	Sciences et Technologies de l'Industrie et du Développement Durable		1 ^{ère} STI2D
	TUTORIEL ARDUINO AVEC TINKERCAD		
			IT

1. Présentation

	<p>CARTE</p> <p>Arduino est une plateforme matérielle et logicielle de développement d'applications embarquées.</p> <p>Elle se compose d'une carte électronique basée autour d'un <u>microcontrôleur</u> (ATMEL AVR) comportant un certain nombre d'entrées et de sorties (les broches ou pins <u>analogiques et numériques</u>) permettant la connexion de <u>capteurs, ou d'actionneurs</u>.</p>
	<p>Logiciel ou IDE :</p> <p>Le logiciel de programmation des modules Arduino est une application <u>Java</u>, libre servant d'éditeur de code et de compilateur, et qui peut transférer le <u>firmware</u> et le programme au travers de la liaison USB.</p> <p>Le langage de programmation utilisé est un mélange de <u>C</u> et de <u>C++</u>, restreint et adapté aux possibilités de la carte.</p>
	<p>Modules « shield »</p> <p>Cartes supplémentaires se connectant sur le module Arduino pour augmenter les possibilités et les capacités de la carte.</p> <p>Exp : afficheur graphique couleur, interface <u>Ethernet</u>, GPS, WIFI, GSM, <u>Carte de puissance moteur</u>.....</p>

Q1 : Utiliser internet et expliquer les mots soulignés en rouge.

Q2 : Chercher les broches analogiques et les broches numériques sur la carte.

2. Utilisation d'un simulateur

Cette partie a pour objectif la mise en œuvre d'un simulateur d'Arduino.

Ouvrir le navigateur **Chrome**, et taper l'adresse "**https://www.tinkercad.com/circuits**" dans la barre des adresses puis cliquer sur "**S'INSCRIRE**" et suivre les étapes de 1 à 12.

The screenshots illustrate the registration process on the Tinkercad website:

- Step 1:** The browser address bar shows `tinkercad.com/circuits`.
- Step 2:** The 'Connexion' menu is open, and the 'S'INSCRIRE' button is highlighted.
- Step 3:** The 'In School' section shows the 'General account' button highlighted.
- Step 4:** The 'Créer un compte' form shows the 'Pays, France' dropdown selected.
- Step 5:** The 'Créer un compte' form shows the 'SUIVANT' button highlighted.
- Step 6:** The 'Créer un compte' form shows the 'Votre adresse mail' field highlighted.
- Step 7:** The 'Créer un compte' form shows the 'Créer un mot de passe, un mail de confirmation vous sera envoyé. Ensuite connectez-vous.' text highlighted.
- Step 8:** The 'Créer un compte' form shows the 'CRÉER UN COMPTE' button highlighted.
- Step 9:** The 'Créer un compte' form shows the 'Circuits' button highlighted.
- Step 10:** The 'Créer un compte' form shows the 'Créer un circuit' button highlighted.
- Step 11:** The 'Créer un compte' form shows the 'Composants De base' dropdown highlighted.
- Step 12:** The 'Créer un compte' form shows the 'Arduino' component highlighted in the search results.

10

11

12

13

14

15

Changer la valeur 1000 par 2000 et démarrer la simulation une deuxième fois

Q3 : Chercher sur le site arduino la signification des lignes de suivantes du code.

digitalWrite :

analogWrite :

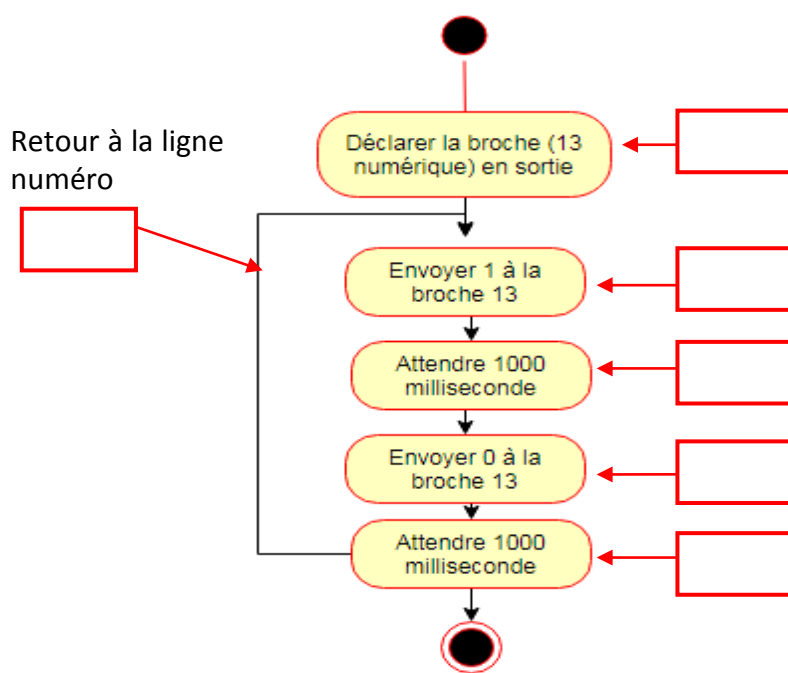
pinMode :

void setup() :

void loop() :

Q4 : Que signifie le numéro 13 sur le code ?

Q5 : Soit l'organigramme suivant, mettre les lignes **8, 14, 15, 17, 18** du programme à l'endroit correspondant.

