в JSON-файл (UTF-8, с отступами) и создающую недостающие

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

родительские каталоги средствами `pathlib`; при ошибке записи функция записывает сообщение уровня ERROR в журнал и повторно выбрасывает исключение.

3. **(8 баллов)** Создайте в базе SQLite таблицу `backup` (`id` – первичный ключ, `db_name`, `size_mb`), добавьте три записи (`shop` – 500, `shop` – 520, `hr` – 80) и напишите функцию `avg_size(conn, name)`, возвращающую средний размер копий базы (0, если копий нет) с использованием параметризованного запроса. Объясните, почему нельзя формировать запрос f-строкой.
4. **(8 баллов)** Напишите функцию `fetch_status(get, url, attempts=3, pause=1)` с теми же требованиями, что в варианте 1, и составьте три теста с заглушкой функции `get`: успех с первой попытки; успех после двух ответов 503; ответ 404 (ровно один вызов `get`).

Ключ и критерии (для преподавателя)

Эталонные решения проверены выполнением на Python 3.11 (с заглушками вместо сети).

Вариант 1

```
import csv, json, logging, sqlite3, time
import requests

logging.basicConfig(filename="app.log", level=logging.INFO,
                    format="%(asctime)s %(levelname)s %(message)s")
log = logging.getLogger("kr3")

def convert(csv_path, json_path):
    try:
        with open(csv_path, newline="", encoding="utf-8") as f:
            rows = list(csv.DictReader(f))
    except FileNotFoundError:
        log.error("файл не найден: %s", csv_path)
        return 2
    result = {}
    for row in rows:
        item = result.setdefault(row["host"], {"checks": 0, "total": 0})
        item["checks"] += 1
        item["total"] += int(row["ms"])
    report = {h: {"checks": v["checks"], "avg_ms": round(v["total"] /
v["checks"], 1)}
              for h, v in result.items()}
    with open(json_path, "w", encoding="utf-8") as f:
        json.dump(report, f, ensure_ascii=False, indent=2)
    return 0

def parse_port(s):
    try:
        port = int(s)
        if not 1 <= port <= 65535:
            raise ValueError("порт вне диапазона")
    except ValueError as e:
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

```
        log.warning("некорректный порт %r: %s", s, e)
        raise
    return port


def downtime(conn, host):
    cur = conn.execute("SELECT COALESCE(SUM(minutes), 0) FROM incident
WHERE host = ?", (host,))
    return cur.fetchone()[0]


def fetch_status(get, url, attempts=3, pause=1):
    for attempt in range(attempts):
        try:
            r = get(url, timeout=5)
        except (requests.ConnectionError, requests.Timeout) as e:
            log.warning("попытка %d: %s", attempt + 1, e)
        else:
            if r.status_code < 400:
                try:
                    return r.json().get("status", "unknown")
                except ValueError:
                    log.error("ответ не является JSON")
                    return "unknown"
            if r.status_code < 500:
                log.error("ошибка клиента: %d", r.status_code)
                return "unknown"
            log.warning("попытка %d: код %d", attempt + 1, r.status_code)
        if attempt < attempts - 1:
            time.sleep(pause * 2 ** attempt)
    return "unknown"
```

Таблица: `CREATE TABLE incident (id INTEGER PRIMARY KEY, host TEXT NOT NULL, minutes INTEGER NOT NULL CHECK (minutes >= 0))`, добавление — `executemany("INSERT INTO incident (host, minutes) VALUES (?, ?)", [...])` и `conn.commit()`. Результат: `downtime(conn, "web-01") == 65`, `downtime(conn, "db-01") == 120`. При `host = "x' OR '1'='1"` функция вернёт 0, так как значение передаётся как данные, а не как часть запроса. Для `convert` на входе из трёх строк (`web-01` — 120 и 80, `db-01` — 0) получается `{"web-01": {"checks": 2, "avg_ms": 100.0}, "db-01": {"checks": 1, "avg_ms": 0.0}}`.

Вариант 2

