**Муниципальное бюджетное общеобразовательное учреждение «Гимназия №1»**

**с. Красногвардейского, МО «Красногвардейского района»,**

**Республики Адыгея, с. Красногвардейского, ул. Чапаева, 87 а**

**Тел: 5-24-75, 5-24-42, адрес электронной почты: kmoug1@mail.ru**

Рабочая программа

внеурочной деятельности с использованием

оборудования центра

«Точка роста»

« Робототехника и информатика»

2022-2023 учебный год

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

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».

Методические рекомендации по созданию и функционированию в общеобразовательных организациях, расположенных в сельской местности и малых городах, центров образования естественно-научной и технологической направленностей («Точка роста») (Утверждены распоряжением Министерства просвещения Российской Федерации от 12 января 2021 г. № Р-6)

Курс, рассчитанный на 34 академических часа. Включает теоретические и практические занятия. Содержание программы «Удивительный микромир» связано с предметами естественнонаучного цикла.

На курс «Удивительный микромир» отводится по 1 час в неделю в 5 классе. Курс входит в раздел учебного плана «Внеурочной деятельности».

Программа курса предназначена для обучающихся в основной школе, интересующихся исследовательской деятельностью, и направлена на формирование у учащихся умения поставить цель и организовать её достижение, а также креативных качеств – гибкость ума, терпимость к противоречиям, критичность, наличие своего мнения, коммуникативных качеств.

Актуальность программы курса обусловлена тем, что знания и умения, необходимые для организации учебно-исследовательской деятельности, в будущем станут основой для реализации учебно-исследовательских проектов в среднем и старшем звене школы. Программа курса позволяет реализовать актуальные в настоящее время компетентностный, личностно ориентированный, деятельностный подходы.

**Условия реализации программы**

* Возраст детей, участвующих в реализации данной программы, 11-15 лет.
* Продолжительность образовательного процесса - 1 год.
* Количество часов – 34 учебных часа в неделю

**Формы организации деятельности учащихся на занятиях**

* Групповая
* Индивидуальная

**Формы и методы, используемые в работе по программе**

**Словесно-иллюстративные методы:** рассказ, беседа, дискуссия, работа с биологической литературой.

**Репродуктивные методы:**воспроизведение полученных знаний во время выступлений.

**Частично-поисковые методы** (при систематизации коллекционного материала).

**Исследовательские методы**(при работе с микроскопом).

**Наглядность:** просмотр видео-, кино-, диа-, слайдфильмов, компьютерных презентаций, биологических коллекций, плакатов, моделей и макетов.

**Ожидаемый результат:**

* положительная динамика социальной и творческой активности обучаемых, подтверждаемая результатами их участия в конкурсах различного уровня, фестивалях, смотрах, соревнованиях.
* повышение коммуникативности;
* появление и поддержание мотивации к углубленному изучению биологии;
* умение пользоваться современными источниками информации и давать аргументированную оценку информации по биологическим вопросам; работать с научной и учебной литературой;
* сформировавшиеся биологические знания, умения и навыки, одновременно приобретенные навыки организации внеклассной работы: проведения викторин, бесед, классных часов с учащимися начальной школы.

Система занятий сориентирована не столько на передачу «готовых знаний», сколько на формирование активной личности, мотивированной к самообразованию, обладающей начальными навыками самостоятельного поиска, отбора, анализа и использования информации.

Важнейшим приоритетом общего образования является формирование обще учебных умений и навыков, которые предопределяют успешность всего последующего обучения ребёнка.

Развитие личностных качеств и способностей школьников опирается на приобретение ими опыта разнообразной деятельности: учебно-познавательной, практической, социальной.

Курс «Удивительный микромир» носит развивающий характер. Целью данного спецкурса является формирование поисково-исследовательских и коммуникативных умений школьников.

Занятия курса разделены на теоретические и практические. Причём деятельность может носить как групповой, так и индивидуальный характер.

**Cодержание программы.**

# Содержание курса

## Программирование в Scratch (34 часа)

Знакомство со средой программирования Scratch. Создание и сохранение документа. Понятия спрайта, сцены, скрипта. Библиотека персонажей. Исполнитель Scratch. Основные ин- струменты встроенного графического редактора программной среды SCRATCH. Линейный алгоритм. Создание блок-схемы. Рисование линий исполнителем Scratch. Конечный и беско- нечный циклы. Цикл в цикле. Анимация исполнителя Scratch на основе готовых костюмов. Дублирование исполнителей. Алгоритмы с ветвлением. Цикл с условием. Перемещение ис- полнителей между слоями. Программирование клавиш. Управ- ление событиями. Координатная плоскость. Создание списков. Использование подпрограмм. Отладка программ с ошибками.

## Работа с текстовым процессором LibreOffice.org Writer (34 часа)

Загрузка и установка LibreOffice. Интерфейс редактора. Стан- дартные действия. Форматирование документа: шрифты, сти- ли, размер шрифта. Работа с цветом. Сложное форматирование. Использование списков. Колонтитулы. Изображения в тексто- вых документах. Графика в текстовых документах. Таблицы в документах. Работа с Google-docs.

# Поурочное планирование

## Модуль 1. Программирование в Scratch

**Урок 1. Знакомство со средой программирования Scratch** Свободное программное обеспечение. Авторы программной среды Scratch. Параметры для скачивания и установки про- граммной среды на домашний компьютер.

Основные элементы пользовательского интерфейса про- граммной среды Scratch. Внешний вид рабочего окна. Блоч- ная структура систематизации информации. Функциональные блоки. Блоки команд, состояний, программ, запуска, действий и исполнителей. Установка русского языка для Scratch.

Создание и сохранение документа. Понятия спрайта, сцены, скрипта. Очистка экрана.

Библиотека персонажей. Сцена и разнообразие сцен, исхо- дя из библиотеки данных. Систематизация данных библиотек персонажей и сцен. Иерархия в организации хранения костю- мов персонажа и фонов для сцен. Импорт костюма, импорт фона.

## Кейс 1. Научи кота бегать и мяукать.

**Урок 2. Исполнитель Scratch, цвет и размер пера.**

Понятие ИСПОЛНИТЕЛЯ.

Команды управления пером: «Опустить перо», «Поднять перо», «Очистить», «Установить цвет пера», «Установить раз- мер пера.

## Кейс 2. Рисуем разноцветные лужи и облака для прогулки кота. Урок 3. Основные инструменты встроенного графического ре- дактора программной среды SCRATCH.

Инструменты растрового графического редактора — кисточка, ластик, заливка, квадрат, круг, линия.

Копирование, поворот, горизонтальное отражение, вертикаль- ное отражение во встроенном редакторе программной среды Scratch.

Командный блок внешность (фиолетовый) — команды началь- ной установки эффекта цвет «Установить эффект цвет в зна- чение 0 и команда начальной установки размера «Установить размер 100%».

Команды: «Изменить цвет эффект на \_», «Изменить размер на

\_»

Командный блок управления пером (зелёный) — команда «Пе- чать» для копирования графического изображения исполните-

ля в нужном месте экрана.

Эффекты, которые могут быть применены к графическим изо- бражениям действующего исполнителя.

В графическом редакторе можно создавать сложные графиче- ские изображения, почти не прикладывая собственных усилий, а лишь правильно применяя встроенные возможности про- граммной среды.

## Кейс 3. Свободное рисование

Придумай, чем можно дополнить проект с прошлого урока (Нарисовать дерево? Воздушный шар? Дом?) и реализуй это.

## Урок № 4. Алгоритм. Линейный алгоритм. Создание блок-схе- мы. Основные графические примитивы векторного редактора LibreOffice.Draw.

Создание собственных изображений в других программах (на- пример, LibreOfficeDraw) и импортирование их в программную среду Scratch.

Знакомство с основными графическими примитивами вектор- ного редактора LibreOfficeDraw. Возможность создания геоме- трических фигур без внутренней заливки, но с текстовым бло- ком внутри. Стрелки, их направление.

Алгоритм, блок-схема как способ записи. Кейс 4. Запиши мой алгоритм!

Ребята разбиваются на пары, в паре формулируют друг другу какую-то повседневную задачу, для решения которой нужен линейный алгоритм, придумывают этот алгоритм и рисуют блок-схему для задачи, заданной им напарником.

## Урок № 5. Линейный алгоритм. Рисование линий исполните- лем Scratch.

Решение поставленной задачи в виде последовательного вы- полнения команд.

Создание блок-схемы линейного алгоритма средствами редак- тора векторной графики.

Последовательное выполнение команд.

Изменение параметров пера.

## Кейс 5. Что бывает полосатое?

Придумайте линейный алгоритм для создания полосатого объ- екта (заранее придумайте, что это будет – шарф, зебра, забор?) Создайте блок-схему этого алгоритма и реализуйте алгоритм в среде Scratch.

## Урок № 6. Линейный алгоритм. Исполнитель Scratch рисует квадраты и прямоугольники линейно.

Создание алгоритма для рисования исполнителем квадрата путем последовательного выполнения команд.

Рисование линейного алгоритма, состоящего из двух колонок блоков команд.

Использование векторного редактора офисного пакета LibreOffice в качестве инструмента для создания блок-схем.

Выбор нужного значения из предлагаемого списка вариантов. Отладка программы для получения верного результата.

Команда «повернуть в направление».

Пошаговое выполнение программы для её отладки. Центр костюма исполнителя Scratch.

**Урок №7. Конечный цикл. Scratch рисует квадраты, линии.** Сохранение готовых программ для дальнейшего использова- ния.

Рисование блок-схемы циклического алгоритма. Использование векторного редактора офисного пакета LibreOffice в качестве инструмента для создания блок-схем; Использование команд поворота на прямой угол (90º) по часо- вой и против часовой стрелки;

Использование циклического алгоритма для рисования ис- полнителем квадрата;

Оптимизация линейного алгоритма за счёт использования циклической конструкции в программной среде Scratch; Команда открыть... из пункта меню File;

Команда сохранить как... из пункта меню File;

Циклический алгоритм;

Блок-схема циклического алгоритма;

Команды: «повернуться на 90º по часовой стрелке»,

«повернуться на 90º против часовой стрелки», «повторить \_».

**Упражнение.** Сравнение двух алгоритмов, рисующих квадрат. Идти 100 шагов Повторить 4

Повернуться на 90º Идти 100 шагов Идти 100 шагов Повернуться на 90º Повернуться на 90º

Идти 100 шагов Повернуться на 90º Идти 100 шагов Повернуться на 90º

Рекомендуется объяснить, что команды, обеспечивающие по- вторяющееся выполнение одной или нескольких команд на- зываются циклическими, показать, что использование коман- ды цикла всегда сокращает количество команд в программе, а значит делает её более красивой. В более короткой программе легче обнаружить ошибку или неточность.

## Урок №8. Конечный цикл. Scratch рисует несколько линий и фигур. Копирование фрагментов программы.

Использование операции цикла для решения учебных задач. Применение поворота на прямой угол (90º) при создании гео- метрических фигур и перемещении исполнителя.

Оптимизация линейного алгоритма за счёт использования циклической конструкции в программной среде Scratch.

Тело цикла.

Конечный и бесконечный циклы. Блок-схема бесконечного цикла. Имя спрайта и костюма.

Изменение костюма исполнителя. Копирование фрагмента программы.

Команды: «Следующий костюм», «Перейти к костюму».

Рекомендуется объяснить, что циклические конструкции могут применяться не только для рисования фигур. Например, в ци- кле можно менять внешность исполнителя.

## Кейс 6. Создай картинку из квадратов.

Предложите ребятам нарисовать свою картинку из квадратов, может быть, это будет лестница, может быть, ковер, может быть, что-то еще.

**Урок №9. Циклический алгоритм. Цикл в цикле.** Использование операции цикла в цикле для решения учебных задач.

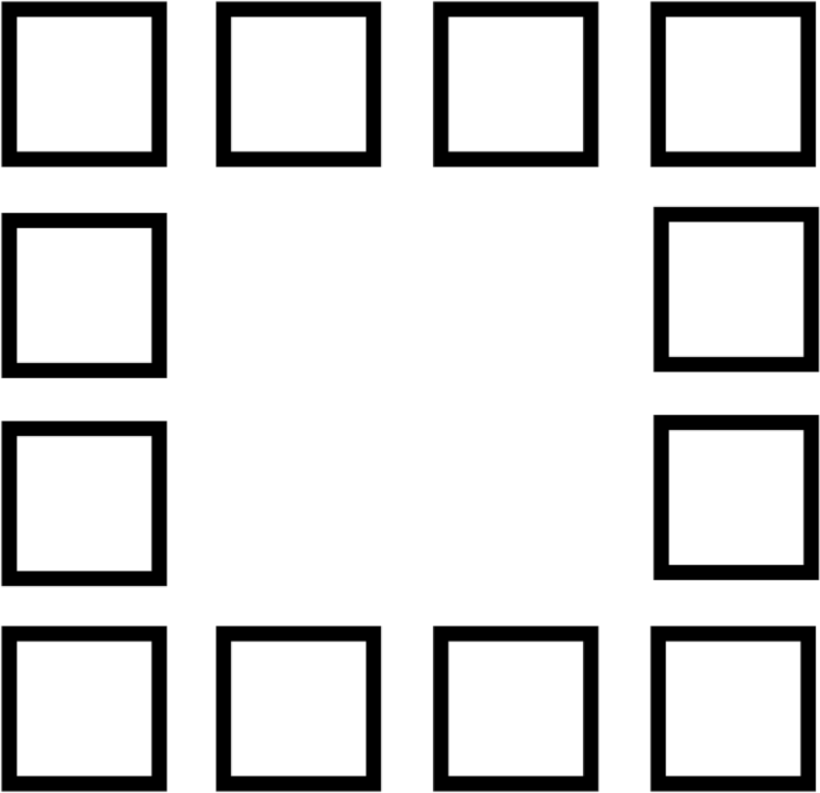
Использование поворота на прямой угол (90º) по часовой и против часовой стрелки.

Оптимизация алгоритма за счёт использования конструкции

«цикл в цикле».

Блок-схема конструкции цикл в цикле.

Например, расписание уроков составляется на неделю, в ме- сяце в среднем четыре недели. Если расписание не меняется в течение учебного года, то получаем цикл из девяти месяцев по четыре недели с одинаковым расписанием.

**Упражнение.** Использование циклов «повторить \_» Нарисуйте квадраты, как показано на рисунке, используя кон- струкцию цикл в цикле.

## Урок №10. Цикл в цикле. Повторение пунктирной линии с по- воротом. Блок-схема цикла.

Использование конструкции «цикл в цикле».

Создание и реализация алгоритма рисования квадрата нес- плошными линиями;

Оптимизация алгоритма за счёт использования конструкции

«цикл в цикле» в программной среде Scratch;

Использование операции копирования внешности исполни- теля путём копирования костюма.

Несплошные линии.

Алгоритм рисования несплошных линий с использованием циклических конструкций.

Рисование квадрата несплошными линиями, используя кон- струкцию «цикл в цикле».

При перемещении исполнителя можно рисовать линии отлич- ные от сплошной. Эти возможности определяются использо- ванием команд из блока ПЕРО.

## Кейс 7. Мой необычный дом.

Предложите ребятам нарисовать дом и что-то рядом с ним, используя линейные и циклические алгоритмы, а также смену костюмов Исполнителя. Может быть, над домом будут звезды разного размера, а около дома будет пунктирная разноцветная дорога, или деревья разного размера.

## Урок №11. Бесконечный цикл. Анимация исполнителя Scratch на основе готовых костюмов.

Бесконечный цикл.

Анимация исполнителя с помощью смены костюмов. Эффект «призрак».

