|Programming with Arduino|

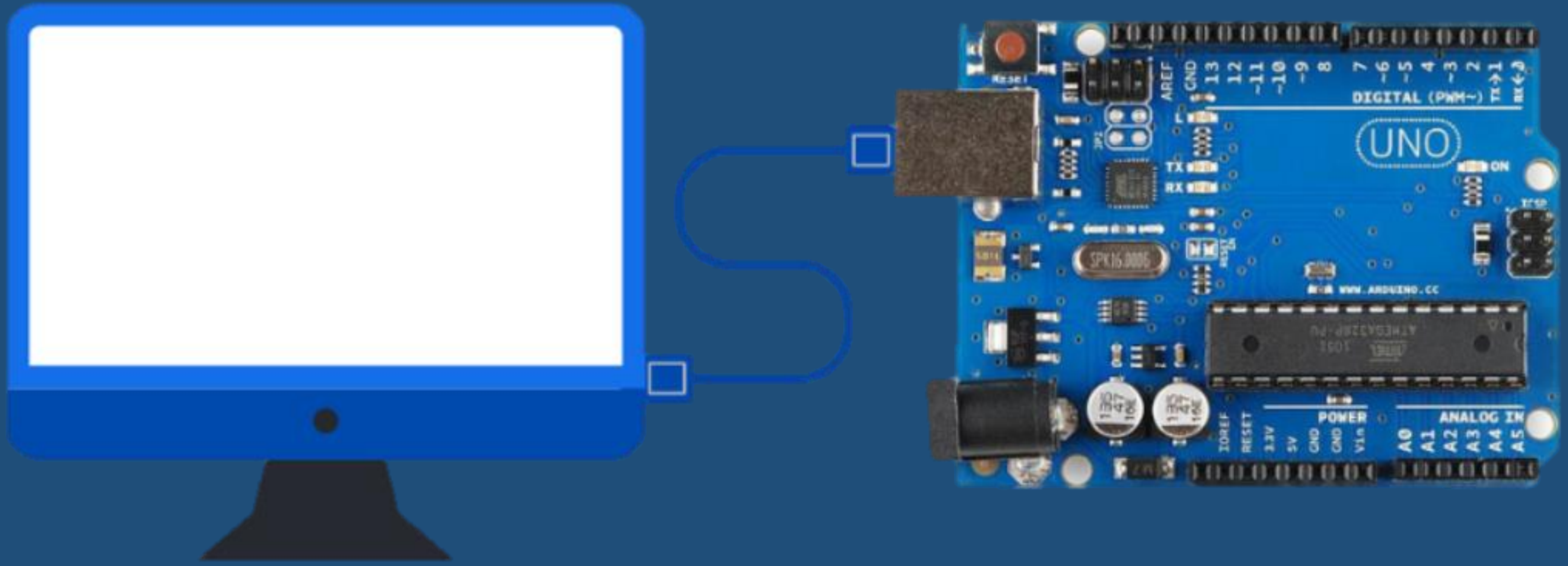


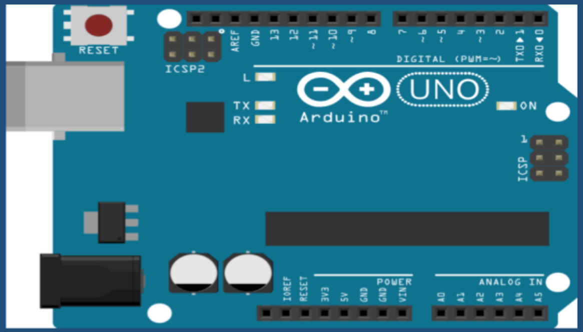
Table of Contents

- Introduction to Arduino
- Basic Electronics
- Arduino Development Environment
- Arduino Programming
- Arduino Examples

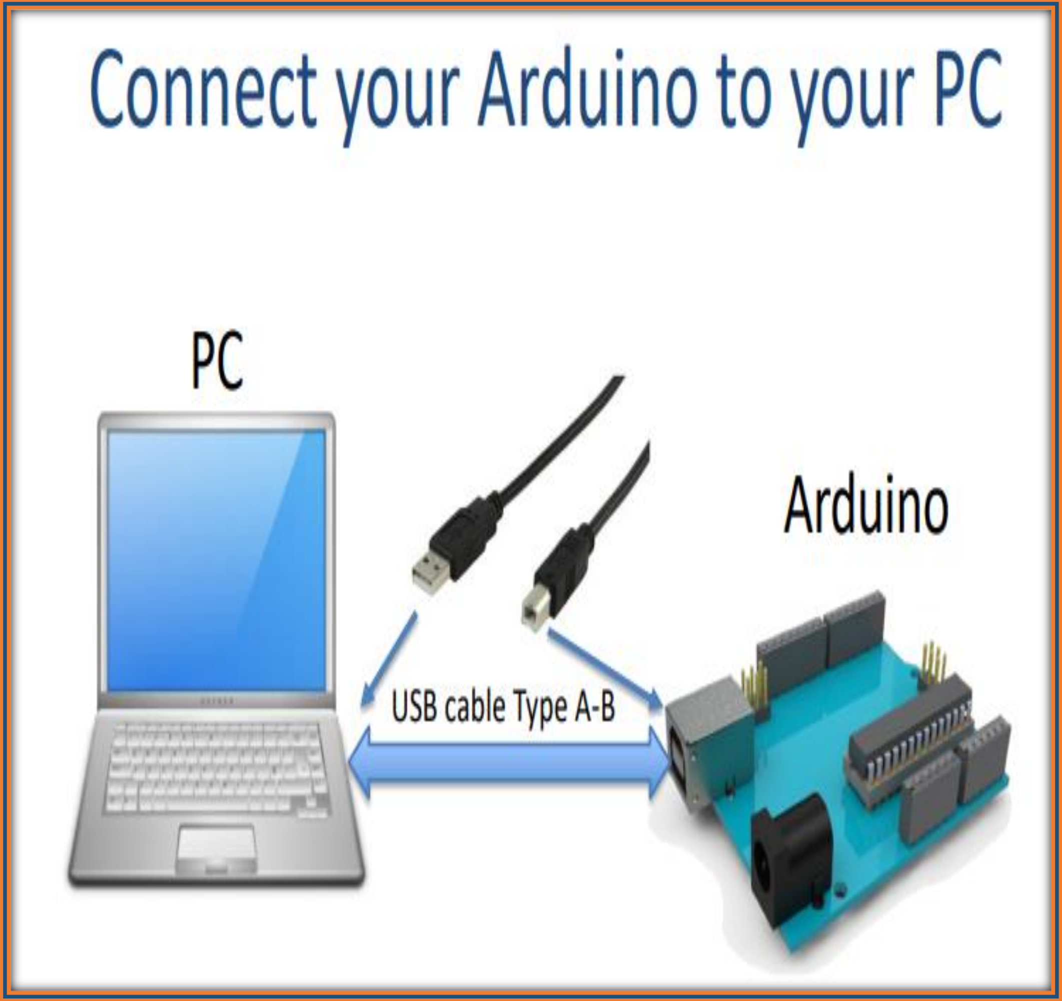
(1. Electrical Circuit , 2. Blinking LED, 3. Switch, 4. Potentiometer,
5. Temperature, 6. Light Sensor, 7. Thermistor)

What do you need?

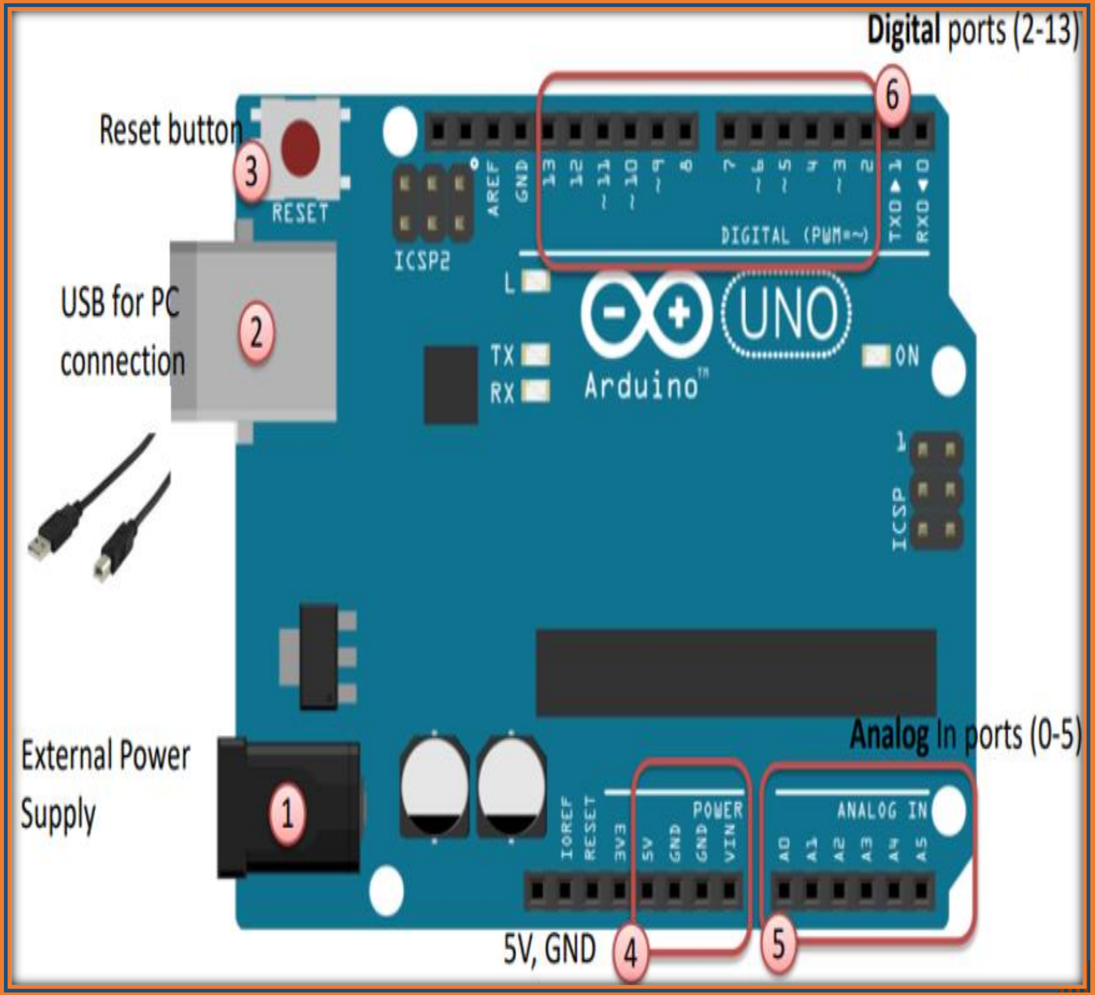
- To get started you need the following:
- PC (Windows, Mac, Linux)
- **Arduino UNO** (~200 NOK) or a Starter Kit (~800 NOK)
- Software (free)
- Electrical components (wires, resistors, etc.)



