**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное   
образовательное учреждения высшего образования   
**«КУЗБАССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ Т. Ф. ГОРБАЧЕВА»**

Филиал КузГТУ в г. Прокопьевске

Кафедра информационных технологий,   
машиностроения и автотранспорта (ИТМА)

**Методические указания  
ДЛЯ выполнениЯ  
ПРАКТИЧЕСКОЙ РАБОТЫ**

по теме «Микроконтроллеры.Arduino» по дисциплине «Информационные технологии»  
для студентов очной формы обучения направления подготовки  
09.03.02 Информационные системы и технологии, профиль — 01   
«Системная интеграция и автоматизация информационных процессов»

Составитель: **А. С. Абрамович**

Рассмотрены и утверждены  
на заседании кафедры  
Протокол № X от X.X.2021 г.

Рекомендованы к печати  
учебно-методической комиссией

Протокол № X от XX.XX.2021 г.

Электронная копия находится  
в библиотеке филиала КузГТУ  
в г. Прокопьевске

Прокопьевск 2021

**Рецензент:**  
старший преподаватель кафедры информационных технологий, машиностроения и автотранспорта в филиале КузГТУ в г. Прокопьевске

**Иноземцев М.А.**

СОДЕРЖАНИЕ

[1 ЦЕЛЬ И ЗАДАЧИ 4](#_Toc91938841)

[1 ОСНОВЫ РОБОТОТЕХНИКИ 5](#_Toc91938842)

[Изучение основ программирования Arduino 6](#_Toc91938843)

[Среда разработки Arduino IDE 8](#_Toc91938844)

[Установка Arduino IDE 9](#_Toc91938845)

[Начало работы с Arduino IDE 10](#_Toc91938846)

[Подключение контроллера Arduino к ПК 11](#_Toc91938847)

[Базовые правила синтаксиса языка C\C++ 14](#_Toc91938848)

[Загрузка программы 14](#_Toc91938849)

[Мониторинг работы программы 15](#_Toc91938850)

[Переменные 15](#_Toc91938851)

[Функции 23](#_Toc91938852)

[Элементы объектно-ориентрованного программирования 24](#_Toc91938853)

[Задания на лабораторную работу 25](#_Toc91938854)

[Самостоятельная работа 32](#_Toc91938855)

[Контрольные вопросы 32](#_Toc91938856)

[СПИСОК ЛИТЕРАТУРЫ 33](#_Toc91938857)

# 1 ЦЕЛЬ И ЗАДАЧИ

**Цель:** закрепление, углубление и обобщение знаний, полученных обучающимися в процессе изучения дисциплины «Информационные технологии» по теме «Микроконтроллеры. Arduino». Обучающийся должен научиться применять эти знания, умения и навыки к решению конкретных задач в сфере информационных технологий и информационных систем.

**Задачи:**

1. ознакомление со средой разработки Arduino IDE, научиться осуществлять настройку программного обеспечения перед началом работы;
2. изучение основ программирования на платформе Arduino на представленных примерах схем и программ;
3. получение навыков самостоятельного изменения и дополнения программного кода в зависимости от поставленной задачи и условий.

# 1 ОСНОВЫ РОБОТОТЕХНИКИ

Появление первых микропроцессоров ознаменовало начало новой эры в развитии микропроцессорной техники. Наличие в одном корпусе большинства системных устройств, сделало микроконтроллер подобным обычному компьютеру. Раньше они назывались однокристальные микро- ЭВМ [[15](#_bookmark26)]. Чтобы собрать устройство и микроконтроллер, необходимо знать основы схемотехники, устройство и работу конкретного процессора, уметь программировать на ассемблере и изготавливать электронную технику. В настоящее время, все изменилось. Сейчас существует такое устройство, как проект Arduino.

Arduino представляет собой наборы, состоящие из готового электронного блока и программного обеспечения. Электронный блок — это печатная плата с микроконтроллером и элементами, которые необходимы для работы. Вторая часть — это программное обеспечение для создания программ, включающее в себя простую среду разработки и язык программирования C/C++ [[15](#_bookmark26)].

Робототехника — область науки и техники, ориентированная на создание роботов и робототехнических систем [[4](#_bookmark16)]. Робототехника возникла на основе мехатроники и кибернетики подразумевает знание механики, электроники, программирования. Является универсальным инструментом для образования.

Предмет робототехники — это создание и применение роботов и других средств робототехники различного назначения. Возникнув на основе кибернетики и механики, робототехника в свою очередь породила новые направления развития и самих этих наук. Для кибернетики это связано, прежде всего, с интеллектуальным управлением, которое требуется для роботов, а для механики с – многозвенными механизмами типа манипуляторов [20].

Робототехника является одним из важнейших направлений научно- технического прогресса (НТП), в котором проблемы механики и новых технологий соприкасаются с проблемами искусственного интеллекта [[3](#_bookmark15)]. В настоящее время проблемы робототехники рассматриваются более масштабно. Происходит внедрение в учебный процесс, организовываются соревнования, конкурсы, обмен новыми идеями, знаниями, технической информацией.

Современное общество старается внедрить роботов в повседневную жизнь путем применения роботов в различных сферах жизни, а так же заменить, усовершенствовать и модернизировать различные процессы жизни. А это значит, что люди, обладающие знаниями робототехники, программирования остаются востребованными на рынке труда.

Следовательно, вопрос внедрения робототехники актуален.

Применяются различные специальные робототехнические комплексы [[17](#_bookmark28)]:

* Lego Mindstorms. Специальный конструктор нового поколения;
* Конструктор Fischertechnik. Данный конструктор является развивающим. Он подходит как для детей, так и для подростков и студентов;
* Scratch Board;
* Arduino;
* Конструкторы УМКИ. Такие модули оснащены микропроцессором, а также наборами датчиков.

# Изучение основ программирования Arduino

С течением времени интерес к программированию продолжает расти, так как специальность программиста является очень востребованной не только в России, но и во многих других странах.

Программа — это описание процесса обработки информации. При выполнении программы рассчитывается совокупность выходных значений исходя из совокупности переменных или постоянных входных значений. Цель выполнения программы — сбор данных либо получение отклика на входные значения [[18](#_bookmark29)].

Программа состоит из строк текста. Каждая строка содержит один или несколько арифметических или управляющих операторов.

Компьютерная программа — это четко формализованный план, состоящий из команд для контроллера (системы принятия решений). Контроллер поочередно читает команды и исполняет [[10](#_bookmark21)]. Команды внутри любого цифрового устройства, например, Arduino, закодированы нулями и единицами, а называется это *двоичным представлением чисел*, то есть вся информация перед поступлением в контроллер перекодируется из привычной для нас десятичной системы счисления в двоичную. Таким образом, при выполнении логических или арифметических операций контроллер сравнивает, делит, вычитает, выполняет прочие действия именно над двоичными числами. Эти числа хранятся в последовательных ячейках памяти, имеющие определенные адреса.

При поступлении питания на контроллер Arduino автоматически начинается выполнение той программы, которая была в него загружена, если же программа отсутствует или написана некорректно, то происходит сбой, который либо останавливает выполнение команд, либо приводит к *зависанию* программы. Номер выполняемой программа хранится в специальной ячейке памяти, которая называется *счетчиком команд* [[10](#_bookmark21)]. Этот номер изменяется на следующий при выполнении арифметической операции, но может измениться и на любой адрес, если выполнялась логическая команда, результатом которой стал переход на некоторый пункт плана, отличный от следующего по порядку.

Алгоритмом называют упорядоченную определенным образом последовательность инструкций программы [[18](#_bookmark29)]. Алгоритм, не имеющий логических блоков и выполняемый строго по пунктам от начала и до конца, называется линейным (рисунок 1).

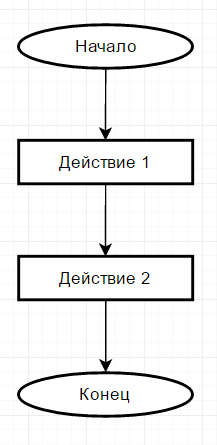


Рисунок 1. Линейный алгоритм.

