



# РАЗДЕЛ 1. КОМПЛЕКС ОСНОВНЫХ ХАРАКТЕРИСТИК ПРОГРАММЫ

## ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Дополнительная общеобразовательная общеразвивающая программа «Программирование в среде Scratch» предназначена для обучающихся 10-12 лет, проявляющих интерес к программированию.

Дополнительная общеобразовательная общеразвивающая программа «Программирование в среде Scratch» (далее – программа) имеет **техническую направленность** и ориентирована на научно-техническую подготовку детей начальной школы, формирование творческого технического мышления. Неотъемлемой составляющей обучения является деятельность, направленная на развитие личности, создание условий для самоопределения и социализации обучающегося.

**Актуальность программы** определяется тем, что на сегодняшний день программирование становится один из ключевых навыков, которому нужно обучать ребенка уже с младших классов. Компьютерный код - тот же иностранный язык, только он позволяет разговаривать с компьютером, ставить ему задачи и контролировать их выполнение. И знание этого языка сегодня так же необходимо, как знание иностранного языка или математики.

Многие талантливые программисты начинали свой путь еще в детстве - с создания простых игр и мультфильмов. Это интересно и понятно ребенку. Именно поэтому в данной программе предлагаются простые проекты игр и анимации, выполняя которые, ребенок будет учиться программировать.

Scratch — это визуальный язык программирования, в котором программа складывается из разноцветных логических блоков. Детям ничего не нужно писать, как в других языках программирования. Блоки имеют защёлки, которые не позволяют соединить несовместимые блоки.

Одним из самых известных неформальных способов организации внеучебной образовательной деятельности по программированию является метод проектов. Самым подходящим инструментом для организации такой деятельности является среда Scratch, которая есть современное направление компьютерного дизайна и анимации. Овладев даже минимальным набором операций, самый неискущённый пользователь может создавать законченные проекты. Scratch - это среда, которая позволяет детям создавать собственные анимированные и интерактивные истории, презентации, модели, игры и другие произведения. Работа в среде Scratch позволяет, с одной стороны, организовать среду для самореализации и самоутверждения учащихся, и, с другой стороны, сформировать у них тягу к творчеству и знаниям и дать подходящие средства её реализации. Быть успешным в такой среде становится проще.

Scratch можно рассматривать как инструмент для творчества, позволяющий параллельно в игровом формате изучать базовые принципы программирования. Обучающиеся смогут сочинять истории, рисовать и оживлять на экране придуманных ими персонажей, учиться работать с графикой и звуком. Применений возможностям Scratch можно найти множество: в этой среде легко создавать анимированные открытки, мини-игры, мультфильмы. В результате выполнения простых команд может складываться сложная модель, в которой будут взаимодействовать множество объектов, наделенных различными свойствами. Начальный уровень программирования настолько прост и доступен, что Scratch рассматривается в качестве средства обучения детей младшего школьного возраста.

Изучение Scratch может серьезно помочь обучающимся освоить азы алгоритмизации и программирования, а полученные знания пригодятся для дальнейшего и более серьезного изучения программирования. Работа в среде Scratch ведется так же, как средство подготовки детей к всевозможным конкурсам и выставкам по данной тематике, которые в настоящее время набирают большие обороты.

Республика Татарстан уже в течение довольно длительного времени не только формирует региональную систему развития дополнительного образования детей, но и развивает политику, поддерживая деятельность детских объединений. На сегодняшний день наш регион, считаясь одним из лидеров в сфере развития детских объединений, обладает колоссальным опытом и значительными

ресурсами для обновления и дальнейшего развития системы развития дополнительного образования детей.

**Актуальность данной программы** заключается в том, что она позволяет осуществить социальный заказ, обусловленный значимостью информатизации и цифровизации современного общества; активизировать познавательную деятельность обучающихся; реализовать их интерес к техническому направлению. Программа дает возможность реализовать обучающимся свои творческие и познавательные способности посредством программирования, создания и анимирования моделей в среде Scratch.

**Педагогическая целесообразность** заключается в разностороннем раскрытии индивидуальных способностей обучаемого, развитию у него интереса к различным видам познавательной деятельности, социализации.

Изучая программирование с младшего школьного возраста, у обучающихся формируется не только логическое мышление, но и навыки работы с мультимедиа, создаются условия для активного, поискового обучения, предоставляются широкие возможности для проектной деятельности. Изучение программирования в графической среде Scratch позволяет организовать процесс обучения в игровой форме, что делает содержание программы доступным и позволяет вовлечь в процесс в том числе учащихся младшего школьного возраста. Разрабатывая творческие проекты, учащиеся учатся работать в команде, планировать свою деятельность, ставить и решать поставленные задачи.

Эффективным для технического развития детей является не только обучение программированию, но и создание условий для самовыражения личности обучающегося через представление продукта своего труда. Программа открывает новый мир, предоставляет возможность в процессе работы приобретать такие социальные качества как любознательность, активность, самостоятельность, ответственность, навыки продуктивного сотрудничества.

**Практическая значимость программы** заключается в помощи ребенку сделать первые шаги в мире программирования, познакомиться с сообществом таких же заинтересованных ребят, подробностями и тонкостями проектной деятельности. Овладевая навыками программирования, ребенок затрагивает и смежные сферы: логика, вычислительная математика, теория вероятности, а также и другие научные области: география, биология, физика и др.- в зависимости от интересов ребенка.

Когда у ребенка сформирован необходимый набор знаний и умений, выполнен ряд задач по разным темам, он может, используя их, работать над собственным проектом. Это позволяет развивать творческие способности, проводить собственные исследования, работать в команде, и, что немаловажно, видеть результат собственной работы, вносить в нее коррективы и развивать ее.

Среда Scratch предоставляет ребенку прекрасную возможность учиться на собственном опыте. Такие знания вызывают у детей желание двигаться на пути открытий и исследований, а любой признанный и оцененный успех добавляет уверенности в себе. Обучение происходит особенно успешно, когда ребенок вовлечен в процесс создания значимого и осмысленного продукта, который представляет для него интерес. Важно, что при этом ребенок сам строит свои знания, а педагог консультирует его.

**Нормативно-правовой базой** разработки дополнительной общеобразовательной общеразвивающей программы «Программирование в среде Scratch» (далее – программа) являются:

- Федеральный закон Российской Федерации от 29.12.2012 № 273-ФЗ «Об образовании в Российской Федерации»;
- Закон Республики Татарстан от 22 июля 2013 года N 68-ЗРТ "Об образовании";
- Указ Президента РФ от 21 июля 2020 г. № 474 «О национальных целях развития России до 2030 года»;
- Распоряжение Правительства РФ от 31.03.2022 N 678-р "Об утверждении Концепции развития дополнительного образования детей и признании утратившим силу Распоряжения Правительства РФ от 04.09.2014 N 1726-р (вместе с Концепцией развития дополнительного образования детей до 2030 года, Планом мероприятий по реализации Концепции развития дополнительного образования детей до 2030 года, I этап (2022 - 2024 годы))";
- Приказ Министерства просвещения РФ от 27 июля 2022 года N 629 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам»;

- Приказ Министерства образования и науки Республики Татарстан от 25.07.2017 № под-1266/17 «Об утверждении отраслевой стратегии развития образования Республики Татарстан на 2017 – 2021 годы и на период до 2030 года»;
- Приказ Министерства труда и социальной защиты Российской Федерации от 22 сентября 2021 года N 652н «Об утверждении профессионального стандарта "Педагог дополнительного образования детей и взрослых"»;
- Методические рекомендации по проектированию дополнительных общеразвивающих программ (включая разноуровневые программы) – Письмо Минобрнауки России от 18.11.2015 № 09-3242 О направлении информации (вместе с Методическими рекомендациями по проектированию дополнительных общеразвивающих программ (включая разноуровневые программы));
- Постановление Главного государственного санитарного врача Российской Федерации от 28 сентября 2020 г. № 28 «Об утверждении санитарных правил СП 2.4.3648-20 «Санитарно-эпидемиологические требования к организациям воспитания и обучения, отдыха и оздоровления детей и молодежи».

**Уровень сложности содержания (уровень освоения).** Программа имеет **стартовый уровень** (ознакомительный, общедоступная сложность содержания программы). Программа предполагает использование и реализацию общедоступных и универсальных форм организации материала, минимальную сложность предлагаемого для освоения содержания программы.

**Новизна программы** заключается в том, что в основу программы положены основные принципы и тенденции развития современной методики обучения в рамках дополнительного образования детей:

- повышение познавательной мотивации;
- коммуникативная направленность;
- индивидуальный подход к обучающемуся.

**Отличительной особенностью** программы является то, что ребенку предлагаются занятия не только обучающего характера, а развивающего и познавательного. Ребенок втянут в процесс игры, и там обретает необходимые для него знания и умения. Все задания и упражнения направлены на комплексное развитие, доставляя ребенку радость от игры и общения. Новое знание не даётся детям в готовом виде, а постигается ими путём самостоятельного анализа, сравнения, выявления существенных признаков. Таким образом, знание входит в жизнь детей как «открытие» закономерных связей и отношений окружающего мира.

Программа имеет практическую направленность. Основное содержание реализуется в форме творческих занятий. Особенностью программы является её простота и доступность, обеспечивающих максимальное удобство организации образовательного процесса для каждого обучающегося.

В основу реализации программы положен учет возрастных и индивидуальных особенностей детей.

Ведущая идея, на которой базируется программа: каждый обучающийся есть неповторимая индивидуальность, обладающая свойственными только ей психическими, физическими и прочими особенностями. Необходимо всестороннее изучение этих особенностей и творческий, комплексный подход к формам и методам их развития.

В разработке программы учитывается научное положение Л.С. Выготского о том, что правильно организованное обучение «ведёт» за собой развитие. Поэтому программа включает в себя большое разнообразие заданий, выполняющих развивающие, обучающие, воспитательные задачи. В основе образовательного процесса лежат педагогические технологии личностно-ориентированного обучения, игровые технологии, а также методы развития познавательной активности.

**Адресат программы:** образовательная программа рассчитана на детей 10-12 лет.

**Особенности возрастной группы детей, которым адресована программа.** В возрасте 10-12 лет решаются специфические задачи личностного развития и взросления человека, идет интенсивное усвоение культурных ценностей, определяющих в дальнейшем его главные жизненные предпочтения. В этот период детям свойственна повышенная активность, стремление к деятельности, происходит уточнение границ и сфер интересов, увлечений. В этот период ребенку становится интересно многое, далеко выходящее за рамки его повседневной жизни. Его начинают интересовать вопросы прошлого и будущего, экологические, технические и социальные темы, возможности познания мира. Многие исследователи рассматривают этот возраст как период “зенита любознательности”, по сравнению с младшими и старшими детьми. Однако эта любознательность весьма поверхностна.

В качестве одной из важнейших потребностей 10-12-летних детей можно выделить потребность в положительной оценке себя во взаимодействии со сверстниками и значимыми взрослыми, в проявлении собственной внутренней позиции.

В этом возрасте, в процессе межличностного взаимодействия детей со сверстниками и значимыми взрослыми происходит рефлексивный оборот на себя. При решении той или иной задачи ребенок ориентируется не только на объективные условия и образец действия, но и на собственные качества (особенности, умения, знания, черты характера) как на решающее условие ее решения. Этот рефлексивный оборот является системообразующим механизмом формирования новообразований детей 10-12-летнего возраста.

Важной потребностью детей 10-12-летнего возраста является и потребность эмоционального самовыражения и взаимодействия. Так как эмоциональная сфера является неотъемлемой от рациональной, когнитивной в структуре самосознания, для развития понятийного и абстрактного мышления ребенку необходимо эмоциональное наполнение его деятельности, общения и поведения.

При построении учебного процесса необходимо учитывать индивидуальные особенности познавательной деятельности обучающихся, но большое внимание уделять игре, созданию ситуации успеха. Дело в том, что учащиеся этой возрастной группы стремятся добиться поставленной цели в течение одного занятия и желают видеть наглядный результат своего труда.

Основные задачи развития на этом возрастном этапе – развитие логического мышления, умения оперировать полученной информацией, развитие самостоятельности детей в учебной деятельности. Для этого необходимо создание учебной ситуации, способствующей удовлетворению познавательных потребностей детей.

**Условия набора обучающихся.** Программа предусматривает свободный набор детей, желающих начать изучать программирование в среде Scratch. Уровень подготовки не требуется, так как программа рассчитана на стартовый (ознакомительный) уровень. Вступительные испытания не предусмотрены.

Группы обучающихся формируются на основе свободного набора, постоянного состава. Набор проводится по заявлению родителей (законных представителей).

**Количество детей в группе:** 4-8 человек.

**Объем программы** - общее количество учебных часов, запланированных на весь период обучения, необходимых для освоения программы - 36 учебных часов.

**Сроки реализации программы:** 5 месяцев.

**Режим занятий:** на реализацию программы отводится 2 часа в неделю (два занятия по 45 мин с 10-минутным перерывом), всего 36 учеб. часов. Занятия проводятся 1 раз в неделю по 2 учебных часа.

Для всех занятий учебный час устанавливается продолжительностью 45 минут.

**Формы организации образовательного процесса:** индивидуально-групповые; мини-группы, занятия с использованием индивидуального подхода к каждому ребёнку.

Занятия по программе состоят из теоретической и практической части. Форму занятий можно определить как интерактивное, практическое обучение (практические занятия), теоретическое обучение. Основной формой обучения является практическая работа, которая выполняется малыми группами (2-3 человека) или индивидуальная работа.

**Особенности организации образовательного процесса:**

Форма обучения по программе: индивидуально-групповая.

Занятия в малых группах проводятся при реализации учебного плана с учетом потребностей обучения.

Индивидуальная форма работы используется при общении с конкретными учащимися. Такой подход используется для более детальной отработки навыков и умений, помогает развитию индивидуальных особенностей обучающихся.

*Принципы построения образовательной деятельности:*

- общепедагогические принципы (обучение, воспитание);
- от простого к сложному;
- принцип доступности.

Учитывая возрастные особенности обучающихся, образовательный процесс по реализации программы разноплановый, основной формой организации учебной деятельности является комплексные занятия.

В процессе реализации программы используются следующие методы обучения:

*Словесные методы обучения:* рассказ; беседа; объяснение; игра.

*Наглядные методы обучения:* демонстрационный; иллюстративный; наблюдения и др.

Формы организации занятий может варьироваться выбирается с учетом возрастных особенностей детей, уровня освоения учащимися программы и их достижений.

*Структура занятия:*

1. Организационный этап.
2. Мотивационный этап (демонстрация или сюжет, ситуация).
3. Постановка проблемы или задачи.
4. Обсуждение–поиск путей решения (в группах различного состава, в зависимости от задачи).
5. Проектирование и программирование.
6. Подготовка демонстрации.
7. Заключительный этап: презентация работ обучающихся друг другу.

**Форма обучения:** очная.

**Язык обучение:** русский.

**Документ по окончании прохождения обучения:** не предусмотрен.

### **Цель и задачи программы**

**Цель программы** – формирование и развитие познавательных и творческих способностей обучающихся, удовлетворение их индивидуальных потребностей в интеллектуальном совершенствовании посредством освоения основ программирование в среде Scratch.

**Задачи программы:**

*Обучающие задачи:*

- сформировать представление об основах программирования в Scratch;
- освоить основные инструменты и операции работы в Scratch;
- изучить основные принципы создания простых проектов в среде Scratch (анимация и игры);
- научить рисовать в векторном и растровом графических редакторах, изменять звук, вводить, выводить и обрабатывать информацию в среде Scratch;

*Развивающие задачи:*

- развивать познавательный интерес, внимание, память;
- развивать логическое, абстрактное и образное мышление;
- развивать воображение и фантазию через создание простых проектов в среде Scratch (анимация и игры);
- развить способности к самостоятельной работе и анализу проделанной работы;
- развивать познавательную и творческую активность в использовании информационных технологий;

*Воспитательные задачи:*

- воспитывать самостоятельность, уверенность в своих силах;
- формировать творческий подход к поставленной задаче;
- воспитывать ценностное отношение к знаниям, интерес к изучению нового;
- воспитывать стремление добиваться поставленной цели;
- воспитывать чувство ответственности за свою работу.

Реализация этих задач будет способствовать дальнейшему формированию взгляда обучающего на мир, раскрытию роли информатики в формировании естественнонаучной картины мира, развитию

мышления, в том числе формированию алгоритмического стиля мышления, подготовке к жизни в информационном обществе.

## **Планируемые результаты обучения**

### **Предметные результаты:**

В результате обучения по данной программе обучающиеся:

- познакомятся с интерфейсом Scratch;
- освоят основные инструменты и операции работы в Scratch, команды движения;
- научатся создавать простые проекты в среде Scratch (анимация и игры);
- научатся использовать векторный и растровый графический редактор, изменять звук, вводить, выводить и обрабатывать информацию в среде Scratch;

### **Метапредметные результаты:**

- развитие познавательного интереса, внимания, памяти;
- развитие логического, абстрактного и образного мышления;
- развитие воображения и фантазии через создание простых проекты в среде Scratch (анимация и игры);
- формирование навыка сознательного и рационального использования программирования в своей деятельности;
- внесение корректив в текущую деятельность на основе анализа изменений для получения запланированных характеристик продукта/результата;
- умение выбирать варианты и самостоятельно искать средства/ресурсы для решения задачи и достижения цели;

### **Личностные результаты:**

- воспитание самостоятельности, уверенности в своих силах;
- использование творческого подхода к поставленной задаче;
- воспитание ценностного отношения к знаниям, интереса к изучению нового;
- сформированное чувство ответственности за свою работу;
- готовность и способность к созданию творческих работ созидательной направленности (игр, анимационных роликов и т.п.).

## Содержание программы

### Основные разделы программы:

Раздел I. Интерфейс программы Scratch (1 ч.)

Раздел II. Начало работы в среде Scratch (4 ч.)

Раздел III. Основные скрипты программы Scratch (18 ч.)

Раздел IV. Работа с несколькими объектами. Синхронизация их работы (4 ч.)

Раздел V. Использование программы Scratch для создания мини-игр (9 ч.)

### Учебный план

№ п/п	Название раздела, темы	Количество часов <sup>1</sup>			Формы аттестации (контроля)
		Всего	Теория	Практика	
<b>1.</b>	<b>Раздел I. Интерфейс программы Scratch (1 ч.)</b>				
1.1.	Тема 1.1. Введение. Что такое Scratch. Основные алгоритмические конструкции. Знакомство с интерфейсом программы Scratch	<b>1</b>	<b>1</b>	-	Педагогическое наблюдение
<b>2.</b>	<b>Раздел II. Начало работы в среде Scratch (4 ч.)</b>				
2.1.	Тема 2.1. Сцена. Редактирование фона. Добавление фона из файла.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
2.2.	Тема 2.2. Понятие спрайтов. Добавление новых спрайтов. Рисование новых объектов.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
<b>3.</b>	<b>Раздел III. Основные скрипты программы Scratch (18 ч.)</b>				
3.1.	Тема 3.1. Синий ящик – команды движения. Темно-зеленый ящик – команды рисования.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
3.2.	Тема 3.2. Фиолетовый ящик – внешний вид объекта. Оживление объекта с помощью добавления костюмов.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
3.3.	Тема 3.3. Желтый ящик – контроль. Лиловый ящик – добавление звуков.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
3.4.	Тема 3.4. Использование в программах условных операторов.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
3.5.	Тема 3.5. Функциональность работы циклов. Цикличность выполнения действий в зависимости от поставленных условий.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
3.6.	Тема 3.6. Зеленый ящик – операторы. Использование арифметических и логических блоков вместе с блоками управления.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение

<sup>1</sup> Для всех занятий учебный час устанавливается продолжительностью 45 минут.

3.7.	Тема 3.7. События. Оранжевый ящик – переменные.	2	1	1	Педагогическое наблюдение
3.8.	Тема 3.8. Списки.	2	1	1	Педагогическое наблюдение
3.9.	Тема 3.9. Голубой ящик – сенсоры. Ввод-вывод данных.	2	1	1	Педагогическое наблюдение
<b>4.</b>	<b>Раздел IV. Работа с несколькими объектами. Синхронизация их работы (4 ч.)</b>				
4.1.	Тема 4.1. Последовательность и параллельность выполнения скриптов	2	1	1	Педагогическое наблюдение
4.2.	Тема 4.2. Взаимодействие между спрайтами. Управление через обмен сообщениями.	2	1	1	Педагогическое наблюдение
<b>5.</b>	<b>Раздел V. Использование программы Scratch для создания мини-игр (9 ч.)</b>				
5.1.	Тема 5.1. Виды компьютерных игр. Алгоритмическая разработка листинга программы.	2	1	1	Педагогическое наблюдение
5.2.	Тема 5.2. Разработка базовых спрайтов для игры. Формирование базовых скриптов.	2	1	1	Педагогическое наблюдение
5.3.	Тема 5.3. Синхронизация работы скриптов для разных спрайтов	2	1	1	Педагогическое наблюдение
5.4.	Тема 5.4. Переход из одной сцены в другую. Создание интерфейса игры.	2	1	1	Педагогическое наблюдение
5.5.	Тема 5.5. Сообщество Scratch в Интернете. Просмотр и публикация проектов.	1	-	1	Педагогическое наблюдение
<b>Итого часов</b>		<b>36</b>	<b>18</b>	<b>18</b>	

## Содержание учебного плана

### Раздел I. Интерфейс программы Scratch

#### **Тема 1.1. Введение. Что такое Scratch. Основные алгоритмические конструкции. Знакомство с интерфейсом программы Scratch.**

**Теория.** История создания среды Scratch. Основные базовые алгоритмические конструкции и их исполнение в среде Scratch. Понятие исполнителя, алгоритма и программы, их назначение, виды и использование. Виды управления исполнителем. Способы записи алгоритма. Основные характеристики исполнителя. Система команд исполнителя. Понятие проект, его структура и реализация в среде Scratch. Основные компоненты проекта Scratch: спрайты и скрипты. Принцип создания анимации и движения объектов. Листинг программы. Сцена. Текущие данные о спрайте. Стилль поворота. Закладки. Панель инструментов, Новый спрайт. Координаты мышки. Режим представления. Окно скриптов. Окно блоков. Блоки стека. Блоки заголовков. Блоки ссылок.

### Раздел II. Начало работы в среде Scratch

#### **Тема 2.1. Сцена. Редактирование фона. Добавление фона из файла.**

**Теория.** Сцена. Ширина и высота сцены. Текущие координаты объекта. Редактирование текущего фона. Вставка нового фона из файла. Вставка стандартного фона из библиотечного модуля среды. Рисование фона в графическом редакторе. Создание нескольких фонов в одной сцене (1 час).

**Практика.** Создание фона сцены на выбранную учащимся тему (1 час).

## **Тема 2.2. Понятие спрайтов. Добавление новых спрайтов. Рисование новых объектов.**

**Теория.** Стандартный объект. Спрайты. Список спрайтов. Редактор рисования для создания новых спрайтов. Инструменты рисования (кисточка, линия, текст, эллипс, ) и редактирования объекта (ластик, заливка, поворот, выбор, печать, пипетка). Центрирование костюма. Масштабирование спрайта. Загрузка на сцену спрайтов из стандартной коллекции среды Scratch. Вставка спрайтов из файлов форматов JPG, BMP, PNG, GIF. Выбор случайного спрайта. Удаление спрайтов (1 час).

**Практика.** Создание фона сцены и прорисовка основных спрайтов для Scratch-истории. (1 час).

## **Раздел III. Основные скрипты программы Scratch**

### **Тема 3.1. Синий ящик – команды движения. Темно-зеленый ящик – команды рисования.**

**Теория.** Команды – *идти; повернуться направо (налево); повернуть в направлении; повернуться к; изменить x (y) на; установить x (y) в; если край, оттолкнуться.* Принципиальное различие действия команд *идти в* и *плыть в*. Назначение сенсоров *положение x, положение y* и *направлении*. Команды – *очистить, опустить перо, поднять перо, установить цвет пера, изменить цвет пера на, установить цвет пера, изменить тень пера, установить тень пера, изменить размер пера на, установить размер пера, печать* (1 час).

**Практика.** Создание программ для передвижения спрайтов по сцене. Создание программ для рисования различных фигур (1 час).

### **Тема 3.2. Фиолетовый ящик – внешний вид объекта. Оживление объекта с помощью добавления костюмов.**

**Теория.** Костюмы спрайта. Копирование и редактирование костюма спрайта с помощью редактора рисования. Переупорядочивание костюмов. Команды – *перейти к костюму, следующий костюм, говорить...в течении...секунд, сказать, думать, думать...секунд, изменить ....эффект на, установить эффект...в значение, убрать графические эффекты, изменить размер на, установить размер, показаться, спрятаться, перейти в верхний слой, перейти назад на...l слоев.* Назначение сенсоров *костюм* и *размер*. Понятие раскадровки движения. Изменение костюма спрайта для имитации движения (1 час).

**Практика.** Создание программы для управления внешним видом объекта. Создание Scratch-историй с имитацией хождения и движения объектов (1 час).

### **Тема 3.3. Желтый ящик – контроль. Лиловый ящик – добавление звуков.**

**Теория.** Кнопка с зеленым флажком и ее назначение. Управление последовательностью выполнения скриптов. Понятие управляющих сообщений. Команды – *передать, передать и ждать, когда я получу.* Скрипты для создания условных конструкций программы – *если, если...или.* Скрипты для управления циклами – *всегда, повторить, всегда, если, повторять до..* Команды – *когда клавиша...нажата, когда щелкнут по, ждать...секунд, ждать до, остановить скрипт, остановить все.* Загрузка звуков из стандартной коллекции и из файлов жесткого диска. Запись звука через микрофон. Принципиальная разница работы команд *играть звук* и *играть звук до завершения.* Команды – *остановить все звуки, барабану играть...тактов, оставшиеся...тактов, ноту...играть...тактов, выбрать инструмент, изменить громкость, установить громкость, изменить темп на, установить темп.* Назначение сенсоров *громкость* и *темп* (1 час).

**Практика.** Создание программ с элементами управления объектом. Озвучивание Scratch-историй (1 час).

### **Тема 3.4. Использование в программах условных операторов.**

**Теория.** Базовая конструкция ветвление, назначение, виды (полная и неполная форма). Понятие условия. Изменение порядка выполнения скриптов в зависимости от условия. Разветвление листинга программы. Скрипты условных операторов. Использование неполной формы ветвления в системе Scratch (1 час).

**Практика.** Создание программ с изменением последовательного выполнения скриптов при наличии условий (1 час).

### **Тема 3.5. Функциональность работы циклов. Цикличность выполнения действий в зависимости от поставленных условий.**

**Теория.** Циклы с фиксированным числом повторений. Заголовок цикла. Тело цикла. Циклы с условным оператором. Заголовок цикла. Тело цикла. Предусловие и постусловие. Зацикливание (1 час).

**Практика.** Создание программ с использованием циклов с фиксированным числом повторений. Создание программ с использованием циклов с предусловием и постусловием (1 час).

### **Тема 3.6. Зеленый ящик – операторы. Использование арифметических и логических блоков вместе с блоками управления.**

**Теория.** Числа. Строинги. Логические величины. Логические выражения. Арифметические операции. Логические операции. Операции сравнения. Команды для работы со строингами – *слить, буква...в, длина строки*. Команда *выдать случайное от...до*. Использование арифметических и логических блоков в листинге программы. Просмотр полученного результата (1 час).

**Практика.** Создание программ с использованием операций сравнения данных. Создание программ с использованием арифметических данных и логических операций (1 час).

### **Тема 3.7. События. Оранжевый ящик – переменные.**

**Теория.** События в проектах Scratch. Понятие переменных и необходимость их использования в листинге программы. Глобальные и локальные переменные. Имя переменной и правила его формирования. Команды для переменных - *поставить...в, изменить...на, показать переменную, спрятать переменную*. Удаление переменных. Создание счетчиков с помощью переменных (1 час).

**Практика.** Разработка сценария Scratch-историй с несколькими событиями. Создание проектов с использованием глобальных и локальных переменных (1 час).

### **Тема 3.8. Списки.**

**Теория.** Создание списков и необходимость их использования в проектах Scratch. Добавление в список данных. Удаление данных из списка. Удаление списка. Команды работы со списками – *добавить...к, удалить...из, поставить...в...из, заменить элемент...в...на, элемент...из, длина списка* (1 час).

**Практика.** Создание программ-тестов по принципу сравнения данных из нескольких списков (1 час).

### **Тема 3.9. Голубой ящик – сенсоры. Ввод-вывод данных.**

**Теория.** Понятие сенсора. Правила применения и область действия команд *касается, касается цвета и цвет.касается*. Функционал команды *спросить...и ждать*. Сенсоры *мышка по x, мышка по y, мышка нажата?, клавиша...нажата?, расстояние до, перезапустить таймер*. Сенсоры, значение которых можно выводить на экран – *ответ, таймер, громкость, громко?, ...значение сенсора и сенсор....* Необходимость ввода данных для их обработки в программе. Ввод данных с помощью команды *спросить*. Вывод конечного результата обработки с помощью команд *говорить* и *сказать* (1 час).

**Практика.** Создание проектов с использованием значений сенсоров и команды *спросить*. Создание программ для обработки данных пользователя с выводом на экран конечного результата (1 час).

## **Раздел IV. Работа с несколькими объектами. Синхронизация их работы**

### **Тема 4.1. Последовательность и параллельность выполнения скриптов**

**Теория.** Последовательные и параллельные потоки в программах Scratch. Одновременная и попеременная работа нескольких исполнителей (1 час).

**Практика.** Создание Scratch-историй с одновременной и попеременной работой нескольких исполнителей (1 час).

#### **Тема 4.2. Взаимодействие между спрайтами. Управление через обмен сообщениями.**

**Теория.** Решение проблемы появления новых исполнителей только после того, как старые исполнители выполнили свои действия. Взаимодействие спрайтов с неподвижными объектами с помощью команд *касается* и *касается цвета*. Взаимодействие спрайтов с помощью команд *передать* и *когда я получу*. Использование сообщений для создания событий (1 час).

**Практика.** Создание Scratch-историй с взаимодействием нескольких исполнителей и неподвижных объектов. Создание Scratch-историй с взаимодействием нескольких исполнителей (1 час).

### **Раздел V. Использование программы Scratch для создания мини-игр**

#### **Тема 5.1. Виды компьютерных игр. Алгоритмическая разработка листинга программы.**

**Теория.** Компьютерные игры – вред или польза. Виды компьютерных игр. Этапы разработки игр программистами (1 час).

**Практика.** Алгоритмическая разработка проекта, запись на естественном языке событий и точек взаимодействия героев будущей игры (1 час).

#### **Тема 5.2. Разработка базовых спрайтов для игры. Формирование базовых скриптов.**

**Теория.** Логика создания персонажей для игры. Перевод алгоритма, написанного на естественном языке, в коды Scratch (1 час).

**Практика.** Разработка и создание основных спрайтов и их костюмов для будущей игры. Разработка скриптов для спрайтов и объектов (1 час).

#### **Тема 5.3. Синхронизация работы скриптов для разных спрайтов**

**Практика.** Доработка основного листинга программы с целью установления связей между спрайтами. Тестирование и отладка программы (1 час).

#### **Тема 5.4. Переход из одной сцены в другую. Создание интерфейса игры.**

**Теория.** Односторонний (без возможности вернуться назад) переход из одного пространства в другое. Понятие интерфейса. Элементы интерфейса. Основные принципы дизайна интерфейсов. Обратная связь. Необходимые элементы меню (1 час).

**Практика.** Создать программу для перемещения объекта по игровой карте и разработать интерфейс для Scratch-проекта (1 час).

#### **Тема 5.5. Сообщество Scratch в Интернете. Просмотр и публикация проектов.**

**Практика.** Правила работы в сети. Интернет-сообщества. Сообщество Scratch. Регистрация на сайте. Использование заимствованных кодов и объектов. Просмотр проектов сообщества и публикация собственных проектов (1 час).

## РАЗДЕЛ 2. КОМПЛЕКС ОРГАНИЗАЦИОННО-ПЕДАГОГИЧЕСКИХ УСЛОВИЙ

### Методическое, дидактическое и материально-техническое обеспечение реализации программы

#### Принципы, методы, формы, технологии обучения, воспитания и развития обучающихся

При отборе обучающего материала и установлении его последовательности соблюдаются следующий **принципы**:

- структурирование учебного материала с учетом объективно существующих связей между его темами;
- актуальность, значимость учебного материала для обучающегося.

#### Основные формы и методы:

Формы работы: при организации занятия органически сочетаются все формы работы с обучающимися: коллективные, индивидуальные, групповые и т.д.

Форма занятий: рассказ; беседа; объяснение; игра; практическая работа и др.

В процессе реализации программы используются следующие **методы обучения**:

- 1) исследовательский, применяемый при самостоятельной работе обучающихся;
- 2) репродуктивный, используемый в процессе применения полученных знаний (воспроизведение действий по применению знаний на практике, деятельности по алгоритму, программирование);
- 3) наглядный, в процессе которого в т.ч. осуществляется демонстрация мультимедийных материалов;
- 4) объяснительно-иллюстративный для формирования знаний и образа действий;
- 5) стимулирующий (развитие познавательного интереса у обучающегося, эмоциональное стимулирование и т.д.); проблемное изложение изучаемого материала (учебные проблемы ставятся и решаются обучающимися с помощью педагога).

Обучение по программе предлагает такие методики и решения, которые помогают становиться детям творчески мыслящими, обучают работе в команде. Эта система предлагает детям проблемы, дает в руки инструменты, позволяющие им найти своё собственное решение. Для успешной реализации данной программы используются современные методы и формы занятий, которые помогают сформировать у обучающихся интерес к данному направлению.

**Педагогические технологии и методики.** В обучении по программе применяются особые технологии, выбор которых будет зависеть от выбранной модели обучения индивидуально с каждым обучающимся. В целях оптимизации и совершенствования образовательного процесса педагог может применять разные педагогические технологии – технологию группового обучения, технологию дифференцированного обучения, технологию сотрудничества, технологию проблемного обучения, технологию индивидуального образовательного маршрута и др.

*Личностно – ориентированные технологии* ставят в центр всей образовательной системы личность обучающегося. Обеспечение комфортных, бесконфликтных условий ее развития, реализацию ее природных потенциалов. Именно на такие технологии опирается программа с индивидуальным форматом обучения.

#### *Технология индивидуального образовательного маршрута*

Данная технология имеет целью реализовать следующие права и возможности обучающегося:

- право на выбор или выявление индивидуального смысла и целей в обучении;
- право выбора индивидуального темпа обучения, форм и методов решения образовательных задач, способов контроля, рефлексии и самооценки своей деятельности;
- превышение (опережение или углубление) осваиваемого содержания учебного плана.

Основные элементы индивидуальной образовательной деятельности обучающегося – это смысл деятельности (зачем я это делаю); постановка личной цели (предвосхищающий результат); план деятельности; реализация плана; рефлексия (осознание собственной деятельности); оценка; корректировка или переопределение целей.

Условием достижения целей и задач личностно-ориентированного обучения является сохранение индивидуальных особенностей обучающегося, его уникальности и разноплановости. Для этого применяются следующие способы; индивидуальные задания; формулировка обучающимся открытых заданий, которые предполагают их выполнение индивидуально каждым обучающимся; предложение обучающемуся составить план занятия для себя, выбрать содержание своего задания для самостоятельной работы.

В целях оптимизации и совершенствования образовательного процесса педагог может применять разные педагогические технологии – технологию группового обучения, технологию дифференцированного обучения, технологию сотрудничества, технологию проблемного обучения, технологию индивидуального образовательного маршрута и др.

### **Формы аттестации. Педагогический контроль**

Для отслеживания результативности образовательной деятельности по программе проводятся текущий контроль.

