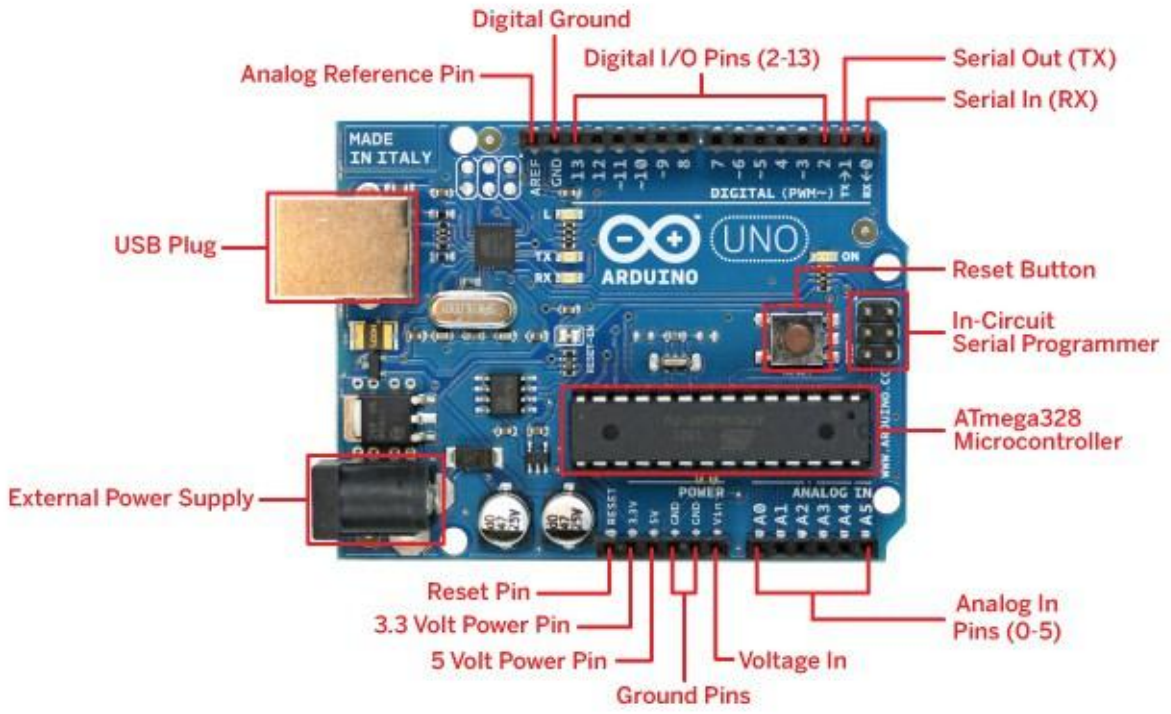
Конспект первого занятия проектной мастерской.

