Государственное автономное образовательное учреждение дополнительного образования «Центр для одаренных детей «Поиск»

УТВЕРЖДЕНО
приказом Центра «Поиск»
№ 148 от 27 марта 2026 г.

Дополнительная общеобразовательная общеразвивающая программа технической направленности

«ПРОГРАММИРОВАНИЕ ДЛЯ ДЕТЕЙ»

Направление:	техническое
Возраст обучающихся:	6 -10 лет
Объем программы:	72 часа в год
Срок освоения:	3 года
Форма обучения:	очная

Авторы программы: Жалыбина Юлия Витальевна, заведующий Центром цифрового образования ЦЦО «IT-куб»
Месропян Татевик Григоровна, педагог дополнительного образования ЦЦО «IT-куб»

Михайловск, 2026

ОГЛАВЛЕНИЕ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Программа «Программирование для детей» разработана в соответствии с требованиями нормативных документов:

Федерального закона РФ от 29.12.2012 г. № 273-ФЗ «Об образовании в Российской Федерации».

Концепции развития дополнительного образования детей, утвержденной распоряжением Правительства РФ от 4 сентября 2014 г. № 1726-р.

Приказа Минпросвещения РФ от 09.11.2018 г. N 196 «Об утверждении порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам».

Постановления Главного государственного санитарного врача РФ от 28.09.2020 г. № 28 СП 2.4.3648-20 «Санитарно-эпидемиологические требования к организациям воспитания и обучения, отдыха и оздоровления детей и молодежи».

Методических рекомендаций по проектированию дополнительных общеразвивающих программ (письмо Минобрнауки РФ от 18.11.2015 г. N 09-3242).

Методических рекомендаций по созданию и функционированию центров цифрового образования «IT-куб» (утв. распоряжением Министерства просвещения Российской Федерации от 12.01.2021 № Р-5). Паспорт национального проекта «Образование» (утв. президиумом Совета при Президенте РФ по стратегическому развитию и национальным проектам, протокол от 24.12.2018 № 16).

Государственной программы Российской Федерации «Развитие образования» (утв. постановлением Правительства РФ от 26.12.2017 № 1642 (ред. от 15.03.2021) «Об утверждении государственной программы Российской Федерации “Развитие образования”»).

Стратегии развития воспитания в Российской Федерации на период до 2025 года (утв. распоряжением Правительства РФ от 29.05.2015 № 996-р «Об

утверждении Стратегии развития воспитания в Российской Федерации на период до 2025 года»).

1. ОСНОВНЫЕ ХАРАКТЕРИСТИКИ ПРОГРАММЫ

1.1. Направленность программы

Программа носит междисциплинарный характер и позволяет решить задачи развития у учащихся научно-исследовательских, технологических и гибких компетенций. В процессе освоения содержания, обучающиеся применяют математические знания (координаты, счёт, логика), реализуют проектную деятельность (планирование, распределение ролей, сроки), а также развивают навыки командной работы, презентации и рефлексии. Таким образом, техническая основа программы служит площадкой для формирования комплекса метапредметных умений.

1.2. Адресат программы

Программа адресована детям в возрасте от 6 до 10 лет и реализуется по трём дифференцированным уровням сложности: для дошкольников 6–7 лет, учащихся 1–2 классов и учащихся 3–4 классов. Предусмотрена разновозрастная структура групп внутри каждого модуля, а также дифференциация заданий по степени сложности для учёта индивидуальных темпов освоения материала.

1.3. Актуальность программы

Актуальность программы обусловлена современными тенденциями развития общества и требованиями к формированию функциональной грамотности обучающихся в условиях цифровой экономики.

В соответствии с запросами государства и рынка труда возрастает значимость развития у детей алгоритмического и логического мышления, навыков работы с информацией, способности решать задачи через проектирование и моделирование. Программа обеспечивает пропедевтику предметной области «Информатика» и создаёт условия для ранней профориентации в инженерно-технических и IT-направлениях.

Поэтапное освоение сред программирования позволяет учитывать возрастные особенности детей и обеспечивать преемственность в формировании цифровых компетенций. Проектная деятельность и защита результатов формируют метапредметные умения: планирование, командную работу, презентацию и рефлексию, — что соответствует современным образовательным стандартам.

1.4. Новизна программы

Новизна программы заключается в системном сочетании блочного программирования, проектной деятельности и межпредметных связей, обеспечивающем преемственность освоения цифровых компетенций с учётом возрастных особенностей детей 6–10 лет.

К числу принципиально новых элементов программы относятся:

– *Дифференцированная трёхуровневая структура,* позволяющая выстраивать индивидуальную траекторию развития: от первых представлений об алгоритме (модуль 1) через освоение базовых конструкций

программирования (модуль 2) к формированию инженерного подхода и подготовке к текстовым языкам (модуль 3).

– *Интеграция нескольких сред разработки* (ПиктоМир, Scratch Jr, Scratch, Kodu Game Lab, MakeCode Arcade) для многократного закрепления одних и тех же алгоритмических концепций в разных визуальных форматах — это повышает устойчивость навыков и снижает когнитивную нагрузку при переходе между средами.

– *Включение практик, характерных для реальной ИТ-разработки:* командные роли (программист, дизайнер, тестировщик), работа с ТЗ и раскадровкой, а также отладка и оптимизация кода — это формирует у детей представление о жизненном цикле цифрового продукта.

– *Систематическая связь с математическим содержанием*: координаты, счёт, логика, расчёт траекторий и игрового баланса интегрированы в проектные задачи, а не подаются изолированно, что обеспечивает практическую значимость математических знаний.

Таким образом, программа выходит за рамки традиционного обучения блочному программированию и формирует комплексную модель ранней ИТ-подготовки, ориентированную на дальнейшее освоение текстовых языков и участие в олимпиадах по информатике.

1.5. Объем и срок освоения программы

Уровень освоения программы — базовый (модуль «Юный исследователь»).

Объём программы — 72 часа.

Срок реализации программы — 1 год.

Уровень освоения программы — базовый (модуль «Создатель игр»).

Объём программы — 72 часа.

Срок реализации программы — 1 год.

Уровень освоения программы — углублённый (модуль «Инженер и архитектор кода»).

Объём программы — 72 часов.

Срок реализации программы — 1 год.

1.6. Цели и задачи программы

Цель программы: формирование алгоритмического и инженерного мышления у детей 6–10 лет через практическую разработку цифровых продуктов (игр, мультфильмов, 3D-миров) и поэтапное освоение инструментов программирования; развитие интереса к IT-сфере, поддержка творческой самореализации и создание основы для дальнейшего углублённого изучения информатики, в том числе в рамках олимпиадной подготовки.

Задачи программы

1. Образовательные:

– сформировать у обучающихся понимание базовых понятий информатики (алгоритм, исполнитель, переменная, функция, событие,

условие, цикл) на доступном возрастном уровне;

– познакомить с различными средами программирования и научить применять их для решения проектных задач;

– научить планировать и реализовывать проекты от идеи до презентации, включая составление технического задания, раскадровку, подбор графики и тестирование;

– обеспечить освоение математических понятий (координаты, счёт, сравнение величин, расчёт траекторий) в контексте программирования;

– сформировать навыки отладки и оптимизации кода, работы со списками, таймерами, генераторами случайных чисел, облачными переменными;

– подготовить обучающихся к переходу на текстовые языки программирования за счёт закрепления универсальных алгоритмических конструкций и инженерных практик.

2. Воспитательные:

– развивать мотивацию к самостоятельному поиску решений, изобретательству и созданию собственных цифровых продуктов;

– воспитывать ответственное отношение к результату труда, привычку доводить проект до завершённого состояния;

– формировать культуру командной работы: умение распределять роли (программист, дизайнер, тестировщик), договариваться, принимать обратную связь и корректировать работу;

– прививать навыки цифровой культуры и этики: осознанное использование инструментов, уважительное отношение к чужим проектам, понимание ограничений и возможностей технологий;

– стимулировать любознательность, внимательность и готовность учиться на ошибках.

3. Развивающие:

– развивать логическое и пространственное мышление, умение формализовать задачу, разбивать её на подзадачи и выстраивать последовательность шагов;

– совершенствовать гибкие навыки: планирование времени, приоритизацию задач, презентацию и аргументацию своей идеи (в том числе на питч-сессиях и защитах);

– расширять творческие способности через создание графики, анимации, игровых механик и нарративов;

– тренировать память, внимание и способность анализировать алгоритмы (в том числе чужие) на предмет ошибок и возможностей улучшения;

– способствовать формированию инженерного подхода: от гипотезы и прототипа к тестированию, исправлению и финальной версии продукта.

1.7. Планируемые результаты освоения программы

Основным результатом обучения является формирование у учащихся 6–10 лет алгоритмической грамотности и базовых цифровых компетенций, позволяющих самостоятельно создавать простые цифровые продукты (игры, мультфильмы, 3D-миры) в визуальных средах программирования, а также готовность к дальнейшему изучению текстовых языков программирования.

В процессе занятий обеспечивается целенаправленная работа на достижение личностных, метапредметных и предметных результатов, соответствующих возрастным возможностям обучающихся и логике программы.

Предметные результаты

По модулю 1 «Юный исследователь» (6–7 лет):

– понимание базовых понятий: «исполнитель», «команда», «алгоритм», «линейный алгоритм», «цикл»;

– умение ориентироваться на координатной сетке (в том числе в формате поля 3×3), определять направления (вверх, вниз, влево, вправо);

– владение базовыми приёмами работы в ПиктоМире: составление последовательности команд для перемещения робота к цели;

– умение работать в Scratch Jr: собирать цепочки блоков, использовать блоки «Начать по щелчку», «Движение», «Повторить N раз»;

– способность реализовывать простые проекты: лабиринт, мультфильм из смены костюмов, викторину с вариантами ответов.

По модулю 2 «Создатель игр» (1–2 класс):

– знание и применение базовых конструкций программирования: циклы, условия, переменные, клоны, события;

– умение использовать сенсоры и логические блоки («если касается…», «повторять пока…») для создания игровой логики;

– владение приёмами проектирования в Scratch: создание меню, счётчиков очков, механики прыжка, генерации препятствий;

– навыки работы в Kodu Game Lab: построение 3D-ландшафта, настройка правил поведения персонажей, управление с клавиатуры, реализация простых игровых механик (сбор предметов, здоровье, финиш);

– способность разрабатывать мини-игры (гонка, сбор монет, платформер-уровень) и защищать проект перед группой.

По модулю 3 «Инженер и архитектор кода» (3–4 класс):

– умение проектировать алгоритмы с использованием функций (собственных блоков), вложенных условий, списков (массивов), генераторов случайных чисел, таймеров;

– владение приёмами отладки и оптимизации кода: пошаговое выполнение, поиск логических ошибок, рефакторинг для сокращения повторяющихся фрагментов;

– навыки разработки в MakeCode Arcade: управление пиксельными спрайтами, обработка столкновений, добавление визуальных эффектов (партиклы), организация многоуровневости;

– понимание принципов игрового баланса и расчёта механик

(скорость, урон, время) через математические зависимости;

– опыт командной работы по ролям (программист, дизайнер, тестировщик);

– готовность к переходу на текстовые языки программирования за счёт освоения универсальных алгоритмических конструкций и инженерных практик.

