

Программирование на Python с роботом РУБИ

Программа курса: 36 очных встреч по 120 минут (45 минут работы + 30 минут перерыва + 45 минут работы).

Возраст: 12–17 лет. **Уровень:** начальный; опыт Scratch полезен, но не требуется.

Учебный объём: 72 академических часа по 45 минут, или 54 астрономических часа. Перерывы составляют ещё 18 часов; полное время присутствия — 72 часа.

Группа: 6–10 человек; каждому нужен компьютер. Для встреч с роботом — один проверенный РУБИ на пару.

Ритм: при одной встрече в неделю — 36 учебных недель без учёта каникул.

Итог: личная программа на Python и работающий интерактивный сценарий с реальным РУБИ.

1. Идея курса

Подросток осваивает Python через короткие понятные задачи: вывод сообщений, числа, выбор, циклы, функции, коллекции, файлы и проверку ошибок. Периодически тот же алгоритм управляет светом и звуком РУБИ либо использует показания его датчика. Так результат кода становится виден на настоящем устройстве. Движение колёс применяется на подставке и только после отдельной проверки педагога. Курс учит прежде всего программированию; сборка электроники и разработка прошивки контроллера не являются задачами начинающих.

Где выполняется Python. Программа запускается на компьютере обучающегося. Показанный в материалах РУБИ контроллер Arduino Nano не предполагает запуска обычного Python непосредственно на нём. Для практики педагог заранее устанавливает на конкретный РУБИ проверенную служебную прошивку и выдаёт простой модуль `rubi.py`: Python передаёт короткие команды через USB-порт и получает ответы. Ученик видит и пишет свой алгоритм на Python; внутреннюю реализацию обмена в начале курса менять не нужно. Для домашних задач тот же учебный интерфейс имеет экранный тренажёр или заранее записанные показания. Это обеспечивает выполнение задания без домашнего робота.

Граница обязательного оборудования: базовая колёсная платформа, контроллер и USB, проверенный световой сигнал, звуковой сигнал, HC-SR04 и моторы с драйвером в фактической сборке. В курс не включены как обязательные Bluetooth, микрофон, PIR, матрица, датчик цвета, шагающая конфигурация и манипулятор: их наличие или совместимость нельзя предполагать для каждого экземпляра. Захват и другие надстройки могут стать темой свободного проекта только после отдельной проверки.

2. Цель и результаты

Цель: научить подростка писать, проверять и объяснять небольшие программы на Python, используя данные и действия физического робота РУБИ как наглядные задачи.

К концу курса обучающийся:

- пишет и запускает Python-программы с переменными, вычислениями, вводом, условиями, циклами, функциями, строками, списками и словарями;
- разбивает задачу на части, читает понятное сообщение об ошибке, проверяет обычный и граничный случаи;
- получает показание датчика через подготовленный интерфейс, проверяет его достоверность и принимает решение по заданному правилу;
- задаёт роботу безопасный световой или звуковой ответ, а на подставке — короткое движение с обязательной остановкой;
- сохраняет данные испытаний в файл и делает вывод по нескольким попыткам;
- представляет личный проект: запускает программу, показывает своё изменение, объясняет входные данные, решение и действие робота.

Не требуется помнить наизусть все команды или разбираться в электрической схеме. Оценивается понятный собственный алгоритм и его проверка, а не скорость печати.

3. Подготовка площадки и учебного интерфейса

До начала курса педагог составляет паспорт каждого экземпляра: ревизия платы, схема питания, реально установленные светодиод и звуковой модуль, модель дальномера, драйвер моторов, назначения выводов, пределы питания и штатная остановка. Готовая прошивка и `rubi.py` проверяются на каждом комплекте. Интерфейс для базовых заданий должен предоставлять понятные операции `connect`, `led`, `beep`, `distance_cm`, `stop`; для занятия с моторами — `drive_for` с ограничением времени. Эти имена являются **учебным контрактом**, а не утверждением о существующей заводской прошивке. Конкретные команды и распиновку педагог документирует после проверки сборки.

Обязательные испытания перед первым уроком с роботом: подключение и отключение USB; корректный ответ при недоступном порте; свет и звук; измерение известной дистанции и отдельный ответ при ошибке датчика; остановка после команды движения, обрыва связи и перезапуска программы. Моторы испытываются на подставке. Одновременное питание через USB и аккумуляторы допускается только после проверки схемы конкретного экземпляра. При отсутствии проверенной схемы моторное занятие и итоговые моторные проекты не запускаются до доработки оборудования и программы курса.

