МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ОБРАЗОВАНИЕ

НАЦИОНАЛЬНЫЕ ПРОЕКТЫ РОССИИ

СЕТЬ ЦЕНТРОВ ЦИФРОВОГО ОБРАЗОВАНИЯ ДЕТЕЙ «IT-КУБ»

РЕАЛИЗАЦИЯ ОБЩЕОБРАЗОВАТЕЛЬНЫХ ПРОГРАММ ПРЕДМЕТНОЙ ОБЛАСТИ «МАТЕМАТИКА И ИНФОРМАТИКА» С ИСПОЛЬЗОВАНИЕМ ОБОРУДОВАНИЯ ЦЕНТРА ЦИФРОВОГО ОБРАЗОВАНИЯ ДЕТЕЙ «IT-КУБ»

МОСКВА 2021

С. Г. Григорьев

И. Е. Вострокнутов

В. Б. Трухманов

М. А. Родионов

О. А. Исайкин

Реализация образовательных программ предметной области «Математика и информатика» с использованием оборудования центра цифрового образования детей «IT-куб»

Методическое пособие

под редакцией С. Г. Григорьева

Москва, 2021

Содержание

Пояснительная записка

В настоящее время человеческое общество стремительно вошло в новый этап своего развития, получивший название четвёртой промышленной революции. Одной из главных её особенностей является глобальная информатизация всех сфер человеческой жизни, включая систему образования. В этих условиях особое значение среди школьных учебных предметов приобретает информатика. Сегодня знания в области информатики, программирования, информационных технологий становятся базовыми, без них становится невозможно получить современную профессию и просто комфортно жить. Человечество столкнулось с поистине беспрецедентным расширением содержания предмета «Информатика», которое уже невозможно вместить в отдельный, даже достаточно объёмный школьный курс.

Выходом из создавшегося положения может быть перенос части содержания (возможно, наиболее сложного) в область дополнительного образования. В последнее время в нашей стране ведутся работы в этом направлении, причём как на федеральном, так и региональном уровне. Так, в соответствии с Национальным проектом «Образование» во всех регионах созданы центры дополнительного образования детей «Сириус», «Кванториум», «IT-Куб» и др. Такие центры создаются не только по целевым федеральным и региональным программам, но и на базе ведущих вузов страны.

Достижения в этой области весьма убедительны. Так, например, в проекте «IT-Куб» были разработаны содержание и методика основных учебных разделов школьной информатики, которые целесообразно изучать в дополнительном образовании, а также перечень необходимого оборудования для этих целей. Но в последнее время наметилась тенденция разрыва и иногда противопоставления содержания школьной информатики и содержания работы центров дополнительного образования, например, «IT-Куб».

Назначение представленного методического пособия по реализации образовательных программ предметной области «Математика и информатика» с использованием оборудования центра цифрового образования детей «IT-Куб» — согласовать содержание школьного курса информатики и центров дополнительного образования с целью повышения эффективности обучения и качества образования школьников в области информатики.

В качестве тем школьного курса информатики были определены:

«Основы алгоритмизации» (8 класс);

«Начала программирования» (8 класс);

«Алгоритмизация и программирование» (9 класс).

Предложены факультативные курсы для 5-9 классов:

«Основы робототехники»;

«Разработка приложений VR/AR».

Нормативная база

1. Конституция Российской Федерации (принята всенародным голосованием 12.12.1993 с изменениями, одобренными в ходе общероссийского голосования 01.07.2020). — URL: http://www.consultant.ru/document/cons_doc_LAW_28399/ (дата обращения: 10.03.2021).

2. Федеральный закон «Об образовании в Российской Федерации» от 29.12.2012 № 273-ФЗ (ред. от 31.07.2020) (с изм. и доп., вступ. в силу с 01.09.2020). — URL: http://www.consultant.ru/document/cons_doc_LAW_140174 дата обращения: 28.09.2020).

3. Паспорт национального проекта «Образование» (утв. президиумом Совета при Президенте РФ по стратегическому развитию и национальным проектам, протокол от 24.12.2018 № 16). — URL: http://www.consultant.ru/document/cons_doc_LAW_319308/ (дата обращения: 10.03.2021).

4. Государственная программа Российской Федерации «Развитие образования» (утв. Постановлением Правительства РФ от 26.12.2017 № 1642 (ред. от 22.02.2021) «Об утверждении государственной программы Российской Федерации «Развитие образования». — URL: http: //www.consultant.ru document cons_doc_LAW_286474 (дата обращения: 10.03.2021).

5. Стратегия развития воспитания в Российской Федерации на период до 2025 года (утверждена распоряжением Правительства РФ от 29.05.2015 № 996-р «Об утверждении Стратегии развития воспитания в Российской Федерации на период до 2025 года»). — URL: http://www.consultant.ru/document/cons_doc_LAW_180402/ (дата обращения: 10.03.2021).

6. Профессиональный стандарт «Педагог (педагогическая деятельность в дошкольном, начальном общем, основном общем, среднем общем образовании), (воспитатель, учитель)» (ред. от 16.06.2019) (приказ Министерства труда и социальной защиты РФ от 18.10.2013 № 544н, с изменениями, внесёнными приказом Министерства труда и соцзащиты РФ от 25.12.2014 № 1115н и от 05.08.2016г. № 422н). — URL: //http://профстандартпедагога.рф (дата обращения: 10.03.2021).

7. Профессиональный стандарт «Педагог дополнительного образования детей и взрослых» (приказ Министерства труда и социальной защиты РФ от 05.05.2018 № 298н «Об утверждении профессионального стандарта «Педагог дополнительного образования детей и взрослых»). — URL: https://profstandart.rosmintrud.ru/obshchiy-informatsionnyy-blok/natsionalnyy-reestr-professionalnykh-standartov/reestr-professionalnykh-standartov/index.php?ELEMENT_ID=48583 (дата обращения: 10.03.2021).

8. Федеральный государственный образовательный стандарт основного общего образования (утв. приказом Министерства образования и науки Российской Федерации от 17.12.2010 № 1897) (ред. 21.12.2020). — URL: https://fgos.ru (дата обращения: 10.03.2021).

9. Федеральный государственный образовательный стандарт среднего общего образования (утверждён приказом Министерства образования и науки Российской Федерации от 17.05.2012 № 413) (ред. 11.12.2020). — URL: https://fgos.ru (дата обращения: 10.03.2021).

10. Федеральный государственный образовательный стандарт начального общего образования (утв. приказом Минобрнауки России от 26.11.2010 № 1241, от 22.09.2011 № 2357). — URL: https://fgos.ru (дата обращения: 10.03.2021).

11. Методические рекомендации по созданию и функционированию детских технопарков «Кванториум» на базе общеобразовательных организаций (утв. распоряжением Министерства просвещения Российской Федерации от 12.01.2021 № Р-4). — URL: http://www.consultant.ru/document/cons_doc_LAW_374695/ (дата обращения: 10.03.2021).

12. Методические рекомендации по созданию и функционированию центров цифрового образования «IT-Куб» (утв. распоряжением Министерства просвещения Российской Федерации от 12.01.2021 № Р-5). — URL: http://www.consultant.ru/document/cons_doc_LAW_374572/ (дата обращения: 10.03.2021).

13. Методические рекомендации по созданию и функционированию в общеобразовательных организациях, расположенных в сельской местности и малых городах, центров образования естественно-научной и технологической направленности («Точка роста») (утв. распоряжением Министерства просвещения Российской Федерации от 12.01.2021

№ Р-6). — URL: http://www.consultant.ru/document/cons_doc_LAW_374694/ (дата обращения: 10.03.2021).

Основные понятия и термины

Алгоритм — конечное точное предписание действий, которые необходимо выполнить для решения поставленной задачи.

Ассеты — компоненты, которые представляют собой графику, звуковое сопровождение или скрипты.

Визуализация — метод предоставления абстрактной информации в форме, удобной для зрительного восприятия и анализа явления или числового значения.

Виртуальная реальность (VR) — совокупность технологий, с помощью которых можно создать искусственный мир, физически не существующий, но ощущаемый органами чувств в реальном времени в соответствии с законами физики.

Датчик — средство измерений, предназначенное для выработки сигнала измерительной информации в форме, удобной для передачи, дальнейшего преобразования, обработки и (или) хранения, но не поддающейся непосредственному восприятию наблюдателя. Датчики, выполненные на основе электронной техники, называются электронными датчиками. Отдельно взятый датчик может быть предназначен для измерения (контроля) и преобразования одной физической величины или одновременно нескольких физических величин.

Дополненная реальность (AR) — среда в реальном времени, дополняющая физический мир, каким мы его видим, цифровыми данными с помощью каких-либо устройств (планшетов, смартфонов или т. д.) и программной части.

Игровое поле — заранее сконфигурированная площадка с заданиями для робота.

Исполнитель алгоритма — некоторый объект (техническое устройство, робот, автомат), способный выполнять определённый набор команд алгоритма.

Консоль экрана — специальное окно для вывода значений и сообщений в ходе выполнения роботом заданий на игровом поле.

Линейный алгоритм — это алгоритм, в котором команды последовательно выполняются однократно одна за другой.

Оператор — символ, который выполняет операцию над одним или несколькими операндами.

Оператор цикла — оператор, который выполняет одну и ту же последовательность действий несколько раз, количество повторений либо задано, либо зависит от истинности или ложности некоторого условия.

Переменная — область памяти компьютера, которая предназначена для хранения данных.

Смешанная реальность (MR) (или гибридная реальность) — модель мировосприятия, в которой объединены реальный и виртуальный миры.

Список — в среде Scratch это сложная переменная, предназначенная для хранения нескольких значений.

Среда Scratch — визуальный язык программирования, позволяющий создавать интерактивные мультимедийные проекты.

Сферическая панорама (виртуальная панорама, 3D-панорама) — один из видов панорамной фотографии. Предназначена, в первую очередь, для показа на компьютере (при помощи специального программного обеспечения).

Трансмиссия — группа команд среды Scratch, задающих различные виды движений исполнителя алгоритма

Трёхмерная графика — вид компьютерной графики, представляющий собой объёмную модель какого-либо объекта.

Условный алгоритм — это алгоритм, порядок выполнения команд которого зависит от истинности или ложности некоторого условия.

Условный оператор — оператор, который используется для выбора среди альтернативных операций на основе истинности или ложности некоторого условия.

Хакатон — короткое (от одного дня до недели), динамичное мероприятие, призванное стимулировать появление новых идей в выбранной предметной области и доведение их участниками до проектной реализации непосредственно на площадке хакатона.

Хромакей — технология совмещения двух и более изображений или кадров в одной композиции.

Циклический алгоритм — это алгоритм, предусматривающий многократное повторение группы команд, называемых телом цикла.

Язык программирования — набор формальных правил, по которым пишут программы.

3D-моделирование — процесс создания трёхмерного представления любой поверхности или объекта путём манипулирования полигонами, рёбрами и вершинами в моделируемом трёхмерном пространстве.

3D-прототипирование — процесс создания трёхмерного прототипа объекта.

3D-модель — объёмное цифровое изображение реального или вымышленного объекта, результат 3D-моделирования.

API (Application Programming Interface) — набор готовых классов, функций, процедур, констант и структур, предоставляемых приложением или операционной системой для использования во внешних программных продуктах.

Python — это язык программирования, применяемый для разработки самостоятельных программ, а также для создания прикладных сценариев в самых разных областях применения.

SDK (Software Development Kit) — набор средств разработки, позволяющий программистам разрабатывать приложения для определённой платформы.

Структурирование учебных и методических материалов

Структура учебных и методических материалов имеет следующий вид.

Примерный учебно-тематический план обучения информатике с использованием оборудования центра «IT-Куб».

Примерный учебно-тематический план изучения тем курса «Информатика» во внеурочной работе.

Учебная программа курсов «Основы алгоритмизации» и «Начала программирования» для 8 класса.

Содержание и формы организации учебных занятий по курсам «Основы алгоритмизации» и «Начала программирования» для 8 класса.

Учебная программа курса «Алгоритмизация и программирование» для 9 класса.

Содержание и формы организации учебных занятий по курсу «Алгоритмизация и программирование» для 9 класса.

Учебная программа факультативного курса «Разработка приложений VR/AR» для 8—9 классов.

Учебная программа факультативного курса «Основы робототехники» для 5—7 классов.

Материалы для организации учебно-исследовательской и проектной деятельности школьников.

Примерный учебно-тематический план обучения информатике с использованием оборудования центра «IT-куб»

Учебно-тематический план изучения тем курса «Информатика» в 8—9 классах

№	Название темы	Количество часов		
		Общее	Теория	Практика
8 класс				
1	Основы алгоритмизации	4	2	2
2	Начала программирования	10	2	8
9 класс				
1	Алгоритмизация и программирование	8	2	6
	Итого:	22	6	16

Учебно-тематический план изучения тем курса «Информатика» во внеурочной работе в 5—9 классах

№	Название факультативного курса	Количество часов		
		Общее	Теория	Практика
5—7 класс				
1	Основы робототехники	36	9	27
8—9 классы				
1	Разработка приложений VR/AR	36	9	27
	Итого:	72	18	54

Описание материально-технической базы

Рекомендуется следующее оборудование лаборатории:

- ноутбук — рабочее место преподавателя;
- рабочее место обучающегося;
- жёсткая неотключаемая клавиатура: наличие;
- русская раскладка клавиатуры: наличие;
- диагональ экрана: не менее 15,6 дюйма;
- разрешение экрана: не менее 1920×1080 пикселей;
- количество ядер процессора: не менее 4;
- количество потоков: не менее 8;
- базовая тактовая частота процессора: не менее 1 ГГц;
- максимальная тактовая частота процессора: не менее 2,5 ГГц;
- кэш-память процессора: не менее 6 Мбайт;
- объём установленной оперативной памяти: не менее 8 Гбайт;
- объём поддерживаемой оперативной памяти (для возможности расширения): не менее 24 Гбайт;
- объём накопителя SSD: не менее 240 Гбайт;
- время автономной работы от батареи: не менее 6 часов;
- вес ноутбука с установленным аккумулятором: не более 1,8 кг;
- внешний интерфейс USB стандарта не ниже 3.0: не менее трёх свободных;
- внешний интерфейс LAN (использование переходников не предусмотрено): наличие;
- наличие модулей и интерфейсов (использование переходников не предусмотрено): VGA, HDMI;
- беспроводная связь Wi-Fi: наличие с поддержкой стандарта IEEE 802.11n или современнее;
- веб-камера: наличие;
- манипулятор «мышь»: наличие;
- предустановленная операционная система с графическим пользовательским интерфейсом, обеспечивающая работу распространённых образовательных и общесистемных приложений: наличие;
- МФУ, веб-камера, интерактивный моноблочный дисплей (диагональ экрана: не менее 65 дюймов, разрешение экрана: не менее 3840×2160 пикселей), оборудованный напольной стойкой.

Рис. 1. Лаборатория центра «IT-Куб»

Лаборатории в центрах «IT-Куб» оборудованы ноутбуками Asus, процессор Intel (R) Core ™ i5-8256UCPU, 1,60 ГГц, ОЗУ 600 Гбайт. Они оснащены также интерактивной доской, моноблочным интерактивным устройством, маркерной доской, МФУ.

На данном оборудовании могут выполняться лабораторные работы по темам «Основы алгоритмизации», «Начала программирования» для 8 класса и «Алгоритмизация и программирование» для 9 класса, а также могут проводиться открытые занятия, защита проектов и т. д.

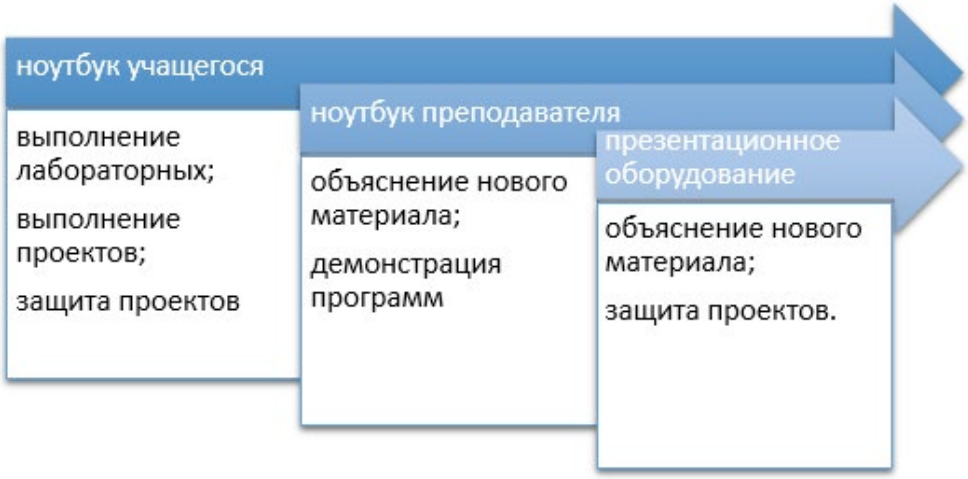

Рис. 2. Использование оборудования центра «IT-Куб» при организации занятий по темам «Основы алгоритмизации», «Начала программирования», «Алгоритмизация и программирование»

Для факультативных курсов «Основы робототехники» и «Разработка приложений AR/VR» необходимо дополнительное оборудование.

Для курса «Разработка приложений VR/AR» потребуется профессиональный шле м виртуальный реальности, например, HTC VIVE PRO Starter Kit, шлем виртуальной реальности полупрофессиональный типа FP-1548, шлем виртуальной реальности любительский типа HTC Vive Focus Plus и наушники, например PANASONIC RP-HTF295E. Также потребуются современные смартфоны, например SAMSUNG Galaxy M21 64Gb, SM-M215F, и бинокулярные прозрачные видеоочки, например Epson bt300.

Для курса «Основы робототехники» потребуются робототехнические конструкторы:
цифровая лаборатория школьника «Тетра»;
— робототехническая платформа «Роббо»;
— Modkit for VEX;
— Lego Education Spike.

Функционал оборудования	Внешний вид оборудования
Виртуальная среда программирования роботов VEX Code VR. Предназначена для отработки навыков программирования роботов в среде Scratch и в дальнейшем с переходом на программирование в Python и C++	 VEX CODE VR

Продолжение

Функционал оборудования	Внешний вид оборудования
Робототехнический конструктор с программируемым контроллером, комплектом датчиков и ресурсным набором комплектующих для разработки сложных мехатронных систем и моделей роботов для участия в робототехнических соревнованиях. Конструктор предназначен для разработки мобильных роботов и углублённой практики программирования. Программируется в редакторе RobotC как графическими блоками, так и в текстовом режиме. Может обучать дистанционно в среде «Виртуальные миры»	
VEX V5 представляет собой пятое поколение образовательных робототехнических систем, разработанных с учётом 20-летнего опыта использования робототехники для обучения принципам STEM. Электроника V5 доступная, гибкая и мощная, в ней используются самые современные технологии для обеспечения соответствующих результатов обучения. Механическая система V5 включает в себя универсальные элементы, которые делают проектирование доступным для начинающих пользователей, в то же время предоставляя опытным пользователям безграничные возможности проектирования	
Робот-манипулятор, разработанный и производимый в России, предназначен для освоения школьниками и студентами основ робототехники и подготовки учащихся к внедрению и последующему использованию роботов в промышленном производстве. В качестве управляющего контроллера взята Arduino-совместимая плата, отлично зарекомендовавшая себя в линейке образовательных наборов для старшего школьного возраста. Благодаря этому достигается методическая и программная совместимость с широко распространённым ПО mBlock, обладающим уникальными особенностями по работе с образовательным робототехническим оборудованием. Данное ПО основано на Scratch, но поддерживает и программирование на языке С, что сильно расширяет возрастные рамки обучающихся	
Серия Lego Mindstorms EV3 разработана специально для обучения детей в образовательных учреждениях робототехнике, а также естественным наукам — физике, математике, информатике и конструированию. В основе набора — микрокомпьютер Mindstorms EV3 с графическим дисплеем и портами для подключения датчиков. Сердцем набора является программируемый микрокомпьютер EV3, с помощью которого роботом можно управлять, контролировать работу моторов и датчиков, а также получать данные на компьютер посредством протоколов Bluetooth и Wi-Fi. Наборы Lego Mindstorms обладают широчайшим учебным потенциалом и могут быть использованы при изучении большинства технических предметов для повышения эффективности учебного процесса	

Рис. 3. Использование оборудования центра «IT-Куб» при организации занятий по теме «Основы робототехники»

Программное обеспечение

Программное обеспечение выбирается на основе соответствия его дидактических возможностей содержанию преподаваемой дисциплины.

Программное обеспечение курсов «Основы алгоритмизации» и «Начала программирования» (8 класс)

Для обучения основам алгоритмизации и началам программирования целесообразно в качестве базового языка использовать визуальную среду программирования Scratch. На сегодняшний день она является наиболее удачной средой (из числа бесплатных) для обучения основам программирования школьников 5–8 классов. Несомненным достоинством Scratch является то, что это красивая, наглядная и вместе с тем несложная среда обучения программированию. Она напоминает виртуальный конструктор, в ней программа собирается из готовых элементов.

В качестве исполнителей алгоритма программы используются спрайты (рыжий Кот, Собачка, Мячик и др.). В программе имеется много готовых исполнителей, есть редактор исполнителей, в котором можно создавать своих исполнителей. По коду программы спрайты могут менять положение, цвет, могут исчезать и добавляться новые персонажи на экран. Можно также выбирать и настраивать сцену, менять её по коду программы. Всё это позволяет обучать основам программирования в наглядной игровой форме.

Платформа оснащена подсказками, возможностями для повтора учебного материала, дополнительными заданиями и онлайн-чатом с преподавателем. Дидактические возможности Scratch таковы, что дают возможность учащимся создавать программы различной степени сложности: от простой анимации до разработки собственной многоуровневой игры со своим сюжетом и звуковым сопровождением.

Программное обеспечение курса «Алгоритмизация и программирование» (9 класс)

В качестве рабочей следует установить версию Python 3.8.0. Эта среда программирования является наиболее устойчивой свободно распространяемой версией языка Python. В самом Python писать и отлаживать программы не очень удобно, поэтому желательно установить один из редакторов кода. Существует достаточно много сред, поддерживающих Python. Например, язык Python поддерживает Visual Studio. Наиболее популярными редакторами кода на Python являются Sublime Text, ATOM, GNU Emacs, Vim, Eclipse, Thonny и др. Рекомендуем в качестве исходного редактора на начальной стадии освоения языка использовать PyCharm.

В редакторе PyCharm есть возможности, которые делают его очень простым и удобным для школьников, особенно на стадии освоения языка. У него простой и интуитивно понятный интерфейс, имеется опция поиска места ошибки и подсветка синтаксиса, мгновенная проверка синтаксиса с сопоставлением скобок и кавычек, автозавершение кода. PyCharm даёт возможность подключать большое количество библиотек, которые существенно расширяют возможности базового языка Python, позволяя разрабатывать более красивые и интересные программы, включая и динамические программы с управлением объектами.

Программное обеспечение курса «Разработка приложений VR/AR»

Для разработки приложений VR/AR потребуется программное обеспечение для трёхмерного моделирования. Можно использовать Autodesk 3ds Max, Blender 3D, Autodesk Maya. Например, в программе Blender 3D удобно создавать 3D объекты. Для создания маркеров дополненной реальности удобно использовать приложение Arloopa. Для раскраски изображений дополненной реальности можно использовать приложение Quiver. Для создания различных эффектов целесообразно использовать Spark AR Studio.

Для сборки проектов потребуется программная среда Unity 3D или Unreal Engine. Также потребуются программы для создания панорам 360°, например Autostitch, Pano2VR, Vrap. Для переноски проекта на платформу Android потребуется плагин Vuforia. Для создания управления с помощью скриптов потребуется Java с редактором IntelliJ либо среда разработки Microsoft Visual Studio.

Выбор в пользу данного программного обеспечения объясняется его простотой и наглядность. Его вполне достаточно для того, чтобы научить школьников разрабатывать приложения VR/AR.

Программное обеспечение курса «Основы робототехники»

В качестве рабочей среды обучения следует выбрать виртуальную среду программирования роботов VEX Code VR. Она предназначена для отработки навыков программирования роботов в среде Scratch с возможностью в дальнейшем перейти на программирование на языках Python и C++.

Робототехнический конструктор «Роббо» предназначен для разработки мобильных роботов и углублённой практики программирования. Он содержит программируемый контроллер, комплект датчиков и набор комплектующих для разработки сложных мехатронных систем и моделей роботов. Комплект достаточно большой и вполне подходит для создания сложных систем, например для участия в робототехнических соревнованиях. Программируется в редакторе RobotC как графическими блоками, так и в текстовом режиме.

VEX V5 представляет собой робототехнический конструктор пятого поколения, разработанный специально для системы образования. Вся электроника надёжная и доступная для восприятия учащимися, обладающая широкими дидактическими возможностями. В ней используются самые современные технологии. Механическая система V5 включает в себя универсальные элементы, которые делают проектирование доступным для начинающих пользователей, в то же время предоставляет опытным пользователям широкие возможности конструирования робототехнических систем.

Можно добавить в конструктор робота-манипулятора. Это позволит расширить подготовку школьников в области использования роботов в промышленном производстве. В VEX V5 используется контроллер на Arduino-совместимой плате. Благодаря этому достигается методическая и программная совместимость с широко распространённым ПО mBlock, обладающим уникальными особенностями работы с образовательным робототехническим оборудованием. Программное обеспечение основано на Scratch, но поддерживает и программирование на языке C, что сильно расширяет возрастные рамки обучающихся.

Серия Lego Mindstorms EV3 разработана специально для обучения детей в образовательных учреждениях робототехнике, а также естественным наукам — физике, математике, информатике и конструированию. В основе набора — микрокомпьютер Mindstorms EV3 с графическим дисплеем и портами для подключения датчиков. В наборе имеется программируемый микрокомпьютер EV3, с помощью которого роботом можно управлять, контролировать работу моторов и датчиков, а также получать данные на компьютер посредством протоколов Bluetooth и Wi-Fi. Lego Mindstorms обладают широкими дидактическими возможностями для обучения школьников основам робототехники.

Учебная программа курсов «Основы алгоритмизации» и «Начала программирования» (8 класс)

Темы «Основы алгоритмизации» и «Начала программирования» являются важной составляющей курса информатики для 8 класса общеобразовательной школы. Под способностью алгоритмически мыслить понимается умение решать задачи различного происхождения, требующие составления плана действий для достижения желаемого результата. Для того чтобы записать алгоритм решения задачи, необходим какой-то формальный язык, например блок-схемы, учебные исполнители. Поэтому с учётом возрастных особенностей учащихся выберем в качестве учебного язык программирования Scratch.

В примерной программе по информатике предполагается рассмотрение основных алгоритмических конструкций: ветвление, цикл, вспомогательный алгоритм. Выбор языка программирования Scratch также обусловлен тем, что знания по алгоритмизации и началам программирования, полученные на его основе, могут быть хорошей базой для обучения более сложному языку, например Python.

Целью обучения школьников по программам «Основы алгоритмизации» и «Начала программирования» является развитие алгоритмического мышления, их творческих способностей, аналитических и логических компетенций.

Планируемые результаты освоения программы

Для реализации поставленной цели планируется достижение личностных, метапредметных (познавательных, регулятивных, коммуникативных УУД) и предметных результатов.

Познавательные УУД:

- развитие алгоритмического и логического мышления;
- развитие умений постановки задачи, выделения основных объектов, построение математической модели задачи;
- развитие умения поиска необходимой учебной информации;
- формирование представления об этапах решения задачи;
- формирование алгоритмического подхода к решению задач;
- формирование ключевых компетенций проектной и исследовательской деятельности;
- формирование мотивации к изучению программирования.

Регулятивные УУД:

- формирование умений целеполагания;
- формирование умений прогнозировать свои действия и действия других участников группы;
- умение делать выводы в процессе работы и по её окончании;
- умение соотносить свои действия с планируемыми результатами;
- умение осуществлять контроль своей деятельности и определять способы действий в рамках предложенных условий, корректировать свои действия в соответствии с изменяющейся ситуацией;
- формирование умений самоконтроля и самокоррекции.

Коммуникативные УУД:

- формирование умений работать индивидуально и в группе для решения поставленной задачи;

– формирование трудолюбия, упорства, желания добиваться поставленной цели; формирование информационной культуры.

Личностные результаты:

– формирование профессионального самоопределения;
– формирование уважительного отношения к интеллектуальному труду;
– формирование личностного самоопределения.
–

Предметные результаты:

– формирование умений построения различных видов алгоритмов (линейных, разветвляющихся, циклических) для решения поставленных задач;
– формирование умений использовать инструменты среды Scratch для решения поставленных задач;
– формирование навыков работы со структурой алгоритма;
– получение навыков работы в среде Scratch.

Тематическое планирование с определением основных видов деятельности

№ п/п	Тема	Содержание	Целевая установка урока	Кол-во часов	Основные виды деятельности обучающихся на уроке/ внеурочном занятии	Использование оборудование
1	Знакомство со средой Scratch	Изучение основных элементов интерфейса среды Scratch, приёмы работы со спрайтами, приёмы работы с фоном, составление простых скриптов из различных блоков	Знакомство со средой Scratch, изучение её основных инструментов	2	Наблюдение за работой учителя, самостоятельная работа со средой Scratch, ответы на контрольные вопросы	Компьютер, проектор, интерактивная доска
2	Линейные алгоритмы	Основные приёмы составления линейных алгоритмов в среде Scratch, решение задач на составление линейных алгоритмов	Знакомство с построением и выполнением линейных алгоритмов, работа с основными блоками в среде Scratch	2	Наблюдение за работой учителя, самостоятельная работа со средой Scratch, ответы на контрольные вопросы	Компьютер, проектор, интерактивная доска
3	Циклические алгоритмы	Знакомство с понятием «циклический алгоритм», основные приёмы составления циклических алгоритмов в среде Scratch, использование основных блоков для составления циклических алгоритмов в среде Scratch	Знакомство с основами работы с циклическими алгоритмами в среде Scratch	4	Наблюдение за работой учителя, самостоятельная работа со средой Scratch, ответы на контрольные вопросы	Компьютер, проектор, интерактивная доска
4	Работа с переменными	Основные приёмы добавления переменных в среде Scratch, использование основных блоков для работы с переменными, основные приёмы составления программ с использованием переменных в среде Scratch	Знакомство с основами работы с переменными в среде Scratch	3	Наблюдение за работой учителя, самостоятельная работа со средой Scratch, ответы на контрольные вопросы	Компьютер, проектор, интерактивная доска
5	Условные алгоритмы	Знакомство с понятием «условный алгоритм», основные приёмы составления условных алгоритмов в среде Scratch, использование основных блоков для составления условных алгоритмов в среде Scratch	Знакомство с основами работы с условными алгоритмами в среде Scratch	3	Наблюдение за работой учителя, самостоятельная работа со средой Scratch, ответы на контрольные вопросы	Компьютер, проектор, интерактивная доска
	Итого			14		

Содержание и формы организации учебных занятий

Тема 1. Знакомство со средой Scratch.

Рекомендованное количество часов: 1 час — комбинированный урок, 1 час — лабораторная работа.

Планируемые результаты: получение навыков работы в среде Scratch, освоение основных инструментов среды.

Урок 1

Уровень образования: основное общее.

Предмет: информатика.

Тема урока: «Знакомство со средой Scratch».

Класс: 8.

Тип урока: комбинированный.

Цель урока: знакомство учащихся со средой Scratch.

Время реализации: 1 академический час.

Ход урока

I. Этап постановки цели и задач урока, мотивации к учебной деятельности — 5 мин.

Деятельность учителя: объяснение особенностей среды программирования Scratch по сравнению с другими средами программирования.

II. Этап изучения нового учебного материала — 25 мин.

Деятельность учителя: демонстрация учащимся основ работы в среде Scratch.

Деятельность учеников: наблюдают за учителем, задают вопросы, отвечают на вопросы учителя.

Дидактические материалы

Scratch — это современная учебная визуальная среда программирования действий исполнителей. Она полностью бесплатная и предназначена для начального обучения программированию. Наиболее популярна online-версия. Чтобы её загрузить, достаточно в браузере набрать https://scratch.mit.edu/. Её интерфейс можно видеть на рисунке 4.

