Les -7- **Strings en strings** 16 april 2019

Een **s**tring is een array van karakters. een **S**tring is een hoeveelheid tekst, een object.
Een string (kleine letter -s-) staat altijd tussen (…) haakjes.
Met **Serial**.print() en **Serial**.println() werk je al met strings.

We kunnen als eerste beginnen met het "**string**" (kleine letter -s-) datatype.
Hierbij is een string een zogenaamde "**array**" van karakters (char)

De tweede manier om met strings te gaan werken is met het "**String**" object (hoofdletter -S-).
Dat komt verderop deze les aan de orde.

De variabele is de **geheugenplaats** waar het eerste karakter staat, dus niet het karakter zelf!
Een string eindigt met een zgn null-karakter, waarmee de laatste geheugenplek is aangegeven.
Het null karakter is daadwerkelijk 'niets'. Wordt aangegeven met Hex 00 of **esc \0**
De array, zoals elk array, begint bij nul te tellen, het eerste karakter in de lijst heeft nummer 0.
Dit wordt zero indexed genoemd.
Een string declareer je als volgt: `char Naam[] = "Hans";` let op de **[ ]** grote blokhaken!
Met de [ ] geeft je de compiler opdracht om zelf de stringlengte te bepalen.
Dat zijn hier dus 5 karakters, 4 + null. Op deze manier is de string: "naam" 5 karakters lang.
Deze lengte is hiermee vastgelegd, heb je later een langere naam, dan kom je in de problemen.
Door niets aan te geven, wordt de lengte éénmalig bij de declaratie bepaald.

Je kan vooraf de lengte bepalen met: `char Naam[5] = "Hans";` hier dus 5 plaatsen.
Door `char Naam[10] = "Hans";` kan je ook langere namen invoeren tot max 9 karakters!

Elk karakter heeft een eigen plek H pos 0
a pos 1
n pos 2
s pos 3
Null pos 4

```
char Naam[] = "Hans";
Serial.println(Naam[2]); // print "n"
Serial.println(Naam[0]); // print "H"
Serial.println(Naam);    // print "Hans"
```

Waarden toekennen aan een string

Als de naam Hans in Bram moet veranderen, werkt de opdracht: Naam = "Bram" niet.
De string geeft niet de string zelf, maar de geheugen locatie van het eerste karakter aan!
Je kan de karakters één voor één wijzigen, maar dat kan sneller, met: **cpystr ()**
Dit staat voor **co**py **str**ing, waaraan je **twee** argumenten moet meegeven.
Het eerste argument is de BESTEMMING, het tweede argument de BRON.
Die bestemming moet natuurlijk wel een passende variabele zijn van het juiste datatype.
De bron mag een variabele, constante of gewoon tekst tussen "aanhalingstekens" zijn.
Een voorbeeld:

```
void setup() {
  Serial.begin(115200);
  char Naam[] = "Hans";
  Serial.println(Naam[2]); // print "n"
  Serial.println(Naam[0]); // print "H"
  Serial.println(Naam);    // print "Hans"
  strcpy(Naam, "Bram");  // dit is de kopiëer opdracht waarin Hans in Bram veranderd.
  Serial.println(Naam[2]); // print "a"
  Serial.println(Naam[0]); // print "B"
  Serial.println(Naam);    // print "Bram"
}
```

Speciale tekens (escape karakters) kunnen ook.
Veronderstel dat we " in de tekst willen gebruiken, dat kan niet, omdat dit het begin of einde van een string aangeeft. Fouten zijn herkenbaar aan tekstverkleuringen.

```
Serial.println("Hallo "gast", welkom bij Arduino");
```

Niet erg duidelijk, gast is zwart en de rest van de tekst is blauw. Je krijgt een foutmelding.
De juiste manier is: `Serial.println("Hallo \"gast\", welkom bij Arduino"); // met backslash \`
De meest gebruikte codes zijn:

Esc code	ASCII	Doel
\n	10	LF (Line Feed) – Start een nieuwe regel.
\r	13	CR (Carriage Return) – Ga naar het begin van een regel
\t	9	Tab – Ga naar de volgende tab-stop (tab)
\”	34	Dubbele aanhalingstekens
\’	39	Enkele aanhalingstekens
\\	92	Backslash – zodat we die dus ook kunnen typen

Een lege string ("") of een string met een enkel karakter ("A") kan ook.
Een char is een karakter, GEEN string, karakter is een heel ander data type.

