Министерство просвещения Российской Федерации
федеральное государственное бюджетное
образовательное учреждение высшего образования
«Уральский государственный педагогический университет»
Институт математики, физики, информатики и технологий
Кафедра информатики, информационных технологий и методики обучения информатике

МЕТОДИКА ОБУЧЕНИЯ МЛАДШИХ ШКОЛЬНИКОВ СОЗДАНИЮ 3D ИГР С ИСПОЛЬЗОВАНИЕМ ВИЗУАЛЬНОГО ЯЗЫКА ПРОГРАММИРОВАНИЯ KODU В РАМКАХ ЭЛЕКТИВНОГО КУРСА

Выпускная квалификационная работа
бакалавра по направлению подготовки
44.03.05 – Педагогическое образование
Профиль «Математика и Информатика»

Квалификационная работа
допущена к защите:
Зав.кафедрой

_____ _____
дата подпись

Исполнитель: Ерондаева Ю.В., МИ-1701
ИМФИиТ

_____
подпись

Руководитель: Рожина И.В., к.п.н.,
доцент кафедры ИИТиМОИ

_____
подпись

Екатеринбург – 2022

РЕФЕРАТ

Ерондаева Ю.В. МЕТОДИКА ОБУЧЕНИЯ МЛАДШИХ ШКОЛЬНИКОВ СОЗДАНИЮ 3D ИГР С ИСПОЛЬЗОВАНИЕМ ВИЗУАЛЬНОГО ЯЗЫКА ПРОГРАММИРОВАНИЯ KODU В РАМКАХ ЭЛЕКТИВНОГО КУРСА. Выпускная квалификационная работа: 58 стр., рис. 33, табл. 7, библ. 44 назв., приложений 2.

Ключевые слова: алгоритм, визуальное программирование, младшие школьники, обучение программированию, Kodu Game Lab.

Объект исследования – процесс обучения информатике.

Цель работы – разработка элементов методики обучения программированию с использованием языка KODU для создания 3D игр.

В работе описаны особенности обучения младших школьников программированию в начальной школе. Рассмотрены среды визуального программирования с использованием конструктора игр для изучения обучающимися младших классов. Выбрана оптимальная среда программирования KODU для создания 3D игр, также обосновано использование данного языка.

Разработан элективный курс *«Создаем 3D игры с Kodu»,* состоящий из 5 занятий. Данный курс позволит овладеть основами записи и выполнения алгоритмов. Учащиеся начальных классов получат знания об алгоритмах и научатся строить простейшие программы в среде программирования Kodu Game Lab.

Апробация разработанного курса проводилась в МОУ «Килачевская СОШ» Свердловской области с обучающимися с возрасте 8-11 лет. Результат апробации помог оценить эффективность курса, а также дополнить в учебно-тематическом планировании количество часов на каждую тему.

Оглавление

Введение

Современный уровень информатизации общества предполагает активное использование информационных технологий во всех сферах жизнедеятельности человека. На сегодняшний день наука развивается очень стремительно, и современное образование часто не успевает за ней, вследствие чего, появляется необходимость реализации элективных курсов по информатике.

Существует множество пропедевтических курсов по предмету «Информатика», направленных на общее знакомство с информационными процессами, информацией, в целом, и носят исключительно развивающий характер. Однако в наши дни этого мало для овладения навыкам эффективного использования компьютера детьми начальных классов.

Согласно ФГОС – 3 начального общего образования предметные результаты по учебному предмету "Информатика" должны обеспечивать развитие логического и алгоритмического мышления, в частности, умение строить простейшие алгоритмы и использовать изученные алгоритмы в учебных ситуациях [35]. Достижение данного результата на уроках информатики у младших школьников становится возможным благодаря включению в образовательный процесс программирования в визуальной среде.

Остаётся лишь выбрать язык программирования, который будет легок и доступен детям начальных классов. Язык должен быть не только прост в изучении, но и иметь возможность решения различных задач, понятных детям младшего возраста. В настоящее время существует много языков программирования для изучения в школе, но для изучения в начальных классах названным критериям соответствует визуальная среда программирования Kodu Game Lab.

Kodu Game Lab – визуальная среда программирования, которая позволяет разрабатывать собственные игры, а также может заинтересовать школьников возможностями в области компьютерных наук [8].

На сегодняшний день разработано достаточное количество курсов по обучению программированию школьников средних классов, но курсов, рассчитанных на детей начальной школы не так много. В связи с этим, **целью работы** является разработка элементов методики обучения программированию с использованием языка KODU для создания 3D игр.

Объектом данной работы является процесс обучения информатике.

Предмет – обучение младших школьников основам программирования.

Для достижения поставленной цели предполагается решение следующих задач:

- изучить научную и учебно-методическую литературу с целью выявления особенностей обучения программированию обучающихся младших классов; изучения понятия «элективный курс»;
- провести сравнительный анализ сред визуального программирования по созданию игр; рассмотреть особенности программирования в среде Kodu Game Lab;
- разработать элективный курс;
- провести апробацию разработанных материалов.

Глава 1. Теоретические основы программирования в начальной школе

1.1. Особенности обучения младших школьников программированию в начальной школе

Младший школьный возраст — это период позитивных изменений и преобразований. Поэтому так важен уровень достижений, осуществленных каждым ребенком на данном возрастном этапе.

Между вторым и третьим классами происходит скачок в умственном развитии учащихся. Именно на этом этапе обучения происходит активное усвоение и формирование мыслительных операций, более интенсивно развивается вербальное мышление, т.е. мышление, оперирующее понятиями. Новые возможности мышления становятся основанием для дальнейшего развития других познавательных процессов: восприятия, внимания, памяти. В связи с происходящими изменениями наиболее целесообразно включить изучение основ программирования во 2-3 классе [11, 17].

Младший школьный возраст является сенситивным для:

- формирования мотивов учения, развития устойчивых познавательных потребностей и интересов;
- развития продуктивных приемов и навыков учебной работы, умения учиться;
- раскрытия индивидуальных особенностей и способностей;
- развития навыков самоконтроля, самоорганизации и саморегуляции;
- становления адекватной самооценки, развития критичности по отношению к себе и окружающим;
- усвоения социальных норм, нравственного развития;
- развития навыков общения со сверстниками, установления прочных дружеских контактов.

В то же время в младшем школьном возрасте стремление проникнуть в сущность явлений, вскрыть их причину заметно не проявляется. Младшего школьника затрудняет выделение существенного, главного.

Вопросы обучения детей основам программирования в научной и научно-методической литературе обсуждаются учеными, учителями и специалистами в области информационных технологий. Рассматривая программирование, как общение с компьютером на языке понятном ему, исследователи поддержали идею обучения детей программированию с раннего возраста [4].

Научно-методическое обоснование раннего обучения информатике приведено в работах И. Н. Антипова, А. П. Ершова, Г. А. Звенигородского, Л. Л. Босовой, С. А. Бешенкова, А. В. Горячева, Н. В. Матвеевой, А. Л. Семенова, Ю. А. Первина и целого ряда других отечественных специалистов. Их позиция созвучна мнению зарубежных экспертов. В частности, профессор Comenius University (Братислава, Словакия) Иван Калаш в своих работах регулярно отмечает не только важность раннего обучения информатике и программированию, но и высокий статус информатики как учебной дисциплины в программе подготовки младших школьников. В частности, профессор И. Калаш настаивает на обязательном, а не факультативном характере курсов информатики [19].

При обучении раннего программирования возникает другая проблема: учащиеся начальных классов не могут запоминать сложные команды, длинные коды, написанные, как правило, на иностранном языке (который они только начинают изучать).

Для решения данного противоречия необходим язык программирования близкий к образу мышления детей, содержащий команды для работы с интересными и понятными для них объектами, но в то же время, дающий прочную основу для изучения других языков программирования.

С 2014 года основы программирования начали изучать в школах Великобритании. Учащиеся начальных классов британских школ с помощью таких программ, как MIT’s Scratch, Kodu, Logo учатся создавать простые программы по блокам, а в одиннадцать лет учащиеся должны иметь

представление о базовых алгоритмических структурах и использовать их при создании учебных программ.

Тенденцию раннего обучения программированию в школе поддерживают многочисленные ведущие компании в области информационных технологий, предоставляя доступные инструменты для программирования, они также поддерживают идею обучения программированию в школе. Огромное количество пользователей таких ресурсов, как MIT’s Scratch и AppInventor, Codecademy, Code.org и других, показывают растущий интерес современного общества людей к знанию и пониманию искусства программирования [12].

Фундаментальные основы алгоритмизации лежат в теоретической области современной математики – теории алгоритмов, но алгоритмизацию в широком смысле можно понимать как набор практических навыков, основанный на принципах рационального мышления. Поэтому изучение алгоритмизации можно рассматривать в двух аспектах, а именно:

- в развивающем аспекте, который подразумевает развитие алгоритмического мышления учащегося, что является наиболее важным для обучающихся младших классов;
- в программистском аспекте, лежащем в основе развития навыков составления программ, то есть практико-ориентированным, что так же важно для детей.

Таким образом, алгоритмическое мышление представляется основой для изучения программирования [27].

Изучение основ алгоритмизации принято начинать с введения понятия алгоритма, которое относится к базовым понятиям и не может быть определено через другие, более простые понятия.

В учебнике Семакина И.Г. алгоритм определяется как понятное и точное предписание исполнителю выполнить конечную последовательность команд, приводящую от исходных данных к искомому результату [7,28].

В повседневной жизни ученики младших классов не встречаются с данным понятием в таком его определении, но деятельность человека всегда происходит по каким-то алгоритмам, о чем важно сообщить учащимся на первом же уроке, аргументируя данное положение примерами. При введении понятия алгоритма, учителю следует обратить внимание на то, что конечная последовательность команд всегда составляется с ориентацией на исполнителя алгоритма [3].

И на начальном этапе изучения темы лучше всего взять за основного исполнителя алгоритмов человека. В таком случае ученикам будет проще самим выступить в роли исполнителей несложных алгоритмов, например рисования геометрических фигур на доске.

К окончанию третьего учебного года дети способны брать на себя роль ведущего в знакомых играх и упражнениях, быть внимательными к остальным участникам, уметь договариваться с ними об условиях игры, давать внятные инструкции, контролировать ход выполнения заданий.

Поэтому составляя систему команд исполнителя необходимо ориентироваться на опыт детей, и строить ее таким образом, чтобы выполнялось первое свойство алгоритма, заключающееся в его понятности как для составителя, так и для исполнителя, т.е. алгоритм не должен быть рассчитан на принятие исполнителем самостоятельных решений, не предусмотренных составителем алгоритма [10].

Основной целью раздела изучения алгоритмизации является овладение учениками методикой построения алгоритмов. Традиционно применяемым дидактическим средством в этом разделе являются учебные исполнители алгоритмов. Главными преимуществами учебных исполнителей являются ясность и наглядность процесса исполнения программы [1].

Обучение программированию лучше организовать в ходе решения задач, подобранных в специально выстроенной последовательности, которая определяется следующими принципами:

- «от простого к сложному» с постепенным усложнением предоставляемых задач;
- принцип «новизны», при котором каждая задача вносит новый элемент знаний;
- принцип «наследования», подразумевающий наложение новых знаний с ранее полученными.

Лучшим способом для предоставления информации на уроках информатики младшим школьникам является игровая форма. Для изучения основ алгоритмизации школьникам в возрасте от 7 до 12 лет (с 1 по 5 классы) отлично подойдут уроки информатики с использованием конструктора игр.

Конструктор игр — это программа, объединяющая в себе игровой движок и интегрированную среду разработки — систему программных средств, используемых программистами для разработки программного обеспечения. Простота процесса создания игр достигается за счет обеспечения конструкторов готовыми деталями и логическими операциями, из которых конструируется игра. Основными примерами деталей являются объекты, играющие роль персонажей игры, а также комнаты, либо уровни, — окна программы, на фоне которых разворачивается игровой процесс [34].

Первоочередными критериями выбора конструктора игр для использования в учебном процессе могут быть:

- максимальное упрощение базовых понятий;
- прикладной характер заданий и реальная польза от их выполнения (освоенный на практике материал легче усваивается, следовательно, задачи должны приносить ощутимый результат и демонстрировать, как полученные знания и навыки помогут в решении реальных проблем);
- интересная форма занятий;
- геймификация занятий (в современном мире игры очень тесно связаны со сферой IT, поэтому игровые техники можно встроить в учебный процесс очень просто и гармонично).

Плюсы использования конструкторов очевидны — большая часть работы сделана профессиональными программистами-разработчиками, включая оптимизацию и структуризацию рабочего процесса. От учащихся требуются только идея, графический и звуковой контент.

Тем самым, конструкторы игр можно использовать в процессе обучения и предлагать обучающимся на элективных курсах.

Рассмотрим понятие, структуру и требования к организации элективных курсов в школе [39].

Элективные курсы – это краткосрочные тематические курсы (модули), которые общеобразовательное учреждение предлагает учащимся на основе изучения их запросов и реализует за счет часов школьного компонента.

Каждый элективный курс представляет собой завершенную дидактическую единицу, нацеленную на получение одного-двух образовательных результатов.

Требования к оформлению программы элективного курса [15]

Структура программы:

1. Название программы.
2. Пояснительная записка.
3. Учебно-тематическое планирование.
4. Тематическое планирование.
5. Методические рекомендации.
6. Список литературы.

Название программы

Название программы задается в соответствии с ее содержанием. Подзаголовок программы указывает, к какому виду элективных курсов относится данная программа по содержанию.

Пояснительная записка

Пояснительная записка должна включать:

- цель (-и) и планируемые результаты программы;
- описание способа (-ов) оценки планируемых результатов;

- характеристику элективного курса;
- описание принципиальных получения заявленных образовательных результатов, способов организации освоения элективного курса учащимися;
- характеристику ресурсов, необходимых для реализации курса.

Цель программы

Под целью курса следует понимать прогнозируемые результаты обучения, сформулированные в обобщенной форме с использованием терминов, принятых в дидактике и одинаково понимаемых всеми участниками образовательного процесса.

Цель курса показывает, что должен сделать педагог, работающий по программе. При определении цели программы должны быть использованы ответственные формулировки (например, «научить», «сформировать», «передать технологию» и т.п.); безответственные формулировки (например, «способствовать», «создать условия для…») недопустимы.

Формулировка цели не должна включать указание на средства ее достижения.

Исходя из специфики содержания и краткосрочности элективных курсов, программа элективного курса, как правило, подчинена одной цели.

Планируемые образовательные результаты

Планируемые результаты обучения должны конкретизировать цели курса. Они должны состоять из одной или нескольких легко вычленяемых и автономно проверяемых единиц содержания. Достаточной считается детализация планируемых образовательных результатов, при которой каждый результат отличается от остальных и существует возможность зафиксировать факт достижения каждого планируемого результата посредством педагогического измерения.