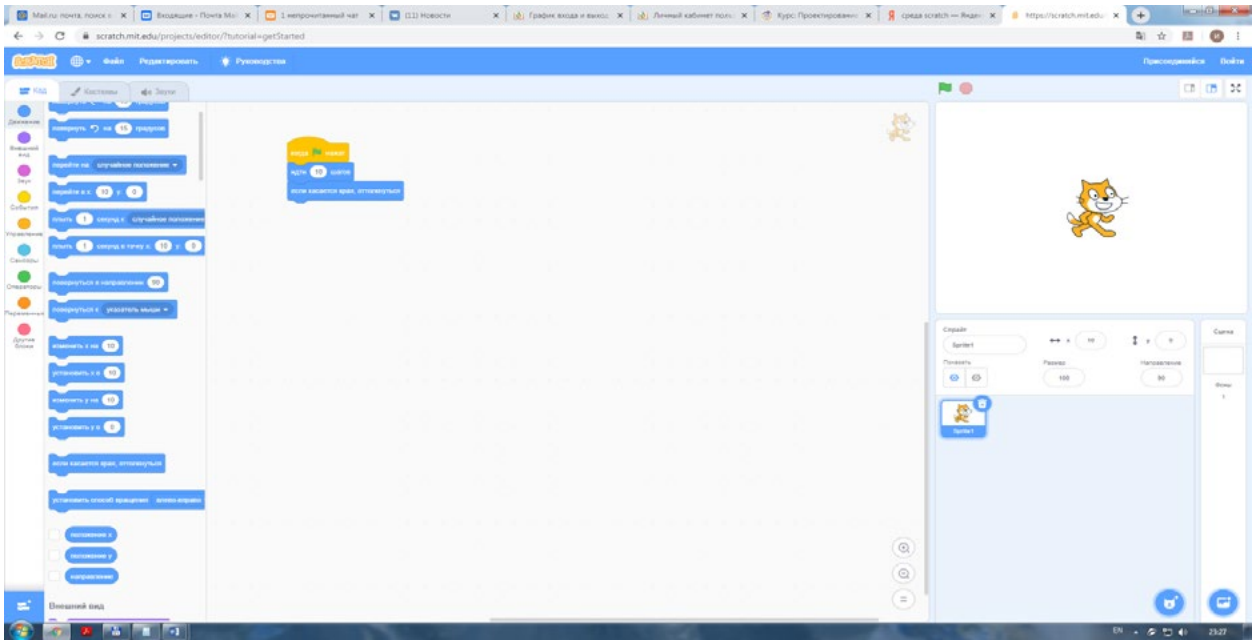

Рис. 4. Интерфейс среды Scratch

Среда полностью русифицирована. Редактор Scratch состоит из нескольких крупных частей: сцены (обычно представлена справа сверху), списка спрайтов (обычно справа снизу), палитры блоков и область скриптов (в левой части экрана).

Спрайты — это объекты, которыми можно управлять с помощью блоков. В их роли могут выступать игровые персонажи и герои мультфильмов. Также в левой части представлены закладки «Костюмы» и «Звуки».

При запуске редактора в проекте по умолчанию всегда появляется персонаж — рыжий Кот, он же является символом Scratch (рис. 5).

Рис. 5. Вид спрайта рыжий Кот

Scratch имеет собственный редактор текста программы, похожий на Lego: все операторы языка и другие его элементы представлены блоками, которые могут соединяться один с другим, образуя скрипт (рис. 6).

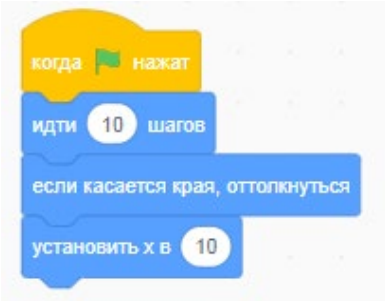

Рис. 6. Фрагмент программы в среде Scratch

В левой части среды, в закладке «Коды», содержится девять кнопок, или блоков, которые называются: «движение», «внешний вид», «звук», «события», «управление», «сенсоры», «операторы», «переменные», другие блоки (рис. 7). При включении одной кнопки все остальные выключаются. Включённая кнопка вся окрашивается в соответствующий ей цвет. При этом в нижней ячейке первого столбца появляются команды, связанные с включённой кнопкой. К каждой кнопке привязаны разные команды (рис. 8).

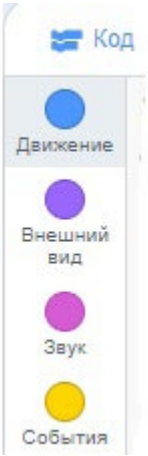

Рис. 7. Вид закладки «Коды»

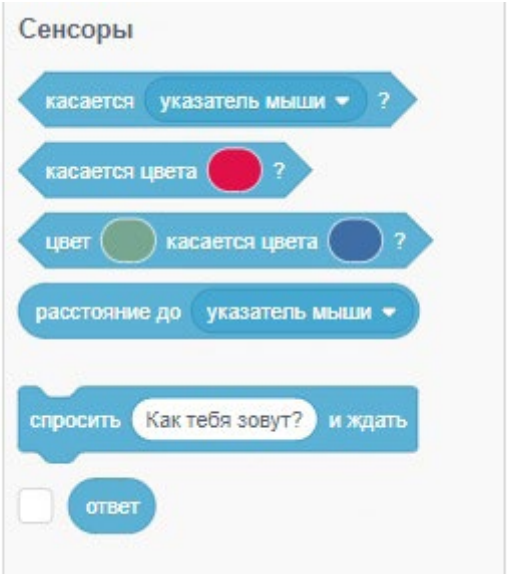

Рис. 8. Команды раздела «Сенсоры»

Имеется возможность сохранять свои проекты на компьютере и загружать их. Для этого нужно выполнить *Файл → Сохранить на свой компьютер* и *Файл → Загрузить с компьютера* (рис. 9).

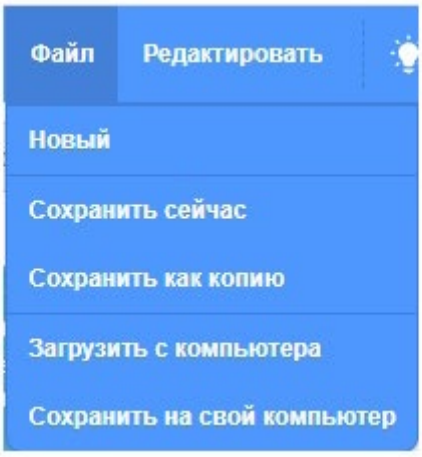

Рис. 9. Команды сохранения и загрузки файлов

Далее следует более подробно рассмотреть некоторые элементы интерфейса. Главным элементом интерфейса является сцена. Она представляет собой место, где спрайты выполняют движения, взаимодействуют друг с другом, рисуют и т. д. Стандартный размер сцены составляет 480 шагов в ширину и 360 шагов в высоту (рис. 10). Есть возможность выбирать сцену.

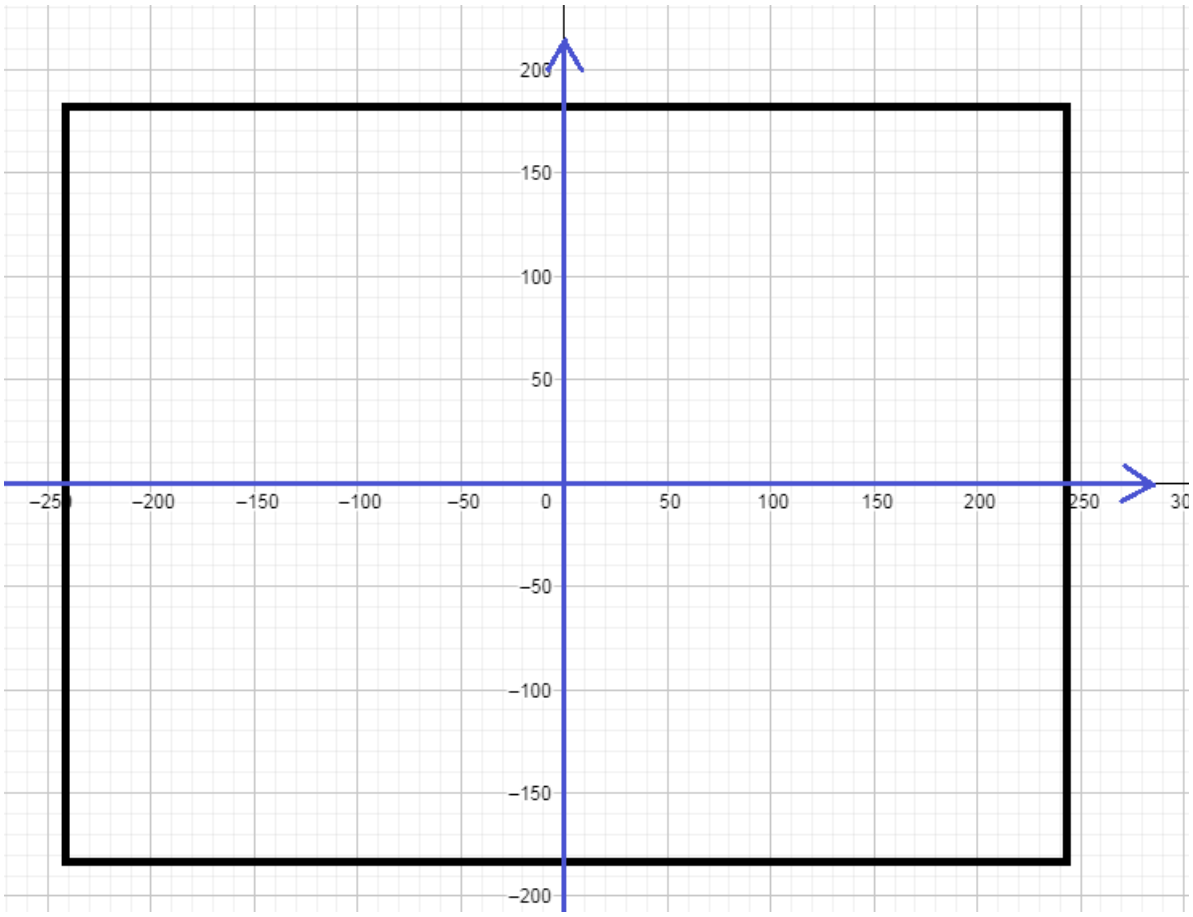

Рис. 10. Вид сцены

Центр сцены — точка (0; 0) в системе координат. Текущее положение спрайта в координатах отображается в каждый момент времени на специальной панели под сценой (рис. 11).

Рис. 11. Вид панели координат спрайта

Сцену можно развернуть во весь экран, для этого используется режим презентации (рис. 12). Зелёный флажок и значок «Стоп» предназначены для запуска и остановки программы.

Рис. 12. Элементы управления сценой

Список спрайтов также представляет собой важный элемент интерфейса. Он расположен внизу под сценой. Как было сказано ранее, основным спрайтом является рыжий Кот.

Спрайты можно выбирать. Можно использовать сразу несколько спрайтов в программе. Для этого используется команда *Выбрать спрайт* (рис. 13)

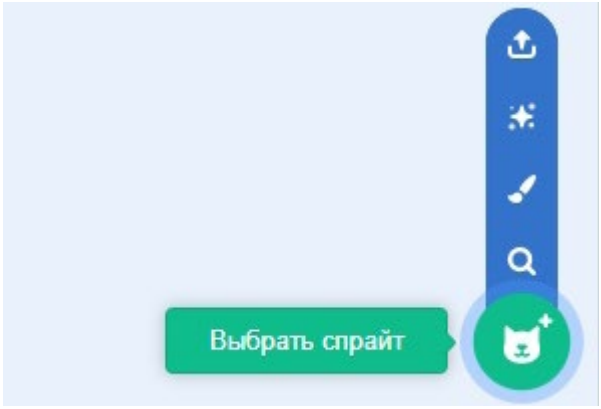

Рис. 13. Выбор спрайта

Новый спрайт можно выбрать несколькими способами:
- выбрать из библиотеки спрайтов;
- создать в собственном редакторе;
- загрузить из файла;
- выбрать случайным образом.

Вид библиотеки спрайтов представлен на рисунке 14.

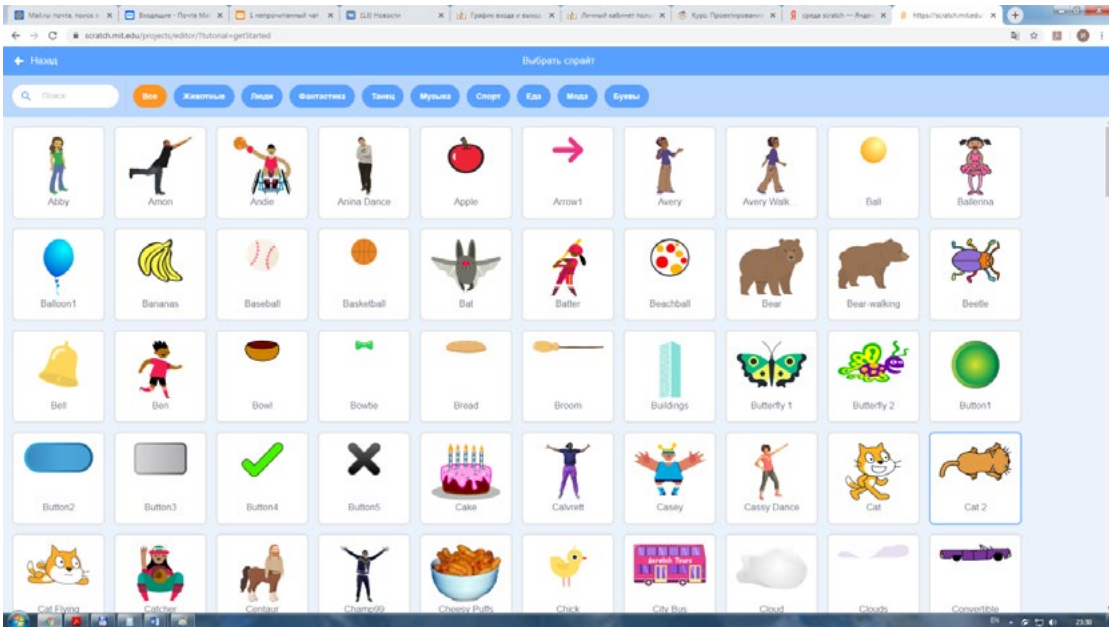

Рис. 14. Вид библиотеки спрайтов

У каждого спрайта в проекте свои скрипты, костюмы и звуки. При выборе спрайта есть возможность посмотреть на его атрибуты. Для этого необходимо либо кликнуть по его иконке в списке, либо дважды кликнуть по самому спрайту на сцене. Если щёлкнуть по спрайту правой клавишей мыши, то можно, например, вызвать команды дублирования либо экспорта и удаления спрайта (рис. 15).

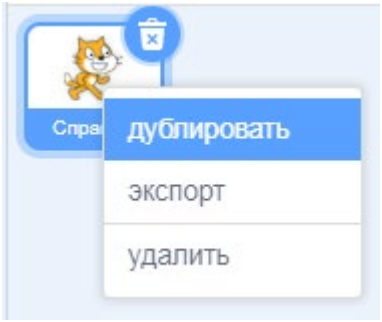

Рис. 15. Команды редактирования спрайта

Фон сцены также можно менять и редактировать, например загрузить из коллекции фонов (рис. 16). Имеется возможность создавать новый фон в специальном редакторе.

Далее можно продемонстрировать учащимся создание простейшего собственного фона сцены.

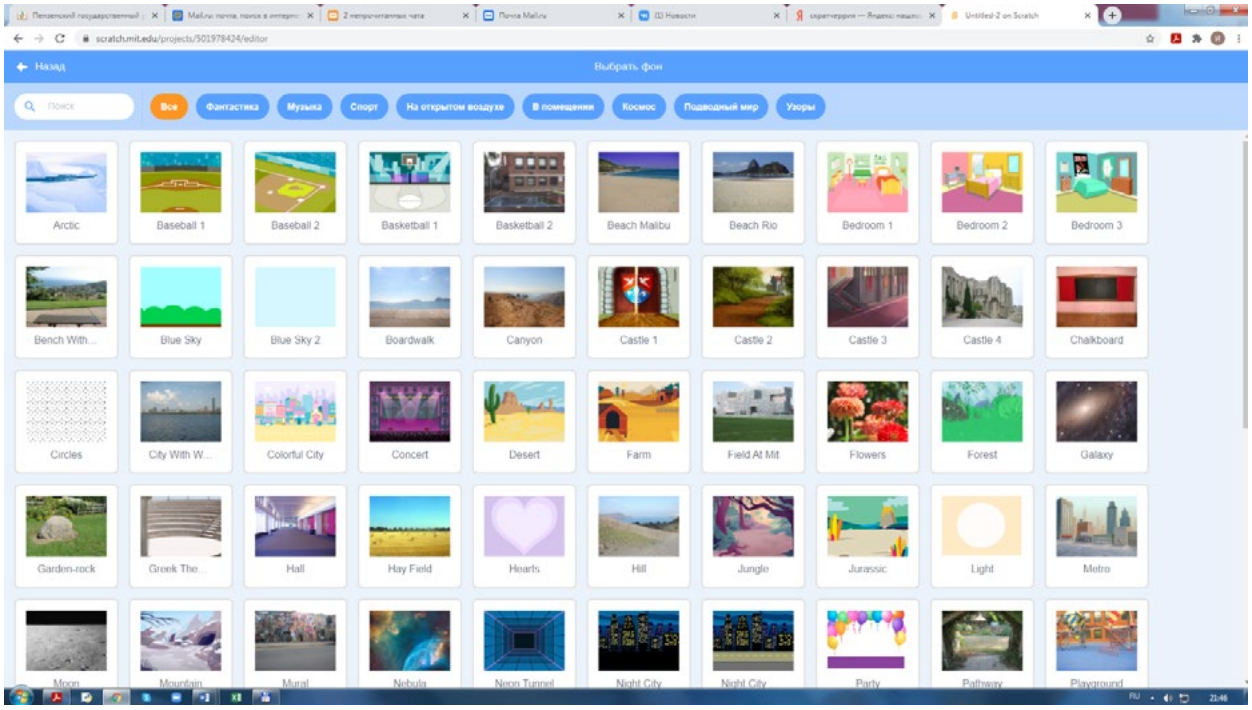

Рис. 16. Вид коллекции фонов

III. Этап проверки понимания и первичного закрепления — 5 мин.

Учитель предлагает учащимся вспомнить кратко новый материал: запуск среды Scratch, что такое сцена, что такое спрайт, как выбирается фон сцены, как выбирается спрайт.

IV. Этап контроля усвоения и коррекции ошибок —5 мин.

Учитель задаёт вопросы ученикам:

С какой средой программирования вы познакомились в ходе урока?

Какие компоненты среды Scratch усвоили на уроке?

V. Информация о домашнем задании, инструктаж по его выполнению — 3 мин.

VI. Этап рефлексии деятельности на уроке — 2 мин.

Учитель спрашивает учащихся об их впечатлениях от урока, что понравилось, а что было непонятно.

Лабораторная работа 1 «Знакомство со средой Scratch»

Теоретическая часть

Воспользоваться материалами урока 1.

Практическая часть

Цель работы: ознакомление со средой Scratch, изучение возможностей основных инструментов среды.

Ход лабораторной работы

Для загрузки среды Scratch нужно в браузере ввести https://scratch.mit.edu/.

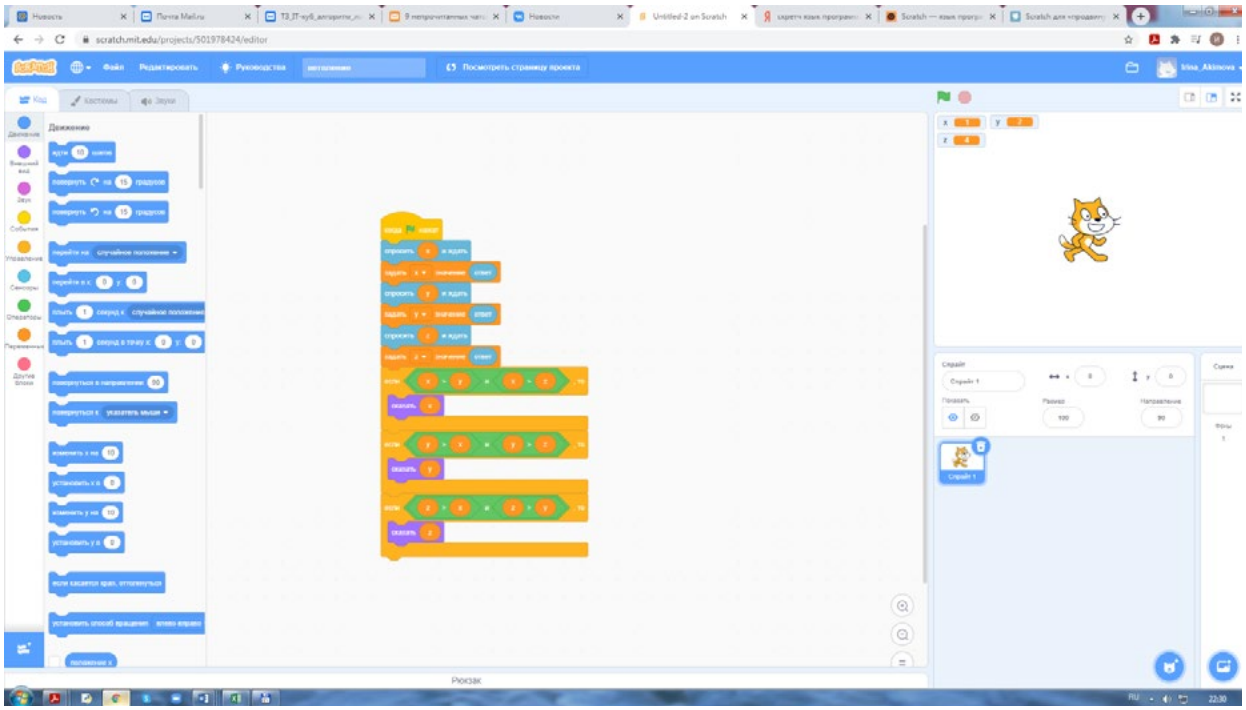

Рис. 17. Интерфейс среды Scratch

Программа на языке Scratch состоит из блоков команд, которые выполняет спрайт. Роль спрайта по умолчанию выполняет рыжий Кот, но есть возможность смены спрайта.

1. Откройте среду Scratch.
2. Выберите спрайт для работы, например Цыплёнок (рис. 18).

Рис. 18. Вид спрайта Цыплёнок

3. Выберите фон из коллекции фонов, например «Голубое небо» (рис. 19).

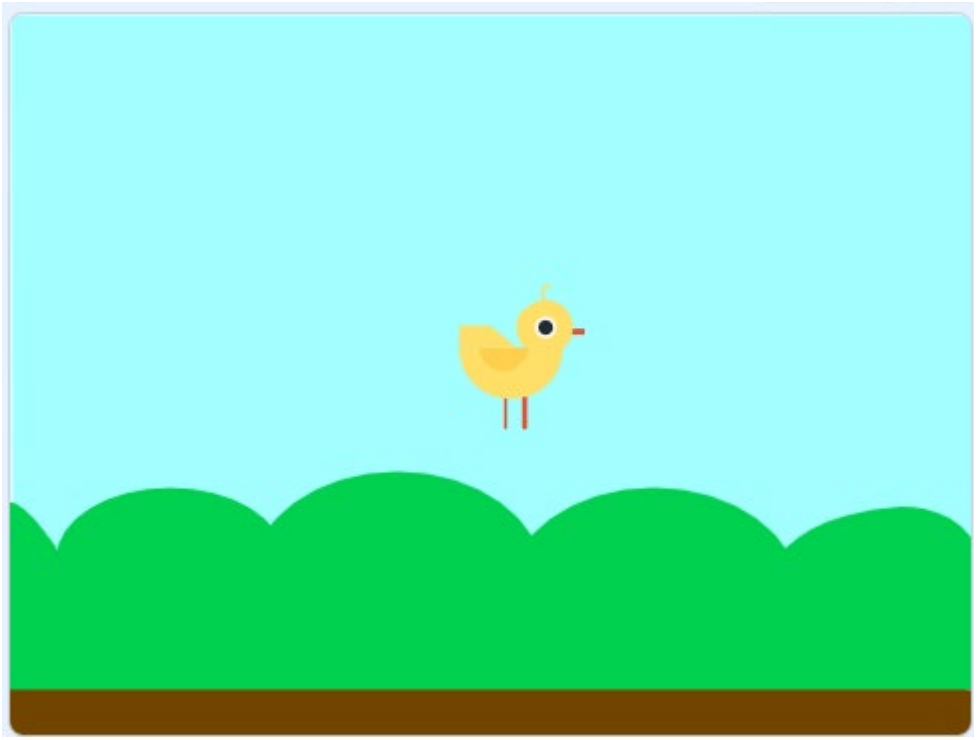

Рис. 19. Вид фона «Голубое небо»

4. Смените спрайт, удалите фон.
5. Выполните следующий скрипт (рис. 20).

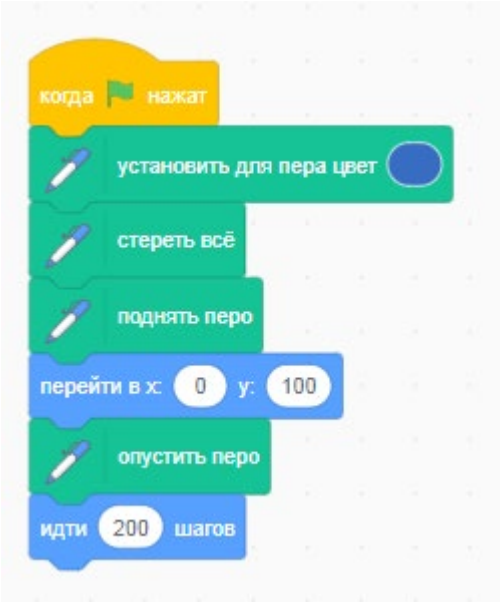

Рис. 20. Вид скрипта

6. Объясните, что выполняет скрипт.
7. Измените спрайт в графическом редакторе Scratch.

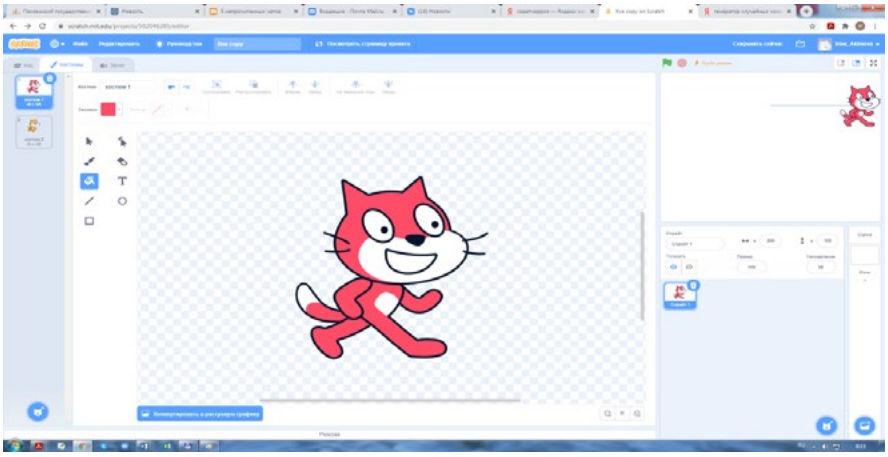

Рис. 21. Редактирование спрайта в графическом редакторе

8. Выполните скрипт, представленный на рисунке 22.

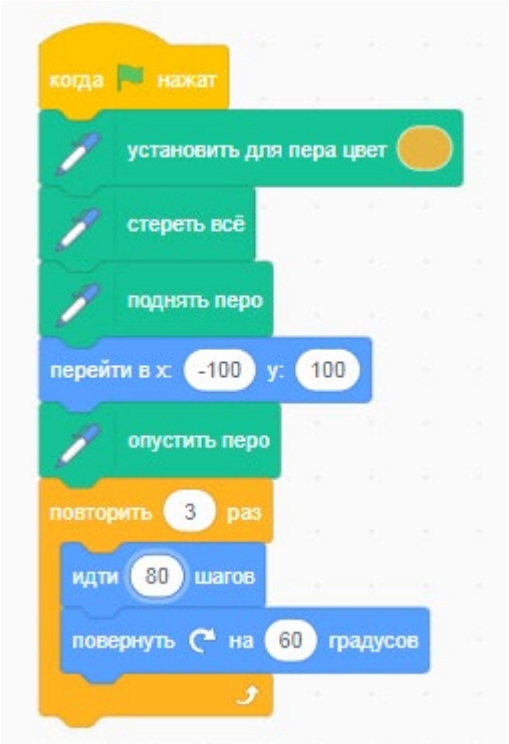

Рис. 22. Вид скрипта

9. Объясните, что выполняет скрипт.
10. Выполните следующий скрипт, размещённый на рисунке 23.

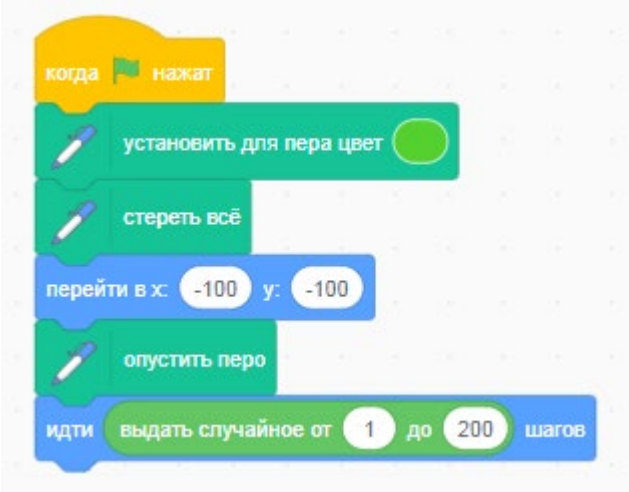

Рис. 23. Вид скрипта

11. Объясните, что выполняет скрипт.

Вывод: в процессе выполнения лабораторной работы вы получили представление о работе в среде Scratch.

Контрольные вопросы:

Как изменить вид спрайта в среде Scratch?

Можно ли редактировать фон в среде Scratch?

Какой вид графики можно создавать с помощью графического редактора в среде Scratch?

Тема 2. Линейные алгоритмы

Рекомендованное количество часов: 1 час — комбинированный урок, 1 час — лабораторная работа.

Планируемые результаты: знание основных приёмов разработки линейных алгоритмов в среде Scratch, умение решать задачи на линейные алгоритмы с использованием основных инструментов среды Scratch, навыки работы с линейными алгоритмами.

Урок 2

Уровень образования: основное общее.

Предмет: информатика.

Тема урока: «Линейные алгоритмы».

Класс: 8.

Тип урока: комбинированный.

Цель урока: знакомство учащихся с линейными алгоритмами и создание первых программ.

Время реализации: 1 академический час.

Ход урока

I. Этап постановки цели и задач урока, мотивации к учебной деятельности — 5 мин.

Деятельность учителя: объясняет особенности среды программирования Scratch при использовании линейных алгоритмов.

II. Этап изучения нового учебного материала — 25 мин.

Деятельность учителя: демонстрирует учащимся основы работы в среде Scratch.

Деятельность учеников: наблюдают за учителем, задают вопросы, отвечают на вопросы учителя.

Дидактические материалы

В левой части панели инструментов имеется закладка «Код». Если она активна, то в левой части рабочего стола будут располагаться палитра операторов среды Scratch. Они все упорядочены по типу выполняемых действий. Например, в закладке «Движение» расположены все операторы, предназначенные для управления движением: идти 10 шагов, повернуть на 15 градусов, перейти в x, y, повернуть в направлении 90 градусов, изменить x на 10 и др. (рис. 24).

Операторы раздела «Внешний вид» предназначены для изменения внешнего вида спрайта: изменить костюм на костюм 2, следующий костюм, переключить фон на фон 1, следующий фон, изменить размер на 10% и др. (рис. 25).

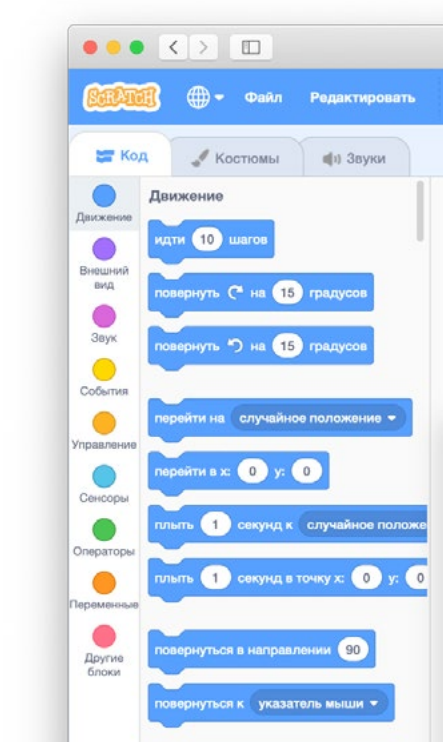

Рис. 24. Фрагмент палитры операторов «Движение»

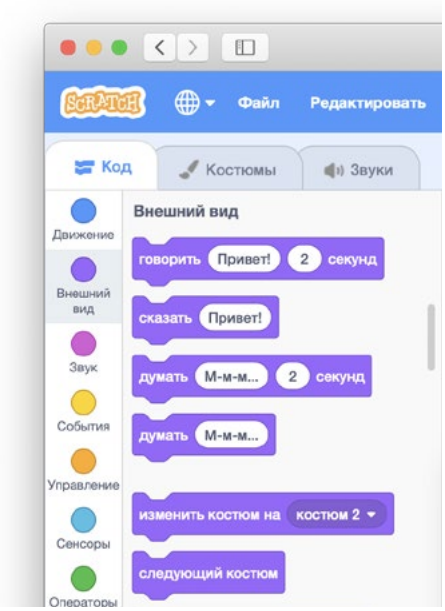

Рис. 25. Фрагмент палитры операторов «Внешний вид»

В разделе «Звук» расположены операторы для воспроизведения звуков: звук «Мяу», включить звук Мяу, остановить все звуки, изменить громкость и др. (рис. 26).