```
import csv, json, logging, sqlite3
from pathlib import Path


log = logging.getLogger("kr3")


def load_inventory(path):
    items, bad = [], 0
    with open(path, newline="", encoding="utf-8") as f:
        for row in csv.DictReader(f):
            try:
                row["ram_gb"] = int(row["ram_gb"])
            except ValueError:
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

```
                log.warning("некорректное значение ram_gb: %r",
row["ram_gb"])
                bad += 1
                continue
            items.append(row)
    return items, bad


def save_json(data, path):
    path = Path(path)
    try:
        path.parent.mkdir(parents=True, exist_ok=True)
        with open(path, "w", encoding="utf-8") as f:
            json.dump(data, f, ensure_ascii=False, indent=2)
    except OSError as e:
        log.error("не удалось записать %s: %s", path, e)
        raise


def avg_size(conn, name):
    row = conn.execute("SELECT AVG(size_mb) FROM backup WHERE db_name = ?",
(name,)).fetchone()
    return row[0] if row[0] is not None else 0
```

Результаты: для файла из строк `web-01,Debian,16`, `db-01,Ubuntu,abc`, `backup-01,Debian,8` функция `load_inventory` вернёт два словаря и значение 1; `avg_size(conn, "shop") == 510`, `avg_size(conn, "hr") == 80`, `avg_size(conn, "none") == 0`. Склейка запроса f-строкой допускает SQL-инъекцию: значение становится частью текста запроса. Функция `fetch_status` – как в варианте 1. Тесты задания 4: функция-заглушка возвращает объект с `status_code` и `json()`; для двух ответов 503 и затем 200 проверяется результат `"up"`, для ответа 404 – результат `"unknown"` и счётчик вызовов, равный 1 (параметр `pause=0`).

Критерии

Задание	Показатели
Задания 1 и 4 (по 8 баллов)	корректность логики – 4; обработка ошибок и граничных случаев – 2; журналирование и оформление – 2
Задание 2 (6 баллов)	корректность – 3; обработка ошибок и журнал – 3
Задание 3 (8 баллов)	схема таблицы и данные – 3; параметризованный запрос – 3; объяснение – 2

Шкала оценивания

Баллы	Оценка
27–30	«5» (отлично)
21–26	«4» (хорошо)
15–20	«3» (удовлетворительно)
менее 15	«2» (неудовлетворительно)

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Контрольная работа №4. ООП, тестирование, сценарии автоматизации и проверка кода ИИ-ассистента

Охватывает: темы 4.1–4.4 и 5.1–5.2 (ООП, чистый код, тестирование, Bash, процессы и Docker, проверка кода ИИ-ассистента); лабораторные работы №27–42. **Время:** 90 минут. Использование ИИ-ассистентов не допускается. Максимум – 30 баллов.

Вариант 1

1. **(8 баллов)** Опишите класс `Server` с атрибутами `name` и `cpu`, методом `describe()` (возвращает строку `<имя>: <cpu> CPU`) и методом `__str__`, возвращающим результат `describe()`. Опишите класс `WebServer(Server)` с дополнительным атрибутом `port` (по умолчанию 80), переопределяющий `describe()` так, чтобы к строке базового класса добавлялось `, port <порт>`. Проверка: для объектов `Server("db-01", 8)`, `WebServer("web-01", 4)`, `WebServer("web-02", 2, 8080)` строки равны `db-01: 8 CPU`, `web-01: 4 CPU, port 80`, `web-02: 2 CPU, port 8080`. Назовите принцип ООП, реализованный при переопределении метода.
2. **(7 баллов)** Дана функция `percent(used, total)`, возвращающая долю в процентах с округлением до десятых и выбрасывающая `ValueError` при `total <= 0`. Напишите параметризованные тесты pytest: не менее четырёх обычных и граничных случаев и проверку исключения для двух недопустимых значений `total`.
3. **(7 баллов)** Напишите Bash-сценарий `cleanup.sh КАТАЛОГ ДНЕЙ`, удаляющий в каталоге (без вложенных) файлы `*.tar.gz` старше указанного числа суток и записывающий в `КАТАЛОГ/cleanup.log` строку с временем и числом удалённых файлов. Требования: режим `set -euo pipefail`; код возврата 2 при неверном числе аргументов или нечисловом значении `ДНЕЙ`, код 1 при отсутствии каталога.
4. **(8 баллов)** ИИ-ассистент предложил функцию перезапуска контейнера:

```
import subprocess


def restart(name):
    subprocess.run("docker restart " + name, shell=True)
    print("OK")
```

