Оглавление

Введение.....3

Глава 1. Использование новых технологий обучению алгоритмики в средней школе.....6

1.1. Место алгоритмики в курсе информатики средней школы.....6

1.2. Роль использования прикладного программного обеспечения в обучении школьников в контексте ФГОС.....11

1.3. Использование программы GoogleScratch в обучение программированию в разделе «Алгоритмика» в курсе информатики средней школы..... 18

Глава 2. Эффективность использования программы GoogleScratch в педагогический процесс курса информатика средней школы.....23

2.1. Цель, задачи, требования к результату изучения программирования по средствам GoogleScratch..... 23

2.2. Материалы теоретических и практических занятий с использованием GoogleScratch.....25

2.3. Контролирующий эксперимент эффективности использования GoogleScratch в учебной деятельности.....46

Заключение.....55

Список использованной литературы.....57

Введение

Действующий в нашей стране ФГОС НОО, представляет новые требования к технологиям и результатам обучения в школе. Компетентностный подход, на котором построены требования стандартов второго поколения, представляет собой развитие личностно-значимых компетенций (другими словами качеств), которые помогут школьнику гармонично развиваться и быть успешным в жизни.

Группа основных компетенций (ценностно-смысловая, общекультурная, информационная, коммуникативная и др.) формируется во время учебной и внеучебной детальности ребенка. Реализация учителем любой из перечисленных видов деятельности в средней школе осуществляется путем использования вариативных технологий обучения.

Использование интерактивных и информационно-технологических методов обучения, позволяет повысить интерес учеников к познавательной деятельности. Вариативность использования программного обеспечения сегодня, позволяет повысить интерес детей и компетенции педагога.

Особое внимание в педагогической среде уделяют изучению возможных подходов к организации проектной внеучебной образовательной деятельности по информатике в средней школе. Опыт таких видных педагогов как М.П. Лапчик, И.Г. Семакин, Е.К. Хеннер свидетельствует, что такая деятельность позволяет, с одной стороны, организовать среду для самореализации и самоутверждения учеников, и, с другой стороны, сформировать у них тягу к творчеству и знаниям и дать подходящие средства её реализации. Решение выше указанной проблемы может базироваться на использовании в обучении информатике программы GoogleScratch (язык программирования Scratch).

Scratch – это объектно-ориентированная среда, в которой блоки программ собираются из разноцветных кирпичиков команд точно также, как машины или другие объекты собираются из разноцветных кирпичиков в

конструкторах Лего. Кроме того, в нем можно найти современные идеи из сред визуального программирования типа Delphi и даже из презентационных систем. GoogleScratch – это мультимедийная система. Большая часть операторов языка направлена на работу с графикой и звуком, создание анимационных и видеоэффектов. Манипуляции с медиаинформацией – главная цель создания Scratch. На данный момент методика преподавания темы «Программирование в среде Scratch» не разработана. Поэтому, изучение вопроса возможностей внедрения программного обеспечения GoogleScratch является **актуальным.**

Изучением психолого-педагогических условий использования проектных методов в школе занимались К.Н. Вентцель, У. Килпатрик, Е.С. Полат, А.Е. Петров, И.М. Соловьев и т.д. Исследованием вопроса использование современных технологий для достижения образовательных целей и задач психологи и педагоги занимаются последние десятилетия. Своим работы данному вопросу посветили В.А. Красильникова, П. Коммерс и др.

Цель дипломной работы - разработка методических рекомендаций к изучению темы «Программирование в среде Scratch», в курсе информатики средней школы. Для достижения цели необходимо решить следующие **задачи:**

1. Определить место алгоритмики в обучении информатике школьников среднего звена;
2. Изучить роль использования прикладного программного обеспечения в обучении школьников среднего звена в контексте ФГОС НОО;
3. Выделить возможности использования программы GoogleScratch в обучении информатике школьников среднего звена;
4. Назвать цель, задачи, требования к результатам изучения программирования по средствам GoogleScratch;
5. Разработать материалы теоретических и практических занятий с использованием GoogleScratch;

6. Провести контролирующий эксперимент эффективности использования GoogleScratch в учебной деятельности.

Объект исследования – курс информатики средней общеобразовательной школы. **Предмет исследования** – методы обучения в рамках темы «Программирование в среде Scratch».

В процессе работы были использованы **метод исследования** анализа психолого-педагогической и научно-методической литературы, эксперемент.

Структура дипломной работы. Работа состоит из введения двух глав, заключения, списка использованной литературы, и приложений.

В первой главе работы рассмотрены теоретические основы проблемы преподавания темы «Программирование в среде Scratch», обосновано значение изучение использования прикладного программного обеспечения в контексте ФГОС НОО.

Во второй главе описывается диагностика умения младших школьников программировать в среде Scratch. Кроме того, в этой главе представлены методические разработки уроков по теме «Программирование в среде Scratch», а так же проводится анализ эффективности разработанной методики.

Экспериментальная проверка разработанной методики прошла в НАЗВАНИЕ ШКОЛЫ и показала свою эффективность, что позволяет рекомендовать методику к использованию учителями информатики, работающих в средней школе.

Глава 1. Использование новых технологий обучению алгоритмики в средней школе

1.1. Место алгоритмики в курсе информатики средней школы

Основной целью курса ОИВТ является обеспечение прочного и сознательного овладения учащимися основами знаний о процессах преобразования, передачи и использования информации, роли информационных процессов в формировании современной научной картины мира, привитие учащимся навыков сознательного и рационального использования ЭВМ в своей учебной, а затем и в профессиональной деятельности.

Цели обучения информатике в школе: формирование у учащихся представлений о свойствах информации, способах работы с ней, в частности с использованием компьютера.

Задачи обучения информатике в школе:

– познакомить школьников с основными свойствами информации, научить приемам организации информации и планирования деятельности, в частности учебной, при решении поставленных задач;

– дать первоначальные представления о компьютере и современных информационных и коммуникационных технологиях;

- дать представления о современном информационном обществе, информационной безопасности личности и государства. [1]

Содержание курса обучения регламентируется государственным образовательным стандартом по данному предмету. Анализ государственного стандарта, а также базовых нормативных документов, в частности примерного календарного планирования по предмету, показал, что в своем первоначальном виде курс ОИВТ, предлагаемый школам содержит в себе множество недостатков и не адаптирован к условиям непрерывного развития информационных технологий.

Таблица 1

Содержание курса ОИВТ в среднем звене общеобразовательной школы

Класс	Содержание курса
5 класс	Первоначальное представление об ОС. Освоение графического редактора Paint. Основы создания текстовых документов. Работа с программой «Блокнот»
6 класс	
7 класс	Базовый курс пользователя ЭВМ
8 класс	Изучение ПО
9 класс	Основной курс пользователя ЭВМ Основы алгоритмизации

Познавательные процессы: восприятие, внимание, воображение, память, мышление, речь – выступают как важнейшие компоненты любой человеческой деятельности. Для того, чтобы удовлетворить свои потребности, общаться, играть, учиться и трудиться, человек должен воспринимать мир, обращать внимание на те или иные моменты или компоненты деятельности, представлять то, что ему нужно делать, запоминать, обдумывать, высказывать суждения. Поэтому, без участия познавательных процессов человеческая деятельность невозможна, они выступают как ее неотъемлемые внутренние моменты. Они развиваются в деятельности, и сами представляют собой особые виды деятельности [18].

Развитие человеческих задатков, превращение их в способности – одна из задач обучения и воспитания, решить которую без знаний и развития познавательных процессов нельзя. По мере их развития, совершенствуются и сами способности, приобретая нужные качества. Знание психологической структуры познавательных процессов, законов их формирования необходимо для правильного выбора метода обучения и воспитания.

Чтобы успешно развивать познавательные процессы в учебной деятельности, необходимо, искать более современные средства и методы

обучения. Использование компьютера с его огромными универсальными возможностями и будет являться одним из таких средств.

Доказано, что процесс обучения школьников может быть более эффективным, если при объяснении определенных заданий будет использован компьютер, так как:

1) его использование оптимизирует деятельность учителя;

2) применение цвета, графики, звука, современных средств видеотехники позволяет моделировать различие ситуации и среды, развивая при этом творческие и познавательные способности учащихся;

3) он позволяет усилить познавательные интересы ученика:

Компьютер естественно вписывается в жизнь школы и является еще одним эффективным техническим средством, при помощи которого можно значительно разнообразить процесс обучения. Каждое занятие вызывает у детей эмоциональный подъем, даже отстающие ученики охотно работают с компьютером, а неудачный ход урока вследствие пробелов в знаниях побуждает часть из них обращаться за помощью к учителю или самостоятельно добиваться знаний.

С другой стороны, такой метод обучения очень привлекателен и для учителей: помогает им лучше оценить способности и знания ребенка, понять его, побуждает искать новые, нетрадиционные формы и методы обучения. Это большая область для проявления творческих способностей для многих: учителей, методистов, психологов, всех, кто хочет и умеет работать, может понять сегодняшних детей, их запросы и интересы, кто их любит и отдает им себя [7].

Кроме того, компьютер позволяет полностью устранить одну из важнейших причин отрицательного отношения к учебе – неуспех, обусловленный непониманием, значительными пробелами в знаниях. Работая на компьютере, ученик получает возможность довести решение задачи до конца, опираясь на необходимую помощь. Одним из источников мотивации является занимательность. Возможности компьютера здесь неисчерпаемы, и

очень важно, чтобы эта занимательность не стала превалирующим фактором, чтобы она не заслоняла учебные цели.

Одним из разделов, изучаемых в курсе информатики средней школы, является «Алгоритмика и основы программирования». Термин алгоритмизация в научно-технической литературе имеет множество несовпадающих значений, поэтому прежде чем обсуждать тему алгоритмизации представим спектр мнений по данному вопросу.

Алгоритмизация — процесс систематического составления алгоритмов для решения поставленных прикладных задач. Алгоритмизация считается обязательным этапом в процессе разработки программ и решении задач на ЭВМ. Именно для прикладных алгоритмов и программ принципиально важны детерминированность, результативность и массовость, а также правильность результатов решения поставленных задач.

Другими словами:

- алгоритмизация - этап решения задачи, состоящий в нахождении по формулировке задачи алгоритма её решения;
- алгоритмизация - раздел информатики, изучающий методы, приёмы построения алгоритмов и их свойства (иногда также называется алгоритмикой) [45].

В статье "Основы алгоритмизации и программирование" я нахожу такое объяснение: Алгоритм означает точное описание некоторого процесса, инструкцию по его выполнению. Разработка алгоритма является сложным и трудоемким процессом. Алгоритмизация – это техника разработки (составления) алгоритма для решения задач на ЭВМ.

Наконец, в Большой Советской энциклопедии в статье Н. П. Бусленко "Алгоритмизация процессов" я нахожу следующее развернутое описание алгоритмизации в части алгоритмизации процессов:

Алгоритмизация процессов – алгоритмическое описание процессов, описание процессов на языке математических символов для получения Алгоритма, отображающего элементарные акты процесса, их

последовательность и взаимосвязь. Алгоритмы, получающиеся путём А. п., предназначаются, как правило, для реализации на ЭВМ [28].

Построение алгоритмов, описывающих реальные процессы, связывается обычно с двумя задачами: нахождением эффективных систем обработки информации и исследованием математическими методами процессов функционирования больших систем. В задачах 1-го типа для построения алгоритма управления необходимо к алгоритму, описывающему процесс функционирования системы, присоединить алгоритм определения оптимального решения или оптимальных значений параметров управления. В задачах 2-го типа А. п. функционирования большой системы позволяет провести количественное и качественное исследования, связанные с оценкой основных её свойств (эффективности, надёжности и др.).

Для проведения алгоритмизации процесс расчленяется на элементарные акты (подпроцессы), применительно к которым может быть дано математическое описание, исходя из известных математических схем алгебры логики, конечных автоматов, случайных процессов, массового обслуживания теории и др. Соотношения, описывающие элементарные акты процесса, объединяются в систему, дополняются описанием взаимосвязей между актами и представляются в виде алгоритма.

Операции и процедуры, являющиеся элементами алгоритмического описания процесса, для программирования и реализации на ЭВМ удобно записывать на языке программирования, с которого при помощи трансляторов-программ алгоритм автоматически переводится на язык команд (операций) конкретной ЭВМ. При этом одной операции алгоритма может соответствовать в общем случае несколько операций ЭВМ.