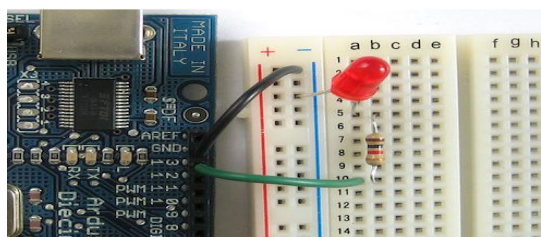


Q6 : remplacer le programme existant par le programme numéro 1 ensuite le programme numéro 2. Comparer en utilisant les mots clés suivants : **VARIABLE, SOUS PROGRAMME.**

Programme 1	Programme 2
<pre> int maled=13; void setup(){//----- pinMode(maled, OUTPUT); } void loop(){ //----- digitalWrite(maled, HIGH); delay(1000); digitalWrite(maled, LOW); delay(1000); } </pre>	<pre> int maled=13; void setup(){//----- pinMode(maled, OUTPUT); } void loop(){ //----- ledoff(); delay(1000); ledon(); delay(1000); } void ledon(){//----- digitalWrite(maled, HIGH); } void ledoff(){//----- digitalWrite(maled, LOW); } </pre>

3. Câblage d'un circuit "virtuel"

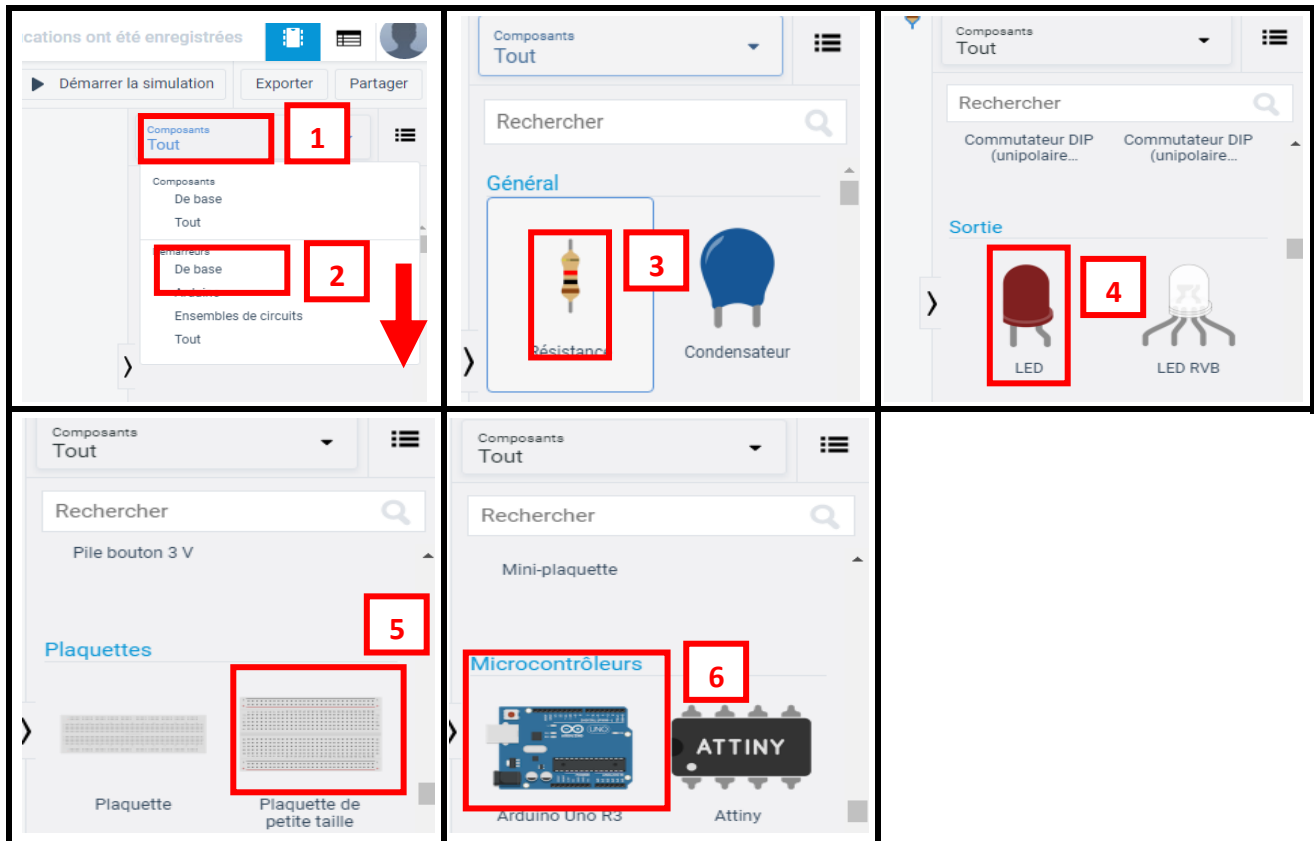
On désire que la carte Arduino fasse clignoter deux **leds** branchées sur deux sorties digitales avec une plaque d'essai.



Il faut tout d'abord **ajouter les composants**. Deux **leds**, deux résistances, Une breadboard (plaque d'essai), un arduino.

Cliquer sur composants et naviguer dans la liste de composants grâce à l'**ascenseur sur la droite**.

Effectuer un "**Click and Drop**" pour déposer les composants dans la zone de conception :

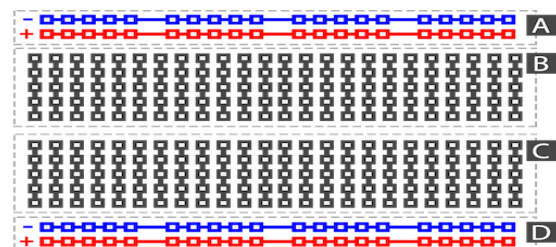


Réflexion et câblage !!!!!!!

Un mot sur la **Breadboard** :

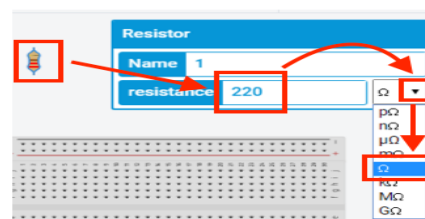
On peut traduire par "Planche à pain". C'est une **platine de prototypage** qui permet de câbler facilement des composants.

