



НТО ЮНИОРЫ 2026/27 · СФЕРА

Роботех

Быстрый старт: главное для подготовки к
отборочному этапу

5-7 класс



Автор-составитель

Артём Осинowskiй, методист и разработчик детского технопарка «Кванториум»

Детский технопарк «Кванториум», Новосибирск · 2026

Роботех

Быстрый старт: главное для подготовки к отборочному этапу НТО Юниоры. 5–7 класс

Автор-составитель – **Артём Осинowski**, методист и разработчик детского технопарка «Кванториум».

Нашли ошибку или неточность? Напишите на osinovsky.aa@kvantorium54.ru (рабочая почта): сфера, версия книги (1.0) и номер страницы – исправим в следующей версии. Актуальная версия и список изменений – nto.kvantorium54.ru/knigi. Личная почта для других вопросов – artem081033@gmail.com.

Важно. Это не официальное пособие олимпиады, а книга методиста «Кванториума». Каких заданий ждать на отборе 2026/27, мы не знаем: организаторы их заранее не показывают, а задания разных лет не повторяются. Книга готовит к темам из программы сферы и приёмам решения, а не к конкретным задачам.

Книга помогает школьнику самостоятельно подготовиться к отборочному этапу сферы «Роботех» Национальной технологической олимпиады «НТО Юниоры»: теория с нуля, подробные разборы задач и задачи для самопроверки с ответами.

Большинство задач составлены автором по образовательной программе сферы, часть – по мотивам открытых заданий прошлых лет; источник указан у каждой задачи. Уровни сложности – шкала этой книги, а не баллы олимпиады. Правила, сроки и формат отбора – на сайте junior.ntcontest.ru.

Это «Быстрый старт» – сжатая версия книги: по каждой главе главное, один разбор и четыре задачи с ответами. Для подготовки к отбору рекомендуем полную версию – все разборы, задачи и полный справочник, главы те же: nto.kvantorium54.ru/knigi.

Быстрый старт · версия 1.0 от 02.10.2026, сборка 02.10.2026. © Артём Осинowski, 2026.

Содержание

Как устроена эта книга	3
Для кого эта книга	3
Как устроен отбор	3
Что пригодится на отборе и где это в книге	4
Как пользоваться книгой	5
Как вводить ответы на отборе	6
Что понадобится	6
План на месяц до отбора	27
Родителям и наставникам	27
Честно об этой книге	28
Глава 1 Как устроен робот	29
Глава 2 Движение: скорость, путь, время	33
Глава 3 Координаты и углы: навигация робота	37
Глава 4 Алгоритмы и исполнители	42
Глава 5 Моторы, датчики и электричество	48
Глава 6 Программирование: Arduino и машины состояний	53
Глава 7 Движение по линии и регуляторы	58
Глава 8 Сигналы и обработка данных	63
Ответы и решения	68
Справочник	71
Шпаргалка: всё «Запомни» из глав	71

Как устроена эта книга

Эта книга – тренажёр для отборочного этапа сферы «Роботех» олимпиады «НТО Юниоры». В ней теория с нуля, подробные разборы задач и сразу после каждого разбора – похожие задачи «Реши сам» с ответами в конце книги. Свой робот и набор деталей не нужны. Скорость, повороты и маршруты считаются в тетради с калькулятором, а схемы и программы собираются в браузере, в том же симуляторе, что и на курсе сферы.

Для кого эта книга

- Для школьников **5–7 классов**, которые выбрали (или думают выбрать) сферу «Роботех».
- Для тех, кто **не проходил физику и ни разу не программировал**. Скорость, ток, напряжение, мотор и датчик объясняем с самого начала, а первую программу разбираем построчно.
- Для тех, у кого **нет кружка и конструктора**. В первом полугодии, которое как раз готовит к отбору и финалу, курс сферы идёт целиком в онлайн-симуляторах, а конструктор нужен только во втором. Книга устроена так же.
- Для **наставников и родителей**, которые хотят помочь. Для них есть отдельный раздел в конце введения.

Как устроен отбор

Таблица В.1

Когда	Что происходит
с 1 октября	демо-этап с пробными заданиями (по сообщению организаторов)
до 9 ноября	регистрация на олимпиаду (наш совет – успеть до 20 октября, тогда будут обе попытки)
20–30 октября	отбор, попытка 1 (3 часа)
31 октября – 10 ноября	отбор, попытка 2 (засчитывают лучшую из двух)
7–17 декабря	финалы в регионах

- Отбор проходит **дома, онлайн, с компьютера**. Задания открываются в личном кабинете на my.ntcontest.ru и решаются на платформе «Орбита» (orbita.education). Каждый решает сам за себя. Три часа идут подряд, остановить таймер нельзя.
- Задания – это задачи с ответом, который проверяет компьютер: ввести число, выбрать верные варианты, сопоставить, расставить команды по порядку, написать короткий код.

- Разработчик сферы на вебинарах августа и сентября 2026 года рассказал, из чего будет состоять отбор: задачи теоретические и задачи-практикумы в онлайн-симуляторе Tinkercad, где собирают схему и пишут программу. Практику из курса он советовал проходить руками, а не только читать. Поэтому в главах 5–8 много схем и программ, которые стоит повторить в симуляторе.
- Книга построена по программе сферы «Роботех. Основы мобильной робототехники» 2026/27. В самой программе есть отдельный модуль «Разбор задач отборочного онлайн-этапа» – значит, задачи отбора растут из тех же тем, что и курс.
- За тест бесплатного курса своей сферы дают **+5 баллов**. Курс – на academy.junior.online, вход через «Талант». Бонус нужно включить самому в личном кабинете («Мои привилегии») до конца отбора. Учти: после этого сферу уже не поменять. И бонус засчитают, только если за задания отбора ты набрал больше нуля.
- Тесты внутри курса – ещё не отбор. По словам разработчика, они несложные и нужны, чтобы ты видел, как продвигаешься по курсу и правильно ли понял модуль.
- Финал – **командный** и очный: участники приезжают на площадку в своём регионе, там их делят на команды по три человека, и команды решают задачи за компьютерами. Задачи снова теоретические и в симуляторе: спроектировать робота, ввести ему координаты, обработать и отфильтровать данные с датчиков, придумать алгоритм движения. Проходной балл в финал заранее не известен: его объявляют ближе к концу отбора.
- В некоторых регионах после финала отдельно проводят хакатон с настоящим конструктором «Роботех Юниор» на платформе Arduino. На баллы финала он не влияет, а программы там пишут те же, что в симуляторе.

Совет. Сроки и правила могут уточняться. Главный источник – сайт олимпиады junior.ntcontest.ru и памятка для Новосибирской области nto.kvantorium54.ru. Свежие новости и ответы разработчика – в публичном канале сферы «Роботех» в мессенджере MAX: он находится поиском по словам «сфера Роботех».

Что пригодится на отборе и где это в книге

На странице сферы перечислено, что пригодится на отборе, по трём предметам. Вот где это в книге:

Таблица В.2

Что пригодится на отборе	Глава
Физика: механика и механизмы	1. Как устроен робот
Математика: расстояние, время, скорость; величины и единицы измерения	2. Движение: скорость, путь, время
Физика: параметры движения, сила трения, инерция	2
Математика: координаты, углы и фигуры, пропорции и масштаб	3. Координаты и углы: навигация робота
Информатика: алгоритмы и исполнители	4. Алгоритмы и исполнители

Что пригодится на отборе	Глава
Физика: электричество и электрические цепи, звук	5. Моторы, датчики и электричество
Информатика: язык и среда программирования, переменные, условия, циклы, подпрограммы	6. Программирование: Arduino и машины состояний
Движение по линии, П- и ПИД-регулятор (из программы сферы)	7. Движение по линии и регуляторы
Математика: графики; обработка сигналов с датчиков (из программы сферы)	8. Сигналы и обработка данных

Таблицы встречаются по всей книге, а мини-шпаргалка по табличному редактору – в справочнике.

Начни с главы 1: в ней видно, из чего собран робот, и дальше книга всё время на это опирается. Главы 2 и 3 – про математику движения: их хватит тетради, карандаша и калькулятора. Третью читай после второй, потому что повороты мы считаем через путь колеса. Главу 4 можно открыть когда угодно. Главу 5 прочитай раньше главы 6: сначала понятно, как мотор и датчик подключены, потом – как ими командовать из программы. Главы 7 и 8 – после шестой: в них датчики, моторы и программа работают вместе.

Как пользоваться книгой

В каждой главе теория и разборы идут вперемешку: за каждым разбором сразу стоят две похожие задачи «Реши сам», чтобы закрепить приём, пока он свежий в голове.

- 1. Теория.** Коротко и по делу, с рисунками. Каждая новая тема начинается с большой картинкой «на пальцах», а самое нужное собрано в рамках «Запомни». Все «Запомни» из всех глав повторяются в конце книги, в шпаргалке.
- 2. Разбираем задачи.** К каждой задаче – решение по шагам: что дано, что делаем и почему, где часто ошибаются, какой ответ получился и можно ли ему верить.
- 3. Реши сам.** Сразу после разбора – две задачи на тот же приём, но с другими числами и другим сюжетом.
- 4. Ответы** – в конце книги, в разделе «Ответы и решения». Там только ответ и иногда короткая подсказка, если задача с подвохом. Полного решения нет специально: если подсказки не хватило, вернись к разбору перед задачей.

Не готов читать всю главу целиком? Есть отдельная короткая книжка «Быстрый старт»: из каждой главы самое нужное, один разбор и несколько задач «Реши сам». Её удобно пройти за неделю, а эту книгу держать под рукой.

У каждой задачи есть уровень – как в обычном задачнике: у трудных задач звёздочка рядом с номером («задача 3*»), в них нужно несколько идей сразу. Лёгкие задачи, в одно действие, подписаны словом «разминка». Это уровни нашей книги, а не баллы олимпиады. Сколько задач будет на отборе и сколько стоит каждая, заранее не объявляют.

Совет. Разбор читай с листком бумаги. Закрой решение, попробуй решить сам, а потом сверяйся по шагам. Если в разборе есть схема или программа, собери её в симуляторе и запусти. Робот, который у тебя на экране поехал не туда, научит большему, чем три страницы правильного текста.

У каждой задачи внизу написано, откуда она взялась. «Своя задача по программе сферы» – мы придумали её по теме из программы 2026/27 года; таких в книге большинство. «По мотивам задачи» – идею мы взяли из задач прошлых лет или из задач старшей олимпиады для 8–11 классов по профилю «Интеллектуальные робототехнические системы», упростили, пересказали своими словами и поменяли числа.

Как вводить ответы на отборе

Так было на отборочных этапах прошлых лет в других сферах. На «Орбите» перед каждой задачей читай инструкцию: там сказано, в каком виде нужен ответ.

- **Только число**, без единиц измерения и без знака градуса. Смотри, в чём просят ответ: расстояние – в сантиметрах или метрах, время – в секундах, миллисекундах или микросекундах, угол – в градусах.
- **Дробную часть** на сайте обычно отделяют **точкой**: `0.75`, а не `0,75`. В книге мы пишем по-русски, через запятую.
- **Несколько верных вариантов** часто записывают строкой из их номеров подряд, без пробелов и запятых: выбрал варианты 1, 3 и 4 – вводишь `134`. Точный формат – в инструкции к задаче.
- **Округляй так, как сказано в условии.** «До десятых» – один знак после точки. Промежуточные результаты не округляй – только ответ.
- **Считай, что можно сосчитать поштучно.** Перекрёстки, шаги исполнителя, показания `analogRead` – целые числа. Если робот проехал 7,4 перекрёстка, ошибка где-то раньше. А вот обороты колеса бывают и дробными: половина оборота – обычный ответ.
- **У каждого могут быть свои числа.** В некоторых задачах условие генерируется для каждого участника отдельно, так что списать ответ не выйдет.
- **Попытки ограничены.** На задачу даётся от 3 до 5 вводов, и каждая неверная попытка снимает 30 % балла за эту задачу. Сначала реши на бумаге или проверь программу в симуляторе, и только потом вводи ответ.
- **«Сохранить» – ещё не отправка.** Ответ засчитан, когда нажата кнопка «Отправить».

Что понадобится

- Тетрадь в клетку, карандаш, линейка и транспортир – маршруты, координаты и углы удобно рисовать прямо по клеткам.
- Калькулятор. Хватит обычного на телефоне, но пусть в нём будут кнопки π и корня.
- Компьютер с интернетом и мышью – для глав 5–8. Планшета мало: схему собирают, протягивая провода мышкой.
- **Tinkercad** – онлайн-симулятор, в котором работают курс, отбор и финал сферы. Раздел «Цепи» (по-английски Circuits): собираешь схему из платы Arduino Uno, моторов, датчи-

ков и проводов, пишешь программу и смотришь, как всё работает. Личная учётная запись не нужна: в курсе сферы есть ссылка на класс «Роботех» и пошаговая инструкция, как в него войти. Заходи именно по ссылке на класс: личные учётные записи на вебинаре называли ограниченными.

- **Wokwi** (wokwi.com) – запасной симулятор Arduino в браузере; если у тебя он не открывается, хватит Tinkercad. На бесплатной учётной записи запуск иногда ждёт в очереди.
- **Кибериада IDE** (Cyberiada IDE) – редактор, где программу не пишут строчками, а рисуют схемой из состояний и переходов (язык ПРИМС, глава 6). Ссылка на неё – в первых модулях курса.
- **Arduino IDE** – по желанию. Разработчик сферы сказал, что для курса она не обязательна: код можно писать прямо в симуляторе. Но в ней удобно набирать программу с подсветкой, а потом копировать в Tinkercad. Для подготовки к отбору хватит симулятора; программу для настоящей платы поможет поставить педагог кружка.
- **Таблицы** – Excel, LibreOffice Calc или любые онлайн-таблицы: для задач, где данные с датчиков нужно посчитать и построить по ним график.

Эти же инструменты – онлайн-симулятор, Кибериада IDE, Arduino IDE и таблицы – названы в официальных требованиях к площадкам финала сферы «Роботех».

Tinkercad: вход в класс и первая схема

Tinkercad (tinkercad.com) – бесплатный симулятор электронных схем и Arduino прямо в браузере: ставить ничего не нужно. В нём идёт практика курса сферы. Регистрироваться тоже не надо – курс даёт ссылку на класс и общее имя ученика. Ниже весь путь по шагам, рамка с номером на картинке – куда нажимать.

Шаг 1. Войти в класс. Открой ссылку на класс курса: tinkercad.com/joinclass/YMRYDIZP (класс «Роботех»; та же ссылка – в курсе сферы, модуль 5 «Теория и практика по мобильной робототехнике. Часть 1»). Нажми «Присоединиться, используя имя учащегося» (1 на левой половине рисунка). Впиши имя учащегося `test` (1 на правой половине) и нажми «Перейти в класс» (2).

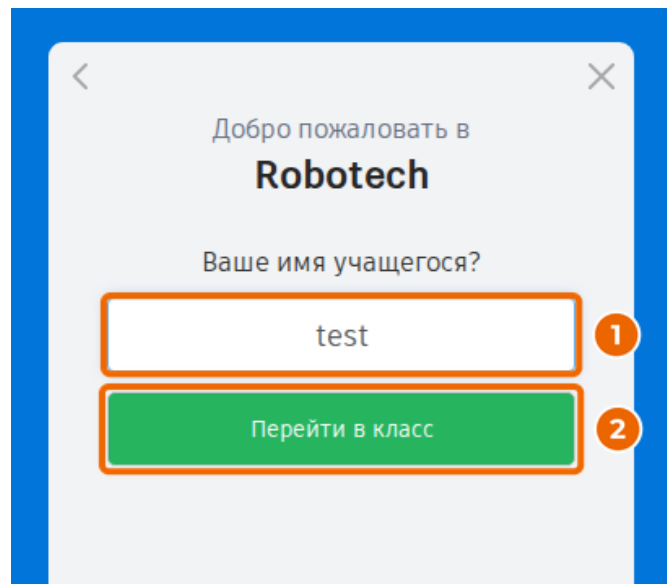
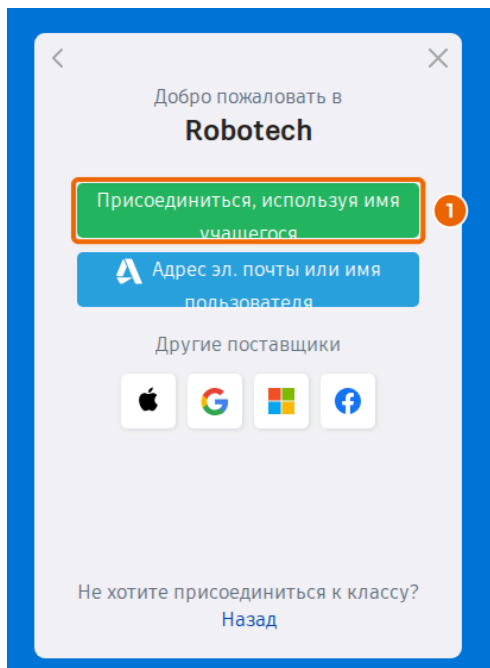


Рисунок В.1 – Вход в класс Tinkercad: слева – выбор способа входа, справа – имя учащегося

Шаг 2. Новая схема. Вверху справа нажми «Создать» (1), в меню – «Цепи» (2). Откроется редактор схем.

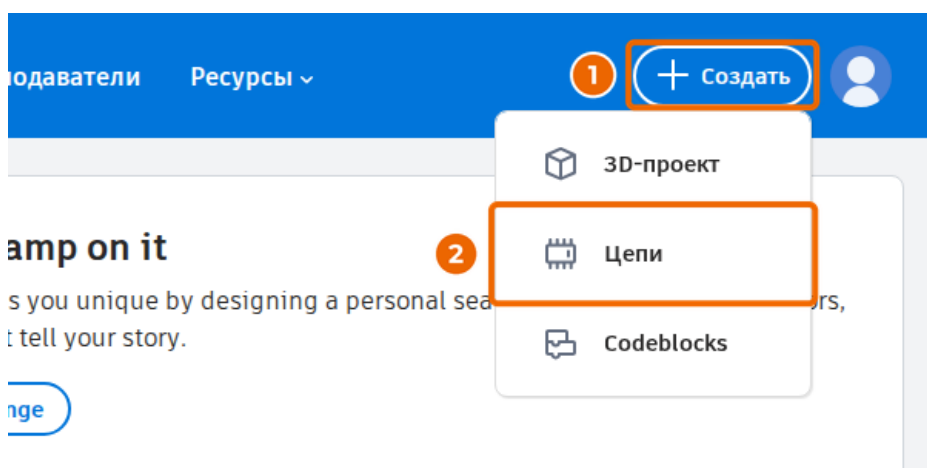


Рисунок В.2 – Новая схема: «Создать» → «Цепи»

Шаг 3. Окно редактора. Вот что в нём где.

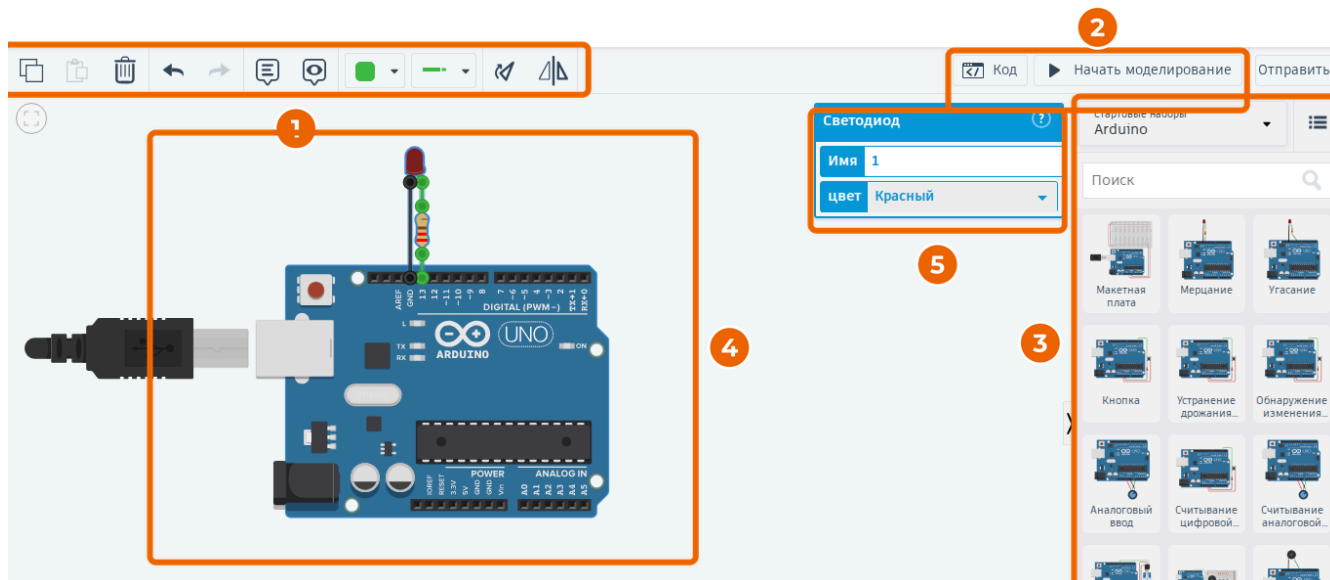


Рисунок В.3 – Редактор схем Tinkercad

1. **Инструменты:** копировать, вставить, удалить, отменить и вернуть действие, повернуть деталь.
2. **«Код»** – программа для Arduino; **«Начать моделирование»** – запуск схемы.
3. **Компоненты:** сверху выбор набора, ниже поиск и сами детали. Деталь перетаскивают мышью на поле.
4. **Рабочее поле** – здесь собирают схему. Провод тянется так: щёлкни по выводу одной детали, потом по выводу другой.
5. **Свойства выбранной детали** – например, цвет светодиода или сопротивление резистора.

Шаг 4. Готовая схема для старта. В списке компонентов выбери «Стартовые наборы» → «Arduino» (1 на левой половине) и перетащи на поле «Мерцание» (1 на правой половине): это Arduino со светодиодом, резистором и уже написанной программой. Рядом лежат и другие готовые схемы – кнопка, сервопривод, ультразвуковой дальномер.

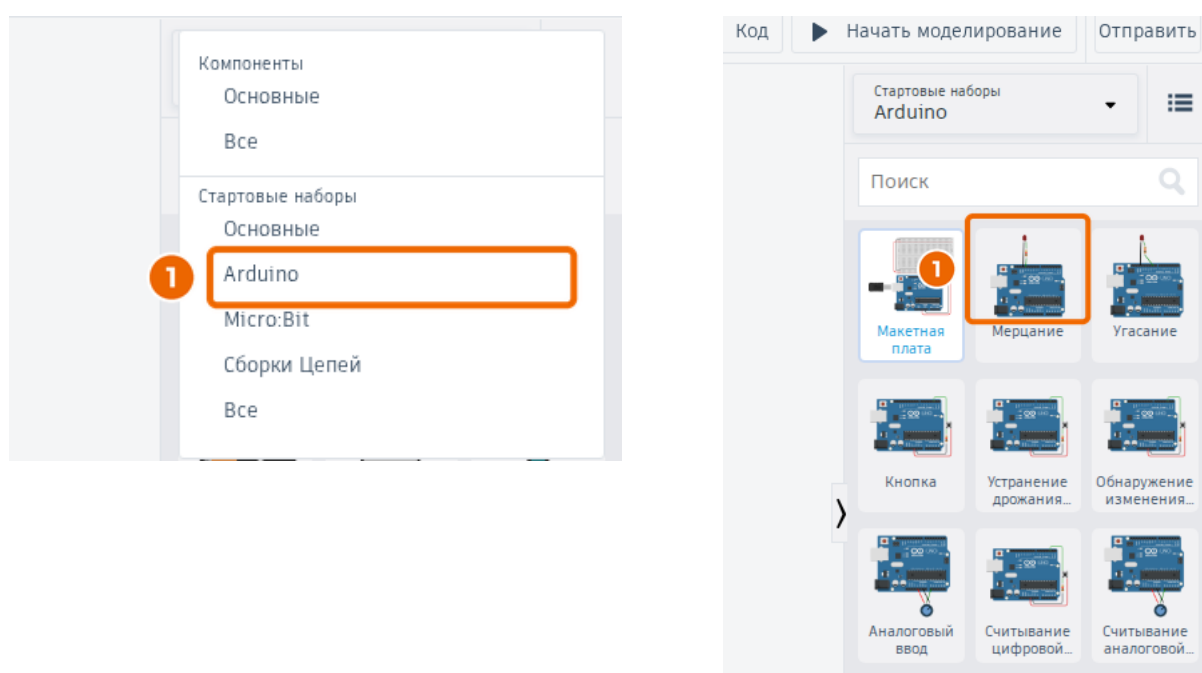


Рисунок В.4 – Стартовые наборы: слева – выбор набора «Arduino», справа – схема «Мерцание»

Шаг 5. Код и запуск. Кнопка «Код» (1) открывает программу. Её можно смотреть блоками или текстом на языке Arduino: переключатель «Блоки» вверху окна кода. Кнопка «Начать моделирование» (2) запускает схему.

