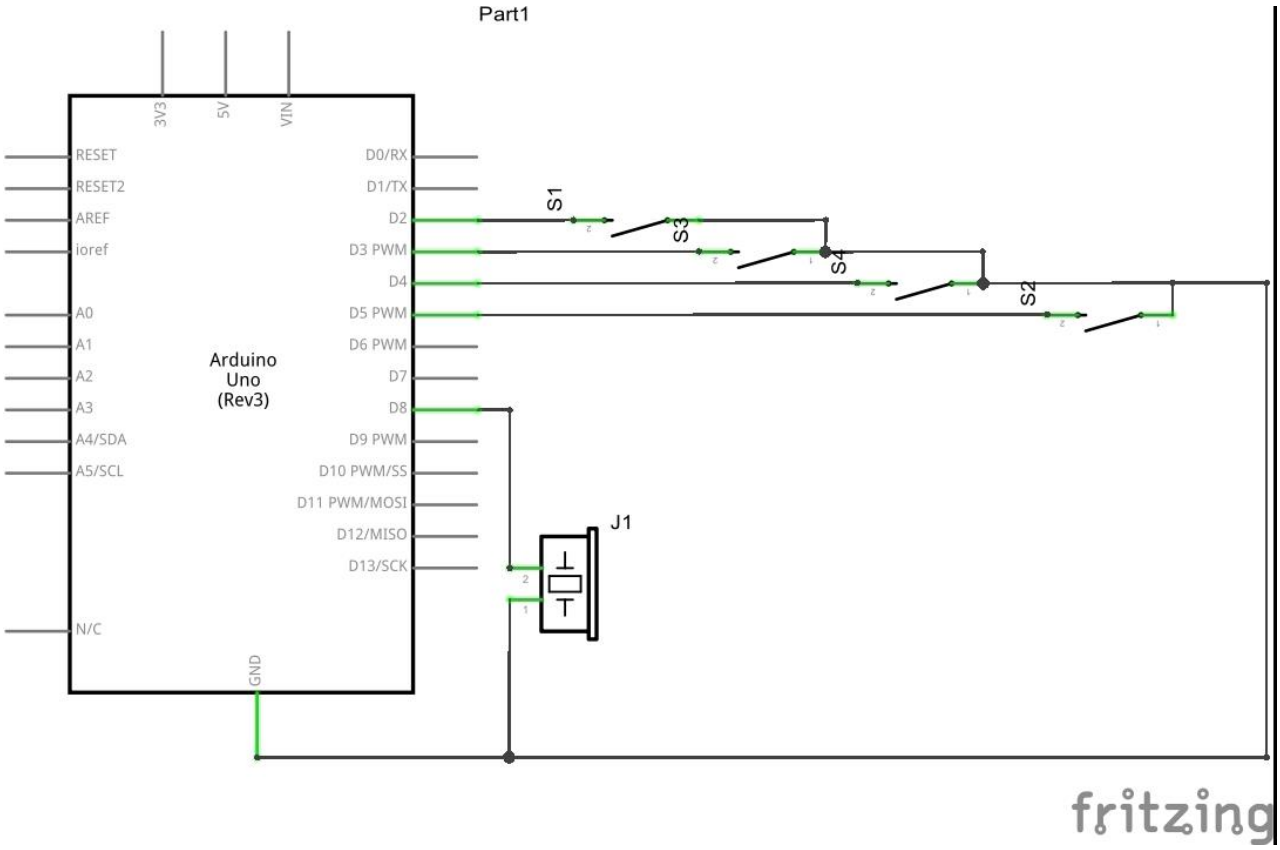
Arduino Keyboard

Before You Start

You should be comfortable uploading code to the Arduino, and you should have basic knowledge of how to build basic circuits on a breadboard (knowing how the breadboard is connected). You should know the basics of output on the Arduino.

The Circuit

Let's take a look at the circuit, and break it down part by part.



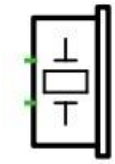
THE SWITCHES

There are many different types of switches that are used for a large variety of tasks. The switches that we are using are referred to as tactile push button switches. This means that they only have two states: on and off. If the button is pressed, it is “ON”. If it is released, it is “OFF”.

In the schematic, the one side of the switch is connected to a pin. The switch is open, and so normally the pin is not connected to GND. In our code, we tell each pin to normally be HIGH (connected to 5V). When the switch is closed, the pin has a direct connection to GND. We can detect this in our code.

