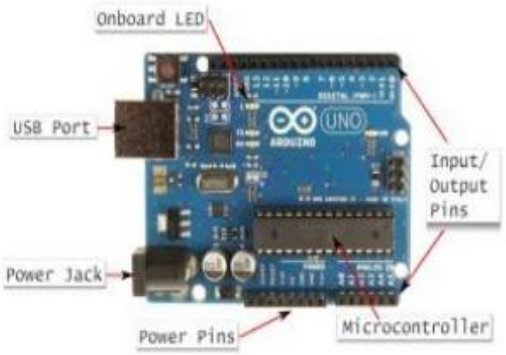
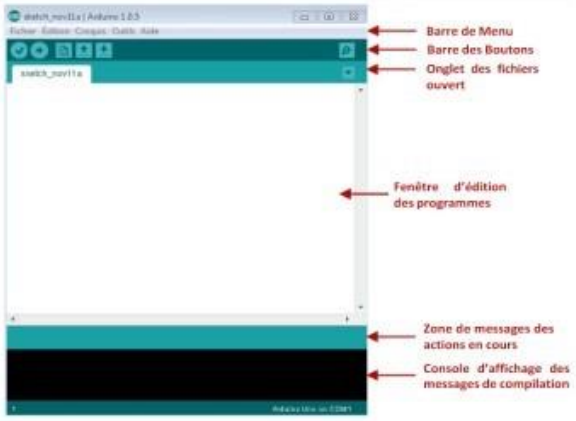
1. Présentation



CARTE

Arduino est une plateforme matérielle et logicielle de développement d'applications embarquées.

Elle se compose d'une carte électronique basée autour d'un microcontrôleur (ATMEL AVR) comportant un certain nombre d'entrées et de sorties (les broches ou pins analogiques et numériques) permettant la connexion de capteurs, ou d'actionneurs.



Logiciel ou IDE :

Le logiciel de programmation des modules Arduino est une application Java, libre servant d'éditeur de code et de compilateur, et qui peut transférer le firmware et le programme au travers de la liaison USB.

Le langage de programmation utilisé est un mélange de C et de C++, restreint et adapté aux possibilités de la carte.

Modules « shield »

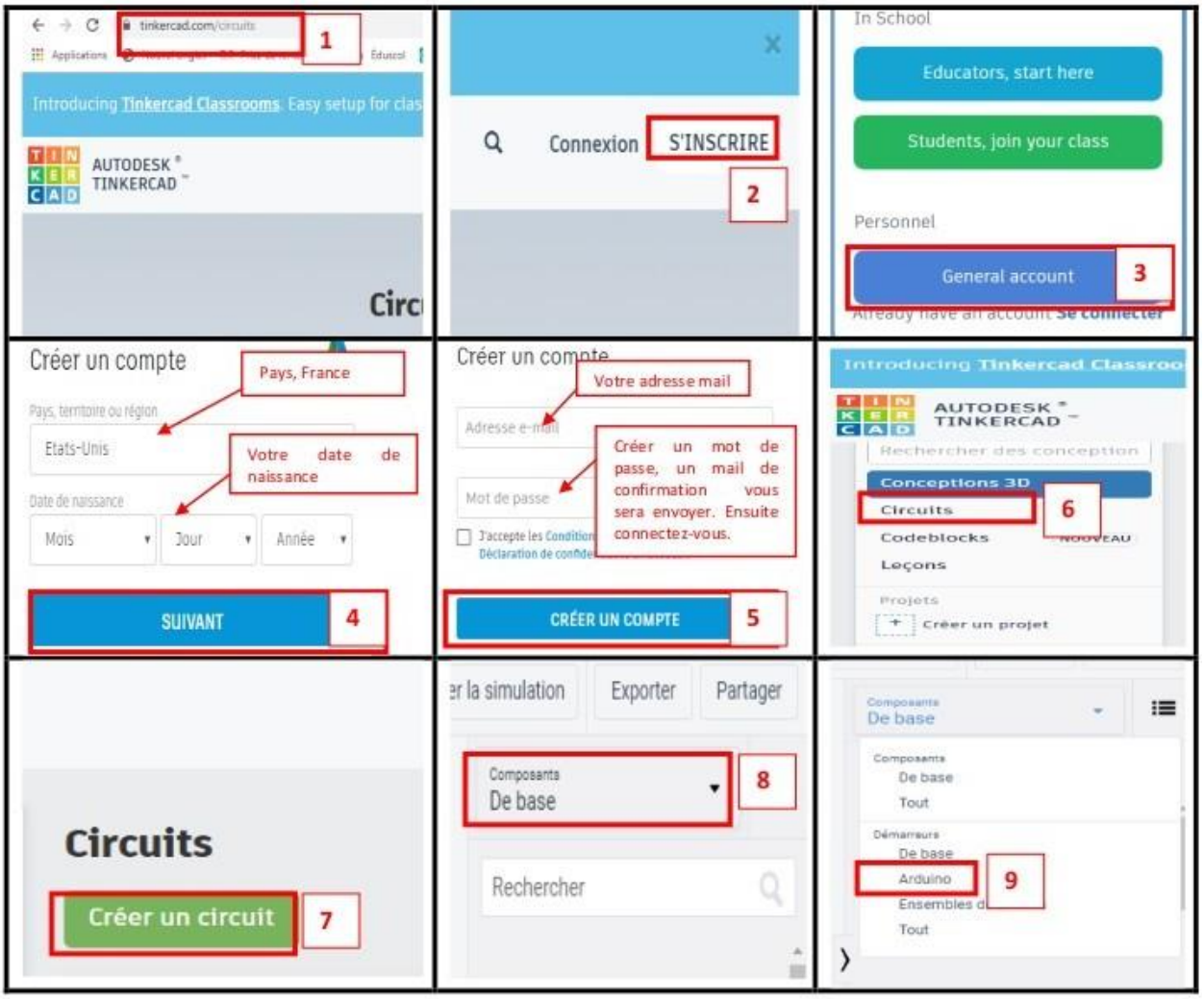
Cartes supplémentaires se connectant sur le module Arduino pour augmenter les possibilités et les capacités de la carte.