Формулировки образовательного результата не могут содержать фраз, имеющих двоякое толкование и\или требующих детализации или конкретизации. Формулировки результатов должны быть написаны языком, доступным для понимания учащихся и их родителей.

Характеристика ресурсов

Следует также указать необходимые для реализации программы ресурсы: привести перечень материальных ресурсов (оборудование, приборы, материалы, возможность выхода в глобальную сеть и т.д.), необходимых и достаточных для достижения планируемых результатов обучения.

Учебно-тематическое планирование

Учебно-тематическое планирование оформляется в виде таблицы. Планирование учебного времени должно давать представление о количестве часов, в том числе аудиторной работы, консультаций, самостоятельной работы.

Тематическое планирование

Тематическое планирование включает в себя название тем и содержание обучения по каждой теме. Программа элективного курса может представлять собой одну тему, тогда тематическое деление содержания обучения не проводится.

Методические рекомендации

Методические рекомендации должны способствовать качественной подготовке и проведению занятий учителем и учащимися и включают:

- основные содержательные компоненты по каждому разделу или теме;
- описание приемов и средств организации учебно-воспитательного процесса;
- описание форм проведения занятий;
- дидактические материалы.

1.2. Языки программирования для детей начальной школы

В начале обучения программированию, учитель сталкивается с нелегкой задачей в выборе языка. Существует множество сред визуального программирования с использованием конструктора игр для изучения обучающимися младших классов. Перечислим некоторые из них:

1. Язык программирования «Logo»

«Logo» один из самых первых разработанный и получивший популярность языков для обучения основам программирования детей младшего возраста. Данный язык разработан в 1967 году педагогом Сеймуром Пейпертом и ученым Идит Харелем.

Древнегреческое слово «logos» — мысль. Подчеркивает основную идею названия языка — научить алгоритмическому мышлению. Отсюда и суть языка заключается в управлении исполнителем. За 55 лет «Logo получил почти 250 модификаций, среди которых есть русскоязычные.

Большое распространение получил исполнитель Черепашка, который реализован определенным набором команд. Этот исполнитель двигается по заданной пользователем траектории и с помощью пера оставляет след.

Наблюдая за поведением черепашки, ученик вникает в смысл каждой написанной им команды, осваивая не только язык программирования, но и основы алгоритмизации.

«Logo» активно используется в школьном и дошкольном образовании, а в ряде стран (Англия, Австралия) — это обязательный предмет образовательной программы [6].

На рисунке 1 представлен интерфейс программы.

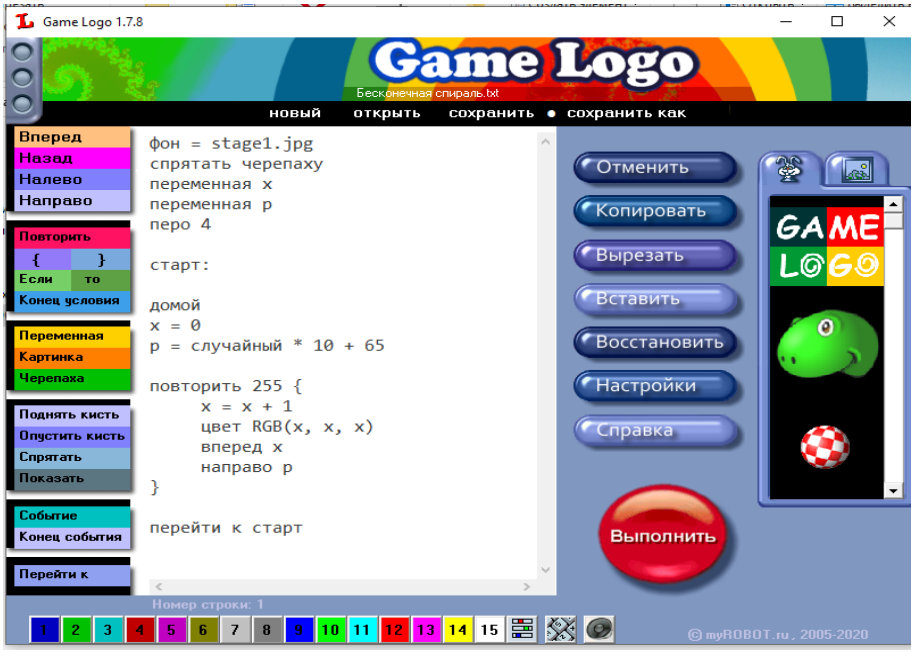

Рис. 1. Среда программирования Logo

2. Игровая платформа «Kodu Game Lab»

Kodu Game Lab – визуальная среда программирования, разработанная компанией Microsoft для обучения учащихся программированию и алгоритмизации.

Особенность Kodu состоит в том, что ученику не нужно сосредотачиваться на способах составления программ. Это открывает возможность посвятить время на разработку сюжета игры, логику, которой подчиняются действия персонажей, на устройство мира, в котором происходят действия. Трехмерная графика, дает, учащимся окунуться в созданный своими руками виртуальный мир [38].

Идеи игр и игровых жанров лишены границ — все зависит от вашей фантазии.

Интерфейс программы представлен ниже на рисунке 2.

Рис. 2. Среда программирования Kodu Game Lab

3. Среда программирования Scratch

Scratch – это бесплатная визуальная среда программирования, для детей младших классов позволяющая создавать свои мультфильмы, анимации, игры и презентации. Scratch создан как продолжение идей языка «Logo» и конструктора Лего.

Главным плюсом программы является простой интерфейс, представленный на рисунке 3, а так же возможность составлять программу из уже готовых команд. В ней ученики управляют объектами-спрайтами [32]. Для них задается графическое представление, которое может быть загружено из любого источника изображения, и последовательности действий, который составляется из блоков по принципу «тащи-и-бросай». Эти блоки бывают нескольких видов: движение, внешность, звук, перо (использование «черепашьей» графики), контроль, сенсоры, операции. Среда Scratch разработана в 2006 г. под руководством Митчела Резника группой Lifelong Kindergarten в лаборатории Media Lab Массачусетского технологического института [23].

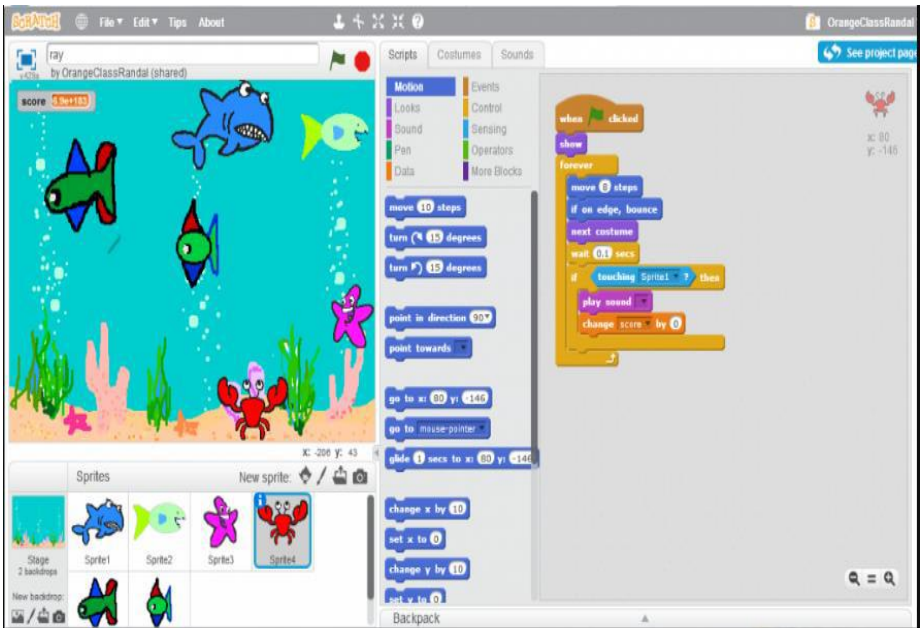

Рис. 3. Визуальная среда программирования Scratch

4. Игровая среда Codmonkey

Codmonkey – это игровая среда для детей школьного возраста, позволяющая детям программировать действия обезьяны. Игра разбита на несколько уровней, в которых пользователю необходимо взяв банан, преодолеть различные препятствия. Действия обезьяны программируется путем выбора команд и ввода их в программную среду [40].

Единственный минус игровой среды, это бесплатный доступ только к первым 30 уровням, за последующие уровни взимается плата. На рисунке 4 представлен скриншот игры, где видно, как пользователь прописывает алгоритм для управления обезьянкой.

Рис. 4. Игровая среда программирования Codmonkey

5. Образовательный ресурс code.org

Code.org – это образовательная среда, цель которой научить детей младшего возраста программированию. На сайте размещены множество уроков по информатике и программированию, а так же различные практические задания для закрепления пользователями пройденного материала.

Одно из заданий представлено на рисунке 5, пользователь составляет алгоритм действий из готовых блоков команд для решения определенной задачи. Code.org каждый год проводит акцию «Час кода», с целью заинтересовать детей и подростков изучением языков программирования, а так же дать возможность детям совершенно бесплатно проверить свои силы в рамках программирования. Акция проводится в 180 странах [16].

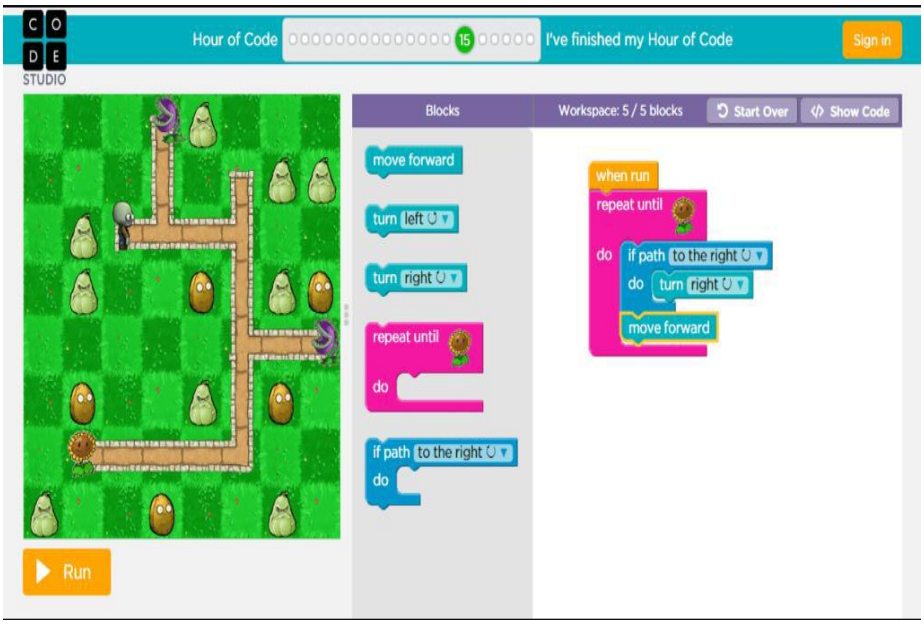

Рис. 5. Образовательный ресурс code.org

6. Акция «Час Кода»

«Час кода» включает в себя 10 уровней сложности, в которых ученик получает возможность, управляя роботами выполнять различные задания. Для управления роботами необходимо написать программу, позволяя ему двигаться по клеточкам и выполнять определенные действия. Каждый уровень сопровождается сюжетной линией, позволяющей повысить интерес ребёнка в изучении программирования. На рисунке 6 изображен один из уровней, на котором пользователю необходимо составить алгоритм, благодаря которому робот сможет распилить все деревянные объекты [26].

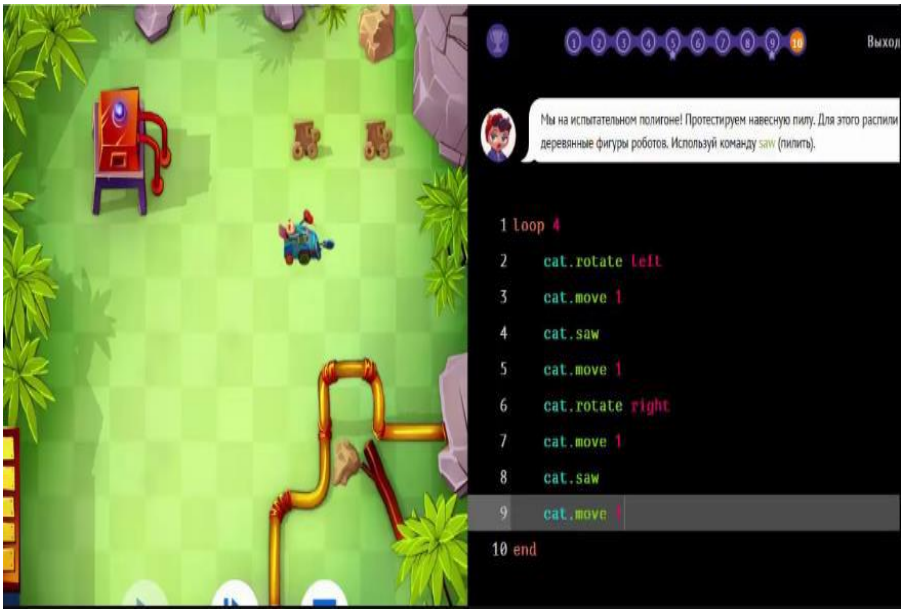

Рис. 6. «Час кода»

7. Игровая платформа Lightbot

Lightbot – игра по программированию, с помощью которой ученики осваивают азы программирования, заставляя робота подсвечивать все голубые клетки на игровом поле [37].

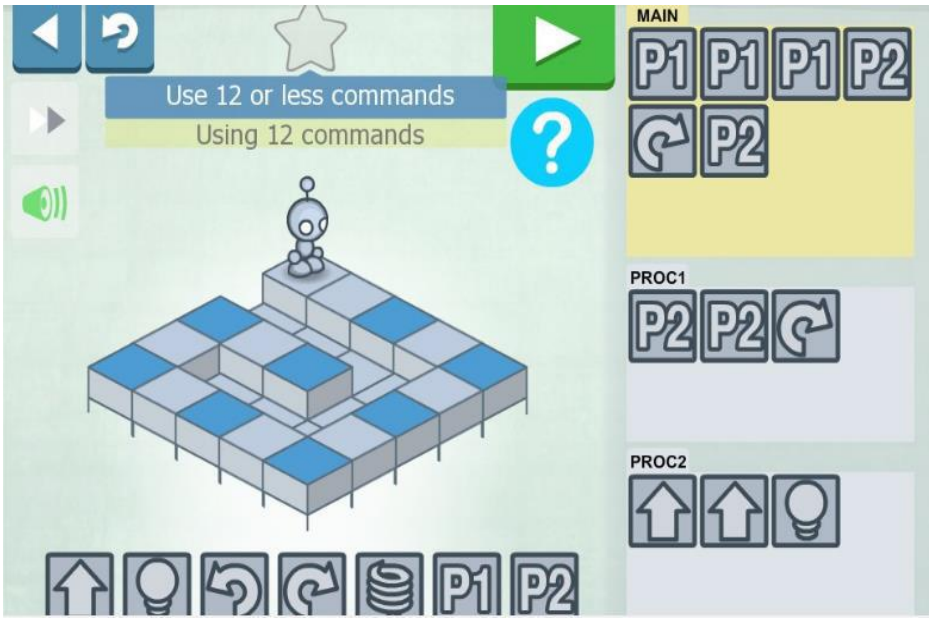

Рис. 7. Игровая платформа «Lightbot»

Игра рассчитана для детей от 4 до 8 лет, широко используется в школах для обучения программированию. На рисунке 7 представлен интерфейс программы. Минус игры – отсутствие бесплатной версии, пользователю предлагается только час бесплатного времени на знакомство с игрой.

