ИНДИВИДУАЛЬНЫЙ ПРЕДПРИНИМАТЕЛЬ
ГРУШИН ЕВГЕНИЙ АНАТОЛЬЕВИЧ

Утверждена
приказом № 9 от «10» сентября 2024 г.
Индивидуальный предприниматель
_____ / Грушин Евгений Анатольевич

Дополнительная общеобразовательная программа -

дополнительная общеразвивающая программа

«ОСНОВЫ ПРОГРАММИРОВАНИЯ»

Срок обучения – 144 ч.

Форма обучения: заочная, с применением исключительного электронного обучения и дистанционных образовательных технологий

Адресат: дети 11-16 лет

Екатеринбург, 2024

СОДЕРЖАНИЕ ПРОГРАММЫ

1. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Дополнительная общеобразовательная программа - дополнительная общеразвивающая программа «Основы программирования» (далее - программа) направлена на удовлетворение индивидуальных потребностей обучающихся в интеллектуальном развитии, создание и обеспечение необходимых условий для личностного развития, организацию свободного времени обучающихся, удовлетворение иных образовательных потребностей и интересов обучающихся, не противоречащих законодательству Российской Федерации, осуществляемых за пределами федеральных государственных образовательных стандартов и федеральных государственных требований.

Программа по виду образования – дополнительное образование, подвид – дополнительное образование детей и взрослых, направленность программы – техническая.

Дополнительная общеразвивающая программа разработана на основе следующих нормативных правовых документов:

- Федерального закона от 29.12.2012 № 273-ФЗ «Об образовании в Российской Федерации»;
- Постановления Правительства РФ от 11.10.2023 № 1678 «Об утверждении Правил применения организациями, осуществляющими образовательную деятельность, электронного обучения, дистанционных образовательных технологий при реализации образовательных программ»;
- Приказа Министерства просвещения РФ от 27.07.2022 № 629 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам».

1.1. Актуальность дополнительной общеобразовательной программы

В современном мире, где технологии развиваются с неимоверной скоростью, знание основ программирования становится не просто преимуществом, а необходимостью для подростков. Онлайн-курс «Основы программирования» предоставляет уникальную возможность юным обучающимся познакомиться с базовыми концепциями и языками программирования, такими как Python, JavaScript и другие. Этот курс актуален не только для тех, кто стремится связать свою будущую профессию с IT-сферой, но и для широкого круга интересующихся, желающих развивать логическое мышление и творческий подход к решению задач.

Во-первых, программирование стимулирует аналитические способности, учит структурировать информацию и искать оптимальные решения. Во-вторых, владение программированием открывает двери к новым карьерным возможностям в будущем, так как многие профессии сегодня требуют базовых знаний в этой области. Наконец, программа курса включает практические задания, что способствует закреплению теоретических знаний.

Таким образом, онлайн-курс «Основы программирования» становится важным шагом в подготовке обучающихся к вызовам современного мира, способствуя их всестороннему развитию и уверенности в завтрашнем дне.

Новизна дополнительной общеобразовательной программы заключается в том, что содержание программы адаптировано под дистанционный формат обучения. Обучающиеся могут проходить обучение в удобном темпе и графике, так как по каждой теме будут размещаться записи проведенных лекций.

Педагогическая целесообразность обучения заключается в том, что структура занятий построена таким образом, что теоретические знания чередуются с практикой, что является наиболее продуктивным и целесообразным. Также при реализации программы имеются обязательные задания, без выполнения которых, нельзя перейти к следующей теме обучения. Это позволяет мотивировать обучающихся для полноценного и качественного освоения программы.

1.2. Адресат дополнительной общеобразовательной программы: дети 11-16 лет.

Специальные способности: не требуются.

Необходимые знания и умения: базовые навыки работы с персональным компьютером (умение запускать и отключать компьютер; понимание назначения периферийных устройств (компьютерная мышь, клавиатура), умение работать со стандартными компьютерными программами); умение пользоваться интернет-браузером; навык работы с электронной почтой.

1.3. Цель и задачи программы

Цель программы – познакомить обучающихся с базовыми принципами программирования, такими как алгоритмы, переменные, типы данных, востребованные языки и особенности их использования. Курс также помогает определить интересы и цели обучающегося, даёт необходимый минимум знаний для дальнейшего обучения и развития навыков в области программирования.

Задачи программы:

Обучающие:

- знакомство с понятиями «алгоритмы», «переменные», «типы данных»;
- изучение востребованных языков программирования и их особенностей;
- освоение написания кода и вёрстки сайтов;
- решение алгоритмических задач;
- создание простых текстовых игр и программ.

Развивающие:

- повысить уровень концентрации и внимания обучающихся;
- развить мотивационные качества обучающихся;

- развить интеллектуальные качества обучающихся, познавательный интерес;
- развить эмоциональные качества и чувства обучающихся;
- развить волевые качества обучающихся, самостоятельность, умение преодолевать трудности в учении используя для этого проблемные ситуации, творческие задания, поощрение;
- развить творческие способности обучающихся, их познавательную активность.

Воспитательные задачи:

- способствовать саморазвитию, профориентации;
- воспитать у обучающихся самостоятельность и ответственность за свое обучение.

1.4. Форма обучения – заочная, с применением исключительно электронного обучения и дистанционных образовательных технологий.

1.5. Срок обучения – 144 час.

1.6. Продолжительность реализации дополнительной общеобразовательной программы составляет – 9 месяцев.

Занятия проводятся согласно учебному расписанию, продолжительность 1 занятия – 40 минут, перерывы между занятиями не менее 10 минут.

1.7. Планируемые образовательные результаты

Знать:

- основы работы с программой Google Sketch Up;
- создание объектов и моделирование в программе Google Sketch Up;
- как создавать анимации и симуляции в программе Google Sketch Up;
- интерфейс программы Blender;
- как работать с позицией, вращением и масштабом в программе Blender;
- единицы измерения программы Blender;
- топологию и ретопологию в Blender;
- основные инструменты текстурирования в Blender;
- как работать с анимацией и ключевыми кадрами в Blender;
- инструменты скульптинга в Blender;
- как создавать собственные иконки и иллюстрации в Figma;
- функции Auto Layout и Variants
- основы языка гипертекстовой разметки (HTML) и каскадных таблиц стилей (CSS);
- особенности стилизации веб-страниц;
- основы работы с программой Adobe Photoshop и работа с дизайн-макетами веб-сайтов;
- базовые конструкции и типы данных в Python;
- условные операторы и циклы Python;
- функции и рекурсию, структуры данных Python;

- объектно-ориентированное программирование Python;
- язык HTML, структуру программы, основные теги языка программирования JavaScript;
- операторы языка программирования JavaScript;
- основы программирования на Java;
- объектно-ориентированное программирование на Java;
- основы PHP, основы SQL;
- основы синтаксиса и базовые конструкции в C++;
- условные операции, циклы и массивы в C++;
- объектно-ориентированное программирование (ООП) в C++;
- цели и задачи языка программирования C#;
- переменные и типы данных, математические операции языка программирования C#;
- условные выражения и конструкции языка программирования C#.

Уметь:

- работать с материалами и текстурами в программе Google Sketch Up;
- редактировать модели и осуществлять сборку сцены в программе Google Sketch Up;
- осуществлять экспорт и импорт файлов в программе Google Sketch Up;
- работать с группами объектов и компонентами в программе Google Sketch Up;
- настраивать Blender;
- привязывать и осуществлять точное моделирование в Blender;
- осуществлять пропорциональное редактирование, моделировать объекты в Blender;
- создавать анимацию в Blender, осуществлять скульптинг;
- работа с компонентами и интерактивными прототипами в Figma;
- применять технику создания web-страниц;
- публиковать свой ресурс в сети, чтобы создать собственный проект в виде полноценного многостраничного сайта в Интернете;
- писать программу на Python;
- осуществлять программирование на базе языка программирования JavaScript;
- писать собственное Java-приложение;
- создавать интернет-порталы и социальные сети, защищать сайты от взлома и незаконного доступа к PHP-данным;
- создавать прикладные программы на языке программирования C++;
- осуществлять программирование на базе языка программирования C#.

1.8. Документ об обучении: лицу, освоившему образовательную программу, выдается документ об обучении по образцу, установленному

индивидуальным предпринимателем самостоятельно, – сертификат.

2. УЧЕБНЫЙ ПЛАН

№ п/п	Наименование модулей	Количество часов					Форма промежуточной аттестации
		Всего	Теоретическое занятие	Практическое занятие	Промежуточная аттестация	Итоговая аттестация	
1	Google Sketch Up	14	2,5	11,5			
2	Blender	14	7	7			
3	Figma	14	2	12			
4	Frontend-разработчик HTML+CSS	14	2	12			
5	Язык программирования Python	14	4	10			
6	Промежуточная аттестация	2			2		Зачет
7	Язык программирования JavaScript	14	5	9			
8	Язык программирования Java	14	4	10			
9	PHP+SQL	14	6	8			
10	Язык программирования C++	14	3	11			
11	Язык программирования C#	14	4,5	9,5			
12	Итоговая аттестация	2				2	Зачет
	Итого	144	40	100	2	2	

3. КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

№ пп	Наименование модулей	Количество часов	Период обучения
1	Google Sketch Up	14	Сентябрь
2	Blender	14	Сентябрь-октябрь
3	Figma	14	Октябрь-ноябрь
4	Frontend-разработчик HTML+CSS	14	Ноябрь-декабрь
5	Язык программирования Python	14	Декабрь-январь
6	Промежуточная аттестация	2	Январь
7	Язык программирования JavaScript	14	Январь-февраль
8	Язык программирования Java	14	Февраль-март
9	PHP+SQL	14	Март-апрель
10	Язык программирования C++	14	Апрель
11	Язык программирования C#	14	Май
12	Итоговая аттестация	2	Май
	Итого	144	9 месяцев обучения

Режим занятий: занятия проводятся по 2 часа 2 раза в неделю.