Arduino UNO Overview



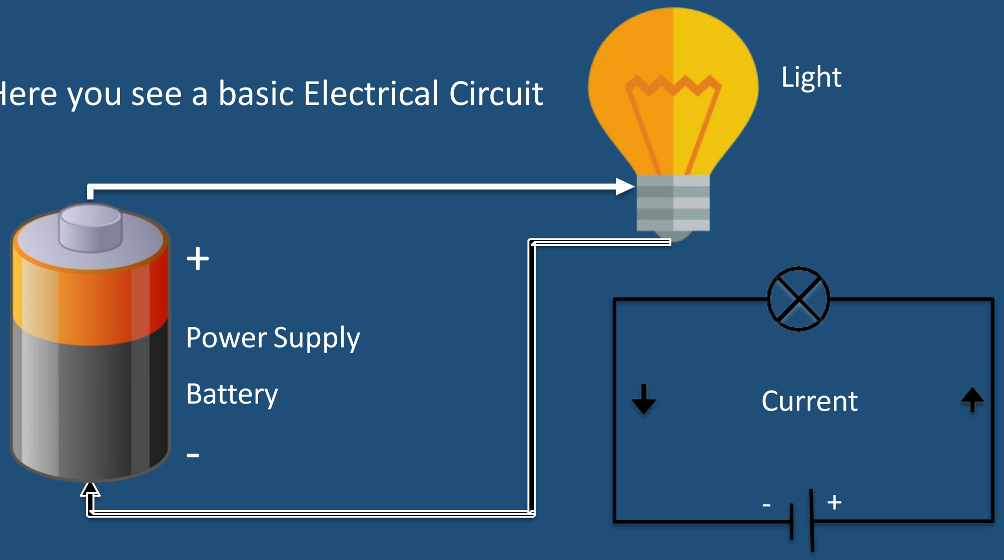
Connect your Arduino to your PC



Electronics Foundation

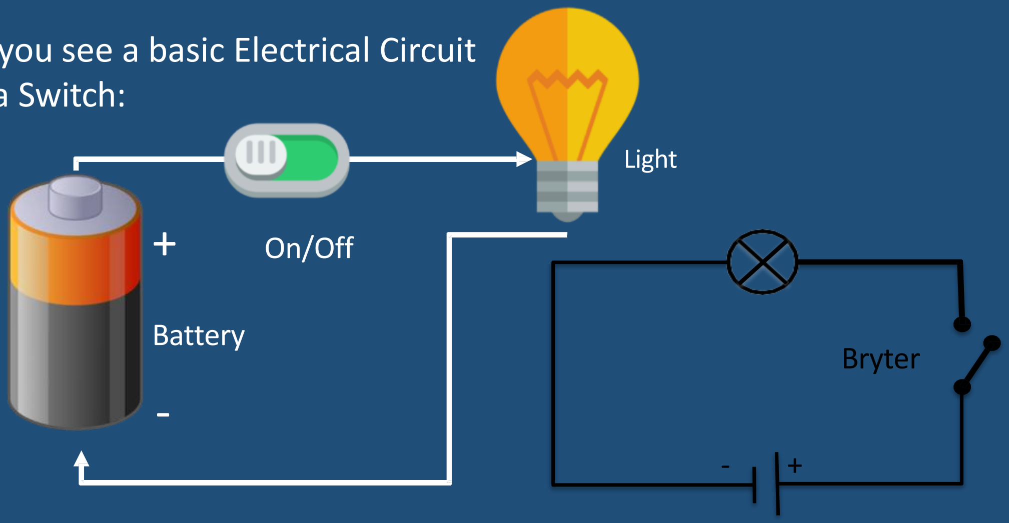
Electrical Circuit

Here you see a basic Electrical Circuit



Electrical Circuit with a Switch

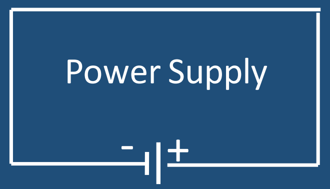
Here you see a basic Electrical Circuit with a Switch:



Short Circuit

- We must never connect positive and negative side to a power source without having an electrical component in between.
- If you do, it is called a short circuit.
- For example, if you short circuit a battery, the battery will get very hot and the battery will run out very quickly.
- Some batteries may also start to burn.
- When it starts to smoke from electrical components, it happens because it has become too hot.
- In most cases, it means that the component is broken

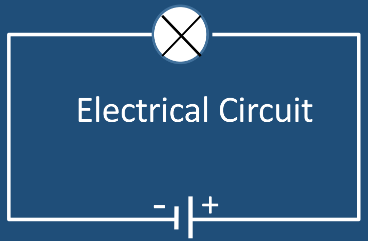
Short Circuit!!

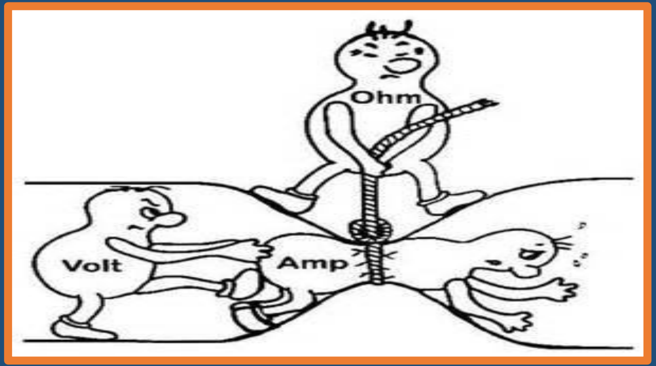


This is Ohms Law:

V= I R

V – Voltage $[V]$
R – Resistance $[\Omega]$
I – Current [A]





Ohms Law

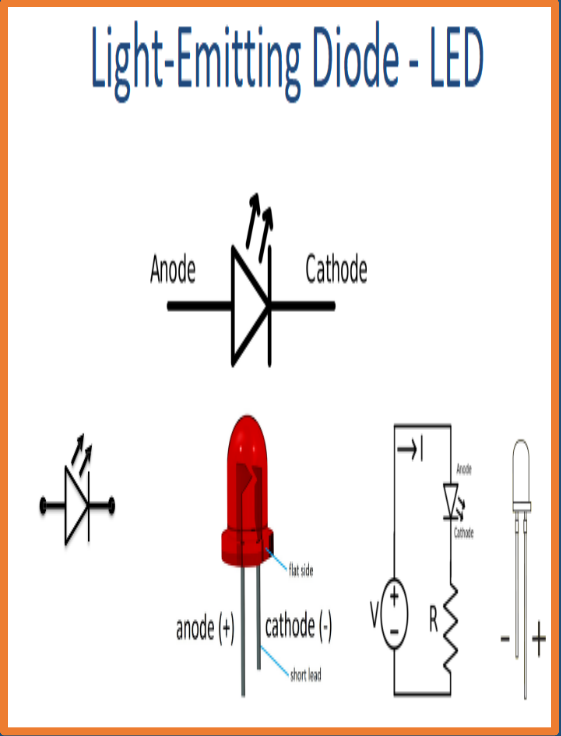
- A semiconductor device that emits light when an electric current flows through it.

Operation:

- When a voltage is applied across its leads, electrons recombine with holes, releasing energy in the form of photons (light).

Characteristics:

- **Forward Voltage**: Typically between 1.8 V to 3.3 V depending on the color and type.
- **Current Rating**: LEDs typically operate at currents between 10 mA and 30 mA.
- **Polarity**: LEDs have a positive terminal (anode) and a negative terminal (cathode). The longer leg is the anode.



Resistors

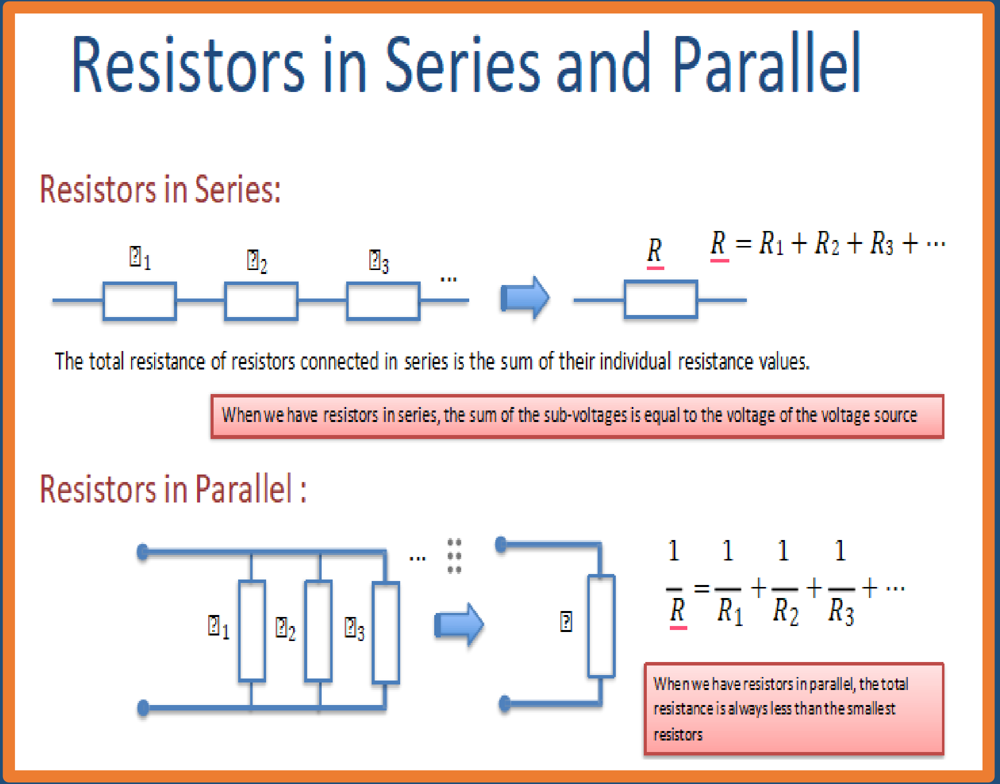
A resistor is a passive electronic component that limits or regulates the flow of electrical current in a circuit.