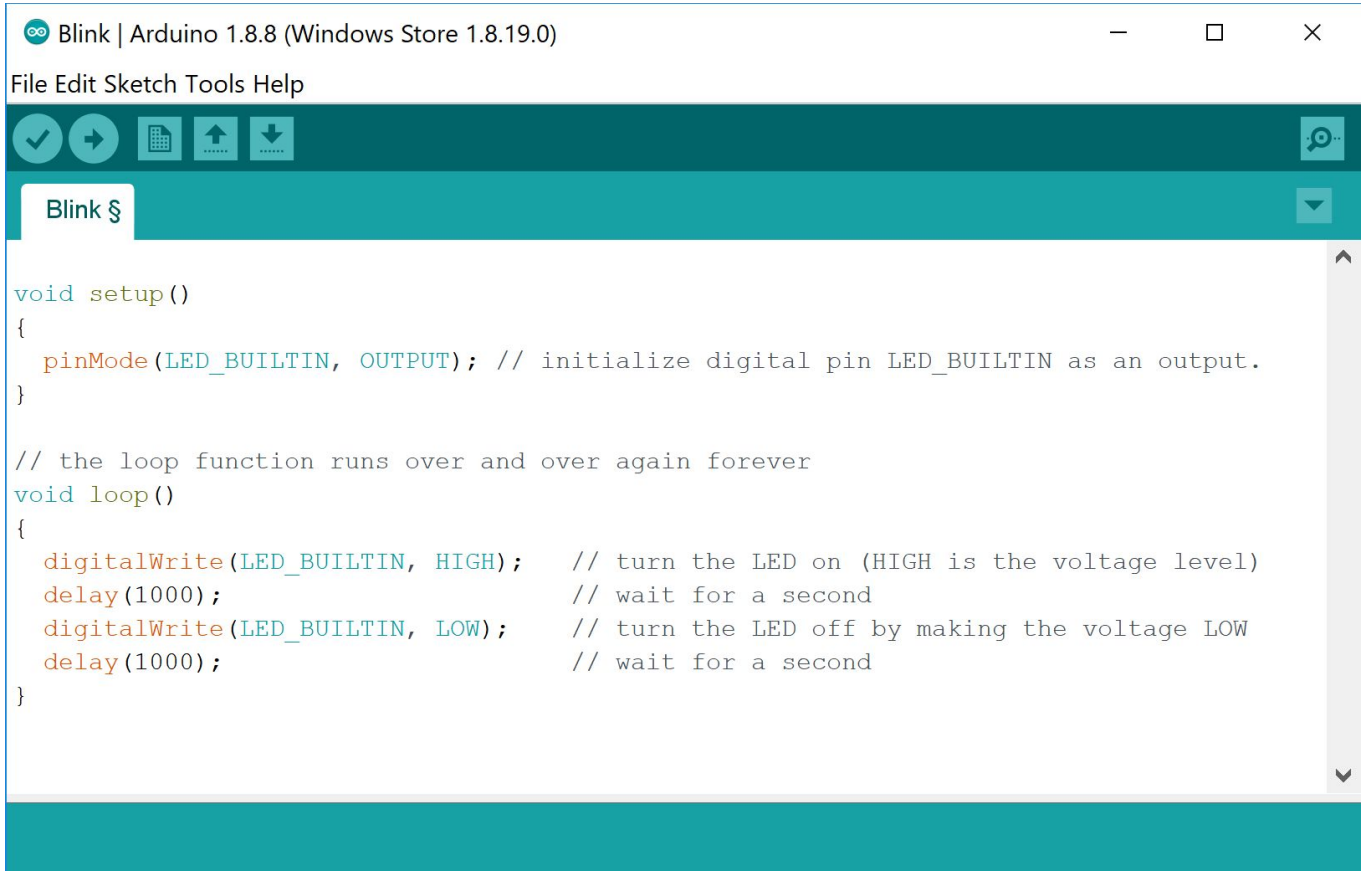
28.02.2019



Каждая плата имеет на борту USB порт, дополнительный разъем питания, и пару десятков пинов (ног). Пины подразделяются на цифровые и аналоговые. Цифровые могут работать в двух режимах – на вход и на выход и способны выдавать(принимать) два уровня сигнала: 1 (5 вольт) и 0 (0 вольт). Аналоговые пины работают только на вход, зато у них больше уровней входного сигнала: 0 соответствует 0 вольт, 1023 – 5 вольт. Все промежуточные значения можно вычислить, решив простую пропорцию.

Arduino — это удобная платформа быстрой разработкии прототипирования электронных устройств. Платформа пользуется огромной популярностью во всем мире благодаря удобству и простоте языка программирования, а также открытой архитектуре и программному коду. Устройство программируется через USB без использования программаторов.

Первым примером, который мы разобрали, был мигающий светодиод.



Внутри функции setup() располагается код, который будет выполнен единожды после перезагрузки микроконтроллера.