8. Игровая платформа Kodable

Kodable – игровая платформа для детей от 5 до 8 лет, представляющая собой лабиринт, состоящий из 105 уровней сложности. На рисунке 8 представлен один из уровней игры, где пользователь должен пройти данный ему лабиринт, программируя персонажа [36].

Игра позволяет ребенку освоить язык программирования, алгоритмизацию, условия, циклы и прочее.

В бесплатной версии игра имеет только 45 уровней, остальные уровни доступны в платной версии игры. Так же игра включает в себя уроки по расширению словарного запаса и инструкции для обучения.

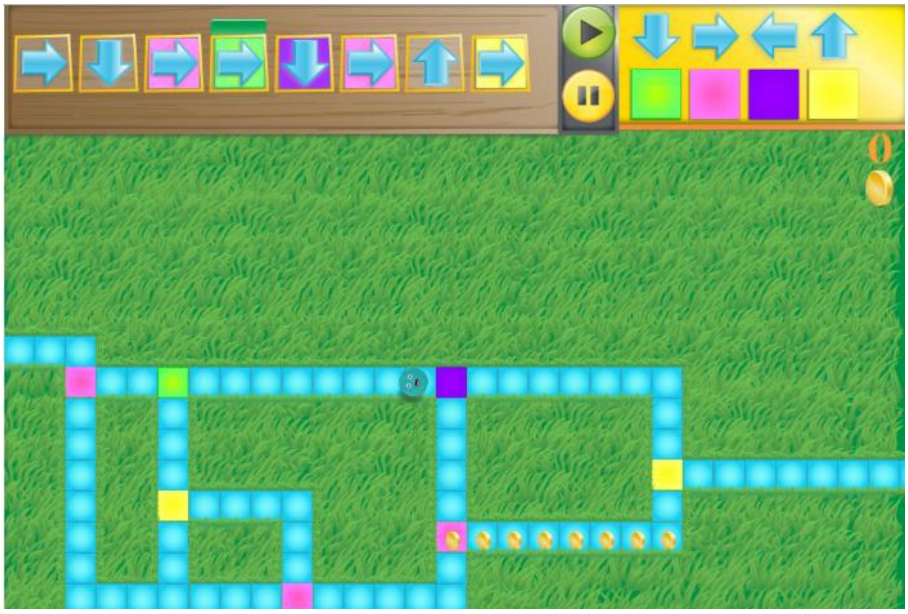

Рис. 8. Игровая платформа «Kodable»

Анализируя и обобщая вышеперечисленные визуальные среды программирования, сравним их по следующим критериям: режим использования, год разработки, перевод интерфейса на русский язык, поддержка платформ, бесплатная версия, создание игры, разработка программного кода, графика (таблица 1).

Сравнительная характеристика визуальных сред программирования

Таблица 1.Сравнительная характеристика сред

Критерии	Визуальная среда							
	Logo	**Kodu Game Lab**	**Scratch**	**Codmonkey**	**code.org**	**Час Кода**	**Lightbot**	**Kodable**
Режим использования	Не требует установки	Установка приложения	Установка приложения	Установка приложения	web	web	Web Установка приложения	Web Установка приложения
Год разработки	1967	2009	2006	2014	2013	2013	2008	2013
Перевод интерфейса на русский язык	Выполнен	Выполнен (но не всех слов)	Выполнен	Выполнен (но не всех слов)	Выполнен	Выполнен	Не выполнен	Выполнен
Поддержка платформ	Windows, Mac, Linux	Windows, Mac, Linux, Xbox	Windows, Mac, Linux	Windows	Все	Все	Web, Android, iOS	Web, Android, iOS
Бесплатная версия	Бесплатная версия	Бесплатная версия	Бесплатная версия	Бесплатно только 30 уровней	Бесплатная версия	Бесплатная версия	Только платная версия	В бесплатной версии всего 45 уровней
Создание игры	С нуля или использование готовых продуктов	С нуля или использование готовых продуктов	С нуля или использование готовых продуктов	Использование готовых продуктов	С нуля или использование готовых продуктов	Использование готовых продуктов	Использование готовых продуктов	Использование готовых продуктов
Разработка программного кода	Блочное программирование конкретного объекта	Блочное программирование конкретного объекта, события или создание дополнительных функций.	Блочное программирование конкретного объекта, события или создание дополнительных функций.	Блочное программирование конкретного объекта	Блочное программирование конкретного объекта, события или создание дополнительных функций.	Блочное программирование конкретного объекта	Блочное программирование конкретного объекта	Блочное программирование конкретного объекта
Графика	2D	3D	2D	2D	3D	2D	2D	2D

На основе данного сравнительного анализа была выбрана оптимальная среда программирования KODU для создания игр, а именно, 3D игр, для младших школьников. Рассмотрим более подробно особенности программирования в этой среде.

1.3. Особенности визуального языка программирования в среду Kodu Game Lab

Kodu Game Lab – это визуальный конструктор для разработки игровых приложений для персонального компьютера, а также для игровой консоли XBOX 360. Загрузить Kodu Game Lab можно с официального сайта (https://www.kodugamelab.com).

Kodu Game Lab не направлен на изучение какого-либо промышленного языка программирования, а обучает детей основам алгоритмизации и взаимодействию с компьютерами. Среда предоставляет инструментарий для проектирования компьютерных игр с помощью визуального интерфейса [20].

Ученик может создать свое игровое поле, добавить туда персонажей и задать для них логику поведения с помощью визуального языка.

Визуальный язык программирования Kodu рассчитан на создание 3D игр без знания самого языка программирования, для реализации чего в данном языке программирования имеется весь необходимый инструментарий, использование которого интуитивно понятно.

Для начального этапа изучения в данном языке программирования имеются несколько обучающих «миров» (поскольку из-за особенностей работы языка программирования назвать их задачами очень сложно), где поэтапно рассмотрено, как можно реализовать ту или иную модель действий или создать модель поведения в соответствии с ситуацией. На этапе программирования для каждого объекта или предмета можно задать свои свойства, свои настройки поведения и реакции на действия игрока, что позволяет настроить игру полностью под свою идею [21].

Основные отличительные признаки визуального языка программирования Kodu:

1. Низкий порог вхождения – за счет интуитивно понятного интерфейса и хорошего справочного материала, предоставляемого напрямую в программе, освоение языка дается очень легко.

2. Многообразие задач – так как этот язык программирования представлен в виде игры, на основе которой можно создавать другие игры, в игре можно смоделировать огромное множество задач, как похожих друг на друга, так и полностью различных между собой.

3. Уникальные способы освоения – программировать в данном языке можно не только при помощи стандартных устройств ввода (такие как клавиатура и мышь), но и при помощи контроллера, чего нет в других языках программирования.

4. Количество действий ограничивается только фантазией программиста. В каждой придуманной игре можно задать до 12 листов с действиями, переход между которыми возможен при определенных условиях, или использовать действия с другого листа, оставаясь в основном. Действия могут быть заданы и без условия, при активации программы действия применяются сразу по умолчанию.

5. Список задач в данной среде изучения программирования не ограничен обучающими примерами, задачи можно придумывать и реализовывать самостоятельно, что позволяет при помощи игры более глубоко разобрать способы организации действий и основы алгоритмов.

Программирование в данной среде происходит следующим образом (поэтапно):

1. Выбирается объект или персонаж для редактирования.
2. Открывается меню действий (программирование объекта).
3. Согласно условию (When) задаются действия (Do).
4. Выход в основное меню мира.
5. Запуск мира.

Поскольку язык программирования представляет собой игру, то, как и в любой игре, ученик может создать своего персонажа и задать ему

определенные характеристики (свойства и модель). Что примечательно, при длительном бездействии, персонажи игры пытаются привлечь к себе внимание – Пушка может перевернуться и улыбнуться, Kodu либо оборачивается по сторонам, либо начинает качаться вперед - назад.

В языке программирования используется визуальный подход к написанию программы, т.е. все действия задаются набором картинок, а не написанием большого количества текста со сложным синтаксисом. И все, что остается – задать этот набор картинок для всех используемых в создаваемой игре персонажей и объектов.

Глава 2. Разработка элективного курса

2.1. Описание элективного курса *«Создаем 3D игры с Kodu»*

Элективный курс *«Создаем 3D игры с Kodu»* является пропедевтическим курсом, который с помощью создания 3D игр позволит овладеть основами записи и выполнения алгоритмов.

Учащиеся начальных классов в возрасте от 7 до 12 лет, получат знания об алгоритмах и научатся строить простейшие программы в среде программирования Kodu Game Lab.

Курс рассчитан на один год обучения 1 раз в две недели, состоящий из 5 занятий (19 часов).

Цель курса – формирование логического и алгоритмического мышления.

Задачи: формирование умения строить простейшие алгоритмы и использовать их в учебных ситуациях; формирование навыков работы в среде программирования Kodu Game Lab.

Курс построен на основе практико-ориентированного подхода по принципу дидактической спирали [5, 22]:

- знакомство обучающихся с понятием или видом деятельности через выполнение практических заданий;
- более подробное изучение понятий или объектов с включением некоторых новых функций, свойств;
- применение изученных понятий на практике в заданиях творческого характера.

Планируемые образовательные результаты

- Предметные:
  - умение строить простейшие алгоритмы;
  - умение использовать изученные алгоритмы в учебных ситуациях.
- Личностные: формирование навыков участия в различных видах трудовой деятельности.
- Метапредметные:

- умение устанавливать причины успеха/неудач учебной деятельности;
- умение планировать действия по решению учебной задачи для получения результата;
- умение выбирать наиболее подходящий вариант решения задачи (на основе предложенных критериев).

Каждое занятие имеет определенную структуру:

1. Дидактическая беседа с учащимися на заданную тему, постановка образовательной задачи.

2. Решение поставленной задачи с помощью манипулирования с реальными объектами и дидактическими карточками.

3. Моделирование задачи с помощью среды Kodu Game Lab, которое может состоять из двух частей:

- *простое программирование:* содержит руководство и задания для учащихся, работая над которыми, учащиеся знакомятся с объектами и простыми понятиями программирования;
- *создание и редактирование ландшафтов:* при наличии времени можно использовать и вторую половину этого плана «Изменение мира» и на ее основе обсудить создание ландшафта.

4. Обсуждение полученных результатов.

Рекомендуется использовать следующие методы работы на занятиях:

- *Индивидуальная работа* используется при отработке основных навыков, а также при проведении самоконтроля;
- *Работа в парах* со сменой пар на каждом занятии для лучшего развития коммуникативных умений;
- *Работа в малых группах* для разработки творческих проектов;
- *Работа с дополнительными заданиями* или подсказками (для детей с особенными образовательными потребностями и одаренных детей);

- *“Сильные помогают слабым”* применяется в группах, где навыки и темп работы обучающихся различается;
- *Игра в игру друга;*
- *Взаимооценка.*

Перечень необходимых материально-технических ресурсов

Класс, оснащенный персональными компьютерами; мультимедийное оборудование; выход в глобальную сеть «Интернет»; среда программирования Kodu Game Lab; раздаточный материал (тексты сюжетов игр и алгоритмы создания).

Учебно-тематическое планирование

Предлагаемое планирование является примерным: учитель может корректировать содержание уроков и распределение часов на изучение материала в соответствии с уровнем подготовки обучающихся (таблица 2).

Учебно-тематическое планирование курса *«Создаем 3D игры с Kodu»*

Таблица 2. Описание занятий

№ п/з	Тема	Содержание	Кол-во часов
Занятие 1	Знакомство с визуальной средой программирования Kodu	Создание первой программы, создание игры для двух игроков, создание ландшафтов, задание для самостоятельной работы	3
Занятие 2	Новые возможности для перемещения объектов и персонажей - пути	Повторение пройденного, создание произвольного пути движения игрового объекта, создание клонов объектов, создание порожденных объектов, задание для самостоятельной работы	3
Занятие 3	Знакомство с визуальной средой программирования Kodu: подсчёт баллов, индикатор здоровья, объект таймер	Назначение времени действия игрового объекта, начисление баллов за действия объекта, использование индикатора уровня жизни, задание для самостоятельной работы	4
Занятие 4	Использование страниц в Kodu Game Lab.	Работа с несколькими страницами. Создание игры по	4

	Создание уникальных историй и персонажей	предложенному сценарию	
Занятие 5	Разработка собственной оригинальной игры от «А» до «Я»	Самостоятельное создание игры по предложенному плану	5

2.2. Тематическое планирование

Занятие 1. Знакомство с визуальной средой программирования Microsoft Kodu Game Lab

Цели занятия: познакомить с пользовательским интерфейсом Kodu; освоить приемы создания и редактирования миров; создать игры по предложенным алгоритмам.

Планируемые образовательные результаты:

1. Предметные: умение строить простейшие алгоритмы.

2. Метапредметные: умение планировать действия по решению учебной задачи для получения результата.

Краткое содержание занятия

Таблица 3. Краткое содержание занятия 1

Содержание основных этапов занятия	**Комментарий (как проводить)**
Этап 1. Установка ПО.	Можно пропустить этап установки, если Kodu уже установлен на ваши компьютеры в классе. Обратите внимание учащихся на то, где можно скачать среду и инструкцию по установке.
Этап 2. Знакомство с Microsoft Kodu Game Lab	Расскажите о запуске среды, основных элементах интерфейса. Проделайте первые шаги вместе с учащимися.
Этап 3. Выполнение Упражнение 1	Создание первой программы. *Сюжет игры:* персонаж Kodu облетает камни и ест красные яблоки. **Скринкаст подробного выполнения данного упражнения:** https://clck.ru/r4ZAd **Ссылка на готовое упражнение:** https://clck.ru/rdqhN **QR-код на выполненное упражнение** (рис. 9)**:**

	 Рис. 9. QR-код упражнение 1
Этап 4. Задание для самостоятельной работы	Учащиеся проводят эксперимент. В созданной программе предусмотрите столкновение Kodu с камнем, измените цвет яблок.
Этап 5. Выполнение Упражнений 2 и 3	Создание игры для двух игроков (Байкеры поедают яблоки, избегая столкновения с камнями). Проектирование ландшафта в игре. **Ссылка на готовое упражнение:** https://clck.ru/rdqmK **QR-код на выполненное упражнение 2** (рис. 10)**:**  Рис. 10. QR-код упражнение 2 **Ссылка на готовое упражнение: https://clck.ru/rdqop** **QR-код на выполненное упражнение 3** (рис. 11)**:**  Рис. 11. QR-код упражнение 3

Этап 6. Практическая работа с миром “Стрельба по рыбам”	Вместе с учащимися проделайте задание: добавьте объект Kodu, напишите программу для стрельбы, проведите эксперименты.
Этап 7. Подведение итогов	Рефлексивная беседа.