Unit of Resistance:

The unit of resistance is the **Ohm (Ω)**.

Types:

- **Fixed Resistors**: Provide a specific resistance value.
- **Variable Resistors (Potentiometers)**: Allow adjusting resistance within a range.
- **Applications**: Used to control current, divide voltages, or provide biasing for active components like transistors.



Resistors in Series and Parallel

Resistors in Series:

$$R = R_1 + R_2 + R_3 + \cdots$$

The total resistance of resistors connected in series is the sum of their individual resistance values.

When we have resistors in series, the sum of the sub-voltages is equal to the voltage of the voltage source

Resistors in Parallel :

$$\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \cdots$$

When we have resistors in parallel, the total resistance is always less than the smallest resistors

Kirchhoff’s Laws

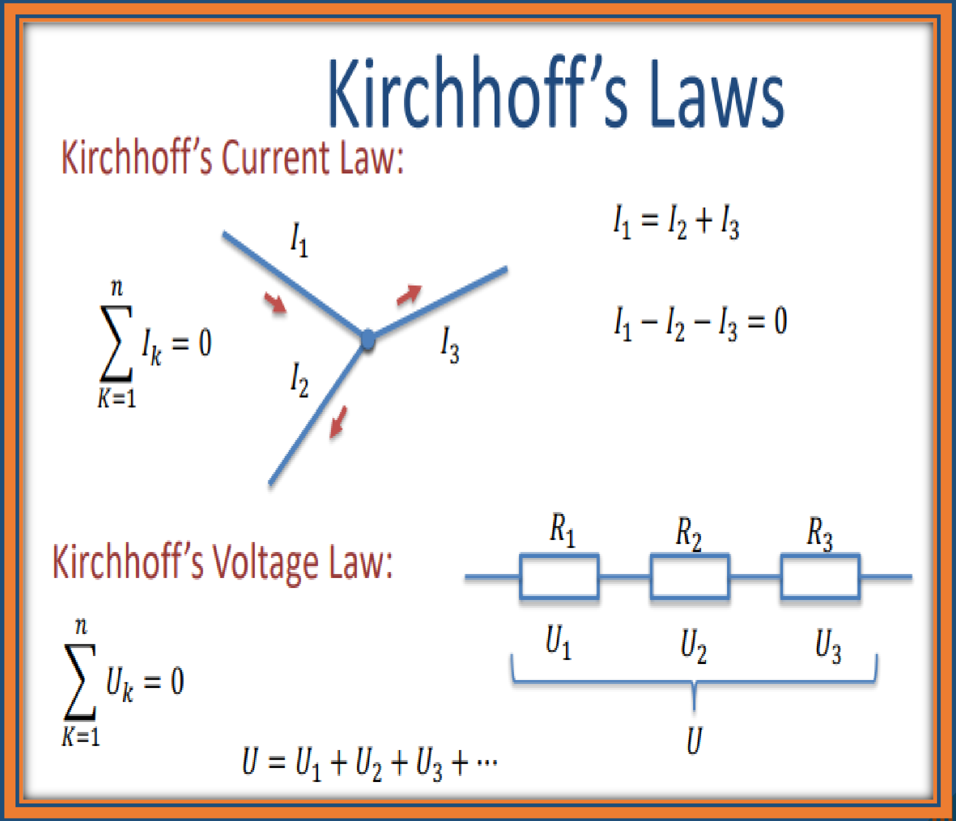
Kirchhoff’s Current Law (KCL):

The total current entering a junction in a circuit equals the total current leaving the junction.

Kirchhoff’s Voltage Law (KVL):

The total voltage around any closed loop in a circuit is equal to zero.

This means that the sum of all voltage drops in a closed loop must equal the sum of all voltage sources.



Kirchhoff’s Laws

Kirchhoff’s Current Law:

$$\sum_{K=1}^{n} I_k = 0$$

$$I_1 = I_2 + I_3$$

$$I_1 - I_2 - I_3 = 0$$

Kirchhoff’s Voltage Law:

$$\sum_{K=1}^{n} U_k = 0$$

$$U = U_1 + U_2 + U_3 + \cdots$$

Switch

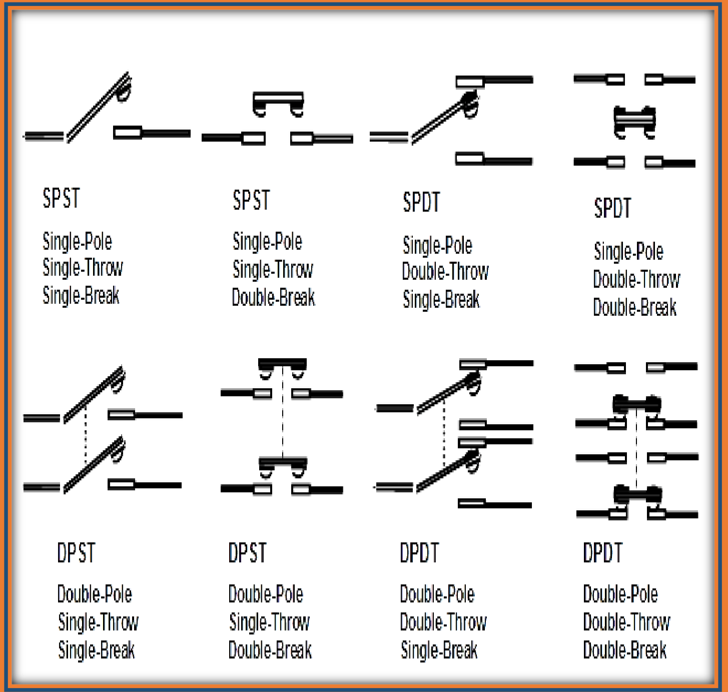
A switch is a device used to interrupt the flow of current in a circuit.

Types:

SPST (Single Pole Single Throw):
A simple on/off switch that controls a single circuit.

SPDT (Single Pole Double Throw):
A switch that connects one input to either of two outputs.

DPDT (Double Pole Double Throw):
Essentially two SPDT switches in one, allowing for control of two circuits simultaneously.



Applications:
Used for turning devices on or off, changing paths in a circuit, and user control.

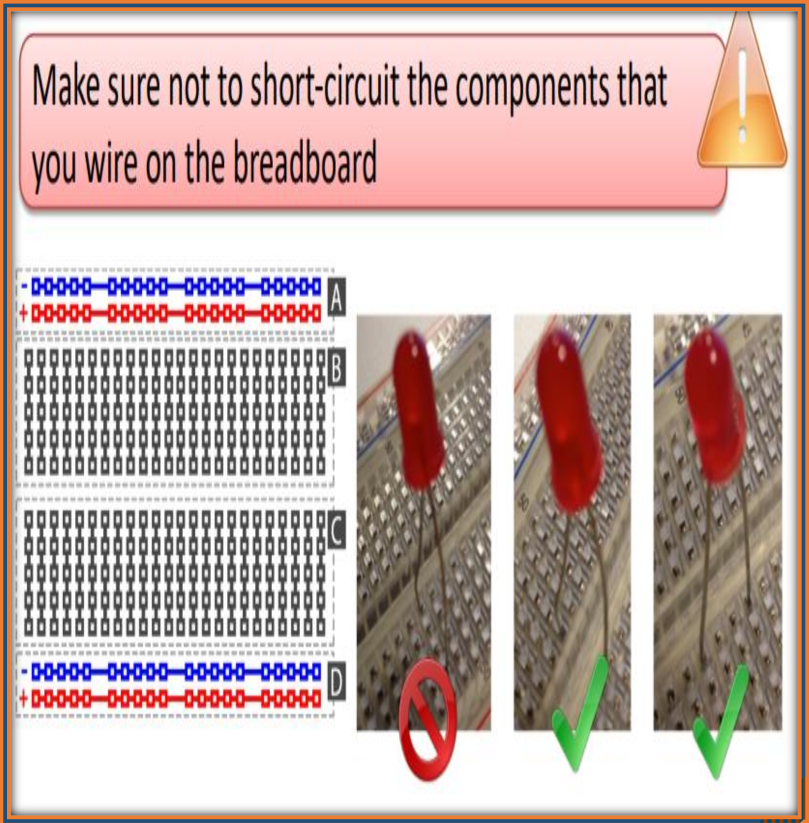
Breadboard

A breadboard is a rectangular board with rows of interconnected holes, used for prototyping electronic circuits without soldering.

Structure:
Power Rails: The outer columns are used for power distribution (marked with red and blue lines).