De opdracht: Naam[1] = "H"; zal dus een foutmelding geven.
Om een enkel karakter in een string zetten, moet je het tussen enkele 'aanhalingstekens' zetten.

➔De juiste manier is: Name[1] = 'H'; ➔
Enkele karakters tussen enkele aanhalingstekens, meerdere karakters tussen dubbele aanhalingstekens. Een voorbeeld met Hans en Bram.

```
void setup() {
  Serial.begin(115200);
  char Naam[] = "Hans";  // hier meerdere karakters, dus dubbele aanhalingstekens.
  Serial.println(Naam);    // print "Hans"
  Naam[0] = 'B';           // hier worden stuk voor stuk de karakters vervangen.
  Naam[1] = 'r';           // dit in tegenstelling met de eerdere opdracht strcpy();
  Naam[2] = 'a';           // telkens een enkel karakter, dus enkele aanhalingstekens.
  Naam[3] = 'm';
  Serial.println(Naam);    // print "Bram"
}
```

Wanneer de nieuwe string korter is dan de oude, moet je WEL het null karakter meesturen.

```
Naam[0] = 'M';
Naam[1] = 'a';
Naam[2] = 'x';
Naam[3] = 0;            // het null karakter, geen aanhalingstekens en gewoon een nul!
```

Een langere naam heb ik uitgeprobeerd, werkt wel, maar als laatste altijd het null karakter!!
Je kan en mag ook de ASCII codes invoeren, maar nooit boven de 255! Zie ASCII tabel.

```
Naam[0] = 66;  // = 'B'
Naam[1] = 114; // = 'r'
Naam[2] = 97;  // = 'a'
Naam[3] = 109; // = 'm'
Naam[4] = 0;
```

Omdat karakters ook nummers zijn, kan je er ook mee tellen.
Het karakter moet dan wel tussen 'enkele' aanhalingstekens staan!

```
void setup() {
 Serial.begin(115200);
 char Letter[] = " ";                  // een array (string) van één krakter
 for(char A='A'; A<='Z'; A++) {  // een for lus, van 65 (A) tot 90 (Z)  let op, enkele '  '!
  Letter[0] = A;                       // hier wordt de string gevuld met de ASCII waarde
  Serial.println(Letter[0]);          // uiteindelijk wordt de letter geprint
 }                                        // zo krijg je het hele alfabet onder elkaar
}
void loop() {
 // leave empty for now
}
```

Andere Array Functies: strlen, sizeof, strcat

Naast de functie “strcpy()” zijn er nog meer van die functies

strlen() string Length heeft maar 1 argument nodig, de string (array) en geeft terug hoe veel tekens er in een string staan. Het NULL karakter wordt daarbij **NIET** meegeteld.

sizeof() doet hetzelfde als “strlen()”, maar deze neemt **WEL** het NULL karakter(s) mee in de telling. Ook als er niets staat, (dit zijn nullen) worden de lege plekken meegeteld.

strcat() String Concatenate (aaneenschakelen) plakt twee array’s (strings) aan elkaar.

Voorbeelden voor strlen, sizeof, strcat

```
void setup() {
 Serial.begin(115200);          // de communicatie snelheid
  char Naam[45] = "Hans";      // Hans krijgt een ruimte van 45 karakters
  int  count = 0;                  // tellertje
  Serial.print("Naam = ");      // tekst
  Serial.println(Naam);          // de naam (Hans in dit geval)
  count = strlen(Naam);          // geeft de lengte van Hans, dus 4
  Serial.print("De lengte van Naam met strlen: ");  // tekst
  Serial.println(count);          // hier wordt het getal -4- weergegeven

  count = sizeof(Naam);          // alle karakter plaatsen incl NULL worden geteld
  Serial.print("De lengte van Naam met sizeof: ");  // tekst
  Serial.println(count);          // hier wordt alles weergegeven, 45 karakters

  strcat(Naam, " heeft 2 neefjes, genaamd Bram en Max!");
// hier worden twee zaken samengevoegd: Hans, en tekst. Vormen samen de string: naam.
  Serial.print("Naam = ");      // hier wordt alleen tekst geprint
  Serial.println(Naam);          // hier wordt het samenstel 'naam' geprint.
  count = strlen(Naam);          // de het aantal karakters van het samenstel wordt geteld
  Serial.print("De lengte van Naam met strlen: "); // tekst
  Serial.println(count);          // het totale aantal is 42 (spaties zijn geen nullen)
  count = sizeof(Naam);          // opnieuw tellen, maar nu inclusief null karakters.
  Serial.print("De lengte van Naam met sizeof: ");  // tekst
  Serial.println(count);       // totale string ruimte is 45 zoals in het begin is opgegeven.
}
void loop() {
   // laat even leeg
}
```