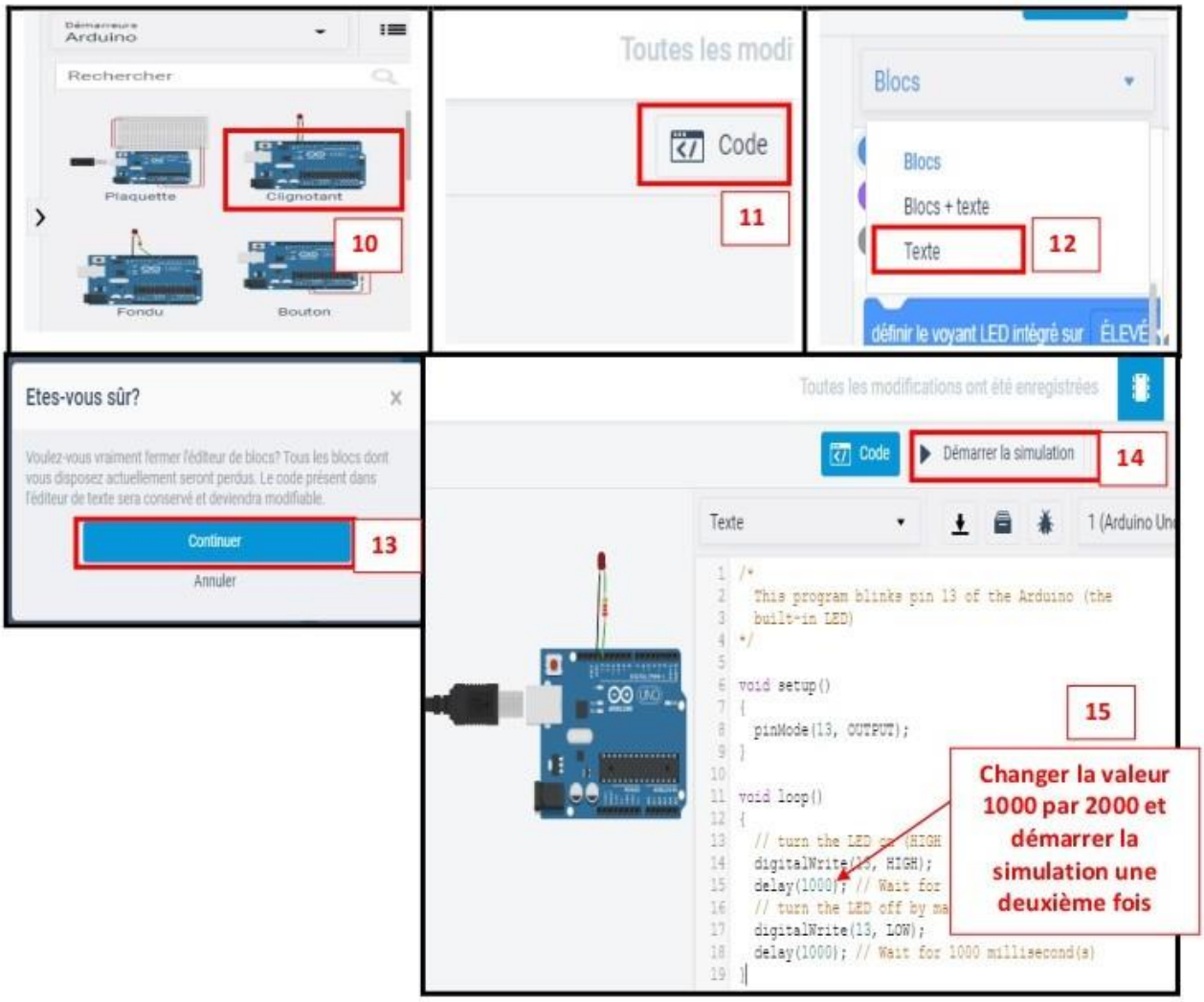
Exp : afficheur graphique couleur, interface Ethernet, GPS, WIFI, GSM, Carte de puissance moteur.....