Текущий контроль – оценка уровня и качества освоения тем программы и личностных качеств обучающихся (осуществляется на занятиях в течение всего учебного периода).

Контроль по программе проводится в форме педагогического наблюдения, мониторинга выполнения практических заданий.

Педагог дополнительного образования свободен в выборе форм текущего контроля, он вправе остановиться на той форме, которая бы будет интересной и увлекательной для обучающихся.

На практических занятиях учащиеся показывают ранее полученные знания, здесь педагог имеет возможность скорректировать действия ребенка, если он недостаточно усвоил теоретический материал.

Результативность обучения по программе определяется в виде наблюдения педагога и оценивается по уровневой системе: «высокий», «средний», «низкий». Формы оценки качества знаний – наблюдения педагога за выполнением практических заданий и др.

#### Основные принципы системы оценки:

- доброжелательное отношение к обучающемуся;
- конкретный анализ трудностей, которые испытал обучающийся, а также допущенных им ошибок;
- конкретные указания на то, как можно улучшить достигнутый результат во время следующей попытки.

Подобный подход к контролю и оценке умений обучающихся ориентирован на успехи, а не на неудачи, на их поощрение, поддержку.

#### **Критерии оценки**

Уровень	Критерий
Высокий	Самостоятельная деятельность обучающегося; при выполнении той или иной деятельности обучающийся не испытывает особых затруднений;
Средний	При выполнении той или иной деятельности обучающийся испытывает минимальные затруднения, прибегает к помощи педагога, стремится исправить указанные ошибки, самостоятельно выполняет задания;
Низкий	Обучающийся испытывает серьезные затруднения при выполнении той или иной деятельности, нуждается в постоянной помощи и контроле педагога; овладел менее чем 1/3 умениями

#### **Оценочные материалы**

**Оценочные материалы** в программе представлены перечнем используемых заданий-диагностик, которые позволяют определить достижение обучающимися планируемых результатов (Приложение).

## Организационно - педагогические условия

Занятия по образовательной программе проходят в группах и индивидуально.

Виды занятий по программе определяются содержанием программы и предусматривают теоретические и практические занятия.

**Режим занятий:** на реализацию программы отводится 2 часа в неделю (два занятия по 45 мин с 10-минутным перерывом), всего 36 часов. Занятия проводятся 1 раз в неделю по 2 учебных часа.

Для всех занятий учебный час устанавливается продолжительностью 45 минут.

**Условия набора обучающихся.** Программа предусматривает свободный набор детей, желающих начать изучать программирование в среде Scratch. Уровень подготовки не требуется, так как программа рассчитана на стартовый (ознакомительный) уровень. Вступительные испытания не предусмотрены.

Группы обучающихся формируются на основе свободного набора, постоянного состава. Набор проводится по заявлению родителей (законных представителей).

**Формы организации образовательного процесса:** индивидуально-групповые; мини-группы, занятия с использованием индивидуального подхода к каждому ребёнку.

Занятия по программе состоят из теоретической и практической части. Форму занятий можно определить как интерактивное, практическое обучение (практические занятия), теоретическое обучение. Основной формой обучения является практическая работа, которая выполняется малыми группами (2-3 человека) или индивидуальная работа.

**Особенности организации образовательного процесса:**

Форма обучения по программе: индивидуально-групповая.

Занятия в малых группах проводятся при реализации учебного плана с учетом потребностей обучения.

Индивидуальная форма работы используется при общении с конкретными учащимися. Такой подход используется для более детальной отработки навыков и умений, помогает развитию индивидуальных особенностей обучающихся.

В процессе реализации программы используются следующие методы обучения:

*Словесные методы обучения:* рассказ; беседа; объяснение; игра.

*Наглядные методы обучения:* демонстрационный; иллюстративный; наблюдения и др.

Формы организации занятий может варьироваться выбирается с учетом возрастных особенностей детей, уровня освоения учащимися программы и их достижений.

*Структура занятия:*

1. Организационный этап.
2. Мотивационный этап (демонстрация или сюжет, ситуация).
3. Постановка проблемы или задачи.
4. Обсуждение–поиск путей решения (в группах различного состава, в зависимости от задачи).
5. Проектирование и программирование.
6. Подготовка демонстрации.
7. Заключительный этап: презентация работ обучающихся друг другу.

**Форма обучения:** очная.

*Учебно-методический комплекс программы состоит из трех компонентов:*

1. учебные и методические материалы для педагогов и обучающихся;
2. система средств обучения;
3. система средств контроля результативности обучения.

## Кадровое обеспечение (педагогические условия)

**Кадровое обеспечение** по программе осуществляется в соответствии с Единым квалификационным справочником должностей руководителей, специалистов и служащих, утвержденным приказом Министерства здравоохранения и социального развития Российской Федерации от 26 августа 2010 года № 761н и профессиональным стандартом "Педагог дополнительного образования детей и взрослых" (утв. приказом Министерства труда и социальной защиты Российской Федерации от 22 сентября 2021 года N 652н).

Программу реализует педагог(и) дополнительного образования.

Реализация программы обеспечивается педагогическими кадрами, имеющими среднее профессиональное или высшее образование (по направлению, соответствующему направлению программы, реализуемой организацией, осуществляющей образовательную деятельность) и отвечающими квалификационным требованиям, указанным в квалификационных справочниках, и (или) профессиональным стандартам.

#### Требования к педагогам дополнительного образования

Требования к образованию и обучению:

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

или

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

или

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

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

#### Особые условия допуска к работе:

Отсутствие ограничений на занятие педагогической деятельностью, установленных законодательством Российской Федерации;

Прохождение обязательных предварительных и периодических медицинских осмотров.

### **Материально-техническое обеспечение**

Образовательная организация располагает материально-технической базой, обеспечивающей проведение предусмотренных программой теоретических и практических занятий.

Материально-техническая база образовательной организации включает в себя: учебное помещение с мебелью (столами, стульями, оборудованием и пр.).

Имеются два учебных кабинета, оборудованных мебелью и ноутбуками.

Помещения для занятий достаточно просторны и освещены согласно нормам СанПин. Мебель соответствует нормам.

#### Материально-техническое оснащение

Наименование оборудованных учебных кабинетов, объектов для проведения практических занятий	Адрес (местоположение) учебных кабинетов (с указанием площади и номера помещения в соответствии с документами бюро технической инвентаризации)	Документ – основание возникновения права (реквизиты и срок действия)
Учебный кабинет № 88а Учебный кабинет № 90	420139, Республика Татарстан, г. Казань, ул. Дубравная, д. 2Д, пом.1020	Договор аренды №АЯ-01-03/2023/1020-1от 01.03.2023

	Помещения по документам бюро технической документации № 88а – 20,9 кв.м., №90 – 23,4 кв.м.	до 29.02.2024 с дальнейшей пролонгацией
--	--	---

### Перечень оборудования

Учебный кабинет	Оборудование
Учебный кабинет № 88а	Ноутбуки – 4 шт., Столы – 8шт., Учебно-маркерная доска – 1 шт., Проектор с экраном – 1шт., Шкаф – 2шт.
Учебный кабинет № 90	Ноутбуки – 4 шт., Столы – 8шт., Стулья – 4шт., Учебно-маркерная доска – 1 шт., Проектор с экраном – 1шт., Шкаф – 2шт.

Реализация программы обеспечена учебно-методической документацией, учебными и учебно-методическими изданиями, справочниками и т.д., формируемой в соответствии с темами учебного плана.

### Список литературы

#### Список литературы, используемой педагогом

1. Рындак В. Г., Дженжер В. О., Денисова Л. В. Проектная деятельность школьника в среде программирования Scratch: учебно-методическое пособие / В. Г. Рындак, В. О. Дженжер, Л. В. Денисова. — Оренбург: Оренб. гос. ин-т. менеджмента, 2009. — 116 с.: ил.
2. Патаракин Е. Д. Учимся готовить в среде Скретч (Учебно-методическое пособие). М: Интуит.ру, 2008. – 61 с.
3. Матяш Н. В. Психология проектной деятельности школьников в условиях технологического образования / Под ред. В. В. Рубцова. Мозырь: РИФ «Белый ветер», 2000. – 285 с.

#### Список рекомендуемой литературы для детей и родителей

1. Голиков Д. В. Scratch 3 для юных программистов. — СПб.: БХВ-Петербург, 2020. — 168 с.
2. Мажед Маржи. Scratch для детей. Самоучитель по программированию. - Издательство: Манн, Иванов и Фербер, 2018. – 288 с.
3. Торгашева Ю. Программирование для детей. Мои первые программы на Scratch. – СПб.: Питер, 2018. – 96 с.
4. Мэтью Хайлэнд. Програмируем с детьми. Создай 10 веселых игр на Scratch. - Бомбора, 2021. – 176 с.
5. Вордерман Кэрл, Макаманус Шон, Вудкок Джон. Программирование для детей. Иллюстрированное руководство по языкам Scratch и Python. - Издательство: Манн, Иванов и Фербер, 2019. – 224 с.

### Интернет-ресурсы

Библиотека знаний о Scratch (Scratch Wiki) [Электронный ресурс]. - Режим доступа свободный: <https://wiki.scratch.mit.edu>

## Календарный учебный график

Календарный учебный график является примерным и утверждается отдельно для каждой учебной группы.

№ п/п	Месяц	Число	Время проведения занятия	Форма занятия	Количество часов	Тема занятия	Место проведения	Форма контроля
1.	сентябрь	02	16.00-16.45	Беседа	1	Тема 1.1. Введение. Что такое Scratch. Основные алгоритмические конструкции. Знакомство с интерфейсом программы Scratch	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
2.	сентябрь	09	16.00-17.40	Беседа, практическая работа	2	Тема 2.1. Сцена. Редактирование фона. Добавление фона из файла	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
3.	сентябрь	16	16.00-17.40	Беседа, практическая работа	2	Тема 2.2. Понятие спрайтов. Добавление новых спрайтов. Рисование новых объектов.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
4.	сентябрь	23	16.00-17.40	Беседа, практическая работа	2	Тема 3.1. Синий ящик – команды движения. Темно-зеленый ящик – команды рисования	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
5.	сентябрь	30	16.00-17.40	Беседа, практическая работа	2	Тема 3.2. Фиолетовый ящик – внешний вид объекта. Оживление объекта с помощью добавления костюмов	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
6.	октябрь	07	16.00-17.40	Беседа, практическая работа	2	Тема 3.3. Желтый ящик – контроль. Лиловый ящик – добавление звуков	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
7.	октябрь	14	16.00-17.40	Беседа, практическая работа	2	Тема 3.4. Использование в программах условных операторов.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
8.	октябрь	21	16.00-17.40	Беседа, практическая работа	2	Тема 3.5. Функциональность работы циклов. Цикличность выполнения действий в зависимости от поставленных условий.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
9.	октябрь	28	16.00-17.40	Беседа, практическая работа	2	Тема 3.6. Зеленый ящик – операторы. Использование арифметических и логических блоков	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение

						вместе с блоками управления		
10.	ноябрь	11	16.00-17.40	Беседа, практическая работа	2	Тема 3.7. События. Оранжевый ящик – переменные.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
11.	ноябрь	18	16.00-17.40	Беседа, практическая работа	2	Тема 3.8. Списки.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
12.	ноябрь	25	16.00-17.40	Беседа, практическая работа	2	Тема 3.9. Голубой ящик – сенсоры. Ввод-вывод данных.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
13.	декабрь	02	16.00-17.40	Беседа, практическая работа	2	Тема 4.1. Последовательность и параллельность выполнения скриптов	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
14.	декабрь	09	16.00-17.40	Беседа, практическая работа	2	Тема 4.2. Взаимодействие между спрайтами. Управление через обмен сообщениями.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
15.	декабрь	16	16.00-17.40	Беседа, практическая работа	2	Тема 5.1. Виды компьютерных игр. Алгоритмическая разработка листинга программы	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
16.	декабрь	23	16.00-17.40	Беседа, практическая работа	2	Тема 5.2. Разработка базовых спрайтов для игры. Формирование базовых скриптов	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
17.	январь	13	16.00-17.40	Беседа, практическая работа	2	Тема 5.3. Синхронизация работы скриптов для разных спрайтов	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
18.	январь	20	16.00-17.40	Беседа, практическая работа	2	Тема 5.4. Переход из одной сцены в другую. Создание интерфейса игры.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
19.	январь	27	16.00-16.45	Практическая работа	1	Тема 5.5. Сообщество Scratch в Интернете. Просмотр и публикация проектов	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение

## РАБОЧИЕ ПРОГРАММЫ РАЗДЕЛОВ

### РАБОЧАЯ ПРОГРАММА РАЗДЕЛА I. ИНТЕРФЕЙС ПРОГРАММЫ SCRATCH

**Цель программы** – формирование и развитие познавательных и творческих способностей обучающихся, удовлетворение их индивидуальных потребностей в интеллектуальном совершенствовании посредством освоения основ программирования в среде Scratch.

**Задачи программы:**

**Обучающие задачи:**

- сформировать представление об основах программирования в Scratch;

**Развивающие задачи:**

- развивать познавательный интерес, внимание, память;
- развивать познавательную и творческую активность в использовании информационных технологий;

**Воспитательные задачи:**

- воспитывать самостоятельность, уверенность в своих силах;
- воспитывать ценностное отношение к знаниям, интерес к изучению нового.

#### Планируемые результаты освоения программы

**Предметные результаты:**

В результате изучения раздела обучающиеся:

- познакомятся с интерфейсом Scratch;
- научатся размещать объекты на сцене, поворачивать их и масштабировать;

**Метапредметные результаты:**

- развитие познавательного интереса, внимания, памяти;

**Личностные результаты:**

- воспитание самостоятельности, уверенности в своих силах;
- воспитание ценностного отношения к знаниям, интереса к изучению нового.

#### Учебный план

№ п/п	Название раздела, темы	Количество часов <sup>2</sup>			Формы аттестации (контроля)
		Всего	Теория	Практика	
1.	<b>Раздел I. Интерфейс программы Scratch</b>				
1.1.	Тема 1.1. Введение. Что такое Scratch. Основные алгоритмические конструкции. Знакомство с интерфейсом программы Scratch	1	1	-	Педагогическое наблюдение
<b>Итого часов</b>		<b>1</b>	<b>1</b>	<b>-</b>	

<sup>2</sup> Для всех занятий учебный час устанавливается продолжительностью 45 минут.

## Содержание

### Раздел I. Интерфейс программы Scratch

#### Тема 1.1. Введение. Что такое Scratch. Основные алгоритмические конструкции. Знакомство с интерфейсом программы Scratch.

**Теория.** История создания среды Scratch. Основные базовые алгоритмические конструкции и их исполнение в среде Scratch. Понятие исполнителя, алгоритма и программы, их назначение, виды и использование. Виды управления исполнителем. Способы записи алгоритма. Основные характеристики исполнителя. Система команд исполнителя. Понятие проект, его структура и реализация в среде Scratch. Основные компоненты проекта Scratch: спрайты и скрипты. Принцип создания анимации и движения объектов. Листинг программы. Сцена. Текущие данные о спрайте. Стилль поворота. Закладки. Панель инструментов, Новый спрайт. Координаты мышки. Режим представления. Окно скриптов. Окно блоков. Блоки стека. Блоки заголовков. Блоки ссылок.

#### Календарный учебный график

№ п/п	Месяц	Число	Время проведения занятия	Форма занятия	Количество часов	Тема занятия	Место проведения	Форма контроля
1.	сентябрь	02	16.00-16.45	Беседа	1	Тема 1.1. Введение. Что такое Scratch. Основные алгоритмические конструкции. Знакомство с интерфейсом программы Scratch	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение

#### Организационно-педагогические условия реализации

Занятия по образовательной программе проходят в группах и индивидуально.

Виды занятий по программе определяются содержанием программы и предусматривают теоретические занятия.

**Формы организации образовательного процесса:** индивидуально-групповые; мини-группы, занятия с использованием индивидуального подхода к каждому ребёнку.

**Особенности организации образовательного процесса:**

Форма обучения по программе: индивидуально-групповая.

Занятия в малых группах проводятся при реализации учебного плана с учетом потребностей обучения.

Индивидуальная форма работы используется при общении с конкретными учащимися. Такой подход используется для более детальной отработки навыков и умений, помогает развитию индивидуальных особенностей обучающихся.

В процессе реализации программы используются следующие методы обучения:

*Словесные методы обучения:* рассказ; беседа; объяснение; игра.

*Наглядные методы обучения:* демонстрационный; иллюстративный; наблюдения и др.

Формы организации занятий может варьироваться выбирается с учетом возрастных особенностей детей, уровня освоения учащимися программы и их достижений.

*Структура занятия:*

1. Организационный этап.

2. Мотивационный этап (демонстрация или сюжет, ситуация).

3. Постановка проблемы или задачи.

4. Обсуждение–поиск путей решения (в группах различного состава, в зависимости от задачи).

5. Проектирование и программирование.

6. Подготовка демонстрации.

7. Заключительный этап: презентация работ обучающихся друг другу.

*Учебно-методический комплекс программы состоит из трех компонентов:*

1. учебные и методические материалы для педагогов и обучающихся;
2. система средств обучения;
3. система средств контроля результативности обучения.

### **Кадровое обеспечение**

Программу реализует педагог(и) дополнительного образования.

Реализация программы обеспечивается педагогическими кадрами, имеющими среднее профессиональное или высшее образование (по направлению, соответствующему направлению программы, реализуемой организацией, осуществляющей образовательную деятельность) и отвечающими квалификационным требованиям, указанным в квалификационных справочниках, и (или) профессиональным стандартам.

### **Материально-технические условия реализации программы**

Образовательная организация располагает материально-технической базой, обеспечивающей проведение предусмотренных программой занятий.

Материально-техническая база образовательной организации включает в себя: учебное помещение с мебелью (столами, стульями и пр.).

Имеются два учебных кабинета, оборудованных мебелью и ноутбуками.

Помещения для занятий достаточно просторны и освещены согласно нормам СанПин. Мебель соответствует нормам.

### **Информационные и учебно-методические условия**

#### **Список литературы, используемой педагогом**

1. Рындак В. Г., Дженжер В. О., Денисова Л. В. Проектная деятельность школьника в среде программирования Scratch: учебно-методическое пособие / В. Г. Рындак, В. О. Дженжер, Л. В. Денисова. — Оренбург: Оренб. гос. ин-т. менеджмента, 2009. — 116 с.: ил.
2. Патаракин Е. Д. Учимся готовить в среде Скретч (Учебно-методическое пособие). М: Интуит.ру, 2008. — 61 с.
3. Матяш Н. В. Психология проектной деятельности школьников в условиях технологического образования / Под ред. В. В. Рубцова. Мозырь: РИФ «Белый ветер», 2000. — 285 с.

#### **Список рекомендуемой литературы для детей и родителей**

1. Голиков Д. В. Scratch 3 для юных программистов. — СПб.: БХВ-Петербург, 2020. — 168 с.
2. Мажед Маржи. Scratch для детей. Самоучитель по программированию. - Издательство: Манн, Иванов и Фербер, 2018. — 288 с.
3. Торгашева Ю. Программирование для детей. Мои первые программы на Scratch. — СПб.: Питер, 2018. — 96 с.
4. Мэтью Хайлэнд. Програмируем с детьми. Создай 10 веселых игр на Scratch. - Бомбора, 2021. — 176 с.
5. Вордерман Кэрол, Макаманус Шон, Вудкок Джон. Программирование для детей. Иллюстрированное руководство по языкам Scratch и Python. - Издательство: Манн, Иванов и Фербер, 2019. — 224 с.

### **Интернет-ресурсы**

Библиотека знаний о Scratch (Scratch Wiki) [Электронный ресурс]. - Режим доступа свободный: <https://wiki.scratch.mit.edu>

## РАБОЧАЯ ПРОГРАММА РАЗДЕЛА II. НАЧАЛО РАБОТЫ В СРЕДЕ SCRATCH

**Цель программы** – формирование и развитие познавательных и творческих способностей обучающихся, удовлетворение их индивидуальных потребностей в интеллектуальном совершенствовании посредством освоения основ программирование в среде Scratch.

### **Задачи программы:**

#### **Обучающие задачи:**

- сформировать представление об основах программирования в Scratch;
- освоить основные инструменты и операции работы в Scratch;
- изучить основные принципы создания простых проектов в среде Scratch (анимация и игры);
- научить рисовать в векторном и растровом графических редакторах в среде Scratch;

#### **Развивающие задачи:**

- развивать познавательный интерес, внимание, память;
- развивать логическое, абстрактное и образное мышление;
- развивать воображение и фантазию через создание простых проектов в среде Scratch (анимация и игры);
- развить способности к самостоятельной работе и анализу проделанной работы;
- развивать познавательную и творческую активность в использовании информационных технологий;

#### **Воспитательные задачи:**

- воспитывать самостоятельность, уверенность в своих силах;
- формировать творческий подход к поставленной задаче;
- воспитывать ценностное отношение к знаниям, интерес к изучению нового;
- воспитывать стремление добиваться поставленной цели;
- воспитывать чувство ответственности за свою работу.

### **Планируемые результаты освоения программы**

#### **Предметные результаты:**

В результате изучения раздела обучающиеся:

- познакомятся с интерфейсом Scratch;
- освоят основные инструменты и операции работы в Scratch;
- научатся вставлять стандартный фон из библиотечного модуля среды. Рисовать фон в графическом редакторе. Добавлять фон из файла. Создавать спрайты с помощью графического редактора среды Scratch. Загружать на сцену спрайты из стандартной коллекции Scratch. Вставлять спрайты из файлов. Центрировать костюм. Масштабировать спрайт. Удалять спрайты;

#### **Метапредметные результаты:**

- развитие познавательного интереса, внимания, памяти;
- развитие логического, абстрактного и образного мышления;
- развитие воображения и фантазии через создание простых проекты в среде Scratch (анимация и игры);
- формирование навыка сознательного и рационального использования программирования в своей деятельности;
- внесение корректив в текущую деятельность на основе анализа изменений для получения запланированных характеристик продукта/результата;

- умение выбирать варианты и самостоятельно искать средства/ресурсы для решения задачи и достижения цели;

#### Личностные результаты:

- воспитание самостоятельности, уверенности в своих силах;
- использование творческого подхода к поставленной задаче;
- воспитание ценностного отношения к знаниям, интереса к изучению нового;
- сформированное чувство ответственности за свою работу;
- готовность и способность к созданию творческих работ созидательной направленности (игр, анимационных роликов и т.п.).

### Учебный план

№ п/п	Название раздела, темы	Количество часов <sup>3</sup>			Формы аттестации (контроля)
		Всего	Теория	Практика	
<b>2.</b>	<b>Раздел II. Начало работы в среде Scratch</b>				
2.1.	Тема 2.1. Сцена. Редактирование фона. Добавление фона из файла.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
2.2.	Тема 2.2. Понятие спрайтов. Добавление новых спрайтов. Рисование новых объектов.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
<b>Итого часов</b>		<b>4</b>	<b>2</b>	<b>2</b>	

### Содержание

#### Раздел II. Начало работы в среде Scratch

##### Тема 2.1. Сцена. Редактирование фона. Добавление фона из файла.

**Теория.** Сцена. Ширина и высота сцены. Текущие координаты объекта. Редактирование текущего фона. Вставка нового фона из файла. Вставка стандартного фона из библиотечного модуля среды. Рисование фона в графическом редакторе. Создание нескольких фонов в одной сцене (1 час).

**Практика.** Создание фона сцены на выбранную учащимся тему (1 час).

##### Тема 2.2. Понятие спрайтов. Добавление новых спрайтов. Рисование новых объектов.

**Теория.** Стандартный объект. Спрайты. Список спрайтов. Редактор рисования для создания новых спрайтов. Инструменты рисования (кисточка, линия, текст, эллипс, ) и редактирования объекта (ластик, заливка, поворот, выбор, печать, пипетка). Центрирование костюма. Масштабирование спрайта. Загрузка на сцену спрайтов из стандартной коллекции среды Scratch. Вставка спрайтов из файлов форматов JPG, BMP, PNG, GIF. Выбор случайного спрайта. Удаление спрайтов (1 час).

**Практика.** Создание фона сцены и прорисовка основных спрайтов для Scratch-истории. (1 час).

#### Календарный учебный график

№ п/п	Месяц	Число	Время проведения занятия	Форма занятия	Количество часов	Тема занятия	Место проведения	Форма контроля
1.	сентябрь	09	16.00-17.40	Беседа, практическая работа	2	Тема 2.1. Сцена. Редактирование фона. Добавление фона из файла	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение

<sup>3</sup> Для всех занятий учебный час устанавливается продолжительностью 45 минут.

2.	сентябрь	16	16.00-17.40	Беседа, практическая работа	2	Тема 2.2. Понятие спрайтов. Добавление новых спрайтов. Рисование новых объектов.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
----	----------	----	-------------	-----------------------------	---	--	---	---------------------------

### Организационно-педагогические условия реализации

Занятия по образовательной программе проходят в группах и индивидуально.

Виды занятий по программе определяются содержанием программы и предусматривают теоретические и практические занятия.

**Формы организации образовательного процесса:** индивидуально-групповые; мини-группы, занятия с использованием индивидуального подхода к каждому ребёнку.

Занятия по программе состоят из теоретической и практической части. Форму занятий можно определить как интерактивное, практическое обучение (практические занятия), теоретическое обучение. Основной формой обучения является практическая работа, которая выполняется малыми группами (2-3 человека) или индивидуальная работа.

**Особенности организации образовательного процесса:**

Форма обучения по программе: индивидуально-групповая.

Занятия в малых группах проводятся при реализации учебного плана с учетом потребностей обучения.

Индивидуальная форма работы используется при общении с конкретными учащимися. Такой подход используется для более детальной отработки навыков и умений, помогает развитию индивидуальных особенностей обучающихся.

В процессе реализации программы используются следующие методы обучения:

*Словесные методы обучения:* рассказ; беседа; объяснение; игра.

*Наглядные методы обучения:* демонстрационный; иллюстративный; наблюдения и др.

Формы организации занятий может варьироваться выбирается с учетом возрастных особенностей детей, уровня освоения учащимися программы и их достижений.

*Структура занятия:*

1. Организационный этап.
2. Мотивационный этап (демонстрация или сюжет, ситуация).
3. Постановка проблемы или задачи.
4. Обсуждение–поиск путей решения (в группах различного состава, в зависимости от задачи).
5. Проектирование и программирование.
6. Подготовка демонстрации.
7. Заключительный этап: презентация работ обучающихся друг другу.

*Учебно-методический комплекс программы состоит из трех компонентов:*

1. учебные и методические материалы для педагогов и обучающихся;
2. система средств обучения;
3. система средств контроля результативности обучения.

### Кадровое обеспечение

Программу реализует педагог(и) дополнительного образования.

Реализация программы обеспечивается педагогическими кадрами, имеющими среднее профессиональное или высшее образование (по направлению, соответствующему направлению программы, реализуемой организацией, осуществляющей образовательную деятельность) и отвечающими квалификационным требованиям, указанным в квалификационных справочниках, и (или) профессиональным стандартам.

### **Материально-технические условия реализации программы**

Образовательная организация располагает материально-технической базой, обеспечивающей проведение предусмотренных программой теоретических и практических занятий.

Материально-техническая база образовательной организации включает в себя: учебное помещение с мебелью (столами, стульями и пр.).

Имеются два учебных кабинета, оборудованных мебелью и ноутбуками.

Помещения для занятий достаточно просторны и освещены согласно нормам СанПин. Мебель соответствует нормам.

### **Информационные и учебно-методические условия реализации**

#### **Список литературы, используемой педагогом**

1. Рындак В. Г., Дженжер В. О., Денисова Л. В. Проектная деятельность школьника в среде программирования Scratch: учебно-методическое пособие / В. Г. Рындак, В. О. Дженжер, Л. В. Денисова. — Оренбург: Оренб. гос. ин-т. менеджмента, 2009. — 116 с.: ил.
2. Патаракин Е. Д. Учимся готовить в среде Скретч (Учебно-методическое пособие). М: Интуит.ру, 2008. — 61 с.
3. Матяш Н. В. Психология проектной деятельности школьников в условиях технологического образования / Под ред. В. В. Рубцова. Мозырь: РИФ «Белый ветер», 2000. — 285 с.

#### **Список рекомендуемой литературы для детей и родителей**

1. Голиков Д. В. Scratch 3 для юных программистов. — СПб.: БХВ-Петербург, 2020. — 168 с.
2. Мажед Маржи. Scratch для детей. Самоучитель по программированию. - Издательство: Манн, Иванов и Фербер, 2018. — 288 с.
3. Торгашева Ю. Программирование для детей. Мои первые программы на Scratch. – СПб.: Питер, 2018. – 96 с.
4. Мэтью Хайлэнд. Програмируем с детьми. Создай 10 веселых игр на Scratch. - Бомбора, 2021. – 176 с.
5. Вордерман Кэрол, Макаманус Шон, Вудкок Джон. Программирование для детей. Иллюстрированное руководство по языкам Scratch и Python. - Издательство: Манн, Иванов и Фербер, 2019. – 224 с.

#### **Интернет-ресурсы**

Библиотека знаний о Scratch (Scratch Wiki) [Электронный ресурс]. - Режим доступа свободный: <https://wiki.scratch.mit.edu>

## РАБОЧАЯ ПРОГРАММА РАЗДЕЛА III. ОСНОВНЫЕ СКРИПТЫ ПРОГРАММЫ SCRATCH

**Цель программы** – формирование и развитие познавательных и творческих способностей обучающихся, удовлетворение их индивидуальных потребностей в интеллектуальном совершенствовании посредством освоения основ программирование в среде Scratch.

### **Задачи программы:**

#### **Обучающие задачи:**

- сформировать представление об основах программирования в Scratch;
- освоить основные инструменты и операции работы в Scratch;
- изучить основные принципы создания простых проектов в среде Scratch (анимация и игры);
- научить рисовать в векторном и растровом графических редакторах, изменять звук, вводить, выводить и обрабатывать информацию в среде Scratch;

#### **Развивающие задачи:**

- развивать познавательный интерес, внимание, память;
- развивать логическое, абстрактное и образное мышление;
- развивать воображение и фантазию через создание простых проектов в среде Scratch (анимация и игры);
- развить способности к самостоятельной работе и анализу проделанной работы;
- развивать познавательную и творческую активность в использовании информационных технологий;

#### **Воспитательные задачи:**

- воспитывать самостоятельность, уверенность в своих силах;
- формировать творческий подход к поставленной задаче;
- воспитывать ценностное отношение к знаниям, интерес к изучению нового;
- воспитывать стремление добиваться поставленной цели;
- воспитывать чувство ответственности за свою работу.

## **Планируемые результаты освоения программы**

### **Предметные результаты:**

В результате изучения раздела обучающиеся:

- освоят основные инструменты и операции работы в Scratch, команды движения;
- научатся создавать простые проекты в среде Scratch (анимация и игры);
- научатся создавать программы для движения спрайтов по сцене, для рисования различных фигур, имитации естественного движения героев в различных направлениях. Озвучивать как полностью проект, так и отдельные события внутри проекта. Создавать программы - с изменением последовательного выполнения скриптов при наличии условий, с использованием циклов с фиксированным числом повторений, с предусловием и постусловием. Использовать в программах операции сравнения данных, арифметические и логические действия над данными, сравнение данных из нескольких списков, глобальные и локальные переменные. Обрабатывать данные с выводом на экран конечного результата;
- научатся использовать векторный и растровый графический редактор, изменять звук, вводить, выводить и обрабатывать информацию в среде Scratch;

### **Метапредметные результаты:**

- развитие познавательного интереса, внимания, памяти;
- развитие логического, абстрактного и образного мышления;
- развитие воображения и фантазии через создание простых проекты в среде Scratch (анимация и игры);

- формирование навыка сознательного и рационального использования программирования в своей деятельности;
- внесение корректив в текущую деятельность на основе анализа изменений для получения запланированных характеристик продукта/результата;
- умение выбирать варианты и самостоятельно искать средства/ресурсы для решения задачи и достижения цели;

#### Личностные результаты:

- воспитание самостоятельности, уверенности в своих силах;
- использование творческого подхода к поставленной задаче;
- воспитание ценностного отношения к знаниям, интереса к изучению нового;
- сформированное чувство ответственности за свою работу;
- готовность и способность к созданию творческих работ созидательной направленности (игр, анимационных роликов и т.п.).

#### Учебный план

№ п/п	Название раздела, темы	Количество часов <sup>4</sup>			Формы аттестации (контроля)
		Всего	Теория	Практика	
<b>3.</b>	<b>Раздел III. Основные скрипты программы Scratch</b>				
3.1.	Тема 3.1. Синий ящик – команды движения. Темно-зеленый ящик – команды рисования.	2	1	1	Педагогическое наблюдение
3.2.	Тема 3.2. Фиолетовый ящик – внешний вид объекта. Оживление объекта с помощью добавления костюмов.	2	1	1	Педагогическое наблюдение
3.3.	Тема 3.3. Желтый ящик – контроль. Лиловый ящик – добавление звуков.	2	1	1	Педагогическое наблюдение
3.4.	Тема 3.4. Использование в программах условных операторов.	2	1	1	Педагогическое наблюдение
3.5.	Тема 3.5. Функциональность работы циклов. Цикличность выполнения действий в зависимости от поставленных условий.	2	1	1	Педагогическое наблюдение
3.6.	Тема 3.6. Зеленый ящик – операторы. Использование арифметических и логических блоков вместе с блоками управления.	2	1	1	Педагогическое наблюдение
3.7.	Тема 3.7. События. Оранжевый ящик – переменные.	2	1	1	Педагогическое наблюдение
3.8.	Тема 3.8. Списки.	2	1	1	Педагогическое наблюдение
3.9.	Тема 3.9. Голубой ящик – сенсоры. Ввод-вывод данных.	2	1	1	Педагогическое наблюдение
<b>Итого часов</b>		<b>18</b>	<b>9</b>	<b>9</b>	

<sup>4</sup> Для всех занятий учебный час устанавливается продолжительностью 45 минут.

## Содержание

### Раздел III. Основные скрипты программы Scratch

#### **Тема 3.1. Синий ящик – команды движения. Темно-зеленый ящик – команды рисования.**

**Теория.** Команды – *идти*; *повернуться направо (налево)*; *повернуть в направлении*; *повернуться к*; *изменить x (y) на*; *установить x (y) в*; *если край, оттолкнуться*. Принципиальное различие действия команд *идти в* и *плыть в*. Назначение сенсоров *положение x*, *положение y* и *направлении*. Команды – *очистить*, *опустить перо*, *поднять перо*, *установить цвет пера*, *изменить цвет пера на*, *установить цвет пера*, *изменить тень пера*, *установить тень пера*, *изменить размер пера на*, *установить размер пера*, *печатать* (1 час).

**Практика.** Создание программ для передвижения спрайтов по сцене. Создание программ для рисования различных фигур (1 час).

#### **Тема 3.2. Фиолетовый ящик – внешний вид объекта. Оживление объекта с помощью добавления костюмов.**

**Теория.** Костюмы спрайта. Копирование и редактирование костюма спрайта с помощью редактора рисования. Переупорядочивание костюмов. Команды – *перейти к костюму*, *следующий костюм*, *говорить...в течении...секунд*, *сказать*, *думать*, *думать...секунд*, *изменить ...эффект на*, *установить эффект...в значение*, *убрать графические эффекты*, *изменить размер на*, *установить размер*, *показаться*, *спрятаться*, *перейти в верхний слой*, *перейти назад на...1 слоев*. Назначение сенсоров *костюм* и *размер*. Понятие раскадровки движения. Изменение костюма спрайта для имитации движения (1 час).

**Практика.** Создание программы для управления внешним видом объекта. Создание Scratch-историй с имитацией хождения и движения объектов (1 час).