Изменение размера исполнителя.

Интерактивное взаимодействие с исполнителем с помощью клавиатуры.

Команды «спросить» и «думать».

## Урок №12. Сцена как исполнитель. Создаем модель таймера.

Анимация сцены (фона).

Синхронная анимация сцены и исполнителя.

## Урок №13. Одинаковые программы для нескольких исполни- телей.

Анимация с помощью вращения. Дублирование исполнителей.

При копировании исполнителей копируются и их скрипты. Использование одинаковых программ, но разных костюмов у исполнителей.

## Урок №14. Параллельное выполнение действий несколькими исполнителями

Дублирование исполнителей. Синхронное выполнение скриптов.

При параллельном выполнении скриптов результат работы всех исполнителей получается гораздо быстрее, чем в тех за- дачах, когда работал один исполнитель.

## Урок №15. Разбиение программы на части для параллельного выполнения исполнителями. Таймер.

Планирование действий разных исполнителей во времени для решения общей задачи.

Сенсор «таймер» Сброс «таймера» Команда «ждать до …»

## Кейс 8. Мини-проект «Смена времени суток».

Создайте скрипт, в котором будет четыре сцены: «утро»,

«день», «вечер», «ночь». Используйте «таймер» для своевре- менного изменения сцен, своевременного запуска скриптов разных исполнителей.

## Урок №16. Два исполнителя со своими программами. Ми- ни-проект «Часы с кукушкой».

Координаты в Scratch.

Команда «идти в x: \_ y:\_» (переместиться в точку с координата- ми (x,y))

Команда «плыть \_ секунд в точку x: \_ y:\_» (плавно двигаться в точку с координатами (x,y))

Команды «спрятаться», «показаться».

## Кейс 9. Мини-проект «Часы с кукушкой».

Создайте сцену в виде часов, и три исполнителя: часовую стрелку, минутную стрелку и кукушку. Стрелки должны вра- щаться на циферблате. При наступлении каждого часа (т.е. в \_ часов : 00 минут) должна появляться кукушка.

## Урок №17. Алгоритмы с ветвлением. Условие ЕСЛИ.

Еще из сказок мы знаем, что на камне было написано: «Нале- во пойдешь – коня потеряешь, направо пойдешь – себя не най- дешь, прямо пойдешь – в сказку попадешь»

Блок-схема алгоритма с ветвлением.

Полная форма ветвления. Команда «если \_ или» Неполная форма ветвления. Команда «если \_»

Сенсор «касается» (касания края, или другого спрайта, или ука- зателя мыши)

## Урок №18. Цикл с условием. Мини-проект «Шарики в лаби- ринте»

Сенсор «касается цвета» Программируем отскок шариков от стен

## Кейс 10. Мини-проект «Шарики в лабиринте»

Создайте сцену в виде биллиардного стола, на котором три ша- рика будут двигаться с разными скоростями и отталкиваться от бортиков.

## Урок №19. Цикл с условием. Исполнитель определяет цвет. Сенсор «касается цвета»

Программируем поведение исполнителя в зависимости от цвета фона.

## Урок № 20. Оператор случайных чисел.

Команда «Выдать случайное число от \_ до \_». Случайные пере- мещения исполнителя в координатной плоскости.

Случайное количество шагов. Случайные координаты.

Поворот на случайный угол.

## Урок №21. Перемещение исполнителей между слоями.

Команда «Перейти в верхний слой» Команда «Перейти назад на \_ слоев» Эффект «Призрак»

## Урок №22. Действия исполнителей в разных слоях. Кейс 10. Мини-проект «Дорога».

Создайте сцену, на которой изображена дорога с двумя поло-

сами, по обочинам дороги расположите кусты и деревья. По дороге должны двигаться автомобили. Правильно расположи- те автомобили, деревья и кусты по слоям.

## Урок №23 Взаимодействие исполнителей.

Исполнители касаются друг друга. Команда «Касается \_»

Поведение исполнителей при столкновении.

## Урок №24. Последовательное выполнение команд исполни- телями.

Команда «Передать» Команда «Когда я получу»

Связи между программами разных исполнителей.

## Урок №25. Программирование клавиш. Мини-проект «Лаби- ринт»

Взаимодействие пользователя с программой. Клавиши управления перемещением исполнителя Команда «Изменить значение x на \_»

Команда «Изменить значение y на \_»

## Кейс 10. Игра «Лабиринт»

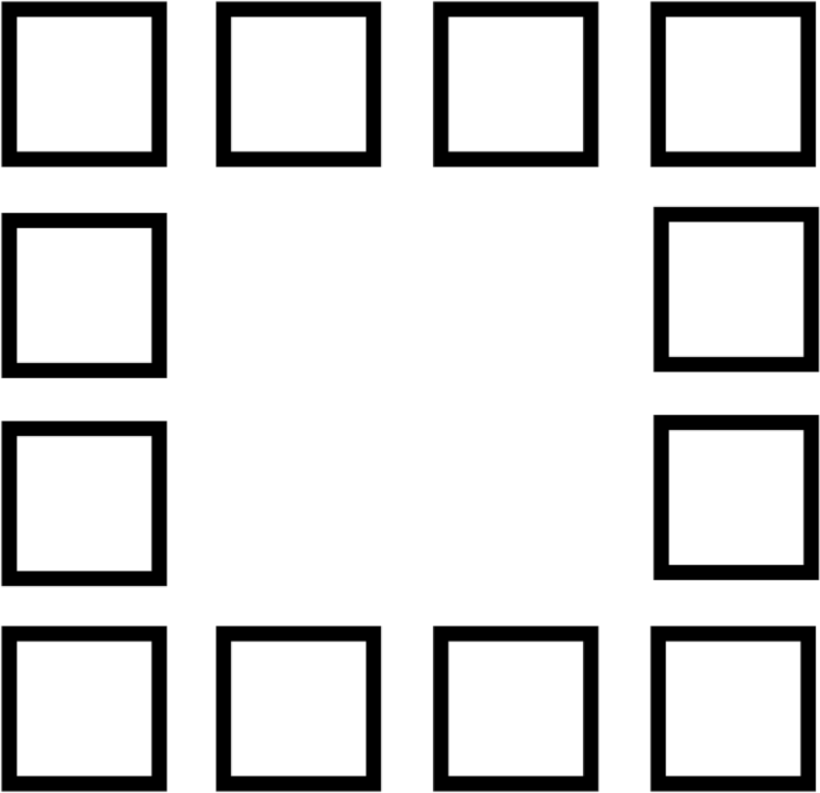
Создайте сцену в виде лабиринта. С помощью клавиш со стрел- ками управляйте движением маленькой мышки. Мышка долж- на добраться до выхода, не задевая стенок лабиринта. В случае касания стенок мышка возвращается в исходную точку.

## Урок №26. Управление событиями.

Передача сообщений между исполнителями и фоном. Запуск программ после получения сообщения.

**Урок №27. Координатная плоскость. Геометрические фигуры.** Рисование геометрических фигур на координатной плоскости. Последовательное выполнение команд одним исполнителем. Параллельное выполнение команд несколькими исполнителя- ми.

**Кейс 11. Исследование времени выполнения программ.** Создайте одного исполнителя, рисующего квадраты, как пока- зано на рисунке. Затем создайте четыре исполнителя, которые будут параллельно рисовать по три квадрата. Сравните время работы в этих двух случаях.



## Урок №28. Координатная плоскость. Переменные.

Блок «Переменные»

Рисование геометрических фигур на координатной плоскости с использованием переменных.

Построение перпендикуляров к координатным осям.

## Урок №29. Создание списков.

Название списка. Элементы списка. Длина списка.

Команда «Создать список». Выбор элемента списка.

## Кейс 11. Мини-проект «Викторина».

Создайте проект «Викторина» в котором будет задано 10 во- просов. Правильный ответ надо выбрать из списка. В случае правильного ответа очки увеличиваются на единицу. В конце выставляется оценка: «отлично» за 9 или 10, «хорошо» за 7 или 8, «удовлетворительно» за 5 или 6 правильных ответов.

## Урок №30. Использование подпрограмм.

Как сделать программу структурированной и более понятной. Команда «Передать \_ и ждать»

Команда «Играть звук \_» Команда «Ноту \_ играть \_ тактов»

## Урок №31. Сообщество Scratch.

Scratch 2.0. Регистрация на сайте. Личный кабинет.

Публикация проектов.

**Урок №32. Отладка программ с ошибками.** <http://scratch.mit.edu/projects/10437040> <http://scratch.mit.edu/projects/10437249> <http://scratch.mit.edu/projects/10437366>

<http://scratch.mit.edu/projects/10437439> <http://scratch.mit.edu/projects/10437476>

## Уроки №33 - 34. Кейс 12. Итоговый проект.

Учащиеся самостоятельно или в парах выполняют индивиду- альный проект, согласованный с учителем. Как итог, можно провести конкурс проектов.

## Обучающийся научится:

* составлять сценарии проектов среды Scratch;
* составлять алгоритмы, определять последовательность вы- полнения команд;
* создавать и редактировать рисунки в графическом редакто- ре;
* использовать обширную библиотеку готовых сцен и испол- нителей;
* изменять размер, костюм, прозрачность исполнителя;
* создавать линейные алгоритмы для исполнителя;
* создавать циклические алгоритмы;
* создавать ветвящиеся алгоритмы;
* управлять одновременной работой нескольких исполните- лей;
* передавать сообщения между исполнителями;
* внедрять звуковые эффекты в алгоритмы исполнителей;
* создавать алгоритмы, которые будут выполняться одновре- менно (параллельно) несколькими исполнителями;
* тестировать и оптимизировать алгоритмы исполнителей.

# Робототехника

## Планируемые результаты освоения учебного предмета с описанием универсальных учебных действий, достигаемых обучающимися

**Личностные:**

— формирование профессионального самоопределения, ознакомление с миром профессий, связанных с информационными и коммуникационными технологиями; — формирование умения работать в команде;

— развитие внимательности, настойчивости, целеустремлённости, умения преодолевать трудности;

— формирование навыков анализа и самоанализа .

**Предметные:**

— формирование понятий о различных компонентах робота и платформы VEXcode VR (программные блоки по разделам, исполнительные устройства, кнопки управления и т . д .);

— формирование основных приёмов составления программ на платформе VEXcode VR;

— формирование алгоритмического и логического стилей мышления;

— формирование понятий об основных конструкциях программирования: условный оператор if/else, цикл while, понятие шага цикла .

**Метапредметные:**

— освоение способов решения проблем творческого характера в жизненных ситуациях;

— формирование умений ставить цель — создание творческой работы, планирование достижения этой цели, создание вспомогательных эскизов в процессе работы;

— использование средств информационных и коммуникационных технологий для решения коммуникативных, познавательных и творческих задач;

— формирование информационной культуры;

— формирование умения аргументировать свою точку зрения на выбор способов решения поставленной задачи .

## Формы контроля

Во время проведения курса предполагается текущий, промежуточный и итоговый контроль .

Текущий контроль проводится на каждом занятии с целью выявления правильности применения теоретических знаний на практике . Текущий контроль может быть реализован посредством следующих форм: наблюдение, индивидуальные беседы, тестирование, творческие работы, проблемные (ситуативные) задачи, практические работы, контрольные вопросы и т . д .

*Примеры ситуативных задач по модулю 1*

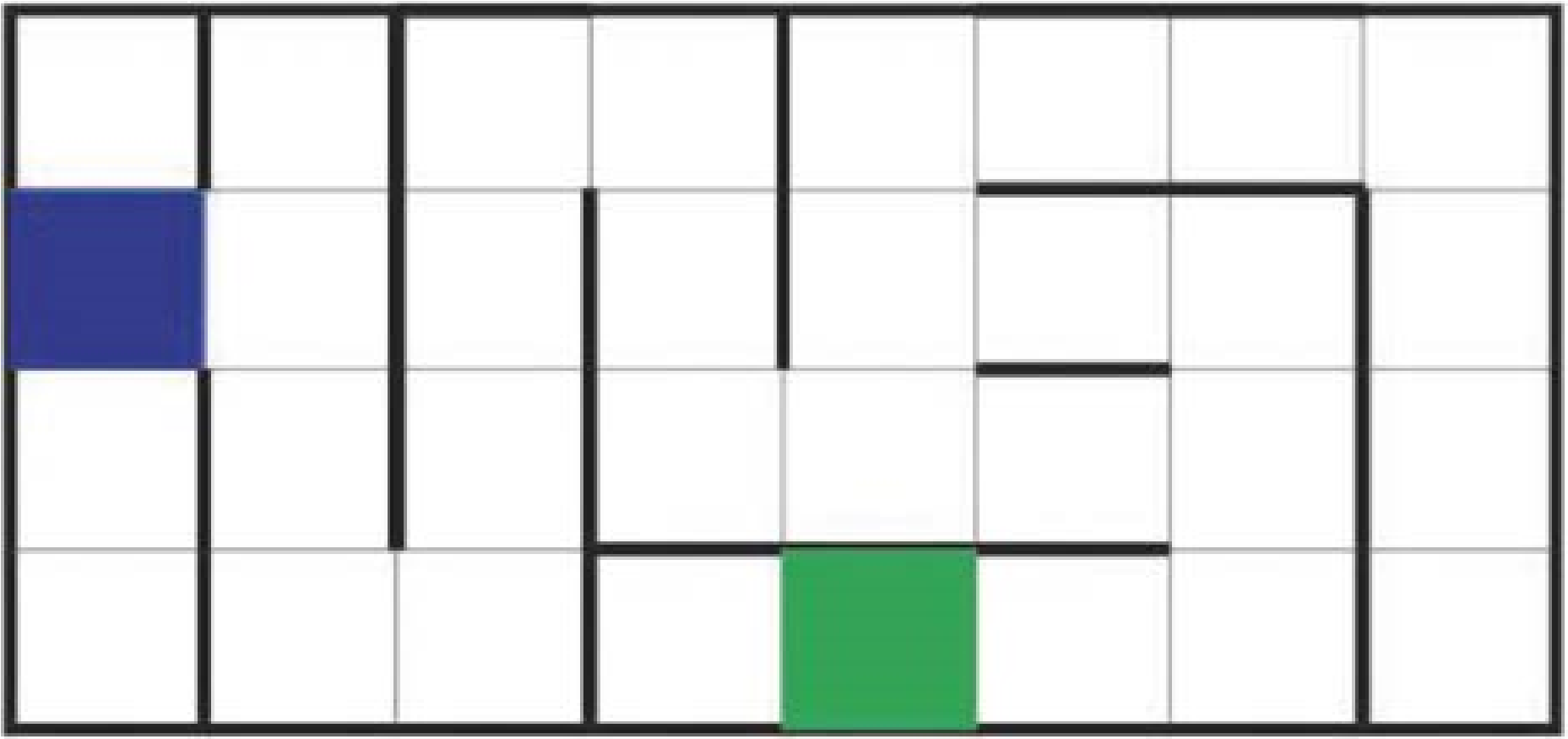
Задача 1 . Петя запустил робота, который двигается по следующей программе:

1. стартует с точки*А* и едет на запад со скоростью *V* = 3 м/мин в течение 60 с;
2. поворачивает на юг и столько же времени движется с удвоенной скоростью 2*V*;
3. поворачивает на восток и едет с утроенной скоростью 3*V* такое же время, что на первых двух участках вместе взятых;
4. поворачивает на север и, проехав 6 м за 1,5 мин, добирается до финиша, расположенного в точке *В* .