2. Utilisation d'un simulateur

Cette partie à 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.

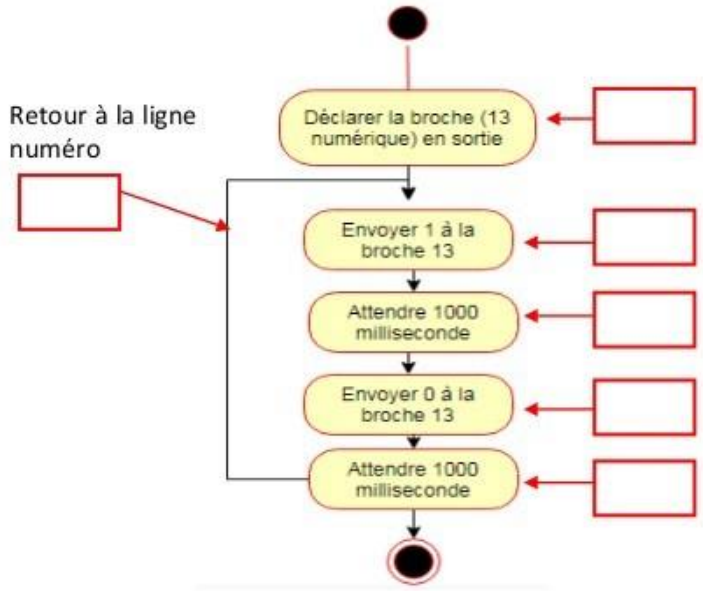




Q3 : Chercher sur le site arduino la signification des lignes de suivantes du code.
digitalWrite :
analogWrite :
pinMode :
void setup() :
void loop() :

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.



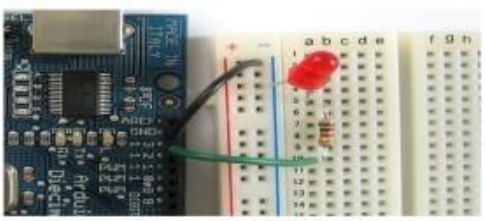
Q6 : remplacer le programme existant par le programme numéro 1 ensuite le programme numéro 2. Comparer en utilisant les mots clés suivants : **VARIABLE, SOUS PROGRAMME.**

Programme 1	Programme 2

int maled=13;
 	
void setup(){//----------
pinMode(maled, OUTPUT);
}
 	
void loop(){ //----------
digitalWrite(maled, HIGH);
delay(1000);
 	
digitalWrite(maled, LOW);
delay(1000);
}
	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);
	}