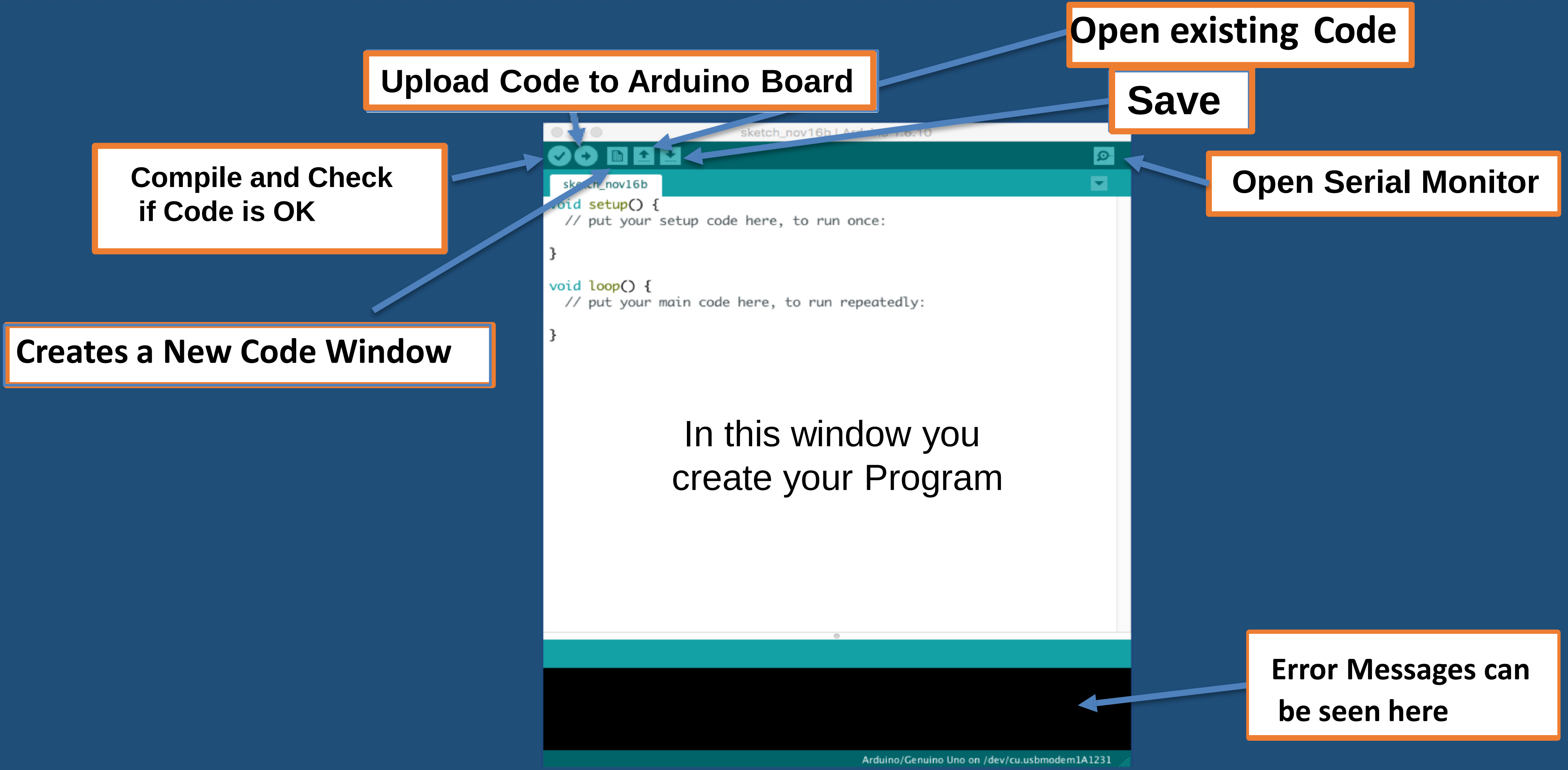
Terminal Strips:
The inner part is used for placing components, where each row of five holes is electrically connected.

Use: Breadboards allow for easy insertion and removal of components, ideal for testing and modifying circuits during the design process.



Make sure not to short-circuit the components that you wire on the breadboard

Arduino Software



Open existing Code

Upload Code to Arduino Board

Save

Compile and Check if Code is OK

Open Serial Monitor

Creates a New Code Window

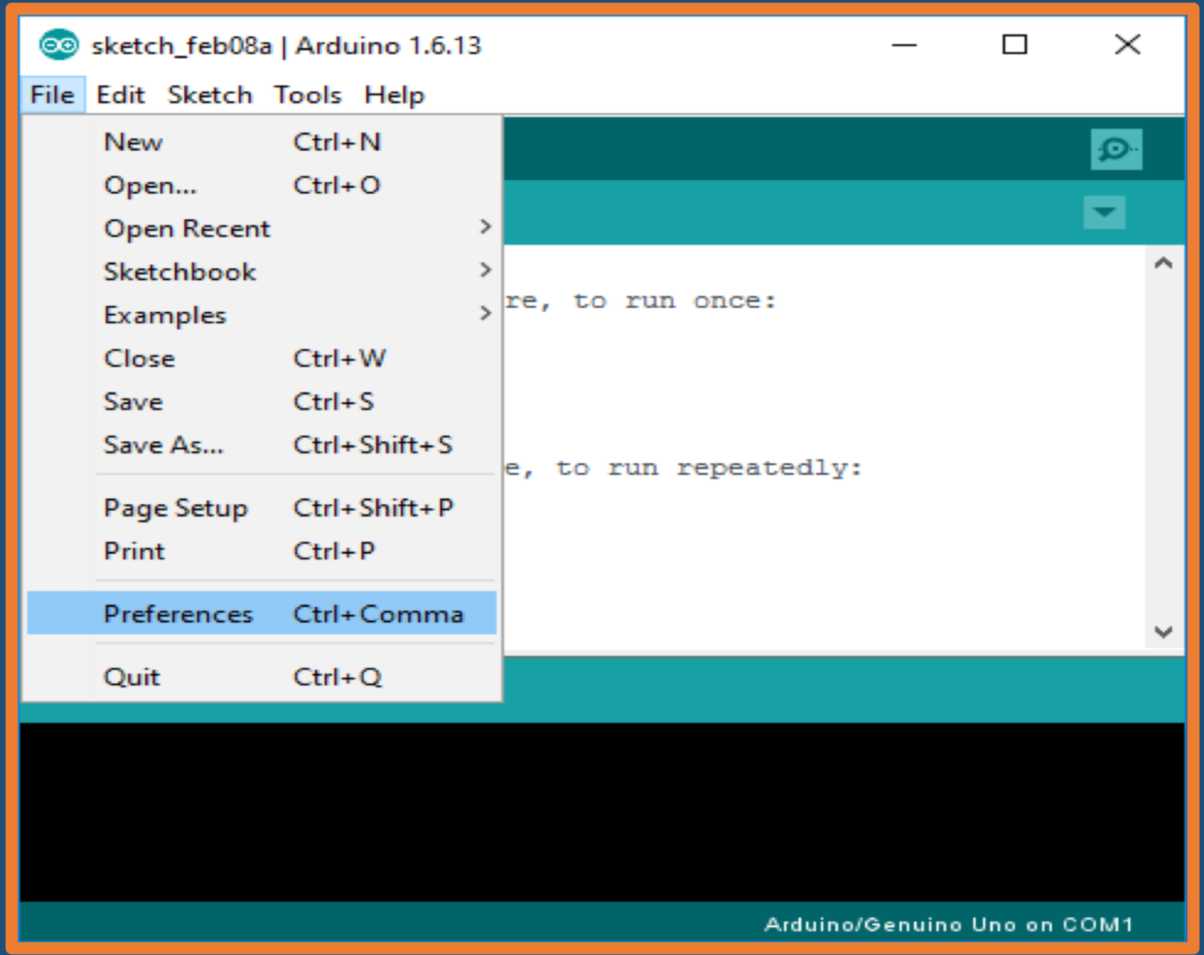
```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

In this window you create your Program

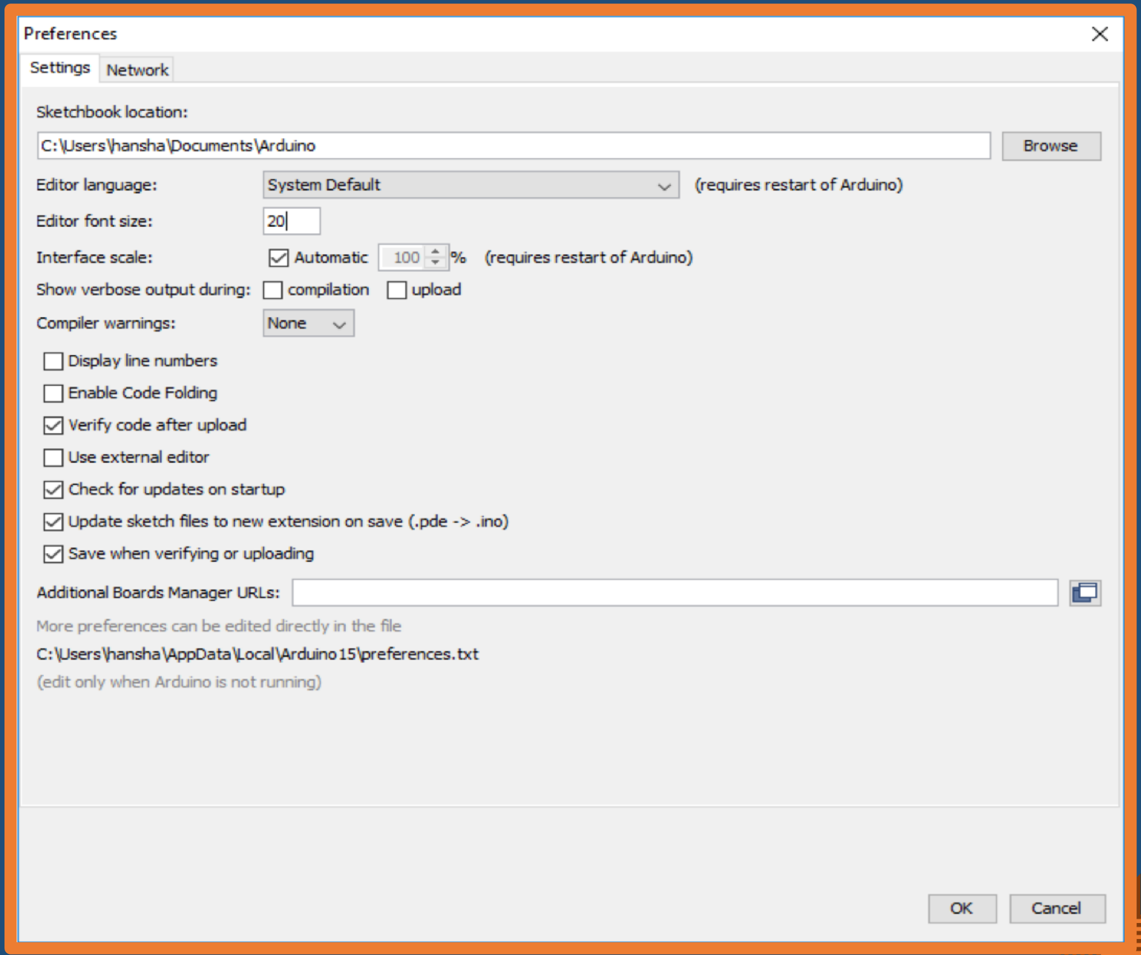
Error Messages can be seen here

Arduino/Genuino Uno on /dev/cu.usbmodem1A1231

Editor Preferences