Этап 1. Установка программного обеспечения.

Как уже отмечалось, в каждом конкретном случае учитель принимает решение о необходимости установки программного обеспечения самими обучающимися. Это зависит как от объективных (технических условий), так и субъективных (возраст и навыки обучающихся) причин.

Если программная среда установлена, пожалуйста, обратите внимание на это еще раз, чтобы дома учащиеся смогли установить ее, например, вместе с родителями.

Этап 2. Знакомство с Microsoft Kodu Game Lab.

Запустим приложение «Kodu - лаборатория игр», щелкнув два раза левой кнопкой мыши по ярлыку данной программы (рисунок 12).

После запуска откроется окно приложения «Kodu - лаборатория игр», в котором впоследствии вы можете выбирать один из предложенных пунктов меню:

Рис. 12. Ярлык Kodu

- «Возобновить», для того чтобы продолжить конструирование и редактирование созданной вами ранее игры;
- «Загрузить мир», чтобы загрузить одну из предложенных игр, а также начать её редактирование;

- «Друзья», чтобы осуществить загрузку понравившихся уровней, уроков и примеров;
- «Параметры», для того чтобы воспользоваться настройками программы «Kodu - лаборатория игр»;
- «Помощь», чтобы получить справку по данному приложению;
- «Выйти», чтобы осуществить выход из программы.

Этап 3. Выполняем Упражнение 1.

Программа в Kodu — это набор правил, которые определяют действия объекта.

Для написания правил в Kodu используются два оператора:

When <условие> Do <действие>

When (англ. «когда», «если», «в то время как») - оператор, определяющий условие;

Do (англ. «делать») - оператор, определяющий непосредственное действие, которое должен выполнить объект при соответствующем условии.

Создание первой программы

Сюжет игры: персонаж Kodu облетает камни и ест красные яблоки.

Для успешного выполнения упражнения и создания первой игры, четко следуйте предложенному алгоритму:

1. Запустите программу Kodu.

2. Выберите команду Новый мир (New World). Появится зеленое поле основа для размещения игровых объектов в мире. Внизу окна размещены иконки, отображающие основные команды программы (рисунок 13).

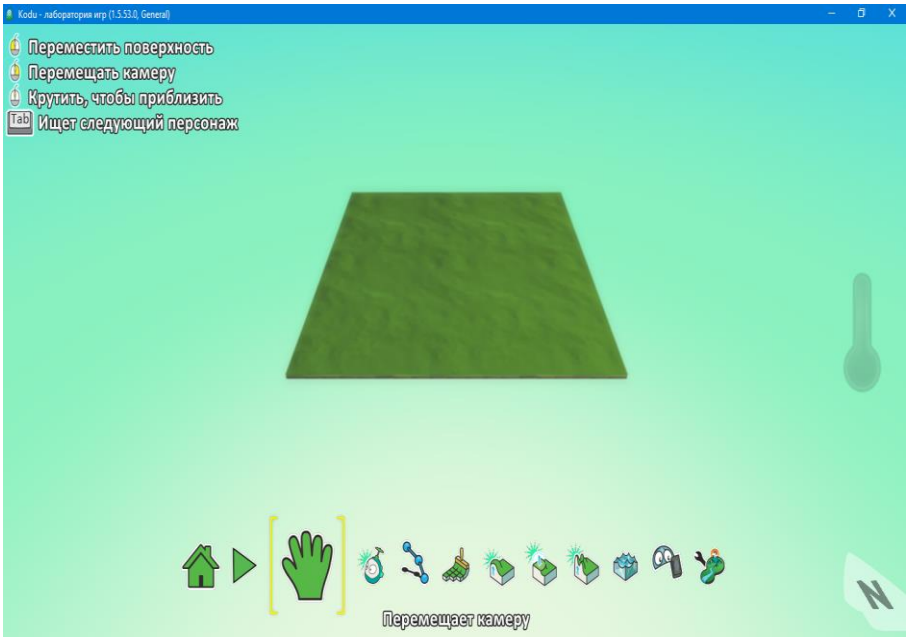

Рис. 13. Режим создания мира в Microsoft Kodu Game Lab

3. Добавьте на поле объект «камень».

Для добавления объектов выберите иконку **Объект** (рисунок 14) и однократно щелкните по полю левой кнопкой мыши. Появится список доступных объектов. Выберите объект камень, затем объект яблоко. Повторите действия ещё несколько раз раза.

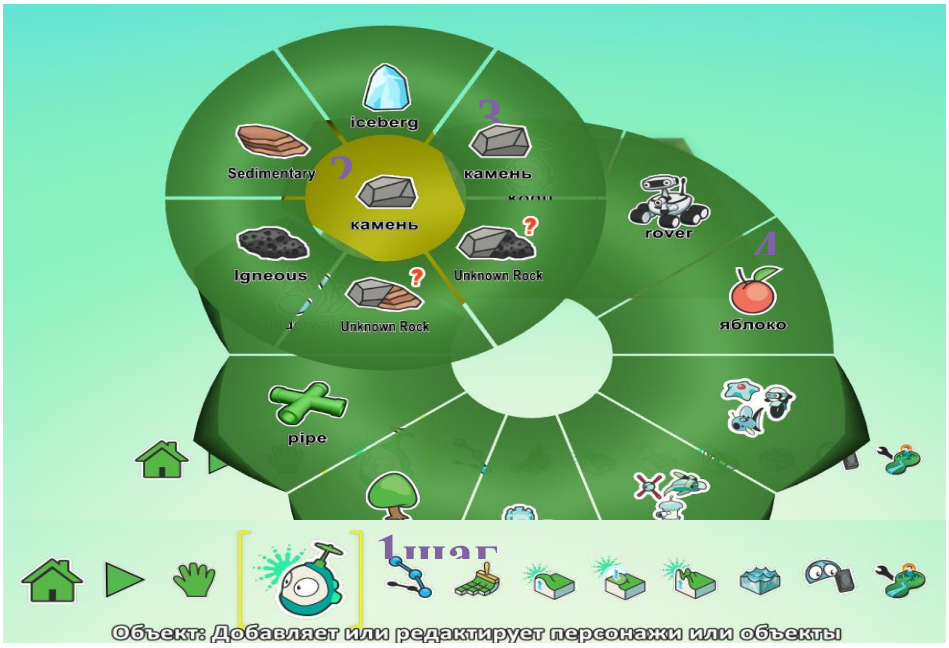

Рис. 14. Режим выбора объектов

4. Добавьте объект Kodu и задайте для него программу действий - набор правил.

Щелкнув **правой кнопкой мыши** на объекте, Kodu, вызовите контекстное меню и перейдите в режим создания **Программы** (рисунок 15).

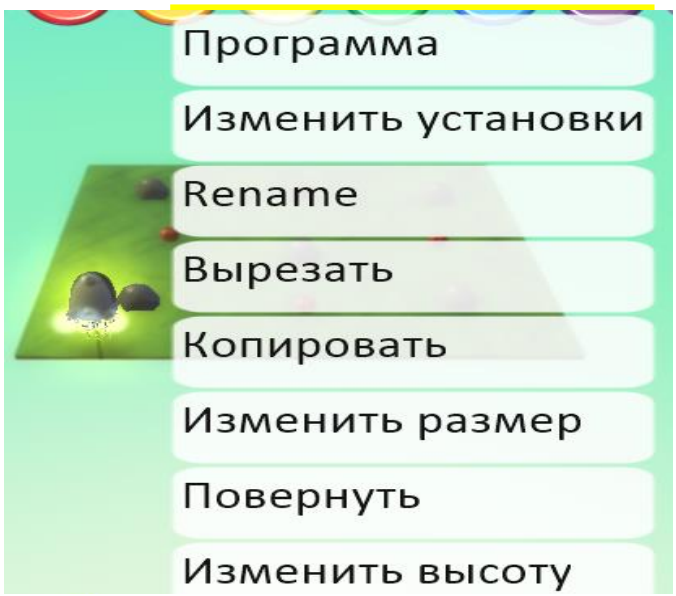

Рис. 15. Контекстное меню объекта

5. В открывшемся окне кода составьте инструкцию для движения между камней и съедания Kodu красных яблок. Команды задаются выбором из списка необходимых инструкций. Пример программы приведен на рисунке 16. При щелчке по карточке (иконка со знаком +) открывается перечень доступных действия, из которых следует выбрать необходимое (подтвердив действия нажатием левой клавиши мыши).

Рис. 16. Пример программы

6. Запустите программу на выполнение клавишей **Esc**. Понаблюдайте за движением **Kodu**. Если траектория движения не соответствуют поставленной задаче (объект не ест яблоки или не объезжает камни), то проверьте корректность кода (рисунок 16).

7.Сохраните программу на жестком диске. Для этого перейдите в главное меню (клавиша **Home**) и выберите команду **Сохранить мой мир**.

Этап 4. **Задание для самостоятельной работы:**

1. Попробуйте изменить цвет одного из яблок, например, на синий.

2. Напишите программу, в которой предусмотрите ситуацию столкновения Kodu с камнем.

Этап 5. Упражнение 2. Создание игры для двух игроков.

Сюжет игры: два **Байкера** поедают яблоки, избегая столкновения с камнями.

Для успешного выполнения упражнения и создания игры, четко следуйте предложенному алгоритму:

Создайте новый мир **(Empty World)**. Разместите в нем объекты: Байкер, яблоки, камень.

Напишите программу, которая позволит Байкеру свободно перемещается по территории мира. Замечая яблоко, Байкер должен двигаться к нему избегая столкновения с камнем (рисунок 17). Если байкер не будет уклоняться от камня. Что может произойти?

Рис. 17. Программа для перемещения и маневрирования

Измените правило управления Байкером. Для этого следует вырезать из программы строку 1 и 3 (щелчок правой кнопкой мыши на номере строки - вырезать) (рисунок 18).

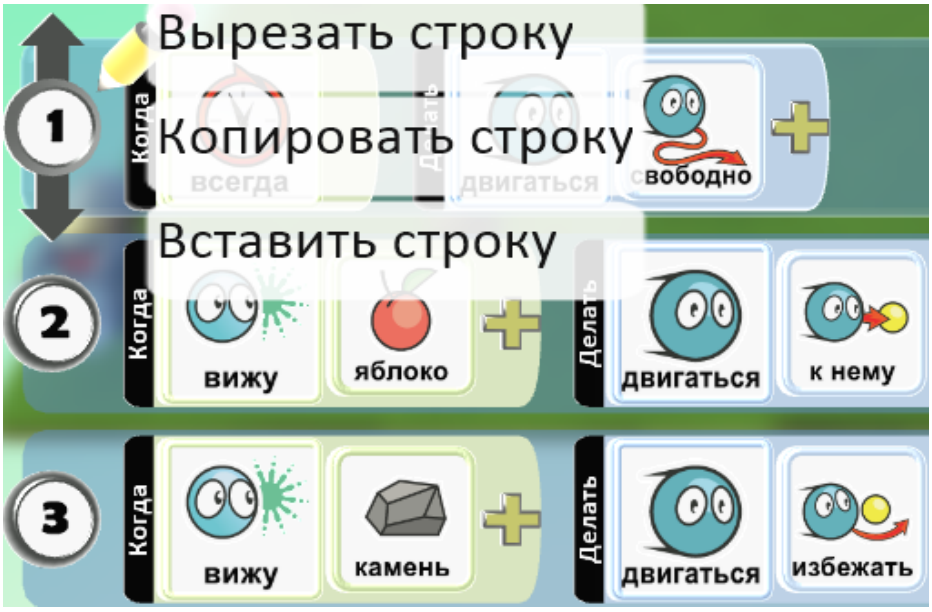

Рис. 18. Вырезание строки из программы

Измените, правило управления движением Байкера на управление при помощи клавиатуры вместо свободного перемещения (если нажимаются клавиши управления курсором, то Байкер должен двигаться), как это сделано на рисунке 19.

Добавьте правило поедания Яблок (когда Байкер коснется яблока, он должен его съесть). Пример кода приведен на рисунке 19.

Рис. 19. Управление Байкером с клавиатуры

Создайте второго Байкера, выполнив операцию копирования (используйте правую кнопку мыши).

Измените, цвет одного из Байкеров, чтобы их можно было отличить. Используйте для этого палитру цветов и клавиши управления курсором (рисунок 20).

Обратите внимание, что при копировании объекта копируется и его программа. Измените правило управления вторым Байкером в его программе. Для этого назначьте новые управляющие клавиши, например, стрелки.

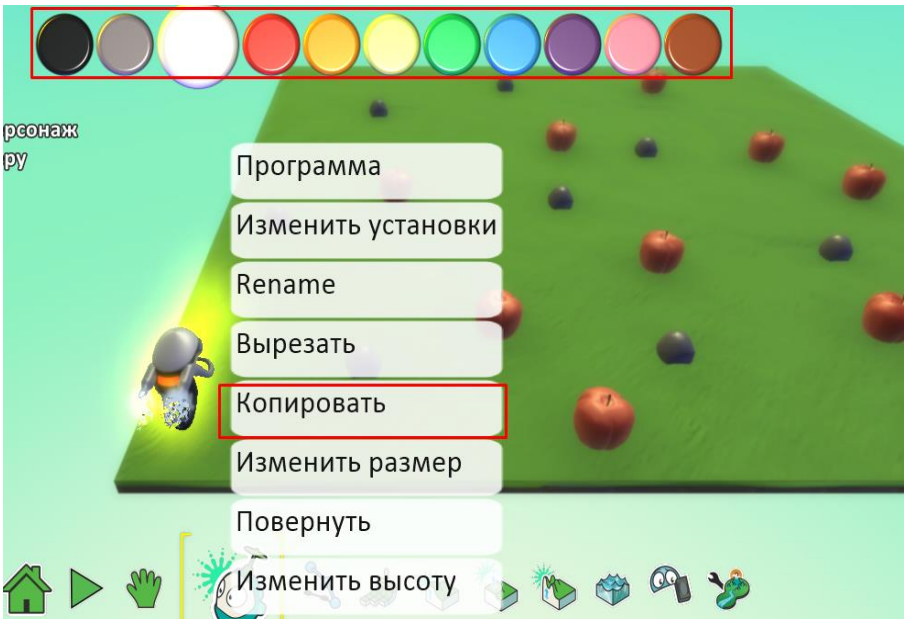

Рис. 20. Изменение цвета объекта

Теперь вы создали игру для двоих участников. Посоревнуйтесь с другом в поедании Яблок.

