MARTIGNAS

Stage
Initiation à la programmation et à l'électronique avec ARDUINO

Jour 2

Programme du jour 2

- PROJET 2: Introduction aux variables
  - Type de variables, utilisation
  - 6 LED à gérer pour simuler les feux d'un croisement
- PROJET 3: L'incrémentation
  - Différentes syntaxes, notion de compteur
- PROJET 4: PWM, variation en douceur d'une LED
  - Les sorties analogiques en PWM

Projet 2 : les variables

- **Le nom d'une variable**
  - Le nom de variable n'accepte que l'alphabet alphanumérique ([a-z], [A-Z], [0-9]) et _ (underscore). Il est unique; il ne peut donc pas y avoir deux variables portant le même nom.

- **Types de variables signées**

Type de variable	Type de nombre stocké	Valeurs maximales du nombre stocké	Nombre sur X bits	Nombre d'octets
char	entier	-128 à +127	8 bits	1 octet
int	entier	-32'768 à +32'767	16 bits	2 octets
long	entier	-2'147'483'648 à +2'147'483'647	32 bits	4 octets
float	décimale	-3.4×10^{38} à $+3.4 \times 10^{38}$	32 bits	4 octets
double	décimale	-3.4×10^{38} à $+3.4 \times 10^{38}$	32 bits	4 octets

Projet 2 : les variables

- **Types de variables non signées (unsigned)**

Type de variable	Type de nombre stocké	Valeurs maximales du nombre stocké	Nombre sur X bits	Nombre d'octets
unsigned char	entier non négatif	0 à 255	8 bits	1 octet
unsigned float	entier non négatif	0 à 65'535	16 bits	2 octets
unsigned double	entier non négatif	0 à 4'294'967'295	32 bits	4 octets

- **Autres types de variables**

Type de variable	Type de nombre stocké	Valeurs maximales du nombre stocké	Nombre sur X bits	Nombre d'octets
boolean	entier non négatif	0 à 1	1 bits	1 octet
byte	entier non négatif	0 à 255	8 bits	1 octet
word	entier non négatif	0 à 65'535	16 bits	2 octets