Найдите не менее четырёх проблем и напишите исправленный вариант (проверка имени, список аргументов, таймаут, проверка кода возврата, журнал).

Вариант 2

1. **(8 баллов)** Опишите класс `Server` с атрибутами `name` и `cpu`, методом `describe()` (возвращает строку `<имя>: <cpu> CPU`) и методом `__str__`. Опишите класс `Database(Server)` с дополнительным атрибутом `engine` (по умолчанию `MariaDB`), переопределяющий `describe()` так, чтобы к строке базового класса добавлялось `, <СУБД>`. Проверка: для объектов

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

`Server("proxy", 2), Database("db-01", 8), Database("db-02", 4, "SQLite")` строки равны `proxy: 2 CPU`, `db-01: 8 CPU, MariaDB`, `db-02: 4 CPU, SQLite`. Покажите, как внедрить зависимость от способа проверки (передать объект-проверяющий в конструктор) и зачем это нужно для тестирования.

2. **(7 баллов)** Дана функция `parse_size(s)`, переводящая строки `512`, `1K`, `2M`, `1G` (регистр не важен, пробелы по краям допустимы) в число байтов и выбрасывающая `ValueError` для пустой строки и некорректных значений. Напишите параметризованные тесты pytest (не менее четырёх допустимых значений и три недопустимых).
3. **(7 баллов)** Напишите Bash-сценарий `wait.sh URL [ПОПЫТОК]`, который проверяет доступность URL утилитой `curl` (до заданного числа попыток, по умолчанию 10, пауза 1 с) и завершается с кодом 0 при успехе и 1 при исчерпании попыток; при отсутствии аргумента URL выводится справка и код возврата 1. Режим `set -euo pipefail`.
4. **(8 баллов)** ИИ-ассистент предложил функцию получения списка пользователей:

```
import requests

TOKEN = "gc-secret-123456"

def get_users():
    return requests.get("https://api.example.org/users",
                        headers={"Authorization": "Bearer " + TOKEN},
                        verify=False).json()
```

Найдите не менее четырёх проблем и напишите исправленный вариант.

Ключ и критерии (для преподавателя)

Эталонные решения проверены выполнением (классы и тесты – pytest; сценарии – запуском на тестовых данных).

Вариант 1

```
class Server:
    def __init__(self, name, cpu):
        self.name = name
        self.cpu = cpu

    def describe(self):
        return f"{self.name}: {self.cpu} CPU"

    def __str__(self):
        return self.describe()


class WebServer(Server):
    def __init__(self, name, cpu, port=80):
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

```
        super().__init__(name, cpu)
        self.port = port


    def describe(self):
        return super().describe() + f", port {self.port}"
```

Принцип – полиморфизм (переопределение метода); `__str__` базового класса вызывает `describe()` подкласса.

```
import pytest


@pytest.mark.parametrize("used, total, expected", [(0, 10, 0.0), (5, 10,
50.0), (10, 10, 100.0), (1, 3, 33.3)])
def test_percent(used, total, expected):
    assert percent(used, total) == expected


@pytest.mark.parametrize("total", [0, -5])
def test_percent_bad_total(total):
    with pytest.raises(ValueError):
        percent(1, total)
```

