REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE LA FORMATION ET DE L'ENSEIGNEMENT PROFESSIONNELS

IFEP SBA

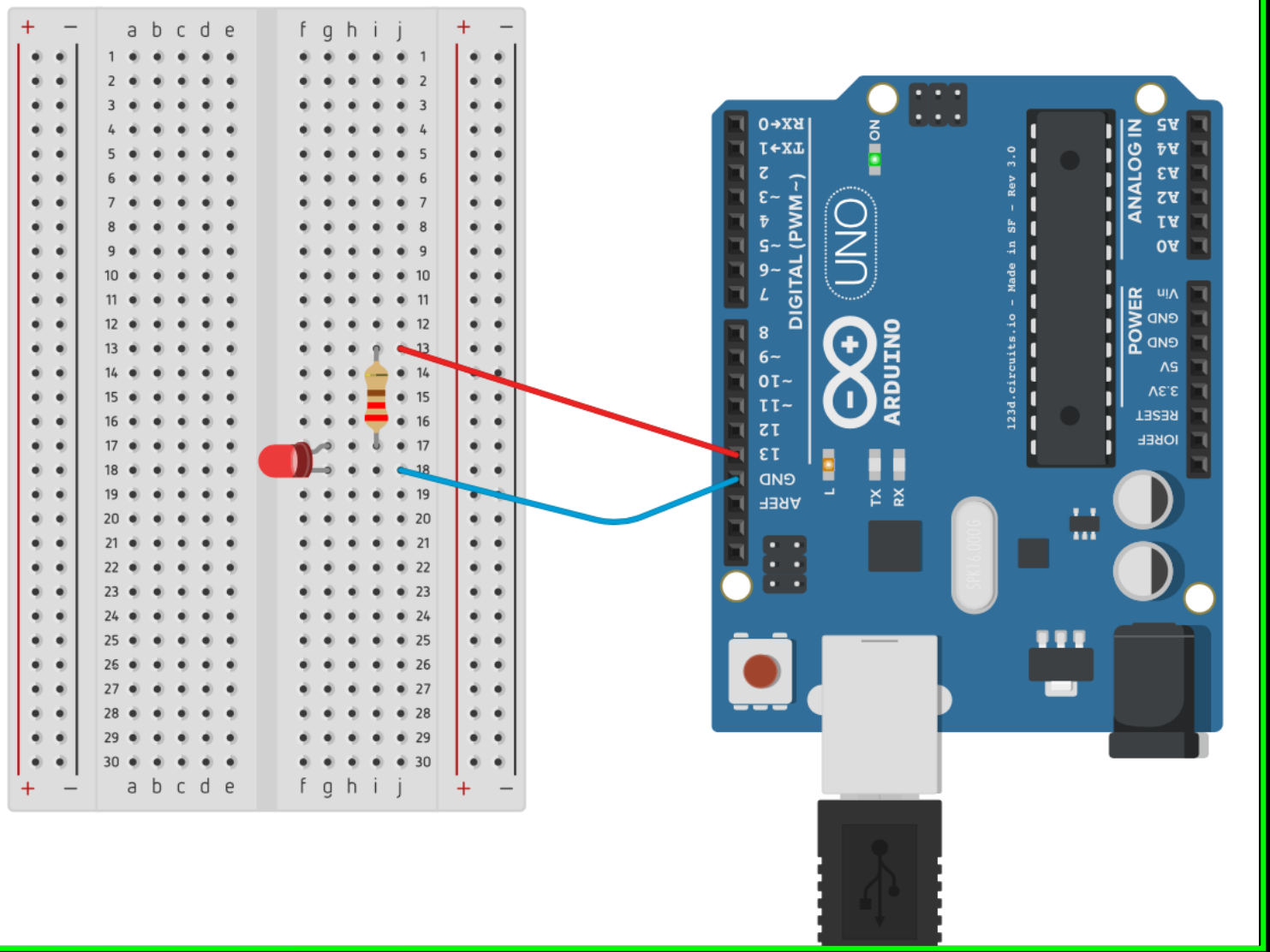
PROGRAMMATION DE ARDOUINO

Présenté par :