#### **Тема 3.3. Желтый ящик – контроль. Лиловый ящик – добавление звуков.**

**Теория.** Кнопка с зеленым флажком и ее назначение. Управление последовательностью выполнения скриптов. Понятие управляющих сообщений. Команды – *передать*, *передать и ждать, когда я получу*. Скрипты для создания условных конструкций программы – *если, если...или*. Скрипты для управления циклами – *всегда*, *повторить, всегда, если, повторять до..* Команды – *когда клавиша...нажата, когда щелкнут по, ждать...секунд, ждать до, остановить скрипт, остановить все*. Загрузка звуков из стандартной коллекции и из файлов жесткого диска. Запись звука через микрофон. Принципиальная разница работы команд *играть звук* и *играть звук до завершения*. Команды – *остановить все звуки, барабану играть...тактов, оставшиеся...тактов, ноту...играть...тактов, выбрать инструмент, изменить громкость, установить громкость, изменить темп на, установить темп*. Назначение сенсоров *громкость* и *темп* (1 час).

**Практика.** Создание программ с элементами управления объектом. Озвучивание Scratch-историй (1 час).

#### **Тема 3.4. Использование в программах условных операторов.**

**Теория.** Базовая конструкция ветвление, назначение, виды (полная и неполная форма). Понятие условия. Изменение порядка выполнения скриптов в зависимости от условия. Разветвление листинга программы. Скрипты условных операторов. Использование неполной формы ветвления в системе Scratch (1 час).

**Практика.** Создание программ с изменением последовательного выполнения скриптов при наличии условий (1 час).

#### **Тема 3.5. Функциональность работы циклов. Цикличность выполнения действий в зависимости от поставленных условий.**

**Теория.** Циклы с фиксированным числом повторений. Заголовок цикла. Тело цикла. Циклы с условным оператором. Заголовок цикла. Тело цикла. Предусловие и постусловие. Зацикливание (1 час).

**Практика.** Создание программ с использованием циклов с фиксированным числом повторений. Создание программ с использованием циклов с предусловием и постусловием (1 час).

### Тема 3.6. Зеленый ящик – операторы. Использование арифметических и логических блоков вместе с блоками управления.

**Теория.** Числа. Строинги. Логические величины. Логические выражения. Арифметические операции. Логические операции. Операции сравнения. Команды для работы со строингами – *слить, буква...в, длина строки*. Команда *выдать случайное от...до*. Использование арифметических и логических блоков в листинге программы. Просмотр полученного результата (1 час).

**Практика.** Создание программ с использованием операций сравнения данных. Создание программ с использованием арифметических данных и логических операций (1 час).

### Тема 3.7. События. Оранжевый ящик – переменные.

**Теория.** События в проектах Scratch. Понятие переменных и необходимость их использования в листинге программы. Глобальные и локальные переменные. Имя переменной и правила его формирования. Команды для переменных - *поставить...в, изменить...на, показать переменную, спрятать переменную*. Удаление переменных. Создание счетчиков с помощью переменных (1 час).

**Практика.** Разработка сценария Scratch-историй с несколькими событиями. Создание проектов с использованием глобальных и локальных переменных (1 час).

### Тема 3.8. Списки.

**Теория.** Создание списков и необходимость их использования в проектах Scratch. Добавление в список данных. Удаление данных из списка. Удаление списка. Команды работы со списками – *добавить...к, удалить...из, поставить...в...из, заменить элемент...в...на, элемент...из, длина списка* (1 час).

**Практика.** Создание программ-тестов по принципу сравнения данных из нескольких списков (1 час).

### Тема 3.9. Голубой ящик – сенсоры. Ввод-вывод данных.

**Теория.** Понятие сенсора. Правила применения и область действия команд *касается, касается цвета и цвет.касается*. Функционал команды *спросить...и ждать*. Сенсоры *мышка по x, мышка по y, мышка нажата?, клавиша...нажата?, расстояние до, перезапустить таймер*. Сенсоры, значение которых можно выводить на экран – *ответ, таймер, громкость, громко?, ...значение сенсора и сенсор....* Необходимость ввода данных для их обработки в программе. Ввод данных с помощью команды *спросить*. Вывод конечного результата обработки с помощью команд *говорить* и *сказать* (1 час).

**Практика.** Создание проектов с использованием значений сенсоров и команды *спросить*. Создание программ для обработки данных пользователя с выводом на экран конечного результата (1 час).

### Календарный учебный график

№ п/п	Месяц	Число	Время проведения занятия	Форма занятия	Количество часов	Тема занятия	Место проведения	Форма контроля
1.	сентябрь	23	16.00-17.40	Беседа, практическая работа	2	Тема 3.1. Синий ящик – команды движения. Темно-зеленый ящик – команды рисования	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
2.	сентябрь	30	16.00-17.40	Беседа, практическая работа	2	Тема 3.2. Фиолетовый ящик – внешний вид объекта. Оживление объекта с помощью	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение

						добавления костюмов		
3.	октябрь	07	16.00-17.40	Беседа, практическая работа	2	Тема 3.3. Желтый ящик – контроль. Лиловый ящик – добавление звуков	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
4.	октябрь	14	16.00-17.40	Беседа, практическая работа	2	Тема 3.4. Использование в программах условных операторов.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
5.	октябрь	21	16.00-17.40	Беседа, практическая работа	2	Тема 3.5. Функциональность работы циклов. Цикличность выполнения действий в зависимости от поставленных условий.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
6.	октябрь	28	16.00-17.40	Беседа, практическая работа	2	Тема 3.6. Зеленый ящик – операторы. Использование арифметических и логических блоков вместе с блоками управления	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
7.	ноябрь	11	16.00-17.40	Беседа, практическая работа	2	Тема 3.7. События. Оранжевый ящик – переменные.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
8.	ноябрь	18	16.00-17.40	Беседа, практическая работа	2	Тема 3.8. Списки.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
9.	ноябрь	25	16.00-17.40	Беседа, практическая работа	2	Тема 3.9. Голубой ящик – сенсоры. Ввод-вывод данных.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение

### Организационно-педагогические условия реализации

Занятия по образовательной программе проходят в группах и индивидуально.

Виды занятий по программе определяются содержанием программы и предусматривают теоретические и практические занятия.

**Формы организации образовательного процесса:** индивидуально-групповые; мини-группы, занятия с использованием индивидуального подхода к каждому ребёнку.

Занятия по программе состоят из теоретической и практической части. Форму занятий можно определить как интерактивное, практическое обучение (практические занятия), теоретическое обучение. Основной формой обучения является практическая работа, которая выполняется малыми группами (2-3 человека) или индивидуальная работа.

**Особенности организации образовательного процесса:**

Форма обучения по программе: индивидуально-групповая.

Занятия в малых группах проводятся при реализации учебного плана с учетом потребностей обучения.

Индивидуальная форма работы используется при общении с конкретными учащимися. Такой подход используется для более детальной отработки навыков и умений, помогает развитию индивидуальных особенностей обучающихся.

В процессе реализации программы используются следующие методы обучения:

*Словесные методы обучения:* рассказ; беседа; объяснение; игра.

*Наглядные методы обучения:* демонстрационный; иллюстративный; наблюдения и др.

Формы организации занятий может варьироваться выбирается с учетом возрастных особенностей детей, уровня освоения учащимися программы и их достижений.

*Структура занятия:*

1. Организационный этап.
2. Мотивационный этап (демонстрация или сюжет, ситуация).
3. Постановка проблемы или задачи.
4. Обсуждение—поиск путей решения (в группах различного состава, в зависимости от задачи).
5. Проектирование и программирование.
6. Подготовка демонстрации.
7. Заключительный этап: презентация работ обучающихся друг другу.

*Учебно-методический комплекс программы состоит из трех компонентов:*

1. учебные и методические материалы для педагогов и обучающихся;
2. система средств обучения;
3. система средств контроля результативности обучения.

### **Кадровое обеспечение**

Программу реализует педагог(и) дополнительного образования.

Реализация программы обеспечивается педагогическими кадрами, имеющими среднее профессиональное или высшее образование (по направлению, соответствующему направлению программы, реализуемой организацией, осуществляющей образовательную деятельность) и отвечающими квалификационным требованиям, указанным в квалификационных справочниках, и (или) профессиональным стандартам.

### **Материально-технические условия реализации программы**

Образовательная организация располагает материально-технической базой, обеспечивающей проведение предусмотренных программой теоретических и практических занятий.

Материально-техническая база образовательной организации включает в себя: учебное помещение с мебелью (столами, стульями и пр.).

Имеются два учебных кабинета, оборудованных мебелью и ноутбуками.

Помещения для занятий достаточно просторны и освещены согласно нормам СанПин. Мебель соответствует нормам.

### **Информационные и учебно-методические условия реализации**

#### **Список литературы, используемой педагогом**

1. Рындак В. Г., Дженжер В. О., Денисова Л. В. Проектная деятельность школьника в среде программирования Scratch: учебно-методическое пособие / В. Г. Рындак, В. О. Дженжер, Л. В. Денисова. — Оренбург: Оренб. гос. ин-т. менеджмента, 2009. — 116 с.: ил.
2. Патаракин Е. Д. Учимся готовить в среде Скретч (Учебно-методическое пособие). М: Интуит.ру, 2008. — 61 с.
3. Матяш Н. В. Психология проектной деятельности школьников в условиях технологического образования / Под ред. В. В. Рубцова. Мозырь: РИФ «Белый ветер», 2000. — 285 с.

### Список рекомендуемой литературы для детей и родителей

1. Голиков Д. В. Scratch 3 для юных программистов. — СПб.: БХВ-Петербург, 2020. — 168 с.
2. Мажед Маржи. Scratch для детей. Самоучитель по программированию. - Издательство: Манн, Иванов и Фербер, 2018. – 288 с.
3. Торгашева Ю. Программирование для детей. Мои первые программы на Scratch. – СПб.: Питер, 2018. – 96 с.
4. Мэтью Хайлэнд. Програмируем с детьми. Создай 10 веселых игр на Scratch. - Бомбора, 2021. – 176 с.
5. Вордерман Кэрол, Макаманус Шон, Вудкок Джон. Программирование для детей. Иллюстрированное руководство по языкам Scratch и Python. - Издательство: Манн, Иванов и Фербер, 2019. – 224 с.

### Интернет-ресурсы

Библиотека знаний о Scratch (Scratch Wiki) [Электронный ресурс]. - Режим доступа свободный: <https://wiki.scratch.mit.edu>

## РАБОЧАЯ ПРОГРАММА РАЗДЕЛА IV. РАБОТА С НЕСКОЛЬКИМИ ОБЪЕКТАМИ. СИНХРОНИЗАЦИЯ ИХ РАБОТЫ

**Цель программы** – формирование и развитие познавательных и творческих способностей обучающихся, удовлетворение их индивидуальных потребностей в интеллектуальном совершенствовании посредством освоения основ программирование в среде Scratch.

### **Задачи программы:**

#### ***Обучающие задачи:***

- сформировать представление об основах программирования в Scratch;
- освоить основные инструменты и операции работы в Scratch;
- изучить основные принципы создания простых проектов в среде Scratch (анимация и игры);
- научить рисовать в векторном и растровом графических редакторах, изменять звук, вводить, выводить и обрабатывать информацию в среде Scratch;

#### ***Развивающие задачи:***

- развивать познавательный интерес, внимание, память;
- развивать логическое, абстрактное и образное мышление;
- развивать воображение и фантазию через создание простых проектов в среде Scratch (анимация и игры);
- развить способности к самостоятельной работе и анализу проделанной работы;
- развивать познавательную и творческую активность в использовании информационных технологий;

#### ***Воспитательные задачи:***

- воспитывать самостоятельность, уверенность в своих силах;
- формировать творческий подход к поставленной задаче;
- воспитывать ценностное отношение к знаниям, интерес к изучению нового;
- воспитывать стремление добиваться поставленной цели;
- воспитывать чувство ответственности за свою работу.

## Планируемые результаты освоения программы

### Предметные результаты:

В результате изучения раздела обучающиеся:

- освоят основные инструменты и операции работы в Scratch, команды движения;
- научатся создавать простые проекты в среде Scratch (анимация и игры);
- научатся создавать Scratch-истории с взаимодействием нескольких исполнителей и неподвижных объектов, а так же с одновременной и попеременной работой нескольких исполнителей;
- научатся использовать векторный и растровый графический редактор, изменять звук, вводить, выводить и обрабатывать информацию в среде Scratch;

### Метапредметные результаты:

- развитие познавательного интереса, внимания, памяти;
- развитие логического, абстрактного и образного мышления;
- развитие воображения и фантазии через создание простых проекты в среде Scratch (анимация и игры);
- формирование навыка сознательного и рационального использования программирования в своей деятельности;
- внесение корректив в текущую деятельность на основе анализа изменений для получения запланированных характеристик продукта/результата;
- умение выбирать варианты и самостоятельно искать средства/ресурсы для решения задачи и достижения цели;

### Личностные результаты:

- воспитание самостоятельности, уверенности в своих силах;
- использование творческого подхода к поставленной задаче;
- воспитание ценностного отношения к знаниям, интереса к изучению нового;
- сформированное чувство ответственности за свою работу;
- готовность и способность к созданию творческих работ созидательной направленности (игр, анимационных роликов и т.п.).

## Учебный план

№ п/п	Название раздела, темы	Количество часов <sup>5</sup>			Формы аттестации (контроля)
		Всего	Теория	Практика	
<b>4.</b>	<b>Раздел IV. Работа с несколькими объектами. Синхронизация их работы</b>				
4.1.	Тема 4.1. Последовательность и параллельность выполнения скриптов	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
4.2.	Тема 4.2. Взаимодействие между спрайтами. Управление через обмен сообщениями.	<b>2</b>	<b>1</b>	<b>1</b>	Педагогическое наблюдение
<b>Итого часов</b>		<b>4</b>	<b>2</b>	<b>2</b>	

<sup>5</sup> Для всех занятий учебный час устанавливается продолжительностью 45 минут.

## Содержание

### Раздел IV. Работа с несколькими объектами. Синхронизация их работы

#### Тема 4.1. Последовательность и параллельность выполнения скриптов

**Теория.** Последовательные и параллельные потоки в программах Scratch. Одновременная и попеременная работа нескольких исполнителей (1 час).

**Практика.** Создание Scratch-историй с одновременной и попеременной работой нескольких исполнителей (1 час).

#### Тема 4.2. Взаимодействие между спрайтами. Управление через обмен сообщениями.

**Теория.** Решение проблемы появления новых исполнителей только после того, как старые исполнители выполнили свои действия. Взаимодействие спрайтов с неподвижными объектами с помощью команд *касается* и *касается цвета*. Взаимодействие спрайтов с помощью команд *передать* и *когда я получу*. Использование сообщений для создания событий (1 час).

**Практика.** Создание Scratch-историй с взаимодействием нескольких исполнителей и неподвижных объектов. Создание Scratch-историй с взаимодействием нескольких исполнителей (1 час).

### Календарный учебный график

№ п/п	Месяц	Число	Время проведения занятия	Форма занятия	Количество часов	Тема занятия	Место проведения	Форма контроля
1.	декабрь	02	16.00-17.40	Беседа, практическая работа	2	Тема 4.1. Последовательность и параллельность выполнения скриптов	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
2.	декабрь	09	16.00-17.40	Беседа, практическая работа	2	Тема 4.2. Взаимодействие между спрайтами. Управление через обмен сообщениями.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение

### Организационно-педагогические условия реализации

Занятия по образовательной программе проходят в группах и индивидуально.

Виды занятий по программе определяются содержанием программы и предусматривают теоретические и практические занятия.

**Формы организации образовательного процесса:** индивидуально-групповые; мини-группы, занятия с использованием индивидуального подхода к каждому ребёнку.

Занятия по программе состоят из теоретической и практической части. Форму занятий можно определить как интерактивное, практическое обучение (практические занятия), теоретическое обучение. Основной формой обучения является практическая работа, которая выполняется малыми группами (2-3 человека) или индивидуальная работа.

#### **Особенности организации образовательного процесса:**

Форма обучения по программе: индивидуально-групповая.

Занятия в малых группах проводятся при реализации учебного плана с учетом потребностей обучения.

Индивидуальная форма работы используется при общении с конкретными учащимися. Такой подход используется для более детальной отработки навыков и умений, помогает развитию индивидуальных особенностей обучающихся.

В процессе реализации программы используются следующие методы обучения:

*Словесные методы обучения:* рассказ; беседа; объяснение; игра.

*Наглядные методы обучения:* демонстрационный; иллюстративный; наблюдения и др.

Формы организации занятий может варьироваться выбирается с учетом возрастных особенностей детей, уровня освоения учащимися программы и их достижений.

*Структура занятия:*

1. Организационный этап.
2. Мотивационный этап (демонстрация или сюжет, ситуация).
3. Постановка проблемы или задачи.
4. Обсуждение–поиск путей решения (в группах различного состава, в зависимости от задачи).
5. Проектирование и программирование.
6. Подготовка демонстрации.
7. Заключительный этап: презентация работ обучающихся друг другу.

*Учебно-методический комплекс программы состоит из трех компонентов:*

1. учебные и методические материалы для педагогов и обучающихся;
2. система средств обучения;
3. система средств контроля результативности обучения.

### **Кадровое обеспечение**

Программу реализует педагог(и) дополнительного образования.

Реализация программы обеспечивается педагогическими кадрами, имеющими среднее профессиональное или высшее образование (по направлению, соответствующему направлению программы, реализуемой организацией, осуществляющей образовательную деятельность) и отвечающими квалификационным требованиям, указанным в квалификационных справочниках, и (или) профессиональным стандартам.

### **Материально-технические условия реализации программы**

Образовательная организация располагает материально-технической базой, обеспечивающей проведение предусмотренных программой теоретических и практических занятий.

Материально-техническая база образовательной организации включает в себя: учебное помещение с мебелью (столами, стульями и пр.).

Имеются два учебных кабинета, оборудованных мебелью и ноутбуками.

Помещения для занятий достаточно просторны и освещены согласно нормам СанПин. Мебель соответствует нормам.

### **Информационные и учебно-методические условия реализации**

#### **Список литературы, используемой педагогом**

1. Рындак В. Г., Дженжер В. О., Денисова Л. В. Проектная деятельность школьника в среде программирования Scratch: учебно-методическое пособие / В. Г. Рындак, В. О. Дженжер, Л. В. Денисова. — Оренбург: Оренб. гос. ин-т. менеджмента, 2009. — 116 с.: ил.
2. Патаракин Е. Д. Учимся готовить в среде Скретч (Учебно-методическое пособие). М: Интуит.ру, 2008. – 61 с.
3. Матяш Н. В. Психология проектной деятельности школьников в условиях технологического образования / Под ред. В. В. Рубцова. Мозырь: РИФ «Белый ветер», 2000. – 285 с.

#### **Список рекомендуемой литературы для детей и родителей**

1. Голиков Д. В. Scratch 3 для юных программистов. — СПб.: БХВ-Петербург, 2020. — 168 с.
2. Мажед Маржи. Scratch для детей. Самоучитель по программированию. - Издательство: Манн, Иванов и Фербер, 2018. – 288 с.
3. Торгашева Ю. Программирование для детей. Мои первые программы на Scratch. – СПб.: Питер, 2018. – 96 с.

4. Мэтью Хайлэнд. Программируем с детьми. Создай 10 веселых игр на Scratch. - Бомбора, 2021. – 176 с.
5. Вордерман Кэрл, Макаманус Шон, Вудкок Джон. Программирование для детей. Иллюстрированное руководство по языкам Scratch и Python. - Издательство: Манн, Иванов и Фербер, 2019. – 224 с.

### **Интернет-ресурсы**

Библиотека знаний о Scratch (Scratch Wiki) [Электронный ресурс]. - Режим доступа свободный: <https://wiki.scratch.mit.edu>

## **РАБОЧАЯ ПРОГРАММА РАЗДЕЛА V. ИСПОЛЬЗОВАНИЕ ПРОГРАММЫ SCRATCH ДЛЯ СОЗДАНИЯ МИНИ-ИГР**

**Цель программы** – формирование и развитие познавательных и творческих способностей обучающихся, удовлетворение их индивидуальных потребностей в интеллектуальном совершенствовании посредством освоения основ программирование в среде Scratch.

### **Задачи программы:**

#### ***Обучающие задачи:***

- сформировать представление об основах программирования в Scratch;
- освоить основные инструменты и операции работы в Scratch;
- изучить основные принципы создания простых проектов в среде Scratch (анимация и игры);
- научатся поэтапно создавать компьютерную игру. Создавать программу для перемещения объекта по игровой карте в одном направлении и в пространстве из нескольких связанных между собой комнат. Разрабатывать интерфейс для Scratch проекта;
- научить рисовать в векторном и растровом графических редакторах, изменять звук, вводить, выводить и обрабатывать информацию в среде Scratch;
- регистрироваться на сайте сообщества Scratch. Просматривать проекты сообщества и публиковать собственные проекты.

#### ***Развивающие задачи:***

- развивать познавательный интерес, внимание, память;
- развивать логическое, абстрактное и образное мышление;
- развивать воображение и фантазию через создание простых проектов в среде Scratch анимация и игры);
- развить способности к самостоятельной работе и анализу проделанной работы;
- развивать познавательную и творческую активность в использовании информационных технологий;

#### ***Воспитательные задачи:***

- воспитывать самостоятельность, уверенность в своих силах;
- формировать творческий подход к поставленной задаче;
- воспитывать ценностное отношение к знаниям, интерес к изучению нового;
- воспитывать стремление добиваться поставленной цели;
- воспитывать чувство ответственности за свою работу.

### **Планируемые результаты освоения программы**

#### **Предметные результаты:**

В результате изучения раздела обучающиеся:

- освоят основные инструменты и операции работы в Scratch, команды движения;
- научатся создавать простые проекты в среде Scratch (анимация и игры);

- научатся использовать векторный и растровый графический редактор, изменять звук, вводить, выводить и обрабатывать информацию в среде Scratch;

#### Метапредметные результаты:

- развитие познавательного интереса, внимания, памяти;
- развитие логического, абстрактного и образного мышления;
- развитие воображения и фантазии через создание простых проекты в среде Scratch (анимация и игры);
- формирование навыка сознательного и рационального использования программирования в своей деятельности;
- внесение корректив в текущую деятельность на основе анализа изменений для получения запланированных характеристик продукта/результата;
- умение выбирать варианты и самостоятельно искать средства/ресурсы для решения задачи и достижения цели;

#### Личностные результаты:

- воспитание самостоятельности, уверенности в своих силах;
- использование творческого подхода к поставленной задаче;
- воспитание ценностного отношения к знаниям, интереса к изучению нового;
- сформированное чувство ответственности за свою работу;
- готовность и способность к созданию творческих работ созидательной направленности (игр, анимационных роликов и т.п.).

### Учебный план

№ п/п	Название раздела, темы	Количество часов <sup>6</sup>			Формы аттестации (контроля)
		Всего	Теория	Практика	
<b>5.</b>	<b>Раздел V. Использование программы Scratch для создания мини-игр</b>				
5.1.	Тема 5.1. Виды компьютерных игр. Алгоритмическая разработка листинга программы.	2	1	1	Педагогическое наблюдение
5.2.	Тема 5.2. Разработка базовых спрайтов для игры. Формирование базовых скриптов.	2	1	1	Педагогическое наблюдение
5.3.	Тема 5.3. Синхронизация работы скриптов для разных спрайтов	2	1	1	Педагогическое наблюдение
5.4.	Тема 5.4. Переход из одной сцены в другую. Создание интерфейса игры.	2	1	1	Педагогическое наблюдение
5.5.	Тема 5.5. Сообщество Scratch в Интернете. Просмотр и публикация проектов.	1	-	1	Педагогическое наблюдение
<b>Итого часов</b>		<b>9</b>	<b>4</b>	<b>5</b>	

<sup>6</sup> Для всех занятий учебный час устанавливается продолжительностью 45 минут.

## Содержание

### Раздел V. Использование программы Scratch для создания мини-игр

#### Тема 5.1. Виды компьютерных игр. Алгоритмическая разработка листинга программы.

**Теория.** Компьютерные игры – вред или польза. Виды компьютерных игр. Этапы разработки игр программистами (1 час).

**Практика.** Алгоритмическая разработка проекта, запись на естественном языке событий и точек взаимодействия героев будущей игры (1 час).

#### Тема 5.2. Разработка базовых спрайтов для игры. Формирование базовых скриптов.

**Теория.** Логика создания персонажей для игры. Перевод алгоритма, написанного на естественном языке, в коды Scratch (1 час).

**Практика.** Разработка и создание основных спрайтов и их костюмов для будущей игры. Разработка скриптов для спрайтов и объектов (1 час).

#### Тема 5.3. Синхронизация работы скриптов для разных спрайтов

**Практика.** Доработка основного листинга программы с целью установления связей между спрайтами. Тестирование и отладка программы (1 час).

#### Тема 5.4. Переход из одной сцены в другую. Создание интерфейса игры.

**Теория.** Односторонний (без возможности вернуться назад) переход из одного пространства в другое. Понятие интерфейса. Элементы интерфейса. Основные принципы дизайна интерфейсов. Обратная связь. Необходимые элементы меню (1 час).

**Практика.** Создать программу для перемещения объекта по игровой карте и разработать интерфейс для Scratch-проекта (1 час).

#### Тема 5.5. Сообщество Scratch в Интернете. Просмотр и публикация проектов.

**Практика.** Правила работы в сети. Интернет-сообщества. Сообщество Scratch. Регистрация на сайте. Использование заимствованных кодов и объектов. Просмотр проектов сообщества и публикация собственных проектов (1 час).

### Календарный учебный график

№ п/п	Месяц	Число	Время проведения занятия	Форма занятия	Количество часов	Тема занятия	Место проведения	Форма контроля
1.	декабрь	16	16.00-17.40	Беседа, практическая работа	2	Тема 5.1. Виды компьютерных игр. Алгоритмическая разработка листинга программы	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
2.	декабрь	23	16.00-17.40	Беседа, практическая работа	2	Тема 5.2. Разработка базовых спрайтов для игры. Формирование базовых скриптов	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
3.	январь	13	16.00-17.40	Беседа, практическая работа	2	Тема 5.3. Синхронизация работы скриптов для разных спрайтов	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение

4.	январь	20	16.00-17.40	Беседа, практическая работа	2	Тема 5.4. Переход из одной сцены в другую. Создание интерфейса игры.	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение
5.	январь	27	16.00-16.45	Практическая работа	1	Тема 5.5. Сообщество Scratch в Интернете. Просмотр и публикация проектов	г. Казань, ул. Дубравная, д. 2Д, пом.1020	Педагогическое наблюдение

### Организационно-педагогические условия реализации

Занятия по образовательной программе проходят в группах и индивидуально.

Виды занятий по программе определяются содержанием программы и предусматривают теоретические и практические занятия.

**Формы организации образовательного процесса:** индивидуально-групповые; мини-группы, занятия с использованием индивидуального подхода к каждому ребёнку.

Занятия по программе состоят из теоретической и практической части. Форму занятий можно определить как интерактивное, практическое обучение (практические занятия), теоретическое обучение. Основной формой обучения является практическая работа, которая выполняется малыми группами (2-3 человека) или индивидуальная работа.

**Особенности организации образовательного процесса:**

Форма обучения по программе: индивидуально-групповая.

Занятия в малых группах проводятся при реализации учебного плана с учетом потребностей обучения.

Индивидуальная форма работы используется при общении с конкретными учащимися. Такой подход используется для более детальной отработки навыков и умений, помогает развитию индивидуальных особенностей обучающихся.

В процессе реализации программы используются следующие методы обучения:

*Словесные методы обучения:* рассказ; беседа; объяснение; игра.

*Наглядные методы обучения:* демонстрационный; иллюстративный; наблюдения и др.

Формы организации занятий может варьироваться выбирается с учетом возрастных особенностей детей, уровня освоения учащимися программы и их достижений.

*Структура занятия:*

1. Организационный этап.
2. Мотивационный этап (демонстрация или сюжет, ситуация).
3. Постановка проблемы или задачи.
4. Обсуждение–поиск путей решения (в группах различного состава, в зависимости от задачи).
5. Проектирование и программирование.
6. Подготовка демонстрации.
7. Заключительный этап: презентация работ обучающихся друг другу.

*Учебно-методический комплекс программы состоит из трех компонентов:*

1. учебные и методические материалы для педагогов и обучающихся;
2. система средств обучения;
3. система средств контроля результативности обучения.

### Кадровое обеспечение

Программу реализует педагог(и) дополнительного образования.

Реализация программы обеспечивается педагогическими кадрами, имеющими среднее профессиональное или высшее образование (по направлению, соответствующему направлению

программы, реализуемой организацией, осуществляющей образовательную деятельность) и отвечающими квалификационным требованиям, указанным в квалификационных справочниках, и (или) профессиональным стандартам.

### **Материально-технические условия реализации программы**

Образовательная организация располагает материально-технической базой, обеспечивающей проведение предусмотренных программой теоретических и практических занятий.

Материально-техническая база образовательной организации включает в себя: учебное помещение с мебелью (столами, стульями и пр.).

Имеются два учебных кабинета, оборудованных мебелью и ноутбуками.

Помещения для занятий достаточно просторны и освещены согласно нормам СанПин. Мебель соответствует нормам.

### **Информационные и учебно-методические условия реализации**

#### **Список литературы, используемой педагогом**

1. Рындак В. Г., Дженжер В. О., Денисова Л. В. Проектная деятельность школьника в среде программирования Scratch: учебно-методическое пособие / В. Г. Рындак, В. О. Дженжер, Л. В. Денисова. — Оренбург: Оренб. гос. ин-т. менеджмента, 2009. — 116 с.: ил.
2. Патаракин Е. Д. Учимся готовить в среде Скретч (Учебно-методическое пособие). М: Интуит.ру, 2008. — 61 с.
3. Матяш Н. В. Психология проектной деятельности школьников в условиях технологического образования / Под ред. В. В. Рубцова. Мозырь: РИФ «Белый ветер», 2000. — 285 с.

#### **Список рекомендуемой литературы для детей и родителей**

1. Голиков Д. В. Scratch 3 для юных программистов. — СПб.: БХВ-Петербург, 2020. — 168 с.
2. Мажед Маржи. Scratch для детей. Самоучитель по программированию. - Издательство: Манн, Иванов и Фербер, 2018. — 288 с.
3. Торгашева Ю. Программирование для детей. Мои первые программы на Scratch. — СПб.: Питер, 2018. — 96 с.
4. Мэтью Хайлэнд. Программируем с детьми. Создай 10 веселых игр на Scratch. - Бомбора, 2021. — 176 с.
5. Вордерман Кэрол, Макаманус Шон, Вудкок Джон. Программирование для детей. Иллюстрированное руководство по языкам Scratch и Python. - Издательство: Манн, Иванов и Фербер, 2019. — 224 с.

#### **Интернет-ресурсы**

Библиотека знаний о Scratch (Scratch Wiki) [Электронный ресурс]. - Режим доступа свободный: <https://wiki.scratch.mit.edu>

## РАБОЧАЯ ПРОГРАММА ВОСПИТАНИЯ

Рабочая программа воспитания предназначена для всех групп обучающихся по дополнительной общеобразовательной общеразвивающей программе «Программирование в среде Scratch»

**Цель:** совершенствование важнейших сторон личности обучающегося, таких как развитие самостоятельности, чувства ответственности за свою работу, целеустремленности и заинтересованности в познании мира.

**Задачи:**

- воспитывать самостоятельность, уверенность в своих силах;
- формировать творческий подход к поставленной задаче;
- воспитывать ценностное отношение к знаниям, интерес к изучению нового;
- воспитывать стремление добиваться поставленной цели;
- воспитывать чувство ответственности за свою работу.

**Планируемые результаты реализации программы воспитания:**

Содержание программы воспитания дает возможность формировать у обучающихся такие результаты, как:

- любознательность, активность, целеустремленность и заинтересованность в познании мира;
- самостоятельность, способность без помощи педагога выполнять игровые и учебные задания.

### Содержание работы с обучающимися

Работа с обучающимися включает:

- обучение умениям и навыкам самостоятельной деятельности, самоорганизации, формированию ответственности;
- развитие творческого потенциала обучающихся;
- содействие формированию активной позиции.

### Оценка результативности реализации программы воспитания

Творческие работы (проекты в среде Scratch) позволяют продемонстрировать успехи учащихся в дополнительном образовании.

В процессе реализации программы воспитания используются следующие диагностические методики:

#### Методики диагностики развития личности ребенка

1. *Методика оценки результативности реализации образовательной программы* (Шаршакова Л.Б. Педагогическая диагностика образовательного процесса. Методическое пособие для педагогов дополнительного образования — СПб.: ГБОУ ДОД Дворец детского (юношеского) творчества «У Вознесенского моста», 2013. — 52 с.) из опыта работы ГБУ ДО ДДЮТ Красносельского района Санкт-Петербурга.

2. *Методика самооценки обучающихся и экспертной оценки педагогом компетентности обучающихся* (Сеничева И.О., Ситник Л.Р., Результативность образовательного процесса УДОД. Итоги реализации вариативных программ исследования // Материалы согласованного исследования проблем дополнительного образования / Информационно-методический бюллетень. – СПб., 2007. – № 6. – 122 с.).

### Карта самооценки

Оцените, пожалуйста, по пятибалльной шкале знания и умения, которые вы получили, при этом впишите соответствующую цифру (1 – самая низкая оценка, 5 – самая высокая).

№ п/п	Характеристика знаний, умений, навыков	Шкала оценки					Сумма баллов	Результат
		1	2	3	4	5		
1.	Освоил теоретический материал по разделам и темам программы (могу ответить на вопросы педагога)							
2.	Понимаю специальные термины, используемые на занятиях							
3.	Научился использовать полученные на занятиях знания в практической деятельности							
4.	Научился самостоятельно выполнять творческие задания							
5.	Умею воплощать свои творческие замыслы							
6.	Могу научить других тому, чему научился сам на занятиях							
7.	Мои достижения в результате занятий							

МЕТОДИЧЕСКИЕ (ДИДАКТИЧЕСКИЕ) И ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

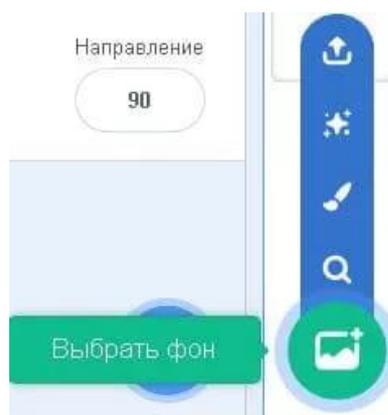
Практические занятия

Тема 2.1. Сцена. Редактирование фона. Добавление фона из файла.