3. Câblage d'un circuit "virtuel"

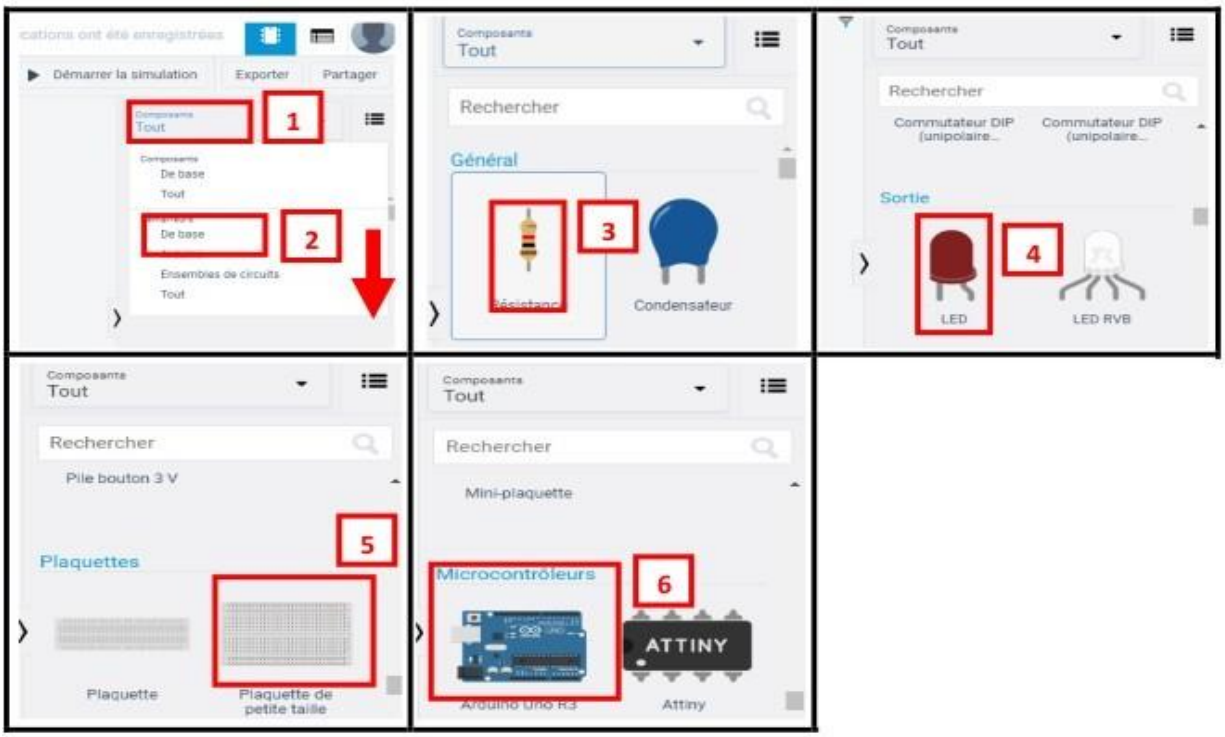
On désire que la carte Arduino fasse clignoter deux **leds branchées sur deux sorties digitales avec une plaque d'essai.**



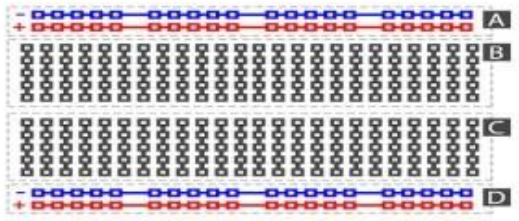
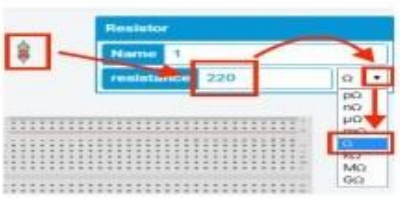
Il faut tout d'abord **ajouter les composants**. Deux **leds**, deux résistances ,Une breadboard(plaque d'essai), un arduino.

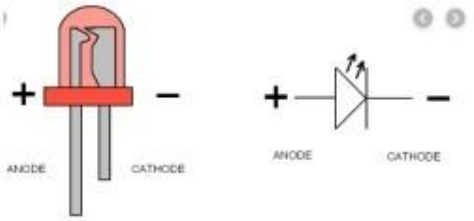
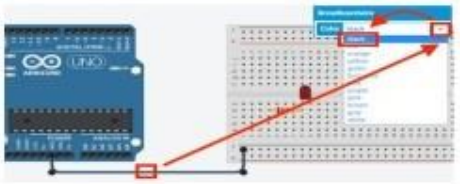
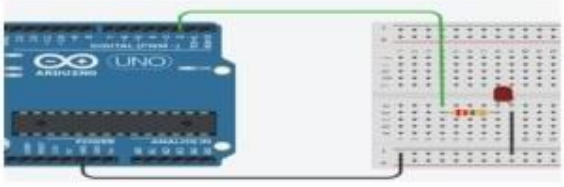
Cliquer sur composants et naviguer dans la liste de composants grâce à l'**ascenseur sur la droite .**

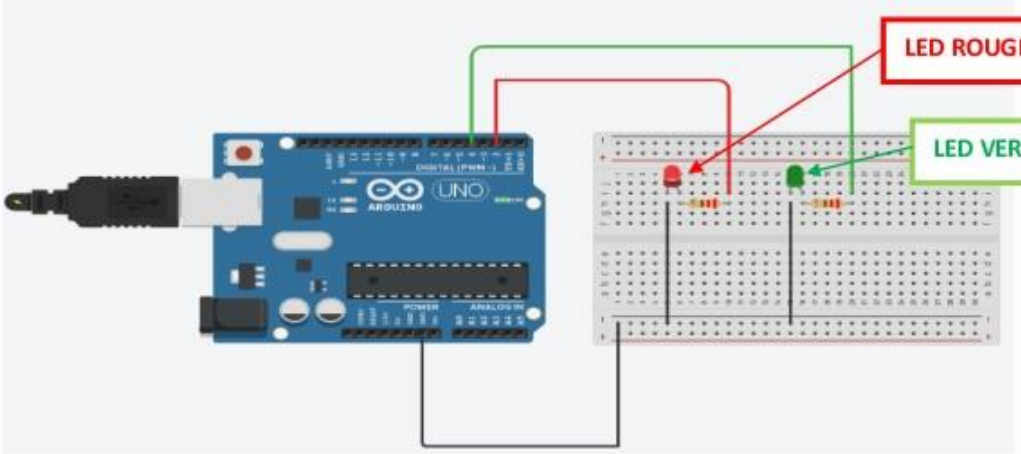
Effectuer un "*Clic and Drop*" pour déposer les composants dans la zone de conception :



