```
<pre>/*
 * Arduino Guitar Tuner
 * by Nicole Grimwood
 *
 * For more information please visit:
 * https://www.instructables.com/id/Arduino-Guitar-Tuner/
 *
 * Based upon:
 * Arduino Frequency Detection
 * created October 7, 2012
 * by Amanda Ghassaei
 *
 * This code is in the public domain.
*/


//data storage variables
byte newData = 0;
byte prevData = 0;
unsigned int time = 0;//keeps time and sends vales to store in timer[] occasionally
int timer[10];//storage for timing of events
int slope[10];//storage for slope of events
unsigned int totalTimer;//used to calculate period
unsigned int period;//storage for period of wave
byte index = 0;//current storage index
float frequency;//storage for frequency calculations
int maxSlope = 0;//used to calculate max slope as trigger point
int newSlope;//storage for incoming slope data

//variables for deciding whether you have a match
byte noMatch = 0;//counts how many non-matches you've received to reset variables if it's been
too long
byte slopeTol = 3;//slope tolerance- adjust this if you need
int timerTol = 10;//timer tolerance- adjust this if you need

//variables for amp detection
unsigned int ampTimer = 0;
byte maxAmp = 0;
byte checkMaxAmp;
byte ampThreshold = 30;//raise if you have a very noisy signal

//variables for tuning
int correctFrequency;//the correct frequency for the string being played
```

```
void setup(){

  Serial.begin(9600);

  //LED pins
  pinMode(7,OUTPUT);
  pinMode(6,OUTPUT);
  pinMode(5,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(2,OUTPUT);
  pinMode(A3,OUTPUT);
  pinMode(A4,OUTPUT);
  pinMode(A5,OUTPUT);
  pinMode(A1,OUTPUT);
  pinMode(A2,OUTPUT);
  pinMode(8,OUTPUT);
  pinMode(9,OUTPUT);


  //Beginning LED sequence
  digitalWrite(7,1);
  digitalWrite(6,1);
  digitalWrite(5,1);
  digitalWrite(4,1);
  digitalWrite(3,1);
  digitalWrite(2,1);
  digitalWrite(8,1);
  analogWrite(A1,255);
  delay(500);
  digitalWrite(9,1);
  analogWrite(A2,255);
  delay(500);
  digitalWrite(A5,255);
  analogWrite(A3,255);
  delay(500);
  analogWrite(A4,255);
  delay(500);



  cli();//disable interrupts

  //set up continuous sampling of analog pin 0 at 38.5kHz
```

```
  //clear ADCSRA and ADCSRB registers
  ADCSRA = 0;
  ADCSRB = 0;

  ADMUX |= (1 << REFS0); //set reference voltage
  ADMUX |= (1 << ADLAR); //left align the ADC value- so we can read highest 8 bits from ADCH
register only

  ADCSRA |= (1 << ADPS2) | (1 << ADPS0); //set ADC clock with 32 prescaler-
16mHz/32=500kHz
  ADCSRA |= (1 << ADATE); //enabble auto trigger
  ADCSRA |= (1 << ADIE); //enable interrupts when measurement complete
  ADCSRA |= (1 << ADEN); //enable ADC
  ADCSRA |= (1 << ADSC); //start ADC measurements

  sei();//enable interrupts
}

ISR(ADC_vect) {//when new ADC value ready

  PORTB &= B11101111;//set pin 12 low
  prevData = newData;//store previous value
  newData = ADCH;//get value from A0
  if (prevData < 127 && newData >=127){//if increasing and crossing midpoint
    newSlope = newData - prevData;//calculate slope
    if (abs(newSlope-maxSlope)<slopeTol){//if slopes are ==
      //record new data and reset time
      slope[index] = newSlope;
      timer[index] = time;
      time = 0;
      if (index == 0){//new max slope just reset
        PORTB |= B00010000;//set pin 12 high
        noMatch = 0;
        index++;//increment index
      }
      else if (abs(timer[0]-timer[index])<timerTol && abs(slope[0]-newSlope)<slopeTol){//if timer
duration and slopes match