Вопросы:

1. . Какова длина первого участка пути? Ответ дайте в метрах с точностью до целых .
2. . С какой постоянной скоростью на всём пути должен двигаться робот, чтобы проехать его за то же время? Ответ укажите в метрах в секунду с точностью до сотых .
3. . Найдите расстояние между точкой старта А и точкой финиша В робота . Ответ дайте в метрах с точностью до целых .

Задача 2 . Три колёсных робота А1, А2 и А3 одинаковой конструкции должны по очереди пройти лабиринт, двигаясь от входа (синий квадрат) к выходу (зелёный квадрат) . Робот А1 содержит в памяти карту лабиринта, на которой отмечены синий и зелёный квадраты и указаны все стенки . Робот А2 не знает карты лабиринта и запрограммирован обходить его по правилу правой руки . Робот А3 не знает карты лабиринта и запрограммирован обходить его по правилу левой руки . Какой из роботов пройдёт лабиринт медленнее всего?



***Рис. 2.*** Вид лабиринта

Промежуточный контроль проводится в рамках промежуточной аттестации после изучения нескольких модулей в виде подготовки и защиты творческих (проектных) работ, соревнований и состязаний .

*Пример соревнования «Динамический лабиринт»*

Цель: запрограммировать робота на решение лабиринта (прибытие на красный квадрат), в кратчайшие сроки .

Команды состоят только из одного участника .

Правила и подсчёт очков:

1. задача состоит в том, чтобы пройти лабиринт в кратчайшие сроки . Лабиринт считается пройдённым, когда все колёса робота касаются красного квадрата;
2. максимальное время — 180 секунд . Если робот не завершил лабиринт за этот промежуток времени, время будет считаться как 200 секунд;
3. победителем становится команда с лучшим средним временем прохождения лабиринта из двух попыток . Если есть ничья, то в качестве тай-брейка используется лучшее время команды .

*Тест по теме «Робот. Базовые понятия»*

1. . В каком году появилось слово РОБОТ?

*А) 1920*

Б) 1925

В) 1930

Г) 1935

1. . Слово «Робот» — произошёл от чешского слова, которое означает…

А) RoboTech

Б) Robot

В) RobotLand

*Г) Robota*

1. . Кто придумал три закона робототехники?

А) Валли

*Б) А. Азимов*

В) Г . Галилей

Г) К . Чапек

1. . С 1968 г . «столицей роботов» считается

А) Китай

Б) Россия

*В) Япония*

Г) США

1. . Как называется разработанный AldebaranRobotics человекоподобный робот, поступивший в массовую продажу?

А) T-800

Б) Atlas

*В) Pepper*

Г) ASIMO

При проведении итоговой аттестации в форме проектной работы задание ориентировано на индивидуальное исполнение . Защита итогового проекта проходит в форме представления обучающимся технического задания на проект, работающего кода, ответов на вопросы преподавателя, обсуждения с учащимися достоинств и недостатков проекта .

*Модуль 1. «Платформа VEXcode VR»*

В результате изучения данного модуля учащиеся должны: знать названия различных компонентов робота и платформы:

— контроллёр (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 .

Урок 1 . Робот . Базовые понятия

Урок 2 . Знакомство со средой VEXcode VR

Урок 3 . Исполнительные механизмы конструкторов VEX

Лабораторная работа 1 . Создание простейших программ (скриптов)

*Модуль 2. «Программирование робота на платформе»*

В результате изучения данного модуля учащиеся должны 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 .

Урок 4 . Программируемый контроллёр

Урок 5 . Основные блоки

Лабораторные работы 2-3 . Программирование блоков управления роботом

*Модуль 3. «Датчики и обратная связь»*

В результате изучения данного модуля учащиеся должны знать:

— принципы работы датчиков; — блоки управления датчиками; — возможности датчиков .

— уметь:

— использовать циклы и ветвления для реализации системы принятия решений; — решать задачу «Лабиринт» .

Урок 6 . Датчик местоположения, направление движения

Урок 7 . Датчики цвета Урок 8 . Датчик расстояния

Урок 9 . Управление магнитом

Лабораторная работа 4 . Скрипты с датчиком местоположения

Лабораторная работа 5 . Игровое поле «Дисковый лабиринт»

Лабораторные работы 6-8 . Простой лабиринт . Динамический лабиринт Лабораторная работа 9 . Игровое поле «Перемещение фишек»

*Модуль 4. «Реализация алгоритмов движения робота»*

В результате изучения данного модуля учащиеся должны 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 .

Урок 10 . Блок команд «Управление»

Урок 11 . Проекты «Разрушение замка» и «Динамическое разрушение замка»

Урок 12 . Проект «Детектор линии»

Лабораторная работа 10 . Ветвления на базе платформы VEXcode VR

Лабораторная работа 11 . Циклы на базе платформы VEXcode VR

Лабораторная работа 12 . Блок «Всегда», блок «Прерывания» и блок «Ждать пока»

Лабораторные работы 13—15 . Проект по уборке территории

Лабораторная работа 16 . Поиск и подсчёт линий

*Модуль 5. «Творческий проект»*

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

## Планы учебных занятий

#### Урок 2 . Знакомство со средой VEXcodeVR

**Тип урока:** комбинированный .

**Цель урока:** знакомство с интерфейсом VEXcode VR .

**Планируемые результаты**

**Предметные:** получение навыков работы с панелью управления, получение навыков по созданию первых программ в среде VEXcode VR .

**Метапредметные:** умение самостоятельно определять цели своего обучения, ставить и формулировать для себя новые задачи в учёбе и познавательной деятельности; умениеконтролировать и корректировать учебную деятельность, способность ставить и формулировать для себя цели действий, прогнозировать результаты, анализировать их .

**Личностные:** готовность и способность обучающихся к саморазвитию и личностному самоопределению; сформированность их мотивации к обучению и целенаправленной познавательной деятельности .

**Время реализации:** 1 академический час .

**Оборудование и материалы:** компьютеры с выходом в сеть Интернет, проектор, интерактивная доска .**Ход урока**

1 . Этап постановки цели и задач урока, мотивации к учебной деятельности — 15 минут .

Деятельность учителя — подача нового материала с демонстрацией платформы [http://vr .vex .com](http://vr.vex.com/) .

VEXcode VR облегчает изучение информатики и робототехники, позволяя обучающимся кодировать виртуального робота, находясь в любом месте, программируя в среде блочного кодирования . VEXcode VR основан на VEXcode, том же интерфейсе программирования, который используется для робототехнических платформ VEX 123, GO, IQ и V5 .

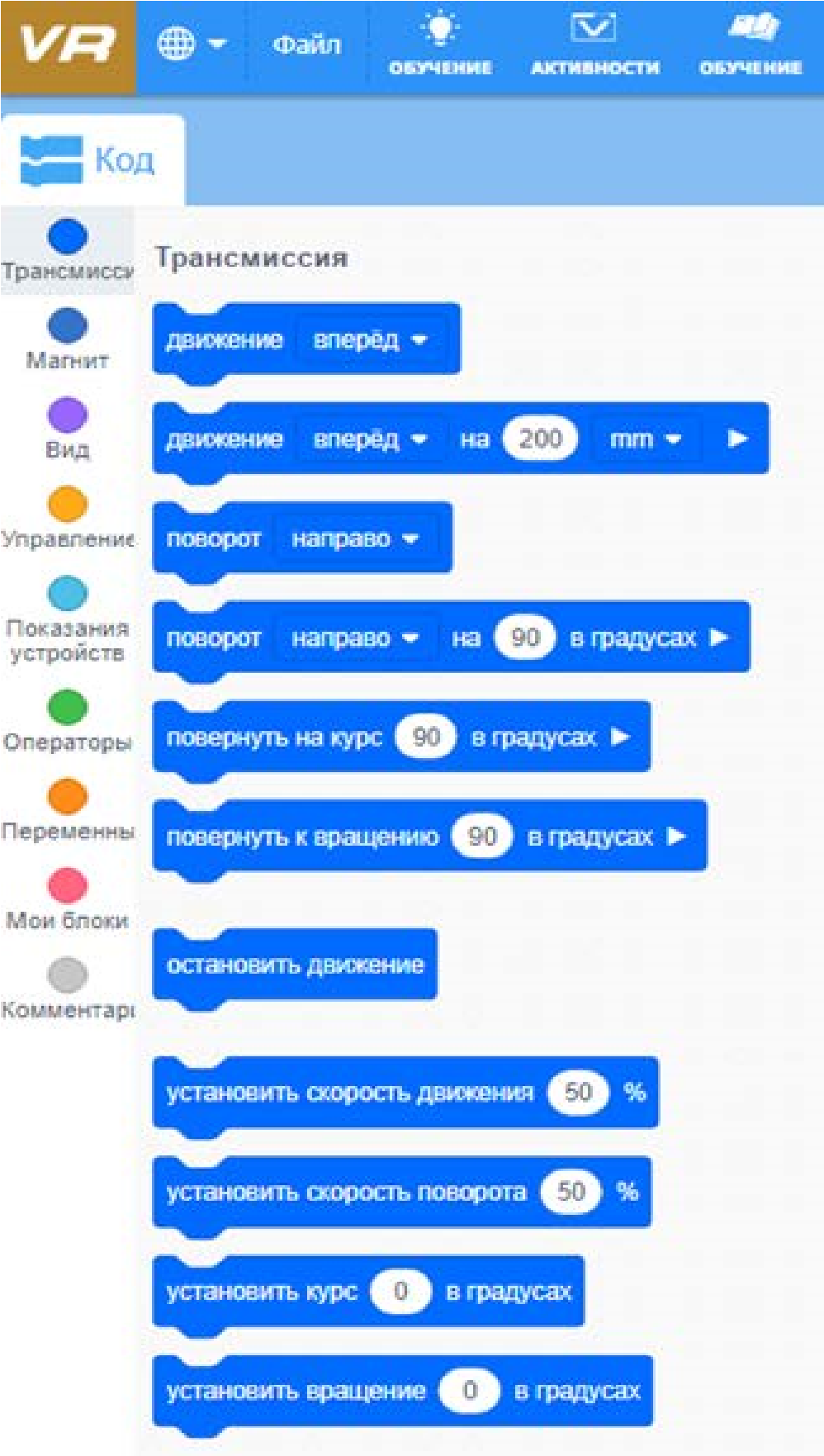
Показать удобство VEXcode VR для различных образовательных учреждений .

VEXcode VR — это способ обогатить опыт компьютерных наук для обучающихся после того, как они открыли для себя программирование в среде Scratch и азарт образовательной робототехники с различными аппаратными платформами . VEXcode VR позволяет учащимся продолжать заниматься робототехникой, даже если поблизости нет физического робота .

Кроме того, VEXcode VR содержит функции, призванные помочь в освоении программирования и сделать STEM и информатику доступными для большего числа обучающихся . Эти функции включают простоту использования, немедленную обратную связь, возможность сделать обучение видимым и помочь с различными реализациями в классе .

**Легко использовать**

VEXcode VR на 100% основан на веб-технологиях, поэтому запустить VEXcode VR просто . Пользовательский интерфейс (ПИ) упрощает навигацию — команды разделены на категории, ПИ построен, избегая большого списка команд в меню . Команды также имеют цветовую кодировку, поэтому пользователи могут легко находить связанные блоки . Область программирования всегда видна; приглашение учащихся начать программирование . VEXcode VR использует готовых роботов и команды трансмиссии . Это позволяет пользователям управлять своим виртуальным роботом за считаные секунды .



***Рис. 3.*** Блок команд «Трансмиссия»

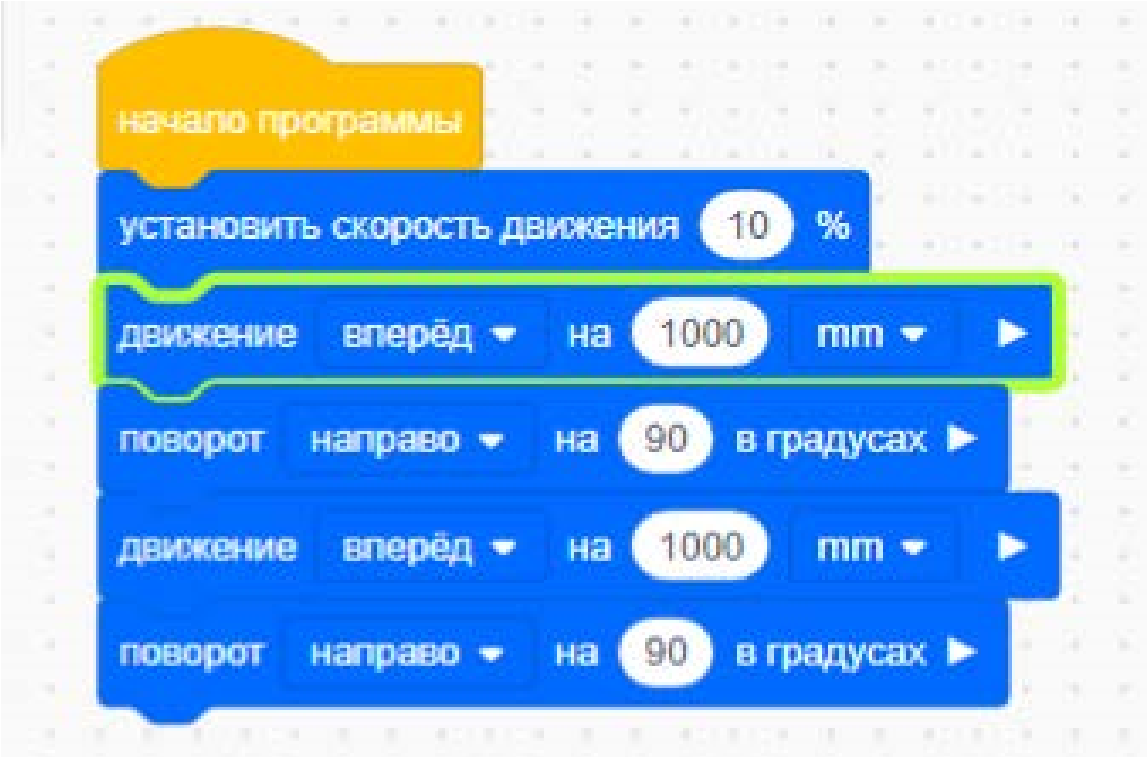
Робот VEX VR оснащён датчиками, элементами управления и множеством физических функций . В VEXcode VR есть только один робот, и он уже настроен заранее . Это устраняет необходимость в настройке конфигурации робота или заранее определённом шаблоне проекта .

**Обратная связь**

VEXcode VR стимулирует проведение экспериментов и игр . Когда обучающиеся запускают проект, они могут сразу увидеть, дал ли их робот желаемый результат . Учителя легко контролируют успехи обучающихся . Проект в VEXcode VR всегда будет работать одинаково — это не всегда происходит с физическим роботом . Это позволяет учителям и ученикам сосредоточиться на логике программирования, а не на физике робота или на игровом поле, на котором работает робот .