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"** :	

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



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

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);
 	
}
	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);
	}

Q7 : Réaliser l'algorigramme du programme méthode 1.

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

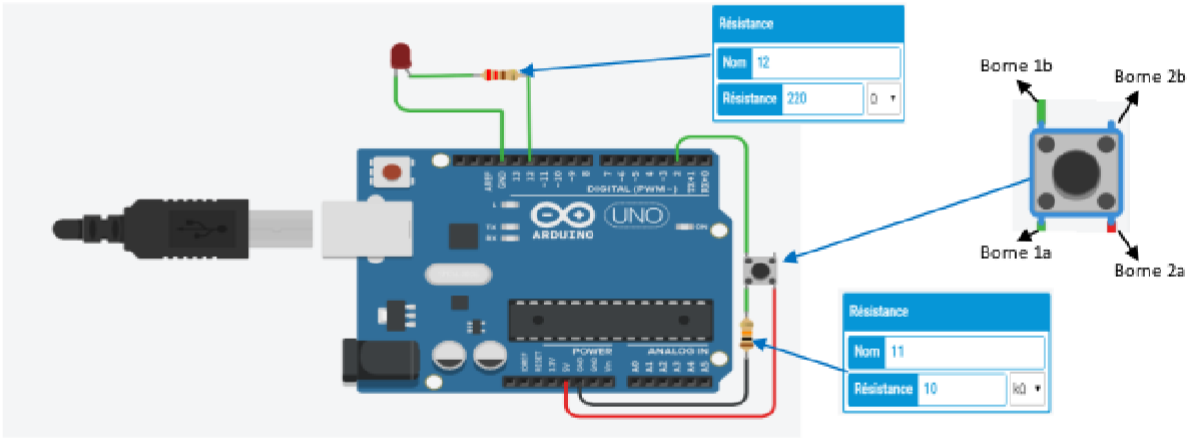
Q9 : Rajouter une troisième led de couleur bleu et faire clignoter les trois une après l'autre avec les deux méthodes.

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.



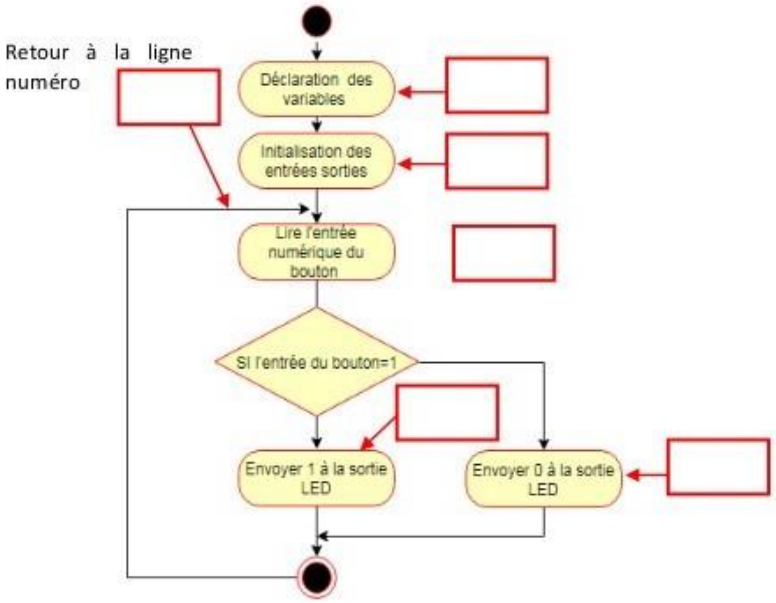
Ensuite, mettre le programme méthode1 dans la zone programmation texte et appuyer sur le bouton .Même chose pour le programme méthode 2.

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

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 !.