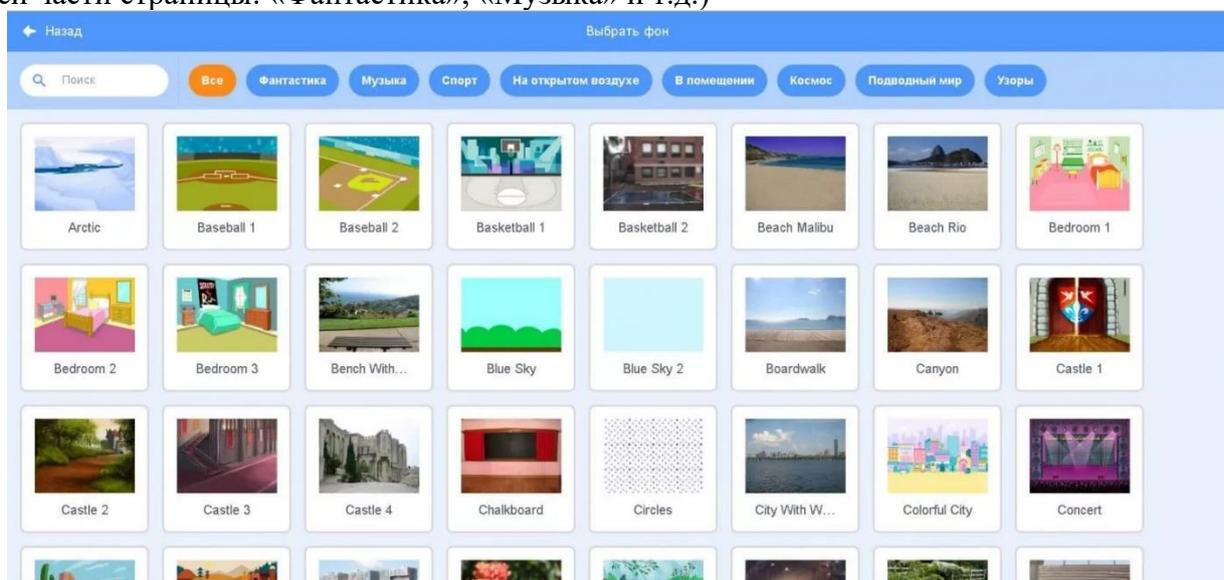
**Практика.** Создание фона сцены на выбранную учащимся тему.

Фон в Scratch — это задний план проектов в Scratch. Существует несколько способов, как добавить его в проект: вы можете выбрать понравившийся из библиотеки готовых фонов или импортировать свое изображение.

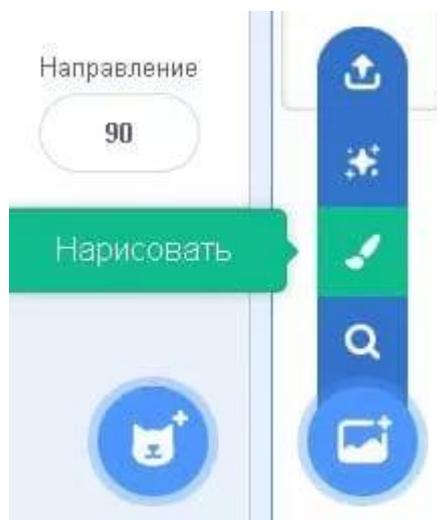
Чтобы выбрать готовый фон, заходим в библиотеку Scratch. Для этого в правом нижнем углу нажимаем на значок «Выбрать фон».



После открытия библиотеки кликаем на нужное изображение, можно воспользоваться функцией «поиск» или отсортировать все предложенные картинки по категориям (они расположены в верхней части страницы: «Фантастика», «Музыка» и т.д.)

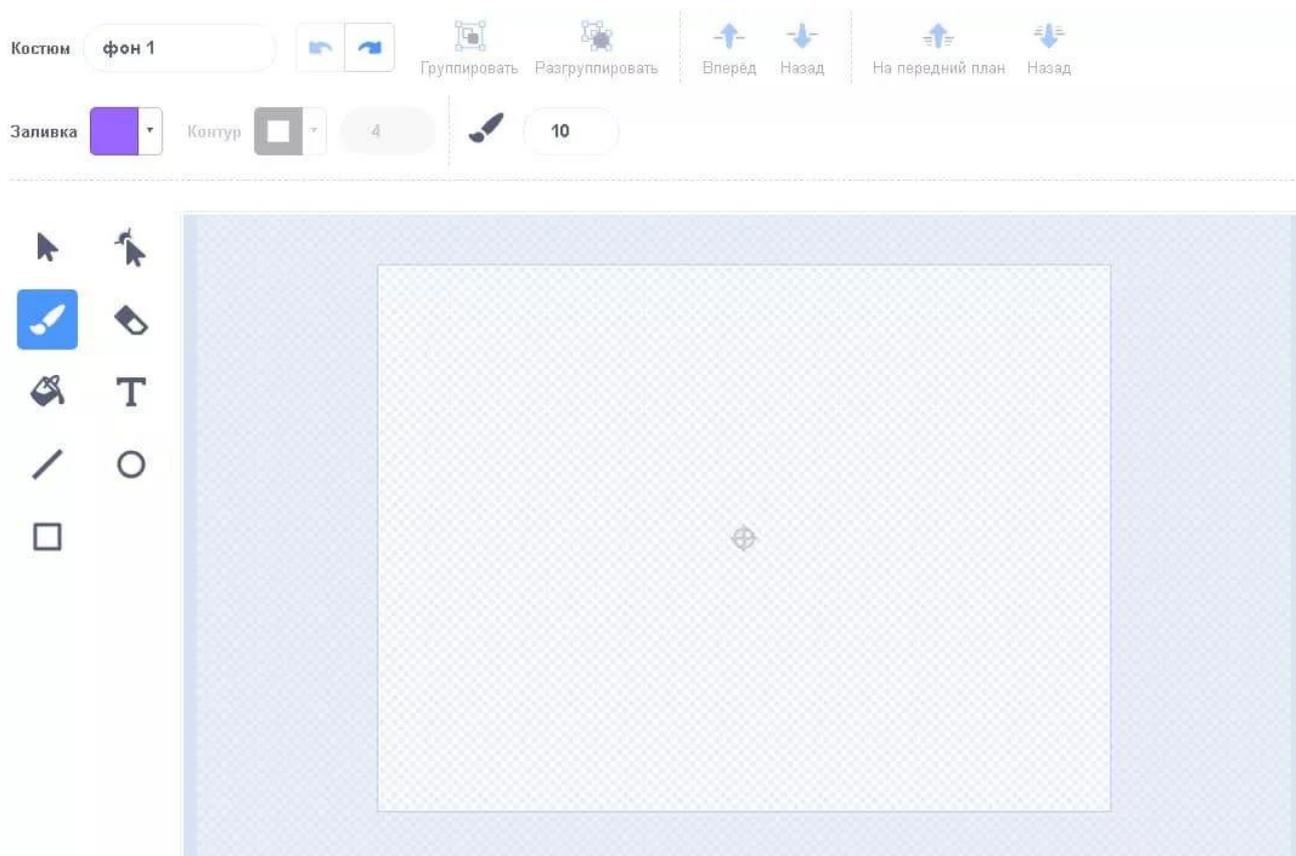


Теперь попробуйте нарисовать что-нибудь самостоятельно, для этого нажмите на значок «Кисть».



Далее откроется редактор Scratch, где вы можете рисовать фон самостоятельно.

- «Заливка» регулирует цвет вашего объекта (на картинке она находится в левом верхнем углу). С его помощью можно окрашивать определенное границы пространство в любой цвет.
- «Ввод текста»: в редакторе можно вводить текст (на панели инструментов эта функция обозначена буквой «Т»).
- Геометрические фигуры также можно добавить в редакторе для упрощения рисования (прямая линия, круг и квадрат находятся внизу на панели инструментов).
- «Изменение формы объекта»: инструмент выглядит как точка с наведённым на неё курсором мыши.
- Для редактирования нарисованных линий можно использовать инструмент в виде стирательной резинки.



## Тема 2.2. Понятие спрайтов. Добавление новых спрайтов. Рисование новых объектов.

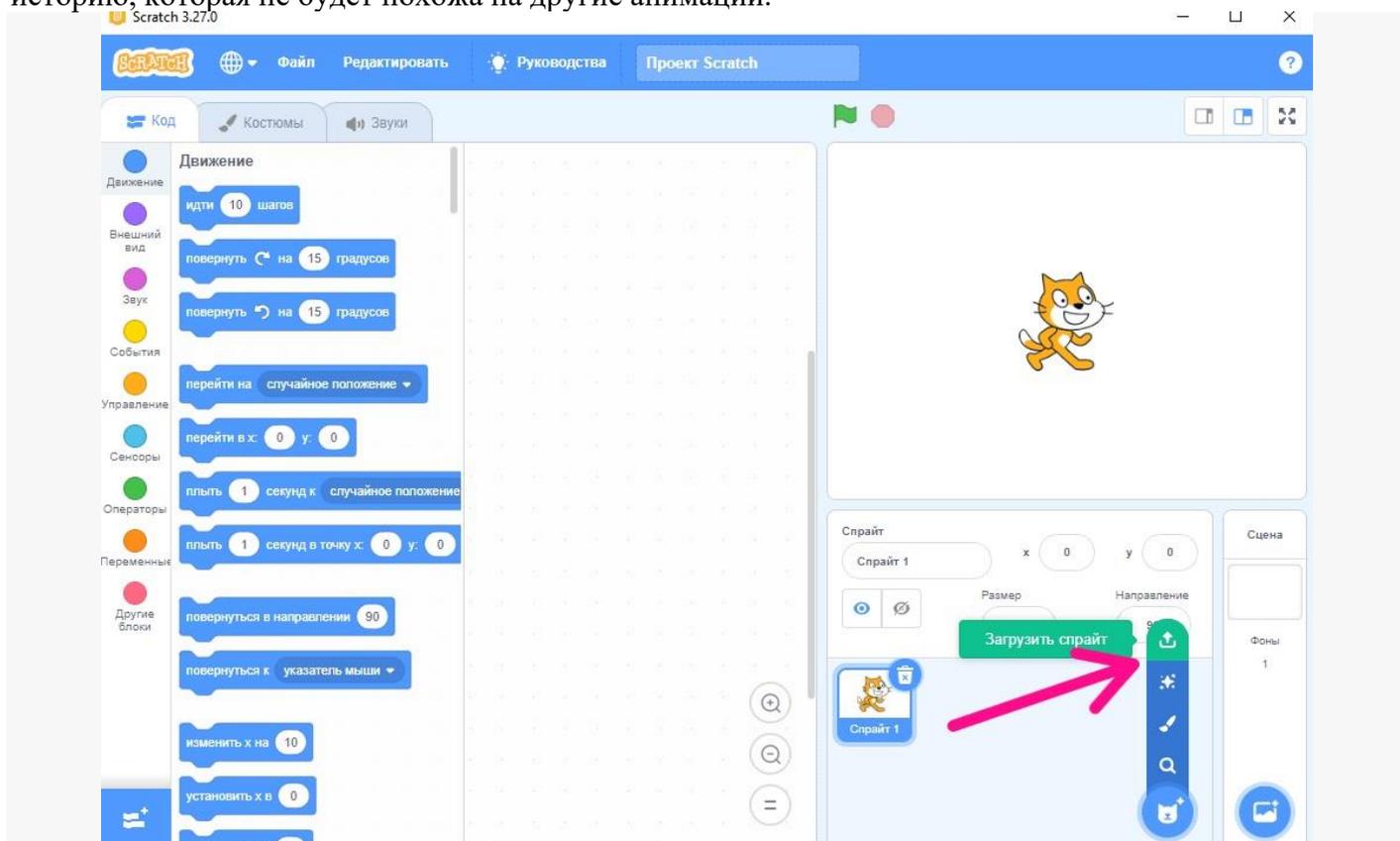
**Практика.** Создание фона сцены и прорисовка основных спрайтов для Scratch-истории (1 час).

С английского языка «спрайт» переводится как фея или эльф.

**Спрайт** – понятие из среды программирования. Оно используется для трактовки графического объекта. Простыми словами спрайт – это персонаж или объект, используемый для создания анимации. Спрайтами являются люди, животные, предметы, транспорт и символы. Каждый спрайт имеет свой программный код. Пользователь взаимодействует с объектом, создаёт для неё команды.

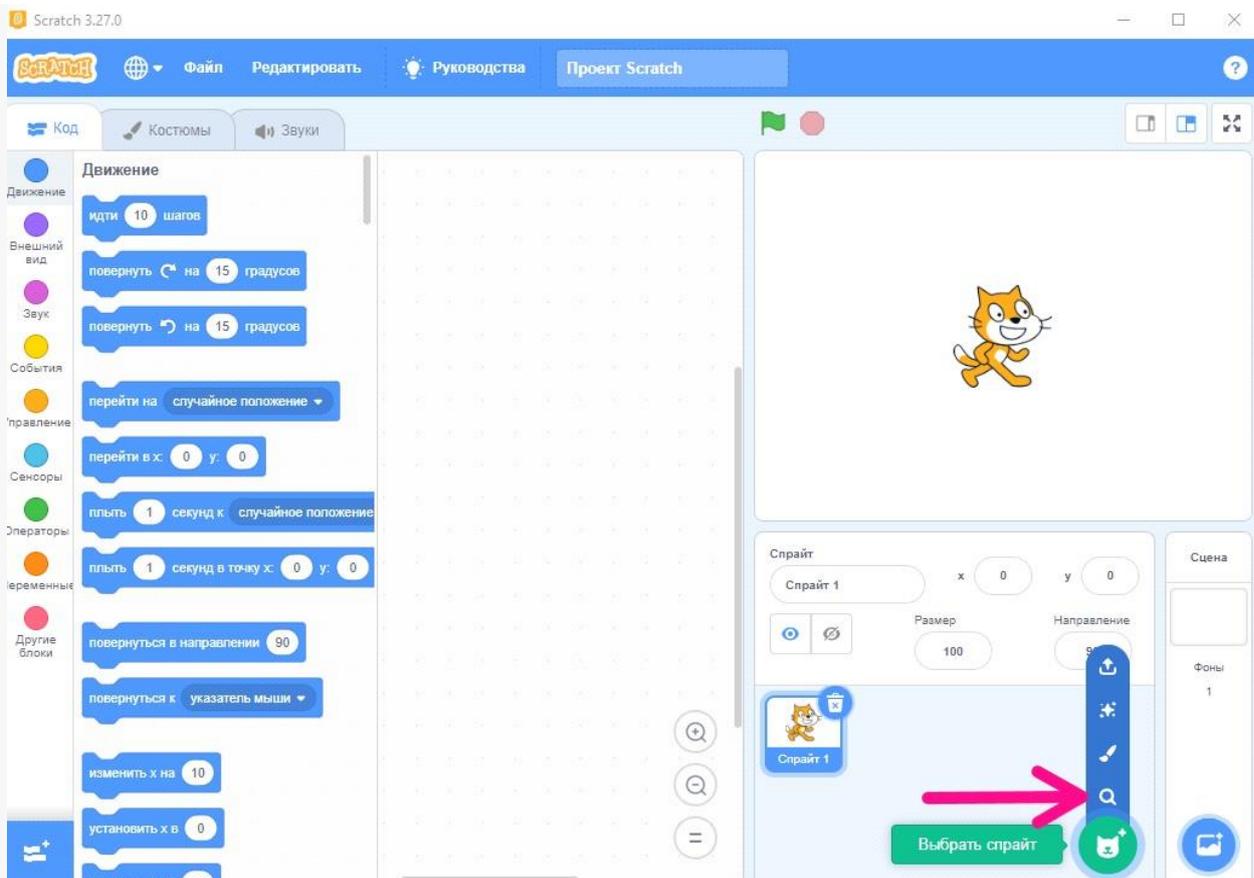
В Scratch нельзя работать с пустым полем, так как главный объект – спрайт. Именно со спрайтом осуществляются все действия. Самый простой способ работы – выбор спрайта из готовых коллекций.

Пользователей не всегда устраивают стандартные варианты, поэтому требуется загрузка картинки с разных источников. Загрузка спрайта может потребоваться, чтобы создать интересную историю, которая не будет похожа на другие анимации.

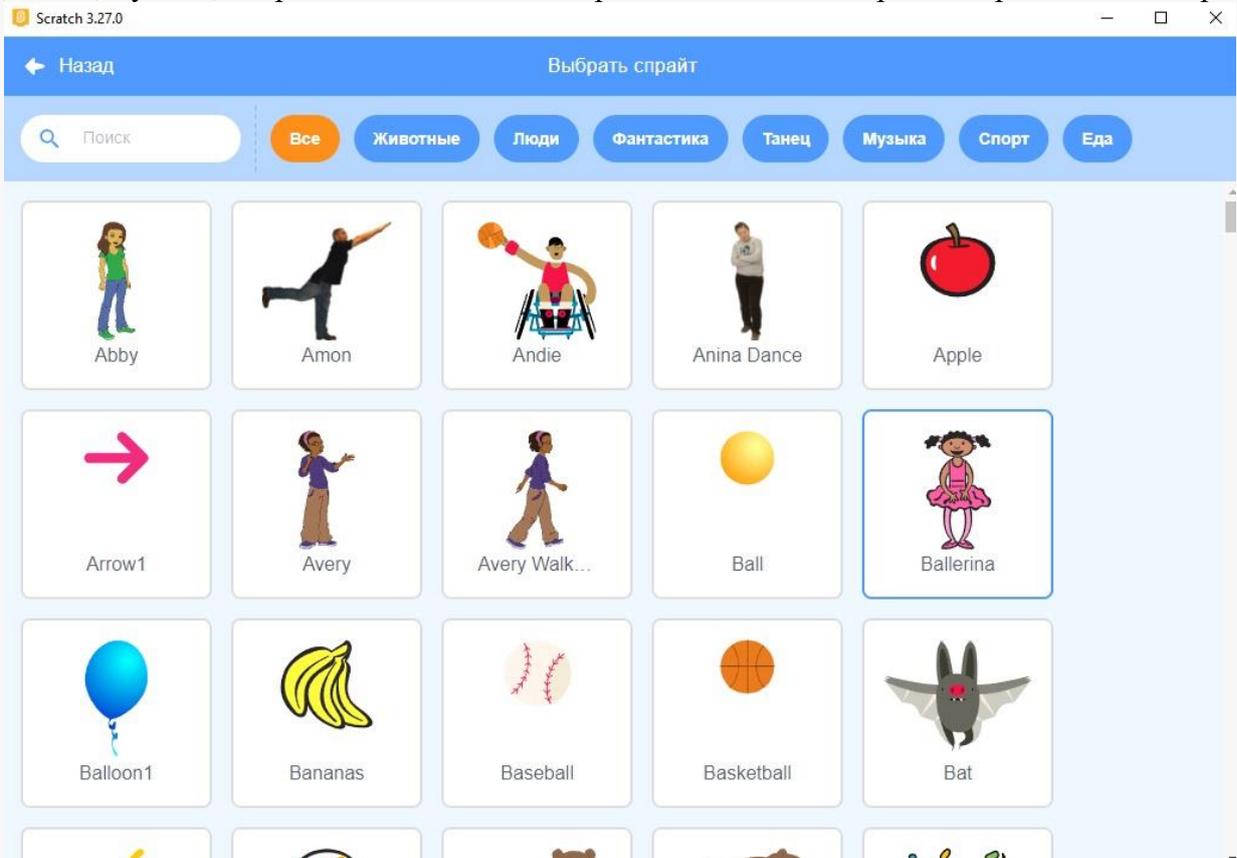


Также загрузка поможет детям дублировать сцены из фильмов или мультфильмов, воссоздать новую историю с участием любимых персонажей.

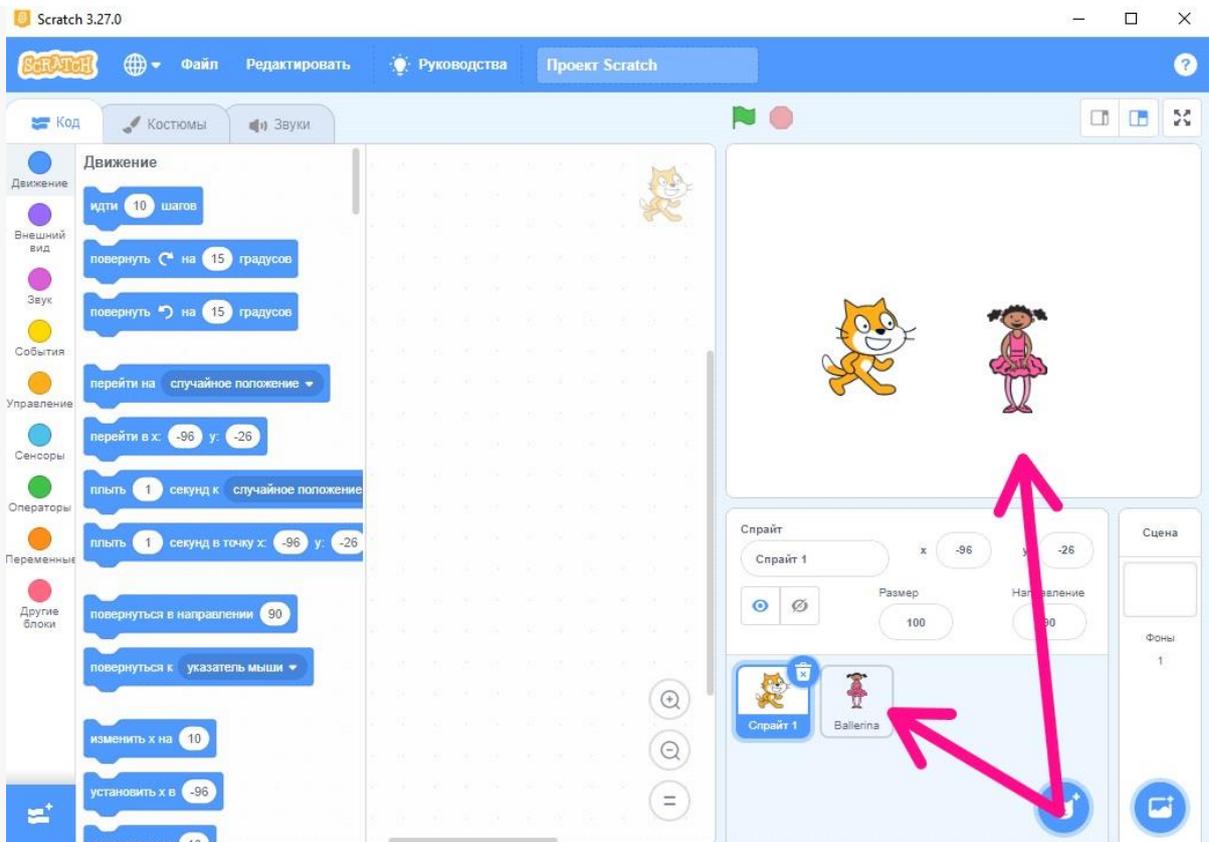
В библиотеке есть множество спрайтов. Чтобы их найти, необходимо кликнуть в правом нижнем углу на иконку «Выбрать спрайт», а затем на значок лупы.



В появившемся окне отобразятся спрайты и разделы: животные, люди, фантастика, еда, буквы, танец, музыка, спорт и так далее. Из этих разделов можно выбрать понравившийся спрайт.

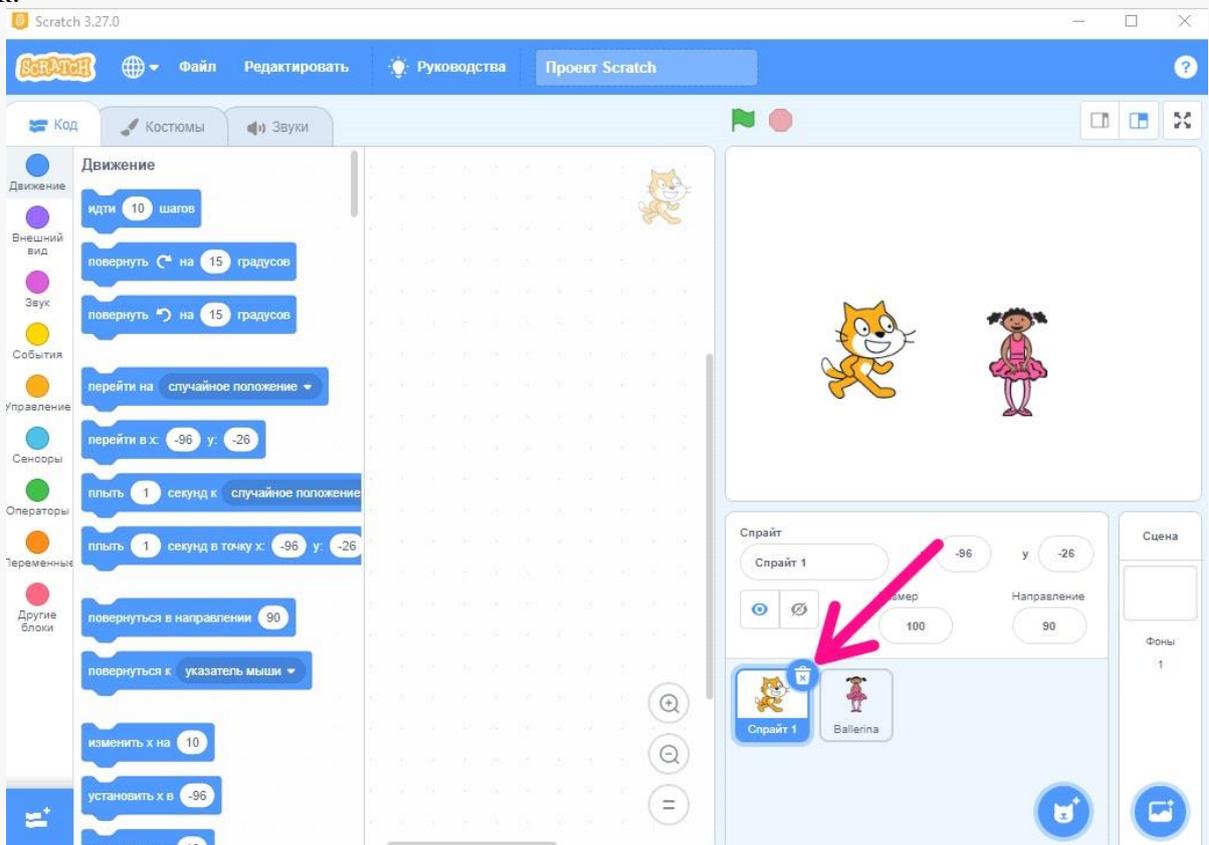


Все добавленные объекты будут размещены в области под Сценой.

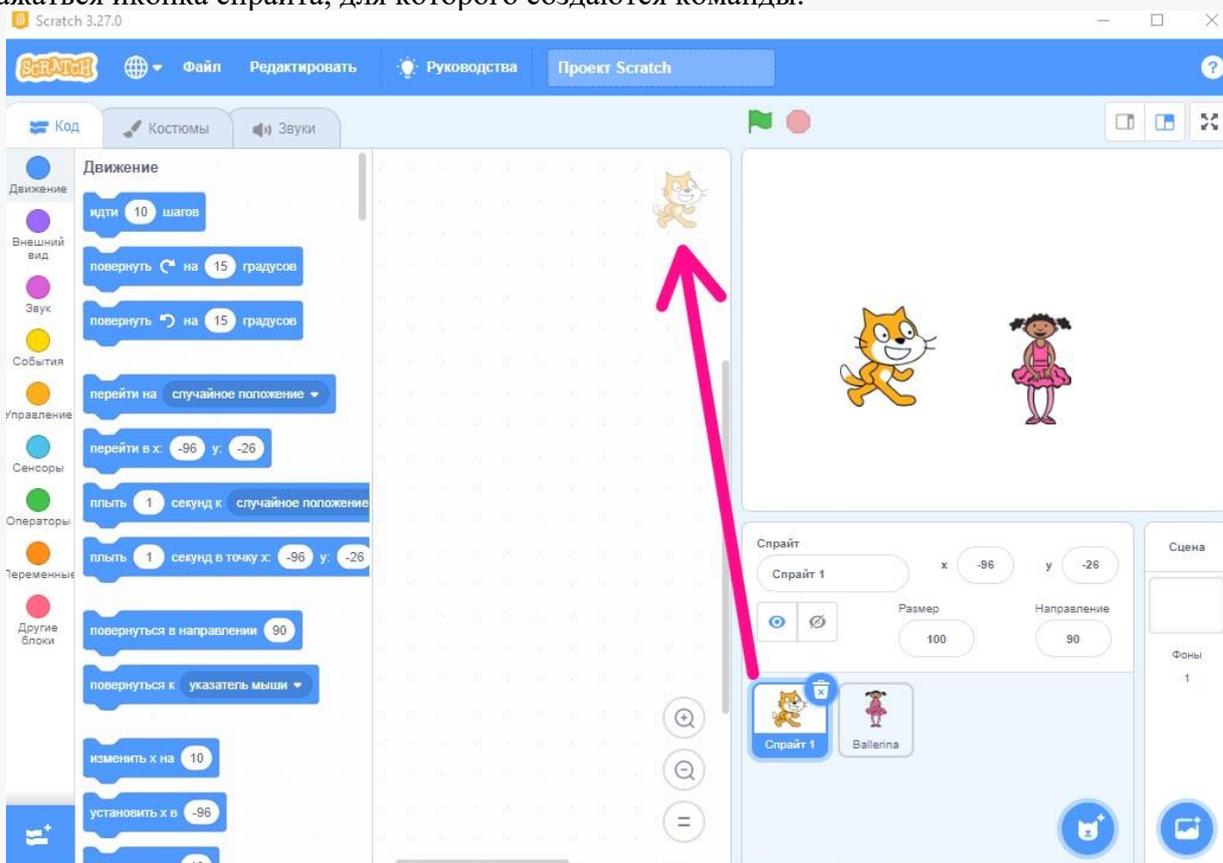


Выбранное изображение можно в дальнейшем редактировать и создать анимацию. У спрайтов из библиотеки есть встроенные костюмы.

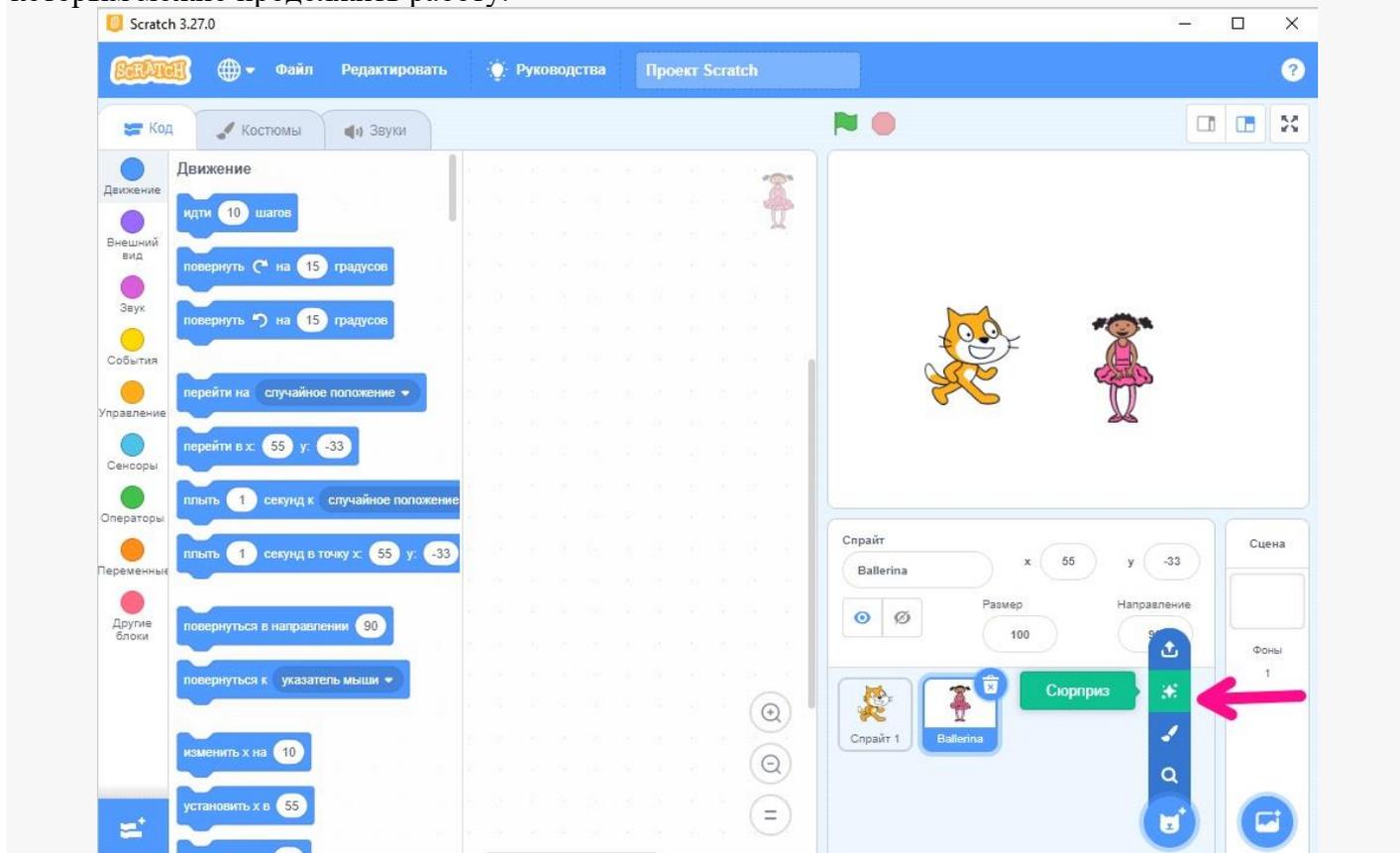
Если спрайт нужно удалить, достаточно нажать на него под свойствами и на появившийся крестик.



В случае, когда пользователь добавил несколько персонажей, чтобы понять, для какого спрайта создаётся задача, нужно обратить внимание на верхний правый угол рабочей области. В углу будет отображаться иконка спрайта, для которого создаются команды.

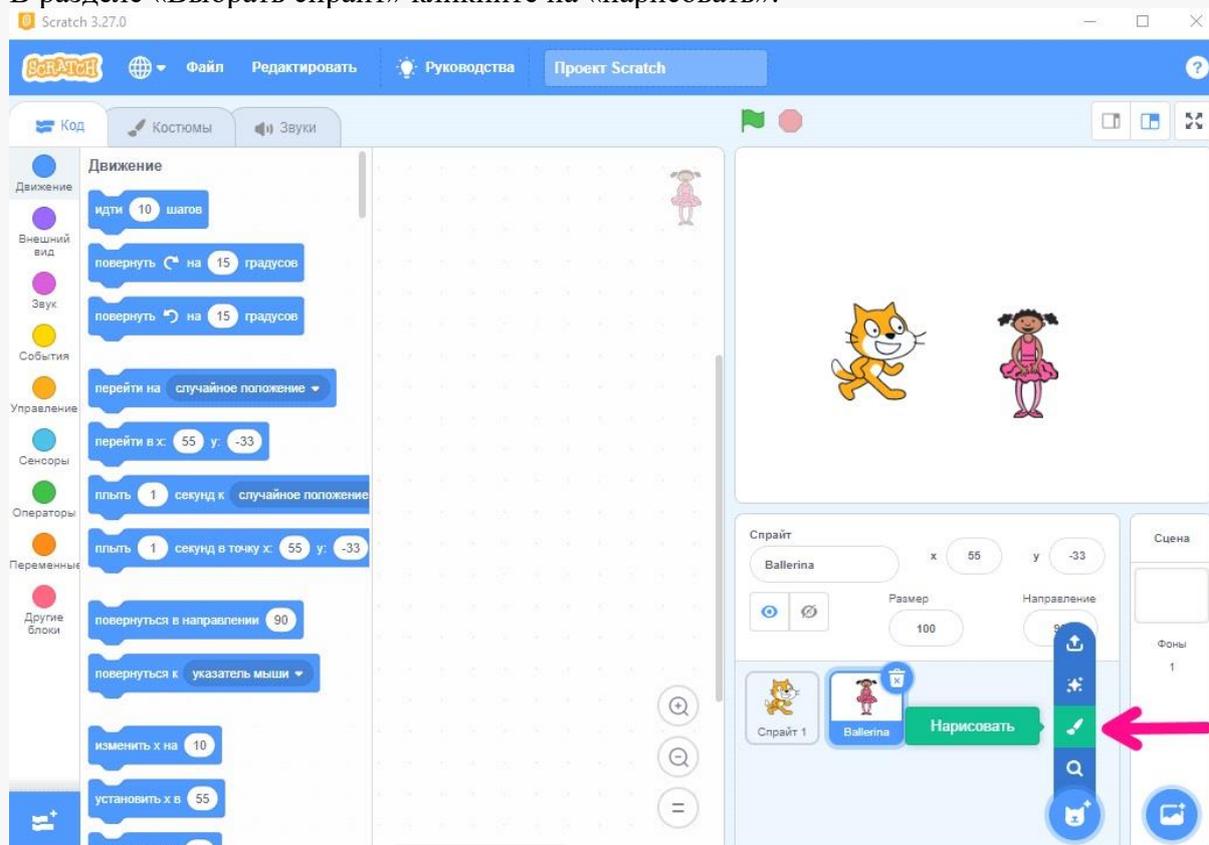


Можно выбрать из библиотеки случайного спрайта. Для этого необходимо во вкладке «Выбрать спрайт» кликнуть по «Сюрприз». На Сцене появится случайно подобранный персонаж, с которым можно продолжить работу.