Сохраните игру, добавив необходимые теги. **Тег**, иногда тэг (англ. tag — ярлык, этикетка, бирка; метить) - здесь: ключевое слово, уточняющее действие. Теги в Kodu используются для того, чтобы объединить игры сходной тематики и лучше ориентироваться при поиске. Например, вы сможете отсортировать игры-приключения. Для созданной вами игры, где байкеры поедают яблоки, подойдут теги Приключения или Гонки (рисунок 21).

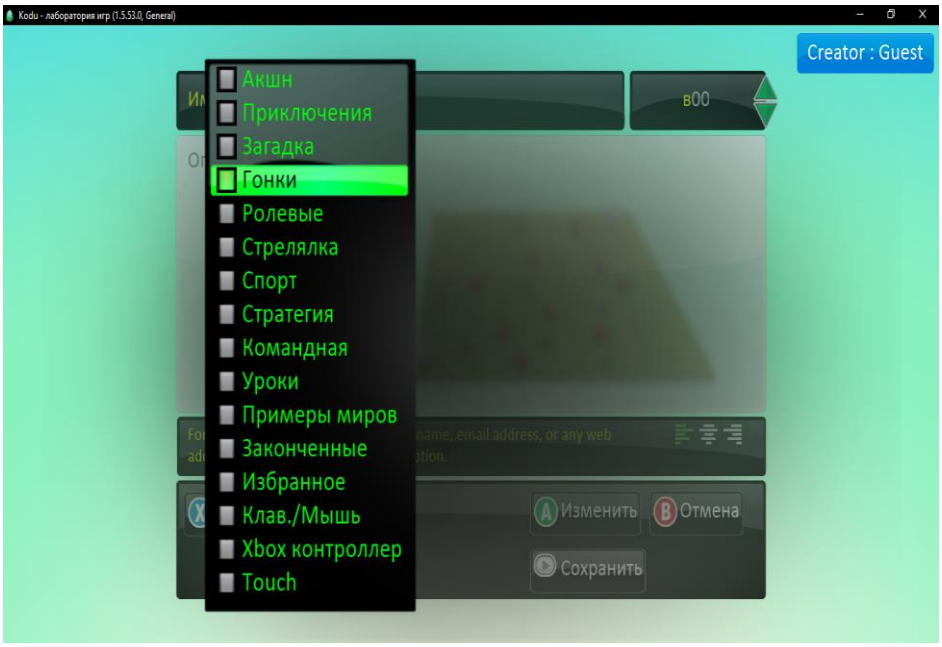

Рис. 21. Сохранение игры с добавлением тегов

Задание для самостоятельной работы:

Сделайте два яблока зеленого цвета. Измените, код Байкеров таким образом, чтобы они не ели зеленых яблок.

Добавьте объект Kodu и запрограммируйте его на поедание только зеленых яблок.

Упражнение 3. Создание ландшафтов.

В этом упражнении вы научитесь проектировать вид местности (ее ландшафт), в которой происходят действия игры.

Ландшафт — вид местности, от Land — земля и schaft — суффикс, выражающий взаимосвязь, взаимозависимость).

Рельеф (фр. relief, от лат. relevo — поднимаю) — совокупность неровностей суши, дна океанов и морей, разнообразных по очертаниям, размерам, происхождению, возрасту и истории развития.

Для успешного выполнения упражнения и создания игры, четко следуйте предложенному алгоритму:

1. Создайте новый пустой мир (Empty World).

2. Создайте ландшафт игрового мира в виде зеленой травы, небольших гор и холмов.

3. Для создания зеленого поля выберите инструмент Кисть для земли и произвольно нарисуйте на игровом поле землю.

4. Цвет земли выберите под номерами 15 или 54.

5. Поднимите отдельные участки земли, сделав из них холмы и горы.

6. Для создания неровностей используйте инструмент Неровности. Добавьте воду в созданный ландшафт при помощи инструмента Вода, цвет воды под номером 8.

7. Измените цвет неба при помощи инструмента Параметры мира.

8. Включите волны для воды.

Инструмент **Параметры мира** позволяет сделать игровой мир **Kodu** более профессиональным. Экспериментируя с различными настройками (включая, отключая или изменяя значения параметров), можно добиться профессиональных эффектов в своей игре.

Рис. 22. Инструменты параметров

Нарисуйте облака на небе. Пример ландшафта приведен на рисунке 23.

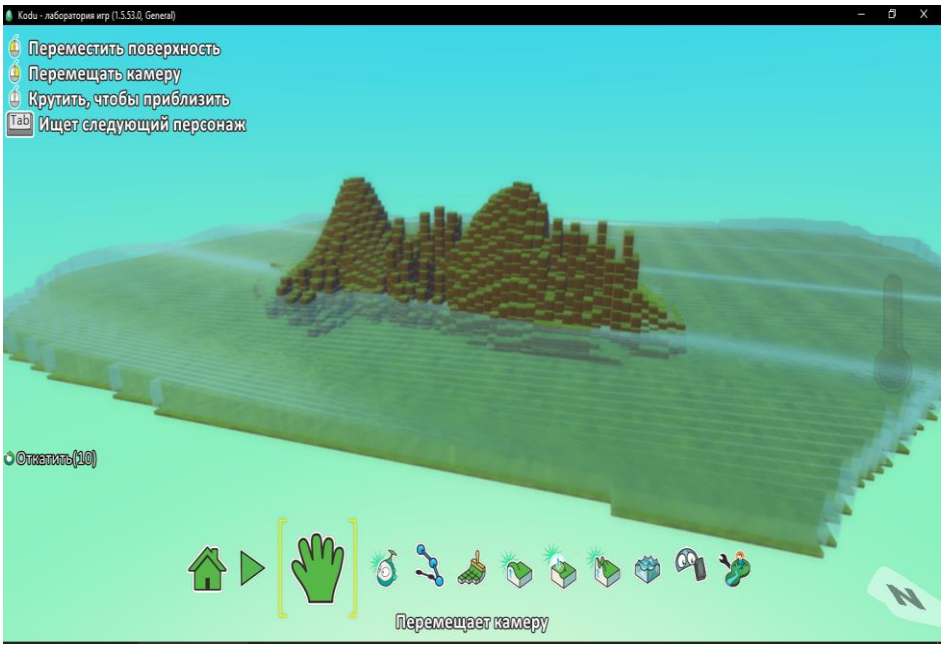

Рис. 23. Пример ландшафта

Сохраните мир под именем Ландшафт и запустите игру на выполнение. Обратите внимание на движение воды и набегающие волны.

Задание для самостоятельной работы:

Создайте в Kodu модель местности, используя кисть и её различные типы, а также увеличение и уменьшение размера кисти. Добавьте на созданном рельефе холмы и впадины, используя соответствующий инструмент. Постройте стены на моделируемой местности при помощи волшебной кисти. Добавьте несколько дорожек на создаваемой территории. Измените модель местности, добавив водоемы, парки, здания.

Этап 6. Практическая работа с миром «Стрельба по рыбам (Shooting fish)»

1. Откройте игровой мир **«Стрельба по рыбам (Shooting fish)».**

2. Добавьте объект **Kodu** на игровое поле.

Напишите программу для **Kodu**, с помощью которой можно будет стрелять по рыбам ракетами в разных направлениях. Проведите эксперимент: сравните действие ракет и пулек. Сделайте вывод о том, какое оружие эффективнее. Изучите различные варианты стрельбы. Настройте стрельбу пульками случайных цветов, вылетающих после каждого нажатия на пробел. Пройдите игру, используя стрельбу ракетами. За какое время это у вас получилось? Сохраните созданный мир.

Этап 7. Подведение итогов

Провести ситуационную рефлексию. Обсудить вместе с учащимися, что усвоили на данном занятии. Что получилось лучше всего? А что ещё необходимо повторить?

Занятие 2. Новые возможности для перемещения объектов и персонажей - пути. Создание клонов и порождаемых объектов. Опция Родитель *(Приложение 1)*

Цели занятия: познакомить с возможностями объектов; освоить принципы использования клонов и порождаемых объектов; создать игры по предложенным алгоритмам.

Планируемые образовательные результаты:

1. Предметные: умение строить простейшие алгоритмы.

2. Метапредметные: умение планировать действия по решению учебной задачи для получения результата.

Краткое содержание занятия

Таблица 4. Краткое содержание занятия 2

Содержание основных этапов занятия	**Комментарий (как проводить)**
Этап 1. Актуализация и повторение пройденного	Попросите учащихся вспомнить, что они делали на предыдущем занятии, с какими элементами Kodu они уже знакомы. Узнайте, удалось ли установить Kodu на домашнем ПК. Расскажите об объектно-ориентированном программировании и целях занятия. Предложите исправить ошибку в коде для Байкера.
Этап 2. Выполнение Упражнения 1	Создайте произвольный путь движения игрового объекта. **Ссылка на готовое упражнение: https://clck.ru/rdqt2** **QR-код на выполненное упражнение** (рис. 24)**:** Рис. 24. QR-код упражнение 1
Этап 3. Задание для самостоятельной работы	Предложите наиболее успешным учащимся самостоятельно перейти на работу с дополнительными заданиями после прохождения предыдущего этапа. Проведите эксперименты с мирами Tutorial 2, 3D Flare Paths. В итоге, обсудите, каким образом используются пути и наклонные поверхности и как они влияют на активность действий в игре?
Этап 4. Выполнение Упражнения 2	Создание клонов объектов. Сюжет игры: три Аэростата стреляют по Самолету. **Ссылка на готовое упражнение: https://clck.ru/rdqtz**

	QR-код на выполненное упражнение (рис. 25)**:**  Рис. 25. QR-код упражнение 2
Этап 5. Выполнение Упражнения 3	Создание порожденных объектов. *Сюжет игры:* новая эскадрилья Аэростатов стреляет по самолету. **Ссылка на готовое упражнение: https://clck.ru/rdqv8** **QR-код на выполненное упражнение** (рис. 26)**:**  Рис. 26. QR-код упражнение 3
Этап 6. Опция Родитель. Выполнение упражнения 4	Включение опции Родитель. Вместе с учащимися проделайте задание: создайте новый объект Летающая рыба, напишите программу для стрельбы камня ракетами по появляющимся рыбам. **Ссылка на готовое упражнение: https://clck.ru/rdqvh** **QR-код на выполненное упражнение** (рис. 27)**:**  Рис. 27. QR-код упражнение 4

Этап 7. Подведение итогов	Порассуждайте вместе с учащимися над вопросами. Обозначьте достижения группы и перспективы.

Занятие 3. Знакомство с визуальной средой программирования Kodu: подсчёт баллов, индикатор здоровья, объект таймер *(Приложение 2)*

Цели занятия: освоить написание программ (сценария игры) с использованием дополнительных возможностей; создать игры по предложенным алгоритмам.

Планируемые образовательные результаты:

1. Предметные: умение строить простейшие алгоритмы.
2. Метапредметные: умения планировать действия по решению учебной задачи для получения результата; умение выбирать наиболее подходящий вариант решения задачи (на основе предложенных критериев).

Краткое содержание занятия

Таблица 5. Краткое содержание занятия 3

Содержание основных этапов занятия	**Комментарий (как проводить)**
Этап 1. Актуализация и повторение пройденного	Выясните ожидания учащихся от занятия. Попросите учащихся сделать предположение о том, для чего может увеличиваться/уменьшаться счёт игре, а также, почему иногда (и для каких функций?) необходимо использовать таймер.
Этап 2. Выполнение Упражнения 1	Знакомство с таймером и подсчётом очков. *Сюжет игры:* Завод выпускает мячи каждые 10 секунд и монеты каждые 5 секунд. Чётко выполнив предложенный алгоритм, ребята смогут сформулировать назначение таймера и счёта в игре. После проведения исследований со случайным временем работы таймера обсудите возможности генерации случайных чисел. **Ссылка на готовое упражнение: https://clck.ru/rdqxL** **QR-код на выполненное упражнение** (рис. 28)**:**

	 Рис. 28. QR-код упражнение 1
Этап 3. Выполнение Упражнения 2	Начисление баллов за действия объекта *Сюжет игры:* за каждый мяч Завод получает 10 баллов, за выпуск монеты - 1 балл. **Ссылка на готовое упражнение: https://clck.ru/rdqya** **QR-код на выполненное упражнение** (рис. 29)**:**  Рис. 29. QR-код упражнение 2
Этап 4. Задание для самостоятельной работы	Попробуйте изменить код программы так, чтобы за каждый мяч начислялось 20 баллов, а за каждую монету 5. Напишите программу, которая позволяет остановить игру, если набрано 100 баллов.
Этап 5. Выполнение Упражнения 3	Использование индикатора уровня жизни. *Сюжет игры:* Объект Kodu теряет здоровье при поедании злёных яблок. При низком уровне здоровья объект Kodu светится красным. **Ссылка на готовое упражнение: https://clck.ru/rdqz2** **QR-код на выполненное упражнение:** (рис. 30)

	 Рис. 30. QR-код упражнение 3
Этап 6. Подведение итогов	Порассуждайте вместе с учащимися над вопросами. Обозначьте достижения группы и перспективы.

Занятие 4. Использование страниц в Kodu Game Lab. Создание уникальных историй и персонажей

Цели занятия: освоить написание программ (сценария игры) с использованием нескольких страниц; создать игры по предложенным алгоритмам.

Планируемые образовательные результаты:

1. Предметные: умение строить простейшие алгоритмы и использовать изученные алгоритмы в учебных ситуациях.

2. Метапредметные: умение устанавливать причины успеха/неудач учебной деятельности.

Краткое содержание занятия

Таблица 6. Краткое содержание занятия 4

Содержание основных этапов занятия	**Комментарий(как проводить)**
Этап 1. Актуализация и повторение пройденного	Попросите учащихся вспомнить, что они уже знают и умеют делать в Kodu. Объясните, что игры можно усложнить, используя многостраничные сценарии.
Этап 2. Выполнение Упражнения 1	Работа с несколькими страницами. Сюжет игры: Завод выпускает сначала мячи, потом монеты, а потом ракеты. **Ссылка на готовое упражнение:**

	https://clck.ru/rdqzd **QR-код на выполненное упражнение** (рис. 31)**:**  Рис. 31. QR-код упражнение 1
Этап 3. Задание для самостоятельной работы	Предложите учащимся модернизировать созданную игру.
Этап 4. Вспомогательные алгоритмы и подпрограммы	Вместе с учащимися ответьте на предлагаемые вопросы.
Этап 5. Выполнение Упражнения 2	*Создание игры по предложенному сценарию.* Спортивный теннисный клуб планирует познакомить учеников начальных школ вашего города с игрой в теннис. Они наняли вас для проектирования игры, чтобы увлечь ребят теннисом и повысить популярность этого вида спорта.
Этап 6. Подведение итогов	Рефлексивная беседа.