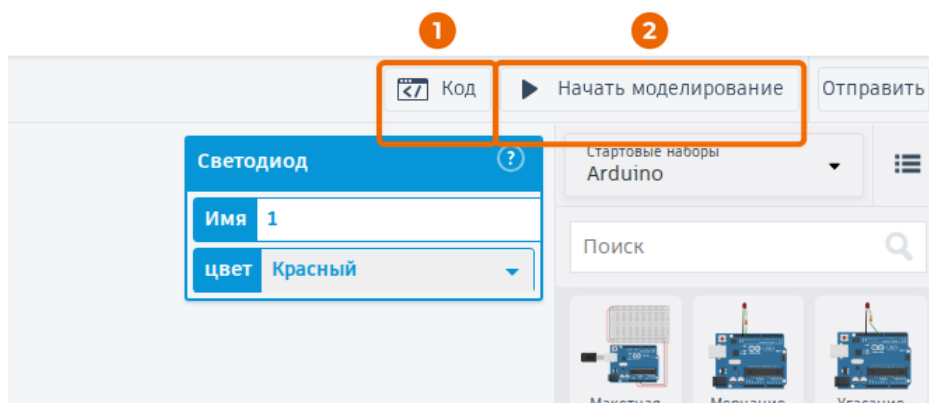


Рисунок В.5 – Кнопки «Код» и «Начать моделирование»

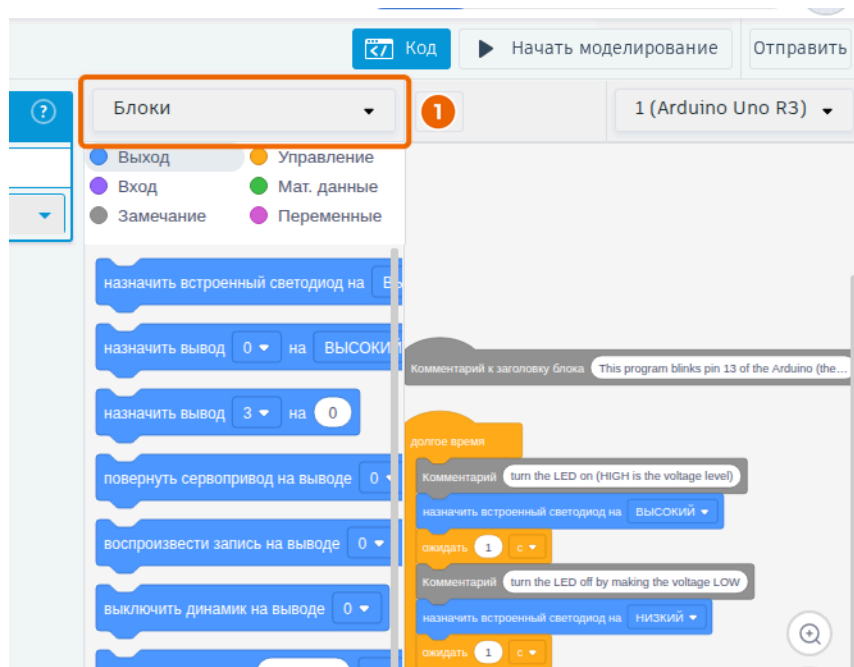


Рисунок В.6 – Программа «Мерцания» блоками

Во время моделирования кнопка называется «Остановить моделирование» (1), а светодиод на плате мигает (2): секунду горит, секунду нет. Остановить – той же кнопкой.

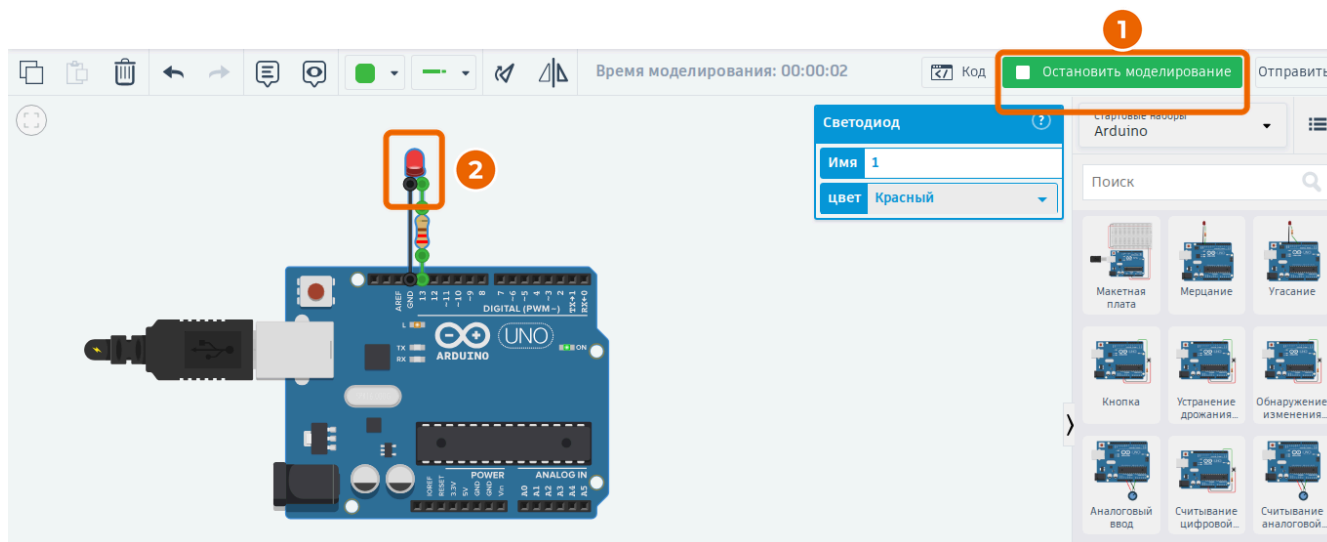


Рисунок В.7 – Схема работает

Важно про имя test . Под ним одновременно сидят все ученики курса: твою схему видят и могут поменять другие, а сохранение не гарантируется. Не пиши туда ничего личного. Если хочешь хранить свои работы, заведите вместе со взрослым свой аккаунт Autodesk – вход «Адрес эл. почты или имя пользователя» на том же экране, где выбирают способ входа.

Кибериада IDE: установка и первая программа

Кибериада IDE – бесплатная отечественная среда, в которой программу не пишут строчками, а рисуют диаграммой на языке ПРИМС: режимы работы – прямоугольники-состояния, смена режима – стрелки-переходы. Ниже весь путь от установки до первой программы «мигалка» – задача из курса сферы. Рамка с номером на картинке – куда нажимать.

Шаг 1. Скачать и поставить. Открой method.insitulab.pro/cyberiaда и нажми «Скачать программу». Выбери файл под свою систему: .exe – Windows, .dmg – macOS, .deb или .AppImage – Linux. Запусти его. Установщик попросит права администратора – он ставит драйверы для плат, так и должно быть; дома нажимай «Да» вместе со взрослым. После установки программа откроется сама.

Шаг 2. Новый документ. Слева – панель «Документ», в ней «Создать...» (1).



Рисунок В.8 – Стартовый экран Кибериады: «Документ» → «Создать...»

Шаг 3. Платформа. Кибериада умеет работать с разными платами, поэтому сначала выбирают, для какой пишем. Дважды щёлкни «Arduino Uno» в списке платформ (1) – она появится в «Выбрано» (2). Нажми «Создать» (3).

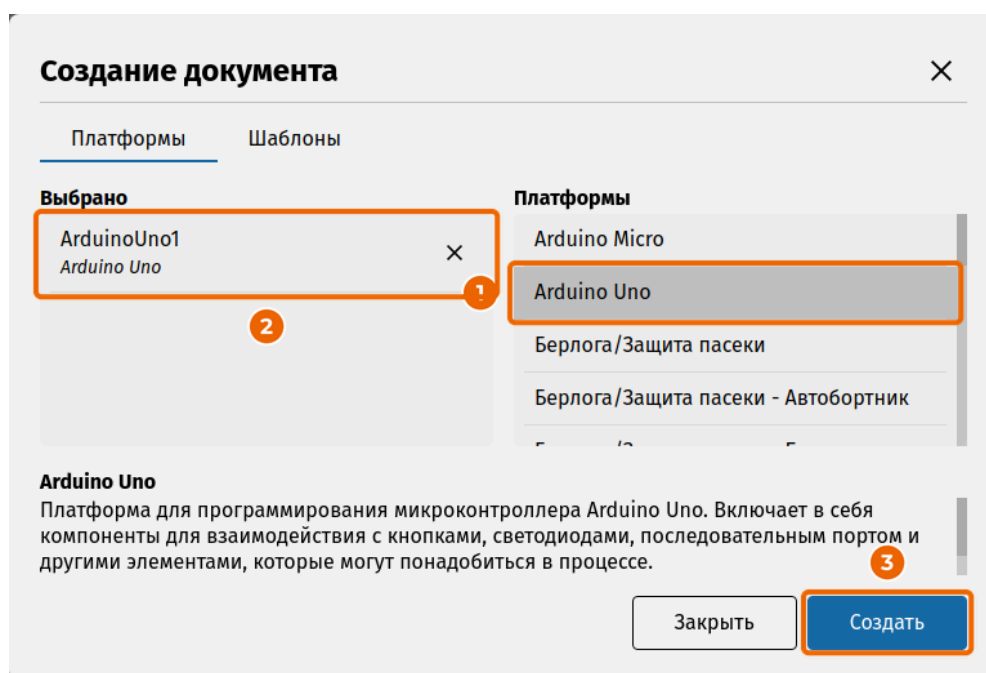


Рисунок В.9 – Выбор платформы для документа

Шаг 4. Компоненты. Компонент – это то, чем программа управляет или от чего ждёт сигнала: светодиод, кнопка, таймер. Нажми «+» у «Компонентов» (1).

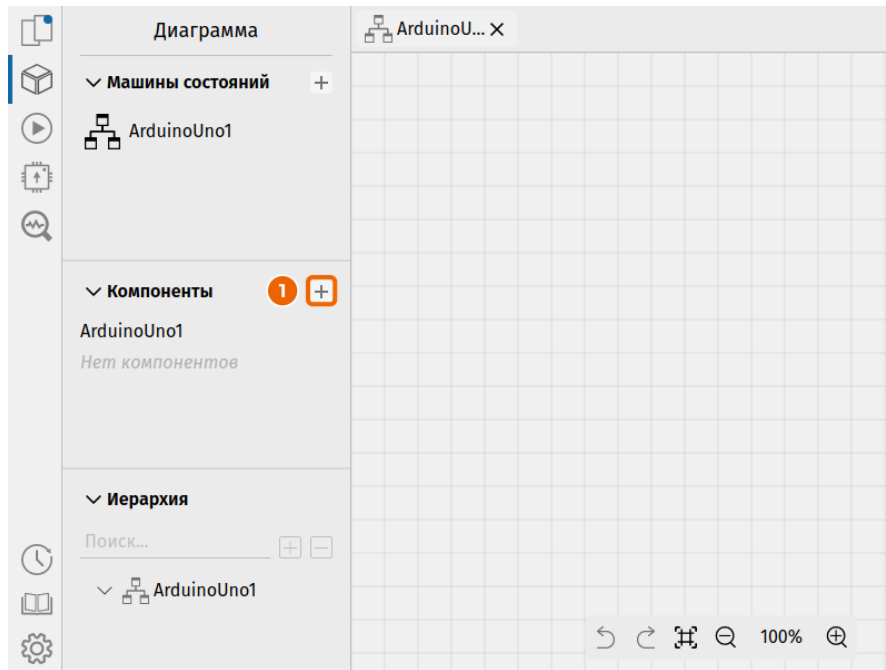


Рисунок В.10 – Окно диаграммы: плюс у «Компонентов»

Выбери «Светодиод» (1 на левой половине) и нажми «Добавить» (2). В форме впиши название «Светодиод» (1 на правой половине) и выбери пин (2) – номер вывода Arduino, к которому подключён светодиод; на занятии курса это 2. Нажми «Применить» (3). Так же добавь «Таймер» – у него пина нет, только название «Таймер».

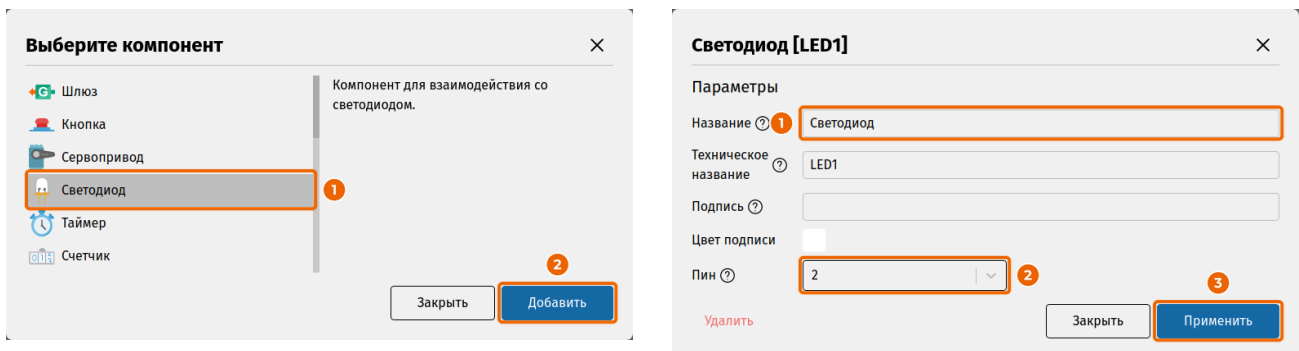


Рисунок В.11 – Новый компонент: слева – выбор «Светодиода», справа – его название и пин

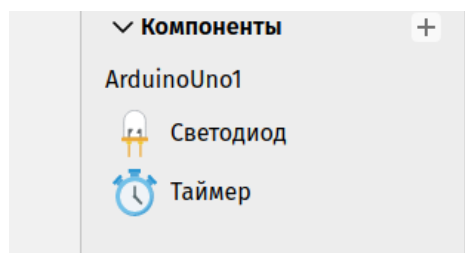


Рисунок В.12 – Оба компонента в списке

Шаг 5. Состояние «Не горит». Дважды щёлкни по пустому месту поля – появится прямоугольник «Состояние». К нему от синего кружка ведёт стрелка: отсюда программа начинает работу. Нажми карандаш (1) и впиши «Не горит», затем Enter.

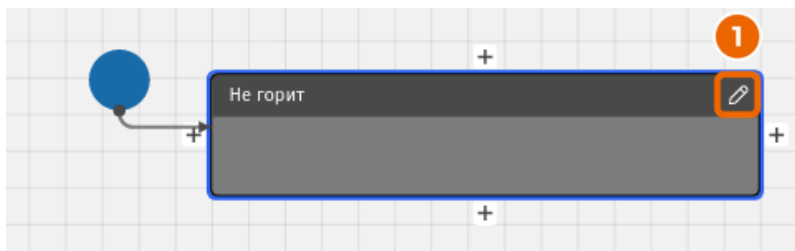


Рисунок В.13 – Новое состояние: карандаш – переименовать

Теперь скажем, что делать в этом состоянии. Дважды щёлкни по нему – откроется «Редактор состояния». Нажми «+» (1).

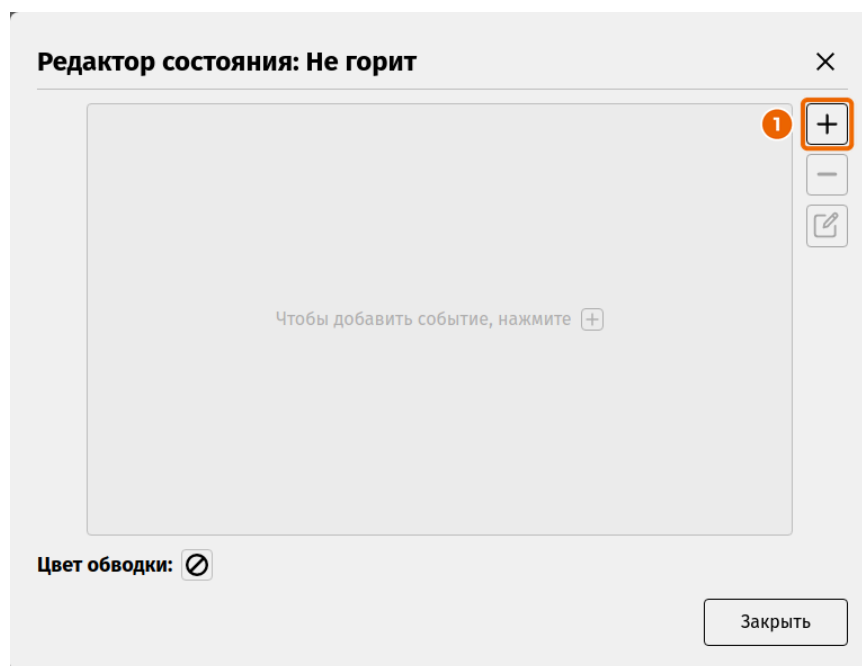


Рисунок В.14 – Редактор состояния

В «Редакторе события» в строке «Когда» уже стоит «Общие → Вход» (1): действия выполняются, как только программа войдёт в это состояние. Так и оставь. Нажми «+» у «Делай» (2).

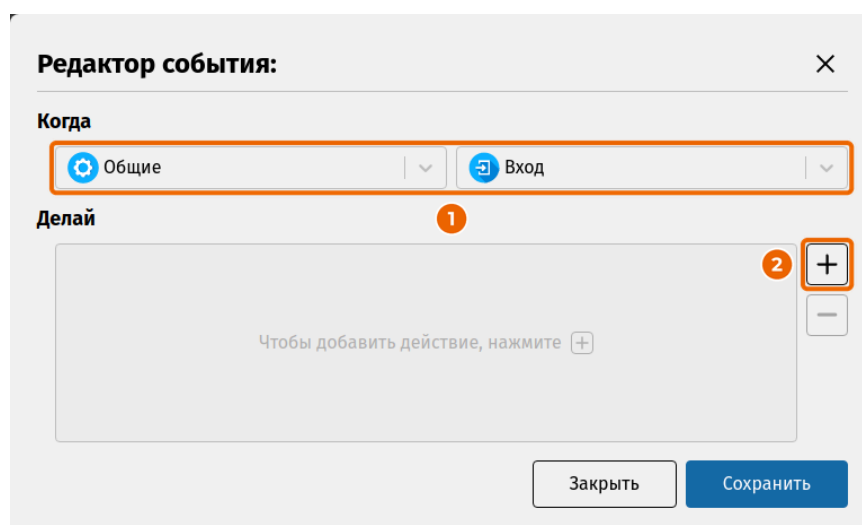


Рисунок В.15 – Редактор события: «Вход» и плюс у «Делай»

Выбери компонент «Светодиод» (1) и действие «Погасить» (2), нажми «Сохранить».

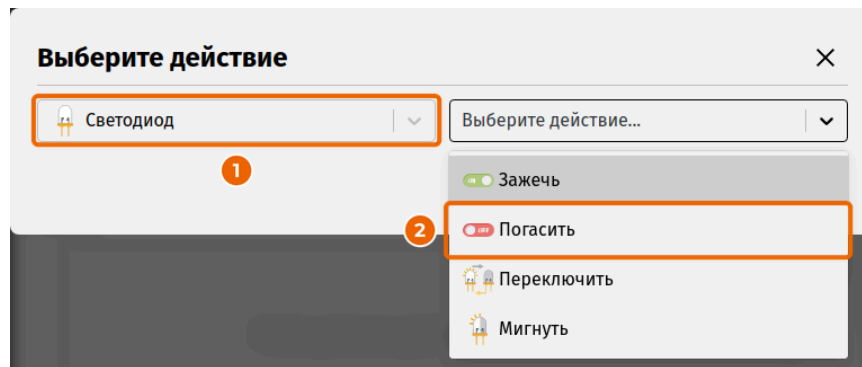


Рисунок В.16 – Действие «Светодиод – Погасить»

Ещё раз «+» у «Делай»: компонент «Таймер» (1), действие «Старт» (2), в поле интервала – 1000 (3), это миллисекунды, то есть одна секунда. Нажми «Сохранить» (4). Переключатель «Интервал» не трогай: он нужен, чтобы брать число из другого компонента, а нам хватит простого числа.



Рисунок В.17 – Действие «Таймер – Старт» на 1000 мс

В «Делай» теперь два действия (1). Нажми «Сохранить» (2), потом «Заккрыть» в редакторе состояния.

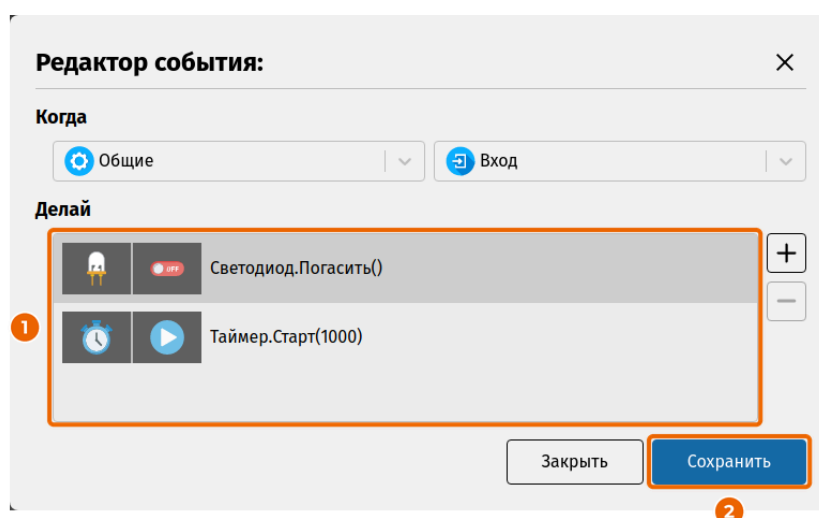


Рисунок В.18 – Событие «Вход» с двумя действиями

Шаг 6. Состояние «Горит». Так же: двойной щелчок по пустому месту, имя «Горит», одно действие – «Светодиод → Зажечь». Таймер второй раз запускать не нужно: на платформе Arduino Uno таймер повторяющийся – запустили один раз, и он срабатывает каждую секунду. На других платформах бывает иначе – смотри описание компонента.

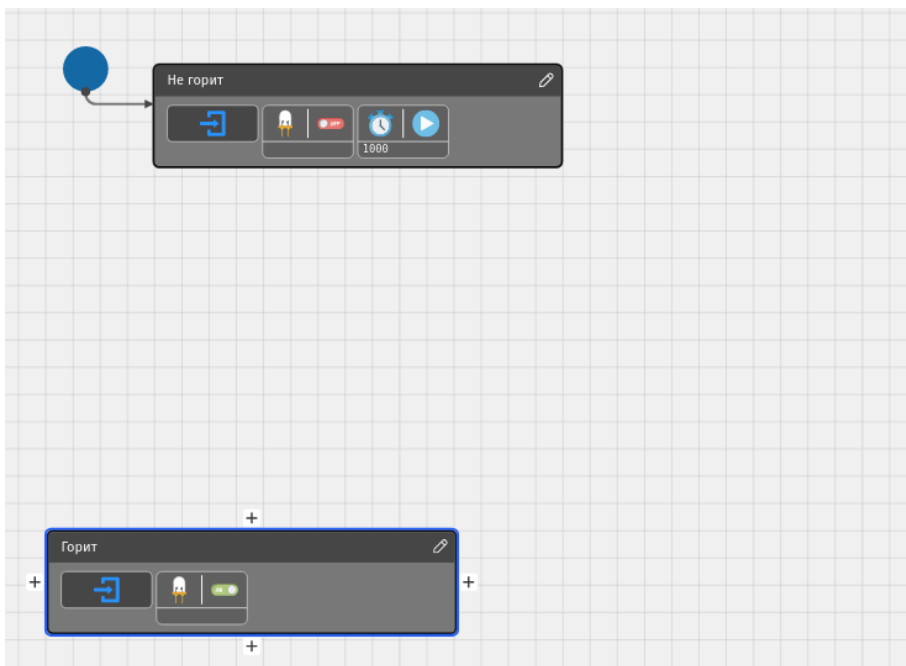


Рисунок В.19 – Два состояния готовы: на каждом видно, что оно делает при входе

Шаг 7. Переходы. Щёлкни по «Не горит» – вокруг появятся плюсики. Потяни мышью нижний плюстик (1) до состояния «Горит» (2) и отпусти.

