Общество с ограниченной ответственностью «ЦЕНТР ОБРАЗОВАТЕЛЬНЫХ ТЕХНОЛОГИЙ»
(ООО «Центр образовательных технологий»)
ИНН 0600023803 ОГРН 1260600000913
386231, Республика Ингушетия, г. Карабулак, ул. Дьякова, д. 108

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ

по выполнению внеаудиторной работы при реализации программы дополнительного общеобразовательного образования

«Подготовка к аттестации за 9 класс для дистанционного семейного обучения»

г. Карабулак,

2026

ВВЕДЕНИЕ

Настоящие методические рекомендации разработаны для обучающихся 9-х классов, завершающих основной курс информатики. В этом году программа достигает максимальной глубины и прикладной направленности, охватывая продвинутые теоретические концепции, такие как кодирование и шифрование информации, моделирование, базы данных и графы, а также профессиональные практические навыки в углубленном программировании на Python, полноценной веб-разработке (HTML, CSS, JavaScript, Tilda), работе с Большими данными, искусственным интеллектом (LLM, промпт-инжиниринг, дообучение), UI/UX дизайне и 3D-моделировании в Blender. Документ призван помочь ученику не только успешно освоить эти сложные и актуальные темы, но и развить компетенции, необходимые для дальнейшего обучения и профессиональной деятельности в IT-сфере.

Материал структурирован таким образом, чтобы обучающийся мог пройти путь от понимания принципов кодирования и моделирования до создания интерактивных веб-сайтов, анализа данных, разработки ИИ-решений и реализации комплексных проектов.

Цель документа

Основная цель рекомендаций — обеспечить глубокое понимание продвинутых теоретических основ информатики, сформировать устойчивые практические навыки в области углубленного программирования на Python, Веб-разработки, работы с Большими данными и ИИ, а также развить компетенции в UI/UX дизайне, 3D-моделировании и создании комплексных цифровых продуктов.

В рамках реализации данной цели предполагается решение следующих задач:

1. Разъяснить принципы кодирования и шифрования информации, научить применять методы моделирования для решения задач.
2. Сформировать понимание структуры и работы баз данных, графов и логических операций.
3. Обучить продвинутым аспектам программирования на Python, включая рефакторинг, отладку, работу с функциями, строками и списками.

4. Развить навыки веб-разработки, включая создание сайтов на Tilda, а также верстку на HTML, CSS и программирование на JavaScript.

5. Обеспечить освоение принципов работы с Большими данными, облачными сервисами, хостингом и доменными именами.

6. Повысить уровень владения инструментами искусственного интеллекта: большие языковые модели, промпт-инжиниринг, дообучение ИИ, применение ИИ в веб-разработке и рефакторинге кода.

7. Сформировать компетенции в графическом дизайне (UI/UX, вайрфреймы, прототипы) и 3D-моделировании (Blender).

Структура документа

Рекомендации разделены на ключевые блоки, соответствующие основным видам учебной деятельности в 9 классе:

1. Контроль знаний (тестирование).

2. Алгоритмический блок: Углубленное программирование на Python: освоение продвинутых концепций Python, отладка и рефакторинг кода.

3. Технологический практикум: Веб-разработка, создание сайтов на Tilda, верстка на HTML/CSS/JS, работа с хостингом и доменами.

4. Искусственный интеллект: продвинутый промпт-инжиниринг, дообучение ИИ, применение ИИ в веб-разработке и рефакторинге.

5. Графический дизайн и 3D-моделирование: UI/UX, вайрфреймы, прототипы, дизайн с ИИ, 3D-моделирование в Blender.

Каждый раздел содержит:

1. Пошаговые алгоритмы действий («чек-листы»).

2. Типичные сценарии выполнения заданий с разбором нюансов.

3. Систему критериев для самопроверки.

Методологические основы

Рекомендации базируются на следующих принципах:

1. Проектно-ориентированный подход — обучение через создание реальных цифровых продуктов (приложения, видео, 3D-модели).
2. Междисциплинарность — интеграция знаний из области дизайна, программирования, логики и этики.
3. Актуальность технологий — использование современных инструментов и платформ (Python, Tilda, HTML/CSS/JS, LLM, Blender, облачные сервисы).
4. Развитие критического мышления — формирование навыков анализа информации, оценки достоверности, этичности и безопасности использования цифровых технологий.
5. Самостоятельность и инициатива — поощрение экспериментов, поиска решений и творческого подхода к задачам.