THE PIEZO ELEMENT

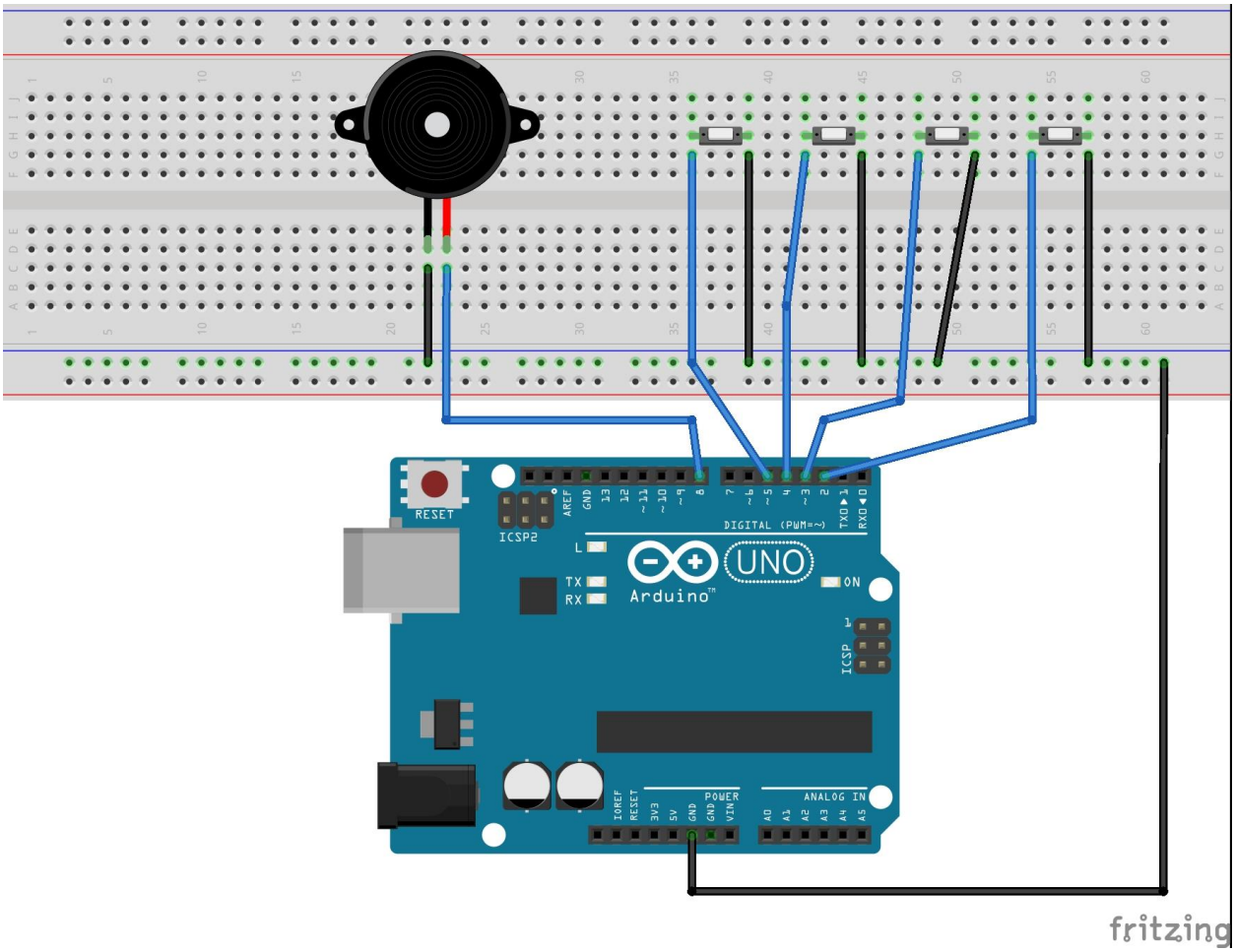
Certain materials vibrate when a voltage is applied to them. In the right configuration, they can vibrate at a frequency that humans can here. By increasing or decreasing the voltage applied to the piezo element, we can change the frequency, and consequently the pitch that we hear.



This is the symbol for the piezo element. It is like an LED in that it has a polarity. On the physical piezo element, there is a plus symbol on the same side as one of the pins. That plus symbol denotes which lead needs to be connected to the output pin of the Arduino. The other pin gets connected to ground.

The Breadboard

Below is an image of how the components will be connected in this project. Note that the piezo element may not look like the one in the picture. It needs to be connected the right way in order to produce a noise. Also note how the switches are connected. One side of the switch is connected to a pin, and the other side is connected to GND.



The Code

The code for this project is below. The comments walk you through step by step on what each part means. To learn about some of the main concepts in this tutorial's code, look of the following on the Arduino website.

- analogWrite()
- #define
- INPUT_PULLUP
- tone()

```
/*
 * In this program we will create a keyboard using input
 * and output
 */

// Below we use the octothorpe, as well as the word "define".
// When we do this, we are saying that we want key_1 to be
// equal to 5. This will be the same throughout our entire
// program.
```

```
#define piezo 8
#define key1 2
#define key2 3
#define key3 4
#define key4 5

// Remember that below is the code that runs only once,
// to set everything up.
void setup() {
  // First, set up the piezo buzzer as an output
  pinMode(piezo, OUTPUT);

  // Next, we set up all the keys as inputs. INPUT_PULLUP
  // creates a pullup that is normally HIGH. Therefore,
  // we will be checking to see if the pins go LOW
  pinMode(key1, INPUT_PULLUP);
  pinMode(key2, INPUT_PULLUP);
  pinMode(key3, INPUT_PULLUP);
  pinMode(key4, INPUT_PULLUP);
}

void loop() {
  // Below we use an if statement. This statement is very straight
  // forward! We simply say "If key4 is LOW, do something. If it
  // isn't, check if key3 is LOW, and do something different.
  // Then check key2, and key1." Finally, the "else" part
  // says "If no keys are pressed, then here is what to do.
  if(digitalRead(key4) == LOW){
    tone(piezo, 523);
  }
  else if(digitalRead(key3) == LOW){
    tone(piezo, 659);
  }
  else if(digitalRead(key2) == LOW){
    tone(piezo, 784);
  }
  else if(digitalRead(key1) == LOW){
    tone(piezo, 1047);
  }
  else{
    noTone(piezo);
  }
  // The tone function is used to make a specific frequency from
  // our keyboard. When creating projects for the Arduino, it
  // is essential to know how to use the documentation.
  // Try and figure out what tone() and noTone() do, and what
  // their parameters are by visiting www.arduino.cc and
  // searching.
}
```

Connect your Arduino to your computer, and open the Arduino IDE. Ensure that your Arduino is connected. Next, enter the code above (you don't need to include the comments) into your Arduino IDE.

Click verify (the check mark) to ensure that your code is error free. If everything looks good, then upload the code to the Arduino.

The Result

If everything has gone correctly, you should be able to press a key and hear a note. The notes should be harmonious.

Do you see why we used the chosen values for tone()?