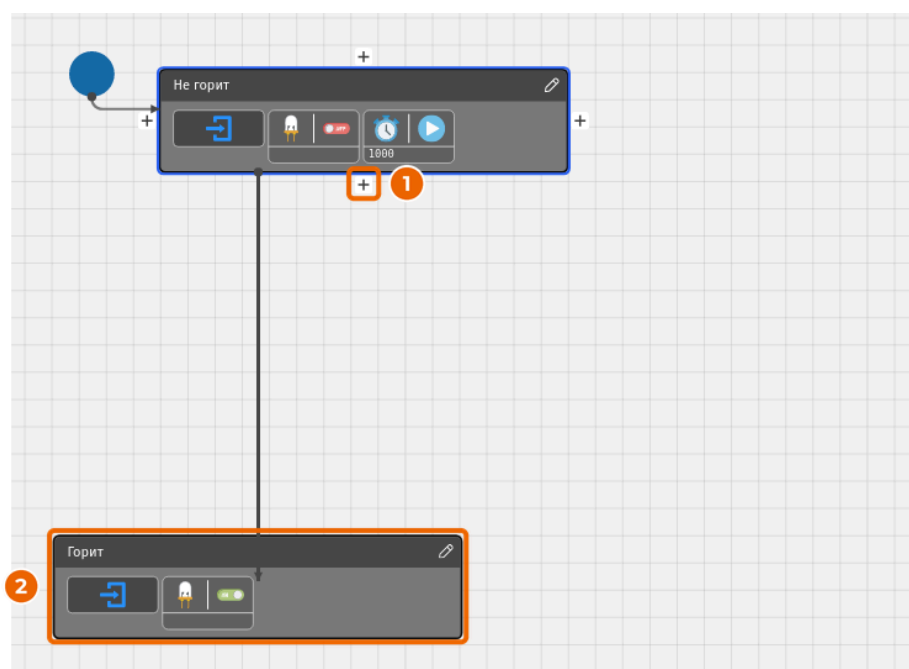


Рисунок В.20 – Тянем стрелку перехода

Откроется «Редактор соединения». В «Когда» выбери «Таймер» (1) и «Тайм-аут» (2) – переход случится, когда таймер отсчитает секунду. Условия и действия не нужны. Нажми «Сохранить» (3).

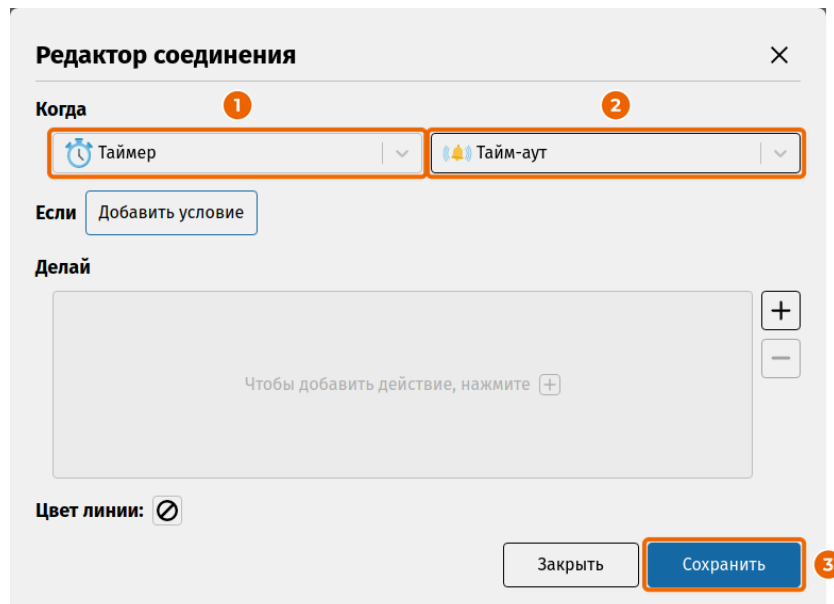


Рисунок В.21 – Переход по тайм-ауту таймера

Так же протяни стрелку обратно – из «Горит» в «Не горит», с тем же событием. Если две стрелки легли одна на другую, перетащи значок перехода мышью в сторону.

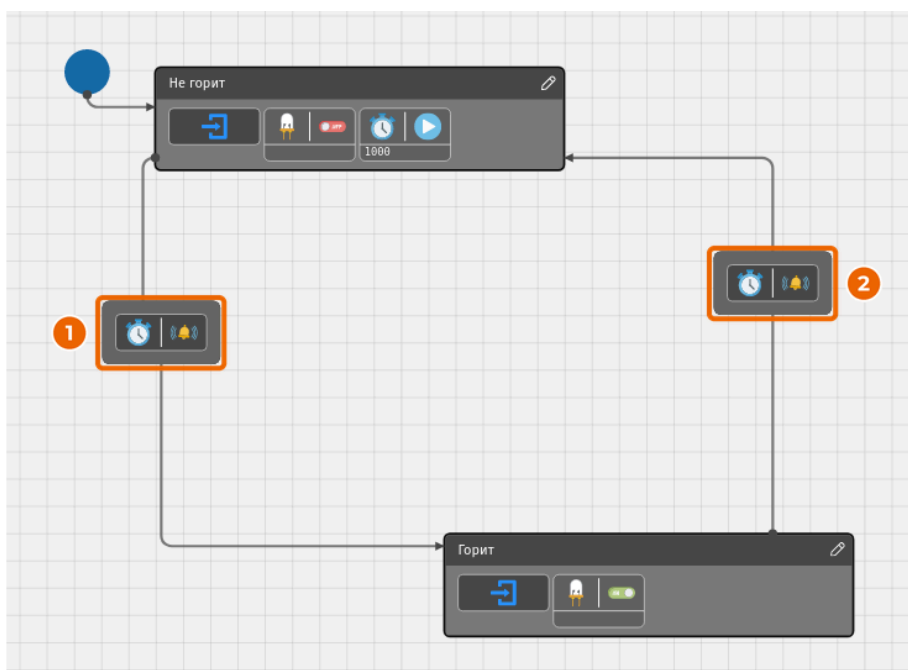


Рисунок В.22 – Готовая «мигалка»: переход «Не горит → Горит» (1) и обратный (2)

Прочитай программу по стрелкам: старт → «Не горит» (погасить, запустить таймер) → через секунду тайм-аут → «Горит» (зажечь) → тайм-аут → «Не горит» – и так по кругу. Светодиод мигает раз в секунду. Сохрани работу: «Документ → Сохранить как...», программа сохранится файлом `.graphml`.

Карта значков слева. По ним переключают панели.

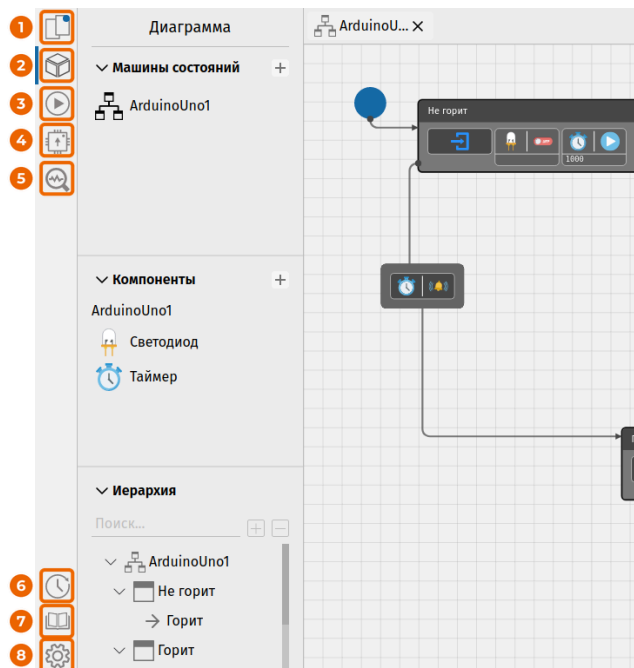


Рисунок В.23 – Значки левой панели

1. **Документ** – создать, открыть, сохранить.
2. **Диаграмма** – машины состояний, компоненты, иерархия состояний.
3. **Компилятор** – превращает диаграмму в прошивку для платы.
4. **Загрузчик** – записывает прошивку в плату через USB.
5. **Монитор порта** – показывает, что плата присылает по USB.
6. **История изменений** – что и когда менялось в документе.
7. **Документация** – справка (её же открывает F1).
8. **Настройки**.

Шаг 8. Компиляция и прошивка – когда есть плата. Открой «Компилятор» (3 на карте) и нажми большую синюю кнопку (1) – «Скомпилировать». Кружок рядом с заголовком (2) показывает связь с компилятором: если он красный, кнопка называется «Переподключиться» – нажми её и дождись, пока кружок станет зелёным.

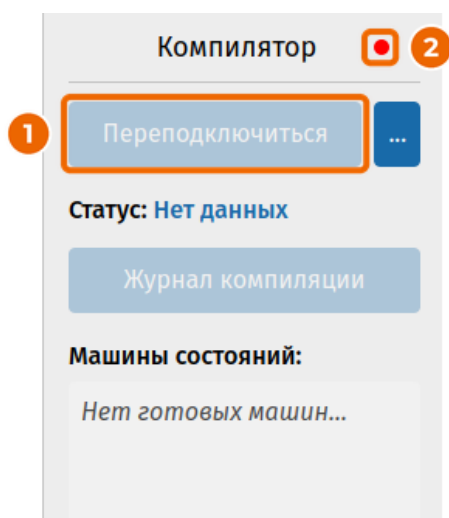


Рисунок В.24 – Панель «Компилятор»

Подключи Arduino к компьютеру кабелем и открой «Загрузчик» (4 на карте). Нажми «Подключить плату» (1), выбери свою плату в списке и нажми синюю кнопку с изображением платы – «Прошить». Прошивка готова, когда в «Журнале действий» появится подтверждение.

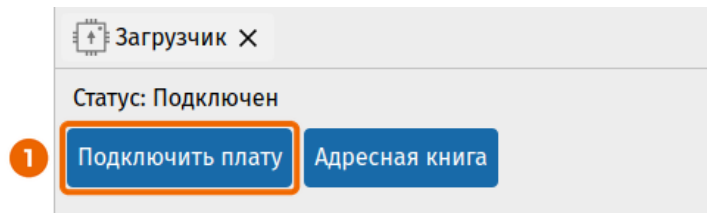


Рисунок В.25 – Панель «Загрузчик»

Платы под рукой нет – диаграмму всё равно можно проверить: пройди пальцем по стрелкам, как в разборах этой книги, и посмотри, в каком состоянии окажется программа после каждого события.

Wokwi: симулятор Arduino в браузере

Wokwi (wokwi.com) – бесплатный симулятор Arduino прямо в браузере. Ставить ничего не нужно, регистрироваться тоже: плату, светодиоды, кнопки и датчики «собирают» на экране проводами, а программу пишут тут же и запускают кнопкой. Ничего не сгорит, даже если ошибёшься. Если на твоём компьютере сайт не открывается, все схемы этой книги собираются и в Tinkercad. Интерфейс английский: названия кнопок ниже даны в кавычках по-английски, с переводом. Рамка с номером на картинке – куда нажимать. Зачем в схеме резистор – в главе об электричестве и электронике, что делают `setup()`, `loop()` и `Serial` – в главе про программирование Arduino. А здесь – как работать в самой программе и три задачи, чтобы руки привыкли.

Шаг 1. Открыть. Самый короткий путь – ссылка wokwi.com/projects/new/arduino-uno: сразу откроется чистый проект с платой Arduino Uno. То же самое кнопками: на главной странице wokwi.com выбери «Arduino (Uno, Mega, Nano)» (1 на левой половине рисунка), а на открывшейся странице в блоке «Start from Scratch» («Начать с нуля») – «Arduino Uno» (1 на правой половине).

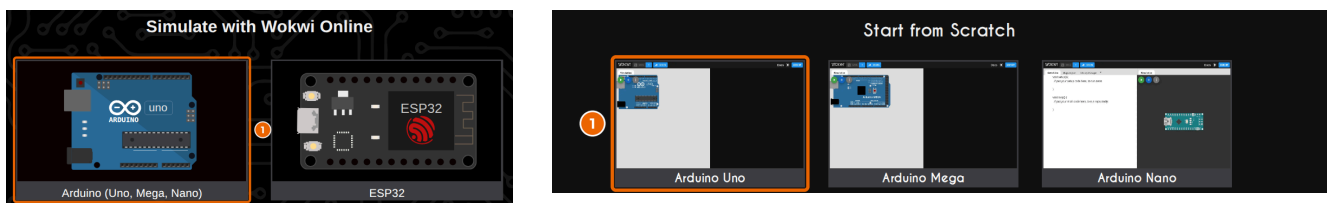


Рисунок В.26 – Как открыть чистый проект: слева – главная страница, справа – выбор платы

Входить в аккаунт не нужно: кнопки «Sign in» и «Sign up» (вход и регистрация) вверху справа не нажимай. Зато у работы без входа есть два неудобства. Первое: проект нигде не хранится. Обновишь страницу или закроешь вкладку – всё пропадёт, так что перед закрытием скачай проект (шаг 7). Второе: перед запуском Wokwi собирает программу на своём сервере. Обычно это доли секунды, иногда до минуты. Подожди, это нормально.

Шаг 2. Окно Wokwi. Слева – программа, справа – плата и схема.

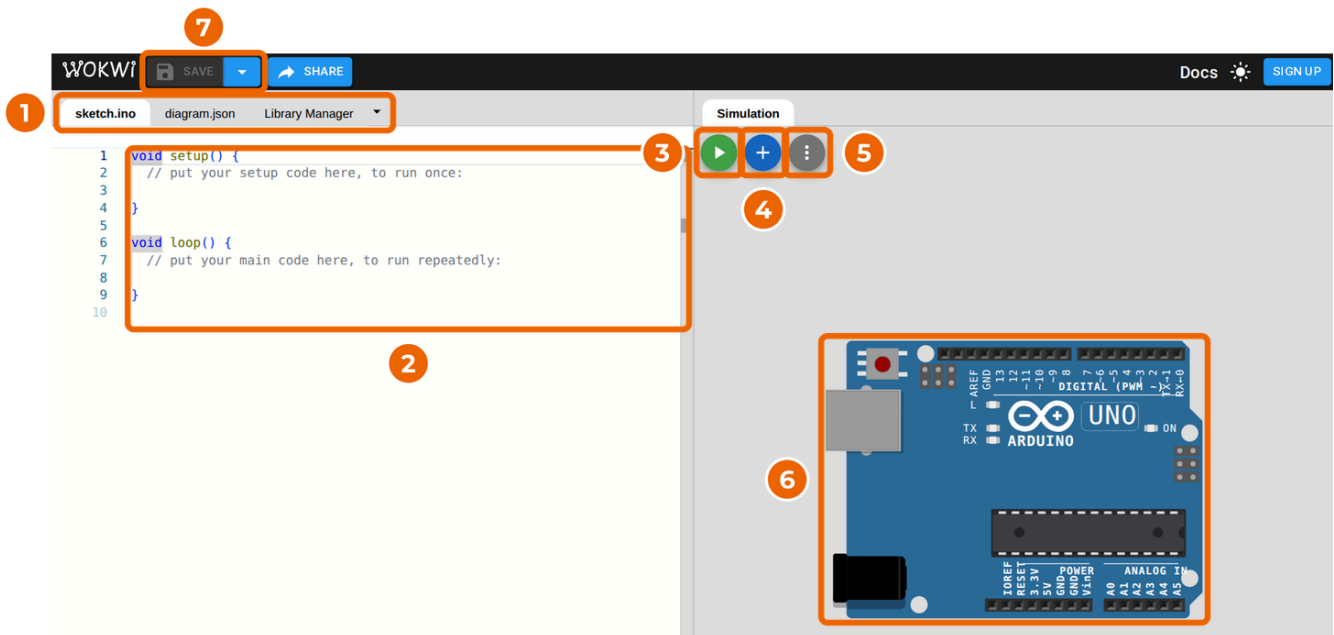


Рисунок В.27 – Окно Wokwi

1. **Вкладки.** «sketch.ino» – твоя программа. «diagram.json» – та же схема, записанная текстом: Wokwi пишет её сам, и в эту вкладку можно вставить готовую схему. «Library Manager» – подключаемые библиотеки, нам не понадобится.
2. **Окно программы.** Сразу лежат две обязательные функции, `setup()` и `loop()`.
3. **«Start the simulation»** («запустить симуляцию») – зелёная кнопка.
4. **«Add a new part»** («добавить деталь») – синяя кнопка «+».
5. **Меню «Zoom, grid, and more»** (три точки). «Undo» – отменить (Ctrl + Z), «Zoom in» и «Zoom out» – приблизить и отдалить, «Fit» – вернуть схеме размер по окну, «Toggle Grid» – показать сетку. Меню само не закрывается: закрой его клавишей Esc.
6. **Плата Arduino Uno.** Ножки подписаны: цифровые 0–13 – сверху, а питание (5V, GND – «земля») и аналоговые A0–A5 – снизу.
7. **«Save»** («сохранить») работает только со входом, поэтому нам нужна стрелка рядом с ним: в её меню лежит «Download project ZIP» (шаг 7).

Шаг 3. Добавить деталь. Нажми «+» (4 на карте окна). Откроется список деталей. В разделе «Basic» («основные») лежат LED – светодиод, Pushbutton и Pushbutton 6mm – кнопки (по-больше и поменьше), Resistor – резистор (2). Нужной детали на виду нет? Впиши её английское название в строку поиска (1), например `potentiometer`. Щёлкни по детали – она появится в левом верхнем углу поля. Нажми на неё и перетащи мышью на свободное место рядом с платой.

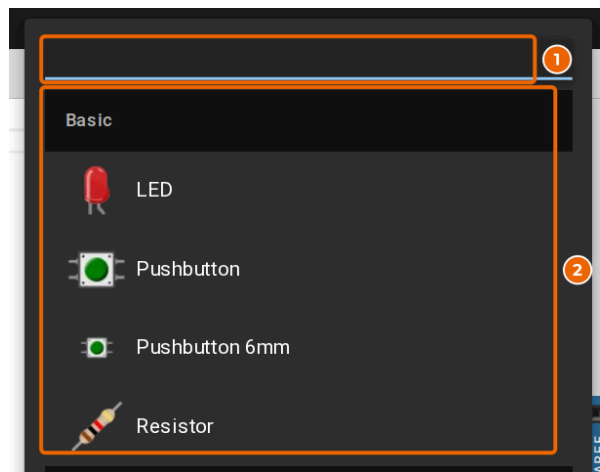


Рисунок В.28 – Список деталей: строка поиска (1) и раздел «Basic» (2)

Шаг 4. Настроить и повернуть. Щёлкни по детали – над полем появится её панель. У резистора в ней поле «Resistance» («сопротивление», 1), кнопка «Rotate» («повернуть», 2) и корзина «Delete» («удалить», 3). Впиши в поле 220 и нажми Enter: полоски на резисторе перекасятся. В поле число – это омы, а буква к – «кило»: 1k – это 1000 Ом.

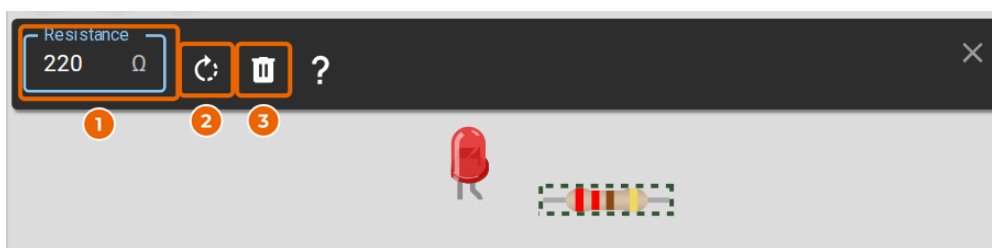


Рисунок В.29 – Панель резистора: «Resistance» (1), «Rotate» (2), «Delete» (3)

Деталь поворачивают кнопкой «Rotate» или клавишей R: одно нажатие – четверть оборота. Удалить деталь можно корзиной или клавишей Delete.

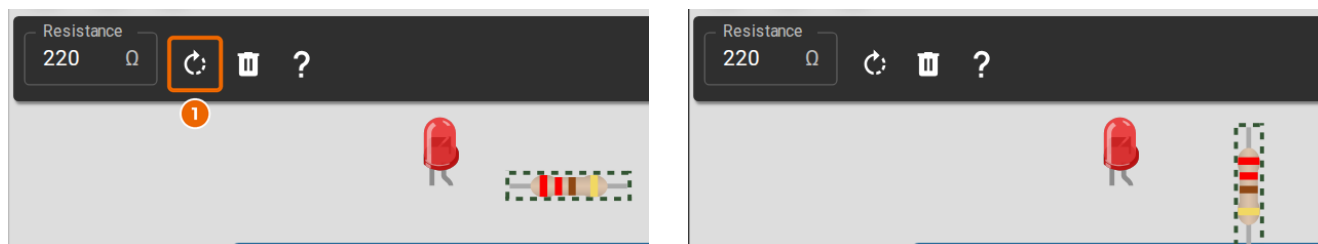


Рисунок В.30 – Поворот: слева резистор лежит, справа – после нажатия «Rotate» (1) он встал вертикально

Шаг 5. Провода. Щёлкни по ножке одной детали – от неё потянется провод. Щёлкни по ножке другой детали – провод закрепится. Передумал, пока тянешь, – нажми Esc, и провода не будет. Наведи мышь на ножку, и Wokwi её подпишет: led1:C – это деталь led1 и её ножка C, катод (у светодиода есть ещё led1:A – анод). Пока тянешь провод, цвет можно сменить цифрой: 2 – красный, 0 – чёрный, 5 – зелёный (как цвета полосок на резисторах). Лишний провод убирают так: щёлкни по нему и нажми Delete.

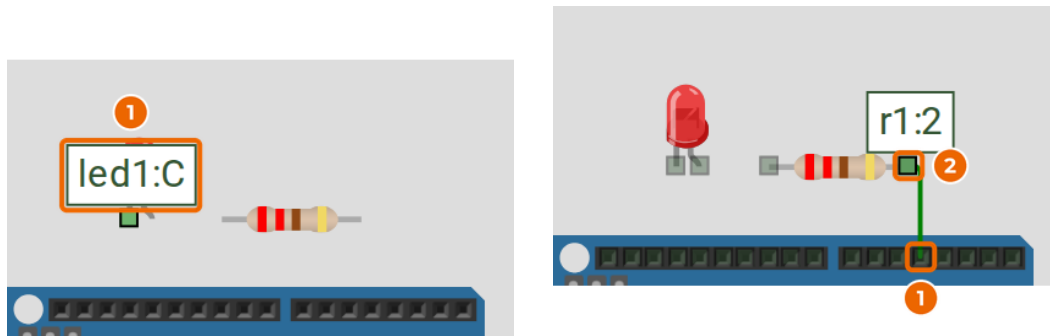


Рисунок В.31 – Слева – подпись ножки при наведении (1), справа – провод тянется от ножки 4 (1) к ножке r1:2 резистора (2)

Шаг 6. Запуск и монитор порта. Нажми «Start the simulation». Пока Wokwi собирает программу, подожди, потом схема оживёт: светодиоды горят, кнопки нажимаются мышью. На месте зелёной кнопки появятся три: «Restart the simulation» (начать заново), «Stop the simulation» (остановить, 1) и «Pause» (пауза). Кнопки «+» и меню с тремя точками при этом скрыты: чтобы поправить схему, сначала нажми «Stop». Вверху справа бежит время симуляции (2). Если в программе есть `Serial.println(...)`, внизу откроется **монитор порта** (3). Это окно, куда плата «печатает» числа и слова. В нижнем ряду у него кнопка графика («Serial plotter»), корзина (очистить окно) и стрелка (свернуть монитор).

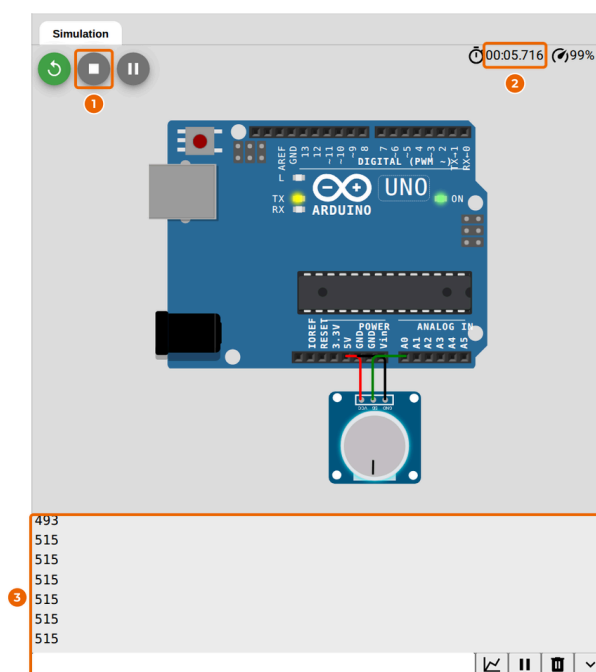


Рисунок В.32 – Идёт симуляция: «Stop the simulation» (1), время (2), монитор порта (3)

Шаг 7. Сохранить. Без входа работу сохраняют архивом: нажми стрелку рядом с «Save» (1) и выбери «Download project ZIP» (2). Скачается `project.zip`: в нём `sketch.ino` – программа, `diagram.json` – схема и ещё `wokwi-project.txt` – ссылка на сайт. Чтобы вернуться к работе, открой новый проект, во вкладку `sketch.ino` вставь текст из `sketch.ino`, а во вкладку `diagram.json` – текст из `diagram.json`.

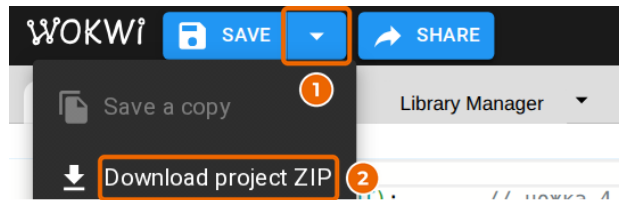


Рисунок В.33 – Меню рядом с «Save»

Задача П.1. Мигалка. Собери схему: ножка 4 → резистор 220 Ом → анод светодиода, катод светодиода → GND. На рисунке провод от ножки 4 (1) идёт к одному концу резистора (2), от другого его конца – к аноду светодиода (3), а от катода (левая ножка) – на GND (4) рядом с ножкой 13. Резистор ограничивает ток: в настоящей схеме без него светодиод бы сгорел.