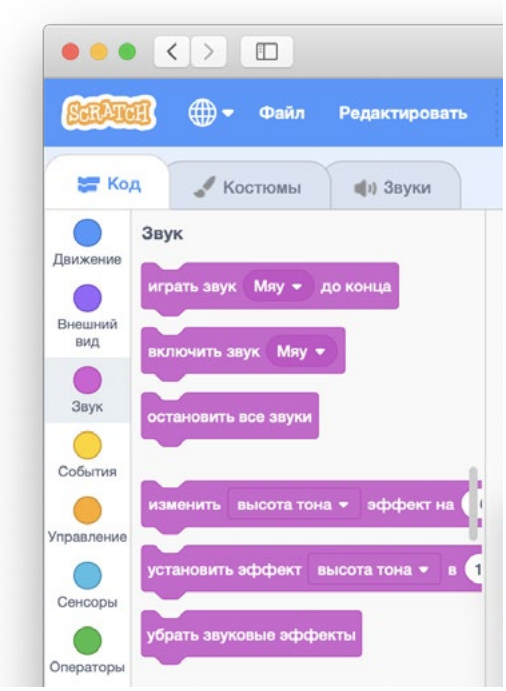

Рис. 26. Фрагмент палитры операторов «Звук»

Важным разделом операторов является раздел «События» (рис. 27). В нём расположены операторы обработки основных возможных событий, которые могут произойти со спрайтом: когда спрайт нажат, когда фон сменился, получить сообщение, передать сообщения и др. Но самым важным оператором является «когда нажат зелёный флажок». Обычно он ставится в начале программы и предназначен для запуска программы при нажатии на зелёный флажок, расположенный в левом верхнем углу поля.

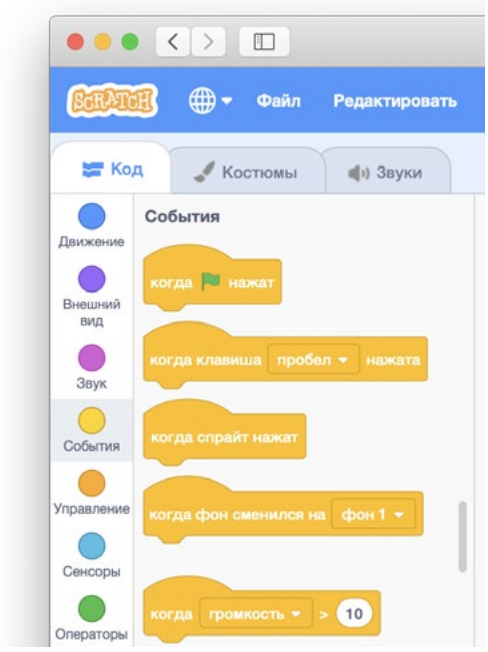

Рис. 27. Фрагмент палитры операторов «События»

Все операторы имеют углубления, благодаря которым они склеиваются в программу.

Линейный алгоритм — это вид алгоритма, который образуют команды, выполняемые однократно, одна за другой, в той последовательности, в которой они были изначально описаны. Линейная структура одна из самых простых в программировании. Примерная блок-схема линейного алгоритма представлена на рисунке 28.

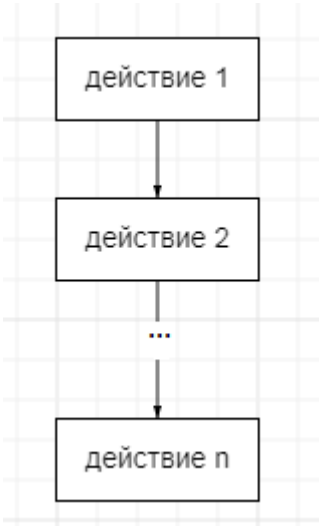

Рис. 28. Блок-схема линейного алгоритма

Следует показать учащимся, как составляется простейшая программа с использованием линейного алгоритма путём перетаскивания нужных операторов из палитры в поле программы. Покажем работу линейного алгоритма на примере движения Кота по полю. На рисунке 29 показан пример программы перемещения Кота по экрану с мяуканьем. Причём скорость Кота увеличивается.

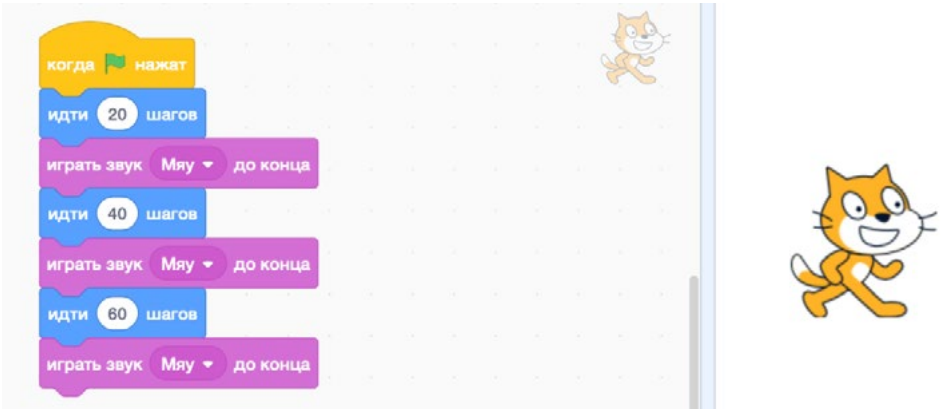

Рис. 29. Программа линейного алгоритма движения Кота

Затем нужно усложнить сюжет программы. Можно добавить другой спрайт, например Собачку, и предложить учащимся создать программу с сюжетом: Кот, убегающий от Собачки. Пример программы показан на рисунке 30.

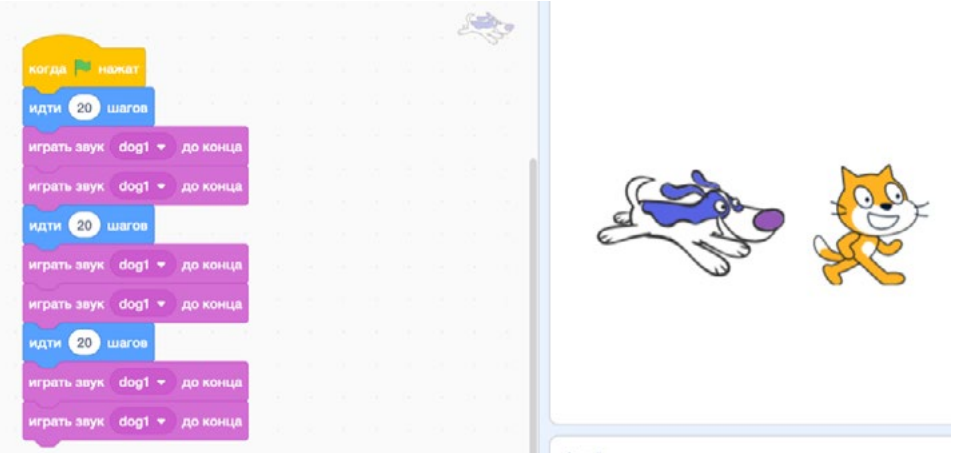

Рис. 30. Пример программы, Кота — убегающего от Собачки

III. Этап проверки понимания и первичного закрепления — 5 мин.

Учитель предлагает учащимся вспомнить кратко новый материал: запуск программы Scratch, линейный алгоритм, как задаётся движение Кота, как задаётся движение Собачки.

IV. Этап контроля усвоения и коррекции ошибок — 5 мин.

Учитель задаёт вопросы ученикам:

Почему Собачка догоняет Кота, если скорость Кота в программе возрастает?

Что нужно сделать в программе, чтобы Кот убежал от Собачки?

V. Информация о домашнем задании, инструктаж по его выполнению — 3 мин.

VI. Этап рефлексии деятельности на уроке — 2 мин.

Лабораторная работа 2 «Линейное программирование».

Теоретическая часть

Воспользоваться материалами урока 2.

Практическая часть

Цель работы: усвоить основные приёмы разработки линейных алгоритмов в среде Scratch, научиться решать задачи с использованием алгоритмов и основных инструментов среды Scratch, получить навыки работы с линейными алгоритмами.

Ход лабораторной работы

1. Откройте среду Scratch.
2. С помощью инструмента «Выбрать фон» выберите фон «Кирпичная стена» (рис. 31).

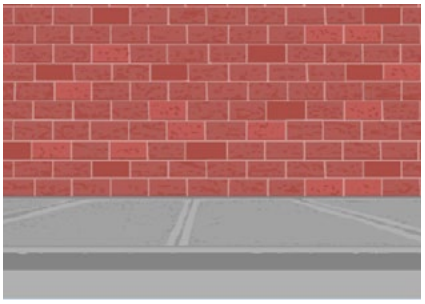

Рис. 31. Фон «Кирпичная стена»

3. Удалите спрайт Кот.

4. С помощью инструмента «Выбрать спрайт» выберите новый спрайт — Жук (рис. 32).

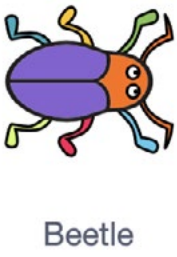

Рис. 32. Спрайт Жук

5. Жук автоматически появился на стене. Но его исходные размеры выглядят непропорционально по сравнению с кирпичной кладкой. Поэтому с помощью инструментов «Размер» и «Направление» уменьшите его размеры и направление движения, как показано на рисунке 33.

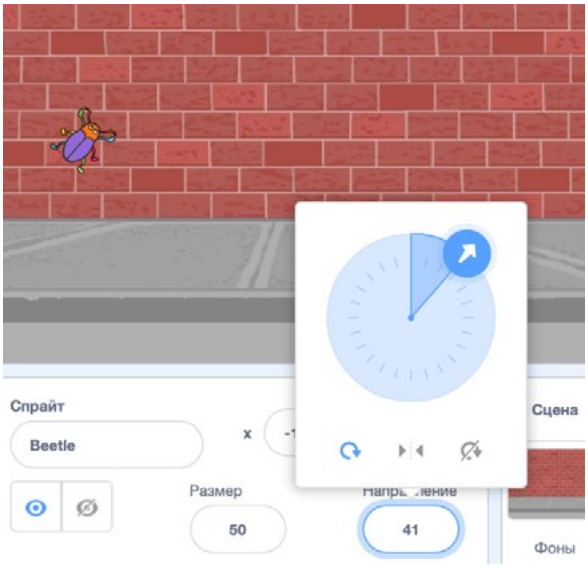

Рис. 33. Установка параметров Жука

6. Для имитации движения Жука нужно использовать конструкции «идти 10 шагов» + «играть звук до конца» или «идти 10 шагов» + «ждать ... секунд», как показано на рисунке 34. Это связано с особенностью человеческого зрения. В программу обязательно нужно включать какую-то задержку по времени, в противном случае движение просто сольётся.

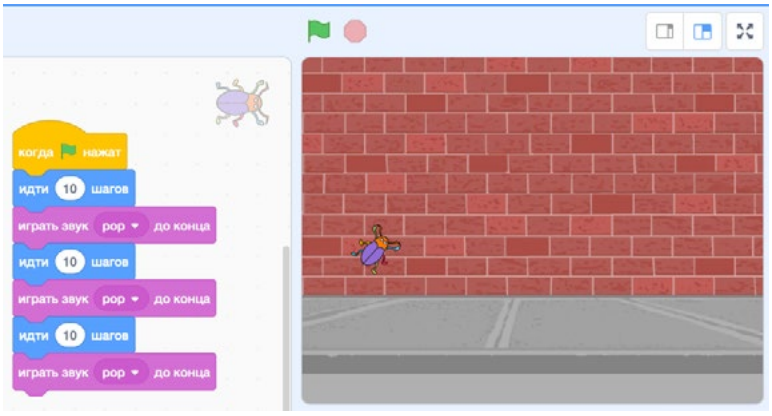

Рис. 34. Фрагмент программы движения жука

7. Для поворота Жука предназначен оператор «повернуть... на... градусов». Пример программы движения с поворотом показан на рисунке 35.

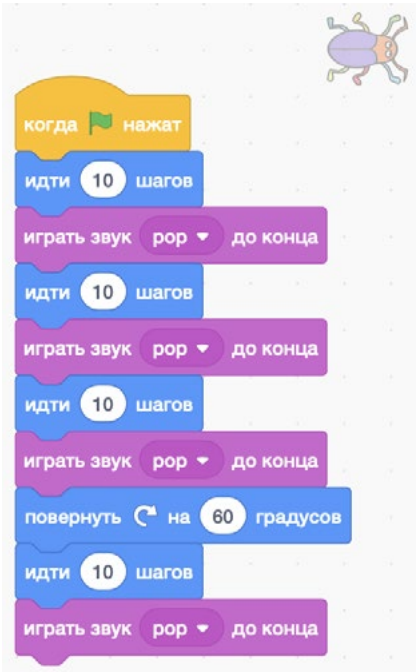

Рис. 35. Программа движения Жука с поворотом

8. Выполните в среде Scratch следующие задания.
1) Составьте программу движения Жука вверх до границы поля, затем возвращения назад.
2) Составьте программу движения двух Жуков навстречу друг другу. После столкновения они должны разбежаться в разные стороны.
Вывод: в ходе выполнения лабораторной работы вы получили представление об основах линейного программирования, операторах движения и поворота.

Контрольные вопросы:

1. Какой алгоритм называется линейным?
2. Какие можно привести примеры линейных алгоритмов?
3. Какие блоки вы использовали в лабораторной работе при создании линейных программ?

Тема 3. Циклические алгоритмы

Рекомендованное количество часов: 1 час — комбинированный урок, 3 часа — лабораторные работы.

Планируемые результаты: знание основных приёмов разработки циклических алгоритмов в среде Scratch, умение создавать проекты с использованием циклических алгоритмов и основных инструментов среды Scratch.

Урок 3

Уровень образования: основное общее.
Предмет: информатика.
Тема урока: «Циклические алгоритмы».
Класс: 8.

Тип урока: комбинированный.

Цель урока: знакомство учащихся с циклическими алгоритмами.

Время реализации: 1 академический час.

Ход урока

I. Этап постановки цели и задач урока, мотивации к учебной деятельности — 5 мин.

Деятельность учителя: объясняет особенности использования циклических алгоритмов в среде программирования Scratch.

II. Этап изучения нового учебного материала — 25 мин.

Деятельность учителя: демонстрирует учащимся основы работы в среде Scratch.

Деятельность учеников: наблюдают за учителем, задают вопросы, отвечают на вопросы учителя.

Дидактические материалы

В программировании циклические операторы используются тогда, когда требуется выполнить какое-то действие либо последовательность действий несколько раз. В среде программирования Scratch используются три циклических оператора: «повторить... раз», «повторять всегда», «повторять, пока не...» (рис. 36).

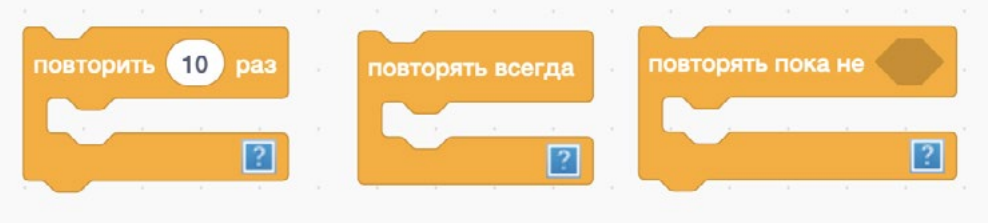

Рис. 36. Циклические операторы среды Scratch

Оператор «повторить... раз» является счётным оператором цикла. В верхнем окошке стоит цифра, обозначающая, сколько раз нужно выполнить действие либо последовательность действий. Её можно менять. В белой полосе располагаются операторы, которые требуется выполнить в цикле. Её называют телом цикла.

Пример программы перемещения Кота на 200 шагов показан на рисунке 37.

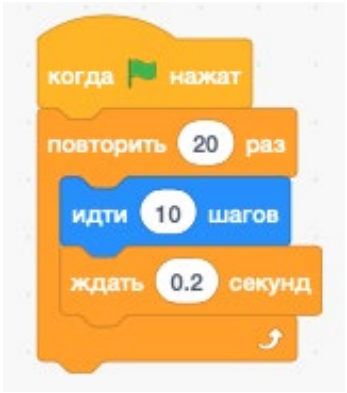

Рис. 37. Фрагмент программы перемещения Кота с помощью оператора цикла

В программировании часто используются алгоритмические конструкции «цикл в цикле», или, как их ещё называют, вложенные циклы. Рассмотрим их на примере движения Кота по кругу (рис. 38).

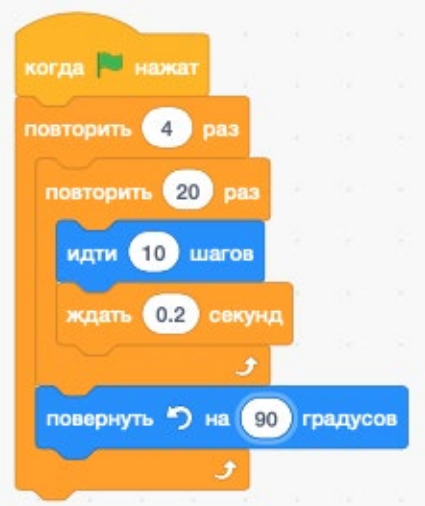

Рис. 38. Фрагмент программы движения Кота по кругу

В программе видим два цикла: «повторить 4 раза» и внутренний «повторить 20 раз». Обратите внимание, что оператор «повернуть на 90 градусов» стоит после внутреннего цикла. Данная конструкция работает следующим образом. Сначала выполняются операторы, стоящие во внутреннем цикле, т. е. Кот делает 20 шагов с задержкой времени 0,2 секунды. Затем выполняется оператор, стоящий после внутреннего цикла, т. е. Кот делает поворот на 90 градусов. Потом опять 20 раз выполняются операторы внутреннего цикла, затем Кот делает поворот на 90 градусов, и так 4 раза по внешнему циклу.

Если заранее неизвестно, сколько раз требуется повторение цикла, то целесообразно использовать оператор цикла «повторять всегда». На рисунке 38 приведён фрагмент программы движения Кота, пока он не покинет сцену.

В Scratch имеется оператор «если касается края, оттолкнуться», который позволяет создавать интересные динамические программы. На рисунке 39 приведён фрагмент программы движения Кота, который будет отскакивать от края сцены и менять траекторию движения. Всё это он будет выполнять до тех пор, пока не остановим программу нажатием на красный кружок, который стоит рядом с флажком запуска программы. Программа будет более интересной, если первоначально направить Кота под некоторым углом к горизонту.

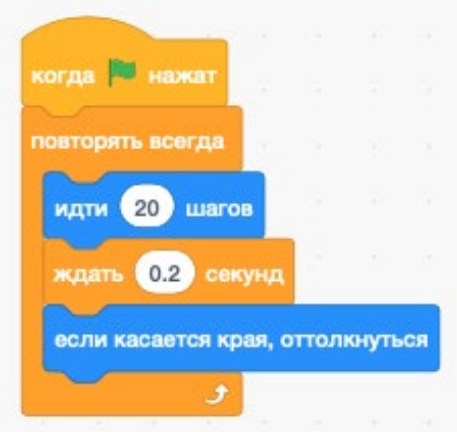

Рис. 39. Фрагмент программы движения Кота, который будет отскакивать от края сены и менять траекторию движения

Циклический оператор «повторять пока не...» работает аналогичным образом до тех пор, пока не возникнет событие, обозначенное в окне. Перечень возможных событий находится во вкладке «Сенсоры». Устанавливаем событие так же — перетаскиванием мыши в окно. На рисунке 40 приведён фрагмент программы движения Кота по сцене, пока по нему не щёлкнуть мышью.

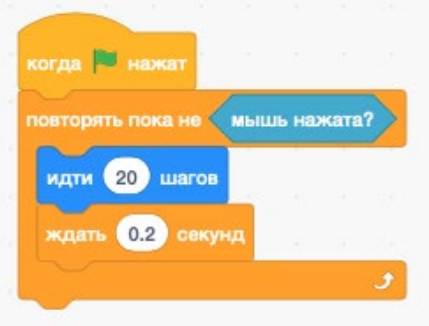

Рис. 40. Фрагмент программы движения Кота, пока по нему не щёлкнуть мышью

III. Этап проверки понимания и первичного закрепления — 5 мин.

Учитель предлагает учащимся вспомнить кратко новый материал: циклические операторы, используемые в Scratch.

IV. Этап контроля усвоения и коррекции ошибок — 5 мин.

Учитель задаёт вопросы ученикам:

1) Как работают циклические операторы Scratch?

2) Каковы особенности циклических операторов Scratch?

V. Информация о домашнем задании, инструктаж по его выполнению — 3 мин.

VI. Этап рефлексии деятельности на уроке — 2 мин.

Учитель спрашивает учащихся об их впечатлениях от урока, что понравилось, что было непонятно.

Лабораторная работа 3 «Циклический алгоритм “повторить ... раз”».

Теоретическая часть

Воспользоваться материалами урока 3.

Практическая часть

Цель работы: усвоить основные приёмы разработки циклических алгоритмов в среде Scratch, научиться создавать проекты с использованием циклических алгоритмов и основных инструментов среды Scratch.

Ход лабораторной работы

1. Откройте среду Scratch.
2. С помощью инструмента «Выбрать фон» выберите фон “Кирпичная стена” (рис. 41).

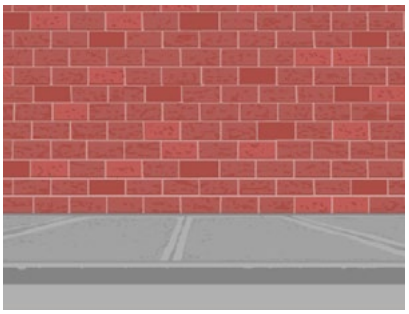

Рис. 41. Фон «Кирпичная стена»

3. Удалите спрайт Кот.
4. С помощью инструмента «Выбрать спрайт» выберите новый спрайт — Жук (рис. 42).

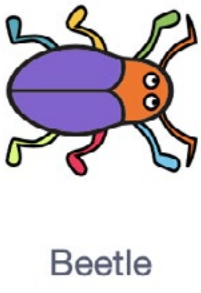

Рис. 42. Спрайт Жук

5. Уменьшите размер Жука, измените направление его движения, как показано на рисунке 43.

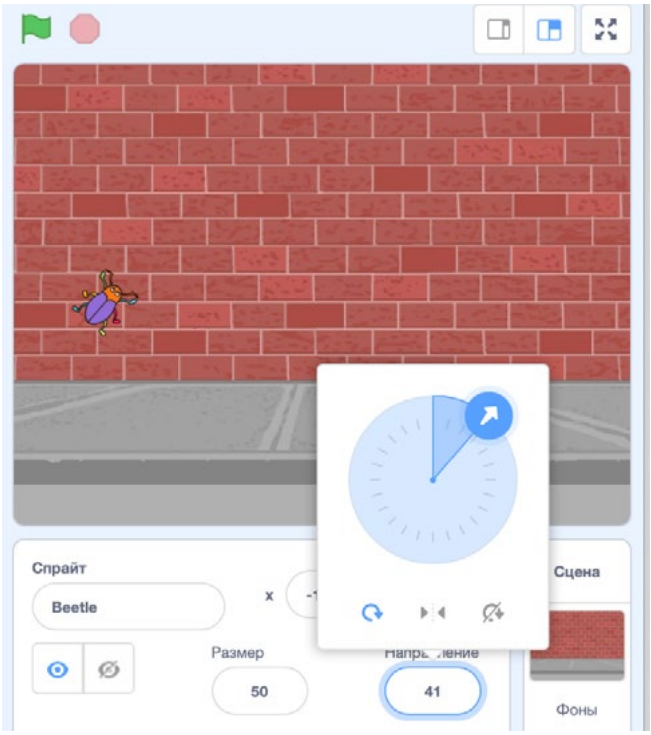

Рис. 43. Установка параметров Жука

6. Создайте программу движения Жука по квадрату с имитацией звука.
7. Измените программу таким образом, чтобы Жук двигался по траектории, напоминающей восьмёрку.

Вывод: в ходе выполнения лабораторной работы вы получили представление о циклическом алгоритме «повторить... раз».

Контрольные вопросы:

1. Как определить циклический алгоритм «повторить... раз»?
2. Что представляет собой вложенный цикл?
3. Какие блоки вы использовали в лабораторной работе при создании программы?

Лабораторная работа 4 «Циклический алгоритм “повторять всегда”».

Теоретическая часть

Воспользоваться материалами урока 3.

Практическая часть

Цель работы: усвоить основные приёмы разработки циклических алгоритмов в среде Scratch, научиться создавать проекты с использованием циклических алгоритмов и основных инструментов среды Scratch.

Ход лабораторной работы

1. Откройте среду Scratch.
2. С помощью инструмента «Выбрать фон» выберите фон «Баскетбольная площадка» (рис. 44).

Рис. 44. Фон «Баскетбольная площадка»

3. Удалите спрайт Кот.
4. С помощью инструмента «Выбрать спрайт» выберите новый спрайт — Мячик (рис. 45).

Рис. 45. Спрайт Мячик

5. Уменьшите размер Мячика, измените направление его движения, как показано на рисунке 46.

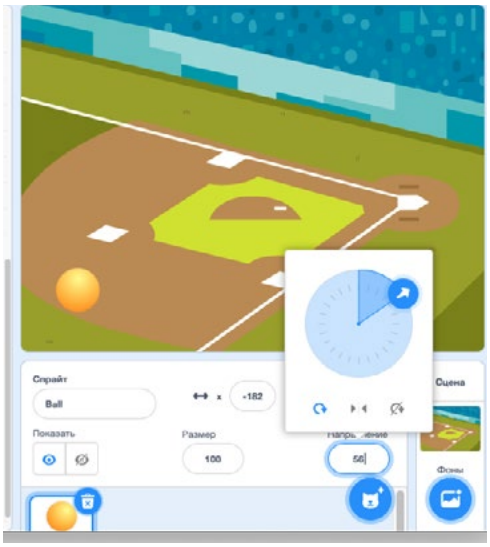

Рис. 46. Установка параметров мячика

6. С помощью циклического оператора «повторять всегда» создайте программу полёта Мячика с отскоками от края сцены.

7. Добавьте второй спрайт Мячик, но меньшего размера. В окне кода создайте вторую программу движения для второго Мячика и запустите их одновременно.

Вывод: в ходе выполнения лабораторной работы вы получили представление о циклическом алгоритме «повторять всегда».

Контрольные вопросы:

1. Как определить циклический алгоритм «повторять всегда»?
2. Что представляет собой вложенный цикл?
3. Какие блоки вы использовали в лабораторной работе при создании программы?

Лабораторная работа 5 «Циклический алгоритм “повторять пока не ...”».

Теоретическая часть

Воспользоваться материалами урока 3.

Практическая часть

Цель работы: усвоить основные приёмы разработки циклических алгоритмов в среде Scratch, научиться создавать проекты с использованием циклических алгоритмов и основных инструментов среды Scratch.

Ход лабораторной работы

1. Откройте среду Scratch.
2. Оставьте исходный фон и спрайт Кот.
3. С помощью циклического оператора «повторять пока не...» создайте программу движения Кота: он будет поворачивать на 90 градусов, если наткнётся на указатель мыши. Кот должен остановиться, если щёлкнуть по нему указателем мыши. При разработке программы следует использовать события «Касается указателя мыши» и «Мышь нажата».

Вывод: в ходе выполнения лабораторной работы вы получили представление о циклическом алгоритме «повторять пока не...».

Контрольные вопросы:

1. Как определить циклический алгоритм «повторять пока не...»?
2. Что представляет собой вложенный цикл?
3. Какие блоки вы использовали в лабораторной работе при создании программы?

Тема 4. Работа с переменными

Рекомендованное количество часов: 1 час — комбинированный урок, 2 часа — лабораторные работы.

Планируемые результаты: знание основных приёмов использования переменных в среде Scratch, умение создавать проекты с использованием переменных и основных инструментов среды Scratch.

Урок 4

Уровень образования: основное общее.

Предмет: информатика.

Тема урока: «Работа с переменными».

Класс: 8.

Тип урока: комбинированный.

Цель урока: знакомство учащихся с переменными.

Время реализации: 1 академический час.

Ход урока

I. Этап постановки цели и задач урока, мотивации к учебной деятельности — 5 мин.

Деятельность учителя: объясняет особенности использования переменных в среде программирования Scratch.

II. Этап изучения нового учебного материала — 25 мин.

Деятельность учителя: демонстрирует учащимся основы работы с переменными в среде Scratch.

Деятельность учеников: наблюдают за учителем, задают вопросы, отвечают на вопросы учителя.

Дидактические материалы

Переменная представляет собой область памяти компьютера, которая имеет название и хранит внутри себя какие-либо данные.

Переменная в среде Scratch бывает простая (хранит одно значение), также может представлять собой список для хранения больше одного значения.

Среда Scratch предоставляет возможность вводить в проект как простые переменные, так и массивы. Для этого необходимо обратиться к разделу «Переменные». Для создания переменной необходимо выбрать команду «Создать переменную» в опциях «Переменные». В появившемся диалоговом окне указываем имя переменной (рис. 47).

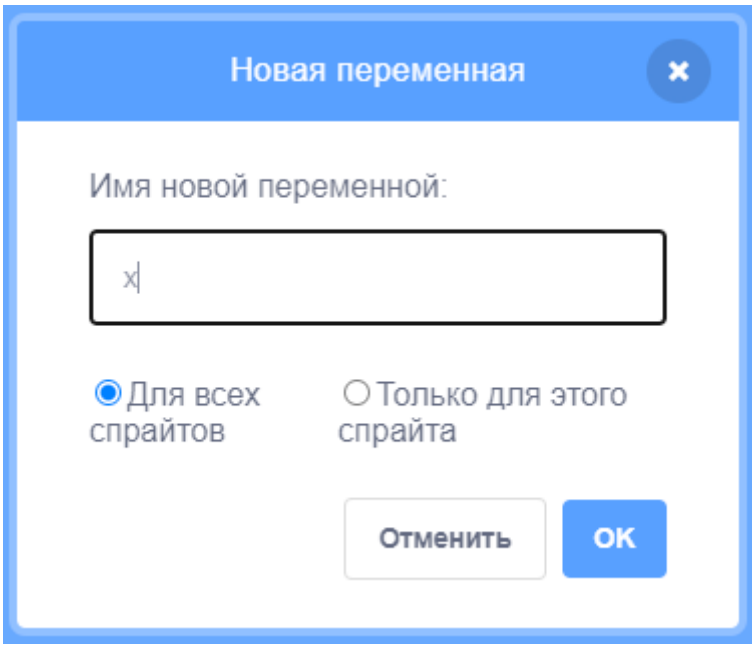

Рис. 47. Диалоговое окно для создания переменной

После задания имени переменной она появляется в проекте (рис. 48). Автоматически рядом с ней появится флажок. В этом случае окно со значением переменной появится на сцене. Если флажок убрать, то окно исчезнет.

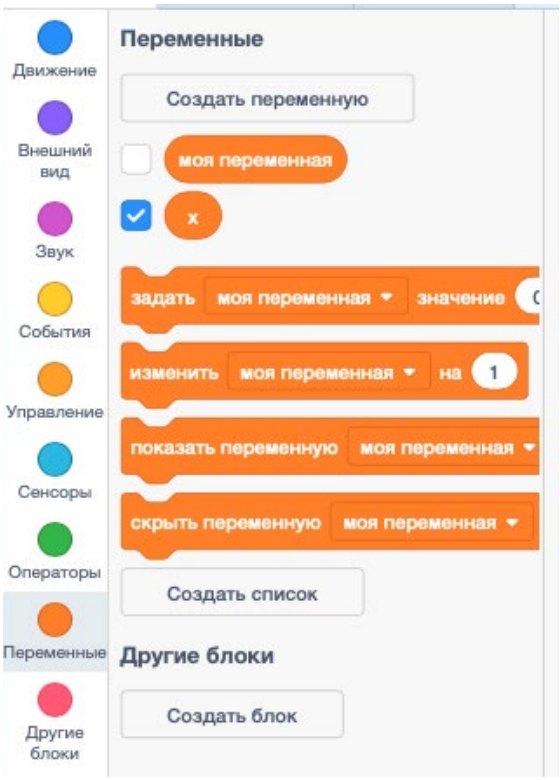

Рис. 48. Вид созданной переменной в разделе «Переменные»

Окно переменной отображается в левом верхнем углу сцены, но можно перемещать его в любое место с помощью мыши. Окно можно настраивать. Для этого нужно щёлкнуть на нём правой кнопкой мыши. Появится меню настройки окна переменной (рис. 49).

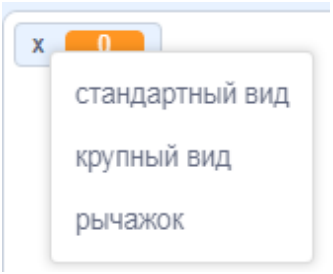

Рис. 49. Меню настройки окна переменной

После создания переменной становятся доступны операторы для работы с ней. Они расположены на панели «Переменные» ниже самой переменной (рис. 48). Например, для задания значения переменной используется оператор «Задать — моя переменная — значение». Его нужно переместить с помощью мыши в поле программы. Затем нужно щёлкнуть мышью в окне «Моя переменная» и выбрать «x». Потом нужно ввести с клавиатуры её значение в окно «Значение» (рис. 50).