Метапредметные результаты

– умение самостоятельно ставить учебную задачу и планировать этапы реализации проекта (от идеи и ТЗ до презентации);

– способность соотносить свои действия с планируемым результатом, корректировать алгоритм при обнаружении ошибок (практика отладки);

– навыки тайм-менеджмента: распределение времени на отдельные части проекта, соблюдение сроков;

– умение анализировать алгоритмы (в том числе чужие): выявлять логику, находить ошибки, предлагать улучшения;

– способность классифицировать задачи по типу (движение, условия, счёт, события), устанавливать причинно-следственные связи между блоками кода и поведением персонажа;

– владение навыками учебного сотрудничества: распределение ролей в команде, принятие обратной связи, совместная доработка проекта;

– умение формулировать, аргументировать и презентовать свою идею.

Личностные результаты

– сформированность устойчивого интереса к программированию, алгоритмическому и инженерному мышлению;

– готовность и способность к самообразованию: умение искать информацию, осваивать новые инструменты и применять их в собственных проектах;

– развитие мотивации к творческой и исследовательской деятельности, привычки доводить начатое до результата;

– осознание ценности командной работы, уважительного отношения к чужому труду и конструктивной критики;

– уверенность в своих силах при решении нестандартных задач, готовность воспринимать ошибки как часть процесса обучения и развития;

– ориентация на дальнейшее изучение информатики, робототехники, олимпиадного программирования или смежных ИТ-направлений.

2. ОРГАНИЗАЦИОННО-ПЕДАГОГИЧЕСКИЕ УСЛОВИЯ РЕАЛИЗАЦИИ ПРОГРАММЫ

2.1. Язык реализации программы

Реализация дополнительной общеобразовательной общеразвивающей программы «Программирование для детей» осуществляется на государственном языке Российской Федерации.

2.2. Форма обучения:

– очная.

2.3. Особенности реализации программы

Программа реализуется по модульному принципу: содержание разделено на логически завершённые модули, каждый из которых завершается промежуточным просмотром проектов, а по итогам года проводится итоговая презентация. Такая структура позволяет гибко отслеживать прогресс, своевременно корректировать траекторию обучения и обеспечивать видимый результат на каждом этапе.

2.4. Условия набора и формирования групп

На обучение зачисляются дети в возрасте от 6 до 10 лет, обучающиеся в общеобразовательных организациях Ставропольского края.

Зачисление на обучение по программе осуществляется в соответствии с Правилами приёма обучающихся в ЦЦО «IT-Куб» на 2026–2027 учебный год.

Условия конкурсного отбора гарантируют соблюдение прав обучающихся в области дополнительного образования и предусматривают зачисление детей с учётом возрастных особенностей и уровня готовности к освоению материала: для модуля «Юный исследователь» (6–7 лет) специальная подготовка не требуется; для модулей «Создатель игр» и «Инженер и архитектор кода» преимущество при наличии свободных мест предоставляется учащимся, ранее осваивавшим программы алгоритмики и программирования либо продемонстрировавшим базовые навыки работы в визуальных средах программирования.

Условия формирования групп: разновозрастные группы в рамках каждого уровня сложности; наполняемость группы — не более 12 человек для обеспечения индивидуального подхода и эффективной работы в командах при реализации проектов.

2.5. Формы организации и проведение занятий

Формы организации занятий:

– аудиторные (под непосредственным руководством преподавателя);

– групповые (с распределением ролей в команде: программист, дизайнер, тестировщик);

– индивидуально-групповые (индивидуальная работа над проектом с консультационной поддержкой педагога и взаимным рецензированием в мини-группах).

Формы проведения занятий:

– теоретические;

– практические;
– проектные;
– контрольные;
– игровые.

Формы организации деятельности обучающихся:

– интерактивные проблемные мини-лекции — предполагают вовлечение всех участников через вопросы, разбор ошибок на примерах, демонстрацию слайд-презентации, фрагментов учебных видео или экрана преподавателя;
– мозговой штурм — применяется для генерации идей проекта, формулировки условий задачи, поиска альтернативных решений (например, «как сделать, чтобы враг не застревал в стене»);
– практикум — выполнение типовых и вариативных заданий;
– отладочная сессия — целенаправленный поиск и исправление ошибок в коде (в своём или чужом проекте) по чек-листу;
– рефлексивное обсуждение — разбор результатов: что получилось, какие ошибки встречались, как их исправили, что можно улучшить.

Режим занятий:

– очная форма обучения;
– частота: 1 раз в неделю;
– продолжительность: 2 академических часа с 5-минутным перерывом (для всех модулей), для модуля «Юный исследователь» (6–7 лет) дополнительно предусмотрена динамическая пауза;
– наполняемость группы: не более 12 человек.

Программа реализуется в г. Михайловске на базе ЦЦО «IT-Куб».

УЧЕБНЫЙ ПЛАН

Наименование модуля, учебного курса	Количество часов			
	Теория	Практика	Всего	Форма контроля/ аттестации
Модуль 1. «Юный исследователь»	31	41	72	Проект
Модуль 2. «Создатель игр»	30	42	72	Проект
Модуль 3. «Инженер и архитектор кода»	23	49	72	Проект
Итого:	**84**	**132**	**216**	

КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

Наименование модуля, учебного курса	Год обучения	Дата начала обучения	Дата окончания обучения	Количество учебных недель	Количество учебных дней	Количество учебных часов	Режим занятий
Модуль 1. «Юный исследователь»	1 год обучения	03.09.2026	28.05.2027	36	36	72 ч.	1 урок 1 раза в неделю
Модуль 2. «Создатель игр»	1 год обучения	03.09.2026	28.05.2027	36	36	72 ч.	1 урок 1 раза в неделю
Модуль 3. «Инженер и архитектор кода»	1 год обучения	03.09.2026	28.05.2027	36	36	72 ч.	1 урок 1 раза в неделю

РАБОЧАЯ ПРОГРАММА УЧЕБНОГО КУРСА «ПРОГРАММИРОВАНИЕ ДЛЯ ДЕТЕЙ»

Курс «Программирование для детей» предназначен для обучающихся в возрасте 6–10 лет с дифференциацией по трём модулям сложности: 6–7 лет, 1–2 класс, 3–4 класс.

Курс позволяет освоить основы алгоритмического и логического мышления через создание цифровых продуктов — игр, мультфильмов, 3D-миров. Дети учатся применять математические понятия (координаты, счёт, сравнение величин) в практической деятельности, работать в команде, планировать этапы проекта, не бояться ошибок и доводить продукт до результата. В качестве инструментов используются визуальные среды программирования: ПиктоМир, Scratch Jr, Scratch, Kodu Game Lab, MakeCode Arcade.

МОДУЛЬ 1. «ЮНЫЙ ИССЛЕДОВАТЕЛЬ»

Кейс 1. Знакомство с компьютером и алгоритмами
Кейс 2. Циклы и закономерности
Кейс 3. Творческие проекты
Кейс 4. Защита проектов

В результате освоения учебного курса обучающийся должен:

Знать:

– правила работы с компьютером и технику безопасности при работе в компьютерном классе;

– основные предметные понятия («алгоритм», «исполнитель», «команда», «программа», «событие») и их смысл на доступном уровне;

– что такое линейный алгоритм и почему важен порядок команд;

– назначение базовых блоков в ПиктоМире и Scratch Jr (движение, повторение, запуск по щелчку, смена костюма и др.);

– основы работы с координатной сеткой (на примере поля 3×3): понятия «вверх/вниз», «влево/вправо», «позиция», «цель»;

– как соотносить простое действие (например, «пройти 3 шага») с последовательностью команд;

– базовые принципы создания анимации через смену костюмов и организацию простых событий («отправить сообщение» / «получить сообщение»).

Уметь:

– безопасно и самостоятельно запускать компьютер и нужную программу (ПиктоМир, Scratch Jr), корректно завершать работу;

– составлять и выполнять линейные алгоритмы из 3–5 команд для достижения цели (например, провести робота к флажку);

– использовать команду повторения (цикл) для сокращения последовательности действий и оптимизации маршрута;

– ориентироваться на координатной сетке 3×3 и перемещать исполнителя в заданную точку;

– собирать простые цепочки блоков в Scratch Jr для анимации персонажа и организации взаимодействия объектов;

– применять условия и события для запуска действий (например, при касании — воспроизвести звук, при получении сообщения — начать движение);

– создавать мини-проекты (лабиринт, мультфильм, викторина, «мой город») с использованием изученных конструкций;

– находить и исправлять очевидные ошибки в простом алгоритме (например, неверный порядок команд, отсутствие повтора, неверное направление);

– оформлять проект: подбирать фон, рисовать или выбирать спрайты, добавлять звуковые эффекты;

– презентовать свой проект: рассказать, какая была цель, какие команды использовались и что получилось в результате.

МОДУЛЬ 2. «СОЗДАТЕЛЬ ИГР»

Кейс 1. Основы Scratch
Кейс 2. Введение в 3D
Кейс 3. Сложная логика в Scratch
Кейс 4. Финальные проекты

В результате освоения учебного курса обучающийся должен:

Знать:

– правила работы с компьютером и технику безопасности при работе в компьютерном классе;

– основные предметные понятия («переменная», «клон», «сенсор», «координата», «событие», «условие», «цикл с условием») и их роль в построении игровых механик;

– типы алгоритмических конструкций, используемые в Scratch и Kodu Game Lab: линейные алгоритмы, ветвления, циклы (в том числе «повторять пока…»), сложные условия (с логическими операциями);

– принципы работы с координатной плоскостью в Scratch (диапазон значений по осям X и Y, связь координат с расположением спрайтов и срабатыванием сенсоров);

– основы организации игровых механик: счётчики очков, система жизней, генерация объектов, управление с клавиатуры, обработка столкновений;

– базовые принципы проектирования 3D-миров в Kodu Game Lab: размещение ландшафта, настройка физики, создание правил поведения персонажей;

– способы структурирования кода для читаемости: группировка блоков, использование клонов и переменных для упрощения логики;

– этапы проектной работы: от идеи и ТЗ до тестирования, отладки и презентации.

Уметь:

– самостоятельно запускать и уверенно ориентироваться в интерфейсах Scratch и Kodu Game Lab, переключаться между режимами редактирования (спрайты, фоны, код);

– создавать игровые механики на базе переменных: счётчики очков, уровни здоровья, скорость движения, ограничение времени;

– применять условные конструкции и сенсоры для реализации логики игры (например, «если касается цвета — остановиться», «если здоровье = 0 — переродиться»);

– использовать циклы с условиями и сложные логические связки (И, ИЛИ) для построения нетривиального поведения объектов;

– работать с клонами: генерировать множество однотипных объектов (монеты, враги) и управлять их поведением через единый скрипт;

– проектировать и реализовывать простые 3D-миры в Kodu: формировать ландшафт, расставлять объекты, настраивать правила взаимодействия и управление персонажем;

– программировать траектории движения и физику игровых элементов (прыжки, гравитация, столкновения) в Scratch;

– создавать интерфейс игры: меню, кнопки, смену фонов, систему сообщений между объектами;

– находить и исправлять ошибки в коде (некорректные условия, бесконечные циклы, неправильные координаты, логические противоречия в правилах);

– планировать и реализовывать мини-проекты (гонка, платформер, квест) с распределением ролей в команде (программист, дизайнер, тестировщик);

– оформлять и презентовать проект: готовить демонстрацию игры, объяснять ключевые алгоритмы, отвечать на вопросы по логике и механикам.