Целевая аудитория

Методические рекомендации предназначены для обучающихся 9 класса и направлены на поддержку их учебной деятельности по информатике.

Ожидаемые результаты реализации

Использование данных рекомендаций позволит:

1. Уверенно ориентироваться в продвинутых теоретических основах информатики: кодирование, шифрование, моделирование, базы данных, графы, логика;
2. Создавать сложные программы на Python, эффективно используя функции, структуры данных, а также навыки отладки и рефакторинга;
3. Разрабатывать полноценные веб-сайты с использованием Tilda, HTML, CSS и JavaScript, включая адаптивную верстку и интерактивные элементы;
4. Эффективно использовать инструменты искусственного интеллекта для промпт-инжиниринга, дообучения моделей, генерации кода и дизайна;
5. Получить базовые навыки работы с Большими данными, облачными сервисами, хостингом и доменными именами;

6. Развивать навыки UI/UX дизайна, создавая вайрфреймы, прототипы и дизайн мобильных адаптаций сайтов;
7. Освоить основы 3D-моделирования в Blender
8. Развивать самостоятельность в обучении;
9. Формировать ответственное отношение к учебной деятельности;
10. Повышать уровень подготовки к текущему и итоговому контролю.

Особенности применения

Рекомендации следует использовать:

1. Как пошаговое руководство при выполнении практических работ по программированию на Python, веб-разработке, UI/UX дизайну и 3D-моделированию;
2. Как справочник при подготовке к контрольным срезам и тестам, включая теоретические вопросы по кодированию, моделированию, базам данных и логике.
3. Во время выполнения домашних заданий;
4. Для устранения пробелов в знаниях при самостоятельном изучении пропущенных тем.

Структура работы с документом

Для наиболее эффективного использования рекомендаций рекомендуется придерживаться следующего порядка работы:

1. Ознакомиться с теоретическим материалом по изучаемой теме.
2. Открыть соответствующий раздел данных рекомендаций.
3. Выполнить подготовительные действия (установка ПО, подготовка медиафайлов, создание аккаунта в онлайн-сервисе).
4. Следовать алгоритму выполнения задания, обращая внимание на советы по оформлению и технические нюансы.
5. Провести самоаудит работы по критериям оценивания.
6. В случае возникновения ошибок — использовать методы отладки.

1. ТЕСТИРОВАНИЕ

Цели проведения тестирования

Тестирование в информатике 9 класса направлено на оценку глубокого понимания фундаментальных теоретических основ, а также способности применять знания в контексте современных технологий и подготовки к итоговой аттестации:

– проверка понимания принципов кодирования и шифрования информации, умения работать с различными моделями (математическими, компьютерными, табличными);

– оценка знаний по базам данных, графам и продвинутым логическим операциям;

– контроль владения терминологией (Большие данные, Хостинг, Домен, LLM, Промпт-инжиниринг, UI/UX, Flexbox).

– проверка умения анализировать информацию, включая критическую оценку ИИ-решений и эффективности веб-технологий;

– формирование у обучающихся навыков самоконтроля и ответственности за результат.

Планируемые результаты

Обучающийся должен уметь:

– Применять различные способы кодирования и шифрования информации.;

– Использовать моделирование как метод познания, строить математические, компьютерные и табличные модели;

– Понимать структуру и принципы работы баз данных, а также виды графов;

– Применять логические операции и логические выражения для решения задач;

– Понимать базовые принципы работы Python, его синтаксис и основные конструкции, включая рефакторинг и отладку;

– Различать компоненты веб-разработки (HTML, CSS, JS, Tilda, хостинг, домены).

– Понимать основные понятия ИИ: LLM, промпт-инжиниринг, дообучение.