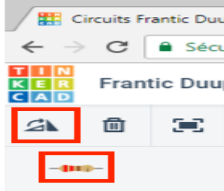
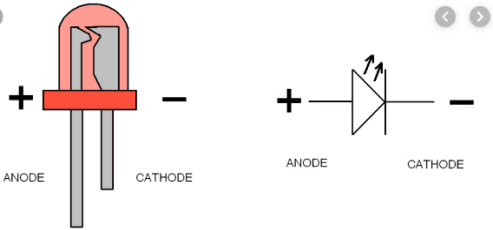


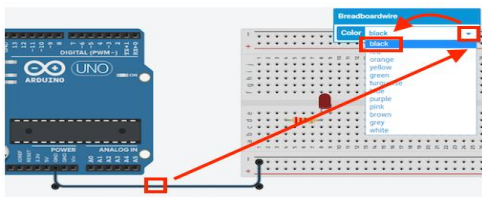
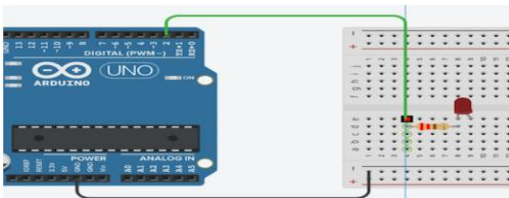
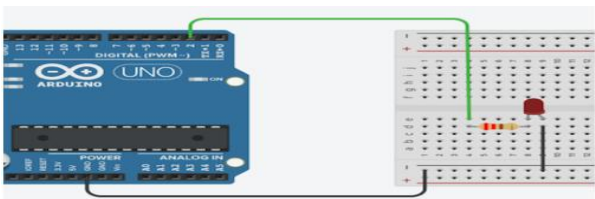
Les **lignes de connexions** sont représentées ci-contre :



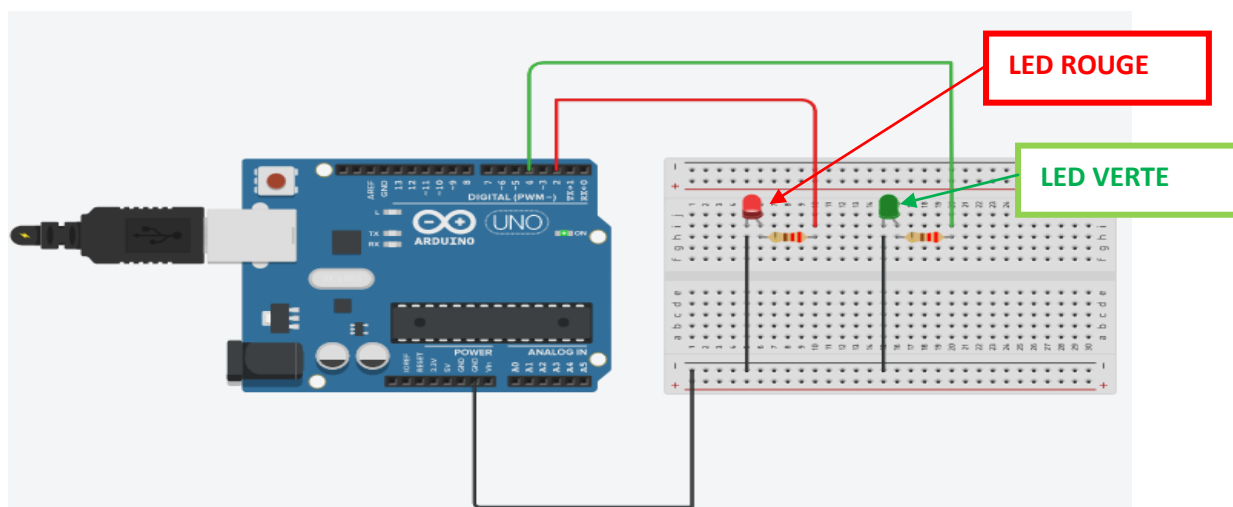
Les sorties digitales de la carte Arduino délivrent une **tension trop élevée pour les LED**. Il faut donc les **protéger avec une résistance de 220 ohm**.

Afin de changer la valeur de la résistance, **cliquer sur la résistance**, puis entrer la **valeur "220"** et enfin sélectionner l'**unité "ohm"** :



<p>La résistance est mal orientée pour notre montage.</p> <p>Cliquer sur la résistance et cliquer (<i>plusieurs fois</i>) sur le bouton orienter afin que la LED soit verticale.</p>	
<p>Rien ne se passe si la LED est câblée à l'envers !</p> <p>En effet la patte "Anode" doit être reliée à la sortie numérique ensuite la résistance, alors que la cathode doit être reliée à la masse.</p> <p>On peut repérer l'anode car sa patte est la plus (+)</p>	
<p>Disposer les composants tel qu'indiqué sur l'illustration ci-contre (<i>attention, il faut que l'une des pattes de la résistance soit connectée avec une patte de la LED</i>) :</p>	
<p>Il faut maintenant réaliser les fils de connexion. On va relier la pin de masse de la carte Arduino (GND) sur la ligne (-) de la breadboard.</p> <p>Cliquer sur la pin GND et rejoindre la ligne (-) en cliquant pour couder le fil :</p>	
<p>Ce premier fil étant le fil de masse (Ground ou GND en anglais), il faut qu'il soit de couleur de noir.</p> <p>Cliquer sur le fil et changer sa couleur pour du noir</p>	
<p>On désire utiliser la pin digitale n°2 de la carte Arduino pour allumer la LED.</p> <p>Réaliser la connexion entre la pin 2 et la ligne de breadboard reliée à la patte de la résistance :</p>	
<p>Enfin, relier la 2ème patte de la LED à la masse avec un fil noir :</p>	

Objectif : Réaliser le schéma ci-dessous :



4. Programmation du montage

Faire clignoter les deux leds (rouge et verte) du montage précédent une après l'autre avec deux méthodes différentes.

Pour cela, il faut mettre le programme méthode 1 dans la zone de programmation texte et simuler. Faire la même chose avec le programme méthode 2 .