```
#!/bin/bash
set -euo pipefail
if [ "$#" -ne 2 ]; then
    echo "Использование: $0 КАТАЛОГ ДНЕЙ" >&2
    exit 2
fi
dir=$1
days=$2
[ -d "$dir" ] || { echo "Нет каталога: $dir" >&2; exit 1; }
[[ "$days" =~ ^[0-9]+$ ]] || { echo "ДНЕЙ должно быть числом" >&2; exit 2;
}
count=$(find "$dir" -maxdepth 1 -type f -name '*.tar.gz' -mtime +"$days" -
print -delete | wc -l)
echo "$(date '+%F %T') удалено архивов: $count" >> "${dir}/cleanup.log"
```

Задание 4. Проблемы: `shell=True` и склейка строки (внедрение команд при `name = "web; rm -rf ~"`); результат команды не проверяется – сообщение «OK» выводится даже при ошибке; нет таймаута; нет журнала и обработки `FileNotFoundError`; имя контейнера не проверяется. Исправление:

```
import logging, re, subprocess


log = logging.getLogger("ops")


def restart(name):
    if not re.fullmatch(r"[A-Za-z0-9][A-Za-z0-9_.-]*", name):
        raise ValueError("недопустимое имя контейнера")
    try:
        subprocess.run(["docker", "restart", name], check=True,
capture_output=True, text=True, timeout=60)
```


ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

```
    except (subprocess.CalledProcessError, subprocess.TimeoutExpired,
FileNotFoundError) as e:
        log.error("не удалось перезапустить %s: %s", name, e)
        return False
    log.info("контейнер %s перезапущен", name)
    return True
```

Вариант 2

```
class Database(Server):
    def __init__(self, name, cpu, engine="MariaDB"):
        super().__init__(name, cpu)
        self.engine = engine

    def describe(self):
        return f"{super().describe()}, {self.engine}"
```

Внедрение зависимости: `class Monitor: def __init__(self, checker): self.checker = checker`; в рабочем коде передаётся объект, выполняющий сетевую проверку, в тесте – заглушка с заранее заданным ответом, поэтому тест не зависит от сети.

```
import pytest

@pytest.mark.parametrize("s, expected", [("512", 512), ("1K", 1024), ("2m",
2097152), (" 1G ", 1073741824)])
def test_parse_size(s, expected):
    assert parse_size(s) == expected

@pytest.mark.parametrize("s", ["", "abc", "1X"])
def test_parse_size_bad(s):
    with pytest.raises(ValueError):
        parse_size(s)
```

```
#!/bin/bash
set -euo pipefail
url=${1:?Использование: $0 URL [ПОПЫТОК]}
tries=${2:-10}
for ((i = 1; i <= tries; i++)); do
    if curl -fsS --max-time 3 -o /dev/null "$url"; then
        echo "сервис доступен (попытка $i)"
        exit 0
    fi
    sleep 1
done
echo "сервис недоступен после $tries попыток" >&2
exit 1
```

Задание 4. Проблемы: токен в тексте программы (попадёт в репозиторий); `verify=False` отключает проверку сертификата (возможна подмена сервера); нет таймаута; не проверяется код ответа (`raise_for_status`) и корректность JSON; нет обработки сетевых ошибок. Исправление:

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

