



DE JONGE ONDERZOEKERS

Kijk, doe, ontdek /

- Les 17: simpele melodie
- Les 18: 7-pin-piano
- Les 19: 1-pin-7-parallelle_werstanden-piano
- Les 20: 1-pin-7-werstanden-in-serie-piano

Figure 1: Boek 5: Muziek
1

Contents

Voorwoord	1
Les 17: Simpele Melodie	2
Les 18: 7-Pin Piano	18
Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano	26
Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano	40

Voorwoord



Figure 1: Het logo van De Jonge Onderzoekers

Dit is het boek van de Arduino cursus. Een Arduino is een machine die je kunt programmeren. Dit boek leert je hoe je elektronica op de Arduino aansluit, en hoe je deze programmeert.

Over dit boek

Dit boek heeft een CC-BY-NC-SA licensie.



Figure 2: De licensie van dit boek

(C) Arduino cursus Groningen 2017

Het is nog een beetje een slordig boek. Er zitten tiepvauten in en de opmaak is niet altijd *even mooi*.

Daarom staat dit boek op een GitHub. Om precies te zijn, op <https://github.com/richelbilderbeek/ArduinoCourse>. Hierdoor kan iedereen die dit boek te slordig vindt minder slordig maken.

Les 17: Simple Melodie

In deze les gaan we een simpele melodie maken, namelijk Vader Jacob!



Figure 3: ‘Vader Jacob’ was een hit in 1975 van de Nederlandse band H2OR

Les 17: Simpele Melodie: Opdracht 1

Sluit figuur ‘Aansluiten van een speaker’ aan.

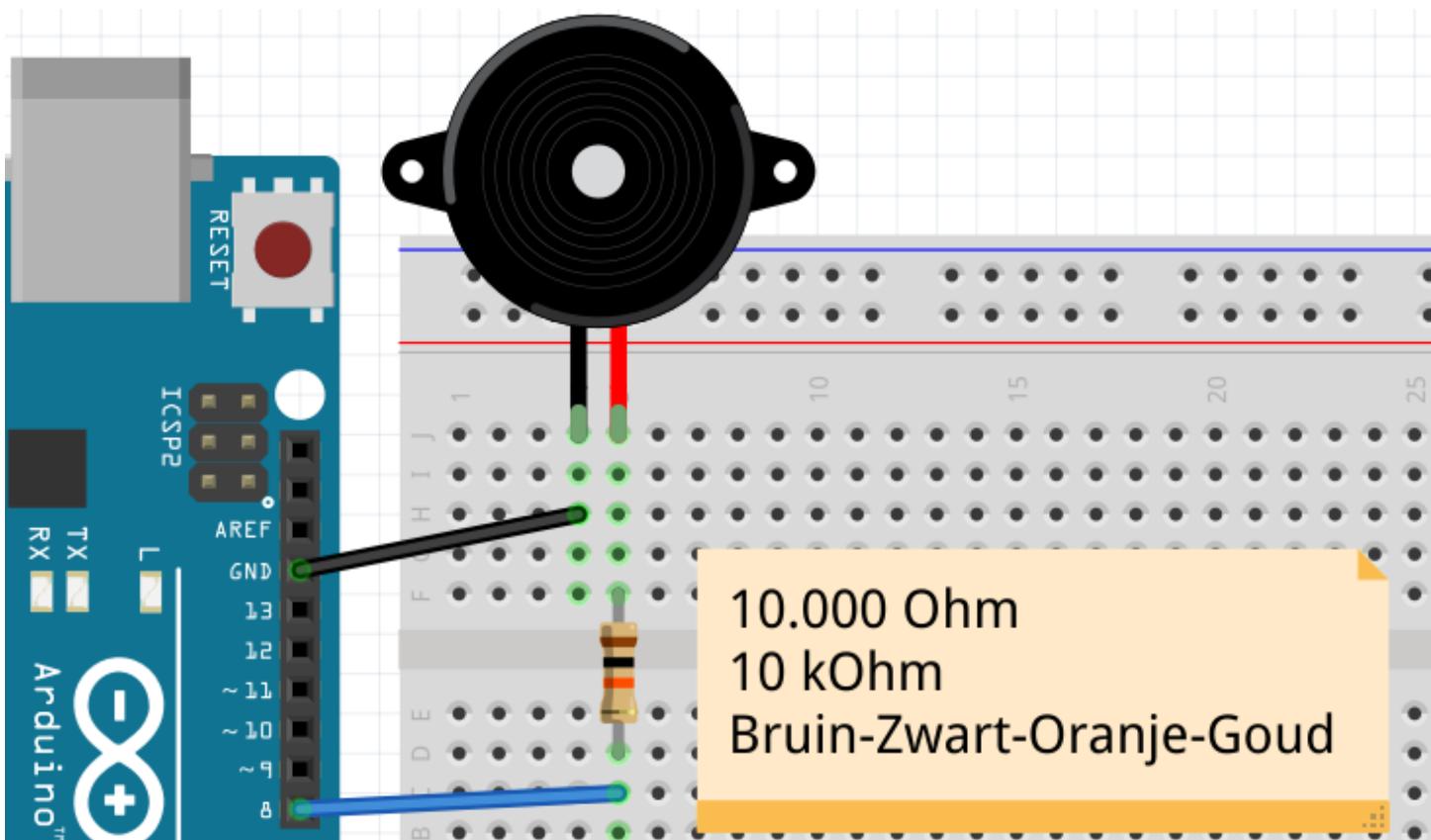


Figure 4: Aansluiten van een speaker

Zet deze code op je Arduino:

```
const int speaker_pin = 8;

void setup()
{
    tone(speaker_pin, 131, 250); // Va
    delay(300);
}

void loop()
{}
```

Wat hoor je?

Les 17: Simpele Melodie: Oplossing 1

Je hoort de eerste noot van Vader Jacob!

In figuur ‘De eerste noot van Vader Jacob’ zie je de eerste noot als bladmuziek. Onder de noot staat de tekst, daaronder de toonhoogte in Hertz.



131

Figure 5: De eerste noot van Vader Jacob



131 Hertz is de toonhoogte van de noot C.



tone(..., 131, ...); (zingt) do!

Les 17: Simpele Melodie: Opdracht 2

De eerste noot van Vader Jacob heeft een toonhoogte van 131 Hertz. De tweede noot van Vader Jacob heeft een toonhoogte van 147 Hertz. Programmeer de eerste twee noten van Vader Jacob.

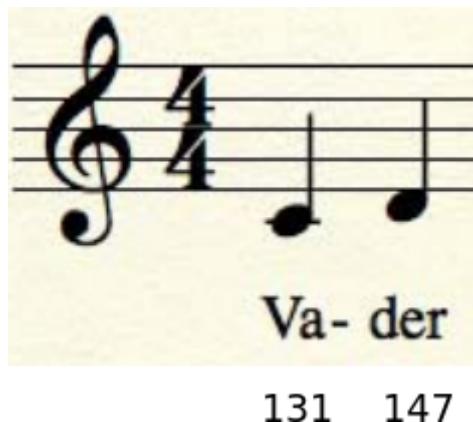


Figure 6: De eerste twee noten van Vader Jacob



Figure 7: Nee, het nummer gaat niet over hem

Les 17: Simpele Melodie: Oplossing 2

```
const int speaker_pin = 8;

void setup()
{
    tone(speaker_pin, 131, 250); // Va
    delay(300);
    tone(speaker_pin, 147, 250); // der
    delay(300);
}

void loop()
{
```



147 Hertz is de toonhoogte van de noot D.



tone(..., 147, ...); (zingt) re!

Les 17: Simpele Melodie: Opdracht 3

De derde noot van Vader Jacob heeft een toonhoogte van 165 Hertz. Programmeer de eerste drie noten van Vader Jacob.

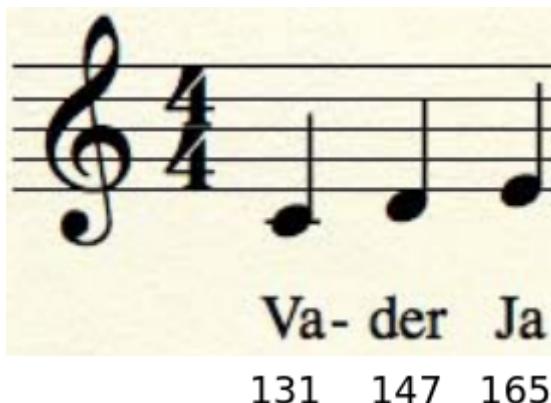


Figure 8: De eerste drie noten van Vader Jacob



Figure 9: Een echte Groninger eindigt de meeste zinnen met 'ja'

Les 17: Simpele Melodie: Oplossing 3

```
const int speaker_pin = 8;

void setup()
{
    tone(speaker_pin, 131, 250); // Va
    delay(300);
    tone(speaker_pin, 147, 250); // der
    delay(300);
    tone(speaker_pin, 165, 250); // Ja
    delay(300);
}

void loop()
{
```



165 Hertz is de toonhoogte van de noot E.



tone(..., 165, ...); (zingt) mi!

Les 17: Simpele Melodie: Opdracht 4

De vierde noot van Vader Jacob heeft dezelfde toonhoogte als de eerste. Programmeer de vierde noot van Vader Jacob.