МОДУЛЬ 3. «ИНЖЕНЕР И АРХИТЕКТОР КОДА»

Кейс 1. Функции и координаты
Кейс 2. MakeCode Arcade
Кейс 3. Продвинутое проектирование
Кейс 4. Итоговое портфолио

В результате освоения учебного курса обучающийся должен:

Знать:

– правила работы с компьютером и технику безопасности при работе в компьютерном классе;

– расширенный набор предметных понятий («функция/собственный

блок», «список/массив», «генератор случайных чисел», «таймер», «облачные переменные», «игровой баланс», «отладка», «рефакторинг») и их назначение в разработке сложных проектов;

– типы алгоритмических конструкций продвинутого уровня: вложенные условия, многоуровневые циклы, комбинированные логические выражения, механизмы передачи данных между объектами;

– принципы работы с координатной плоскостью и математическими расчётами в играх: вычисление траекторий (в т. ч. параболических), определение расстояний, использование тригонометрических и арифметических зависимостей для управления движением;

– способы организации кода для масштабируемости и читаемости: применение функций для устранения дублирования, структурирование логики по модулям, использование списков для хранения динамических данных (рекорды, имена игроков, параметры врагов);

– особенности разработки в разных средах и переход к текстовым языкам: различия блочного и текстового кода (на примере MakeCode Arcade), базовые принципы работы с событиями и таймерами, обработка столкновений и визуальных эффектов (партиклы);

– основы проектирования ИИ и адаптивной сложности: настройка поведения врагов (патрулирование, преследование), расчёт баланса (скорость, урон, здоровье) с учётом прогресса игрока;

– этапы профессионального цикла разработки ИТ-продукта: от анализа рынка и ТЗ до раскадровки, прототипирования, тестирования, оптимизации и финальной презентации;

– механизмы сохранения и передачи данных в проектах (в т. ч. cloud-переменные в Scratch), принципы работы с файлами и внешними ресурсами.

Уметь:

– уверенно работать в продвинутых средах программирования (продвинутый Scratch, MakeCode Arcade, Kodu Game Lab), свободно переключаться между инструментами и выбирать оптимальную среду под задачу;

– проектировать и создавать собственные функции (блоки) для повторного использования кода, оптимизировать скрипты, устранять дублирование и повышать читаемость программы;

– применять списки (массивы) для хранения и обработки динамических данных: вести таблицу рекордов, управлять пулом объектов, хранить параметры уровней и врагов;

– использовать генераторы случайных чисел и таймеры для создания вариативности и временных ограничений: рандомизировать появление предметов, задавать интервалы спавна врагов, реализовывать отсчёт времени на уровень;

– реализовывать сложные игровые механики: систему жизней и урона, многоуровневость с нарастающей сложностью, адаптивный ИИ врагов,

порталы и переходы между мирами, визуальные эффекты (взрывы, искры, частицы);

– выполнять математические расчёты в коде: определять траектории движения, рассчитывать расстояния между объектами, балансировать параметры (скорость, урон, здоровье) для обеспечения честной и интересной игры;

– проводить отладку и рефакторинг: находить и исправлять логические ошибки, тестировать крайние случаи, оптимизировать производительность, упрощать избыточные конструкции;

– разрабатывать 3D-миры в Kodu с продвинутой логикой: создавать сложные ландшафты, настраивать правила взаимодействия объектов, реализовывать переключение между мирами через порталы, проектировать поведение ИИ;

– работать с графикой и анимацией: рисовать и анимировать спрайты вручную, создавать циклы кадров (например, анимация бега из 4 кадров), подбирать визуальные стили под жанр игры;

– организовывать командную работу по ролям (программист, дизайнер, тестировщик) и выполнять задачи в условиях ограниченного времени;

– готовить полноценное портфолио проекта: составлять ТЗ и раскадровку, описывать ключевые алгоритмы, демонстрировать работу всех механик, аргументировать принятые технические и дизайнерские решения;

– презентовать проект перед аудиторией: структурированно рассказывать об идее, этапах разработки, сложностях и их преодолении, перспективах развития проекта и возможных переходах к текстовым языкам программирования.

УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН ПРОГРАММЫ
«ПРОГРАММИРОВАНИЕ ДЛЯ ДЕТЕЙ»
МОДУЛЬ 1. «ЮНЫЙ ИССЛЕДОВАТЕЛЬ» (6 – 7 лет)

№	Наименование кейса, темы	Количество часов		
		Теория	Практика	Всего
Кейс 1. Знакомство с компьютером и алгоритмами		**8**	**8**	**16**
	Тема 1.1. Вводное занятие. Техника безопасности	1	1	2
	Тема 1.2. Понятие алгоритма. Исполнитель	1	1	2
	Тема 1.3. Линейный алгоритм	1	1	2
	Тема 1.4. ПиктоМир: Работа с полем	1	1	2
	Тема 1.5. Счет в пределах 10. Математика в игре	1	1	2
	Тема 1.6. Геометрические фигуры	1	1	2
	Тема 1.7. Scratch Junior: Интерфейс	1	1	2
	Тема 1.8. Оживляем персонажей	1	1	2
Кейс 2. Циклы и закономерности		**10**	**10**	**20**
	Тема 2.1. Понятие цикла	1	1	2
	Тема 2.2. Ритм и закономерности	1	1	2
	Тема 2.3. Scratch Jr: Циклы	1	1	2
	Тема 2.4. Счет предметов	1	1	2
	Тема 2.5. Проект: "Мой город"	1	1	2
	Тема 2.6. Алгоритмы с повторением	1	1	2
	Тема 2.7. Задачи на внимание	1	1	2
	Тема 2.8. Интерактивные истории	1	1	2
	Тема 2.9. Проект "Мое настроение"	1	1	2
	Тема 2.10. Работа с координатами	1	1	2
Кейс 3. Творческие проекты		**9**	**13**	**22**
	Тема 3.1. Введение в мультипликацию	1	1	2
	Тема 3.2. Логическая игра "Лабиринт"	1	1	2
	Тема 3.3. События	1	1	2
	Тема 3.4. Мультфильм "В лесу"	1	1	2
	Тема 3.5. Математическая игра "Собери урожай"	1	1	2
	Тема 3.6. Решение логических задач	1	1	2

	Тема 3.7. Проект: Викторина	2	2	4
	Тема 3.8. Отрисовка фигур в координатах	1	1	2
	Тема 3.10. Работа над финальным проектом	0	4	4
Кейс 4. Защита проектов		**2**	**12**	**14**
	Тема 4.1. Работа над проектом	0	8	8
	Тема 4.2. Репетиция защиты	2	2	4
	Тема 4.3. Итоговая защита	0	2	2
Итого		**29**	**41**	**72**

МОДУЛЬ 2. «СОЗДАТЕЛЬ ИГР» (1 – 2 класс)

№	Наименование кейса, темы	Количество часов		
		Теория	Практика	Всего
Кейс 1. Основы Scratch		**9**	**9**	**18**
	Тема 1.1. Введение. Интерфейс Scratch	1	1	2
	Тема 1.2. Линейные алгоритмы. Движение	1	1	2
	Тема 1.3. Переменные	1	1	2
	Тема 1.4. Сенсоры и условия (Если - то)	1	1	2
	Тема 1.5. Счетчики в играх. Таблицы	1	1	2
	Тема 1.6. Циклы с условием	1	1	2
	Тема 1.7. Проект "Гонка по линии"	1	1	2
	Тема 1.8. Клоны	1	1	2
	Тема 1.9. Проект "Сбор монет"	1	1	2
Кейс 2. Введение в 3D		**9**	**9**	**18**
	Тема 2.1. Знакомство с Kodu. 3D-ландшафт	1	1	2
	Тема 2.2. Объекты и физика	1	1	2
	Тема 2.3. Программирование в Kodu	1	1	2
	Тема 2.4. Управление с клавиатуры	1	1	2
	Тема 2.5. Сравнение скоростей	1	1	2
	Тема 2.6. Проект "Гонки в 3D"	1	1	2
	Тема 2.7. Условная конструкция в Kodu	1	1	2
	Тема 2.8. Сбор предметов в 3D	1	1	2
	Тема 2.9. Проект "Квест в замке"	1	1	2
Кейс 3. Сложная логика в Scratch		**8**	**8**	**16**
	Тема 3.1. Логические операции (И, ИЛИ)	1	1	2
	Тема 3.2. Траектория движения	1	1	2
	Тема 3.3. Создание меню игры	1	1	2

	Тема 3.4. Платформер. Физика прыжка	1	1	2
	Тема 3.5. Генерация препятствий	1	1	2
	Тема 3.6. Проект "Платформер-уровень"	2	2	4
	Тема 3.7. Работа с графикой	1	1	2
Кейс 4. Защита проектов		**5**	**15**	**20**
	Тема 4.1. Выбор темы. Структура проекта	1	1	2
	Тема 4.2. Работа над индивидуальными проектами	0	6	6
	Тема 4.3. Тестирование и отладка	2	2	4
	Тема 4.4. Предзащита	0	4	4
	Тема 4.5. Итоговая защита	2	2	4
Итого		**31**	**41**	**72**

МОДУЛЬ 3. «ИНЖЕНЕР И АРХИТЕКТОР КОДА» (3 – 4 класс)

№	Наименование кейса, темы	Количество часов		
		Теория	Практика	Всего
Кейс 1. Функции и координаты		**9**	**9**	**18**
	Тема 1.1. Вводный инструктаж	1	1	2
	Тема 1.2. Собственные блоки	1	1	2
	Тема 1.3. Математика в играх. Траектории	1	1	2
	Тема 1.4. Вложенные условия	1	1	2
	Тема 1.5. Списки	1	1	2
	Тема 1.6. Генератор случайных чисел	1	1	2
	Тема 1.7. Проект "Система жизней"	2	2	4
	Тема 1.8. Таймеры	1	1	2
Кейс 2. Пиксельные игры		**5**	**13**	**18**
	Тема 2.1. Интерфейс MakeCode	1	1	2
	Тема 2.2. Управление спрайтом	1	1	2
	Тема 2.3. Физика столкновений	1	1	2
	Тема 2.4. Проект "Змейка"	0	4	4
	Тема 2.5. Визуальные эффекты	1	1	2
	Тема 2.6. Многоуровневость	1	1	2
	Тема 2.7. Проект "Собери 5 уровней"	0	4	4
Кейс 3. Продвинутое проектирование		**6**	**10**	**16**
	Тема 3.1. Игровой баланс	1	1	2

	Тема 3.2. Знакомство с 3D.	1	1	2
	Тема 3.3. Интерфейс Kodu Game Lab	1	1	2
	Тема 3.4. От идеи к игре: строим логику в Kodu по шагам	1	1	2
	Тема 3.5. Создание собственного 3D-мира	1	1	2
	Тема 3.6. Настройка персонажей и объектов	1	1	2
	Тема 3.7. Создание 3D-игр	0	4	4
Кейс 4. Проектная деятельность		**3**	**17**	**20**
	Тема 4.1. Выбор идеи проекта	1	1	2
	Тема 4.2. Планирование проекта	1	1	2
	Тема 4.3. Разработка итогового проекта	0	6	6
	Тема 4.4. Тестирование проекта	1	1	2
	Тема 4.5. Подготовка презентации	0	2	2
	Тема 4.4. Предзащита	0	2	2
	Тема 4.5. Итоговая защита	0	4	4
Итого		**23**	**49**	**72**

СОДЕРЖАНИЕ ПРОГРАММЫ

«ПРОГРАММИРОВАНИЕ ДЛЯ ДЕТЕЙ»

МОДУЛЬ 1. «ЮНЫЙ ИССЛЕДОВАТЕЛЬ» (6–7 лет) (72 часа)

Инструменты: ПиктоМир, Scratch Jr.