Занятие 5. Разработка своей оригинальной игры

Цели занятия: разработать и представить оригинальную игру; познакомиться с принципами групповой работы.

Планируемые образовательные результаты:

1. Предметные: умение строить простейшие алгоритмы и использовать изученные алгоритмы в учебных ситуациях.

2. Метапредметные: умение планировать действия по решению учебной задачи для получения результата; умение выбирать наиболее подходящий вариант решения задачи (на основе предложенных критериев).

3. Личностные: формирование навыков участия в различных видах трудовой деятельности.

Краткое содержание занятия

Таблица 7. Краткое содержание занятия 5

Содержание основных этапов занятия	**Комментарий (как проводить)**
Этап 1. Актуализация	Подведите учащихся к мысли о том, что они уже готовы разработать собственную игру в Kodu от “А”до “Я”. Подготовьте группу к разработке проектов.
Этап 2. Выбор жанра	Рассказать о видах жанра и их выборе.
Этап 3. Проектируем сюжет игры.	Поясните, что такое сюжет; предложите создать эскиз, укажите, что в игре обязательно должны быть главные и второстепенные герои, основной противник, препятствия.
Этап 4. Детализируем цель игры и миссии персонажей	Определить цель проектируемой вами игры и миссию игровых персонажей (героев и объектов) в процессе достижения этой цели.
Этап 5. Создание игры	Создаем игровой мир, ландшафт, атмосферу согласно нашему сюжету. Добавляем персонажей. Программируем действия героев (персонажей, объектов) согласно разработанным правилам и сюжету игры. Проводим тестирование игры.
Этап 6. Презентация игры	Организуйте презентацию разработанных проектов и их обсуждение.
Этап 7. Подведение итогов занятия и комплекса учебных занятий	Рефлексивная беседа

В результате исследования был разработан элективный курс «*Создаем 3D игры с Kodu*» для младших школьников, а также описаны методические рекомендации по подготовке педагога и компьютерного класс к реализации курса.

Познакомиться с курсом в pdf-формате можно по следующей ссылке: https://clck.ru/rdu5U

QR-код на элективный курс (рис. 32):

Рисунок 32. QR-код на элективный курс *«Создаем 3D игры с Kodu»*

2.3. Методические рекомендации и апробация разработанных материалов

Подготовка педагога к реализации курса

Перед началом проведения элективного курса преподавателю, рекомендуется самому порешать все занятия, и создать свою игру. Это необходимо для лучшего понимания среды программирования для того, чтобы избежать ситуаций, когда педагог не сможет помочь обучающимся при самостоятельной работе.

Рекомендуется начать проверку с самого «нуля», то есть с установки программного обеспечения на различных платформах, а также персональных компьютеров с различными операционными системами. Если преподаватель ограничен в техническом оснащении, рекомендуется использования программного обеспечения для создания виртуальных машин, например, Oracle VM VirtualBox.

Предлагается создание коллекции своих примеров. Примеры необходимы на осваиваемые возможности среды, а также объединены с уже накопленным обучающимися опытом.

Педагог может записать свои объяснения на диктофон, прослушать себя и проверить свою речь на корректность использования терминологии,

понятности объяснений для младших школьников, порядок и уверенности изложения курса, эмоционального окрашивания речи.

Постараться заранее ознакомиться с подготовкой обучающихся.

В случае отсутствия такой возможности провести входную анкету для обучающихся. Это нужно для корректировки содержания курса и стиля общения к конкретной аудитории.

Важно, что от подготовки педагога, его организованности и доброжелательности зависит успешность занятий.

Подготовка компьютерного класса

Педагог решает эту задачу с учетом условий площадки, на которой проводится курс.

Приветствуется, когда обучающиеся получают опыт установки программного обеспечения на первом занятии. Это позволяет проделать и обсудить все нюансы, касающиеся скачивания и установки Kodu.

Если среда уже установлена на рабочих станциях обучающихся, необходимо реализовать это в демонстрационном режиме. Сконцентрировать внимание, где можно скачать среду.

Рекомендуется записать или раздать листовки с адресом сайта, на котором доступна ссылка на скачивание Kodu и инструкцией, чтобы учащиеся могли установить Kodu дома (например, вместе с родителями).

В процессе установки могут возникнуть сложности следующего характера:

- может возникнуть сообщения о том, что Kodu не будет корректно работать на данном компьютере (рисунок 33);

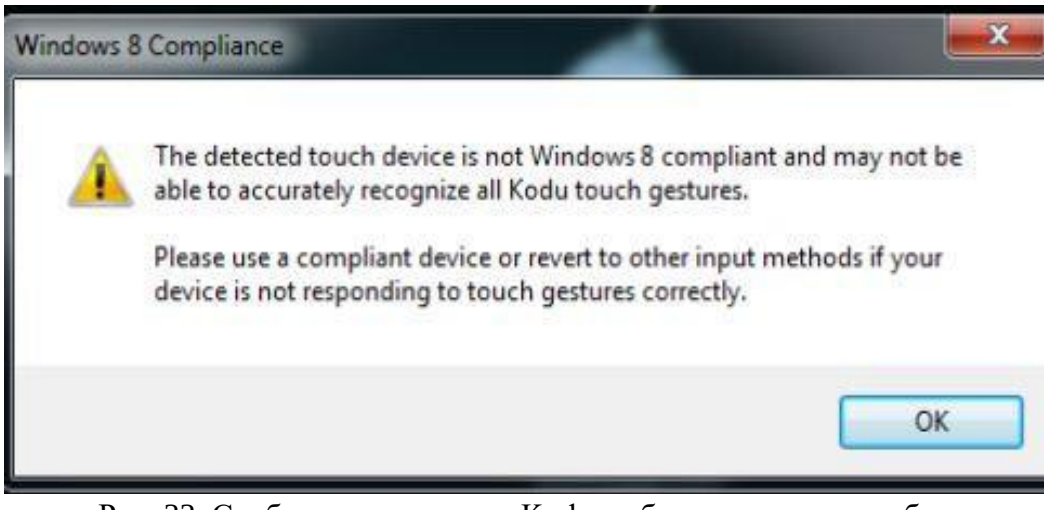

Рис. 33. Сообщения о том, что Kodu не будет корректно работать

Решение: После установки обязательно запустите среду и проверьте доступность всех функций.

Если некоторые функции недоступны, следует проверить рабочую станцию на вирусы, либо использовать другой персональный компьютер, возможно, что даже после такого сообщения программа установится, и будет работать корректно.

- Вместе с визуальной средой Kodu может установиться подозрительное программное обеспечение.

Решение: отменить установку среды, и проверить адрес сайта, на котором доступна, ссылка на скачивание Kodu. Более вероятно, что вы скачали дистрибутив не с официального сайта.

Апробация разработанных материалов

Апробация разработанного курса по разработке 3D игр с использованием визуального языка программирования Kodu проводилась в рамках преддипломной практики в МОУ «Килачевская СОШ» Свердловской области с обучающимися с возрасте 8-11 лет.

За период практики были проведены 3 занятия курса *«Создаем 3D игры с Kodu»* (10 часов). Занятия прошли успешно, детям нравилось заниматься в среде программирования Kodu Game Lab, материал был освоен. Обучающиеся, начиная со второго занятия, создавали простейшие алгоритмы и использовали их в учебных ситуациях.

Всего в апробации участвовало две группы по 7 и 8 человек. Результат апробации помог оценить эффективность курса, а также дополнить в учебно-тематическом планировании количество часов на каждую тему.

Заключение

В соответствии с целью и задачами, сформулированными в выпускной работе, были:

- изучены особенности обучения программированию обучающихся младших классов:
  - профессор Иван Калаш в своих работах регулярно отмечает не только важность раннего обучения информатике и программированию, но и высокий статус информатики как учебной дисциплины в программе подготовки младших школьников;
  - обучение программированию лучше организовать в ходе решения задач, подобранных в специально выстроенной последовательности;
  - лучшим способом для предоставления информации на уроках информатики младшим школьникам является игровая форма;
  - для изучения основ алгоритмизации школьникам в возрасте от 7 до 12 лет (с 1 по 5 классы) отлично подойдут уроки информатики с использованием конструктора игр.
- проведен сравнительный анализ сред визуального программирования по созданию игр по следующим критериям: режим использования, год разработки, перевод интерфейса на русский язык, поддержка платформ, бесплатная версия, создание игры, разработка программного кода, графика; на основе данного сравнительного анализа была выбрана оптимальная среда программирования KODU для создания 3D игр для младших школьников;
- рассмотрены особенности программирования в среде Kodu Game Lab:
  - Kodu Game Lab обучает детей основам алгоритмизации и взаимодействию с компьютерами;
  - рассчитан на создание 3D игр без знания самого языка программирования, для реализации чего в данном языке

программирования имеется весь необходимый инструментарий, использование которого интуитивно понятно.

- разработан элективный курс *«Создаем 3D игры с Kodu»*, состоящий из 5 занятий и рассчитан на один год обучения 1 раз в две недели;
- проведена апробация разработанных материалов в МОУ «Килачевская СОШ» Свердловской области;
- написана статья «Формирование метапредметных результатов обучения у школьников 5-6 классов на уроках по программированию в средах Kodu и Scratch», опубликованная в сборнике «Актуальные вопросы преподавания математики, информатики и информационных технологий», 2022 год.

Таким образом, можно утверждать, что цель работы достигнута, задачи выполнены в полном объеме.

Список использованных источников

1. 5 простых шагов к созданию 3D игр вместе с KODU / О.Ф. Брыскина [и др.]. М., 2013. 51 с. URL: https://clck.ru/gvZ2y (дата обращения: 15.03.2022).

2. Абдуразаков М. М., Зенкина С. В., Ниматулаев М. М. Как реализовать основную образовательную программу на основе трех требований федерального государственного образовательного стандарта // Информатика и образование. 2019. № 10. С. 5-13. URL: https://www.elibrary.ru/item.asp?id=41534797 (дата обращения: 17.05.2022).

3. Акимова И. В., Губанова О. М., Титова Н. В. Изучение элементов логического программирования в рамках факультатива по информатике // Современные проблемы науки и образования. 2018. № 4. С. 38-47. URL: https://www.elibrary.ru/item.asp?id=36344847 (дата обращения: 05.12.2021).

4. Алейникова О. М. Методика преподавания непрерывного курса алгоритмизации в общеобразовательной школе // Известия РГПУ им. А.И. Герцена. 2007. № 45. С. 311-314. URL: https://www.elibrary.ru/item.asp?id=12833297 (дата обращения: 10.04.2022).

5. Батура В. К., Чубрикова О. В. Разработка учебного пособия Kodu game lab // Интеллектуальный потенциал Сибири: сборник науч. тр. Рег. науч. студ. конф. Новосибирск, 2019. С. 313-315. URL: https://www.elibrary.ru/item.asp?id=41410784 (дата обращения: 14.11.2021).

6. Белова Г. В. Программирование в среде ЛОГО. Первые шаги М., 2007. 215 с. URL: https://clck.ru/gvYRQ (дата обращения: 15.03.2022).

7. Босова Л. Л., Сорокина Т. Е. Методика применения интерактивных сред для обучения младших школьников программированию // Информатика и образование. 2014. № 7 (256). С. 61-68. URL: https://elibrary.ru/item.asp?id=22589444 (дата обращения: 04.12.2021).

8. Брыксина О. Ф. Внеурочная деятельность в условиях ФГОС. Визуальное программирование в Kodu: первый шаг к ИТ-образованию //

Информатика и образование. 2013. № 1 (250). С. 33-39. URL: https://clck.ru/gvYWy (дата обращения: 11.10.2022).

9. Булгакова Н. Н. Активизация учебно-познавательной деятельности младших школьников на уроках информатики. СПб., 2004. 482 с.

10. Визуальное программирование в Microsoft Kodu Game Lab: первый шаг к ИТ-образованию / О.Ф. Брыскина [и др.]. М., 2014. С. 33-39. URL: https://elibrary.ru/item.asp?id=21541160 (дата обращения: 14.01.2022).

11. Выготский Л. С. Мышление и речь. М., 1996. 414 с.

12. Вычисления. Алгоритмизация. Программирование: пособие для учителя / под ред. М. П. Лапчик. М., 1988. 208 с.

13. Гермашев И. В., Данильчук Е. В., Куликова Н. Ю. Методические особенности формирования готовности будущего учителя информатики к разработке и использованию компьютерных игр в обучении алгоритмизации и программированию // Известия ВГПУ. 2018. № 5 (128). С. 42-49. URL: https://clck.ru/gvYd7 (дата обращения: 10.02.2022).

14. Горячев А. В., Суворова Н.И. Информатика «Логика и алгоритмы». М., 2015. 95 с.

15. Даниелян С. Д. Разработка элективного курса языка программирования Kodu для младших школьников // Вопросы педагогики. 2022. № 4-2. С. 80-83. URL: https://www.elibrary.ru/item.asp?id=48443622 (дата обращения: 12.04.2022).

16. Данильчук Е.В., Куликова Н. Ю., Малова А. И. Онлайн – курс «Разработка компьютерных игр для мобильных устройств» при обучении школьников алгоритмизации и программированию // Известия ВГПУ. 2021. №6 (159). С. 38-44. URL: https://www.elibrary.ru/item.asp?id=46511216 (дата обращения: 12.03.2022).

17. Дидактические аспекты организации факультативов [Электронный ресурс]. URL: https://clck.ru/gvh8g (дата обращения: 04.10.2021).

18. Ерондаева Ю. В., Кокшарова В. Е., Рожина И. В. Формирование метапредметных результатов обучения у школьников 5-6 классов на уроках

по программированию в средах Kodu и Scratch // Актуальные вопросы преподавания математики, информатики и информационных технологий: межвузовский сборник научных работ. 2022. С. 284-290. (дата обращения: 02.05.2022).

19. Жемчужников Д. Г. Создание компьютерных игр как средство обучения школьников программированию // Информатика и образование. 2012. № 8. С. 49-51. URL: https://elibrary.ru/item.asp?id=18251079 (дата обращения: 04.12.2022).

20.Ильина Т. Ю. Методика проблемного обучения информатике младших школьников // Информатика в школе. 2012. № 8. С. 70-72. URL: https://www.elibrary.ru/item.asp?id=18251205 (дата обращения: 30.03.2022).