Если алгоритм после достижения последнего пункта подразумевает бесконечное повторение, его называют *циклическим* (рисунок 2).

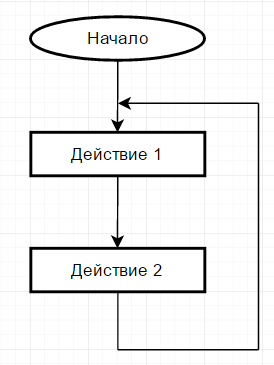


Рисунок 2. Циклический алгоритм.

При наличии в алгоритме анализа некоторого значения и принятия решения в результате такого анализа, этот алгоритм называется *разветвляющимся* (рисунок 3).

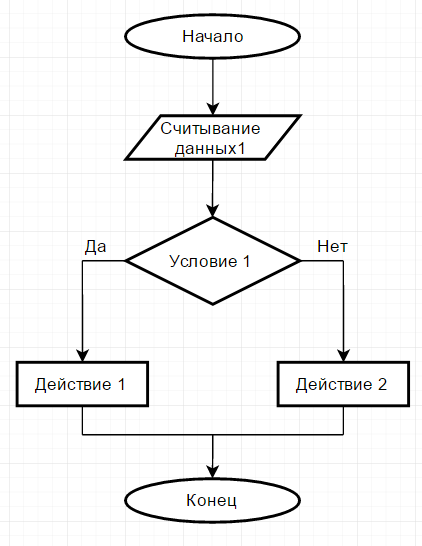


Рисунок 3. Разветвляющийся алгоритм.

При изучении темы обучающимся предлагается разработать и проанализировать алгоритм задачи.

Алгоритмы можно записывать и в простой текстовой форме: Действие 1.

Действие 2.

А для разветвляющегося и циклического алгоритмов:

Считывание данных о наличии каких-либо изменений в окружающей среде с датчика.

Если изменения есть, то перейти на пункт 4. Действие 2.

Перейти на пункт 6. Выполнение действия 1. Перейти на пункт 1.

Текстовая форма алгоритма более похожа на компьютерную программу, но визуально воспринимается хуже, в ней труднее найти логические ошибки, особенно если переходов много.

Для того, чтобы создать компьютерную программу, необходимо ясно понять, что она должна делать и главное, как она это будет делать. Если некоторые вопросы вами еще не проработаны, а вы уже пишете серьезную программу, то, скорее всего, она не будет работать так, как вы рассчитываете. Поэтому нужно идти от простого к сложному, разбивать программы на маленькие простые блоки, добиваться их работоспособности, а только затем собирать из них полую программу.

# Среда разработки Arduino IDE

В основу языка программирования, используемого в проектах Arduino, положен C++ — один из самых широко используемых языков программирования, поддерживающий как работу с низкоуровневыми командами, так и построение сложных объектов [[10](#_bookmark21)]. Программирование контроллеров Arduino удобно осуществлять в специальной среде Arduino IDE, поскольку в нее включен основной функционал для работы с ними.

# Установка Arduino IDE

Изучать основы программирования на платформе Arduino можно как на линейке физических устройств, так и в виртуальной онлайн среде Electronics Lab, предоставляемой компанией Autodesk на сайте [https://circuits.io](https://circuits.io/).

Arduino IDE можно скачать в Интернете по адресу основного сайта проекта: [www.arduino.cc/en/Main/Software](http://www.arduino.cc/en/Main/Software), программное обеспечение свободно распространяется и для скачивания нужно выбрать в разделе **Download the Arduino IDE** вариант **Windows** Installer (рисунок 4), а затем **JUST DOWNLOAD** (рисунок 4,5).

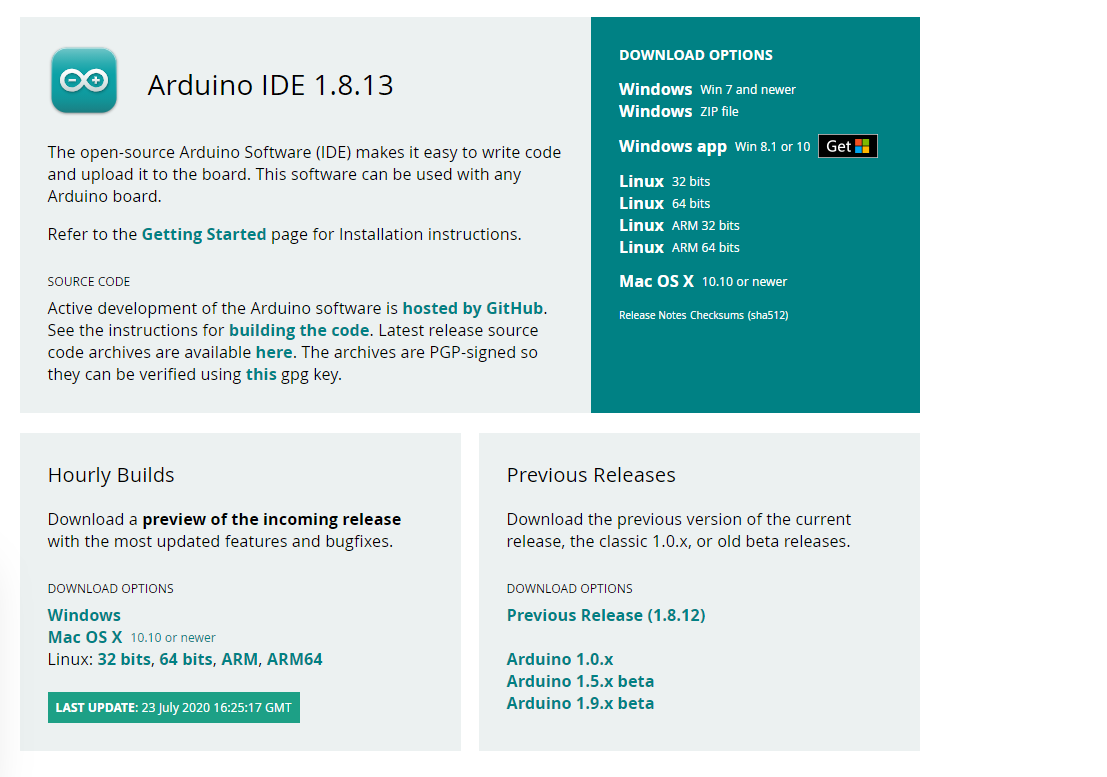


Рисунок 4. Окно скачивания программы Arduino IDE.

Рисунок 5. Кнопка загрузки установочного файла Arduino IDE.

На компьютер необходимо инсталлировать программу Arduino IDE

* в настоящий момент это файл *arduino-1.8.2-windows.exe*. Его надо запустить на выполнение с административными полномочиями, принять условия лицензии GNU LESSER GENERAL PUBLIC LICENSE и согласиться с предложенным вариантом установки.

На все предупреждения Windows в процессе установки следует отвечать утвердительно (продолжать установку). Когда установщик предложит установить драйверы порта, также ответить утвердительно.

# Начало работы с Arduino IDE

Среда разработки имеет интуитивно понятный русскоязычный интерфейс. При запуске установленной Arduino IDE откроется окно (рисунок 6), в котором уже содержится заготовка программы. Она состоит из двух функций: *setup* и *loop*. Функция setup содержит команды, выполняемые одни раз при включении Arduino, — это установка номеров портов ввода/вывода для управления мониторами и установки скорости обмена данными между Arduino и компьютером. Функция *loop* выполняется бесконечное число раз — до тех пор, пока мы не отключим питание, фактически она зациклена, алгоритмически это изображено на рисунке 7.

Интерфейс позволяет реализовать ранее изученные базовые алгоритмы.

