Weekendschool – Programmeren – Les 3 – Begeleiders instructie

# introductie

Dit is de beschrijving van de derde les programmeren. Het geeft achtergrond informatie, maar ook de uitgewerkte stappen.

# Hoe leerlingen helpen?

Deze les probeert een balans uit tussen programmadelen te laten kopieren en plakken en om ze zelf te laten bedenken hoe het moet. Dat is mogelijk nog niet optimaal.

In deze begeleiding staan daarom ook de uitwerkingen van de stappen. Als het helemaal niet lukt, dan kun je ze zo’n stap laten overtikken. Maar blijf er wel bij om na te gaan dat ze snappen wat ze doen.

# Doen voor de les begint: sluit de juiste Raspberry Pi aan

We gebruiken een andere Raspberry Pi. Wat eigenlijk nodig is, is een ander image, maar het is handiger de Raspberry Pi te verwisselen dan om het kaartje te verwisselen. Het zou allemaal horen te werken met een enkel image, echter bij het nieuwe release van Raspian wil Arduino niet meer programma's naar de Arduino Nano clones downloaden. Daarom gebruiken we daar nog een oud image voor.

# Opmerkingen per lesonderdeel

## Uittesten

Als er iets anders staat dan 

dan moet je in Arduino gaan naar

Extra → Board → Arduino Nano w/ Atmega328

en

Extra → Seriele poort → /dev/ttyUSB0

## Nu breiden we uit en … zoeken we de fout!

Oplossing: er mist een punt-komma achter de regel

delay(500)

## Jullie eerste eigen programmaregels

Dit zijn de regels die ze moeten toevoegen na het laten branden van de rode LED.

digitalWrite(LEDGroenRechts, HIGH); // Laat de rechter LED branden

delay(500); // Wacht 500 milliseconde (ms) = 0,5 seconde

digitalWrite(LEDGroenRechts, LOW); // Doe de rechter LED weer uit

delay(200); // Wacht 200 milliseconde (ms) = 0,2 seconde

## Hoe wordt de puntentelling vertoond?

Dit is het programma op dit moment. Met geel is aangegeven waar de regels erbij kwamen. Zie hieronder voor toelichting

#include <TM1637Display.h>

// Pennen van de Arduino. We geven ze een naam zodat we in het programma een naam kunnen

// gebruiken (dat is makkelijker) en omdat we dan, als we een wijziging moeten maken

// (bijvoorbeeld een knop op een andere pin aansluiten), slechts op 1 plek een wijziging

// hoeven te maken.

// Knoppen

#define KnopLinks 3 // De linker knop zit op pen 3

#define KnopRechts 12 // De rechter knop zit op pen 12

#define KnopKlein 4 // De kleine knop Linksonder zit op pen 4

// LEDs

#define LEDGroenLinks 2 // De groene LED Links is aangesloten op pen 2

#define LEDGroenRechts 11 // De rechter groene LED is aangesloten op pen 11

#define LEDRood 5 // De rode LED zit op pen 5

#define LEDNano 13 // De kleine LED op de Arduino Nano zelf

// Analoge ingangen

#define PotMeter A0 // De potentiometer (potmeter) is een regelbare

// weerstand. Die is aangesloten op pen A0

// De 4 cijfers worden bestuurd door een chip TM1637 die aan de onderkant zit

// (je kunt het transparante plaatje omdraaien en dan zie je de chip met het nummer.

// Je ziet dan ook dat er 4 pennen zijn: GND, VCC, CLK en DIO.

// VCC en GND zijn de draden voor de voeding en die gaan naar resp. +5 Volt en GND op

// de Arduino zodat de chip en het blok cijfers spanning krijgen.

// DIO staat voor Digitale Input / Output. Over deze draad gaat de data die vertelt welke

// cijfers moeten oplichten.

// CLK staat voor klok (clock in het Engels). Die vertelt wanneer de TM1637 chip op de DIO draad de data moet bekijken.

#define CLK 8 // Dit is de klok

#define DIO 9 // Dit is de digitale input / output

TM1637Display VierCijfers(CLK, DIO);

// ==========================================================================================

// Wat tussen de accoulades achter setup staat wordt uitgevoerd nadat de Arduino is gestart

void setup()

{ // Dit is openingsaccoulade die bij setup hoort.

pinMode(KnopLinks, INPUT\_PULLUP); // De pen waaraan de linker knop zit is voor invoer

// (input).

// En er moet een optrekweerstand (pullup) worden

// geconfigureerd om te zorgen dat

// er een 5 volt signaal (HIGH) op de pen staat als de

// knop NIET wordt ingedrukt.

