# Pumpkin Project

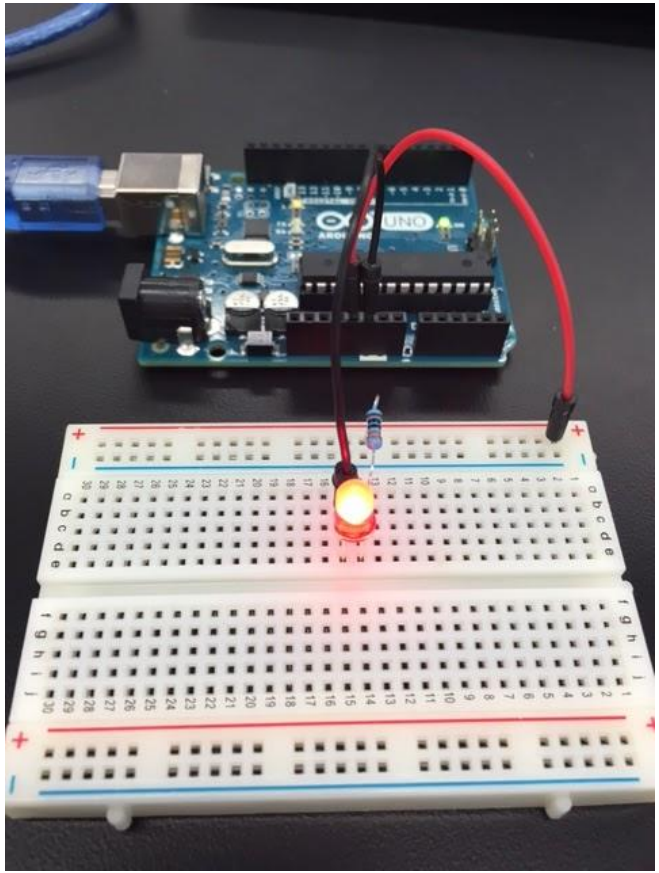
## Day #2: Writing and Arduino Sketch!

# Where we left off

Now connect your Arduino to your computer using the USB cable

The LED should light up!

Note: we haven’t actually written a program yet, we’re just using the computer as a power source



# STEP 5: Make the LED Blink!

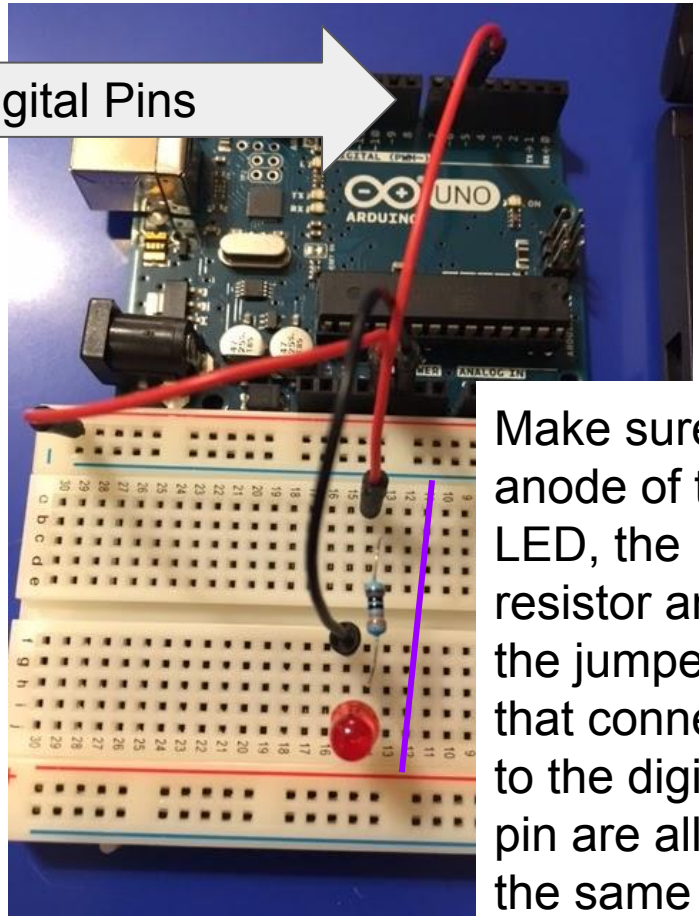
In this step we are going to write our first Arduino program! (We will call programs sketches in Arduino). We will use many of the same programming tools we used in Snap!

# Rearranging the Breadboard

Before we can program the LED we need to change the breadboard slightly.