Цель модуля: формирование алгоритмического мышления через игру, знакомство с основами математики и логики, освоение базовых приёмов работы в визуальных средах программирования.

Кейс 1. «Знакомство с компьютером и алгоритмами» (16 часов)

Данный кейс предназначен для знакомства с правилами работы за компьютером, базовым интерфейсом цифровых сред и понятием алгоритма как последовательности команд. Учащиеся учатся воспринимать исполнителя (робота/персонажа) как объект, выполняющий чёткие инструкции.

Учащиеся должны знать:

– правила работы с компьютером и технику безопасности в компьютерном классе;

– основные понятия: «алгоритм», «исполнитель», «команда», «программа»;

– что такое линейный алгоритм и почему важен порядок команд;

– назначение базовых блоков в ПиктоМире и Scratch Jr (движение, запуск по щелчку, смена костюма и др.);

– основы работы с координатной сеткой на примере поля 3×3: понятия «вверх/вниз», «влево/вправо», «позиция», «цель».

Учащиеся должны уметь:

– безопасно и самостоятельно запускать компьютер и нужную программу (ПиктоМир, Scratch Jr), корректно завершать работу;

– составлять и выполнять линейные алгоритмы из 3–5 команд для достижения цели (например, провести робота к флажку);

– ориентироваться на координатной сетке 3×3 и перемещать исполнителя в заданную точку;

– собирать простые цепочки блоков в Scratch Jr для анимации персонажа и организации взаимодействия объектов;

– находить и исправлять очевидные ошибки в простом алгоритме (неверный порядок команд, неверное направление).

Формы занятий, используемые при изучении данного кейса:

– лекционная (мини-объяснение с демонстрацией);

– индивидуальная работа за ПК;

– парная работа (взаимная проверка алгоритмов);

– игровая практика.

Тема 1.1. Вводное занятие. Техника безопасности (2 часа)

Теория. Правила поведения в компьютерном классе, гигиена работы за экраном, безопасное обращение с техникой.

Практика. Ознакомление с рабочим местом, запуск и корректное завершение работы программ.

Тема 1.2. Понятие алгоритма. Исполнитель (2 часа)

Теория. Что такое алгоритм, кто такой исполнитель, почему порядок команд важен.

Практика. Составление цепочек команд «Вперёд», «Назад», «Влево», «Вправо» для перемещения робота к цели.

Тема 1.3. Линейный алгоритм (2 часа)

Теория. Линейная последовательность действий, понятие «шаг алгоритма».

Практика. Создание маршрутов из 3–5 команд, сравнение длины путей (какой короче).

Тема 1.4. ПиктоМир: работа с полем (2 часа)

Теория. Сетка координат, ориентация в пространстве (верх, низ, лево, право).

Практика. Перемещение робота по полю 3×3, достижение заданных точек.

Тема 1.5. Счёт в пределах 10. Математика в игре (2 часа)

Теория. Связь шагов и чисел, сравнение величин (>, <, =).

Практика. Подсчёт шагов до цели, выбор кратчайшего маршрута.

Тема 1.6. Геометрические фигуры (2 часа)

Теория. Простые фигуры (квадрат, треугольник) как траектории движения.

Практика. Построение маршрута в форме квадрата, соотнесение фигуры и пути.

Тема 1.7. Scratch Jr: интерфейс (2 часа)

Теория. Сцена, персонажи (спрайты), инструмент «Перо».

Практика. Знакомство с рабочей областью, размещение объектов, выбор фона.

Тема 1.8. Оживляем персонажей (2 часа)

Теория. Блоки «Начать по щелчку» и «Движение».

Практика. Создание простой анимации: заставить кота пройти дорожку.

Кейс 2. Циклы и закономерности (20 часов)

Кейс направлен на освоение понятия цикла как способа сократить повторяющиеся действия, а также на развитие умения видеть и продолжать закономерности. Учащиеся учатся оптимизировать алгоритмы и применять циклы для рисования орнаментов и сбора объектов.

Учащиеся должны знать:

- понятие цикла и его назначение;
- как цикл экономит команды и делает алгоритм компактнее;
- принципы поиска закономерностей в последовательностях;
- способы использования блока «Повторить N раз» в Scratch Jr.

Учащиеся должны уметь:

- использовать команду повторения (цикл) для сокращения последовательности действий;
- находить пропущенный элемент в ряду и продолжать последовательность;
- рисовать простые фигуры и орнаменты с помощью циклов;
- создавать сцены с заданным количеством объектов и сравнивать их («больше/меньше»);
- оптимизировать маршрут через повтор (сбор всех монет на поле в ПиктоМире).

Формы занятий:

- мини-лекция с примерами;
- индивидуальная практика;
- парное программирование;
- проектная работа в малых группах.

Тема 2.1. Понятие цикла (2 часа)

Теория. Зачем нужны циклы, как они упрощают алгоритмы.

Практика. Использование команды «Повторить» в ПиктоМире.

Тема 2.2. Ритм и закономерности (2 часа)

Теория. Чередование фигур, поиск пропущенного элемента.

Практика. Продолжение рядов, составление собственных последовательностей.

Тема 2.3. Scratch Jr: циклы (2 часа)

Теория. Блок «Повторить N раз», варианты использования.

Практика. Рисование звезды или домика с помощью циклов.

Тема 2.4. Счёт предметов (2 часа)

Теория. Добавление объектов на сцену, сравнение количества.

Практика. Создание сцены с заданным числом фруктов, сравнение «больше/меньше».

Тема 2.5. Проект «Мой город» (2 часа)

Теория. Планирование траектории движения машинок, применение циклов.

Практика. Проектирование дорог и маршрутов для машинок.

Тема 2.6. Алгоритмы с повторением (2 часа)

Теория. Оптимизация маршрута через повтор.

Практика. Сбор всех монет на поле с использованием циклов.

Тема 2.7. Задачи на внимание (2 часа)

Теория. Понятие ошибки в алгоритме, способы её обнаружения.

Практика. Поиск и исправление ошибок в готовых цепочках команд.

Тема 2.8. Интерактивные истории (2 часа)

Теория. Диалоги персонажей, синхронизация действий.

Практика. Создание анимированной истории с двумя персонажами.

Тема 2.9. Проект «Моё настроение» (2 часа)

Теория. Смена фонов и костюмов персонажа в зависимости от событий.

Практика. Реализация смены внешности спрайта по сценарию.

Тема 2.10. Работа с координатами (2 часа)

Теория. Оси координат, перемещение в заданную точку (X и Y).

Практика. Точное позиционирование персонажа на сцене.

Кейс 3. Творческие проекты (22 часа)

Кейс ориентирован на применение изученных конструкций в творческих задачах: создание мультфильмов, логических игр и викторин. Учащиеся осваивают события («отправить/получить сообщение»), симметрию и базовые приёмы анимации.

Учащиеся должны знать:

– принцип работы событий: «отправить сообщение» и «получить сообщение»;

– основы анимации через смену костюмов;

– понятие симметрии и зеркального отражения;

– этапы создания мини-проекта: идея, план, реализация, презентация.

Учащиеся должны уметь:

– запускать действие одного объекта сигналом от другого;

– создавать анимацию движения через смену костюмов;

– строить игровые уровни (лабиринты) и управлять объектами стрелками;

– реализовывать простые викторины с вариантами ответов;

– дорисовывать вторую половину рисунка по клеткам, соблюдая симметрию;

– презентовать свой проект: рассказать идею, показать ключевые блоки, объяснить логику.

Формы занятий:

– инструктаж и планирование;

– самостоятельная работа над проектом;

– взаимное рецензирование;

– подготовка и проведение презентаций.

Тема 3.1. Введение в мультипликацию (2 часа)

Теория. Смена кадров для создания иллюзии движения.

Практика. Анимация «Летящая птица» через последовательную смену костюмов.

Тема 3.2. Логическая игра «Лабиринт» (2 часа)

Теория. Структура лабиринта, управление стрелками.

Практика. Создание игрового уровня и программирование движения.

Тема 3.3. События (2 часа)

Теория. Взаимодействие объектов через сообщения.

Практика. Запуск действия одного персонажа сигналом от другого.

Тема 3.4. Мультфильм «В лесу» (2 часа)

Теория. Композиция сцены, использование циклов для анимации облаков.

Практика. Создание сцены с деревьями, животными и движущимися облаками.

Тема 3.5. Математическая игра «Собери урожай» (2 часа)

Теория. Соотнесение действия и количества, простой счёт.

Практика. Персонаж собирает яблоки, подсчёт очков.

Тема 3.6. Решение логических задач (2 часа)

Теория. Задачи на переливания и переправы (без ПК).

Практика. Обсуждение и поиск решений, перевод логики в алгоритм.

Тема 3.7. Проект: викторина (4 часа)

Теория. Условные алгоритмы для выбора ответа.

Практика. Игра «Вопрос-ответ» с вариантами и проверкой правильности.

Тема 3.8. Отрисовка фигур в координатах (2 часа)

Теория. Симметрия, зеркальное отражение.

Практика. Дорисовать вторую половину рисунка по клеткам.

Тема 3.10. Работа над финальным проектом (4 часа)

Практика. Создание игры или мультфильма на свободную тему.

Кейс 4. Защита проектов (14 часов)

Кейс завершает модуль и направлен на отладку, оформление и публичную презентацию результатов. Учащиеся учатся тестировать проекты, улучшать дизайн и уверенно рассказывать о своей работе.

Учащиеся должны знать:

– этапы подготовки проекта к защите: тестирование, отладка, оформление;

– критерии качественной презентации: ясность, наглядность, структура рассказа;

– приёмы работы с аудиторией (ответы на вопросы, объяснение логики).

Учащиеся должны уметь:

– тестировать проект и исправлять ошибки в логике;

– подбирать фоны, костюмы, звуки для улучшения восприятия;

– структурировать рассказ о проекте: цель, идея, алгоритмы, сложности и решения;

– отвечать на вопросы и учитывать обратную связь;

– применять знания в нестандартных ситуациях (командные игры на алгоритмику).

Формы занятий:

– индивидуальные консультации;

– репетиции защиты;

– публичная демонстрация проектов;

– рефлексия и подведение итогов.

Тема 4.1. Работа над проектом (8 часов)

Практика. Тестирование, исправление логических недочётов

финального проекта.

Тема 4.2. Репетиция защиты (4 часа)

Теория. Структура выступления, работа с вопросами.

Практика. Пробные презентации перед группой.

Тема 4.3. Итоговая защита (2 часа)

Практика. Демонстрация проектов родителям и педагогам, ответы на вопросы.

МОДУЛЬ 2. «СОЗДАТЕЛЬ ИГР» (1–2 класс) (72 часа)

Инструменты: Scratch, Kodu Game Lab.

Цель модуля: освоение базовых конструкций программирования (циклы, условия, переменные) через разработку игр и 3D-миров; формирование навыков проектирования игровых механик, работы с сенсорами и параметрами объектов, развитие системного подхода к созданию цифрового продукта.

Кейс 1. Основы Scratch (18 часов)

Кейс предназначен для перехода от простейших алгоритмов к полноценным игровым механикам. Учащиеся осваивают интерфейс Scratch, учатся работать с координатной сеткой, переменными и сенсорами, реализуют первые игровые проекты с подсчётом очков и условиями победы.