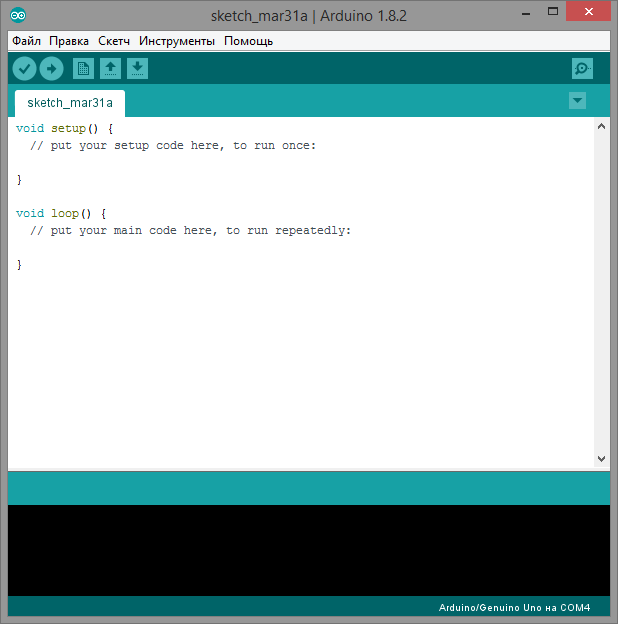


Рисунок 6. Окно Arduino IDE.

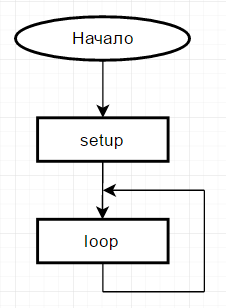


Рисунок 7. Типовой алгоритм программы Arduino IDE.

# Подключение контроллера Arduino к ПК

Возможности среды позволяют в зависимости от контроллера

Arduino использовать, разные USB-кабели (рисунок 8) [[1](#_bookmark14)]:

* + для Arduino UNO R3 и Mega — это кабель с разъемом под USB- принтер с одной стороны и стандартным USB-разъемом с другой;
  + для контроллера Nano требуется кабель с разъемом mini-USB;
  + для Arduino Micro — это micro-USB, а для программирования контроллера Arduino Mini и Pro Mini потребуется программатор, так как у него отсутствует стандартный интерфейс для подключения его к компьютеру.



Рисунок 8. Варианты USB-кабелей: а) USB для Arduino UNO и Mega; б) кабель для

Arduino Nano; в) micro-USB для Arduino Mini.

Некоторые контроллеры требуют для своей работы нестандартный драйвер. Они требуют установки отдельного драйвера, не входящего в комплект Arduino IDE, — его название *ch341ser.exe*. Найти его не трудно, после установки драйвера проблема будет решена.

Контроллер подключается к ПК через USB интерфейс и устанавливает связи между ним и оболочкой Arduino IDE. Для этого нужно задать номер порта, к которому подключен контроллер (рисунок 9). Если портов много, и найти нужный сложно, рекомендуется запомнить все имеющиеся, а затем физически отсоединить Arduino от кабеля, и снова проанализировать список портов, — тот, который исчез, и есть нужный. После подключения необходимо выбрать нужный порт — для этого нужно установить соответствующий ему флажок. Иногда для появления порта в списке требуется некоторое время, за которое операционная система компьютера анализирует и проверяет подключенное устройство, так что следует немного подождать до завершения этого процесса.

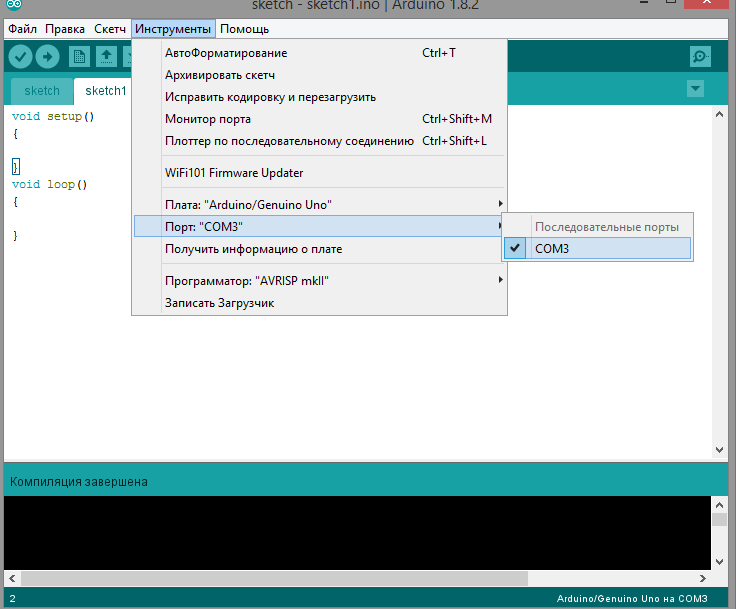


Рисунок 9. Выбор нужного порта.

Затем учащиеся выбирают тип контроллера Arduino. На рисунке 10 можно видеть, что выбран Arduino/ Genuino Uno.

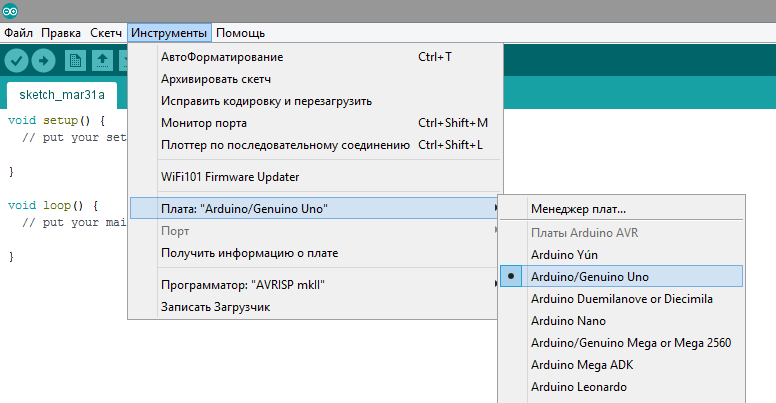


Рисунок 10. Выбор типов контроллера Arduino.

Если контроллер выбран системой автоматически, необходимо проверить правильность этого выбора. Для некоторых контроллеров необходимо еще выбрать подвид микроконтроллера, на котором реализована плата Arduino. Название микроконтроллера можно найти на самой его микросхеме — это, как правило, самая большая микросхема платы и расположена она в ее центре.

# Базовые правила синтаксиса языка C\C++

Прежде чем начать изучение языка C\C++, необходима интегрированная среда разработки — IDE.

Язык С\С++ представляет собой набор команд. Этот самый набор команд иными словами называется кодом или исходным кодом.

# Загрузка программы

Написанная программа в Arduino IDE загружается в контроллер нажатием кнопки со стрелкой вправо  (см. рисунок 6). Оболочка проверит программу на наличие ошибок, а затем переведет ее в двоичный код данных и команд выбранного микроконтроллера и запишет в Arduino.

Если после появления слова **Загрузка** в нижней строке окна Arduino IDE замигали светодиоды TX и RX на плате Arduino, то загрузка программы в плату началась успешно. Если после этого фраза **Загрузка завершена,** не появилась, то вероятнее всего неверно выбран тип платы Arduino. Если же диоды TX и RX не замигали вообще, то проблемы заключаются в выборе порта платы Arduino.

Если все пойдет штатно, появится надпись, **Загрузка завершена**, и программа автоматически начнет выполняться. Если загружена пустая программа, то ничего происходить не будет, — пустой код будет бесконечно повторяться, пока к плате подключено электропитание. Сказанное выше можно продемонстрировать на примере: заставим Arduino мигать встроенным светодиодом на 13-м порту (рисунок 11). Диод — это электронный элемент, который пропускает электрический ток, только в одном направлении, а светодиод — это диод, который начинает светиться при протекании по нему тока. На плате контроллера Arduino, как правило, уже установлен один светодиод — именно на 13-м порту (ножке).

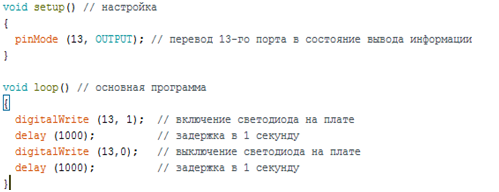


Рисунок 11. Программа мигания светодиода.