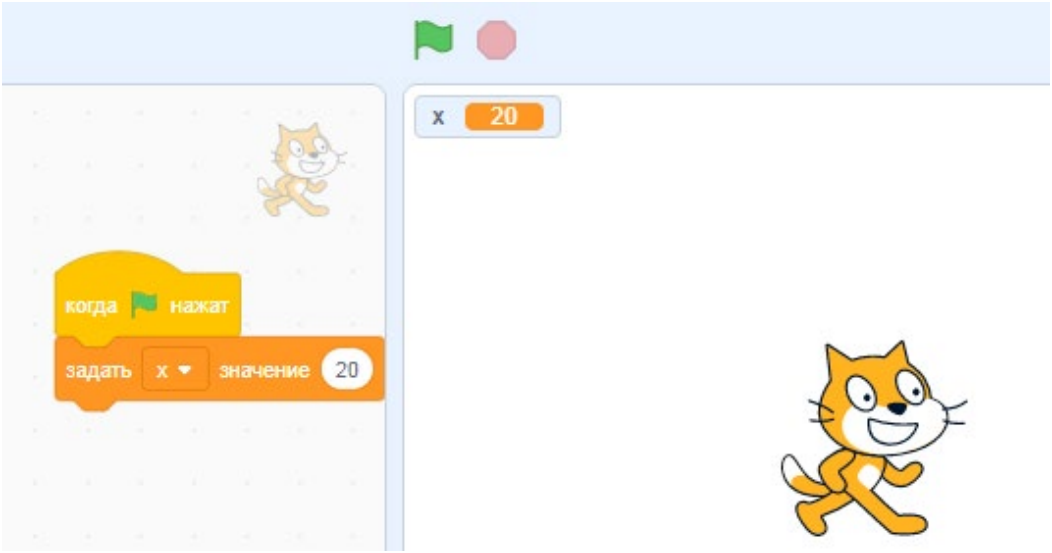

Рис. 50. Пример задания значения переменной

Рассмотрим работу с основными блоками из раздела «Переменные».

Оператор «Изменить — моя переменная — значение» используется для изменения переменной на заданное значение. Причём вводимое значение может быть как целым числом, так и дробным.

Оператор «Показать переменную...» используется для того, чтобы показывать значение переменной на сцене. Блок «скрыть переменную...» используется, чтобы перестать показывать значение переменной на сцене. По умолчанию переменная и её значение отображаются на сцене, но с помощью данных операторов её можно скрывать или удалять со сцены уже в ходе выполнения программы.

Для ввода чисел с клавиатуры удобно использовать конструкцию, показанную на рисунке 51. В конструкции используется последовательность операторов «Спросить — Введите число — и ждать» и «Задать — x — значение — ответ». Первый оператор синего цвета, поэтому он находится в разделе «Сенсоры» тоже синего цвета. Вводится он следующим образом. Сначала нужно переместить мышью в поле программы оператор «Спросить — Как тебя зовут? — и ждать». Затем в белом поле заменить надпись на «Введите число». После этого нужно переместить из раздела «Переменные» оператор «Задать — моя переменная — значение», выбрать в поле «моя переменная» имя «x» и переместить мышью из раздела «Сенсоры» оператор «ответ» в поле «значение».

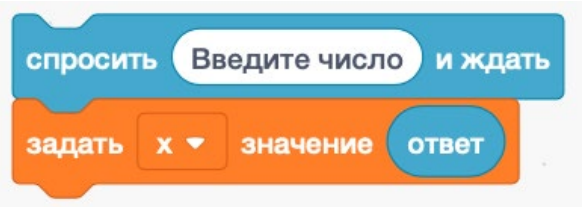

Рис. 51. Конструкция для ввода чисел

Работает эта конструкция следующим образом. Оператор «спросить — Введите число — и ждать» формирует около спрайта запрос «Введите число» и открывает поле ввода числа. Программа приостанавливает свою работу и переходит в режим ожидания ввода числа (рис. 52). Нужно набрать с клавиатуры число, например 15 и нажать галочку около поля ввода либо клавишу Enter клавиатуры. После ввода значения оно будет оператором «Задать — x — значение — ответ» присвоено переменной x.

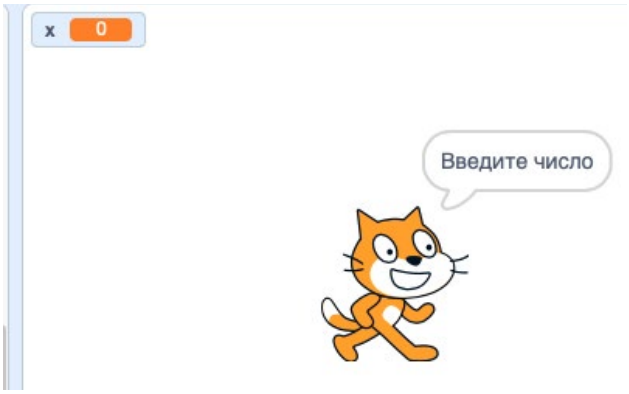

Рис. 52. Выполнение оператора «Спросить — Введите число — и ждать»

Для вывода значений переменных используется оператор «сказать». Он находится в разделе «Внешний вид». После перемещения его в программу нужно в белое поле переместить из раздела «Сенсоры» оператор «ответ».

На рисунке 53 показан фрагмент программы ввода числа.

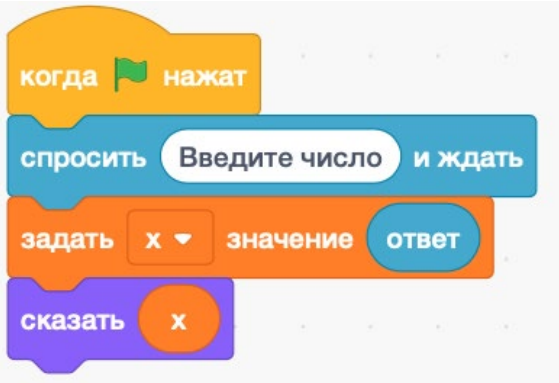

Рис. 53. Фрагмент программы ввода и вывода числа

На рисунке 54 показан пример программы, которая запрашивает ввод чисел x и y, затем выводит результат суммы чисел. В примере используется оператор «x + y», который находится в разделе «Операторы».

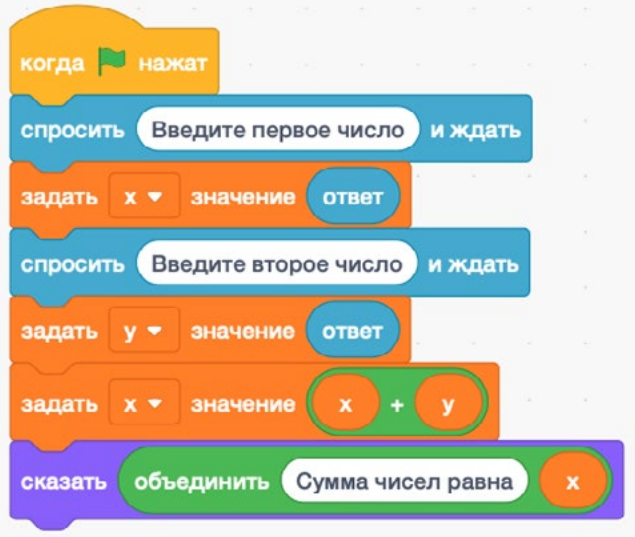

Рис. 54. Фрагмент программы нахождения суммы двух введённых чисел

III. Этап проверки понимания и первичного закрепления — 5 мин.

Учитель предлагает учащимся вспомнить кратко новый материал: как объявляются переменные Scratch, как вводятся и выводятся значения переменных.

IV. Этап контроля усвоения и коррекции ошибок — 5 мин.

Учитель задаёт вопросы ученикам:

1) Какие математические операции используются в Scratch?

2) Как изменяются значения переменных в Scratch?

V. Информация о домашнем задании, инструктаж по его выполнению — 3 мин.

VI. Этап рефлексии деятельности на уроке — 2 мин.

Учитель спрашивает учащихся об их впечатлениях от урока, что понравилось, а что было непонятно.

Лабораторная работа 6 «Работа с переменными»

Теоретическая часть

Воспользоваться материалами урока 4.

Практическая часть

Цель работы: знакомство с особенностями работы с переменными в среде Scratch.

Ход лабораторной работы

1. Откройте среду Scratch.

2. С помощью инструмента «Выбрать фон» выберите фон «Театральная сцена» (рис. 55).

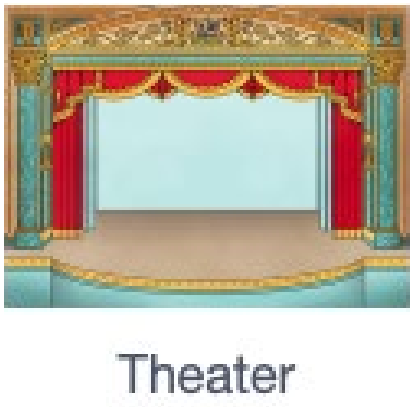

Рис. 55. Фон «Театральная сцена»

3. Удалите спрайт Кот.

4. С помощью инструмента «Выбрать спрайт» выберите новый спрайт — Человечек (рис. 56).

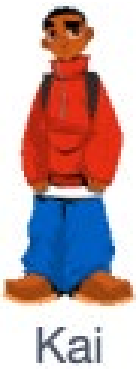

Рис. 56. Спрайт Человечек

5. В закладке «Переменные» выберите команду «Создать переменную» и создайте переменную a.

6. Аналогично создайте переменную b.

7. В разделе «Переменные» нужно рядом с переменными поставить флажки. В этом случае на поле сцены появятся окошки, в которых будут отображаться значения переменных.

8. Создайте программу деления числа x на число y, которые вводятся с клавиатуры, ответ запишите в переменную z и выведите на экран. Текст программы показан на рисунке 57.

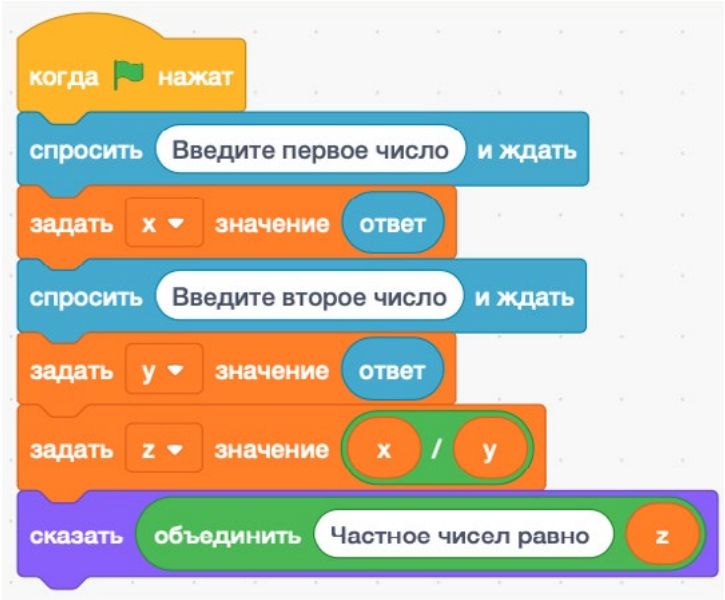

Рис. 57. Текст программы деления числа a на число b

9. Запустите программу.

10. Введите в переменную x значение 35, в переменную y значение 7. Получим ответ в виде рис. 58.

Рис. 58. Ответ к заданию

11. Выполните самостоятельно задание: введите с клавиатуры числа x и y.
Вычислите значение 2x — 3y.
Вывод: в ходе выполнения лабораторной работы вы получили представление о работе с переменными.

Контрольные вопросы:

1. Какие конструкции используются для ввода переменных?
2. Где находится поле вывода ответа?
3. Обязательно ли нужно выводить на сцену значения переменных?

Лабораторная работа 7 «Работа с переменными»

Теоретическая часть

Воспользоваться материалами урока 4 и лабораторной работы 6.

Существенным недостатком программы, созданной в ходе предыдущей лабораторной работы, было то, что персонаж выводил сообщения только текстом. Хотя среда Scratch позволяет озвучивать сообщения. Для этого необходимо установить дополнение кода. Чтобы установить новое дополнение, нужно нажать значок  в нижнем левом углу кода программы. Затем выбрать дополнение «Текст в речь». В панели «Код» появится новая закладка «Текст в речь» с операторами «сказать», «установить голос» и «установить язык» (рис. 59).

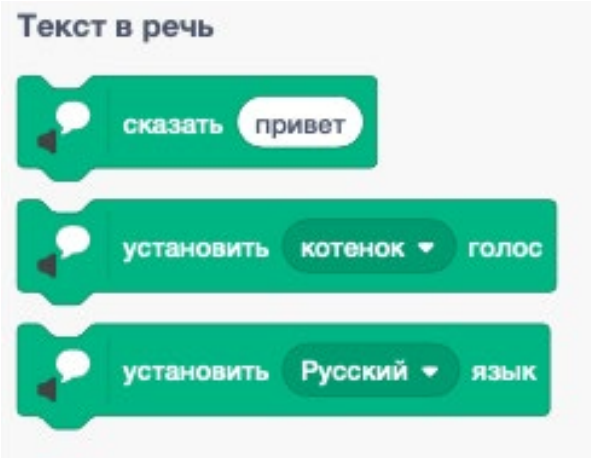

Рис. 59. Операторы в закладке «Текст в речь»

Оператор «сказать» может озвучивать как текст, введённый в белое поле, так и значение переменной, введённой в него. Оператор «установить голос» позволяет менять голос спрайта. Так, для персонажа Человечек больше подходит голос «тенор».

В ходе выполнения лабораторной работы потребуется извлечь квадратный корень. Для ввода оператора квадратного корня нужно переместить в программу оператор «модуль — от». Он находится на панели «Операторы». Затем нужно щёлкнуть мышью в окне «Модуль». В открывшемся окне меню выбора функций следует выбрать «Квадратный корень» и ввести нужное выражение, как на рисунке 60.

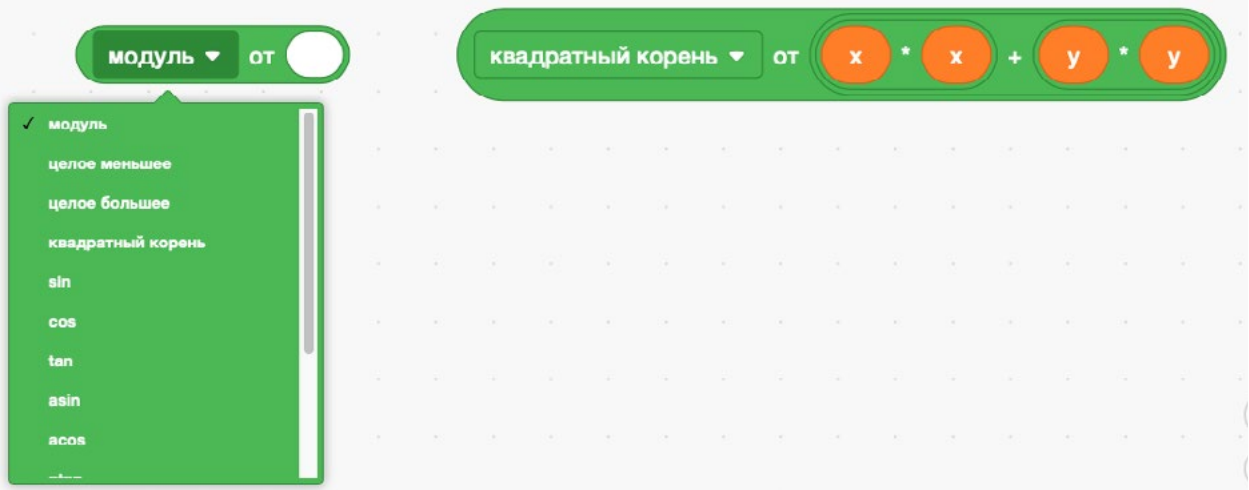

Рис. 60. Оператор извлечения квадратного корня

Практическая часть

Цель работы: знакомство с особенностями работы с переменными в среде Scratch.

Ход лабораторной работы

1. Откройте среду Scratch.
2. Настройте сцену и спрайт аналогично предыдущей лабораторной работе.
3. С помощью инструмента «Выбрать фон» выберите фон «Театральная сцена» (рис. 54).
4. Создайте и озвучьте программу нахождения длины гипотенузы прямоугольного треугольника по двум катетам, длины которых вводятся с клавиатуры.
5. Для озвучивания программы можно использовать конструкции, показанные на рисунке 61.

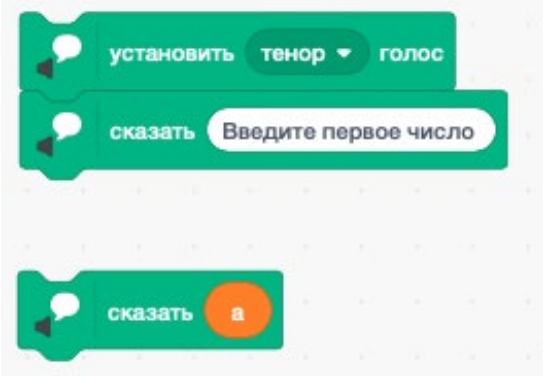

Рис. 61. Фрагмент программы озвучивания

Вывод: в ходе выполнения лабораторной работы вы получили представление о работе с переменными.

Контрольные вопросы:

1. Какие конструкции используются для ввода переменных?
2. Какие конструкции используются для озвучивания переменных?
3. Можно ли одновременно озвучивать несколько персонажей?

Тема 5 «Условные алгоритмы»

Рекомендованное количество часов: 1 час — комбинированный урок, 2 часа — лабораторные работы.

Планируемые результаты: знание основных приёмов использования переменных в среде Scratch, умение создавать проекты с использованием условных алгоритмов и основных инструментов среды Scratch.

Урок 5

Уровень образования: основное общее.

Предмет: информатика.

Тема урока: «Условные алгоритмы».

Класс: 8.

Тип урока: комбинированный.

Цель урока: знакомство учащихся с условными алгоритмами.

Время реализации: 1 академический час.

Ход урока

I. Этап постановки цели и задач урока, мотивации к учебной деятельности — 5 мин.

Деятельность учителя: объясняет особенности использования условных алгоритмов в среде программирования Scratch.

II. Этап изучения нового учебного материала — 25 мин.

Деятельность учителя: демонстрирует учащимся основы работы в среде Scratch.

Деятельность учеников: наблюдают за учителем, задают вопросы, отвечают на вопросы учителя.

Дидактические материалы

Условный алгоритм — это алгоритм, порядок выполнения действий в котором зависит от выполнения (или невыполнения) какого-либо условия.

Условный, или, как ещё его называют, разветвляющийся алгоритм, обеспечивает выбор одного из двух альтернативных путей в зависимости от истинности условия.

В этом случае логика принятия решения может быть записана следующим образом:

ЕСЛИ <условие> ТО <действия 1> ИНАЧЕ <действия 2> ВСЁ

Ветвление может быть реализовано в полной или в сокращённой форме (рис. 62, 63).

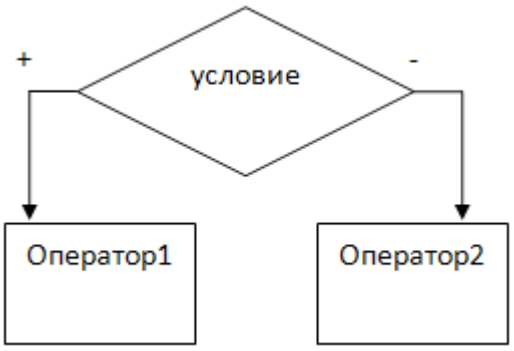

Рис. 62. Полная форма условного алгоритма

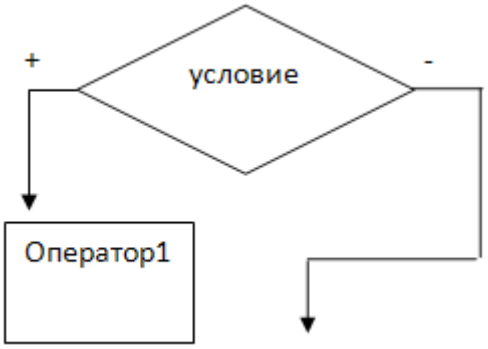

Рис. 63. Сокращённая форма условного алгоритма

Условные алгоритмы реализуются практически в любом языке программирования. В среде Scratch для реализации разветвляющихся алгоритмов можно использовать два вида блоков, которые представлены в разделе «Управление».

Рис. 64. Условные операторы среды Scratch

Рассмотрим, как работают условные алгоритмы, на примере задачи ввода двух чисел с клавиатуры и определения максимального числа. Фрагмент программы имеет вид, как на рисунке 65.

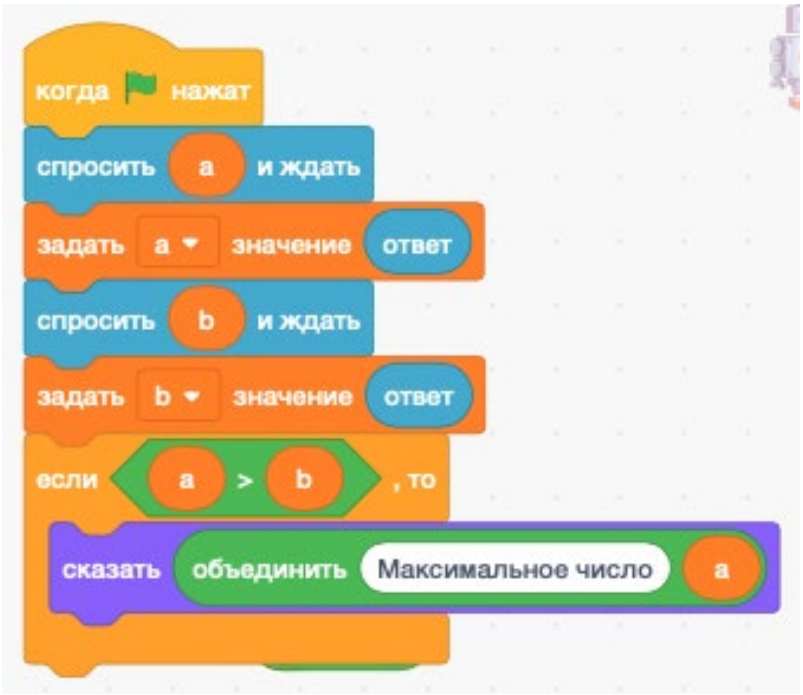

Рис. 65. Фрагмент программы ввода и сравнения двух чисел

Первые две строчки программы служат для ввода числа с клавиатуры и записи его в переменную a. Третья и четвёртая строчки, соответственно, для ввода и записи числа в переменную b. Далее проверяется условие «a > b». Если условие выполняется, то выводится сообщение «Максимальное число» и значение числа a. В случае невыполнения условия никакого сообщения выводиться не будет. Для того чтобы выполнялось какое-то действие в случае невыполнения условия, нужно использовать блок полной формы (рис. 62). Фрагмент такой программы будет иметь вид, как на рисунке 66.

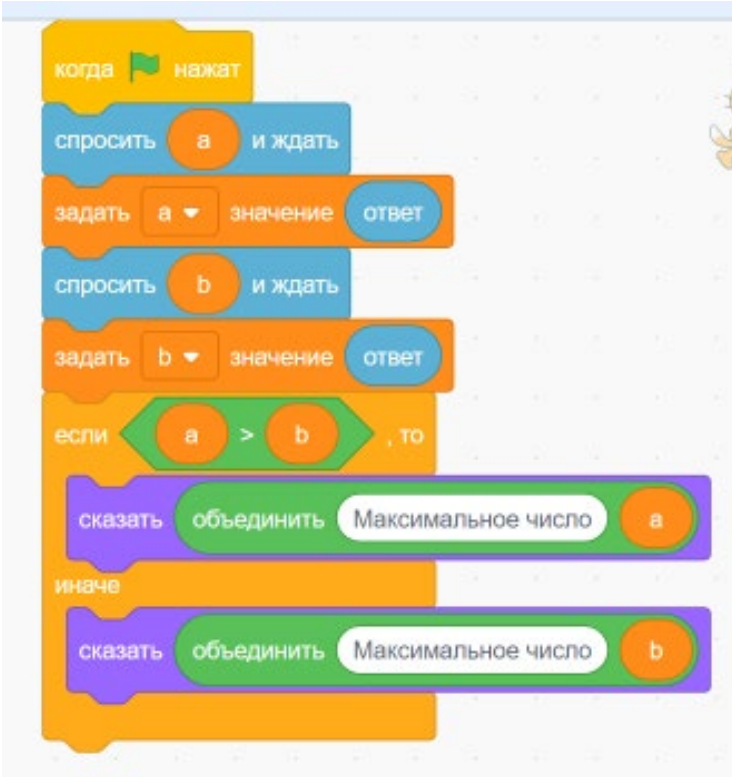

Рис. 66. Фрагмент программы ввода и сравнения двух чисел

В программировании часто используются конструкции вложенных условий, например «если — условие — то, иначе — если — то», что означает: если условие выполняется, то переходим к проверке второго условия, вложенного в первое. Покажем, как это работает, на том же примере (рис. 67).

Рис. 67. Фрагмент программы с вложенными условиями

Программа работает следующим образом. После ввода чисел проверяется условие «a > b», если условие выполняется, то выводится сообщение «Максимальное число» и значение числа a. Если условие не выполняется, то проверяется условие «a < b». Если оно выполняется, то выводится сообщение «Максимальное число» и значение числа b. Если это условие не выполняется, то выводится сообщение «Числа равны».

С целью закрепления учебного материала можно предложить учащимся немного изменить программу, чтобы она выводила числа по возрастанию.

III. Этап проверки понимания и первичного закрепления — 5 мин.

Учитель предлагает учащимся вспомнить кратко новый материал: как работают условные алгоритмы, чем они отличаются друг от друга.

IV. Этап контроля усвоения и коррекции ошибок — 5 мин.

Учитель задает вопросы ученикам:

Какие условные алгоритмы используются в Scratch?

Что такое вложенное условие?

V. Информация о домашнем задании, инструктаж по его выполнению — 3 мин.

VI. Этап рефлексии деятельности на уроке — 2 мин.

Учитель спрашивает учащихся об их впечатлениях от урока, что понравилось, а что было непонятно.

Лабораторная работа 8 «Условные алгоритмы».

Теоретическая часть

Воспользоваться материалами урока 5.

Для выполнения лабораторной работы потребуется использовать составное условие. Составное условие можно рассматривать, как два условия, объединённые в одно. Например, условие равенства трёх чисел можно записать в виде вложенного условия «если a = b, то если b = c, то числа равны» либо составного условия «если a = b и b = c, то числа равны». Фрагменты программ определения равенства трёх чисел с помощью вложенного условия и составного условия можно видеть на рисунке 68. Очевидно, что использовать составное условие удобнее.

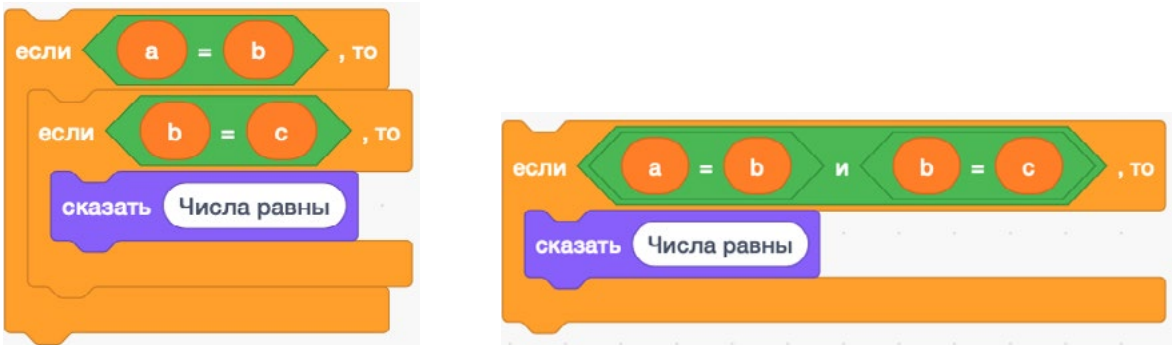

Рис. 68. Фрагменты программ определения равенства трёх чисел с помощью вложенного и составного условий

Для выведения последовательности трёх чисел удобно использовать конструкцию вида, как на рисунке 69.

Рис. 69. Фрагмент программы выведения последовательности трёх чисел

Практическая часть

Цель работы: знакомство с особенностями работы с условными алгоритмами в среде Scratch.

Ход лабораторной работы

1. Откройте среду Scratch.
2. С помощью инструмента «Выбрать фон» выберите фон «Театральная сцена» (рис. 70).

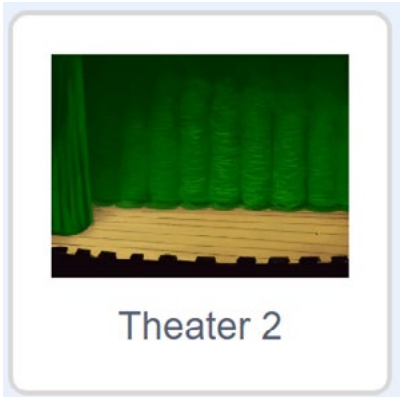

Рис. 70. Фон «Театральная сцена»

3. Удалите спрайт Кот.

4. С помощью инструмента «Выбрать спрайт» выберите новый спрайт — Робот (рис. 71).

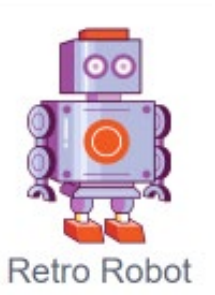

Рис. 71. Спрайт Робот

5. Создайте программу, которая выводит по убыванию три числа, введённые с клавиатуры. Пример вида сцены после решения задания показан на рисунке 72.

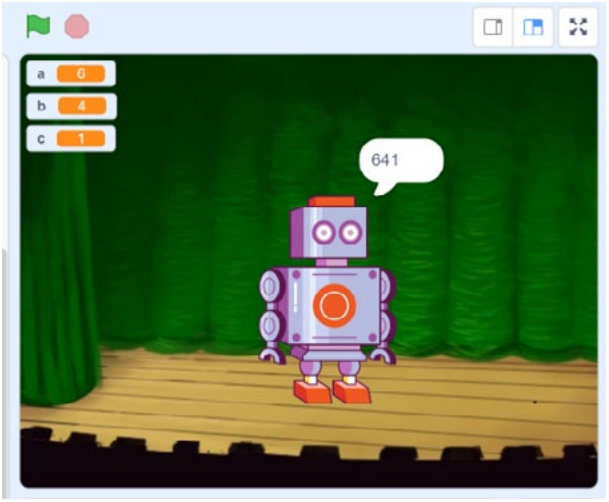

Рис. 72. Вид сцены после выполнения задания

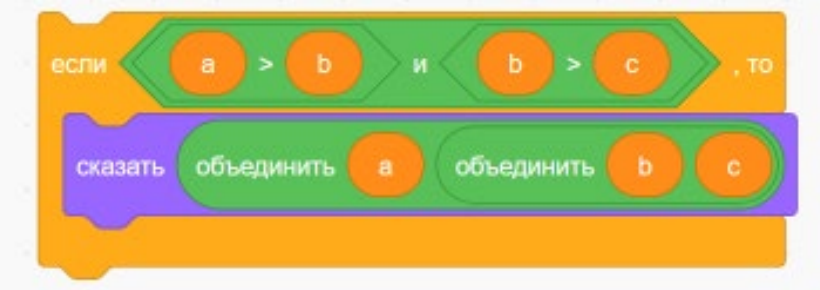

Рис. 73. Фрагмент сравнения и вывода результата программы

Вывод: в ходе выполнения лабораторной работы вы получили представление о составном условии.

Контрольные вопросы:

1. Что представляет собой составное условие?
2. Чем оно отличается от вложенного условия?
3. Сколько условных операторов будет использовано в программе?

Лабораторная работа 9 «Условные алгоритмы».

Теоретическая часть

Воспользоваться материалами уроков 4 и 5.

В ходе выполнения лабораторной работы потребуется вычислять корни квадратного уравнения. Для этого придётся проводить сложные вычисления. Для вычисления дискриминанта можно использовать конструкцию вида, как на рисунке 74, а для вычисления корней уравнения — конструкцией вида, как на рисунке 75.

Рис. 74. Фрагмент программы вычисления дискриминанта

Рис. 75. Фрагмент программы вычисления корня уравнения

Практическая часть

Цель работы: знакомство с особенностями работы с условными алгоритмами и сложными вычислениями в среде Scratch.

Ход лабораторной работы

1. Откройте среду Scratch.

2. С помощью инструмента «Выбрать фон» выберите фон «Театральная сцена» (рис. 76).

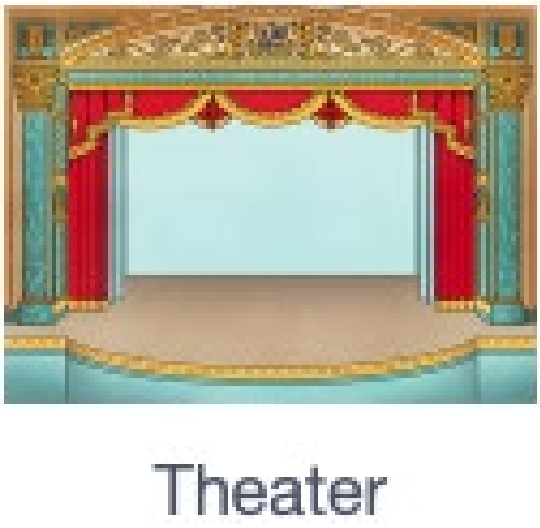

Рис. 76. Фон «Театральная сцена»

3. С помощью инструмента «Выбрать спрайт» выберите новый спрайт — Человечек (рис. 77).