HALAILI Mohamed

TP N°01 Faire clignoté une LED

1/ Câbler le circuit suivant



TP N°01 Faire clignoté une LED

2/ Charger le programme suivant

```
/*
Allume la LED pendant 1 seconde, puis l'éteint pendant 1 seconde.
*/
// le code dans cette fonction est exécuté une fois au début
void setup() {
pinMode(13, OUTPUT); // Initialise la broche 13 comme sortie
}
// le code dans cette fonction est exécuté en boucle
void loop() {
digitalWrite(13, HIGH); // allumer la LED, tension 5V sur la broche 13
delay(1000); // attendre 1000ms=1s
digitalWrite(13, LOW); // éteindre la LED, tension 0V sur la broche 13
delay(1000); // attendre 1 seconde
}
```

TP N°01 Faire clignoté une LED

2/ Charger le programme suivant

```
void setup() {
pinMode(13, OUTPUT);
}

void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

TP N°01 Faire clignoté une LED

2/ Charger le programme suivant

```
void setup() {
pinMode(13, OUTPUT);
}

void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

TP N°02 Faire clignoté une LED sur la Broche 10

1. Brancher la LED sur la broche 10
2. Modifier le programme pour faire clignoter la LED (Allumé pondant 0,5s , éteinte pondant 1s)

```
void setup() {
pinMode(10, OUTPUT);
}
void loop() {
digitalWrite(10, HIGH);
delay(500);
digitalWrite(10, LOW);
delay(1000);
}
```

ANALYSE DU PROGRAMME

Programme qui fait clignoté une LED connecté à la broche 13

```
void setup() {
pinMode(13, OUTPUT);
}

void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

ANALYSE DU PROGRAMME

La ligne pinMode(13, OUTPUT); initialise la broche 13 du microcontrôleur comme sortie, c'est-à-dire que des données seront envoyées depuis le microcontrôleur vers cette broche

```
void setup() {
pinMode(13, OUTPUT);
}

void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

ANALYSE DU PROGRAMME

Avec l'instruction digitalWrite(13, HIGH);, le microcontrôleur connecte la broche D13 au +5V ce qui a pour effet d'allumer la LED .

```
void setup() {
pinMode(13, OUTPUT);
}

void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

ANALYSE DU PROGRAMME

L'instruction delay(1000); indique au microcontrôleur de ne rien faire pendant 1000 millisecondes, soit 1 seconde.

```
void setup() {
pinMode(13, OUTPUT);
}

void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

ANALYSE DU PROGRAMME

Avec l'instruction digitalWrite(13, LOW);, le microcontrôleur connecte la broche D13 à la masse (Gnd) ce qui a pour effet d'éteindre la LED.

```
void setup() {
pinMode(13, OUTPUT);
}

void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

ANALYSE DU PROGRAMME

L'instruction delay(1000); indique au microcontrôleur à nouveau de ne rien faire pendant 1000ms soit 1seconde.

```
void setup() {
pinMode(13, OUTPUT);
}

void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

ANALYSE DU PROGRAMME

Le résultat est donc que la LED s'allume pendant 1seconde, puis s'éteint pendant une 1seconde puis s'allume pendant 1seconde et s'éteint pendant une 1seconde et ainsi de suite (donc elle clignote).

```
void setup() {
pinMode(13, OUTPUT);
}

void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

TP N°03 Faire clignoté 2 LEDs

1. Réaliser une simulation qui fait clignoter simultanément deux LED (LED1 et LED2)