Цифровые порты могут получать и передавать информацию только в виде последовательности нулей и единиц. В приведенном примере как раз и передается на порт 13 подобная последовательность — визуально это отслеживается по морганию светодиода на плате контроллера. Когда светодиод горит, на порту высокое электрическое напряжение 5В, а когда потушен — низкое (0В).

# Мониторинг работы программы

Мониторинг программы позволяет выполнить отладку устройства. Для того чтобы контроллер мог общаться с компьютером во время выполнения программы записывается в функции setup команда *Serial.begin(9600)*, которая указывает, с какой скоростью происходит обмен информацией с ПК. Для обмена информацией требуется, чтобы приемник и передатчик осуществляли обмен на одинаковой скорости, а так

как контроллер Arduino достаточно медленный, то наиболее подходящей для обмена с ним является скорость 9600 бод. Обмен информацией на более высоких скоростях также возможен, но задействует большее количество ресурсов контроллера, при этом чаще происходят прерывания текущей программы, и основной код может выполняться медленнее.

Команда *Serial.println()* передает с контроллера на ПК значение, указанное в скобках. Здесь это значение, возвращаемое функцией *millis ()*, которая возвращает время работы в миллисекундах. Результат работы приведен на рисунке 12. Для получения на экране компьютера полученных результатов после запуска программы потребуется открыть вкладку **Инструменты | Монитор порта** и скорректировать скорость порта (9600 бод) — в окне **Монитор порта** скорость задается в правом нижнем углу выбором из всплывающего списка.

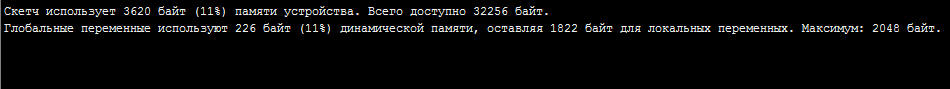


Рисунок 12. Протокол загрузки программы.

# Переменные

Важной составляющей синтаксиса языков программирования являются переменные. Переменные — это области памяти, имеющие имя, которое иначе называют идентификатором. До начала работы с переменными их нужно объявить, как и в любой другой среде программирования, например Turbo Pascal. В языке С/C++ все переменные должны быть объявлены. Это означает, что, во-первых, в начале каждой программы или функции должен быть приведен список всех используемых переменных, а во-вторых, указан тип каждой из них [[14](#_bookmark25)].

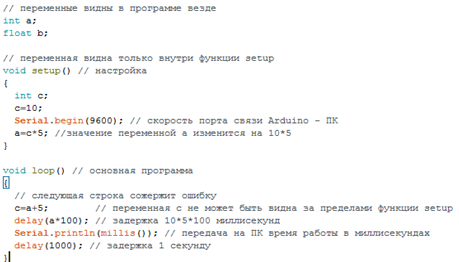


Рисунок 13. Определение переменных. Зоны видимости. Сообщение об ошибке.

Рисунок 13 в демонстрационных целях содержит специально добавленную в него ошибку объявления переменной, и при попытке его компилировать об этой ошибке будет выведено сообщение: **‘c’ was not declared in this scope**.

Переменные могут отличаться по типу данных, для хранения которых созданы (таблица 1).

Таблица 1 —

Типы данных.

| **Обозначение в**  **программе** | **Название** | **Принимаемые**  **значения** | **Размер в**  **памяти** |
| --- | --- | --- | --- |
| boolean | Логический | True/false, 1/0 | 1 байт |
| char | Символьный | -128/+127 | 1 байт |
| byte | Короткое  беззнаковое | 0-255 | 1 байт |
| int | Целое число | -32768/32767 | 2 байта |
| long | Длинное число | -2147483648/214748367 | 4 байта |
| float | Число с плавающей точкой | -3,4028235·1038/3,4028235·1038 | 4 байта |

При работе с переменными следует понимать, что они не являются идеальным хранилищем информации — так, например, целочисленные переменные могут переполняться. Это происходит в тех случаях, когда значение, которое следует записать в переменную, больше максимально возможного для этого типа данных. Переменные с плавающей точкой подвержены другой проблеме — они округляют свои значения при сложении большого числа с малым. Так же Arduino IDE не всегда корректно работает с преобразованием типов данных.

На следующем примере можно продемонстрировать небольшую программу, осуществляющую получение данных от компьютера через порт ввода/вывода (рисунок 14).

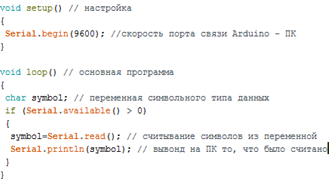


Рисунок 14. Получение данных от компьютера через порт ввода/вывода данных.

Скомпилировав и загрузив эту программу, можно ввести в верхней части окна **Монитор порта** слова «Hello, World!» и **отправить** их на порт. Программа отправит сообщение обратно на ПК посимвольно (рисунок 15).

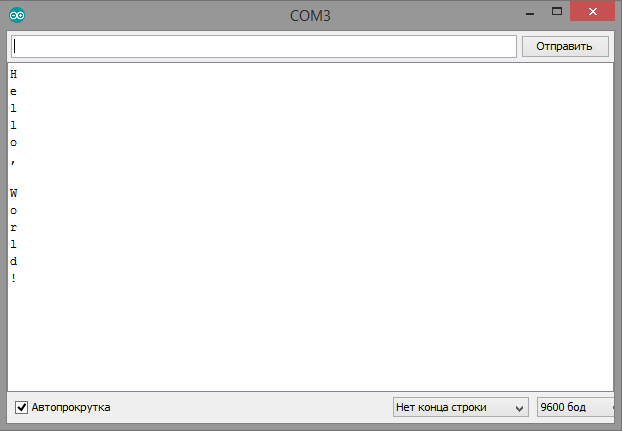


Рисунок 15. Вывод сообщения на ПК.

* + 1. Условные операторы, операторы выбора, операторы циклов Ниже приведено описание и реализация ранее изученных учащимися

алгоритмов: условных, операторов выбора и циклов.

Оператор *if… else* графически, в виде блок-схемы алгоритма, представлен на рисунке 16, а в виде фрагмента кода — на рисунок 17.

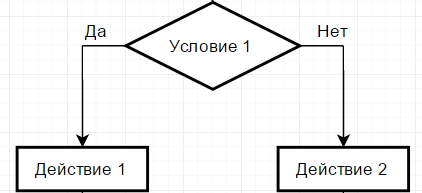


Рисунок 16. Представление условного оператора.

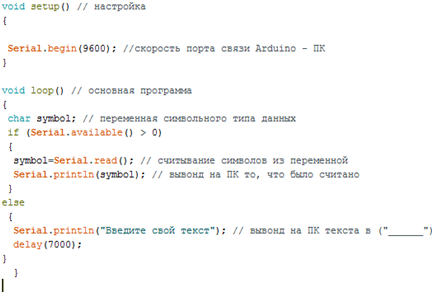


Рисунок 17. Применение оператора *if… else* при отправке сообщения на порт.

Реализация данного фрагмента программы предлагает пользователю ввести свой текст каждые 7 секунд.

Следующий пример — программа, которая изменяет частоту моргания встроенного светодиода на 13-м порту (рисунок 18) в зависимости от посланной с компьютера команды. Код программы представлен на рисунке 19. Блок-схема алгоритма это программы представлена на рисунке 20.