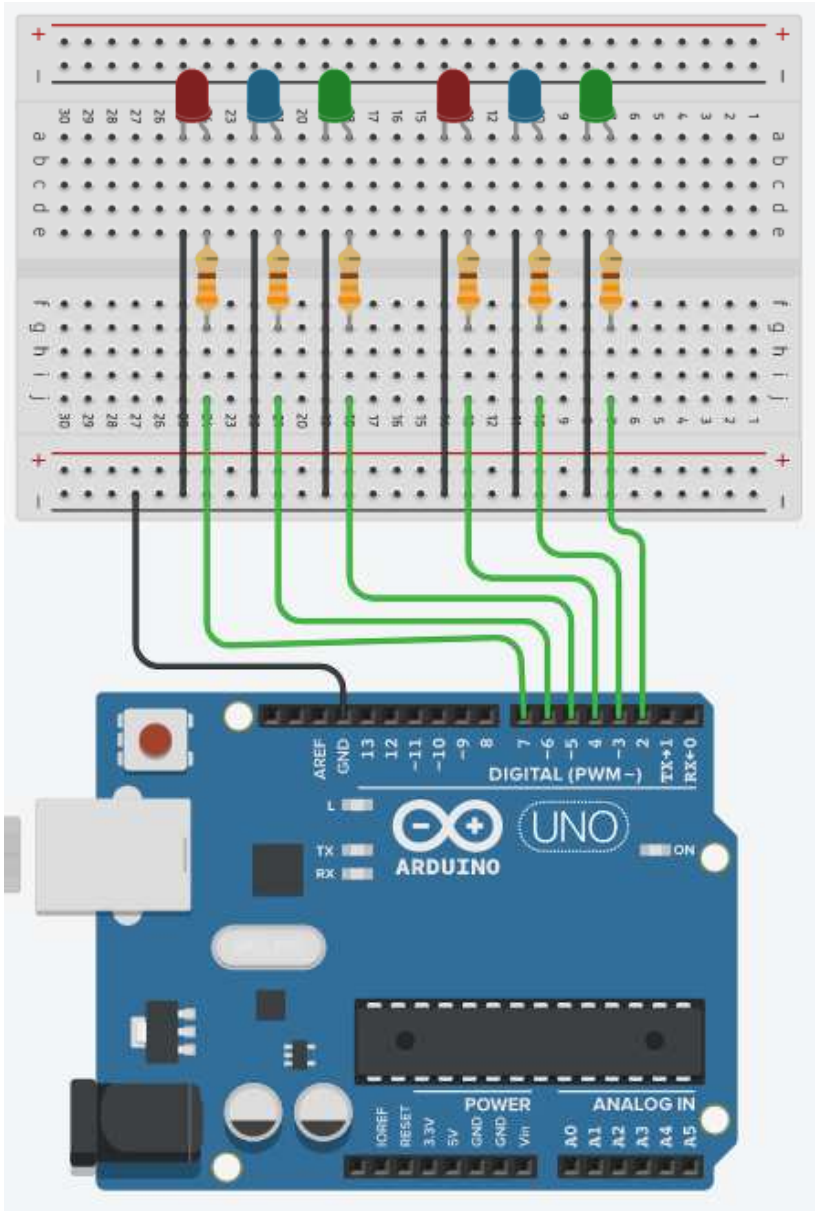
Projet 2 : les variables

- **Exemple pour définir les broches du microcontrôleur**
  - Jusqu'à maintenant, nous avons identifié les broches du microcontrôleur à l'aide de leurs numéros, comme dans l'exemple suivant: *pinMode(0, OUTPUT);*. Cela ne pose pas de problème quand on a une ou deux LEDs connectées. Mais dès qu'on a des montages plus compliqués, cela devient difficile de savoir qui fait quoi. Il est donc possible de renommer chaque broche du microcontrôleur à l'aide de variables.
  - Premièrement, définissons la broche utilisée du microcontrôleur en tant que variable
    - ***const char led_R = 0;***
  - Le terme **const** signifie que l'on définit la variable ***led_R*** comme étant constante, sa valeur ne changera pas dans le programme.

Projet 2 : Gérer les feux de circulation

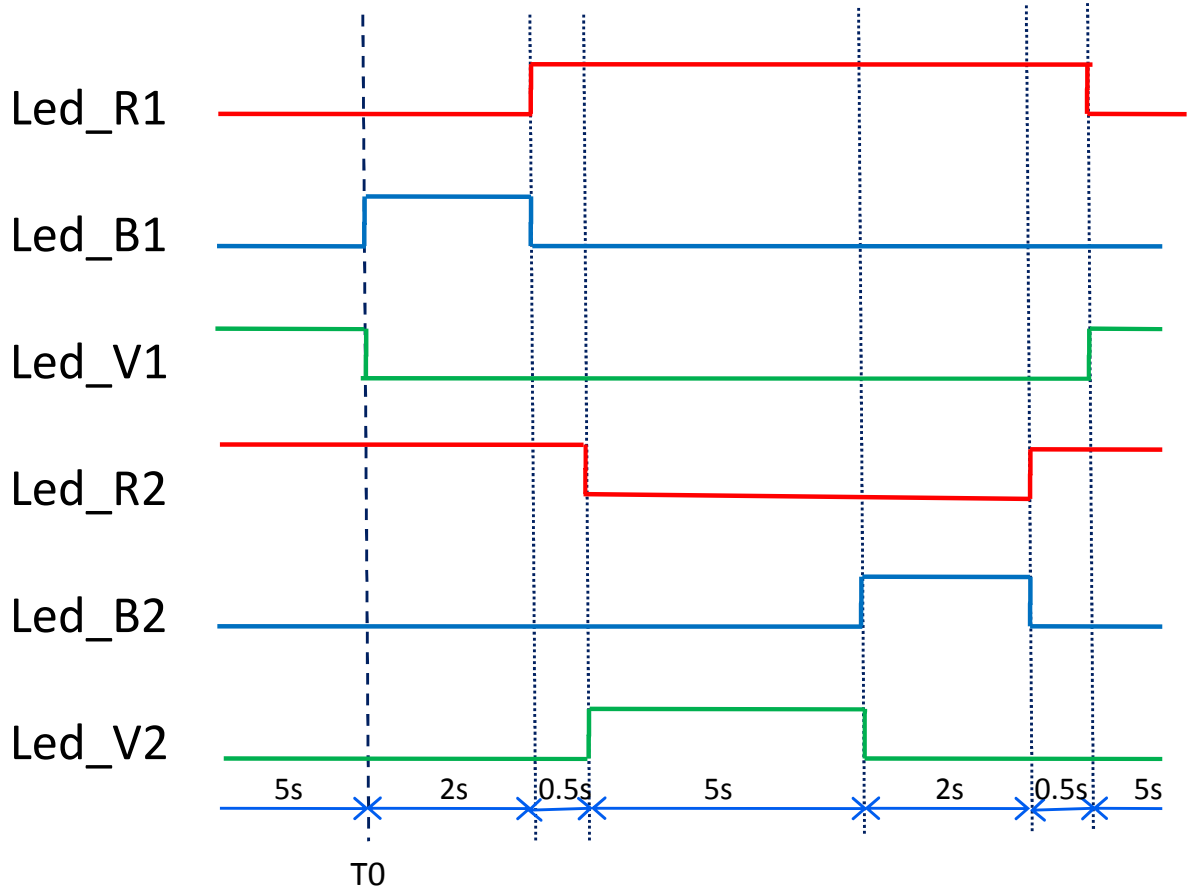
- Données d'entrée :
  - 2 LED rouge + 2 LED verte + 2 LED bleue
  - A connecter sur les pin 2 a 7 de la carte
  - Résistances de limitation de courant de 330 Ω
  - Délais du vert au bleu : 5 s
  - Délais du bleu au rouge : 2s
  - Délais du rouge au vert : 0.5 s