4. СОДЕРЖАНИЕ РАБОЧЕЙ ПРОГРАММЫ

МОДУЛЬ 1. GOOGLE SKETCH UP

Google Sketch Up - программа для дизайна и архитектурного проектирования, которая подходит для изучения основ профессионального 3D-моделирования. Обучающийся научится работать с графическими объектами, приобретет навыки пространственного мышления и визуализации, что является неотъемлемой частью создания трехмерной графики.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие
1	Основы работы с программой	2	2	
2	Создание объектов и моделирование	2		2
3	Работа с материалами и текстурами	2	0,5	1,5
4	Освещение и тени	1		1
5	Редактирование моделей и сборка сцены	1		1
6	Экспорт и импорт файлов	1		1
7	Продвинутое моделирование и оптимизация	1		1
8	Создание анимации и симуляции	1		1
9	Работа с группами объектов и компонентами	1		1
10	Визуализация и презентация проектов	2		2
	Итого	14	2,5	11,5

Тема 1. Основы работы с программой

Теоретическое занятие

Введение в SketchUp: программа, концепция и геометрия. Слои и организация объектов в SketchUp. Группы и компоненты: создание и использование. Инструменты просмотра и панорамирования в SketchUp. Вставка и работа с внешними файлами. Редактирование и модификация объектов в SketchUp. Визуализация и сохранение проектов в SketchUp.

Тема 2. Создание объектов и моделирование

Практическое занятие

Задание:

1. Откройте новый файл в Google SketchUp.
2. Используйте инструмент «Прямоугольник», чтобы создать основание скворечника.
3. Добавьте две вертикальные стенки с помощью инструмента «Тяни/толкай».
4. Создайте крышу с использованием инструмента «Крыша».
5. Добавьте декоративные элементы, такие как окна и дверь, используя инструменты «Вырезать» и «Тяни/толкай».

6. Отредактируйте размеры и форму деталей скворечника при необходимости.

7. Сохраните файл под именем «Скворечник».

Тема 3. Работа с материалами и текстурами

Теоретическое занятие

Материалы и текстуры: использование и настройка.

Практическое занятие

Задание:

1. Создайте новую модель здания с помощью инструмента «Прямоугольник».

2. Добавьте детали, такие как окна, двери и крыша, используя инструменты «Тяни/толкай» и «Вырезать».

3. Разделите модель на слои для удобства работы с разными материалами и текстурами.

4. Выберите материалы для стен, крыши и окон, используя меню «Материалы».

5. Настройте параметры материалов, такие как цвет, текстура и прозрачность.

6. Добавьте текстуры к стенам, крыше и окнам, используя инструмент «Текстура».

7. Экспериментируйте с различными комбинациями материалов и текстур для достижения желаемого визуального эффекта.

8. Сохраните готовую модель с использованием выбранного вами формата файла.

Тема 4. Освещение и тени

Практическое занятие

Задание:

1. Создайте новую модель комнаты с использованием инструментов SketchUp.

2. Разместите источник света (например, солнце или лампу) в окне или другом подходящем месте.

3. Используйте инструменты настройки света (например, интенсивность, направление и тип света) для создания реалистичного освещения.

4. Добавьте тени от предметов и объектов в комнате, используя инструменты SketchUp для создания теней.

5. Экспериментируйте с настройками света и теней, чтобы достичь желаемого визуального эффекта.

6. Сохраните готовую модель с освещением и тенями для дальнейшего использования или демонстрации.

Тема 5. Редактирование моделей и сборка сцены
Практическое занятие
Задание:

1. Создайте базовую модель города, используя инструменты SketchUp для создания основных зданий, дорог, тротуаров и растительности.

2. Проработайте детали зданий, добавляя окна, двери, лестницы и другие элементы.

3. Создайте рельеф местности, используя инструменты SketchUp для создания холмов, гор и водоёмов.

4. Добавьте деревья, кусты и другую растительность, используя инструменты SketchUp для создания листьев, веток и стволов.

5. Соберите сцену, размещая созданные объекты на фоне друг друга и настраивая их положение и ориентацию.

6. Отредактируйте модель, исправляя ошибки, улучшая пропорции зданий и ландшафта, а также добавляя дополнительные детали и элементы.

7. Сохраните готовую модель в формате файла SketchUp (.skp) для дальнейшей работы или демонстрации.

Тема 6. Экспорт и импорт файлов
Практическое занятие
Задание 1: экспорт файла.

- Создайте модель дома в Google SketchUp.
- Сохраните файл в формате *.skp.
- Загрузите файл на облачное хранилище, например, Google Drive или Dropbox.

Задание 2: импорт файла.

- Откройте файл, загруженный на облачное хранилище.
- Импортируйте файл обратно в Google SketchUp.
- Проверьте, что модель успешно импортирована и не содержит ошибок.

Задание 3: экспорт и импорт нескольких файлов.

- Создайте ещё одну модель дома в Google SketchUp.
- Сохраните обе модели в форматах *.skp.
- Загрузите оба файла на облачное хранилище.
- Импортируйте оба файла обратно в Google SketchUp.
- Проверьте, что модели успешно импортированы и не содержат ошибок.

Тема 7. Продвинутое моделирование и оптимизация
Практическое занятие
Задание:

1. Создайте базовую 3D-модель здания, используя инструменты «Прямоугольник», «Тяни/толкай» и «Ластик».

2. Добавьте детали, такие как окна, двери, крыша и декоративные элементы, используя инструменты продвинутого моделирования.

3. Оптимизируйте модель, удаляя лишние линии и упрощая геометрию, чтобы улучшить производительность и уменьшить размер файла.

4. Настройте материалы и текстуры для каждого элемента модели, используя меню «Материалы» и инструмент «Текстура».

Тема 8. Создание анимации и симуляции

Практическое занятие

Задание:

1. Создайте модель объекта, например, автомобиля, используя инструменты SketchUp.

2. Разместите объект на сцене, выбрав подходящее место и ориентацию.

3. Создайте анимацию движения объекта, используя инструмент «Анимация».

4. Определите начальное и конечное положение объекта, а также время перехода между ними.

5. Настройте параметры анимации, такие как скорость, ускорение и длительность.

6. Запустите анимацию, чтобы увидеть результат.

7. При необходимости внесите корректировки в параметры анимации и повторите шаги 4–6 для достижения желаемого результата.

8. Сохраните готовую анимацию в формате файла SketchUp (.skp) для дальнейшего использования или демонстрации.

Тема 9. Работа с группами объектов и компонентами

Практическое занятие

Задание:

1. Создайте модель комнаты с несколькими объектами: стены, пол, потолок, окно и дверь.

2. Объедините эти объекты в группы, используя инструмент «Выбрать» и «Группировать».

3. Создайте новый компонент, выбрав группу объектов и нажав правой кнопкой мыши «Создать компонент».

Назовите компонент, например, «Комната_1».

4. Сохраните компонент в папке «Мои компоненты» или создайте новую папку для хранения компонентов.

5. Откройте модель другой комнаты и добавьте компонент «Комната_1» с помощью инструмента «Перетащить и отпустить».

Убедитесь, что все объекты в компоненте корректно перенесены и связаны с новой моделью.

6. При необходимости внесите корректировки в параметры компонентов и повторите шаги 4–7 для создания дополнительных компонентов.

Тема 10. Визуализация и презентация проектов

Практическое занятие

Задание:

1. Создайте модель дома в Google SketchUp, используя инструменты прямоугольника, линии, смещения и тяни-толкай.

2. Добавьте детали, такие как окна, двери, крыша и декоративные элементы, используя инструменты продвинутого моделирования.

3. Настройте материалы и текстуры для каждого элемента модели, используя меню «Материалы» и инструмент «Текстура».

4. Воспользуйтесь инструментами визуализации, такими как рендеринг, освещение и камера, чтобы создать качественные изображения и видео вашего проекта.

5. Подготовьте презентацию, включая 3D-модель дома, визуализированные изображения и краткое описание проекта.

МОДУЛЬ 2. BLENDER

Одна из самых востребованных программ в IT-сфере по созданию профессиональной трехмерной графики, 3D-анимации и архитектурных проектов. Обучающиеся научатся не только моделированию, но иск ульптингу, текстурированию и композиции для создания индивидуального проекта, что необходимо в инженерии, архитектуре и многих других областях.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие
1	Знакомство с Blender	2	2	
2	Основы Blender и CG	2	0,5	1,5
3	Моделирование	1	1	
4	Освещение	2	0,5	1,5
5	Текстурирование	2	0,5	1,5
6	Запекание	0,5	0,5	
7	Анимация	2	0,5	1,5
8	Импорт/экспорт ассетов	0,5	0,5	
9	Скульптинг в Blender	1,5	0,5	1
10	Motion Tracking	0,5	0,5	
	Итого	14	7	7

Тема 1. Знакомство с Blender

Теоретическое занятие

Где скачать Blender. Интерфейс программы. Навигация в окне 3D-вида. Настройки Blender.

Тема 2. Основы Blender и CG

Теоретическое занятие

Позиция, вращение и масштаб. Объекты и данные. Outliner. Сложность вычислений.

Практическое занятие

Задание:

Описание задания

Вам нужно будет навести порядок в проекте и разобраться с интерфейсом программы. В доп. материалах к курсу, вы найдёте проект, который вам нужно будет реорганизовать. После внесения исправлений, сохраните проект и залейте его в облако (Яндекс.Диск, Google Drive, Dropbox и т. д.), а ссылку пришлите нам.

Перед отправкой ссылки на проверку убедитесь:

1. открыт ли доступ к файлам в облаке;
2. что в сцене запакованы текстуры. Чтобы запаковать текстуры, активируем чекбокс File → External Data → Automatically Pack Resources;
3. не архивируйте файлы, как правило это не сильно уменьшает размер, просто закиньте все в одну папку; контролируйте размер файлов, желательно уместиться до 400 МБ.

Тема 3. Моделирование

Теоретическое занятие

Структура мешей. Единицы измерения. Трансформации и ориентации. Способы вращения. Привязка и точное моделирование. Пропорциональное редактирование. Генерирующие модификаторы. Деформирующие модификаторы. Топология. Ретопология. Моделирование светофора. Моделирование гитары. Моделирование дрона.