На работе работает пара. В первом блоке один ученик пишет и запускает программу, другой сверяет ожидаемый результат и следит за состоянием робота; во втором роли меняются. Каждый сохраняет **свой** файл и лично меняет, запускает и объясняет хотя бы одну команду. Педагог выделяет время каждой паре на физический комплект; просмотр демонстрации другой пары личную попытку не заменяет.

Правила площадки: перед переподключением питание снимает взрослый; дети не настраивают преобразователь и аккумуляторы; свободные колёса и механизмы проверяются на подставке; провода не попадают под колёса; длительность движения ограничена и доступен физический выключатель. Во время 30-минутного перерыва робот переводится в безопасное состояние, работа за компьютером прекращается.

4. Постоянный ритм встречи

Время от начала	Содержание
0–5 минут	Миссия и повторение одного знакомого приёма
5–13 минут	Короткий показ нового понятия на простом примере
13–35 минут	Личная программа: написать, запустить, увидеть результат
35–45 минут	Проверка обычного и ошибочного случая, сохранение файла
45–75 минут	Перерыв 30 минут вне рабочего места
75–80 минут	Возврат к задаче; при работе с РУБИ — смена ролей
80–110 минут	Практическая миссия с одним новым условием или данными робота
110–120 минут	Повторная проверка, объяснение и запись следующего шага

На сложных встречах педагог даёт готовый минимальный файл с уже настроенным соединением. Сначала ученик меняет одно правило, затем пишет собственную функцию. Усложнение предлагается после работающего базового результата. Домашняя работа добровольна и выполняется без робота.

5. Учебный план: 36 встреч

В каждой строке указан минимальный результат. **РУБИ** означает личную практику с физическим роботом; **экран** — Python на компьютере с текстовыми данными или тренажёром. Нагрузка каждого этапа — 12 академических часов; всего 72.

Этап 1. Первые команды и выбор — занятия 1–6

Занятие 1. РУБИ получает имя

Понятие: запуск Python, `print`, файл программы.

Первый блок: открыть готовый файл, вывести приветствие и имя робота; изменить текст самостоятельно.

Второй блок: составить три строки «имя — задача — ответ», сохранить и повторно запустить файл.

Результат: личная программа-приветствие; **экран**.

Занятие 2. Считаем энергию миссии

Понятие: переменные, целые числа, присваивание.

Первый блок: хранить условный запас энергии и расход на действие; вывести остаток.

Второй блок: заменить одно число и предсказать результат до запуска; исправить ошибку в имени переменной.

Результат: расчёт с двумя наборами данных; **экран**.

Занятие 3. Метры, сантиметры и секунды

Понятие: арифметика, единицы измерения, `int` и `float`.

Первый блок: перевести измеренную линейкой дистанцию в сантиметры; вычислить время условного маршрута.

Второй блок: сравнить две настройки и объяснить, почему число с единицей измерения нельзя путать с другим.

Результат: калькулятор параметров миссии; **экран**.

Занятие 4. Спросим оператора

Понятие: `input`, преобразование текста в число.

Первый блок: запросить имя и расстояние до мишени; вывести понятную инструкцию.

Второй блок: проверить ввод числа и пустой ответ на готовой заготовке с обработкой ошибки.

Результат: диалоговая программа, которая не завершается неожиданно от пустого ввода; **экран**.

Занятие 5. Если предмет близко

Понятие: сравнения, `if` и `else`.

Первый блок: решить по введённому расстоянию, нужен ли сигнал «близко».

Второй блок: проверить числа ниже, выше и ровно на пороге; изменить порог и объяснить результат.

Результат: программа с двумя ветвями и тремя проверками; **экран**.

Занятие 6. Первая встреча с настоящим РУБИ

Понятие: команда, результат, безопасное подключение.

Первый блок: изучить порт и штатную остановку; запустить проверенный пример `led` и изменить длительность света.

Второй блок: поменяться ролями; по условию из занятия 5 выбрать один из двух световых ответов.

Результат: каждый лично запускает два разных сигнала; **РУБИ**, неподвижный.

Этап 2. Повторение, функции и сигналы — занятия 7–12

Занятие 7. Повторяем команду без копирования

Понятие: `for`, `range`, отступы.

Первый блок: напечатать последовательность шагов миссии циклом.

Второй блок: сделать обратный отсчёт и сравнить 0, 1 и 5 повторений.

Результат: две программы с управляемым числом повторов; **экран**.

Занятие 8. Световой код РУБИ

Понятие: цикл в физическом действии.

Первый блок: подготовить короткий световой рисунок и предсказать число вспышек.

Второй блок: по очереди изменить число повторов и паузу; партнёр расшифровывает сигнал.