Методические рекомендации

Тесты по информатике 9 класса требуют не только знания определений, но и умения применять теоретические знания для решения комплексных задач, часто с элементами анализа кода или логических выражений. При выполнении следует:

– анализировать визуальные данные, если в вопросе есть изображение окна программы, внимательно изучите активные кнопки и вкладки;

– стремиться «убирать лишнее» в вопросах про устройства или понятия часто можно отсеять заведомо неверные варианты, вспомнив основную функцию или определение.;

– стремиться «оптимизировать» ответ, облегчить содержание и конкретизировать ответ, особенно важно в заданиях с написанием кода;

– обращать особенное внимание к деталям, особенно это касается вопросов по системам счисления и алгоритмам, где важна каждая цифра или порядок действий.

– критически оценивать варианты: в вопросах по Большим данным, ИИ или веб-технологиям часто встречаются правдоподобные, но неверные утверждения

Все задания основаны исключительно на изученном учебном материале, поэтому перед выполнением тестовой работы рекомендуется повторить основные понятия, определения, формулы и правила по соответствующей теме.

Временной регламент

Стандартный тест из 10-15 вопросов рассчитан на 25–30 минут. Практические задания внутри теста могут занимать больше времени.

Пример структуры теста

Общее количество заданий — 10, разделенных на три тематических модуля. Ниже представлен детальный разбор каждого блока с примерами вопросов

Блок 1. Теоретические основы информатики (3 задания):

– Кодирование и шифрование: Принципы кодирования информации (например, «Какой метод кодирования используется для передачи данных по сети?»), шифрование (например, «Что такое шифр Цезаря и как он работает»).

– Моделирование: Виды моделей (математические, компьютерные, табличные), этапы моделирования (например, «Приведите пример математической модели для прогнозирования роста населения).

– Базы данных и графы: Основные понятия баз данных (таблица, запись, поле, ключ), виды графов (например, «Что такое реляционная база данных?», «Назовите основные элементы графа»).

Блок 2. Программирование на Python (4-5 заданий):

– Синтаксис Python: Базовые конструкции, ввод-вывод данных, переменные, типы данных (например, «Что выведет на экран следующий код Python: a = 10; b = 3; print(a // b)?»).

– Условия и циклы: Применение if/else, for, while для решения задач (например, «Напишите фрагмент кода Python, который выводит все четные числа от 1 до 10»).

– Строки и списки: Базовые операции со строками и списками (например, «Как получить третий элемент списка my_list = [1, 5, 9, 12]?»).

– Функции: Понимание принципов работы функций, их вызов (например, «Что делает следующая функция Python: def multiply(x, y): return x * y?»).

Блок 3. Веб-разработка и Современные технологии (3 задания):

– Веб-разработка: Основы HTML (заголовки, абзацы, списки, ссылки, изображения), CSS (стили, классы, ID, отступы, Flexbox), JavaScript (формы, кнопки, интерактивность) (например, «Что такое Flexbox и для чего он используется в CSS?»).

– Tilda: Принципы работы с конструктором сайтов, подключение баз данных, публикация (например, «Опишите процесс публикации сайта, созданного в Tilda»).

– Современные технологии: Понятия Больших данных, облачных сервисов, хостинга, доменных имен, профессий в IT (например, «Что такое Большие данные и где они применяются?»)

– Искусственный интеллект: Большие языковые модели (LLM), промпт-инжиниринг, дообучение ИИ (Fine-Tuning), применение ИИ в веб-разработке.

Уровень	***Описание***	***Пример задания***
Базовый (40%)	Знание терминов, простых определений, базовых синтаксических конструкций Python, основ HTML/CSS, понятий ИИ.	Что такое HTML?
Повышенный (40%)	Применение знаний в знакомой ситуации, анализ фрагментов кода Python/HTML/CSS, решение задач по логике, кодированию, моделированию.	Напишите Python-код, который запрашивает у пользователя имя и выводит приветствие.
Творческий (20%)	Анализ, синтез, решение проблем, критическая оценка, понимание сложных концепций и их применение, например, в контексте веб-разработки или ИИ.	Предложите, как можно использовать ИИ для автоматизации создания контента для сайта, и объясните, какие этапы будут автоматизированы.

2. АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ (УГЛУБЛЕННЫЙ PYTHON)

Цели практических работ

В 9 классе мы углубляем наши знания Python, переходя от базовых конструкций к более сложным концепциям, необходимым для создания полноценных приложений и веб-сервисов. Особое внимание уделяется качеству кода, его отладке и рефакторингу.