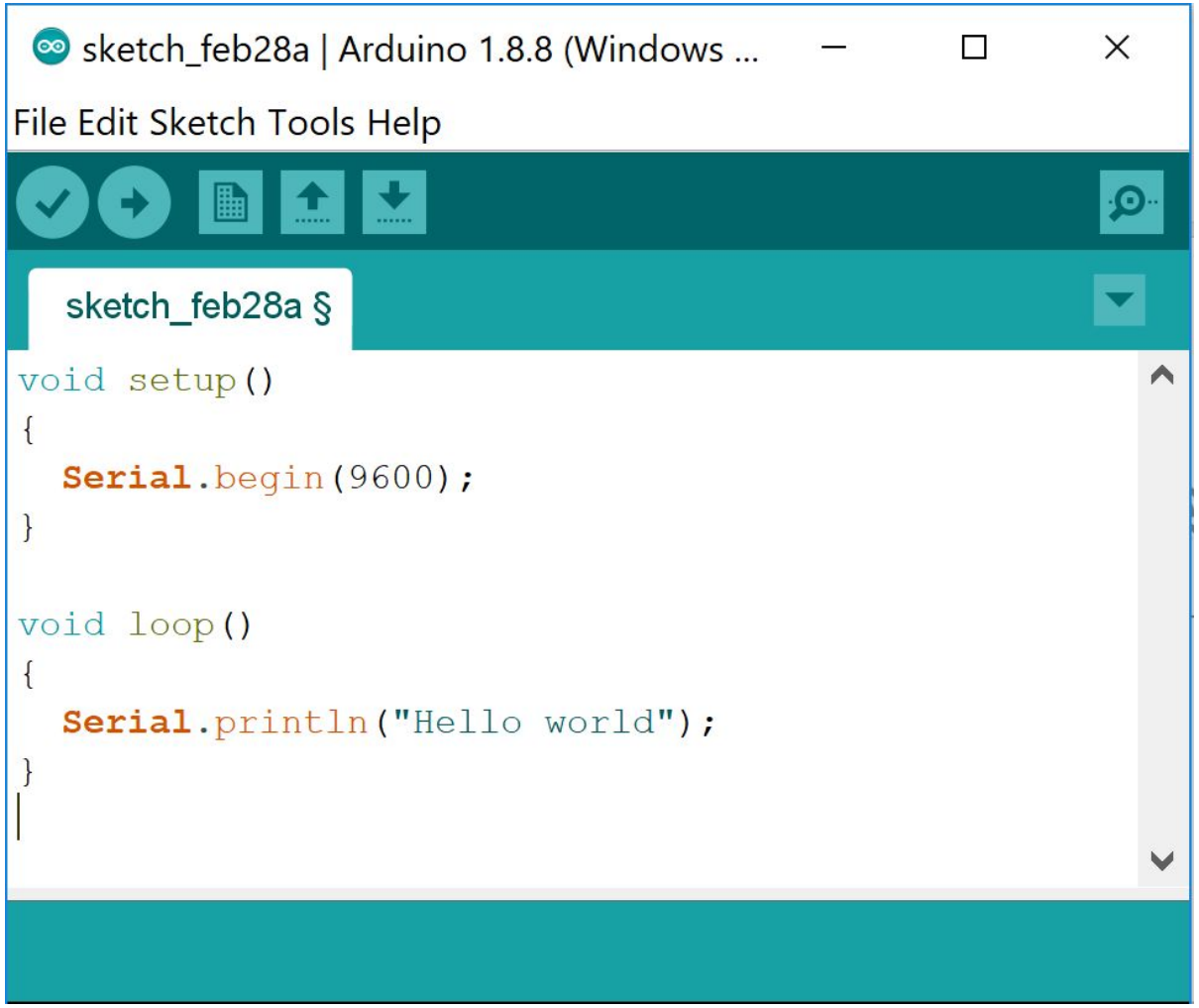
Код, расположенный между фигурными скобками функции loop() будет выполняться бесконечно. То есть, когда будет выполнена последняя инструкция, указатель переместится в начало функции, и она снова начнёт выполняться.

Функция pinMode() задаёт режим работы пина. Первый аргумент внутри круглых скобок означает номер пина, второй – его режим работы. В данном случае их роль играют константы LED_BUILTIN, равная 13 и INPUT/OUTPUT – вход и выход соответственно.

digitalWrite(n, level) подаёт напряжение уровня level на пин с номером n.
Уровень может принимать 2 значения: LOW (0 вольт) и HIGH (5 вольт).
delay(x) – притормаживает работу программы на x миллисекунд.

LED_BUILTIN неслучайно равна 13. Именно к этому пину подпаян встроенный светодиод, который замигает при выполнении этой программы.