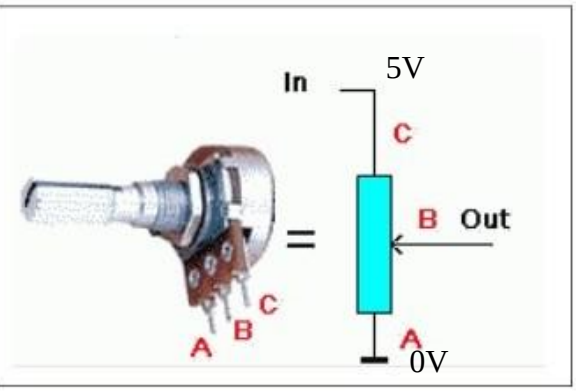


5.2) Le potentiomètre.

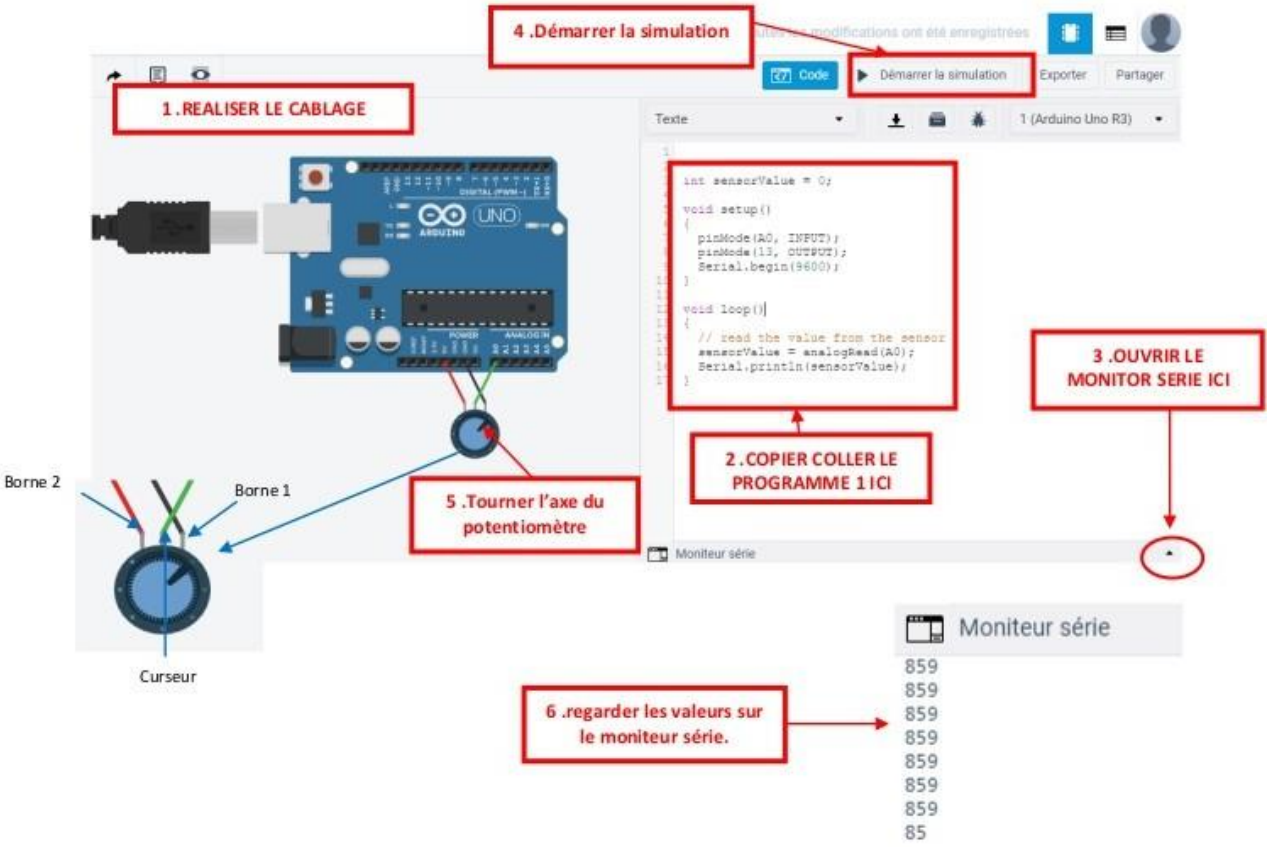
Le potentiomètre c'est le fameux **bouton "volume"** que l'on rencontre toujours dans les équipements audio.

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.

Dans le cas de notre carte arduino la tension le sera comprise entre 0V et 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.