Figure 10: De eerste vier noten van Vader Jacob



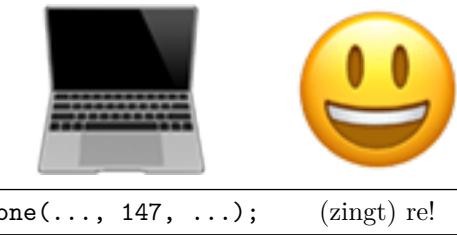
Figure 11: Jacob Black kan in een wolf veranderen

Oplossing 4

```
const int speaker_pin = 8;

void setup()
{
    tone(speaker_pin, 131, 250); // Va
    delay(300);
    tone(speaker_pin, 147, 250); // der
    delay(300);
    tone(speaker_pin, 165, 250); // Ja
    delay(300);
    tone(speaker_pin, 131, 250); // cob
    delay(300);
}

void loop()
{
```



tone(..., 147, ...); (zingt) re!

Les 17: Simpele Melodie: Opdracht 5

De vijfde, zesde, zevende en achtste noot zijn dezelfde als de eerste vier. Programmeer dit.

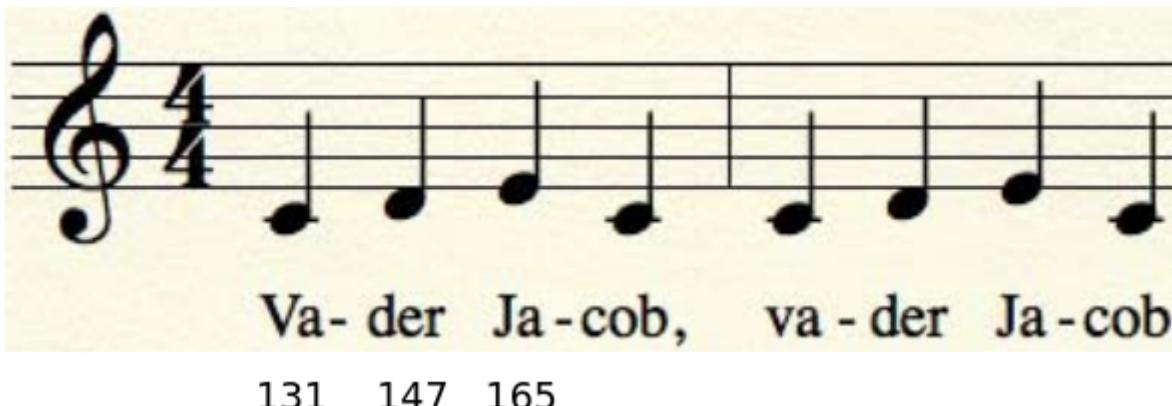


Figure 12: De eerste acht noten van Vader Jacob

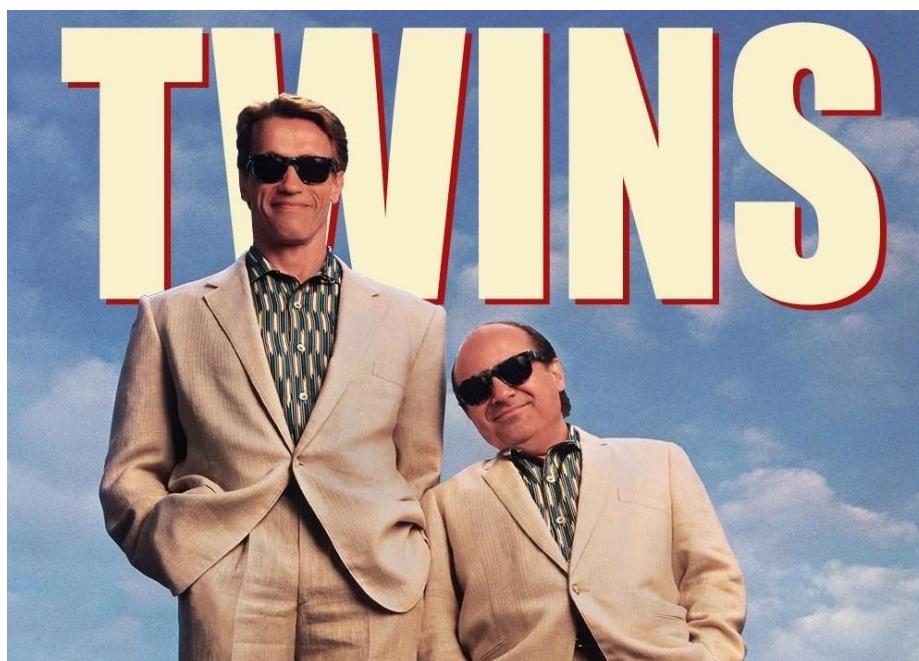


Figure 13: Een tweeling is een soort van herhaalde eenling

Les 17: Simpele Melodie: Oplossing 5

```
const int speaker_pin = 8;

void setup()
{
    tone(speaker_pin, 131, 250); // Va
    delay(300);
    tone(speaker_pin, 147, 250); // der
    delay(300);
    tone(speaker_pin, 165, 250); // Ja
    delay(300);
    tone(speaker_pin, 131, 250); // cob
    delay(300);
    tone(speaker_pin, 131, 250); // Va
    delay(300);
    tone(speaker_pin, 147, 250); // der
    delay(300);
    tone(speaker_pin, 165, 250); // Ja
    delay(300);
    tone(speaker_pin, 131, 250); // cob
    delay(300);
}

void loop()
{}
```



Je mag de herhaling ook in een **for** loop zetten!



```
for (int i = 0; i <  
    2; ++i) { ... }
```

'Doe wat tussen accolades staat twee keer'

Les 17: Simpele Melodie: Opdracht 6

Nu komt twee keer ‘Slaapt gij nog’. ‘Slaapt’ dezelfde hoogte als ‘Ja’, ‘gij’ is 175 Hertz, ‘nog’ is 196 Hertz. Tot nu toe duurden alle noten 250 milliseconden. De derde noot, ‘nog’ moet 500 milliseconden duren.



Figure 14: Slaapt gij nog

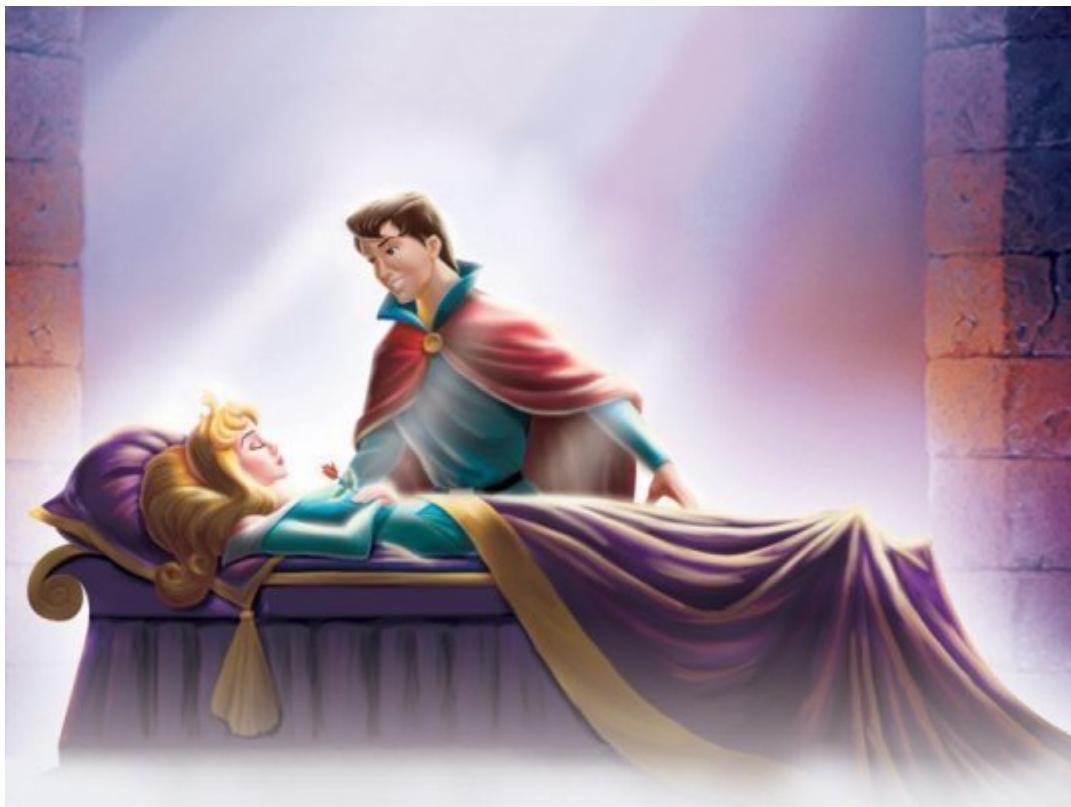


Figure 15: Sommige mensen zijn erg beroemd geworden door ... te slapen?