Serial Monitor

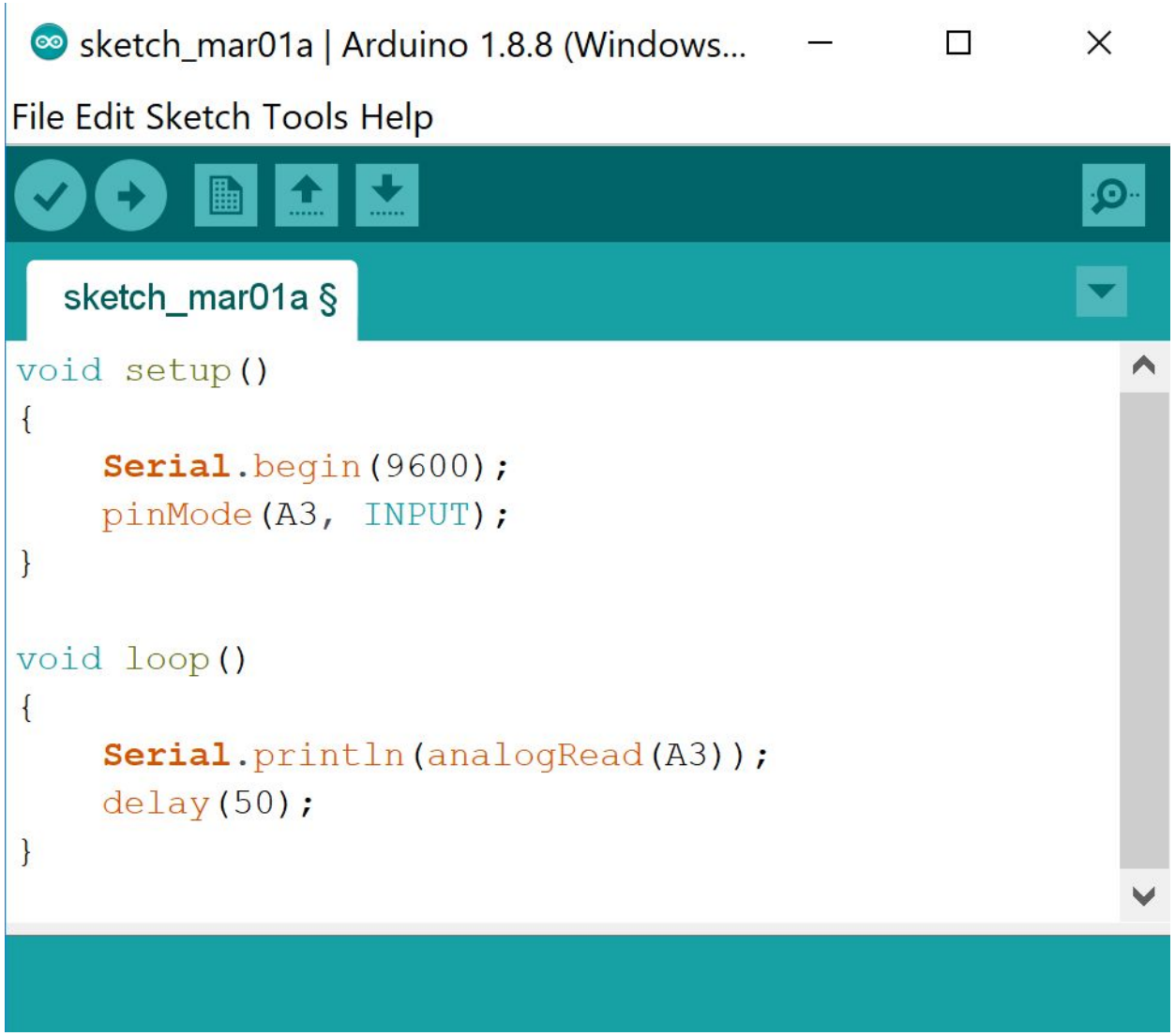


Для отладки бывает полезно выводить какую-то информацию на экран. В случае Arduino это проще всего сделать при помощи монитора порта(Serial monitor).

Serial.begin(9600); запускает монитор порта на скорости 9600 бод (https://ru.wikipedia.org/wiki/Бод). Это значение может быть изменено как большую, так и в меньшую сторону, но мы оставим всё как есть.