Programme1	Programme2

int sensorValue = 0;
 	
void setup()
{
pinMode(A0, INPUT);
pinMode(13, OUTPUT);
Serial.begin(9600);
}
 	
void loop()
{
 	
sensorValue = analogRead(A0);
Serial.println(sensorValue);
}
	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);
	}

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

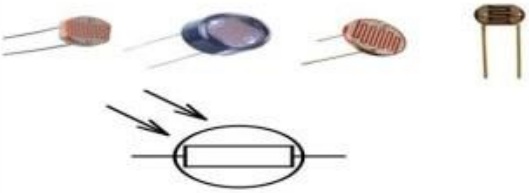
Q14 : Chercher sur le site Arduino la fonction de la ligne de code **Sérialbegin(9600).**

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);**
Chercher sur le site Arduino le rôle de la fonction map

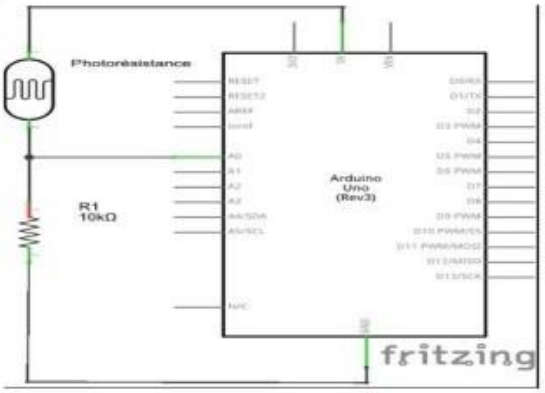
5.3) Capteur de lumière ou résistance photoélectrique (LDR))

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

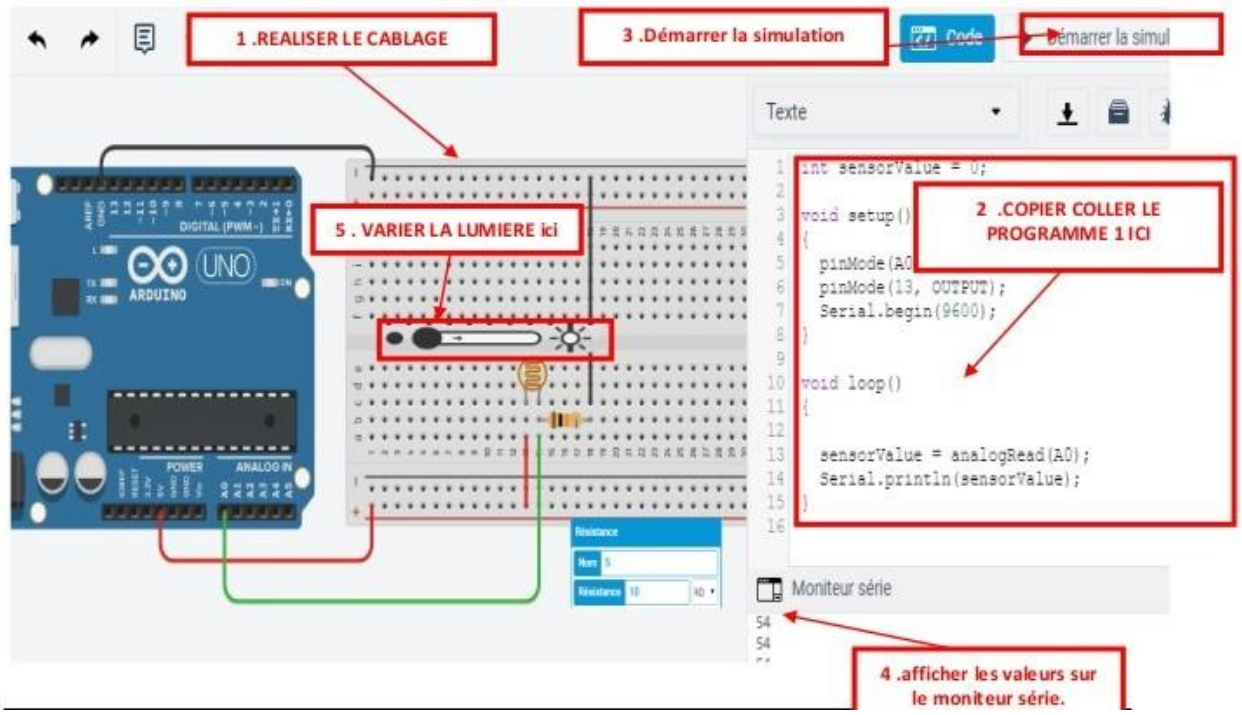


Montée en pont avec une autre résistance fixe, elle permet de convertir une variation d'intensité lumineuse en variation de **tension**.

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.



