

---

# BE MAKER



# BE MAKER

ELETTRONICA, ROBOTICA E CODING PER RAGAZZI... E NON SOLO !

**COURSE OF ELECTRONICS, ROBOTICS AND CODING FOR  
CHILDREN... and not only**

**BASIC COURSE – lesson 3**

---

## Index

|  |           |
|--|-----------|
| <b>Warnings .....</b>                                      | <b>3</b>  |
| <b>Notes sul Copyright .....</b>                           | <b>3</b>  |
| <b>Robotics with Arduino.....</b>                          | <b>4</b>  |
| <b>The photoresistor or also called Light Sensor .....</b> | <b>6</b>  |
| <b>Project 8 – Twilight Switch with Photoresistor.....</b> | <b>10</b> |
| Sketch Analysis: Twilight Switch with Photoresistor .....  | 13        |
| <b>Fun Fact: What is Light .....</b>                       | <b>14</b> |
| <b>Project 9 – Luxmetro with Arduino .....</b>             | <b>15</b> |
| Sketch Analysis: Luxmetro with Arduino.....                | 19        |
| <b>The PWM signal - Pulse Width Modulation .....</b>       | <b>20</b> |
| Relationship between Analog PIN and PWM Digital PIN .....  | 23        |
| <b>Project 10 – Night and Day with PWM signal.....</b>     | <b>24</b> |
| Sketch Analysis: Night and Day with PWM Signal .....       | 27        |
| <b>Project 11 – The Return of the Cylons.....</b>          | <b>28</b> |
| Sketch Analysis: The Return of the Cylons.....             | 32        |

---

## Warnings

*With regard to the safety aspects, since the projects are based on a very low voltage power supply supplied by the USB port of the PC or by support batteries or power supplies with a maximum of 9V output, there are no particular risks of an electrical nature. It is however necessary to specify that any short circuits caused during the exercise phase could produce damage to the PC, to the furnishings and in extreme cases even to burns, for this reason every time a circuit is assembled, or changes are made on it, it will be necessary to do so in the absence of power and at the end of the exercise it will be necessary to provide for the disconnection of the circuit by removing both the USB cable connecting to the PC and any batteries from the appropriate compartments or external power connectors. In addition, always for safety reasons, it is strongly recommended to carry out projects on insulating and heat-resistant carpets that can be purchased in any electronics store or even on specialized websites.*

*At the end of the exercises it is advisable to wash your hands, as the electronic components could have processing residues that could cause damage if ingested or if in contact with eyes, mouth, skin, etc. Although the individual projects have been tested and safe, those who decide to follow what is reported in this document, assume full responsibility for what could happen in the execution of the exercises provided for in the same. For younger children and / or the first experiences in the field of Electronics, it is advisable to perform the exercises with the help and in the presence of an adult.*

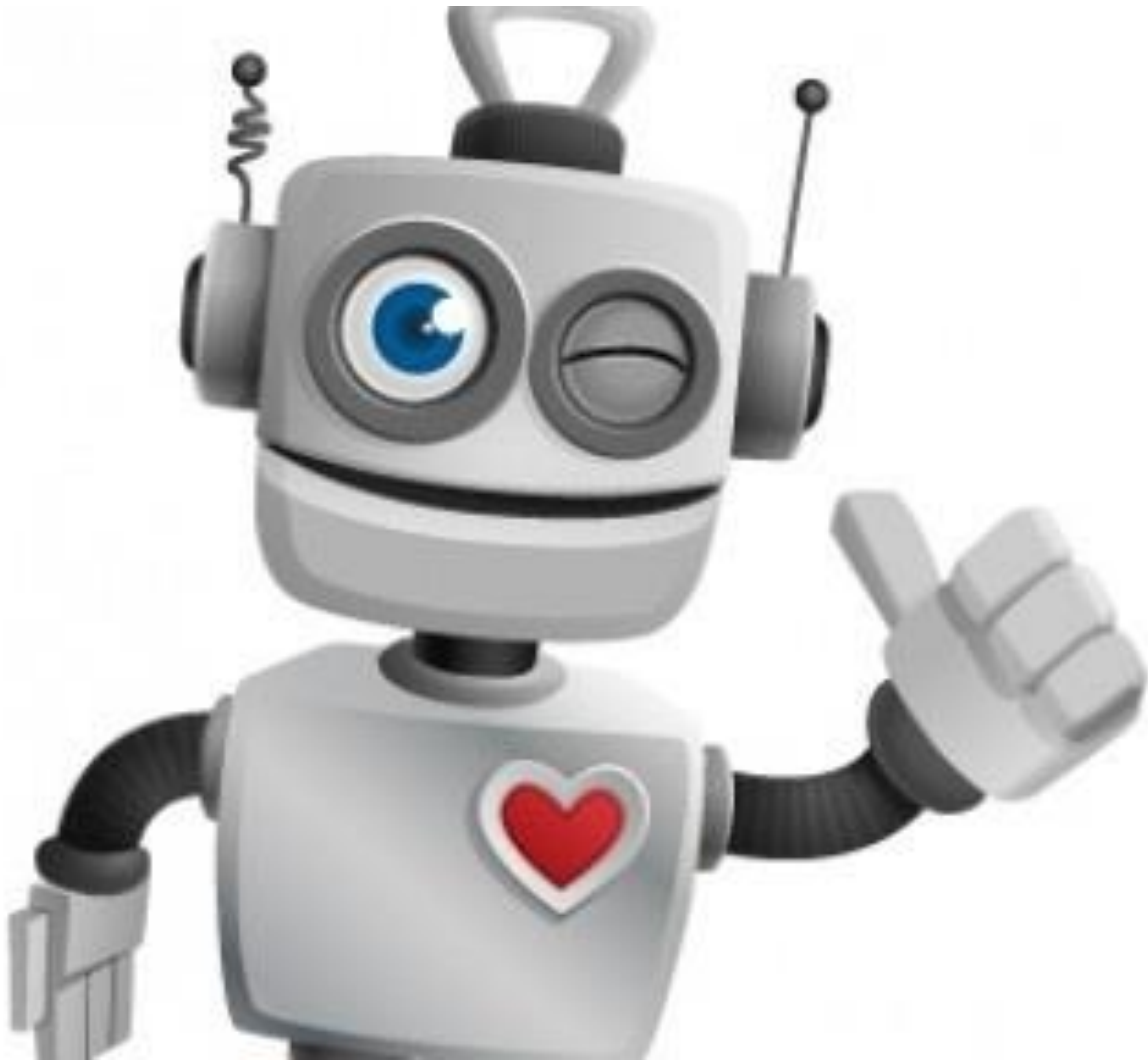
## Notes sul Copyright

*All trademarks are the property of their respective owners; third-party trademarks, product names, trade names, corporate names and companies mentioned may be trademarks owned by their respective owners or registered trademarks of other companies and have been used for purely explanatory purposes and for the benefit of the owner, without any purpose of violation of the copyright rights in force. What is reported in this document is the property of Roberto Francavilla, Italian and European laws on copyright are applicable to it – any texts taken from other sources are also protected by the Copyright and property of the respective Owners. All the information and contents (texts, graphics and images, etc.) reported are, to the best of my knowledge, in the public domain. If, unintentionally, you have published material that is subject to copyright or in violation of the law, please notify [bemaker.school@gmail.com](mailto:bemaker.school@gmail.com) by email and I will promptly remove it.*

*Roberto Francavilla*

---

## Robotics with Arduino



*With this lesson we begin to explore the world of sensors that we can connect to Arduino and then with this we also begin to talk about robotics.*

*Did you know that with Arduino you can build real robots? No!*

*But siiiii!!! It all depends on your imagination.*

*The word "robot" is presumed to derive from the Czech language which means "slave" and was a word used in ancient Europe to indicate people who did heavy and repetitive work, but the best use of the word Robot is, in my opinion, of the great writer **Asimov** who also fixed its laws:*

***First Law:** "A robot cannot harm a human being nor can it allow a human being to receive harm because of its failure to intervene."*

***Second law:** "A robot must obey orders given by humans as long as such orders do not contravene the first law."*

***Third law:** "A robot must protect its existence as long as this does not conflict with the first and second laws".*

*Obviously today with the word robot we indicate everything that performs operations mechanically automatically thanks to an electronic controller, with particular reference to*

programmable appliances used in some sectors of industry, but also for domestic uses (see for example food processors, robot vacuum cleaners ..)

What allows a machine to have reactions and therefore a certain behavior, are the **SENSORS**, or those electronic devices that detect the values of a physical quantity and transmit its variations.

Sensors are also called **TRANSDUCERS** which can be classified into **active** if the transducers generate a signal in current or voltage or **passive** if the transducers produce at the output a variation of an electrical parameter such as resistance, capacity etc ...

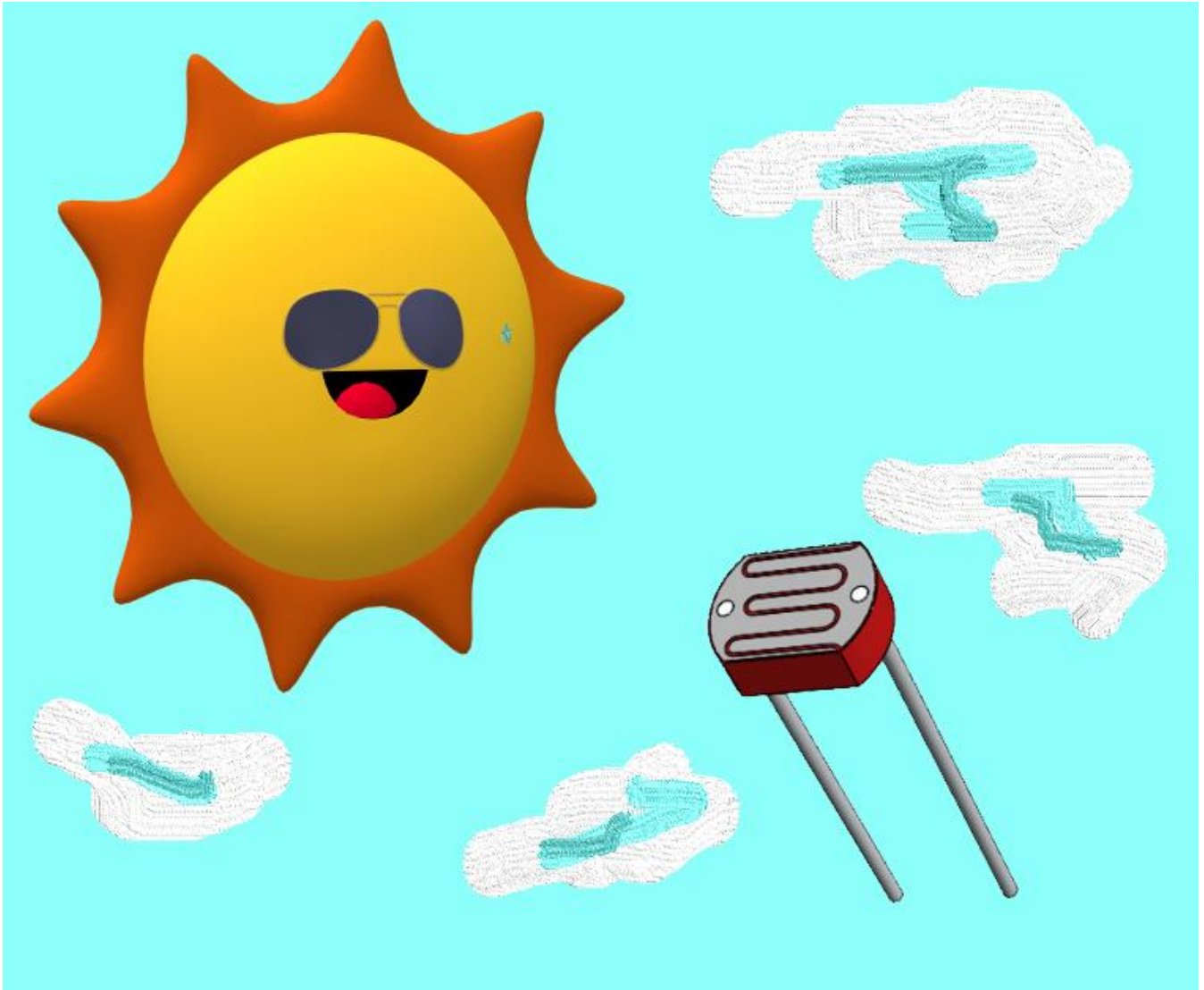
Another classification criterion refers to the type of signal produced at the output.