De output ziet er als volgt uit:

```
Naam = Hans
De lengte van Naam met strlen: 4
De lengte van Naam met sizeof: 45
Naam = Hans heeft 2 neefjes, genaamd Bram en Max!
De lengte van Naam met strlen: 42
De lengte van Naam met sizeof: 45
```

Als we een fout maken: char Naam[] = "Hans" door geen lengte op te geven krijg je rommel. Omdat er geen null is gaat de weergave door, dit geeft onzin en andere onverwachte zaken.

String als Object (String met hoofdletter “S”)

Een “object” is een “**ding**” of “item” met eigenschappen en methoden.
Methoden moet je zien als “functies”. Zo een “**ding**” kan van alles zijn, b.v. een “**auto**“.

De auto heeft een aantal eigenschappen (properties) zoals: kleur, motor, benzine, deuren, etc. Properties doen op zichzelf niets, maar geven eigenlijk alleen maar feiten door met betrekking tot het object “auto”.
Een property kan een waarde hebben of de waarde kan ontbreken. Zo kan de auto wel of geen “bestuurder” hebben. In dat laatste geval is “bestuurder” gelijk aan NULL.
Als “**bestuurder != NULL**” dan zit er dus een bestuurder in de auto.

Een auto kan bepaalde dingen doen: Rijden, parkeren, toeteren, open of sluit achterklep, etc. Dit zijn **functies** van het object “auto” en in “object georiënteerd programmeren” noemt men deze functies “**methods**” (methodes).

Methodes kunnen dus iets “doen”; dus acties die de auto kan uitvoeren.
Voordeel: alles staat bij elkaar – de eigenschappen en de functies hangen vast aan het object.

Een object kan zich als een soort datatype gedragen, waarbij elke variabele van dat type zich op dezelfde manier gedraagt en dezelfde soort eigenschappen (properties) heeft.
We kunnen dus een variabele “MijnAuto” van het object type “auto” definiëren.

Kleine moeite en ineens zijn alle methoden en properties beschikbaar voor “MijnAuto”.
Als we nu een garage vol met auto’s hebben dan kunnen we zelfs een array maken van dit object type of meerder variabelen definieren: MijnAuto, JouwAuto, MammasAuto, PappasAuto, of als array: Autos[0], Autos[1], Autos[2],….

De “String” (met hoofdletter “S”) en “Serial” zijn dat soort objecten.
We hebben met “Serial” al een aantal methods (functies of methoden) zoals “begin”, “print” en “println” gezien. We gebruiken deze methoden om ons object (Serial) iets te laten doen.

Een dergelijk functie roepen we altijd aan in dit formaat: OBJECT.METHODE of OBJECT.PROPERTY ….? Altijd een punt tussen object en property of methode.

Het “String” Object

Een **S**tring variabele declareer je zo: String naam = "Hans"; (char Naam[45] = "Hans";)
Geen gezeur over de lengte van de tekst – de vierkant haakjes worden hier dus **niet** gebruikt.
“String” kan je gewoon gebruiken als je “Serial.print()” aanroept.
Dus een klein voorbeeld:

```
void setup() {
  Serial.begin(115200);
  String Naam = "Hans";
  Serial.print("Naam = ");
  Serial.println(Naam);
}
```

Dit werkt hetzelfde als bij de “string” en de seriële monitor laat dat zien: Naam = Hans
Nog steeds eigenlijk weinig nieuws en ook niks spannend of ingewikkeld.
Laten we nu eens de tekst “Hans” naar “Bram” veranderen door de “String” een nieuwe “String” te geven:

```
void setup() {
  Serial.begin(115200);
  String Naam = "Hans";    // Hoofdletter -S-
  Serial.print("Naam = "); // tekst
  Serial.println(Naam);     // print  Hans
  Naam = "Bram";            // wijzigt de naam in Bram
  Serial.print("Naam = "); // tekst
  Serial.println(Naam);     // print Bram
}
```