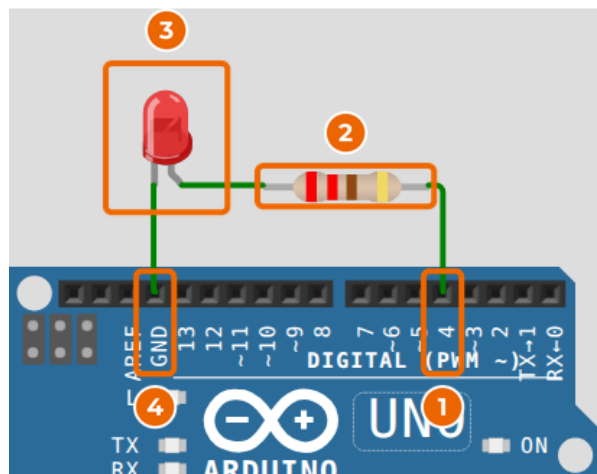


Рисунок В.34 – Схема мигалки

Вставь в окно программы текст и нажми «Start the simulation».

```
void setup() {
  pinMode(4, OUTPUT);      // ножка 4 - выход
}

void loop() {
  digitalWrite(4, HIGH);   // светодиод включить
  delay(1000);             // ждать 1000 мс = 1 с
  digitalWrite(4, LOW);    // выключить
  delay(1000);             // ждать ещё 1 с
}
```

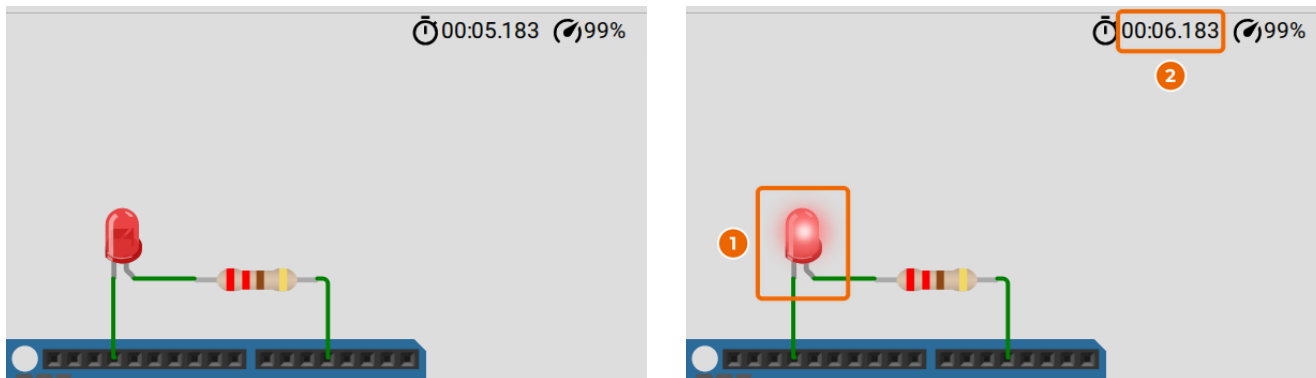


Рисунок В.35 – Слева светодиод не горит, справа горит (1); время симуляции – (2)

Что должно получиться: секунду светодиод горит, секунду нет. Теперь нажми «Stop», поменяй оба `delay(1000)` на `delay(250)` и запусти снова. Сколько раз вспыхнет светодиод за 10 секунд по времени симуляции? Ответ: 20. Одна вспышка – это 250 мс горит и 250 мс не горит, то есть 0,5 с, а $10 : 0,5 = 20$. Мы прогнали это в Wokwi: за 10 секунд ровно 20 вспышек. Светодиод не горит совсем? Проверь полярность: анод (правая ножка) должен смотреть в сторону резистора, а катод – на GND. Включённый наоборот светодиод не загорится, но и не сгорит.

Задача П.2. Кнопка. Открой новый проект (шаг 1): схема из П.1 больше не нужна. Лампочку возьмём ту, что уже стоит на плате: это светодиод L, он подключён к ножке 13. Добавь Pushbutton. У кнопки четыре вывода – два сверху и два снизу. Внутри кнопки два верхних вывода соединены всегда, два нижних тоже, а нажатие соединяет верхнюю пару с нижней. Поэтому провода берут так: один вывод верхней пары – к ножке 2, один вывод нижней пары – к GND. На рисунке это верхний правый вывод (1) и ножка 2 (3), нижний левый вывод (2) и GND (4).

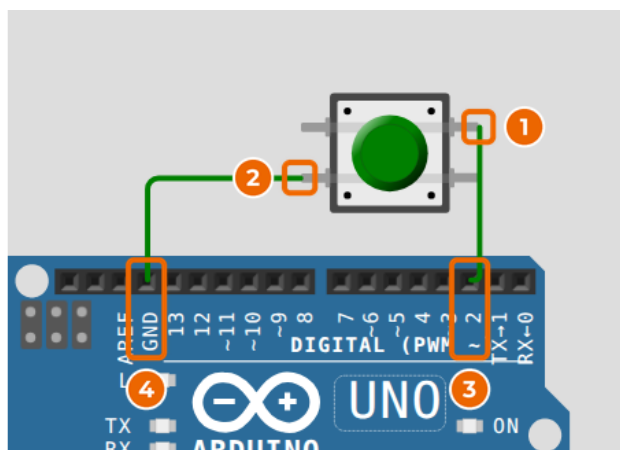


Рисунок В.36 – Схема с кнопкой

Программа. Здесь кнопка – вход с подтяжкой (`INPUT_PULLUP`): пока кнопка отпущена, на ножке 1.

```

void setup() {
  pinMode(2, INPUT_PULLUP); // кнопка: вход, подтянутый к 5 В
  pinMode(13, OUTPUT);      // светодиод L на плате
  Serial.begin(9600);
}

void loop() {
  int knopka = digitalRead(2);
  Serial.println(knopka);
  if (knopka == LOW) {      // нажата = LOW
    digitalWrite(13, HIGH);
  } else {
    digitalWrite(13, LOW);
  }
  delay(200);
}

```

Запусти, потом нажми на кнопку мышью и подержи, а затем отпусти.

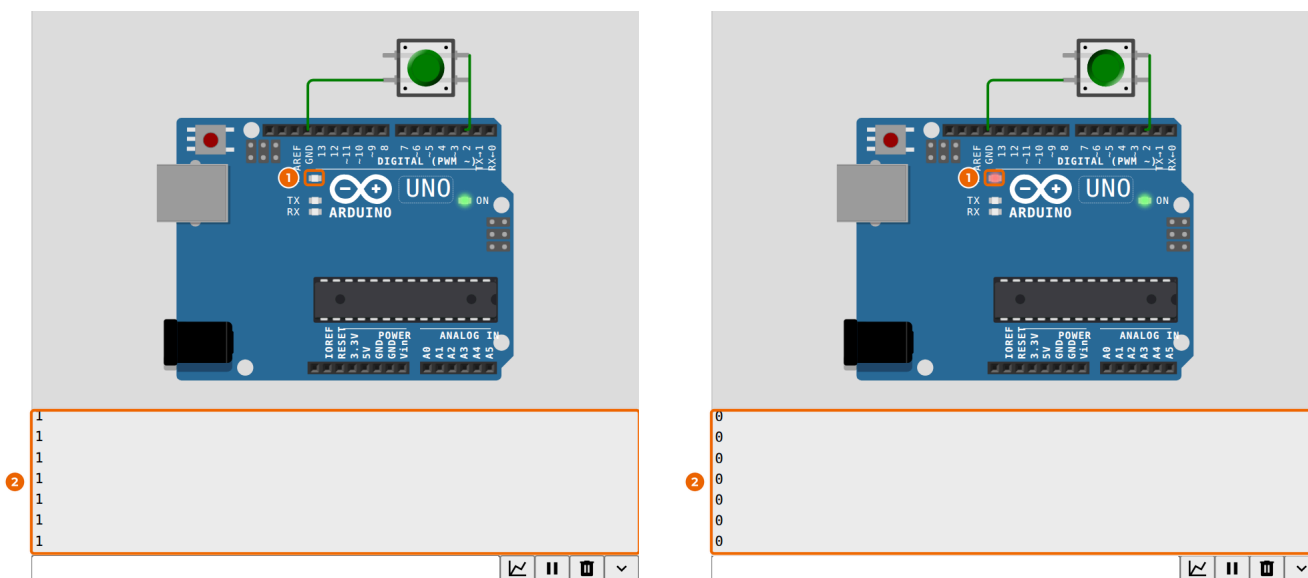


Рисунок В.37 – Слева кнопка отпущена, справа нажата: лампочка L (1) и монитор порта (2)

Что должно получиться: пока кнопка отпущена, в мониторе порта единицы, а L не горит. Нажал и держишь – пошли нули, и L загорается. Отпустил – снова единицы. Ответ: нажатой кнопке соответствует 0 (LOW), отпущенной – 1. Сплошные нули без нажатия? Значит, оба провода сели на выводы одной пары, например на два верхних: такая кнопка «нажата» всегда. Перенеси один провод на вывод другой пары.

Задача П.3. Потенциометр. Открой новый проект. Перетащи плату повыше: под ней нужно место. Добавь «Potentiometer» (строка поиска, potentiometer), поставь его под платой, щёлкни по нему и дважды нажми R: выводы окажутся сверху, ближе к плате. Они подписаны мелко: VCC, SIG, GND. Провода: красный – от VCC на 5V, чёрный – от GND на GND рядом с 5V, зелёный – от SIG на A0. На рисунке это выводы потенциометра (1), питание платы (2) и аналоговая ножка A0 (3).

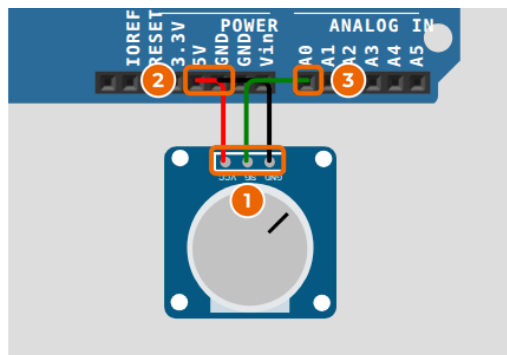


Рисунок В.38 – Схема с потенциометром

Программа печатает в монитор порта число с ножки A0.