Méthode1	Méthode2
<pre>void setup() { pinMode(2, OUTPUT); pinMode(4, OUTPUT); } void loop() { digitalWrite(2, HIGH); delay(1000); // Wait for 1000 millisecond(s) digitalWrite(2, LOW); digitalWrite(4, HIGH); delay(1000); // Wait for 1000 millisecond(s) digitalWrite(4, LOW); } }</pre>	<pre>int rouge=2; int vert=4; void setup(){ pinMode(rouge, OUTPUT); pinMode(vert, OUTPUT); } void loop(){ ledon(rouge); ledoff(rouge); ledon(vert); ledoff(vert); } void ledon(int n){ digitalWrite(n, HIGH); delay(1000); } void ledoff(int s){ digitalWrite(s, LOW); } }</pre>

Q7 : Réaliser l'algorithme du programme méthode 1.

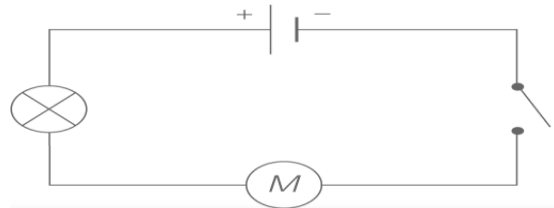
Q8 : Comparer les deux programmes. Quel est l'intérêt d'utiliser la méthode2 alors que son programme est plus long ?.

Q9 : Rajouter une troisième led de couleur bleu et faire clignoter les trois une après l'autre avec les deux méthodes.

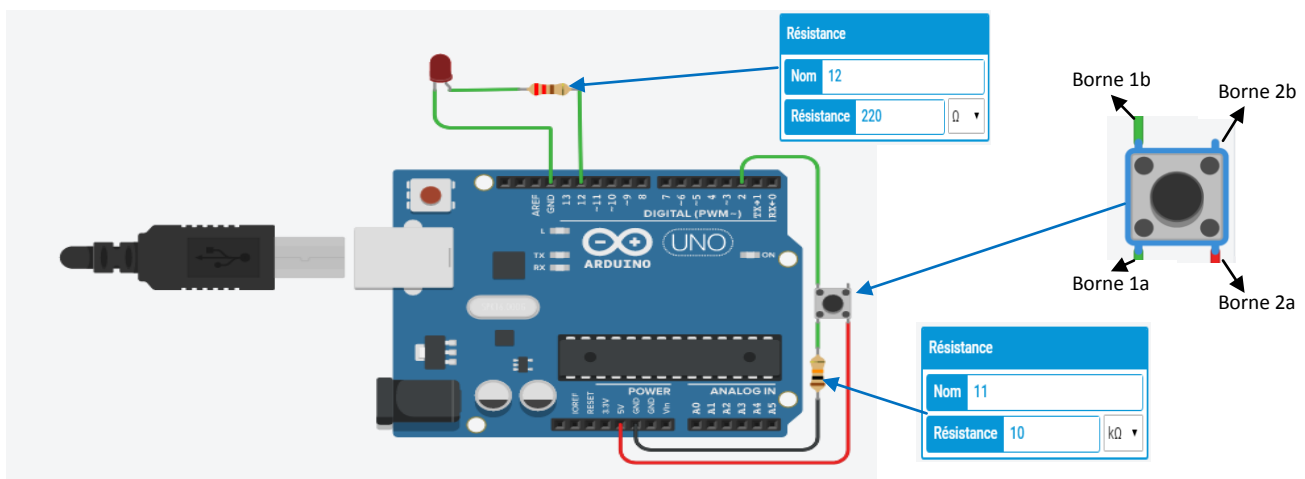
5. Les différents composants électroniques

5.1) L'interrupteur

Un **interrupteur** est un organe, physique ou virtuel, permettant d'interrompre ou d'autoriser le passage d'un **flux électrique**.



Dans la bibliothèque des composants chercher un **bouton poussoir**, une **led**, un **Arduino Uno**, deux **résistances** et réaliser le câblage ci-dessous.



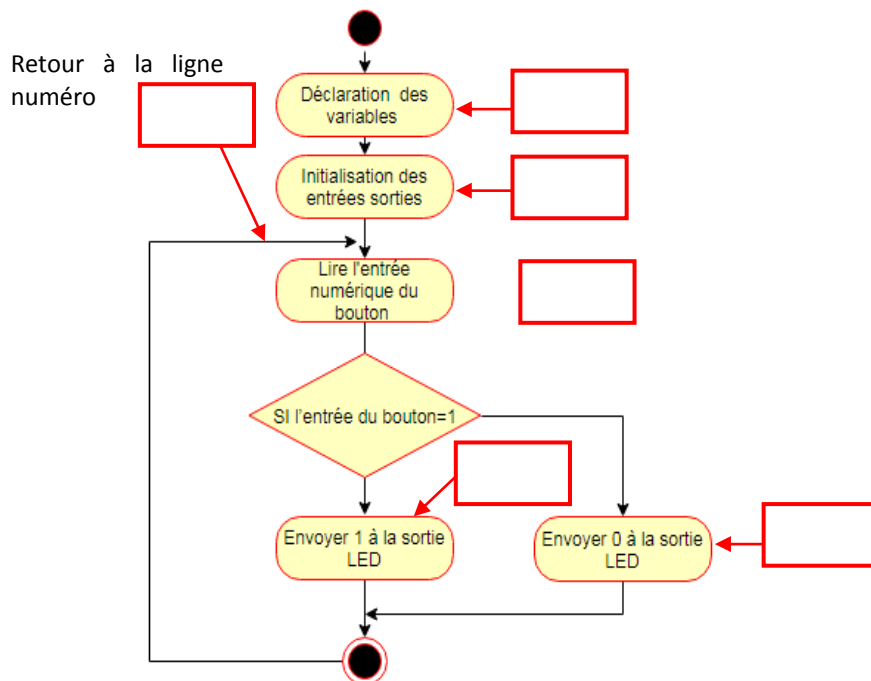
Ensuite, mettre le programme méthode1 dans la zone programmation texte et appuyer sur le bouton .Même chose pour le programme méthode 2.