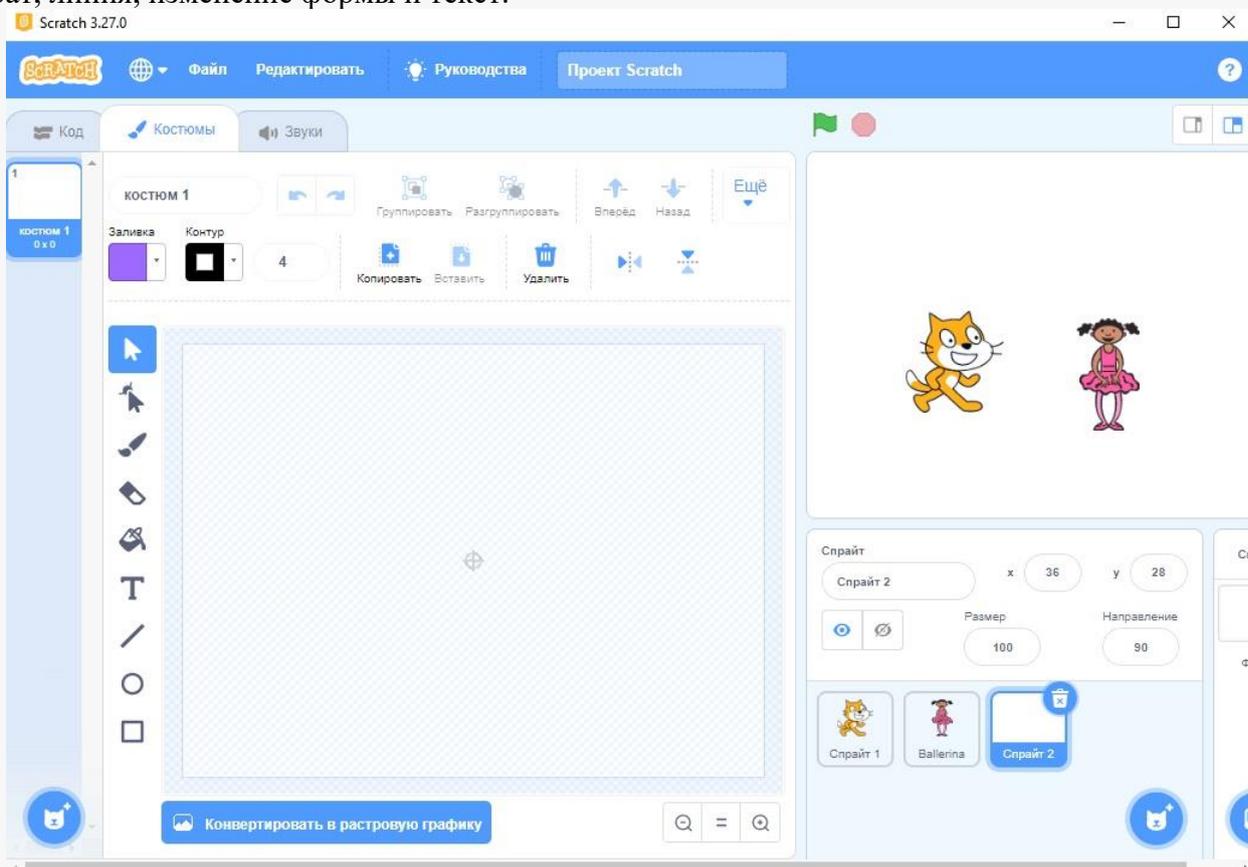


Такая функция полезна, если ребёнок не знает каких персонажей выбрать для своей истории. Рандомные спрайты позволят создать непредсказуемый сюжет и оригинальную анимацию. Если вам не понравились встроенные спрайты, можно создать свои. Спрайты для scratch сделать несложно, достаточно знать основной принцип.

В разделе «Выбрать спрайт» кликните на «нарисовать».



Откроется редактор. В редакторе будут следующие инструменты: кисть, заливка, ластик, круг, квадрат, линия, изменение формы и текст.



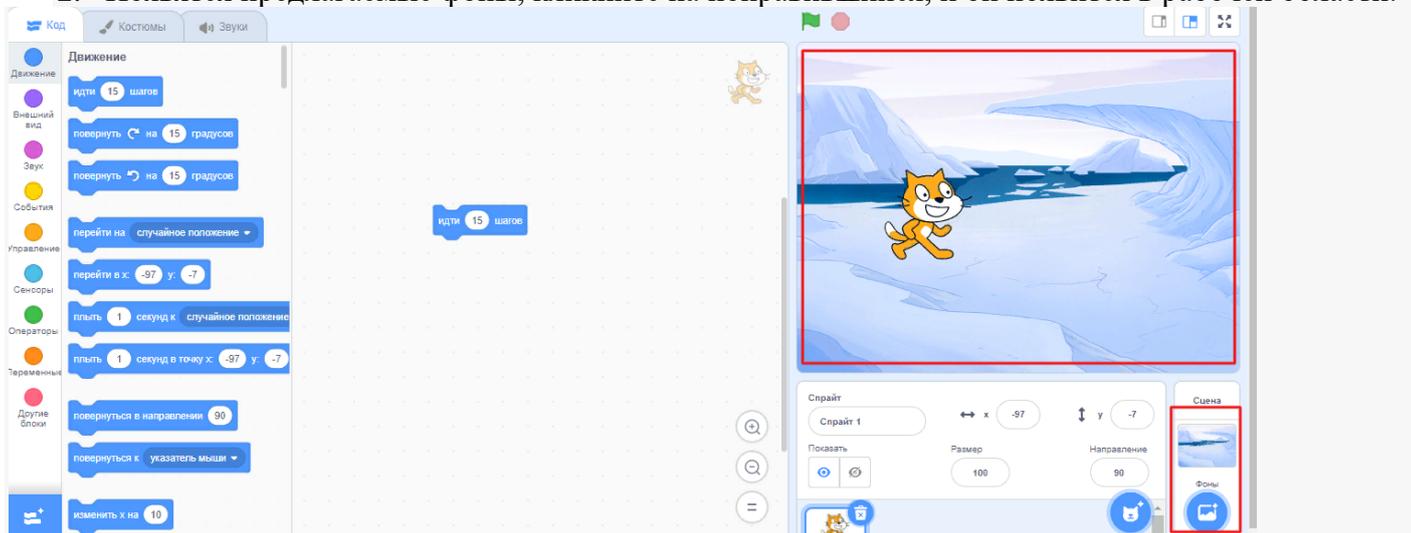
Нажмите на кисть, выберите необходимый цвет и начните рисунок. Если вам нужны ровные линии, воспользуйтесь предлагаемыми геометрическими фигурами. Если вы нарисовали лишний фрагмент, его можно удалить при помощи ластика.

Нарисованные своей рукой спрайты для скретч развивают творческое мышление ребёнка, так как он отражает свои идеи в виде рисунка.

### Арктический котик

Помимо анимации персонажа можно добавить фон. Чтобы это сделать, необходимо:

1. В правом нижнем углу кликнуть на «Выбрать фон» (иконка с изображением).
2. Появятся предлагаемые фоны, кликните на понравившийся, и он появится в рабочей области.

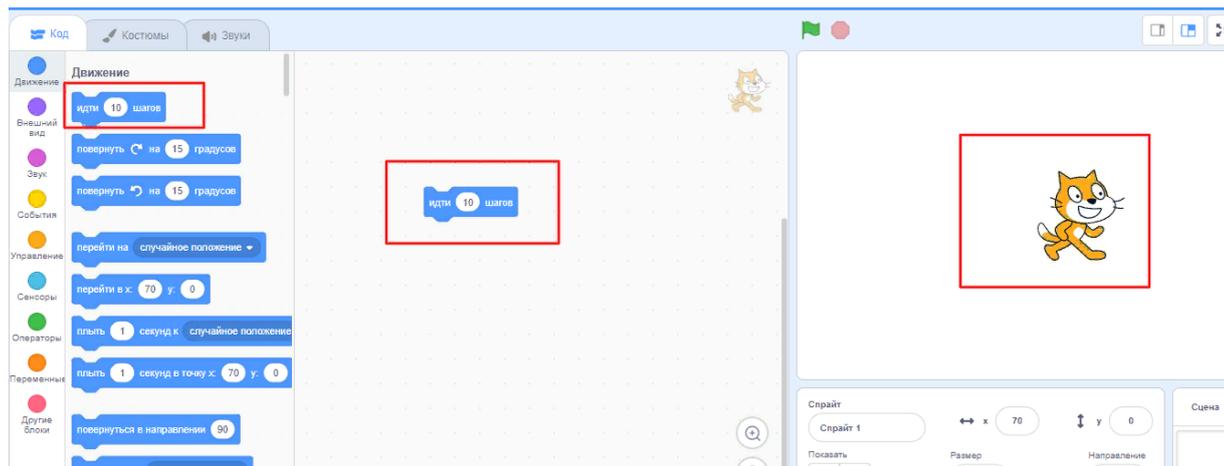


### Тема 3.1. Синий ящик – команды движения. Темно-зеленый ящик – команды рисования.

**Практика.** Создание программ для передвижения спрайтов по сцене. Создание программ для рисования различных фигур (1 час).

### Идущий котик

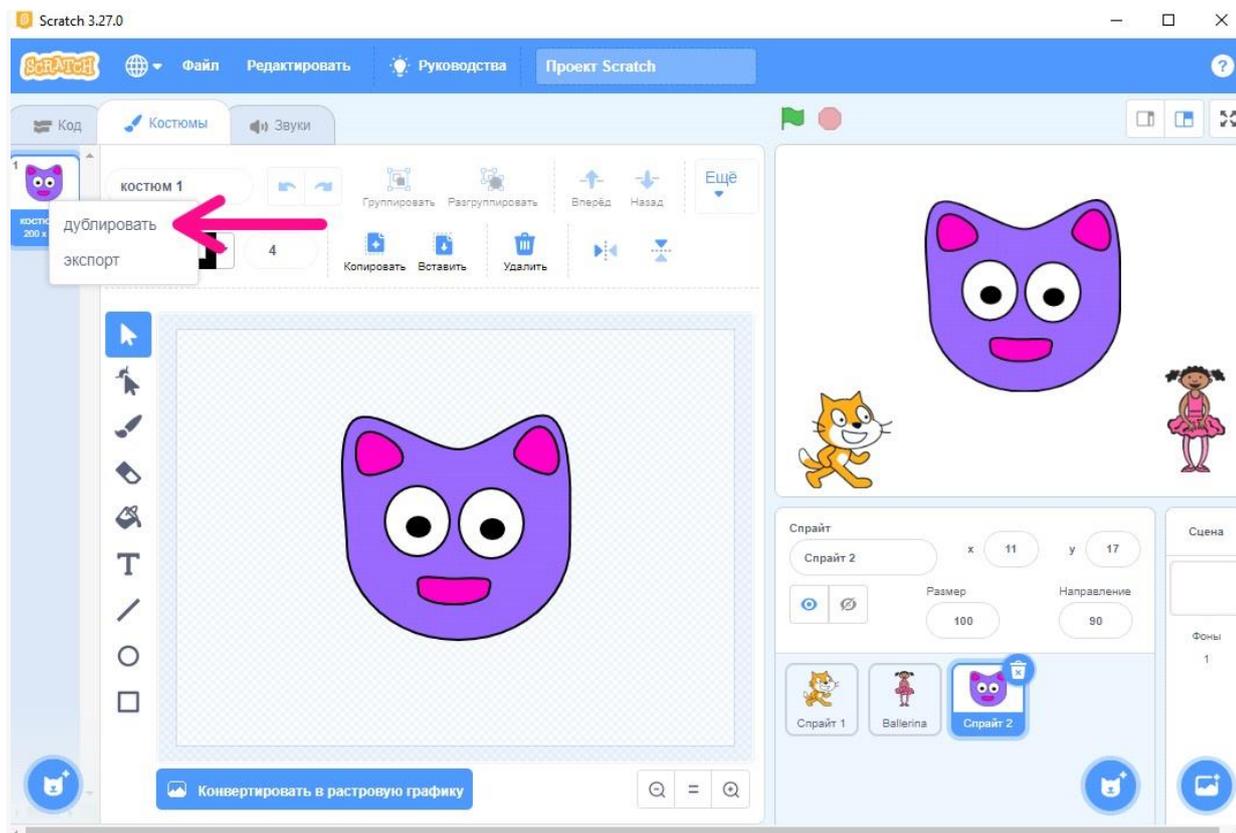
1. Выберите в разделе «Движение» «идти на 10 шагов» (значение можно изменить на своё).
2. Переместите блок в рабочую зону.
3. Чтобы котик начал идти, нужно кликнуть по перенесённому блоку левой кнопкой мыши.



## Тема 3.2. Фиолетовый ящик – внешний вид объекта. Оживление объекта с помощью добавления костюмов

**Практика.** Создание программы для управления внешним видом объекта. Создание Scratch-историй с имитацией хождения и движения объектов (1 час).

Чтобы создать костюмы для нарисованного спрайта, нужно его копировать при помощи нажатия правой кнопкой мыши по иконке спрайта в левом верхнем углу рабочей области. В появившемся меню выберите действие «дублировать». Спрайт дублирован, теперь вы можете добавить для него новые объекты, эмоции и обстановку.



Стандартного или загруженного спрайта можно редактировать. Для этого необходимо перейти во вкладку «Костюмы». Появится редактор, в котором можно с использованием кисти изменить цвет персонажа и добавить дополнительные элементы.

**Отредактированный спрайт можно повторно изменять или создавать для него новые образы и движения.** Для этого нужно дублировать спрайт (в ленте с левой стороны кликнуть на правую кнопку мыши и нажать «дублировать»). После копирования спрайту можно добавить улыбку, другой цвет одежды, новую позу или дополнить его персонажами и предметами.

Кроме инструмента «кисть» можно удалять части спрайта, добавлять геометрические фигуры или изменять форму. Инструмент «Изменение формы» позволит из привычного объекта создать оригинального персонажа. В качестве редактирования можно воспользоваться функциями «Разгруппировать» и «Группировать». При помощи этих клавиш можно разделить на части спрайт или наоборот добавить дополнительные элементы.

Редактирование спрайта включает в себя несколько функций. Первую мы рассмотрели. Обратимся к редактированию в плане анимации. В разделе «Код» можно совершить следующие действия:

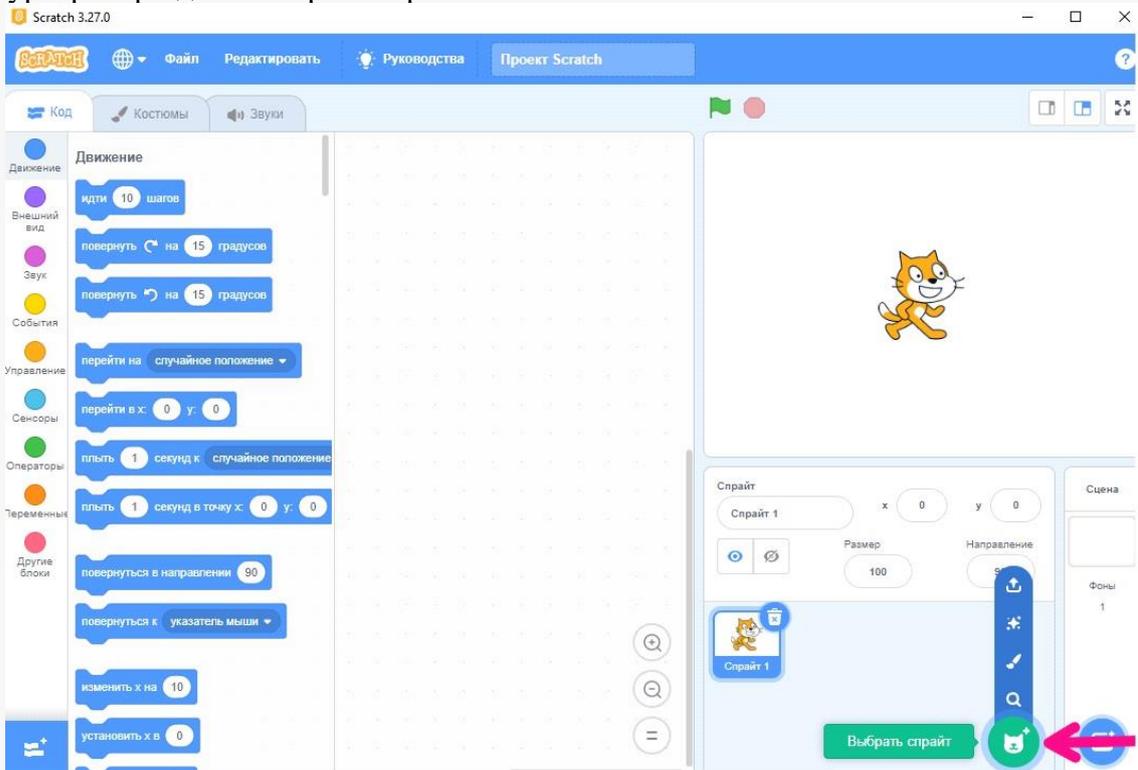
- добавить движение;
- изменить положение и размер;
- повернуть;
- установить плывущий эффект;

- добавить реплики и звук;
- установить фон;
- отрегулировать управление спрайтом.

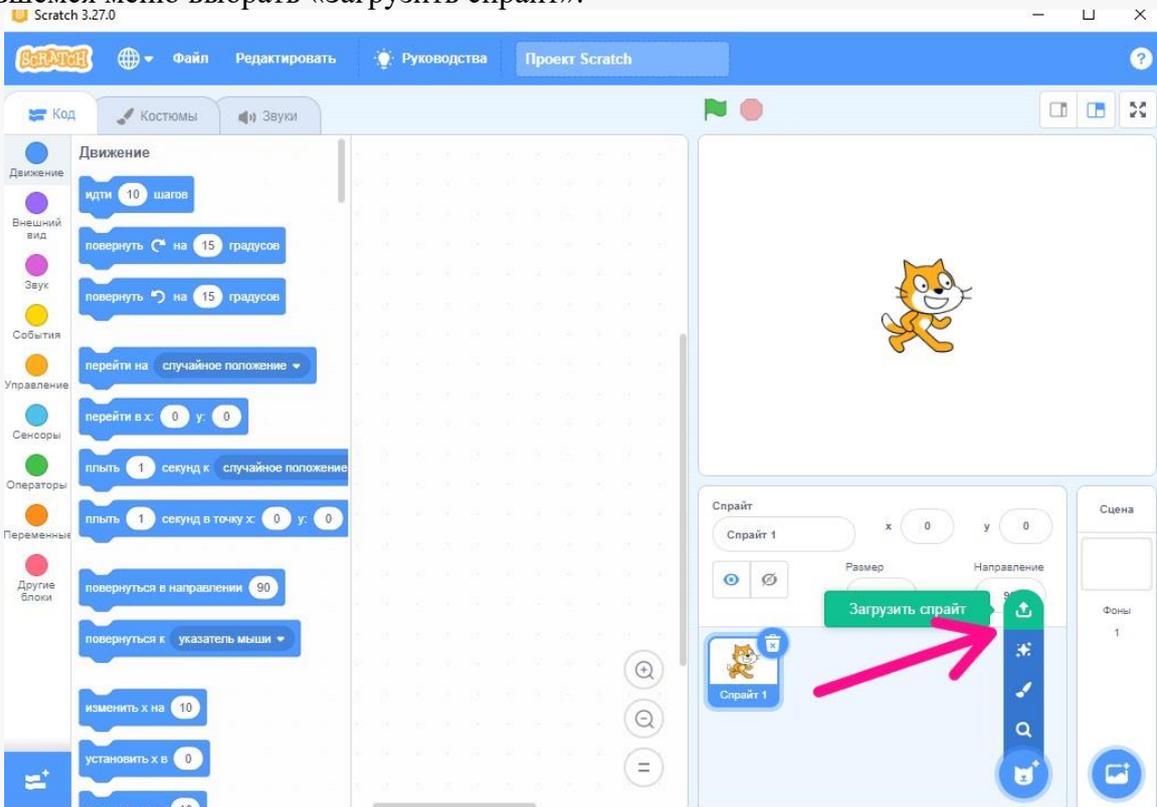
Если в редакторе увеличены пиксели, значит у вас включен режим растровой графики. Чтобы изменить режим, достаточно кликнуть по клавише «Конвертировать в векторную графику». Фантазия детей безгранична. Чтобы воплотить идеи в реальность недостаточно стандартных спрайтов из библиотеки. В программе нет знаменитых мультипликационных персонажей или личностей, поэтому их можно загрузить с устройства.

**Для загрузки из компьютера необходимо:**

- Навести курсор на раздел «Выбрать спрайт».



- В появившемся меню выбрать «Загрузить спрайт».

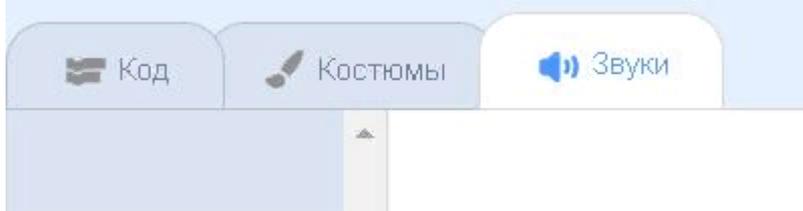


- Из библиотеки устройства выбрать подходящее изображение, нажать «Открыть» и подождать несколько секунд пока оно откроется.

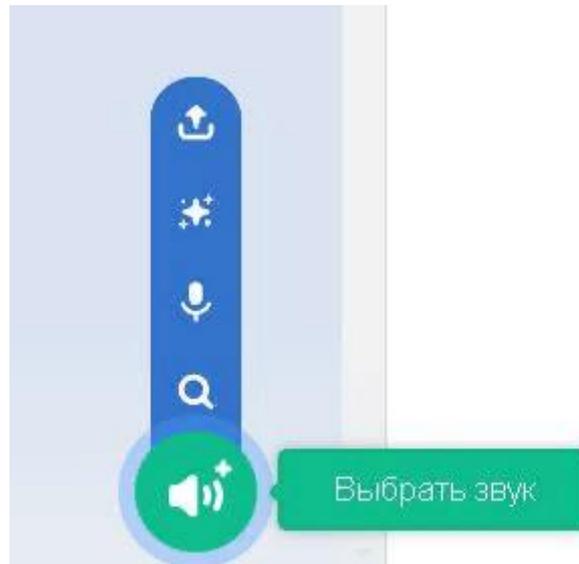
### Тема 3.3. Желтый ящик – контроль. Лиловый ящик – добавление звуков.

**Практика.** Создание программ с элементами управления объектом. Озвучивание Scratch-историй (1 час).

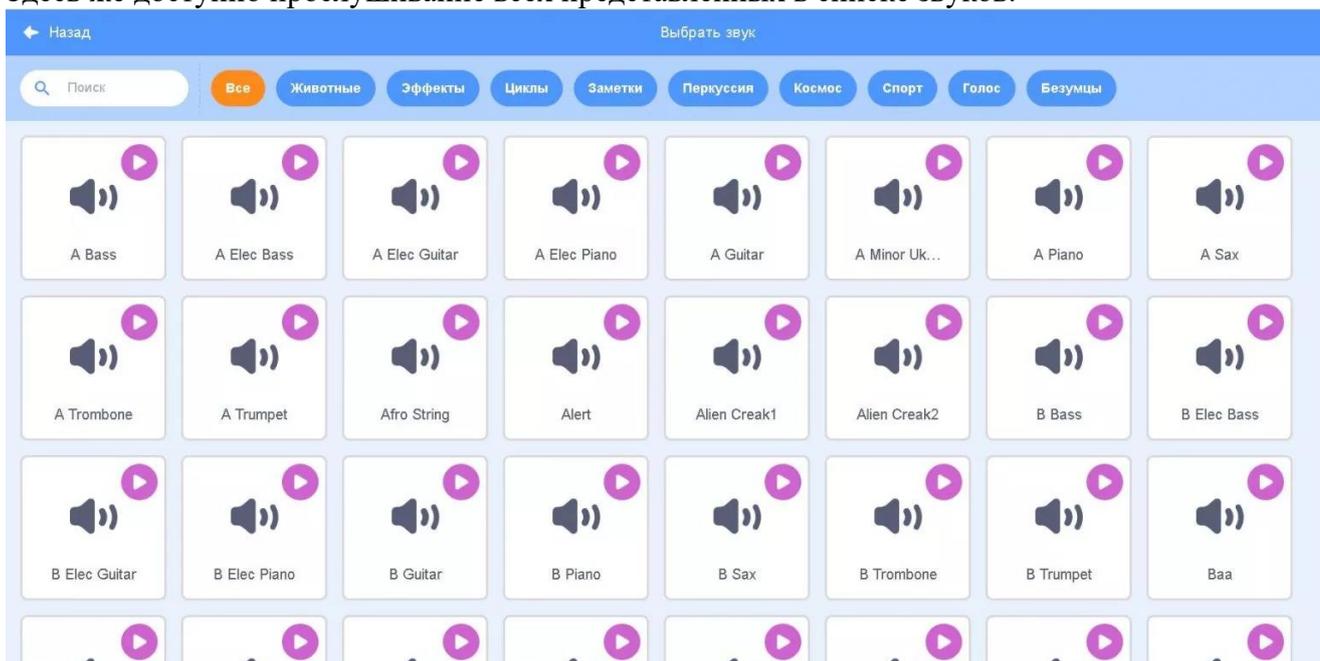
Для того чтобы добавить мелодию в ваш проект в Scratch, нажимаем на вкладку «Звуки».



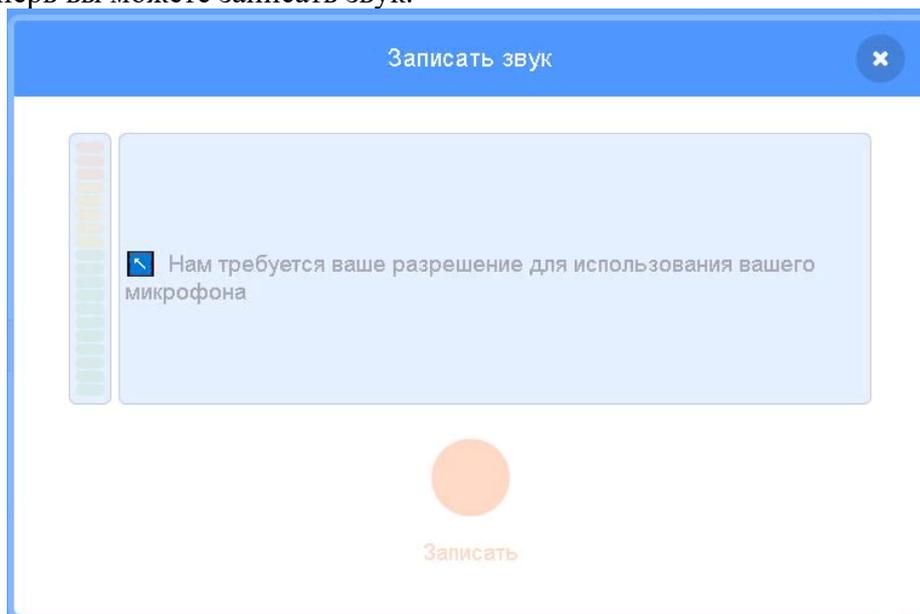
При нажатии кнопки «Выбрать звук» в левом нижнем углу откроется библиотека звуков Scratch.



Здесь все мелодии распределены по категориям («Животные», «Эффекты», «Циклы» и т.д.), найти конкретный звук вы можете, воспользовавшись поиском в левом верхнем углу библиотеки. Здесь же доступно прослушивание всех представленных в списке звуков.



Также вы можете записать свой звук, нажав на значок с микрофоном. Scratch потребует разрешение для использования микрофона вашего компьютера, в левом верхнем углу нажмите «Разрешить». Теперь вы можете записать звук.

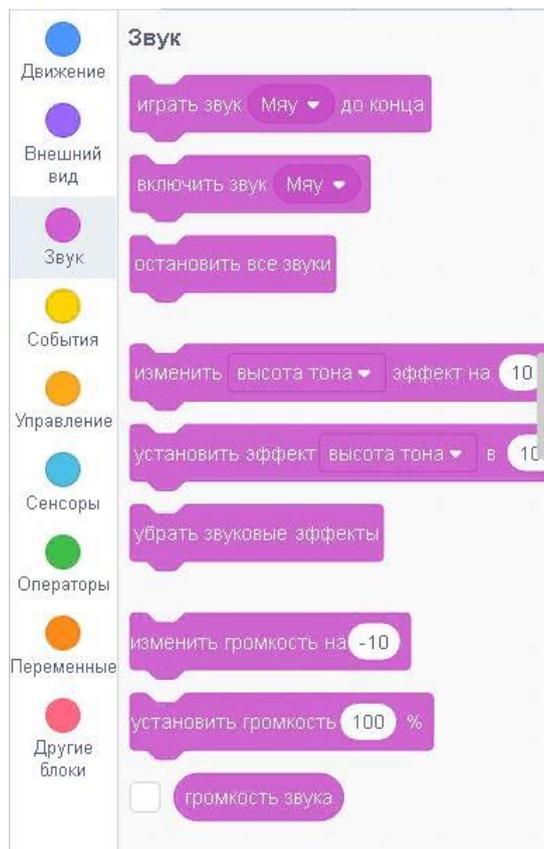


В Scratch также доступен импорт звука с компьютера пользователя, для этого нужно нажать на кнопку «загрузить звук» и в открывшемся окне выбрать файл формата «wav» или «mp3». Новые звуки вы можете скачать в бесплатных библиотеках, в таком случае файлы вероятнее всего будут находиться в разделе «Загрузки», или найти подходящую мелодию на своём устройстве.

После выбора звука откроется редактор звуков. Здесь вы можете изменять громкость и скорость звуковой дорожки, пользоваться функциями усиления и затухания звука. Тут доступен эффект «робот», можно копировать звук и добавлять кнопкой «вставить» к звуковой дорожке, тем самым дублируя его.



Перейдём к звуковым командам, для того чтобы воспользоваться ими, открыть вкладку «Код». Здесь можно добавить звуковые эффекты, к примеру, изменить высоту тона или звучание слева/справа. А также отрегулировать громкость и настроить звучание мелодии в проекте.

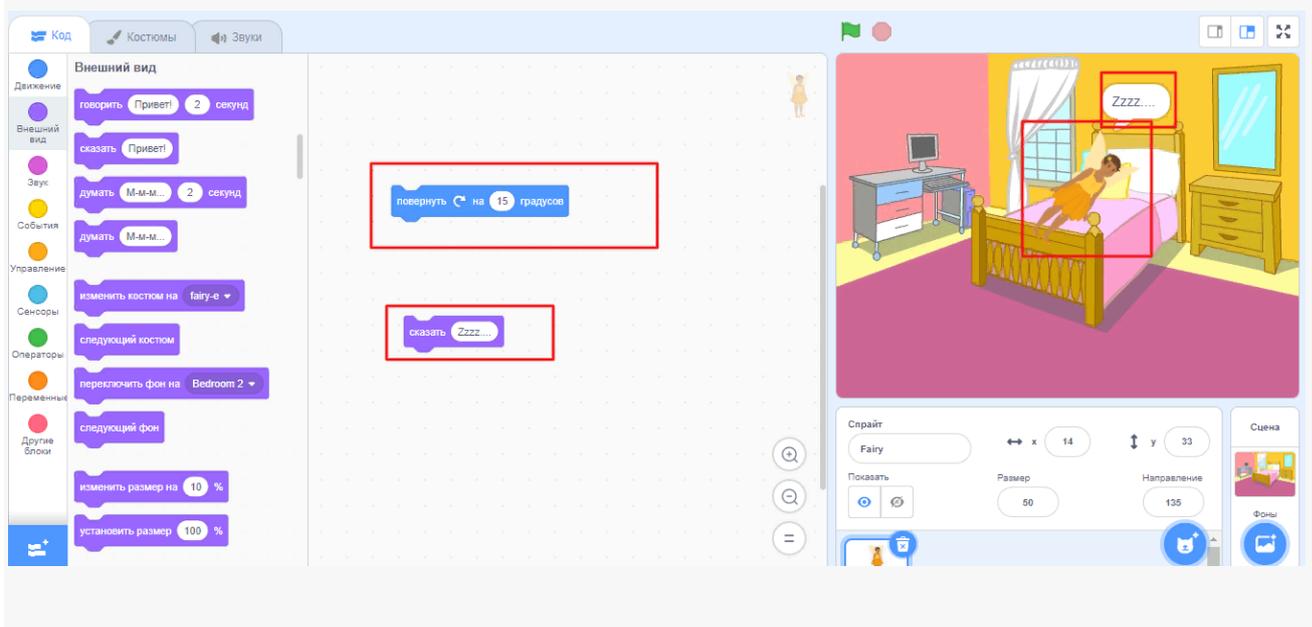


### Пример работы со звуком в Scratch

#### Спящая красавица

Создадим уютную домашнюю атмосферу со спящей принцессой.

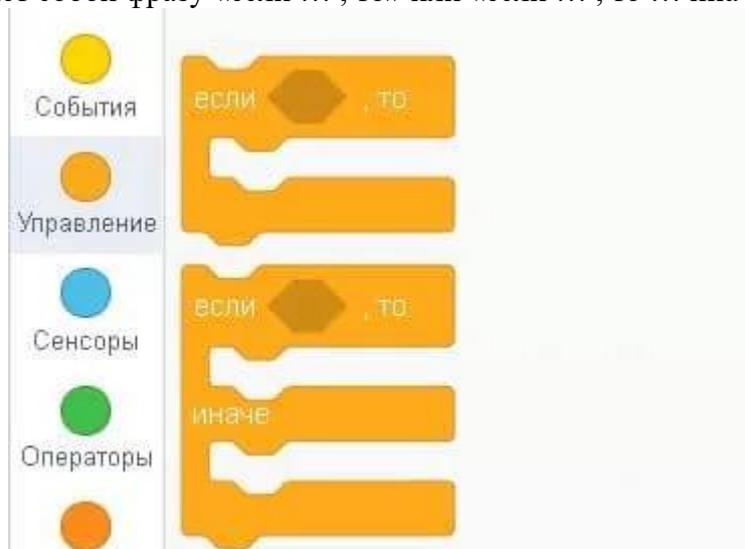
1. Выберите необходимый фон и спрайт.
2. Поместите спрайт в реалистичное положение при помощи клавиши «повернуть на».
3. Чтобы создать обстановку сна, добавьте специальный звук «Zzz...» (выберите «Внешний вид», «Сказать Привет», вместо «привет» напишите свой текст).
4. Перетащите «повернуть на» и «Сказать...» в рабочую область.