21. Ионова Т. А. Создание игр в Kodu game lab // Информационные технологии в образовательном процессе вуза и школы: материалы Всер. XV науч.-прак. конф. Воронеж, 2021. С. 186-189. URL: https://www.elibrary.ru/item.asp?id=46300629 (дата обращения: 14.11.2021).

22. Каплан А. В., Павлов Д.И. Разработка методических подходов к реализации пропедевтического курса информатики в начальной школе средствами Kodu Game Lab // Информатика и образование. 2019. № 8. С. 14-23. URL: https://www.elibrary.ru/item.asp?id=41264946 (дата обращения: 14.04.2022).

23. Куликова Н. Ю., Малова А. И. Использование визуальных сред разработки компьютерных игр при обучении алгоритмизации и программированию // Образование и проблемы развития общества: сборник науч. ст. Междунар. науч. конф. Курск, 2019. С. 18-21. URL: https://elibrary.ru/item.asp?id=41205251 (дата обращения: 14.11.2021).

24. Куликова Н. Ю., Пономарева Ю. С. Возможности интерактивных сетевых средств при обучении информатике и ИКТ в школе // Математика. Информатика. Образование. 2020. № 2 (18). С. 96-106. URL: https://elibrary.ru/item.asp?id=43060407 (дата обращения: 09.10.2021).

25. Куличкова А.Г. Информатика. Волгоград, 2015. 125 с.

26. Лесневский А. С. Объектно-ориентированное программирование для начинающих. М., 2005. 232 с. URL: https://booksee.org/book/487088 (дата обращения: 06.12.2021).

27. Малова А.И. Обучение основам алгоритмизации и программирования в процессе создания школьниками игр для мобильных устройств // Студенческий электронный журнал СтРИЖ. 2021. № 2-1 (37). С. 142–145. URL: https://www.elibrary.ru/item.asp?id=45428377 (дата обращения: 10.02.2022).

28. Редькина А. В. Обучение синтезу алгоритмов // Вестник СибГАУ. 2008. №1 С. 30-34. URL: https://elibrary.ru/item.asp?id=10227713 (дата обращения: 10.10.2021).

29. Селевко Г. К. Педагогические технологии на основе информационно-коммуникационных средств. М., 2005. 208 с.

30. Селевко Г. К. Педагогические технологии на основе эффективности управления и организации учебного процесса. Компьютерные (новые информационные) технологии обучения. М., 2004. 556 с.

31. Софронова Н. В. Теория и методика обучения информатике. М., 2004. 145 с.

32. Ступеньки к информатике: учеб. для 3 кл. общеобразоват. учеб. заведений / Г. В. Ломаковская [и др.]. М., 2013. 160 с. URL: https://clck.ru/giHVa (дата обращения: 05.02.2022).

33. Теория и методика обучения информатике / М. П. Лапчик [и др.]. М., 2008. 592 с.

34. Ткаченко В. А. О выборе конструкторов игр для использования в программах дополнительного образования детей // Вестник НВГУ. 2011. №3. С. 69-74. URL: https://www.elibrary.ru/item.asp?id=17728367 (дата обращения: 15.04.2022).

35. Федеральный государственный образовательный стандарт. М., 2021. 124 с. URL: https://www.garant.ru/products/ipo/prime/doc/400807193/ (дата обращения: 12.02.2022).

36. Хузеева Ф. Ф. Среды программирования в обучении детей младшего возраста // Скиф. вопросы студенческой науки. 2021. № 1. С. 292-295. URL: https://www.elibrary.ru/item.asp?id=44679523 (дата обращения: 28.01.2022).

37. Цветкова М.С. Информатика 3-4 класс. М., 2013. 68 с.

38. Шелестович А. А. Kodu Game Lab как способ развития базовых основ программирования в игровой форме у школьников младшего и среднего возраста // Информатизация образования. 2016. № 1. С. 42-46. URL: https://www.elibrary.ru/item.asp?id=32844255 (дата обращения: 12.04.2022).

39. Шиянова Ю. В. Методика преподавания программирования в школе // Актуальные проблемы гуманитарных и естественных наук. 2014. №12-2. С.144-146. URL: https://www.elibrary.ru/item.asp?id=22814533 (дата обращения: 15.11.2021).

40. Энциклопедия языков программирования. URL: http://progopedia.ru/implementation/squeak/ (дата обращения: 17.10.2021).

41. Kodu Game Lab. 3D game programming for kids. URL: http://www.kodugamelab.com/downloads/ (дата обращения: 02.06.2022).

42. Medveckis A. Enhancement of educators' digital competences in the acquisition programming fundamentals in programming environment scratch // World Journal on Educational Technology, 2021. №4. С. 934-946. URL: https://www.sciencegate.app/keyword/156691 (дата обращения: 06.04.2022).

43. Mladenović M. Comparing Elementary Students' Programming Success based on Programming Environment // International Journal of Modern Education and Computer Science, 2016. № 8. С. 1-10. URL: https://www.elibrary.ru/item.asp?id=44907094 (дата обращения: 04.05.2022).

44. Scratch и Kodu. URL: https://www.yanikova.com/single-post/2017/07/12/scratch-и-kodu-сходства-и-отличия (дата обращения: 15.04.2022).

Приложения

Приложение 1

Занятие 2. Новые возможности для перемещения объектов и персонажей - пути. Создание клонов и порождаемых объектов. Опция Родитель

На этом занятии познакомимся с тем, как создаются пути движения персонажей, научимся создавать клоны и порождения объектов, а также узнаем, как научить объект« говорить». Опция Родитель познакомит с объектно-ориентированным программированием, которое используется для создания сложных программных систем.

Объектно-ориентированное программирование (ООП) – это программирование, в котором основными понятиями являются объекты и классы. При таком программировании создается класс (например,— офис), в который входят различные объекты (например,— кабинет). При этом каждый объект имеет общие характеристики, присущие всему классу (стены, пол, потолок, окно, дверь), но может приобретать и индивидуальные черты(например: кресло, стол, персональный компьютер и тому подобное.).

Этап 1. Актуализация и повторение пройденного.

При изучении упражнений занятия 1 изучили написание кодов для объектов игрового мира. Вспомним пройденный материал.

В задании к самостоятельной работе упражнения 2 вам надо было добавить в игровой мир яблоки зеленого цвета и написать код, который запрещал Байкерам есть зеленые яблоки. На рис.1 представлен код с ошибкой. Найдите и устраните эту ошибку.

Рис. 1. Ошибочный код

Этап 2. Упражнение 1. Создание произвольного пути движения игрового объекта.

Сюжет игры: Rover движется по заданному маршруту.

Для успешного выполнения упражнения и создания игры, четко следуйте предложенному алгоритму:

1. Запустите программу Kodu.

2. Выберите команду **Новый пустой мир (Empty World).** Появится зеленое поле - основа для размещения игровых объектов в мире.

3. Добавьте объект **Rover**. Для добавления объекта щелкните левой кнопкой мыши по зеленому игровому полю и выберите объект **Rover**.

4. Нарисуйте произвольную траекторию движения **Rover-а**. Для создания пути следования объекта выберите инструмент **Путь** (рис. 2) и при помощи точек отобразите путь следования объекта **Rover**, пример на рис. 3. Траекторию можно оборвать, нажав на **Esc**.

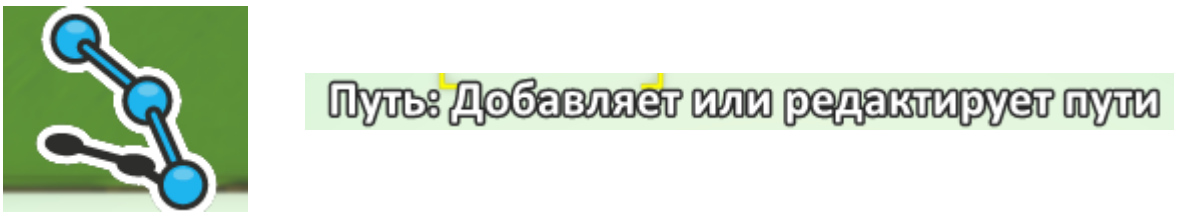

Рис. 2. Инструмент Путь

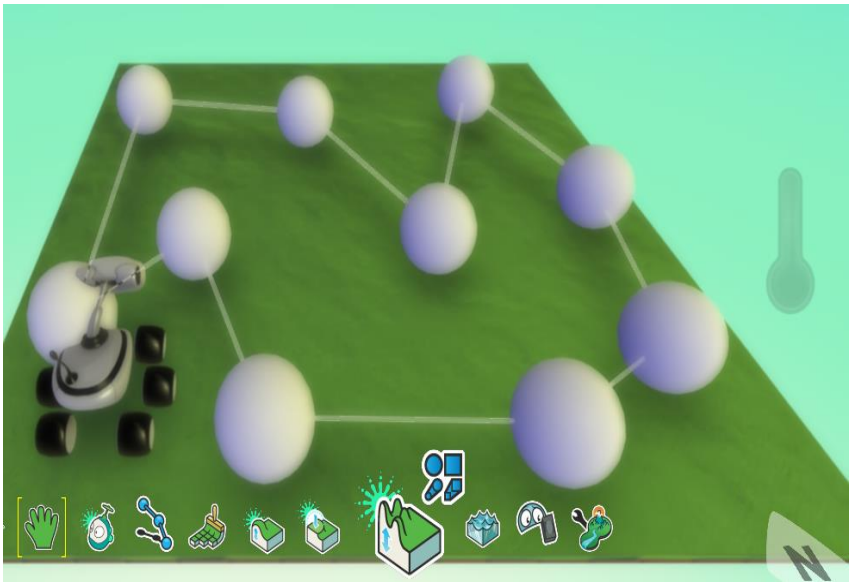

Рис. 3. Создание произвольного пути следования объекта Rover

5.Запрограммируйте движение объекта **Rover** по созданному пути, пример кода на рис. 4.

Рис. 4. Код для движения объекта по нарисованному пути

6. Запустите программу клавишей **Esc** и проверьте её работу. Rover должен ездить по заданной траектории.

7. Остановите программу клавишей **Esc**.

8. Сохраните программу под именем **Rover**. Для сохранения программы надо выйти в главное меню (клавиша **Home**) и выбрать пункт **Сохранить мой мир**.

Этап 3. Задание для самостоятельной работы:

Добавьте ещё одного объекта **Rover**, измените его цвет. Нарисуйте путь его движения и обязательно измените цвет пути, например, на красный.

Напишите программу, в которой **Rover** движется по красному пути. Напишите код, где в случае столкновения объектов **Rover** друг с другом они говорят: «Извините!». Примеры пути и кода на рис. 5 и 6. Код для столкновения следует написать для обоих Байкеров!

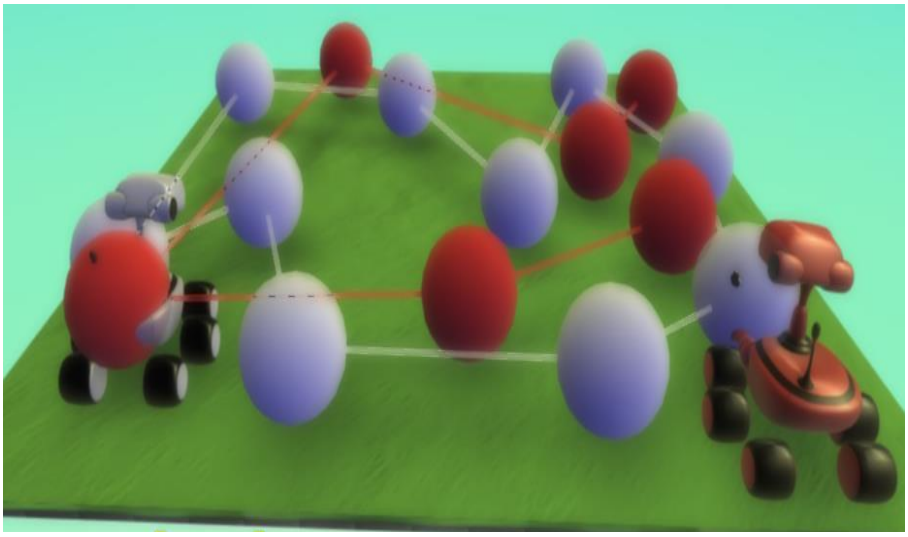

Рис. 5. Создание пути движения для двух объектов Rover

Рис.6. Код программы, в которой при столкновении объектов Rover произносятся извинения

Дополнительное задание:

Выполнив задания этого занятия, поэкспериментируйте с игрой **Tutorial 2**. Проверьте, сможете ли вы составить программу, позволяющую управлять движением и стрельбой мотоцикла с помощью клавиатуры и мыши.

Изучите также мир **3D Flare Paths**, содержащий интересные графические эффекты и примеры представления трехмерных объектов.

Этап 4. Упражнение 2. Создание клонов объектов.

Сюжет игры: три Аэростата стреляют по Самолету.

Для успешного выполнения упражнения и создания игры, четко следуйте предложенному алгоритму:

1. Создайте новый мир **(Empty World)**. Разместите в нем объекты Самолет и Аэростат. Напишите для **Аэростата** код, выполняющий следующие действия: когда **Аэростат** видит вдали **Самолет**, звучит громкая музыка. Приближаясь к **Самолёту**, **Аэростат** начинает светиться.

2. Составьте программу, позволяющую управлять **Аэростатом** с помощью клавиатуры. Пример программы приведен на рис. 7, строки 1 - 3.

3. Сохраните мир под именем **Самолет**.

4. Запустите программу на выполнение клавишей Esc. Подведите Аэростат

к Самолету, послушайте музыку, которая была выбрана для игры.

Рис. 7. Код, управляющий Аэростатом

5. Создайте эскадрилью **Аэростатов** из трех объектов. Для этого щелкните на уже имеющемся Аэростате правой кнопкой мыши и выберите пункт **«Копировать»**, затем поместите указатель мыши туда, где надо создать новый Аэростат, щелкните правой кнопкой мыши и выберите пункт **Вставить (Аэростат)**. Повторите вставку так, чтобы на поле оказалось **три Аэростата.**

6. Запустите игру и убедитесь, что все Аэростаты управляются и двигаются одновременно.

7. Добавьте код, позволяющий каждому **Аэростату** выстреливать **Пульки**, рис. 7, строка 4. Сохраните игру, запустите её и убедитесь в работоспособности.

Этап 5. Упражнение 3. Создание порожденных объектов.

Сюжет игры: новая эскадрилья Аэростатов стреляет по самолету.

Для успешного выполнения упражнения и создания игры, четко следуйте предложенному алгоритму:

1. Откройте созданный в упражнении 2 мир **Самолет**.

2. Добавьте на игровое поле ещё одну группу Аэростатов, созданных при помощи технологии порождения. Для этого добавьте новый **Аэростат**, поменяйте его цвет, например на желтый.

3. Переместите созданный **Аэростат** за пределы игрового поля. Данный объект не будет виден в процессе игры, а нужен только для того, чтобы создавать новые объекты Аэростат.