- Travail à effectuer:
  - Utiliser des variables pour définir les pin de la carte
  - Initialiser les pin en sorties
  - Ecrire le code pour animer les LED



Projet 2 : les variables

Séquence des LEDs à réaliser



Projet 2 : les variables

- Rappel des fonctions de base :
  - **pinMode(*broche, mode*);** : Configure la broche spécifiée pour qu'elle se comporte soit en entrée, soit en sortie 0=entrée, 1=sortie.
  - **digitalWrite(*broche, valeur*);** : Met un niveau logique HIGH (1) ou LOW (0) sur une broche numérique définie en sortie.
  - **delay (ms);** : Réalise une pause dans l'exécution du programme pour la durée en millisecondes indiquée en paramètre.

Projet 2 : Gérer les feux de circulation

- Le code du projet
  - On peut faire mieux à vous de proposer un amélioration

```
1  const int LV1 = 2;  // LED Verte droite
2  const int Lb1 = 3;  // LED Bleue droite
3  const int LR1 = 4;  // LED Rouge droite
4  const int LV2 = 5;  // LED Verte gauche
5  const int Lb2 = 6;  // LED Bleue gauche
6  const int LR2 = 7;  // LED Rouge gaiche
7
8  void setup()
9  {
10   pinMode(LV1, OUTPUT);
11   pinMode(Lb1, OUTPUT);
12   pinMode(LR1, OUTPUT);
13   pinMode(LV2, OUTPUT);
14   pinMode(Lb2, OUTPUT);
15   pinMode(LR2, OUTPUT);
16 }
```

```
18 void loop()
19 {
20   digitalWrite(Lb2, LOW);
21   digitalWrite(LR2, HIGH);
22   delay(500); // Wait for 0.5S
23   digitalWrite(LR1, LOW);
24   digitalWrite(LV1, HIGH);
25   delay(5000); // Wait for 5S
26   digitalWrite(LV1, LOW);
27   digitalWrite(Lb1, HIGH);
28   delay(2000); // Wait for 2S
29   digitalWrite(Lb1, LOW);
30   digitalWrite(LR1, HIGH);
31   delay(500); // Wait for 0.5S
32   digitalWrite(LR2, LOW);
33   digitalWrite(LV2, HIGH);
34   delay(5000); // Wait for 5S
35   digitalWrite(LV2, LOW);
36   digitalWrite(Lb2, HIGH);
37   delay(2000); // Wait for 2S
38 }
```