### Тема 3.4. Использование в программах условных операторов.

**Практика.** Создание программ с изменением последовательного выполнения скриптов при наличии условий (1 час).

Создать условие вы можете в разделе «Управление», окрашенном оранжевым цветом. Этот инструмент представляет собой фразу «если ... , то» или «если ... , то ... иначе».



#### Команда «если... , то»

В поле после слова «если» вы можете вставить команду, которая в списке представлена в форме шестиугольника. Такие блоки называются «логическое выражение». Это могут быть команды, находящиеся в разделах «Сенсоры» и «Операторы».



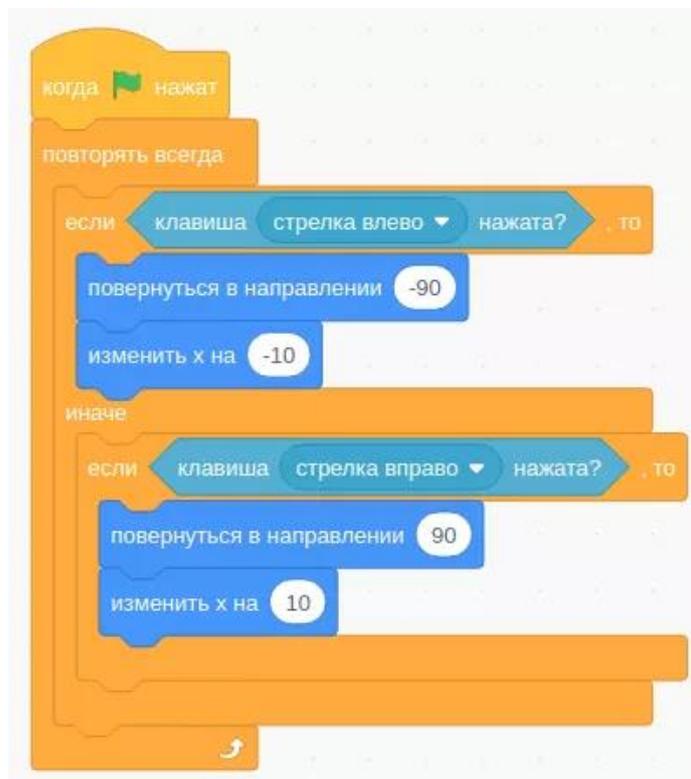
В оператор ветвления вы можете вставлять команды из раздела «Управление». Туда также могут войти и функции под другими обозначениями. Следите за тем, чтобы они подходили по форме.

Если же вы хотите проверить обратное условие, можно добавить второй блок «если». Или же, что в некоторых случаях более правильно, воспользоваться условным оператором с двумя ветками, то есть блоком «если... , то ... иначе».

#### Условия «если... , то... иначе»

Здесь вам нужно будет предусмотреть другой исход событий, противоположный заложенному в блоке «если». Команды в ветке «иначе» выполняются только тогда, когда условие при ветке «если» оказывается ложным. Всегда выполняется либо то, либо другое действие. Никогда вместе они быть выполнены не могут.

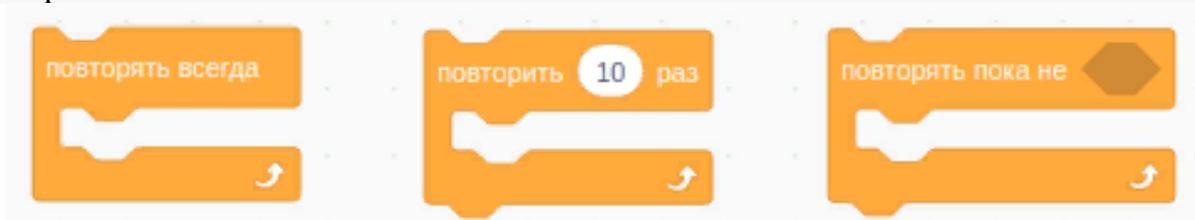
В оператор ветвления после «иначе», как и после «если», можно поместить дополнительные блоки. К примеру, туда можно добавить команды из раздела «Движение».



### Тема 3.5. Функциональность работы циклов. Цикличность выполнения действий в зависимости от поставленных условий.

**Практика.** Создание программ с использованием циклов с фиксированным числом повторений. Создание программ с использованием циклов с предусловием и постусловием (1 часа).

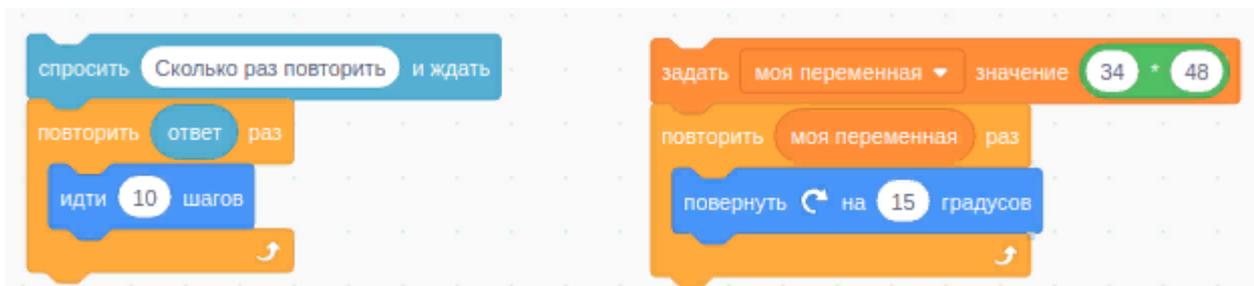
Всего в Scratch три разновидности цикла. Это блоки "повторять всегда", "повторить ... раз", "повторять пока не ...".



Цикл "повторять всегда" не дает скрипту самому закончить свою работу. В результате он работает бесконечно, повторяя и повторяя выполнение вложенных в тело цикла блоков. Прервать такое заикливание можно, остановив работу всей программы (нажать на красный кружок над сценой) или с помощью специальной команды в теле цикла, которая срабатывает при заданном условии.

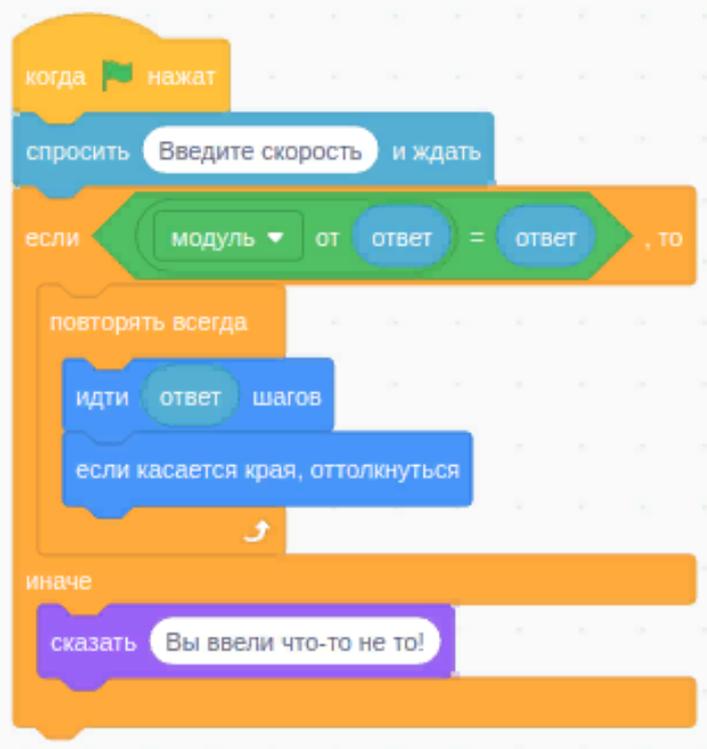
С циклом "повторить ... раз" мы тоже знакомы. Он повторяет вложенные в него команды количество раз, которое указано в его заголовке. Особенностью этого цикла является то, что заранее известно количество повторов.

Конечно, мы можем запрашивать у пользователя количество повторов, или оно может определяться в результате арифметической операции, результат которой мы не знаем. Но все-равно, когда поток выполнения программы доходит до цикла "повторить ... раз", программе уже известно количество повторов.



Новым для нас и самым сложным из всех циклов является цикл "повторять пока не ...". Тело этого цикла выполняется до тех пор, пока условие в его заголовке не вернет правду. Как только условное выражение станет истинным, цикл прекратит свою работу. Но когда условие станет правдой, мы не знаем. Поэтому неизвестно количество повторов цикла.

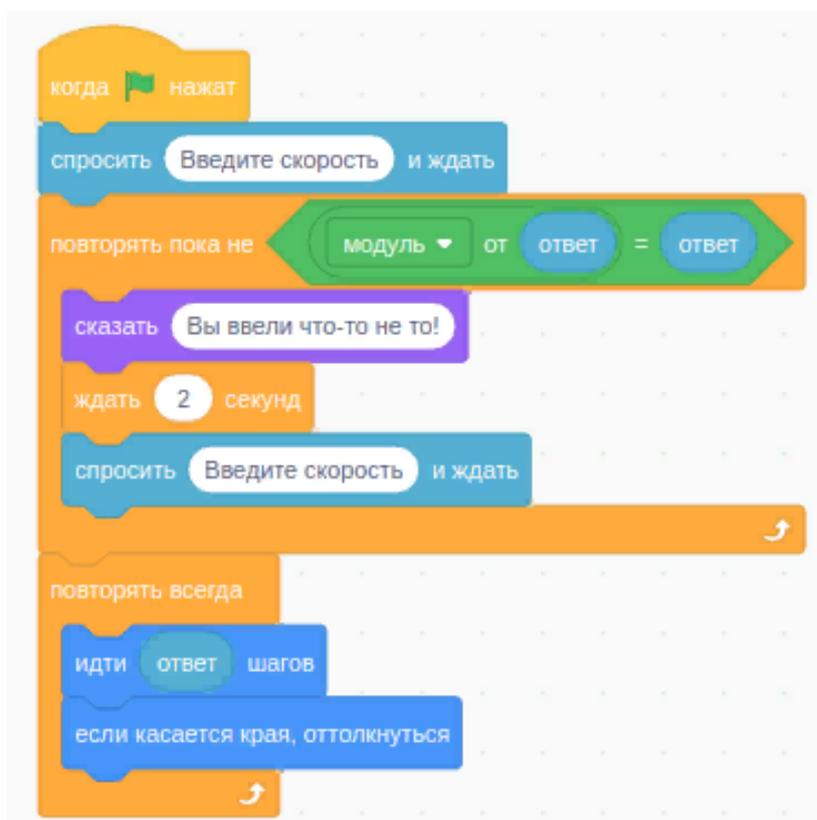
Тело цикла "повторять пока не ..." может вообще ни разу не выполниться, если условие сразу вернет правду. Или же цикл так и не сможет завершиться, если условие всегда будет оставаться ложью.



Если человек вводит положительное число, то программа зацикливается в цикле "повторять всегда" и спрайт ходит по сцене.

Однако, если человек вводит строку или отрицательное число, программа выполняет ветку "иначе" и завершается.

Что если мы хотим, чтобы программа продолжала запрашивать у пользователя число, пока он его не введет. То есть, когда человек вводит строку, программа должна сообщать ему о неправильном вводе и снова запрашивать число. В таком случае не обойтись без цикла "повторять пока не ...".

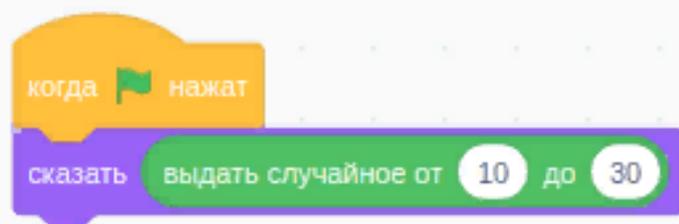


Логика здесь следующая. Пользователя просят ввести скорость. Если он вводит положительное число, то выражение "модуль от ответ = ответ" возвращает правду. Поэтому тело цикла "повторять пока не ..." не будет выполнено ни разу. Поток выполнения программы сразу перейдет к циклу "повторять всегда", внутри которого реализована ходьба спрайта.

Если человек вводит строку, выражение "модуль от ответ = ответ" возвращает ложь. Тело цикла "повторять пока не ..." будет выполняться. Пользователю покажут предупреждение и снова попросят ввести скорость. Если он опять введет строку, ему снова покажут предупреждение и снова попросят ввести скорость.

И так будет до тех пор, пока человек не введет положительное число. Когда это произойдет, мы не знаем. Но только после этого программа сможет перейти к циклу "повторять всегда".

Рассмотрим второй пример использования цикла "повторять пока не ...". Составим программу, в которой человеку предлагается угадать загаданное компьютером число. Однако перед этим познакомимся с блоком, генерирующим случайные цифры.



Команда "выдать случайное от ... до ..." генерирует случайное число в диапазоне от первого указанного числа до второго включительно. Например, если дана команда "выдать случайное от 10 до 30", то программа выдаст любое число, которое не меньше 10 и не больше 30. Это, например, может быть число 10, а может быть 17, или 25. Также возможно получить число 30.

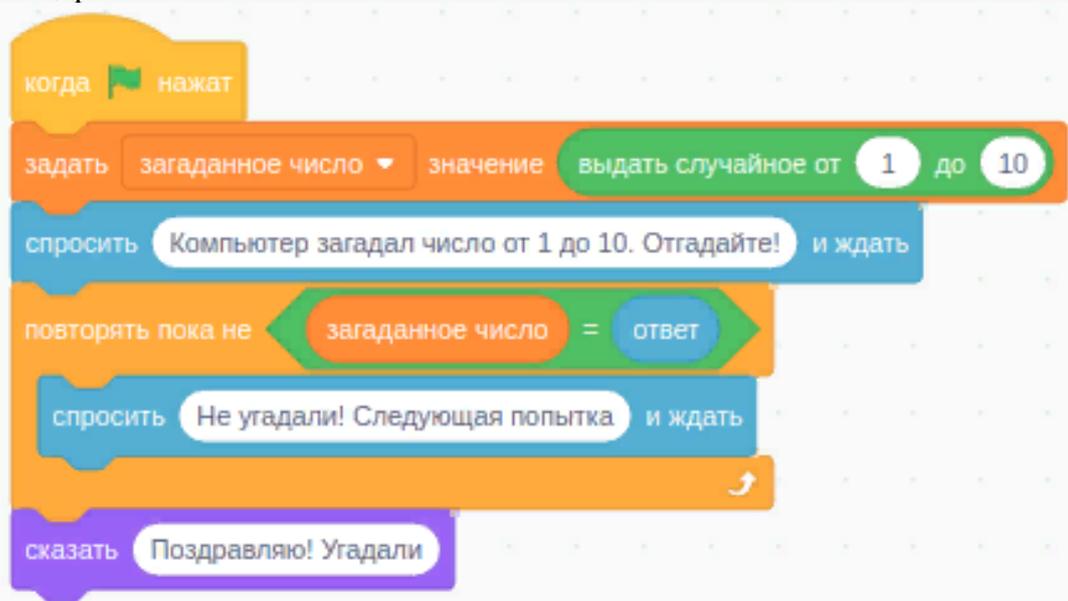
В Scratch допустимо сначала указать большее число, а затем меньшее. Например, "выдать случайное от 100 до 50". Команда и в этом случае будет правильно работать и генерировать любое число от 50 до 100.

Кроме того, можно указывать отрицательные и дробные числа. В последнем случае целая и дробная часть разделяются точкой, а не запятой.

Итак, чтобы компьютер загадал случайное число, нам нужна команда "выдать случайное от ... до ...". Это число надо будет где-то сохранить, чтобы потом сравнивать с тем, что вводит человек. Значит, нам понадобится переменная.

Человек должен отгадывать число до тех пор, пока не отгадает. Значит, нам понадобится цикл "повторять пока не ...". Что значит, пока не угадает? Это значит, что пока то, что ввел человек не совпадет с тем, что загадал компьютер, цикл должен продолжать крутиться и все просить и просить ввести очередное число.

В случае совпадения чисел, цикл должен прекращать свою работу, а на сцене должно появляться поздравление.



### Тема 3.6. Зеленый ящик – операторы. Использование арифметических и логических блоков вместе с блоками управления.

**Практика.** Создание программ с использованием операций сравнения данных. Создание программ с использованием арифметических данных и логических операций (1 час).

В программировании важное место занимает **условный оператор "если"**. С его помощью в программах организуется ветвление, поэтому его также называют **оператором ветвления**. Это значит, что при определенных условиях программа выполняется одним способом, а при отсутствии этих условий или наличии других условий программа выполняется другим способом, то есть идет по другому пути, другой ветке.

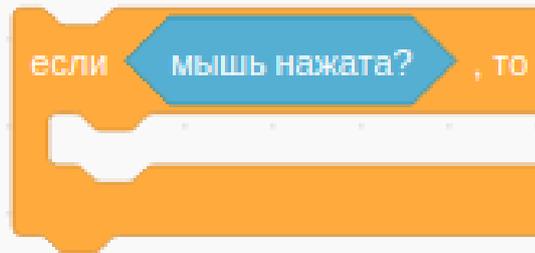
Допустим, в Scratch кот будет идти только в том случае, если нажата клавиша мыши. Если условие с мышкой не выполняется, кот никуда не пойдет. Чтобы реализовать такой сценарий, понадобится блок "если ... , то", который находится в оранжевом разделе "Управление".

В заголовок этого блока вставляется другой блок особого типа – **логическое выражение**. Такие блоки как бы что-либо спрашивают. И ответом на их вопрос может быть либо "да", либо "нет". Никаких "не знаю" или "надо подумать". Только да или нет.

Также можно представить, что логические выражения не спрашивают, а просто что-то утверждают. При этом то, что они утверждают, либо правда, либо ложь.

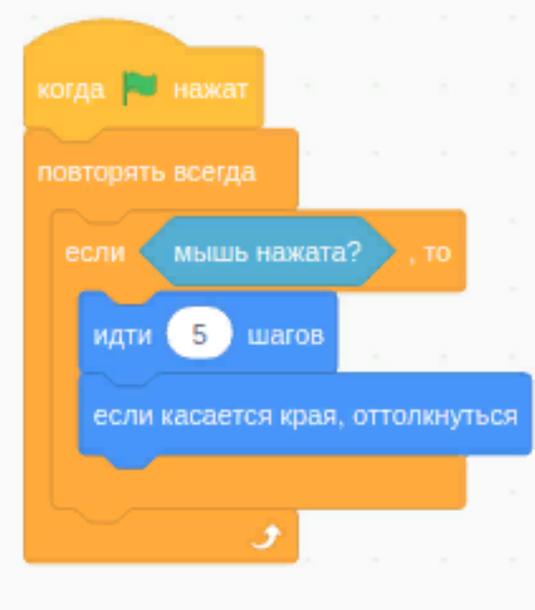
Блоки логических выражений не находятся в одном разделе. Их можно найти среди блоков как бирюзового раздела "Сенсоры", так и зеленого раздела "Операторы". В Scratch логические блоки имеют форму с острыми углами справа и слева.

В данном случае нам нужен блок "мышь нажата?", который мы вставляем в поле после слова "если" в оператор ветвления "если ... , то".



Теперь, если клавиша на мышке будет зажата, будут выполняться команды, которые обрамляет блок "если мышка нажата, то". Но если клавиша на мышке не будет зажата, то команды внутри блока условного оператора выполняться не будут.

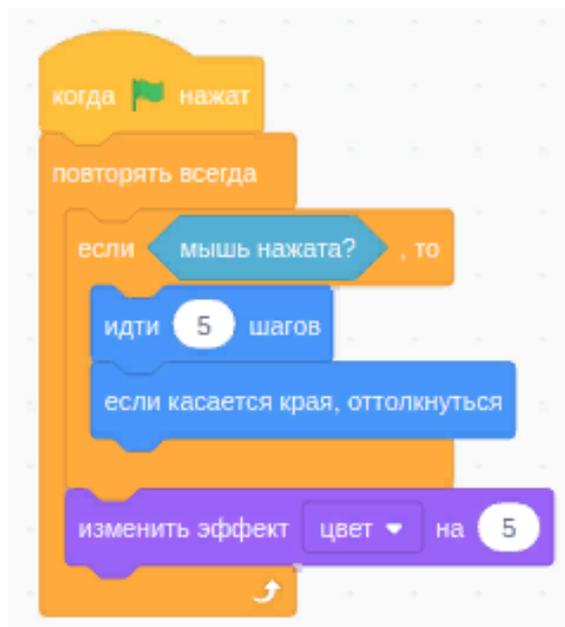
Сейчас внутри оператора ветвления, то есть в его теле, ничего нет. Поместим сюда команды движения, а сам блок "если" вставим в цикл "повторять всегда", чтобы программа работала всегда, пока мы ее сами не остановим.



В результате выполнения данного скрипта всегда будет проверяться, нажата ли какая-нибудь клавиша мышки. И если она нажата, кот будет перемещаться. Когда клавиша мыши будет отпускаться, спрайт будет останавливаться, потому что условие "мышка нажата?" станет неправдой.

А что будет, если мы поместим команды до или после всего блока "если". Влияет ли на них выполнение условия нажатия мышки? Абсолютно нет. Условие влияет только на те команды, которые находятся внутри "если". Команды за пределами "если" будут выполняться всегда.

Разместим команду "изменить эффект цвет на ... " после блока "если".

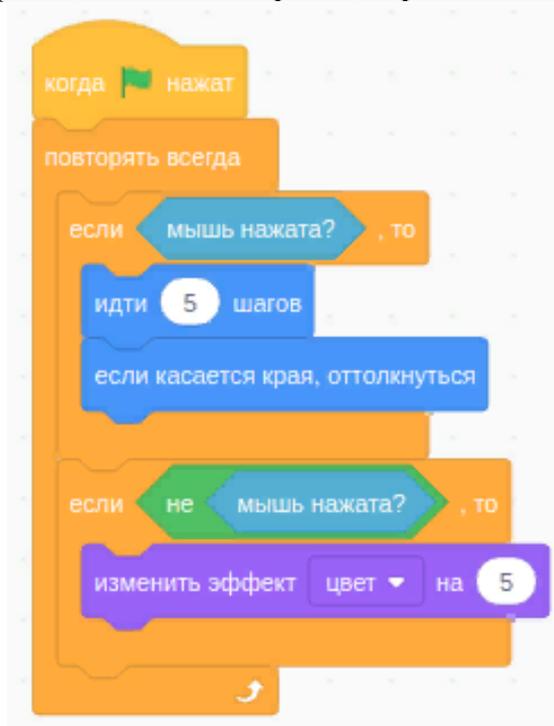


Теперь кот будет мигать разными цветами независимо от того, стоит он или движется. Блок изменения цвета находится за пределами условного оператора, а значит выполняется всегда.

Но допустим, мы хотим, чтобы кот изменял свой цвет только тогда, когда стоит. Когда же он движется, цвет не должен меняться.

Эту задачу можно решить двумя способами. Добавить второй блок "если", в котором проверять обратное условие, то есть то, что мышка не нажата. Или же, что более правильно, воспользоваться условным оператором с двумя ветками, то есть блоком "если ... иначе".

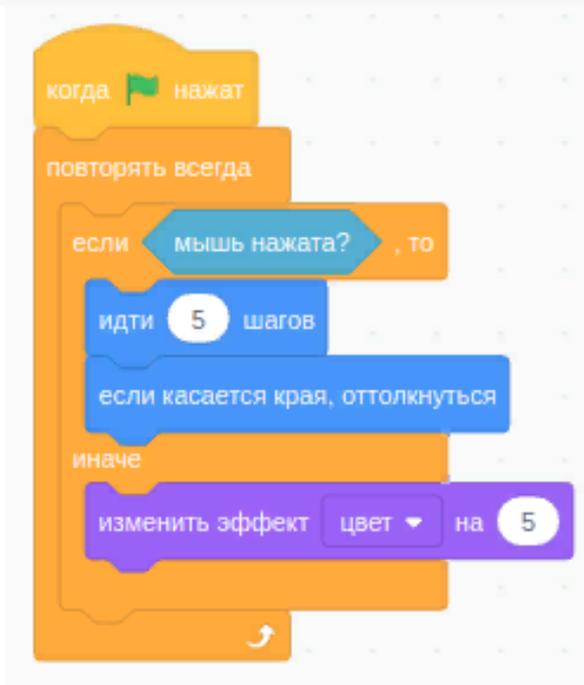
Рассмотрим сначала первый, не совсем верный, вариант:



Здесь мы используем новый для нас зеленый блок "не ...", в который вставляем "мышка нажата?". **Сложное логическое выражение** "не мышка нажата?" помещаем в заголовок "если".

Таким образом, второй "если" проверяет, что мышка не нажата. И если она не нажата, то будут выполняться команды тела второго "если". В данном случае это одна команда изменения цвета.

Рассмотрим второй вариант решения задачи, который делает то же самое что первый, но использует блок "если ... иначе":



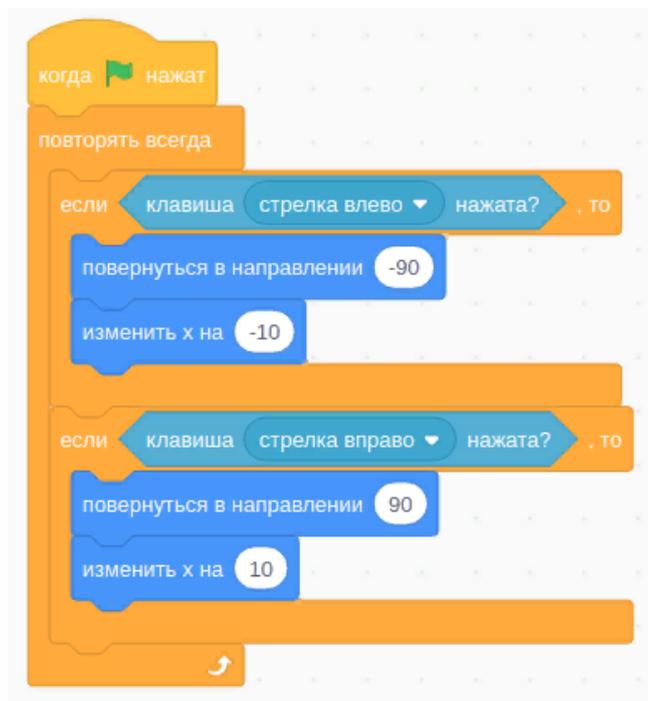
Этот вариант лучше первого не только потому, что содержит немного меньше команд. Важна логика. И здесь она более правильная.

Команды, вложенные в ветку "иначе" выполняются только тогда, когда условие при ветке "если" оказывается ложным. В данном случае если выражение "мышь нажата?" неправда, то будут выполняться команды внутри ветки "иначе".

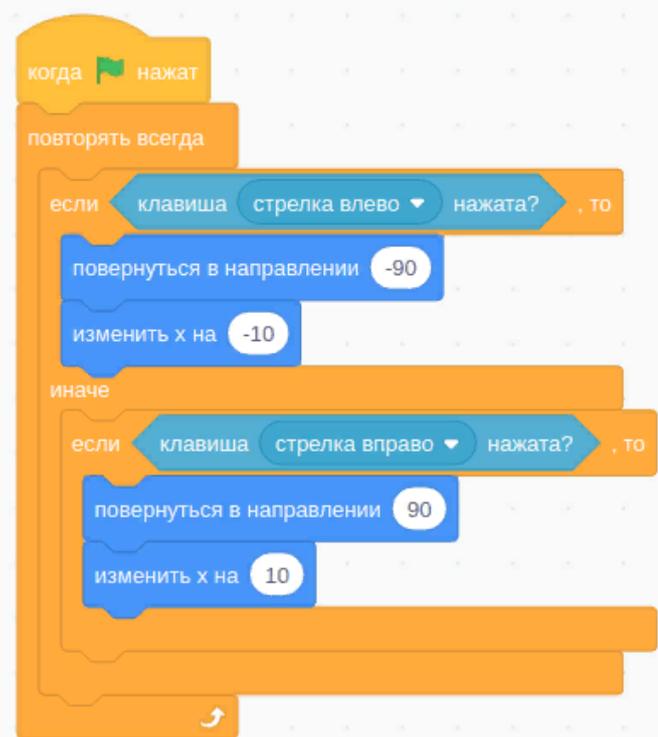
Программе не надо два раза проверять условие нажатия мыши, как в первом варианте. За один оборот цикла она проверяет его единожды, в заголовке "если ... иначе". И если условие правдиво, будут выполняться команды внутри ветки "если". Если условие оказывается ложным, то программа сразу без дополнительных проверок пойдет выполнять команды в ветке "иначе".

Выполняется либо то, либо другое. Никогда и то и другое. Не может быть, чтобы мышка одновременно была нажата и отпущена.

А как быть, если мы хотим, чтобы кот двигался налево, когда мы нажимаем на клавиатуре клавишу налево. И кот двигался бы направо, когда мы нажимаем клавишу направо. Можно решить задачу, используя два отдельных "если":



Однако мы также можем вложить второй "если" в ветку "иначе":

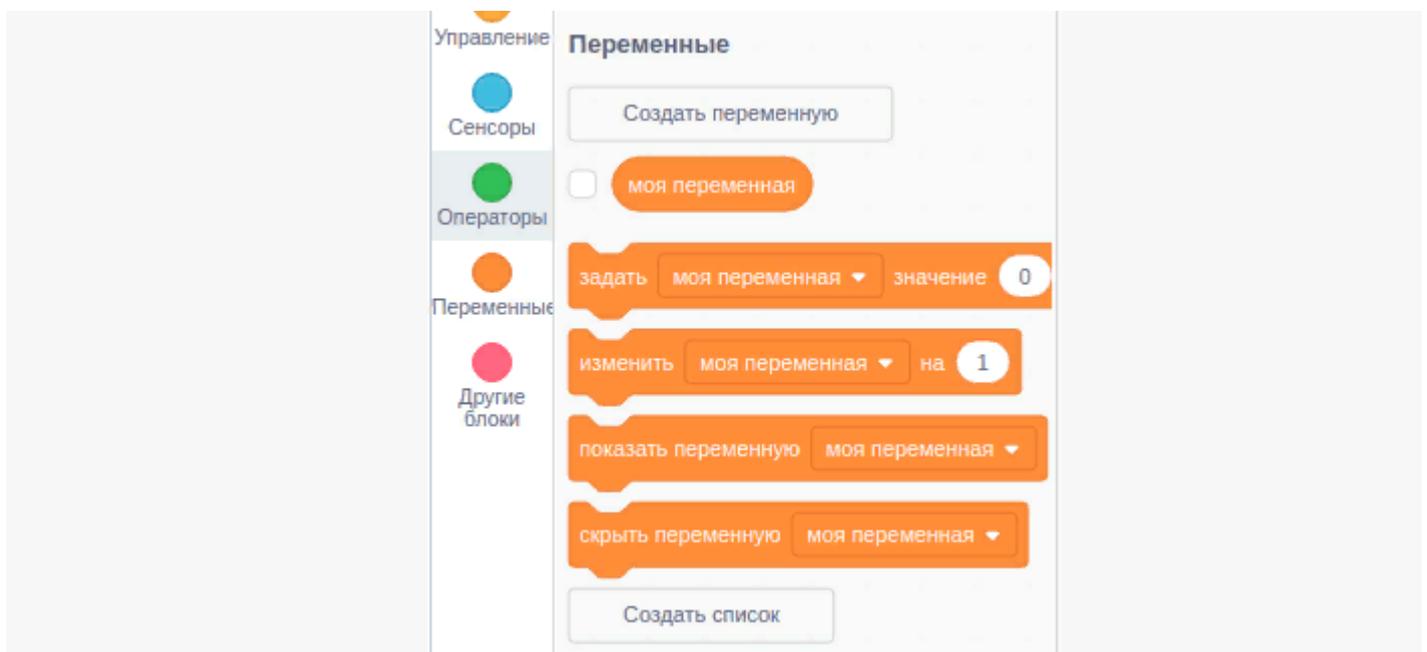


В данном случае в Scratch лучше работает первый вариант. Здесь если вы зажмете сразу две клавиши, кот никуда не будет идти, потому что шаг влево гасится шагом вправо. Во втором варианте, кот всегда будет предпочитать левую сторону, так как в случае зажатия двух клавиш до ветки "иначе" поток выполнения просто не доходит.

### Тема 3.7. События. Оранжевый ящик – переменные.

**Практика.** Разработка сценария Scratch-историй с несколькими событиями. Создание проектов с использованием глобальных и локальных переменных (1 час).

В Scratch ярко-оранжевый раздел "Переменные" содержит несколько блоков:



В программе может быть множество переменных. Поэтому есть кнопка "Создать переменную", при нажатии на которую открывается диалоговое окно, куда надо ввести имя переменной и выбрать, будет ли она доступна всем спрайтам или только текущему. Во втором случае созданная переменная не будет видна остальным спрайтам. Ей сможет пользоваться только один спрайт.

После создания переменную можно выбирать в выпадающих списках блоков раздела "Переменные".

Имя переменной может быть любым. Однако желательно, чтобы оно было описательное. Например, если переменная предназначена для хранения количества мячей, ее лучше так и назвать: "количество мячей" или просто "мячи".

На изображении выше мы видим, что сначала в Scratch есть только одна переменная под именем "моя переменная". Если около нее установить флажок, то ее имя и значение отобразятся на сцене.

Если в процессе выполнения программы требуется то показывать на сцене переменную, то скрывать ее, следует использовать блоки "показать переменную ..." и "скрыть переменную ...".

Команда "здать <переменной> значение ..." записывает в указанную переменную те или иные данные. Например, число 10.

Команда "изменить <переменную> на ..." в случае чисел изменяет значение переменной на указанную величину. То есть, если переменная содержала число 10, а мы дали команду изменить ее на 2, переменная будет хранить значение 12, а не 2.

Если же переменная содержала 10, а мы хотим присвоить ей значение 2, надо воспользоваться командой "здать <переменной> значение ...".

Рассмотрим как работать с переменными на примере. Пусть в нашем проекте по сцене ходят два спрайта – кот и краб. Ходить они будут в разных направлениях, но с одинаковой скоростью.

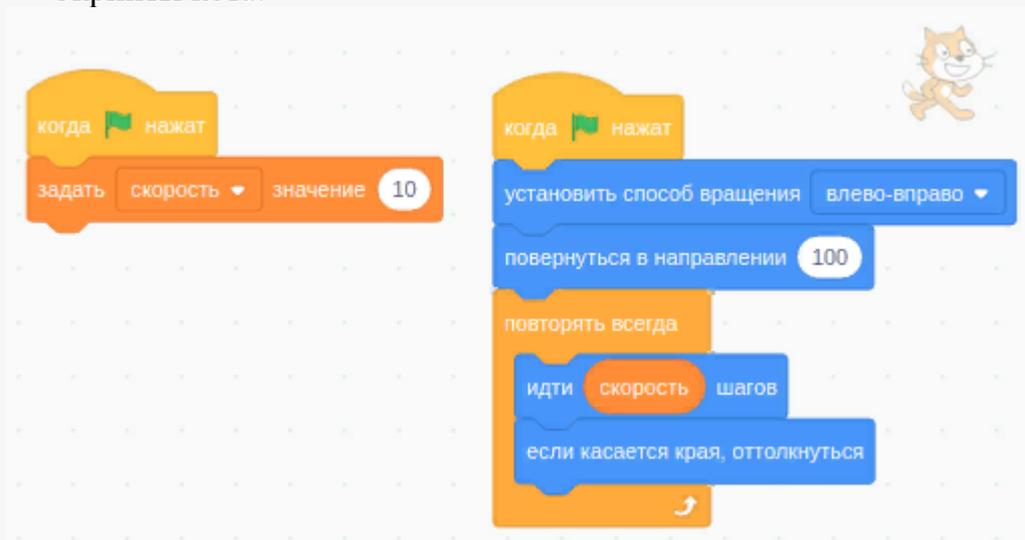
Мы пока не знаем точно, какую общую для спрайтов скорость хотим задать. Нам надо протестировать программу при разных скоростях. Если мы будем просто вписывать число в команду "идти ... шагов", придется потом изменять ее два раза, у каждого спрайта. Это не совсем удобно, особенно если бы спрайтов было много. Для кого-то можно забыть изменить величину.