```
import os
import requests

def get_users():
    token = os.environ["API_TOKEN"]
    r = requests.get("https://api.example.org/users",
                     headers={"Authorization": f"Bearer {token}"},
timeout=10)
    r.raise_for_status()
    return r.json()
```

Токен, попавший в репозиторий или в запрос к ИИ-ассистенту, считается скомпрометированным и подлежит замене.

Критерии

Задание	Показатели
Задание 1 (8 баллов)	классы и наследование – 4; корректный вывод – 2; пояснение принципа – 2
Задание 2 (7 баллов)	параметризация и граничные значения – 4; проверка исключений – 3
Задание 3 (7 баллов)	логика сценария – 4; коды возврата и проверки аргументов – 3
Задание 4 (8 баллов)	найденные проблемы (по 1 баллу за каждую, максимум 4) – 4; исправленный вариант – 4

Шкала оценивания

Баллы	Оценка
27–30	«5» (отлично)
21–26	«4» (хорошо)
15–20	«3» (удовлетворительно)
менее 15	«2» (неудовлетворительно)

5.3.4 Лабораторные и практические работы

Лабораторные работы выполняются в лаборатории на рабочих местах с Linux (Debian/Ubuntu) или в виртуальных машинах. Методические указания к работам по Git и GitVerse (ЛР №2–6) размещены в курсе Moodle «Основы алгоритмизации и программирования» и адаптированы под проект на Python; остальные работы разработаны для специальности 09.02.12 на основе тем курса (алгоритмизация, функции, исключения, SQLite, API, ООП, чистый код) с заменой языка PHP на Python и Bash. По итогам каждой работы обучающийся оформляет отчёт (описание решения, вывод программы, результаты тестов) и размещает программы, тесты и сценарии в личном репозитории GitVerse; работы сдаются через запрос на слияние, автоматическая проверка в GitVerse Actions является обязательным условием приёма (с ЛР №6).

Перечень лабораторных работ (90 часов, 45 работ по 2 часа)

№	Лабораторная работа	Цель работы	Источник
1	Среда разработки: Python, виртуальное окружение, pip, запуск и отладка программ в	Установить и настроить интерпретатор Python, создать виртуальное окружение, установить пакет через pip, запустить программу и пройти её в	новая работа

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

№	Лабораторная работа	Цель работы	Источник
	VS Code	отладчике VS Code.	
2	Первый репозиторий и коммиты: цикл init – add – commit – log	Создать репозиторий учебного проекта на Python и освоить базовый цикл Git: init, add, commit, log; оформлять осмысленные сообщения коммитов и файл .gitignore.	Moodle: ЛР 0.1 (адаптация)
3	Ветвление и слияние: ветки, слияние, разрешение конфликтов	Создавать ветки, выполнять слияние, намеренно получить конфликт слияния и разрешить его без потери смысла обеих версий.	Moodle: ЛР 0.2 (адаптация)
4	Публикация проекта в GitVerse: удалённые репозитории, ключи SSH, push и pull	Подключить удалённый репозиторий GitVerse по SSH, опубликовать ветки (push), получить изменения (pull, fetch), оформить файл README.	Moodle: ЛР 0.3 (адаптация)
5	Запросы на слияние (Pull Request) и код-ревью в GitVerse	Оформить запрос на слияние (Pull Request) по шаблону, провести код-ревью чужого запроса с комментариями к строкам кода, внести исправления по замечаниям.	Moodle: ЛР 0.4 (адаптация)
6	GitVerse Actions: автоматическая проверка Python-проекта	Описать workflow GitVerse Actions, запускающий установку зависимостей, проверку кода и тесты Python-проекта при каждом push, и добиться успешного статуса проверки.	Moodle: ЛР 0.5 (адаптация)
7	Алгоритмы и блок-схемы: линейные и разветвляющиеся алгоритмы	Записать алгоритмы решения задач в виде блок-схем и пошагового описания, реализовать линейные и разветвляющиеся алгоритмы на Python и проверить их на тестовых наборах.	новая работа
8	Типы данных Python, ввод-вывод и форматирование	Освоить основные типы данных (int, float, str, bool), преобразование типов, ввод с клавиатуры, форматированный вывод (f-строки), получение числовых характеристик.	Moodle: ЛР 1.1 (адаптация на Python)