Méthode 1	Méthode 2
<pre> 1 int bouton=2; 2 int led=12; 3 int boutonvalue; 4 void setup(){ 5 pinMode(bouton, INPUT); 6 pinMode(led, OUTPUT); 7 } 8 void loop(){ 9 boutonvalue = digitalRead(bouton); 10 if(boutonvalue == HIGH) {digitalWrite(led, HIGH); } else {digitalWrite(led, LOW); } } </pre>	<pre> int bouton=2; int led=12; int boutonvalue; void setup(){ pinMode(bouton, INPUT); pinMode(led, OUTPUT); } void loop(){ if(entreenumerique(bouton) == HIGH) {digitalWrite(led, HIGH); } else {digitalWrite(led, LOW); } } //----- int entreenumerique (int entree) {boutonvalue = digitalRead(entree); return boutonvalue ;} </pre>

Q10 : Cherche sur le site Arduino la définition de la ligne de code **digitalRead**.

Q11 : Comparer la méthode 1 et la méthode2 en utilisant les mots clés : **variable de retour**, **fonction**.

Q12 : Mettre les numéros des lignes de code de la **méthode1** à l'endroit correspondant sur l'organigramme suivant. **Une case peut prendre plusieurs numéros !.**

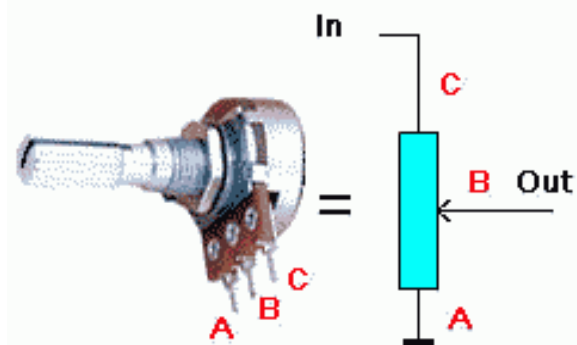


5.2) Le potentiomètre.

Le potentiomètre c'est le fameux **bouton "volume"** que l'on rencontre toujours dans les équipements audio.

C'est une **résistance variable** à trois bornes, dont une est reliée à un curseur se déplaçant sur une piste résistante terminée par les deux autres bornes.

En électronique sa valeur analogique de tension varie de 0 à 5V



Dans la bibliothèque des composants, chercher le potentiomètre ensuite réaliser le schéma ci-dessous et suivre les instructions 1,2,3,4,5,6.

1 .REALISER LE CABLAGE

4 .Démarrer la simulation

2 .COPIER COLLER LE PROGRAMME 1 ICI

3 .OUVRIR LE MONITEUR SERIE ICI

5 .Tourner l'axe du potentiomètre

6 .regarder les valeurs sur le moniteur série.

Borne 2

Borne 1

Curseur

```

int sensorValue = 0;

void setup()
{
  pinMode(A0, INPUT);
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  // read the value from the sensor
  sensorValue = analogRead(A0);
  Serial.println(sensorValue);
}

```

Moniteur série

Moniteur série

859
859
859
859
859
859
859
85

Programme1	Programme2
<pre> int sensorValue = 0; void setup() { pinMode(A0, INPUT); pinMode(13, OUTPUT); Serial.begin(9600); } void loop() { sensorValue = analogRead(A0); Serial.println(sensorValue); } </pre>	<pre> int sensorValue = 0; void setup() { pinMode(A0, INPUT); pinMode(13, OUTPUT); Serial.begin(9600); } void loop() { // read the value from the sensor sensorValue = analogRead(A0); float NewValue= map(sensorValue, 0, 1023, 0, 5); Serial.println(NewValue); } </pre>

Q13 : l'entrée du potentiomètre, est-elle une entrée analogique ou numérique ?

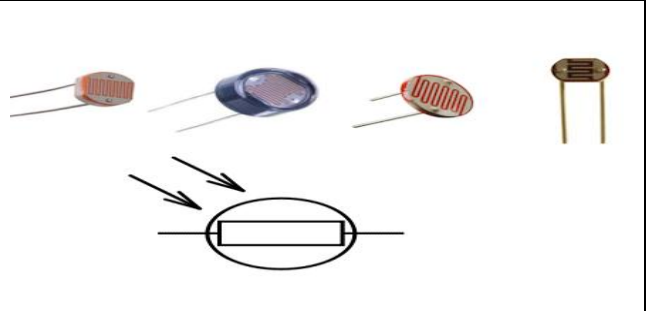
Q14 : Chercher sur le site Arduino la fonction de la ligne de code **Sérialbegin(9600)**.

Q15 : Tournez l'axe du potentiomètre et relever la valeur minimum et maximum du programme1 et du programme 2.

Q16 : que fait la ligne de code **float NewValue= map(sensorValue, 0, 1023, 0, 5);**

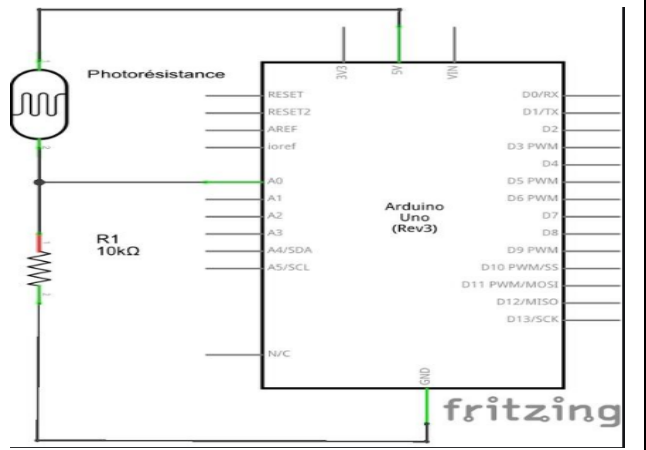
5.3) Capteur de lumière ou résistance photoélectrique (LDR))

La **résistance photoélectrique** ou **LDR** (light dépendent resistor) peut être utilisée comme un **capteur lumière**. En effet sa résistance varie en fonction de la luminosité.