Использование переменной решает эту проблему. Достаточно задать ей значение в одном месте, а потом вставить имя переменной во все места, где она используется.

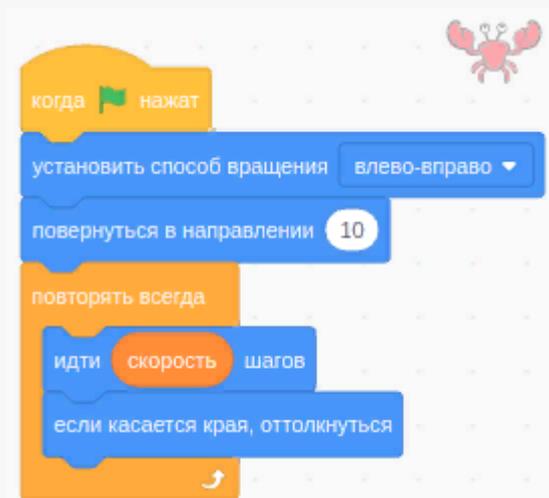
Создадим переменную под именем "скорость", доступную для всех спрайтов. Для этого надо нажать кнопку "Создать переменную" или переименовать существующую переменную, кликнув по ней правой кнопкой мыши и выбрав "Переименовать переменную".

Задать переменной значение можно у любого спрайта, второму значению будет доступно автоматически.

Скрипты кота:



Скрипт краба:

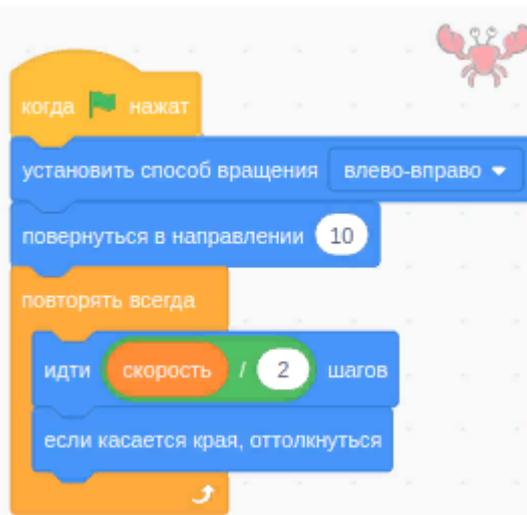


Теперь если мы захотим поменять количество шагов за один оборот цикла, достаточно сделать это в одном месте. Там, где задается значение переменной.

Когда выполняются команды "идти <скорость> шагов", то на место переменной "скорость" на самом деле подставляется ее значение. Например, число 10. В результате получается команда "идти 10 шагов", хотя мы этого не видим.

Более того, с помощью арифметических операций можно видоизменить нашу программу. Пусть краб ходит в два раза медленнее кота. То есть, если у кота скорость 10, у краба она должна быть 5. Если же у кота скорость 8, у краба она должна стать 4.

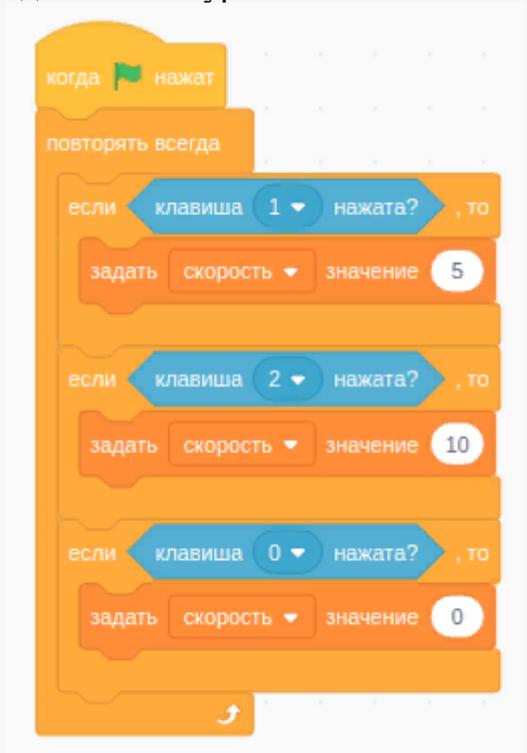
Для этого воспользуемся блоком с делением из зеленого раздела "Операции". Слева от знака деления поставим переменную "скорость", а справа – число 2.



Получается, что хотя скорость одного спрайта отличается от другого, скорости обоих взаимосвязаны через общую переменную.

Усложним нашу программу. Пусть значение переменной "скорость" задает пользователь. Если он нажмет на клавиатуре число 1, то скорость будет иметь значение 5, а если нажмет на клавиатуре число 2, скорость будет иметь значение 10. Если же человек нажмет 0, скорость должна установиться также в 0, то есть спрайты попросту останутся.

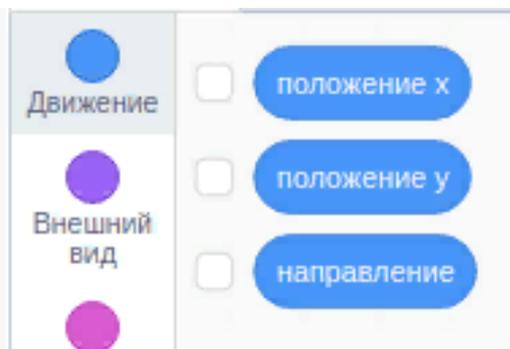
Тут нам потребуются условные операторы, а также цикл "повторять всегда", чтобы программа всегда ожидала ввода с клавиатуры.



Этот скрипт можно добавить любому спрайту.

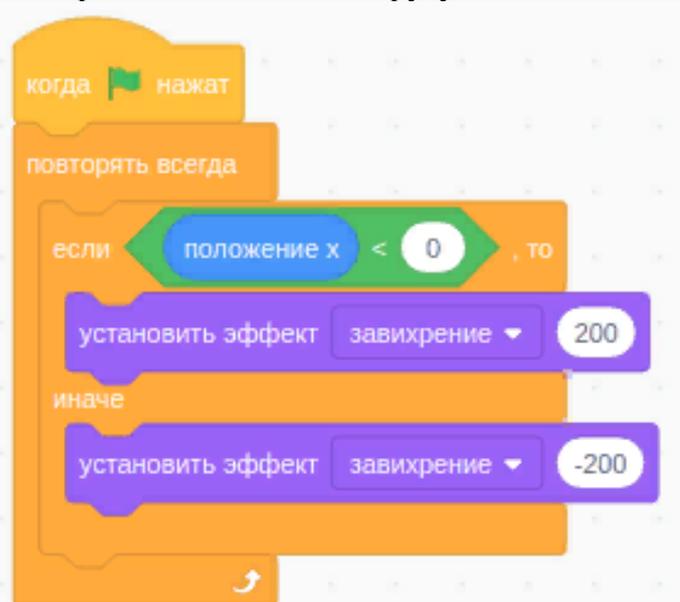
Таким образом значение переменной может меняться в процессе выполнения программы в зависимости от тех или иных условий.

Обратите внимание на форму блока, который содержит только имя переменной. Это прямоугольник со скругленными углами. Блоки подобной формы есть во многих разделах вкладки "Код". Например, в секции "Движение" это блоки "положение x", "положение y", "направление":



Подобные блоки можно назвать встроенными в Scratch переменными. В них записываются значения, которые извлекаются из каких-нибудь объектов. Так в переменную "положение x" записывается значение текущей координаты  $x$  спрайта. Мы можем извлекать это значение и использовать его в программе.

Допустим, наш кот, находясь на левой половине сцены, будет скручиваться в одну сторону, а когда находится на правой половине – в другую.



Здесь мы используем переменную "положение x", чтобы проверить, находится кот справа или слева. Вспомним, что 0 – это середина холста. Налево идут значения меньше нуля, направо – больше.

### Тема 3.8. Списки.

**Практика.** Создание программ-тестов по принципу сравнения данных из нескольких списков (1 час).

В Scratch в категории блоков кода "Переменные" есть кнопка "Создать список". В отличие от обычной переменной, список может хранить не одно значение, а много.

Например, мы создаем обычную переменную "число" и связываем ее с хранящимся в памяти компьютера числом 5. Но если мы создаем список "числа", то можем связать его с несколькими числами, расположенными в памяти друг за другом. Пусть это будут числа 3, 8, 1, 5.

У элементов принадлежащих одному списку есть порядковые номера. Так в нашем примере число 3 имеет номер 1, число 8 – номер 2, число 1 – номер 3, число 5 – номер 4. Таким образом, используя номера элементов, мы можем извлекать значения элементов. Например, команда "элемент 2 в числа" вернет нам 8.

В программировании часто используется понятие массива. Списки и массивы – близкие понятия. Однако обычно их различают следующим образом. В массиве могут содержаться

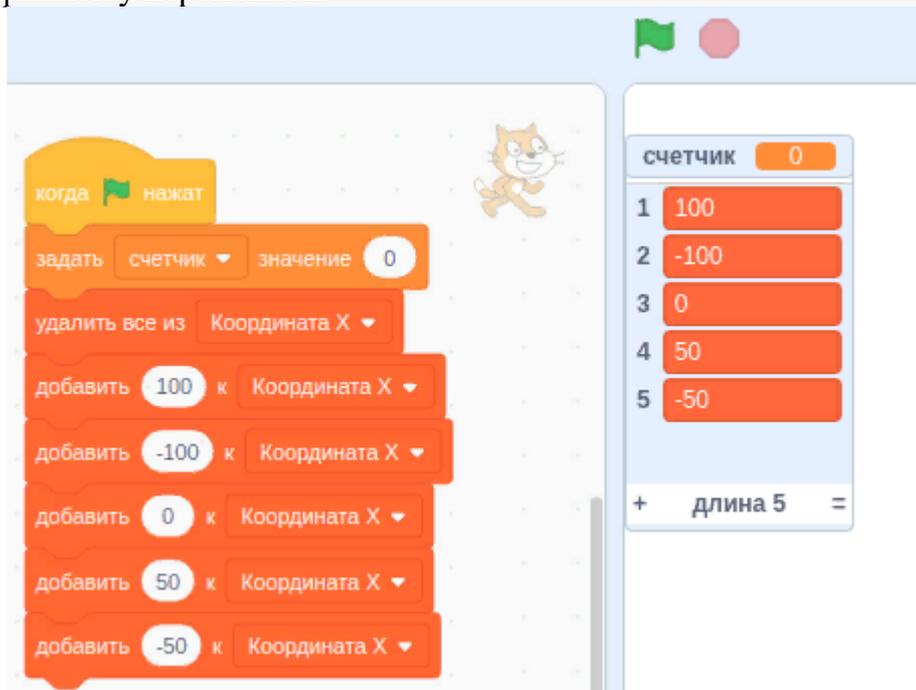
только данные, принадлежащие одинаковому типу. Например, только слова или только числа. Элементами одного и того же массива не могут быть и числа и слова.

А вот в случае со списком такое может быть. Например, список [3, А, 4, 5.2, Кот] содержит целые числа, дробное число, букву и слово.

Представьте, что вам надо изменить нескольких чисел. Если каждое число будет храниться в своей переменной, то придется помнить имена всех переменных и обращаться к каждой индивидуально.

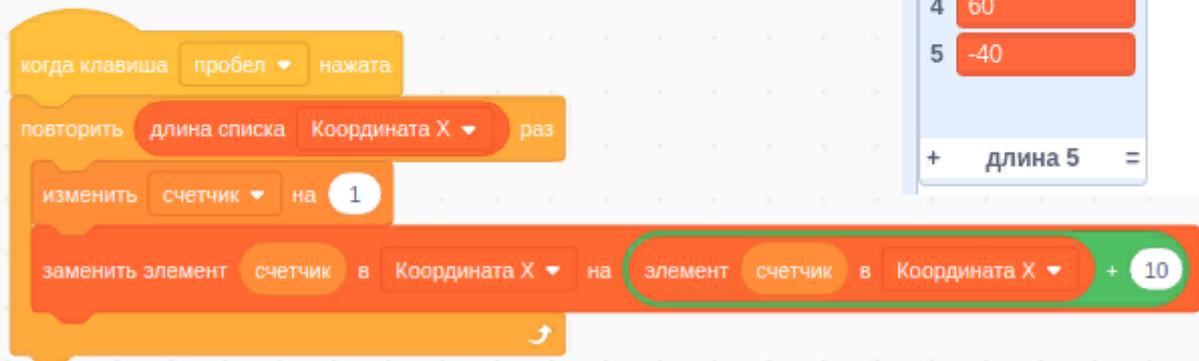
Если же ряд чисел хранится в одном списке, у нас имеется только одна переменная — имя списка. Извлекать же отдельные числа мы можем по их порядковым номерам, то есть индексам.

Рассмотрим пример. Заполним список "Координата X" числами и отобразим список на сцене, установив флажок у переменной.



Также у нас здесь есть обычная переменная "счетчик".

Теперь представим, что при нажатии пробела все координаты должны увеличиться на 10. Это значит, надо перебрать все элементы списка и изменить их. Такое обычно выполняют в цикле.



Справа на изображении показан список после нажатия пробела.

Сколько оборотов делает цикл "повторить ... раз"? Столько, какова длина списка. А чему равна длина списка? В списке 5 элементов, значит, длина равна пяти.

Что происходит с переменной "счетчик" на каждом обороте цикла? Она увеличивается на единицу. Если исходное значение счетчика 0, то на первом обороте цикла он получит значение 1, на втором 2, на третьем 3, на четвертом 4, на пятом 5.

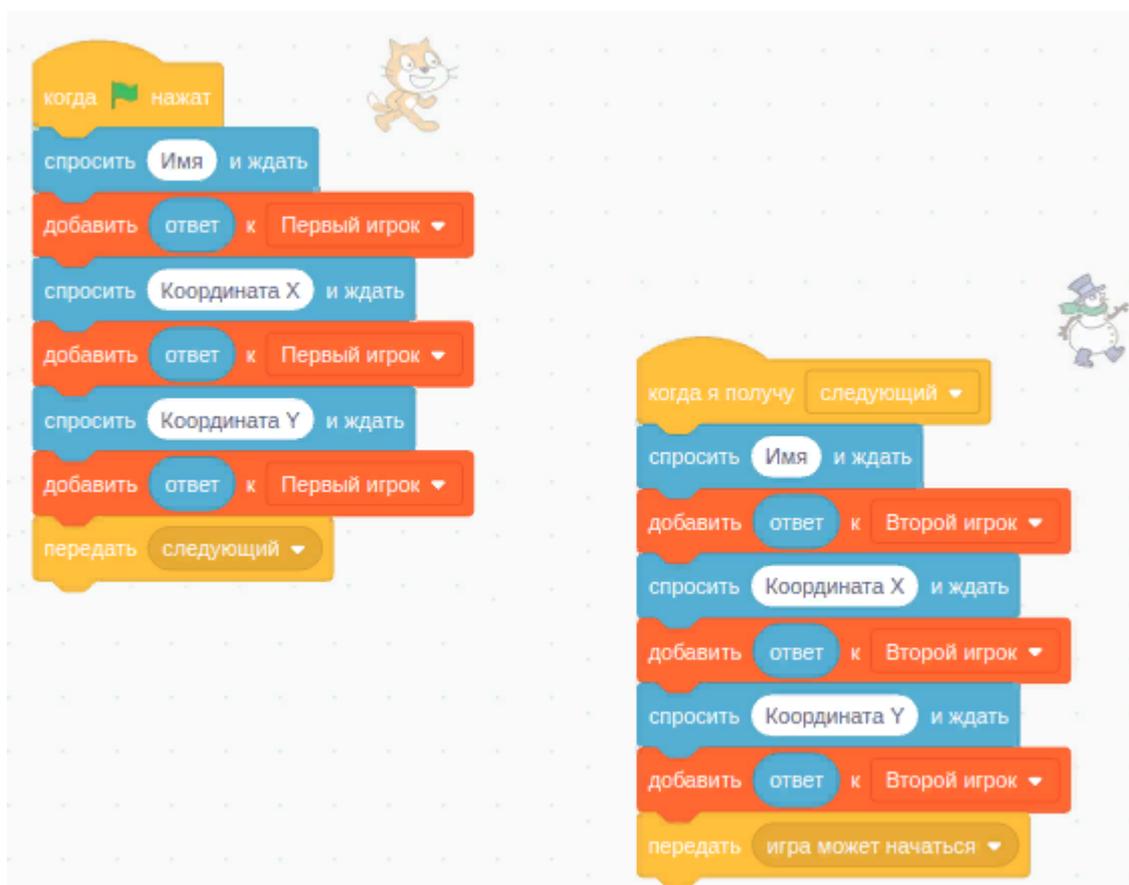
Что происходит внутри зеленого блока, который суммирует два числа? Сначала из списка "Координата X" извлекается значение элемента, номер которого равен счетчику. Например, при первом обороте цикла счетчик равен 1, значит из списка будет извлечено число 100.

К этому числу добавляется 10. Получиться 110. После этого будет выполняться команда "заменить элемент ... в ... на ...", которая заменит число 100 в позиции, на которую указывает счетчик, на число 110.

Рассмотрим следующий пример. Есть два спрайта-игрока, каждый из которых должен получить имя и начальные координаты. Их задает пользователь. В данном случае имеет смысл хранить данные о каждом игроке в отдельных списках – "Первый игрок" и "Второй игрок".

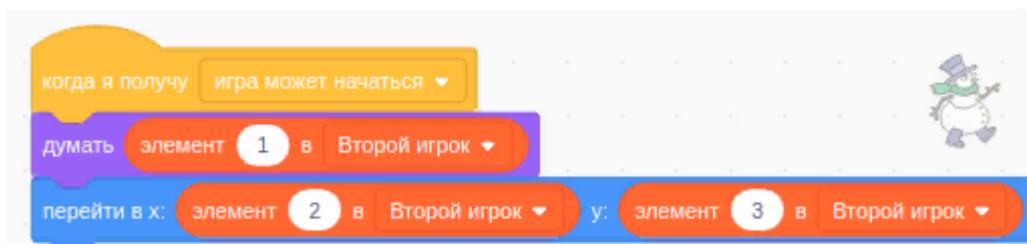
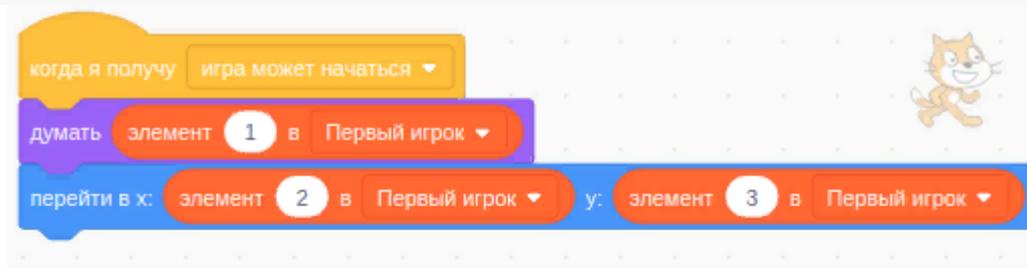
Договоримся, что первым элементом каждого списка будет имя игрока, вторым – координата X, а третьим – координата Y.

Заполняющие списки скрипты спрайтов.



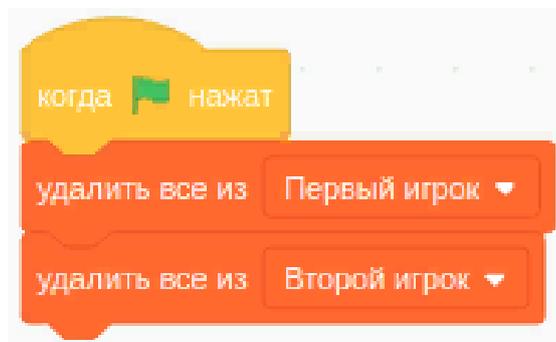
Обратите внимание, что добавление элементов происходит в конец списка. Поэтому имя будет иметь порядковый номер 1, а координата Y – номер 3.

В процессе игры у спрайтов могут выполняться сценарии, извлекающие данные из списков.



Можно заподозрить, что мы сделали много бессмысленных вещей. Зачем добавлять данные в список, если можно сразу использовать "ответ" для команд "думать ..." или перемещения? Однако если данные будут требоваться неоднократно, то надо сохранять их. Причем мы обошлись только двумя переменными типа список – "Первый игрок" и "Второй игрок", а не шестью, как если бы пользовались обычными переменными.

Чтобы приведенная выше программа правильно работала во второй и последующий разы, надо удалять элементы, помещенные в списки ранее.

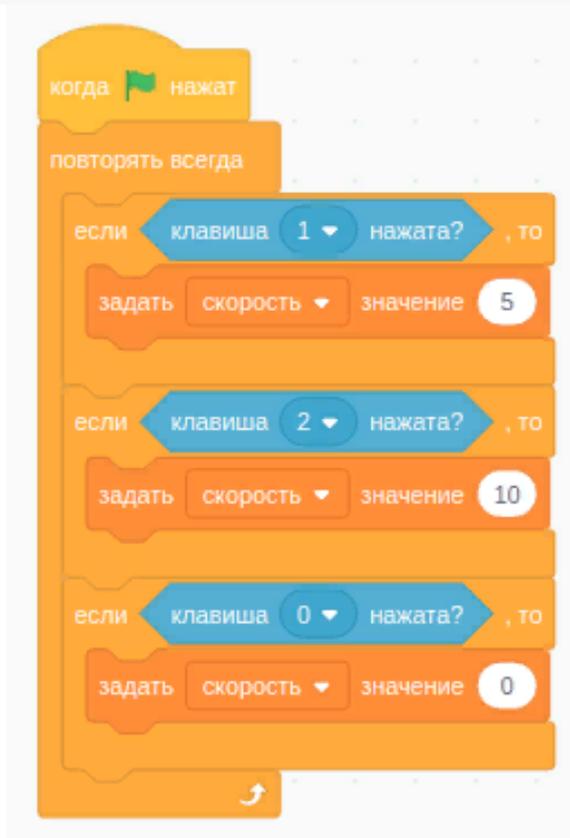


### Тема 3.9. Голубой ящик – сенсоры. Ввод-вывод данных.

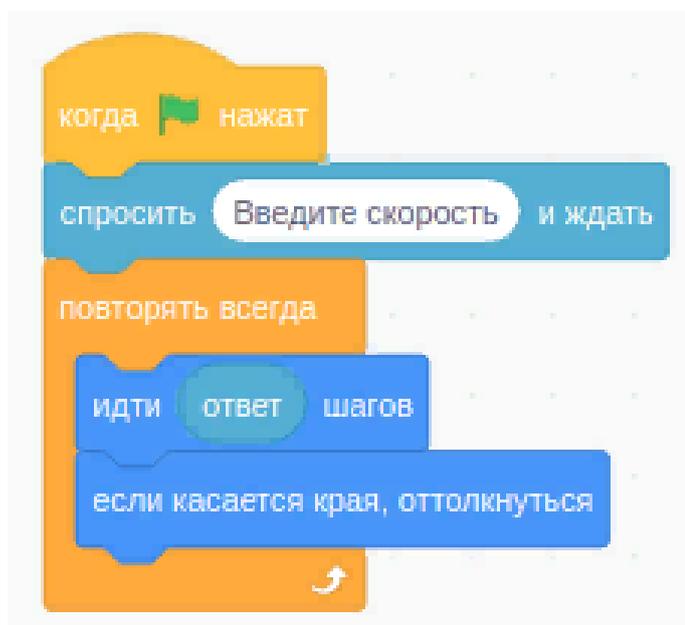
**Практика.** Создание проектов с использованием значений сенсоров и команды спросить. Создание программ для обработки данных пользователя с выводом на экран конечного результата (1 час).

Часто в программах требуется, чтобы человек ввел какое-нибудь число или строку или выполнил любое другое действие. Это можно назвать вводом данных в программу. Мы уже с этим сталкивались в скриптах, ожидающих от пользователя нажатия клавиш клавиатуры или мыши.

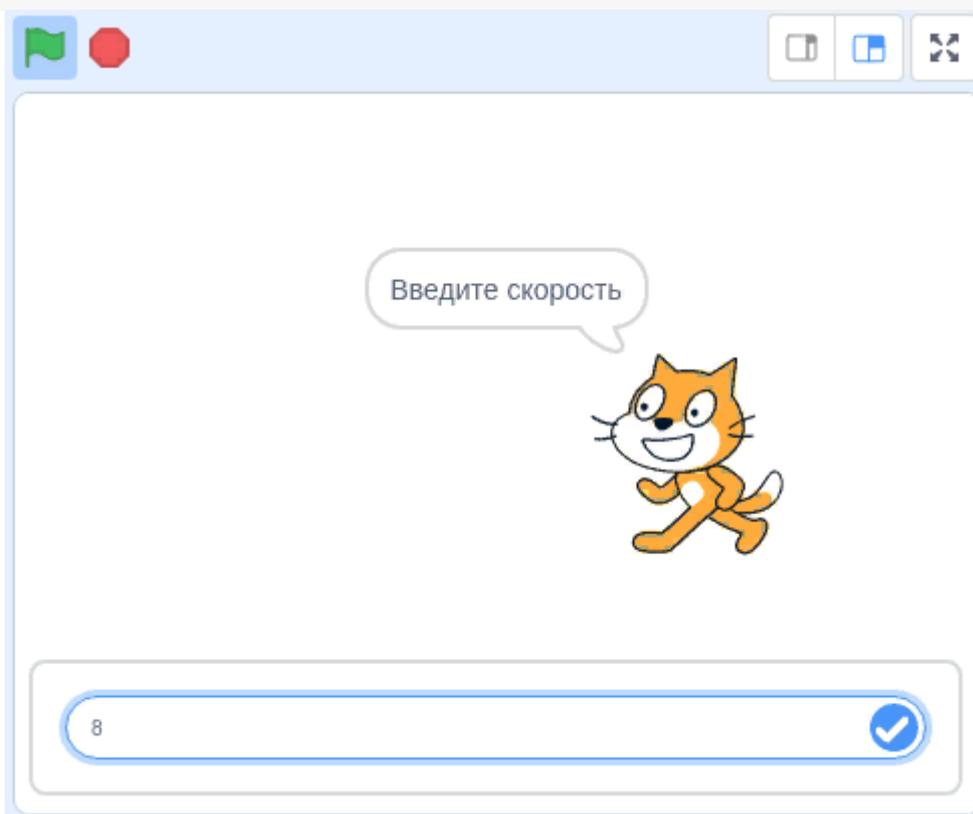
Также программы могут просить человека ввести какое-нибудь число или строку. После этого программа связывает полученные данные с именем какой-нибудь переменной, чтобы сохранить данные и потом их использовать.



Если мы хотим предоставить человеку самому определять скорость, а не ограничивать ее только тремя вариантами, код станет проще:



Когда выполняется команда "спросить ... и ждать", на сцене появляется поле, куда пользователь должен что-то ввести и нажать Enter на клавиатуре. Также он может ничего не вводить, а просто нажать Enter.



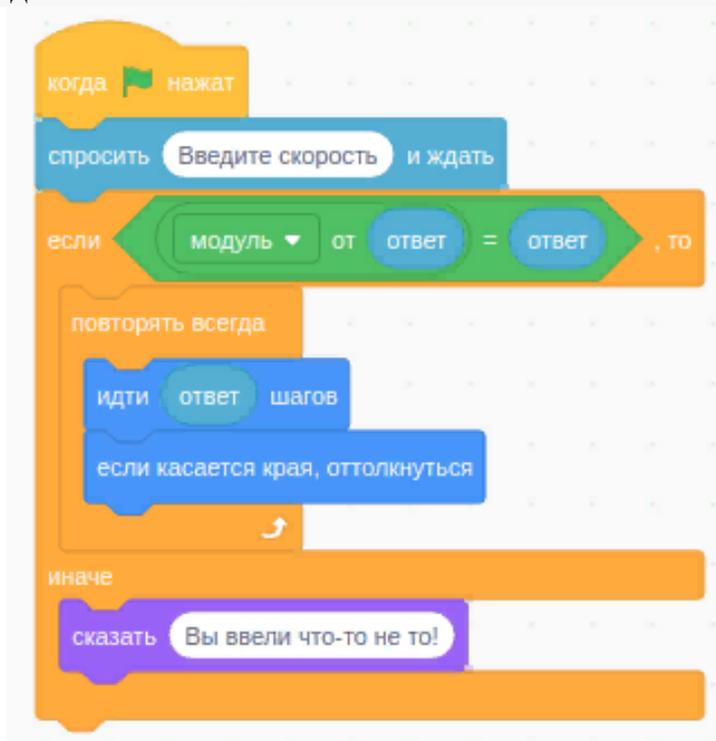
В Scratch команда "спросить ... и ждать" блокирует свой скрипт, но не другие. Блокирует она свой скрипт до тех пор, пока не будет нажат Enter в поле ввода. После этого то, что было введено в поле, присваивается встроенной переменной "ответ", которая далее используется в скрипте.

Что будет, если человек введет не число, а слова? Переменная "ответ" будет содержать введенную строку текста. Но поскольку в команду "идти ... шагов" бессмысленно подставлять строку, скорость кота не изменится.

Допустим, мы хотим, чтобы в случае ввода текста, скрипт не пыталась его подставлять в команду "идти ... шагов". Вместо этого на сцене появлялось бы сообщение о неправильном вводе.

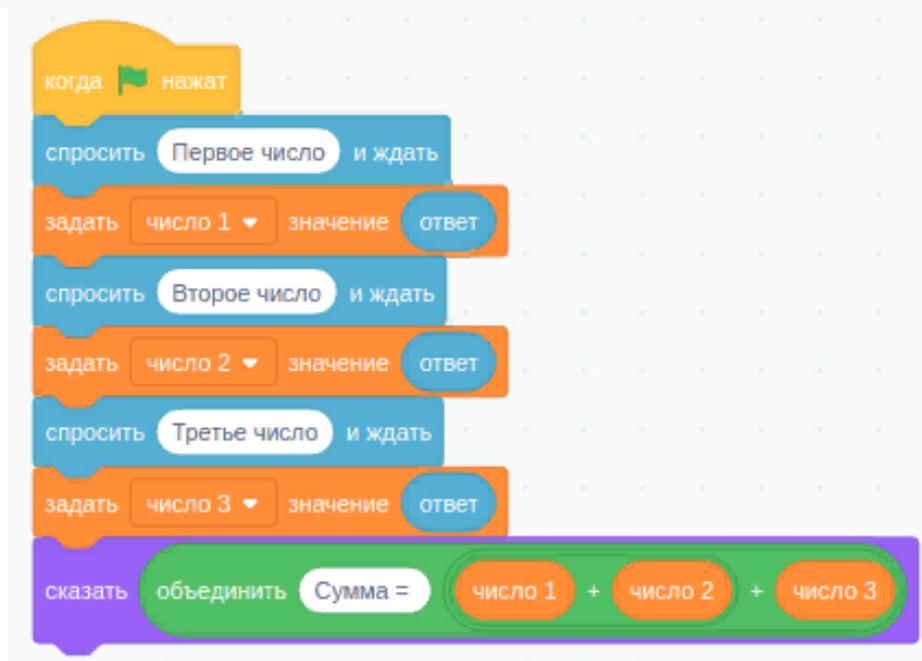
В Scratch нет команды проверки, введено число или строка. Однако можно придумать способ это узнать. В Scratch если выполнить над строкой арифметическую операцию, то результатом будет число 0. Например, если взять из положительного числа модуль, то вернется само число. Если же попытаться взять модуль из строки, что бессмысленно, будет получен 0.

Таким образом, если модуль ответа не совпадает с самим ответом, значит было введено что-то не то: либо строка, либо отрицательное число. Если же модуль ответа совпадает с самим ответом, значит было введено положительное число.

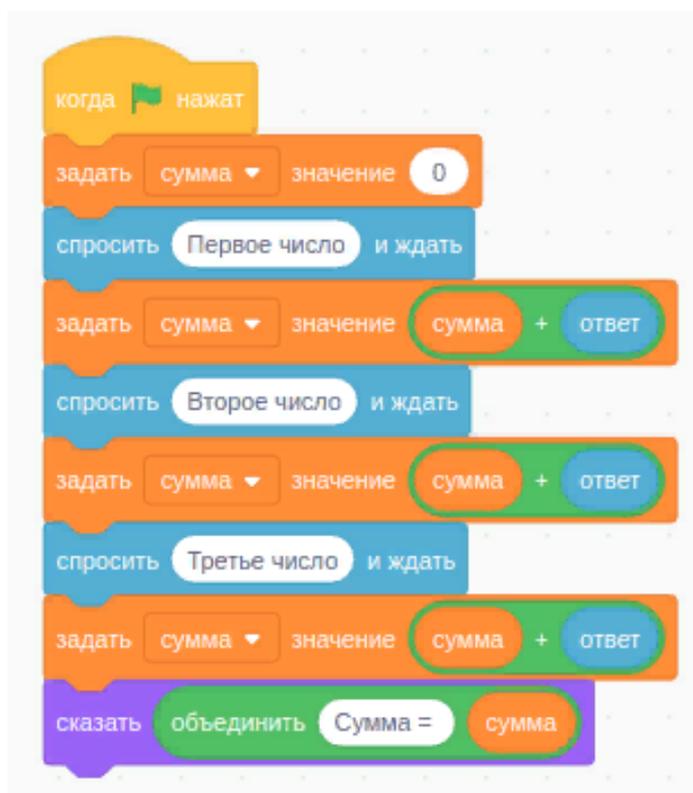


Рассмотрим второй пример использования команды "спросить ... и ждать". Пусть наша программа складывает три числа, которые вводит пользователь. Здесь уже не обойтись одной встроенной переменной "ответ". Придется создавать свои. При этом задачу можно решить разными способами.

Во первых, каждое число можно присваивать отдельной переменной, а потом выводить их сумму.



Второй вариант – создать одну переменную, в которой постепенно накапливать сумму.



Когда выполняется команда "задать <сумма> значение <сумма + ответ>", сначала выполняется подвыражение <сумма + ответ>. И только после этого результат этого подвыражения записывается в переменную "сумма".

Например, переменная "сумма" содержит число 3. Пользователь вводит число 5. Когда выполняется выражение <сумма + ответ>, из переменной "сумма" извлекается число 3 и к нему добавляется 5. После этого результат 8 записывается в переменную "сумма". Старое значение 3 переменной при этом затирается, то есть теряется.