**Analog** transducers have an output signal that continuously follows the variations in the input physical quantity. **Digital transducers** provide output value of magnitude in numeric terms with discrete values. However, do not worry, at the end of the course you will be very good at understanding the operation of many sensors.



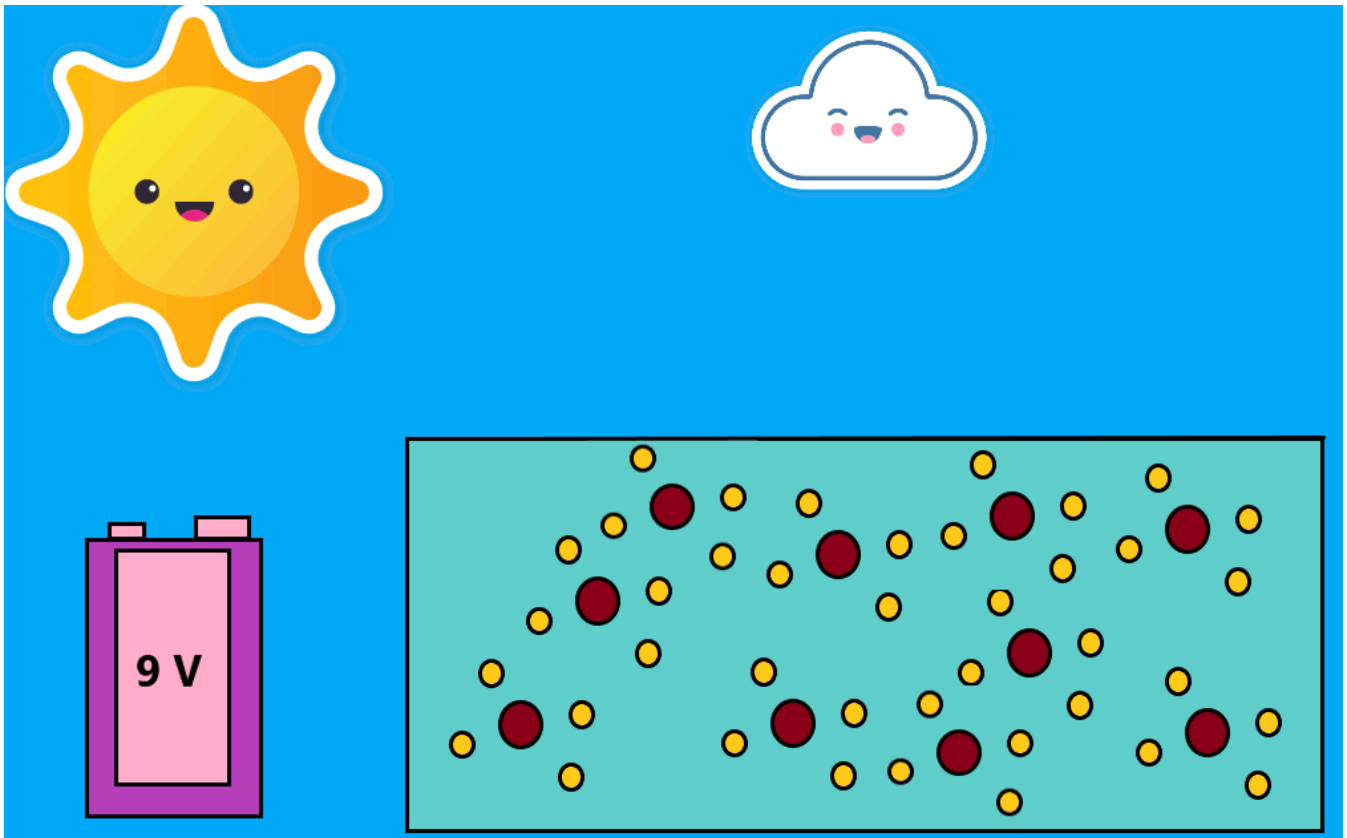


## The photoresistor or also called Light Sensor



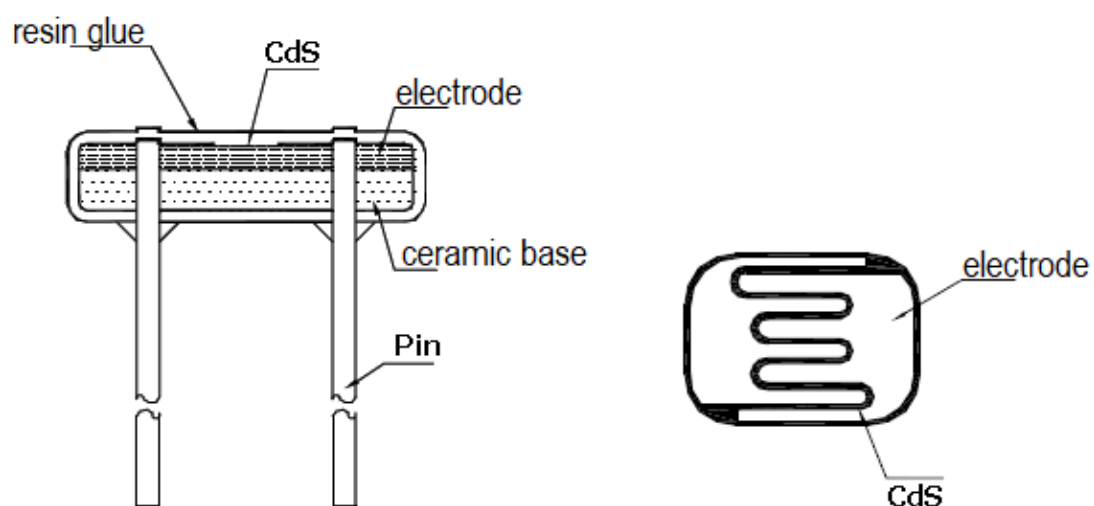
*In this lesson we will talk about the light sensor. In particular, we will talk about photoresistors its commercial English name is LDR that is Light Dependent Resistor, or that electronic component that as the brightness of the environment in which it is positioned changes its electrical resistance and therefore, with this variation, we can establish different behaviors to be performed by Arduino.*

*The principle of operation of an LDR is based on particular characteristics possessed by some substances that subjected to the action of light (photons), modify their resistivity characteristics. Electrical resistivity, also called specific electrical resistance, is the ability of a material to oppose resistance to the passage of electric charges. In the International System, resistivity is measured in Ohms per meter ( $\Omega \cdot m$ ). Resistivity differs from strength in that it is a specific property of the material in question and depends mainly on the microscopic structure of the medium; on the contrary, the resistance  $R$  is a property of the entire conductor under examination (macroscopic property).*

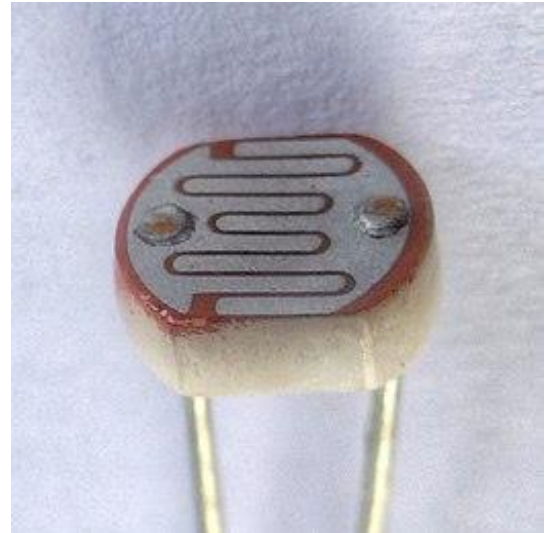
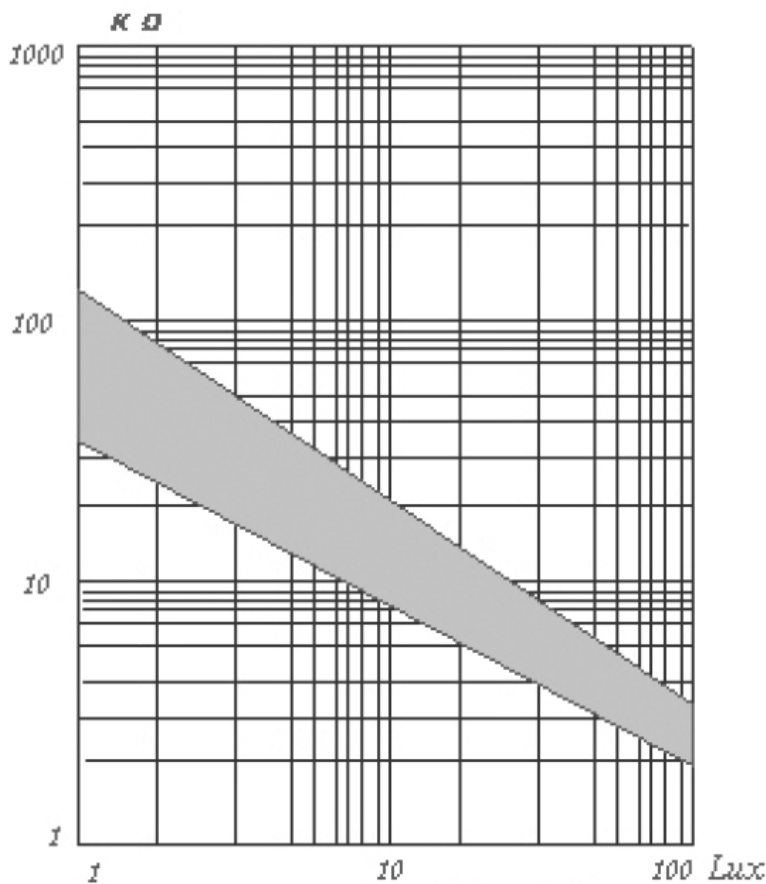


*The animated figure above shows how some materials affected by photons (light) absorb energy and are able to conduct more electric current when subjected to an electrical voltage (electrons are released from their bonding.) These materials, therefore, modify their resistivity, lowering it, when they are hit by light. In the absence of light, photoresistors act as materials with high resistivity.*

*For our projects we will refer to the GL 5528 photoresistor:*



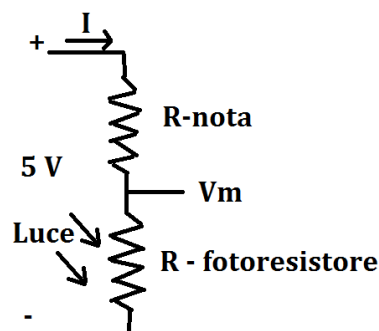
As we have said there is a link between the amount of light that hits the photoresistor and the Resistance value that the sensor assumes, this bond is expressed by means of a graph (logarithmic base) shown below:



On the x-axis (abscissa), there are Lux (i.e. light intensity), on the y-axis (ordinate) there are kOhm (i.e. Resistance). As you can see from the graph, the greater the light and the smaller the Resistance.