```
void setup() {
pinMode(10, OUTPUT);
pinMode(11, OUTPUT);
}
void loop() {
digitalWrite(10, HIGH);
digitalWrite(11, HIGH);
delay(1000);
digitalWrite(10, LOW);
digitalWrite(11, LOW);
delay(1000);
}
```

TP N°03 Faire clignoté 2 LEDs

1. Modifier le programme des 2 LEDs pour qu'ils puissent clignoter par alternance

```
void setup() {
pinMode(10, OUTPUT);
pinMode(11, OUTPUT);
}
void loop() {
digitalWrite(10, HIGH);
digitalWrite(11, LOW);
delay(1000);
digitalWrite(10, LOW);
digitalWrite(11, HIGH);
delay(1000);
}
```

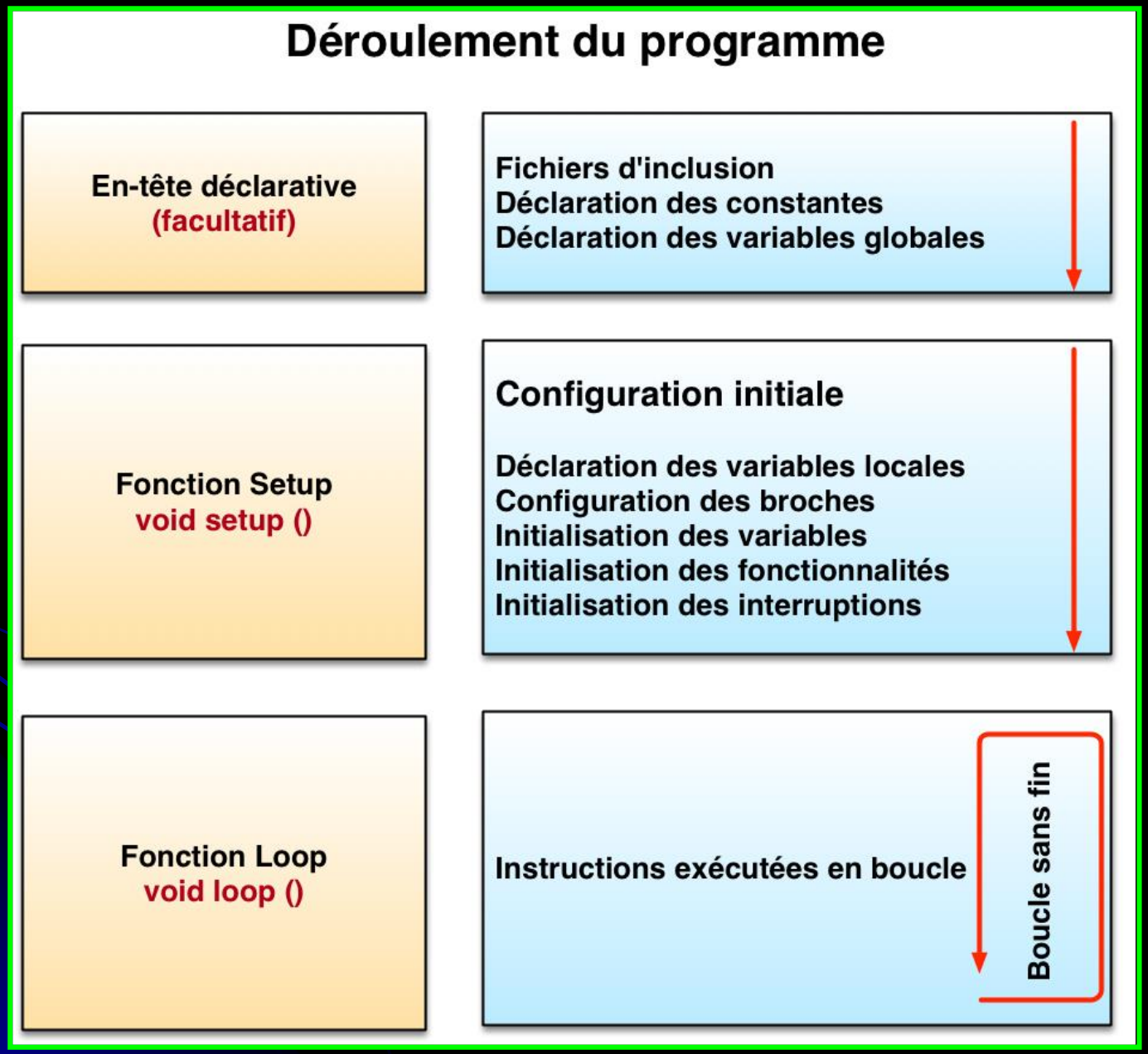
Programmation de l'Ardouino « Langage C »

A. Le déroulement du programme

Le programme se déroule de la façon suivante :

1. Prise en compte des instructions de la partie déclarative
2. Exécution de la partie configuration (fonction setup( )),
3. Exécution de la boucle sans fin (fonction loop ( )): le code compris dans la boucle sans fin est exécuté indéfiniment.

Programmation de l'Ardouino « Langage C »



Programmation de l'Ardouino « Langage C »

B. Le code minimal

Avec Arduino, nous devons utiliser un code minimal lorsqu'on crée un programme. Ce code permet de diviser le programme en deux parties.