In deze regel: **Naam = "Bram";** wordt een nieuw String object gemaakt met de tekst "Bram".
Dit kan je ook mag schrijven als: **Naam = String("Bram");**
De tweede methode wordt vaak gebruikt om b.v. een “string” om te zetten naar een “String”.
De lengte van de String is nu onbelangrijk voor de code, het object regelt dit automatisch.

```
void setup() {
  Serial.begin(115200);
  String Naam = "Hans";      // objectnaam Hans
  Serial.print("Naam = ");    // tekst: Naam =
  Serial.println(Naam);       // weergave: Naam = Hans
  Naam = "Hans heeft 2 neefjes: Bram en Max!";  // wijzigen van het object inhoud.
  Serial.print("Naam = ");    // tekst: naam =
  Serial.println(Naam);       // weergave: Naam = Hans heeft 2 neefjes: Bram en Max!
                              // Hier wordt de nieuwe inhoud van naam weergegeven.
}
void loop() {
  // leave empty for now
}
```

Met de “string”, moesten we dus goed opletten of de tekst in de lengte van de array zou passen. Met het String object is dat echter niet het geval. We krijgen geen maffe output te zien als de String langer wordt dan de oorspronkelijke lengte was.

Als dat nieuwe “String” object dan zo handig is,… waarom zouden we dan nog kijken naar die oude “string” (array of char)? Daar zijn twee mogelijke redenen voor:
-1- Als eerste neemt het object meer geheugen in beslag – je kunt het als een aanvulling zien.
-2- Het object bevat intern de “string”, dus je moet wat weten van de string met de kleine s.

String object Methoden (functies)

Nu moet je weten dat het “String” object veel meer methoden (functies) heeft, die het een en ander eenvoudiger maken.

Een voorbeeld van enkele methoden, kijk eens bij de **Arduino String Object Reference**.
https://www.arduino.cc/reference/en/language/variables/data-types/stringobject/
Dit is helaas in het Engels, maar het laat je nog meer methoden zien.

Length(), Concat() en Nummers converteren

Voorbeeld: **S**trings aan elkaar plakken, String lengte bepalen en getallen toevoegen.

```
void setup() {
 Serial.begin(115200);
 String Naam = "Hans";
 Serial.print("Naam = ");         // tekst
 Serial.println(Naam);            // Naam = Hans
 Serial.print("De lengte van Naam met strlen: ");  // tekst
 Serial.println(Naam.length());// print De lengte van Naam met strlen: 4
 Naam.concat(" heeft 2 neefjes: Bram en Max!"); // voeg tekst toe aan de string.
                                                // vergelijk dit met strcat bij string
 Serial.print("Naam = ");         // tekst
 Serial.println(Naam);            // Naam = Hans heeft 2 neefjes: Bram en Max!
                                  // de string Naam is nu veel langer geworden.
 Serial.print("De lengte van Naam met strlen: ");  // tekst
 Serial.println(Naam.length()); // De lengte van Naam met strlen: 34
 // Alternatieve manier om tekst toe te voegen
 Naam = "Hans";                   // even terug naar de oorspronkelijke tekst
 Serial.print("Naam= ");          // tekst
 Serial.println(Naam);            // Naam= Hans
 Naam += " heeft 2 neefjes";    // hier wordt een stukje tekst toegevoegd aan de String.
          // hier wordt  +=  gebruikt, + en +=  kan alleen bij String, niet bij string
 Naam = Naam + ", ze heten Bram" + " en Max"; // nogmaals stukje tekst toegevoegd
 Serial.print("Naam = ");         // tekst
 Serial.println(Naam);            // Naam = Hans heeft 2 neefjes, ze heten Bram en Max
           // het volgende werkt niet met arrays
 Naam = "Een groot getal is: ";// de string 'naam' krijgt deze tekst als inhoud
 Naam = Naam + 32768;             // dit grote getal wordt toegevoegd aan de String.
                                  // met string kan je geen getallen toevoegen
 Serial.print("Naam = ");         // tekst
 Serial.println(Naam);            // Naam = Een groot getal is: 32768
}
void loop() {
  // leave empty for now
}
```