The gray band represents the uncertainty band (tolerance) of the LDR sensor - GL5528.

The circuit in which a photoresistance is inserted is called **a voltage divider** and is the same circuit used to measure the unknown resistance in the ohmmeter seen in the previous lesson, that is, two resistances are put in series, one of which is a note and the voltage value ( $V_m$ ) is measured at the junction point of the two resistances. Knowing the voltage applied to the circuit (5 V) it is possible to determine the value of the unknown resistance of the photoresistor and then to measure the brightness:





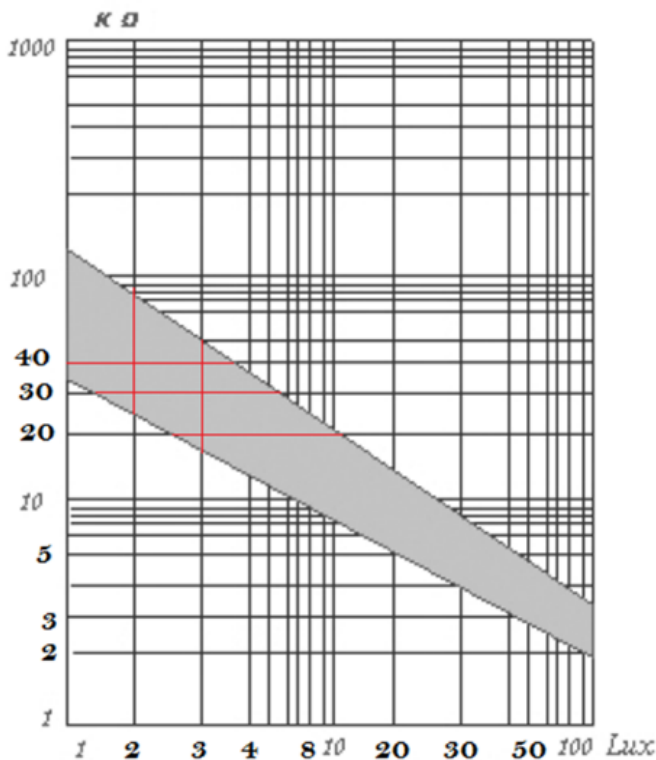
The voltage  $V_m$  we measure to Arduino bringing the signal to its analog PIN, for example the A0, at this point, note the voltage  $V_m$ , we know the voltage applied on the known resistance, which is obtained by deduction, that is:  $5\text{ V} - V_m$ , consequently we can calculate the electric current " $I$ " which for the law of Ohm ( $V = R \times I$ ) is equal to (using the inverse formula):

$$I = \frac{5 - V_m}{R_{\text{nota}}}$$

and finally we can calculate the value of the unknown resistance:

$$R_{\text{fotorestore}} = \frac{V_m}{I}$$

If we also have the table of correspondence between the degree of brightness (which is measured in lux) and the resistance value, we also know how much light is hitting the photoresistor moment by moment. For example, the manufacturers of photoresistors release what in jargon are called data-sheets (ie data sheets) of the components (an example is the figure below referred to the photoresistor GL5528):










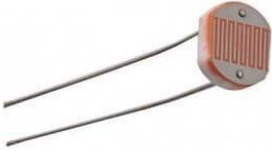
| kOhm     | Lux      |
|----------|----------|
| 2        | > 100    |
| 3        | circa 80 |
| 4        | circa 40 |
| 5        | circa 30 |
| 5 - 10   | circa 20 |
| 10 - 30  | circa 5  |
| 30 - 100 | circa 1  |

which tells us that with 10-15 lux of brightness, corresponds to a resistance value of about 10-15 kOhm.

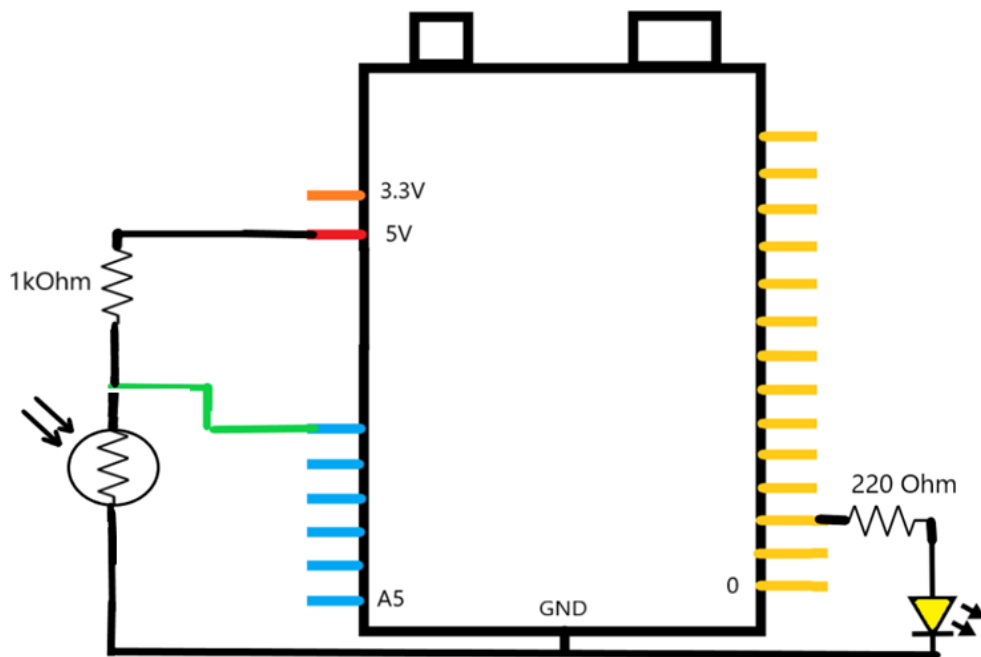
Because I believe I have reached and passed the attention curve... and so I am sure that from this moment on you will no longer receive anything ..., let's move on to practical things and therefore move on to carry out the project that follows.

## Project 8 – Twilight Switch with Photoresistor

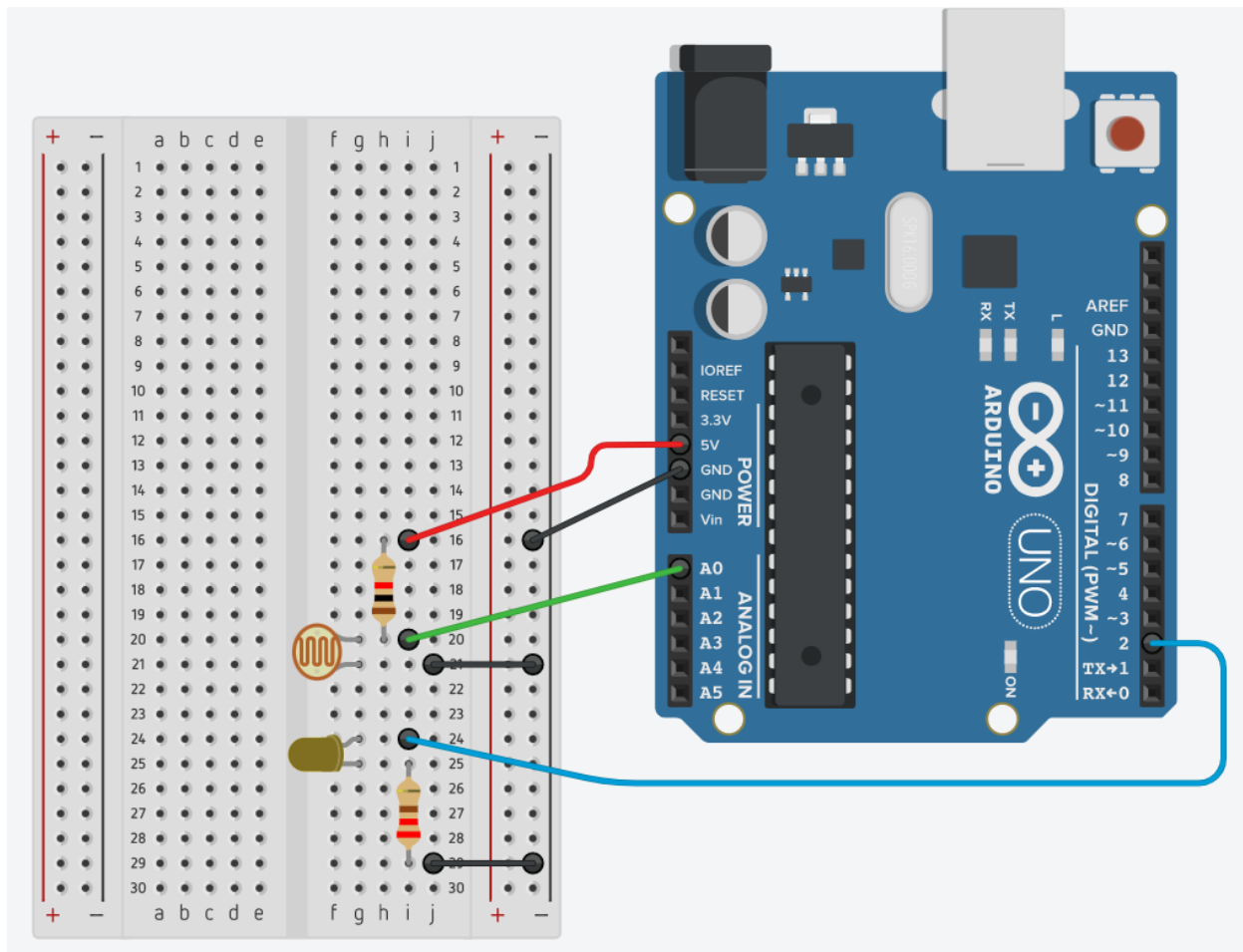
For this project we need:

|   |   |  |   |
|---|---|--|---|
|  |  |  |  |
| Arduino Uno R3 or compatible  | Breadboard  | Dupont cables male - male  | USB connection cable  |
|  |  |  |  |
| Resistance 220 Ohm  | Yellow LED  | Resistance 1 kOhm  | Photoresistor GL5528  |

The wiring diagram is:



For assembly it is necessary to follow the scheme below:



After the links we move on to write the sketch.