// Als er op de knop wordt gedrukt komt er 0 volt (LOW)

// op.

pinMode(KnopRechts, INPUT\_PULLUP); // Idem voor de rechter knop

pinMode(KnopKlein, INPUT\_PULLUP); // Idem voor de kleine knop Links onder

pinMode(LEDGroenLinks, OUTPUT); // De pen waaraan de linker groene LED zit is een

// uitvoer (output).

pinMode(LEDGroenRechts, OUTPUT); // Idem voor de rechter groene LED.

pinMode(LEDRood, OUTPUT); // Idem voor de rode LED.

Serial.begin(115200); // Deze opdracht is nodig om straks te kunnen zien op

// de Raspberry Pi wat de Arduino aan het doen is.

// 115200 wil zeggen dat de communicatie zal gaan met

// 115200 bit/seconde.

// Hoe je op de Raspberry Pi kunt zien wat de Arduino

// doet komt later.

} // Dit is de sluitaccoulade die bij setup hoort.

// Dit sluit de rij opdrachten af die bij

// het starten van de Arduino worden uitgevoerd.

// ==========================================================================================

// Wat tussen de accoulades staat wordt uitgevoerd na setup()

void loop()

{

digitalWrite(LEDGroenLinks, HIGH); // Laat de linker LED branden

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenLinks, LOW); // Doe de linker LED weer uit

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, HIGH); // Laat de rode LED branden

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, LOW); // Doe de rode LED weer uit

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenRechts, HIGH); // Laat de rechter LED branden

delay(500); // Wacht 500 milliseconde (ms) = 0,5 seconde

digitalWrite(LEDGroenRechts, LOW); // Doe de rechter LED weer uit

delay(200); // Wacht 200 milliseconde (ms) = 0,2 seconde

VierCijfers.setBrightness(15); // Zet de helderheid op 15. 8 = zwak,

// 15 = fel

VierCijfers.showNumberDec (1234, false, 4, 0); // Laat het getal 1234 zien op 4 plekken

// met het eerste cijfer

// helemaal Links (0)

delay(1000);

VierCijfers.showNumberDec (56, false, 2, 0); // Laat het getal 56 zien met 2 cijfers,

// eerste cijfer helemaal Links

delay(1000);

VierCijfers.showNumberDec (78, false, 2, 2); // Laat het getal 78 zien met 2 cijfers,

// start vanaf het derde cijfer (plek 2,

// want het eerste cijfer is plek 0)

delay(1000);

const uint8\_t TweeCijfersLeeg[] = {0, 0}; // declareer een rij van 2 cijfers en

// maak ze allemaal 0

VierCijfers.setSegments(TweeCijfersLeeg,2,0); // Wis de linker 2 cijfers

VierCijfers.setSegments(TweeCijfersLeeg,2,2); // Wis de rechter 2 cijfers

}

### Toelichting

Met #include zorgen we dat de referenties naar de bibliotheek opgelost kunnen worden. Dus dat het object TM1637Display kan worden gemaakt met de bijbehorende methoden.

De regel TM1637Display VierCijfers (CLK, DIO); maakt het object VierCijfers van het type TM1637Display.

De regel const uint8\_t TweeCijfersLeeg[] = {0, 0}; definieert een array met twee elementen van een byte (uint8\_t is een unsigned integer van 8 bits, dus 1 byte). De bits in de getallen geven aan welke segmenten van het 7-segment display aan moeten. Alles is 0, dus geen enkel segment moet dan oplichten.

De regel VierCijfers.setSegments(TweeCijfersLeeg,2,0); zet de segmenten die moeten oplichten. Er wordt een array van 2 cijfers meegegeven en dan specificeren we dat er 2 cijfers moeten worden veranderd startend vanaf plek 0, het meest linkse cijfer.

We hadden ook een array van 4 nullen kunnen maken en dan alle 4 cijfers met 1 opdracht kunnen wissen, echter we hebben straks nodig dat we het per 2 cijfers kunnen doen en dan hebben ze dit al gezien. Daarom doen we het hier per 2 cijfers.

## De knoppen en variabelen

Dit is het aangepaste programma, met in geel de aangepaste regels. Het is alleen het deel van loop(). De rest is ongewijzigd.