```
Void setup()
{
}
Void loop()
{
}
```

B.1 La fonction

Dans le code 1, se trouvent deux fonctions. Les fonctions sont en fait des portions de code

Programmation de l'Ardouino « Langage C »

Première fonction:

```
Void setup()
{
}
```

Cette fonction **setup()** est appelée une seule fois lorsque le programme commence. C'est pourquoi c'est dans cette fonction que l'on va écrire le code qui n'a besoin d'être exécuté qu'une seule fois. On appelle cette fonction : "**fonction d'initialisation**". On y retrouvera la mise en place des différentes sorties et quelques autres réglages.

Programmation de l'Ardouino « Langage C »

Une fois que l'on a initialisé le programme on utilisant la première fonction **void setup()**, il faut ensuite créer son "**coeur**", autrement dit le programme en lui même.

Deuxième fonction:

```
void loop()
{
}
```

Programmation de l'Ardouino « Langage C »

Deuxième fonction:

```
void loop()
{
}
```

C'est donc dans cette fonction **loop()** que l'on va écrire le contenu du programme. Il faut savoir que cette fonction est appelée en permanence, c'est-à-dire qu'elle est exécutée une fois, puis lorsque son exécution est terminée, on la **réexécute**, encore et encore. On parle de **boucle infinie**.

Programmation de l'Ardouino « Langage C »

Les instructions

Les instructions sont des lignes de code . Ce sont donc les ordres qui seront exécutés par l'Arduino. Il est très important de respecter exactement la syntaxe; faute de quoi, le code ne pourra pas être exécuté.

Programmation de l'Ardouino « Langage C »

Les points virgules ;

Les **points virgules** terminent les instructions. Attention: Les points virgules (;) sont synonymes d'erreurs car il arrive très souvent de les oublier à la fin des instructions. Par conséquent le code ne marche pas et la recherche de l'erreur peut nous prendre un temps conséquent ! Donc faites bien attention.

Exemple:

```
pinMode(13, OUTPUT);
digitalWrite(13, HIGH);
delay(1000);
```

Programmation de l'Ardouino « Langage C »

Les accolades { }

Les accolades sont les « **conteneurs** » du code du programme. Elles sont propres aux fonctions, aux conditions et aux boucles. Les instructions du programme sont écrites à l'intérieur de ces accolades. Parfois elles ne sont pas obligatoires dans les conditions, mais il est recommandable de les mettre tout le temps ! Cela rendra plus lisible votre programme.

Programmation de l'Ardouino « Langage C »

Les commentaires

Les commentaires sont des lignes de codes qui seront ignorées par le programme. Elles ne servent en rien lors de l'exécution du programme. Ils permettent d'annoter et de commenter le programme.

EXEMPLE:

Ligne unique de commentaire :