Connect Arduino to the PC via the USB cable and launch the Arduino IDE application by clicking twice on the relevant icon.



An empty window opens, or you need to open a new empty one and write the sketch below:

```
/*  
Be Maker School - STEM - Project 8 – Twilight  
*/  
void setup() {  
  pinMode(2,OUTPUT); you set THE PIN 2 as the output  
  Serial.begin(9600); you set the serial port  
}  
  
void loop() {  
  int reading=analogRead(A0); reads PIN A0  
  float Vout=(read*5)/1024.0; determines the Vout based on  
                                of the value read on pin A0  
  float Rnota=1000.0; value of the known Resistance i.e. 1000 Ohm
```

`current float=(5-Vout)/Rnota;` calculates the current  
`float ResIncognita=Vout/current;` calculates the unknown resistance

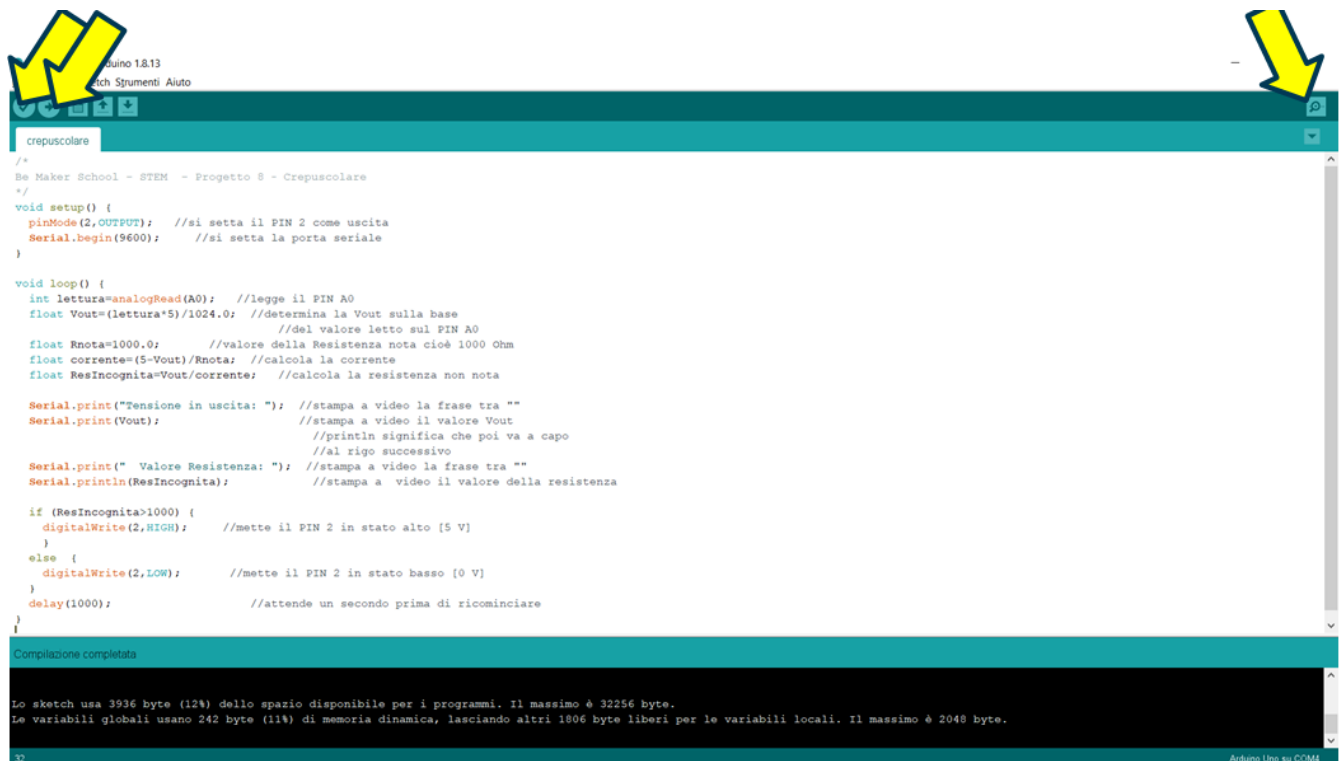
`Serial.print("Output voltage: ");` print on the screen the phrase between ""  
`Serial.print(Vout);` print the Vout value on screen

`println` means that then wraps  
to the next line

`Serial.print(" Resistance Value: ");` print on the screen the phrase between ""  
`Serial.println(ResIncognita);` print on screen the value of the resistance

```
if (ResIncognita>1000) {  
    digitalWrite(2,HIGH); puts PIN 2 in a high state [5 V]  
}  
else {  
    digitalWrite(2,LOW); puts PIN 2 in low state [0 V]  
}  
delay(1000); wait a second before starting again  
}
```

Once the code has been written, launch the verification precompilation (check mark), it will ask you to save the sketch (you can change its name) and then click on the arrow for loading and on the lens for the serial monitor:



Once the sketch has started, make checks, lowering the brightness of your camera and raising it, when the camera becomes dark then Arduino commands the lighting of an LED (the reference value can be lowered or increased according to the lighting conditions of your room).

[For the video-project click [here](#)]

---

### **Sketch Analysis: Twilight Switch with Photoresistor**

Analyzing the sketch we see that it is basically similar to the previous one on the Ohmeter, so there would be nothing to highlight apart from the use of the control structure of the "if".

The control structure works like this: if the condition between the round brackets of the if is true, then Arduino will execute the instructions contained in the braces of the if, otherwise it means that the condition is not verified and everything that is enclosed between the braces of the else will be executed.

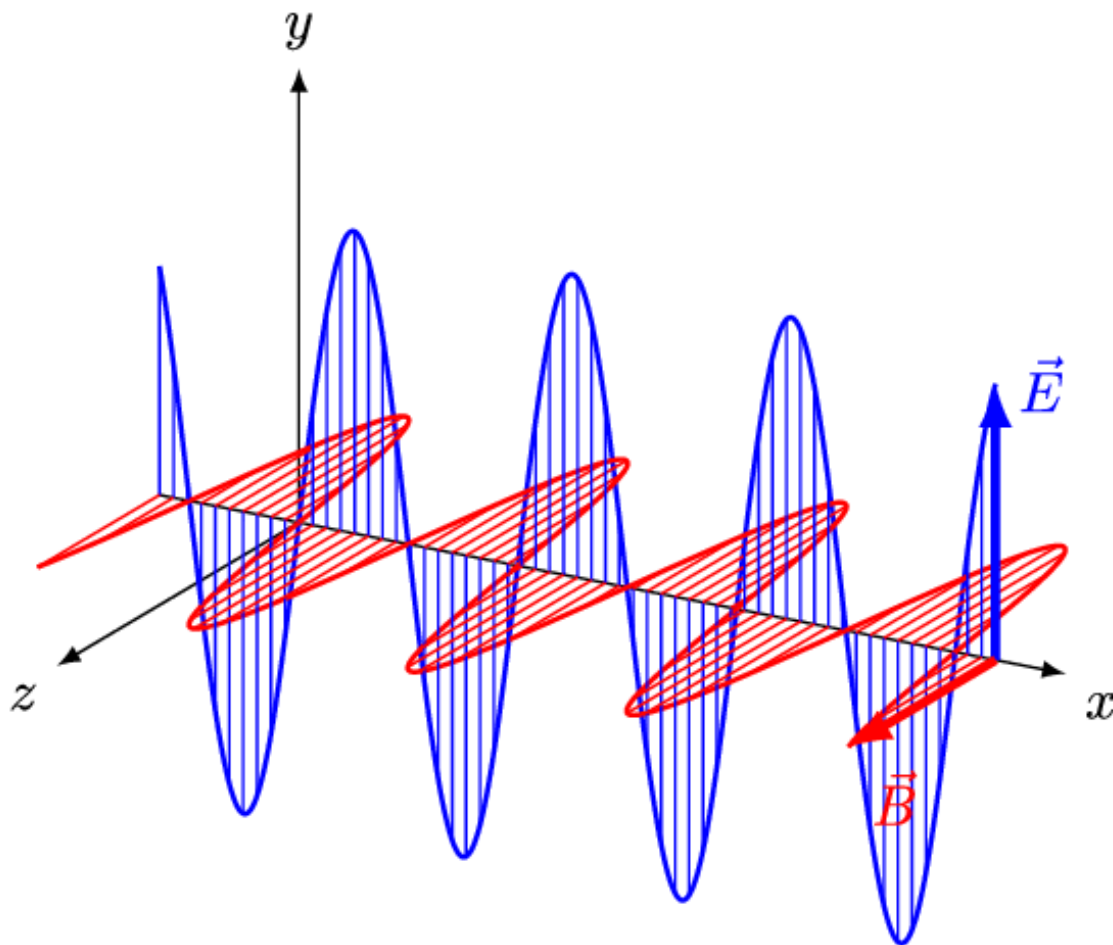
The syntax is as follows:

```
if (condition) {  
  statement(s)  
}  
else ) {  
  statement(s)  
}
```



## Fun Fact: What is Light

Until the early 1900s Light was thought to consist of **electromagnetic waves** and as such followed the laws of classical electromagnetism; waves consisted of two sinusoidal components perpendicular to each other (see animated gif: blue magnetic field and red electric field):



The frequency of these waves, that is, the periodicity with which the waves repeat, if it was in the range between 400 and 700 nanometers in wavelength (it is called **the spectrum**, the frequency range), could be visible to the human eye (**visible light spectrum**). Each frequency, of this spectrum, also defines the color of the light and the sum of all the frequencies of visible light form the so-called **White Light** which is then the one emitted by the sun.

In the early 1900s, in particular between 1900 and 1905, Max Planck and Albert Einstein with the theories and experiments of quantum neo-physics, introduced the concept of **photon**, and therefore defined the light composed of photons or particles of zero mass, but made of energy alone.


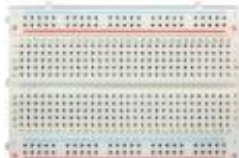






This particle, the photon, also has the properties of being an electromagnetic wave.