// ==========================================================================================

// Wat tussen de accoulades staat wordt uitgevoerd na setup()

void loop()

{

int StatusKnopKleinNu;

digitalWrite(LEDGroenLinks, HIGH); // Laat de linker LED branden

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenLinks, LOW); // Doe de linker LED weer uit

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, HIGH); // Laat de rode LED branden

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, LOW); // Doe de rode LED weer uit

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenRechts, HIGH); // Laat de rechter LED branden

delay(500); // Wacht 500 milliseconde (ms) = 0,5 seconde

digitalWrite(LEDGroenRechts, LOW); // Doe de rechter LED weer uit

delay(200); // Wacht 200 milliseconde (ms) = 0,2 seconde

VierCijfers.setBrightness(15); // Zet de helderheid op 15. 8 = zwak,

// 15 = fel

VierCijfers.showNumberDec (1234, false, 4, 0); // Laat het getal 1234 zien op 4 plekken

// met het eerste cijfer

// helemaal Links (0)

delay(1000);

VierCijfers.showNumberDec (56, false, 2, 0); // Laat het getal 56 zien met 2 cijfers,

// eerste cijfer helemaal Links

delay(1000);

VierCijfers.showNumberDec (78, false, 2, 2); // Laat het getal 78 zien met 2 cijfers,

// start vanaf het derde cijfer (plek 2,

// want het eerste cijfer is plek 0)

delay(1000);

const uint8\_t TweeCijfersLeeg[] = {0, 0}; // declareer een rij van 2 cijfers en

// maak ze allemaal 0

VierCijfers.setSegments(TweeCijfersLeeg,2,0); // Wis de linker 2 cijfers

VierCijfers.setSegments(TweeCijfersLeeg,2,2); // Wis de rechter 2 cijfers

StatusKnopKleinNu = digitalRead(KnopKlein);

if (StatusKnopKleinNu == LOW) {

digitalWrite(LEDNano,HIGH);

} else {

digitalWrite(LEDNano,LOW);

}

}

### De operatoren voor vergelijking

Bedenk dat je ==, && en || moet gebruiken in een test. Als je bijvoorbeeld zou schrijven   
if (a = 0), dan wordt a 0 gemaakt. Een & is een bitwise AND en | is een bitwise OR. In vergelijkingen daarom altijd de dubbele tekens gebruiken.

### De compactere vorm

Je kunt hetzelfde bereiken met een enkele opdracht

digitalWrite(LEDNano,!digitalRead(KnopKlein));

De opdracht digitalRead geeft LOW of HIGH. De opdracht digitalWrite gebruikt dezelfde waarden. Echter als de knop ingedrukt is, is de waarde LOW, terwijl we dan de LED de waarde HIGH willen geven.

Omdat LOW en HIGH eigenlijk 0 en 1 zijn, net zoals true en false, kunnen we de NOT operator gebruiken. Dat is in C een uitroepteken. Daarmee keren we de waarden om. 0 wordt 1 en 1 wordt 0.

## Eerst voor de linker speler

Dit is wat de code in loop() nu is, met in geel de toevoegingen.

// ==========================================================================================

// Wat tussen de accoulades staat wordt uitgevoerd na setup()

void loop()

{

int StatusKnopKleinNu;

int StatusKnopLinksNu; // Gaat de huidige status van de knop bevatten

int StatusKnopLinksLaatst; // Om de status van de knop te bewaren toen we

// de vorige keer door de lus kwamen.

int PuntenLinks; // De punten van de linker speler

unsigned long StartTijd; // De tijd (in milli seconde vanaf de

// de start van de Arduino kan heel

// groot worden. Dat past niet in een

// int. Daar is de maximum waarde 32767.

// Daarom nemen we een long.

// En het getal zal nooit negatief worden,

// daarom nemen we unsigned (zonder +/- teken,

// dus alleen maar positieve getallen.

unsigned long SpeelTijd = 15; // de speeltijd is 15 seconde

digitalWrite(LEDGroenLinks, HIGH); // Laat de linker LED branden

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenLinks, LOW); // Doe de linker LED weer uit

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, HIGH); // Laat de rode LED branden

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, LOW); // Doe de rode LED weer uit

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenRechts, HIGH); // Laat de rechter LED branden

delay(500); // Wacht 500 milliseconde (ms) = 0,5 seconde

digitalWrite(LEDGroenRechts, LOW); // Doe de rechter LED weer uit

delay(200); // Wacht 200 milliseconde (ms) = 0,2 seconde

VierCijfers.setBrightness(15); // Zet de helderheid op 15. 8 = zwak,

// 15 = fel

VierCijfers.showNumberDec (1234, false, 4, 0); // Laat het getal 1234 zien op 4 plekken

// met het eerste cijfer

// helemaal Links (0)

delay(1000);

VierCijfers.showNumberDec (56, false, 2, 0); // Laat het getal 56 zien met 2 cijfers,

// eerste cijfer helemaal Links

delay(1000);

VierCijfers.showNumberDec (78, false, 2, 2); // Laat het getal 78 zien met 2 cijfers,