Тема 4. Освещение

Теоретическое занятие

Свет — это всё. Виды источников света. Способы освещения.

Практическое занятие

Задание:

Воспроизвести три заданных сценария освещения на примере своей сцены.

Тема 5. Текстурирование

Теоретическое занятие

Основные термины. Создание и назначение материалов. Рендеры Cycles и EEVEE. PBR-шейдинг. UV-развертка.

Практическое занятие

Задание:

Создать атмосферную сцену с заправочной станции.

Тема 6. Запекание

Теоретическое занятие

Для чего используется запекание? Запекание текстур.

Тема 7. Анимация

Теоретическое занятие

Анимация и ключевые кадры. Редактор графов. Ключевые формы. Видеоредатор Blender (VSE). Анимирование диорамы.

Практическое занятие

Задание:

Создать диораму и добавить в неё анимированные элементы.

Тема 8. Импорт/экспорт ассетов

Теоретическое занятие

Браузер ассетов Blender. Импорт/экспорт ассетов.

Тема 9. Скульптинг в Blender

Теоретическое занятие

Разбор инструментов скульптинга. Ретопология объекта под скульптинг.

Практическое занятие

Задание:

- Создание деревянной доски.
- Создание пенька.
- Создание стилизованного топора.

Тема 10. Motion Tracking

Теоретическое занятие

Знакомство с Motion Tracking. Трекеры и редактор Movie Clip. Маски и постобработка.

МОДУЛЬ 3. FIGMA

Графический редактор, который используют лучшие web-дизайнеры мира. С помощью Figma обучающиеся научатся создавать макеты сайтов,

разрабатывать интерфейсы для мобильных приложений и воспользуются полученными навыками в создании собственного интернет-проекта.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие
1	Введение в Figma	2	2	
2	Создание собственных иконок и иллюстраций	2		2
3	Работа с компонентами и интерактивными прототипами	2		2
4	Библиотека стилей и шаблонов	2		2
5	Функции Auto Layout и Variants	2		2
6	Плагины	2		2
7	Итоговый проект	2		2
	Итого	14	2	12

Тема 1. Введение в Figma

Теоретическое занятие

Знакомство с главным меню Figma: регистрация, скачивание приложения, работа с браузером, раздел Community. Порядок создания макета. Использование горячих клавиш для быстрой работы с примитивами и слоями. Применение модульных сеток для выравнивания элементов на макете. Загрузка изображений и текста на слои. Настройка параметров текста. Изменение цвета заливки и прозрачности слоёв. Создание и настройка градиентов и эффектов.

Тема 2. Создание собственных иконок и иллюстраций

Практическое занятие

Задание:

1. Выберите инструмент Vector на панели инструментов.
2. Выберите инструмент Rectangle, Ellipse или Polygon для рисования основы иконки.
3. Добавьте другие фигуры, измените их размеры и цвета.
4. Используйте функцию Boolean Operations для комбинирования и пересечения фигур.
5. Добавьте текст и другие элементы дизайна при необходимости.
6. Экспортируйте иконку в формате PNG или SVG**.**

Тема 3. Работа с компонентами и интерактивными прототипами

Практическое занятие

Задание:

1. Создайте новый проект в Figma и назовите его «Интернет-магазин одежды».

2. Добавьте фреймы для разных экранов сайта: главная страница, страница товара, корзина, страница регистрации и страница авторизации.

3. Используя инструменты Shape и Text, добавьте элементы дизайна: кнопки, текстовые поля, иконки и изображения товаров.

4. Организуйте элементы внутри фреймов, используя панель слоёв.

5. Создайте компоненты для часто используемых элементов, таких как кнопки, иконки и текстовые блоки.

6. Добавьте интерактивные переходы между экранами, настроив стрелки и триггеры.

7. Настройте анимации для элементов, выбрав тип анимации и настроив параметры.

8. Протестируйте прототип, пройдя через все интерактивные элементы и убедившись в корректной работе.

Тема 4. Библиотека стилей и шаблонов

Практическое занятие

Задание:

1. Создайте новый проект в Figma и назовите его «Мобильное приложение».

2. Разработайте базовый дизайн интерфейса приложения, включая экраны приветствия, списка задач, редактирования задачи и списка контактов.

3. Создайте слои для каждого элемента интерфейса, такие как фон, кнопки, текстовые поля и иконки.

4. Определите основные стили и шаблоны для каждого элемента, учитывая цвета, шрифты и графические элементы.

5. Создайте библиотеку стилей и шаблонов, группируя элементы по категориям (например, цвета, шрифты, иконки).

6. Экспортируйте библиотеку стилей и шаблонов в формате JSON или SCSS для дальнейшего использования в других проектах.

7. Протестируйте работу библиотеки стилей и шаблонов, создав новый проект и используя готовые стили и шаблоны для быстрого создания макетов интерфейса приложения.

Тема 5. Функции Auto Layout и Variants

Практическое занятие

Задание:

1. Создайте новый проект в Figma и назовите его «Адаптивное меню».

2. Создайте фрейм для меню, используя инструмент Auto Layout. Разместите элементы меню (например, ссылки на страницы) равномерно внутри фрейма.

3. Установите разные варианты отображения меню для разных устройств (например, смартфон, планшет, компьютер). Для этого используйте функцию Variants.

4. Настройте параметры Auto Layout и Variants, чтобы меню автоматически адаптировалось к различным размерам экрана и ориентациям устройств.

5. Протестируйте работу меню на разных устройствах и убедитесь, что оно корректно отображается и функционирует.

Тема 6. Плагины

Практическое занятие

Задание:

1. Откройте проект в Figma и выберите вкладку «Plugins».

2. Найдите и установите плагин Unsplash.

3. Перейдите на сайт Unsplash и найдите подходящие изображения для вашей формы.

4. Перетащите изображения из Unsplash на форму в вашем проекте Figma.

5. Настройте размеры и положение изображений в соответствии с вашим дизайном.

6. Протестируйте работу формы с загруженными изображениями, убедившись, что они отображаются корректно.

Тема 7. Итоговый проект

Практическое занятие

Задание:

Практическое задание для подростков по теме «Итоговый проект в Figma»: создание дизайн-макета сайта для портфолио художника.

1. Создание фрейма и сетки для макета.

2. Работа со слоями и масками.

3. Композиция и декомпозиция элементов дизайна.

4. Использование типографики и теории цвета.

5. Адаптация дизайн-макета под разные экраны и устройства.

6. Экспорт файла из Figma и подготовка к публикации на сайте.

МОДУЛЬ 4. FRONTEND-РАЗРАБОТЧИК HTML+CSS

В рейтинге IT- профессий, web-разработка - одно из самых востребованных и высокооплачиваемых направлений. В модуле HTML+CSS обучающиеся изучат основы языка гипертекстовой разметки, освоят технику создания web-страниц, научатся публиковать свой ресурс в сети, чтобы создать собственный проект в виде полноценного многостраничного сайта в Интернете.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие

1	Основы языка гипертекстовой разметки (HTML) и каскадных таблиц стилей (CSS)	2	2	
2	Вёрстка современных веб-сайтов	4		4
3	Особенности стилизации веб-страниц	4		4
4	Основы работы с программой Adobe Photoshop и работа с дизайн-макетами веб-сайтов	4		4
	Итого	14	2	12

Тема 1. Основы языка гипертекстовой разметки (HTML) и каскадных таблиц стилей (CSS)

Теоретическое занятие

Основные понятия и структура HTML-документа. Элементы и атрибуты HTML. Основы CSS: селекторы, свойства и значения. Применение CSS для оформления веб-страниц. Примеры использования HTML и CSS для создания простых веб-страниц.

Тема 2. Вёрстка современных веб-сайтов

Практическое занятие

Задание:

1. Создайте личную папку для сохранения файлов сайта.
2. Откройте программу Блокнот и наберите простейший файл HTML с расписанием занятий на вторник.
3. Сохраните файл под именем RASP.HTML в личной папке.
4. Откройте файл в браузере двойным кликом по значку файла.
5. Разместите файл графического изображения CLOCK.JPG во вложенной папке и внесите изменения в файл HTML.
6. Сохраните файл в личной рабочей папке под именем 5.HTML.
7. Загрузите браузер и просмотрите созданную веб-страницу.

Тема 3. Особенности стилизации веб-страниц

Практическое занятие

Задание: Создайте HTML-файл с базовым текстом и разметкой.

1. Создайте CSS-файл с правилами стиля для текста и элементов страницы.
2. Подключите CSS-файл к HTML-файлу с помощью тега <link>.
3. Измените стиль текста и элементов страницы, используя свойства CSS.
4. Добавьте дополнительные элементы на страницу и стилизуйте их.
5. Протестируйте работу вашего проекта в разных браузерах и на разных устройствах.
6. Внесите необходимые корректировки и улучшения в ваш код.

Тема 4. Основы работы с программой Adobe Photoshop и работа с дизайн-макетами веб-сайтов

Практическое занятие

Задание:

1. Откройте Adobe Photoshop и создайте новый документ.
2. Разработайте дизайн главной страницы веб-сайта, включая логотип, меню, информационные блоки и контактную информацию.
3. Сохраните файл с расширением PSD.
4. Откройте файл в текстовом редакторе, таком как Notepad или Sublime Text, и преобразуйте его в HTML-код.
5. Создайте папку для хранения файлов веб-сайта и загрузите преобразованный HTML-код в эту папку.
6. Откройте файл index.html в браузере, чтобы увидеть результат вашей работы.
7. Протестируйте сайт на разных устройствах и браузерах, чтобы убедиться в корректном отображении дизайна.
8. Внесите необходимые корректировки и улучшения в дизайн и код сайта, если это необходимо.