Результат: два различных световых сообщения; **РУБИ**, неподвижный.

Занятие 9. Функция как знакомая команда

Понятие: `def`, вызов, параметр.

Первый блок: вынести повторяющееся текстовое сообщение в функцию.

Второй блок: создать `report_distance(value)`, которая получает число и возвращает понятную фразу.

Результат: собственная функция, вызванная с двумя значениями; **экран**.

Занятие 10. Звуковые ответы

Понятие: функция, аргумент и допустимый диапазон.

Первый блок: сравнить на проверенном модуле короткие звуковые сигналы; ограничить длительность.

Второй блок: написать функцию «готов» или «внимание» со светом и звуком; роли меняются.

Результат: две узнаваемые реакции без непрерывного звука; **РУБИ**, неподвижный.

Занятие 11. Текстовые команды

Понятие: строки, `strip`, `lower`, проверка значения.

Первый блок: распознать слова «старт», «стоп», «помощь» независимо от регистра.

Второй блок: для неизвестного слова вывести безопасный ответ, а не случайную команду.

Результат: текстовый пульт с четырьмя проверенными вводами; **экран**.

Занятие 12. Робот отвечает оператору

Понятие: связать строку, условие и функцию.

Первый блок: сделать на экране таблицу «слово → сигнал»; проверить неизвестную команду.

Второй блок: передать команды света и звука через `rubi.py`; при «стоп» выключить сигнал.

Результат: безопасный текстовый пульт для неподвижного РУБИ; **РУБИ**.

Этап 3. Данные датчика и проверка — занятия 13–18

Занятие 13. Список измерений

Понятие: список, `append`, перебор.

Первый блок: хранить пять чисел дистанции и вывести каждое с номером.

Второй блок: найти минимум и максимум готовыми средствами; сравнить с ручным подсчётом.

Результат: краткий отчёт по пяти измерениям; **экран**.

Занятие 14. Измеряем настоящую дистанцию

Понятие: получение данных из подготовленного интерфейса.

Первый блок: на неподвижном РУБИ снять три показания дальномера для мишени на известном расстоянии.

Второй блок: сменить роли, снять ещё три показания и сравнить их с линейкой; ошибочный ответ отметить отдельно.

Результат: личный список показаний и одно объяснение разброса; **РУБИ**.

Занятие 15. Что делать, если датчик ошибся

Понятие: `None`, проверка допустимого диапазона, ранний выход из функции.

Первый блок: на готовых примерах отличить число от отсутствующего измерения.

Второй блок: написать `classify_distance`, которая возвращает «близко», «далеко» или «нет данных».

Результат: функция с проверкой трёх случаев; **экран**.

Занятие 16. РУБИ предупреждает о препятствии

Понятие: решение по показанию датчика.

Первый блок: проверить функцию занятия 15 на трёх записанных данных.

Второй блок: подключить дальномер и свет: близкий предмет вызывает сигнал, дальний — тишину, ошибка — отдельный сигнал; поменяться ролями.

Результат: три различные реакции без движения базы; **РУБИ**.

Занятие 17. Словарь правил

Понятие: словарь «состояние → сообщение».

Первый блок: сопоставить состояниям «готов», «близко», «ошибка» короткие фразы.

Второй блок: заменить длинную цепочку печати сообщений обращением к словарю; добавить неизвестное состояние.

Результат: изменяемая таблица правил; **экран**.

Занятие 18. Испытание у границы

Понятие: граничный случай и повторяемость.

Первый блок: предсказать реакцию для расстояний по обе стороны порога и около него.

Второй блок: выполнить пять проб с РУБИ, записать срабатывания и изменить только порог либо правило устойчивости.

Результат: таблица «ожидалось — получилось — исправил»; **РУБИ**, неподвижный.

Этап 4. Файлы, модули и состояния — занятия 19–24

Занятие 19. Дневник миссии

Понятие: запись текста и CSV-файла, `with open`.

Первый блок: сохранить пять учебных измерений в CSV с заголовком и единицей измерения.

Второй блок: открыть файл снова, подсчитать число строк и обнаружить пропущенное значение.

Результат: собственный файл данных и программа его чтения; **экран**.

Занятие 20. Дневник настоящего датчика

Понятие: разделение получения данных и сохранения.

Первый блок: записать пять ответов датчика РУБИ с отметкой «достоверно/ошибка».

Второй блок: партнёры меняются; сравнить два набора, назвать причину различий и сохранить личный файл.

Результат: два независимых журнала измерений пары; **РУБИ**, неподвижный.

Занятие 21. Свой модуль Python

Понятие: `import` и разбиение программы на файлы.