Итак, согласно спектру представленных мнений, алгоритмизация – это процесс составления алгоритмов; этап решения задачи; раздел информатики (алгоритмика); описание очередности выполнения операций; техника разработки (составления) алгоритма. Кроме того, алгоритмизация процессов – это алгоритмическое описание процесса с целью получения алгоритма.

1.2. Роль использования прикладного программного обеспечения в обучении школьников в контексте ФГОС

Стандарт второго поколения определил требования, которым должны соответствовать образовательный процесс, его результат и, что не менее важно, условия обучения. [1] Инновации в системе образования связаны с внесением изменений:

- в цели, содержание, методы и технологии, формы организации и систему управления;
- в стили педагогической деятельности и организацию учебно-познавательного процесса;
- в систему контроля и оценки уровня образования;
- в систему финансирования;
- в учебно-методическое обеспечение;
- систему воспитательной работы;
- учебный план и учебные программы;
- в деятельность учащегося и преподавателя.

В стандарте 2009 года принципиально содержание образования детально и подробно не прописано, зато четко обозначены требования к его результатам, не только предметным, но и метапредметным, и личностным. И теперь задача системы образования - делать все возможное для достижения обозначенных результатов [1]:

1. разрабатывать новые образовательные программы;
2. применять эффективные образовательные технологии;
3. совершенствовать условия, в которых учатся дети.

Использование возможностей современных развивающих технологий, позволяет решать задачи, которые ставит школе ФГОС.

Роль информационно-коммуникативных технологий в ключе развивающего обучения очень велика. И.Т. Виштынейцкий определяет

понятие ИКТ, как совокупность методов, производственных процессов и программно-технических средств, интегрированных с целью сбора, обработки, хранения, отображения и использования информации в интересах ее пользователей. [11]

Основной компонент ИК-технологий – прикладное программное обеспечение (ППО). ППО – программа, предназначенная для выполнения определенных задач, рассчитанная на непосредственное взаимодействие с пользователем [39].

Другими словами это программы, при помощи которых ребенок может получать знания и развивать группы умений, согласно своим интересам, а так же для достижения образовательных задач.

Изучив возможные классификации прикладного обеспечения, были определены основные программы, которые могут использовать в учебной и внеучебной деятельности учеников средней школы.

Таблица 2

Вариативность использование ППО в учебной и внеучебной деятельности учеников средней школы

№	Элемент ППО	Использование
Программные средства общего назначения		
1	Текстовые редакторы	оформление рефератов, докладов, сочинений, таблиц
2	Графические редакторы	работа с изображениями, создание графиков (роза ветров и др.), диаграмм
3	Электронные таблицы	поиск и обработка табличной информации (наблюдения за погодой, численность и популяция животных, исторические события и др.)
4	Веб-браузеры	поиск информации

Программные средства специального назначения		
5	Экспертные системы	поисковые справочные (энциклопедические) системы, программы по созданию и решению тестов
6	Мультимедиа-приложения	создание, редактирование и защита презентаций
7	Гипертекстовые системы	электронные словари, энциклопедии, справочные системы
Профессиональные программные средства		
8	Геоинформационные системы	создание картосхем, наблюдения за объектами и явлениями
9	Программы программирования	создание простых и сложных алгоритмов и программ

Анализируя возможности использования ППО в рамках учебной и внеучебной деятельности учеников средней школы, можно установить, что варианты использования данных программ позволяют отвечать притязаниям практически всех учеников, в зависимости от их желаний и задач, которые они ставят перед собой.

Одним из вариантов использования ППО может стать проект, в независимости от своего типа, использования различных программ помогут сделать проект интересным для учеников, информативным для учителя и полезным для ребенка, занимающегося его созданием.

В контексте темы исследование, выделяются следующие типы прикладного программного обеспечения, используемого в контексте раздела «Алгоритмикп и основы программирования»:

1) Pascal;

2) MS Visual Basic;

3) Технологии VBA для офисных приложений, таких как MS Word, MS Excel, MS PowerPoint.

Однако данные программы, являются сложными и неадаптированными для школьного возраста. В последнее время большое внимание уделяется внедрению программам Visual C+ и GoogleScratch.

Использование таких простейших программ позволяет достигать образовательных, воспитательных и обучающих задач. Достижение целей происходит при соблюдении следующих условий:

- для решения алгоритмических задач выбрана понятная и интересная ученикам предметная область;
- программная реализация учебной среды программирования имеет дружественный пользовательский интерфейс;
- для создания алгоритма решения задачи используются визуальные средства представления структур данных и структур управления, не требующие запоминания большого количества служебных слов и синтаксических правил записи программы.

Такая программная среда, в основе которой лежит язык программирования Scratch, была разработана группой ученых из Массачусетского технологического института (MIT). Создавая этот язык, ученые поставили более широкую задачу, чем научить учеников младших классов программированию. Основная идея проекта (http://scratch.mit.edu) – стать частью образовательной программы детей и подростков, развить у них творческие способности, логическое мышление, свободу в использовании информационных технологий. Все это решается путем вовлечения учеников в создание мультфильмов, демонстрационных роликов, программ-тренажеров, компьютерных игр, интерактивных мультимедийных презентаций.

В основе программного продукта Scratch лежит графический язык программирования, включающий различные типы данных и структуры

управления, реализованный в понятном, дружественном интерфейсе, доступном для понимания и использования учениками начальных классов. Кроме того, развитые алгоритмические конструкции языка Scratch и концепции объектно-ориентированного программирования позволяют создавать достаточно сложные и эффективные программы, делают эту среду полезной для старшеклассников и студентов (правда, в большей степени для непрофильных специальностей).

Программа на языке Scratch представляет собой скрипт (сценарий), созданный путем подбора и совмещения графических блоков, представляющих данные и структуры управления. Причем данные имеют мультимедийную природу, то есть ученик может создавать рисунки, музыку, оперировать с числовой и графической информацией.

Перечислим особенности Scratch, позволяющие эффективно использовать его на начальном этапе обучения программированию и информационным технологиям [29].

1) Для того чтобы дети эффективно осваивали знания в новой области, необходимо наличие мотивации. А мотивация учебной деятельности возникает, если они чувствуют личную заинтересованность в приобретении знаний для достижения своей цели. Scratch позволяет создавать проекты, которые интересны различным возрастным группам от 8 до 16 лет и старше – интерактивные презентации, мультфильмы, компьютерные игры, программы-тренажеры с использованием графики и мультимедиа.

2) При создании программ в Scratch не требуется написания текстов программ на формализованных языках программирования, так как здесь предоставлены все необходимые графические средства для изображения данных и структур управления. Совмещая графические блоки, можно создать программу и запустить ее на выполнение в той же самой среде Scratch.

3) Для облегчения процесса создания программы разработчики реализовали защиту системы от синтаксических ошибок. То есть при

совмещении графических блоков можно получить только синтаксически правильные конструкции.

4) Широкие возможности манипуляции с визуальными данными развивают навыки работы с мультимедиа информацией, облегчают понимание принципов выполнения алгоритмических конструкций и отладку программ. Наличие операторов «идти в указатель мыши», «повернуться к указателю мыши», «если край, оттолкнуться» и так далее позволяет создавать динамичную графику детям, еще не владеющим математическим аппаратом манипулирования графическими объектами в координатной системе.

5) Использование шаблонов картинок и звуков из существующих библиотек, создание собственных файлов, выполнение таких операций с файлами проектов, как «Сохранить, Открыть, Создать», позволяет ученикам быстро освоить работу с файловой системой и стандартными приложениями.

6) Получение навыков общения в IT-сообществе, умения провести преобразование ил редактирование существующего проекта, изменить картинки и скрипты в проектах других людей и добавить их в свой собственный проект путем совместной работы и обмена через сеть «интернет» создает условия для подготовки человека к активной жизни в информационном обществе. Как было отмечено, возрастной диапазон пользователей Scratch достаточно широкий.

Школьники могут создавать проекты, не требующие сложной алгоритмической структуры. При этом они могут выразить свои творческие способности в сочинении сценария проекта, подготовке графических и звуковых компонент. Для старшеклассников среда Scratch может быть использована на вводных занятиях по изучению языков программирования высокого уровня для демонстрации основных понятий: переменная, значение переменной, логическое условие, типы данных, структуры управления.

Опыт работы аспирантов И.Г. Семакина с учащимися средней школы показывает, что после 6–8 часов учебных занятий с преподавателем дети

выходят на индивидуальную самостоятельную работу над проектами. При этом они с большим энтузиазмом относятся к публикации своих разработок в Сети, отслеживают интерес Scratch-сообщества к своим проектам. Кроме того, после двух недель работы ребята уверенно пользуются как самой средой Scratch, так и сопутствующими инструментами – графическим редактором, операционной системой Windows (уверенно оперируют с графическими и звуковыми файлами), программой Проводник и интернет – браузерами: Internet Explorer и Firefox Mozilla.

Количество объектов в одном проекте увеличивается от 1–2 до 5–10, соответственно увеличивается размер и сложность скриптов. Самое интересное – это то, что появляется мотивация к изучению новых возможностей Scratch, которые осваиваются самостоятельно, без участия педагога. Обмен проектами в Сети позволяет перейти ученикам на новую ступень обучения. Создается открытое образовательное пространство, где каждый участник проекта является носителем знания и новых идей его реализации. Наблюдая за развитием проекта Scratch и деятельностью школьников, можно отметить, что мы имеем дело с новой формой обучения. Педагоги, обратившись к миру мультимедиа и программирования, выпускают ученика в информационную среду творчества и познавательной деятельности, а дети кроме предметных знаний приобретают качества, необходимые каждому человеку для успешной жизни и профессиональной карьеры.

1.3. Использование программы GoogleScratch в обучение программированию в разделе «Алгоритмика» в курсе информатики средней школы

Scratch – среда программирования, появившаяся относительно недавно, даёт возможность учащимся младшего и среднего школьного возраста создавать игры, фильмы, анимированные истории и многое другое.

Программа Scratch в объектно-ориентированной среде «собирается» из разноцветных блоков команд так же, как собираются из разноцветных кирпичиков в конструкторах Лего различные объекты. Создание программ в Scratch происходит путём совмещения графических блоков в стеках. При этом блоки сделаны так, что их можно совмещать только в синтаксически верные конструкции, что исключает ошибки. Различные типы данных имеют разные формы блоков, подчеркивая совместимость / несовместимость объектов между собой. Имеется возможность внесения изменений в программу даже тогда, когда она запущена, что позволяет экспериментировать с новыми идеями по ходу решения задачи. В результате выполнения простых команд создаётся сложная модель, в которой взаимодействуют множество объектов, наделенных различными свойствами. После того как проект создан в Scratch, есть возможность его разместить на сайте http://scratch.mit.edu/.

Одним из принципиальных достоинств данной среды является то, что она является свободно распространяемым программным продуктом, таким образом, любое образовательное учреждение может скачать программу из интернета и приступить к непосредственному изучению и работе в новой среде программирования. Scratch не требует установки [23].

Сама идеология Scratch позволяет использовать при обучении современные методики и технологии обучения, такие как проблемный подход и метод проектов. После изучения основных конструкций языка и возможностей среды ставится задача по созданию и разработке соответствующего проекта. Это могут быть различные истории, тематику которых учитель предлагает с учётом возрастных особенностей учащихся, например, «Моя семья», «Мои увлечения», «Талантливые люди» (традиционная тематика проектов для пропедевтического обучения); рекламные ролики; анимированные истории по стихам и сказкам, изучаемым в школе и просто любимым учениками и т.д.

Данная технология обучения стимулирует учащихся к освоению возможностей языка программирования, изучению предмета «Информатика и ИКТ», подчеркивая их практическую личностную значимость. Анализ работы в Scratch показывает, что программа достаточно проста и легко осваивается. Однако, несмотря на свою простоту, Scratch предоставляет пользователю разнообразные средства работы с мультимедийными ресурсами, что вызывает интерес у учащихся, способствует развитию положительной мотивации к предмету в целом.