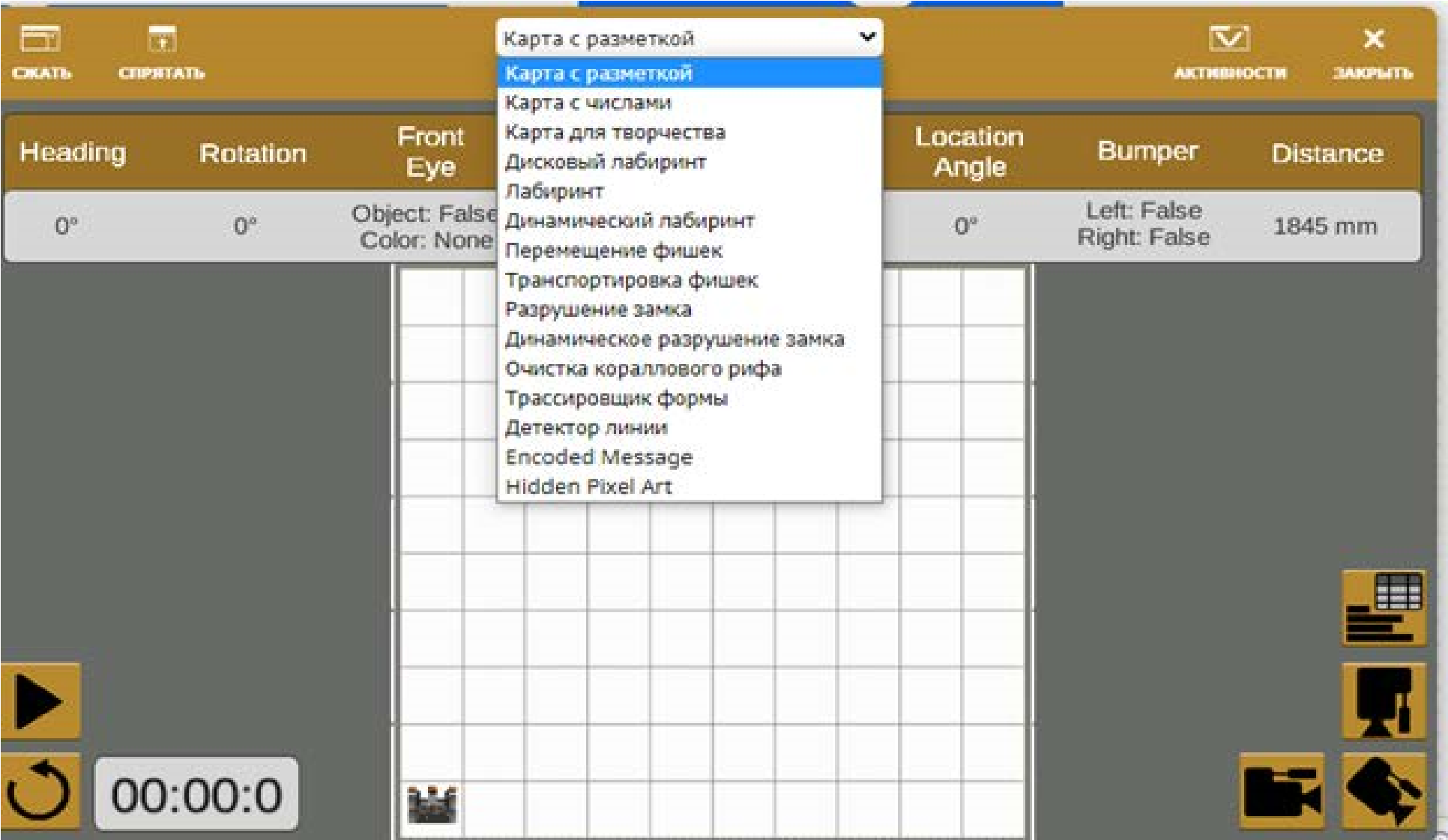
**Игровое поле (площадка) VEXcode VR**

Робот VR всегда начинает выполнение программы с одной и той же точки . Обучающиеся могут добавлять блоки во время работы своего проекта, останавливать проект в любой момент и перезагружать свою виртуальную площадку одним щелчком мыши . Блоки, не подключённые к основному скрипту, просто игнорируются при запуске проекта .



##### *Рис. 4*

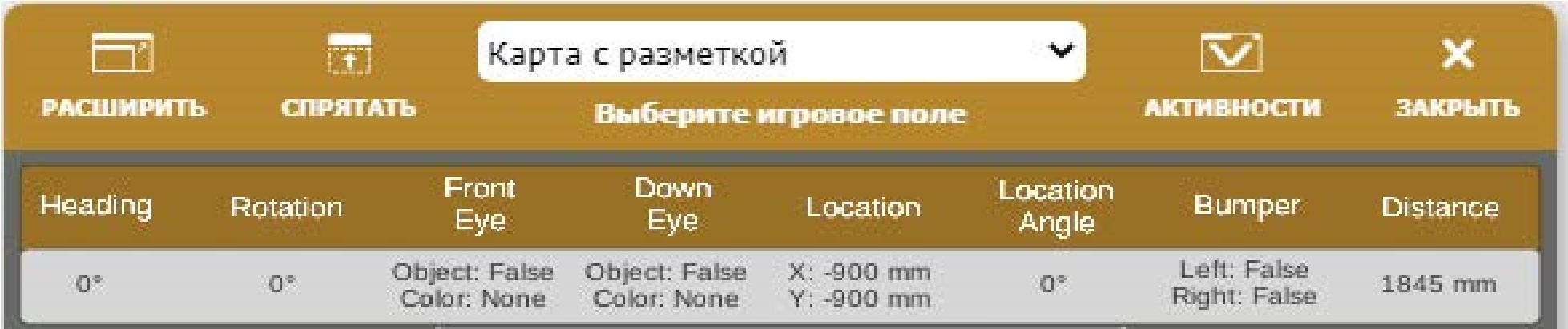
В VEXcode VR ошибок нет . Обучающиеся могут делать логические ошибки при кодировании, но они не будут разочарованы тем, что их проекты не компилируются и не запускаются . Способность VEXcode VR обеспечивать немедленную обратную связь и простота использования побуждают обучающихся учиться в процессе написания кода создавать код с помощью небольших фрагментов, постепенно формируя окончательный вариант программы (скрипта) .



***Рис. 5.*** Игровое поле с выпадающим меню выбора игрового поля

**Обучение стало наглядным**

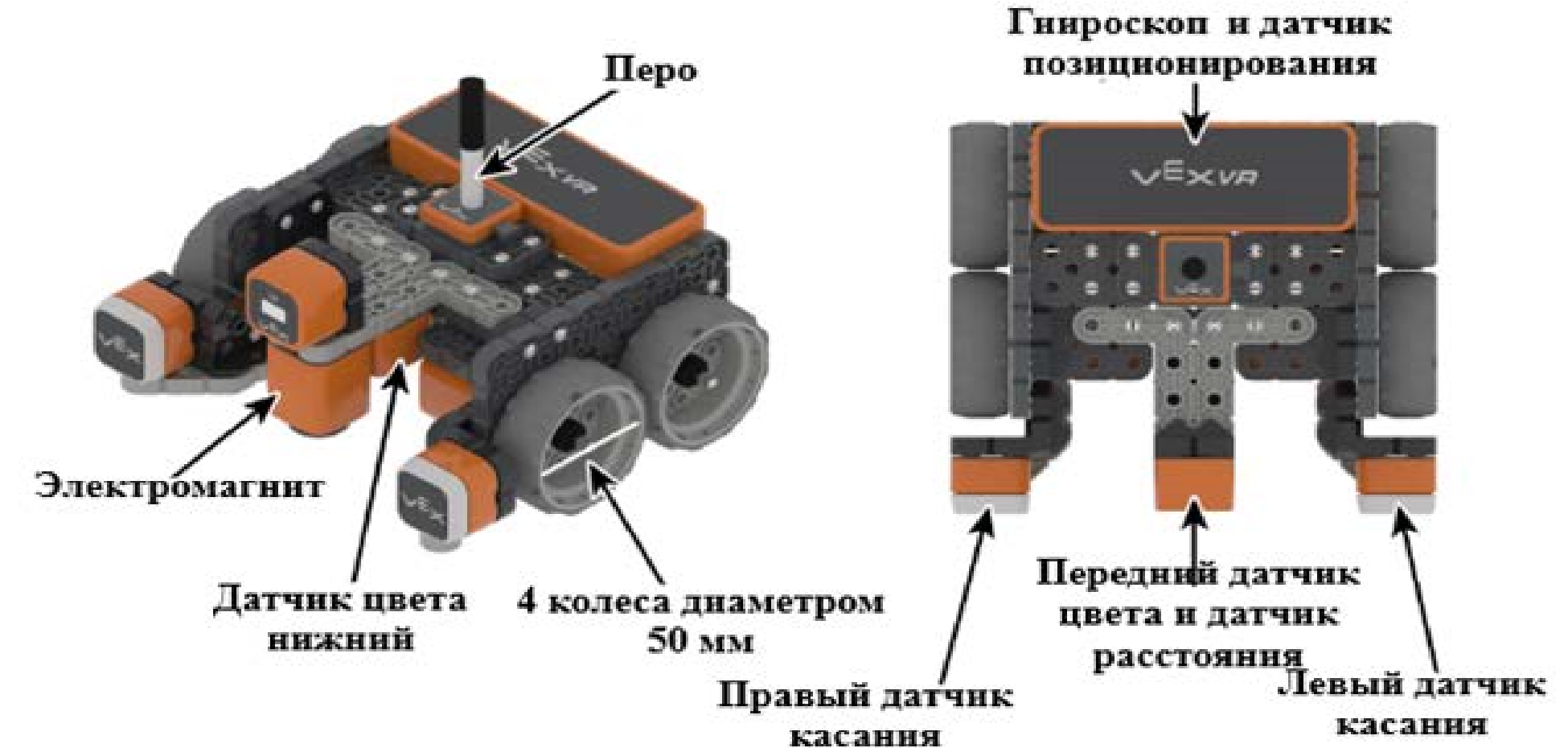
Окно «Игровое поле» в VEXcode VR содержит приборную информационную панель, на которой отображаются все данные датчиков от VR-робота . Каждый раз, когда работает робот VR, учащиеся могут видеть обновление данных датчика в режиме реального времени, получая информацию о том, как эти данные могут быть использованы .



***Рис. 6.*** Приборная панель с показаниями датчиков

VEXcode VR также выделяет блоки в проекте, когда эти блоки выполняются зелёным контуром . Эта функция позволяет учащимся наблюдать за ходом выполнения своих проектов . Когда проект VEXcode VR запущен, выполняемый блок окружён светящейся зелёной рамкой . Эта обратная связь помогает обучающимся понять, почему робот VR выполняет определённый манёвр . Эти данные в реальном времени вместе с выделением выполненных блоков могут помочь обучающимся понять, как робот принимает решение, сделать этот процесс более заметным и конкретным .

VEXcode VR также помогает организовать дистанционную работу . Во всех этих (и не только) ситуациях VEXcode VR может стать отличным дополнением для физических роботов .



***Рис. 7.*** Конфигурация датчиков робота

**Управление роботом**

Робот VR имеет следующие элементы управления:

* трансмиссия с гироскопом . Это включает категорию блоков «Трансмиссия» в наборе инструментов VEXcode VR;
* функция «рисования пером», позволяющая размещать перо вверх (чтобы не рисовать) или вниз (для рисования);
* электромагнит для поднятия дисков с металлическими сердечниками .

**Физические характеристики робота**

Робот VR обладает следующими физическими характеристиками:

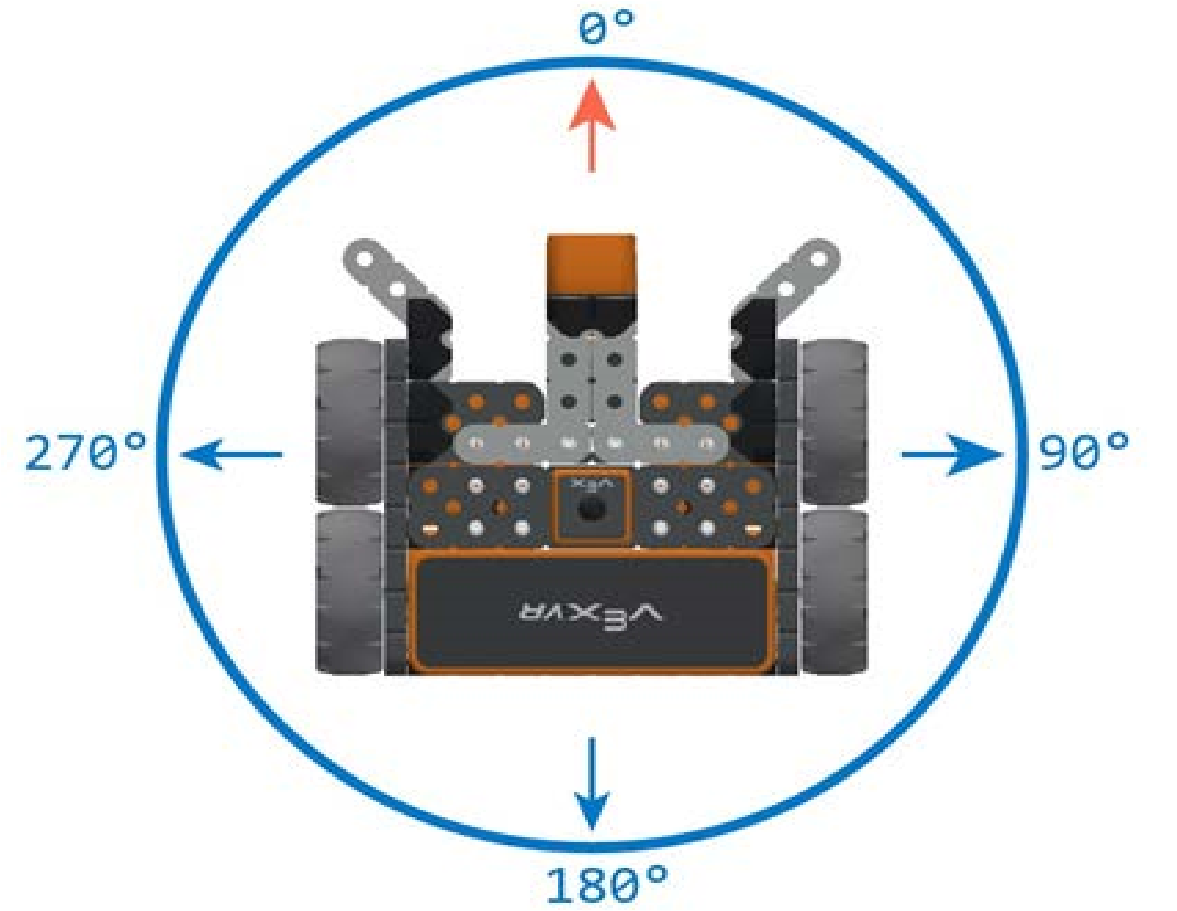
* колёса имеют диаметр 50 мм;
* колёсная база (расстояние между центром переднего колеса и центром заднего колеса) составляет примерно 50,8 мм;
* Длина робота VR составляет 133 мм .

**Датчики роботов**

Робот VR оснащён следующими датчиками:

* моторные энкодеры с углом поворота 360 ° на оборот колеса;
* передний датчик глаза также действует как датчик расстояния и возвращает расстояние до обнаруженного объекта в миллиметрах и дюймах;
* гироскопический датчик, встроенный в трансмиссию . По часовой стрелке положительный .

Вид сверху на робота VEXcode VR



***Рис. 8.*** Направления курса, задаваемые блоком «Курс»

Два датчика цвета, один обращён вперёд, а другой вниз . Эти датчики могут определять наличие объекта . Если есть, датчик также может определять цвет (красный, зелёный, синий, нет) .

Датчик местоположения, который считывает координаты (X, *Y* ) от центральной поворотной точки VR-робота .

2 . Этап актуализации знаний и пробного учебного действия — 15 минут .

Учитель предлагает обучающимся под его контролем выполнить некоторые действия на платформе:

* запрограммировать робота на движение вперёд и назад . Блок программ «Трансмиссия»;
* запрограммировать робота на движение на определенное расстояние . Блок программ «Трансмиссия»;
* запрограммировать робота на повороты в градусах и по курсу . Блок программ «Трансмиссия»;
* проверить загрузку различных учебных полей .