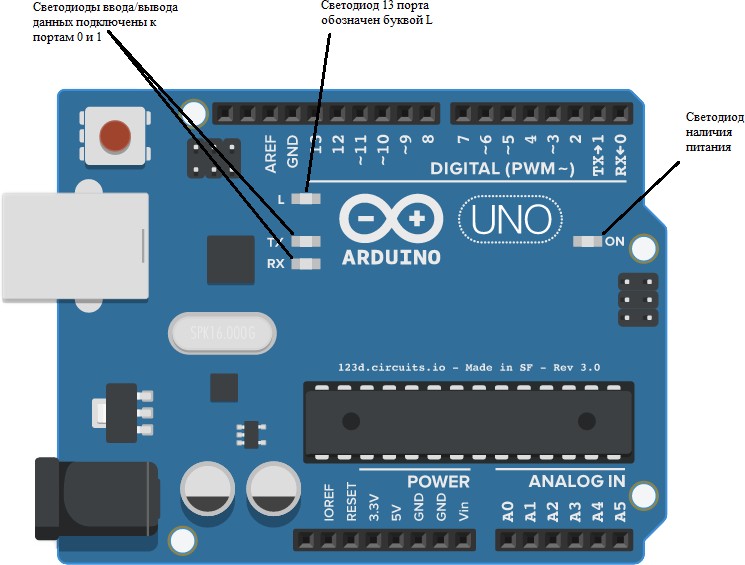


Рисунок 18. Светодиоды на плате Arduino UNO.

Программа не завершается, пока есть электропитание, в ней присутствуют три условных блока: один проверяет наличие данных на порту ввода/вывода, а последующие два сравнивают полученный символ с условиями, в зависимости от которых изменяют величину переменной *time\_pick*, задающую время задержки при мигании светодиода.

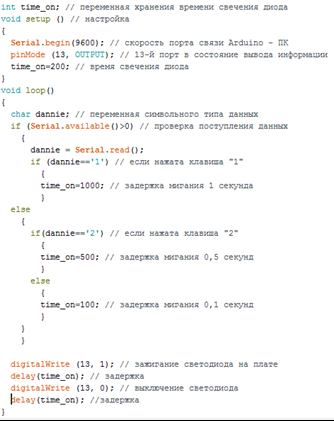


Рисунок 19. Программа управления миганием светодиода с ПК.

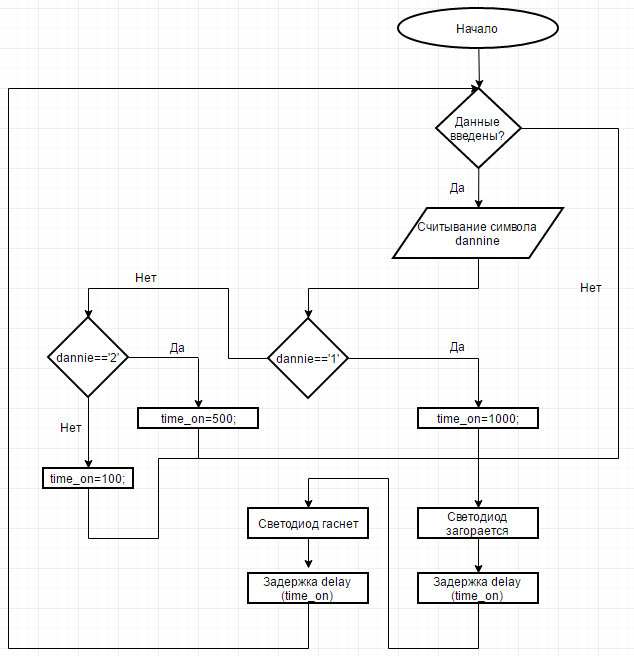


Рисунок 20. Алгоритм изменения частоты светодиода.

Следующий оператор *switch… case* позволяет быстро сделать выбор из набора определенных значений, программа с его использованием выглядит намного проще, ее легче анализировать (рисунок 21).

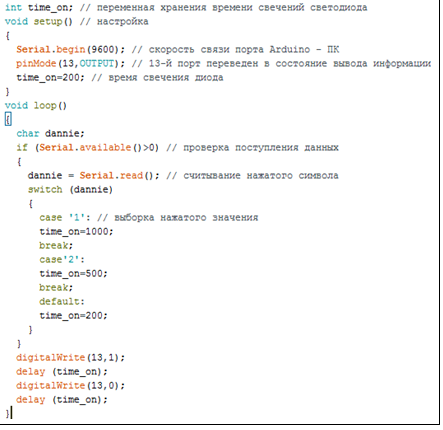


Рисунок 21. Программа управления миганием светодиода с ПК с использованием   
оператора *switch… case.*

Далее приводится описание работы операторов организации циклов.

Два оператора: *while* и *for*.

Рассмотрим цикл *while* на примере программы с рисунка 11, заменив функцию ожидания *delay ()* на оператор цикла по условию (рисунок 22). Применим при этом функцию *millis ()*, которая возвращает количество миллисекунд от старта программы. В результате работы этой программы светодиод включается на 1 сек. и выключается на 1 сек.

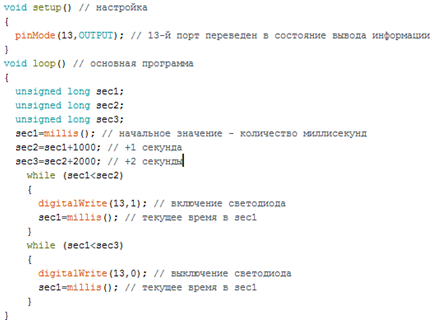


Рисунок 22. Программа управления миганием светодиода с ПК с использованием оператора цикла *while.*

Оператор цикла *for* хорошо подходит тогда, когда требуется выполнить определенное количество повторений. При построении алгоритмов в виде блок-схем для него есть отдельная фигура (рисунок 23). Для примера на рисунке 24 в коде программы светодиод моргает

500 раз с увеличением времени свечения на 1 миллисекунду каждый повтор.



Рисунок 23. Представление оператора *for.*

Рисунок 24 демонстрирует программу, управляющую миганием встроенного светодиода на 13-м порту, при этом частота мигания уменьшается по мере приближения к концу цикла.

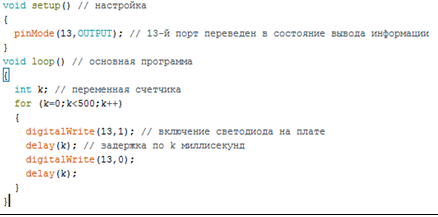


Рисунок 24. Пример использования оператора *for* на 500 повторений.

# Функции

Функции являются поименованными блоками команд, которые по их имени можно вызвать из любого места программы. Функции нужно применять там, где существует необходимость одинаковый программный код повторять много раз, или если нагромождение команд в основной программе делает ее нечитаемой [[10](#_bookmark21)].

Функция — это блок кодов программы, у которых есть имя и ряд команд, которые запускаются при вызове функции. Например, функции void setup и void loop [[18](#_bookmark29)].

Существуют встроенные функции — они уже присутствуют в библиотеке функций Arduino IDE — и функции, созданные в рамках текущей программы. Также существуют библиотеки, которые можно подключать к текущей программе и пользоваться имеющимися в них функциями. Подключаются библиотеки из пункта оболочки **Скетч | Подключить библиотеку**. После подключения той или иной библиотеки, можно использовать содержащиеся в ней готовые программные решения, написанные для различных устройств и датчиков.

На рисунке 25 приведен пример написания и использования простой функции — она сравнивает два числа и возвращает наибольшее из сравниваемых. Для функции, которые могут возвращать значения, при их описании в строке перед именем функции устанавливается тип возвращаемого значения, на данном случае — это целое число: *int*. Для возврата значения из функции используется ключевое слово *return*, за которым следует возвращаемое значение.

При описании функции, не возвращающих значения, в строке перед их именем должно стоять слово *void*.

Действия, производимые функцией, заключаются в фигурные скобки и называются *телом функции*.

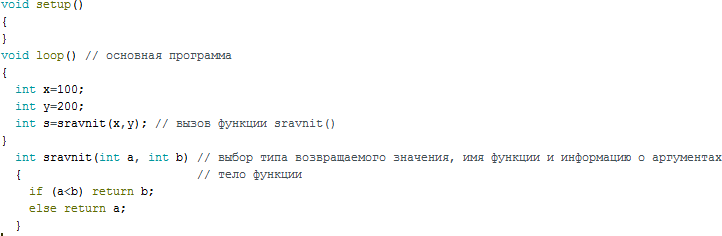


Рисунок 25. Пример простой функции.