Montée en pont avec une autre résistance fixe, elle permet de convertir une variation d'intensité lumineuse en variation de **tension**.

Cette variation de tension est **analogique** (variation progressive en 0 et 5V). Il faut donc utiliser une **entrée analogique** (A0 à A5) de la carte Arduino enfin d'exploiter ce capteur.



Chercher une photorésistance, une résistance, ensuite réaliser le schéma ci-dessous et suivre les instructions 1,2,3,4,5.

1 .REALISER LE CABLAGE

3 .Démarrer la simulation

2 .COPIER COLLER LE PROGRAMME 1 ICI

5 . VARIER LA LUMIERE ici

```

1 int sensorValue = 0;
2
3 void setup()
4 {
5   pinMode(A0, INPUT);
6   pinMode(13, OUTPUT);
7   Serial.begin(9600);
8 }
9
10 void loop()
11 {
12   sensorValue = analogRead(A0);
13   Serial.println(sensorValue);
14 }
15
16

```

4 .afficher les valeurs sur le moniteur série.

Programme1	Programme2
<pre>int sensorValue = 0; void setup() { pinMode(A0, INPUT); Serial.begin(9600); } void loop() { sensorValue = analogRead(A0); Serial.println(sensorValue); }</pre>	<pre>int sensorValue; void setup(){ pinMode(A0, INPUT); Serial.begin(9600); } void loop(){ Serial.println(entreeanalogique (A0)); //----- int entreeanalogique (char entree) { sensorValue = analogRead(entree); return sensorValue ;}</pre>

Q17 : Faire varier l'intensité lumineuse et relever la valeur minimum et maximum du programme1.

Q18 : Utiliser la fonction **map(valeur, min1, max1,min2 , max2);** pour afficher un minimum de 0 et un maximum de 5.

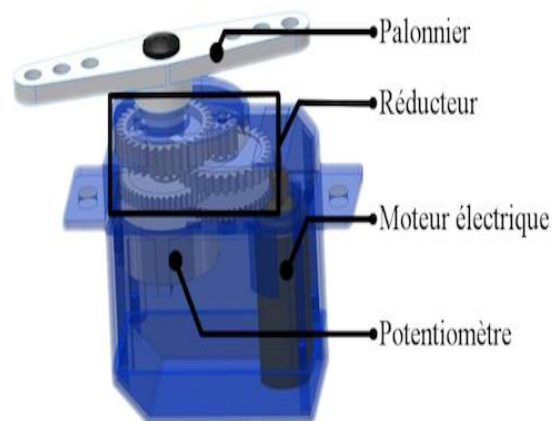
Q19: Lancer le programme 2 et donner la valeur d'entrée et de sortie de la fonction **entreeanalogique**.

5.3) Le Servomoteur

Un **servomoteur** est un système permettant de **mettre en rotation** un objet.

Comparé à un simple moteur, le servomoteur est capable de se positionner à un **angle précis** et **Maintenir cette position**, même si un effort est exercé sur le palonnier.

Remarque : Les servomoteurs utilisés dans cette activité sont commandés directement en angle en degrés (0 à 180°).



Chercher dans la bibliothèque des composants Arduino et choisir Servo.



5 .Démarrer la simulation

4 .COPIER COLLER LE PROGRAMME 1 ICI

```

1
2 #include <Servo.h>
3 int angle;
4 Servo monservo;
5
6 void setup()
7 {
8   Serial.begin(9600);
9   monservo.attach(9);
10 }
11
12 void loop()
13 {
14   if(Serial.available()>0){
15     angle=Serial.parseInt();
16     monservo.write(angle);
17     Serial.println("POSITION DU SERVO:");Serial.println(angle);
18     delay(500);
19   }
20 }

```

6 .OUVRIR LE MONITEUR SERIE ICI

Moniteur série

6 .Mettre l'angle de rotation du servomoteur (de 0 à 180) ICI et appuyer sur entrée