Среда Scratch разработана и поддерживается группой MIT Media Lab из Массачусетского института технологии (http://scratch.mit.edu). Scratch является бесплатным продуктом, что немаловажно для российских школ. Его разработка ведётся на языке Squeak, одном из наследников Smalltalk. Алан Кей, стоящий у истоков Smalltalk, а значит и у самых корней объектно-ориентированных технологий вообще, очень заинтересованно относится к развитию мышления и креативности детей. Причём Кей считает, что такое развитие должно начинаться как можно раньше: «…the media that powerfully shape our ways of thinking must be made accessible as early in life as possible». Среда (и язык) программирования Scratch, по задумке его создателей, является как раз тем средством, которое способно формировать «способы мышления».

Одной из главных концепций языка Scratch, является развитие собственных задумок с первой идеи до конечного программного продукта. Для этого в Scratch имеются все необходимые средства:

– стандартные для языков процедурного типа: следование, ветвление, циклы, переменные, типы данных (целые и вещественные числа, строки, логические, списки – динамические массивы), псевдослучайные числа;

– объектно-ориентированные: объекты (их поля и методы), передача сообщений и обработка событий;

– интерактивные: обработка взаимодействия объектов между собой, с пользователем, а также событий вне компьютера (при помощи подключаемого сенсорного блока);

– параллельное выполнение: запуск методов объектов в параллельных потоках с возможностью координации и синхронизации;

– создание простого интерфейса пользователя.

Вместе с тем в Scratch пока отсутствуют такие важные компоненты языка вообще и объектно-ориентированного языка в частности, как процедуры и функции, передача параметров и возвращение значений (кроме как через глобальные переменные), рекурсия, определение классов объектов, наследование и полиморфизм, обработка исключений, текстовый ввод и файловый ввод и вывод. Однако не вызывает сомнений, что в ближайшем будущем язык пополнится этими и, вероятно, некоторыми другими компонентами. Как мы видим, Scratch действительно имеет богатые возможности. При этом для начала его изучения не требуется ничего (на первых порах), кроме умения читать, поскольку программа составляется из готовых цветных блоков. Этому уровню соответствуют практически все первоклассники.

Одной из важнейших особенностей проекта Scratch является большое постоянно действующее сообщество пользователей. Собственные разработки можно выкладывать в интернет и затем просматривать их на любом компьютере, где установлена Java (или сам Scratch). В настоящее время актуальной является версия Scratch 1.3, поддерживающая кириллицу (и несколько десятков других языков).

Уже в средней школе дети легко могут освоить такие понятия как «параллельность» и «синхронизация». При этом важным является не «знание» терминологии, но понимание взаимной связи выполняющихся потоков [30].

Scratch берет все лучшее от вычислительной техники и дизайна интерфейсов для того, чтобы сделать процесс программирования более

привлекательным и доступным для детей, подростков и тех, кто хочет научиться программированию. Основные особенности Scratch:

- Блочное программирование. Для создания программ в Scratch, вы просто совмещаете графические блоки вместе в стеках. Блоки сделаны так, чтобы их можно было собрать только в синтаксически верных конструкциях, что исключает ошибки. Различные типы данных имеют разные формы, подчеркивая несовместимость. Вы можете сделать изменения в стеках, даже когда программа запущена, что позволяет больше экспериментировать с новыми идеями снова и снова.
- Манипуляции данными. Со Scratch вы можете создать программы, которые управляют и смешивают графику, анимацию, музыку и звуки. Scratch расширяет возможности управления визуальными данными, которые популярны в сегодняшней культуре – например, добавляя программируемость, похожих на Photoshop фильтров.
- Совместная работа и обмен. Сайт проекта Scratch предлагает вдохновение и аудиторию: вы можете посмотреть проекты других людей, использовать и изменить их картинки и скрипты, и добавить ваш собственный проект. Самое большое достижение – это общая среда и культура, созданная вокруг Scratch.

Scratch предлагает низкий пол (легко начать), высокий потолок (возможность создавать сложные проекты) и широкие стены (поддержка большого многообразия проектов). В работе со Scratch уделяется особое внимание простоте, иногда даже в ущерб функциональности, для большей понятности.

Когда учащиеся работают над проектом в Scratch, они имеют шанс выучить важные вычислительные концепции, такие как повторения, условия, переменные, типы данных, события и процессы. Scratch уже был использован для представления этих понятий учащимся различных возрастов, от элементарной школы до колледжа. Некоторые учащиеся перешли от

традиционных, текстовых, языков программирования, после того, как им показали программирование в Scratch.

Scratch основан на лучшем от языка программирования Squeak. Он был вдохновлен предыдущей работой над Logo и Squeak Etoys, но задуман более простым и интуитивным.

Scratch – проект с открытым исходным кодом, но закрытой командой разработчиков. Исходный код доступен в свободном доступе, но приложение разрабатывается маленькой командой ученых из MIT Media Lab.

На основе анализа учебников и возможностей программы нами сделан вывод, что обучение учащихся средней школы программированию в среде Scratch должно проходить поэтапно, от простого к сложному. Для большего понимания того, как устроена программа, учащимся, на начальном этапе нужно привести пример из жизни. Таким примером может служить конструктор Лего, в котором фигуры и конструкции собираются из различных частей – кубиков, что в свою очередь показывает необходимость подбора правильных кубиков, так как если это не учитывать, то нужную фигуру или конструкцию мы не получим. Также учащимся легче понять понятие «алгоритм» из примера – инструкции по сборке.

Глава 2. Эффективность использования программы GoogleScratch в педагогический процесс курса информатика средней школы

2.1. Цель, задачи, требования к результату изучения программирования по средствам GoogleScratch

Программа GoogleScratch, является одной из адаптированных программ для изучения основ программирования в средней школе.

Цель изучения раздела «Алгоритмика и основы программирования» - ознакомление детей с основами программирования посредствам программы GoogleScratch.

Задачи изучения раздела:

1. Обучать основам программирования и алгоритмики;

2. Способствовать самостоятельной деятельности для получения основных навыков программирования;

3. Развивать познавательные интересы, интеллектуальных и творческих способностей детей;

4. Воспитывать ответственное отношение информации с учетом правовых и этических аспектов ее распространения; избирательного отношения к полученной информации;

5. Содействовать формированию опыта использование информационных технологий в индивидуальной, познавательной и проектной деятельности.

Содержание раздела разработана на основе:

- Федерального компонента государственного стандарта среднего (полного) общего образования. Информатика и ИКТ (профильный уровень) (одобрен решением коллегии Министерства образования России и Президиумом Российской академии образования от 23 декабря 2003г. № 21/ 12, утвержден приказом Минобразования России «Об утверждении федерального компонента государственных стандартов начального общего,

основного общего и среднего (полного) общего образования» от 5 марта 2004г. № 1089).

- Примерной программы курса информатики и информационных технологий (базовый уровень) // Программы для общеобразовательных учреждений: Информатика. 2 - 11 классы. - 2-е изд., испр. и доп. - М.: БИНОМ. Лаборатория знаний, 2007.

При изучении данного раздела, предполагается проведение практических работ, направленных на отработку отдельных технологических приемов, а также практикума - интегрированных практических работ (проектов), ориентированных на получение целостного содержательного результата, осмысленного и интересного для учащихся.

Основной методический принцип изучения раздела - все познается через труд, через преодоление ошибок, через процесс решения задач. Материал разбит на несколько занятий, в каждом из которых есть разноуровневые задания. Реализация обучения происходит по следующей схеме: «ознакомление - усвоение - проверка».

Таблица 2

Тематическое планирование занятий в рамках раздела «Алгоритмика и основы программирования» в курсе ОИВТ в средней школе

№	Этап	Практические задания	Тип урока
1	Введение в Scratch-программирование	Изучение интерфейса программы	Урок первичного предъявления новых знаний
2	Основы Scratch-программирования	Выполнение заданий, направленных на использование различных функций программ	Урок формирования первоначальных предметных навыков, овладения предметными умениями
3	Графический редактор	Создание кораблика посредствам	Урок повторения предметных задач,

		графических функций программы	умений и навыков
4	«Электронное пианино»	Создание простейших программ по прописанному заданию	Урок решения практических, проектных задач
5	«Игра «Арканоид»		

2.2. Материалы теоретических и практических занятий с использованием GoogleScratch

Конспект урока №1 «Введение в Scratch- программирование»

Одно из самых удивительных занятий нашего времени - программирование. Повелителей компьютеров называют программистами. Они знают слова языков программирования, которым подчиняются компьютеры, и умеют соединять их в компьютерные программы.

В 2003 году группа исследователей под руководством Митчела Резника из MIT Media Lab решила сделать общедоступный язык программирования. Через 4 года появился Скретч (англ. Scratch - линия старта). Делать в нем компьютерные программы смогли даже дошкольники.

Секрет в том, что в Скретче нет слов, которые нужно знать наизусть и уметь писать без ошибок. Программы в Скретче не пишут, а собирают с помощью мышки из готовых блоков-команд, похожих на блоки конструктора Лего.

В любой момент в программу можно вносить любые изменения. При этом вы сразу увидите, как они отражаются на ее работе.

Создаваемые в Скретче программы называют проектами. Этот термин используется во всех сферах человеческой деятельности.

Проект — это создание чего-то нового для решения проблемы.

Мы живем среди проблем (от греч. problema — преграда, задача): маленьких и больших, простых и сложных, временных и вечных. Все вещи,

машины, произведения искусства, компьютерные программы - созданы людьми для их решения, это - результаты проектов.

Скретч - инструмент создания разнообразных программных проектов: мультфильмов, игр, рекламных роликов, музыки, “живых” рисунков, интерактивных историй и презентаций, компьютерных моделей, обучающих программ для решения проблем: обучения, обработки и отображения данных, моделирования, управления устройствами и развлечения.



Рис. 1. Блоки конструктора Лего

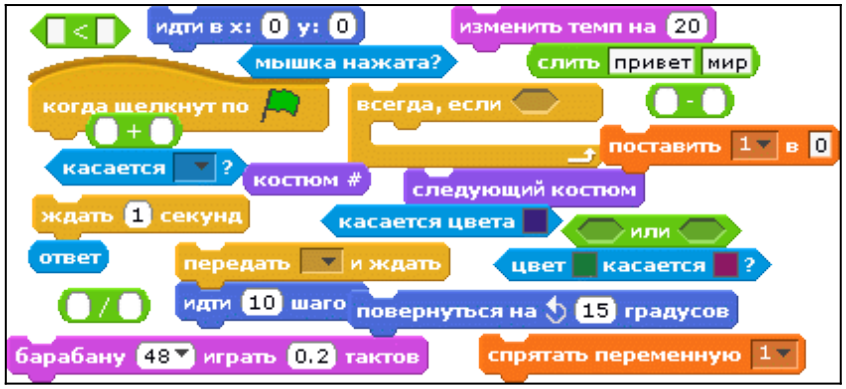


Рис. 2. Блоки команд языка «Скретч»

Для создания программных проектов Скретч имеет все необходимые средства: язык программирования, движок (интерпретатор) языка, графический редактор, систему помощи, образцы проектов, библиотеку рисунков и звуковых файлов. Все можно сделать здесь - в самой программе, и сейчас - сразу, без установки дополнительных программ.

Скретч – это не только среда программирования, но и социальная сеть(http://scratch.mit.edu), которая объединяет пользователей из разных

стран. Здесь можно опубликовать свои проекты и посмотреть чужие, скачать их, изменить. Отсюда можно бесплатно загрузить последнюю версию программы по ссылке: http://scratch.mit.edu/download.

Откроем для себя Скретч! Запустите программу с помощью ярлыка на рабочем столе или меню пуск - программы. Если программа не установлена - сделайте это сейчас.

Вы увидите окно программы. Исследуйте части интерфейса, выделенные красным цветом. Поиграйте некоторое время и попробуйте выяснить назначение основных элементов интерфейса.

Слово интерфейс переводится с англ. как поверхность, это то, что вы видите после запуска программы: ее окна, кнопки, символы. Он предназначен для управления программой. Кстати, как вы думаете, что можно считать интерфейсом велосипеда?