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  int n = analogRead(A0);    // число от 0 до 1023
  Serial.println(n);
  delay(300);
}
```

Запусти, нажми мышью на круглую ручку и води по кругу: ручка идёт за курсором, а числа в мониторе меняются сразу. Между двумя упорами у ручки есть пустой промежуток. Если провести через него, число перескочит с 1023 на 0 или обратно.

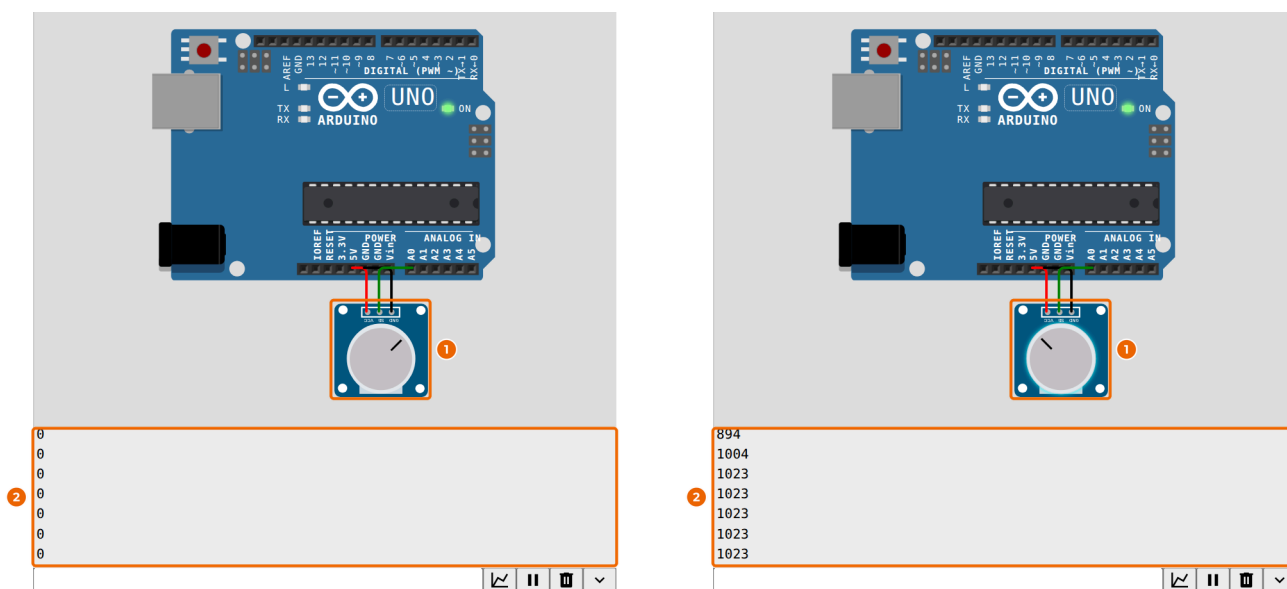


Рисунок В.39 – Слева ручка в одном крайнем положении, справа – в другом: ручка (1) и монитор порта (2)

Что должно получиться: в одном крайнем положении ручки в мониторе 0, в другом – 1023, а посередине – около 500. Сразу после добавления ручка стоит на нуле, поэтому первые числа в мониторе – нули. Ответ: 0 и 1023. Это границы `analogRead`: 0 соответствует 0 В, а 1023 – 5 В. Вопрос на подумать: какое напряжение на A0, когда в мониторе 512? Ответ: около 2,5 В, потому что $512 \cdot 5 / 1023 \approx 2,5$.

План на месяц до отбора

Если начинаешь в конце сентября, вот спокойный план: примерно по часу в день.

Таблица В.3

Неделя: что читаем	Главное дело недели
1: главы 1–3	записаться на курс сферы
2: главы 4–5	собрать схемы в Tinkercad
3: главы 6–7	запустить программы из разборов
4: глава 8 и шпаргалка	тест курса сферы (+5 баллов)

Подробнее по неделям:

- **Неделя 1.** Запишись на курс сферы и войди в класс «Роботех» в Tinkercad – просто чтобы убедиться, что всё открывается. Каждый день решай «Реши сам» к свежим разборам: путь колеса, скорость, координаты, повороты.
- **Неделя 2.** Прогони в тетради алгоритмы исполнителей из главы 4. Собери в Tinkercad схемы из главы 5: мотор через драйвер, ультразвуковой дальномер, сервомотор.
- **Неделя 3.** Повтори программы из разборов глав 6 и 7, посмотри показания датчиков в мониторе порта. Нарисуй в Кибериаде хотя бы одну машину состояний.
- **Неделя 4.** Заново реши задачи, где были ошибки, пролистай шпаргалку и пройди тест курса сферы.

С 1 октября у олимпиады по плану открывается демо-этап с пробными заданиями. Попробуй их, как только появятся: так заранее увидишь, как выглядят задачи на «Орбите» и куда вводить ответы, и не будешь тратить на это время отбора.

Отбор длится до 10 ноября, и попыток две. Если первая прошла неудачно, вернись к главам, где потерял баллы, и иди на вторую.

Родителям и наставникам

- **Помогайте разбираться, но не решайте за ребёнка.** На командном финале сразу видно, кто разобрался сам, а за кого решили. Лучше пусть ребёнок ошибётся на отборе и научится, чем попадёт в финал и растеряется там.
- **Садитесь рядом на первых задачах в симуляторе.** Детей останавливает скорее «не могу войти в класс», «схема пропала» или «сервер недоступен», чем трудная задача. Две подсказки из вебинаров сферы: в общем классе Tinkercad проект может не сохраниться до следующего захода, поэтому код стоит копировать в файл, а схему зарисовывать в тетрадь; если симулятор пишет, что сервер недоступен, помогает подождать и обновить страницу.
- **Если дома есть настоящий робот или плата Arduino,** схему меняют только при выключенном питании, мотор подключают только через драйвер, а пальцы держат подальше от колёс и шестерёнок, пока робот включён. В симуляторе ничего не сгорит, а настоящий светодиод без резистора сгорает за секунду.

- **Всё про регистрацию, бонусы и финал** для Новосибирской области – на nto.kvantorium54.ru (вкладки «Я наставник» и «Я участник или родитель»). Там же – про бонус за нормативы ТехноГТО и его сроки.
- **Региональный координатор НТО Юниоры в Новосибирской области** – детский технопарк «Кванториум», info@kvantorium54.ru.

Честно об этой книге

Главное, что нужно знать про эту книгу: заданий отбора 2026/27 мы не видели – организаторы их заранее не показывают. Угадывать их бесполезно: наставники, которые следят за олимпиадой не первый год, говорят, что задания из года в год не повторяются. Поэтому книга учит темам из программы сферы и приёмам решения, а не готовит к конкретным задачам. Когда организаторы выложат пробные задания, мы сверим с ними книгу и выпустим новую версию.

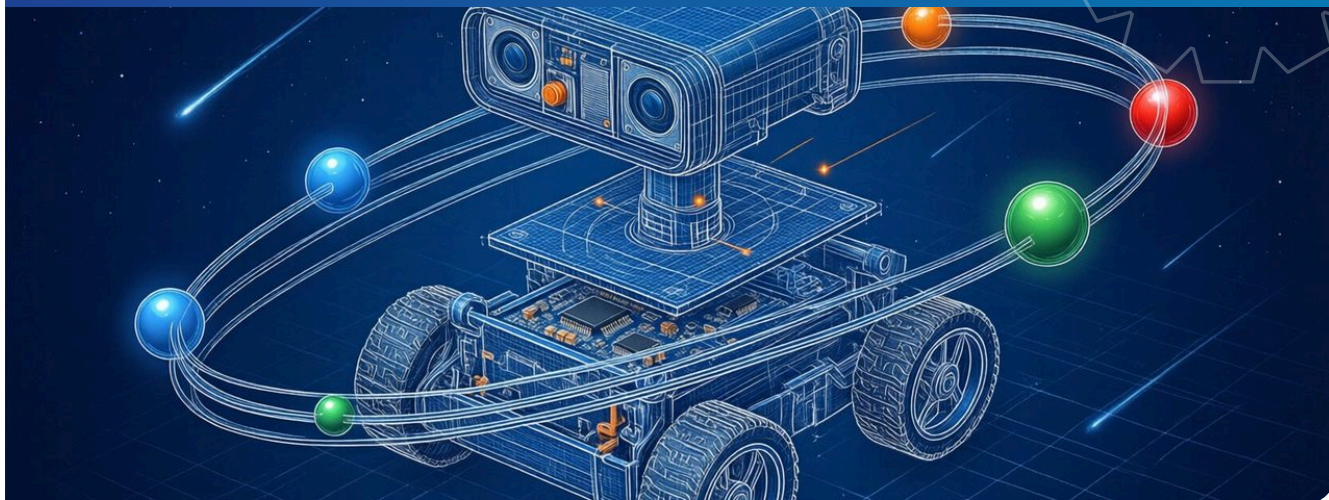
Это не официальное издание НТО. Книгу составил методист «Кванториума» по открытым материалам олимпиады: программе сферы «Роботех. Основы мобильной робототехники» 2026/27, записям вебинаров сферы для наставников и участников, официальным требованиям к площадкам финала.

У «Роботеха» были предшественницы: «Технологии для мира роботов», «Технологии и роботы», а в 2022 году – сразу две сферы, «Технологии и роботы-исследователи» и «Технологии и роботы на производстве». Но тексты их отборочных задач в открытом доступе закрыты: от курса 2023 года видны только названия уроков вроде «Математика наше всё!» и «Пора разобратся со схемами». Эти названия подсказали нам, о чём спрашивают. Ещё несколько идей мы взяли из сборника задач старшей олимпиады по профилю «Интеллектуальные робототехнические системы» 2024/25 – там задачи для 8–11 классов, поэтому мы их сильно упростили. Остальные задачи придуманы заново по программе сферы.

Книга написана до демо-этапа с пробными заданиями. Когда он откроется, мы сверим книгу с настоящим форматом задач и обновим её. Если на пробном этапе тебе попало что-то, чего в книге нет, смело пиши – это поможет следующей версии. Кнопки Tinkerscad и Кибериады описаны такими, какими их показывали на вебинарах сферы в августе и сентябре 2026 года: если интерфейс изменится, смысл останется тем же.

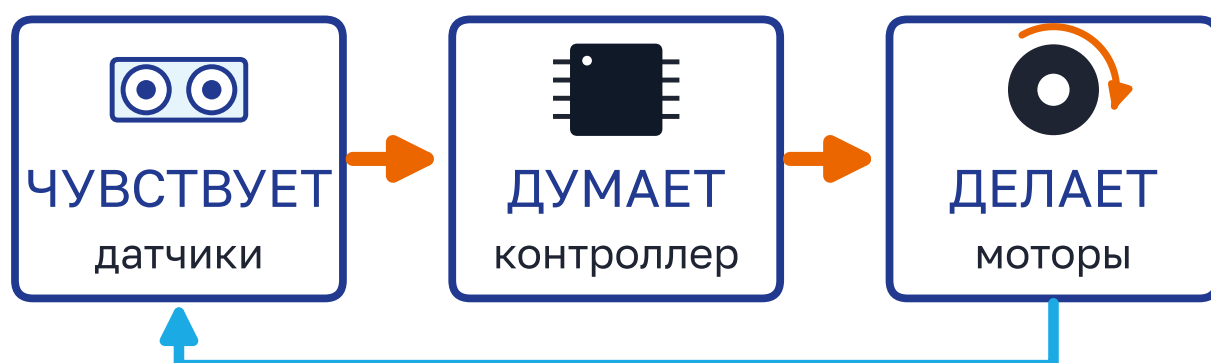
Каждое число в ответах мы пересчитали на компьютере, а каждую программу запустили. Если ты всё-таки нашёл ошибку или что-то осталось непонятным, напиши автору: osinovsky.aa@kvantorium54.ru. Следующие версии книги станут лучше благодаря тебе.

Как устроен робот



С этой главы начинается вся книга: прежде чем учить робота ездить по линии и искать выход из лабиринта, надо знать, из каких деталей он собран и что делает каждая. Здесь почти нет формул – зато много названий, которые дальше встретятся на каждой странице. В конце главы – две механические штуки, без которых не обходится ни один робот: зубчатая передача и рычаг. На отборе по этой теме бывают задачи «сопоставь деталь и её работу», «выбери все верные утверждения» (в ответ вводишь цифры выбранных пунктов) и короткие расчёты: сколько оборотов сделает колесо, если между ним и мотором стоят шестерёнки.

НА ПАЛЬЦАХ



мир вокруг изменился – датчики это заметят

Рисунок 1.1 – Робот всё время делает три вещи по кругу

Датчики замечают, что творится вокруг. Контроллер решает, как поступить. Моторы выполняют решение – и мир вокруг робота меняется, а датчики это снова замечают.

Запомни. Спорный случай «робот или нет» решай одним вопросом: **кто принимает решение?** Человек с пультом – значит, это управляемая машина. Программа по сигналам датчиков – робот. Мобильный робот вдобавок сам перемещается: на колёсах, гусеницах, ногах или винтах.

НА ПАЛЬЦАХ

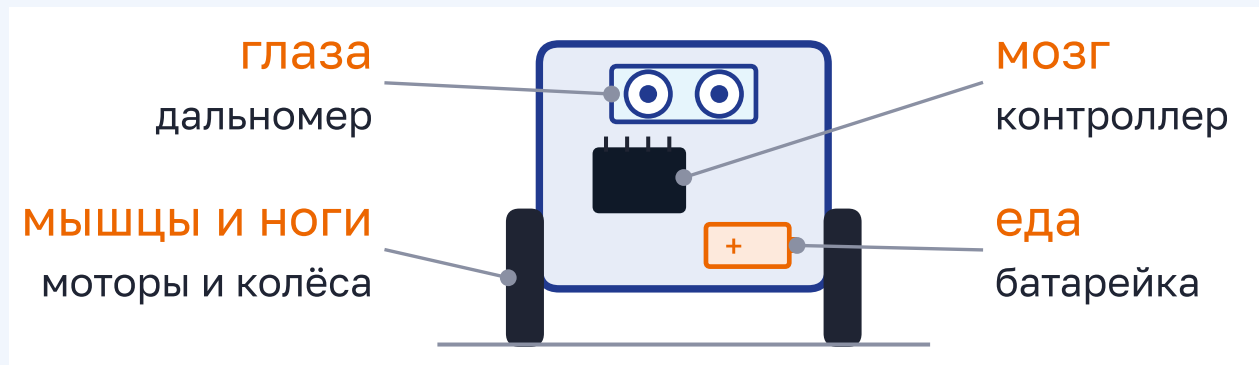


Рисунок 1.2 – Робот похож на человека: у каждой его детали есть «родственник» в нашем теле

Датчики – органы чувств, контроллер – мозг, моторы – мышцы, батарейка – еда, которая даёт силы.

Запомни. Провод от вывода Arduino прямо к мотору – ошибка, даже если в схеме на экране так «короче». Контроллер только командует, а кормит мотор батарея. Между контроллером и мотором всегда стоит драйвер.

НА ПАЛЬЦАХ



Рисунок 1.3 – Руля нет – поворачивают колёса, которые крутятся по-разному

Колёса крутятся одинаково – робот едет прямо. Одно отстаёт – робот сворачивает в его сторону. Колёса крутятся в разные стороны – робот вертится на месте.

Запомни. Робот поворачивает в сторону того колеса, которое крутится медленнее (или стоит, или едет назад). Сам себя проверяй так: встань, вытяни руки в стороны – это колёса. Шагни вперёд только левой стороной тела – и повернёшься направо.

Запомни. $i = z_{\text{ведомой}} / z_{\text{ведущей}}$. Скорость ведомой = скорость ведущей : i . Во сколько раз медленнее – во столько же раз сильнее. Ступени – перемножаем. Каждое зацепление меняет направление вращения, паразитная шестерня скорость не меняет.

Запомни. Сила × плечо = сила × плечо. Лёгкий груз уравновесит тяжёлый, если висит дальше от опоры – во столько же раз дальше, во сколько он легче. Плечо меряют от опоры, а не от края рычага.

Разбор для старта

Задача 1.1 Деталь и её работа

разминка · устройство мобильного робота, сопоставление

В кружке собирают робота-разведчика для макета лунной базы. На столе лежат детали:

1 – плата Arduino Uno; 2 – ультразвуковой дальномер; 3 – драйвер моторов; 4 – мотор с редуктором; 5 – сервомотор; 6 – батарейный отсек; 7 – датчик линии.

Что делает деталь:

- **А** – измеряет расстояние до стены;
- **Б** – выполняет программу и решает, что делать;
- **В** – крутит колесо;
- **Г** – даёт энергию всем устройствам;
- **Д** – поворачивает клешню захвата на заданный угол и держит её;
- **Е** – отличает чёрную линию от белого поля;
- **Ж** – получает слабую команду от контроллера и пропускает к моторам сильный ток от батареи.

Для каждой работы А, Б, В, Г, Д, Е, Ж запиши номер детали. Ответ – строка из семи цифр.

Дано: 7 деталей (номера 1–7), 7 работ (буквы А–Ж).

Найти: номера деталей в порядке А, Б, В, Г, Д, Е, Ж.

Решение

Шаг 1. Сначала то, что узнаётся с ходу. «Измеряет расстояние» – дальномер, само слово так и собрано: «даль» + «мерить». А – 2. «Крутит колесо» – мотор с редуктором. В – 4. «Даёт энергию» – батарейный отсек. Г – 6.

Шаг 2. Датчик линии. Работа Е – отличать чёрное от белого. Это датчик линии, он ловит отражённый от поля свет. Е – 7.

Шаг 3. Мотор или сервомотор. Остались два мотора. Работа Д – повернуть на **заданный угол** и держать. Это умеет сервомотор, а обычный мотор просто крутится, пока на него подадут ток. Д – 5.

Шаг 4. Мозг и посредник. Остались плата и драйвер. «Выполняет программу» – плата Arduino, Б – 1. «Пропускает к моторам сильный ток по команде контроллера» – драйвер, Ж – 3.

Шаг 5. Собираем строку. А – 2, Б – 1, В – 4, Г – 6, Д – 5, Е – 7, Ж – 3. Каждая цифра от 1 до 7 встретилась по одному разу – ни одна деталь не потерялась и не досталась двум работам.

 **Типичная ошибка.** Отдать работу Ж плате Arduino: «командует же контроллер». Командует он, но сильный ток идёт мимо него – через драйвер. Вторая частая путаница – мотор и сервомотор в работах В и Д. Ищи в условии слова «на заданный угол»: это про сервомотор.

Ответ: 2146573.

Своя задача по программе сферы, модуль 2

Проверь себя

1.1. Детали: 1 – ведущее колесо; 2 – шасси; 3 – опорное колесо-шарик; 4 – кнопка; 5 – светодиод.

Работы: **А** – подаёт контроллеру сигнал «старт»; **Б** – держит на себе все остальные детали; **В** – светится и показывает, в каком режиме робот; **Г** – третья точка опоры, чтобы робот не клевал носом; **Д** – толкает робота, отталкиваясь от пола; **Е** – измеряет температуру воздуха.

Для каждой работы А–Е запиши номер детали. Если ни одна деталь из списка эту работу не делает, поставь 0. Ответ – строка из шести цифр.

1.2. Выбери все **датчики**: 1 – кнопка; 2 – ультразвуковой дальномер; 3 – сервомотор; 4 – фоторезистор; 5 – пищалка; 6 – датчик линии; 7 – драйвер моторов; 8 – датчик температуры. Ответ – цифры выбранных пунктов по возрастанию.

1.3. На валу мотора шестерня с 10 зубьями, на оси колеса – с 50 зубьями, они сцеплены напрямую. Мотор делает 250 об/мин. Сколько оборотов в минуту делает колесо?

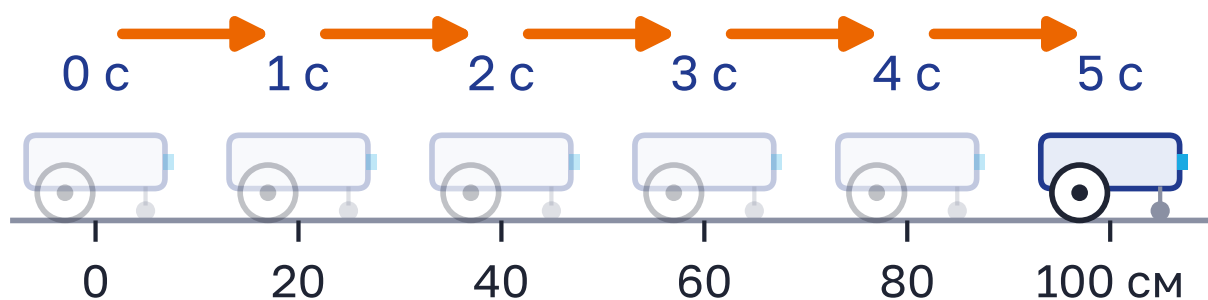
1.4. Ведущая шестерня с 15 зубьями делает 120 об/мин. Сколько зубьев должно быть у ведомой шестерни, чтобы она делала 40 об/мин?

Движение: скорость, путь, время



Робот на полигоне всё время куда-то едет, и задачи про него начинаются со слов «сколько» и «где»: сколько секунд ехать до стенки, сколько оборотов сделает колесо, где робот встанет после команды «стоп». Формула тут одна, и она простая. Баллы теряют на единицах: в одной задаче встречаются миллиметры, метры, миллисекунды и обороты в минуту, и стоит одну не перевести – ответ уезжает в десять или сто раз. Вторая половина главы – про то, почему робот не встаёт как вкопанный и почему на гладком полу колёса буксуют.

каждую секунду – ещё 20 см



за 5 секунд: $5 \cdot 20 = 100$ см

Рисунок 2.1 – Робот едет с одной и той же скоростью – 20 см каждую секунду

Скорость – сколько робот проезжает за одну секунду. Прошло 5 секунд – значит, пять раз по 20 см.

Запомни. $s = v \cdot t$. Ищешь время – путь дели на скорость, ищешь скорость – путь дели на время. Средняя скорость – весь путь, делённый на всё время вместе с остановками; две скорости пополам не складывают.

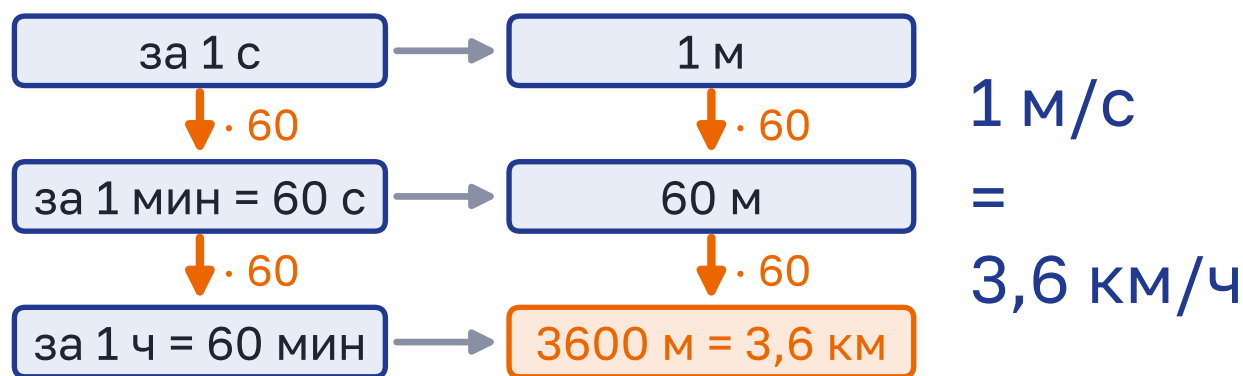


Рисунок 2.2 – Растягиваем одну секунду до целого часа

Робот, который за секунду проезжает метр, за минуту проедет 60 м, а за час – 3600 м, то есть 3,6 км. Поэтому 1 м/с – то же самое, что 3,6 км/ч.

Запомни. $1 \text{ м/с} = 100 \text{ см/с} = 3,6 \text{ км/ч}$. $1 \text{ об/с} = 60 \text{ об/мин}$. $1 \text{ с} = 1000 \text{ мс}$. Чем мельче единица, тем больше число: $0,3 \text{ м/с} = 30 \text{ см/с} = 300 \text{ мм/с}$.

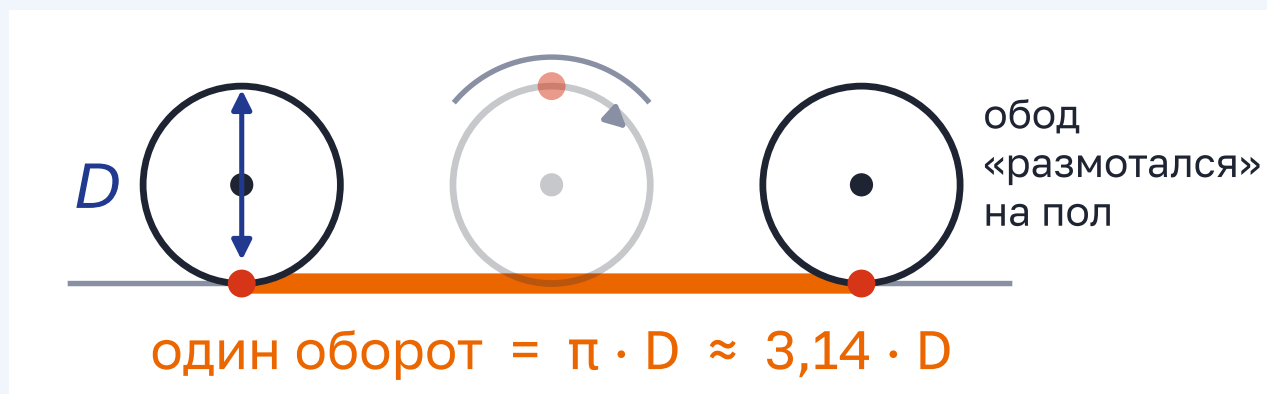


Рисунок 2.3 – Колесо с каплей краски на ободе прокатилось на один оборот

Капля снова коснулась пола, когда колесо прокатилось на длину своей окружности. Столько колесо проезжает за каждый оборот: $\pi \cdot D$.

Запомни. Один оборот колеса – путь πD . Оборотов нужно: путь, делённый на πD . Скорость робота – обороты в секунду, умноженные на πD ; обороты в минуту сначала делим на 60. В формулу идёт диаметр: дали радиус – удвой его.

Запомни. На графике «путь – время» скорость видна по крутизне линии: путь участка дели на его время. Полочка – стоянка. Сначала прочитай подписи осей и узнай, сколько стоит одна клетка.

Запомни. После «стопа» робот ещё проезжает тормозной путь, поэтому команду дают раньше на эту длину. Скорость падает (или растёт) равномерно – значит, в среднем робот едет с половиной полной скорости, и путь торможения или разгона равен $\frac{v}{2} \cdot t$.

Запомни. Колесо едет, пока трение не даёт ему проскользнуть. Мотор крутит сильнее, чем трение держит, – колесо буксует, и робот проезжает меньше, чем по оборотам. Недобор в процентах считают от пути по оборотам – от того, сколько робот должен был проехать.

Разбор для старта

Задача 2.1 До стенки полигона

разминка · путь, скорость, время, перевод единиц

Робот стоит на старте и едет прямо к стенке полигона со скоростью 25 см/с. До стенки 1,8 м. Через сколько секунд робот коснётся стенки? Разгон не учитывай. Ответ дай в секундах.

Дано: $v = 25$ см/с; $s = 1,8$ м.

Найти: t – ?

Решение

Шаг 1. Единицы. Скорость дана в сантиметрах в секунду, а путь – в метрах. Переводим путь в сантиметры: $1,8 \text{ м} = 1,8 \cdot 100 = 180 \text{ см}$.

Шаг 2. Время. Время – это путь, делённый на скорость: $t = 180 : 25 = 7,2 \text{ с}$.

Прикинем: каждую секунду робот проезжает 25 см, за 7 с – 175 см. До 180 см не хватает совсем чуть-чуть, так что «чуть больше семи секунд» – как раз наш ответ.

 **Типичная ошибка.** Поделить 1,8 на 25 и записать 0,072 с. Метры поделили на сантиметры в секунду – и робот пересекает почти два метра быстрее, чем ты моргнёшь. Бывает и так: $25 \cdot 1,8 = 45$, но скорость, умноженная на путь, – вообще не время.

Ответ: 7,2 с.

Своя задача по программе сферы, модуль 2

Проверь себя

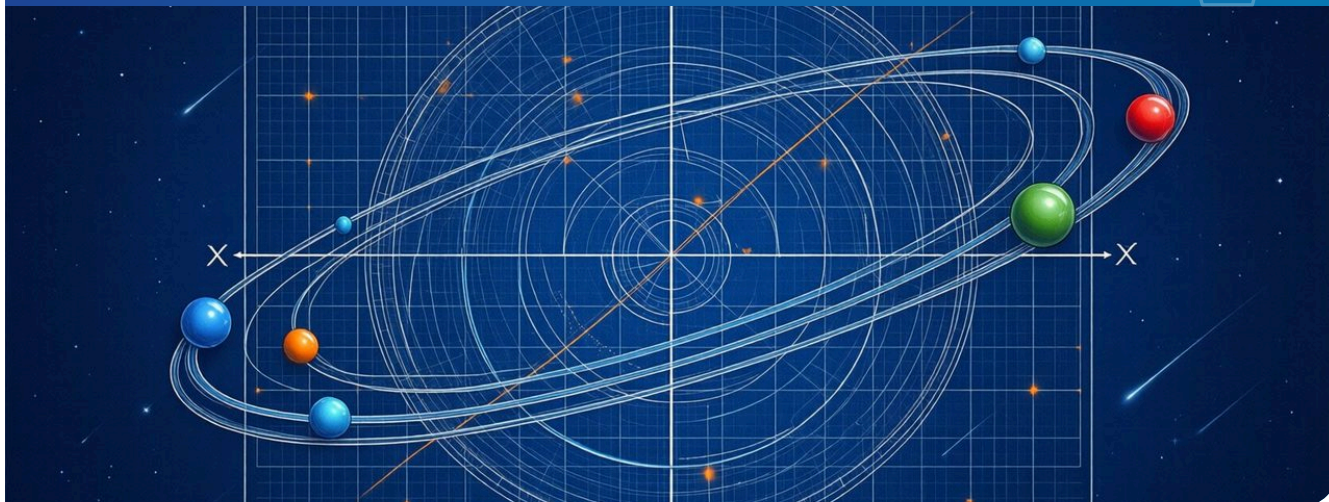
2.1. Робот проехал по коридору 2,4 м за 16 с. С какой скоростью он ехал? Ответ дай в сантиметрах в секунду.

2.2. Два робота стоят на концах прямой дорожки длиной 3,5 м и одновременно выезжают навстречу друг другу: первый со скоростью 20 см/с, второй – 30 см/с. На каком расстоянии от места старта первого робота они встретятся? Ответ в сантиметрах.

2.3. Колесо диаметром 6 см сделало 15 оборотов без проскальзывания. Какой путь проехал робот? Ответ в сантиметрах, округли до целых.

2.4. В описании робота указан радиус колеса – 3,5 см. Сколько оборотов сделает колесо, пока робот проедет 1,1 м? Округли до десятых.

Координаты и углы: навигация робота



Робот на полигоне не понимает слов «у окна» или «возле двери». Ему нужны числа: где он стоит, куда смотрит, сколько клеток ехать и на сколько градусов повернуть. Этими числами мы и займёмся. На отборе задача может дать программу из команд «вперёд» и «поворот» и спрашивает, где робот окажется. Или показывает, куда он смотрит, и просит угол поворота. Или рисует маршрут на плане и хочет знать его длину на настоящем поле.

НА ПАЛЬЦАХ

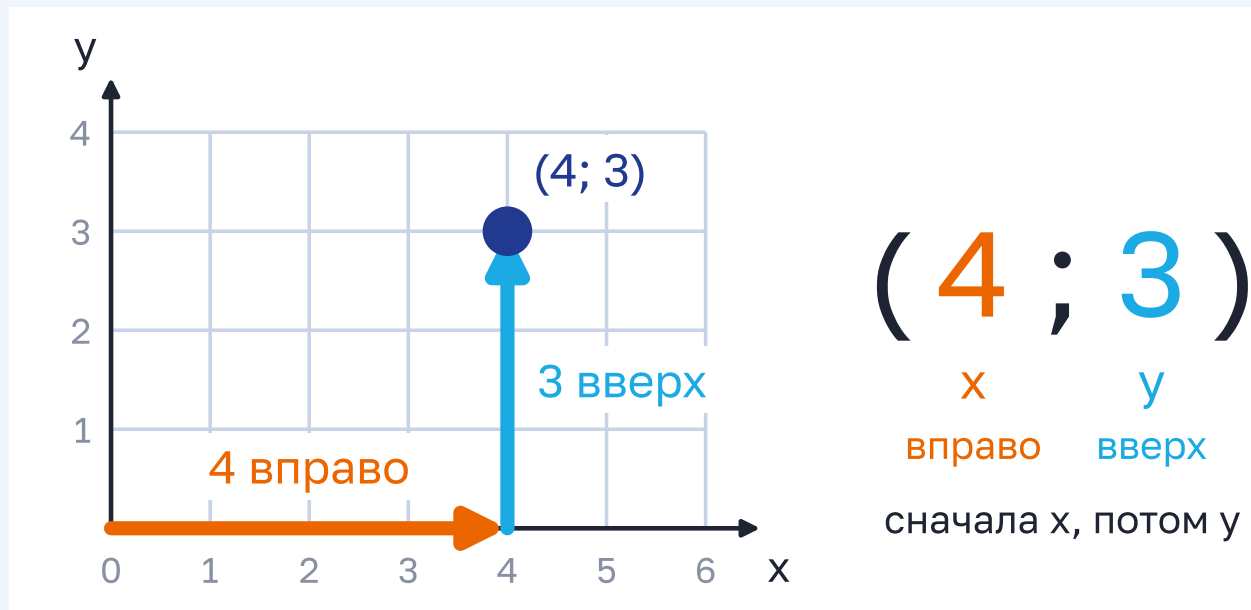


Рисунок 3.1 – Адрес точки – два числа

От угла поля сначала отсчитываем шаги вправо, потом вверх. Точка (4; 3) – четыре шага вправо и три вверх.

Запомни. Точка (x; y): x – шаги вправо от начала отсчёта, y – шаги вверх. Сначала всегда x. Считай промежутки между линиями, а не сами линии.

НА ПАЛЬЦАХ



Рисунок 3.2 – «Направо» у робота – своё

Робот смотрит вниз – и его «направо» на карте показывает влево. Повороты всегда считаем от носа робота.

Запомни. «Вверх-вниз-вправо-влево» – по карте, «направо-налево» – от носа робота. Курс: 0° вверх, 90° вправо, 180° вниз, 270° влево. Направо – прибавь угол, налево – вычти, вышло за 360 или ниже нуля – поправь на 360. В вершине правильной фигуры робот поворачивает на $360^\circ : n$.

НА ПАЛЬЦАХ

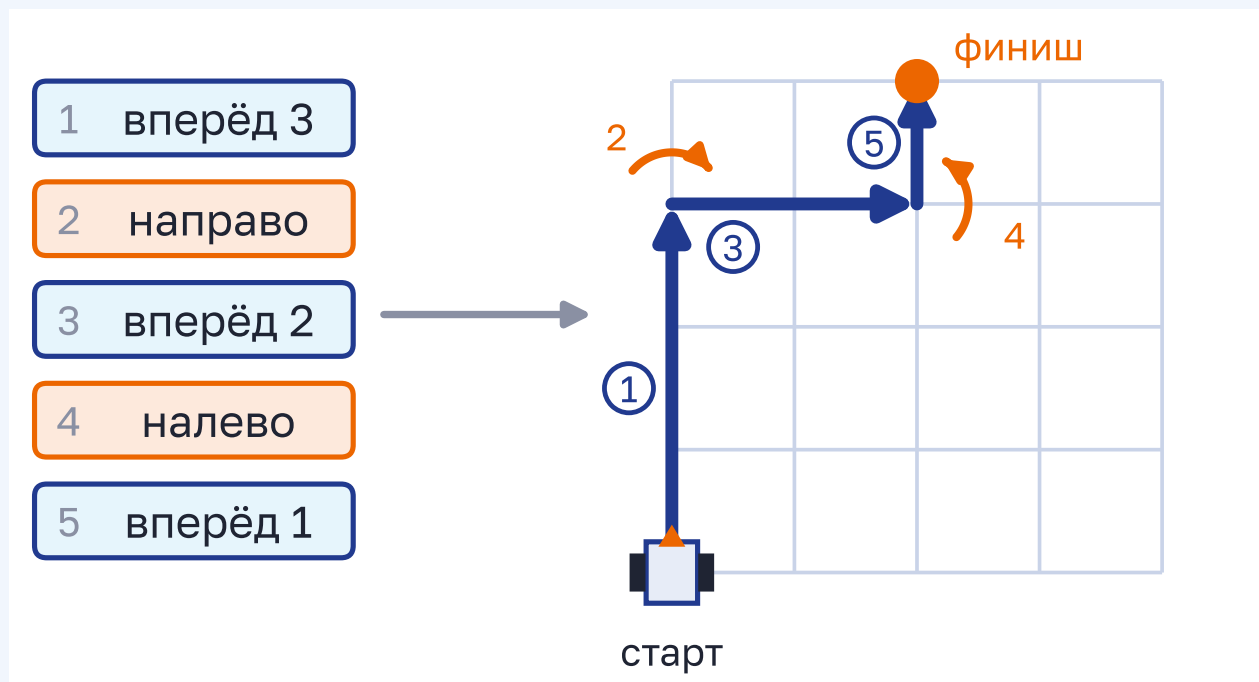


Рисунок 3.3 – Программа – список шагов и поворотов

Робот выполняет команды по одной, сверху вниз. На поворотах он крутится на месте, как фигурист, – колёсами никуда не уезжает.

Запомни. В журнале на каждую команду – своя строка. У строки с поворотом x и y переписываются без изменений, у строки «вперед» меняется одно число из двух, а курс остаётся прежним.

Запомни. Масштаб $1 : k$: с плана на поле – умножь на k , с поля на план – раздели на k . Если масштаб не дан, а даны размеры одного и того же отрезка на плане и на поле, переведи их в одни единицы и раздели большее на меньшее.

Запомни. По линиям: сдвиг по x плюс сдвиг по y , оба без минуса. Маршрут через точки – сумма кусков, каждый от предыдущей точки. Напрямую – корень из суммы квадратов сдвигов, и это всегда не длиннее, чем по линиям.

Запомни. Разворот на месте: колёса едут по окружности диаметром колеса B . Путь колеса – $\pi B \cdot \alpha / 360$, угол колеса – $\alpha \cdot B / D$. Вокруг стоящего колеса путь вдвое длиннее.

Разбор для старта

Задача 3.1 Где окажется робот

разминка · координаты, команды «вперёд» и «поворот», курс

Робот-курьер стоит на полигоне в точке (1; 1) и смотрит вверх, вдоль оси y . Он выполняет программу:

вперёд 3; направо; вперёд 4; направо; вперёд 1; налево; вперёд 2.

«Вперёд N » – проехать N клеток, «направо» и «налево» – повернуть на месте на 90° . В какой точке робот остановится и куда будет смотреть?

Дано: старт (1; 1), курс 0° (вверх); семь команд.

Найти: точку финиша и курс.

Решение

Шаг 1. Заводим журнал. Столбцы: команда, курс, x , y . Первая строка – старт: курс 0° , $x = 1$, $y = 1$.

Шаг 2. Идём по командам. Курс после поворота считаем по правилу из раздела 3.2: направо – плюс 90° , налево – минус 90° . «Вперёд» меняет x или y по таблице из того же раздела.

Таблица 3.1

Команда	Курс	x	y
старт	0° (вверх)	1	1
вперёд 3	0°	1	4
направо	90° (вправо)	1	4
вперёд 4	90°	5	4
направо	180° (вниз)	5	4
вперёд 1	180°	5	3
налево	90° (вправо)	5	3
вперёд 2	90°	7	3

Задержимся на двух строках. Второе «направо» робот делает, глядя вправо: его «направо» на карте – вниз, курс $90 + 90 = 180^\circ$. Потом «налево», глядя вниз: $180 - 90 = 90^\circ$, и робот снова смотрит вправо, а не влево.

Шаг 3. Проверяем по сдвигам. Вправо робот проехал $4 + 2 = 6$ клеток: $1 + 6 = 7$. Вверх 3 и вниз 1, итого вверх на 2: $1 + 2 = 3$. Точка (7; 3) – та же, что в журнале.

 **Типичная ошибка.** Путать «налево» робота с «влево» на карте. После «вперёд 1» робот смотрит вниз, и его «налево» показывает вправо по карте. Кто поехал влево, получил $x = 5 - 2 = 3$ и ответ (3; 3). Бывает и так: на повороте робота мысленно сдвигают на клетку, хотя он крутится на месте и x с y не трогает.

Ответ: (7; 3), робот смотрит вправо.

Своя задача по программе сферы, модуль 10

Проверь себя

3.1. Робот стоит в точке (0; 0) и смотрит вверх. Программа: вперёд 2; направо; вперёд 3; налево; вперёд 2; налево; вперёд 1. Все повороты – на 90° на месте. В какой точке остановится робот?

3.2. Робот стоит в точке (4; 5) и смотрит вправо. Программа: вперёд 2; направо; вперёд 3; развернуться; вперёд 5; налево; вперёд 1. «Развернуться» – поворот на месте на 180° , остальные повороты – на 90° . Где остановится робот и куда он будет смотреть?

3.3. Робот стоит в точке (2; 2) и смотрит вправо. Ему нужно ехать напрямую к точке (5; 5). На какой наименьший угол и в какую сторону ему повернуться?

3.4. Робот проехал 30 см, повернул направо на 40° , снова проехал 30 см, снова повернул направо на 40° – и так далее, пока не вернулся в точку старта. Сколько отрезков по 30 см он проехал?

Алгоритмы и исполнители



В этой главе нет ни формул, ни единиц измерения – только робот на клетчатом поле и короткие программы для него. Зато здесь нужна внимательность: одна перепутанная команда «налево» уведит робота совсем в другой угол поля. Мы научимся проходить программу шаг за шагом, сокращать её циклом, собирать из подпрограмм, выводить робота из лабиринта по правилу правой руки и замечать, когда он попал в круг, из которого не выйдет. На отборе такая задача – поле с роботом и несколько строк программы, а спрашивают, где робот окажется, какую команду вставить или сколько команд он выполнит. В ответ вводится число, клетка или номер варианта.

НА ПАЛЬЦАХ

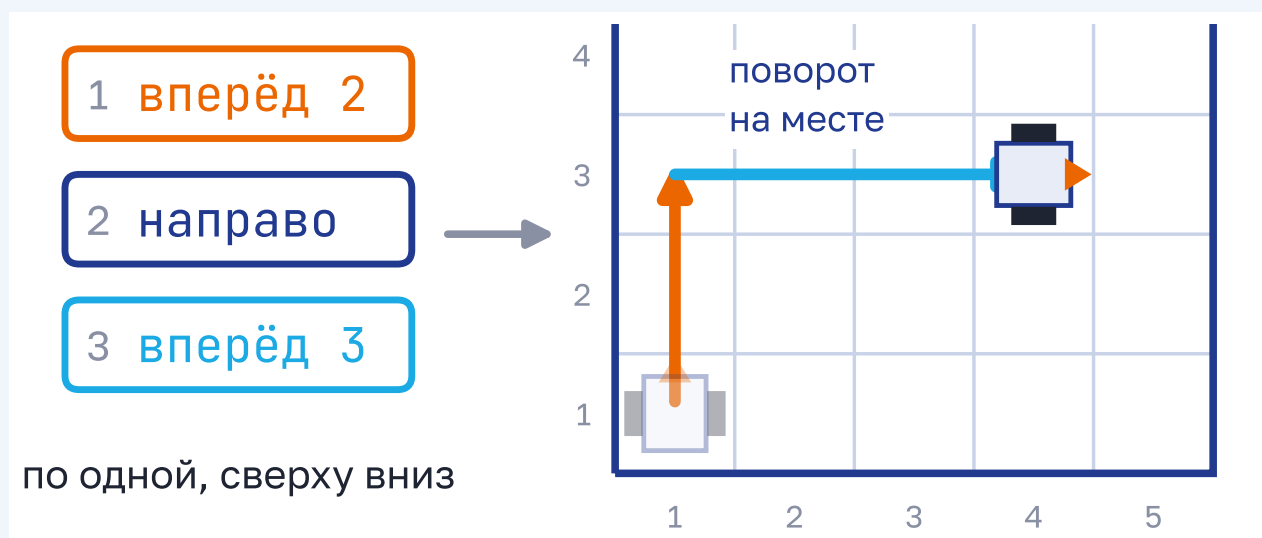
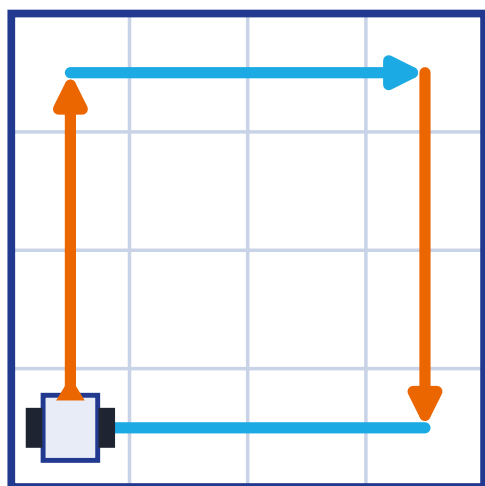


Рисунок 4.1 – Три карточки с командами – и три участка пути робота

Робот берёт карточки по одной, сверху вниз, и делает то, что на них написано. Сам он ничего не придумывает.

Запомни. Робот делает только то, что есть в его системе команд, и строго по порядку. Повороты меняют взгляд, а не клетку, и «направо» считаем от носа робота. Клетка (x; y): сначала столбец, потом строка.

НА ПАЛЬЦАХ



повтори 4:
вперёд 3
направо

4 раза по 2 команды –
8 команд робот выполнит,
а в программе 3 строки

Рисунок 4.2 – Четыре одинаковых куска пути – одна запись «повтори 4»

Учитель физкультуры не говорит «присядь, присядь, присядь...» двадцать раз. Он говорит «двадцать приседаний». Цикл – то же самое для робота.

Запомни. повтори N: – тело выполняется N раз. пока условие: – тело выполняется, пока условие верно; проверка перед каждым повтором. Длина программы и число выполненных команд – разные числа: у повтори 5: с тремя командами в теле длина 4, а выполнено 15.

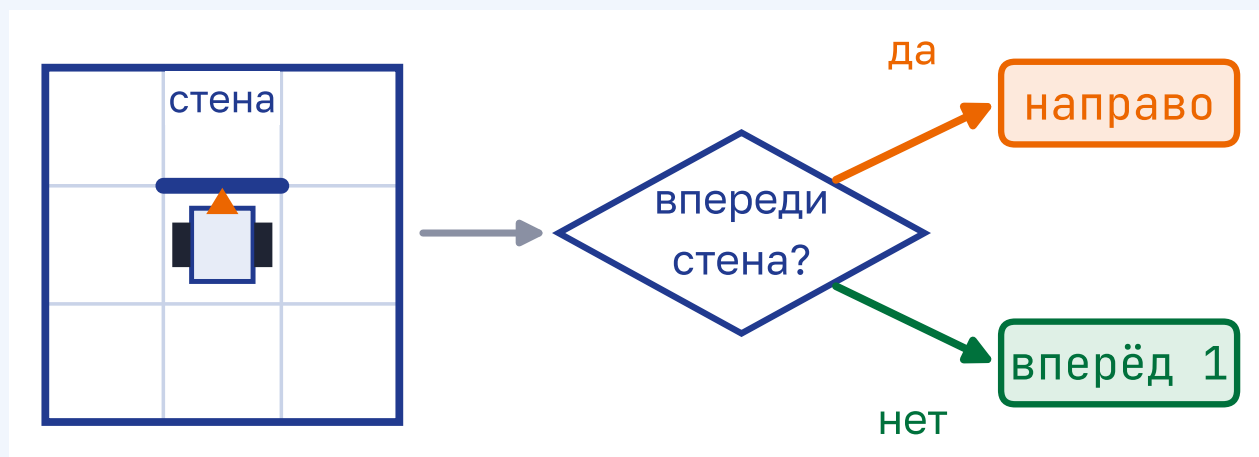


Рисунок 4.3 – Робот упёрся в стену и спрашивает датчик

Как ты утром смотришь в окно: идёт дождь – берёшь зонт, нет – не берёшь. Робот тоже выбирает одну из двух дорог по ответу датчика.

Запомни. В ветвлении выполняется одна ветка: под **если** – когда условие верно, под **иначе** – когда нет. В цепочке **если** – **иначе** **если** побеждает первое верное условие.

Запомни. Подпрограмму описывают один раз и вызывают по имени сколько угодно раз. Само описание ничего не двигает. Команд выполнено = (команд в подпрограмме) · (число вызовов).

Запомни. Правая рука: справа → прямо → слева → разворот, в таком порядке, и после каждого поворота – шаг. Левая рука – зеркально. Путь обратно: список развилок с конца, «направо» ↔ «налево».

Запомни. Трассировка – таблица «шаг – команда – клетка – куда смотрит», по строке на каждую команду. Циклы и подпрограммы при этом раскрываем. Ошибку ищи в первой строке, где «должно» и «получилось» разошлись.

Запомни. Клетка повторилась – робот заиклился и до финиша не доедет. Чтобы узнать, где он будет через N переездов: отними подъезд, то, что осталось, раздели на длину круга с остатком и от первой клетки круга отсчитай вперёд столько переездов, сколько вышло в остатке.

Разбор для старта

Задача 4.1 Доедет или врежется

разминка · линейный алгоритм, трассировка, повороты относительно робота, край поля

Поле 6×5 клеток, стен внутри нет. Робот стоит в клетке (2; 1) и смотрит вверх. Он выполняет программу:

```
вперёд 3
направо
вперёд 3
направо
вперёд 2
налево
вперёд 2
```

Выполнит ли робот программу до конца? Если да – в какой клетке он окажется. Если нет – на какой по счёту команде он врежется и в какой клетке будет стоять в этот момент.

Дано: поле 6×5 ; старт (2; 1), смотрит вверх; 7 команд.

Найти: доедет ли; клетку в конце или в момент аварии.

Решение

Шаг 1. Готовим таблицу. Записываем начальное положение, а потом по строке на каждую команду. **вперёд** меняет клетку, повороты меняют только направление. Держим в голове размер поля: столбцы от 1 до 6, строки от 1 до 5.

Шаг 2. Трассировка.

Таблица 4.1

Шаг	Команда	Клетка	Смотрит
старт	–	(2; 1)	вверх
1	вперёд 3	(2; 4)	вверх
2	направо	(2; 4)	вправо
3	вперёд 3	(5; 4)	вправо
4	направо	(5; 4)	вниз
5	вперёд 2	(5; 2)	вниз
6	налево	(5; 2)	вправо
7	вперёд 2	(6; 2), дальше край	вправо

Шаг 3. Самое опасное место – шаг 6. Робот смотрит вниз. Поверни тетрадь вверх ногами: теперь робот смотрит от тебя, и его «налево» совпадает с твоим левым. Верни тетрадь обратно – эта сторона оказалась справа по листу. Налево от робота, который смотрит вниз, – это вправо по листу.

Шаг 4. Последняя команда. Робот в (5; 2) смотрит вправо, ему нужно проехать 2 клетки. Первая – в (6; 2), это последний столбец поля. Вторая клетка была бы в седьмом столбце, а его нет: робот врежется в край. Он стоит в (6; 2), программа остановилась на 7-й команде.

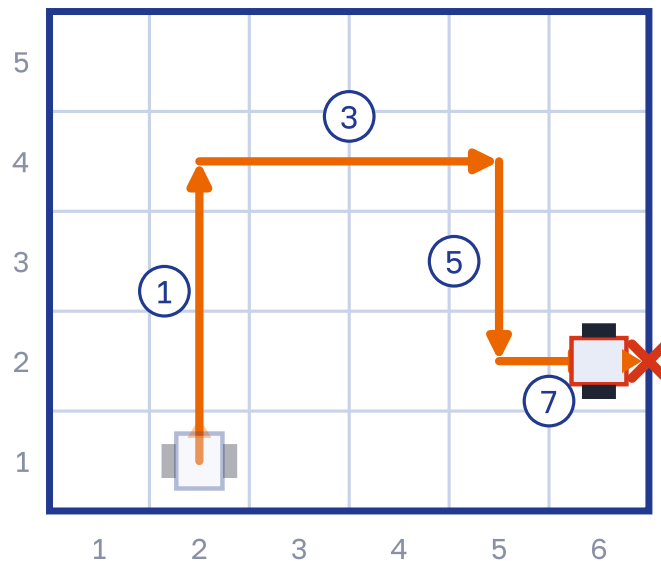


Рисунок 4.4 – Путь робота в задаче 4.1. Цифры – номера команд `вперёд`, красный крест – место аварии

Проверить себя можно программой на Python из раздела 4.6 – это та же задача, и она печатает то же самое.

⚠ Типичная ошибка. На шаге 6 повернуть «налево по листу» – влево. Тогда робот спокойно проедет 2 клетки влево, в (3; 2), и покажется, что программа выполнена до конца. Одна неверно понятая команда поменяла не только клетку, но и сам ответ «доедет – не доедет». Вторая ошибка поменьше: решить, что робот при аварии остаётся в (5; 2). По нашему правилу он проезжает, сколько может, и останавливается у края.

Ответ: не доедет: робот врежется на 7-й команде, стоя в клетке (6; 2).

Своя задача по программе сферы, модуль 10

Проверь себя

4.1. Поле 5 × 5 без стен. Робот стоит в клетке (1; 1) и смотрит вправо. Программа: `вперёд 3`, `налево`, `вперёд 2`, `налево`, `вперёд 4`. Выполнит ли робот её до конца? Если нет – на какой команде он врежется и в какой клетке будет стоять?

4.2. Поле 6 × 6 без стен. Робот стоит в клетке (1; 1) и смотрит вверх. Ему нужно попасть в клетку (5; 4) и в конце смотреть вниз. Какое наименьшее число команд для этого нужно? `вперёд N` с любым N считается одной командой.

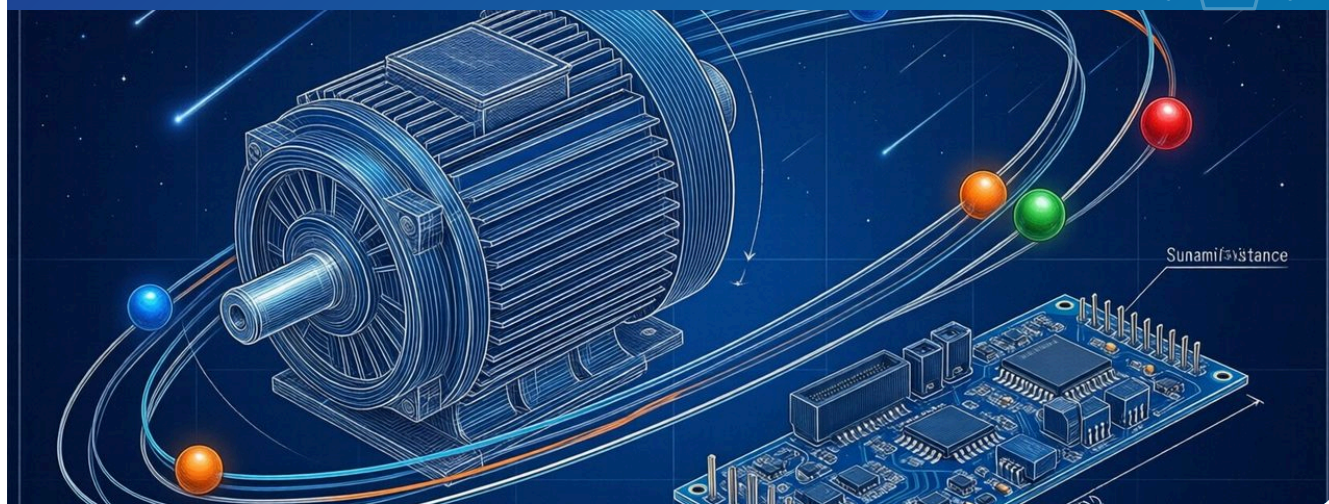
4.3. Программа: `вперёд 2, направо, вперёд 2, направо, вперёд 2, направо, вперёд 2, направо`. Перепиши её с одним циклом как можно короче (длину считаем по правилу задачи 4.2). Какой длины получилась программа? Где окажется робот, если он начал в (1; 1) и смотрел вверх?

4.4. Робот стоит в (1; 1) поля 6×6 и смотрит вверх. Программа:

```
повтори 5:  
  вперёд 1  
  направо  
  вперёд 1  
  налево
```

Сколько команд движения и поворота выполнит робот? Где он окажется и куда будет смотреть?

Моторы, датчики и электричество



Всё, что робот делает, он делает электричеством. Батарея толкает ток, драйвер пропускает его к моторам, датчики превращают свет и звук в напряжение, а контроллер – напряжение в числа. Физики здесь больше, чем в любой другой главе книги, но начинаем с нуля: что такое ток и вольты, узнаешь из первого же раздела. На отборе такие задачи короткие: посчитать ток или подобрать резистор, найти значение ШИМ для нужной скорости, перевести число с датчика в вольты, время эха – в сантиметры, заряд аккумулятора – в минуты работы. Баллы здесь теряют в основном на единицах: миллиамперы вместо ампер, микросекунды вместо миллисекунд.

НА ПАЛЬЦАХ



Рисунок 5.1 – Ток бежит по проводам, как вода по трубам, замкнутым в кольцо

Батарейка – насос, который создаёт напор. Мотор – колесо, которое крутит поток. Узкая труба мешает воде – это сопротивление.

Запомни. Нет замкнутого круга – нет тока. Вольты толкают, амперы текут, омы мешают. 1 А = 1000 мА, 1 кОм = 1000 Ом. На Arduino 5V – плюс, GND – минус.

НА ПАЛЬЦАХ

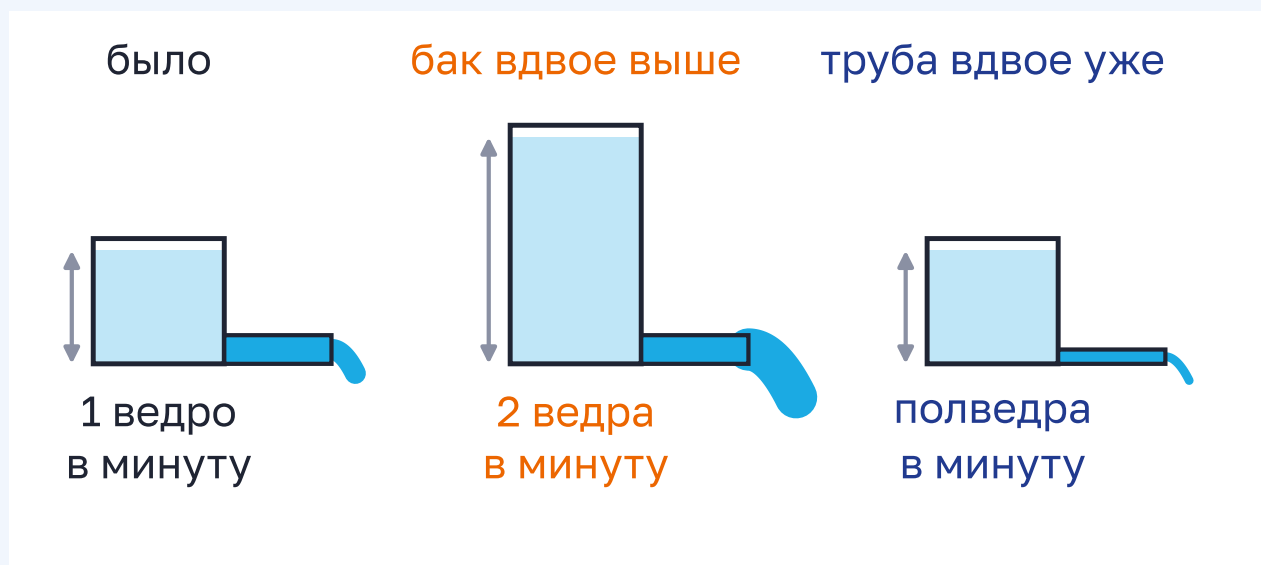


Рисунок 5.2 – Что меняет поток воды

Подняли бак вдвое выше – воды течёт вдвое больше. Сузили трубу вдвое – вдвое меньше. С током то же самое: он растёт вместе с напряжением и падает, когда растёт сопротивление.

Запомни. $I = U / R$, и ток в эту формулу идёт в амперах. Резистор для светодиода считают по напряжению, которое осталось: $(U_{\text{пит}} - U_{\text{св}}) / I$. Из ряда берут ближайший **больший** номинал.

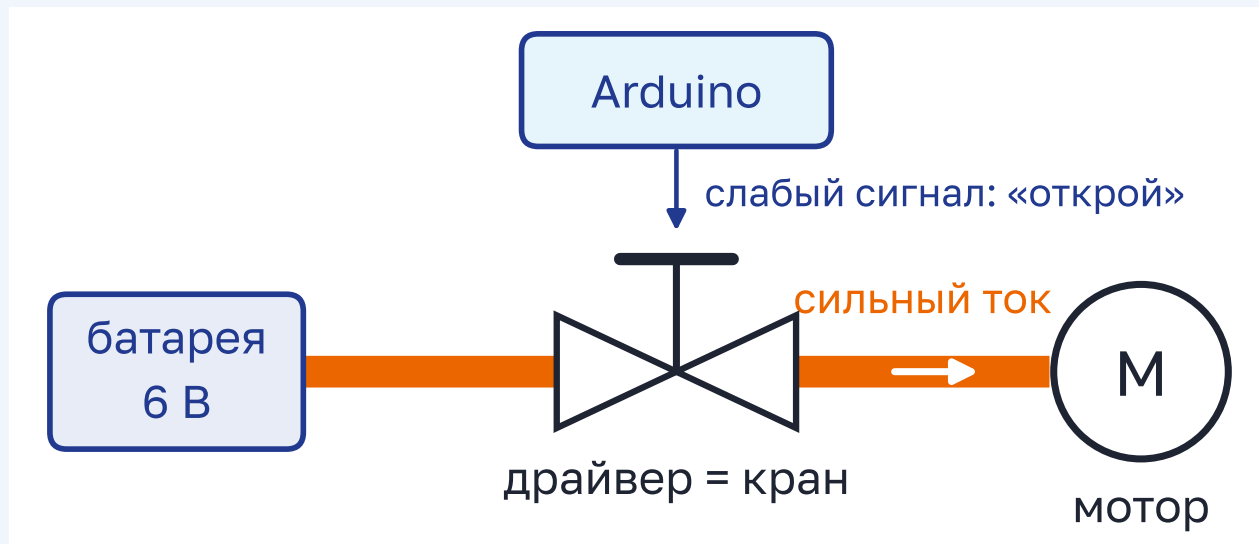


Рисунок 5.3 – Драйвер моторов работает как кран

Контроллер не носит ток к мотору сам. Он только поворачивает ручку крана, а сильный ток бежит к мотору от батареи.

Запомни. Мотор – только через драйвер: вывод платы даёт до 40 мА (безопасно – около 20), мотору нужны сотни. IN1 и IN2 задают направление (разные сигналы – крутится, одинаковые – стоит), EN – можно ли ехать и как быстро. Минус батареи и GND платы соединяют.

Запомни. `analogWrite` : от 0 до 255, только выводы с ~. Значение = доля · 255. Не путай с `analogRead`, у него шкала от 0 до 1023 – о нём раздел 5.6.

Запомни. Время работы = ёмкость : ток; мА·ч делим на мА – получаем часы. Токи всех устройств складываем. Последовательно соединённые батарейки прибавляют вольты, но не мА·ч.

Запомни. `analogRead` – от 0 до 1023, это 0...5 В. Число в вольты: $N \cdot 5 : 1023$. Вольты в число: $U \cdot 5 \cdot 1023$, дробную часть отбрасываем.

Запомни. Белое отражает много света, чёрное – мало. Порог – среднее между показаниями над белым и над чёрным. Какое число у линии больше, а какое меньше, проверяй по показаниям своего датчика.

Запомни. $s = v \cdot t / 2$: звук летит туда и обратно. $340 \text{ м/с} = 0,034 \text{ см/мкс}$; время в микросекундах даёт расстояние в сантиметрах. ИК-дальномер переводят в сантиметры только по таблице или формуле.

Разбор для старта

Задача 5.1 Мотор прямо к плате

разминка · закон Ома, миллиамперы, почему мотор подключают через драйвер

Серёжа хочет подключить мотор робота прямо к выводу Arduino, без драйвера. На выводе 5 В, сопротивление обмотки мотора 12,5 Ом. Вывод безопасно отдаёт 20 мА. Какой ток потечёт через мотор в момент включения? Во сколько раз он больше допустимого? Ток дай в миллиамперах.

Дано: $U = 5 \text{ В}$; $R = 12,5 \text{ Ом}$; $I_{\text{доп}} = 20 \text{ мА}$.

Найти: I – ?, во сколько раз I больше $I_{\text{доп}}$.

Решение

Шаг 1. Закон Ома. Знаем напряжение и сопротивление, ищем ток: $I = U/R = 5:12,5 = 0,4 \text{ А}$.

Шаг 2. Переводим в миллиамперы. В одном ампере 1000 мА: $0,4 \cdot 1000 = 400 \text{ мА}$.

Шаг 3. Сравниваем. Оба тока теперь в одних единицах: $400:20 = 20$. Ток в 20 раз больше того, что может отдать вывод.

Для мотора 400 мА – вполне обычный ток при трогании с места. А вывод платы рассчитан на светодиод. В условии принято безопасное значение 20 мА; если сравнивать с пределом 40 мА из курса, получится в 10 раз. Вывод Серёжиной платы, скорее всего, сгорел бы в первую же секунду.

 **Типичная ошибка.** Сравнить 0,4 и 20 без перевода единиц и решить, что ток «в 50 раз меньше допустимого – можно подключать». Амперы и миллиамперы отличаются в 1000 раз, сравнивать можно только числа в одинаковых единицах.

Ответ: 400 мА; в 20 раз больше допустимого.

Своя задача по программе сферы, модуль 2

Проверь себя

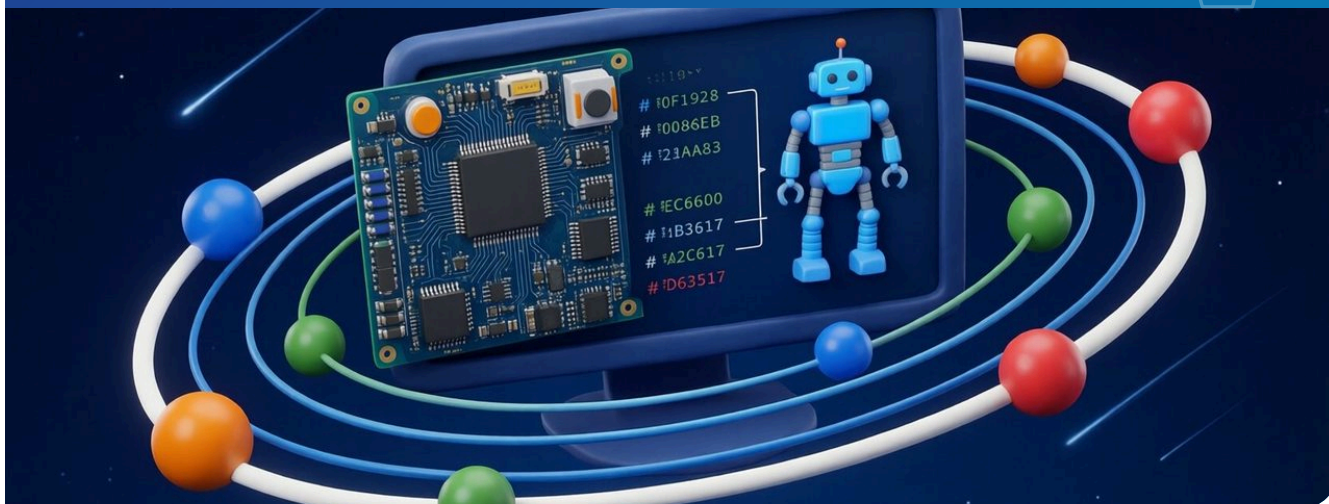
5.1. Через резистор 1,5 кОм течёт ток 4 мА. Какое напряжение на резисторе? Ответ в вольтах.

5.2. При трогании с места мотор берёт от батареи 6 В ток 0,75 А. Каково сопротивление его обмотки? Во сколько раз этот ток больше тех 20 мА, которые безопасно отдаёт вывод Arduino?

5.3. Красный светодиод (на нём оседает 2 В, ток не больше 20 мА) хотят зажечь от батарейки «Крона» на 9 В. Какой резистор нужно взять из ряда 220, 330, 390, 470, 560 Ом? Ответ в омах.

5.4. Два красных светодиода соединили последовательно, один за другим, и через резистор подключили к выводу Arduino (5 В). На каждом светодиоде оседает 2 В, ток через них должен быть 10 мА. Какое сопротивление нужно? Ответ в омах.

Программирование: Arduino и машины состояний



В главе 4 робот понимал команды «вперёд», «направо» и «пока впереди свободно». Теперь те же мысли запишем на языке C++, на котором программируют плату Arduino. Робот научится помнить числа, получит свои команды из нескольких строк, схему поведения из состояний и память о пройденном пути. Задача отбора по этой теме – короткий скетч и вопрос к нему: что напечатает монитор порта, куда поедет робот, в какой строке ошибка, в каком порядке поставить перемешанные строки. Ответ – число или строка из цифр.

НА ПАЛЬЦАХ

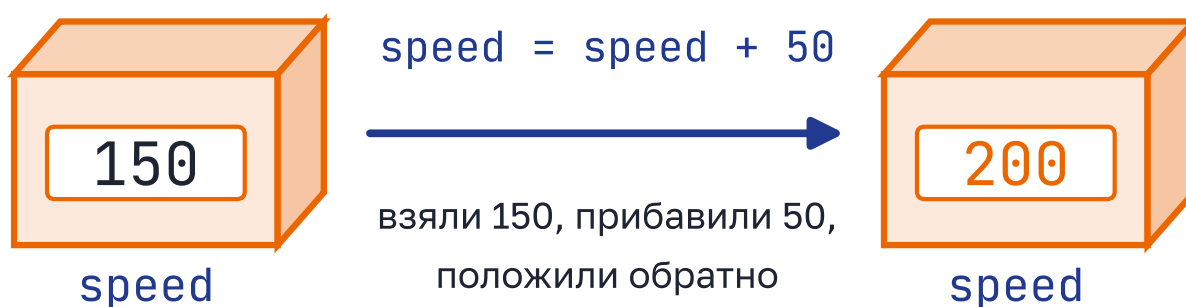


Рисунок 6.1 – Переменная – коробка с именем на боку

В коробке всегда лежит одно число. Кладёшь новое – старое пропадает.

НА ПАЛЬЦАХ



Рисунок 6.2 – Семь конфет на двоих: $7 / 2$ – сколько досталось каждому, $7 \% 2$ – сколько осталось

Резать конфету пополам C++ не умеет. Каждому по три, а одна остаётся в вазе.

Запомни. `=` в программе – «положи», а не «равно». Целое делённое на целое – дробь отбрасывается: $7 / 2 = 3$, $7 \% 2 = 1$. `println` переводит строку, `print` – нет. Счётчик, который должен копить, объявляй до `setup()`.

НА ПАЛЬЦАХ

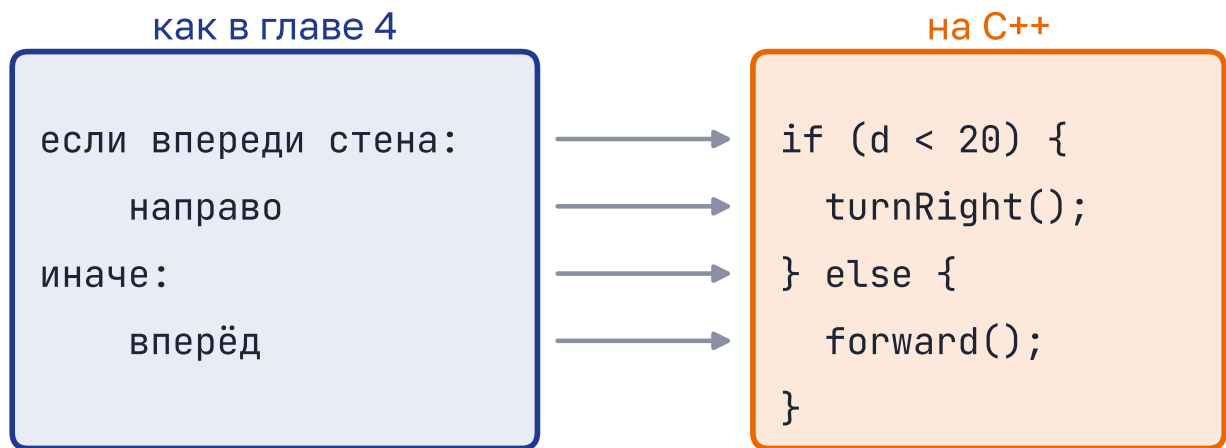


Рисунок 6.3 – Та же программа, что в главе 4, только словами C++

Слова другие, мысль та же. «Если» стало `if`, «иначе» – `else`, а вместо отступов команды собирают фигурные скобки.

Запомни. Сравнение – `=`, а `=` – «положи». `for (int i = 0; i < N; i++)` – тело выполнится N раз, `<=` – на один раз больше. `while` проверяет условие перед каждым кругом.

Запомни. Функцию описывают один раз: `void имя() { ... }`, вызывают по имени со скобками: `имя();`. Параметр передаёт в функцию число, `return` отдаёт число обратно. Функция моторов только переключает сигналы – как долго ехать, решает `delay` после неё.

Запомни. В каждый момент робот в одном состоянии. Переход срабатывает, только когда верно его условие; иначе робот остаётся, где был. Условия переходов проверяются только у стрелок, выходящих из текущего состояния. Время в состоянии – `millis() - t0`, где `t0` засекли при входе.

Запомни. Индексы массива – с нуля: у массива из N ячеек последняя – $[N - 1]$. `d[3]` – четвёртая ячейка. Обратный путь по записи: развернуться, пройти команды с конца и поменять местами «налево» и «направо».

Разбор для старта

Задача 6.1 Что покажет монитор порта

разминка · переменные, цикл for, целочисленное деление, Serial

Робот каждый круг цикла проезжает 25 см, а программа складывает путь в переменную. Что появится во второй строке монитора порта?

```
void setup() {
  Serial.begin(9600);
  int path = 0;
  for (int i = 0; i < 4; i++) {
    path = path + 25;
  }
  Serial.println(path);
  Serial.println(path / 3);
}

void loop() {
}
```

В ответ введи одно число.

Дано: начальное значение `path = 0`; в теле цикла `path = path + 25`; цикл `for` с `i = 0` и условием `i < 4`.

Найти: второе напечатанное число.

Решение

Шаг 1. Сколько раз выполнится тело цикла. `i` принимает значения 0, 1, 2, 3. При `i = 4` условие `4 < 4` неверно, цикл заканчивается. Значит, тело выполнится 4 раза.

Шаг 2. Следим за коробкой path . Было 0. После первого круга $0 + 25 = 25$, после второго 50, после третьего 75, после четвёртого 100. Первая строка монитора: 100.

Шаг 3. Вторая строка. `path / 3` – оба числа целые, поэтому дробная часть отбрасывается. $100 : 3 = 33,33\dots$, остаётся 33. `loop()` пустой, больше ничего не печатается.

Четыре круга по 25 см – это метр, а треть метра – 33 см с хвостиком. Хвостик C++ и выбросил.

 **Типичная ошибка.** Ввести 33,3 – забыть, что целое делится на целое без дробной части. Вторая частая ошибка – насчитать 5 кругов («от 0 до 4») и получить 125 и 41.

Ответ: 33.

Своя задача по программе сферы, модуль 1

Проверь себя

6.1. Какие два числа напечатает монитор порта? Запиши их через пробел.

```
void setup() {
  Serial.begin(9600);
  int path = 0;
  for (int i = 0; i ≤ 3; i++) {
    path = path + 15;
  }
  Serial.println(path);
  Serial.println(path / 7);
}

void loop() {
}
```

6.2. Программа повышает скорость робота, а потом печатает её в процентах от полной (255). Какие два числа появятся в мониторе порта? Запиши через пробел.

```
void setup() {
  Serial.begin(9600);
  int speed = 90;
  for (int i = 0; i < 3; i++) {
    speed = speed + 40;
  }
  Serial.println(speed);
  Serial.println(speed * 100 / 255);
}

void loop() {
}
```

6.3. Тот же робот, но при скорости 150 он едет со скоростью 15 см/с, а `turnLeft()` за 600 мс разворачивает его на месте на 120°. В `setup()` :

```
for (int i = 0; i < 3; i++) {  
    forward();  
    delay(2000);  
    turnLeft();  
    delay(600);  
}  
stopMotors();
```

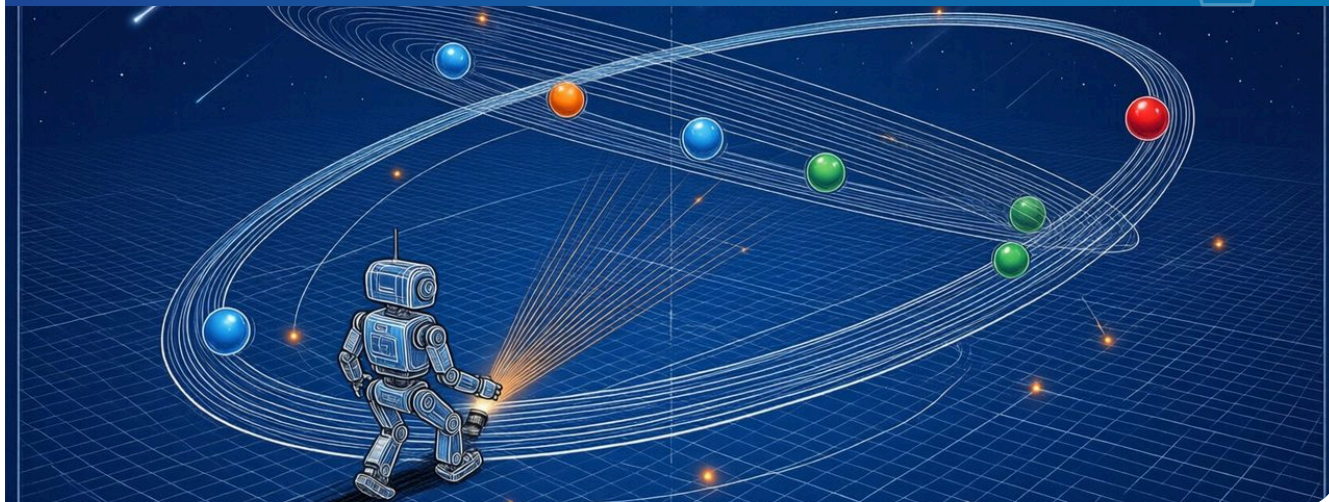
Какую фигуру нарисует робот? Какой длины будет след в сантиметрах?

6.4. Робот едет со скоростью 20 см/с, `turnRight()` за 500 мс поворачивает его на 90°. На этот раз команды стоят в `loop()` :