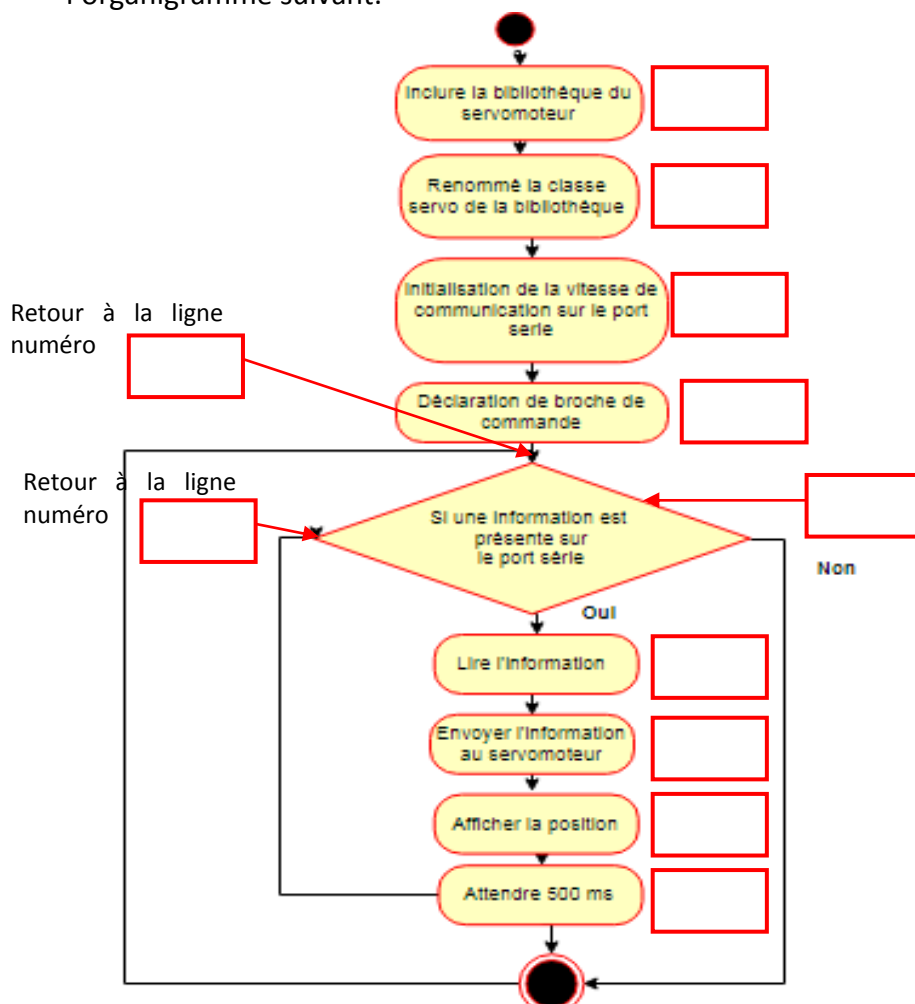


Programme1	Programme2
<pre> 1 #include <Servo.h> 2 Servo monservo; 3 int angle; 4 5 void setup() 6 { 7 Serial.begin(9600); 8 monservo.attach(9); 9 } 10 11 void loop() 12 { 13 if(Serial.available()>0){ 14 angle=Serial.parseInt(); 15 monservo.write(angle); 16 Serial.println("POSITION DU SERVO:");Serial.println(angle); 17 delay(500); 18 } 19 } </pre>	<pre> #include <Servo.h> Servo monservo; void setup(){ Serial.begin(9600); monservo.attach(9); } void loop(){ if(Serial.available()>0){commandeservo(lireleport());} delay(500); } //----- int lireleport(){ int angle=Serial.parseInt(); Serial.println("VALEUR RECUE:");Serial.println(angle); return angle; } //----- void commandeservo(int entree){ monservo.write(entree); Serial.println("POSITION:");Serial.println(entree); } </pre>

Q20: Chercher sur le site Arduino les fonctions des la lignes de code suivante :

```
#include <Servo.h>,  
monservo.attach(9);  
monservo.write(angle);  
Serial.available().
```

Q21: Mettre le numéro des lignes de code du programme1 à l'endroit correspondant sur l'organigramme suivant.



Q22: Lancer le programme 2 et donner la valeur d'entrée et de sortie de la fonction **lireleport** et La fonction **commandeservo** .

6. Réalisation d'un suiveur solaire (Tracker solaire).

Ajouter un capteur de lumières, une résistance, sur le schéma précédent du servomoteur et suivre les étapes 1,2,3,4,5 ci-dessous.

1 .Réaliser le câblage

3 .Démarrer la simulation

2 . COPIER COLLER LE PROGRAMME ICI

```

1 #include <Servo.h>
2 Servo monservo;
3 int angle;
4
5 int sensorValue = 0;
6
7 void setup()
8 {
9   pinMode(A0, INPUT);
10  Serial.begin(9600);
11  monservo.attach(9);
12 }
13
14 void loop()
15 {
16
17   sensorValue = analogRead(A0);
18   Serial.println(sensorValue);
19   if (sensorValue < 940 && angle > 0 ){angle=angle-5; Serial.println(angle);}
20   if (sensorValue > 940 && angle < 180 ){angle=angle+5; Serial.println(angle);}
21 }
22
23

```

5 . Cliquer sur le capteur et faire varier la lumière

4 . OUVRIR LE MONITOR SERIE ICI

PROGRAMME

```

#include <Servo.h>
Servo monservo;
int angle;

int sensorValue = 0;

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);
  monservo.attach(9);
}

void loop()
{

  sensorValue = analogRead(A0);
  Serial.println(sensorValue);

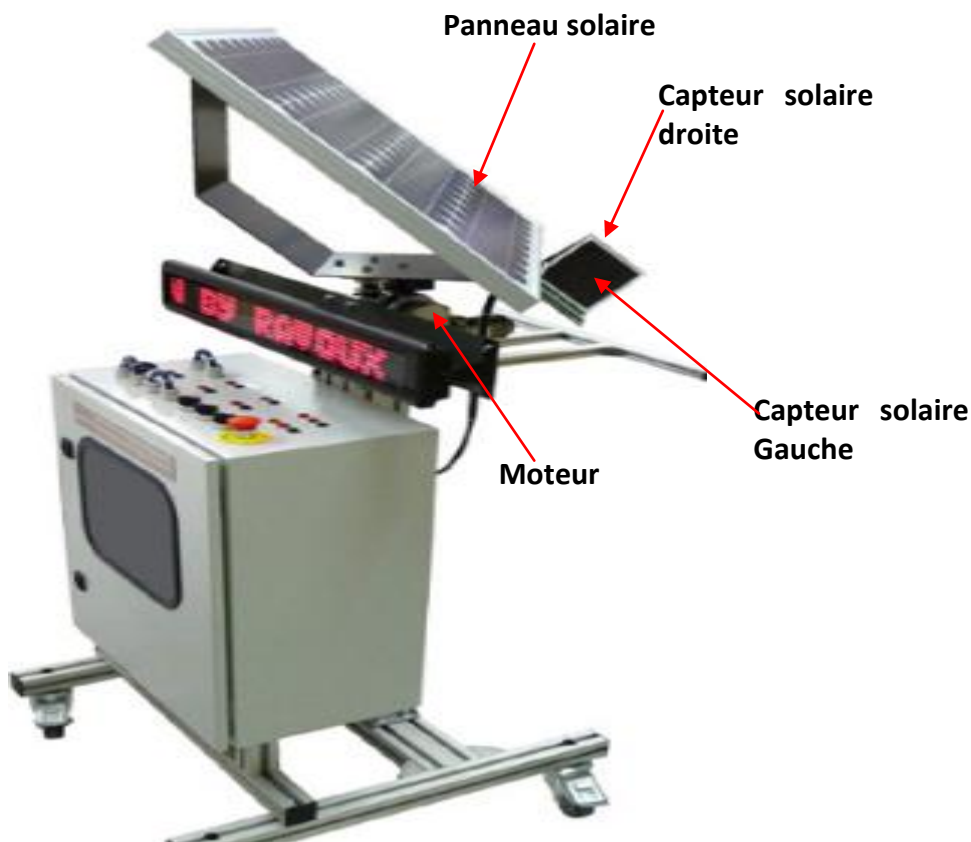
  if (sensorValue < 940 && angle > 0 ){angle=angle-5; Serial.println("ARRIERE");monservo.write(angle); }
  if (sensorValue > 940 && angle < 180 ){angle=angle+5; Serial.println("AVANT");monservo.write(angle);}
}