Les 17: Simpele Melodie: Oplossing 6

In de code staat nu //.... Dit betekent dat daar de oude code moet blijven staan.

```
const int speaker_pin = 8;

void setup()
{
    //...
    tone(speaker_pin, 165, 250); // Slaapt
    delay(300);
    tone(speaker_pin, 175, 250); // gij
    delay(300);
    tone(speaker_pin, 196, 500); // nog
    delay(500);
    tone(speaker_pin, 165, 250); // Slaapt
    delay(300);
    tone(speaker_pin, 175, 250); // gij
    delay(300);
    tone(speaker_pin, 196, 500); // nog
    delay(500);
}

void loop()
{
```



Je mag de herhaling ook in een **for** loop zetten!

Les 17: Simpele Melodie: Opdracht 7

Nu komt twee keer ‘Alle klokken luiden’. In de figuur ‘Alle klokken luiden’ staan de toonhoogten. De noten die aan elkaar vastzitten (‘Alle klokken’) duren elk 125 milliseconden.

The image shows musical notation on a staff. The first measure consists of six eighth notes grouped together by vertical stems. The second measure also consists of six eighth notes grouped together by vertical stems. Below the staff, the lyrics 'Al-le klok-ken lui-den,' are written twice, once above each measure. The tempo markings '220' and '131' are positioned under their respective measures.

Al-le klok-ken lui-den,
al-le klok-ken lui-den,

220 131

Figure 16: Alle klokken luiden



Figure 17: Ook met klokken luiden kun je beroemd worden

Les 17: Simpele Melodie: Oplossing 7

```
const int speaker_pin = 8;

void setup()
{
    //...
    tone(speaker_pin, 131, 125); // Al
    delay(125);
    tone(speaker_pin, 147, 125); // le
    delay(125);
    tone(speaker_pin, 165, 125); // klok
    delay(125);
    tone(speaker_pin, 131, 125); // ken
    delay(125);
    tone(speaker_pin, 131, 250); // lui
    delay(300);
    tone(speaker_pin, 147, 250); // den
    delay(300);
}

void loop()
{
```



(zingt) Vader Jacob, Vader Jacob! (maar niet erg goed)

Les 17: Simpele Melodie: Eindopdracht

Maak het liedje Vader Jacob af. Zie figuur ‘Vader Jacob bladmuziek’ hoe de laatste noten moeten.

The image shows three staves of musical notation in G clef, 4/4 time. The first staff consists of eight eighth notes. Below it is the lyrics: 'Va- der Ja-cob, va - der Ja - cob, slaapt gij nog,'. The second staff has six eighth notes followed by two sixteenth-note groups. Below it are the lyrics: 'slaapt gij nog? Al-le klok-ken lui-den, al-le klok-ken lui-den,'. The third staff has six eighth notes, with the last one being a dotted half note. Below it are the lyrics: 'bim bam bom, bim bam bom.' A vertical bar line is positioned between the first and second staves, and another at the end of the third staff.

Figure 18: Vader Jacob bladmuziek

Les 18: 7-Pin Piano

In deze les gaan we een simpele piano maken, die 7 pinnen gebruikt.



Figure 19: Een pino

Les 18: 7-Pin Piano: Opdracht 1

Sluit figuur ‘Een pin’ aan.

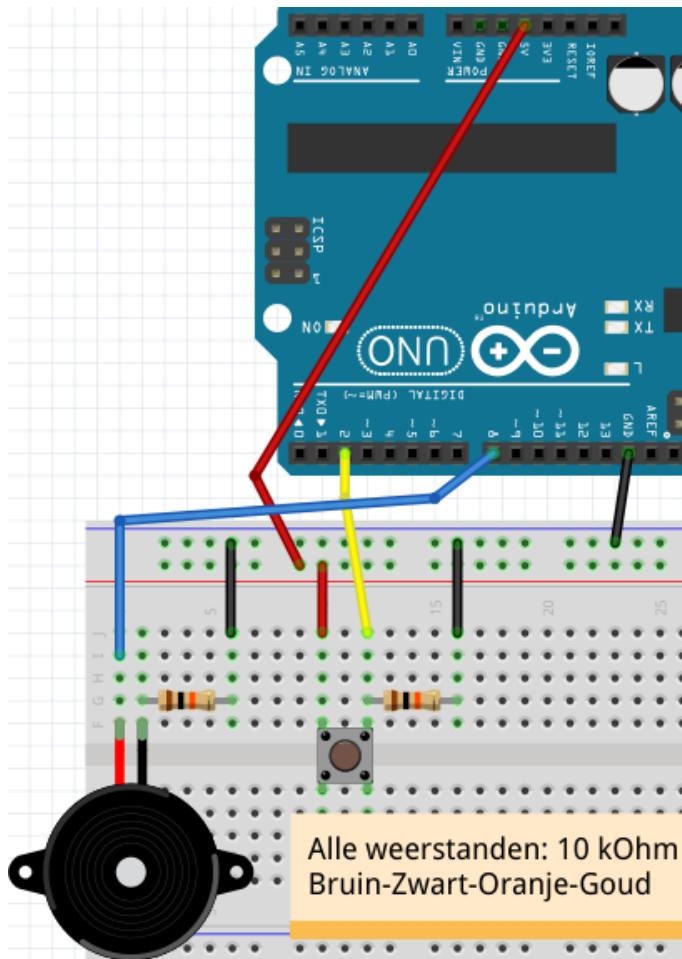


Figure 20: Een pin

Zet deze code op je Arduino:

```
const int speaker_pin = 8;
const int pin_1 = 2;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(pin_1, INPUT);
}

void loop()
{
    if (digitalRead(pin_1) == LOW)
    {
        tone(speaker_pin, 175, 250);
        delay(250);
    }
}
```

We maken een piano. Dit is de eerste toets met een toonhoogte van 175 Hertz. Maar er zit een fout in de code! Repareer de code.

Les 18: 7-Pin Piano: Oplossing 1

```
const int speaker_pin = 8;
const int pin_1 = 2;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(pin_1, INPUT);
}

void loop()
{
    if (digitalRead(pin_1) == HIGH)
    {
        tone(speaker_pin, 175, 250);
        delay(250);
    }
}
```



Ah, `digitalRead` moet HIGH zijn, in plaats van LOW!



Als de code het *wel* deed, heb je GND en 5V omgedraaid

Les 18: 7-Pin Piano: Opdracht 2

Bouw een tweede toets erbij, op pin 3. Deze heeft ook een eigen weestandje nodig. Deze moet een toonhoogte krijgen van 196 Hertz.



Een tweede knop bouwen gaat net als de eerste



De code voor de tweede knop gaat ook net als de eerste

Les 18: 7-Pin Piano: Oplossing 2

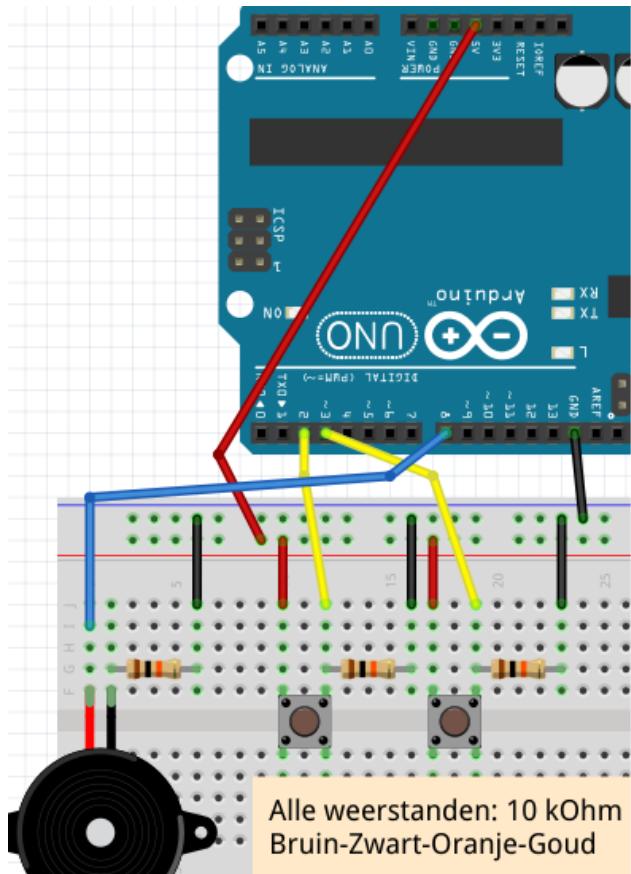


Figure 21: Oplossing 2

```

const int speaker_pin = 8;
const int pin_1 = 2;
const int pin_2 = 3;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(pin_1, INPUT);
    pinMode(pin_2, INPUT);
}

void loop()
{
    if (digitalRead(pin_1) == HIGH)
    {
        tone(speaker_pin, 175, 250);
        delay(250);
    }
    if (digitalRead(pin_2) == HIGH)
    {
        tone(speaker_pin, 196, 250);
        delay(196);
    }
}

```

Les 18: 7-Pin Piano: Opdracht 3

Bouw een derde toets erbij, op pin 4. Deze heeft ook een eigen weerstandje nodig. De toets moet een toonhoogte krijgen van 220 Hertz.



Figure 22: Een echte Grunninger kan nu al los!