```
re, to run once:

e, to run repeatedly:
```



Sketchbook location: C:\Users\hansha\Documents\Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 20

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Show verbose output during: □ compilation □ upload

Compiler warnings: None

□ Display line numbers

□ Enable Code Folding

☑ Verify code after upload

□ Use external editor

☑ Check for updates on startup

☑ Update sketch files to new extension on save (.pde -> .ino)

☑ Save when verifying or uploading

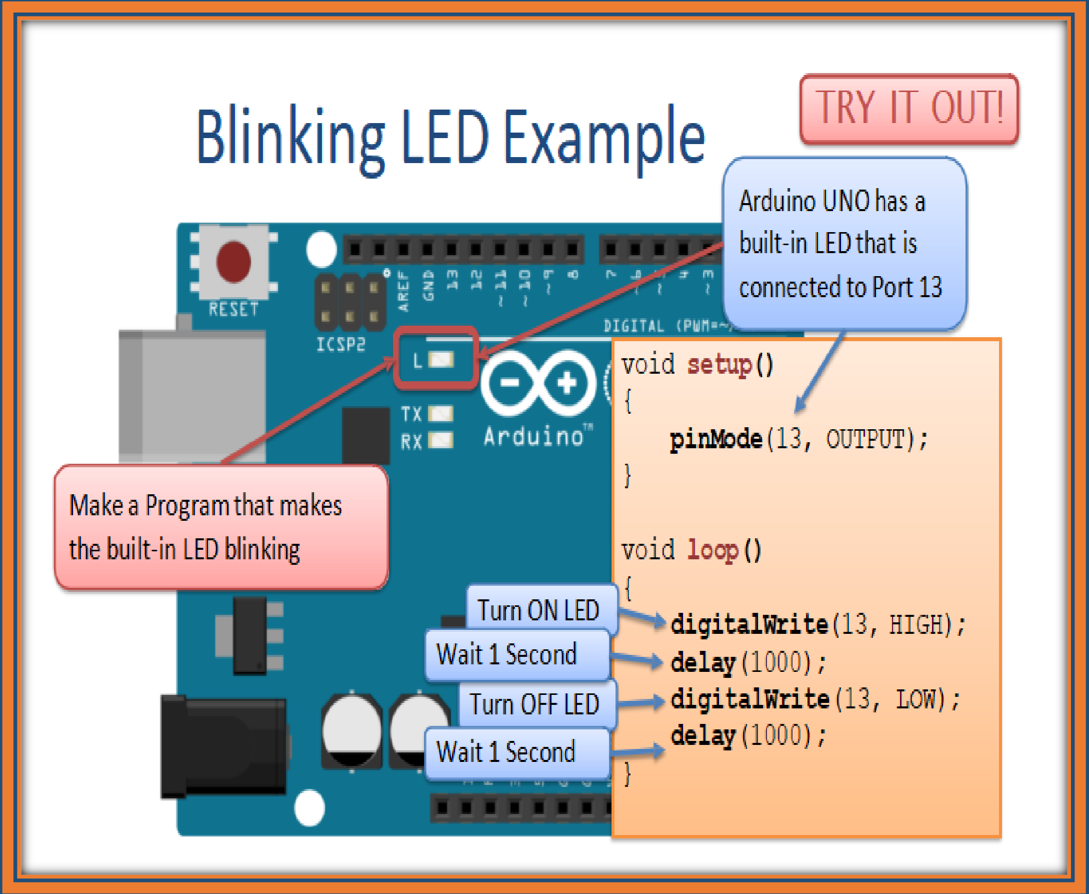
Additional Boards Manager URLs:

More preferences can be edited directly in the file

C:\Users\hansha\AppData\Local\Arduino15\preferences.txt

(edit only when Arduino is not running)

Blinking a LED



Blinking LED Example

TRY IT OUT!

Arduino UNO has a built-in LED that is connected to Port 13

Make a Program that makes the built-in LED blinking

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

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

Turn ON LED

Wait 1 Second

Turn OFF LED

Wait 1 Second

This Program makes the built-in LED blinking

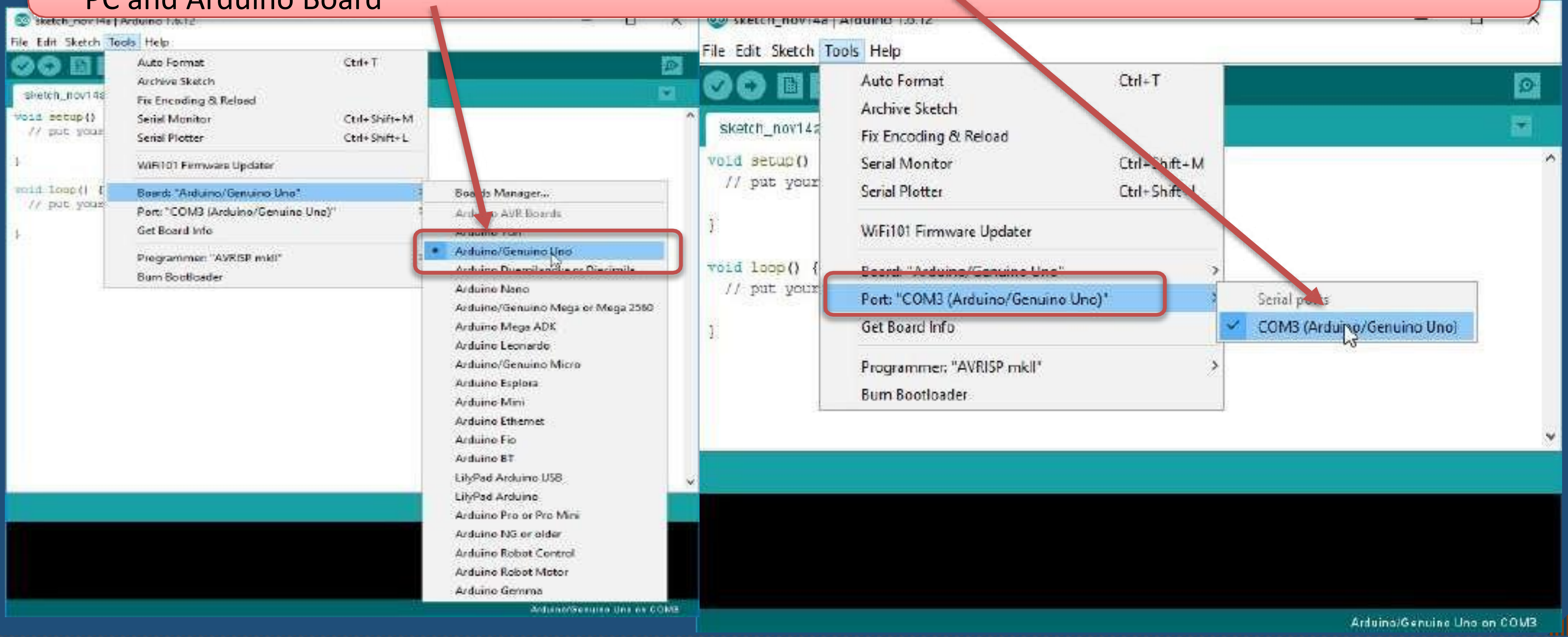
Blinking LED Example

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

Try to change from 1000 to 100 – What happens then?

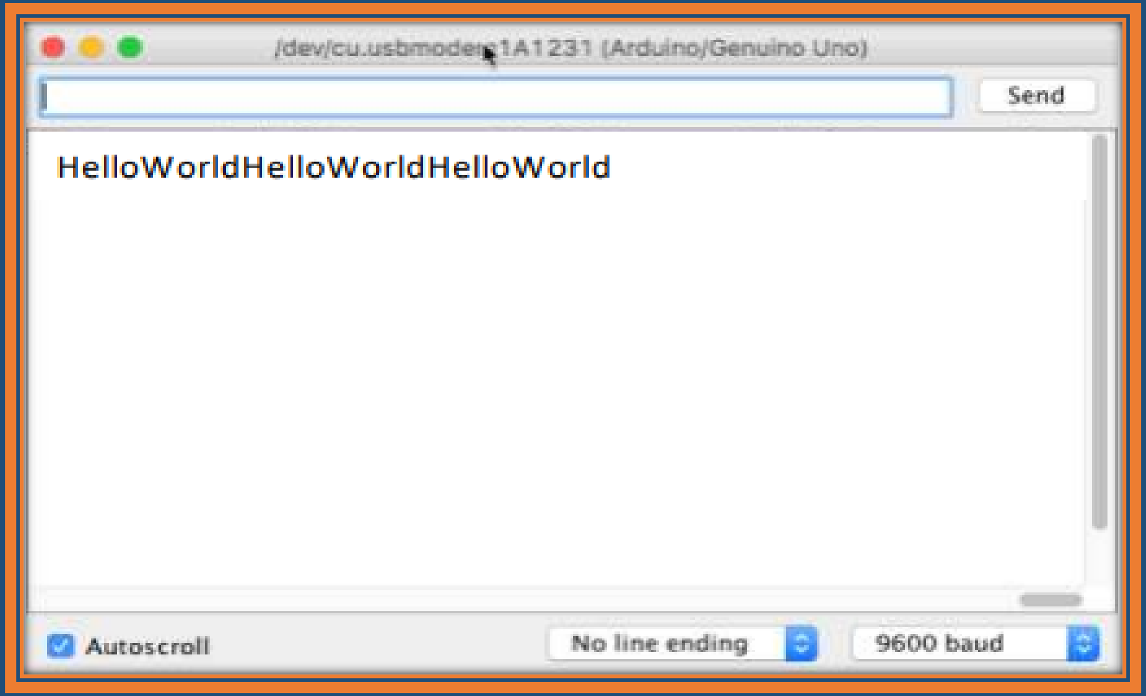
Do you get an Error Message?

Choose correct Board (Arduino UNO) and the correct Port for Communication between PC and Arduino Board



Serial Monitor

You use the Serial Monitor when Debugging Arduino programs or when you want to show data or values from your program. You need to have Arduino connected to your PC in order to use the Serial Monitor



/dev/cu.usbmodem1A1231 (Arduino/Genuino Uno)