МОДУЛЬ 5. ЯЗЫК ПРОГРАММИРОВАНИЯ PYTHON

Язык программирования, который широко используется российскими и международными IT-корпорациями. Очередь на Python-разработчика составляет 2-3 месяца среди работодателей, поэтому мы учим обучающихся не только тому, что интересно, но и тому, что актуально.

На занятиях обучающиеся освоят синтаксис языка, напишут свою первую программу, научатся подключать библиотеки и создадут собственный проект.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие
1	Введение в программирование и Python	1	1	
2	Базовые конструкции и типы данных	1,25	0,25	1
3	Условные операторы и циклы	1,25	0,25	1
4	Функции и рекурсия	1,25	0,25	1
5	Структуры данных	1,25	0,25	1
6	Файлы и работа с ними	1,5	0,5	1
7	Объектно-ориентированное программирование	1,5	0,5	1
8	Работа с библиотеками и модулями	1,5	0,5	1
9	Тестирование и отладка кода	1,5	0,5	1
10	Итоговый проект	2		2
	Итого	14	4	10

Тема 1. Введение в программирование и Python
Теоретическое занятие
Установка Python и среды разработки. Структура программы на Python.

Тема 2. Базовые конструкции и типы данных
Теоретическое занятие
Переменные и константы. Арифметические операции. Логические операции. Строковые и числовые типы данных.

Практическое занятие
Задание:
1. Напишите программу на Python, которая приветствует пользователя фразой «Привет, мир!» и выводит её на экран.
2. Создайте программу на Python, которая запрашивает у пользователя два числа и выполняет с ними арифметические операции: сложение, вычитание, умножение и деление.

Тема 3. Условные операторы и циклы
Теоретическое занятие
Условный оператор if. Множественные условия и вложенные операторы if. Цикл while. Цикл for.

Практическое занятие
Задание:
1. Создайте программу на Python, которая запрашивает у пользователя число и определяет, является ли оно положительным, отрицательным или нулём, используя условные операторы.

```
num = int(input("Введите число: "))
if num > 0:
 print("Число положительное")
elif num < 0:
 print("Число отрицательное")
else:
 print("Число равно нулю")
```

2. Напишите программу на Python, которая выводит все числа от 1 до 10, используя цикл for.

```
for i in range(1, 11):
 print(i)
```

Тема 4. Функции и рекурсия

Теоретическое занятие

Определение и вызов функций. Передача аргументов функциям. Рекурсия и её применение.

Практическое занятие

Задание:

1. Определите функцию, которая принимает целое число n и возвращает сумму первых n натуральных чисел. Используйте рекурсию для вычисления этой суммы.

```
def sum_naturals(n):
 if n == 1:
 return 1
 else:
 return n + sum_naturals(n - 1)

result = sum_naturals(5)
print(result) # Вывод: 15
```

2. Напишите функцию, которая принимает строку s и возвращает новую строку, где каждый символ заменяется на его ASCII-код. Используйте рекурсию для выполнения этой задачи.

Тема 5. Структуры данных

Теоретическое занятие

Списки. Кортежи. Множества. Словари.

Практическое занятие

Задание:

1. Создайте список чисел:

```
numbers = [1, 2, 3, 4, 5]
```

2. Добавьте элемент в начало списка:

```
numbers.insert(0, 0)
```

3. Добавьте элемент в конец списка:

```
numbers.append(6)
```

4

. Измените значение элемента с индексом 2:

```
numbers[2] = 8
```

5

. Удалите элемент с индексом 3:

```
del numbers[3]
```

6

. Переверните список:

numbers.reverse()

7. Вычислите сумму всех элементов списка:
sum_of_numbers = sum(numbers)

8. Преобразуйте список в строку, разделяя элементы символом «,»:
string_representation = ','.join(map(str, numbers))

9. Проверьте, есть ли в списке число 3:

```
if 3 in numbers:
 print("Число 3 присутствует в списке")
else:
 print("Число 3 отсутствует в списке")
```

10. Создайте новый список, состоящий из квадратов элементов исходного списка:

```
squared_numbers = map(lambda x: x**2, numbers)
new_list = list(squared_numbers)
```

Тема 6. Файлы и работа с ними

Теоретическое занятие

Открытие и закрытие файлов. Чтение и запись данных в файлы. Обработка текстовых файлов.

Практическое занятие

Задание:

1. Откройте файл «input.txt» и прочитайте первые 10 символов.

```
with open('input.txt', 'r', encoding='utf-8') as f:
 first_10_chars = f.read(10)
print(first_10_chars)
```

2. Запишите строку «Привет, мир!» в файл «output.txt».

```
with open('output.txt', 'w', encoding='utf-8') as f:
 f.write('Привет, мир!')
```

3. Создайте новый файл «new_file.txt» и запишите в него список продуктов, разделённых запятыми.

```
products = ['Морковь', 'Яблоки', 'Мука', 'Молоко']
with open('new_file.txt', 'a', encoding='utf-8') as f:
 f.writelines(products)
```

4. Откройте файл «new_file.txt» и прочитайте все строки.

```
with open('new_file.txt', 'r', encoding='utf-8') as f:
 products = f.readlines()
print(products)
```

Тема 7. Объектно-ориентированное программирование

Теоретическое занятие

Классы и объекты. Методы и атрибуты объектов. Наследование и полиморфизм.

Практическое занятие

Задание:

1. Создайте класс Car с атрибутами марка, модель и год выпуска.
2. Создайте метод display_info(), который выводит информацию об автомобиле (марка, модель, год выпуска).
3. Создайте объект my_car класса Car и вызовите метод display_info().
4. Наследуйте от класса Car новый класс ElectricCar, добавив атрибут тип_двигателя и метод заряда().
5. Создайте объект my_electric_car класса ElectricCar и вызовите метод заряда().
6. Реализуйте полиморфизм, создав метод speak() для каждого класса Car и ElectricCar. Вызовите этот метод для объектов my_car и my_electric_car.

Тема 8. Работа с библиотеками и модулями

Теоретическое занятие

Установка и импорт модулей. Использование стандартных библиотек. Написание собственных модулей.

Практическое занятие

Задание: разработать набор базовых функций для работы с книжной библиотекой и оформить этот набор в виде модуля.

Данные для работы: библиотека представляет собой папку с текстовыми файлами, расширение имени файла с книгой —.txt. Файлы представлены в особом формате: первая строка файла — имя автора, вторая строка — название книги, третья строка — аннотация. Начиная с четвёртой строки и до конца файла идёт текст книги. Кроме того, в каталоге может лежать специальный файл description.txt, содержащий описание библиотеки.

Интерфейс модуля: функции для работы с библиотекой должны включать:

read_library_description(library_dir) — функция находит файл description.txt в папке с библиотекой и возвращает его содержимое.

get_book_files(library_dir) — функция возвращает список имён файлов книг в указанной папке, игнорируя файл description.txt.

read_book_info(library_dir, book_file) — функция возвращает кортеж из трёх значений: автор, название, аннотация для указанного файла книги.

get_authors(library_dir) — функция возвращает список имён авторов, книги которых есть в библиотеке.

get_all_books_info(library_dir) — функция возвращает список книг в данной папке, каждая книга представлена в виде кортежа (автор, название, аннотация, имя файла).

Все функции должны быть оформлены в виде модуля library_dir.py. К каждой функции желательно написать документацию и создать тестовую программу для проверки работы функций.

Тема 9. Тестирование и отладка кода

Теоретическое занятие

Тестирование кода. Отладка кода с использованием отладчика. Использование assert для проверки условий.

Практическое занятие

Задание:

1. Проанализируйте задачу: определите ключевые компоненты и возможные подводные камни.

2. Создайте алгоритм решения задачи.

3. Напишите код на Python, следуя принципам DRY (Don’t Repeat Yourself) и используя функции и модули для повторного использования кода.

4. Протестируйте код, используя модульное, интеграционное и системное тестирование, а также ручное тестирование.

5. После успешного прохождения тестов выполните отладку кода, выявляя и исправляя ошибки.

6. Выполните рефакторинг кода, улучшая его структуру и эффективность.

Тема 10. Итоговый проект

Практическое занятие

Задание:

Создайте архив по творчеству Николая Лескова, используя язык программирования Python. В архиве должны быть представлены следующие разделы:

- Биография писателя.
- Основные произведения.
- Анализ выбранных произведений.
- Иллюстрации к произведениям.

Для создания архива используйте следующие шаги:

1. Сбор и ознакомление с информацией о Н. С. Лескове и его творчестве.
2. Анализ и систематизация полученной информации.
3. Написание алгоритма создания программы.
4. Создание программы на языке программирования Python.

Ваша программа должна предоставлять удобный интерфейс для просмотра и навигации по разделам архива, а также позволять добавлять новые произведения и иллюстрации.

ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ

Задания:

Задача 1: Списки и циклы

Напишите функцию average_of_evens(lst), которая принимает список целых чисел и возвращает среднее значение всех четных чисел в списке. Если четных чисел нет, функция должна вернуть 0.

```
def average_of_evens(lst):
    # Ваш код здесь
    pass
```

Пример использования:

```
print(average_of_evens([1, 2, 3, 4, 5, 6]))  # Ожидаемый результат: 4.0
print(average_of_evens([1, 3, 5]))           # Ожидаемый результат: 0
```

Задача 2: Работа со строками

Напишите функцию is_palindrome(s), которая принимает строку и возвращает True, если строка является палиндромом (читается одинаково слева направо и справа налево), и False в противном случае.

```
def is_palindrome(s):
    # Ваш код здесь
    pass
```

Пример использования:

```
print(is_palindrome("radar"))  # Ожидаемый результат: True
print(is_palindrome("hello"))  # Ожидаемый результат: False
```

МОДУЛЬ 6. ЯЗЫК ПРОГРАММИРОВАНИЯ JAVASCRIPT

JavaScript-разработчики всегда необходимы, поэтому выучив этот язык вы можете быть уверены, что без работы и хорошей зарплаты вы точно не останетесь. Он нужен практически во всех сферах экономики: банковской системе, IT-компаниях, логистике и т.д. Чтобы обучающийся в будущем мог стать востребованным специалистом, его познакомят с JavaScript и основами практического программирования, для разработки собственного мобильного приложение с нуля.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие
1	Язык HTML. Структура программы, основные теги	0,5	0,5	
2	Форматирование текста	1,5	0,5	1
3	Списки, гиперссылки, добавление мультимедиа	1,5	0,5	1
4	Таблицы в HTML	1,5	0,5	1
5	Операторы языка программирования JavaScript	1,5	0,5	1
6	Условный оператор	1,5	0,5	1
7	Циклы	1,5	0,5	1
8	Функции и команды	1,5	0,5	1
9	Объект, его свойства и методы	1,5	0,5	1
10	Формы	1,5	0,5	1
	Итого	14	5	9

Тема 1. Язык HTML. Структура программы, основные теги

Теоретическое занятие

Язык программирования JavaScript. Переменные, функции, события, DOM. Примеры использования HTML, CSS и JavaScript.