Учащиеся должны знать:

– структуру интерфейса Scratch: спрайты, сцены, координатная сетка (от −240 до 240), категории блоков;

– понятие переменной как «коробочки для хранения значений» и способы её изменения;

– принцип работы сенсоров и условных конструкций («Если… то…») для реакции на события;

– назначение циклов с условием («Повторять пока…») и их отличие от простых повторений;

– механизм клонирования объектов и особенности поведения клонов;

– базовые игровые механики: счётчик очков, таблица рекордов, ускорение/замедление персонажа.

Учащиеся должны уметь:

– самостоятельно запускать и уверенно ориентироваться в интерфейсе Scratch, переключаться между режимами редактирования (спрайты, фоны, код);

– создавать игровые механики на базе переменных: счётчики очков, уровни здоровья, скорость движения, ограничение времени;

– применять условные конструкции и сенсоры для реализации логики игры (например, «если касается цвета — остановиться», «если здоровье = 0 — переродиться»);

– использовать циклы с условиями и простые логические связки (И, ИЛИ) для построения нетривиального поведения объектов;

– работать с клонами: генерировать множество однотипных объектов (монеты, враги) и управлять их поведением через единый скрипт;

– программировать траектории движения и физику игровых элементов (прыжки, гравитация, столкновения) в Scratch;

– создавать интерфейс игры: меню, кнопки, смену фонов, систему сообщений между объектами;

– находить и исправлять ошибки в коде (некорректные условия, бесконечные циклы, неправильные координаты).

Формы занятий, используемые при изучении данного кейса:

– мини-лекция с демонстрацией;
– индивидуальная работа за ПК;
– парное программирование;
– командная разработка мини-игры;
– защита проектов.

Тема 1.1. Введение. Интерфейс Scratch (2 часа)

Теория. Отличие от Scratch Jr. Спрайты, сцены, координатная сетка (−240…240).

Практика. Знакомство с рабочей областью, размещение объектов, выбор фона, работа со слоями.

Тема 1.2. Линейные алгоритмы. Движение (2 часа)

Теория. Блоки движения, изменение координат, плавное движение.

Практика. Создание маршрута персонажа, достижение заданных точек, обход препятствий.

Тема 1.3. Переменные (2 часа)

Теория. Понятие переменной как «коробочки для чисел». Создание и изменение значений.

Практика. Реализация счётчика очков, таймера, шкалы здоровья.

Тема 1.4. Сенсоры и условия («Если-то») (2 часа)

Теория. Работа с сенсорами: «касается цвета», «на краю сцены», «расстояние до объекта».

Практика. Программирование реакции на события: остановка при касании стены, переход на следующий уровень.

Тема 1.5. Счётчики в играх. Таблицы (2 часа)

Теория. Отображение результатов, сравнение рекордов, визуализация данных.

Практика. Вывод счёта на сцену, сохранение лучшего результата, анимация цифр.

Тема 1.6. Циклы с условием (2 часа)

Теория. Конструкция «повторять пока…», условия остановки, бесконечный цикл.

Практика. Движение персонажа до достижения цели, сбор всех монет на уровне.

Тема 1.7. Проект «Гонка по линии» (2 часа)

Теория. Механика гонок: ускорение, препятствия, финиш.

Практика. Игра, где машина едет по трассе, ускоряется (переменная скорости), объезжает препятствия.

Тема 1.8. Клоны (2 часа)

Теория. Создание множества объектов из одного шаблона, управление через один скрипт.

Практика. Генерация врагов или бонусов, реализация механики «спавна» и исчезновения.

Тема 1.9. Проект «Сбор монет» (2 часа)

Теория. Логика сбора предметов, подсчёт очков, условия завершения уровня.

Практика. Разработка мини-игры: персонаж собирает монеты, избегает врагов, набирает очки.

Кейс 2. Введение в 3D (18 часов)

Кейс направлен на освоение 3D-среды и проектирование игровых миров с правилами поведения персонажей. Учащиеся учатся создавать ландшафт, настраивать физику и логику взаимодействия объектов, реализуют простые игровые сценарии.

Учащиеся должны знать:

– базовые принципы проектирования 3D-миров: размещение ландшафта, настройка физики, создание правил поведения персонажей;

– способы структурирования кода для читаемости: группировка блоков, использование клонов и переменных для упрощения логики;

– этапы проектной работы: от идеи и ТЗ до тестирования, отладки и презентации.

Учащиеся должны уметь:

– проектировать и реализовывать простые 3D-миры в Kodu: формировать ландшафт, расставлять объекты, настраивать правила взаимодействия и управление персонажем;

– задавать поведение персонажей через правила «когда — делай»: реакция на события, столкновения, сбор предметов;

– балансировать игровые механики: регулировать скорость, здоровье, урон, количество ресурсов;

– тестировать и отлаживать мир: проверять работоспособность правил, устранять логические ошибки;

– презентовать проект: объяснять логику правил, демонстрировать ключевые механики, отвечать на вопросы.

Формы занятий:

– инструктаж и планирование;

– самостоятельная работа в Kodu;

– взаимное рецензирование;

– демонстрация и защита проектов.

Тема 2.1. Знакомство с Kodu. 3D-ландшафт (2 часа)

Теория. Интерфейс, инструменты создания ландшафта, объекты и персонажи.

Практика. Освоение базовых инструментов: рисование земли, добавление воды, размещение деревьев.

Тема 2.2. Объекты и физика (2 часа)

Теория. Типы объектов, их свойства и поведение. Настройка гравитации, скорости, трения.

Практика. Размещение персонажей, проверка движения по склонам, платформам, воде.

Тема 2.3. Программирование в Kodu (2 часа)

Теория. Правила «когда — делай», логика условий, приоритеты действий.

Практика. Настройка реакции персонажа на события: сбор монет, столкновение с врагом, достижение финиша.

Тема 2.4. Управление с клавиатуры (2 часа)

Теория. Привязка клавиш к действиям, настройка чувствительности.

Практика. Реализация управления персонажем, тестирование удобства.

Тема 2.5. Сравнение скоростей (2 часа)

Теория. Влияние скорости на геймплей, баланс сложности.

Практика. Настройка разных скоростей для персонажей и объектов, сравнение ощущений.

Тема 2.6. Проект «Гонки в 3D» (2 часа)

Теория. Структура гоночного трека, механика обгона, финишная линия.

Практика. Построение трассы, расстановка препятствий, настройка правил победы.

Тема 2.7. Условная конструкция в Kodu (2 часа)

Теория. Вложенные условия, сложные правила, исключения.

Практика. Добавление условий: «если скорость высокая — тормозить», «если враг рядом — прятаться».

Тема 2.8. Сбор предметов в 3D (2 часа)

Теория. Механика сбора, начисление очков, визуальная обратная связь.

Практика. Сбор монет, кристаллов, бонусов; отображение счёта.

Тема 2.9. Проект «Квест в замке» (2 часа)

Теория. Структура квеста: цели, награды, логика дверей и ловушек.

Практика. Создание замка, расстановка предметов, настройка правил прохождения.

Кейс 3. Сложная логика в Scratch (16 часов)

Кейс ориентирован на углубление навыков работы в Scratch: учащиеся осваивают логические операции, физику прыжка, генерацию препятствий и создание меню. Реализуются проекты с более сложной механикой и улучшенной графикой.

Учащиеся должны знать:

– логические операции (И, ИЛИ, НЕ) и их применение в условиях;
– принципы реализации физики движения: прыжки, гравитация, инерция;
– методы генерации динамических объектов (препятствий, врагов);
– приёмы работы с графикой: костюмы, слои, эффекты.

Учащиеся должны уметь:

– комбинировать несколько условий с помощью логических операций для точной логики;
– реализовывать физику прыжка через изменение координат и циклов;
– генерировать препятствия случайным образом или по шаблону;
– создавать меню игры: кнопки, переходы между сценами, выбор уровней;
– оптимизировать код для стабильной работы при большом количестве объектов.

Формы занятий:

– лекция с разбором примеров;
– индивидуальные и парные задания;
– проектная работа в командах;
– защита и рецензирование проектов.

Тема 3.1. Логические операции (И, ИЛИ) (2 часа)

Теория. Смысл операций, примеры в играх: «если здоровье > 0 И скорость высокая».

Практика. Реализация сложных условий: двойной прыжок при зажатом Shift, ускоренный бег.

Тема 3.2. Траектория движения (2 часа)

Теория. Расчёт траектории, углы, направления, координатные формулы.

Практика. Движение по дуге, полёт снаряда, обход углов.

Тема 3.3. Создание меню игры (2 часа)

Теория. Структура меню: кнопки, состояния, переходы.

Практика. Меню «Старт», «Выбор уровня», «Инструкция», «Выход».

Тема 3.4. Платформер. Физика прыжка (2 часа)

Теория. Гравитация, ускорение, инерция, расчёт высоты прыжка.

Практика. Реализация прыжка: изменение Y-координаты, проверка пола, анимация.

Тема 3.5. Генерация препятствий (2 часа)

Теория. Случайная генерация, шаблоны уровней, таймеры появления.

Практика. Появление шипов, платформ, врагов по расписанию или случайно.

Тема 3.6. Проект «Платформер-уровень» (4 часа)

Теория. Структура уровня: старт, финиш, ловушки, бонусы.

Практика. Сборка уровня, балансировка сложности, тестирование.

Тема 3.7. Работа с графикой (2 часа)

Теория. Костюмы, слои, прозрачность, эффекты (тень, свечение).

Практика. Рисование костюмов, анимация бега и прыжка, настройка слоёв.

Кейс 4. Защита проектов (20 часов)

Кейс завершает модуль и направлен на самостоятельную разработку, тестирование и публичную презентацию итогового проекта. Учащиеся применяют все освоенные навыки, учатся планировать время, распределять задачи и грамотно представлять результаты.

Учащиеся должны знать:

– этапы подготовки проекта к защите: планирование, реализация, тестирование, доработка, презентация;

– критерии качественной презентации: ясность, наглядность, структура рассказа;

– приёмы работы с аудиторией: ответы на вопросы, объяснение логики, аргументация решений.

Учащиеся должны уметь:

– планировать индивидуальный или командный проект: формулировать идею, разбивать на подзадачи, оценивать сроки;

– реализовывать проект с использованием всех изученных механик (переменные, условия, циклы, клоны, физика);

– тестировать и отлаживать проект: находить и устранять ошибки, балансировать сложность, улучшать интерфейс;

– оформлять проект: подбирать фоны, костюмы, звуки, добавлять подсказки и меню;

– уверенно презентовать проект: рассказывать о целях, идеях, алгоритмах, отвечать на вопросы жюри и одноклассников.

Формы занятий:

– индивидуальные консультации;
– работа над проектом (индивидуально или в команде);
– предзащита и получение обратной связи;
– публичная демонстрация проектов;
– рефлексия и подведение итогов.

Тема 4.1. Выбор темы. Структура проекта (2 часа)

Теория. Критерии хорошей идеи, структура проекта: цель, механики, интерфейс.

Практика. Формулирование идеи, составление плана, распределение ролей (если команда).

Тема 4.2. Работа над индивидуальными проектами (6 часов)

Практика. Самостоятельная реализация проекта: учащиеся применяют изученные конструкции для построения игровой механики; педагог проводит консультации и помогает в решении возникающих задач.

Тема 4.3. Тестирование и отладка (4 часа)

Теория. Виды ошибок, методы поиска и исправления.

Практика. Проверка всех механик, устранение багов, балансировка.

Тема 4.4. Предзащита (4 часа)

Практика. Демонстрация проекта, получение обратной связи от педагога и одноклассников, внесение правок.