Instead of connecting the LED directly to the 5V power, we will connect it to a digital pin on the Arduino so we can control when we light the LED

It doesn't matter which pin you pick, here, I'm using pin #5




Make sure the anode of the LED, the resistor and the jumper that connects to the digital pin are all in the same row

# On to Programming!

Now that we have rewired our circuit it’s time to move on to programming.

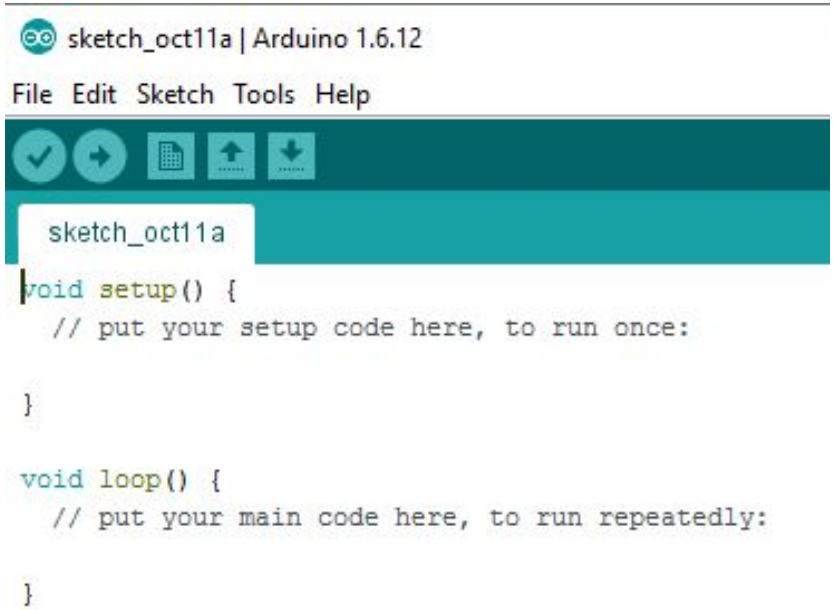
Start up the Arduino IDE on your computer

# A Basic Arduino Sketch

There are two main parts of an Arduino sketch

The **setup** function is called when a sketch starts. It is only run once right when a sketch is reset

After setup is finished, the **loop** function loops continuously



# How blinking the LED works

In our first circuit, the anode of the LED was connected to the 5V pin on the Arduino. When the Arduino was powered, we were constantly sending current through the LED

We move the anode to pin #5 which is a pin we can control through our code. We can send a HIGH signal to turn the LED on, and a LOW signal to turn the LED off. If we do this with a short delay in between each signal, we can make the LED blink

# Coding which pin we've used

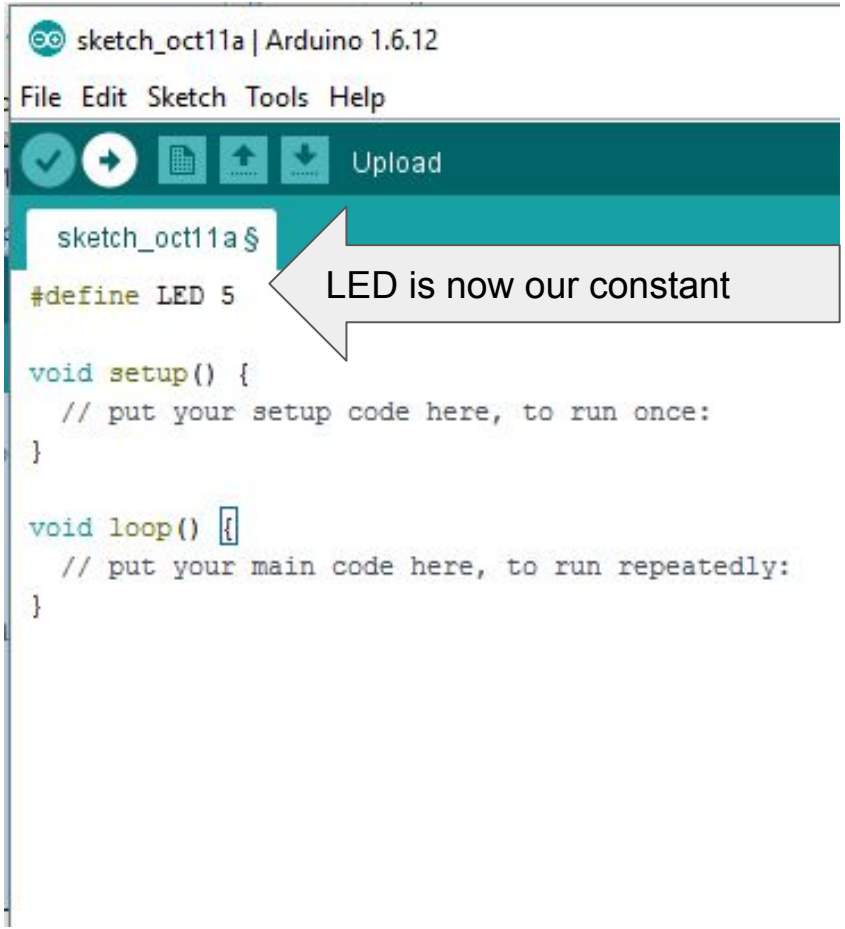
We need to create a **constant** in our program that specifies which pin we are using for our LED