Тема 2. Форматирование текста

Теоретическое занятие

Введение в форматирование текста в JavaScript. Использование пробелов и табуляции для выравнивания кода. Применение комментариев для пояснения кода. Управление отступами и переносами строк. Использование фигурных и круглых скобок для группировки кода. Применение ключевых слов и операторов для создания выражений.

Практическое занятие

Задание:

1. Создайте массив с именами и возрастом студентов.
2. Отформатируйте каждую строку массива, заменив имя студента на «Студент №» и добавив возраст после пробела.
3. Используйте метод join() для объединения отформатированных строк в одну строку.
4. Выведите полученную строку на экран.

Тема 3. Списки, гиперссылки, добавление мультимедиа

Теоретическое занятие

Создание гиперссылок. Описание тега гипертекстовой ссылки. Фреймы в HTML. Неупорядоченные списки. Упорядоченные списки. Списки определений: создание и оформление списков терминов и определений.

Практическое занятие

Задание:

1. Создайте веб-страницу с двумя разделами: «Список» и «Гиперссылки».
2. В разделе «Список» используйте неупорядоченный список для отображения списка товаров.
3. В разделе «Гиперссылки» создайте гиперссылки на другие страницы вашего сайта или внешние ресурсы.
4. Добавьте мультимедиа: загрузите изображение товара и видео с YouTube.
5. Реализуйте возможность просмотра видео при нажатии на соответствующую гиперссылку.
6. Сохраните изменения и проверьте работу вашей веб-страницы в разных браузерах.

Тема 4. Таблицы в HTML

Теоретическое занятие

Таблицы в HTML и их структура. Создание таблиц с помощью HTML: тег <table>; атрибуты width и height; cellpadding и cellspacing. Стилизация таблиц с помощью CSS. Создание таблиц с использованием JavaScript. Добавление данных в ячейки таблицы. Обработка событий в таблицах.

Практическое занятие

Задание:

Создайте таблицу, состоящую из трёх столбцов и пяти строк, используя методы и свойства:

document.createElement() для создания элемента <table>;

insertRow() для добавления строк;

insertCell() для добавления ячеек;

innerHTML для заполнения контента.

Пример кода:

```
lettable = document.createElement('table');
let row = table.insertRow();
let cell1 = row.insertCell();
let cell2 = row.insertCell();
let cell3 = row.insertCell();
cell1.innerHTML = 'Текст ячейки 1';
cell2.innerHTML = 'Текст ячейки 2';
cell3.innerHTML = 'Текст ячейки 3';
document.body.appendChild(table);
```

Тема 5. Операторы языка программирования JavaScript

Теоретическое занятие

Основные операторы JavaScript: математические, логические и битовые операции. Унарные операторы и их использование. Таблица приоритетов операторов и порядок выполнения операций. Операторы присваивания: +=, -=, *=, /=, %=. Сравнение переменных: строгое равенство (===) и неравенство (!==). Арифметические операторы. Битовые (поразрядные) операторы. Битовые операторы сдвига. Логические операторы.

Практическое занятие

Задание:

1. Создайте HTML-файл с формой для ввода чисел и операторами сложения, вычитания, умножения и деления.

2. Добавьте следующий код JavaScript для обработки нажатий на кнопки:

```
function calculate() {
 let result = 0;
 let operator = "";

 if (isNaN(inputValue)) {
 alert("Введено не число!");
 return;
 }

 if (inputValue == "-=") {
```

```
 operator = "-";
 inputValue = inputValue.replace("-=", "");
 } else if (inputValue == "+=") {
 operator = "+";
 inputValue = inputValue.replace("+=", "");
 } else if (inputValue == "*=") {
 operator = "*";
 inputValue = inputValue.replace("*=", "");
 } else if (inputValue == "/=") {
 operator = "/";
 inputValue = inputValue.replace("/=", "");
 } else {
 result = eval(inputValue);
 }

 document.getElementById("result").innerHTML = result;
}

function clearInput() {
 document.getElementById("inputValue").value = "";
 document.getElementById("result").innerHTML = "";
}

window.onload = function () {
 clearInput();
};
```

3. Протестируйте калькулятор, вводя различные числа и нажимая на кнопки операторов. Убедитесь, что результат отображается корректно.

Тема 6. Условный оператор

Теоретическое занятие

Синтаксис условного оператора if…then. Составные условия. Альтернативный оператор: (условная операция). Вложенные условные операторы. Оператор switch.

Практическое занятие

Задание:

1. Создайте функцию isPrime(), которая принимает число n в качестве аргумента и возвращает true, если число простое, и false в противном случае.

2. Используйте условный оператор if…else для проверки числа на простоту.

3. Протестируйте программу, вводя различные числа и проверяя результаты.

Тема 7. Циклы

Теоретическое занятие

Введение в циклы: зачем они нужны в JavaScript. Понятие цикла и его определение. Виды циклов в JavaScript. Синтаксис и основные элементы циклов. Необязательные части циклов и их использование. Инструкции break и continue в циклах. Меты в циклах и их использование.

Практическое занятие

Задание:

Используя цикл for, создайте таблицу умножения от 1 до 10. Каждая строка таблицы должна содержать числа от 1 до 10, разделённые пробелами.

Тема 8. Функции и команды

Теоретическое занятие

Функции и команды JavaScript как основа веб-разработки. Основы функций. Функции как объекты первого класса. Область видимости и замыкания: локальные и глобальные переменные. Рекурсия. Анонимные функции и стрелочные функции. Функциональное программирование: преимущества и применение в реальных проектах. Методы массивов и объектов: map, filter, reduce и другие полезные методы. Тестирование и отладка функций: инструменты и подходы.

Практическое занятие

Задание:

1. Создайте функцию sumArray(массив), которая принимает массив чисел в качестве аргумента.
2. Внутри функции вычислите сумму всех элементов массива и сохраните результат в переменной.
3. Верните значение переменной с суммой элементов массива.
4. Протестируйте функцию, передавая различные массивы чисел и проверяя результаты.

Тема 9. Объект, его свойства и методы

Теоретическое занятие

Объекты и их роль в JavaScript. Создание объекта: использование конструктора Object и литеральной нотации. Свойства объекта: определение и присвоение значений. Сокращённый способ определения свойств. Методы объекта: определение и вызов. Ключевые слова this и ссылки на текущий объект. Сокращённый способ определения методов.

Практическое занятие

Задание:

Создание объекта с именем person, содержащего свойства name (строка) и age (число). Создание метода printInfo(), который выводит информацию об объекте на экран.

Тема 10. Формы

Теоретическое занятие

Формы HTML и их использование. Взаимодействие с формами с помощью JavaScript. Основные элементы формы: форма, текстовые поля, поля для ввода пароля, переключатели и флажки. Обращение к элементам формы с использованием JavaScript. Обработчики событий: onclick, onsubmit и другие. Примеры использования форм и обработчиков событий в JavaScript. Безопасность и защита данных в формах.

Практическое занятие

Задание:

Создайте форму с текстовым полем и кнопкой отправки. При отправке формы проверьте правильность введённого значения и, если оно корректно, отправьте данные на сервер с помощью AJAX.

МОДУЛЬ 7. ЯЗЫК ПРОГРАММИРОВАНИЯ JAVA

Востребованность Java-разработчиков сложно переоценить. Большинство Android-приложений написано на данном языке программирования, а это более 7 миллиардов устройств во всем мире. Наша задача передать детям как можно больше практических навыков, для этого обучающийся познакомится с основами суперпопулярного языка Java, узнает, что такое ООП (Объектно-Ориентированное Программирование) и почему это круто, напишет собственное Java-приложение, которое в силу универсальности языка, будет работать на любом компьютере.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие
1	Основы программирования на Java	3	3	
2	Объектно-ориентированное программирование на Java	3	1	2
3	Работа над проектом	8		8
	Итого	14	4	10

Тема 1. Основы программирования на Java

Теоретическое занятие

Лексика и система типов данных в Java: примитивные и ссылочные типы, переменные и константы. Объекты и классы: создание, наследование и полиморфизм. Методы и конструкторы: определение, вызов и перегрузка.

Управление памятью: сборка мусора и работа с памятью. Исключения и обработка ошибок: создание и обработка исключений, try-catch-finally блок. Строки и символьные литералы: работа со строками и символами. Массивы и коллекции: одномерные и многомерные массивы, работа с коллекциями. Ввод/вывод данных: чтение и запись данных с использованием потоков. Графика и пользовательский интерфейс: работа с графикой и создание оконного интерфейса. Многопоточность и параллельное выполнение: основы многопоточности и синхронизация. Сетевое взаимодействие и работа с базами данных: основы сетевого взаимодействия и работа с базами данных. Тестирование и отладка кода.

Тема 2. Объектно-ориентированное программирование на Java

Теоретическое занятие

Введение в принципы объектно-ориентированного программирования (ООП). Абстракция: отвлечение от целостности объекта и выделение его главных свойств и составляющих. Инкапсуляция: улучшение качества взаимодействия вещей за счёт упрощения доступа к данным и функциям. Наследование: создание новых классов на основе существующих с возможностью добавления или изменения функционала. Полиморфизм: способность объектов с одинаковым интерфейсом выполнять различные действия в зависимости от их реализации.