Projet 3 : L'incrémentation

- L'incrémentation :
  - Derrière ce nom barbare se cache une simple opération d'addition avec plusieurs syntaxes possibles
  - Prenons le cas de la variable *var*
    - *var = var + x;* incrémentation de x par une addition classique
    - *var += x;* pour incrémenter de x il est inutile de répéter le nom de la variable
    - *var ++;* c'est une incrémentation par pas de 1 seulement
  - Ces syntaxes sont utilisées par exemple dans la boucle FOR
    - *for(var = 0; var < valeur de fin; var++) {* par pas de 1 à partir de 0
      lignes de code à exécuter
      }
    - *for(var = x; var < valeur de fin; var+=y) {* par pas de y à partir de x
      lignes de code à exécuter
      }

Projet 3 : L'incrémentation

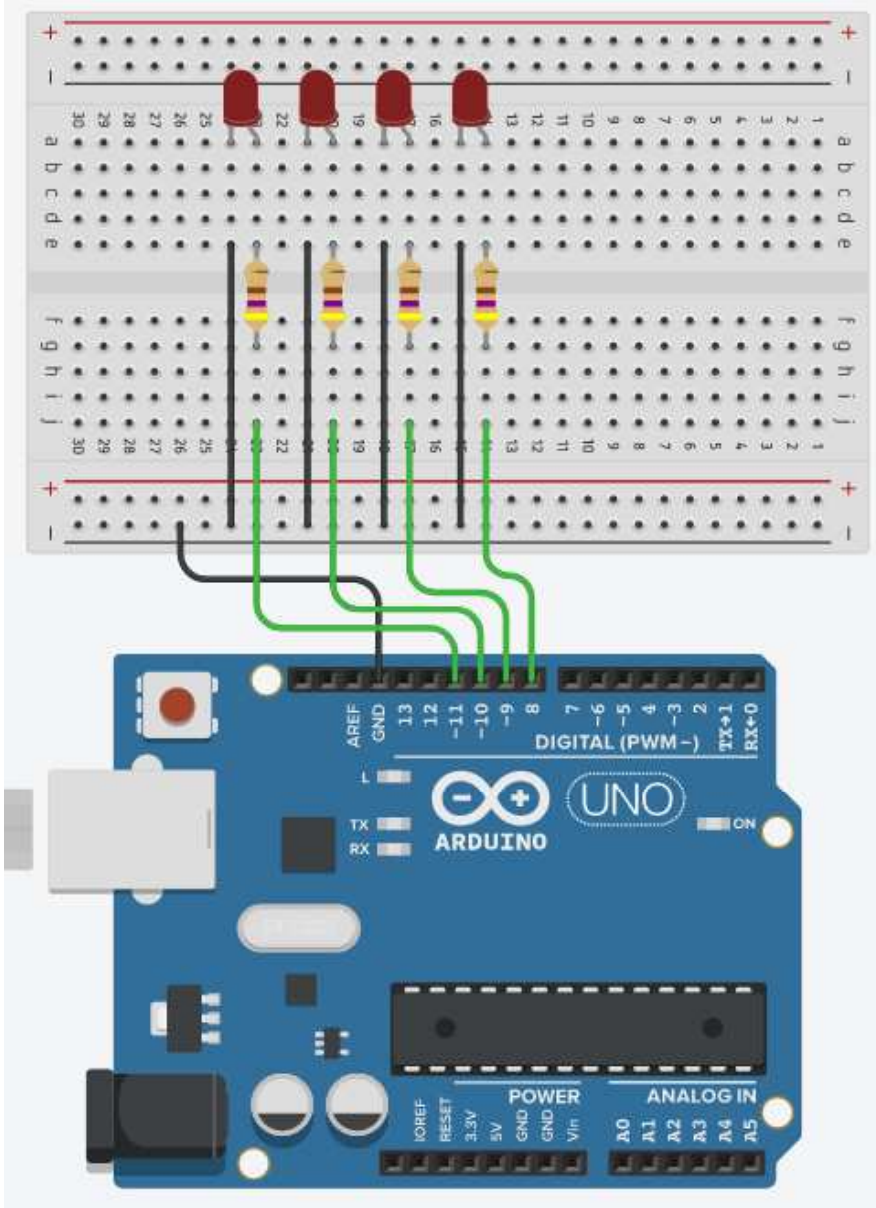
- Boucle for()
  - **Description**
    - L'instruction ***for*** est utilisée pour répéter l'exécution d'un bloc d'instructions regroupées entre des accolades. L'instruction for est très utile pour toutes les opérations répétitives et est souvent utilisées en association avec des tableaux de variables pour agir sur un ensemble de données ou broches.
    - L'initialisation a lieu en premier et une seule fois. A chaque exécution de la boucle, la condition est testée; si elle est VRAIE, le bloc d'instructions et l'incrémentation sont exécutés. Lorsque la condition devient FAUSSE, la boucle stoppe.