Serial.println(message) выводит сообщение message в монитор порта
Serial.print(message) делает ту же функцию, только после вывода не происходит перевода строки.

Ещё одна рассмотренная функция – analogRead()

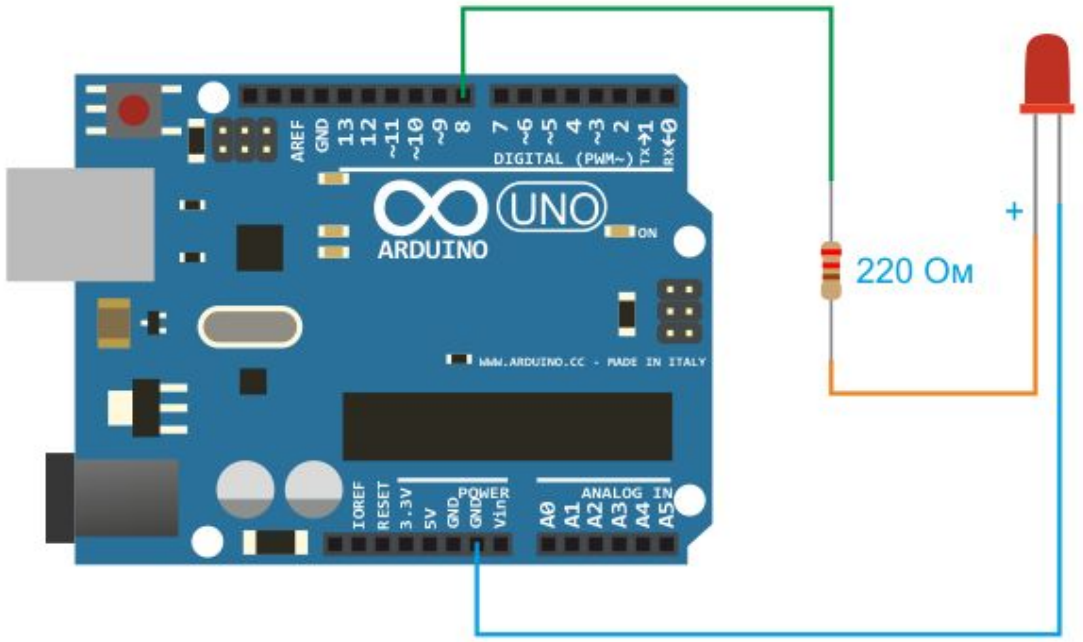


Используется она для того, чтобы считать значение напряжение с аналогового входа. 0 соответствует 0 вольт, 1023 – 5V.

Имеет 1 обязательный аргрумент – номер порта. Перед использованием порт должен быть инициализирован посредством вызова pinMode(port, INPUT)

В примере слева, считанное с A3 значение передаётся на монитор порта.

Как это всё использовать?



Не подключайте светодиоды без дополнительного сопротивления!!!

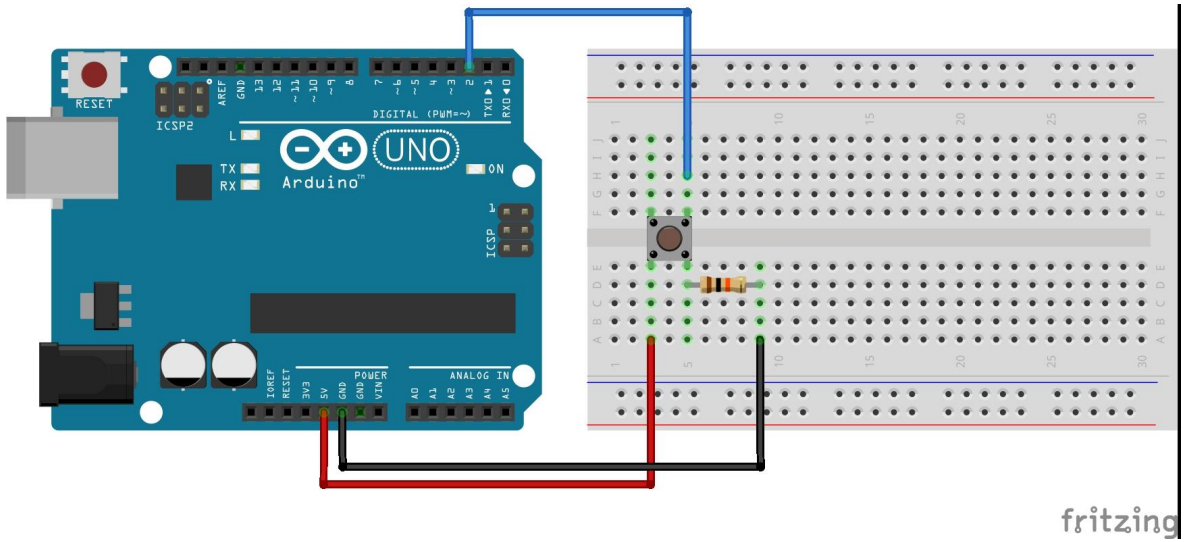
При помощи функции digitalWrite() можно управлять различными устройствами. Давайте помигаем внешним светодиодом.