Планируемые результаты

Освоение Python заложит фундамент для вашей будущей карьеры в IT:

– Продвинутый синтаксис: углубленное изучение переменных, чисел, операций, операторов сравнения и логических связок;

– Управление потоком: освоение инструкций break и continue для эффективного управления циклами;

– Работа с данными: продвинутые операции со строками и списками, включая методы обработки и преобразования данных;

– Функции: создание и использование функций с различными типами аргументов, возвращаемыми значениями, рекурсия (по желанию);

– Отладка и рефакторинг: развитие навыков поиска, исправления ошибок и улучшения структуры существующего кода.

– Строки и списки: работа с текстовыми данными и коллекциями элементов;

– Функции: понимание принципов модульности кода, создание и вызов собственных функций;

– Отладка кода: развитие навыка поиска и исправления ошибок в Python-коде;

– Практикум: анализ и поиск ошибок в готовом коде, создание более сложных программ на Python.

Основные концепции Python

– Переменные: используются для хранения данных. В Python тип переменной определяется автоматически. Пример: name = "Амина", age = 14;

– Типы данных: Понимать разницу между строками (str), целыми числами (int), числами с плавающей точкой (float), булевыми значениями (bool);

– Операторы: Арифметические (+, -, *, /, //, %, **), сравнения (==, !=, >, <, >=, <=), логические (and, or, not);

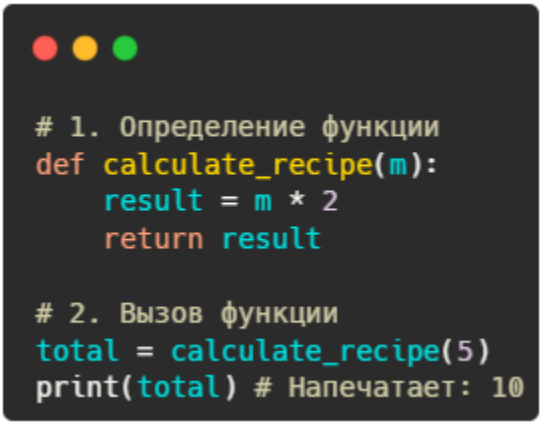
– Условное ветвление: if условие: ... elif условие: ... else: ... — для выполнения разных действий в зависимости от условия;

– Циклы. for item in collection: ... — для перебора элементов коллекции или выполнения действий заданное количество раз. while условие: ... — для выполнения действий, пока условие истинно. Пример: Цикл for для вывода чисел от 1 до 5; цикл while для игры «Угадай число»;

– Строки: Работа с текстовыми данными (конкатенация, индексация, срезы, методы строк). Пример: message = "Привет, мир!"; print(message[0]);

– Списки: Упорядоченные изменяемые коллекции элементов (добавление, удаление, изменение элементов, итерация). Пример: fruits = ["яблоко", "банан", "апельсин"];

– Функции: Блоки кода, которые можно вызывать по имени. Позволяют избежать дублирования кода и делают его более организованным. Пример:



```python
# 1. Определение функции
def calculate_recipe(m):
    result = m * 2
    return result

# 2. Вызов функции
total = calculate_recipe(5)
print(total) # Напечатает: 10
```

Рекомендации по рефакторингу и отладке кода на Python

Рефакторинг — это процесс изменения внутренней структуры программы без изменения её внешнего поведения. Цель — улучшить читаемость, поддерживаемость и расширяемость кода.

Главные аспекты рефакторинга:

– Читаемость: Используйте осмысленные имена переменных и функций (например, calculate_total_price вместо ctp). Добавляйте комментарии для сложных участков кода;

– Функции: Разделяйте большие блоки кода на небольшие, специализированные функции. Каждая функция должна выполнять одну конкретную задачу;

– Устранение дублирования: Если один и тот же фрагмент кода встречается несколько раз, вынесите его в отдельную функцию.