  - **Syntaxe**
    - for (initialisation; condition; incrémentation) {
      //instruction(s)à exécuter;
      }

Projet 3 : L'incrémentation

- Données d'entrée :
  - 4 LED rouge
  - A connecter sur les pin 8 a 11 de la carte
  - Résistances de limitation de courant de 330 Ω
  - Durée d'allumage : 100 ms

- Travail à effectuer:
  - Utiliser une boucle FOR pour initialiser les pin de la carte en sorties
  - Ecrire le code pour animer les LED, balayage de droite à gauche puis de gauche à droite en continu (chenillard)



Projet 3 : l'incrémentation

- Le code du projet :
  - on notera que la boucle pour animer les sorties de 13 vers 8 (gauche vers droite) met en œuvre une décrémentation « i-- »

```
1  int timer = 100;     // tempo en ms
2   int i;              // variable de boucle
3
4  void setup()
5  {
6    // Initialisation des sorties à l'aide d'un for
7    for(i=8; i<12; i++)
8      pinMode(i, OUTPUT);
9  }
10
11 void loop()
12 {
13   // boucle de la sortie 8 vers 11
14   for(i=8; i<12; i++){
15     digitalWrite(i, HIGH);  // Allume la LED
16     delay(timer);
17     digitalWrite(i, LOW);   // Eteind la LED
18   }
19
20   // boucle de la sortie 11 vers 8
21   for(i=11; i>7; i--){
22     digitalWrite(i, HIGH);  // Allume la LED
23     delay(timer);
24     digitalWrite(i, LOW);   // Eteind la LED
25   }
26 }
```

Projet 4 : Le PWM

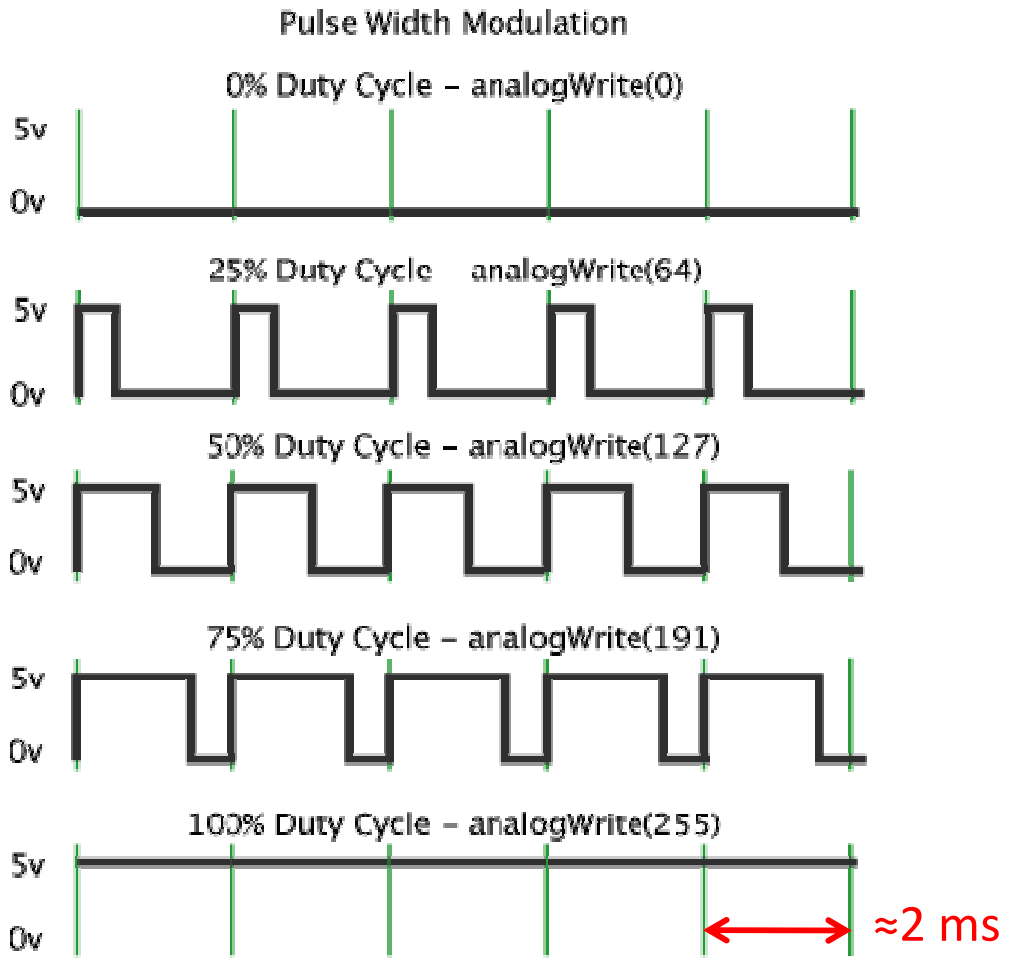
- PWM ça veut dire : Pulse Width Modulation et en français cela donne Modulation à Largeur d'Impulsion (MLI). Le PWM est en fait un signal numérique qui, à une fréquence donnée, a un rapport cyclique qui change (duty cycle).