Первый блок: вынести `classify_distance` в отдельный файл и импортировать её.

Второй блок: создать простой тестовый файл с обычным, граничным и ошибочным значением.

Результат: два небольших файла, которые можно запустить повторно; **экран**.

Занятие 22. Команда и подтверждение

Понятие: запрос, ответ, тайм-аут.

Первый блок: разобрать на карточках путь команды от Python через USB к контроллеру и обратно.

Второй блок: отправить короткий световой сигнал, проверить ответ интерфейса; затем аккуратно обработать отключённый порт готовым блоком `try/except`.

Результат: программа сообщает о связи понятным текстом и завершает работу безопасно; **РУБИ**, неподвижный.

Занятие 23. Состояния робота

Понятие: переменная состояния, переходы и ограниченный цикл `while`.

Первый блок: описать на схеме состояния «жду», «вижу предмет», «ошибка датчика»; повторить чтение записанных значений циклом с ограничением числа шагов.

Второй блок: реализовать переходы на записанных показаниях и объяснить, почему при ошибке нельзя разрешать движение.

Результат: текстовый автомат состояний; **экран**.

Занятие 24. Настольный сторож

Понятие: соединение функций, датчика и состояния.

Первый блок: на неподвижном РУБИ проверить каждую реакцию отдельно.

Второй блок: запустить цикл «измерить → выбрать состояние → ответить светом/звуком», ограничив время работы; поменяться ролями.

Результат: работающий настольный сторож с отдельной реакцией на ошибку; **РУБИ**.

Этап 5. Моделирование и безопасное движение — занятия 25–30

Занятие 25. Маршрут сначала на бумаге

Понятие: последовательность команд и декомпозиция.

Первый блок: составить маршрут из коротких команд «вперёд — стоп — поворот — стоп» без робота.

Второй блок: смоделировать координату на экране и найти ошибку в порядке команд.

Результат: текстовый симулятор короткой миссии; **экран**.

Занятие 26. Функция движения с ограничением

Понятие: проверка аргумента и обязательная очистка в `finally`.

Первый блок: написать функцию, которая отвергает слишком большое время движения.

Второй блок: на экранном интерфейсе проверить, что после успешного действия и ошибки вызывается `stop`.

Результат: безопасная обёртка команд с тремя тестами; **экран**.

Занятие 27. Колёса на подставке

Понятие: наблюдение фактического действия программы.

Первый блок: педагог проверяет питание и подставку; ученики по очереди дают короткую команду одной стороне привода и `stop`.

Второй блок: испытать личную функцию занятия 26 с минимальным временем, поменяться ролями, объяснить направление и остановку.

Результат: каждый лично демонстрирует короткое движение и штатную остановку; **РУБИ**, колёса вывешены.

Занятие 28. Почему маршрут не всегда точен

Понятие: испытания, отклонение и одна изменяемая величина.

Первый блок: по данным трёх коротких проездов из подготовленной демонстрации вычислить разброс; не обещать точного позиционирования по времени.

Второй блок: на подставке сравнить два коротких времени включения моторов; записать наблюдения и остановить привод между пробами.

Результат: вывод о пределах управления по времени; **РУБИ** на подставке.

Занятие 29. Правила безопасного патруля

Понятие: приоритет условий.

Первый блок: смоделировать три входа: препятствие, ошибка датчика, свободный путь.

Второй блок: написать функцию решения, где препятствие и ошибка всегда дают `stop`, затем проверить таблицу случаев.

Результат: правило приоритета с автоматическими проверками на записанных данных; **экран**.

Занятие 30. Проверяем приоритет на РУБИ

Понятие: перенос проверенного правила на устройство.

Первый блок: на подставке связать датчик и короткое вращение колёс через проверенный интерфейс; сначала испытать «стоп».

Второй блок: проверить свободный путь, близкий предмет и ошибку датчика; при двух последних состояниях моторы остановлены.

Результат: три зарегистрированных испытания безопасного правила; **РУБИ**, колёса вывешены.

Этап 6. Собственный проект — занятия 31–36

Занятие 31. Придумываем миссию

Понятие: требования и минимальная версия.

Первый блок: выбрать проект «настойный сторож», «сигнальщик расстояния», «робот-реакция» либо согласованную идею из проверенных функций.

Второй блок: нарисовать цепочку «вход → решение → действие», записать три признака успеха и один безопасный случай ошибки.

Результат: личная карточка проекта и короткий план; **экран**.

Занятие 32. Первая рабочая версия

Понятие: минимальный результат и функции.

Первый блок: написать программу с одним входом и одним действием на экранном интерфейсе.