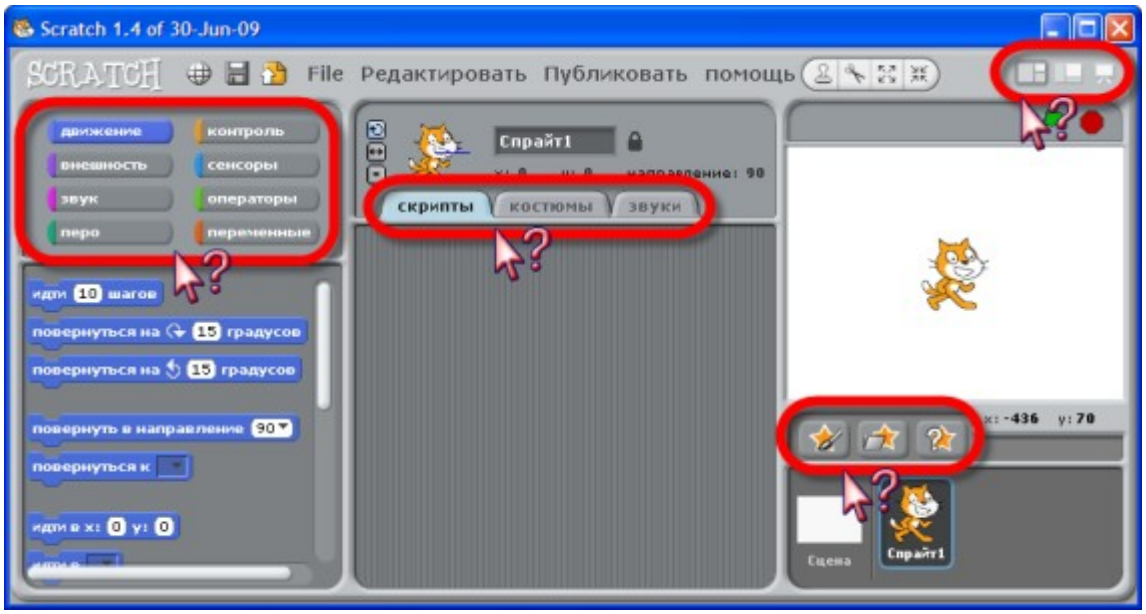


Рис. 3. Элементы интерфейса «Скретч»

Конспект урока №2 «Основы Scratch- программирования»

Скретч - компьютерная модель реального мира. Мир состоит из множества объектов (лат. objectum - предмет), обитающих в общем пространстве. Объекты - это все, что существует в природе: люди, животные, ветер, снег, дерево, солнце, буквы, мороженое, конфеты и т.д.

В Скретче объекты называют спрайтами (англ. Sprite - вымышленный герой, эльф), а пространство - сценой.

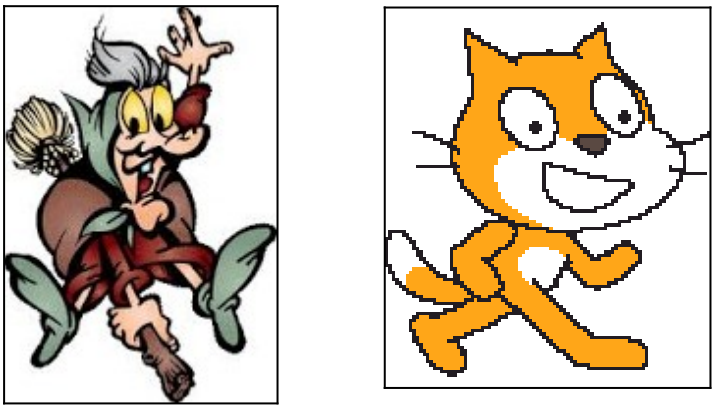


Рис. 4. Объекты «скретч-программирования»

При включении программы автоматически создается спрайт кота с именем Спрайт1.

Как вы думаете, можно ли сделать в Скретче спрайт Бабы-Яги?

Можно создать множество новых спрайтов с помощью трех звездных кнопок . Список всех спрайтов проекта размещен под сценой (Рис. 5.).



Рис. 5. Элемент интерфейса программы Google Scratch

Запустите Скретч. В программе есть библиотека готовых спрайтов:

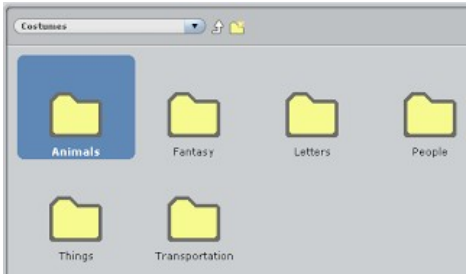


Рис. 6. Библиотека готовых спрайтов

Animals - животные; Fantasy - сказочные персонажи; Letters - буквы и цифры; People - люди; Things - вещи; Transportation – транспорт. Чтобы добавить новый спрайт из библиотеки, нажмите на среднюю кнопку под сценой, которая выглядит как папка.

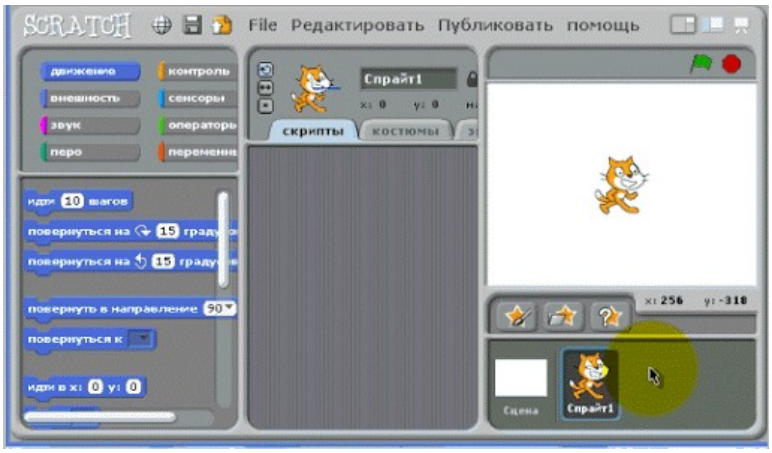


Рис. 7. Добавление спрайтов

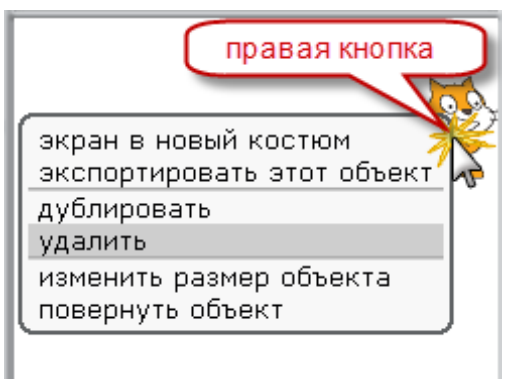
Постарайтесь выяснить, что делают две другие звездные кнопки?

Попробуйте переименовать спрайты. Имя можно изменить на более понятное в специальном окошке. Где оно расположено в интерфейсе программы?



Рис. 8. Переименование спрайтов

Чтобы удалить спрайт, воспользуйтесь инструментом ножницы над сценой или щелкните по спрайту правой кнопкой мыши для выбора команды удаления. Узнайте, что делают другие инструменты слева и справа от ножниц, а также другие команды правой кнопки.



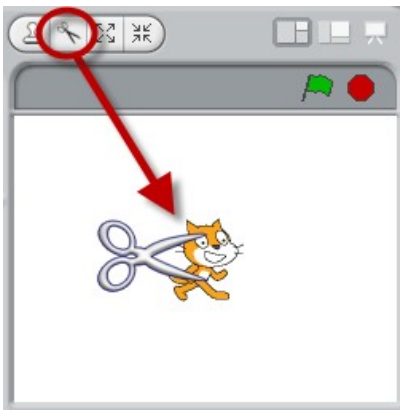


Рис. 9. Способы удаления спрайтов

При включении программы автоматически создается фон1: белый прямоугольник, имеющий 480 точек по ширине и 360 точек по высоте. В компьютерной графике минимальную точку изображения и экрана называют пиксел (можно говорить - пиксель). Сцена неподвижна, как аквариум, но ее обитатели ведут себя, словно непоседы, постоянно перемещаясь вверх и вниз, налево и направо.



Рис. 10. Неподвижный аквариум и его обитатели - подвижные объекты

Чтобы перемещениями спрайтов можно было легко управлять, сцена имеет прямоугольную систему координат - декартову (в честь французского математика Рене Декарта, который придумал ее в 1637 г.):

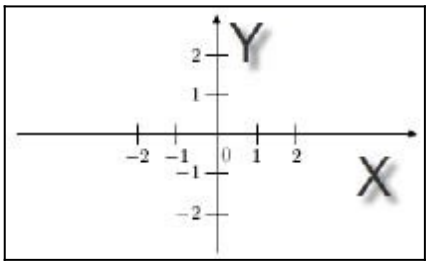


Рис. 11. Прямоугольная система координат

Пространство сцены условно разделено осями координат X и У, которые пересекаются в центре сцены. Здесь - начальная точка отсчета. На каждой оси есть отрицательное и положительное направления.

Координаты позволяют спрайтам не заблудиться на сцене и найти любую точку, т.к. каждый пиксел имеет точный адрес из двух чисел.

Сцену также можно представить как электронную таблицу или доску для игры в шашки и шахматы, в которой каждая строка и колонка имеют числовое название, а их пересечение образует адрес каждой ячейки.

-240 / 180	-239	...	...	-1	0	1	...	...	239	240
179										
...										
...										
1										
0					0					
-1										
...										
...										
-179										
-240 / 180										

Рис. 12. Сцена похожа на электронную таблицу

Попробуйте самостоятельно определить, где на сцене находятся положительные и отрицательные области значений X, У. Что такое направление?

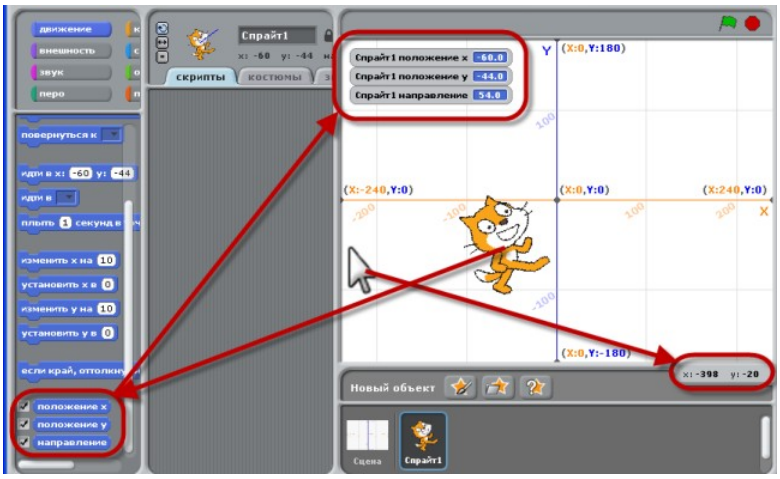


Рис. 13. Мониторы положения курсора мыши и активного спрайта

Сцена и спрайты могут иметь графические (изображения - рисунки, фотографии) и звуковые файлы. Изображения спрайтов называют костюмами, а сцены - фонами.

Образы и звуки для каждого объекта хранятся в независимых областях в сборочном центре программы. Там же расположены скрипты объектов, но с ними мы познакомимся немного позже.