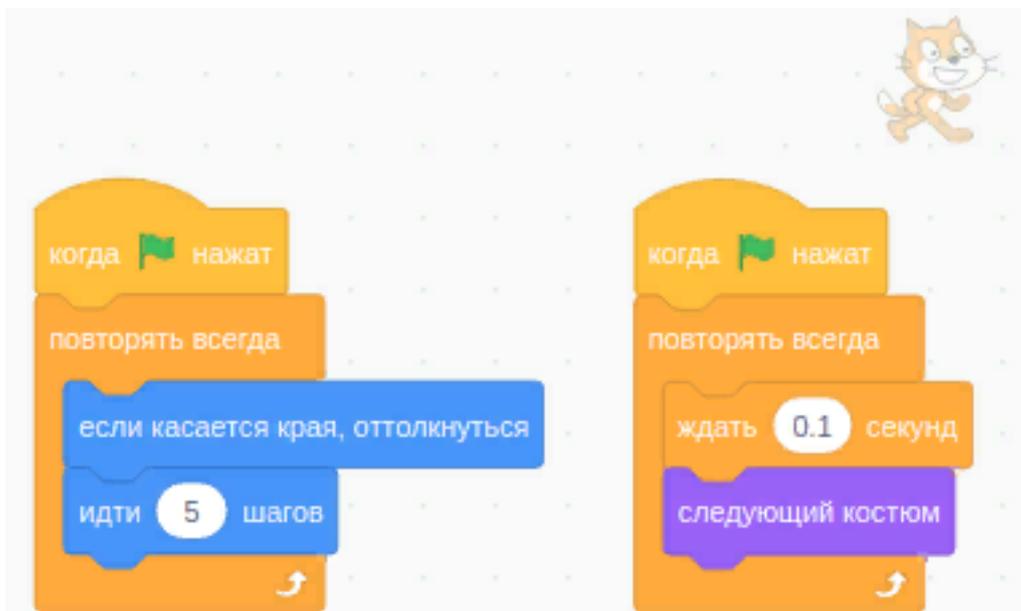
#### Тема 4.1. Последовательность и параллельность выполнения скриптов

**Практика.** Создание Scratch-историй с одновременной и попеременной работой нескольких исполнителей (1 час).

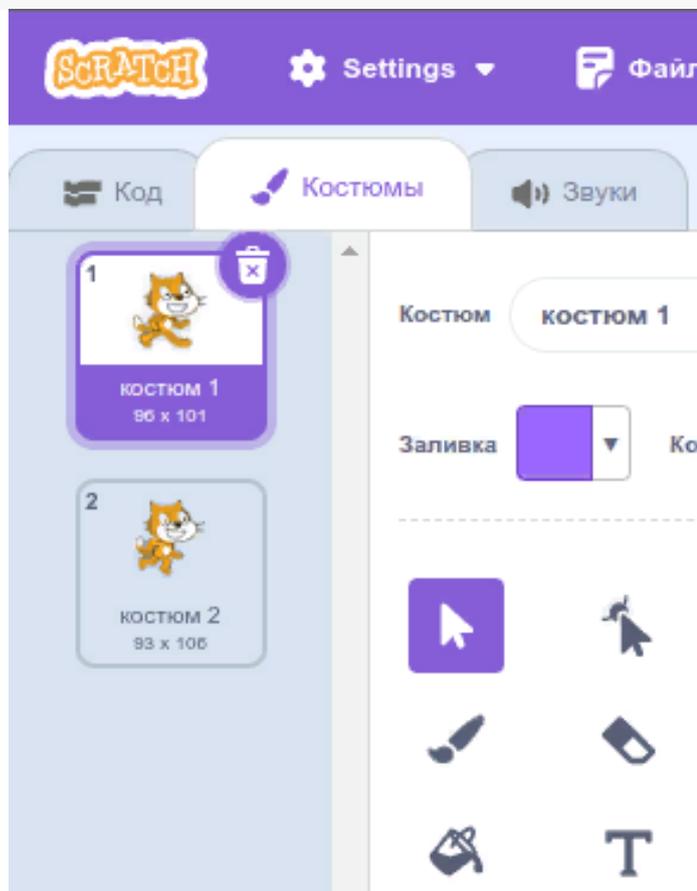
В программировании отдельные части большой программы могут выполняться либо по очереди друг за другом, то есть последовательно, либо одновременно друг с другом, то есть параллельно. Во втором случае говорят о многопоточности, то есть каждый скрипт работает в своем потоке, а разные потоки текут во времени одновременно.

Когда для каждого спрайта одного проекта вы добавляете свою конструкцию блоков, которая начинается с команды "когда флажок нажат", то как только игра запускается, оба героя начинают параллельно независимо друг от друга выполнять свои команды. Это пример одновременного выполнения скриптов.

Более того, один и тот же спрайт может содержать два независимых скрипта, каждый из которых будет выполняться в своем потоке. Рассмотрим пример. Пусть кот ходит по сцене и при этом меняет свой костюм.



В Scratch многие спрайты на самом деле состоят не из одного изображения, а из нескольких. Это можно увидеть на вкладке "Костюмы".



Поэтому когда выполняется команда "следующий костюм", происходит переключение на следующее изображение спрайта. Если костюмов всего два, то они будут чередоваться.

В приведенном выше примере блоков кода оба скрипта будут выполняться одновременно. Мы увидим, что кот не только перемещается, но еще двигает ногами. Этот эффект возникает от того, что изображения-костюмы быстро меняются.

Если отсоединить блок "когда флажок нажат" от первого скрипта, то при запуске игры кот будет стоять на месте, будут двигаться только его ноги.

Если же мы отсоединим блок "когда флажок нажат" только от второго скрипта, то кот будет перемещаться, но его ноги не будут двигаться.

Таким образом, каждый скрипт вносит свой вклад и работает независимо от другого, но одновременно с ним.

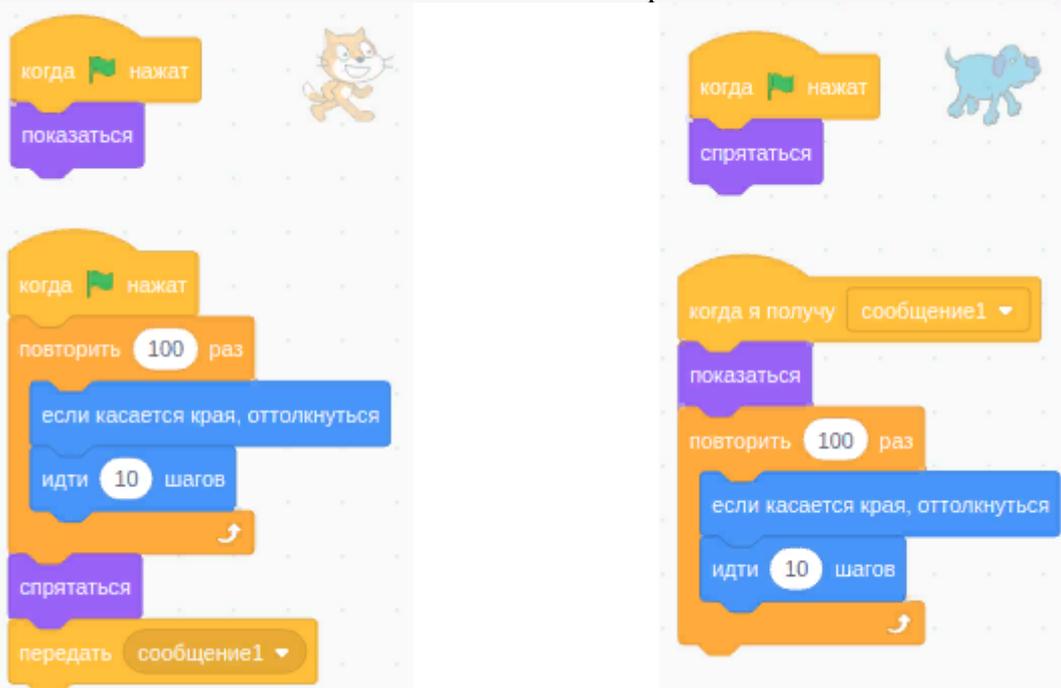
Вы спросите, почему нельзя было вставить команду "следующий костюм" в цикл первого скрипта и вовсе отказаться от второго? На самом деле можно. Однако в этом случае кот начнет менять костюмы так часто, что у вас зарябит в глазах. Тогда почему бы сюда же не добавить команду "ждать ... секунд"? Потому что в этом случае, кот будет останавливаться, и плавного движения не получится. Движение будет урывистым, перед каждой сменой костюма будет возникать задержка.

Теперь поговорим о последовательном выполнении скриптов. На самом деле такое поведение встречается чаще хотя бы потому, что команды в одном скрипте выполняются последовательно. А чтобы друг за другом выполнялись разные скрипты, один из них должен вызывать другой, то есть передавать своего рода сообщение.

Также бывает, что скрипт начинает выполняться только при определенных условиях. Например, когда нажимается одна из клавиш на клавиатуре. При этом работа других скриптов может как останавливаться, так и не останавливаться. Во втором случае это уже не может считаться последовательным исполнением.

Давайте рассмотрим вариант передачи сообщения, когда один скрипт вызывает другой. Пусть у нас будет два спрайта – кот и собака. Сначала кот ходит по сцене. И только когда он останавливается и исчезает, на сцене появляется собака и начинает ходить.

Поскольку при запуске игры один из спрайтов должен быть видимым, а другой невидимым, воспользуемся командами "показаться" и "спрятаться". Вынесем их в отдельные скрипты, чтобы они не мешались в основной логике игры.



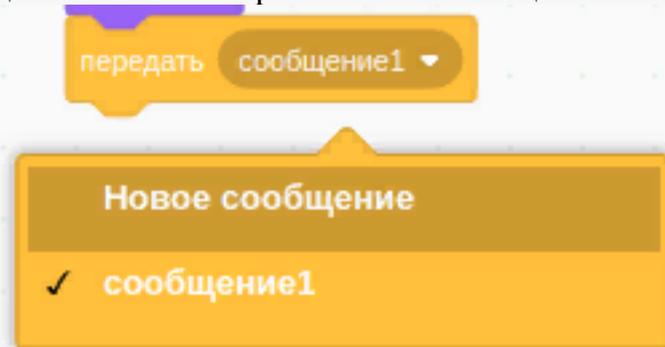
Теперь когда кот выполнит свою порцию шагов, с помощью блока "передать сообщение1" он пошлет сообщение, которое может перехватить любой объект программы. В данном случае сообщение будет ждать только собака, потому что только ей мы добавили команду "когда я получу сообщение1".

И как только она получит сообщение, она покажется и начнет ходить.

В итоге получится, что скрипты, отвечающие за ходьбу кота и собаки, будут выполняться последовательно, один за другим.

Команда "передать сообщение" и "когда я получу сообщение" могут передавать любое сообщение, а не только "сообщение1". В проекте могут циркулировать десятки различных сообщений, которые одни объекты посылают, а другие ждут.

Чтобы добавить новое сообщение, надо кликнуть по маленькому треугольнику в этих командах и в раскрывающемся списке выбрать "Новое сообщение".



После этого на экране появится диалоговое окно, куда вводится сообщение. Например, "кот вызывает собаку".

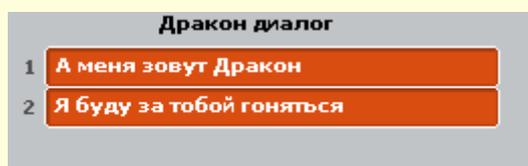
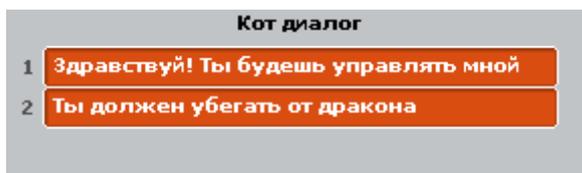
Когда сообщение создано, оно становится доступным через раскрывающийся список команд "передать ..." и "когда я получу ...", где вместо точек может стоять любое сообщение.

При этом надо понимать, что если, например, кот передает сообщение "кот вызывает собаку", то собака должна получать именно это сообщение, а не какое-либо другое. Иначе она не будет реагировать.

#### Тема 4.2. Взаимодействие между спрайтами. Управление через обмен сообщениями

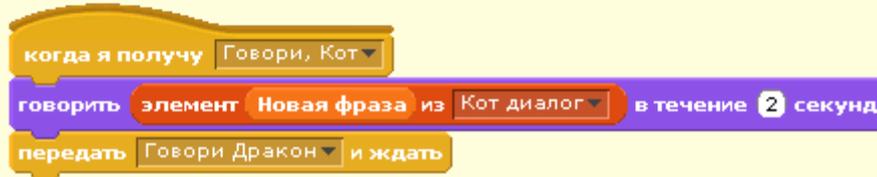
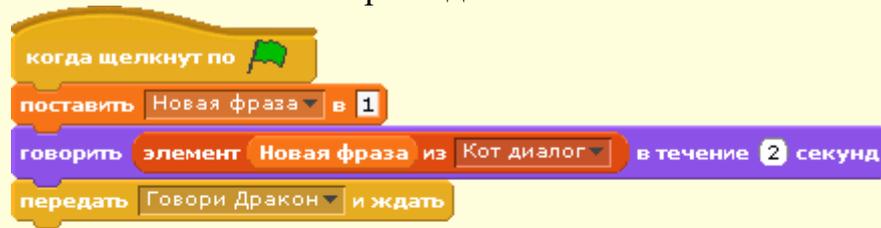
**Практика.** Создание Scratch-историй с взаимодействием нескольких исполнителей и неподвижных объектов. Создание Scratch-историй с взаимодействием нескольких исполнителей (1 час).

Создадим проект, в котором осуществим диалог между двумя спрайтами. Пусть у нас два героя. Создадим диалог между ними с использованием ящика переменные. Для организации диалога аналогично пояснениям к слайдам презентации создадим список переменных (один список – это фразы для Котенка, а второй – для Дракона) и назовем их соответственно: Кот диалог и Дракон диалог. Оба списка сделаем локальными, т.е. доступными только для конкретного Спрайта.

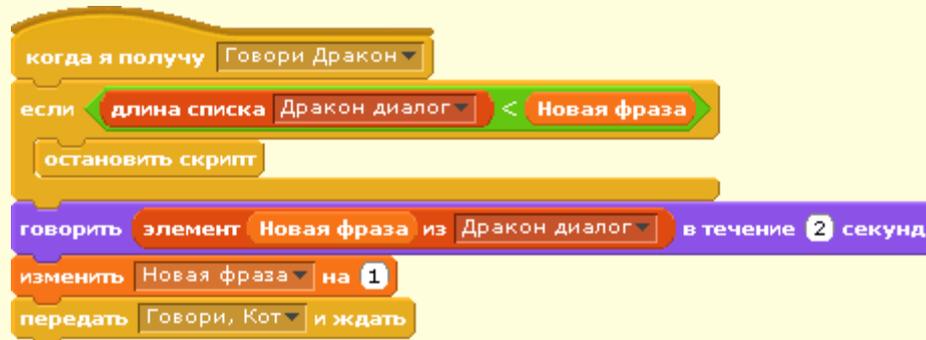


Теперь создадим глобальную переменную Новая фраза (этой переменной могут пользоваться и Кот и Дракон) и запишем скрипты для наших героев, которые заключаются в следующем: как только пользователь запустил проект, щелкнув по зеленому флажку, переменной Новая фраза присваивается значение 1 и Кот говорит первую фразу из списка, который описывает его слова. Затем Дракону передается сообщение, чтобы он начинал говорить. В то же время аналогичное сообщение будет передаваться и Коту от Дракона. Такой же скрипт прописывается и для Дракона.

### Скрипт для Кота:



### Скрипт для Дракона:



## Тема 5.1. Виды компьютерных игр. Алгоритмическая разработка листинга программы.

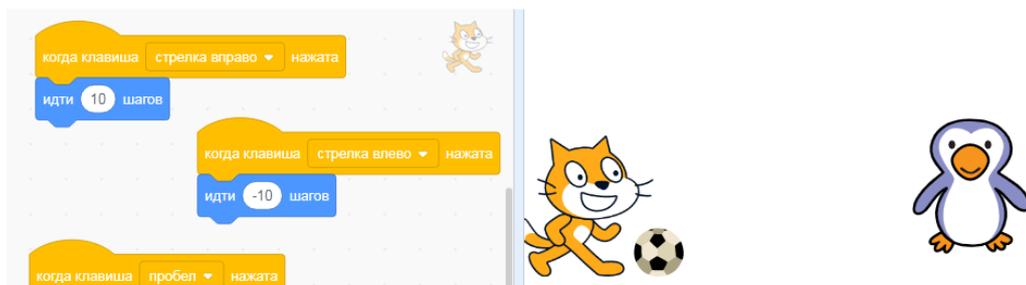
**Практика.** Алгоритмическая разработка проекта, запись на естественном языке событий и точек взаимодействия героев будущей игры (1 час).

### Игра «Футбол»

**Задача** – кот футболист пинает мяч, а пингвин его ловит и говорит: «Поймал!». Добавь фон Футбольное поле, спрайтов – Кот, Мяч и Пингвин.

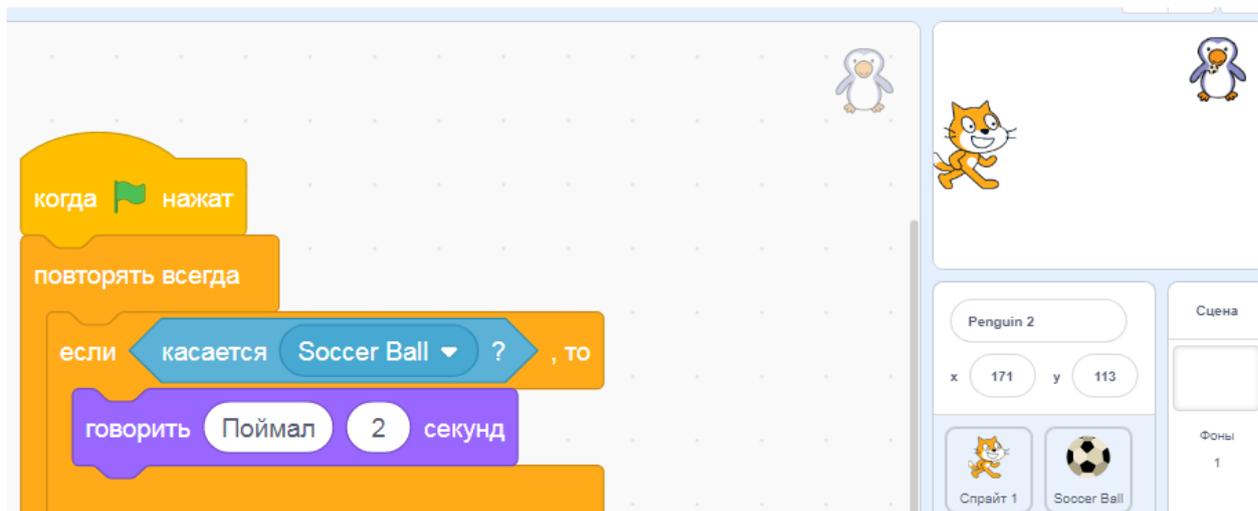
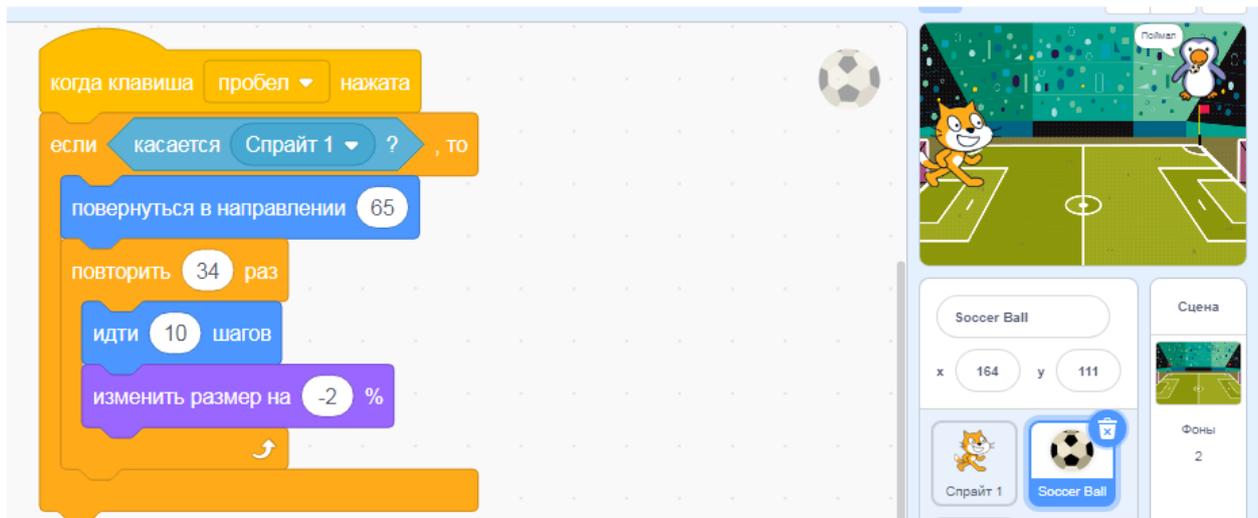
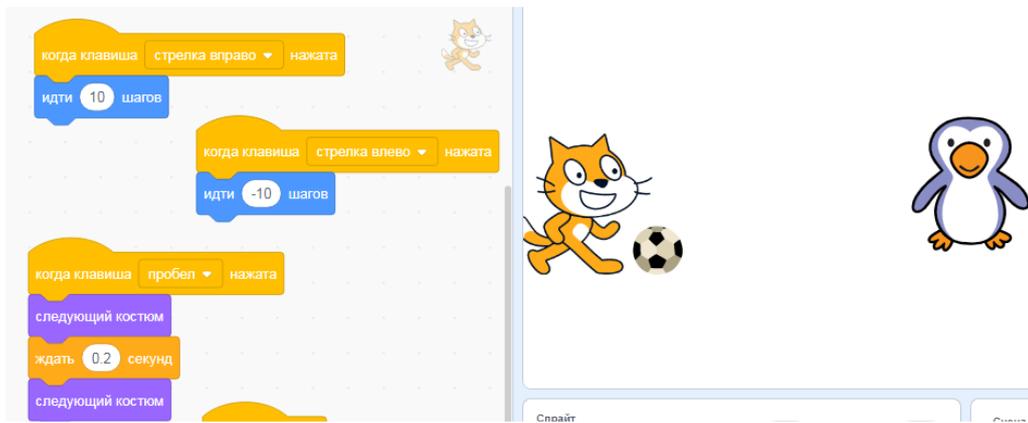
## Тема 5.2. Разработка базовых спрайтов для игры. Формирование базовых скриптов.

**Практика.** Разработка и создание основных спрайтов и их костюмов для будущей игры. Разработка скриптов для спрайтов и объектов (1 час).



## Тема 5.3. Синхронизация работы скриптов для разных спрайтов

**Практика.** Доработка основного листинга программы с целью установления связей между спрайтами. Тестирование и отладка программы (1 час).



**Тема 5.4. Переход из одной сцены в другую. Создание интерфейса игры.**

**Практика.** Создать программу для перемещения объекта по игровой карте и разработать интерфейс для Scratch-проекта (1 час).

Scratch code editor showing the following blocks:

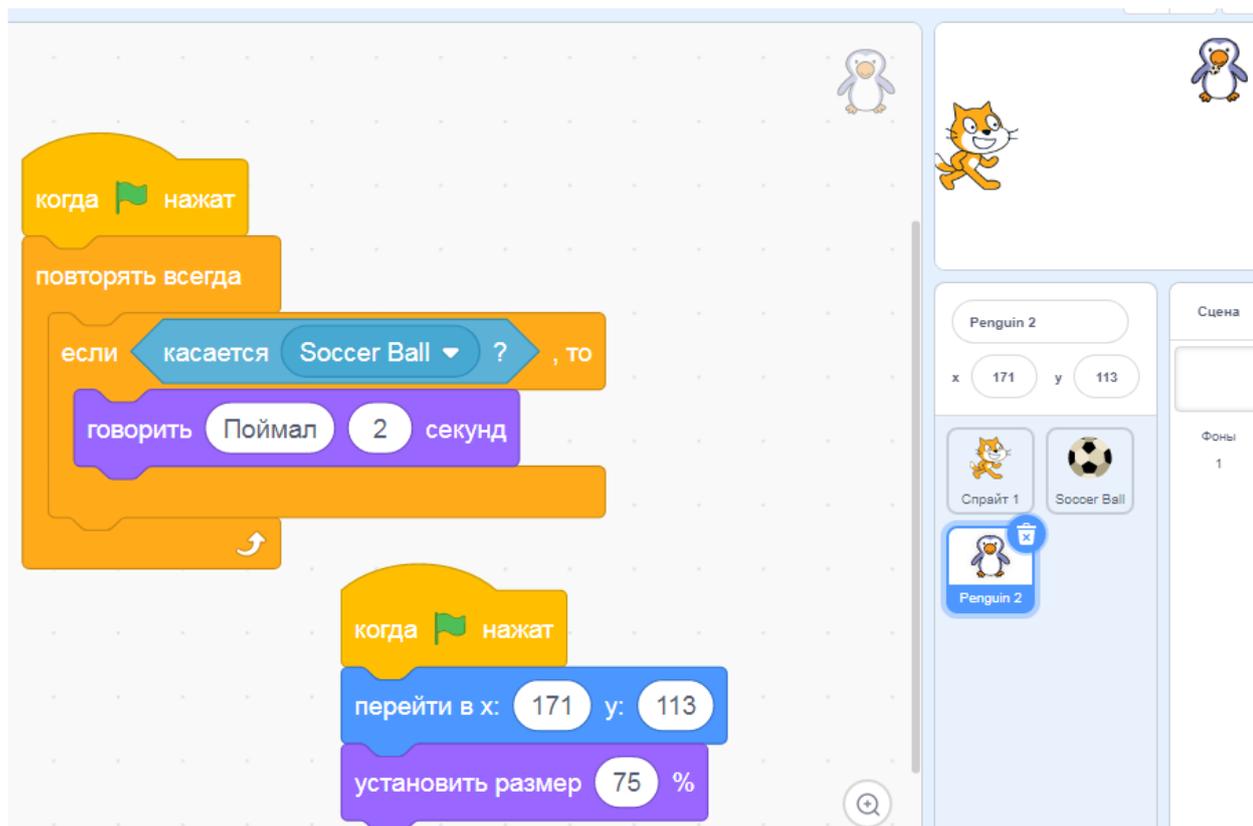
- When key pressed: right arrow key pressed
- Move 10 steps
- When key pressed: left arrow key pressed
- Move -10 steps
- When key pressed: space key pressed
- Next costume
- Wait 0.2 seconds
- Next costume
- When green flag clicked
- Go to x: -181 y: 1
- Set size to 130%

Stage view shows a cat sprite (Sprite 1) and a penguin sprite (Penguin 2). The cat is currently holding a soccer ball.

Scratch code editor showing the following blocks:

- When key pressed: space key pressed
- If Sprite 1 touches? (collision event)
- Turn 65 degrees
- Repeat 34 times
  - Move 10 steps
  - Change size by -2%
- When green flag clicked
- Go to front layer
- Go to x: -144 y: -33
- Set size to 100%

Stage view shows a soccer field scene (Scene 2) with a cat sprite (Sprite 1) and a soccer ball sprite (Soccer Ball). The penguin sprite (Penguin 2) is also visible in the scene.

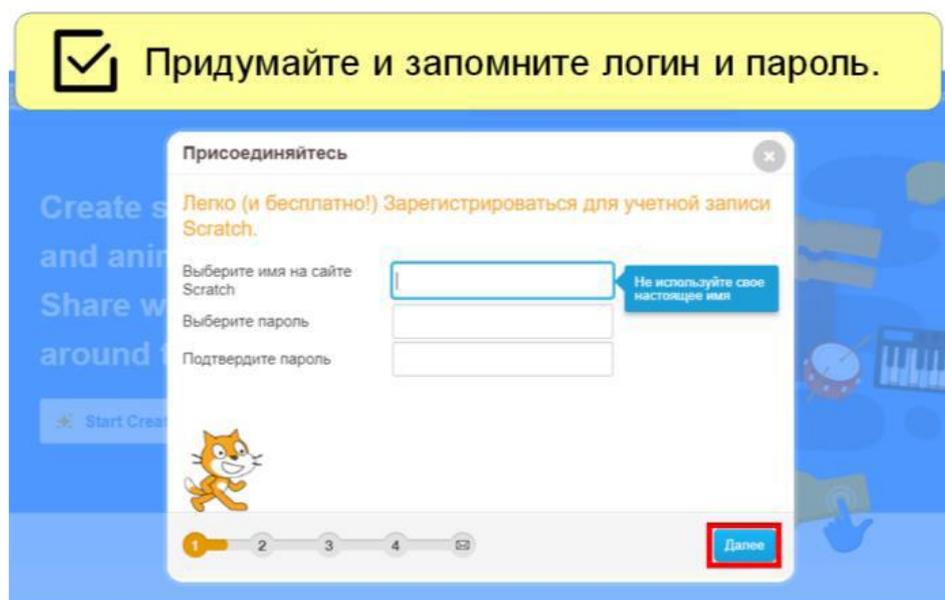


### Тема 5.5. Сообщество Scratch в Интернете. Просмотр и публикация проектов.

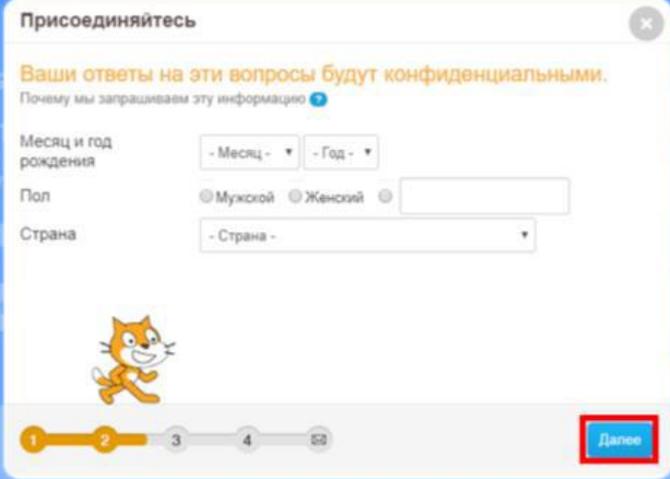
**Практика.** Регистрация на сайте сообщества Scratch. Просмотр проектов сообщества и публикация собственных проектов (1 час).

Для сохранения проектов регистрируйтесь на сайте <https://scratch.mit.edu/>.

**Придумайте и запомните логин и пароль**, под этими данными вы сможете заходить на сайт Scratch с любого компьютера, где есть доступ к сети Интернет – из дома, на занятиях, с планшета, телефона и т.д.



Выберите дату рождения, пол и страну.



Присоединяйтесь

Ваши ответы на эти вопросы будут конфиденциальными.  
Почему мы запрашиваем эту информацию?

Месяц и год рождения: - Месяц - | - Год -

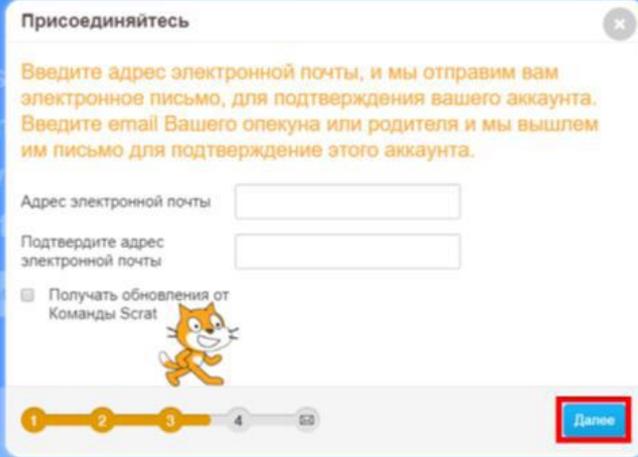
Пол:  Мужской  Женский

Страна: - Страна -

Progress: 1 | 2 | 3 | 4 | 50

Далее

Введите адрес вашей электронной почты.



Присоединяйтесь

Введите адрес электронной почты, и мы отправим вам электронное письмо, для подтверждения вашего аккаунта. Введите email Вашего опекуна или родителя и мы вышлем им письмо для подтверждения этого аккаунта.

Адрес электронной почты:

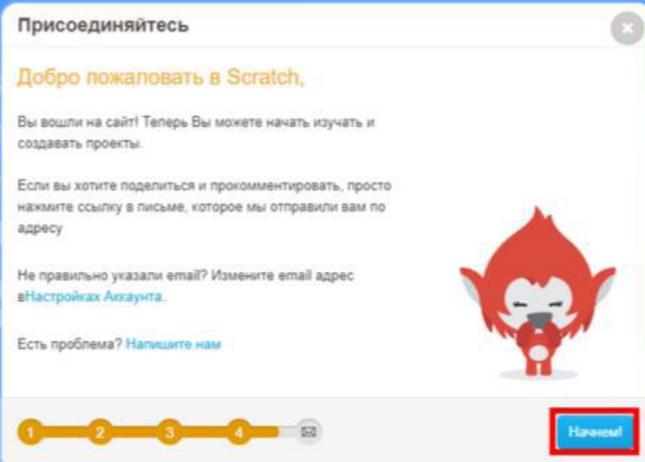
Подтвердите адрес электронной почты:

Получать обновления от Команды Scratch

Progress: 1 | 2 | 3 | 4 | 50

Далее

Учетная запись готова. Начинаем!



Присоединяйтесь

Добро пожаловать в Scratch,

Вы вошли на сайт! Теперь Вы можете начать изучать и создавать проекты.

Если вы хотите поделиться и прокомментировать, просто нажмите ссылку в письме, которое мы отправили вам по адресу

Не правильно указали email? Измените email адрес в [Настройках Аккаунта](#).

Есть проблема? [Напишите нам](#)

Progress: 1 | 2 | 3 | 4 | 50

Начнем!

## КРИТЕРИИ ОЦЕНИВАНИЯ РЕЗУЛЬТАТОВ ОСВОЕНИЯ ПРОГРАММЫ

Ф.И. ребенка	Умение правильно подбирать инструменты работы в Scratch по инструкции педагога	Умение правильно создавать простые проекты в среде Scratch по алгоритму	Умение правильно использовать векторный и растровый графический редактор, изменять звук, вводить, выводить и обрабатывать информацию в среде Scratch по образцу	Умение правильно выбирать варианты и самостоятельно искать средства/ресурсы для решения задачи и достижения цели при создании проекта в среде Scratch по замыслу	Умение вносить коррективы в текущую деятельность на основе анализа изменений для получения запланированных характеристик продукта/результата	Умение ребенка моделировать объекты и самостоятельно их программировать

Степени мастерства ребенка:

Высшее мастерство: красный цвет

Достаточное мастерство: синий цвет

Недостаточное мастерство: зелёный цвет

<b>Уровень развития ребёнка</b>	<b>Создание проекта в среде Scratch по алгоритму</b>	<b>Создание проекта в среде Scratch по замыслу</b>
Высокий	Обучающийся самостоятельно создает простой проект в среде Scratch, используя образец / алгоритм, действует самостоятельно и практически без ошибок в размещении логических блоков.	Обучающийся самостоятельно разрабатывает проект в среде Scratch. При выполнении той или иной деятельности не испытывает особых затруднений.
Средний	Обучающийся делает незначительные ошибки при работе по образцу / алгоритму, правильно выбирает логические блоки, но требуется помощь при определении их расположения	Тему проекта обучающийся определяет заранее. Способ построения проекта находит путём практических проб, требуется помощь педагога. Стремится самостоятельно исправить ошибки, указанные педагогом.
Низкий	Обучающийся испытывает серьезные затруднения, постоянно ошибается в выборе логических блоков и их расположении относительно друг друга.	Замысел у обучающегося неустойчивый, тема проекта меняется в процессе практических действий с деталями. Элементы нечеткие по содержанию. Объяснить их смысл и способ построения, обучающийся не может.

<b>Уровень</b>	<b>Критерий</b>
Высокий	Самостоятельная деятельность обучающегося; при выполнении той или иной деятельности обучающийся не испытывает особых затруднений;
Средний	При выполнении той или иной деятельности обучающийся испытывает минимальные затруднения, прибегает к помощи педагога, стремится исправить указанные ошибки, самостоятельно выполняет задания;
Низкий	Обучающийся испытывает серьезные затруднения при выполнении той или иной деятельности, нуждается в постоянной помощи и контроле педагога; овладел менее чем 1/3 умениями