## Project 9 – Luxmetro with Arduino

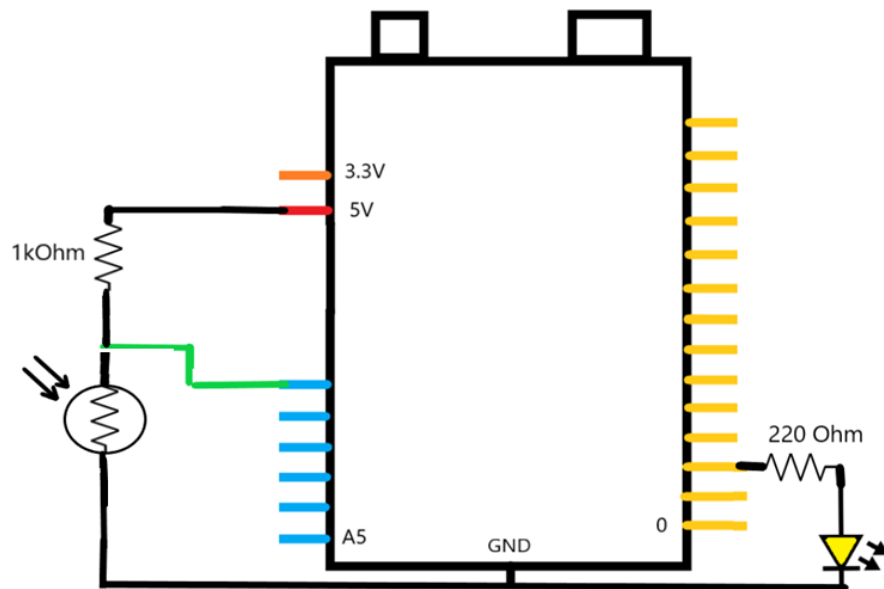


Also in this project with Arduino we will create a measuring instrument, namely the Luxmetro. What is the Lux meter for? Serve to measure the light intensity present in a given environment or how much light is hitting an object... do you know the tools that photographers used? well! those were essentially lux meters; once the light was measured, they gave indications on the shutter opening time and the aperture of the aperture.

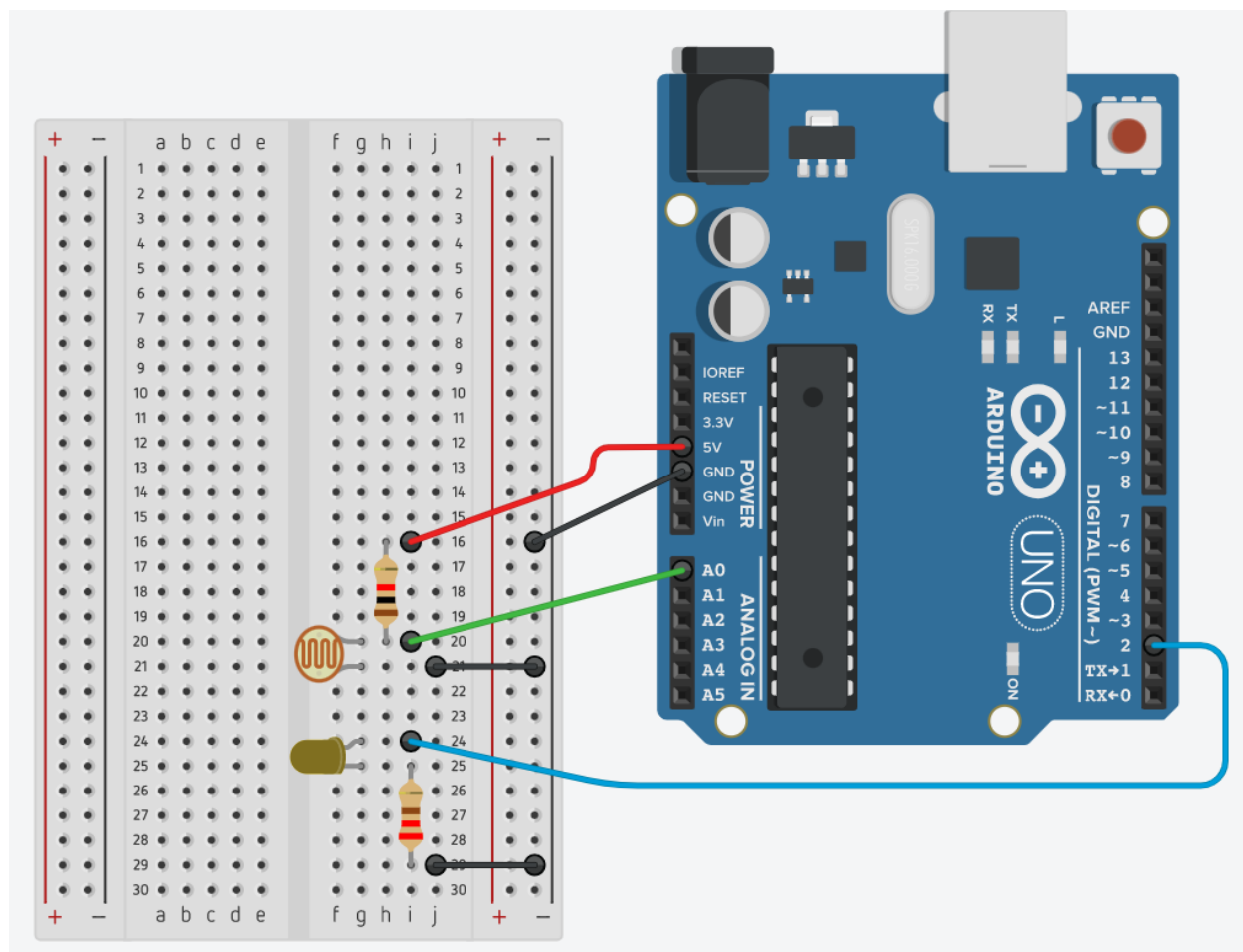
For this project we need:

|   |   |  |   |
|---|---|--|---|
|  |  |  |  |
| Arduino Uno R3 or compatible  | Breadboard  | Resistance 1kOhm   | Dupont cables male - male   |
|  |  |  |  |
| USB connection cable  | LDR Photoresistor GL5528  | 220 Ohm resistance   | LED   |

The wiring diagram is as follows and is identical to the previous one:



As well as the assembly scheme is identical to that of the previous project:



Before moving on to the sketch, you need a little explanation on how to derive the value in lux of the brightness by measuring the value of the resistance of the photoresistor, in essence we will use the table already described above and that is:

| kOhm     | Lux      |
|----------|----------|
| 2        | > 100    |
| 3        | circa 80 |
| 4        | circa 40 |
| 5        | circa 30 |
| 5 - 10   | circa 20 |
| 10 - 30  | circa 5  |
| 30 - 100 | circa 1  |

the table refers to the GL5528, obviously if you were to use another type of photoresistor, the table will be reconstructed in the same way as I did above.

For the association of values in the table we will use what in computer jargon is called the "nested if" method, this method is generally avoided, but for educational reasons I propose it to you.

At this point we click twice on the Arduino icon on the Desktop and the IDE opens and we copy the following sketch, I recommend you follow each step indicated:

```
/*
Be Maker School - STEM - Project 9 – Luxmetro
*/
void setup() {
  pinMode(2,OUTPUT);  you set THE PIN 2 as the output
  Serial.begin(9600);  you set the serial port
}

void loop() {
  int reading=analogRead(A0);  reads PIN A0
  float Vout=(read*5)/1024.0;  determines the Vout based on
                               of the value read on pin A0
  float Rnota=1000.0;          value of the known Resistance i.e. 1000 Ohm
  current float=(5-Vout)/Rnota; calculates the current
  float ResIncognita=Vout/current;  calculates the resistance of the photoresistor

  Serial.print(" Photoresistor Value [kOhm]: ");  print on the screen the phrase between ""
  Serial.print(ResIncognita);                    prints the value of the resistance
  Serial.print (" Illuminance Value [lux]: ");    print on the screen the phrase between ""

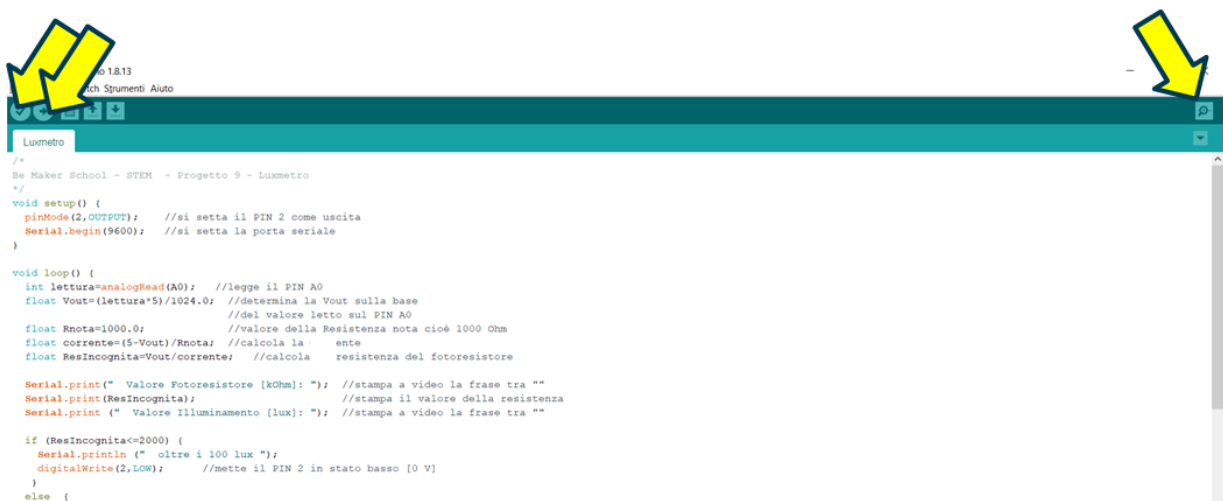
  if (ResIncognita<=2000) {
    Serial.println (" over 100 lux ");
    digitalWrite(2,LOW);  puts PIN 2 in low state [0 V]
  }
}
```

```

else {
  if (ResIncognita>2000 && ResIncognita<3500) {
    Serial.println (" about 80 lux ");
    digitalWrite(2,LOW);
  }
  else {
    if (ResIncognita>=3500 && ResIncognita<4500) {
      Serial.println (" about 40 lux ");
      digitalWrite(2,LOW);
    }
    else {
      if (ResIncognita>=4500 && ResIncognita<5500) {
        Serial.println (" about 30 lux ");
        digitalWrite(2,LOW);
      }
      else {
        if (ResIncognita>=5500 && ResIncognita<9500) {
          Serial.println (" about 20 lux ");
          digitalWrite(2,LOW);
        }
        else {
          if (ResIncognita>=9500 && ResIncognita<29500) {
            Serial.println (" about 5 lux ");
            digitalWrite(2,LOW);
          }
          else {
            if (ResIncognita>=29500) {
              Serial.println (" about 1 lux ");
              digitalWrite(2,HIGH); puts PIN 2 in a high state [5 V]
            }
          }
        }
      }
    }
  }
}
delay(1000);          wait a second before starting again
}

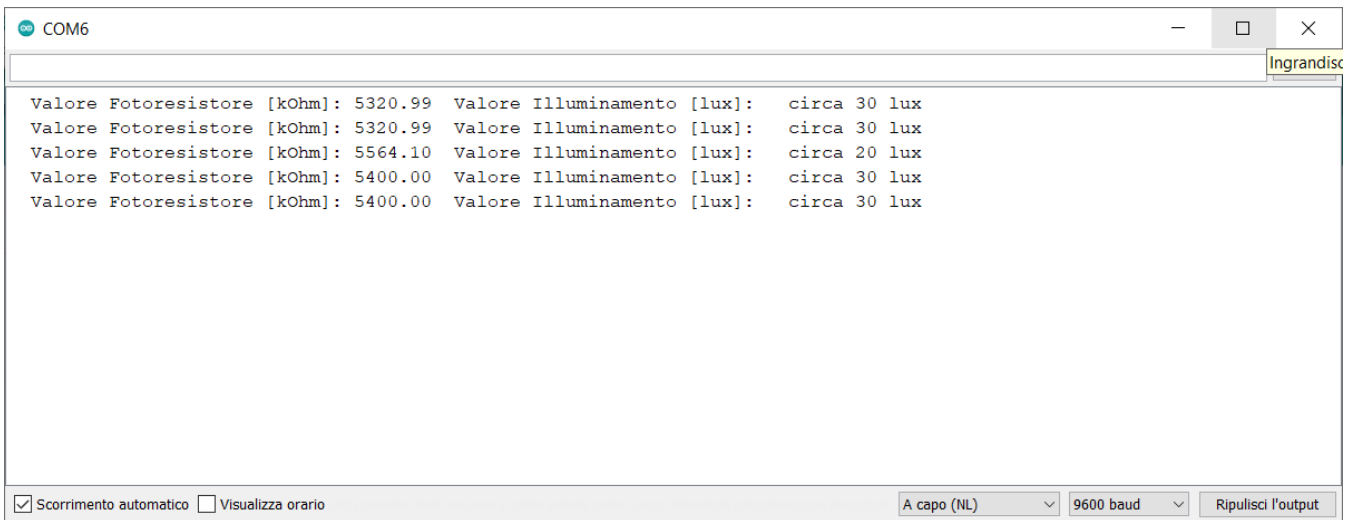
```