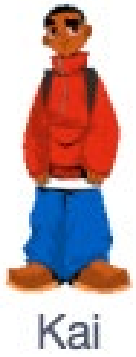

Рис. 77. Спрайт Человечек

4. Создайте программу вычисления корней квадратного уравнения. Все коэффициенты вводятся с клавиатуры. Если дискриминант больше нуля, то Человечек должен говорить: «Дискриминант больше нуля» — и выводить ответ в окнах «x1» и «x2». Если дискриминант равен нулю, то Человечек должен говорить: «Дискриминант равен нулю» — и выводить ответ в окне «x1». Если дискриминант меньше нуля, то он должен говорить: «Дискриминант меньше нуля. Корней нет».

5. Пример вида сцены после решения задания показан на рисунке 78.

Рис. 78. Вид сцены после выполнения задания

Вывод: в ходе выполнения лабораторной работы вы получили представление об условных операторах и сложных вычислениях.

Контрольные вопросы:

1. Какова последовательность условных операторов в разработанной программе?
2. Какова последовательность составления конструкции вычисления дискриминанта квадратного уравнения?
3. Какова последовательность составления конструкции вычисления корней квадратного уравнения?

Учебная программа курса «Алгоритмизация и программирование» (9 класс)

Целью программы «Алгоритмизация и программирование» является изучение основ программирования на языке Python, основных приёмов написания программ на современном языке программирования, развитие алгоритмического мышления учащихся, творческих способностей, аналитических и логических компетенций.

Планируемые результаты освоения программы

Для реализации поставленной цели планируется достижение личностных, метапредметных (познавательных, регулятивных, коммуникативных УУД) и предметных результатов.

Познавательные УУД:
- формирование представления об этапах решения задачи;
- умение делать выводы в процессе работы и по её окончании;
- формирование алгоритмического подхода к решению задач на языке программирования Python;
- формирование ключевых компетенций проектной и исследовательской деятельности.

Регулятивные УУД:
- формирование умений целеполагания;
- формирование умений прогнозировать свои действия и действия других участников группы;
- умение соотносить свои действия с планируемыми результатами;
- умение корректировать намеченный план действий и ставить новые цели;
- формирование умений самоконтроля и самокоррекции.

Коммуникативные УУД:
- формирование умений работать индивидуально и в группе для решения поставленной задачи;
- формирование трудолюбия, упорства, желания добиваться поставленной цели;
- формирование информационной культуры.

Личностные результаты:
- формирование профессионального самоопределения;
- формирование уважительного отношения к интеллектуальному труду;
- формирование личностного самоопределния.

Предметные результаты:
- формирование умений построения различных видов алгоритмов (линейных, разветвляющихся, циклических) для решения поставленных задач;
- изучение основных инструментов среды;
- получение навыков создания первых программ в среде программирования Python;
- формирование умений использовать основные конструкции языка программирования Python для решения поставленных задач;
- формирование умений работать в интегрированной среде программирования Python;
- формирование навыков работы с основными структурами данных в среде программирования Python;

- формирование навыков работы со структурой алгоритма;
- изучение понятия «переменная», задание значения переменной;
- изучение основных операторов Python, ввода/вывода данных, математических операторов;
- получение навыков использования условного оператора и разработки программ, реализующих разветвляющийся алгоритм;
- получение навыков использования операторов цикла и разработки программ, реализующих циклический алгоритм;
- получение навыков использования списков и разработки программ, реализующих работу со структурами данных;
- получение навыков использования строк и разработки программ, реализующих работу со строковыми данными;
- получение навыков использования функций и разработки программ, реализующих работу со вспомогательными алгоритмами;
- получение навыков использования кортежей и разработки программ, реализующих работу со структурами данных.

Тематическое планирование с определением основных видов деятельности

№ п/п	Тема	Содержание	Целевая установка урока	Кол-во часов			Основные виды деятельности обучающихся на уроке/ внеурочном занятии	Использование оборудования
				общее	теория	практика		
1	Первые программы на языке Python, основные операторы	Написание простых программ на языке программирования Python, знакомство с операторами присвоения, ввода/вывода данных, разработка программ, реализующих линейные алгоритмы, на языке программирования Python	Знакомство с основами написания программ на языке программирования Python, работа с операторами присвоения, ввода/вывода данных	2	1	1	Наблюдение за работой учителя, самостоятельная работа со средой программирования Python, ответы на контрольные вопросы	Компьютер, проектор, интерактивная доска
2	Условные операторы	Формат оператора ветвления if на языке программирования Python, разработка программ, реализующих условные алгоритмы	Знакомство с условным оператором if на языке программирования Python	3	1	2	Наблюдение за работой учителя, самостоятельная работа со средой программирования Python, ответы на контрольные вопросы	Компьютер, проектор, интерактивная доска
3	Циклические операторы	Формат оператора ветвления цикла с предусловием while, оператором цикла с параметром for на языке программирования Python, разработка программ, циклические алгоритмы	Знакомство с операторами цикла for, while в языке программирования Python	3	1	2	Наблюдение за работой учителя, самостоятельная работа со средой программирования Python, ответы на контрольные вопросы	Компьютер, проектор, интерактивная доска
	Итого:			8	3	5		

В содержание

Содержание и формы организации учебных занятий

Тема 1. Первые программы на языке Python, основные операторы

Рекомендованное количество часов: 1 час — комбинированный урок, 1 час — лабораторная работа.

Планируемые результаты: знакомство учащихся со средой программирования Python и создание первых программ.

Урок 1

Уровень образования: основное общее.

Предмет: информатика.

Тема урока: «Знакомство со средой программирования Python и создание первых программ».

Класс: 9.

Тип урока: комбинированный.

Цель урока: знакомство учащихся со средой программирования Python и создание первых программ.

Время реализации: 1 академический час.

Ход урока

I. Этап постановки цели и задач урока, мотивации к учебной деятельности — 5 мин.

Деятельность учителя: объясняет особенности среды программирования Python.

II. Этап изучения нового учебного материала — 25 мин.

Деятельность учителя: объясняет новый материал с использованием PyCharm.

Дидактические материалы

Хорошей традицией при изучении языка программирования является написание первой программы с простым приветствием на экране «Привет, я изучаю Python!»

На жёстком диске надо создать каталог с названием Python Project., запустить программу PyCharm, выбрать вкладку «Открыть» и указать вашу папку Python Project. На экране перед пользователем открывается рабочее окно проекта (рис. 79).

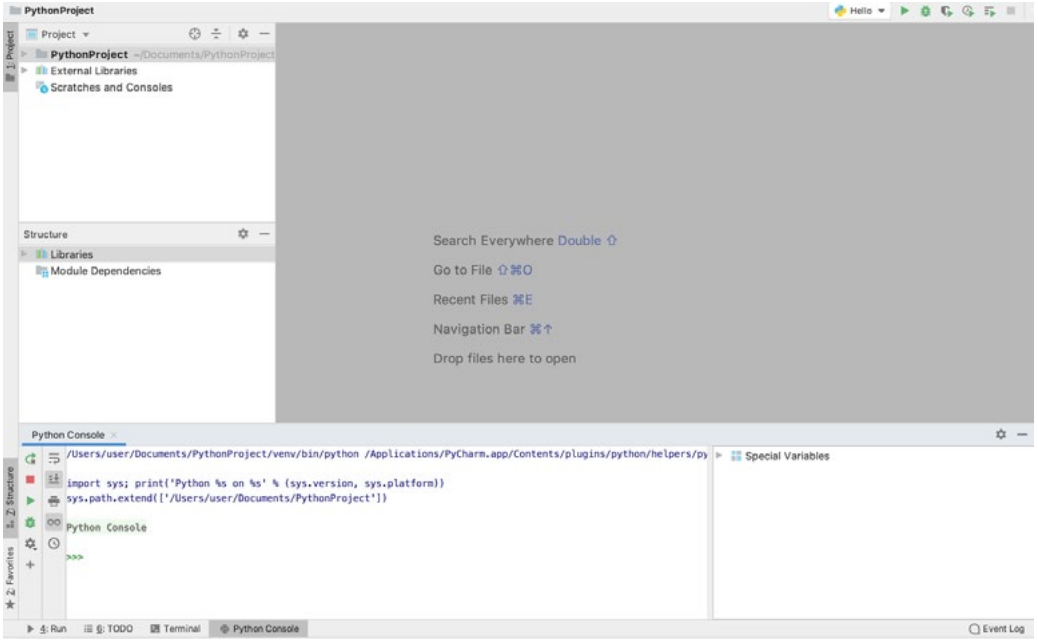

Рис. 79. Вид окна проекта

Далее в папке с проектом нужно нажать «Файл» –> «Новый» –> «Новый файл Python» (рис. 80).

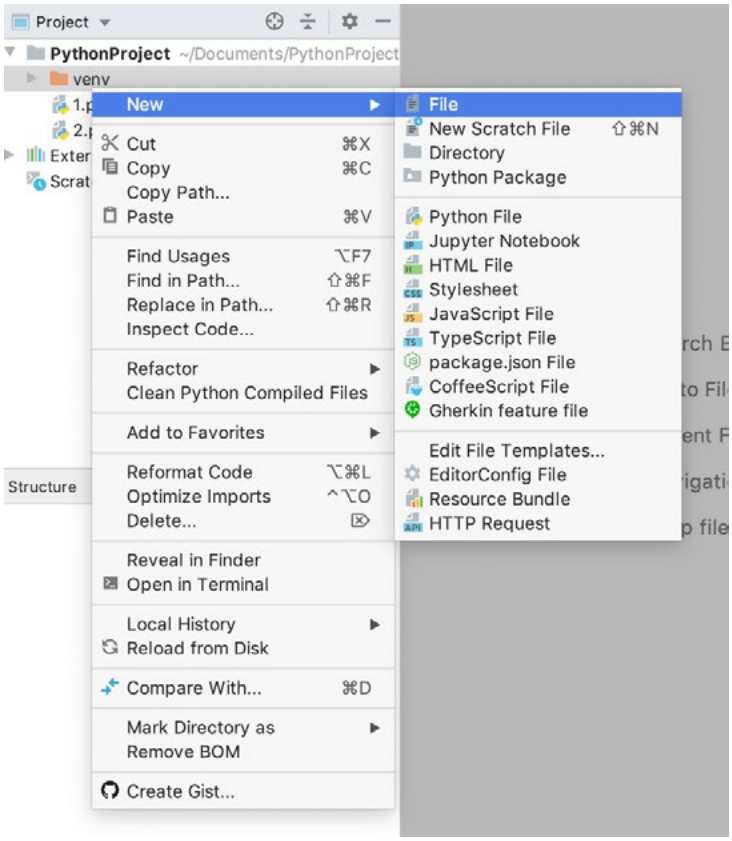

Рис. 80. Вид программы

Дать название созданному файлу: Hello (рис. 81, 82). Заметим, что расширение файла .py добавилось автоматически к созданному файлу. В правой области окна откроется область с файлом, в которой предполагается набирать код создаваемой программы.

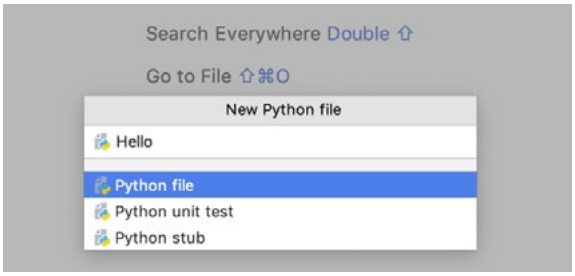

Рис. 81. Задание названия программы

Пусть первая команда будет print. Как только в редакторе PyCharm начинается набор, то PyCharm автоматически предлагает автозаполнение команды (рис. 82).

Рис. 82. Пример автозаполнения

В скобках в кавычках указывается текст, который необходимо вывести в результате работы программы (рис. 83).

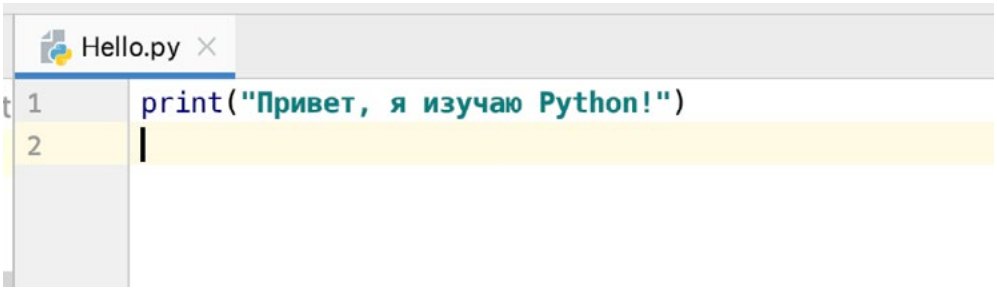

Рис. 83. Вид программы

После того как выполнен набор кода программы, необходимо её запустить и получить результат. Для запуска Python кода нужно в меню «Запуск» и «Отладки конфигураций» указать интерпретатор (Python нужной нам версии) и script path: это имя требуемого файла hello.py (рис. 84, 85). После этого надо применить настройки и закрыть окно конфигурации.

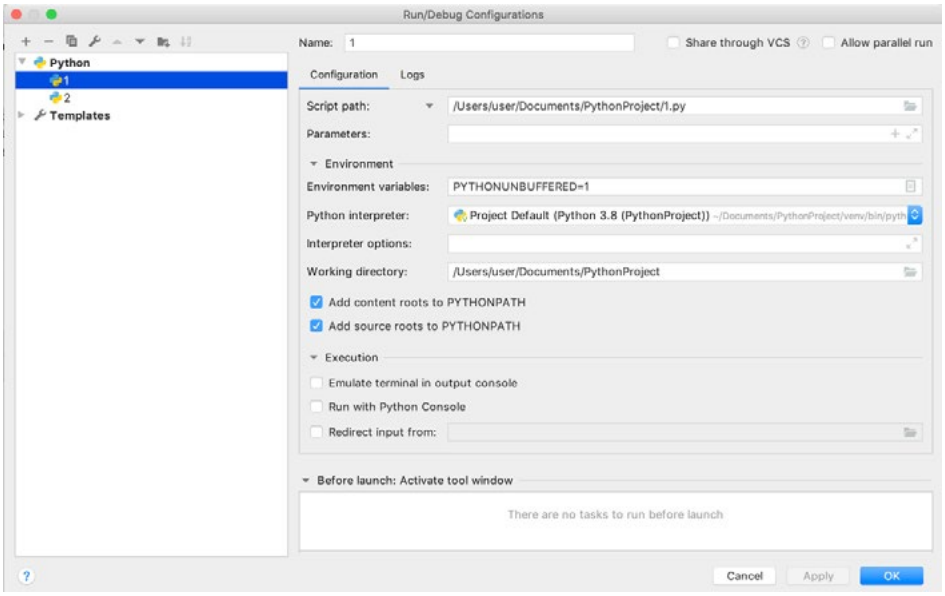

Рис. 84. Окно конфигурации

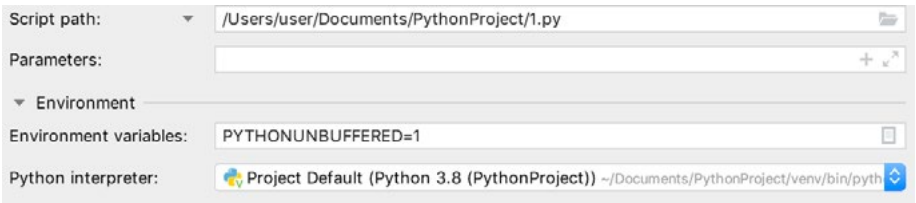

Рис. 85. Окно конфигурации

В правом верхнем углу программы PyCharm расположена панель запуска (рис. 86).

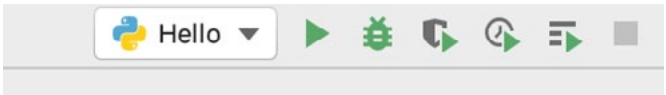

Рис. 86. Панель запуска

Для выполнения кода программы нужно нажать кнопку запуска Play. Результат работы программы появится мгновенно в окне вывода результатов (рис. 87).

Рис. 87. Результат работы программы

Далее предлагается рассмотреть следующий механизм, помогающий в написании хорошего кода. Здесь имеются в виду комментарии. Комментарии — это то, что пишется после символа # и является заметкой для читающего программу.

Пример 1.

print(Привет, я изучаю Python!) # print — это функция вывода

Рис. 88. Вид комментария в программе

Комментарии полезны для тех, кто будет читать разработанную программу, так им легче будет понять, что программа делает.

В следующем примере предлагается рассмотреть случай, когда в

строку вывода требуется добавить какие-либо данные. Для этого в Python применяют метод format().

Пример 2.

age=5

print(“Привет, я изучаю Python {0} лет!”. format(age))

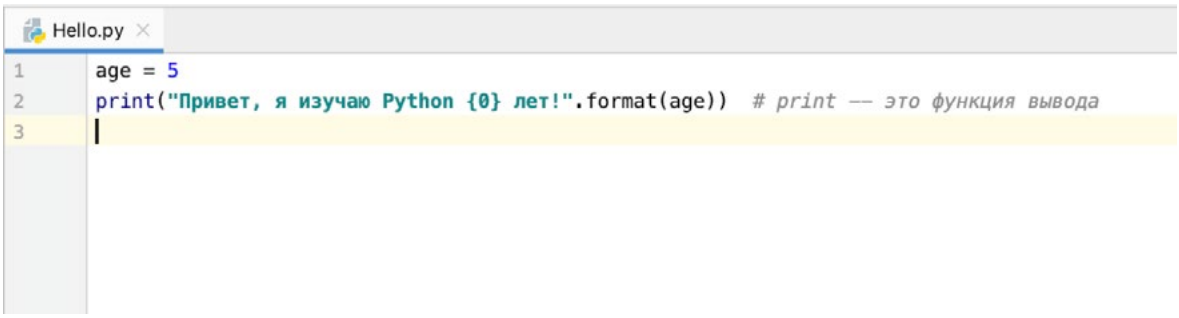

Рис. 89. Вид программы

Рис. 90. Результат работы программы

Заметим, что в строку могут быть включены определённые значения, а метод format замещает эти обозначения соответствующими аргументами. Обозначение {0} соответствует переменной age, которая является первым аргументом метода format. Python начинает отсчёт с нуля, поэтому первая позиция именно 0.

Учитель предлагает учащимся под его контролем создать первую программу на Python.

III. Этап проверки понимания и первичного закрепления — 5 мин.

Учитель предлагает учащимся вспомнить кратко новый материал: запуск программы Python, оператор print, оператор format.

IV. Этап контроля усвоения и коррекции ошибок — 5 мин.

Учитель задаёт вопросы ученикам:

1) С какой средой программирования вы познакомились в ходе урока?

2) Какие операторы выполняла первая программа на языке программирования Python?

V. Информация о домашнем задании, инструктаж по его выполнению — 3 мин.

VI. Этап рефлексии деятельности на уроке — 2 мин.

Учитель спрашивает учащихся об их впечатлениях от урока, что понравилось, а что было непонятно.

Лабораторная работа 1
«Первые программы на языке Python, основные операторы».

Теоретическая часть

Любую программу на языке Python можно представить, как набор лексем (допустимых символов), записанных в определённом порядке и по определённым правилам. Лексема может представлять собой: комментарии, литералы, знаки пунктуации, переменные, специальные ключевые слова.

Программа на языке Python может содержать достаточное количество комментариев, каждый комментарий начинается с символа # (решетка).

Литералы представляют собой значения, заданные в коде программы, например числа (25) или строки (“привет”). В языке Python представлены некоторые встроенные типы объектов. В нём используется динамическая типизация (типы данных определяются автоматически, и их не требуется объявлять в программном коде), но при этом он является языком со строгой типизацией (вы сможете выполнять над объектом только те операции, которые применимы к его типу).

Если говорить об использовании знаков пунктуации, то стоит отметить, что каждая строка в программе на языке Python не должна заканчиваться точкой с запятой, как, например, в C++, но если есть необходимость записать несколько инструкций в одну строчку, то их можно разделять точкой с запятой.

Переменная или идентификатор используются для хранения данных различного типа. В языке Python нет специального раздела описания переменных, в котором указывается тип переменной перед её первым использованием. Есть определённые правила для задания имён переменных в языке Python: это последовательность букв, которая не может начинаться с цифры, но может содержать символ подчёркивания (_). Имена переменных чувствительны к регистру и не могут совпадать со специальными ключевыми словами.

Ключевые слова в языке Python имеют специальное назначение и представляют собой управляющие конструкции языка, например: and, break, for и т. д.

При составлении программ лексемы объединяются в синтаксические конструкции, которые могут вкладываться друг в друга. В результате могут образовываться блочные конструкции, каждый блок кода начинается двоеточием (:), а тело блока выделяется обязательным отступом в виде четырёх пробелов. Обычно среда программирования сразу делает отступ для блока после двоеточия. В зависимости от используемой среды программирования блоки могут иметь визуальное выделение.

Перейдём к рассмотрению основных операторов языка.

Операторы языка Python можно разделить на 7 типов:

— арифметические операторы;
— операторы сравнения;
— операторы присваивания;
— логические операторы;
— операторы принадлежности;
— операторы тождественности;
— битовые операторы.

Основные операторы языка Python

Оператор	Описание
+	Сложение
–	Вычитание
*	Умножение
/	Деление
//	Целочисленное деление
%	Остаток от деления
**	Возведение в степень
<	Меньше
>	Больше
>=	Больше или равно
<=	Меньше или равно
==	Равно
!=	Не равно

Рассмотрим несколько примеров использования данных операторов.

Пример 1.

```
>>> 2+4
>>>
```

Пример 2.
>>> 3**6
>>>

Пример 3.
>>> 74%7
>>>

Результат работы программы представлен на рисунке 91.

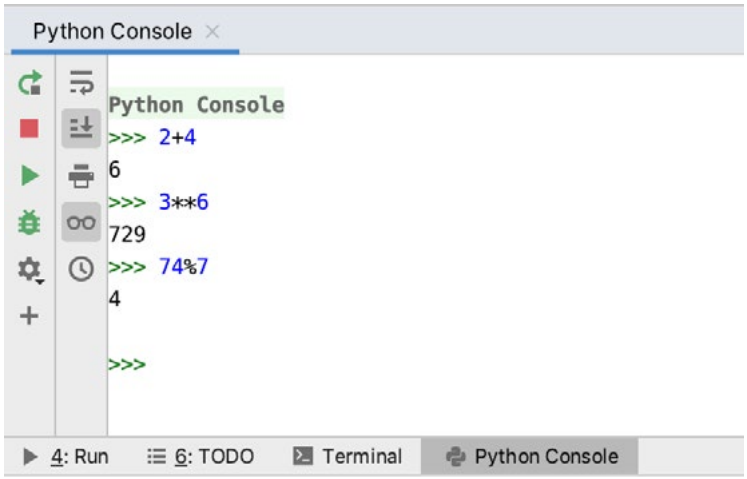

Рис. 91. Вид программы в среде разработки

Один из самых часто используемых операторов — оператор присваивания. Он в языке Python имеет вид:
<идентификатор>=<выражение>
Пример 4.
>>> x=6
>>> x
>>> x+=8
>>> x

Результат работы программы представлен на рисунке 92.

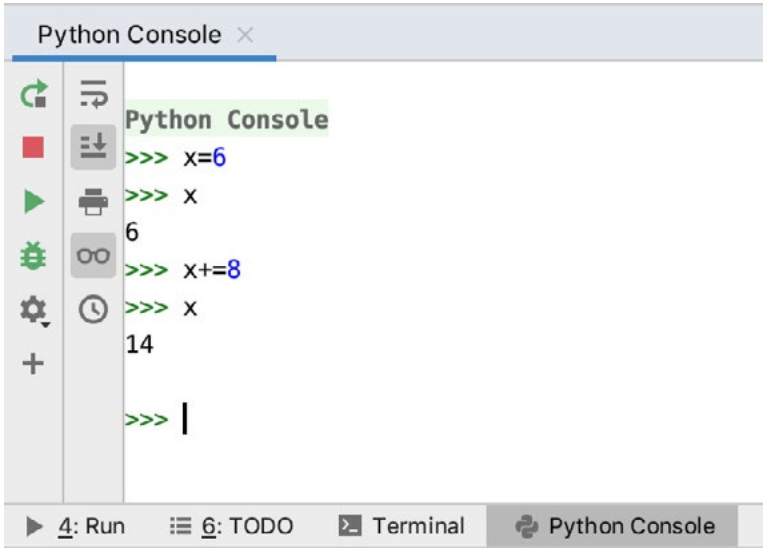

Рис. 92. Результат работы программы

Часто при решении задач необходимо выполнять ввод данных с клавиатуры и выводить данные на экран. Рассмотрим работу соответствующих операторов в языке Python.

Ввод данных с клавиатуры осуществляется с помощью оператора print. Формат оператора:

print()

После выполнения оператора происходит переход курсора на новую строку.

Пример 5.

```
x=3
y=6
z=-10
print(x,y,z)
```

Результат работы программы представлен на рисунке 93.

Рис. 93. Результат работы программы

Пример 6.

```
x=3
y=6
z=-10
print(x)
print(y,z)
```

Результат работы программы представлен на рисунке 94.

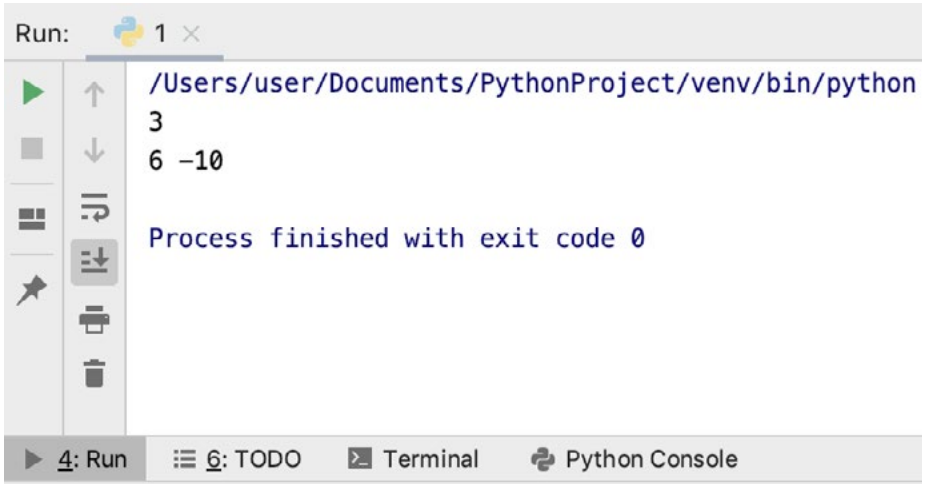

Рис. 94. Результат работы программы

Сложные операторы присваивания

Обозначение	Описание
+=	Левая и правая части суммируются, результат присваивается переменной в левой части
−=	Вычитается значение правой части из значения переменной в левой части, результат присваивается переменной в левой части
*=	Левая и правая части умножаются, результат присваивается переменной в левой части
/=	Делится значение переменной из левой части на значение выражения в правой части, результат присваивается переменной в левой части
%=	Находится остаток от деления значения переменной из левой части на значение выражения в правой части, результат присваивается переменной в левой части
**=	Выполняется возведение левой части в степень значения выражения из правой части, результат присваивается переменной в левой части
//=	Делится нацело значение переменной из левой части на значение выражения в правой части, результат присваивается переменной в левой части

Пример 7.

```
x=3
y=6
z=−10
print(x)
x+=6
print("x=",x,end='')
print(y,z)
```

Результат работы программы представлен на рисунке 95.

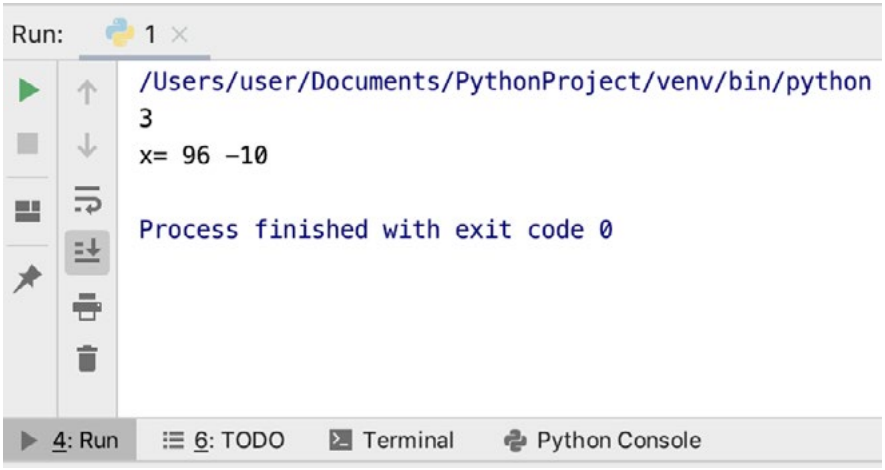

Рис. 95. Результат работы программы

Выводить на экран можно не только значения переменных, но и значения выражений.
Пример 8.

```
x=3
y=6
z=−10
print(x)
x+=6
print("x=",x,end='')
print(y+z)
```

Результат работы программы представлен на рисунке 96.

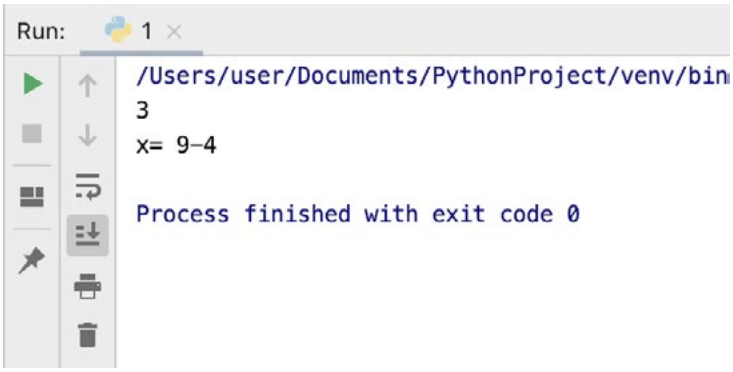

Рис. 96. Результат работы программы

Также в языке Python можно использовать форматированный вывод. Для этого необходимо использовать встроенную функцию format(). Синтаксис функции выглядит следующим образом:

<строка>.format(<формат>)

«Строка» представляет собой значение для форматированного вывода, «формат» — спецификация формата «Mini-Language».

Пример 8.

```
a = 12
b = 13
c=a+b
print( «{}+{}={}».format(a, b, c) )
```

Результат работы программы представлен на рисунке 97:

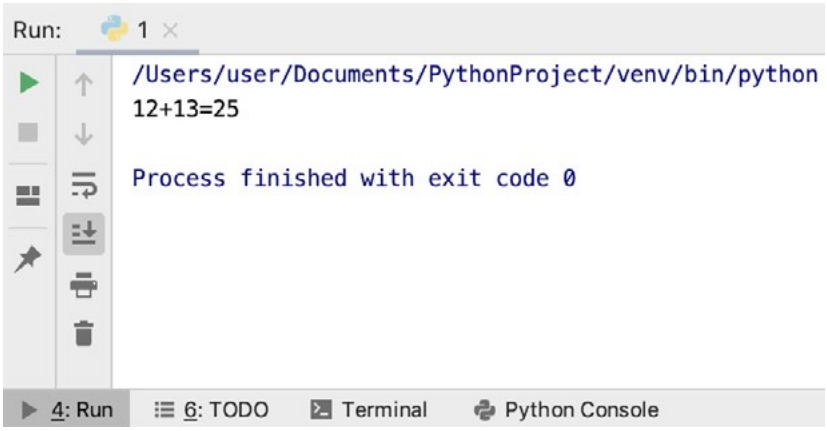

Рис. 97. Результат работы программы

Для ввода данных с клавиатуры используется встроенная функция input(). Данная функция возвращает в качестве результата строку.

В результате выполнения программы

```
a=input()
c=input()
z=a+c
print(z)
```

На экран выводится не сумма чисел.

Результат работы программы представлен на рисунке 98.

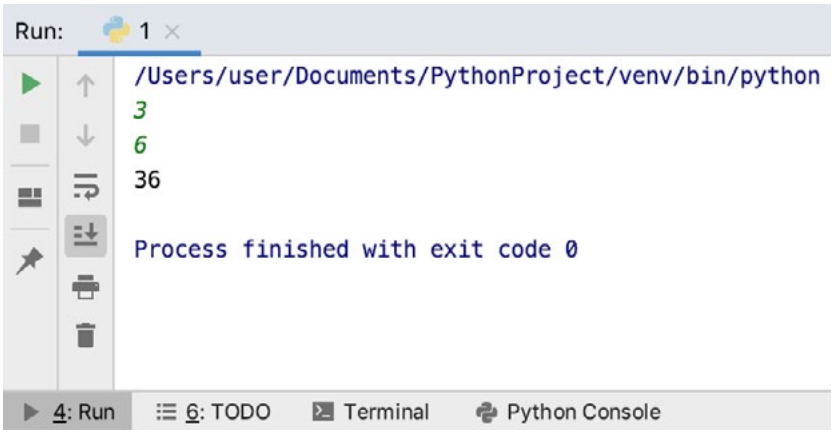

Рис. 98. Результат работы программы

Если необходимо ввести числовые значения, то следует использовать встроенные функции int и float.