Les 18: 7-Pin Piano: Oplossing 3

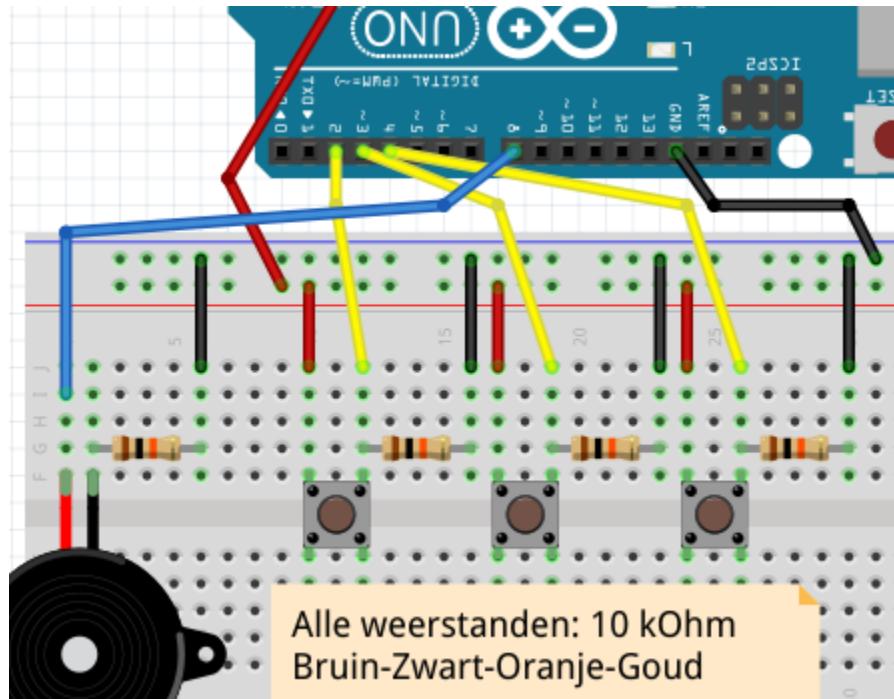


Figure 23: Oplossing 3

```

const int speaker_pin = 8;
const int pin_1 = 2;
const int pin_2 = 3;
const int pin_3 = 4;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(pin_1, INPUT);
    pinMode(pin_2, INPUT);
    pinMode(pin_3, INPUT);
}

void loop()
{
    if (digitalRead(pin_1) == HIGH)
    {
        tone(speaker_pin, 175, 250);
        delay(250);
    }
    if (digitalRead(pin_2) == HIGH)
    {
        tone(speaker_pin, 196, 250);
        delay(196);
    }
    if (digitalRead(pin_3) == HIGH)
    {
        tone(speaker_pin, 220, 250);
        delay(196);
    }
}

```

Les 18: 7-Pin Piano: Eindopdracht

Maak een piano van zeven toetsen. Zie figuur ‘Frequenties’ voor de andere getallen.



Figure 24: Frequenties



Figure 25: Pianisten kunnen los op je piano!

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano

In deze les gaan we een simpele piano maken, die 1 pin gebruikt en 7 parallelle weerstanden.

We bouwen de piano stap voor stap op en testen elke stap apart.

Het uitlezen van de knoppen hebben we al eerder gezien in lesboekje 2, bladzijde 16. Het afspelen van een geluidje hebben we al eerder gezien in lesboekje 3, bladzijde 17.



We beginnen met de middelste toets met frequentie 247. Dan bouwen we naar links 3 toetsen erbij. Daarna bouwen we rechts 3 toetsen erbij.
De frequenties staan in dit plaatje,

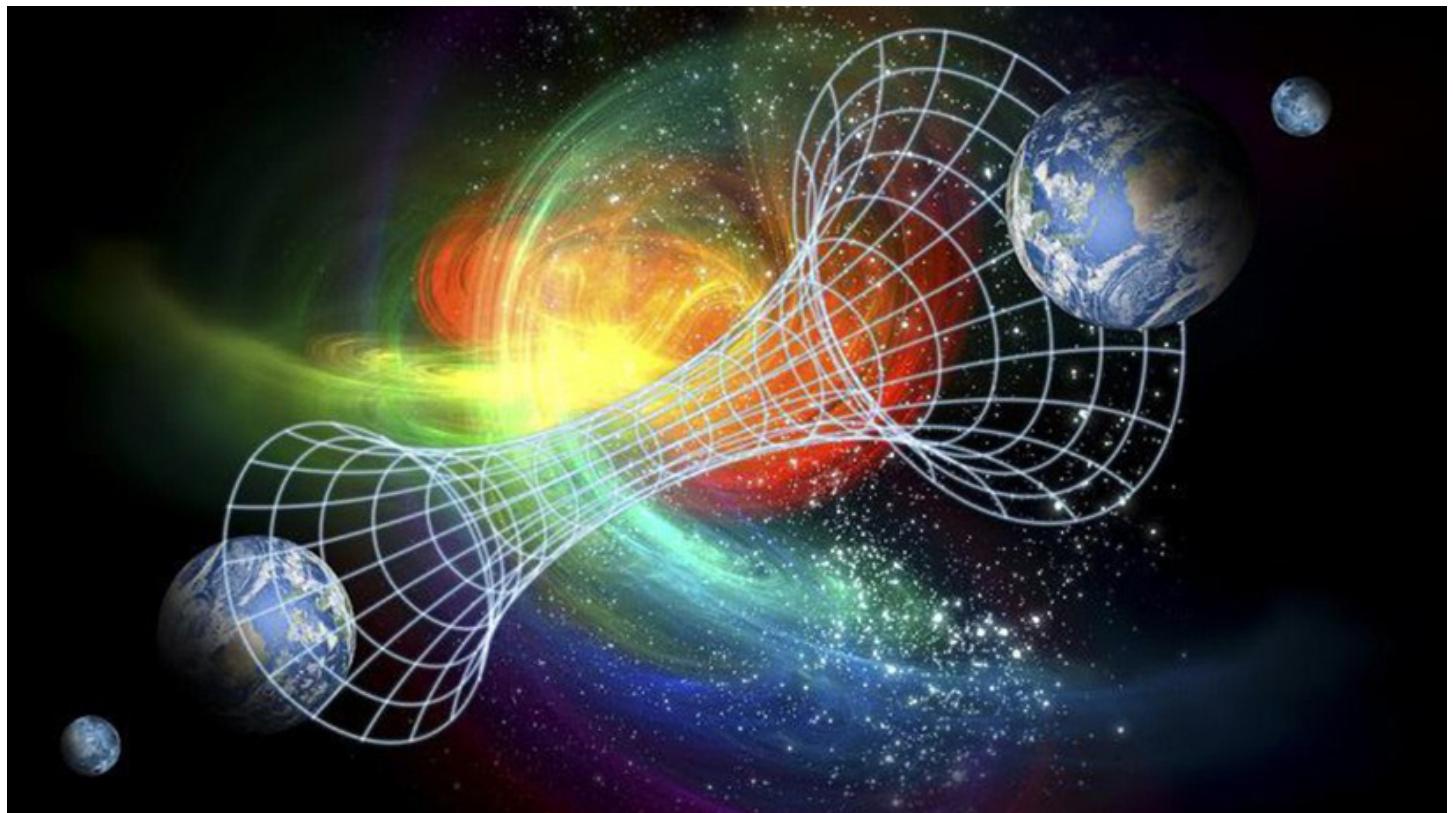


Figure 26: In les 231763256 maken we een parallelle universa piano!

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Opdracht 1 aansluiten

Sluit de eerste knop aan volgens het plaatje. Zet de knop in het midden van je breadboard!

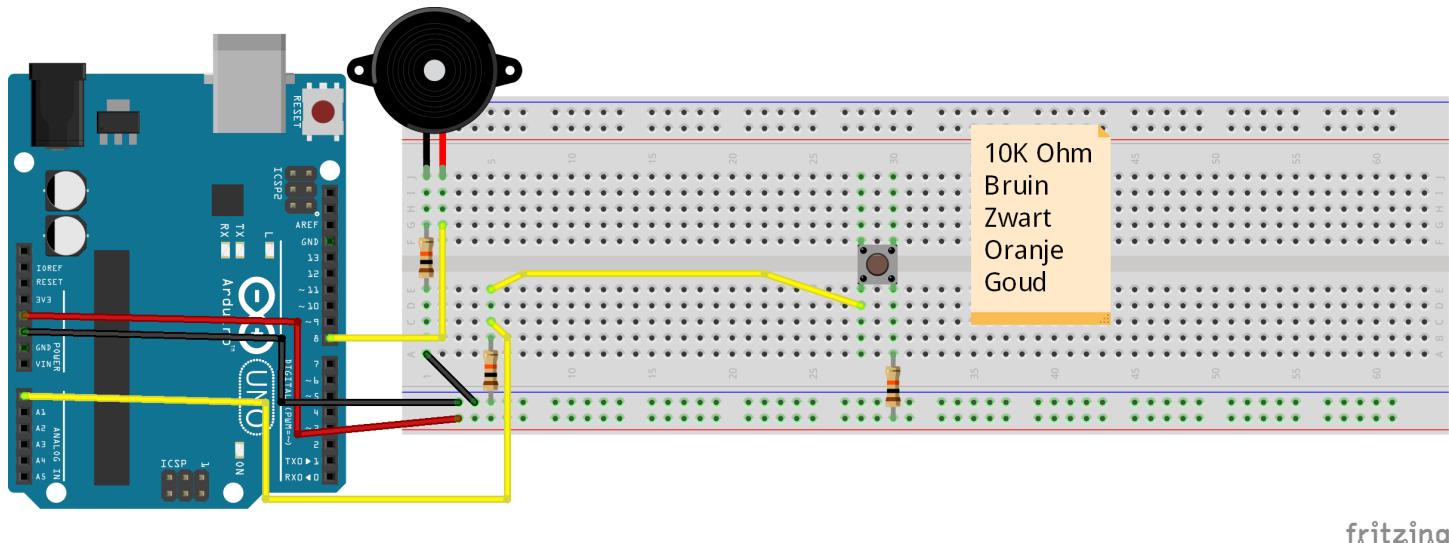


Figure 27: Een pin



De ‘Pull Down’ weerstand zorgt dat pin 2 verbonden is met GND als de knop niet ingedrukt is

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Opdracht 1 code

Zet deze code op je Arduino:

```
const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(piano_pin, INPUT);
    Serial.begin(9600);
}

void loop()
{
    Serial.println(analogRead(piano_pin));
    if (analogRead(piano_pin) > 510)
    {
        tone(speaker_pin, 247);
        delay(250);
        noTone(speaker_pin);
        delay(250);
    }
}
```



'>' betekent 'groter dan'. De waarde A0 is nooit precies een getal. In de seriële monitor lezen we het getal dat bij de knop hoort af en testen dan op een getal dat daar net iets onder ligt.