Once you have written the code launch the verification precompilation (check mark), it will ask you to save the sketch (you can change its name) and then click on the arrow to load and then on the lens to access the serial monitor.





Once the Serial Monitor is started, the result is this:



The screenshot shows the Arduino IDE Serial Monitor window for COM6. The window title is "COM6" and it has standard window controls. The output text is as follows:

```
Valore Fotorresistore [kOhm]: 5320.99 Valore Illuminamento [lux]: circa 30 lux
Valore Fotorresistore [kOhm]: 5320.99 Valore Illuminamento [lux]: circa 30 lux
Valore Fotorresistore [kOhm]: 5564.10 Valore Illuminamento [lux]: circa 20 lux
Valore Fotorresistore [kOhm]: 5400.00 Valore Illuminamento [lux]: circa 30 lux
Valore Fotorresistore [kOhm]: 5400.00 Valore Illuminamento [lux]: circa 30 lux
```

At the bottom of the window, there are checkboxes for "Scorrimento automatico" (checked) and "Visualizza orario" (unchecked). To the right, there are dropdown menus for "A capo (NL)" and "9600 baud", and a button labeled "Ripulisci l'output".

[For the Video-project click [here](#)]

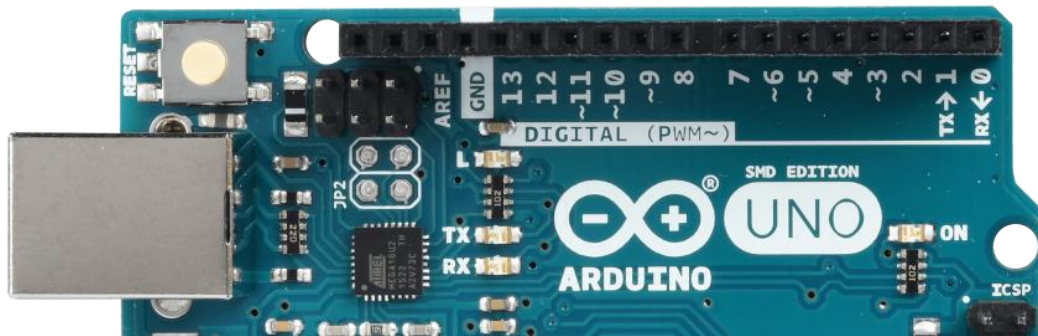
### Sketch Analysis: Luxmetro with Arduino

The peculiarity of the sketch of this project is to have created the correspondence ranges of the resistance values to the illuminance values and therefore as a function of the resistance value the most appropriate illuminance value is chosen, this was done using the nested ifs, that is, after the **else** which theoretically establishes the alternative to a condition and closes the verification (as happens to the sketch of the previous project), a new condition is immediately reinserted to be verified with a new "if" and this happens 6 times. As I said at the beginning, this programming methodology has only a didactic purpose, generally we try to avoid nested if and C++ also provides other solutions that we will see in other projects.

## The PWM signal - Pulse Width Modulation

We know that Arduino has 14 digital PINs, that is, they are able to receive input, or give an output signal with only two states, HIGH and LOW.

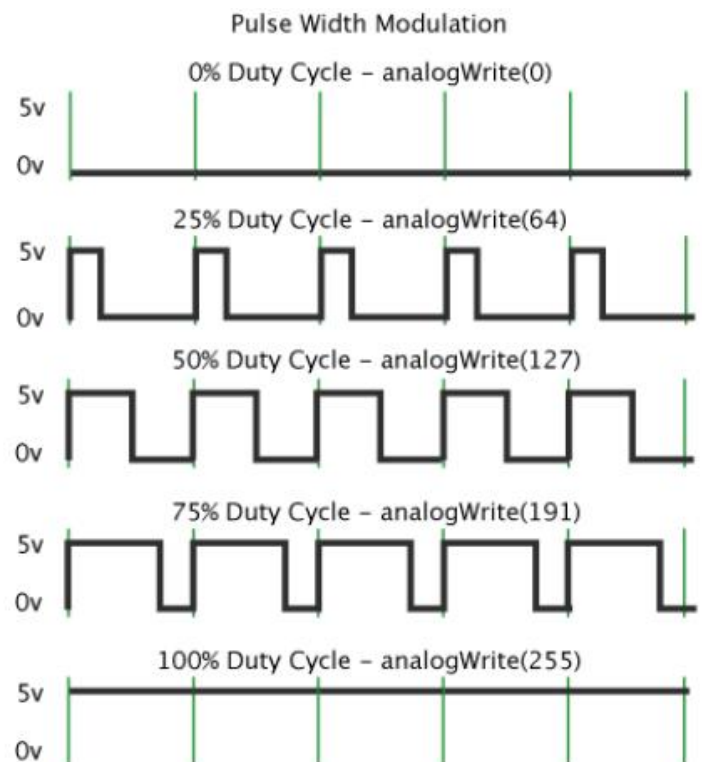
Digital PINs are indicated with the numbers che range from 0 to 13, but paying attention some of them also have in addition to the number a wave symbol "~". Those PINs are Digital PINs with the possibility of giving a PWM output voltage, in English **Pulse Width Modulation**, that is, an electric voltage with modulated pulse width. The Digital PINs with this functionality are: **3, 5, 6, 9, 10 and 11**.

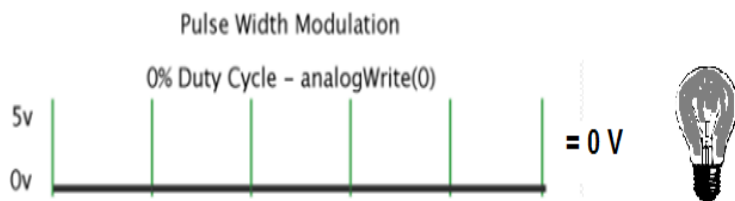


Let's see what we can do with these PINs with the ondina (in jargon it is called "tilde").

The PWM signal is a signal that is always based on the principle of on or off (so it maintains the characteristics of the two states HIGH and LOW), but if we consider a period of time (called: cycle) and in this period we go to change the duration of holding the high state, rather than the low one, and we repeat it for all cycles, the average value of the signal that we have at the PIN is no longer 0V or 5V, but will be a value between 0 and 5V.

The figure next to it explains this concept, but let's see it even better, with the next figure.





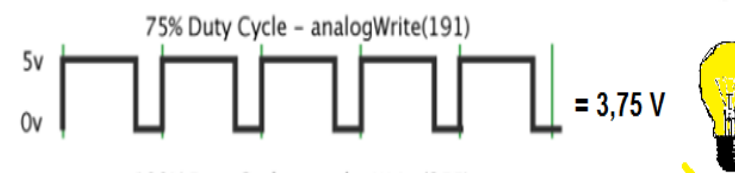
*In the first case we have that the value of 5 V is not maintained in the cycle time, so the signal is equal to 0% of the 5 V voltage, that is, the digital PIN has an average output value of 0 V.*



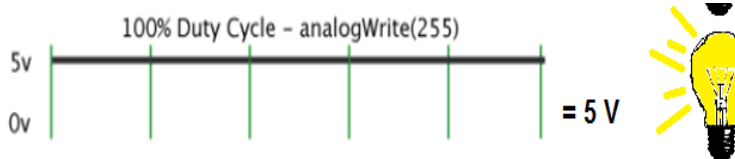
*In the second case, the 5 V are maintained for 25% of the cycle time, that is, 25% of 5V, so the voltage at the output digital PIN has an average value of 1.25 V.*



*In the third case, the 5V are maintained for 50% of the cycle time, that is, 50% of 5V, so the voltage at the output digital PIN has an average value of 2.5 V.*



*In the fourth case, the 5V are maintained for 75% of the cycle time, that is, 75% of 5V, so the voltage to the output digital PIN has an average value of 3.75 V.*



*In the fifth case, the 5V are maintained for 100% of the cycle time, that is, 100% of 5V, so the voltage at the output digital PIN has an average value of 5 V.*

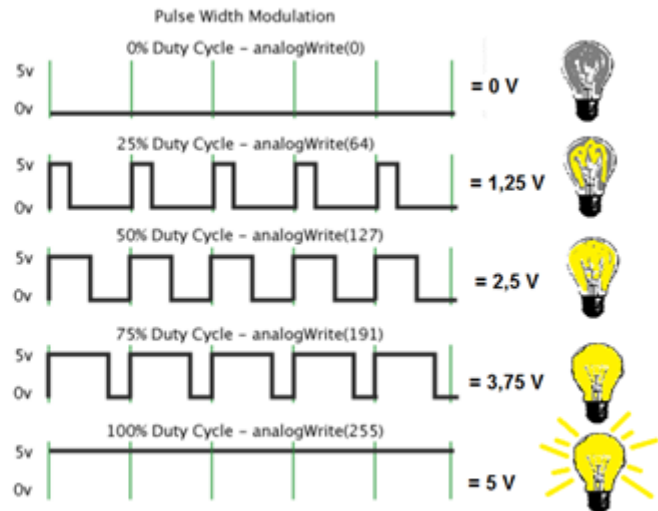
*To a PWM digital PIN we can give values between 0 and 255 (where 0 indicates 0 V and 255 indicates 5 V – basically 256 values).*

*The instruction, with its syntax, to attribute a PWM signal to a digital PIN is as follows:*

**analogWrite(numero\_PIN\_PWM, valore\_pwm);**

*So summarizing in a schematization what has been said above, and assuming to use the PIN n. 11 for a PWM type signal, we have:*

```
analogWrite ( 11 , 0);  
analogWrite ( 11 , 64);  
analogWrite ( 11 , 127);  
analogWrite ( 11 , 191);  
analogWrite ( 11 , 255);
```



Obviously we can also use the intermediate values and not only those indicated.