A constant makes it easy to change the number in our code if we decide to use another port later

At the very beginning of your file, type

**#define LED 5**

Use the pin number you are using if not 5



# What is a constant?

In Snap! we used lots of variables

Constants are like variables in that they are named storage locations

The biggest difference is that once you give a constant a value, you cannot change it

```
#define LED 5
```

This is read “pound define”

LED is the name of the constant

The value of the constant is 5

# Set the pin to Output

Lighting up the LED is a type of output

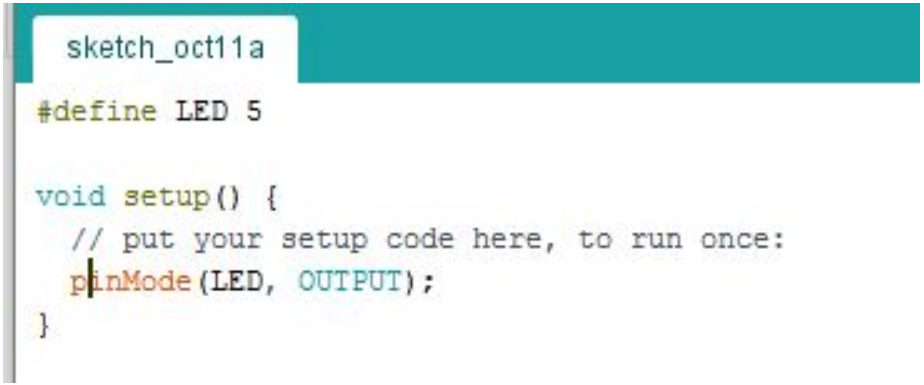
We have to specify that we want to use our LED pin for output

We only need to do this once, in our setup function

Add the line

**pinMode(LED, OUTPUT);**

To your setup function




Most of the lines of code that we write inside our setup() and loop() functions will end with a semicolon

# Function calls in Arduino

A **function** in Arduino is like a block in Snap - it is a reusable set of expressions that we can insert into our program to do a task.

Just like a block can take **inputs**, a function can take **parameters**

This line of code **calls** the pinMode function and passes it two values:

* Our LED constant so the function knows which pin we are setting
* OUTPUT to indicate we want to use that pin for output, not input

```
pinMode(LED, OUTPUT);
```

# Make the LED blink!

Now we want to make the LED blink

We will do this by setting the LED pin to HIGH, waiting for a set amount of time, then setting the pin to LOW, and waiting again

We will repeat this over and over again -  we will need two new functions

**digitalWrite()**  we will use digitalWrite(LED, HIGH) to light the LED and digitalWrite(LED, low) to turn it off

**delay()** we will use delay(100) to pause between turning the LED off and on. The time that we put into the delay function is in milliseconds

# Updated Code

sketch_oct11a

```
#define LED 5

void setup() {
  // put your setup code here, to run once:
  pinMode(LED, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(LED, HIGH);
  delay(200);
  digitalWrite(LED, LOW);
  delay(200);
}
```