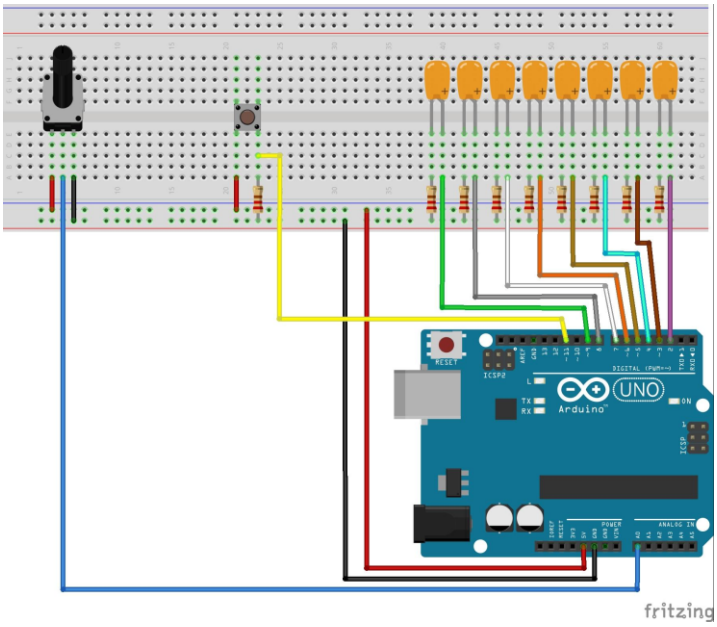
De uitvoer:

```
Naam = Hans
De lengte van Naam met strlen: 4
Naam = Hans heeft 2 neefjes: Bram en Max!
De lengte van Naam met strlen: 34
Naam= Hans
Naam = Hans heeft 2 neefjes, ze heten Bram en Max
Naam = Een groot getal is: 32768
```

Strings kan je ook vergelijken, met wel: “**strcmp()**” (lees dat als “string compare”)
Maar voor besturingen is dat niet nodig. Voor wie het weten wil, het staat in de les!
Deze vergelijkingen zijn gebaseerd op de ASCII waarden van het karakter.

Groeten, Dré

Hieronder het huiswerk van vorige week: de 8 LEDjes als spotschaal en thermometer schaal.
Louis noemde dat staaf en punt diagram met een knopje te selecteren.
De schakeling:



Het programma:

```
// potmeter met 8 LEDjes  incl. knop voor staaf- of puntdiagram

const int led[8]{9,8,7,6,5,4,3,2};      // LED pinnen
const int knop=11;                      // knop pin
const int pot=A0;                       // potmeter pin
unsigned long laatstePot=millis();      // tijdstip waarop voor het laatst op de knop gedrukt is
unsigned long laatsteKnop=millis();     // tijdstip laatste potmeter uitlezing
boolean staaf=true;                     // vlag voor e enkel LEDje (false) of LEDjes vanaf 0
int potWaarde=0;                        // gelezen waarde potmeter (0 - 1023)
int waarde=0;                           // gecalibreerde pot waarde
int i=0;                                // loop teller

void setup() {
  for(i=0;i<8;i++){pinMode(led[i],OUTPUT);}  // 8 LED pinnen op OUTPUT
  pinMode(knop,INPUT);
  pinMode(pot,INPUT);
  Serial.begin(9600);
}

void loop() {
  if(digitalRead(knop)==HIGH){
    if(millis()-laatsteKnop>1000){      // tijd tussen 2 knop bedieningen moet minstens 1
                                        // seconde zijn
      staaf = !staaf;                   // switch de vlag
      laatsteKnop=millis();
    }
  }
  if(millis()-laatstePot>300){          // de potmeter wordt elke 300 milliseconden uitgelezen
    Serial.print(staaf);
    Serial.print(" ");
    potWaarde=analogRead(pot);
    Serial.print(potWaarde);
    Serial.print(" ");
    waarde=potWaarde/128;               // calibreer de pot waarde naar 0 - 7
    Serial.println(waarde);
    if(potWaarde==0){                   // zet alle leds uit als de pot waarde 0 is
```

```
      for(i=7;i>=0;i--){
        digitalWrite(led[i],LOW);
      }
    }
    else{                                   // maak staaf- of puntdiagram
      for(i=7;i>=0;i--){
        if(i>waarde){
          digitalWrite(led[i],LOW);         // zet de LEDjes, die 'boven' de waarde (0 - 7) liggen, uit
        }
        else{
         if(i==waarde){
           digitalWrite(led[i],HIGH);       // zet het LEDje, dat 'gelijk' is aan waarde, aan
         }
         else{
          if(staaf==true){                  // zet de LEDjes, die 'beneden' waarde liggen,
            digitalWrite(led[i],HIGH);      // afhankelijk van de vlag staaf, aan of uit
          }
          else{
            digitalWrite(led[i],LOW);
          }
         }
        }
      }
    }
    laatstePot=millis();
  }
}
```