```
// le code dans cette fonction est exécuté une fois au début
```

Ligne ou paragraphe sur plusieurs lignes :

```
/*
Allume la LED pendant 1 seconde, puis l'éteint pendant 1
seconde.
*/
```

Programmation de l'Ardouino « Langage C »

Les accents

Il est formellement interdit de mettre des accents en programmation! Sauf dans les commentaires...

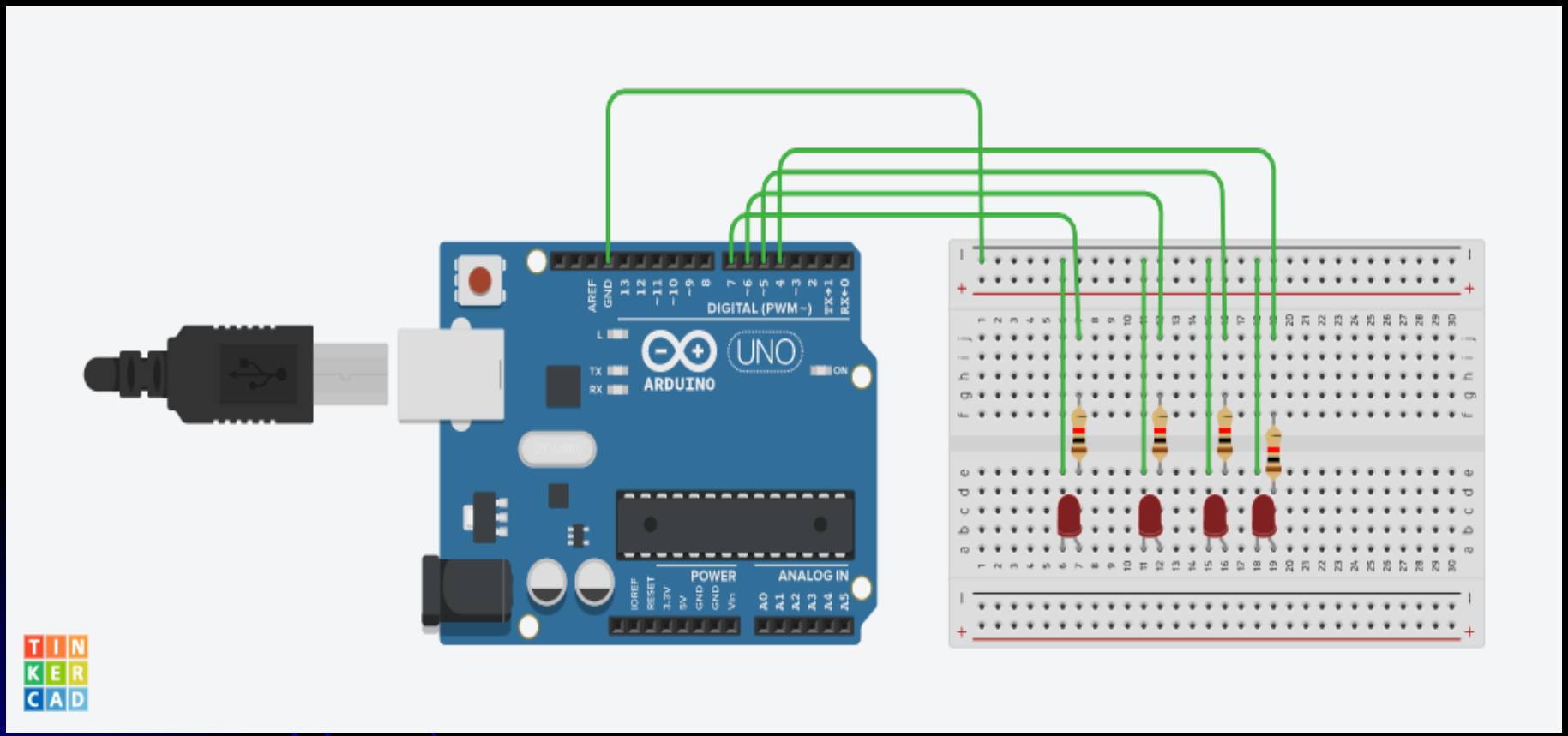
Programmation de l'Ardouino « Langage C »

TPN°04: Réaliser un chenillard à 4 LEDs.

Un chenillard est un ensemble de LED dont le programme contrôle leur allumage et leur extinction les unes à la suite des autres.

Programmation de l'Ardouino « Langage C »

TPN°04: Chenillard à 4 LEDs.



TPN°04: Chenillard à 4 LEDs.

Programme : Première Méthode