# Элементы объектно-ориентрованного программирования

В отличие от Turbo Pascal, который является структурным языком, язык C++, составляющий основу для программирования в проекте Arduino, является объектно-ориентированным. Объекты — это сложные структуры, которые имеют оригинальное название, могут включать в себя как переменные, так и функции. Фактически, объекты созданы для удобного обращения к сложным структурам. Например, когда программно подключается сервомотор, то сначала подключается библиотека (*#include<Servo.h>*), а затем создается объект *servomotor* (рисунок 26).

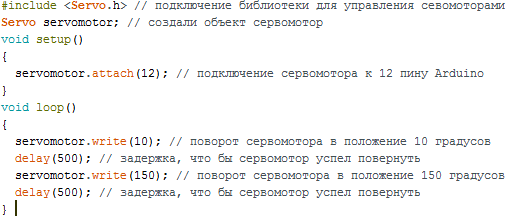


Рисунок 26. Пример управления сервомотором.

Естественно, что при этом контакт сервомотора должен быть физически подключен к порту Arduino, и на мотор подано электропитание (рисунок 27).

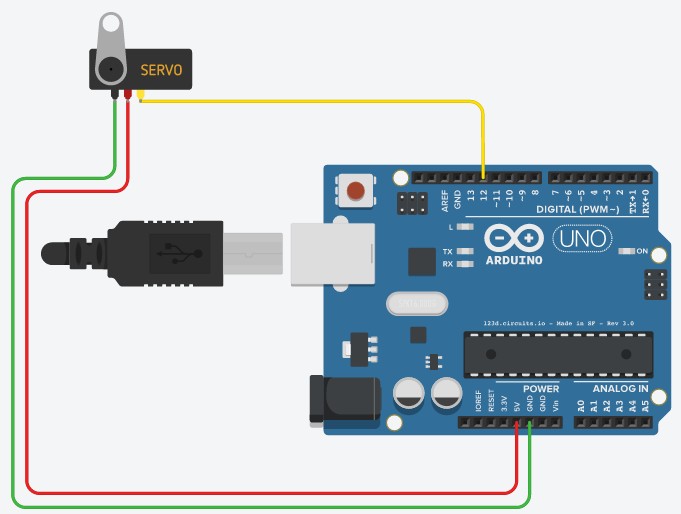


Рисунок 27. Подключение сервомотора к контроллеру Arduino.

# Задания на лабораторную работу

*Arduino* (Ардуино) — аппаратная вычислительная платформа, основными компонентами которой является плата ввода-вывода и среда разработки.

*Алгоритм —* набор последовательных действий/команд, соблюдающих логичный порядок, который позволяет решать какие-либо задачи.

Для графического представления алгоритмов используются, так называемые *блок-схемы. Блок-схемы* — это графическое представление алгоритма, состоящее из набора геометрических фигур. Каждая фигура изображает определенную операцию или действие (рисунок 28).

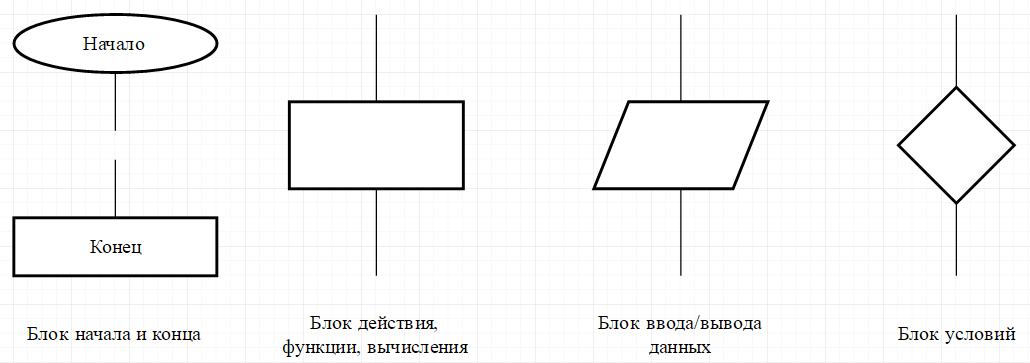


Рисунок 28. Графическое представление алгоритма.

Алгоритмы бывают нескольких типов: линейные, циклические, разветвляющиеся.

*Среда разработки Arduino IDE* — в основу языка программирования в данной среде положен C++ — один из самых популярных языков программирования.

Структура простой программы, написанной в среде Arduino IDE, в сравнении с Turbo Pascal (Таблица 2).

Таблица 2 —

Сравнение простых программ.

|  |  |
| --- | --- |
| **Arduino IDE** | **Turbo Pascal** |
| int a; //обозначение переменных  int b;  void setup () { //настройки программы  Serial.begin(9600); //скорость связи данных  }  void loop() { // основная программа  b=5;  a=b\*10;  Serial.println(a); //вывод на ПК ответ  delay(5000); // задержка 5 секунд  } | PROGRAM sum; {заголовок программы} USES CRT; { подключение библиотек} VAR a,b: integer;{объявление перемен- ных}  BEGIN {начало блоков операторов}  CLRSCR; {очистка экрана}  b:=5; {оператор 1}  a:=b\*10; {оператор 2} WRITELN(a); {Вывод а}  END. {конец блока операторов} |

Переменные отличаются по типу данных (таблица 3).

Таблица 3 —

Типы данных в сравнении с Turbo Pascal.

| **Arduino**  **IDE** | **Turbo Pascal** | **Принимаемые значения** |
| --- | --- | --- |
| *boolean* | BOOLEAN | True/False |
| *char* | CHAR | -128/+127 |
| *byte* | *BYTE* | 0-255 |
| *int* | *INTEGER* | -32768/32767 |
| *long* | *LONGINT* | -2147483648/214748367 |
| *float* | RE- AL/SINGLE/DOUBLE/EX  TENDED | -3,4028235·1038/3,4028235·1038 |

Программы с линейным алгоритмом (таблица 4). Сравниваемые части программ выделены жирным курсивом.

Таблица 4 —

Примеры линейной программы Arduino IDE в сравнении с Turbo Pascal.

|  |  |  |
| --- | --- | --- |
| **Arduino IDE** | **Блок-схема** | **Turbo Pascal** |
| int a; |  | PROGRAM sum; |
| int b; | USES CRT; |
| void setup () | VAR a,b: integer; |
| { | BEGIN |
| Serial.begin(9600); | CLRSCR; |
| } | **b:=5;** |
| void loop() | **a:=b\*10;** |
| { | WRITELN(a); |
| **b=5;** | END. |
| **a=b\*10;** |  |
| Serial.println(a); |  |
| delay(5000); |  |
| } |  |

**Задание № 1**

**Примечание:** установить среду разработки Arduino IDE, настроить оборудование перед началом работы.

1. Для начала зайдем на официальный сайт проекта Arduino в Интернете по адресу [**www.arduino.cc/en/Main/Software**](http://www.arduino.cc/en/Main/Software). Для того, чтобы скачать инсталлятор выберем в разделе **Download the Arduino IDE** вариант **Windows** Installer (рисунок 29), затем нажмите **JUST DOWNLOAD** (рисунок 30).