Функция int используется для преобразования строки в целое число. Изменим предыдущий пример:

```
a=int(input())
c=int(input())
z=a+c
print(z)
```

Результат работы — сумма чисел.

Результат работы программы представлен на рисунке 99.

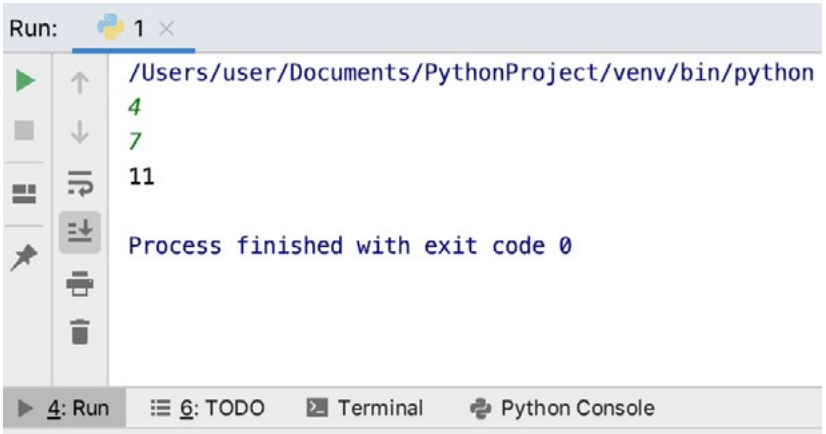

Рис. 99. Результат работы программы

Также можно указывать аргумент функции input, в этом случае на экран будет выводиться «пригласительное сообщение».

```
a=int(input(«a= «))
c=int(input(«c= «))
z=a+c
print(z)
```

Результат работы программы представлен на рисунке 100.

Рис. 100. Результат работы программы

Аналогично работает и функция float, но в результате её работы введённое значение преобразовывается в вещественное число.

```
a=float(input(«a= «))
x=float(input(«x= «))
z=a+x
print(z)
```

Результат работы программы представлен на рисунке 101.

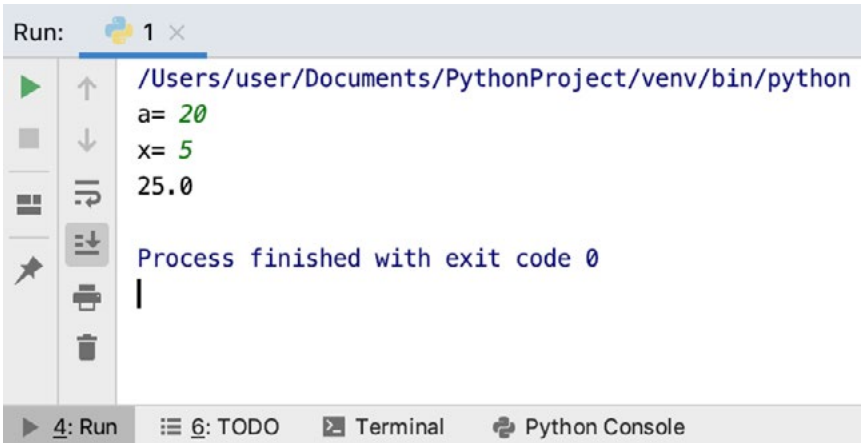

Рис. 101. Результат работы программы

В этом же разделе упомянем работу со встроенными математическими функциями. Все математические функции в языке Python объединены в модуль math. Данный модуль при необходимости нужно импортировать с помощью команды import: import math.

После подключения можно использовать всё множество математических функций языка Python.

Пример 9. Вычислить выражение , где *a*, *b* — целые числа, введённые с клавиатуры.

```
a=int(input(«a= «))
b=int(input(«b= «))
c=a**2+b**2
print(“c=”, c)
```

Результат работы программы представлен на рисунке 102.

Рис. 102. Результат работы программы

Практическая часть

Цель работы: знакомство с основами написания программ на языке программирования Python, работа с операторами присвоения, ввода/вывода данных.

Ход лабораторной работы

1. Откройте среду разработки Python.
2. Составьте программу нахождения периметра и площади прямоугольника (значения длин сторон вводятся с клавиатуры).

Указание. Примерный листинг программы может иметь вид:

```
a=float(input(«a= «))
b=float(input(«b= «))
print(«периметр= «, 2*(a+b))
print(«площадь= «,a*b)
```

Вывод: в ходе выполнения лабораторной работы вы получили представление о написании простых программ на языке программирования Python.

Контрольные вопросы:

1. Какие операторы языка Python использовались при выполнении лабораторной работы?
2. Какой оператор используется для ввода целых чисел с клавиатуры, вещественных чисел с клавиатуры?
3. Как вывести данные на экран в языке Python?

Тема 2. Условные операторы

Рекомендованное количество часов: 1 час — комбинированный урок, 2 часа — лабораторные работы.

Планируемые результаты: знакомство учащихся с условными операторами.

Урок 2

Уровень образования: основное общее.

Предмет: информатика.

Тема урока: «Условные операторы».

Класс: 9.

Тип урока: комбинированный.

Цель урока: знакомство учащихся с условными операторами среды программирования Python.

Время реализации: 1 академический час.

Ход урока

I. Этап постановки цели и задач урока, мотивации к учебной деятельности — 5 мин.

Деятельность учителя: обосновывает необходимость использования условных операторов в программировании.

II. Этап изучения нового учебного материала — 25 мин.

Деятельность учителя: объясняет новый материал с использованием PyCharm.

Дидактические материалы

Часто при решении задач требуется выполнить алгоритм ветвления — выполнение каких-то действий при выполнении или невыполнении какого-то условия. Общая форма условного оператора выглядит следующим образом:

```
if <условие1>:
   оператор1
   elif <условие2>:
      оператор2
   else:
   оператор3
```

Части else и elif являются необязательными. Elif представляет собой вложенное условие.

В условиях могут использоваться логические операции and (конъюнкция), or (дизъюнкция), not — отрицание. Равенство обозначается == (чтобы отличить знак равенства от оператора присвоения), а неравенство !=. Также в Python можно записывать двойное условие, например 2<=a<=5, −10<v<=9.

После логического условия обязательно стоит двоеточие, для того чтобы показать, что далее идёт блок выражений. Блок выражений записывается после отступа.

Рассмотрим работу оператора более подробно.

В самом простом случае оператор ветвления имеет вид:

```
if <условие1>:
   оператор1
```

Блок-схема работы данного оператора представлена на рисунке 103.

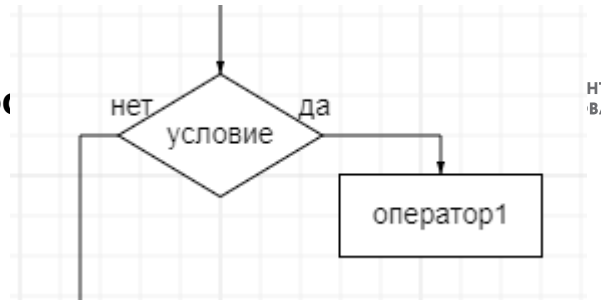

Рис. 103. Блок-схема условного оператора

В случае выполнения условия выполняется оператор1, затем управление передаётся оператору, следующему за оператором if.

Приведём пример:

Пример 1.

```
a = int(input())
if a < 0:
    print("Ниже")
```

Результат работы программы представлен на рисунке 104.

Рис. 104. Результат работы программы

В данном примере в качестве условия используется сравнение a<0. Если оно выполняется, то на экран выводится «Ниже». Если же условие ложно, то программа ничего не выполняет.

Пример 2.

```
a = int(input())
if (a < 0) and(a>=-3):
    print('Ниже')
```

Результат работы программы представлен на рисунке 105.

Рис. 105. Результат работы программы

В данном примере условие составное: состоит из двух условий, объединённых операцией and (логическое «и»).

Рассмотрим далее более сложный вид оператора ветвления:

```
if <условие1>:
   опертор1
else:
   оператор2
```

Обратите внимание на порядок отступов в формате оператора!

Блок-схема работы данного оператора представлена на рисунке 106.

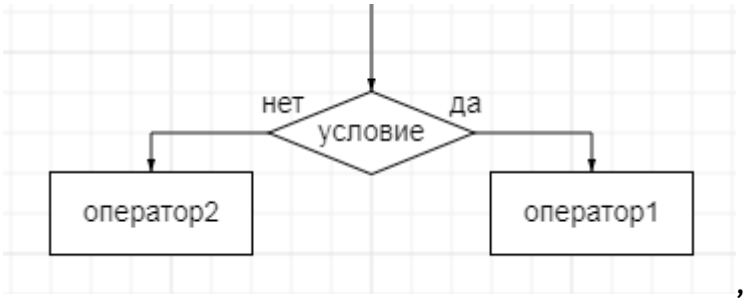

Рис. 106. Блок-схема условного оператора

Если условие истинно, то далее выполняется оператор1, а в противном случае (если условие ложно), то выполняется оператор2. А далее управление переходит к оператору, который следует за оператором ветвления.

Приведём примеры работы более сложной формы оператора ветвления.

Пример 3.

```
a = int(input())
b = int(input())
if a+b>10:
   print(‹Yes›)
else:
   print('No')
```

Результат работы программы представлен на рисунке 107.

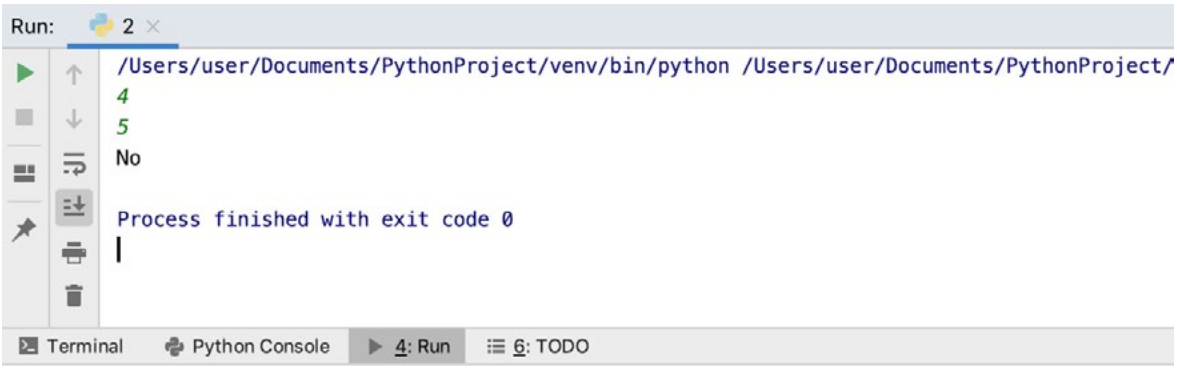

Рис. 107. Результат работы программы

В данном примере в случае истинности условия a+b>10 выполняется оператор print("Yes"), а в противном случае — print("No").

Обратите внимание, что после условия и после части else можно указывать несколько операторов, но тогда все они идут после отступа! Для сравнения рассмотрим два примера.

Пример 4.1.

```
a=10
b=15
c=3
if a+b>10:
   print('Yes')
else:
   print('No')
   c=a+b
print(c)
```

Результат работы программы представлен на рисунке 108.

Рис. 108. Результат работы программы

Пример 4.2.

```
a=10
b=15
c=3
if a+b>10:
   print('Yes')
else:
   print('No')
c=a+b
print(c)
```

Результат работы программы представлен на рисунке 109.

Рис. 109. Результат работы программы

Как видим, результаты работы программ отличаются, потому что в примере 4.2. оператор c=a+b выполняется в любом случае, так как стоит вне блока оператора ветвления.

Третья форма оператора ветвления выглядит следующим образом:

```
if <условие1>:
   опертор1
elif <условие2>:
   опертор2
else:
   опертор3
```

Блок-схема работы данного оператора представлена на рисунке 110.

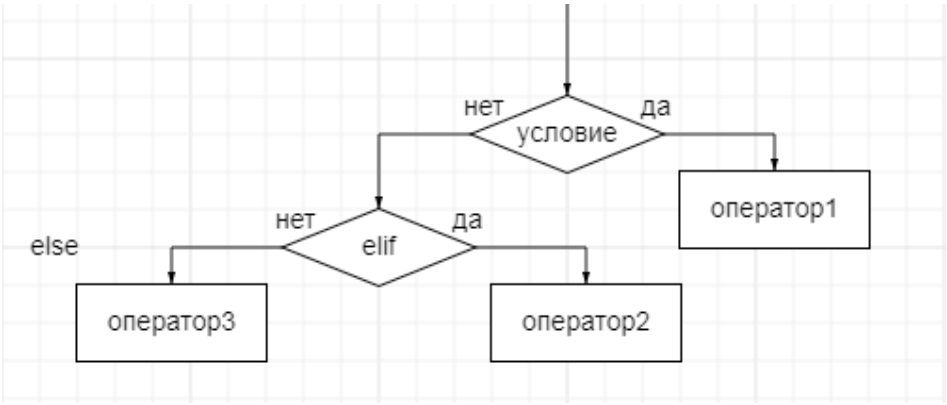

Рис. 110. Блок-схема условного оператора

В данном случае осуществляется проверка нескольких условий: после if (условие) и после elif. Оператор после else выполняется в том случае, если не выполнилось условие 2 после части elif.

Рассмотрим пример использования данной формы оператора if.

Пример 5.

```
a=10
if a < -5:
   print('No')
elif -5 <= a <= 5:
   print('Maybe')
else:
   print('No')
```

Результат работы программы представлен на рисунке 111.

Рис. 111. Результат работы программы

Учитель предлагает обучающимся под его контролем выполнить задания 3, 4 и 5.

III. Этап проверки понимания и первичного закрепления — 5 мин.

Учитель предлагает учащимся вспомнить кратко новый материал: что представляет собой условный оператор, какой оператор будет выполняться в случае выполнения условия, а какой в случае невыполнения условия, что такое вложенные условия.

IV. Этап контроля усвоения и коррекции ошибок — 5 мин.

Учитель задаёт вопросы ученикам:

1) Для чего нужны условные операторы?

2) Какой можно привести пример, когда удобно использовать условные операторы?

V. Информация о домашнем задании, инструктаж по его выполнению — 3 мин.

VI. Этап рефлексии деятельности на уроке — 2 мин.

Учитель спрашивает учащихся об их впечатлениях от урока, что понравилось, а что было непонятно.

Лабораторная работа 2 «Условные операторы»

Теоретическая часть

Воспользоваться материалами урока 2.

Пример 1. Написать программу вычисления стоимости покупки с учётом скидки. Скидка 5% предоставляется в том случае, если стоимость покупки превысила 1000 р., а 10% — если стоимость выше 3000 р.

Листинг программы представлен ниже:

```
s=int(input())
if s<1000:
   print(s)
elif 1000<=s<3000:
   print(s*0.97)
else:
   print(s*0.95)
```

Результат работы программы представлен на рисунке 112.

Рис. 112. Результат работы программы

Ещё раз обратимся к синтаксическим правилам языка Python.

В языке Python отсутствуют фигурные скобки, как в языке C/C++, или разделители «begin/end», как в языке Pascal, окружающие блоки программного кода. Вместо этого принадлежность операторов к вложенному блоку определяется по величине отступов. Также операторы в языке Python обычно не завершаются точкой с запятой; обычно признаком конца инструкции служит конец строки с этой инструкцией.

Все составные операторы в языке Python оформляются одинаково: строка с заголовком завершается двоеточием, далее следуют одна или более вложенных инструкций, обычно с отступом относительно заголовка. Эти инструкции с отступами называются блоком (или иногда набором). Интерпретатор автоматически определяет границы блоков по величине отступов, т. е. по ширине пустого пространства слева от программного кода. Все инструкции, смещённые вправо на одинаковое расстояние, принадлежат к одному и тому же блоку кода.

В инструкции if предложения elif и else являются не только частями инструкции if, но и заголовками с собственными вложенными блоками. На рисунке 113 представлена схема с отступами для оператора if.

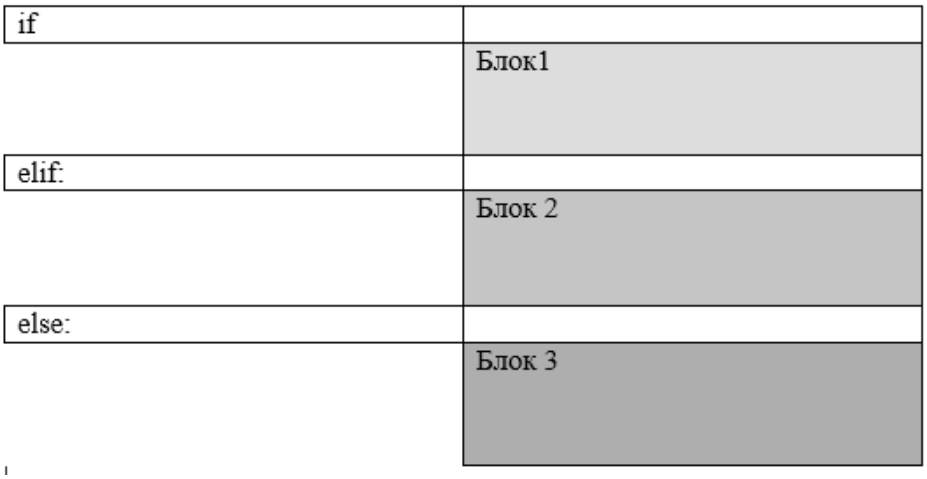

Рис. 113. Схема вложенных блоков

Практическая часть

Цель работы: знакомство с условным оператором if на языке программирования Python.

Ход лабораторной работы

Откройте среду разработки Python PyCharm.

Напишите программу вычисления оптимального веса: оптимальный вес вычисляется по формуле: рост (см) — 100. Пользователю выдаётся рекомендация о снижении или набора веса.

Указание. Примерный листинг программы может иметь вид:

```
v=int(input(«Ваш вес — «))
h=int(input(«Ваш рост — «))
ideal_v=h-100
if v==ideal_v:
   print(“Ваш вес в норме»)
elif v>ideal_v:
   print(“Ваш вес выше нормы! Необходимо сбросить вес”)
else:
   print(“Ваш вес ниже нормы! Необходимо набрать вес»)
```

Результат работы программы представлен на рисунке 114.

```
Ваш вес — 68
Ваш рост — 150
Ваш вес выше нормы! Необходимо сбросить вес
```

Рис. 114. Результат работы программы

Составьте программу, определяющую вид треугольника (равносторонний, равнобедренный, разносторонний) по заданным длинам сторон.

Напишите программу, которая считывает три вещественных числа и заменяет каждое чётное значение его частным при делении на 2, а единицу — числом 2.

Определите, является ли введённый пользователем год високосным.

Указание. Примерный листинг программы может иметь вид:

```
y = int(input(«Введите год «))
if (y % 4 == 0 and y%100 != 0) or (y%400 == 0):
   print(«Високосный»)
else:
   print(«Обычный»)
```

Составьте программу, проверяющую, чётным или нечётным является введённое число.

Вывод: в ходе выполнения лабораторной работы вы получили представление о составлении условных алгоритмов с использованием оператора if — elif — else в языке программирования Python.

Контрольные вопросы:

1. Для чего используются операторы ветвления в программировании?
2. Как выглядит синтаксис оператора ветвления в языке Python?
3. Какие формы оператора ветвления можно использовать в языке Python?

Лабораторная работа 3 «Условные операторы»

Теоретическая часть

Воспользоваться материалами лабораторной работы 2.

Практическая часть

Цель работы: знакомство с условным оператором if в языке программирования Python.

Ход лабораторной работы

1. Откройте среду разработки Python PyCharm.
2. Составьте программу, в которой требуется ввести рост человека. Затем появляется надпись “ВЫСОКИЙ”, если его рост превышает 180 см, и “НЕ ОЧЕНЬ ВЫСОКИЙ” в противном случае.
3. Составьте программу, в которой требуется ввести три вещественных числа. Затем надо вывести их на экран в порядке возрастания.
4. Составьте программу, в которой требуется ввести три вещественных числа. Затем надо вывести их на экран в порядке убывания.
5. Составьте программу, в которой требуется ввести целые числа a, b, c. Программа должна проверить, имеет ли уравнение $ax^2 + bx + c = 0$ действительные корни. Если имеет, то вывести их на экран. В противном случае должно появиться сообщение, что корней нет.

Вывод: в ходе выполнения лабораторной работы вы получили представление о составлении условных алгоритмов с использованием оператора if — elif — else в языке программирования Python.

Контрольные вопросы:

1. Как выглядит полная форма оператора ветвления в языке Python?
2. Для чего используется часть else в операторе ветвления?
3. Какие основные операторы вы использовали при решении задач из лабораторной работы?

Тема 3. Циклические операторы

Рекомендованное количество часов: 1 час — комбинированный урок, 2 часа — лабораторные работы.

Планируемые результаты: знакомство учащихся с циклическими операторами среды программирования Python.

Урок 3

Уровень образования: основное общее.

Предмет: информатика.

Тема урока: «Циклические операторы».

Класс: 9.

Тип урока: комбинированный.

Цель урока: знакомство учащихся с циклическими операторами среды программирования Python.

Время реализации: 1 академический час.

Ход урока

I. Этап постановки цели и задач урока, мотивации к учебной деятельности — 5 мин.

Деятельность учителя: обосновывает необходимость использования циклических операторов в программировании.

II. Этап изучения нового учебного материала — 25 мин.

Деятельность учителя: объясняет новый материал с использованием PyCharm.

Дидактические материалы

Цикл в языке программирования представляет собой конструкцию, многократно выполняющую одну и ту же группу операторов. Число повторений циклов, или итераций, может быть задано заранее или зависеть от истинности того или иного условия. В реальной жизни постоянно применяются циклы, поэтому циклический алгоритм часто используются при решении задач по программированию.

В языке программирования Python может быть реализовано два вида циклов:

с предусловием — цикл while;

с параметром — цикл for.

Цикл while является одним из самых часто используемых и самых универсальных циклов в Python. Полный формат данного цикла представлен ниже:

```
while <условие>:
<оператор1>
else:
<оператор2>
```

Часть else является необязательной, она выполняется, когда управление передаётся за пределы цикла без использования инструкции break. Блок-схема работы цикла while представлена на рис. 115.

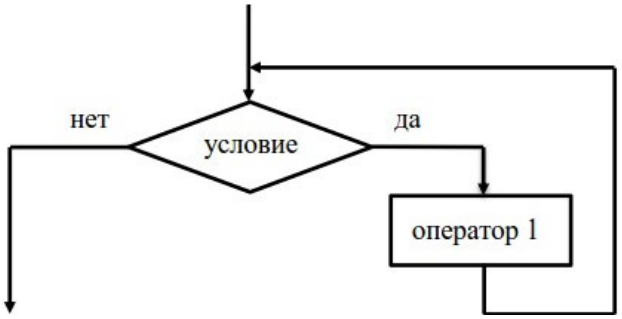

Рис. 115. Блок-схема работы цикла с предусловием while

Выполнение цикла while начинается с вычисления выражения. Если оно истинно (не равно false), выполняется оператор цикла. Если при первой проверке выражение равно false, цикл не выполнится ни разу. Тип выражения должен быть арифметическим или приводимым к нему. Если условие в цикле while никогда не будет ложным, то не будет причин остановки цикла и программа «зациклится». Чтобы этого не произошло, необходимо организовать момент выхода из цикла, т. е. ложность выражения в заголовке. Так, например, изменяя значение какой-нибудь переменной в теле цикла, можно довести логическое выражение до ложности. Обратите внимание, что операторы тела цикла должны быть отделены отступом.

Пример 1.

```
i = 5
while i < 15:
    print(i)
    i+= 2
```

Результат работы программы представлен на рисунке 116:

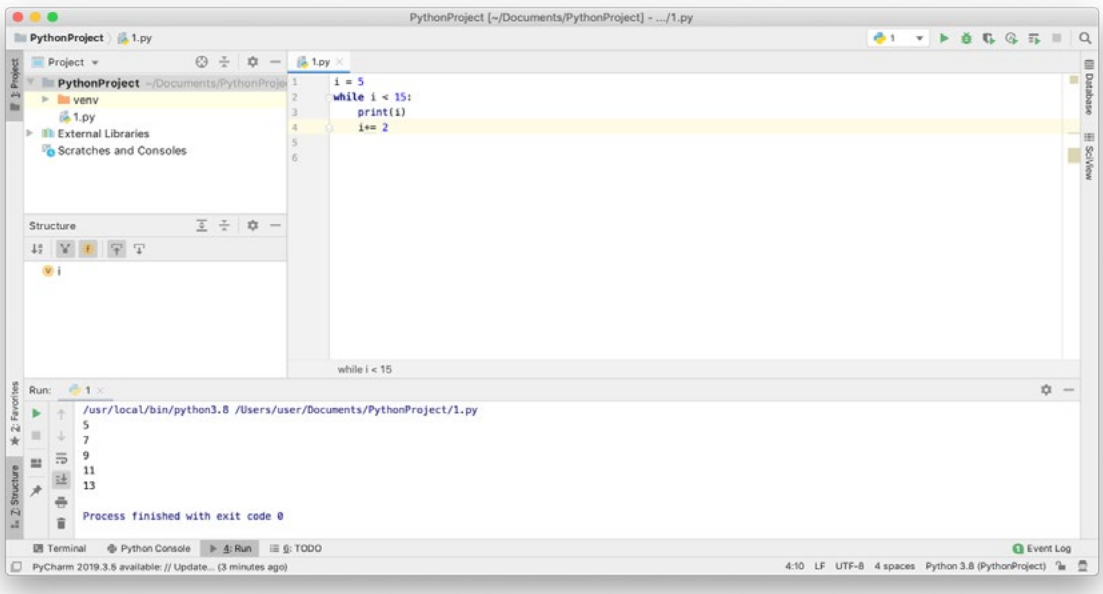

Рис. 116. Результат работы программы

В данном примере организован перебор значений переменной i с шагом 2.

Результат работы программы представлен на рисунке 117:

Рис. 117. Результат работы программы

Условие работы цикла i < 15. В теле цикла происходит изменение переменной i, поэтому цикл не будет бесконечным.

Пример 2.

```
a = 0
while a < 7:
    print(«Python»)
    a += 1
```

В данном примере цикл выполняется, пока истинно условие a < 7, значение переменной a меняется в теле цикла.

Результат работы программы представлен на рисунке 118.

Рис. 118. Результат работы программы

Второй цикл, используемый в языке Python, это цикл с параметром. Синтаксис данного цикла представлен ниже.

```
for <переменная> in <объект>:
    <оператор1>
else:
    <оператор2>
```

Блок-схема работы цикла выглядит следующим образом:

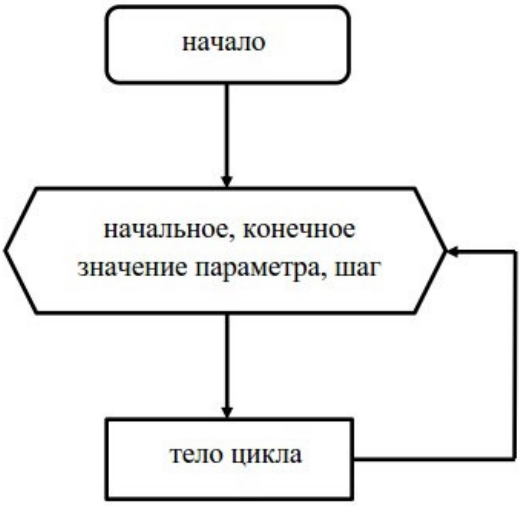

Рис. 119. Блок-схема работы цикла с параметром

В цикле for имеется необязательная часть else, которая работает точно так же, как и в цикле while. Она выполняется, если выход из цикла производится не по инструкции break.

Часто с параметром for используется функция range.

Функция range() возвращает последовательность чисел, регулируемую количеством переданных в неё аргументов. Возможны следующие варианты обращения к данной функции:

- range(finish);
- range(start, finish);
- range(start, finish, step).

Здесь start — это первый элемент последовательности (включительно), finish — последний (не включительно), а step — разность между следующим и предыдущим членами последовательности.

Например, range(5) возвращает последовательность 0, 1, 2, 3, 4. Вызов range(2, 8) возвращает последовательность 2, 3, 4, 5, 6, 7.

Рассмотрим примеры организации работы цикла с параметром.

Пример 3.

```
for a in range(10):
    print(a)
```

Результат работы программы представлен на рисунке 120:

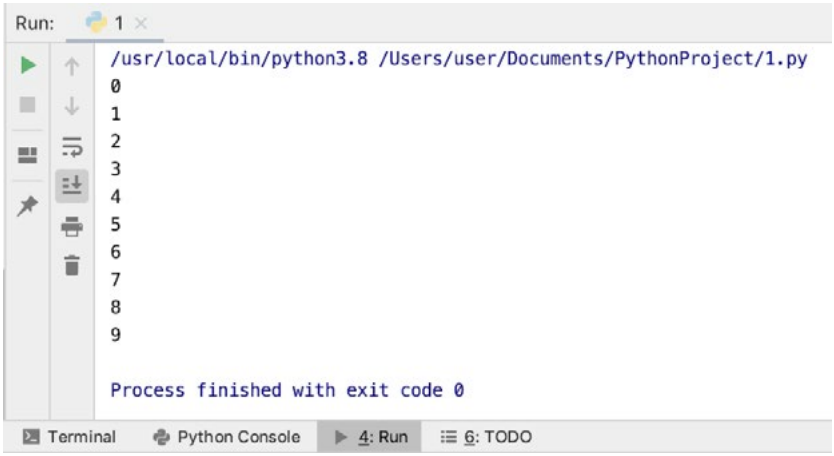

Рис. 120. Результат работы программы

В данном примере цикл выводит на экран последовательность чисел от 0 до 9 включительно.

Пример 4.

```
for c in range(0, 22, 3):
    print(c, end=» «)
```

В данном примере выводится на экран последовательность чисел от 0 до 21 с шагом 3.

Результат работы программы представлен на рисунке 121.

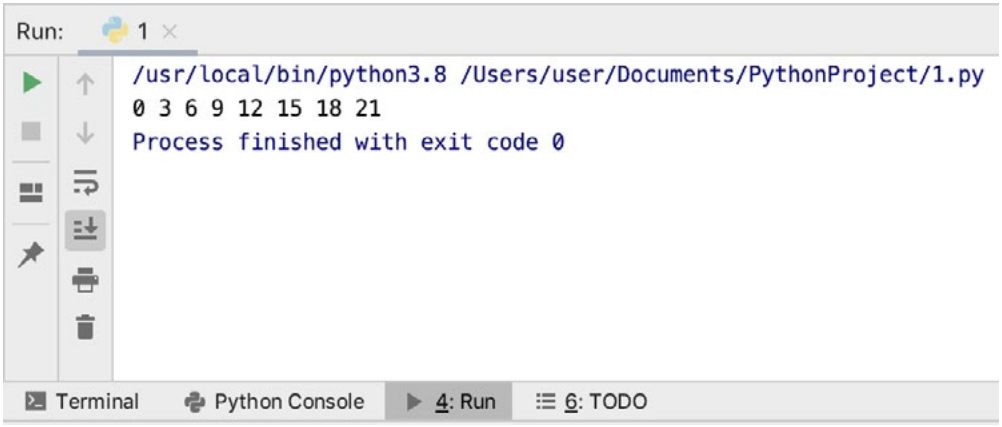

Рис. 121. Результат работы программы

Также, говоря про работу циклов в языке Python, необходимо упомянуть про операторы continue, break, else.

Оператор continue используется, чтобы перейти на следующую итерацию цикла, пропуская следующие после него операторы тела цикла.

Пример 5.

```
for i in range(10):
    if i==5:
        continue
    print(i * 2, end=› ‹)
```

Результат работы программы представлен на рисунке 122.

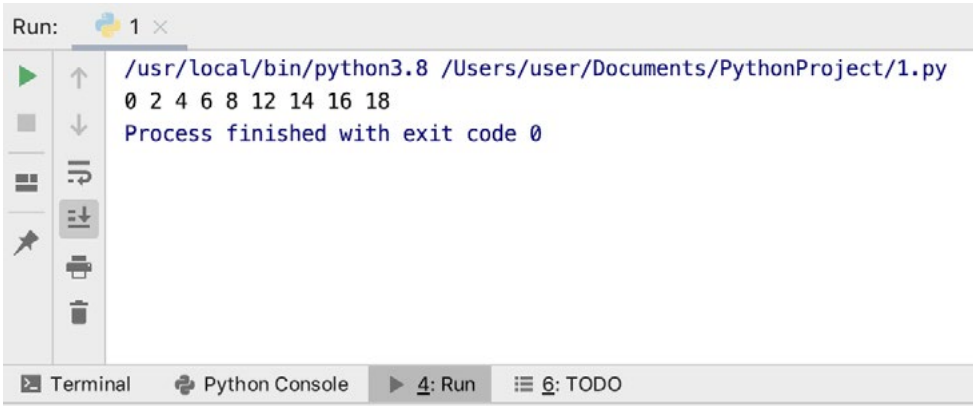

Рис. 122. Результат работы программы

В данном примере при равенстве переменной i==5 используется оператор continue, в результате чего пропускается оператор print(i * 2, end=› ‹). Поэтому число 10 не выводится на экран.

Оператор break используется для организации немедленного выхода из цикла. То есть происходит досрочное завершение работы цикла.