Krijg je een geluid als je op de knop drukt? Dan kun je door naar opdracht 2.

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Opdracht 2

Sluit een tweede knop aan **links** van de eerste, met twee weerstanden ervoor die parallel staan [dus 5k], zie plaatje.



Twee parallele weerstanden van 10k geeft een weerstand van 5k.



Twee weerstanden van 10k in serie geeft een weerstand van 20k.

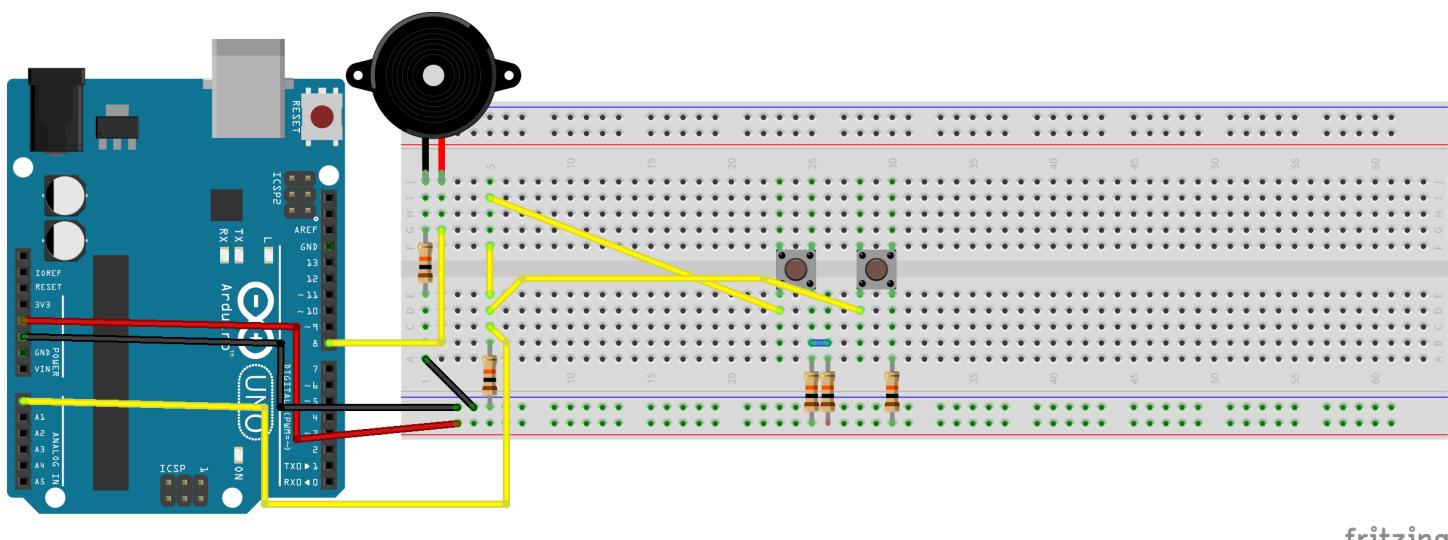


Figure 28: Een pin



Gebruik de seriële monitor om de waarde van de nieuwe knop te bepalen.



De nieuwe waarde is hoger dan de waarde van de andere knop. Het nieuwe if-statement moet bovenaan komen.



Welke frequentie krijgt de nieuwe knop?

Gebruik deze code:



Is het nodig om de hele code opnieuw in te voeren?

```
const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(piano_pin, INPUT);
    Serial.begin(9600);
}

void loop()
{
    const int sensorValue = analogRead(piano_pin);
    Serial.println(sensorValue);
    if (sensorValue > 680)
    {
        tone(speaker_pin, 220);
        delay(250);
        noTone(speaker_pin);
        delay(250);
    }
    else if (sensorValue > 510)
    {
        tone(speaker_pin, 247);
        delay(250);
        noTone(speaker_pin);
        delay(250);
    }
}
```

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Opdracht 3

Bouw nu een derde toets, links van de vorige.

Gebruik nu 3 parallele weerstanden.



Gebruik de seriële monitor om de waarde van de nieuwe knop te bepalen.



Welke frequentie krijgt de nieuwe knop?

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Oplossing 3

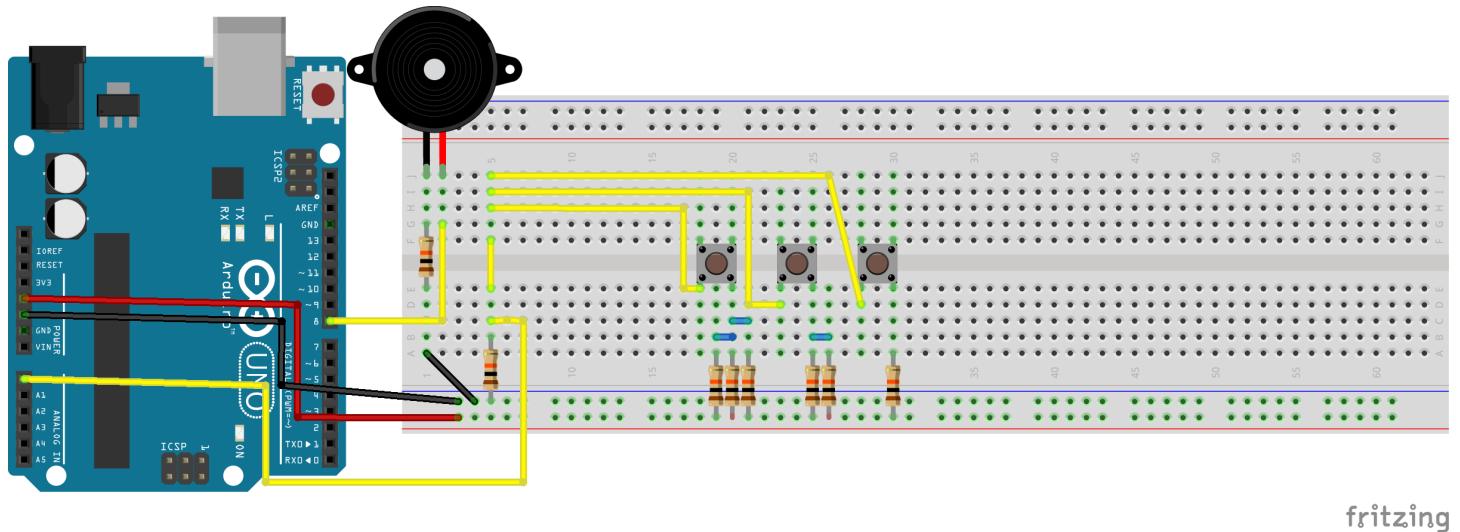


Figure 29: Een pin

```

const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(piano_pin, INPUT);
    Serial.begin(9600);
}

void loop()
{
    const int sensorValue = analogRead(piano_pin);
    Serial.println(sensorValue);
    if (sensorValue > 820)
    {
        tone(speaker_pin, 196);
        delay(250);
        noTone(speaker_pin);
        delay(250);
    }
    else if (sensorValue > 680)
    {
        tone(speaker_pin, 220);
        delay(250);
        noTone(speaker_pin);
        delay(250);
    }
    else if (sensorValue > 510)
    {
        tone(speaker_pin, 247);
        delay(250);
        noTone(speaker_pin);
        delay(250);
    }
}

```

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Opdracht 4

Bouw nu een vierde toets, links van de vorige.

Gebruik nu 4 parallelle weerstanden.



Gebruik de seriële monitor om de waarde van de nieuwe knop te bepalen.



Welke frequentie krijgt de nieuwe knop?