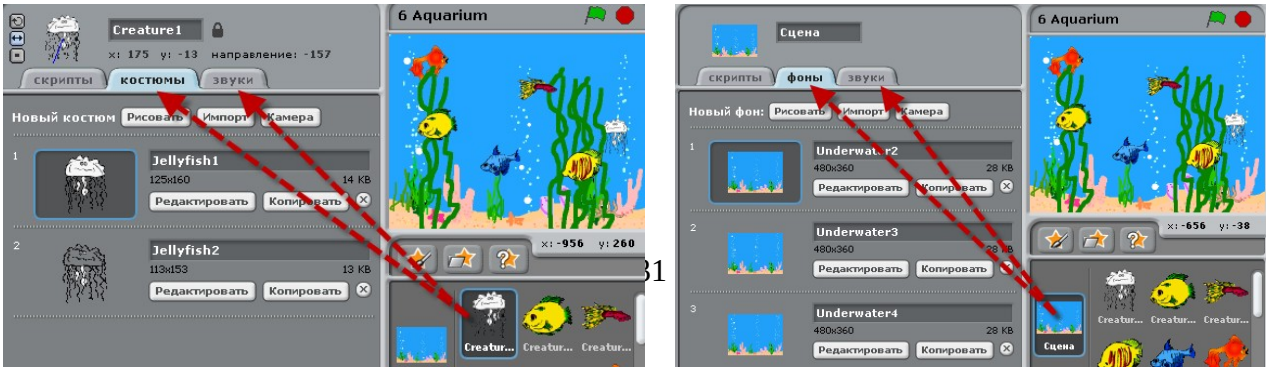
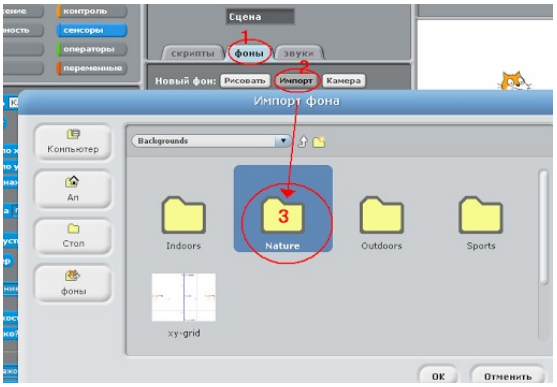


Рис. 14. Графические и звуковые файлы хранятся в центральном окне

Фон и костюм можно нарисовать, нажав на кнопку рисовать, или взять готовый рисунок из файла с расширением JPG, BMP, PNG, GIF через кнопку импорт. Чтобы выполнять действия со сценой или спрайтом, их нужно выбрать мышью в списке объектов. Активный объект выделяется голубой рамкой вокруг, как на рисунке.

Отправим героя в пустыню или, куда вы захотите. Для этого достаточно добавить рисунок пустыни в список фонов сцены через кнопку импорт.

1. Выберите Сцену в списке объектов.
2. Выберите вкладку Фоны.
3. Нажмите кнопку Импорт
4. Откройте папку Nature (природа):

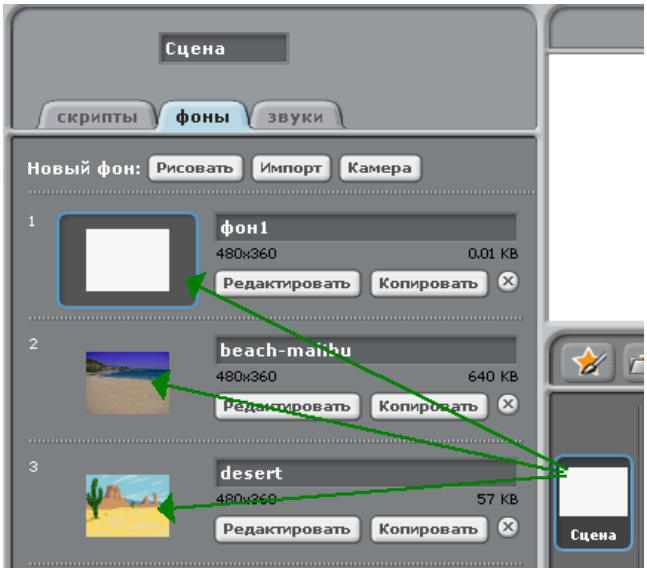


5. Выберите левой кнопкой мыши файл beach-malibu.
6. Нажмите кнопку Ок для добавления выбранного фона:



7. Повторите шаги 4-6 для импорта изображения desert

8. Теперь у вас в списке 3 фона. Любой фон можно отобразить на сцене, кликнув по нему левой кнопкой мыши. Выберите мышкой поочередно разные фоны, чтобы отобразить их на сцене:



9. Порядок фонов в списке можно менять, перемещая их мышью в нужную позицию. Переместите любой фон с одного на другое место в списке.

10. Удалите ненужный белый фон1, нажав напротив него кнопку с крестиком.

11. Сохраните проект в файле Упражнение_1.

Поведение есть у всех, даже у неподвижного пространства. Оно, например, может менять внешность. Ученикам за поведение порой ставят оценки и угощают чем-нибудь вкусненьким.

Бутерброд состоит из хлеба, масла и колбасы, а из чего состоит поведение? Из действий и сообщений. Люди, животные, насекомые, технические устройства способны действовать и сообщаться. Вы, наверняка, видели, как светофор с помощью цвета сообщает водителям и пешеходам, когда можно двигаться на перекрестке.



Рис. 15. Светофор передает сообщения

У поведения есть две тайны, которые позволяют предсказывать будущее.

Секрет №1

Поведением управляет сценарий - точный порядок действий и сообщений. Если вам известен, например, сценарий фильма, то вы точно знаете, когда и как именно поступит каждый герой, что и кому он скажет.

Секрет №2

Поведение всегда запускается событием, которым может быть действиеили сообщение.

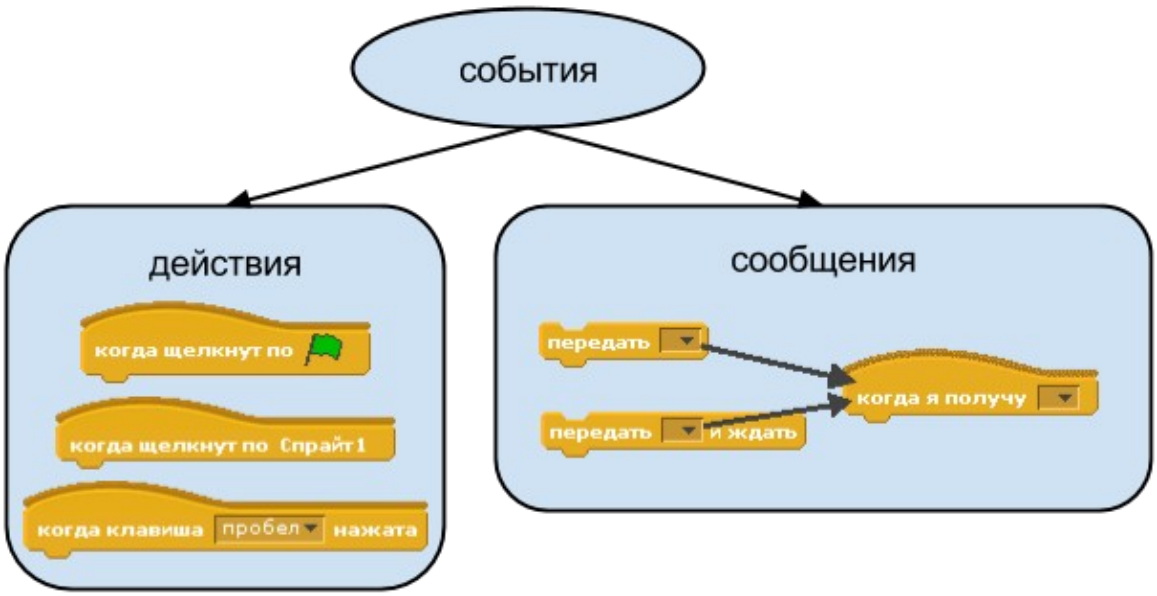


Рис. 16. Два типа событий

В Скретче действия совершает человек (пользователь программы) мышью иклавиатурой, а сообщения передают друг другу спрайты и сцена.

Поведение - это упорядоченные действия и сообщения объекта в ответ на событие.

В Скретче сценарий называют английским словом скрипт, которое переводится как сценарий.

скрипт=сценарий

Скрипты также, как изображения и звуки, добавляются во внутрь именно того объекта (спрайта или сцены), который был выделен мышкой в списке объектов. Чтобы собрать скрипт, нужно для выделенного объекта перетянуть блоки действий из левого окна в центр, соединяя их в стопку (или стек - стопка в переводе с англ. stack) .

Первый блок - это всегда блок события, запускающего скрипт на выполнение.

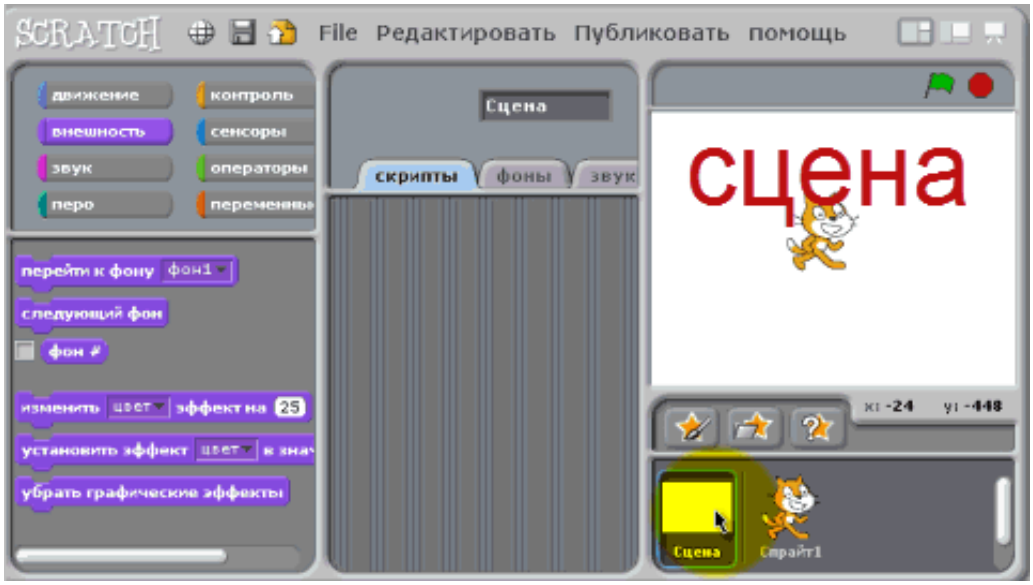


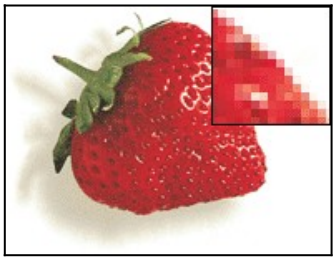
Рис. 17. Порядок сборки скриптов

Попробуйте собрать сценарий. В Скретче объекты называют спрайтами, пространство - сценой, а порядок их действий - скриптами. Каждая точка сцены имеет точные координаты из двух чисел: X - У. Графические и звуковые файлы, а также скрипты каждого объекта являются их личной собственностью и хранятся изолированно для каждого.

Конспект урока №3 «Графический редактор»

В Скретче есть свой редактор изображений, в котором можно сделать и отредактировать рисунок для спрайта и сцены. В версии программы Скретч 1.4 он работает только с растровыми изображениями, как и стандартный редактор Paint. В новой версии Скретч 2, которая доступна для зарегистрированных пользователей в их аккаунтах на сайте Скретч http://scratch.mit.edu/, есть и растровый редактор, и векторный.

Слово растр пришло из латинского языка, его буквальное значение - грабли.



Растровые изображения состоят из цветных точек (пикселов - сокращение от двух английских слов pix element, т.е. элемент изображений), упорядоченных в строки и столбцы. Если растровое изображение сильно увеличить, то каждая его точка (пиксел) превратится в квадрат. Чтобы убедиться в этом, достаточно увеличить масштаб просмотра любой фотографии. Попробуйте сами!

Как вы думаете, почему растровые изображения называют граблями?

Мы с вами научимся делать растровые изображения в Скретче.

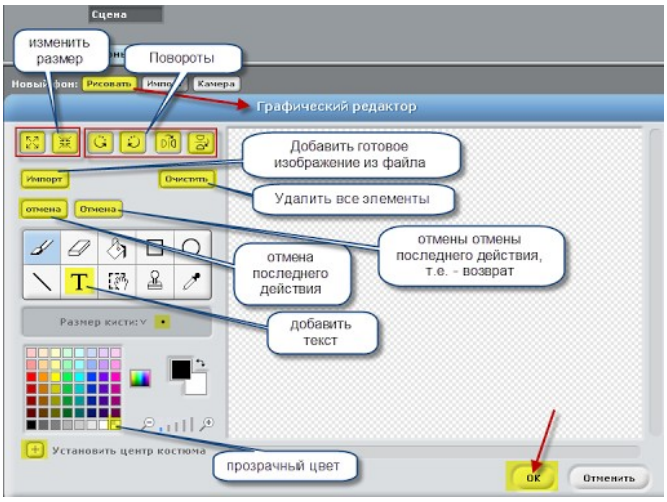


Рис. 18. Графический редактор Скретч

Есть пять вариантов использования и запуска редактора. Завершается работа с рисунком нажатием на кнопку OK для сохранения изображения или Отменить, если вы решили не хранить результат.