The next project shows in a practical way an interesting application of pwm signal to digital PINs. In fact, just think of a function of Arduino that allows us to adjust the brightness of an LED (but it could be anything else, for example the lights of our house, the speed of rotation of a motor, ....) acting on the PWM digital PIN, but I will say more, we can adjust this variation also to other external phenomena.

The following project links the measurement of external brightness detected by a photoresistor to the brightness that we want to give in an environment, but before going into the project I still want to highlight something that will be very useful in many applications.

---

### *Relationship between Analog PIN and PWM Digital PIN*

As already explained above, an Analog PIN assumes values between 0 and 1023 (where 0 indicates 0V and 1023 indicates 5V- essentially 1024 values), while a PWM digital PIN assumes values between 0 and 255 (where 0 indicates 0V and 255 indicates 5V – essentially 256 values). At this point, thanks to a simple proportion of our holy Mathematics, we are able to write a formula that defines the voltage of a PWM digital PIN as a function of the value detected by the analog PIN (for example: A0). In fact, the proportion is as follows:

$$256 : 1024 = X : \text{reading at PIN A0}$$

(it reads: 256 is to 1024 as the unknown value to be obtained and sent to the PWM digital PIN is to the reading to Pin A0)

When the unknown is in the proportional averages, we know that to derive it we multiply between the proportional extremes and divide by the term known among the proportional averages.

Therefore:

$$X = \frac{256 * \text{lettura}}{1024} = \frac{\text{lettura}}{4}$$

but  $256/1024 = 1/4$ , so to have the corresponding PWM signal value having an analog signal, just divide the latter value by 4.


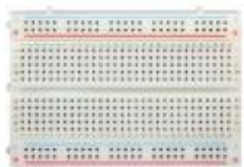








## Project 10 – Night and Day with PWM signal

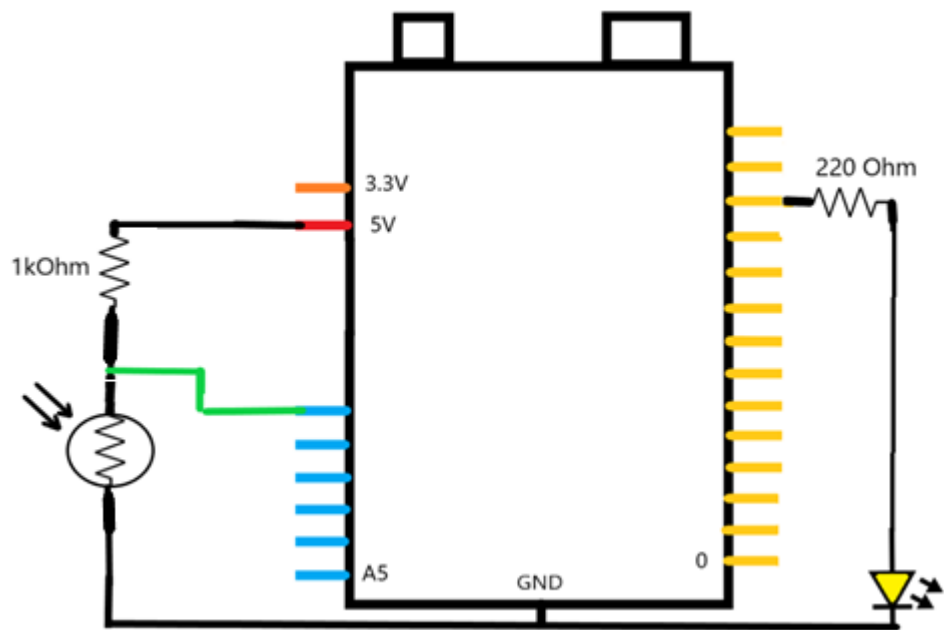


Suppose we want to keep the light in an always constant environment, so at sunset, gradually the artificial lighting goes to integrate the sunlight to always maintain the same level of brightness in the environment. Well the project that I present to you allows precisely this functionality.

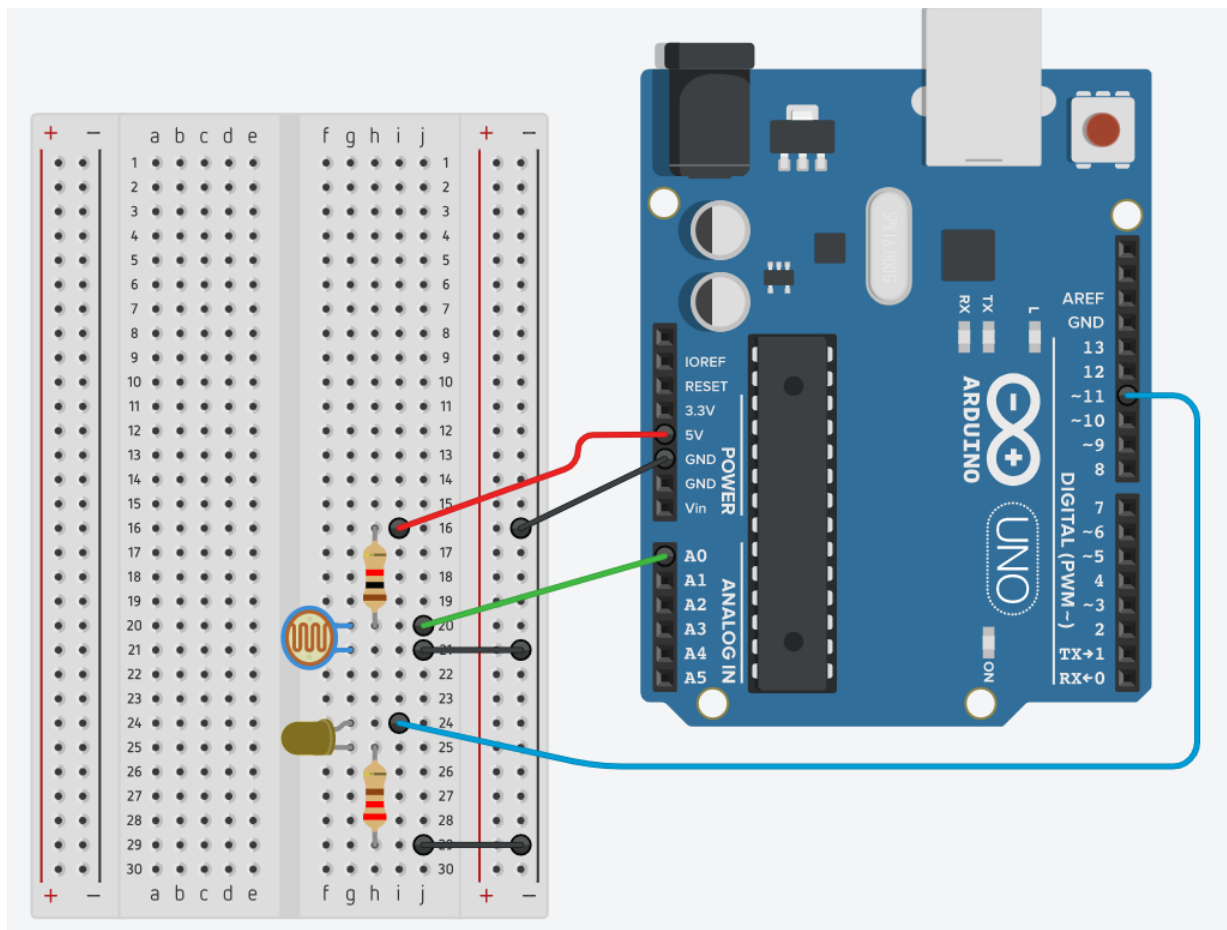
For this project we need:

|   |   |  |   |
|---|---|--|---|
|  |  |  |  |
| <i>Arduino Uno R3 or compatible</i>   | <i>Breadboard</i>   | <i>Resistance 1kOhm</i>  | <i>Dupont cables male - male</i>  |
|  |  |  |  |
| <i>USB connection cable</i>   | <i>LDR<br/>GL5528</i>   | <i>220 Ohm resistance</i>  | <i>LED</i>  |

The wiring diagram is as follows:



While below is the mounting scheme:



---

At this point we click twice on the Arduino icon on the Desktop and the IDE opens and we copy the following sketch, I recommend you follow each step indicated:

```
/*
Be Maker School - STEM - Project 10 – Night and Day
*/
void setup() {
  pinMode(11,OUTPUT); you set the PIN 11 as output
  Serial.begin(9600); you set the serial port
}

void loop() {
  int reading=analogRead(A0);  reads PIN A0
  int valuepwm=read/4;        converts the reading into the PWM signal
  float Vout=(read*5)/1024.0; determines the Vout based on
                             of the value read on pin A0
  float Rnota=1000.0;         value of the known Resistance i.e. 1000 Ohm
  current float=(5-Vout)/Rnota; calculates the current
  float ResIncognita=Vout/current; calculates the unknown resistance

  Serial.print("Photoresistor Resistance [Ohm]: ");
  Serial.print(ResIncognita);
  Serial.print(" Read value A0: ");
  Serial.print(read);
  Serial.print(" PWM value before map: ");
  Serial.print(pwm value);
  valuepwm = map(valuepwm, 200, 255, 0, 255); remappa value according to new scale
  Serial.print(" PWM value after map: ");
  Serial.println(pwm value);
  if (pwm value<0) {pwmvalue=0;}
  analogWrite(11,pwm value); puts the PIN 11 at a voltage varying between 0 and 5V
  delay(200);                waits 200 ms before starting again
}
```

Once you have written the code launch the verification precompilation (check mark), it will ask you to save the sketch (you can change its name) and then click on the arrow to load and then on the lens to access the serial monitor.

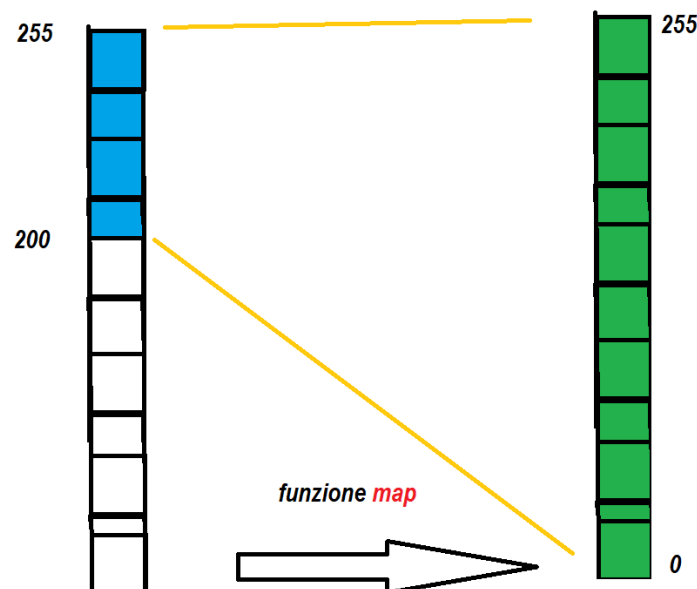
Started the sketch on Arduino, if everything has been done correctly, you have that by shadowing the photoresistance sensor (even with a pen cover), the brightness of the LED increases, while on the contrary, by getting more light to the sensor, the LED tends to turn off, all with a **FADE** effect that is with a gradual variation.