Практическое занятие

Задание:

Создайте класс Animal с методами move() и sound(). Затем создайте подклассы Cat и Dog, которые наследуют методы от класса Animal. В классе Cat переопределите метод sound() и добавьте новый метод purr(). В классе Dog переопределите метод sound() и добавьте новый метод bark().

В основном классе создайте экземпляры объектов Animal, Cat и Dog и вызовите все доступные методы для вывода информации о поведении животных.

Тема 3. Работа над проектом

Практическое занятие

Создайте приложение на Java, которое будет считать сумму чисел, введённых пользователем с клавиатуры. Вот шаги для выполнения задания:

1. Создайте новый проект в среде разработки, например, в Eclipse.
2. Создайте новый класс с именем SumCalculator.
3. Добавьте следующий код в класс SumCalculator:

```
public class SumCalculator {
 int sum;

 public void calculateSum(int num1, int num2) {
```

```
sum = num1 + num2;
}

public int getSum() {
return sum;
}
}
```

4. Создайте новый файл с именем SumCalculatorTest.java и добавьте следующий код:

```
public class SumCalculatorTest {
public static void main(String args) {
SumCalculator calculator = new SumCalculator();

System.out.println("Введите первое число: ");
int num1 = Integer.parseInt(System.console().readLine());

System.out.println("Введите второе число: ");
int num2 = Integer.parseInt(System.console().readLine());

calculator.calculateSum(num1, num2);

System.out.println("Сумма чисел равна " + calculator.getSum());
}
}
```

5. Запустите программу, выполнив команду javac SumCalculatorTest.java в командной строке, а затем java SumCalculatorTest.

МОДУЛЬ 8. PHP+SQL

Каждый день вы проверяете электронную почту, переводите деньги онлайн, покупаете на маркетплейсах или пользуетесь поисковыми системами.

Для работы большинства сервисов используется базы данных и язык запросов SQL, а совокупность PHP + SQL предоставляет возможность создавать крупные и надежные web-проекты. Знания PHP + SQL помогут обучающимся создавать интернет-порталы и социальные сети.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие
1	Основы PHP	2	2	
2	Основы SQL	2	2	
3	Практика	2		2
4	Продвинутое программирование	2	2	
5	Проектная работа	6		6

	Итого	14	6	8

Тема 1. Основы PHP

Теоретическое занятие

Введение в язык PHP. Переменные и типы данных. Операторы и условия. Циклы и функции. Работа с массивами и сессиями. Подключение файлов и работа с ошибками.

Тема 2. Основы SQL

Теоретическое занятие

Введение в язык SQL. Команды SELECT, INSERT, UPDATE и DELETE. Операторы WHERE, AND, OR и NOT. Использование функций и операторов. Работа с данными и таблицами.

Тема 3. Практика

Практическое занятие

Задание:

1. Создание простого сайта на PHP с базой данных MySQL.
2. Разработка и реализация функций и методов.
3. Работа с формами и отправка данных на сервер.
4. Защита сайта от взлома и уязвимостей.

Тема 4. Продвинутое программирование

Теоретическое занятие

Работа с переменными и типами данных. Использование ООП и шаблонов проектирования. Интеграция с другими языками и технологиями. Оптимизация производительности и масштабируемость.

Тема 5. Проектная работа

Практическое занятие

Задание:

1. Выбор и разработка проекта на основе полученных знаний.
2. Реализация проекта с использованием изученных технологий.
3. Тестирование и отладка проекта.
4. Презентация проекта перед педагогом и обучающимися.

МОДУЛЬ 9. ЯЗЫК ПРОГРАММИРОВАНИЯ C++

На языках программирования вы можете написать все, что угодно, а на C++ вы можете написать сам язык программирования!

В процессе обучения мы изучим основы высокоуровнего языка, создадим разнообразные прикладные программы, напишем драйвера для установки устройств и приложений.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие
1	Введение в C++	2	2	
2	Основы синтаксиса и базовые конструкции	1		1
3	Условная операция, циклы и массивы	1		1
4	Ссылки, функции и строки	1		1
5	Исключения, шаблоны функций и пользовательские типы данных	1		1
6	Введение в объектно-ориентированное программирование (ООП)	2		2
7	Конструкторы, перегрузка операций и функторы	1	1	
8	Контейнеры, итераторы и наследование	2		2
9	Полиморфизм и шаблоны проектирования	3		3
	Итого	14	3	11

Тема 1. Введение в C++

Теоретическое занятие

Алфавит языка. Ключевые слова и идентификаторы. Структура программы на C++. Функции в C++. Основные операторы и выражения. Типы данных и модификаторы. Ввод и вывод информации на консоль. Пространство имён и конфликты имён. Библиотека iostream и стандартные потоки.

Тема 2. Основы синтаксиса и базовые конструкции

Практическое занятие

Задание:

Создайте программу на C++, которая выводит на экран фразу «Привет, мир!».

Тема 3. Условная операция, циклы и массивы

Практическое занятие

Задание:

Дан массив чисел. Если в нём есть два соседних элемента одного знака, выведите эти числа. Если соседних элементов одного знака нет — не выводите ничего. Если таких пар соседей несколько — выведите первую пару.

Выведите значение наименьшего чётного элемента списка, а если в списке нет чётных элементов — выведите число 0.

Тема 4. Ссылки, функции и строки

Практическое занятие

Задание:

1. Напишите функцию, которая принимает строку (std::string) по ссылке и добавляет к ней восклицательный знак '!'.

2. Что произойдет, если передать в функцию, принимающую параметр по ссылке, временный объект? Приведите пример кода и объясните поведение.

```
void printAndModify(std::string& s) {
    std::cout << s << std::endl;
    s = "Modified";
}

int main() {
    printAndModify("Temporary");
    return 0;
}
```

Тема 5. Исключения, шаблоны функций и пользовательские типы данных

Практическое занятие

Задание:

1. Реализуйте функцию, которая принимает два целых числа и делит первое число на второе. Если второе число равно нулю, должно выбрасываться исключение. Напишите код для перехвата этого исключения и вывода сообщения об ошибке.

```
#include <iostream>
#include <stdexcept>

int divide(int a, int b) {
    if (b == 0) {
        throw std::invalid_argument("Division by zero is undefined.");
    }
    return a / b;
}

int main() {
    try {
        int result = divide(10, 0);
        std::cout << "Result: " << result << std::endl;
    } catch (const std::invalid_argument& e) {
        std::cerr << "Error: " << e.what() << std::endl;
    }
    return 0;
}
```

2. Шаблоны функций:

Напишите шаблон функции, которая принимает два аргумента любого типа и возвращает их сумму. Протестируйте вашу функцию с аргументами различных типов (int, double, и т.д.).

```
#include <iostream>

template<typename T>
T add(T a, T b) {
    return a + b;
}

int main() {
    std::cout << "Sum of 3 and 7: " << add(3, 7) << std::endl;
    std::cout << "Sum of 4.3 and 5.2: " << add(4.3, 5.2) << std::endl;
    std::cout << "Sum of 'hello' and 'world': " << add(std::string("hello"), std::string("world")) << std::endl;
    return 0;
}
```

3. Пользовательские типы данных:

Определите класс Rectangle, представляющий прямоугольник, с закрытыми полями width и height, а также публичными методами для установки и получения этих полей. Реализуйте метод, вычисляющий площадь прямоугольника. Протестируйте ваш класс в main().

```
#include <iostream>

class Rectangle {
private:
    double width;
    double height;

public:
    void setWidth(double w) {
        if (w >= 0) width = w;
    }
    void setHeight(double h) {
        if (h >= 0) height = h;
    }

    double getWidth() const {
        return width;
    }
    double getHeight() const {
        return height;
    }

    double area() const {
        return width * height;
    }
};

int main() {
    Rectangle rect;
    rect.setWidth(4.5);
    rect.setHeight(3.2);
    std::cout << "Area of rectangle: " << rect.area() << std::endl;
    return 0;
}
```

Тема 6. Введение в объектно-ориентированное программирование (ООП)

Практическое занятие

Задание 1: Создание классов и объектов

Создайте класс Person со следующими атрибутами:

Имя (строка)

Возраст (целое число)

Пол (строка)

Добавьте методы:

Конструктор для инициализации данных членов.

Метод displayInfo(), который выводит информацию о человеке.

Напишите программу, которая создает несколько объектов класса Person и выводит информацию о каждом из них.

Задание 2: Инкапсуляция

Добавьте к классу Person приватные члены для имени, возраста и пола.

Создайте геттеры и сеттеры для каждого из этих атрибутов.

Продемонстрируйте использование этих методов в программе.

Тема 7. Конструкторы, перегрузка операций и функторы

Теоретическое занятие

Важность конструкторов, перегрузки операций и функторов в C++. Примеры использования этих инструментов в реальных проектах. Конструкторы: определение конструктора; виды конструкторов (конструктор по умолчанию, конструктор копирования, конструктор преобразования); правила создания и инициализации объектов с использованием конструкторов. Перегрузка операций. Функторы.

Тема 8. Контейнеры, итераторы и наследование

Практическое занятие

Задание 1: Наследование

Создайте класс Student, который наследует класс Person.

Добавьте к классу Student следующие атрибуты:

Студенческий билет (строка)

Специальность (строка)

Добавьте методы:

Конструктор для инициализации данных членов, включая члены базового класса.

Метод displayInfo(), который переопределяет метод из базового класса и добавляет информацию о студенческом билете и специальности.

Напишите программу, которая создает объект класса Student и выводит его информацию.

Задание 2: Полиморфизм

Создайте класс Teacher, который также наследует класс Person.

Добавьте к классу Teacher следующие атрибуты:

Предмет (строка)

Добавьте методы:

Конструктор для инициализации данных членов, включая члены базового класса.

Метод displayInfo(), который переопределяет метод из базового класса и добавляет информацию о предмете.