// start vanaf het derde cijfer (plek 2,

// want het eerste cijfer is plek 0)

delay(1000);

const uint8\_t TweeCijfersLeeg[] = {0, 0}; // declareer een rij van 2 cijfers en

// maak ze allemaal 0

VierCijfers.setSegments(TweeCijfersLeeg,2,0); // Wis de linker 2 cijfers

VierCijfers.setSegments(TweeCijfersLeeg,2,2); // Wis de rechter 2 cijfers

StatusKnopKleinNu = digitalRead(KnopKlein);

if (StatusKnopKleinNu == LOW) {

digitalWrite(LEDNano,HIGH);

} else {

digitalWrite(LEDNano,LOW);

}

while (true) { // Herhaal dit eindeloos

if (digitalRead(KnopKlein) == LOW) { // Als het kleine knopje ingedrukt wordt

// dan gaan we van start

// ==== Start het spel

StartTijd = millis(); // Onthoud hoe laat het nu is

PuntenLinks = 0; // Zet de punten van de linker speler op

// nul

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

// Vertoon het aantal punten van de linker

// speler in de twee linker cijfers

StatusKnopLinksNu = digitalRead(KnopLinks); // Lees de status van de linker knop

StatusKnopLinksLaatst = StatusKnopLinksNu; // En dat is dan ook de beginwaarde

while ((millis() < StartTijd + SpeelTijd \* 1000) && (PuntenLinks < 99) ) {

// Als de huidige tijd (millis()) kleiner

// is dan de starttijd plus de speeltijd

// (die we met 1000 vermenigvuldigen om

// ook milliseconde te krijgen), dan

// kunnen we doorgaan: de speeltijd is

// nog niet om.

// Maar als de speler al 99 punten heeft

// dan moeten we toch stoppen, want

// meer dan 99 punten kunnen we niet

// laten zien.

StatusKnopLinksNu = digitalRead(KnopLinks);

if (StatusKnopLinksNu == LOW && StatusKnopLinksLaatst == HIGH) {

// Als de knop nu LOW is en hij was HIGH

// in de vorige keer dat we door de lus

// kwamen, dan is de knop nu net ingedrukt

PuntenLinks = PuntenLinks + 1; // De linker speler krijgt er een punt bij

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

// Vertoon het aantal punten van de linker

// speler in de twee linker cijfers

}

StatusKnopLinksLaatst = StatusKnopLinksNu; // Onthoud de waarde voor de volgende keer

// dat we door de lus komen

}

}

)

## Nu voor de rechter speler

Hieronder staat het programma met de toevoegingen voor de rechter speler in geel.

// ==========================================================================================

// Wat tussen de accoulades staat wordt uitgevoerd na setup()

void loop()

{

int StatusKnopKleinNu;

int StatusKnopLinksNu; // Gaat de huidige status van de knop bevatten

int StatusKnopLinksLaatst; // Om de status van de knop te bewaren toen we

// de vorige keer door de lus kwamen.

int PuntenLinks; // De punten van de linker speler

int StatusKnopRechtsNu; // Gaat de huidige status van de knop bevatten

int StatusKnopRechtsLaatst; // Om de status van de knop te bewaren toen we

// de vorige keer door de lus kwamen.

int PuntenRechts; // De punten van de linker speler

unsigned long StartTijd; // De tijd (in milli seconde vanaf de

// de start van de Arduino kan heel

// groot worden. Dat past niet in een

// int. Daar is de maximum waarde 32767.

// Daarom nemen we een long.

// En het getal zal nooit negatief worden,

// daarom nemen we unsigned (zonder +/- teken,

// dus alleen maar positieve getallen.

unsigned long SpeelTijd = 15; // de speeltijd is 15 seconde

digitalWrite(LEDGroenLinks, HIGH); // Laat de linker LED branden

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenLinks, LOW); // Doe de linker LED weer uit

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, HIGH); // Laat de rode LED branden

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, LOW); // Doe de rode LED weer uit

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenRechts, HIGH); // Laat de rechter LED branden

delay(500); // Wacht 500 milliseconde (ms) = 0,5 seconde

digitalWrite(LEDGroenRechts, LOW); // Doe de rechter LED weer uit

delay(200); // Wacht 200 milliseconde (ms) = 0,2 seconde

VierCijfers.setBrightness(15); // Zet de helderheid op 15. 8 = zwak,

// 15 = fel

VierCijfers.showNumberDec (1234, false, 4, 0); // Laat het getal 1234 zien op 4 plekken

// met het eerste cijfer

// helemaal Links (0)

delay(1000);

VierCijfers.showNumberDec (56, false, 2, 0); // Laat het getal 56 zien met 2 cijfers,

// eerste cijfer helemaal Links

delay(1000);