– Оптимизация: Ищите более эффективные алгоритмы или встроенные функции Python для решения задач

Главные аспекты отладки кода:

– Сообщения об ошибках (Traceback): Внимательно читайте Traceback. Он указывает на тип ошибки (например, SyntaxError, NameError, TypeError) и строку, где она произошла.;

– print() для отладки: Вставляйте print() в разных местах кода, чтобы выводить значения переменных и отслеживать ход выполнения программы. Это самый простой и эффективный способ.;

– Пошаговое выполнение (отладчик): для более сложных случаев используйте встроенный отладчик Python (pdb) или отладочные инструменты в IDE (например, PyCharm, VS Code). Они позволяют выполнять код пошагово, просматривать значения переменных и устанавливать точки останова.

– Тестирование: Пишите небольшие тесты для отдельных функций, чтобы убедиться в их корректной работе

Критерии оценивания работ	
Работоспособность	программа запускается и выполняет заявленный функционал без ошибок.
Корректность кода	Python соблюден, нет логических ошибок.
Эффективность	код решает задачу оптимальным способом, без избыточных вычислений

Читаемость кода	код хорошо структурирован, используются осмысленные имена, есть комментарии, применен рефакторинг
Использование продвинутых конструкций	продемонстрировано умение применять функции, списки, строки, циклы с break/continue.?
Отладка	продемонстрировано умение находить и исправлять ошибки в коде

3. ТЕХНОЛОГИЧЕСКИЙ ПРАКТИКУМ: ВЕБ-РАЗРАБОТКА

В 9 классе веб-разработка становится ключевым практическим навыком. Вы научитесь создавать полноценные сайты, используя как конструкторы (Tilda), так и базовые языки веб-разметки и программирования (HTML, CSS, JavaScript). Это позволит вам создавать интерактивные и адаптивные веб-ресурсы.

Цели и задачи раздела:

– Tilda: освоение создания сайтов, подключения баз данных, публикации и использования Zero Block;

– HTML: углубленное изучение структуры веб-страницы, использования заголовков, абзацев, списков, ссылок, изображений, iframe-элементов;

– CSS: освоение стилизации элементов, работы с классами и ID, отступами (margin, padding), а также адаптивной верстки с помощью Flexbox;

– JavaScript: введение в интерактивность, работа с формами и кнопками, базовые скрипты для динамического поведения сайта.

– Хостинг и домены: понимание принципов размещения сайтов в интернете и выбора доменных имен.

– Практикум: создание сайта цветочного магазина, сайта-помощника для ОГЭ на Tilda, верстка многостраничного сайта и сайта по дизайн-макету на HTML/CSS/JS

Рекомендации по выполнению работ

– Структура и контент: Начните с планирования структуры сайта и подготовки контента (тексты, изображения). Tilda позволяет быстро собрать прототип.

– Zero Block: Освойте Zero Block для создания уникальных, нестандартных секций, которые невозможно сделать с помощью стандартных блоков. Это дает максимальную свободу дизайна.

– Подключение базы данных: Изучите, как Tilda интегрируется с внешними сервисами для сбора данных (например, формы обратной связи, заявки).

– Публикация: Поймите процесс публикации сайта, подключения собственного домена. Оптимизируйте сайт для поисковых систем.

– Используйте семантические теги (<header>, <nav>, <main>, <footer>, <section>, <article>) для создания понятной и доступной структуры.

– Формы и кнопки: Создавайте интерактивные формы (<form>, <input>, <button>, <textarea>, <select>) для сбора данных от пользователя.

– Iframe элементы: Используйте <iframe> для встраивания контента с других сайтов (например, карты, видео.

– Классы и ID: Поймите разницу между классами (.class-name) и ID (#id-name) и когда их использовать для стилизации.

– Отступы (margin и padding): Освойте управление внешними (margin) и внутренними (padding) отступами для правильного расположения элементов.

Критерии оценивания работ	
Работоспособность	сайт работает без ошибок, все интерактивные элементы функционируют.
Техническая грамотность	корректное использование HTML, CSS, JavaScript. Чистота кода.
Дизайн и адаптивность	сайт выглядит эстетично, адаптивен под разные устройства (мобильные, планшеты, десктопы
Структура и навигация	логичная структура сайта, удобная навигация
Публикация	сайт успешно опубликован на хостинге, доступен по доменному имени.
Использование Tilda	эффективное применение Zero Block, интеграция с Базой Данных

4. ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ: ПРОДВИНУТЫЕ ТЕХНИКИ

В 9 классе мы углубляем наше понимание искусственного интеллекта, фокусируясь на практическом применении Больших языковых моделей (LLM), промпт-инжиниринге, дообучении ИИ и его интеграции в процессы веб-разработки и рефакторинга кода. ИИ становится не просто инструментом, а полноценным ассистентом в создании сложных цифровых продуктов.