Тема 4.5. Итоговая защита (4 часа)

Теория. Структура выступления, работа с вопросами, аргументация технических решений.

Практика. Публичная защита проекта, демонстрация работы, ответы на вопросы жюри.

МОДУЛЬ 3. «ИНЖЕНЕР И АРХИТЕКТОР КОДА» (3–4 класс) (72 часа)

Инструменты: продвинутый Scratch, MakeCode Arcade, Kodu Game Lab.

Цель модуля: развитие системного мышления, освоение функций,

отладки, работы с координатной плоскостью и подготовка к переходу на текстовые языки программирования.

Кейс 1. Функции и координаты (18 часов)

Кейс направлен на освоение модульного подхода к программированию: учащиеся учатся создавать собственные блоки (функции), работать с математикой в играх, использовать списки, генераторы случайных чисел и таймеры. Отдельное внимание уделяется отладке и построению игровых механик на базе точных расчётов.

Учащиеся должны знать:

– понятие собственного блока, параметры и область применения;
– основы математики в играх: координаты, траектории, расстояния, углы;
– принципы работы вложенных условий и их влияние на логику программы;
– устройство и применение списков для хранения игровых данных;
– работу генератора случайных чисел и способы контроля вариативности;
– методы отладки: поиск ошибок, тестирование крайних случаев, упрощение логики;
– назначение и реализацию таймеров и временных ограничений в игровых механиках.

Учащиеся должны уметь:

– создавать и использовать собственные блоки для структурирования кода и устранения дублирования;
– рассчитывать траектории движения и позиционировать объекты по координатам;
– применять вложенные условия для реализации сложной логики поведения персонажей;
– использовать списки для хранения и обработки динамических данных (счёт, инвентарь, уровни);
– управлять случайностью: генерировать объекты, уровни, события с контролируемой вероятностью;
– находить и исправлять ошибки в коде, оптимизировать производительность;
– реализовывать таймеры, отсчёт времени, временные эффекты и механики с ограничениями по времени.

Формы занятий:

– мини-лекция с демонстрацией примеров;
– индивидуальная работа за ПК;
– парное программирование для обмена приёмами отладки;
– проектная работа с поэтапной сдачей механик;

– защита и рецензирование проектов.

Тема 1.1. Вводный инструктаж (2 часа)

Теория. Техника безопасности, правила работы в компьютерном классе. Повторение базовых конструкций: переменные, условия, циклы, сенсоры. Разбор типичных ошибок в простых проектах.

Практика. Восстановление и доработка старого проекта: проверка логики, исправление багов, оптимизация кода.

Тема 1.2. Собственные блоки (2 часа)

Теория. Понятие функции (собственного блока) в Scratch. Преимущества выноса повторяющихся действий в отдельные блоки.

Практика. Создание функций «прыжок», «стрельба», «сбор предмета». Перенос повторяющихся фрагментов кода в собственные блоки, проверка работы на разных сценариях.

Тема 1.3. Математика в играх. Траектории (2 часа)

Теория. Координаты в Scratch, расчёт траекторий, углы и направления движения. Простые формулы для позиционирования и перемещения объектов.

Практика. Реализация движения по дуге, полёта снаряда, прицеливания и «радиуса обзора». Настройка точности попадания и визуальной обратной связи.

Тема 1.4. Вложенные условия (2 часа)

Теория. Вложенные конструкции «если-то», приоритет проверки условий, логические связки. Примеры из игр: проверка здоровья и скорости.

Практика. Программирование сложной логики: «если здоровье > 0 и скорость высокая и враг рядом — атаковать», «если игрок вне зоны и время вышло — завершить уровень».

Тема 1.5. Списки (2 часа)

Теория. Списки как хранилище данных. Добавление, удаление, поиск элементов. Индексация и перебор. Применение для инвентаря, таблицы рекордов.

Практика. Ведение таблицы рекордов, управление пулом врагов, хранение параметров уровней. Реализация простейшего инвентаря персонажа.

Тема 1.6. Генератор случайных чисел (2 часа)

Теория. Принцип работы генератора случайных чисел, области применения, контроль вариативности. Настройка «весов» для разной вероятности появления объектов.

Практика. Рандомизация появления предметов, генерация уровней, случайный выбор противника. Балансировка редкости бонусов.

Тема 1.7. Проект «Система жизней» (4 часа)

Теория: Механика жизней: как сделать игру живой.

Практика. Разработка механики жизней и урона: подсчёт жизней, обработка столкновений, визуальные и звуковые эффекты. Реализация экрана проигрыша и перезапуска уровня.

Тема 1.8. Таймеры (2 часа)

Теория. Таймеры и временные ограничения в играх. Отсчёт времени, задержки, «перезарядка» способностей.

Практика. Ограничение времени на уровень, отсчёт до события, задержка действий. Реализация механики «щита» или «ускорения» с таймером.

Кейс 2. Пиксельные игры (18 часов)

Кейс ориентирован на освоение MakeCode Arcade и создание пиксельных игр: учащиеся учатся управлять спрайтами, обрабатывать столкновения, создавать визуальные эффекты, реализовывать многоуровневость и применять функции для структурирования кода.

Учащиеся должны знать:

– интерфейс MakeCode Arcade, переключение между блоками и текстом;
– управление спрайтом: перемещение, скорость, границы сцены;
– физику столкновений и способы обработки взаимодействий;
– принципы создания многоуровневых игр и переключения между уровнями;
– применение функций для повторного использования кода и упрощения логики;
– способы создания визуальных эффектов (партиклов) и их влияние на восприятие игры.

Учащиеся должны уметь:

– уверенно работать в MakeCode Arcade, свободно переключаться между блоками и текстовым режимом;
– программировать движение и управление персонажем с учётом границ сцены и физики;
– обрабатывать столкновения и реализовывать реакции на них (урон, сбор предметов, переход на следующий уровень);
– создавать несколько уровней, настраивать уникальные механики для каждого;
– структурировать код с помощью функций, уменьшать дублирование и повышать читаемость;
– добавлять визуальные эффекты (искры, взрывы, частицы) для улучшения геймплея.

Формы занятий:

– демонстрационные уроки с разбором механик;

– индивидуальные и командные проекты;

– мини-проекты для отработки навыков в условиях ограниченного времени;

– защита проектов и взаимное рецензирование.

Тема 2.1. Интерфейс MakeCode (2 часа)

Теория. Обзор интерфейса MakeCode Arcade: панель инструментов, режимы работы, консоль отладки. Сравнение с Scratch: сходства и отличия.

Практика. Запуск первого проекта, изучение базовых блоков и их текстовых аналогов. Написание простого скрипта в обоих режимах.

Тема 2.2. Управление спрайтом (2 часа)

Теория. Управление движением, настройка скорости, обработка нажатий клавиш. Работа с границами сцены и предотвращение выхода за пределы.

Практика. Реализация плавного управления персонажем, настройка чувствительности, добавление визуальной обратной связи при движении.

Тема 2.3. Физика столкновений (2 часа)

Теория. Обработка столкновений, определение типов взаимодействий (сбор, урон, переход). Настройка реакции на разные типы объектов.

Практика. Программирование столкновений: сбор монет, получение урона, открытие дверей. Тестирование на разных скоростях и траекториях.

Тема 2.4. Проект «Змейка» (4 часа)

Практика. Разработка классической игры «Змейка»: движение, рост, столкновения со стенами и собственным хвостом. Реализация подсчёта очков и экрана проигрыша.

Тема 2.5. Визуальные эффекты (2 часа)

Теория. Работа с частицами (партиклами): искры, взрывы, следы. Влияние эффектов на восприятие и производительность.

Практика. Добавление взрывов, искр, следов движения. Настройка длительности, цвета и количества частиц.

Тема 2.6. Многоуровневость (2 часа)

Теория. Создание нескольких уровней, переключение между ними, сохранение прогресса. Настройка уникальных механик для каждого уровня.

Практика. Реализация 3–5 уровней с разной сложностью, врагами и препятствиями. Настройка переходов и экранов загрузки.

Тема 2.7. Проект «Собери 5 уровней» (4 часа)

Практика. Сборка полноценного проекта из 5 уровней: настройка механик, балансировка сложности, добавление визуальных и звуковых эффектов. Финальное тестирование и устранение ошибок.

Кейс 3. Продвинутое проектирование (16 часов)

Кейс посвящён углублённому проектированию игровых продуктов.

Учащиеся переходят от отдельных механик к комплексному проектированию 3D-игр в Kodu Game Lab: учатся выстраивать логику по шагам, создавать собственные миры, настраивать персонажей и объекты, а также проектировать полноценные 3D-игры.

Учащиеся должны знать:

– принципы игрового баланса: сложность, награды, прогрессия, обратная связь;
– основы 3D-пространства и навигации в Kodu;
– интерфейс Kodu и инструменты для создания ландшафта, объектов и правил;
– пошаговый подход к проектированию логики игры: от идеи к прототипу;
– методы создания собственного 3D-мира: ландшафт, освещение, детализация;
– настройку персонажей и объектов: поведение, скорость, взаимодействие;
– особенности разработки 3D-игр: камера, масштаб, физика, оптимизация.

Учащиеся должны уметь:

– настраивать баланс сложности: регулировать здоровье, урон, скорость, частоту появления врагов;
– анализировать и настраивать игровой баланс;
– ориентироваться в 3D-пространстве и управлять камерой;
– работать с интерфейсом Kodu и использовать инструменты для создания мира;
– выстраивать логику игры по шагам и тестировать каждую часть;
– создавать собственный 3D-мир с ландшафтом и деталями;
– настраивать персонажей и объекты, задавать правила поведения;
– разрабатывать полноценные 3D-игры и презентовать их.

Формы занятий:

– лекции с разбором игровых примеров;
– практические занятия по созданию игровых миров;
– работа с графическими редакторами и инструментами Kodu;
– командная работа и планирование.

Тема 3.1. Игровой баланс (2 часа)

Теория. Принципы баланса: сложность, награды, темп, обратная связь. Примеры из известных игр.

Практика. Настройка параметров игры: здоровье, урон, скорость врагов. Тестирование баланса на фокус-группе.

Тема 3.2. Знакомство с 3D (2 часа)

Теория. Основы 3D-пространства; навигация и управление камерой; масштаб и пропорции; восприятие глубины.

Практика. Освоение управления камерой; перемещение по 3D-миру; оценка масштаба объектов; эксперименты с ракурсами.

Тема 3.3. Интерфейс Kodu Game Lab (2 часа)

Теория. Обзор интерфейса Kodu; инструменты для создания ландшафта, объектов, правил; организация проекта; горячие клавиши.

Практика. Создание тестового проекта; освоение инструментов; настройка ландшафта и объектов; сохранение проекта.

Тема 3.4. От идеи к игре: строим логику в Kodu по шагам (2 часа)

Теория. Пошаговый подход к проектированию; декомпозиция идеи на механики; прототипирование и тестирование; итерации.

Практика. Формулирование идеи игры; разбиение на механики; создание прототипа; тестирование и корректировка.

Тема 3.5. Создание собственного 3D-мира (2 часа)

Теория. Методы создания ландшафта: горы, вода, равнины; детализация и текстуры; освещение и атмосфера.

Практика. Построение собственного 3D-мира; добавление деталей; настройка освещения; эксперименты с текстурами.

Тема 3.6. Настройка персонажей и объектов (2 часа)

Теория. Настройка параметров персонажей: скорость, здоровье, поведение; взаимодействие объектов; правила и приоритеты.