Zelf had ik ook iets in elkaar geknutseld, ik had ook nog de knight rider functie toegevoegd. Een en ander ook met een knopje te kiezen. Keuze uit drie standen.
Dat knopje werkte niet zo lekker, maar dat heb ik bij onderstaand programma met tips van Louis aangepast! Ook zaten er nog wat schoonheidsfoutjes in, oa efficiënter programmeren. Toch bleef het tobben met het knopje, uiteindelijk met een interrupt routine opgelost.
Mijn ledjes zijn aangesloten op pootjes 3…..11, het knopje op pootje 2, de potm pootje A0.
De LEDs zijn aangesloten als common Anode, waarbij 'aan' een laag niveau op het pootje is.

```
int potm = 0;                       // analoge ingang
int leds[8] = {3, 4, 5, 6, 7, 8, 9, 10}; // ledjes
int potmeter = 0;                   // waarde van de potmeter
int ledstand = 0;                   // de actuele led
int knopje = 2;                     //knopje op pootje 2
const bool uit = HIGH;              // lampje uit als uitgang hoog is
const bool aan = LOW;               // lampje aan bij lage uitgang
volatile int keus = 0;              // keuzestand  voilatile = werkt dwars door alles heen!
int n;                              // hulpvariabele, tellertje

void setup() {
  Serial.begin(115200);                                 // communicatie
  for (n = 0; n < 8; n++) { pinMode(leds[n], OUTPUT); } // zet alle 8 LED pinnen op OUTPUT
  for (n = 0; n < 8; n++) { digitalWrite(leds[n], uit); } // zet alle leds uit
  pinMode(knopje, INPUT);                               // knopje is ingang
  attachInterrupt(0, keuze, FALLING);                   // interrupt -0- bij neergaande flank
}                           // bij neergaande flank wordt de interrupt routine (functie) keuze aangeroepen
                            // interrupt -0- hangt aan pootje -2- en interrupt -1- zit aan pootje -3-

void loop() {
  if (keus == 1) { knightrider(); }             // als de keuzeschakelaar -1- is knightrider uitvoeren
  if (keus == 2) { spotschaal(); }              // als de keuzeschakelaar -2- is, spotschaal uitvoeren
  if (keus == 3) { thermometerschaal(); }       // als de keuzeschakelaar -3- is, thermometerschaal
```

```
}
void keuze(){                                   // keuze functie, interrupt routine
  keus ++;                                      //volgende keuze
  if (keus > 3) { keus = 1; }                   // terug naar keuze 1
  if (keus == 1) { Serial.println("knightrider"); }// tekst op scherm
  if (keus == 2) { Serial.println("spotschaal"); } // tekst op scherm
  if (keus == 3) { Serial.println("thermometerschaal"); } // tekst op scherm
}
void knightrider() {                            // heen en weer gaande LEDs
  for (n = 0; n < 8; n++) {
    digitalWrite(leds[n], aan);                 // zet led aan
    potmeter = analogRead(potm);                // lees potmeterstand
    ledstand = map(potmeter, 0, 1015, 25, 500); // potm omrekenen 25 ms tot500 msec
    delay(ledstand);                            // pauze om LED zichtbaar te maken.
    digitalWrite(leds[n], uit);                 // zet led uit
  }
  for (n = 7; n > 0; n--) {                     // zelfde verhaal in tegengestelde richting
    digitalWrite(leds[n], aan); // zet led aan
    potmeter = analogRead(potm);
    ledstand = map(potmeter, 0, 1015, 25, 500); // potmeterstand varieert van 25 ms tot 500 ms
    delay(ledstand);
    digitalWrite(leds[n], uit);
  }
}
void spotschaal() {
  potmeter = analogRead(potm);
  ledstand = map(potmeter, 0, 1015, 0, 7);      //  omrekenen naar actuele led
  digitalWrite(leds[ledstand], aan);            // zet led aan
  delay(10);                                    // korte pauze om led zichtbaar te maken
  digitalWrite(leds[ledstand], uit);            // zet led uit
}
void thermometerschaal() {
  potmeter = analogRead(0);
  ledstand = map(potmeter, 0, 1015, 0, 8);      //  omrekenen naar actuele led
  for (n = 0; n < 8; n++) {
    if (n < ledstand) {
      digitalWrite(leds[n], uit);               // onderliggende led uitzetten
    }
    else {
      digitalWrite (leds[n], aan);              // bovenliggende LEDs aan
    } // zet led aan
  }
}
```

De 'huiswerk' opdracht van komende week:
Maak iets waarbij strings op het scherm worden geplaatst
Al dan niet met het toetsenbord invoer naar Arduino.
Eventueel ook op een LCD schermpje, al dan niet met I^2C aangestuurd.

Groeten, Dré