1. . Этап повторения нового материала — 5 минут .

Учитель повторяет кратко новый материал: роботы, датчики, скрипты, игровые поля, как заставить робота двигаться, поворачивать .

1. . Этап проверки понимания и первичного закрепления — 5 минут .

Учитель задаёт вопросы ученикам:

Для чего нужна программа?

Что такое датчики и для чего они нужны?

1. . Информация о домашнем задании, инструктаж по его выполнению — 3 минуты . Задание 1 . Зайти на платформу VEX code VR .

Задание 2 . Написать программу движения робота вперёд и назад на 1000 мм (одна клетка поля равна 200 мм) .

1. . Этап рефлексии деятельности на уроке — 2 минуты .

Учитель интересуется у обучающихся их впечатлениями от урока, что им понравилось и что было непонятно .

**Урок 3 . Исполнительные механизмы конструкторов VEX Тип урока:** комбинированный .

**Цель урока:** создание простейших программ (скриптов), сохранение и загрузка проекта .

**Планируемые результаты**

**Предметные:** получение навыков написания программ (скриптов) на платформе VEXcode VR .

**Метапредметные:** умение соотносить свои действия с планируемыми результатами; осуществлять контроль своей деятельности в процессе достижения результата; определять способы действий в рамках предложенных условий и требований, корректировать свои действия в соответствии с изменяющейся ситуацией .

**Личностные:** стимулирование поиска вариантов на основе имеющихся знаний; готовность и способность обучающихся к саморазвитию и личностному самоопределению; развитие находчивости, умения преодолевать трудности для достижения намеченной цели .

**Время реализации:** 1 академический час .

**Оборудование и материалы:** компьютеры с выходом в сеть Интернет, проектор, интерактивная доска .**Ход урока**

1 . Этап постановки цели и задач урока, мотивации к учебной деятельности 15 — минут .

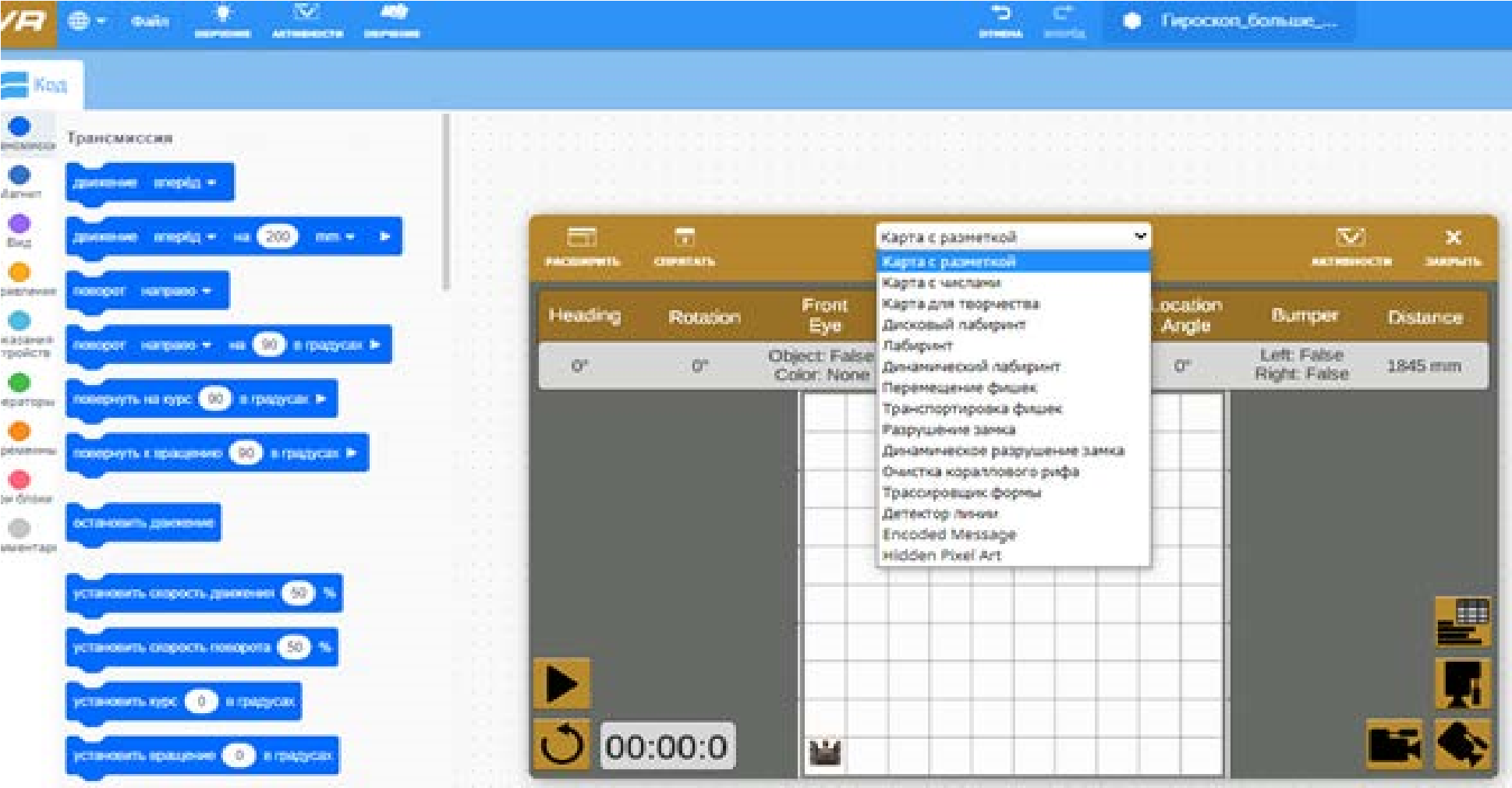
Деятельность учителя — подача нового материала с демонстрацией платформы [http://vr .vex .com](http://vr.vex.com/) .

Примеры различных программ, созданных с помощью платформы, можно открыть в меню **Файл-Открыть примеры .**



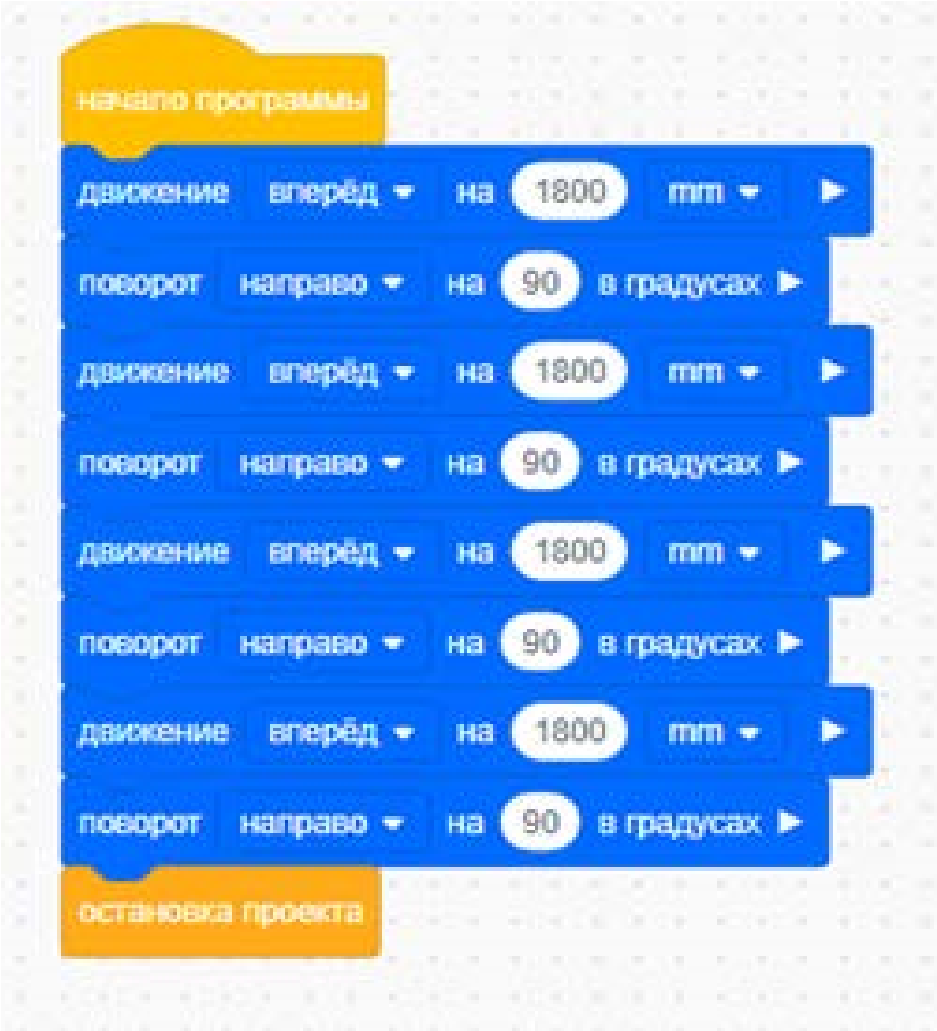
***Рис. 9.*** Примеры программ, созданных на платформе VEXcode VR

Создайте простую программу движения робота по периметру игрового поля **Карта с разметкой** (размеры клетки 200  200 мм) .



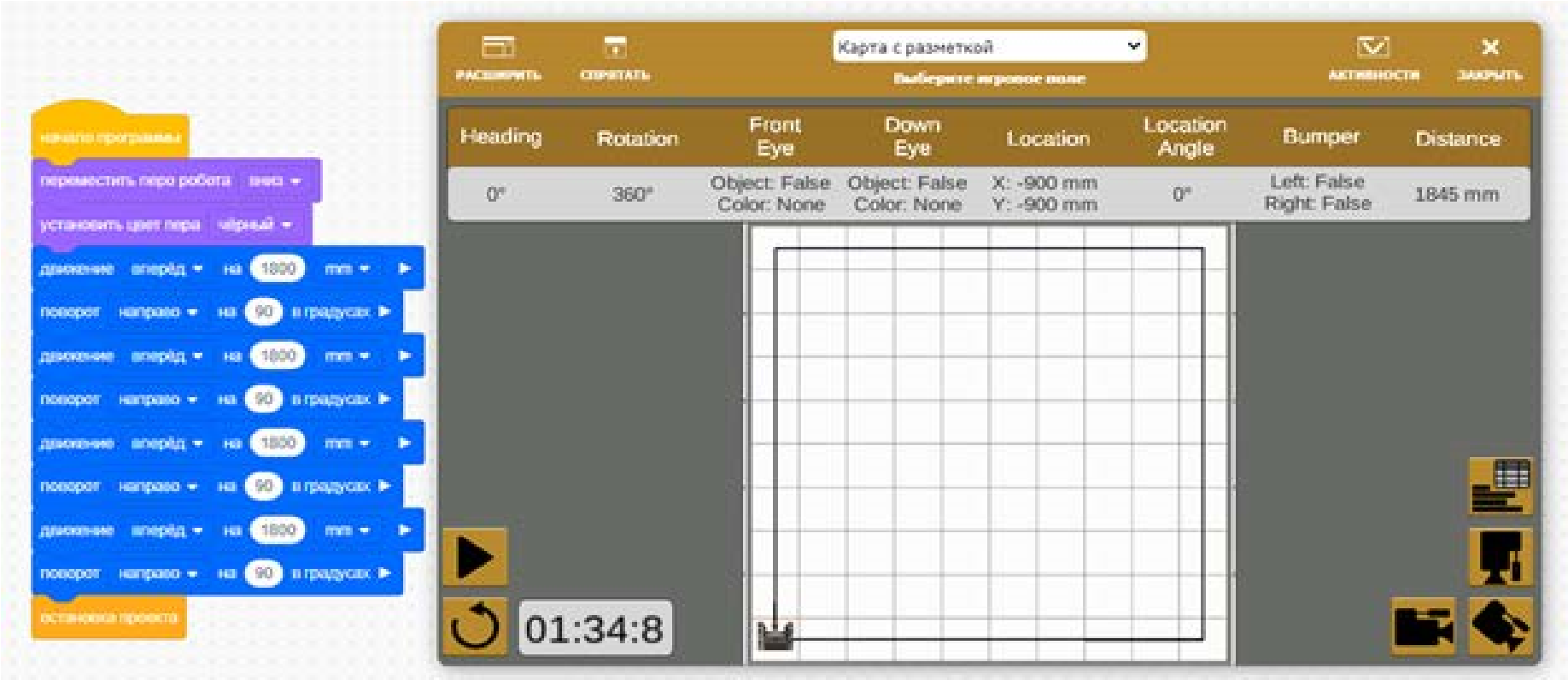
***Рис. 10.*** Игровое поле **Карта с разметкой**

Простая программа будет выглядеть следующим образом:



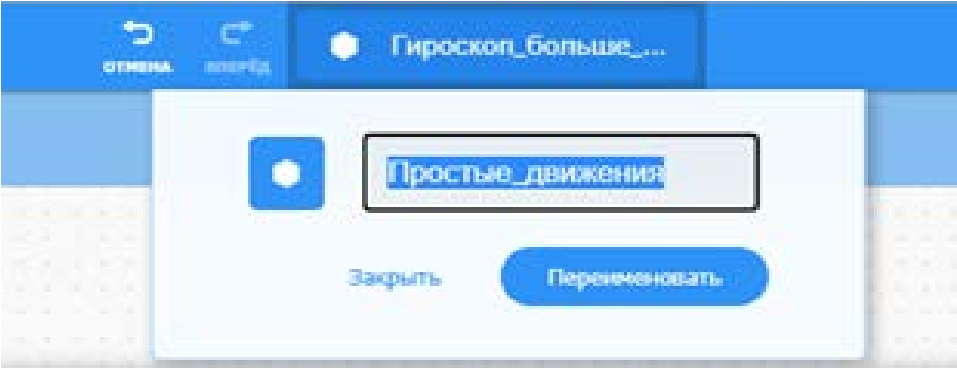
***Рис. 11.*** Скрипт обхода игрового поля по периметру

Можно немного усложнить программу . Пусть робот при обходе поля рисует линию . Для этого перейдём в блок программ **Вид** и перед началом движения поставим два блока, показанные на рисунке ниже .



***Рис. 12.*** Скрипт рисования роботом квадрата

Созданную программу необходимо сохранить на своём компьютере или планшете . Для этого нужно кликнуть мышкой на верхней панели и ввести имя файла .