```
void loop() {  
    forward();  
    delay(2000);  
    turnRight();  
    delay(500);  
}
```

Сколько сантиметров проедет робот за первую минуту после включения? Где он окажется через 60 с после включения?

Движение по линии и регуляторы



Чёрная линия на белом поле – самая старая задача робототехники и одна из самых честных: робот либо едет по ней, либо нет. Видит линию робот двумя датчиками, а куда рулить, решает **регулятор** – короткое правило в программе. Их здесь два, «зиг-заг» и плавный пропорциональный, а в конце – зачем к нему приписывают ещё две буквы и получают ПИД. На отборе робота на столе не будет: по замерам ищут порог, по показаниям датчиков – куда он повернёт, по формуле – поправку и скорости моторов. Ответ – одно число или строка из цифр.

НА ПАЛЬЦАХ



Рисунок 7.1 – Пятно света датчика съезжает с белого поля на чёрную линию

Датчик смотрит на пятнышко поля размером в несколько миллиметров. Чем больше в этом пятне чёрного, тем меньше число. Половина пятна на линии – половина пути от «белого» числа к «чёрному».

Запомни. Датчик видит пятно, а не точку, поэтому его число меняется плавно. Два датчика стоят по бокам от линии, на прямой оба видят белое. Какой датчик «потемнел», с той стороны и линия.

НА ПАЛЬЦАХ

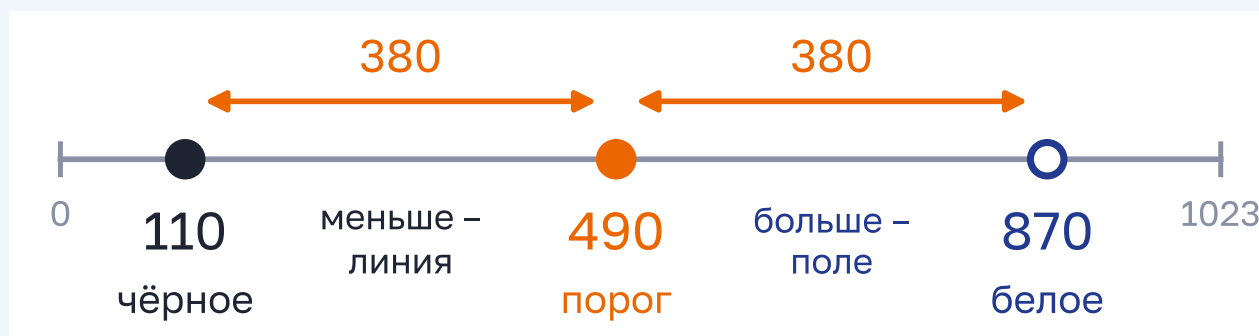


Рисунок 7.2 – Порог – посередине между белым и чёрным

От порога до «белого» и до «чёрного» числа одинаково далеко. Поэтому датчик ошибётся, только если число уплывёт больше чем на половину разницы.

Запомни. Порог = (белое + чёрное) : 2, и у каждого датчика он свой. Белое и чёрное – средние по нескольким замерам, на том поле и при том свете, где поедет робот. Разные датчики сравнивают в процентах белизны.

НА ПАЛЬЦАХ



Рисунок 7.3 – Четыре случая: оба датчика на белом, линия под левым, линия под правым, оба на чёрном

Робот поворачивает в ту сторону, где датчик увидел линию, – ловит её обратно между датчиками. А если потемнели оба, под ним поперечная полоса.

Запомни. Робот поворачивает в сторону медленного колеса и в сторону датчика, который увидел линию. Оба на белом – прямо, оба на чёрном – перекрёсток. Перекрёсток в программе проверяют первым, а считают по началу серии «оба на чёрном».

Запомни. Ошибка = левый – правый; поправка = $k \cdot \text{ошибка}$; левый мотор = база + поправка, правый = база – поправка. Отрицательная ошибка – робот поворачивает налево, положительная – направо. Скорости зажимают между 0 и 255.

Запомни. Влияние – это перелёт через линию. Его дают большой k , большая скорость и опоздание реакции. Маленький k не влияет, но на повороте вылетает. Подбирают k так: увеличивают, пока не начнётся влияние, и чуть уменьшают.

Запомни. ПИД = П (ошибка сейчас) + И (сумма ошибок) + Д (ошибка минус прошлая ошибка), каждая часть со своим коэффициентом. Д отрицательная, когда ошибка уменьшается, – она гасит влияние. И помогает, когда робот долго едет чуть сбоку.

Разбор для старта

Задача 7.1 Три замера на каждом цвете

разминка · калибровка, среднее, порог датчика линии

Перед заездом Вика калибрует датчик линии. Над белым полем она сделала три замера: 904, 896, 912. Над чёрной линией – тоже три: 118, 126, 122. Порог считают посередине между средним на белом и средним на чёрном. а) Какой получится порог? б) Какие из показаний означают, что под датчиком линия: 1) 480; 2) 530; 3) 212; 4) 760? В ответ введи порог, а затем номера показаний подряд.

Дано: белое – 904, 896, 912; чёрное – 118, 126, 122.

Найти: порог; какие показания – линия.

Решение

Шаг 1. Среднее на белом. Складываем три замера и делим на 3: $904 + 896 + 912 = 2712$, $2712 : 3 = 904$.

Шаг 2. Среднее на чёрном. $118 + 126 + 122 = 366$, $366 : 3 = 122$.

Шаг 3. Порог. $(904 + 122) : 2 = 1026 : 2 = 513$.

Шаг 4. Где линия. У этого датчика чёрное – маленькое число, значит, линия – это показания меньше 513. 480 – меньше, линия. 530 – больше, поле: робот у самого края, но ещё над белым. 212 – меньше, линия. 760 – больше, поле. Линию означают показания 1 и 3.

 **Типичная ошибка.** Перепутать направление: «больше порога – линия». Тогда выйдут показания 2 и 4 – всё наоборот. Перед сравнением посмотри, какое число у линии в замерах: здесь это 118–126, маленькие.

Ответ: а) 513; б) 13.

Своя задача по программе сферы, модуль 7

Проверь себя

7.1. Датчик линии над белым полем показал 820, 812 и 816, над чёрной линией – 204, 196 и 200. Найди порог.

7.2. В Tinkercad датчик линии изобразили фоторезистором, но подключили так, что над белым полем он показывает 150, а над чёрной линией – 870. Порог берут посередине. Какие из показаний означают линию: 1) 495; 2) 620; 3) 880; 4) 140? В ответ введи номера подряд.

7.3. Датчики робота откалибровали: левый – белое 900, чёрное 140; правый – белое 820, чёрное 180. Пороги посередине, действия те же, что в задаче 7.2 (1 – прямо, 2 – налево, 3 – направо, 4 – перекрёсток). Показания (левый, правый): А – 870 и 410; Б – 515 и 800; В – 530 и 505; Г – 150 и 160; Д – 880 и 810. Ответ – пять цифр для А, Б, В, Г, Д.

7.4. Робот едет по линии и записывает показания двух датчиков каждые 20 мс. Порог у обоих датчиков 500, линия – меньше порога. Перекрёсток – это когда оба датчика одновременно на чёрном. Поперечную полосу робот переезжает не мгновенно и успевает записать на ней от одной до трёх строк. Вот записи подряд (левый, правый): 850 и 820; 830 и 300; 820 и 790; 210 и 180; 190 и 160; 800 и 780; 240 и 830; 820 и 810; 150 и 170; 810 и 790; 790 и 260; 830 и 800; 170 и 190; 160 и 150; 180 и 200; 840 и 800. Сколько перекрёстков проехал робот?

Сигналы и обработка данных



Пока у робота есть питание, датчик выдаёт число за числом. Среди них попадаются странные: дальномер посреди комнаты вдруг показывает 0 см, датчик линии «видит» линию на чистом листе. Одному числу верить нельзя – смотреть надо на ряд целиком. Здесь ты узнаешь, как часто снимать показания, чтобы не проскочить полосу, и как сгладить ряд, чтобы соринка на поле не превратилась в лишний перекрёсток. На отборе такие задачи приходят рядом чисел или графиком: сгладь, найди выброс, посчитай события, найди период. Ответ – число или строка из цифр.

НА ПАЛЬЦАХ

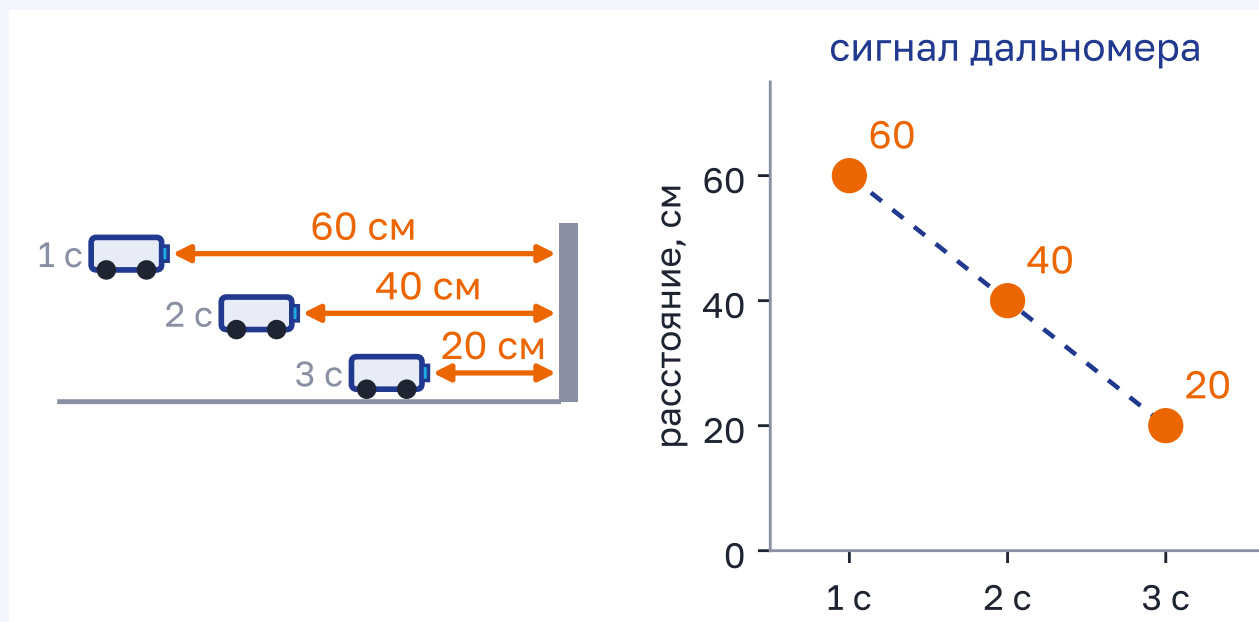


Рисунок 8.1 – Робот подъезжает к стене и раз в секунду меряет расстояние

Каждое измерение – точка на графике. Точки подряд складываются в сигнал, и по нему видно: робот приближается к стене, по 20 см в секунду.

НА ПАЛЬЦАХ

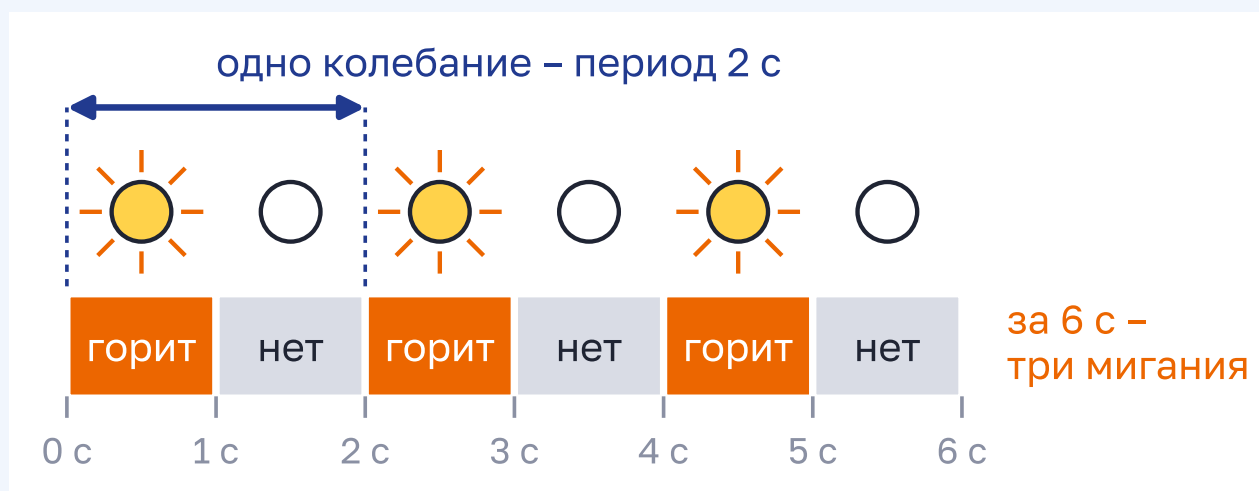


Рисунок 8.2 – Светодиод секунду горит, секунду не горит – и так по кругу

Кусок «горит + не горит» повторяется снова и снова. Его длина – период, здесь 2 с. За 6 секунд светодиод успел мигнуть три раза.

Запомни. Сигнал – величина, которая меняется во времени. Период T – время одного колебания, частота f – сколько колебаний в секунду, $f = 1 : T$. Размах – от минимума до максимума, амплитуда – его половина.

НА ПАЛЬЦАХ

смотрит каждые 6 мм



смотрит каждые 30 мм



→ робот едет

Рисунок 8.3 – Датчик смотрит на поле короткими взглядами

Между двумя взглядами робот успевает проехать какой-то путь. Если этот путь больше ширины полосы, полоса может проскочить незамеченной.

Запомни. Отсчёт – одно измерение. Промежуток между отсчётами $\Delta t = 1 : f_{\text{опр}}$, путь робота за это время $s = v \cdot \Delta t$. Наверняка на полосу попадёт столько отсчётов, сколько целых шагов в её ширине.

Запомни. Шум – мелкая дрожь во всех отсчётах. Выброс – одиночный скачок далеко от соседей. Выброс ищем сравнением с обоими соседями и допуском: сколько робот мог проехать между отсчётами, плюс запас.

Запомни. Скользящее среднее гасит дрожь, но размазывает выброс. Медиана убирает одиночный выброс целиком. Окно из k отсчётов даёт из n отсчётов $n - k + 1$ значений. Окно с центром стоит под средним отсчётом и не опаздывает, но ему нужны соседи с обеих сторон. Окно «назад» (так считает робот на ходу) опаздывает: на один отсчёт при окне из трёх, на два при окне из пяти.

Запомни. По уровню считаем пересечения порога, а не отсчёты за ним. От ложных срабатываний спасают выдержка, два порога и сглаживание. По периоду меряем время между началами соседних событий: $f = 1 : T$, скорость колеса $v = \pi \cdot D \cdot f$. Если меток на колесе несколько, оборот – это столько промежутков, сколько меток.

Запомни. Спектр показывает, каких колебаний в сигнале больше – медленных или быстрых. Полезное у датчика медленное, шум быстрый, поэтому фильтр низких частот чистит сигнал. Несущая – быстрое ровное мигание, огибающая – линия по её верхушкам; сообщение едет в огибающей.

Разбор для старта

Задача 8.1 Сколько миллиметров между взглядами

разминка · дискретизация, частота опроса, путь между отсчётами

Робот едет по полю со скоростью 0,25 м/с. В `loop` программа читает датчик линии и ждёт `delay(40)`; остальные строки выполняются так быстро, что их время не считаем. а) Сколько отсчётов в секунду снимает программа? б) Сколько миллиметров проезжает робот между двумя отсчётами? Ответ – два числа.

Дано: $v = 0,25$ м/с; $\Delta t = 40$ мс.

Найти: $f_{\text{опр}}$ – ?; s – ?

Решение

Шаг 1. Промежуток в секундах. $1 \text{ с} = 1000 \text{ мс}$, поэтому $40 \text{ мс} = 40 : 1000 = 0,04 \text{ с}$.

Шаг 2. Частота опроса. $f_{\text{опр}} = 1 : 0,04 = 25$ отсчётов в секунду. Проверка без дробей: в секунде 1000 мс, по 40 мс их помещается $1000 : 40 = 25$.

Шаг 3. Скорость в миллиметрах в секунду. $1 \text{ м} = 1000 \text{ мм}$, поэтому $0,25 \text{ м/с} = 0,25 \cdot 1000 = 250 \text{ мм/с}$.

Шаг 4. Путь между отсчётами. $s = v \cdot \Delta t = 250 \cdot 0,04 = 10 \text{ мм}$.

Сантиметр между взглядами – для полосы шириной 18 мм это впритык: $18 : 10 = 1,8$, наверняка на неё попадёт только один отсчёт. Если программа ждёт три тёмных отсчёта подряд (раздел 8.5), такой робот перекрёсток может не засчитать.

⚠ Типичная ошибка. Посчитать $0,25 \cdot 0,04 = 0,01$ и записать «0,01 мм». Скорость была в метрах в секунду, значит, и путь вышел в метрах: 0,01 м – это 10 мм. Вторая ошибка – решить, что `delay(40)` означает «40 раз в секунду». В скобках стоит пауза в миллисекундах, а не число отсчётов.

Ответ: а) 25; б) 10.

Своя задача по программе сферы, модуль 3

Проверь себя

8.1. Робот едет со скоростью 0,32 м/с, датчик читают каждые 20 мс. а) Сколько отсчётов в секунду? б) Сколько миллиметров робот проезжает между двумя отсчётами?

8.2. Робот едет по полю со скоростью 0,3 м/с. Поперёк пути нарисованы полосы шириной 15 мм. Программа засчитывает полосу, если увидела её не меньше трёх отсчётов подряд. С какой наименьшей частотой опроса (отсчётов в секунду) нужно читать датчик, чтобы на любую полосу наверняка пришлось хотя бы три отсчёта?

8.3. Робот медленно подъезжает к стене, дальномер меряет каждые 0,1 с. Показания в сантиметрах: 52, 50, 49, 47, 46, 260, 43, 41, 40, 0, 37, 35. Выброс – отсчёт, который отличается от обоих соседей больше чем на 10 см. Назови номера выбросов через пробел.

8.4. Робот отъезжает от стены, дальномер показывает (см): 20, 25, 30, 36, 18, 45, 50, 55, 97, 65. Допуск тот же, 10 см. Найди номера выбросов и замени каждый выброс средним его соседей. Ответ: два номера и два новых значения.

Ответы и решения

Сначала реши задачу сам и только потом сверяйся. Не сошлось – найди в решении шаг, на котором твой путь разошёлся с нашим.

К главе 1. Как устроен робот

1.1. Ответ: 425310.

Подсказка: температуру не измеряет ни одна деталь из списка – поэтому у работы Е стоит 0.

1.2. Ответ: 12468.

Подсказка: датчик сообщает контроллеру, что происходит вокруг; сервомотор, пищалка и драйвер сами ничего не измеряют.

1.3. Ответ: 50 об/мин.

1.4. Ответ: 45 зубьев.

Подсказка: ведомая должна крутиться в $120:40 = 3$ раза медленнее, значит, зубьев у неё в 3 раза больше.

К главе 2. Движение: скорость, путь, время

2.1. Ответ: 15 см/с.

2.2. Ответ: 140 см.

Подсказка: роботы сближаются на $20 + 30 = 50$ см за секунду; сначала найди, через сколько секунд они встретятся.

2.3. Ответ: 283 см.

2.4. Ответ: 5,0 оборота.

Подсказка: дан радиус, поэтому диаметр колеса – 7 см.

К главе 3. Координаты и углы: навигация робота

3.1. Ответ: (2; 4).

3.2. Ответ: (5; 7), робот смотрит влево.

Подсказка: после разворота робот смотрит вверх, и «вперёд 5» увеличивает у, а не уменьшает.

3.3. Ответ: налево на 45° .

Подсказка: сдвиги по х и по у одинаковые (по 3), значит, цель на диагонали вверх-вправо – курс 45° , а робот смотрит с курсом 90° .

3.4. Ответ: 9 отрезков.

Подсказка: задача обратная: за весь объезд робот повернется на 360° , по 40° в каждой вершине – $360:40$.

К главе 4. Алгоритмы и исполнители

4.1. Ответ: не выполнит: робот врежется на 5-й команде, стоя в клетке (1; 3).

4.2. Ответ: 4 команды (например, `вперёд 3`, `направо`, `вперёд 4`, `направо`).

Подсказка: в конце робот должен смотреть вниз, поэтому кроме поворота между двумя отрезками нужен ещё один.

4.3. Ответ: длина 3 (повтори 4: и в теле `вперёд 2`, `направо`); робот вернется в (1; 1) и будет смотреть вверх.

4.4. Ответ: 20 команд; клетка (6; 6), смотрит вверх.

К главе 5. Моторы, датчики и электричество

5.1. Ответ: 6 В.

Подсказка: здесь ищем напряжение: $U = I \cdot R$, ток – в амперах, сопротивление – в омах.

5.2. Ответ: 8 Ом; в 37,5 раза.

5.3. Ответ: 390 Ом.

Подсказка: нужно не меньше 350 Ом, а такого в ряду нет.

5.4. Ответ: 100 Ом.

Подсказка: на двух светодиодах вместе оседает 4 В, резистору остаётся 1 В.

К главе 6. Программирование: Arduino и машины состояний

6.1. Ответ: 60 8.

Подсказка: с условием `i ≤ 3` цикл делает 4 круга: $i = 0, 1, 2, 3$.

6.2. Ответ: 210 82.

Подсказка: $210 \cdot 100 = 21\,000$, а $21\,000 / 255 = 82,35\dots$, дробь отброшена.

6.3. Ответ: равносторонний треугольник со стороной 30 см, след 90 см.

Подсказка: три поворота по 120° – это полный оборот, 360° , поэтому фигура замкнется.

6.4. Ответ: 960 см; робот в точке старта и смотрит туда же, куда в начале.

Подсказка: один круг `loop()` длится 2,5 с, за 60 с их 24 – это 6 полных квадратов.

К главе 7. Движение по линии и регуляторы

7.1. Ответ: 508.

7.2. Ответ: 23.

Подсказка: у этого датчика линия – большое число, поэтому линия – показания больше порога 510.

7.3. Ответ: 32141.

7.4. Ответ: 3.

Подсказка: записей «оба на чёрном» шесть, но идут они тремя сериями; каждая серия – один перекрёсток.

К главе 8. Сигналы и обработка данных

8.1. Ответ: а) 50; б) 6,4 мм.

8.2. Ответ: 60.

Подсказка: здесь задача обратная – сначала найди шаг: чтобы на 15 мм наверняка пришлось три отсчёта, шаг должен быть не больше $15 : 3 = 5$ мм.

8.3. Ответ: 6 10.

8.4. Ответ: отсчёты 5 и 9; новые значения 40,5 и 60.

Всё, что может понадобиться под рукой во время решения: числа, которые используются в книге, перевод единиц, формулы и правила из глав, мини-шпаргалки по Arduino, Tinkercad, ПРИМС и таблицам, словарь и все «Запомни» из глав на нескольких страницах.

Шпаргалка: всё «Запомни» из глав

Глава 1. Как устроен робот

Запомни. Спорный случай «робот или нет» решай одним вопросом: **кто принимает решение?** Человек с пультом – значит, это управляемая машина. Программа по сигналам датчиков – робот. Мобильный робот вдобавок сам перемещается: на колёсах, гусеницах, ногах или винтах.

Запомни. Провод от вывода Arduino прямо к мотору – ошибка, даже если в схеме на экране так «короче». Контроллер только командует, а кормит мотор батарея. Между контроллером и мотором всегда стоит драйвер.

Запомни. Робот поворачивает в сторону того колеса, которое крутится медленнее (или стоит, или едет назад). Сам себя проверяй так: встань, вытяни руки в стороны – это колёса. Шагни вперёд только левой стороной тела – и повернёшься направо.

Запомни. $i = z_{\text{ведомой}} / z_{\text{ведущей}}$. Скорость ведомой = скорость ведущей : i . Во сколько раз медленнее – во столько же раз сильнее. Ступени – перемножаем. Каждое зацепление меняет направление вращения, паразитная шестерня скорость не меняет.

Запомни. Сила × плечо = сила × плечо. Лёгкий груз уравновесит тяжёлый, если висит дальше от опоры – во столько же раз дальше, во сколько он легче. Плечо меряют от опоры, а не от края рычага.

Глава 2. Движение: скорость, путь, время

Запомни. $s = v \cdot t$. Ищешь время – путь дели на скорость, ищешь скорость – путь дели на время. Средняя скорость – весь путь, делённый на всё время вместе с остановками; две скорости пополам не складывают.

Запомни. $1 \text{ м/с} = 100 \text{ см/с} = 3,6 \text{ км/ч}$. $1 \text{ об/с} = 60 \text{ об/мин}$. $1 \text{ с} = 1000 \text{ мс}$. Чем мельче единица, тем больше число: $0,3 \text{ м/с} = 30 \text{ см/с} = 300 \text{ мм/с}$.

Запомни. Один оборот колеса – путь πD . Оборотов нужно: путь, делённый на πD . Скорость робота – обороты в секунду, умноженные на πD ; обороты в минуту сначала делим на 60. В формулу идёт диаметр: дали радиус – удвой его.

Запомни. На графике «путь – время» скорость видна по крутизне линии: путь участка дели на его время. Полочка – стоянка. Сначала прочитай подписи осей и узнай, сколько стоит одна клетка.

Запомни. После «стопа» робот ещё проезжает тормозной путь, поэтому команду дают раньше на эту длину. Скорость падает (или растёт) равномерно – значит, в среднем робот едет с половиной полной скорости, и путь торможения или разгона равен $\frac{v}{2} \cdot t$.

Запомни. Колесо едет, пока трение не даёт ему проскользнуть. Мотор крутит сильнее, чем трение держит, – колесо буксует, и робот проезжает меньше, чем по оборотам. Недобор в процентах считают от пути по оборотам – от того, сколько робот должен был проехать.

Глава 3. Координаты и углы: навигация робота

Запомни. Точка (x; y): x – шаги вправо от начала отсчёта, y – шаги вверх. Сначала всегда x. Считай промежутки между линиями, а не сами линии.

Запомни. «Вверх–вниз–вправо–влево» – по карте, «направо–налево» – от носа робота. Курс: 0° вверх, 90° вправо, 180° вниз, 270° влево. Направо – прибавь угол, налево – вычти, вышло за 360 или ниже нуля – поправь на 360. В вершине правильной фигуры робот поворачивает на $360^\circ : n$.

Запомни. В журнале на каждую команду – своя строка. У строки с поворотом x и y переписываются без изменений, у строки «вперёд» меняется одно число из двух, а курс остаётся прежним.

Запомни. Масштаб $1 : k$: с плана на поле – умножь на k , с поля на план – раздели на k . Если масштаб не дан, а даны размеры одного и того же отрезка на плане и на поле, переведи их в одни единицы и раздели большее на меньшее.

Запомни. По линиям: сдвиг по x плюс сдвиг по y , оба без минуса. Маршрут через точки – сумма кусков, каждый от предыдущей точки. Напрямую – корень из суммы квадратов сдвигов, и это всегда не длиннее, чем по линиям.

Запомни. Разворот на месте: колёса едут по окружности диаметром колеса B . Путь колеса – $\pi B \cdot \alpha / 360$, угол колеса – $\alpha \cdot B / D$. Вокруг стоящего колеса путь вдвое длиннее.

Глава 4. Алгоритмы и исполнители

Запомни. Робот делает только то, что есть в его системе команд, и строго по порядку. Повороты меняют взгляд, а не клетку, и «направо» считаем от носа робота. Клетка $(x; y)$: сначала столбец, потом строка.

Запомни. повтори N : – тело выполняется N раз. пока условие: – тело выполняется, пока условие верно; проверка перед каждым повтором. Длина программы и число выполненных команд – разные числа: у повтори 5: с тремя командами в теле длина 4, а выполнено 15.

Запомни. В ветвлении выполняется одна ветка: под если – когда условие верно, под иначе – когда нет. В цепочке если – иначе если побеждает первое верное условие.

Запомни. Подпрограмму описывают один раз и вызывают по имени сколько угодно раз. Само описание ничего не двигает. Команд выполнено = (команд в подпрограмме) · (число вызовов).

Запомни. Правая рука: справа → прямо → слева → разворот, в таком порядке, и после каждого поворота – шаг. Левая рука – зеркально. Путь обратно: список развилок с конца, «направо» ↔ «налево».

Запомни. Трассировка – таблица «шаг – команда – клетка – куда смотрит», по строке на каждую команду. Циклы и подпрограммы при этом раскрываем. Ошибку ищи в первой строке, где «должно» и «получилось» разошлись.

Запомни. Клетка повторилась – робот зациклился и до финиша не доедет. Чтобы узнать, где он будет через N переездов: отними подъезд, то, что осталось, раздели на длину круга с остатком и от первой клетки круга отсчитай вперёд столько переездов, сколько вышло в остатке.

Глава 5. Моторы, датчики и электричество

Запомни. Нет замкнутого круга – нет тока. Вольты толкают, амперы текут, омы мешают. 1 A = 1000 мА, 1 кОм = 1000 Ом. На Arduino 5V – плюс, GND – минус.

Запомни. $I = U / R$, и ток в эту формулу идёт в амперах. Резистор для светодиода считают по напряжению, которое осталось: $(U_{\text{пит}} - U_{\text{св}}) / I$. Из ряда берут ближайший **больший** номинал.

Запомни. Мотор – только через драйвер: вывод платы даёт до 40 мА (безопасно – около 20), мотору нужны сотни. IN1 и IN2 задают направление (разные сигналы – крутится, одинаковые – стоит), EN – можно ли ехать и как быстро. Минус батареи и GND платы соединяют.

Запомни. analogWrite : от 0 до 255, только выводы с ~ . Значение = доля · 255. Не путай с analogRead , у него шкала от 0 до 1023 – о нём раздел 5.6.

Запомни. Время работы = ёмкость : ток; мА·ч делим на мА – получаем часы. Токи всех устройств складываем. Последовательно соединённые батарейки прибавляют вольты, но не мА·ч.

Запомни. `analogRead` – от 0 до 1023, это 0...5 В. Число в вольты: $N \cdot 5 : 1023$. Вольты в число: $U : 5 \cdot 1023$, дробную часть отбрасываем.

Запомни. Белое отражает много света, чёрное – мало. Порог – среднее между показаниями над белым и над чёрным. Какое число у линии больше, а какое меньше, проверяй по показаниям своего датчика.

Запомни. $s = v \cdot t / 2$: звук летит туда и обратно. 340 м/с = 0,034 см/мкс; время в микросекундах даёт расстояние в сантиметрах. ИК-дальномер переводят в сантиметры только по таблице или формуле.

Глава 6. Программирование: Arduino и машины состояний

Запомни. `=` в программе – «положи», а не «равно». Целое делённое на целое – дробь отбрасывается: $7 / 2 = 3$, $7 \% 2 = 1$. `println` переводит строку, `print` – нет. Счётчик, который должен копить, объявляй до `setup()`.

Запомни. Сравнение – `==`, а `=` – «положи». `for (int i = 0; i < N; i++)` – тело выполнится N раз, `s <=` – на один раз больше. `while` проверяет условие перед каждым кругом.

Запомни. Функцию описывают один раз: `void имя() { ... }`, вызывают по имени со скобками: `имя();`. Параметр передаёт в функцию число, `return` отдаёт число обратно. Функция моторов только переключает сигналы – как долго ехать, решает `delay` после неё.

Запомни. В каждый момент робот в одном состоянии. Переход срабатывает, только когда верно его условие; иначе робот остаётся, где был. Условия переходов проверяются только у стрелок, выходящих из текущего состояния. Время в состоянии – `millis() - t0`, где `t0` засекли при входе.

Запомни. Индексы массива – с нуля: у массива из N ячеек последняя – `[N - 1]`. `d[3]` – четвёртая ячейка. Обратный путь по записи: развернуться, пройти команды с конца и поменять местами «налево» и «направо».

Глава 7. Движение по линии и регуляторы

Запомни. Датчик видит пятно, а не точку, поэтому его число меняется плавно. Два датчика стоят по бокам от линии, на прямой оба видят белое. Какой датчик «потемнел», с той стороны и линия.

Запомни. Порог = (белое + чёрное) : 2, и у каждого датчика он свой. Белое и чёрное – средние по нескольким замерам, на том поле и при том свете, где поедет робот. Разные датчики сравнивают в процентах белизны.

Запомни. Робот поворачивает в сторону медленного колеса и в сторону датчика, который увидел линию. Оба на белом – прямо, оба на чёрном – перекрёсток. Перекрёсток в программе проверяют первым, а считают по началу серии «оба на чёрном».

Запомни. Ошибка = левый – правый; поправка = $k \cdot \text{ошибка}$; левый мотор = база + поправка, правый = база – поправка. Отрицательная ошибка – робот поворачивает налево, положительная – направо. Скорости зажимают между 0 и 255.

Запомни. Влияние – это перелёт через линию. Его дают большой k , большая скорость и опоздание реакции. Маленький k не влияет, но на повороте вылетает. Подбирают k так: увеличивают, пока не начнётся влияние, и чуть уменьшают.

Запомни. ПИД = П (ошибка сейчас) + И (сумма ошибок) + Д (ошибка минус прошлая ошибка), каждая часть со своим коэффициентом. Д отрицательная, когда ошибка уменьшается, – она гасит влияние. И помогает, когда робот долго едет чуть сбоку.

Глава 8. Сигналы и обработка данных

Запомни. Сигнал – величина, которая меняется во времени. Период T – время одного колебания, частота f – сколько колебаний в секунду, $f = 1 : T$. Размах – от минимума до максимума, амплитуда – его половина.

Запомни. Отсчёт – одно измерение. Промежуток между отсчётами $\Delta t = 1 : f_{\text{опр}}$, путь робота за это время $s = v \cdot \Delta t$. Наверняка на полосу попадёт столько отсчётов, сколько целых шагов в её ширине.

Запомни. Шум – мелкая дрожь во всех отсчётах. Выброс – одиночный скачок далеко от соседей. Выброс ищем сравнением с обоими соседями и допуском: сколько робот мог проехать между отсчётами, плюс запас.

Запомни. Скользящее среднее гасит дрожь, но размазывает выброс. Медиана убирает одиночный выброс целиком. Окно из k отсчётов даёт из n отсчётов $n - k + 1$ значений. Окно с центром стоит под средним отсчётом и не опаздывает, но ему нужны соседи с обеих сторон. Окно «назад» (так считает робот на ходу) опаздывает: на один отсчёт при окне из трёх, на два при окне из пяти.

Запомни. По уровню считаем пересечения порога, а не отсчёты за ним. От ложных срабатываний спасают выдержка, два порога и сглаживание. По периоду меряем время между началами соседних событий: $f = 1 : T$, скорость колеса $v = \pi \cdot D \cdot f$. Если меток на колесе несколько, оборот – это столько промежутков, сколько меток.

Запомни. Спектр показывает, каких колебаний в сигнале больше – медленных или быстрых. Полезное у датчика медленное, шум быстрый, поэтому фильтр низких частот чистит сигнал. Несущая – быстрое ровное мигание, огибающая – линия по её верхушкам; сообщение едет в огибающей.

Об авторе-составителе



Артём Осиновский

Методист и разработчик детского технопарка «Кванториум» (Новосибирск). Придумывает образовательные программы и сам делает для них цифровые инструменты: платформу смены «Миссия на Луну» для фестиваля «Технопром-2026», сайт с памятками НТО Юниоры для Новосибирской области. Эту книгу собрал для школьников области, чтобы к отбору можно было подготовиться самостоятельно – даже без кружка рядом с

домом.

Нашли ошибку – пишите на osinovsky.aa@kvantorium54.ru (рабочая почта): сфера, версия книги и страница. По другим вопросам и проектам – личная: artem081033@gmail.com.