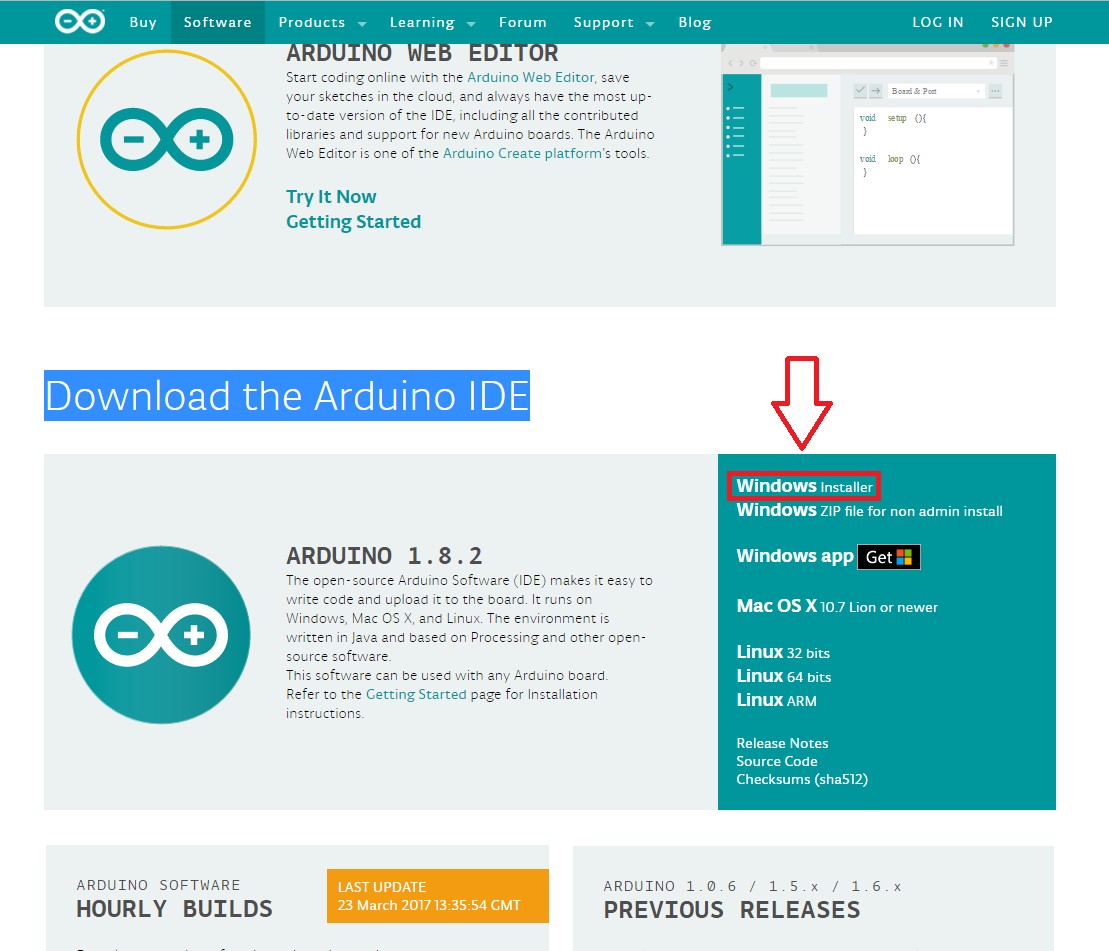


Рисунок 29. Окно скачивания программы Arduino IDE.



Рисунок 30. Кнопка загрузки установочного файла Arduino IDE.

1. После загрузки файла *arduino-1.8.2-windows.exe* запустите его с административными полномочиями, примите условия лицензионного соглашения и принимайте последующие варианты установки.

Установщик будет предлагать установить драйвера порта, на что следует отвечать утвердительно.

1. После установки запустите на рабочем столе ярлык **Arduino.exe**.

При запуске появится окно (рисунок 31), в котором содержится заготовка программы. В окне имеется заготовка, которая состоит из функций setup и loop. Функция setup содержит команды, выполняемые одни раз при включении — это так называемые настройки программы. A loop включает в себя основную программу. Она выполняется бесконечное число раз, до тех пор, пока мы не отключим питание.

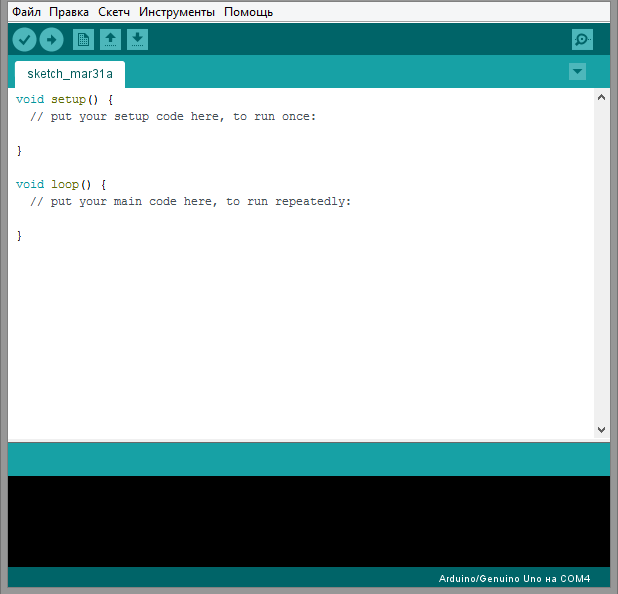


Рисунок 31. Окно программы Arduino IDE.

1. Для подключения контроллера используются USB-кабели. Подключите контроллер Arduino к компьютеру кабелем. После того как физический контакт ПК с контроллером установлен, нужно установить связи между ним и средой разработки Arduino IDE. Для этого необходимо выбрать номер порт (рисунок 32). Если портов много, рекомендуется запомнить все имеющиеся, а затем физически отсоединить Arduino от кабеля, и снова проанализировать список портов, — тот, который исчез, и есть нужный. Далее установите флажок на нужном порте.

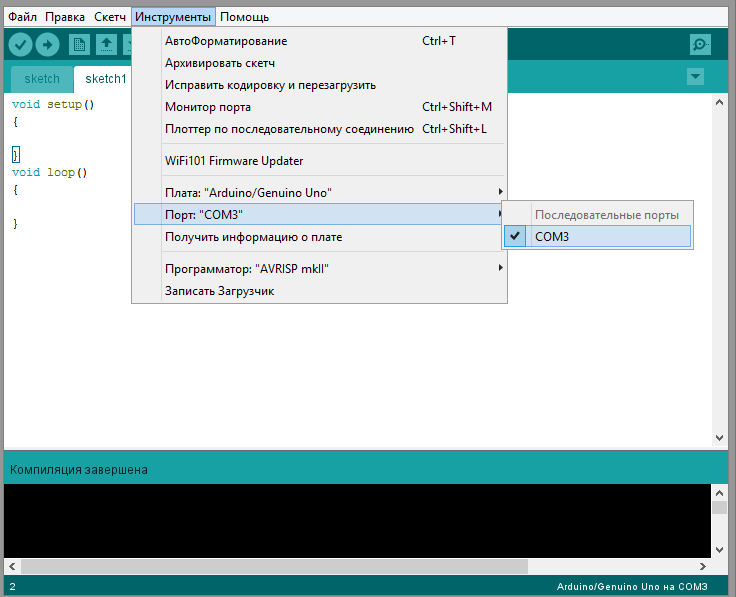


Рисунок 32. Выбор порта.

У пользователей Windows XP может возникнуть проблема — драйвер порта не установился автоматически. Решение проблемы описано на сайте по адресу: <http://arduino.ru/Guide/Windows>.

1. Далее выберите тип вашего контроллера. На рисунок 33 можно видеть, что выбрать Arduino/Genuino Uno.



Рисунок 33. Выбор типов контроллера Arduino.

Если контроллер выбран автоматически, то убедитесь в правильности выбора.

1. Настройка закончена. Чтобы загрузить программу необходимо нажать кнопку  (см. рисунок 31). Оболочка проверит программу на наличие ошибок, а затем переведет ее в двоичный код данных и команд выбранного микроконтроллера и запишет в Arduino.
2. В качестве примера, заставим Arduino мигать встроенным светодиодом на 13-м порту (рисунок 34).

**Диод** — это электронный элемент, который пропускает электрический ток, только в одном направлении.

**Светодиод** — это диод, который начинает светиться при протекании по нему тока. На плате контроллера Arduino, как правило, уже установлен один светодиод — на 13-м порту (ножке).



Рисунок 34. Программа мигания светодиодом.

Аналогичным образом перепишем программу и загрузим ее в микроконтроллер. На выходе получим мигающий на плате светодиод с задержкой в 1 секунду (рисунок 35).