4. Задайте созданному Аэростату параметр **Родитель**. Для этого щелкните правой кнопкой мыши по Аэростату и выберите пункт **«Изменить установки»**. Активируйте параметр Родитель. Запустите программу на выполнение клавишей **Esc** и убедитесь, что новый Аэростат не виден. Остановите программу и перейдите в режим редактирования (клавиша Esc).

5. Создайте три **Аэростата** в новой эскадрилье. Для этого скопируйте созданный родительский **Аэростат**(правая кнопка мыши), а затем трижды вставьте объект **Аэростат** в любое место игрового поля. Если поместить указатель мыши на один из **Аэростатов**, между исходным и новым объектом появится пунктирная линия, показывающая, что эти **Аэростаты** входят в цепочку порождаемых объектов (рис. 8).

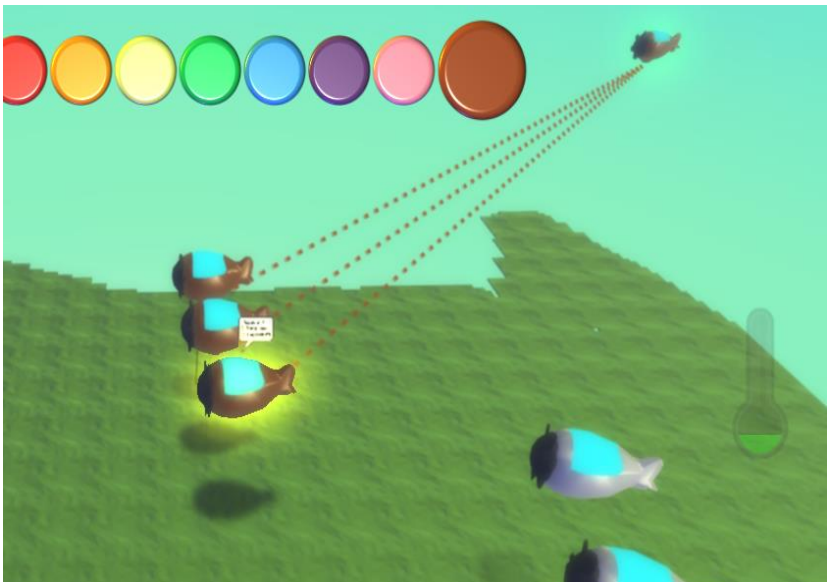

Рис. 8. Объекты« Аэростат» связаны с родительским объектом

1.Напишите код для родительского Аэростата, заставляющий объекты двигаться к Самолету. Обратите внимание, что теперь код надо написать

только один раз (для родительского объекта), в порождаемых объектах все строки кода добавляются автоматически.

2. Сохраните программу, запустите её на выполнение, убедитесь в работоспособности.

Дополнительное задание:

Напишите код, позволяющий при столкновении **Аэростатов** с **Самолётом** издавать звук.

Этап 6. Упражнение 4. Опция Родитель

Сюжет игры: Каждые 4 секунды объект Камень создает новый объект Летающая рыба. Камень стреляет ракетами по появляющимся рыбам.

Для успешного выполнения упражнения и создания игры, четко следуйте предложенному алгоритму:

Загрузите программу Kodu;

Выберите команду Новый пустой мир (Empty World). Создайте объект Летающая Рыба и измените её установки, включив опцию «Родитель». Добавьте объект Камень и задайте ему действия таким образом, чтобы каждые 4 секунды он создавал объект Летающая Рыба. Используйте таймер и опцию Родитель. Пример кода приведен на рис. 9.

Рис. 9. Пример кода для создания объекта Рыба

Создайте программу для Kodu, благодаря которой он сможет стрелять по появляющимся рыбам ракетами. Измените установки Kodu так, чтобы перезарядка ракетами происходила за минимальное время.

Сохраните мир.

Этап 7. Подведение итогов

Экспериментируя с местностью в среде Kodu, ответьте на вопросы:

Что позволяет опция Родитель?

Какие параметры (опции) объектов можно изменять? Как получить информацию по опциям? По каким признакам можно понять, что для персонажа включена опция Родитель?

Провести ситуационную рефлексию. Обсудить вместе с учащимися, что усвоили на данном занятии. Что получилось лучше всего? А что ещё необходимо повторить?

Приложение 2

Занятие 3. Знакомство с визуальной средой программирования Kodu: подсчёт баллов, индикатор здоровья, объект таймер

На этом занятии познакомишься с тем, как заставить игровой объект выполнять команды в определенные моменты времени, программировать характеристики и поведение персонажей, начислять им баллы за определенные действия, отслеживать успех действий в виде индикатора здоровья.

Использование таймера, индикатора здоровья и функции подсчета очков позволит придать игре более завершенный вид.

Этап 1. Актуализация и повторение пройденного.

Подведите учащихся к выводу о том, что счёт может уменьшаться и увеличиваться в зависимости от действий, совершаемых героями игры. Таймер же необходим при программировании событий. Его использование может влиять на исход игры: определённые действия могут(или должны) выполняться за определённое время, что регламентируется правилами игры.

Этап 2. Упражнение 1. Назначение времени действия игрового объекта

Сюжет игры: Завод выпускает мячи каждые 10 секунд и монеты каждые 5 секунд. Для успешного выполнения упражнения и создания игры, четко следуйте предложенному алгоритму:

1. Загрузите программу Kodu.

2. Создайте новый пустой мир.

3. Добавьте объект **Завод**. При необходимости уменьшите размеры объекта. Уменьшение размеров объекта выполняется через выбор команды *Уменьшить размер* в меню объекта (открывается щелчком правой кнопки мыши по объекту - рис 10).

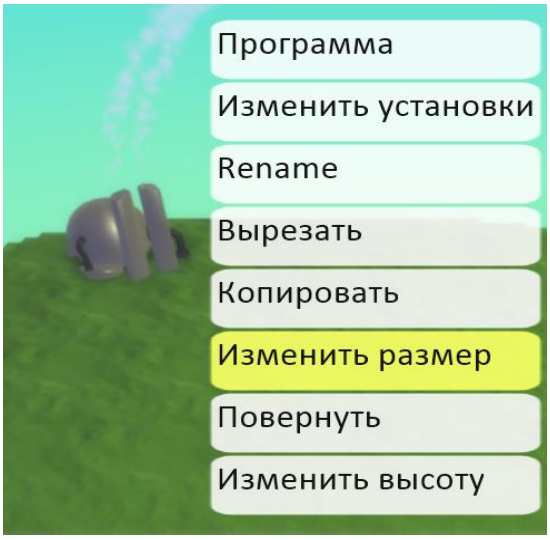

Рис. 10. Изменение размера

4. Напишите код для объекта **Завод**, согласно которому каждые 10 секунд

Завод производит футбольный мяч, а каждые 5 секунд - монеты.

Для написания кода щелкните правой кнопкой мыши по объекту Завод и выберите команду **Программа**. В открывшемся окне введите программу, содержащую команды (пример приведен на рис. 11):

When Таймер - 5 сек **Do** действия - запуск мяч.

When Таймер - 10 сек **Do** действия - создать - монета.

Рис. 11. Пример кода с использованием опций таймера

5. Запустите программу на выполнение клавишей **Esc**. Обратите внимание на отличие действий **Запуск и Создать**. Действие Запуск характеризуется более динамичным действием по созданию объекта. Сохраните созданный игровой мир под именем Завод. Для сохранения

нажмите клавишу **Home** и в главном меню выберите команду Сохранить мой мир.

Задание для самостоятельной работы:

Добавьте объект **Спутник** и сделайте так, чтобы он запускал объекты **Звезда** в интервале от 0 до 10 секунд. Пример кода приведен на рис. 12.

Рис. 12. Пример кода, когда таймер срабатывает случайным образом в интервале между 0 и 10 секундами

Этап 3. Упражнение 2. Начисление баллов за действия объекта

Сюжет игры: за каждый мяч **Завод** получает 10 баллов, за выпуск монеты - 1 балл. Для успешного выполнения упражнения и создания игры, четко следуйте предложенному алгоритму:

1. Загрузите программу **Kodu**.

2. Откройте созданный в упражнении 1 игровой мир Завод. Для загрузки мира выберите команду **Загрузить мир** в главном меню. Переход в главное меню через кнопку **Home**.

3. Добавьте код для подсчета баллов. Пример кода приведен на рис. 13 в строках 3 и 4. В строке 5 записана команда остановить игру, если будет нажата клавиша **Пробел**:

When клавиша Пробел **Do** конец.

Рис. 13. Пример кода с использованием опций таймера и подсчета баллов

4. Запустите программу на выполнение клавишей **Esc.** Примерный результат игры приведен на рис. 14. Нажмите клавишу **Пробел** для выхода из игры.

5. Сохраните игру.

Рис. 14. Пример игры по выпуску мячей и монет, и начисления

Этап 4. Задание для самостоятельной работы:

Попробуйте изменить код программы так, чтобы за каждый мяч начислялось 20 баллов, а за каждую монету 5. Напишите программу, которая позволяет остановить игру, если набрано 100 баллов.

Этап 5. Упражнение 3. Использование индикатора уровня жизни.

Сюжет игры: Объект Kodu теряет здоровье при поедании синих яблок. При низком уровне здоровья объект **Kodu** светится красным. Для успешного выполнения упражнения и создания игры, четко следуйте предложенному алгоритму:

1. Откройте игру, созданную в упражнении 1 занятия 1. Сделайте так, чтобы **Kodu** терял здоровье при поедании синих яблок. При низком уровне здоровья добавьте красное свечение объекта **Kodu**.

2. Используя меню **Изменить установки**, включите для объекта Kodu опцию **Показать жизни**(рис. 15).

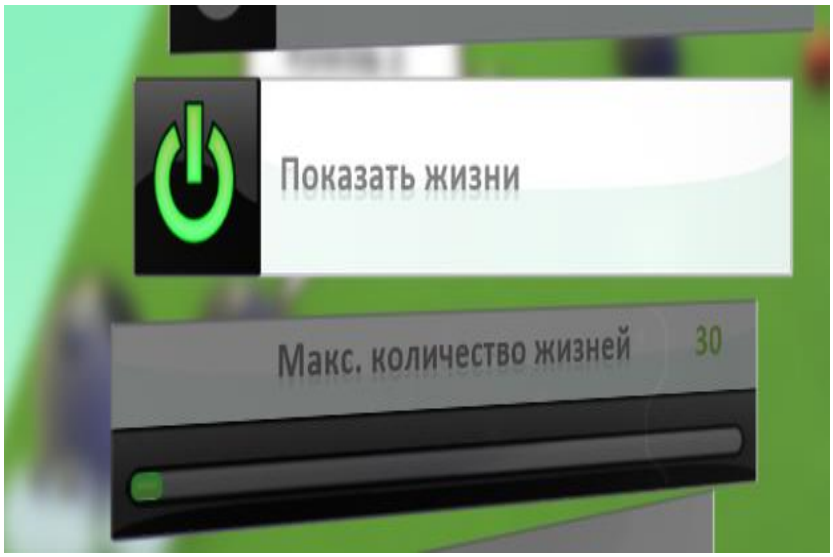

Рис. 15. Изменение установки. Показать жизни

3. Здесь же, в меню опций, установите максимальное количество жизней -30

4. При выходе из меню Изменить установки, вы увидите над объектом **Kodu** зелёную полосу, которая и является индикатором здоровья (рис. 16). Проверьте, остаётся ли индикатор видимым в режиме игры.

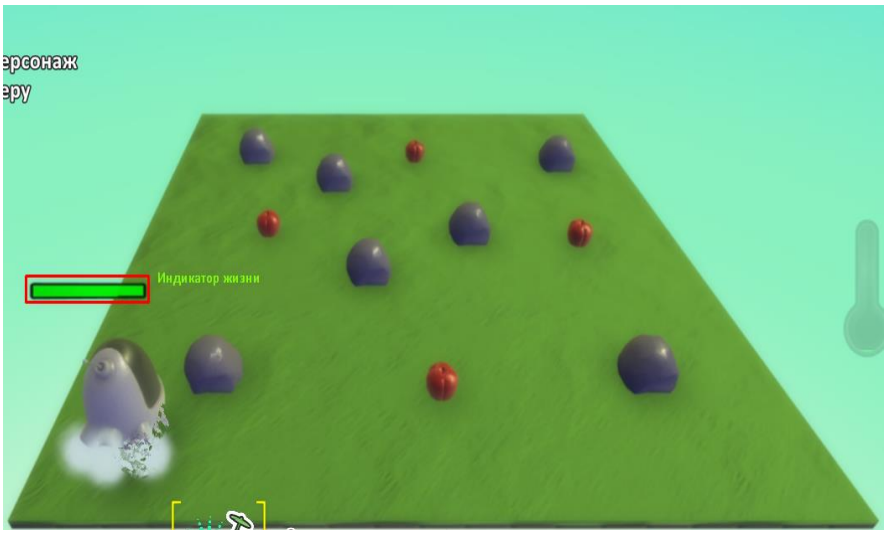

Рис. 16. Индикатор здоровья

5. В код объекта **Kodu** добавьте строку, уменьшающую количество жизней на одну при поедании синих яблок (рис. 17). Учтите, что ранее запрограммировано поедание яблок при их касании, поэтому для условия используется **Касание**. Для действия воспользуйтесь карточкой Ущерб, которая находится в команде Сражение.

Рис. 17 Строка кода, отвечающая за уменьшение уровня жизни

6. Запустите игру на выполнение и проследите за изменениями, происходящими с индикатором здоровья при поедании **Kodu** синих яблок (рис 18).

Рис. 18. Уменьшение уровня жизни

7. Добавьте свечение Kodu, сигнализирующее о критическом уровне жизни, используя соответствующий код в программе(рис. 19).

Рис. 19. Строка кода, программирующая свечение Kodu

8. Сохраните игру.

Этап 6. Подведение итогов

Порассуждайте вместе с учащимися над вопросами и интересными фактами, представленными в пособии.

● Выясните, для чего в программировании используется генератор случайных чисел.

Предполагаемый ответ: случайные числа используют в играх, лотереях, для шифрования информации, которая передается по сетям. Случайным образом можно показывать фото или проигрывать музыку.

● Найдите в Интернете информацию о том, какое отношение к программированию имеет город Монте-Карло. Подсказка: это связано со случайными числами.

Предполагаемый ответ: Годом рождения метода Монте-Карло считается 1949 год, когда в свет выходит статья Метрополиса и Улама« Метод Монте-Карло». Название метода происходит от названия коммуны в княжестве Монако, широко известного своими многочисленными казино, поскольку именно рулетка является одним из самых широко известных генераторов случайных чисел. Станислав Улам пишет в своей автобиографии «Приключения математика», что название было предложено Николасом Метрополисом в честь его дяди, который был азартным игроком.