[For the video-project click [here](#)]

## Sketch Analysis: Night and Day with PWM Signal

In this sketch there are so many interesting things to analyze. Let's review them all starting from the main things:

- 1) You can see how you apply the PWM signal to the digital PIN, in fact the `analogWrite` statement is used for a PWM digital PIN instead of the classic `digitalWrite` statement. You can see it in the line where there is: `analogWrite(11,valuepwm);`
- 2) We have seen how to convert a value from an analog PIN into a corresponding value of a PWM Digital PIN, just divide the reading on the Analog PIN by 4, this can be seen on the line: `int valuepwm = reading / 4;`
- 3) Since at the time of execution of the project it was night and in my room at most I could get 50 lux and not more, you understand very well that the LED always started from a lighting condition. So I used a function called **map** that basically does nothing but calculate the proportions and then what was my possible range of PWM signal values (unable to perform the project in bright sunlight) ranging from a minimum of 200 to a maximum of 255, I remapped it in a new scale from 0 to 255.



The syntax for the map function is:

**map(value, fromLow, fromHigh, toLow, toHigh)**


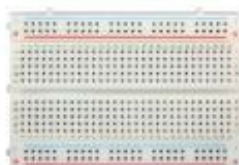




you see it applied to the sketch to the line: `valuepwm = map(valuepwm, 200, 255, 0, 255);` where the `pwm-value` variable is assigned the new remapped value of the value that the same variable had before. Since in the remapping it could happen to have negative values that are considered by the PWM digital PIN as a HIGH value, then I also had to enter in control with an if to the line: `if (valuepwm<0) {valuepwm=0;}`

## Project 11 – The Return of the Cylons



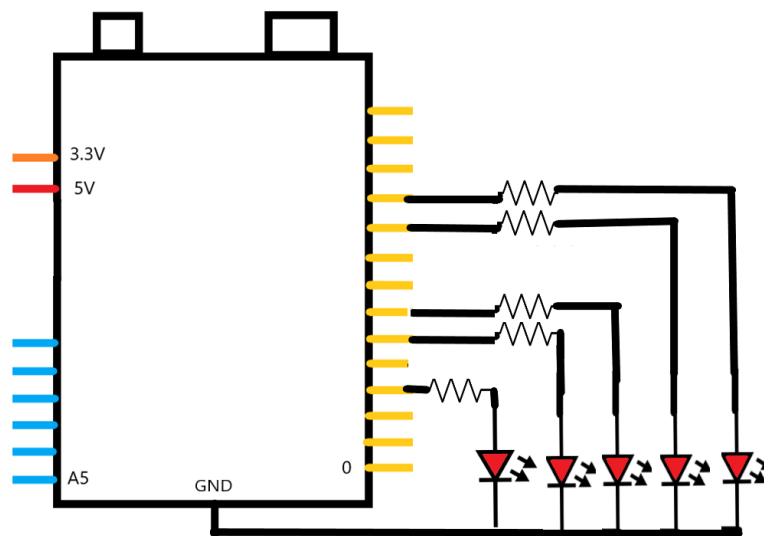
I've always been passionate about the Battlestar Galactica TV series and the thing that intrigued me was to know how they could give the movement of light that somehow had to represent the view of Cylon robots. Well! with the project that we are going to develop you can also make a Cylon head and why not, use it for carnival.

For this project we need:

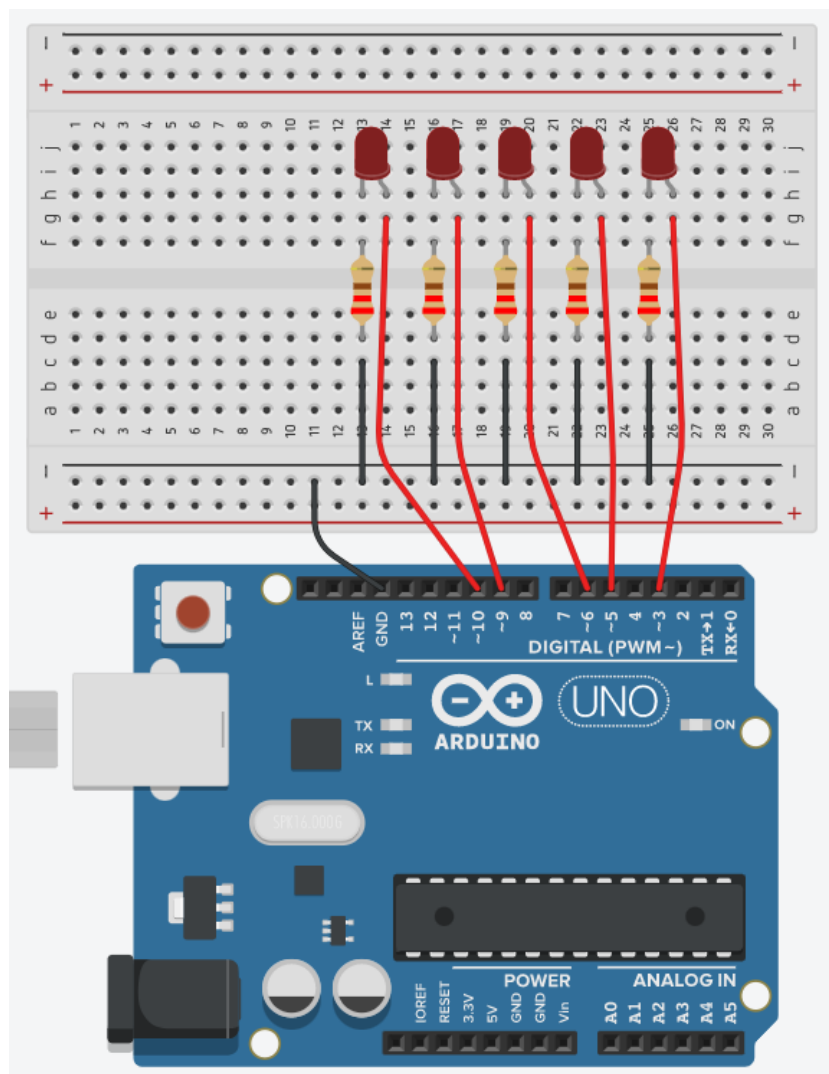
|   |   |  |   |
|---|---|--|---|
|  |  |  |  |
| <i>Arduino Uno R3 or compatible</i>   | <i>Breadboard</i>   | <i>USB connection cable</i>  | <i>Dupont cables male - male</i>  |
|  |  |  |   |
| <i>No. 5 Resistors of 220 Ohm</i>   | <i>N. 5 Red LEDs</i>  |  |   |



The wiring diagram is as follows:



The mounting diagram is the one below:



---

Now open a new window for sketching by clicking twice on the Arduino application and write the code below and save the file:

```
/*  
Be Maker School - STEM - Project 11 – Cylon  
*/  
int delay=100;  
int delay2=1;  
  
void setup()  
{  
  I configure the PWM PINs output  
  pinMode(5, OUTPUT);  
  pinMode(6, OUTPUT);  
  pinMode(9, OUTPUT);  
  pinMode(10, OUTPUT);  
}  
  
void loop()  
{  
  analogWrite(9, 255);  
  for cycle with the in degrowth to make  
  gradual turning off of the LED  
  for (int i = 255; i >= 0; i--) {  
    analogWrite(10, i);  
    delay(delay2);  
  }  
  delay;delay;  
  analogWrite(6, 255);  
  for (int i = 255; i >= 0; i--) {  
    analogWrite(9, i);  
    delay(delay2);  
  }  
  delay;delay;  
  analogWrite(5, 255);  
  for (int i = 255; i >= 0; i--) {  
    analogWrite(6, i);  
    delay(delay2);  
  }  
  delay;delay;  
  analogWrite(3, 255);  
  for (int i = 255; i >= 0; i--) {  
    analogWrite(5, i);  
    delay(delay2);  
  }  
  delay;delay;  
  analogWrite(5, 255);  
  for (int i = 255; i >= 0; i--) {  
    analogWrite(3, i);  
    delay(delay2);  
  }  
  delay;delay;  
}
```

---

```

analogWrite(6, 255);
for (int i = 255; i >= 0; i--) {
  analogWrite(5, i);
  delay(delay2);
}
delay;delay;
analogWrite(9, 255);
for (int i = 255; i >= 0; i--) {
  analogWrite(6, i);
  delay(delay2);
}
delay;delay;
analogWrite(10, 255);
for (int i = 255; i >= 0; i--) {
  analogWrite(9, i);
  delay(delay2);
}
delay;delay;
}

```

Important: the sketch above was written in that form to take into account the still little knowledge of the programming language for Arduino, but using arrays the code can be very simplified:

```

/*
Be Maker School - STEM - Project 11 – Cylon array version
*/

```

```

int delay1=100;
int delay2=1;
int myLED[] = {3,5,6,9,10};
int count=0;

void setup()
{
  I configure the PWM PIN output with for loop
  for (count=0; count<6; count++) {
    pinMode(myLED[count], OUTPUT);
  }
}

void loop()
{
  for (count=0; count<5; count++) {
    analogWrite(myLED[count], 255);
    if (count>0) {
      for (int i = 255; i >= 0; i--) {
        analogWrite(myLED[count-1], i);
        delay(delay2);
      }
    }
  }
  delay(delay1);
}

```

---

```

}
for (count=4; count>0; count--) {
  if (count>0) {
    analogWrite(myLED[count-1], 255);
  }
  for (int i = 255; i >= 0; i--) {
    analogWrite(myLED[count], i);
    delay(delay2);
  }
  delay(delay1);
}
}
}

```

Load the sketch and you will get a nice effect of robotic eye scrutinizing, especially if you cut out the cardboard and create the slot in which to insert the LEDs.... The famous KITT Supercar car also had a similar effect...

[For the video-project click [here](#)]

### *Sketch Analysis: The Return of the Cylons*

Not considering the second sketch transcribed because the topics covered will be seen later, from the first sketch we can observe that the only new topic introduced is the for cycle.

The for cycle is used when we need, within the body of the program, to perform repetitive operations for a certain number of times. In fact, in this case it was used to drive the switching off of the LED gradually using the variable i that starting from the maximum value of 255, at each cycle decreased and the light intensity of the LED consequently also decreased.

The syntax for the for loop is:

```

for (initialization; condition; increment) {
  statement(s);
}

```

In our sketch it was using for example:

```

for (int i = 255; i >= 0; i--) {
  analogWrite(10, i);
  delay(delay2);
}

```

The integer variable "i" starts from the value of 255 and decreases to zero; as long as this condition is true, the for cycle is repeated and in particular performs what is between the braces, or goes to write in the PWM digital PIN a value equal to the variable i (which is decreasing) and then goes to gradually turn off the LED connected to PIN 10.

---

(print and crop)