- La fréquence PWM générée par la carte Arduino de l'ordre de 490Hz, l'espace entre 2 impulsions mesure ≈2 millisecondes

- Un appel de la fonction ***analogWrite(valeur)*** utilise une valeur comprise entre 0 et 255, tel que analogWrite(255) donne 100% du cycle (toujours au niveau HAUT), et analogWrite(127) donne 50% du cycle de travail (la moitié du temps) par exemple. La valeur 0 correspond ainsi à 0% du cycle de travail (duty cycle)

Projet 4 : Le PWM

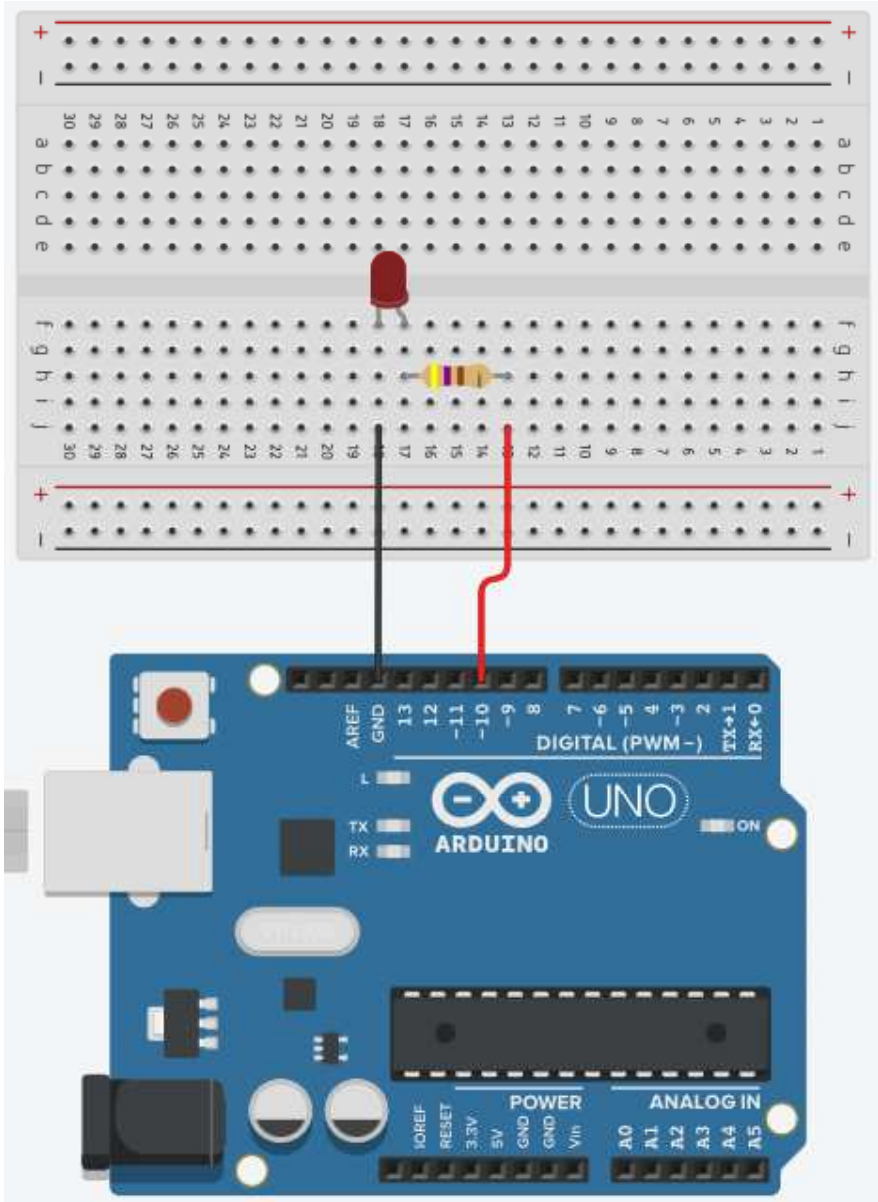
- Résultat sur les pin de sortie en fonction de la valeur dans analogWrite():