VierCijfers.showNumberDec (78, false, 2, 2); // Laat het getal 78 zien met 2 cijfers,

// start vanaf het derde cijfer (plek 2,

// want het eerste cijfer is plek 0)

delay(1000);

const uint8\_t TweeCijfersLeeg[] = {0, 0}; // declareer een rij van 2 cijfers en

// maak ze allemaal 0

VierCijfers.setSegments(TweeCijfersLeeg,2,0); // Wis de linker 2 cijfers

VierCijfers.setSegments(TweeCijfersLeeg,2,2); // Wis de rechter 2 cijfers

StatusKnopKleinNu = digitalRead(KnopKlein);

if (StatusKnopKleinNu == LOW) {

digitalWrite(LEDNano,HIGH);

} else {

digitalWrite(LEDNano,LOW);

}

while (true) { // Herhaal dit eindeloos

if (digitalRead(KnopKlein) == LOW) { // Als het kleine knopje ingedrukt wordt

// dan gaan we van start

// ==== Start het spel

StartTijd = millis(); // Onthoud hoe laat het nu is

PuntenLinks = 0; // Zet de punten van de linker speler op

// nul

PuntenRechts = 0; // Zet de punten van de rechter speler op

// nul

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

// Vertoon het aantal punten van de linker

// speler in de twee linker cijfers

VierCijfers.showNumberDec(PuntenRechts, false, 2, 2);

// Vertoon het aantal punten van de

// rechter speler in de twee

// rechter cijfers

StatusKnopLinksNu = digitalRead(KnopLinks); // Lees de status van de linker knop

StatusKnopLinksLaatst = StatusKnopLinksNu; // En dat is dan ook de beginwaarde

StatusKnopRechtsNu = digitalRead(KnopRechts); // Lees de status van de linker knop

StatusKnopRechtsLaatst = StatusKnopRechtsNu; // En dat is dan ook de beginwaarde

while ((millis() < StartTijd + SpeelTijd \* 1000) && (PuntenLinks < 99) && (PuntenRechts < 99)) {

// Als de huidige tijd (millis()) kleiner

// is dan de starttijd plus de speeltijd

// (die we met 1000 vermenigvuldigen om

// ook milliseconde te krijgen), dan

// kunnen we doorgaan: de speeltijd is

// nog niet om.

// Maar als de speler al 99 punten heeft

// dan moeten we toch stoppen, want

// meer dan 99 punten kunnen we niet

// laten zien.

StatusKnopLinksNu = digitalRead(KnopLinks);

if (StatusKnopLinksNu == LOW && StatusKnopLinksLaatst == HIGH) {

// Als de knop nu LOW is en hij was HIGH

// in de vorige keer dat we door de lus

// kwamen, dan is de knop nu net ingedrukt

PuntenLinks = PuntenLinks + 1; // De linker speler krijgt er een punt bij

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

// Vertoon het aantal punten van de linker

// speler in de twee linker cijfers

}

StatusKnopLinksLaatst = StatusKnopLinksNu; // Onthoud de waarde voor de volgende keer

// dat we door de lus komen

StatusKnopRechtsNu = digitalRead(KnopRechts);

if (StatusKnopRechtsNu == LOW && StatusKnopRechtsLaatst == HIGH) {

// Als de knop nu LOW is en hij was HIGH

// in de vorige keer dat we door de lus

// kwamen, dan is de knop nu net ingedrukt

PuntenRechts = PuntenRechts + 1; // De rechter speler krijgt er een punt

// bij

VierCijfers.showNumberDec(PuntenRechts, false, 2, 2);

// Vertoon het aantal punten van de

// rechter speler in de twee rechter

// cijfers

}

StatusKnopRechtsLaatst = StatusKnopRechtsNu;// Onthoud de waarde voor de volgende keer

// dat we door de lus komen

}

}

}

}

## Wie heeft er gewonnen?

// ==========================================================================================

// Wat tussen de accoulades staat wordt uitgevoerd na setup()

void loop()

{

int StatusKnopKleinNu;

int StatusKnopLinksNu; // Gaat de huidige status van de knop bevatten

int StatusKnopLinksLaatst; // Om de status van de knop te bewaren toen we

// de vorige keer door de lus kwamen.

int PuntenLinks; // De punten van de linker speler

int StatusKnopRechtsNu; // Gaat de huidige status van de knop bevatten

int StatusKnopRechtsLaatst; // Om de status van de knop te bewaren toen we

// de vorige keer door de lus kwamen.

int PuntenRechts; // De punten van de linker speler

unsigned long StartTijd; // De tijd (in milli seconde vanaf de

// de start van de Arduino kan heel

// groot worden. Dat past niet in een

// int. Daar is de maximum waarde 32767.