Цели и задачи раздела

– Большие языковые модели (LLM): понимание принципов работы, возможностей и ограничений современных LLM;

– Промпт-инжиниринг: освоение продвинутых техник составления эффективных промптов для LLM, включая Chain-of-Thought, Few-shot Learning, Persona-based prompting;

– Дообучение ИИ (Fine-Tuning): понимание концепции адаптации предобученных моделей под специфические задачи и данные;

– ИИ в веб-разработке: использование ИИ для генерации HTML/CSS/JS кода, создания макетов, написания текстов для сайта;

– Создание Веб-приложений с помощью ИИ: применение ИИ для генерации логики и компонентов веб-приложений.

– Рефакторинг кода с помощью ИИ: использование ИИ-помощников для анализа и улучшения качества Python-кода

Рекомендации по работе с ИИ

– Изучите, как работают LLM (например, DeepSeek или Qwen), их архитектуру (трансформеры) и основные возможности (генерация текста, переводы, ответы на вопросы.

– Просите ИИ рассуждать шаг за шагом, прежде чем дать окончательный ответ. Это улучшает качество сложных ответов.

– Предоставляйте ИИ несколько примеров желаемого результата, чтобы он лучше понял задачу.

– Указывайте ИИ, какую роль он должен играть (например, «Ты — опытный UX-дизайнер...».

– Четко формулируйте ограничения и желаемый формат вывода (JSON, Markdown, код).

Дообучение ИИ (Fine-Tuning)

– Концепция: Поймите, что такое дообучение и зачем оно нужно. Это процесс адаптации большой, универсальной модели под конкретную, узкую задачу с использованием небольшого набора специфических данных).

– Применение: изучите, как дообученные модели могут быть использованы для улучшения качества ответов в специфических областях (например, для чат-бота по истории школы).

ИИ в веб-разработке

– Генерация кода: Используйте ИИ для генерации фрагментов HTML, CSS, JavaScript по текстовому описанию. Это значительно ускоряет процесс разработки.

– Создание макетов: ИИ может предложить идеи для дизайна веб-страниц или даже сгенерировать готовые макеты на основе ваших требований.

– Написание текстов: Используйте ИИ для создания заголовков, описаний, статей для вашего сайта

Создание веб-приложений с ИИ

– Генерация логики: ИИ может помочь в написании серверной логики для веб-приложений (например, на Python с использованием Flask/Django).

Компоненты интерфейса: ИИ может генерировать готовые компоненты пользовательского интерфейса (кнопки, формы, навигационные панели).

Рефакторинг кода с помощью ИИ

– Анализ кода: ИИ-помощники могут анализировать ваш Python-код и предлагать улучшения для читаемости, производительности и устранения дублирования.

– Автоматический рефакторинг: Некоторые ИИ-инструменты могут автоматически применять рефакторинг, делая ваш код более чистым и эффективным).

Критерии оценки проектов с ИИ	
Эффективность промптинга	Насколько качественно и детально были составлены промпты для ИИ?
Интеграция ИИ	Насколько глубоко и осмысленно ИИ был интегрирован в проект (генерация кода, дизайна, текста, логики)?
Доработка и критический анализ	Насколько качественно вы доработали сгенерированный ИИ контент и критически оценили его результаты?
Понимание	Продемонстрировано ли понимание принципов работы LLM, промпт-инжиниринга и дообучения?
Этика	Соблюдены ли этические нормы при использовании ИИ, указано ли его применение?

5. ГРАФИЧЕСКИЙ ДИЗАЙН И 3D-МОДЕЛИРОВАНИЕ

В 9 классе графический дизайн и 3D-моделирование выходят на профессиональный уровень, фокусируясь на UI/UX, создании прототипов и использовании продвинутых инструментов, таких как Blender. Особое внимание уделяется интеграции ИИ в процесс дизайна.