9	Условные операторы: разветвляющиеся программы	Реализовать разветвляющиеся алгоритмы с операторами if, elif, else и логическими выражениями; обрабатывать граничные случаи.	Moodle: ЛР 1.2 (адаптация на Python)
10	Циклы: перебор, накопление, поиск	Реализовать алгоритмы с циклами for и while: суммирование, подсчёт, поиск максимума, перебор последовательностей, досрочный выход (break, continue).	Moodle: ЛР 1.2 (адаптация на Python)
11	Списки и вложенные списки (двумерные массивы)	Работать со списками: срезы, методы, генераторы списков; обрабатывать двумерные массивы (таблицы), выполнять поиск и подсчёт по строкам и столбцам.	новая работа
12	Строки и регулярные выражения	Обрабатывать строки (разбиение, поиск, замена, форматирование) и применять регулярные выражения (модуль re) для проверки и извлечения данных из текстов журналов.	Moodle: ЛР 1.9 (адаптация на Python)
13	Словари, множества и кортежи: учёт и подсчёт	Использовать словари, множества и кортежи для подсчёта частот, группировки и удаления дубликатов; сопоставить структуры по скорости поиска.	новая работа
14	Функции: параметры, возвращаемые значения, рефакторинг	Выделять повторяющийся код в функции, использовать параметры по умолчанию, именованные аргументы, возвращаемые значения; документировать функции (docstring).	Moodle: ЛР 1.3 (адаптация на Python)
15	Рекурсия	Реализовать рекурсивные алгоритмы (факториал, числа Фибоначчи, обход вложенных структур), определить базовый случай и глубину рекурсии,	новая работа

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

№	Лабораторная работа	Цель работы	Источник
		сравнить с итеративным решением.	
16	Модули и пакеты	Разделить программу на модули, организовать пакет, использовать импорт и стандартную библиотеку; понимать назначение конструкции if __name__ == '__main__'.	Moodle: ЛР 2.1 (адаптация на Python)
17	Алгоритмы поиска и сортировки	Реализовать линейный и бинарный поиск, сортировку пузырьком и выбором, сравнить с встроенной сортировкой sorted; проверить корректность на тестовых данных.	новая работа
18	Оценка сложности алгоритмов: измерение времени выполнения	Измерить время выполнения алгоритмов (timeit), построить зависимость времени от размера входных данных и сопоставить с теоретической оценкой O(n), O(n log n), O(n²).	новая работа
19	Текстовые файлы, CSV и JSON	Читать и записывать текстовые файлы, файлы CSV (модуль csv) и JSON (модуль json); выбирать кодировку UTF-8, использовать менеджер контекста with.	новая работа
20	Каталоги и файловая система: os, pathlib, shutil	Обходить каталоги, получать сведения о файлах и правах доступа, копировать и перемещать файлы средствами os, pathlib и shutil; написать утилиту поиска крупных и устаревших файлов.	новая работа
21	Обработка исключений	Перехватывать и обрабатывать исключения (try, except, else, finally), создавать собственные классы исключений, проверять входные данные.	Moodle: ЛР 1.8 (адаптация на Python)
22	Журналирование: модуль logging и разбор журнала	Настроить модуль logging (уровни, формат, вывод в файл и в консоль), использовать ротацию журналов и разобрать журнал с подсчётом ошибок по уровням.	новая работа
23	Работа с базой данных SQLite из Python	Создать базу данных SQLite из программы на Python, выполнить создание таблиц, вставку, выборку, изменение и удаление данных с использованием транзакций.	Moodle: ЛР 1.7 (адаптация на Python)
24	Параметризованные запросы и защита от SQL-инъекций	Воспроизвести SQL-инъекцию на учебном коде, устранить уязвимость параметризованными запросами, проверить результат тестами.	новая работа
25	HTTP-клиент: библиотека requests, коды ответа, таймауты	Отправлять HTTP-запросы библиотекой requests (GET, POST), разбирать JSON-ответ, анализировать коды состояния, заголовки; задавать таймаут.	Moodle: ЛР 1.10 (адаптация на Python)
26	Клиент REST API: обработка ошибок и повторные попытки	Реализовать клиент REST API: обработка сетевых ошибок и кодов 4xx и 5xx, повторные попытки с задержкой, журналирование обращений; проверить клиент на учебном сервисе, не раскрывая ключи доступа.	Moodle: ЛР 1.10 (адаптация на Python)