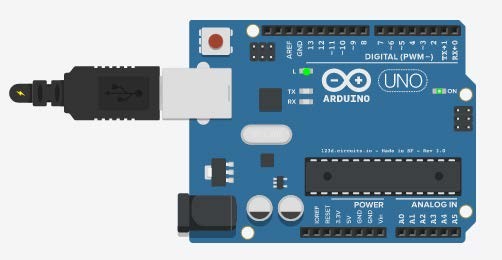


Рисунок 35. Результат работы светодиода (L — светодиод).

1. Для подключения монитора порта, необходимо открыть вкладку **Инструменты | Монитор** порта и скорректировать скорость порта (9600 бод) — в окне Монитор порта скорость задается в правом нижнем углу выбором из всплывающего списка.

Команда Serial.println( ) передает с контроллера на ПК значение, указанное в скобках.

# Самостоятельная работа

1. Измените код программы так, чтобы светодиод мигал с каждым разом быстрее в течении 10 секунд, путем добавления команд digitalWrite() и delay ().
2. Напишите программный код и запрограммируйте микроконтроллер Arduino Uno таким образом, чтобы каждые N-секунд он выводил случайно сгенерированное целое число на порт.
3. Создайте базу данных на СУБД Microsoft SQL Server с одной таблицей (код: int, показание с ардуино: float/decimal, дата и время: datetime).
4. Создайте Windows – приложение на платформе Microsoft Visual Studio с одной экранной формой, расположив на форме текстовое поле.
5. Написать программный код для приложения, который получает данные, отправляемые Arduino на COM-PORT и занести эти данные в таблицу созданной базы данных.

# Контрольные вопросы

1. Что такое Arduino?
2. Что такое алгоритм? Какие алгоритмы Вы знаете? Дайте определение понятию блок-схема?
3. Типы данных, используемые при программировании Arduino.
4. Операторы, используемые при программировании Arduino.

# СПИСОК ЛИТЕРАТУРЫ

1. Аппаратная платформа Arduino[Электронный ресурс]. — Режим доступа: <http://arduino.ru/Hardware>(дата обращения 24.04.17).
2. Белослудцева Л. И., Прончев Г. Б. Курс робототехники для дополнительного образования [Текст] // Проблемы и перспективы развития образования: материалы II Междунар. науч. конф. (г. Пермь, май 2012 г.). — Пермь: Меркурий, 2012. — С. 102-104.
3. Гайсина И. Р. Развитие робототехники в школе [Текст] // Педагогическое мастерство: материалы II Междунар. науч. конф. (г. Москва, декабрь 2012 г.). — М.: Буки-Веди, 2012. — С. 105-107.
4. Готлиб Б. М. Введение в специальность «Мехатроника и робототехника» [Текст]: курс лекций / Б. М. Готлиб, А. А. Вакалюк. — Екатеринбург: УрГУПС, 2012. — 134 с.

* Золотарева А. В. Дополнительное образование детей: история и современность [Текст]: учеб. пособие для СПО / А. В. Золотарева, А. Л. Пикина, Н. А. Мухамедьярова, Н. Г. Тихомирова; отв. ред. А. В. Золотарева. —2-е изд., испр. и доп. — М.: Издательство Юрайт, 2016. 353 с. — (Профессиональное образование).
* Золотарева А. В. Методика преподавания по программам дополнительного образования детей [Текст]: учебник и практикум для СПО / А. В. Золотарева, Г. М. Криницкая, А. Л. Пикина. — 2-е изд., испр. и доп. М.: Издательство Юрайт, 2016. — 399 с. — (Профессиональное образование).
* Кругликов Г. И. Методика профессионального обучения с практикумом [Текст]: учеб. пос. для студ. высш. учеб. зав-ий / Г. И. Кругликов. 2-е изд., стер. — М.: Издательский центр «Академия», 2007. — 288 с.

1. Курс «Arduino для начинающих» [Электронный ресурс]. — Режим доступа: <http://edurobots.ru/kurs-arduino-dlya-nachinayushhix> (дата об- ращения 10.05.17).
2. Максимов П. В. Анализ одноплатных компьютеров, потенциально пригодных для использования в обучении [Текст] / П. В. Максимов, Ю. В. Корнилов // Педагогическое мастерство и педагогические технологии: материалы VI Междунар. науч.-практ. конф. (Чебоксары, 27 нояб. 2015 г.). В 2 т. Т. 2 / Редкол.: О. Н. Широков [и др.]. — Чебоксары: ЦНС «Интерактив плюс», 2015. — №4 (6). — С. 244-246.
3. Момот М. В. Мобильные роботы на базе Arduino [Текст] / М. В. Момот. — СПб.: БХВ-Петербург, 2017. — 288 с.: ил.
4. Морозова Н. А. Дополнительное образование — многоуровневая система в непрерывном образовании России [Текст] / Н. А. Морозова. — М.: МГУП, 2001. — 277 с.
5. Осадчий В. В., Осадчая Е. П. Анализ проблемы профессиональной подготовки программиста и пути ее решения [Текст]: журнал «Образовательные технологии и общество» / В. В. Осадчий, Е. Е. Осадчая. — Республика Татарстан, Казань: ФГБОУ ВПО «КНИТУ», 2014, том 17, вы- пуск № 3. — С. 362-377.
6. Осин А. В. Мультимедиа в образовании: контекст информатизации [Текст] / А. В. Осин. — М.: Агентство «Издательский сервис», 2004. – 320 с.
7. Петин В. А. Проекты с использованием контроллера Arduino [Текст] / В. А. Петин. — 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2016. — 464 с.: ил.
8. Родионов В. И. Подготовка электронных публикаций в InDesign

CS6 [Текст] / В. И. Родионов. — СПб.: БХВ-Петербург, 2013. — 224 с.: ил.

1. Современное образование: робототехника в школе [Электронный ресурс]: Technoguide. Technologies of the future — Режим доступа: [http://techno-guide.ru/robototekhnika/sovremennoe-obrazovanie-](http://techno-guide.ru/robototekhnika/sovremennoe-obrazovanie-robototekhnika-v-shkole.html) [robototekhnika-v-shkole.html](http://techno-guide.ru/robototekhnika/sovremennoe-obrazovanie-robototekhnika-v-shkole.html) (дата обращения: 05.05.2017).
2. Соммер У. Программирование микроконтроллерных плат Arduino / Freeduino [Текст] / У. Соммер. — СПб.: БХВ — Петербург, 2012. — 256 с. ил.
3. Старикова Л. Д. Методика профессионального обучения [Текст]: практикум / Л. Д. Старикова, Ю. С. Касьянова. — Екатеринбург: изд-во Рос. гос. проф.-пед. ун-та, 2013. — 131 с.
4. Филиппов С. А. Робототехника для детей и родителей [Текст]: учеб. пособие / С. А. Филиппов. — СПб.: Наука, 2013. – 319 с.
5. Халамов В. Н. Образовательная робототехника на уроках информатики и физики в средней школе [Текст]: уч.-метод. пособие / В. Н. Халамов. — Челябинск: Взгляд, 2011. — 160 с.
6. Чернилевский Д. В. Дидактические технологии в высшей школе [Текст]: учебное пособие для вузов / Д. В. Чернилевский. Москва: ЮНИТИ-ДАНА, 2002. — 437 с.
7. Эрганова Н. Е. Методика профессионального обучения [Текст]: учебное пособие / Н. Е. Эрганова. — 3-е изд., испр, и доп. — М: Академия, 2007. — 160 с.
8. Юревич Е. И. Основы робототехники [Текст] / Е. И. Юревич. — 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2005. — 416 с.

Составитель

Александр Сергеевич Абрамович

**Методические указания  
ДЛЯ ВЫПОЛНЕНИЯ  
ПРАКТИЧЕСКОЙ РАБОТЫ**

по теме «Микроконтроллеры. Arduino» по дисциплине   
«Информационные технологии»  
для студентов очной формы обучения направления подготовки  
09.03.02 Информационные системы и технологии, профиль — 01   
«Системная интеграция и автоматизация информационных   
процессов»

Сверстано и отпечатано в филиале КузГТУ в г. Прокопьевске

653039, г. Прокопьевск, ул. Ноградская, 19а

Формат 60x84 1/16. Объем 3,1 п. л. Заказ 287