```
/*
Objectif: faire un chenillard à 4 LEDs montées sur les broches 10 à 13
*/
void setup() // début de la fonction setup()
{
pinMode( 4, OUTPUT); // Initialise la broche 10 comme sortie
pinMode( 5, OUTPUT); // Initialise la broche 11 comme sortie
pinMode(6, OUTPUT); // Initialise la broche 12 comme sortie
pinMode( 7, OUTPUT); // Initialise la broche 13 comme sortie
Serial.begin(9600); // Ouvre le port série à 9600 bauds
} // fin de la fonction setup()
```

Serial.begin(9600) initialise **le** port série qui permet à l'Arduino d'envoyer et de recevoir des informations à l'ordinateur.

TPN°04: Chenillard à 4 LEDs.

Programme : Première Méthode

```
void loop() // début de la fonction loop()
{
digitalWrite(4, HIGH); // Met la broche 4 au niveau haut = allume la LED
digitalWrite(5, LOW); // Met la broche 5 au niveau bas = éteint la LED
digitalWrite(6, LOW); // Met la broche 6 au niveau bas = éteint la LED
digitalWrite(7, LOW); // Met la broche 7 au niveau bas = éteint la LED
delay(500); // Pause de 100ms
digitalWrite(4, LOW); // Met la broche 4 au niveau bas = éteint la LED
digitalWrite(5, HIGH); // Met la broche 5 au niveau haut = allume la LED
delay(500); // Pause de 100ms
digitalWrite(5, LOW); // Met la broche 5 au niveau bas = éteint la LED
digitalWrite(6, HIGH); // Met la broche 6 au niveau haut = allume la LED
delay(500); // Pause de 100ms
digitalWrite(6, LOW); // Met la broche 6 au niveau bas = éteint la LED
digitalWrite(7, HIGH); // Met la broche 7 au niveau haut = allume la LED
delay(500); // Pause de 100ms
} // fin de la fonction loop
```

TPN°04: Chenillard à 4 LEDs.

Programme : Deuxième Méthode

```
// Initialisation des lignes 4 à  7 en sortie
void setup ()
{
  pinMode (4, OUTPUT) ;
  pinMode (5, OUTPUT) ;
  pinMode (6, OUTPUT) ;
  pinMode (7, OUTPUT) ;
}
```

TPN°04: Chenillard à 4 LEDs.

Programme : Deuxième Méthode

```
// Fonction loop
void loop ()
{    // Extinction de toutes les DEL au départ du programme
  for (byte i = 4 ; i <= 7 ; i++) {
    digitalWrite (i, LOW) ; // éteint la DEL reliée a  la broche i
  }
  // Boucle pour faire flasher les DEL
  for (byte i = 4 ; i <= 7 ; i++) {
    digitalWrite (i, HIGH) ; // allume la DEL sur broche i
    delay (50) ; // durée du flash 50 millisecondes
    digitalWrite (i, LOW) ; // éteint la DEL
  }
  delay (500) ; // délai de 500 millisecondes
  // Recommence la séquence
}
```