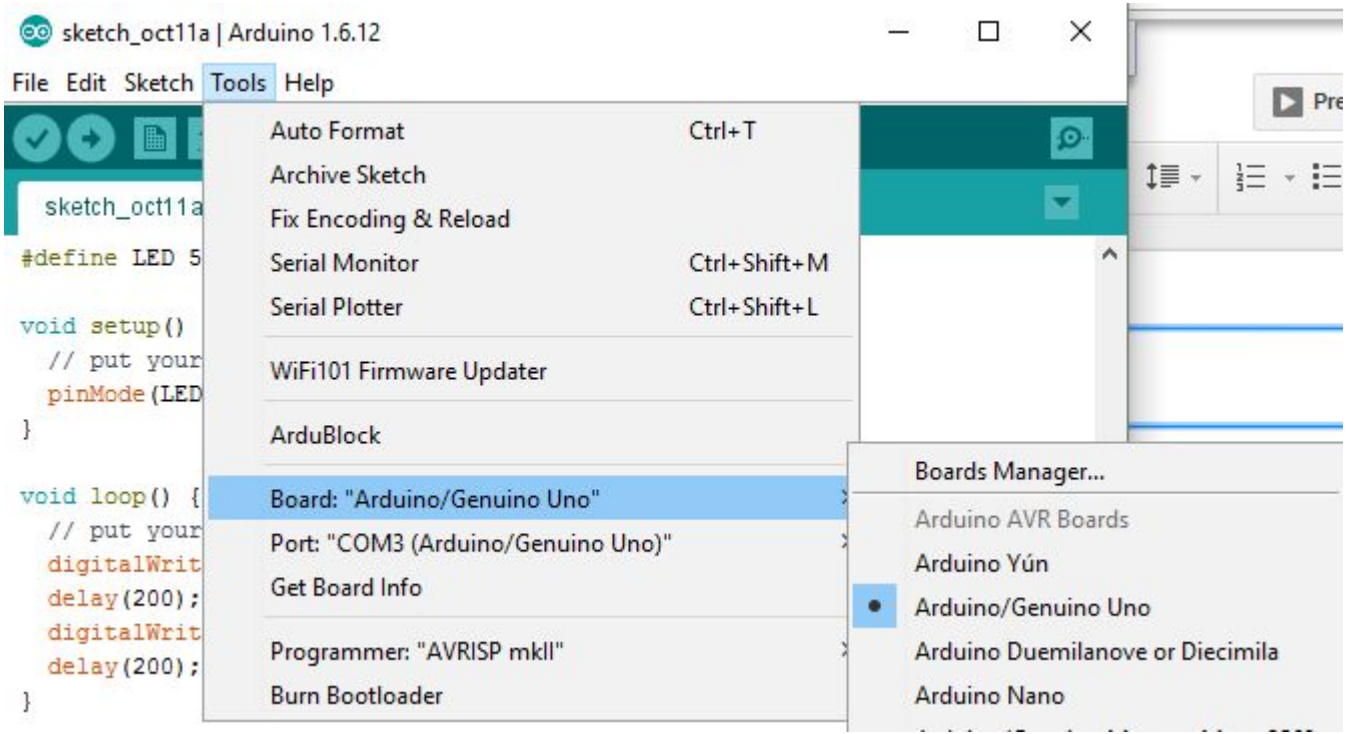
# My LED isn’t blinking!

We’ve wired the circuit and written the code, but the next step is to put the code onto the Arduino

Plug your Arduino into the USB port on your computer using your USB cable

# Loading the Code onto the Arduino

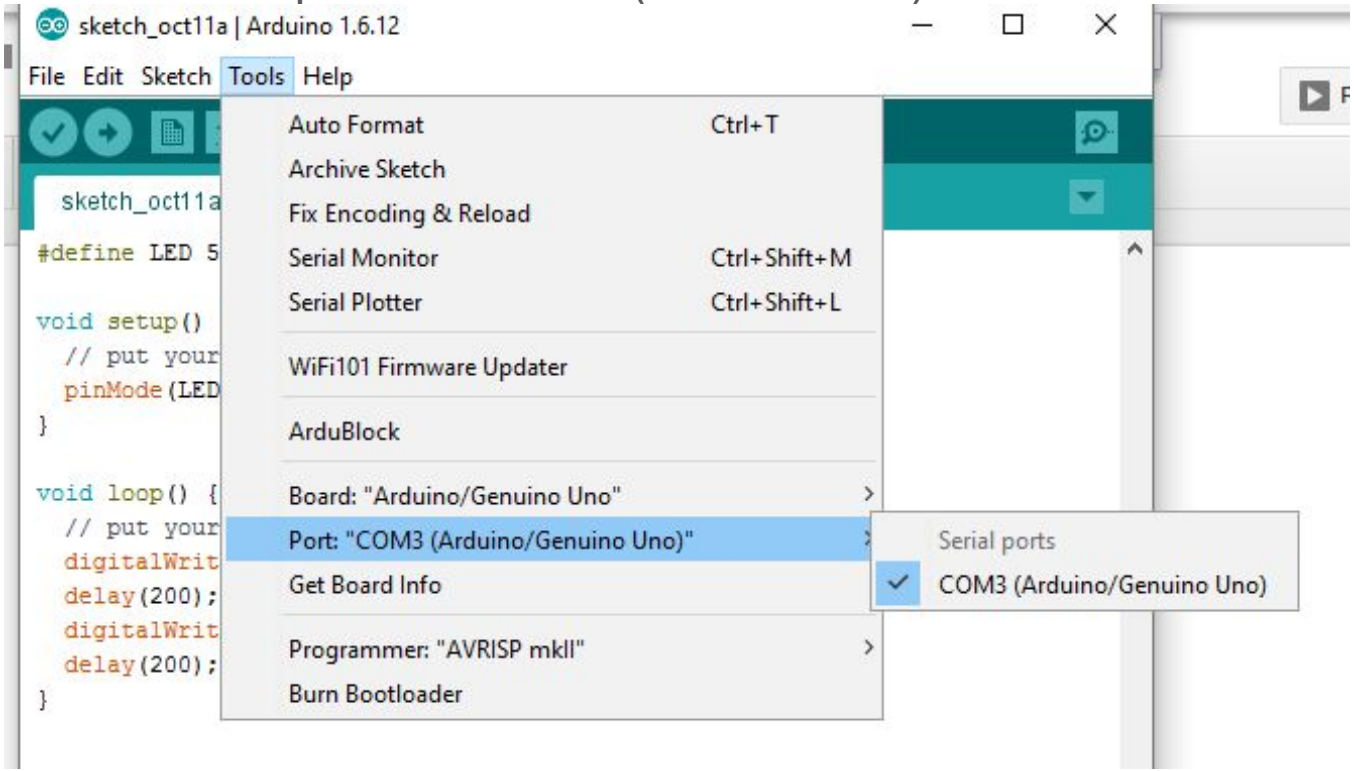
Check that the correct board is selected (Tools - > Board)