Пример 6.

```
for i in range(10):
    if i == 5:
        break
    print(i * 2, end=› ‹)
```

Результат работы программы представлен на рисунке 123.

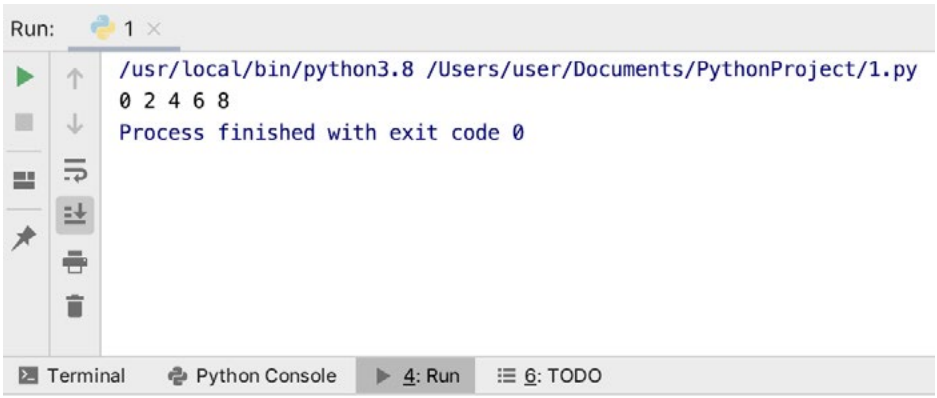

Рис. 123. Результат работы программы

В данном примере при равенстве переменной i==5 используется оператор break, в результате чего происходит завершение работы цикла. То есть последнее значение, рассмотренное в теле цикла, будет i=4.

Оператор else используется для проверки, был ли произведён выход из цикла инструкцией break, или же цикл завершился обычным образом.

Пример 7.

```
for i in range(10):
    if i == 20:
        continue
    print(i * 2, end=› ‹)
else:
print(“значение не найдено”)
```

Результат работы программы представлен на рисунке 124.

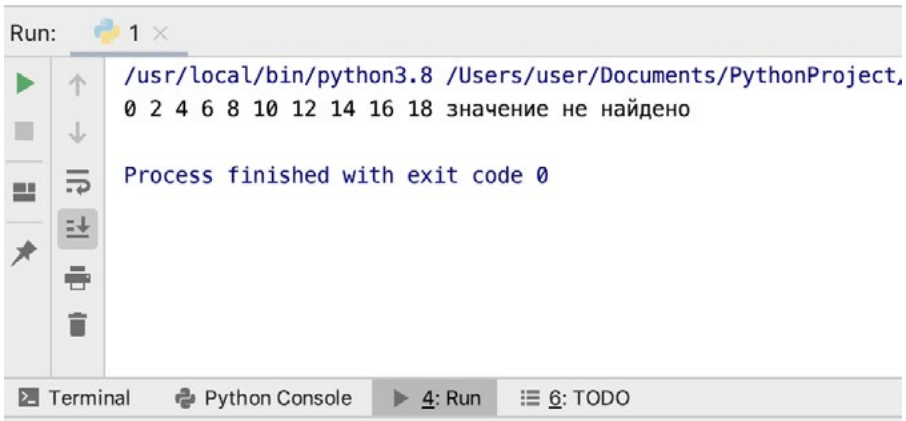

Рис. 124. Результат работы программы

Также в языке Python возможно использование вложенных циклов, когда есть один внешний цикл и один или несколько вложенных. Стоит отметить, что использование вложенных циклов может замедлить работу программы.

Приведём ещё несколько примеров работы с циклами for и while.

Пример 8. На тренировке спортсмен ежедневно пробегает некоторую дистанцию, с каждым днём увеличивая её на 10%. Составить программу, определяющую по расстоянию, преодолённому спортсменом в первый день тренировки, длину дистанции на k-й день.

```
n= float(input(«Введите начальную дистанцию «))
k= int(input(«Введите количество дней «))
for i in range (k):
    n+=n**0.1
print(“конечная дистанция “, n)
```

Результат работы программы представлен на рисунке 125.

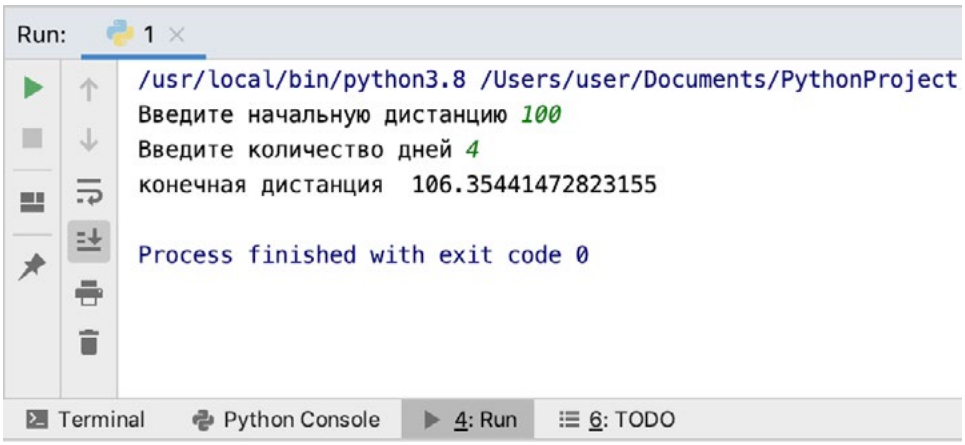

Рис. 125. Результат работы программы

В данной задаче используется цикл for, так как известно количество повторов цикла — k дней. Внутри цикла идёт увеличение переменной n, обозначающей длину дистанции, n+=n**0.1.

Пример 9. Перевести введённое пользователем десятичное число в двоичное. Известно, что число меньше 512.

```
dec=int(input(«Введите десятичное число «))
v=128
for i in range(1,9):
    if dec>=v:
        print(‘1’, end=’’)
        dec=dec–v
    else:
        print(‘0’, end=’’)
    v=v//2
```

Результат работы программы представлен на рисунке 126.

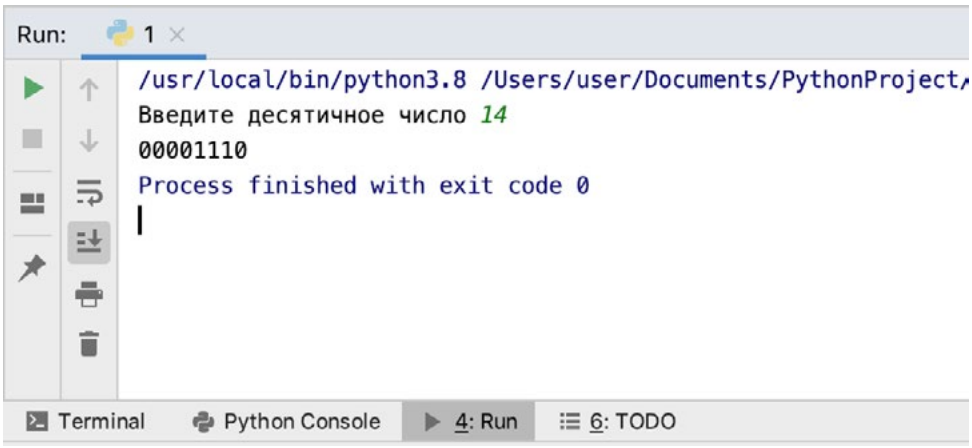

Рис. 126. Результат работы программы

В данном примере с помощью переменной v задаётся вес старшего разряда. Так как по условию число должно быть меньше 512, то старший разряд будет иметь значение 128. Для перевода десятичного числа в двоичную запись из него каждый раз вычитается старший разряд, если это возможно, затем значение разряда уменьшается.

Пример 10. Разложить натуральное число на простые множители.

```
k=int(input(«Введите число «))
print(k,'= ')
l=2
while not(k==1):
    if k%l==0:
        k=k/l
        print(l, end=' ')
    else:
        l+=1
```

Результат работы программы представлен на рисунке 127.

Рис. 127. Результат работы программы

В данном примере первое простое число l=2. В цикле while введённое k несколько раз делится на потенциальный простой делитель l, если целочисленное деление не может быть выполнено, то ищется следующий простой делитель.

III. Этап проверки понимания и первичного закрепления — 5 мин.

Учитель предлагает учащимся вспомнить кратко новый материал: что представляет собой циклический оператор for, что представляет собой циклический оператор while, в чём их различие.

IV. Этап контроля усвоения и коррекции ошибок — 5 мин.

Учитель задает вопросы ученикам:

1) Для чего нужны циклические операторы?

2) Какие можно привести примеры, когда удобно использовать циклические операторы?

V. Информация о домашнем задании, инструктаж по его выполнению — 3 мин.

VI. Этап рефлексии деятельности на уроке — 2 мин.

Учитель спрашивает учащихся об их впечатлениях от урока, что понравилось, а что было непонятно.

Лабораторная работа 4 «Циклические операторы»

Теоретическая часть

Воспользоваться материалами урока 3.

Практическая часть

Цель работы: знакомство с операторами цикла while, for в языке программирования Python.

Ход лабораторной работы

1. Откройте среду разработки Python PyCharm.
2. Население города на 2000 г. насчитывало 620 тыс. человек. Считая емп прироста населения за год равным 3,7%, определите, в каком году оно превысит 1,5 млн человек.
3. Найдите сумму нечётных делителей введённого с клавиатуры натурального числа.
4. Найдите все натуральные числа из промежутка [1; 200], у которых количество делителей равно n (где n вводится с клавиатуры).

Указание. Примерный листинг программы может иметь вид:

```
n=int(input(«Введите кол-во делителей «))
for i in range(1,201):
    k=2
    for j in range (2,i):
        if i%j==0:
            k+=1
    if k==n:
print(i)
```

5. Найдите все четырёхзначные числа, у которых сумма крайних цифр равна сумме средних (например, 3221).
6. Найдите все двухзначные числа, которые при умножении на 2 заканчиваются на 8, а при умножении на 3 — на 4.

Вывод: в ходе выполнения лабораторной работы вы получили представление о составлении циклических алгоритмов с использованием операторов while и for в языке программирования Python.

Контрольные вопросы:

1. Для чего используются циклы в языке программирования?

2. Какие виды циклов реализованы в языке Python?
3. Каков синтаксис оператора цикла while?
4. Каков синтаксис оператора цикла for?
5. Для чего используется оператор break внутри тела цикла?

Лабораторная работа 5 «Циклические операторы»

Теоретическая часть

Воспользоваться материалами урока 3.

Практическая часть

Цель работы: знакомство с операторами цикла while, for в языке программирования Python.

Ход лабораторной работы

1. Откройте среду разработки Python PyCharm.
2. Два числа называются дружественными, если каждое равно сумме делителей другого, исключая само это число. Проверьте, являются ли два введённых числа дружественными.

Указание. Примерный листинг программы может иметь вид:

```
n=int(input(«Введите первое число «))
m=int(input(«Введите второе число «))
s1=0
for i in range(1,n):
    if n%i==0:
        s1+=i
s2=0
for i in range(1,m):
    if m%i==0:
        s2+=i
if (s1==m) and (s2==n):
    print(«yes»)
else:
    print(«no»)
```

3. Выведите на экран все четырёхзначные числа, у которых первая и последняя цифры равны.
4. В процессе лечебного голодания вес пациента уменьшается на 5% за день. Определите вес пациента на k день голодания, если изначально он весил n кг.
5. Выведите на экран таблицу умножения на n, где n вводится с клавиатуры.

Вывод: в ходе выполнения лабораторной работы вы получили представление о составлении циклических алгоритмов с использованием операторов while и for в языке программирования Python.

Контрольные вопросы:

1. Какие основные операторы языка Python использовались при решении задач лабораторной работы?
2. Какой оператор используется для организации цикла с предусловием в языке Python?
3. Как организовать цикл с постусловием в языке Python?

Учебная программа факультативного курса «Разработка приложений VR/AR»

Целью программы «Разработка виртуальной и дополненной реальности» является формирование информационной культуры учащихся, соответствующей требованиям современного мира, а также формирование компетенций по работе с технологиями AR/VR.

Планируемые результаты освоения программы

Для реализации поставленной цели планируется достижение личностных, метапредметных (познавательных, регулятивных, коммуникативных УУД) и предметных результатов.

Познавательные УУД:

- развитие умения работать с информацией, представленной в различных формах (таблицы, графики, рисунки и т. д.), на различных носителях (книги, CD, периодические издания и т. д.);
- развитие умения излагать мысли в чёткой логической последовательности, анализировать ситуацию, отстаивать свою точку зрения, самостоятельно находить ответы на вопросы путём логических рассуждений;
- развитие умения делать выводы в процессе работы и по её окончании;
- формирование ключевых компетенций проектной и исследовательской деятельности.

Регулятивные УУД:

- формирование умения целеполагания;
- формирование умения оценивать учебные действия в соответствии с поставленной задачей;
- формирование умения прогнозировать предстоящую работу (составлять план);
- способность прогнозировать результаты учебных действий и анализировать их (как положительные, так и отрицательные);
- умение корректировать намеченный план действий и ставить новые цели;
- умение соотносить свои действия с планируемыми результатами;
- формирование умения осуществлять познавательную и личностную рефлексию.

Коммуникативные УУД:

- формирование собственного мнения и позиции, допуская возможность существования иных точек зрения;
- формирование умения договариваться и приходить к общему решению, в том числе в ситуации столкновения интересов;
- умение вести дискуссии, работать в группах, выступать с сообщениями;
- формирования умения выбирать и использовать способы и средства речевой деятельности, адекватные коммуникативной задаче;
- формирование умения выступать публично, формулировать мысли и отстаивать свою точку зрения.

Личностные результаты:

- формирование умения ориентироваться в информационном пространстве;
- формирование навыков ведения проекта и выбора наиболее эффективного пути решения задачи в зависимости от конкретных условий;
- формирование мотивации к обучению и целенаправленной познавательной деятельности.

Предметные результаты:

- формирование представления о дополненной, виртуальной и смешанной реальности, базовых понятиях, актуальности и перспективах технологий AR/VR;
- формирование умения активировать запуск приложений дополненной и виртуальной реальности, устанавливать их на устройство и тестировать;
- формирование представления о разнообразии, конструктивных особенностях и принципах работы AR/VR-устройств;
- формирование навыков создания AR/VR-приложений;
- формирование основ 3D-моделирования;
- формирование навыков разработки панорам 360°;
- формирование навыков программирования на языке C#;
- усвоение основных алгоритмических конструкций языка C#;
- получение навыков работы с готовыми 3D-моделями и адаптации их под свои задачи, создание несложных моделей.

Программа рассчитана на учащихся в возрасте от 12 до 15 лет, не требует предварительных знаний и входного тестирования.

Занятия проводятся в группах до 12 человек, продолжительность занятия — 45 минут, общая продолжительность программы — 36 часов.

По уровню освоения программа является общеразвивающей (базовый уровень). Она обеспечивает возможность обучения обучающихся с любым уровнем подготовки. Методика обучения ориентирована на индивидуальный подход. Для того чтобы каждый обучающийся получил наилучший результат, программой предусмотрены индивидуальные домашние задания для самостоятельного выполнения.

Формы обучения по программе: очная, очная с применением дистанционных технологий.

Форма организации деятельности: групповая, а при реализации программы с применением дистанционных технологий — персональная, материалы курса будут размещены в виртуальной обучающей среде.

Виды занятий: основной тип занятий — комбинированный, сочетающий в себе элементы теории и практики. Большинство заданий курса выполняется самостоятельно с помощью персонального компьютера и необходимых программных средств. Также программа курса включает групповые и индивидуальные формы работы обучающихся (в зависимости от темы занятия).

Методы: основы технологии SMART, кейс-метод, словесные (беседа, опрос и т. д.), метод проблемного изложения (постановка проблемы и решение её самостоятельно или группой), наглядные (демонстрация схем, таблиц, инфографики, презентаций и т. д.), практические (практические задания, анализ и решение проблемных ситуаций, показ учителем готовой модели и т. д.), метод проектов.

Содержание обучения представлено следующими модулями.

Модуль 1. «Введение в 3D-моделирование».

Модуль 2. «Технология дополненной реальности».

Модуль 3. «Технология виртуальной реальности».

Предложенная примерная программа допускает творческий, вариативный подход со стороны учителя с возможностью замены порядка тем, введения дополнительного материала, разнообразия включаемых методик проведения занятий и выбора учебных ситуаций для самостоятельной творческой деятельности учащихся. Руководствуясь данной программой, учитель имеет возможность увеличить или уменьшить степень технической сложности материала в зависимости от состава группы и конкретных условий работы.

Содержание программы позволяет учащимся сформировать базовые компетенции по работе с AR/VR-технологиями путём погружения в проектную деятельность.

Далее представим методические рекомендации для учителей по организации и проведению занятий.

1. В начале занятия рекомендуется краткое выступление с презентацией (инфографикой, учебным видеороликом и т. д.).
2. Необходимо приводить больше примеров по обсуждаемым темам.
3. Очень важно спрашивать у учащихся, какие примеры они могут привести сами из своей жизни.
4. Важно акцентировать внимание, что AR/VR — это не только игры и развлечения. Сегодня существует достаточно большой спектр областей, где применяется дополненная и виртуальная реальность. Предпочтительны примеры из сферы образования и культуры. Нужно рассказывать учащимся об интересном использовании технологии в музеях, театрах и др.
5. Сравнивайте и анализируйте готовые проекты и приложения. Приложения дополненной реальности могут быть самыми разными: от интерактивных наложений на карты и виртуальных демонстрационных залов до массивных многопользовательских квестов и шутеров.
6. Рассказывайте обучающимся о многочисленных фестивалях и хакатонах виртуальной и дополненной реальности, на которых участники генерируют и воплощают в жизнь самые невероятные идеи.
7. Время, указанное на прохождение каждого кейса, строго не регламентировано. Вероятно, что одной группе (команде) захочется моделировать, а кому-то придётся по душе создавать виртуальные туры. Здесь важна своевременная корректировка содержания модулей.
8. Основные задания являются обязательными для выполнения всеми обучающимися. Задания выполняются на компьютере с установленным соответствующим программным обеспечением.

Примерное распределение учебных часов по модулям и темам представлено в таблице.

№ п/п	Основные модули программы	Количество часов			Формы аттестации/ контроля
		Всего	Теория	Практика	
1	**Модуль 1 «Введение в 3D-моделирование»**	**12**	**3**	**9**	
1.1	Введение. Основные понятия трёхмерной графики	2	1	1	Опрос
1.2	Принципы создания 3D-моделей, виды 3D-моделирования	2	1	1	Кейс
1.3	Основы полигонального моделирования	2	1	1	Опрос
1.4	Создание 3D-модели	6	—	6	Опрос
2	**Модуль 2 «Технология дополненной реальности»**	**12**	**3**	**9**	
2.1	Технология создания дополненной реальности	2	1	1	Тестирование

Продолжение

№ п/п	Основные модули программы	Количество часов			Формы аттестации/ контроля
		Всего	Теория	Практика	
2.2	Знакомство со средой разработки Unity	3	1	2	Тестирование
2.3	Сборка и тестирование AR-приложения в Unity	3	1	2	Кейс, тестирование
2.4	Учебный проект: «Простой AR»	4	—	4	Демонстрация проектов
3	**Модуль 3 «Технология виртуальной реальности»**	**12**	**3**	**9**	
3.1	Создание проектов VR на базе интернет-технологий.	2	1	1	Тестирование
3.2	Панорамная съёмка (фото и видео) 360°	2	1	1	Опрос
3.3	Создание проектов VR на базе программного обеспечения	4	1	3	Кейс
3.4	Учебный проект «Приложение AR»	4	—	4	Демонстрация проектов
	ВСЕГО	**36**	**9**	**27**	

Модуль 1 «Введение в 3D-моделирование».

Цель: знакомство с основами 3D-моделирования.

Содержание:

- принципы создания 3D-моделей, виды 3D-моделирования;
- анализ 3D-графических пакетов для моделирования;
- разработка 3D-модели, покраска и текстурирование модели.

Модуль 2 «Технология дополненной реальности».

Цель: изучение технологии дополненной реальности.

Содержание:

- история и тенденции развития AR, использование в различных сферах деятельности человека;
- основные понятия AR;
- мобильные приложения для AR-проектов;
- знакомство с межплатформенной средой разработки компьютерных игр Unity 3D;
- знакомство с материалами и текстурами Unity, базовая физика;
- основы программирования на C# в Unity;
- этапы разработки AR-приложения.

Модуль 3 «Технология виртуальной реальности».

Цель: изучение принципов работы с VR.

Содержание:

- предпосылки, история, области применения систем виртуальной реальности;

- основные понятия, принципы и инструментарии разработки систем VR, а также оборудование для их реализации;
- панорамная съёмка (фото и видео) 360°;
- этапы и технологии создания систем VR, структура и компоненты;
- обзор современных 3D-движков: основные понятия, возможности, условия использования, сравнительный анализ;
- создание приложения для VR-устройств.

Тематическое планирование с определением основных видов деятельности

№ п/п	Тема	Содержание	Целевая установка урока	Кол-во часов	Основные виды деятельности обучающихся на уроке/внеурочном занятии	Использование оборудования
	Модуль 1 «Введение в 3D-моделирование»					
1.1	Введение. Основные понятия трёхмерной графики	Общее представление о работе с программами 3D-моделирования. Сравнительный анализ программ и их возможностей, выявление наиболее выгодных возможностей программ, их функции и особенности.	Изучение основных понятий 3D-моделирования, обзор программ для 3D-моделирования	2	Слушают объяснение учителя. Наблюдают за демонстрациями учителя. Выполняют лабораторные работы	Компьютер, проектор, интерактивная доска
1.2	Принципы создания 3D-моделей, виды 3D-моделирования	Этапы создания 3D-модели, из чего состоят, где применяется 3D-моделирование. Рассмотрение существующих стандартных моделей на различных информационных ресурсах, проверка работоспособности моделей, их уровень качества и возможности видоизменения.	Знакомство с этапами создания 3D-моделей и видами 3D-моделирования	2	Слушают объяснение учителя. Выполняют лабораторные работы	Компьютер, проектор, интерактивная доска
1.3	Основы полигонального моделирования	Разбор интерфейса и логики создания моделей в контексте полигонального моделирования, основных функций программы. Камера и рендеринг. Настройка рабочего окна, создание примитивных моделей	Изучение основ работы программ для полигонального моделирования	2	Слушают объяснение учителя. Выполняют лабораторные работы	Компьютер, проектор, интерактивная доска

Продолжение

№ п/п	Тема	Содержание	Целевая установка урока	Кол-во часов	Основные виды деятельности обучающихся на уроке/внеурочном занятии	Использование оборудования
1.4	Создания 3D-модели	Создание стандартных и видоизменённых моделей. Фотореалистичная визуализация 3D-модели	Формирование умения создавать 3D-модели	6	Слушают объяснение учителя. Выполняют лабораторные работы	Компьютер, проектор, интерактивная доска
	Модуль 2 «Технология дополненной реальности»					
2.1	Технология создания дополненной реальности	Обзор AR-библиотек и плагинов для создания приложений с дополненной реальностью.	Изучение популярных AR-библиотек и плагинов	2	Слушают объяснение учителя. Выполняют лабораторные работы. Объясняют наблюдаемые явления	Компьютер, проектор, интерактивная доска, очки дополненной реальности, смартфон, веб-камера, МФУ
2.2	Знакомство со средой разработки Unity	Программа Unity, интерфейс, основные инструменты. Особенности установки программы и работы с ней. Создание и настройка сцены для работы с дополненной реальностью. Работа с видео в Unity. Импорт объектов из 3D-редакторов в Unity. Особенности, основные проблемы Unity и способы их решения	Знакомство с программой Unity	3	Слушают объяснения учителя. Выполняют лабораторные работы. Программируют	Компьютер, проектор, интерактивная доска, очки дополненной реальности, смартфон, веб-камера, МФУ
2.3	Сборка и тестирование приложения AR в Unity	Создание простейшего AR-приложения в Unity. Настройка анимации 3D-модели в Unity и исполь-	Организация деятельности учащихся по разработке	3	Слушают объяснения учителя. Выполняют лабораторные работы. Программируют	Компьютер, проектор, интерактивная

В содержание

		зование в AR. Сборка приложения AR в Unity для мобильных устройств на базе Android	AR-приложений в Unity			доска, очки дополненной реальности, смартфон, веб-камера, МФУ
2.4	Учебный проект «Приложение AR»	Разработка индивидуального или группового проекта. Подготовка к презентации и защите работы	Проверка полученных навыков разработки приложений AR.	4	Слушают объяснение учителя, моделируют и конструируют. Редактируют программы. Слушают и анализируют выступления	Компьютер, проектор, интерактивная доска, очки дополненной реальности, смартфон, веб -камера, МФУ
	Модуль 3 «Технология виртуальной реальности»					
3.1	Создание проектов VR на базе интернет-технологий	VR-устройства, их конструктивные особенности управления	Формирование представления о создании VR-приложений на базе интернет-технологий	2	Слушают объяснение учителя. Выполняют лабораторную работу. Анализируют проблемные ситуации	Компьютер, проектор, интерактивная доска, шлем виртуальной реальности (любительский, профессиональный), смартфон
3.2	Панорамная съёмка (фото и видео) 360°	Информация о видах оборудования панорамной съёмки 360°, история появления технологий. Программа монтажа панорамных роликов	Изучение основных понятий «360°» и принципов работы программ видео монтажа панорамных роликов	2	Слушают объяснение учителя. Выполняют лабораторную работу. Объясняют наблюдаемые явления	Компьютер, проектор, интерактивная доска, шлем виртуальной реальности (любительский, профессиональный), смартфон

Продолжение

№ п/п	Тема	Содержание	Целевая установка урока	Кол-во часов	Основные виды деятельности обучающихся на уроке/внеурочном занятии	Использование оборудования
3.3	Создание проекта VR на базе программного обеспечения	Создание первого проекта VR в Unity	Организация деятельности учащихся по разработке VR-приложений в Unity	4	Слушают объяснение учителя. Выполняют лабораторные работы. Программируют	Компьютер, проектор, интерактивная доска, шлем виртуальной реальности (любительский, профессиональный), смартфон
3.4	Учебный проект «Приложение VR»	Разработка индивидуального или группового проекта. Подготовка к презентации и защите работы	Проверка полученных навыков разработки приложений VR	4	Слушают объяснение учителя, моделируют и конструируют. Редактируют программы. Слушают и анализируют выступления своих товарищей	Компьютер, проектор, интерактивная доска, шлем виртуальной реальности (любительский, профессиональный), смартфон
	Итого			**36**		

В содержание

Учебная программа факультативного курса «Основы робототехники»

Целью программы «Основы робототехники» является развитие алгоритмического мышления обучающихся, их творческих способностей, аналитических и логических компетенций, а также создание основы для будущего изучения программирования роботов на одном из современных языков.

Планируемые результаты освоения программы

Для достижения поставленной цели планируется достижение личностных, метапредметных (познавательных, регулятивных, коммуникативных УУД) и предметных результатов.

Познавательные УУД:

- развитие пространственного воображения, логического и визуального
- мышления, наблюдательности, креативности;
- овладение умением применять теоретические знания на практике.

Регулятивные УУД:

- выработка навыков планирования и определения последовательности промежуточных целей с учетом конечного результата;
- овладение способами планирования и организации созидательной деятельности;
- освоение способов контроля в форме сопоставления способа действия и его результата с заданным образцом с целью обнаружения отличий.

Коммуникативные УУД:

- выработка умения работать над проектом в команде;
- выработка умения эффективно распределять обязанности в группе.

Личностные результаты:

- формирование профессионального самоопределения;
- формирование уважительного отношения к интеллектуальному труду;
- формирование первоначальных представлений о профессиях, в которых информационные технологии играют ведущую роль;
- воспитание интереса к информационным технологиям;
- формирование личностного самоопределения.

Предметные результаты:

- начальное освоение компьютерной среды Scratch в качестве инструмента программирования роботов;
- создание завершённых проектов с использованием освоенных навыков структурного программирования;
- знакомство с основами робототехники с помощью универсальной робототехнической платформы VEXcode VR или аналогичной (виртуальной или реальной);
- систематизация знаний по теме «Алгоритмы» на примере работы программной среды Scratch с использованием блок-схем программных блоков;
- овладение трудовыми умениями и навыками при работе с платформой (конструктором), приобретение опыта практической деятельности по созданию автоматизированных систем управления, полезных для человека и общества;
- усвоение знаний о роли автоматизированных систем управления в пре-
- образовании окружающего мира;
- формирование алгоритмического мышления через составление алгоритмов в компьютерной среде Scratch VEXcode VR.

Программа рассчитана на учащихся в возрасте от 12 до 15 лет, не требует предварительных знаний и входного тестирования. Занятия проводятся в группах до 12 человек, продолжительность занятия — 45 минут, общая продолжительность программы — 36 часов.

Внутренняя структура задач освоения навыков программирования допускает модульную организацию программы. Предлагается следующий набор учебных модулей.

Модуль 1 «Знакомство с платформой VEXcode VR».
Модуль 2 «Программирование робота на платформе».
Модуль 3 «Датчики и обратная связь».
Модуль 4 «Организация алгоритмов движения робота».
Модуль 5. «Творческий проект».
Модуль 6.«Дальнейшее развитие».
Поэтому достижение **предметных результатов** можно указать к отдельным модулям.

Модуль 1 «Знакомство с платформой VEXcode VR»

В результате изучения данного модуля учащиеся будут:
знать названия различных компонентов робота и платформы:
— контроллер (специализированный микрокомпьютер);
— исполнительные устройства (мотор, колёса, перо, электромагнит);
— датчики (цвета, расстояния, местоположения, касания);
— панель управления, ракурсы наблюдения робота;
— программные блоки по разделам;
— виды игровых полей (площадок);
— кнопки управления;
уметь:
— программировать управление роботом;
— использовать датчики для организации обратной связи и управления роботом;
— сохранять и загружать проект;

Модуль 2 «Программирование робота на платформе».

В результате изучения данного модуля учащиеся будут:
знать:
— математические и логические операторы;
— блоки вывода информации в окно вывода;
уметь:
— применять на практике логические и математические операции;
— использовать блоки для работы с окном вывода;
— составлять с помощью блоков математические выражения.

Модуль 3 «Датчики и обратная связь».

В результате изучения данного модуля учащиеся будут:
знать:
— принципы работы датчиков;
— блоки управления датчиками;
— возможности датчиков;
уметь:
использовать циклы и ветвления для реализации системы принятия решений;
решать задачу «Лабиринт».

Модуль 4 «Реализация алгоритмов движения робота».

В результате изучения данного модуля учащиеся будут:

знать:
- условный оператор if/else;
- цикл while;
- понятие шага цикла;

уметь:
- применять на практике циклы и ветвления;
- использовать циклы и ветвления для решения математических задач;
- использовать циклы для объезда повторяющихся траекторий.

Модуль 5 «Творческий проект»

При выполнении творческих проектных заданий учащиеся будут разрабатывать свои собственные программы.

Перечень используемого оборудования и материалов:
- рабочее место для работы с компьютером;
- компьютер с ОС Windows и выходом в Интернет;
- рабочая тетрадь ученика.

Модуль 6 «Дальнейшее развитие»

При выполнении творческих проектных заданий учащиеся будут разрабатывать свои собственные программы на языке Си.

Перечень используемого оборудования и материалов:
- рабочее место для работы с компьютером;
- компьютер с ОС Windows и выходом в Интернет;
- рабочая тетрадь ученика.

Метапредметные результаты

I. Технологический компонент

Регулятивные универсальные учебные действия:
- освоение способов решения проблем творческого характера в жизненных ситуациях;
- формирование умений ставить цель создания творческой работы, планирование достижения этой цели, создание вспомогательных эскизов в процессе работы;
- оценивание создаваемого творческого продукта и соотнесение его с изначальным замыслом, выполнение при необходимости коррекции либо самого продукта, либо его замысла.

Познавательные универсальные учебные действия:
- поиск информации учащимся в различных источниках: своих архивах, информационной среде образовательного учреждения, в федеральных и региональных хранилищах информационных образовательных ресурсов;
- использование средств цифровых технологий для решения коммуникативных, познавательных и творческих задач.

Коммуникативные универсальные учебные действия:
- умение подготовить и провести выступление.

II. Логико-алгоритмический компонент

Регулятивные универсальные учебные действия:
- планирование последовательности шагов алгоритма для достижения цели;
- поиск ошибок в плане действий и внесение в него изменений.

Познавательные универсальные учебные действия:
- моделирование — преобразование объекта из чувственной формы в модель, где выделены существенные характеристики;

— анализ объектов с целью выделения признаков (существенных, несущественных);
— синтез — составление целого из частей, в том числе самостоятельное достраивание с восполнением недостающих компонентов;
— установление причинно-следственных связей;
— построение логической цепи рассуждений.

Коммуникативные универсальные учебные действия:

— аргументирование своей точки зрения на выбор способов решения поставленной задачи;
— выслушивание собеседника и ведение диалога.

Личностные результаты

К личностным результатам освоения навыков программирования роботов относятся:

— осмысление мотивов своих действий при выполнении заданий с жизненными ситуациями;
— начало профессионального самоопределения, ознакомление с миром профессий, связанных с информационными и коммуникационными технологиями.