Используя инструменты,  линия и  заливка, нарисуйте Кораблик.

Если программа Скретч еще не запущена, запустите ее.

Запустите редактор, нажав кнопку рисования нового спрайта 

Выберите инструмент Линия на панели инструментов и нужную толщину линии.

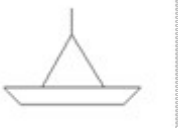
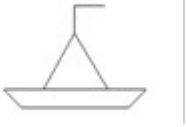
На рабочем поле выберите место, где будет располагаться Ваш рисунок и, нажав левую кнопку мыши, не отпуская ее, проведите линию необходимой вам длины (Шаг 1). Чтобы получить ровную линию, удерживайте при ее рисования клавишу Shift. Чтобы убрать лишнее, используйте инструмент ластик 

Выполните последовательно шаги со 2 по 11, как показано в Таблице 3.

Чтобы сделать изображение цветным, используйте инструмент Заливка . Для выбора цвета щелкните один раз левой кнопкой мыши по цвету в палитре, который вам необходим. Раскрасьте кораблик, выполняя шаги с 12 по 14 в таблице 3.

Таблица 3

Шаги рисования кораблика

Шаг 1	Шаг 2	Шаг 3
		
Шаг 4	Шаг 5	Шаг 6
		
Шаг 7	Шаг 8	Шаг 9
		
Шаг 10	Шаг 11	Шаг 12
		
Шаг 13	Шаг 14	
		

7. Установите центр рисунка.

8. Сохраните рисунок образа спрайта, нажав на ОК.

9. Используя команду экспорт костюма, сохраните рисунок в графическом файле с именем Кораблик. Сохраненный таким способом рисунок можно в дальнейшем использовать в любом документе в качестве иллюстрации.

Конспект урока №4 «Электронное пианино»

Цель проекта: учащиеся должны создать фрагмент пианино, которое способно воспроизводить звуки, играть готовые мелодии и показывать анимацию.

Реализация проекта: учащиеся были поделены на 3 группы. Первая группа – создание фрагмента пианино, написание скриптов воспроизведения звуков, запись мелодии. Вторая группа – разработка фона и спрайтов,

создание воспроизведения звуков, содействие в записи мелодии. Третья группа – разработка скрипта движения спрайта.

Для начала необходимо заранее подготовить изображения клавиш. Их можно нарисовать в графическом редакторе Paint, или с фотографии пианино аккуратно вырезать, так как мы сделали в нашем проекте.

При помощи объектов, собирается фрагмент пианино. Каждая картинка является отдельным объектом, называемым спрайтом. Каждый спрайт должен будет иметь 2 костюма – обычное состояние и нажатое состояние, а также воспроизводить нужный звук (ноты). В итоге мы должны получить фрагмент пианино.

Создается этот фрагмент следующим образом: в области спрайтов нажимается кнопка создания нового спрайта . В появившемся окне следует нажать кнопку 

, и в появившемся окне выбрать нужный спрайт. Данная процедура проводится столько раз, сколько спрайтов нужно отобразить на сцене. После того, как фрагмент готов, нужно разработать небольшое меню, которое будет упрощать работу с программой.

Левая кнопка «Готовые мелодии» является выпадающим меню, т.е. при нажатии на кнопку появляется новая кнопка рядом. Правая кнопка «Стоп» останавливает скрипт, прекращается воспроизведение звуков и анимации, на экране появляется информация о проекте.

Основное оформление готово. В дальнейшем мы еще вернемся к оформлению, для того чтобы предать нашему проекту больше интерактивности и красоты. Но на данном этапе все нужные объекты уже созданы.

Теперь переходим к созданию скриптов каждому спрайту. Начнем с нот, так как все остальные объекты будут опираться именно на эти скрипты. Сразу нужно отметить, что скрипты нот будут отличаться только непосредственно самой нотой, т.е. звуком, который данный спрайт должен воспроизвести. Поэтому целесообразнее сначала полностью решить, что

должно быть в скрипте каждой ноты, для того чтобы в дальнейшем использовать уже готовый скрипт для всех остальных нот.

В данном проекте у спрайта ноты должны быть следующие возможности: при нажатии на спрайт левой кнопкой мыши – воспроизводить нужный звук; при проигрывании готовой мелодии подсвечиваться в тот момент, когда нота, закрепленная за спрайтом звучит; перейти к костюму отображающему не нажатое состояние, в тот момент когда нажимается кнопка «Стоп».

Выбираем крайнюю левую ноту в нашем фрагменте, ноту «соль». Ищем спрайт этой ноты в панели спрайтов и щелкаем по нему два раза левой кнопкой мыши – данное действие откроет панель скриптов именно для этого спрайта. Также можно щелкнуть два раза левой кнопкой мыши по самому спрайту.

Исходя из этого, видно, что в работе спрайта участвуют три скрипта. Первый скрипт воспроизводит звук при нажатии на спрайт. При этом спрайт меняет свой костюм на следующий, т.е. нажатую ноту, воспроизводит звук под номером 43 (нота соль) продолжительностью 0,25 секунды и затем возвращается в исходное положение. Второй скрипт применяется тогда, когда играется готовая мелодия. «Когда я получу 1» означает то, что выполнение данного скрипта будет только после получения сообщения «1». Номера сообщений выбираются следующим образом: крайняя левая нота имеет сообщение 1, следующая – сообщение 2 и так далее. В данном случае меняются только лишь костюмы с интервалом в 0,25 секунды. Третий скрипт отменяет выполнение двух предыдущих скриптов и возвращает спрайт в исходное положение.

Для того чтобы передать готовые скрипты остальным спрайтам нот нужно нажать на начало скрипта (оранжевый контейнер) правой кнопкой мыши и выбрать кнопку 

. Возле курсора появится копия скрипта, которая «прилипает» к курсору до тех пор, пока не нажмем левую кнопку мыши.

Затем скопированный скрипт можно поместить в окно скриптов данного спрайта, а также присоединить к любому другому спрайту или сцене. В нашем случае нужно прикрепить к следующему спрайту ноты. Для этого щелкаем по спрайту следующей ноты. Если все сделано правильно, то возле спрайта в который был перемещен скрипт будет выполнен графический эффект (квадратик уменьшающийся в центр). Далее двойным щелчком левой кнопкой мыши выбираем спрайт в который мы перемещали скрипт и меняем воспроизводимую ноту. Для того, чтобы совершать меньше ошибок, скрипты белых нот рекомендуется копировать только в идентичные скрипты. Аналогично с черными нотами. Так будет совершенно намного меньше ошибок.

Такой способ применяется для копирования остальных скриптов. При копировании второго скрипта необходимо менять получаемое сообщение, для того чтобы избежать повторов. Сообщения создаются так: на стрелочку рядом с выбранным ранее сообщением нажимаем левой кнопкой мыши.

В появившемся меню ищем кнопку 

и нажимаем на нее. Появится диалоговое окно.

В этом окне мы вписываем название сообщения, при создании нот, это порядковый номер ноты слева направо.

При копировании третьего скрипта ничего менять не надо, если скрипты белых нот мы копируем только белым нотам, скрипты черных – только черным.

Заранее нужно разобраться с количеством и назначением сообщений в будущем проекте.

Листинг скриптов говорит о том, что выполнение одного действия – проигрывания музыки, разбито на множество маленьких скриптов. Данный способ реализации разгружает скрипт, Scratch работает быстрее, и более понятен для восприятия. Каждое действие, каждая строчка имеет свой скрипт, что позволяет легко находить ошибки или дописывать новые скрипты, копируя старые и изменяя небольшую часть.

При выполнения скрипта задействовано большое количество сообщений. Это нужно для того, чтобы программа понимала, что нужно выполнять в данный момент. Скрипты строчек куплета и припева очень похожи, единственное отличие это проигрываемая нота и длительность звучания.

Управляющие скрипты предназначены для того, чтобы запускать проигрывание мелодии в нужный момент, а именно после нажатия на кнопку 

, и останавливать выполнение скрипта после проигрывания всей мелодии или после нажатия на кнопку 

.

Кнопка 

служит для остановки всех выполняемых скриптов, возвращению всех спрайтов в первоначальное положение и выводу на сцену заставки.

Для придания проекту интерактивности, было решено добавить два объекта: полоса загрузки и танцующее и поющее существо.

Полоса загрузки применяется в трех случаях: при нажатии на кнопку 

, после окончания проигрывания музыки и после нажатия на спрайт заставки. Для создания анимации необходимо нарисовать несколько костюмов.

Листинг скрипта говорит о том, что спрайт при появлении начинает менять поочередно костюму, и когда одевает последний – исчезает. Получается красивый эффект ленты загрузки, реализованный самым простым способом.

Танцующее и поющее существо нужно для того, чтобы прослушивание мелодии сопровождалось анимационными эффектами, также реализуется функция караоке, так как на сцене появляются в нужный момент слова песни.

С точки зрения проверки знаний учащихся, реализация данного скрипта показывает умение работать учащихся с блоком «Движение», «Внешность», «Контроль» и «Переменные».

Реализация скрипта на первый взгляд смотрится простой, но при использовании в нашем проекте есть небольшая тонкость – анимация должна происходить только тогда, когда идет звучание музыки. Сделать это возможно двумя путями: Первый путь – рассчитать продолжительность по времени звучания музыки и создать скрипт, который будет проигрываться такое же количество времени; Второй путь – ввести переменную А, принимающая значение 0, которое скрипту говорит о том, что должна воспроизводиться анимация. После того, как будет сыграна последняя строчка припева, переменной А передается значение 1, и скрипт останавливается.

После этого создание проекта полностью завершено, цели и задачи выполнены. В дальнейшем проект можно разнообразить различными мелодиями, добавить звуковые эффекты, анимацию и многое другое.

Конспект урока №5 «Игра «Арканоид»»

Цель проекта: учащиеся должны реализовать игру «Арканоид» в среде программирования Scratch

Реализация проекта: создание спрайтов фишек, шарика и ракетки. Редактирование фона и обозначение условий игры. Создание переменных и границ сцены.

Для того чтобы создать игру «Арканоид» в среде программирования Scratch для начала необходимо решить несколько функциональных моментов: в начале игры шарик должен двигаться вниз, разный угол отскока шарика от ракетки, исчезновение фишки при попадании по ней шариком, игровые очки, получаемые игроком при попадании по фишкам, выигрыш игрока при сбивании всех фишек и проигрыш при падении шарика.

Сперва отредактируем сцену. Сцена состоит из трех частей, – нейтральное поле, по которому двигается шарик, – верхняя граница игрового поля, от которой шарик отскакивает и – нижняя граница игрового поля, попадание в нее шарика останавливает игру.

После создания игрового поля нужно переходить к разработке игровой ракетки.

Спрайт ракетки состоит из трех частей: красной, синей и зеленой. Такое строение нужно для того, чтобы при попадании шариком по ракетке, шарик отскакивал по разному. Т.е. при попадании в красную часть, шарик будет отскакивать больше влево, при попадании в синюю часть – отскок будет вверх под небольшим градусом и соответственно при попадании в зеленую часть – отскок больше вправо. Реализация, также возможна, используя большее количество цветов, так как чем больше цветов мы задействуем, тем точнее мы запрограммируем отскок шарика. Осталось создать скрипт движения ракетки. Существует два способа: движение с помощью клавиатуры или при помощи мыши. Минусы первого варианта в том, что движения получаются не плавные и не удобными при использовании, так как приходится очень часто нажимать на клавиши управления ракеткой. В то же время управления мышью позволяет сделать движение ракетки плавным и управление удобным.

Скрипт состоит из двух частей: первая – устанавливает ракетку в начальное положение (0; – 150), вторая – запускает движение ракетки за курсором мыши.

Следующий объект, который нужно создать – это фишка. Фишек в проекте может быть любое количество, в нашем проекте 14 фишек. Создавать фишки нужно так: создать одну фишку, назначить ей скрипты, а затем скопировать необходимое количество раз.

Скрипт фишки также состоит из двух частей. Первая часть – установка фишки в нужную позицию на игровом поле, путем задания каждой фишки значения по X и по Y. Вторая часть – при касании шариком фишки, фишка должна исчезнуть.