Programme1	Programme2

int sensorValue = 0;
 	
void setup()
{
pinMode(A0, INPUT);
Serial.begin(9600);
}
 	
void loop()
{
 	
sensorValue = analogRead(A0);
Serial.println(sensorValue);
}
	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 ;}

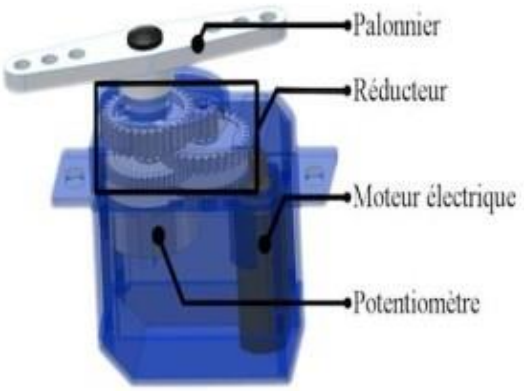
Q17 : Faire varier l'intensité lumineuse et et relever la valeur minimum et maximum du programme1.
Q18 : Utiliser la fonction **map(valeur, min1, max1,min2 , max2);** pour afficher un minimum de 0 et un maximum de 5.
Q19: Lancer le programme 2 et donner la valeur d'entrée et de sortie de la fonction **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 commandé directement en angle en degrès (0 à 180°).



Chercher dans la bibliothèque des composants Arduino et choisir Servo.

	Programme1	Programme2
1	#include <Servo.h> Servo monservo; int angle;	#include <Servo.h> Servo monservo;
2		
3		

4	void setup()
	{
		void setup(){
		Serial.begin(9600);
		monservo.attach(9);
		}

5	Serial.begin(9600);	
6	monservo.attach(9);	

	}	void loop(){
		if(Serial.available()>0){commandeservo(lireleport());}
		delay(500);
		}

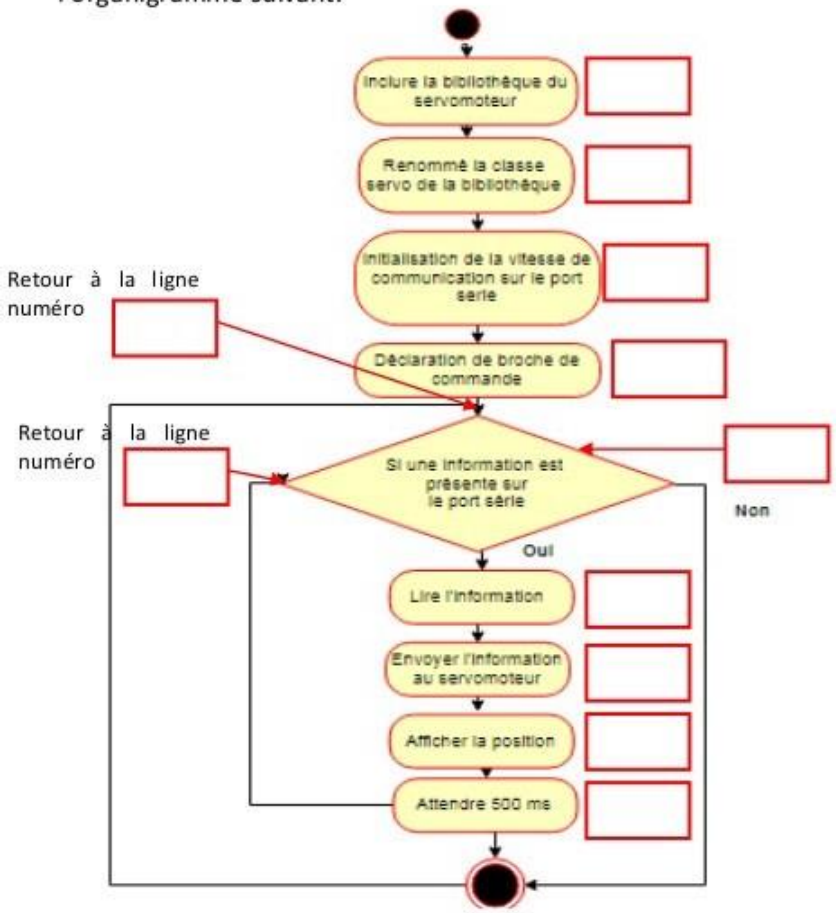
7	void loop() {	//----------
8	if(Serial.available()>0){	int lireleport(){ int angle=Serial.parseInt();
9	angle=Serial.parseInt();	Serial.println("VALEUR RECUE:");Serial.println(angle);
10	monservo.write(angle);	return angle;
11	Serial.println("POSITION DU SERVO:");Serial.println(angle);	}
12	delay(500);	//----------

	}	void commandeservo(int entree){
		monservo.write(entree);
		Serial.println("POSITION:");Serial.println(entree);
		}

	}	

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 .**