Тематическое планирование с определением основных видов деятельности

№ п/п	Тема	Содержание	Целевая установка урока	Кол-во часов	Основные виды деятельности обучающихся на уроке/внеурочном занятии	Использование оборудования
1	Модуль 1 «Знакомство с платформой VEXcode VR»	Основные фрагменты интерфейса платформы. Панель управления, блоки программы, датчики, игровая площадка, экран датчиков и переменных, кнопки управления. Создание простейших программ (скриптов), сохранение и загрузка проекта	Знакомство учащихся с интерфейсом платформы, принципами программирования виртуального робота, видами игровых полей (площадок), основными блоками управления.	3	Наблюдение за работой преподавателя, самостоятельная работа со средой, ответы на контрольные вопросы	Виртуальная среда VEXcode VR
2	Модуль 2 «Программирование робота на платформе»	Математические и логические операторы, блоки вывода информации в окно вывода, блоки трансмиссии. Блоки управления, блоки переменных, блоки датчиков, блоки вида, магнит	Знакомство учащихся с блоками логических и математических операторов, приёмы работы с ними. Организация движения робота с помощью блоков трансмиссии. Применение блоков переменных. Основные виды датчиков. Применение магнита	4	Наблюдение за работой преподавателя, самостоятельная работа со средой, ответы на контрольные вопросы	Виртуальная среда VEXcode VR
3	Модуль 3 «Датчики и обратная связь»	Датчик местоположения, направление движения. Датчики цвета. Дисковый лабиринт. Датчик расстояния. Простой лабиринт. Динамический лабиринт	Знакомство с основными видами датчиков и принципами их работы. Применение датчиков в различных игровых полях. Создание скриптов для прохождения простого	10	Наблюдение за работой преподавателя, самостоятельная работа со средой, ответы на контрольные вопросы	Виртуальная среда VEXcode VR

Продолжение

№ п/п	Тема	Содержание	Целевая установка урока	Кол-во часов	Основные виды деятельности обучающихся на уроке/внеурочном занятии	Использование оборудования
		Управление магнитом. Сбор фишек	и динамического лабиринтов. Программа сбора фишек с помощью магнита и размещение их по цветам			
4	Модуль 4 «Организация алгоритмов движения робота»	Блок команд «Управление» и организация циклов и ветвлений. Проекты «Разрушение замка» и «Динамическое разрушение замка» Проект «Детектор линии»	Подробный разбор блока команд «Управление» и создание скриптов для реализации различных проектов игровых полей	10	Наблюдение за работой преподавателя, самостоятельная работа со средой, ответы на контрольные вопросы	Виртуальная среда VEXcode VR
5	Модуль 5 «Творческий проект»	Создание собственного проекта с использованием максимально возможного количества датчиков	Создание собственного проекта на основе полученных знаний по работе с платформой	4	Наблюдение за работой преподавателя, самостоятельная работа со средой, ответы на контрольные вопросы	Виртуальная среда VEXcode VR
6	Модуль 6 «Дальнейшее развитие».	Основы программирования роботов на языке Си. Простейшие программы для роботов	Используя полученные знания, учащиеся знакомятся с принципами программирования роботов в текстовом редакторе RobotC на языке программирования Си	5	Наблюдение за работой преподавателя, самостоятельная работа со средой, ответы на контрольные вопросы	Виртуальная среда VEXcode VR
			Итого:	36		

В содержание

Материалы для организации и проведения учебно-исследовательской и проектной деятельности школьников

Особенности учебно-исследовательской и проектной деятельности школьников

Важное место в учебно-исследовательской и проектной деятельности школьников занимают организация и проведение конкурсов, олимпиад, соревнований, лекториев, чемпионатов, хакатонов, интенсивов, мастер-классов, воркшопов, фестивалей. Проходить они могут в очном и дистанционном режимах по направлениям: конструирование, моделирование, робототехника, программирование, дополненная и виртуальная реальность и др. Реализация таких мероприятий позволяет повысить интерес школьников к профессиям IT-сферы, и создаёт условия для мотивации обучающихся к участию в предпрофессиональных пробах в сфере информационных технологий.

Фактором успешного освоения знаний учащихся служат результаты соревнований, в которых важно не только обладать необходимыми знаниями и навыками, а также работать в команде, подстраиваться под изменяющиеся условия, применять свои знания на практике в стрессовых ситуациях.

Соревновательные мероприятия должны включать в себя:

— ознакомление с регламентами и заданиями;

— подготовку ресурсов (конструкторы, необходимое оборудование и программное обеспечение и т. д.);

— поэтапное выполнение предложенных кейсов по мере их усложнения или в зависимости от возраста участников;

— формирование команд;

— участие в соревнованиях (сначала в рамках кабинета, затем районный и городской, зональный и региональный уровни);

— рефлексию.

Содержание соревнований и факторы, обусловливающие высокий результат, служат основным ориентиром при планировании подготовки участников.

В последнее время всё более распространённым форматом мероприятий становится хакатон. Это ограниченное по времени соревнование, в течение которого участники в составе команд до 5 человек создают прототипы цифровых решений в рамках поставленных задач. За каждой командой закрепляется свой наставник. Как правило, задачей хакатона является создание полноценного программного обеспечения, но существуют хакатоны, которые предназначены для образовательных или социальных целей. Они сфокусированы на решении проблем в определённой области.

Этапы проведения хакатона:

1. Открытие хакатона.
2. Постановка задач предложенных для разработки проектов.
3. Формирование команд.
4. Командная работа над проектами (основной этап).
5. Презентация проектов.
6. Если хакатон носит соревновательный характер, то работы оцениваются и определяются победители.

Рассмотрим подробнее некоторые этапы подготовки и проведения соревнований.

Основные принципы формирования команды участников соревнований. К участию в соревновании допускаются учащиеся образовательных организаций, учреждений дополнительного образования и учащиеся различных центров цифрового образования «IT-Куб» Российской Федерации в возрасте от 7 до 18 лет, которые в зависимости от вида соревнований делятся на возрастные категории, далее оцениваются в рамках своей возрастной категории.

Обычно участники соревнований разделяются на следующие возрастные группы:

I группа — 1—4 классы;

II группа — 5—6 классы;

III группа — 7—8 классы;

IV группа — 9—11 классы.

Для различных возрастных групп участников организуются следующие соревнования:

1—4 классы: программирование на Scratch, робототехника.

5—6 классы: программирование на Scratch, программирование на Python, робототехника.

7—8 классы: программирование на Scratch, программирование на Python, программирование на Java, робототехника, анализ Больших данных, системное администрирование, разработка приложений VR/AR, 3D-моделирование.

9—11 классы: мобильная разработка, программирование на Python, программирование на Java, разработка приложений VR/AR, системное администрирование, разработка дизайна интерфейса приложений.

Участники, прошедшие предварительный отбор, могут самостоятельно создать проектную команду в количестве до 5 человек (в зависимости от выбранного направления) либо попасть в команду, созданную организационным комитетом соревнований.

Организация и проведение отборочных туров соревнований. Перед началом соревнований проводится отборочный этап соревнований (в формате онлайн или офлайн). Он может включать ознакомительные лекции, разбор кейсов, мастер-классы, образовательные интенсивы. Также предоставляется возможность общения с экспертами и спикерами.

Соревнования можно проводить в два этапа: теоретический и практический. На теоретическом (отборочном) этапе предстоит решить предложенную организаторами задачу. Участники, которые справятся с теоретическим заданием, принимают участие в практическом (основном) этапе.

Пример материалов для проведения межшкольных соревнований по программированию

В рамках программы реализуются два направления программирования: Scratch, Python. Приведем примеры материалов для проведения межшкольных соревнований по программированию на различных языках.

В соревнованиях могут принимать участие все желающие в возрасте до 14 лет. Они проводятся по номинациям: игровое приложение; социальный мультфильм; приложение, позволяющее на основе данных, введённых пользователем, решить оптимизационную задачу; анимация статичных рисованных рисунков или фотографий.

Работа должна содержать: законченное приложение или обучающий мультфильм с возможностью просмотра исходного кода.

Работы оцениваются по следующим критериям (примерный список):

— актуальность и практическая значимость разработки;

- работоспособность (программа работает без сбоев);
- насыщенность элементами мультимедийности;
- наличие анимации объектов;
- понятный и законченный сюжет истории;
- наличие титров;
- сложность программирования;
- креативность подхода (создание собственных спрайтов, фонов);
- соответствие теме соревнований.

Для участников соревнований по программированию на **Scratch** могут быть предложены следующие кейсы.

Кейс 1. Обыграть известную сказку «Теремок» с новыми персонажами и в новых условиях: «Стоит в поле теремок. Он не низок, не высок. В нём живет Лесовичок. Скучно Лесовичку стало жить одному в большом доме, и решил он найти друзей и пригласить их к себе жить. Лесовичок очень любил мультфильмы, и поэтому электронное письмо-приглашение направил героям 14 мультфильмов».

Кейс 2. Проект «Знайки». Необходимо подготовить фон и героев для проекта. Участник выбирает одну из предметных/межпредметных областей: «Математика», «История», «Робототехника» и т. д., придумывает сюжет, создаёт персонажи, выполняющие роль ведущих викторины, составляет разные типы вопросов, программирует счётчик правильных/неправильных ответов. Необходимо продумать звуковое сопровождение к проверке ответа игрока (например, аплодисменты).

Кейс 3. Разработать приложение, позволяющее рассчитать оптимальную загрузку корабля и максимальную прибыль от миссии. Требуется разработать приложение (desktop/mobile/web на выбор) с использованием элементов GUI, позволяющее на основе данных, введённых пользователем, решить оптимизационную задачу. Пользователь добавляет предметы, которые он хотел бы видеть на корабле.

Описание предмета состоит из трёх частей: название, вес (тонны), стоимость (красные камни). Пример: книги, 10, 10. После чего при нажатии на кнопку «Укомплектовать» должен появиться список предметов, в котором прибыль от доставки будет максимальна, а их общий вес не превысит грузоподъёмность корабля. В случае, когда невозможно провести расчёт (нет элементов, общий вес превышает допустимый, стоимость доставки меньше минимальной), пользователь должен получить соответствующее предупреждение. Максимальная грузоподъёмность корабля 30 тонн. Миссия не может быть отправлена, если планируемый доход меньше 500 красных камней.

Кейс 4. «STREAM-проект». Создать электронный образовательный ресурс на стыке Science, Technology, Robotik, Engineering, Art, Mathematics. Участник разрабатывает на выбор: интерактивную модель реального процесса или явления; тренажёр с диагностикой навыка пользователя; обучающий квест и т. д.

Для участников соревнований по программированию на **Python** могут быть предложены следующие кейсы.

Кейс 1. Запрограммировать графический редактор. Предусмотреть рисование нескольких графических объектов мышью. Реализовать дополнительную возможность рисовать с помощью клавиатуры, реализовать возможность выбора пера.

Кейс 2. Создать мобильное приложение и/или сайт с картой доступных общественных пространств (смарт-офисы, коворкинги, переговорные комнаты, спортивные залы и площадки на открытом воздухе) с указанием характеристик и условий бронирования. Возможность добавления пользователем отзывов, построения навигации до выбранной точки общественного пространства. Возможность получения информации офлайн.

Кейс 3. Разработать приложение, с помощью которого можно запрашивать данные о местоположении пунктов доставки различных сервисов, интернет-магазинов с возможностью фильтрации по типу и т. д.

Для оценивания кейсов по программированию могут быть использованы следующие критерии.

Критерии	Оценка (0—5 баллов)
Соответствие конкурсной работы заявленной номинации	
Оригинальность идеи и содержание проекта	
Творческий подход	
Сложность проекта	
Качество исполнения: понятность интерфейса, дизайн, удобство структуры и навигации, качество алгоритма	
Использование интерактивных возможностей	
Умение планировать совместную работу	
Дополнительные задачи	
Отсутствие ошибок в программе	
Сложность проекта	
Умение объяснить и защитить свои идеи	
Всего:	

Пример материалов для проведения межшкольных соревнований по робототехнике

Соревнования по робототехнике делятся на два направления: регламентированные и творческие. Регламентированные предусматривают заранее известное задание и строгие критерии оценивания (движение по линии, выполнение определённого количества заданий, скорость выполнения). В творческом направлении регламентируется только тематика задания и форма представления работы. Дальше происходит разработка творческого проекта или решения. Ребёнок, находясь в таких условиях, имеет возможность выбирать то, что для него интереснее.

Теоретический тур — участникам необходимо выполнить тестовые задания, связанные с робототехникой. Время выполнения заданий — N минут с момента публикации задания. Данный тур проводится для всех возрастных групп.

Конструкторский тур — в этом туре необходимо выполнить определённое задание в виртуальном конструкторе. Время выполнения заданий — N минут с момента публикации задания. Данный тур проводится для средней и старшей групп.

Программный тур младшей группы — в этом туре необходимо написать программы под конкретные задачи от организаторов.

Программный тур для старшей и средней групп — в этом туре необходимо написать программу для робота, собранного организаторами.

В соревнованиях принимают участие дети от 7 лет. Команды допускаются к участию по решению оргкомитета. Конкретные возрастные ограничения оговариваются для каждой номинации. На соревновательных мероприятиях чаще всего обучающиеся делятся на команды, где каждый участник отвечает за выполнение определённой работы в зависимости от его индивидуальных возможностей. Один может выступать как программист, отвечающий за написание программы работы робота, другой — за творческую составляющую, третий — за яркую презентацию их совместного проекта. Работа в таких условиях даёт обучающимся возможность как развить свои способности, так и совершенствовать менее развитые навыки.

Существует большой перечень необычных регламентов соревнований, творческих направлений, из которых обучающийся может выбирать необходимое. Работа в коллективе и в уникальных условиях соревновательных мероприятий способствует развитию ребёнка.

Соревнования по робототехнике имеют ряд особенностей.

- Зрелищность: участники видят продуктивную работу своих сверстников, передовые инженерно-технические достижения, новые решения в области робототехники.
- Состязательность: позволяет выявить наиболее подготовленную команду, способную оперативно решить поставленную тренером (организатором) задачу.
- Азартность: стремление участников к лидерству, быстрому решению поставленной задачи как нельзя лучше проявляется во время соревнований по робототехнике.

При подготовке к соревнованиям предусматривается работа в малых группах. Допускается работа в разновозрастных группах, которые могут быть сформированы в зависимости от способностей обучающихся.

Соревнования по робототехнике можно проводить по следующим категориям:

— «Юный робототехник»;
— «Робо-баскетбол» (биатлон) (Lego и Arduino);
— «Гонки по линиям» (роботы, построенные на основе наборов Lego и линейки микроконтроллеров Arduino);
— «Творческая категория»;
— «Решение задач Arduino»;
— «Шагающие роботы»;
— «Траектория-квест»;
— «Лабиринт» и т. д.

Возраст участников определяется регламентом конкретной номинации соревнования.

Категория 1 «Робо-баскетбол».

Состав команды — не более двух человек.

1. Игровое поле

1.1. Игровое поле для «Робо-баскетбола» имеет размеры 2100 ´ 1100 мм.

1.2. Высота расположения корзины — 200 мм, диаметр корзины 230 мм.

2. Роботы

2.1. Роботы должны быть построены только из фирменных элементов, моторов и датчиков LEGO.

2.2. Роботы будут измеряться в вертикальном положении, при этом они не должны ни на что опираться и их подвижные части должны быть максимально выдвинуты.

2.3. Высота робота должна составлять не более 350 мм.

2.4. В конструкции роботов допускается наличие третьего мотора.

2.5. Участники состязания должны оформить своего робота (обозначить метками, украсить) так, чтобы была видна его принадлежность к команде. Это не должно влиять на игровой процесс. Оформление робота также не попадает под ограничение по высоте.

2.6. Роботы должны работать с помощью пульта дистанционного управления (допускается использование планшетов, смартфонов или других пультов дистанционного управления, подключённых к роботам посредством Bluetooth или Wi-Fi).

3. Правила поведения участников

3.1. Решение судьи является окончательным. Счёт матча остаётся неизменным при любых обстоятельствах.

3.2. Участники и тренеры команд должны придерживаться такого поведения, которое не затрудняет проведение мероприятия.

3.3. Организаторы не несут ответственности за поломку роботов.

4. Команда роботов

4.1. Замены роботов строго запрещены. Команда участников, заменившая робота, будет отстранена от участия в состязании.

4.2. Разрешено в перерывах между матчами совершенствовать своего робота, менять конструкцию.

5. Схема проведения отборочных состязаний

5.1. Состязание состоит из индивидуальных матчей, каждый из которых длится до пяти забитых мячей в корзину, но не более четырёх минут (мяч представляет собой шарик для игры в настольный теннис). Порядок выступления участников определяется по номеру регистрации в день соревнований.

5.2. Судья фиксирует время каждого забитого мяча и заносит его в таблицу.

№ п/п	Название команды	Время 1-го забитого мяча	Время 2-го забитого мяча	Время 3-го забитого мяча	Время 4-го забитого мяча	Время 5-го забитого мяча	Место

5.3. Робот изначально находится в исходной точке (квадрат 30 × 30 см) с мячом в ковше. Расположение точки будет определено в день соревнований. Кроме этого, между роботом и корзиной расположены два препятствия (ширина 5 см, высота 10 см). Задача игрока: управляя роботом, объехать эти препятствия и забросить мяч в корзину.

5.4. В случае опрокидывания препятствия игроку необходимо вернуться в исходную точку. Судья возвращает препятствие на место, участник продолжает игру.

5.5. В случае опрокидывания робота и в течение 30 секунд невозможности продолжить игру участник выбывает.

5.5. После броска в корзину, независимо от результата, робот отправляется в исходную точку. Участник кладёт мяч роботу руками только при нахождении робота СТРОГО в исходной точке.

6. Определение победителя первого тура

6.1. По итогам первого тура составляется рейтинг команд на основании количества забитых мячей.

Лучшие восемь команд выходят в 1/4 финала и определяются по лучшему времени. В случае совпадения результатов у двух или нескольких команд места выхода в следую-

щий этап между ними распределяются по лучшему времени забитого первого, второго, третьего и т. д. мяча.

6.2. 1/4 финала.

В 1/4 финала команда с лучшим результатом по итогам первого тура будет играть с командой с худшим результатом. Все остальные пары соперников образуются соответственно. Матчи будут проходить одновременно на двух полях до четырёх забитых мячей. Перед матчем команды бросают жребий для выбора поля. Победившая команда проходит в полуфинал, проигравшая выбывает из соревнований. Участники 1/4 финала заносятся в таблицу.

Место в отборочном туре	**Название команды**	**Победитель**	**Номер в полуфинальной таблице**
1			1
8			
2			2
7			
3			3
6			
4			4
5			

6.3. Полуфинал

Полуфинальные матчи проводятся в соответствии с таблицей.

					Название команды	Место
№ п/п	Название команды					
1						
2						
3					Название команды	Место
4						

Матчи будут проходить одновременно на двух полях до четырёх забитых мячей. Перед матчем команды кидают жребий для выбора поля. Победившие команды проходят в финал, где на тех же условиях разыграют 1-е и 2-е места, проигравшие команды соответственно будут играть за 3-е и 4-е места.

Категория 2 «Гонки по линиям» (роботы, построенные на основе наборов Lego и линейки микроконтроллеров Arduino).

В состязаниях участвуют учащиеся 9—15 лет.

Цель робота — за минимальное время проехать по линии N полных кругов (количество кругов определяет главный судья в день соревнований). Движение осуществляется в направлении по часовой стрелке.

Круг — полный проезд роботом трассы, с возвращением в место старта, пересекая при этом линию старта-финиша.

Размеры игрового поля — 3700 × 2400 мм.

Поле представляет собой белое основание с чёрной линией траектории.

Линии на поле могут быть прямыми, дугообразными, пересекаться под прямым углом.

Толщина чёрной линии — 45—52 мм.

Команда является на соревнования с готовым (собранным) роботом.

В состязании принимают участие только роботы, построенные на основе линейки микроконтроллеров Arduino.

Максимальные размеры робота — 200 × 200 × 200 мм.

Во время заезда робот не может изменять свои размеры.

Допускается использование только одного контроллера в конструкции робота.

Движение роботов начинается после команды судьи и нажатия оператором кнопки или с помощью датчика, при этом робот стоит на полигоне.

Правила проведения состязаний

1. Квалификационные заезды.

1.1. Квалификационный заезд состоит из двух попыток прохождения трассы, в нём участвует один робот. Заезд останавливается судьёй, если робот не может продолжить движение в течение 15 секунд или время прохождения трассы превышает 90 секунд.

1.2. Заезд на квалификационном этапе состоит из одного полного круга.

1.3. Время прохождения трассы фиксируется судьёй состязания.

1.4. Если робот сходит с дистанции (оказывается всеми колесами с одной стороны линии), то он снимается с заезда и заезд не засчитывается. Победитель определяется по лучшему времени из двух заездов.

1.5. В случае одинакового лучшего времени у нескольких участников, победитель определяется по суммарному времени двух заездов.

2. Финальные заезды

2.1. В финальных заездах участвуют четыре робота, показавшие лучшие результаты в квалификационных заездах.

2.2. Финальный заезд состоит из одной попытки и составляет два круга.

2.3. Три победителя определяются по лучшему времени прохождения трассы.

2.4. Если во время финального заезда робот сходит с трассы, то призовые места распределяются между участниками, прошедшими финальный заезд с наименьшим временем.

Категория 3 «Творческая категория» («Робототехническое творчество»).

К участию в «Творческой категории» допускаются коллективы учащихся и отдельные учащиеся, представляющие свою авторскую разработку в области робототехники (модели роботов и роботизированные устройства, созданные обучающимися самостоятельно или при консультационной поддержке учителей). Задача участников — придумать, создать и запрограммировать устройство (робота), которое сможет помочь в повседневных делах и/или будет источником положительных эмоций.

Детали проекта могут быть использованы от любых конструкторских наборов либо изготовлены самостоятельно. В проекте обязательно должен быть использован хотя бы один микроконтроллер. Ограничений на выбор языка программирования нет. Робот может быть автономным, с дистанционным управлением или без управления.

Количество участников в команде — не более двух человек.

Оценка в «Творческой категории» осуществляется по следующим критериям:

№ п/п	Критерий	Максимальный балл
1	Актуальность	3 балла
2	Новизна	3 балла
3	Сложность конструкции	3 балла
4	Алгоритмическая сложность	3 балла
5	Работоспособность	3 балла
6	Презентация	5 баллов
7	Эстетика	3 балла
8	Качество (фото, описание, видео)	4 балла
9	Особое мнение судьи	3 балла
	Итого максимум:	30 баллов

Итоговым результатом команды является сумма её судейских оценок.

Категория 4 «Юный робототехник».

В состязаниях участвуют учащиеся 7—9 лет.

Соревнования в данной номинации проводятся по индивидуальной системе (один участник — один конструктор).

Участники должны собрать модель по опорной схеме, составить для неё программу по предложенному заданию и объяснить её.

Требования к материалам, оборудованию и программному обеспечению. При себе иметь заряженный ноутбук (зарядное устройство) с необходимым установленным программным обеспечением, конструктор LEGO WeDo (проверить наличие полного комплекта деталей, исправность мотора, коммутатора LEGO, датчика наклона, датчика расстояния).

Схема проведения соревнований:

— Размещение участников на рабочих местах.

— Получение инструкций для выполнения заданий.

— Сборка модели и выполнение заданий.

— Подсчёт итоговых баллов в номинации.

Работа участников над заданием осуществляется следующим образом:

- Объявляется начало старта. Участники приступают к сборке модели по опорной схеме в задании.
- Участник заканчивает сборку модели и приступает к программированию конструкции по заданию.
- Завершается работа над сборкой модели и её программирование. Участник вызывает экспертов. Они фиксируют время, проверяют и оценивают результат, задавая необходимые вопросы по заданию и выставляя набранные баллы в судейском протоколе.

Участникам может быть объявлено дополнительное задание по изменению конструкции и/или алгоритму поведения модели.

По окончании общего времени эксперты суммируют набранные баллы и определяют победителей.

Правила определения победителя. Победителем в номинации объявляется участник, набравший наибольшее количество баллов:

1. сборка робота согласно схеме — до 20 баллов;
2. программирование робота — до 20 баллов;
3. презентация модели — до 10 баллов.

Максимальное количество баллов — 50.

Если участники набирают равное количество баллов, то победитель определяется по лучшему времени.

Пример материалов для проведения межшкольных соревнований по разработке VR/AR приложений

В соревнованиях по разработке приложений VR/AR принимают участие учащиеся образовательных учреждений всех типов и видов и негосударственных образовательных организаций. Возрастные группы участников — 7—18 лет:

I группа — 6—10 лет (допускается индивидуальная и командная работа);

II группа — 11—13 лет (допускается индивидуальная и командная работа);

III группа — 14—18 лет (допускается индивидуальная и командная работа).

Один участник может войти только в одну команду. Регистрация одновременно в двух командах запрещается. Участники команды могут быть разного возраста.

В конкурсе центра цифрового образования детей «IT-Куб» по 3D-моделированию могут принять участие команды школьников и студентов от двух до трёх человек. Участники должны разработать 3D-модель в программе Blender-3D.

Кейс 1. Проект «Культура и город» (3D-моделирование).

Разработать трёхмерную модель любого культурного объекта города, области или села:

— 3D-модель по фотографии памятника культуры или архитектуры;

— 3D-модель любой достопримечательности или интересного места;

— 3D-модель малой архитектурной формы, которая могла бы дополнить общую композицию архитектурного ансамбля застройки

Участники должны разработать 3D-модель в программе Blender-3D.

Кейс 2. Проект «Интерактивный экспонат».

Создать интерактивную модель (экспонат) и приложение к нему с применением технологии смешанной реальности (MR). Интерактивный экспонат представляет собой геометрическую фигуру, например куб с нанесёнными на каждую грань изображениями. Приложение с помощью камеры должно определять объекты на геометрической фигуре и выводить на экран смартфона или планшета дополнительную информацию, связанную с этим объектом (текст, изображение, видео, музыку, анимацию и т. д.).

Объектом на интерактивном экспонате могут выступать: фотографии, иллюстрации, картины, объекты культуры или архитектуры, достопримечательности, интересные места, знаменитые деятели культуры и искусства. Приложение должно запускаться на операционной системе Android. В проекте должна быть реализована одна из предложенных идей.

Кейс 3. «Создание игр в VR/AR».

В игре должна быть реализована минимум одна тема из списка заданий, представленных жюри, и присутствовать система поощрений. Функционал программы должен содержать минимальный набор действий (запуск, информационные блоки, практика, выход из игры). Обязательна разработка минимум одной 3D-модели для игры.

Кейс 4. «Создание приложений AR».

Разработать приложение дополненной реальности для любого школьного предмета, которое будет демонстрировать жизненный процесс из школьной программы (например, химический, физический опыт, круговорот воды в природе и т. д.), под платформу Android или IOS. Обязательно должна быть создана минимум одна 3D-модель, она должна быть анимирована и интерактивна. Желательно наличие звукового сопровождения происходящего процесса.

Кейс 5. Создать интерактивного помощника в дополненной реальности, выполняющего какую-либо функцию, под платформу Android или IOS. Обязательно должна быть создана минимум одна 3D-модель помощника. Помощник должен быть анимирован и интерактивен. Желательно наличие звукового сопровождения. Готовый проект должен быть в формате apk.

Подведение итогов хакатона осуществляется по сумме баллов в рейтинговой системе. Для оценки кейсов могут использоваться следующие критерии.

Критерии	**Оценка (0—5 баллов)**
Работа в команде	
Соответствие содержания работы заявленной теме	
Практичность	
Оригинальность решения, новизна	
Прототип	
Наличие визуальных эффектов	
Умение формирования запускающих приложений	
Дополнительные задачи	
Эстетическая привлекательность	
Сложность программирования	
Убедительность и яркость представления решений, визуальное оформление	
Умение объяснить и защитить свои идеи	
Всего:	

Перечень источников информации

1. Босова Л. Л. Информатика. 8 класс: учебник. / Л. Л. Босова. — М.: БИНОМ. Лаборатория знаний, 2016. — 176 с.
2. Буйначев С. К. Основы программирования на языке Python: учебное пособие . / С. К. Буйначев. — Екатеринбург: Изд-во Урал. ун-та, 2014. — 91 с.
3. Бхаргава А. Грокаем алгоритмы: иллюстрированное пособие для программистов и любопытствующих/ А. Бхаргава. — СПб.: Питер, 2017. — 288 с.
4. Бэрри П. Изучаем программирование на Python. / П. Бэрри. — М., 2017. — 624 с.
5. Винницкий Ю. А. Scratch и Arduino для юных программистов и конструкторов. / Ю. А. Винницкий. — СПб.: БХВ-Петербург, 2018. — 176 с.
6. Голиков Д. В. Scratch для юных программистов./ Д. В. Голиков. — СПб.: БХВ-Петербург, 2017. — 192 с.
7. Гэддис Т. Начинаем программировать на Python. / Д. В. Голиков. — 4-е изд.: пер. с англ. — СПб.: БХВ-Петербург, 2019. — 768 с.
8. Лаборатория юного линуксоида. Введение в Scratch. URL: http://younglinux.info/scratch
9. Луридас П. Алгоритмы для начинающих: теория и практика для разработчика. / П. Луридас. — М.: Эксмо, 2018. — 608 с.
10. Лутц М. Изучаем Python / Лутц М. — 3-е изд./ Пер. с англ. — СПб.: Символ Плюс, 2009. — 848 с.
11. Маржи М. Scratch для детей. Самоучитель по программированию/ — пер. с англ. М. Гескиной и С. Таскаевой — М.: Манн, Иванов и Фербер, 2017. — 288 с.
12. Мюллер Дж. Python для чайников. / Дж. Мюллер. — СПб.: Диалектика, 2019. — 416 с.
13. Пашковская Ю. В. Творческие задания в среде Scratch: рабочая тетрадь для 5—6 классов /Ю. В. Пашковская. — М., 2018. — 195 с.
14. Первин Ю. А. Методика раннего обучения информатике/ Ю. А. Первин. — М.: БИНОМ. Лаборатория знаний, 2008. — 228 с.
15. Поляков К. Ю. Информатика. 7 класс (в 2 частях): учебник. Ч. 1 / К. Ю. Поляков, Е. А. Ерёмин. — М.: БИНОМ. Лаборатория знаний, 2019. — 160 с.
16. Рафгарден Т. Совершенный алгоритм. Жадные алгоритмы и динамическое программирование. / Т. Рафгарден. — СПб.: Питер, 2020. — 256 с.
17. Рейтц К. Автостопом по Python. / К. Рейтц, Т. Шлюссер. — СПб.: Питер, 2017. — 336 с.
18. Рындак В. Г. Проектная деятельность школьника в среде программирования Scratch: учебно-методическое пособие. / В. Г. Рындак В, В. О. Дженжер, Л. В. Денисова. — Оренбург: Оренб. гос. ин-т. менеджмента, 2009. — 116 с.
19. Свейгарт Эл. Программирование для детей. Делай игры и учи язык Scratch! / Эл. Свейгарт. — М.: Эксмо, 2017. — 304 с.
20. Семакин И. Г. Информатика и ИКТ: учебник для 9 класса/ И. Г. Семакин, Л. А. Залогова и др. — М: БИНОМ. Лаборатория знаний, 2014. — 171 с.
21. Торгашева Ю. В. Первая книга юного программиста. Учимся писать программы на Scratch/ Ю. В. Торгашева. — СПб.: Питер, 2016. — 128 с.

22. Уфимцева П. Е. Обучение программированию младших школьников в системе дополнительного образования с использованием среды разработки Scratch/ П. Е. Уфимцева, И. В. Рожина // Наука и перспективы. — 2018. — № 1. — С. 29—35.
23. Фёдоров Д. Ю. Программирование на языке высокого уровня Python: учебное пособие для прикладного бакалавриата / Д. Ю. Федоров. — М.: Издательство Юрайт, 2019. — 161 с.
24. https://scratch.mit.edu/ Сообщество Scratch.
25. Практический Python 3 для начинающих URL: https://pythonworld.ru/samouchitel-python.
26. Учебник по языку программирования Python (хабраиндекс) URL: https://habr.com/ru/post/61905/
27. Python / Учебник Python 3.1. URL: https://ru.wikibooks.org/wiki/ Python/%D0%A3%D1%87%D0%B5%D0%B1%D0%BD%D0%B8%D0%BA_Python_3.1.
28. Python для начинающих 2021 — уроки, задачи и тесты. URL: https://pythonru.com/uroki/python-dlja-nachinajushhih.
29. Сайт itProger [электронный ресурс] // URL: https://itproger.com/course/c-programming/2 (дата обращения: 15.04.2021).
30. Портал обучения VEX Академия [Электронный ресурс] // URL: http://vexacademy.ru/ (дата обращения: 15.04.2021).

Григорьев Сергей Георгиевич
Востроknутов Игорь Евгеньевич
Родионов Михаил Алексеевич
Исайкин Олег Анатольевич
Трухманов Вячеслав Борисович

Реализация образовательных программ предметной области «Математика и информатика» с использованием оборудования центра цифрового образования детей «IT-Куб»

Методическое пособие

Под редакцией С. Г. Григорьева

Центр естественно-научного и математического образования
Руководитель Центра *З.Г. Гапонюк*
Ответственный за выпуск *М.В. Кузнецова*
Редактор *М.В. Кузнецова*
Художественное оформление *Т.В. Глушкова*
Компьютерная вёрстка и техническое редактирование *Н.А. Разворотнева*
Корректор *Н.А. Смирнова*