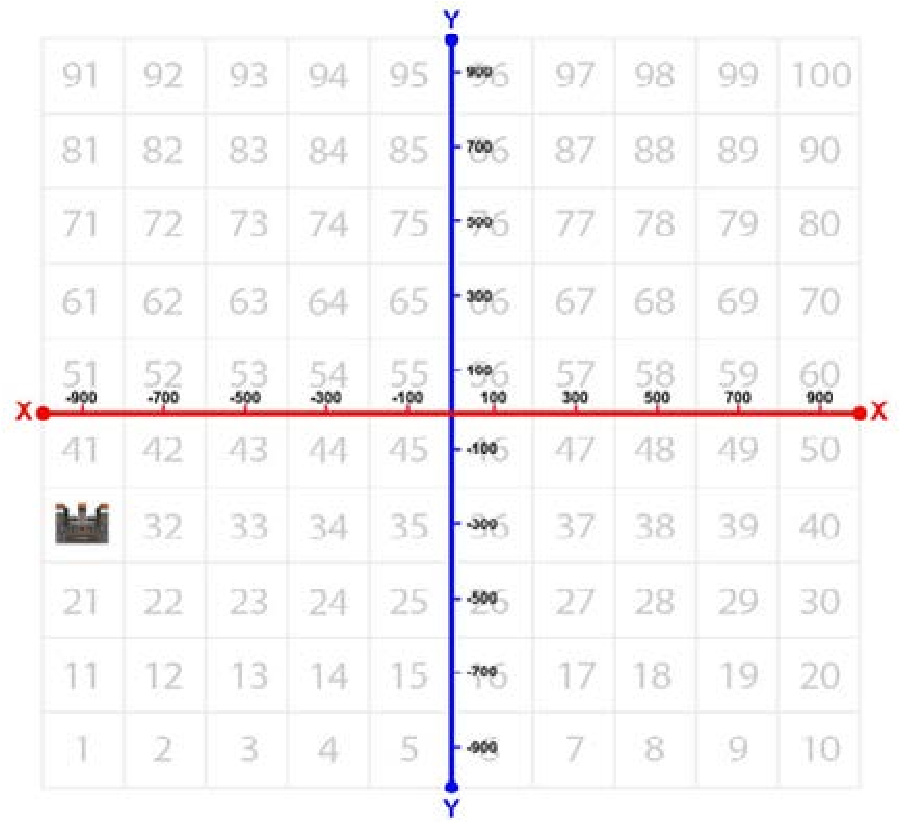


***Рис. 13.*** Переименование рабочего файла

Сохранить этот файл на своё устройство можно через меню **Файл-Сохранить на ваше устройство**, файл будет скачан в папку **Загрузки** рабочего устройства .

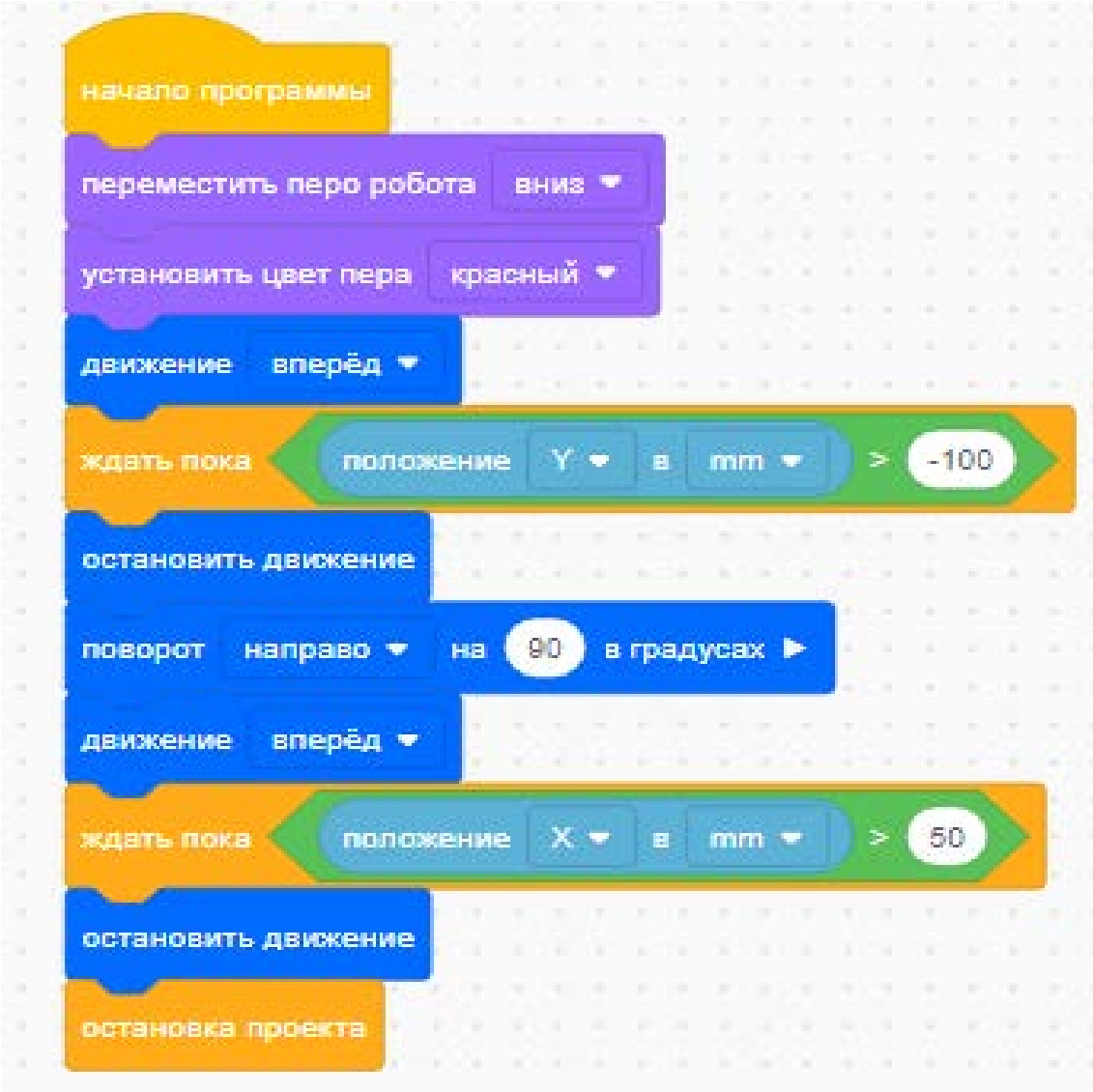
Также рассмотрим позиционирование робота на игровой площадке с помощью датчика положения (гироскопа) .

Игровая площадка разделена на координатные плоскости (декартовы координаты) как показано на рисунке 14 .



***Рис. 14.*** Координатная плоскость игрового поля

Движение по координатной плоскости задаётся следующим образом:



***Рис. 15.*** Перемещение робота по координатам

1. . Этап актуализации знаний и пробного учебного действия — 15 минут .

Учитель предлагает обучающимся под его контролем выполнить некоторые действия на платформе, используя знания, полученные в ходе изложения нового материала .

1. . Этап актуализации знаний и пробного учебного действия — 15 минут .

Учитель предлагает обучающимся под его контролем выполнить некоторые действия на платформе:

Задание 1 . Напишите скрипт рисования роботом квадрата со сторонами разного цвета .

Задание 2 . Напишите скрипт рисования роботом ромба со сторонами разного цвета, используя гироскоп .

1. . Этап повторения нового материала — 5 минут .

Учитель повторяет кратко новый материал: программирование простых движений робота, датчики, скрипты, игровые поля, как заставить робота двигаться, поворачивать, рисовать .

1. . Этап проверки понимания и первичного закрепления — 5 минут .

Учитель задаёт вопросы ученикам:

* Для чего нужна программа «Трансмиссия»?
* Какой блок программ отвечает за движение робота?
* Как регулировать скорость движения робота?
* Как работает блок программ «Курс»?
* Как направить движение робота по диагонали игрового поля?
* Как управлять магнитом и для чего?

1. . Информация о домашнем задании, инструктаж по его выполнению — 3 минут .

Задание . Написать программу рисования на игровом поле роботом инициалов обучающегося .

1. . Этап рефлексии деятельности на уроке — 2 минут .

Учитель интересуется у обучающихся об их впечатлениях от урока, что им понравилось и что было непонятно .

#### Урок 10 . Блок команд «Управление»

**Цель урока:** ознакомить обучающихся с ветвлениями и циклами .

**Тип урока:** комбинированный .

**Планируемые результаты**

**Предметные:** получение навыков по использованию условного оператора и операторов цикла, разработка программ, реализующих разветвляющийся и циклический алгоритмы .

**Метапредметные:** делать выводы в процессе работы и по её окончании, корректировать намеченный план, ставить новые цели; умение соотносить свои действия с планируемыми результатами, осуществлять контроль своей деятельности, корректировать свои действия в соответствии с изменяющейся ситуацией; умение оценивать правильность выполнения учебной задачи .

**Личностные:** формирование мотивации к обучению и целенаправленной познавательной деятельности; формирование целеустремлённости и усидчивости в процессе творческой, исследовательской работы и учебной деятельности .

**Время реализации:** 1 академический час .

**Оборудование и материалы:** компьютеры с выходом в сеть Интернет, проектор, интерактивная доска

**Ход урока**

2 . Этап постановки цели и задач урока, мотивации к учебной деятельности — 15 минут .

Деятельность учителя: подача нового материала с демонстрацией командам управления в среде vr .vex .com .

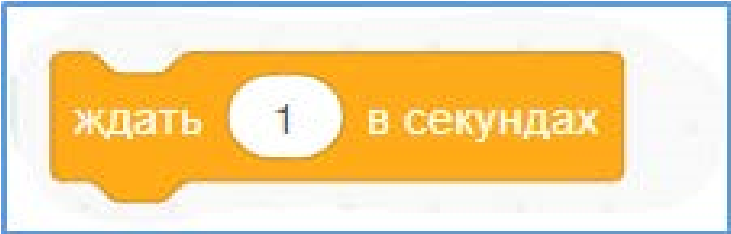
**Блок команд управления в VEX VR**

Вы уже знакомы с понятиями циклов и ветвлений, поэтому самое время перейти к изучению этих структур в среде программирования VEX VR .

Почему данная группа блоков имеет название «Управление»? На самом деле всё довольно просто . Эти блоки позволяют управлять поведением нашего робота в зависимости от обстановки и окружающих его факторов . Без этих блоков нельзя получить интеллектуальную систему, способную ориентироваться в пространстве без дополнительной помощи со стороны оператора или команды разработчиков . На рисунке 16 показан блок команд управления . Обратите внимание, что многие блоки схожи между с собой по названиям и, соответственно, обладают похожими свойствами . Сегодня познакомимся с блоками**Ждать** и **Повто-**

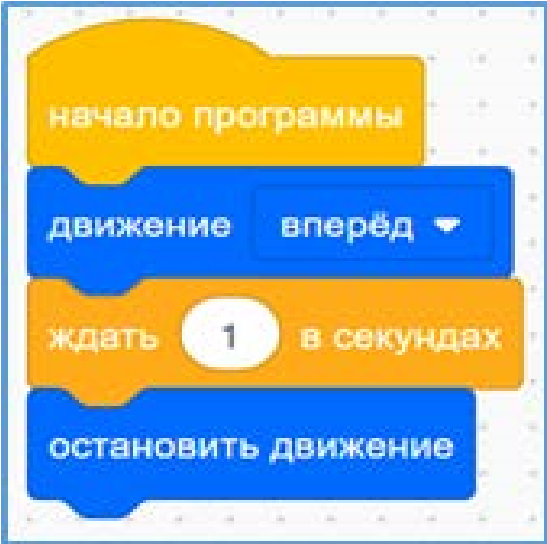
**рять**(рис . 16) . ***Рис. 16.*** Блок команд управления

Рассмотрим блок **Ожидания** (рис . 17) .



***Рис. 17.*** Блок **Ожидания**

Данный блок позволяет удерживать действия, которые происходили до его вызова . Рассмотрим пример (рис . 18) .

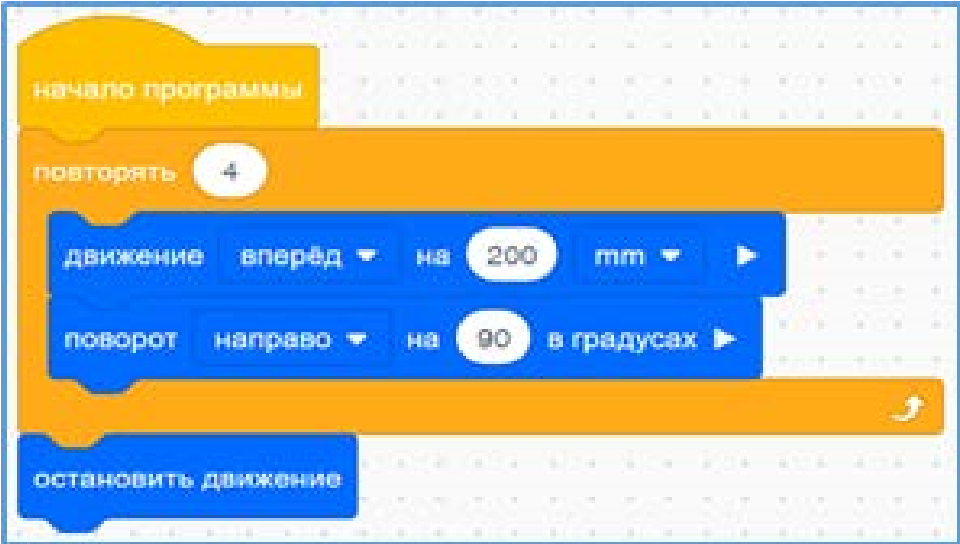


***Рис. 18.*** Пример программы с использованием блока **Ожидания**

Что произойдёт в данном отрывке кода? Робот будет двигаться вперёд, после чего остановится . Если же убрать данный блок, то, соответственно, визуально ничего не произойдёт, так как робот мгновенно выполнит обе команды и даже не сдвинется с места .

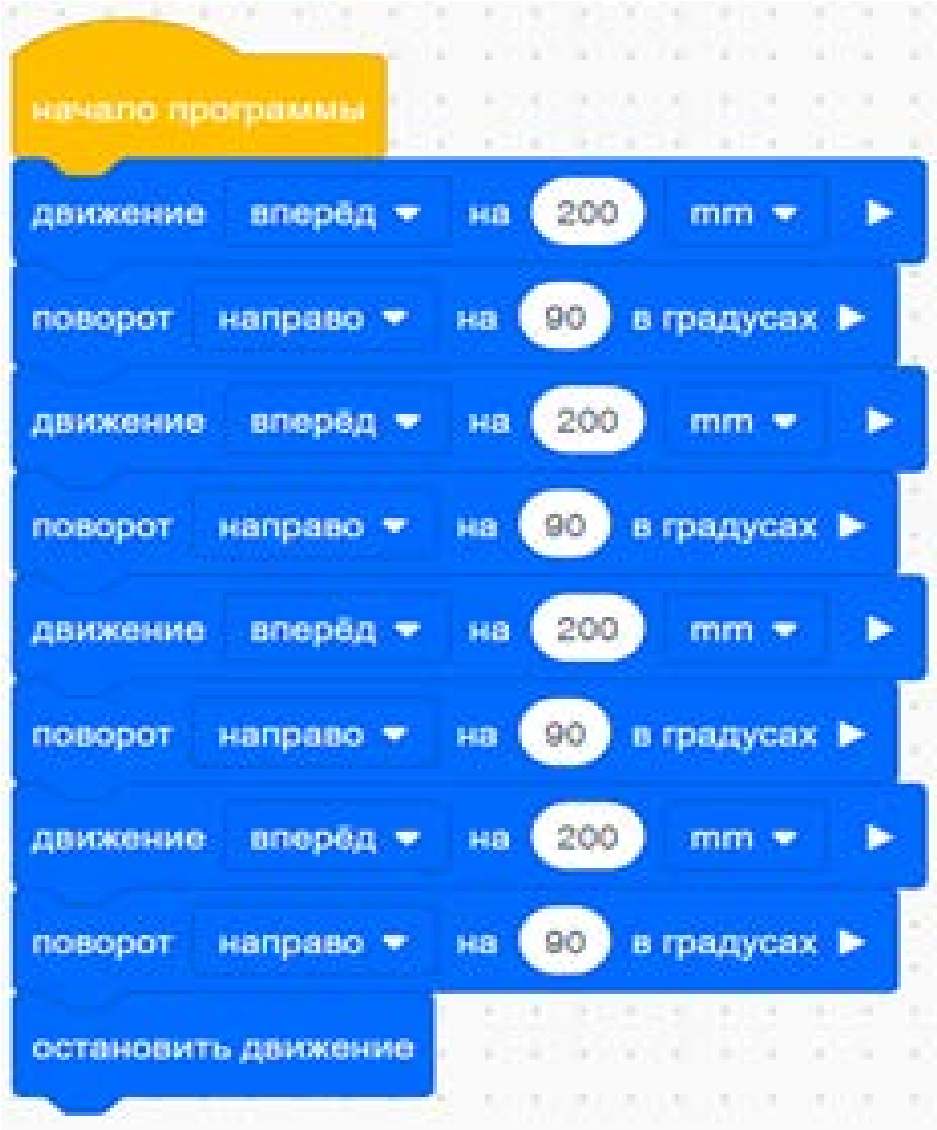
**Важно!** Блоки программы после блока **Ожидания** не будут выполняться, пока не истечёт время, установленное внутри этого блока .