Напишите программу, которая создает объект класса Student и объект класса Teacher, сохраняет их в массив указателей типа Person* и использует полиморфизм, чтобы вызвать метод displayInfo() для каждого объекта.

Указания по выполнению:

Вся работа должна быть выполнена в одном или нескольких файлах C++.

Постарайтесь придерживаться принципов чистого кода и комментариев, чтобы сделать ваш код читаемым и понятным.

Используйте стандартный ввод и вывод для взаимодействия с пользователем (если это требуется).

Тема 9. Полиморфизм и шаблоны проектирования

Практическое занятие

Задание:

Реализуйте следующую иерархию классов с использованием полиморфизма:

1. Создайте абстрактный базовый класс Shape, который содержит одну чисто виртуальную функцию:

virtual double area() const = 0;

2. Создайте классы-наследники Circle, Rectangle и Triangle, которые будут наследовать класс Shape и реализовывать функцию area().

3. Класс Circle должен содержать одно поле radius, класс Rectangle - два поля width и height, класс Triangle - два поля base и height.

4. Напишите тестовую программу, которая создаст массив указателей на объекты класса Shape, заполнит его объектами Circle, Rectangle и Triangle с различными значениями полей и выведет площади всех фигур на экран.

Примерный скелет кода:

```
#include <iostream>
#include <vector>
#include <memory>

class Shape {
public:
    virtual double area() const = 0;
```

```
    virtual ~Shape() = default;
};

class Circle : public Shape {
    double radius;
public:
    Circle(double r) : radius(r) {}
    double area() const override {
        return 3.14159 * radius * radius;
    }
};

class Rectangle : public Shape {
    double width, height;
public:
    Rectangle(double w, double h) : width(w), height(h) {}
    double area() const override {
        return width * height;
    }
};

class Triangle : public Shape {
    double base, height;
public:
    Triangle(double b, double h) : base(b), height(h) {}
    double area() const override {
        return 0.5 * base * height;
    }
};

int main() {
    std::vector<std::unique_ptr<Shape>> shapes;
    shapes.push_back(std::make_unique<Circle>(5.0));
    shapes.push_back(std::make_unique<Rectangle>(4.0, 6.0));
    shapes.push_back(std::make_unique<Triangle>(3.0, 7.0));

    for (const auto& shape : shapes) {
        std::cout << "Area: " << shape->area() << std::endl;
    }
    return 0;
}
```

МОДУЛЬ 10. ЯЗЫК ПРОГРАММИРОВАНИЯ C#

Язык программирования C# (Си шарп) разработан компанией Microsoft. Изначально его создавали для проектов под Windows, но теперь это по-настоящему универсальный язык, на котором обучающиеся смогут написать десктопные приложения, веб-сервисы, нейросети и даже графику для метавселенных. Вне зависимости от того, захочет ли обучающийся стать программистом или же он предпочтет другую сферу занятий, знание программирования станет ценным инструментом в любой профессиональной области.

Учебно-тематический план

№ п/п	Наименование тем	Всего	Теоретическое занятие	Практическое занятие
1	Введение в программирование на C#	1,5	1,5	
2	Условные выражения и конструкции	2,5	1	1,5
3	Циклы	2,5	1	1,5
4	Массивы	2,5	1	1,5
5	Проектное занятие	5		5
	Итого	14	4,5	9,5

Тема 1. Введение в программирование на C#

Теоретическое занятие

Цели и задачи программирования, знакомство с переменными и типами данных, математические операции.

Тема 2. Условные выражения и конструкции

Теоретическое занятие

Введение: условные выражения и конструкции в C#. Оператор if: структура и примеры использования. Оператор else: добавление альтернативного кода. Оператор else if: вложение нескольких условий. Тернарный оператор.

Практическое занятие

Задание:

Создайте программу на C#, которая проверяет, является ли число чётным или нечётным. Используйте условный оператор if.

Тема 3. Циклы

Теоретическое занятие

Зачем нужны циклы в программировании. Цикл while: синтаксис, примеры использования. Цикл do-while: отличия от цикла while, синтаксис, примеры использования. Цикл for: синтаксис, примеры использования.

Инкремент и декремент: операторы увеличения и уменьшения значений переменных.

Практическое занятие

Задание:

Напишите программу на C#, которая выводит на экран степени двойки до заданного пользователем числа N. Используйте цикл while или do-while.

Тема 4. Массивы

Теоретическое занятие

Понятие массива и его использование в программировании. Объявление и инициализация одномерных массивов. Доступ к элементам одномерных массивов и изменение значений элементов. Многомерные массивы: объявление и инициализация. Доступ к элементам многомерных массивов и изменение значений элементов. Сортировка и поиск в массивах: методы и алгоритмы.

Практическое занятие

Задание:

Создание программы на C# для работы с одномерными массивами.

Тема 5. Проектное занятие

Практическое занятие

Мини-проект на C#: создание калькулятора для расчёта стоимости товаров.

1. Описание программы: калькулятор должен иметь функции сложения, вычитания, умножения и деления, а также возможность ввода цены товара и количества.

2. Краткое описание языка программирования C#: используйте язык C# для создания приложения Windows Forms.

3. Разработка программы: создайте форму с четырьмя текстовыми полями (для цены товара, количества, суммы и итога), четырьмя кнопками (сложение, вычитание, умножение и деление) и двумя метками (для отображения суммы и итога).

4. Написание инструкции для программы: предоставьте пользователю подробное описание функций калькулятора и инструкции по использованию программы.

5. Создание руководства пользователя: разработайте руководство, которое поможет пользователям легко освоить работу с калькулятором.

6. Тестирование программы: проверьте работоспособность калькулятора, вводя различные данные и выполняя арифметические операции.

7. Защита проекта: подготовьте презентацию или доклад, в котором опишите основные этапы разработки программы, её функциональность и результаты тестирования.

Итоговая аттестация

Итоговая аттестация проводится в форме зачета в соответствии с оценочными материалами, установленными приложением 1 настоящей образовательной программы.

5. УСЛОВИЯ РЕАЛИЗАЦИИ ДОПОЛНИТЕЛЬНОЙ ОБЩЕОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

5.1. Кадровые условия

Согласно профессиональному стандарту «Педагог дополнительного образования детей и взрослых» (утв. приказом Министерства труда и социальной защиты РФ от 22.09.2021 № 652н) к педагогу дополнительного образования предъявляются следующие требования к образованию и обучению:

Высшее образование или среднее профессиональное образование в рамках укрупненных групп специальностей и направлений подготовки высшего образования и специальностей среднего профессионального образования «Образование и педагогические науки»

или

Высшее образование либо среднее профессиональное образование в рамках иных укрупненных групп специальностей и направлений подготовки высшего образования и специальностей среднего профессионального образования при условии его соответствия дополнительным общеразвивающим программам, дополнительным предпрофессиональным программам, реализуемым организацией, осуществляющей образовательную деятельность, и получение при необходимости после трудоустройства дополнительного профессионального образования педагогической направленности

или

Успешное прохождение обучающимися промежуточной аттестации не менее чем за два года обучения по образовательным программам высшего образования по специальностям и направлениям подготовки, соответствующим направленности дополнительных общеобразовательных программ.

Требования к опыту практической работы: не менее двух лет в должности педагога дополнительного образования, иной должности педагогического работника - для старшего педагога дополнительного образования.

Особые условия допуска к работе:

- отсутствие ограничений на занятие педагогической деятельностью, установленных законодательством Российской Федерации;
- прохождение обязательных предварительных и периодических медицинских осмотров.

5.2. Материально-технические условия реализации дополнительной общеобразовательной программы

Образовательная организация располагает на законном основании материально-технической базой, обеспечивающей проведение всех видов учебной деятельности обучающихся, предусмотренных учебным планом.

Рабочее место педагога, оборудованное персональным компьютером с необходимым программным обеспечением общего и профессионального назначения.

Обучение осуществляется через систему дистанционного обучения – образовательную платформу «Skillspace» https://skillspace.ru/ .

Обеспечение функционирования информационно-коммуникационной системы:

1) ноутбук – 1 шт.
2) наушники – 1 шт.;
3) принтер – 1 шт.;
4) мультимедийный проектор с экраном – 1 шт.;
5) устройства для накопления и хранения информации – 1 шт.;
6) роутер – 1 шт.

5.3. Учебно-методическое обеспечение дополнительной общеобразовательной программы:

1. Богун В. В. Обработка форм статических Интернет-страниц с применением языка сценариев JavaScript [Текст]: учебное пособие. – Ярославль: РИО ЯГПУ, 2015. – 80 с. Режим доступа: https://www.elibrary.ru/item.asp?id=23670881

2. Горячкин Б.С., Белоногов И.Б. Практикум для редактирования и моделирования 3D графики на основе программного продукта Blender 2.92: Учебно-методическое пособие. Часть 3. – М.: Издательство «Спутник +», 2021. – 28 с. Режим доступа: https://www.elibrary.ru/item.asp?id=46153927

3. Наумов В. Ю. Информатика и программирование: лабораторный практикум по программированию на языке С++: учеб.-метод. пособие / В. Ю. Наумов, О. А. Авдеюк; ВолгГТУ. – 2-е изд., испр. – Волгоград, 2018. – 136 с. Режим доступа: https://www.elibrary.ru/item.asp?id=32683157

4. Основы объектно-ориентированного анализа и программирования в Python: учебно-методическое пособие / сост. Т. М. Босенко. – М.: МГПУ, 2023. – 80 с. Режим доступа: https://www.elibrary.ru/item.asp?id=50448967

5. Программирование. Объектно-ориентированное программирование: Учебное пособие / Ярославское высшее военное училище противовоздушной обороны. – Ярославль, 2015 – 412 с. Режим доступа: https://www.elibrary.ru/item.asp?id=32416822

5.4. Режим занятий и организация учебного процесса

Образовательный процесс осуществляется на основании учебного плана, календарного учебного графика и регламентируется расписанием занятий для каждой учебной группы.

Продолжительность одного академического часа составляет 40 минут.