27	Классы и объекты: инкапсуляция, конструкторы, методы	Описать класс предметной области, использовать конструктор, атрибуты и методы, инкапсуляцию (защищённые атрибуты, свойства), специальные методы __str__ и __repr__.	Moodle: ЛР 1.6 (адаптация на Python)
28	Наследование и полиморфизм	Реализовать иерархию классов с наследованием и переопределением методов, абстрактный базовый класс, полиморфное поведение; сопоставить наследование и композицию.	Moodle: ЛР 1.6 (адаптация на Python)
29	Оформление кода: PEP 8, линтер flake8, чистка кода	Привести код к требованиям PEP 8, проверить его линтером flake8, устранить «запахи кода» (длинные функции, дублирование,	Moodle: ЛР 2.2, ЛР 2.3 (адаптация

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

№	Лабораторная работа	Цель работы	Источник
		неинформативные имена).	на Python)
30	Принципы SOLID и внедрение зависимостей	Выполнить рефакторинг учебного кода по принципам SOLID и внедрить зависимости через конструктор для упрощения тестирования.	Moodle: ЛР 2.4, ЛР 2.6 (адаптация на Python)
31	Модульное тестирование с pytest	Писать модульные тесты в pytest: утверждения, параметризованные тесты, фикстуры, проверка исключений; запускать тесты из командной строки.	новая работа
32	Тестовые заглушки (mock) и измерение покрытия	Изолировать зависимости от внешних сервисов и файловой системы с помощью unittest.mock, измерить покрытие кода тестами (pytest-cov) и повысить его.	новая работа
33	Структура проекта, зависимости и документация	Организовать структуру проекта (пакет, тесты, каталог документации), описать зависимости в requirements.txt и pyproject.toml, оформить README и инструкцию запуска.	Moodle: ЛР 2.1 (адаптация на Python)
34	Консольное приложение: аргументы командной строки (argparse)	Разработать консольную утилиту с аргументами и подкомандами (argparse), кодами возврата и справкой; подготовить её к запуску из сценариев автоматизации.	новая работа
35	Bash: анализ журнала средствами grep, awk, sort и uniq	Анализировать журнал веб-сервера и системный журнал конвейером grep, awk, sort, uniq, cut; получить топ ошибок и топ источников запросов.	новая работа
36	Bash-сценарии: параметры, коды возврата, set -euo pipefail и trap	Написать Bash-сценарий с параметрами, проверкой кодов возврата, режимом set -euo pipefail, функциями и перехватом сигналов (trap); проверить сценарий утилитой shellcheck либо вручную.	новая работа
37	Запуск внешних команд из Python: subprocess	Безопасно запускать внешние команды из Python (subprocess.run со списком аргументов, таймаутом, проверкой кода возврата), обрабатывать вывод и ошибки; объяснить риск shell=True.	новая работа
38	Мониторинг ресурсов: чтение /proc из Python и Bash	Получать показатели нагрузки (процессор, память, диски, процессы) из файловой системы /proc и штатных утилит, формировать отчёт и пороговые предупреждения.	новая работа
39	Управление контейнерами Docker из Python	Управлять контейнерами Docker из программы: запуск, остановка, проверка состояния, чтение журнала; оформить сценарий с обработкой ошибок.	новая работа
40	Развёртывание Docker Compose-проекта сценарием и проверка работоспособности	Написать сценарий развёртывания проекта на Docker Compose (запуск, ожидание готовности по healthcheck, проверка конечной точки, откат при ошибке) и подключить его к проверке GitVerse Actions.	новая работа
41	Российские ИИ-ассистенты (GigaChat, GigaCode, Алиса AI) в разработке: постановка задачи, генерация и проверка кода	Формулировать корректные запросы к GigaChat, GigaCode и Алиса AI для генерации функций, тестов и пояснения кода, фиксировать запросы и ответы, соблюдать правила защиты данных и оформлять указание использования ИИ.	новая работа
42	Поиск дефектов в коде, подготовленном ИИ-ассистентом: тесты и ревью	Найти и исправить ошибки в программах, подготовленных ИИ-ассистентом (логические, безопасность, устаревшие вызовы), с помощью тестов и ревью; оформить протокол проверки.	новая работа