Projet 4 : Le PWM

- Données d'entrée :
  - 1 LED rouge
  - A connecter sur la pin 10 de la carte
  - Résistances de limitation de courant de 330 Ω
  - Durée temporisation : 100 ms

- Travail à effectuer:
  - Initialiser la pin 10 en sortie
  - Utiliser une boucle for pour faire varier la luminosité de la LED de 0 à 100% par pas de 10%
  - Variation croissante puis décroissante



Projet 4 : Le PWM

- Fonction : analogWrite()
  - **Description**
    - Génère une impulsion de largeur / période voulue sur une broche de la carte Arduino (onde PWM - Pulse Width Modulation en anglais). Ceci peut-être utilisé pour faire briller une LED avec une luminosité variable ou contrôler un moteur à des vitesses variables.
    - Après avoir appelé l'instruction analogWrite(), la broche générera une onde carrée stable avec un "duty cycle" (fraction de la période où la broche est au niveau haut) de longueur spécifiée (en %), jusqu'à l'appel suivant de l'instruction analogWrite() (ou bien encore l'appel d'une instruction digitalRead() ou digitalWrite() sur la même broche). La fréquence de l'onde PWM est approximativement de 490 Hz
  - **Syntaxe**
    - analogWrite(*broche, valeur*);
  - **Paramètres**
    - ***broche***: la broche utilisée pour "écrire" l'impulsion. Cette broche devra être une broche ayant la fonction PWM, sur la UNO, ce pourra être une des broches 3, 5 ,6 ,9 ,10 ou 11
    - ***valeur*** : la largeur du "duty cycle" (proportion de l'onde carrée qui est au niveau HAUT) : entre 0 (0% HAUT donc toujours au niveau BAS) et 255 (100% HAUT donc toujours au niveau HAUT)
  - **Valeur retournée**
    - Aucune

Projet 4 : Le PWM

- Le code du projet PWM

```
const int pwm = 10; // sortie pwm choisie
int pas = 25;       // incrémentation du PWM par pas de 10%
int i;              // variable de boucle
int tempo = 100;    // temporisation en ms

void setup()
{
  pinMode(pwm, OUTPUT);
}

void loop()
{
  // incrémentation
  for(i=0; i<256; i+=pas) {
    analogWrite(pwm, i);
    delay(tempo);
  }

  // décrémentation
  for(i=255; i>0; i-=pas) {
    analogWrite(pwm, i);
    delay(tempo);
  }
}
```

MARTIGNAS

Merci pour votre attention

Avez-vous des questions ?