Перейдём к блоку **Повторять** . На самом деле, как вы уже поняли, данный блок повторяет некоторое число действий, количество которых определено внутри этого блока . Рассмотрим программу для изображения квадрата (рис . 19) .



***Рис. 19.*** Программа по объезду квадрата

Давайте более подробно рассмотрим данную программу . Почему бы просто не использовать группу из одинаковых блоков для решения задачи? Никто не запрещает использовать несколько одинаковых блоков вместо циклов, но при написании кода большего объёма получим гораздо больше строк, что, несомненно, усложнит читаемость всей программы . Для сравнения на рисунке 20 показан тот же самый код, но без использования цикла .



***Рис. 20.*** Объезд по квадрату без цикла

Как вы видите, циклы упрощают читаемость кода, не влияя на его функциональность .

1. . Этап актуализации знаний и пробного учебного действия — 15 минут .

Учитель предлагает обучающимся под его контролем выполнить некоторые действия на платформе:

Задание . Написать программу для объезда шестиугольника, при этом используя только блоки движения без указания расстояния, блок поворота без указания угла и блок ожидания .

1. . Этап повторения нового материала — 5 минут .

Учитель повторяет кратко новый материал: блоки **Управления**, блок **Ожидания** и блок **Повторения** .

1. . Этап проверки понимания и первичного закрепления — 5 минут .

Учитель задаёт вопросы ученикам:

* Почему блоки управления так называются?
* Зачем нужны блоки управления?
* Что позволяет делать блок ожидания?
* Что делает блок повторения?

1. . Информация о домашнем задании, инструктаж по его выполнению — 3 минут .

Задание 1 . Реализовать алгоритм объезда роботом треугольника и прямоугольника .

Задание 2 . Реализовать алгоритм объезда роботом треугольника и прямоугольника без использования блоков поворота с указанием угла поворота (необходимо использовать только блок ожидания) .

1. . Этап рефлексии деятельности на уроке — 2 минуты .

Учитель интересуется у обучающихся об их впечатлениях от урока, что им понравилось и что было непонятно .

**Лабораторная работа 10**

#### Ветвления на базе платформы VEXcodeVRТеоретическая часть

Иногда роботу бывает необходимо единожды принять выбор относительно своего текущего действия . Это могут быть повороты направо или налево, отъезды вперёд или назад в определённый момент времени без необходимости перепроверки действий, как в циклах . В этом случае лучше всего использовать блоки **Если-тогда** и **Если-тогда-иначе**, которые позволяют преобразовать действия, совершаемые роботом, в систему с выходами да/нет .

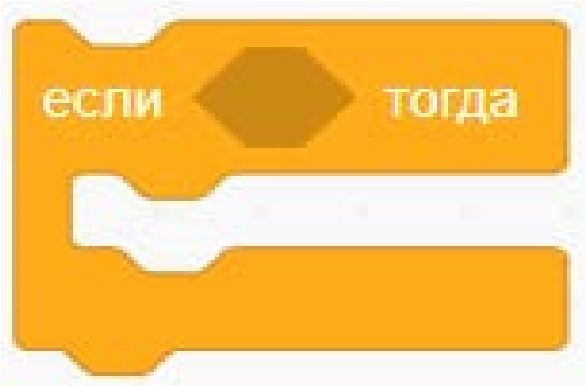
**Практическая часть**

**Цель работы:** ознакомить обучающихся с ветвлениями на базе платформы vr .vex .com .

##### Ход лабораторной работы

На предыдущем занятии рассмотрели блоки ожидания и повторения и сегодня продолжим изучение остальных блоков модуля управления .

В первую очередь обратим внимание на блок **Если-тогда** (рис . 21) .



***Рис. 21.*** Блок **Если-тогда**

Тело данного блока выполняется в том случае, если условие, обозначенное в шестиугольной вставке, является правдой, и не выполняется в обратном случае . Рассмотрим следующий пример (рис . 22) .



***Рис. 22.*** Пример использования условного блока

Давайте подробнее рассмотрим, что будет происходить в данном примере . Как вы видите, в качестве условия в блоке дано логическое выражение, содержащее справа сумму чисел, а слева константу . Если сравнение окажется правдой, то в окно вывода будет напечатано True, если условие окажется ложным — False .

Кроме того, существует ещё один блок управления, схожий с вышеуказанным, но при этом имеющий ещё одну дополнительную ветвь иначе (рис . 23) .



***Рис. 23.*** Блок **Если-тогда-иначе**

Рассмотрим следующую программу для понимания разницы между блоками

**Если-тогда** и **Если-тогда-иначе** (рис . 24) .



***Рис. 24.*** Демонстрация работы **Если-тогда-иначе**

Рассмотрим код данной программы . Выражение *после если* в данном случае является ложным, поэтому будет выполняться часть после *иначе*, то есть перо робота станет зелёного цвета и на экране появится надпись SecondOver . Обратите внимание, что Over будет выведено в любом случае, так как находится за пределами блока условного оператора .

Задание 1 . Написать программу для вывода на экран True или False в зависимости от решения простых математических выражений, которые учащиеся должны разместить внутри условия условного блока .

Задание 2 . Реализовать программу для поворота роботом перед препятствием на произвольный угол .

**Контрольные вопросы:**

1. . Как можно использовать блок **Если-тогда**?
2. . В чём отличие блоков **Если-тогда** и **Если-тогда-иначе**?
3. . В каких ситуациях лучше всего использовать блок **Если-тогда-иначе**?

**Лабораторная работа 11**

#### Циклы на базе платформы VEXcodeVRТеоретическая часть

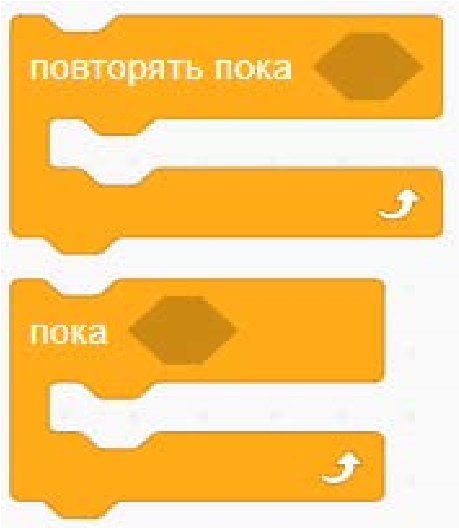
Известно, что бывают ситуации, при которых необходимо проверять условие несколько раз, прежде чем принять окончательное решение . Можно использовать совокупность блоков повторения и условные блоки, но для корректной работы повторений должно быть огромное число (десятки миллионов) и подбор точного числа может сильно затруднить общее время создания программы . Во избежание этого в VEX VR используются циклы **Повторять пока** и **Пока .**

**Практическая часть**

**Цель работы:** ознакомить обучающихся с циклами на базе платформы vr .vex .com

##### Ход лабораторной работы

В VEX VR используются циклы**Повторять пока** и **Пока** (рис . 25) .



***Рис. 25*** — Блоки циклов

Рассмотрим простой пример программы, который покажет отличия между двумя видами циклов (рис . 26) .

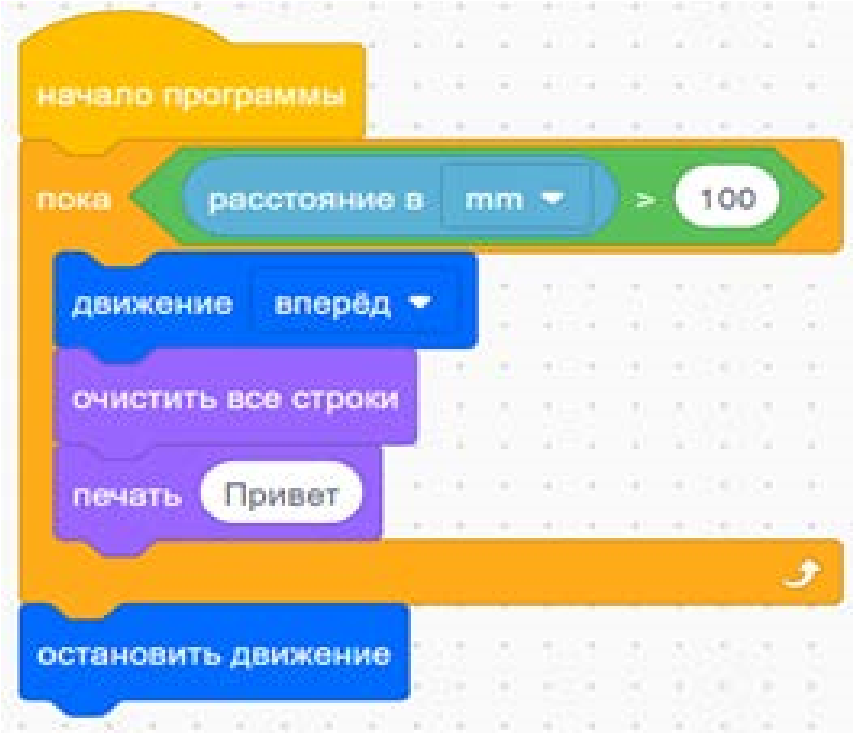


***Рис. 26.*** Отличия циклов в VEX VR

На первый взгляд может показаться, что отличия в условиях блоков циклов должны привести к выполнению абсолютно разных действий, но это не так . Природа блока **Повторять пока** такова, что действия внутри блока будут выполняться до тех пор, пока условие будет ложным . Об этом важно помнить при составлении программы .

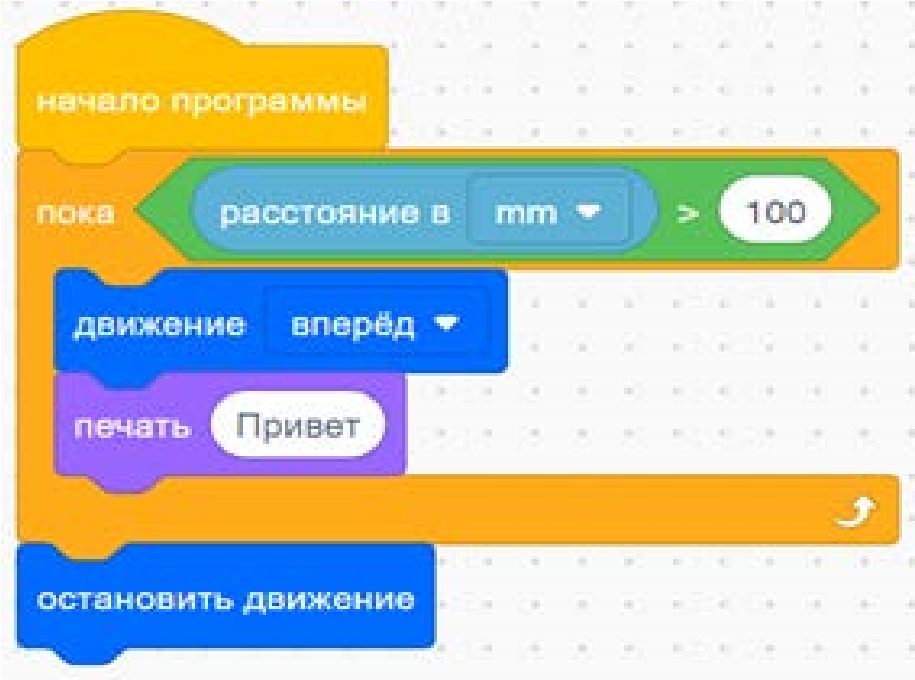
Давайте пока отложим блок**Повторять пока** и напишем простую программу для вывода на экран нескольких строк текста в зависимости от показаний датчика расстояния

(рис . 27) .



***Рис. 27.*** Работа с циклом**Пока**

Давайте построчно рассмотрим данную программу . В первую очередь создаём блок цикла, который будет работать до тех пор, пока робот не увидит перед собой какое-либо препятствие . На пути всего следования до препятствия робот будет выводить в консоль вывода слово «Привет» огромное число раз в силу высокой производительности управляющего контроллера . Если добавить к программе всего одну строку, то слово «Привет» будет выведено только один раз (рис . 28) .



***Рис. 28.*** Вывод одного слова на экран во время использования цикла

Задание 1 . Написать программу для вывода на экран *привет* . При каждом проходе цикла привет должно выводиться с новой строки .

Задание 2 . Написать программу по изображению роботом квадрата (в качестве повторения пройденного материала) .

**Контрольные вопросы:**

1. . В чём отличия блоков **пока** и **повторять пока**?
2. . Приведите примеры программы, использующей блок **пока** (в виде псевдокода) .
3. . Приведите примеры программы, использующей блок **повторять пока** (в виде псевдокода) .

**Лабораторная работа 12**

#### Блок Всегда, блок Прерывания и блок Ждать пока

##### Теоретическая часть

На предыдущем занятии рассмотрели блоки циклов и сегодня продолжим изучение оставшихся блоков модуля управления: блока **Всегда**, блока **Прерывания** и блока **Ждать пока** .

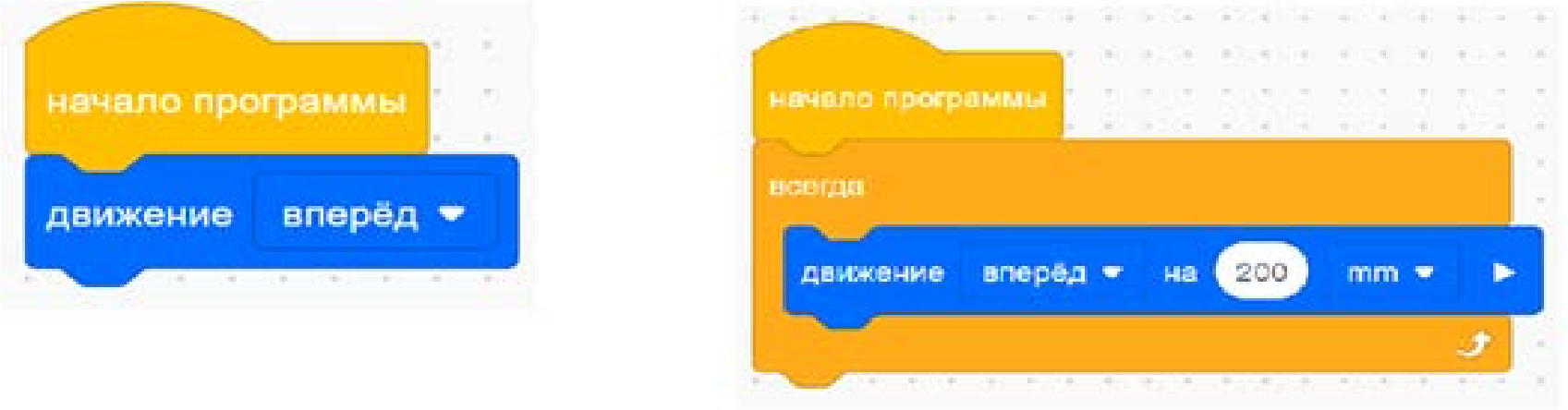
Блок**Всегда** представляет собой бесконечный цикл без условия, который будет выполняться на протяжении всей программы независимо от действий, происходящих внутри и снаружи него

##### Практическая часть

**Цель работы:** ознакомить обучающихся с блоками **Всегда**, блоками **Прерывания** и блоками **Ждать пока .**

##### Ход лабораторной работы

Рассмотрим две программы, которые являются абсолютно идентичными, но при этом продемонстрируют принцип работы блока**Всегда** (рис . 29) .