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

№	Лабораторная работа	Цель работы	Источник
43	Итоговый проект: постановка задачи, проектирование и репозиторий	Сформулировать требования к итоговому проекту (утилита сопровождения информационной системы), выбрать структуру, создать репозиторий в GitVerse и план задач.	новая работа
44	Итоговый проект: реализация, тесты и проверка в GitVerse Actions	Реализовать функциональность проекта, написать тесты, добиться успешной проверки в GitVerse Actions, оформить запрос на слияние и пройти код-ревью.	новая работа
45	Итоговый проект: защита	Представить проект: демонстрация работы, ответы на вопросы, отчёт об использовании ИИ-ассистентов и результатах проверки сгенерированного кода.	новая работа

Критерии оценки лабораторной работы

Оценка	Показатели
«5»	Все задания выполнены самостоятельно и верно; программы запускаются, тесты проходят; код оформлен по PEP 8; история коммитов осмысленна; обучающийся объясняет решение и отвечает на вопросы; при использовании ИИ-ассистента представлены запросы, найденные недостатки ответа и исправленный вариант, указано использование ИИ.
«4»	Работа выполнена полностью, есть неточности в оформлении, тестах или пояснениях; обучающийся устраняет замечания после наводящих вопросов.
«3»	Работа выполнена не полностью (не менее 70 % заданий) или с существенными ошибками; пояснения неполные; автоматическая проверка не проходит с первой попытки.
«2»	Работа не выполнена или выполнена менее чем на 70 %; обучающийся не может объяснить решение.

Итоговая оценка за лабораторные работы раздела выставляется как среднее арифметическое оценок за работы раздела с учётом своевременности сдачи. Итоговый проект (ЛР №43–45) оценивается по критериям: соответствие требованиям и работоспособность (30 %), качество кода и структура (20 %), тесты и автоматическая проверка (20 %), работа в репозитории: ветки, запросы на слияние, ревью (15 %), защита и отчёт об использовании ИИ (15 %).

5.3.5 Компьютерные программы для контроля знаний студентов

Банки тестовых вопросов по Git и GitVerse и тесты по остальным разделам размещены в системе дистанционного обучения колледжа (Moodle) и импортируются в формате Moodle XML. Для проверки лабораторных работ используется платформа GitVerse с автоматической проверкой (GitVerse Actions) на основе общего сценария оценки; тесты pytest выполняются в контейнере, одинаковом для всех обучающихся. Контрольные работы выполняются на компьютерах без доступа к ИИ-ассистентам и к сети Интернет, кроме локальной документации Python. Экзамен проводится в лаборатории: практическое задание выполняется на бумаге либо в текстовом редакторе без запуска ИИ-ассистентов. Использование ИИ-ассистентов на контрольных работах и экзамене не допускается.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
1 подпись
Сведения о подписях — на листе подписей. Проверка подлинности — по QR-коду.
ГАПОУ СО «Нижнетагильский торгово-экономический колледж»

Лист подписей

ГАПОУ СО «Нижнетагильский торгово-экономический колледж»
SHA-256 исходного документа: 87deb7f55c7cdd10fab6baa716966f8c08c68f615346f123f321805c06e137bf
Документ: 57 страниц
Проверка подписей: https://sig.nttek.ru/v/qFVjVMjU5IByZ5QxiplJkVLNW11AVXSdny201sYNOI6Rn18J
Печатная форма сформирована: 15.10.2026 09:45

№	Роль в подписи	Подписант	Сертификат	Дата и время подписания	Результат проверки
1	Утверждаю	Дмитриева Марина Георгиевна Заместитель директора по учебной работе	№ 7E9C18960643A37EF2C929CD1E513CCD, действителен с 08.10.2026 по 08.10.2027	15.10.2026 09:43:03	**подпись действительна** роль и должность подтверждены подписью

Печатная форма не подписывается и служит для удобства чтения. Юридическую силу имеют исходный PDF-документ и файлы электронных подписей (.sig). Подпись — усиленная неквалифицированная электронная подпись (ч. 3 ст. 5 Федерального закона № 63-ФЗ), выданная удостоверяющим центром колледжа.
Время указано по часовому поясу колледжа (Asia/Yekaterinburg, UTC+5).