```

Q22: A partir de quelle valeur le servomoteur commence à bouger ?.

Q23: Quelle est la valeur de l'incrément ?.

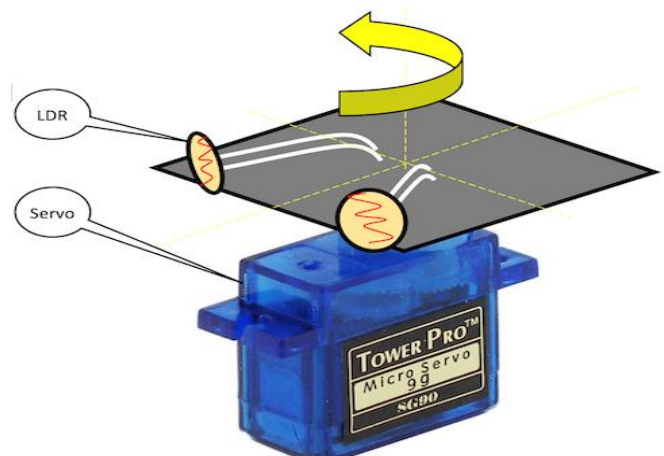
On souhaite maintenant réaliser le suiveur solaire du SOLEOTECH qui permet aux panneaux photovoltaïques de s'orienter face au soleil tout au long de la journée afin de maximiser le chargement de la batterie.

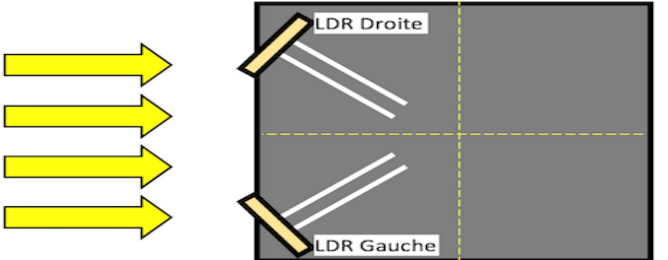
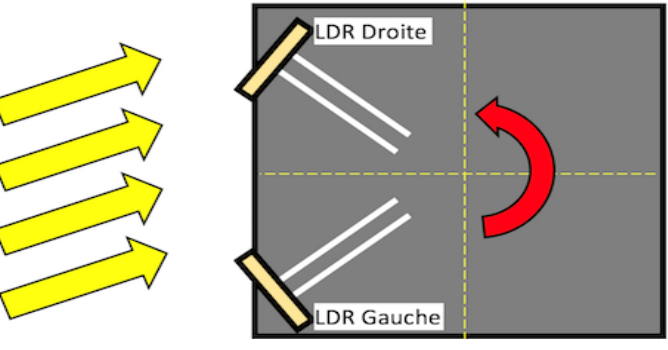
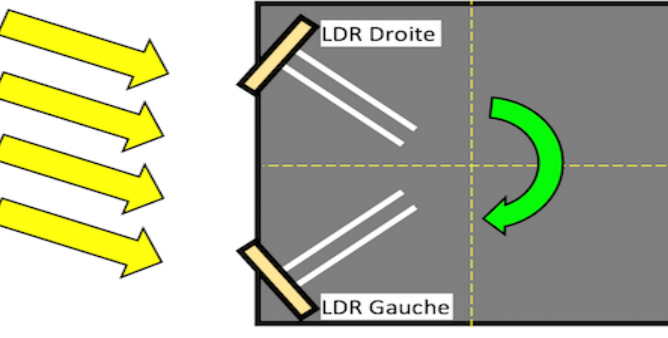


Pour cela, nous allons utiliser :

- 1 **servomoteur**
- 2 **LDR** inclinées par rapport à la direction centrale. LDR seront soudées sur une platine solidaire du palonnier du servo-moteur.

Le principe général est de **mesurer l'intensité lumineuse atteignant chacune des LDR**, qui est dépendante de **l'angle entre la LDR et les rayons lumineux**.



<p>1er cas : Le dispositif pointe vers la lumière.</p> <p>Les LDR droite et gauche reçoivent la même quantité de luminosité puisque leur angle avec les rayons est le même.</p> <p>Tout va bien, aucun mouvement du dispositif.</p>	
<p>2ème cas : Le dispositif pointe dans une direction inclinée par rapport aux rayons. Ici il ya un décalage vers la droite.</p> <p>La LDR gauche reçoit plus de lumière que la LDR droite (car elle est moins perpendiculaire par rapport aux rayons).</p> <p>Le dispositif tente de rattrapper le décalage : le servo tourne dans le sens rouge.</p>	
<p>3ème cas : Le dispositif pointe dans une direction inclinée par rapport aux rayons. Ici il ya un décalage vers la gauche.</p> <p>La LDR droite reçoit plus de lumière que la LDR gauche.</p> <p>Le dispositif tente de rattrapper le décalage : le servo tourne dans le sens vert.</p>	

Travail demandé :

- Réaliser la simulation avec TINKERCAD (câblage, programme, simulation).
- Réaliser le prototype sur breadboard (câblage, test, mise au point)
- Réaliser le prototype sur platine (soudage, test).

On donne l'algorithme de principe :