Практика. Создание и настройка персонажей; задание правил поведения; тестирование взаимодействия; балансировка параметров.

Тема 3.7. Создание 3D-игр (4 часа)

Теория. Архитектура 3D-игры: уровни, цели, награды; интеграция механик; критерии качества.

Практика. Разработка полноценной 3D-игры; сборка механик; тестирование и балансировка; подготовка к презентации.

Кейс 4. Проектная деятельность (20 часов)

Кейс завершает модуль и направлен на организацию самостоятельной работы над индивидуальным проектом, обеспечить качественную отладку, подготовку презентации и публичную защиту. Учащиеся учатся планировать работу, распределять задачи, тестировать и улучшать продукт, а также грамотно презентовать его аудитории.

Учащиеся должны знать:

– тапы работы над проектом: выбор идеи, планирование, реализация, тестирование, доработка, презентация;

– структуру технического задания и критерии хорошего игрового

проекта;

– методы тестирования и виды ошибок (логические, интерфейсные, балансовые), способы их поиска и устранения;

– правила подготовки выступления: структура рассказа, работа с аудиторией, ответы на вопросы, использование наглядности, критерии оценки проектов.

Учащиеся должны уметь:

– самостоятельно планировать этапы работы, соблюдать сроки, вести дневник разработки и распределять роли в команде;

– писать код с использованием функций, списков, таймеров, случайных чисел, вложенных условий и оптимизировать его;

– находить и исправлять ошибки в логике, интерфейсе и балансе, применять чек-листы и методы тестирования;

– готовить презентацию и защищать проект перед группой и родителями, структурировать рассказ (цель, идея, алгоритмы, сложности, решения);

– отвечать на вопросы и учитывать обратную связь для улучшения проекта, демонстрировать портфолио решений и объяснять ключевые механики;

– проводить рефлексию, выделять сильные и слабые стороны проекта, формулировать планы доработки.

Формы занятий:

– аналитические занятия и мозговые штурмы;
– проектная работа в командах или индивидуально;
– консультации и промежуточный контроль;
– подготовка презентации и репетиции;
– защита проектов и рефлексия.

Тема 4.1. Выбор идеи проекта (2 часа)

Теория. Критерии выбора идеи; анализ референсов; формулирование концепции; структура ТЗ; критерии оценки.

Практика. Сбор идей; анализ референсов; формулирование концепции; составление ТЗ; согласование идеи с педагогом.

Тема 4.2. Планирование проекта (2 часа)

Теория. Этапы проекта и сроки; распределение задач.

Практика. Составление плана работы.

Тема 4.3. Разработка итогового проекта (6 часов)

Практика. Самостоятельная работа над проектом; использование освоенных конструкций; индивидуальные консультации педагога; последовательная доработка.

Тема 4.4. Тестирование проекта (2 часа)

Теория. Методы тестирования, виды ошибок и способы их обнаружения, чек-листы проверки, оптимизация кода, принципы чистого кода.

Практика. Проверка всех механик игры; поиск и исправление ошибок; оптимизация кода; подготовка к предзащите; составление списка доработок.

Тема 4.5. Подготовка презентации (2 часа)

Практика. Подготовка слайдов и демонстрационных материалов; репетиция выступления; сбор вопросов и подготовка ответов.

Тема 4.6. Предзащита (2 часа)

Практика. Демонстрация проекта; получение обратной связи; доработка по замечаниям; корректировка интерфейса и механик.

Тема 4.7. Итоговая защита (4 часа)

Практика. Публичная презентация проекта; демонстрация работы; ответы на вопросы жюри и одноклассников; рефлексия; обсуждение сильных и слабых сторон проекта.

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

В ходе курса реализуется трёхуровневая система контроля: текущий, промежуточный и итоговый.

Входной контроль не проводится.

Текущий контроль — осуществляется на каждом занятии для отслежив ания освоения учебного материала.

Формы:

- опрос теоретического материала;
- контрольные тесты;
- проектные мини-задания.

Промежуточная аттестация — проводится по завершении каждого кейс а для оценки освоения программы и развития личностных качеств.

Формы:

- опрос теоретического материала;
- контрольные тесты;
- практические работы;
- лабораторные работы;
- промежуточные проектные работы.

Итоговое оценивание — проводится в конце курса.

Форма: защита итогового проекта .

Оценка	Результат
Высокий уровень	- Сформирована целостная система знаний по ключевым инструментам и понятиям программирования (функции, списки, таймеры, генераторы случайных чисел, обработка столкновений и др.). - Учащийся уверенно интегрирует изученные механики в проекты, грамотно комбинирует инструменты разных сред (Scratch, MakeCode Arcade, Kodu). - Проявляет самостоятельность в планировании: распределяет задачи, соблюдает сроки, при необходимости корректирует план работы. - Эффективно отлаживает код: находит и устраняет ошибки, оптимизирует производительность, аргументированно объясняет логику принятых решений. - Демонстрирует творческий подход: предлагает оригинальные идеи, дорабатывает базовые механики, внедряет дополнительные элементы (визуальные и звуковые эффекты, системы баланса, сохранения и пр.). - Качественно презентует проект: ясно формулирует идею, наглядно показывает ключевые функции, уверенно и аргументированно отвечает на вопросы. - Ответственно относится к занятиям, соблюдает правила цифровой среды, конструктивно взаимодействует в команде.
Средний уровень	- Знания основных понятий и механик программирования есть, но не систематизированы; допускает отдельные ошибки в понимании терминов и логики работы инструментов. - Реализует базовые игровые механики, но для сложных задач требуется помощь педагога. - Планирует работу с опорой на шаблон или подсказки, иногда отклоняется от сроков, нуждается в контроле этапов. - Находит очевидные ошибки, но не всегда видит причины сбоев или способы оптимизации; при анализе кода нужна направляющая помощь. - В проектной работе реализует заданную идею, но редко предлагает собственные улучшения; творческая составляющая проявляется эпизодически. - Во время презентации уверенно демонстрирует работу проекта, но не раскрывает полностью логику реализации; на уточняющие вопросы отвечает с подсказками. - Выполняет требования к занятиям, в команде действует в основном по распределённым ролям, проявляет самостоятельность при внешней стимуляции.
Низкий уровень	- Фрагментарные или ошибочные представления об основных понятиях и инструментах программирования; не может корректно назвать или объяснить базовые механики. - Не способен самостоятельно реализовать даже простые игровые механики; при выполнении заданий требуется пошаговое сопровождение. - Не планирует работу, не отслеживает сроки, часто не завершает отдельные этапы проекта.

	- Не выявляет и не исправляет ошибки без прямых указаний; не понимает причин некорректной работы кода. - Пассивен в проектной деятельности: не предлагает идей, не участвует в обсуждениях, не стремится улучшить проект. - При презентации демонстрирует только запуск проекта, не объясняет логику, не отвечает на вопросы или даёт неверные ответы. - Низкая самостоятельность и ответственность: нуждается в постоянной внешней мотивации, в командной работе не согласовывает действия, может игнорировать задачи или конфликтовать.

МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ПРОГРАММЫ «ПРОГРАММИРОВАНИЕ ДЛЯ ДЕТЕЙ»

Тема кейса	Форма занятий	Приёмы и методы организации образовательного процесса	Дидактический материал. Электронные источники	Техническое оснащение и расходный материал	Форма подведения итогов
МОДУЛЬ 1 «ЮНЫЙ ИССЛЕДОВАТЕЛЬ» (6–7 лет)					
Кейс 1. Знакомство с компьютером и алгоритмами	Комбинированная (теория + практика с динамической паузой)	Игровой метод; наглядная демонстрация; метод «делай как я»; работа по образцу; элементы парного взаимодействия (по желанию ребёнка).	1. Методические карточки «Команды для робота» (стрелки, направления). 2. Рабочие листы «Считаем шаги», «Найди короткий путь». 3. Задания на ориентацию в пространстве (поле 3×3, 4×4). 4. Карточки с простыми маршрутами для устного проговаривания алгоритма. 5. ПиктоМир: встроенные обучающие сценарии. 6. Scratch Jr: подборка стартовых проектов («Кот идёт по	- ноутбуки с установленным ПиктоМиром и Scratch Jr. - Проектор/экран для показа примеров. - Печатные карточки-стрелки (для офлайн-разминки). - Листы в клетку 3×3 для тренировки ориентации.	Промежуточный просмотр мини-проектов: алгоритм из 3–5 команд, сцена с персонажем (демонстрация на проекторе). Мини-опрос «Что запомнил?» (3 простых вопроса).

			дорожке», «Машина едет к дому»). 7. Короткие видеоинструкции (до 2 мин) по запуску программ и базовым действиям.		
Кейс 2. Циклы и закономерности	Комбинированная (с динамической паузой)	Метод моделирования (выкладываем команды фишками); мозговой штурм по поиску закономерностей; мини-исследование (сравнение маршрутов); парное программирование (по желанию).	1. Шаблоны раскадровок (3–4 кадра) для мультфильма. 2. Карточки «События» (флажки, сообщения). 3. Задания на симметрию и зеркальное отражение («дорисуй вторую половину»). 4. Рабочие листы с ритмическими последовательностями. 5. Карточки-алгоритмы с пропусками для заполнения. 6. ПиктоМир: сценарии с циклами. 7. Scratch Jr: проекты с повторяющимися действиями.	- Ноутбуки с ПиктоМиром и Scratch Jr. - Проектор для разбора ошибок. - Цветные маркеры для выделения повторяющихся частей.	Промежуточный просмотр проектов «Мой город», «Моё настроение»; мини-выставка лучших алгоритмов. Рефлексия «Что было самым простым/сложным?» (выбор пиктограммы).

			8 Примеры «неправильных» алгоритмов (с лишними или пропущенными шагами) для анализа.		
Кейс 3. Творческие проекты	Комбинированная (с динамической паузой)	Эвристический метод (вопросы «а что, если?»); метод проектов; работа с раскадровкой на бумаге; обсуждение без ПК (логические задачи); приём «совет от соседа» (краткий комментарий по 1 пункту).	1. Чек-лист «Готов ли мой проект?» (работает ли старт, нет ли критических багов, понятны ли правила). 2. Шаблон рассказа: «Идея → Алгоритм → Что было сложно». 3. Карточки ролей для зрителей («задавай вопрос», «скажи 1 плюс», «предложи 1 идею»). 4. Примеры коротких видео с защитой проектов прошлых лет. 5. Шаблоны слайдов для презентации (если используется).	- Ноутбуки с Scratch Jr. - Цветные карандаши, листы А4 для раскадровок. - Проектор для показа лучших решений.	Промежуточный просмотр черновиков, обсуждение идей финальных проектов; мини-показ 2–3 лучших черновиков. Анкета «Что хочу улучшить?» (рисуночная шкала).