Выбор методов обучения для каждого занятия определяется педагогом в соответствии с составом и уровнем подготовленности обучающихся, степенью сложности излагаемого материала, наличием и состоянием учебного оборудования, технических средств обучения, продолжительностью проведения занятий.

Теоретические занятия проводятся с целью изучения нового учебного материала. Изложение материала необходимо вести в форме, доступной для

понимания обучающихся, соблюдать единство терминологии, определений и условных обозначений, соответствующих международным договорам и нормативным правовым актам. В ходе занятий педагог обязан соотносить новый материал с ранее изученным, дополнять основные положения примерами из практики, соблюдать логическую последовательность изложения.

Практические занятия проводятся с целью закрепления теоретических знаний и выработки у обучающихся основных умений и навыков работы в ситуациях, максимально имитирующих реальные процессы.

6. ФОРМЫ АТТЕСТАЦИИ

При реализации образовательной программы оценка результатов освоения программы проводится в рамках текущего контроля успеваемости, промежуточной и итоговой аттестации.

Текущий контроль успеваемости

Текущий контроль успеваемости осуществляется в ходе изучения соответствующих тем программы.

Текущий контроль успеваемости проводится с целью получения оперативной информации о качестве усвоения обучающимися учебного материала, управления учебным процессом и совершенствования методики проведения занятий.

Форма текущего контроля – контроль активности на образовательной платформе, анализ выполненных практических заданий.

Промежуточная аттестация

Промежуточная аттестация обучающихся предназначена для определения степени достижения планируемых результатов обучения.

Форма промежуточной аттестации – зачет.

Промежуточная аттестация проводится после освоения отдельных тем образовательной программы в соответствии с расписанием учебных занятий.

При проведении промежуточной аттестации используются **оценочные материалы,** установленные рабочей программой (раздел 4 настоящей образовательной программы).

При проведении промежуточной аттестации применятся зачетная система оценки: «зачтено»/ «не зачтено».

При проведении промежуточной аттестации применятся зачетная система оценки: «зачтено»/ «не зачтено».

Оценка	Критерии оценки
Зачтено	Обучающийся демонстрирует исчерпывающие знания всего программного материала, глубокое понимание сущности и взаимосвязи рассматриваемых процессов и явлений, твёрдое знание изученного материала программы. Дает логически последовательные, содержательные, полные, правильные и конкретные ответы на все вопросы. Умело использует полученные теоретические знания.
Не зачтено	Обучающийся демонстрирует знание и понимание большей части основных вопросов, дает частичные ответы на поставленные вопросы. При этом обучающийся не может использовать основные знания по каждому вопросу, не всегда может синтезировать имеющуюся информацию и интегрировать знания. Обучающийся недостаточно полно использует полученные знания для решения поставленных задач. Допущены неточности и ошибки в ответах на дополнительные вопросы.

Итоговая аттестация обучающихся

Освоение программы завершается итоговой аттестацией обучающихся.

Форма итоговой аттестации: зачет. Зачет проводится в виде тестирования.

К итоговой аттестации допускается обучающийся, не имеющий академической задолженности и в полном объеме выполнивший учебный план (индивидуальный учебный план) по программе.

Объем времени аттестационных испытаний, входящих в итоговую аттестацию обучающихся, устанавливается учебным планом.

Лицам, не прошедшим итоговой аттестации или получившим на итоговой аттестации неудовлетворительные результаты, а также лицам, освоившим часть образовательной программы и (или) отчисленным из организации, осуществляющей образовательную деятельность, выдается справка об обучении или о периоде обучения по образцу, самостоятельно устанавливаемой организацией, осуществляющей образовательную деятельность.

Оценочные материалы итоговой аттестации устанавливаются Приложением 1 настоящей образовательной программы.

При проведении итоговой аттестации применятся зачетная система оценки: «зачтено»/ «не зачтено».

Критерием оценки служит следующая шкала количества верных ответов (в %):

0–70 % – не зачтено;

71%–100% – зачтено.

7. МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ

Методические указания по освоению программы

Вид учебных занятий	**Методические указания для обучающихся по освоению модуля по видам учебных занятий**
Теоретическое занятие	При изучении материалов темы необходимо в первую очередь прослушать теорию. При прослушивании теории необходимо вести конспект. Изучение тем требует систематического и последовательного накопления знаний, поэтому пропуски отдельных тем не позволяют глубоко освоить программу. При конспектировании целесообразно вначале понять основную мысль, излагаемую педагогом, а затем записать ее. Желательно запись осуществлять на одной странице листа или оставляя поля, на которых позднее, при самостоятельной работе с конспектом, можно сделать дополнительные записи, отметить непонятные места. Обучающимся необходимо: - перед каждым теоретическим занятием просматривать рабочую программу, что позволит сэкономить время на записывание темы занятия, ее основных вопросов; - перед очередным теоретическим занятием необходимо просмотреть по конспекту материал предыдущего занятия. В случае возникновения вопросов по теме занятия педагогом предоставляется обратная связь – через электронную почту, комментарии на образовательной платформе или онлайн-консультациях. Обратная связь, диалог, общение с обучающимися является неотъемлемой частью обучения. Теоретические занятия проводятся в режиме видеоконференцсвязи через сервис «МТС Линк» https://events.webinar.ru/
Практическое занятие	К выполнению практического задания необходимо приступать после прослушивания теории. Практическое задание необходимо выполнить в соответствии с условиями, размещенными на образовательной платформе. В случае возникновения вопросов по заданию педагогом предоставляется обратная связь. Обратная связь, диалог, общение с обучающимися является неотъемлемой частью обучения.

	В случае возникновения вопросов по теме занятия педагогом предоставляется обратная связь – через электронную почту, комментарии на образовательной платформе или онлайн-консультациях.
Подготовка к промежуточной аттестации	Прежде чем приступить к выполнению задания промежуточной аттестации обучающийся должен освоить теорию, выполнить практические задания. Работа с конспектом теоретических занятий, записями теоретических занятий, учебно-методическими материалами, размещенными в электронной библиотеке на образовательной платформе.
Подготовка к итоговой аттестации	Прежде чем приступить к выполнению задания итоговой аттестации обучающийся должен освоить теорию, выполнить практические задания. Работа с конспектом теоретических занятий, записями теоретических занятий, учебно-методическими материалами, размещенными в электронной библиотеке на образовательной платформе.

Приложение 1

Оценочные материалы итоговой аттестации

1. Что представляет собой объект «Cube» по умолчанию в Blender?
Многоугольник с пятью вершинами
Трехмерный объект с шестью квадратными гранями
Плоский объект, состоящий из одного треугольника
Объект, имеющий только одну вершину

2. Какой модификатор используется для создания плавности и сглаживания модели?
Decimate
Array
Subdivision Surface
Boolean

3. Какая клавиша на клавиатуре позволяет быстро переключаться между режимом объекта и режимом редактирования?
Tab
Spacebar
E
Ctrl + E

4. В каком из режимов можно добавлять свет и камеры в сцену?
Режим скульптинга
Режим текстуризирования
Режим редактирования
Режим объекта

5. Какой тип света используется в Blender для имитации света от солнца?
Point Light (точечный источник света)
Area Light (плоский источник света)
Sun Light (солнечный свет)
Spot Light (прожекторный свет)

6. Что такое Figma и для чего используется этот инструмент?
Это инструмент для создания 3D-моделей
Это текстовый редактор, предназначенный для написания кода
Это онлайн-инструмент для дизайна пользовательских интерфейсов и совместной работы над проектами
Это программа для редактирования фотографий

7. Какие из следующих функций НЕ поддерживаются в Figma?

Создание векторной графики

Совместное редактирование в реальном времени

Управление версионностью дизайна

Написание и выполнение JavaScript кода

8. В каком разделе Figma можно найти все элементы, используемые в проекте, включая слои, группы и фреймы?

Панель инструментов

Панель слоев

Панель свойств

Панель шрифтов

9. Какую функцию выполняет инструмент «Frame» в Figma?

Создание текстовых блоков

Расположение и группировка элементов дизайна в соответствии с выбранными размерами экрана

Генерация CSS кода для элементов дизайна

Отмена последних изменений

10. Как можно поделиться проектом в Figma с другими пользователями, чтобы они могли его редактировать?

Сохранить проект в формате PDF и отправить по электронной почте

Создать ссылку на проект и предоставить определенные права доступа пользователям, используя настройки общего доступа

Экспортировать все элементы проекта и передать их пользователям для редактирования в другом редакторе

Сделать скриншоты проекта и отправить их пользователям для ознакомления

11. Что из перечисленного НЕ является валидным HTML-элементом?

<header>

<footer>

<section>

<bottom>

12. Какой атрибут используется для определения уникального идентификатора HTML-элемента?

class
id
name
data-id

13. Какой CSS-правило применит стиль color: red; к тексту внутри всех параграфов <p>?
p { background-color: red; }
p { color: red; }
.p { color: red; }
#p { color: red; }

14. Какой CSS-свойство используется для того, чтобы разметить элементы внутри контейнера с использованием Flexbox?
display: grid;
display: flex;
display: block;
display: inline-block;

15. Какое значение CSS-свойства position закрепляет элемент на странице относительно его начального положения, даже при прокрутке?
static
relative
absolute
d) fixed

16. Какой тип данных возвращает функция len() в Python?
int
float
list
str

17. Что выведет следующий код?
a = [1, 2, 3]
b = a
b.append(4)
print(a)

a) [1, 2, 3]
b) [1, 2, 3, 4]
c) Ошибка выполнения
d) [4, 3, 2, 1]

18. Какая функция используется для получения пользовательского ввода в Python 3?
raw_input()
input()
get_input()
read()

19. Какой результат выполнения следующего выражения?
result = 3 * '2'
print(result)
a) '6'
b) '32'
c) '222'
d) Ошибка выполнения

20. Что произойдет, если попытаться изменить кортеж (tuple) в Python? Например:
t = (1, 2, 3)
t[0] = 4
a) Кортеж будет изменен
b) Ошибка выполнения
c) Первое значение станет 4
d) Кортеж будет расширен