```
HelloWorldHelloWorldHelloWorld
```

☑ Autoscroll

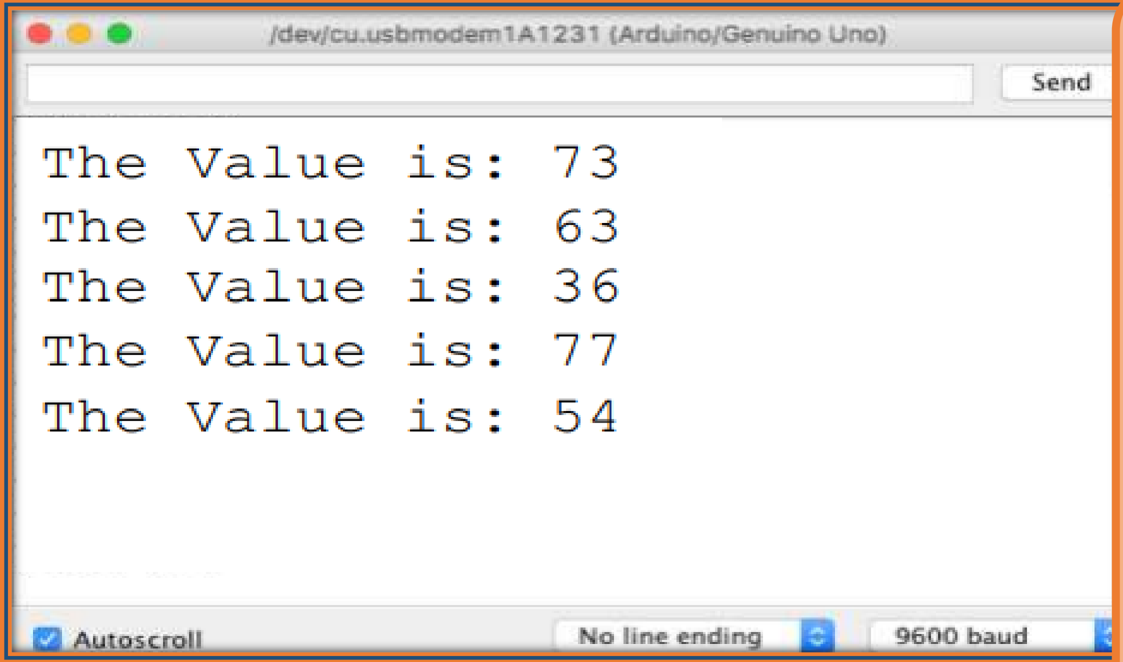
No line ending

9600 baud

TRY IT OUT!

```
void setup()
 {
Serial.begin (9600);
}
void loop()
{
Serial.print (”Hello World");
delay(1000);
}
```

Serial Monitor



```
The Value is: 73
The Value is: 63
The Value is: 36
The Value is: 77
The Value is: 54
```

☑ Autoscroll   No line ending   9600 baud

Here you see how we can write a value to the Serial Monitor.
This can be a value from a sensor, e.g., a temperature sensor.

TRY IT OUT!

```
int myValue = 0;
void setup()
{
Serial.begin(9600);
}
void loop()
{
myValue = random(100);
Serial.print (“The Value is: ”);
Serial.println (myValue);
delay(1000);
}
```

Arduino Programs

All Arduino programs must follow the following main structure:

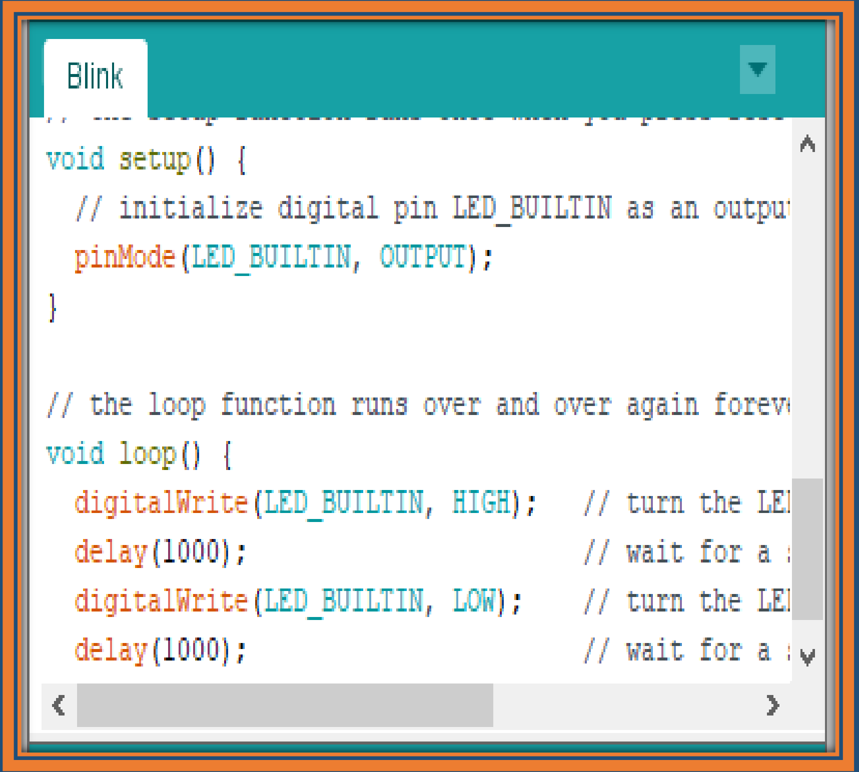
```
// Initialization, define variables, etc.
void setup()
{
  // Initialization ...
}
void loop()
{
//Main Program
...
}
```

Arduino Program - Example

```
void setup()
{
pinMode(11, OUTPUT);                    //Set the Pin as an Output
}
void loop()
{ digitalWrite(11, HIGH);              // Turn on the LED
delay(1000);                           // Wait for one second
digitalWrite(11, LOW);                 // Turn off the LED
delay(1000);                           // Wait for one second
}
```

Arduino Program – Using Comments

```
void setup()
{
pinMode(11, OUTPUT);            //Set the Pin as an Output
}
void loop()
{
digitalWrite(11, HIGH);            // Turn on the LED
/* ... This will not be executed by the program because it is
a comment... */
}
```



```
void setup() {
  // initialize digital pin LED_BUILTIN as an outpu
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forev
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LE
  delay(1000);                       // wait for a
  digitalWrite(LED_BUILTIN, LOW);    // turn the LE
  delay(1000);                       // wait for a
```

Creating and Using Functions

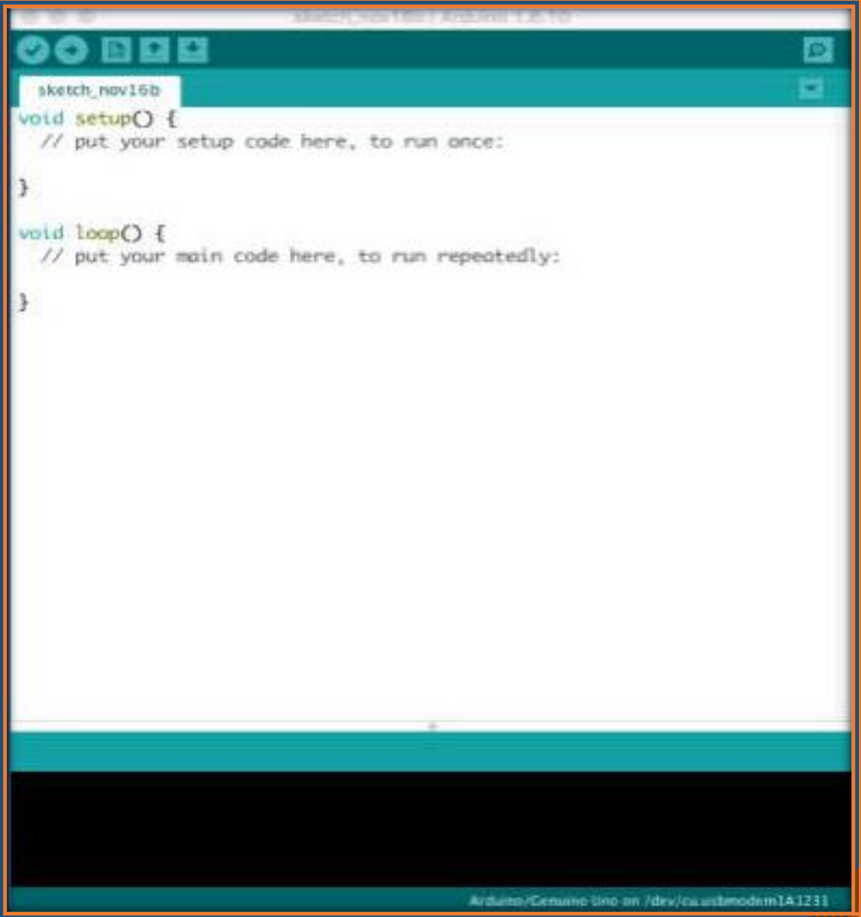
```
int z;

void setup()

{

        }

void loop()

{

   z = calculate(2,3);

}

float calculate(int x, int y)

{

     return (x + y);