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Oplossing 4

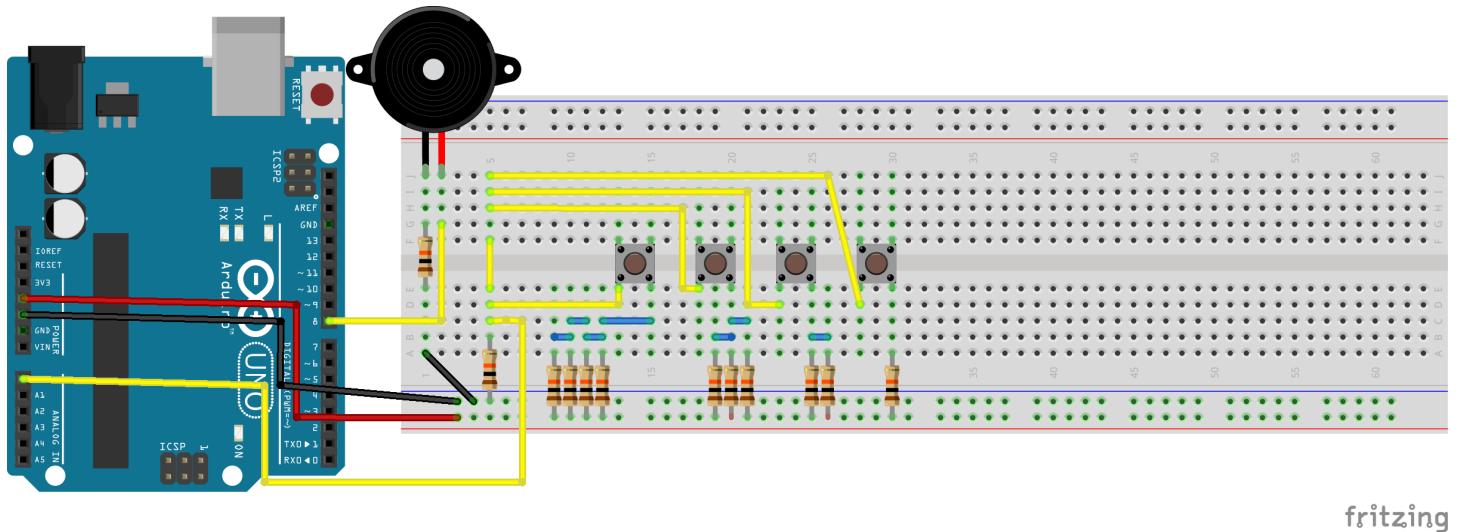


Figure 30: Een pin

```

const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(piano_pin, INPUT);
    Serial.begin(9600);
}

void loop()
{
    //... vorige code hier
    else if (sensorValue > 510)
    {
        tone(speaker_pin, 247);
        delay(250);
        noTone(speaker_pin);
        delay(250);
    }
}

```

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Opdracht 5

Bouw nu de 5e toets rechts van de vorige toetsen. Gebruik nu geen parallele weerstanden, maar 2 in serie geschakelde weerstanden van 10k Ohm.

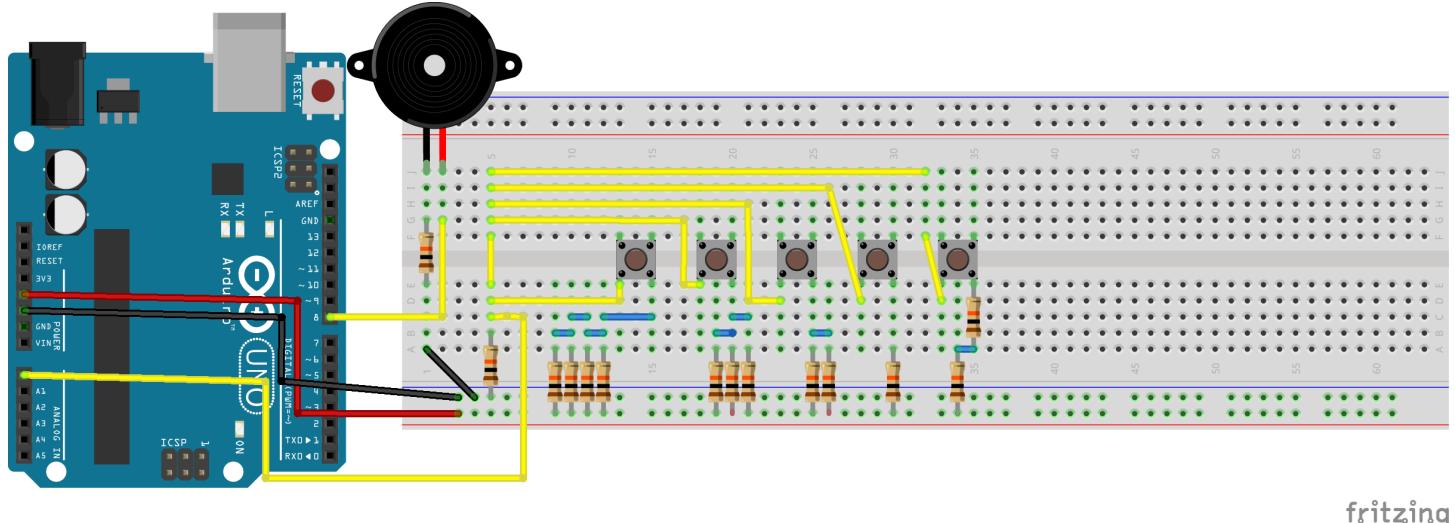


Gebruik de seriële monitor om de waarde van de nieuwe knop te bepalen.



Welke frequentie krijgt de nieuwe knop?

De schakeling komt er zo uit te zien.



fritzing

Figure 31: Een pin

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Oplossing 5

```
const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(piano_pin, INPUT);
    Serial.begin(9600);
}

void loop()
{
    //... vorige code hier
    else if (sensorValue > 310)
    {
        tone(speaker_pin, 262);
        delay(250);
        noTone(speaker_pin);
        delay(250);
    }
}
```



Figure 32: Nu kan elk lid van de succesvolle boyband 5ive een knop van je piano bespelen!

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Opdracht 6

Bouw nu de 6e toets rechts van de vorige toetsen. Gebruik ook nu geen parallele weerstanden, maar 3 in serie geschakelde weerstanden van 10k Ohm.



Gebruik de seriële monitor om de waarde van de nieuwe knop te bepalen.



Welke frequentie krijgt de nieuwe knop?

De schakeling komt er zo uit te zien.

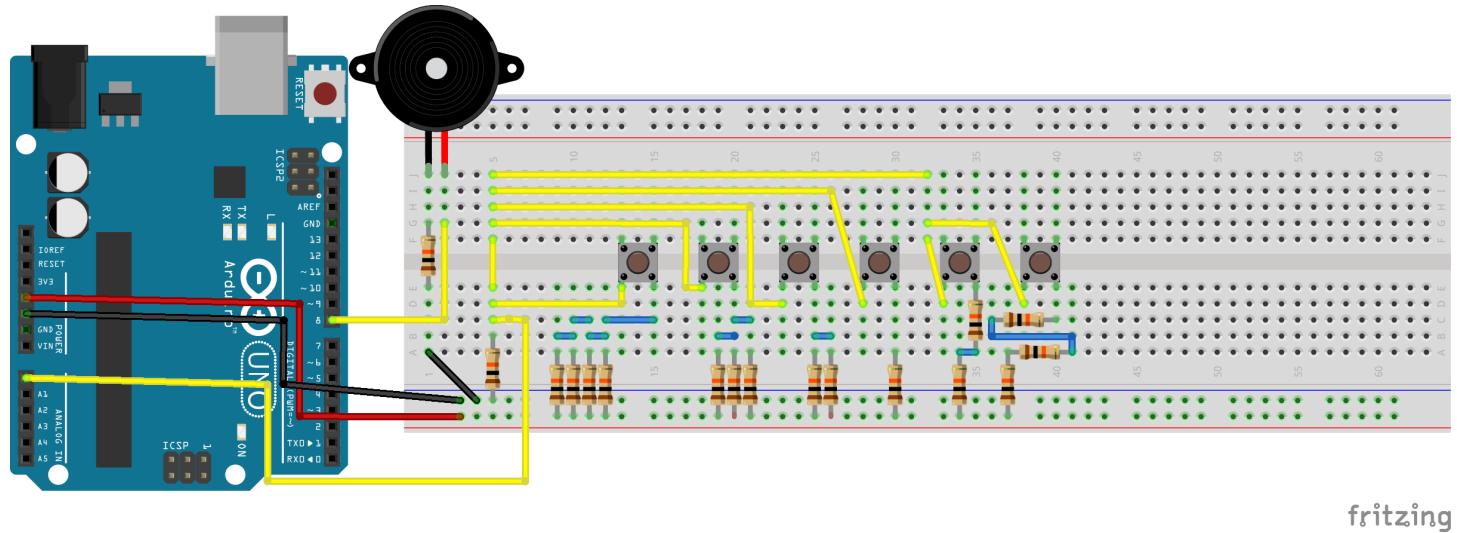


Figure 33: Een pin

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Oplossing 6

```
const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    pinMode(speaker_pin, OUTPUT);
    pinMode(piano_pin, INPUT);
    Serial.begin(9600);
}

void loop()
{
    //... vorige code hier
    else if (sensorValue > 210)
    {
        tone(speaker_pin, 294);
        delay(250);
        noTone(speaker_pin);
        delay(250);
    }
}
```



Figure 34: Het subphylum van de zespotigen (hexapoda) kan nu met een poot per toets op jouw piano spelen!