        //sum timer values
        totalTimer = 0;
        for (byte i=0;i<index;i++){
          totalTimer+=timer[i];
        }
        period = totalTimer;//set period
```

```
        //reset new zero index values to compare with
        timer[0] = timer[index];
        slope[0] = slope[index];
        index = 1;//set index to 1
        PORTB |= B00010000;//set pin 12 high
        noMatch = 0;
      }
      else{//crossing midpoint but not match
        index++;//increment index
        if (index > 9){
          reset();
        }
      }
    }
    else if (newSlope>maxSlope){//if new slope is much larger than max slope
      maxSlope = newSlope;
      time = 0;//reset clock
      noMatch = 0;
      index = 0;//reset index
    }
    else{//slope not steep enough
      noMatch++;//increment no match counter
      if (noMatch>9){
        reset();
      }
    }
  }

  time++;//increment timer at rate of 38.5kHz

  ampTimer++;//increment amplitude timer
  if (abs(127-ADCH)>maxAmp){
    maxAmp = abs(127-ADCH);
  }
  if (ampTimer==1000){
    ampTimer = 0;
    checkMaxAmp = maxAmp;
    maxAmp = 0;
  }

}

void reset(){//clean out some variables
  index = 0;//reset index
```

```
  noMatch = 0;//reset match couner
  maxSlope = 0;//reset slope
}

//Turn off 5 out the 6 LEDs for the guitar strings
void otherLEDsOff(int LED1, int LED2,int LED3,int LED4,int LED5){
  digitalWrite(LED1,0);
  digitalWrite(LED2,0);
  digitalWrite(LED3,0);
  digitalWrite(LED4,0);
  digitalWrite(LED5,0);
}

//Determine the correct frequency and light up
//the appropriate LED for the string being played
void stringCheck(){
  if(frequency>70&frequency<90){
    otherLEDsOff(2,3,5,6,7);
    digitalWrite(2,1);
    correctFrequency = 82.4;
  }
  if(frequency>100&frequency<120){
    otherLEDsOff(2,3,4,5,6);
    digitalWrite(3,1);
    correctFrequency = 110;
  }
  if(frequency>135&frequency<155){
    otherLEDsOff(2,3,4,6,7);
    digitalWrite(4,1);
    correctFrequency = 146.8;
  }
  if(frequency>186&frequency<205){
    otherLEDsOff(2,3,5,6,7);
    digitalWrite(5,1);
    correctFrequency = 196;
  }
  if(frequency>235&frequency<255){
    otherLEDsOff(2,4,5,6,7);
    digitalWrite(6,1);
    correctFrequency = 246.9;
  }
  if(frequency>320&frequency<340){
    otherLEDsOff(3,4,5,6,7);
    digitalWrite(7,1);
```

```
    correctFrequency = 329.6;
  }
}

//Compare the frequency input to the correct
//frequency and light up the appropriate LEDS
void frequencyCheck(){
  if(frequency>correctFrequency+1){
    analogWrite(A3,255);
  }
  if(frequency>correctFrequency+4){
    analogWrite(A2,255);
  }
  if(frequency>correctFrequency+6){
    analogWrite(A1,255);
  }
  if(frequency<correctFrequency-1){
    analogWrite(A5,255);
  }
  if(frequency<correctFrequency-4){
    digitalWrite(9,1);
  }
  if(frequency<correctFrequency-6){
    digitalWrite(8,1);
  }
  if(frequency>correctFrequency-1&frequency<correctFrequency+1){
    analogWrite(A4,255);
  }
}

void allLEDsOff(){
  digitalWrite(2,0);
  digitalWrite(3,0);
  digitalWrite(4,0);
  digitalWrite(5,0);
  digitalWrite(6,0);
  digitalWrite(7,0);
  digitalWrite(8,0);
  digitalWrite(9,0);
  analogWrite(A1,0);
  analogWrite(A2,0);
  analogWrite(A3,0);
  analogWrite(A4,0);
  analogWrite(A5,0);
```

```
}

void loop(){

  allLEDsOff();

  if (checkMaxAmp>ampThreshold){
    frequency = 38462/float(period);//calculate frequency timer rate/period
  }

  stringCheck();
  frequencyCheck();

  delay(100);

}
```