Кейс 4. Защита проектов	Комбинированная	Репетиция выступлений; метод «три вопроса» (что хотел сделать, что получилось, что улучшу); рефлексия в игровой форме; приём «тихий старт» (проверка работоспособности до показа).	1. Шаблоны раскадровок (3–4 кадра). 2. Карточки «События» (флажки, сообщения). 3. Задания на симметрию, зеркальность, последовательности. 4. Чек-лист «3 простых правила хорошего проекта» (старт, логика, оформление). 5. Набор сюжетных картинок для генерации идей. 6. Scratch Jr: библиотека фонов и спрайтов. 7. Шаблоны для интерактивных историй. 8. Галерея мини-примеров (3–5 секундные ролики) с разными механиками.	- Ноутбуки для демонстрации. - Экран/проектор. - Сертификаты/значки для награждения. - • Таймер для ограничения выступления (1–2 минуты).	Итоговая презентация проектов родителям; командные алгоритмические игры. Рефлексивная анкета «Что запомнилось больше всего?».
МОДУЛЬ 2: «СОЗДАТЕЛЬ ИГР» (1–2 класс)					

Кейс 1. Основы Scratch	Комбинированная (с перерывом)	Демонстрация с комментированием; «найди ошибку» (разбор чужого кода); парное программирование; мини-соревнование на самый короткий алгоритм; приём «объясни соседу» (закрепление через объяснение).	1. Карточки блоков Scratch (движение, переменные, сенсоры). 2. Готовые сцены-шаблоны для «Гонки по линии» и «Сбора монет». 3. Примеры проектов с намеренно внесёнными ошибками (для тренировки отладки). 4. Листы с мини-заданиями на 1 конструкцию (например, «добавь условие»). 5. Scratch (онлайн/офлайн): стартовые проекты по темам. 6. Библиотека спрайтов и фонов для быстрого старта.	- Ноутбуки с доступом в Интернет. - Scratch (онлайн/офлайн). - Проектор для показа примеров и разбора ошибок. - Принтер для распечатки карточек. - Маркеры/фломастеры для схем алгоритмов.	Промежуточный просмотр проектов («Гонка по линии», «Сбор монет»).
Кейс 2. Введение в 3D	Комбинированная (с перерывом)	Метод «сначала строим, потом программируем»; ролевая игра «Архитектор мира»; мини-турнир «Чья трасса круче»; приём «дополни правило» (дети предлагают 1 условие для поведения персонажа).	1 1. Инструкции по инструментам ландшафта (горы, вода, плоскость). 2. Примеры простых правил Kodu («Когда видишь объект → делай действие»). 3. Шаблоны трасс и квестов. 4. Карточки с идеями для 3D-уровней.	- Ноутбуки с Kodu Game Lab. - Проектор. - Карточки с идеями для 3D-уровней. - Рабочие листы для эскизов ландшафта. - Цветные карандаши/фломастеры.	Промежуточная защита проектов «Гонки в 3D», «Квест в замке»; мини-голосование за лучший мир. Самооценка «Что получилось лучше всего?» (шкала 1–3).

			5. Листы для эскизов ландшафта. 6. Kodu Game Lab: обучающие видео по базовым инструментам. 7. Галерея примеров простых 3D-миров.		
Кейс 3. Сложная логика в Scratch	Комбинированная (с перерывом)	Разбор механик популярных игр; мозговой штурм «как сделать прыжок реалистичным»; работа по ТЗ (меню, враги, финиш).	1. ТЗ-карточки для платформера (кнопки, гравитация, враги). 2. Сценарии генерации препятствий (таймеры, случайные числа). 3. Готовые фоны и спрайты для ускорения старта. 4. Примеры логических ошибок и их решений. 5. Scratch: примеры платформеров и лабиринтов.	- Ноутбуки со Scratch. - Проектор. - Набор спрайтов и фонов (если нет времени рисовать).	Промежуточный просмотр прототипов («Платформер-уровень»).

Кейс 4. Защита проектов	Комбинированная (с перерывом)	Консультации по индивидуальным проектам; метод «обратной связи от сверстников»; мини-предзащиты.	1. Шаблон ТЗ (тема, механики, список блоков). 2. Чек-листы отладки (старт, столкновения, счётчик, звук). 3. Примеры питчей (короткие видео с защитой проектов прошлых лет).	- Компьютеры со Scratch и Kodu. - Редакторы презентаций PowerPoint. - Проектор для предзащит и итоговой защиты.	Итоговая защита проектов + игровой турнир (оценка проекта и презентации), рефлексивный круг «Чему я научился». Анкета самооценки проекта (по 5 критериям).
МОДУЛЬ 3: «ИНЖЕНЕР И АРХИТЕКТОР КОДА» (3–4 класс)					
Кейс 1. Функции и координаты	Комбинированная (с перерывом)	Проблемные вопросы («зачем выносить блок отдельно?»); разбор чужого кода с поиском дублирования; мини-задачи на математику в играх; приём «оптимизируй код» (сократить количество блоков).	1. Карточки функций («Пойти на обед», «Выстрелить») для тренировки. 2. Задания на расчёт траектории (парабола, углы). 3. Примеры списков (рекорды, инвентарь) и генераторов случайных чисел.	- Компьютеры со Scratch. - Проектор. - Рабочие листы с координатной сеткой и формулами.	Промежуточный просмотр проектов («Система жизней», механика с таймером).

Кейс 2. Пиксельные игры	Комбинированная (с перерывом)	Приём «Механика на карточках» (собрать цепочку: столкновение → счёт → переход уровня);); разбор механик «Змейки» и шутера; мини-задача «Исправь одну ошибку» в проекте «Змейка»; приём «Сделай заметнее» (1 эффект — вспышка/тряска/звук); оптимизация кода через функции	1. Документация MakeCode Arcade. 2. Карточки механик с пиктограммами. 3. Листы планирования уровня (старт, финиш, враг, препятствие). 4. Пары скриншотов «Плохо/Хорошо» для визуальных эффектов. 5. MakeCode Arcade: шаблоны «Змейка», «Простой уровень». 6. Галерея простых пиксель-спрайтов (кот, робот, монстр). 7. Скринкасты (до 2 мин) «3 приёма для понятного эффекта».	- Ноутбуки с MakeCode Arcade (веб-версия). - Проектор. - Примеры пиксельных спрайтов для вдохновения.	Промежуточная защита мини-игр: проверка по 3 критериям (механика работает, уровень понятен, эффект не мешает).
Кейс 3. Продвинутое проектирование	Комбинированная (с перерывом)	Ролевая смена «на 10 минут»: ребёнок пробует 3 роли по очереди — «Создатель мира» (ставит объекты), «Логик» (добавляет 1 правило персонажу), «Тестировщик» (проверяет, всё ли срабатывает).	1. Листы ролей с пиктограммами и простыми обязанностями. 2. Шаблон «Идея → 1 механика → 1 объект → эскиз». 3. Карточки «Логика Kodu» («Когда вижу → делаю», «Если касается → останавливаюсь»). 4. Чек-лист тестирования (старт, победа/поражение, понятность правил).	- Ноутбуки со Scratch, MakeCode, Kodu. - Проектор. Бумага и карандаши для эскизов.	Промежуточный показ прототипов: 1 механика + 1 персонаж + 1 условие. Оценка по 3 критериям («Понятно?», «Работает?», «Не перегружено?»), баллы 1–3.

			5. Kodu: 3–4 примера с патрулированием и базовыми условиями. 6. Галерея 3D-миров с 2–3 объектами (акцент на понятность).		
Кейс 4. Проектная деятельность	Комбинированная (с перерывом)	Обсуждение «Что делает игру интересной?» (на примере 2–3 детских игр — разбираем по 1–2 пунктам: цель, награда, понятные правила); индивидуальный мозговой штурм «3 идеи за 3 минута» (ребёнок записывает 3 варианта, потом выбирает 1); приём «1 вопрос педагогу» (каждый формулирует 1 простой уточняющий вопрос по своему проекту); репетиция «Выступление на 1 минуту» (индивидуально: идея + 1 механика + эскиз).	1. Шаблон презентации (3–5 слайдов: идея, механики, скриншоты, ссылка). 2. Список критериев оценки (логика, баланс, оформление, выступление). 3. Чек-лист для проекта (старт, победа/поражение, понятность) — для самопроверки. 4. Галерея примеров детских проектов (скриншоты + 3–4 предложения).	- Компьютеры для доработки проектов. - Редакторы презентаций. - Проектор для защиты.	Индивидуальная питч-сессия: 1 минута на ребёнка (идея + механика + эскиз). Рефлексия «3 вещи, которые я научился делать».

КАДРОВОЕ ОБЕСПЕЧЕНИЕ

Преподавание данной программы могут осуществлять педагогические работники, владеющие набором профессиональных навыков в области информационно-коммуникационных технологий, при наличии необходимых компетенций и уровня профильной подготовки.

ОПИСАНИЕ МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЙ БАЗЫ, НЕОБХОДИМОЙ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО КУРСУ

Для реализации курса «Программирование для детей» помещение должно соответствовать следующим характеристикам:

– аудитории, оборудованы интерактивной доской, проектором, ноутбуком;

– каждый обучающийся выполняет практические работы за отдельным компьютером с сохранением результатов в облачном хранилище;

– обеспечено стабильное подключение к сети Интернет для работы с онлайн-версиями сред программирования, загрузки шаблонов и справочных материалов.

ПЕРЕЧЕНЬ ЛИТЕРАТУРЫ, НЕОБХОДИМОЙ ДЛЯ ОСВОЕНИЯ ПРОГРАММЫ

Перечень литературы, использованной при написании программы:

1. Бердитт Р. Программирование на Scratch с нуля. Создаём весёлые игры. — М.: Эксмо, 2021. — 224 с.
2. Голиков Д. 42 проекта на Scratch 3 для юных программистов. — СПб.: БХВ-Петербург, 2021. — 208 с.
3. Голиков Д. Scratch 3 для юных программистов. — СПб.: БХВ-Петербург, 2020. — 168 с.
4. Свичкарева Л. С., Цыбарт Р. Р. Учимся думать. Алгоритмика для начальной школы. — М.: Феникс, 2024. — 31 с
5. Вордерман К. Программирование для детей. Иллюстрированное руководство по языкам Scratch и Python. — М.: Манн, Иванов и Фербер, 2019. — 240 с.
6. Пиксели! Курс по пиксельной графике и анимации для новичков. — М.: Питер, 2023. — 176 с.
7. Астахова К. И. Создаём игры с Kodu Game Lab. — М.: Лаборатория знаний, 2019. — 122 с

Перечень литературы, рекомендованной обучающимся:

1. Рабочие тетради «Алгоритмы в играх» (серии для 6–7 лет, 1–2 класс, 3–4 класс) — тематические задания на логику, счёт, симметрию и

планирование действий.

2. Голиков Д. «42 проекта на Scratch 3 для юных программистов». — СПб.: БХВ-Петербург, 2021. — 208 с

3. . Астахова К. И. «Создаём игры с Kodu Game Lab». — М.: Лаборатория знаний, 2019. — 122 с.

Перечень раздаточного материала:

1. Тематические презентации по каждому кейсу.

ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ

Программное обеспечение:

- операционная система: Windows;
- офисное программное обеспечение: MS Office;
- среды программирования по модулям:
  - модуль 1 (6–7 лет): ПиктоМир, Scratch Jr (планшеты/компьютеры);
  - модуль 2 (1–2 класс): Scratch (онлайн/офлайн), Kodu Game Lab;
  - модуль 3 (3–4 класс): Scratch, Microsoft MakeCode Arcade, Kodu Game Lab.

Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения программы:

1. https://code.org/ — международный сайт с курсами по программированию.
2. https://codewards.ru/ — сайт с курсами по программированию, онлайн-платформа по обучению языкам программирования.
3. https://scratch.mit.edu/ — официальный сайт среды Scratch.
4. makecode.com — среда Microsoft MakeCode Arcade с примерами и документацией.
5. kodu.game.lab — материалы и шаблоны для Kodu Game Lab.
6. microbit.org — дополнительные примеры логики и сенсоров (для междисциплинарных связей).
7. https://урок.рф — образовательный портал с методическими материалами.