// Daarom nemen we een long.

// En het getal zal nooit negatief worden,

// daarom nemen we unsigned (zonder +/- teken,

// dus alleen maar positieve getallen.

unsigned long SpeelTijd = 15; // de speeltijd is 15 seconde

digitalWrite(LEDGroenLinks, HIGH); // Laat de linker LED branden

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenLinks, LOW); // Doe de linker LED weer uit

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, HIGH); // Laat de rode LED branden

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, LOW); // Doe de rode LED weer uit

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenRechts, HIGH); // Laat de rechter LED branden

delay(500); // Wacht 500 milliseconde (ms) = 0,5 seconde

digitalWrite(LEDGroenRechts, LOW); // Doe de rechter LED weer uit

delay(200); // Wacht 200 milliseconde (ms) = 0,2 seconde

VierCijfers.setBrightness(15); // Zet de helderheid op 15. 8 = zwak,

// 15 = fel

VierCijfers.showNumberDec (1234, false, 4, 0); // Laat het getal 1234 zien op 4 plekken

// met het eerste cijfer

// helemaal Links (0)

delay(1000);

VierCijfers.showNumberDec (56, false, 2, 0); // Laat het getal 56 zien met 2 cijfers,

// eerste cijfer helemaal Links

delay(1000);

VierCijfers.showNumberDec (78, false, 2, 2); // Laat het getal 78 zien met 2 cijfers,

// start vanaf het derde cijfer (plek 2,

// want het eerste cijfer is plek 0)

delay(1000);

const uint8\_t TweeCijfersLeeg[] = {0, 0}; // declareer een rij van 2 cijfers en

// maak ze allemaal 0

VierCijfers.setSegments(TweeCijfersLeeg,2,0); // Wis de linker 2 cijfers

VierCijfers.setSegments(TweeCijfersLeeg,2,2); // Wis de rechter 2 cijfers

StatusKnopKleinNu = digitalRead(KnopKlein);

if (StatusKnopKleinNu == LOW) {

digitalWrite(LEDNano,HIGH);

} else {

digitalWrite(LEDNano,LOW);

}

while (true) { // Herhaal dit eindeloos

if (digitalRead(KnopKlein) == LOW) { // Als het kleine knopje ingedrukt wordt

// dan gaan we van start

// ==== Start het spel

StartTijd = millis(); // Onthoud hoe laat het nu is

PuntenLinks = 0; // Zet de punten van de linker speler op

// nul

PuntenRechts = 0; // Zet de punten van de rechter speler op

// nul

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

// Vertoon het aantal punten van de linker

// speler in de twee linker cijfers

VierCijfers.showNumberDec(PuntenRechts, false, 2, 2);

// Vertoon het aantal punten van de

// rechter speler in de twee

// rechter cijfers

StatusKnopLinksNu = digitalRead(KnopLinks); // Lees de status van de linker knop

StatusKnopLinksLaatst = StatusKnopLinksNu; // En dat is dan ook de beginwaarde

StatusKnopRechtsNu = digitalRead(KnopRechts); // Lees de status van de linker knop

StatusKnopRechtsLaatst = StatusKnopRechtsNu; // En dat is dan ook de beginwaarde

while ((millis() < StartTijd + SpeelTijd \* 1000) && (PuntenLinks < 99) && (PuntenRechts < 99)) {

// Als de huidige tijd (millis()) kleiner

// is dan de starttijd plus de speeltijd

// (die we met 1000 vermenigvuldigen om

// ook milliseconde te krijgen), dan

// kunnen we doorgaan: de speeltijd is

// nog niet om.

// Maar als de speler al 99 punten heeft

// dan moeten we toch stoppen, want

// meer dan 99 punten kunnen we niet

// laten zien.

StatusKnopLinksNu = digitalRead(KnopLinks);

if (StatusKnopLinksNu == LOW && StatusKnopLinksLaatst == HIGH) {

// Als de knop nu LOW is en hij was HIGH

// in de vorige keer dat we door de lus

// kwamen, dan is de knop nu net ingedrukt

PuntenLinks = PuntenLinks + 1; // De linker speler krijgt er een punt bij

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

// Vertoon het aantal punten van de linker

// speler in de twee linker cijfers

}

StatusKnopLinksLaatst = StatusKnopLinksNu; // Onthoud de waarde voor de volgende keer

// dat we door de lus komen

StatusKnopRechtsNu = digitalRead(KnopRechts);

if (StatusKnopRechtsNu == LOW && StatusKnopRechtsLaatst == HIGH) {

// Als de knop nu LOW is en hij was HIGH

// in de vorige keer dat we door de lus

// kwamen, dan is de knop nu net ingedrukt

PuntenRechts = PuntenRechts + 1; // De rechter speler krijgt er een punt

// bij

VierCijfers.showNumberDec(PuntenRechts, false, 2, 2);