Make sure Arduino is connected via USB to your computer

# Loading the Code onto the Arduino

Check that the correct port is selected (Tools-> Port)



# Build and load your code

Click the arrow, the code will build and load

Any errors will show up in the console at the bottom of the window

When you see the **Done uploading** message your code is on your Arduino and your LED should start blinking!!

# Try it out

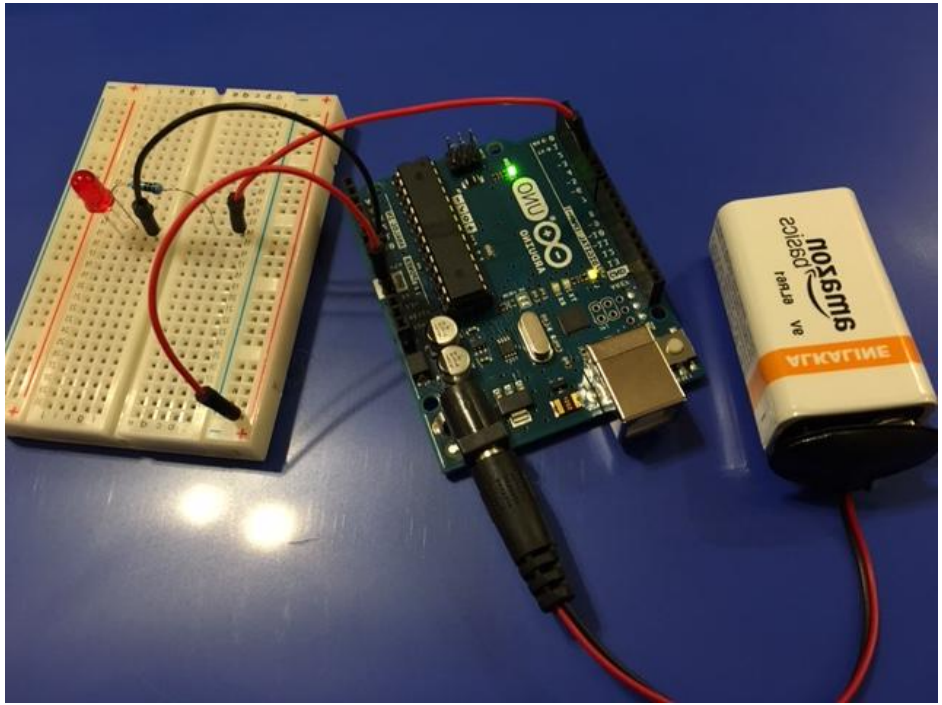
Can you make the LED blink faster? slower?

Change your code to change the speed of the blink.

Reload the code and watch the results.

# Running off the battery

To run “unplugged”, disconnect the arduino from the laptop and connect the 9V battery clip



# STEP 6: Add A Second Blinking LED

Work with your partner to add a second blinking LED to your project. Can you make it blink opposite your first LED? (So when the first LED is on, the new one is off. When the first LED is off, the new one is on). Think about what you need to change in the wiring and in the code.

If you finish early try to make the LEDs blink in different patterns. Try adding additional LEDs too!