Les 19: 1-Pin-7-Parallelle-Weerstanden-Piano: Eindopdracht

Maak een piano van zeven toetsen af door de 7e knop rechts bij te zetten. Gebruik nu 4 in serie geschakelde weerstanden van 10 kOhm.



Gebruik de seriële monitor om de waarde van de nieuwe knop te bepalen.



Welke frequentie krijgt de nieuwe knop?



Figure 35: Wibi Soerjadi gebruikt meestal meer dan zeven toetsen

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano

In deze les gaan we een piano maken die maar een pin nodig heeft en 8 weerstanden.

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \dots = \frac{\pi}{4}$$

Figure 36: Met deze serie van breuken kun je pi berekenen



pi is een beroemd getal tussen de drie en vier



Dit heeft niets met piano's te maken...

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Opdracht 1

Bouw ‘Opdracht 1’ na:

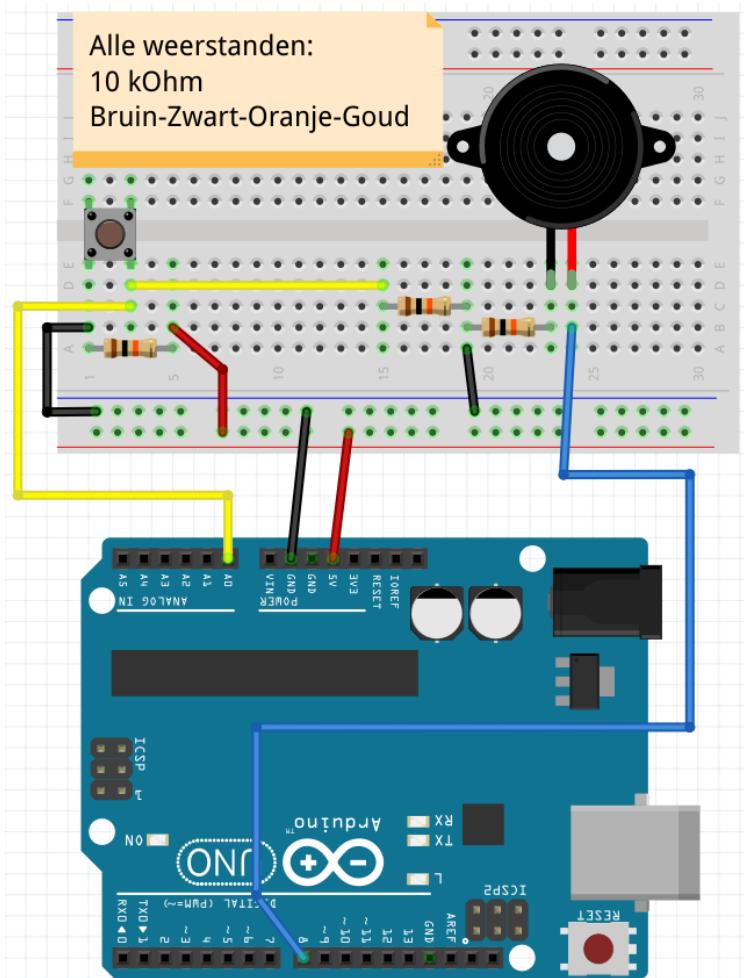


Figure 37: Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Opdracht 1

Zet deze code op je Arduino:

```
const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    Serial.begin(9600);
    pinMode(A0, INPUT);
}

void loop()
{
    const int piano_waarde = analogRead(piano_pin);
    Serial.println(piano_waarde);
    delay(100);
}
```

- Welke waarde krijgt de Arduino als de knop is ingedrukt?
- Welke waarde krijgt de Arduino als de knop niet is ingedrukt?

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Oplossing 1

Als de knop is ingedrukt krijg je waarde 0. Als de knop niet is ingedrukt krijg je waarde 1023.

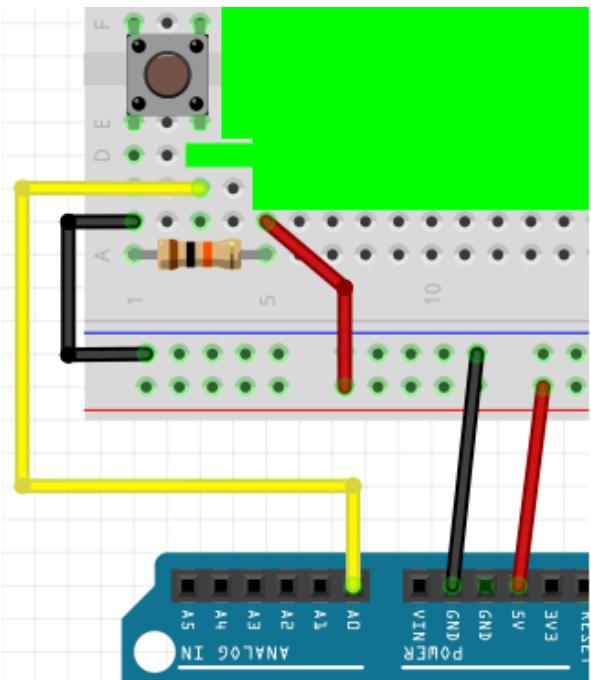


Figure 38: Als de knop is ingedrukt

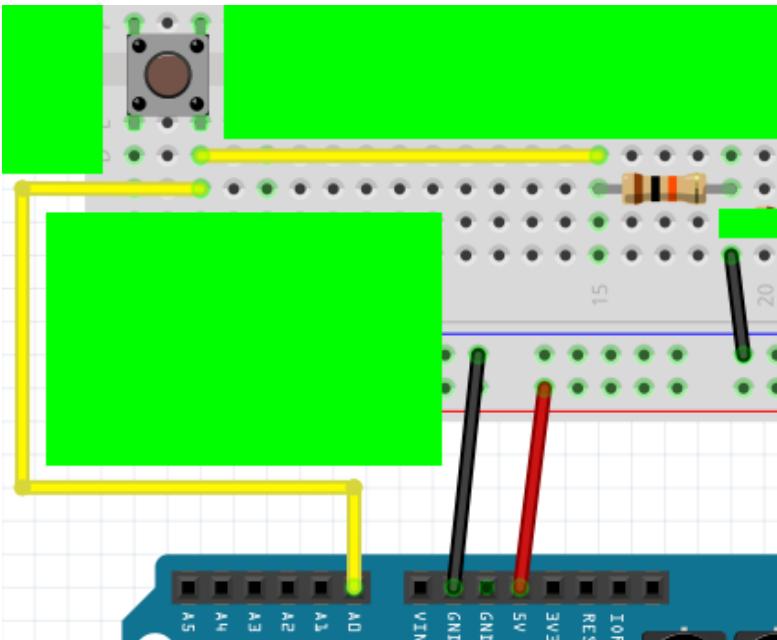


Figure 39: Als de knop niet is ingedrukt

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Opdracht 2

Als de knop is ingedrukt, laat dan de piezo 250 milliseconden piepen op een frequentie van 175 Hertz. Laat de Arduino aan het eind van loop 1 milliseconde wachten.



Als je dit niet meer weet, spiek dan bij de vorige les!

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Oplossing 2

```
const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    Serial.begin(9600);
    pinMode(A0, INPUT);
}

void loop()
{
    const int piano_waarde = analogRead(piano_pin);
    Serial.println(piano_waarde);
    if (piano_waarde < 150)
    {
        tone(speaker_pin, 175, 250);
        delay(250);
    }
    delay(1);
}
```



Oh ja, zo moest het!

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Opdracht 3

Bouw het volgende na:

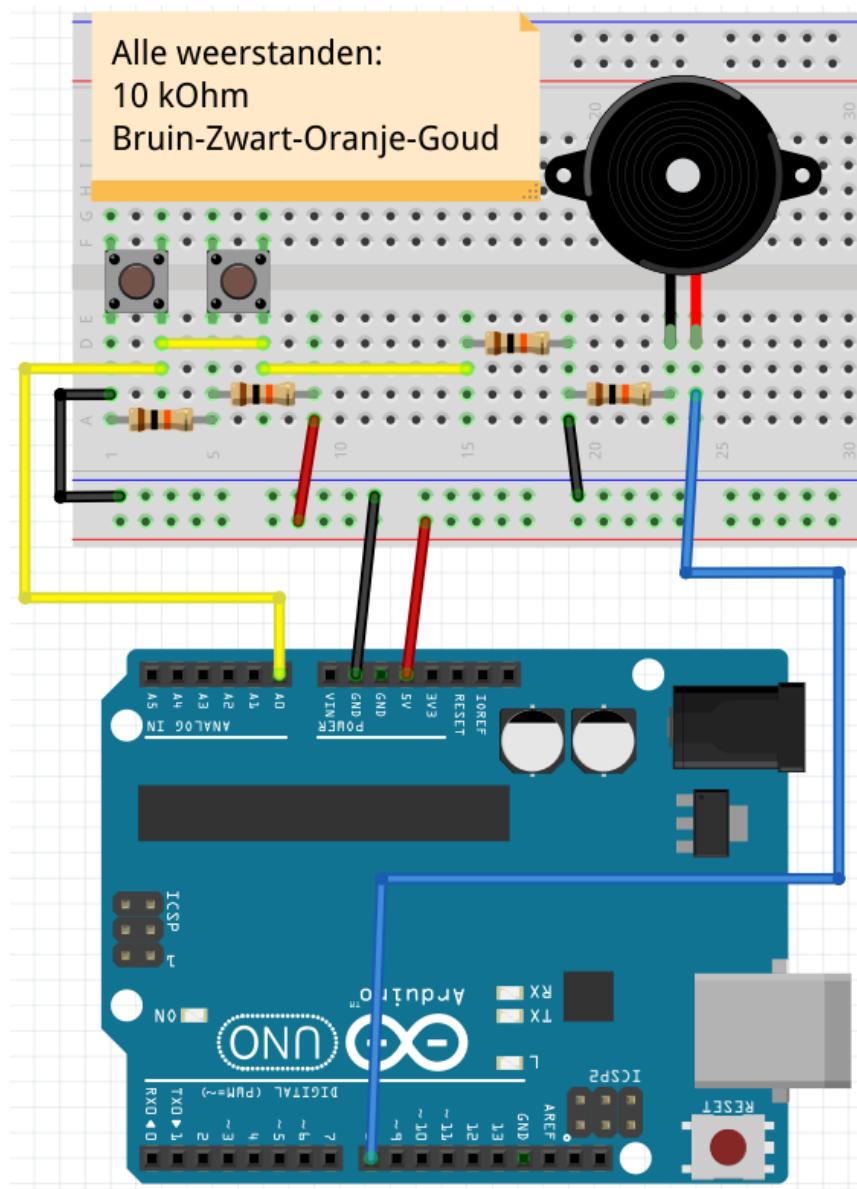


Figure 40: Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Opdracht 3

Programmeer dat de tweede knop een toon maakt van 196 Hertz.

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Oplossing 3

```
const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    Serial.begin(9600);
    pinMode(A0, INPUT);
}

void loop()
{
    const int piano_waarde = analogRead(piano_pin);
    Serial.println(piano_waarde);
    if (piano_waarde < 150)
    {
        tone(speaker_pin, 175, 250);
        delay(250);
    }
    else if (piano_waarde < 300)
    {
        tone(speaker_pin, 196, 250);
        delay(250);
    }
    delay(1);
}
```

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Opdracht 4

Bouw het volgende na:

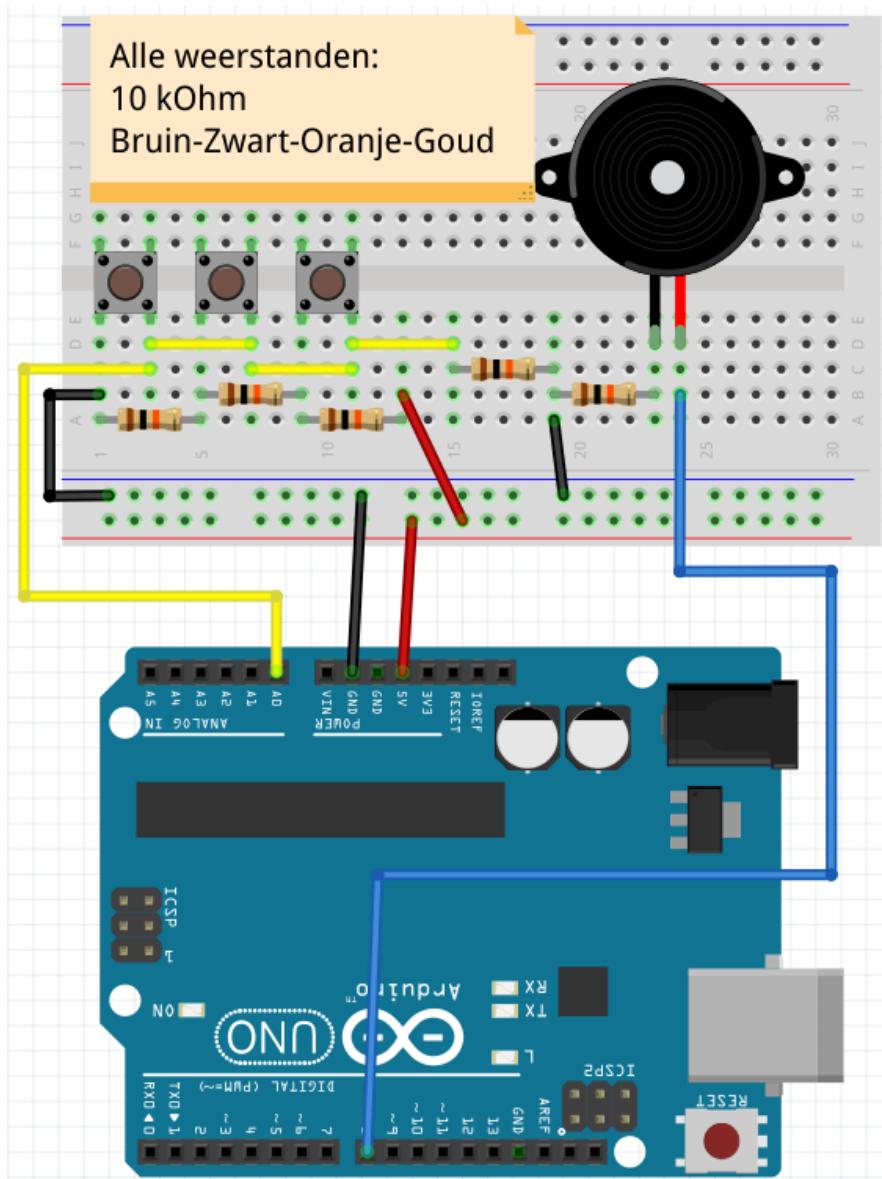


Figure 41: Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Opdracht 4

Programmeer dat de derde knop een toon maakt van 220 Hertz.

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Oplossing 4

```
const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    Serial.begin(9600);
    pinMode(A0, INPUT);
}

void loop()
{
    const int piano_waarde = analogRead(piano_pin);
    Serial.println(piano_waarde);
    if (piano_waarde < 150)
    {
        tone(speaker_pin, 175, 250);
        delay(250);
    }
    else if (piano_waarde < 300)
    {
        tone(speaker_pin, 196, 250);
        delay(250);
    }
    else if (piano_waarde < 450)
    {
        tone(speaker_pin, 220, 250);
        delay(250);
    }
    delay(1);
}
```

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Opdracht 5

Bouw nu zelf een vierde knop.

Programmeer dat de vierde knop een toon maakt van 247 Hertz.



Figure 42: Een NES controller heeft ook 4 knoppen

Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Oplossing 5

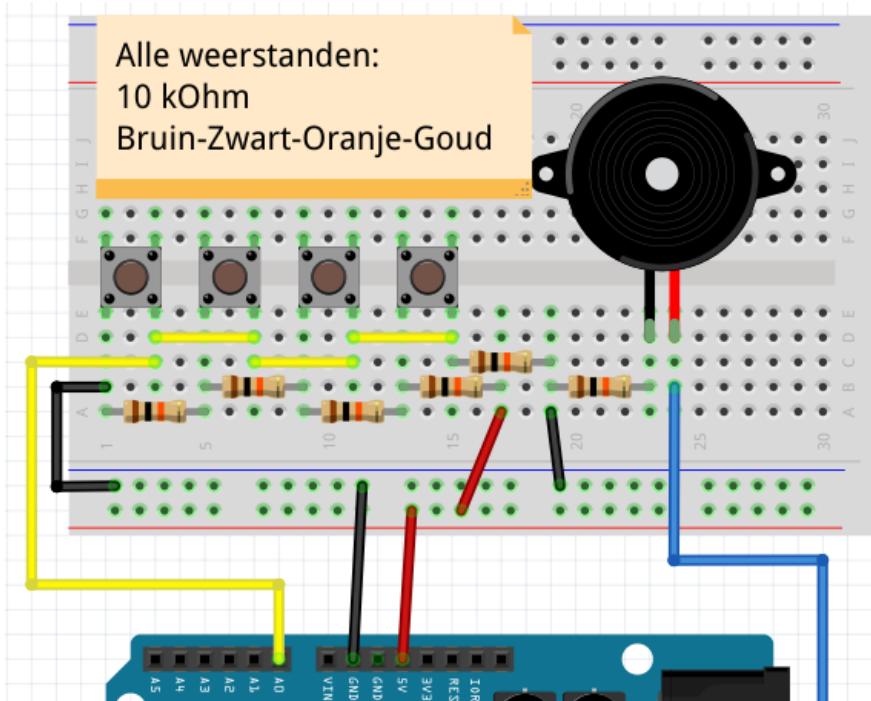


Figure 43: Les 20: 1-Pin-7-Weerstanden-In-Serie-Piano: Opdracht 5

```

const int speaker_pin = 8;
const int piano_pin = A0;

void setup()
{
    Serial.begin(9600);
    pinMode(A0, INPUT);
}

void loop()
{
    // ... vorige code hier
    else if (piano_waarde < 600)
    {
        tone(speaker_pin, 247, 250);
        delay(250);
    }
    delay(1);
}

```

Eindopdracht

Maak de piano af met zeven knoppen. De laatste tonen zijn 262, 294 en 330 Hertz.



Figure 44: Amina Figarova heeft echt niet genoeg aan zeven toetsen