Цели и задачи раздела

– UI/UX: углубленное изучение принципов User Interface (UI) и User Experience (UX) дизайна, элементов современного интерфейса.

– Adele Design: освоение работы с фреймами, элементами, создание вайрфреймов и прототипов.

– Адаптивный дизайн: разработка дизайна мобильной адаптации сайтов.

– Генерация дизайна с помощью ИИ: использование ИИ для создания макетов, цветовых схем, элементов интерфейса.

– Создание изображений с помощью ИИ: продвинутые техники генерации и редактирования изображений.

– 3D-моделирование в Blender: освоение базовых инструментов Blender для создания 3D-моделей.

– Практикум: дизайн сайта-магазина и сайта-лендинга в Adele Design, 3D-моделирование в Blender.

Рекомендации по графическому дизайну и 3D-моделированию

– Вайрфреймы (Wireframes): Создавайте схематичные наброски интерфейса, фокусируясь на расположении элементов и функциональности, без деталей дизайна.

– Прототипы: Превращайте вайрфреймы в интерактивные прототипы, которые имитируют работу реального приложения или сайта. Это позволяет тестировать пользовательский опыт.

Создание изображений с помощью ИИ

– Продвинутый промптинг: Применяйте техники промпт-инжиниринга для генерации высококачественных изображений, указывая стиль, детали, композицию.

– Редактирование с ИИ: Используйте ИИ для редактирования изображений (удаление объектов, изменение фона, стилизация).

3D-моделирование в Blender

– Интерфейс Blender: Освойте базовый интерфейс Blender: навигация, работа с объектами, режимы редактирования (Object Mode, Edit Mode).

– Моделирование: Начните с простых геометрических фигур. Изучите инструменты экструзии, фаски, подразделения поверхностей для создания более сложных форм.

– Материалы и текстуры: Применяйте материалы и текстуры для придания реалистичности вашим моделям.

Критерии оценки работ по графическому дизайну и 3D-моделированию	
Соответствие заданию	Насколько полно и точно выполнены требования к дизайну/модели?
Техническая реализация	Качество использования инструментов Adele Design/Blender, соблюдение технических стандартов
UI/UX	Удобство использования интерфейса, логичность навигации, качество пользовательского опыта.
Эстетика и креативность	Оригинальность идеи, гармония цвета, типографики, композиции. Качество 3D-модели.
Использование ИИ	Эффективность применения ИИ для генерации или доработки контента.

ЗАКЛЮЧЕНИЕ

1. **Постоянное обучение**: Мир IT меняется очень быстро. Регулярно изучайте новые технологии, языки программирования, инструменты и тренды в веб-разработке, ИИ и мультимедиа.
2. **Практика — это ключ к успеху:** Теория важна, но без постоянной практики в кодировании, дизайне, видеомонтаже и промптинге вы не сможете освоить материал. Создавайте собственные проекты**!**
3. **Дебаггинг ваш лучший друг:** Ошибки в коде или дизайне это нормально. Умение находить, анализировать и исправлять их (дебаггинг) — один из самых ценных навыков программиста и разработчика**.**
4. **Сначала логика, потом дизайн:** сначала добейтесь того, чтобы программа или таблица работала правильно, и только потом приступайте к украшению, подбору шрифтов и картинок.
5. **Чистота и структура кода**: Пишите аккуратный, читаемый Python код. Используйте комментарии, осмысленные имена переменных и функций. Это облегчит работу вам и другим разработчикам.
6. **Здоровье прежде всего:** Правильная осанка за компьютером — залог отсутствия усталости.
7. **Умение искать ответы:** если программа выдает ошибку или вы не знаете, как сделать действие, используйте поиск в интернете или встроенную справку. Умение находить решение — это главный навык айтишника.
8. **Эффективный промптинг:** научитесь четко и точно формулировать свои мысли для ИИ. Это навык будущего, который позволит вам максимально использовать потенциал искусственного интеллекта.
9. **Этика общения**: помните о правилах сетевого этикета (нетикета) при переписке в почте или работе в общих документах. Уважайте чужой труд и личное пространство.
10. **Оптимизация ресурсов**: при создании веб-сайтов и видеороликов думайте об оптимизации изображений, видео и кода для быстрой загрузки и эффективной работы.