Прежде чем переходить к созданию шарика, нужно реализовать три состояния игры: перед началом, выигрыш и проигрыш.

Начало игры, движение шарика, начинается после нажатия клавиши «пробел». В верхней части сцены находится информационная зона, которая сообщает игроку скорость движения шарика, набранные очки и оставшееся количество фишек на игровом поле.

Скорость шарика можно сделать статической, т.е. постоянной, или динамической, т.е. изменяемой при каких-либо условиях. В нашем проекте таким условием является попадание шарика на ракетку. Первоначальное количество очков равно 0. Очки начисляются после попадания шариком по фишке и отнимаются при ударе об ракетку. Количество фишек в начале игры равно 14, после попадания шариком по фишке, число фишек уменьшается, так как пропадает выбитая фишка, и тогда когда значение будет равняться 0 появляется сообщение о выигрыше.

Если шарик касается нижней полосы, то появляется сообщение о проигрыше.

Теперь все готово для того, чтобы переходить к созданию самого шарика и скриптов. Скрипт шарика состоит из 2 частей: первая – отвечает за установку шарика в нужную точку и за сброс переменных, вторая – отвечает за движение шарика по игровому полю, попадания по ракетке и фишкам, а также за увеличение скорости, начислению очков.

Скрипт собран таким образом, что вначале описывается движение шарика, угол отскока от фишек, верхней границы и ракетки. Угол отскока от фишки и верхней границы равен 180 градусам. Если шарик касается боковых границ, то он отскакивает зеркально, т.е. на тот же угол, который был до касания. При попадании на ракетку написаны 3 варианта отскока, о которых было сказано выше. Также при касании шарика и ракетки воспроизводится звук.

После идет описание случая попадания шарика по фишке, для каждой фишке находящейся на игровом поле свой скрипт. Из-за этого скрипт выглядит большим и сложным, но если присмотреться, то сразу видно, что скрипты удаления фишек с экрана одинаковые, отличается только лишь

посылаемое сообщение, у каждой фишки оно свое. В самом конце стоит условие, которое проверяет значение переменной «фишки», и если оно равно 0, то на экране появляется сообщение о выигрыше.

Проект «Игра «Арканоид»» готова. В дальнейшем возможно создание нескольких уровней игры, различных степеней сложности.

Проведенная экспериментальная проверка разработанной методики преподавания темы «Программирование в среде Scratch» учащимся средней школы на базе ГОУ СОШ «Школы здоровья» №1136 показала свою эффективность. Результатом проведенной работы стало создание учащимися средней школы двух полноценных проектов.

При проведении занятий была замечена заинтересованность учащихся в изучении темы «Программирование в среде Scratch».

Несмотря на то, что Scratch является языком программирования, с помощью которого изучаются такие понятия как алгоритм, скрипт, программа, творческая деятельность учащихся была на хорошем уровне. Ребята с интересом изучали новые возможности, при этом создавались интересные проекты. Примером таких проектов могут служить «Электронное пианино» и «Игра «Арканоид»».

2.3. Контролирующий эксперимент эффективности использования GoogleScratch в учебной деятельности

Цель контролирующего эксперимента – установить эффективность использования выбранного теоретического и практического материала в контексте изучения раздела «Алгоритмика и основы программирования».

Таблица 4

Формы контроля используемы в эксперименте

№	Вид контроля	Форма контроля
1	Текущий	тестирование
2	Промежуточный	письменная работа

3	Итоговый	практическая работа

Текущая форма контроля – тестирование, позволило определить, на каком уровне находятся знания детей, после проведения 1 и 2 урока.

Вопросы текущего тестирования

1. Можно ли сделать проект, в котором не будет сцены?

- да
- нет

2. Может ли спрайт быть больше сцены? Почему?

- да
- нет, спрайты всегда внутри сцены

3. Звуковые файлы - это не обязательный атрибут. А как вы думаете, можно ли создать сцену или спрайт, не добавив ни одного изображения?

- нет
- да

4. Ширина сцены:

- 460
- 480
- 360
- 420
- любая

5. Высота сцены:

- 460
- 480
- 360
- 420
- любая

6. Чему равна координата У в центре сцены?

- 240
- - 240
- 0

- -180
- 180

7. Чему равна координата X в центре сцены?
   - 240
   - - 240
   - 0
   - -180
   - 180

8. Могут ли разные спрайты иметь одинаковые имена и почему?
   - да, для экономии времени
   - нет, чтобы не было путаницы

В результате проведенного тестирование установлено, что в классе, средний уровень оценки, проученной по итогу контроля = 4,1. Оценку выше среднего, получили 23 ребенка (из 26), что составляет 88%. 3 ребенка (12%), выполнили тестирование на оценку «удовлетворительно».

Полученные результаты показывают, что в процессе изучения тем связанных с организацией работы в GoogleScratch, были подобраны верные методы и технологии, в связи с чем, уровень общий уровень выполнения задания достаточно высокий.



Рис. 19. Оценки, полученные в результате текущего тестирования

Промежуточная форма контроля, представленная письменной работы, кроме контроля знаний, активизирует мыслительную, познавательную

активность учеников. Письменная работа, включала в себя задание, создания алгоритма разработки списков по средствам GoogleScratch.

В результате оценки результатов выполнения учениками письменной работы, установлено, что большинство детей (76%), справились с заданием, из них 58% воспользовались дополнительными источниками получения информации (интернет-ресурсы). 24% учеников не справились с заданием.

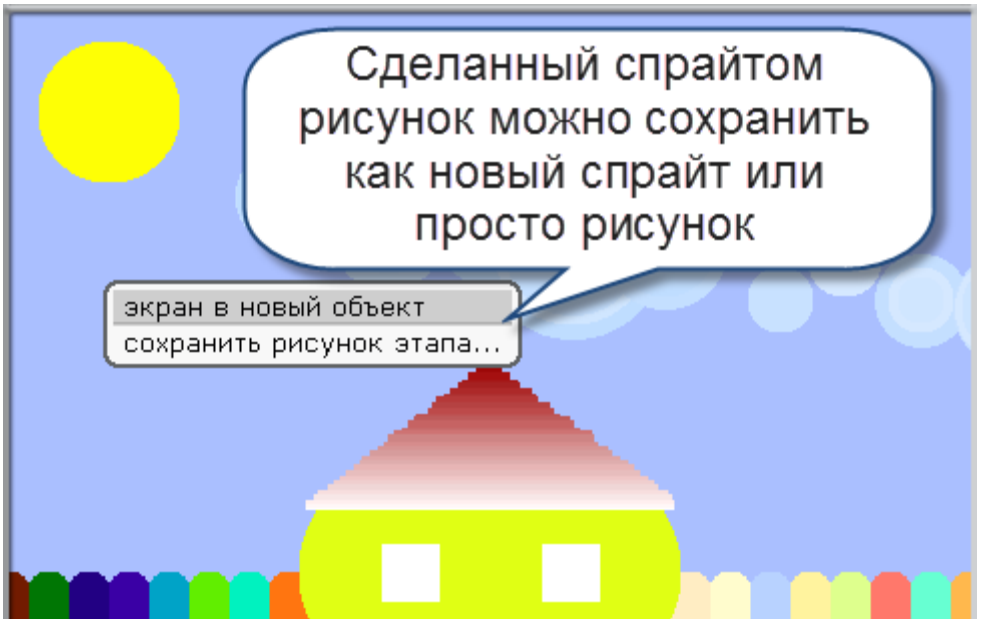
Последняя форма контроля – самостоятельная практическая работа «Домик под солнцем».

Задание к выполнению практической работы:

Сделаем проект, в котором спрайт выступит в роли художника и нарисует домик под солнцем. Подобный фон для сцены мы уже делали в графическом редакторе. При желании сделанный спрайтом рисунок можно сохранить, нажав на правую кнопку мышки и выбрав нужную команду.

Алгоритм создания этого рисунка состоит из 8 шагов:

1. Рисуем небо.
2. Рисуем дорожку.
3. Рисуем забор.
4. Рисуем солнце.
5. Рисуем облака.
6. Рисуем корпус дома.
7. Рисуем окна.
8. Рисуем крышу.
9. Ставим спрайт в угол



Для большей наглядности каждому шагу алгоритма будет соответствовать свой скрипт. А в качестве заголовков будем использовать команду “Когда нажата клавиша 1,2,3...”

1. Рисуем небо



Среди блоков рисования нет команды заливки какой-либо фигуры цветом. Выйдем из положения, установив размер пера в 300 точек. Высота сцены равны 360 точкам, таким образом одним широким мазком мы закрасим нужным цветом почти все пространство. Главное - опустить перо и поднять в нужной точке. В нашем примере мы проводим линию слева-направо от точки: x=-240, y=40 до точки: x=260 (специально выходим за пределы сцены), y=40. Для создания такой линии хорошо подходит команда движения “плыть () секунд в точку x() y()”, которая дает возможность задать и продолжительность создания линии и ее конечную точку. Хотя тот же конечный результат можно получить и командой “идти 480 шагов”: 

. До начала рисунка на всякий случай перо лучше поднимать, иначе вы можете неожиданно обнаружить случайные линии, которые спрайт сделал бесконтрольно.

2. Рисуем дорожку



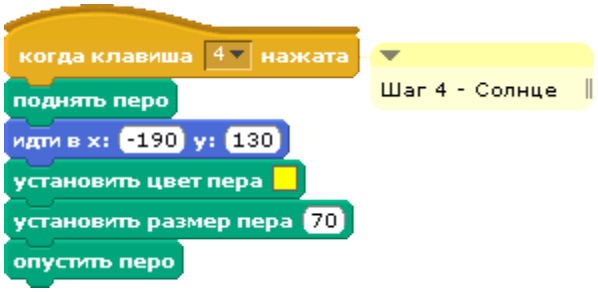
Перенесем перо в нижнюю часть сцены, поменяем его цвет и ширину, чтобы нарисовать линию дорожки.

3. Рисуем забор



Для рисунка забора мы используем цикл, с помощью которого нарисуем 25 разноцветных дощечек забора. А чтобы получить уникальную расцветку, используем блок случайных чисел для цвета и оттенка каждой дощечки.

4. Рисуем солнце



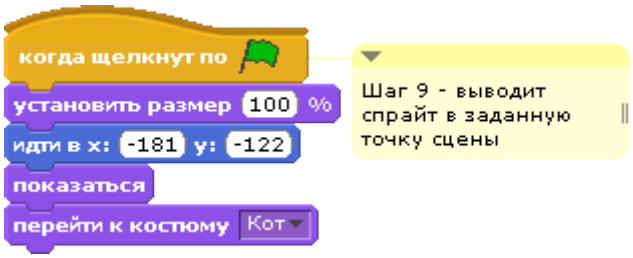
Просто 1 раз опустим перо желтого цвета величиной в 70 точек в указанном месте.

5. Рисуем облака



Подобным образом нарисуем и облака, только поменяем цвет, размер и координаты. Сколько облаков нарисует наш скрипт? Как это можно определить?

6. Рисуем корпус дома



Здесь все очень просто - проведем линию определенного цвета и размера от одной точки координат (x=-40, y=-100) до другой (x=40, y=-100). Размер пера равен высоте корпуса нашего домика.

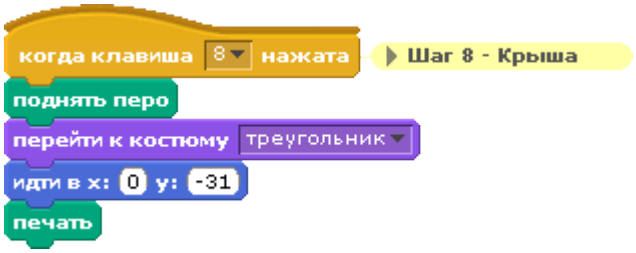
7. Рисуем окна



Т.к. нельзя поменять форму пера для рисования квадратных окон, закрашенных каким-либо цветом (мы сделаем их белыми), нашему художнику нужен специальный костюм - квадрат, который мы отпечатаем 2

раза в тех местах рисунка, где должны быть окна. Сделайте костюм с помощью графического редактора. Назовите его “квадрат”, как показано на рисунке. При желании можно изобразить не просто белый квадрат, а нарисовать занавески, цветок и т.п. Напечатаем 2 окна:

8. Рисуем крышу



Используем тот же подход, как и с окнами, чтобы нарисовать треугольную крышу определенного цвета. Нарисуйте в графическом редакторе костюм в форме треугольника, чтобы можно было напечатать его на месте крыши нашего домика. Назовите его “треугольник”, как показано на рисунке. Напечатайте в нужном месте сцены этот костюм.