}
```

Using the Function

Creating the Function

TRY IT OUT!

Here are some Arduino Examples you should try.
Make sure your Arduino is connected to the PC and start the Code Editor



sketch_nov16b

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Arduino/Genuino Uno on /dev/cu.usbmodem1A1231

“Hello World” Example

Create the following program:

Open the ”Serial Monitor” in order to see the output

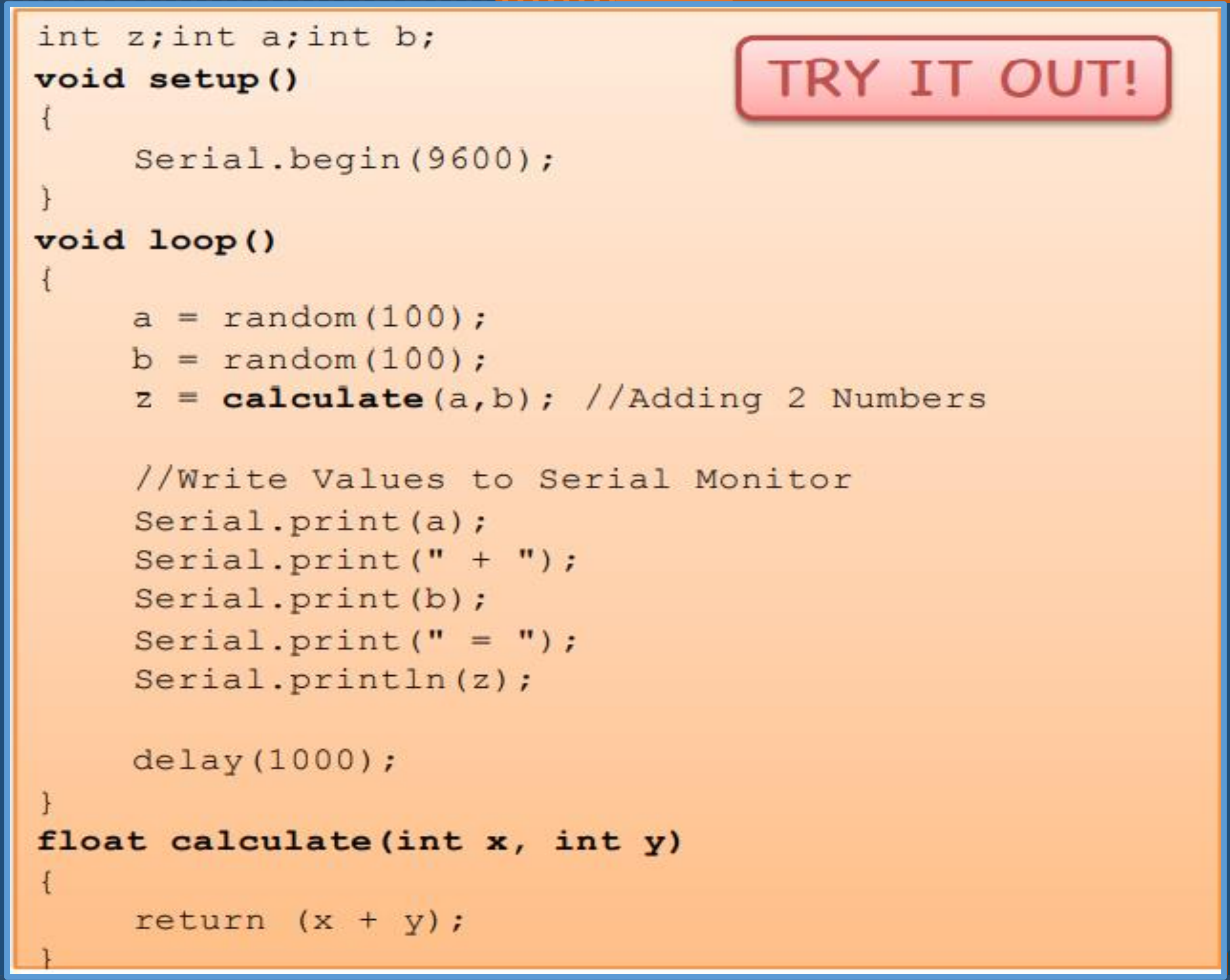
TRY IT OUT!

```
void setup()
{
Serial.begin(9600);
Serial.println("Hello, world!");
}
void loop()
{
}
```

Example

Create the following program:

Open the ”Serial Monitor” in order to see the output



```
int z;int a;int b;
void setup()
{
     Serial.begin(9600);
}
void loop()
{
     a = random(100);
     b = random(100);
     z = calculate(a,b); //Adding 2 Numbers

     //Write Values to Serial Monitor
     Serial.print(a);
     Serial.print(" + ");
     Serial.print(b);
     Serial.print(" = ");
     Serial.println(z);

     delay(1000);
}
float calculate(int x, int y)
{
     return (x + y);
}
```

Creating Functions

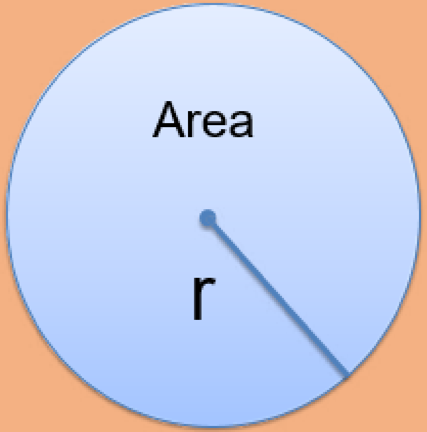
TRY IT OUT!

Create a function that calculates the area of a circle with a given radius.

Write the result to the Serial Monitor.

Solution

```
void setup()
{
   float area;
   Serial.begin(9600);
   // calculate the area of a circle with radius of 9.2  float r=9,2;
   area = CircleArea(r);
   Serial.print("Area of circle is: ");
   // print area to 4 decimal places
   Serial.println(area, 4);
}

void loop()
{
}

// calculate the area of a circle  float Circle Area(float radius)
{
   float result;
   const float pi = 3.14;
   result = pi * radius * radius;  return result;
}
```



For Loop

In this program we use a For Loop to find the Sum of 100 Random Numbers.

Then we find the Average.

The Sum and Average should be written to the Serial Monitor.

TRY IT OUT!

```
int x; int sum = 0; float gjennomsnitt = 0;
void setup()
{
   Serial.begin(9600)
   ;
}
void loop()
{
   sum = 0;
   for (int i = 0; i<100; i++)
   {
      x = random(100);
      sum = sum + x;
   }
   average = sum / 100;  Serial.print(" Sum = ");
   Serial.print(sum);
   Serial.print(" ,
   Average = ");
   Serial.println(average);
    delay(1000);
}
```

Arrays

Here we shall use arrays in the Arduino program

Create this program from scratch and open the Serial Monitor to see the result.

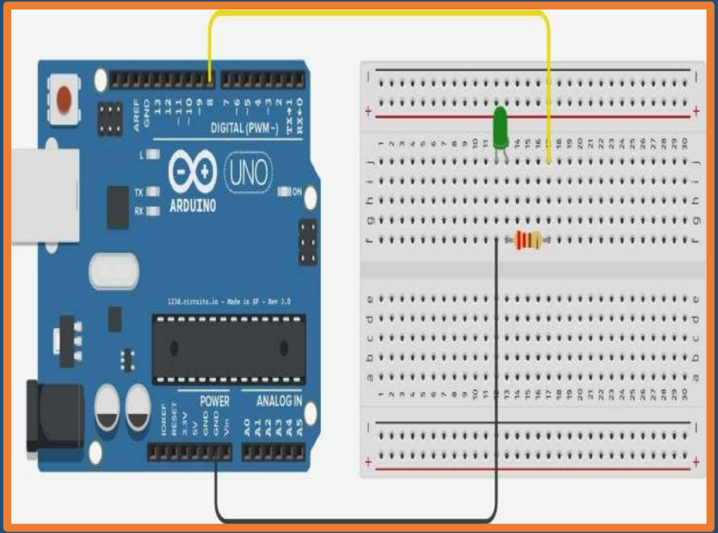
TRY IT OUT!

```
const int arraysize = 100;
int x; int sum = 0;
float average = 0;
int myarray [arraysize];
void setup()
{ Serial.begin(9600);
}
void loop()
{
sum = 0;
for (int i = 0; i < arraysize; i++)
{ x = random(200); myarray[i] = x; }
sum = calculate Sum(myarray);
average = sum / 100;
Serial.print(" Sum = ");
Serial.print(sum);
Serial.print(" , Average = ");
Serial.println(average);
delay(1000);
}
int calculate Sum (int sumarray[])
{
for (int i = 0; i < arraysize; i++)
{ sum = sum + sumarray[i];
}
return sum;
}
```

Blinking LED Example

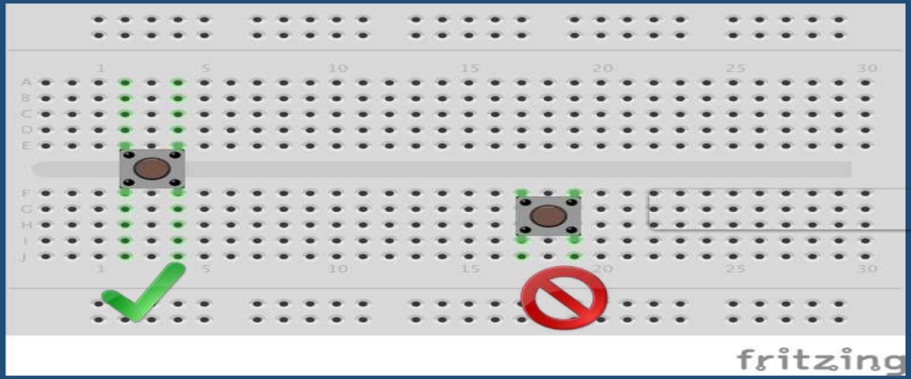
```
int ledPin = 8;                             // Pin where the LED is connected

void setup() {
  pinMode( ledPin, OUTPUT);                 // Set the LED pin as output
}

void loop() {
  digitalWrite( ledPin, HIGH);                  // Turn the LED on
  delay(1000);                                  // Wait for a second
  digitalWrite( ledPin, LOW);                   // Turn the LED off
  delay(1000);                                  // Wait for a second
}
```

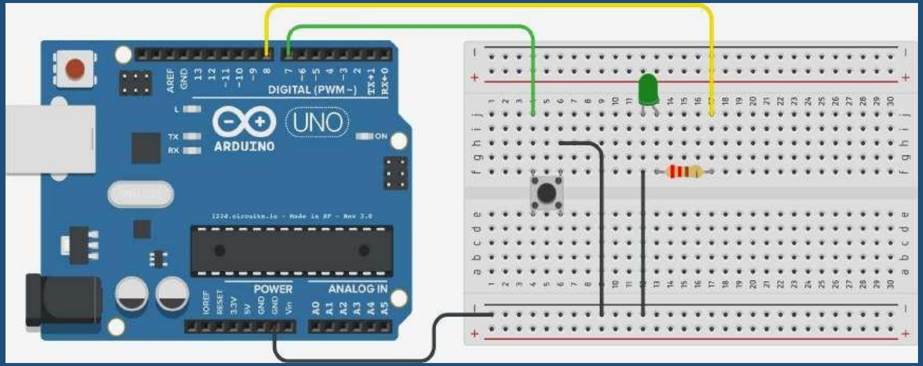
TRY IT OUT!



Switch Example