// Vertoon het aantal punten van de

// rechter speler in de twee rechter

// cijfers

}

StatusKnopRechtsLaatst = StatusKnopRechtsNu;// Onthoud de waarde voor de volgende keer

// dat we door de lus komen

}

// Deze ronde is afgelopen. Laat nu de punten van de winnaar, of van beide bij gelijk

// spel knipperen. Dat doen we door eerst de cijfers van de winnaar (of beiden) weg te

// halen en dan beide weer te laten zien.

while (digitalRead(KnopKlein) == HIGH) {

if (PuntenLinks >= PuntenRechts) { // Als Links heeft gewonnen of als

// het gelijk is ...

VierCijfers.setSegments(TweeCijfersLeeg,2,0); // Wis de linker 2 cijfers

}

if (PuntenRechts >= PuntenLinks) {

VierCijfers.setSegments(TweeCijfersLeeg,2,2); // Wis de rechter 2 cijfers

}

delay (100);

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

VierCijfers.showNumberDec(PuntenRechts, false, 2, 2);

delay (100);

}

}

}

}

## Speeltijd instellen

// ==========================================================================================

// Wat tussen de accoulades staat wordt uitgevoerd na setup()

void loop()

{

int StatusKnopKleinNu;

int StatusKnopLinksNu; // Gaat de huidige status van de knop bevatten

int StatusKnopLinksLaatst; // Om de status van de knop te bewaren toen we

// de vorige keer door de lus kwamen.

int PuntenLinks; // De punten van de linker speler

int StatusKnopRechtsNu; // Gaat de huidige status van de knop bevatten

int StatusKnopRechtsLaatst; // Om de status van de knop te bewaren toen we

// de vorige keer door de lus kwamen.

int PuntenRechts; // De punten van de linker speler

unsigned long StartTijd; // De tijd (in milli seconde vanaf de

// de start van de Arduino kan heel

// groot worden. Dat past niet in een

// int. Daar is de maximum waarde 32767.

// Daarom nemen we een long.

// En het getal zal nooit negatief worden,

// daarom nemen we unsigned (zonder +/- teken,

// dus alleen maar positieve getallen.

unsigned long SpeelTijd = 15; // de speeltijd is 15 seconde

digitalWrite(LEDGroenLinks, HIGH); // Laat de linker LED branden

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenLinks, LOW); // Doe de linker LED weer uit

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, HIGH); // Laat de rode LED branden

delay(500); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDRood, LOW); // Doe de rode LED weer uit

delay(100); // Wacht 1000 milliseconde (ms) = 1 seconde

digitalWrite(LEDGroenRechts, HIGH); // Laat de rechter LED branden

delay(500); // Wacht 500 milliseconde (ms) = 0,5 seconde

digitalWrite(LEDGroenRechts, LOW); // Doe de rechter LED weer uit

delay(200); // Wacht 200 milliseconde (ms) = 0,2 seconde

VierCijfers.setBrightness(15); // Zet de helderheid op 15. 8 = zwak,

// 15 = fel

VierCijfers.showNumberDec (1234, false, 4, 0); // Laat het getal 1234 zien op 4 plekken

// met het eerste cijfer

// helemaal Links (0)

delay(1000);

VierCijfers.showNumberDec (56, false, 2, 0); // Laat het getal 56 zien met 2 cijfers,

// eerste cijfer helemaal Links

delay(1000);

VierCijfers.showNumberDec (78, false, 2, 2); // Laat het getal 78 zien met 2 cijfers,

// start vanaf het derde cijfer (plek 2,

// want het eerste cijfer is plek 0)

delay(1000);

const uint8\_t TweeCijfersLeeg[] = {0, 0}; // declareer een rij van 2 cijfers en

// maak ze allemaal 0

VierCijfers.setSegments(TweeCijfersLeeg,2,0); // Wis de linker 2 cijfers

VierCijfers.setSegments(TweeCijfersLeeg,2,2); // Wis de rechter 2 cijfers

StatusKnopKleinNu = digitalRead(KnopKlein);

if (StatusKnopKleinNu == LOW) {

digitalWrite(LEDNano,HIGH);

} else {

digitalWrite(LEDNano,LOW);