Второй блок: отделить получение данных от решения и ответа; проверить два набора входа.

Результат: личная версия 0.1 с понятной структурой; **экран**.

Занятие 33. Переносим миссию на РУБИ

Понятие: интеграция без одновременного усложнения.

Первый блок: отдельно проверить соединение, датчик и сигнал; исправить один обнаруженный сбой.

Второй блок: подключить личный алгоритм к физическому РУБИ, поменяться ролями; каждый сохраняет свою версию.

Результат: одна полная рабочая попытка проекта; **РУБИ**, неподвижный либо на подставке для согласованного моторного проекта.

Занятие 34. Пять испытаний

Понятие: повторяемость и журнал.

Первый блок: определить одинаковые условия пяти проб и ожидаемый результат.

Второй блок: лично провести пробы, записать успехи и ошибки, выбрать одну причину для исправления.

Результат: журнал пяти попыток и безопасная реакция на один предусмотренный сбой; **РУБИ**.

Занятие 35. Улучшаем и готовим объяснение

Понятие: точечная правка, версия и презентация.

Первый блок: изменить только одну причину ошибки, сохранить версии «до» и «после».

Второй блок: повторить испытание и подготовить рассказ на две минуты: цель, вход, правило, действие, ограничение.

Результат: улучшенная программа и краткая инструкция запуска; **экран**, при необходимости короткая повторная проба на РУБИ.

Занятие 36. Фестиваль Python и РУБИ

Понятие: демонстрация и объяснение собственного решения.

Первый блок: контрольный запуск и первая часть личных показов.

Второй блок: остальные показы; каждый отвечает на вопрос о своём коде и демонстрирует проверку ошибки или безопасную остановку.

Результат: исходный код, журнал испытаний и личная защита работающей миссии; **РУБИ**.

6. Контроль и итоговое задание

Контрольные точки после занятий 6, 12, 18, 24 и 30: ребёнок показывает свой файл, запускает его, объясняет одно условие или функцию и называет обнаруженную ошибку. Если результат не достигнут, в начале следующей встречи предлагается короткая повторная попытка; усложнение можно пропустить. Педагог фиксирует личный вклад в паре.

Итоговое задание: написать на Python интерактивную миссию для РУБИ с одним реальным входом (дальномер либо команда пользователя), одним правилом выбора и видимым или слышимым ответом робота. Обязательны минимум одна собственная функция, обработка отсутствующего измерения или соединения, безопасное завершение работы, пять испытаний и краткое объяснение личного кода. Моторы не обязательны; если используются, проект выполняется только на подставке по проверенной схеме.

Критерий	0	1	2
Личный код Python	Нет работающей личной версии	Работает с подсказкой	Ученик сам меняет и объясняет функцию и условие
Связь с РУБИ	Нет собственного запуска	Запуск с помощью педагога	Личный запуск и понятный ответ робота
Обработка ошибки	Ошибка приводит к опасному или непонятному действию	Ошибка замечена, реакция требует подсказки	Ошибка датчика/связи обработана безопасно и объяснена
Испытания	Нет журнала	Несколько проб без ясного вывода	Пять проб, записан результат и сделана одна правка
Представление	Ученик не объясняет решение	Объясняет часть по вопросам	Называет вход, правило, действие и ограничение

Максимум — 10 баллов. «Зачтено» — от 7 баллов при обязательном наличии личного запуска, безопасной реакции на выбранный сбой и исходного кода. Недостающий элемент дорабатывается с конкретной обратной связью и проверяется повторно. Для совместного робота оценочный лист заполняется отдельно на каждого ученика.

7. Методический комплект и границы готовности

Для проведения нужны 36 карточек занятий, экранные заготовки и набор данных для занятий без робота, проверенные учебная прошивка и rubi.py, паспорта подключений каждого экземпляра, контрольные программы и карта типичных ошибок. Этот документ задаёт содержание курса; он не утверждает, что USB-протокол и Python-модуль уже существуют в исходных материалах РУБИ. Их необходимо реализовать и испытать до первого потока. После пилотной группы педагог фиксирует, сколько времени уходит на личный запуск каждого участника, какие задания требуют сокращения и насколько устойчиво работает связь.

Основа программы: материалы базовой конструкции РУБИ в каталоге [РУБИ_патент_на_конструкцию](#), фотографии и конкурсная инструкция из каталога [Исходные материалы Родная игрушка](#), учебные проекты mBlock и существующая программа «Робототехника с РУБИ». Схемы и функции переносить на конкретный экземпляр только после сверки его ревизии. Язык Python изучается по установленной в классе версии и официальной документации <https://docs.python.org/>.