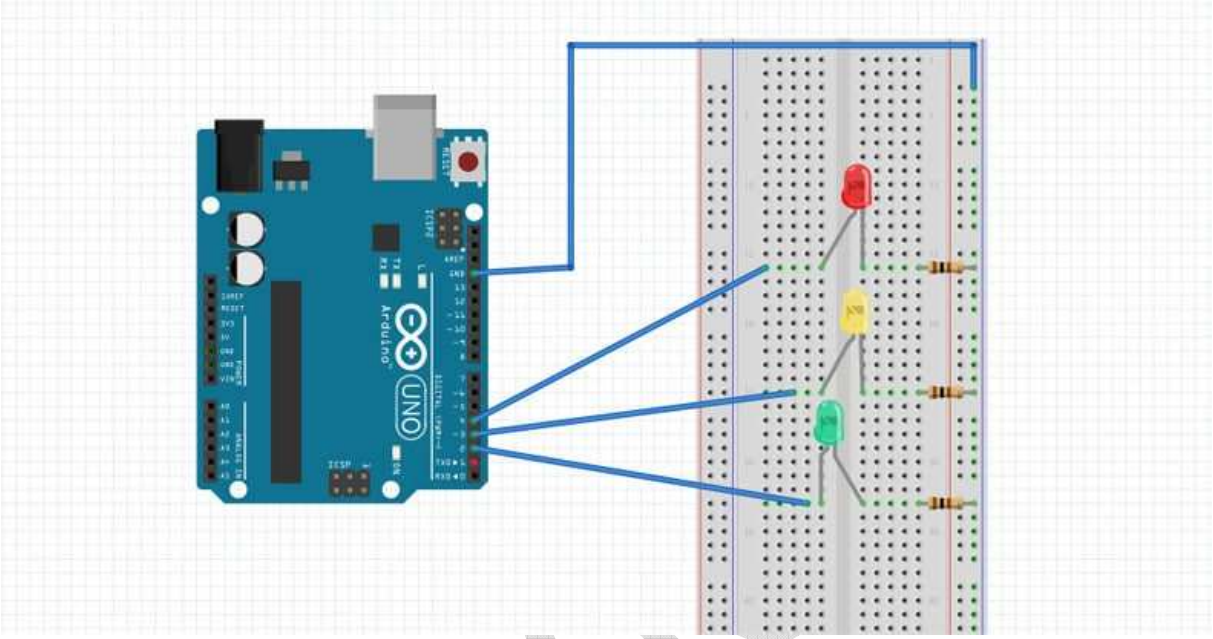
rd

Tweede sessie

1. In deze sessie gaan we even voort spelen met wat we in vorige sessie gezien hebben namelijk het gebruik van een LED met de controller. In deze sessie gaan w een verkeerslicht (rood geel groen) trachten te simuleren.



2. Zoals u ziet op bovenstaande tekening ie het maar een zelfde opstelling maar nu met drie LED's het grote verschil zit in de programmatie van het system. Immers nu moeten we rekening houden met 3 LED, de respectievelijke weerstanden (herinner je nog), 3 aansluitingen op het Arduino bord.
3. Voor deze de opstelling maken laat ons eerst even kijken naar het programma wélke onze sturing zal regelen, natuurlijk terug in de Arduino programmeer omgeving.

```
// variables
int GREEN = 2;
int YELLOW = 3;
int RED = 4;
int DELAY_GREEN = 5000;
int DELAY_YELLOW = 2000;
int DELAY_RED = 5000;
// basic functions
void setup()
{
  pinMode(GREEN, OUTPUT);
  pinMode(YELLOW, OUTPUT);
  pinMode(RED, OUTPUT);
 }
 void loop()
{
  green_light();
  delay(DELAY_GREEN);
  yellow_light();
  delay(DELAY_YELLOW);
  red_light();
```

```
  delay(DELAY_RED);
}

void green_light()
{
  digitalWrite(GREEN, HIGH);
  digitalWrite(YELLOW, LOW);
  digitalWrite(RED, LOW);
}

void yellow_light()
{
  digitalWrite(GREEN, LOW);
  digitalWrite(YELLOW, HIGH);
  digitalWrite(RED, LOW);
}

void red_light()
{
  digitalWrite(GREEN, LOW);
  digitalWrite(YELLOW, LOW);
  digitalWrite(RED, HIGH);
}
```

4. Zoals je ziet is dit programma enkel een uitbreiding van het vorige. Laat het ons eens bekijken:

4.1 we beginnen met de nodige variabelen. Immers zoals we vorige keer gezegd hebben moeten we zorgen dat ons Arduino bord de LED weet staan. Er is een opmerking aan verbonden. Bij het programmeren, en dat in gelijk welke programmeer taal, is de plaats van de code van uitermate belang. In dit geval staan de variabelen bovenaan de maken code. Dat wil zeggen dat alle items hierin beschreven tellen voor gans de lengte van het programma. Dat is handig in sommige gevallen. De benaming is dan ook *publieke variabelen* in tegen stelling tot dze die gebonden zijn aan een onderdeel van een programma welke we dan ook *private variabelen* noemen. Deze zijn dan van tel in het onderdeel waarin ze staan. Eens daarbuiten sterven ze af.

4.2 ons volgende onderdeel **void setup()** ook gezien vorige keer. Dit onderdeel vergeet niet afgesloten met twee ronde haakjes, noemen we een *functie* in ons programma. Dat heb ik in vorige sessie ook uitgelegd. Hierin geven we de verschillende onderdelen een vaste reden van bestaan. Hier is dat output. Zijn tegenhanger, INPUT, wordt gebruikt als we iets aan de processor moeten laten weten. Later komen daar oefeningen over.

4.3 In de functie loop komen we iets anders tegen, namelijk het gebruik van aparte functies. Wat zijn nu aparte functies. Daarvoor laat me daar eens over uitweiden. Een functie in een programmeertaal, heeft tot doel een bepaalde actie te ondernemen op een bepaald tijdstip in het programma. Als men nu een bepaald aantal codes moet gebruiken, kan men dze inbedden in een functie. Als men naderhand die codes nodig heeft, kan men eenvoudig die functie oproepen. Stel dat men die code repetitief (op regelmatige basis) is het veel handiger te werken met een functie i.p.v. dit steeds te moeten herhalen. Tevens als er iets moet verander worden in die code kan dat voor gans het programma door te veranderen in

functie. Dat was de theorie (gedeelte) van de reden voor het bestaan van een functie, althans voor wat we ze nodig hebben.

4.4 In ons geval zien we bijvoorbeeld green_liight(). Verder in het programma komen we de functie green_light(). Daarin staat de code om de groene LED te laten oplichten en de andere te doven. Als we die functie oproepen zal hij de codes doorgeven aan onze Arduino en deze als dusdanig uitvoeren. Let op deze functie heeft geen uitgang (void) en verwacht alsook geen ingangswaarden (ledige ronde haakjes achteraan). Dus dat noemen we een eenvoudige functie vroeger ook procedure genoemd.

4.5 Na deze functie maakt men gebruik van de delay functie (vertraging). Deze functie heeft als ingangswaarde een variabele die publiek is gedeclareerd. Dit is de grootte van de vertragingstijd in ms. Door ze publiek als variabele te declareren hoeft men niet steeds de waarde te typen, enkel de variabele. Dit is een vereenvoudiging als je programma's schrijft van 100-den lijnen.

5. Na dit geprogrammeerd te hebben laat ons dat nu even in praktijk brengen. En vergeet niet wat ons vorige sessie bijgebracht heeft!