}

while (true) { // Herhaal dit eindeloos

if (digitalRead(KnopKlein) == LOW) { // Als het kleine knopje ingedrukt wordt

// dan gaan we van start

// ==== Stel nu eerst de speeltijd in

VierCijfers.setSegments(TweeCijfersLeeg,2,0); // Wis de linker 2 cijfers

VierCijfers.setSegments(TweeCijfersLeeg,2,2); // Wis de rechter 2 cijfers

int i;

int PotMeterStand;

for (i=1; i<1000; i++) { // doe het 1000 keer om de spelers

// een beetje tijd te geven

PotMeterStand = analogRead(PotMeter); // lees de stand van de potmeter

SpeelTijd = map(PotMeterStand,0,1024,5,60); // vertaal het naar een brukkbare

// waarde voor de speeltijd

VierCijfers.showNumberDec(SpeelTijd, false,2, 1); // laat de speeltijd zien

}

// ==== Start het spel

StartTijd = millis(); // Onthoud hoe laat het nu is

PuntenLinks = 0; // Zet de punten van de linker speler op

// nul

PuntenRechts = 0; // Zet de punten van de rechter speler op

// nul

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

// Vertoon het aantal punten van de linker

// speler in de twee linker cijfers

VierCijfers.showNumberDec(PuntenRechts, false, 2, 2);

// Vertoon het aantal punten van de

// rechter speler in de twee

// rechter cijfers

StatusKnopLinksNu = digitalRead(KnopLinks); // Lees de status van de linker knop

StatusKnopLinksLaatst = StatusKnopLinksNu; // En dat is dan ook de beginwaarde

StatusKnopRechtsNu = digitalRead(KnopRechts); // Lees de status van de linker knop

StatusKnopRechtsLaatst = StatusKnopRechtsNu; // En dat is dan ook de beginwaarde

while ((millis() < StartTijd + SpeelTijd \* 1000) && (PuntenLinks < 99) && (PuntenRechts < 99)) {

// Als de huidige tijd (millis()) kleiner

// is dan de starttijd plus de speeltijd

// (die we met 1000 vermenigvuldigen om

// ook milliseconde te krijgen), dan

// kunnen we doorgaan: de speeltijd is

// nog niet om.

// Maar als de speler al 99 punten heeft

// dan moeten we toch stoppen, want

// meer dan 99 punten kunnen we niet

// laten zien.

StatusKnopLinksNu = digitalRead(KnopLinks);

if (StatusKnopLinksNu == LOW && StatusKnopLinksLaatst == HIGH) {

// Als de knop nu LOW is en hij was HIGH

// in de vorige keer dat we door de lus

// kwamen, dan is de knop nu net ingedrukt

PuntenLinks = PuntenLinks + 1; // De linker speler krijgt er een punt bij

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

// Vertoon het aantal punten van de linker

// speler in de twee linker cijfers

}

StatusKnopLinksLaatst = StatusKnopLinksNu; // Onthoud de waarde voor de volgende keer

// dat we door de lus komen

StatusKnopRechtsNu = digitalRead(KnopRechts);

if (StatusKnopRechtsNu == LOW && StatusKnopRechtsLaatst == HIGH) {

// Als de knop nu LOW is en hij was HIGH

// in de vorige keer dat we door de lus

// kwamen, dan is de knop nu net ingedrukt

PuntenRechts = PuntenRechts + 1; // De rechter speler krijgt er een punt

// bij

VierCijfers.showNumberDec(PuntenRechts, false, 2, 2);

// Vertoon het aantal punten van de

// rechter speler in de twee

// rechter cijfers

}

StatusKnopRechtsLaatst = StatusKnopRechtsNu;// Onthoud de waarde voor de volgende keer

// dat we door de lus komen

}

// Deze ronde is afgelopen. Laat nu de punten van de winnaar, of van beide bij gelijk

// spel knipperen. Dat doen we door eerst de cijfers van de winnaar (of beiden) weg te

// halen en dan beide weer te laten zien.

while (digitalRead(KnopKlein) == HIGH) {

if (PuntenLinks >= PuntenRechts) { // Als Links heeft gewonnen of als

// het gelijk is ...

VierCijfers.setSegments(TweeCijfersLeeg,2,0); // Wis de linker 2 cijfers

}

if (PuntenRechts >= PuntenLinks) {

VierCijfers.setSegments(TweeCijfersLeeg,2,2); // Wis de rechter 2 cijfers

}

delay (100);

VierCijfers.showNumberDec(PuntenLinks, false, 2, 0);

VierCijfers.showNumberDec(PuntenRechts, false, 2, 2);

delay (100);

}

}

}

}

### Werkt de potmeter logisch?

De range van 0 tot 1024 moet vertaald worden naar een range van 60 tot 5 i.p.v. 5 tot 60. Dit is de aangepaste regel: draai de 60 en de 5 om.

SpeelTijd = map(PotMeterStand,0,1024,60,5);