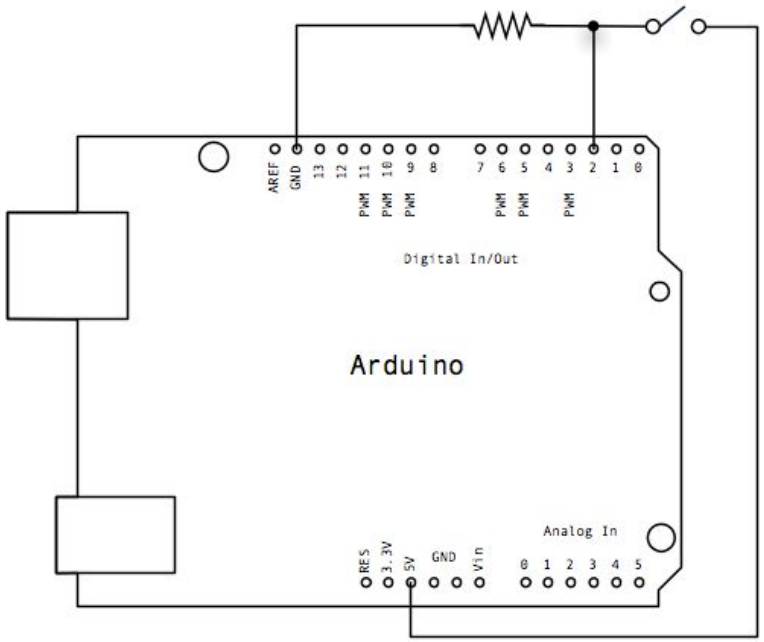
Для этого нужно проинициализировать нужный порт на выход (на картинке он имеет номер 8 и вызывавать digitalWrite(), передавая этот номер этого порта в качестве первого аргумента.

Например код

```
digitalWrite(8, HIGH);
delay(1000);
digitalWrite(8, LOW);
```

зажжет светодиод на одну секунду, а затем погасит.





При помощи функции analogRead() можно получить какие-нибудь данные из внешнего мира путём измерения входного напряжения. Например определить, нажата ли кнопка.

Если вызвать функцию в момент, когда кнопка замкнута, analogRead(A2) вернёт значение 1023, соответсвующее 5V, если кнопка в данный момент разомкнута то 0.

Обратите внимание на подтягивающий резистор. Он нужен, чтобы при разомкнутой кнопке гарантировать низкий уровень сигнала на входе. Более подробно об этом написано здесь:
http://www.texnic.ru/data/other/013.html

Теперь комбинируя эти функции можно написать множество программ

- Включение светодиода по нажатию кнопки
- Включение и выключение светодиодов, двигателей и иных устройств
- Таймер

И многое другое…

На следующем занятии мы более подробно займёмся изучением синтаксиса языка C/C++
Чтобы не тратить время на совсем элементарные вещи, имеет смысл посмотреть пару курсов по языку C/C++
Ну а если вы всё схватываете на лету, можно не тратить время на их просмотр, а изучать язык самостоятельно: https://code-live.ru/tag/cpp-manual/

Для следующего занятия будет достаточно изучить операторы и функции. И обязательно практикуйтесь! Одной теорией в программировании не обойтись.