```
int switchPin = 2;   // Pin where the switch is connected
int ledPin = 13; // Pin where the LED is connected
int switchState = 0; // Variable to store switch state

void setup() {
  pinMode( switchPin, INPUT); // Set the switch pin as input
  pinMode(ledPin, OUTPUT); // Set the LED pin as output
}

void loop() {
  switchState = digitalRead( switchPin); // Read the switch state

  if ( switchState == HIGH) {
    digitalWrite(ledPin, HIGH); // Turn the LED on if switch is pressed
  } else {
    digitalWrite(ledPin, LOW); // Turn the LED off if switch is not pressed
  }
}
```

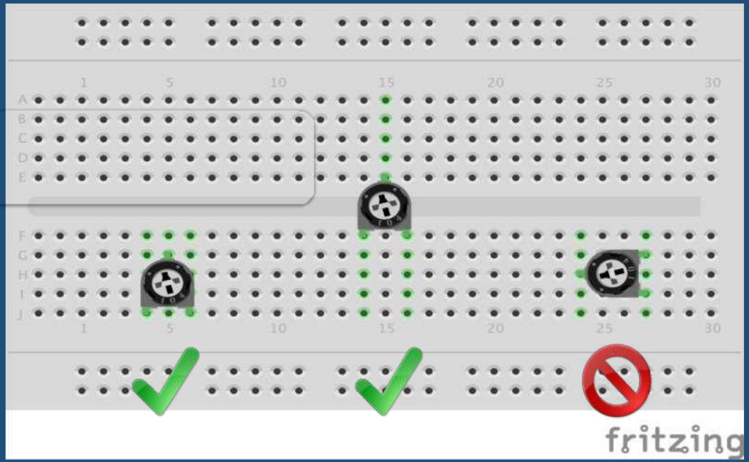




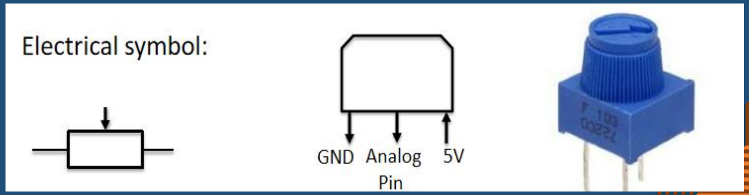
Note! In this configuration, we use an internal "pull-up" resistor to prevent "short-circuiting".

Potentiometer Example

```
int potPin = A0;                              // Pin where the potentiometer is connected
int ledPin = 9;                               // Pin where the LED is connected
int potValue = 0;                             // Variable to store the potentiometer value

void setup()
{
  pinMode(ledPin, OUTPUT);                              // Set the LED pin as output
}

void loop()
{
  potValue = analogRead (potPin);                       // Read the potentiometer value
  int brightness = map( potValue, 0, 1023, 0, 255);     // Map value to range 0-255
  analogWrite (ledPin, brightness);                     // Set LED brightness
}
```

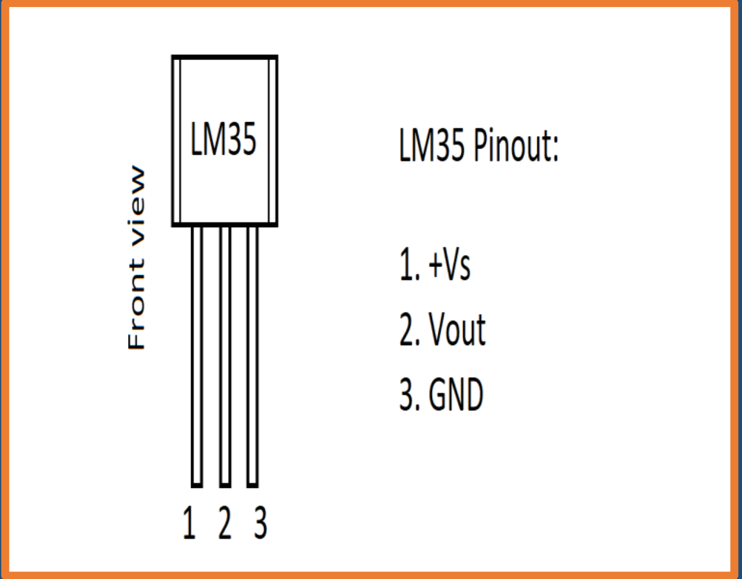


Make sure to place the Potentiometer correctly on the Breadboard



Temperature Sensor (LM35) Example

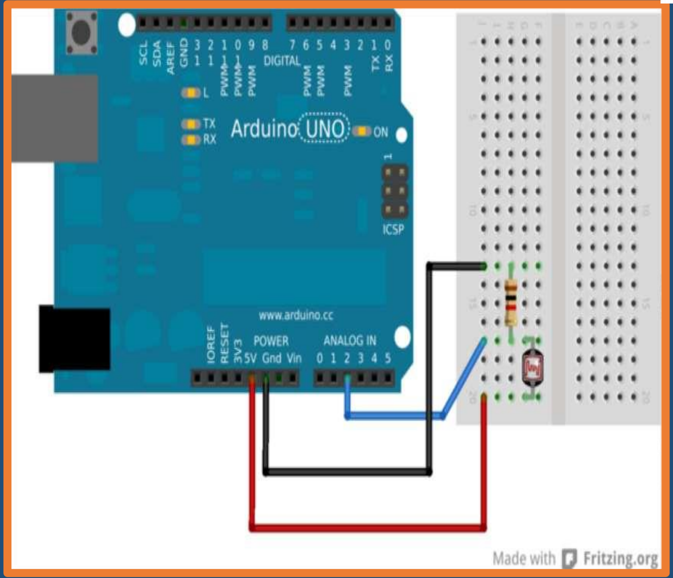
```
int tempPin = A0;                  // Pin where the temperature sensor is connected
int ledPin = 13;                   // Pin where the LED is connected
float tempC = 0;                   // Variable to store temperature in Celsius
void setup() {
  pinMode(ledPin, OUTPUT);              // Set the LED pin as output
  Serial.begin(9600);                   // Begin serial communication for debugging
}
void loop() {
  int reading = analogRead( tempPin);          // Read the temperature sensor value
  tempC = reading * (5.0 / 1023.0) * 100;      // Convert to Celsius
  Serial.println( tempC);                      // Print temperature to the serial monitor

  if ( tempC > 30) {
    digitalWrite(ledPin, HIGH);                // Turn on LED if temp > 30°C
  } else {
    digitalWrite(ledPin, LOW);                 // Turn off LED otherwise
  }
  delay(1000);
}
```



Light Sensor (LDR) Example

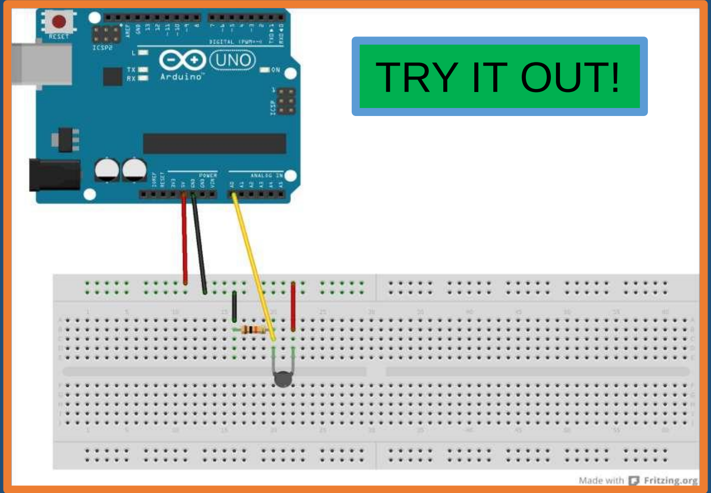
```
int ldrPin = A0;                          // Pin where the LDR is connected
int ledPin = 13;                          // Pin where the LED is connected
int ldrValue = 0;                         // Variable to store LDR value

void setup() {
  pinMode(ledPin, OUTPUT);      // Set the LED pin as output
  Serial.begin(9600);                  // Begin serial communication for debugging
}
void loop() {
  ldrValue = analogRead(ldrPin);       // Read the LDR value
  Serial.println(ldrValue);                // Print LDR value to serial monitor

  if (ldrValue < 500) {                       // If it’s dark (low LDR value)
    digitalWrite(ledPin, HIGH);          // Turn the LED on
  } else {
    digitalWrite(ledPin, LOW);           // Turn the LED off
  }
  delay(500);
}
```

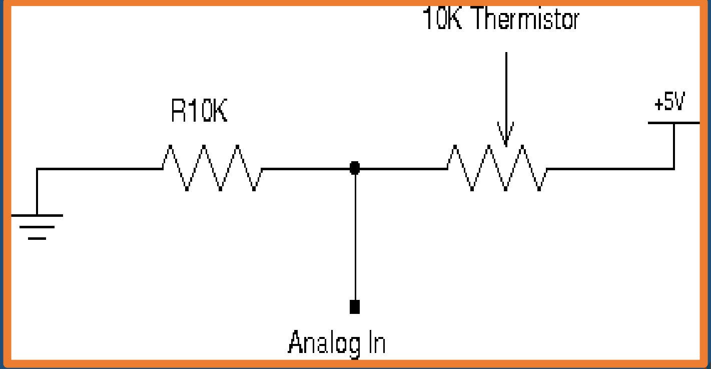
TRY IT OUT!



Thermistor Example

```
int thermistorPin = A0;                     // Pin where the thermistor is connected
int ledPin = 13;                            // Pin where the LED is connected
float tempC = 0;                            // Variable to store temperature in Celsius
void setup() {
  pinMode(ledPin, OUTPUT);                  // Set the LED pin as output
  Serial.begin(9600);                       // Begin serial communication for debugging
}
void loop() {
  int reading = analogRead( thermistorPin);      // Read the thermistor value
  float resistance = (1023.0 / reading) - 1.0;
  resistance = 10000.0 / resistance;             // Assuming a 10K thermistor
  tempC = resistance / 10000;         // Simplified formula to approximate temp in Celsius
  Serial.println(tempC);              // Print temperature to serial monitor
  if (tempC > 30) {                   // If temperature exceeds 30°C
    digitalWrite(ledPin, HIGH);       // Turn on LED
  } else {
    digitalWrite(ledPin, LOW);        // Turn off LED
  }
  delay(1000);
}
```





Thank You..