9. Ставим спрайт в угол



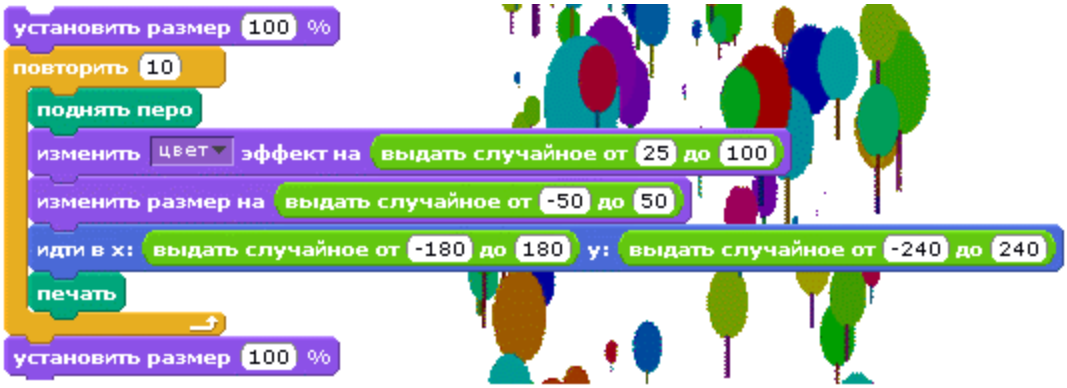
Поставим спрайт в левый нижний угол сцены, указав нужный костюм.

Творческие задания

1. Можно ли изменить проект Домик под солнцем таким образом, чтобы в нем был один скрипт, а не 9, как сейчас? Если вы знаете, как это сделать, внесите изменения в проект. Сохраните полученный результат.

2. Используя команды рисования, в сочетании с блоками контроля и движения, можно создавать программируемые произведения компьютерной графики - живые рисунки, которые меняются во времени и общаются со зрителем. Для создания своего живого полотна вы можете параллельно запустить целую группу спрайтов-художников с перьями разных цветов, оттенков и размеров.

3. Используя блок печати копий спрайтов можно создать лес из одного изображения дерева или цветочную поляну из одного изображения цветка. Чтобы копии были уникальными, можно случайным образом менять цвет, размер и форму оригинала до применения команды печать. Посмотрите на устройство этого скрипта. Попробуйте с его помощью создать сказочный лес из одного дерева!



Анализ результатов выполнения практической работы установил, что 100% учеников, справились с выполнением задания, что показывает эффективность использованных методов и форм обучения в разделе «Алгоритмика и основы программирования».

Заключение

Scratch разрабатывался как новая учебная среда для обучения школьников программированию. В то же время учащиеся в полной мере могут раскрыть свои творческие таланты, так как в нем можно легко создавать фильмы, игры, анимированные открытки и презентации; придумывать и реализовывать различные объекты, определять, как они выглядят в разных условиях, перемещать по экрану, устанавливать способы взаимодействия между объектами. Дети могут сочинять истории, рисовать и оживлять на экране придуманных ими персонажей, учиться работать с графикой и звуком. Важно и то, что ребенок имеет возможность поделиться результатом своего творчества с друзьями или другими пользователями.

Scratch создавался для учащихся начиная от 8 и до 16 лет, но и дети более раннего возраста могут работать в этой среде над проектами вместе с родителями или старшими братьями и сестрами. С другой стороны, даже студенты многих вызов могут использовать Scratch на занятиях в компьютерных классах.

Анализ учебников и учебных пособий показал то, что программирование учащимся средней школы должно тесно идти с развитием творческого потенциала учащихся. Одним из способов организации учебной деятельности учащихся является метод проектов, где подходящим инструментом для реализации является среда программирования Scratch. Так как простота освоения и огромный потенциал отлично сочетаются в одной среде, Scratch, интересен для начального уровня изучения программирования.

В настоящее время информатику начинают изучать уже в средней школе. Возникает вопрос: можно ли в средней школе детей обучать алгоритмизации и программированию?

Можно, но при соблюдении следующих условий:

· для решения алгоритмических задач выбрана понятная и интересная ученикам предметная область;

· программная реализация учебной среды программирования имеет дружественный пользовательский интерфейс;

· для создания алгоритма решения задачи используются визуальные средства представления структур данных и структур управления, не требующие запоминания большого количества служебных слов и синтаксических правил записи программы.

Такая программная среда, в основе которой лежит язык программирования Scratch, была разработана группой ученых из Массачусетского технологического института (MIT).

На основе анализа учебников и возможностей программы нами сделан вывод, что обучение учащихся средней школы программированию в среде Scratch должно проходить поэтапно, от простого к сложному.

Для того чтобы дети эффективно усваивали знания в новой области, необходимо наличие мотивации. А мотивация учебной деятельности возникает, если они чувствуют личную заинтересованность в приобретении знаний для достижения своей цели. Scratch позволяет создавать проекты, которые интересны различным возрастным группам от 8 до 16 лет и старше - интерактивные презентации, мультфильмы, компьютерные игры, программы-тренажеры с использованием графики и мультимедиа.

Проведенная экспериментальная проверка разработанной методики преподавания раздела «Алгоритмика и основа программирования» показало свою эффективность. Результатом проведенной работы стало создание учащимися средней школы нескольких полноценных проектов.

При проведении занятий была замечена заинтересованность учащихся в изучении темы «Алгоритмика и основа программирования».

Несмотря на то, что Scratch является языком программирования, с помощью которого изучаются такие понятия как алгоритм, скрипт, программа, творческая деятельность учащихся была на хорошем уровне. Ребята с интересом изучали новые возможности, при этом создавались интересные проекты. Примером таких проектов могут служить «Электронное пианино» и «Игра «Арканоид»».

Список использованной литературы

1. Приказ Минобрнауки России от 06.10.2009 N 373 (ред. от 18.12.2012) // Бюллетень нормативных актов федеральных органов исполнительной власти. - N 12. - 22.03.2010.
2. Белова Г.В. Программирование в среде ЛОГО. Первые шаги. – М.: Солон, 2007.
3. Бин Нгуен. Объектно-ориентированное программирование на IBM Smalltalk. – М.: Диалог-МГУ, 1996.
4. Бокучава Т.П., Тур С.Н. Методическое пособие по информатике для учителей 2–4 классов. М.: BHV, 2007.
5. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений – М.: Вильямс, 2008.
6. Бородин Г. Аниматоры: краткая история обучения профессии. Электронный ресурс: Библиотека театра. URL: http://biblioteka.teatr-obraz.ru/page/animatory-kratkaya-istoriya-obucheniya-professii-georgiy-borodin (дата обращения: 18.02.2016г.)
7. Вано И. Рисованный фильм. - М.:Госкиноиздат, 1950.
8. Великович Л., Цветкова М. Программирование для начинающих. – М.: Бином, 2007.
9. Вентцель К.Н. Избранные произведения. - М.: Изд. дом Шалвы, 1999.
10. Гейн А.Г., Сенокосов А.И., Юнерман Н.А. Информатика: Учебник для 10–11 классов общеобразовательных учреждений. – М.: Просвещение, 2005.

11. Гейн А.Г. Информатика. 10 класс. Учебник. М.: Просвещение, 2008.

12. Горячев А.В., Волкова Т.О., Горина т.и. Информатика в играх и задачах: Учебник-тетрадь для 2 класса четырехлетней начальной школы: В 2 томах. М.: Баласс, 2006.

13. Еремин Е.А. Газета «Информатика». Среда Scratch – первое знакомство. – М.: Первое сентября, 2008 – №20 (573) – С. 17–24.

14. Еремин Е.А. Газета «Информатика». Среда Scratch – первое знакомство. – М.: Первое сентября, 2008 – №20 (573) – С. 16–28.

15. Ершов А.П. и др. Основы информатики и вычислительной техники. Учебник для 10–11-ых классов средних школ. – М.: Просвещение, 1985.

16. Землянская Е.Н. Учебные проекты младших школьников // Начальная школа, 2005. № 9.

17. Игушева И.А. Сетевое образовательное сообщество. URL: http://sandbox.openclass.ru/wiki-pages/79809,http://www.academy.edu.by/sites/logoped/kommtechno.htm (дата обращения: 20.02.2016г.)

18. Каймин В.А. Информатика. Учебник. М.: Инфра-М, 2004.

19. Каймин В.А. Информатика. Учебник. М.: Инфра-м, 2001.

20. Кнут Е. Дональд. Искусство программирования. Том 1. Основные алгоритмы. – М.: Вильямс, 2007.

21. Конопатова Н.К., Матвеева Н.В., Панкратова Л.П., Челак Е.Н., Нурова Н.А. Учебник для 4 класса. М.: Бином. Лаборатория знаний, 2007.

22. Конопатова Н.К., Матвеева Н.В., Челак Е.Н. Рабочая тетрадь для 4 класса, М.: Бином. Лаборатория знаний, 2006.

23. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы. Построение и анализ. 2-е издание, 2007.

24. Комаровский Б.Б. Русская педагогическая терминология. - М.: Просвещение, 1967.

25. Конышева Н. М. Проектная деятельность школьников. Современное состояние и проблемы. Начальная школа, 2006.

26. Красиков И.В., Красикова И.Е. Алгоритмы. Просто как дважды два. – М.:ИКФ «ЭКСМОС», 2006.

27. Кузнецов А.А., Апатова Н.В. Основы информатики. 8–9 классы. – М.: Дрофа, 2003.

28. Кушниренко А.Г., Лебедев Г.В., Зайдельман Я.Н. Информатика 7–9 классы. – М.: Дрофа, 2002.

29. Лаптев В. С++. Объектно-ориентированное программирование: Учебное пособие. – СП.: Учебное издание, 2008.

30. Левитин А. Алгоритмы. Введение в разработку и анализ. – М.: Вильямс, 2006.

31. Михайлова Ю.В. Дом свободного ребенка // Советская педагогика, 1983. № 4. с.102–106.

32. Назарова Е. В. Метод проектов в процессе формирования основ экологического сознания учащихся, 2009.

33. Первин Ю.А. Методика раннего обучения информатике: Методическое пособие для учителей начальной школы и методистов Изд. 1-е/ 2-е. М.: Бином. Лаборатория знаний, 2008.

34. Полат Е.С. Новые педагогические и информационные технологии в системе образования. – М. Академия. – 2006.

35. Савенков А.И. Методика исследовательского обучения младших школьников: / Савенков А.И – Самара: Учебная литература, 2008. – 119с.

36. Семакин И.Г., Залогова Л.А., Русаков Н.С., Шестакова Л.В. Информатика. Базовый курс. 9 класс. – М.: Лаборатория базовых знаний, 2002.

37. Семакин И.Г., Хеннер Е.К. Информатика и ИКТ. Базовый уровень: учебник для 10–11 классов. – М.: БИНОМ, 2007.

38. Семакин И.Г., Хеннер Е.К., Шеина Т.Ю. Практикум по информатике и ИКТ для 10–11 классов. Базовый уровень. – М.: БИНОМ, 2007.

39. Симонович С. В. Информатика для юристов и экономистов. — СПб.: Питер, 2001. — 688с.

40. Сорокина В.В. Психологическое неблагополучие детей в начальной школе. – М.: Генезис, 2005.

41. Степашко Л.А. Философия и история образования. - М.: Флинтас, 1999.

42. Ожегов С.И. Толковый словарь русского языка. – М: Оникс, Мир и образование, 2008.

43. Угринович Н.Д. Информатика и информационные технологии. Учебник для 10–11 классов. – М.: Лаборатория базовых знаний, 2006.

44. Шлинке Т. Н. Метод проектов как одно из условий повышения мотивации обучения учащихся Начальная школа, 2008. № 9.

45. Юдина А. Информатика. Методическое пособие к практикуму по информатике в среде Logo Writer. М.: Мнемозина, 2005.

46. Яшуев Р.Н. Работа со школьниками в области информатики. – М., 2007.

47. Яховский Н.Г. Обучение программированию в начальной школе. – М., 2008.