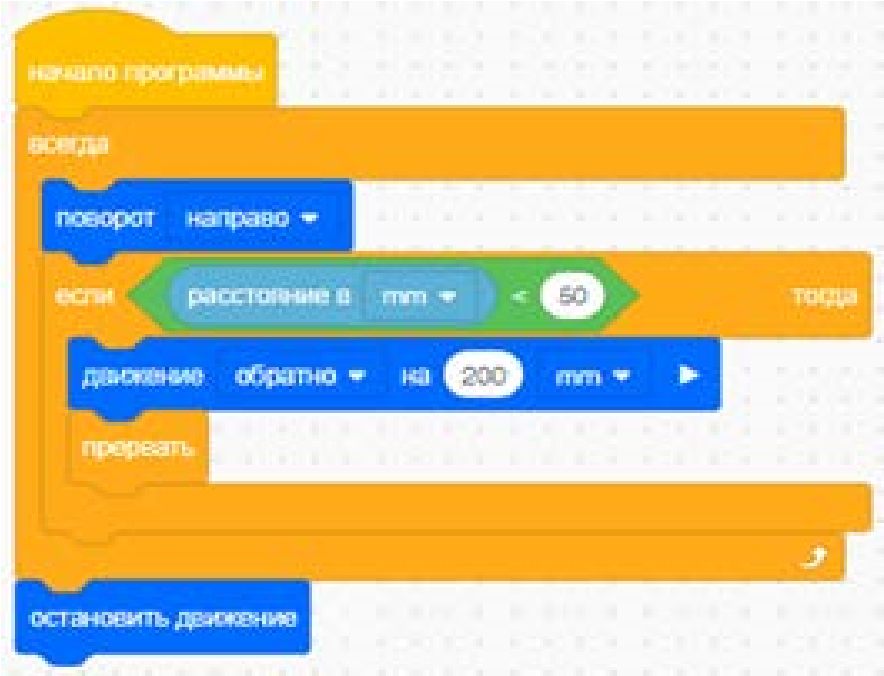


***Рис. 29.*** Аналог бесконечного цикла

Как вы можете заметить, в данной программе блок движения включает себя бесконечный цикл, что позволяет роботу двигаться бесконечно вперёд при добавлении этого блока в программу .

Для того чтобы прервать выполнение бесконечного цикла, можно использовать блок **Прерывания** .

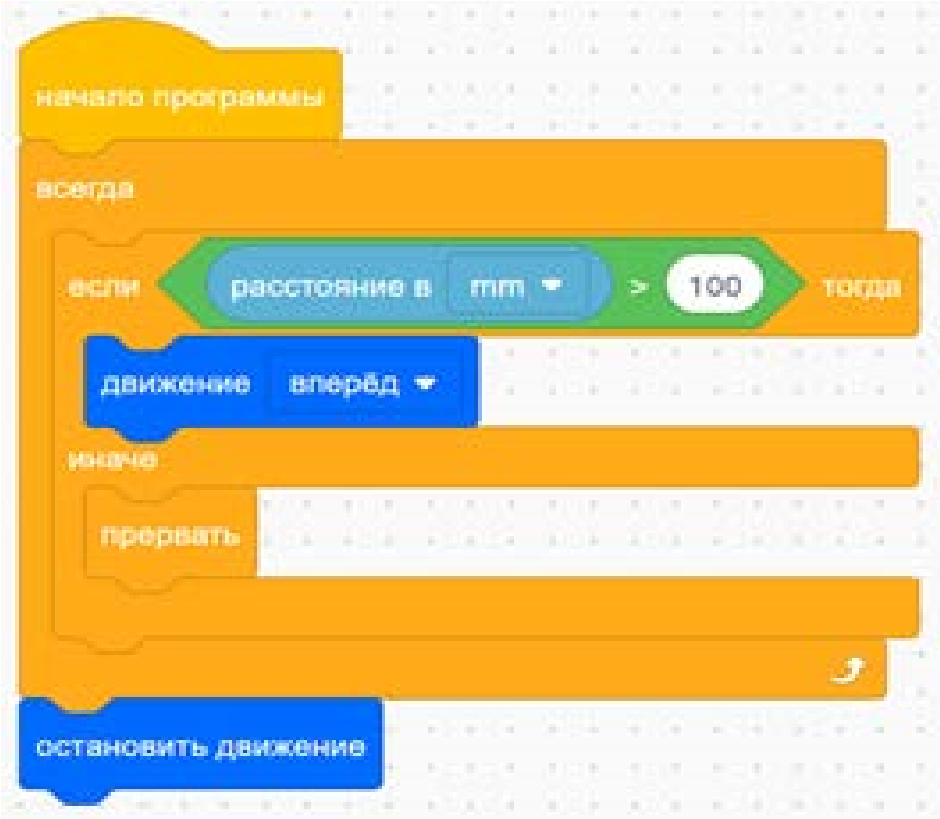
Рассмотрим программу, в которой робот будет ехать вперёд до тех пор, пока датчик расстояния не увидит перед собой некоторое препятствие на расстоянии 100 мм (рис . 30) .



***Рис. 30.*** Программа для демонстрации работы цикла прерывания

Данная программа выполняет все те же действия, что и использование цикла Пока в предыдущем уроке, но при этом наглядно демонстрирует способ прерывания цикла Всегда .Использованиеблока**Прервать** помогает избежать поломок робота, так как обычно используется в контексте аварийного прерывания выполнения программы . Давайте рассмотрим более сложный вариант программы, в котором использование цикла будет оправдано срабатыванием датчика расстояния на расстоянии меньшем, чем планируемое

(рис . 31) .

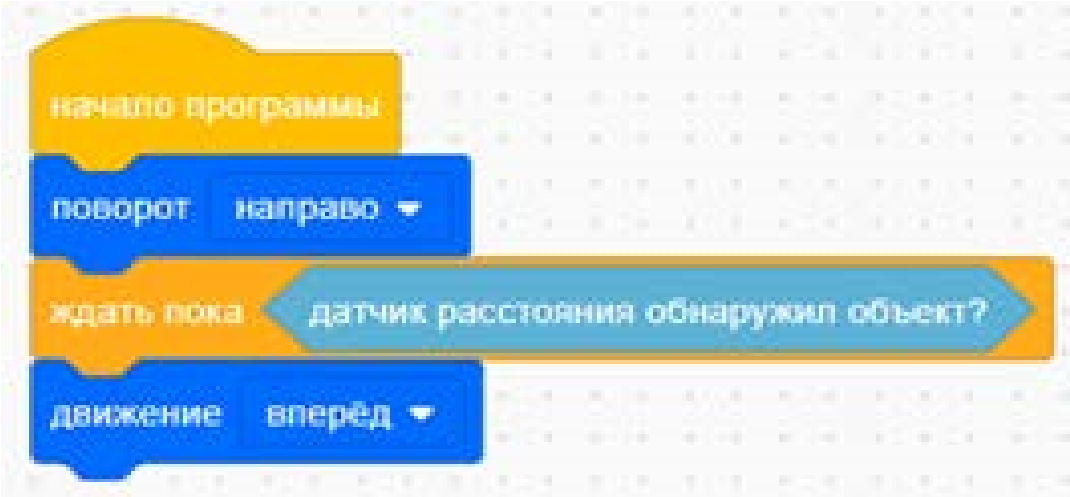


***Рис. 31.*** Пример программы

В данной программе робот будет совершать поворот до тех пор, пока не обнаружит перед собой препятствие . После этого он отъедет назад на 200 мм и остановит своё движение . Данный алгоритм может быть применим к случаям, когда на роботе установлено техническое зрение и ему необходим лучший обзор для идентификации перед собой объектов .

Кроме того, необходимо обратить внимание на блок **Ждать пока** (рис . 32), который позволяет удержать выполнение действий до этого блока .

Рассмотрим программу, которая может помочь в прохождении некоторых испытаний в среде VEX VR .



***Рис. 32.*** Блок**Ждать пока**

В данной программе робот при обнаружении объекта постарается вытолкнуть его с поля . Данный код поможет в решении задач на последующих уроках .

Задание 1 . Написать программу объезда всего периметра поля «Карта с разметкой» с использованием блока **Ждать пока .**

Задание 2 .Написать программу по завершении роботом действий при обнаружении под собой красного цвета .

Задание 3 . Реализовать программу для активации роботом отъезда назад при первоначальном движении робота вперёд .

**Контрольные вопросы:**

1. . Зачем нужен блок **Прерывания**?
2. . Приведите примеры программы, использующей блок **Прерывания** (в виде псевдокода)
3. . Как работает блок**Ждать пока**?

Тематическое планирование «Робототехника»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **п/п** | **Тема** | **Содержание** | **Целевая**  **Установка урока** | **Кол-во**  **часов** | **Основные виды**  **деятельности**  **обучающихся на уроке/**  **внеурочном занятии** | **Использование**  **оборудование** |
| 1 | Робот. Базовые  понятия | История развития робототехники.  Введение понятия «робот». Поко-  ления роботов. Классификация  роботов. Кибернетическая систе-  ма. Обратная и прямая связь. Датчики | Вводное занятие.  Знакомство. Прави-  ла техники безопасности | 1 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Отве-  ты на контрольные во-просы | Компьютер, проектор, интерактивная доска |
| 2 | Знакомство со  средой VEXcodeVR | Основные фрагменты интерфейса  платформы. Панель управления,  блоки программы, датчики, игро-  вая площадка, экран датчиков и  переменных, кнопки управления | Ознакомить обучаю-  щихся с платформой VEXcode VR | 1 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные во-просы | Компьютер, проектор, интерактивная доска |
| 3 | Исполнительные  механизмы кон-  структоров VEX | Создание простейших программ  (скриптов), сохранение и загрузка проекта | Научить обучающих-  ся создавать про-  стейшие программы  (скрипты) на платформе VEXcode VR | 2 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 4 | Программируемый контроллёр | Математические и логические  операторы, блоки вывода информации в окно вывода, блоки  трансмиссии | Ознакомить обучаю-  щихся с блоками  управления роботом  (блоки вывода, бло-  ки трансмиссии) | 2 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 5 | Основные блоки | Блоки управления, блоки переменных, блоки датчиков | Ознакомить обучаю-  щихся с группой  блоков  управления  роботом и возможно-  стями программирования  с их помощью. | 2 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные вопросы. | Компьютер, проектор, интерактивная доска |
| 6 | Датчик местоположения, направление движения | Местоположение VR-робота.  Скрипт проекта с датчиком местоположения | Ознакомить обучаю-  щихся с **датчиком**  **местоположения** | 2 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 7 | Датчики цвета | Датчики цвета и их направление.  Игровое поле «Дисковый лабиринт» | Ознакомить обучаю-  щихся с датчиками  цвета (верхний и  нижний), движением  робота по дисковому  лабиринту, рассмо-  треть отражения  данных на панели  управления и консо-  ли экрана | 2 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 8 | Датчик расстояния | Датчик расстояния. Простой лабиринт. Динамический лабиринт | Ознакомить обучаю-  щихся с датчиком  расстояния, рассмо-  трение различных  типов лабиринта  (простой и динами-  ческий) | 1 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 9 | Знакомство со  средой Scratch | Изучение основных элементов интерфейса среды Scratch, приёмы  работы со спрайтами, приёмы работы с фоном, составление простых скриптов из различных блоков | Ознакомление со  средой Scratch, изу-  чение основных ин-  струментов среды | 2 | Наблюдение за работой учителя, самостоятельная работа со средой  Scratch, ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 10 | Линейные алгоритмы | Основные приёмы составления  линейных алгоритмов в среде Scratch, решение задач на составление линейных алгоритмов | Ознакомление с по-  строением и выпол-  нением линейных ал-  горитмов, 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 |
| 11 | Работа с переменными | Основные приёмы добавления  переменных в среде Scratch, использование основных блоков  для работы с переменными, основные приёмы составления про-  грамм с использованием переменных в среде Scratch | Ознакомление с ос-  новами работы с пе-  ременными в среде  Scratch | 2 | Наблюдение за работой  учителя, самостоятельная работа со средой  Scratch, ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 12 | Условные алгоритмы | Ознакомление с понятием «условный алгоритм», основные приёмы составления условных алгоритмов в среде Scratch, использование основных блоков для составления условных алгоритмов в среде Scratch | Ознакомление с ос-  новами работы с ус-  ловными алгоритма-  ми в среде Scratch | 2 | Наблюдение за работой  учителя, самостоятельная работа со средой  Scratch, ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 13 | Циклические алгоритмы | Ознакомление с понятием  «циклический алгоритм», основные приёмы составления циклических алгоритмов в среде  Scratch, использование основных блоков для составления циклических алгоритмов в среде Scratch | Ознакомление с ос-  новами работы с  циклическими алгоритмами в среде  Scratc | 3 | Наблюдение за работой учителя, самостоятельная работа со средой  Scratch, ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 14 | Создание подпрграмм | Ознакомление с возможностью  создания подпрограмм в среде Scratch. Раздел Другие блоки, создание блока, параметры блок | Ознакомление с ос-  новами работы по  созданию блоков-подпрограмм в  среде Scratch | 3 | Наблюдение за работой учителя, самостоятельная работа со средой  Scratch, ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 15 | Блок команд  «Управление» | Изучение циклов и ветвлений в среде программирования  VEXcode VR | Ознакомить обучаю-  щихся с ветвлениями  циклами | 3 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 16 | Проекты «Разру-  шение замка» и  «Динамическое  разрушение замка» | Использование датчиков для улучшения алгоритма по очистке территории | Ознакомить обучаю-  щихся с выполнением проектов по уборке территории  на vr.vex.com | 2 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |
| 17 | Проект «Детектор  линии» | Подсчёт количества линий. Программа для поиска и подсчёта линий | Ознакомить обучаю-  щихся с выполнением проектов обнаржению линий | 2 | Слушание объяснений  учителя. Наблюдение за  работой учителя. Ответы на контрольные вопросы | Компьютер, проектор, интерактивная доска |

## Дидактическиематериалы

1. . Платформа программирования роботов VEXCode VR [Электронный ресурс] //

URL: [https://vr .vex .com](https://vr.vex.com/) (Дата обращения: 15 .04 .2021) .

1. . Информатика . Уровень1-Блоки [Электронный ресурс] //URL: [https://education .](https://education.vex.com/stemlabs/cs/computer-science-level-1-blocks)

[vex .com/stemlabs/cs/computer-science-level-1-blocks](https://education.vex.com/stemlabs/cs/computer-science-level-1-blocks) (Датаобращения: 15 .04 .2021) .

1. . Официальный сайт среды программирования Scratch [Электронный ресурс] //URL: [https://scratch .mit .edu/](https://scratch.mit.edu/) (Дата обращения: 15 .04 .2021) .
2. . STEM Education channel by Mark Johnston // URL: [https://www .mjstem .com/](https://www.mjstem.com/)(Датаобращения: 15 .04 .2021) .